UN ENFOQUE ORIENTADO A USUARIOS FINALES PARA EL DESARROLLO DE APLICACIONES DE AUMENTACIÓN WEB MÓVIL

GABRIELA A. BOSETTI

Director: Dr. Gustavo Rossi

Director: Dr. Sergio Firmenich

Codirector: Dr. Marco Winckler

Presentado en cumplimiento de los requisitos para la obtención del grado de

DOCTOR EN CIENCIAS INFORMÁTICAS

Facultad de Informática

Universidad Nacional de La Plata

Septiembre, 2017

Agradecimientos

A mis directores, Gustavo y Sergio, quienes me enseñaron, acompañaron y ayudaron a construir las alas que hoy me dejan ser y hacer lo que más me gusta.

A Silvia y Cecilia, quienes me ayudaron a dar mis primeros pasos en dirección al camino que hoy transito y a construir el conocimiento que me hacía falta para comenzar.

A Marco, quien me ayudó a comprender diferentes formas de interacción y validación cuando se trata de enfoques con usuarios finales.

A quienes construyeron y sostienen día a día el espacio en el cual me desempeño; el LIFIA. Este logro no hubiese sido posible si el contexto no lo hubiese permitido. Gracias por recibirme y hacerme parte.

A todos aquellos que eligen y dan sustento a un modelo de educación pública, gratuita y de calidad. Es difícil imaginarme en esta misma posición si nuestra sociedad no hubiese garantizado este derecho, incluso desde la más temprana de las etapas.

A la gestión de gobierno que impulsó fuertemente a la ciencia argentina durante una década coincidente con el inicio de mi carrera, y que me ofreció la oportunidad de tener mi primer contacto con la investigación.

A mis compañeros de laboratorio, que llenaron mi día a día, compartiendo y creciendo juntos.

A mis compañeros de doctorado, con quienes aún hoy comparto risas y desafíos de una misma etapa.

A todos los que, de una u otra manera, dedicaron parte de su tiempo en mi formación como profesional e integrante de esta sociedad.

A mi familia y amigos, por aceptar mis sueños y consecuentemente, muchas veces, mis ausencias.

Gabriela

Resumen

La información existente en la Web podría utilizarse para resolver una amplia gama de problemas de diferentes dominios, pero dicha información no siempre se encuentra reunida en un mismo contexto y resulta difícil encontrar una manera de relacionarla para satisfacer necesidades que, a veces, son requeridas por una minoría de usuarios. Los sitios Web son diseñados para un único usuario, sino más bien enfocados en un conjunto de ellos con características similares y para llevar a cabo tareas específicas, que pueden diferir de las que un usuario particular necesita. Aquí es donde la Aumentación Web se presenta como un medio para lograr la adaptación de los recursos Web existentes de acuerdo con los requisitos del usuario, tanto en contenido como en comportamiento, estructura y/o forma.

Por su parte, la tendencia hacia el uso activo de los dispositivos móviles ha hecho posible que la Web sea concebida no sólo como un espacio de información, sino también como una plataforma omnipresente donde sus usuarios realizan todo tipo de tareas. Sin embargo, pese a su uso, muchas aplicaciones Web aún no tienen una contraparte móvil que haga uso de sus características, e incluso algunas ni siquiera responden a un diseño Web *responsive*.

La información del contexto permite mejorar la experiencia del usuario. Por ejemplo, permite filtrar y solo presentar aquella información que coincide con la posición del usuario, la presión sonora en el ambiente o un rango horario determinado. La Aumentación Web Móvil puede ayudar a agregar incorporar características, y mediante técnicas de Desarrollo por Usuarios Finales es posible convertir a las personas que poseen tal necesidad, en los autores de su propia solución de aumentación.

Aunque algunos enfoques existen para aumentar aplicaciones Web, incluso con características móviles, los mismos son dirigidos y limitados a personas con –al menos– conocimiento en programación. También existe un gran número de herramientas de autoría que permiten la creación de aplicaciones móviles desde entornos de escritorio, nativos o móviles, pero ninguna de ellas genera *aplicaciones Web puras*; ejecutables en un navegador Web móvil convencional y sin depender de algún componente nativo para su ejecución.

En esta obra se presenta un nuevo enfoque para permitir que los usuarios finales mejoren sus sitios Web preferidos con características móviles. El proceso de autoría es asistido por formularios, y un paso final que combina live programming y composición mediante widgets visuales. El enfoque fue evaluado mediante el uso de una herramienta de soporte, por 21 usuarios finales, y sus resultados demuestran que el enfoque es viable y prometedor, puesto que los usuarios finales fueron capaces de completar, en promedio, el 84% de los requerimientos del experimento.

Palabras clave. Aumentación Web, Desarrollo por Usuarios Finales, Web Móvil.

Abstract

Information on the Web could be used to solve a broad range of problems from different domains, but this information is not always in a same context and it is hard to find a way to combine it to satisfy a particular need. Web sites are not designed for a single user, but rather focused on a set of users with similar characteristics and to carry out specific tasks, which may differ from those that a particular user needs. Here is where Web Augmentation comes into play as a means to achieve the adaptation of the existing Web resources according to the requirements of the user, concerning its content as well as its behavior, structure or style.

In turn, the trend towards the active use of mobile devices has made it possible for the Web to be conceived not only as a space of information but as an omnipresent platform where its users perform a myriad of tasks. However, despite its use, many Web applications still do not have a mobile counterpart making use of its features, and even some do not even have a responsive Web design.

Context information allows enhancing the user experience. For example, it is possible to filter and only present information according to the current position of the user, the sound pressure in the environment or the current time in relation to certain time range. Mobile Web Augmentation is a means to add such features, and End User Development techniques allow to turn people who have such a need into the authors of their own augmentation solutions.

Although some approaches exist to augment Web applications –even with mobile features– they are aimed at and limited to people with –at least– some programming knowledge. There is also a large number of authoring tools allowing the creation of mobile applications from desktop, native or mobile environments, but none of them generate pure Web applications; the ones executable in a conventional mobile Web browser and without depending on some native component.

In this work, a new approach is presented allowing end users to improve their favorite Web sites with mobile features. Forms assist the authoring process, and a final step combines live programming and visual widgets composition. The approach was evaluated through the usage of a support tool by 21 end users. The results show that the approach is feasible and promising, since end users were able to complete, on average, 84% of the experiment requirements.

Keywords. Web Augmentation, End User Development, Mobile Web.

Tabla de contenidos

Lista de Tablas

Lista de Figuras

Acrónimos

- **API.** Interfaz de Programación de Aplicaciones, del inglés «Application Programming Interface».
- **dB**. Decibel.
- **DOM**. Modelo de Objetos del Documento, del inglés «Document Object Model».
- **DSL**. Lenguaje Específico de Dominio, del inglés «Domain Specific Language».
- **EUD**. Desarrollo por Usuarios Finales, del inglés «End-User Development».
- **EUP**. Programación por Usuarios Finales, del inglés «End-User Programming».
- **PBD**. Programación Por Demostración, del inglés «Programming by Demonstration».
- **PBE**. Programación Por Ejemplificación, del inglés «Programming by Example».
- **UI**. Interfaz de usuario, del inglés «User Interface».
- **WA**. Aumentación Web, del inglés «Web Augmentation».
- **CA**. Sensibilidad al Contexto, del inglés «Context Awareness».
- **ITU**. Unión de Telecomunicaciones, del inglés «International Telecommunication Union».

Glosario

- **Adaptación Web**. Capacidad de las aplicaciones Web de ser resiliente a los cambios en las necesidades del usuario.
- **Add-on script**. Script que forma parte del núcleo de una extensión de Firefox y, en consecuencia, tiene mayores privilegios que el código de una aplicación Web tradicional.
- **Aplicación Web pura**. Aplicación Web cuya ejecución no depende más que de un navegador Web convencional, es decir, que no requiere de otro componente nativo o híbrido para su correcto funcionamiento.
- **Aumentación Web**. Conjunto de técnicas que permite manipular aplicaciones Web existentes de acuerdo con los requisitos del usuario.
- **Augmenter**. Unidad básica de aumentación.
- **Builder**. Artefacto que hace posible la instanciación de componentes específicos de una aplicación MoWA, solicitando la información necesaria mediante una interfaz visual a los usuarios finales.
- **Carrousel**. Componente de interfaz visual que presenta una lista de elementos que se desplazan, a menudo, en forma circular.
- **Crowdsourcing**. Actividad online participativa en la cual una entidad propone, mediante una convocatoria abierta, la realización voluntaria de una tarea a cambio de alguna forma de recompensa.
- **Crowdsourcer**. Persona que propone una tarea en una plataforma de Crowdsourcing.
- **Crowdworker**. Persona que propone una solución en respuesta a una solicitud de tarea dentro de una plataforma de Crowdsourcing.
- **Content-script**. Script que es inyectado, mediante un addon-script, en el contexto de una página Web y que tiene los permisos necesarios para manipular su DOM.
- **Contexto**. Cualquier información que sirva para caracterizar la situación de una entidad.
- **Decibel**. En este trabajo, unidad de medida utilizada para expresar el nivel de intensidad del sonido.
- **Demo session**. Sesión de demostraciones en el marco de una conferencia, en la cual los participantes tienen la oportunidad de demostrar, en vivo, la capacidad de sus sistemas propuestos.
- **div**. Elemento divisor HTML que sirve de contenedor genérico para agrupar otros elementos.

- **Framework**. Conjunto de clases y conceptos que cooperan y conforman un diseño reutilizable para un tipo específico de aplicación.
- **From scratch**. Expresión inglesa que significa «desde cero».
- **Full screen**. Expresión inglesa que significa «pantalla completa».
- **Hipermedia Móvil**. Es la extensión del concepto de navegación digital, utilizado en la Hipermedia tradicional, al mundo real. Este tipo de aplicaciones comprende funcionalidad de movilidad y aspectos de navegación digital y física.
- **In-situ**. Expresión latina que significa «en el lugar», para referirse a un fenómeno que ocurre en ese sitio o contexto determinado.
- **Landscape**. Es un modo de presentación de los elementos de la interfaz gráfica de acuerdo con la orientación del dispositivo; en este caso, cuando el mismo se encuentra en posición «horizontal».
- **Layout**. Disposición de los elementos en una página.
- **Listener**. Se refiere al rol «observer», dentro del patrón de nombre homónimo.
- **Live programming**. Práctica de programar realizando cambios en la aplicación mientras se encuentra en ejecución.
- **Locative media**. Contenido multimedia naturalmente geo-posicionado. Las aplicaciones que proveen una experiencia de este tipo proporcionan información al usuario dependiendo de su ubicación.
- **Logging**. Registro de todos eventos o acciones que ocurren dentro de un sistema.
- **Lux**. Unidad de medida para el nivel de iluminación.
- **Mirror**. Réplica exacta de otro sitio Web.
- **Namespace**. Término inglés que significa «espacio de nombres». Es un conjunto de identificadores únicos que permiten organizar artefactos de diferente naturaleza, por ejemplo, clases extendidas por una organización.
- **On-demand**. Expresión inglesa que significa «bajo demanda».
- **On-the-fly**. Expresión inglesa que significa «en el momento».
- **Portrait**. Es un modo de presentación de los elementos de la interfaz gráfica de acuerdo con la orientación del dispositivo; en este caso, cuando el mismo se encuentra en posición «vertical».
- **Proxy**. Servidor que actúa como intermediario entre clientes y otros servidores, consumiendo información de estos últimos en respuesta a las peticiones de los clientes.
- **Release**. Término inglés que significa «lanzamiento».
- **Responsive**. Diseño Web que intenta adaptar la presentación del contenido de las

páginas en respuesta a las dimensiones disponibles (ej. las del dispositivo o del navegador redimensionado).

- **Restartless**. Término inglés que significa «sin reinicio». En esta obra, hace referencia a extensiones cuya instalación no requiere el reinicio del navegador.
- **Script**. Conjunto de instrucciones, generalmente interpretadas, que permite la automatización de una tarea dentro de un entorno de ejecución determinado (ej. una página Web, un navegador, un sistema operativo, etc.).
- **Sensibilidad al Contexto**. Característica de las aplicaciones que les permite capturar información del contexto y reaccionar, adaptándose en consecuencia.
- **Tipo de contexto**. Descriptor o unidad de medida de una propiedad del contexto. Por ejemplo, la posición, la orientación, decibeles, luxes, un perfil de usuario, el nivel de carga de la batería, etc.
- **Valor del Contexto**. Valor concreto de un *tipo de contexto*. Por ejemplo, el valor de un «Lux» puede ser «5».
- **Walking Link**. Link «caminable», que asiste digitalmente en la navegación física entre dos puntos.
- **Weaver**. Herramienta que permite la ejecución de scripts sobre páginas Web existentes.
- **Wrapper**. Término inglés que significa «envoltorio», y hace referencia a una clase que envuelve (está asociada) a otra, ya sea para adaptar su interfaz o para agregarle responsabilidades. En la literatura, «wrapper» suele ser utilizado como sinónimo de «Adapter» o «Decorator».
- **WYSIWYG**. Acrónimo de «What You See Is What You Get», expresión inglesa que significa «Lo que ves es lo que obtenés». Hace referencia a aquellos sistemas en los que su contenido puede ser visualizado durante su edición en una forma muy similar a su apariencia como producto final.
- **XPath**. Expresión que permite localizar un elemento dentro de un documento XML.

1. Introducción

De acuerdo al reporte correspondiente al año 2016 [19] de la International Telecommunication Union (ITU), aproximadamente una de cada dos personas utiliza Internet. El 95% de la población mundial vive en un área cubierta por redes móviles, y las mismas son utilizadas por el 84% de la población (excluyendo zonas rurales). En el año 2015, el mismo autor ya destacaba a la banda ancha móvil como el segmento de mercado más dinámico, con un aumento del 47%, aumentado doce veces desde el año 2007 [86].

Tanto el crecimiento tanto de la Web como de la tecnología móvil y de las comunicaciones generó nuevas formas de participación de los usuarios en la Web. La funcionalidad de los primeros dispositivos móviles, en sus inicios, era limitada. Tal como se presenta en [56], la primera generación de móviles era soportada por tecnología analógica y solo estaba destinada a servicios de voz. Si bien algunos dispositivos soportaban roamming, no era posible operar entre las redes de diferentes países. A finales de los ‘80, surge la segunda generación de móviles que, soportada por tecnología digital, ofrece no solo servicios voz sino también mensajería de texto (de longitud corta) y transmisión de datos. Esta generación ofreció mejoras en la interoperabilidad entre redes, por ejemplo, GSM se desplegó en Europa para proporcionar un solo estándar unificado entre varios países. Posteriormente, la ITU especificó el estándar IMT-2000, con todos los requerimientos para la tercera generación, que se destacaba por tasas de datos a más altas, como 144 kbps sobre entornos de movilidad rápida (medios de transportes), 384 kbps a velocidad pedestre y 2 Mbps en interiores. Esto terminó por convertir a las plataformas móviles en dispositivos ideales para servicios multimedia. En 2008, la ITU publica los requerimientos de la cuarta generación bajo el nombre IMT-Advanced y, entre sus requerimientos, se destaca la alta velocidad de transmisión de datos. Se esperaban al menos 100 Mbit/s para comunicaciones de alta movilidad (como usuarios viajando en trenes, autos) y 1 Gbit/s para baja movilidad.

Actualmente, la posibilidad de hacer uso de una conexión de alta velocidad favorece el desarrollo y la utilización de aplicaciones móviles que requieren consumo de datos, tanto nativas como Web. Los dispositivos móviles hoy pueden ser utilizados como computadoras personales e incluso en su reemplazo, ya que una innumerable lista de tareas de la vida cotidiana se puede realizar haciendo uso de este soporte. Por ejemplo, enviar un email, hacer una transferencia bancaria, comprar pasajes de tren, hacer el check-in de un vuelo, ordenar productos al supermercado, postear entradas en un blog, editar documentos en línea, asistir a un curso, etc.

El uso de aplicaciones móviles crece día a día. Por ejemplo, la compañía Comscore reporta un estudio [28] realizado en Estados Unidos, durante el período 2013-2016, en

el que se puede observar que el tiempo de uso de los medios digitales ha crecido un 53%, especialmente en el segmento de las aplicaciones móviles. El reporte discrimina el uso desde aplicaciones móviles, Web móviles y de escritorio. Y aunque estas cifras han comenzado a mantenerse más estables durante el último periodo, el uso de aplicaciones móviles sigue creciendo: el uso de las aplicaciones nativas lo ha hecho en un 11% y las Web móviles un 5%, mientras que el uso desde entornos de escritorio ha disminuido en un 11%. Centrando la atención en entornos móviles, si bien los usuarios pasan más tiempo utilizando aplicaciones nativas que Web móviles, estas últimas tienen una audiencia casi tres veces más grande y que crece dos veces más rápido que la de las aplicaciones nativas.

Sin embargo, existen aplicaciones reconocidas a nivel mundial que no cuentan con una contraparte móvil adaptada a dispositivos móviles. Ejemplo de ello es Blogger; el mismo se encuentra en el puesto 113 de Alexa a nivel global[1] y, tal como se puede apreciar en la Figura 1, no ofrece al menos un diseño Web *responsive*. Del lado izquierdo de la figura puede observarse la administración de las entradas del blog, mientras que del lado derecho se observa la edición de una de las entradas del blog. En ambos casos, la interacción con los componentes de la interfaz puede resultar difícil sin llevar a cabo interacciones extras, como la de hacer zoom.

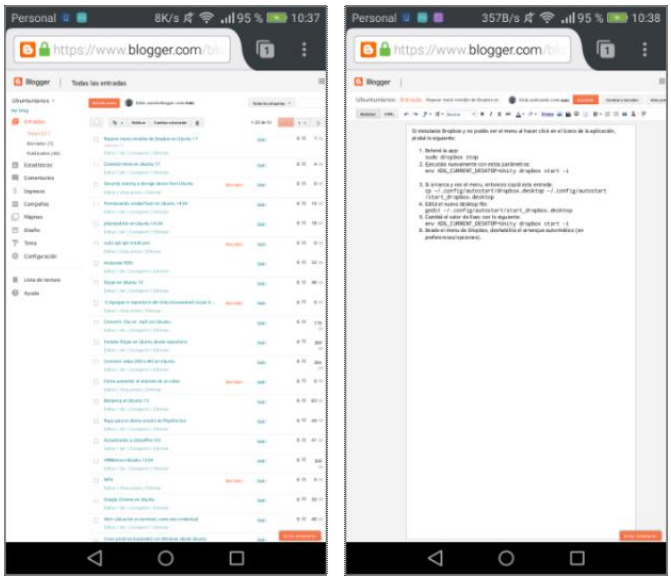

Figura 1. Aplicación Web sin una versión responsiva

Las aplicaciones móviles son tendencia, y la cantidad de funcionalidades innovadoras crece a medida que los recursos de los dispositivos incrementan. La capacidad de procesamiento y de almacenamiento, sumado a las dimensiones actuales de pantallas –más amplias y de mayor resolución–, hacen posible concebir hoy una gran

[1] Ver «Anexo A: Análisis de sitios Web populares»

gama de aplicaciones que cubren diferentes dominios. Esto también hace factible concebir la Web no solo como un espacio de información, sino como una plataforma ubicua en la que sus usuarios llevan a cabo todo tipo de tareas.

En algunos casos, los usuarios acceden la Web mediante aplicaciones móviles nativas desarrolladas para sitios Web populares. Por ejemplo, tomando los 10 primeros sitios listados en el top 500 de Alexa[2], sin repetir aquellos que bajo el mismo nombre utilizan un dominio de nivel superior diferente (ej. Google.com y Google.co.in), los sitios resultantes son: Google, YouTube, Facebook, Baidu, Yahoo!, Wikipedia, Amazon, Qq, Live y Taobao. Con tan solo tomar una plataforma de distribución digital como Google Play, es posible verificar que existe una aplicación nativa como contraparte al sitio Web[3], aunque también una versión Web móvil. Estas aplicaciones, las más populares a nivel mundial en este ranking, han tenido los recursos para cubrir ambas alternativas, pero no significa que absolutamente todos los propietarios de aplicaciones Web existentes la tengan o las consideren.

El avance tecnológico hoy da lugar a la decisión de qué tipo de aplicación crear para dar soporte a los usuarios móviles. Aunque hoy el escenario es mucho más parejo respecto al abanico de características técnicas que se puede cubrir con ambos tipos de aplicaciones, existen estudios que demuestran la tendencia de los usuarios a mantener su fidelidad al uso de aplicaciones nativas que a las Web móviles [28]. Pese a esto, muchas aplicaciones aún no cuentan con una contraparte nativa, y sus usuarios deben acceder a ellas utilizando un navegador Web. Por ejemplo, si del ranking de sitios top de Argentina de Alexa[4] se toman aquellos sitios exclusivamente dedicados a la publicación de noticias, sin contar blogs ni diarios de historias, se puede observar que sólo 36 de 52 sitios tienen una contraparte nativa. Si bien esta cifra representa el 69,2% de los sitios, se debe tener en cuenta que puede existir una relación con el nivel de tráfico de estos. Tan solo 2 de los 53 sitios no poseen una versión *full-responsive*, sin embargo, para uno de ellos ésta representa la única forma de acceso a la aplicación, pues no posee una contraparte nativa. Se debe resaltar que este sitio, el cual se muestra en Figura 2, no es poco utilizado, sino que fue elegido de entre los más populares del ranking de Alexa. Del lado izquierdo de la figura («a») puede observarse la página de inicio del sitio, con una lista principal de noticias pertenecientes a distintas categorías. Si bien las páginas del sitio son cargadas con un nivel de zoom que posibilita su lectura, tal como se muestra en la segunda captura («b»), no es posible una lectura fluida; requiere que el usuario se deslice constantemente por la página para poder visualizar

[2] http://www.alexa.com/topsites/global;0 consultado el 13/01/2017 a las 10:05 am.
[3] Ver «Anexo A: Análisis de sitios Web populares»
[4] http://www.alexa.com/topsites/countries%3B0/AR consultado el 13/01/2017 a las 09:50 am.

todo su contenido, y una vista completa de la página resulta en un tamaño de fuente difícil de leer («c»).

En sus inicios, las aplicaciones nativas tuvieron ciertos privilegios respecto al acceso a la información del contexto sobre las aplicaciones Web, debido a que sus frameworks de desarrollo fueron concebidos teniendo a las características móviles en mente. Por ejemplo, Phonegap[5] posibilitó, desde un enfoque híbrido, el acceso a la API para obtener datos del contexto mediante los sensores del teléfono.

Sin embargo, el continuo avance de los navegadores móviles y la aplicación de estándares Web han permitido que las aplicaciones tengan acceso a un conjunto cada vez más amplio de características que antes solo estaban al alcance del código nativo. Las aplicaciones Web hoy pueden acceder a información del contexto fácilmente, aunque no todas se adapten a ella. El incremento en el uso de dispositivos móviles hace de esta una característica deseable para las aplicaciones Web, ya que permite fácilmente mejorar la experiencia del usuario. No es difícil imaginar un escenario en el que, eventualmente, los usuarios requieran agregar funcionalidad móvil en sitios Web existentes.

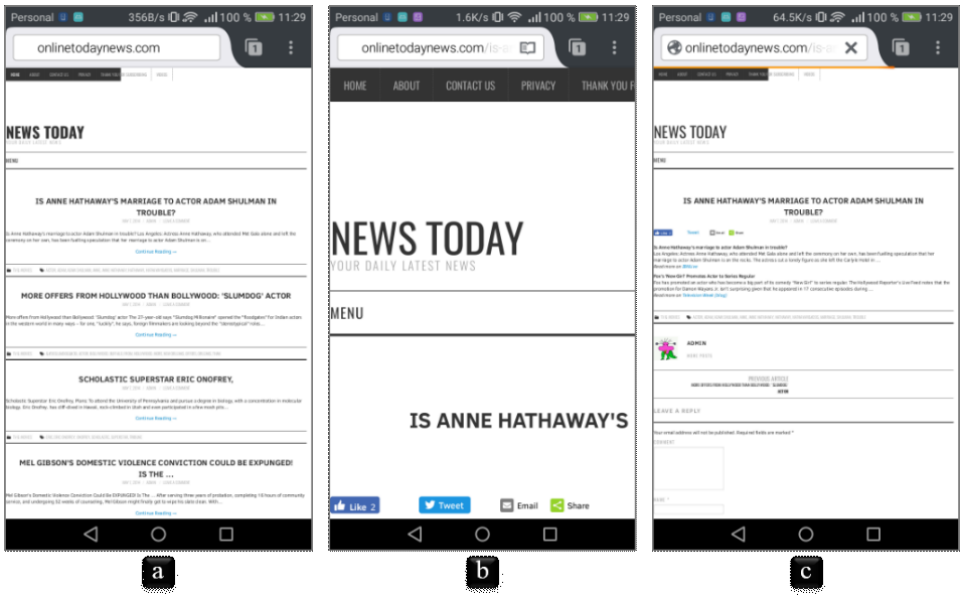

Figura 2. Sitio Web popular sin versión Web móvil-responsiva ni contraparte nativa

Muchas aplicaciones, como Booking, TripAdvisor, Foursquare o Yelp, proveen funcionalidad basada en la ubicación del usuario, sacando partido a la naturaleza móvil de los dispositivos donde se ejecutan. Sin embargo, muchos sitios son aún accedidos desde navegadores Web y no siempre proveen características móviles. Por ejemplo, se puede observar que los resultados de búsqueda de Google no varían dependiendo de si

[5] https://www.phonegap.com/ consultado el 22/06/2017 a las 5:57 pm

el contexto en el que se encuentra el usuario es ruidoso o no. Y una posible ventaja de ello sería dar prioridad a aquellos resultados con contenido multimedia cuando el usuario está en un lugar silencioso, y al contenido textual cuando el entorno esté altamente contaminado por ruido.

Los usuarios pueden necesitar satisfacer requisitos particulares de su contexto de uso [94], por lo tanto, proveer al usuario con aplicaciones Web para dispositivos estacionarios no es suficiente. Brindar este soporte implica el desarrollo de una aplicación nativa o de un diseño adaptable a diferentes dispositivos (*adaptive* o responsive Web design) [57], que además pueda acceder a la información del contexto y adaptarse en consecuencia a ella. El no hacerlo puede deberse a falta de recursos o de interés, dificultad para modificar el código de la aplicación existente, etc. En cualquier caso, la situación para los usuarios finales de estas aplicaciones es la misma: se encuentran carentes de la posibilidad de acceder en forma eficiente a los recursos y servicios ofrecidos.

Esta situación empeora cuando la información provista por las aplicaciones Web es naturalmente geo-posicionada, vinculada a una contraparte física, como la información presentada en museos o guías turísticas. Acceder a esta información *in-situ* permite enriquecer la visita, proporcionando al usuario final experiencias de locative media, como se ha demostrado en docenas de casos [27][90].

Por ejemplo, algunas organizaciones proveen esta información soportando características móviles mediante códigos QR. De esta manera, consiguen asociar puntos de interés en el mundo real (como piezas de una colección, monumentos, etc.) a una contraparte digital, que el usuario puede explorar con el uso de dispositivos móviles. Si se tiene una colección de piezas de arte y un conjunto de artículos en Wikipedia que los describen, se pueden generar fácilmente códigos mediante QRpedia[6]. Los mismos pueden imprimirse y colocarse junto a cada pieza, de modo que cualquier usuario pueda escanearlos con su dispositivo móvil y obtener información sobre el objeto de interés, en el lugar donde se encuentra físicamente posicionado. Aunque la interacción con códigos QR facilita el acceso a una contraparte digital ligada a un objeto del mundo real, Wikipedia no ofrece características que permitan sacarle provecho a dicha asociación, como podría serlo la asistencia para el recorrido físico de las piezas. Una alternativa para construir una solución de este tipo, que además permita reutilizar el contenido ya existente en la Web, es mediante Aumentación Web (WA) [35][17].

Una forma de poner en práctica la WA es mediante técnicas de adaptación del lado

[6] http://qrpedia.org/

del cliente, manipulando la interfaz de usuario (UI) cuando una página ya ha sido cargada en el navegador, tras lo cual pueden realizarse cambios en su contenido, estilo, estructura y/o funcionalidad sin necesidad de alterar el código fuente original de la aplicación en el servidor.

Durante los últimos años, muchos usuarios comenzaron a agregar funcionalidad extra a las aplicaciones Web existentes para soportar, extraoficialmente, requerimientos que no fueron contemplados originalmente por sus propietarios [36]. El surgimiento de grandes comunidades de scripters y de desarrolladores de extensiones de navegador[7] demuestran esta tendencia. Por ejemplo, userscripts.com fue un repositorio que registró un gran volumen de scripts por año. El sitio pasó a estar fuera de servicio en 2014, pero como consecuencia se creó un *mirror* que se mantiene en línea hasta la fecha; el mismo posee 130.956 scripts y registra un total de 1.203.605.530[8]. Otros dos repositorios similares surgieron el mismo año en que userscripts.com pasó a estar fuera de línea y aún permanecen activos. Uno de ellos es GreasyFork[9], con 13.603 scripts y 1.259.866 instalaciones, y otro es OpenUserJS10, con 3.919 scripts y 13.305.857.

Tanto la creación como el consumo de scripts que adaptan la Web creció rápido y lo sigue haciendo actualmente, sosteniendo esta tendencia. Por ejemplo, analizando la creación de scripts en Greasyfork, se puede observar que 4.224 scripts fueron creados en 2014, 4276 en 2015 y 5103 en 2016. De modo que actualmente la cantidad de scripts que registra el repositorio supera por más de tres veces la cantidad registrada en su primer año (2014). Además, en importante destacar que estos scripts no permanecen intactos en el tiempo; en la Figura 3 se puede observar el número de «última actualización» por año de la totalidad de scripts, y demuestra que los scripts tienden a ser actualizados por la comunidad a lo largo del tiempo.

Los scripts que llevan a cabo la Aumentación Web son llamados augmenters [35], y es usual que sus creadores posean conocimiento en JavaScript, por ejemplo, para desarrollar extensiones de navegador o los previamente mencionados userscripts. Sin embargo, también existen enfoques que facilitan al usuario final su construcción o entendimiento, como por ejemplo [36].

Actualmente, los usuarios finales participan en la Web no solo bajo el rol de *consumidor*es de aplicaciones sino también como *productor*es [36] de sus propias soluciones. Ser un *productor* no es necesariamente sinónimo de tener conocimiento o experiencia en programación. Bajo este rol se agrupan personas con habilidad para programar textual o visualmente, como así también aquellas que no poseen experticia

[7] como https://addons.mozilla.org/ o https://chrome.google.com/webstore/category/extensions
[8] http://userscripts-mirror.org/ consultada el 18/12/2016. Cálculos realizados mediante scripting.
[9] https://greasyfork.org consultada el 18/12/2016
[10] https://openuserjs.org/ consultada el 18/12/2016

técnica, pero pueden construir aplicaciones utilizando una herramienta simple o una que provea un proceso asistido de construcción. Por ejemplo, herramientas basadas en formularios en la técnica Programming By Example [71].

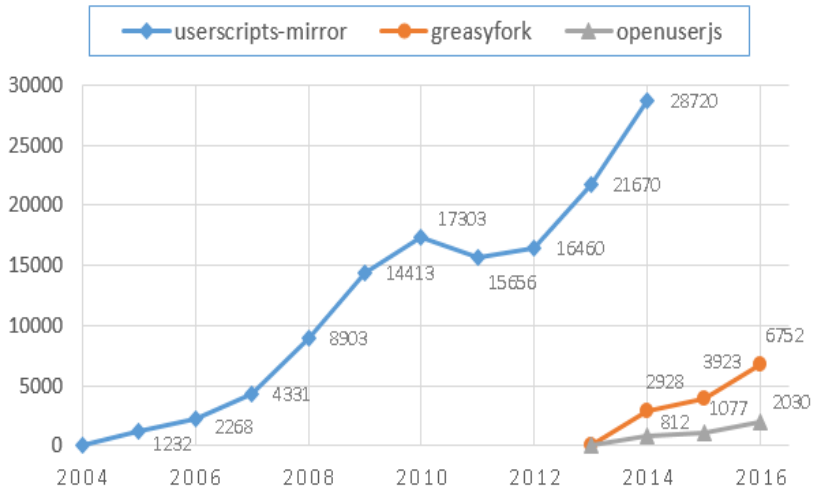

Figura 3. Cantidad de «últimas actualizaciones» de scripts por repositorio

La cantidad de características que pueden ser agregadas a una aplicación mediante WA son innumerables, y uno de los beneficios principales es poder consumir e integrar información de diversas fuentes. Por ejemplo, los artículos publicados en Ebay pueden ser aumentados con información extra que permita al usuario: decidir cuándo comprarlos según un análisis de la evolución sus precios; si se trata de un producto de calidad, en base al rating y los comentarios de otros usuarios; si es conveniente adquirirlo en otro sitio, verificando precios externos. De la misma manera, libros en Goodreads[11] pueden ser aumentados con más opciones de compras o con recomendaciones de audios relacionados al libro o su autor, como audiolibros extraídos de ivoox[12]. Las páginas de actores en IMDB[13] pueden ser mejoradas con fotos de Google Images, resultando de utilidad especialmente en aquellos perfiles que cuenten con pocas o ninguna foto del artista en cuestión. Las noticias de un diario online, como Página 12[14], pueden ser aumentados con tweets relacionados a sus titulares.

Aplicar técnicas de WA sobre aplicaciones Web móviles implica que, además de las posibilidades comúnmente adoptadas (como la personalización o la incorporación de recomendaciones), las aplicaciones pueden ser mejoradas con características del

[11] https://www.goodreads.com/
[12] http://www.ivoox.com
[13] http://www.imdb.com/
[14] https://www.pagina12.com.ar/

contexto. Por ejemplo, retomando el ejemplo de «Página 12» y teniendo la posibilidad de percibir el nivel de presión sonora en el entorno del usuario (el acceso al micrófono del dispositivo basta para tal fin), se podría recuperar tweets con contenido multimedia ante escenarios silenciosos y con contenido textual ante entornos ruidosos. Adicionalmente, se podría construir la consulta de búsqueda mediante la API REST de Twitter[15] para recuperar tweets según la ubicación actual del usuario, con la intención de recuperar solo aquellos tweets publicados a no más de 20 km. Se podría tener en cuenta la hora actual para resaltar aquellas noticias que fueron publicadas en la última hora, y ofrecer una vista especializada en la que estas noticias sean las únicas visibles cuando el dispositivo se encuentre en modo *landscape*. El acceso a la hora actual puede obtenerse simplemente instanciando la clase Date[16] en JavaScript. Tanto el nivel de presión sonora como la ubicación y la orientación del dispositivo requieren de APIs especializadas: la de WebRTC[17], la de Geolocalización[18] y la de orientación[19], respectivamente. Del mismo modo que con Twitter, es posible utilizar otras APIs, como la de YouTube[20], o definir Servicios de Búsqueda [11] basados en motores de búsqueda existentes en la Web y sus componentes de interfaz del usuario. Finalmente, una manera de inyectar el contenido recuperado mediante estas APIS en el portal de noticias es gracias a los privilegios del código de una extensión de navegador, por ejemplo, utilizando el SDK de Firefox for Android[21].

Aunque existen enfoques que permiten aumentar aplicaciones Web desde dispositivos móviles [22][55][100], los mismos han sido creados y limitados para usuarios bajo el rol de *productores* con conocimiento en programación (de ahora en más, los llamaremos «desarrolladores»). En relación a ello, en una primera instancia se presentó un enfoque llamado MoWA [24][12], soportado por un framework de desarrollo. Básicamente, MoWA facilita a los desarrolladores el proceso de creación de aplicaciones Web móviles basadas en adaptación del lado del cliente, soportando la incorporación de nuevo contenido y funcionalidad dependientes del contexto, directamente desde el front-end (el navegador). En dicho enfoque, los scripts de aumentación llegaban a manos de los usuarios finales mediante una plataforma de Crowdsourcing. En ella, los desarrolladores compartían las soluciones desarrolladas en respuesta a la demanda de algún usuario final, o simplemente haciendo sus propias soluciones públicas en el repositorio. Sin embargo, el usuario final dependía de un

[15] https://dev.twitter.com/rest/reference/get/geo/search
[16] http://www.ecma-international.org/ecma-262/7.0/index.html#sec-date-constructor
[17] https://webrtc.github.io/samples/src/content/getusermedia/volume/
[18] https://developer.mozilla.org/en-US/docs/Web/API/Geolocation
[19] https://developer.mozilla.org/en-US/docs/Web/API/DeviceOrientationEvent
[20] https://developers.google.com/youtube/v3/docs/search/list
[21] https://developer.mozilla.org/en-US/Add-ons/Firefox_for_Android

desarrollador para obtener una solución de aumentación que cubra sus necesidades en un escenario concreto.

En la presente obra el objetivo cambia en dirección a un rol de usuario diferente: *productor*es sin conocimiento en programación. El desafío consiste en proveer a este tipo de usuarios con herramientas que les permita especificar cómo sus aplicaciones Web de interés deben ser aumentadas, en relación a valores del contexto. Esta es, además, una manera de conseguir que ellos mismos puedan mejorar sus propias experiencias Web e incluso ayudar a otros que no poseen este conocimiento, convirtiéndose en proveedores de soluciones. Esta afirmación es razonable, ya que existen estudios [87][95] que han demostrado la tendencia de los usuarios finales a ser quienes satisfacen sus propias necesidades ante escenarios de dominios específicos, creando sus propias aplicaciones mediante una herramienta de Desarrollo por Usuarios finales (EUD, del inglés «End-User Development») [71].

El enfoque propuesto incorpora principios de EUD para permitir que los usuarios finales aumenten sus sitios Web preferidos con características móviles. De esta manera, satisfacen sus propias necesidades creando sus propias aplicaciones Web y sacando provecho de la información y los servicios ya existentes en la Web, enriqueciéndolos con diferentes características móviles. Como soporte al mismo, se presenta un framework para la creación de aplicaciones de Aumentación Web Móvil, una herramienta de autoría para la creación de dichas aplicaciones en forma visual, y una evaluación con 21 usuarios finales.

1.1. Objetivos

En este trabajo se extiende el enfoque de MoWA con el objetivo de soportar un rol de usuario diferente; se busca desarrollar un enfoque que permita a los usuarios finales adaptar la Web de acuerdo con sus propios requisitos y a la información del contexto, mediante la creación de aplicaciones de Aumentación Web Móvil construidas *on-demand*, desde sus propios dispositivos móviles, utilizando una herramienta visual de Programación por Usuarios Finales. Para conseguirlo, se establecieron los siguientes objetivos específicos:

- Analizar las alternativas para el desarrollo de aplicaciones Web Móviles mediante el uso de un enfoque EUD basado en aumentación.
- Diseñar un enfoque en el cual los desarrolladores crean componentes específicos del dominio, llamados *builders*, que representan unidades de construcción que el usuario final puede combinar haciendo uso de una herramienta de autoría, para crear sus propias aplicaciones específicas de dominio.
- Demostrar la factibilidad técnica del enfoque creando una solución tecnológica

que soporte la autoría, desde móviles, de aplicaciones que aumenten sitios Web existentes con características del contexto, destinada a usuarios finales sin conocimiento o experiencia en programación.

- Posibilitar el desarrollo *in-situ*, que permita al usuario final crear la aplicación en el mismo contexto físico en el que será utilizada, *on-demand*, de manera oportuna.
- Contemplar la flexibilidad del enfoque para resolver problemas de dominios específicos, tales como Hipermedia Móvil, Context Awareness o Realidad Aumentada.
- Demostrar la factibilidad operativa del enfoque, diseñando y llevando a cabo un experimento que tome la herramienta de autoría como unidad experimental, y un grupo de usuarios finales como participantes.

Como resultado de aplicar este enfoque, el usuario será capaz de crear aplicaciones móviles de Aumentación Web desde su dispositivo móvil y compartirlas en una comunidad Web, para que otros usuarios puedan utilizarlas o hacer sus contribuciones al proyecto.

1.2. Contribución

Los siguientes puntos resumen la contribución del presente trabajo:

- Un enfoque orientado a los usuarios finales para el desarrollo visual de aplicaciones de Aumentación Web Móvil del lado del cliente y desde dispositivos móviles, posibilitando la modalidad de creación *on-demand* e *in-situ*.
- Un framework, que propone la estructura tanto conceptual como de soporte para la construcción de aplicaciones de Aumentación Web Móvil.
- Una herramienta soportando la creación de aplicaciones de Aumentación Web Móvil, basada en los artefactos contemplados en el framework.
- Un experimento, validando el enfoque mediante el uso de la herramienta de soporte por parte de un grupo de 21 usuarios finales.

1.3. Publicaciones

Durante el proceso de investigación se publicaron cuatro artículos directamente relacionadas al tema de esta obra; uno de ellos fue presentado en un workshop, dos en revistas indexadas y, finalmente, uno en una conferencia. En todos los casos, se trató de un evento de alcance internacional:

1. Challiol, C., Firmenich, S., Bosetti, G. A., Gordillo, S. E., & Rossi, G. *Crowdsourcing mobile web applications* [24].

2. Bosetti, G., Firmenich, S., Gordillo, S. E., Rossi, G., & Winckler, M. *An End-User Development approach for Mobile Web Augmentation* [13].
3. Bosetti, G., Firmenich, S., Gordillo, S. E. & Rossi, G. *An approach for building Mobile Web Applications through Web Augmentation* [12].
4. Bosetti, G., Firmenich, S., Rossi, G., & Winckler, M. *Supporting Mobile Web Augmentation by End Users* [14].

Adicionalmente, durante este período surgieron varios temas de investigación relacionados con el tema principal de esta obra (Aumentación Web) que permitieron obtener conocimiento de diferentes dominios y desarrollar una visión mucho más amplia sobre las soluciones a implementar con el framework del trabajo de tesis. Gracias a esto, también se pudo poner en práctica otras técnicas de Programación para el Usuario Final y buenas prácticas en la resolución de problemas de aumentación que, tras su maduración, se vieron reflejados en el framework de desarrollo propuesto en este trabajo. Estos trabajos dieron origen a las siguientes publicaciones:

1. Firmenich, S., Bosetti, G., Rossi, G., Winckler, M. & Barbieri, T. *Abstracting and Structuring Web Contents for Supporting Personal Web Experiences*. [46]
2. Bosetti, G., Firmenich, S., Rossi, G., Winckler, M. & Barbieri, T. *Web Objects Ambient: An Integrated Platform Supporting New Kinds of Personal Web Experiences* [15].
3. Firmenich, S., Bosetti, G., Rossi, G. & Winckler, M. *Flexible Distribution of Existing Web Interfaces: An Architecture Involving Developers and End-Users* [44].
4. Bosetti, G., Firmenich, S., Fernandez, A., Winckler, M., & Rossi, G. *From Search Engines to Augmented Search Services: An End-User Development Approach* [11].
5. Firmenich, S., Bosetti, G., Rossi, G., & Winckler, M. *End-user software engineering for the personal web: poster*. [45]
6. Firmenich, S., Bosetti, G., Rossi, G., Winckler, M. & Corletto, J.M. Distributed Web Browsing: supporting frequent uses and opportunistic requirements [47].

1.4. Estructura del documento

El presente trabajo está estructurado en 9 capítulos que, excluyendo el presente, se detallan a continuación:

- Capítulo 2. Introduce los principales temas abordados en el enfoque propuesto: Aumentación Web, Aplicaciones Web Móviles, Desarrollo por Usuarios Finales y Crowdsourcing. También se incluye una subsección especial para presentar el framework MoWA, orientado a desarrolladores.
- Capítulo 3. Describe los trabajos existentes con relación al aporte de este

trabajo. Se presentan, primero, trabajos cuya herramienta de soporte consiste en una aplicación adaptable al contexto. Luego, se presentan aquellos enfoques que contemplan al usuario final como productor de aplicaciones Web y móviles, ya sea construyéndolas desde entornos estacionarios como móviles. También se presentan frameworks enfocados en la adaptación Web y, a continuación, aquellos que contemplan Aumentación Web por usuarios finales. Finalmente, se mencionan enfoques de crowdsourcing relacionados al mundo móvil.

- Capítulo 4. Identifica el problema de investigación sobre el cual se motiva el enfoque propuesto, y se plantean preguntas de investigación. Adicionalmente, se describen escenarios motivadores en los que la adaptación Web Móvil es necesaria para mejorar la experiencia de usuario y una validación inicial que asegura la factibilidad técnica de tal adaptación mediante scripting.
- Capítulo 5. Presenta un enfoque para el desarrollo de aplicaciones de Aumentación Web Móvil por usuarios finales. El mismo es concebido como parte de un contexto de Crowdsourcing de aumentaciones, que es detallado junto a los posibles roles de sus participantes y el proceso de autoría concreto.
- Capítulo 6. Presenta el soporte conceptual y la herramienta de soporte al enfoque. Este capítulo comienza presentando detalles técnicos sobre aumentación del lado del cliente, proponiendo un modelo conceptual de aplicaciones MoWA y presentando finalmente la herramienta mediante el uso de escenarios descriptivos.
- Capítulo 7. Presenta una evaluación del enfoque en el que participaron 21 personas de diversas características demográficas, que utilizaron la herramienta para la construcción de una aplicación de MoWA soportando la asistencia al recorrido en un museo. Se presenta un análisis de las características de los participantes y de sus dispositivos, el diseño del experimento y los resultados obtenidos. Finalmente, se presenta un análisis de los factores que amenazan la validez del diseño y una discusión sobre los resultados.
- Capítulo 8. Expone las conclusiones del trabajo, menciona el estado actual del mismo y las direcciones futuras de investigación.

Además, se incluyen anexos que amplían y respaldan la información presentada:

- Anexo A. Presenta los datos que sustentan las afirmaciones sobre sitios Web populares y sus contrapartes nativas.
- Anexo B. Describe un script ejemplo para la adaptación Web del lado del cliente. El mismo fue utilizado para la generación de la Figura 4, con la intención de clarificar el concepto de Aumentación Web.
- Anexo C. Presenta un conjunto de scripts utilizados para la generación de las figuras con los escenarios motivadores del Capítulo 4.

2. Fundamentos

En este capítulo se describen conceptos sobre las áreas de conocimiento relacionada a la contribución de este trabajo. En concreto, se abordarán aquellos referidos a los campos de la Aumentación Web, las Aplicaciones Móviles, el Desarrollo por Usuarios Finales y el Crowdsourcing. En todos los casos se expone un problema que el campo resuelve, su definición, conceptos fundamentales y técnicas existentes. Por último, y con la finalidad de sentar las bases necesarias para una mejor comprensión del tema principal de la obra, se presenta un enfoque previo que dio origen a este trabajo.

2.1. Aumentación Web

Tal como lo menciona Oppermann [80], los sistemas no se diseñan para un usuario y una tarea en particular, sino más bien enfocados en un conjunto de usuarios de características similares y para llevar a cabo tareas específicas. Es aquí donde surge la necesidad de adaptarlos.

El término adaptación, en su más amplio sentido, implica reacomodar, modificar algo para que tenga una función distinta para la cual fue concebido. Dentro del campo de la Computación, el mismo hace referencia a aquellos sistemas que tienen la capacidad de ser resilientes a −o de modificarse en función de− cambios en las necesidades del usuario. Existen aquellos sistemas adaptables que proveen al usuario las herramientas para personalizarlo (ej. cambiar el tamaño de fuente, el orden de las secciones) y aquellos que automáticamente se modifican de acuerdo con las necesidades percibidas (ej. mostrar en un menú principal las secciones que el usuario navega frecuentemente, o aumentar la luminosidad de la pantalla cuando se está en un ambiente obscuro). Esta diferenciación sugiere llamar al primer tipo de aplicación «adaptables», mientras que al segundo «adaptativas».

La adaptación dentro del contexto Web puede ocurrir como parte de una aplicación, pero también como parte de una segunda aplicación que decora a la primera. Dentro del primer grupo, existen herramientas y frameworks como [18], pensados para contemplar la posibilidad de adaptación *from scratch* o mediante modificaciones en el código fuente de una aplicación ya desarrollada. Pero cuando no se tiene acceso al mismo o no se lo quiere modificar, se puede optar por realizar cambios sobre la aplicación Web adyacente, modificando su capa de presentación.

La Aumentación Web (WA, del inglés «Web Augmentation») [35][17] es un conjunto de técnicas que permite manipular aplicaciones Web existentes de acuerdo a los requisitos del usuario. De esta manera, los usuarios obtienen un valor agregado sobre lo ofrecido por una página o sitio Web. Esta modificación se logra manipulando el documento que especifica una página Web, y los cambios pueden involucrar

comportamiento, contenido, estructura y/o estilo. Por ejemplo, mediante un userscript, se puede tomar como base los artículos relacionados a webextensions en la MDN[22] y aumentarlos con nuevo contenido, tal como se muestra en la Figura 4, agregando una sección extra con preguntas y respuestas obtenidas de la comunidad de StackOverflow. Estos resultados pueden recuperarse construyendo una query («webextensions» + el título del documento en MDN) y utilizando la rest-API de stackoverflow[23] para luego recuperar su documento, parsearlo y evaluar las expresiones que permitan obtener los elementos de interés (en este caso, las preguntas, tal como se muestra en el «Anexo B: Adaptación Web client-side»).

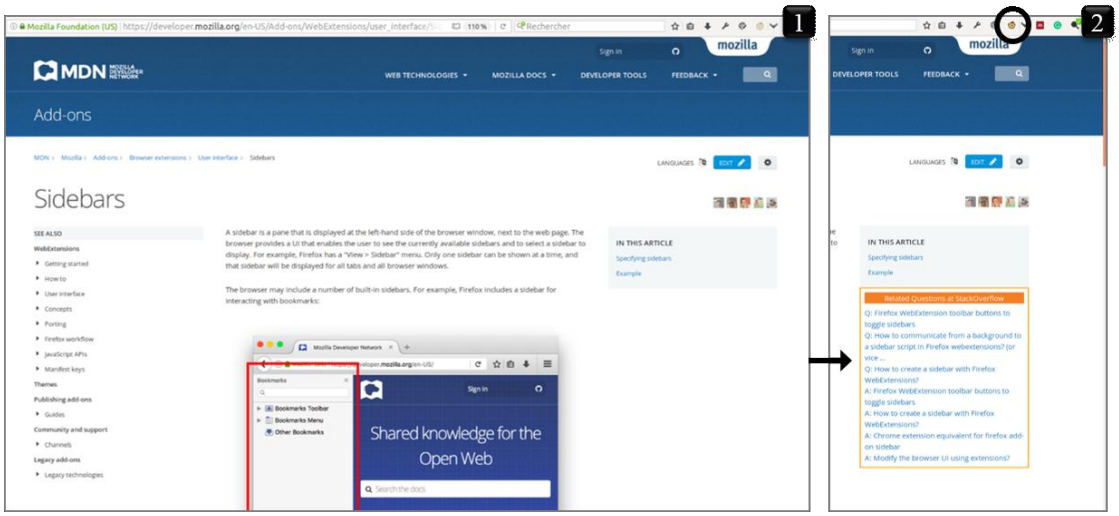

Figura 4. Ejemplo de Aumentación Web sobre Páginas Amarillas

Mediante estas transformaciones, la WA permite que los usuarios Web accedan a una versión mejorada de una aplicación existente que no fue diseñada originalmente para afrontar las necesidades que cubren esas características agregadas. Esto puede contemplar requisitos funcionales o no funcionales. Desde el punto de vista del propietario de una aplicación Web, es imposible anticipar o cubrir todas las necesidades de sus usuarios. La WA le permite cubrir −al menos temporalmente− las necesidades de clientes que no están completamente satisfechos con la aplicación y también detectar posibles nuevas características a considerar para un futuro *release* de la misma.

Es preciso resaltar que el valor agregado mediante técnicas de WA no está limitado al contexto de una sola aplicación Web, sino que se pueden integrar elementos de diferentes orígenes. El «cómo», está estrechamente relacionado a la implementación de la técnica, que será abordado con posterioridad en esta obra. En este punto basta con comprender que, si la aumentación se realiza del lado del cliente, se puede conseguir

[22] Mozilla Developer Network: https://developer.mozilla.org/en-US/Add-ons/WebExtensions/user_interface/Sidebars
[23] Por ejemplo, https://stackoverflow.com/search?q=webextensions+sidebars

accediendo al Modelo de Objetos del Documento (DOM, del inglés «Document Object Model»), que permite modificar dinámicamente un documento obtenido mediante peticiones HTTP, e inyectándole contenido externo dentro de un marco en el cual se posean privilegios suficientes para evitar la política del mismo origen[24], como lo es el de una extensión de navegador Web.

La integración de contenido Web desde diferentes orígenes es un punto de convergencia con el propósito de un mashup Web [33]. La diferencia está en que, en la WA, esta combinación es posible −y de hecho es requisito hacerlo− en el contexto de una de las fuentes involucradas, es decir, tomando al menos una página Web como base y adicionando el contenido del resto de los orígenes a ella. Los mashups Web, en cambio, presentan la información de las diferentes fuentes en una nueva manera, un nuevo contexto con un layout único, y son aplicaciones autónomas.

Existen varias maneras de llevar a cabo la WA, y en este trabajo se presentan de acuerdo con el contexto en el cual se manipula un documento. En la Figura 5 se puede apreciar que la aumentación puede ocurrir en el servidor, en el proxy o en el cliente.

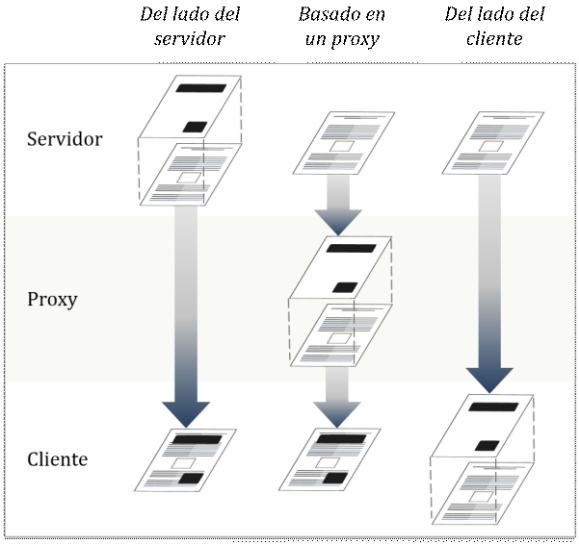

Figura 5. Aumentación Web según contexto de ejecución

Si bien la primera es la menos frecuente, un sistema de aumentación podría estar situado en el mismo contexto que la aplicación base, como en [5]. Si bien lo que percibe el cliente puede parecer una aplicación autónoma, la aplicación base no deja de ser una unidad independiente del sistema de aumentación y podría, por ejemplo, ser ofrecida bajo diferentes subdominios; uno para acceder a la aplicación pura y otro para la

[24] Same Origin Policy by the W3C: https://www.w3.org/Security/wiki/Same_Origin_Policy

aplicación aumentada. En el segundo caso (ej. [55]), la responsabilidad de aumentar recae en el proxy, tal como se explica en [6]: cuando el cliente solicita navegar una página, el proxy recibe la solicitud y la redirecciona al servidor. Cuando obtiene la respuesta, el proxy aplica los cambios necesarios sobre el documento y devuelve esta nueva versión al cliente. En algunos casos la aumentación es transparente mientras el usuario navega la Web, pues el navegador está configurado para redireccionar todas las solicitudes al proxy, mientras que en otros se espera que el usuario acceda a una URL determinada desde donde se accede a la aplicación en su versión aumentada, o desde donde tendrá que ingresar la URL de la página a adaptar.

Por su parte, que la aumentación ocurra del lado del cliente implica que el navegador o una aplicación nativa consuman la especificación de una página Web y aplique cambios a ella. La técnica más común es introducir estos cambios mediante scripting, utilizando la API del DOM, una vez que el documento ha sido parseado. Las implementaciones más comunes de este mecanismo son las extensiones de navegador (como la de Horváth & Šimko [64]) y los userscripts, ejecutables mediante weavers [35] tales como GreaseMonkey[25] o Sticklet [36]. Un weaver[26], entonces, es una aplicación que permite ejecutar dichos scripts sobre cualquier página Web. También existen enfoques basados en Bookmarklets [22], o aplicaciones nativas que aumentan los documentos de las páginas que consumen de acuerdo a un criterio determinado y sin necesidad de ser un weaver [40][100].

Es preciso resaltar que, si bien el término WA puede resultar no tan popular, su puesta en práctica si lo es. Existe una gran comunidad de desarrolladores y aficionados que crean y mantienen artefactos de aumentación, como lo son los userscripts y extensiones de navegadores Web. En el primer caso se puede citar repositorios como Userscripts[27], Greasy Fork[28], OpenUserJs[29], UserStyles[30]. De ellos, Greasyfork cuenta actualmente con un total de 11201 scripts y 975027 instalaciones, mientras que Userscripts mantiene 130.956 scripts, con un total de 1.203.605.530 instalaciones. Dichos números fueron calculados tras el análisis de elementos del DOM mediante scripting. En el segundo caso, se pueden mencionar los repositorios oficiales de extensiones de navegadores Web, como el de Firefox o Chrome. El equipo de Mozilla

[25] Greasemonkey: http://www.greasespot.net/
[26] Los weavers no son de uso exclusivo del lado del cliente. Puede ocurrir, como en [5], que el mismo weaver sea encuentre en el mismo host que la aplicación.
[27] Userscripts: http://userscripts-mirror.org/
[28] Grease Fork: https://greasyfork.org/
[29] OpenUserJs: https://openuserjs.org/
[30] UserStyles: https://userstyles.org/

reportó 2 billones de descargas[31] en julio de 2010. Dos años después fueron 3 billones[32] y finalmente, en enero de 2015, resultó en un total de 4 billones[33]. Esto demuestra que, aunque el flujo puede variar, existe aún la tendencia al desarrollo y al consumo de extensiones. Actualmente, el repositorio ofrece un total de 20.045 extensiones[34].

El presente trabajo presenta la Aumentación Web Móvil como una especialización de la Aumentación Web tradicional. En la primera, las aumentaciones son notificadas sobre cambios en el contexto y tienen la capacidad de adaptarse en consecuencia. Además, este trabajo está enmarcado en el lado del cliente y la adaptación se realiza mediante un weaver implementado como extensión de navegador Web móvil que permite ejecutar scripts de aumentación. El beneficio bajo esta modalidad es tener acceso a API de alto y bajo nivel del navegador, que permite llevar a cabo operaciones que desde el contexto de una aplicación Web no se podría. Por ejemplo, se puede integrar y manipular libremente contenido proveniente de más de una sola fuente, implementar interacciones cross-domain, agregar elementos de la interfaz del navegador como parte de una solución, tener un acceso menos limitado al sistema de archivos, habilitar o deshabilitar componentes sin necesidad de intervención del usuario, etc. Más aún, con la evolución de los dispositivos móviles y la inclusión de cada vez más características en sus navegadores Web, hoy es posible el desarrollo de extensiones de navegador móvil que accedan a información del contexto y a APIs[35] especializadas para crear interfaces de usuario, interactuar con la Web o con el navegador específico de la plataforma.

2.2. Aplicaciones Web Móviles

Dentro de la Ciencia de la Computación, la Movilidad ha cobrado notable importancia en los últimos años, dada la creciente necesidad del hombre de moverse en diferentes entornos haciendo uso de sistemas informáticos para llevar a cabo tareas tales como acceder a recursos en línea, recolectar información directa de su entorno, obtener asistencia al recorrer lugares, etc. Actualmente, las Aplicaciones Móviles [49] se encuentran presentes en cada vez más dominios: turismo, entretenimiento, educación, marketing, tareas asistidas, recolección de datos, realidad aumentada, redes sociales,

[31] https://blog.mozilla.org/addons/2010/07/01/2-billion-downloads/
[32] http://blog.mozilla.org/theden/2012/07/30/the-best-of-3-billion-add-on-downloads
[33] http://blog.mozilla.org/theden/2015/01/19/firefox-add-ons-hit-4-billion-downloads
[34] https://addons.mozilla.org/es/firefox/extensions/?sort=name Consultado el 04/09/2017
[35] Por ejemplo, la API de bajo nivel en https://developer.mozilla.org/es/Add-ons/SDK/Low-Level_APIs o de alto nivel en https://developer.mozilla.org/en-US/Add-ons/SDK/High-Level_APIs

entre otras. Y no es casualidad que en 2015 la ITU (International Telecommunication Union) haya registrado más de 7 billones de suscripciones móviles en el mundo, que para el año 2000 eran menos de un billón (738 millones) [86]. El acceso a un dispositivo móvil es cada vez más frecuente, y muchas de las tareas que un usuario llevaba a cabo en una computadora hoy puede realizarlo desde su móvil. El acceso a un plan de datos también se ha convertido en moneda corriente; la ITU, además, reportó que un plan de banda ancha móvil a nivel global es −1.7 veces− más económico que uno fijo, y que la penetración de la banda ancha móvil en el mercado creció 12 veces desde 2007 [86].

Sin embargo, pese a su importancia actual, la definición de Aplicaciones Móviles ha sido un tema controversial a lo largo de los años. En términos generales, una Aplicación Móvil es un sistema de información diseñado para ser ejecutado sobre plataformas móviles. A partir de este punto, algunos autores sostienen que además es fundamental que provean acceso continuo a recursos remotos mientras el usuario se desplaza físicamente [50][96]. Sin embargo, existe un gran número de aplicaciones que dicen ser móviles, pero no cumplen con estas características. Basta con ingresar a alguno de los mercados de aplicaciones −como Google Play o App Store− para verificarlo; calculadoras, sticky notes, juegos que no requieren ni siquiera conexión a Internet.

Tal como se menciona en [43], los primeras experiencias de percepción del contexto mediante móvil se lograron mediante la interconexión de diferentes dispositivos de mucho mayor tamaño a los que hoy estamos acostumbrados; estos dispositivos eran creados de acuerdo a las necesidades de investigación del momento. Luego, los avances tecnológicos permitieron a los teléfonos básicos (feature phones) comunicarse con sensores y servidores externos, y más tarde aparecieron sensores de tamaño tan pequeño que pudieron ser integrados a los dispositivos móviles; los smartphones y las tablets son un claro ejemplo de ello.

Actualmente, los dispositivos móviles vienen equipados con una gran cantidad de sensores embebidos −GPS, giroscopio, cámara, micrófono, etc.−, a partir de los cuales se puede determinar valores concretos de tipos de contexto [2][8], tales como la posición, la orientación, el nivel de sonido o de luz en el ambiente, velocidad, perfil de usuario, etc. La obtención de esta información, sumado al avance en la capacidad de procesamiento y almacenamiento de los dispositivos, hicieron posibles nuevos tipos de aplicaciones móviles; conscientes de y resilientes al contexto de uso, es decir, que pueden utilizar esta información para ofrecer al usuario una experiencia adaptada a su actual situación y su constante cambio. Por ejemplo, adaptando el nivel de luminosidad de la pantalla cuando el usuario se encuentre en un ambiente obscuro, o reacomodando los elementos de un menú principal de acuerdo a la posición del usuario (podría haber comportamiento que es más utilizado desde su casa, desde su trabajo, desde la plaza de su barrio, etc.).

Existen diversos tipos de aplicaciones que utilizan información del contexto. Aquellas denominadas Sensibles al Contexto (CA, del inglés «Context Aware») [89],

monitorean constantemente el entorno del usuario y adaptan su comportamiento en consecuencia. Entre las variables más explotadas del contexto se encuentra la ubicación geográfica o geo-posición, y en base a ella surgieron las denominadas aplicaciones Basadas en el Posicionamiento (del inglés «Location-Based») [9]. Un usuario de este tipo de aplicación podría, por ejemplo, obtener la lista de estaciones de subte más cercanas a su posición actual. Haciendo uso de la posición detectada por el dispositivo, también existen las aplicaciones de Hipermedia Móvil (del inglés «Mobile Hypermedia») [16]. Las mismas conocen, además, la posición de un conjunto de Puntos de Interés (PDI) y utiliza esta información para asistir al usuario en la navegación por el mundo real. Algunas aplicaciones de Realidad Aumentada Móvil (del inglés «Mobile Augmented Reality») [62] también consideran la posición y otros tipos de contexto para, por ejemplo, calcular la posición y orientación en que debe renderizar objetos digitales sobre la capa virtual superpuesta a la captura de la realidad. También existen aplicaciones móviles anticipatorias (del inglés «Anticipatory Mobile») [83]; aquellas que conscientes de su estado y del entorno en el que se encuentran, tienen la capacidad de cambiar de acuerdo a un modelo predictivo, ofreciendo al usuario información o funcionalidad que puede llegar a necesitar. Estos son algunos de los tipos de aplicaciones móviles que consumen información del contexto para adaptar su comportamiento, aunque representan un pequeño subconjunto del amplio espectro que abarca la Computación Móvil.

El presente trabajo está enfocado en soportar una gran variedad de aplicaciones móviles, sin embargo, se hará foco en solo dos tipos (Sensibles al Contexto e Hipermedia Móvil), poniendo en evidencia la factibilidad técnica y la posibilidad de reúso y extensión para soportar nuevos tipos de aplicaciones móviles.

La mayoría de los tipos de contexto utilizados por las aplicaciones Sensibles al Contexto, previamente mencionadas, están relacionados con el entorno y el dispositivo del usuario. Por ejemplo, la ubicación, la orientación, el ruido, la luz, el tiempo, el consumo de energía o las condiciones de la red. Sin embargo, tal como lo postulan Abowd et al. [2]:

> "*Context is any information that can be used to characterize the situation of an entity.*" [Contexto es cualquier información que pueda ser utilizada para caracterizar la situación de una entidad].

En consecuencia, el contexto implica muchos más tipos que los anteriormente mencionados. Esa lista puede extenderse con, por ejemplo, aspectos sociales o relacionados al usuario, como su actividad o su estado emocional, de relación o de interacción. Un amplio conjunto de tipos de contextos fue descripto por Emmanouilidis et al [39].

Por su parte, las aplicaciones de Hipermedia Móvil extienden el concepto de

navegación utilizado en la Hipermedia tradicional al mundo real. En ellas, el usuario puede acceder a la información tanto digital como físicamente. En el primer caso, su acceso implica el uso de enlaces digitales tradicionales, mientras que el segundo de links «caminables» (del inglés «Walking Links») o también conocidos como «del mundo real» (del inglés « Real-World Links) [75]. Mientras que navegar en el mundo digital implica hacer click en un enlace, en el segundo caso, también llamada «navegación física directa sensible al contexto» (del inglés Context Aware browsing based on Direct Physical Navigation) [16], requiere que el usuario cambie parámetros en su contexto físico, por ejemplo desplazándose físicamente consigue modificar su geo-posición y la hora actual. De esta manera, el sistema puede encontrar información asociada en el contexto digital y presentarla al usuario.

La navegación de un link caminable implica el movimiento físico del usuario [59]. Cuando el mismo selecciona uno de esos links, expresa la intención de caminar hacia un objetivo. El usuario se encuentra físicamente en una determinada geo-posición –la cual representa un PDI– y desde allí debe utilizar algún mecanismo de detección que le permita obtener información asociada a su contexto actual, es decir, la contraparte digital de dicho punto. Esta última debe contener instrucciones le permita al usuario final alcanzar el objetivo deseado (el próximo punto de interés a visitar). Por ejemplo, la aplicación puede mostrar un plano 2D con el recorrido hacia el objetivo seleccionado. En la Figura 6, se muestra un ejemplo del acceso a la información de acuerdo con el enfoque de Hipermedia Móvil.

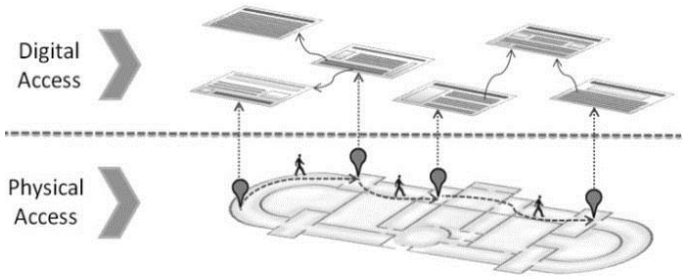

Figura 6. Ejemplo de acceso a la información de acuerdo a la Hipermedia Móvil

Sin importar a qué campo pertenezcan, las aplicaciones móviles pueden ser de diferente naturaleza; nativas, Web o híbridas [26]. En sus inicios, las aplicaciones móviles eran desarrolladas mediante código nativo y, entre sus ventajas, estaba el acceso a la API del sistema de base. Cuando las primeras aplicaciones Web móviles emergieron, no tenían los mismos privilegios que las nativas. Por ejemplo, no se podía acceder a información del contexto como la posición o la orientación del dispositivo. En ese marco surgieron frameworks especializados que producían aplicaciones híbridas, resultantes de una combinación entre lo nativo y lo Web.

Phonegap −actualmente Apache Cordova[36]− es un ejemplo de framework que permitió a los desarrolladores programar haciendo uso de lenguajes Web, pero con acceso a la API del sistema nativo. En aquel entonces, tampoco disponían de mecanismos que soporten el desarrollo de extensiones de navegador Web, por lo que las posibilidades de aumentar una aplicación Web también eran limitadas (al uso de proxies, por ejemplo). Sin embargo, los avances en la implementación de los navegadores móviles permiten actualmente al desarrollador −tanto de una aplicación Web como de una extensión del navegador− acceder a un conjunto de APIs que permiten el acceso a los valores sensados por el dispositivo. En consecuencia, las *aplicaciones Web puras* hoy pueden adaptar su comportamiento a estos valores.

Esta obra se encuadra en el desarrollo de aplicaciones Web móviles, en contraposición a aquellas nativas, y se considera que el valor agregado de una aplicación móvil frente a otros sistemas de información es, precisamente, su capacidad de movilidad y el uso de información recuperada mediante los sensores del dispositivo para adaptarse a diferentes contextos de uso. Además, dentro del amplio espectro de tipos de aplicaciones Móviles, se centra en aquellas de tipo Sensibles al Contexto e Hipermedia Móvil, que serán tenidas en cuenta al momento de desarrollar los escenarios descriptivos y el experimento final.

2.3. Desarrollo por Usuarios Finales

Actualmente, una gran cantidad de personas hacen uso de sistemas informáticos para resolver problemas de dominios específicos en sus vidas cotidianas. Cuando un nuevo problema se presenta, el usuario final puede tomar diferentes decisiones. El mejor escenario es aquel en el que existe −y el usuario final conoce− la herramienta adecuada que resuelve el problema y decide utilizarla. Lo cierto es que no existen aplicaciones que cubran las necesidades de todos sus usuarios. Pueden existir soluciones que cubran parcialmente los requisitos de algunos, pero una solución específica para cubrir esas necesidades insatisfechas requiere del desarrollo de una nueva aplicación. De allí se desprende el peor escenario, en el que el usuario no cuenta con los recursos necesarios para obtenerla (tiempo, conocimiento, financiamiento, etc.) y desiste de ello. Entre estos dos extremos, existen varias alternativas.

Si el usuario no posee el conocimiento o no dispone del tiempo necesario, puede encargarle esta tarea a un tercero. Pero delegar su construcción a un desarrollador implica tiempo (para encontrar al profesional apropiado, comunicar los requisitos,

[36] Apache Cordova: https://cordova.apache.org/

desarrollar y probar la aplicación, etc.) y probablemente dinero. También puede valerse de la ayuda de comunidades de crowdsourcing especializadas (ver Sección 2.4), en donde tanto usuarios como desarrolladores unen esfuerzos para dar soporte a sus necesidades a través de la Web. Se trata de una externalización abierta de tareas, en donde la parte interesada puede presentar convocatorias abiertas para que los miembros de la comunidad apliquen e intenten resolver un problema específico. Esta alternativa también conlleva desventajas similares a las asociadas a la alternativa de depender de un desarrollador, pues también se requiere de tiempo para la creación de la convocatoria, para que un miembro de la comunidad intente resolver el problema y, dependiendo de la plataforma seleccionada, también puede significar un costo económico.

Dejando de lado los motivos, si no existe o el usuario no conoce una herramienta que resuelva su problema, ni tampoco puede obtener una solución con la ayuda de terceros, todavía queda una alternativa a considerar: tiene la posibilidad de satisfacer sus necesidades creando él mismo su propia aplicación. Esto no implica que este sea o que exista un orden entre las alternativas mencionadas. Esta última podría ser la primera de las alternativas que un usuario adopte. De hecho, existen estudios que demuestran que existe una fuerte tendencia [87][95] en la que los usuarios finales crean, modifican o extienden sus propios artefactos de software.

La necesidad de los usuarios finales de crear rápidamente sus propias soluciones, ya sea para escenarios oportunistas o frecuentes, dio origen a lo que se conoce actualmente como EUD (Desarrollo por Usuarios Finales, del inglés «End-User Development») [71][82]. EUD ha sido definido en [71] como:

> "*a set of methods, techniques, and tools that allow users of software systems, who are acting as non-professional software developers, at some point to create, modify or extend a software artefact*" [un conjunto de métodos, técnicas y herramientas que permite a los usuarios de sistemas de software, quienes actúan como desarrolladores de software no profesionales, a crear, modificar o extender en cierta medida un artefacto de software].

Este paradigma es de interés para los campos de la Ingeniería de Software y de Interacción Persona-Ordenador. En comparación a los enfoques de desarrollo de software tradicional, en EUD una misma persona puede jugar el rol de desarrollador y de usuario final sin que sea un requisito tener conocimiento en procesos formales de desarrollo de software. Lo significativo de esta situación es que el usuario final conoce sus necesidades y las del contexto del problema a solucionar mejor que nadie, y puede satisfacerlas utilizando entornos de programación de alto nivel que, de alguna manera, abstraen la complejidad de este proceso.

Algunos autores [36] también hacen una separación de roles dentro de la categoría de usuarios finales: entre *productor*es y *consumidor*es. Los primeros son aquellos que construyen aplicaciones mediante alguna herramienta de EUD, mientras que los segundos son quienes simplemente hacen uso de la aplicación que desarrolladores y

productores crean.

En EUD existen diferentes técnicas de programación que le permiten al usuario final construir sus aplicaciones:

- PBE (Programación Por Ejemplos, del inglés «Programming by Example») [58], también conocida como PBD (Programación Por Demostración, del inglés «Programming by Demonstration»). Este enfoque no requiere que el usuario programe, sino que grabe una secuencia de acciones que realiza sobre un sistema, generando una aplicación en base a la especificación de dichas acciones. Es posible parametrizar algunas entidades dentro de la especificación, de modo que la aplicación resultante pueda reproducir las acciones que el usuario realizó, pero también sobre un conjunto de datos de entrada diferentes. Una de las grandes ventajas de este enfoque es que el usuario construye programas a través de componentes de la UI a los que está acostumbrado. Un aspecto desfavorable es la dificultad de acceder a la funcionalidad interna del sistema que decora, o de permitirle especificar estructuras de control.
- Lenguaje Específico de Dominio Textual (del inglés «Textual Domain Specific Language») [71]. Los lenguajes basados en construcciones textuales son considerados uni-dimensionales, y requieren que el usuario aprenda los elementos de un lenguaje formal y una sintaxis. Si bien resulta difícil considerar al usuario final escribiendo líneas de código, existen diversos enfoques que demuestran que un usuario final puede hacer uso de ellos fácilmente. Establecer un límite entre qué lenguajes tienen un nivel de abstracción suficiente para ser considerados de usuarios finales es un tema con múltiples posiciones al respecto. Dentro de esta categoría y tal como se menciona en [66], existen diferentes tipos de usuarios finales con diferentes niveles de conocimiento y tareas por llevar a cabo; científicos usando MathLab para hacer simulaciones, músicos utilizando sintetizadores para crear música digital, contadores utilizando planillas de cálculos, etc.
- Programación Visual (del inglés «Visual Programming») [25]. Permite al usuario final crear un programa combinando constructores visuales, en lugar de textuales, como por ejemplo widgets o piezas de rompecabezas. Detrás del objetivo es proveer al programador representaciones más propicias para la comprensión humana [93], lo visual representa una ventaja, puesto a que no tiene las mismas limitaciones que un lenguaje natural: puede ser comprendido por mucha gente, independientemente de su idioma. Es importante destacar que, aunque generalmente el público de este tipo de aplicaciones son usuarios finales, no toda herramienta de programación visual es necesariamente de EUD. Las herramientas de programación visual están orientadas a abstraer el nivel de complejidad de los lenguajes textuales mediante la representación visual de entidades y operadores combinables. La configuración de

estos elementos puede hacerse mediante asistencia con formularios. WYSIWYG (del inglés «What You See Is What You Get») es un ejemplo comúnmente utilizado como lenguaje visual, que permite la manipulación directa de elementos [66]. Sin embargo, resulta difícil determinar cuál es el límite que determina si una herramienta es EUD o no. Por ejemplo, Visual Studio[37] permite a sus usuarios componer interfaces visuales para sus aplicaciones mediante drag-and-drop. Sin embargo, el comportamiento de la aplicación requiere la codificación en un lenguaje textual de propósito general, como lo es Visual Basic o C#.

- Natural Programming. La idea detrás de este enfoque es permitir a las personas expresar sus ideas de la misma manera en que las piensan; es decir, expresar algoritmos y datos de la forma más cercana a sus expresiones naturales [76]. Tal como se explica en [81], esto no necesariamente significa utilizar exactamente la sintaxis de un lenguaje natural, puesto que eso hace más difícil al usuario comprender los límites computacionales [77], sino diseñar una herramienta que resulte natural –al menos– para un grupo determinado de personas. Esto, por supuesto, puede resultar en una herramienta que no es óptima para uso universal o general. La idea es reducir las transformaciones mentales que el usuario debe hacer (cognitive directness [68]); acortar la distancia entre el plan mental del usuario sobre cómo resolver un problema y su representación en el lenguaje de programación particular que se está utilizando. Cuanto más natural es una interfaz y menos sean las transformaciones mentales que el usuario debe realizar, más fácil resulta al usuario aprenderla y utilizarla.

- Programación Dirigida por Modelos (del inglés «Model-Driven») [7]. Enfoques bajo esta técnica permiten al usuario final construir aplicaciones mediante la especificación de un modelo que expresa conceptos específicos de un dominio, y a partir del cual se puede derivar el código fuente de una aplicación. Al igual que las modalidades anteriores, no toda herramienta de modelado es EUD, puesto que existen aquellas que requieren del conocimiento específico de un programador profesional [66].

Es preciso señalar que mientras que en el EUD el usuario compone sus aplicaciones combinando constructores de diferente naturaleza (textual, lenguajes visuales, widgets, etc.), existen enfoques meramente de configuración o parametrización [71] que no deberían ser tenidos en cuenta al momento de hablar de EUD, ya que el usuario no puede resolver problemas de un dominio diferente al cual fueron concebidos. El desafío del EUD, entonces, consiste no solo en proveer a los usuarios finales con herramientas fáciles de aprender y utilizar, sino que también les permita resolver problemas de

[37] https://www.visualstudio.com/

diferente naturaleza.

Numerosos enfoques dieron origen a un gran número de herramientas de EUD en relación a la Web; algunos para el propio desarrollo de aplicaciones, otros para la extensión de aplicaciones existentes. Es allí y en torno a estas aplicaciones que se originaron comunidades donde los usuarios comenzaron a compartir sus producciones, como las previamente mencionadas Userscripts o GreaseFork.

Con relación al mundo móvil, existen enfoques que permiten construir aplicaciones sacando provecho tanto de la movilidad como de la información del contexto. Dentro de estas características, se puede distinguir aquellos enfoques que contemplan una aplicación, un proceso de desarrollo, o ambas cosas. En cualquier caso, el desafío consiste en que, tanto las aplicaciones finales como aquellas herramientas que soportan el proceso de EUD, consideren fundamentalmente la movilidad del usuario, pero también la percepción de otros tipos de contexto gracias a los sensores disponibles en el dispositivo móvil, tales como la orientación, la posición, el nivel de luz o del sonido. Estudios como el de Li et al. [70] demuestran la creciente tendencia del EUD en el campo móvil.

2.4. Crowdsourcing

Durante años, individuos y organizaciones aprendieron que una manera de alcanzar sus objetivos en forma eficiente es mediante la externalización de tareas, comúnmente denominada «outsourcing». De esta manera, se delega una parte del negocio del interesado a un tercero especializado en la tarea a resolver, que permite tanto aumentar la eficiencia como abaratar costos [103]. Las razones para hacerlo pueden ser variadas: no posee el conocimiento especializado para llevarla a cabo, resulta económicamente conveniente, los tiempos de ejecución o la calidad del trabajo serán mejores, se estiman menores riesgos para su ejecución, el conjunto de tareas no representa la parte central del negocio, etc. Sin embargo, hacer outsourcing implica conocer a la entidad adecuada para llevar a cabo la tarea, y a veces no se dispone de algunos recursos, como el dinero para afrontar el monto que presupuestan, o esperar el tiempo que requieren para cumplir con lo solicitado.

Una forma similar de resolver este problema es mediante Crowdsourcing; la diferencia radica en que, en este último, las tareas a delegar pasan a manos de toda una comunidad en lugar de a un individuo u organización particular. En ambos casos, tanto el outsourcing como el crowdsourcing confluyen en la descentralización de las tareas.

El término Crowdsouring fue presentado en 2006 por Jeff Howe [65] y, desde entonces, han existido múltiples interpretaciones y clasificaciones del mismo. Existen definiciones más generales, como la de Doan et al. [37], quienes la plantean como un método para la resolución de problemas. En este, se recluta a una multitud de personas

para ayudar a resolver el problema definido por los propietarios del sistema. Bajo esta definición, es posible considerar a Wikipedia como una plataforma de Crowdsourcing, por presentarse como una solución en el que las masas pueden construir conjuntamente conocimiento. Sin embargo, otros autores no la consideran como tal.

Para [41], las iniciativas de Crowdsourcing deben contar con ciertas características, que no permiten considerar a Wikipedia como tal. Los autores llevaron a cabo un análisis en el cual ponen en común 40 definiciones encontradas tras analizar 209 documentos científicos (artículos de revistas, conferencias, workshops, etc.). La idea que prevalece sobre Crowdsourcing en dicho trabajo es la de un tipo de actividad online participativa en la cual una entidad propone a un grupo de personas, mediante una convocatoria abierta, la realización voluntaria de una tarea cuya realización implica un beneficio para ambas partes: el interesado obtiene una solución, y el trabajador una recompensa. En ella, el iniciador (o crowdsourcer) es cualquier entidad que pueda llevar adelante la iniciativa, y puede ser tanto un individuo como una organización. La multitud está conformada por un grupo de individuos (crowdworkers), de cantidad y conocimiento variable (la cantidad puede variar de acuerdo a la iniciativa y también al conocimiento requerido para completar las tareas solicitadas). Respecto a la tarea propuesta, debe ser difundida mediante una convocatoria abierta. Su complejidad puede variar, abarcando desde aquellas tareas de bajo esfuerzo cognitivo hasta aquellas de mayor nivel, o que requieren de cierto nivel de creatividad para su resolución. A cambio, los trabajadores de la multitud reciben una recompensa que puede interpretarse como un beneficio económico, un reconocimiento social, incremento en su autoestima, la oportunidad de desarrollar una habilidad, etc.

Respecto al uso de Internet como único medio para una iniciativa de crowdsourcing [41], el transcurso del tiempo trajo consigo nuevas alternativas de la mano de los dispositivos móviles. Lejos de encontrarse en correspondencia a la concepción más estricta de aplicación móvil, Eagle plantea en [38] un modelo de crowdsourcing basado en SMS, que sin hacer uso de la capacidad de movilidad simplemente soporta el proceso mediante plataformas móviles. Una gran cantidad de tareas pueden realizarse sobre canales de texto y voz, entre las que mencionan: transcripción, localización (traducción) de software, periodismo ciudadano y estudios de mercado.

Otros enfoques dentro del ámbito móvil permiten hacer uso tanto de la posición del usuario (ej. mCrowd [101], TruCentive [61]) como de la información de otros tipos de contexto, tales como el tiempo, el perfil o la actividad actual del usuario (ej. SensorCivico [97]).

Si bien el foco principal de esta obra no es el crowdsourcing, el enfoque fue concebido como soporte a tal fin; como parte de una plataforma mediante la cual algunos usuarios juegan el rol de crowdsourcers, quienes externalizan tareas proponiendo iniciativas para la resolución de un problema, y otros el rol de crowdworkers, que intentarán resolverlo.

2.5. MoWA framework

La constante evolución de la Web, junto a la disponibilidad masiva de tecnología impulsaron el desarrollo de aplicaciones Web sensibles al contexto en una gran diversidad de dominios, como el turismo, el marketing, la logística, el control del tráfico, etc. Diseñar este tipo de aplicaciones es una tarea compleja debido a la gran variedad de aspectos a considerar [63]. Por ejemplo, la cantidad o tipos de dispositivos a soportar, los tipos de contexto, sensores y tecnologías disponibles, el modelo del contexto, las capas de la arquitectura, etc. Como consecuencia, incorporar estas características como requisito de una aplicación existente conlleva un esfuerzo que no todos los propietarios están dispuestos a afrontar; mucho menos si esta funcionalidad representa una funcionalidad volátil [52] o un posible escenario para un número acotado de usuarios.

Muchas aplicaciones Web son actualmente accedidas desde navegadores Web móviles, pero no siempre hacen uso de la información de contexto para mejorar la experiencia del usuario. Por ejemplo, para adaptar la navegación o el contenido de un sitio en base a la posición del usuario, o para brindar asistencia en el recorrido de un conjunto de puntos de interés descritos en una página. Sin embargo, un posible camino para satisfacer estas necesidades no cubiertas viene de la mano del Crowdsourcing y la Aumentación Web Móvil.

MoWA (del inglés «**Mo**bile **W**eb **A**ugmentation») fue concebido en este marco, y representa una solución tanto para el desarrollo ágil de aplicaciones de Aumentación Web Móvil como para la delegación de tareas a una multitud de scripters bajo los principios del Crowdsourcing. Este enfoque, orientado a desarrolladores, les permite extender las características de un sitio Web mediante la creación de scripts de Aumentación Web que se ejecutan del lado del cliente [24][12]. Como soporte al mismo, se diseñó e implementó un framework (***MoWA framework***) y una herramienta (***MoWA weaver***) capaz de ejecutar los scripts (***aplicación MoWA***) resultantes sobre la Web; ambos, como parte de una plataforma de Crowdsourcing destinada a satisfacer la demanda de aplicaciones de Aumentación Web móvil.

El propósito fundamental de correr scripts sobre la Web es automatizar la ejecución de tareas sobre sus páginas, que de otra manera podrían realizarse manualmente. Los scripts de MoWA permiten automatizar tareas y adaptar la Web de acuerdo a los requisitos del usuario, no solo respecto a una página o aplicación concreta sino también permitiendo la integración de varias de ellas. De esta manera, una aplicación de aumentación Web móvil dentro del enfoque MoWA se construye sobre la base de una o varias aplicaciones Web, y puede responder a necesidades de diferentes dominios o a tecnologías asociadas. Por ejemplo, se podría aumentar la Web con comportamiento especializado para una guía turística y, a su vez, se la podría soportar mediante

diferentes tipos de aplicaciones, como ser de Hipermedia Móvil, Basadas en el Posicionamiento, Sensibles al Contexto, etc.

Bajo este enfoque, los desarrolladores pueden crear sus propias aumentaciones basados en los siguientes supuestos:

- Existe –al menos– una página Web a aumentar.
- Existe una contraparte física para la(s) Web a aumentar, en correspondencia con algún valor del contexto (ej. una posición)
- Existe la necesidad de agregar algo extra a la(s) página(s), ya sea en términos de comportamiento, contenido, estilo o estructura.
- Se cuenta con –al menos– un mecanismo de detección de valores de algún tipo de contexto.

Tal como se puede observar en la Figura 7, un *usuario final* hace uso de *MoWA weaver* para correr *aplicaciones* que un *desarrollador* creó mediante scripting, haciendo uso de *MoWA framework*. Estas aplicaciones son expresadas en JavaScript y se ejecutan sobre la capa de presentación de *páginas Web existentes*, una vez cargadas en el navegador. Es decir, que no existen de forma independiente, sino que dependen de los componentes de la interfaz gráfica de otra aplicación. Dichos scripts tienen como finalidad manipular el DOM de una aplicación Web –producir una aumentación y crear una *nueva experiencia*– de acuerdo con cambios en la información del contexto del usuario. Por ejemplo, incorporando tráileres y videos relacionados a los artículos de Wikipedia solo cuando el usuario se encuentra en su casa, o modificando el color de fondo y de fuente de cualquier página Web por colores completamente opuestos, para mejorar la legibilidad del texto cuando el usuario se encuentra en entornos con mucha iluminación (por ejemplo, mediodía bajo el rayo del sol).

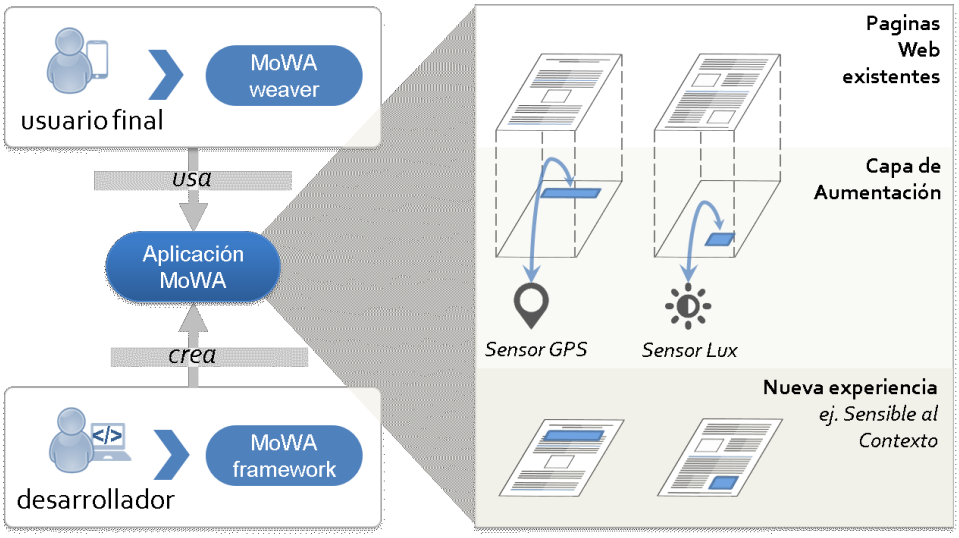

Figura 7. El enfoque MoWA soportando desarrollo mediante scripting

El framework permite contemplar diferentes tipos de aplicaciones de aumentación

móvil y, a modo de demostrar su factibilidad como así también su flexibilidad y la reutilización de sus componentes, se detallaron dos escenarios con diferentes tipos de aplicaciones [12]:

- Una aplicación de Hipermedia Móvil. En este escenario se mostró cómo un desarrollador puede dar soporte al recorrido de un Museo, tomando la información existente de cada pieza tanto en Wikipedia como en el sitio oficial del museo, y la ubicación física de cada una de ellas. Se mostró cómo dar soporte a la integración del contenido de ambas fuentes y, además, a la asistencia en el recorrido de todas las piezas. Para ello, se extendió una aplicación del framework, se asignó una correspondencia entre las ubicaciones físicas de cada pieza y los marcadores visuales en el mapa del museo, y se definió una aumentación por cada página Web a aumentar (como resultado de la combinación de diferentes augmenters).
- Una aplicación Sensible al Contexto. Este escenario, también pensado dentro del mismo dominio de aplicaciones turísticas, difiere del anterior en el propósito: mientras que el primero busca asistir al usuario en el recorrido, este busca personalizar la experiencia de acuerdo a su perfil. La idea es extender el framework con una aplicación que permita aumentar las mismas piezas del museo, pero esta vez no solo en relación con la ubicación del usuario, sino también a su perfil y a la hora actual. Las aumentaciones deben mostrarse en relación con el PDI que el usuario está visitando, pero el contenido a mostrar se adecúa a los siguientes perfiles de usuario: público en general, investigador o kinder. Por ejemplo, un usuario con un perfil «público general» verá la información oficial proporcionada por el museo en su sitio oficial, mientras que un usuario «investigador» verá, adicionalmente, una sección con artículos académicos relacionados a la pieza visitada. Finalmente, los visitantes bajo el perfil «kinder» accederán a contenido multimedia, evitando descripciones textuales.

En cualquier caso, tal como se muestra en la Figura 8, crear un script de aumentación con MoWA implica extender una aplicación (líneas 4 y 5), definir un conjunto de contrapartes digitales a aumentar (líneas 11-24) relacionadas a un *valor del contento de interés* (en este caso, PDI. Líneas 13-15) y a una determinada aumentación (líneas 25-29). En el caso de la figura, la aumentación es una sola, genérica, asociada al método que se ejecuta cada vez que un PDI concreto es detectado. Por ejemplo, el de macrauchenia (líneas 30-32) o glyptodon (línea 33). Cabe aclarar que las aplicaciones pueden suscribirse a diferentes sensores de tipos de contexto, aunque en este ejemplo, la subclase de la aplicación de Hipermedia Móvil solo hace uso de la ubicación.

```
1  // @name        DarwinTour
2  // @namespace   dev01
3  // ...
4  function DarwinTour() {...};
5  DarwinTour.prototype = new AbstractMobileHypermedia();
6  DarwinTour.prototype.initialize = function(){
7      this.createPois();
8      this.create2DMapRepresentation();
9      this.useQrCodePositioning();
10 }
11 DarwinTour.prototype.createPois = function(){
12     var name = "Macrauchenia";
13     var url = "https://es.m.wikipedia.org/wiki/Macrauchenia";
14     var location = new QRLocation(url);
15     var macrauchenia = this.createConcretePoi(name, url, location);
16     macrauchenia.createProperty("image", "http://www.fcnym.../macrauquenia.jpg");
17     macrauquenia.createPropertyFrom("description", "http://www.fcnym.../pieza1.html",
18         "//*[@id='contenido']");
19     ...
20     var glyptodon = this.createConcretePoi(..., ..., ...);
21     ...
22     this.createWalkingLink(macrauchenia, glyptodon);
23     ...
24 }
25 DarwinTour.prototype.commonAugmentation = function(poi, dom, xpath){
26     var params = {"poi": poi, "dom": dom, "elemPosition":xpath};
27     this.augmenters["mowa:WalkingLinksAugmenter"].execute(params);
28     this.augmenters["mowa:PoiAugmenter"].execute(params);
29 }
30 DarwinTour.prototype.macrauchenia = function(poi, dom){
31     this.commonAugmentation(poi, dom, "//*[@id='section_0']");
32 }
33 DarwinTour.prototype.glyptodon = function(poi, dom){...}
34 ...
35 var myApplication = new DarwinTour();
```

Figura 8. Un script que extiende e instancia una aplicación MoWA

Los scripts desarrollados con MoWA pueden ser instalados mediante la extensión *MoWA weaver* por los usuarios finales. Una vez instalado y habilitado, un script será ejecutado sobre los sitios cuya contraparte digital se ha indicado por cada valor del contexto de interés (línea 13 de la Figura 8). Cuando una aplicación es notificada sobre un cambio de valor en el contexto, la misma chequea que aquel valor detectado se corresponda con alguno de los definidos en la aplicación, y ejecuta la aumentación correspondiente.

El framework presenta algunos puntos de variabilidad que permite a los desarrolladores reutilizar los artefactos de MoWA y su funcionalidad para soportar diferentes contextos de uso. Es posible extender aplicaciones, tipos de contexto, augmenters, representaciones del espacio, sensores. Extender alguno de ellos requiere más esfuerzo que otros: no sólo por la complejidad que requieren individualmente, sino porque algunos de ellos dependen o están compuestos por otras clases. Por ejemplo, soportar un nuevo tipo de contexto también requiere soportar un nuevo sensor.

El enfoque fue sometido a un minucioso análisis de trabajos relacionados, en el que no sólo se verificó la inexistencia de herramientas de propósito similares, sino también la presencia de catorce características que hacen a este enfoque único. Inicialmente este análisis estuvo enfocado en encontrar herramientas alternativas de características similares contra las cuales comparar resultados al momento de llevar a cabo un experimento, pero finalmente no se encontraron herramientas que permitan construir

una experiencia similar o que faciliten su desarrollo.

De esta manera, la evaluación inicial del enfoque demostró que el mismo es viable y que MoWA facilita el desarrollo de aumentaciones móviles en términos de eficiencia [24]. Se reclutó un total de 16 participantes, que fueron divididos en dos grupos de igual cantidad de participantes; ambos resolvieron un mismo problema, pero mientras el «Grupo 1» lo hizo con técnicas tradicionales, el «Grupo 2» lo hizo mediante el uso de nuestro framework. La tarea consistió en desarrollar una aplicación para realizar un tour en un museo, cumpliendo los siguientes requisitos:

1. Contemplar la información provista para cada pieza sobre Wikipedia (puesto que esta información es la realmente ofrecida a los usuarios in-situ, mediante códigos QrPedia) y sobre el sitio oficial del museo.
2. Soportar Walking Links para mostrar a los usuarios el recorrido a la siguiente pieza a visitar.
3. Soportar la actualización y corrección del recorrido, cuando el usuario llega a una pieza equivocada.

Con este experimento se demostró no solo la factibilidad del enfoque sino también –y principalmente– su eficiencia. Solo 4 participantes del «Grupo 1» pudieron terminar la aplicación cumpliendo los tres requisitos; los otros 4 participantes no cumplieron con ninguno de los requisitos. En contraste, 7 de los 8 participantes del «Grupo 2» terminaron su aplicación cumpliendo todos los requisitos, mientras que el participante restante terminó su aplicación sin cumplir uno solo de los requisitos. Respecto a la productividad, los usuarios del framework MoWA también obtuvieron mejores resultados; el tiempo promedio para el «Grupo 1» fue de 18:52h con una desviación estándar de 7:28h, mientras que para el «Grupo 2» fue de 5:17h con una desviación de 3h.

Al final del proceso, los participantes del «Grupo 2» completaron un cuestionario SUS [20] acerca del uso de MoWA y sus resultados fueron favorables, obteniendo un promedio de 81.3 puntos. Los participantes también dieron feedback acerca de la experiencia y, entre los comentarios más generalizados, se encontró la siguiente observación: ellos percibieron la etapa de definición de PDIs como la más fastidiosa del proceso de desarrollo. Uno de los participantes sugirió alguna forma de automatización de ese proceso.

Teniendo en cuenta los resultados de dicha experiencia, se comenzó a diseñar una alternativa que permita facilitar y automatizar el trabajo de los desarrolladores mediante herramientas visuales que luego les permita generar código fuente que posteriormente puedan manipular. En esta dirección nace *MoWA Authoring*; tema central de esta obra.

Cabe mencionar un segundo problema respecto al enfoque inicial: la falta de medios

para que los usuarios finales puedan crear sus propias soluciones de Aumentación Web Móvil, sin importar el dominio que resuelven. MoWA fue concebido como parte de una plataforma de Crowdsourcing, en donde aquellos usuarios con el conocimiento para hacer scripting (*desarrolladores*) pueden ayudar a quienes no lo poseen (*consumidores*) a resolver problemas de escenarios concretos. Las aplicaciones resultantes pueden ser creadas en respuesta a una solicitud de un usuario dentro de la plataforma o simplemente compartidas por el desarrollador que las creó, para luego ser utilizadas por otros usuarios bajo el mismo escenario para el cual fueron desarrolladas.

Si solo la gente con el conocimiento técnico puede desarrollar aplicaciones, los *consumidores* dependen de la disponibilidad de los desarrolladores para obtener nuevas soluciones o incluso para el mantenimiento de las existentes. Esto último representa un importante problema para la WA en general, puesto que las aumentaciones están sujetas a la variación de las aplicaciones Web de base, resultantes de la incorporación de nueva tecnología, contextos de uso, nuevos requisitos, el cambio en la estructura del contenido, o en el nombre de las clases de estilo, etc. Cuando estos cambios ocurren, es necesario que alguien actualice, al menos, las expresiones de las cuales depende el posicionamiento de los elementos de la aumentación.

3. Trabajos relacionados

Identificar un problema y formular preguntas de investigación requiere de un amplio dominio del tema de investigación, y ello no involucra simplemente una base conceptual sino también saber de qué maneras ya ha sido previamente abordado. En esta sección se presenta un conjunto de trabajos que permiten alcanzar parcialmente los objetivos planteados en esta tesis.

Este trabajo está orientado a permitir que los usuarios finales sean capaces de aumentar sitios Web existentes con características móviles, de acuerdo a sus propios requerimientos y desde sus propios dispositivos móviles con herramientas visuales de programación por usuario final. Se utilizan móviles para sacar un máximo partido a la movilidad del usuario y a la percepción del contexto durante el proceso de construcción, sin embargo, no se descarta la posibilidad de realizar la construcción desde un dispositivo estacionario.

En la Subsección 3.1 se abordan aquellos trabajos que proponen aplicaciones sensibles al contexto que resuelven problemas dentro de un dominio concreto. Luego, la Subsección 3.2 presenta aquellos que permiten el desarrollo por usuarios finales de este tipo de aplicaciones mediante una técnica visual. Posteriormente, se presentan aquellos que contemplan el desarrollo de aplicaciones Web adaptables; en la Subsección 3.3 mediante scripting y haciendo uso de un framework, mientras que en la Subsección 3.4 mediante el uso de herramientas de EUD. Finalmente, la Subsección 3.5 presenta trabajos con relación al Crowdsourcing de aplicaciones Web y/o móviles.

3.1. Aplicaciones Sensibles al Contexto

Dentro del campo de las aplicaciones móviles, las Guías Móviles [39] a menudo hacen uso de la ubicación del usuario como tipo de contexto para proveer asistencia al usuario en el recorrido de un espacio físico. En el marco del proyecto Cyberguide [1], se crearon un conjunto de prototipos de Guías Móviles Sensibles al Contexto, con la finalidad de capturar la esencia de una aplicación de este tipo y proponer componentes para una arquitectura genérica. Estas guías hacen uso de la ubicación, la orientación y el perfil del usuario como tipos de contextos. Schaller propone un sistema para la planificación de experiencias de viajes personalizadas [88], donde la creación de un recorrido inicial puede ser posteriormente modificada durante la ejecución del mismo. Etaati y Sundaram presentan un sistema turístico adaptativo [42] que ofrece recomendaciones de contenido a sus usuarios de acuerdo a características que monitorea, como el estado del recorrido, preferencias del usuario desde otras aplicaciones, valores del contexto, etc. En todos los casos, estos enfoques: son orientados al dominio del turismo, no contemplan técnicas de Aumentación Web y sus aplicaciones resultantes no son *aplicaciones Web puras*.

Espada et al. [40] presentan un enfoque que permite a las aplicaciones Web acceder a la información del contexto –que el dispositivo puede detectar– mediante un set de etiquetas XML extensibles. Los mismos pueden ser especificados en la capa de presentación para solicitar información de algún valor del contexto. Para que estas etiquetas sean correctamente interpretadas, desarrollaron un navegador Web Sensible al Contexto. El mismo pide permiso al usuario para acceder a los sensores específicos y ejecutar las tareas correspondientes. Sin embargo, este trabajo requiere modificar el código fuente de la aplicación que va a incorporar este comportamiento, y no puede ser presentado por cualquier navegador Web móvil; dependen de la implementación de su navegador concreto.

Los autores de [99][100] presentan un enfoque para mejorar sitios Web existentes con características del contexto *on-the-fly*. El enfoque está soportado por SCOUT, un framework para aplicaciones Android con el que crearon una aplicación como prueba de concepto, a la que llamaron COIN. Esta aplicación extrae información semántica de un sitio Web (puede ser de terceros) y la compara con los valores actuales del contexto del usuario, para luego enriquecer (mediante técnicas no intrusivas) cualquier página Web con características Sensibles al Contexto mediante la manipulación de su DOM. Por ejemplo, permite ofrecer recomendaciones o información basada en la posición del usuario. Mientras SCOUT [99] está enfocado en desarrolladores, COIN [100] es simplemente una aplicación final creada con dicho framework. La misma explota la información semántica extraída de los sitios Web (anotaciones RDFa) y aumenta las páginas Web en relación a las entidades detectadas en el contexto del usuario. Si bien la adaptación surge ante interacciones desde el dispositivo móvil del usuario (cliente), la extracción de información, la vinculación y la adaptación de una página se resuelven del lado del servidor.

Existen también aplicaciones que aumentan sitios Web desde dispositivos móviles [22][55][100] pero que carecen de soporte para el EUD. Ninguna de ellas lleva a cabo la aumentación como una extensión de un navegador Web móvil, del lado del cliente, y su funcionamiento depende de componentes nativos, por lo que tampoco pueden ser consideradas *aplicaciones Web puras*.

3.2. Aplicaciones producidas por usuarios finales

En los últimos años, muchas herramientas soportando EUD [71] han sido desarrolladas para permitir a los usuarios finales crear diferentes tipos de aplicaciones. En esta sección se presentan este tipo de herramientas, con especial énfasis en la producción de aplicaciones Web, divididas entre aquellas cuyo entorno de producción es estacionario o móvil.

3.2.1. EUD desde entornos estacionarios

En relación a la integración de contenido, CAMUS [29] es un enfoque para la

composición de mashups soportado por un framework para el desarrollo de aplicaciones Web móviles Sensibles al Contexto. Las aplicaciones resultantes integran recursos Web de acuerdo con el contexto de uso; de esta manera, el contenido de la aplicación generada no es pre-empaquetado sino dinámico. Los recursos son registrados por un administrador del CDT Universal de la plataforma, para que luego los diseñadores puedan hacer uso de estos desde una aplicación específica. Sin embargo, CAMUS no está orientada a aumentar la Web, precisa de componentes del lado del servidor y soporta un solo tipo de aplicación (mashups). Respecto a la independencia de componentes nativos, los autores mencionan que los motores de ejecución de estos mashups son aplicaciones nativas desarrolladas para diferentes plataformas móviles.

También existe una herramienta de autoría que combina mashups con Realidad Aumentada [102], y que permite a sus usuarios crear aplicaciones tanto para interiores como para exteriores. La misma es una aplicación móvil nativa desde la cual el usuario puede seleccionar un área física determinada tomando varias fotos o grabando un video. En cualquier caso, un conjunto de imágenes es analizado posteriormente por un componente del lado del servidor con la finalidad de extraer características 2D e identificar escenas. Tras este procesamiento, los usuarios pueden comenzar a asociar recursos Web sobre la zona identificada, asignándoles una posición, un tamaño, una rotación y una frecuencia de actualización (para refrescar el contenido desde el servidor). La diferencia principal con MoWA reside en el alcance del tipo de aplicaciones que es posible crear; MoWA propone una solución genérica, extensible a una amplia gama de tipos de aplicaciones y dominios. Además, la herramienta de mashups [102] permite utilizar fuentes de datos existentes, mientras que con MoWA es posible definir propiedades de una contraparte digital que incluso pueden ser extraídas de otros sitios Web y sin necesidad de contar con un servicio para su consumo. De esta manera, si bien no se está creando un servicio reutilizable desde cualquier aplicación (o aumentación) Web, es posible consumir información desde la interfaz visual de cualquier página Web.

Cappiello et al. [21] presentan una aplicación Web que permite a sus usuarios finales crear mashups para múltiples dispositivos mediante la composición de recursos, utilizando la técnica WYSIWYG. Los usuarios deben seleccionar y combinar componentes de datos y plantillas de UI preexistentes en un repositorio para, finalmente, obtener un esquema (independiente de la plataforma) que es automáticamente generado y almacenado en el repositorio. Posteriormente, el usuario puede acceder a este repositorio, descargar su solución y ejecutarla desde una aplicación nativa, aunque disponible para múltiples plataformas.

Manjunath et al. presentan un enfoque en el cual los usuarios finales crean sus propios widgets, los cuales se definen como aplicaciones muy simples que representan una interacción Web específica. Estos artefactos, llamados Tasklets [72], son creados

utilizando PBE y pueden ser grabados mediante una extensión de navegador Web que, además, detecta y define posibles parámetros para que la misma pueda reutilizarse en escenarios similares con diferentes valores. La diferencia con MoWA, en este caso, es que este enfoque se centra en la automatización de tareas, es decir, comportamiento, mientras que con MoWA se puede adaptar una aplicación con relación al estilo, estructura o contenido, y sin necesidad de grabar acciones para poder conseguir un determinado resultado.

3.2.2. EUD desde entornos móviles

Algunas plataformas permiten llevar a cabo el proceso de autoría desde entornos móviles [32][73][84][91], pero ninguno de ellos contempla la creación o aumentación de *aplicaciones Web puras*, ya que dependen de algún componente nativo o híbrido.

Puzzle [32] es un framework de EUD para producir aplicaciones nativas, aunque basadas en Web, que corren sobre plataformas táctiles. Está orientado a usuarios finales sin habilidades de programación. Los mismos deben combinar bloques de construcción a través de una metáfora basada en un rompecabezas, con piezas cuyas esquinas de colores indican una posible combinación. Se pueden crear diversas combinaciones, obteniendo soluciones para múltiples propósitos. La herramienta fue evaluada con 13 participantes que no trabajaban en relación a las Tecnologías de la Información, y que no fueron expuestos a entrenamiento previo. Los autores mencionan que no hay necesidad de plugins para acceder a la aplicación Web generada, pero puede apreciarse que la implementación que presentan en su publicación se trata de una aplicación Android equipada de un visualizador de HTML. Esto sugiere que la aplicación no podría ser accedida desde un navegador Web convencional, como lo hace MoWA.

Martín et al. [73] proponen una plataforma de servicios móviles sensibles al contexto, que pueden ser consumidos desde aplicaciones nativas. Dicha plataforma es accesible desde una aplicación basada en tecnologías Web, donde el *productor* puede asociar un conjunto de valores del contexto (ubicaciones, áreas, horas, fechas, etc.) con información concreta que se entregará solo a los clientes que cumplan con dichas condiciones. Los autores realizaron un experimento con 10 expertos en el dominio del turismo, pero sin conocimientos técnicos ni formación previa en el uso de la herramienta. Lo hicieron a partir de una aplicación nativa preinstalada en sus dispositivos móviles. En contraste a MoWA, las aplicaciones resultantes están limitadas al suministro de información.

CASTOR [84] es una plataforma para crear, administrar y ejecutar historias *in-situ* con Sensibilidad al Contexto. La característica *in-situ* permite que las historias sean creadas en el mismo sitio en el que luego serán relatadas. La misma está conformada por dos aplicaciones móviles híbridas que, del lado del cliente, permiten la creación y presentación de las historias, y una aplicación Web para la administración de las que ya han sido creadas. Las aplicaciones híbridas fueron soportadas por el framework

Phonegap. El proceso de autoría es asistido mediante formularios, que permiten al usuario seleccionar la estructura de la historia (simple, secuencial o encrucijada), definir sus etapas y los valores del contexto ante los cuales se presentará la historia o alguno de sus fragmentos. Las historias que se pueden crear con esta plataforma pueden cubrir diferentes campos, como la historia o la literatura, pero el propósito es siempre el mismo: la narración de historias. Una vez creadas, los usuarios pueden compartir sus producciones mediante un repositorio compartido. La herramienta demostró ser efectiva tras evaluarla con 19 alumnos de una escuela primaria que no tenían conocimiento en programación.

Seifert et al. presentan Mobidev [91], una plataforma de desarrollo para la creación de aplicaciones móviles a partir de dispositivos móviles. La misma está basada en Android y permite al usuario crear la interfaz gráfica de una aplicación escribiendo código fuente, diseñando maquetas con un editor visual o también dibujando un mockup sobre papel y tomándole una foto, para que luego el sistema lo analice e interprete, y genere un diseño manipulable desde el editor visual. El enfoque no solo contempla a usuarios finales, sin habilidades de programación, desarrollando aplicaciones con un flujo de control básico, sino también a desarrolladores que definen un comportamiento más específico a través de código JavaScript. El enfoque fue evaluado con 16 estudiantes pertenecientes al departamento de Ciencias de la Computación, previamente entrenados en el uso de la herramienta. El experimento tuvo éxito, pero a diferencia de MoWA, sus requisitos no contemplaron el uso de características móviles.

Francese et al. [53] presentan MicroApps, una plataforma móvil nativa para la composición de aplicaciones de la misma naturaleza. Sus usuarios pueden integrar características móviles provistas mediante la información recuperada por los sensores, pero también de servicios Web. Los usuarios especifican actividades mediante componentes visuales que luego son interpretados por un intérprete que genera automáticamente la UI de la aplicación. El enfoque incluye un repositorio para permitir a los usuarios finales compartir sus producciones y volver a configurarlas. El equipo evaluó su enfoque con 40 estudiantes (sin habilidades de programación) del primer año de la carrera de Ciencias de la Computación.

MyService [69] es un sistema que permite al usuario construir mashups en base a la definición de reglas Sensibles al Contexto y la selección de servicios Web listados en un directorio. El sistema fue implementado como una aplicación Android y el proceso de construcción se lleva a cabo mediante una interfaz gráfica, basada en formularios. El sistema detecta valores del contexto y selecciona, en relación a ellos, aquellos servicios del directorio pueden ser utilizados en la composición del mashup. Finalmente, el sistema genera el código del mashup resultante como una aplicación Web móvil.

3.3. Frameworks para la Adaptación de aplicaciones Web

Tal como se mencionó en la Sección 2.1, existe una necesidad de hacer que la Web sea adaptable de acuerdo a los requerimientos de sus usuarios [6][17][35]. Esto puede implicar no solo el modelado del usuario y sus preferencias, sino también del contexto. Ceri et al. presentan un enfoque dirigido por modelos soportando Sensibilidad activa al Contexto [23]. Los autores contemplan al contexto como un actor activo que puede disparar adaptaciones automáticamente sobre la Web, ante cambios en el contexto sensado y sin intervención del usuario. El framework que soporta este enfoque es resultado de una extensión de WebML[38] pero, aunque este enfoque está basado en algunos componentes visuales para el desarrollo de aplicaciones, no contempla técnicas de Programación por Usuarios Finales.

Otra manera de ofrecer adaptación es mediante la personalización del contenido. D'Amico et al. [31] proponen un framework y un sistema de soporte para el turismo inteligente que permite al usuario construir su propia visita individual a partir de una lista genérica de PDIs, y la posibilidad de hacer cambios en el plan de visita durante su recorrido. Este sistema tampoco podría considerarse de EUD, puesto que los usuarios no pueden resolver otros problemas más que el de la falta de personalización de recorridos y no pueden incorporar nuevos PDI o funcionalidad a su recorrido. Los usuarios deben utilizar solo los PDIs ya existentes en el entorno y solo pueden configurar la aplicación funcionar, de la misma manera, con unos u otros, pero no pudiendo incorporar nuevo comportamiento.

Ghiani et al. [55] presentan una plataforma para la aumentación de aplicaciones existentes –de acuerdo al contexto de uso– con capacidades de interacción multimodal. Los autores contemplan la capacidad de adaptación de cualquier página ante cambios en las condiciones del contexto, pero a diferencia de MoWA, este enfoque está orientado a facilitar nuevas formas de interacción y no nuevo comportamiento específico de múltiples dominios o la integración de contenido de diversas fuentes.

En relación a la personalización, Wasinger et al. [98] presentan PersonisJ, un framework para la personalización de aplicaciones sobre móviles (Android). La idea es mantener un modelo del usuario del lado del cliente con alguno de sus componentes del lado del servidor, y que las aplicaciones construidas accedan a información del modelo declarando permisos de seguridad que les permitan ejecutar determinadas operaciones: read/write/tell. Este modelo está compuesto por una estructura jerárquica de características del contexto, que son actualizadas por un *listener*. Sin embargo, esta solución no resuelve todo del lado del cliente; existen algunos componentes del lado del servidor. De hecho, la decisión de mantener el modelo del usuario del lado del

[38] http://webml.deib.polimi.it

cliente es para generar confianza respecto a la privacidad de los datos del usuario.

Un enfoque más reciente es Ambient AMP [22]; sus autores lo presentan como un framework para aplicaciones Sensibles al Contexto que, además, tienen la capacidad de interactuar con objetos inteligentes. Estas aplicaciones corren sobre un navegador que ellos mismos desarrollaron, que hace posible tanto el uso de las características del contexto como también la interacción con los objetos. Por ejemplo, los autores muestran cómo aumentar el sitio de Flickr con la capacidad de compartir el contenido multimedia de la página actual con otros dispositivos que se encuentran próximos al usuario. Sin embargo, este enfoque carece de soporte al EUD y la aumentación depende de componentes nativos.

Por último, existe un enfoque que contempla la creación de aumentaciones, pero sin considerar características móviles durante el proceso de desarrollo ni tampoco como parte de sus aplicaciones resultantes. Díaz et al. [36] proponen un Lenguaje Específico de Dominio (DSL, del inglés «Domain Specific Language») textual para el desarrollo de aumentaciones que abstrae la complejidad y la comprensión de las estructuras de programación. Esto último no solo está orientado a elevar el nivel de abstracción favoreciendo a quien escribe los scripts, sino también a proporcionar confianza de quienes los consumen, ya que pueden leer e interpretarlos fácilmente. La construcción está basada en una metáfora que consiste en concebir a cada aplicación Web como una pared que puede ser decorada con stickers, que representan fragmentos de contenido externos a la página. A una unidad de aumentación la llaman Sticklet, y si bien se pueden crear para un amplio espectro de propósitos, el trabajo no menciona aspectos relacionados a la movilidad, la Sensibilidad al Contexto ni al desarrollo visual o *in-situ*.

3.4. Aumentación Web por usuarios finales

Existen diversos enfoques de Adaptación Web soportados por técnicas Desarrollo por Usuarios Finales [3]. Sin embargo, ninguno de ellos contempla la creación de aumentaciones móviles y un proceso de desarrollo realizable desde este tipo de dispositivos.

Tal como se mencionó en la sección 3.3, Sticklet [36] propone un lenguaje para el desarrollo de aplicaciones de Aumentación Web mediante scripting que abstrae la complejidad y facilita la comprensión de los mismos. Además de las diferencias previamente resaltadas en dicha sección, respecto al EUD, este enfoque fue evaluado con personas que representan un sector reducido de usuarios finales; aquellos con conocimiento en programación (estudiantes universitarios de Ciencias de la Computación). MoWA, en contraste, fue evaluado con un espectro más amplio de participantes provenientes de diferentes áreas, no solo informáticos.

Web Objects Ambient [15] es un enfoque para la abstracción y estructuración de

contenido Web. El mismo cuenta con una herramienta de soporte que permite a los usuarios crear/extraer contenido en forma de objetos que puedan ser manipulados y decorados en función del tipo de aplicación que se quiera crear. Esto promueve la integración de objetos provenientes de diferentes orígenes y la reutilización de contenido para distintos propósitos. Sin embargo, no se contemplan aumentaciones «para» ni «desde» dispositivos móviles.

Mediante un mecanismo similar de abstracción de contenido, existe un enfoque para adaptar la Web con nuevas formas de interacción para la Distribución de Interfaces de Usuario [47]. En dicho enfoque se contemplan plataformas móviles, pero limitando las aumentaciones posibles a un tipo concreto: aquellas que soporten un nuevo mecanismo de distribución.

WebMakeUp [34] es una herramienta que permite la personalización de contenido Web de manera visual. Mediante un click en cualquier elemento del DOM, el mismo se convierte en un widget cuyo comportamiento permite reorganizarlo, eliminarlo, actualizarlo o almacenarlo para su reutilización posterior. Además, el enfoque permite adaptar de acuerdo con interacciones, y las mismas pueden afectar no solo al widget involucrado sino también a aquellos relacionados al mismo. En relación con MoWA, el enfoque no contempla la información del contexto físico y la herramienta de soporte fue desarrollada para la versión desktop del navegador.

3.5. Crowdsourcing de aplicaciones Web

En relación al Crowdsourcing, Korthaus & Dai [67] presentan un enfoque que permite a los crowdsourcers definir y ofrecer tareas personalizadas y sensibles al contexto. Es decir, pueden hacerlo en relación con sus intereses, habilidades, disponibilidad, conectividad, etc. SensoRcivico [97] es sistema similar, pero orientado a la participación cívica ciudadana. Mediante una interacción top-down, permite a la administración pública enviar encuestas o proponer tareas a los ciudadanos; tal como se menciona en su trabajo, se trata de entregar las tareas adecuadas a las personas adecuadas en las circunstancias adecuadas. La plataforma intenta presentar a los trabajadores sólo aquellas tareas relevantes de acuerdo a los valores de sus contextos. La aplicación cliente instalada en los dispositivos de los trabajadores descarga periódicamente una lista de tareas a realizar, pero solo notifica al usuario sobre aquellas cuyos valores esperados del contexto coinciden con los del contexto actual del usuario (posición geográfica, el tiempo, el perfil o la actividad actual del usuario, etc.).

mCrowd [101] es una plataforma que permite a sus trabajadores utilizar los sensores del móvil para resolver micro-tareas como obtención de imágenes (ej. de un lago) y consultas basadas en posicionamiento (ej. cuál es el supermercado más cercano a una ubicación determinada). TruCentive [61] permite que los trabajadores contribuyan en la plataforma con información sobre la disponibilidad de estacionamiento en un lugar determinado, en relación a un lugar especificado por un iniciador (o *consumidor*).

Cuando un trabajador reporta un estacionamiento libre, la aplicación obtiene automáticamente su posición y la hora en la que fue reportada, y esta información es ofrecida al *consumidor*.

Los enfoques mencionados anteriormente demuestran que los usuarios están dispuestos a participar en la creación de tareas o en la recolección de datos, aunque no contemplan a usuarios finales adaptando una aplicación Web de acuerdo a sus propias necesidades, sino más bien a una aplicación tomando la decisión de adaptar su contenido de acuerdo a la información de uno u otro tipo de contexto.

En relación a la adaptación, Firmenich et al. [48] proponen un enfoque que permite a los usuarios finales crear sus propias adaptaciones, desde un entorno estacionario, soportando sus tareas mientras se encuentran navegando diferentes aplicaciones Web. Si bien el foco principal de este trabajo no está puesto en el crowdsourcing, el enfoque es concebido como soporte a tal fin y contempla como trabajo futuro la implementación de una plataforma de Crowdsourcing.

Nebeling y Norrie [78] presentan un enfoque para la creación y recomendación de adaptaciones Web creadas por usuarios finales. En él, los desarrolladores de una aplicación Web proveen una interfaz en base a la cual los usuarios finales pueden especificar adaptaciones que se construyen mediante una herramienta visual del lado del cliente. Entre las operaciones soportadas se encuentran las de mover, esconder, redimensionar, espaciar, cambiar el tamaño de la fuente de los elementos y modificar el layout de elementos repetitivos. Un componente remoto, del lado del servidor, provee recomendaciones sobre qué operaciones de adaptación pueden aplicarse a una página Web de acuerdo a la información del contexto de uso que recibe del cliente, como las características de un dispositivo y las preferencias del usuario. Una vez construidas, las adaptaciones son almacenadas en un repositorio donde quedan expuestas a una revisión multinivel y a un sistema de calificaciones, para finalmente pasar a estar disponibles para otros usuarios. El enfoque persigue la mejora de la experiencia del usuario mediante adaptación de los elementos visuales de la aplicación, no de su comportamiento, como si lo hace MoWA.

Arellano et al. presentan Modding Interfaces [5], un enfoque que contempla la definición de interfaces que regulan la comunicación entre el código de una aplicación Web y el de una aumentación, con el fin de permitir que las aplicaciones Web sean aumentadas en forma segura. En él, existen scripters jugando también el rol de crowdworkers, que realizan aumentaciones suscriptas a eventos conceptuales que conforman la interfaz. Esta última es definida por los desarrolladores de la aplicación, quienes establecen qué parte de la aplicación es aumentable. Los scripts son almacenados en el servidor (en el mismo contexto que el de la aplicación host) y ofrecidos desde ella, por lo que sus creadores tienen la posibilidad de adquirir la notoriedad de una mayor cantidad de usuarios de la aplicación.

4. Identificación del problema y preguntas de investigación

La observación es fundamental en la investigación; se trata de examinar atentamente los hechos y fenómenos que tienen lugar en la realidad para poder cuestionarla y detectar problemas a resolver. Habiendo elegido un tema sobre el cual investigar, estudiado los fundamentos del mismo y habiendo analizado los trabajos existentes en relación a él, el objetivo de este capítulo es presentar el problema identificado que se pretende resolver en esta obra.

Según lo desarrollado en los capítulos precedentes, es posible afirmar que la Aumentación Web responde a la necesidad de adaptación de la Web, pues una misma aplicación Web no puede cubrir todas las expectativas de todos sus usuarios. Lo confirma la existencia de grandes comunidades que crean y mantienen un conjunto cada vez más numeroso de scripts de aumentación. Mediante esta técnica, es posible manipular el contenido, estilo, comportamiento y/o estructura de una aplicación Web en pos de mejorar la experiencia del usuario de acuerdo a sus requisitos. Estos requisitos muchas veces pueden depender de a información del contexto. Abowd et al [2] han abordado previamente la importancia de su obtención mediante medios automáticos y su posterior interpretación con el fin de adaptar una aplicación.

Además, es posible afirmar que existe una tendencia por parte de los usuarios finales a crear sus propias soluciones, y dentro de las herramientas que utilizan con tal fin se encuentra un subconjunto de aquellas que permiten adaptar la Web. Tanto las aplicaciones resultantes de este proceso como las herramientas utilizadas para su creación pueden ser aplicaciones nativas, Web, o Web móvil. Muy pocas de estas herramientas son independientes de una contraparte del lado del servidor.

Actualmente, tanto las aplicaciones Web como las extensiones de navegador pueden acceder a valores de diferentes tipos de contexto (como la ubicación, la orientación, el nivel de batería, etc.) de acuerdo a los sensores del dispositivo que el navegador observe. En la misma dirección, la capacidad de los recursos de un dispositivo móvil y el número de usuarios a nivel mundial ha crecido en los últimos años.

MoWA surgió en respuesta a la necesidad de Adaptación Web, al beneficio de la Sensibilidad al contexto y al crecimiento de la tecnología móvil. Sin embargo, deja de lado a aquellas personas que no poseen conocimiento en programación. Pese a todos los avances mencionados, aquellos usuarios finales sin conocimiento en programación no poseen aún los medios para crear aplicaciones de WA que les permita mejorar sus experiencias con características del contexto y desde sus propios dispositivos móviles, permitiéndoles ser los creadores de sus propias soluciones en diversos escenarios específicos de un dominio, tanto fijos como oportunos, distantes o bajo una modalidad *in-situ*. Esta situación es contradictoria a las posibilidades y tendencias actuales, y requiere de una solución innovadora.

Las preguntas de investigación que surgen a partir del problema descrito son muy diversas, y su validación requiere de diferentes niveles de rigurosidad. Como la autoría de aumentaciones desde dispositivos móviles no ha sido explorada aún, el primer objetivo de este trabajo fue demostrar la factibilidad técnica y, luego, la operativa.

La factibilidad técnica trae aparejadas preguntas como ¿existe algún impedimento técnico para aumentar diferentes aplicaciones Web Móviles? ¿existen limitaciones respecto a los componentes utilizables del SDK del navegador móvil? ¿cuán rápida es la instanciación y ejecución de una aumentación del lado del cliente, dada la capacidad de un dispositivo móvil estándar? Estas preguntas pueden responderse mediante validación por construcción del software.

En cuanto a la factibilidad operativa es posible plantear si un usuario final, que no necesariamente tiene conocimientos de programación, es capaz de dominar los conceptos necesarios en forma satisfactoria para construir una aplicación de este tipo. Esta pregunta surge ante la diversidad de campos y conceptos que es necesario que un usuario comprenda para poder poner en práctica un enfoque de estas características. Producir este tipo de aplicaciones requiere de conocimiento sobre:

- Aumentación Web Móvil. Como, por ejemplo, página a aumentar, capa de aumentación, *augmenter*, sensores, tipos de contexto, valores del contexto, etc.
- El dominio de la aplicación a aumentar. Por ejemplo, si se aumenta Wikipedia, comprender que la misma comprende un conjunto de artículos que pueden describir personas, lugares, hechos históricos, etc. y que pueden estar interrelacionados y estructurados de diferentes maneras. Por ejemplo, si bien existen convenciones sobre las secciones principales[39], existen subsecciones que pueden ser específicas de un solo artículo, de modo que su estructura no es necesariamente la misma en cada uno de ellos.
- El dominio del problema a resolver. Si se aumenta con la finalidad de presentar un recorrido sobre los artículos Wikipedia de una ciudad, hay ciertos conceptos del dominio del turismo que se deben tener en cuenta, como PDI, recorrido, mapa, marcador, etc. Y también conceptos relacionados a tal fin con una tecnología particular, como por ejemplo la Hipermedia Móvil y los *walking links*.

En este sentido, y en concordancia con una de las preguntas destacadas por Barry Boehm [10] desde la ingeniería humana, es posible preguntarse: ¿el enfoque proveerá una forma satisfactoria para que los usuarios construyan sus aplicaciones de Aumentación Web Móvil?. Este tipo de pregunta, si bien ha sido crítica de diferentes publicaciones en campos diferentes a la Ingeniería de Software, es reconocida por Mary

39 https://es.wikipedia.org/wiki/Wikipedia:Estructura_de_un_art%C3%ADculo#Secciones_y_subsecciones

Shaw en [92] como una de las preguntas típicas de investigación aceptadas en el campo, específicamente relacionada a la demostración de factibilidad de un enfoque: ¿es posible lograr X? (del inglés, «Is it possible to accomplish X at all? »).

De esa pregunta inicial, se desprenden otras como: ¿el nivel de experticia de un usuario afecta la velocidad en la realización de las tareas o el nivel de éxito en las tareas de aumentación? En caso de que todos los artefactos requeridos para construir una aplicación sean soportados tanto por el framework como por la herramienta de autoría, ¿la creación mediante el uso del framework para desarrolladores es más ágil que haciendo uso de la herramienta de autoría? La respuesta a estas preguntas puede encontrarse detrás del diseño y la realización de un experimento, presentado en la sección 7, que permita recolectar la información necesaria.

A continuación, presentamos un conjunto de escenarios que permiten una mejor comprensión del problema identificado y que demuestran que actualmente existen problemas de adaptación en la Web que pueden ser resueltos mediante scripting.

4.1. Escenarios motivadores

Aumentar aplicaciones Web con características de contexto requiere de habilitar mecanismos que permitan recuperar los valores actuales de diferentes tipos de contexto a través de los sensores del dispositivo móvil, para poder adaptarlas en consecuencia a estos valores y mejorar la experiencia del usuario.

Con el fin de demostrar el potencial y la flexibilidad del enfoque propuesto en esta obra, se abordan a continuación un conjunto de potenciales escenarios que se caracterizan, precisamente, por el uso de información de contexto en sus aumentaciones. Los mismos dependen de alguno de los siguientes *tipos de contexto*: posición, orientación, nivel de presión sonora, hora actual y nivel de carga de la batería del dispositivo. Sin embargo, es posible contemplar otros escenarios que requieran del sensado de otros *tipos de contexto* (ej. nivel de luminosidad, velocidad de desplazamiento, categoría de usuario, estado de red, etc.) o de la combinación de varios de ellos.

4.1.1. Adaptando las entradas de Blogger para móviles

Blogger[40] es una plataforma que ofrece a sus usuarios la oportunidad de crear y administrar sus propios blogs. La misma fue ofrecida inicialmente como una aplicación

[40] https://www.blogger.com/

Web que luego fue, adicionalmente, lanzada como aplicación nativa para dispositivos móviles. Pese a su popularidad (puesto 113 del ranking global según Alexa[41]), no ofrece una versión Web adaptable a los dispositivos móviles.

Como se puede observar en la Figura 9, existen diferentes problemas que dificultan la interacción del usuario con esta UI; el tamaño de fuente es muy pequeño, por lo que hay que hacer zoom para poder leer tanto las etiquetas de los controles como el texto que el mismo usuario escribe. Si el tamaño de fuente de esta vista fuese el apropiado, aún existiría un segundo problema que es la forma de presentación del comportamiento de la aplicación; la cantidad de botones y links disponibles podría resumirse en, a lo sumo, los 10 más frecuentemente utilizados. Si el no contar con la totalidad de ellos es un impedimento, es posible ubicar al resto en un menú desplegable.

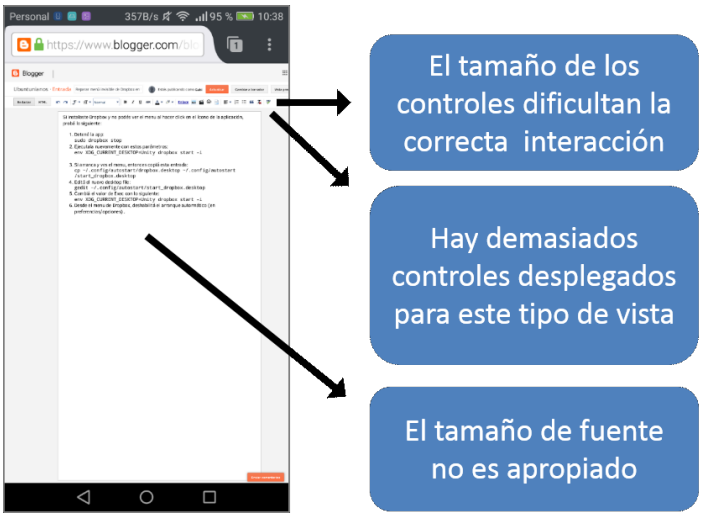

Figura 9. Vista de la edición de una entrada en Blogger

Mediante Aumentación Web es posible adaptar su interfaz, tal como se muestra en la Figura 10. Del lado izquierdo de la misma, se muestra cómo luce el editor de entradas de un blog mientras que, del lado derecho, se aprecia una vista donde el tamaño de fuente ha sido aumentado y la cantidad de controles, disminuida. El Script 4 del «Anexo B: Adaptación Web client-side» presenta y permite obtener dicha adaptación.

[41] Ver Anexo A: Análisis de sitios Web populares

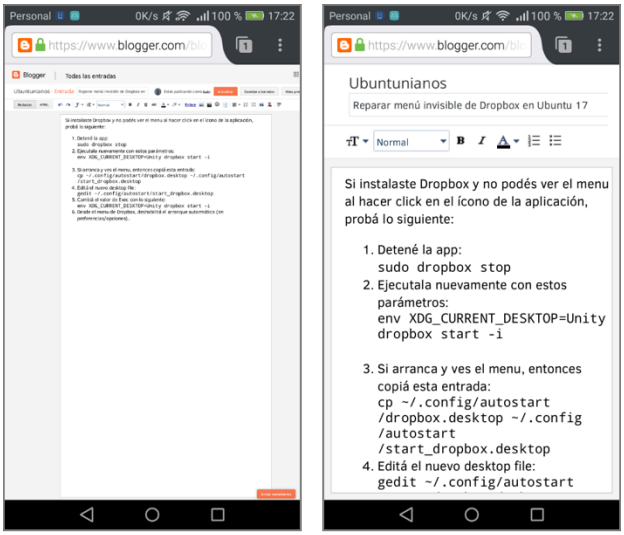

Figura 10. Adaptando la apariencia de Blogger para dispositivos móviles

4.1.2. Reproducción de videos y la orientación del dispositivo

Existen sitios cuya parte de su contenido es presentado en formato de video. Por ejemplo, están aquellas plataformas que permiten a sus usuarios subir, compartir y acceder a videos creados por otros usuarios de la comunidad, como YouTube, Vimeo, Dailymotion, VidMax o Wikimedia Commons. Otros permiten reproducir videos, tanto de su propia producción como externos, pero siempre seleccionados por sus editores, tal como lo hace Netflix o periódicos online como *Página 12* o *Le Monde*. Actualmente[42], los sitios mencionados simplemente adaptan su *layout*, pero el contenido restante sigue presente y su adaptación provoca que se pierda el foco del video, aun cuando el mismo se encuentra en reproducción. La Figura 11 muestra una página del diario *Página 12* presentando un video que, tras rotar el dispositivo en modo *landscape*, queda fuera de foco y parcialmente cubierto por un pie de página fijo.

En cualquier caso, la reproducción de dichos videos podría adaptarse a la orientación del dispositivo móvil, tal como se muestra en la Figura 12. Por ejemplo, haciendo que al rotar la pantalla en modo *landscape* se muestre automáticamente el video en modo *full screen*, ocultando todo el resto de contenido de la página y permitiéndole al usuario visualizar el video sin elementos superpuestos.

Una adaptación como la de la Figura 12 puede lograrse manipulando el estilo de los elementos que la componen, tal como se muestra en el Script 7 del «Anexo C: Adaptación Web Móvil del lado del client».

[42] Consultado el 04/08/2017, entre las 14:55 y las 15:10.

Figura 11. Comportamiento normal al reproducir un video en Página 12

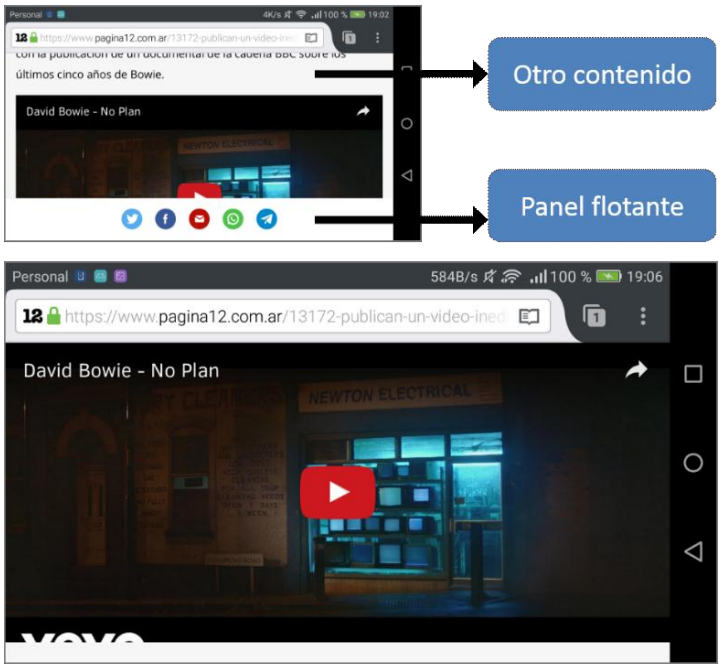

Figura 12. Adaptando la vista landscape de una noticia en Página 12

4.1.3. Streaming de audio y el nivel de presión sonora

Existen contextos en los cuales el nivel de presión sonora no permite escuchar claramente los sonidos que se están reproduciendo mediante el móvil. Aun cuando el usuario utiliza auriculares, el volumen que requiere para escuchar un audio en el interior de un edificio puede no ser el mismo que en el corazón de una ciudad a en horas pico.

Consideremos una persona a la que le gusta escuchar una radio local de su ciudad a diario, y asiste a una conferencia en el extranjero. El alcance de esta radio por los medios convencionales no es posible, y tampoco existe una aplicación nativa soportando su streaming, pero existe una versión Web que puede ser accedida desde su

dispositivo móvil[43]. Desde la tranquilidad de su casa, el usuario escucha la radio y su entorno registra una presión sonora de entre 40 y 70 dB. Pero al dirigirse a la conferencia, debe caminar algunas cuadras por el centro de la ciudad. Allí, el nivel de presión sonora fluctúa entre 90 y 120 dB, por lo que decide aumentar el volumen de los sonidos multimedia del dispositivo.

Esta adaptación podría ser automatizada mediante aumentación Web, evitándole al usuario tener que manipular el dispositivo o sus accesorios para controlar el volumen manualmente. Adicionalmente, la vista de este reproductor podría ser adaptada acorde a las dimensiones del dispositivo, de manera similar al escenario de la sección 4.1.1. El resultado de la adaptación de la vista se muestra en la Figura 13, mientras que lo necesario para la adaptación sonora se presenta en detalle, en el Script 8 del *Anexo C: Adaptación Web Móvil del lado del cliente*. En líneas generales, el script se encarga de procesar el streaming proveniente del micrófono para calcular el volumen actual, y ante determinados niveles solicita al reproductor de audio aumentar o disminuir su volumen, para que el usuario tenga una mejor experiencia.

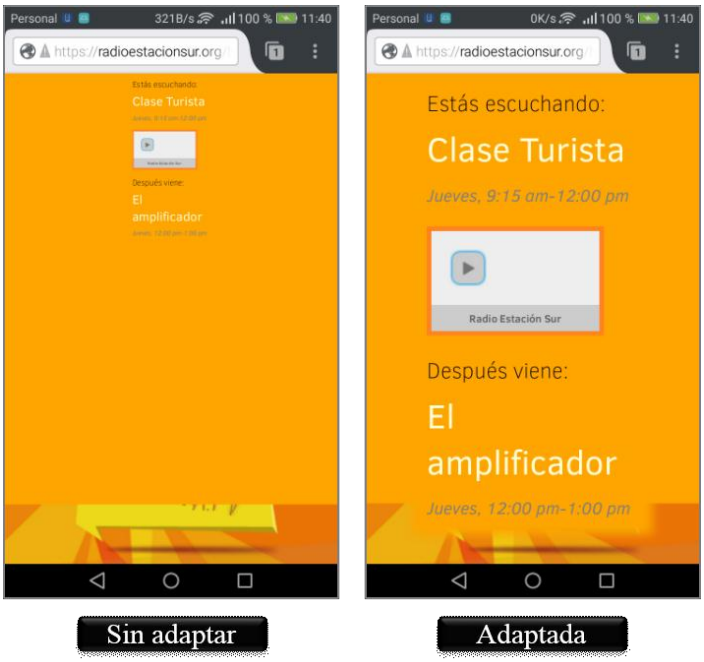

Figura 13. Adaptando la vista y el volumen de un reproductor de radio

[43] http://www.radioestacionsur.org/ consultada y analizada el 18/01/2016

4.1.4. Cines y la hora actual

Algunas cadenas de cine tienen portales comunes a diferentes ciudades, en las que ofrecen una cartelera pública de las películas en estreno. Por ejemplo, Cinemacity[44] ofrece información sobre la cartelera disponible en diferentes salas distribuidas en La Plata. En ella, los usuarios pueden acceder a un listado de películas en estreno y ver, por cada una de ellas, en qué salas y horarios se proyecta. Del lado izquierdo de la Figura 14 puede observarse que cada conjunto de horarios por cine está especificado dentro de un mismo elemento del DOM, sin estilo y ni estructura que permita identificar un horario fácilmente en la lista.

En este escenario se podría soportar el resaltado de las películas de acuerdo a la hora actual, ofreciendo una visión general de aquellas que están próximas a proyectarse o acaban de comenzar, tal como se muestra del lado derecho de la Figura 14. Los detalles sobre la implementación de esta adaptación se encuentran en el Script 9 del *Anexo C: Adaptación Web Móvil del lado del cliente*.

Figura 14. Resaltando próximas funciones de acuerdo a la hora actual

4.1.5. Esquema de colores «light-on-dark» y el nivel de carga de la batería

«Dark-on-light» es un esquema de colores que utiliza fuentes obscuras sobre fondos claros, y muchos sitios en la Web fueron diseñados con él en mente, especialmente aquellos a favor del *green computing*[45]. Un ejemplo es Blackle[46], una del motor de

[44] http://www.cinemacity.com.ar
[45] uso eficiente de los recursos computacionales para minimizar el impacto ambiental
[46] http://www.blackle.com/about/

búsqueda de Google que permite ahorrar energía alterando sus colores originales. Sin embargo, el ahorro de energía no es la única razón por la que un usuario podría desear adaptar una aplicación Web.

Consideremos un usuario que acostumbra a leer noticias y clasificados como primera actividad del día y le molesta el brillo que genera el estilo de un sitio, o que se encuentra viajando de noche y no quiere molestar a otros pasajeros con el resplandor de su dispositivo. Los portales que este usuario acostumbra a acceder tienen un fondo blanco, y a él le gustaría poder invertir su estilo. «Dark Background and Light Text» es una extensión de navegador desktop de más de 35.000 usuarios, que provee dicha funcionalidad, pero no está disponible para la versión de su navegador móvil (v. 46).

Como se puede apreciar en la Figura 15, el sitio donde el usuario consulta los clasificados fue diseñado con un esquema de colores «light-on-dark», y podría ser adaptado con la capacidad de aplicar un esquema opuesto. Para ello, basta con suscribirse a los cambios del nivel de luminosidad percibida mediante un listener al evento devicelight[47], y realizar las transformaciones correspondientes sobre el estilo de la página mediante la aplicación del filtro «invert»[48]. Para más detalles sobre la implementación de esta adaptación, consultar el

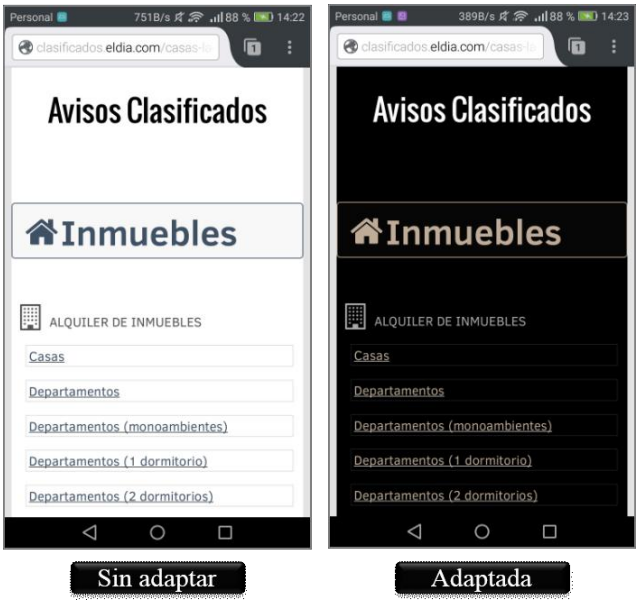

Figura 15. Aplicando un esquema light-on-dark sobre un sitio de clasificados

[47] https://developer.mozilla.org/en-US/docs/Web/API/DeviceLightEvent
[48] https://www.w3schools.com/cssref/css3_pr_filter.asp

4.1.6. Guías turísticas y la posición del usuario

Las guías turísticas son un tipo de aplicaciones móviles, las cuales suelen considerar: 1) un conjunto de PDIs, 2) un camino predefinido para navegar entre estos PDIs, y 3) un método para detectar la posición actual del usuario (códigos QR, GPS, etc.). Los escenarios posibles para este tipo de aplicación son diversos:

- Exhibiciones fijas de interior en museos. Algunos museos utilizan códigos QR para identificar cada pieza su colección, con el fin de que los visitantes pueden escanearlos con sus dispositivos y acceder a una contraparte digital de la pieza. Por ejemplo, se pueden considerar los códigos generados por *QRPedia*, que permite generar códigos que redireccionan a artículos de *Wikipedia* de acuerdo con el idioma utilizado en el idioma de preferencia del usuario. Sin embargo, utilizar *Wikipedia* como soporte digital de las piezas del museo carece de exclusividad y no es posible editar su información libremente para el beneficio de una institución, mucho menos asegurar que dicha información, en caso de ser aceptada por los miembros de una comunidad, permanezca vigente con el paso del tiempo. Esta podría ser actualizada o eliminada. Mediante Aumentación Web se podría mostrar en información provista exclusivamente por el museo sobre la base de los artículos de *Wikipedia*, como por ejemplo la ubicación de la pieza en el mismo, el camino que debe recorrer el usuario desde su posición actual hasta la pieza, o sugerencias de qué otras piezas se recomienda visitar previa posteriormente, y cómo llegar a ellas. Un ejemplo de este tipo de recorrido es utilizado en el *Museo de Ciencias Naturales de La Plata*, bajo el nombre «Siguiendo a Darwin en el Museo de La Plata»[49], y las piezas del Museo que coinciden con las de este recorrido tenían asociado un código QR generado mediante QRPedia.
- Exhibiciones temporales. En esta modalidad, el ente a cargo presenta diferentes colecciones a través del tiempo, usualmente por períodos de menores a dos meses. El problema con esta modalidad es que, si las colecciones no son parte de un conjunto permanente, el continuo cambio podría reducir la probabilidad de proveer asistencia a los usuarios con información sobre el recorrido y de cada pieza. Ejemplos de este tipo de recorridos son los presentados por el *Museo de Arte Contemporáneo Latinoamericano*[50], como «El cuerpo y los interrogantes» llevado a cabo en enero de 2017. Dicha exhibición está compuesta por una obra de once artistas, y el mismo museo ha publicado información por cada uno de ellos[51] que, si bien no está explícitamente vinculada al tour, puede utilizarse como contraparte

[49] http://www.fcnym.unlp.edu.ar/museo/educativa/darwin/darwinenmuseo.html
[50] http://www.macla.com.ar/exposiciones/2016
[51] Por ejemplo, de Magyar Ladislao: http://www.macla.com.ar/patrimonio/colecci%C3%B3n-general/artistas -con-m/magyar-ladislao

digital de cada cuadro y aumentada con información sobre el recorrido en general.

- Exhibiciones itinerantes. Diseñadas para ser llevadas a cabo en las facilidades de cualquier instalación o espacio abierto, presentan una colección de piezas enfocadas en diferentes tipos de manifestaciones artísticas, históricas o culturales. Los puntos de interés en este tipo de tours pueden distribuirse entre las diferentes aulas de una escuela, y pueden estar vinculados a una contraparte digital. Un ejemplo de este tipo de recorridos es «Dinosaurios de la Patagonia» ofrecido por el *Museo Paleontológico Egidio Feruglio*[52], cuyas piezas han sido trasladados y presentados en diferentes países como España, Portugal, Alemania y República Checa. Para este tipo de recorrido oportunista se podría ofrecer aumentaciones que muestren un orden en el recorrido de las piezas, y que recuperen comentarios sobre la misma en Twitter, presentando en una sección aparte aquellos que fueron posteados geográficamente cerca de la posición actual del usuario, a fin de obtener un pantallazo sobre las reflexiones que hicieron las personas del lugar (o que realizaron el tour) sobre una pieza concreta.
- Recorridos al aire libre. Son recorridos que se desarrollan a la intemperie, conformados generalmente por los sitios patrimoniales más significativos de una ciudad, permitiendo a los turistas conocer los mejores lugares para experimentar y apreciar su historia. Por ejemplo, *Tripin* sugiere un conjunto de lugares para visitar y conocer en ciudad de La Plata, con información como contraparte digital dedicada a cada punto de interés, aunque sin un orden establecido. En este caso, se podría asociar una coordenada a cada contraparte digital, y aumentar la aplicación Web existente para que cada vez que el usuario se acerque a 15 metros de dichas ubicaciones, la página correspondiente sea cargada y aumentada con asistencia para el recorrido (en base a la posición actual del usuario y recomendaciones de recorridos a seguir). El contenido de cada página puede ser ampliado también con críticas extraídas de *TripAdvisor* o contenido geo-posicionado de *Twitter*.

En cualquier caso –itinerante o fija, de interior o exterior, artísticas, históricas, culturales–, las exhibiciones pueden ser aumentadas con contenido externo en relación a la posición del usuario. La misma puede ser interpretada suscribiéndose a los cambios en la posición GPS del mismo[53], aunque también puede ser obtenida de otras maneras, como decodificando códigos QR, una etiqueta RFID, etc. En la Figura 16 se muestra cómo un artículo de Wikipedia puede ser adaptado al visitar una posición específica de un museo, desplazando su contenido hacia abajo (captura central) y agregando nuevo

[52] http://www.mef.org.ar/exhibiciones/muestras-itinerantes
[53] https://developer.mozilla.org/en-US/docs/Web/API/Geolocation/watchPosition

contenido específico del museo (captura derecha). Esta integración de contenido es posible gracias a que el Script 11, del *Anexo C: Adaptación Web Móvil del lado del cliente*, se ejecuta en el contexto y con los privilegios del navegador y no de la página Web.

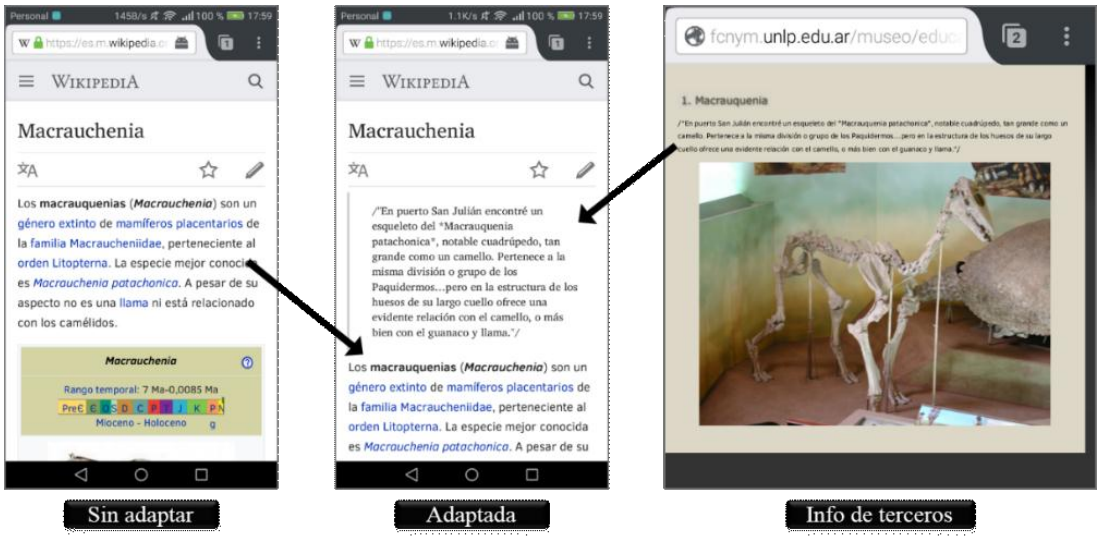

Figura 16. Aumentando un artículo de Wikipedia con contenido externo

5. Un nuevo enfoque

En este trabajo, el desafío consiste en proveer al usuario final con los medios suficientes para mejorar sus experiencias Web de acuerdo con la información del contexto. Para ello, se establecieron una serie de pasos para la construcción de una aplicación de Aumentación Web Móvil y se desarrolló un framework, con relación al cual se creó una herramienta visual de EUD.

Haciendo uso de esta herramienta, los usuarios finales deben ser capaces de mejorar sus experiencias en la Web Móvil *on-demand* y, aunque no necesariamente, bajo la modalidad in-situ. Un beneficio subyacente al de permitirle al usuario final crear sus propias soluciones, es el de facilitar la validación de requisitos funcionales. Puesto que los mismos ya no deben ser comunicados entre –al menos– dos personas, sino que es el mismo usuario quien establece y satisface sus propias necesidades y condiciones a satisfacer. La modalidad in-situ también facilita esta validación, pues la aplicación es construida y testeada, mediante técnicas de *live programming* [74], bajo el mismo contexto en el que la aplicación será puesta en producción.

5.1. Roles de usuarios involucrados

La idea principal detrás de esta obra comprende a un *productor* creando una aplicación de Aumentación Web móvil con una herramienta de autoría [13], construida en base a componentes de un framework [12]. Sin embargo, este aporte se encuadra dentro de un enfoque más general, de crowdsourcing de aumentaciones Web móviles [24], en el que diferentes roles de usuarios cooperan en la realización de un fin, tal como se observa en la Figura 17.

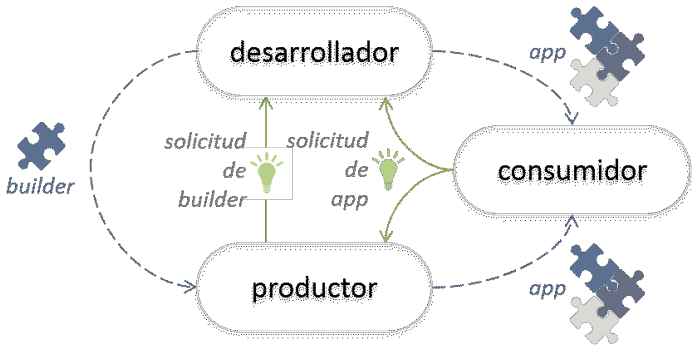

Figura 17. Roles de usuarios en MoWA y sus relaciones

Desde una perspectiva general, tanto *desarrolladores* como *productores* pueden crear aumentaciones –aunque por diferentes medios– que un *consumidor* puede finalmente operar sobre un *weaver* de navegador Web Móvil. A continuación, se detallan los diferentes roles:

- Usuarios desarrolladores
    - Scripters: son aquellas personas que crean aplicaciones mediante scripting y haciendo uso de MoWA framework [12]. Dentro de una plataforma de Crowdsourcing, son quienes satisfacen las necesidades de los usuarios finales que solicitan aplicaciones para resolver problemas en escenarios específicos, como así también los artefactos de construcción requeridos por los *productor*es.
- Usuarios finales
    - Productores: son usuarios finales con la capacidad de crear sus propias aplicaciones mediante el uso de una herramienta de Desarrollo por Usuarios Finales. No necesariamente tienen conocimiento formal en el desarrollo de software; simplemente dominan una herramienta de autoría. Por lo general, crean sus aplicaciones para solucionar problemas de su vida cotidiana, pero en el contexto de una plataforma de Crowdsourcing pueden también ser quienes ayuden a otros usuarios construyendo las aplicaciones que satisfagan sus necesidades. Finalmente, un *productor* puede iniciar una petición para la creación de un componente de construcción –*builder*– que su herramienta de autoría no le ofrece. Por ejemplo, la incorporación de un nuevo sensor o augmenter específico.
    - Consumidores: son quienes instalan y utilizan las aplicaciones que los *desarrolladores* y los *productores* crean. Un mismo usuario puede jugar el rol tanto de *productor* como de *consumidor*. Pero mientras un *productor* extiende software para las soluciones dentro de uno o varios dominios, el *consumidor* simplemente las utiliza, lo cual puede involucrar configurar aplicaciones parametrizables. Es preciso recordar que parametrizar no implica desarrollar [71], por lo que los problemas que un *consumidor* resuelve están limitados a los que la aplicación ofrece. En el contexto de la plataforma de Crowdsourcing, los *consumidor*es son quienes crean solicitudes de nuevas aplicaciones y/o descargan aplicaciones para ser utilizadas desde sus dispositivos.

Otro enfoques que plantea roles similares es [36]. En él, los usuarios finales de aplicaciones Web “prosumen” scripts de aumentación y pueden ser considerados tanto *productor*es como *consumidores* que necesitan escribir o confiar en JavaScript, respectivamente. Cotermann & Kumar [30] examinaron trabajos que ya en el ’89 contemplaban al usuario como *productor* o *consumidor* de información dentro del contexto general de sistemas de información basados en computadoras. Otros trabajos también hacen referencia a estos roles, aunque no presentan un debate tan elaborado, como los anteriores trabajos mencionados, sobre qué implica un rol u otro. Por ejemplo,

Nestler [79] propone potenciar a los *consumidores* de soluciones SOA pre-definidas para convertirse en *productor*es de sus propias aplicaciones, y Lieberman et al. [71] mencionan que, en lo que respecta a la creación de contenido y modificación e funcionalidad por parte de no-profesionales, existe una división de labor entre aquellos que producen y aquellos que consumen.

En cualquier caso, existe un límite de difícil definición entre un producer y un desarrollador. Algunos autores señalan que lo que realmente define si una «persona que programa» es un «end-user programmer», es la intención de su producción; si la misma es para uso personal y de corto plazo, o si otros usuarios pueden hacer uso y ayudar a mantenerlo [66]. Para Rod et al. [85], el EUD debe permitir al usuario desarrollar de manera oportunista, ocultar la complejidad innecesaria e ir permitiéndole al usuario crear soluciones más específicas conforme su conocimiento va creciendo. Esto da lugar a la existencia de end-user programmers novatos o experimentados. En dicho trabajo, los autores mencionan otros tipos de herramientas que, si bien asisten y facilitan la construcción de aplicaciones, no se consideran de EUD porque asumen el conocimiento, cultura de trabajo y expectativas de un desarrollador.

5.2. El contexto y las soluciones para cada rol

En relación a los roles previamente mencionados, se proponen tres herramientas que soportan la oferta y demanda de aplicaciones de aumentación:

- *MoWA authoring*, una extensión de navegador que soporta la creación de aplicaciones Web Móvil del lado del cliente.
- *MoWA crowd*, una plataforma online soportando servicios de Crowdsourcing, de almacenamiento y para compartir artefactos MoWA; tanto aplicaciones como componentes del framework para *scripters* o *producers*.
- *MoWA weaver*, otra extensión de navegador que soporta la gestión y ejecución de aplicaciones MoWA.

Tal como se muestra en la Figura 18, un *productor* puede interactuar con *MoWA Authoring* para crear experiencias Web Móviles a partir de aplicaciones Web existentes. Puede crear estas aplicaciones de acuerdo con sus propios requisitos, aunque también puede hacerlo de acuerdo con los de otro *consumidor*, materializados en una petición dentro de la *MoWA Crowd*. En ambos casos, crear una aplicación implica la extensión de una clase de aplicación existente; para ello, el usuario debe seleccionar una aplicación existente, ya sea que la misma se encuentra en el almacenamiento local o en un repositorio remoto. El usuario debe seguir una serie de pasos y, al final del proceso de construcción, la aplicación resultante es guardada en el almacenamiento local de la extensión, con la finalidad de editarla o simplemente ejecutarla posteriormente. Por

último, el *productor* tiene la posibilidad de compartir su aplicación en *MoWA crowd*, ya sea para permitir a otros mejorar su experiencia Web o en respuesta a una solicitud de otro usuario en la plataforma.

Un *consumidor* puede instalar *MoWA weaver* en su navegador Web Móvil y comenzar a experimentar la Web de una manera diferente, aumentada. Para ello, es necesario que instale al menos un script de aumentación que define e instancia una aplicación MoWA. Estas aplicaciones pueden ser importadas desde el repositorio de *MoWA Crowd* o estar ya presentes en el almacenamiento local del dispositivo (porque el mismo usuario juega el rol de *productor* y las creó). Si el usuario no encuentra en el repositorio una aplicación que cubra sus expectativas, puede iniciar una nueva petición en *MoWA crowd* y esperar la ayuda de otro usuario. Por último, un *consumidor* puede administrar las aplicaciones instaladas en su dispositivo; editarlas, borrarlas, extenderlas, cambiar su orden de ejecución (con relación a otras aplicaciones), etc.

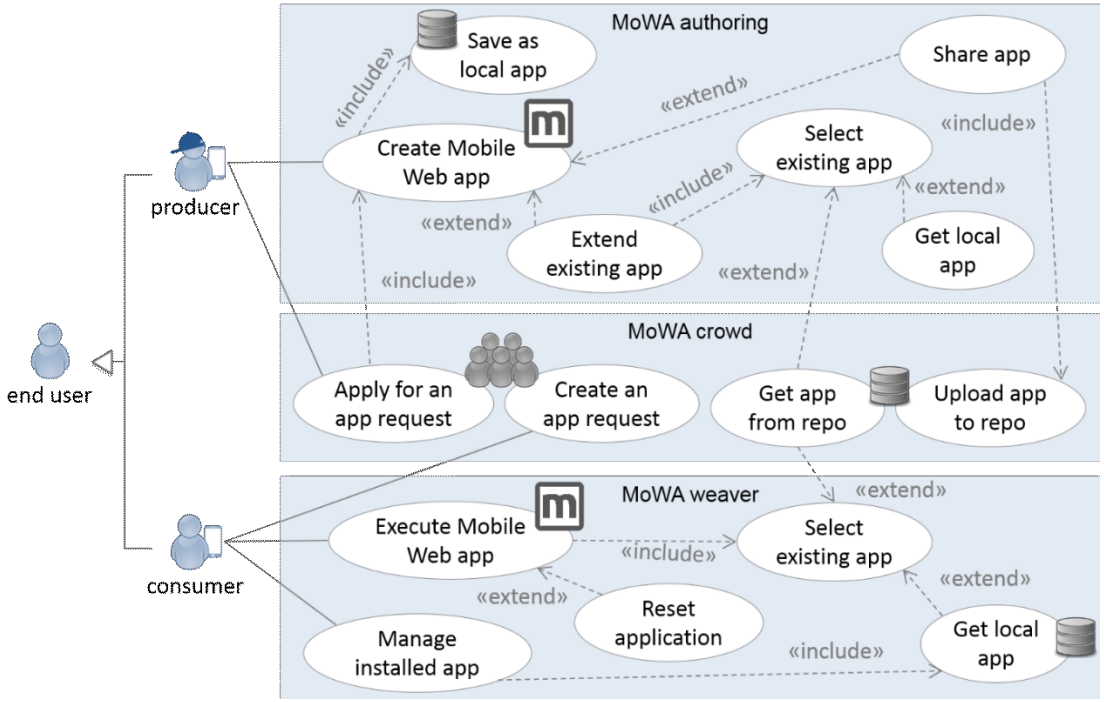

Figura 18. Crowdsourcing de aumentaciones Web móviles

5.3. MoWA para usuarios finales

Si bien a nivel general participan múltiples roles y se requieren múltiples herramientas, esta obra se centra en el proceso de autoría de las aumentaciones móviles y su herramienta de soporte –*MoWA Authoring*–, cuyo contexto se describe a continuación.

En la Figura 19 se presentan dos herramientas MoWA del lado del cliente. Ambas están dirigidas a usuarios finales y son extensiones de un navegador Web móvil. Una de ellas soporta el proceso de construcción y, la otra, la ejecución de las aplicaciones resultantes. También existe una contraparte del lado del servidor que soporta el almacenamiento compartido de aplicaciones y los servicios de crowdsourcing.

Del lado derecho de la figura, se puede apreciar un dispositivo de un cliente que ha instalado *MoWA weaver* en su navegador Web Móvil. Con esta herramienta, él puede descargar aplicaciones desde el repositorio, instalarlas y utilizarlas en los escenarios para los cuales fueron definidas, mejorando su experiencia Web con características móviles. Tal como se mencionó previamente, dichas aplicaciones pueden ser creadas de dos maneras: por desarrolladores mediante scripting [12], o por *productores* utilizando *MoWA Authoring* [13].

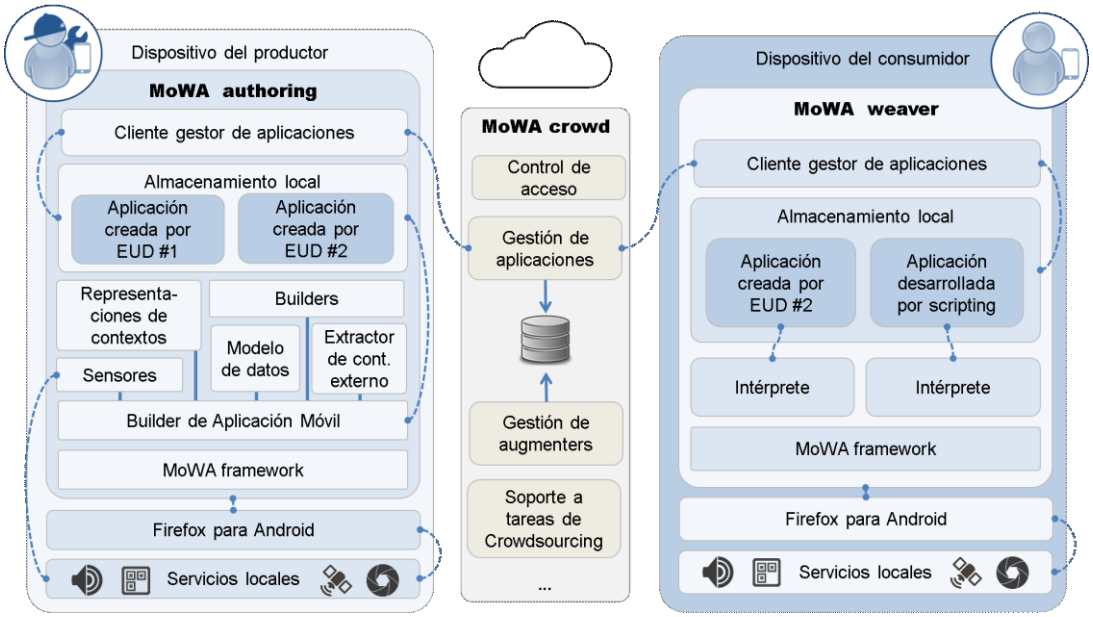

Figura 19. Representación de alto nivel de la plataforma para usuarios finales

En la zona central de la figura, se presentan algunos de los componentes de MoWA Crowd, que soportan la gestión de tareas de Crowdsourcing y un repositorio de artefactos MoWA.

Del lado izquierdo de la figura se presenta la arquitectura de la herramienta de autoría. La misma se instala como una extensión del navegador móvil en el dispositivo del *productor*. Comprende un conjunto de clases especializadas que tienen una referencia a las clases base que componen una aplicación MoWA y extiende sus funcionalidades con la capacidad de solicitar a los usuarios finales los parámetros requeridos para su correcto funcionamiento. Se trata de un proceso asistido, basado en formularios, con la finalidad de poder instanciar dichas clases con los valores correspondientes al escenario que pretende resolver el *productor*.

Por cada componente de una aplicación MoWA existe un *builder* que se encarga de una parte del proceso de autoría, ayudando a los usuarios finales a configurar las instancias específicas del componente. La herramienta de autoría conoce a todos los *builders* a cargo del proceso de construcción de la aplicación y sus dependencias, para poder ejecutar la aplicación aún mientras está siendo creada. Los *builders* son

presentados en cierto orden, tal como se presenta en la Sección 5.4, asegurando que las instancias de componentes dependientes ya sean posibles de crear. Por ejemplo, no sería posible seleccionar y configurar un *augmenter* dependiente de la posición del usuario si, al menos, no se ha definido una página Web a aumentar o si no se ha seleccionado un sensor de posición y un valor del contexto de interés.

Los *builders* pueden depender de algunos valores del contexto del usuario para configurar el componente de la aplicación subyacente. Por ejemplo, si un *productor* construye una aplicación mientras recorre una ciudad, podría beneficiarse de un mecanismo que le permita detectar su posición actual y crear un marcador en el mapa. Si el usuario está construyendo un tour basado en códigos QR existentes en un edificio, es de utilidad que la herramienta le provea un mecanismo para decodificar esos códigos y asociarlos fácilmente a un valor del contexto, por ejemplo, a una posición física determinada. De esta manera se puede apreciar que, si un usuario elige utilizar un *augmenter* dependiente del contexto, el *builder* a cargo de su configuración también debe contar con los medios suficientes para percibir el contexto.

La herramienta de autoría proporciona al usuario final la capacidad de reutilizar el contenido Web existente, pero no sólo tomándolo como base para las aumentaciones, sino también para aumentarlas. Por ejemplo, se puede tomar el perfil de los actores en IMDB como páginas Web a aumentar e inyectarle un *carrousel* con tráileres relacionados, extraídos de YouTube. La extracción de contenido es responsabilidad de un componente denominado External Content Extractor, disponible para todos los *builders*. Dicho componente se instancia dentro de un contexto privilegiado (como parte de la extensión del navegador), que le permite añadir comportamiento adicional en cualquier página Web sin restricciones (ej. no preocuparse de la política del mismo origen). Esto también hace factible la manipulación de elementos del DOM para obtener sus datos de posicionamiento (por ejemplo, el XPath) y consumir dinámicamente su contenido desde contextos externos (otras páginas Web que no comparten el mismo nombre de dominio).

5.4. El proceso de autoría

En líneas generales, existen cuatro componentes fundamentales en la creación de una aplicación MoWA: sensores y valores del contexto, contrapartes digitales a aumentar y *augmenter*s. La Figura 20 presenta una serie de etapas para soportar el proceso de autoría de una aplicación MoWA (el caso de uso «create Mobile Web app» de la Figura 18).

En primer lugar, el usuario debe *establecer la información de base* de la aplicación como, por ejemplo, el nombre que tendrá la misma. De forma transparente, una herramienta soportando este proceso debería ser capaz de generar en este paso aquellos datos necesarios para su administración, como ser un espacio de nombres, el nombre de archivo, etc.

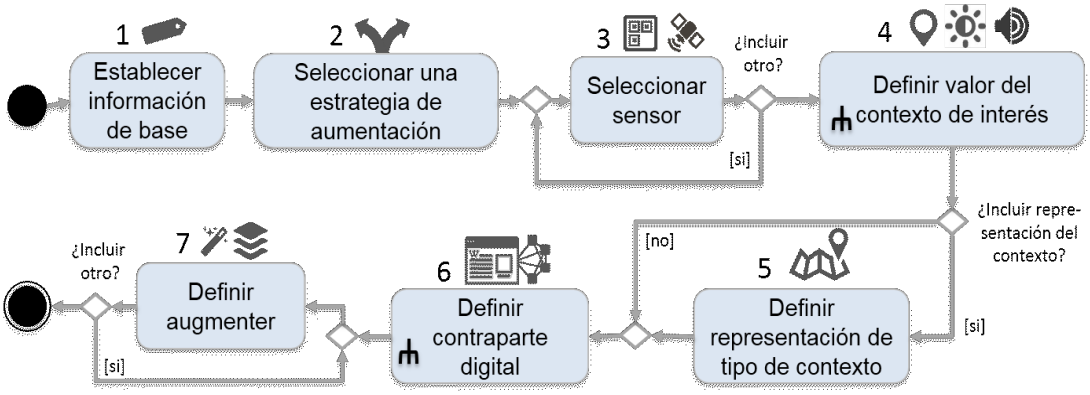

Figura 20. Proceso de creación de una aplicación con MoWA Authoring

Luego, debe *seleccionar una estrategia de aumentación*, para lo cual se ofrecen dos alternativas. La primera permite aumentar cualquier página que el usuario se encuentre navegando, siempre que la URL sea coincidente con un patrón definido para la aplicación. La segunda, verifica que el valor del contexto detectado se corresponda con alguno de los valores definidos como «de interés» para la aplicación. De ser así, carga la URL vinculada a la contraparte digital del valor «de interés» y, una vez que se carga la página Web, aplica una capa de aumentación con los *augmenters* correspondientes.

El siguiente paso involucra *seleccionar uno o más sensores*. Para cada tipo de contexto soportado se ofrece al usuario un conjunto de sensores que permiten escuchar sus cambios. Por ejemplo, tanto un «sensor GPS» como un «sensor QR» podrían detectar la ubicación del usuario; un «sensor de Luxes» puede notificar los cambios en el nivel de luz percibida por el dispositivo móvil; un «sensor de dBs» puede notificar los cambios en el nivel de presión sonora en el ambiente.

Por cada uno de los sensores seleccionados, se debe definir al menos un *valor de contexto de interés* para la aplicación. Cada sensor notifica los cambios de un tipo de contexto a los *augmenters* suscritos (se definen en la siguiente etapa), para que estos puedan adaptar la Web en consecuencia. Pero para poder conseguirlo, es necesario que la aplicación sea capaz de reconocer qué valores son representativos para su propósito. Por ejemplo, un usuario final podría estar construyendo una aplicación basada en el posicionamiento. Su aplicación necesitaría conocer esencialmente un conjunto de PDI o ubicaciones que le permita activar las aumentaciones en determinadas circunstancias.

La *definición de una representación de tipo de contexto* es un paso alternativo, puesto que no todas las aumentaciones hacen uso de ella. Tal como se explicará en secciones posteriores, algunos *augmenters* pueden necesitar hacer uso de una representación visual de los valores del contexto. Para ello, cada valor debe tener una representación dentro del marco de un espacio común de valores, como lo es un mapa para un marcador (relacionado a un par de coordenadas GPS) o una simple escala para

un marcador de lux (que coincide con un valor lux determinado). Sin embargo, como algunos *augmenters* no necesitan de esta representación, este paso es opcional.

El siguiente paso consiste en decidir qué se quiere aumentar; *definir una contraparte digital*. Cada valor de contexto debe tener asociada una contraparte digital. La misma debe poseer una URL (página Web a aumentar) y tener asociado un conjunto de *augmenters* a ejecutar. Además, puede tener algunas propiedades extra definidas por el usuario. Por ejemplo, si un usuario final está creando una gira por la ciudad de La Plata, definirá diferentes PDI conforman un recorrido. Estos PDI están asociados a una contraparte digital de la aplicación. Supongamos que la contraparte «Plaza Moreno» tiene como ubicación las coordenadas «-34.920609, -57.954393», como URL «https://es.wikipedia.org/wiki/Plaza_Moreno» y la aumentación que provee solo consta del *augmenter* «TitleSection». Estos valores permiten que una aplicación cargue el artículo de Wikipedia cuando el *consumidor* se encuentra cercano a las coordenadas especificadas y que ejecute el *augmenter* que muestra el título de una nueva sección. Dicho *augmenter* se definirá en el paso siguiente, pero el mismo puede utilizar información definida para la contraparte digital como, por ejemplo, su nombre o alguna descripción que el usuario haya definido como propiedad extra, ya sea introducida manualmente o definida en base a contenido externo a la página (usualmente de terceros).

Definir *augmenter*. En esta etapa del proceso, la herramienta pide al usuario que defina un conjunto de aumentadores para cada página Web a aumentar. Proporciona al *productor* un conjunto de aumentadores de acuerdo a los sensores que ha elegido y las representaciones de tipo de contexto que ha definido. Esta presentación se hace en base a un simple mecanismo de etiquetado que se especifica mediante metadatos en el archivo de clase del *augmenter*. Los *augmenters* se definen en el contexto de una contraparte digital, y el *productor* puede agregar tantos como quiera. Algunos *augmenters* pueden necesitar ser configurados con valores de entrada concretos para ejecutarse correctamente. Por ello, se ofrecen tres alternativas al usuario final para la definición de los valores de dichos parámetros: 1) puede reutilizar los datos ya definidos, asociados a un valor del contexto –ej. el nombre de un PDI– al acceder al modelo de datos y seleccionar una propiedad; 2) puede manualmente Introducir dichos datos en el formulario; o 3) puede usar un asistente para recuperar contenido ya existente en la Web (generalmente externo).

6. Soporte conceptual y de software

En esta sección se presenta la herramienta de soporte al enfoque[54]. Tal como se ha explicado en la Sección 5, esta herramienta toma como base los componentes de *MoWA framework* y los hace configurables mediante *builders*. Sin embargo, es preciso aclarar que la arquitectura presentada [12] fue adaptada en beneficio del nuevo enfoque, y si bien el propósito respecto a la aumentación el mismo, algunos componentes difieren. Un ejemplo sustancial es que mientras en [12] se contemplaban diferentes tipos de aplicaciones, en la presente obra no se modela más que un solo tipo de aplicación que puede estar suscrita a diferentes sensores y estar compuesta por *augmenters* especializados en uno o varios dominios concretos.

La herramienta de soporte al enfoque propuesto fue diseñada tratando de afrontar las barreras de aprendizaje descritas por Ko et al. [4], como también en relación a los hallazgos y recomendaciones de otros autores [51][75][84]. La misma asiste al usuario en el proceso de construcción, principalmente, mediante formularios que le solicitan la información necesaria para instanciar una aplicación móvil. Sin embargo, al momento de crear la capa de aumentación, se intenta reproducir un entorno de desarrollo *live-programming* [74], que le permite previsualizar las aumentaciones en cualquier momento, chequear si las mismas funcionan correctamente y de acuerdo a sus expectativas. Esta parte soportando la construcción de la capa de aumentación está compuesta principalmente por widgets preconstruidos arrastrables y configurables mediante formularios. De esta manera, los *productor*es no requieren tener conocimiento en un lenguaje de scripting textual, pero si deben tener en claro qué tipo de experiencia móvil quieren construir sobre una aplicación Web existente, fundamentalmente qué sensores y tipos de contextos se requiere para tal fin.

Para el desarrollo de la solución propuesta, también se tomaron en cuenta los siguientes aspectos de variabilidad [51]: modelo de contexto dinámico, soporte de sensores y adaptación específica de dominio. Se adoptó un asistente para la construcción basado en formularios, tal como lo hicieron Pittarello & Bertani [84], ya que su trabajo demostró ser efectivo incluso con niños sin conocimiento en programación. En este sentido, si bien la construcción no es orientada a la Aumentación Web, presenta similitudes. Por ejemplo, mientras ellos permiten a los usuarios finales componer incrementalmente una historia, *MoWA Authoring* permite definir, bajo la misma modalidad, los valores de interés de una aplicación (ej. coordenadas, luxes, niveles de presión sonora) como así también las unidades que conformarán la capa de

[54] https://github.com/gbosetti/mowa-authoring

aumentación (para mostrar un título, la descripción de un PDI, sus walking links, etc.). Pittarello & Bertani también ofrecen un mecanismo para recuperar automáticamente la posición del usuario; *MoWA Authoring*, en cambio, soporta un rango más amplio de tipos de contexto, aunque no es obligatorio hacer uso de todos: posición, lux, dB, orientación, etc. Finalmente, y en relación al dominio escogido para los escenarios descriptivos y la evaluación, se tuvo en cuenta la «visualización de anclajes» y el «soporte al recorrido del enlace» explicados por Millard et al. [75].

6.1. Aumentación móvil del lado del cliente

Las ventajas de desarrollar la herramienta propuesta como una extensión del navegador y no como una aplicación nativa independiente es, principalmente, la posibilidad de reutilizar el comportamiento de un navegador construido en base a estándares y no tener que forzar al usuario a aprender a utilizar un entorno completamente nuevo para navegar la Web. Además, este navegador fue de los precursores en ofrecer acceso a las capacidades subyacentes al dispositivo mediante APIS, rompiendo la brecha entre las capacidades de aplicaciones nativas y Web [26], que otros navegadores no permitían en su momento. Por ejemplo, el equipo de Chrome for Android aún hoy no tiene planes de incorporar extensiones[55].

En concreto, Firefox para Android expone APIS en relación con la batería, la cámara, la geolocalización, la orientación, la mensajería móvil, la telefonía Web y a todas las APIs que dispone una aplicación Web convencional. Gracias a esto, es posible la implementación de diferentes tipos de sensores y de *augmenter*s que recuperen información de terceros en el contexto de una aplicación MoWA.

Dentro de una extensión de esta naturaleza, existen dos diferentes tipos de scripts[56] y de privilegios. Un *add-on script* forma parte del núcleo de la extensión y tiene mayores privilegios que el código de una aplicación Web tradicional. Por ejemplo, puede enviar peticiones, recuperar y parsear contenido externo sin las restricciones propias de la política del mismo origen[57]. También tiene acceso a los objetos globales definidos como parte del núcleo del lenguaje JavaScript, sin embargo, no tiene acceso directo a los de la especificación de HTML5[58], como por ejemplo *window* o *document*. Un *content-script* tiene acceso a ambos tipos de objetos, pero no puede hacer uso de las

[55] "Chrome apps and extensions are currently not supported on Chrome for Android. We have no plans to announce at this time" [Las aplicaciones y extensiones de Chrome no son compatibles con Chrome para Android. No tenemos planes de anunciarlo en este momento].
https://developer.chrome.com/multidevice/faq consultado el 06/09/2017
[56] https://developer.mozilla.org/en-US/Add-ons/SDK/Guides/Two_Types_of_Scripts
[57] https://www.w3.org/Security/wiki/Same_Origin_Policy
[58] http://w3c.github.io/html/webappapis.html

APIs que ofrece el Addon-SDK[59].

Mediante la comunicación entre estos dos tipos de scripts es posible manipular los componentes de la capa de presentación de una aplicación Web, fundamental para poner en práctica la Aumentación Web. Tal como con un script para una aplicación Web tradicional, se puede agregar, modificar y hasta eliminar nodos dentro mediante la interfaz DOM, que trata a los documentos HTML como una estructura jerárquica de árbol. Si esta manipulación se realiza sobre contenido de un documento externo inyectado en el DOM actual, es decir desde un contexto menos privilegiado, algunas propiedades y operaciones no son posibles de realizar. Por ejemplo, las propiedades expando[60] no son visibles. Pero para ello es posible acceder en forma transparente al mismo mediante waiveXrays[61] (llamado desde el lado del addon-script).

Teniendo entonces un total control sobre los objetos de la aplicación Web subyacente, es posible no solo manipular sus elementos directamente desde un addon-script, sino también agregarle archivos de estilo (CSS) o comportamiento (JS) al head del documento mediante el método appendChild, e incluso abrir documentos html que vienen integrados como parte de la extensión (ej. formularios). También es posible cargar scripts del lado del addon (loadSubScript), clonar objetos de contextos privilegiados en aquellos que no lo son (cloneInto), acceder al sistema de archivos (nsiFile) del dispositivo o evaluar código JavaScript dentro de un sandbox (evalInSandbox) gracias a la interfaz XPCOM62. Todas estas posibilidades hacen viable la creación de una capa de aumentación por sobre la de presentación de cualquier aplicación Web existente.

La herramienta de autoría fue implementada como una extensión para el navegador Firefox for Android[63], siguiendo el enfoque *restartless add-on*[64]. La primera versión fue desarrollada y testeada sobre la versión 41 de dicho navegador, que en ese momento se encontraba publicada como versión Nightly[65], esto es, una versión experimental de actualización diaria, apropiada para los contribuyentes del núcleo de Mozilla o aquellos desarrolladores que quieren utilizar sus nuevas características con anticipación. La extensión también corrió en forma satisfactoria sobre las versiones 38 y 39 del

[59] https://developer.mozilla.org/en-US/Add-ons/Firefox_for_Android/API

[60] aquellas añadidas a elementos del DOM mediante JavaScript, que no son parte de la especificación inicial del objeto. Por ejemplo, window.foo o document.bar.

[61] https://developer.mozilla.org/en-US/docs/Mozilla/Tech/Xray_vision

[62] https://developer.mozilla.org/en-US/docs/Mozilla/Tech/XUL/Tutorial/XPCOM_Interfaces

[63] https://www.mozilla.org/en-US/firefox/android/

[64] https://developer.mozilla.org/en-US/Add-ons/Firefox_for_Android/Walkthrough

[65] https://www.mozilla.org/en-US/firefox/android/nightly/all/

navegador, aunque los usuarios reportaron algunos errores sobre la versión 37. Las versiones 38 y 39 representaban las versiones estándar y beta ofrecidas en Google Play a la fecha, mientras que la versión 37 se trataba de una versión desactualizada del mismo.

Tras la evaluación con la primera versión de la herramienta y gracias al feedback de los participantes, se pudo construir una segunda versión de esta, que mantiene los mismos pasos del proceso de autoría, aunque hace uso de otras dependencias y soluciona problemas de presentación y usabilidad. Esta versión mejorada es la que finalmente se presenta a lo largo de esta sección. La misma fue probada en una *demo session* [14] sobre la versión 46 del mismo navegador Web móvil.

6.2. Aplicaciones MoWA

Antes de profundizar en los detalles respecto a cómo la herramienta soporta el proceso de autoría, es conveniente presentar primero la estructura de una aplicación MoWA. Tal como se muestra en la Figura 21, una aplicación MoWA puede estar suscrita a cualquiera de los sensores soportados por el framework y detectado en la plataforma sobre la cual corre *MoWA Authoring*.

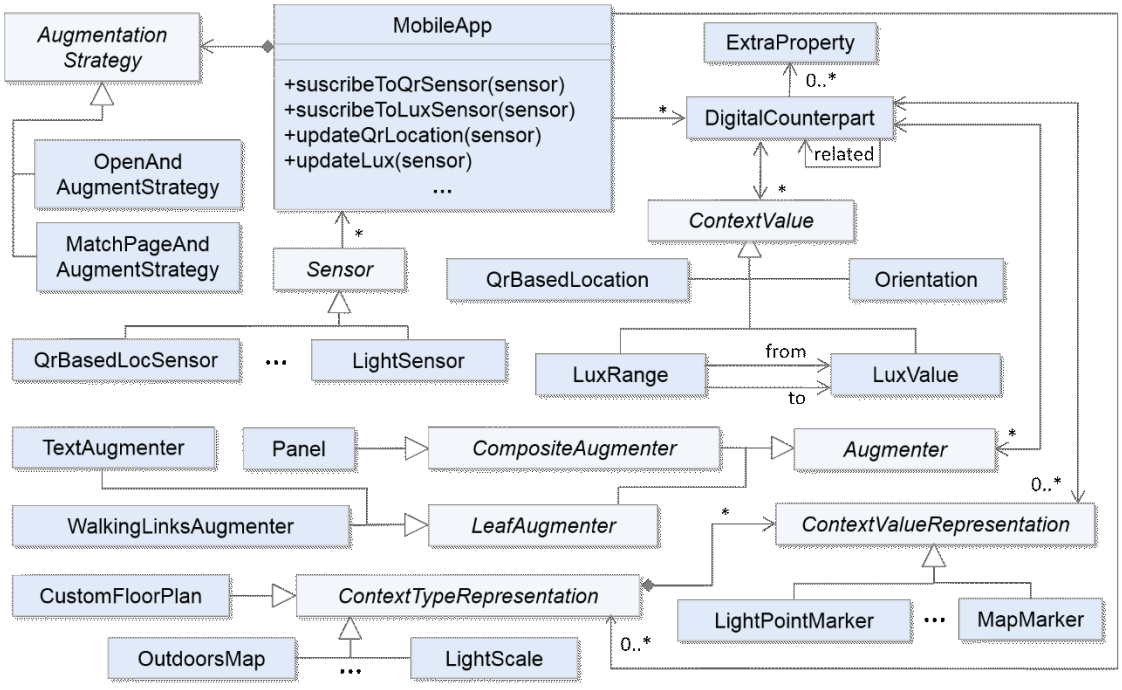

Figura 21. Diagrama de clases de una aplicación MoWA

Cuando se envía un mensaje de suscripción (ej. *suscribeToQrSensor*), la aplicación pide al sensor que la registre como un Observer [54] de él. De esta manera, el sensor puede notificar sobre cambios en el contexto percibido enviando un mensaje *update* (*updateQrLocation*, *updateLux*) a todos sus *listeners*. Este mensaje se refiere al enviado por una clase que juega el rol de «Concrete Subject» dentro del patrón «Observer» [54]

a sus observers (o *listeners*) cuando su estado cambia. Dichos cambios pueden ser iniciados manual o automáticamente. Por ejemplo, en el caso de un *QrLocationsManager* será manual, ya que el usuario debe interactuar con el dispositivo para indicar que se tiene que decodificar un código al que está siendo percibido mediante la cámara del dispositivo. Sin embargo, un *GpsBasedSensor* puede observar automáticamente los cambios en la posición del usuario utilizando la API del navegador[66].

Cuando un nuevo valor del contexto es notificado a la aplicación, la misma delega el proceso de aumentación a una estrategia [54] concreta; en una subclase de *AugmentationStrategy*. Estas pueden ser:

1. *MatchPageAndAugmentStrategy*. Es la más simple de ellas, puesto que solo verifica que la página Web actual coincida con un patrón de URLs y, en caso de hacerlo, lleva a cabo la aumentación, tal como se muestra en la Figura 22.

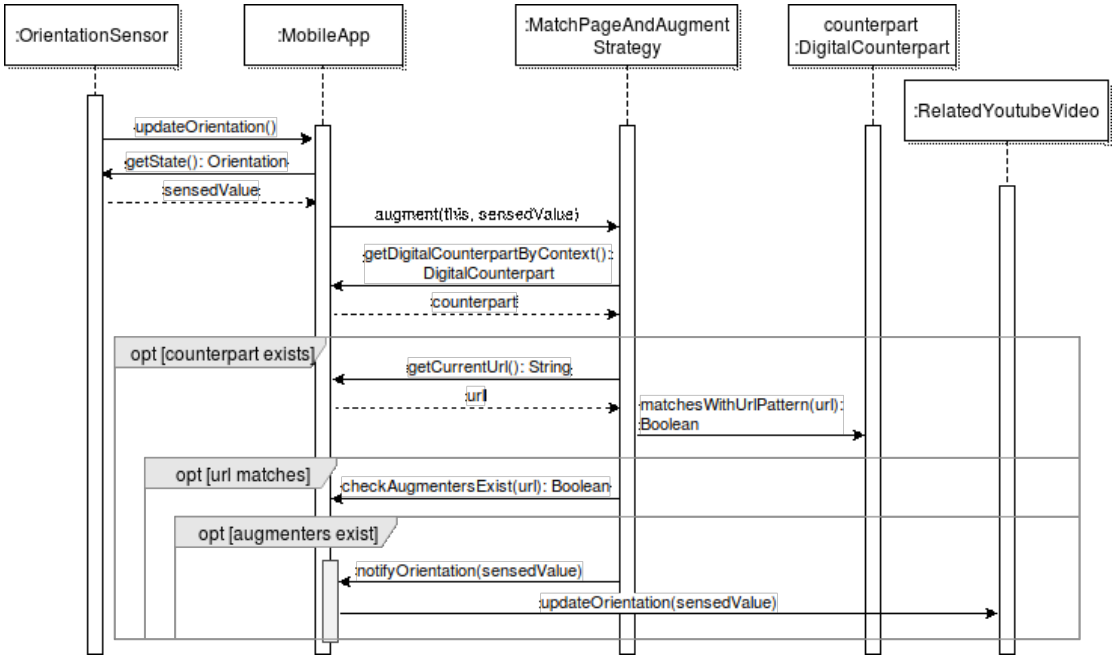

Figura 22. Actualizando aumentación mediante MatchPageAndAugmentStrategy

2. *OpenAndAugmentStrategy*. En este caso, la estrategia llevará a cabo la aumentación siempre que el valor notificado por un sensor coincida con uno de los valores de interés definidos en una aplicación MoWA, tal como se muestra en la Figura 23. Para ello, se compara el valor percibido del entorno contra los valores de interés de la aplicación. Si coincide, la URL correspondiente

[66] mediante el método *watchPosition*: https://dev.w3.org/geo/api/spec-source.html#watch-position

(asociada a una contraparte digital, *DigitalCounterpart*) es cargada en el navegador. Una contraparte digital tiene un conjunto de *augmenter*s a ejecutar cuando la página es cargada (ej. un Panel, un *WalkingLinksAugmenter*, etc.) y, además, puede tener una colección de propiedades personalizadas (*ExtraProperty*) definidas por el usuario. El primer escenario descriptivo del Capítulo 6 y el escenario del Capítulo 7 están basados en esta estrategia.

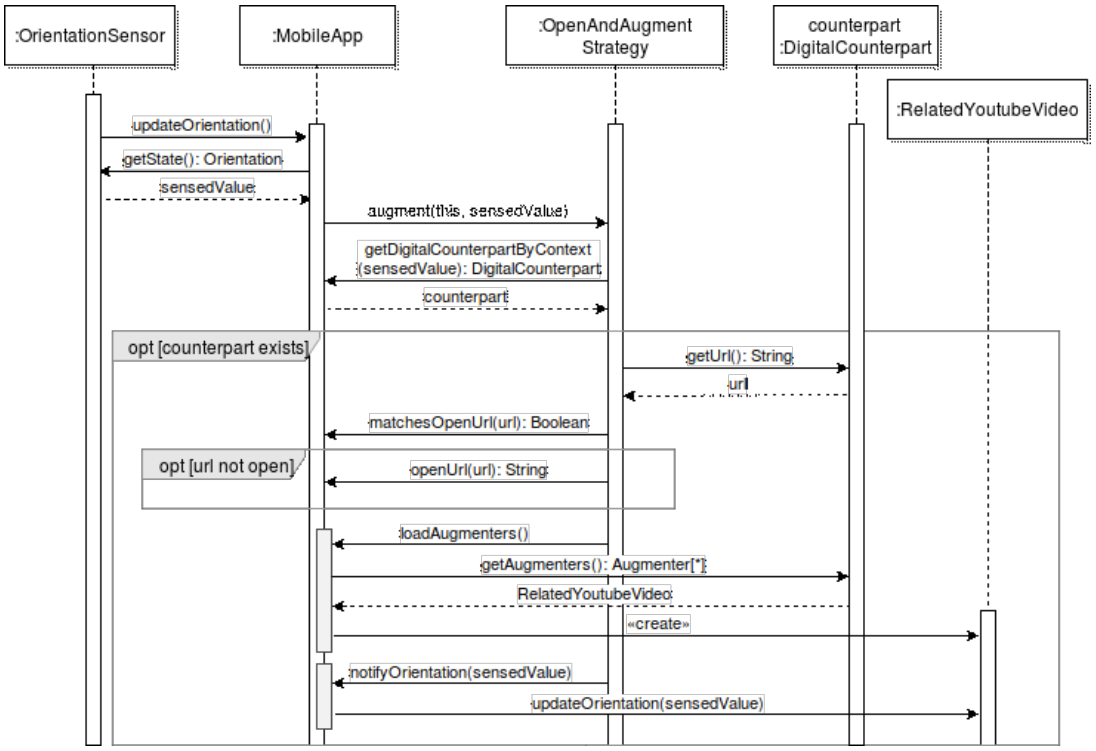

Figura 23. Actualizando aumentación mediante OpenAndAugmentStrategy

Independientemente de la estrategia, una contraparte digital puede tener asociada una representación de un valor de contexto (*ContextValueRepresentation*), que será contenida y visualizada dentro de una representación (*ContextTypeRepresentation*. Las subclases concretas de un *ContextTypeRepresentation* implementan una representación visual contenedora, tal como un mapa o una escala, sobre la cual se visualizarán representaciones de valores concretos del contexto, como un punto dentro de una escala o un marcador en el mapa. La razón de que estos componentes no sean obligatorios es que solo algunos *augmenters* necesitan de ellas para implementar sus características. Por ejemplo, el *TextAugmenter* no precisa de una representación del contexto para cumplir con su objetivo, le basta con poder agregar un *div* en alguna posición del cuerpo del documento para mostrar el texto con el que fue configurado. Sin embargo, el objetivo de un *WalkingLinksAugmenter* es indicar al usuario cuál es el siguiente PDI y asistirlo en su recorrido. La aumentación consiste en una lista con uno o varios PDI en forma de links que, al ser disparados, muestran las indicaciones para llegar a ellos. Una forma de mostrarlas es mediante un mapa que muestre la posición del usuario y la del siguiente PDI con marcadores.

6.3. Sensores soportados

Lo sensores representan un punto de extensión dentro *MoWA framework*, pensado para que los desarrolladores pueden incluir nuevas formas de percibir información del contexto fácilmente y utilizarse en nuevas formas de aumentaciones. Para demostrar su flexibilidad, en la actual versión de la herramienta de autoría se han implementado sensores de orientación (*OrientationSensor*) y de nivel de luminosidad (*LuxSensor*), como se muestra en la Figura 24. Sin embargo, el framework es extensible con otros tipos de sensores, tales como un sensor del nivel de presión sonora, de carga de batería o, inclusive, se puede extender un sensor existente, como el de orientación, para crear un sensor de gestos. Lo requerido es que, por cada sensor que se extienda, debe existir el tipo de contexto (*ContextValue*) que notificará (ver *AbstractSensor* » *lastSensedValue* en la figura) a sus *listeners*, y también un *builder* que permita la definición de los valores del tipo de contexto que soporta. Sobre este último, un mayor nivel de detalle se presenta en la Sección 6.5.

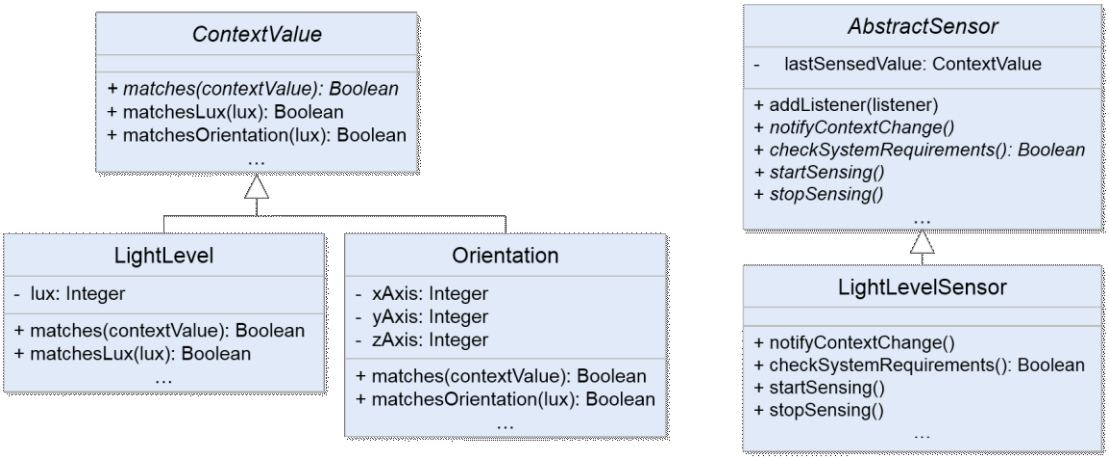

Figura 24. Valores del contexto y sensores

La especificación del sensor, el tipo de contexto y el builder deben hacerse dentro de un mismo directorio. Sin embargo, mientras el sensor y el tipo de contexto deben ser especificados en un archivo «.js», el builder debe hacerse por separado, en un archivo «.jsm» (JavaScript module). Dentro del primero no debe hacer un uso de código privilegiado (componentes del SDK con alto nivel de acceso), puesto que los sensores serán instanciados como *content-script* en: 1) el formulario de definición de valores del contexto de interés para la aplicación, y 2) dentro de la página a aumentar. El archivo que especifica el *builder* es instanciado dentro del contexto de mayores privilegios, desde donde es posible hacer un uso libre del SDK del navegador, pues es ejecutado como un *addon-script*. Las instancias de este tipo de script, además, se mantienen hasta que el navegador es cerrado. Los sensores, en cambio, se instancian por cada documento HTML abierto que los requiera.

Adicionalmente, todos aquellos formularios requeridos para la configuración de los valores de interés (asistencia proporcionada por el builder) deben crearse dentro del directorio «augmenters» dentro de «content/data». Esto es así porque el último directorio es el registrado como content-package[67] con contenido accesible desde la extensión. Esto permite que se cree un alias para poder hacer referencia a los archivos que vienen como parte de la extensión, para poder resolver su URI y ser cargados en un contexto determinado. Por ejemplo, para cargar el formulario "sensors/config-luxes.html" mediante *window.BrowserApp.loadURI* [68]en la ventana actualmente abierta, es preciso indicar que se trata de una Chrome URI (chrome://mowa/content/data/ sensors/config-luxes.html), para que el SDK resuelva la ubicación específica dentro del dispositivo en tiempo de ejecución.

Si el sensor requiere dependencias, se debe crear una carpeta con el mismo nombre que el archivo conteniendo todos los recursos necesarios para la correcta ejecución del sensor. Por ejemplo, el *GpsSensor* no requiere más que de las APIs del propio navegador, pero un sensor de QRs (tal como el utilizado en la primera versión de la herramienta) es necesario resolver no solo como capturar una imagen, sino también cómo decodificarla, para lo cual es apropiado hacer uso de librerías de terceros, como jsqrcode[69]. En la cabecera del archivo principal del sensor se puede incluir metadatos, como ser el nombre del creador, el *namespace*, o la superclase del sensor, para que la extensión pueda instanciar los sensores encontrados en el directorio de manera adecuada.

Finalmente, cabe destacar que, al momento de su inicialización, un sensor debe verificar que tiene acceso a todos los componentes del dispositivo que necesita utilizar. Por ejemplo, para detectar si se tiene acceso a la cámara o al micrófono, se debe comenzar verificando si el navegador soporta *navigator.mozCameras*. En caso de no soportarlo, el sensor pasa a estar deshabilitado. Esto es importante no solo para suscribir *listeners* que nunca serán notificados, sino para deshabilitar esta opción desde la herramienta de autoría y otros componentes asociados, por ejemplo, el *QrBasedLocationSensor* debe deshabilitar el submenú que permite acceder a su escáner.

6.4. Augmenters soportados

Para verificar la factibilidad técnica de las posibles aumentaciones mencionadas, se desarrollaron un conjunto de unidades de aumentación [13]. Estos augmenters

[67] https://mdn-jake.hashbase.io/en-US/Chrome_Registration.html#contentaccessible
[68] https://developer.mozilla.org/en-US/Add-ons/Legacy_Firefox_for_Android/API/BrowserApp/loadURI
[69] https://github.com/LazarSoft/jsqrcode

contemplaron comportamientos básicos, como mostrar una imagen o texto ingresados por el *productor* o automáticamente extraídos desde otra página (*TextAugmenter, ImageAugmenter*), y funcionalidad más compleja como, por ejemplo:

- mostrar los siguientes PDI en un recorrido (*WalkingLinksAugmenter*),
- mostrar qué ubicaciones el usuario ha frecuentado (*VisitedLocations*) o cuáles le quedan por visitar (*UnvisitedLocations*),
- ajustar automáticamente el volumen de un audio (*AudioVolumeLeveler*),
- reproducir/pausar un video concreto (*LightsCameraAction*) cuando un cambio repentino en el nivel de luz es percibido, deshabilita temporalmente la interacción con todos los audios y videos cuando el nivel de carga de la batería es menor al 25% (*DontLetMeWaste*),
- recuperar resultados desde la UI de un motor de búsqueda existente (*RelatedDomainInfoItems*).

Cualquier augmenter puede ser reutilizado. Por ejemplo, el *PoiAugmenter* hace uso de un *TextAugmenter* para mostrar la descripción del PDI como de un *ImageAugmenter* para presentar una imagen descriptiva. En cualquier caso, proveer a los productores con un nuevo augmenter requiere:

1. Extender alguna de las clases existentes de augmenters.
2. Crear un builder que permita la correcta instanciación de dicho augmenter.
3. Crear los formularios que el builder manipulará, en una ubicación específica para que la extensión pueda cargarlos sin inconvenientes (content/data/augmenters).

Por ejemplo, en la Figura 25 se muestra cómo se extiende la clase *SingleContainerAugmenter* con el augmenter *RelatedYoutubeVideo*.

El protocolo de un augmenter está conformado por los métodos *execute*, *undo* y un mensaje por cada tipo de contexto soportado en el framework. Cuando una aplicación es notificada por un sensor sobre un cambio de valor en el contexto, la aplicación comunica este nuevo valor a sus augmenters. Por ejemplo, si un nuevo lux es sensado, la aplicación enviará el mensaje *onLuxChange* a sus augmenters, y cada cual sabrá qué adaptar en relación a dicho valor.

Todo augmenter debe contar con –al menos– una implementación concreta de los mensajes abstractos de la clase Augmenter. El método *execute* es un *template method* en el cual se define el esqueleto de ejecución de la aumentación. Una particularidad con los augmenters es que su correcta ejecución depende, básicamente, de dos factores:

1. Una correcta configuración, especialmente si sus propiedades fueron asignadas por usuarios finales. Todas las propiedades requeridas deben tener un valor y, el mismo, debe ser acorde a lo esperado (ej. si se espera una URL, la misma debe ser una cadena válida y, además, la URL debe existir).
2. Cambios en su entorno de ejecución como, por ejemplo, la desaparición de determinados nodos o su cambio de posición en el DOM que afecten el posicionamiento de la aumentación o la extracción de información.

Es por ello que la ejecución debe asegurarse de verificar esto antes de llamar al método *buildAugmentation*, responsable de la aumentación concreta. Este método debe ser redefinido por las subclases, al igual que el método *undo*. Por ejemplo, *SingleContainerAugmenter* provee la implementación específica para augmenters que agregan componentes estrictamente visuales y que presentan su contenido en un contenedor principal. Todo agregado es agregado dentro de un contenedor principal que luego es insertado en el DOM, de modo que el mensaje *undo* no necesita más que remover dicho elemento del árbol para deshacer los cambios (*removeContainer*). El método *buildAugmentation*, en este caso, llama a *buildContainer* para obtener el div contenedor sobre el cual se insertará cualquier *DomElement* que las subclases permitan construir mediante *buildContent*. En concreto, el RelatedYoutubeVideo devuelve una instancia de *HTMLVideoElement*.

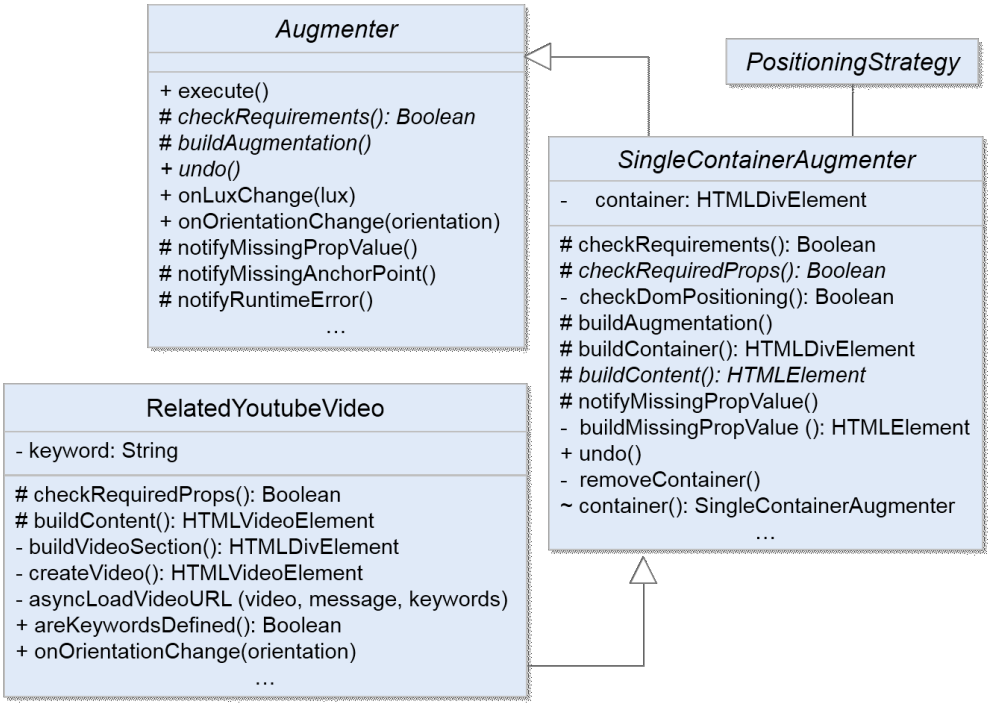

Figura 25. Extendiendo augmenters

El método *checkRequirements* es redefinido por el *SingleContainerAugmenter*, el cual verifica dos cosas:

1. *checkDomPositioning*: que la información proporcionada baste para llevar a cabo el posicionamiento (ej. que exista un elemento de referencia para insertar a la misma altura. Depende de la estrategia de posicionamiento utilizada).
2. *checkRequiredProps*: que las propiedades requeridas tengan un correcto valor asociado. Sin embargo, este método deberá ser redefinido por las subclases, ya que depende de las propiedades específicas necesarias para construir cada aumentación.

Los mensajes de error en *Augmenter* son, por defecto, una notificación creada mediante un *alert*. Sin embargo, pueden ser redefinidos, como en el caso del *SingleContainerAugmenter* que inserta el mensaje en un div.

Finalmente, cada augmenter debe saber reaccionar ante un cambio de valor de algún contexto. Por ejemplo, el *RelatedYoutubeVideo* redefine *onOrientationChange* de modo que el contenedor sea ocultado cuando la orientación del dispositivo sea diferente a la «de interés» para la aumentación.

De manera transversal, cabe señalar que todos los *augmenters* pueden hacer uso de un propio storage, que les permite guardar pares de clave-valor entre sesiones y mantener el estado de las aumentaciones persistentes en caso de abandonar la sesión en el navegador. En algunos casos puede ser necesario reiniciar los valores en el storage (ej. para volver a utilizar la asistencia de un determinado tour). Todos cuentan con un mecanismo que permite reiniciar sus valores y también con la capacidad de recuperar y manipular contenido externo.

Respecto a los builders de augmenters, su definición debe ser especificada en un archivo «.jsm» (JavaScript module) desde donde se tiene un alto nivel de acceso. El builder no es más que una clase que carga fomularios para solicitar al usuario la información necesaria para instanciar correctamente un augmenter. En el caso del *RelatedYoutubeVideo*, por ejemplo, se debe permitir al usuario seleccionar un conjunto de palabras clave a utilizar en la búsqueda del video, tal como se presenta en la Sección 6.6.2.

6.5. MoWA Authoring

En esta sección se presenta el diseño que soporta el proceso de creación una aplicación MoWA. Se propone un conjunto de clases denominadas «builders» con la responsabilidad de permitir al usuario final, de forma transparente, completar lo necesario para la instanciación de cada componente de una aplicación MoWA.

Con el fin de mostrar las relaciones entre las clases de una aplicación (de la Figura 21) y sus respectivos *builders*, la Figura 26 presenta los *builders* requeridos para

instanciar y configurar una aplicación muy sencilla, suscrita a un único sensor. Sin embargo, es posible contemplar la extensión y utilización de otros *builders* para valores de contexto de interés y *augmenters*, como explicaremos posteriormente. En la figura, los *builders* están representados con un color de fondo más obscuro.

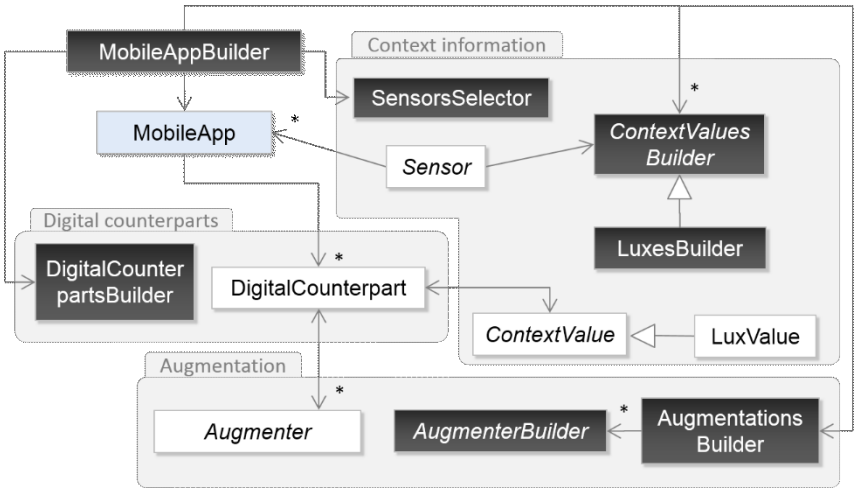

Figura 26. Soporte al proceso de autoría de una aplicación MoWA

La clase principal en este diseño es *MobileAppBuilder*, es la encargada de orquestar la construcción de una aplicación. Para ello, debe solicitar al usuario final que ingrese determinada información, como el nombre de la aplicación, el nombre, URL y las propiedades de cada contraparte digital, qué luxes interesan a la aplicación, que *augmenters* se utilizarán y qué mostrarán. Para ello, este *builder* principal cuenta con tres colaboradores que se encargarán de una parte específica del proceso:

- *SensorsSelector*, muestra al usuario una lista de sensores disponibles y le solicita seleccionar aquellos que desea utilizar en su aplicación. Posteriormente, el *MobileAppBuilder* instanciará, por cada sensor seleccionado, la subclase de *ContextValuesBuilder* correspondiente.
- *ContextValuesBuilder*, sus subclases se encargan de solicitar al usuario la especificación de un conjunto de valores de contexto que le interesan para su aumentación. Por ejemplo, si le interesa mostrar algún mensaje cuando está en un ambiente obscuro, debería agregar un lux cuyo valor sea bajo, como un «5». Cada *builder* tiene la capacidad de cargar un archivo «.html» que le permita al usuario seleccionar un conjunto de valores del contexto (por ejemplo, luxes), sin restricciones sobre qué incluir en él. De esta manera, se podrían incluir mapas para la definición de nuevas coordenadas o simplemente formularios con inputs que se autocompleten con valores del contexto. En cualquier caso, el *builder* debe soportar la definición de valores extraídos automáticamente del contexto actual del usuario y, para ello, debe estar suscripto al sensor correspondiente. Una vez que el usuario termina de definir los valores del contexto que le interesan, los mismos serán utilizados posteriormente como valor del contexto de alguna contraparte digital.

- *DigitalCounterpartsBuilder*, se encarga de la definición de las contrapartes digitales de la aplicación. Para ello, solicita al usuario que se ingrese una URL a aumentar, que se le dé un nombre y, opcionalmente, que se le asocien un conjunto de propiedades extra que podrán ser utilizadas por los augmenters. Por ejemplo, si se está creando un artículo «X» de Amazon como contraparte digital, podría ser interesante contar con su precio o ubicación como propiedades a utilizar desde una aumentación que las compara con artículos en Mercadolibre o Ebay. Cuando el usuario termina de especificar contrapartes digitales, el *MobileAppBuilder* envía esta colección al *AugmentationsBuilder*.
- *AugmentationsBuilder*, en base a una colección de contrapartes digitales recibidas, este builder debe permitirle al usuario especificar un conjunto de augmenters a ejecutar sobre la capa de presentación de las mismas. Los augmenters se deben presentar en relación a los sensores utilizados; carecería de sentido permitirle utilizar aquellos que no puede ejecutar correctamente. Por cada augmenter disponible, el *AugmentationsBuilder* delega a cada *AugmenterBuilder* su construcción. Las subclases de este último conocen qué información solicitarle al usuario para la instanciación de un augmenter concreto.

A continuación, se presenta el soporte de software mediante dos escenarios descriptivos en donde puede apreciarse cada uno de estos builders en marcha, cumpliendo con los pasos del proceso de autoría (ver Sección 5.4).

6.6. Escenarios descriptivos

En esta sección se presentan dos escenario descriptivos [60], tanto para presentar la herramienta propuesta como para demostrar su utilidad. El primero de ellos se enmarca en el dominio del turismo y la aplicación es creada con la primera versión de la herramienta, utilizada en la primera publicación del enfoque para usuarios finales [13]. El segundo, presenta un escenario en el dominio del cine y la aplicación se construye con la segunda versión de la herramienta.

La diferencia entre ambas versiones es sustancial en términos del proceso que soporta: el mayor cambio en este aspecto es que en la primera versión, la definición de la representación del contexto estaba atada a la definición de valores del contexto de interés para la aplicación; flexibilidad que se ganó en la segunda versión. Sin embargo, se produjo un gran cambio a nivel soporte tecnológico: si bien se siguió utilizando la misma versión del SDK de Firefox, el motor de XForms utilizado en la primera versión fue descartado y reemplazado por simples formularios HTML. El motivo fue poder lidiar más fácil con la persistencia y el estilo del documento, y evitar un esfuerzo extra en modificar los permisos de ejecución de su motor al ser cargado como parte de la

extensión.

Si bien actualmente la herramienta ha evolucionado en correspondencia con el contenido de los capítulos 5 y 6, es importante mostrar el uso de la primera versión de la herramienta, ya que fue parte del experimento realizado y facilitará una mejor comprensión del mismo.

6.6.1. Asistencia a la navegación física en un museo

El primer escenario corresponde al dominio del turismo y resuelve uno de los tipos de escenarios motivadores de la Sección 4.1.6. En este escenario se muestra cómo un guía de museo puede crear una aplicación MoWA de Hipermedia Móvil para la asistencia de un recorrido *indoor*. Este mismo escenario se utilizó además en el experimento, que se presenta en el Capítulo 7.

El proceso de autoría, presentado en la Sección 5.4, es similar en otros dominios y escenarios. La elección de este el dominio fue debido a su nivel de complejidad en comparación a otro tipo de aplicaciones soportadas; el usuario final debe ser capaz no sólo de configurar sensores, valores de contexto y *augmenter*s, sino también configurar características de navegación en dos niveles: el digital y el físico. Esto último requiere la definición de un espacio de representación, algunos puntos de interés con información relacionada, y sus enlaces, para asistir en el recorrido entre ellos.

El museo elegido para realizar la experiencia fue el Museo de Ciencias Naturales de la Universidad Nacional de La Plata[70]. El mismo cuenta con un paseo para una de sus colecciones, llamado «Siguiendo a Darwin en el Museo de La Plata». En su sitio Web, tal como se muestra del lado derecho de la Figura 27, el recorrido[71] se presenta mediante un mapa con marcadores que representan las piezas del museo que deberían ser visitadas en un determinado orden. Físicamente, en el Museo, cada una de dichas piezas está asociada a un código QR que redirige a una página de Wikipedia. El lado izquierdo de la misma figura muestra una captura tras escanear el código QR de la primera pieza del recorrido.

Una aplicación que integre la información de ambos sitios, aumentando uno de ellos, sería una manera de mejorar la experiencia de los visitantes en el museo. Como la única contraparte digital de las piezas del museo son los artículos de Wikipedia, una posibilidad es aprovechar el acceso a esta contraparte y aumentarla con la información provista por el propio museo (que no está asociada a los objetos físicos, se debe llegar navegando la Web).

[70] http://www.museo.fcnym.unlp.edu.ar/
[71] http://www.fcnym.unlp.edu.ar/museo/educativa/darwin/darwinenmuseo.html

Figura 27. Información del sobre el recorrido y sus piezas

Ante esta situación, un guía turístico –jugando el rol de *productor*– puede llevar a cabo el proceso de creación de una aplicación de aumentación con *MoWA Authoring*. Este caso fue presentado con la primera versión de *MoWA Authoring* [13], y su construcción implicó:

1. Configurar los datos básicos de la aplicación, tal como se muestra en la Figura 28. Una vez habilitado el modo de creación (S1) y el modo de pantalla completa, el *productor* elige crear una nueva aplicación (S2) y establece un nombre para la aplicación (S3).
2. Seleccionar la estrategia de aumentación. La aumentación debe ser configurada con una de las estrategias de aumentación disponibles. Este paso se muestra en «S4» de la Figura 28.
3. Seleccionar el/los sensor/es del contexto. En «S5, S6», el usuario elige entre los sensores disponibles en su navegador (según las características móviles que posea). Dichos sensores se muestran en relación con los tipos de contexto que notifican (por ejemplo, una ubicación, un nivel de luz, etc.). En la figura, el usuario selecciona la ubicación y es por ello que, en «S7», se muestran dos sensores disponibles para este tipo de contexto: el de GPS y el basado en QRs.
4. Definir una representación del tipo de contexto. En este paso, el usuario final elige si desea definir la representación o no. En el paso «S8», la herramienta ofrece la posibilidad de elegir entre las alternativas disponibles y el usuario selecciona el «Plano 2D personalizado», que debe ser configurado con una imagen de fondo y un nombre, tal como en el paso «S9» de la Figura 29. Se debe tener en cuenta que este paso y el siguiente se encuentran actualmente invertidos respecto a su orden según el diseño de la nueva herramienta (ver secciones 5.4 y 6.5).

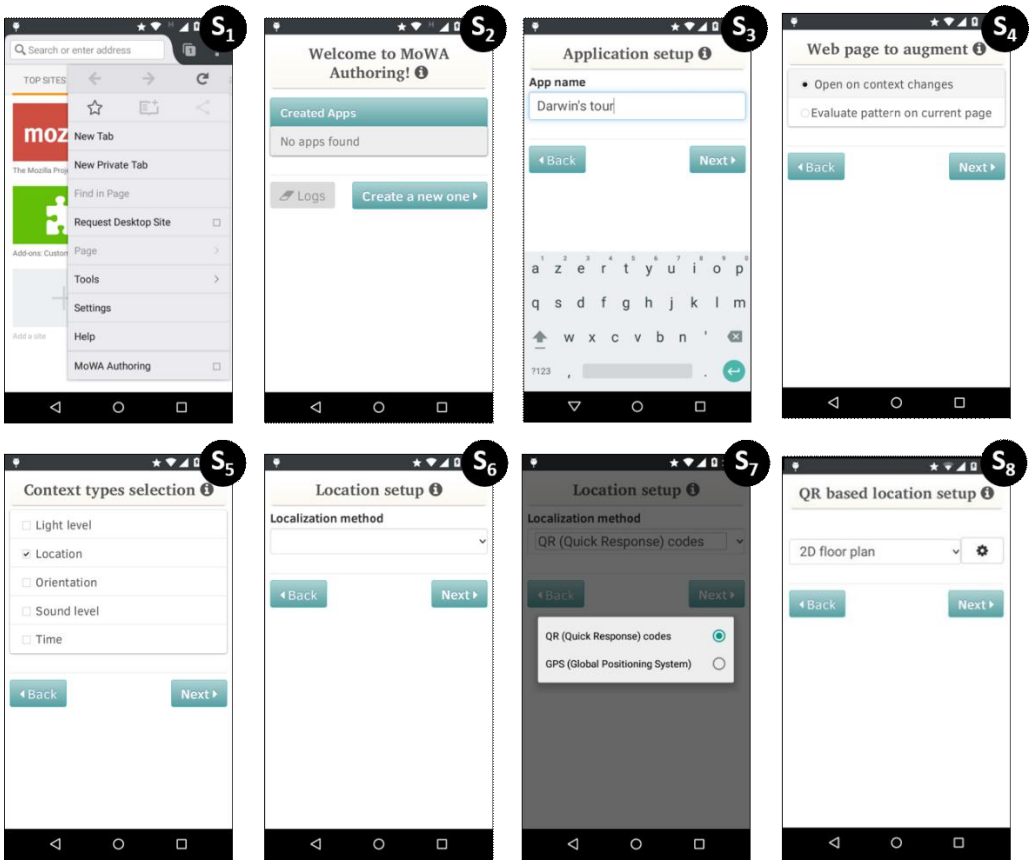

Figura 28. Configurando el nombre de la aplicación y los sensores a observar

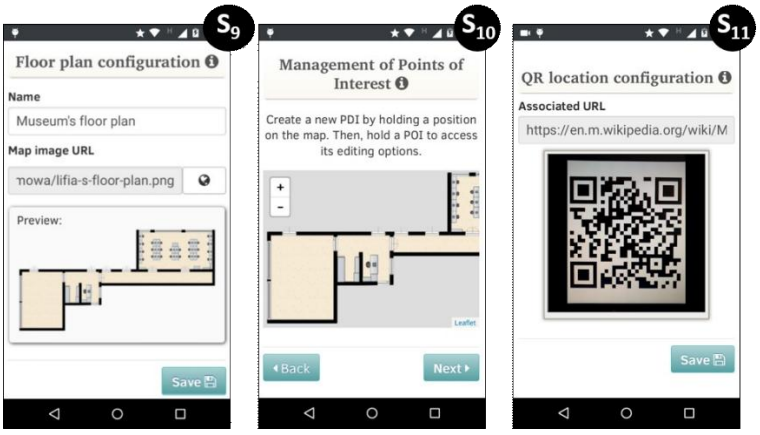

Figura 29. Definiendo una representación del contexto y sus valores

5. Definir valores de contexto de interés para la aplicación. Como se ha seleccionado un sensor de ubicación basado en QR y un plano como forma de representación del contexto, ahora el usuario debe definir qué ubicaciones quiere soportar. En la Figura 30 se puede observar que dicha tarea se logra creando marcadores en un mapa, tal como en el décimo paso (S10). Para agregar una nueva posición, el *productor* toca la pantalla y mantiene presionada una posición en el mapa. En respuesta, se crea un marcador en dicha posición, que requiere escanear un código QR (S11) para poder instanciar una *QrBasedLocation*.

6. Definir una contraparte digital. Una vez que se define una ubicación

(*QrBasedLocation)*, el usuario es automáticamente dirigido a la definición de una contraparte Digital. La misma implica escribir un nombre, una URL (en este caso, escaneando un código QR para obtenerla) y, opcionalmente, definir algunas de sus propiedades, como se muestra en la Figura 30. Algún contenido externo es definido como propiedad del PoI en «S13, S14, S15». En el último paso, el usuario definió dos propiedades: «poi-desc» y «poi-pic», que serán utilizadas por un *augmenter* en etapas futuras. Una vez que se definen estos parámetros, el *productor* accede nuevamente a la gestión de propiedades externas (S13). Cuando el usuario define más de una contraparte digital, el usuario puede, adicionalmente, comenzar a conectar los marcadores mediante rutas. Las mismas se crean manteniendo presionados los marcadores, lo cual resulta en la aparición de un menú contextual, donde el usuario puede elegir entre establecer el marcador como origen o destino de la ruta. El paso «S16» es la vista resultante después de que todas las contrapartes digitales fueron definidas y conectadas.

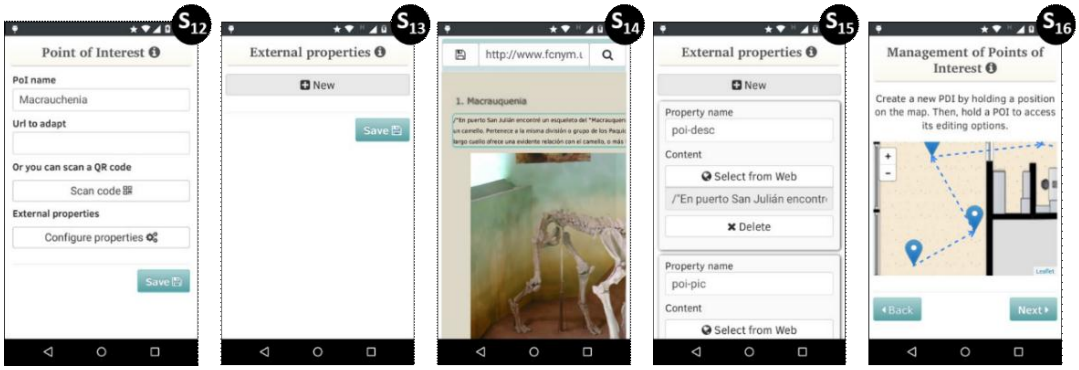

Figura 30. Configuración de contrapartes digitales

7. Definir *augmenters*. Como se muestra en la Figura 31, el *productor* puede añadir múltiples augmenters por cada contraparte digital. En «S17», la herramienta cargó automáticamente la URL –previamente definida para la contraparte digital– en un *iframe*. En la parte inferior hay un *carrousel* con aumentadores disponibles. Al hacer clic en alguno de ellos, es posible instanciar un *builder* en el documento de la página Web cargada. Este *builder* tiene una representación visual dentro del documento; inicialmente se trata de un elemento flotante en el centro de la pantalla que, posteriormente, debe ser reposicionado en relación a otro elemento DOM existente, mediante el gesto «arrastrar y soltar», tal como en «S18, S19». El elemento de referencia sirve como base para generar un *XPath* que servirá para posicionar el contenido del aumentador en posteriores accesos a la página. Si la herramienta falla al evaluar el *XPath* de un aumentador, se posiciona como un elemento flotante en la posición predeterminada (zona central de la pantalla), para que el usuario pueda reposicionarlo. Además de la posición,

el usuario debe configurar otros parámetros específicos de cada augmenter, como se muestra en «S20, S21». Una vez configurado, el augmenter puede ser correctamente previsualizado, como en «S22». Cada subclase de *AugmenterBuilder* implementa su propio proceso de configuración para recopilar los datos requeridos para la ejecución del aumentador, es por ello que los pasos «S20» y «S21» son diferentes para configurar otros augmenters; la Figura 32 muestra la configuración de un *PoiAugmenter* en «Sa» mientras que la de un *WalkingLinksAugmenter* en «Sb». Al final de esta etapa, el usuario puede previsualizar todos los aumentadores juntos y apreciar el resultado de la capa, como en el paso «Sn».

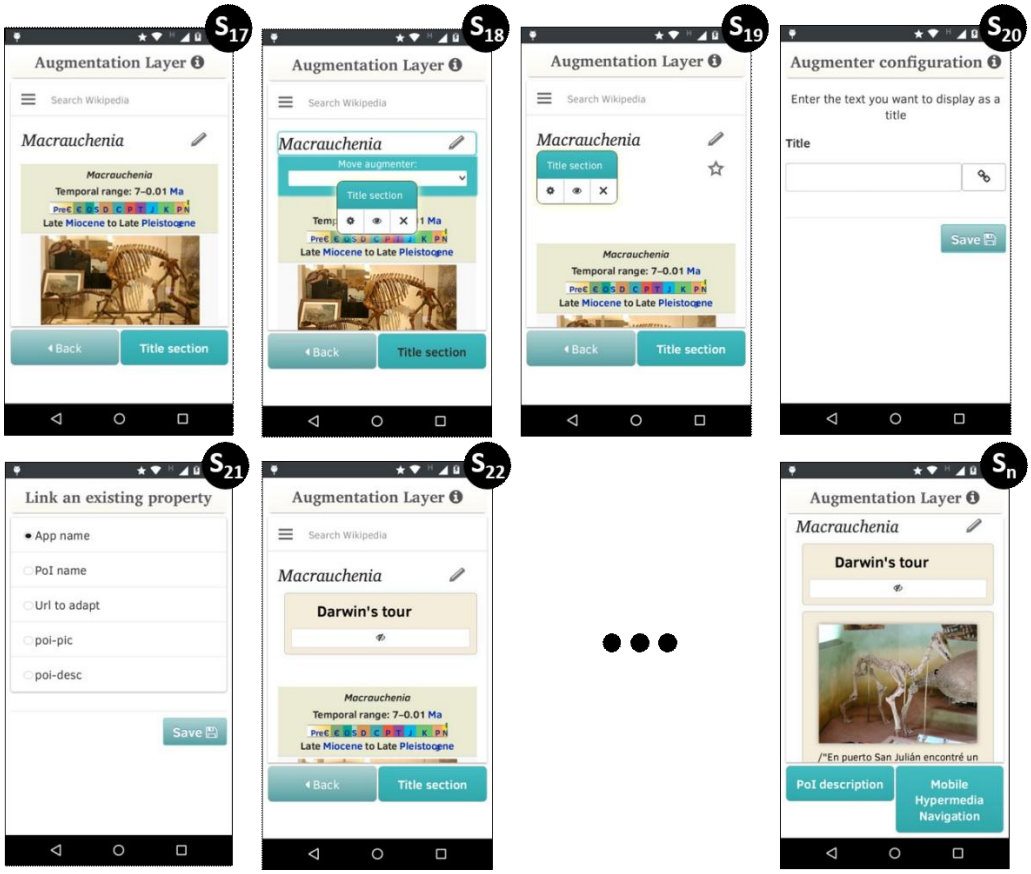

Figura 31. Configuración de un augmenter

Los aumentadores que utilizados en este escenario fueron:

- TitleSection, que muestra un título en la parte superior de la aumentación.
- PoiAugmenter, que presenta información específica (descripción e imagen) de la pieza en cuestión, extraída del sitio Web oficial del Museo.
- WalkingLinksAugmenter, que asistirá al usuario *consumidor* mientras recorre el Museo, mostrando un mapa con una ruta entre su posición actual y la próxima pieza a visitar en el tour.
- VisitLocations, que muestra una lista de piezas aún no visitadas por el usuario.

- UnvisitedLocations, que presenta las piezas que el usuario ya ha visitado.

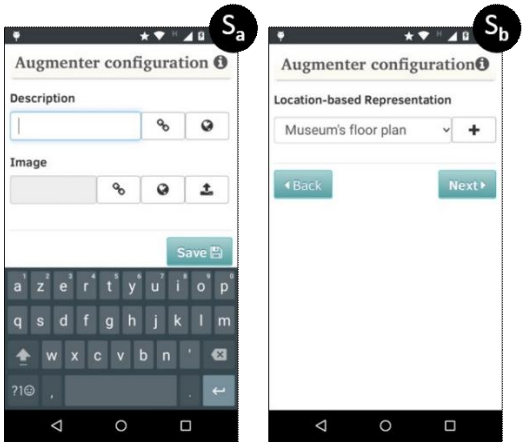

Figura 32. Configuración especializada por cada augmenter

Como resultado del proceso, se obtiene una especificación que hace posible la instanciación de una aplicación MoWA, y el *productor* puede compartirla con los *consumidor*es. Las aumentaciones creadas para un PDI pueden ser reutilizadas para otras contrapartes digitales; como el posicionamiento es por XPath, la misma puede ser evaluada en más de un artículo de Wikipedia. Y como algunos augmenters pueden ser configurados con propiedades de la contraparte digital, como el paso «S21» de la Figura 31, el valor es recuperado por dicha instancia.

6.6.2. Tráileres relacionados a artículos de Wikipedia

Con la finalidad de demostrar la flexibilidad del enfoque para ser adaptado a otros dominios, no solo al turismo, se presenta un nuevo escenario en el que se pretende integrar contenido de dos sitios: YouTube y Wikipedia. Adicionalmente, de manera similar al escenario motivador en 4.1.2, será requisito que las aumentaciones respondan de acuerdo con la orientación del dispositivo móvil.

En este escenario, un cinéfilo y miembro activo de Wikipedia decide unir sus intereses y se propone encontrar una nueva forma de descubrir producciones (cortos, series, películas) y enriquecer la enciclopedia; quiere poder recuperar, automáticamente, aquellos films relacionados a cada artículo de Wikipedia que visita. Pero solo quiere que esto sea así cuando su teléfono está posición *portrait* y a aproximadamente 90 grados sobre su eje horizontal.

El objeto de aumentación serán los artículos de Wikipedia y, sin importar cuál de ellos sea cargado en el navegador del *consumidor*, deberá ser aumentado con un tráiler que se corresponda al título del artículo. Por ejemplo:

- Si se encuentra visitando el artículo «José de San Martín» (que presenta un personaje histórico argentino) y se realiza una búsqueda en YouTube con

«“trailer” + José de San Martín» como palabras clave, el primer resultado de la lista es un tráiler de «Revolución. El cruce de los Andes».

- Si se encuentra visitando el artículo «Tormenta de espadas» (que presenta un libro) el primer resultado tras la búsqueda de su título en adición a la palabra «tráiler» en YouTube, es un tráiler de la tercera temporada de la serie «Game of Thrones».
- Si se encuentra visitando el artículo «Nueve reinas» (que presenta una película) el primer resultado tras la búsqueda de su título en adición a la palabra «tráiler» en YouTube, es un tráiler de la película de nombre homónimo. Este ejemplo se ilustra en la Figura 33.

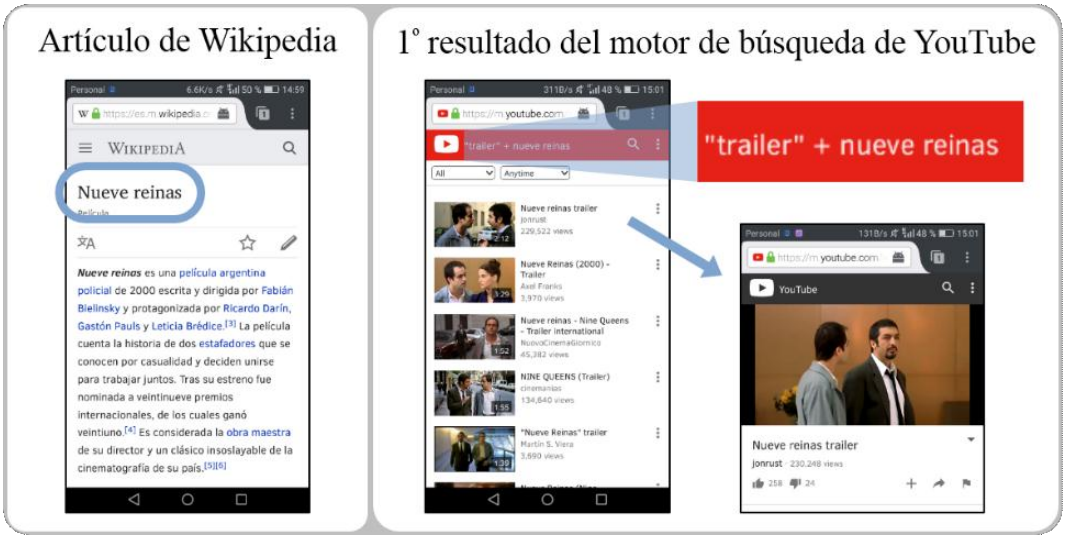

Figura 33. Recuperando un tráiler de acuerdo con el título de un artículo Wikipedia

Como se puede apreciar, es posible recuperar tráileres a partir de títulos de Wikipedia, sin importar si el artículo trata de una película o no. El objetivo de este escenario es construir una aumentación que permita recuperar estos tráileres de manera automática, y mostrarlos siempre y cuando el usuario se encuentre próximo a su casa y conectado a WIFI.

Haciendo uso de la segunda versión de MoWA [14], es necesario:

1. Configurar los datos básicos de la aplicación, tal como se muestra en la Figura 34. Se habilita el modo de construcción en pantalla completa (S1), luego, el *productor* elige crear una nueva aplicación (S2) y establece un nombre para la misma (S3).

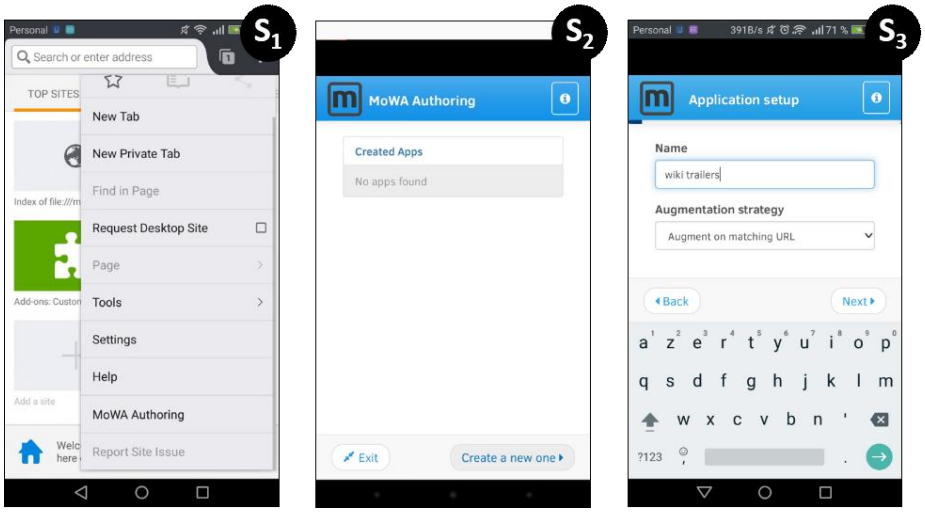

Figura 34. Configuración básica de una aplicación

2. Seleccionar la estrategia de aumentación. A fin de reducir los nodos navegables de la aplicación, este paso fue incorporado en el mismo formulario que establece el nombre de la aplicación (S3 en la Figura 34).

3. Seleccionar el/los sensor/es del contexto. En el paso «S4» de la Figura 35, el usuario debe elegir entre los sensores disponibles en su navegador, y en este caso elije solamente el sensor de orientación.

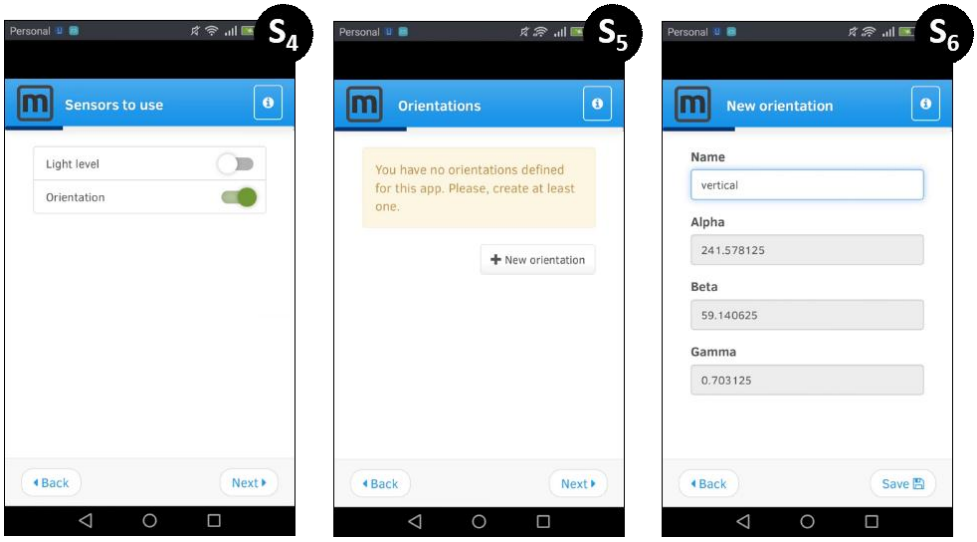

Figura 35. Seleccionando sensores y valores del contexto

4. Definir valores de contexto de interés para la aplicación. Como se ha seleccionado un sensor de orientación, el siguiente formulario que MoWA cargará será el del *builder* especializado en la definición de orientaciones. En el paso «S5» de la Figura 35 se puede observar que aún no se han definido orientaciones de interés, pero tras tocar el botón «new orientation», el usuario puede definir una nueva orientación de interés para su aplicación. En el paso

«S6», se muestra una captura en la que el usuario ya ha tipeado un nombre que representa la orientación deseada («vertical»), mientras que el valor de las coordenadas es automáticamente completado por el *builder*, quien está suscrito al sensor GPS y actualiza el valor de los puntos (alpha, beta, gamma) tras cada notificación. Otros mecanismos son posibles aquí, como una visualización tridimensional del dispositivo. Sin embargo, a fines demostrativos, esta es una alternativa práctica. Una vez que un valor de interés es creado, se regresa al formulario de gestión de orientaciones («S5»), que esta vez tendrá la lista actualizada de valores creados. Una vez que se definen todas las orientaciones deseadas (en este caso una sola), el usuario es automáticamente redirigido a un checkpoint, tal como figura en el paso «S7» de la Figura 36.

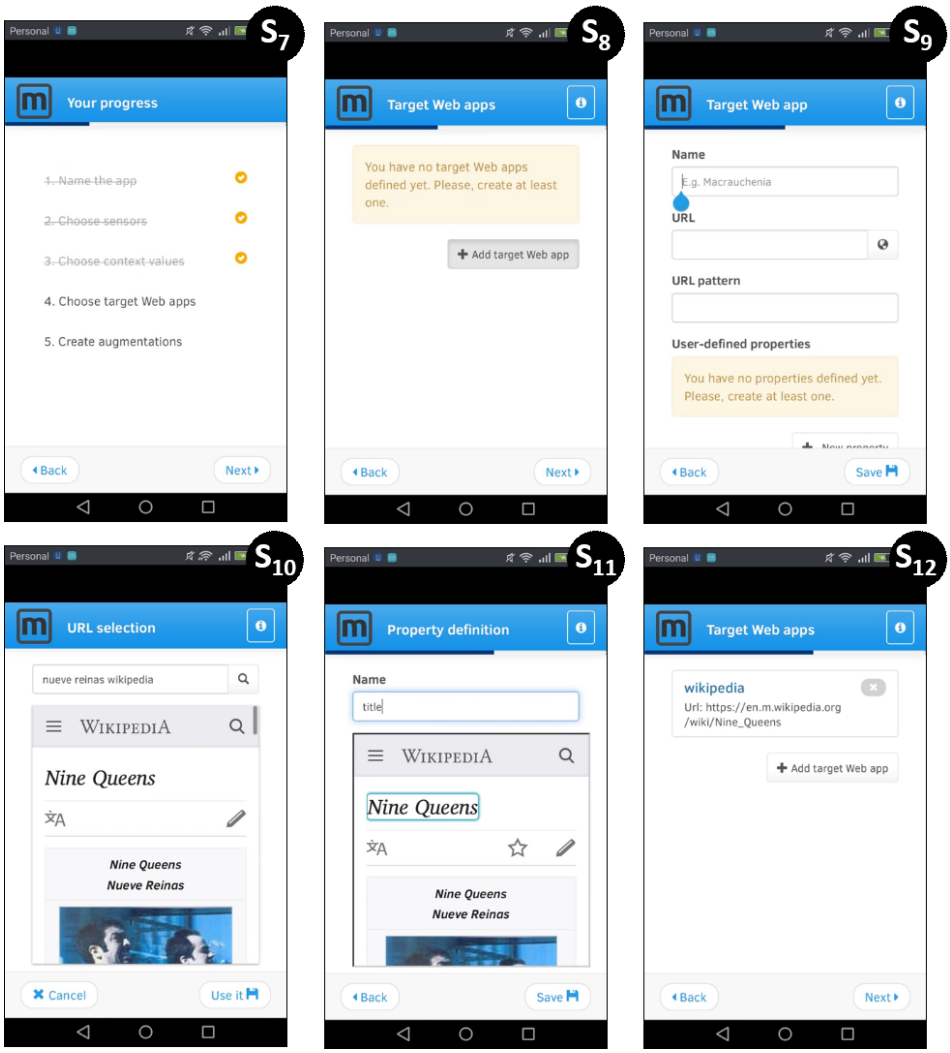

Figura 36. Definición de contraparte digital

5. Definir una contraparte digital. En el paso «S8» de la Figura 36, se muestra el formulario de gestión de contrapartes digitales o, para una mejor comprensión del usuario final, de «aplicaciones a aumentar». El siguiente paso es cargado

como resultado de la creación de una nueva contraparte digital (botón «add target web app»). En él, se puede observar que la contraparte digital requiere de un nombre, una URL (que se define en «S10») y de una propiedad extra: el título del artículo (que mantiene el mismo XPath en todos los artículos, pero no su contenido, y se define en «S11»). El valor de esta propiedad, recuperado tras evaluar el XPath del artículo correspondiente, será utilizada por un *augmenter* para recuperar tráileres acordes al valor del artículo. Una vez guardada, se regresa al formulario de gestión de «aplicaciones a aumentar», tal como se muestra en el paso «S12».

6. Definir una representación del tipo de contexto. En este paso, el usuario final elige si desea definir la representación o no. En el paso «S8», la herramienta ofrece la posibilidad de elegir entre las alternativas disponibles y el usuario selecciona el «Plano 2D personalizado», que debe ser configurado con una imagen de fondo y un nombre, tal como en el paso «S9» de la Figura 29. Se debe tener en cuenta que este paso y el siguiente se encuentran actualmente invertidos respecto a su orden según el diseño de la nueva herramienta (ver secciones 5.4 y 6.5).
7. Definir *augmenters*. En la Figura 37, se muestra cómo el *productor* puede construir una o varias aumentaciones («S13»). Cuando una nueva aumentación es creada, la URL –previamente definida para la contraparte digital– es cargada dentro de un *iframe* en el centro de la pantalla. Dentro de este *iframe* se mostrarán las previsualizaciones de los augmenters seleccionados por el usuario. En la parte inferior del formulario hay un botón que permite iniciar la selección e instanciación del builder de augmenter correspondiente: «Insert augmenter» en el paso «S15». Cuando el usuario toca este botón, aparece un popup con los augmenters disponibles para el/los tipo/s de contexto considerado/s por la aplicación. Cuando el usuario elije uno de ellos, un builder es instanciado en el contexto del *iframe* mencionado, tal como se muestra en el paso «S15» y de la misma manera que como fue soportado en la versión anterior de la herramienta.

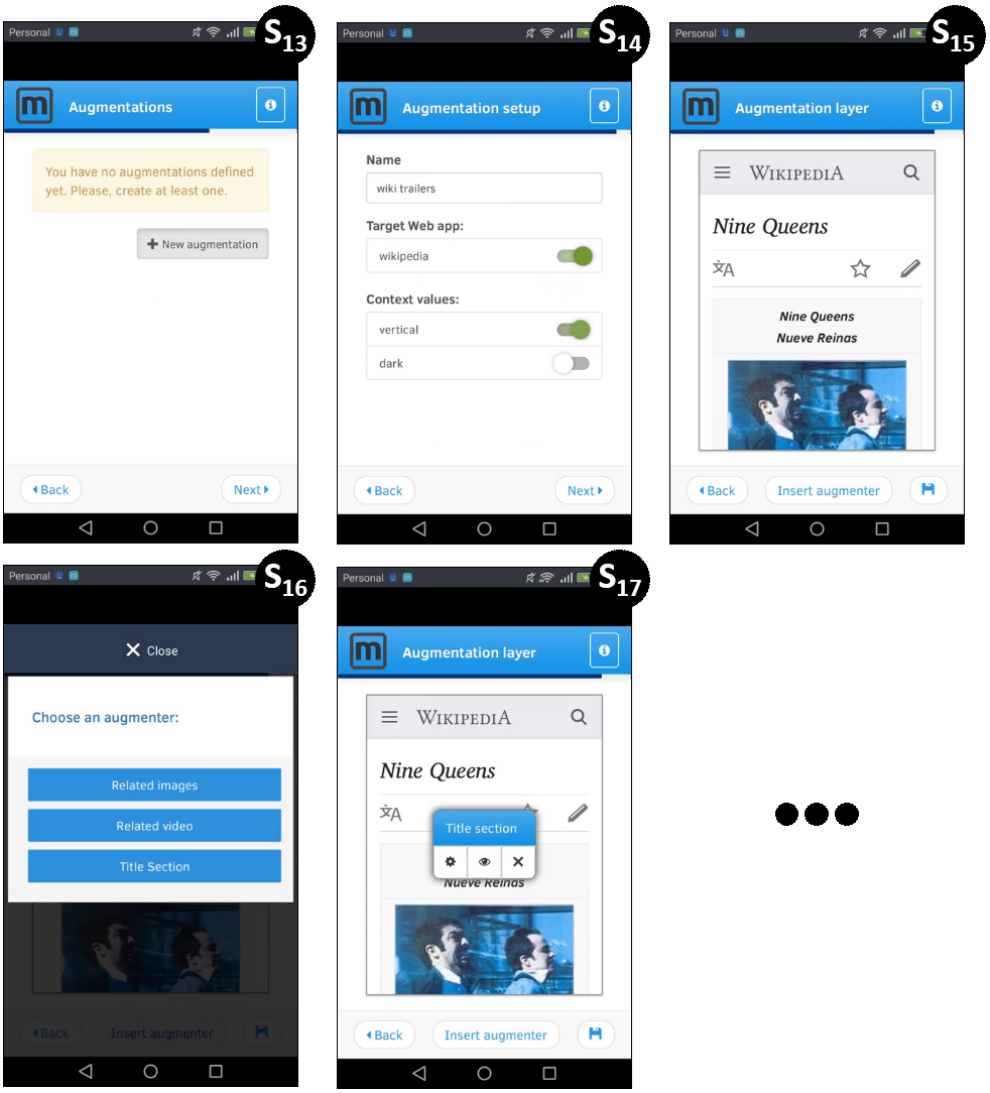

Figura 37. Iniciando la creación de una aumentación

La interfaz visual del builder del augmenter seleccionado debe ser reposicionada en relación con otro elemento DOM existente. Para ello, el usuario puede mantenerla presionada para habilitar el mecanismo de posicionamiento, que resulta en la ofuscación del builder y el resaltado temporal de los elementos del DOM sobre los cuales el usuario está deslizando su dedo («S18»). Cuando el usuario deja de presionar la pantalla, la interfaz visual del builder reaparece en relación con el último elemento tocado. En el paso «S19» se le consulta al usuario si desea posicionar el builder antes o después del tal elemento. El paso posterior presenta el builder reposicionado y, en el paso «S21», se le pide ejecutar el augmenter, aunque por defecto solo muestra un mensaje de advertencia, ya que aún no ha sido configurado.

En «S22», el usuario agrega un segundo augmenter, y posiciona su builder en relación con el primero creado. De esta manera, el primer augmenter mostrará un título y el segundo un tráiler. Pero además de la posición, el usuario debe configurar otros parámetros específicos de cada augmenter, como se muestra en «S23». En este formulario se está configurando el augmenter «RelatedYoutube Video», que para su

funcionamiento requiere de una cadena de texto con palabras claves. El formulario permite componer dichas palabras claves con una parte «estática» y otra «dinámica»; esto es, un texto que no variará en la ejecución de una consulta (la palabra "trailer") y otra cuyo valor dependerá de la propiedad del contexto digital actual (el título del artículo de Wikipedia). De esta manera, para el artículo actual, las palabras claves serán «trailer + Nueve Reinas».

Una vez configurado, los augmenters pueden ser correctamente previsualizados, como en «S24». Los augmenters son ejecutados sobre un contenedor que posee un control para permitirle al usuario regresar al modo de edición. En «S25» se puede observar que las aumentaciones desaparecen, y es debido al cambio de orientación del dispositivo, que no coincide con la orientación «vertical» marcada como «de interés». Una vez que todos los augmenters son configurados, cuando el usuario presiona el botón «Guardar», en la esquina inferior derecha del formulario, se accede nuevamente al formulario de gestión de aumentaciones, como el del paso «S13» de la Figura 37. Si el usuario presiona el botón «Siguiente», es presentado con el último checkpoint, en el cual se le indica que ha finalizado («S25»). Al presionar «Finish», la aplicación se muestra en la lista de aplicaciones creadas del usuario.

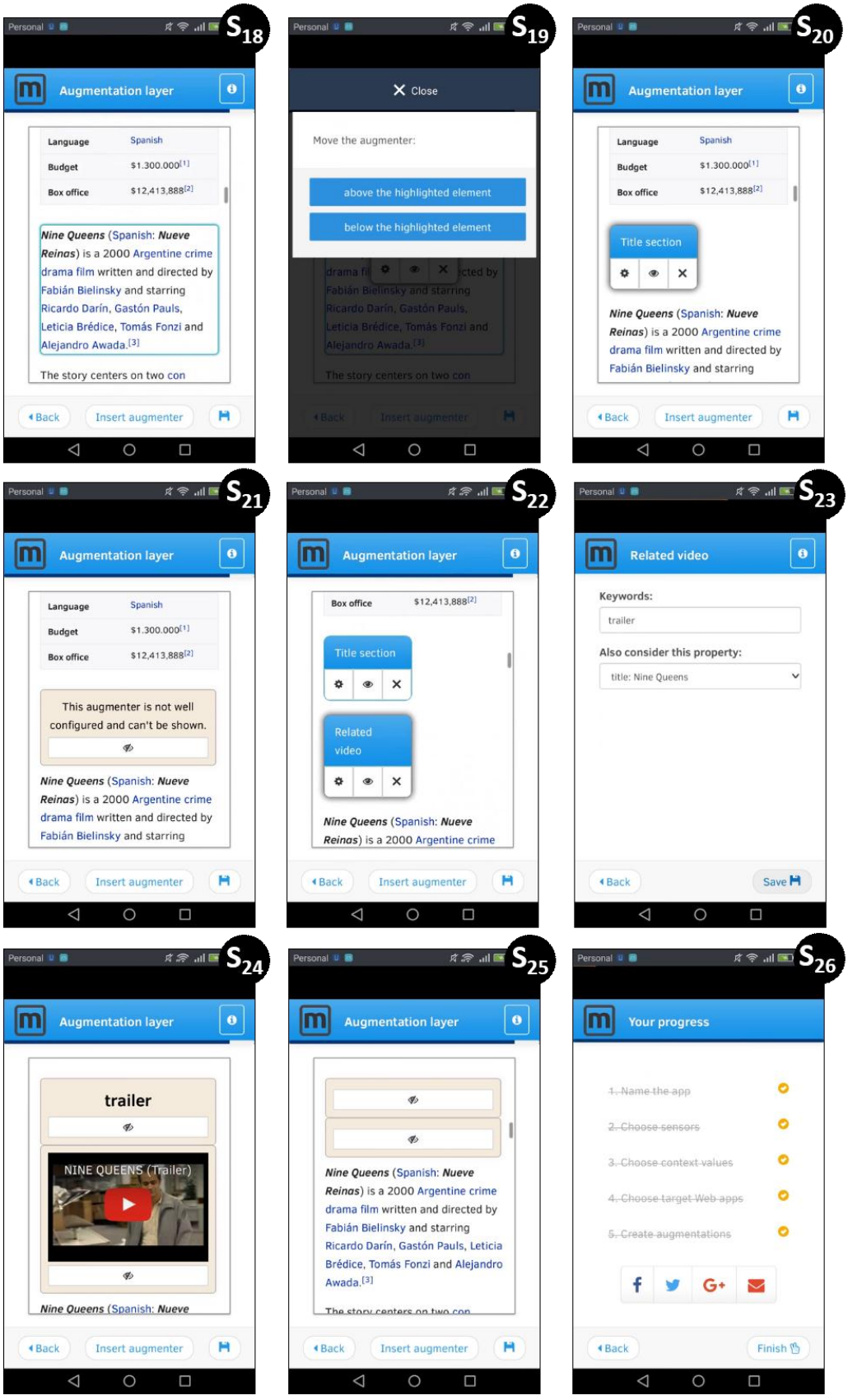

Figura 38. Componiendo la aumentación y finalizando el proceso de autoría

7. Evaluación

Con relación al escenario descriptivo, se llevó a cabo un experimento con 21 participantes para validar el enfoque de manera cuantitativa. Se pidió a la totalidad de los participantes utilizar *MoWA Authoring* para resolver un problema en un escenario concreto, construyendo una aplicación de Hipermedia Móvil [16] orientada al turismo. El objetivo de este experimento es demostrar la factibilidad de la herramienta y colectar métricas de uso de la misma. Este estudio es preliminar y no cubre todas las dimensiones de usabilidad o experiencia del usuario. Sin embargo, permite alcanzar un mejor entendimiento del potencial de la herramienta para usuarios finales. Para tal fin, se pretende verificar si los usuarios finales son capaces de comprender los conceptos necesarios y aplicarlos aumentando una aplicación Web existente.

7.1. Descripción demográfica

El experimento se llevó a cabo con 21 sujetos de experimentación con un amplio rango de características demográficas. Debido a la cantidad de participantes en la sesión, se eligió la encuesta como mecanismo de recolección de datos. La Figura 41 presenta un análisis de dichos datos. Del lado izquierdo de la misma se puede observar que el 71,43% de los miembros se identificaron con el género masculino, contra un 28,57% de género femenino. La edad del grupo resultó homogénea, con participantes de entre 22 a 39 años y un promedio de casi 27 años, tal como se muestra del lado derecho.

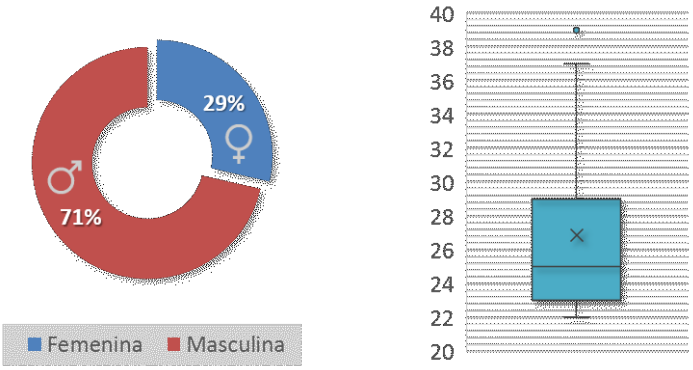

Figura 39. Participantes por género y edad

La Figura 40 refleja que los participantes tampoco compartían una misma nacionalidad, siendo la nacionalidad argentina la predominante con un total de 14 personas. También participaron 3 colombianos, 2 franceses, 2 venezolanos y una peruana.

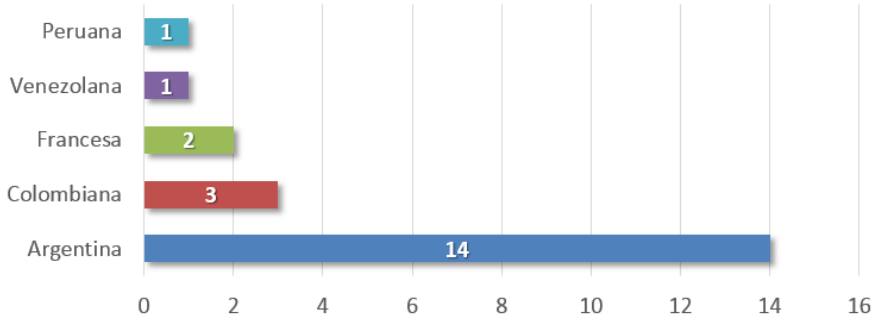

Figura 40. Participantes por nacionalidad

La mayor parte del grupo eran estudiantes de diferentes carreras de grado, representando el 52% de la muestra, precedida por casi un 29% que ya había terminado sus estudios universitarios o terciarios. Tal como se presenta en la Figura 41, el porcentaje restante se vio dividido en igual proporción entre aquellas personas que solo completaron el nivel de secundaria, que abandonaron sus estudios universitarios, que se encontraban cursando su carrera de postgrado y aquellos que ya la completaron.

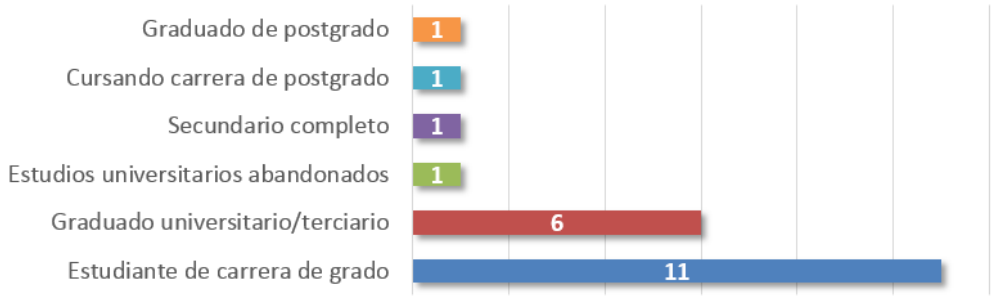

Figura 41.Participantes por nivel de estudio

La intención al momento de reclutar participantes fue contar con personas de diferentes contextos y con un amplio rango de intereses, en lugar de hacerlo con alumnos informáticos. La Figura 42 muestra que la mayoría de los participantes, conformada por once personas, provino de carreras vinculadas a las Ciencias Básicas; cuatro de Ciencias Sociales, dos de Arte, dos de Ciencias Naturales, uno de Ciencias Económicas y finalmente uno sin estudios especializados.

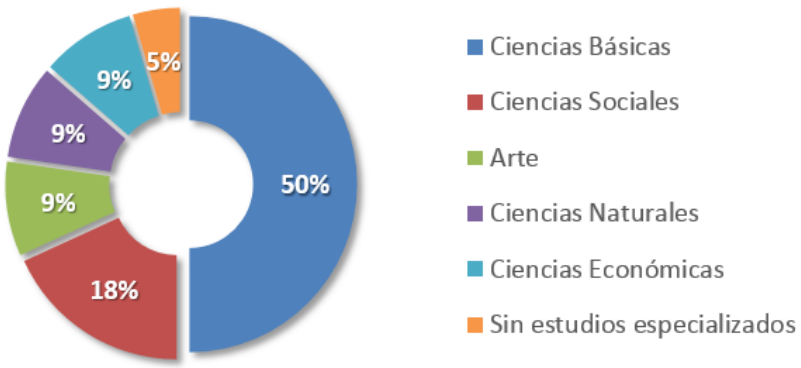

Figura 42. Participantes por área de estudio

También se hicieron preguntas para poder clasificar el nivel de experticia de los usuarios respecto al conocimiento y al uso de la tecnología involucrada en el experimento. La intención es poder determinar si esta variable influye en la performance de las tareas requeridas al participante. Tal como se detalla en [13] y se presenta en la Figura 43, estas preguntas determinaron que el 29% de los usuarios eran principiantes, el 28% eran regulares y el 43% restante eran expertos.

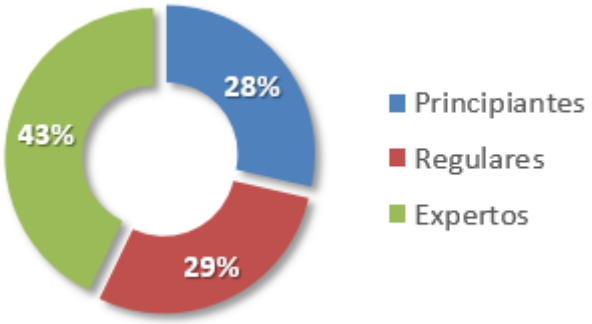

Figura 43. Participantes por nivel de experticia según tecnología requerida

7.2. Plataformas utilizadas

Uno de los requisitos para realizar el experimento fue utilizar un dispositivo basado en Android, capaz de ejecutar *Firefox for Android*. Los participantes se presentaron con sus propios teléfonos e instalaron el software requerido; tanto el navegador como la extensión. La Figura 44 muestra la distribución de dispositivos en los que la herramienta de autoría se ejecutó; ocho Samsung Galaxy S3 I9300, seis Motorola Moto G, dos LG Pro Light, dos Motorola Moto E, y sólo un Blu Studio 5.5, un Huawei G6 L33 y un Samsung Galaxy S2.

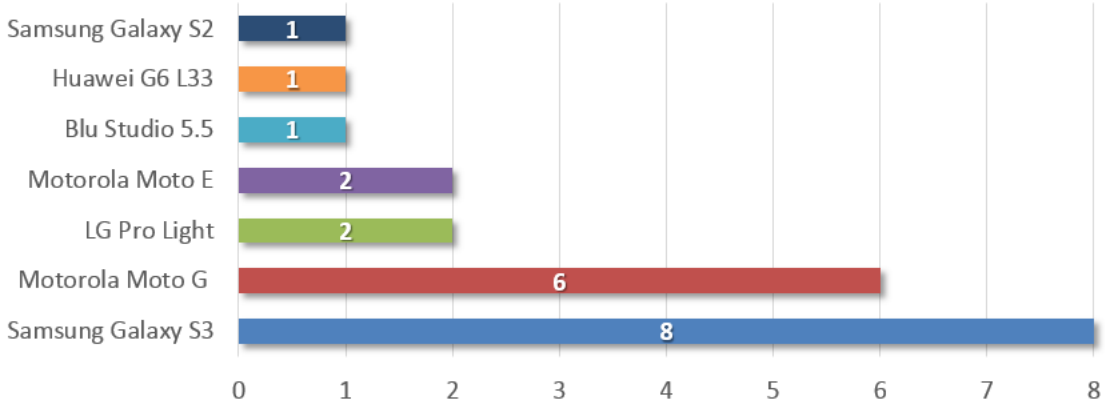

Figura 44. Modelos de smartphones utilizados

En lo que respecta a la plataforma Android, en la Figura 45 se puede apreciar con qué versiones se llevaron a cabo las sesiones experimentales. En primer lugar, se encuentra Android Jelly Bean con tres versiones de la API: uno dispositivo con la #16

(v.4.1.2), otro con la #17 (v.4.2.2) y once dispositivos con el #18 (v.4.3). Este último representa el 52% del total de los dispositivos. Las cantidades restantes corresponden en igual proporción, con solo 4 dispositivos, a Android KitKat (API nivel #19, v.4.4.4) y Lollipop (#21, v.5.0.2).

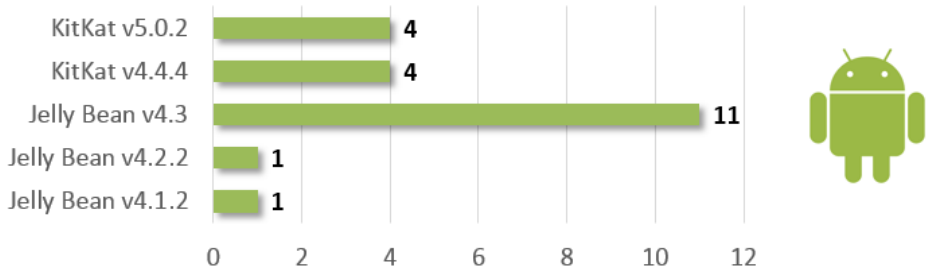

Figura 45. Versiones de Android utilizadas en el experimento

Respecto al navegador móvil, la Figura 46 muestra que la mayoría de las sesiones se realizaron con la versión 38, luego 5 con la versión 39, solo uno con la 37 y otro con la 41. El primero de ellos representaba la versión oficial estable en ese momento, ofrecida a través de Google Play. El segundo era la edición Beta. El tercero era un lanzamiento previamente instalado en el dispositivo y no actualizado. El último, la versión Nightly.

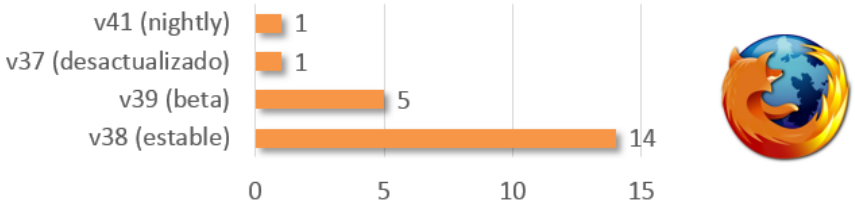

Figura 46. Versiones de Firefox for Android utilizadas en el experimento

7.3. Diseño del experimento

Tal como se explica en el primer artículo de MoWA para usuarios finales [13], el experimento fue diseñado para un escenario en el dominio del turismo. Se pidió a los participantes crear una aplicación que de asistencia al recorrido de determinadas piezas de un Museo. Este escenario fue el mismo que el utilizado para evaluar MoWA framework [24], con la finalidad de finalmente contrastar sus resultados.

El Museo de Ciencias Naturales de La Plata fue en las instalaciones del Laboratorio de Investigación y Formación en Informática Avanzada (LIFIA)[72], imprimiendo los mismos códigos QRPedia encontrados en el museo y distribuyéndolos por cuatro de sus espacios (tres oficinas y un corredor, como se muestra en la Figura 47). La razón

[72] http://www.lifia.info.unlp.edu.ar

para ubicarlos en diferentes ambientes fue, principalmente, que los usuarios tengan que desplazarse y realmente ser asistidos en la búsqueda de cada pieza, tal como lo tendrían que hacer en un Museo. De los cuatro espacios, la primera sala fue tomada como un espacio de entrenamiento donde los usuarios podían interactuar con nosotros y entre ellos para comprender el uso de la herramienta; los PDI definidos dentro de esta sala no se tienen en cuenta en el análisis de los resultados.

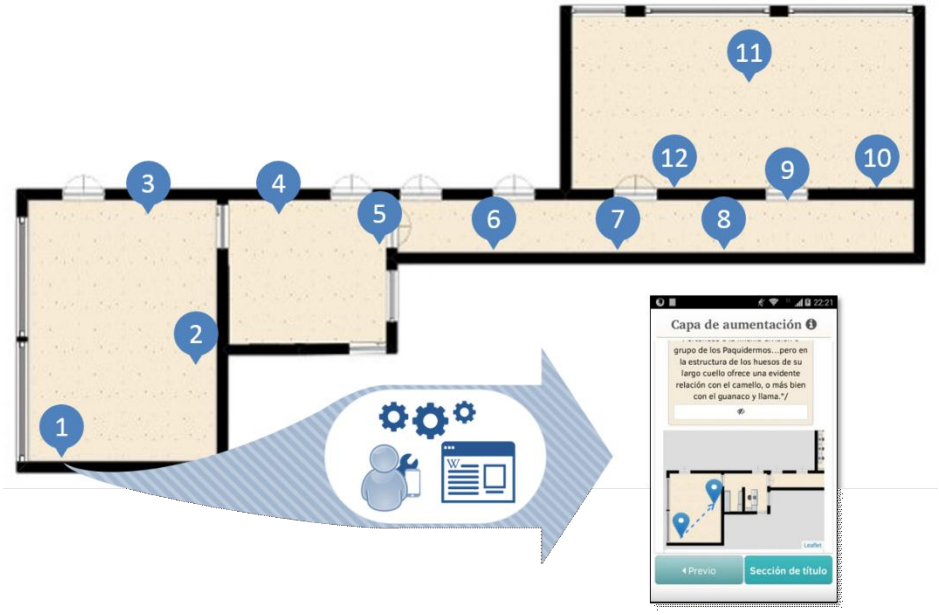

Figura 47. Distribución de códigos QR en las instalaciones del LIFIA

Antes de comenzar el experimento, explicamos a los participantes la motivación de nuestro enfoque y algunos conceptos tecnológicos específicos del dominio, relacionados con el tipo de aplicación requerida para realizar las tareas del experimento. Se explicó sobre aplicaciones de WA, Hipermedia Móvil y desarrollo *in-situ* en una charla de 30 minutos. Después de eso, se pidió a los participantes que completaran una encuesta para la recolección de datos demográficos.

Una vez que cada participante terminó la encuesta, se les presentó un escenario:

> *Estás a punto de visitar el Museo de Ciencias Naturales de la Universidad Nacional de La Plata para realizar un recorrido especial, conformado por un conjunto de piezas relacionadas a las descripciones de Darwin en su libro «Viaje del Beagle». Sabías de este tour porque lo viste en el sitio Web del Museo, así que sabés que el Museo provee información sobre cada pieza y que sugieren orden determinado para recorrerlas.*
>
> *Cuando llegás al Museo, observás que las piezas –no solo las que componen el tour que vas a recorrer– tienen un código QR que redirige a Wikipedia. Escaneás uno de ellos y te das cuenta de que la información que leíste en el sitio Web no es presentada por este medio, de modo que quienes no acceden al sitio Web del Museo se pueden estar perdiendo de la misma. Se te ocurre*

combinar la información de ambas fuentes y, de paso, agregar comportamiento que permita asistir al usuario en el recorrido de las piezas en el orden sugerido por el Museo. Para ello, necesitás crear una Aplicación Web móvil que soporte los siguientes requisitos:

R1. Aumentación de la página Web vinculada a cada pieza con información extraída del sitio oficial del Museo.

R2. Asistencia a los usuarios en el recorrido, mostrando las siguientes piezas a ser visitadas.

R3. Asistencia al usuario para llegar a la pieza correcta, en caso de que se encuentre en una pieza equivocada.

Cuando todos los participantes dijeron haber entendido el enunciado, los mismos recibieron instrucciones impresas donde se les explicaba que tenían que instalar la herramienta de creación MoWA y construir con ella una aplicación que resuelva el problema del escenario mencionado.

La recolección de datos fue mediante encuestas, *logging* y observación. Las experiencias individuales fueron registradas mediante *logging* y, al final de cada tarea, las instrucciones impresas pedían al usuario asignar un nivel de dificultad percibido. Al final del proceso, se pidió a los usuarios que completaran otra encuesta, esta vez sobre su percepción en el uso de la herramienta.

7.4. Resultados

7.4.1. Completitud y performance

En la Tabla 1 se presentan los resultados de las tareas realizadas por los participantes (PS) en relación con cada requisito (R1, R2 y R3).

Tabla 1. Porcentaje de tareas completadas por participante y sus tiempos

PS.	*R1*			*R2 & R3*				*SR*	*tiempo*
	a	*b*	*%*	*c*	*d*	*e*	*%*		*h:mm:ss*
1	1.00	0.56	0.78	0.56	1.00	1.00	0.85	**0.83**	0:49:20
2	1.00	1.00	1.00	1.00	1.00	1.00	1.00	**1.00**	0:57:53
3	0.61	0.61	0.61	0.89	1.00	1.00	0.96	**0.85**	0:57:52
4	1.00	1.00	1.00	1.00	1.00	0.75	0.92	**0.94**	0:58:20
5	0.11	0.00	0.06	0.00	0.11	0.00	0.04	**0.04**	0:16:04
6	1.00	0.00	0.50	0.00	1.00	1.00	0.67	**0.61**	0:46:29
7	0.50	0.50	0.50	1.00	1.00	1.00	1.00	**0.83**	1:05:57
8	0.89	0.89	0.89	0.89	0.89	0.38	0.72	**0.78**	1:02:22
9	1.00	1.00	1.00	1.00	1.00	0.63	0.88	**0.92**	1:12:09
10	0.56	0.22	0.39	0.22	1.00	0.38	0.53	**0.49**	0:54:07
11	1.00	1.00	1.00	1.00	1.00	1.00	1.00	**1.00**	1:06:11
12	0.50	0.50	0.50	1.00	1.00	1.00	1.00	**0.83**	0:49:54
13	1.00	1.00	1.00	1.00	1.00	1.00	1.00	**1.00**	0:59:13

14	1.00	0.78	0.89	0.78	1.00	1.00	0.93	**0.91**	0:49:02
15	1.00	1.00	1.00	1.00	1.00	0.50	0.83	**0.89**	0:35:55
16	1.00	1.00	1.00	1.00	1.00	0.63	0.88	**0.92**	0:54:41
17	0.81	0.83	0.82	0.89	0.89	0.63	0.80	**0.81**	0:33:22
18	1.00	1.00	1.00	1.00	1.00	1.00	1.00	**1.00**	0:57:10
19	1.00	1.00	1.00	1.00	1.00	1.00	1.00	**1.00**	0:23:06
20	1.00	0.50	0.75	1.00	1.00	1.00	1.00	**0.92**	0:22:22
21	1.00	1.00	1.00	1.00	1.00	1.00	1.00	**1.00**	0:25:57
%			0.79				0.86	**0.84**	**0:48:27**

Para determinar qué porcentaje de las tareas cumplió cada usuario se aplicó un criterio concreto que determina un puntaje por cada tarea. El criterio para el primer requisito (R1 en la Tabla 1) fue que las aplicaciones tengan definidos los 9 PDIs y el *PoIAugmenter* para las 9 páginas Web a aumentar. En el primer caso, se obtuvo un porcentaje analizando si cada PDIs tenía definido un nombre, una URL a aumentar y una imagen y descripción externas, ambas referenciadas desde la página del sitio Web oficial correspondiente. En el segundo caso, se verificó que el *PoIAugmenter* haya sido correctamente posicionado y configurado con los valores externos correctos. Cada propiedad fue evaluada por separado, y luego su promedio resultó el valor porcentual final. Siempre que una propiedad se encontró configurada parcialmente, fue marcada como incompleta (0), marcado con (1) sólo los valores correctos. Los resultados mostrados en la «columna B» representan el promedio de ambas propiedades.

El criterio para los requisitos R2 y R3 fue el mismo, ya que ambos se pueden lograr definiendo: 1) el *WalkingLinksAugmenter*, 2) los 9 PDIs y 3) las conexiones que establecen el orden entre ellos. Estas tres cosas se ven representadas por las columnas «C», «D» y «E» respectivamente. Los valores en la columna «C» representan un porcentaje de augmenters posicionados correctamente para los 9 PDIs; los valores en la «D», el porcentaje de PDIs definidos en relación con los 9 solicitados (sólo teniendo en cuenta el nombre y la URL a aumentar como atributos requeridos); los de la «E», el porcentaje de PDI conectados en relación con los 8 esperados. Las tasas de éxito (SR) de las tareas completas se obtuvieron como el promedio de los porcentajes de los tres requisitos.

Adicionalmente, la Figura 48 permite una mejor visualización de la distribución de los porcentajes de tareas completadas y de tiempos. Como se puede observar en el eje «x» de la misma, la mayor parte de los participantes cumplieron con más del 80% de los requisitos. Tanto de dicho subgrupo como del total de las experiencias, la mayor cantidad de ocurrencias en relación con el tiempo de realización (eje «y») se encuentran en el rango que va de 44:38 a 59:02 minutos.

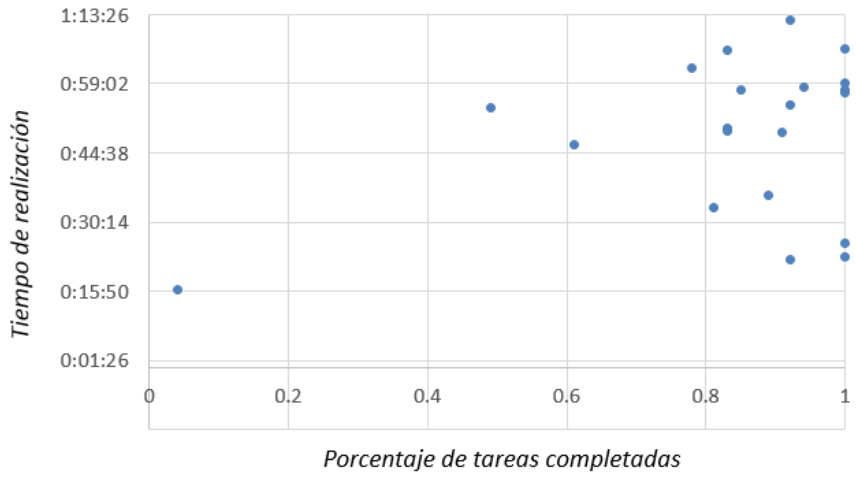

Figura 48. Distribución de los porcentajes de tareas completadas y sus tiempos

El resultado total de la Tabla 1 permite apreciar que los usuarios finales son capaces de completar, en promedio, el 84% de los requerimientos del experimento. La desviación estándar para estos valores es de 22 puntos (en este caso, un porcentaje). Entre las 21 observaciones analizadas, 3 de ellas obtuvieron un porcentaje por debajo del 75% y 18 por encima de él.

Con respecto a los tiempos, todo el proceso tomó a los participantes un promedio de 48:27 minutos, que varían desde 22 a 72 minutos. Las interacciones del usuario comenzaron a ser registradas tras disparar el botón «Crear aplicación», y se detuvo al exportar la aplicación creada. Cada interacción del usuario fue almacenada en un archivo local, que finalmente se exportó junto con la aplicación a un servidor. Estos registros permitieron observar el tiempo consumido real por cada usuario para todo el proceso, no solo para las tareas relativas a los requisitos.

El nivel de experticia no influyó de manera determinante en los resultados de los participantes. Se detectó una diferencia de sólo 2,4% entre los resultados de los participantes con un bajo nivel de experticia y los usuarios regulares, y solo un 10,42% entre los inexpertos frente a los expertos. Es preciso resaltar que las diferencias en su conocimiento eran amplias. Por ejemplo, el 83,3% de los participantes principiantes no habían utilizado Firefox for Android previamente, y absolutamente ninguno en esta categoría utilizó extensiones para este navegador móvil. Un 66.6% dijo incluso nunca haber usado extensiones de navegador en general, aplicaciones de Aumentación Web o escáneres de códigos QR. El 83,3% nunca utilizó una herramienta de autoría, ni tampoco manejó marcadores en un mapa. Muchos de estos fueron desafíos que el usuario tuvo que enfrentar por primera vez para llevar a cabo el experimento y, de todos modos, obtuvieron resultados prometedores incluso en relación con usuarios experimentados.

El nivel de experticia no influyó de manera determinante en los resultados de los participantes. Se detectó una diferencia de sólo 2,4% entre los resultados de los participantes con un bajo nivel de experticia y los usuarios regulares, y solo un 10,42%

entre los inexpertos frente a los expertos. Es preciso resaltar que las diferencias en su conocimiento eran amplias. Por ejemplo, el 83,3% de los participantes principiantes no habían utilizado Firefox for Android previamente, y absolutamente ninguno en esta categoría utilizó extensiones para este navegador móvil. Un 66.6% dijo incluso nunca haber usado extensiones de navegador en general, aplicaciones de Aumentación Web o escáneres de códigos QR. El 83,3% nunca utilizó una herramienta de autoría, ni tampoco manejó marcadores en un mapa. Muchos de estos fueron desafíos que el usuario tuvo que enfrentar por primera vez para llevar a cabo el experimento y, de todos modos, obtuvieron resultados prometedores incluso en relación con usuarios experimentados.

7.4.2. Nivel de dificultad percibido

También se pidió a los participantes que calificaran la dificultad de los pasos del proceso, cuyos resultados se muestran en la Tabla 2. Para cada tarea en la primera columna de la tabla, los usuarios tuvieron que elegir entre cinco notas que coincidan con diferentes niveles de dificultad: 1) muy Fácil, 2) fácil, 3) regular, 4) difícil y 5) muy difícil. Se obtuvieron promedios para cada caso, a partir de las respuestas de los participantes. Sin embargo, el único paso que acumuló 21 calificaciones fue el primero (a); hubo un participante que se negó a calificar las tareas restantes porque decidió abandonar el experimento y no terminó el proceso. Este participante marcó como «muy fácil» a la única tarea que calificó. Ninguna tarea fue calificada como «muy difícil» por ningún participante.

Tabla 2. Dificultad percibida por los participantes

	Tarea	***Promedio por usuarios***	***Desviación estándar***	***Aprox***	***Categoría coincidente***
a	Instalar la extensión	1.40	0.60	1	Very easy
b	Configurar la app (nombre, estrategia de aumentación, etc.)	1.53	0.61	2	Easy
c	Configurar el tipo de representación	1.84	0.76	2	Easy
d	Definir valores del context de interés y sus contrapartes digitales	2.58	0.84	3	Regular
e	Configurar augmenters (promedio del promedio de notas por cada augmenter)	2.66	0.86	3	Regular
f	Exportar la aplicación	1.39	0.50	1	Very easy

7.4.3. Feedback sobre problemas y sugerencias

En las encuestas también se pidió a los participantes que informaran sobre los problemas que habían experimentado, que hicieran sugerencias sobre la herramienta y comenten que otros posibles usos tendría la herramienta.

En cuanto a los problemas experimentados, tres participantes consideraron que la

aplicación era lenta para ellos. El paso que resaltaron de larga duración fue el escaneo de los códigos QR. Como el *QrBasedLocationSensor* utiliza una biblioteca embebida al navegador para decodificarlos del lado del cliente, la capacidad de procesamiento del dispositivo jugó un papel importante.

Cinco participantes también dijeron que las instrucciones impresas no eran suficientemente claras para ellos, y que les tomó algún tiempo extra comprender qué hacer, incluso con las explicaciones de la charla inicial y la etapa de entrenamiento. Dos participantes sufrieron de problemas técnicos. Uno de ellos tuvo que reiniciar el navegador 3 veces porque la herramienta se congeló. Este fue el caso del participante que utilizó la versión 37 de Firefox, y esta podría ser la razón de tal inconveniente. Otro de ellos tuvo problemas al para definir el primer PDI; tras cargar por primera vez el formulario, el *listener* a cargo de crear un nuevo marcador no respondía, pero el problema se vio resuelto al recargar el formulario.

Respecto a las mejoras, un participante expresó que le gustaría haber tenido una interfaz con íconos más grandes. Otro, apreciaría tener un indicador de progreso en la construcción de la aplicación, aunque luego argumentó que quizá sería confuso, puesto que el porcentaje depende de variables que no son determinadas de antemano, como la cantidad de PDI o de augmenters que va a contemplar la aplicación. Dos participantes intentaron, sin éxito, copiar y pegar texto entre algunas propiedades y lo señalaron como una mejora para la herramienta. Sabíamos de esta limitación, pero no pudimos implementar la característica para esas versiones concretas de la API del navegador mientras el mismo se mantenía en modo de pantalla completa. Otro participante sugirió la función de copiar y pegar, pero esta vez no para el texto sino para importar la lista de propiedades definidas para otros PDIs. Dos participantes consideraron que la inserción de augmenters en el documento de la página a aumentar debería materializarse mediante el gesto «arrastrar y soltar». Versiones iniciales de la extensión lo hacían de esta manera, pero finalmente decidimos buscar una alternativa debido al costo en el rendimiento sobre algunos dispositivos. Finalmente, un participante expresó que le hubiera gustado contar con alguna manera de hacer zoom sobre el contenido de los *iframes* utilizados para seleccionar contenido desde una aplicación Web externa.

En cuanto a otros posibles usos para la herramienta, todos los participantes limitaron sus escenarios a la asistencia de recorridos. Cuatro de ellos dijeron que sería bueno para el turismo en general, contemplando recorridos en otros museos o ciudades. Un participante pensó que podría ser útil para presentar información sobre el uso de las instalaciones de una Universidad, y otros tres para la programación de actividades en la misma. Un último participante consideró la creación de algún juego, como la búsqueda del tesoro.

7.5. Factores que amenazan la validez del diseño

Como la omisión de algunos factores en la interpretación de los resultados puede conducir a conclusiones erróneas, se presentan a continuación algunos puntos relacionados a la validez del diseño del experimento llevado a cabo.

En primer lugar, se listan aquellos factores que podrían representar una diferencia entre los participantes del caso experimental presentado:

- *Experiencia de usuario*. Si bien ningún participante conocía o utilizó alguna vez la herramienta *MoWA Authoring*, hemos realizado el experimento con un grupo heterogéneo de participantes de diferentes capacidades respecto al uso de dispositivos móviles, navegadores móviles y extensiones de navegación. Los clasificamos haciendo algunas preguntas sobre su conocimiento tecnológico, que los categoriza en usuarios principiantes, regulares y expertos, tal como se presenta en la Sub-Sección 6.1.
- *Aprendizaje*. Algunos usuarios tienden a aprender las tareas requeridas más rápido que otros, y algunos de ellos requieren más práctica o repeticiones para lograr los mismos resultados. Se presentaron algunas tareas repetitivas, como definir un marcador para los PDIs, llenar los datos requeridos y extraer datos de sitios externos, colocando y configurando augmenters.
- *Fase de entrenamiento*. Con relación al punto anterior, a los participantes se les permitió interactuar durante una primera fase del experimento para despejar dudas respecto al uso de la herramienta. Esta interacción fue posible tanto con los experimentadores como con el resto de los participantes por medio de una conversación cara a cara, pero no estaba permitido realizar las tareas de otros participantes en su lugar. Sin embargo, no se observó el mismo nivel de desenvolvimiento por parte de todos los participantes y es difícil saber qué hubiera ocurrido sin llevar a cabo esta etapa inicial.
- *Maduración*. Como los usuarios tuvieron que crear un total de 12 PDIs (9 para el experimento más 3 extras para su entrenamiento) y su creación conllevó un tiempo prolongado (en promedio, alrededor de 48 minutos), pudieron experimentar diferentes cambios psicológicos, como fatiga, causando un impacto en el nivel de éxito de las tareas cumplidas. También pueden sentirse frustrados y dejar de fumar. De hecho, hubo un caso de un participante que no terminó el proceso.

Además, otros factores pueden alterar la representatividad de los resultados, haciendo las conclusiones generalizables sólo bajo ciertas condiciones. En primer lugar, porque el experimento fue controlado, en un ambiente en el cual algunas condiciones pueden diferir del mundo real, y realizado con una muestra pequeña de participantes.

También fue diseñado para resolver un problema específico de un solo dominio, aunque la herramienta en si misma fue concebida para cubrir diferentes dominios. A continuación, presentamos algunos puntos requeridos para la replicación del experimento:

- *Selección de participantes*. El objetivo era obtener una muestra representativa de usuarios finales para participar en el experimento, para lo cual se invitó a personas de diferentes contextos. Ellos decidieron participar voluntariamente en el experimento, y ningún participante fue descartado, ni siquiera si no era habitué del uso de dispositivos móviles o smartphones. Esto aseguró la heterogeneidad del grupo, aunque hubo factores con valores predominantes entre los participantes: 71,43% de los participantes identificaron con el género masculino; 66,67% de los ciudadanos eran de nacionalidad argentina; 90,48% tenían, al menos, un nivel de estudios de grado o se encontraban cursándolo; 52,38% de los participantes estaba vinculado a las ciencias exactas; 42,86% tienen una edad en el rango de 20 a 40 años.
- *Selección del contexto*. El experimento se realizó en las instalaciones del LIFIA, abarcando cuatro ambientes independientes con el fin de asegurar el desplazamiento del usuario en búsqueda de los diferentes PDIs. Sin embargo, las dimensiones del museo, la disposición y representación de las piezas de la colección fueron diferentes. Además, nuestras instalaciones proporcionaron una conexión inalámbrica estable a Internet, que puede no estar disponible en cualquier espacio público.
- *Selección del dominio y la complejidad de las tareas*. Para poder comparar los resultados del presente experimento en relación con los resultados en [Challiol et al., 2013] (por desarrolladores), decidimos llevar a cabo el experimento bajo el mismo escenario, teniendo en cuenta un único nivel de complejidad. Esto, además, pudo resultar complejo para algunos usuarios, ya que no solo tuvieron que comprender el problema a resolver, sino también el marco de una aplicación de Hipermedia Móvil.

7.6. Discusión

Tras la evaluación se demostró una alta probabilidad de que un usuario final, sin conocimientos de programación, sea capaz de mejorar una experiencia Web móvil reutilizando el contenido existente en la Web con características móviles. Sin embargo, aún quedan muchas preguntas que responder, y se despliegan de los siguientes hechos.

Uno de los participantes tuvo un comportamiento inesperado: aunque iba en la dirección correcta, abandonó el experimento argumentando que no sabía cómo hacerlo y no quería continuar. Configuró la aplicación, el mapa, creo y conectó algunos PDIs, y hasta configuró correctamente los augmenters de dos PDIs. Sin embargo, completó

sólo un 4% del proceso requerido.

También podríamos observar que, a diferencia de otros participantes, él no intentó resolver el problema pidiéndonos o interactuando con otras personas. Formaba parte de la categoría de usuarios principiantes y, sin embargo, los resultados de esta categoría resultaron prometedores; lograron, en promedio, un 78,5% de los requerimientos, aunque con una desviación estándar de 37,3 puntos. En este sentido, los usuarios regulares y los expertos obtuvieron mejores cifras: un 80,9% con una desviación estándar de 17,6 y 88,9% con una desviación de 11,9%, respectivamente.

Finalmente, se observaron algunas complicaciones respecto al uso de algunos elementos de la UI. En primer lugar, se utilizó un diseño poco apropiado para presentar las propiedades de un PDI, puesto que muchos participantes tuvieron dificultades para interpretar esta distribución como parte de un formulario en una pantalla de pequeña dimensión. Un segundo problema estuvo vinculado a las interacciones esperadas por los usuarios finales. En concreto, los usuarios identificaron como problemática la interacción para insertar un nuevo aumentador en la página Web a aumentar. Como el contexto de instanciación del *carrousel* con los aumentadores difería del de la página Web a aumentar, y en parte también debido a las limitaciones en la capacidad de procesamiento de un dispositivo móvil, era difícil simular un el efecto «drag and drop» entre ambos contextos sin provocar una renderización lenta. En su lugar, se implementó la inserción mediante un toque («tap») en la lista de aumentadores, que instancia el *builder* del aumentador en el contexto de la página Web, desde donde el usuario puede arrastrarlo y soltarlo para posicionarlo. También hubo un problema con la interacción para la captura y decodificación de QRs; se optó por dispararla ante un simple toque sobre el área de la captura de la cámara (como en muchas cámaras de teléfonos móviles). Pero la falta de un botón confundió a un participante, quien intentó sin éxito recargar el formulario algunas veces antes de intentar de tocar la pantalla. Este usuario cargó 36 veces dicho formulario, siendo 12 veces el número óptimo.

8. Conclusión, trabajo en curso y futuro

Una única aplicación Web no siempre se ajusta a las necesidades de un usuario en particular; muchas veces carece de alguna información o comportamiento requerido. La *Aumentación Web* permite a los usuarios manipular la Web según sus propios requisitos. A su vez, los dispositivos móviles traen aparejado un nuevo reto: crear aplicaciones adaptables que saquen ventaja de la información que el dispositivo puede sensar de su contexto. Muchos sitios cuentan con tales características e incluso algunos proporcionan una contraparte móvil nativa. Pero la realidad es que también es posible encontrar sitios sin soporte a dispositivos móviles. La Aumentación Web Móvil se presenta, ante este escenario global, como una forma de permitirle al usuario una mejor experiencia con sitios existentes que no contemplan características móviles. Por su parte, el *Desarrollo por Usuarios Finales* permite que el usuario no dependa de un tercero, de una persona especializada en desarrollo Web, para la creación de una solución que satisfaga sus necesidades. El Capítulo 2 de esta obra presentó los campos y conceptos fundamentales detrás de estas necesidades.

Actualmente existen enfoques que permiten aumentar aplicaciones Web pero que requieren conocimiento en *scripting*. Por otra parte, existen aplicaciones que se adaptan al contexto, y también un gran número de herramientas de autoría que permiten la creación de este tipo de aplicaciones, pero ninguna de ellas genera *aplicaciones Web puras*, que puedan ser ejecutadas en navegadores móviles convencionales sin depender de componentes nativos. En el Capítulo 3 se presentó una recopilación de trabajos que, si bien están relacionados a este enfoque, no coinciden en su aporte.

Habiendo presentado el marco teórico en el cual se encuadra el aporte de esta obra, y un conjunto de trabajos cuyos aportes están relacionados a ella, el Capítulo 4 hizo mención explícita del problema y presentó las preguntas de investigación que se pretenden responder. Además, presentó seis escenarios basados en sitios reales cuya experiencia podría ser mejorada mediante adaptación. Y, como se trata de sitios de terceros, la manera de cumplir con cada uno de los requerimientos planteados es mediante Aumentación Web móvil del lado del cliente. Con la finalidad de validar la factibilidad técnica de las mismas, y antes de avanzar en la construcción de una solución concreta, se desarrollaron scripts simples de aumentación para solucionar parcialmente los problemas planteados en los escenarios. Los mismos fueron ejecutados en el contexto de la consola del navegador (distinta de la consola Web convencional), cuyos privilegios coinciden a los de la solución propuesta en esta obra.

Ante la necesidad de adaptación de aplicaciones Web y el uso popular de tecnologías móviles nace MoWA; un enfoque de *Desarrollo por Usuarios Finales* que permite crear aplicaciones de *Aumentación Web móvil* mediante un proceso basado en formularios, widgets visuales y *live programming*. El enfoque fue desarrollado en el

Capítulo 5, mientras que la herramienta de soporte fue presentada en el Capítulo 6.

MoWA comprende elementos de los diferentes campos presentados en el Capítulo 2. Por un lado, utiliza una técnica de Aumentación Web del lado del cliente mediante la extensión del navegador Web móvil. De esta manera, las características contempladas por las aumentaciones cuentan con sus mismos privilegios, haciendo posible la inyección de nuevo contenido y comportamiento en correspondencia a la información del contexto. Estas aumentaciones pueden ser creadas no solo por *desarrolladores*, sino también por usuarios finales sin conocimiento en desarrollo Web, mediante el uso de una herramienta de autoría que combina un proceso dirigido por formularios con widgets que permiten configurar y previsualizar los *augmenters* que componen la capa de aumentación de una o varias páginas Web.

El beneficio de que un usuario pueda jugar el rol de *productor* no se reduce solo a la satisfacción de sus propias necesidades; dichas aplicaciones pueden ser compartidas o publicadas en respuesta a las solicitudes de otros usuarios dentro de una plataforma de Crowdsourcing, de la que tanto desarrolladores como usuarios finales pueden ser partícipes. De esta manera, las posibilidades de obtención de una respuesta son mucho más amplias, pues ya no se limitan a las de aquellos Crowdworkers que poseen conocimiento en tecnologías de más bajo nivel o en procesos de desarrollo formal.

En sus inicios, *MoWA* contemplaba solo a desarrolladores que extendían el framework mediante scripting y compartían sus aplicaciones resultantes con los usuarios finales a través de una plataforma de Crowdsourcing. Los usuarios finales, por su parte, estaban limitados a utilizar las soluciones existentes para un dominio y escenario específico, o a solicitar la implementación de nuevas. Esta situación, junto a los comentarios de los desarrolladores tras un experimento con el framework, impulsó la creación de *MoWA Authoring*: una herramienta pensada para que los usuarios finales sean quienes puedan satisfacer sus propias necesidades de adaptación y abarcando múltiples dominios y escenarios.

Adicionalmente, esta herramienta permite el desarrollo in-situ y oportunista, en el que el usuario construye su solución en el que será posteriormente el contexto de ejecución de la aplicación resultante, ya sea virtual (una página web) o física (una posición geográfica). De esta manera, ciertas características no deberían representar un problema al momento de ejecución. Por ejemplo, que la adaptación sea apropiada para dispositivos de otras dimensiones, con otros modos de interacción, etc. El desarrollo también permitió omitir un paso dentro del proceso de desarrollo formal de software, donde la comunicación entre las personas suele tener algunos problemas de comprensión: la elicitación de requerimientos. En este enfoque, este paso es realizado por la misma persona que estará a cargo de la creación de la aplicación.

El proceso de autoría fue soportado a través de formularios, asistiendo al usuario a través de una serie de pasos que les permiten elegir la mejor configuración para su escenario. Cada componente en este proceso es configurado por *builders*, que cualquier

desarrollador puede extender y adaptar a su necesidad. Estos *builders* son creados por desarrolladores que toman las decisiones técnicas que consideran más adecuadas para resolver problemas específicos de un dominio. Luego, es el usuario final quien, utilizando estos *builders*, elije la mejor configuración para un escenario concreto.

En este trabajo no solo se satisface una necesidad del usuario final llevando el Desarrollo por Usuarios Finales al mundo de la Aumentación Web Móvil, sino que también se la presenta como una especialización de la Aumentación Web tradicional, se comprende y se expanden los límites de la tecnología.

La viabilidad del enfoque fue demostrada mediante la implementación de una herramienta de soporte, presentada en el Capítulo 6. La misma, fue objeto de escenario descriptivo y utilizada por un grupo de 21 usuarios finales de un variado espectro de características demográficas, que construyó una aumentación para la asistencia turística sobre la base de un conjunto limitado de artículos de Wikipedia. Los escenarios descriptivos se presentaron en el Capítulo 6, mientras que el experimento fue presentado en el Capítulo 7. Los resultados mostraron que el enfoque no sólo era factible sino prometedor ya que los participantes pudieron completar, en promedio, el 84% de los requerimientos del experimento en un tiempo 4 veces más rápido que los desarrolladores mediante scripting. Además, el feedback obtenido permitió mejorar la herramienta y obtener una segunda versión, la cual fue presentada en una sesión de demostraciones en una conferencia internacional: ICWE 2017.

El peor escenario de uso que se contempló para *MoWA Authoring* es aquel en el que un usuario final no encuentra un *builder* o sensor específico, o no entiende cómo combinar componentes para resolver un problema particular. La solución a lo primero es la demanda mediante la plataforma de Crowdsourcing, mientras que para enfrentar lo segundo se están considerando los beneficios de permitir a los desarrolladores crear herramientas de autorización específicas de dominio, de modo que los *productor*es puedan utilizarlas para crear sus soluciones con un bajo nivel de conocimiento técnico, sin la necesidad de tomar decisiones sobre cuán apropiado es utilizar uno u otro sensor, *augmenter* u otro componente del framework.

Se comenzó también con el diseño de un DSL para la especificación de aplicaciones de *Aumentación Web móvil*, dirigido a usuarios con diferentes niveles de conocimiento y de requerimientos. El objetivo es construir un espacio común entre desarrolladores y usuarios finales, tratando de generar nuevas formas de desarrollo colaborativo.

Además, se están diseñando nuevos procesos soportando el desarrollo de aplicaciones MoWA desde un ambiente colaborativo, en el que algunas tareas intermedias puedan delegarse a la multitud, integrada por usuarios de diferentes niveles de experiencia. Por ejemplo, un usuario final avanzado puede iniciar un proyecto en la comunidad, definiendo la estructura básica de la aplicación. A continuación, los

desarrolladores pueden contribuir mejorando el código con sus conocimientos técnicos de bajo nivel y algunos usuarios pueden recopilar información sobre el mundo real, como imágenes, ubicaciones, descripción de lugares in situ. Otro participante puede contribuir en el proyecto recolectando nuevas fuentes de datos desde la Web para ser utilizados desde las aumentaciones. Los usuarios deben ser capaces de sugerir cambios y también para bifurcar un proyecto completo, siempre y cuando la visibilidad de este sea pública, o privada, ya sea porque el proyecto es propio o porque ha sido compartido con ellos.

Una vez que la plataforma Crowdsourcing se encuentre en marcha públicamente, se dispondrá de un mayor volumen de datos para analizar, lo cual nos permitirá comprender mejor cómo los usuarios de descubren y utilizan los augmenters disponibles. Esto también nos llevará a analizar la evolución de los proyectos y, en particular, las necesidades, comportamientos y formas de aprendizaje del usuario.

Finalmente, aunque solo respecta a cambios en la implementación de la herramienta de soporte al enfoque, cabe mencionar que la misma deberá migrarse a una nueva tecnología llamada «webextensions», ya que a partir de Firefox 57 este será el único tipo de extensión soportada por el navegador[73].

[73] https://blog.mozilla.org/addons/2017/02/16/the-road-to-firefox-57-compatibility-milestones/

Referencias

[1] G.D. Abowd, C.G. Atkeson, J. Hong, S. Long, R. Kooper, M. Pinkerton, Cyberguide: A mobile context-aware tour guide, Wirel. Networks. 3 (1997) 421–433.

[2] G.D. Abowd, A.K. Dey, P.J. Brown, N. Davies, M. Smith, P. Steggles, Towards a better understanding of context and context-awareness, Proc. First Int. Symp. Handheld Ubiquitous Comput. (HUC '99). (1999) 304–307.

[3] I. Aldalur, M. Winckler, O. Díaz, P. Palanque, Web Augmentation as a Promising Technology for End User Development, in: New Perspect. End-User Dev., Springer International Publishing, Cham, 2017: pp. 433–459.

[4] A.J. Andrew J. Ko, B.A. Myers, H.H. Aung, Six Learning Barriers in End-User Programming Systems, in: 2004 IEEE Symp. Vis. Lang. - Hum. Centric Comput., IEEE, 2004: pp. 199–206.

[5] C. Arellano, O. Díaz, J. Iturrioz, Crowdsourced web augmentation: A security model, in: Lect. Notes Comput. Sci., 2010: pp. 294–307.

[6] C. Asakawa, H. Takagi, Transcoding, in: Springer London, 2008: pp. 231–260.

[7] F.T. Balagtas-Fernandez, H. Hussmann, Model-Driven Development of Mobile Applications, MoDELS. 1 (2014) 509–512.

[8] M. Baldauf, S. Dustdar, F. Rosenberg, A survey on context-aware systems, Int. J. Ad Hoc Ubiquitous Comput. 2 (2007) 263.

[9] L. Barkuus, A. Dey, Location-Based Services for Mobile Telephony : a Study of Users' Privacy Concerns, Proc. INTERACT 2003, 9TH IFIP TC13 Int. Conf. Human-Computer Interact. (2003) 1–5.

[10] B.W. Boehm, Verifying and Validating Software Requirements and Design Specifications, IEEE Softw. 1 (1984) 75–88.

[11] G. Bosetti, S. Firmenich, A. Fernandez, M. Winckler, G. Rossi, From Search Engines to Augmented Search Services: An End-User Development Approach, in: ICWE 2017 Springer LNCS Proc., 2017.

[12] G. Bosetti, S. Firmenich, S.E. Gordillo, G. Rossi, An approach for building Mobile Web Applications through Web Augmentation, J. Web Eng. 1 & 2 (2017) 75–102.

[13] G. Bosetti, S. Firmenich, S.E. Gordillo, G. Rossi, M. Winckler, An End User Development Approach for Mobile Web Augmentation, Mob. Inf. Syst. 2017 (2017) 1–28.

[14] G. Bosetti, S. Firmenich, G. Rossi, M. Winckler, Supporting Mobile Web Augmentation by End Users, in: Springer, Cham, 2017: pp. 539–543.

[15] G. Bosetti, S. Firmenich, G. Rossi, M. Winckler, T. Barbieri, Web objects ambient: An integrated platform supporting new kinds of personal web experiences, in: Lect. Notes Comput. Sci. (Including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics), 2016: pp. 563–566.

[16] N.O. Bouvin, B.G. Christensen, K. Grønbæk, F.A. Hansen, HyCon: A framework for context-aware mobile hypermedia, New Rev. Hypermedia Multimed. 9 (2003) 59–88.

[17] N.O. Bouvin, N. Olof, Unifying strategies for Web augmentation, in: Proc. Tenth ACM Conf. Hypertext Hypermedia Return. to Our Divers. Roots Return. to Our Divers. Roots - HYPERTEXT '99, ACM Press, New York, New York, USA, 1999: pp. 91–100.

[18] P. De Bra, A. Aerts, D. Smits, N. Stash, AHA! Version 2.0 More Adaptation Flexibility for Authors, Proc. AACE ELearn'2002. (2002) 240–246.

[19] Brahima Sanou, ICT Facts and Figures 2016, 2017.

[20] J. Brooke, SUS - A quick and dirty usability scale, Usability Eval. Ind. 189 (1996) 4–7.

[21] C. Cappiello, M. Matera, M. Picozzi, A UI-Centric Approach for the End-User Development of Multidevice Mashups, ACM Trans. Web. 9 (2015) 1–40.

[22] D. Carlson, L. Ruge, Ambient Amp: An open framework for dynamically augmenting legacy Websites with context-awareness, in: 2014 IEEE Ninth Int. Conf. Intell. Sensors, Sens. Networks Inf. Process., IEEE, 2014: pp. 1–6.

[23] S. Ceri, F. Daniel, F.M. Facca, M. Matera, Model-driven Engineering of Active

Context-awareness, World Wide Web. 10 (2007) 387–413.

[24] C. Challiol, S. Firmenich, G.A. Bosetti, S.E. Gordillo, G. Rossi, LNCS 8295 - Crowdsourcing Mobile Web Applications, LNCS. 8295 (2013) 223–237.

[25] S.-K. Chang, Visual Languages, Springer Science & Business Media, 2012.

[26] A. Charland, B. Leroux, Mobile application development, Commun. ACM. 54 (2011) 49.

[27] K. Cheverst, H. Turner, T. Do, D. Fitton, Supporting the consumption and co-authoring of locative media experiences for a rural village community: design and field trial evaluation of the SHARC2.0 framework, Multimed. Tools Appl. 76 (2017) 5243–5274.

[28] ComScore, US Mobile App Report 2016, comScore. (2016).

[29] F. Corvetta, M. Matera, R. Medana, E. Quintarelli, V. Rizzo, L. Tanca, Designing and Developing Context-Aware Mobile Mashups: The CAMUS Approach, in: Springer International Publishing, 2015: pp. 651–654.

[30] W.W. Cotterman, K. Kumar, User cube: a taxonomy of end users, Commun. ACM. 32 (1989) 1313–1320.

[31] G. D'Amico, S. Ercoli, A. Del Bimbo, A framework for itinerary personalization in cultural tourism of smart cities, in: CEUR Workshop Proc., 2013.

[32] J. Danado, F. Paternò, Puzzle: A Visual-Based Environment for End User Development in Touch-Based Mobile Phones, in: Proc. 4th Int. Conf. Human-Centered Softw. Eng., Springer-Verlag, 2012: pp. 199–216.

[33] F. Daniel, M. Matera, Mashups : concepts, models and architectures, n.d.

[34] O. Díaz, I. Aldalur, C. Arellano, H. Medina, S. Firmenich, Web mashups with webmakeup, in: Commun. Comput. Inf. Sci., 2016: pp. 82–97.

[35] O. Díaz, C. Arellano, The Augmented Web: Rationales, Opportunities, and Challenges on Browser-Side Transcoding, ACM Trans. Web. 9 (2015) 1–30.

[36] O. Díaz, C. Arellano, M. Azanza, A language for end-user web augmentation, ACM Trans. Web. 7 (2013) 1–51.

[37] A. Doan, R. Ramakrishnan, A.Y. Halevy, Crowdsourcing systems on the World-Wide Web, Commun. ACM. 54 (2011) 86.

[38] N. Eagle, Txteagle: Mobile crowdsourcing, in: Lect. Notes Comput. Sci. (Including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics), 2009: pp. 447–456.

[39] C. Emmanouilidis, R.A. Koutsiamanis, A. Tasidou, Mobile guides: Taxonomy of architectures, context awareness, technologies and applications, J. Netw. Comput. Appl. 36 (2013) 103–125.

[40] J.P. Espada, R.G. Crespo, O.S. Martínez, B. Cristina Pelayo G-Bustelo, J.M.C. Lovelle, Extensible architecture for context-aware mobile web applications, Expert Syst. Appl. 39 (2012) 9686–9694.

[41] E. Estellés-Arolas, F. González-Ladrón-de-Guevara, Towards an integrated crowdsourcing definition, J. Inf. Sci. 38 (2012) 189–200.

[42] L. Etaati, D. Sundaram, Adaptive tourist recommendation system: conceptual frameworks and implementations, Vietnam J. Comput. Sci. 2 (2014) 95–107.

[43] I.R. Félix, L.A. Castro, L.-F. Rodríguez, É.C. Ruiz, Mobile Phone Sensing: Current Trends and Challenges, in: Int. Conf. Ubiquitous Comput. Ambient Intell., 2015: pp. 369–374.

[44] S. Firmenich, G. Bosetti, G. Rossi, M. Winckler, Flexible Distribution of Existing Web Interfaces: An Architecture Involving Developers and End-Users, in: Springer, Cham, 2016: pp. 200–207.

[45] S. Firmenich, G. Bosetti, G. Rossi, M. Winckler, End-user software engineering for the personal web, in: 2017 IEEE/ACM 39th Int. Conf. Softw. Eng. Companion, IEEE, 2017: pp. 216–218.

[46] S. Firmenich, G. Bosetti, G. Rossi, M. Winckler, T. Barbieri, Abstracting and Structuring Web Contents for Supporting Personal Web Experiences, in: Springer, Cham, 2016: pp. 77–95.

[47] S. Firmenich, G. Bosetti, G. Rossi, M. Winckler, J.M. Corletto, Distributed Web Browsing: supporting frequent uses and opportunistic requirements, Univers. Access Inf. Soc. (n.d.).

[48] S. Firmenich, G. Rossi, S. Gordillo, M. Winckler, A crowdsourced approach for concern-sensitive integration of information across the web, J. Web Eng. 10 (2011) 289–315.

[49] B. Fling, Mobile Design and Development: Practical concepts and techniques for creating mobile sites and web apps, O'Reilly, 2009.

[50] G.H. Forman, J. Zahorjan, The challenges of mobile computing, Computer (Long. Beach. Calif). 27 (1994) 38–47.

[51] A. Fortier, G. Rossi, S.E. Gordillo, C. Challiol, Dealing with variability in context-aware mobile software, J. Syst. Softw. 83 (2010) 915–936.

[52] D. Frajberg, M. Urbieta, G. Rossi, W. Schwinger, Volatile Functionality in Action: Methods, Techniques and Assessment, in: Springer, Cham, 2016: pp. 59–76.

[53] R. Francese, M. Risi, G. Tortora, M. Tucci, Visual Mobile Computing for Mobile End-Users, IEEE Trans. Mob. Comput. 15 (2016) 1033–1046.

[54] E. Gamma, Design patterns : elements of reusable object-oriented software, Addison-Wesley, 1995.

[55] G. Ghiani, M. Manca, F. Paternó, C. Porta, Beyond responsive design: Context-dependent multimodal augmentation of web applications, in: Lect. Notes Comput. Sci., 2014: pp. 71–85.

[56] P. Gupta, EVOLVEMENT OF MOBILE GENERATIONS : 1G T o 5G, Int. J. Technol. Res. Eng. 1 (2013) 152–157.

[57] A. Gustafson, J. Zeldman, Adaptive web design : crafting rich experiences with progressive enhancement, Easy Readers, 2011.

[58] D. Halbert, Programming by example, 1984.

[59] S. Harper, C. Goble, S. Pettitt, proXimity: Walking the link, J. Digit. Inf. 5 (2004).

[60] A.R. Hevner, S.T. March, J. Park, S. Ram, Design Science in Information Systems Research, MIS Q. 28 (2004) 75–105.

[61] B. Hoh, T. Yan, D. Ganesan, K. Tracton, T. Iwuchukwu, J.S. Lee, TruCentive: A game-theoretic incentive platform for trustworthy mobile crowdsourcing parking services, in: IEEE Conf. Intell. Transp. Syst. Proceedings, ITSC, 2012: pp. 160–166.

[62] T. Höllerer, S. Feiner, Mobile Augmented Reality, 2004.

[63] J. yi Hong, E. ho Suh, S.J. Kim, Context-aware systems: A literature review and classification, Expert Syst. Appl. 36 (2009) 8509–8522.

[64] R. Horváth, M. Šimko, Enriching the web for vocabulary learning, in: Lect. Notes Comput. Sci. (Including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics), 2013: pp. 609–610.

[65] J. Howe, The Rise of Crowdsourcing, Wired Mag. 14 (2006) 1–5.

[66] A.J. Ko, R. Abraham, L. Beckwith, A. Blackwell, M. Burnett, M. Erwig, C. Scaffidi, J. Lawrance, H. Lieberman, B. Myers, M.B. Rosson, G. Rothermel, M. Shaw, S. Wiedenbeck, The state of the art in end-user software engineering., ACM Comput. Surv. 43 (2011) 1–44.

[67] A. Korthaus, W. Dai, Crowdsourcing in heterogeneous networked environments - Opportunities and challenges, in: Proc. 2012 15th Int. Conf. Network-Based Inf. Syst. NBIS 2012, 2012: pp. 483–488.

[68] M. Leatherbury, Developing User Interfaces: Ensuring Usability Through Product & Process, Inf. Process. Manag. 31 (1995) 156–157.

[69] E. Lee, H.-J. Joo, Developing lightweight context-aware service mashup applications, in: 15th Int. Conf. Adv. Commun. Technol., IEEE, 2013.

[70] S. Li, T. Xie, N. Tillmann, A comprehensive field study of end-user programming on mobile devices, in: 2013 IEEE Symp. Vis. Lang. Hum. Centric Comput., IEEE, 2013: pp. 43–50.

[71] H. Lieberman, F. Paternò, M. Klann, V. Wulf, End-User Development: An Emerging Paradigm, in: End User Dev., Springer Netherlands, Dordrecht, 2006: pp. 1–8.

[72] G. Manjunath, M.N. Murty, D. Sitaram, Tasklets: enabling end user programming of web widgets, Int. J. Web Eng. Technol. 8 (2013) 264.

[73] D. Martín, C. Lamsfus, A. Alzua-Sorzabal, A cloud-based platform to develop context-aware mobile applications by domain experts, Comput. Stand. Interfaces. 44 (2016) 177–184.

[74] S. McDirmid, Living it up with a live programming language, ACM SIGPLAN Not. 42 (2007) 623.

[75] D.E. Millard, D.C. De Roure, D.T. Michaelides, M.K. Thompson, M.J. Weal, Navigational Hypertext Models for Physical Hypermedia environments, in: Proc. Fifteenth ACM Conf. Hypertext Hypermedia - HYPERTEXT '04, ACM Press, New York, New York, USA, 2004: p. 110.

[76] B. Myers, J. Pane, A. Ko, Natural programming languages and environments, Commun. ACM. 47 (2004) 47.

[77] B.A. Nardi, A Small Matter of Programming: Perspectives on End User Computing, 1993.

[78] M. Nebeling, M.C. Norrie, Tools and architectural support for crowdsourced adaptation of web interfaces, in: Lect. Notes Comput. Sci. (Including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics), 2011: pp. 243–257.

[79] T. Nestler, Towards a mashup-driven end-user programming of SOA-based applications, Proc. 10th Int. Conf. Inf. Integr. Web-Based Appl. Serv. - iiWAS '08. (2008) 551.

[80] R. Oppermann, Adaptively supported adaptability, Int. J. Hum. Comput. Stud. 40 (1994) 455–472.

[81] J.F. Pane, C. Ratanamahatana, B. Myers, Studying the language and structure in non-programmers' solutions to programming problems, Int. J. Hum. Comput. Stud. 54 (2001) 237–264.

[82] F. Paternò, End User Development: Survey of an Emerging Field for Empowering People, ISRN Softw. Eng. 2013 (2013) 1–11.

[83] V. Pejovic, M. Musolesi, Anticipatory mobile computing: A survey of the state of the art and research challenges, ACM Comput. Surv. 47 (2015) 1–29.

[84] F. Pittarello, L. Bertani, CASTOR: learning to create context-sensitive and

emotionally engaging narrations in-situ, in: Proc. 11th Int. Conf. Interact. Des. Child. - IDC '12, ACM Press, New York, New York, USA, 2012: p. 1.

[85] J. Rode, Y. Bhardwaj, M. Pérez-Quiñones, M. Rosson, J. Howarth, As Easy as "Click": End-User Web Engineering, Web Eng. 3579 (2005) 478–488.

[86] B. Sanou, ICT Facts & Figures. The world in 2015., Itu 150 Años (1865 - 2015). (2015) 6.

[87] C. Scaffidi, M. Shaw, B. Myers, Estimating the numbers of end users and end user programmers, in: Proc. - 2005 IEEE Symp. Vis. Lang. Human-Centric Comput., 2005: pp. 207–214.

[88] R. Schaller, Mobile tourist guides: bridging the gap between automation and users retaining control of their itineraries, in: Proc. 5th Inf. Interact. Context Symp. - IIiX '14, ACM Press, New York, USA, 2014: pp. 320–323.

[89] Schilit, W. Noah, A system architecture for context-aware mobile computing, (1995).

[90] W. Schwinger, C. Grun, B. Proll, W. Retschitzegger, a Schauerhuber, Context-awareness in Mobile Tourism Guides – A Comprehensive Survey, Handb. Res. Mob. Multimedia, Second Ed. 2 (2007) 298–314.

[91] J. Seifert, B. Pfleging, E. del Carmen Valderrama Bahamóndez, M. Hermes, E. Rukzio, A. Schmidt, Mobidev: a tool for creating apps on mobile phones, in: Proc. 13th Int. Conf. Hum. Comput. Interact. with Mob. Devices Serv. - MobileHCI '11, ACM Press, New York, New York, USA, 2011: p. 109.

[92] M. Shaw, What makes good research in software engineering?, Int. J. Softw. Tools Technol. …. 4 (2002) 1–7.

[93] N.C. Shu, Visual Programming Languages: A Perspective and a Dimensional Analysis, in: Vis. Lang., Springer US, Boston, MA, 1986: pp. 11–34.

[94] T. Sohn, K.A. Li, W.G. Griswold, J.D. Hollan, A diary study of mobile information needs, in: Proceeding Twenty-Sixth Annu. CHI Conf. Hum. Factors Comput. Syst. - CHI '08, 2008: p. 433.

[95] K.T. Stolee, S. Elbaum, A. Sarma, End-user programmers and their communities: An artifact-based analysis, Int. Symp. Empir. Softw. Eng. Meas.

(2011) 147–156.

[96] A.K. Talukdar, Mobile Computing, 2E, Tata McGraw-Hill Education, 2010.

[97] A. Tamilin, I. Carreras, E. Ssebaggala, A. Opira, N. Conci, Context-aware mobile crowdsourcing, in: Proc. 2012 ACM Conf. Ubiquitous Comput. - UbiComp '12, 2012: p. 717.

[98] R. Wasinger, M. Fry, S. Gerber, O. Iivonen, J. Kay, B. Kummerfeld, Client-side user modelling across multiple mobile applications / Rainer Wasinger et al. [...]. - Version details - Trove, Sydney, 2011.

[99] W. Van Woensel, S. Casteleyn, O. De Troyer, A Framework for Decentralized, Context-Aware Mobile Applications Using Semantic Web Technology, in: Confed. Int. Work. Posters, 2009: pp. 88–97.

[100] W. Van Woensel, S. Casteleyn, O. De Troyer, A generic approach for on-the-fly adding of context-aware features to existing websites, in: Proc. 22nd ACM Conf. Hypertext Hypermedia - HT '11, ACM Press, New York, New York, USA, 2011: p. 143.

[101] T. Yan, M. Marzilli, R. Holmes, D. Ganesan, M. Corner, mCrowd: a platform for mobile crowdsourcing, Conf. Embed. Networked Sens. Syst. (2009) 347–348.

[102] Y. You, V.-V. Mattila, Visualizing Web Mash-ups for In-situ Vision-based Mobile AR Applications, in: Proc. 2013 ACM Conf. Pervasive Ubiquitous Comput. Adjun. Publ., 2013: pp. 271–274.

[103] X. Zhu, B. Song, Y. Ni, Y. Ren, R. Li, Outsourcing and Crowdsourcing - From Building All-Round Capabilities to Outsourcing and Crowdsourcing, in: Bus. Trends Digit. Era, Springer Singapore, Singapore, 2016: pp. 105–121.

Anexo A: Análisis de sitios Web populares

Las tablas de este anexo se han construido con los datos extraídos del sitio oficial de Alexa[74], consultado el 13/01/2017 a las 10:05 am, y extraídos mediante scripting.

Tabla 3. 10 primeros sitios del Top500 de Alexa

	Native counterpart in Google Play			*Website*	
	Supported	Name	URL	Responsive	URL
Google	✓	googlequicksearchbox	https://play.google.com/store/apps/details?id=com.google.android.googlequicksearchbox	✓	https://m.youtube.com
YouTube	✓	youtube	https://play.google.com/store/apps/details?id=com.google.android.youtube	✓	https://m.youtube.com
Facebook	✓	Facebook	https://play.google.com/store/apps/details?id=com.facebook.katana	✓	https://m.facebook.com
Baidu	✓	baidu.searchbox, baidu.news, baidutranslate, baidu.dict	https://play.google.com/store/apps/details?id=com.baidu.searchbox	✓	https://m.baidu.com/?from=844b&vit=fps
Yahoo	✓	yahoo search	https://play.google.com/store/apps/details?id=com.yahoo.mobile.client.android.search	✓	https://espanol.yahoo.com
Wikipedia	✓	wikipedia	https://play.google.com/store/apps/details?id=org.wikipedia	✓	https://es.m.wikipedia.org/wiki/Wikipedia:Portada
Amazon	✓	amazon	https://play.google.com/store/apps/details?id=com.amazon.mShop.android.shopping	✓	https://www.amazon.com

[74] http://www.alexa.com/topsites/global;0

Qq.com	✓	mobile qq	https://play.google.com/store/apps/details?id=com.tencent.mobileqq	✓ http://xw.qq.com/index.htm
Live.com	✓	Microsoft Outlook	https://play.google.com/store/apps/details?id=com.microsoft.office.outlook	✓ https://outlook.live.com/owa/#id=64855
Taobao.com	✓	taobao	https://play.google.com/store/apps/details?id=com.taobao.taobao&hl=es	✓ http://m.intl.taobao.com/?sprefer=sypc00

Tabla 4. Top500 de Argentina por Alexa

Ranking	*Web site*	*News*	*Responsive*	*In Google Play*	*Ranking*	*Web site*	*News*	*Responsive*	*In Google Play*
1	Google.com.ar				251	Gamestorrents.com			
2	Youtube.com				252	Animeflv.me			
3	Google.com				253	Slideshare.net			
4	Mercadolibre.com.ar				254	Eltribuno.info	✓	✓	✓
5	Facebook.com				255	Elpais.com	✓	✓	✓
6	Live.com				256	1fichier.com			
7	Yahoo.com				257	Pulseonclick.com			
8	Taringa.net				258	Mundosexanuncio.com			
9	Blogspot.com.ar				259	Upornia.com			
10	Wikipedia.org				260	Leadzupc.com			
11	Clarin.com	✓	✓	✓	261	Chaturbate.com			
12	Infobae.com	✓	✓	✓	262	Falabella.com.ar			
13	Lanacion.com.ar	✓	✓	✓	263	Lapaginamillonaria.com			
14	Instagram.com				264	Subtorrents.com			
15	Lagaceta.com.ar	✓	✓	✓	265	Radiocut.fm			
16	Adf.ly				266	Promiedos.com.ar			
17	Twitter.com				267	Corazondemelon.es			
18	Msn.com				268	Wetransfer.com			
19	Netflix.com				269	Laarena.com.ar			
20	Reimageplus.com				270	Google.com.co			
21	Mediafire.com				271	Twimg.com			
22	Popads.net				272	Uba.ar			
23	Onclkds.com				273	Gba.gov.ar			
24	Afip.gob.ar				274	Youporn.com			

25	Whatsapp.com				275	Bc.vc			
26	Ole.com.ar	✓	✓	✓	276	Piliapp.com			
27	T.co				277	Diariodecuyo.com.ar	✓	✓	x
28	Uptodown.com				278	Office.com			
29	Bigbangnews.com	✓	✓	✓	279	Redirectvoluum.com			
30	Tn.com.ar	✓	✓	✓	280	Disneylatino.com			
31	Mundo.com				281	Abc.gov.ar			
32	Redlink.com.ar				282	Intercambiosvirtuales.org			
33	Wordpress.com				283	Files.wordpress.com			
34	Santanderrio.com.ar				284	9gag.com			
35	Wikia.com				285	Filmaffinity.com			
36	Tumblr.com				286	Pronto.com.ar			
37	Linkedin.com				287	Trivago.com.ar			
38	Pornhub.com				288	Doubleclick.net			
39	Popcash.net				289	Identi.li			
40	Ouo.io				290	Arba.gov.ar			
41	Paisdelosjuegos.com.ar				291	Adexchangeprediction.com			
42	Minutouno.com	✓	✓	x	292	Xl415.com			
43	Animeflv.net				293	Telefe.com			
44	Misionesonline.net	✓	✓	x	294	Yaske.ro			
45	Shink.in				295	Bbc.com			
46	Pagomiscuentas.com				296	Nicesearches.com			
47	Rackcdn.com				297	Linkshrink.net			
48	Anses.gob.ar				298	Weather.com			
49	Askcom.me				299	Teatroporno.com			
50	Repelis.tv				300	Exoclick.com			
51	Plataforma10.com.ar				301	Juegosdechicas.com			
52	Microsoft.com				302	Flirt4free.com			
53	Bancogalicia.com.ar				303	Ruta0.com			
54	Mercadolibre.com				304	Rionegro.com.ar	✓	✓	✓
55	Despegar.com.ar				305	Groupon.com.ar			
56	Perfil.com	✓	✓	x	306	Wittyfeed.com			
57	Adbooth.com				307	Citibank.com			
58	Bongacams.com				308	Amazon.es			
59	4dsply.com				309	Scribd.com			
60	Humbleness.xyz				310	Hackstore.net			
61	Lavoz.com.ar	✓	✓	✓	311	Rarbg.to			
62	Savefrom.ne				312	Mitula.com.ar			
63	Diarioregistrado.com	✓	✓	x	313	Somosmovies.com			
64	Amazon.com				314	Givemesport.com			
65	Booking.com				315	Cookpad.com			
66	Visa.com.ar				316	Bcvc.mobi			
67	Pelisplus.tv				317	Cronica.com.ar	✓	✓	✓
68	Smn.gov.ar				318	Directv.com.ar			
69	Afip.gov.ar				319	Paypal.com			
70	Pagina12.com.ar	✓	✓	✓	320	Media.tumblr.com			
71	Pelispedia.tv				321	Contextotucuman.com			
72	Juegos.com				322	Steamcommunity.com			
73	Sh.st				323	Windguru.cz			
74	Xvideos.com				324	Hipotecario.com.ar			
75	Blogspot.com				325	Hilltopads.net			
76	Olx.com.ar				326	Cablevisionfibertel.com.ar			
77	Friv.com				327	Sensacine.com			
78	Musica.com				328	Argenprop.com			

79	Elliberal.com.ar	✓	✓	x	329	Rosario3.com	✓	✓	✓
80	Imdb.com				330	Outbrain.com			
81	Youtube-mp3.org				331	Xnxx.com			
82	Google.es				332	Cnet.com			
83	Ebay.com				333	Dafiti.com.ar			
84	Gnula.nu				334	Thepiratebay.org			
85	Buenosaires.gob.ar				335	Yts.ag			
86	Bancainternet.com.ar				336	Telecom.com.ar			
87	Ciudad.com.ar				337	Latam.com			
88	Mega-cine.com.ar				338	Y8.com			
89	Bet365.com				339	Lan.com			
90	Reddit.com				340	Yoob.com			
91	Codeonclick.com				341	G2a.com			
92	Bna.com.ar				342	Santafe.gov.ar			
93	Roblox.com				343	Abc.gob.ar			
94	Poringa.net				344	Neobux.com			
95	Newtabtv.com				345	Wixsite.com			
96	Ambito.com	✓	✓	✓	346	Cartoonnetwork.com.ar			
97	Zytpirwai.net				347	A10.com			
98	Cloudfront.net				348	Bajalogratis.com			
99	Paraloscuriosos.com				349	Hclips.com			
100	Cienradios.com				350	Thewhizproducts.com			
101	Jkanime.net				351	Mylucky123.com			
102	Softonic.com				352	Icbc.com.ar			
103	Bbv.com.ar				353	Cba.gov.ar			
104	Alibaba.com				354	Tiendeo.com.ar			
105	Bancogalicia.com				355	Lagacetasalta.com.ar	✓	✓	✓
106	Ask.com				356	Elintransigente.com	✓	✓	x
107	Xhamster.com				357	Infd.edu.ar			
108	Txxx.com				358	Muyzorras.com			
109	Dailymotion.com				359	Trovitargentina.com.ar			
110	Deviantart.com				360	Kismia.com			
111	Steampowered.com				361	Telefonica.com.ar			
112	Hola.com				362	Viid.me			
113	Blogger.com				363	Seriesbang.to			
114	Conectarigualdad.gob.ar				364	Trafficserving.com			
115	Bbvafrances.com.ar				365	Prestoris.com			
116	Tripadvisor.com.ar				366	20minutos.es	✓	✓	✓
117	Wordreference.com				367	Diariopopular.com.ar	✓	✓	✓
118	Clicksgear.com				368	Oca.com.ar			
119	Nanoadexchange.com				369	Canalporno.com			
120	Vix.com				370	Bidverdrs.com			
121	Personal.com.ar				371	Genial.guru			
122	Subdivx.com				372	Mendoza.gov.ar			
123	Seriesflv.net				373	Paginasamarillas.com.ar			
124	Stackoverflow.com				374	Bezimeni.biz			
125	Sosodesktop.com				375	Spankbang.com			
126	Upsocl.com				376	Inkagames.com			
127	Adnetworkperformance.com				377	Animeid.tv			
128	Adobe.com				378	Easy.com.ar			
129	Lacasadelcurioso.com				379	Dumedia.ru			

130	Redtube.com			
131	Fiuxy.net			
132	Zapmeta.com.ar			
133	Alamaula.com			
134	Pinterest.com			
135	Computrabajo.com.ar			
136	Iwanttodeliver.com			
137	Animeyt.tv			
138	Eldestapeweb.com	✓	✓	x
139	Bp.blogspot.com			
140	Minijuegos.com			
141	Lacapital.com.ar	✓	✓	✓
142	Zippyshare.com			
143	Socialnewpages.com			
144	Leagueoflegends.com			
145	Onlinevideoconverter.com			
146	Ccm.net			
147	Garbarino.com			
148	Diariopanorama.com	✓	✓	✓
149	Googleusercontent.com			
150	Serviporno.com			
151	Trackingclick.net			
152	Aliexpress.com			
153	Macro.com.ar			
154	Short.am			
155	Eltrecetv.com.ar			
156	Claro.com.ar			
157	Fravega.com			
158	Miradetodo.net			
159	Macrojuegos.com			
160	Spotify.com			
161	Mega.nz			
162	Cinecalidad.com			
163	Bancoprovincia.com.ar			
164	Musimundo.com			
165	Tradeadexchange.com			
166	Bing.com			
167	Lacuerda.net			
168	Nametests.com			
169	Bancopatagonia.com.ar			
170	Mysagagame.com			
171	Imgur.com			
172	Twitch.tv			
173	Dropbox.com			
174	Movistar.com.ar			
175	Compucalitv.com			
176	Zonaprop.com.ar			
177	Verseriesynovelas.tv			
178	Argentino.com.ar			
179	Vimeo.com			
180	Github.com			
181	3djuegos.com			
182	Accuweather.com			
380	Towpath.xyz			
381	Coto.com.ar			
382	Uploaded.net			
383	Witchcraftcash.com			
384	Guioteca.com			
385	Manhunt.net			
386	Elsol.com.ar	✓	✓	✓
387	Supervielle.com.ar			
388	Ellitoral.com	✓	✓	✓
389	Minecrafteo.com			
390	Locopelis.com			
391	Employeesdirectorships.com			
392	Airbnb.com.ar			
393	Utn.edu.ar			
394	Utorrent.com			
395	Samsung.com			
396	All-czech.com			
397	Camdolls.com			
398	Freepik.es			
399	Mundotkm.com			
400	Aquipelis.net			
401	Kogama.com			
402	Directrev.com			
403	Yoreparo.com			
404	Mozilla.org			
405	Jw.org			
406	Infoclima.com			
407	Anses.gov.ar			
408	Pibeporno.com			
409	Kn3.net			
410	Facilisimo.com			
411	Worldoftanks.com			
412	Tube8.com			
413	Iprofesional.com			
414	Playgroundmag.net			
415	Bancopatagonia.com			
416	Descargatelocorp.com			
417	Depositfiles.org			
418	Amazonaws.com			
419	Mbtrx.com			
420	Lanueva.com	✓	✓	✓
421	Twoo.com			
422	Petardas.com			
423	Telefenoticias.com.ar	✓	✓	✓
424	Diarioviral.com			
425	Jugandoonline.com.ar			
426	Pwwysydh.com			
427	Diaadia.com.ar	✓	✓	✓
428	Elonce.com	✓	✓	x
429	Bancociudad.com.ar			
430	Agar.io			
431	Dafont.com			
432	Skype.com			

183	Cronista.com	✓	✓	✓	433	Baiduccdn.org			
184	Vk.com				434	Microsoftonline.com			
185	Adplxmd.com				435	Masterconsultas.com.ar			
186	Livejasmin.com				436	Pelis24.com			
187	Amisites.com				437	Ytimg.com			
188	Netshoes.com.ar				438	Gameforge.com			
189	Mercadopago.com.ar				439	Atozmanuals.com			
190	Videodownloadconverter.com				440	Trueactivist.com			
191	Doublepimp.com				441	Porn.es			
192	Porngames.adult				442	Amigaironica.es			
193	Rolloid.net				443	Lapoliticaonline.com	✓	✓	x
194	Fbcdn.net				444	Demotores.com.ar			
195	Onlinetodaynews.com	✓	x	x	445	Videosxxxputas.xxx			
196	Blastingnews.com	✓	x	x	446	Marca.com			
197	Ratingcero.com	✓	✓	x	447	Speedtest.net			
198	Apple.com				448	Quepasasalta.com.ar			
199	Tumangaonline.com				449	Mejortorrent.com			
200	Tarjetanaranja.com				450	Carrefour.com.ar			
201	Losandes.com.ar	✓	✓	✓	451	Fenix951.com.ar	✓	x	✓
202	Giphy.com				452	Buenamente.com			
203	Popmyads.com				453	Jimdo.com			
204	Ampclicks.com				454	Eldoce.tv			
205	Mundogaturro.com				455	Akamaihd.net			
206	Wattpad.com				456	Abc.es			
207	Wikihow.com				457	Googlevideo.com			
208	Thewhizmarketing.com				458	Badoo.com			
209	Kizi.com				459	Advleniv.com			
210	Soundcloud.com				460	Elmundo.es	✓	✓	✓
211	Theopenads.com				461	Thefreedictionary.com			
212	Mercadopago.com				462	Zonajobs.com.ar			
213	Pinimg.com				463	Mercadolibre.cl			
214	Diariouno.com.ar	✓	✓	x	464	Mercadoshops.com.ar			
215	Rt.com	✓	✓	✓	465	Wikimedia.org			
216	Vercomicsporno.com				466	Ehowenespanol.com			
217	Cadena3.com	✓	✓	✓	467	Espn.com.ar	✓	✓	✓
218	Srvr.work				468	Gamesfull.org			
219	Openload.co				469	Almundo.com.ar			
220	Offerreality.com				470	Hoyts.com.ar			
221	Ouo.press				471	Quizzstar.com			
222	Correoargentino.com.ar				472	Buscardatos.com			
223	Avg.com				473	Gamepedia.com			
224	Fvpimageviewer.com				474	Videosdemadurasx.com			
225	Laafoka.com				475	Eslamoda.com			
226	Adscpm.net				476	Wix.com			
227	Onclickpredictiv.com				477	Educ.ar			
228	Eldia.com	✓	✓	✓	478	Xataka.com			
229	Meteored.com.ar				479	Pjn.gov.ar			
230	Macronline.com.ar				480	Saltbox.xyz			
231	Terraclicks.com				481	Cuitonline.com			
232	Myway.com				482	Plusnetwork.com			
233	Speedy.com.ar				483	Autoblog.com.ar			

234	Primiciasya.co	✓	✓	x	484	Lightinthebox.com			
235	Cuevana2.tv				485	Argenteam.net			
236	Besoccer.com				486	Nuevodiarioweb.com.ar	✓	✓	✓
237	Spotscenered.info				487	Todopelishd.com			
238	Tycsports.com	✓	✓	✓	488	Azlyrics.com			
239	Mdzol.com				489	Lolskill.net			
240	Telam.com.ar	✓	✓	✓	490	Binbox.io			
241	Accessbanking.com.ar				491	Estrenosdoramas.net			
242	Pelis24.tv				492	Slither.io			
243	Bancocredicoop.coop				493	Pasionfutbol.com	✓	✓	x
244	Hsbc.com.ar				494	Mega-dvdrip.com			
245	Sintelevisor.com				495	Compraensanjuan.com			
246	Bumeran.com.ar				496	Tujugada.com.ar			
247	Axeso5.com				497	Donpornogratis.com			
248	4shared.com				498	Ypf.com			
249	Aerolineas.com.ar				499	Deviantart.net			
250	Onclickads.net				500	Github.io			

Anexo B: Adaptación Web client-side

Cuando se trata de un ambiente de escritorio, aumentar la Web del lado del cliente es una tarea para la cual existen diferentes estrategias. Una de ellas es utilizando un weaver, como Greasemonkey, que permita instalar un script que manipule el DOM de una o varias páginas Web.

A continuación, se presenta el Script 1, que permitió generar la aumentación presentada en la Figura 4. Tal como se puede apreciar en los metadatos, en la parte superior comentada, este script correrá sobre todos los artículos de MDN dentro de la jerarquía de webextensions (ver `include`).

La primera instrucción del script permite obtener el elemento que será "contenedor" de la aumentación, `rightPanel`, sobre el cual se insertará un panel extra con preguntas de Stackoverflow. Luego, se crea un div que representa el `panel` de preguntas, con otro div interno como `header` para indicar el título «Related Questions at StackOverflow».

Después se envía una petición a Stackoverflow mediante `GM_xmlhttpRequest`, cuya URL contiene un query con el título del artículo en MDN como palabra clave. Como respuesta de la petición, se obtiene el texto que conforma el documento de la página con los resultados de stackoverflow. El mismo puede ser utilizado para parsear el documento y seleccionar (línea 33) todos los nodos que contienen una pregunta. Por cada uno de ellos, se crea un anchor por cada pregunta (línea 36) y se lo agrega al panel (línea 44). Finalmente, se agrega el panel creado al `rightPanel`.

```
1 // ==UserScript==
2 // @name         MDN + StackOverflow
3 // @namespace    lifia
4 // @include      https://developer.mozilla.org/en-US/Add-
ons/WebExtensions/*
5 // @version      1
6 // @grant        GM_xmlhttpRequest
7 // ==/UserScript==
8
9 try{
10     var rightPanel = document.querySelector("#wiki-right");
11
12     var panel = document.createElement("div");
13         panel.style.border = "2px solid orange";
14         panel.style.padding = "0.5em";
15
16     var header = document.createElement("div");
17         header.style.marginBottom = "0.5em";
18         header.style.color = "white";
19         header.style.background = "#F48024";
20         header.style.textAlign = "center";
21         header.innerHTML = "Related Questions at StackOverflow";
22
23         panel.appendChild(header);
24
```

```
25          GM_xmlhttpRequest({
26              method: "GET",
27              url: "https://stackoverflow.com/search?q=webextensions+" +
28                  document.querySelector("h1").textContent,
29              onload: function(response) {
30
31                  var responseXML = new DOMParser().parseFromString(
32                      response.responseText, "text/html");
33                  var questions = responseXML.querySelectorAll(
34                      ".search-result.question-summary").forEach(
35                      function(elem) {
36                      var question = elem.children[1].querySelector("a");
37                          question.onclick = function(evt){
38                              evt.preventDefault(); evt.preventDefault();
39                              var url = question.href.replace(
40                                  question.origin,
41                                  "https://stackoverflow.com");
42                              window.open(url);
43                          };
44                      panel.appendChild(elem.children[1].children[0]);
45                  });
46              }
47          });
48
49      rightPanel.appendChild(panel);
50
51  }catch(err){
52      console.log(err)
53  }
```

Script 1. Userscript que injecta un panel con preguntas de StackOverflow en MDN

Anexo C: Adaptación Web Móvil del lado del cliente

Para demostrar la factibilidad técnica del enfoque, se llevaron a cabo algunas adaptaciones mediante el uso de la consola del navegador, dado que desde ella se cuenta con los mismos privilegios y la misma API que desde una extensión. Cada adaptación se presenta en los siguientes scripts, que deben copiarse y ejecutarse en dicha consola. Para ello, es necesario contar con:

- Firefox Desktop[75] en un entorno de escritorio (se utilizó la versión 55.0b8)
- Android Debug Bridge[76]
- Firefox for Android[77] (46 onwards)
- Un cable USB para conectar ambos dispositivos

Desde el entorno de escritorio se debe habilitar el debugging remoto den navegador[78], mientras que desde el dispositivo móvil se debe habilitar tanto el debugging remoto del dispositivo[79] como del navegador[80].

Para poder copiar y ejecutar los scripts en el contexto de la página que adaptan, se requiere establecer una conexión desde el navegador desktop al navegador móvil[81] y seleccionar el proceso principal, que abrirá una instancia de las Firefox Developer Tools[82]. Una de sus pestañas permite acceder a la consola del navegador remoto. Se debe tener en cuenta que la misma permite ejecutar código desde el contexto del navegador, con sus respectivos permisos, y no desde el de una página Web. Otra diferencia con la consola Web es que, en la del navegador, es posible acceder a los objetos disponibles a la consola Web pero no siempre en forma directa. Por ejemplo, el objeto *window* no es el que convencionalmente se encuentra en el contexto de una página Web. En este caso, se trata de una instancia de *Chrome Window*, que además permite acceder y manipular componentes propios del navegador. Si se quiere acceder al documento de la página Web actual, es necesario obtener una instancia del *window* correspondiente, que se obtiene solicitándosela al browser seleccionado, tal como se muestra en la primera línea del Script 2. Luego, para poder solicitarle a este objeto su documento, primero se le debe aplicar un wrapper que otorgue un acceso transparente al objeto subyacente. Cada vez que un objeto dentro de un contexto privilegiado del

[75] https://www.mozilla.org/fr/firefox/channel/desktop/
[76] https://developer.android.com/studio/command-line/adb.html
[77] https://ftp.mozilla.org/pub/firefox/releases/
[78] https://developer.mozilla.org/en-US/docs/Tools/Remote_Debugging/Debugging_Firefox_Desktop
[79] https://developer.android.com/studio/debug/dev-options.html
[80] https://developer.mozilla.org/en-US/docs/Tools/Remote_Debugging/Debugging_Firefox_for_Android_with_WebIDE
[81] https://developer.mozilla.org/fr/docs/Outils/D%C3%A9bogage_distant/Firefox_for_Android
[82] https://developer.mozilla.org/en-US/docs/Tools

navegador accede a código con menores privilegios, debe aplicar este wrapper, tal como se muestra en la segunda línea del script.

```
var waivedWindow = window.BrowserApp.selectedBrowser.contentWindow;
Components.utils.waiveXrays(waivedWindow).document)
```

Script 2. Accediendo al documento de la página Web actual

A continuación, se presentan los scripts que fueron ejecutados para conseguir las adaptaciones de los escenarios motivadores de la Sección 4.1. En todos los casos, una clase abstracta fue definida adicionalmente al inicio de cada ejecución. La misma se presenta en el Script 3, y representa un augmenter con toda su funcionalidad común a las adaptaciones subyacentes (como ocultar todos los elementos del DOM, obtener un elemento del mismo en base a un query selector[83]) y métodos abstractos (*adapt*, que dispara la adaptación).

```
function Augmenter(document){
    this.hideAllDomElements = function(){
        var domElems = this.getAllDomElements();
        for (var i = domElems.length - 1; i >= 0; i--) {
            domElems[i].style.display="none";
        }
    };
    this.getAllDomElements = function(){
        return document.querySelectorAll(
                "div, a, img, span, label, ul, li, p");
    };
    this.showElementsToTheRoot = function(elem){
        while (elem.style) {
            elem.style.display = "";
            elem = elem.parentNode;
        };
    };
    this.showElementsToTheLeaf = function(elem){

        var childNodes = elem.querySelectorAll("*");
        for (var i = childNodes.length - 1; i >= 0; i--) {
            childNodes[i].style.display = "";
        }
    };
    this.getElement = function(selector){
        return document.querySelector(selector);
    };
    this.adapt = function(){}
}
```

Script 3. Clase abstracta del augmenter

En todos los casos, estos scripts representan una versión reducida de la idea propuesta, focalizada en demostrar la factibilidad técnica de lo que se propone. Esto quiere decir que solo se encuentra a continuación una posibilidad para adaptar una o varias páginas,

[83] https://www.w3schools.com/jsref/met_document_queryselector.asp

sin necesariamente revertir la adaptación ante cambios en el contexto o suscribirse a los cambios en el contexto. Scripts de esa complejidad serán presentados como parte del enfoque, mientras que aquí se hace foco la posibilidad de construcción de una de características concretas, teniendo en cuenta que diferentes tipos de scripts tienen diferentes niveles de privilegios.

Generando una versión responsive de Blogger

Tal como se mostró en la Sección 4.1.1, Blogger no cuenta con una versión responsive y, para obtenerla, se llevó a cabo una adaptación que:

1. Agrega la especificación del *viewport*, que permite controlar las dimensiones y escala de una página,
2. Inicialmente oculta todos los elementos de la interfaz de la página, y
3. Luego, hace visibles solo de aquellos componentes que se mostrarán en la nueva versión acorde a las dimensiones del dispositivo, y
4. Finalmente, modifica el layout de los elementos que lo requieran (porque no se adaptan al ancho disponible en el dispositivo).

Para ello, se definió una subclase de *Augmenter*, llamada *ResponsivePage*, tal como se muestra en el Script 4. La misma establece un template method [54] de adaptación, *adapt*, el cual consiste básicamente en llevar a cabo los pasos anteriormente mencionados. El único de estos pasos implementados en la clase es el primero, mientras que los subsecuentes deben ser implementados en su totalidad por las subclases. Para ello, estas últimas pueden hacer uso los métodos restantes (protegidos): *changeIframeElementStyle* y *adaptWidthToTheRoot*.

```
function ResponsivePage(document){

    Augmenter.call(this, document);

    this.adapt = function(){
        this.addViewportMetaTag();
        this.hideAllDomElements();
        this.showTargetDomElements();
        this.adaptTargetDomElements();
    };
    this.addViewportMetaTag = function(){

        var metaTag = document.createElement('meta');
            metaTag.name = "viewport";
            metaTag.content = "width=device-width,\
                height=device-height,initial-scale=1.0";

        document.getElementsByTagName('head')[0].appendChild(metaTag);
    }
    this.changeIframeElementStyle = function(iframeSelector,
        elemSelector, style){

        var elem = document.querySelector(iframeSelector)
            .contentDocument.querySelector(elemSelector);

        if(elem)
            for(prop in style){
                elem.style[prop] = style[prop];
            }
    };
    this.adaptWidthToTheRoot = function(elem, width){

        while (elem.style) {
            elem.style.width = width;
            elem.style["max-width"] = width;
            elem = elem.parentNode;
        };
    };
}
```

Script 4. Clase abstracta de augmenter que genera una versión responsive de una página

Finalmente, la clase *ResponsiveBloggerEntry*, cuya implementación se muestra en el Script 5, es la encargada de adaptar específicamente la página de edición de entradas del blog. Su instanciación y ejecución se muestra en el Script 6, el cual fue ejecutado sobre el editor de una entrada de Blogger[84].

84 https://www.blogger.com/blogger.g?blogID=1392705246275834184#editor/target=post;postID=8007513119 213526564; consultada el 10 de Agosto de 2017

```
function ResponsiveBloggerEntry(document){
    ResponsivePage.call(this, document);

    this.showTargetDomElements = function(){

        this.showHeaderSection();
        this.showToolbar();
        this.showEditor();
    };
    this.adaptTargetDomElements = function(){
        this.adaptContentArea(this.getElement(".postsNew"), "65vh");
    };
    this.showHeaderSection = function(){
        this.showHeader();
        this.showTitle();
    };
    this.showHeader = function(){

        this.showElementsToTheRoot(this.getElement("div a"));
        this.getElement(".topHolder h2").style.display = "table";
        this.getElement(".topHolder h2 a").style.display = "table-row";
        this.showElementsToTheRoot(
            this.getElement(".topHolder h2 .blogg-title"));
    };
    this.showTitle = function(){
        this.showElementsToTheRoot(this.getElement(".topHolder input"));
    };
    this.showToolbar = function(){
        var controlsSelectors = ["#\\+fontSize", "#\\+formatBlock",
            "#\\+bold", "#\\+italic", "#\\+foreColor",
            "#\\+insertOrderedList", "#\\+insertUnorderedList",
            "#postingComposeBox"];

        var me = this;
        controlsSelectors.forEach(selector => {
            var elem = me.getElement(selector);
            me.showElementsToTheRoot(elem);
            me.showElementsToTheLeaf(elem);
        });
    };
    this.adaptContentArea = function(elem, width){
        this.adaptWidthToTheRoot(elem, width);
        this.adaptEditorHolder(".editorHolder");
        this.changeIframeElementStyle("#postingComposeBox",
            "#postingComposeBox", {width: "95%"});
    };
    this.showEditor = function(){
        this.getElement(".editor").style.position = "fixed";
    };
    this.adaptEditorHolder = function(selector){
        var holder = this.getElement(selector);
            holder.style.width = "100%";
            holder.style.top = "40px";
    };
}
```

Script 5. Generando una versión responsive de Blogger

```
1   new ResponsiveBloggerEntry(Components.utils.waiveXrays(
2       window.BrowserApp.selectedBrowser.contentWindow
3   ).document).adapt();
```

Script 6. Instanciación y ejecución del augmenter ResponsiveBloggerEntry

Adaptando la vista landscape de una noticia en Página 12

En la línea 43 puede observarse la instanciación de la clase que implementa este comportamiento, seguido del envío del mensaje «adapt». Este mensaje debe ser llamado por un *listener* al evento «deviceorientation» del objeto «window», tal como se hace convencionalmente para una aplicación Web, pero para simplificar, puede ser ejecutado directamente como se encuentra especificado mediante la consola remota del navegador[85]. Tal como se puede observar, el script ocultará primero todos los elementos visibles del DOM; primero recuperándolos mediante «getAllDomElements» y luego cambiándoles su visibilidad mediante la propiedad de estilo asociada. En este punto es recomendable almacenar el valor precedente, como una propiedad extra del elemento, para poder restaurarlo cuando sea necesario, pero por simplicidad solo se muestra cómo ocultarlo. Luego, llama a «showElementsToTheRoot», que obtendrá el elemento del DOM a partir de un query selector para luego enviarlo como argumento a los mensajes encargados de hacerlo visible y darle un posicionamiento apropiado. Es preciso aclarar que para acceder a los elementos del DOM es preciso recuperar una instancia de window «desprotegida», tal como se muestra en la línea 18, lo cual permite manipular el DOM sin las restricciones que convencionalmente aplican a los scripts de terceros (por ejemplo, la política del mismo origen).

[85] https://developer.mozilla.org/en-US/docs/Tools/Remote_Debugging/Debugging_Firefox_for_Android_with_WebIDE consultada el 10 de Agosto de 2017

```
function FullLandscapeElement(document, selector){

    Augmenter.call(this, document);
    this.adapt = function(){
        this.hideAllDomElements();
        var elem = this.getElement(selector);
        this.showElementsToTheRoot(elem);
        this.setFullModeStyle(elem);
    };
    this.setFullModeStyle = function(elem){
        elem.style["position"] = "fixed";
        elem.style["top"] = "0";
        elem.style["right"] = "0";
        elem.style["bottom"] = "0";
        elem.style["left"] = "0";
    };
}

new FullLandscapeElement(
    Components.utils.waiveXrays(
        window.BrowserApp.selectedBrowser.contentWindow).document,
    ".video-embed-field-responsive-video"
).adapt();
```

Script 7. Adaptando Página 12 con un full-screen video al rotar el dispositivo

Adaptando el layout y el volumen de Radio Estación Sur

Tal como se presentó en la Sección 4.1.3, la página de reproducción del streaming de Radio Estación Sur[86] no presenta una versión responsive y, adicionalmente, se podría adaptar el volumen de reproducción del streaming. Para ello, es necesario definir, en el contexto de dicha página, las clases Augmenter (Script 3) y ResponsivePage (Script 4), como así también cargar el script «volumen-meter.js»[87] que facilita el procesamiento y detección del volumen detectado por el navegador. En cualquier caso, se requiere simplemente copiar el contenido de los scripts, pegarlos y ejecutarlos en la consola.

Luego, debe hacerse lo mismo con la clase «VolumeAutoControl». La misma resuelve la adaptación responsive enviando el mensaje «addViewportMetaTag», heredado de ResponsivePage y cuyo comportamiento ha sido previamente presentado en el Script 4.

Fundamentalmente, lo que se busca lograr con este script es detectar el nivel de presión sonora en el ambiente del usuario, para lo cual se ejecuta el método «loadSoundListener», y como respuesta a sus cambios ajustar el control del volumen del streaming reproducido por la radio, mediante «setRadioVolume». Para lo primero, es necesario obtener los permisos del usuario sobre la entrada de audio (mediante

[86] https://radioestacionsur.org/beta/radio-popup.php consultada el 10 de Agosto de 2017
[87] https://github.com/cwilso/volume-meter/blob/master/volume-meter.js commit «cc16b5f»

«mozGetUserMedia»), que dará acceso a su «source». Este último podrá ser manipulado mediante un «MediaStreamSource», necesario para la interpretación del «AudioContext» por un «AudioMeter». Este último es el encargado de observar el streaming y convertir la entrada de audio en un nivel de sonido detectado.

```
function VolumeAutoControl(document, selector){

    ResponsivePage.call(this, document);

    this.setRadioVolume = function(volume){
        document.querySelector("#jp_audio_0").volume = volume;
    };
    this.loadSoundListener = function(volume){
        var me = this, audioContext = new AudioContext();
        navigator.mozGetUserMedia({
            "audio": {}
        }, function gotStream(stream) {

            mediaStreamSource =
                audioContext.createMediaStreamSource(stream);
            meter = createAudioMeter(audioContext);
            mediaStreamSource.connect(meter);

            me.sensingInterval = setInterval(function(){

                if(meter.volume > 0.02)
                    me.setRadioVolume(0.9);
                else me.setRadioVolume(0.2);

            }, 2000);

        }, function didntGetStream() {});
    };
    this.adapt = function(){

        this.addViewportMetaTag();
        this.loadSoundListener();
    }
}

new VolumeAutoControl(
    Components.utils.waiveXrays( window.BrowserApp
        .selectedBrowser.contentWindow).document,
    "#jp_audio_0"
).adapt();
```

Script 8. Automatically controlling the audio volume

Resaltando las funciones próximas en cartelera

Este script muestra cómo adaptar la cartelera del sitio de Cinema La Plata, de modo que los horarios de las películas a proyectarse en el rango de una hora (a partir de la actual) sean resaltadas en el documento. Para ello, se inyecta una nueva clase de estilo en el documento (`injectHighlightingStyle`), que agregará un fondo blanco, borde amarillo redondeado y una fuente más grande de la normal a los elementos a los cuales sea aplicada. Posteriormente, se recuperan los `div` de las películas presentadas en el

DOM (`highlightCloserTimeMovies`) y, uno a uno, son analizados para resaltar los horarios por cada uno de los cines listados (`highlightByCinema`). El problema aquí es que todos los horarios están expresados como parte de un mismo párrafo, sin distinción entre elementos, por lo que es necesario primero procesar la cadena para identificar horarios (mediante una expresión regular en `getTimeLabels`), para luego reemplazar dichas ocurrencias por un conjunto de `span` con el mismo contenido, pero a los cuales puede asociárseles una clase de estilo (definida en `injectHighlightingStyle`) para resaltarlas del resto de horarios, tal como se lleva a cabo en `wrapTextWithASpan`.

```
1   function CloserTimesHighlighter(document, selector){
2       Augmenter.call(this, document);
3
4       this.adapt = function(){
5           this.injectHighlightingStyle(document);
6           this.highlightCloserTimeMovies(
                document.querySelectorAll(selector));
7       }
8       this.highlightCloserTimeMovies = function(movies) {
9           for(movie of movies){
10
this.highlightByCinema(movie.querySelectorAll("p"),
                    Date.now());
11          }
12      }
13      this.getTimeLabels = function(paragraph) {
14          return paragraph.textContent.match(/[0-9]*:[0-9]+/g);
15      }
16      this.getMillisecondsFromTime = function(label){
17          return new Date().setHours(label.substring(0,2),
                label.substring(5,5),0,0);
18      }
19      this.highlightByCinema = function(cinemas, targetTime) {
20
21          for(cinema of cinemas){
22              var timeLabels = this.getTimeLabels(cinema);
23              if(timeLabels)
24                  for(label of timeLabels){
25                      if(this.isCloseToCurrentTime(label,
targetTime))
26                          this.wrapTextWithASpan(label, cinema);
27                  };
28          };
29      }
30      this.isCloseToCurrentTime = function(label, targetTime){
31          return Math.abs(this.getMillisecondsFromTime(label) -
                targetTime) < 3600000
32      }
```

```
33      this.wrapTextWithASpan = function(text, container){
34          container.innerHTML = container.innerHTML.replace(
35              text, "<span class='highlighted'>"+text+"</span>");
36      }
37      this.injectHighlightingStyle = function(doc) {
38          var style = document.createElement('style');
39              style.type = 'text/css';
40              style.innerHTML = '.highlighted { \
41                  background: white; border-radius: 1em; \
42                  font-size: larger; border: 3px solid yellow; \
43              }';
44          doc.getElementsByTagName('head')[0].appendChild(style);
45      }
46  }
47
48  new CloserTimesHighlighter(
49      Components.utils.waiveXrays( window.BrowserApp.selectedBrowser
            .contentWindow).document, ".singlepost"
50  ).adapt();
```

Script 9. Resaltando los horarios de las próximas funciones en cartelera

El Script 9 fue ejecutado sobre la versión móvil de «Cartelera»[88] de «Cinema La Plata». El método `turnDark` debería ser llamado ante vaores menores a 30 luxes.

Invirtiendo el esquema de colores de acuerdo con el nivel de luminosidad

Con la finalidad de adaptar las páginas de clasificados de un sitio, se creó un Augmenter llamado `WhiteOnDark`, capaz de inyectar el estilo necesario para invertir los colores de todos los elementos del DOM. Para ello, basta con crear dinámicamente un estilo (`turnDark` e `injectStyle`) cuya regla sea invertir, mediante un filtro, el elemento principal «html» (y, por consecuencia, todo lo que el contenga).

[88] http://www.cinemalaplata.com/Cartelera.aspx consultada el 11 de Agosto de 2017

```
function WhiteOnDark(document){

    Augmenter.call(this);

    this.adapt = function(){
        this.turnDark();
    };
    this.turnDark = function(){
        this.injectStyle('html { filter: invert(100%); }',
            document);
    };
    this.injectStyle = function(css, doc){

        var style = doc.createElement('style');
            style.type = 'text/css';
            style.appendChild(doc.createTextNode(css));

        doc.getElementsByTagName('head')[0].appendChild(style);
    };
};

new WhiteOnDark(Components.utils.waiveXrays( window.BrowserApp
        .selectedBrowser.contentWindow).document).adapt();
```

Script 10. Aplicando un esquema de colores light-on-dark a los clasificados de El Día

Este script fue ejecutado sobre la versión móvil de «Avisos Clasificados»[89] de «El Día». El método `turnDark` debería ser llamado ante valores menores a 30 luxes.

Inyectando contenido externo en Wikipedia

Para este escenario es preciso suscribirse a los cambios en la posición del usuario, que pueden ocurrir de diferentes maneras. Una manera de hacerlo es mediante el método watchPosition[90] del objeto geolocation del navegador, al cual cualquier aplicación Web puede acceder. El desafío en este caso es recuperar contenido externo y poder inyectarlo en el DOM de un artículo de Wikipedia, tal como se muestra en el siguiente script.

[89] http://clasificados.eldia.com consultada el 14 de Agosto de 2017
[90] https://developer.mozilla.org/en-US/docs/Web/API/Geolocation/watchPosition

```
function ExternalInfoBox (document, extenralSource){
    Augmenter.call(this);

    this.adapt = function(){
        this.addDescriptionFromMuseum(extenralSource);
    };
    this.addDescriptionFromMuseum = function(source){
        var bodyContent = document.querySelector("#bodyContent");
            bodyContent.parentNode.insertBefore(this.createInfoBox(
                source), bodyContent);
    };
    this.createInfoBox = function(source){
        var infoBox = document.createElement("blockquote");
            infoBox.className = "citado";
            infoBox.innerHTML = this.getExternalContent(source);
        return infoBox;
    };
    this.getExternalContent = function(source){
        const req = new window.XMLHttpRequest();
            req.open('GET', source.url, false);
            req.send(null);
        return this.evaluateSelector(source.selector,
                    this.getParsedDocument(req.responseText));
    };
    this.getParsedDocument = function(responseText){
        return new window.DOMParser().parseFromString(
            responseText, "text/html");
    };
    this.evaluateSelector = function(selector, doc){
        return doc.querySelector(selector).textContent;
    };
};

new ExternalInfoBox(
    Components.utils.waiveXrays( window.BrowserApp.selectedBrowser
        .contentWindow).document,
    { url: "http://www.fcnym.unlp.edu.ar/museo/educativa/darwin
            /piezas/pieza1.html",
      selector: "p:nth-child(2)" }
).adapt();
```

Script 11. Augmenting Wikipedia with third-party content