

Ingeniería de artefactos de Aumentación Web basada en crowdsourcing



Autor: Diego Andrés Firmenich
Director: Dr. Gustavo Rossi
Co-Director: Dr. Sergio Firmenich

Tesis presentada para obtener el grado de Doctor en Ciencias Informáticas

Septiembre, 2017

Facultad de Informática

Universidad Nacional de La Plata

Agradecimientos

Alejandro Dolina suele decir, que el verdadero objetivo del poeta es tener oportunidad de agradecer. No es esto un poema, ni es el tesista poeta.. sí mientras anhelaba concluir este trabajo, solía motivarme pensando que algún día tendría oportunidad de escribir estas líneas. Quisiera agradecer, recursivamente, a Silvia Gordillo, a Gustavo Rossi y a Sergio Firmenich por haber hecho posible que este agradecimiento suceda. A Leandro Antonelli, Matías Rivero, Marco Winckler y Damiano Distante, por su colaboración en la realización de esta investigación. Agradezco a mis padres, Pepa y Guillermo, también a Diego, Carol y Sergio por su hospitalidad y al resto de mi familia. Especialmente, por su amor y compañía, a mi compañera María Rosa y a nuestro pequeño hijo Emilio, que supo que solía estar *con la tesis*, pero no sabemos qué imagina que ello significa..

A los alumnos y colegas de la UNPSJB, a los profesores y compañeros de la carrera, al LIFIA, a la UNLP y la educación pública en general.

*Todo lo que es,
puede ser de otra manera.*

D. Sztajnszrajber

Índice general

Índice general	1
Índice de figuras	5
Índice de tablas	8
1. Introducción	9
1.1. Contexto y motivación	9
1.2. Objetivos	14
1.3. Contribución	14
1.4. Estructura del documento	15
2. Aumentación Web y Comunidades de usuarios finales	17
2.1. Aumentación Web	17
2.1.1. Aumentación vs Adaptación	19
2.2. Software para Aumentaciones Web	24
2.2.1. Aumentación Web vía complementos de terceros	25
2.2.2. Aumentación Web por usuarios finales	26
2.3. Aumentación Web por usuarios finales: ¿en qué consisten las aumentaciones?	28
2.3.1. Alteraciones de contenido	29
2.3.2. Alteraciones de funcionalidad	29
2.3.3. Alteraciones de interfaz de usuario	30
2.3.4. Alteraciones de interacción	30
2.3.5. Cardinalidad de las alteraciones	31

2.4.	Comunidades de Aumentación Web por usuarios finales	31
2.4.1.	Greasyfork en la telaraña	32
2.4.2.	Dinámica de actualización y técnicas de referenciación	38
2.4.3.	Greasemonkey API y Librerías de terceros	43
3.	CrowdMock	47
3.1.	Perspectiva general	47
3.2.	Negociación de adaptaciones basadas en Aumentación Web	50
3.3.	Actividades y herramientas propuestas	53
3.3.1.	Herramientas	53
3.3.2.	Actividades	56
4.	Requerimientos de Aumentación Web	61
4.1.	Definición de requerimientos	62
4.2.	Productos y actividades	64
4.3.	Expresividad	67
4.3.1.	Requerimientos no funcionales	67
4.3.2.	Requerimientos funcionales	69
4.4.	Ejemplo de definición, refinamiento y priorización	72
4.5.	Herramientas para la definición de requerimientos	76
4.5.1.	MockPlug	77
4.5.2.	Widgets	80
4.5.3.	UserRequirements	85
4.5.4.	Integración MockPlug y UserRequirements	86
5.	Desarrollo y prueba de artefactos de aumentación	89
5.1.	Metamodelo y generación de código	90
5.2.	MockPlug para desarrolladores	92
5.3.	CrowdMock Framework	93
5.4.	Testeo y casos de prueba	95
5.4.1.	Derivación y ejecución de casos de prueba	97
5.4.2.	Integración	99
6.	Distribución y mantenimiento	101
6.1.	Distribución de artefactos de Aumentación Web	102

6.2.	Componentes de aplicación para la distribución de artefactos .	103
6.2.1.	Caso de estudio	104
6.3.	Mantenimiento por usuarios finales	107
6.3.1.	Mantenimiento de referencias DEOI	109
6.3.2.	Validación de nuevas referencias	111
7.	Validación	112
7.1.	Definición de requerimientos	112
7.1.1.	Protocolo	113
7.1.2.	Ejecución	118
7.1.3.	Análisis	121
7.1.4.	Validez	122
7.2.	Mantenimiento de referencias	123
7.2.1.	Protocolo	124
7.2.2.	Ejecución	129
7.2.3.	Análisis	132
7.2.4.	Validez	133
8.	Trabajos relacionados	135
8.1.	Personalización y adaptabilidad por usuarios finales	135
8.2.	Ingeniería de requerimientos y crowdsourcing	137
9.	Conclusiones	140
9.1.	Objetivos y contribuciones	141
9.1.1.	Requerimientos de Aumentación Web	142
9.1.2.	Construcción y prueba	143
9.1.3.	Distribución y mantenimiento	144
9.1.4.	Integración	145
9.2.	Trabajo futuro	146
	Bibliografía	149
	Anexos	159
	A. Publicaciones	160

B. Encuesta a usuarios de la web	162
C. Expresiones regulares	165
D. Librerías de terceros	167
E. Artefactos analizados	173

Índice de figuras

2.1. Artefacto de aumentación en GreasyFork.	33
2.2. Incorporación de desarrolladores y de artefactos.	34
2.3. Instalaciones de artefactos por trimestre.	35
2.4. Cantidad de artefactos por desarrollador.	36
2.5. Instalaciones y Valoraciones.	36
2.6. Artefactos y actualizaciones.	37
2.7. Actualizaciones y comentarios.	38
2.8. Métodos de referenciación identificados automáticamente.	40
2.9. Métodos de referenciación identificados manualmente.	41
2.10. Artefactos con cambios de referencias entre versiones detectadas de modo automatizado.	42
2.11. Artefactos con cambios de referencias entre versiones detectadas de manualmente.	43
2.12. Artefactos y tipos de solicitudes HTTP detectadas automáticamente.	45
2.13. Utilización API Greasemonkey e importaciones de terceros.	46
3.1. Visión general CrowdMock.	48
3.2. Adaptación negociada. Relaciones entre actores.	52
3.3. Actividades y herramientas.	54
4.1. Requerimiento MockPlug.	63
4.2. Requerimientos de usabilidad.	68
4.3. Nuevo panel de navegación.	70
4.4. Sobresaltado de artículos de un autor.	71

4.5.	Resultado de búsqueda aumentado con enlace “citado por x” .	72
4.6.	Resultado de búsqueda UI original.	72
4.7.	Primer definición del modelo MockPlug.	73
4.8.	Refinamiento definido por Jorge sobre DBLP.	74
4.9.	Refinamiento definido por Jorge sobre DBLP.(cont.)	75
4.10.	Evolución del requerimiento de Pedro.	76
4.11.	Paleta de componentes predefinidos en MockPlug.	78
4.12.	Efecto de inserción de componente UI en el DOM Tree.	79
4.13.	Menú contextual de un widget (componente UI).	80
4.14.	Editores de propiedades y acciones.	81
4.15.	Componentes UI colectados.	82
4.16.	Componentes UI colectados y removidos.	83
4.17.	Pestaña Pocket en MockPlug.	84
4.18.	Colectando componente UI en el Pocket.	84
4.19.	Detalles de un requerimiento en UserRequirements.	85
4.20.	Evolución de requerimiento.	87
4.21.	Integración MockPlug y UserRequirements.	88
5.1.	Metamodelo CrowdMock.	90
5.2.	XML Scheme y XML de un modelo MockPlug.	91
5.3.	Código fuente generado.	92
5.4.	MockPlug para desarrolladores.	94
5.5.	Código fuente generado basado en CMFW.	95
5.6.	Testeo de modelo MockPlug.	98
5.7.	Testeo de artefacto de aumentación que implementa un modelo MockPlug.	98
5.8.	Integración y Testeo en MockPlug.	100
5.9.	Integración y Testeo en MockPlug. (cont.)	100
6.1.	Resultado de búsqueda en DBLP.	104
6.2.	Artefactos implementados por un desarrollador.	105
6.3.	Inclusión AAP en el sitio web principal de DBLP.	106
6.4.	Artefacto certificado por el responsable del sitio web.	106
6.5.	Selector de artefactos de aumentación.	107
6.6.	ADRM: componente para el mantenimiento de referencias DEOI.110	

7.1.	Estadístico $W+$ calculado en R.	122
7.2.	Extensión de experimentación utilizada.	126
7.3.	Procedimiento experimentación mantenimiento.	129
7.4.	Ejecuciones fallidas (rojo) y exitosas (verde) por artefacto y día en términos relativos. (versión CrowdMock)	131
7.5.	Ejecuciones fallidas (rojo) y exitosas (verde) por artefacto y día en términos relativos. (versión Tradicional)	131
7.6.	Estadístico χ^2 calculado en R.	133
B.1.	Primera y segunda pregunta.	162
B.2.	Tercera y cuarta pregunta.	163
B.3.	Formulario entregado a los encuestados en formato impreso.	164

Índice de tablas

2.1. Intercambios en adaptaciones cerradas.	22
2.2. Intercambios en adaptaciones basadas en extensiones.	23
2.3. Intercambios en adaptaciones basadas en Aumentación Web.	23
2.4. Intercambios en adaptaciones basadas en Aumentación Web por Usuarios Finales.	24
2.5. Artefactos que utilizan API Greasemonkey.	44
4.1. Categorías de componentes UI predefinidos.	81
7.1. Distribución de requerimientos.	118
7.2. Distribución de técnicas tradicionales.	119
7.3. Dificultad percibida al especificar requerimientos.	120
7.4. Nivel de satisfacción percibido por técnica.	120
7.5. Nivel de satisfacción percibido agrupado por pares.	120
7.6. Cantidad de registros de utilización de artefactos.	130
7.7. Resultado de ejecuciones por artefacto.	130
7.8. Resultado de ejecuciones por técnica de mantenimiento.	131
7.9. Tabla de contingencia fases 1 y 2.	132

Capítulo 1

Introducción

1.1. Contexto y motivación

La web es increíblemente diversa y es utilizada por usuarios con distintos tipos de conocimientos, culturas y habilidades; con distintas necesidades de accesibilidad y a través de una gran variedad de dispositivos con distintas capacidades. A su vez, el constante avance de las tecnologías y los estándares utilizados en la web, configuran un gran universo de posibilidades para la construcción de aplicaciones en las que transcurren las experiencias de los usuarios, donde conviven desde los más actualizados contenidos y aplicaciones hasta los más antiguos. Los usuarios de la web, transcurren por esta gran diversidad navegando y utilizando aplicaciones a la vez de aportar su propio contenido y sus propias particularidades en relación a sus redes, sus necesidades, sus preferencias, sus dispositivos y sus plataformas. Al intentar dimensionar esta enorme diversidad en su conjunto, donde conviven mil millones de sitios y miles de millones de usuarios, sobre este espacio virtual en constante movimiento y evolución, no puede dejar de advertirse que la web, es simplemente grandiosa. Como suele decirse: *Un mundo, una web*.

Sin embargo, este segundo mundo virtual no deja de ser muy joven en términos relativos y su expansión y crecimiento no se detiene. La evolución de la web no cesa y progresivamente se acerca a una plataforma compu-

tacional en si misma con increíbles capacidades de predecir las intenciones y necesidades de los usuarios [2]. Su expansión y crecimiento, también se encuentra relacionado al empoderamiento real que los usuarios tienen sobre la web, en términos de poder modificarla, agrandarla y adaptarla según sus intereses y conveniencia. Toda esta gran diversidad de aplicaciones, contenidos, tecnologías, estándares, usuarios y dispositivos que la componen, da lugar a necesidades o mejoras de tan distinta naturaleza, que resulta imposible ofrecer o proveer aplicaciones y contenidos que satisfagan a todas ellas por parte de aquellos que las construyen, aportan y mantienen.

Así es que los usuarios de la web, a través del tiempo y las tecnologías, siempre tendrán la necesidad de adaptar a otra forma aquello que ya existe, y lo que exista hoy deberá ser adaptado a otras formas en el futuro. Debido a la naturaleza dinámica de estos requerimientos y al dinamismo del propio medio en el que estas necesidades transcurren, las necesidades de adaptación y personalización sobre las aplicaciones webs, no pueden ser previstas ni contempladas en su totalidad de modo tradicional [13] por las aplicaciones webs en sí mismas, ni por aquellos que las construyen profesionalmente.

En consecuencia a esta realidad y a que la web termina por suceder en el cliente, en los últimos años técnicas de *Client Side Adaptation* (CSA) también llamadas *Web Augmentation* (WA) [10, 21] han crecido en popularidad otorgando la posibilidad a comunidades de usuarios de adaptar las aplicaciones y contenidos webs existentes a sus propias necesidades.

Básicamente, la Aumentación Web, consiste en agregar, alterar o quitar elementos de la interfaz de usuario (esto es, los recursos que la aplicación servidor entrega al cliente luego de cada solicitud), utilizando artefactos de software que usualmente corren en el dispositivo del usuario, normalmente en el explorador. Los artefactos de aumentación son usados para adaptar contenido, funcionalidad y presentación de los sitios webs existentes, por lo que suele decirse que la Aumentación Web es a la web, lo que la Realidad Aumentada a la realidad [21] y ha demostrado su potencial ofreciendo soluciones en distintos escenarios de aumentación como ser, aumentaciones orientadas a las tareas del usuario [38]; aumentaciones orientadas a la accesibilidad [6, 56]; aumentaciones orientadas a dispositivos móviles [8, 9], etc.

Los artefactos de Aumentación Web, suelen ser construidos o bien como

extensiones del explorador o bien como scripts de usuario que son ejecutados a nivel explorador. Independientemente de ello, en materia de Aumentación Web, la masa de usuarios juega un rol fundamental. Son los usuarios quienes necesitan las aumentaciones y son ellos quienes las proveen y las mantienen *ad-hoc*, valiendo destacar por cierto, por si la diversidad no fuere suficiente, que estas aumentaciones no son otra cosa que una forma muy particular de software. Así es como gran parte de los artefactos de aumentación, suelen ser contruidos originalmente por un usuario en busca de satisfacer su propia necesidad modificando sitios webs existentes con contenido específico o funcionalidad que no era soportada, constituyéndose así como un resultado de actividades relativas al *End-user Programming* [47]. Sin embargo, al ser estos artefactos compartidos con otros usuarios a través de repositorios, existen productos de aumentación muy populares que cuentan con miles de instalaciones. Permittiendo esto afirmar que la Aumentación Web, es una técnica propicia para la construcción de artefactos, tanto en el marco de la programación de usuarios finales que persiguen su propio objetivo, como para la construcción de productos de software en términos profesionales para satisfacer las necesidades de otros individuos. Existe evidencia que muestra el éxito de esta técnica como una manera de adaptar sitios webs, desde repositorios públicos de artefactos de propósito general, a dominios muy específicos como química y biología [84].

Las particularidades de este tipo software, son inherentes a la propia naturaleza desacoplada de los artefactos de aumentación respecto de los sitios o aplicaciones web aumentadas. Ello genera por consecuencia, altas probabilidades de obsolescencia cuando éstos últimos cambian de forma, estructura o contenido.

Las comunidades de usuarios que han surgido en el ámbito de la Aumentación Web, no son ajenas a la diversidad descripta. Conviven allí usuarios con distintos intereses, necesidades y con distintas habilidades en relación a la construcción de artefactos de aumentación. Es muy difícil de determinar con precisión cuantos usuarios componen activamente estas comunidades. Pero puede afirmarse que estas técnicas están creciendo en popularidad, al observarse que a febrero del año 2013, existían miles de artefactos de aumentación y los más populares cuentan ya con cientos de miles de descargas.

Sin embargo, las posibilidades que la Aumentación Web ofrece no han sido del todo descubiertas aún en relación a su potencial. Los artefactos de aumentación suelen ser compartidos a través de repositorios completamente desacoplados de los sitios webs, dificultando ello su disseminación y descubrimiento entre otros posibles usuarios interesados que navegan aquellos sitios que fueron aumentados. Por otro lado, los dueños o responsables de los sitios webs, no tienen ningún tipo de control o conocimiento sobre las aumentaciones que los usuarios ejecutan sobre sus contenidos.

Si bien han surgido distintos enfoques y herramientas afines del tipo WYSIWYG (*What You See Is What You Get*) como *WebMakeUp* [22] y *Platypus*¹, el nivel de expresividad que puede alcanzarse con este tipo de enfoques no resulta suficiente para gran parte de los requerimientos de aumentación, donde habilidades de programación son requeridas para lograr su implementación y donde en muchos casos, los artefactos cuentan de miles de líneas de código y lógica compleja que no puede ser expresada con este tipo de herramientas.

Las técnicas y herramientas utilizadas en la construcción de estos artefactos por parte de los usuarios, propias de actividades de programación por usuarios finales, careciendo de un proceso de construcción ingenieril, dificultan las posibilidades de colaboración entre aquellos usuarios que, con distintas habilidades, intentan afrontar las dificultades inherentes al objetivo que los convoca a compartir los artefactos. En consecuencia, pueden encontrarse en estos repositorios muchos artefactos de aumentación desmantenidos, de mala calidad, obsoletos, o simplemente ineficaces en su objetivo.

Por otro lado, si bien el término *crowdsourcing* [42] data del año 2006, muchos trabajos y definiciones han surgido posteriormente. En la búsqueda de una definición integrada [28], se define crowdsourcing como “*Un tipo de actividad participativa, en la cual un individuo, organización o compañía, propone a un grupo de individuos heterogéneo, variable en conocimiento y cantidad, mediante una convocatoria abierta, la realización de una tarea voluntariamente. La tarea a realizar, variable en complejidad y modularidad y en la cual la masa debe participar aportando su trabajo, conocimiento y/o experiencia, siempre supondrá un mutuo beneficio..*”.

¹<https://addons.mozilla.org/es/firefox/addon/platypus/>

En relación a la ingeniería de software, el estudio y la utilización de crowdsourcing en distintas actividades de sus procesos, ha marcado una tendencia en crecimiento, como puede observarse en [54]. En su definición más aceptada, *Crowdsourced Software Engineering* ha sido definido como el “*acto de delegar cualquiera de sus tareas a un grupo indeterminado y potencialmente amplio de trabajadores online en una convocatoria abierta*”. Numerosos y diversos estudios existen acerca de cómo los procesos de la ingeniería del software, pueden verse beneficiados por la incorporación del crowdsourcing en distintas actividades propias de sus procesos [54].

Las posibilidades que ofrece el crowdsourcing en materia de elicitación, desarrollo y testeo de adaptaciones no han sido descubiertas por las comunidades de Aumentación Web en su totalidad, siendo esta una tendencia en otros ámbitos de aplicación con características similares en cuanto a la masa de usuarios involucrada. Existen trabajos afines, en dónde la masa de usuarios es convocada a colaborar por parte de los desarrolladores de aplicaciones webs, en la adaptación de interfaces de usuario en términos de interfaces de usuario responsivas [60], como así también en la definición de nuevos componentes de aplicación en términos más generales [59].

Las comunidades dedicadas a la generación de artefactos de Aumentación Web, configuran un escenario propicio, desde su propia naturaleza, para la aplicación de nuevas estrategias de construcción de estos artefactos basadas en crowdsourcing, en donde distintas actividades relativas a la elicitación, definición, construcción, testeo y mantenimiento de los artefactos, puedan ser delegadas a la masa de usuarios interesada por parte de la misma masa de usuarios y donde ésta, pueda descubrir y compartir sus necesidades y verse involucrada activamente en ello, independientemente de las habilidades particulares de cada uno de sus individuos y en el marco de un proceso de integración con aquellos sitios webs que resultaren aumentados.

En esta tesis doctoral, se parte del estudio de las técnicas de Aumentación Web utilizadas por usuarios finales y de una de sus comunidades de relevancia, y se propone un enfoque ingenieril para la construcción de artefactos de Aumentación Web, basado fundamentalmente en el crowdsourcing de sus actividades y utilizando técnicas propias de otras áreas del conocimiento de nuestra ciencia.

1.2. Objetivos

El objetivo general de esta tesis doctoral es *investigar y generar los métodos de desarrollo y las arquitecturas de software necesarias, para facilitar la creación en colaboración mediante crowdsourcing, de artefactos de Aumentación Web por parte de usuarios finales, que se acoplen a las aplicaciones webs existentes, facilitando su definición, su diseminación y su mantenimiento a la masa de usuarios.*

Este objetivo puede descomponerse en los siguientes objetivos específicos/particulares:

1. Diseñar e implementar métodos de relevamiento de requerimientos de artefactos de Aumentación Web por parte de la masa de usuarios.
2. Diseñar e implementar métodos de construcción y prueba artefactos de aumentación a través de crowdsourcing.
3. Diseñar e implementar métodos de diseminación y mantenimiento de artefactos de aumentación a través de crowdsourcing.
4. Obtener un framework para la generación de artefactos de Aumentación Web que implemente un workflow basado en los métodos obtenidos.

1.3. Contribución

La contribución de esta tesis doctoral consiste en:

- Un enfoque ingenieril para la construcción de artefactos de Aumentación Web basado en crowdsourcing.
- Herramientas experimentales para la definición de requerimientos de Aumentación Web.

- Herramientas experimentales para la evolución de requerimientos de Aumentación Web.
- Un framework experimental y extensible para el desarrollado, testeo, distribución y mantenimiento de artefactos de Aumentación Web.
- Una serie de experimentos y correspondiente análisis de resultados en la utilización de las herramientas desarrolladas.

1.4. Estructura del documento

En esta sección se describe la estructura de esta tesis doctoral, presentando una breve reseña de cada uno de los capítulos subsiguientes.

Capítulo 2: Este capítulo describe en mayor profundidad las técnicas para la construcción de artefactos de Aumentación Web, presentando un estudio detallado de una de las comunidades más relevantes de artefactos de Aumentación Web por usuarios finales.

Capítulo 3: Presenta el enfoque propuesto, al que se ha denominado CrowdMock. Desde una perspectiva general y simplificada, se presenta el proceso propuesto distinguiendo sus actividades más relevantes y presentando las herramientas que dan soporte a su realización por parte de los distintos usuarios.

Capítulo 4: La definición de los requerimientos de Aumentación Web por parte de la masa de usuarios es uno de los pilares fundamentales de este trabajo. En este capítulo se presentan en detalle los aspectos relacionados a la definición de requerimientos de artefactos de Aumentación Web, como así también las herramientas experimentales implementadas que dan soporte a esta tarea.

Capítulo 5: Las actividades relacionadas con la construcción de los artefactos y sus correspondientes pruebas, son llevadas a delante desde versiones inicialmente incompletas, tanto de los artefactos, como de los casos de prueba que deben superarse. En este capítulo se describen en mayor detalle estas

actividades y las herramientas experimentales construidas para dar soporte a su realización.

Capítulo 6: Este capítulo describe cómo es realizada la distribución de los artefactos de Aumentación Web entre los usuarios finales de los sitios webs aumentados y cómo la masa de usuarios finales interesada es involucrada en su mantenimiento.

Capítulo 7: Durante el transcurso de este estudio, fueron realizados dos experimentos con el objetivo de validar la eficacia del enfoque propuesto en los aspectos que han sido considerados más novedosos. En este capítulo se presentan los protocolos y los resultados de las experimentaciones realizadas.

Capítulo 8: En este capítulo se seleccionan y analizan trabajos relacionados, primero desde el punto de vista de la programación por usuarios finales y la adaptación de los sistemas webs tradicionales. Luego se analizan trabajos relacionados tanto en relación a la definición de requerimientos como en la utilización de crowdsourcing en sus propuestas.

Capítulo 9: Finalmente, en este capítulo, se presentan las conclusiones de este trabajo, analizando y repasando los objetivos planteados y las contribuciones alcanzadas, como así también se plantea una serie de posibles trabajos futuros.

Capítulo 2

Aumentación Web y Comunidades de usuarios finales

En el capítulo introductorio se describió superficialmente en qué consiste la Aumentación Web. En este capítulo, se describe en mayor detalle esta técnica en función de presentar con mayor claridad el dominio alcanzado en relación a ello por esta tesis doctoral.

En la sección 2.1 se presenta y define Aumentación Web. En la sección 2.2 se describen las dos técnicas más importantes para la construcción de artefactos de Aumentación Web. En la sección 2.3 se presenta un análisis detallado acerca de qué tipo de alteraciones realizan los artefactos de aumentación construidos por usuarios finales sobre los sitios webs aumentados. En la sección 2.4 se presenta un detallado estudio de uno de los repositorios de artefactos de aumentación por usuarios finales más populares.

2.1. Aumentación Web

Si bien el contenido que navegamos tiene su origen en los servidores

webs, cómo este contenido se muestra y funciona es consecuencia también de cómo es presentado por nuestros exploradores, ya que la web sucede en nuestras computadoras y dispositivos, es decir, en el cliente. Es aquí donde entra en escena, la Aumentación Web.

Desde su definición más amplia y temprana [10], *“una herramienta o artefacto, debe ser considerado un artefacto de Aumentación Web, si a través de su integración con un explorador web, un servidor proxy o un servidor web, agrega contenidos o controles no contenidos por las páginas web en sí mismas originalmente, con el propósito de organizar, asociar o estructurar contenido encontrado en la web.”*

Así es que, la Aumentación Web puede suceder en distintas etapas de la resolución de una solicitud por parte de un cliente. Es decir, puede suceder en un servidor web, en un servidor intermedio, o bien, puede suceder directamente en el explorador, una vez que la solicitud ya hubiere sido atendida. Por otro lado, la aumentación consiste en modificar el contenido original de las páginas webs, y una vez que la aumentación suceda, serán paginas webs aumentadas. Dicha modificación, puede realizarse básicamente de dos maneras, o bien alterando el documento hipermedia al momento de atender o procesar la solicitud en un servidor, o bien alterando su *DOM Tree* resultante a través de la interfaz *DOM*¹ (*Document object model*) una vez procesado en el explorador el documento solicitado. Siendo este último mecanismo ampliamente utilizado, vale destacar que la interfaz DOM es una interfaz neutral, independiente de cualquier plataforma o lenguaje.

En términos tradicionales, la adaptación de las interfaces de usuario (UI) en los sistemas hipermedia adaptativos, requiere la definición de: qué objetivos tiene la adaptación, qué será adaptado, qué eventos dispararán la adaptación y qué técnicas de programación serán utilizadas para realizar la adaptación [14]. La siguiente sección distingue las técnicas de Aumentación Web de las técnicas de adaptación tradicionales, desde el punto de vista de los actores involucrados.

¹<https://www.w3.org/DOM/>

2.1.1. Aumentación vs Adaptación

Por su definición, el objetivo de adaptación de un artefacto de Aumentación Web, es modificar cualquier aspecto de la UI en cliente, tanto en su contenido o estructura como en su comportamiento o funcionalidad. Estas adaptaciones buscan cambiar la forma en que los usuarios realizan tareas [38], o cambiar la estructura del contenido e incluso el contenido que es presentado [60] en busca de mejorar su accesibilidad, usabilidad, personalización, etc. Los eventos que disparan las adaptaciones realizadas por un artefacto de Aumentación Web, son los eventos definidos por la DOM API, una vez que el contenido hipermedia que ha de ser aumentado es procesado por el explorador.

El desarrollo, la distribución y el uso de las técnicas de adaptación tradicionales y de Aumentación Web, involucran en distinta medida, a distintos actores incluidos los dueños o creadores de los sitios webs (*site-owners*), a desarrolladores externos y a los usuarios finales. El termino *site-owners*, designa a los desarrolladores responsables de los sitios webs, que tienen control total sobre el sitio original. En oposición los desarrolladores externos, son quienes suelen colaborar en la construcción de los artefactos de adaptación y los usuarios finales, refiere a la población de usuarios que consume ese contenido web.

Han sido identificadas cuatro estrategias para la adaptación de aplicaciones webs involucrando a estos actores [35]:

1. Cerrada (*closed adaptation*): refiere a las técnicas de adaptación que son embebidas como parte del sitio web original y se encuentran bajo el exclusivo control de los *site-owners*. Este tipo de adaptaciones pueden ser realizadas tanto en el cliente, como en el servidor. En cualquier caso, los desarrolladores del sitio web no tienen ninguna limitación ni restricción para acceder al código fuente del sitio web. Las adaptaciones pueden realizarse considerando las distintas características de los usuarios que pueden ser colectadas a través de perfiles de usuarios explícitamente (por ej. haciendo uso de formularios), o implícitamente a través de un seguimiento del comportamiento de los visitantes [48]. En este tipo de adaptaciones, las modificaciones sobre el sitio web son consecuencia de una relación directa entre los usuarios finales y los *site-owners*. Los

usuarios finales contribuyen con información que puede ser útil para personalizar las adaptaciones, y solo utilizan aquellas adaptaciones que los site-owners han predefinido. Los sistemas de recomendaciones y de filtrado colaborativo pertenecen a esta categoría [1].

2. Basada en extensiones (*Extension-based customization*): este tipo de adaptaciones es obtenida en base a la utilización de APIs específicas que permiten la construcción de artefactos por parte desarrolladores externos. Esta estrategia de adaptación es bien conocida en aplicaciones como *Google Drive*, *GMail*, *Facebook*, etc. A través del uso de APIs, los desarrolladores externos pueden crear nuevas formas de adaptación que no hubieran sido consideradas por los site-owners. En general, las adaptaciones realizadas de esta manera, implican una triangulación entre qué es posible adaptar en el sitio web vía dichas APIs, qué pueden hacer los desarrolladores externos con dicha API y finalmente, las expectativas de los usuarios finales sobre dichas adaptaciones. Para poder crear estas adaptaciones se requieren habilidades de programación y un conocimiento profundo tanto del sitio web a ser adaptado como de las funcionalidades provistas por las APIs específicas. Los usuarios finales necesitan algún soporte que les permita navegar y descubrir las extensiones disponibles que pueden utilizar, que suele ser provisto por los site-owners.
3. Aumentación Web (*Web Augmentation*): La Aumentación Web permite la creación de artefactos de adaptación por parte de terceras partes en las aplicaciones webs tradicionales, en el cliente y sin la intervención ni autorización de los site-owners de los sitios webs adaptados. Los complementos para exploradores webs son, por antonomasia, los artefactos de Aumentación Web que una vez instalados aumentan los sitios webs visitados por sus usuarios. La relación entre los actores se encuentra simplificada, ya que los site-owners son excluidos y no participan del proceso de adaptación. Los usuarios finales descubren las extensiones disponibles navegando mercados de complementos que los fabricantes de los exploradores hacen disponibles para tal fin.
4. Aumentación Web por usuarios finales (*End-user Web Augmentation*):

La Aumentación Web por usuarios finales se basa en la Aumentación Web, para empoderar a los usuarios finales con herramientas [24, 37] que les permitan crear y ejecutar sus propios artefactos de aumentación sobre los sitios webs de su interés. Pudiendo utilizar distintas herramientas para lograr su objetivos, la idea subyacente, es que los usuarios finales con pocas habilidades de programación, puedan canalizar sus necesidades de adaptación hacia la construcción de artefactos de aumentación basada en la utilización de herramientas creadas con ese objetivo.

Las estrategias para lograr la adaptabilidad de los sitios y aplicaciones webs en consideración de las distintas preferencias de sus usuarios, no son excluyentes entre sí, siendo posible que distintos usuarios finales para el mismo sitio web, terminen por participar en la ejecución de procesos de adaptación y de aumentación pertenecientes a alguna, varias o incluso a todas las estrategias antes mencionadas.

Así mismo, estas estrategias establecen distintas relaciones entre los actores involucrados. La Tabla 2.1 muestra como los dueños de los sitios webs y los usuarios finales son involucrados en la estrategia de adaptación cerrada. Mientras los usuarios suelen tener que proveer información personal, estas estrategias de adaptación no consideran la posibilidad de que terceras partes puedan colaborar y todos los costos del desarrollo de las adaptaciones recae sobre los site-owners. Sin embargo, los site-owners, tienen control total sobre las adaptaciones que los usuarios finalmente pueden utilizar.

Al exponer una API a la comunidad, los dueños de los sitios webs permiten a terceras partes colaborar y contribuir generando las adaptaciones de su propio interés. Como muestra la Tabla 2.2, construir y mantener una API, implica sus propios costos a los site-owners, pero en esta estrategia de adaptación ello se espera ver compensado por una contribución de la comunidad de desarrolladores que la utilizan, en la construcción de adaptaciones, que terminaran por utilizar los usuarios finales. Esta estrategia extiende las posibilidades de adaptación a terceras partes, pero sosteniendo cierto control sobre lo que se puede y no se puede hacer, a través del diseño de la API por parte de los site-owners. Mientras los usuarios finales, no necesariamente podrán satisfacer todas sus necesidades de personalización, las terceras par-

Tabla 2.1: Intercambios en adaptaciones cerradas.

Site-owner	Usuario final
<ul style="list-style-type: none"> - Costo de Implementación de modelo de adaptabilidad. + Los usuarios proveen información personal. + Control total de los mecanismos de adaptabilidad propuestos. - Mecanismo de adaptación depende de cada sitio. 	<ul style="list-style-type: none"> + Los usuarios no necesitan instalar nada. + Personalización vía sistemas de recomendación y filtrado colaborativo. - Los usuarios suelen tener que crear una cuenta y un perfil para obtener los beneficios de la adaptación. - Requerir perfiles de usuario puede generar problemas de privacidad. + Las adaptaciones pueden ser personalizadas para las tareas usuales con el sitio. - Pueden existir requerimientos de adaptación que no puedan satisfacerse.

tes podrán contar con el sitio web y su API para construir sus aplicaciones, en lugar de tener que empezar desde cero, alcanzando el universo de usuarios que ya utilizaba el sitio web adaptado. En el caso de las estrategias de adaptación basadas en Aumentación Web, son los dueños de los sitios webs quienes son excluidos de este proceso. Como muestran la Tablas 2.3 y la Tabla 2.4, la relación ocurre entre los desarrolladores de los artefactos de aumentación, es decir, terceras partes y los usuarios finales. Los primeros, como proveedores de artefactos de aumentación y los segundos como consumidores de dichos artefactos. Los site-owners no tienen ningún tipo de control ni información sobre como los usuarios finales terminan por utilizar los sitios webs adaptados de esta manera.

En relación a las técnicas de programación utilizadas para llevar a cabo estas adaptaciones en el marco de la Aumentación Web, existen dos técnicas ampliamente utilizadas vinculadas a las últimas dos estrategias mencionadas: la construcción de complementos para exploradores y la construcción

Tabla 2.2: Intercambios en adaptaciones basadas en extensiones.

Site-owner	Desarrollador	Usuario final
<ul style="list-style-type: none"> - Costo de Implementación de APIs. + Control total sobre la funcionalidad ofrecida por las APIs. + Los artefactos creados por terceros pueden contribuir a la popularidad del sitio. - Perdida de control sobre las adaptaciones realizadas por terceros. 	<ul style="list-style-type: none"> + APIs, documentación y soporte gratuito, generalmente. + Los desarrolladores puede crear sus aplicaciones a partir de las existentes. - La creatividad de los desarrolladores puede encontrarse limitada por las APIs disponibles. 	<ul style="list-style-type: none"> + Los usuarios no necesitan instalar nada en el explorador. +/- No hay garantías de que las aplicaciones basadas en APIs sean gratuitas para los usuarios finales. + Permiten a los usuarios customizar los sitios webs, mediante la instalación de las extensiones. - Pueden existir requerimientos de adaptación que no sean satisfechos.

Tabla 2.3: Intercambios en adaptaciones basadas en Aumentación Web.

Desarrollador	Usuario final
<ul style="list-style-type: none"> - Las adaptaciones pueden dejar de funcionar cuando los sitios webs son actualizados. + Independencia de APIs específicas permite crear adaptaciones genéricas para muchos sitios. + La creatividad de los desarrolladores no se encuentra limitada por APIs específicas. + No hay limitaciones de que aspectos pueden ser adaptados en el cliente. - Artefactos desarrollados por diferentes desarrolladores pueden ser incompatibles entre sí. 	<ul style="list-style-type: none"> - Necesitan instalar extensiones en el explorador. + Cuentan con más alternativas de adaptación. + Permiten a los usuarios customizar los sitios webs, mediante la instalación de las extensiones. - Otros mecanismos de adaptación provistos de modo nativo pueden estropearse. - Como los artefactos no son certificados, pueden existir artefactos maliciosos. - Pueden existir requerimientos de adaptación que no sean satisfechos.

Tabla 2.4: Intercambios en adaptaciones basadas en Aumentación Web por Usuarios Finales.

Site-owner	Desarrollador	Usuario final
		<ul style="list-style-type: none"> + Los usuarios finales cuentan con herramientas para realizar las adaptaciones que ellos requieren. - Los usuarios finales necesitan desarrollar ciertas habilidades de programación. - Las adaptaciones se encuentran limitadas a las habilidades de los usuarios finales y a la expresividad de las herramientas disponibles.

de scripts de usuario por parte de usuarios finales. A continuación, ambas técnicas serán descritas en mayor profundidad.

2.2. Software para Aumentaciones Web

Los artefactos de aumentación suelen ser construidos en busca de diversos objetivos, como por ejemplo, accesibilidad, usabilidad, funcionalidad, personalización, integración, etc. En función de ello es que un artefacto de Aumentación Web, puede ser construido para aumentar una sola URL, varias URLs de un dominio, un dominio completo, e incluso, todos los sitios webs. En relación a como las aumentaciones pueden lograrse técnicamente, es decir independientemente de quién las construya y con qué objetivo específico, y descartando desde ya la posibilidad de desarrollar un explorador web en si mismo, existen dos maneras ampliamente utilizadas. La primera, a través del desarrollo de complementos o extensiones de terceros para exploradores y la segunda, a través del desarrollo de aumentaciones por usuarios finales. Ambos métodos son muy relevantes para este estudio y son descritos a continuación.

2.2.1. Aumentación Web vía complementos de terceros

Las aumentaciones en forma de complementos o extensiones para exploradores, son distribuidas a través de mercados de extensiones. Así es como entre *Mozilla Addons*² para *Mozilla Firefox* y *Chrome Web Store*³ para *Google Chrome*, pueden encontrarse miles de extensiones con muy diversos propósitos y objetivos. Los usuarios finales pueden descargarlas gratuitamente e instalarlas en sus exploradores o bien navegando los sitios webs de los mercados, o través de menús que los exploradores suelen tener integrados para la administración de estos complementos. Por su parte, el desarrollo de aumentaciones en forma de extensiones, requiere los conocimientos y habilidades propias de un desarrollador de software. Las APIs que los exploradores proveen para facilitar su extensibilidad, son complejas conceptual y técnicamente, requiriéndose la utilización de *SDKs (Software Development Kit)* específicos y guías de desarrollo propias y atentas a las particularidades de cada explorador [45, 52].

El desarrollador de este tipo de aumentaciones, tiene grandes posibilidades y muy pocas limitaciones a la hora de implementar su producto, ya que su artefacto de aumentación correrá a nivel explorador y podrá tener acceso al historial de navegación del usuario, a sus formularios guardados, podrá abrir nuevas ventanas y pestañas y podrá acceder y aumentar cualquier contenido que el usuario se encuentre navegando; podrá entre estas y otras cosas, enviar y recibir contenido a dominios distintos al aumentado. Como supo decir el filósofo Ortega y Gasset, *la moneda falsa circula sostenida sobre la buena*: a través de este tipo de software puede distribuirse *malware*. Existen diversos estudios a este respecto [5, 15, 41, 44] y en consecuencia, los mercados de distribución de complementos suelen ser muy rigurosos en el proceso de aceptación y publicación de un nuevo artefacto. El código fuente, suele ser analizado automáticamente por herramientas *online* al momento de su publicación en busca de la utilización de prácticas no admitidas por las políticas de estos mercados, y luego en conjunto con los reportes producidos

²<https://addons.mozilla.org/es/firefox/>

³<https://chrome.google.com/webstore/category/extensions>

por estas herramientas, el código fuente de los artefactos es analizado por un integrante de la comunidad al que se le asigna esa responsabilidad. Desde ya, el proceso de admisión como miembro revisor en estas comunidades tiene sus propias particularidades⁴. Las aumentaciones webs como complementos de exploradores son de gran relevancia para este estudio. Ya que es a través de esta forma de software que han sido implementadas parte de las herramientas experimentales que soportan el enfoque propuesto en esta tesis doctoral y que serán presentadas en capítulos posteriores.

2.2.2. Aumentación Web por usuarios finales

Hacia el año 2005 surgieron herramientas que resultaron ser muy importantes para la aumentación Web por parte de usuarios finales. *Chickenfoot* [7], una extensión para el explorador Web Mozilla Firefox, permitiría a los usuarios finales crear sus propias aumentaciones. Según cuenta su historia en [63], inspirado en ello surgiría *Greasemonkey* [62], otra extensión para el explorador Web Mozilla Firefox, que ganaría popularidad e inmediatamente desde aquellos años generaría expectativas en la comunidad científica [11,12,19,31]. Hoy en día, estas herramientas que permiten a los usuarios finales la ejecución de sus propios artefactos de aumentación, son llamadas *weavers* [21]. Los *weavers* suelen distribuirse como extensiones para los exploradores, como lo son *Greasemonkey*⁵ para Mozilla Firefox y *Tampermonkey*⁶ para Google Chrome. Sin embargo en Google Chrome no es necesario instalarlos, ya que el mismo explorador ha integrado dicha funcionalidad permitiéndole al usuario crear y ejecutar sus propios artefactos. El desarrollo de aumentaciones por usuarios finales, requiere menos conocimientos y habilidades por parte de quienes las construyen en comparación a los requeridos en la construcción de artefactos de aumentación como complementos de exploradores, ya que no se requiere utilizar las APIs ni disponer de las SDKs propias de los exploradores. El desarrollo de aumentaciones por usuarios finales, requiere el conocimiento de tecnologías webs como ser *ECMAScripting (JS)*, *HTML*,

⁴<https://wiki.mozilla.org/Add-ons/Reviewers/Applying>

⁵<https://addons.mozilla.org/en-US/firefox/addon/greasemonkey/>

⁶<http://tampermonkey.net/>

CSS, DOM API, etc. Al momento de definir su artefacto de aumentación, el usuario debe indicar cuál o cuáles son los sitios webs aumentados por su artefacto. Así, el weaver, al momento de que el explorador termina por procesar y presentar algún sitio web correspondiente a este conjunto de URLs, inserta en el DOM Tree resultante, el artefacto desarrollado por el usuario, que en términos técnicos es un script JS de extensión `.user.js`.

A su vez, los weavers, suelen proveer algunas funcionalidades a través de pequeñas APIs. El desarrollador de un artefacto de Aumentación Web de estas características, contará por ejemplo con la posibilidad de que su artefacto haga solicitudes asincrónicas a dominios distintos al aumentado. Estas solicitudes, serán resueltas por el weaver, que al ser una extensión para el explorador (o el explorador propiamente dicho), no encontrará limitaciones como las impuestas desde el mismo estándar para las solicitudes *cross-domain* (entre distintos dominios) en la ejecución de código JavaScript sobre un sitio en particular (*SOP*⁷, *same origin policy*).

Por su parte, los usuarios que construyen estos artefactos son libres de utilizar cualquier otra librería JS de terceros. Siendo muy popular la utilización de la librería *jQuery*⁸. Esto será descripto posteriormente en mayor profundidad.

Así como en el caso de las aumentaciones webs a través de complementos para exploradores donde existían mercados de extensiones, en el caso de las aumentaciones webs por usuarios finales, sus creadores suelen compartir sus aumentaciones con el resto de los usuarios a través de repositorios *online*. Del mismo modo en el que un usuario es capaz de instalar un complemento para un explorador web desde un mercado de extensiones, también puede instalar un artefacto de aumentación de usuario final desde un repositorio de scripts de usuarios. A diferencia de los mercados de extensiones para los exploradores, estas comunidades no suelen efectuar ningún tipo de revisión sobre los artefactos compartidos. Los artefactos son aceptados y publicados sin ningún tipo de revisión y los usuarios de la comunidad los descargan y utilizan bajo su propia responsabilidad, pudiendo los usuarios en algunas de estas comunidades, indicar que el artefacto es peligroso. Del mismo modo que

⁷https://www.w3.org/Security/wiki/Same-Origin_Policy/

⁸<https://jquery.com/>

fue descrito anteriormente en relación a los artefactos de aumentación en forma de complementos, esta forma de software y estas comunidades, también pueden ser una vía de generación y distribución de malware y existen también diversos estudios al respecto [80], [27].

Son estos tipos de artefactos, los creados por usuarios finales, el *objeto sujeto* de esta tesis doctoral. Existen distintas técnicas en relación a las tecnologías involucradas en la creación de estos artefactos por parte de los usuarios finales y alrededor de ellas se han formado distintas comunidades. En los siguientes apartados se describe en qué consisten estas aumentaciones y se documenta un estudio descriptivo sobre una de las comunidades de artefactos de Aumentación Web por usuarios finales de mayor relevancia, que ha surgido durante los años en los que se ha realizado este estudio.

2.3. Aumentación Web por usuarios finales: ¿en qué consisten las aumentaciones?

Para comprender qué tipo de alteraciones realizan estos artefactos creados por usuarios finales sobre los sitios webs aumentados, fue considerada la taxonomía propuesta por un estudio reciente [23] para el análisis de artefactos de aumentación como complementos para exploradores. Allí pudo ser establecido que las aumentaciones en forma de complementos son utilizadas principalmente para adaptar contenido, interfaces de usuario, navegación o funcionalidad, ya sea por agregar, remover, o quitar elementos en el sitio web aumentado. Se analizaron manualmente los 50⁹ artefactos de aumentación por usuarios finales más populares de uno de los repositorios más importantes en la actualidad (Anexo E). Cada uno de estos artefactos fue descargado, instalado y utilizado con el objetivo de poder clasificar las alteraciones que estos realizaran sobre los sitios webs aumentados, pudiendo un artefacto en particular, realizar más de una de las categorías de alteraciones que se detallan a continuación.

⁹Listado online: <https://goo.gl/aW465n>

2.3.1. Alteraciones de contenido

La mayoría de los artefactos analizados, aumentaban los sitios webs mediante alteraciones de su contenido. Dichas alteraciones de contenido fueron subcategorizadas de la siguiente manera:

- Filtrar/ocultar contenido existente: $2/50$ artefactos filtraban o ocultaban contenido de alguna manera. Por ejemplo, uno de estos artefactos, ocultaba los episodios de series que ya habían sido visualizados por el usuario.
- Remover contenido: $17/50$ artefactos removían contenido con el objetivo principal de quitar publicidades.
- Agregar contenido: $42/50$ artefactos agregaban contenido desde la misma aplicación web (o dominio). Por ejemplo, un artefacto de actualización para *YouTube*¹⁰, agregaba información a los resultados de búsqueda, obteniéndola de cada una de las páginas individuales de los videos listados en el resultado.
- Integrar contenido de otros sitios: $8/50$ artefactos integraron contenido proveniente de otros sitios webs, que resultaba obtenido dinámicamente (cada vez que el artefacto se ejecuta). Por ejemplo, para integrar el rating de *IMBD*¹¹, con otros sitios webs similares como *Filmaffinity*¹².

2.3.2. Alteraciones de funcionalidad

Varios de los artefactos analizados agregaban o cambiaban el comportamiento original del sitio web aumentado de algún modo.

- Agregar funcionalidad: $39/50$ artefactos agregaron algún tipo de comportamiento. Por ejemplo, un artefacto agregaba un minimapa en un

¹⁰<http://www.youtube.com>

¹¹<http://www.imdb.com>

¹²<http://www.filmaffinity.com>

juego llamado *Agar*¹³.

- Eliminar funcionalidad: ninguno de los artefactos analizados quitaba comportamiento existente.
- Modificar funcionalidad: $4/50$ artefactos cambiaban el comportamiento original de alguna manera. Por ejemplo, un artefacto cambiaba el funcionamiento del click secundario.

2.3.3. Alteraciones de interfaz de usuario

Varios de los artefactos analizados alteraban la UI original del sitio web aumentado, reordenando sus elementos de algún modo.

- Reordenar elementos de la interfaz: $14/40$ artefactos reordenaban los elementos de la interfaz de la aplicación web aumentada de alguna manera.

2.3.4. Alteraciones de interacción

Varios de los artefactos analizados alteraban aspectos relativos a la interacción con la UI original del sitio web aumentado.

- Navegación: $14/50$ artefactos agregaron alguna forma de navegación. Por ejemplo, uno de los artefactos convertía las URLs existentes en forma de texto plano, en URLs en forma de enlaces.
- Interacciones de teclado: $7/50$ artefactos agregaron atajos de teclado en la aplicación web aumentada. Por ejemplo alguno de ellos agregaba atajos de teclado sobre las páginas de reproducción de videos en YouTube.
- Interacciones de puntero: $5/50$ artefactos agregaban interacciones basadas eventos del *mouse*. Por ejemplo, agregando controladores para el

¹³<http://agar.io/>

evento *OnMouseOver*, de modo que cuando el usuario posara el puntero sobre una imagen, ésta sea mostrada automáticamente más grande en un diálogo modal.

2.3.5. Cardinalidad de las alteraciones

Las alteraciones clasificadas anteriormente, fueron aplicadas por los artefactos analizados a un conjunto de ítems de la misma naturaleza del sitio web aumentado, o bien a uno solo, o a varios de distinta naturaleza. Considerando esta distinción como la cardinalidad de la alteración, se distingue que:

- Conjunto de ítems: *11/50* artefactos alteraban un conjunto de ítems (de la misma naturaleza) del sitio web aumentado. Por ejemplo, un artefacto, aumentaba todas las imágenes de los resultados de búsqueda en *Google*¹⁴, con un enlace directo hacia ella.
- Un solo ítem, o varios (distinta naturaleza): *39/50* artefactos alteraban un solo ítem, o varios pero de distinta naturaleza sobre el sitio web aumentado. Por ejemplo, un artefacto modificaba la interfaz para la visualización de videos en YouTube y también agregaba información en la descripción del video.

2.4. Comunidades de Aumentación Web por usuarios finales

Distintas comunidades de usuarios han surgido en los últimos años alrededor de las técnicas y herramientas de aumentación más populares y también de otras técnicas relacionadas donde el usuario final de las aplicaciones webs, resuelve sus necesidades particulares siendo el protagonista fundamental de su desarrollo. Entre las comunidades más populares se destacan *Yahoo*

¹⁴<https://www.google.com>

pipes!, *UserStyles* y *UserScripts*. Existen estudios acerca del comportamiento de estas comunidades [75, 76], pero ninguno de ellos describe el comportamiento haciendo foco en las comunidades relacionadas con las herramientas y los usuarios del interés propio de esta tesis doctoral. Así como Yahoo pipes! alcanzó su fin hacia finales del año 2015, el repositorio de artefactos de aumentación de usuarios finales de interés de esta tesis que supo congregar a la mayor cantidad de usuarios, UserScripts, lamentablemente fue desmantenido durante el transcurso del año 2014. Ello dio lugar a que sus usuarios migren hacia nuevos repositorios conformando nuevas comunidades como *OpenUserJS*¹⁵ y *Greasyfork*¹⁶, siendo este último mantenido por las mismas personas que mantienen UserStyles.

En la subsección 2.4.1, se documentan los resultados obtenidos en el estudio de una de estas comunidades, considerándola una de las más importantes y representativas de hoy en día. Pretende generar un antecedente en función de comprender por qué un enfoque como el propuesto en este estudio, resulta relevante para este tipo de comunidades.

En la subsección 2.4.2, se documentan los resultados obtenidos en relación a como los artefactos de aumentación por usuarios finales hacen referencia a los sitios webs aumentados, y cómo y en qué medida los usuarios de los artefactos se involucran proveyendo valoraciones y comentarios a los usuarios con habilidades de programación que los crean y los mantienen.

Finalmente, en la En la subsección 2.4.3, se documentan los resultados obtenidos en relación a cómo los artefactos de aumentación por usuarios finales utilizan APIs y librerías de terceras partes.

2.4.1. Greasyfork en la telaraña

Greasyfork es una reciente comunidad de usuarios de Aumentación Web por usuarios finales, que abrió sus puertas online durante el año 2014. Desde aquel tiempo a esta parte su popularidad ha ido en aumento y en términos relativos, la cantidad de los artefactos allí publicados resulta considerable,

¹⁵<https://openuserjs.org/>

¹⁶<https://greasyfork.org/>

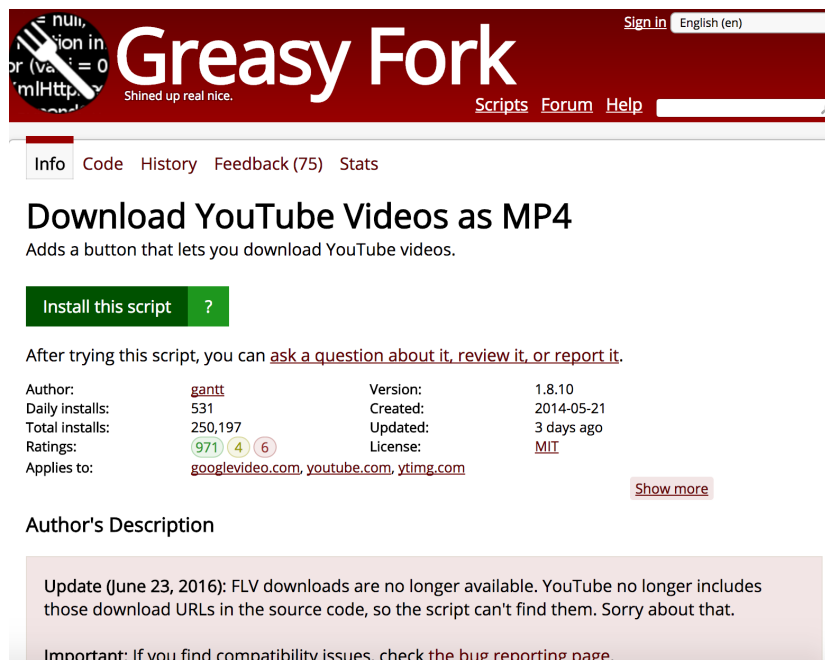


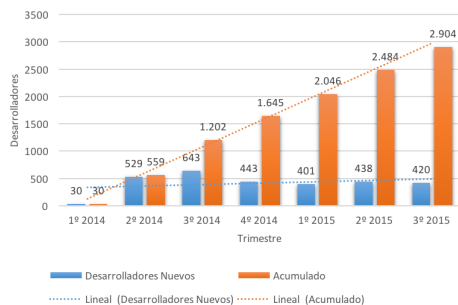
Figura 2.1: Artefacto de aumentación en GreasyFork.

encontrándose aproximadamente un total *14 mil* artefactos hacia fines del año 2016. Dicho repositorio fue descargado completamente durante el mes de octubre del año 2015. Esto es, los artefactos en sí mismos junto con todas sus actualizaciones y todas las páginas webs correspondientes a sus descripciones, valoraciones, etc. fueron descargadas. Con los documentos *offline* y haciendo uso de técnicas afines, como *webscraping* [57], fueron analizados los documentos HTML del repositorio que describían los artefactos de aumentación (Figura 2.1) y daban cuenta de la cantidad de instalaciones, usuarios desarrolladores, valoraciones, etc. y se logró construir y popular una base de datos con el fin de poder procesar y analizar descriptivamente distintos aspectos de relevancia para este estudio.

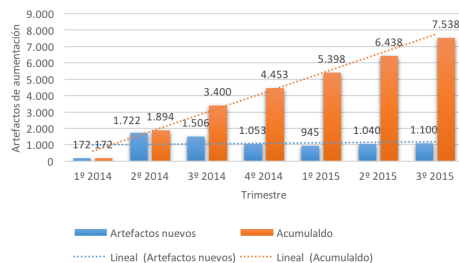
En total, al momento de realizar dicho relevamiento, Greasyfork congregaba *2.904* usuarios finales de Aumentación Web con habilidades de programación. Como puede apreciarse en la Figura 2.2a, la cantidad de usuarios con habilidades de programación sostuvo un aumento casi constante, incor-

porando alrededor de 500 desarrolladores nuevos trimestre a trimestre y habiendo casi duplicado los usuarios con habilidades de programación de un año al otro (notar 2015 incompleto).

También puede claramente apreciarse como una gran cantidad de usuarios con habilidades de programación fue incorporada al mismo tiempo que el repositorio históricamente más importante, userscripts.org, resultaba ser dado de baja, a comienzos del año 2014.



(a) Incorporación desarrolladores.



(b) Incorporación artefactos.

Figura 2.2: Incorporación de desarrolladores y de artefactos.

Estos usuarios finales con habilidades de programación, han construido y compartido en esta comunidad un total de 7.538 artefactos de aumentación al momento de realizar el relevamiento. Estos artefactos están compuestos por un total de 3.020.905 de líneas de código. Es notable como, la velocidad en el aumento de la cantidad de artefactos de aumentación disponibles de un trimestre al otro, acompaña la tasa de crecimiento de usuarios con habilidad de programación, Figura 2.2b. Del mismo modo que la cantidad de desarrolladores, de un año a otro, la cantidad de artefactos de aumentación disponibles y compartidos con la comunidad evidencia haber sido duplicada.

La cantidad de instalaciones de artefactos puede ser considerada como un indicador de uso e interés por parte de la comunidad en general. Involucrando a los usuarios con y sin habilidades de programación se han producido un total de 2.448.251 instalaciones de los artefactos de aumentación disponibles en el repositorio, Figura 2.3. La cantidad de instalaciones trimestre a trimestre, muestra una tasa de crecimiento inestable, y se hace evidente que los artefactos más antiguos, han tenido más oportunidad de ser descubiertos

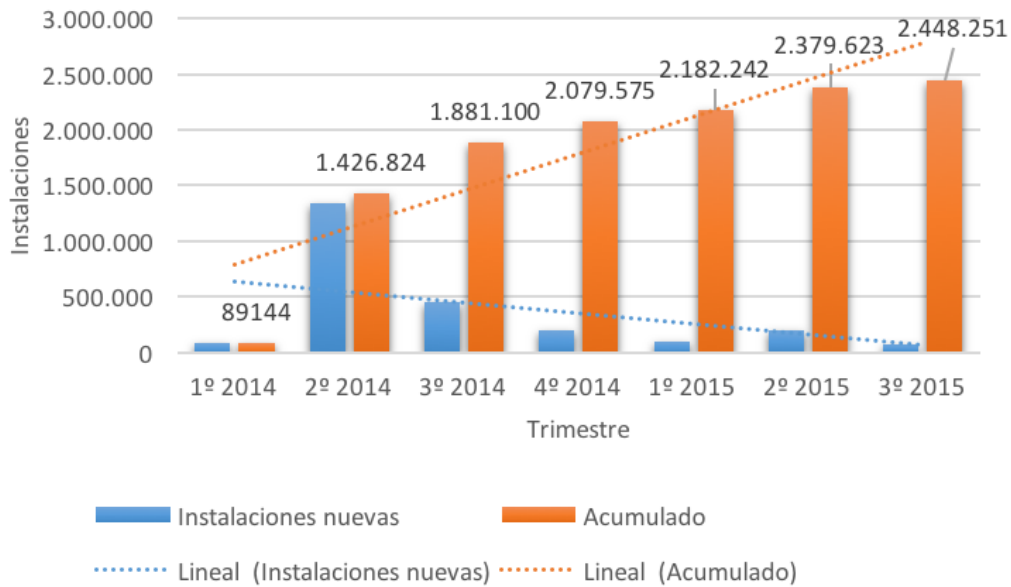


Figura 2.3: Instalaciones de artefactos por trimestre.

por el resto da la comunidad, por la simple razón, de permanecer disponibles más tiempo.

En relación a la productividad de los usuarios finales con habilidades de programación considerando la cantidad de artefactos que han publicado, Figura 2.4, la gran mayoría de ellos, precisamente el *78,92 %*, se aboca al desarrollo y al mantenimiento de a lo sumo *dos* artefactos de aumentación. En cuanto al nivel de interés y utilización que han generado los artefactos, puede apreciarse en la Figura 2.5a como un porcentaje significativo de éstos han resultado muy poco utilizados y que un porcentaje mucho más importante, han resultado ser de interés para una gran cantidad de usuarios de la comunidad.

En relación a las valoraciones por parte de la comunidad de los artefactos disponibles en el repositorio, Figura 2.5b, los usuarios que los instalan pueden valorarlos asignando un valor de la escala: *malo*, *aceptable*, *favorito*. Si bien ha sido detectada una escasa participación por parte de la masa de usuarios, encontrándose una gran cantidad de artefactos sin ninguna valora-

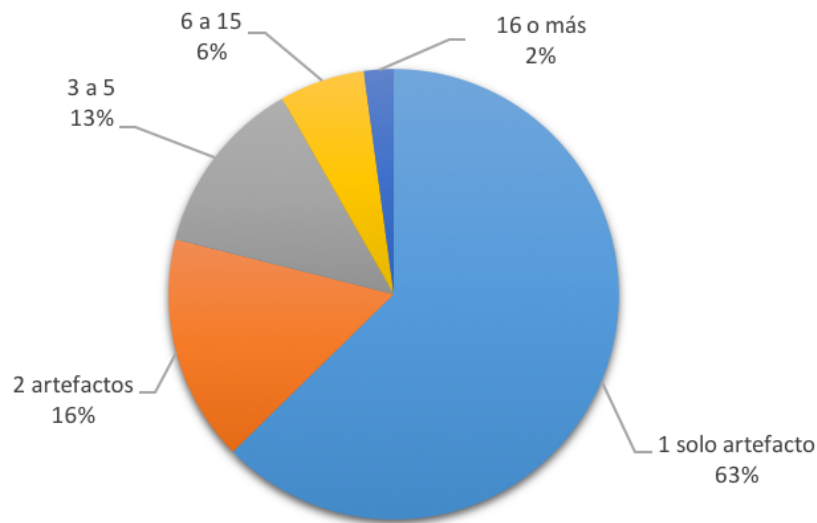
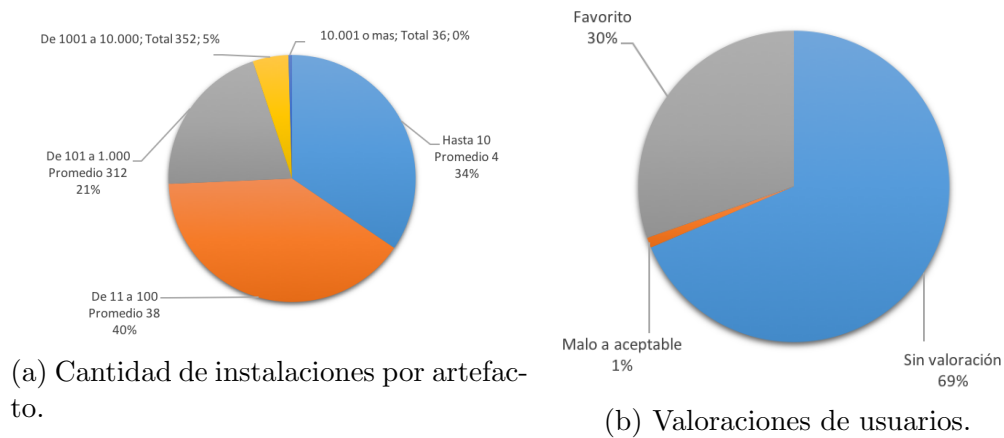


Figura 2.4: Cantidad de artefactos por desarrollador.



(a) Cantidad de instalaciones por artefacto.

(b) Valoraciones de usuarios.

Figura 2.5: Instalaciones y Valoraciones.

ción, puede apreciarse que los usuarios tienden a votar por positivo en lugar de hacerlo por negativo cuando lo hacen, siendo que la gran mayoría de las valoraciones registradas son en sentido, existiendo muy pocas valoraciones negativas en comparación.

Por último, ha sido analizada la cantidad de artefactos que resultaron ser actualizados con posterioridad a su publicación. Los motivos por los cua-

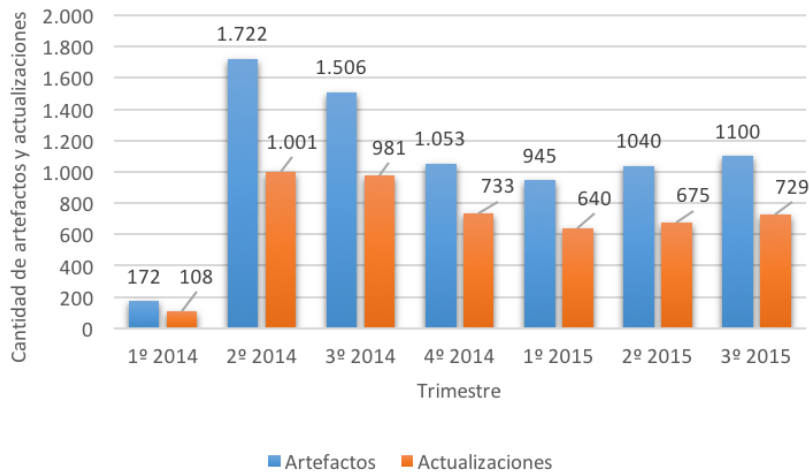


Figura 2.6: Artefactos y actualizaciones.

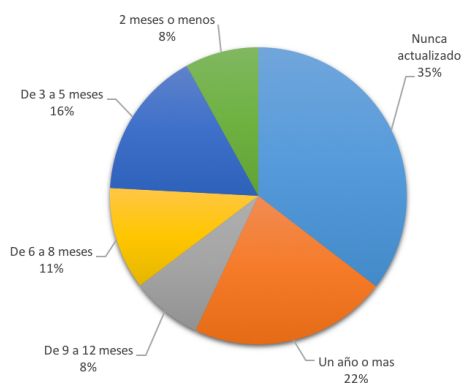
les un artefacto de aumentación necesite ser mantenido o actualizado en el repositorio son diversos. Los usuarios que los han compartido con el resto de la comunidad originalmente, tienen oportunidad de subir las actualizaciones cuando necesitan hacerlo. En el repositorio, se cuenta con una pequeña herramienta *online* que permite ver, a cualquier usuario interesado, las diferencias en el código fuente entre las distintas versiones del artefacto. Al momento del relevamiento, en el repositorio existen 33.303 actualizaciones correspondientes de 4.867 artefactos que fueron al menos una vez actualizados. Independientemente de que un 36% de los artefactos se encontraran en el mismo estado que en el que fueron originalmente publicados, en la Figura 2.6 se muestra como los usuarios con habilidades de programación, así como publican nuevos artefactos, también publican en menor medida, pero considerablemente, su actualizaciones.

De los datos compartidos anteriormente, se considera importante destacar que la comunidad bajo estudio sostiene un lento pero marcado crecimiento, tanto en cantidad de usuarios con habilidades de programación, como en cantidad de artefactos y de descargas. Sin embargo e independientemente del hecho concreto de la discontinuación de la comunidad más popular durante el año 2014, ninguna de estas tasas de crecimiento ha mostrado un salto distinguido que pueda ser atribuido a ninguna variable en particular.

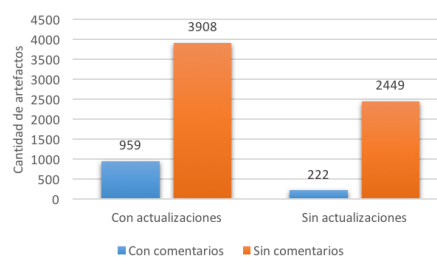
La naturaleza desintegrada de los artefactos de aumentación en relación a su instalación, condiciona la posibilidad de que los usuarios finales que los utilizan puedan proveer valoraciones a los usuarios que los construyen, donde a su vez se ha podido determinar, que en su gran mayoría, se encuentran abocados a su desarrollado y mantenimiento en soledad. En relación al mantenimiento de los artefactos, se distingue que un porcentaje muy considerable de éstos nunca fue actualizado con posterioridad a su fecha de publicación.

2.4.2. Dinámica de actualización y técnicas de referenciación

Al analizar la dinámica de las actualizaciones de modo de poder determinar la antigüedad de la última actualización provista para cada artefacto, al momento del relevamiento, solo el $24,16\%$ contaban al menos una actualización en los últimos cinco meses y un $56,95\%$ o bien no tenían actualizaciones o bien sus últimas actualizaciones tenían más de un año de antigüedad, como puede apreciarse en la Figura 2.7a).



(a) Antigüedad última actualización.



(b) Actualizaciones vs. comentarios.

Figura 2.7: Actualizaciones y comentarios.

Los usuarios sin habilidades de programación a través de comentarios colaboran con los desarrolladores. Si bien no fue posible verificar que la cantidad

de actualizaciones de los artefactos, se encuentre correlacionada directamente con la cantidad de comentarios, es posible considerar que los comentarios que los usuarios finales hacen, tienen un impacto positivo en el mantenimiento, siendo que la gran mayoría de los artefactos con comentarios tienen alguna actualización, Figura 2.7b.

Intentando determinar el modo en que los artefactos referencian a los sitios webs aumentados, se realizó un estudio completo de sus respectivos códigos fuentes en sus últimas versiones. Es necesario considerar, que los artefactos de aumentación en sus códigos fuentes, contienen instrucciones para seleccionar elementos del DOM Tree del sitio aumentado. Estas instrucciones son generalmente invocaciones a la DOM API. Por ejemplo, un artefacto puede hacer uso de la sentencia `document.getElementById('un_identificador')`; con el objetivo de seleccionar el nodo del DOM Tree que el sitio web aumentado ha identificado como `'un_identificador'`. En este ejemplo, y no siendo desde ya el único modo posible de seleccionar elementos del DOM Tree, la referencia al nodo `'un_identificador'` ha sido *harcodeada*¹⁷ (incrustada) en la sentencia.

Luego de procesar el texto de los 7.538 código fuentes correspondientes a los artefactos disponibles en el repositorio, haciendo uso de expresiones regulares (Anexo C), fue posible determinar que el 71,5% de los artefactos contienen este tipo de referencias harcodeadas detectables de manera automatizada. Las 68.925 referencias detectadas en estos artefactos, han sido categorizadas en función de la técnica/modo de referenciación utilizada/o (Figura 2.8), ya sea utilizando el identificador de los nodos (30%), por selectores CSS (11%), por selectores estilo jQuery (57%) o en menor medida con expresiones XPath (2%).

Del 29,5% restante de los artefactos (2.152) de los que no se pudo detectar referencias de modo automático, se seleccionó una muestra aleatoria de 200¹⁸ artefactos con el objetivo de analizar manualmente su código fuente en función de completar el estudio acerca de qué manera estos artefactos referencian el DOM, si es que lo hacen (Anexo E). En la Figura 2.9 puede apreciarse que fue determinado que el 61% de ellos no tenía ningún tipo de

¹⁷https://es.wikipedia.org/wiki/Hard_code

¹⁸Listado online: <https://goo.gl/st66hL>

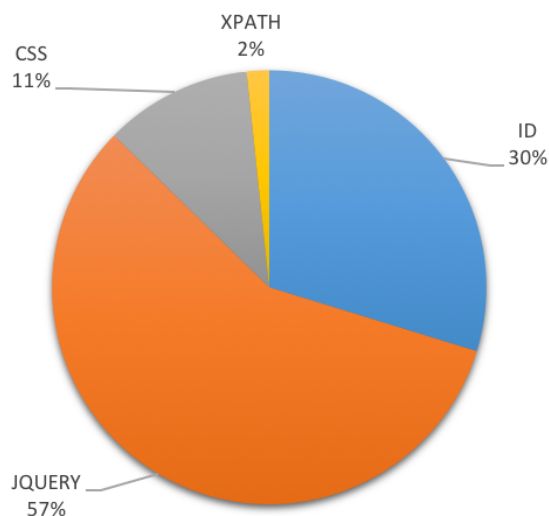


Figura 2.8: Métodos de referenciación identificados automáticamente.

referencia al DOM Tree aumentado, y que el 39 % restante si utiliza referencias de alguna u otra manera no detectada por las expresiones regulares diseñadas para tal fin.

El tipo de referenciación utilizado por este 39 % de artefactos, fue categorizado del mismo modo que anteriormente, determinando que el 12,5 % utilizan identificadores de los nodos, que el 11 % por selectores CSS, por selectores estilo jQuery 13,5 % o en menor medida con expresiones XPath 2 %.

En resumen, tanto del procesamiento automatizado que cubre 71,5 % del universo muestral, más el análisis manual realizado sobre una muestra al azar del 29,5 % restante de los artefactos, se concluye que más del 80 % de los artefactos contienen al menos algún tipo de referencia harcodeada al DOM Tree, siendo las referencias al estilo jQuery las más populares, seguidas por la selección de elementos a través de `document.getElementById`, luego por la utilización de selectores CSS a través de `document.querySelectorAll` y `document.querySelector` y finalmente podemos concluir que las referencias a través de xPaths invocando `document.evaluate`, en este tipo de artefactos son muy poco utilizadas por los desarrolladores.

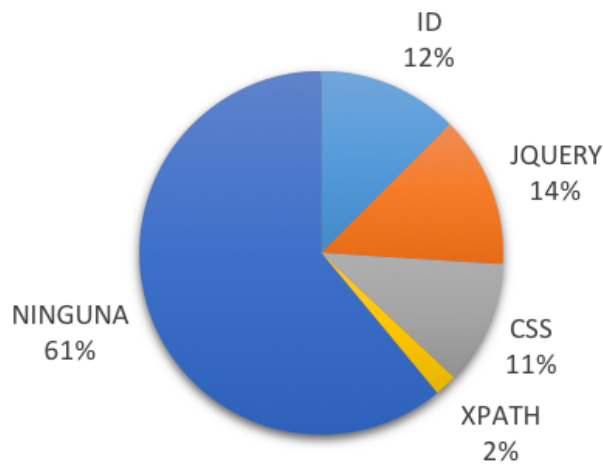


Figura 2.9: Métodos de referenciación identificados manualmente.

Con el objetivo de poder cuantificar los cambios en las referencias al DOM que los artefactos evidencian entre sus distintas versiones a través del tiempo en el repositorio, del mismo modo que fue descrito anteriormente, fueron analizadas de modo automatizado cada uno de los códigos fuentes correspondientes a cada una de las versiones de cada artefacto. En total fueron detectadas un total *903.342* referencias sobre un total de *3.697* artefactos con actualizaciones en sus códigos fuentes y referencias harcodeadas que han podido ser detectadas de este modo. Al analizar los cambios en la referencias entre versión y versión de cada artefacto, se ha determinado que que el *51,3 %* de los artefactos han cambiado alguna de sus referencias, y que un *48,7 %* las ha mantenido inalteradas, Figura 2.10.

También se ha analizado manualmente la evolución del código fuente de los artefactos correspondientes a la muestra de los *200*¹⁹ artefactos seleccionados aleatoriamente sobre el *29,5 %* de artefactos a los que automáticamente no pudo encontrarse ninguna referencia. De este análisis puede apreciarse en la Figura 2.11 que el *21 %* de los artefactos no evidencian cambios en sus referencias entre versión y versión, mientras que si ha sido detectado al menos un cambio en sus referencias en el *18 %* de la muestra, en donde el *61 %* restante de los artefactos no tenían referencias para ser analizadas.

¹⁹Listado online: <https://goo.gl/st66hL>

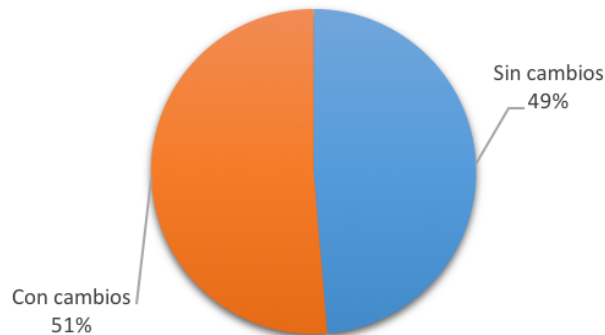


Figura 2.10: Artefactos con cambios de referencias entre versiones detectadas de modo automatizado.

Resulta oportuno mencionar que al haber sido este análisis realizado sobre los artefactos que han sido alguna vez mantenidos y actualizados en el repositorio, no resulta posible medir con mayor exactitud el impacto que las actualizaciones y modificaciones de los sitios webs aumentados tienen sobre los artefactos que allí se encuentran.

Sin embargo, en resumen, y si bien resulta imposible realizar un análisis de correlación entre las versiones de los artefactos y las versiones de los sitios webs aumentados a cada momento, por no estar estas últimas disponibles, y por encontrarse en muchos casos los primeros desmantenidos, luego del análisis realizado, donde se ha podido determinar que al rededor del *50 %* de los artefactos actualizados han cambiado al menos una vez alguna de sus referencias entre versión y versión, es posible inferir con seguridad, que una cantidad muy importante de las actualizaciones realizadas en estos artefactos se encuentra relacionada con el mantenimiento de las referenciaciones al DOM Tree. Ya que ello resulta ser necesario cuando un sitio web aumentado cambia los identificadores en el documento HTML, o bien cambia su forma o estructura, afectando las referencias que habían sido seleccionadas por el desarrollador del artefacto, al momento de su creación o de su última actualización.

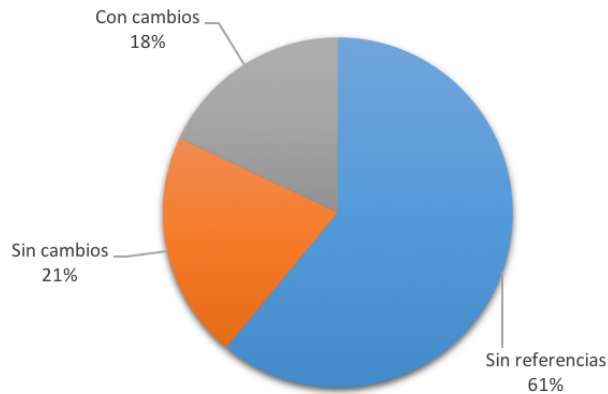


Figura 2.11: Artefactos con cambios de referencias entre versiones detectadas de manualmente.

2.4.3. Greasemonkey API y Librerías de terceros

Como fue mencionado anteriormente, los artefactos de Aumentación Web creados por usuarios finales, suelen utilizar una pequeña API²⁰ provista por el weaver Greasemonkey. Con el objetivo de poder determinar en qué medida los artefactos utilizan esta API, se analizaron los códigos fuentes correspondientes en busca de invocaciones a la funcionalidad provista por dicha API.

Para ello y haciendo uso nuevamente de expresiones regulares, (Anexo C, segundo apartado) se procesaron todos los textos correspondientes al código fuente de la última versión de cada artefacto. Este procesamiento permitió determinar que *1.942* artefactos del total de *7.538* hacían uso alguna vez de alguna de las funcionalidades provistas por la API, representando ello el *25,7%* de los artefactos.

Entre las funcionalidades provistas, existen operaciones para persistir y recuperar pares (*clave, valor*), para agregar estilos CSS al sitio web aumentado, para hacer solicitudes, logs, etc. Considerando que los nombres de las operaciones disponibles son lo suficientemente autoexplicativos, en la Tabla 2.5, puede apreciarse el ranking de utilización de estas operaciones por parte

²⁰https://wiki.greasepot.net/Greasemonkey_Manual:API

Tabla 2.5: Artefactos que utilizan API Greasemonkey.

Operación	Cantidad de Artefactos
GM_getValue	883
GM_setValue	877
GM_addStyle	775
GM_xmlHttpRequest	711
GM_log	299
GM_registerMenuCommand	278
GM_deleteValue	240
GM_openInTab	134
GM_getResourceText	93
GM_listValues	80
GM_setClipboard	78
GM_getResourceURL	36

de los artefactos compartidos en la comunidad a la fecha del relevamiento, en donde naturalmente, cada artefacto puede hacer uso de más de una las operaciones.

De este ranking, puede destacarse, que solo 711 artefactos (9,4 % del total) contenían invocaciones a la operación `GM_xmlHttpRequest`. Es a través de esta operación que estos artefactos pueden, potencialmente, realizar las solicitudes *cross-domain* mencionadas con anterioridad. Sin embargo, para completar este análisis en relación a las solicitudes HTTP, de igual modo fueron procesados los códigos fuente en busca de invocaciones a la DOM API o bien a la operación `ajax` provista por la librería jQuery. De ello resultó que un 6 % de los artefactos, exactamente 460, hacen solicitudes de manera nativa (DOM API) y que un 4,6 % de los artefactos hacen solicitudes utilizando jQuery (337 artefactos).

Vale la pena mencionar que estas solicitudes nativas y a través de jQuery, se encuentran sujetas a las limitaciones SOP, no pudiendo el artefacto a través de su uso, hacer solicitudes a dominios distintos del aumentado de modo irrestricto. Habiéndose verificado que se trata de conjuntos disjuntos, la Figura 2.12 ilustra la utilización de solicitudes por parte de los artefactos.

Además de poder utilizar la API del weaver Greasemonkey, los arte-

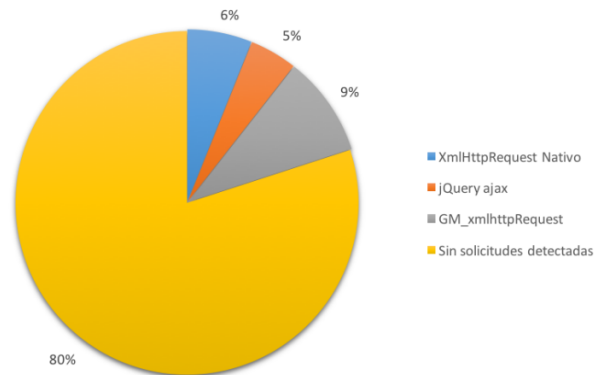


Figura 2.12: Artefactos y tipos de solicitudes HTTP detectadas automáticamente.

factos de Aumentación Web creados por usuarios finales, suelen importar en su código fuente librerías de terceros. Para ello, el desarrollador del artefacto, indica en su definición, que desea disponer de la funcionalidad provista por una librería externa a la definición de su propio artefacto. Al analizar nuevamente los códigos fuentes de cada uno de los artefactos publicados en el repositorio en sus últimas versiones, se tuvo la oportunidad de extraer esta información con el objetivo de poder comprender en detalle qué tipo de importaciones realizan los desarrolladores. En total, pudo determinarse, que *2.114* importaciones fueron realizadas por *1.696* artefactos que incorporaban al menos una importación de librerías de terceros, esto representa el *22,5 %* de los artefactos publicados en el repositorio.

De los datos obtenidos, resulta destacable la popularidad de la librería jQuery entre los desarrolladores de esta comunidad, ya que se detectó que dicha librería es importada, en alguna de sus versiones, en un total de *1.628* artefactos. Esto permite afirmar que casi la totalidad de los artefactos que realizan importaciones de terceros, precisamente un *95,9 %* de ellos, importan la librería jQuery.

jQuery, es una librería que facilita la manipulación del DOM Tree, agilizando su manipulación en la selección de sus nodos, el manejo de eventos, el manejo de las solicitudes asincrónicas mencionadas anteriormente, y proveyendo la posibilidad de realizar algunas animaciones rápidamente y de forma

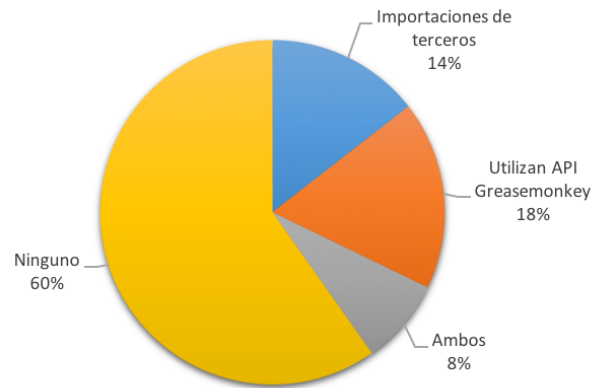


Figura 2.13: Utilización API Greasemonkey e importaciones de terceros.

compatible entre los exploradores. Ninguna del resto de las librerías de terceros importadas tiene el mismo nivel de relevancia relativa. Sin embargo, se destaca la importación de librerías para generación de gráficos, manipulación de tablas, de contenido multimedia, de fechas, de criptografía, diálogos modales, etc (Anexo D).

Final y naturalmente, no todos los artefactos que realizan importaciones de librerías de terceros, utilizan la API del weaver Greasemonkey, del mismo modo que no todos los artefactos que utilizan esta API, realizan importaciones de terceros. La Figura 2.13 muestra que tan disjuntos son estos conjuntos, donde además puede apreciarse rápidamente, que una gran cantidad de artefactos del total de los artefactos disponibles en el repositorio, no se apoyan ni en librerías de terceros, ni en la API Greasemonkey.

Capítulo 3

CrowdMock

El proceso propuesto ha sido denominado *CrowdMock*, y propone, desde un punto de vista ingenieril, el involucramiento de todos los actores en el proceso de construcción de los artefactos de Aumentación Web por usuarios finales. Esto es, a través de un proceso basado en el crowdsourcing de sus actividades, se provee a la masa de usuarios la posibilidad de delegar a la propia masa, distintas actividades que abarcan desde la elicitación y la definición de los requerimientos de aumentación, hasta la construcción, el testeo, la distribución y el mantenimiento de los artefactos de Aumentación Web obtenidos; en el marco de un proceso que provee la integración de los actores y fundamentalmente, de los artefactos de aumentación con los sitios webs aumentados. En las siguientes secciones se describen las características fundamentales de este proceso, identificando los actores involucrados y las actividades a realizarse, partiendo desde una perspectiva global hacia una particular para cada una de sus etapas.

3.1. Perspectiva general

CrowdMock involucra activamente tanto a los dueños de los sitios webs, como a los desarrolladores y a los usuarios finales de los artefactos de au-

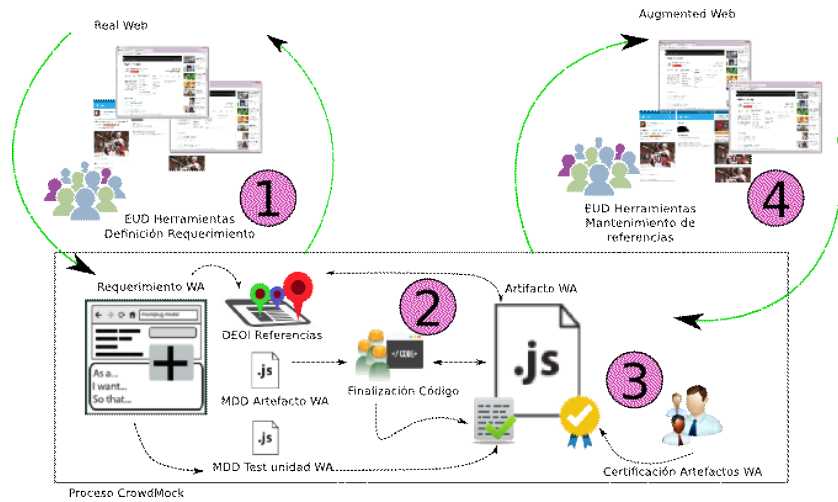


Figura 3.1: Visión general CrowdMock.

mentación. La masa de usuarios podrá descubrir y definir sus propios requerimientos, haciendo uso de herramientas provistas para ese objetivo (Figura 3.1₁), al tiempo que los usuarios con habilidades de programación (los desarrolladores), podrán completar las definiciones iniciales de los artefactos de aumentación que son originalmente derivadas de los requerimientos (Figura 3.1₂). Los dueños de los sitios webs, tendrán oportunidad de certificar/aprobar estos artefactos de aumentación (Figura 3.1₃, para finalmente distribuirlos a todos los usuarios que navegan dichos sitios (Figura 3.1₄), proveyéndoles la oportunidad de ejecutarlos e incluso involucrarse activamente en su mantenimiento en relación a las referencias *DEOI* (*DOM Element Of Interest*) que pudieran fallar durante su uso y que serán descriptas posteriormente.

Esta visión general del proceso propuesto, permite distinguir y destacar sus productos intermedios más relevantes. En un principio, cuando la masa de usuarios define los requerimientos de aumentación, al hacerlo, obtiene por resultado la especificación de sus requerimientos en términos de un prototipo de alta fidelidad. Este prototipo inicial permite la generación automática de: un conjunto de referencias de interés al sitio web aumentado (referencias *DEOI*), una versión inicial e incompleta del artefacto de aumentación, y un conjunto de casos de prueba que son derivados automáticamente de los requerimientos. Al ser completado el código fuente del artefacto de au-

mentación, los usuarios finales con habilidades de programación completan también los casos de prueba, agregando aquellos que consideren adecuados y obteniéndose así, el artefacto de aumentación y la definición del conjunto de casos prueba definitivo para el artefacto. Este artefacto estará listo para ser utilizado mientras sean superados los casos de prueba. Los dueños de los sitios webs, tendrán oportunidad de seleccionar aquellos artefactos que deberán ser sugeridos a los visitantes de sus sitios webs en las sesiones de navegación, certificando/aprobando su diseminación entre sus usuarios.

Una vez finalizada la construcción y realizada la publicación del artefacto de Aumentación Web, aquellos usuarios de la masa de usuarios que resultó involucrada en las distintas etapas del proceso, y aquellos que quieran involucrarse posteriormente, mantienen los artefactos intermedios principales, de modo que a través de un soporte adecuado por parte de las herramientas provistas, tendrán oportunidad de conocer el estado de los distintos artefactos en relación a la superación de los distintos casos de prueba que fueron creados originalmente y actuar en consecuencia (naturalmente los casos de prueba pueden también ser evolucionados).

Como fue estudiado en el capítulo anterior, los artefactos de aumentación suelen requerir mantenimiento cuando las referencias al sitio web aumentado necesitan ser actualizadas, debido a que el sitio web, por algún motivo, las hubiere cambiado. Este conjunto de referencias del artefacto, denominadas DEOI, podrá ser mantenido por toda la masa de usuarios del artefacto, debido a que las referencias a los sitios webs aumentados, fueron desacopladas del artefacto durante el proceso y que éstas pueden ser coleccionadas y descubiertas por toda la masa de usuarios interesada en el artefacto de aumentación, al mismo momento de su utilización. Para ello, se proveen herramientas específicas que serán descritas en mayor detalle en capítulos posteriores.

Si bien en esta primer presentación del proceso, las herramientas para la especificación de requerimientos y para el mantenimiento de los artefactos han sido resaltadas explícitamente en la Figura 3.1, es oportuno destacar que el proceso requiere de un repositorio en donde todos sus productos y subproductos puedan ser gestionados por los distintos actores involucrados. A este repositorio *AR* (*Augmenters Repository*), que da soporte a todo el proceso, se lo ha denominado *UserRequirements*.

Del mismo modo, el enfoque provee las herramientas para dar soporte a las actividades relacionadas con la finalización del código fuente de los artefactos por parte de los usuarios finales con habilidades de programación. Teniendo presente el capítulo anterior, en donde pudo apreciarse que en las comunidades de desarrolladores de artefactos de Aumentación Web, la librería jQuery tiene una gran popularidad, se provee de un soporte basado en *jQuery Plugins*, de modo que los desarrolladores puedan contar con herramientas afines para poder completar la funcionalidad de los requerimientos de aumentación en la implementación de los artefactos. Estas herramientas también serán descritas en mayor profundidad en capítulos posteriores.

En la siguiente sección, se describen en detalle las actividades del proceso y la delegación de responsabilidades a los distintos actores involucrados, haciendo hincapié en las ventajas obtenidas por estos actores, que desde el punto de vista de la adaptabilidad de las aplicaciones webs, ha sido denominado un *enfoque de adaptación negociada* [35].

3.2. Negociación de adaptaciones basadas en Aumentación Web

Como fue presentado en el capítulo anterior, existe una masa de usuarios finales con habilidades de programación capaz de construir y mantener una gran cantidad de artefactos de aumentación que resultan de interés a una gran cantidad de usuarios finales, existiendo entre los artefactos populares, artefactos con miles de instalaciones. En muchos casos, estos usuarios finales han contribuido a través de comentarios y devoluciones con sus desarrolladores. Sin embargo, los dueños de los sitios webs, resultan excluidos de esas comunidades, y como fue visto, no tienen ningún tipo de control sobre los artefactos que allí se publican, mientras los desarrolladores de estos artefactos se abocan a su construcción y mantenimiento en soledad, motivados y conducidos por su propia necesidad de adaptación, en donde habiendo tenido las habilidades de programación requeridas, pudieron satisfacerlas por sí mismos, terminando por compartir su esfuerzo con el resto de la comunidad.

A diferencia de las estrategias de adaptación vistas en el capítulo anterior, que fallaban en proveer una colaboración sin fisuras entre los actores involucrados, y en dónde en la mayoría de ellas, alguno de los actores era o bien excluido, o bien limitado en sus posibilidades de colaboración; el enfoque de adaptación negociada propuesto, se basa en tres principios básicos: todos los actores deben encontrar una ventaja por participar, los actores deben colaborar y por último la colaboración debe ser incentivada y facilitada a través de herramientas adecuadas.

La Figura 3.2 muestra ventajas que los actores pueden obtener con el enfoque propuesto. Los dueños de los sitios webs pueden establecer una relación de confianza con los desarrolladores, ya que al poder avalar los artefactos que los desarrolladores crean y comparten, los desarrolladores tienen la posibilidad de que sus artefactos sean ofrecidos a los usuarios del sitio web aumentado en sus sesiones de navegación, sin que estos últimos deban descubrirlos por si mismos. Este beneficio se extiende hasta los usuarios finales, que al poder distinguir cuáles artefactos cuentan con el aval por parte de los creadores de los sitios webs, de los que no, podrán obtener mayores certezas de que los artefactos no son maliciosos. Por otro lado, mantener a los usuarios finales involucrados, beneficia tanto a los desarrolladores de artefactos como a los dueños de los sitios webs. Los dueños de los sitios webs pueden obtener y coleccionar información acerca del comportamiento de sus usuarios mientras descubren nuevas expectativas y necesidades de personalización. Como fue estudiado en el capítulo anterior, las devoluciones de los usuarios finales generan un efecto positivo, que contribuye a mejorar las probabilidades de que un artefacto de actualización evolucione y sea mantenido en su funcionamiento a través del tiempo.

El enfoque negociado implica un acuerdo que puede ser alcanzado. Para ello debe existir un compromiso a colaborar. Esta colaboración y compromiso deriva en un esfuerzo de comunicación y coordinación ineludible, que requiere recursos adicionales, tanto en términos de tiempo, como en términos de esfuerzo cognitivo para sostener las relaciones entre los actores [53]. Para prevenir que estas tareas de comunicación y coordinación opaquen las ventajas propias de la colaboración en sí misma, el enfoque propone que los actores puedan realizar sus tareas de modo independiente, centrando sus es-

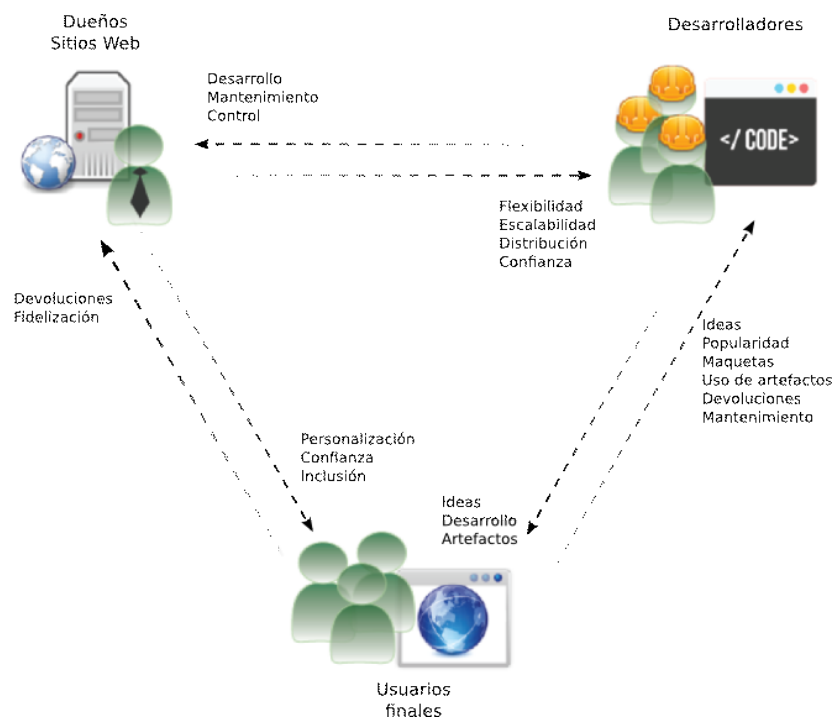


Figura 3.2: Adaptación negociada. Relaciones entre actores.

fuerzos en aquellas tareas que les implican un beneficio directo y evitando que puedan ser condicionados o impedidos en su realización por el resto de los actores en lo que sus tareas requirieren. A su vez, al proveerse herramientas integradas a los distintos actores que permiten facilitar la realización de las actividades, los esfuerzos relativos a la coordinación y comunicación pueden ser superados, para así alcanzar el beneficio común de todos los participantes.

3.3. Actividades y herramientas propuestas

En la Figura 3.3 se enumeran las actividades que deben realizar los distintos actores, como así también las herramientas que dan soporte a su realización. Para facilitar su comprensión, las actividades se presentan agrupadas por etapas, en dónde en cada etapa, es requerida la intervención de distintos actores y la utilización de distintas herramientas, sin que ello implique que las tareas deban ser realizadas de un modo estrictamente secuencial.

3.3.1. Herramientas

Como ha sido mencionado, el proceso propuesto, por su propia naturaleza, involucra actores con diferentes habilidades y necesidades, que realizan sus actividades de modo independiente, con distintos usos horarios y sin posibilidades de conocerse entre sí. Entre las herramientas que dan soporte al desarrollo de las actividades a llevarse a cabo por los distintos actores, es importante destacar, que si bien parte de ellas son herramientas para ser utilizadas por usuarios finales sin habilidades de programación, las herramientas se encuentran integradas, utilizando un repositorio y un *front-end* en común, dando lugar a la coordinación de los usuarios en la realización de sus actividades en el contexto descripto. En la Figura 3.3, se indica el soporte provisto por las herramientas en la realización de las actividades. A continuación se

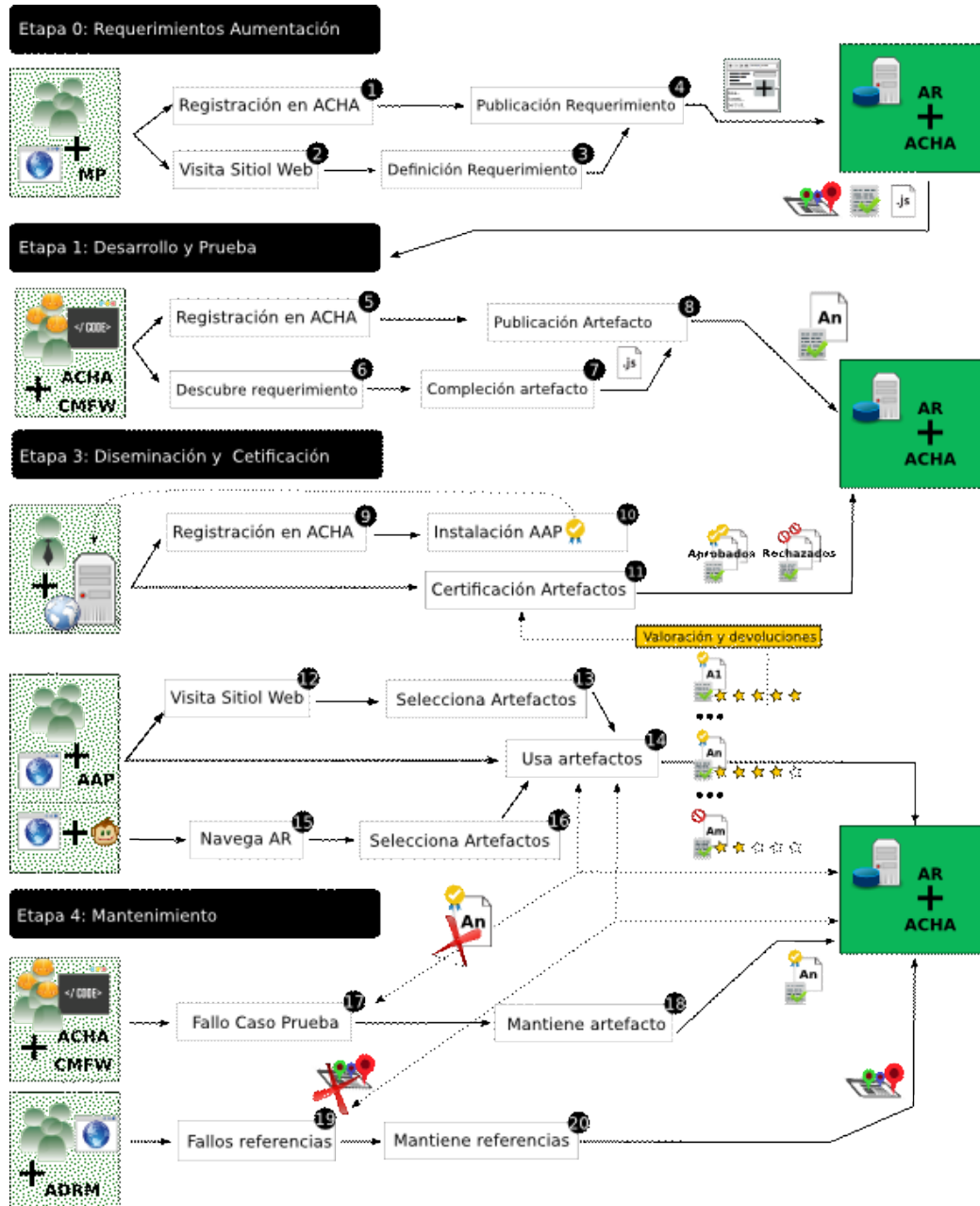


Figura 3.3: Actividades y herramientas.

las describe brevemente.

- MP (*MockPlug*): es un herramienta de Aumentación Web para la definición de requerimientos de aumentación por usuarios finales, implementada como una extensión/complemento para el explorador Firefox. Su utilización permite la construcción de prototipos de alta fidelidad sobre los sitios webs, en virtud de poder especificar los requerimientos en de un modo ágil y conveniente para los usuarios y provechoso para el resto del proceso.
- AR (*Augmenters Repository*): es una aplicación web, denominada UserRequirements, en donde son gestionados y publicados los artefactos y los productos intermedios del proceso. Los usuarios pueden navegar el repositorio, descubriendo, valorando y evolucionando requerimientos, artefactos, etc.
- ACHA (*Augmenters Central Hub Application*): Es el front-end de la aplicación web UserRequirements, permite entre otras cosas, la diseminación de los artefactos certificados a todos los visitantes de un sitio web, como así también la recolección de información sobre la utilización de los artefactos, las valoraciones de los usuarios finales, etc.
- AAP (*Augmenters Access Point*): Es un componente *client-side* embebido en los sitios webs registrados en el ACHA. Su instalación por parte de los dueños de los sitios webs, permite que sus visitantes descubran y utilicen los artefactos que han sido certificados, mientras transcurren sus sesiones de navegación. Su instalación, por parte de los dueños de los sitios webs, simplemente consiste en la inclusión de una librería JS.
- CMFW (*CrowdMock Framework*): Es un framework provisto a los desarrolladores para completar los artefactos de aumentación. Este framework, entre otras funcionalidades, provee soporte para el manejo de referencias DEOI de un modo desacoplado del artefacto de Aumentación Web. Si bien su utilización no es un requisito, los artefactos que importen esta librería habilitarán a los usuarios finales a colaborar en el mantenimiento de las referencias DEOI.

- ADRM (*Augmenters DEOI Reference Maintenance*): Es componente *client-side* de Aumentación Web para el mantenimiento de referencias DEOI por parte de la masa de usuarios. Al momento del uso de un artefacto, cuando sus referencias DEOI fallan, los usuarios tienen oportunidad de utilizar esta herramienta para descubrir nuevas referencias y compartirlas con la masa de usuarios. En este contexto, se habilita a los usuarios finales a poder recuperar el artefacto por ellos mismos, sin necesidad de intervención de ningún otro actor.

3.3.2. Actividades

En lo sucesivo se describen y analizan las actividades correspondientes a las distintas etapas del proceso propuesto, desde la definición de los requerimientos de aumentación, hasta su utilización en el marco de un proceso de integración que facilita y promueve el mantenimiento.

Requerimientos de Aumentación Web

El enfoque propone que cualquier usuario final, provisto de la herramienta MockPlug integrada en su explorador, pueda al momento de navegar sus sitios webs de interés, definir su propio requerimiento de aumentación a través de un prototipo de alta fidelidad. Esto es, durante una sesión de navegación con el sitio web, sobre el que el usuario final tiene una necesidad de aumentación, el usuario especifica sus necesidades agregando, quitando y modificando los elementos del DOM Tree del sitio web en función de su necesidad. Al finalizar, el usuario obtendrá una maqueta o prototipo de alta fidelidad que especifica su requerimiento. Una vez compartido su requerimiento con el resto de los actores, a través de su publicación en el repositorio de aumentación, el resto de los usuarios interesados en el mismo sitio web, podrá reproducir su requerimiento como así también colaborar en su definición y refinamiento. De este modo el usuario original, junto con el resto de los usuarios finales interesados, tendrán oportunidad de descubrir los requere-

rimientos de aumentación para un sitio web en particular, en un proceso de elicitación en el que la masa de usuarios por si misma descubre y refina sus propios requerimientos.

La utilización de la herramienta MockPlug no supone ni requiere ninguna habilidad de programación en particular por parte del usuario final. Siendo una herramienta de maquetado, también le permite al usuario final agregar anotaciones y observaciones, con el objetivo de que su requerimiento pueda ser comprendido fácilmente por el resto de los actores involucrados. Al momento de publicar y compartir su requerimiento, el usuario final debe completar un formulario en relación a la *historia de usuario* correspondiente. Así, tanto el prototipo de alta fidelidad, como una historia de usuario en formato textual, en conjunto, conformarán la definición del requerimiento de aumentación. Estas descripciones textuales en forma de historias de usuario, facilitan al resto de los usuarios navegar el repositorio en busca de artefactos y requerimientos, mientras los prototipos, permiten la definición de los requerimientos en términos de componentes de interfaz de usuario concretos que cualquier interesado puede comprender con facilidad.

Lo novedoso, en términos de maquetado, es que el prototipo se construye con técnicas de Aumentación Web *sobre* el sitio web a aumentar. A diferencia de las herramientas de maquetado tradicional, en donde el usuario empieza desde cero sobre una hoja en blanco y de modo desintegrado de la aplicación relacionada, en el enfoque propuesto, el usuario final genera su prototipo *sobre* el sitio web original, con el cual puede interactuar del mismo modo que con la aplicación web aumentada. Por este motivo esta nueva forma de prototipo ha sido denominada *RAM (Runnable Augmentation Mockup)*.

Aquellos usuarios finales con requerimientos de aumentación que no dispongan de la posibilidad de utilizar MockPlug en sus sesiones de navegación, pueden realizar su requerimiento a través del repositorio, especificando su historia de usuario y solicitando la realización del prototipo a la masa de usuarios.

Desarrollo y prueba

Una vez que el requerimiento es descubierto y definido por la masa de usuarios finales, los usuarios con habilidades de programación, los desarrolladores, completan su funcionalidad. Para ello, cuentan con una versión inicial e incompleta del artefacto, derivada automáticamente del prototipo. Así mismo cuentan con el framework CMFW, que les permite abstraerse de las referencias al DOM Tree y concentrarse en la implementación del comportamiento requerido para cada uno de los componentes del requerimiento. Como fue mencionado anteriormente, este framework se encuentra basado en jQuery Plugins. Sin embargo, la utilización de estas herramientas es opcional y el desarrollador cuenta con total libertad para implementar el artefacto de aumentación requerido. Es decir, el enfoque propuesto, no se restringe a los artefactos desarrollados con las herramientas provistas, sino que pretende ser compatible en términos técnicos, con el modo tradicional de construcción de artefactos de aumentación por usuarios finales, en dónde los desarrolladores cuentan con distintos niveles de habilidades de programación y total libertad en relación a las herramientas a utilizar en su proceso de construcción.

La masa de usuarios podrá obtener varios artefactos de aumentación que den respuesta al mismo requerimiento, pudiendo valorar y descubrir aquel que de la mejor solución.

El enfoque propone la ejecución temprana y distribuida de los casos de prueba entre los usuarios finales y los desarrolladores, de modo que puedan advertir si los artefactos de su interés necesitan su revisión, y cuando así sea, los usuarios finales deben ser prevenidos automáticamente de ejecutar estos artefactos defectuosos hasta que éstos puedan ser recuperados.

Diseminación y certificación

Los usuarios finales tienen el control sobre los artefactos de aumentación que deben incorporarse cuando navegan los sitios webs. Al momento de visitar un sitio web, podrán seleccionar aquellos artefactos de aumentación que hubieran sido aprobados por los dueños de los sitios webs, a través del

componente AAP (Augmenters Access Point). Utilizando esta herramienta podrán elegir las historias de usuario de su interés, pudiendo incluso seleccionar el artefacto de aumentación de su preferencia, si existiera más de uno que la implemente.

Por su parte, los dueños de los sitios webs, deben inspeccionar los artefactos en el repositorio y aprobar aquellos que consideren apropiados y que desean que sean ofrecidos a todos los visitantes de su sitio web. Para ello, naturalmente deben registrarse y para validar su rol de dueño/administrador de un sitio web en particular, también deben llevar adelante un proceso de verificación, que consiste básicamente en la instalación de un archivo (cuyo contenido es generado con ese objetivo) en una URL de su dominio. Una vez realizado el proceso de verificación, deben también incorporar una pequeña modificación en su sitio web, que consiste en incluir el componente AAP en las páginas webs principales de su sitio.

Los artefactos no certificados por los dueños de los sitios webs, no serán ofrecidos a los visitantes del sitio. Cuando un usuario final desee utilizar un artefacto no certificado, podrá hacerlo, pero no a través de la utilización del AAP, debiendo utilizar el weaver de su preferencia para poder incluirlo al momento de visitar el sitio. Para descubrir estos artefactos no certificados, el usuario final podrá explorar el repositorio haciendo uso del AR, del mismo modo que en las comunidades de Aumentación Web por usuarios finales.

En cualquier caso, el usuario final, una vez seleccionados los artefactos de aumentación que desea utilizar, podrá valorar y realizar devoluciones acerca del mismo. El ACHA, registrará el uso de los distintos artefactos por parte de los distintos usuarios, y tanto los desarrolladores, como los dueños de los sitios webs, podrán disponer de esta información para conocer las preferencias de los usuarios finales.

Mantenimiento

En la Aumentación Web por usuarios finales tradicional, cuando por algún motivo un artefacto deja de funcionar, los usuarios finales no tienen ningún tipo de información al respecto. Los artefactos defectuosos simplemen-

te se ejecutan de modo incompleto o incorrecto sin proveer ningún tipo de advertencia, pudiendo los usuarios finales advertir o no el estado defectuoso del artefacto. Si éstos son capaces de advertir que el artefacto es defectuoso, pueden dejar de utilizarlo, pueden intentar repararlo por ellos mismos para ellos mismos, o pueden realizar algún tipo de devolución al desarrollador que los compartió para advertirle. Vale destacar que solo puede actualizar un artefacto en la comunidad, aquel usuario que lo haya creado, de modo que la posibilidad de que una nueva versión de un artefacto se distribuya a los usuarios finales, se encuentra sujeta a la disponibilidad, posibilidades y voluntad de una única persona.

Esta realidad, consecuencia de la naturaleza desintegrada de los artefactos de aumentación, respecto de los sitios webs aumentados, en suma con la poca cantidad de desarrolladores involucrados en el desarrollo de los artefactos y la alta probabilidad de que un sitio web, cambie su forma, estructura o contenido, atenta contra la calidad y la sobrevivencia de los artefactos a través del tiempo.

Habiendo sido estudiado que los cambios en las referencias que los artefactos sostienen sobre los sitios webs aumentados, generan gran cantidad de requerimientos de mantenimiento, el enfoque propone involucrar a los usuarios finales en su realización. A través del componente ADRM, cuando fallan las referencias de un artefacto sobre un sitio web, el usuario es advertido teniendo oportunidad de seleccionar nuevos elementos y referencias. Sin requerir de su parte habilidades de programación, el usuario final tendrá oportunidad de recuperar el artefacto y compartir con el resto de los usuarios las nuevas referencias encontradas, cuando estas nuevas referencias seleccionadas, tengan por efecto que el artefacto vuelva a superar todos los casos de prueba.

Capítulo 4

Requerimientos de Aumentación Web

En CrowdMock los *stakeholders* (interesados), tanto usuarios finales como desarrolladores, pueden colaborar en la gestión de los requerimientos. Se ha intentado orientar esta etapa del proceso y las herramientas y los productos que la soportan, en busca de facilitar la colaboración por parte de los desarrolladores y los usuarios, con el objetivo de obtener rápidamente una definición validada del requerimiento sobre un escenario contrastable, previo a la implementación del artefacto de aumentación.

Como ha sido estudiado en el capítulo 2, en las comunidades de Aumentación Web, el contrato entre los usuarios finales de los artefactos y sus desarrolladores es muy débil y por ello se ha buscado que los interesados puedan obtener un escenario de utilización que les permita verificar la aceptación de los requerimientos rápidamente, previo a la implementación de los artefactos. El modo en que los requerimientos de Aumentación Web son especificados, debe ser realmente cercano a la solución esperada, facilitando además su desarrollo.

En este capítulo se describen las actividades relacionadas con la definición de requerimientos de Aumentación Web por parte de la masa de usuarios. Esta etapa del proceso es soportada por la herramienta MockPlug, un complemento para el explorador Firefox que utiliza técnicas de Aumentación

Web tradicionales con el propósito de permitir a los usuarios finales descubrir y definir sus propias necesidades de aumentación. La masa de usuarios descubre sus necesidades al mismo tiempo que las define en términos de requerimientos. Estos requerimientos son especificados a través de prototipos de alta fidelidad, donde la utilización de técnicas afines para su definición es una pieza clave y fundamental para todo el enfoque.

4.1. Definición de requerimientos

Al momento de la definición del requerimiento de Aumentación Web, diferentes interesados, usuarios finales con y sin habilidades de programación, colaboran activamente en su especificación. Para ello, el enfoque propone la utilización de dos tipos de artefactos para la definición de los requerimientos: historias de usuario y *mockups* (prototipos UI). Tanto los prototipos como las historias de usuario, suelen ser artefactos muy utilizados en el marco de las metodologías ágiles de desarrollo de software [17, 67]. Sin embargo, en el enfoque propuesto, los prototipos UI difieren en relación a los prototipos tradicionales, ya que los usuarios definen sus prototipos de alta fidelidad *sobre* el sitio web a aumentar en sí mismo.

Son definidas tres actividades en relación a los artefactos mencionados anteriormente: i) definición del requerimiento, ii) refinado del requerimiento y iii) priorización del requerimiento. En la Figura 4.1 se ilustran estas actividades y los productos e interesados involucrados en su realización.

La primer actividad, correspondiente a la definición del requerimiento, está relacionada con producir la primer especificación del mismo. Un interesado, escribe una historia de usuario y define un primer prototipo acerca de su necesidad particular. Ambos productos, la historia de usuario y el prototipo, conforman la especificación de su requerimiento en CrowdMock.

La actividad de refinamiento permite enriquecer y mejorar su definición, constituyéndose como un punto en el que los demás usuarios interesados puedan involucrarse colaborando activamente y generando nuevas versiones refinadas, estableciendo una relación de trazabilidad entre los distintos refinamientos, donde una especificación puede ser *refinada por* otra que la

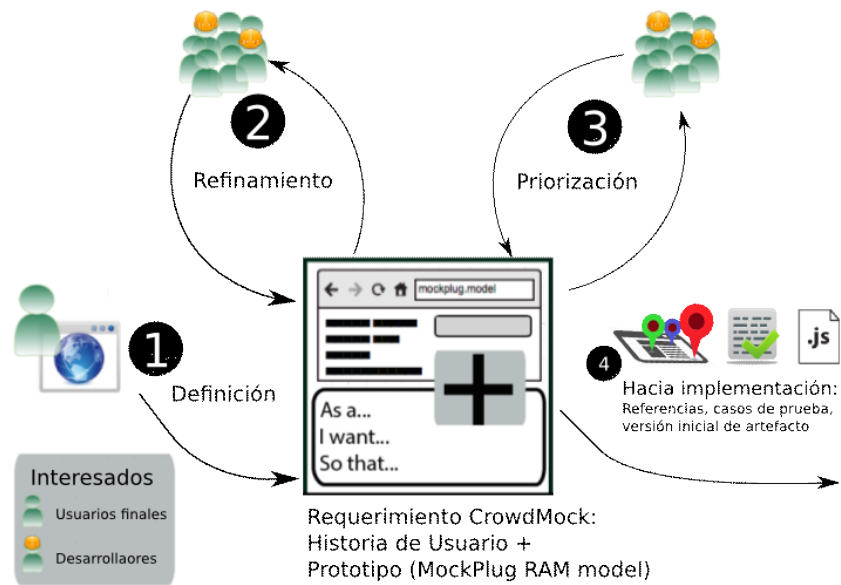


Figura 4.1: Requerimiento MockPlug.

enriquece o se distingue en algún aspecto.

Finalmente, la actividad de priorización busca canalizar los esfuerzos de implementación hacia aquellos refinamientos que la masa de usuarios desea que sean implementados. Esta actividad es soportada por una actividad común y bien conocida, como lo es la votación. Los interesados votan los refinamientos y los desarrolladores pueden conocer cuál de las versiones del requerimiento, es la que la mayoría de los interesados desea que sea implementada. De cualquier modo, son soportadas también otras actividades propias de estas comunidades como los foros de discusión, en dónde los usuarios pueden comentar los distintos refinamientos. Naturalmente, el orden de las actividades dos y tres, puede variar y continuar indefinidamente. Mientras los requerimientos evolucionan los usuarios los votan, siendo posible que nuevos refinamientos surjan incluso una vez que el proceso haya alcanzado la implementación de los artefactos.

4.2. Productos y actividades

En esta sección se describen los productos y las actividades relativas a la definición de los requerimientos. Como fue mencionado anteriormente, la definición de los requerimientos es una pieza clave de todo el enfoque y estos se encuentran definidos en términos de historias de usuario y prototipos.

Una historia de usuario es una descripción en lenguaje natural que expresa lo que el usuario desea obtener. Las historias de usuario son utilizadas en las metodologías ágiles de desarrollo de software y generalmente se ajustan a una plantilla que considera tres atributos: un rol, un objetivo/necesidad y una razón [17]. El rol define el tipo de usuario que interactúa con la aplicación que desea disponer de la característica descrita en el objetivo/necesidad. El objetivo/necesidad describe de forma textual el requerimiento que la aplicación debería incorporar. Complementariamente, la razón, pertenece al contexto de la aplicación cuyo estado motiva que los usuarios requieran que ésta provea la funcionalidad descrita en el objetivo/necesidad.

Los mockups (prototipos) son una forma de maqueta UI en papel o utilizando imágenes digitales, construidas con el objetivo de acordar requerimientos de presentación, entre otros, con usuarios finales. Comúnmente, una maqueta o prototipo UI de media o alta fidelidad se parecerá a la UI de la Aplicación, pero sin constituir un tipo de artefacto con el cual pueda interactuarse. Ha sido probado que cuando se utilizan maquetas en el desarrollo de software, el costo y el esfuerzo de su desarrollo e implementación es reducido [66]. Una de las principales ventajas de la utilización de prototipos en el proceso de desarrollo, es que permiten la definición temprana de aspectos relativos a la interacción y a la presentación, pudiendo ser utilizadas como un lenguaje visual apropiado para comunicar y acordar requerimientos entre usuarios y analistas o desarrolladores [58]. Las maquetas o prototipos incorporados en el enfoque, son representados a través de *modelos* MockPlug. Como ha sido mencionado anteriormente, a estos modelos se los ha denominado RAMs.

Un RAM contiene definiciones sobre las alteraciones realizadas en el sitio web que desea aumentarse, con el objetivo de definir el prototipo de la aumentación esperada. A primera vista, estos modelos son prototipos de

alta fidelidad [82], donde el usuario final define y representa el artefacto indicando las modificaciones que desea que este provea. Sin embargo, desde el punto de vista del desarrollador, cuando estos refinan un requerimiento, puede considerarse que están programando visualmente lo que el artefacto va a adaptar/aumentar en la UI del sitio web original, pudiendo incluso especificar aspectos de menor nivel de abstracción, oportunos para la generación de código fuente en etapas posteriores. Así es que los RAMs, además de proveer un soporte documental en la utilización de maquetas para describir aspectos visuales e interactivos de un requerimiento, proveen la primer versión parcialmente funcional del artefacto, con el cual es posible interactuar y definir cierto tipo de comportamiento inicial.

Como será descrito posteriormente, los RAMs son instancias del meta-modelo MockPlug [33]. Estos modelos se corresponden con una única historia de usuario, pero una historia de usuario admite la coexistencia de múltiples modelos en consecuencia a la actividad de refinamiento. En comparación a otras herramientas de maquetado, lo novedoso de los modelos MockPlug, es que están definidos sobre las aplicaciones webs en sí mismas, constituyendo una aumentación real, con el objetivo de definir un requerimiento. En lugar de ser simples imágenes, estos modelos permiten agregar un conjunto de objetos con semántica precisa en términos de comportamiento y en términos de efectos a producir sobre el sitio web que se desea aumentar. Esta capacidad hace que los artefactos de requerimientos resultantes, sean muy cercanos a la definición de su implementación, permitiendo la incorporación de funcionalidad rápidamente previo a la generación del artefacto propiamente dicho.

En el enfoque propuesto, las historias de usuario tienen un rol relacionado a la propia dinámica de las comunidades de Aumentación Web estudiadas en el capítulo 2. Los repositorios utilizados por estas comunidades proveen funcionalidades a los usuarios para buscar artefactos de su interés de varias maneras: por relevancia (popularidad), por sitio web, o simplemente buscando texto en las descripciones de los artefactos. La decisión de incorporar historias de usuario a la definición de los requerimientos en el enfoque, se encuentra motivada en que, según estudios recientes¹, son el método de documentación más utilizado en el contexto de las metodologías ágiles de

¹10 th Annual State of Agile Survey - <http://stateofagile.versionone.com/>

desarrollo y permiten extraer la esencia de los requerimientos de forma textual, facilitando ello la estructuración de los requerimientos y artefactos en el repositorio para su posterior descubrimiento.

En relación a las tres actividades descritas anteriormente, es importante destacar que en la realización de la primera de ellas por parte de un usuario final, será definido el requerimiento CrowdMock, esto es: la historia de usuario y el modelo MockPlug. El usuario interesado escribe la historia de usuario y también define el prototipo de su requerimiento utilizando MockPlug, obteniendo así un modelo original. Esta actividad implica que el usuario especifica por primera vez un requerimiento que nunca antes había sido especificado. De este modo, la primera actividad implica dos actividades básicas de la ingeniería clásica de requerimientos: elicitación o descubrimiento de la necesidad y definirla en términos de un requerimiento. En el enfoque, la misma persona es la que tiene el conocimiento acerca de su necesidad y a su vez la especifica al definir el requerimiento original.

La segunda actividad, en donde el requerimiento original es refinado, implica el refinamiento de al menos uno de los dos artefactos que componen el requerimiento: la historia de usuario o el modelo MockPlug. En cualquier caso, la nueva versión del requerimiento es derivada de la anterior y posteriormente podrá ser trazada esta evolución.

Finalmente, como ha sido mencionado, la actividad de priorización de los requerimientos se basa en la votación. En particular se ha decidido utilizar la herramienta *like* en lugar de otras comunes como *rankings*, ya que ha sido mostrado que resulta ser más efectiva para filtrar requerimientos [70]. Aunque resulte trivial, es considerable que la actividad de refinamiento en sí misma, implica también una forma de priorización, ya que al ser realizada por un interesado, este último implícitamente considera que su refinamiento es más apropiado.

4.3. Expresividad

En relación al tipo de requerimientos de aumentación que pueden ser expresados en CrowdMock, resulta importante considerar qué tipos de modificaciones son realizadas por los artefactos publicados en las comunidades de Aumentación Web por usuarios finales. En el capítulo 2 sección 2.3, fueron analizados los artefactos tipificando las alteraciones que realizan sobre los sitios webs aumentados. En este apartado se los analiza desde un punto de vista tradicional en relación a la ingeniería de requerimientos.

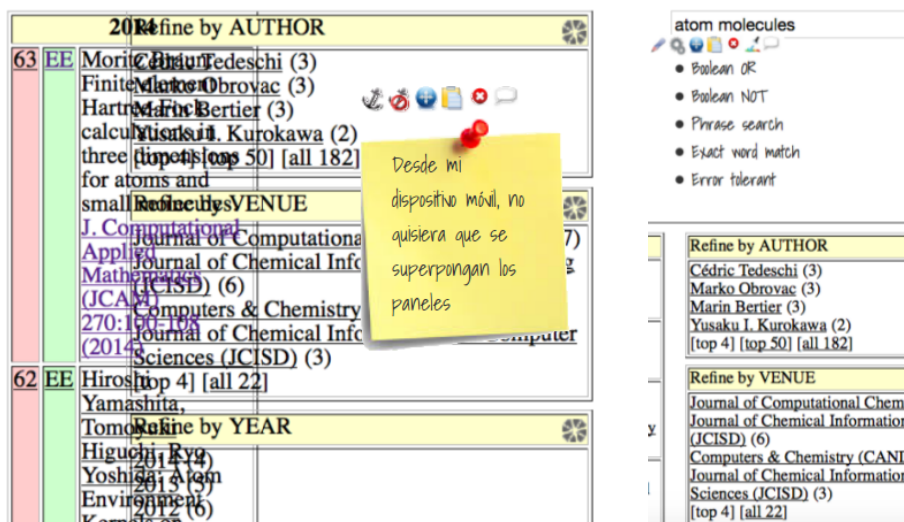
Considerando que un requerimiento en CrowdMock está compuesto por una historia de usuario y un prototipo, puede apreciarse que el enfoque se encuentra orientado a la definición de requerimientos funcionales, ya que ambos tipos de artefactos son utilizados para especificar este tipo de requerimientos.

La mayoría de los requerimientos implícitos de Aumentación Web que dan lugar a los artefactos compartidos en las comunidades, son requerimientos funcionales. Esto adquiere mayor sentido al considerar que las técnicas de Aumentación Web, por su propia naturaleza y definición, se encuentran orientadas a modificar los componentes del sitio web aumentado. Ciertos tipos de requerimientos no funcionales bien conocidos como seguridad, eficiencia, etc. son imposibles de resolver con técnicas de Aumentación Web, ya que ello requeriría cambios en el sitio web original, del lado del servidor, mientras que las técnicas de Aumentación Web están restringidas a la manipulación de los documentos hipermedia o de los DOMs Tree correspondientes, por fuera del ambiente de aplicación que los genera en el servidor web. Sin embargo, algunos requerimientos no funcionales pueden ser alcanzados por el enfoque.

4.3.1. Requerimientos no funcionales

Los requerimientos no funcionales tradicionalmente cubren el espectro de requerimientos relativos a la eficiencia, portabilidad, usabilidad, seguridad, etc [40]. Como fue mencionado anteriormente, aquellos requerimientos no funcionales que requieren modificaciones en el servidor no pueden ser alcanzados por la Aumentación Web. Sin embargo, pueden ser alcanzados

ciertos requerimientos no funcionales relativos a la usabilidad y a la accesibilidad, cercanos a la interfaz de usuario. Cuando los requerimientos no funcionales se encuentran de algún modo relacionados con la UI, el enfoque permite a los usuarios finales convertir cualquier elemento existente en un *widget* (componente) y manipularlo para especificar su necesidad. Esto hace posible, por ejemplo, especificar nuevos *layouts* (diseños) UI de acuerdo a las necesidades de accesibilidad de un usuario en particular o incluso de acuerdo al dispositivo utilizado para conseguir un layout más responsivo. La Figura 4.2a, muestra como un usuario ha utilizado MockPlug para definir un modelo que muestra problemas de superposición de los elementos. En este caso el usuario ha definido un modelo especificando que desea que la UI se adapte al tamaño de su dispositivo.



(a) Requerimiento responsivo DBLP (b) Menú Conectores de búsqueda

Figura 4.2: Requerimientos de usabilidad.

Algunos aspectos relativos a la usabilidad, no se encuentran relacionados con el layout existente, pero si a como una funcionalidad soportada es utilizada por sus usuarios. Por ejemplo, el motor de búsqueda del sitio *DBLP*², permite a los usuarios especificar varias condiciones de búsqueda

²<http://dblp.uni-trier.de/>

utilizando caracteres especiales en la expresión a buscar. Por ejemplo, si un usuario busca por “átomos — moléculas” está indicando que desea resultados que contengan o la palabra átomos, o la palabra moléculas. Hay varias condiciones de búsqueda que pueden ser especificadas de esta manera y probablemente un usuario encontraría más fácil utilizarlas si tuviera algún tipo de asistencia que lo ayudara a escribir sus consultas, como es mostrado en la Figura 4.2b.

4.3.2. Requerimientos funcionales

Los requerimientos funcionales que pueden ser expresados en el enfoque fueron categorizados considerando los tipos de aumentaciones más comunes en los repositorios:

- *navigation-based* relacionados a la navegación,
- *behavior-based* relativos a la funcionalidad/comportamiento,
- *content-based* relativos al contenido.

Además de facilitar la especificación de estas categorías de requerimientos, y del mismo modo que en las herramientas tradicionales de maquetado, cuando una característica a especificar no pudiere ser representada por los componentes visuales disponibles, se facilita al usuario la posibilidad de utilizar anotaciones. Estas anotaciones pueden ser expresadas por ejemplo con notas adhesivas, como puede apreciarse en la Figura 4.2a.

Requerimientos funcionales relativos a la navegación

Los modelos MockPlug soportan la posibilidad de especificar requerimientos basados en navegación, que involucran la adición, remoción o alteración de enlaces en los sitios webs, o bien la especificación de comportamiento

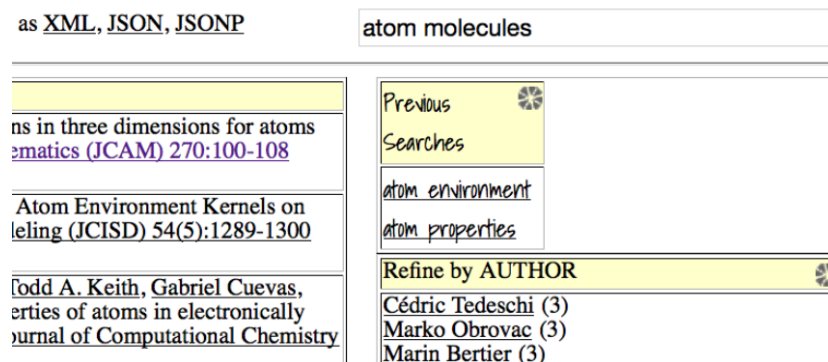


Figura 4.3: Nuevo panel de navegación.

que implique la ejecución de acciones de navegación a algún punto correspondiente de otras aplicaciones webs. Como ejemplo de este tipo de requerimiento de aumentación, considere la necesidad de agregar un panel de navegación a búsquedas anteriores en el motor de búsqueda de DBLP. Como es mostrado en la Figura 4.3, MockPlug permite a los usuarios especificar fácilmente este tipo de requerimientos agregando un conjunto de enlaces que disparan nuevas navegaciones no contempladas en el sitio web original. En el ejemplo, el usuario ha agregado un nuevo panel de “Búsquedas anteriores”.

Requerimientos funcionales relativos al comportamiento

Nuevas funcionalidades pueden especificarse utilizando modelos Mock-Plug agregando distintos tipos de elementos. Además, es posible especificar la necesidad de modificar funcionalidad existente mediante la introducción de cambios en la UI original y su comportamiento. Como ejemplo, es posible considerar que un usuario del sitio DBLP, desea modificar la funcionalidad “Refinar por autor”, de modo que en lugar de filtrar los artículos encontrados correspondientes a un autor en particular, éstos simplemente sean resaltados en el listado. Como se muestra en la Figura 4.4, el usuario ha realizado un modelo MockPlug donde fueron agregados dos widgets: un ícono de sobresaltado cerca del nombre del primer autor en el panel “Refinar por autor” y




2014	Refine by AUTHOR 
n, Jürgen Vogt, Heinz Dieter Rudolph: Why it is sometimes difficult position of a hydrogen atom by the semiexperimental method: aining the OH or the CH ₃ group. <i>Journal of Computational</i> 33-2342 (2014)	Cédric Tedeschi (3)  Marko Obrovac (3) Marin Bertier (3) Yusaku I. Kurokawa (2) [top 4] [top 50] [all 182]
2013	Refine by VENUE 
ence Hartree-Fock program for atoms and diatomic molecules. ications (CPHYICS) 184(3):799-811 (2013)	Journal of Computational Chemistry (JCC) (27) Journal of Chemical Information and Modeling (JCISD) (6) Computers & Chemistry (CANDC) (4) Journal of Chemical Information and Computer Sciences (JCISD) (3) [top 4] [all 22]
Cho, Joonghyun Ryu, Jae-Kwan Kim, Donguk Kim: Anomalies in a-complexes of spherical atoms in molecules. <i>Computer-Aided</i> (2013)	
vac, Cédric Tedeschi: Adaptive atomic capture of multiple molecules. (JPDC) 73(9):1251-1266 (2013)	
olodzki, Inken Groth, Stefan Heuser, Matthias Rarey: Reading PDB:	

Figura 4.4: Sobresaltado de artículos de un autor.

uno de los artículos pertenecientes al autor correspondiente fue sobresaltado.

Requerimientos funcionales relativos al contenido

Los requerimientos relativos al contenido pueden ser expresados quitando o moviendo contenido existente e incluso integrando contenido existente de otros sitios webs en el sitio web original. Como ejemplo, Figura 4.5, un usuario de DBLP podría querer ver en los artículos listados en un resultado de búsqueda, cuantas citas tiene cada uno de ellos según *Google Scholar*³. En este caso, el usuario puede especificarlo fácilmente, dirigiéndose primero a Google Scholar puede *colectar* el enlace a “citado por x” y luego agregarlo al artículo correspondiente en el listado. El enlace copiado sostiene sus atributos originales, y cualquier otro usuario podrá comprender fácilmente cual es el objetivo especificado.

³<http://scholar.google.com>

2013	
59	EE Jacek Kobus: A finite difference Hartree-Fock program for atoms and diatomic molecules. <i>Communications in Physics</i> 184(3):799-811 (2013) Cited by 12
58	EE Deok-Soo Kim, Youngsong Cho, Joonghyun Ryu, Jae-Kwan Kim, Donguk Kim: Anomalies in quasi-triangulations and beta-complexes of spherical atoms in molecules. <i>Computer-Aided Design (CAD)</i> 45(1):35-52 (2013)

Figura 4.5: Resultado de búsqueda aumentado con enlace “citado por x” .

CompleteSearch DBLP

a DBLP mirror with extended search capabilities maintained by [Hannah Bast, University of Fr](#)

zoomed in on 63 documents ... NEW: get these search results as [XML](#), [JSON](#), [JSONP](#)

2014	
63	EE Moritz Braun: Finite element Hartree-Fock calculations in three dimensions for atoms. <i>Communications in Physics</i> 270:100-108 (2014)
62	EE Hiroshi Yamashita, Tomoyuki Higuchi, Ryo Yoshida: Atom Environment Kernels on M. <i>Communications in Physics</i> 54(5):1289-1300 (2014)

Figura 4.6: Resultado de búsqueda UI original.

4.4. Ejemplo de definición, refinamiento y priorización

En este apartado se presenta un ejemplo de las actividades descritas anteriormente.

El usuario Pedro, desea aumentar los resultados de búsqueda en el sitio web DBLP, de modo que al listar los resultados de búsqueda, sean mostrados los resúmenes de los artículos, además de facilitarle la posibilidad de agregar artículos a su librería en *Mendeley*⁴. Para ello, Pedro ha definido un RAM, que se muestra en la Figura 4.7. A diferencia del sitio web original mostrado en la Figura 4.6, este modelo MockPlug, contiene dos nuevos enlaces “ver resumen” y “agregar a Mendeley”.

Aquí, puede apreciarse una de las características más importantes del

⁴<http://www.mendeley.com>

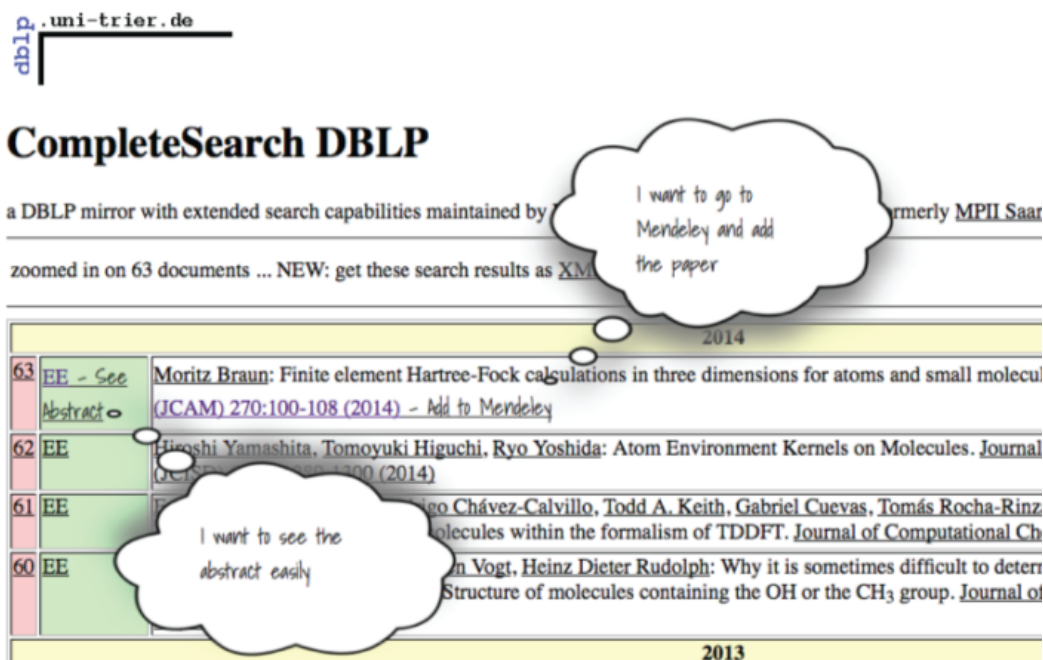


Figura 4.7: Primer definición del modelo MockPlug.

enfoque. En lugar de definir el requerimiento textualmente, o dibujándolo a mano alzada o utilizando herramientas de maquetado tradicional, Crowd-Mock le permite especificar su requerimiento de aumentación visual e interactivamente sobre el sitio web a aumentar. Como fue mencionado anteriormente, Pedro deberá complementar su modelo con una historia de usuario:

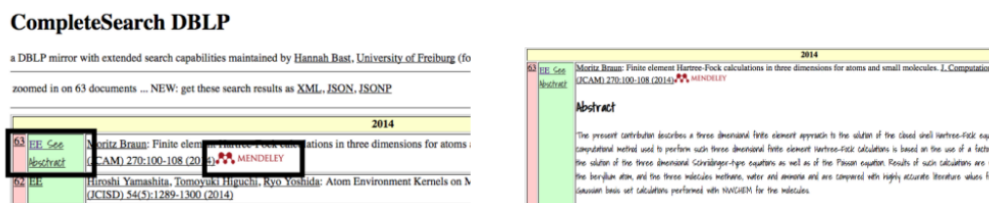
- En rol de: Investigador.
- Deseo: Agregar artículos a mi librería Mendeley fácilmente desde DBLP cuando los resúmenes de los artículos listados son relevantes.
- Para: poder obtener los artículos de mi interés actualizados y organizados.

Motivado por la historia de Pedro, otro usuario de DBLP llamado Juan, considera que sería más fácil realizar la tarea de Pedro, integrando Mendeley en DBLP en lugar de navegar a Mendeley y llenar el formulario con la infor-

mación del artículo. Con su idea en mente, Juan refina la historia de usuario original de Pedro:

- En rol de: Investigador.
- Deseo: Integrar Mendeley a DBLP para agregar artículos a la librería cuando los resúmenes son relevantes.
- Para: poder obtener los artículos de mi interés actualizados y organizados.

En este caso Juan no ha definido un nuevo modelo MockPlug, sin embargo, otro usuario de la comunidad llamado Jorge estuvo atento a las dos versiones del requerimiento y decidió colaborar en la especificación de un nuevo modelo MockPlug para la historia de Juan. De este modo, define el prototipo completo considerando ambos aspectos: el resumen y el formulario para agregar un artículo a la librería. Jorge, partiendo del modelo original definido por Pedro, ya tiene dos enlaces nuevos y los modifica de modo que en lugar de navegar a sitios webs externos, estos enlaces muestren/oculten otros componentes UI que fueron agregados por Jorge en relación a la historia de Juan. Así el enlace “ver resumen” muestra/oculta el resumen correspondiente, Figura 4.8a y 4.8b. El enlace “agregar a mendeley” muestra/oculta el formulario de Mendeley integrado en DBLP, Figura 4.9.



(a) Resultado de búsqueda en DBLP au- (b) “Ver resumen” muestra el resumen mentado con nuevos enlaces

Figura 4.8: Refinamiento definido por Jorge sobre DBLP.

En las metodologías ágiles, hay una sesión de refinamiento llamada *grooming backlog session* [17] que consiste en el refinado del *backlog* agregando,

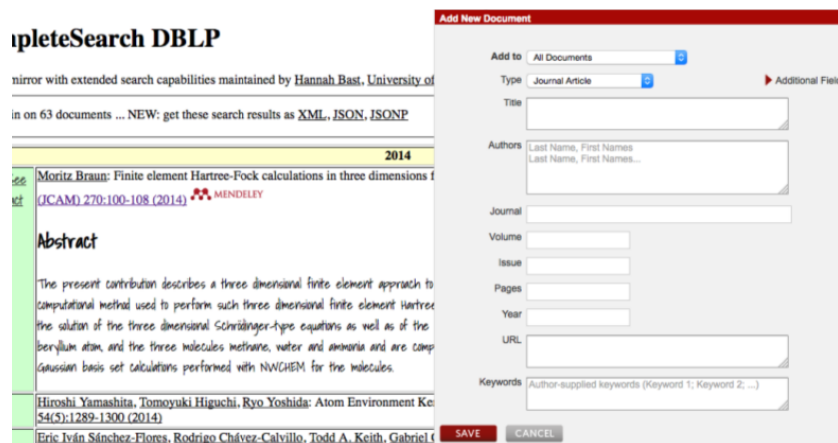


Figura 4.9: Refinamiento definido por Jorge sobre DBLP.(cont.)

removiendo o cambiando las tareas en él. La diferencia entre el enfoque propuesto y este tipo de sesiones es que en el caso de CrowdMock el refinado es colaborativo, en el sentido de que un usuario final, que puede también tener habilidades de programación, escribe un requerimiento y otro usuario distinto puede refinarlo. La misma persona puede realizar la definición completa, como es regular en los enfoques ágiles, donde el dueño producto es quien realiza esta actividad. Sin embargo, CrowdMock busca facilitar la colaboración en el refinamiento al mismo tiempo que es validado. En los enfoques ágiles la validación suele ocurrir en la sesión de validación, después de que el requerimiento es implementado. En CrowdMock la validación ocurre previo a la implementación, como es recomendado en la ingeniería clásica de requerimientos. La utilización de RAMs en la definición de los requerimientos provee el beneficio de poder validarlos desde muy temprano, ya que los interesados pueden visualizar la UI e incluso contemplar algún nivel interacción antes de que el artefacto sea implementado.

La Figura 4.10 muestra la evolución del requerimiento de Pedro. El árbol presentado en la figura muestra las distintas versiones del requerimiento como consecuencia de la actividad de refinamiento. Cada nodo representa una de las versiones que se encuentra etiquetada con el nombre del autor que contribuyó en su definición, excepto por la raíz cuya etiqueta es el nombre del requerimiento original. Como puede apreciarse existe una rama corres-

pendiente a la historia de usuario refinada por Juan y luego otra versión es alcanzada cuando el usuario Jorge que define el último modelo esta historia de usuario, presentada en las Figuras 4.8 y 4.9.

La Figura 4.10, muestra como el refinamiento realizado por Jorge, es

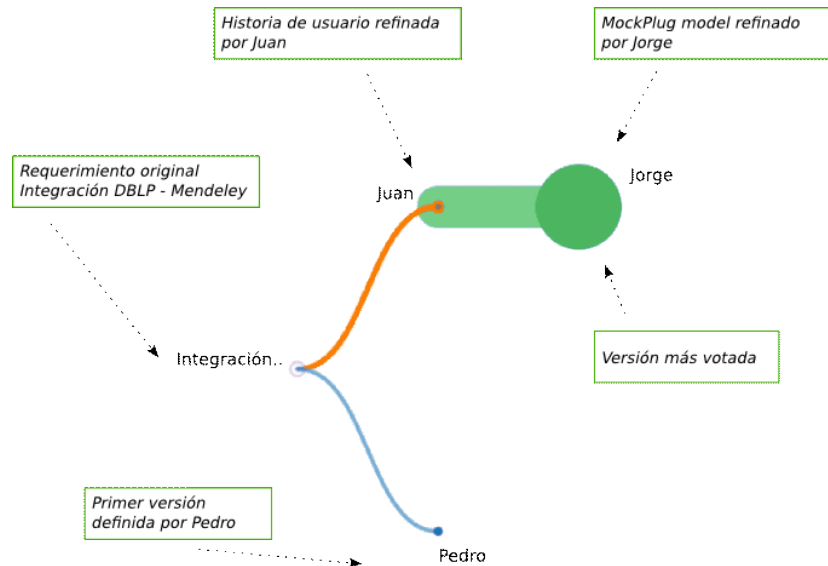


Figura 4.10: Evolución del requerimiento de Pedro.

destacado de los demás con un nodo más grande, por haber sido el más votado por la masa de usuarios en la actividad de priorización.

4.5. Herramientas para la definición de requerimientos

Las actividades para la definición, refinamiento y priorización de los requerimientos son soportadas por dos herramientas: MockPlug y UserRequirements. En esta sección se describen sus características principales y como funcionan juntas para dar soporte a los usuarios involucrados en su realización.

4.5.1. MockPlug

MockPlug es una herramienta de Aumentación Web desarrollada como un complemento para el explorador Firefox. Permite a los usuarios finales construir los prototipos RAMs (formalmente llamados modelos MockPlug) sobre sitios webs existentes. Usando esta herramienta, los usuarios finales pueden especificar sus requerimientos de aumentación en un contexto natural: el sitio web que desean que sea aumentado. En esta sección se describe MockPlug y sus modelos.

Una de las prioridades en el diseño de MockPlug, es proveer una herramienta que le permita a los usuarios ser utilizada para definir sus modelos, sin requerir conocimientos ni habilidades en programación. Un modelo MockPlug representa un conjunto de intenciones de modificación sobre un sitio web existente. Los usuarios materializan estas intenciones agregando y manipulando distintos tipos de widgets (componentes UI). Desde el punto de vista de los usuarios estos componentes UI tienen un estilo visual distintivo, aludiendo a un dibujo realizado a mano alzada, como es usual en herramientas como *Pencil*⁵ o *Balsamiq*⁶. De todos modos MockPlug trata a estos componentes como elementos ordinarios del DOM Tree, haciendo posible especificar fácilmente aspectos relativos a su futuro comportamiento y consiguiendo así la construcción de prototipos de alta fidelidad [33].

Al visitar un sitio web de interés, el usuario puede desplegar el menú principal de MockPlug. Esta simple acción habilita al usuario a especificar un requerimiento de aumentación sobre ese sitio web y el panel principal de la aplicación le ofrece una paleta de componentes UI predefinidos que el usuario podrá *arrastrar y soltar* sobre el sitio web visitado. La Figura 4.11 muestra el panel principal y una colección de componentes UI ofrecidos en la paleta. Esta misma figura muestra como el usuario ha agregado un botón “ver online” (y ha editado sus propiedades) en el detalle de una película en el sitio web

⁵<http://pencil.evolus.vn/>

⁶<https://balsamiq.com/>

IMDB. Como se ha mencionado, MockPlug manipula estos componentes UI como elementos reales del DOM Tree, donde una inserción de un componente UI tiene un efecto real sobre el sitio web para esa sesión de navegación.

Para mostrar dicho efecto en la Figura 4.12 se muestra la estructura

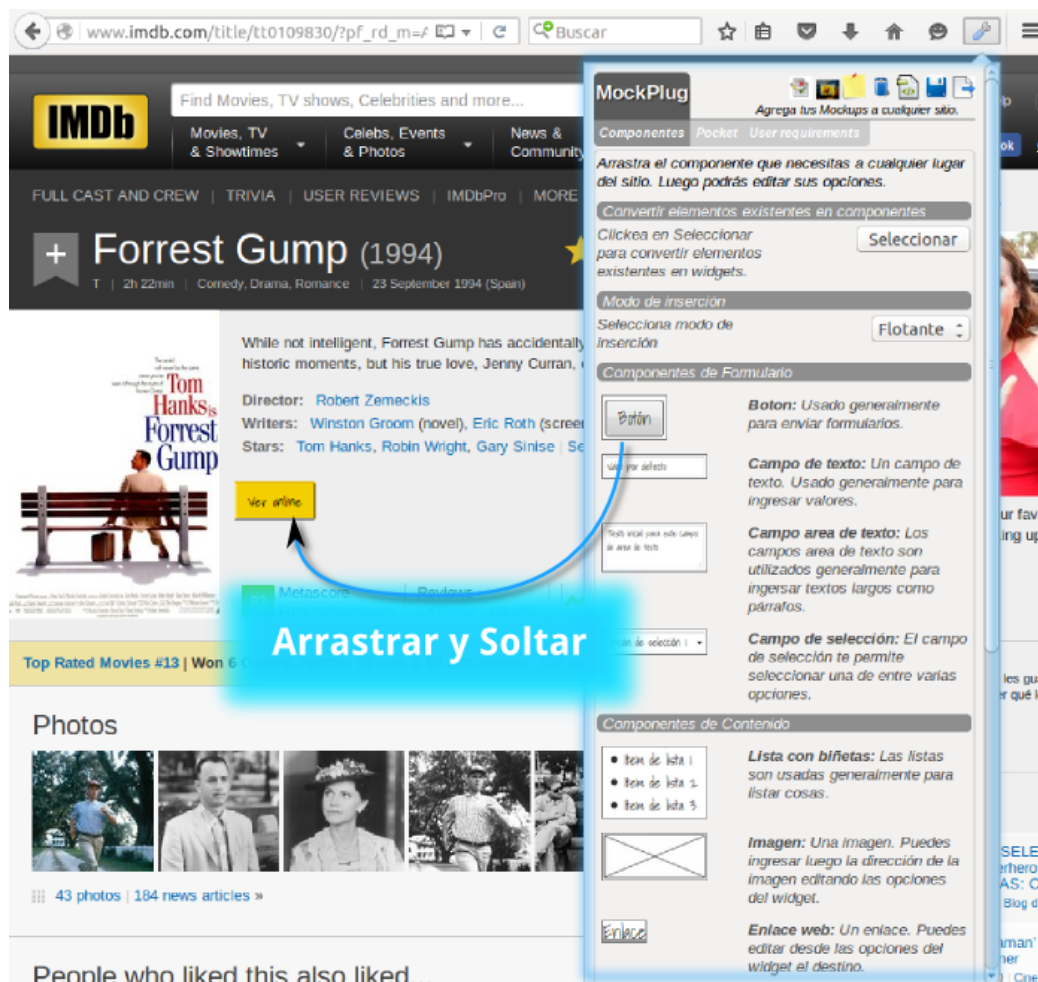


Figura 4.11: Paleta de componentes predefinidos en MockPlug.

del DOM Tree previo a la inserción del nuevo componente y después de que el usuario realiza la acción de *arrastrar y soltar*. A su vez, en la Figura 4.12 puede apreciarse que los elementos agregados con MockPlug tienen atributos propios de la herramienta. Aunque los elementos UI especificados con la he-

ramienta tengan un estilo distintivo, que le permite al usuario distinguirlos fácilmente de los elementos UI originales, es muy importante para el enfoque, destacar que estos elementos son simple elementos del DOM Tree, como cualquier otro elemento del sitio web original.

Los componentes UI del requerimiento pueden ser manipulados por el

```
DOM Tree original
<div class="plot_summary minPlotHeightWithPoster">
  <div class="summary_text" itemprop="description">While not intelligent, Forres...</div>
  > <div class="credit_summary_item"></div>
  > <div class="credit_summary_item"></div>
  > <div class="credit_summary_item"></div>
</div>

DOM Tree modificado con MockPlug
<div class="plot_summary minPlotHeightWithPoster">
  <div class="summary_text" itemprop="description">While not intelligent, Forres...</div>
  > <div class="credit_summary_item"></div>
  > <div class="credit_summary_item"></div>
  > <div class="credit_summary_item"></div>
  <button class="mockUp" title="Botón 1" abstract-widget="mpwidgetbutton" mockplug_widget_id="mpwidgetbutton5"
  style="cursor: default;">Ver online</button>
</div>
```

Figura 4.12: Efecto de inserción de componente UI en el DOM Tree.

usuario a través de un menú contextual desde el que puede realizar diferentes operaciones. La Figura 4.13 muestra como entre estas operaciones el usuario puede acceder al menú de edición de propiedades, quitar o mover el componente, clonarlo, etc.

En relación a las operaciones disponibles en el menú contextual del elemento agregado es propicio mencionar, que en el caso de que los componentes UI hayan sido agregados de modo *flotante*, es posible *fijar/soltar* su posición de modo relativo a otros elementos del sitio web en el que se especifica el requerimiento. Esto resulta útil para que otros usuarios con otras resoluciones o dimensiones de ventana del explorador, puedan encontrar ubicados los elementos en el mismo lugar que fueron agregados originalmente. En relación a la edición de propiedades y acciones, estas son relativas a la naturaleza del componente agregado. En las Figuras 4.14a y 4.14b se muestran los diálogos modales correspondientes a estas opciones del menú contextual para el caso de un botón como el mostrado en la Figura 4.13. Para el widget *Botón*, el usuario puede editar el texto del botón, el título y un nombre para futuras

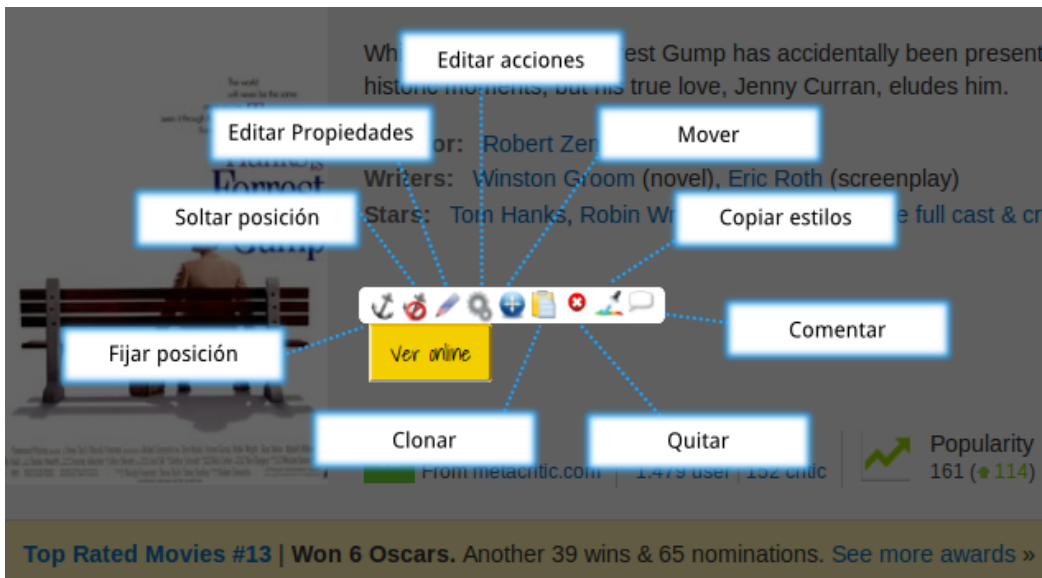


Figura 4.13: Menú contextual de un widget (componente UI).

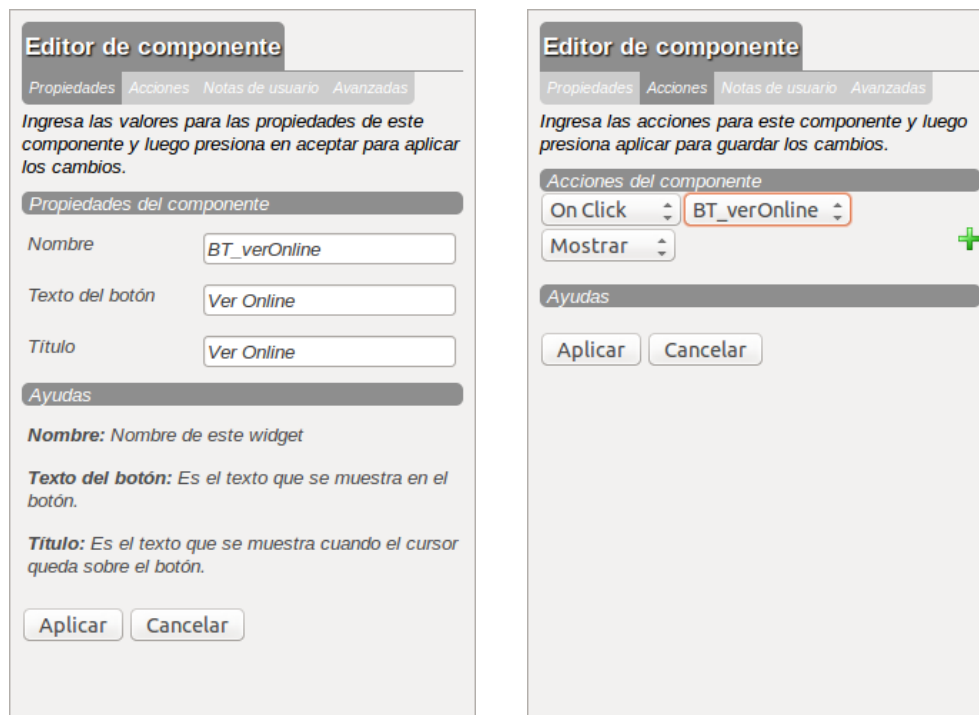
referencias.

4.5.2. Widgets

MockPlug ofrece la posibilidad de utilizar tres categorías de componentes UI: *predefined widgets* (predefinidos), *collected widgets* (colectados) y *pocket widgets* (de bolsillo) [33].

Los componentes UI predefinidos contemplan elementos como enlaces, botones, listas, cajas de texto, etc. y son ofrecidos en el panel principal de MockPlug. La paleta de componentes UI predefinidos se encuentra organizada en tres subcategorías: componentes para formularios, para contenido y de anotaciones (Tabla 4.1).

Además de los componentes predefinidos, el usuario puede crear componentes colectados. Esto es, el usuario puede *convertir* en un componente del requerimiento a cualquier elemento existente en el sitio web original. Esto se puede realizar utilizando la funcionalidad de colectado provista por Mock-



(a) Editor de propiedades

(b) Editor de acciones

Figura 4.14: Editores de propiedades y acciones.

Tabla 4.1: Categorías de componentes UI predefinidos.

Categoría	Objetivo	Ejemplos
Componentes para formularios	Especificar requerimientos que requieren ingreso de datos.	Text input, text área, select menu, botón
Componentes para Contenido	Especificar requerimientos que requieren componentes no editables.	Listas, imágenes, texto, enlaces
Anotaciones	Ofrecer mecanismos para describir requerimientos textualmente.	Notas adhesivas, burbujas de pensamiento.

Plug en su panel principal, que puede apreciarse en la Figura 4.11. A través de este tipo de componentes, el usuario puede especificar fácilmente requerimientos relativos a quitar o organizar elementos existentes en la UI original.

Como ejemplo, considere que un usuario del sitio *Cuevana2*⁷ quiere indicar que en lugar de ver las descripciones y el reparto de las series que sigue, desea que sean listados los episodios de la última temporada. El usuario puede convertir a componentes del requerimiento los elementos correspondientes a la descripción y el reparto de la serie. Como se muestra en la Figura 4.15, el usuario los ha colectado y además ha activado el sobresaltado de componentes de su modelo, con el objetivo de resaltar los componentes UI utilizados hasta el momento.

Una vez seleccionados los elementos de su interés, los quita a través de



Figura 4.15: Componentes UI colectados.

la opción *eliminar* de sus correspondientes menús contextuales, y luego en su lugar el usuario agrega el listado de episodios de su interés. El resultado

⁷<http://www.cuevana2.tv>

de esta serie de acciones es mostrado en la Figura 4.16.

De modo similar al de los componentes UI colectados en el sitio web



Figura 4.16: Componentes UI colectados y removidos.

sobre el que el usuario especifica su requerimiento, el usuario puede coleccionar cualquier componente UI de cualquier otro sitio web, con el objetivo de utilizarlo posteriormente en su requerimiento y poder expresar necesidades de integración entre los distintos sitios webs. Como en otro enfoque para el soporte de tareas de usuario [38], a este conjunto de widgets colectados en otros sitios webs, se los ha denominado *pocket widgets*, en alusión a un bolsillo que el usuario dispone para coleccionar elementos y utilizarlos luego. La pestaña correspondiente a este tipo de componentes es mostrada en la Figura 4.17. Cuando el usuario hace *click* sobre el botón *Colectar*, el sitio web es sobresaltado de modo que el usuario pueda distinguir el elemento que desea seleccionar del resto de los elementos existentes.

En la Figura 4.18a se muestra como el usuario esta a punto de seleccionar el enlace “citado por X” para agregarlo al bolsillo y poder utilizarlo luego en su requerimiento. En la Figura 4.18b se muestra el enlace colectado ya en el bolsillo y listo para ser utilizado.

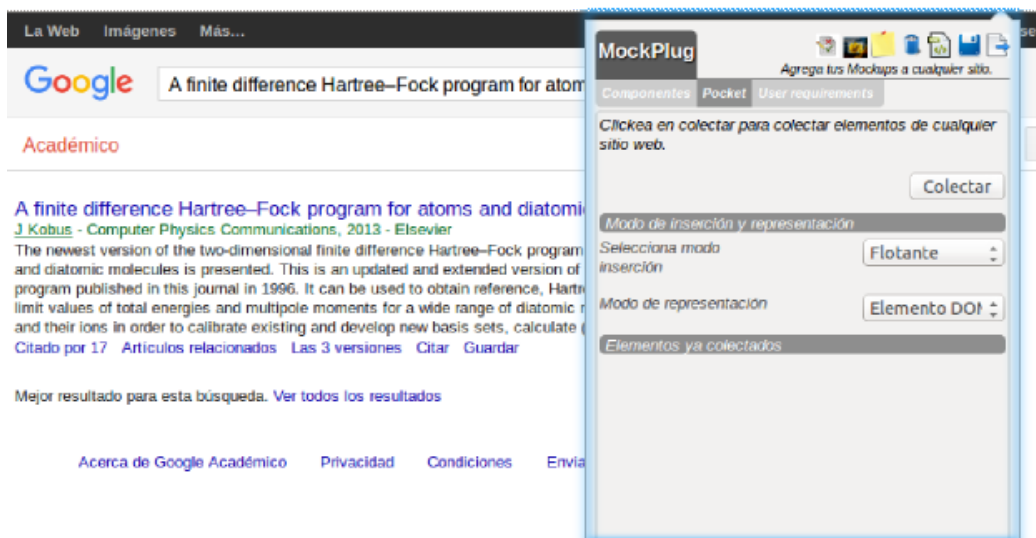
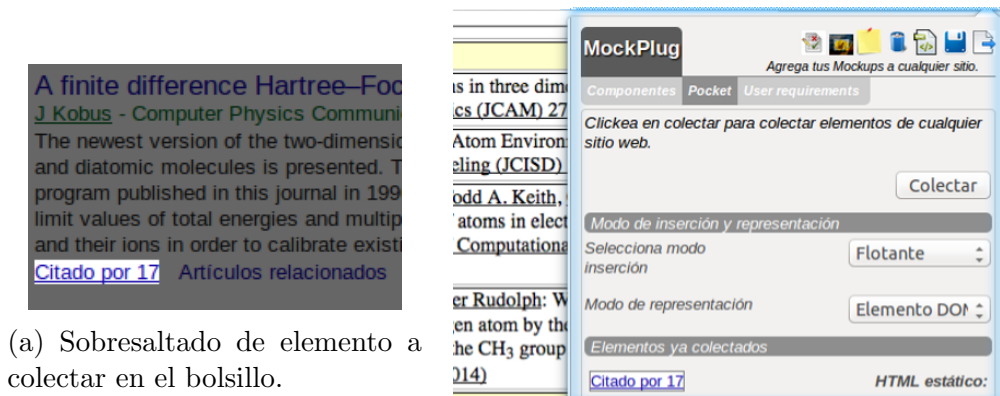


Figura 4.17: Pestaña Pocket en MockPlug.

Una vez colectados, estos componentes UI pueden ser arrastrados y sol-



(a) Sobresaltado de elemento a colectar en el bolsillo.

(b) Elemento colectado en el Pocket

Figura 4.18: Colectando componente UI en el Pocket.

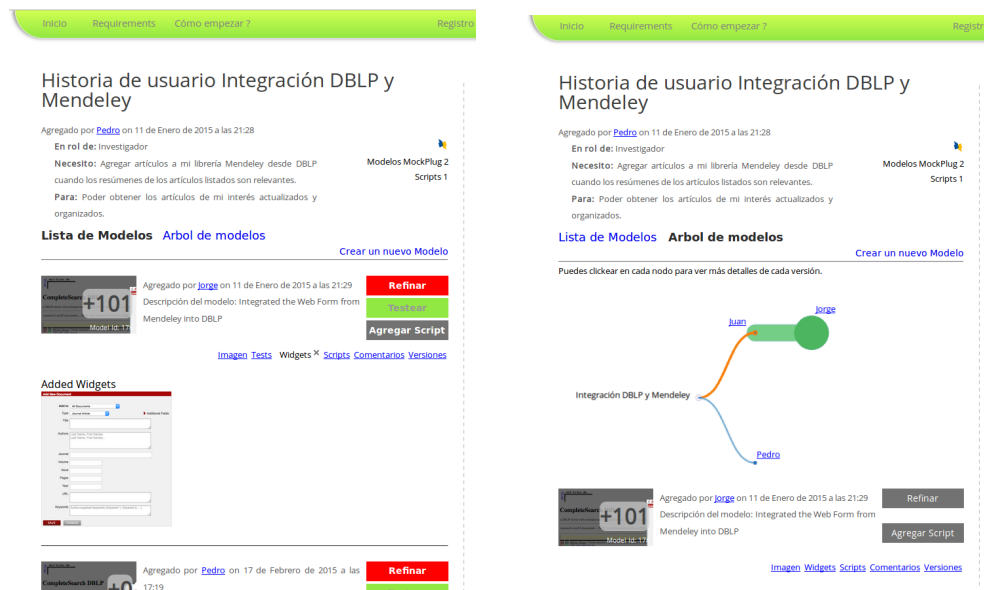
tados al sitio web de interés del mismo modo que los widgets predefinidos. Sin embargo, por tener estos componentes UI, estilos y comportamiento relativo al sitio web de origen, la correcta visualización sobre el sitio web al que resultaren agregados no puede ser garantizada. Así es que MockPlug facilita

la incorporación de estos elementos al requerimiento de dos maneras: como elemento del DOM Tree o como una imagen estática tomada del sitio web de donde provienen.

4.5.3. UserRequirements

UserRequirements es una implementación del repositorio de artefactos de aumentación requerido por el enfoque. Provee funcionalidad para gestionar los requerimientos CrowdMock descritos con un RAM y una historia de usuario, permitiéndole a la masa de usuarios, compartir y evolucionar sus requerimientos, priorizarlos y comentarlos. La Figura 4.19 muestra muestra diferentes vistas de un requerimiento en el repositorio.

En la Figura 4.19, puede apreciarse como cada refinamiento tiene un



(a) Vista de requerimiento: listado de modelos. (b) Vista de requerimiento: árbol de evolución.

Figura 4.19: Detalles de un requerimiento en UserRequirements.

número de votos y un conjunto de componentes. Cualquier usuario del repo-

ditorio puede ver la imagen del modelo MockPlug con los componentes UI sobresaltados. Sin embargo, cuando el usuario tiene instalado MockPlug en su explorador, puede reproducir y/o refinar los modelos existentes o crear uno nuevo. La lista de refinamientos es ordenada por votos, mostrando primero los refinamientos más votados por la masa de usuarios.

Con el objetivo de facilitar a los usuarios la rápida comprensión de la evolución del requerimiento, se ofrece la vista del árbol de modelos, que permite distinguir rápidamente la versión más votada y más discutida. En relación a ello, hay varios aspectos a considerar para poder comprender esta representación. Cada usuario involucrado tiene un color distinto en el gráfico y el tamaño de los nodos es relativo a la cantidad de votos obtenidos por cada versión, donde los nodos más grandes tienen mayor cantidad de votos. Los nodos pueden tener un borde de distinto color que representa el nivel de discusión (cantidad de comentarios) correspondiente a cada versión. Mientras más grande sea este borde, más comentado ha resultado el refinamiento (en términos relativos a los otros nodos). Cuando el borde de un nodo es verde, significa que el usuario original aceptó el refinamiento y si fuera rojo, significa que la ha rechazado. En la Figura 4.20, se muestra un árbol de evolución más complejo, donde puede apreciarse que el refinamiento realizado por Angie es el más votado, el más discutido y que Pedro lo ha aceptado. A diferencia del refinamiento realizado por Teresa, que habiendo sido discutido no obtuvo votos y fue rechazado por Pedro.

4.5.4. Integración MockPlug y UserRequirements

La integración de las herramientas MockPlug y UserRequirements es muy importante para poder agilizar y facilitar las actividades que la masa de usuarios debe realizar. Para crear un nuevo requerimiento en UserRequirements, esto es la definición de una historia de usuario y un modelo MockPlug, un usuario final puede utilizar MockPlug sobre cualquier sitio web de su interés y luego completar la historia de usuario a fines de compartir su requerimiento con el resto de la masa de usuarios. La Figura 4.21a muestra

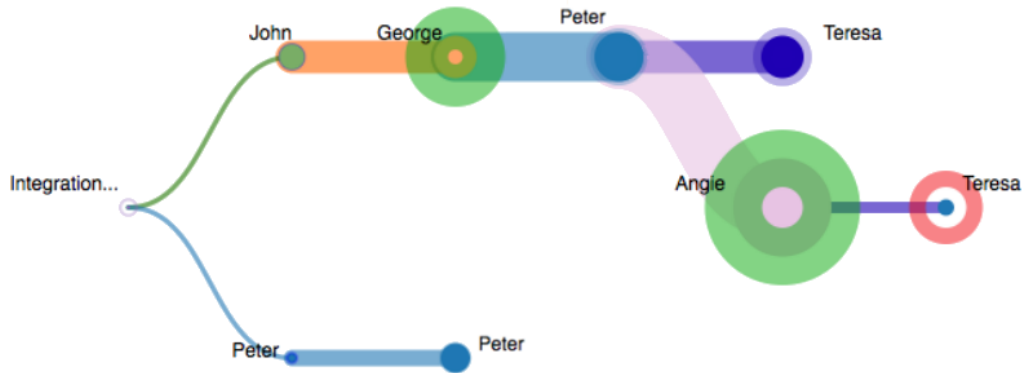
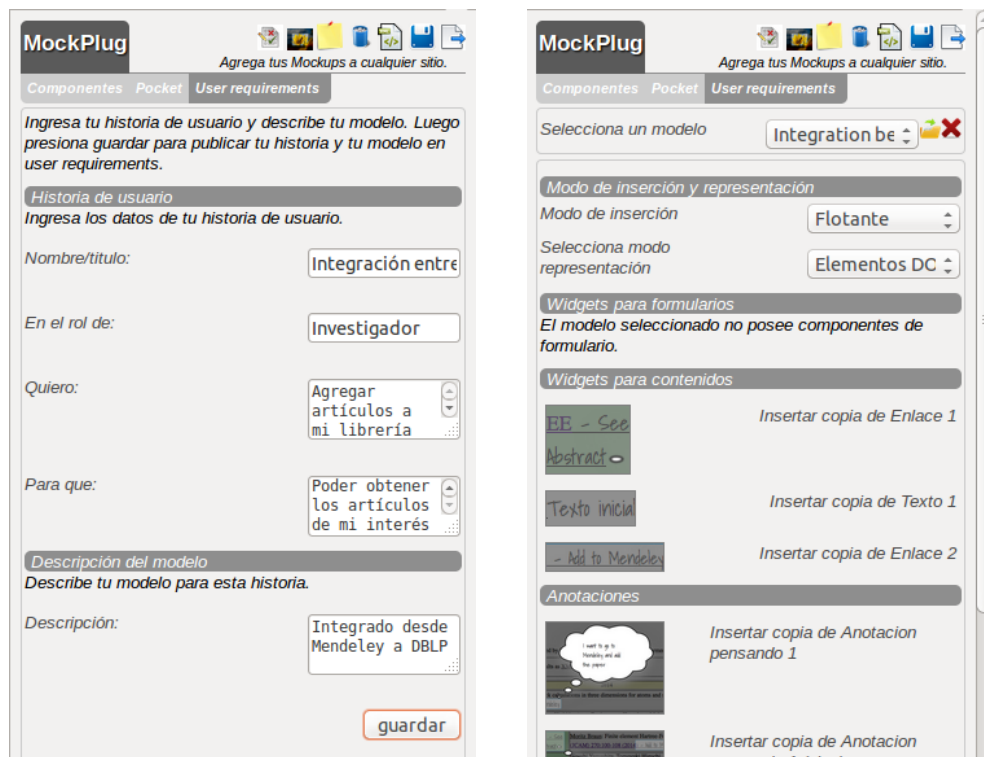


Figura 4.20: Evolución de requerimiento.

como desde la pestaña UserRequirements, el usuario puede realizar esta tarea. Así mismo se distingue la posibilidad de que un usuario pueda reutilizar componentes UI que ya ha definido en otros modelos MockPlug.

La Figura 4.21b muestra como el usuario puede seleccionar un modelo desde la pestaña UserRequirements y luego reutilizar sus componentes arrastrándolos y soltándolos del mismo modo que con los componentes predefinidos. Así mismo, al seleccionar un modelo desde el menú de requerimientos que el usuario ha guardado en UserRequirements, el usuario puede indicar que desea abrirlo en una nueva pestaña, ya sea con el objetivo refinarlo, o simplemente volver a verlo.



(a) Guardar modelo en UserRequirements desde MockPlug. (b) Reutilización de componentes UI de otros modelos.

Figura 4.21: Integración MockPlug y UserRequirements.

Capítulo 5

Desarrollo y prueba de artefactos de aumentación

Una vez que el requerimiento es descubierto y definido por la masa de usuarios finales, los usuarios con habilidades de programación interesados completan su funcionalidad. Para ello, contarán con la posibilidad de refinar el requerimiento considerando cualquier aspecto que los usuarios sin habilidades de programación puedan no haber advertido relevante y luego contarán con una versión inicial e incompleta del artefacto, derivada automáticamente del prototipo.

El enfoque propone la ejecución temprana de los casos de prueba entre los usuarios finales y los desarrolladores, de modo que éstos puedan advertir si los artefactos de su interés y los productos intermedios en los que han participado, necesitan su revisión.

Para lograr estos objetivos, el enfoque se basa la definición de un meta-modelo de requerimientos de Aumentación Web, que da lugar a la derivación de artefactos intermedios a través del procesamiento de sus instancias, como es común en los enfoques dirigidos por modelos [46]. En este capítulo se presenta el metamodelo CrowdMock y se describen en mayor detalle los aspectos relativos a la construcción de los artefactos de aumentación y sus correspondientes casos de prueba.

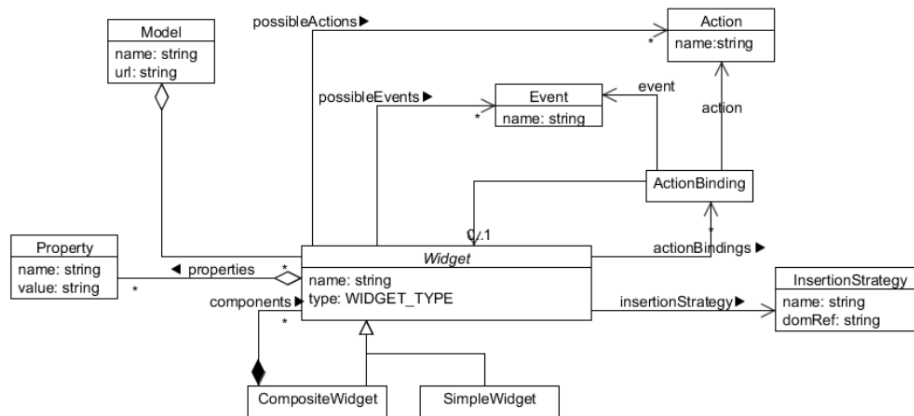


Figura 5.1: Metamodelo CrowdMock.

5.1. Metamodelo y generación de código

Como ha sido mencionado, los prototipos RAM utilizados al definir los requerimientos de aumentación Web, son instancias del metamodelo CrowdMock, que es presentado en la Figura 5.1. Estas instancias son procesadas con el objetivo de generar código fuente y casos de prueba automáticamente, que los usuarios desarrolladores pueden completar en la funcionalidad que no fuere posible expresar con las herramientas provistas.

Cada RAM definido por la masa de usuarios constituye, en términos formales, un modelo MockPlug que es instancia del metamodelo CrowdMock. Como puede ser apreciado en la Figura 5.1, estos modelos tienen un nombre y una URL, que es la URL propia del sitio web donde fueron definidos. Así mismo definen un conjunto de widgets, donde cada widget tiene un nombre, un tipo y un conjunto de propiedades asociadas.

Para que estos widgets puedan ser agregados al sitio web que se encuentra siendo aumentado, resulta necesario indicar cómo serán agregados al DOM Tree original. Para ello, el metamodelo CrowdMock, define la entidad *InsertionStrategy*, de modo que cada widget perteneciente a un modelo, contendrá una estrategia de inserción asociada (de modo flotante, antes de, después de, dentro de, etc) en donde un elemento del DOM Tree original es

referenciado. Así mismo, cada widget es capaz de responder a un conjunto de eventos y de realizar una serie de acciones, dando lugar a la definición de acciones a realizar cuando estos eventos ocurren.

La Figura 5.2a muestra el esquema (*XML Scheme*) que describe los modelos MockPlug. Cada instancia de modelo MockPlug, puede ser exportada a XML, como se muestra en la Figura 5.2b.

```

-<xsd:schema elementFormDefault="qualified" version="1.0">
  <xsd:element name="xml" type="xmlType"/>
  <xsd:complexType name="xmlType">
    <xsd:sequence>
      <xsd:element name="MockPlugModel" type="MockPlugModelType"/>
    </xsd:sequence>
  </xsd:complexType>
  <xsd:complexType name="MockPlugModelType">
    <xsd:sequence>
      <xsd:element name="name" type="xsd:string"/>
      <xsd:element name="description" type="xsd:string"/>
      <xsd:element name="title" type="xsd:string"/>
      <xsd:element name="url" type="xsd:string"/>
      <xsd:element name="widgets" type="widgetsType"/>
    </xsd:sequence>
  </xsd:complexType>
  <xsd:complexType name="widgetsType">
    <xsd:sequence>
      <xsd:element maxOccurs="unbounded" name="widget" type="widgetType"/>
    </xsd:sequence>
  </xsd:complexType>
  <xsd:complexType name="widgetType">
    <xsd:sequence>
      <xsd:element name="name" type="xsd:string"/>
      <xsd:element name="properties" type="propertiesType"/>
      <xsd:element name="abstractWidget" type="abstractWidgetType"/>
      <xsd:element name="insertionStrategy" type="insertionStrategyType"/>
    </xsd:sequence>
  </xsd:complexType>
  <xsd:complexType name="insertionStrategyType"></xsd:complexType>
  <xsd:complexType name="attachToType"></xsd:complexType>
  <xsd:complexType name="abstractWidgetType"></xsd:complexType>
  <xsd:complexType name="propertiesType"></xsd:complexType>
</xsd:schema>

```

```

<xml>
-<MockPlugModel>
  <name>IMDB - Ver películas online</name>
  <description>
    Agrega un botón en la ficha de una película para buscarlas online.
  </description>
  <url>
    http://www.imdb.com/title/tt0109830/?pf_rd_m=A2FGELUUNOQJNL&pf_rd_p=2396
  </url>
  <widgets>
    <widget>
      <name>BT_verOnline</name>
      <properties>
        <name>BT_verOnline</name>
        <text>Ver Online</text>
        <title>Ver Online</title>
      </properties>
      <abstractWidget>
        <name>button</name>
        <id>mpwidgetbutton</id>
        <humanReadable>Boton</humanReadable>
        <description>Usado generalmente para enviar formularios.</description>
      </abstractWidget>
      <insertionStrategy>
        <name>floating</name>
        <top>369</top>
        <left>411</left>
        <domReference>id('title-overview-widget')/DIV[3]/DIV[2]/DIV[1]</domRefer
      </insertionStrategy>
    </widget>
  </widgets>
</MockPlugModel>
</xml>

```

(a) XML Scheme modelos MockPlug.

(b) Versión XML de un modelo MockPlug.

Figura 5.2: XML Scheme y XML de un modelo MockPlug.

Finalmente en la Figura 5.3, puede apreciarse el código generado a través de las herramientas haciendo uso de hojas de transformación *XSLT*. El usuario obtiene una versión inicial del artefacto que debe completar para implementar la funcionalidad deseada, sin embargo actividades repetitivas como la definición de la cabecera del propio artefacto de aumentación y la construcción de los elementos pudo ser automatizada.

En el contexto de los artefactos de Aumentación Web, no es posible garantizar que el código fuente generado automáticamente resulte ser siempre adoptado por los desarrolladores. Este problema afecta a los enfoques dirigidos por modelos en general [83].

```

// ==UserScript==
// @name      IMDB - Ver películas online
// @description  Agrega un botón en la ficha de una película para buscarlas online
// @include   http://www.imdb.com/title/*
// @require   http://localhost:8000/medios/js/cmfw/cmfw.js
// @grant     GM_xmlHttpRequest
// @grant     GM_log
// ==/UserScript==

$("body").cmfw({debug:true});
$("body").cmfw("addAugmenter", {model:idModelo, callback: init});

function init(){
    var aumentador = $("body").cmfw("getAugmenter", idModelo);
    aumentador.insertWidgets();
    var BT_verOnline = aumentador.getWidgetByName("BT_verOnline");
    //TODO: hacer algo con el botón BT_verOnline
};

```

Figura 5.3: Código fuente generado.

Aunque el enfoque y las herramientas que lo implementan intentan facilitar las tareas relativas a la construcción del código fuente del artefacto propiamente dicho, las herramientas permiten la exportación de ambos, del código fuente y de la versión XML de los modelos. Se han implementado tres hojas XSLT de transformación que le permiten al usuario desarrollador obtener la plantilla del artefacto en tres versiones diferentes: JavaScript nativo, utilizando jQuery y utilizando el framework CMFW (Figura 5.4a).

5.2. MockPlug para desarrolladores

Aunque los desarrolladores pueden hacer uso de la misma funcionalidad que los usuarios finales disponen a través de la herramienta MockPlug e involucrarse en el refinamiento de los requerimientos, fue considerada la posibilidad de otorgarles dos funcionalidades avanzadas a la hora de utilizar esta herramienta.

Al momento de editar las propiedades de un componente del modelo MockPlug, el desarrollador dispone de una pestaña de opciones avanzadas, que le permite agregar comportamiento y anotaciones al componente seleccionado, Figura 5.4b.

Las anotaciones hacen posible definir aspectos relativos a la complejidad de lo que deberá implementarse a la hora de completar el artefacto. Por ejemplo, si el requerimiento implica obtener imágenes de un sitio distinto al sitio web que resulta estar siendo aumentado, entonces el desarrollador puede indicar, que ante un evento sobre un componente, debe invocarse una funcionalidad que la hora de su implementación deberá realizar una solicitud asincrónica a un sitio web distinto del aumentado, anotándolo como `XMLHttpRequest(cross-origin)`.

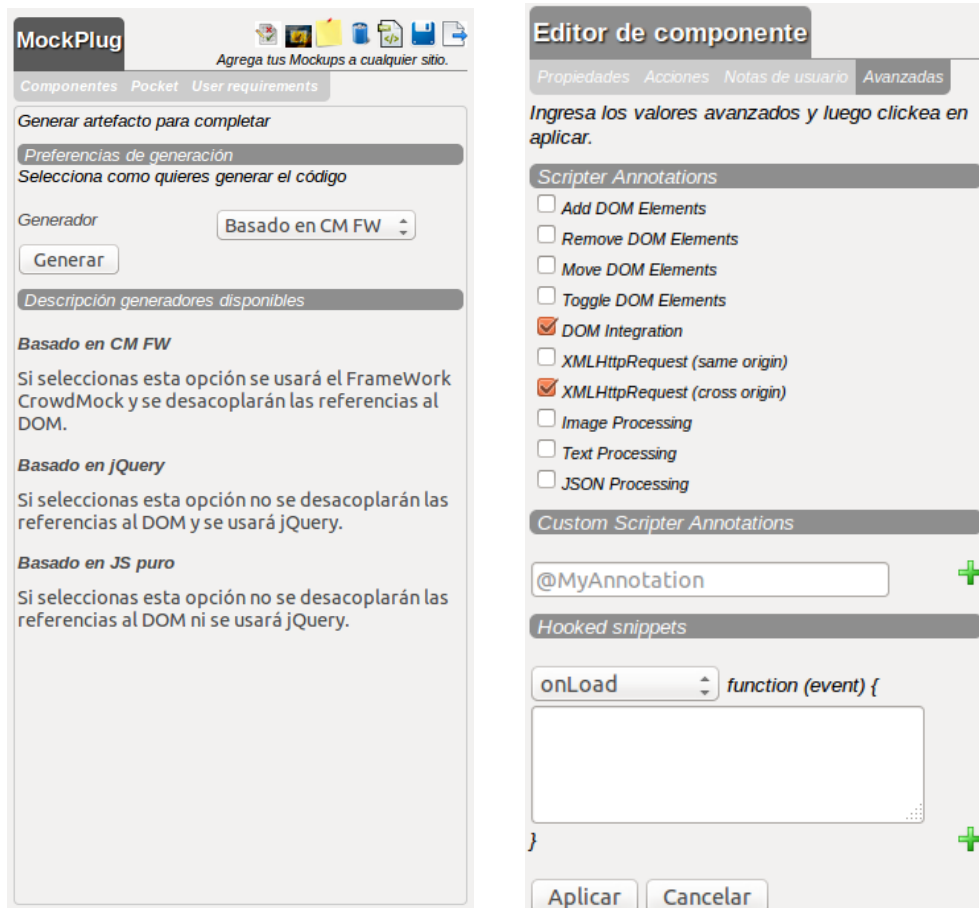
Adicionalmente, si el comportamiento requerido por la acción a realizar, al momento de definir el modelo, ya estuviera disponible por parte del desarrollador, es posible incorporarlo tempranamente a través de la posibilidad que la herramienta le ofrece de incorporar *snippets* (porciones de código fuente) al modelo que está refinando.

5.3. CrowdMock Framework

Con el objetivo de proveerles herramientas a los desarrolladores que puedan ser provechosas para el enfoque, se provee el CrowdMock Framework (CMFW) que permite al desarrollador la implementación de los artefactos de Aumentación Web enfocándose en el comportamiento a completar y delegando tareas de construcción y configuración de componentes a la librería.

CMFW se basa en extensiones de la librería jQuery y haciendo uso de jQuery Plugins, se provee una clase para cada tipo de componente que los usuarios finales disponen para especificar sus modelos.

Para poder ser utilizado, el desarrollador solo debe importar una librería JS e instanciar un artefacto indicando el modelo MockPlug que desea completar. El framework, se encarga de la construcción de los componentes, de su inserción y de asignarles sus propiedades. Una característica importante, es que provee la capacidad de agregar los componentes al sitio web aumentado utilizando las referencias de los modelos MockPlug de modo tal, que si las referencias DEOI cambian en el modelo, porque este resultare refinado o mantenido, estos cambios impactan automáticamente en el artefacto, por



(a) Generador de código. (b) Opciones avanzadas de componente.

Figura 5.4: MockPlug para desarrolladores.

estar estas referencias desacopladas del mismo.

En la Figura 5.5 se muestra el código fuente generado para un artefacto muy sencillo. En la indicación 1 de la figura, se agrega al cuerpo del documento aumentado el modelo de interés. Esta acción implica que el framework descargará la versión del modelo del repositorio. Una vez descargado, se invoca a la función `init`, en dónde el desarrollador ha obtenido una referencia al aumentador y ha insertado sus widgets al DOM Tree. Luego puede obtener

```

// ==UserScript==
// @name      IMDB - Ver películas online
// @description  Agrega un botón en la ficha de una película para buscarlas online
// @include   http://www.imdb.com/title/*
// @require   http://localhost:8000/medios/js/cmfw/cmfw.js
// @grant     GM_xmlhttpRequest
// @grant     GM_log
// ==/UserScript==

$("body").cmfw({debug:true});
$("body").cmfw("addAugmenter", {model:idModelo, callback: init}); 1

function init(){
  2 var aumentador = $("body").cmfw("getAugmenter", idModelo);
  aumentador.insertWidgets();
  var BT_verOnline = aumentador.getWidgetByName("BT_verOnline"); 3
  //TODO: hacer algo con el botón BT_verOnline
};

```

Figura 5.5: Código fuente generado basado en CMFW.

una referencia al widget de su interés a través de su nombre para manipularlo. Note que el desarrollador nunca tuvo que especificar las propiedades del botón ni conocer las referencias al DOM Tree en dónde debía agregarse. Simplemente delega en el CMFW, la descarga del modelo y la construcción e inserción de los elementos.

5.4. Testeo y casos de prueba

Como fue estudiado en el capítulo 2, la naturaleza desintegrada de los artefactos de Aumentación Web respecto del sitio web aumentado, en suma a la falta de un proceso para la construcción de este tipo de artefactos atentan contra su calidad y mantenimiento a través del tiempo. Los artefactos de aumentación son distribuidos a los usuarios tal como sus desarrolladores los crean y no existe forma de saber si un artefacto se encuentra en estado defectuoso o no, hasta que es ejecutado. A su vez, cuando los artefactos de aumentación por algún motivo fallan en su ejecución, no existe ningún tipo de soporte que permita a los usuarios interesados conocer y difundir el problema de un modo automatizado. El reporte de los fallos de los artefactos de aumentación queda a voluntad de sus usuarios, que en el caso de advertir

que un artefacto falla, podrán a lo sumo dirigirse a la comunidad y comunicárselo a su desarrollador a través de un foro de discusión. Mientras no sea mantenido, el artefacto defectuoso seguirá estando disponible para el resto de la comunidad, que desprevénidamente podrá instalarlo y ejecutarlo con consecuencias impredecibles.

CrowdMock intenta reducir la posibilidad de que los usuarios finales utilicen artefactos de aumentación defectuosos, a la vez de que provee herramientas para facilitar la detección de posibles fallas antes de que el artefacto sea implementado y distribuido entre los usuarios finales interesados. Así mismo se provee la posibilidad de que los fallos en las ejecuciones de los artefactos sean conocidos por todos los involucrados, de modo que la comunidad pueda descubrir y conocer el estado de los artefactos, sin depender exclusivamente de la voluntad de una única persona.

Para lograr estos objetivos, el enfoque incorpora prácticas comunes del desarrollo de software en el marco de las metodologías ágiles, como lo son el testeo y la integración continua [39]. Entre los principios fundamentales del testeo ágil [49] se destacan a continuación aquellos en los que se ha hecho hincapié, habiéndoselos considerado particularmente relevantes en CrowdMock:

- El testeo debe ser realizado en etapas tempranas y debe ser paralelo al desarrollo de los artefactos.
- Los usuarios finales interesados deben ser incluidos en el proceso de testeo.
- Los testeos deben poder realizarse en cada iteración.
- El testeo debe ser automatizado.

En relación a los principios fundamentales de los sistemas que dan soporte a la integración continua [74], se han considerado los siguientes especialmente importantes para el enfoque en general y para la construcción de este tipo de software en particular, que como se ha mencionado anteriormente, es desde su propia naturaleza desintegrada, muy susceptible a estos aspectos:

- Mantener un único repositorio.
- Testear en ambientes de producción.
- Hacer fácil para cualquier usuario obtener la última versión de los artefactos.
- Todos los usuarios pueden conocer el estado de los artefactos.

5.4.1. Derivación y ejecución de casos de prueba

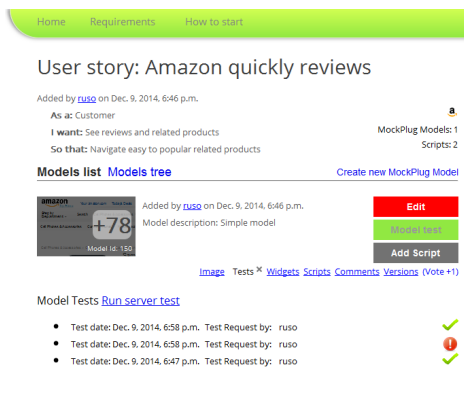
Distintos casos de prueba son derivados del modelo MockPlug del requerimiento original. Estos casos de prueba son derivados tanto para el modelo MockPlug en sí mismo como para los artefactos que los implementen.

En el caso de los casos de prueba para los modelos MockPlug, el testeo se reduce a comprobar que el modelo es válido y se encuentra vigente, simplemente validando cada una de las referencias DEOI utilizadas en el modelo.

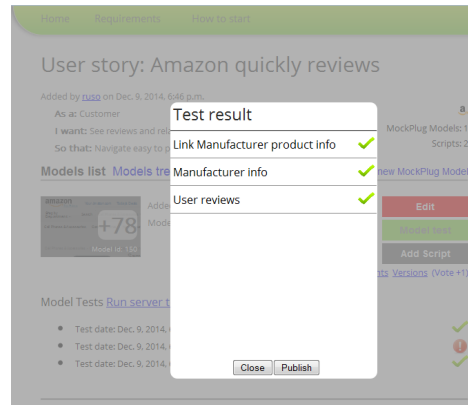
La ejecución de estos casos de prueba puede ser realizada por cualquier usuario y la publicación de un resultado no satisfactorio podrá ser compartida con toda la comunidad para su posterior utilización en la prevención de ejecución de los artefactos correspondientes. En la Figura 5.6a se muestran los resultados publicados para un modelo MockPlug en particular y en la Figura 5.6b se muestra como un usuario interesado ha ejecutado los casos de prueba para dicho modelo.

Los casos de prueba para los artefactos de aumentación que implementan los modelos MockPlug, pueden ser completados por los desarrolladores dentro del mismo artefacto, aunque distintos casos de prueba son derivados del modelo para ser testeados posteriormente a la ejecución del artefacto:

- ¿Todos los componentes especificados en el modelo fueron incluidos una vez que el artefacto fue ejecutado?
- ¿Los componentes tienen las propiedades indicadas en el modelo Mock-



(a) Resultados publicados.

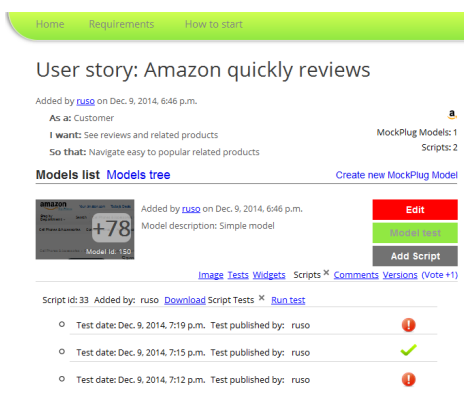


(b) Ejecución Test de modelo.

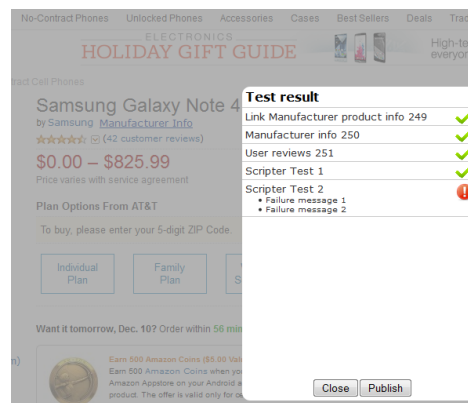
Figura 5.6: Testeo de modelo MockPlug.

Plug?

La Figura 5.7a muestra los resultados de las ejecuciones de los casos de prueba para aun artefacto en particular, mientras la Figura 5.7b muestra la ejecución de los casos de prueba para el mismo artefacto, por parte de un usuario de la comunidad.



(a) Resultados publicados.



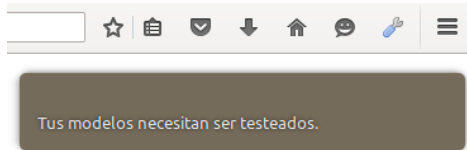
(b) Ejecución Test de artefacto.

Figura 5.7: Testeo de artefacto de aumentación que implementa un modelo MockPlug.

En la Figura 5.7b se distingue que fueron ejecutados casos de prueba agregados por el desarrollador. Esto es posible debido a que, como se ha mencionado, en la implementación del artefacto el desarrollador puede agregar todos los casos de prueba que considere adecuados.

5.4.2. Integración

Retomando el ejemplo del usuario Pedro visto en el capítulo 4, acerca de una historia de usuario de integración entre DBLP y Mendeley, mientras DBPL mantuvo su versión clásica los usuarios canalizaron su necesidad de aumentación, definiendo el requerimiento, el modelo, y obteniendo finalmente su artefacto de aumentación. Pasado un tiempo, DBLP cambió completamente su sitio web mejorando la presentación de su contenido con un estilo visual más moderno y responsivo. Consecuencia de ello, la historia de usuario en la que se habían involucrado Pedro, Juan, Jorge y algunos otros usuarios, quedó desactualizada, y todos los productos alcanzados en relación a ella quedaron defectuosos. Consecuencia de ello la masa de usuarios publica ejecuciones fallidas de los casos de prueba para dichos artefactos, y al usuario Pedro, le es notificado en su explorador (Figura 5.8a). Como se muestra en la Figura 5.8b, Pedro ejecuta los casos de prueba de sus modelos, y finalmente advierte que su historia se encuentra desactualizada y tiene oportunidad de recuperarla, como se muestra en la Figura 5.9.

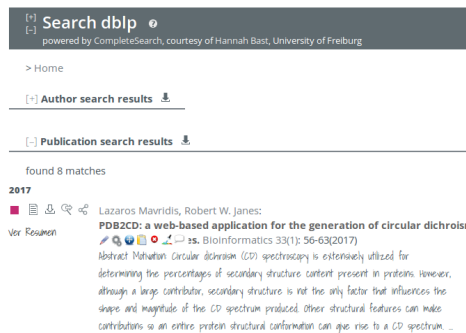


(a) Notificación Test requerido.

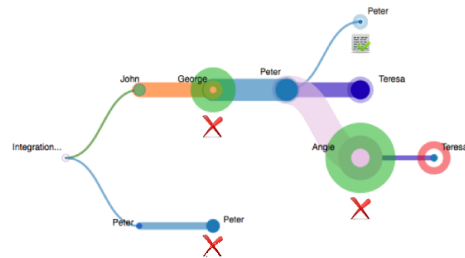


(b) Ejecución de Casos de Prueba.

Figura 5.8: Integración y Testeo en MockPlug.



(a) Nuevo refinamiento.



(b) Nuevo árbol de evolución.

Figura 5.9: Integración y Testeo en MockPlug. (cont.)

Capítulo 6

Distribución y mantenimiento

En el capítulo 3 fue presentado el enfoque negociado propuesto, Crowd-Mock, que involucra a los dueños de los sitios webs, a los desarrolladores de artefactos de aumentación y a los usuarios finales. Hasta aquí fueron presentados en mayor detalle los aspectos del enfoque relativos a la definición de los requerimientos de aumentación por parte de la masa de usuarios, a su implementación por parte de los desarrolladores y al testeo de los artefactos por parte de ambos.

En este capítulo se presentan en mayor detalle los aspectos relativos a la distribución de los artefactos de Aumentación Web entre los usuarios finales y su posible participación en el mantenimiento. En particular, es en la distribución de los artefactos de Aumentación Web, donde son involucrados los dueños de los sitios webs aumentados. Así mismo, el enfoque propone que los usuarios finales puedan involucrarse en el mantenimiento de los artefactos una vez que estos son distribuidos y se encuentran disponibles para ser utilizados. En este capítulo se describen cómo estos actores son involucrados en el enfoque y el soporte que las herramientas les proveen para llevar a delante sus actividades.

6.1. Distribución de artefactos de Aumentación Web

En relación al estudio realizado sobre las comunidades de Aumentación Web, resulta oportuno distinguir que los usuarios finales de los artefactos de aumentación de este tipo de comunidades, conocen a priori la posibilidad de aumentar un sitio web para alcanzar sus objetivos. Sin embargo, es importante destacar que no todos los usuarios de los sitios webs conocen estas posibilidades. Independientemente de ello, para descubrir la existencia de un artefacto, los usuarios deben navegar los repositorios de estas comunidades buscando aquellos artefactos que pudieran dar solución a su necesidad. Si bien estas actividades pueden ser realizadas sin ningún inconveniente, el enfoque propone que todos los visitantes de un sitio web puedan descubrir artefactos de aumentación desde el mismo sitio web que se encuentran navegando. Para hacer más claro el concepto, esto puede ser considerado como otorgar la posibilidad a cada sitio web, de disponer de un mercado de artefactos de aumentación que sus usuarios pueden acceder en busca de artefactos, justo al mismo momento en que visitan el sitio web en sí mismo.

Con este objetivo presente, en donde el enfoque propone que todos los visitantes de un sitio web puedan utilizar los artefactos de aumentación que fueron aprobados/certificados por su dueño, la distribución de estos artefactos de aumentación entre los visitantes, en el marco del proceso de adaptación negociada, requiere la introducción de diversos componentes de aplicación, para los cuales los siguientes aspectos fueron considerados muy importantes:

- **Fácil instalación:** la instalación de los componentes de aplicación a incorporar por parte de los dueños de los sitios webs debe ser tan simple como sea posible.
- **Customizable:** el aspecto de los componentes de aplicación debe poder ser customizable por parte de los dueños de los sitios webs.
- *Plug and Play:* desde el punto de vista de los usuarios finales, la actividad de seleccionar artefactos de aumentación para su uso debe ser lo más sencillo posible, confiriéndoles absoluto control y conocimiento

sobre los artefactos de aumentación que pueden utilizar y los que se encuentran utilizando.

- **Compatible:** el enfoque negociado y las herramientas que lo implementan, deben ser compatibles con los artefactos de aumentación existentes en las comunidades de Aumentación Web populares (como la estudiada en el capítulo 2).
- **Independencia:** Si bien en la implementación del enfoque propuesto el repositorio utilizado es compartido por todos los sitios webs y es de público acceso, los componentes de aplicación relativos a la distribución de artefactos de aumentación entre los visitantes de un sitio, debe ser independiente de un repositorio en particular.

6.2. Componentes de aplicación para la distribución de artefactos

Como fue introducido en el capítulo 3, la distribución de los artefactos de aumentación se encuentra basada en dos componentes conceptuales fundamentales: el ACHA, un componente de aplicación que centraliza la gestión de los artefactos de aumentación en un servidor y el AAP, un componente de aplicación que es instalado en los sitios webs para descubrir y ofrecer a sus navegantes artefactos de aumentación que pueden ser utilizados en sus sesiones de navegación.

El componente de aplicación ACHA, del lado del servidor, interactúa con el repositorio de artefactos de aumentación y al mismo tiempo ofrece distintas vistas y funcionalidades para la gestión de los artefactos por parte de los usuarios en sus distintos roles (dueño, desarrollador, usuario final).

Por su parte en el cliente, el componente de aplicación AAP, ofrece un punto de acceso a los artefactos de aumentación al momento de visitar los sitios webs. Este componente se encuentra conformado por:

- **Selector de aumentadores para usuarios finales:** Permite a los usuarios

finally select the artifacts of augmentation that they want to use among those that are available for the website that they are visiting.

- Injector de artefacto de aumento: permite descargar y ejecutar los artefactos de Aumentación Web que resulten seleccionados.
- Emulador de motor de aumento: permite que los artefactos de aumento de usuarios finales tradicionales sean compatibles con el enfoque.

6.2.1. Caso de estudio

In this section we present the application components implemented, through a case study published in [35] based on the website DBLP. The examples in this section are relative to the search result in the website DBLP shown in Figure 6.1.

Assuming that a developer has implemented two artifacts of au-

2014		Refine by WORD
338	EE Tsaraj Lalch, Arash Khodadadi, Sargani A. Mokhey, Joey Paquet, Yuhong Yan: Toward Policy-Based Dynamic Context-Aware Adaptation Architecture for Web Service Composition. <i>CIS2E</i> 2014:23	adaptation (323) adaptations (15)
337	EE Annie Lewis, Bonnie L. Webber: Structured and Unstructured Cache Models for SMT Domain Adaptation. <i>EACL</i> 2014:155-163	
336	EE Alaa A. Ouffas, Alexandra I. Cristina: How to Create an E-Advertising Adaptation Strategy: The AEADS Approach. <i>EC-Web</i> 2014:171-178	Refine by AUTHOR
335	EE Dongsong Zhang, Anil Jangam, Lina Zhou, Iail Yukui: User-centered, device-aware multimedia content adaptation for mobile web. <i>ICIS</i> 2014:275-280	Geert-Jan Houben (9) Claudia Casti (6) Rocco de Troiano (6) Roberto De Virgilio (6) [top 4] [top 50] [top 250]
334	EE Lina Zhou, Vikas Bansal, Dongsong Zhang: Color adaptation for improving mobile web accessibility. <i>ICIS</i> 2014:291-296	
333	EE Edson J. Mendes, Thain Webber, César A. M. Marcon, Fernando Moraes, Ney Calazans: A monitored NoC with runtime path adaptation. <i>ISCA</i> 2014:1965-1968	Refine by VENUE
332	EE Dries Goebel, Kristof Goebel, Eddy Triyem, Sam Michiels, Johan A. K. Suykens, Joos Vandewalle, Wimster Jossen: QoS prediction for web service compositions using kernel-based estimation with online adaptation of the constant offset. <i>Inf. Sci. (ISCI)</i> 268-397:424 (2014)	WEBIST (7) WebMedia (7) ICI (7) Web Intelligence (5) [top 4] [top 50] [top 241]
331	EE Hatthem Mezni, Walid Chaib, Khaled Ghédir: Extending Policy Languages for Expressing the Self-Adaptation of Web Services. <i>I. UCS (IUCS)</i> 2008:1130-1151 (2014)	
330	EE Apostolos Papageorgiou, André Miede, Stefan Schulte, Dieter Schaller, Ralf Steinmetz: Decision support for Web service adaptation. <i>Persasive and Mobile Computing (PERCOM)</i> 12:197-213 (2014)	Refine by YEAR
329	EE Laisa Fernanda Barreto, Angélica Carrillo Ramos, Leonardo Floruz, València, Jaime A. Puylich-Mariscal, Nadia Alejandra Mejía-Molina: Integrating Adaptation and HCI Concepts to Support Usability in User Interfaces - A Rule-based Approach. <i>WEBIST</i> 2014:82-90	2014 (12) 2013 (27) 2012 (23) 2011 (23) [top 4] [all 19]
328	EE Wei Gao, Pei Yang: Democracy is good for ranking: towards multi-view rank learning and adaptation in web search. <i>WSDM</i> 2014:63-72	
327	EE Woralak Kongdejthua, Hamid R. Motahari-Nasab, Boualem Benatallah, Régis Saint-Paul: Web Service Adaptation: Mismatch Patterns and Semi-Automated Approach to Mismatch Identification and Adapter Development. <i>Web Services Foundations</i> 2014:245-272	Refine by TYPE
326	EE Chenguang Wang, Nan Duan, Ming Zhou, Ming Zhang: Paraphrasing Adaptation for Web Search Ranking. <i>ACL</i> 2013:81-86	Conference (264) Journal (30) Editor (2) Book (1) [top 4] [all 5]
325	EE Habib Louadi, Stéphane Coulombe, Umesh Chandr: Efficient Near-Optimal Dynamic Content Adaptation Applied to JPEG Slides Presentations in Mobile Web Conferencing. <i>AINA</i> 2013:724-731	
324	EE Michael Nebeling, Maximilian Speicher, Moira C. Noeris: WInouch: metrics-based web page adaptation for touch. <i>CHI</i> 2013:2311-2320	
323	FF Metzger Chen, Dinoshine Zhang, Zhichao Wang, Jirun Pan, Yuehong Yan: Web-Based Language Model Domain Adaptation for Real World	

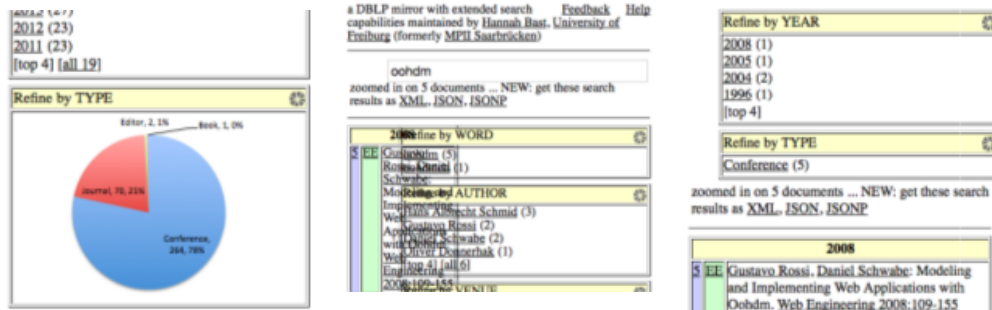
Figura 6.1: Resultado de búsqueda en DBLP.

mentación para los requerimientos de la Figura 6.2, el primero de ellos analiza los resultados de la búsqueda y genera un gráfico de tortas que es agregado al sitio web aumentado, Figura 6.2a. El segundo de los artefactos modifica

la interfaz de usuario del sitio (Figura 6.2b) con el objetivo de hacerlo más responsivo en relación al tamaño de la pantalla, Figura 6.2c.

Los artefactos implementados los ha compartido con el resto de la masa de usuarios y están listos para ser utilizados, pero a este punto, el responsable del sitio web DBLP no los ha certificado.

Asumiendo que los dueños y responsables del sitio web DBLP, se en-



(a) Artefacto para agregar gráfico visualizando publicaciones por tipo. (b) Visualización original con superposición de elementos en tamaños de pantalla pequeños. (c) Artefacto para solucionar problema de responsabilidad.

Figura 6.2: Artefactos implementados por un desarrollador.

cuentran interesados en facilitar a sus visitantes la posibilidad de utilizar estos artefactos de aumentación, se registran en el ACHA realizando los siguientes pasos:

- Crean una cuenta.
- Validan la cuenta como perteneciente a un responsable del sitio web DBLP. Para realizar este paso deberán validar su identidad subiendo y haciendo accesible un archivo en la raíz de su dominio que deberá contener un *token* de seguridad provisto con ese objetivo por el ACHA.
- Instalan el AAP en su sitio web.

Habiendo realizado estos pasos para la validación de la cuenta y la instalación del AAP en su sitio web (Figura 6.3), un usuario que ingrese al ACHA

con las credenciales validadas de la cuenta creada, podrá navegar el repositorio de artefactos de aumentación y seleccionar aquellos que desea que sean ofrecidos a su visitantes.

El proceso de certificación implica simplemente seleccionar los artefac-

```
1 <!DOCTYPE html>
2 <link rel="stylesheet" type="text/css" href="http://www.dblp.org/autocomplete-php/autocomplete/logging.css">
3 <script type="text/javascript" src="http://www.dblp.org/autocomplete-php/autocomplete/autocomplete.js"></script>
4 .....
5 <script src="http://www.dblp.org/./AugmenterAccessPoint.js" onload="URM_init('aug.userrequirements');"></script>
6 .....
7 <html>
8 <head>
```

Figura 6.3: Inclusión AAP en el sitio web principal de DBLP.

tos activando o desactivando su distribución a los visitantes del sitio. En la Figura 6.4, el responsable del sitio ha aprobado uno de los artefactos de aumentación disponibles en el repositorio habilitando su distribución.

Cuando los usuarios finales visitan el sitio web, son advertidos de la



Figura 6.4: Artefacto certificado por el responsable del sitio web.

existencia de artefactos de aumentación disponibles con un pequeño ícono, ubicado en la esquina superior derecha de la pantalla. Al hacer *click* sobre el ícono, el selector de artefactos de aumentación es desplegado permitiéndoles conocer qué artefactos pueden ser utilizados en el sitio que está visitando actualmente. Para activar el artefacto, el visitante simplemente lo selecciona y el inyector de artefactos de aumentación incorporará el artefacto a la sesión de navegación. La Figura 6.5 muestra como el selector de artefactos de aumentación, solo ofrece a los usuarios aquellos artefactos que hubieren sido

certificados por los dueños de los sitios webs que visitan, en el ejemplo, el artefacto correspondiente a la Figura 6.2c.

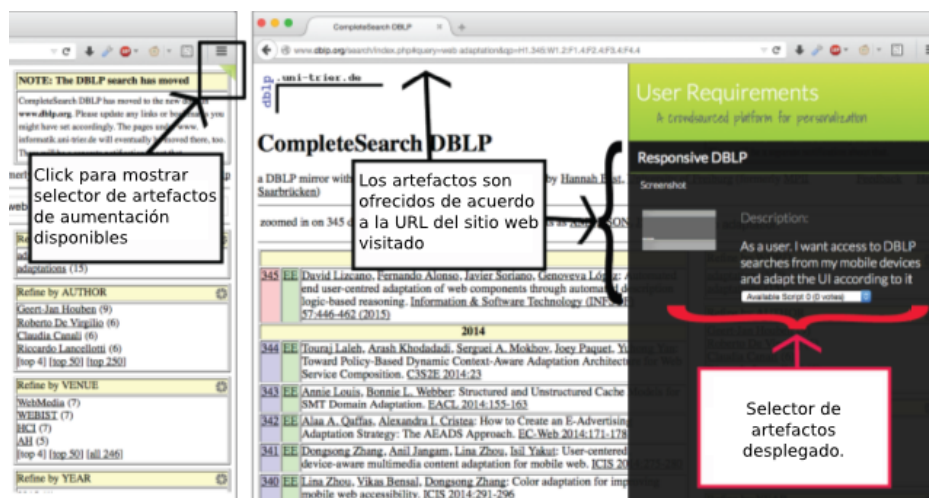


Figura 6.5: Selector de artefactos de aumentación.

6.3. Mantenimiento por usuarios finales

Como ha sido mencionado anteriormente, suele decirse que la Aumentación Web es la web, como la realidad aumentada es a la realidad. Esta analogía silenciosamente manifiesta un gran desafío al que se enfrentan los artefactos de aumentación: en la realidad los edificios, las plazas, los lugares de interés no suelen cambiar de lugar. En contraste los artefactos de Aumentación Web se encuentran ante la inevitable problemática de que los sitios webs suelen frecuentemente cambiar su forma, contenido, o estructura. Como fue estudiado en el capítulo 2, los artefactos de aumentación mantienen las referencias a los sitios webs aumentados de modo *hardcodeado* en su definición y ello genera necesidades de mantenimiento de los artefactos cuando estas referencias deben ser actualizadas. Este es un problema reconocido por la comunidad [23] y compartido con otras áreas como *web annotation* [16] y *web*

data extraction [29]. Han surgido distintos enfoques para el mantenimiento de referencias a sitios webs en distintas áreas de interés [18, 51, 79].

Estos enfoques suelen ser categorizados en función de qué estrategia de referenciación es utilizada, en función de si son preventivos o curativos y en función de si requieren o no la supervisión de un usuario. Independientemente de ello, su aplicación requiere habilidades muy particulares o bien por parte de los desarrolladores o bien por parte de los usuarios y como fue estudiado en el capítulo 2, en las comunidades de Aumentación Web, las referencias simplemente son incrustadas y los desarrolladores las mantienen por su propia cuenta actualizando el artefacto. El mantenimiento de estas referencias, requiere que el desarrollador del artefacto analice los cambios en el sitio web aumentado, con el fin de obtener nuevas referencias que permitan reemplazar las desactualizadas. Generalmente esta tarea es llevada a cabo mediante los inspectores del DOM Tree que los exploradores tienen integrados.

Aunque resulte trivial, las referencias al sitio web aumentado solo tienen oportunidad de fallar cuando el artefacto de aumentación es ejecutado. El instante de tiempo en que un artefacto mantiene referencias desactualizadas y el instante de tiempo en que ello es advertido define un intervalo de tiempo indeterminado. A la postre, suele suceder que las aplicaciones webs generen distintas estructuras o contenidos en sus documentos en relación al perfil del usuario que las utiliza. Por lo que distintos usuarios de un mismo artefacto de aumentación, podrían verse afectados de distinto modo en relación a la necesidad de actualizar estas referencias.

Como fue descrito en el capítulo 3, el enfoque propone involucrar a la masa de usuarios finales en el mantenimiento de estas referencias. De este modo, cuando un usuario de un artefacto se encuentra impedido en la correcta ejecución de un artefacto de aumentación porque alguna de sus referencias falla, tendrá oportunidad de colaborar inmediatamente en la selección supervisada de nuevas referencias y compartirlas con el resto de los usuarios del artefacto. Independientemente de ello, el enfoque prevé que cuando este tipo de fallas ocurren, sean notificados aquellos actores que se vieron involucrados en alguna de las etapas correspondientes a la construcción del artefacto.

En esta sección se describen en mayor profundidad esta actividad y las herramientas que dan soporte a los usuarios finales para poder realizarlas.

6.3.1. Mantenimiento de referencias DEOI

En relación al posible mantenimiento de las referencias DEOI fueron consideradas las siguientes premisas fundamentales:

- Cada elemento de interés en el sitio web aumentado debe ser referenciado a través de un conjunto de referencias.
- Las referencias DEOI deben ser desacopladas del artefacto.
- Los usuarios finales pueden descubrir y compartir nuevas referencias.
- Cuando ninguna referencia es válida, los usuarios finales pueden colaborar en la selección de las mismas de un modo supervisado.

Con estas premisas en mente, es importante destacar que para poder llevar a delante esta estrategia, los artefactos de Aumentación Web deben haber sido construidos de un modo en particular. Es decir, los desarrolladores de los artefactos de aumentación que tuvieron oportunidad de implementar los requerimientos de aumentación, deben haber considerado, al momento de su implementación, los beneficios de implementar su artefacto haciendo uso del framework CMFW, descrito en el apartado anterior. Al encontrarse las referencias DEOI desacopladas del artefacto de aumentación y mantenidas en un repositorio, los usuarios finales podrán actualizarlas sin requerir ningún tipo de habilidad en programación.

En relación al componente de aplicación que facilite a los usuarios finales la realización de estas tareas, fue considerado importante no requerir la instalación de extensiones, para alcanzar a cualquier usuario que se encuentre utilizando el artefacto. Del mismo modo, fue considerado importante utilizar la información disponible en los modelos MockPlug que dieron lugar a la implementación del artefacto tanto como sea posible. Así se podría guiar a los usuarios finales a la correcta selección de nuevas referencias.

En la Figura 6.6, se muestra como un usuario final fue advertido de

la necesidad de mantenimiento de referencias ante un fallo producido en la ejecución de un artefacto para el sitio web YouTube. El componente de aplicación ADRM fue desplegado a la izquierda de su pantalla permitiéndole conocer la imagen del modelo MockPlug del requerimiento correspondiente al artefacto que se encontraba utilizando. En una operación muy similar a las realizadas a la hora de agregar componentes en los modelos MockPlug, el usuario puede *arrastrar y soltar* los componentes en el lugar que corresponda o *colectar* elementos utilizando una *mira* en el caso de los componentes colectados.

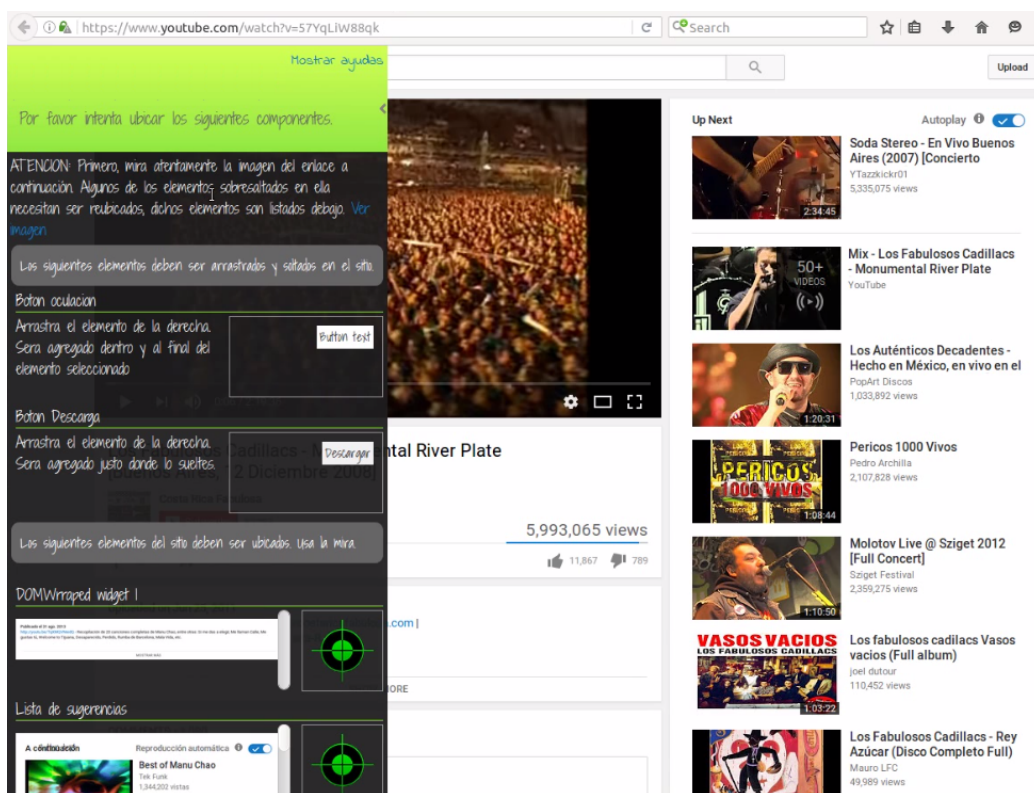


Figura 6.6: ADRM: componente para el mantenimiento de referencias DEOI.

6.3.2. Validación de nuevas referencias

Si bien la selección de nuevas referencias para los componentes del artefacto puede ser llevada a cabo del modo descripto, ello implica un nuevo desafío: cómo saber que las referencias seleccionadas por el usuario final son válidas para el artefacto y la historia de usuario en general.

En un principio, vale la pena destacar que las nuevas referencias coleccionadas de esta manera, no reemplazan las referencias del modelo original del artefacto, y que en su lugar, son coleccionadas y compartidas como un conjunto de referencias adicional y secundario al cual el framework recurre cuando las originales fallan. Habiendo realizado esta aclaración también es importante señalar, que si el usuario final decidiera no colaborar en el mantenimiento de referencias del artefacto, no podrá utilizarlo hasta que el mismo sea mantenido por los actores involucrados en su construcción originalmente, o hasta que algún otro usuario final decida mantenerlas y compartirlas.

Los usuarios desarrolladores de artefactos de aumentación que realicen su implementación utilizando CMFW, podrán indicar condicionamientos de aceptación de nuevas referencias siendo tan permisivos o restrictivos como deseen. Por otro lado, las referencias, para ser aceptadas antes de ser compartidas con el resto de los usuarios, deben lograr que el artefacto supere todos los casos de prueba.

Capítulo 7

Validación

En este capítulo se presentan las evaluaciones realizadas para validar el enfoque propuesto y las herramientas que lo soportan. La sección 7.1 describe una experimentación realizada con el objetivo de evaluar la capacidad del enfoque a la hora de expresar requerimientos de Aumentación Web. En la sección 7.2 se describe una experimentación que permitió evaluar la factibilidad de incluir a la masa de usuarios finales en el mantenimiento de las referencias DEOI de los artefactos de Aumentación Web. Tratándose de experimentos controlados, se describen ambas experimentaciones según la guía para el reporte de este tipo de experimentos sugerida en [73].

7.1. Definición de requerimientos

En esta sección se describe una evaluación que hace foco en la etapa de definición de requerimientos y las herramientas provistas para definirlos.

Empoderar a la masa de usuarios finales con herramientas para definir sus propios requerimientos de aumentación en términos de RAMs es un pilar fundamental de todo el enfoque. Estos prototipos de alta fidelidad constituyen una novedad en términos de definición de requerimientos, ya que son un

tipo de soporte no utilizado antes en el marco de la ingeniería de requerimientos tradicional. Por ello, fue desarrollado un experimento controlado donde usuarios web experimentados jugaron el rol de usuarios finales para especificar requerimientos de aumentación sobre sitios webs existentes, permitiendo medir el nivel de satisfacción obtenido en la utilización de las herramientas y el nivel de satisfacción producido por la implementación de artefactos de Aumentación Web, a partir de la definición de estos requerimientos.

El experimento compara los resultados obtenidos al especificar requerimientos usando CrowdMock, en contraste a los resultados obtenidos al especificar los requerimientos de un modo tradicional. Para poder realizar esta comparación los participantes tuvieron que producir especificaciones de requerimientos de Aumentación Web y luego un grupo de desarrollares los implementaron, para finalmente ser evaluados por sus interesados en relación a si éstos satisfacían o no las necesidades que habían especificado.

El objetivo de la experimentación puede ser formulado de la siguiente manera:

- Analizar: *CrowdMock y enfoques tradicionales en el desarrollo de artefactos de aumentación*
- Con el propósito de: *evaluar el enfoque CrowdMock*
- Respecto de su: *efectividad*
- Desde el punto de vista: *de los usuarios*
- En relación a: *el nivel de satisfacción alcanzado por los usuarios que especifican requerimientos utilizando productos construidos a partir de éstos.*

7.1.1. Protocolo

Esta sección describe el protocolo utilizado para llevar adelante el experimento y analizar sus resultados.

Participantes

Un grupo de 20 personas participó de la experimentación, 9 mujeres y 11 varones, con un rango etario de entre 22 y 60 años. Todas las personas eran de profesión informáticos y con experiencia en ingeniería web. Entre ellos había analistas, desarrolladores y líderes de equipos de desarrollo. La gran mayoría de los participantes fueron argentinos, pero también participaron personas provenientes de Ecuador y de Suecia. La experimentación tuvo lugar en el marco de cursos de capacitación y posgrado llevados a delante en distintas universidades de nuestro país: Universidad Abierta Interamericana (8 participantes), Universidad Nacional de La Plata (8 participantes), Universidad Nacional de la Patagonia San Juan Bosco (4 participantes).

Ninguno de los participantes tenía experiencia o conocimiento previo en relación al enfoque y sus herramientas. Sin embargo al ser los participantes profesionales de nuestra disciplina con experiencia en ingeniería web, fueron capaces de evaluar, luego de haber tenido oportunidad de experimentar su uso, si las técnicas tradicionales resultaban más efectivas o no que la propuesta por CrowdMock.

Materiales

Técnicas Las técnicas usadas para especificar requerimientos de aumentación fueron dos. Una de ellas soportada por la herramienta MockPlug descrita en capítulo 4. Por otro lado los participantes utilizaron un soporte tradicional de su preferencia ya por ellos conocido (descripción textual, historia de usuario, mockup). En ambos casos se utilizaron procesadores de texto, pudiendo también contener imágenes, para construir el soporte documental de los requerimientos.

Formularios Fueron utilizados formularios para obtener información acerca de las tareas relativas a la especificación de requerimientos y para obtener

información acerca de los artefactos que fueron implementados a partir de éstos. El primer formulario constituía un cuestionario que solicitaba información como la dificultad percibida a la hora de especificar el requerimiento y el tiempo empleado en su definición, mientras el segundo formulario solicitaba información acerca del nivel de satisfacción alcanzando en la ejecución de los artefactos.

Meta requerimientos Los requerimientos de aumentación fueron especificados sobre dos sitios webs bien conocidos: Youtube e IMDB. En función del estudio realizado acerca de las categorías de alteración producidas por los artefactos de Aumentación Web visto en el capítulo 2 (sección 3), fueron establecidos 5 meta requerimientos:

IMDB

- F1.1: Filtrar ranking de mejores 250 películas con algún criterio.
- F1.2: Agregar información al ranking de 250 mejores películas.
- F1.3: Cambiar la estructura/contenido de la ficha de una película.

YouTube

- F2.1: Agregar información a los resultados de búsqueda.
- F2.2: Cambiar la estructura/contenido de la vista de un video.

Tanto para el meta requerimiento F1.3, como para el meta requerimiento F2.2, basados en alteraciones de contenido/estructura, las alteraciones debían basarse en: i) mover elementos del sitio web, ii) eliminar elementos del sitio web, iii) agregar elementos al sitio web, iv) agregar contenido relacionado proveniente de otro sitio web.

Ejecución de artefactos Los artefactos producidos en respuesta a los requerimientos definidos, fueron construidos como scripts de usuario a ejecutarse sobre Greasemonkey. Por tal motivo los participantes debieron instalar

esta extensión previo a la evaluación de los artefactos.

Tareas

Los participantes tuvieron que realizar dos tareas en el hogar: especificar los requerimientos de Aumentación Web que desearon, sujetos a los meta requerimientos establecidos, y utilizar los artefactos producidos a partir de ellos. Al finalizar cada una de las tareas debían llenar los formularios correspondientes.

Cada participante tuvo que especificar 4 requerimientos: dos de ellos utilizando técnicas tradicionales bien conocidas de su preferencia y dos de ellos utilizando RAMs. Durante la definición de los requerimientos debieron llevar un diario a de actividades registrando cada principio y fin de sesión de trabajo, la actividad realizada y una posible descripción de problemas encontrados.

Una vez finalizada la especificación de requerimientos y completados los formularios en relación a ello, los participantes debieron instalar y utilizar los artefactos producidos y finalmente completar el cuestionario acerca del nivel de aceptación alcanzado por la utilización de dichos artefactos de aumentación en respuesta a sus respectivos requerimientos.

Hipótesis y variables

El experimento tiene un único objetivo: comparar la satisfacción percibida por los participantes cuando utilizaron artefactos cuya funcionalidad fue requerida utilizando requerimientos CrowdMock versus técnicas tradicionales. De este modo las hipótesis resultan:

- H_0 : No se producen diferencias de satisfacción en consecuencia de la utilización de CrowdMock y técnicas tradicionales para especificar los requerimientos.

- H_a : Se producen diferencias de satisfacción al usar CrowdMock y técnicas tradicionales para especificar los requerimientos.

Tres tipos de variables pueden definirse para el experimento: independiente, dependiente, de control. La técnica utilizada (CrowdMock, Tradicional) es una variable independiente. El nivel de satisfacción obtenido (no satisfactorio, parcialmente satisfactorio, satisfactorio) es la variable dependiente ya que se desea medir cómo varía en relación a la técnica utilizada. Finalmente, los meta requerimientos son una variable de control, ya que es un conjunto limitado y compartido al que los participantes deben ajustarse haciendo comparables las mediciones.

Procedimiento

Antes de comenzar el experimento, se introdujo a los participantes en el procedimiento general y se los entrenó en la utilización de MockPlug.

El experimento fue organizado en dos fases. En cada fase, los participantes especificaron dos requerimientos, luego un grupo de desarrolladores implementó los requerimientos obteniendo los artefactos correspondientes y finalmente los participantes validaron si la implementación obtenida satisfacía o no sus expectativas.

Un grupo de participantes debió utilizar técnicas tradicionales en la especificación de sus requerimientos en la primer fase, mientras otro grupo utilizó CrowdMock en la primer fase. Durante la segunda fase, los participantes debían utilizar la técnica (tradicional/CrowdMock) que no hubieran utilizado en la fase anterior. Cada participante debió seleccionar meta requerimientos distintos en cada fase, no pudiendo repetir el meta requerimiento.

Cada fase de definición de requerimientos debió completarse en 3 días corridos, debiendo los participantes al finalizar, enviar un correo electrónico con un archivo PDF conteniendo la especificación de su requerimiento. En el caso de los RAMs, el PDF debía contener su correspondiente URL. Pasados 2 días para la construcción de los artefactos por parte de un grupo de desarrolladores, los artefactos fueron enviados por correo electrónico y los

participantes contaron con 2 días corridos más para llevar a adelante su evaluación.

Durante estos periodos de tiempo debieron completar los formularios.

Procedimiento de análisis

El análisis a llevar a delante sobre los datos obtenidos, debe ser un análisis no paramétrico, debido al pequeño conjunto de muestras obtenido. En este contexto, y tratándose de variables cualitativas y ordinales, una prueba de aceptación hipótesis apareada utilizando el método *Wilcoxon Signed Rank Test* fue llevado a delante [85].

7.1.2. Ejecución

En esta sección se describen aspectos relacionados a la ejecución del experimento, que fue realizada de acuerdo al procedimiento previamente descrito.

La primer tarea a realizar por parte de los participantes fue la selección de los meta requerimientos a refinar. La Tabla 7.1 muestra la distribución de los requerimientos definidos, que resultó bastante balanceada.

Los participantes realizaron sus tareas de acuerdo al cronograma es-

Tabla 7.1: Distribución de requerimientos.

Meta requerimiento	% de selección
F1.1	12
F1.2	22
F1.3	17
F2.1	25
F2.2	24

tablecido. Las técnicas tradicionales utilizadas fueron mockups (prototipos

tradicionales), historias de usuario, descripciones narrativas y combinaciones de prototipos con descripciones textuales. La Tabla 7.2, muestra en qué medida estas técnicas tradicionales fueron seleccionadas.

Respecto de los cuestionarios de usabilidad, la dificultad en la utiliza-

Tabla 7.2: Distribución de técnicas tradicionales.

Técnica Tradicional	% de utilización
<i>Descripción textual</i>	26
<i>Descripción Textual + prototipo</i>	24
<i>Prototipo</i>	39
<i>Historia de usuario</i>	11

ción de las técnicas utilizadas para la especificación de los requerimientos fue medida de acuerdo a una escala discreta desde *Muy fácil* a *Muy Difícil*, considerando *fácil*, *normal* y *difícil* como valores intermedios. Cuando fueron especificados los requerimientos con técnicas tradicionales, la dificultad fue percibida mayormente como normal. Mientras que en el caso de la especificación de requerimientos utilizando CrowdMock, la dificultad percibida (si bien resultó ser también considera mayormente normal), la cantidad de casos en las que fue percibida entre *fácil* y *muy fácil* es mayor en comparación. Si bien esto último también sucede para el caso de las dificultades percibidas como *muy difícil*, es importante mencionar que hubieron casos en que los participantes tuvieron inconvenientes técnicos relativos a la instalación y compatibilidad de su explorador con la herramienta MockPlug, como así también en relación a la correcta internacionalización de la herramienta. La Tabla 7.3, muestra como fueron distribuidas estas calificaciones para cada una de las técnicas.

Si bien el equipo de desarrolladores implementó los requerimientos y los envió a cada participante en todos los casos, los participantes pudieron descargarlos, instalarlos y evaluarlos en el 95 % de los casos. El nivel de satisfacción reportado por los participantes al utilizar los artefactos se muestra en la Tabla 7.4.

Con el objetivo aparear los resultados por sujeto, en la Tabla 7.5 se muestran los niveles de satisfacción obtenidos agrupados por participante y número de requerimiento (primero, segundo) en cada fase, dando lugar a to-

Tabla 7.3: Dificultad percibida al especificar requerimientos.

Nivel	% Tradicional	% CrowdMock
Muy fácil	5	11
Fácil	21	26
Normal	63	42
Difícil	11	16
Muy difícil	0	5

Tabla 7.4: Nivel de satisfacción percibido por técnica.

Nivel	% Tradicional	% CrowdMock
No satisfactorio	8	5
Parcialmente satisfactorio	50	32
Satisfactorio	42	63

das las combinaciones posibles de valoraciones y al conteo de casos en los que se reportó esa combinación.

En la mayoría de los casos, los participantes percibieron el mismo nivel

Tabla 7.5: Nivel de satisfacción percibido agrupado por pares.

Tradicional	CrowdMock	Cantidad
No satisfactorio	No satisfactorio	1
No satisfactorio	Parcialmente satisfactorio	1
No satisfactorio	Satisfactorio	1
Parcialmente satisfactorio	No satisfactorio	1
Parcialmente satisfactorio	Parcialmente satisfactorio	7
Parcialmente satisfactorio	Satisfactorio	11
Satisfactorio	No satisfactorio	0
Satisfactorio	Parcialmente satisfactorio	4
Satisfactorio	Satisfactorio	12

de satisfacción independientemente de la técnica utilizada para especificación de los requerimientos (20 casos). Sin embargo, en los 18 casos restantes, los participantes obtuvieron diferentes niveles de satisfacción, en los que hubo una diferencia a favor a de CrowdMock, donde en 13 casos (72% de los casos

restantes), el nivel de satisfacción fue mejor en comparación (No satisfactorio/Parcialmente satisfactorio, No satisfactorio/Satisfactorio, Parcialmente satisfactorio/Satisfactorio).

7.1.3. Análisis


En el protocolo del requerimiento fueron definidas las hipótesis, las variables y el procedimiento de análisis a llevar adelante, cuyos resultados se describen en esta sección.

Se analizaron los 38 pares correspondientes a los niveles de satisfacción obtenidos por los participantes que pudieron evaluar los artefactos (95 % de los casos). Sobre estos pares agrupados por participante, que fueron resumidos en la Tabla 7.5, se aplicó la prueba de hipótesis Wilcoxon Signed Rank Test. Al realizar esta prueba, se reduce la muestra original desechando los pares empatados, reduciendo ello la muestra a 18 pares. Con un estadístico $W+ = 45$ y un $N=18$. Esta prueba de hipótesis establece que la hipótesis nula debe ser rechazada, si el estadístico $W+$ obtenido es menor que el establecido en tablas para un determinado nivel de confianza. Las tablas indican un valor crítico $W=47$ [55], por lo que la hipótesis nula debe ser rechazada (con un $p=0,05$). La Figura 7.1 muestra el resultado de ejecutar el test sobre los datos obtenidos, utilizando R^1 .

En resumen y con un nivel de confianza del 95 % ($p<0,05$), hay una diferencia sustancial entre la aplicación de CrowdMock versus las técnicas tradicionales para la especificación de requerimientos, donde la hipótesis nula es rechazada.

El experimento fue enfocado en la capacidad de CrowdMock de expresar requerimientos. En particular fue enfocado en evaluar el nivel de satisfacción producido por los artefactos implementados a partir de sus especificaciones. Los participantes utilizaron dos técnicas distintas: CrowdMock y una tradicional de su preferencia. Esta situación era adversa para la comparación. Ya que las técnicas tradicionales, a diferencia de CrowdMock, eran bien cono-

¹<https://www.r-project.org/>



```
> wilcox.test(pares$Tradicional,pares$CrowdMock, paired=TRUE,
              Wilcoxon signed rank test
data: pares$Tradicional and pares$CrowdMock
V = 45, p-value = 0.04953
alternative hypothesis: true location shift is not equal to 0
```

Figura 7.1: Estadístico $W+$ calculado en R.

cidas por los participantes. Sin embargo, CrowdMock obtuvo mejores resultados. Por otro lado, en la mayoría de los casos, el tiempo requerido en la utilización de CrowdMock fue equivalente a la mitad del tiempo empleado en la utilización de las técnicas tradicionales y CrowdMock obtuvo mejores resultados en términos de nivel de dificultad percibida en su utilización, incluso habiendo permitido a los participantes la elección de la técnica tradicional que más conocían. Se considera que el hecho de que CrowdMock se basa en prototipos aplicados sobre el sitio web original, es la clave de los resultados satisfactorios del experimento.

7.1.4. Validez

En relación a la validez del experimento es importante destacar que en el análisis de su resultado se utilizó un test estadístico de validación de hipótesis no paramétrico como Wilcoxon Signed Rank Test, que al ser no paramétrico, no requiere ninguna certeza acerca de las distribuciones que describen las poblaciones de las variables analizadas. Por otro lado, este test requiere variables nominales ordinales, como lo fue la escala de nivel de satisfacción utilizada en los formularios, donde los usuarios pudieron distinguir fácilmente el valor a reportar en su nivel de satisfacción. Así mismo, este tipo de test de aceptación requiere de pares de muestras del tipo *antes/después*, en donde al ser agrupados por participante, el criterio de relación entre los pares evita discrepancias relativas a la subjetividad en la utilización de la escala

de conformidad.

En relación a la heterogeneidad/homogeneidad de los participantes, puede apreciarse que fue lo suficientemente variado en edad y género destacándose su homogeneidad en ser todos ellos de profesión informática. Esto último les permitió elegir su técnica tradicional de preferencia para la definición de requerimientos, que en conjunto a la posibilidad de seleccionar el meta requerimiento, evaden las amenazas por selecciones restringidas a un determinado tipo de muestras.

En la ejecución del experimento, los participantes fueron divididos de modo que parte de ellos tuvo que utilizar CrowdMock en la primer fase y parte de ellos CrowdMock en la segunda fase. De este modo se redujeron las posibilidades de introducir desvíos por maduración en la experimentación. En relación a los artefactos construidos, posibles desvíos fueron evadidos por involucrar en su implementación a desarrolladores de experiencia que no tenían conocimiento del protocolo experimental llevado a delante. Por otro lado, los participantes involucrados nunca supieron qué parte de los formularios iba a ser utilizada para el análisis de la experimentación permitiendo esto evadir desvíos basados en su posición.

7.2. Mantenimiento de referencias

En esta sección se describe una evaluación que hace foco en la etapa de mantenimiento de los artefactos de Aumentación Web, en relación a sus referencias a los sitios webs aumentados.

El experimento compara los resultados obtenidos al utilizar mantenimiento de referencias tradicional, en contraste con los resultados obtenidos en el mantenimiento de referencias por la masa de usuarios propuesto en CrowdMock.

Para poder realizar esta comparación los participantes tuvieron que utilizar artefactos de aumentación mantenidos con los distintos enfoques, registrándose la cantidad de ejecuciones exitosas y fallidas producidas durante el transcurso de la experimentación.

El objetivo de esta experimentación puede ser expresado de la siguiente manera:

- Analizar: *CrowdMock* y enfoques tradicionales en el mantenimiento de artefactos de aumentación
- Con el propósito de: *evaluar el enfoque CrowdMock*
- Respecto de su: *efectividad*
- Desde el punto de vista: *de los usuarios*
- En relación a: *recuperar el correcto funcionamiento de los artefactos cuando las referencias DEOI cambian en el sitio web aumentado.*

7.2.1. Protocolo

Esta sección describe el protocolo utilizado para llevar adelante el experimento y analizar sus resultados.

Participantes

Un grupo de 48 personas participó de la experimentación, 8 mujeres y 40 varones. Todas las personas eran estudiantes de la carreras de informática de la Facultad de Ingeniería, UNPSJB, sede Trelew. La gran mayoría de los participantes fueron argentinos pero también participó una persona extranjera.

Ninguno de los participantes tenía experiencia o conocimiento previo en relación al enfoque y sus herramientas.

Materiales

Técnicas Las técnicas usadas para el mantenimiento de los artefactos de aumentación fueron dos. La propuesta por CrowdMock, soportada por la herramienta ADRM descrita en capítulo 6 y por otro lado, la técnica tradicional para mantenimiento de artefactos de Aumentación Web, basada en la actualización de las referencias harcodeadas del artefacto, estudiada en el capítulo 2.

Artefactos y requerimientos Para llevar a delante el experimento fue necesario disponer de los artefactos a mantener en relación a la técnica tradicional y la propuesta por CrowdMock.

Estos artefactos de Aumentación Web daban solución a dos requerimientos: uno sobre el sitio web Wikipedia² y otro sobre el sitio web YouTube. Por cada uno de estos dos requerimientos se implementaron dos artefactos: uno fue implementado utilizando el enfoque tradicional (usando jQuery con referencias harcodeadas en el artefacto) y el segundo, utilizando el framework CrowdMock. En total se dispuso de un conjunto de 4 artefactos.

Extensión del experimento Fue desarrollada una extensión para el explorador web Firefox que los participantes debieron instalar para participar del experimentación. La extensión se encargaba de recibir a los participantes con un formulario de inscripción y recordarles que debían visitar el sitio web aumentado durante el transcurso de los días durante los cuales el experimento fue llevado a delante. La extensión se muestra en la Figura 7.2, la Figura 7.2a muestra el formulario de inscripción donde datos del participante fueron solicitados con el único objetivo de entregar certificados de participación posteriormente. La Figura 7.2b muestra el menú principal de la extensión, que simplemente le recordaba al participante visitar el sitio web, a la vez de ofrecerle acceso al registro de sus visitas en relación a si había o no visitado

²<https://es.wikipedia.org/>

el sitio web de interés durante los días en los que transcurrió la experimentación.

Experimentación ¡Bienvenido!

Para dar comienzo al experimento, debes ingresar tus datos personales y tu correo electrónico.
¡Muchas gracias por tu participación!

Ingresá tus datos

Nombre Roberto

Apellido Fontanarrosa

DNI 99999999

Correo Electrónico roberto.fontanarrosa@gmail.c

Confirma tus datos para comenzar la experimentación

Confirmar

UNIVERSIDAD NACIONAL DE LA PLATA DIT Lafia

(a) Formulario de inscripción.

Experimento

Instrucciones Registro

Descripción

Diariamente, durante 10 días, debes visitar el sitio que te ha tocado.
MUY IMPORTANTE: Si apareciera un menu de experimentación a tu izquierda, sigue atentamente sus instrucciones.

Visitar Sitio

Clickea en visitar para visitar tu sitio.

Visitar

(b) Menú principal de la extensión.

Figura 7.2: Extensión de experimentación utilizada.

Fallos programados Tratándose de un experimento controlado, con el objetivo de evaluar la eficacia de las técnicas de mantenimiento de referencias DEOI, fue necesario programar una serie de fallos programados, que consistían en *simular* cambios en las estructuras de los DOM Trees de modo que los artefactos que los referenciaban fallen en su ejecución requiriendo actualizar sus referencias.

Tareas

Independientemente de las tareas iniciales relativas a la instalación de la extensión experimental y la correspondiente inscripción en la experimen-

tación, los participantes tuvieron que realizar dos tareas en el hogar: visitar los sitios webs dando lugar a que los artefactos de aumentación fueran ejecutados y en el caso de los artefactos de aumentación basados en el framework CrowdMock, debían intentar mantener las referencias utilizando la herramienta ADRM cuando los artefactos fallaran, consecuencia de la eventual ejecución de los fallos programados.

Al momento de registrarse en el experimento, a cada participante le fue asociado uno de los cuatro artefactos desarrollados. En orden de llegada, los participantes fueron agrupados por artefacto, dando lugar a cuatro grupos de 12 participantes para cada artefacto.

Una vez finalizada la experimentación, debieron confirmar la desinstalación de la extensión de su explorador, que fue sugerida automáticamente.

Hipótesis y variables

El experimento tiene un único objetivo: comparar la cantidad de ejecuciones fallidas y exitosas de los artefactos por parte de los participantes cuando utilizaron artefactos cuyo mantenimiento fue realizado de modo tradicional y utilizando CrowdMock. De este modo las hipótesis resultan:

- H_0 : No se producen diferencias en la cantidad de ejecuciones fallidas y exitosas consecuencia de la utilización de CrowdMock y técnicas tradicionales en el mantenimiento de los artefactos.
- H_a : Se producen diferencias en la cantidad de ejecuciones fallidas y exitosas al usar CrowdMock y técnicas tradicionales.

Tres tipos de variables pueden definirse para el experimento: independiente, dependiente, de control. La técnica utilizada (CrowdMock, Tradicional) es una variable independiente. El resultado de ejecución (*fallido*, *exitoso*) es la variable dependiente ya que se desea medir cómo varía en relación a la técnica de mantenimiento utilizada. Finalmente, tanto los artefactos como los fallos programados provistos configuran una variable de control, ya que

son un conjunto limitado y compartido que los participantes debieron utilizar, haciendo comparables las mediciones.

Procedimiento

Antes de comenzar el experimento, se introdujo a los participantes en el experimentación en general y se les indicó como instalar la extensión y completar su registro en la experimentación, para lo que dispusieron de 6 días previo al inicio de las actividades del experimento y 2 días posteriores de sincronización.

El experimento fue organizado en dos fases. Al inicio de cada fase, se introdujo un fallo programado con el objetivo de hacer fallar los artefactos. Este fallo se sostuvo durante 2 días consecutivos, posterior a los cuales, los artefactos volvieron a funcionar normalmente durante un igual periodo de tiempo, para dar lugar a una segunda fase con las mismas características.

Un grupo de participantes debió utilizar CrowdMock para intentar reparar las referencias durante los días en los que los fallos programados hacían fallar la ejecución de los artefactos. El otro grupo de participantes, definido por la utilización de artefactos mantenidos de modo tradicional, simplemente visitaba los sitios webs, sin tener oportunidad de reparar las referencias por ellos mismos, como sucede en dicho enfoque.

En total se requirieron 10 días consecutivos para realizar la experimentación y 8 días para tomar las muestras en donde los artefactos estuvieron defectuosos el 50 % del tiempo. La Figura 7.3 ilustra este procedimiento.

Procedimiento de análisis

Con los datos obtenidos sobre la cantidad de ejecuciones exitosas y fallidas producidas por los artefactos de aumentación durante las fases uno y dos, en las distintas técnicas utilizadas para su mantenimiento, se realizó una prueba de aceptación de hipótesis χ^2 [85] de independencia de grupos, ya que

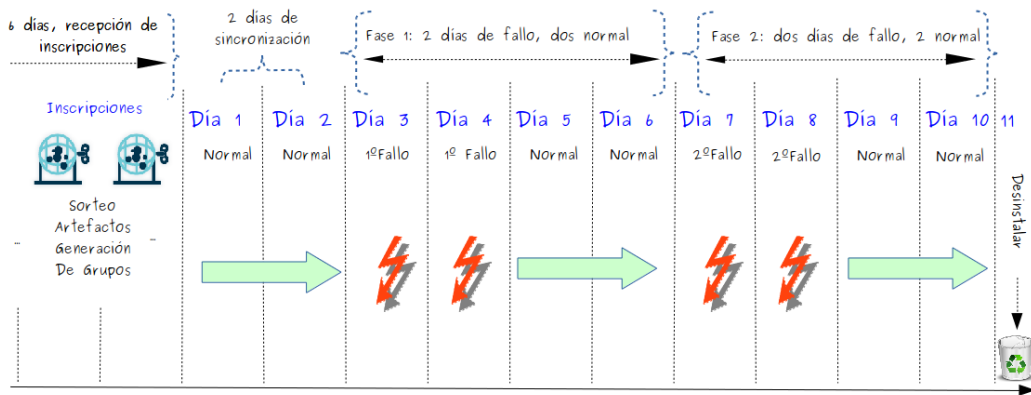


Figura 7.3: Procedimiento experimentación mantenimiento.

se trata de variables cualitativas, dicotómicas y nominales. Para ello se confeccionó una tabla de contingencia con las muestras obtenidas a partir del tercer día (inclusive).

7.2.2. Ejecución

En esta sección se describen aspectos relacionados a la ejecución del experimento, que fue realizada de acuerdo al procedimiento previamente descrito.

Transcurridos los seis días previstos para el proceso de inscripción, 48 participantes se registraron en la experimentación, generando 4 grupos de 12 participantes. Los participantes visitaron los sitios webs aumentados diariamente según fue establecido en el protocolo. Habiéndoles sido solicitado visitar el sitio web al menos una vez al día, la Tabla 7.6 muestra la cantidad de ejecuciones registradas para cada artefacto durante las fases uno y dos, donde fue posible tomar un total de 677 muestras.

La Tabla 7.7 muestra el resultado de las distintas ejecuciones para cada artefacto, en relación a si fueron exitosas o fallidas. Puede apreciarse que en el caso del artefacto de Aumentación Web para YouTube que utilizaba CrowdMock, solo fueron registrados 2 fallos, resultando ello el mínimo po-

Tabla 7.6: Cantidad de registros de utilización de artefactos.

Sitio	Versión Tradicional	Versión CrowdMock
YouTube	205	109
Wikipedia	187	176

sible, ya que en cada una de las fases del experimento, el primer usuario en utilizar el artefacto una vez introducidos los fallos, generó las nuevas referencias y las compartió con el resto del grupo. En el caso del artefacto de Aumentación Web para Wikipedia que utilizaba CrowdMock, se registraron 4 fallos, dos por cada fallo programado. En este caso en particular, durante la primer fase, sucedió que dos usuarios visitaron el sitio al mismo momento que el artefacto necesitaba mantenimiento. Se verificó que ambos usuarios aportaron nuevas referencias sin inconveniente, ya que el enfoque no genera competencias, todas las referencias nuevas fueron compartidas con el resto del grupo. En el caso de la segunda fase, también se registraron 2 fallos para este artefacto. A diferencia de la primer fase, solo uno de estos usuarios aportó nuevas referencias.

Las Figuras 7.4 y 7.5 muestran las ejecuciones exitosas y fallidas en

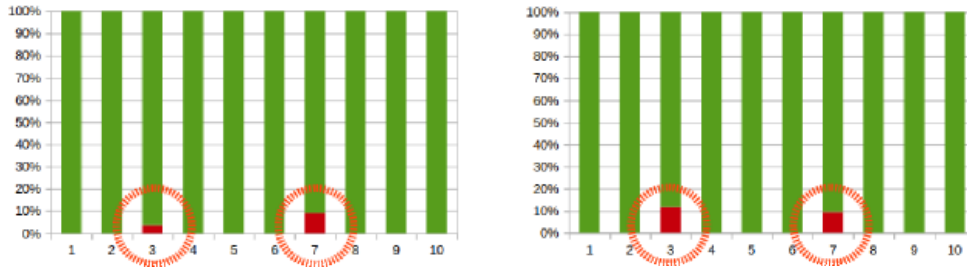
Tabla 7.7: Resultado de ejecuciones por artefacto.

Artefacto	Ejecuciones exitosas	Ejecuciones fallidas
Youtube (Tradicional)	85	120
Youtube (CrowdMock)	107	2
Wikipedia (Tradicional)	106	81
Wikipedia (CrowdMock)	172	4

términos relativos para cada artefacto. En relación a los artefactos mantenidos con el enfoque tradicional, Figuras 7.5a y 7.5b, es preciso notar que fueron registradas ejecuciones fallidas incluso después de que los artefactos fueron mantenidos en sus referencias. Esto se debe a que los artefactos de Aumentación Web en el enfoque tradicional, son actualizados al momento de que el explorador es iniciado. En los casos en los que los usuarios no reiniciaron el explorador, no tuvieron oportunidad de utilizar el artefacto actualizado.

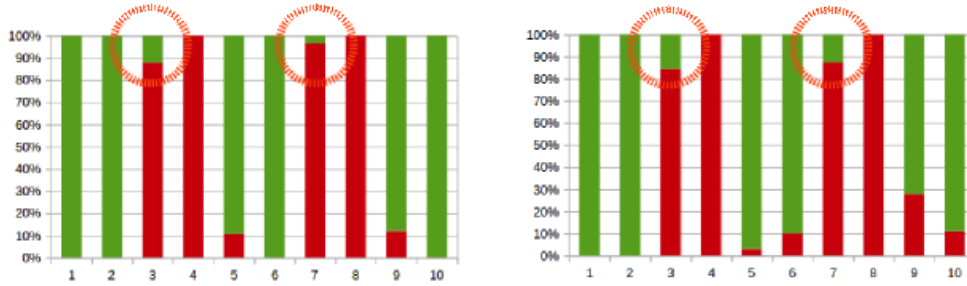
Así mismo, vale la pena mencionar que en el caso de los artefactos man-

tenidos de modo tradicional, se obtuvieron muestras de ejecuciones fallidas en todos los días posteriores a la incorporación del primer fallo programado.



(a) Ejecuciones YouTube CrowdMock. (b) Ejecuciones Wikipedia CrowdMock.

Figura 7.4: Ejecuciones fallidas (rojo) y exitosas (verde) por artefacto y día en términos relativos. (versión CrowdMock)



(a) Ejecuciones YouTube Tradicional. (b) Ejecuciones Wikipedia Tradicional.

Figura 7.5: Ejecuciones fallidas (rojo) y exitosas (verde) por artefacto y día en términos relativos. (versión Tradicional)

Finalmente la Tabla 7.8 resume la cantidad de ejecuciones exitosas y fallidas para cada tipo de artefacto durante las fases uno y dos.

Tabla 7.8: Resultado de ejecuciones por técnica de mantenimiento.

Técnica mantenimiento	Ejecuciones exitosas	Ejecuciones fallidas
Tradicional	191	201
CrowdMock	279	6

7.2.3. Análisis

En el protocolo del experimento fueron definidas las hipótesis, las variables y el procedimiento de análisis a llevar adelante, cuyos resultados se describen en esta sección.

Se analizaron las muestras obtenidas durante las fases uno y dos construyendo la correspondiente tabla de contingencia que se muestra en la Tabla 7.9. Con un estadístico $\chi^2 = 185$ (Figura 7.6), esta prueba de hipótesis establece que la hipótesis nula debe ser rechazada cuando el estadístico χ^2 resultare superior al establecido en tablas para un nivel de confianza desaseado. Las tablas indican un valor crítico $\chi^2 = 6,635$ para $p = 0,01$; por lo tanto, la hipótesis nula es rechazada.

Habiendo rechazado la hipótesis nula, puede afirmarse con un nivel de

Tabla 7.9: Tabla de contingencia fases 1 y 2.

Técnica mantenimiento	Tradicional	CrowdMock	Totales
Fallidos	279	6	285
Exitosos	191	201	392
Totales	470	207	677

confianza superior al 99% que el enfoque tradicional y el enfoque CrowdMock, generan distintas cantidades de ejecuciones fallidas. Al analizar descriptivamente las muestras obtenidas, puede apreciarse que en el enfoque tradicional, la cantidad de ejecuciones fallidas se encuentra claramente relacionada con la cantidad de tiempo que el artefacto de actualización se mantuvo desactualizado. En contraste, en CrowdMock, los usuarios del artefacto tuvieron oportunidad de descubrir nuevas referencias y compartirlas con el resto de los usuarios, de modo que no habiendo necesidad de actualizar el artefacto de actualización, los usuarios lograron ejecutarlo satisfactoriamente en la gran mayoría de los casos.

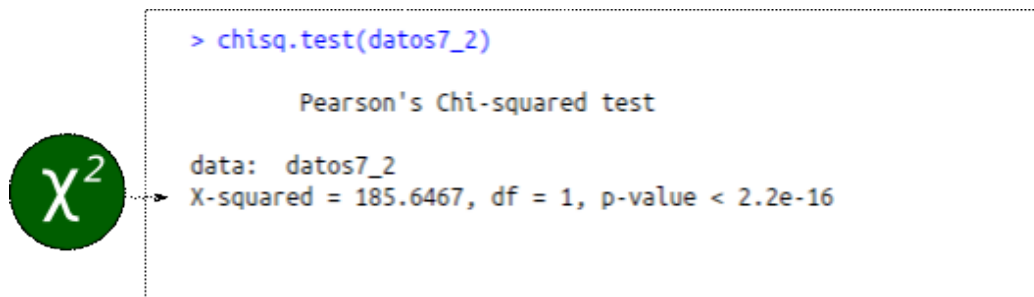


Figura 7.6: Estadístico χ^2 calculado en R.

7.2.4. Validez

En relación a la validez del experimento es importante destacar que en el análisis de su resultado se utilizó un test estadístico de validación de hipótesis no paramétrico como χ^2 , que al ser no paramétrico, no requiere ninguna certeza acerca de las distribuciones que describen las poblaciones de las variables analizadas. Por otro lado, este test requiere variables nominales, como lo fue el resultado de ejecución, donde los usuarios no pudieron interferir en el proceso de muestreo, ya que el resultado de las ejecuciones fue registrado automáticamente.

En relación a la heterogeneidad/homogeneidad de los participantes, si bien se trató de un grupo homogéneo en relación a ser todos ellos estudiantes universitarios, se considera que los participantes conformaban un grupo heterogéneo al azar en relación a sus conocimientos y habilidades, ya que voluntariamente participaron estudiantes de todos los años del plan de estudio. La selección del artefacto a utilizar por cada uno de ellos fue realizada al azar, donde en orden de llegada fueron distribuidos entre los distintos artefactos, evadiendo así amenazas por selecciones restringidas a un determinado tipo de participantes.

Del mismo modo fueron evadidos desvíos por selección, al mantenerse los artefactos defectuosos por la misma cantidad de tiempo que se los mantuvo correctos.

Aunque es bien conocido que los voluntarios suelen estar motivados en participar, pudiendo generar desvíos, se considera que esta motivación no constituye una amenaza de selección, considerando la motivación natural

por la que un usuario final de este tipo de software se involucra en su construcción y mantenimiento: satisfacer su propia necesidad.

En relación a los participantes que tuvieron oportunidad de seleccionar nuevas referencias para los artefactos correspondientes al enfoque Crowd-Mock, los desvíos por maduración fueron evadidos debido a que los participantes no tenían ningún conocimiento acerca de cuándo los fallos sucederían. Por otro lado, los participantes no tenían ninguna experiencia anterior con del enfoque evaluado ni sus herramientas, se considera que se han evadido los desvíos de validación externa.

En relación a los artefactos que fueron ejecutados, si bien se trataba de artefactos conceptualmente sencillos, estos contenían elementos UI agregados y colectados, dando lugar a la necesidad de mantener los dos tipos de elementos referenciados por igual sobre sitios webs reales, evadiendo también así amenazas a la validez externa en la configuración del experimento.

Capítulo 8

Trabajos relacionados

En este capítulo se describen los trabajos relacionados con esta tesis doctoral. El enfoque propuesto y las técnicas que lo soportan, tienen interconexiones con varias áreas temáticas de interés y relevancia como personalización y adaptabilidad de aplicaciones webs, programación por usuarios finales, ingeniería de requerimientos y crowdsourcing.

8.1. Personalización y adaptabilidad por usuarios finales

Desde 1996, la mayoría de los artículos relativos a la adaptabilidad de sistemas hipermedia se encuentran enfocados en aplicaciones web [69] [13]. La mayoría de los métodos bien conocidos para el diseño de aplicaciones webs han incorporado mecanismos de adaptación. La utilización de perfiles de usuario [48] se ha convertido también en un concepto importante en aplicaciones webs adaptativas, como así también el diseño de sistemas de recomendación [71]. Sin embargo, el uso de la web no solo continúa creciendo dando lugar a los enfoques de adaptabilidad tradicionales, sino que también la forma en que la web es utilizada está cambiando, y como ha sido estu-

diado en el capítulo 2 de este trabajo, los usuarios finales de la web están resolviendo sus necesidades de personalización por ellos mismos utilizando técnicas de aumentación.

Varias comunidades de Aumentación Web se han constituido en la utilización de repositorios públicos donde comparten los artefactos de aumentación. Como se ha mencionado muchos de estos artefactos cuentan con miles de instalaciones y año a año, con un crecimiento sostenido, tanto la cantidad de artefactos como la cantidad de usuarios con y sin habilidades de programación ha ido en aumento. De cualquier modo, actualmente estas comunidades se encuentran desintegradas de los sitios webs en sí mismos, de modo que los dueños o responsables de los sitios webs aumentados no tienen ningún tipo de intervención en la producción y divulgación de este tipo de software. Desde la academia, en los últimos años, han surgido trabajos de investigación acerca del potencial en la utilización de la programación por usuarios finales con el objetivo de permitir a los usuarios personalizar las aplicaciones [24, 36]. La participación de las masas de usuarios es presentada como una alternativa propicia para la personalización [4]. Trabajos como [36] hacen foco en proveer herramientas a los usuarios finales para la realización de tareas que involucran la utilización de varios sitios webs. Otros autores han propuesto un enfoque para dar soporte los usuarios finales en la definición de sus artefactos utilizando lenguajes para dominios específicos [25]. Los mismos autores han definido el enfoque WebMakeUp [22, 61], aportando herramientas para dar soporte a los usuarios finales a la hora de construir aumentaciones webs basadas en *mashups*.

Si bien se considera que estos enfoques han constituido importantes contribuciones, en la mayoría de ellos el nivel de habilidades requerido a los usuarios finales es considerable, limitando el número de usuarios que pueden verse alcanzados. En los enfoques WYSIWYG como WebMakeUp, donde no se requieren habilidades específicas por parte de los usuarios finales, el nivel de expresividad alcanzado se encuentra limitado al considerar que muchos de los artefactos de Aumentación Web no pueden ser alcanzados utilizando solo este tipo de herramientas, debido a que los requerimientos funcionales a los que dan respuesta, requieren de una lógica compleja que no puede ser expresada.

El enfoque propuesto intenta cubrir esta brecha, considerando que existe evidencia suficiente en la posible colaboración entre los usuarios con y sin habilidades de programación. Permitiéndoles a los usuarios finales sin habilidades especificar sus requerimientos de Aumentación Web en función de un metamodelo, los usuarios finales con habilidades de programación podrán completar la implementación de los artefactos, partiendo de la transformación de dichas especificaciones en artefactos de aumentación parcialmente funcionales.

Existen estudios que señalan la importancia de involucrar a las comunidades de desarrolladores en la creación de artefactos para adaptar las aplicaciones webs existentes [76] e incluso estudios que explican el rol de estos desarrolladores [43], como así también existen estudios que analizan cómo estas comunidades utilizan las herramientas de las que disponen [75].

Otros trabajos intentan dar solución a problemáticas particulares relativas a este tipo de software. Por ejemplo, algunos autores han estudiado formas de lograr que los artefactos de Aumentación Web sean más robustos en relación a tolerar cambios en los sitios web aumentados [26]. Los mismos autores han propuesto un modelo de seguridad para que los dueños de los sitios webs puedan controlar las alteraciones que los artefactos de aumentación realizan sobre sus sitios, a través de interfaces [3]. Sin embargo, no se encontraron precedentes en relación a un enfoque de adaptación como el propuesto, que permita el involucramiento de todos los actores: los dueños de los sitios webs, los programadores y los usuarios finales.

8.2. Ingeniería de requerimientos y crowdsourcing

La utilización de prototipos para la definición de requerimientos es una tendencia en crecimiento que puede ser apreciada simplemente al observar la cantidad de herramientas para la construcción de prototipos basadas en apli-

caciones webs y también para escritorio, como Balsamiq y *MockingBird*¹, que han aparecido en los últimos años. Estudios estadísticos han probado que el uso de prototipos puede mejorar todo el proceso de desarrollo en comparación con el uso de otro tipo artefactos para la especificación de requerimientos [66]. La utilización de prototipos ha sido introducida en diferentes enfoques y metodologías. Por un lado, han sido incluidos como un artefacto para la especificación de requerimientos esencial en las metodologías ágiles [30, 78]. Por otro lado, han sido utilizados como una forma formal de generar especificaciones en enfoques dirigidos por modelos [68]. En este trabajo se ha propuesto especificar alteraciones a realizar sobre los sitios webs aumentados, utilizando prototipos y técnicas de Aumentación Web tradicionales, incorporando las capacidades bien conocidas de anotación de este tipo de herramientas, que también pueden ser utilizadas para expresar requerimientos [58].

Aunque CrowdMock se enfoca en la construcción de artefactos de Aumentación Web, existen enfoques afines en relación a la colaboración en la gestión de los requerimientos, en la especificación de los mismos e incluso en su descubrimiento [32]. En [72] y [20] se proponen enfoques similares en este sentido, donde son involucrados muchos interesados para identificar, discutir y priorizar los requerimientos. Sin embargo estos enfoques difieren con CrowdMock porque proponen una serie de pasos más rígidos a los que se ha considerados poco apropiados para un proceso colaborativo, donde un enfoque más flexible resulta más conveniente. Estos enfoques requieren incluso de dos pasos fundamentales: primero construyen una estructura de interesados y luego éstos trabajan en los requerimientos. Esta característica impide que un nuevo interesado se involucre una vez que la estructura de interesados fue definida, mientras en CrowdMock, los interesados pueden involucrarse en cualquier momento durante la definición del requerimiento, e incluso posteriormente. Otra característica distintiva de CrowdMock, es que, de acuerdo con [81], los interesados pueden jugar dos roles muy importantes: pueden solicitar nueva funcionalidad y también puede proveer nueva funcionalidad. Ya que los usuarios con habilidades de programación pueden involucrarse en el refinamiento de los requerimientos.

En relación a la priorización de los requerimientos, mientras otros en-

¹<https://gomockingbird.com/>

foques para el descubrimiento de requerimientos a gran escala como el propuesto en [50], utilizan escalas de 0 a 5, en CrowdMock se prefirió simplificar la priorización a través de la técnica *like*, ya que ha sido mostrado que resulta más efectiva para seleccionar los requerimientos más importantes [70]. De acuerdo con [86], la colaboración en el proceso de definición de los requerimientos, resulta muy relevante en relación a obtener una diversidad representativa de refinamientos, que en otros enfoques como [64] son ignorados por completo.

En [77] se propone un enfoque similar a CrowdMock, donde son considerados ciclos iterativos en los que diferentes interesados producen y refinan modelos abstractos y representaciones concretas (escenarios, maquetas, historias). La diferencia radica en que se requiere de un rol de moderador. Otros enfoques [50], no solo agregan nuevos roles, sino que también incluyen un análisis más complejo de los interesados basado en redes sociales, donde los interesados son priorizados utilizando una variedad de medidas para considerar sus votos de distintos modos. Aún más, en el caso de [65], se proponen mecanismos para la identificación de interesados maliciosos, o *no interesados*, mientras que CrowdMock, se basa en la premisa de que los interesados tienen el mismo nivel de influencia y que colaboran intentando canalizar su propia necesidad sin malicia.

Capítulo 9

Conclusiones

Habiendo transcurrido un plan de trabajo de cuatro años, iniciado durante el año 2013, existieron en su transcurso varios cambios y nuevos aportes en relación a la temática estudiada. Entre estos cambios, lamentablemente, a principios del año 2014 el repositorio de artefactos de Aumentación Web por usuarios finales más importante resultaba desmantenido.

Sin embargo, afortunada y rápidamente surgieron nuevos repositorios para contener a esta comunidad. Aunque resulte trivial, este simple hecho y evidencia de pronta recuperación, da cuenta de la importancia que las técnicas de Aumentación Web tienen para miles de usuarios de la web. Estos usuarios no hubieran podido canalizar sus necesidades de otro modo, ya que las técnicas de Aumentación Web por usuarios finales, ofrecen la posibilidad de obtener soluciones que de ningún otro modo podrían haber sido alcanzadas. Como fue estudiado y documentado, estas comunidades crecen a un ritmo sostenido, duplicando año a año la cantidad de artefactos y la cantidad de usuarios con habilidades para crearlos y compartirlos. Sin embargo, se considera que en términos relativos, la Aumentación Web por usuarios finales aún no alcanzado todo su potencial. Aunque esta técnica ha ganado popularidad, muchísimos usuarios de la web simplemente desconocen su existencia (Anexo B). En este trabajo se ha intentado proveer un enfoque atento a las limitaciones propias de estas comunidades, en relación a su visibilidad, expansión y crecimiento. Independientemente de las contribuciones

desde el ámbito académico, como este estudio y otros relacionados, vale la pena mencionar que, aun a la fecha de estas conclusiones, los artefactos de Aumentación Web por usuarios finales, se construyen, comparten y mantienen del mismo modo que al inicio de este trabajo.

A la par, diversos aportes académicos en los últimos años han surgido, en donde hasta incluso ha sido sugerida la posibilidad de crear un foro/conferencia exclusivo para esta temática. En relación a ello y como fruto de este trabajo, fueron realizadas varias publicaciones científicas que comprenden gran parte los distintos capítulos de esta tesis. En este capítulo se presentan sus conclusiones, desde el punto de vista de los objetivos planteados y las contribuciones alcanzadas, y finalmente se plantea una serie de trabajos futuros.

9.1. Objetivos y contribuciones

El objetivo general de esta tesis doctoral consistió en:

“Investigar y generar los métodos de desarrollo y las arquitecturas de software necesarias, para facilitar la creación en colaboración mediante crowdsourcing, de artefactos de Aumentación Web por parte de usuarios finales, que se acoplen a las aplicaciones webs existentes, facilitando su definición, su diseminación y su mantenimiento a la masa de usuarios.”

Como fue estudiado en los distintos capítulos de esta tesis, partiendo de un análisis exhaustivo de una de las comunidades de Aumentación Web por usuarios finales más relevantes de la actualidad, fue propuesto un proceso conceptual para la construcción de este tipo de software, basado en la delegación de sus actividades a la masa de usuarios, contemplando los aspectos a los que se han considerado claves para el efectivo involucramiento de sus usuarios. Este proceso fue soportado por un conjunto de herramientas experimentales a las que se ha tenido oportunidad de evaluar en su efectividad,

obteniendo resultados que no solo confirman la validez del enfoque, sino que además, resultaron alentadores respecto del gran potencial que las técnicas de Aumentación Web tienen para ofrecer.

En lo subsiguiente se retoman los objetivos específicos analizando su cumplimiento y los aportes realizados en consecuencia de ello.

9.1.1. Requerimientos de Aumentación Web

El primero de los objetivos específicos planteados, relativo a los requerimientos de Aumentación Web, fue definido como:

“Diseñar e implementar métodos de relevamiento de requerimientos de artefactos de Aumentación Web por parte de la masa de usuarios.”

La definición de los requerimientos de aumentación, constituye una pieza clave del enfoque propuesto. Son los usuarios finales, con y sin habilidades de programación, quienes sostienen necesidades de aumentación sobre los sitios webs aumentados. Para cumplir con este objetivo fueron diseñadas y desarrolladas las herramientas experimentales MockPlug y UserRequirements.

MockPlug, una herramienta implementada con técnicas de Aumentación Web, permite a la masa de usuarios definir sus requerimientos en términos de RAMs. Mientras los RAMs constituyen una novedad en términos de artefactos para la especificación de requerimientos, la masa de usuarios puede refinarlos y priorizarlos a través de la utilización del repositorio UR.

El capítulo 4 describe estas actividades y herramientas, que pudieron ser evaluadas en su efectividad como es descripto en el capítulo 7. La primer versión de esta etapa del enfoque y sus resultados experimentales preliminares, fueron publicados en [33] durante el año 2014. En [34], a mediados del año 2016, fueron publicados los resultados experimentales presentados en la sección 7.1.

9.1.2. Construcción y prueba

El segundo objetivo específico, relativo a la construcción y a la prueba de los artefactos, fue definido como:

“Diseñar e implementar métodos de construcción y prueba artefactos de aumentación a través de crowdsourcing.”

La construcción y la prueba de los artefactos de aumentación, fue soportada desde la definición de un metamodelo que da lugar a la transformación de sus instancias, como en los enfoques dirigidos por modelos bien conocidos. La masa de usuarios a través de llamadas abiertas, es convocada completar y probar los artefactos. Los usuarios con y sin habilidades de programación, conducidos por su interés en satisfacer sus propios requerimientos, pueden colaborar en la construcción y la prueba de los artefactos de cara a la comunidad, permitiendo a toda la masa de usuarios conocer el estado de los artefactos. Para ello, la etapa del proceso que da lugar a estas actividades, fue diseñada considerando la posibilidad de proveer a los usuarios de herramientas que faciliten la realización de estas tareas.

En el capítulo 5, fueron presentadas las implementaciones experimentales de estas herramientas, donde a través de MockPlug, CMFW y UserRequirements, los usuarios pueden obtener la versión inicial de los artefactos y los casos de prueba tanto para los modelos RAMs como para los artefactos. Los usuarios con habilidades de programación, disponen de un framework que les permite concentrarse en completar los aspectos relativos a el comportamiento del artefacto requerido y finalmente compartirlos con el resto de la masa de usuarios a través de UR. Parte fundamental de los aportes alcanzados en la compleción de este objetivo, en relación al metamodelo, fueron publicados en [33] durante el año 2014.

9.1.3. Distribución y mantenimiento

El tercer objetivo específico, relativo a la distribución y al mantenimiento de los artefactos, fue definido como:

“Diseñar e implementar métodos de diseminación y mantenimiento de artefactos de aumentación a través de crowdsourcing.”

Las actividades relativas a la distribución y al mantenimiento de los artefactos de aumentación entre la masa de usuarios, fueron contempladas considerando dos aspectos muy débiles en la Aumentación Web por usuarios finales tradicional. Uno de ellos, es que en la Aumentación Web tradicional, los responsables de los sitios webs son ajenos al proceso de aumentación. Al contemplar el involucramiento de los responsables de los sitios webs (en la certificación de los artefactos), estos cuentan con la posibilidad de ofrecer a sus usuarios un nuevo mecanismo de personalización, a la vez de sostener control en la implementación de las políticas propias de su dominio. Negociando con la comunidad, cuáles de los artefactos de aumentación generados, son propicios para ser ofrecidos a toda la masa de usuarios, los usuarios de los sitios webs pueden descubrir y seleccionar aquellos artefactos de aumentación de su interés, con la certeza de que éstos fueron avalados en su utilización por los propios responsables de los sitios webs aumentados.

Por otro lado, como fue estudiado, los artefactos de aumentación tradicionales son muy rígidos en relación a las técnicas de referenciación utilizadas. Esta rigidez, constituye una gran debilidad y ha sido una problemática reconocida por la comunidad académica. En la búsqueda de cumplir con este objetivo específico, se han diseñado e implementado mecanismos para desacoplar las referencias de los artefactos a los sitios webs aumentados, posibilitando su posterior mantenimiento a toda la masa de usuarios.

En el capítulo 6, fueron descritas las actividades y las herramientas experimentales diseñadas e implementadas hacia el alcance de este objetivo. Mientras en el apartado 7.2 se presenta el resultado obtenido en la evaluación de las herramientas para el mantenimiento de referencias, el enfoque de

adaptación negociada que hace posible que todos los actores sean involucrados, fue publicado en [35] durante el año 2015.

9.1.4. Integración

Finalmente, el objetivo de integración de las actividades y las herramientas que soportan el proceso propuesto fue enunciado de la siguiente manera:

“Obtener un framework para la generación de artefactos de Aumentación Web que implemente un workflow basado en los métodos obtenidos.”

Independientemente de que la masa de usuarios pueda descubrir y definir sus requerimientos de aumentación, como así también construir los artefactos en relación a ello, e incluso involucrarse en su mantenimiento, el enfoque propuesto parte de que estas capacidades, se encuentran soportadas por herramientas cuya utilización es enmarcada en la realización de actividades propias de un proceso establecido, que requiere de cierta sincronización y cierto nivel de integración en sus herramientas.

CrowdMock y las herramientas experimentales implementadas que lo soportan, constituyen así un framework para el desarrollo de artefactos de Aumentación Web por usuarios finales, en donde las actividades a realizar son delegadas a la masa de usuarios por la propia masa de usuarios interesada.

Los capítulos 3 al 6, describen las herramientas que soportan el proceso y la integración de sus distintos componentes de aplicación. Gran parte de estos capítulos fue publicada en [33–35]. Estas publicaciones son descriptas en el Anexo A.

9.2. Trabajo futuro

Mientras al momento de estas conclusiones se está trabajando en la publicación de los aspectos del enfoque relativos al mantenimiento de referencias por parte de la masa de usuarios y los resultados experimentales obtenidos en su evaluación, en esta sección se consideran posibles trabajos futuros.

Las herramientas experimentales desarrolladas no se encuentran lejos de poder convertirse en herramientas productivas. Sin embargo, lamentable y positivamente Mozilla Firefox ha cambiado en sus últimas versiones el soporte para la construcción de extensiones. El soporte a las extensiones construidas con Mozilla Add-on SDK (como es el caso de MockPlug), será retirado a fines del año 2017, dando lugar a un nuevo tipo de soporte que será compatible con el resto de los exploradores. Debido a la importancia de MockPlug para el enfoque en general y posibles líneas futuras de investigación relacionadas, este trabajo de migración se considera ineludible y constituirá un esfuerzo considerable.

Distintos aspectos relativos a la usabilidad y expresividad de las herramientas experimentales se consideran importantes de ser superados. En el caso de MockPlug, los widgets colectados no son en la actualidad inspeccionados para considerar su naturaleza en todos los casos. Se considera posible mejorar la expresividad de MockPlug, de modo que al coleccionar elementos, ya sea en el sitio web aumentado, o en otros sitios webs (a la hora de utilizar widgets de bolsillo), sea inspeccionada la naturaleza de los elementos colectados, con el fin de poder editar sus propiedades en consecuencia.

La posible interoperabilidad entre distintos artefactos de aumentación no fue estudiada y se considera que partiendo de un soporte como el alcanzado experimentalmente durante este trabajo, es posible facilitar la interoperabilidad de múltiples artefactos de aumentación sobre un mismo sitio.

No existen trabajos que solucionen posibles problemas de competencias entre distintos artefactos de aumentación sobre los elementos de los sitios webs aumentados. Se considera que el framework desarrollado es un buen punto de partida para poder auto detectar potenciales incompatibilidades entre artefactos de aumentación automáticamente, por poder el framework conocer y explotar con ese objetivo, la información relacionada con los ele-

mentos DEOI referenciados.

Del mismo modo, se considera que es posible potenciar la reusabilidad del comportamiento desarrollado por los usuarios con habilidades de programación.

En relación a la seguridad, el enfoque propuesto supone no agregar ningún nuevo tipo de vulnerabilidad a los que las comunidades de Aumentación Web por usuarios finales, se encuentran expuestas tradicionalmente. Aunque conceptualmente el enfoque propuesto, supone el involucramiento de los dueños de los sitios webs, y a partir de su colaboración en la certificación de los artefactos, se considera que los usuarios finales podrían hacer uso de estas técnicas con mayor seguridad, se considera posible el desarrollo de herramientas de inspección de código fuente de este tipo de artefactos de aumentación, en la búsqueda de prácticas potencialmente maliciosas que faciliten su inspección a los interesados.

Los resultados experimentales obtenidos en la definición de requerimientos de Aumentación Web, mostrados en la sección 7.1, hacen posible considerar la posibilidad de utilizar este nuevo tipo de artefactos de requerimientos (los RAMs), en los procesos de desarrollo de software tradicional. Los participantes que tuvieron oportunidad de participar en aquellas experimentaciones, reportaron haber definido sus requerimientos en la mitad del tiempo requerido para definirlos con otros artefactos tradicionales. Si bien el enfoque se encuentra enmarcado en la obtención de artefactos de aumentación, se considera que esta nueva forma de especificar requerimientos, puede ser utilizada en el desarrollo de aplicaciones webs tradicionales, y que este nuevo tipo de artefactos para la especificación de requerimientos, podría ser integrado con otras herramientas existentes para la gestión de requerimientos en los procesos tradicionales de desarrollo de software.

*Y de otra manera, y de otra,
y de otra...*

Javascript InternalError: too much recursion.

Bibliografía

- [1] ADOMAVICIUS, G., AND TUZHILIN, A. Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions, 2005.
- [2] ALGOSAIBI, A. A., ALBAHLI, S., AND MELTON, A. World Wide Web : A Survey of its Development and Possible Future Trends. *The 16th International Conference on Internet Computing and Big Data-ICOMP'15*, August (2015).
- [3] ARELLANO, C., DÍAZ, O., AND ITURRIOZ, J. Crowdsourced web augmentation: A security model. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) 6488 LNCS* (2010), 294–307.
- [4] ARELLANO, C., DÍAZ, O., AND ITURRIOZ, J. Opening personalization to partners: An architecture of participation for websites. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) 7387 LNCS* (2012), 91–105.
- [5] BARTH, A., FELT, A. P., SAXENA, P., AND BOODMAN, A. Protecting Browsers from Extension Vulnerabilities. *Ndss 147* (2010), 1315–1329.
- [6] BIGHAM, J. P. Accessmonkey: enabling and sharing end user accessibility improvements. *SIGACCESS Access. Comput.*, 89 (2007), 3–6.

- [7] BOLIN, M., WEBBER, M., RHA, P., WILSON, T., AND MILLER, R. C. Automation and customization of rendered web pages. *Proceedings of the 18th annual ACM symposium on User interface software and technology - UIST '05* (2005), 163.
- [8] BOSETTI, G., FIRMENICH, S., GORDILLO, S. E., ROSSI, G., AND WINCKLER, M. An End-User Development approach for Mobile Web Augmentation. *Mobile Information Systems 2017* (2017), 1–28.
- [9] BOSETTI, G. A., FIRMENICH, S., GORDILLO, S. E., AND ROSSI, G. An approach for building mobile web applications through web augmentation. *JOURNAL OF WEB ENGINEERING* 16, 1 (2017), 75–102.
- [10] BOUVIN, N. O. Unifying strategies for Web augmentation. *Proceedings of the tenth ACM Conference on Hypertext and hypermedia : returning to our diverse roots returning to our diverse roots - HYPERTEXT '99* (1999), 91–100.
- [11] BROOKS, T. A. Watch this: Greasemonkey the web, jul 2005.
- [12] BROOKS, T. A. No bad web pages: Reader empowerment and the Web. *Information Research* 11, 3 (apr 2006), 119–120.
- [13] BRUSILLOVSKY, P. Adaptive Hypermedia. *User modelling and User-Adapted Interaction* (2001).
- [14] BRUSILOVSKY, P. Methods and techniques of adaptive hypermedia. *User Modeling and User-Adapted Interaction* 11, 2-3 (jul 2001), 87–129.
- [15] CARLINI, N., FELT, A., AND WAGNER, D. An evaluation of the google chrome extension security architecture. *Proceedings of the 21st USENIX conference on Security symposium* (2012), 7.
- [16] CHANG, C.-H., KAYED, M., GIRGIS, R., AND SHAALAN, K. A Survey of Web Information Extraction Systems. *IEEE Transactions on Knowledge and Data Engineering* 18, October (2006), 1411–1428.
- [17] COHN, M. *User stories applied: For agile software development*. Addison-Wesley Professional, 2004.

- [18] CRESCENZI, V., MERIALDO, P., AND QIU, D. Crowdsourcing large scale wrapper inference. *Distributed and Parallel Databases* 33, 1 (2014), 95–122.
- [19] DEVOE, K. Innovations Affecting Us – What’s Greasemonkey, and Do I Want it in the Library? *Against the Grain* 20, 3 (jun 2008), 12.
- [20] DHEEPA, V., ARAVINDHAR, D. J., AND VIJAYALAKSHMI, C. A Novel Method for Large Scale Requirement Elicitation. *ISO Certified International Journal of Engineering and Innovative Technology* (2008).
- [21] DÍAZ, O. Understanding web augmentation. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* (2012), vol. 7703 LNCS, pp. 79–80.
- [22] DÍAZ, O., ALDALUR, I., ARELLANO, C., MEDINA, H., AND FIRME-NICH, S. WebMakeup: Empowering Users to Mod Websites. 1–4.
- [23] DÍAZ, O., AND ARELLANO, C. The Augmented Web: Rationales, Opportunities and Challenges on Browser-Side Transcoding. *ACM Transactions on the Web* 9, 2 (2015), 1–30.
- [24] DÍAZ, O., ARELLANO, C., ALDALUR, I., MEDINA, H., AND FIRME-NICH, S. End-User Browser-Side Modification of Web Pages. *Springer International Publishing Switzerland 2014* (2014).
- [25] DÍAZ, O., ARELLANO, C., AND AZANZA, M. A language for end-user web augmentation. *ACM Transactions on the Web* (2013).
- [26] DÍAZ, O., ARELLANO, C., AND ITURRIOZ, J. Interfaces for scripting: Making Greasemonkey scripts resilient to website upgrades. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* 6189 LNCS (2010), 233–247.
- [27] DJERIC, V., AND GOEL, A. Securing script-based extensibility in web browsers. *Proceedings of the 19th USENIX conference on Security* (2010), 23.

- [28] ESTELLÉS-AROLAS, E., AND GONZÁLEZ-LADRÓN-DE GUEVARA, F. Towards an integrated crowdsourcing definition. *Journal of Information Science* 38, X (2012), 1–14.
- [29] FERRARA, E., DE MEO, P., FIUMARA, G., AND BAUMGARTNER, R. Web data extraction, applications and techniques: A survey. *Knowledge-Based Systems* 70 (2014), 301–323.
- [30] FERREIRA, J., NOBLE, J., AND BIDDLE, R. Agile development iterations and UI design. In *Proceedings - AGILE 2007* (2007).
- [31] FILMAN, R. E. Taking Back the Web. *IEEE Internet Computing* 10, February (2006), 3–5.
- [32] FINKELSTEIN, A. StakeRare: Using Social Networks and Collaborative Filtering for Large-Scale Requirements Elicitation. *IEEE Transactions on Software Engineering* 38, 3 (may 2012), 707–735.
- [33] FIRMENICH, D., FIRMENICH, S., RIVERO, J., AND ANTONELLI, L. A platform for web augmentation requirements specification. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 8541. 2014.
- [34] FIRMENICH, D., FIRMENICH, S., RIVERO, J. M., ANTONELLI, L., AND ROSSI, G. CrowdMock: an approach for defining and evolving web augmentation requirements. *Requirements Engineering* (2016), 1–29.
- [35] FIRMENICH, D., FIRMENICH, S., ROSSI, G., WINCKLER, M., AND DISTANTE, D. *User interface adaptation using web augmentation techniques: Towards a negotiated approach*, vol. 9114. 2015.
- [36] FIRMENICH, S., ROSSI, G., AND WINCKLER, M. A Domain Specific Language for Orchestrating User Tasks Whilst Navigation Web Sites. *Springer, Heidelberg 7977*, July (2013), 224–232.
- [37] FIRMENICH, S., ROSSI, G., WINCKLER, M., AND PALANQUE, P. An approach for supporting distributed user interface orchestration over the Web. *International Journal of Human Computer Studies* (2014).

- [38] FIRMENICH, S., WINCKLER, M., ROSSI, G., AND GORDILLO, S. A framework for concern-sensitive, client-side adaptation. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* (2011), vol. 6757 LNCS, pp. 198–213.
- [39] FOWLER, M., AND FOEMMEL, M. Continuous integration. *ThoughtWorks*) [http://www.thoughtworks.com/Continuous Integration.pdf](http://www.thoughtworks.com/Continuous%20Integration.pdf) (2006).
- [40] GLINZ, M. On Non-Functional Requirements. In *Requirements Engineering, IEEE International Conference on* (2007).
- [41] GUHA, A., FREDRIKSON, M., LIVSHITS, B., AND SWAMY, N. Verified security for browser extensions. *Proceedings - IEEE Symposium on Security and Privacy* (2011), 115–130.
- [42] HOWE, J. The Rise of Crowdsourcing. *North 14*, 14 (2006), 1–5.
- [43] JONES, M. C., AND CHURCHILL, E. F. Conversations in developer communities. *Proceedings of the fourth international conference on Communities and technologies - C&T '09* (2009), 195.
- [44] KAPRAVELOS, A., GRIER, C., AND CHACHRA, N. Hulk: eliciting malicious behavior in browser extensions. *Proceedings of the 23rd ...* (2014).
- [45] KARIM, R., DHAWAN, M., GANAPATHY, V., AND SHAN, C. C. An analysis of the Mozilla jetpack extension framework. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* 7313 LNCS (2012), 333–355.
- [46] KELLY, S., AND TOLVANEN, J. P. *Domain-Specific Modeling: Enabling Full Code Generation*. 2007.
- [47] KO, A. J., MYERS, B., ROSSON, M. B., ROTHERMEL, G., SHAW, M., WIEDENBECK, S., ABRAHAM, R., BECKWITH, L., BLACKWELL, A., BURNETT, M., ERWIG, M., SCAFFIDI, C., LAWRENCE, J., AND

- LIEBERMAN, H. The state of the art in end-user software engineering. *ACM Computing Surveys* 43, 3 (2011), 1–44.
- [48] KOBSA, A. Generic user modeling systems. *User Modeling and User-Adapted Interaction* (2001).
- [49] KOTESKA, B., AND MISHEV, A. Agile Software Testing Technologies in a Large Scale Project. 121–124.
- [50] LIM, S. L., DAMIAN, D., AND FINKELSTEIN, A. StakeSource2.0: using social networks of stakeholders to identify and prioritise requirements. *2011 33rd International Conference on Software Engineering (ICSE)* (2011).
- [51] LIU, D., WANG, X., LI, H., AND YAN, Z. Robust Web Extraction Based on Minimum Cost Script Edit Model. *Procedia Engineering* 29, 0 (2012), 1119–1125.
- [52] LIU, L., ZHANG, X., YAN, G., AND CHEN, S. Chrome extensions: Threat analysis and countermeasures. ... *of the Network and Distributed Systems ...* (2012).
- [53] MALONE, T., AND CROWSTON, K. The interdisciplinary study of coordination. *Computing Surveys (CSUR)* (1994).
- [54] MAO, K., CAPRA, L., HARMAN, M., AND JIA, Y. A Survey of the Use of Crowdsourcing in Software Engineering. *Rn* 15 (2015), 1.
- [55] MCCORNACK, R. L. Extended Tables of the Wilcoxon Matched Pair Signed Rank Statistic. *Journal of the American Statistical Association* (1965).
- [56] MIRRI, S., SALOMONI, P., AND PRANDI, C. Augment browsing and standard profiling for enhancing web accessibility. *Proceedings of the International Cross-Disciplinary Conference on Web Accessibility - W4A '11* (2011), 1.
- [57] MITCHELL, R. *Web Scraping with Python: Collecting Data from the Modern Web*. O'Reilly Media, Inc., 2015.

- [58] MUKASA, K. S., AND KAINDL, H. An integration of requirements and user interface specifications. In *Proceedings of the 16th IEEE International Requirements Engineering Conference, RE'08* (2008).
- [59] NEBELING, M., LEONE, S., AND NORRIE, M. C. Crowdsourced Web Engineering and Design. *Web Engineering - Lecture Notes in Computer Science - Springer 7387* (2012), 31–45.
- [60] NEBELING, M., SPEICHER, M., AND NORRIE, M. CrowdAdapt: enabling crowdsourced web page adaptation for individual viewing conditions and preferences. *Proceedings of the 5th ACM SIGCHI symposium on Engineering interactive computing system* (2013), 23–32.
- [61] OSCAR DÍAZ , CRISTÓBAL ARELLANO, IÑIGO ALDALUR, HARITZ MEDINA, S. F. WebMakeup: An End-user Tool for Web Page Customization. 99–102.
- [62] PILGRIM, M. *Dive Into Greasemonkey*. 2005.
- [63] PILGRIM, M. *Greasemonkey Hacks: Tips & Tools for Remixing the Web with Firefox*. 2005.
- [64] PONZANELLI, L., BACCHELLI, A., AND LANZA, M. Leveraging crowd knowledge for software comprehension and development. In *Proceedings of the European Conference on Software Maintenance and Reengineering, CSMR* (2013).
- [65] REENADEVI, R., AND DUGALYA, P. Identifying Malicious Stakeholders Using Algorithm For Large Scale Requirement-Elicitation. *International Journal of Computer and Communication Technology* 3, 6 (2012), 106–108.
- [66] RICCA, F., SCANNIELLO, G., TORCHIANO, M., REGGIO, G., AND ASTESIANO, E. On the Effectiveness of Screen Mockups in Requirements Engineering: Results from an Internal Replication. *Proceedings of the 2010 {ACM-IEEE} International Symposium on Empirical Software Engineering and Measurement* (2010).

- [67] RIVERO, J. M., GRIGERA, J., ROSSI, G., ROBLES LUNA, E., MONTERO, F., AND GAEDKE, M. Mockup-Driven Development: Providing agile support for Model-Driven Web Engineering. *Information and Software Technology* 56, 6 (2014).
- [68] RIVERO, J. M., AND ROSSI, G. MockupDD: Facilitating agile support for Model-Driven Web Engineering. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* (2013).
- [69] ROSSI, G., SCHWABE, D., AND GUIMARÃES, R. Designing personalized web applications. *Proceedings of the tenth international conference on World Wide Web - WWW '01* (2001).
- [70] SAKAMOTO, Y., NICKERSON, J. V., BAO, J., SAKAMOTO, Y., AND NICKERSON, J. V. Evaluating Design Solutions Using Crowds. In *AMCIS 2011 Proceedings* (2011).
- [71] SCHAFER, J., FRANKOWSKI, D., HERLOCKER, J., AND SEN, S. Collaborative Filtering Recommender Systems. *The Adaptive Web* (2007), 291–324.
- [72] SHIMAKAGE, M., AND HAZEYAMA, A. A Requirement Elicitation Method in Collaborative Software Development Community. *International Conference on Product Focused Software Process Improvement 3009* (2004), 509–522.
- [73] SHULL, F., SINGER, J., AND SJØBERG, D. I. K. *Guide to advanced empirical software engineering*. 2008.
- [74] STOLBERG, S. Enabling agile testing through continuous integration. *Proceedings - 2009 Agile Conference, AGILE 2009* (2009), 369–374.
- [75] STOLEE, K. T., ELBAUM, S., AND SARMA, A. End-User Programmers and their Communities: An Artifact-based Analysis. *2011 International Symposium on Empirical Software Engineering and Measurement* (2011), 147–156.

- [76] STOLEE, K. T., ELBAUM, S., AND SARMA, A. Discovering how end-user programmers and their communities use public repositories: A study on Yahoo! Pipes. *Information and Software Technology* 55, 7 (2013), 1289–1303.
- [77] SUTCLIFFE, A. Collaborative requirements engineering: Bridging the gulfs between worlds. In *Intentional Perspectives on Information Systems Engineering*. 2010.
- [78] TON, H. A strategy for balancing business value and story size. In *Proceedings - AGILE 2007* (2007).
- [79] TRIESCHNIGG, D., TJIN-KAM-JET, K., AND HIEMSTRA, D. Ranking XPathS for extracting search result records. *CTIT Technical report* (2012).
- [80] VAN ACKER, S., NIKIFORAKIS, N., DESMET, L., PIESSENS, F., AND JOOSEN, W. Monkey-in-the-browser: Malware and vulnerabilities in augmented browsing script markets—extended version. *CW Reports* (2014), 525–530.
- [81] VUKOVIC, M. Crowdsourcing for Enterprises. *2009 Congress on Services - I* (jul 2009), 686–692.
- [82] WALKER, M., TAKAYAMA, L., AND LANDAY, J. A. High-Fidelity or Low-Fidelity, Paper or Computer? Choosing Attributes when Testing Web Prototypes. In *Proceedings of the Human Factors and Ergonomics Society Annual Meeting* (2002).
- [83] WHITTLE, J., HUTCHINSON, J., AND ROUNCEFIELD, M. The State of Practice in Model-Driven Engineering. *Software, IEEE* (2014).
- [84] WILLIGHAGEN, E. L., O’BOYLE, N. M., GOPALAKRISHNAN, H., JIAO, D., GUHA, R., STEINBECK, C., AND WILD, D. J. Userscripts for the life sciences. *BMC bioinformatics* 8 (2007), 487.
- [85] WOHLIN, C., RUNESON, P., HÖST, M., OHLSSON, M. C., REGNELL, B., AND WESSLÉN, A. *Experimentation in software engineering*. 2012.

- [86] WU, W., TSAI, W. T., AND LI, W. Creative software crowdsourcing: from components and algorithm development to project concept formations. *International Journal of Creative Computing* (2013).

Anexos

Anexo A

Publicaciones

A continuación, junto a una breve descripción, se enumeran las publicaciones científicas realizadas durante el transcurso del plan de trabajo de esta tesis doctoral.

- *Firmenich D., Firmenich S., Rivero J.M., Antonelli L. (2014) A Platform for Web Augmentation Requirements Specification. In: Casteleyn S., Rossi G., Winckler M. (eds) Web Engineering. ICWE 2014. Lecture Notes in Computer Science, vol 8541. Springer, Cham*

En este artículo, fueron publicadas las herramientas para la definición de requerimientos de aumentación, el metamodelo CrowdMock y los primeros resultados experimentales.

- *Firmenich D., Firmenich S., Rossi G., Winckler M., Distante D. (2015) User Interface Adaptation Using Web Augmentation Techniques: Towards a Negotiated Approach. In: Cimiano P., Frasincar F., Houben G.J., Schwabe D. (eds) Engineering the Web in the Big Data Era. ICWE 2015. Lecture Notes in Computer Science, vol 9114. Springer, Cham*

Esta publicación presenta el enfoque de adaptación negociada y las herramientas implementadas para la disseminación de artefactos de aumentación

contemplando una muestra de compatibilidad con artefactos tradicionales.

- *Firmenich, D., Firmenich, S., Rivero, J. M., Antonelli, L., and Rossi, G. (2016). CrowdMock: an approach for defining and evolving web augmentation requirements. Requirements Engineering, 1-29.*

En esta publicación se presenta un caso de estudio y los resultados experimentales obtenidos en la evaluación de la efectividad de las herramientas para la definición de requerimientos de aumentación.

Anexo B

Encuesta a usuarios de la web

En el mes de febrero del año 2017, fueron encuestados 52 usuarios de internet, ingresantes de la Facultad de Ingeniería de la UNPSJB. A todos ellos se les consultó primero si navegaban la Web regularmente, Figura B.1a y luego Figura B.1b: *¿En dónde están los sitios webs?*

Mientras el 87% de las personas encuestadas navegan la web diariamente, el 77% de ellas contestó que los sitios webs están en los servidores de Internet.

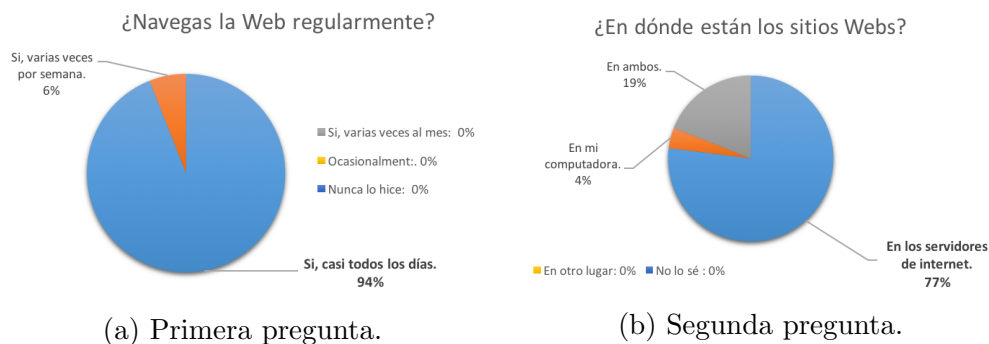


Figura B.1: Primera y segunda pregunta.

Inmediatamente después, Figura B.2a se les consultó: *¿Quién define cómo se ve y cómo funciona un sitio web?*

El 75 % de las personas encuestadas contestó los dueños y los creadores de los sitios webs. Mientras que el 4 % contestó el usuario que navega, y el 21 % restante contestó ambos.

En la última pregunta de la encuesta, Figura B.2b, les fue consultado: *En alguna ocasión: ¿hubieras preferido que un sitio web que has navegado, se vea o funcione de un modo distinto a como lo hizo?*

El 82 % de los encuestados contestó: *sí, me ha ocurrido.*

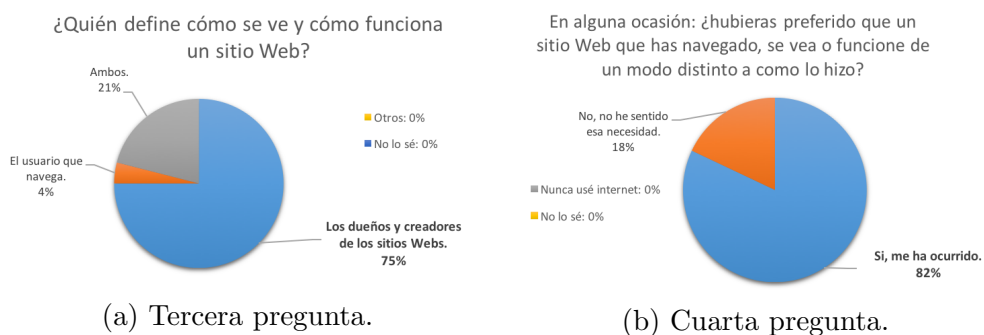


Figura B.2: Tercera y cuarta pregunta.

¿Navegas la Web regularmente?

- Si, casi todos los días.
- Si, varias veces por semana.
- Si, varias veces al mes.
- Ocasionalmente.
- Nunca lo hice.

¿En dónde están los sitios Webs?

- En los servidores de internet.
- En mi computadora.
- En ambos.
- En otro lugar.
- No lo sé.

¿Quién define cómo se ve y cómo funciona un sitio Web?

- Los dueños y creadores de los sitios Webs.
- El usuario que navega.
- Ambos.
- Otros.
- No lo sé.

En alguna ocasión: ¿hubieras preferido que un sitio Web que has navegado, se vea o funcione de un modo distinto a como lo hizo?

- Si, me ha ocurrido.
- No, no he sentido esa necesidad.
- Nunca usé internet.
- No lo sé.

Figura B.3: Formulario entregado a los encuestados en formato impreso.

Anexo C

Expresiones regulares

Detección referencias harcodeadas XPath

```
r"document\.\evaluate\s?\(\s?[\'\"](?:P<ref>[^\'\"]*)[\'\"]\).*\,"
```

Detección referencias harcodeadas estilo jQuery

```
r"\$\s?\(\s?[\'\"](?:P<ref>[^\$%[\]\<\'\"]*)[\'\"]\)\.*)"
r"\jQuery\s?\(\s?[\'\"](?:P<ref>[^\$%[\]\<\'\"]*)[\'\"]\)\.*)"
```

Detección referencias harcodeadas selectores CSS

```
r"document\.\querySelector\s?\(\s?[\'\"](?:P<ref>[^\'\"]*)[\'\"]\).*"
r"document\.\querySelectorAll\s?\(\s?[\'\"](?:P<ref>[^\'\"]*)[\'\"]\).*"
r"document\.\getElementsByClassName\s?\(\s?[\'\"](?:P<ref>[^\'\"]*)[\'\"]\).*"
```


Anexo D

Librerías de terceros

Nombre	Importaciones	Descripcion
jQuery	1628	jQuery is a fast, small, and feature-rich JavaScript library. It makes things like HTML document traversal and manipulation, event handling, animation, and Ajax much simpler with an easy-to-use API that works across a multitude of browsers. With a combination of versatility and extensibility, jQuery has changed the way that millions of people write JavaScript.

GM_config	28	GM.config is a user script library that allows the user to edit certain saved values through a graphical settings menu. This framework was specifically designed to abstract the creation of this menu so the script author could focus more time on integrating these saved values into their script. GM.config dynamically builds its graphical interface using the DOM API, which means it doesn't contain a single string of HTML.
jqplot	27	jqPlot is a plotting and charting plugin for the jQuery Javascript framework. jqPlot produces beautiful line, bar and pie charts with many features.
InstaSync	26	Basic function that are needed by several script in instasync.com
WaitForKey	25	A utility function, for Greasemonkey scripts, that detects and handles AJAXed content.
moment	20	Parse, validate, manipulate, and display dates in JavaScript.
MutationObserver	16	MutationObserver wrapper to wait for the specified CSS selector
tablesorter	11	Tablesorter is a jQuery plugin for turning a standard HTML table with THEAD and TBODY tags into a sortable table without page refreshes.
underscore	8	Underscore is a JavaScript library that provides a whole mess of useful functional programming helpers without extending any built-in objects.

cryptojs	7	JavaScript library of crypto standards.
TogglLib	7	Library for Toggl-Button scripts used on platforms like drupal, github, you-track and others.
jscolor	6	jscolor is a web color picker component that aims to be super easy both for developers to install and end users to use.
maps	4	Customize maps with your own content and imagery.
webfont	4	The Web Font Loader is a JavaScript library that gives you more control over font loading than the Google Fonts API provides.
toastr	3	toastr is a Javascript library for non-blocking notifications. jQuery is required. The goal is to create a simple core library that can be customized and extended.
filesaver	3	Pops up a file saver box in javascript
highcharts	3	Highcharts makes it easy for developers to set up interactive charts in their web pages
canvasjs	3	HTML5 JavaScript Charting Library with a simple API and 10x better performance. Charts are responsive & can run across devices including iPhone, Android, Desktops, etc.
jqModal	3	jqModal helps you display modals, popups, and notices. It is flexible and tiny, and provides a general-purpose base for all your windowing needs.
spectrum	3	The No Hassle jQuery Colorpicker

bootstrap	2	Bootstrap is the most popular HTML, CSS, and JS framework for developing responsive, mobile first projects on the web.
datejs	2	Datejs is an open-source JavaScript Date Library.
jszip	2	JSZip is a javascript library for creating, reading and editing .zip files, with a lovely and simple API.
prototype	1	Prototype takes the complexity out of client-side web programming. Built to solve real-world problems, it adds useful extensions to the browser scripting environment and provides elegant APIs around the clumsy interfaces of Ajax and the Document Object Model.
sugar	1	Sugar is a Javascript utility library for working with native objects.
video	1	Video.js is an open source library for working with video on the web, also known as an HTML video player
jscookie	1	A simple, lightweight JavaScript API for handling cookies
mustache	1	mustache.js is an implementation of the mustache template system in JavaScript.
marked	1	A markdown parser built for speed

liningjs	1	In CSS we already have the selector <code>::first-line</code> to apply style on the first line of element. But there is no selector like <code>::nth-line()</code> , <code>::nth-last-line()</code> or even <code>::last-line</code> . Then I read an article A Call for ::nth-everything from CSS tricks. <code>::nth-line()</code> is actually really useful in some situation.
localforage	1	localForage is a JavaScript library that improves the offline experience of your web app by using an asynchronous data store with a simple, localStorage-like API. It allows developers to store many types of data instead of just strings.
raphael	1	Raphaël is a small JavaScript library that should simplify your work with vector graphics on the web.
icheck	1	Highly customizable checkboxes and radio buttons (jQuery and Zepto)
keymaster	1	Keymaster is a simple micro-library for defining and dispatching keyboard shortcuts in web applications.
purl	1	A JS utility for parsing URLs and extracting information out of them.
zepto	1	Zepto is a minimalist JavaScript library for modern browsers with a largely jQuery-compatible API. If you use jQuery, you already know how to use Zepto.
accountingjs	1	accounting-js is a tiny JavaScript library for number, money and currency parsing/formatting.

scrollmagic	1	The javascript library for magical scroll interactions.
-------------	---	---

Anexo E

Artefactos analizados

El listado de los *50* artefactos analizados durante el análisis de alteraciones producidas por artefactos de aumentación puede ser descargado desde <https://goo.gl/aW465n>. La plantilla contiene los nombres de los artefactos, sus descripciones, URLs y la clasificación las alteraciones correspondiente.

El listado de los *200* artefactos analizados manualmente en sus códigos fuentes, durante el análisis de las técnicas de referenciación y cambios de referencias entre versiones, puede ser descargado desde <https://goo.gl/st66hL>. La plantilla contiene los nombres de los artefactos, sus descripciones, URLs y la clasificación de las técnicas de referenciación correspondiente.