Una Revisión Sistemática de la Literatura en Pruebas de Compatibilidad Web

Leandro N. Sabaren, Maximiliano A. Mascheroni, Cristina L. Greiner, Emanuel Irrazábal,
Departamento de Informática. Facultad de Ciencias Exactas y Naturales y Agrimensura.
Universidad Nacional del Nordeste. Corrientes, Argentina
leans177@gmail.com, {mascheroni, cgreiner, eirrazabal}@exa.unne.edu.ar

Abstract. Los usuarios acceden a aplicaciones web desde diferentes navegadores esperando obtener la misma experiencia de usuario. Sin embargo diversas causas producen defectos de compatibilidad que afectan negativamente las funcionalidades y la interfaz gráfica. En este trabajo se busca identificar, analizar y sintetizar las técnicas, herramientas y desafíos encontrados en artículos científicos publicados sobre pruebas de compatibilidad web. La metodología elegida es la revisión sistemática de la literatura. Se analizaron y sintetizaron los hallazgos para responder los interrogantes propuestos. La técnica de pruebas más elegida es el análisis visual. El principal desafío detectado es la detección de elementos variables. La compatibilidad web es cada vez más importante de acuerdo al crecimiento de las publicaciones, pero aún no se han desarrollado técnicas adecuadas soportadas por herramientas automatizadas.

Keywords: aplicación web, prueba de compatibilidad web, pruebas cruzadas de navegador, revisión sistemática de la literatura.

1 Introducción

Al desarrollar una aplicación web, un objetivo es que sea visualizada correctamente por la mayor cantidad de usuarios posible [1], [2]. Debido a la naturaleza de la distribución de las aplicaciones web, basadas en una arquitectura cliente servidor [3], los usuarios pueden acceder a la página web desde una gran variedad de navegadores, en dispositivos y plataformas distintas. Sin embargo, las diferencias entre los navegadores y su interpretación del código de la página web pueden causar defectos de incompatibilidad, que afectan negativamente la experiencia de usuario. Es trabajo del desarrollador el proveer una experiencia aceptable a los usuarios, aun cuando ésta no sea exactamente la misma para todos.

La compatibilidad web es la característica de una página web que permite que la aplicación funcione correctamente en un número aceptable de navegadores web [4]. Resulta imposible probar el funcionamiento de la aplicación para el 100% de navegadores, en todas las plataformas [4]. El sitio web será compatible si se logra la compatibilidad web con el conjunto de navegadores acordado en los requerimientos.

Existen herramientas industriales para las pruebas, como CrossBrowserTesting[1] o BrowserStacks[2], aunque es necesaria la inspección visual de las capturas que generan éstas por parte de un usuario. Cada vez más, investigadores han propuesto técnicas y herramientas para realizar las pruebas de compatibilidad en los últimos años. Mientras el estado del arte crece y se diversifica, surge la necesidad de revisar y resumir sistemáticamente las soluciones presentadas. Entonces, se propone un estudio en el dominio de pruebas de compatibilidad web llevando a cabo una revisión sistemática de la literatura, orientado a encontrar las técnicas de pruebas diseñadas, las herramientas implementadas y los desafíos encontrados.

2 Metodología de revisión

La metodología elegida es la revisión sistemática de la literatura (RSL), propuesta por Kitchenham y Charters [5]. Se identificó la necesidad para la revisión y se formularon las preguntas de investigación (PI). El protocolo de revisión se observa en la Fig. 1.

El objetivo es analizar trabajos académicos relacionados al tema de compatibilidad web, haciendo énfasis en las técnicas y herramientas propuestas. Las PI servirán como guía y el análisis del conocimiento encontrado buscará responder los interrogantes propuestos. Una síntesis del estado del arte servirá para trabajos y proyectos futuros en esta línea de investigación. Para ello se formulan las siguientes PI:

- PI 1 – ¿Cuáles son las técnicas propuestas para realizar pruebas de compatibilidad web?
- PI 2 – ¿Qué herramientas se proponen para detectar defectos de incompatibilidad en aplicaciones web?
- PI 3 – ¿Qué desafíos se enfrentaron al implementar las pruebas?

Para los términos de búsqueda en los repositorios, se trabajó de forma iterativa. Las palabras clave se obtuvieron de las PI y se tuvieron en consideración sinónimos que contribuyan al resultado final. Se realizaron búsquedas sucesivas, para ajustar los parámetros usados. La expresión de búsqueda obtenida es la siguiente: (web OR website) AND ("cross-browser" OR "cross browser" OR crossbrowser) AND (test OR testing OR defect OR failure OR issue) AND (technique OR method OR tool).

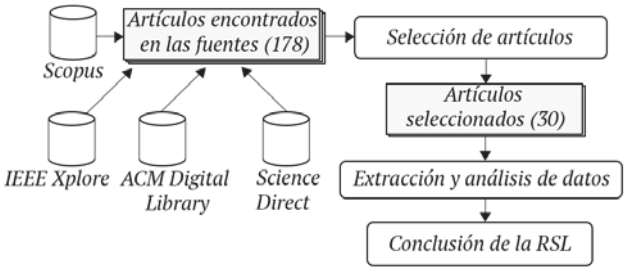

Fig. 1. Diagrama del protocolo de revisión utilizado en esta RSL.

[1] https://crossbrowsertesting.com/
[2] https://www.browserstack.com/

En la Fig. 1 se detallan las actividades propuestas del protocolo de revisión. Seleccionadas las fuentes se comienza la búsqueda con los términos definidos. A continuación se seleccionan los estudios mediante la aplicación de los criterios de inclusión y exclusión. Una vez obtenidos los artículos que se consideran relevantes, se extraen y analizan los datos para contestar las PI formuladas. Finalmente, se elabora una síntesis y las conclusiones de la RSL.
Para la recopilación de artículos, se seleccionaron repositorios utilizados en [6], [7], [8], [9]: Scopus, ACM Digital Library, IEEE Xplore, Science Direct.

Para descartar estudios irrelevantes se aplicaron criterios de inclusión y exclusión en cada publicación revisada. Se examinó cada estudio para decidir su inclusión. Se tomó el título, el resumen y las palabras clave como punto de partida, luego se revisó la introducción y conclusión. Los criterios usados fueron los siguientes:

- Examinar la relevancia del artículo en el área de pruebas de compatibilidad.
- Evaluar si el artículo provee información que sea de real utilidad para resolver las cuestiones propuestas con las PI.

3 Artículos seleccionados

Luego de realizar la búsqueda en los repositorios académicos, se encontraron 178 artículos. Al aplicar los criterios de inclusión y exclusión se seleccionaron 30 artículos para el análisis guiado por las PI propuestas. En la Tabla 1 se detallan los artículos.

Tabla 1. Artículos científicos seleccionados para esta RSL.

ID	Ref.	Título
S1	[10]	A Cross-browser Web Application Testing Tool
S2	[11]	Webdiff: Automated Identification of Cross-browser Issues in Web Applications
S3	[12]	Detecting cross-browser issues in web applications
S4	[13]	Automated cross-browser compatibility testing
S5	[14]	Cross Browser Incompatibility Reasons and Solutions
S6	[15]	CrossCheck: Combining crawling and differencing to better detect cross-browser incompatibilities in web applications
S7	[16]	WebMate: A Tool for Testing Web 2.0 Applications
S8	[17]	Visual testing of Graphical User Interfaces; An exploratory study towards systematic definitions and approaches
S9	[18]	Measuring and Improving Website User Experience using UX Methodologies; A Case Study on Cross Browser Compatibility Heuristic
S10	[19]	X-PERT Accurate identification of cross-browser issues in web applications
S11	[20]	WebMate: Generating test cases for web 2.0
S12	[21]	Browserbite: Accurate cross-browser testing via machine learning over image features
S13	[22]	X-PERT: A web application testing tool for cross-browser inconsistency detection
S14	[23]	Crawl-based analysis of web applications; Prospects and challenges
S15	[24]	Cross Browser Testing Using Automated Test Tools
S16	[25]	Modeling web application for cross-browser compatibility testing
S17	[26]	Finding HTML presentation failures using image comparison techniques
S18	[27]	Adaptive random testing for image comparison in regression web testing

Tabla 1. (Continuación)

ID	Ref.	Título
S19	[28]	A crowdsourcing framework for detecting cross-browser issues in web Application
S20	[29]	An oracle based on image comparison for regression testing of web applications
S21	[30]	Detection and Localization of HTML Presentation Failures Using Computer Vision-Based Techniques
S22	[31]	Browserbite: cross-browser testing via image processing
S23	[32]	Cross-Browser Compatibility Testing Based on Model Comparison
S24	[33]	Static Analysis Technique of Cross-Browser Compatibility Detecting
S25	[34]	A Survey on Cross Browser Inconsistencies in Web Application
S26	[35]	X-Check A Novel Cross-browser Testing Service based on Record/Replay
S27	[36]	Using Visual Symptoms for Debugging Presentation Failures in Web Applications
S28	[37]	An Automated Approach for Cross-Browser Inconsistency (XBI) Detection
S29	[38]	Detect cross-browser issues for javascript-based web applications based on record-replay
S30	[39]	Detection of Cross Browser Inconsistency by Comparing Extracted Attributes

4 Resultados

4.1 PI 1 – ¿Cuáles son las técnicas propuestas para realizar pruebas de compatibilidad web?

En total 28 artículos presentaron técnicas para realizar pruebas (93.3%), algunos de ellos proponen una combinación de éstas. A continuación se resumen las técnicas:

Análisis de modelos DOM. Document Object Model (DOM) es una interfaz multi-plataforma e independiente del lenguaje para representar documentos HTML, XHTML o XML como estructuras de árboles [40]. Cada nodo del árbol es un objeto de la aplicación. La técnica consiste en comparar los modelos DOM de una página web en distintas configuraciones. Se define una configuración como una combinación navegador web-sistema operativo [21].

Análisis visual. Se comparan imágenes que son obtenidas mediante capturas de pantalla de la aplicación. En general, se comparan de a pares. Una imagen se considera una representación correcta de la página web, la segunda captura será tomada en una configuración distinta en la que se quiere probar la compatibilidad.

Análisis de modelos de navegación. Primero se produce un modelo de comportamiento de la aplicación mediante rastreadores web (Web Crawlers), éste es expresado como máquinas de estados finitos. Estos programas realizan una exploración de los posibles estados en que pueda encontrarse la página web. Luego se analizan las diferencias entre los modelos producidos.

Grabación/reproducción. Se ejecuta en etapas, en primer lugar un usuario realiza un conjunto de acciones en la aplicación, que son grabadas para luego ser reproducidas automáticamente en una configuración diferente. Luego se compara los resultados obtenidos para encontrar diferencias que provoquen incompatibilidades.

Análisis estático. Se realiza un análisis directo del código de la aplicación en lugar

de evaluar la página mientras esta es procesada por un navegador. Se centra en detectar posibles elementos en conflicto, muy ligados al estándar HTML5. El análisis de código se maneja mediante detección de expresiones regulares.

Comparación de atributos. Comienza generando gráficos que contienen atributos propios de los elementos de una página web en diferentes configuraciones mediante rastreadores web. Luego compara los atributos de los mismos elementos en los distintos gráficos para detectar incompatibilidades.

Evaluación heurística. Es un método de inspección para realizar una evaluación de usabilidad en aplicaciones, enfocado en detectar problemas en las interfaces de usuario (UI). Se basa en evaluadores que examinan una interfaz emitiendo un juicio de acuerdo a si se cumplen ciertos requerimientos establecidos.

En la Tabla 2 se resumen las técnicas propuestas y la cantidad de artículos por cada una. Las técnicas más elegidas fueron el análisis visual, análisis de modelos DOM y análisis de modelo de navegación.

Tabla 2. Resumen de técnicas propuestas por cada artículo seleccionado.

Técnica	Artículos científicos	N° art.
Análisis de modelos DOM	(S1, S2, S6, S10, S13, S3, S4, S7, S14, S19, S26)	11
Análisis visual	(S1, S2, S6, S10, S13, S3, S8, S12, S17, S21, S18, S20, S22, S26, S27)	15
Análisis de modelo de navegación	(S4, S7, S11, S10, S13, S14, S16, S23)	8
Grabación/reproducción	(S19, S26, S29)	3
Análisis estático	(S24)	1
Comparación de atributos	(S28, S30)	2
Evaluación heurística	(S9)	1

En la Fig. 2 se observa la tendencia de uso de las técnicas en 2010-2017. El análisis de modelos DOM fue la más usada en los primeros años. Se produjo un incremento de popularidad del análisis de modelos de navegación, aunque tuvo una disminución en los últimos dos años. El análisis visual es la técnica más elegida, además su uso se incrementó a lo largo del tiempo. El análisis estático y la comparación de atributos son las técnicas más nuevas.

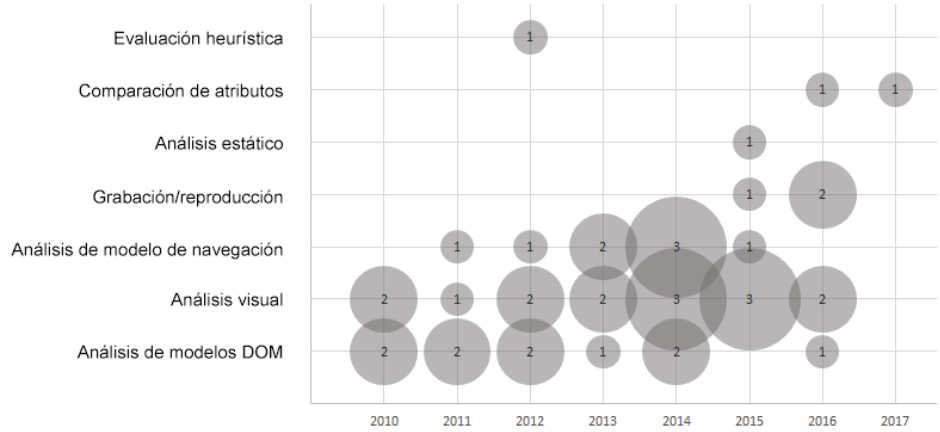

Fig. 2. Cantidad de artículos que seleccionaron cada técnica de pruebas por año de publicación.

4.2 PI 2 – ¿Qué herramientas se proponen para detectar defectos de incompatibilidad en aplicaciones web?

Se analizaron dos tipos de artículos: los que proponen una herramienta de desarrollo propio de los autores (16 artículos, 53.3%), y los que analizan herramientas comerciales (3 estudios, 10%). Un artículo (3.3%) propone una técnica de pruebas utilizando una herramienta ya desarrollada (S23).

En la Tabla 3 se listan las herramientas desarrolladas por los autores de los artículos, que además proponen alguna técnica de pruebas de compatibilidad web.

Tabla 3. Herramientas propuestas por los autores para implementar las técnicas elegidas.

Herramienta	Ref.	Técnicas de prueba
Webdiff	(S1, S2, S3)	Análisis de modelos DOM, análisis visual
CrossT	(S4)	Análisis de modelos DOM, análisis de modelo de navegación
CrossCheck	(S6)	Análisis de modelos DOM, análisis visual
WebMate	(S7, S11)	Análisis de modelos DOM, análisis de modelo de navegación
X-PERT	(S10, S14)	Análisis de modelos DOM, análisis visual, análisis de modelo de navegación
Browserbite	(S12, S22)	Análisis visual
WebSee	(S21)	Análisis visual
Crowdcheck	(S19)	Grabación/Reproducción, análisis de modelos DOM
Crawljax	(S23)	Análisis de modelo de navegación
X-Check	(S26, S29)	Grabación/reproducción, análisis de modelos DOM, análisis visual
FieryEye	(S27)	Análisis visual

En la Tabla 4 se detallan las herramientas comerciales propuestas. Algunas ya no tienen mantenimiento y otras solo funcionan en ciertas configuraciones.

Tabla 4. Herramientas comerciales para pruebas de compatibilidad web.

Herramienta	Observaciones	Ref.
Adobe Browserlab	Obsoleto desde el 13 de Marzo, 2013.	(S5)
IE Netrenderer	Sólo para Internet Explorer. Gratuita.	
Browsera	Realiza pruebas de sitios completos.	
Litmusapp	Obsoleto desde el 28 de Febrero, 2017.	
Browsrcamp	Obsoleto.	
IETester	Sólo para Internet Explorer.	(S15)
SuperPreview	De Expression Web. Ya no desarrollada.	
BrowserStack		
BrowserShots	Herramienta gratuita.	
CrossBrowserTesting		
Browser Sandbox	Herramienta gratuita.	(S25)
IE Tab	Extensión para Firefox y Chrome.	
BrowserCam	Obsoleto.	
Browserseal	Obsoleto.	

4.3 PI 3 – ¿Qué desafíos se presentan al implementar las pruebas?

En la Fig. 3 se aprecia que la detección de elementos variables fue uno de los desafíos

más mencionados. Se refiere a regiones en las aplicaciones web que no permanecen estáticas, incluyen animaciones, estadísticas y principalmente publicidades que pueden cambiar con sucesivas cargas de la página. En (S1, S2, S20) se maneja esta problemática con una estrategia de detección de regiones variables. Se carga la misma página múltiples veces en una misma configuración y se verifica qué sectores presentan cambios, para descartarlos de las pruebas.

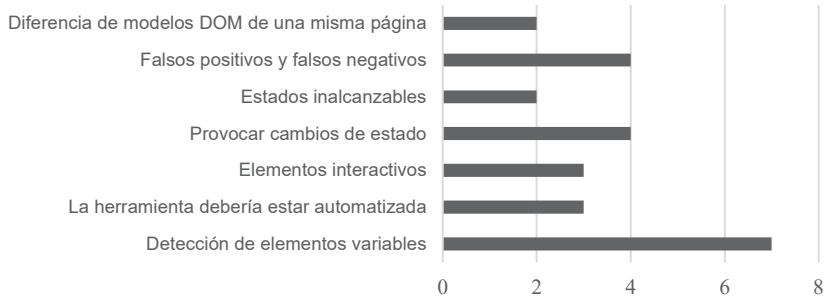

Fig. 3. Desafíos más enfrentados según los artículos científicos seleccionados.

Falsos positivos se refieren a pruebas que resultan positivas cuando no deberían, ya que existe compatibilidad web. Falsos negativos son pruebas que resultan negativas aun con la existencia de incompatibilidades web. Este problema se da en todas las pruebas al utilizar cualquiera de las técnicas mencionadas. Según (S6), información obtenida de modelos DOM puede conducir a falsos positivos, por lo que se emplea ésta técnica en conjunto con el análisis visual. En (S12, S22) se detalla un módulo de clasificación de potenciales incompatibilidades en capturas de pantalla. Se clasifican en verdaderos positivos y falsos positivos. Para utilizar este criterio de clasificación se emplea una técnica de aprendizaje automático con redes neuronales. El aprendizaje automático es una rama de la inteligencia artificial que permite a las computadoras aprender comportamiento a partir de datos empíricos [15]. También se usan técnicas de aprendizaje automático en (S6), para construir un detector de diferencias visuales más preciso. En (S29) se señala que la técnica de análisis de modelo de navegación es propensa a generar falsos positivos y negativos. Al rastrear la aplicación con los web crawlers, se ignoran fuentes de acciones no determinísticas, es decir, sólo se consideran actividades explícitas cargadas por los usuarios.

Provocar cambios de estado está relacionado al análisis de modelo de navegación. Los artículos (S7, S14) mencionan la dificultad que resulta de la necesidad de pasar de estados al elaborar el modelo de navegación. Particularmente cuando los eventos son muy numerosos, donde cualquier clic puede desencadenar un estado nuevo. En (S16) se explica que acciones ejecutadas en distinto orden producen diferentes resultados.

Elementos interactivos son una problemática recurrente en los estudios elegidos (S7, S16). Tecnologías web 2.0 promueven comportamiento dinámico con código que se ejecuta del lado del cliente, como JavaScript y HTML5 [41].

La necesidad de que la herramienta para las pruebas de compatibilidad web sea automatizada se hace evidente en varios estudios (S6, S9, S29). Se menciona lo costoso que resulta tener un usuario que realice acciones de forma manual (S6). Aun así, (S8, S9) proponen técnicas totalmente manuales para las pruebas.

Estados inalcanzables es una problemática de los análisis de modelo de navegación. Según (S14, S29) existen estados que no pueden alcanzarse desde enlaces en la página. Entonces, el modelo producido estará incompleto, dejando parte del sistema fuera de las pruebas.

5 Conclusiones y trabajos futuros

Actualmente la plataforma elegida para la comunicación y explotación de las aplicaciones es la web. Debido a la naturaleza de las aplicaciones, asíncronas y dinámicas, realizar pruebas efectivas se vuelve un desafío para los desarrolladores. La rápida evolución y popularidad de tecnologías han fomentado la producción de aplicaciones complejas y dinámicas [41], [43]. Esto hace que la usabilidad, la compatibilidad y la disponibilidad sean los criterios clave para el éxito de un sitio web [1]. Pero distintos navegadores producen contenido diferente [44], [45], [46]. De aquí surge la necesidad de realizar pruebas de compatibilidad.

Se estudiaron las técnicas con las cuales se diseñan pruebas de compatibilidad. El análisis visual resultó la técnica más elegida, quedando el análisis de modelos DOM y el análisis de modelos de navegación en segundo y tercer lugar, respectivamente. Asimismo, se revisaron las aplicaciones propuestas para desarrollar las pruebas. Se las clasificó según el origen: desarrolladas por los autores de los estudios (11, 44%) o aplicativos comerciales que el artículo evaluó (14, 56%).

Finalmente se listaron sistemáticamente los desafíos encontrados en los artículos revisados. Una de las dificultades más mencionadas fue la presencia de elementos variables, abundantes en las aplicaciones web actuales [43]. Otra de las problemáticas encontradas fue la necesidad de automatizar las pruebas. Si bien se proponen herramientas y técnicas con distintos niveles de automatización (algunas soluciones son totalmente manuales), se observa el alto costo asociado al requerir la participación constante de un usuario. Dependiendo de la popularidad de un sitio web, el número de configuraciones que se deben probar para cubrir el 90-95% de usuarios asciende a 20-30 [31]. Por esto es necesario automatizar el proceso de pruebas.

Como trabajo futuro se propone diseñar y desarrollar una herramienta para pruebas de compatibilidad web, intentando resolver los desafíos encontrados en la RSL.

Referencias

1. Dustin, E., Rashka, J., McDiarmid, D.: Quality Web Systems: Performance, Security, and Usability. Addison Wesley, Boston, Massachusetts, USA (2001)
2. Moustakis, V., Litos, C., Dalivigas, A., Tsironis, L.: Website Quality Assessment Criteria. IQ, 59-73 (2004)
3. Kurose, J., Ross, K.: Redes de computadoras. Un enfoque descendente. Pearson (2010)
4. Mozilla Corporation In: Mozilla Developer Network. (Accessed 2017) Available at: https://developer.mozilla.org/en-US/docs/Learn/Tools_and_testing/Cross_browser_testing
5. Kitchenham, B., Charters, S.: Guidelines for performing Systematic Literature Reviews in Software Engineering., Keele University, Durham University (2007)
6. Zakaria, Z., Atan, R., Ghani , A., Sani, N.: Unit Testing Approaches for BPEL: A Systematic

Review. In : Software Engineering Conference. Asia-Pacific, Malaysia (2009)
7. Nabil, E. I.: Specifications for Web Services Testing: A Systematic Review. In : 2015 IEEE World Congress on Services, New York, New York, USA (2015)
8. Mendes, E.: A systematic review of Web engineering research. In : International Symposium on Empirical Software Engineering, Noosa Heads, Queensland, Australia (2005)
9. Dogan, S., Betin-Can, A., Garousi, V.: Web application testing: A systematic literature review. Journal of Systems and Software 91, 174-201 (2014)
10. Choudhary, S., Varsee, H., Orso, A.: A cross-browser web application testing tool. In : 2010 IEEE International Conference on Software Maintenance, Timișoara, Romania (2010)
11. Choudhary, S., Versee, H., Orso, A.: WEBDIFF: Automated Identification of Cross-browser Issues in Web Applications. In : 26th IEEE International Conference on Software Maintenance, Timișoara, Romania (2010)
12. Choudhary, S.: Detecting Cross-browser Issues in Web Applications. In : 2011 33rd International Conference on Software Engineering (ICSE), Honolulu, Hawaii, USA (2011)
13. Mesbah, A., Prasad, M.: Automated Cross-Browser Compatibility Testing. In : 2011 33rd International Conference on Software Engineering, Honolulu, Hawaii, USA (2011)
14. Ochin, J.: Cross Browser Incompatibility: Reasons and Solutions. International Journal of Software Engineering & Applications, July 2011 Vol.2(No.3) (2011)
15. Choudhary, S., Prasad, M., Orso, A.: CROSSCHECK: Combining Crawling and Differencing To Better Detect Cross-browser Incompatibilities in Web Applications. In : IEEE Fifth International Conference on Software Testing, Verification and Validation, Canada (2012)
16. Dallmeier, V., Burger, M., Orth, T., Zeller, A.: WebMate: A Tool for Testing Web 2.0 Applications. In : Workshop on JavaScript Tools, Beijing, China (2012)
17. Issa, A., Sillito, J., Garousi, V.: Visual Testing of Graphical User Interfaces: an Exploratory Study Towards Systematic Definitions and Approaches. In : 2012 14th IEEE International Symposium on Web Systems Evolution, Trento, Italy (2012)
18. Sivaji, A., Ramli, N., Nor, Z., Chuan, N.-K., Wan, F., Wan, A., Shi-Tzuaan, S.: Measuring and Improving Website User Experience using UX Methodologies: A Case Study on Cross Browser Compatibility Heuristic. In : Southeast Asian Network of Ergonomics Societies, Langkawi, Kedah, Malasia (2012)
19. Choudhary , S., Prasad, M., Orso, A.: X-PERT: Accurate identification of cross-browser issues in web applications. In : 2013 35th International Conference on Software Engineering, San Francisco, California, USA (2013)
20. Dallmeier, V., Burger, M., Orth, T., Zeller, A.: WebMate: Generating Test Cases for Web 2.0. In : International Conference on Software Quality, Software Quality. Increasing Value in Software and Systems Development, Vienna, Austria (2013)
21. Semenenko, N., Dumas, M., Saar, T.: Browserbite: Accurate Cross-Browser Testing via Machine Learning over Image Features. In : 2013 29th IEEE International Conference on Software Maintenance, Eindhoven, Netherlands, Netherlands (2013)
22. Choudhary, S., Prasad, M., Orso, A.: X-PERT: a web application testing tool for cross-browser inconsistency detection. In : 2014 International Symposium on Software Testing and Analysis , San Jose, California, USA (2014)
23. Deursen, A. v., Mesbah, A., Nederlof, A.: Crawl-based analysis of web applications: Prospects and challenges. Science of Computer Programming 87, 173-180 (2014)
24. Kaalra, B., Gowthaman, K.: Cross Browser Testing Using Automated Test Tools. International Journal of advanced studies in Computer Science and Engineering 3(10), 7-12 (2014)
25. Li, X., Zeng, H.: Modeling web application for cross-browser compatibility testing. In : 2014 15th IEEE/ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing, Las Vegas, Nevada, USA (2014)
26. Mahajan, S., Halfond, W.: Finding HTML presentation failures using image comparison techniques. In : 29th ACM/IEEE international conference on Automated software

engineering, Vasteras, Sweden (2014)
27. Selay, E., Zhou, Z., Zou, J.: Adaptive Random Testing for Image Comparison in Regression Web Testing. In : 2014 International Conference on Digital lmage Computing: Techniques and Applications, Wollongong, New South Wales, Australia (2014)
28. He, M., Tang, H., Wu, G., Zhong, H.: A Crowdsourcing framework for Detecting Cross-Browser Issues in Web Application. In : the 7th Asia-Pacific Symposium on Internetware, Wuhan, China (2015)
29. Hori, A., Takada, S., Tanno, H., Oinuma, M.: An oracle based on image comparison for regression testing of web applications. In : 27th International Conference on Software Engineering and Knowledge Engineering, Pittsburgh, USA (2015)
30. Mahajan, S., Halfond, W.: Detection and Localization of HTML Presentation Failures Using Computer Vision-Based Techniques. In : 2015 IEEE 8th International Conference on Software Testing, Verification and Validation, Graz, Austria (2015)
31. Saar, T., Dumas, M., Kaljuve, M., Semenenko, N.: Browserbite: cross-browser testing via image processing. Software—Practice & Experience 46(11), 1459-1477 (2015)
32. Shi, H., Zeng, H.: Cross-Browser Compatibility Testing Based on Model Comparison. In : 2015 International Conference on Computer Application Technologies, Matsue, Japan (2015)
33. Xu, S., Zeng, H.: Static Analysis Technique of Cross-Browser Compatibility Detecting. In : 2015 3rd International Conference on Applied Computing and Information Technology/2nd International Conference on Computational Science and Intelligence, Okayama, Japan (2015)
34. Barskar, N., Patidar, C. P.: A Survey on Cross Browser Inconsistencies in Web Application. International Journal of Computer Applications 137(4), 37-41 (2016)
35. He, M., Wu, G., Tang, H., Chen, W., Wei, J., Zhong, H., Huang, T.: X-Check: A Novel Cross-browser Testing Service based on Record/Replay. In : 2016 IEEE International Conference on Web Services, San Francisco, California, USA (2016)
36. Mahajan, S., Li, B., Behnamghader, P., Halfond, W.: Using Visual Symptoms for Debugging Presentation Failures in Web Applications. In : 2016 IEEE International Conference on Software Testing, Verification and Validation, Chicago, Illinois, USA (2016)
37. Sharma, M., Patidar, C.: An Automated Approach for Cross-Browser Inconsistency (XBI) Detection. In : 9th Annual ACM India Conference, Gandhinagar, India (2016)
38. Wu, G., He, M., Tang, H., Wei, J.: Detect Cross-Browser Issues for JavaScript-Based Web Applications Based on Record/Replay. In : 2016 IEEE International Conference on Software Maintenance and Evolution, Raleigh, North Carolina, USA (2016)
39. Patidar, C. P., Sharma, M., Sharda, V.: Detection of Cross Browser Inconsistency by Comparing Extracted Attributes. International Journal of Scientific Research in Computer Science and Engineering 5(1), 1-6 (2017)
40. W3C Document Object Model. (Accessed 2005) Available at: https://www.w3.org/DOM/
41. O'Reilly, T.: What Is Web 2.0 - Design Patterns and Business Models for the Next Generation of Software. Available at: http://www.oreilly.com/pub/a/web2/archive/what-is-web-20.html
42. Powers, D.: Evaluation: From precision, recall and F-measure to ROC, informedness, markedness & correlation. Journal of Machine Learning Technologies 2(1), 37-63 (2011)
43. Jazayeri, M.: Some Trends in Web Application Development. In : Future of Software Engineering, 2007. FOSE '07, Minneapolis, Minesota, USA (2007)
44. Kiciman, E., Livshits, B.: AjaxScope: a platform for remotely monitoring the client-side behavior of web 2.0 applications. In : Proceedings of twenty-first ACM SIGOPS symposium on Operating systems principles, Stevenson, Washington, USA (2007)
45. Ricca, F., Tonella, P.: Web testing: a roadmap for the empirical research. In : Seventh IEEE International Symposium on Web Site Evolution, Budapest, Hungary (2005)
46. Ramler, R., Weippl, E., Winterer, M., Schwinger, W., Altmann, J.: A Quality-Driven Approach to Web Testing. In : Ibero American Conference on Web Engineering, Santa Fé, Argentina (2002)