Ciencia Latina
Revista Multidisciplinar

DOI: https://doi.org/10.37811/cl_rcm.v6i4.2878

Automatización de pruebas de accesibilidad en la web

José Boris Bellido Santa María
bellido.boris@usfx.bo
https://orcid.org/0000-0002-2380-7785
Universidad Mayor, Real y Pontificia de San Francisco,
Xavier de Chuquisaca,
Sucre – Bolivia

RESUMEN

El uso de las computadoras es cada vez más frecuente por personas de todas las ocupaciones y oficios, para garantizar que los sitios y aplicaciones web beneficien a más personas es necesario tener en cuenta la accesibilidad.

El objetivo es mejorar la aplicación de las pruebas de accesibilidad en aplicaciones web, a partir de la automatización.

Se emplearon diferentes métodos de investigación: Método del Análisis documental, para elaborar el sustento teórico. Método de análisis y síntesis, se han analizado los estándares y herramientas de accesibilidad; y los marcos de trabajo para automatización de pruebas; se han sintetizado relaciones para la realización de pruebas de accesibilidad automatizadas. Método Experimental: Se ha desarrollado una biblioteca para Robot Framework que permite usar la prueba de accesibilidad automatizada.

Con Robot Framework y axe-core es posible realizar pruebas de accesibilidad automatizadas a cualquier sitio web a partir de su dirección web, lo cual mejora la aplicación de las pruebas de accesibilidad en vista que es posible analizar un sitio web completo en poco tiempo.

Palabras clave: *automatización; accesibilidad; python; robot; axe-core.*

Correspondencia bellido.boris@usfx.bo
Artículo recibido: 05 agosto 2022. Aceptado para publicación: 15 agosto 2022.
Conflictos de Interés: Ninguna que declarar

Como citar: Bellido Santa María, J. B. (2022) Automatización de pruebas de accesibilidad en la web. Ciencia Latina Revista Científica Multidisciplinar, 6(4) 3671-3688. DOI: https://doi.org/10.37811/cl_rcm.v6i4.2878

Automation of accessibility tests on the web

ABSTRACT

The use of computers is becoming more frequent by people of all occupations and trades, to ensure that web sites and applications benefit more people, accessibility must be taken into account.

The objective is to improve the application of accessibility tests in web applications, based on automation.

Different research methods were used: Documentary Analysis Method, to elaborate the theoretical support. Analysis and synthesis method, accessibility standards and tools have been analyzed; and test automation frameworks; relationships have been synthesized for performing automated accessibility tests. Experimental Method: A library has been developed for the Robot Framework that allows automated accessibility testing to be used.

With Robot Framework and axe-core it is possible to carry out automated accessibility tests on any website from its web address, which improves the application of accessibility tests since it is possible to analyze an entire website in a short time.

Keywords: automation; accessibility; python; robot; axe-core.

INTRODUCCIÓN

El uso de las computadoras y de los celulares inteligentes es cada vez más frecuente por personas de todas las ocupaciones y oficios. Han pasado de ser un lujo a una necesidad, en vista que muchas actividades de la vida cotidiana han sido afectadas por Internet, como ser el trabajo, la educación, el ocio y la diversión; situación que se ha enfatizado con la pandemia de la COVID-19.

Se requiere desarrollar software con alta calidad en tiempos cortos que satisfaga las necesidades, cada vez más exigentes, de los usuarios. El proceso de pruebas es indispensable para garantizar la calidad del producto entregado; la automatización permite a los probadores realizar pruebas repetitivas y tediosas de manera efectiva.

Para garantizar que el software llegue y beneficie a más personas es necesario tener en cuenta la accesibilidad, por lo tanto es necesario incluir pruebas que verifiquen el comportamiento del software cuando es utilizado por personas con discapacidades permanentes o temporales.

El sistema universitario en Bolivia en totalidad se ha adaptado a las restricciones de movilidad y aglomeración de personas aplicando soluciones basadas en Internet. Los sitios web de las universidades han adquirido un papel muy importante a causa de la pandemia de la COVID-19, con las instalaciones cerradas, la mejor forma de encontrar información ha sido consultar los sitios web de las universidades.

Cada vez más instituciones públicas y privadas, en especial las universidades, ven la importancia y necesidad de contar con software amigable con personas que poseen alguna discapacidad. Si bien en el proceso de prueba de accesibilidad se cuentan con muchas herramientas que ayudan a realizar la tarea sigue siguiendo un proceso que requiere de la intervención del probador, muchas de las herramientas solo pueden probar una página y no todo el sitio o aplicación web.

Las pruebas automatizadas de accesibilidad permitirían a los probadores realizar su trabajo de una forma más rápida.

Según Foord (2017), buenas prácticas para el desarrollo y prueba de software son:

- [5] Fallar rápido, verificar lo más pronto posible el comportamiento ante datos y estados incorrectos.
- [6] Las pruebas unitarias son pruebas del comportamiento no la implementación.

- [7] Las pruebas unitarias deberían cubrir todos los caminos posibles.
- [10] Escribir código defensivo, es decir tener en cuenta casos donde las cosas puedan ir mal.
- [21] La primera vez escribir el código que funciona, luego hacer que funcione más rápido.
- [22] El alcance de las pruebas unitarias debe ser pequeño, dando la mayor información posible cuando la prueba falle.
- [26] Ser ingenieros, pensar en el diseño y construir software robusto y bien implementado.

Muchos de los proyectos de desarrollo de software no contemplan la accesibilidad, entonces el producto cumple con el comportamiento y la funcionalidad pero no es accesible. Al momento de construir los casos de prueba se debería tener en cuenta caminos que contemplen la accesibilidad; las pruebas deberían informar de los problemas relacionados con la accesibilidad cuando se presenten y etiquetarlos como fallas. No contemplar la accesibilidad dentro de un proyecto de software implica que muchas personas no se podrán beneficiar del resultado.

Bolivia, sin duda, es uno de los países con un marco legal abundante. La Ley N° 223. Ley General Para Personas con Discapacidad garantiza “a las personas con discapacidad, el ejercicio pleno de sus derechos y deberes en igualdad de condiciones y equiparación de oportunidades” (Bolivia, 2012). La tecnología debe ser un pilar importante para conseguir que todas las personas puedan ejercer sus derechos por sí mismas. Construyendo una sociedad más equitativa y justa para todas las personas que habitan el país.

Garantizar la accesibilidad del contenido y de los sistemas ayudará a que las personas con alguna discapacidad puedan ejercer de mejor manera sus derechos, la Ley 223 define el derecho a la accesibilidad.

> Artículo 17. (DERECHO A LA ACCESIBILIDAD). El Estado Plurinacional de Bolivia garantiza el derecho de las personas con discapacidad a gozar de condiciones de accesibilidad que les permitan utilizar la infraestructura y los servicios de las instituciones públicas, privadas, espacios públicos, medios y sistemas de comunicación, tecnología y transporte, para su utilización y disfrute de manera autónoma con independencia de su condición de discapacidad y a exigir a las

> instituciones del Estado la adopción de medidas de acción positiva para el ejercicio de éste derecho (Bolivia, 2012).

Para que las personas puedan ejercer su derecho a la accesibilidad es necesario que todas las instituciones adecuen sus sistemas actuales para ser accesibles e introduzcan como parte de los requerimientos de funcionalidades y sistemas nuevos a la accesibilidad. La ley 223, el artículo 37, dispone que todas las instituciones públicas y privadas ven tener sistemas accesibles.

> *Artículo 37. (ÁMBITO DE LA ACCESIBILIDAD A INFRAESTRUCTURAS Y OTROS DERECHOS).*
>
> I. El Estado Plurinacional de Bolivia definirá políticas públicas en materia de accesibilidad que garanticen el ejercicio pleno de este derecho.
>
> II. Todos los Órganos del Estado Plurinacional, en sus distintos niveles, instituciones públicas y privadas, deberán adecuar su estructura arquitectónica, sistemas, medios de comunicación y medios de transporte, de manera gradual, a partir de la promulgación de la presente Ley, para garantizar la accesibilidad a las Personas con Discapacidad.
>
> III. Las nuevas construcciones, sistemas, medios de comunicación y medios de transporte deberán contar con las condiciones de accesibilidad establecidas por la presente Ley a partir de su promulgación
>
> (Bolivia, 2012).

En vista que hoy en día la mayoría de los sistemas informáticos se desarrollan utilizando tecnologías web, sin importar el dispositivo que lo renderiza, es necesario tener en cuenta la accesibilidad en la web.

La web sin duda es el servicio más usado de Internet por las personas, para muchos la web es sinónimo de Internet, pues permite tener acceso a otros servicios como el correo electrónico, chat y otros. Para que las personas con alguna discapacidad tengan un mejor acceso a la web es necesario que el contenido de la web y los navegadores sigan un conjunto de estándares de accesibilidad.

Según Henry (2021), en los componentes esenciales de la accesibilidad web, se puede ver que la accesibilidad depende de varios componentes de desarrollo web que interactúan juntos y cómo se aplican las pautas de WAI (WCAG, ATAG, UAAG).

Pautas de accesibilidad para el contenido web (WCAG): Por contenido web se refiere a la información que se encuentra en la web, que incluye: textos, imágenes y sonidos; además del código que define la estructura y la presentación; Las WCAG se aplican al contenido dinámico, multimedia y móvil (Henry, 2021). El contenido web es el producto final que consumen los usuarios, un buen contenido web debe ser accesible por cualquier usuario sin importar si tiene o no una discapacidad. Las WCAG guían en la evaluación de la calidad del contenido desde el punto de vista de la accesibilidad.

Pautas de accesibilidad para las herramientas de creación de contenido (ATAG): Las herramientas de creación de contenido son programas que los autores utilizan para producir contenido web. Las ATAG explican cómo: hacer que dichas herramientas sean accesibles en sí mismas, de forma que las personas con discapacidad puedan crear contenido web (Henry, 2021). Los usuarios con alguna discapacidad tienen los mismos derechos que los otros usuarios, por lo tanto no solo consumen contenido, deben ser capaces de producirlo. Las herramientas de creación de contenido deben ser capaces de adaptarse a las limitaciones de los usuarios para que los mismos puedan crear, actualizar y eliminar contenido por sí mismos.

Pautas de accesibilidad para el agente de usuario (UAAG): Los agentes de usuario son navegadores, extensiones de los navegadores, reproductores multimedia, lectores y otras aplicaciones que presentan contenido web (Henry, 2021). El contenido web se procesa y renderiza en un navegador para ser presentado al usuario final. Es importante que el contenido y el navegador cumplan las mismas reglas y que se actualicen de manera conjunta. Las UAAG guían a los desarrolladores de navegadores para que presenten el contenido web accesible de manera correcta.

Las pautas de accesibilidad para el contenido conocidas como WCAG, por el término en inglés, Web Content Accessibility Guidelines. La versión actual tiene más de una década desde su publicación, “WCAG 2.0 se publicó el 11 de diciembre del 2008. WCAG 2.1 se publicó el 5 de junio del 2018. Todos los requisitos (criterios de conformidad) de la versión 2.0 están incluidos en 2.1” (Henry, 2021); la versión 2.1 no modifica ningún criterio de la

versión 2.0, por lo tanto al cumplir con la versión 2.1 se está cumpliendo con la versión 2.0 al mismo tiempo.

Las WCAG 2.1 es un estándar técnico estable y de referencia; contienen 13 pautas agrupadas en cuatro principios: perceptible, operable, comprensible y robusto. Cada pauta incluye criterios de conformidad, comprobables y clasificados en tres niveles: A, AA y AAA (Henry, 2021). El cumplimiento de todas las pautas avala la accesibilidad del contenido web; el contenido puede encontrarse en algún nivel de conformidad. World *Wide Web Consortium (2020) sostiene que los niveles de conformidad son:*

- El Nivel A es el nivel mínimo.
- El Nivel AA incluye todos los requisitos de los niveles A y AA. Muchas organizaciones se esfuerzan por alcanzar este nivel.
- El Nivel AAA incluye todos los requisitos de los niveles A, AA y AAA.

Para que una página se encuentre en el nivel A se deben cumplir con todos sus criterios de conformidad de dicho nivel. El nivel AA incluye al nivel A, por lo tanto para que una página se encuentre en nivel AA es necesario que se cumplan con los criterios de los niveles A y AA. Finalmente una página se encuentra en nivel AAA cuando cumple con todos sus criterios de las WCAG.

Henry & Dick (2018) presenta los cuatro principios con sus respectivas pautas:

1. *Perceptible:*
    a. Proporcione alternativas textuales para contenido no textual.
    b. Proporcione subtítulos y otras alternativas para multimedia.
    c. Cree contenido que se pueda presentar de diferentes formas, incluyendo a las tecnologías de apoyo, sin perder información.
    d. Facilite que los usuarios puedan ver y oír el contenido.
2. *Operable:*
    a. Proporcione acceso a todas las funcionalidades mediante el teclado.
    b. Conceda a los usuarios tiempo suficiente para leer y usar el contenido.
    c. No use contenido que pudiera causar convulsiones o reacciones físicas.
    d. Ayude a los usuarios a navegar y encontrar el contenido. Facilite métodos de entrada diferentes al teclado.

3. *Comprensible:*
    a. Proporcione texto legible y comprensible.
    b. Proporcione contenido que sea predecible en apariencia y operación.
    c. Ayude a los usuarios a evitar y corregir errores.
4. *Robusto:*
    a. Maximice la compatibilidad con herramientas de usuario actuales y futuras.

Deque Systems Inc. (2021) señala que, axe-core es un motor de pruebas de accesibilidad para sitios web; es rápido, seguro, liviano y se creó para integrarse con cualquier entorno de prueba, de modo que pueda automatizar las pruebas de accesibilidad junto con las pruebas funcionales habituales. Es un marco de trabajo para el lenguaje de programación JavaScript por lo tanto se puede incluir en cualquier proyecto web, para usarlo es necesario cargar y ejecutar el módulo dentro de la página web que se desea probar.

Deque Systems Inc. (2021) sostiene que, axe-core tiene diferentes tipos de reglas, para WCAG 2.0 y 2.1 en el nivel A y AA, así como una serie de mejores prácticas que lo ayudan a identificar prácticas de accesibilidad comunes. Las reglas de axe-core pueden aplicar varios criterios de conformidad de WCAG al mismo tiempo.

Según Deque Systems Inc. (2021), axe-core puede encontrar una media del 57% de los problemas de WCAG automáticamente y devolverá elementos como incompletos donde se necesita una revisión manual. En la última versión de axe-core se han incorporado reglas para los tres niveles, como se puede ver en la tabla 1.

Con axe-core en el proceso de verificación de la accesibilidad del contenido web se realiza de una manera más rápida, en vista que tiene una cobertura del 53.33% de los criterios de conformidad del nivel A de las WCAG. Además, el 35% de los criterios de nivel AA cuentan con reglas para su verificación automática; por lo tanto el nivel AA tiene una cobertura del 46%. Finalmente el nivel AAA solo cuenta con un 10.71% de criterios que cuenta con reglas en axe-core, que los verifican; lo que implica que la verificación el nivel AAA está automatizada en un 33.33%.

axe-selenium-python integra axe-core y selenium para permitir pruebas de accesibilidad web automatizadas (Sereduck, 2018). Permite ejecutar las reglas de axe-core sobre un controlador de selenium desde Python, los resultados de la evaluación de accesibilidad se obtienen en formato json. Lamentablemente la última versión es del 12 de noviembre

de 2018 que incluye la versión 3.1.1 de axe-core, mientras la última versión de axe-core es la 4.4.3 del 13 de julio de 2022 (Deque Systems Inc., 2022).

Python es un lenguaje de programación con una gran comunidad de desarrolladores que constantemente genera herramientas que enriquecen al lenguaje. En el caso de las pruebas automatizadas existen muchas opciones. Según Software Testing Help (2021), los principales marcos de trabajo para pruebas de Python son: Robot, PyTest, Unittest, DocTest, Nose2 y Testify, en ese orden; las principales características de los mismo se pueden observar en la tabla 2.

Se puede observar que los principales marcos de trabajo cuentan con licencias libres y son de acceso gratuito, por lo tanto se pueden utilizar en cualquier proyecto sin importar su naturaleza y presupuesto.

Software Testing Help (2021) concluye, al considerar las características, que los mejores marcos de prueba para pruebas funcionales son: Robot, PyTest, Unittest. Robot es el mejor marco para aquellos que son nuevos en las pruebas con Python y desean tener un comienzo sólido.

Robot Framework es un marco de trabajo basado en Python, es extensible guiado por palabras de automatización para pruebas de aceptación, desarrollo dirigido por pruebas de aceptación (ATDD), desarrollo dirigido por comportamiento (BDD) y automatización robótica de procesos (RPA) (Robot Framework Foundation, 2021). Robot Framework no es solo un conjunto de clases, funciones y variables que puedan usar los desarrolladores y probadores para construir y automatizar pruebas; proporciona un lenguaje, extensible, que permite a los usuarios y miembros del equipo con pocas habilidades de programación construir pruebas de manera fácil.

Robot Framework tiene una arquitectura muy modular como se puede ver en la figura 1. Que permite que se puedan construir bibliotecas de forma rápida y de esta manera extender el lenguaje con nuevas palabras y frases.

La arquitectura permite interactuar de maneras diferentes con Robot Framework, cada capa requiere un nivel mayor de conocimientos y habilidades de programación; siendo la capa de Test Data, la destinada a las personas con menos condiciones para el desarrollo de software.

> Los datos de prueba están en formato tabular simple y fácil de editar. Cuando se inicia, procesa los datos, ejecuta casos de

> prueba y genera registros e informes. El marco central no sabe nada sobre el objetivo de la prueba y la interacción con él es manejada por bibliotecas. Las bibliotecas pueden usar interfaces de aplicaciones directamente o usar herramientas de prueba de bajo nivel como controladores (Robot Framework Foundation, 2021).

Robot Framework permite escribir los casos de prueba usando diferentes formatos, los más relevantes son:

- Formato separado por espacios.
- Formato separado por tuberías.
- reStructuredText.

El enfoque más común para crear pruebas es usar el formato separado por espacios; donde las palabras clave y sus argumentos se separan entre sí con dos o más espacios (Robot Framework Foundation, 2021). Es fácil de entender en vista que parecen instrucciones en idioma inglés; en primer lugar se debe poner la palabra clave, que puede se una palabra o frase, una frase está compuesta de varias palabras separadas por un espacio, como ser: Convert To Integer; a continuación van los argumentos necesarios para que la palabra clave funcione adecuadamente, todo separados por al menos dos espacios. Es necesario tener mucho cuidado con los espacios, si se agregan espacios innecesarios entre los elementos de una frase Robot Framework asumirá de que se trata de una palabra con sus argumentos; por otro lado si solo se usa un espacio entre los argumentos se asumirá de que se trata de una frase; al principio puede ser incómodo pero a medida que se aprenden más palabras clave se hace más fácil escribir los casos de prueba.

Una alternativa es usar el formato separado por tuberías donde el separador es el carácter de tubería rodeado por espacios (|) (Robot Framework Foundation, 2021). Separar las palabras clave de los argumentos usando tuberías (|) ayuda a delimitar claramente las palabras clave de sus argumentos, este formato puede ser un poco incómodo para escribir debido a que la tecla de tubería se encuentra en una posición dificultosa en comparación del espacio.

También admite archivos reStructuredText para que los datos normales de Robot Framework se incrusten en bloques de código de Python (Robot Framework Foundation,

2021). Permite incluir bloque con información adicional como ser código fuente, que no se ejecutarán en la prueba. reStructuredText es un sistema de análisis de marcado de texto plano, fácil de leer del tipo WYSIWYG; es útil para la documentación en línea de programas, para crear rápidamente páginas web simples y para documentos independientes (Docutils, 2021). Permite construir la documentación del software al mismo tiempo que las pruebas, posibilita renderizar como HTML para tener una documentación muy útil que cualquier interesado pueda revisar fácilmente; al mismo tiempo se tiene como ejemplos los casos de prueba totalmente funcionales que permiten validar la calidad del software. Incluir las pruebas y la documentación en el mismo documento puede ser confuso para un miembro nuevo en el equipo.

La capacidad de Robot Framework de incluir diferentes tipos de formatos para escribir los casos de prueba permite al equipo usar el que mejor se adapte a sus necesidades y capacidades. Si bien Robot Framework permite usar todos los formatos en el mismo proyecto es recomendable elegir solo uno para todo el proyecto.

La calidad de las pruebas que pueden escribir los miembros del equipo con pocas capacidades de programación depende de las palabras clave que tengan a su disposición. Robot Framework incluye muchas palabras clave organizadas en diferentes bibliotecas, como se puede ver en la tabla 3.

Además de las más de 300 palabras clave, Robot Framework incluye estructuras selectivas y repetitivas que permiten construir casos de prueba complejos.

Existen bibliotecas que permiten extender las capacidades de Robot Framework, algunas de ellas se pueden ver en la tabla 4, que permiten conectar usar Selenium y axe-core.

El objetivo es mejorar la aplicación de las pruebas de accesibilidad en aplicaciones web, a partir de la automatización.

METODOLOGÍA

Método del Análisis documental: La consulta bibliográfica ha permitido identificar los estándares y herramientas de accesibilidad web, así mismo ha posibilitado determinar los marcos de trabajo más usado en Python para la automatización de pruebas.

Método de análisis y síntesis: Se han analizado los estándares y herramientas de accesibilidad disponibles; además de los marcos de trabajo para automatización de pruebas usando Python. Se han sintetizado y generado relaciones para permitir la realización de pruebas de accesibilidad automatizadas.

Método Experimental: Se ha desarrollado una biblioteca para Robot Framework que permite usar axe-core para verificar la accesibilidad de un sitio o aplicación web de manera automática.

RESULTADOS Y DISCUSIÓN

La biblioteca axelibrary, de Robot Framework, depende de la biblioteca para Python axe-selenium-python, la cual permite especificar el archivo del motor de pruebas de accesibilidad axe-core, lo cual ayuda a usar versiones diferentes de axe-core; además del contexto y opciones para la ejecución de la prueba. Lamentablemente axelibrary no permite suministrar dichos parámetros.

Se modificó la palabra clave Run Accessibility Tests, de axelibrary, para que sea posible especificar la ruta del archivo de axe-core y los parámetros de ejecución. Se agregaron los parámetros al método run_accessibility_tests, encargada de implementar la palabra clave Run Accessibility Tests.

Los parámetros agregados al método fueron:

- axe_script_url: Ruta del archivo de axe-core.
- context: Define el alcance del análisis, es decir la parte del DOM que se va analizar. Por lo general, será el documento o un selector específico.
- options: Conjunto de opciones que cambian el funcionamiento de axe-core, incluidas las reglas que se ejecutarán.

Al momento de crear la instancia de la clase Axe es posible especificar la ruta del archivo de axe-core, se realizó la modificación para aprovechar dicha característica, enviando como argumento axe_script_url.

Finalmente, el método run, de la clase Axe, permite definir el contexto y las opciones de la ejecución, las variables context y options se envían como argumento de dicho método.

Para analizar un sitio web es necesario encontrar los enlaces que contiene el mismo, *dichos enlaces deben cumplir las siguientes características:*

- Pertenecer al mismo sitio web.
- No debe ser a la misma página web.
- No debe ser a un documento.
- No debe ser a una imagen.

A partir de la url del sitio web y la profundidad, que determina la calidad de los niveles desde el sitio web, la ha definido la palabra clave Get All Site Links, que permite obtener todos los enlaces del sitio web, de sus páginas hijas y de todas las páginas descendientes que tengan un nivel igual o menor al proporcionado como argumento; para encontrar los enlaces se usa el algoritmo de búsqueda primero en anchura.

Se realizó la prueba de accesibilidad al sitio web de una universidad perteneciente al Comité Ejecutivo de la Universidad Boliviana, para verificar si cumple con las pautas de accesibilidad para el contenido web. Para que axe-core solo evalúe las WCAG 2.0, es necesario usar el argumento runOnly : [' wcag2a ' , ' wcag2aa '].

Se obtuvieron todos los enlaces de las páginas hijas de la página principal, es decir el nivel de profundidad de la búsqueda fue uno. A cada una de las páginas verificó si accesibilidad usando axe-core, los resultados fueron agregados al reporte que proporciona Robot Framework.

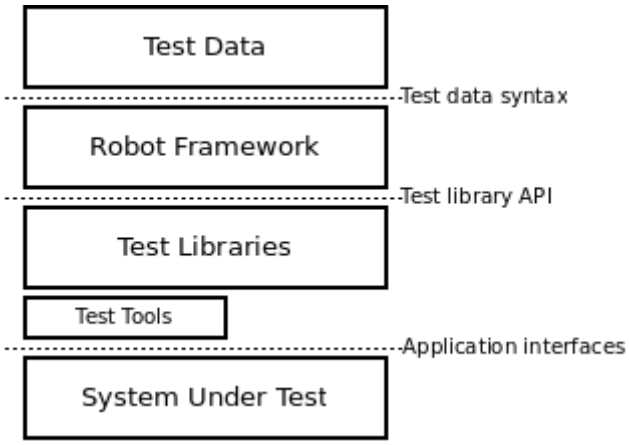
La ejecución de la prueba tuvo una duración de 11 minutos y 46 segundos donde se evaluaron 71 páginas web. En promedio las páginas no cumplían con reglas fue de 4.30, siendo 6 el número más alto de reglas no cumplidas y 3 el más bajo. Se han encontrado 5063 problemas de accesibilidad, la media de dichos problemas es de 71.31 por página web, siendo 182 la incidencia más alta y 59 la más baja. Se han incumplido siete reglas como se puede ver en la tabla 5.

Las reglas con más páginas afectadas son color-contrast, duplicate-id e image-alt. Ninguna de las páginas web cumple con la relación de contraste esperada de 4.5:1, lo cual puede dificultar la lectura a algunas personas con problemas visuales. duplicate-id es la que tiene mayor cantidad de problemas en las páginas web exigen elementos con el mismo identificador lo genera problemas de navegabilidad. image-alt aparece en todas las páginas web cómo un problema en cada una de ellas, eso se debe a que se trata de una imagen en el pie de la plantilla que no cumple con la regla.

La combinación de Robot Framework y axe-core generan un reporte muy detallado de los problemas, con información detallada sobre cada problema; además, en los casos que es posible, se incluyen sugerencias para corregir el problema. El reporte incluye enlaces a la documentación de axe-core donde se explica con mayor detalle el problema.

ILUSTRACIONES, TABLAS, FIGURAS

Figura 1. *Arquitectura de Robot Framework.*



Fuente: Robot Framework Foundation (2021).

Tabla 1. *Cantidad de criterios WCAG y reglas de axe-core.*

Nivel	Criterios de WCAG	Reglas de axe-core
A	30	16
AA	20	7
AAA	28	3

Fuente: Elaboración propia.

Tabla 2. *Comparación de marcos de trabajo para pruebas de Python*.

Nombre (Licencia)	Características
Robot (ASF)	▪ Está completamente desarrollado en Python y se utiliza para pruebas de aceptación y desarrollo dirigido por pruebas. ▪ Admite pruebas de automatización en plataformas cruzadas como Windows, Mac OS y Linux para aplicaciones de escritorio, aplicaciones móviles y aplicaciones web. ▪ Se utiliza para la automatización robótica de procesos (RPA). ▪ Hace uso de sintaxis de datos tabulares, pruebas basadas en palabras clave, bibliotecas enriquecidas y conjunto de herramientas y pruebas paralelas.
PyTest (MIT)	▪ Basado en Python que generalmente es para todo uso, pero especialmente para pruebas funcionales y de API. ▪ Admite código de texto simple o complejo para probar API, bases de datos y UI. ▪ La sintaxis simple es útil para una fácil ejecución de la prueba. ▪ Complementos enriquecidos y es capaz de ejecutar pruebas en paralelo. ▪ Puede ejecutar cualquier subconjunto específico de pruebas.

UnitTest (MIT)	▪ Basado en Python fue diseñado para funcionar con la biblioteca estándar de Python. ▪ Apoya la reutilización de conjuntos de prueba y la organización de pruebas. ▪ Admite la automatización de pruebas, incluidas las colecciones de pruebas, la independencia de las pruebas, el código de configuración para las pruebas.
DocTest (MIT)	▪ Se incluye en la distribución estándar de Python y se utiliza para pruebas unitarias de caja blanca. ▪ Busca sesiones interactivas de Python para verificar si funcionan exactamente como se requiere. ▪ Hace uso de capacidades selectivas de Python, como cadenas de documentos, el shell interactivo de Python y la introspección de Python (que determina las propiedades de los objetos en tiempo de ejecución).
Nose2 (BSD)	▪ Es un marco de pruebas unitarias basado en Python que puede ejecutar Doctests y UnitTests. ▪ Se basa en unittest, por lo que se denomina extender unittest que fue diseñado para hacer las pruebas simples y fáciles. ▪ Usa pruebas colectivas de unittest.testcase y admite múltiples funciones para escribir pruebas y excepciones. ▪ Admite accesorios, clases, módulos e inicialización compleja que se definen de una sola vez en lugar de escribir varias veces.
Testify (ASF)	▪ Diseñado para reemplazar a UnitTest y Nose. Tiene funciones más avanzadas que UnitTest. ▪ Es popular como una implementación de Java de pruebas semánticas (fácil de aprender e implementar la especificación de pruebas de software). ▪ Es fácil realizar pruebas automatizadas de unidad, integración y de sistema.

Fuente: Software Testing Help (2021).

Tabla 3. *Bibliotecas incluidas en Robot Framework*.

Biblioteca	Descripción	Palabras clave
BuiltIn	Contiene palabras clave genéricas que a menudo se necesitan. Importado automáticamente y por lo tanto siempre disponible.	108
Collections	Contiene palabras clave para manejar listas y diccionarios.	43
DateTime	Admite la creación y verificación de valores de fecha y hora, así como cálculos entre ellos.	8
Dialogs	Admite pausar la ejecución de la prueba y obtener información de los usuarios.	5
OperatingSystem	Permite realizar varias tareas relacionadas con el sistema operativo.	56

Process	Apoya la ejecución de procesos en el sistema.	15
Screenshot	Proporciona palabras clave para capturar y almacenar capturas de pantalla del escritorio.	3
String	Biblioteca para manipular cadenas y verificar su contenido.	32
Telnet	Admite conectarse a servidores Telnet y ejecutar comandos en las conexiones abiertas.	20
XML	Biblioteca para verificar y modificar documentos XML.	37
Total		327

Fuente: Robot Framework Foundation (2021).

Tabla 4. *Bibliotecas no incluidas en Robot Framework.*

Biblioteca	Descripción	Palabras clave
SeleniumLibrary	Utiliza los módulos de Selenium WebDriver internamente para controlar un navegador web (Robot Framework Foundation, 2021).	173
axelibrary	Envoltura de axe-selenium-python para pruebas de accesibilidad (Adirala & Bollweg, 2021).	3

Fuente: Elaboración propia.

Tabla 5. *Reglas incumplidas.*

Regla	Páginas afectadas	Problemas	Promedio	Máximo	Mínimo
color-contrast	71	516	7.27	118	1
duplicate-id	71	4224	59.49	65	57
frame-title	22	22	0.31	1	0
image-alt	71	71	1.00	1	1
label	1	1	0.01	1	0
link-name	68	228	3.21	31	0
nested-interactive	1	1	0.01	1	0

Fuente: Elaboración propia.

CONCLUSIONES

Axe-core es una herramienta para realizar pruebas de accesibilidad compatible con las WCAG, que proporciona bastante documentación y es fácil de utilizar.

Robot Framework es un marco de trabajo para realizar pruebas automatizadas tanto a personas con altas habilidades en desarrollo de software como a personas con pocas o nulas habilidades en programación. Escribir las pruebas en un lenguaje parecido a un lenguaje humano facilita el trabajo de automatización de pruebas.

Para usar en conjunto axe-core y Robot Framework fue necesario utilizar axe-selenium-python que genera un envoltorio para usar axe-core desde Python y finalmente se utilizó la biblioteca, para Robot Framework, llamada axelibrary que es un envoltorio para usar axe-selenium-python desde Robot Framework.

Se realizó la prueba de accesibilidad de manera automática a la página web de una universidad boliviana, se analizaron 71 páginas en menos de 12 minutos, obteniendo un reporte detallado de todos los problemas.

Con los artefactos de software desarrollados es posible realizar pruebas de accesibilidad a cualquier sitio o aplicación web a partir de su dirección web, lo cual permite mejorar la aplicación de las pruebas de accesibilidad en vista que es posible analizar un sitio o aplicación web en poco tiempo.

LISTA DE REFERENCIAS

Adirala, S., & Bollweg, N. (2021). *robotframework-axelibrary*. https://github.com/adiralashiva8/robotframework-axelibrary

Bolivia. (2012). *Ley N° 223. Ley General Para Personas con Discapacidad*. Gaceta Oficial del Estado Plurinacional de Bolivia.

Deque Systems Inc. (2021, 11 5). *axe-core*. https://github.com/dequelabs/axe-core/blob/develop/README.md

Deque Systems Inc. (2022, 7 12). *Release 4.4.3*. https://github.com/dequelabs/axe-core/releases/tag/v4.4.3

Docutils. (2021). *reStructuredText*. https://docutils.sourceforge.io/rst.html

Foord, M. (2017). *30 best practices for software development and testing*. https://opensource.com/article/17/5/30-best-practices-software-development-and-testing.

Henry, S. L. (2021). *Introducción a las Pautas de Accesibilidad para el Contenido Web (WCAG)*. https://www.w3.org/WAI/standards-guidelines/wcag/es

Henry, S. L. (2021). *Resumen de los estándares de accesibilidad de W3C*. https://www.w3.org/WAI/standards-guidelines/es.

Henry, S. L., & Dick, W. (2018). *WCAG 2.1 de un vistazo*. https://www.w3.org/WAI/standards-guidelines/wcag/glance/es

Robot Framework Foundation. (2021). *Keyword Documentation*. https://robotframework.org/SeleniumLibrary/SeleniumLibrary.html

Robot Framework Foundation. (2021). *Robot Framework documentation*. https://robotframework.org/robotframework/

Robot Framework Foundation. (2021). *Robot Framework User Guide*. https://robotframework.org/robotframework/latest/RobotFrameworkUserGuide.html

Sereduck, K. (2018). *axe-selenium-python 2.1.6*. https://pypi.org/project/axe-selenium-python/

Software Testing Help. (2021, 11 30). *Top 6 BEST Python Testing Frameworks*. https://www.softwaretestinghelp.com/python-testing-frameworks/

World Wide Web Consortium. (2020). *Web Content Accessibility Guidelines (WCAG) 2 Level A Conformance*. https://www.w3.org/WAI/WCAG2A-Conformance