



UNIVERSIDAD CENTRAL DE VENEZUELA
FACULTAD DE CIENCIAS
ESCUELA DE COMPUTACIÓN

IMPLEMENTACIÓN DE UNA APLICACIÓN PARA DETECCIÓN AUTOMATIZADA DE CAMBIOS EN PÁGINAS WEB PARA EL PROTOTIPO DEL ARCHIVO WEB DE VENEZUELA

**Trabajo Especial de Grado presentado ante la ilustre
Universidad Central de Venezuela por
Br. Luis Aguiar**

**Para optar al título de Licenciado en Computación
Tutora: Profa. Mercy Ospina
Caracas, 2019**

Acta

Quienes suscriben, miembros del jurado designado por el Consejo de la Escuela de Computación, para examinar el Trabajo Especial de Grado titulado "Implementación De una Aplicación para Detección Automatizada de cambios en Páginas Web para el Prototipo del Archivo Web de Venezuela " y presentado por el Bachiller Luis Rafael Aguiar Castro, cédula de identidad N° V – 20.674.812, a los fines de optar al título de Licenciado en Computación, dejamos constancia de lo siguiente:

Leído como fue dicho trabajo, por cada uno de los miembros del jurado, se fijó el día dieciséis (16) de octubre de 2019, a las 8:00 a.m. horas, para que el autor lo defendiera en forma pública, lo que éste hizo en la Sala PAIII de la Escuela de Computación, mediante una presentación oral de su contenido, luego de lo cual respondió a las preguntas formuladas. Finalizada la defensa pública del Trabajo Especial de Grado, el jurado decidió aprobar con la nota de 20 puntos.

En fe de lo cual se levanta la presente Acta, en Caracas el día 16 de octubre de 2019.



Profá. Ospina Mercy (Tutora)



Profá. Carballo Yusneyi (Jurado)



Prof. Andres Sanója (Jurado)

Agradecimientos y Dedicatorias

El presente trabajo se lo dedico a mis padres por haberme dado la vida y haber estado siempre ahí para mí, siendo pilares incondicionales de apoyo y amor a lo largo de mi existencia; a mi abuela Naylet, por ser mi segunda madre y ayudar a criarme y hacerme un hombre de bien de la mejor manera que pudo, apoyarme cuando lo necesitaba y ayudarme constantemente a pesar de las diferencias de opiniones que a veces pudimos tener; a mi abuelo Orlando, por mostrarme con el ejemplo como ser una persona de bien, por crear en mí un amor incondicional a la Universidad desde mi tierna infancia, demostrándome que no es sólo una casa de estudios, sino un hogar que siempre me ha recibido y que siempre estará en mi corazón. A mi abuela Iraida y mi tía Yoleida, por siempre recibirme en sus hogares y crear en mí el amor a aprender y aprender y seguir aprendiendo, ya que esto lograría expandir mis horizontes, a mi tío Orlando Antonio, por siempre ayudarme sin pedir algo a cambio con todo lo que estuviese dentro de su poder.

A la Universidad Central de Venezuela, por ser un segundo hogar para mí a lo largo de toda mi carrera, a mis profesores, por guiarme durante cada uno de estos años con sus lecciones tanto dentro como fuera del aula de clases, al Centro de Investigación de Sistemas de Información, donde conocí personas maravillosas como la profesora Mercy y la profesora Tina, que me mantuvieron motivado a terminar mis estudios de pregrado a pesar de las dificultades por las cuales está pasando el país.

A mi novia Roselyn, por nunca faltar cuando lo necesitaba, por darme su apoyo y su amor de manera incondicional y convertir mi vida tanto dentro como fuera de la universidad en una experiencia aún más maravillosa y darme motivos para ser mejor cada día.

A todos mis compañeros estudiantes, tanto de mi carrera como de otras, ya que ellos lograron que mi vida estudiantil fuese algo más que sólo participar en las clases, con quienes pasé períodos difíciles y no tan difíciles, pero todos igualmente divertidos, durante éstos últimos años.

A todas las personas que nombro aquí (y algunas que probablemente se me estén olvidando, les ruego me disculpen por eso), de verdad quiero, desde el fondo de mi corazón, decirles **“Gracias por todo, no lo hubiese logrado de no haberlos conocido”**.

Resumen

El patrimonio digital, definido por la Unesco en su Carta sobre la prevención del patrimonio digital (UNESCO, 2009), está conformado por todos aquellos contenidos culturales cuyo origen es digital, que incluye tanto los sitios web como el contenido audiovisual, los documentos y libros digitales. En relación con los contenidos publicados en la web, debido a la naturaleza volátil que tiene este medio, tiende a desaparecer con el tiempo, por lo que si no existe un proceso o sistema que se encargue de almacenarlo, se pierde, es decir, el patrimonio digital no es auto-preservable. Debido a esta ocurrencia, se han venido desarrollando diversos sistemas como los Archivos Web cuyo propósito es, precisamente, guardar el patrimonio digital para su preservación, para generaciones futuras. El presente Trabajo Especial de Grado, que forma parte del Prototipo de Archivo Web de Venezuela (desarrollado en la Escuela de Computación de la Facultad de Ciencias de la Universidad Central de Venezuela) tiene como objetivo desarrollar una aplicación Web para la detección de cambios en los sitios web, con el fin de automatizar este proceso con miras a dar soporte a los Módulos de Adquisición y Almacenamiento de contenidos, y al Módulo de Predicción de Cambios. La presente aplicación analiza el contenido del código de dos versiones de un sitio web, uno almacenado y otro en proceso de adquisición, con el fin de detectar y calcular una ponderación de los cambios entre ellas, así como la ubicación espacial de dichas modificaciones, para calcular la importancia de los cambios lo que servirá de entrada al módulo de almacenamiento para decidir si se almacena la versión nueva en el Archivo. La metodología usada durante el desarrollo de la aplicación fue una variación de eXtreme Programming, entre otras razones, debido al tamaño del equipo, su compatibilidad para el desarrollo de diferentes componentes, cada cual está enmarcado dentro de una iteración y la gran tolerancia a cambios en los requerimientos que esta filosofía profesa. Los resultados obtenidos fueron satisfactorios, logrando la realización de la detección de los cambios que se presentan en el código de las páginas web que sean recibidas, adicionalmente la aplicación logra determinar la importancia de los cambios de acuerdo con su ubicación espacial con respecto al navegador al momento de realizar la renderización del sitio analizado.

Palabras clave: Archivo Web, detección, cambios, Importancia, preservación Web

Tabla de contenido

ACTA.....	ERROR! BOOKMARK NOT DEFINED.
AGRADECIMIENTOS Y DEDICATORIAS.....	C
RESUMEN.....	D
TABLA DE CONTENIDO	5
ÍNDICE DE FIGURAS	7
ÍNDICE DE TABLAS	9
INTRODUCCIÓN.....	10
CAPÍTULO 1 PLANTEAMIENTO DEL PROBLEMA	12
1.1. EL MÓDULO DE ADQUISICIÓN.....	13
1.2. JUSTIFICACIÓN	17
1.3. OBJETIVOS	17
1.3.1. <i>Objetivo General</i>	17
1.3.2. <i>Objetivos Específicos</i>	17
1.4. ALCANCE	17
CAPÍTULO 2 MARCO TEÓRICO.....	19
2.1. PRESERVACIÓN WEB.....	19
2.2. ARCHIVO WEB	19
2.3. PRINCIPIO DE PARETO	20
2.4. HERRAMIENTAS TECNOLÓGICAS	21
2.4.1. <i>Python</i>	21
2.4.2. <i>Flask</i>	22
2.4.3. <i>MVC</i>	22
2.4.4. <i>MTV</i>	23
2.4.5. <i>Servicios Web</i>	24
2.4.6. <i>Módulos Python</i>	28
2.4.7. <i>Detección de cambios en páginas web</i>	29
CAPÍTULO 3 MARCO PROCEDIMENTAL.....	34
3.1. METODOLOGÍAS DE DESARROLLO DE SOFTWARE.....	34
3.1.1. <i>Metodologías Tradicionales</i>	34
3.1.2. <i>Metodologías Ágiles</i>	36
3.2. ARQUITECTURA DE SOFTWARE BASADA EN COMPONENTES	41
3.2.1. <i>Herramientas que pertenecen al desarrollo de software basado en componentes</i>	41

CAPÍTULO 4 MARCO APLICATIVO	45
4.1. DEFINICIÓN DE REQUERIMIENTOS	45
4.2. OBJETIVO DE LA APLICACIÓN.....	45
4.3. ARQUITECTURA DE LA SOLUCIÓN PROPUESTA	45
4.4. METAS DE LA APLICACIÓN	46
4.5. ADAPTACIÓN DE LA METODOLOGÍA EXTREME PROGRAMMING (XP) USANDO UNA ARQUITECTURA DE SOFTWARE BASADA EN COMPONENTES	47
4.5.1. <i>Primera Iteración (Investigación):</i>	47
4.5.2. <i>Segunda Iteración (Cambio cuantitativo):</i>	51
4.5.3. <i>Tercera Iteración (Desarrollo de la Interfaz):</i>	56
4.5.4. <i>Cuarta Iteración (Cambio cualitativo):</i>	64
4.5.5. <i>Quinta Iteración (Pruebas funcionales del sistema):</i>	70
CONCLUSIONES Y RECOMENDACIONES	83
ANEXOS	85
5.1. SCRIPTS DE HERITRIX.....	85
REFERENCIAS BIBLIOGRÁFICAS Y ELECTRÓNICAS.....	88

Índice de figuras

FIGURA 1 ACTIVIDADES PARA LA PRESERVACIÓN WEB (Ospina, Martinez, Leon, & Kabchi, 2014).....	12
FIGURA 2 ARQUITECTURA DEL ARCHIVO WEB (Ospina, Martinez, Leon, & Kabchi, 2014).....	13
FIGURA 3 COMPONENTES DEL MÓDULO DE ADQUISICIÓN (Carabali & Casanova, 2014).....	14
Figura 4 Algoritmo de predicción simple (Carabali & Casanova, 2014)	15
Figura 5 Clasificación de Sitios Web basado en el Algoritmo Simple de Predicción (Carabali & Casanova, 2014).....	16
Figura 6 Cambio Importante.....	18
FIGURA 7 PATRÓN MVC (Patrón Modelo Vista Controlador, n.d.)	22
FIGURA 8 FLUJO WEB SERVICE (W3C, 2004).....	25
FIGURA 9 FASES DEL MSF (Microsoft Corporation, 2006).....	35
Figura 10 Interfaces de componentes (Sommerville, Ingeniería del Software, 2005) ...	43
Figura 11 ELEMENTOS DEL MODELO DE COMPONENTES (Sommerville, Ingeniería del Software, 2005).....	43
Figura 12 TIPOS DE COMPOSICIÓN DE COMPONENTES (Sommerville, Ingeniería del Software, 2005).....	44
Figura 13 Arquitectura del Prototipo del AWW.....	46
Figura 14 Arquitectura de la aplicación	46
Figura 15 Instalación Python.....	50
Figura 16 Diseño Historia de Usuario 1. Iteración 2.....	52
Figura 17 Importación de dependencias	53
Figura 18 Scraping y almacenamiento del código HTML.....	53
Figura 19 Porcentaje de diferenciación entre bloques	54
Figura 20 Ponderación cuantitativa según el Principio de Pareto	55
Figura 21 Diseño Historia de Usuario 1. Iteración 3.....	58
Figura 22 Diseño Historia de Usuario 2. Iteración 3.....	58
Figura 23 Historia de Usuario 1. Iteración 1. Instalación Flask	59
Figura 24 Ejemplo de uso de plantillas en Flask.....	59
Figura 25 Archivo app.py	60
Figura 26 Reconstrucción y captura del sitio web actualizado.....	61

Figura 27 Creación gráfico de barras con resultados.....	62
Figura 28 Representación de Sitio Web en una Matriz de Posición.....	66
Figura 29 Definición de Segmentos en el Sitio Web	67
Figura 30 Modificación de filas en Matriz de Posición.....	67
Figura 31 Localización espacial de Bloques con Cambios.....	68
Figura 32 Correlación de posición entre Bloques con Cambios y Matriz de Posición ...	68
Figura 33 Ponderación de Importancias Cualitativa y Cuantitativa	69
FIGURA 34 SCRIPT CREAR_JOB_EN_REMOTO.SH (Carreño, 2016).....	85
FIGURA 35 SCRIPT MOVER_ARCHIVO.SH (Carreño, 2016)	85
FIGURA 36 SCRIPT MOVER_ARCHIVO.SH (Carreño, 2016)	86

Índice de Tablas

Tabla 1 Comparativo entre enfoques metodológicos	36
Tabla 2 Características de los Componentes.....	42
Tabla 3 Historia de Usuario 1. Iteración 1	48
Tabla 4 Historia de Usuario 2. Iteración 1	48
Tabla 5 Caso de Prueba 1. Iteración 1	50
Tabla 6 Caso de Prueba 2. Iteración 1	51
Tabla 7 Historia de Usuario 1. Iteración 2	51
Tabla 8 Caso de Prueba 1. Iteración 2.....	55
Tabla 9 Historia de Usuario 1. Iteración 3	56
Tabla 10 Historia de Usuario 2. Iteración 3	57
Tabla 11 Caso de Prueba 1. Iteración 3.....	62
Tabla 12 Caso de Prueba 2. Iteración 3.....	63
Tabla 13 Caso de Prueba 3. Iteración 3.....	63
Tabla 14 Historia de Usuario 1. Iteración 4	64
Tabla 15 Caso de Prueba 1. Iteración 4.....	69
Tabla 16 Historia de Usuario 1. Iteración 5	70
Tabla 17 Caso de Prueba 1. Iteración 5.....	72
Tabla 18 Caso de Prueba 2. Iteración 5.....	72
Tabla 19 Caso de Prueba 3. Iteración 5.....	74
Tabla 20 Caso de Prueba 4. Iteración 5.....	75
Tabla 21 Caso de Prueba 5. Iteración 5.....	76
Tabla 22 Caso de Prueba 6. Iteración 5.....	77
Tabla 23 Caso de Prueba 7. Iteración 5.....	78
Tabla 24 Caso de Prueba 8. Iteración 5.....	79
Tabla 25 Caso de Prueba 9. Iteración 5.....	81
Tabla 26 Caso de Prueba 10. Iteración 5.....	82

Introducción

El patrimonio cultural es aquello que ha sido considerado como relevante para la historia de una sociedad, comunidad o pueblo y que forma parte de sus tradiciones, costumbres, folclor y educación. Es posible realizar una división de este en dos grandes grupos, los cuales se suelen denominar “patrimonio tangible” y “patrimonio intangible, definido por la Unesco en su Carta sobre la prevención del patrimonio digital (UNESCO, 2009), entra dentro de esta última categoría, donde están , archivos multimedia, documentos y libros digitales, y entre ellos podemos destacar las páginas web que se ha denominado .

La preservación del patrimonio cultural es un deber de la humanidad para salvaguardar el conocimiento que allí se refleja y los Archivos Web (Web Archive, WA), son Sistemas de Información usados para resguardar el contenido de sitios web, proceso denominado preservación de la web o preservación web. La naturaleza de los contenidos web, los cuales cambian constantemente, hace que este proceso, brinde un histórico de los cambios que se presentan en estos, siendo un insumo importante para aquellos que deseen realizar estudios sociales, antropológicos o tecnológicos.

En Venezuela se está desarrollando un prototipo de Archivo Web, que cumple con las funciones principales de adquisición, almacenamiento y acceso a los contenidos preservados que caracterizan a este tipo de sistemas. Dicho prototipo es responsabilidad del Centro de Investigación de Sistemas de Información de la UCV.

Para mantener el histórico nombrado anteriormente se debe realizar una adquisición regular de los sitios web que se preservan, para lo cual es importante determinar si los contenidos han cambiado, sin necesidad de intervención de un humano. En el presente trabajo se propone una implementación de la aplicación de Detección de Cambios para el mencionado Archivo Web que permita determinar las variaciones entre dos versiones de un sitio de manera eficiente y automatizada, reduciendo el tiempo usado para la revisión (la cual actualmente se realiza de forma manual) y los recursos de almacenamiento, ya que sólo serán guardadas las versiones que contengan una variación que sea relevante de acuerdo a los cálculos realizados por la aplicación.

El presente documento está estructurado en cuatro (04) capítulos de la siguiente manera:

En el Capítulo 1 se expresan los motivos por los cuales se realizó el presente Trabajo Especial de Grado, así como los objetivos a lograr que fueron establecidos para el desarrollo de este.

En el Capítulo 2 se mencionan las definiciones teóricas y las herramientas tecnológicas que fueron utilizadas como base fundamental para el desarrollo completo del presente trabajo.

En el Capítulo 3 son explicadas las metodologías de desarrollo consideradas para la realización del proyecto y se detalla cuál fue escogida y las razones detrás de la elección.

En el Capítulo 4 se exponen las tareas realizadas en conjunto con la aplicación de los conceptos explicados anteriormente con el fin de llevar a cabo exitosamente el alcance de los objetivos planteados, de igual forma se describen de manera separada cada una de las fases de desarrollo que fueron definidas por la metodología establecida.

Capítulo 1

PLANTEAMIENTO DEL PROBLEMA

Un archivo web es un sistema de información cuyo propósito es preservar de manera histórica contenido web de interés como patrimonio cultural. Para ello, deben llevarse a cabo un conjunto de tareas (Masanés, 2006) que hagan factible la preservación de la web. Estas tareas son preservadas en la Figura 1



FIGURA 1 ACTIVIDADES PARA LA PRESERVACIÓN WEB (Ospina, Martínez, Leon, & Kabchi, 2014)

- **La selección** permite limitar el ámbito del archivo, pudiendo preservar contenidos locales o de un tipo en particular. Por ejemplo, contenidos de un país o educativos.
- **La adquisición** logra que se puedan almacenar los cambios que se generan sobre los contenidos que se preservan a través del tiempo
- **El almacenamiento** requiere estrategias que permitan preservar grandes volúmenes de información (del orden de los Terabytes), archivos (millones de archivos) y formatos.
- **El acceso o recuperación** de los contenidos está estrechamente ligado a la forma en que se encuentran almacenados, pero debido a la naturaleza hipertextual y multimedia de la web, se espera que el usuario final pueda acceder a este contenido de manera similar a cuando lo hace en los servidores originales.

Actualmente en la Escuela de Computación de la Universidad Central de Venezuela UCV se está desarrollando, como un proyecto de investigación, un prototipo de Archivo Web basado en tecnologías de software libre, dirigido a preservar sitios Web de Venezuela. La arquitectura general de dicho sistema se puede observar en la Fig. 2, donde se distinguen los tres módulos que se describen a continuación:

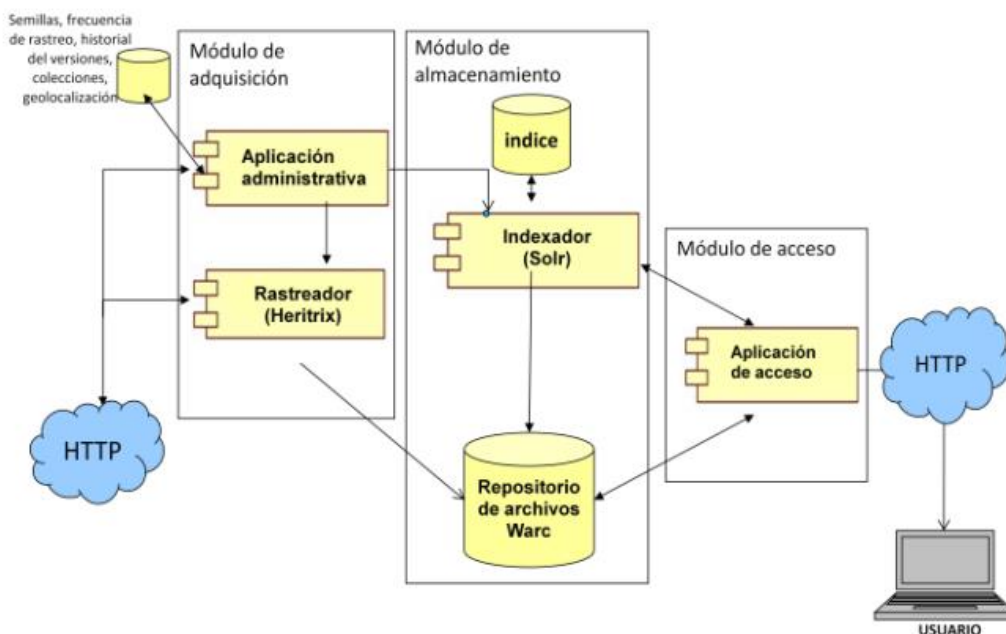


FIGURA 2 ARQUITECTURA DEL ARCHIVO WEB (Ospina, Martínez, Leon, & Kabchi, 2014)

- *El módulo de adquisición:* encargado de obtener de manera regular una copia de los documentos web que confirman los sitios web seleccionados. Cada adquisición realizada en una fecha dada se considera una versión. El contenido es obtenido a través del rastreador Heritrix, el cual genera archivos en formato WARC por cada versión de un sitio web rastreado en una fecha dada.
- *El módulo de almacenamiento e indexación:* encargado de almacenar de manera histórica las versiones de los sitios web adquiridos y generar un índice para su búsqueda. Recibe los archivos WARC, los asocia a una URL y los indexa para permitir búsquedas por palabra clave [URL] o por colecciones [agrupaciones de sitios web de acuerdo con una categoría determinada] que da como resultado un listado de los archivos WARC correspondientes a las versiones del sitio web que corresponde a dicha URL.
- *El módulo de acceso:* permite recuperar las versiones de los sitios web almacenados de manera que se pueden visualizar y navegar de la misma forma en que se realizó originalmente. Accede a los contenidos almacenados en formato WARC, permitiendo que los usuarios puedan buscar las versiones de un sitio y seleccionar la versión que desean visitar.

1.1. El módulo de adquisición

El módulo de adquisición es la entrada del sistema. Es el primer paso del proceso de preservación. En la Fig. 3 podemos observar las tareas principales este módulo e identificar sus componentes principales.

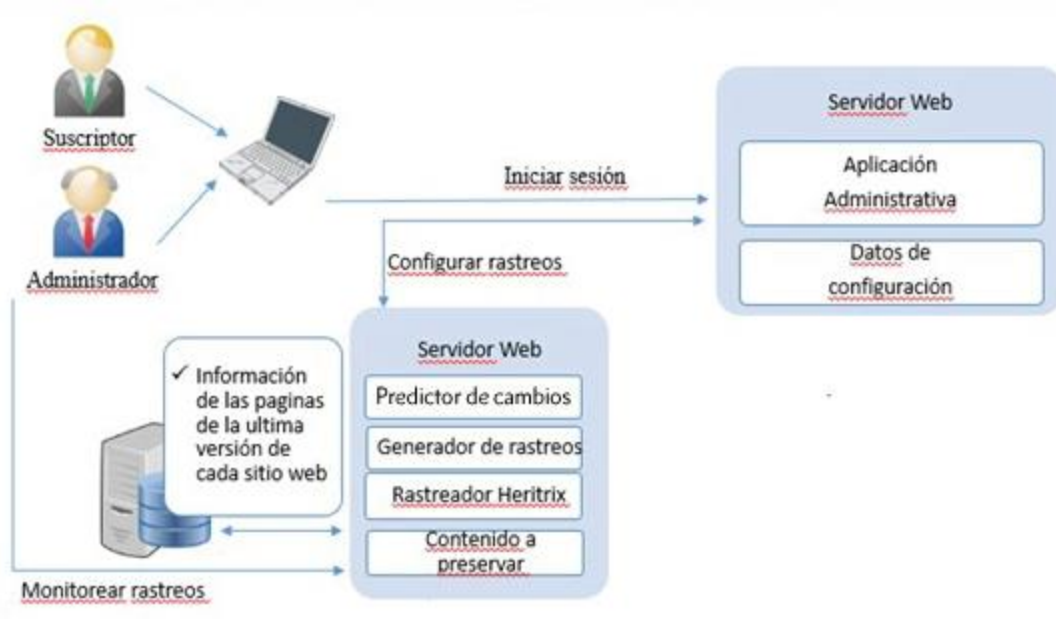


FIGURA 3 COMPONENTES DEL MÓDULO DE ADQUISICIÓN (Carabali & Casanova, 2014)

Heritrix es un rastreador web, una herramienta que busca preservar objetos digitales con el fin de disponer de ellos en el futuro con fines culturales o fuentes de investigación. Existen actualmente tres máquinas virtuales con esta aplicación. El objetivo es poder realizar rastros simultáneos en varios sitios. Los datos generados por los rastreadores son transferidos a una máquina central donde se almacena la salida de los rastros para luego ser transferidos al módulo de almacenamiento.

Para hacer funcionar el rastreador del archivo web sin participación del usuario (en segundo plano), es necesario que se ejecuten una serie de shell scripts los cuales se pueden observar en los anexos que llaman al API REST de Heritrix, los cuales pueden observarse .

Actualmente, el Archivo Web de la UCV cuenta con un sencillo módulo de predicción implementado en Python. Este módulo, como su nombre lo dice, es el responsable de inferir cuándo un sitio que ha sido rastreado ha sufrido cambios suficientes para considerar su estado actual como uno nuevo, diferente al almacenado previamente. Este módulo fue desarrollado como parte del Trabajo Especial de Grado de Marco Casanova y Willibert Carabali. Su función es ejecutar los rastros sucesivos o versiones de los sitios web que se desean preservar, buscando minimizar los rastros innecesarios, es decir de sitios que no han cambiado.

Carabali y Casanova desarrollaron el módulo de predicción basado en la distribución Bernoulli para modelar la situación de cada página. Se planteó el siguiente escenario. Sea T el tiempo transcurrido entre dos mediciones consecutivas de un mismo sitio web. Inicialmente $T = 24h$ indica el valor mínimo que puede tomar T . Si en estas mediciones consecutivas no se detectan cambios en el sitio, entonces se procede a aumentar gradualmente T , mediante el cálculo $T = T + 24$ (hasta un máximo de $T = 720$,

que representa 30 días). En caso de detectar algún cambio en una medición de dicho sitio, debemos reducir T por medio de la función $T = \text{Min}(T, 24)$.

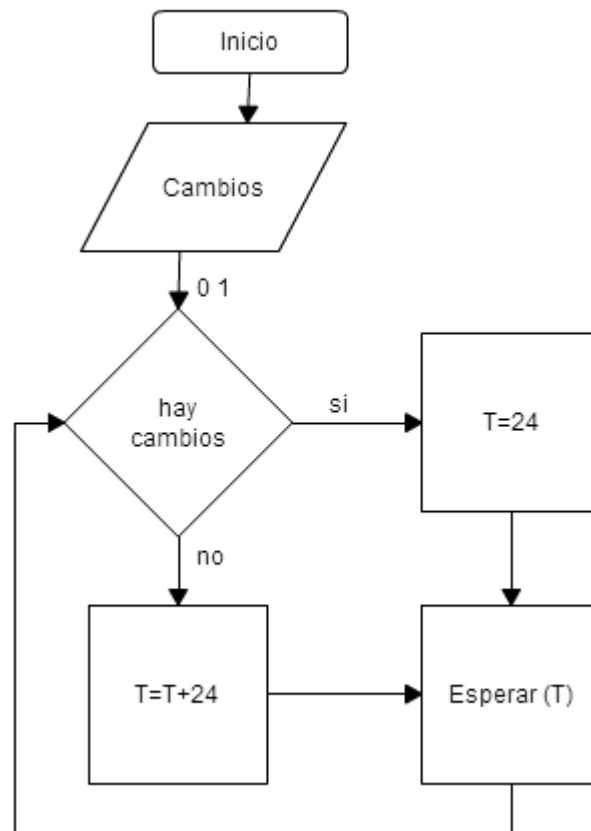


Figura 4 Algoritmo de predicción simple (Carabali & Casanova, 2014)

En base a este planteamiento, Carabali y Casanova desarrollaron un algoritmo de predicción simple que puede ser observado en la Figura 4. Este algoritmo sirvió como base para su trabajo. A partir de dicho algoritmo, implementaron otros que les permitieran cubrir la mayor cantidad de casos posibles. Mediante el uso de este algoritmo, crearon una clasificación que puede ser observada en la Figura 5 para los sitios web que basada en la frecuencia de los cambios de manera tal que se puedan desarrollar políticas de rastreo basadas en el estudio realizado minimizando así la cantidad de rastreos duplicados y de cambios perdidos

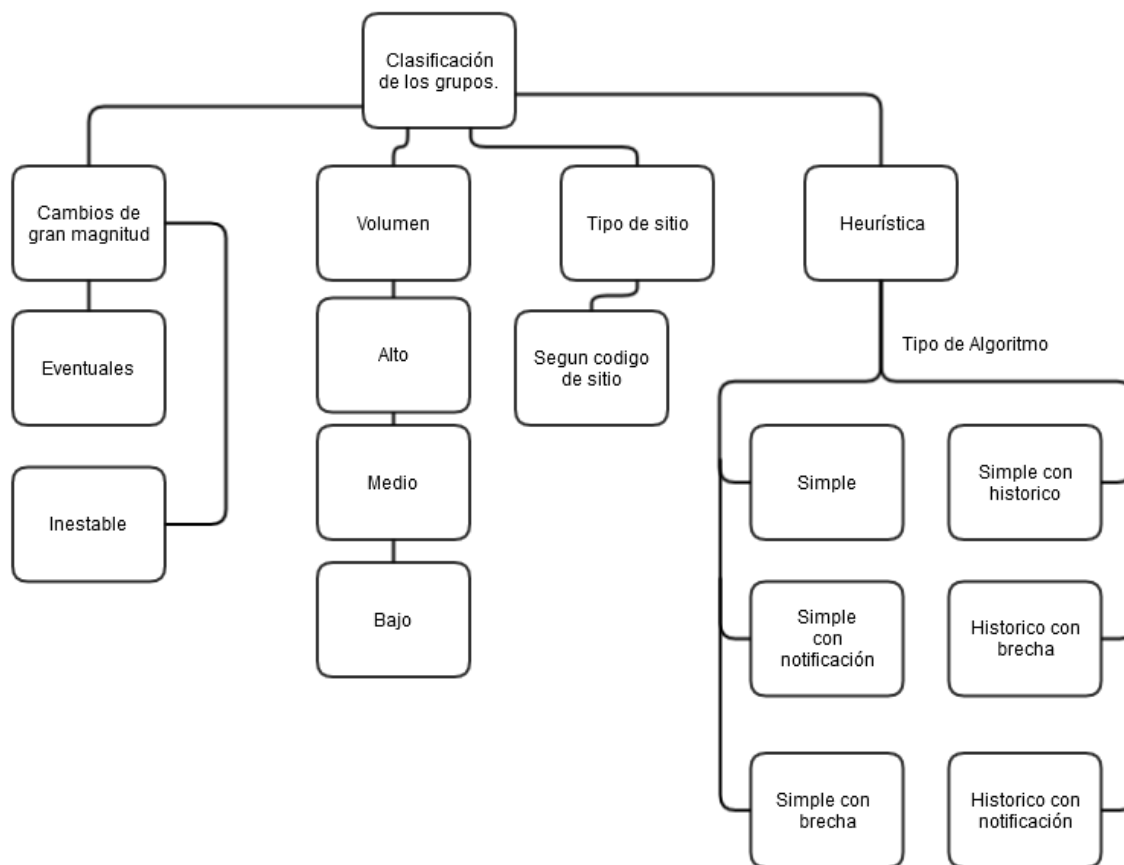


Figura 5 Clasificación de Sitios Web basado en el Algoritmo Simple de Predicción (Carabali & Casanova, 2014)

Actualmente, el predictor del archivo web está completamente separado del módulo de adquisición. A menos que el administrador verifique los registros de la aplicación, no hay manera de saber si se tomó la decisión de generar un nuevo rastreo pues no existe una comunicación entre este componente y el módulo de administración.

La aplicación administrativa es el punto de entrada del sistema. Un visitante debe ser capaz de crear una cuenta de usuario y de acceder a la información estática del sitio como, por ejemplo, el objetivo del sistema, los responsables del proyecto, las preguntas frecuentes, etc.

Un usuario del archivo web, debe ser capaz de iniciar sesión, agregar sitios que desee preservar al archivo, consultar los sitios archivados y gestionar sus datos de usuario.

Un administrador del archivo web, debe ser capaz de acceder a toda la información pública de los usuarios, al contenido del archivo y, principalmente, debe poder configurar parámetros de búsqueda, de rastreo y detección de cambios, así como programar trabajos del sistema.

1.2. Justificación

Actualmente, los componentes principales del módulo de adquisición presentan una serie de fallas que limitan su labor y minimizan los resultados que se pueden obtener del módulo.

- No existe una interfaz donde el administrador pueda modificar los parámetros de detección.
- La detección de cambios realizada no está siendo automatizada.
- El usuario es el que debe comparar ambos archivos, ya sea mediante revisión de código o visualización de ambos sitios web luego de su respectiva carga en el navegador.

1.3. Objetivos

Los objetivos que se desean alcanzar con el presente trabajo se especifican a continuación.

1.3.1. Objetivo General

Implementar una aplicación de detección automatizada de cambios en páginas web para el prototipo del Archivo Web de Venezuela.

1.3.2. Objetivos Específicos

1. Desarrollar los componentes que conformarían la aplicación de detección de cambios, usando como base métodos matemáticos, estadísticos y visuales.
2. Elaborar una interfaz administrativa que permita al usuario acceder y monitorear los cambios, así como la importancia de cada uno respectivo al resto.
3. Establecer una clasificación temporal de acuerdo con la importancia de cada uno de los cambios dependiendo de su densidad de texto y ubicación espacial dada la renderización del sitio web.
4. Realizar pruebas funcionales.

1.4. Alcance

Esta investigación desarrolla una aplicación que permite la detección de cambios en páginas web y una medida para su importancia. Para ello se usarán técnicas de detección de cambio basadas en segmentación, las cuales serán detalladas en el Capítulo 2, tales como la densidad de texto de los diferentes elementos, la ubicación espacial del cambio en la página renderizada, y el principio de Pareto.

También se ubican las métricas relevantes recolectadas durante la adquisición y el almacenamiento del contenido, para realizar los cálculos cuyos resultados serán mostrados en la aplicación de consulta. En este estudio un cambio es considerado importante si la suma de la importancia cuantitativa (determinada por el análisis del texto) y la importancia cualitativa (la cual se calcula tomando como base la posición del bloque cambiado en el visualizador) da como resultado un valor entre cero punto cinco (0.5) y uno (1), es decir, si cumple con la siguiente condición:

Sea $I(x)$ una función que retorna un valor donde $0 < I(x) < 1$, sea $I_c(y)$ una función que retorna un valor donde $0 < I_c(y) < 1$ y cuyo valor sea la importancia cuantitativa (importancia que se obtiene al calcular la diferencia basada en la densidad del texto), sea α la importancia relativa de los cambios cuantitativos con respecto a la importancia total, sea $I_q(z)$ la función que retorna un valor donde $0 < I_q(z) < 1$ y cuyo valor sea la importancia cualitativa (la cual se obtiene al calcular el impacto del cambio dependiendo de su ubicación respecto al visualizador), sea β la importancia relativa de los cambios cualitativos respecto a la importancia total de los mismos. La Figura 6 determina si un cambio es importante:

Si $0.5 \leq I(x) = \alpha I_c + \beta I_q \leq 1$, entonces es un cambio importante

Figura 6 Cambio Importante

La investigación se limita al desarrollo de las siguientes actividades:

- Estudio de métodos de detección de cambios previamente desarrollados dependiendo del enfoque de cada uno.
- Selección de los métodos que se consideraron más apropiados para el desarrollo de la aplicación del Archivo Web de Venezuela.
- Obtención del código HTML de diferentes tipos de sitios web para realizar el análisis necesario.
- Separación del código HTML de cada uno de los sitios web luego de su renderización completa en bloques de texto.
- Análisis y comparación de los bloques de texto de una versión del sitio web contra la versión anterior para determinar si hubo cambios.
- Cálculo de la importancia cuantitativa asociada a cada sitio web, basado en la densidad de texto de cada uno de los bloques.
- Cálculo de la importancia cualitativa asociada a cada sitio web, basado en la posición relativa de los bloques con respecto a lo que el navegador puede visualizar al momento de la carga, dada la resolución y tamaño de la ventana de este.
- Cálculo de la importancia total basado en heurísticas que determinan los valores a usar de las importancias cualitativas y cuantitativas.
- Aplicación administrativa basada en web realizada con el fin de observar los cambios de acuerdo con cada uno de los bloques y la posición de éstos, comparando a su vez con la versión anterior del sitio web.

Capítulo 2 MARCO TEÓRICO

En este capítulo se describen las teorías, conceptos procesos relacionados con la preservación web y los conceptos relacionados con la solución propuesta.

2.1. Preservación Web

La preservación Web hace referencia al proceso de recolectar información disponible en la World Wide Web y almacenarla en un formato de archivo digital, para así garantizar que el contenido pueda ser consultado posteriormente (Masanès, 2006)

De acuerdo con el Internet World Stats (Internet World Stats, 2017) aproximadamente el 49% de la población mundial tiene acceso a internet. A diario se producen millones de bytes de información que se encuentran en internet disponibles para todo el mundo. Este conocimiento generalmente persiste en el tiempo, debido a la estructura misma de la Web. Cuando se genera contenido, este está disponible para ser consultado por cualquier usuario de la red. Muchos investigadores consideran a la Web un medio auto-preservable. El contenido valioso será conservado y el que no, será destruido con el tiempo. Y cuando algo desaparece de internet, lo hace para siempre.

Por lo tanto, surge la necesidad de un sistema dedicado exclusivamente a la preservación del conocimiento en la Web. Un sistema que refuerce la capacidad de perdurar en el tiempo de dicho contenido y que facilite su acceso a los usuarios de la red. Para llevar a cabo este procedimiento, se han creado archivadores digitales, destinados a almacenar tanta información como sea posible.

Existen distintos tipos de sistema de información cuyo objetivo es siempre el mismo. Es de nuestro interés estudiar un tipo específico de sistema de información cuyo propósito es preservar la web.

2.2. Archivo Web

Para llevar a cabo el proceso de preservación web se han venido desarrollando sistemas de información cuyo objetivo es almacenar de manera histórica contenidos de la Web, mediante la replicación y/o migración de su formato original a otra representación en ambientes seguros los cuales se les denomina Archivos Web o iniciativas de preservación web (Ospina, Martinez, Leon, & Kabchi, 2014)

Como ejemplo de esto tenemos el Internet Archive, una organización sin fines de lucro que resguarda libros, películas, música, software y otros tipos de contenido Web desde hace más de 15 años. Los usuarios pueden acceder sin restricciones a todo el contenido del archivo y pueden crear y preservar colecciones de contenido digital utilizando *Archive-It* (Archive-it, 2006).

Los Archivos Web deben llevar a cabo un conjunto de tareas que hagan factible la preservación. A continuación, se presenta una breve descripción de estas tareas (Ospina, Martinez, Leon, & Kabchi, 2014)

- La **selección** permite limitar el ámbito del Archivo pudiendo preservar contenidos locales, o de un tipo en particular, por ejemplo, contenidos de un país, o educativos solamente.
- La **adquisición** logra que se puedan almacenar los cambios que se generan sobre los contenidos que se preservan a través del tiempo.
- El **almacenamiento** requiere estrategias que permitan preservar grandes volúmenes de: información (del orden de los Terabytes), millones de archivos y diferentes formatos. Para este fin se han desarrollado formatos de archivos contenedores específicos, cuyo objetivo principal es superar la limitación de los sistemas de archivos propios de los sistemas operativos donde se alojan los Archivos Web, siendo los más usados el formato ARC, desarrollado por Internet Archive, y el formato WARC un estándar reconocido desarrollado por el IIPC para la preservación web como una mejora del formato ARC.
- El **acceso** o recuperación de los contenidos está estrechamente ligado a la forma en que se encuentran almacenados, pero debido a la naturaleza hipertextual y multimedia de la web, se espera que el usuario final pueda acceder a este contenido de manera similar a cuando lo hace en los servidores originales.

Estas tareas requieren de una serie de herramientas para ser llevadas a cabo.

2.3.Principio de Pareto

El Principio de Pareto, también conocido como “la regla del 80/20” o la “Ley de los pocos vitales” es una relación que describe causalidad y resultados. Predica que el ochenta por ciento (80%) de la salida de un sistema es resultado del veinte por ciento (20%) de la entrada. La primera observación de este fenómeno fue realizada en 1906 por el economista italiano Vilfredo Pareto, en honor a quien se nombra este concepto, con relación a la tierra y la población, su análisis mostró que el 20% de la población italiana era dueña el 80% de los terrenos; Pareto confirmó este principio al analizar las vainas de guisantes en su jardín, donde también observó que el 20% de las vainas producía el 80% de los guisantes totales, luego de esto decidió realizar encuestas en diferentes países donde, de nuevo, confirmó los resultados recabados en su país natal. El principio de Pareto ha sido aplicado a una variedad de campos, desde economía y negocios (observando que el 80% de las ganancias de una empresa provienen del 20% de los clientes, así como el 80% de las quejas del negocio también son obtenidas del 20% de los clientes), hasta biología y criminología, no sólo con el fin de explicar las observaciones realizadas en el campo, sino también con la idea de ajustar las prácticas en el mismo para mejorar la eficacia. (Akunda, 2011)

Un ejemplo claro de la aplicación de este principio en áreas relacionadas con el presente trabajo es en el Desarrollo Rápido de Aplicaciones (cuyas siglas en inglés son RAD) y se determinó que era una manera eficiente de dividir el tiempo de investigación y desarrollo en diferentes modelos rápidos. (Rizwan & Iqbal, 2011)

Debido a la gran cantidad de estudios en diferentes campos que se han basado en esta regla y que han comprobado su solidez, se decidió aplicar durante el desarrollo del presente T.E.G., asumiendo que el 80% de la importancia del cambio provendrá del 20% de las divisiones que se realizarán sobre el texto. Luego de subdividir el texto del código

en base a las etiquetas `<div>` se asignará cada sección a un bloque los cuales serán ordenados en base a la cantidad de caracteres que contengan, mientras más caracteres contenga, será colocado más cerca del inicio, luego de esto, se tomará una cantidad de bloques que se aproxime al 20% del total y, éstos tendrán una ponderación final del 80% sobre el cálculo de las importancias. El resto de los bloques tendrán el 20% restante de esta ponderación.

2.4.Herramientas tecnológicas

2.4.1. Python

Python es un lenguaje de programación dinámico, orientado a objetos y multipropósito. Es multiplataforma por lo que puede ser ejecutado en la mayoría de los sistemas operativos modernos. Es distribuido bajo una licencia de código abierto gracias a la cual cuenta con una comunidad muy amplia a nivel mundial. Fue creado por Guido van Rossum y lanzado en 1991. Es notable por su sintaxis elegante orientada a facilitar la lectura y escritura de los programas, su gran cantidad de librerías que brindan soporte a muchas tareas comunes de programación como conexión a bases de datos, conexión a servidores, búsquedas avanzadas haciendo uso de expresiones regulares y muchas otras (Python Software Foundation, 2017).

Actualmente, la última versión estable es la 3.6.1, lanzada a la comunidad el 17 de diciembre de 2016. Los usuarios de Python se refieren a menudo a la Filosofía Python que es bastante análoga a la filosofía de Unix. El código que sigue los principios de Python de legibilidad y transparencia se dice que es "pythonico". Contrariamente, el código opaco u ofuscado es bautizado como "no pythonico" ("unpythonic" en inglés). Estos principios (Peters, 2004) son:

- Bello es mejor que feo.
- Explícito es mejor que implícito.
- Simple es mejor que complejo.
- Complejo es mejor que complicado.
- Plano es mejor que anidado.
- Disperso es mejor que denso.
- La legibilidad cuenta.
- Los casos especiales no son tan especiales como para quebrantar las reglas.
- Los errores nunca deberían dejarse pasar silenciosamente.
- Frente a la ambigüedad, rechaza la tentación de adivinar.
- Debería haber una y preferiblemente sólo una manera obvia de hacerlo.
- Ahora es mejor que nunca.
- Si la implementación es difícil de explicar, no está bien ideada.
- Si la implementación es fácil de explicar, puede que sea una buena idea.
- Los espacios de nombres (namespaces) son una gran idea. ¡Hagamos más de esos!

Vemos entonces que Python tiene estándares bien definidos orientados a optimizar el código creado por los programadores y que sirven como criterio para solventar ambigüedades de diseño.

2.4.2. Flask

Es un framework de peso ligero para desarrollo de aplicaciones web, fue desarrollado bajo la especificación WSGI (siglas en inglés de Interfaz de Puerta de Enlace de Servidor Web). Está diseñado para ser fácil de entender y con el que se puede iniciar a trabajar rápidamente, con la capacidad de escalar hacia aplicaciones complejas. Inició como un empaquetador simple usado alrededor de Wrezkreug y Jinja y se ha convertido en uno de los entornos más populares para desarrollo de aplicaciones web en Python. (The Pallets Projects, 2019)

Flask ofrece sugerencias, pero no hace cumplir dependencias o estructura al proyecto. El desarrollador es el que se encarga de escoger las herramientas y librerías que quiere usar. Existe una inmensa cantidad de extensiones añadidas por la comunidad que hacen que agregar nuevas funcionalidades sea relativamente sencillo.

2.4.3. MVC

MVC es un patrón de diseño de arquitectura para la implementación de la interfaz de usuario. Consiste en tres tipos de objetos. El modelo es el objeto de aplicación, la vista es la pantalla de presentación y el controlador define la manera en la que la interfaz de usuario reacciona a la entrada por parte del usuario. (Gamma, Helm, Johnson, & Vlissides, 1994).

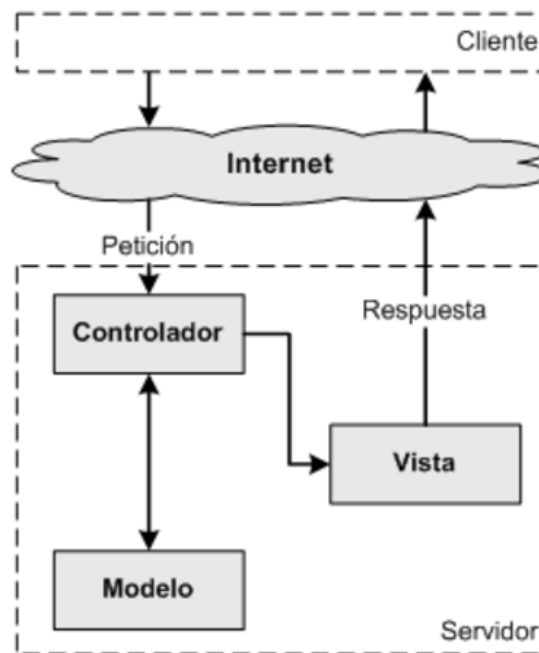


FIGURA 7 PATRÓN MVC (Patrón Modelo Vista Controlador, n.d.)

Antes de la aplicación de este patrón las aplicaciones Web caracterizaban por la existencia de componentes encargados de múltiples responsabilidades, una integración entre los objetos que atienden las solicitudes del usuario con los que se encargan de los datos. Es común aún hoy en día encontrar sitios web donde la lógica de negocio está fuertemente ligada al modelo de datos. Si se realiza un cambio en dicho modelo, toda la

aplicación debe ser reescrita. La ausencia de este patrón (u otros patrones de diseño de aplicaciones) usualmente conlleva a ignorar el principio DRY y dificulta la reutilización del código.

El flujo básico de una aplicación que sigue el patrón MVC es el siguiente. Cuando un usuario hace una solicitud a través de la interfaz la solicitud es enviada al controlador. Este se encarga de procesar la solicitud, ya sea solicitando al modelo datos requeridos por el usuario o procesando cambios en los datos y asegurando la persistencia de dichos cambios en el modelo. Una vez procesada la solicitud, se envía el resultado de esta a la donde son consumidos por el usuario.

En programación orientada a objetos (OOP) cada modelo de datos es representada por una clase. De igual forma cada controlador. Generalmente las vistas son archivos html planos o con algún motor de plantillas que permite inyectar contenido manteniendo la sintaxis externa al mínimo.

Este patrón fue introducido por primera vez en los años 70 por Trygve Reenskaug. Sin embargo, el primer documento significativo al respecto no fue publicado sino hasta 1988 por Glenn Krasner y Stephen Pope (Cunningham, 2014).

En la actualidad es la arquitectura más usada por los más notables lenguajes de programación para proyectos Web y cada lenguaje tiene uno o varios frameworks orientados a trabajar con esta arquitectura. La mayor parte de ellos ubican la lógica de negocio del lado del servidor. Sin embargo, a medida que las tecnologías del lado del cliente han mejorado, se han creado algunos frameworks que permiten a sus componentes ejecutarse del lado del cliente.

Las principales ventajas de utilizar MVC son las siguientes (Patrón Modelo Vista Controlador, n.d.):

- Permite la reutilización del código
- Facilita el desarrollo en paralelo
- Permite la agrupación de tareas relacionadas en un mismo controlador.
- Hay un bajo nivel de acoplamiento entre los componentes. Es decir, el modelo cada componente es capaz de llevar a cabo sus responsabilidades sin participación de las otras.
- El bajo acoplamiento permite que sea más sencillo el desarrollo modular
- Facilita el mantenimiento de los componentes
- Minimiza los riesgos de reemplazar o reescribir clases o métodos

2.4.4. MTV

Como establecimos previamente, el patrón MVC es un patrón de diseño de arquitectura para aplicaciones con interfaz de usuario ampliamente utilizado en aplicaciones Web. El patrón separa las tareas del sistema en tres componentes principales. El modelo representa la data y permite su manipulación. La vista representa la capa de presentación del modelo. Es lo que los usuarios ven y con lo que interactúan. El controlador se encarga de manejar el flujo de información entre el modelo y la vista. Recibe las solicitudes hechas por el usuario a través de la vista y las envía al modelo. Luego, toma la respuesta producida por el modelo y la envía a la vista (George, 2017).

Sin embargo, cada framework implementa este patrón de manera diferente. Por lo que a veces no queda claro qué responsabilidades corresponden a cada componente.

Django es un framework que se rige en términos generales por el patrón MVC, pero usa su propia lógica de implementación. El framework se encarga de ser el intermediario entre las vistas y los modelos por lo que el controlador es absorbido por la aplicación. De allí surge el patrón Modelo Plantilla Vista o (Model Template View en inglés). En este patrón de arquitectura los componentes se encargan de las siguientes tareas (Django Software Foundation, 2012):

- *El modelo* representa la capa de acceso a los datos. Esta capa contiene todo lo relacionado a los datos. Cómo accederlos, cómo validarlos, qué comportamiento tienen, cómo se relacionan, etc.
- *La plantilla* representa la capa de presentación. Define cómo se debe mostrar cualquier cosa en la Web.
- *La vista* contiene la lógica de acceso a los modelos y distingue las plantillas apropiadas para cada modelo.

2.4.5. Servicios Web

De acuerdo con la W3, un servicio web es un sistema de software diseñado para permitir la interacción punto a punto entre dos máquinas conectadas a través de una red. Generalmente se utiliza para establecer conexiones con equipos remotos y realizar un intercambio de información a través de la World Wide Web. Utiliza tecnologías como el protocolo HTTP para establecer comunicación entre las máquinas y para transferir archivos. El concepto de servicio web es una noción abstracta que debe ser implementada por un agente concreto, es decir, por el software que se encarga de enviar o recibir los mensajes. De esta manera, se puede escribir el mismo servicio utilizando distintos agentes, como navegadores o lenguajes de programación (W3C, 2004).

El objetivo de un servicio web es proveer funcionalidades en nombre de un proveedor, es decir a la persona u organización que implementa el servicio. Por ejemplo, compañías como Facebook, o Twitter crean servicios web para permitir a aplicaciones externas acceder a los datos alojados en sus servidores.

Así como existe un proveedor de servicios, existe un solicitante. Esta persona u organización desea hacer uso de la implementación de un servicio web ofrecido por un proveedor.

Para describir el funcionamiento de los servicios web, es necesario especificar en lenguaje entendible por la máquina cómo se puede llamar el servicio web, qué parámetros espera y qué tipo de datos retornará. El método más básico es haciendo uso de un archivo WSDL (Web Service Definition Language). Este es un archivo en formato XML que permite especificar la información requerida por un servicio para establecer la comunicación entre máquinas y el respectivo intercambio que se producirá entre ellas. Haciendo uso de este se puede intercambiar información a través de protocolos comunes de la Web como HTTP, FTP, SFTP y SOAP.

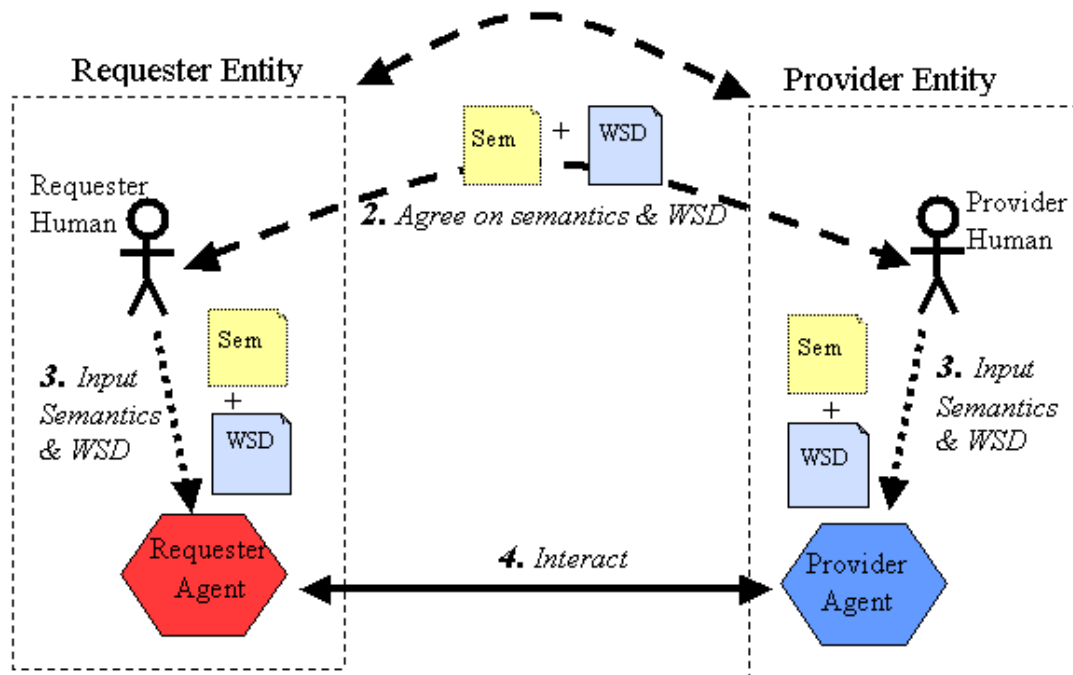


FIGURA 8 FLUJO WEB SERVICE (W3C, 2004)

2.4.5.1. Servicios REST

Un servicio web es un sistema de software diseñado para dar soporte a interacciones máquina a máquina sobre una red. (W3C, 2004). Contiene una interfaz descrita en un formato que puede ser procesado por una máquina. Otros sistemas interactúan con estos servicios de acuerdo con lo especificado por la interfaz en forma de intercambio de mensajes. Por lo general, estos mensajes son enviados a través de HTTP o XML.

Cuando se habla de operaciones **sin estado** (stateless en adelante) quiere decir que cada solicitud hecha por el cliente debe contener toda la información necesaria para entender la solicitud, y no puede tomar ventaja de ningún contexto almacenado en el servidor (Fielding, 2000). De igual manera, el servidor no conserva información sobre la solicitud realizada. Un cliente realiza una petición y no espera una confirmación de recepción.

2.4.5.2. Métodos HTTP

Las operaciones que se pueden realizar con un servicio Web vienen definidas por los métodos HTTP. Estas son palabras reservadas enviadas en cada solicitud hecha por un cliente. De acuerdo con la W3C, el organismo internacional encargado de los estándares para la WWW es importante que tomar en cuenta que el software representa la interacción de los usuarios en internet y es necesario que un usuario sea consciente del significado de sus acciones en la web (W3C, 1999). Es decir, un usuario debe ser capaz de reconocer qué tipo de acción está realizando y la respuesta del software debe

ser realizar exactamente la acción requerida. De acuerdo con este principio existen dos tipos de métodos. Los métodos seguros (Safe Methods) y los métodos idempotentes (Idempotent Methods).

Los métodos seguros son aquellos métodos que no representan una acción insegura. En este caso, hace referencia a los métodos dedicados a recuperar información. Los métodos **GET** y **HEAD** se consideran métodos seguros ya su principal función es y siempre debe ser recuperar información.

Los métodos idempotentes son aquellos métodos que cumplen con la propiedad de que, a excepción de los errores o expiraciones, los resultados de ejecutar muchas veces la misma solicitud son idénticos a los obtenidos al realizar una sola solicitud. Los métodos GET, HEAD, PUT, DELETE, OPTIONS y TRACE son métodos idempotentes.

GET es un método que recupera cualquier información, en la forma de una entidad, identificada en la URL de la petición. Existen modificadores que pueden ser agregados a una solicitud GET que alteran la semántica de esta:

- Si el mensaje de la solicitud contiene un modificador (If-Modified-Since, If-Unmodified-Since, If-Match, If-None-Match, or If-Range), se convierte en un GET condicional. En este caso, la respuesta será transferida sólo bajo las condiciones impuestas por el modificador. Su objetivo es reducir el uso de la red permitiendo que las entidades en caché se refresquen sin requerir múltiples solicitudes
- Si el mensaje de la solicitud contiene un encabezado de rango (Range), se convierte en un GET parcial. En este caso, la respuesta sólo contendrá la parte del mensaje solicitado. Su objetivo es reducir el uso de la red al permitir a recursos parcialmente recuperados ser completados sin tener que transmitir data que ya contiene el cliente nuevamente.

HEAD es un método idéntico a GET con la excepción de que el servidor no debe devolver un cuerpo del mensaje en la respuesta. Únicamente deben ser transmitidos los metadatos contenidos en los encabezados HTTP. Este método puede ser utilizado para recuperar los metadatos de una entidad implícita en una solicitud sin la necesidad de enviar el cuerpo de la entidad como tal.

OPTIONS es un método para solicitar información sobre las opciones de comunicación disponibles. Permite al cliente saber de qué manera puede comunicarse con el servidor sin necesidad de hacer una solicitud de una entidad. Este método es usando, por ejemplo, en aplicaciones móviles y otros sistemas para identificar si el intercambio de recursos de distintos orígenes (CORS) está habilitado. Un cliente enviará una consulta con OPTIONS antes de enviar una solicitud de cualquier otro tipo. Si el servidor no acepta solicitudes de orígenes distintos, entonces las subsecuentes solicitudes no serán realizadas.

POST es el método utilizado para solicitar que el servidor acepte una entidad encapsulada en la solicitud, es decir, preservar nuevo contenido o procesar datos. El mejor ejemplo de un POST es el envío de un formulario de datos. Un usuario llena los

campos con la información solicitada. Este formulario representa los campos de una entidad. Cuando la solicitud es enviada, se incluye en la misma dicha entidad para ser almacenada.

PUT es un método parecido a POST. La solicitud incluye una entidad. La principal diferencia radica en el objetivo del método y en el tipo de URL asociada. En el caso de un POST, la URL describe la acción que se quiere realizar e indica al servidor que un nuevo elemento debe ser creado, bajo un nuevo identificador. En el caso del PUT, la URL incluye un identificador, de manera tal que el método lo que le indica al servidor es que la entidad que se está enviando debe ser guardada bajo ese identificador. Si el identificador incluido en la URL existe, entonces la entidad contenida en la solicitud debe ser considerada una versión modificada de la entidad existente. En caso contrario, se considera como una nueva entidad y el método se comporta como un POST

DELETE es el método para solicitar al servidor la eliminación de un recurso identificado por la URL. El cliente no puede estar seguro de que la operación ha sido ejecutada, aun cuando el código de estado retornado lo indica de esta manera (W3C, 1999).

Finalmente, existen otros dos métodos HTTP. TRACE y CONNECT. El primero es un método utilizado al depurar el código ya que su función es devolver al usuario la solicitud enviada. El receptor debe devolver exactamente el mismo mensaje recibido de vuelta al cliente. El segundo, es un método utilizado con un proxy que puede convertirse dinámicamente en un túnel (SSL, por ejemplo).

2.4.5.3. Objetivo de la arquitectura REST para servicios Web

Los principales objetivos de REST son:

- Permitir la escalabilidad de la interacción entre los componentes
- Ofrecer una interfaz que permita a cualquier cliente interactuar con cualquier servidor a través de HTTP sin ningún tipo de implementación acoplada.
- Ofrecer una interfaz autónoma que no requiera supervisión y permita la comunicación entre clientes y servidores durante largos períodos de tiempo.

REST logra satisfacer estos objetivos aplicando tres restricciones de acuerdo con lo establecido en (Navarro, 2007)

- **Identificación de recursos y manipulación de ellos a través de representaciones:** Identificación de recursos y manipulación de ellos a través de representaciones. Esto se consigue mediante el uso de URIs. HTTP es un protocolo centrado en URIs. Los recursos son los objetos lógicos a los que se le envían mensajes. Los recursos no pueden ser directamente accedidos o modificados.
- **Mensajes autodescriptivos:** REST dicta que los mensajes HTTP deberían ser tan descriptivos como sea posible. Esto hace posible que los intermediarios interpreten los mensajes y ejecuten servicios en nombre del usuario. Uno de los modos que HTTP logra esto es por medio del uso de varios métodos estándares, muchos encabezamientos y un mecanismo de direccionamiento. Por ejemplo, las cachés.

- **Hipermedia como un mecanismo del estado de la aplicación:** El estado actual de una aplicación Web debería ser capturada en uno o más documentos de hipertexto, residiendo tanto en el cliente como en el servidor. El servidor conoce sobre el estado de sus recursos, aunque no intenta seguirles la pista a las sesiones individuales de los clientes. Esta es la misión del navegador, él sabe cómo navegar de recurso a recurso, recogiendo información que el necesita o cambiar el estado que el necesita cambiar.

2.4.6. Módulos Python

Si el intérprete Python se cierra y se inicializa de nuevo, las definiciones que se habían creado (funciones y variables) se pierden. Por lo tanto, si se quiere escribir un programa más largo y complejo, es mejor usar un editor de texto para preparar la entrada para el intérprete y así ejecutarlo con ese archivo de entrada. Este proceso se conoce como crear un “script”. Mientras más largo sea el programa, más se recomienda separar el código en diferentes archivos para facilitar el mantenimiento. Para apoyar esto, Python tiene una manera de colocar las definiciones en un archivo y usarlas en el script o en una instancia interactiva del intérprete. Este archivo es conocido como “módulo” (o librería); las definiciones pertenecientes a un módulo pueden ser “importadas” hacia otros módulos o hacia el código principal haciendo uso de la instrucción **import**.

Un módulo es un archivo que contiene definiciones y sentencias en Python, el nombre de este archivo debe ser el nombre del módulo seguido por el sufijo **.py** (Python Software Foundation, 2019), para la realización de este proyecto se usaron los siguientes módulos Python:

2.4.6.1. BeautifulSoup

Librería que facilita la recuperación de información de páginas web. Se asienta sobre un analizador (en inglés “parser”) XML o HTML, proveyendo modismos pitónicos (de Python) para iterar, buscar y modificar el árbol DOM del sitio. (Python Software Foundation, 2019)

2.4.6.2. DiffLib

Este módulo provee clases y funciones para comparación de secuencias. Puede ser usado para comparar distintos archivos y producir diferentes tipos de información en formatos variados, incluyendo (pero no limitándose a) HTML. (Python Software Foundation, 2019)

2.4.6.3. Math

Este módulo otorga acceso a las funciones matemáticas definidas por el estándar C. No puede ser usado con números complejos, esta distinción se hace debido a que la mayoría de los usuarios no quieren aprender las matemáticas necesarias para comprender los números complejos. Recibir una excepción en lugar del resultado complejo permite detección más ágil del número complejo inesperado que se usa como parámetro, para que el programador determine cómo y por qué fue generado en primer lugar. (Python Software Foundation, 2019)

2.4.6.4. Statistics

Este módulo provee funciones para la realización de cálculos estadísticos de data numérica con valores reales. (Python Software Foundation, 2019)

2.4.6.5. NumPy

NumPy es el módulo fundamental para computación científica en Python, contiene entre otras cosas:

- Un poderoso objeto para manipular arreglos de N dimensiones.
- Funciones sofisticadas de transmisión.
- Herramientas para integrar código en C/C++ y Fortran.
- Capacidad de manejar álgebra lineal, transformadas de Fourier y números aleatorios.

Además de sus obvios usos científicos, NumPy también puede ser usado como un contenedor multidimensional eficiente de data genérica. Tipos arbitrarios de datos pueden ser definidos, lo que permite a NumPy integrarse de manera rápida y perfecta con una amplia variedad de bases de datos. Está licenciado bajo la licencia BSD, permitiendo su reutilización con pocas restricciones. (NumPy, 2019)

2.4.6.6. Matplotlib

Es un módulo de graficación en dos dimensiones (2D) que produce figuras de calidad en una gran variedad de formatos y ambientes interactivos entre plataformas. Puede ser usada tanto en scripts de Python como en consolas IPython, servidores de aplicaciones web y cuatro (4) kits de herramientas de interfaz gráfica. Matplotlib trata de lograr que tanto las cosas fáciles como las difíciles sean posibles de realizar. Se pueden generar tramas, histogramas, gráficos de barras, diagramas de error, etc., con sólo unas líneas de código. (Matplotlib, 2019)

2.4.6.7. Selenium

Conjunto de herramientas usadas alrededor de diferentes plataformas con el fin de automatizar los clientes web. Principalmente se usa para automatización de pruebas de aplicaciones web, pero no se limita solamente a esa tarea. Actividades de administración basadas en web también pueden ser automatizadas. No sólo tiene el soporte de los más grandes vendedores de browsers, sino que también es la tecnología núcleo en incontables herramientas de automatización, APIs y entornos de trabajo. (SeleniumHQ, 2019)

2.4.7. Detección de cambios en páginas web

De acuerdo con el diccionario de Oxford, “detección” se puede definir como:

La acción o el proceso de identificar la presencia o existencia de algo.
(Oxford, 2019)

De la misma manera, el diccionario de Oxford también define “cambios” de la siguiente manera:

Un acto o proceso a través del cual algo se convierte en otro elemento distinto. (Oxford Dictionary, 2019)

Usando estas definiciones como base, la detección de cambios puede ser explicada como la identificación de las transformaciones realizadas por diferentes elementos en un contexto dado. Esto es aplicable a los sitios web, sobre los cuales podemos identificar dichos cambios mediante una diversidad de técnicas que se dividen en dos grandes grupos, las pertenecientes a la detección basadas en el texto obtenido del código de estos y las técnicas pertenecientes a la segmentación de los sitios web de acuerdo con la ubicación espacial de las alteraciones.

2.4.7.1. Detección de Cambios basada en Texto

El enfoque basado en texto no toma en cuenta la estructura del sitio web visualizada luego de que éste sea renderizado por el navegador, sino el contenido escrito en el código de este, el cual trata como un archivo de texto plano. Los algoritmos pertenecientes a esta aproximación sólo observan al contenido textual y analizan ciertas características como densidad de texto o densidad de enlaces de ciertas partes de una página. Estas técnicas se basan en resultados de lingüísticas cuantitativas que indican que, estadísticamente, bloques de texto con características similares probablemente pertenezcan al mismo grupo y por lo tanto pueden ser unidos en un solo bloque. La tolerancia óptima depende del nivel de granularidad y debe ser determinada experimentalmente.

2.4.7.2. Detección por Segmentación Web

La segmentación web es el proceso de tomar una página web y particionarla en bloques llamados “bloques semánticos” (o “segmentos”) que definimos como:

Un fragmento contiguo HTML que es renderizado como un bloque gráficamente consistente y cuyo contenido está unido semánticamente.
(Kreuzer, Hage, & Feelders, 2014)

Los bloques semánticos pueden, en principio, estar profundamente anidados, aunque en la práctica, raramente se consideran los anidamientos mayores a dos o tres niveles. Hay diferentes algoritmos que se usan para la segmentación web, y son demasiados como para considerarlos todos, así que se optó por observar los que más se aproximan a aquellos que trabajarían mejor la segmentación con sitios web actuales. Los algoritmos por analizar son los siguientes:

2.4.7.3. Segmentación de Páginas (PageSegmenter)

Una aproximación DOM se basa en observar el árbol DOM creado al analizar el código HTML buscando señales de cómo segmentar el sitio web, sin tomar en cuenta las

propiedades añadidas por archivos externos, como por ejemplo archivos CSS, ya que éstos se consideran es en el árbol renderizado. La idea es que la estructura HTML debería reflejar las semánticas de la página web. La calidad de este tipo de aproximaciones, por ende, dependerá de que tan cierto sea este caso. Para realizar este tipo de segmentación se debe confiar en la detección de patrones recurrentes, como listas y tablas; así como en heurísticas, como el uso de etiquetas de cabecera siendo usadas como separadores o enlaces siendo parte del texto circundante. Este enfoque es sencillo de implementar y eficiente al ejecutar, ya que sólo debe analizar el HTML en vez de cargar la página. Las complicaciones que pueden presentarse es que hay diferentes maneras de construir una estructura HTML para el mismo contenido; adicionalmente, el estilo y la información de estructurado son ignorados por completo, tampoco se trabajará de inmediato cuando hay presencia de código JavaScript a menos que el DOM sea serializado a priori.

El algoritmo PageSegmenter es un ejemplo claro del enfoque mencionado cuya idea principal es que los caminos raíz-hoja de nodos extremos (hojas) pueden ser analizados en búsqueda de similitudes para encontrar nodos que son posibles que pertenezcan al mismo bloque semántico. Un ejemplo de tal camino sería **/html/body/p/ul/li**. Ahora, si múltiples nodos hermanos pertenecen a un camino muy similar es seguro asumir que semánticamente deben estar contenidos juntos, ya que estructuralmente son parte de la misma lista. (Kreuzer, Hage, & Feelders, 2014)

Algoritmo de Segmentación de Página basado en Visión (VIPS)

Las aproximaciones visuales son las que más se parecen a como un humano segmentaría una página de manera natural, es decir, operan sobre el sitio web ya renderizado y como se vería en el browser. Por lo tanto, los algoritmos que pertenecen a este tipo de enfoque son los que tienen más información disponible al momento de realizar su análisis, pero también son los más costosos computacionalmente debido al proceso de renderización mismo. Usualmente dividen el sitio web en separadores tales como líneas, espacios en blanco, imágenes y contenido; luego de esto construyen una estructura a partir de toda esta información. También pueden considerar características únicamente visuales como color de fondo, estilo, diseño, tamaño de fuente, y tipo y ubicación en la página. Para realizar el renderizado necesitamos acceso al motor de un browser, lo que complica la implementación del algoritmo, además que requiere elementos externos para funcionar de manera correcta (como archivos CSS o imágenes). (Kreuzer, Hage, & Feelders, 2014)

De esta manera se escogió analizar el algoritmo más popular de los que son derivados de este enfoque, llamado VIPS y que, como su nombre lo indica, está basado en la renderización completa de un sitio web. Analiza el DOM después de que toda la información de estilo proveniente del CSS ha sido aplicada y luego de que se ejecuten todos los archivos JavaScript que son parte del sitio y que potencialmente modificaron el árbol. Está estrictamente integrado con el motor de renderizado del browser ya que necesita consultar información tal como las dimensiones en pantalla de un elemento determinado. Por lo tanto, es necesario que se determine el tamaño del visualizador previo a la ejecución del algoritmo. En concreto, VIPS construye una estructura de contenido basada en lo que se visualiza, independiente del documento HTML

subyacente, decidiéndose durante un recorrido desde el tope hacia el piso si algo representa un bloque visual o si debiese ser subdividido aún más, usando heurísticas tales como “si un subárbol contiene una etiqueta separadora `<hr>`, realiza la subdivisión”.

2.4.7.4. Fusión de Bloques (BlockFusion)

El enfoque basado en texto difiere de los dos anteriores en que no toma en cuenta la estructura del HTML en lo absoluto. Los algoritmos pertenecientes a esta aproximación sólo observan al contenido textual y analizan ciertas características como densidad de texto o densidad de enlaces de ciertas partes de una página. Estas técnicas se basan en resultados de lingüísticas cuantitativas que indican que, estadísticamente, bloques de texto con características similares probablemente pertenezcan al mismo grupo y por lo tanto pueden ser unidos en un solo bloque. La tolerancia óptima depende del nivel de granularidad y debe ser determinada experimentalmente. Este tipo de algoritmos suelen ser rápidos y sencillos de implementar ya que trabajan de manera independiente del DOM, pero, al igual que el enfoque basado en DOM, no trabajará con JavaScript (a menos que se serialice primero el DOM), no toma en cuenta características visuales o estructurales, y la extracción de los subbloques requiere de cambios locales al umbral de tolerancia de densidad de texto (ya que no podemos emplear la estructura del documento). (Kreuzer, Hage, & Feelders, 2014)

El algoritmo BlockFusion se basa en que el token de densidad de un elemento puede ser una heurística valiosa para segmentar documentos de texto. Este token puede ser calculado simplemente tomando el número de palabras en el texto y dividido entre el número de líneas, donde cada línea tiene un máximo de 80 caracteres. Un documento HTML es primeramente preprocesado en una lista de bloques atómicos de texto, separándolos con algún tipo de indicador, los cuales son las etiquetas HTML a excepción de la etiqueta `<a>`. Por cada bloque se calcula el token de densidad. Una estrategia de mezcla es usada entonces para unir bloques en bloques progresivamente mayores, siempre que la diferencia de los valores del token de dos bloques adyacentes esté dentro de límites ya establecidos. Se repite el proceso de fusión hasta que ya no queden bloques restantes que puedan ser unidos. Debido a este diseño, BlockFusion no soporta múltiples niveles de bloques por defecto, pero si extendemos el algoritmo de manera que localmente introduzcamos un segundo valor límite más pequeño, y luego llamamos al algoritmo en cada uno de los bloques ya unidos, podemos conseguir una jerarquía de dos niveles. (Kreuzer, Hage, & Feelders, 2014)

2.4.7.5. Terreno Web (WebTerrain)

Este algoritmo no pertenece a ninguno de los enfoques tratados previamente, fue desarrollado por Robert Kreuzer, Hage Jurriaan y Ad Feelders como su contribución al problema de la segmentación. La idea principal era determinar si era posible combinar las diferentes aproximaciones realizadas por los algoritmos analizados para mejorar los resultados obtenidos mediante el uso de las características positivas de los mismos. WebTerrain está basado en una heurística que ellos consideraron novedosa, la cual además le dio nombre al algoritmo y cuya inspiración provino del navegador Firefox, el cual contiene una funcionalidad poco conocida que le permite al usuario observar una renderización en tres dimensiones (3D) de cualquier sitio web que sea cargado. El

resultado es similar a un mapa geográfico de un terreno cualquiera. Esta funcionalidad agrega un valor adicional de profundidad a cada elemento visible, el cual simplemente es el valor de profundidad del elemento inspeccionado en el árbol. Experimentos realizados demostraron que el perfil de elevación corresponde de manera satisfactoria con lo que consideraron los bloques semánticos de un sitio web. La heurística a la que llegaron mediante esa observación tiene la interesante propiedad de que combina un enfoque de estructura plana con un enfoque basado en la renderización, ya que no sólo toma en cuenta el árbol DOM sino también la visibilidad y las dimensiones de cada elemento. (Kreuzer, Hage, & Feelders, 2014)

Capítulo 3 MARCO PROCEDIMENTAL

Uno de los pasos necesarios para llevar a buen término la realización de un proyecto de desarrollo de software se debe escoger un método y establecer un plan de trabajo en conjunto con el fin de regular las actividades diarias, establecer metas y definir lapsos de tiempo para los cuales éstas deben ser completadas, en otras palabras, para definir una estrategia de trabajo que permita completar las tareas establecidas de manera eficiente. A continuación, se procederá a realizar una explicación de algunas metodologías establecidas, así como a realizar una comparación entre ellas con el fin de escoger la que sea más acorde al proyecto establecido, de no haber alguna que se adapte completamente al trabajo, se tomará la más eficaz y se procederá a hacer las modificaciones que se consideren pertinentes.

3.1. Metodologías de Desarrollo de Software

Una metodología de desarrollo de software puede ser definido como un conjunto estandarizado de conceptos, prácticas y criterios con la finalidad de estructurar, planear y controlar el proceso de desarrollo de software de calidad, a menudo son vinculados a algún tipo de organización, la cual promueve el uso y hace refinamientos a la metodología.

3.1.1. Metodologías Tradicionales

Se empezaron a desarrollar en la década de 1960 para desarrollar a una enorme escala funcional de sistemas de negocios, sus procesos son muy planificados y metódicos, es necesarios describir herramientas de Análisis y Diseño Orientados a Objetos. En la actualidad son llamadas Metodologías “Tradicionales” o “Pesadas” a aquellas que pongan un mayor énfasis en la planificación y control del proyecto, esto es, una especificación muy precisa de los requerimientos y el modelado del sistema. Entre ellas podemos mencionar:

RUP (del inglés Rational Unified Process):

Desarrollado por la empresa Rational Software es parte de la metodología más usada para el análisis, diseño, documentación e implementación de sistemas orientados a objetos en conjunto con el Lenguaje Unificado de Modelado (UML por sus siglas en inglés). Más que un conjunto de pasos establecidos es un conjunto de ideas y metodologías adaptables a cada organización. Está basado en seis (6) principios clave:

- Adaptar el proceso.
- Equilibrar prioridades.
- Demostrar valor iterativamente.
- Colaboración entre equipos.
- Enfoque en la calidad.

- Elevar el nivel de abstracción.

De igual manera establece un conjunto de fases que determinan el ciclo de vida de la metodología (inicio, elaboración, desarrollo y transición) y un conjunto de artefactos de software utilizados para la mejor comprensión del producto en cada una de las fases (como los diagramas de secuencia, estado o clases y los modelos de dominio o, de ser requerido, entidad-relación). (Rational Software , 2001)

3.1.1.1. MSF (viene de Microsoft Solution Framework):

Desarrollado por Microsoft, es definido como

“un enfoque personalizable para entregar con éxito soluciones tecnológicas de manera más rápida, con menos recursos humanos y menos riesgos, pero con resultados de más calidad.” (Microsoft Corporation, 2006)

MSF está centrado en:

- Alinear los principios del negocio y la tecnología.
- Establecer objetivos, roles y responsabilidades de manera clara y específica.
- Implementar un proceso iterativo regulado por puntos de control.
- Gestionar los riesgos de manera activa.
- Responder con eficacia ante los cambios.

Al igual que RUP, establece un conjunto de fases divididas en lo que llaman “*Pistas de ejecución*” superpuestas entre sí y reguladas por una persistente que las abarca todas llamada “*Pista de Gobernanza*”.

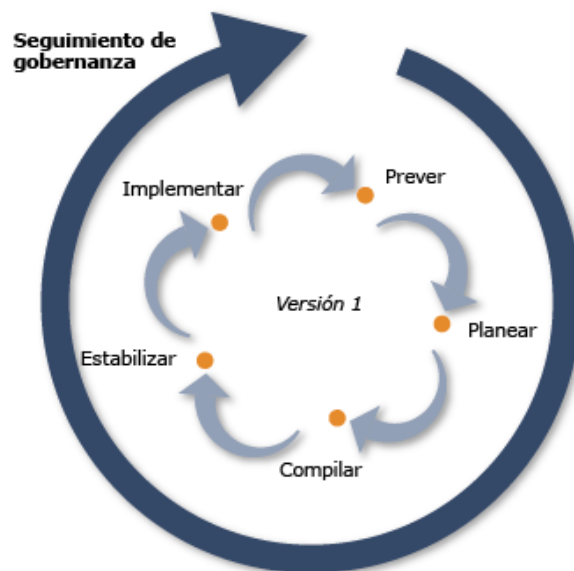


FIGURA 9 FASES DEL MSF (Microsoft Corporation, 2006)

La Pista de Gobernanza se centra en equilibrar el uso de los recursos del proyecto, así como la mejora continua de cada uno de los procesos, siempre respetando las restricciones que hayan sido definidas por la organización; tiene tres (3) objetivos principales, los cuales son:

- Guiar las actividades de ejecución para entregar una solución con resultados repetibles y confiables.
- Optimizar y mejorar continuamente el rendimiento y la capacidad del equipo, la calidad de las soluciones y la mejora de los procesos.
- Obtener aprobación de los Usuarios, las Operaciones y el Cliente.

Las pistas de ejecución son las que ayudan al equipo a llegar a un acuerdo de alto nivel sobre el futuro del proyecto y crear opciones de enfoque para hacerlo realidad (pista de visión); evaluar dichas opciones y planear la seleccionada (pista de planeación); compilar la solución desarrollada (pista de compilación); asegurarse que la solución es entregada de acuerdo con lo planeado (pista de estabilización); y realizar la implementación de la solución (pista de implementación).

3.1.2. Metodologías Ágiles

Con el uso de las metodologías tradicionales, no se tomaba en cuenta el factor humano presente en cada actividad del desarrollo de software, así como tampoco se hacía énfasis en los problemas y cambios que pudiesen surgir a lo largo del proyecto, por este motivo en 2001, durante una reunión de 17 expertos celebrada en Utah (EE. UU.), nace el término “ágil” aplicado al desarrollo de software. El objetivo principal de ésta fue esbozar un nuevo grupo de valores y principios que permitieran una respuesta rápida a los problemas surgidos durante el desarrollo. Tras esta reunión fue creada The Agile Alliance, una organización sin ánimo de lucro dedicada a promover los conceptos relacionados con el desarrollo ágil del software, su punto de partida fue el Manifiesto Ágil, un documento que resume esta filosofía y que enumera sus principios básicos. (Beck, y otros, 2001)

La Tabla 1 mostrada a continuación, ofrece una comparación detallada entre las características principales de las metodologías tradicionales y las metodologías ágiles:

Tabla 1 Comparativo entre enfoques metodológicos

Metodologías Ágiles	Metodologías Tradicionales
Pocos artefactos de modelado son prescindibles y desechables.	El modelado es esencial, hay gran presencia de artefactos de software.
Pocos roles, genéricos y flexibles.	Gran cantidad de roles definidos que se destacan por su especificidad.

El contrato debe ser flexible.	El contrato es fijado con anterioridad y debe ser seguido al pie de la letra.
El cliente es parte del equipo de desarrollo.	El cliente no es parte del equipo de trabajo, interactúa con éste mediante reuniones agendadas.
Está orientada a equipos con pocos integrantes y entregas frecuentes y pequeñas.	Se pueden aplicar a proyectos de cualquier tamaño, pero suelen ser usadas para proyectos de gran tamaño con equipos muy dispersos. Entregas grandes y poco frecuentes.
La arquitectura se define y mejora a lo largo del proyecto.	La arquitectura debe ser definida en las primeras fases del proyecto.
Se hace énfasis en el aspecto humano.	El énfasis está en las fases y roles del proyecto.
Se basan en heurísticas provenientes de la práctica de producción de código	Se basa en normas provenientes de estándares seguidos por el entorno de desarrollo.
Los cambios en el proyecto son esperados y deseables.	Los cambios de gran impacto en el proyecto no son esperados.

3.1.2.1. Extreme Programming (XP)

Es el método de desarrollo ágil más popular debido a varios factores, entre los que podemos mencionar la presencia del cliente dentro del equipo de trabajo, la reutilización de código y el desarrollo de pruebas como parte de sus valores fundamentales y la simplicidad de código como la característica principal. Los defensores de XP consideran que los cambios de requisitos sobre la marcha no sólo son inevitables, sino deseables, así que deben estar preparados para realizar las modificaciones necesarias durante el proceso de desarrollo y por lo tanto ser capaz de adaptarse a éstos es una cualidad indispensable para un desarrollador. De acuerdo con la segunda edición del libro mencionado, XP tiene cinco (5) valores fundamentales: (Beck K. , 1999)

- Simplicidad.
- Comunicación.

- Retroalimentación.
- Coraje (o valentía).
- Respeto.

Debido a la gran agilización requerida para el proyecto y el tamaño tan pequeño del equipo de programación, XP fue la metodología escogida para ser usada como guía durante el desarrollo de la aplicación, por supuesto fue necesaria adaptarla para el trabajo individual, a pesar de que XP fue desarrollada con la idea de una pareja de programadores, lo que no ocurrió en este caso.

3.1.2.1.1. Fases del Desarrollo

- **Planeación:** es planteada como una comunicación constante entre las partes involucradas, incluye a todo el equipo de trabajo. Inicia recopilando historias de usuario, que sustituirán a los casos de uso tradicionales. Luego de ser definidas las historias, los programadores evalúan el tiempo de desarrollo de cada una. Los conceptos básicos manejados en esta fase son:
 - **Historias de Usuarios:** escritas por el cliente en lenguaje natural, son descripciones de las tareas a realizar.
 - **Plan de Entregas:** establece qué historias serán agrupadas para cada entrega y el orden de ellas.
 - **Plan de Iteraciones:** cada historia seleccionada en el plan es desarrollada y probada en el ciclo de iteración correspondiente de acuerdo con el orden ya decidido.
 - **Reuniones de seguimiento:** mantener la comunicación dentro del equipo con el objetivo de compartir los problemas que se generaron y sus posibles soluciones.
- **Diseño:** la metodología XP hace especial énfasis en que los diseños deben de ser los más sencillos posibles.
- **Codificación:** esta es la fase en la que se realiza el proceso de creación del software, sus conceptos principales los podemos resumir de la siguiente manera:
 - **Disponibilidad del cliente:** el cliente debe estar trabajando en conjunto con el equipo de programadores en todo momento, proporcionando y discutiendo con los programadores los detalles sobre las historias de usuarios que no surgieron durante la fase de planeación.
 - **Uso de Estándares:** es promovido el uso de estándares durante la programación para facilitar el entendimiento del código de parte de todo el equipo de trabajo.
 - **Ritmo Sostenido:** se debe llevar un ritmo constante y razonable de trabajo, de manera que no se sobrecargue al equipo

- **Pruebas:** es la última fase, determinan con su éxito si la historia del usuario ha sido correctamente implementada. Los conceptos principales que se manejan aquí son:
 - **Pruebas Unitarias:** cada módulo desarrollado debe ser probado antes de su entrega, como se mencionó con anterioridad, es recomendable que las pruebas sean escritas antes del desarrollo del código.
 - **Detección y corrección de errores:** de ser encontrado cualquier tipo de error, debe de corregirse de manera inmediata y se deben establecer medidas para que no vuelva a ocurrir ningún error parecido. Nuevas pruebas deben generarse para realizar la verificación de la solución correcta del error.
 - **Pruebas de Aceptación:** creadas en base a las historias de los usuarios durante cada iteración del desarrollo, son provistas por el cliente, quien especificará uno o más escenarios con el fin de comprobar la correcta implementación de la historia. En caso de fallar varias pruebas, el cliente se encargará de especificar el orden de resolución a seguir, al momento de pasar de manera correcta todas las pruebas de aceptación, la historia de usuario se considerará terminada

3.1.2.1.2. Roles

A pesar de que XP especifica prácticas particulares para que el equipo siga durante el trabajo, realmente no establece roles específicos, éstos varían de acuerdo con la fuente a la que uno se adhiera, sin embargo, se encuentra que los proyectos desarrollados en XP usualmente contienen roles definidos de la siguiente manera: (Agile Alliance, 2019)

- **Cliente:** es el rol responsable de la toma de decisiones del negocio en lo relacionado al proyecto incluyendo, pero no limitándose a
 - ¿Qué debe hacer el sistema (qué características tiene y qué logra cada una)?
 - ¿Cómo se conoce que el sistema está listo (cuáles son los criterios de aceptación)?
 - ¿Cuánto se puede gastar (presupuesto disponible)?
 - ¿Qué se debe hacer luego (en qué orden se realiza cada tarea)?

Se espera que el cliente esté activamente integrado al proyecto, idealmente se convertirá en parte del equipo. Se asume que el cliente es una sola persona, sin embargo, la experiencia ha demostrado que una persona no puede proveer toda la información relacionada al negocio de manera adecuada, por lo que esto puede variar. (Agile Alliance, 2019)

Para el presente Trabajo Especial de Grado, la tutora asumió el rol de cliente.

- **Desarrollador:** debido a que XP no tiene necesidad de roles, cada miembro del equipo es asignado este rol a excepción del cliente, el “tracker” y el coach. Los desarrolladores se encargan de realizar las historias descritas por el cliente. (Agile Alliance, 2019)

Este fue el rol asumido por el bachiller durante la realización del Trabajo Especial de Grado.

- **Tracker:** también llamado “Encargado de Seguimiento”, es un rol que no siempre es necesario, incluso es usual que uno de los desarrolladores del equipo sea el que cumpla con las tareas asignadas a éste. Su propósito principal es (como su nombre lo dice), llevar el seguimiento de las métricas que el equipo considere relevantes e identificar áreas que necesiten mejoría. (Agile Alliance, 2019)

Las tareas asignadas a este rol fueron asumidas por la tutora durante el desarrollo del presente trabajo.

- **Coach:** si el equipo empezó a aplicar XP recientemente, es útil tener alguien asignado a este rol, usualmente es un consultor externo al equipo que ha usado la metodología con anterioridad y su trabajo es ayudar y enseñar al resto de los miembros sobre las prácticas de XP. Su valor principal es que reduce la cantidad de errores en el desarrollo gracias al conocimiento que obtuvo durante sus experiencias previas. (Agile Alliance, 2019)

Este rol fue considerado innecesario y, por lo tanto, no se tomó en cuenta para este T.E.G.

3.1.2.1.3. Ciclo de Vida

Para describir XP en términos usados al hablar de un ciclo de vida lo más apropiado es revisar los conceptos de Ciclo Semanal y Ciclo Trimestral.

Primero, se inicia describiendo los resultados deseados del proyecto mediante el uso de las historias definidas por el Cliente. Al tiempo que se crean estas historias, el equipo de trabajo estima la duración de cada una. Este estimado junto con el beneficio relativo considerado por el cliente se usa para proveer una indicación del valor potencial de cada historia y se toman como base para asignar un orden de prioridades a cada historia.

Si el equipo identifica historias a las cuales no se les puede asignar un estimado, es recomendado añadir un período adicional para la investigación del tópico y dificultades técnicas que puedan presentarse.

Luego, el equipo completo organiza una reunión para crear un plan de lanzamientos que sea considerado razonable por todos los miembros, incluyendo al cliente. Este plan define las historias que se esperan sean entregadas en un lanzamiento particular. El orden de entregas dentro del plan está definido por la importancia que les fue asignada con anterioridad.

Posteriormente el equipo inicia una serie de ciclos semanales, al principio de cada ciclo, el equipo (de nuevo, incluyendo al cliente) se reúne para decidir qué historias serán hechas esa semana y las separan en tareas a ser completadas.

Al final de cada ciclo, los desarrolladores y el cliente revisan el progreso hecho y el cliente decide si el proyecto debe continuar o si suficiente valor ya ha sido entregado. (Agile Alliance, 2019)

3.2. Arquitectura de Software Basada en Componentes

El Desarrollo Basado en Componentes (conocido por sus siglas en inglés CBD) se puede ver de manera simple como una aplicación de la técnica *dividir y vencer* que se usa para reducir la complejidad durante el desarrollo de software, se basa en la idea de definir, diseñar e implementar diferentes elementos de software de manera independiente y luego integrarlos entre sí creando un sistema de componentes débilmente acoplado. Se diferencia de otros métodos de desarrollo en que el análisis y el diseño se realizan dentro del mismo paradigma que la implementación; durante su origen este enfoque se usó como complemento del paradigma de Desarrollo Orientado a Objetos, cuyo objetivo principal es el reuso, por lo tanto, la idea detrás del desarrollo basado en componentes fue mejorar otros aspectos, siendo el más importante la mantenibilidad. El reuso es un objetivo admirable, pero no siempre es fácil de lograr, así que los sistemas creados bajo este paradigma no son focalizados bajo el pensamiento de que serán usados a posteriori en otras partes, sino que cada uno sea fácil de reemplazar, lo que implica que una nueva implementación de éstos podrá ser usadas sin afectar el rendimiento de los demás. (Vignaga & Perovich, 2014)

De acuerdo con Sommerville, los fundamentos de la ingeniería de software basada en componentes (o por sus siglas en inglés CBSE) son los siguientes (Sommerville, Ingeniería del Software, 2005):

- **Componentes independientes**, los cuales son completamente especificados por sus interfaces con una clara separación entre la interfaz y la implementación, para que esta última pueda reemplazarse de manera sencilla y sin cambiar el sistema.
- **Estándares para los componentes**, se incluyen en el modelo de componentes y definen la manera de especificar tanto los componentes como las interfaces con el fin de facilitar la integración. Si los componentes cumplen los estándares, entonces se vuelven independientes del lenguaje de programación por lo que componentes escritos en diferentes lenguajes podrán integrarse sin problemas.
- **El middleware**, que proporciona soporte a la integración de los componentes además de manejar la asignación de recursos, gestión de transacciones, seguridad y concurrencia.
- **Un proceso de desarrollo** que se adapte a la ingeniería de software basada en componentes y que no esté limitada por otro paradigma de desarrollo por suposiciones inherentes a éste.

3.2.1. Herramientas que pertenecen al desarrollo de software basado en componentes

3.2.1.1. Componentes

Un componente es una unidad de software cuya funcionalidad y dependencias están completamente definidas por un conjunto de interfaces públicas. Los componentes pueden ser combinados con otros sin hacer referencia a su implementación y pueden desplegarse como unidades ejecutables. (Sommerville, Ingeniería del Software, 2005)

En la Tabla 2 podemos observar las características esenciales definidas para los componentes en la CBSE.

Tabla 2 Características de los Componentes

Característica	Descripción
Estandarizado	La estandarización significa que un componente usado en uno de los procesos de la CBSE tiene que ajustarse a algún modelo estandarizado. Este modelo debe definir interfaces de componentes, metadata de componentes, documentación, composición y despliegue.
Independiente	Esto se define como que cada componente debe poder ser desplegado y compuesto sin tener que utilizar otros componentes específicos. De requerirse servicios proporcionados de manera externa, deben ser explícitos en una interfaz del tipo “requiere”.
Componible	Todas las interacciones externas deben tener lugar a través de interfaces definidas públicamente. Además, debe proporcionar acceso externo a la información sobre sí mismo, es decir, sus métodos y atributos.
Desplegable	Debe ser capaz de funcionar como una entidad autónoma o sobre una plataforma de componentes que implemente el modelo de componentes. Esto normalmente significa que el componente es binario y que no tiene que compilarse antes de ser desplegado.
Documentado	Cada componente debe estar correctamente documentado para que los usuarios potenciales puedan decidir si satisfacen o no sus necesidades. La sintaxis e, idealmente, la semántica de cada una de las interfaces debe ser especificada.

Fuente: (Sommerville, Ingeniería del Software, 2005)

3.2.1.2. Interfaces de componentes

Cada componente es definido por sus interfaces y, en los casos más generales, puede considerarse que tienen dos interfaces relacionadas, como muestra la Figura 10.



Figura 10 Interfaces de componentes (Sommerville, Ingeniería del Software, 2005)

Una interfaz es, fundamentalmente, el API del componente y define los métodos que pueden ser llamados por un usuario del componente.

3.2.1.3. Modelo de componentes

Un modelo de componentes es una definición de los estándares para la definición, documentación y despliegue de los componentes. Estos estándares son utilizados para asegurar la interoperabilidad de los componentes y por los programadores de los middlewares, quienes proporcionan soporte en forma de infraestructuras de comunicación entre componentes. (Sommerville, Ingeniería del Software, 2005)

La Figura 11 muestra las posibles clasificaciones de elementos dentro de un modelo de componentes, así como los procesos y métodos incluidos en cada uno de ellos.

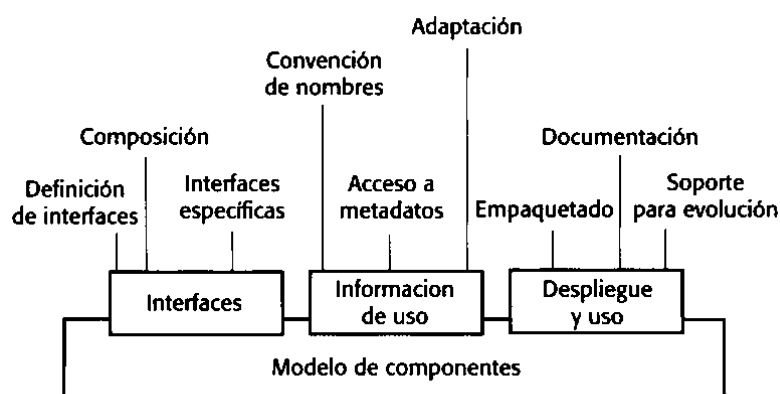


Figura 11 ELEMENTOS DEL MODELO DE COMPONENTES (Sommerville, Ingeniería del Software, 2005)

3.2.1.4. Composición de componentes

Se describe como el proceso de ensamblar componentes para la creación de un sistema, lo que no es una simple tarea; existen diversos tipos de composiciones, entre los que tenemos: (Sommerville, Ingeniería del Software, 2005)

- **Composición Secuencial:** tiene lugar cuando los componentes pertenecientes al diseño general se ejecutan en una secuencia establecida. Es requerido algún código adicional para el enlace entre ellos.
- **Composición Jerárquica:** este caso es dado cuando un componente realiza una llamada de manera directa a los servicios ofrecidos por otro, es decir, cuando un componente depende de otro.

- **Composición Aditiva:** esto tiene lugar cuando las interfaces de dos componentes se unen para crear un componente nuevo. Su interfaz es creada uniendo las interfaces de los componentes que forman parte de ella, eliminando operaciones duplicadas.

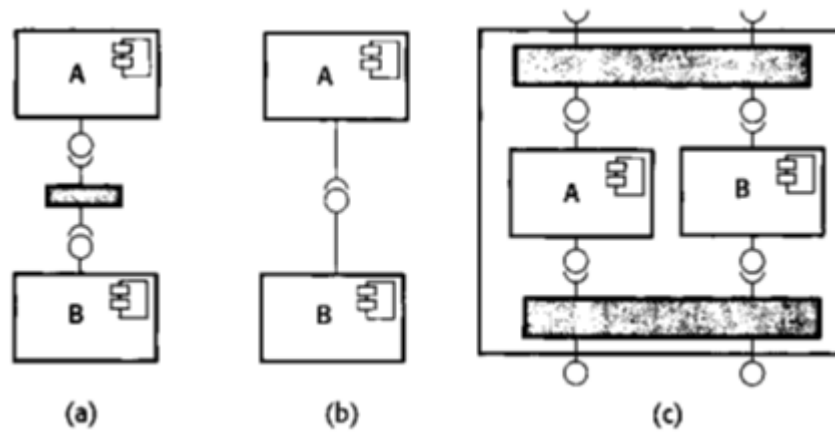


Figura 12 TIPOS DE COMPOSICIÓN DE COMPONENTES (Sommerville, Ingeniería del Software, 2005)

Gracias a que los componentes son independientes entre sí, es posible usar uno o más tipos de composiciones en el mismo sistema sin que eso afecte el rendimiento o la integridad de este.

Capítulo 4 MARCO APLICATIVO

A continuación, se presenta el desarrollo de la aplicación automatizada de Detección de Cambios, siguiendo la metodología de Extreme Programming (XP) como se mencionó en el capítulo anterior, así como también las tecnologías utilizadas en cada actividad del proyecto.

4.1. Definición de Requerimientos

En el presente Trabajo Especial de Grado se agrega una solución a un sistema que actualmente se encuentra en funcionamiento, el cual preserva páginas web venezolanas, pero no detecta cambios de manera automatizada, este proceso actualmente se realiza de manera manual por el administrador del proyecto. El requerimiento principal, por lo tanto, es la creación de una aplicación que permita realizar este proceso, usando una mezcla de técnicas de detección cuantitativas basada en el texto del sitio web a analizar y técnicas cualitativas basadas en la posición espacial del cambio de acuerdo con lo mostrado en el navegador durante la carga inicial.

4.2. Objetivo de la aplicación

El objetivo de la aplicación desarrollada para el presente Trabajo Especial de Grado es continuar con la implementación del Prototipo para el Archivo Web de Venezuela. El Módulo de Adquisición actualmente permite realizar suscripciones de rastreos para cosechar sitios web y almacenarlos de forma indexada. Con la inclusión de la Aplicación de Detección de Cambios será posible determinar si la nueva versión del sitio web posee cambios lo suficientemente significativos como para realizar el almacenamiento de esta de manera automatizada, mejorando el uso de los recursos y proveyendo una base sobre la cual construir un modelo de predicción con el cual sea factible realizar optimizaciones sobre el flujo de trabajo del rastreador.

4.3. Arquitectura de la solución propuesta

La arquitectura propuesta para el Prototipo de Archivo Web de Venezuela, luego de la implementación y añadidura de la aplicación de Detección de Cambios se muestra a continuación en la Figura 13:

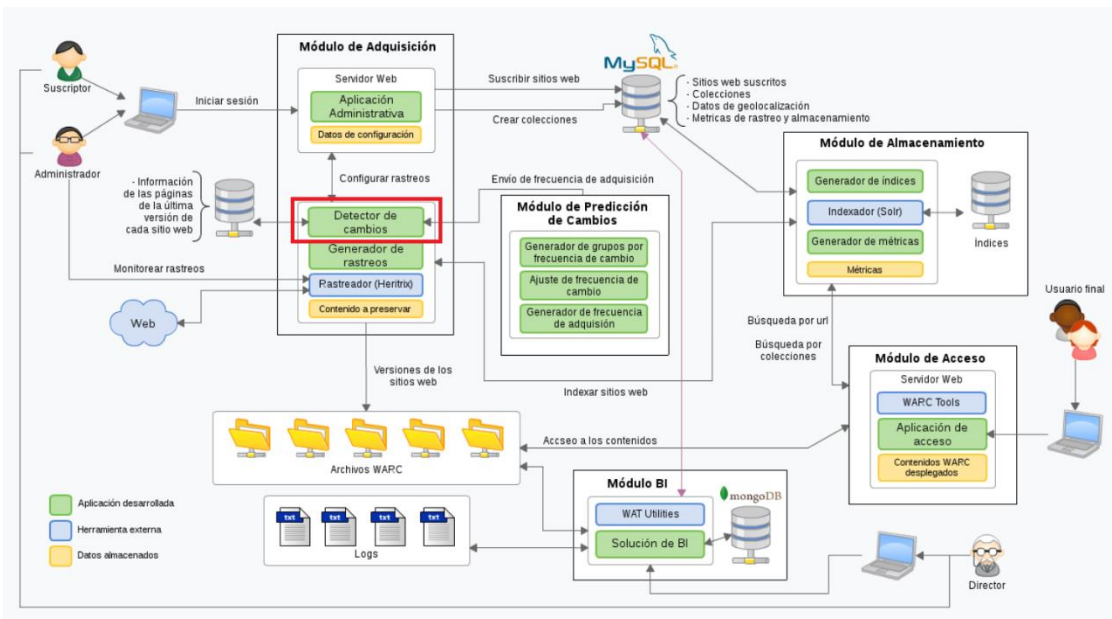


Figura 13 Arquitectura del Prototipo del AWW

Mientras que la arquitectura propuesta para el desarrollo final de la aplicación se muestra a continuación en la Figura 14:

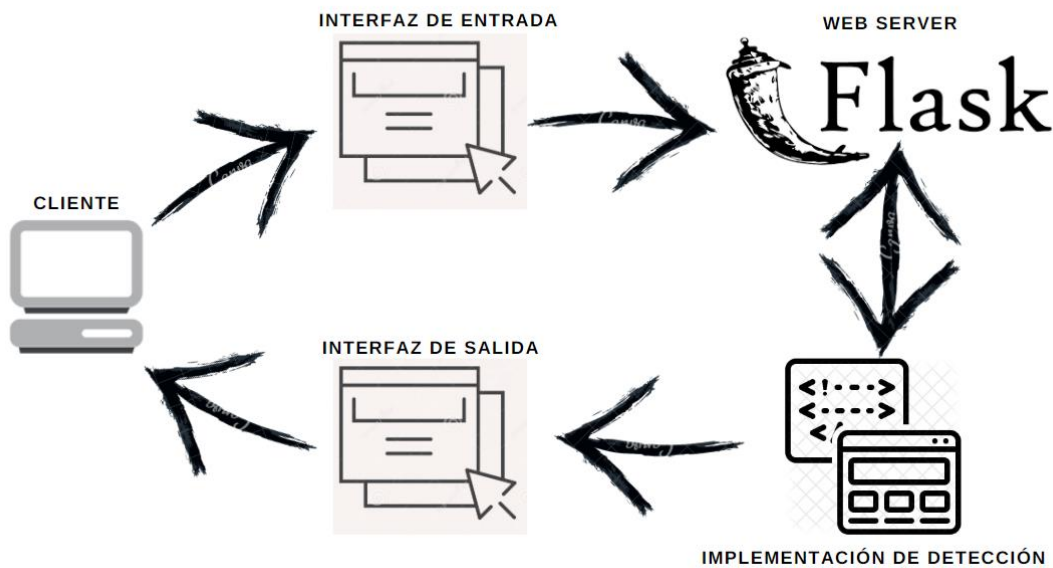


Figura 14 Arquitectura de la aplicación

4.4. Metas de la aplicación

- Proporcionar una interfaz al componente de software conformado por la aplicación de detección de cambios.
- Crear la aplicación de detección de cambios en sitios web.

- Definir la segmentación del código del sitio web a analizar en bloques de texto con el fin de realizar una comparación modular.
- Detectar los cambios de manera cuantitativa mediante el uso de técnicas basadas en texto.
- Detectar los cambios de manera cualitativa mediante el uso de técnicas basadas en posicionamiento espacial de los bloques que han tenido cambios.
- Cuantificar los cambios, tanto de manera cualitativa como cuantitativa para determinar la importancia absoluta de éstos.
- Especificar el rango de importancia absoluta con el fin de determinar si la relevancia del cambio es lo suficientemente significativa como para almacenar la nueva versión.
- Emplear las tecnologías existentes para el desarrollo de la aplicación.
- Permitir la observación de los cambios y su importancia al cliente mediante la creación de una interfaz gráfica.

4.5. Adaptación de la metodología eXtreme Programming (XP) usando una Arquitectura de Software Basada en Componentes

Durante el desarrollo de la aplicación de detección se implementaron los procesos involucrados en la metodología XP, como se mencionó con anterioridad, esta metodología no sólo cree que los cambios en los requerimientos son inevitables, sino deseables y, en pro del beneficio del equipo, se decidió asumir que esto sería cierto durante el desarrollo del presente trabajo, lo que fue beneficioso ya que, efectivamente, se presentaron cambios y añadiduras que no fueron contempladas desde el principio.

En este mismo orden de ideas se usaron diferentes artefactos pertenecientes a XP entre los cuales podemos mencionar las Historias de Usuario, Pruebas de Aceptación y, por último, las Tarjetas de Tareas (del inglés “Task Cards”) diseñadas con el fin de llevar a cabo un seguimiento del trabajo que se estaba realizando.

Las iteraciones que tuvieron lugar para la culminación del presente trabajo fueron las siguientes:

4.5.1. Primera Iteración (Investigación):

Esta iteración tuvo una duración de cuatro (4) semanas, se realizó debido a los cambios de requerimientos detectados con respecto al trabajo anterior. Se realizaron los siguientes procesos

4.5.1.1. Planeación

En un principio se contemplaba como objetivo el desarrollar un prototipo del módulo de predicción de cambios, cuyo fin era la creación de un modelo predictivo basado en los tipos de páginas web a estudiar (blogs, sitios de noticias, wikis, sitios de empresas y sitios de almacenamiento de media digital), de modo que el rastreo se hiciera de acuerdo con el estimado originado por ese desarrollo. Sin embargo, al hacer el análisis previo al inicio del desarrollo, se encontró que la detección de cambios no estaba automatizada, por lo que era necesario analizar individualmente cada rastreo realizado

para determinar si el cambio era lo suficientemente sustancial como para tomarlo en cuenta dentro del modelo predictivo.

En base al estudio realizado se determinó que el método correcto a seguir era detener la creación del modelo predictivo y, en cambio darle prioridad a la creación de una aplicación que automatizase el proceso previamente explicado. Se realizó la creación de las Historias de Usuario tal y cómo están descritas a continuación en la Tabla 3 y Tabla 4.

Tabla 3 Historia de Usuario 1. Iteración 1

Historia de Usuario	
Número: 1	Usuario: Mercy Ospina
Nombre de Historia: Investigación de técnicas de detección de cambios	
Prioridad (Alta/Media/Baja): Alta	Riesgo (Alto/Medio/Bajo): Bajo
Puntos estimados: 1	Iteración asignada: 1
Programador responsable: Luis Aguiar	
Descripción: Realizar una exhaustiva investigación y estudio de las técnicas de detección de cambios que han sido desarrolladas, incluyendo, pero no limitándose a las técnicas basadas en texto e imágenes, de igual forma, realizar un análisis superficial para determinar si es factible combinar diferentes métodos en un mismo análisis.	
Validación: El programador demuestra que conoce las técnicas de detección de cambios que investigó y es capaz de determinar cuáles deben ser usadas o descartadas para el éxito del desarrollo de la aplicación.	

Tabla 4 Historia de Usuario 2. Iteración 1

Historia de Usuario	
Número: 2	Usuario: Mercy Ospina
Nombre de Historia: Preparación de entorno de desarrollo con los elementos necesarios para la construcción de la aplicación.	
Prioridad (Alta/Media/Baja): Alta	Riesgo (Alto/Medio/Bajo): Bajo
Puntos estimados: 1	Iteración asignada: 1

Programador responsable: Luis Aguiar
Descripción: Realizar la instalación de las herramientas necesarias para el desarrollo de la aplicación.
Validación: El programador crea un ambiente de que contenga las herramientas necesarias para la construcción de la aplicación.

4.5.1.2. Diseño

No fue necesaria la elaboración de un diseño en la presente iteración.

4.5.1.3. Codificación

Para la configuración del entorno de desarrollo, es necesario realizar la instalación de diversas herramientas.

En primer lugar, se realiza la instalación de Python, que será la herramienta principal por usar durante el desarrollo de la aplicación, esto se realiza de la siguiente manera:

- Se realiza la descarga del instalador correspondiente a la versión del sistema operativo adecuada desde el sitio web <https://www.python.org/downloads/>
- Se ejecuta el instalador descargado.
 - Es necesario verificar que Python se agrega al PATH para asegurar la correcta ubicación del intérprete, como muestra la Figura 15.
- Por último, se hace clic en el botón de “Siguiente” y se espera a que la instalación se realice.

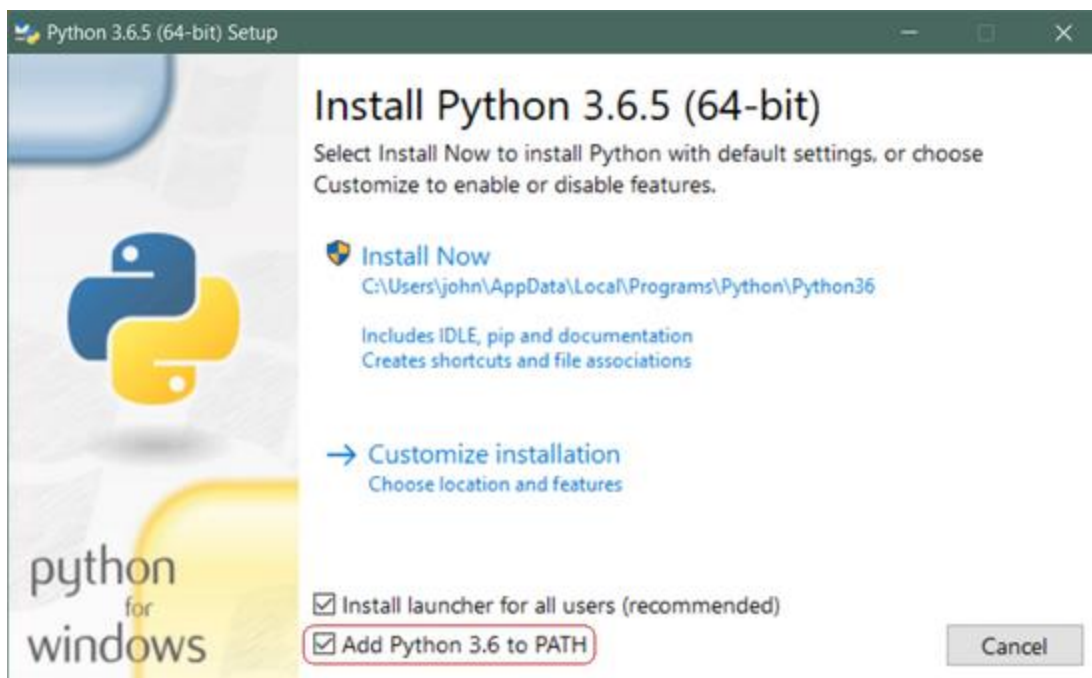


Figura 15 Instalación Python

No es necesario realizar una instalación adicional del manejador de paquetes de Python (conocido como Pip), ya que está incluido en la instalación de Python a partir de su versión 2.7.9 (en el caso de Python 2) y 3.4 (en el caso de Python 3).

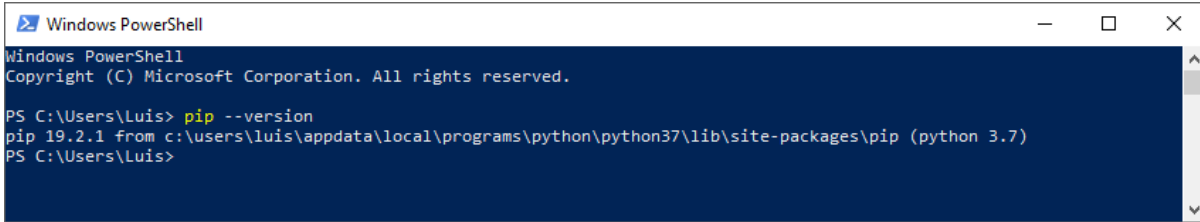
4.5.1.4. Pruebas

A continuación, se muestran las pruebas realizadas durante esta iteración:

Tabla 5 Caso de Prueba 1. Iteración 1

Caso de Prueba	
Número de Caso de Prueba: 1	Número de Historia de Usuario: 2
Descripción: Comprobar la instalación de Python mediante la impresión por consola de la versión correspondiente.	
Resultado:	

Tabla 6 Caso de Prueba 2. Iteración 1

Caso de Prueba	
Número de Caso de Prueba: 2	Número de Historia de Usuario: 2
Descripción: Comprobar la instalación del manejador de paquetes Pip de Python mediante la impresión por consola de la versión correspondiente.	
Resultado:	
 <pre> Windows PowerShell Copyright (C) Microsoft Corporation. All rights reserved. PS C:\Users\Luis> pip --version pip 19.2.1 from c:\users\luis\appdata\local\programs\python\python37\lib\site-packages\pip (python 3.7) PS C:\Users\Luis> </pre>	

4.5.2. Segunda Iteración (Cambio cuantitativo):

Esta iteración tuvo una duración de dos (2) semanas, se realizó inmediatamente al terminar la anterior y fue el período durante el cual se creó el código encargado de realizar el análisis cuantitativo de los cambios mediante el uso de técnicas de análisis de texto, tomando como base el módulo **DiffLib** de Python.

4.5.2.1. Planeación

Luego de que el equipo de trabajo discutiese los resultados logrados en la primera iteración, se definieron los requisitos funcionales mínimos que se necesitaban para determinar que la aplicación operase de manera satisfactoria, éstos fueron descritos en una única Historia de Usuario descrita en la Tabla 7, que se muestra a continuación:

Tabla 7 Historia de Usuario 1. Iteración 2

Historia de Usuario	
Número: 2	Usuario: Mercy Ospina
Nombre de Historia: Creación de aplicación de detección de cambios mediante el uso de técnicas cuantitativas basada en métodos de análisis de texto.	
Prioridad (Alta/Media/Baja): Alta	Riesgo (Alto/Medio/Bajo): Medio
Puntos estimados: 1	Iteración asignada: 2
Programador responsable: Luis Aguiar	

Descripción: Crear un código que parametriza los cambios de texto, tomando el código fuente de la página web a analizar y comparándolo con el código fuente de una versión anterior del mismo sitio, separándolos en bloques de texto con el fin de comparar cada uno de los bloques de manera directa, luego de asignarles un valor de identificación único a cada uno. Luego de esto, obtener un valor final entre cero (0) y cien (100), tomando en cuenta el *Principio de Pareto*, donde la mayor ponderación será otorgada a los bloques que contengan trozos de texto con la mayor cantidad de caracteres.

Validación: El programador desarrolla un trozo de código cuya entrada es un par de direcciones URL donde una de éstas está enlazada con la versión almacenada del sitio web y la segunda dirige al sitio en su versión más reciente y devuelve un valor entre cero (0) y cien (100), tomando en cuenta que la mayor ponderación debe estar sobre los bloques de texto que contengan la mayor cantidad de caracteres, y la división de ponderación debe ser basada en el Principio de Pareto.

4.5.2.2. Diseño

Se hizo seguimiento de la doctrina fundamental de diseño en XP, la que dice que éste debe ser lo más simple posible, por lo tanto, se creó un modelo de caja negra, donde para el usuario lo que ocurre dentro del código no es observado, se introducen los datos requeridos por la consola y se leen los resultados arrojados en la misma ventana, este diseño se puede observar en la Figura 16 mostrada a continuación:

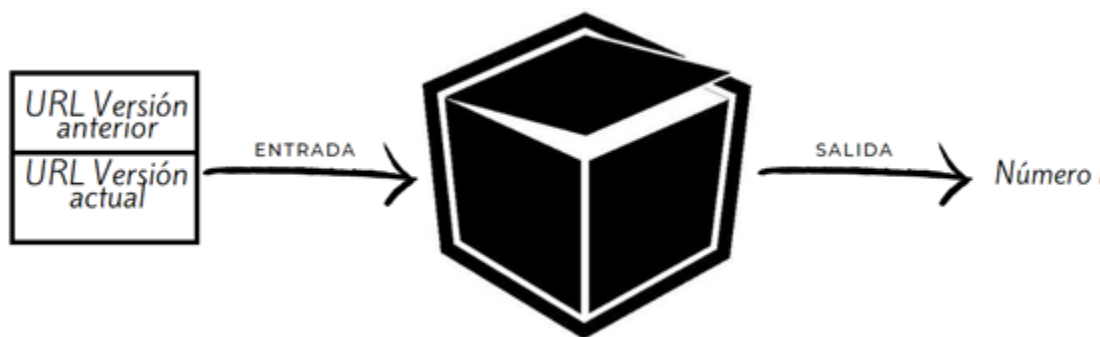


Figura 16 Diseño Historia de Usuario 1. Iteración 2

4.5.2.3. Codificación

En primer lugar, se realizó la importación e instalación de las dependencias necesarias para realizar un proceso de “scraping” de sitios web, entre las que están los módulos **Selenium**, **BeautifulSoup** y **os** de Python, así como el módulo **DiffLib**, que será usado para el análisis de variación de texto. Adicionalmente es necesaria la importación de los módulos **math** y **statistics**, que hacen más expeditos los cálculos a efectuar, las importaciones se realizan de manera muy sencilla, sólo es necesario agregar las líneas de código mostradas en la Figura 17 al principio del script.

```
import difflib
import os
from selenium import webdriver
from bs4 import BeautifulSoup
import math
import statistics
```

Figura 17 Importación de dependencias

Al ya tener esto, se procede a la definición de la función, que acepta como parámetros los dos elementos mencionados anteriormente y se hace uso de las dependencias **Selenium** y **BeautifulSoup** para abrir una ventana del navegador, cargar el sitio web, obtener una copia del código HTML luego de realizar la renderización completa de la página, separarlo usando las etiquetas `<div>` navegando a través del árbol DOM renderizado y almacenar cada bloque en un elemento de una lista de diccionarios para su posterior análisis. El código encargado de este proceso puede ser observado en la Figura 18.

```
# Opciones del browser
alloptions = webdriver.ChromeOptions()
alloptions.add_argument('headless')
alloptions.add_argument("--window-size=1920,1080")

browser = webdriver.Chrome(options=alloptions)
browser.get(p_url)
p_html = browser.page_source
# Se guarda un screenshot de la página en el lugar correspondiente
screenshot = util.fullpage_screenshot(browser, 'static/old_screenshot.png')
browser.quit()

browser = webdriver.Chrome(options=alloptions)
browser.get(url)
html = browser.page_source
browser.quit()

text = BeautifulSoup(html, "lxml")
divs = text.find("div")
head = text.find("head")
body = text.find("body")
p_text = BeautifulSoup(p_html, "lxml")
p_divs = p_text.find("div")
salida = ""

counter = 0
for item in divs.next_siblings:
    try:
        if counter == 0:
            unsorted_dicctexts.append({'id': counter, 'value': str(divs)})
            counter += 1

        unsorted_dicctexts.append({'id': counter, 'value': str(item.next)})
        counter += 1
    except AttributeError:
        if counter == 0:
            unsorted_dicctexts.append({'id': counter, 'value': str(divs)})
            counter += 1

        unsorted_dicctexts.append({'id': counter, 'value': str(item.next)})
        counter += 1

counter = 0
for item in p_divs.next_siblings:
    try:
        if counter == 0:
            p_unsorted_dicctexts.append({'id': counter, 'value': str(p_divs)})
            counter += 1

        p_unsorted_dicctexts.append({'id': counter, 'value': str(item.next)})
        counter += 1
    except AttributeError:
        if counter == 0:
            p_unsorted_dicctexts.append({'id': counter, 'value': str(p_divs)})
            counter += 1

        p_unsorted_dicctexts.append({'id': counter, 'value': str(item.next)})
        counter += 1

for i in unsorted_dicctexts:
    texts.append(i['value'])

for i in p_unsorted_dicctexts:
    p_texts.append(i['value'])
```

Figura 18 Scraping y almacenamiento del código HTML

Los diccionarios son elementos que, en este caso, tienen dos (2) partes; cada una conformada por un par clave valor. El primer elemento está conformado por un par con la estructura “id: <entero>” donde entero es un número único mayor o igual a cero (0) que será usado como identificador al momento de realizar la reconstrucción del sitio web, y el segundo par está definido de la manera “value: <string>”, donde <string> es el código del bloque en cuestión.

El siguiente paso es verificar que no haya elementos con valor **None**, situación que es posible que ocurra dada la manera en que **BeautifulSoup** maneja los elementos extraídos (asignando valores nulos a bloques de texto vacíos), en caso de haber, simplemente se sustituyen por cadenas de caracteres vacías para poder realizar métodos que se ejecuten sobre texto en éstas. Luego de esto se procede a ordenar ambas listas por la cantidad de caracteres que tengan sus bloques (en el alcance de esta aplicación se asume que los bloques con mayor cantidad de caracteres son los más importantes y que si hubo al menos un bloque donde la diferencia es tan extrema que alteró el orden entre las versiones éste siempre será significativo) al momento de realizar la comparación usando el método **SequenceMatcher** del módulo **DiffLib**, el cual elimina texto basura (se define como texto basura cualquier cadena de caracteres que se repita tanto que conforme más del uno por ciento de toda la secuencia) y devuelve un número entre cero (0) y uno (1) que se interpreta así: mientras mayor sea el valor, mayor igualdad existe entre los textos analizados, se resta el resultado del valor máximo 1 y se multiplica por cien con el fin de adquirir un valor porcentual de diferenciación, por lo que esto arroja como resultado final una lista de enteros entre cero (0) y cien (100) que se leen como el porcentaje de diferencias entre los bloques analizados; esto lo podemos ver en la Figura 19.

```

for iterator in range(0, len(texts)):
    if texts[iterator] is None:
        texts[iterator] = ""
    if p_texts[iterator] is None:
        p_texts[iterator] = ""

    result = difflib.SequenceMatcher(None, texts[iterator], p_texts[iterator]).ratio()
    difference.append(round((1 - result) * 100, 2))

texts.sort(key=len, reverse=True)
p_texts.sort(key=len, reverse=True)

for iterator in range(0, len(texts)):
    result = difflib.SequenceMatcher(None, texts[iterator], p_texts[iterator]).ratio()
    sorted_difference.append(round((1 - result) * 100, 2))

```

Figura 19 Porcentaje de diferenciación entre bloques

Por último, aplicamos el Principio de Pareto sobre el arreglo de resultados, así que para facilitar esto lo dividimos en dos listas diferentes, la primera tendrá el veinte por ciento (20%) de los resultados (éstos son los porcentajes de diferencias entre los bloques con mayor cantidad de texto) y la segunda, tendrá el ochenta por ciento (80%) restante.

Debido a que necesitamos una lista de enteros y esta división creó una lista de listas similar a lo siguiente: `[[a, b, c, ...]]`; es necesario realizar un proceso conocido como “aplanamiento”, donde simplemente trasladamos los elementos de las listas internas a otra para que queden de la manera `[a, b, c, ...]` y procedemos a realizar la ponderación de sus valores. Sobre la primera lista (que contiene el veinte por ciento de los elementos) se ejecuta el método `mean` del módulo `statistics` y se multiplica por cero punto ocho (0.8) con lo que se obtiene la ponderación del promedio de sus valores y sobre la segunda lista se realiza un proceso similar pero el factor de ponderación será cero punto dos (0.2), realizados ambos cálculos, se suman los valores y se imprimen por consola. El código que realiza todo esto puede ser observado en la Figura 20.

```
# 20% de los bloques tienen 80% del peso
twenty_percent_elements.append(sorted_difference[0:math.floor(len(difference) * 0.2)]) # Elementos con 80% del valor
eighty_percent_elements.append(sorted_difference[math.floor(len(difference) * 0.2):]) # Elementos con 20% del valor

for sublist in twenty_percent_elements:
    for item in sublist:
        flatten20.append(item)

for sublist in eighty_percent_elements:
    for item in sublist:
        flatten80.append(item)

# Asignación del peso cuantitativo
porcentajes_cambios_ordenados.append(flatten20)
porcentajes_cambios_ordenados.append(flatten80)
cambio_peso_80 = statistics.mean(porcentajes_cambios_ordenados[0]) * 0.8
cambio_peso_20 = statistics.mean(porcentajes_cambios_ordenados[1]) * 0.2
cambio_total = cambio_peso_80 + cambio_peso_20

print("El valor de cambio cuantitativo es: {}".format(cambio_total))
```

Figura 20 Ponderación cuantitativa según el Principio de Pareto

4.5.2.4. Pruebas

A continuación, se muestran las pruebas realizadas durante esta iteración:

Tabla 8 Caso de Prueba 1. Iteración 2

Caso de Prueba	
Número de Caso de Prueba: 1	Número de Historia de Usuario: 1
Descripción: Comprobar el funcionamiento correcto del cálculo del análisis cuantitativo mediante la impresión del resultado por consola.	
Resultado:	

```

Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

PS C:\Users\Luis> python cambios.py
El valor de cambio cuantitativo es: 0.024
    
```

4.5.3. Tercera Iteración (Desarrollo de la Interfaz):

La presente iteración tuvo una duración de cuatro (4) semanas, se inició al terminar la anterior y fue el período durante el cual se creó la interfaz gráfica de la aplicación mediante el uso del framework **Flask**.

4.5.3.1. Planeación

Durante la reunión inicial que se realiza al empezar cada iteración, fue definido que se necesitaría una interfaz gráfica con la cual el usuario pudiese interactuar y que mostrase los resultados de la comparación de texto entre versiones de la manera descrita en las Historias de Usuario mostradas a continuación:

Tabla 9 Historia de Usuario 1. Iteración 3

Historia de Usuario	
Número: 1	Usuario: Mercy Ospina
Nombre de Historia: Creación de Interfaz de Entrada de la aplicación de Detección de Cambios	
Prioridad (Alta/Media/Baja): Alta	Riesgo (Alto/Medio/Bajo): Bajo
Puntos estimados: 2	Iteración asignada: 3
Programador responsable: Luis Aguiar	
<p>Descripción: Realizar la instalación del framework Flask con el fin de usarlo para construir las interfaces web que serán usadas por el usuario para interactuar con la aplicación.</p> <p>Crear una interfaz sencilla en donde se inserten las direcciones URL o rutas locales donde estén almacenados los sitios web y ejecute el código creado con anterioridad al hacer clic en un botón.</p>	

Validación: Luego de hacer clic sobre el botón correspondiente, se imprimen los resultados por la consola del servidor comprobando que efectivamente se realizó la ejecución del código de manera satisfactoria.

Tabla 10 Historia de Usuario 2. Iteración 3

Historia de Usuario	
Número: 2	Usuario: Mercy Ospina
Nombre de Historia: Creación de interfaz descriptiva de salida que muestre los resultados obtenidos.	
Prioridad (Alta/Media/Baja): Alta	Riesgo (Alto/Medio/Bajo): Medio
Puntos estimados: 1	Iteración asignada: 3
Programador responsable: Luis Aguiar	
<p>Descripción: Se debe realizar una interfaz que contenga una captura de la renderización de la versión almacenada del sitio web, una captura de la versión contra la que se está comparando con los bloques que contengan cambios resaltados de alguna forma (por ejemplo, un borde de color rojo alrededor de este) y una gráfica que muestre el porcentaje de cambios por cada uno de los bloques junto a una leyenda que explique los resultados de la gráfica. Adicionalmente, se requiere un botón que realice un retorno a la interfaz descrita en la Historia de Usuario previa, en caso de ser necesaria otra comparación.</p>	
<p>Validación: La interfaz de salida es renderizada luego de la ejecución del código descrita en la Historia de Usuario anterior y se muestran, de manera legible, los elementos solicitados.</p>	

4.5.3.2. Diseño

Continuando la filosofía de XP, se decidió mantener un diseño simple al momento de diseñar cada una de las interfaces solicitadas en las Historias de Usuario previamente descritas, bosquejos simples fueron creados para describir una idea de cómo debía ser construida cada interfaz, los cuales mostraremos a continuación:

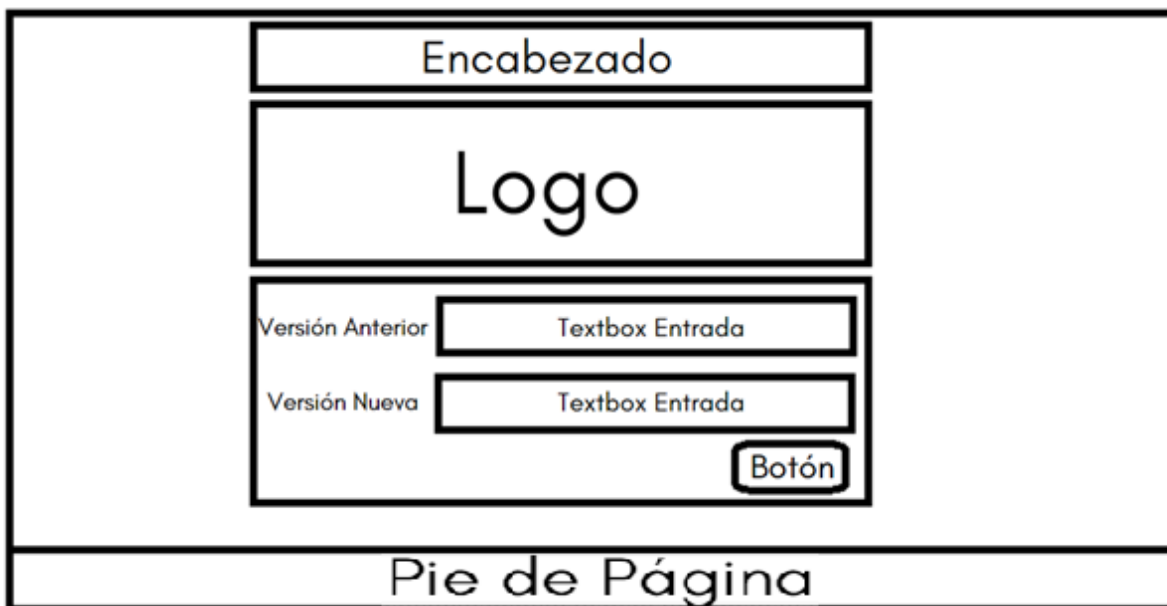


Figura 21 Diseño Historia de Usuario 1. Iteración 3

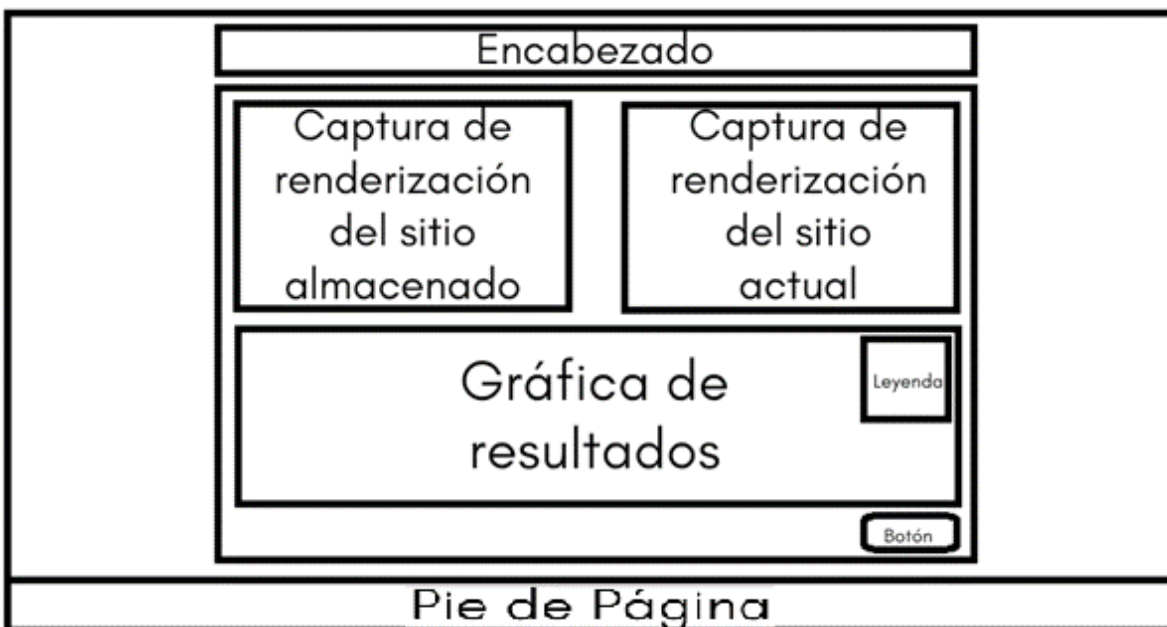


Figura 22 Diseño Historia de Usuario 2. Iteración 3

4.5.3.3. Codificación

Se inició con la instalación del framework **Flask**, el cual será usado para llevar a cabo diversos procesos que forman parte del desarrollo de la interfaz web de la aplicación, como son el inicio del servidor web y el direccionamiento correcto de las rutas donde se encuentren los archivos html y los scripts. La instalación se ejecutó el comando “pip install flask” desde la consola de comandos de la manera que muestra la Figura 23:

```

Administrator: Windows PowerShell
(venv) PS C:\Users\Luis\Tesis> pip install Flask
Collecting Flask
  Using cached https://files.pythonhosted.org/packages/9b/93/628509b8d5dc749656a9641f4caf13540e2cdec85276964ff8f43bbb1d3b/Flask-1.1.1-py2.py3-none-any.whl
Collecting Jinja2>=2.10.1 (from Flask)
  Using cached https://files.pythonhosted.org/packages/1d/e7/fd8b501e7a6dfe492a433deb7b9d833d39ca74916fa8bc63dd1a4947a671/Jinja2-2.10.1-py2.py3-none-any.whl
Collecting click>=5.1 (from Flask)
  Using cached https://files.pythonhosted.org/packages/fa/37/45185cb5abbc30d7257104c434fe0b07e5a195a6847506c074527aa599ec/Click-7.0-py2.py3-none-any.whl
Collecting itsdangerous>=0.24 (from Flask)
  Using cached https://files.pythonhosted.org/packages/76/ae/44b03b253d6fade317f32c24d100b3b35c2239807046a4c953c7b89fa49e/itsdangerous-1.1.0-py2.py3-none-any.whl
Collecting Werkzeug>=0.15 (from Flask)
  Downloading https://files.pythonhosted.org/packages/b7/61/c0a1adf9ad80db012ed7191af98fa05faa95fa09e6b71bb6fa8b66e6a43/Werkzeug-0.15.6-py2.py3-none-any.whl (328kB)
    100% |#####| 337kB 354kB/s
Collecting MarkupSafe>=0.23 (from Jinja2>=2.10.1->Flask)
  Using cached https://files.pythonhosted.org/packages/65/c6/2399700d236d1dd681af8aebff1725558cddf6e43d7a5184a675f4711f5/MarkupSafe-1.1.1-cp37-cp37m-win_amd64.whl
Installing collected packages: MarkupSafe, Jinja2, click, itsdangerous, Werkzeug, Flask
Successfully installed Flask-1.1.1 Jinja2-2.10.1 MarkupSafe-1.1.1 Werkzeug-0.15.6 click-7.0 itsdangerous-1.1.0
You are using pip version 18.1, however version 19.2.3 is available.
You should consider upgrading via the 'python -m pip install --upgrade pip' command.
(venv) PS C:\Users\Luis\Tesis>

```

Figura 23 Historia de Usuario 1. Iteración 1. Instalación Flask

Posterior a esto se creó un directorio con la estructura usada en el estándar de Flask, la cual está conformada por un archivo llamado **app.py** que es el principal de la aplicación, una carpeta **static**, la cual contendrá los elementos estáticos de la aplicación (imágenes, videos, archivos de sonido, etc.) y una carpeta llamada **templates**, que es dónde se almacenarán los archivos html que conforman las interfaces.

El siguiente paso es la creación de los archivos html que contendrán el código que será renderizado por el navegador al momento de iniciar el servidor, iniciamos con la interfaz diseñada en la Historia de Usuario 1, dónde es importante resaltar que ambos elementos “textbox” deben encontrarse en un formulario html y el botón debe ser un elemento de tipo “submit”, de no ser así, no se podrá enviar el contenido de cada textbox como parámetro a la función creada en la iteración previa; a este archivo lo nombramos “index.html”. Luego de este proceso, se inicia la creación de la interfaz descrita en la Historia de Usuario 2, durante esta creación es importante recalcar el uso de lo que Flask llama “plantillas”, que son trozos de código que se insertan en las etiquetas html y que permiten al intérprete de Flask ubicar los recursos ubicados en el directorio del proyecto y que serán cargados al momento de renderizar el sitio web. Podemos observar un ejemplo de esto a continuación:

```

<div id="new_rendered_page" name="new_rendered_page" class="col-sm-6">
  
</div>

```

Figura 24 Ejemplo de uso de plantillas en Flask

Donde el código que se está asignando al parámetro **src** es interpretado por Flask como “ruta del archivo con nombre ‘old_screenshot.png’ ubicado en la carpeta **static**”. Este último archivo tiene como nombre “resultados.html”.

Luego de realizado este proceso, se modifica el archivo **app.py** con el fin de señalar al intérprete de Flask cuál es el archivo html que se renderizará al momento de ser iniciado, así como de cuáles elementos serán los parámetros de la función a ejecutar y la ruta del archivo “resultados.html”. Esto se observa a continuación:

```
import changeDetector
from flask import Flask, render_template, request

app = Flask(__name__)
app.config['SEND_FILE_MAX_AGE_DEFAULT'] = 0 # Duración de imágenes en caché = 0

@app.route("/")
def main():
    return render_template('index.html')

@app.route("/changes", methods=['POST'])
def changes():
    old_url = request.form['old_url']
    new_url = request.form['new_url']
    changeDetector.change_detector(old_url, new_url, 'diferencia'.upper())
    return render_template('resultados.html')

if __name__ == "__main__":
    app.run(debug=True)
```

Figura 25 Archivo app.py

Finalmente, se modifica el código de la función de detección de cambios para que reconstruya el sitio web a partir de los bloques en los que se dividió el código obtenido de la versión actualizada del sitio durante la iteración previa añadiendo un borde a los bloques que contienen algún cambio, adicionalmente se añade la creación del gráfico que será mostrado con el porcentaje de cambio por cada uno de los bloques, la renderización de esta reconstrucción, captura de éste en una imagen y guardado de cada elemento en la carpeta **static** para que el intérprete de Flask los inserte correctamente en la interfaz. Este proceso se divide en varias partes, que serán explicadas a continuación:

- Se inyecta una cadena de caracteres que contenga las instrucciones para la carga de JavaScript, en caso de que sea necesario.
- Se reordenan los bloques usando como base su número de id único, agregado durante la iteración anterior.
- Se concatena el texto contenido dentro de cada elemento de la lista de bloques en una variable de salida.
- De haberse detectado cambios en algún bloque, se inyecta una cadena de caracteres, agregando un borde de color rojo al parámetro "style" del bloque (de tenerlo, en caso contrario, se añade el parámetro y su contenido)
- Se crea un archivo en modo de escritura cuyo nombre y ruta serán similares al del archivo que contiene la versión previa, añadiéndole la cadena "_updated.html" al nombre.
- Se escribe el contenido de la variable de salida en este nuevo archivo y se cierra. El proceso hasta este punto ejecuta el código que se muestra en la Figura 26, mostrada a continuación:

```

for element in indices_cambios:
    for element2 in salidafinal:
        if element2['id'] == element:
            if 'style' in element2['value']:
                element2['value'] = element2['value'].replace('style=', 'style="border: 3px solid red; ', 1)
            else:
                reemplazo_border = '< ' + ' style="border: 3px solid red;"'
                element2['value'] = element2['value'].replace('<', reemplazo_border, 1)

for element in salidafinal:
    if element['value'] != 'None':
        salida = salida + element['value'] + '\n'

salida = salida + "</body>" + '\n'
salida = salida + "</html>"

url2 = url.replace('.html', '_updated.html')
g = open(url2, "w", encoding="utf-8")
g.write(salida)
g.close()

# Screenshot
# Opciones del browser
aloptions = webdriver.ChromeOptions()
aloptions.add_argument('headless')
aloptions.add_argument("--window-size=1920,1080")
screenshot = util.fullpage_screenshot(browser, 'static/new_screenshot.png')
browser.quit()

```

Figura 26 Reconstrucción y captura del sitio web actualizado

- Usando Selenium renderizamos el archivo recientemente construido y el archivo que contiene la versión almacenada y se toma una captura de pantalla de ambos elementos, ambas capturas son almacenadas en la carpeta **static** del proyecto.
- Se importan los módulos **matplotlib** y **numpy** de Python, los cuáles se usarán para la creación del gráfico de barras y la leyenda de este, se guarda un archivo de imagen del gráfico en la carpeta **static** del proyecto, la descripción del código que realiza este trabajo se puede observar en la Figura 27:

```
# Creacion Grafico de barras
obj = list(range(len(difference)))
objects = [x + 1 for x in obj]
qualitative_difference = [0] * len(difference)
contador = 0

for i in range(len(difference)):
    if difference[i] != 0:
        qualitative_difference[i] = qualitative_i[contador]
        contador +=1

# Revisar valores de importancia para la gráfica
quantitative_importance = [x * 0.4 for x in difference]
y_pos = np.arange(len(objects))
performance = quantitative_importance
plt.tick_params(axis='x', which='major', labelsize=5)
plt.figure(figsize=(10,3.8))
barlist = plt.bar(y_pos,performance,align='center',alpha=0.5)

for i in range(0, len(barlist)):
    height = barlist[i].get_height()
    if height >= 50:
        barlist[i].set_color('r')
    elif height >= 20 and height < 50:
        barlist[i].set_color('y')
    elif height > 0 and height < 20:
        barlist[i].set_color('g')

c_significativo = mpatches.Patch(color='r', label = 'Significativo')
c_intermedio = mpatches.Patch(color='y', label = 'Intermedio')
c_no_significativo = mpatches.Patch(color='g', label = 'No significativo')
plt.legend(handles=[c_significativo, c_intermedio, c_no_significativo], title='Tipo de Cambio', fontsize='small', fancybox=True)
plt.xticks(y_pos, objects)
plt.ylabel('% de cambios')
plt.xlabel('ID de bloques')
plt.title('Cambios por bloque de texto')
plt.savefig('static/block_text_change_percentage.png')
# Fin Grafico
```

Figura 27 Creación gráfico de barras con resultados

4.5.3.4. Pruebas

A continuación, se muestran las pruebas realizadas durante esta iteración:

Tabla 11 Caso de Prueba 1. Iteración 3

Caso de Prueba	
Número de Caso de Prueba: 1	Número de Historia de Usuario: 1
Descripción: Comprobar la instalación de Flask mediante la impresión por consola de la versión correspondiente.	
Resultado:	

```

Windows PowerShell
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

PS C:\Users\Luis> python
Python 3.7.2 (tags/v3.7.2:9a3ffc0492, Dec 23 2018, 23:09:28) [MSC v.1916 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> import flask
>>> flask.__version__
'1.1.1'
>>>
    
```

Tabla 12 Caso de Prueba 2. Iteración 3


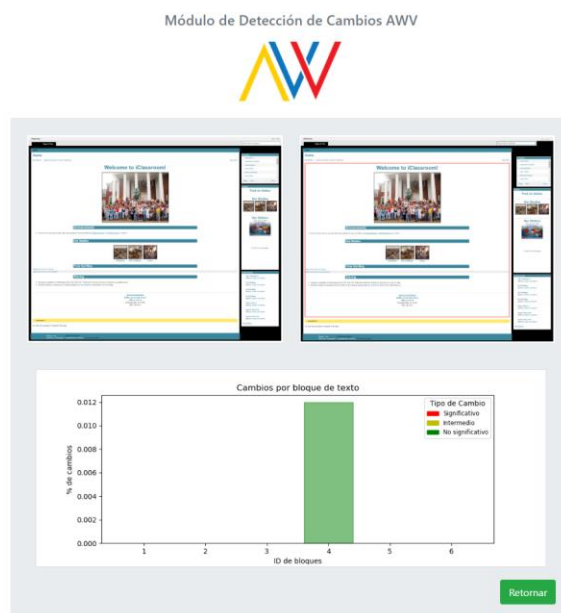
Caso de Prueba	
Número de Caso de Prueba: 2	Número de Historia de Usuario: 1
Descripción: Comprobar la creación y muestra correcta de la interfaz de inicio descrita en la Historia de Usuario 1	
Resultado:	
	

Tabla 13 Caso de Prueba 3. Iteración 3

Caso de Prueba	
Número de Caso de Prueba: 1	Número de Historia de Usuario: 2

Descripción: Comprobar la creación y muestra correcta de la interfaz de salida descrita en la Historia de Usuario 2, así como de los resultados correctos luego de la ejecución de la aplicación.

Resultado:



4.5.4. Cuarta Iteración (Cambio cualitativo):

Esta iteración tuvo una duración de dos (2) semanas, se agregó la funcionalidad a la aplicación de detectar la posición de los cambios con respecto a lo que visualiza el navegador al momento de cargar la página web y acorde a esto, se le asigna un valor dependiendo de lo visible que sea durante la renderización inicial y se ponderó con el valor obtenido por el análisis cuantitativo obtenido durante la segunda iteración.

4.5.4.1. Planeación

Luego de ser terminada y revisada la iteración anterior se observó que era posible obtener los datos de la ubicación espacial de los bloques que poseen cambios, por lo que fue decidido agregar un elemento más al análisis, con el objetivo de realizar una ponderación de cambios aún más precisa ya que los cambios realizados en el área visible para el usuario durante la carga inicial de una página web son más relevantes que aquellos que se realicen en un área no visible, por lo que se deben evaluar ambos casos de manera distinta. Los requerimientos fueron descritos en la Historia de Usuario que se muestra en la Tabla 14:

Tabla 14 Historia de Usuario 1. Iteración 4

Historia de Usuario

Número: 1	Usuario: Mercy Ospina		
Nombre de Historia: Agregación de cálculos cualitativos de importancia de cambio basados en la posición del bloque.			
Prioridad (Alta/Media/Baja): Alta		Riesgo (Alto/Medio/Bajo): Alto	
Puntos estimados: 1	Iteración asignada: 4		
Programador responsable: Luis Aguiar			
Descripción: Se debe agregar un elemento de análisis cualitativo a la aplicación de detección de cambios, basando éste en la posición espacial del bloque donde está ubicado cada cambio con respecto a lo que se puede visualizar al iniciar un sitio web, ya que no sólo importa el tamaño del cambio en términos de texto, sino también en términos de si el usuario puede observarlo a simple vista o no.			
Validación: Se debe analizar la posición espacial del (o los) bloque(s) donde se encuentren cambios y agregar el valor obtenido como resultado de esto a la ponderación de importancia de un cambio.			

4.5.4.2. Diseño

Para poder realizar el cálculo de la importancia cualitativa es necesario colocarle un valor a cada posición de lo que se puede observar en el navegador al momento de su carga inicial, ya que los elementos visibles tendrán un mayor grado de importancia en comparación con los no visibles. Adaptando el modelo descrito por (Sanoja, 2015), el cual separa el contenido del sitio web en bloques y los representa por medio de una estructura de dos dimensiones, donde cada celda será un valor asignado de acuerdo con el área donde se ubique. El contenido del sitio web renderizado se divide en un número n de subsecciones, a cada subsección se le asigna un valor que depende de dos elementos principales: el primero es el tipo de contenido que se quiere visualizar, donde si es contenido principal se le asigna un puntaje mayor a las celdas centrales de la matriz (a la que llamaremos **Matriz de Posición**), ya que es la ubicación usual de éste dentro de los sitios web, mientras que si por el contrario, lo que se quiere es evaluar la publicidad, se le asignarían los puntajes mayores a las celdas borde de la matriz, de nuevo, porque es el sitio usual donde están ubicados en la mayoría de los sitios web. El segundo elemento que evaluar al momento de colocar el puntaje es que tan alejados están de la celda con el mayor valor, los elementos adyacentes tendrán el valor del elemento mayor menos la distancia con respecto a éste.

El número n se escoge de manera arbitraria, dependiendo de la precisión con la que se quiera evaluar cada elemento, mientras más alto sea, más precisa será la evaluación, pero también tendrá una complejidad más alta. Para este proyecto se escogió $n = 5$ como la base de la cual partir para la evaluación, y se creó un diseño que está representado en la Figura 28, la cual será mostrada a continuación:

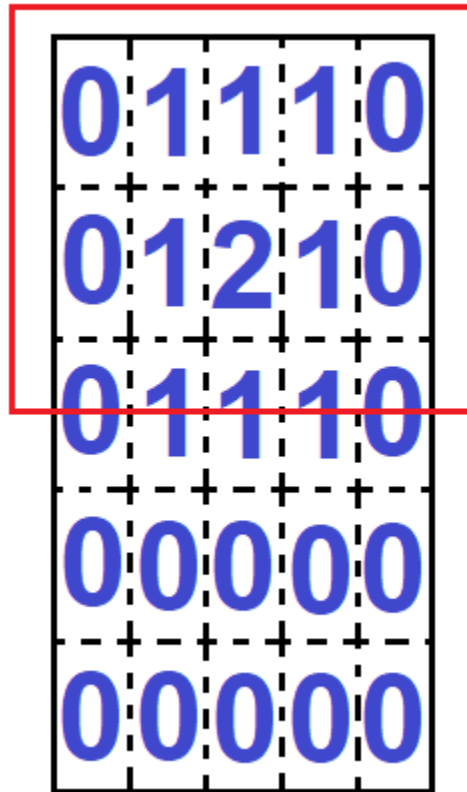


Figura 28 Representación de Sitio Web en una Matriz de Posición

La cual representa cada elemento que interactúa de manera visual con el sitio web, descrito de la forma siguiente:

- El área rodeada por la línea negra sólida representa el contenido del sitio web luego de la renderización.
- El área rodeada por la línea roja sólida representa lo que el navegador muestra en un momento determinado (en este caso luego de la carga inicial del sitio web).
- Las áreas divididas por las líneas punteadas negras son las n secciones en las que se dividió el contenido del sitio web, tanto de manera vertical como de manera horizontal, por lo que la Matriz de Posición será una matriz cuadrada.
- Los números en azul representan el valor asignado a cada una de las áreas dependiendo de la importancia que tienen de acuerdo con su posición durante la carga inicial. Nótese que el valor más alto se le fue asignado a la celda central y de ahí se colocó un valor menor dependiendo de la distancia que se encuentra con respecto al centro.

4.5.4.3. Codificación

El primer paso fue la creación de la matriz descrita en el diseño de la presente iteración, para asegurar una adecuada correspondencia con el visualizador, se procedió a obtener los valores de altura y anchura de la ventana del navegador, así como los del sitio completo luego de su renderización, incluyendo el espacio que no es posible visualizar durante la carga inicial del sitio, esto se realizó mediante el uso de **Selenium**

(para realizar el renderizado, al igual que en iteraciones anteriores) e inyecciones de código **JavaScript** mediante la función “execute_script”, así mismo, se procedió a la creación de un arreglo que contuviese los valores de las coordenadas donde se ubican los separadores definidos en el diseño (siendo designado un arreglo para los valores de la coordenada X y otro para los valores de la coordenada Y), como se muestra en la Figura 29, a continuación:

```
# Operaciones con el browser
browser = webdriver.Chrome(options=alloptions)
# Trabajaremos con el centro del viewport porque queremos determinar cambios en contenido
viewport_width = browser.execute_script("return document.body.clientWidth")
viewport_height = browser.execute_script("return window.innerHeight")
site_width = browser.execute_script("return document.body.offsetWidth")
site_height = browser.execute_script("return document.body.parentNode.scrollHeight")
site_n_horizontal = site_width / n_cualitativo
site_n_vertical = site_height / n_cualitativo
site_x_axis = [0, site_n_horizontal, site_n_horizontal*2, site_n_horizontal*3, site_n_horizontal*4, site_n_horizontal*5]
site_y_axis = [0, site_n_vertical, site_n_vertical*2, site_n_vertical*3, site_n_vertical*4, site_n_vertical*5]
viewport_center_width = viewport_width / 2
viewport_center_height = viewport_height / 2
```

Figura 29 Definición de Segmentos en el Sitio Web

El siguiente elemento trabajado fue el ajuste de la Matriz de Posición, en principio separadas. Para esto fue necesario realizar una comparación entre las coordenadas que definen los bloques en el sitio y el visualizador, esta modificación se realiza individualmente por cada una de las filas que contiene la matriz, como se muestra en la Figura 30:

```
# Fila 1
if site_y_axis[0] <= viewport_center_height < site_y_axis[1]:
    if site_x_axis[0] <= viewport_center_width < site_x_axis[1]:
        first_row = [2,1,0,0,0]
        second_row = [1,1,0,0,0]
    elif site_x_axis[1] <= viewport_center_width < site_x_axis[2]:
        first_row = [1,2,1,0,0]
        second_row = [1,1,1,0,0]
    elif site_x_axis[2] <= viewport_center_width < site_x_axis[3]:
        first_row = [0,1,2,1,0]
        second_row = [0,1,1,1,0]
    elif site_x_axis[3] <= viewport_center_width < site_x_axis[4]:
        first_row = [0,0,1,2,1]
        second_row = [0,0,1,1,1]
    elif site_x_axis[4] <= viewport_center_width <= site_x_axis[5]:
        first_row = [0,0,0,1,2]
        second_row = [0,0,0,1,1]
```

Figura 30 Modificación de filas en Matriz de Posición

Este proceso es repetido cinco (5) veces debido a que fue el **n** escogido durante el desarrollo del diseño.

Luego se realizó la obtención de las coordenadas X e Y del bloque (o de los bloques) donde se encuentren los cambios, esto se consigue mediante la búsqueda de elementos que contengan el estilo agregado durante la iteración previa, como representa la Figura 31:

```
url_updated = 'file://' + url2
browser.get(url_updated)
bloques_con_cambios = browser.find_elements_by_xpath('//*[@contains(@style, "border: 3px solid red")]')
```

Figura 31 Localización espacial de Bloques con Cambios

Posteriormente se calcula la importancia cualitativa, basando este proceso en la posición que tenga el bloque (o los bloques) que contenga(n) cambios junto con su anchura, asignando los valores de los índices de la Matriz de Posición entre los que está ubicado el bloque con el fin de obtener los elementos que serán sumados entre sí dentro de esta matriz, esto está ilustrado por la :

```
for i in range(0, len(bloques_con_cambios)):
    cambio_cualitativo = 0
    x1_value = bloques_con_cambios[i].location.get('x')
    x2_value = bloques_con_cambios[i].size.get('width') + x1_value

    if site_x_axis[0] <= x1_value < site_x_axis[1]: indice_x1 = 0
    elif site_x_axis[1] <= x1_value < site_x_axis[2]: indice_x1 = 1
    elif site_x_axis[2] <= x1_value < site_x_axis[3]: indice_x1 = 2
    elif site_x_axis[3] <= x1_value < site_x_axis[4]: indice_x1 = 3
    elif site_x_axis[4] <= x1_value < site_x_axis[5]: indice_x1 = 4

    if site_x_axis[0] <= x2_value < site_x_axis[1]: indice_x2 = 0
    elif site_x_axis[1] <= x2_value < site_x_axis[2]: indice_x2 = 1
    elif site_x_axis[2] <= x2_value < site_x_axis[3]: indice_x2 = 2
    elif site_x_axis[3] <= x2_value < site_x_axis[4]: indice_x2 = 3
    elif (site_x_axis[4] <= x2_value < site_x_axis[5]) or (x2_value >= site_x_axis[5]): indice_x2 = 4

    y1_value = bloques_con_cambios[i].location.get('y')
    y2_value = bloques_con_cambios[i].size.get('height') + y1_value

    if site_y_axis[0] <= y1_value < site_y_axis[1]: indice_y1 = 0
    elif site_y_axis[1] <= y1_value < site_y_axis[2]: indice_y1 = 1
    elif site_y_axis[2] <= y1_value < site_y_axis[3]: indice_y1 = 2
    elif site_y_axis[3] <= y1_value < site_y_axis[4]: indice_y1 = 3
    elif site_y_axis[4] <= y1_value < site_y_axis[5]: indice_y1 = 4

    if site_y_axis[0] <= y2_value < site_y_axis[1]: indice_y2 = 0
    elif site_y_axis[1] <= y2_value < site_y_axis[2]: indice_y2 = 1
    elif site_y_axis[2] <= y2_value < site_y_axis[3]: indice_y2 = 2
    elif site_y_axis[3] <= y2_value < site_y_axis[4]: indice_y2 = 3
    elif (site_y_axis[4] <= y2_value < site_y_axis[5]) or (y2_value >= site_y_axis[5]): indice_y2 = 4
```

Figura 32 Correlación de posición entre Bloques con Cambios y Matriz de Posición

La suma de los valores que estén correlacionados al bloque (o los bloques) con cambios, será su importancia cualitativa.

Finalmente, se modificó el trozo de código que calculaba la importancia creado durante la segunda iteración para que también se ponderase la importancia cualitativa en dicho proceso, en principio se le asignó un cuarenta por ciento (40%) del valor final a la importancia cuantitativa y el sesenta por ciento (60%) restante a la importancia cualitativa recién obtenida (este cálculo se sigue realizando para cada uno de los bloques que contengan cambios) y se agregó este resultado final también a la gráfica definida en la iteración previa, como se puede observar en la Figura 33. Estos valores se esperan ser modificados dependiendo de la data obtenida durante las pruebas funcionales de la aplicación.

```

quantitative_importance = [x * 0.4 for x in difference]
qualitative_importance = [x * 0.6 for x in qualitative_difference]
weighted_importance = [x + y for x, y in zip(quantitative_importance, qualitative_importance)]
y_pos = np.arange(len(objects))
performance = weighted_importance
plt.tick_params(axis='x', which='major', labelsize=5)
plt.figure(figsize=(10,3.8))
barlist = plt.bar(y_pos,performance,align='center',alpha=0.5)

for i in range(0, len(barlist)):
    height = barlist[i].get_height()
    if height >= 50:
        barlist[i].set_color('r')
    elif height >= 20 and height < 50:
        barlist[i].set_color('y')
    elif height > 0 and height < 20:
        barlist[i].set_color('g')

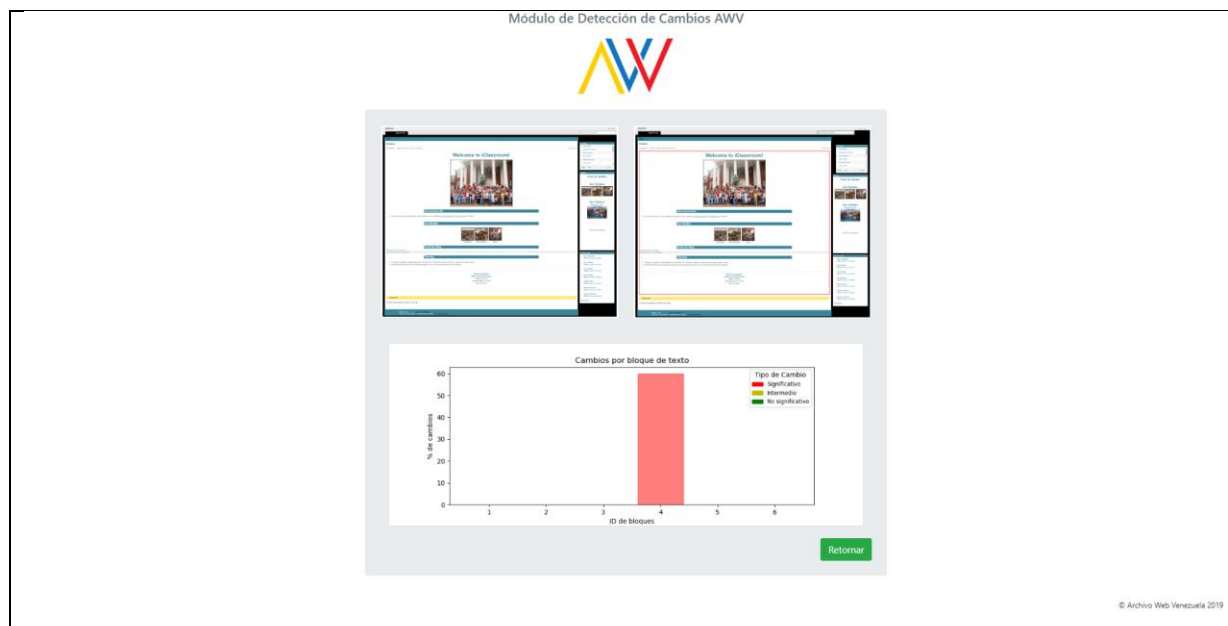
```

Figura 33 Ponderación de Importancias Cualitativa y Cuantitativa

4.5.4.4. Pruebas

Tabla 15 Caso de Prueba 1. Iteración 4

Caso de Prueba	
Número de Caso de Prueba: 1	Número de Historia de Usuario: 1
Descripción: Comprobar que el resultado ahora toma en cuenta la importancia cualitativa calculada durante esta iteración	
Resultado:	



4.5.5. Quinta Iteración (Pruebas funcionales del sistema):

Esta iteración tuvo una duración de cuatro (4) semanas, se realizaron las pruebas funcionales de la versión final de la aplicación, con el fin de verificar de que es una aplicación funcional, con tolerancia a fallos y que cumple con el comportamiento esperado para finalmente ponerla en el ambiente de producción acoplándola con el resto del Archivo Web de Venezuela.

4.5.5.1. Planeación

Se decidió aplicar la detección sobre cinco (5) tipos de sitios web, los cuales son **Blogs, Foros, Wikis, Sitios web Empresariales y Catálogos de imágenes**, con el fin de comprobar que, efectivamente, la aplicación es lo suficientemente genérica para procesar una alta variabilidad en el contenido que será analizado, así como también se decidió tomar varios sitios por cada uno de los tipos y analizarlas luego de que sufriesen diferentes tipos de cambio, las distintas pruebas a realizar fueron descritas en la Historias de Usuario de la siguiente manera:

Tabla 16 Historia de Usuario 1. Iteración 5

Historia de Usuario	
Número: 1	Usuario: Mercy Ospina
Nombre de Historia: Realización de pruebas funcionales de la Aplicación de Detección de Cambios.	
Prioridad (Alta/Media/Baja): Alta	Riesgo (Alto/Medio/Bajo): Alto

Puntos estimados: 5 (uno por cada tipo de sitio web)	Iteración asignada: 5
Programador responsable: Luis Aguiar	
Descripción: Tomando como base diversos sitios web usando como base los tipos acordados con antelación (Blogs, Foros, Sitios web Empresariales y Catálogos de Imágenes) realizar diferentes modificaciones a cada uno, incluyendo, pero no limitándose a: modificación de un bloque, modificación de más de un bloque, eliminación de un bloque, agregación de contenido. Luego proceder a aplicar la detección de cambios desarrollada a cada uno y realizar una observación de resultados, así como el apropiado ajuste de los valores de las importancias cualitativas o cuantitativas, de acuerdo con el caso estudiado.	
Validación: Se debe obtener el resultado esperado de acuerdo con las funcionalidades de la aplicación para todos los casos descritos en el punto anterior,	

4.5.5.2. Diseño

Se decidió mantener un diseño simple de los casos de prueba, tratando la aplicación como una caja negra, de manera similar a lo mostrado anteriormente en la Figura 16 y, de acuerdo con los resultados obtenidos se produjeron diversos análisis

4.5.5.3. Codificación

Para la realización de cada uno de los casos de prueba se realizó el almacenamiento de diversas páginas web obtenidas del repositorio ubicado en la dirección <http://bom.ciens.ucv.ve/dataset/> y, dependiendo de la prueba se modificó el código html de estos sitios de diversas maneras, ya sea eliminando bloques de texto, alterando el contenido que se muestra en texto plano, el código propiamente escrito de la página web o incluso no se realizó ningún cambio en lo absoluto con el fin de observar también el comportamiento de la aplicación en dicho entorno.

4.5.5.4. Pruebas

Los resultados de las pruebas se muestran a continuación.

Tabla 17 Caso de Prueba 1. Iteración 5

Caso de Prueba																																																										
Número de Caso de Prueba: 1	Número de Historia de Usuario: 1																																																									
Descripción: Prueba de Blogs, cambio en un bloque																																																										
Resultado: Resultado obtenido, de acuerdo con el resultado que se esperaba de este caso de prueba.																																																										
<p>Módulo de Detección de Cambios AWW</p> 																																																										
<div style="display: flex; justify-content: space-around;">   </div>																																																										
<p style="text-align: center;">Cambios por bloque de texto</p>  <table border="1" style="margin-top: 10px; width: 100%; border-collapse: collapse;"> <caption>Datos del Gráfico: Cambios por bloque de texto</caption> <thead> <tr> <th>ID de bloques</th> <th>Tipo de Cambio</th> <th>% de cambios</th> </tr> </thead> <tbody> <tr> <td>1</td> <td>No significativo</td> <td>0.000</td> </tr> <tr> <td>2</td> <td>No significativo</td> <td>0.000</td> </tr> <tr> <td>3</td> <td>No significativo</td> <td>0.000</td> </tr> <tr> <td>4</td> <td>No significativo</td> <td>0.000</td> </tr> <tr> <td>5</td> <td>No significativo</td> <td>0.000</td> </tr> <tr> <td>6</td> <td>No significativo</td> <td>0.000</td> </tr> <tr> <td>7</td> <td>No significativo</td> <td>0.000</td> </tr> <tr> <td>8</td> <td>No significativo</td> <td>0.040</td> </tr> <tr> <td>9</td> <td>No significativo</td> <td>0.000</td> </tr> <tr> <td>10</td> <td>No significativo</td> <td>0.000</td> </tr> <tr> <td>11</td> <td>No significativo</td> <td>0.000</td> </tr> <tr> <td>12</td> <td>No significativo</td> <td>0.000</td> </tr> <tr> <td>13</td> <td>No significativo</td> <td>0.000</td> </tr> <tr> <td>14</td> <td>No significativo</td> <td>0.000</td> </tr> <tr> <td>15</td> <td>No significativo</td> <td>0.000</td> </tr> <tr> <td>16</td> <td>No significativo</td> <td>0.000</td> </tr> <tr> <td>17</td> <td>No significativo</td> <td>0.000</td> </tr> <tr> <td>18</td> <td>No significativo</td> <td>0.000</td> </tr> </tbody> </table>		ID de bloques	Tipo de Cambio	% de cambios	1	No significativo	0.000	2	No significativo	0.000	3	No significativo	0.000	4	No significativo	0.000	5	No significativo	0.000	6	No significativo	0.000	7	No significativo	0.000	8	No significativo	0.040	9	No significativo	0.000	10	No significativo	0.000	11	No significativo	0.000	12	No significativo	0.000	13	No significativo	0.000	14	No significativo	0.000	15	No significativo	0.000	16	No significativo	0.000	17	No significativo	0.000	18	No significativo	0.000
ID de bloques	Tipo de Cambio	% de cambios																																																								
1	No significativo	0.000																																																								
2	No significativo	0.000																																																								
3	No significativo	0.000																																																								
4	No significativo	0.000																																																								
5	No significativo	0.000																																																								
6	No significativo	0.000																																																								
7	No significativo	0.000																																																								
8	No significativo	0.040																																																								
9	No significativo	0.000																																																								
10	No significativo	0.000																																																								
11	No significativo	0.000																																																								
12	No significativo	0.000																																																								
13	No significativo	0.000																																																								
14	No significativo	0.000																																																								
15	No significativo	0.000																																																								
16	No significativo	0.000																																																								
17	No significativo	0.000																																																								
18	No significativo	0.000																																																								
Retornar																																																										

Tabla 18 Caso de Prueba 2. Iteración 5

Caso de Prueba

Número de Caso de Prueba: 2

Número de Historia de Usuario: 1

Descripción: Prueba de Blogs, agregación y eliminación de un bloque.

Resultado: Debido a las limitaciones de la aplicación, sólo es posible dibujar el marco del primer bloque que haya cambiado, limitando la usabilidad del cálculo cualitativo, esto se da debido a que se colocó un límite de un elemento a remarcar ya que algún número mayor dibujaría tantos cuadros que se dificultaría la visualización porque la función compara los cambios de manera recursiva y de haber trabajado de otra manera, se habrían remarcado no sólo el bloque que contiene el cambio sino cada elemento de su contenido

Módulo de Detección de Cambios AWV

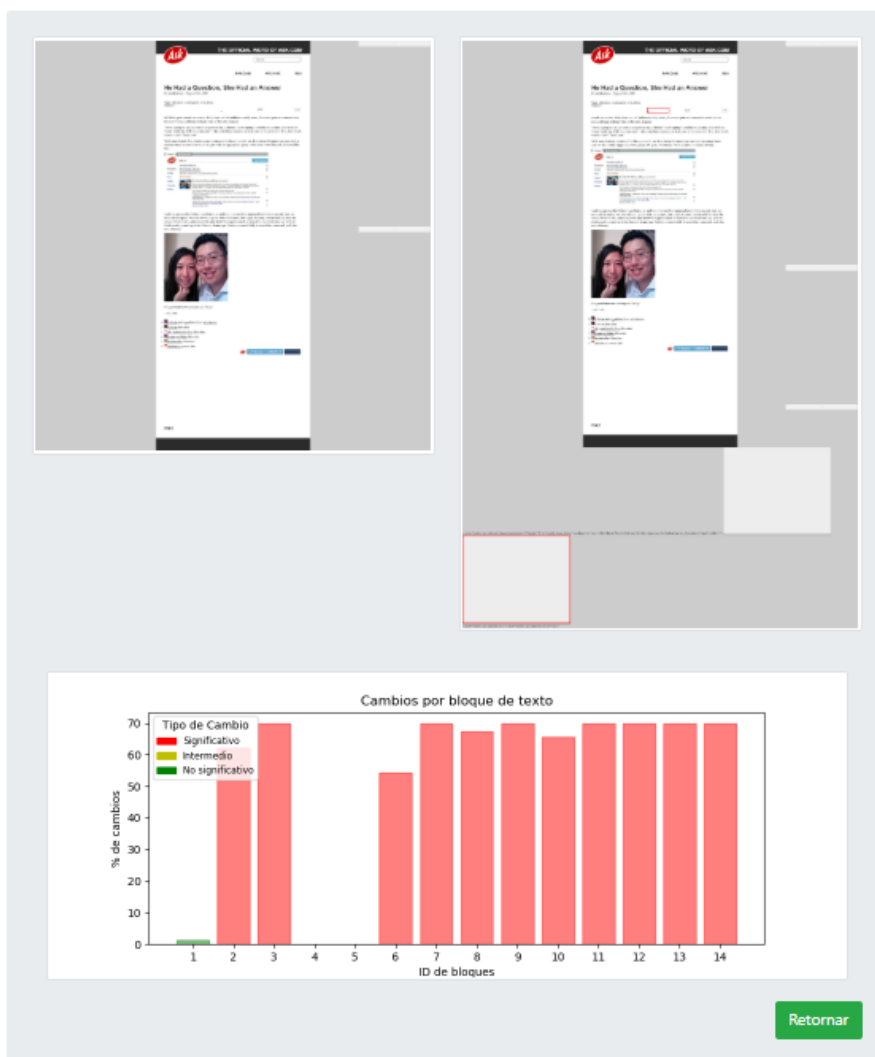


Tabla 19 Caso de Prueba 3. Iteración 5

Caso de Prueba																
Número de Caso de Prueba: 3	Número de Historia de Usuario: 1															
Descripción: Prueba de sitio de Empresas, no hay cambios																
<p>Resultado: El resultado obtenido está acorde con el resultado que se esperaba de este caso de prueba.</p> <p style="text-align: center;">Módulo de Detección de Cambios AWW</p>  <div style="display: flex; justify-content: space-around; margin-top: 20px;">   </div> <div style="text-align: center; margin-top: 20px;"> <p>Cambios por bloque de texto</p>  <table border="1" style="margin-top: 10px; width: 100%; border-collapse: collapse;"> <caption>Cambios por bloque de texto</caption> <thead> <tr> <th>ID de bloques</th> <th>% de cambios</th> <th>Tipo de Cambio</th> </tr> </thead> <tbody> <tr> <td>1</td> <td>0.00</td> <td>No significativo</td> </tr> <tr> <td>2</td> <td>0.00</td> <td>No significativo</td> </tr> <tr> <td>3</td> <td>0.00</td> <td>No significativo</td> </tr> <tr> <td>4</td> <td>0.00</td> <td>No significativo</td> </tr> </tbody> </table> </div> <div style="text-align: right; margin-top: 10px;"> Retornar </div>		ID de bloques	% de cambios	Tipo de Cambio	1	0.00	No significativo	2	0.00	No significativo	3	0.00	No significativo	4	0.00	No significativo
ID de bloques	% de cambios	Tipo de Cambio														
1	0.00	No significativo														
2	0.00	No significativo														
3	0.00	No significativo														
4	0.00	No significativo														

Tabla 20 Caso de Prueba 4. Iteración 5

Caso de Prueba																																		
Número de Caso de Prueba: 4	Número de Historia de Usuario: 1																																	
Descripción: Prueba de sitio de Empresas, eliminación de un bloque																																		
Resultado: El resultado obtenido está acorde con el resultado que se esperaba de este caso de prueba.																																		
<p>Módulo de Detección de Cambios AWW</p> 																																		
																																		
 <table border="1" style="margin-top: 10px; width: 100%; border-collapse: collapse;"> <caption>Cambios por bloque de texto</caption> <thead> <tr> <th>ID de bloques</th> <th>Tipo de Cambio</th> <th>% de cambios</th> </tr> </thead> <tbody> <tr> <td>1</td> <td>Significativo</td> <td>~68%</td> </tr> <tr> <td>2</td> <td>No significativo</td> <td>0%</td> </tr> <tr> <td>3</td> <td>No significativo</td> <td>0%</td> </tr> <tr> <td>4</td> <td>No significativo</td> <td>0%</td> </tr> <tr> <td>5</td> <td>No significativo</td> <td>0%</td> </tr> <tr> <td>6</td> <td>No significativo</td> <td>0%</td> </tr> <tr> <td>7</td> <td>No significativo</td> <td>0%</td> </tr> <tr> <td>8</td> <td>No significativo</td> <td>0%</td> </tr> <tr> <td>9</td> <td>No significativo</td> <td>0%</td> </tr> <tr> <td>10</td> <td>No significativo</td> <td>0%</td> </tr> </tbody> </table>		ID de bloques	Tipo de Cambio	% de cambios	1	Significativo	~68%	2	No significativo	0%	3	No significativo	0%	4	No significativo	0%	5	No significativo	0%	6	No significativo	0%	7	No significativo	0%	8	No significativo	0%	9	No significativo	0%	10	No significativo	0%
ID de bloques	Tipo de Cambio	% de cambios																																
1	Significativo	~68%																																
2	No significativo	0%																																
3	No significativo	0%																																
4	No significativo	0%																																
5	No significativo	0%																																
6	No significativo	0%																																
7	No significativo	0%																																
8	No significativo	0%																																
9	No significativo	0%																																
10	No significativo	0%																																
Retornar																																		

Tabla 21 Caso de Prueba 5. Iteración 5

Caso de Prueba	
Número de Caso de Prueba: 5	Número de Historia de Usuario: 1
Descripción: Prueba de Foro, sin cambios.	
Resultado: El resultado obtenido está acorde con el resultado que se esperaba de este caso de prueba.	

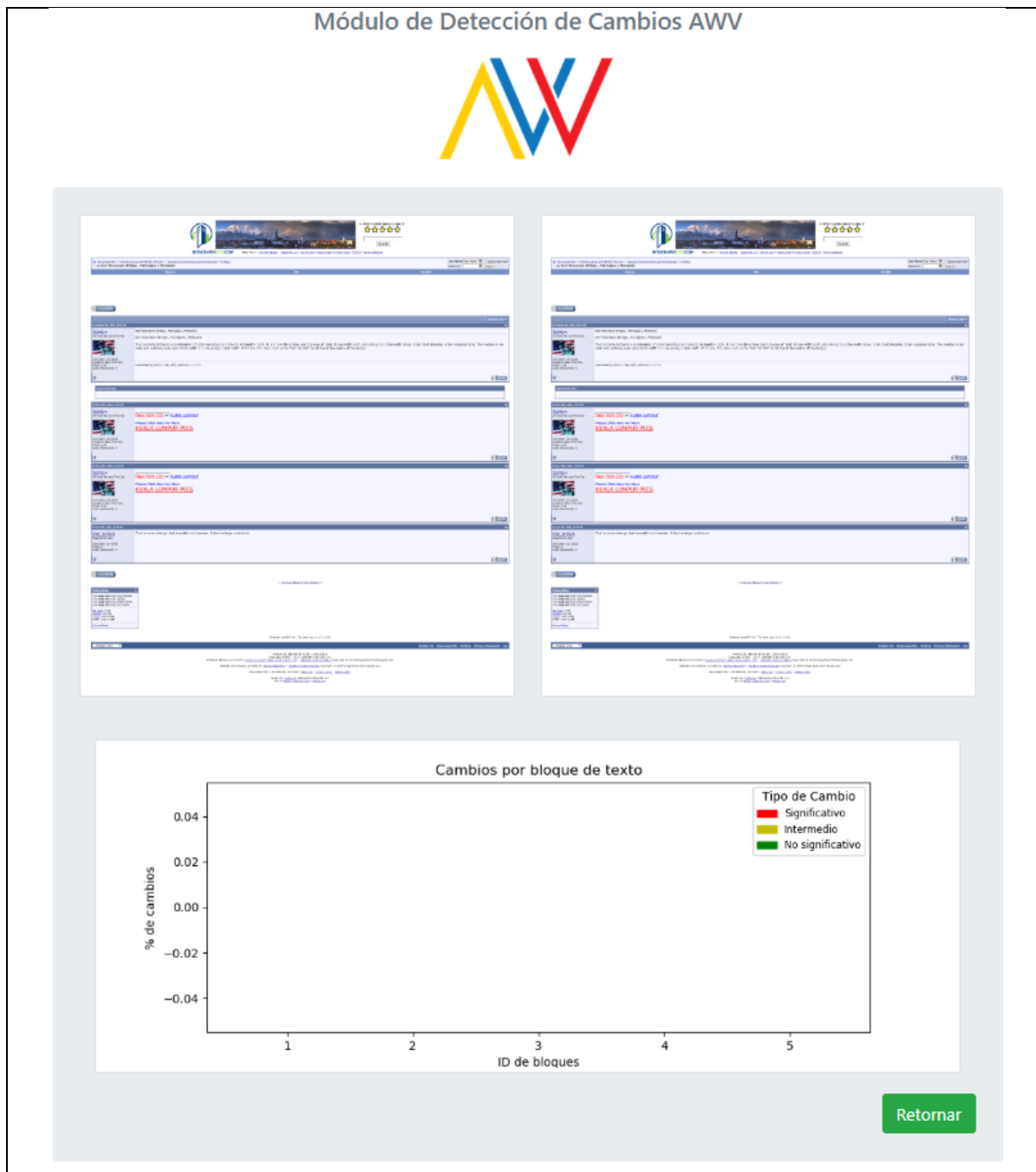


Tabla 22 Caso de Prueba 6. Iteración 5

Caso de Prueba	
Número de Caso de Prueba: 6	Número de Historia de Usuario: 1
Descripción: Prueba de Wiki número 1, cambio mínimo en un bloque de contenido.	

Resultado: El resultado obtenido está acorde con el resultado que se esperaba de este caso de prueba.

Módulo de Detección de Cambios AWV

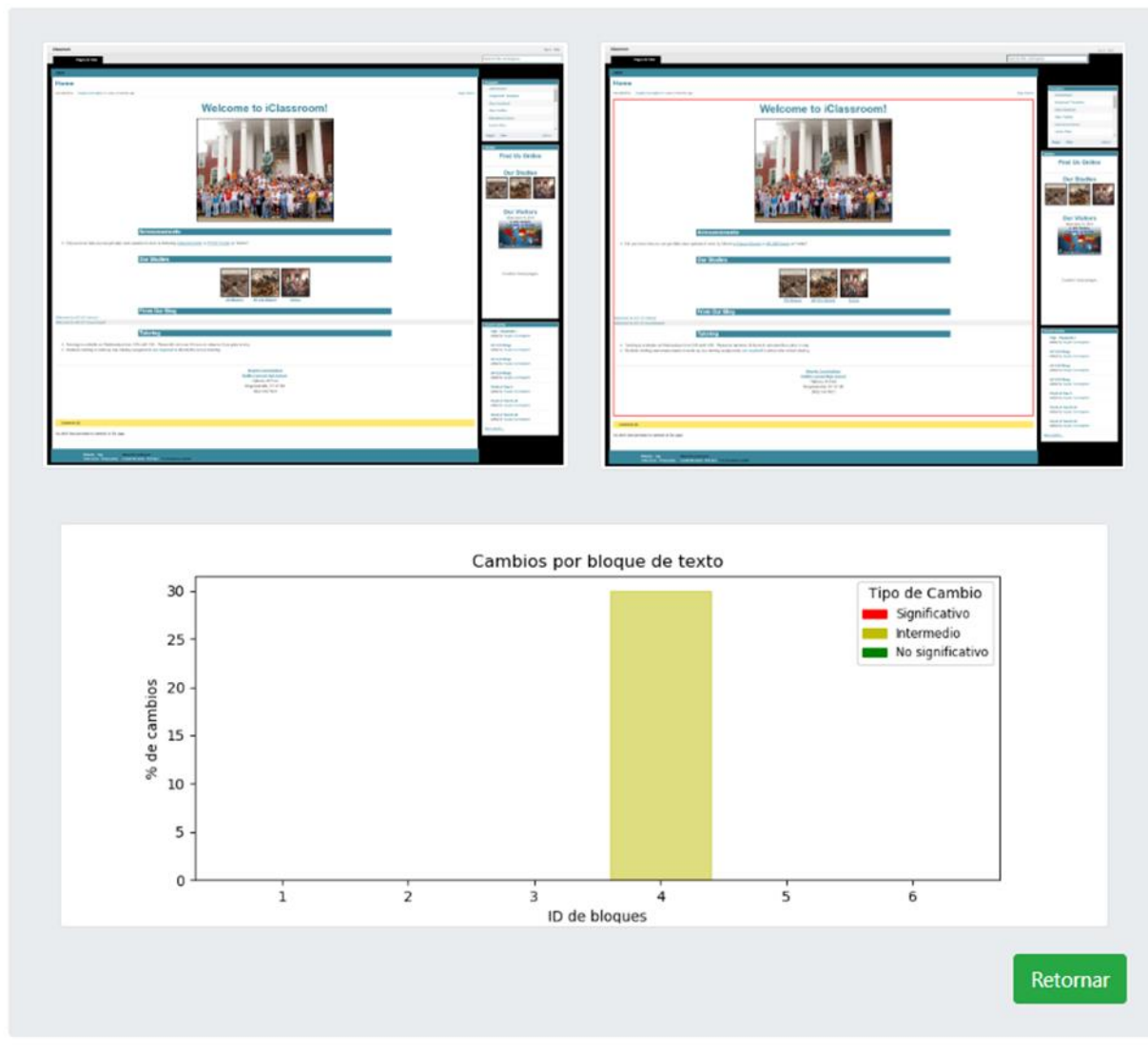


Tabla 23 Caso de Prueba 7. Iteración 5

Caso de Prueba	
Número de Caso de Prueba: 7	Número de Historia de Usuario: 1

Descripción: Prueba de Wiki, cambios en más de un bloque sin eliminación

Resultado: Similar a los casos 1 y 2, ya que sólo es posible dibujar el marco del primer bloque que haya cambiado, no se puede realizar la comparación de las listas de cambio cuantitativo y cambio cualitativo por tener diferente cantidad de elementos.

Módulo de Detección de Cambios AWW



Tabla 24 Caso de Prueba 8. Iteración 5

Caso de Prueba

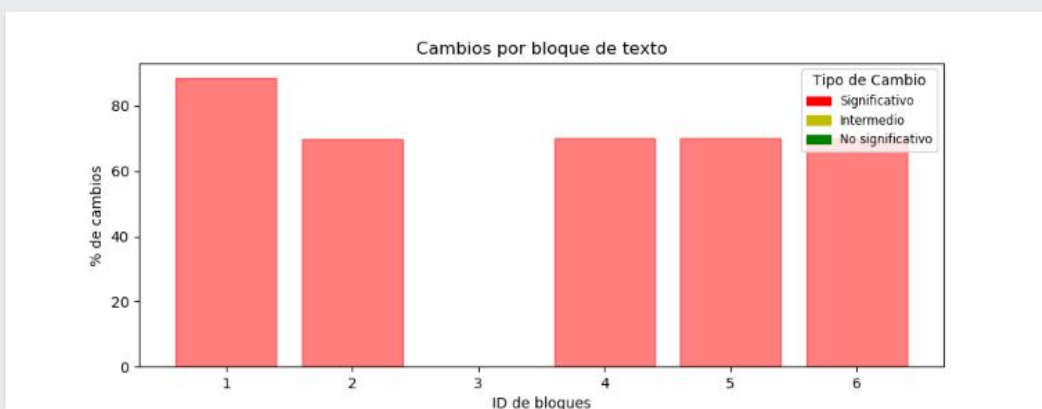
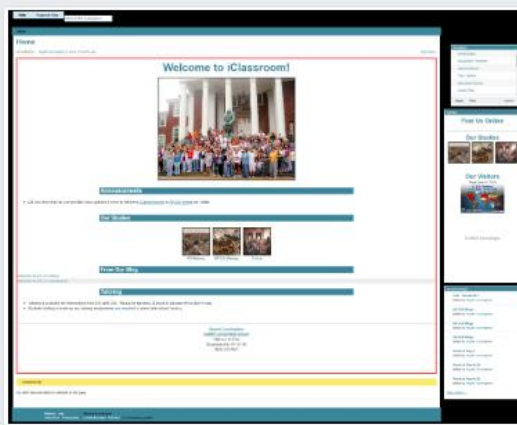
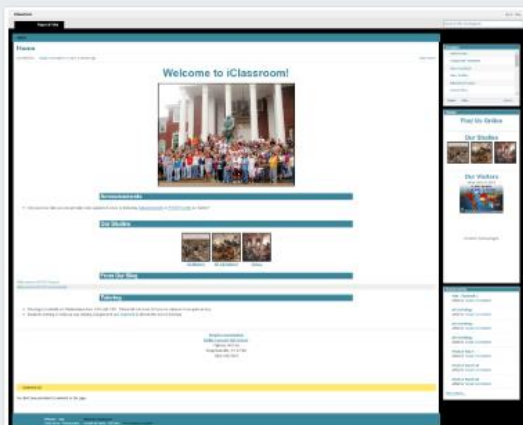
Número de Caso de Prueba: 8

Número de Historia de Usuario: 1

Descripción: Prueba de Wiki número 3, eliminación y cambios de bloques de contenido.

Resultado: Al ser eliminado un bloque, todo el sitio web sufre modificaciones de acuerdo al nivel actual de granularidad. Esto coincide con el resultado esperado.

Módulo de Detección de Cambios AWW




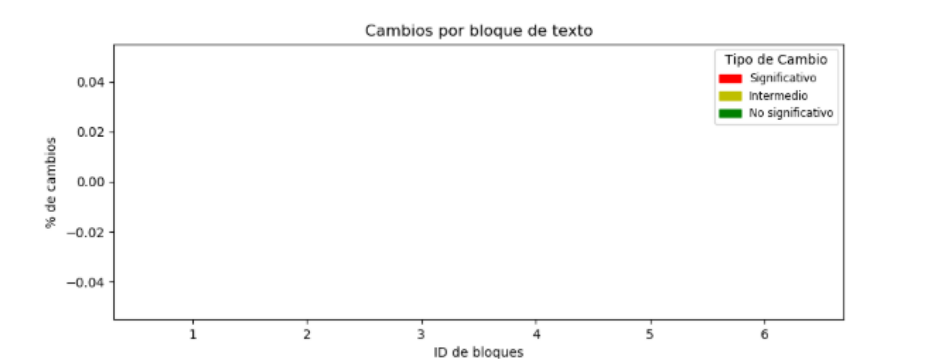


Retornar

Tabla 25 Caso de Prueba 9. Iteración 5

Caso de Prueba																						
Número de Caso de Prueba: 9	Número de Historia de Usuario: 1																					
Descripción: Prueba de Catálogo número 1, eliminación de una imagen.																						
Resultado: El resultado obtenido está acorde con el resultado que se esperaba de este caso de prueba.																						
<p>Módulo de Detección de Cambios AWW</p> 																						
<div style="display: flex; justify-content: space-around;"> <div style="border: 1px solid #ccc; padding: 5px; width: 45%;">  </div> <div style="border: 1px solid #ccc; padding: 5px; width: 45%;">  </div> </div>																						
<div style="border: 1px solid #ccc; padding: 10px;"> <p style="text-align: center;">Cambios por bloque de texto</p>  <table border="1" style="margin-top: 10px; width: 100%; border-collapse: collapse;"> <caption>Datos del gráfico de barras</caption> <thead> <tr> <th>ID de bloques</th> <th>% de cambios</th> <th>Tipo de Cambio</th> </tr> </thead> <tbody> <tr> <td>1</td> <td>0.00</td> <td>No significativo</td> </tr> <tr> <td>2</td> <td>0.45</td> <td>No significativo</td> </tr> <tr> <td>3</td> <td>0.00</td> <td>No significativo</td> </tr> <tr> <td>4</td> <td>0.00</td> <td>No significativo</td> </tr> <tr> <td>5</td> <td>0.00</td> <td>No significativo</td> </tr> <tr> <td>6</td> <td>0.00</td> <td>No significativo</td> </tr> </tbody> </table> </div>		ID de bloques	% de cambios	Tipo de Cambio	1	0.00	No significativo	2	0.45	No significativo	3	0.00	No significativo	4	0.00	No significativo	5	0.00	No significativo	6	0.00	No significativo
ID de bloques	% de cambios	Tipo de Cambio																				
1	0.00	No significativo																				
2	0.45	No significativo																				
3	0.00	No significativo																				
4	0.00	No significativo																				
5	0.00	No significativo																				
6	0.00	No significativo																				
Retornar																						

Tabla 26 Caso de Prueba 10. Iteración 5

Caso de Prueba																						
Número de Caso de Prueba: 10	Número de Historia de Usuario: 1																					
Descripción: Prueba de Catálogo número 3, sin cambios.																						
Resultado: El resultado obtenido está acorde con el resultado que se esperaba de este caso de prueba.																						
<p>Módulo de Detección de Cambios AWW</p>  <div style="display: flex; justify-content: space-around; align-items: flex-start; padding: 20px;"> <div style="width: 45%;">  </div> <div style="width: 45%;">  </div> </div> <div style="text-align: center; margin-top: 20px;"> <p>Cambios por bloque de texto</p>  <table border="1" style="margin-top: 10px; width: 100%; border-collapse: collapse;"> <caption>Datos del gráfico: Cambios por bloque de texto</caption> <thead> <tr> <th>ID de bloques</th> <th>Tipo de Cambio</th> <th>% de cambios</th> </tr> </thead> <tbody> <tr><td>1</td><td>No significativo</td><td>0.00</td></tr> <tr><td>2</td><td>No significativo</td><td>0.00</td></tr> <tr><td>3</td><td>No significativo</td><td>0.00</td></tr> <tr><td>4</td><td>No significativo</td><td>0.00</td></tr> <tr><td>5</td><td>No significativo</td><td>0.00</td></tr> <tr><td>6</td><td>No significativo</td><td>0.00</td></tr> </tbody> </table> </div> <div style="text-align: right; margin-top: 10px;"> Retornar </div>		ID de bloques	Tipo de Cambio	% de cambios	1	No significativo	0.00	2	No significativo	0.00	3	No significativo	0.00	4	No significativo	0.00	5	No significativo	0.00	6	No significativo	0.00
ID de bloques	Tipo de Cambio	% de cambios																				
1	No significativo	0.00																				
2	No significativo	0.00																				
3	No significativo	0.00																				
4	No significativo	0.00																				
5	No significativo	0.00																				
6	No significativo	0.00																				

CONCLUSIONES Y RECOMENDACIONES

Al culminar este Trabajo Especial de Grado (T.E.G.), se desarrolló una solución de Cálculo de Importancia para la Detección de Cambios en Páginas Web, basada tanto en texto como en la ubicación espacial del elemento que fue modificado (este último aspecto condicionado a la cantidad de bloques que sufrieron cambios), luego de analizar las herramientas disponibles en el mercado para poder construir la aplicación acorde a las necesidades de los usuarios del Prototipo del Archivo Web de Venezuela.

Debido al uso de elementos no tradicionales encontrados durante el desarrollo y el tamaño del equipo de trabajo, fue necesario realizar una adaptación de la metodología eXtreme Programming (o por sus siglas XP), la cual permitió establecer una serie de actividades que sirvieron como guía para la construcción de los componentes que forman parte de la aplicación. De esta manera se pudo avanzar en el desarrollo de forma rápida y ordenada, lo que dio como resultado un producto usable y que satisface las necesidades definidas por los usuarios, además de hacer al Prototipo de Archivo Web un sistema de tecnología innovador, debido a que, a la fecha, no se encontró información de otros Archivos Web que combinen técnicas de detección basadas tanto en el texto como en la ubicación del bloque posterior al renderizado del sitio web.

De acuerdo con los resultados obtenidos al finalizar el desarrollo del presente T.E.G. es posible determinar que se cumplieron los objetivos de acuerdo al alcance establecido, debido a que se logró el desarrollo de la aplicación de detección de cambios, sin embargo, ésta muestra un alto margen de error al momento de calcular la importancia de los cambios en los casos estudiados cuando el cambio se analiza de manera visual, el problema observado es que mostrar los cambios en bloques de textos posterior al primero le daría poca información al usuario, por lo que se programó una limitación necesaria para la presentación del presente trabajo, de otra manera, cada elemento que hubiese cambiado sería enmarcado lo cual hubiese complicado la visualización durante la toma de la captura de pantalla, esto se da ya que, si un bloque es modificado de alguna manera, eliminado o un nuevo bloque es agregado, se determina que los bloques subsiguientes son partes del cambio.

A pesar de esta limitación, el desarrollo de la interfaz se realizó de manera exitosa de acuerdo con el objetivo planteado, facilitando el uso de la aplicación y el análisis de la información adquirida con ésta por parte del usuario final.

De acuerdo con los resultados obtenidos se estableció una clasificación para precisar la importancia de cada cambio, determinando una ponderación para estos dependiendo de la ubicación dónde estuviese ubicado en el visualizador y el contenido de este.

Esta solución ofrece a los usuarios una serie de indicadores que ayudarán al Prototipo a tomar decisiones basadas en la importancia de las modificaciones realizadas sobre los sitios web de manera automatizada, asistiendo así a la optimización del almacenamiento de nuevas versiones. Actualmente el análisis manual toma entre 30 segundos hasta 4 minutos, sin tomar en cuenta el tiempo de carga de ambos sitios web.

Esta variación se da de acuerdo con el tipo de cambio realizado y del tamaño de la página. La aplicación de detección de cambios desarrollada logra realizar una detección prácticamente instantánea luego de la realización de las renderizaciones de ambas versiones del sitio web.

Durante el desarrollo de la solución se presentaron diversas limitantes, siendo la principal la relativamente escasa documentación sobre algunas de las técnicas de análisis visual debido a que los dueños de las patentes decidieron bloquear el acceso a su documentación por la duración de éstas, específicamente VIPS que es propiedad de Microsoft.

A través de pruebas con el cliente realizadas al final de cada una de las iteraciones, se obtuvieron diversos comentarios y sugerencias sobre la aplicación, las cuales fueron tomadas en cuenta por el equipo de trabajo con el fin de obtener un software de calidad y alta funcionalidad que agregase algo de valor al Prototipo de Archivo Web.

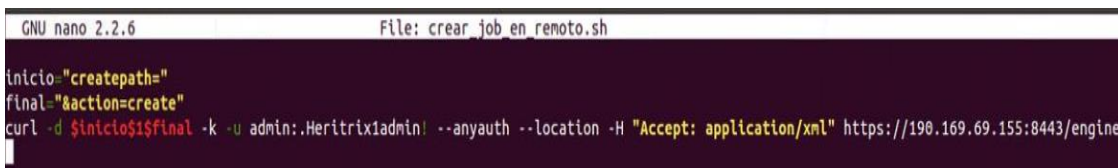
Las limitaciones de la aplicación, junto con otras ideas presentadas por el equipo de trabajo de la Aplicación de Detección de Cambios, son elementos que recomendamos realizar en trabajos futuros para la mejora de la aplicación, éstas son:

- Refinamiento de la aplicación para que se logre el enmarcado visual de los bloques que presentan cambios (si una interfaz no es requerida, esto no supondría ningún problema).
- Creación de una aplicación de predicción que use los datos provistos como resultados de este trabajo para facilitar la automatización de los rastreos hechos por el módulo correspondiente.
- Utilización de herramientas de virtualización como Docker para convertir al Prototipo de Archivo Web en una aplicación modularizada y completamente portable.
- Realizar la integración de la Aplicación de Detección de Cambios con el Módulo de Adquisición del Prototipo de Archivo Web de Venezuela.

ANEXOS

5.1.Scripts de Heritrix

Las siguientes figuras muestran los principales scripts que deben ser ejecutados para realizar un rastreo. Primero, se ejecuta el shell script para crear el trabajo en el servidor remoto. Este script crea un nuevo trabajo indicando qué máquina virtual será utilizada.



```
GNU nano 2.2.6 File: crear_job_en_remoto.sh
inicio="createpath="
final="&action=create"
curl -d $inicio$final -k -u admin:Heritrix1admin! --anyauth --location -H "Accept: application/xml" https://190.169.69.155:8443/engine
```

FIGURA 34 SCRIPT CREAR_JOB_EN_REMOTO.SH (Carreño, 2016)

Luego, se copia el archivo de configuración de Heritrix y el archivo de semilla (seeds en adelante) en el directorio del rastreo. Toda la salida del rastreo será temporalmente almacenada en este directorio. El siguiente paso es agregar el nuevo trabajo de rastreo al archivo crontab, que contiene la lista de trabajos agendados, o cron jobs, del sistema. Esto se hace con el script “agregar_cron”. De esta manera, el rastreo será ejecutado en una fecha determinada, con la finalidad de capturar cualquier nuevo cambio realizado en la página desde el último rastreo.



```
GNU nano 2.2.6 File: mover_archivo.sh
#!/bin/bash
cp /home/crawler-beans.cxml /home/allData/$1
cp /home/seeds.txt /home/allData/$1
```

FIGURA 35 SCRIPT MOVER_ARCHIVO.SH (Carreño, 2016)

Luego, se ejecuta el rastreo. Este script se encarga de invocar los comandos de ejecución y posterior almacenamiento de la salida del rastreo. Con esto, concluye el proceso de crear un nuevo rastreo.

```

GNU nano 2.2.6 File: ejecutar rastreo.sh
#!/bin/sh
HERITRIX='/home/allData/'
ultima_version=$(sh /home/ultima_version.sh $1)
echo $ultima_version

#Se comprueba que el job exista
if [ $ultima_version != "no_existe_job" ];
then
#Verificar que no exista una version del rastreo en ejecucion
ejecuta="false"
if [ $ultima_version = "primera_vez" ]
then
#Es la primera vez que se quiere correr el rastreo, por $
ejecuta="true"
else
job_log=$HERITRIX$1/"$ultima_version"/reports"
if test -e $job_log ;

```

FIGURA 36 SCRIPT MOVER_ARCHIVO.SH (Carreño, 2016)

Si bien estos scripts permiten utilizar Heritrix hay una serie de inconvenientes con su ejecución.

Para comenzar, los scripts deben ser ejecutados por un usuario. Esto hace que la responsabilidad de detectar cambios caiga sobre el administrador del sitio y no sobre el sistema, incrementando la probabilidad de que un error humano interfiera con el buen funcionamiento del archivo. Un ejemplo de esto sería la omisión de cambios importantes por no realizar rastreos en el momento oportuno.

Otra falla de este sistema se aprecia en el momento en el que se ejecuta el rastreo. La ejecución de los scripts no da información al usuario ni al sistema con respecto a la ejecución del rastreo. Es imposible para el administrador saber cuándo un script comenzará, a menos que revise la tabla cron (crontab). Tampoco se puede saber el estado del rastreo y qué causó que terminara su ejecución. Si el rastreo termina, podrá verse en la tabla cron haciendo uso de otro script (verificar_finalizados.sh). El problema es, que este archivo sólo informa de la finalización de la ejecución, no del estado de esta. Si se presenta un error que interrumpió el rastreo, el administrador no será notificado y no podrá reprogramar el rastreo comprometiendo así la misión del sistema de resguardar los cambios del sitio rastreado.

Finalmente, desde el punto de vista de UI/ UX, ejecutar tareas importantes a través de un conjunto de scripts no es aceptable. Esto limita el sistema a personas con cierto conocimiento tecnológico y deja la detección y recuperación de errores completamente en manos del usuario, faltando así a varios principios básicos del diseño de UI.

Otro componente fundamental del módulo de adquisición es el predictor de cambios. La aplicación debe ser capaz de manejar de manera autónoma los rastreos de los sitios web que se están archivando. Rastreos muy frecuentes pueden ocasionar un volumen de datos duplicados que sobrecargue rápidamente la capacidad de

almacenamiento del sistema. Rastros muy espaciados en el tiempo conllevan a pérdidas de contenido y, por lo tanto, a fallar en la misión del sistema de preservar un sitio web.

REFERENCIAS BIBLIOGRÁFICAS Y ELECTRÓNICAS

- Agile Alliance. (29 de Agosto de 2019). *Extreme Programming*. Obtenido de What is Extreme Programming (XP)? - Agile Alliance: [agilealliance.org/glossary/xp/#q=~\(infinite~false~filters~\(postType~\(~'post~'aa_book~'aa_event_session~'aa_experience_report~'aa_glossary~'aa_research_paper~'aa_video\)~tags~\(~'xp\)\)~searchTerm~'~sort~false~sortDirection~'asc~page~1\)](http://agilealliance.org/glossary/xp/#q=~(infinite~false~filters~(postType~(~'post~'aa_book~'aa_event_session~'aa_experience_report~'aa_glossary~'aa_research_paper~'aa_video)~tags~(~'xp))~searchTerm~'~sort~false~sortDirection~'asc~page~1))
- Akunda, K. (19 de Octubre de 2011). *The Application of the Pareto Principle in Software Engineering*. Ruston, Luisiana, Estados Unidos de América.
- Archive-it. (2006). *Home: Archive-it*. Recuperado el 2012, de Archive-it Web site: <http://www.archive-it.org/>
- Arsanjani, A., Thomas, E., McKendrick, J., & Otros. (2013). *SOA Manifesto*. Obtenido de SOA Manifesto: <http://www.soa-manifesto.org/>
- Bayer, F., & Harald, K. (2013). *Prozessmanagement für Experten - Impulse für aktuelle und wiederkehrende Themen*.
- Beck, K. (1999). *Extreme Programming Explained: Embrace Change*. Boston: Addison-Wesley Longman Publishing Co., Inc.
- Beck, K., Beedle, M., van Bennekum, A., Cockburn, A., Cunningham, W., Fowler, M., . . . Thomas, D. (2001). *Principles behind the Agile Manifesto*. Obtenido de <http://agilemanifesto.org/principles.html>
- Boehm, B. (1986). *A Spiral Model of Software Development and Enhancement*. Southern California, United States of America.
- Burner, M., & Kahle, B. (15 de Sep de 1996). *Archivos WARC*. Obtenido de Web Archive Jira: <https://webarchive.jira.com/wiki/spaces/Heritrix/pages/459199/ARC+File+Format>
- Calzarossa, M., & Daniele, T. (2015). Modeling and predicting temporal patterns of web content changes. *Journal of Network and Computer Applications*.
- Carabali, W., & Casanova, M. (2014). *COMPARACIÓN DE TÉCNICAS DE PREDICCIÓN PARA EL DESARROLLO DE UN COMPONENTE DE PREDICCIÓN DE CAMBIOS DE SITIOS WEB PARA UN PROTOTIPO DE ARCHIVO WEB*. Caracas: UCV.
- Carabali, W., & Casanova, M. (2014). *Desarrollo e implementación del módulo de predicción de cambios de sitios web para un prototipo de Archivo Web*. Caracas.
- Carreño, A. (2016). *Información de los shell script para ejecutar un rastreo nuevo*. Caracas.
- Club BPM. (3 de November de 2009). *Club BPM*. Obtenido de Club BPM: <http://www.club-bpm.com/ApuntesBPM/ApuntesBPM01.pdf>

- CONCISA. (2014). *Memorias de la Segunda Conferencia Nacional de Computación*. Universidad Católica Andrés Bello, Caracas - Venezuela.
- Cunningham, W. (26 de Diciembre de 2014). *Model View Controller History*. Obtenido de <http://wiki.c2.com/?ModelViewControllerHistory>
- Da Silva, M., & Matos, A. (2017). *Estudio de una solución basada en Gestión de Procesos de Negocio que permita visualizar y orquestar los módulos del Sistema de Preservación de Archivos Web*. Caracas: UCV.
- Diazaraque, J. (08 de Marzo de 2001). *Series Temporales*. Obtenido de <http://halweb.uc3m.es/esp/Personal/personas/jmmarin/esp/EDescrip/tema7.pdf>
- Django Software Foundation. (2012). *Django Project*. Recuperado el Diciembre de 2012, de <https://docs.djangoproject.com/en/1.4/>
- Django Software Foundation. (2017). *FAQ: General*. Obtenido de Django Project: <https://docs.djangoproject.com/en/dev/faq/general/#what-does-django-mean-and-how-do-you-pronounce-it>
- Django Software Foundation. (2017). *Design Philosophies*. Obtenido de Django Project: <https://docs.djangoproject.com/en/1.11/misc/design-philosophies/>
- Docker Inc. (2018). *Docker Documentation*. Obtenido de Docker Docs: <https://docs.docker.com/>
- Dumas, M., Marcello, L. R., Jan, M., & Reijers, H. (2013). *Fundamentals of Business Process Management*. Springer.
- Endrei et al. (2004). *Patterns: Service-Oriented Architecture and Web*.
- Fielding, R. T. (2000). *Architectural Styles and*. Obtenido de www.ics.uci.edu: https://www.ics.uci.edu/~fielding/pubs/dissertation/rest_arch_style.htm
- Galvis-Lista, E., González-Zabala, M. P., & Vera, P. (December de 2011). *Tecnologías de información para acercar al ciudadano a los servicios de justicia en Colombia: El caso del mapa de oferta de justicia*. Obtenido de ResearchGate: https://www.researchgate.net/publication/305301468_Tecnologias_de_informacion_para_acercar_al_ciudadano_a_los_servicios_de_justicia_en_Colombia_El_caso_del_mapa_de_oferta_de_justicia
- Gamma, E., Helm, R., Johnson, R., & Vlissides, J. (1994). *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison-Wesley.
- García, J., & Rivero, L. (2013). *Desarrollo de prototipo para el Archivo Web de Venezuela. (Tesis de pregrado)*. Universidad Central de Venezuela, Caracas - Venezuela.
- Garimella, K., Lees, M., & Williams, B. (2008). *Introducción a BPM para Dummies*. Indianapolis: Wiley Publishing.
- George, N. (2017). *The Model-View-Controller Design Pattern*. Obtenido de The Django Book: <https://djangobook.com/model-view-controller-design-pattern/>

- Gutiérrez Gómez, I., & Otón Tortosa, S. (s.f.). *Arquitecturas Orientadas a Servicios. Arquitecturas Orientadas a Servicios*. Alcalá, España: Universidad de Alcalá, Departamento de Ciencias de la Computación.
- Haight, F. (1967). *Handbook of the Poisson Distribution*. New York: John Wiley & Sons.
- Hamilton Institute. (10 de Oct de 2010). *The Binomial Distribution*. Obtenido de Hamilton Institute: <http://www.hamilton.ie/oilie/EE304/Binom.pdf>
- Heritrix. (2014). Obtenido de <https://webarchive.jira.com/wiki/display/Heritrix/Heritrix>
- Heritrix. (s.f.). *New Features in Heritrix 3.0 and 3.1*. Obtenido de Web Archive Jira: <https://webarchive.jira.com/wiki/spaces/Heritrix/pages/5735559/New+Features+in+Heritrix+3.0+and+3.1>
- IIPC. (2012). Obtenido de <http://netpreserve.org/working-groups>
- IIPC. (2014). Obtenido de <http://netpreserve.org/about-us/members>
- Iniciativas de Archivo Web. (Junio de 2014). Obtenido de http://en.wikipedia.org/wiki/List_of_Web_archiving_initiatives#mediaviewer/File:Map_of_Web_archiving_initiatives_worldwide.png
- Internet Archive. (2001). *Storage and Preservation*. Obtenido de <https://archive.org/about/>
- Internet Archive. (s.f.). *Heritrix*. Recuperado el 2013, de Heritrix: <http://crawler.archive.org/>
- Internet Archive. (2013). *Internet Archive*. Recuperado el 2014, de <http://crawler.archive.org/>
- Internet Memory Foundation. (2010). *Web Archiving in Europe*. Obtenido de http://internetmemory.org/images/uploads/Web_Archiving_Survey.pdf, 2010.
- Internet World Stats. (30 de June de 2017). *Internet World Stats*. Obtenido de Internet World Stats: <http://www.internetworldstats.com/stats.htm>
- ISO. (2009). *ISO. 28500 Information and documentation-WARC file format*. Nueva Zelanda.
- Jack, P., & Binns, A. (2012). *Web Archive - Heritrix*. Obtenido de <https://webarchive.jira.com/wiki/display/Heritrix/Heritrix>
- Jacobson, I. (1998). *Applying UML in the Unified Process*. Obtenido de http://www.powershow.com/view/f2189-ZTk1N/Applying_UML_in_The_Unified_Process_Ivar_Jacobson_Rational_Software_email_ivar_rationalcom_powerpoint_ppt_presentation
- Kreuzer, R., Hage, J. H., & Feelders, A. (16 de Junio de 2014). *A Quantitative Comparison of Semantic Web Page Segmentation Approaches. A Quantitative Comparison of Semantic Web Page Segmentation Approaches*. Utrecht, Holanda: Departamento de Información y Ciencias de la Computación, Universidad de Utrecht.

- Kroll, P., & Royce, W. (15 de October de 2005). *IBM Developer Works*. Obtenido de IBM Web Site: <https://www.ibm.com/developerworks/rational/library/oct05/kroll/index.html>
- Lambert, K. (2012). *Fundamentals of Python: First Programs*. Canada: Course Technology.
- Library of Congress Web Archives. (2000). *Home: Library of Congress Web Archives*. Recuperado el 2012, de Library of Congress Web Archives Web site: <http://lcweb2.loc.gov/diglib/lcwa/html/lcwa-home.html>
- Mantura, K., & Martinez, M. (2014). *Definición de las estrategias para el desarrollo del Módulo de Acceso de los contenidos Preservados en Formato WARC para el Prototipo de Archivo Web en Venezuela. (Tesis de Pregrado)*. Caracas. Universidad Central de Venezuela, Caracas - Venezuela.
- Masanès, J. (2006). *Web Archive*. New York: ISBN-10 3-540-23338-5.
- Matplotlib. (08 de Agosto de 2019). *matplotlib version 3.1.1*. Obtenido de Matplotlib: Python Plotting -- Matplotlib 3.1.1 Documentation: <https://matplotlib.org/>
- Meyer, P. (1992). *Probabilidad y Aplicaciones Estadísticas*. Addison-Wesley .
- Microsoft Corporation. (01 de Septiembre de 2006). *Descripción general de Microsoft Solutions Framework (MSF)*. Obtenido de Microsoft: [https://msdn.microsoft.com/es-es/library/jj161047\(v=vs.110\).aspx](https://msdn.microsoft.com/es-es/library/jj161047(v=vs.110).aspx)
- Mohr, G. (09 de Ago de 2010). *WARC (Web ARChive)*. Obtenido de Web Archive Jira: <https://webarchive.jira.com/wiki/spaces/Heritrix/pages/4817/WARC+Web+ARChive>
- National Library of New Zealand; British Library; IIPC. (2012). *Web Curator*. Obtenido de <http://webcurator.sf.net/>
- Navarro, R. (2007). *Rest vs Web Services*. Obtenido de Universidad Politécnica de Valencia: Departamento de Sistemas Informáticos y Computación: <http://users.dsic.upv.es/~rnavarro/NewWeb/docs/RestVsWebServices.pdf>
- NumPy. (2019). *NumPy*. Obtenido de NumPy -- NumPy: <https://www.numpy.org/>
- Open Group. (2016). *Service-Oriented Architecture – What Is SOA?* Obtenido de Open Group: http://www.opengroup.org/soa/source-book/soa/p1.htm#soa_definition
- Organización de las Naciones Unidas para la Educación, I. C. (2003). *DIRECTRICES PARA LA PRESERVACIÓN DEL PATRIMONIO DIGITAL*. Australia : Biblioteca Nacional de Australia.
- Ospina, M. (2014). *Un marco de referencia para la implementación de Archivos Web. Trabajo de Grado de Maestría*. Universidad Central de Venezuela. Caracas - Venezuela.
- Ospina, Martinez, Leon, & Kabchi. (2014). *Desarrollo de una Aplicación para Acceder a Contenidos de un Archivo Web en Formato*. Universidad Central de Venezuela, Facultad de Ciencias, Escuela de Computación.

- Oxford. (8 de Octubre de 2019). *Detect | Definition of Detect*. Obtenido de Detect | Definition of Detect by Lexico: <https://www.lexico.com/en/definition/detect>
- Oxford Dictionary. (8 de Octubre de 2019). *Change | Definition of Change*. Obtenido de Change | Definition of Change by Lexico: <https://www.lexico.com/en/definition/change>
- PANDORA. (1996). Obtenido de <http://pandora.nla.gov.au/>
- PANDORA Australia's Web Archive. (s.f.). *About PANDORA DIGITAL ARCHIVING SYSTEM (PANDAS): Overview*. Recuperado el 2012, de PANDORA Australia's Web Archive Web site: <http://pandora.nla.gov.au/pandas.html>
- Patrón Modelo Vista Controlador*. (s.f.). Obtenido de Junta de Andalucía: <http://www.juntadeandalucia.es/servicios/madeja/contenido/recurso/122>
- Patrón Modelo Vista Controlador*. (s.f.). Obtenido de Junta de Andalucía: <http://www.juntadeandalucia.es/servicios/madeja/contenido/recurso/122>
- Percy, S. (2014). *Phyton*. Obtenido de *Web Frameworks*. Obtenido de <https://wiki.python.org/moin/WebFrameworks>
- Peters, T. (2004). *The Zen of Python*. Obtenido de www.python.org: <https://www.python.org/dev/peps/pep-0020/#id3>
- Portuguese Web Archive. (s.f.). *Home: Portuguese Web Archive*. Recuperado el 2012, de Portuguese Web Archive Web site: <http://sobre.arquivo.pt/>
- Python Software Foundation. (20 de Nov de 2017). *What is Python*. Obtenido de Python Org: <https://docs.python.org/3/faq/general.html#what-is-python>
- Python Software Foundation. (20 de Julio de 2019). *beautifulsoup4*. Obtenido de beautifulsoup4 - PyPi: <https://pypi.org/project/beautifulsoup4/>
- Python Software Foundation. (14 de Agosto de 2019). *difflib — Helpers for computing deltas*. Obtenido de *difflib — Helpers for computing deltas — Python 3.7.4 documentation*: <https://docs.python.org/3/library/difflib.html>
- Python Software Foundation. (14 de Agosto de 2019). *math — Mathematical functions*. Obtenido de *math — Mathematical functions — Python 3.7.4 Documentation*: <https://docs.python.org/3/library/math.html>
- Python Software Foundation. (14 de Agosto de 2019). *Modules*. Obtenido de *Modules -- Python 3.7.4 Documentation*: <https://docs.python.org/3/tutorial/modules.html>
- Python Software Foundation. (14 de Agosto de 2019). *statistics — Mathematical statistics functions*. Obtenido de *statistics — Mathematical statistics functions — Python 3.7.4 Documentation*: <https://docs.python.org/3/library/statistics.html>
- Rational Software . (November de 2001). *Rational Unified Process Best Practices for Software Development Teams*. Obtenido de https://www.ibm.com/developerworks/rational/library/content/03July/1000/1251/1251_bestpractices_TP026B.pdf
- REBUIN. (2009). *Preservación digital: Guía de recursos*. España.

- Sánchez, L., & Milano, G. (2014). *Definición de las estrategias para el desarrollo del Módulo de Administración a los Contenidos Preservados en formato WARC para el Prototipo de Archivo Web de Venezuela*. Caracas: UCV.
- Sanoja, A. (19 de Marzo de 2015). Web page segmentation, evaluation and applications. París, Francia.
- Schwaber, K., & Sutherland, J. (November de 2017). *The Scrum Guide*. Obtenido de Scrum Alliance: <http://www.scrumguides.org/docs/scrumguide/v2017/2017-Scrum-Guide-US.pdf#zoom=100>
- SeleniumHQ. (14 de Agosto de 2019). *SeleniumHQ Browser Automation*. Obtenido de Selenium - Web Browser Automation: <https://www.seleniumhq.org/>
- Sommerville, I. (2005). *Ingeniería del Software*. Madrid: Pearson Educación S.A.
- Sommerville, I. (2005). *Ingeniería del Software*. Madrid: Pearson Educación S.A.
- Sutherland, J. (22 de October de 2011). *Takeuchi and Nonaka: The Roots of Scrum*. Obtenido de Scrum Inc.: <https://www.scruminc.com/takeuchi-and-nonaka-roots-of-scrum/>
- The Pallets Projects. (2019). *Flask | The Pallets Projects*. Obtenido de The Pallets Projects Web Site: <https://palletsprojects.com/p/flask/>
- Troeger, V. E. (2013). *Warwick Department of Economics*. Obtenido de Warwick Department of Economics Web Site: https://warwick.ac.uk/fac/soc/economics/staff/vetroeger/teaching/po906_week8910.pdf
- UNESCO. (2003). *Charter on the Preservation of the Digital Heritage*. Australia.
- UNESCO. (2003). *Directrices para la preservación del patrimonio digital*. Australia.
- UNESCO. (2003). *Noción de preservación digital*. Obtenido de <http://www.unesco.org/new/es/communication-and-information/access-to-knowledge/preservation-of-documentary-heritage/digital-heritage/concept-of-digital-preservation/>
- UNESCO. (16 de January de 2009). *Charter on the Preservation of the Digital Heritage*. Obtenido de Charter on the Preservation of the Digital Heritage - UNESCO Biblioteca Digital: <https://unesdoc.unesco.org/ark:/48223/pf0000179529.page=2>
- Universidad Simón Bolívar. (julio de 2009). *Universidad Simón Bolívar*. Obtenido de <http://ldc.usb.ve/~abianc/materias/ci4713/metodologiasagiles.pdf>
- Vignaga, A., & Perovich, D. (15 de Mayo de 2014). *Enfoque Metodológico para el Desarrollo Basado en Componentes*. Obtenido de ResearchGate Web Site: https://www.researchgate.net/publication/228766936_Enfoque_Metodologico_para_el_Desarrollo_Basado_en_Componentes
- W3 Consortium. (11 de Feb de 2004). *Web Services Architecture*. Obtenido de W3.org: <https://www.w3.org/TR/ws-arch/>

- W3C. (June de 1999). *Hypertext Transfer Protocol -- HTTP/1.1*. Obtenido de W3C Org: <https://www.w3.org/Protocols/rfc2616/rfc2616-sec9.html>
- W3C. (11 de Febrero de 2004). *Web Services Architecture*. Obtenido de W3C: <https://www.w3.org/TR/2004/NOTE-ws-arch-20040211/#relwwwrest>
- Walpole, R., Myers, R., & Myers, S. (2002). *Probabilidad y estadística para ingeniería y ciencias*. México: Pearson.
- Wikipedia. (s.f.). *Binomial Distribution*. Obtenido de Wikipedia: https://en.wikipedia.org/wiki/Binomial_distribution
- XP *Extreme Programming. What is Agile*. (2006). Obtenido de http://www.objectmentor.com/omSolutions/agile_what.html