UNIVERSIDAD CENTRAL DE VENEZUELA

FACULTAD DE CIENCIAS

ESCUELA DE COMPUTACIÓN

Desarrollo e implementación del módulo de predicción de cambios de sitios web para un prototipo de Archivo Web.

Trabajo especial de grado presentado ante la ilustre

Universidad Central de Venezuela por

Br. Casanova Diaz Marco Antonio.

Br. Carabali Sandoval Willibert.

Para optar al título de Licenciado en Computación

Tutora: Profa. Mercy Ospina

2015

Agradecimientos y dedicatorias

Dedicamos este trabajo principalmente a Dios, por habernos dado la vida y permitirnos el haber llegado hasta este momento tan importante de nuestra formación profesional. A nuestras madres, por ser el pilar más importante y por demostrarnos siempre su cariño y apoyo incondicional sin importar nuestras diferencias de opiniones. A nuestros padres, en especial a Marco Casanova por apoyar una iniciativa que en los comienzos fue cuesta arriba y que a pesar de la distancia, sentimos que está con nosotros siempre. Agradecimientos a todas las personas que nos dieron apoyo, en especial al Prof. Adrián Bottini que se le aprecia con mucho cariño al igual que la profesora Mercy Ospina nuestra guía en todo momento.

Desarrollo e implementación del módulo de predicción de cambios de sitios web para un prototipo de Archivo Web.

Autores: Casanova Marco
Carabali Wilibert
Tutora: Profa. Mercy Ospina Torres.
Fecha: 14/06/2015.

RESUMEN

El patrimonio cultural es la herencia cultural propia del pasado y presente de los pueblos, el cual ha sido definido como tangible (objetos arqueológicos, artísticos, entre otros) e intangible (costumbres, conocimiento científico, entre otros). Cada vez más, parte de este patrimonio es publicado en Internet, el cual ha sido denominado patrimonio web siendo parte del patrimonio intangible. Esto hace necesario establecer formas de preservar este contenido, lo cual ha sido implementado con el desarrollo de sistemas que copien el contenido a preservar en ambientes seguros e independientes a los servidores originales, llamados Archivos Web. Una característica importante del patrimonio web es su naturaleza cambiante y dinámica, lo que presenta un reto a estos sistemas, el cual es disponer de un mecanismo que permita en cierto grado predecir los cambios en los contenidos que son preservados, de esta manera se estarán almacenando los cambios presentados en un momento dado, con lo que el Archivo sería de mayor utilidad a aquellos usuarios que necesiten ver estos cambios, evitando por otro lado almacenar una imagen repetida del mismo sitio, por lo que también se ahorra en recursos de almacenamiento. En este Trabajo Especial de Grado se desarrolló e implemento un módulo de predicción de cambios de sitios web, usando variables aleatorias, en base a las cuales se elaboró una serie de algoritmos de predicción. Cabe destacar, que los algoritmos se implementaron utilizando el método heurístico, donde se reconocen nuevos patrones de comportamiento en los sitios web, y se aplica una solución particular con un algoritmo que resuelve dicha problemática. Se probaron 50 páginas, obteniendo como resultado una mejora sustancial en el proceso de rastreo, almacenamiento de versiones y reducción en las versiones duplicadas.

Palabras Claves: Archivo Web, preservación, predicción, variables aleatorias

Indice

Índice de Figuras

Introducción

El patrimonio cultural define a los pueblos y representa su herencia histórica, por lo cual, es importante su preservación, El patrimonio cultural ha sido clasificado como tangible (obras materiales) o intangible (generación y transmisión de conocimiento, costumbres, etc.), dentro de este último se define el patrimonio digital como aquel que ha sido originado de manera digital y abarca recursos como páginas o recursos web, bases de datos, libros digitales, material multimedia, grabaciones, programas informáticos, entre otros. Entre estos, los recursos web que se diferencian de cualquier otro tipo de recurso digital por su naturaleza cambiante y su estructura de hiperenlaces, por lo que su preservación tiene características propias y retos que deben ser abarcados por los sistemas que buscan su preservación histórica.

En la UCV se desarrolló un prototipo de Archivo Web cuyo objetivo es preservar sitios web de interés en Venezuela, en este desarrollo se ha observado que se hace necesario verificar cambios en los sitios web que se preservan para determinar si se almacena una nueva versión del mismo ya que de no hacerlo se pueden perder cambios importantes en dichos sitios.

Antes de la implementación del componente de predicción, la cosecha de nuevas versiones tenía una frecuencia fija, esta era establecida por el usuario que solicita la preservación de este sitio, pero esto conllevaba a que se almacenara información redundante en frecuencias más rápidas que la tasa o relación de cambio del sitio, o que se perdieran cambios importantes si ocurría el caso contrario.

El presente trabajo especial de grado propone el desarrollo del componente de predicción que se integra a los módulos de adquisición y gestión de almacenamiento del prototipo, para modificar la frecuencia de cambio de cada sitio, y evitar tener versiones duplicadas, basados en estándares y usando tecnología de software libre.

En el Capítulo 1 se plantea el problema, se describen los módulos que integran el archivo web y se procede a explicar la situación del módulo de adquisición el cual es de nuestra competencia en el Capítulo 2 se define que es un archivo web, y se exponen la técnica predictiva que funciona como base a los modelos de predicción de cambios de las páginas web, en el Capítulo 3 se trata todo lo referente a la metodología Ágil Unificada de Procesos (AUP) adaptada a una arquitectura basada en componentes. En el Capítulo 4, se define formalmente el desarrollo de la aplicación y los resultados obtenidos.

Capítulo 1. La Investigación

En este capítulo se describe el problema de estudio y sus antecedentes; así como los objetivos planteados en esta investigación y la metodología para llevarlos a cabo.

1.1. 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 lograr esto, se deben realizar un conjunto de tareas [8] que hagan factible la preservación de la web. Estas se presentan en la Figura 1-1.

Figura 1-1 Tareas que lleva acabo un Archivo Web para el resguardo de un sitio web.

La selección de páginas o sitios web a resguardar: permite limitar el ámbito del archivo; pudiendo preservar contenidos locales o de un tipo en particular. Por ejemplo, contenidos de un país o estrictamente educativos.

La adquisición del contenido regular de las páginas: logra que se puedan almacenar los cambios que se generan en los contenidos que se preservan a través del tiempo.

El almacenamiento e indexación de páginas resguardadas: permite preservar grandes volúmenes de información (del orden de los Terabytes), archivos (millones de archivos) y formatos.

El acceso o recuperación de los contenidos resguardados: 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 como lo hace en los servidores originales.

Actualmente, en la Universidad Central de Venezuela, por medio de un equipo liderado por la Profa. Mercy Ospina, se desarrolló un prototipo de Archivo Web en Venezuela. Este desarrollo cuenta con los siguientes módulos y funcionalidades:

Módulo de adquisición:

- Suscripción a sitios web
- Establecer una frecuencia de adquisición
- Programación periódica de rastreo

Módulo de Almacenamiento:

- Almacenar los contenidos adquiridos de los sitios suscritos en formato WARC, el cual especifica un método para múltiples recursos digitales combinados en un único archivo. El formato WARC es una revisión del Internet Archive's ARC File Format bajo la norma ISO 28500:2009.
- Indexar las versiones de dichos sitios de acuerdo con su fecha de adquisición.

Módulo de Acceso:

- Búsquedas de contenido por URL.
- Búsquedas por colección.
- Seleccionar versión de un sitio web.
- Desplegar versión para su navegación.
- Registro de usuarios, manejo de historias de búsqueda y favoritos.

En este prototipo se determinó que el uso de una frecuencia fija para el rastreo de las páginas web no es una solución apropiada, pues se presentan las siguientes características:

1. El sitio web puede tener una tasa de cambio diferente a esta frecuencia.
2. Los sitios web no tienen la misma tasa de cambio a través del tiempo.

Esto trae como consecuencia que el sitio cambie más lentamente que la frecuencia establecida, produciéndose duplicidad y; por ende, no aprovecha eficientemente el espacio de

almacenamiento; y, en consecuencia, se encarecen los costos. En el caso contrario, si el sitio cambia con mucha frecuencia, pueden perderse muchos cambios importantes que no serán almacenados.

1.2. Antecedentes

Los Archivos Web son sistemas de información que se han venido desarrollando desde finales de los años 90 para llevar a cabo la preservación de la Web. El primer Archivo Web conocido comenzó a desarrollarse en el año 1996 cuando Brewster Kahle creó "Internet Archive" con la finalidad de archivar de manera histórica sitios web de cualquier parte del mundo. Este Archivo opera en los Estados Unidos con un presupuesto anual de $10 millones, provenientes entre otros, del servicio de rastreo y almacenamiento de la Web [4]. A través de los años, se han ampliado sus colecciones, pues además de páginas web de texto, ahora incluye libros, audio, imágenes y software.

Sumado a esto, surgieron un gran número de iniciativas que incluyen mejoras significativas en cuanto al rendimiento del Archivo Web, específicamente, el poder predecir el comportamiento asociado al cambio de las páginas web. Los modelos de predicción son abstracciones de representaciones del mundo real, que buscan aproximaciones sobre eventos y comportamientos futuros para la toma de decisiones, jugando un papel clave en el proceso de la determinación de la frecuencia de cambio de un sitio web. Actualmente, no se concibe que se realicen predicciones sin la ayuda de alguno de los múltiples modelos numéricos que existen, probabilísticos o basados en conocimiento.

En Agosto de 2003, Junghoo Cho y Hector García publican en una revista el artículo *Estimación de una frecuencia de cambio*, donde plantean que muchas fuentes de datos en línea se actualizan de forma autónoma e independiente. Por lo tanto, es de suma importancia estimar la frecuencia de cambio de datos para mejorar los rastreadores web, cachés web y ayudar en el proceso de minería de datos. Se deben identificar varios escenarios en los que diferentes aplicaciones tienen diferentes requisitos sobre la precisión de la frecuencia estimada. Es entonces cuando se procede a ajustar varias frecuencias (estimadores) en los escenarios identificados. En muchos casos, los estimadores propuestos predicen frecuencias de cambio con mucha más precisión mejorando la eficacia de las aplicaciones. [5]

En el 2010, Qingzhao Tan y Prasenjit publican el artículo *Clustering basado en los rastreadores incrementales web*, donde proponen que al rastrear los recursos, el número de máquinas y materiales de los que se dispone, puede producirse un problema que limita la eficacia del rastreo, puesto que tiene que decidirse un orden óptimo sobre las páginas que se volverán a rastrear nuevamente. Es ideal que los rastreadores soliciten solo aquellas páginas que han cambiado desde el último rastreo. En la práctica, un rastreador no puede saber si una página web ha cambiado antes de descargarla. Identificar las características de las páginas web que estén en correspondencia con su frecuencia de cambio es fundamental, además de la agrupación de las páginas web sobre la base de características que se correlacionan con sus frecuencias de cambio; siendo este determinado por el histórico. El rastreador descarga una muestra de las páginas web de cada grupo y en función de que un número significativo de estas páginas han cambiado en el último ciclo de rastreo, decide volver a rastrear todo el clúster. El algoritmo de muestreo rastrea eficazmente las páginas con los patrones de cambio similares, tomando como ventaja la previa agrupación de las páginas web por tener una frecuencia de cambio en común. [6]

En el 2013, Kira Radinsky y Paul N. Bennett publicaron un documento bajo el título *Prediciendo el cambio del contenido en la web*, donde explican que un algoritmo de predicción se puede diseñar de tal manera que vuelva a rastrear páginas cuando sea necesario, convirtiéndose así en un mecanismo proactivo que permite la interacción de un usuario que puede personalizar la configuración directamente. Aunque actualmente muchas técnicas y modelos de predicción de cambio se centran simplemente en el histórico, resulta ser de interés el grado y la relación entre los cambios observados de la página, la relación con otras páginas y la similitud en los tipos de cambios que experimentan. Se considera además, que un marco de predicción de sistema experto incorpora la información recolectada o técnicas básicas de aprendizaje relacional. Es importante tomar en cuenta numerosas medidas de similitud para identificar páginas relacionadas y enfocar específicamente las medidas de similitud temporal del contenido. [7]

En el caso de Venezuela, no se encontraron referencias de iniciativas gubernamentales o privadas en el desarrollo de componentes que permitan la predicción de cambios en páginas web.

1.3. Preguntas de Investigación

¿Cuáles son las herramientas, tecnologías y estándares actuales para el desarrollo de modelos de predicción de cambios de sitios web?

¿Qué elementos principales debe tener un modelo probabilístico para la predicción de cambios de sitios web?

¿Cuáles beneficios ofrece el desarrollo de un modelo probabilístico para predicción de cambios de sitios web y su implementación en un Archivo Web?

1.4. Objetivos

Objetivo general del TEG

Desarrollar e implementar el modulo de predicción de cambios de sitios web para un prototipo de Archivo Web.

Objetivos específicos del TEG

- Adaptar el método de Ágil UP en conjunto con una arquitectura basada en componentes.
- Desarrollar los componentes que conformarían el módulo de predicción de cambios, usando como base métodos de estadística y probabilidades descriptivas.
- Elaborar un módulo administrativo que permita al usuario acceder y monitorear las frecuencias de cambio en los sitios web preservados.
- Calibrar el módulo de predicción utilizando el histórico.
- Establecer una clasificación temporal de los sitios web de acuerdo con su comportamiento
- Integrar el módulo de predicción como un componente del módulo de adquisición y realizar pruebas de integración.
- Realizar pruebas de conformidad.

1.5. Justificación

El principal objetivo de la preservación web es garantizar la disponibilidad del contenido para

que pueda ser utilizado en investigaciones presentes y futuras, incluso si deja de estar en línea. Así, la eliminación de los cambios o contenidos en las páginas, no significará el que se pierdan para siempre. Sin embargo, este proceso ocupa altos volúmenes de almacenamiento, por lo que se deben tomar en cuenta los costos del proceso, que pueden ser muy altos. En el caso del prototipo en desarrollo, el almacenamiento de versiones se hace por sitio web. En una versión, se pueden almacenar cientos o miles de documentos. En la tabla Tabla 1-1 se observa que, en un periodo de 6 meses, se almacenaron 446 versiones de 20 sitios web de muestra. Estos abarcan alrededor de cuatro millones y medio de documentos entre páginas web, imágenes, videos y archivos de texto, entre otros; ocupando un total de 811 Gigabytes de almacenamiento. También puede observarse que se repite el almacenamiento de la misma versión del sitio web (pues no se registraron cambios). Tal es el caso del sitio www.rena.edu.ve .

Como la finalidad de un Archivo Web es preservar la mayor información útil posible en el tiempo; es importante considerar factores que influyen en el tamaño del espacio de almacenamiento, como el evitar guardar dos versiones de un sitio web en el que no se hayan producido cambios (Ver Tabla 1-2).

Por otra parte, no deben perderse cambios importantes en una página o sitio suscrito en el Archivo Web. Es necesario contar con técnicas que permitan predecir el comportamiento de una página o sitio web en el transcurso del tiempo, debido a que los mismos pueden:

- Presentar una tasa de cambio diario constante. Por ejemplo, páginas de noticias.
- Presentar una tasa de cambio muy baja. Por ejemplo, páginas de investigación científica.
- Presentar una tasa de cambio constante en los mismos periodos de tiempo. Por ejemplo, sitios de universidades o institutos educativos, donde hay más actividad en periodos de inscripción y menos durante el desarrollo de los cursos y las vacaciones.
- Puede no presentar un patrón o patrones definidos.

El poder almacenar el historial de cambio en un sitio web, permite poder predecir patrones de cambio y realizar análisis sobre el comportamiento de los sitios web suscritos. Estos datos permitirán también realizar proyecciones de crecimiento del Archivo Web a futuro.

Tabla 1-2 Almacenamiento de sitios web

Colección / URL sitio web	Versiones	Total documentos	G bytes
Científico	98	862.077	348,84
http://www.ivic.gob.ve	71	650.247	347,50
http://www.mcti.gob.ve	27	211.830	1,34
Cultural	50	349.655	44,43
http://www.fmn.gob.ve	14	9.140	0,83
http://www.pdvsalaestancia.com	36	340.515	43,59
Educativo	276	3.340.411	410,00
http://profnanotic.blogspot.com	12	67.078	11,58
http://www.ciens.ucv.ve	4	618.450	44,40
http://www.faces.ucv.ve	17	10.742	0,30
http://www.ing.ucv.ve	35	10.390	0,29
http://www.luz.edu.ve	33	191.959	2,18
http://www.mppeu.gob.ve	42	909.617	161,10
http://www.rena.edu.ve	3	67.371	0,81
http://www.uc.edu.ve	5	114.287	5,15
http://www.ucv.ve	28	748.067	130,90
http://www.udo.edu.ve	21	266.406	4,30
http://www.ula.ve	16	33.816	0,96
http://www.unefa.edu.ve	13	164.583	25,80
http://www.unet.edu.ve	4	43.805	12,20
http://www.usb.ve	43	93.840	10,05
Tecnológico	22	14.900	0,43
http://smartappsla.com	21	3.768	0,13
http://www.venezuela.net.ve	1	11.132	0,30
Total general	446	4.567.043	811,41

1.5. Alcance

Esta investigación desarrolla un prototipo que permite la predicción de cambios en sitios web en Venezuela. Al momento de hacer la adquisición del contenido, se determina si fue adquirido anteriormente. Es decir, se hace la detección de duplicados.

También se ubican las métricas relevantes recolectadas durante la adquisición y el almacenamiento del contenido, para mostrarlas dentro de la aplicación de consulta. Cabe destacar que se analizaron algunas de estas métricas; como es el caso de la identificación de sitios que presenten cambios importantes. Un cambio es considerado importante si cumple con la siguiente condición:

Sea F(μ) una función que retorna un valor donde $0 < F(\mu) < 1$, (este valor está determinado por el componente de comparación integrado al módulo de adquisición), sea αi un vector de cambios que mide los cambios asociados con la vista(esto es párrafos, colores, disposición de elementos, entre otros), βi el vector de cambios asociados con la forma y sea γ el vector de cambios asociados con la adición de contenido. En la Figura 1-2 podemos observar la fórmula para determinar si un cambio es considerado importante.

$$\text{Si } F(\mu) < \sum_{i=0}^{n} \alpha i + \sum_{i=0}^{n} \beta i + \sum_{i=0}^{n} \gamma i, \text{ existe un cambio importante.}$$

Figura 1-2 Cambio importante.

La investigación se limitó al desarrollo de las siguientes actividades:

- Cálculo de la frecuencia de cambio asociado a cada sitio web, calibración del componente y desarrollo de una interfaz para configurar y acceder a las características asociadas a cada sitio web.
- Identificación de servicios fundamentales para el módulo de adquisición, así como los recursos tecnológicos necesarios para su funcionamiento. De igual forma, se identificaron los modelos existentes de predicción
- Desarrollo de un prototipo de terminal de comandos que permite realizar seguimiento de la frecuencia de cambio asociada a cada página web, además de permitir realizar ajustes manuales en las frecuencias de cambio o visualización de métricas.

En esta investigación se desarrolló un modelo metodológico que da soporte a la administración, desarrollo y gerencia del proyecto; adaptado en un contexto que utiliza una arquitectura orientada a los componentes.

Capítulo 2. Marco Teórico

En este capítulo se describen las teorías, conceptos procesos relacionados con la preservación web y la predicción de cambios

2.1.Preservación de Archivos Digitales

La preservación de archivos digitales o preservación Web se refiere al proceso de recolectar información disponible en la *World Wide Web,* almacenarla en un formato de archivo digital, y para así garantizar que el contenido pueda ser consultado posteriormente. [8]

2.1.1. Patrimonio Web

El patrimonio cultural puede ser considerado como la herencia cultural propia del pasado de una comunidad, con la que ésta vive en la actualidad y que transmite a las generaciones presentes y futuras. La UNESCO (Organización de las Naciones Unidas para la Educación, la Ciencia y la Cultura) ha clasificado el patrimonio en tangible e intangible, a continuación se detallan:

Patrimonio Tangible: Comprende los objetos arqueológicos, históricos, artísticos, etnográficos, tecnológicos, religiosos y aquellos de origen artesanal o folklórico que constituyen colecciones importantes para las ciencias, la historia del arte y la conservación de la diversidad cultural del país, como los monumentos, conjuntos de construcciones y sitios con valor histórico, estético, arqueológico, científico, etnológico y antropológico.

Patrimonio Intangible: Expresiones y prácticas culturales, constituido, entre otros elementos, por la poesía, los ritos, los modos de vida, investigación científica, la medicina tradicional, la religiosidad popular y las tecnologías tradicionales de cada país.

La idea de patrimonio se ha extendido a categorías que van más allá de los sectores artísticos, las cuales también tienen un gran valor para la humanidad, una vertiente moderna del patrimonio es aquella que no sólo toma en cuenta las memorias pasada sino también los testimonios presentes, lo cuales se almacenan cada vez más en formato digital. Estos objetos digitales pueden ser textos, bases de datos, imágenes fijas o en movimiento, grabaciones sonoras, material gráfico, programas informáticos o páginas Web.

Preservación web

La preservación web debe garantizar que el contenido salvaguardado pueda ser reproducido o accedido de manera similar a como funcionaba en su entorno operativo [9], para lograr esto debe mantenerse la integridad el grupo de páginas seleccionadas para ser conservadas, es decir, acompañadas de imágenes, formatos de archivos, gráficas, entre otros.

2.2.Archivo Web

Un archivo web es un sistema de información cuyo objetivo es la preservación web y su definición está estrechamente ligada con el concepto de Archivo histórico documental. Este debe cumplir las exigencias propias de la archivista, entre estas exigencias se encuentran las siguientes:

- Debe ser un repositorio histórico de los documentos a preservar.
- El material almacenado debe ser fiel al original, con la menor cantidad de cambios posibles
- El sistema debe ser seguro, de manera que garantice la preservación de los documentos almacenados
- Debe proveer una forma de acceder a los contenidos de los documentos preservados.

Para realizar la preservación web es necesario que el Archivo Web lleve a cabo cada una de las siguientes tareas [8]:

- **Selección:** es necesario tener bien definido un sistema o criterio de selección, los criterios de selección se adaptan a las necesidades de la organización encargada de llevar a cabo la preservación.

- **Adquisición:** Los sitios web son recuperados utilizando un software para descargar imágenes, documentos, archivos HTML y otros archivos necesarios para reproducir fielmente el sitios web tal cual se veía en el momento en que fue capturado, este software también se encarga de coleccionar metadata asociada al proceso de recolección.

- **Almacenamiento e indexación:** La preservación web tiene entre sus objetivos preservar el contenido adquirido sin que sufra modificaciones. Para lograr esta meta las herramientas, estándares, políticas y mejores prácticas deben estar presentes en el lugar que se encargará de

la gestión de los archivos web a través del tiempo. En este aspecto, también se contemplan las medidas a utilizar para facilitar la posterior búsqueda del contenido preservado.

- **Acceso:** El Archivo Web debe permitir al usuario visualizar cada versión histórica de los sitios web archivados página por página o procesarlos por colecciones, asimismo se pueden consultar características generales de la versión, los sitios, o las colecciones.

2.2.1 Versiones

Una versión es el estado en que se encontraba un sitio web en un momento dado (se puede decir que es una foto del sitio), las versiones nacen de la necesidad de reflejar y almacenar los cambios que se producen en el contenido y la estructura del sitio web, en la figura 2-1 puede verse la relación.

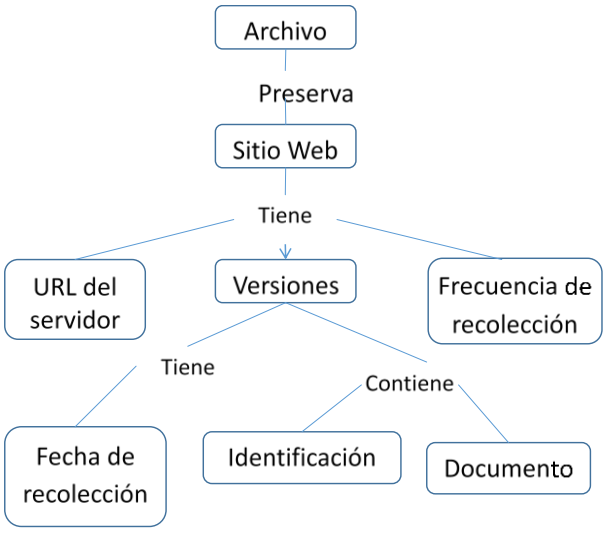


Figura 2-1 Diagrama de la Versión de un sitio WEB

2.3.Problemáticas asociadas a la adquisición y almacenamiento en un Archivo Web

Existen diversos Archivos Web que han sido desarrollados alrededor del mundo, donde se han planteado problemas similares con respecto al manejo del almacenamiento. Las soluciones que se den a los problemas presentados en nuestro prototipo pueden ser referenciadas para solventar dificultades que presenten similitudes con otros Archivos Web. Las problemáticas se exponen a continuación:

- **Visualización de métricas:** para que el usuario pueda visualizar las métricas relevantes, se debe hacer uso de herramientas no nativas al sistema como tal, lo que genera un procesamiento y carga adicional sobre el sistema.
- **Ausencia de elementos para determinar comportamiento de un sitio web:** Los sitios web no están clasificados según la manera en que estos cambian, como consecuencia son tratadas con una frecuencia de rastreo fija que no refleja los cambios que puedan presentarse en dichos sitios.
- **Redundancia de versiones o pérdida de cambios importantes:** la ausencia de la asignación de una frecuencia de cambio dinámica conlleva a utilizar una frecuencia fija, si esta es más rápida que la tasa de cambio del sitio se presenta duplicidad de datos, pero si es más lenta se pueden perder cambios importantes que no serán almacenados.

2.3.1. Formato WARC

El formato WARC (Web Archive) es un formato contenedor de archivos para el almacenamiento en Archivos Web, propuesto por el IIP(Programa Internacional de Información) en el 2004, fue aprobado como estándar ISO en el 2009 , este estándar, especifica un método para combinar múltiples recursos digitales en un archivo de archivos agregados junto con la información relacionada. Un archivo WARC registra una secuencia de registros en donde se almacenan las páginas y demás documentos web adquiridos usando un rastreador, cada registro está precedido por un encabezado que describe brevemente su contenido.

Como podemos ver en la figura 2-3, con el Archivado con Servidor Web el sitio original se rastrea y las respuestas se almacenan sin cambios en el contenedor (archivos WARC). Lo que permite evadir el mapeo del archivo del sistema para la asignación de nombres por convenio y el cambio de la estructura del enlace. El acceso requiere que un Servidor Web que obtenga lo que este almacenado en los contenedores y los envía como una respuesta al usuario final.

La conservación del esquema de nombre original (incluyendo los parámetros de las páginas dinámicas) permite la navegación en el sitio tal cual como ha sido rastreado. El usuario del archivo puede recorrer de nuevo todos los caminos seguidos por el rastreador

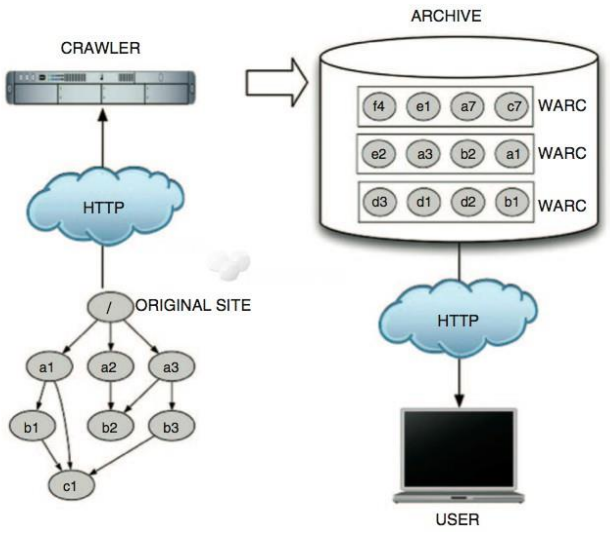


Figura 2-2 Archivado con Servidor Web [8]

.

La principal ventaja de la utilización de contenedores WARC es la posibilidad de superar la limitación del sistema de archivos de almacenamiento en términos de tamaño (al final se almacenan menos archivos individuales en el sistema de archivo del archivo Web) y el espacio de los nombres de cada uno de los archivos se puede conservar.

Descripción del formato WARC: Cuando se concatenan uno o más registros WARC, esta concatenación es considerada un archivo de formato WARC. Es el primer registro quien suministra la información de los registros WARC que siguen. El contenido de un registro es usualmente el resultado de la adquisición de páginas WEB, imágenes e información de re-direccionamiento URL como consecuencia de búsqueda de nombres a través de protocolos DNS, archivos únicos o material sintetizado(metadatos, contenido transformado). Un registro WARC consiste de una cabecera seguida de un bloque de contenido. [10]

De manera de analizar un conjunto de datos y predecir futuras tendencias, comportamientos o eventos ya sea basado en datos históricos o no, son utilizadas tecnicas de prediccion. En el siguiente apartado se explicara los conceptos claves que forman parte del Algoritmo de predicción basado en probabilidades y estadística.

2.4. Técnicas de predicción

Las tecnicas de predicción de cambios en sitios web son un conjunto de procedimientos y recursos [26] que se abstraen de la representación del mundo real, generando aproximaciones del comportamiento y propiedades que se quieren estudiar de un sitio en específico. Una buena técnica de predicción debe proveer las siguientes características:

Adaptación al cambio: El cambio de los sitios es una realidad, y más aún si se habla de cambios de gran magnitud. Las técnicas predictivas deben hacer frente a los cambios actuales y futuros por medio de la adaptación, acorde al comportamiento que esté generando el sitio. **Información oportuna y precisa:** La adopción de decisiones basadas en información precisa y oportuna permite sustentar el crecimiento del componente de predicción. Un margen de error elevado en la recolección de información y procesamiento conllevara a establecer conclusiones erradas.

Planificación de la capacidad de disco: El espacio en disco ganado por sustituir la frecuencia de cambio fija y utilizar la generada por el componente de predicción, permite incorporar nuevos sitios para ser rastreados y extender el tiempo de observación sobre estos.

En los siguientes puntos, se definirán una serie de teorías y distribuciones que se utilizaron en todo momento para el desarrollo del Algoritmo de predicción de cambios.

2.4.1. Modelo de Laplace

Un modelo probabilístico finito es un par (Ω, p) donde Ω es un conjunto no vacío finito (conocido como espacio muestral) y p es una funcion p: $\Omega \rightarrow R^+$, conocida como función de densidad de probabilidad (ver Figura 2-3), tal que [27]:

$$i) p(\omega) \geq 0 \quad \forall \omega \in \Omega$$

$$ii) \sum_{\omega \in \Omega} p(\omega) = 1$$

$$\Omega = \{\omega_1, \omega_2, \omega_3, \ldots, \omega_k\}$$

Figura 2-3 Función de densidad de probabilidad.

Un modelo probabilístico (Ω, p) se llama Modelo de Laplace (es decir: todos los resultados son equiprobables) sí [27]:

$$\forall \omega \in \Omega$$
$$p(\omega) = \frac{1}{|\Omega|}$$

Figura 2-4 Modelo de Laplace.

Se conoce como función probabilística P: partes (Ω) → R^+

$$P(E) = \sum_{\omega \in E} p(\omega) = \frac{|E|}{|\Omega|}$$

Figura 2-5 Función probabilística partes de omega.

Según el modelo de Laplace, para cualquier subconjunto (conocido como evento) E⊆Ω.

2.4.2. Distribución Bernoulli.

La Bernoulli es una distribución de probabilidad discreta dicotómica, usada para modelar experimentos aleatorios individuales con 2 posibles resultados: 1 (denominado 'éxito') o 0 (denominado 'fracaso'). Una variable aleatoria X con distribución Bernoulli toma valor 1 con probabilidad (p) y valor 0 con probabilidad (1 – p), siendo su función de densidad de probabilidad la siguiente (ver Figura 2-6):

$$P(x) = p^x (1 - p)^{1-x} \qquad x = 0,1$$

Figura 2-6 Función de probabilidad Bernoulli.

Si X es una variable aleatoria que mide el "número de éxitos", y se realiza un único experimento con dos posibles resultados (éxito o fracaso), se dice que la variable aleatoria X se distribuye como una Bernoulli de parámetro *p*.

$$X \sim Be(p)$$

Figura 2-7 X se distribuye como una Bernoulli.

Se conoce como función de distribución de probabilidad acumulada caso discreto a la siguiente expresión:

F(x) = Prob{Variable aleatoria toma valor menor o igual que x} = ∑ fdp hasta x

La Función de distribución de probabilidad acumulada Bernoulli es:

$$F(x) = \begin{cases} 0, \text{ para } x<0 \\ 1-p, \text{ para } 0<=x<1 \\ 1, \text{ para } x>=1 \end{cases}$$

Figura 2-8 Función de distribución Bernoulli.

2.5. Control de versiones

Un sistema de control de versiones (o sistema de control de revisiones) es una combinación de tecnologías y prácticas para seguir y controlar los cambios realizados en los ficheros del proyecto, en particular en el código fuente, en la documentación y en las páginas web.

La razón por la cual el control de versiones es universal es porque ayuda virtualmente en todos los aspectos al dirigir un proyecto: comunicación entre los desarrolladores, manejo de los lanzamientos, administración de fallos, estabilidad entre el código y los esfuerzos de desarrollo experimental y atribución y autorización en los cambios de los desarrolladores. El sistema de control de versiones permite a una fuerza coordinadora central abarcar todas estas áreas. El núcleo del sistema es la gestión de cambios: identificar cada cambio a los ficheros del proyecto, anotar cada cambio con meta-data como la fecha y el autor de la modificación y disponer esta información para quien sea y como sea. Es un mecanismo de comunicación donde el cambio es la unidad básica de información. [12]

Una versión, es el estado en el que se encuentra el mismo en un momento dado de su desarrollo o modificación. Un sistema de control de versiones debe proporcionar:

- Un Sistema para almacenar los formatos de archivos o elementos a ser gestionados, desde archivos txt, php, html, como tipos de imágenes como png o jpg hasta formatos de diseño como psd
- Posibilidad de realizar cambios sobre los elementos almacenados (ej. modificaciones parciales, añadir, borrar, renombrar o mover elementos).
- Registro histórico de las acciones realizadas con cada elemento o conjunto de elementos (normalmente pudiendo volver o extraer un estado anterior del producto. [12]

A continuación se realiza una comparación entre 3 controles de versiones con el objetivo de escoger el control de versiones que se adapta y cumpla los requerimientos que el proyecto requiere [22]:

Apache Subversion (SVN)

SVN fue creado como una alternativa a CVS corrigiendo algunos errores en el sistema CVS buscando al mismo tiempo preservar la compatibilidad. Al igual que CVS, SVN es libre y de código abierto, con la diferencia de ser distribuido bajo la licencia Apache en lugar de GNU.

Ventajas:

- Basado en CVS
- Incluye operaciones atomicas
- Amplia disponibilidad de plug-ins embebidos para IDEs
- No utiliza el modelo peer-to-peer

Desventajas:

- Presenta bugs a la hora de renombrar archivos y directorios del
- Los comandos para la gerencia de los repositorios son insuficientes

Git

Git surge con la motivación de hacer un sistema de control de revisiones más rápido, distribuido y poder afrontar abiertamente las convenciones y prácticas utilizadas en CVS . Está desarrollado principalmente para Linux y cuenta con todo el soporte que el sistema operativo pueda brindarle. También puede funcionar en otros sistemas tipo Unix, y puertos nativos de Git están disponibles para Windows como msysgit. Como no hay un servidor centralizado, Git no se presta a los proyectos para desarrolladores individuales o en pequeños grupos como el código no necesariamente puede estar disponible cuando se utiliza un equipo que no repositorio. Existen soluciones para este problema, y algunos ven mejoras en la velocidad de Git como un compromiso decente para la molestia.

Ventajas:

- Histórico disponible en modo fuera de línea
- Totalmente Distribuido
- Utiliza el modelo peer-to-peer.

Desventajas:

- No esta optimizado para el desarrollo individual
- Soporte limitado para los usuarios de Windows

Mercurial

Ventajas:

- Basado en un Modelo Distribuido

Desventajas:

- No se puede realizar merge de dos nodos padres

2.6.1. Git

Git es un sistema de control de versiones distribuido cuyo objetivo es el de permitir mantener una gran cantidad de código a una gran cantidad de programadores eficientemente.

La primera gran diferencia de Git con respecto a otros sistemas de control de versiones es la forma que tiene de manejar los cambios en los ficheros. Mientras que otros Scv(Sistema de Control de Versiones) almacenan los archivos originales, conservando una lista de los cambios

realizados a dichos archivos en cada versión, Git guarda una "foto" (snapshot) del estado de cada archivo en un momento concreto. Si uno de los archivos no ha cambiado no crea una nueva copia del mismo, simplemente crea una referencia al archivo original. [25]

La segunda es la eficiencia. Git se basa en que cada programador almacena una copia completa del repositorio en su máquina de forma local, incluido el historial de cambios. Esto implica que muchas de las operaciones realizadas sobre el código fuente no tienen lugar en la red, permitiendo que la velocidad de proceso dependa únicamente en los recursos locales. [13]

Fundamentos de Git:

- **Casi cada operación es local:** La mayoría de las operaciones en Git sólo necesitan archivos y recursos locales para operar, generalmente no es necesaria la información de otro equipo de la red.
- **Integridad:** Es imposible cambiar el contenido de cualquier archivo o directorio sin Git lo sepa. Esta funcionalidad está integrada en Git en los niveles más bajos y es parte integral de su filosofía. No se puede perder información en tránsito o conseguir la corrupción de archivos sin Git ser capaz de detectarlo.
- **Git generalmente solo añade datos:** Al realizar acciones en Git, casi todos ellas se limitan a agregar datos a la base de datos de Git.

Git posee tres secciones principales de un proyecto las cuales son el directorio de Git, el directorio de trabajo, y el área de preparación que son descritas a continuación. [13]

- **El directorio Git:** es donde se almacenan los metadatos y la base de datos de objetos para el proyecto.
- **El directorio de trabajo:** es una versión del proyecto. Estos archivos se extraen de la base de datos comprimida en el directorio Git y se colocan en el disco para que puedan ser utilizados o modificados.
- **El área de preparación:** es un archivo, generalmente contenido en el directorio de Git, que almacena información acerca de lo que va a pasar en la próxima confirmación. Se refiere a veces como el "índice", pero también es común referirse a él como el área de ensayo.

2.6.2. Github

GitHub es un hosting online para repositorios utilizado por Git para el mantenimiento y versionado del código fuente, añadiendo una serie de servicios extras para la gestión del proyecto y el código fuente. La parte gratuita de este hosting permite alojar el código en repositorios públicos, si se quiere repositorios privados se debe adquirir la parte "premium" del servicio. [14]

En la actualidad, GitHub es mucho más que un servicio de alojamiento de código. Además de éste, se ofrecen varias herramientas útiles para el trabajo en equipo. Entre ellas, cabe destacar:

- Una wiki para el mantenimiento de las distintas versiones de las páginas.
- Un sistema de seguimiento de problemas que permiten a los miembros de tu equipo detallar un problema con tu software o una sugerencia que deseen hacer.
- Una herramienta de revisión de código, donde se pueden añadir anotaciones en cualquier punto de un fichero y debatir sobre determinados cambios realizados en un commit específico.
- Un visor de ramas donde se pueden comparar los progresos realizados en las distintas ramas de nuestro repositorio.

En el siguiente apartado titulado Lenguajes de programación se describirá el lenguaje de programación a utilizar en la implementación y desarrollo del componente de predicción de cambios, que va a ser explicado con mayor detalle a continuación.

2.6. Lenguaje de programación.

En esta sección analizamos las razones por las cuales se eligió Python para desarrollar el módulo de predicción. Asimismo, se explica el lenguaje y sus características, complementando con la descripción los beneficios que ha aportado el uso de Python a su entorno de desarrollo de software y se ofrecen ejemplos que demuestran que Python es algo más que un lenguaje de secuencia de comandos. Por último, se examina la utilización de bibliotecas de código abierto en y la implementación de las mismas en el proyecto.

2.6.1. Python

Python es un lenguaje de programación dinámico y orientado a objetos, de propósito general en el desarrollo de software. Ofrece gran soporte e integración con otros lenguajes y herramientas, y provee una extensiva cantidad de librerías para el desarrollo de aplicaciones. (Lambert, 2012)

Python se puede ejecutar en Windows, Linux/Unix, Mac OS X, OS/2, Amiga, Palm Handhelds, y teléfonos celulares Nokia. Python también ha sido portado para las máquinas virtuales de Java y .NET. Python es distribuido bajo la licencia open source OSI (La Open Source Initiative es una organización dedicada a la promoción del código abierto.) Que lo hace libre para ser usado inclusive en el desarrollo de productos comerciales. [15]

Filosofía: 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 fueron descritos por el desarrollador de Python Tim Peters en El Zen de Python (referencia) algunos de ellos 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.

Características:

- Lenguaje Interpretado y orientado a objetos
- Multiplataforma
- De sintaxis sencilla.
- permite dividir su programa en módulos reutilizables desde otros programas en Python.
- Incorpora una gran colección de módulos estándar que puedes utilizar como base de los programas (o como ejemplos para empezar a aprender Python).
- hay módulos incluidos que proporcionan E/S de archivos llamadas al sistema, ``sockets" e interfaces gráficas con el usuario.
- Permite escribir programas muy compactos y legibles. Con compactos se quiere expresar que con muy pocas líneas de código se puede lograr diversas funcionalidades. Los programas escritos en Python son normalmente mucho más cortos que sus equivalentes en C o C++, por varios motivos: Los tipos de datos de alto nivel permiten expresar operaciones complejas en una sola sentencia, el agrupamiento de sentencias se realiza mediante sangrado (indentación) en lugar de begin/end o llaves y no son necesarias las declaraciones argumentos y variables. [15]

Utilidades del Lenguaje:

- Para llevar a cabo prototipos del sistema.
- Para elaboración de aplicaciones cliente.
- Para desarrollos web y de sistemas distribuidos.
- Para el desarrollo de tareas científicas, en los que hay que simular y prototipar rápidamente.
- Como primer lenguaje para el aprender.

Es importante tomar en cuenta que no hay un único lenguaje que cubra todos los requerimientos que exija un proyecto, Sin embargo, es recomendable evitar tareas relacionadas con programación de bajo nivel como por ejemplo desarrollo de drivers y kernels, ya que Python es un lenguaje de alto nivel y como consecuencia no hay control directo sobre memoria y otras tareas de bajo nivel.

Velocidad de desarrollo

Python y la comunidad de desarrolladores se extiende por todo el mundo y comprende campos tan variados como el cómputo científico de alto rendimiento, con paquetes como Numpy, o el manejo de centros de datos con miles de servidores a través de proyectos como OpenStack, pasando por la educación, la automatización industrial o la gestión de redes están estrechamente integrados, Elegir Python indica dedicar mayor parte a codificar las funcionalidades y no la interfaz de usuario.

Merece la pena destacar que el tiempo de desarrollo para el programador también se reduce debido a la naturaleza misma de Python. Guido van Rossum, el autor de Python, creó el lenguaje para que resultara sencillo e intuitivo, de código abierto, fácil de comprender y adecuado para las tareas cotidianas. Para usted, el analista SIG convertido en desarrollador o mero aficionado, significa que Python es fácil de aprender y de leer. Dedicará menos tiempo a aprender y más a crear soluciones y mejorar el flujo de trabajo. [16]

Facilidad de implementación

Uno de los aspectos más atractivos de una solución Python es su fácil implementación. No es preciso registrar .dll o ejecutar complicadas instalaciones, ni hay dependencias COM de las que preocuparse. Con la caja de herramientas geodésicas, podemos simplemente comprimir la solución en nuestra oficina y descomprimirla en una ubicación accesible en la red del cliente. Con el código disponible, el cliente sólo debe agregar la caja de herramientas a ArcGIS Desktop para disfrutar de acceso a la funcionalidad.

En muchas grandes corporaciones, la instalación del software es dominio del departamento de TI, y facilitar el software a los empleados de la empresa es su tarea y, con frecuencia, su principal problema. Como proveedor de soluciones, cuanto más sencilla se haga la instalación, más agradecido estará el personal de TI y antes podrán los clientes acceder a la nueva funcionalidad. Para Integrated Informatics, el uso de Python supone que garantizamos el proceso de instalación más sencillo posible y el tiempo de respuesta más breve para nuestros consumidores, lo que se traduce en clientes satisfechos. [16]

Pruebas automatizadas

Se dice que Python viene con "las baterías incluidas", por lo que para las pruebas sólo fue necesario examinar el propio Python para al menos parte de la solución. Python cuenta con un

excelente módulo estándar llamado unittest, que forma parte de la instalación de Python central. Para cada herramienta, existe una secuencia de comandos de prueba independiente con la que se ponen a prueba muchas permutaciones diferentes de parámetros y entradas de datos.

Los conjuntos de prueba individuales para cada herramienta son un buen paso, pero resulta esencial que estas secuencias de comandos de prueba se ejecuten de forma automatizada con regularidad, no sólo cuando se recuerde hacerlo. Últimamente está en boga la noción de "integración continua", la idea de que, después de cada cambio introducido en la base del código, un mecanismo de activación inicia todas las pruebas del conjunto. Esta opción puede resultar excelente para ciertos tipos de base de código y pruebas, sin embargo, no siempre es práctica para las herramientas que realizan las tareas de procesamiento más pesadas, como las pruebas de alta frecuencia. Más importante es la idea de que la prueba se active de forma regular. Puede ser cada noche, semanal o incluso mensualmente, dependiendo de la frecuencia con la que se actualice la base de código. Con la ejecución automatizada del conjunto de pruebas, siempre se estará al tanto de lo que sucede con el código para, en caso de producirse un error en la base de código, saber con rapidez que existe un problema y corregirlo. [16]

Servicios de paquetes de código abierto

Python se ha convertido en uno de los lenguajes de programación de código abierto más populares. De esta forma, los usuarios de Python han creado, literalmente, miles de paquetes de código abierto, muchos de los cuales son directamente aplicables a los tipos de operaciones que uno desea en sus aplicaciones. En la caja de herramientas geodésicas, empleamos muchos paquetes de código abierto para propiciar que nuestros clientes logren sus objetivos.

Como ejemplo, tengamos en cuenta una petición habitual del cliente: crear un informe en pdf del análisis realizado en ArcGIS. Resulta que existe un sencillo paquete de código abierto para varias plataformas disponible llamado ReportLab Toolkit, que tiene la capacidad de crear documentos en pdf. Este paquete contiene completas y eficaces capacidades de manipulación de pdf, además de excelente documentación y un tutorial de iniciación. Gracias a este paquete, pudimos escribir informes y datos en documentos pdf con relativa facilidad y en un tiempo de desarrollo total muy breve. Por lo tanto, la próxima vez que reciba una petición, pregúntese si alguien más ha hecho esto antes y busque en Internet antes de sumergirse directamente en su creación.[16]

La importancia de la licencia

Cuando se encuentra un paquete que hace exactamente lo que uno necesita, lo primero que se debe hacer es leer la licencia. Las licencias de código abierto se proporcionan en diferentes formas, y muchas de ellas están escritas para evitar que el código del software pueda "cerrarse". Es extremadamente importante leer la licencia con mucha atención y asegurarse de que se utiliza correctamente el paquete. En el caso de ReportLab Toolkit, la licencia es una forma de licencia Berkeley Software Distribution (comúnmente conocida como licencia "BSD"). Esta licencia es muy permisiva y autoriza a usar el software y distribuirlo en otro software no libre bajo algunas condiciones. Otras licencias no son tan permisivas y se han diseñado para garantizar que el software que usa otro software de código abierto también es de código abierto (por ejemplo, GPL). Familiarícese con los tipos de licencias más comunes para saber qué puede usar de los paquetes de código abierto y cómo. [16]

Utilización de paquetes de código abierto

SciPy es un ecosistema basado en Python de software de código abierto para las matemáticas, la ciencia y la ingeniería. En particular, estos son algunos de los paquetes principales [17]:

- NumPy, el paquete fundamental para el cálculo numérico. En él se definen los tipos de matriz y de la matriz numérica y operaciones básicas en los datos.
- La biblioteca SciPy, una colección de algoritmos numéricos y cajas de herramientas específicas de dominio, incluyendo el procesamiento de señales, la optimización, estadísticas y mucho más.
- Matplotlib, un paquete de trazado maduro y popular, que ofrece 2D calidad publicación trazado, así como el trazado 3D rudimentaria.
- Pandas, proporcionando alto rendimiento, facilitando el uso de estructuras de datos.
- SymPy, para las matemáticas simbólicas y de álgebra computacional.
- IPython, una rica interfaz interactiva, que le permite procesar rápidamente los datos y las ideas de la prueba. El portátil IPython trabaja en su navegador web, lo que le permite documentar su cálculo en una forma fácilmente reproducible.
- Nose, un marco de código de prueba Python.

Todos estos paquetes de código poseen licencia BSD.

Para centrarnos en el verdadero problema y abarcar de una manera ágil los objetivos generales planteados en la TEG y no preocuparnos por implementar funcionalidades que son de uso común en muchas aplicaciones, como podría ser el proceso de login de usuarios o establecer la conexión con la base de datos, usamos el framework Django que va de la mano con Python.

2.6.2. Django

Es un framework de desarrollo web de código abierto, escrito en Python, que respeta el paradigma conocido como Modelo-Template-Vista (Que será explicado brevemente en el apartado Django y el patrón MTV) Fue desarrollado en origen para gestionar varias páginas orientadas a noticias de la WorldCompany de Lawrence, Kansas, y fue liberada al público bajo una licencia BSD en julio de 2005. [18]

Características:

- Es un framework de desarrollo web
- Código abierto
- Permite construir aplicaciones web más rápido
- Utilizando menos código
- Principio DRY (Don’t Repeat Yourself).
- Legible, casi pseudocódigo

Django, es un *framework* que permite construir aplicaciones web más rápido y con menos código. Fue inicialmente desarrollado para gestionar aplicaciones web de páginas orientadas a noticias de World Online, más tarde se liberó bajo licencia BSD. Django se centra en automatizar todo lo posible y se adhiere al principio DRY (No te repitas a ti mismo).

La meta fundamental de Django es facilitar la creación de sitios web complejo poniendo énfasis en el re-uso, la conectividad y extensibilidad de componentes y el desarrollo rápido. Python es usado en todas las partes del *framework*, incluso en configuraciones, archivos, y en los modelos de datos.

Arquitectura: Aunque Django está fuertemente inspirado en la filosofía de desarrollo Modelo Vista Controlador, sus desarrolladores declaran públicamente que no se sienten especialmente atados a observar estrictamente ningún paradigma particular, y en cambio prefieren hacer "lo que les parece correcto". Como resultado, por ejemplo, lo que se llamaría "controlador" en un

"verdadero" *framework* MVC se llama en Django "vista", y lo que se llamaría "vista" se llama "plantilla". Gracias al poder de las capas mediator y foundation, Django permite que los desarrolladores se dediquen a construir los objetos Entity y la lógica de presentación y control para ellos.

- **Presentación:** Aquí se maneja la interacción entre el usuario y el computador. En Django, ésta tarea la realizan el templateengine y el templateloader que toman la información y la presentan al usuario (vía HTML, por ejemplo). El sistema de configuración de URLs es también parte de la capa de presentación
- **Control:** En esta capa reside el programa o la lógica de aplicación en sí. En Django son representados por las views y manipulators. La capa de presentación depende de ésta y a su vez ésta lo hace de la capa de dominio.
- **Mediator:** Es el encargado de manejar la interacción entre el subsistema Entity y foundation. Aquí se realiza el mapeo objeto-relacional a cargo del motor de Django.
- **Entity:** El subsistema entity maneja los objetos de negocio. El mapeo objeto-relacional de Django permite escribir objetos de tipo entity de una forma fácil y estándar.
- **Foundation:** La principal tarea del subsistema foundation es la de manejar a bajo nivel el trabajo con la base de datos. Se provee soporte a nivel de foundation para varias bases de datos y otras están en etapa de prueba.

Requerimientos: Django requiere Python 2.5 o superior. No se necesitan otras bibliotecas de Python para poder obtener una funcionalidad básica. En un entorno de desarrollo (especialmente si queremos experimentar con Django) no es necesario un web server instalado, ya que Django trae su propio servidor liviano para éste propósito, con la restricción de solo permitir un usuario a la vez.Las Bases de datos que permite conectarse Django sonPostgreSQL, MySQL, Oracle o SQLite.

Django y el Patrón MTV

Django aparenta implementar el patrón MVC, pero el controlador es llamado vista y la vista es llamada plantilla (*template*). Primero, se debe aclarar que al momento de diseñar Django, no se buscó apegarse a nada en particular, sino desarrollar una herramienta que funcione lo mejor posible.

Si bien es cierto que se asemeja mucho a la implementación del patrón MVC, para Django la Vista describe "qué" datos serán presentados y no "cómo" se verán los mismos. Aquí es donde entran en juego las plantillas, que describen "cómo los datos son presentados".[18]

Se dice que el "controller" de un MVC clásico está representado por el propio *framework*. Es decir, el sistema que envía un request a la vista correspondiente, de acuerdo a la configuración de URL de Django (archivo de configuración).

En el caso de querer hacer una correspondencia, entonces diríamos que éste es un *framework* "MTV": modelo, plantilla, vista.

Para la comunicación entre componentes se usan middlewares, uno de los más sencillos basados en estándares web es REST. A continuación se procederá a mencionar y explicar los servicios web y la Transferencia de Estado Representacional (REST)

2.7. REST (Representational State Transfer)

Es una técnica de arquitectura software para sistemas hipermedia distribuidos como la World Wide Web (Marset, 2007), REST está basado en el concepto de representación de recursos , un recurso es cualquier concepto coherente y significativo que pueda ser utilizado y es accedido utilizando un identificador global, un URI es un ejemplo de recurso, para manipular los recursos los componentes de la red (cliente y servidor) se comunican a través de una interfaz estándar (el protocolo HTTP) e intercambian representaciones de estos recursos , un ejemplo de representación de recurso es una página HTML.

Los sistemas que siguen los principios de REST se llaman Restful, e implementan un API web que sigue los siguientes principios:

- El URI es la base para la web API , por ejemplo http://example.com/resources/
- La Web API soporta *internet media type,* con frecuencia se usa JSON pero otros tipos de datos son posibles.
- Soporta un conjunto de operaciones asociados a los métodos HTTP (GET, PUT, POST, DELETE).
- El API debe ser hypertext driven [19]

En base a los métodos seleccionados se realizó una revisión de una serie de herramientas de software libre que soportan dichos métodos y se escogieron las siguientes.

2.8. Heritrix

Heritrix es un rastreador (o *crawler*) de archivos web a través de Internet. Su licencia es *open-source* y está escrito completamente en JAVA. Su interfaz de configuración es accesible usando un navegador Web, también puede ser lanzado desde línea de comandos usando un API REST (a partir de la versión 3).

El Internet Archive [8] y la biblioteca nacional nórdica comenzaron el desarrollo de Heritrix a principios de 2003, con la intención de desarrollar un rastreador con el propósito específico del archivado de sitios Web.

El hecho de que sea código abierto permite la colaboración de organismos similares que necesiten servicio de rastreo, actualmente el rastreador cuenta con 2 versiones estables la 1.1.14 y la 3.x

Arquitectura

Heritrix fue diseñado como un *framework* de rastreo genérico donde diversos componentes pueden ser conectados. Los rastreos son configurables, para ello se eligen y configuran un conjunto de componentes específicos y se ponen en funcionamiento, la ejecución de un rastreo repite el siguiente proceso recursivamente. (este proceso es común en todos los rastreadores Web)

1. Elegir un URI de entre todas las programadas.
2. Buscar el URI.
3. Analizar o archivar los resultados.
4. Seleccionar los URI descubiertos que sean de interés, y sumarlos a los ya programados.
5. Se termina el procesamiento de la URI actual y se repite el proceso.

Los 3 componentes principales de Heritrix son:

1. El Alcance (*Scope*).
2. La Frontera *(Frontier*).
3. La cadena de procesamiento (*ProcessorChains*).

A continuación de describen cada uno de ellos:

- **Alcance:** Se encarga de validar si una URI esta fuera o dentro de las reglas de rastreo, el alcance incluye las semillas: URI que se usan para iniciar el rastreo, el alcance también interviene en la selección de URIs mencionadas en el paso 4 del proceso de rastreo.
- **Frontera:** Es el responsable de seleccionar el siguiente URI a ser procesado, además lleva un registros de las URIs cosechadas y otro de las URIs que ya han sido procesadas, estos registros se implementan con colas.
- **Cadena de procesamiento**: Son un conjunto de procesadores modulares que realizan tareas específicas en cada URI, esto incluye; búsqueda del URI, análisis de los resultados devueltos y pase de URIs descubiertas a la frontera

En la Figura 14 se describe la arquitectura de Heritrix:

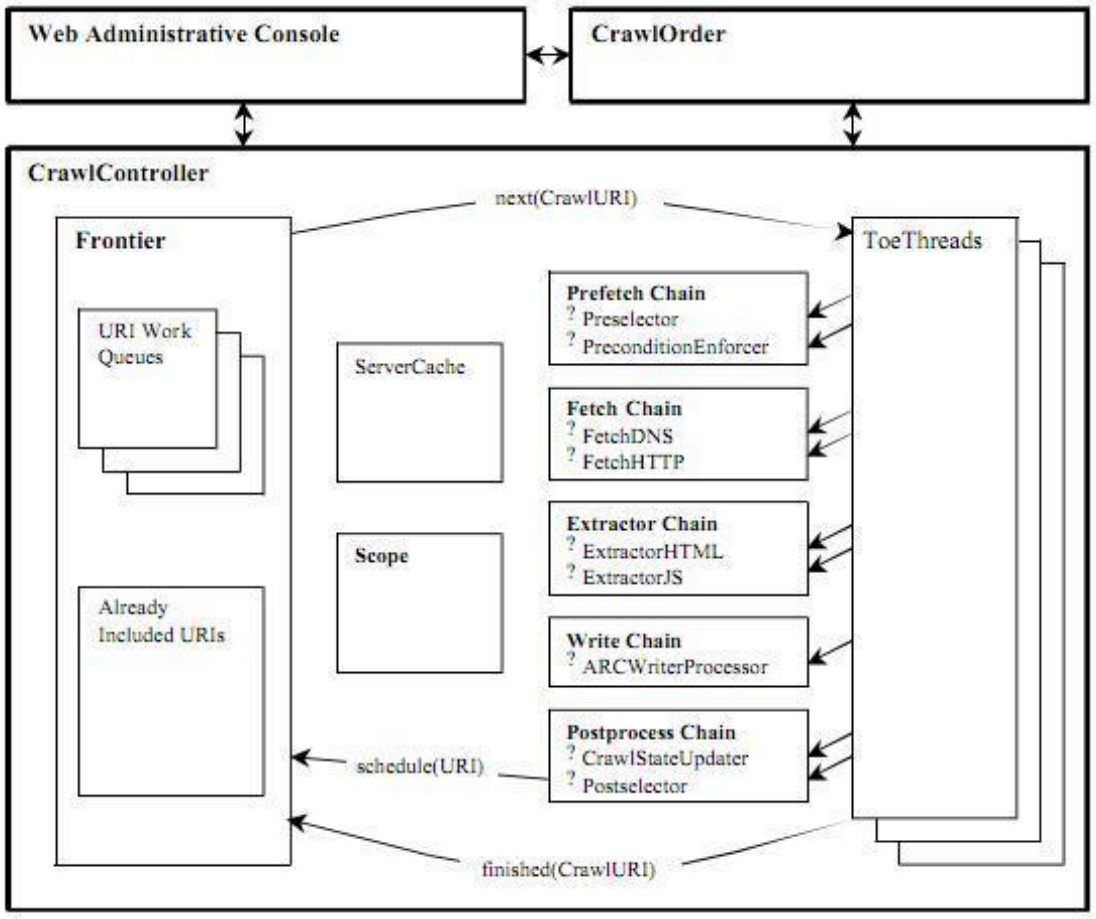


Figura 2-9 Arquitectura de Heritrix [26]

La **Web Administrative Console** puede ser vista como una aplicación independiente (*standalone*), su funcionalidad es la de permitir elegir los componentes de un rastreo y especificar los parámetros de un **Crawlorder,** con la **Web Administrative Console** es posible ver el estado del rastreo actual, sus logs y generar reportes.

En caso de que Heritrix se utilice por línea de comando la **Web Administrative Console** no es necesaria.

Un rastreo es iniciado cuando el **CrawlController** aprueba la **Crawlorder,** el **CrawlController** crea una instancia de todos los módulos necesarios para el rastreo, la **Web AdministrativeConsole** controla el rastreo a través del **CrawlController.**

El **CrawlOrder** contiene suficiente información para crear el ámbito de aplicación. Datos tales como: el alcance, la semilla con la cual el **Frontier** iniciará el rastreo e información de qué hacer con las URIs que sean descubiertas posteriormente.

El **Frontier** es el responsable de ordenar los URIs a ser visitados, así como de asegurarse que las URIs no sean revisadas innecesariamente y moderar las visitas que el rastreador hace a cualquier sitio remoto. Se consiguen estos objetivos mediante el mantenimiento de una serie de colas internas de URIs para ser visitados y una lista de todos los URIs que ya han sido visitados o encolados, por defecto el **Frontier** tiene una implementación de búsqueda en anchura, esto determina las políticas para seleccionar las URIs a procesar, con una opción de preferiblemente terminar el rastreo de los sitios en progreso antes de iniciar con nuevos sitios. Es posible reemplazar la implementación del **Frontier** por una personalizada.

El rastreador Heritrix es multiproceso, por lo que un rastreo se puede ejecutar de manera concurrente. Cada subproceso de trabajo se denomina **ToeThread** y mientras se mantiene un rastreo activo cada **ToeThread** recorre los pasos que corresponden al proceso genérico de rastreo descrito anteriormente.

El número de **ToeThread** ejecutándose en un rastreo se puede ajustar para lograr el máximo rendimiento con los recursos locales. El número de **ToeThread** por lo general oscila en el rango de los cientos.

Cada URI es representada por una instancia de **CrawlURI**, se empaqueta la URI con información adicional recogida durante el procesos de rastreo, los componentes del sistema

comunican su progreso y la salida a través de **CrawlURI,** que lleva los resultados de procesamientos anteriores, para ser revisados posteriormente y finalmente el **CrawlURI** regresa a la frontera para influir en futuros reintentos o programación de rastreos.

El **ServerCache** contiene datos persistentes acerca de los servidores a través de **CrawlURI.** Contiene cualquier número de entidades **CrawlServer** recogiendo información como direcciones IP, política de exclusión de robots, capacidad de respuesta histórica y estadísticas de rastreos del host.

La funcionalidad global de un rastreador con respecto a una URI programada es en gran parte especificada por la serie de procesadores configurados para ejecutarse, cada procesador a su vez realiza sus tareas marcando el estado del **CrawlURI** y retornando, las tareas realizadas varían en función del tipo de URI, historia o el contenido recuperado.

Los procesadores se agrupan en cinco cadenas:

- Procesadores en el PrefetchChain reciben el CrawlURI antes de resolver cualquier actividad de red o de ir a buscar la URI, Por lo general cada procesador demora, reordena o veta la tramitación posterior de una CrawlURI , por ejemplo para asegurar que las políticas de exclusión de robots se captan y son consideradas antes de que la URI sea procesada.
- Procesadores en el FetchChain , la actividad de red intenta adquirir el recurso que se refiere a un CrawlURI, es el caso típico de una transacción HTTP, un procesador de búsqueda llenará la demanda y respuesta en el buffers de la CrawlURI o indicará cualquier condición de error que impida que el buffer sea llenado.
- Procesadores en el ExtractChain realizan seguimiento al procesamiento de un CrawlURI que busca las URIs ya finalizadas, extrayendo característica de interés, por lo general estos son nuevos URIs que también pueden ser elegibles para ser visitados, En este paso las URIs solo se descubren, no se evalúan.
- Procesadores en el WriteChain almacenan el resultado del rastreo –retornando contenido o características importantes – para almacenamiento permanente. El rastreador estándar simplemente escribe los datos en el formato de archivos de Internet, conocidos como archivos ARC o WARC, sin embargo terceros han creado procesadores que escriben los datos en otros formatos o indexan los resultados del rastreo.

- Por último, procesadores en el PostProcessChain realiza el mantenimiento final de las acciones de rastreo en el CrawlURI como las pruebas descubiertas.

Principales Características

Heritrix tiene las siguientes características:

- Recoge contenido vía HTTP recursivamente de múltiples sitios Web en un solo rastreo, con cientos a miles de sitios Web independientes y millones de recursos distintos.
- Rastrea por dominio de sitio, *host* exacto, o patrones configurables de URIs, a partir de una semilla o conjunto de URIs.
- Ubicaciones de salidas ajustables para los registros, archivos comprimidos, informes y archivos temporales.
- Se pueden configurar los bytes máximo de descargas, número máximo de documentos a descargar, y tiempo máximo para pasar el rastreo.
- Número configurable de hilos de rastreo (trabajadores).
- Ajuste de la cota superior del ancho de banda a utilizar.
- Configuración de cortesía que permite establecer un tiempo mínimo/máximo entre las solicitudes.
- Configurable inclusión/exclusión de mecanismos de filtrado, incluye expresiones regulares, profundidad de ruta de URI y vincular los filtros de saltos que pueden ser combinados de diversas maneras y se adjuntan en los puntos clave a los largo de la cadena de procesamiento.
- Por cada rastreo se crean archivos log que permiten verificar problemas, errores, URIs cosechadas, estadísticas, entre otros.
- En la versión 3.x sólo está disponible BdbFrontier como implementación de Frontier, se visitan las URIs y se descubren nuevas páginas aplicando búsqueda en anchura.

- La versión 3.x de Heritrix está basada en el spring java *framework* [14], este framework define unas estructuras llamadas bean, los bean son componentes configurables que constan de propiedades y otros beans, cada beans es representado como un elemento XML, los archivos de configuración de los jobs están conformados por beans, en la Figura 15 se ejemplifica la configuración de un bean.

fetchProcessors Spring Bean

```
<bean id="fetchProcessors" class="org.archive.modules.FetchChain">
<property name="processors">
<list>
<!-- re-check scope, if so enabled... -->
<ref bean="preselector"/>
<!--
...then verify or trigger prerequisite URIs fetched, allow crawling...
-->
<ref bean="preconditions"/>
<!-- ...fetch if DNS URI... -->
<ref bean="fetchDns"/>
<!-- <ref bean="fetchWhois"/> -->
<!-- ...fetch if HTTP URI... -->
<ref bean="fetchHttp"/>
<!-- ...extract outlinks from HTTP headers... -->
<ref bean="extractorHttp"/>
<!-- ...extract outlinks from HTML content... -->
<ref bean="extractorHtml"/>
<!-- ...extract outlinks from CSS content... -->
<ref bean="extractorCss"/>
<!-- ...extract outlinks from Javascript content... -->
<ref bean="extractorJs"/>
<!-- ...extract outlinks from Flash content... -->
<ref bean="extractorSwf"/>
</list>
</property>
</bean>
```

Figura 2-10 Configuración de una semilla.

- API REST: Heritrix 3.1.1 usa REST para exponer sus funcionalidades, la implementación REST de Heritrix está basada en Restlet que es un framework RestFul para java [15] Heritrix expone esta API a través de HTTPS, con este protocolo se hacen peticiones para recuperar o modificar configuraciones y manejos de rastreos. Cualquier cliente que soporte HTTPS puede ser usado para invocar el API REST, ejemplos de clientes de línea de comandos serian curl y wget.

Limitaciones de Heritrix

Las principales limitaciones actuales para tener en cuenta son:

- Oficialmente sólo soportado y probado en Linux.
- Cada ejecución de rastreo es independiente, sin apoyo para programar revisita programada para un área de interés.
- Capacidad limitada de rastreos ante fallas del sistema/hardware.
- Los cambios que se hagan en la configuración de un job que se encuentre en estado *running,* no se reflejarán automáticamente en la configuración en futuros *launch* del job.

2.9. Simuladores

Son objetos de aprendizaje que mediante un programa de software, intentan modelar parte de una réplica de los fenómenos de la realidad y su propósito es que el usuario construya conocimiento a partir del trabajo exploratorio, la inferencia y el aprendizaje por descubrimiento. Los simuladores se desarrollan en un entorno interactivo, que permite al usuario modificar parámetros y ver cómo reacciona el sistema ante el cambio producido. Un simulador permite la simulación de un sistema, reproduciendo su comportamiento.

La interactividad puede definirse como la relación activa que se establece entre el usuario y la computadora. Esta relación permite a las personas observar la relación que existe entre las variables y algunas veces obtener modelos matemáticos sencillos para explicar su comportamiento. El contexto donde se modela la situación o situaciones es un ambiente controlado por una serie de parámetros y en respuesta al modelado se obtienen resultados de interés, que van desde el comportamiento que se generó con la observación. [24]

Capítulo 3. Marco Metodológico

Ágil UP (Agile Unified Process, AUP) es un proceso de ingeniería de software completo basado en la simplificación del Rational Unified Process (RUP) de IBM. Cuenta con un enfoque disciplinado hacia las prácticas de pruebas industriales para el diseño de software y sistemas dentro de una organización de desarrollo.

Este enfoque aplica técnicas ágiles e incluyen desarrollo basado en pruebas (TDD), Modelado basado en desarrollo ágil (AMDD), gestión de cambios ágil, y refactorización de base de datos para mejorar su productividad.

El ciclo de vida de Agile UP es serial en lo grande e iterativo en lo pequeño, liberando entregables incrementales en el tiempo. El ciclo de vida de AUP se muestra en la Figura 3-1.

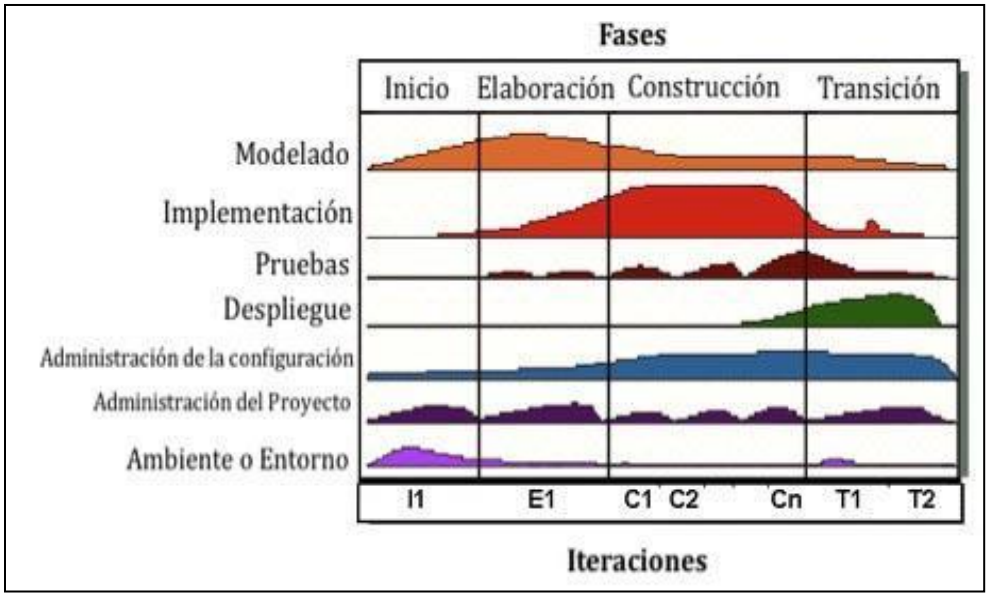


Figura 3-1 Ciclo de vida de Ágil UP [20].

3.1. Disciplinas o Iteraciones.

a) Modelado: La meta de ésta disciplina es entender el negocio de la organización, el dominio del problema que el proyecto aborda e identificar una solución viable para abordar el dominio del problema. No es necesario crear todos los modelos que existan para trabajar en un sistema pero es importante que se tenga en el proyecto aquellos que sean adecuados para la situación Su objetivo es crear modelos que son solamente lo suficientemente buenos para

su situación a la mano, siempre puede ir atrás y mejorar su objetivo después cuando necesite más detalles o la situación de cambios.

b) Implementación: La meta de ésta disciplina es transformar los modelos en un código ejecutable y realizar una prueba de nivel básico en una unidad particular de prueba.

c) Pruebas: La meta de ésta disciplina es ejecutar una evaluación de los objetivos para asegurar la calidad. Esto incluye encontrar defectos, validar que el sistema funcione como fue diseñado y verificar que los requerimientos están completos.

d) Despliegue: La meta de ésta disciplina es planificar la entrega del sistema y ejecutar el plan para que el sistema esté disponible para los usuarios finales.

e) Administración de la Configuración: La meta de ésta disciplina es administrar el acceso a los entregables o productos del proyecto. Esto incluye no sólo el rastreo de versiones del producto en el tiempo, sino que también incluye controlar y administrar los cambios que ocurran.

f) Administración del Proyecto: La meta de ésta disciplina es dirigir las actividades que se llevan a cabo en el proyecto. Esto incluye administración del riesgo, la dirección de personas (asignar tareas, seguimiento de los procesos, etc.), y coordinar con los sistemas y personas fuera del alcance del proyecto para que el este termine a tiempo y dentro del presupuesto.

g) Ambiente o entorno: La meta de ésta disciplina es apoyar el resto de los esfuerzos para garantizar que, el proceso adecuado, la orientación (normas y directrices) y herramientas (hardware, software, etc.) estén disponibles para el equipo según sea necesario.

3.2. Fases

Las fases son implementadas de una forma serial a lo largo de un proyecto de Agile UP. Estas fases son:

1) Inicio: El objetivo principal de la fase de inicio es definir el alcance del proyecto, donde se establecen los límites desde dónde el equipo operará. También se realiza un potencial de arquitectura de la aplicación para así obtener financiamiento y la aceptación de las partes interesadas.

Para esta fase, se muestran las actividades principales que se deben realizar por cada disciplina del esquema AUP (Ver tabla 3-1).

Tabla 3-1 Actividades Principales de las disciplinas en la fase de Inicio

Disciplina	**Actividades Principales**
Modelado	• Modelado de requerimientos de alto nivel (Casos de uso) • Modelado de la arquitectura de alto nivel (Diagrama de componentes)
Implementación	• Prototipo técnico • Prototipo de interfaces de usuario
Pruebas	• Plan de pruebas iniciales • Revisión inicial de prototipo del proyecto • Revisión inicial de modelos
Despliegue	• Identificar la ventana potencial de liberación • Iniciar el plan de despliegue de alto nivel (Diagrama de Despliegue)
Administración de la Configuración	• Establecer la configuración del entorno • Colocar todos los productos bajo el Control de la Configuración.
Administración del Proyecto	• Inicia la creación del equipo. • Crear relaciones con los interesados del proyecto. • Determinar la factibilidad del proyecto. • Determinar un cronograma de alto nivel para el proyecto. • Desarrollar un plan de iteraciones. • Estimar el riesgo.

Disciplina	Actividades Principales
	• Obtenga el apoyo y financiamiento de los interesados.
Entorno	• Establecer el entorno de trabajo • Identificar la categoría del proyecto

2) Elaboración: El principal objetivo de la fase de elaboración es probar la arquitectura del sistema que se va a desarrollar. El punto es asegurar que el equipo puede desarrollar un sistema que pueda satisfacer los requisitos planteados.

En esta fase, el equipo también se prepara para la Fase de Construcción. Se comienza con la creación del ambiente propicio para la Construcción mediante la configuración de hardware, software y herramientas.

Para la fase de Elaboración, se muestran las actividades principales que se deben realizar por cada disciplina del esquema AUP (ver Tabla 3-2):

Tabla 3-2 Actividades Principales de las disciplinas en la fase de Elaboración

Disciplina	Actividades Principales
Modelado	• Identificar los riesgos técnicos • Modelado de la Arquitectura (Diagrama de Componentes) • Prototipo de Interfaces de Usuario
Implementación	• Implementar la arquitectura
Pruebas	• Validar la arquitectura • Evolucionar su modelo de pruebas
Despliegue	• Actualizar su plan de desarrollo

Disciplina	Actividades Principales
Administración de la Configuración	• Poner todos los productos bajo el control de Administración de la Configuración (CM control)
Administración del Proyecto	• Construir el equipo • Obtener los recursos • Actualizar el plan de proyecto
Entorno	• Evolucione el entorno de trabajo • Ajuste los materiales de procesos

3) Construcción: El objetivo de la fase de Construcción consiste en construir software de trabajo, en una base regular e incremental que cumpla las necesidades prioritarias de los interesados en el proyecto.

Para la fase de Construcción, se muestran las actividades principales que se deben realizar por cada disciplina del esquema AUP(ver Tabla 3-3):

Tabla 3-3 Actividades Principales de las disciplinas en la fase de Construcción

Disciplina	Actividades Principales
Modelado	• Análisis del Modelado • Documentación
Implementación	• Primeras pruebas • Evolucionar las interfaces de usuario • Evolucionar el esquema de datos • Desarrollo de interfaces de activos legados
Pruebas	• Pruebas de software • Evolucionar el modelo de pruebas

Disciplina	Actividades Principales
Despliegue	• Desplegar el script de instalación • Desplegar documentación inicial • Actualizar el plan de proyecto • Desplegar el sistema en un ambiente de pre-producción
Administración de la Configuración	• Poner todos los productos bajo el Control CMontrol
Administración del Proyecto	• Administrar el equipo del Proyecto • Manejo del riesgo • Actualizar el plan de proyecto
Entorno	• Evolucionar el entorno de trabajo • Establecer el ambiente de capacitación

4) Transición: Se enfoca en llevar el sistema a producción. Se debe realizar pruebas extensivas a lo largo de esta fase, incluyendo las pruebas beta. Una buena afinación del proyecto tiene lugar en esta fase, incluyendo el trabajo dirigido a los defectos no tan relevantes. El tiempo y esfuerzo necesarios en la Transición varía según el desarrollo del proyecto completo.

Las actividades que se deben realizar para esta fase por cada disciplina del esquema AUP son las siguientes (ver Tabla 3-4):

Tabla 3-4 Actividades Principales de las disciplinas en la fase de Transición

Disciplina	Actividades Principales
Modelado	• Modelado por Lluvia de Ideas • Finalizar la documentación general del sistema.
Implementación	• Corregir defectos

Pruebas	• Validar el Sistema • Validar la documentación • Finalizar su modelo de pruebas
Despliegue	• Finalizar el paquete de entrega o liberación. • Finalizar la documentación • Anunciar el despliegue del proyecto. • Capacitar al personal • Liberar el sistema en producción.
Administración de la Configuración	• Poner todos los productos bajo el CM Control.
Administración del Proyecto	• Administrar el equipo de Proyecto • Inicie el próximo ciclo del proyecto
Entorno	• Establezca las operaciones y / o el ambiente de soporte • Recupere las licencias del software

3.3. Entrega de versiones incrementales en el tiempo

Los equipos de desarrollo AUP suelen emitir revisiones al final de cada iteración en escenarios de reproducción, cosa que no sucede en el enfoque "big bang", donde se cumple con la entrega de todos los programas a la vez. Una versión de desarrollo de una aplicación es algo que podría ser liberado en producción si pasa a través de un módulo de aseguramiento de calidad (también llamado ambiente QA), las pruebas y procesos de despliegue, en pre-producción.

3.4. Arquitectura de software basada en componentes

La arquitectura basada en componentes consiste en una rama de la Ingeniería de software que se enfoca principalmente en la descomposición del software en componentes funcionales [21]. Esta descomposición permite convertir componentes pre-existentes en piezas más grandes de software.

Un Componente es un elemento de un sistema software que ofrece un conjunto de servicios, o funcionalidades, a través de interfaces definidas [21].

La Figura 3-2 muestra la notación UML utilizada para representar un componente, las funcionalidades que el componente provee a través de una interfaz son representadas con un círculo al final de una línea desde el componente (1)

Figura 3-2 Componente de software

Para especificar los servicios que necesita unos componentes se utiliza una interfaz representada con un semicírculo al final de una línea desde el componente (2)

El proceso de construcción de una pieza de software con componentes ya existentes, da origen al principio de reutilización del software, mediante el cual se promueve que los componentes sean implementados de una forma que permita su utilización funcional sobre diferentes sistemas en el futuro.

Un componente de software es similar a una caja negra, en donde para el exterior se tiene una especificación general, la cual es independiente de la especificación interna [21]. Existen tres conceptos ligados a la definición de un componente:

- **Interior del componente:** es una pieza de software que cumple con un conjunto de propiedades y que se encuentra conformada como un artefacto del cual se espera que sea reutilizable.
- **Exterior del componente:** es una interface que cumple con un conjunto de propiedades y provee un servicio a usuarios humanos u otros artefactos de software.
- **Relación interior-exterior:** proceso de relación entre el interior y exterior del componente, aquí son claves conceptos como especificación, implementación y encapsulación.

Existen 4 principios que definen a un componente de software como elemento de la arquitectura [21]:

- **Múltiple uso**: Un componente debe ser significativo es decir, debe estar dentro de un contexto que permita que su funcionalidad sea útil en la creación de distintas piezas de software.
- **Contexto no específico**: La especificación de un componente debe estar planteada forma general que permita su adaptación en distintos sistemas, sin que el contexto tenga prioridad.
- **Encapsulación**: Especificación interna o implementación del componente, no es investigable a través de la interfaz. El resto de componentes o piezas de software dentro de un sistema no se ven afectados por cambios en el diseño de uno de los componentes.
- **Una unidad independiente de desarrollo con su propio control de versiones**: Permite que un componente pueda ser desarrollado de manera independiente, cambiando el diseño o agregando nuevas funcionalidades, sin afectar significativamente el resto del sistema.

Existen 3 aspectos fundamentales respecto a la estructura de la arquitectura basada en componentes

1. **El nombre de los componentes:** Debe identificar las funcionalidades de un componente así como su uso que tiene como software, generalmente se siguen convenciones, estándares para facilitar la identificación de los componentes.
2. **La interfaz de los componentes:** La interfaz puede ser vista como el área de intercambio entre el interior y el exterior de un componente de software, a través de la interfaz es que se puede acceder a las funcionalidades que brinda el componente de software más sin embargo no se puede acceder a sus especificación interna, además de la interfaz se debe proporcionar documentación relacionada a como se debe utilizar el componente.
3. **Cuerpo y código de Implementación:** Es el área del componente donde se encuentra el código o implementación de los servicios y funcionalidades brindados por el componente, debe cumplir con el principio de encapsulación.

En la estructura de la arquitectura basada en componentes existen 2 procesos fundamentales para el desarrollo [21]:

1. **Ensamblajes de sistema a partir de componentes de software**: Este proceso está compuesto por 4 actividades:

- **Análisis de los componentes**: En esta fase de determinar qué tan adecuado es un componte para la construcción del sistema: Esta evaluación se realiza de acuerdo a un conjunto de métricas y criterios establecidos por los analistas y diseñadores de la arquitectura

- **Adaptación de los componentes**: Es posible que el componente esté orientado hacia cierto contexto, atado a funcionalidades específicas , es por ello que en necesario adaptar el componente: Existen 3 maneras de adaptar un componente

  - **White-Box:** cuando el componente debe ser reescrito para operar en conjunto con el resto de componentes del sistema.

  - **Grey-Box**: cuando el componente incorpora su propio API (Programming Interface).

  - **Black-Box**: cuando el componente no posee un API. Una interfaz completamente independiente es construida para acceder a los servicios del componente.

- **Ensamblaje de los componentes:** Se integran los componentes a través de la estructura mediante la cual fueron definidos.
- **Mantenimiento:** Una vez el funcionamiento ya está en funcionamiento puede que sea necesario hacer cambios en los componentes ya sea por nuevos requerimientos o inconvenientes con los mismos, estos cambios pueden consistir en la reescritura o sustitución del componente.

2. **Reusabilidad**: La capacidad de poder utilizar una pieza de software dentro de otro sistema es una de las características más importantes en una arquitectura basada en componentes, para lograr la reutilización es necesario un esfuerzo extra de los desarrolladores basado en lo siguiente:

- Una documentación completa de cada atributo y funcionalidad del componente.
- Una etapa de pruebas organizada y certera que certifique el correcto funcionamiento del componente.
- Una definición de comprobaciones precisa para el chequeo de cada parámetro de entrada (input) del componente.
- Un manejo de notificaciones de errores preciso, que advierta de la existencia de estos de una forma apropiada.
- Desarrollar teniendo en cuenta que el componente puede ser requerido para trabajar en muchos contextos muy diferentes unos de otros (tomar en cuenta la eficiencia, uso de memoria y recursos.

Capítulo 4. Desarrollo de la aplicación

La preservación de contenido siempre ha sido de gran importancia para el ser humano y su desarrollo integral, personal y profesional. La era digital ha hecho que la necesidad de conservar la información publicada en la web aumente, ya que con el pasar de los años y el crecimiento exponencial de la web, las tecnologías asociadas se vuelven obsoletas. Servidores que contienen grandes cantidades de información dejan de estar en línea, caducan o se deterioran a causa de su uso intensivo. Esto produce, en la mayoría de los casos, una pérdida de información que puede ser valiosa. Para minimizar estos riesgos, las herramientas y tecnologías que forman parte de la Web deben pasar por un proceso de adaptaciones y actualización constantes. Es por esto que el Archivo Web en Venezuela y el método de almacenamiento implementado para el mismo, juegan un rol fundamental a la hora de preservar el contenido web, pues se logra determinar la frecuencia de cambio y permite ajustar posibles modificaciones de dicha frecuencia asociadas a los sitios web que son rastreados. Esto puede evitar la pérdida de cambios importantes o la duplicidad de datos. Se propone como Trabajo Especial de Grado la creación del componente para la predicción de frecuencia de cambio del prototipo de archivado de páginas web de Venezuela (desarrollado por la Escuela de Computación bajo el proyecto "Desarrollo de un prototipo de Archivo Web para la preservación del patrimonio Web de Venezuela" de código PI-03-8139-2011/P), el cual consiste en una aplicación Web que permite a los usuarios acceder y monitorear la frecuencia de cambio de cada página. Para lograr esto, se desarrolló un componente que se integrará al módulo de adquisición del prototipo Archivo Web que forma parte de la primera iniciativa formal de preservación web en la Universidad Central de Venezuela.

4.1 Objetivo general de la aplicación

Los módulos desarrollados para el presente Trabajo Especial de Grado tienen como objetivo satisfacer las necesidades del proyecto asignado a la Facultad de Ciencias de la UCV, de realizar la incorporación del componente de predicción para un archivado Web de Venezuela, esta aplicación pertenece al módulos de Adquisición el cual permiten realizar suscripciones de rastreos para cosechar sitios web y almacenarlos.

4.2 Objetivos específicos de la aplicación

- Proporcionar una interfaz administrativa la visualización de métricas asociados a los rastreos.

- Crear el módulo de predicción de cambios.
- Crear el módulo de simulación para crear data experimental.
- Integrar el módulo de predicción al módulo de adquisición y gestión de almacenamiento.
- Crear un sitio Web de consulta de los sitios rastreados, para conocer sus características principales y las métricas asociadas.
- Hacer uso de tecnologías de apoyo para el desarrollo de los módulos correspondientes.

4.3 Alcance de la aplicación

Debido a la proyección que puede tener el desarrollo de los módulos de Adquisición y Almacenamiento y de manera de garantizar una implementación de calidad que sea completamente funcional, se ha desarrollado una primera versión del prototipo con las bases y funcionalidades necesarias para cumplir con los requerimientos.

Se adapta como política que los sitios Web a preservar sean de investigación científica, educativos o en general representativos de la cultura venezolana, sin embargo no se cuenta con un mecanismo que valide que los sitios Web reúnan estas características , queda como tarea del usuario de la aplicación validar que el sitio Web a ser preservado sea de interés.

Se tendrá un espacio limitado para el almacenamiento de los WARC file que hayan sido recolectado a lo largo del proceso de rastreo y el acceso a los documentos en el archivo será a través de direcciones URL.

Se contará dispondrá de una interfaz de usuario para la suscripción de rastreos, también se contará con una interfaz para la visualización de las métricas de los sitios rastreados.

Cabe destacar que el desarrollo del prototipo de preservación Web propuesto en este trabajo especial de grado abarca el módulo de adquisición.

4.4. Adaptación de la metodología AUP usando una arquitectura de software basada en componentes

Para el desarrollo del prototipo de Archivado Web se implementaron las fases que sigue la metodología de Proceso Unificado Ágil (Agile Unified Process, AUP). Esta sigue los principios de la Modelación Ágil y nos da la oportunidad de crear de manera fácil, rápida y sencilla la documentación del proyecto adaptada a una arquitectura de software basada en componentes.

A continuación, se describe la adaptación de las fases del Proceso Unificado Ágil (AUP) del presente trabajo especial de grado.

4.4.1. Fase de inicio.

En esta fase, como se ha descrito anteriormente se indican los requerimientos, objetivos y alcance de la aplicación, algunos de estos aspectos ya han sido detallados en secciones anteriores del documento.

4.4.1.1. Principales requerimientos:

- Asignar, modificar y calcular una frecuencia de cambio para cada página web.
- Agrupar todas aquellas páginas web que presenten comportamientos similares en intervalos de tiempos de observación.
- Establecer la comunicación entre la herramienta de programación para el acceso Python-Django y la Base de datos.
- Construir un módulo de búsqueda de páginas Web a través del URL que solicite el usuario.
- Implementar un módulo que le permita al usuario monitorear la frecuencia de cambio asociada a cada página web.

4.4.1.2. Principales funcionalidades o herramientas provistas por la aplicación

- Se proveerá una interfaz para el uso de la aplicación Web por parte de los usuarios.
- Se proporcionará una herramienta para que el usuario pueda visualizar listas organizadas por grupos según su comportamiento y frecuencia de cambio.

4.4.1.3. Usuario del Sistema

El módulo de acceso y control de frecuencia para el prototipo de archivo Web en Venezuela, al ser una aplicación Web, estará dirigido a los siguientes actores: (Ospina, 2014)

- **Suscriptor:** Rol desempeñado por las personas o los clientes, que proporcionan la información a ser conservada. Toman decisiones para incluir o excluir elementos (semillas) o grupos de elementos (colecciones) en cada etapa de flujo, desde la adquisición hasta el almacenamiento. Tienen la responsabilidad de cumplir la política de selección.
- **Usuario Final:** Rol desempeñado por las personas o los sistemas cliente, que interactúa con los servicios del Archivo para encontrar y adquirir información conservada de interés y estadísticas acerca de las métricas recolectadas.
- **Director:** Responsable del manejo de los componentes funcionales, análisis de riesgos costos y definición de las políticas del Archivo a un nivel superior, así como de la coordinación entre administradores y suscriptores.
- **Administrador:** Técnicos u operadores de rastreo que controlan el flujo de trabajo y su operación diaria, su tarea es desarrollar, construir, mantener y controlar el flujo del trabajo del Archivo. Arquitectura de información y módulos.

4.4.1.4. Arquitectura de la aplicación

En este trabajo se propone seguir la arquitectura general para Archivos Web el cual puede apreciarse a continuación en la figura 4-1, planteada por la Prof. Mercy Ospina (Ospina, 2014).

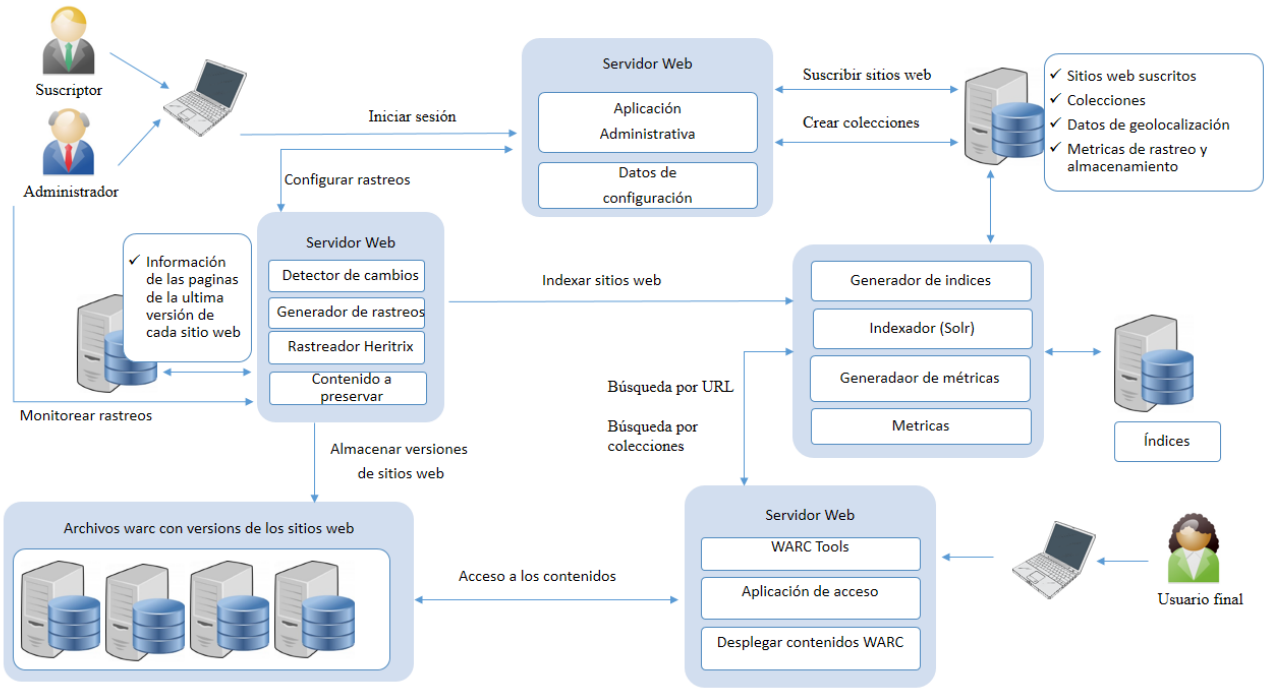


Figura 4-1 Arquitectura de información modelo tradicional (Ospina, 2014)

En la siguiente gráfica (Figura 4-2) se incluye en la arquitectura un componente adicional (el componente se puede diferenciar de los demás por tener un fondo amarillo) que forma parte del módulo de adquisición y permite como primer objetivo establecer una frecuencia de adquisición para los diferentes sitios web suscritos.

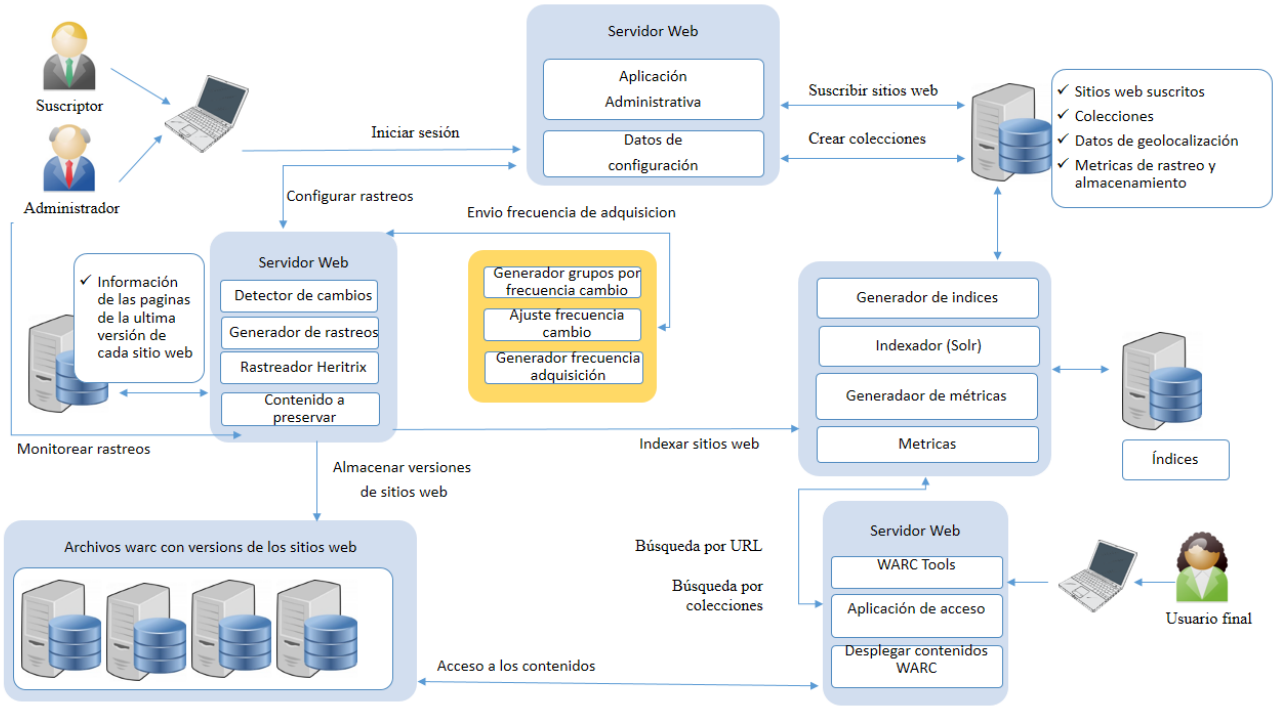


Figura 4-2 Arquitectura de información modelo Actual (Ospina, 2014)

4.4.1.5. Tecnologías, lenguajes, modelos, patrones o estándares a ser incorporados

Las tecnologías a utilizar en el desarrollo del componente de predicción de cambios para el prototipo de Archivo Web en Venezuela son:

- REST como arquitectura de software para la transferencia de datos.
- HTTP como protocolo de comunicación e integración del prototipo.
- Python como lenguaje de programación, con su framework Django.
- Arquitectura cliente/servidor.
- El patrón arquitectónico Modelo Vista Template (MTV).
- Se utilizará un Servidor HP ProLiant ML110 G7 E3-1220 1P, con procesador Intel QuadCoreXeon de 3.1GHz y 10 GB de RAM.

4.4.1.6. Entorno de trabajo

Luego de realizar las gestiones pertinentes, se obtuvieron los recursos necesarios para adquirir el hardware donde corre el prototipo de preservación web. Se cuenta con un servidor que fue dividido en cinco máquinas virtuales. Adicionalmente, fue adquirido un disco duro de 1 Terabyte que fue dividido y asignado a cuatro de las máquinas virtuales. Este espacio de almacenamiento es necesario para almacenar los archivos WARC cosechados. Desde este mismo momento, se toma en cuenta la escalabilidad del sistema desde el punto de vista del espacio de almacenamiento; para determinar si se requiere más espacio y la posibilidad de integrar discos duros adicionales.

A cada una de las máquinas virtuales se le asignó un nombre y una dirección IP:

- ImagenSolr - 190.169.69.154
- Heritrix1 - 190.169.69.155
- Heritrix2 - 190.169.69.156
- Heritrix3 - 190.169.69.157
- Heritrix4 - 190.169.69.158

El software requerido y/o seleccionado es el siguiente:

- Sistema Operativo Linux-Debian.
- Lenguaje de programación Python y su framework Django.
- Rastreador Heritrix.
- Servidor de base de datos MySQL.

4.4.2. Fase de Elaboración

En esta fase se diseña y modela la arquitectura del sistema la cual debe dar soporte para lograr la implementación de todos los requerimientos.

4.4.2.1. Prototipos de interfaz de usuario.

A continuación se muestran los prototipos de interfaces de usuario diseñados para el sistema.

Iniciar sesión

Como se puede observar en la Figura 4-3 se cuenta con un formulario que tiene 2 campos que el usuario debe completar para iniciar sesión en el sistema.

Figura 4-3 Prototipo de interfaz Iniciar Sesión

Lista de solicitudes de rastreo

En esta interfaz se puede visualizar de forma tabular los datos de las solicitudes de rastreo que el usuario ha creado, esto se puede apreciar en la Figura 4-4, adicionalmente se cuenta con un botón para visualizar las métricas asociadas a cada rastreo.

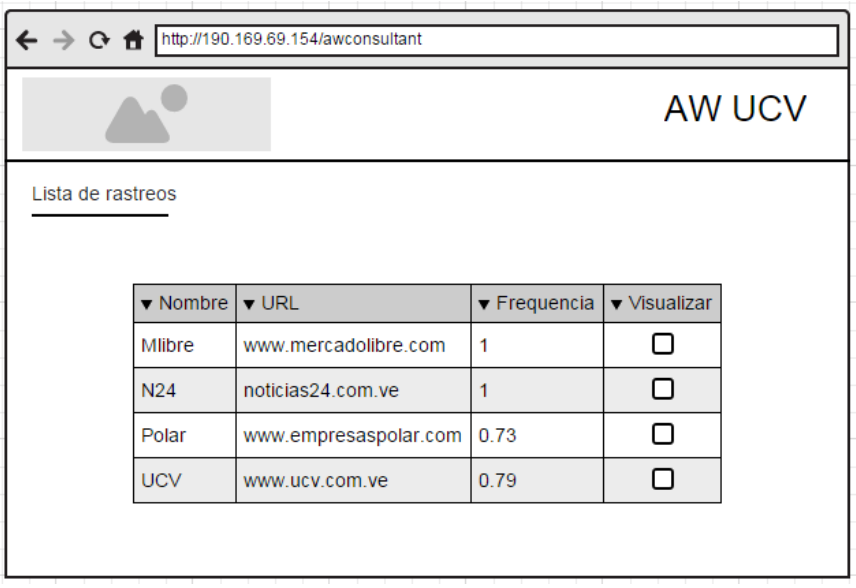


Figura 4-4 Prototipo de interfaz Lista de solicitudes de rastreo

Listar Métricas de Adquisición

En la Figura 4-5 se aprecia una lista donde se ubica la información pertinente con las métricas del proceso de adquisición

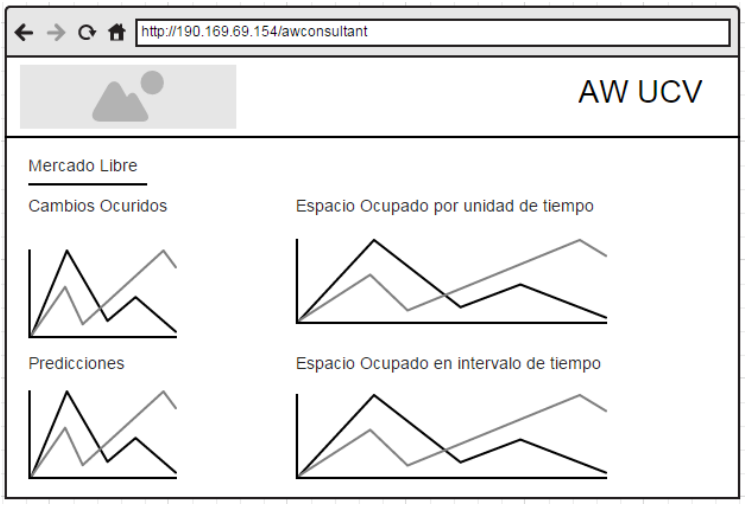


Figura 4-5 Listar métricas de adquisición

Lista de solicitudes de rastreo

En esta interfaz se puede visualizar de forma tabular los datos de las solicitudes de rastreo agrupadas por frecuencia de cambio.

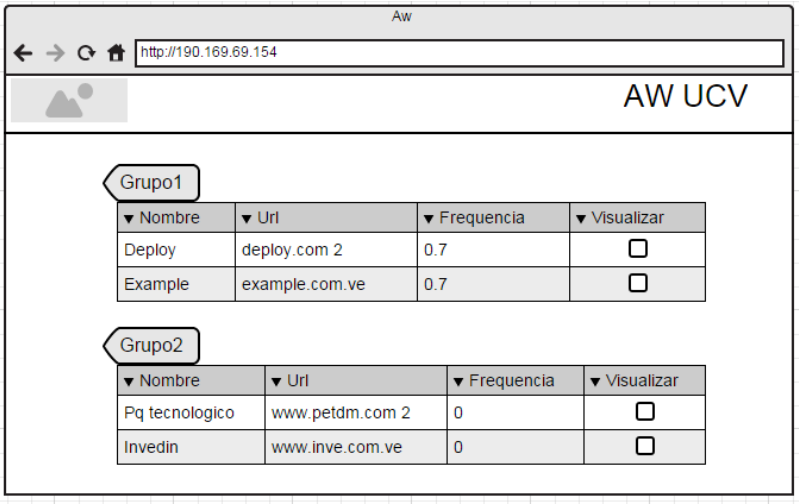


Figura 4-6 Lista de rastreos agrupados por frecuencia de cambio.

4.4.2.2. Diagramas de componentes

Como se ha comentado anteriormente se decidió utilizar una arquitectura basada en componentes, en la Figura 4-7 se puede apreciar el diagrama de componentes del módulo de adquisición.

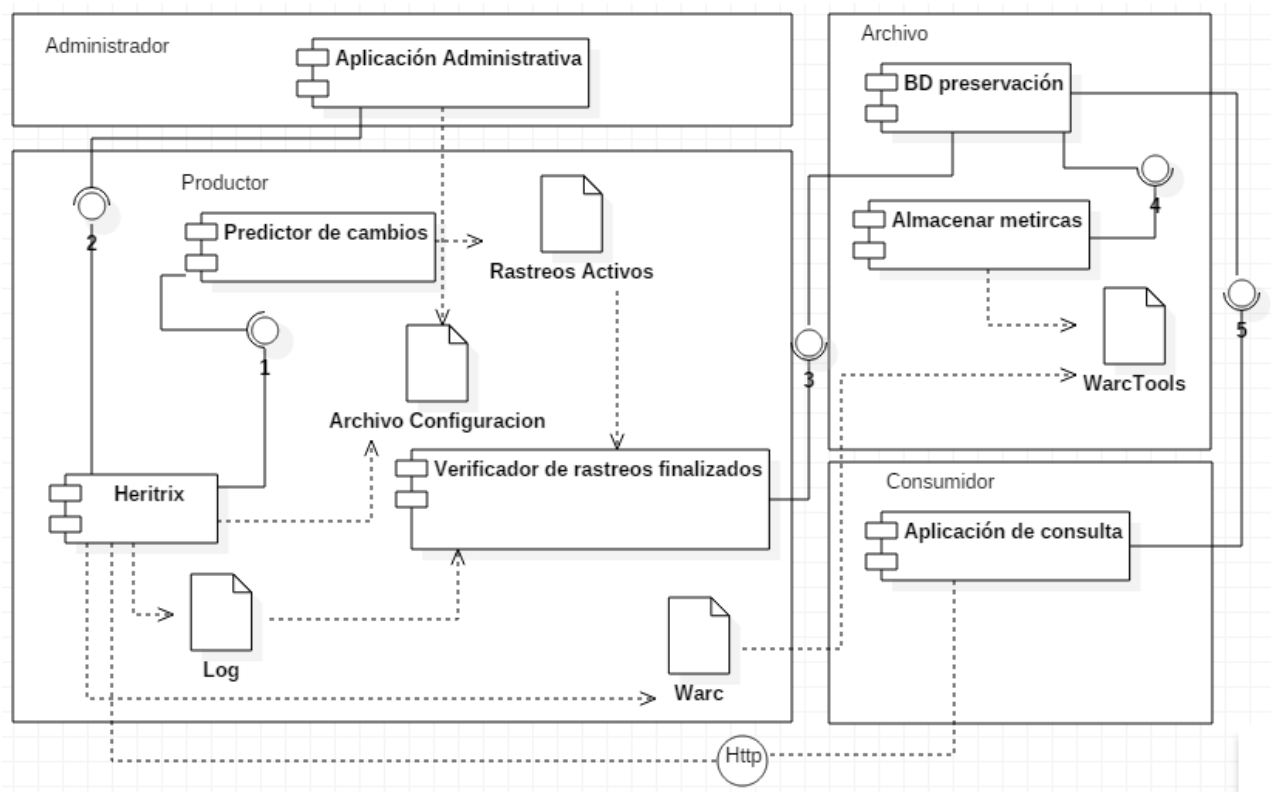


Figura 4-7 diagrama de componentes del módulo de adquisición.

En la Figura 4-8 se puede observar el diagrama de componentes de todo el sistema, esta arquitectura sigue el modelo de referencia de la IIPC que se mostró en la Figura 8 del presente documento.

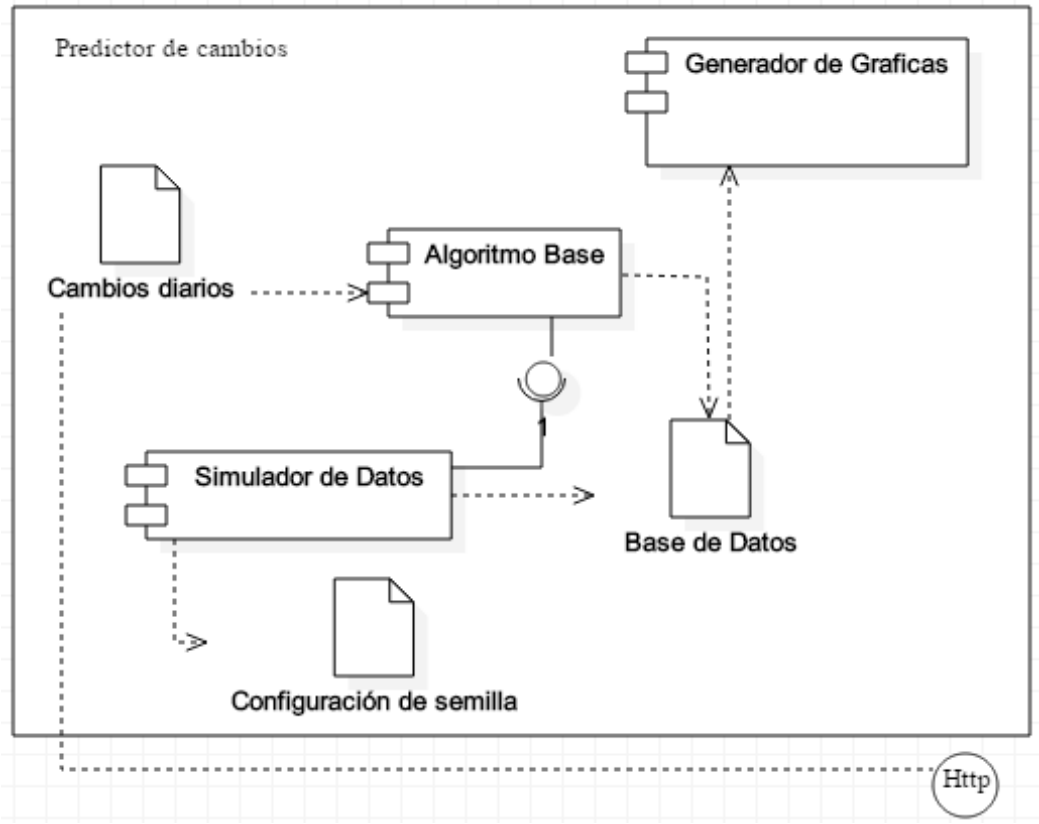


Figura 4-8 Diagrama de componentes de módulo de predicción.

4.4.2.3. Diagramas de actividades

En esta sección se describe las principales actividades que realiza el sistema

Crear rastreo

En la Figura 4-9 se muestran todos los pasos necesarios para crear una solicitud de rastreo

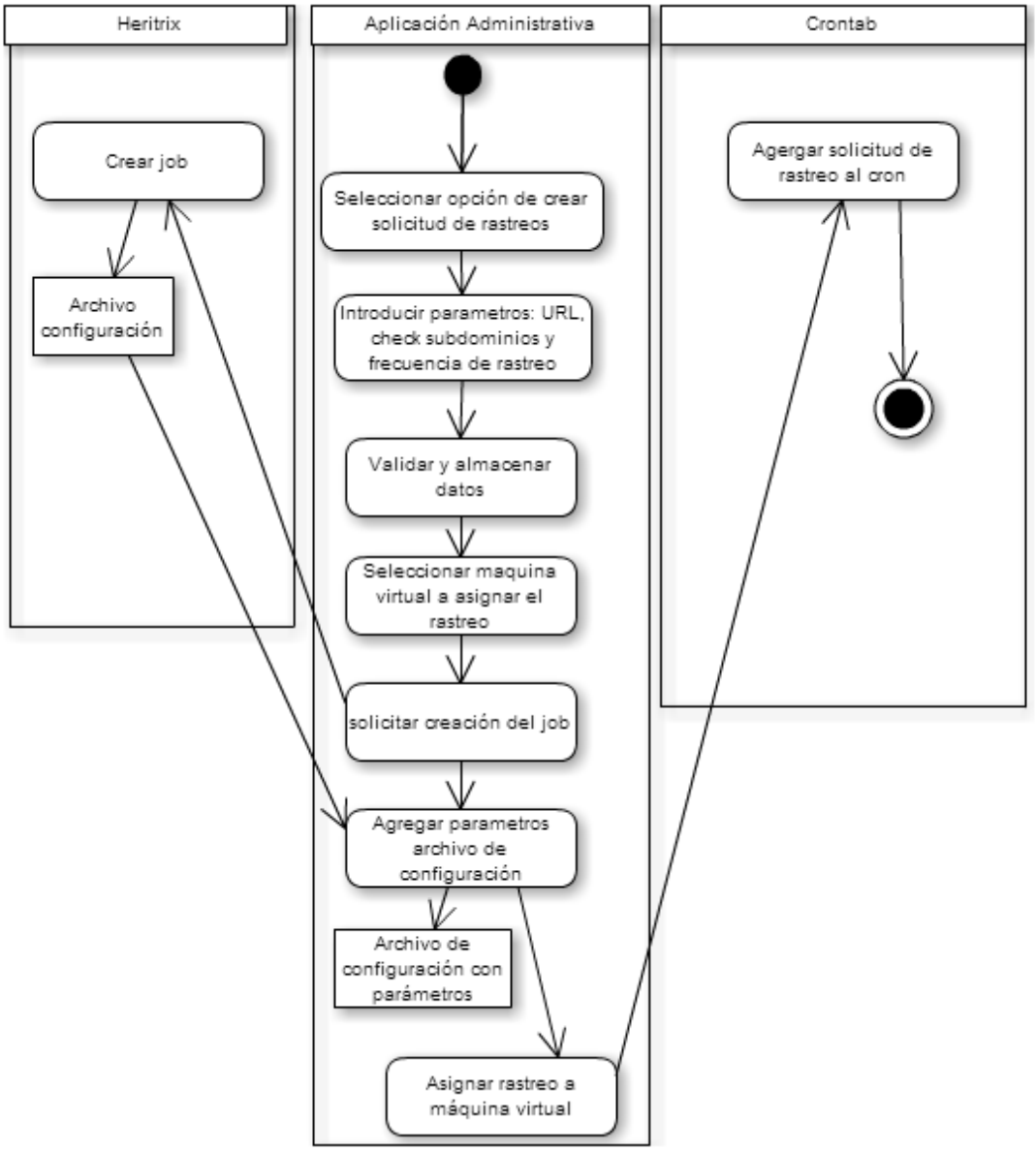


Figura 4-9 Diagrama de actividades Crear rastreo.

Generar rastreo

Una vez se encuentra registrada una solicitud de rastreo debe ser colocada en ejecución, en la Figura 4-10 se pueden observar los pasos necesarios para completar esta actividad

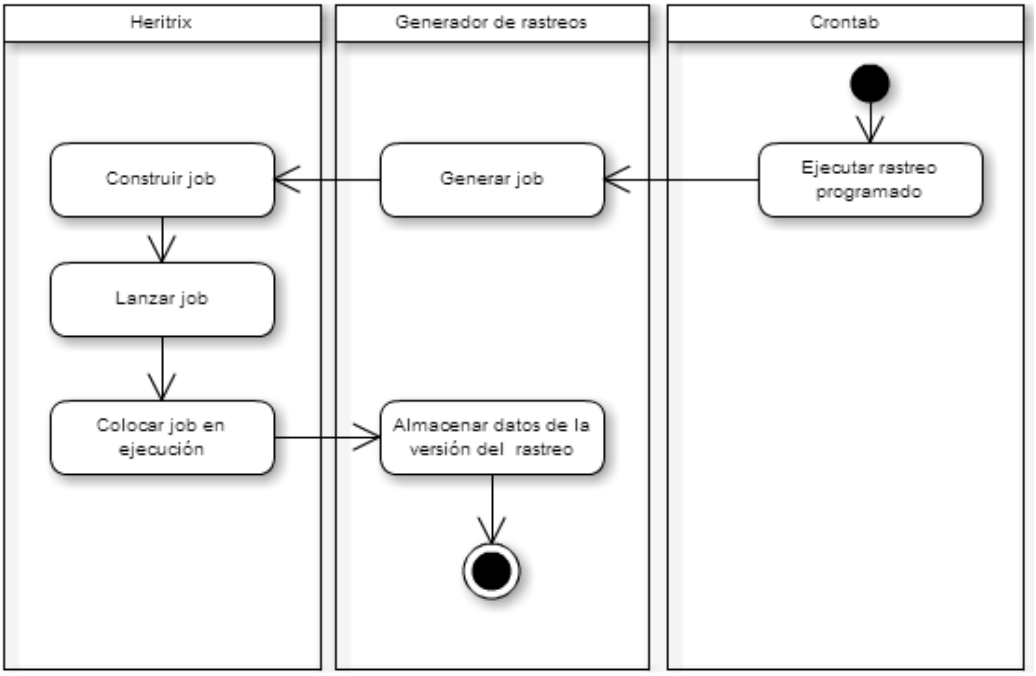


Figura 4-10 Diagrama de actividades Generar rastreo

Verificar fin de rastreo

Para verificar que rastreos han finalizado se siguen la serie de pasos mostrados en la Figura 4-11.

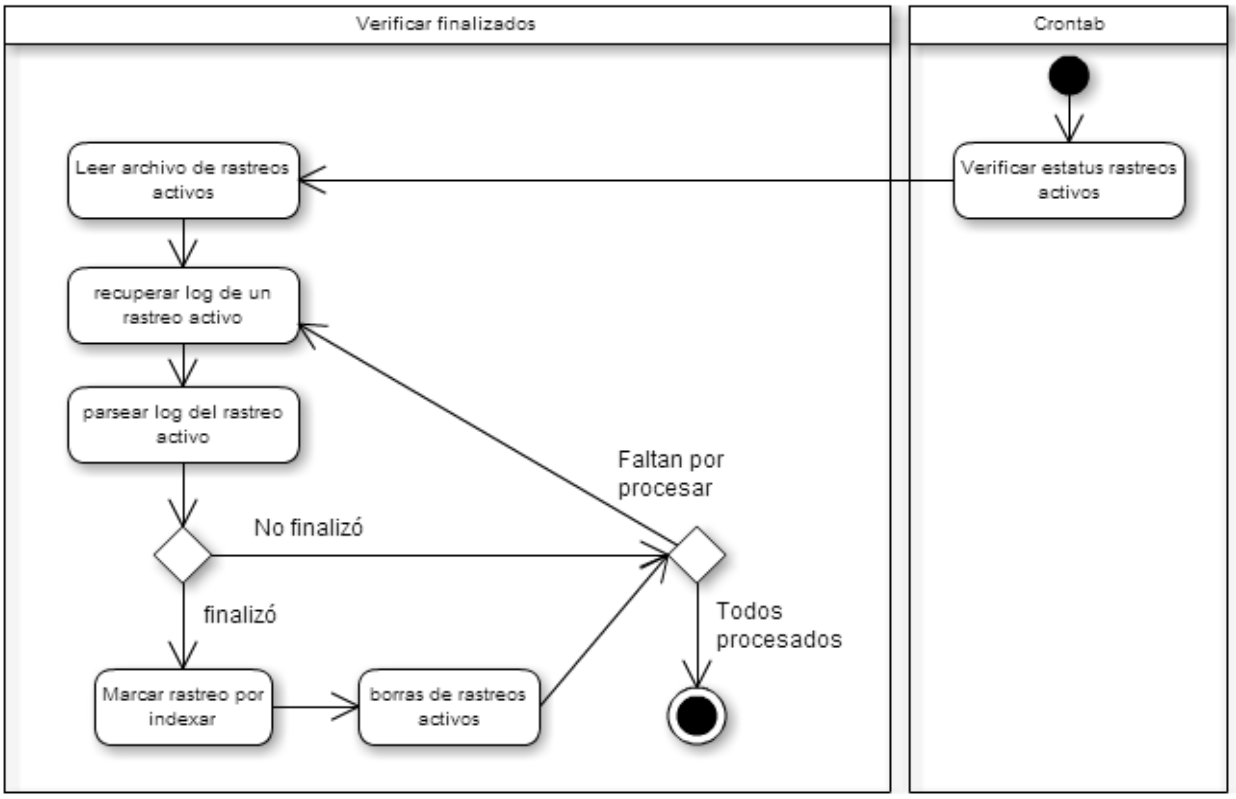


Figura 4-11 Diagrama de actividades Verificar fin de rastreo

4.4.2.4. Casos de uso

Por cuestiones de legibilidad y mayor entendimiento se decidió separar los casos de uso por módulo ver Figura 4-12 hasta la 4-16.

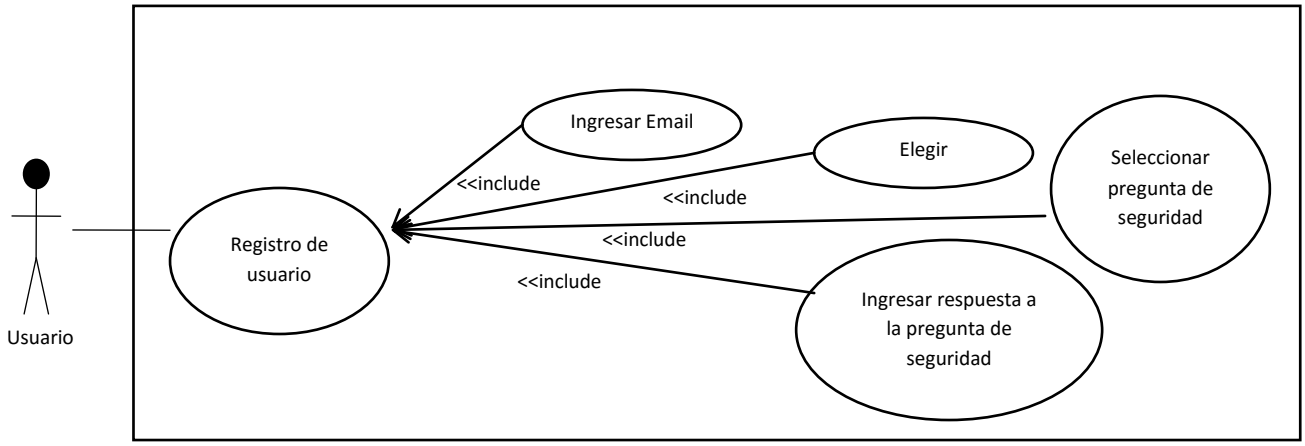


Figura 4-12 Caso de uso registro de usuario.

Nombre:	Registro de usuario [Figura 4-12]
Autor:	Willibert Carabali
Número de CU	CU-1

Descripción:
Permite al usuario el registrase en el sistema para que pueda acceder a la aplicación

Actores:
Usuario

Pre-condiciones:
Ninguna

Flujo Normal:

1. El usuario debe ingresar su email.
2. El usuario debe elegir e ingresar su contraseña.
3. El usuario debe seleccionar una pregunta de seguridad la posible recuperación de su contraseña.
4. El usuario debe ingresar la respuesta a la pregunta de seguridad.

Flujo Alternativo:
Ninguno

Post-condiciones:
El usuario logra el registro con éxito y ya es capaz de ingresar y hacer uso de la aplicación

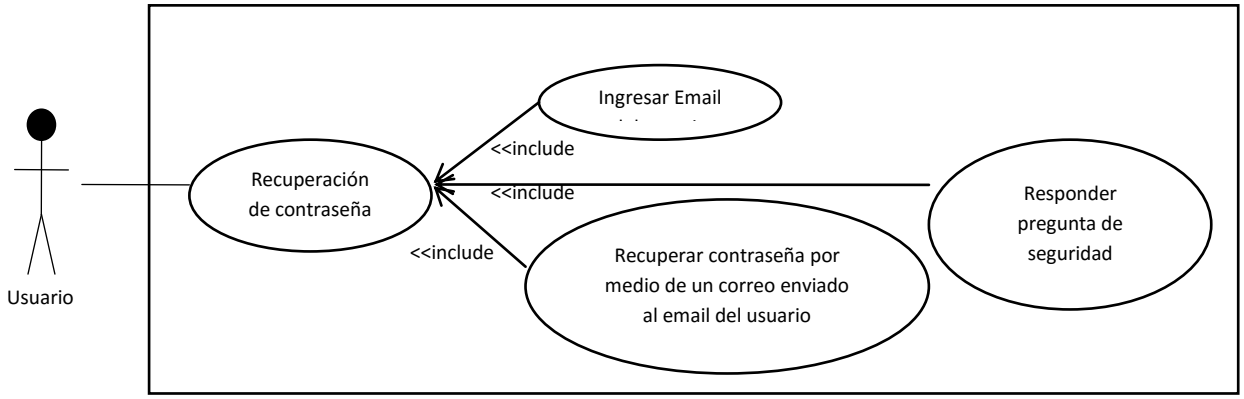


Figura 4-13 Recuperación de contraseña.

Nombre:	Recuperación de contraseña. [Figura 4-13]
Autor:	Willibert Carabali
Número de CU	CU-2

Descripción:
Permite al usuario recuperar su contraseña en caso de haberla olvidado.
Actores:
Usuario
Pre-condiciones:
CU-1
Flujo Normal:

1. El usuario debe ingresar su email.
2. El usuario debe responder correctamente a la pregunta de seguridad.
3. El usuario debe finalizar la recuperación de su contraseña por medio de un correo enviado a su email asociado.

Flujo Alternativo:
Ninguno
Post-condiciones:
El usuario logra recuperar su contraseña.

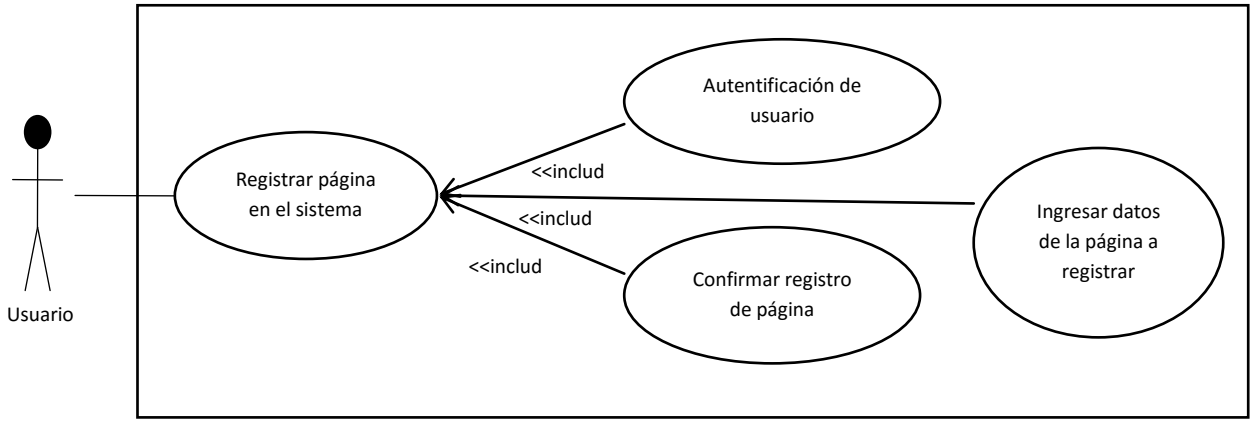


Figura 4-14 Casos de uso registrar página en el sistema

Nombre:	Registrar página en el sistema. [Figura 4-14]
Autor:	Willibert Carabali
Número de CU	CU-3

Descripción:
Permite al usuario el registro de una nueva página en el sistema.
Actores:
Usuario

Pre-condiciones:
CU-1
Flujo Normal:
1. El usuario ingresar al sistema por medio de una autentificación.
2. El usuario debe ingresar los datos de la página a registrar.
3. El usuario confirma el registro de la página.

Flujo Alternativo:
Ninguno
Post-condiciones:
El usuario logra registrar una nueva página en el sistema.

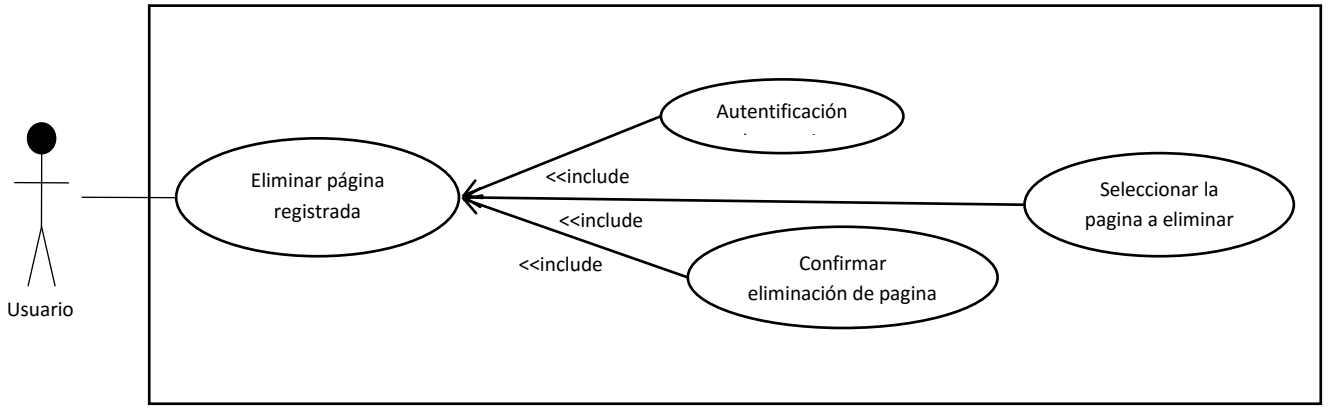


Figura 4-15 Eliminar página registrada.

Nombre:	Eliminar página registrada. [Figura 4-15]
Autor:	Willibert Carabali
Número de CU	CU-4

Descripción:
Permite al usuario la eliminación de una página en el sistema.
Actores:
Usuario
Pre-condiciones:
CU-3
Flujo Normal:
1. El usuario ingresar al sistema por medio de una autentificación.
2. El usuario debe seleccionar la página a eliminar.
3. El usuario confirma la eliminación de la página.

Flujo Alternativo:
Ninguno
Post-condiciones:
El usuario logra la eliminación de la página seleccionada del sistema.

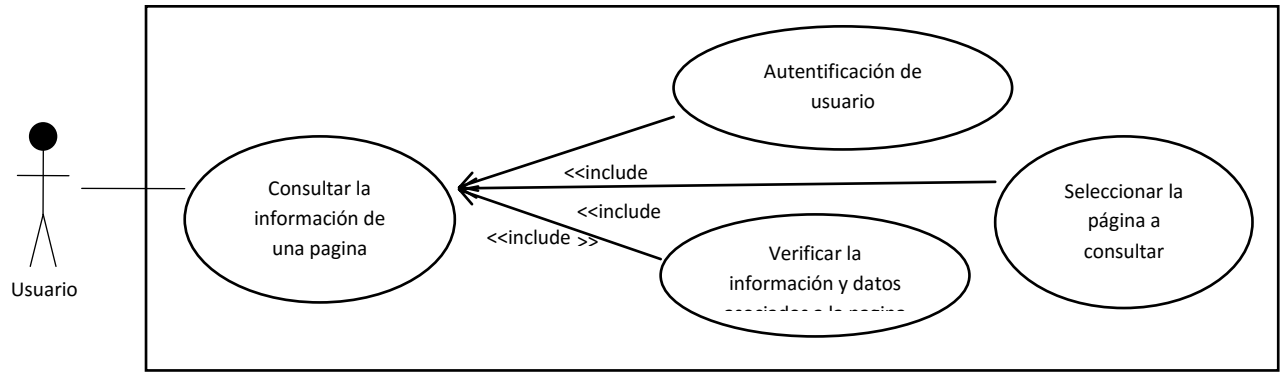


Figura 4-16 Consultar la información de una pagina

Nombre:	Consultar Información de una página. [Figura 4-16]
Autor:	Willibert Carabali
Número de CU	CU-5

Descripción:
Permite al usuario consultar la información y datos asociados a una página.

Actores:
Usuario

Pre-condiciones:
CU-3

Flujo Normal:

1. El usuario ingresar al sistema por medio de una autentificación.
2. El usuario debe seleccionar la página a consultar entre las que están registradas en la aplicación.
3. El usuario visualiza la información y datos de la página seleccionada.

Flujo Alternativo:
Ninguno

Post-condiciones:
El usuario logra consultar la información de la página seleccionada.

4.4.2.5. Diagramas de secuencia

Por cada caso de uso se diseñó un diagrama de secuencia para ilustrar de mejor manera su funcionamiento, al igual que los casos de uso fueron separados por módulos.

Diagramas de secuencia módulo de adquisición

Los diagramas pertenecientes al módulo de adquisición son los siguientes:

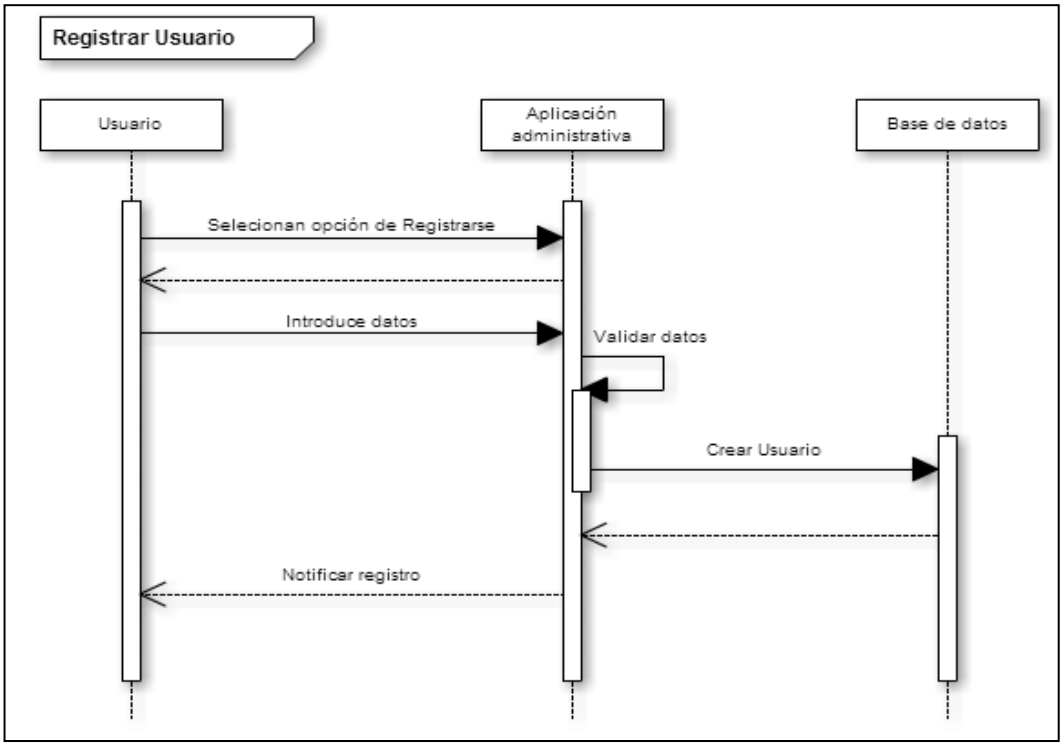


Figura 4-17 Diagrama de secuencia registrar usuario

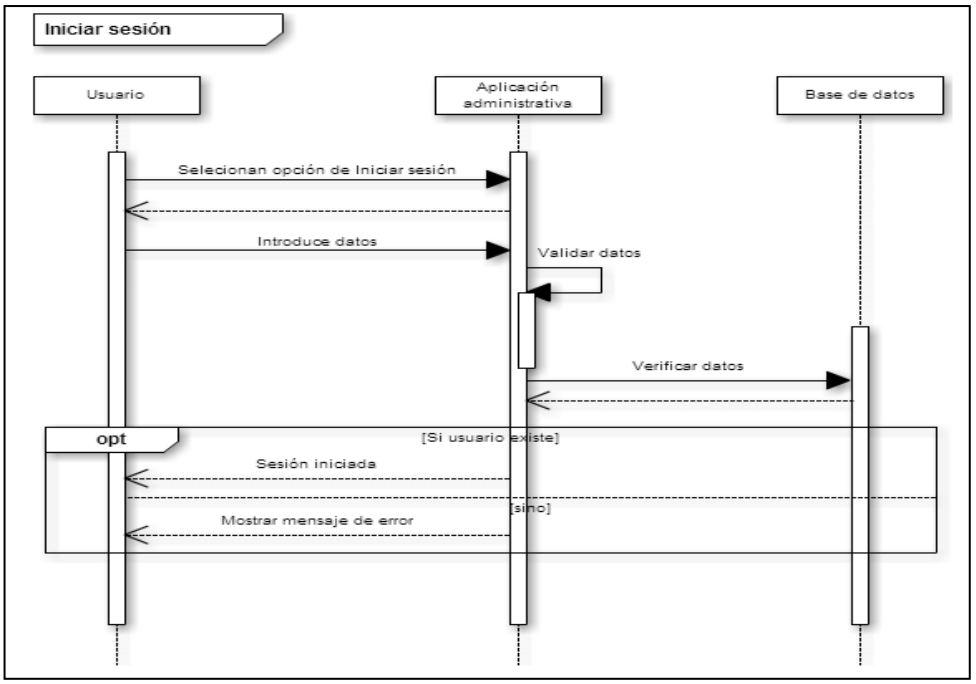


Figura 4-18 Diagrama de secuencia Iniciar Sesión

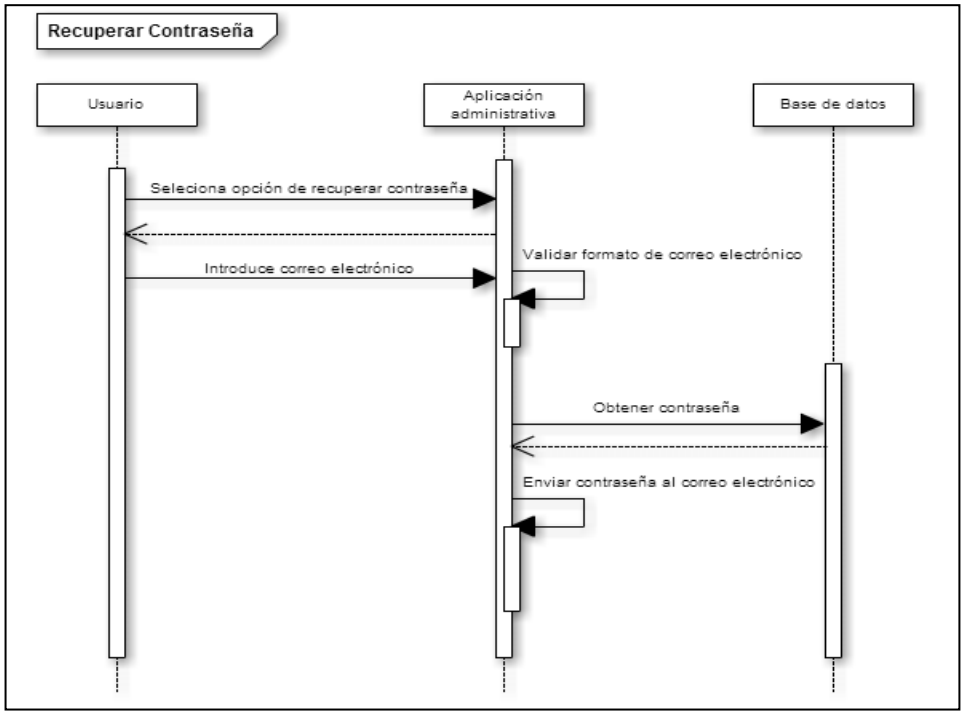


Figura 4-19 Diagrama de secuencia Recuperar Contraseña.

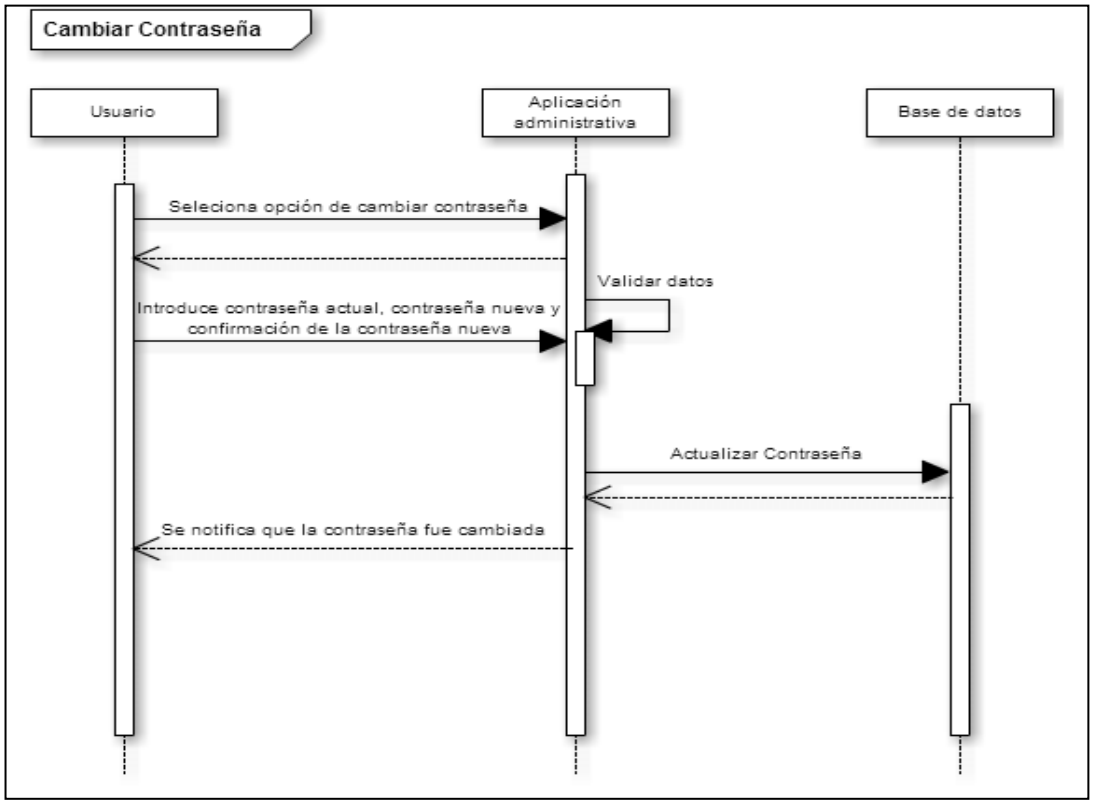


Figura 4-19 Diagrama de secuencia Cambiar Contraseña

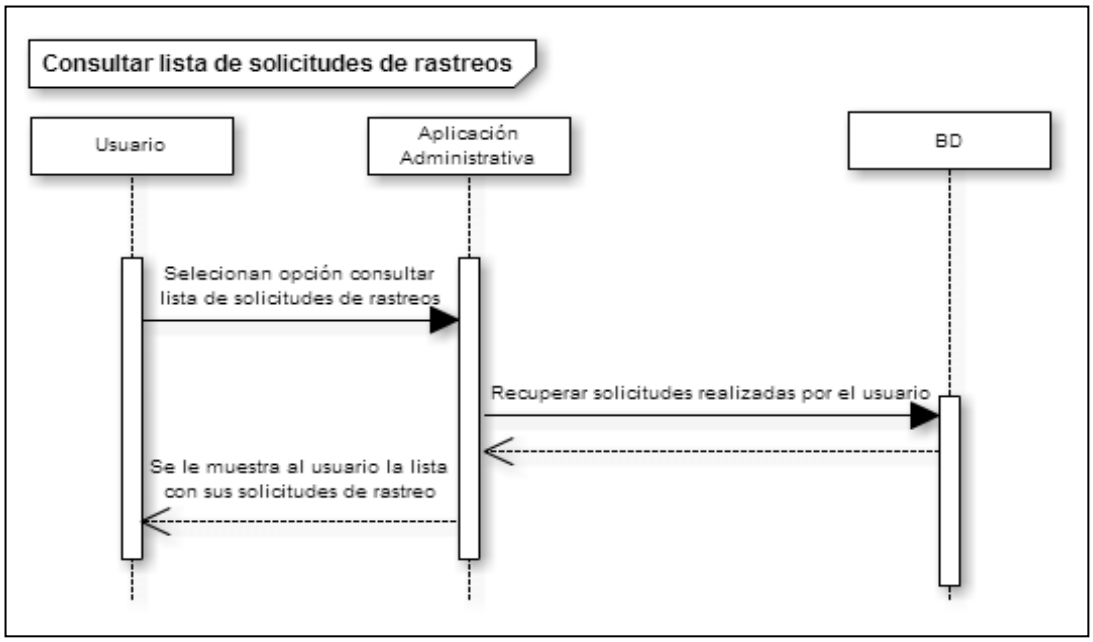


Figura 4-21 Diagrama de secuencia Consultar lista de solicitudes de rastreos

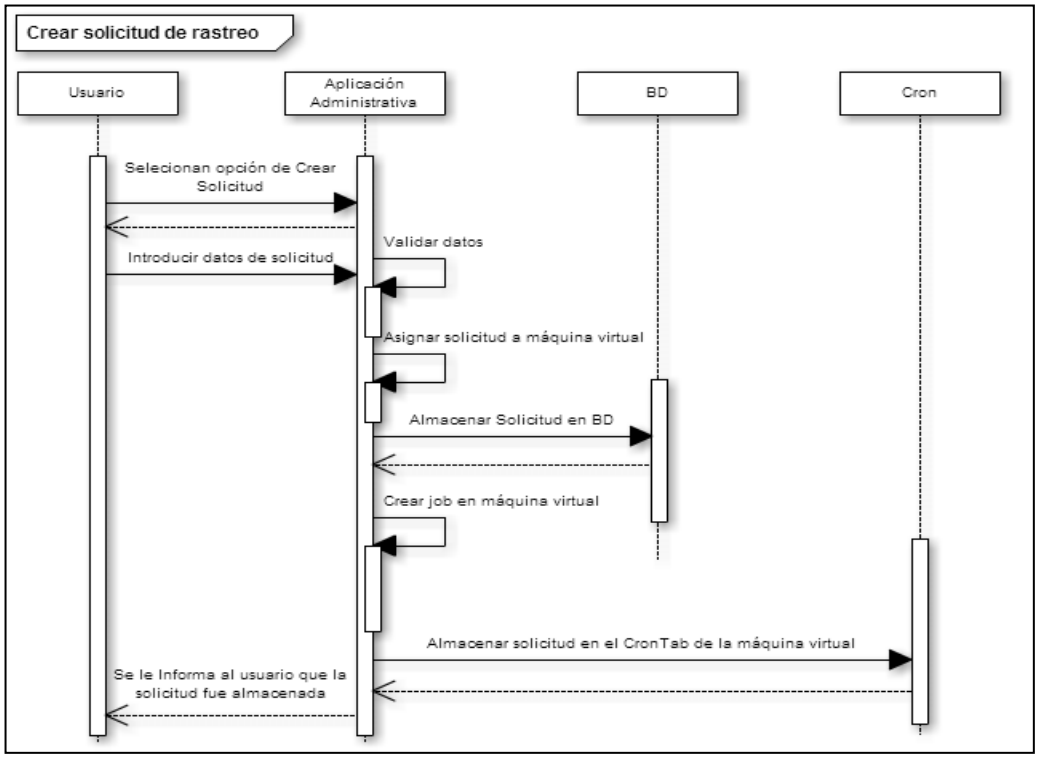


Figura 4-22 Diagrama de secuencia Crear Solicitud de Rastreo

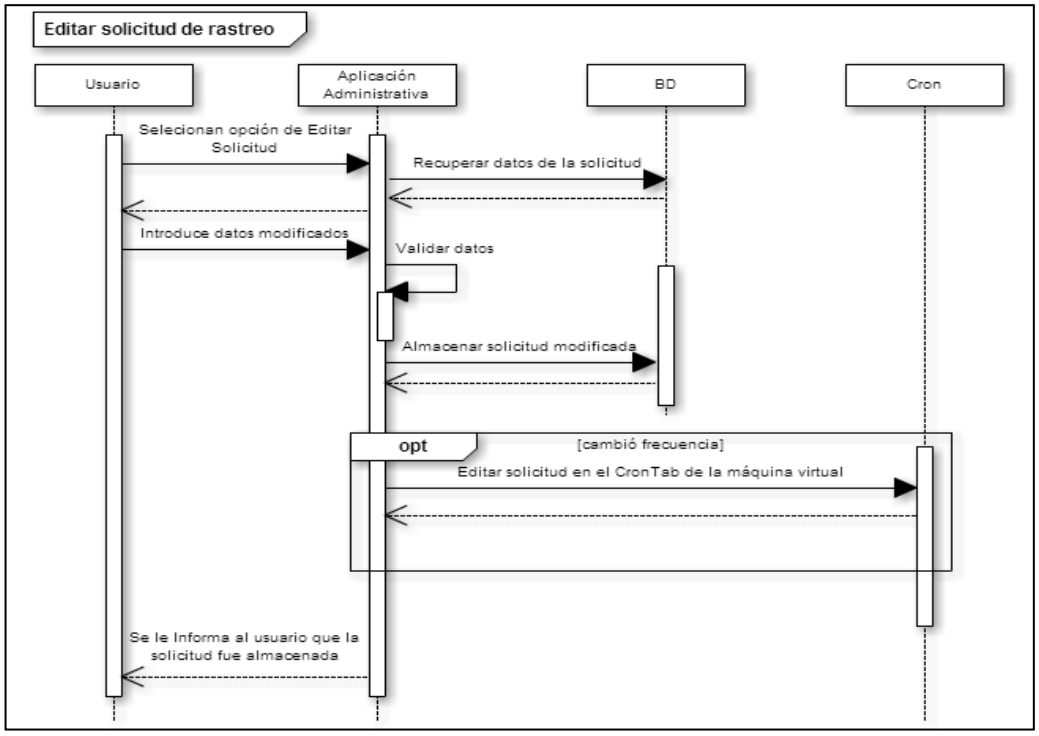


Figura 4-23 Diagrama de secuencia Editar Solicitud de Rastreo

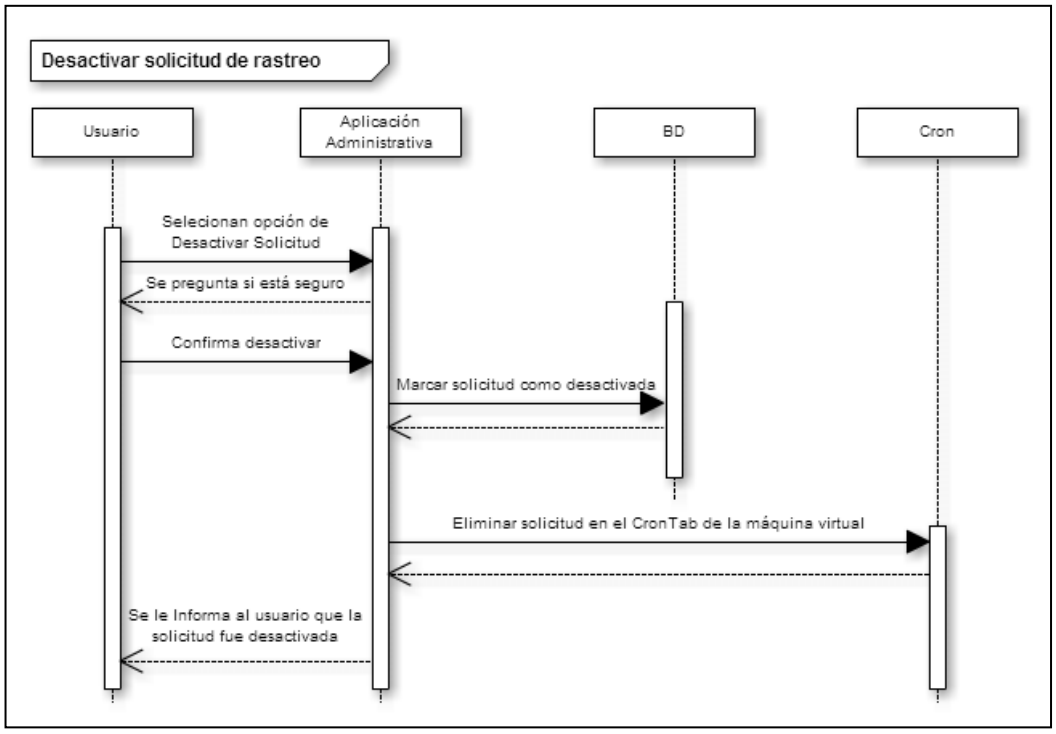


Figura 4-24 Diagrama de secuencia Desactivar Solicitud de rastreo

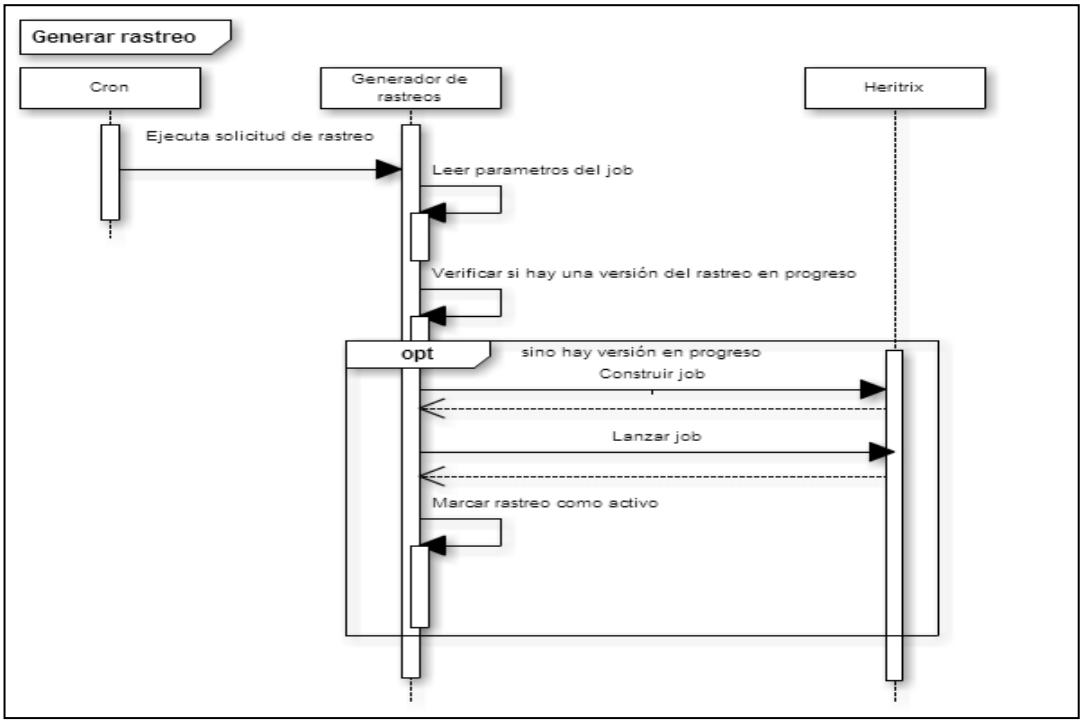


Figura 4-25Diagrama de secuencia Generar Rastreo

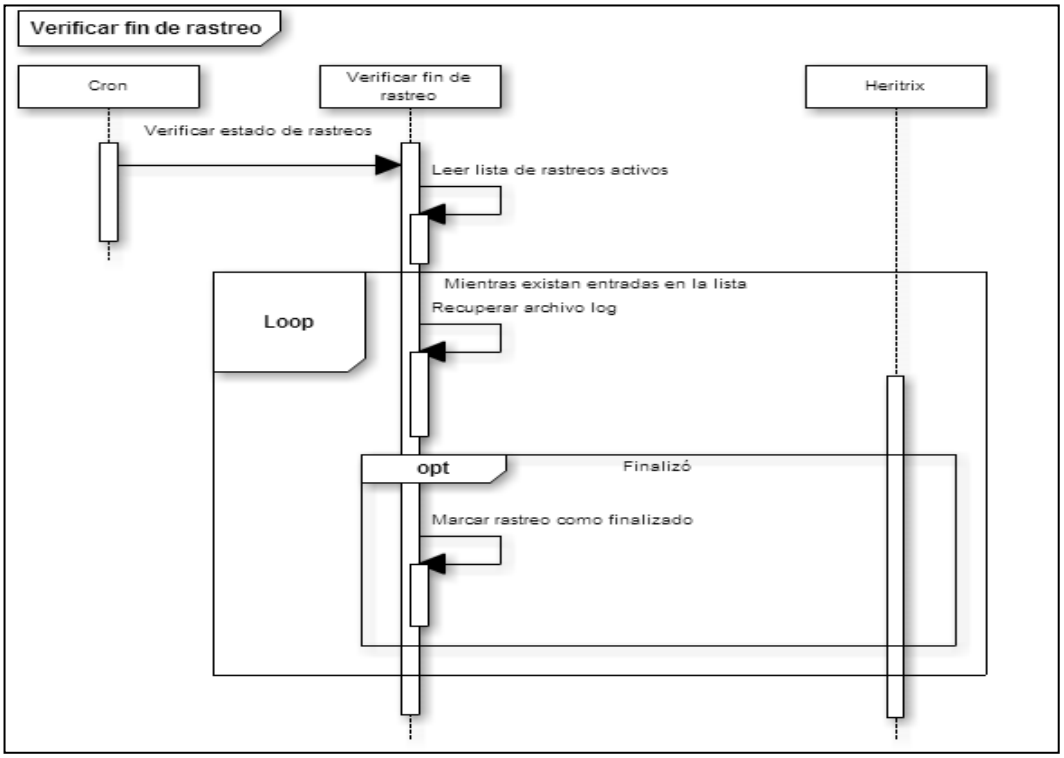


Figura 4-26 Diagrama de secuencia Verificar Fin de Rastreo

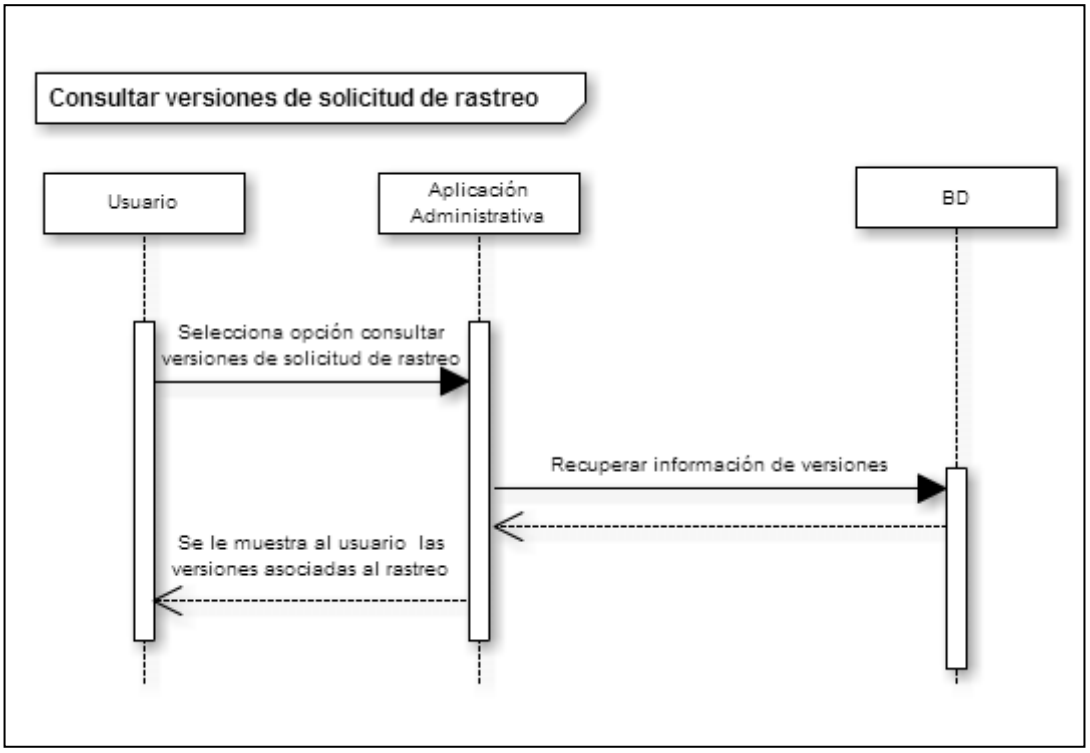


Figura 4-27 Diagrama de secuencia Consultar Versiones de Solicitud de Rastreo

4.4.2.6. Modelo de datos

En la siguiente gráfica (Figura 4-3), observamos el diseño general de la base de datos, la zona pre punteada en rojo indica las tablas las cuales el componente de predicción estará accediendo y ejecutando consultas SQL.

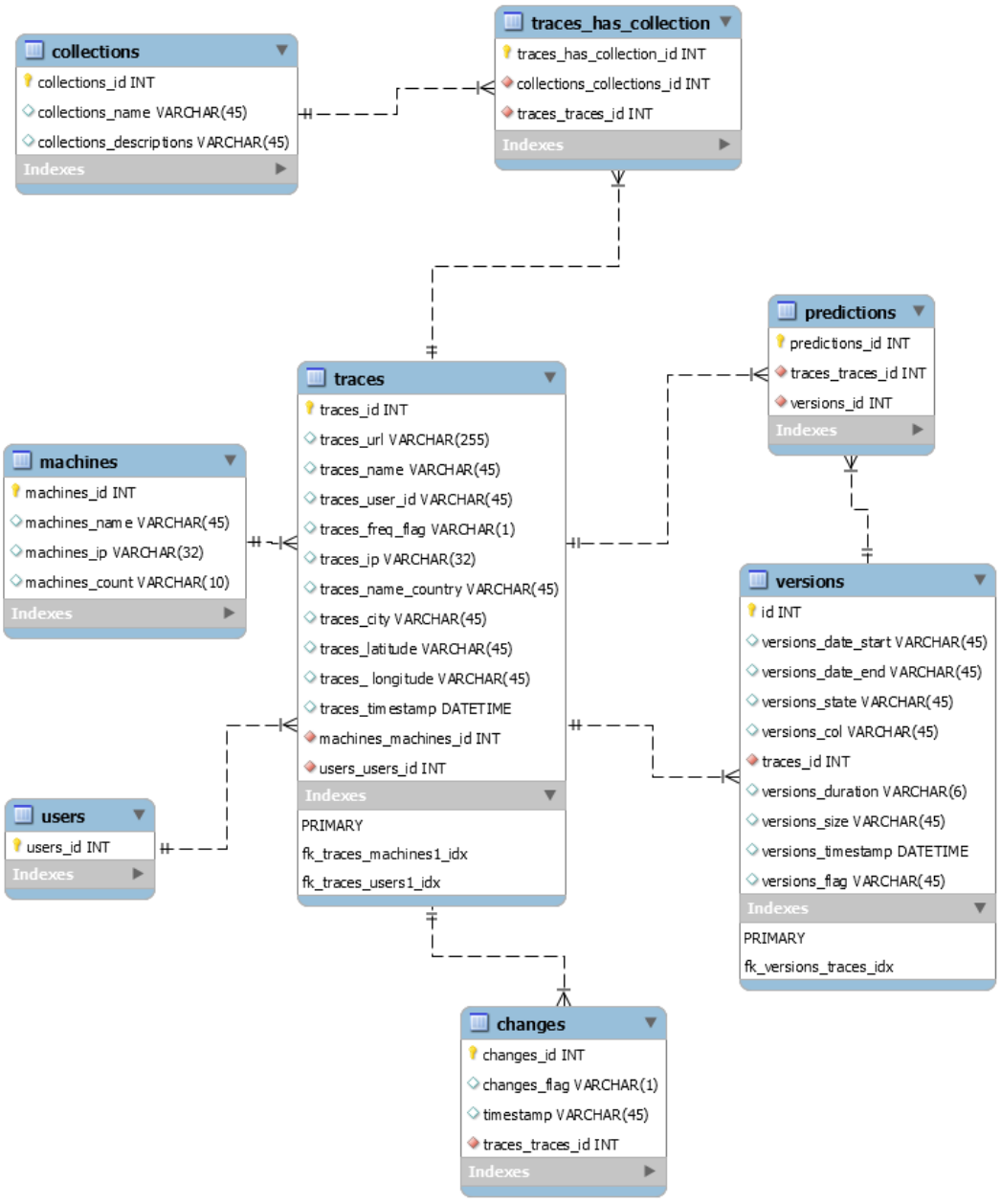


Figura 4-20 Diseño de la base de datos

4.4.3 Fase de Construcción

En este punto se explica cómo se llevó a cabo la implementación de los requerimientos del prototipo de preservación web, adaptando el desarrollo a una arquitectura basada en componentes.

4.4.3.1. Implementación de componentes

Para el desarrollo del prototipo de preservación web se utilizó software existente tales como el rastreador Heritrix, la librería de software libre WARCTools, la base de datos MySQL y el programador de tareas automáticas cron del sistema operativo Debian.

Adicionalmente se implementaron varios componentes para cubrir todos los requerimientos que el prototipo debe satisfacer, a continuación se explicará cómo fue llevada a cabo la implementación de cada uno de ellos.

4.4.3.2. Aplicación administrativa

Se desarrolló una aplicación administrativa para gestionar la creación y configuración de rastreos, consultar su estatus y una vez finalizados visualizar algunas de sus métricas.

Esta aplicación fue implementada utilizando el lenguaje Ruby y su framework Rails, para la persistencia de datos se utilizó la base de datos MySQL.

La aplicación cuenta con autenticación y registro de usuarios, para implementar esta funcionalidad se utilizó la gema devise (para información de instalación y configuración revisar la sección A-2 del anexo)

Las opciones para gestionar los rastreos se describen a continuación:

4.4.3.3. Composición de los Algoritmos.

Dada la particularidad de cambios que sufren los sitios se plantean heurísticas particulares para un dominio específico del problema. La heurística se elabora con la composición de uno o varios tipos y un volumen. Cada composición puede o no resolver un escenario particular de la problemática (Ver Figura 4-30).

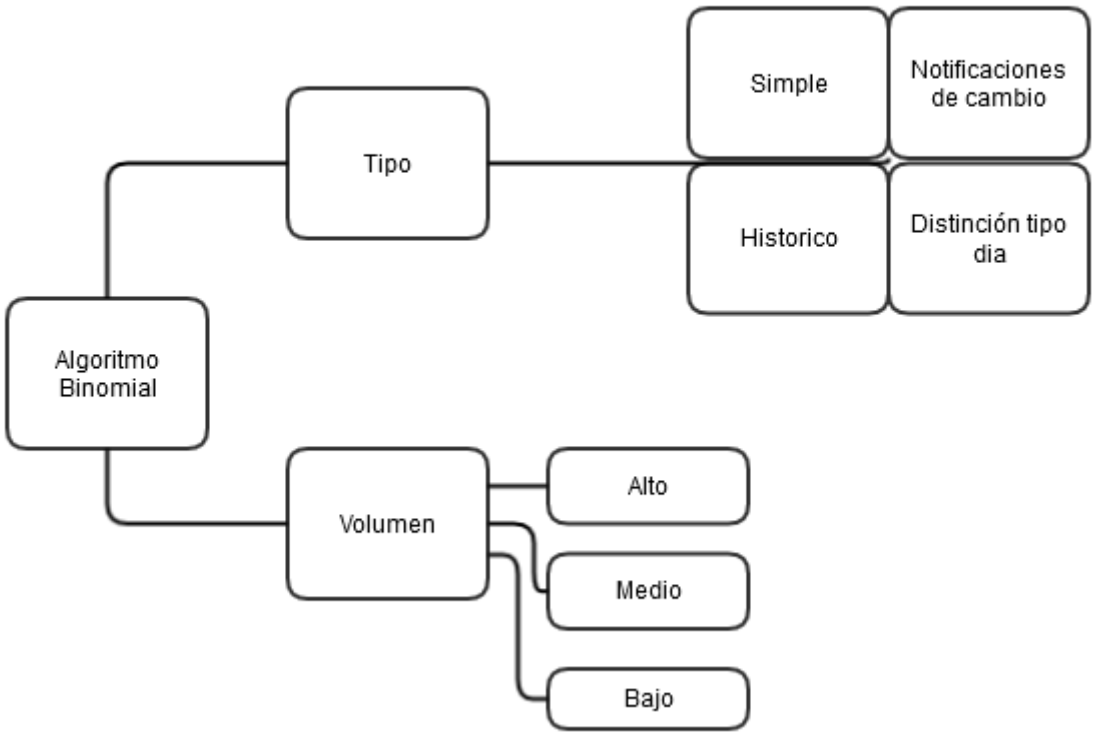


Figura 4-21 Composicion interna de un Algoritmo de predicción.

Tipo: especifica el alcance del algoritmo de predicción. El uso de una característica especifica busca una solución a una problemática en un contexto con características bien particulares, de igual forma ocurre cuando se realiza una adición de componentes. Los tipos de algoritmos están basados en el Algoritmo de predicción simple y poseen un agregado adicional:

- Simple: Se evalúan los cambios en función del presente
- Notificación de cambio: El algoritmo ejecuta las instrucciones pautadas, a menos que sea notificado de un comportamiento inusual lo que obliga a restaurar las variables de conteo y frecuencia a su mínimo valor.
- Histórico: los próximos cambios por venir, estarán sujetos inmediatamente a los eventos pasados (El comportamiento anterior está ligado con el comportamiento futuro).

- Distinción tipo día: existe una diferenciación en cuando a probabilidad de ocurrencia de cambio si se está en presencia de un día laboral o no.

Volumen: el volumen determina la magnitud de cambio que ocurren en una unidad determinada de tiempo, de esta manera se puede establecer el valor mínimo a considerar en el algoritmo.

4.4.3.4. Método de predicción propuesto para la solución del problema

Para la solución del problema planteado en el presente trabajo, se optó por usar la distribución Bernoulli para modelar la situación de cada página cada día como sigue:

$X = 1$ si la página cambia, en caso contrario $X = 0$.

Para la solución del problema 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 =Mín(T,24).

Para ilustrar mejor el funcionamiento del Algoritmo tomaremos como ejemplo el portal web de la Universidad de Carabobo (UC). En la aplicación podremos observar la diferencia entre los cambios que ocurrieron en la página y el curso del algoritmo. En la Figura 4-5 podemos observar el volumen de cambios por años de la UC.

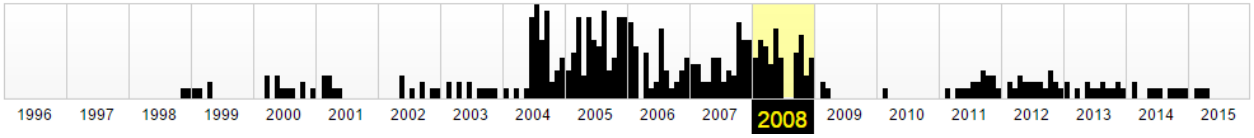


Figura 4-22 Volumen de cambios por años de la Universidad de Carabobo http://www.uc.edu.ve/ [23]

Algoritmo simple:

El diagrama del algoritmo se puede observar gráficamente en la Figura 4-6. Este método es excelente en situaciones impredecibles y tiene un alto nivel de sensibilidad al cambio para aproximar y ajustar las variaciones del predictor. Además, se puede aprovechar la sencillez del

algoritmo propuesto (Predicción de cambios base, Figura 4-6|), y construir modelos que presenten lógica y procesos adaptados a uno o más escenarios.

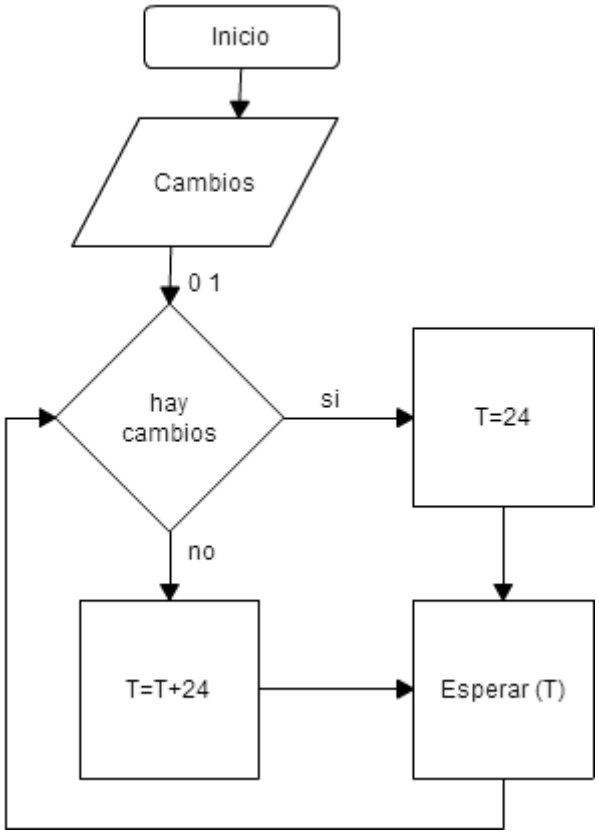


Figura 4-23 Algoritmo de predicción simple.

El algoritmo presentado en la Figura 4-6 representa la base fundamental en todos los prototipos y modelos para la predicción de cambios. Por lo tanto, todos los modelos planteados en la siguiente sección, parten del Algoritmo de predicción de cambios base. El desarrollo de los modelos presentados a continuación responde a la necesidad de obtener datos más precisos y acertar en mayor medida las predicciones y hacer llamados certeros para iniciar los rastreos.

A continuación se puede observar en la Figura 4-7 el funcionamiento del algoritmo simple en un lapso real de observación.

Leyenda:

✖ Refleja que un cambio se perdió.

El algoritmo ejecuto la orden para rastrear el sitio.(rastreos)

El algoritmo no recibió notificación para ejecutar el rastreo del sitio.(consultas)

Datos Generales:

Algoritmo implementado: Si / Rastreos: 34 / Total Cambios Perdidos: 8 / Consultas: 55 / MB ocupados: 226MB

Frecuencia fija: Frecuencia-diaria: 2 / MB ocupados: 912MB

Figura 4-24 Algoritmo de predicción de cambios base aplicado al sitio de la Universidad de Carabobo http://www.uc.edu.ve/ [23]

Cuando ocurren cambios de gran magnitud, el Algoritmo simple en el peor de los caso puede llegar a perder $\frac{T}{24}$ - 1 cambios. Los sitios rastreados que tienen servicio de notificación activo pueden utilizar el algoritmo simple con notificación para la predicción explicado en la siguiente sección.

Algoritmo simple con notificación:

El diagrama del algoritmo se puede observar gráficamente en la Figura 4-34. Este método es excelente en situaciones impredecibles y tiene un alto nivel de sensibilidad al cambio para aproximar y ajustar las variaciones del predictor. Además, se puede aprovechar la sencillez del algoritmo propuesto (Algoritmo de predicción de cambios simple, Figura 4-32|), y construir modelos que presenten lógica y procesos adaptados a uno o más escenarios.

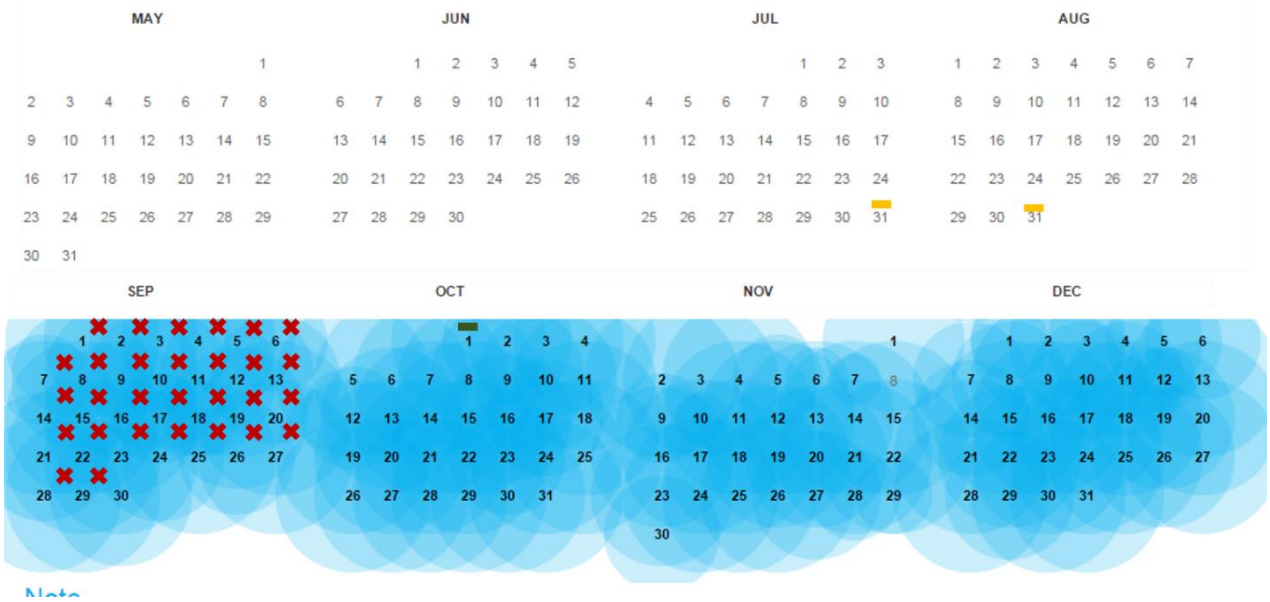


Figura 4-25 Caso borde cambio de gran magnitud.

Dado a que la notificación de cambios consiste en una sobrecarga de memoria adicional deben ocurrir 2 condiciones fundamentales para que pueda entrar en funcionamiento.

- $T > \frac{\text{Máximo(T)}}{2}$, Esto es, la página posee un intervalo de tiempo de espera
- La página web rastreada debe poder registrarse en una herramienta tercerizada que notifique si ocurrió o no un cambio reciente en un intervalo de tiempo de observación variable.

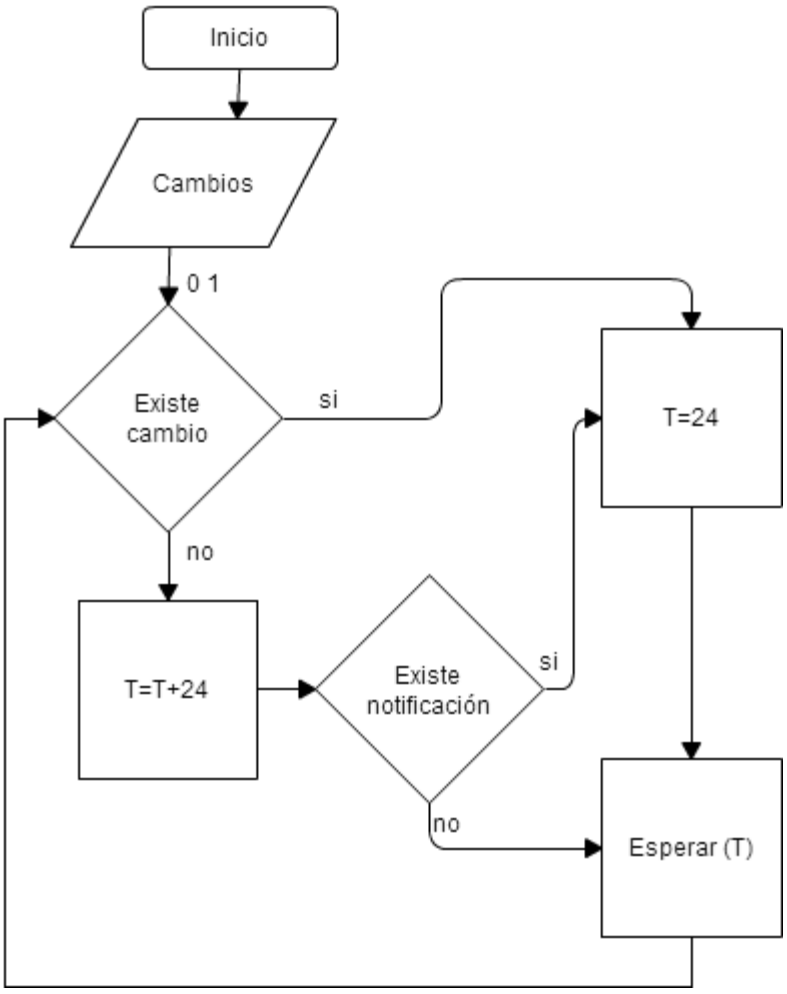


Figura 4-26 Algoritmo de predicción simple con notificación de cambios.

Leyenda:

✖ Refleja que un cambio se perdio.

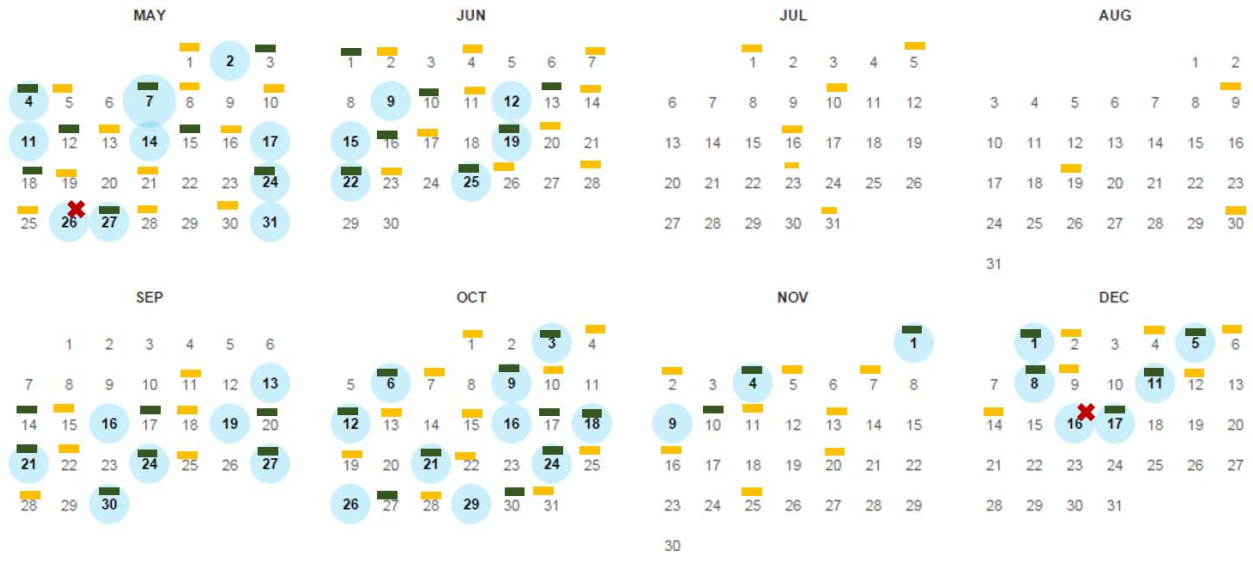
El algoritmo ejecuto la orden para rastrear el sitio.

El algoritmo no recibio notificacion para ejecutar el rastreo del sitio.

Datos Generales:

Algoritmo implementado: Si / Rastreos: 40 / Total Cambios Perdidos: 2 / Consultas: 65 / MB ocupados: 226MB

Frecuencia fija: Frecuencia-diaria: 2 / MB ocupados: 912MB



Antes de entrar en detalle con la explicación formal de los siguientes algoritmos, debemos entender una serie de funciones que permiten interpretar el significado de los datos arrojados por los procesos incluidos en cada algoritmo:

Y: Esta función retorna como resultado la sumatoria de todos los cambios que sufrió una página en un intervalo de N días de consulta (Figura 4-7).

$$Y = \sum_{k=0}^{n} \mathrm{T(k)}$$

Figura 4-27 Sumatoria de los cambios que ha tenido la página.

Z (cod-pagina/ día/ año**):** Esta función recibe como parámetros los siguientes valores: día (intervalo de 1 a 365 días,"1-365"); año (expresado en cuatro cifras, "YYYY") y un número de

identificación específico denominado como “cod-pagina”; que permite identificar la página web que está siendo consultada. Al especificar los tres parámetros, la función retorna el histórico de cambios que fue almacenado en ese día y año específicos. Si en esa fecha la página web cambió, retorna 1; en caso contrario, retorna 0.

Ejemplos:

Z(“#301”,1,2014)=1

Z(“#301”,2,2014)=1

Z((“#301”,365,2014)=0

4.4.3.5. Aplicación de Consulta

Se desarrolló una aplicación para permitir al usuario consultar la información relevante sobre los rastreos que fueron cosechados, las métricas de cada rastreo y graficas referentes a los procesos de almacenamiento, cambios que sufrió la página y los rastreos disparados por el predictor de cambios.

Esta aplicación fue implementada utilizando el lenguaje Phyton y su framework Django, para la persistencia de datos referidos a las métricas sobre el contenedor se utilizó la base de datos MySQL.

Lo primero que se debe hacer al momento de consultar los rastreos disponibles es acceder al sitio de interés, mostrando la información más relevante, además de las métricas de adquisición.

La información que se guarda en primer momento es toda la información que fue almacenada en la base de datos Mysql, esta información abarca:

- Nombre del rastreo.
- URL.
- Frequencia.
- Grupo.
- Máquina en la cual se almacenó el WARC.
- Fecha de Inicio del proceso de rastreo.
- Fecha en la que finalizó el rastreo.
- Nombre del WARC

4.4.4. Fase de transición

En esta sección se explicará cómo fue realizada la transición desde el entorno de desarrollo hacia el entorno de producción, se explicará qué pruebas se realizaron para validar el prototipo y se mostrará información de métricas pertenecientes a los rastreos realizados en las pruebas y los tiempos de indexación de los contenedores.

Además se mostrará información relevante asociada a los contenedores WARC y a los documentos contenidos dentro del mismo.

4.4.4.1. Puesta en producción del prototipo de preservación web

En este punto se hace la transición desde el entorno de desarrollo hacia el entorno de producción, a continuación se listan las máquinas virtuales y se menciona que componentes del sistema fueron alojados en cada una de ellas:

- **Heritrix 1**: En esta máquina se encuentra la aplicación administrativa para configuración de rastreos y Heritrix, generador de rastreos, verificador de rastreos finalizados y herramientas WARCTools.
- **Heritrix 2**: Heritrix, generador de rastreos, verificador de rastreos finalizados y herramientas WARCTools.
- **Heritrix 3**: Heritrix, generador de rastreos, verificador de rastreos finalizados y herramientas WARCTools.
- **Heritrix 4:** Heritrix, generador de rastreos, verificador de rastreos finalizados y herramientas WARCTools.
- **Imagen Solr:** En esta máquina se encuentra la aplicación de consulta, el servidor de base de datos MySQL y el software correspondiente a la plataforma de búsqueda Solr.

4.4.4.2. Pruebas Funcionales

En esta sección se realizarán una serie de pruebas para garantizar que el prototipo funciona de forma correcta y cumple con los requisitos funcionales.

Lista de sitios Web a rastrear:

Para hacer pruebas en el prototipo se decidió elaborar una lista de sitios web a rastrear, estos sitios pertenecen a universidades venezolanas así como a instituciones culturales, comerciales y/o investigativas, la lista es la siguiente (Ver figura 4-38):

titulo	url
Universidad Nacional Experimental del Tachira	www.unet.edu.ve/
IESA	www.iesa.edu.ve/
Universidad Catolica Andres Bello	www.ucab.edu.ve/
Universidad Centroccidental Lisandro Alvarado	www.ucla.edu.ve/
Universidad de Oriente	www.udo.edu.ve/
Universidad Nacional Abierta	www.una.edu.ve/
Universidad Nueva Esparta	www.une.edu.ve/
UNEFA	www.unefa.edu.ve/
Universidad Experimental de Guayana	www.uneg.edu.ve
Universidad Nacional Experimental Simon Rodríguez	www.unesr.edu.ve
UNEXPO	www.unexpo.edu.ve/
UNIMET	www.unimet.edu.ve
Universidad de Carabobo	www.uc.edu.ve
Universidad de los Andes	www.ula.ve/
USB	www.usb.ve/
Universidad Pedagogica Experimental Libertador	www.upel.edu.ve/
ISUM	www.isum.com.ve/
Instituto Universitario Tecnologico Americo Vespuc...	www.gav.edu.ve/
Instituto Iberoamericano de Altos Estudios Judicia...	iiaej.tsj.gob.ve/
Instituto Universitario Nuevas Profesiones	http://iunp.edu.ve/
Parque Tecnologico de Merida	www.cptm.ula.ve
Ico Group	www.icogroup.com
Apensar	www.todosapensar.com/
7aniversario	www.7aniversario.com.ve
twistos	http://www.twistos.com.ve/
Consejo municipal chacao	http://www.concejochacao.gob.ve/
Alcaldia Valencia	www.alcaldiadevalencia.gob.ve/
Clinica la floresta	http://www.clinicalafloresta.com/
Ciudades digitales 2013	http://cidiweb.net/evento/cd2013/inicio.php
Margarita Virtual	www.margaritavirtual.com.ve
Carros el tuy	www.autocarroseltuy.com.ve
Uneweb	www.uneweb.com.ve
Noticias 24	http://www.noticias24.com/index.html
El Nacional	www.el-nacional.com
El universal	www.eluniversal.com
El carabobeño	www.el-carabobeno.com/
Tal Cual	www.talcualdigital.com
Noticiero Digital	www.noticierodigital.com/
Noticiero Venevision	www.noticierovenevision.net/
Globovision	http://globovision.com/
Informe 21	http://informe21.com/
Mercadolibre	http://www.mercadolibre.com.ve/

Figura 4-28 Lista de páginas web a cosechar.

Como ejemplo se muestran los datos recopilados la página del IESA En la FIgura 4-35 podemos observar los cambios que se presentan diariamente durante un periodo de tres meses.

Figura 4-29 Observación sobre los cambios ocurridos en el sitio web www.iesa.com.ve.

Resultados de la Busqueda

Rastreos Encontrados:

Nombre	URL	Frequencia	Grupo	Editar	Desactivar	Observaciones
1	Universidad Nacional Expe	0.56846473029046	Medio	Editar	Desactivar	
2	IESA	0.5103734439834	Medio	Editar	Desactivar	
3	Universidad Cat�lica And	0.4896265560166	Medio	Editar	Desactivar	
4	Universidad Centroccident	0.49377593360996	Medio	Editar	Desactivar	
5	Universidad de Oriente	0.48547717842324	Medio	Editar	Desactivar	
6	Universidad Nacional Abie	0.51452282157676	Medio	Editar	Desactivar	
7	Universidad Nueva Espart	0.45643153526971	Medio	Editar	Desactivar	
8	UNEFA	0.50207468879668	Medio	Editar	Desactivar	
9	Universidad Experimental	0.48132780082988	Medio	Editar	Desactivar	
10	Universidad Nacional Expe	0.49792531120332	Medio	Editar	Desactivar	
11	UNEXPO	0.46473029045643	Medio	Editar	Desactivar	
12	UNIMET	0.45228215767635	Medio	Editar	Desactivar	
13	Universidad de Carabobo	0.50207468879668	Medio	Editar	Desactivar	
14	Universidad de los Andes	0.44813278008299	Medio	Editar	Desactivar	
15	USB	0.51867219917012	Medio	Editar	Desactivar	
16	Universidad Pedag�gica I	0.50207468879668	Medio	Editar	Desactivar	
17	ISUM	0.55601659751037	Medio	Editar	Desactivar	
18	Instituto Universitario Tecn	0.5103734439834	Medio	Editar	Desactivar	
19	Instituto Iberoamericano d	0.52282157676349	Medio	Editar	Desactivar	
20	Instituto Universitario Nuev	0.51452282157676	Medio	Editar	Desactivar	
21	Parque Tecnologico de Me	0	Bajo	Editar	Desactivar	
22	MercadoLibre	1	Alto	Editar	Desactivar	

Archivo Web de la Universidad Central de Venezuela.

Figura 4-30

Los resultados obtenidos abarcan todos los días de los meses comprendidos entre agosto y noviembre. Para el método que proponemos en la versión simple, no se diferencian los días festivos, fines de semana y días laborales.

En la Tabla 4-3 se muestran los datos recopilados durante el mes de diciembre del año 2013. Para el procesamiento y obtención de los mimos se utilizaron tres grupos de prueba y se midieron los cambios que ocurrieron cada día en cada una de las páginas de las organizaciones señaladas en cada grupo.

Los cambios son medidos por el detector de cambios, mandando una notificación sobre un sitio web en particular, indicando que hubo cambios recientes. Esta información recopilada será utilizada para calibrar el módulo de predicción de cambios.

4.4.4.3. Distribución de grupos

La agrupación de sitios se puede realizar de diversas formas. Listamos 4 diferentes agrupaciones según su categoría de clasificación:

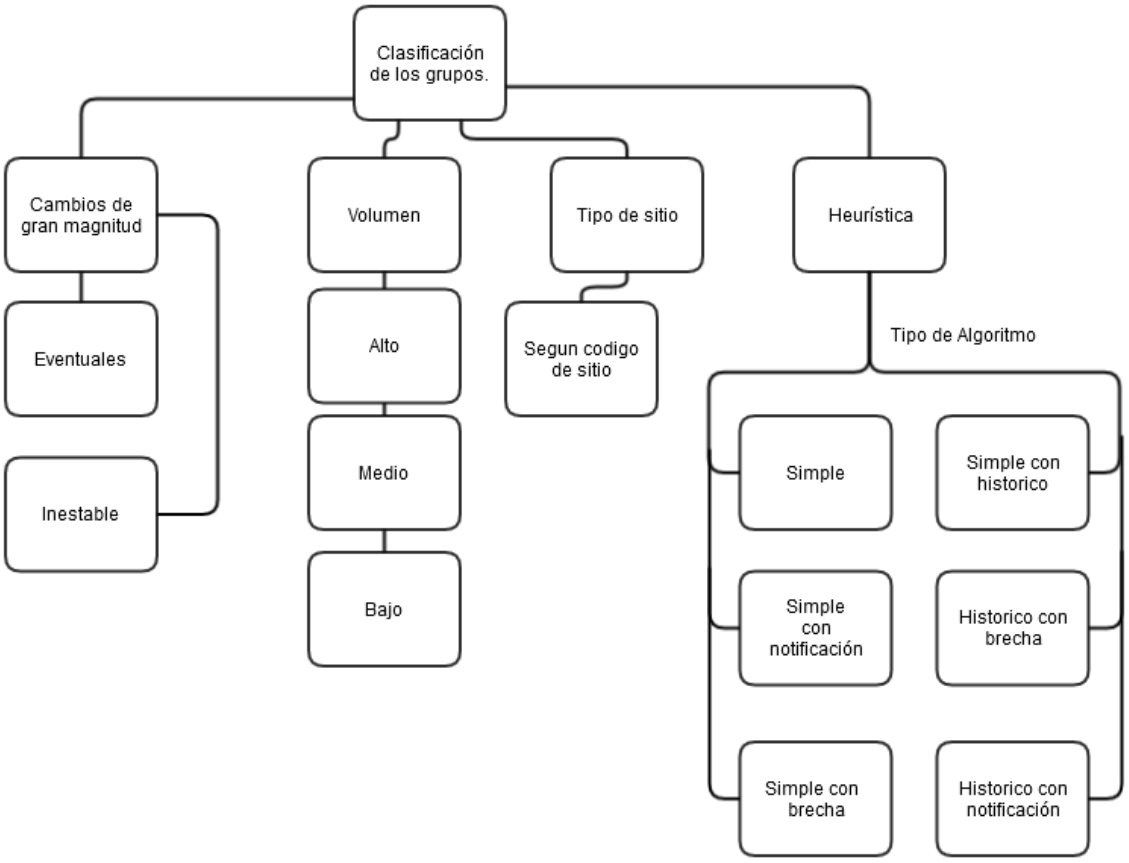


Figura 4-31 Agrupación de rastreos por grupos.

El cálculo de la frecuencia de cambio viene determinado por la siguiente expresión:

$$P(x) = \frac{1}{T} \sum_{i=o}^{\lim(\alpha)} (a_i)$$

Expresión	Descripción
T	Representa el intervalo de observación medido en días.
lim(α)	Representa la cantidad de elementos máximos que contiene α
α	Representa el vector de cambios y contiene almacenado de manera binaria los cambios que hubo o no en el transcurso de una observación.

Para efectos de la aplicación, se organizan los grupos por frecuencia de cambio.

Grupo 1 Alto: P(X) > 0.7. Esta probabilidad se asocia con un alto grado de cambio, los cambios son registrados en la página la mayor parte del tiempo.

Grupo 2 Medio: 0.4 < P(X) < 0.7. El sitio web mantiene una tasa de cambio estable.

Grupo 3 Bajo: P(X) < 0.4. El sitio casi no presenta cambios en el transcurso de la observación, e incluso, algunos sitios pueden llegar a no presentar cambios.

Para que un sitio web, pueda recibir predicciones del algoritmo simple debe cumplir con los siguientes requisitos:

- Dado un intervalo de observación, el número de cambios perdidos no debe sobrepasar el 50% de cambios que realmente ocurrieron en el intervalo de observación. Esto significa que más de la mitad de los cambios no se pierdan.
- Dado un comportamiento de cambio generado por una página web o un grupo de páginas web, el costo de almacenamiento generado por el sitio debe ser siempre igual o menor al de una proyección mínima establecida como frecuencia fija.
- El Algoritmo a ser implementado, debe ocupar menos del 5% de uso de rendimiento disponible por el servidor, esto es, mantener una línea de sencillez y optimización en el desarrollo de los algoritmos para evitar una sobrecarga en el sistema. (incluir las gráficas del uso del AW de los recursos)

4.4.4.4. Categorías de los sitios web

Los sitios web están orientados a uno o varios temas específicos, como un primer acercamiento, el prototipo de Archivo Web esta originalmente pensado para rastrear y preservar contenido orientado a las ciencias aplicadas, a continuación se puede observar una lista de temas con sus correspondientes códigos únicos para identificar dichos temas en el Archivo Web.

Sitios web de arte (codigo 300)

Sitios web de ciencia (codigo 301)

Sitios web de arquitectura (codigo 302)

Sitios web de deportes (codigo 303)

Sitios web de humor (codigo 304)

Sitios web de moda (codigo 305)

Sitios web de noticias (codigo 306)

Sitios web políticos (codigo 307)

Sitios web de compartición de archivos (codigo 308)

Sitios web de inteligencia (codigo 309)

Sitios web de juegos de mesa (codigo 310)

Sitios web de medio ambiente (codigo 311)

Sitios web de traducción (codigo 312)

Sitios web de viajes (codigo 313)

Sitios web de videojuegos (codigo 314)

4.4.4.5. Simulador de datos

El simulador de datos es un script realizado en Python. Este posee una consola de comandos que permiten interactuar directamente con los parámetros los cuales crearan un escenario donde será evaluado el comportamiento del sitio. Mediante la consola, podemos modelar situaciones basadas en históricos de cambios o situaciones irreales. Podemos escoger qué algoritmo implementar para el análisis posterior de la predicción. El simulador permita además incorporar nuevos algoritmos

de predicción. El simulador de datos permite generar data experimental, creando una aproximación sobre resultados de interés, como podría ser, el valor esperado de cambios que se perdieron en el sitio A, utilizando un Algoritmo de predicción tipo C en un lapso de observación de 30 días.

4.4.4.6. Análisis de tiempos de rastreo:

Una vez realizada las pruebas pertinentes, se muestra el resultado de rendimiento de las mismas. Cabe destacar que, para estos resultados, se cuenta con un universo de 62 contenedores de distintos tamaños. En cuanto al tiempo de indexación de los contenedores, en la Figura 4-16 se observa el tiempo en milisegundos que el proceso de indexación tardó en llevarse a cabo.

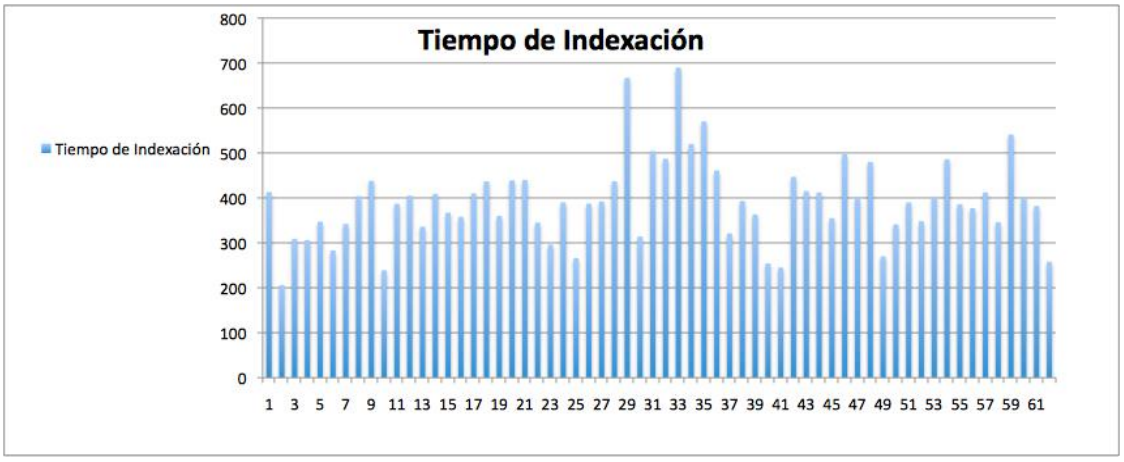


Figura 4-32 Tiempos de indexación por contenedor

En la Figura 4-11 se aprecia que existe un máximo aproximado de 690 milisegundos en lo que tardaría en indexar un solo contenedor y un mínimo de 200 milisegundos. Las 62 muestras tomadas nos indican que el proceso de indexación tardaría en promedio 391.67 milisegundos en indexar un contenedor. Cabe destacar que estos tiempos no dependen del tamaño del contenedor puesto que la indexación viene dada por la información que se recauda a través de los reportes que vienen en conjunto con el WARC.

A continuación se muestran los tiempos de rastreo medido en segundos de los contenedores agrupados por la URL que representan (medido en KB)

Tabla 4-1 Duración de rastreo por peso de contenedores asociado a la URL www.fmn.gob.ve

www.fmn.gob.ve	
Peso KB	**Duración de Rastreo (seg)**
61	2992,291
61	2153,088
61	2689,99
61	2708,996
61	2119,718
61	2091,531
61	2248,026
Promedio	**2429,091**
D. Estándar	**361,07**
Moda	**N/A**

Tabla 4-2 Duración de rastreo por peso de contenedores asociado a la URL www.ivic.gob.ve.

www.ivic.gob.ve	
Peso KB	**Duración de Rastreo (seg)**
5138022,4 (4.9 GB)	73207,814
5138022,4 (4.9 GB)	68740,518
5138022,4 (4.9 GB)	70719,55
5138022,4 (4.9 GB)	64510,72
Promedio	**69294,65**
D. Estándar	**3675,878**
Moda	**N/A**

Tabla 4-3 Duración de rastreo por peso de contenedores asociado a la URL www.luz.edu.ve

www.luz.edu.ve	
Peso KB	**Duración de Rastreo (seg)**
69632	64510,72
69632	27082,234
69632	26159,041
69632	26173,131
69632	26159,459
69632	26186,04
69632	26186,04
Promedio	**26303,131**
D. Estándar	**343,728**
Moda	**26186,04**

Tabla 4-4 Duración de rastreo por peso de contenedores asociado a la URLwww.mcti.gob.ve

www.mcti.gob.ve	
Peso KB	**Duración de Rastreo (seg)**
48128	22373,786
53248	23394,788
53248	23184,765
52224	23105,041
52224	23105,041
53248	23376,72
Promedio	**23090,0235**
D. Estándar	**373,497**
Moda	**23105,041**

Tabla 4-5 Duración de rastreo por peso de contenedores asociado a la URL www.pdvsalaestancia.com

www.pdvsalaestancia.com	
Peso KB	**Duración de Rastreo (seg)**
1258291,2	43891,351
1258291,2	42488,262
1258291,2	43879,398
1258291,2	41684,179
1258291,2	41982,399
1258291,2	41460,214
218112	10609,007
Promedio	**42564,30**
D. Estándar	**1079,681**
Moda	**N/A**

Tabla 4-6 Duración de rastreo por peso de contenedores asociado a la URL www.uc.edu.ve

www.uc.edu.ve	
Peso KB	**Duración de Rastreo (seg)**
677888	86400,916
1153433,6	86400,54
1153433,6	86402,014
Promedio	**86401,16**
D. Estándar	**0,77**
Moda	**N/A**

Tabla 4-7 Duración de rastreo por peso de contenedores asociado a la URL www.ucv.ve

www.ucv.ve	
Peso KB	**Duración de Rastreo (seg)**
4404019,2	86400,522
4404019,2	86402,092
4613734,4	86401,11
Promedio	**86401,24**
D. Estándar	**0,79**
Moda	**N/A**

Tabla 4-8 Duración de rastreo por peso de contenedores asociado a la URL www.udo.edu.ve

www.udo.edu.ve	
Peso KB	**Duración de Rastreo (seg)**
217088	86400,522
217088	86402,092
217088	86401,11
207872	43107,415
207872	42535,22
217088	42836,258
64512	43686,353
Promedio	**37810,158**
D. Estándar	**12940,34**
Moda	**N/A**

Tabla 4-9 Duración de rastreo por peso de contenedores asociado a la URL www.ula.ve

www.ula.ve	
Peso KB	**Duración de Rastreo (seg)**
63488	49611,935
63488	26553,064
63488	26554,004
63488	26666,857
63488	47791,708
63488	28177,267
63488	28091,121
Promedio	**33349,42**
D. Estándar	**10523,58**
Moda	**N/A**

Tabla 4-10 Duración de rastreo por peso de contenedores asociado a la URL www.usb.ve

www.usb.ve	
Peso KB	**Duración de Rastreo (seg)**
254976	26118,014
251904	26119,683
253952	26121,247
253952	26117,831
254976	26117,893
254976	26117,952
254976	10220,159
Promedio	**23847,54**
D. Estándar	**6009,11**
Moda	**N/A**

En las tablas anteriormente expuestas (Tabla 4-6 a la 4-15), se aprecia que a medida que va aumentando el tamaño de los contenedores, va aumentando el tiempo en el que rastreador cosecha el sitio. Cabe destacar que, a pesar de no haber un cambio considerable en los tamaños de los contenedores por URL (medido en KB), el tiempo es bastante variante. Una de las razones por las cuales esto puede suceder, es que el rastreador debe esperar la respuesta de los servidores en donde se encuentran alojados los sitios para poder procesar cada documento y almacenarlo dentro del contenedor WARC, por lo tanto se depende de la conexión hacia internet que se dispone y por razones aisladas a este Trabajo Especial de Grado se conoce que la conexiones suelen ser bastante inestables por factores externos propios de la red de la región.

A continuación, en la Figura 4-42, se muestra un gráfico de dispersión que muestra el tiempo de duración de los contenedores a medida que va aumentado el tamaño de los mismos.

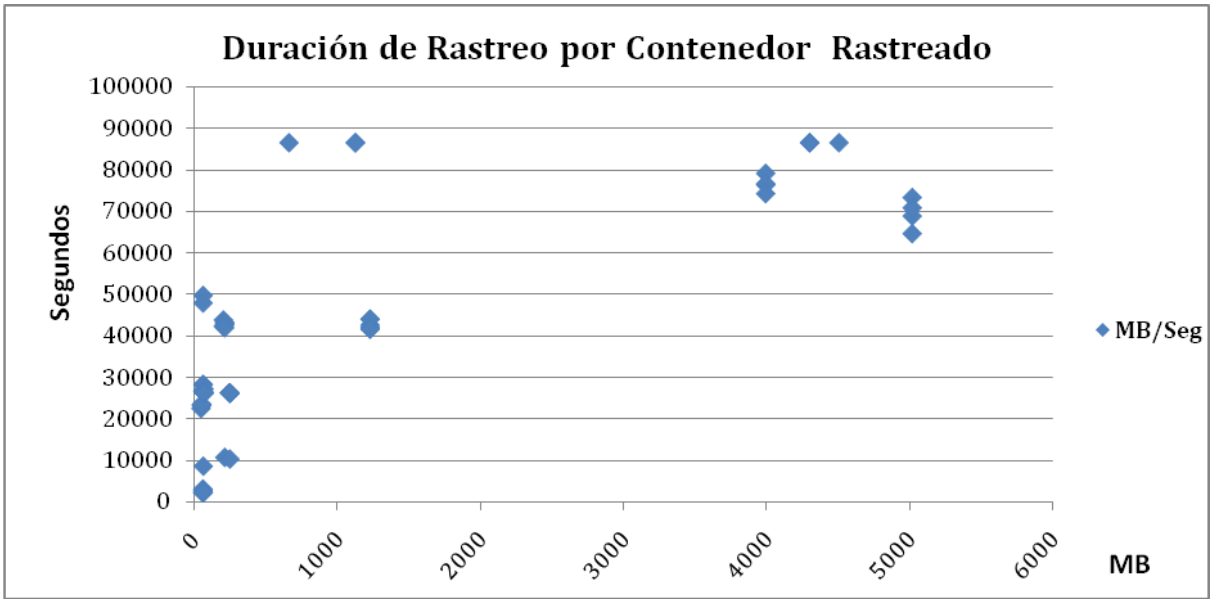


Figura 4-33 KB/seg dependiendo del peso del contenedor.

Efectivamente, en la Figura 4-43, se corrobora que a medida que los contenedores van subiendo de tamaño se observa un crecimiento del tiempo que toma rastrearlos, lo cual es un resultado esperado, más sin embargo, existen casos en los que esta regla no se cumple. Estas excepciones puede venir unidas tanto a problemas en la conexión de internet como a procesamiento, ya que cada máquina procesa los contenedores a velocidades distintas.

Con los casos expuestos se observa que a pesar de haber excepciones en el tiempo de rastreo de contenedores, el rastreador suele seguir un comportamiento esperado por el equipo de trabajo ya que se toma como premisa que entre mayor tamaño de contenedor mayor tiempo se tomará en ser rastreado. Por lo tanto, el tiempo necesario para realizar el rastreo de las paginas corresponden a 24h, en dicho intervalo la página debe ser rastreada para detectar si cambio o no.

4.4.4.7. Análisis e Interpretación de los resultados del algoritmo:

Desempeño del algoritmo:

Se evaluaron 42 páginas, las cuales están agrupadas en 3 grupos por frecuencia de cambio de las cuales se consideró analizar los datos relacionados con cada sitio web en particular, siendo estos los cambios perdidos, cambios ocurridos, el espacio generado (medido en mb), el espacio almacenado (medido en mb) los rastreos que se realizaron y la frecuencia de cambio que tiene el sitio web. En la figura 4-46 mostrada a continuación se puede observar la información general asociada al sitio web, se puede notar que se los cambios:

Información general:

Cambios Perdidos	Cambios Ocurridos	Espacio Generado(Mb)	Espacio Almacenado(Mb)	Rastreos	Frequencia
49	137	1231.4	779	157	0.56846473029046

Figura 4-34 Información general del sitio web Universidad Nacional Experimental del Táchira.

Con estos datos, podemos determinar si el sitio web puede aplicar con la integración del algoritmo simple (Esto es que cumpla con los requerimientos propuestos en la sección 4.4.4.8.

Distribución de grupos). En la tabla 17 se pueden verificar las condiciones, que se cumplen para todos los grupos y se toma como ejemplo el sitio web de Universidad Nacional Experimental del Táchira:

Tabla 4-11 Requisitos para integrar las predicciones con el Algoritmo simple.

Condición	Calculo	Resultado
Numero de cambios perdidos no puede ser mayor al 50% de cambios ocurridos	$\frac{Cambios\ perdidos}{Cambios\ Ocurridos} = \frac{49}{137}$	0.35%
Almacenamiento igual o menor al de una proyección mínima establecida	[Dinámica] = 779 [Estática] = 2096	El almacenamiento generado por el uso del algoritmo es menor en comparación al

como frecuencia fija.		uso de una frecuencia fija
El Algoritmo a ser implementado, debe ocupar menos del 5% de uso de rendimiento disponible por el servidor.	Memoria total: 9229 GbRam Memoria Usada: 4074 GbRam Aplicación: 124 GbRam/ 0,013%	El algoritmo utiliza solo el 0,013% del rendimiento del servidor.

Por otro lado se procede a realizar una comparación entre el algoritmo y el sitio web con una frecuencia de cambio fija, manera en que se venía trabajando anteriormente. Los cambios que realmente ocurrieron en el transcurso de la observación, el espacio ocupado en Mb que genera un sitio web en una fecha determinada y la acumulación a lo largo de la observación, además del número de llamas que hizo el algoritmo al rastreador así como también los cambios que se perdieron en el transcurso de la observación, se compararon evidenciándose lo siguiente:

Información general: Universidad Nacional Experimental del Tachira

Seleccion de frecuencia fija:		1	1	Universidad Nacional Exp	Enviar
Freq. Cambio	Cambios	Espacio DD	C. Perdidos	LLamadas Heritrix	Frequencia
Dinamica	137	779	49	157	0.56846473029046
Estatica	-	2096	1	240	0.99585062240664

Figura 4-35 Comparación de datos con frecuencia fija en 1.

Cuando la frecuencia de cambio se fija en 1 o en valores muy bajos el número de cambios perdidos es menor o nulo, sin embargo la duplicidad de versiones es mayor y evidentemente el espacio en disco duro está siendo ocupado por versiones repetidas. (Ver Figura 4-36)

En las figuras 4-37 y Figura 4-38 podemos observar que a medida que aumentados la frecuencia fija con intervalos mayores o iguales a 15 días, se empieza a perder una cantidad considerable de cambios importantes ganando espacio en disco duro a favor.

Información general: Universidad Nacional Experimental del Tachira

Seleccion de frecuencia fija:		15	1	Universidad Nacional Exp	Enviar
Freq. Cambio	Cambios	Espacio DD	C. Perdidos	LLamadas Heritrix	Frequencia
Dinamica	137	779	49	157	0.56846473029046
Estatica	-	125.1	127	16	0.06639004149377 6

Figura 4-46 Comparación de datos con frecuencia fija en 15

Información general: Universidad Nacional Experimental del Tachira

Seleccion de frecuencia fija:		30	1	Universidad Nacional Exp	Enviar
Freq. Cambio	Cambios	Espacio DD	C. Perdidos	LLamadas Heritrix	Frequencia
Dinamica	137	779	49	157	0.56846473029046
Estatica	-	67.5	130	8	0.033195020746888

Figura 4-47 Comparación de datos con frecuencia fija en 30

Información sobre los distintos grupos:

Para el grupo 1 se puede concluir que dado un intervalo de observación T hay una tendencia entre los cambios ocurridos y los rastreas realizados donde la frecuencia promedio de cambio es $\frac{Total\ Cambios\ Registrados}{Intervalo\ Total\ de\ observaciòn} = 0.49\%$, esto es que en un intervalo T de observación se espera que ocurran aproximadamente $\frac{1}{2}.(T)$ cambios, siendo este grupo integrado en su gran totalidad por sitios web asociados con sitios universitarios, culturales y de investigación(Ver figura 4-48) .

Grupo 1 información general:

Información de interes para el grupo 1:

Freq. Cambio	Intervalo	Total de sitios rastreados	Frecuencia Promedio	Cambios Perdidos	% Cambios Perdidos	Llamadas a Heritrix	Cambios Registrados	Consultas Innecesarias	%C. Innecesarias
Dinamica	5784 Horas	20	0.45%	902	0.18%	2981	2408	573	0.11%

Figura 4-48 Información general del grupo 1

Para el grupo 2 podemos denotar que dado un intervalo de observación T, los sitios web que presentaron una frecuencia de cambio de 0 no registraron en el transcurso de la observación cambios. (Ver figura 4-49).

Grupo **2** información general:

Información de interes para el grupo 1:

Freq. Cambio	Intervalo	Total de sitios rastreados	Frecuencia Promedio	Cambios Perdidos	% Cambios Perdidos	Llamadas a Heritrix	Cambios Registrados	Consultas Innecesarias	%C. Innecesarias
Dinamica	5784 Horas	10	0	0	0%	210	0	210	0.08%

Figura 4-39 Información general del grupo 2

Para el grupo 3 se puede observar que dado un intervalo de observación T, los sitios web que presentaron una frecuencia de cambio de 1 siempre estuvieron registrando cambios activamente y fluctuaciones o cambios de gran magnitud no fueron observados en el intervalo de observación. (Ver figura 4-49).

Grupo **3** información general:

Información de interes para el grupo 1:

Freq. Cambio	Intervalo	Total de sitios rastreados	Frecuencia Promedio	Cambios Perdidos	% Cambios Perdidos	Llamadas a Heritrix	Cambios Registrados	Consultas Innecesarias	%C. Innecesarias
Dinamica	5784 Horas	10	0	0	0%	2410	2410	0	0

Figura 4-40 Información general del grupo 3

4.4.4.8. Prototipo de las interfaces

En este apartado se muestran capturas de pantalla de un prototipo de la Interfaz de control de frecuencias (Figuras 4-50 a la 4-20).

Rastreos Encontrados:

Nombre	URL [Ver Datos]	Frequencia	Grupo	Comparar	Desactivar	Observaciones
1	Universidad Nacional Expe	0.56846473029046	2	Comparacion	Desactivar	
2	IESA	0.5103734439834	2	Comparacion	Desactivar	
3	Universidad Catolica Andr	0.4896265560166	2	Comparacion	Desactivar	
4	Universidad Centroccident	0.49377593360996	2	Comparacion	Desactivar	
5	Universidad de Oriente	0.48547717842324	2	Comparacion	Desactivar	
6	Universidad Nacional Abie	0.51452282157676	2	Comparacion	Desactivar	
7	Universidad Nueva Espart	0.45643153526971	2	Comparacion	Desactivar	
8	UNEFA	0.50207468879668	2	Comparacion	Desactivar	
9	Universidad Experimental	0.48132780082988	2	Comparacion	Desactivar	
10	Universidad Nacional Expe	0.49792531120332	2	Comparacion	Desactivar	
11	UNEXPO	0.46473029045643	2	Comparacion	Desactivar	
12	UNIMET	0.45228215767635	2	Comparacion	Desactivar	
13	Universidad de Carabobo	0.50207468879668	2	Comparacion	Desactivar	
14	Universidad de los Andes	0.44813278008299	2	Comparacion	Desactivar	
15	USB	0.51867219917012	2	Comparacion	Desactivar	
16	Universidad Pedagogica E	0.50207468879668	2	Comparacion	Desactivar	
17	ISUM	0.55601659751037	2	Comparacion	Desactivar	
18	Instituto Universitario Tecr	0.5103734439834	2	Comparacion	Desactivar	
19	Instituto Iberoamericano d	0.52282157676349	2	Comparacion	Desactivar	
20	Instituto Universitario Nue	0.51452282157676	2	Comparacion	Desactivar	
21	Parque Tecnologico de Me	0.004149377593361	3	Comparacion	Desactivar	
22	Ico Group	0	3	Comparacion	Desactivar	

Figura 4-41 Interfaz listar rastreos.

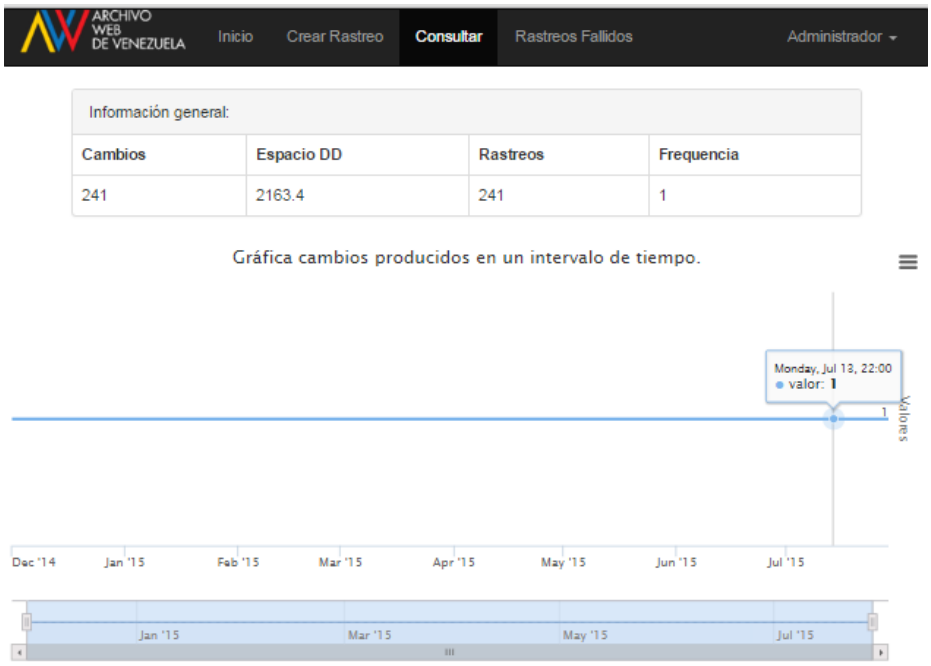


Figura 4-42 Interfaz graficas de metricas e información general.

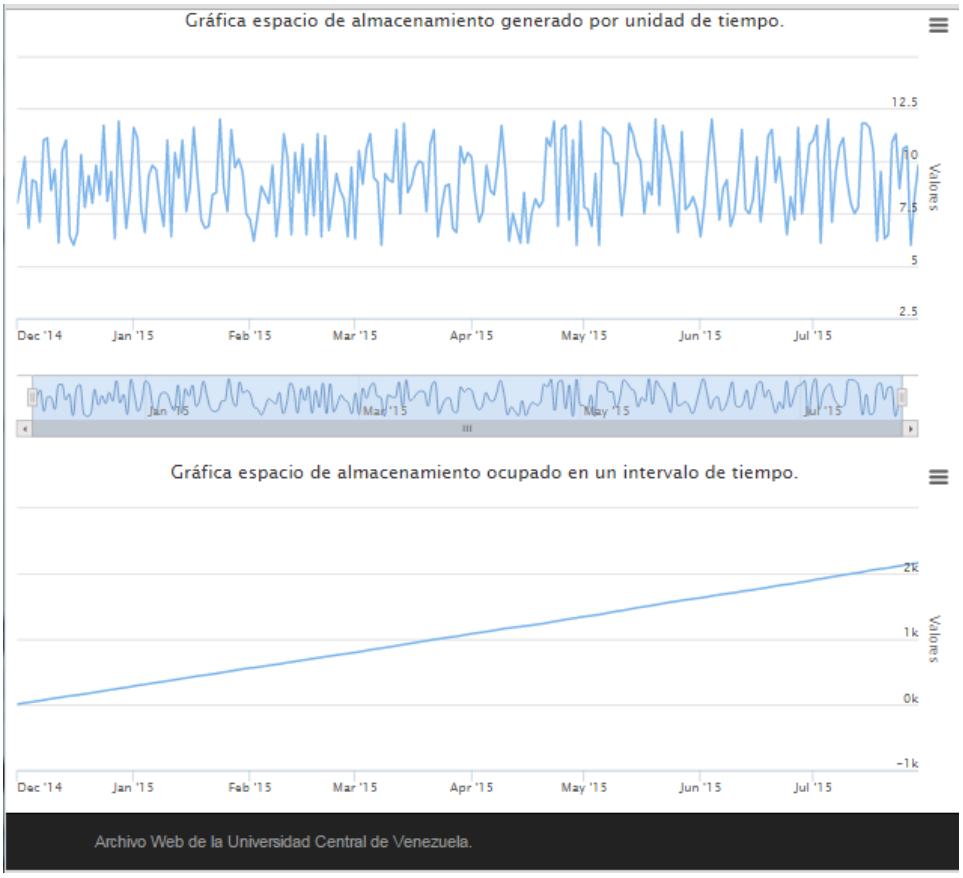


Figura 4-43 Interfaz gráfica de almacenamiento de un rastreo.

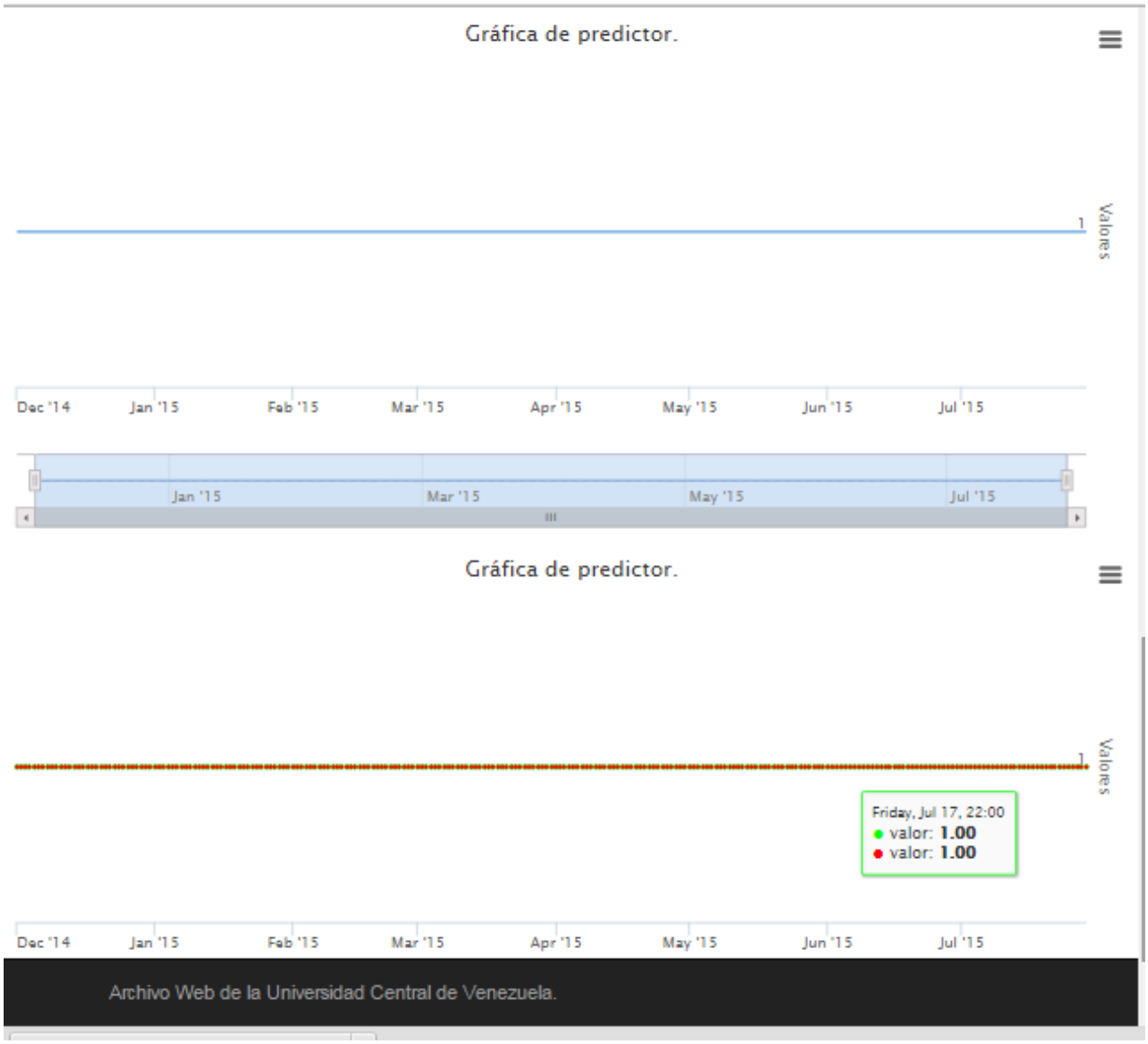


Figura 4-44 Interfaz gráfica de los rastreos iniciados por el predictor.

4.4.4.9. Cronograma tentativo de actividades para completar el Trabajo Especial de Grado.

Tabla 4-1218 Cronograma de actividades

Semana	**Actividades**
Todas las Semanas	Recopilación de datos asociados a las páginas web candidatas al prototipo web
Semana 1	Diseño de requerimientos

Semana 2	Discusión y toma de decisión sobre el modelo de predicción a utilizar
Semana 3-4	Familiarizarse y configurar el entorno de trabajo, realizar pruebas básica de integración con el prototipo web, evaluar herramientas de soporte
Semana 5-12	Programación del componente de predicción de cambio
Semana 13	Comunicar el componente con la base de datos
Semana 14-20	Pruebas unitarias, pruebas de interacción del componente con el prototipo web
*Durante el desarrollo de cada una de las actividades descritas en esta tabla se realizará el documento del Trabajo.	

CONCLUSIONES Y RECOMENDACIONES

Es evidente que para comenzar cualquier proyecto; independientemente de su alcance, éxito o fracaso, se debe conformar el mejor equipo para poder llevarlo a feliz término. Por lo tanto, las personas que comparten una dirección común y tienen sentido de pertenencia y responsabilidad grupal, serán los que permitan superar los retos que plantea una investigación con eficacia y eficiencia. El alcanzar las metas se convierte en una tarea más fácil y rápida, pues los integrantes de un equipo trabajan coordinadamente, asumen sus funciones y responsabilidades y se apoyan mutuamente.

Para el primer objetivo, se adaptó el método AUP, combinándolo con una arquitectura basada en componentes. El estricto cumplimiento del plan de trabajo y una adecuada distribución de funciones, permitieron un correcto desarrollo del componente de predicción, cuyos resultados amplían el espectro de análisis de la metodología inicial e incrementan la confiabilidad de los resultados obtenidos.

Para el segundo objetivo, se creó un algoritmo de predicción basado en probabilidades y estadística descriptiva. El algoritmo funcionó con un alto rendimiento; observándose como los principales logros alcanzados en su desarrollo: la sencillez del algoritmo, su fácil comprensión, escaso consumo de recursos y alta eficiencia.

Para el tercer objetivo, se elaboró el módulo administrativo. Este permite a los usuarios acceder y monitorear las frecuencias de cambio en los sitios web preservados con una alta eficacia. El módulo de administración permite realizar consultas de gran utilidad, además de que es capaz de generar una importante cantidad de datos pre-procesados. Esto permite a los usuarios del módulo obtener resultados y tomar decisiones certeras en un entorno gráfico, que facilita la lectura de los datos y su contraste con los datos arrojados por el simulador.

Para el cuarto objetivo, se utilizaron los históricos para calibrar el componente de predicción. Esta utilización no representó una mejora sustancial o notable en la etapa de simulación del

componente.

Para el quinto objetivo, se clasificaron los sitios de acuerdo con criterios específicos. Esto permitió tener una noción general del comportamiento de las páginas por categorías. Los datos recopilados en el transcurso de las observaciones permitieron concluir que una página web no necesariamente tiene un comportamiento definido en el transcurso de su observación, y que con el transcurso del tiempo puede variar significativamente su comportamiento.

En lo referente a las recomendaciones derivadas de la presente investigación, nos permitimos sugerir las siguientes:

Se recomienda la implementación formal de un detector de ráfagas, esto evitaría la perdida de cambios significativos en el caso particular de sitios que posean comportamiento inestable.

Como un proceso de análisis posterior al propuesto en esta investigación, se podría ejecutar el análisis estadístico del transcurso de tiempo transcurrido entre cada medición realizada en cada sitio web específico, y la influencia que tendría en la detección de cambios en el sitio.

Como un proceso de análisis posterior al propuesto en esta investigación, al tener una cantidad sustancial de información e histórico mayor o igual a un año, es de interés, comparar, analizar y estudiar si existe una relación entre el comportamiento que presente el sitio dependiendo del t de día en que sean evaluadas, siendo más específicos, fechas feriadas, fines de semanas e incluso días de la semana que en ese momento en particular presentaron un acontecimiento de magnitud desproporcionada(desastres naturales, confrontaciones entre naciones, entre otros…).

Se proponen el siguientes algoritmos para ser evaluado:

Simple con históricos:

Sean las Variable T y N, donde T representa un tiempo dado en horas, donde $24 \leq T \leq 144$ y N, donde N es un intervalo de observación representado en días ($1 \leq N \leq$ TopeN) y La variable aleatoria bernoulli X = 1 Si hay cambios y X = 0 Si no hay cambios en el N-esimo intervalo de tiempo T, obtenido con la función $Z_N(D, M, A)$, con TopeN mediciones consecutivas de un

mismo sitio web. Inicialmente P(X) depende directamente de las mediciones anteriores, donde se debe cumplir $0 \leq P(X) \leq 1$. Si en este intervalo de mediciones consecutivas no se detectan cambios en el sitio, entonces se asigna P(X)=0, dicho valor corresponde con un tiempo de espera T=144h (6 dias). En caso de detectar algún cambio en una medición de dicho sitio, debemos calcular P(X), por medio de la función (ver Figura 4-9):

$$P(x) \quad = \frac{1}{TopeN} \sum_{N=-TopeN}^{0} Z_N(D, M, A)$$

Figura 4-45 Sumatorio de todos los cambios en una observación.

El algoritmo ajusta el valor de T de acurdo al valor calculado de *P(X)*. El algoritmo está diseñado para calcular la probabilidad diaria de cambio enmarcada en un intervalo determinado por un valor. Busca establecer un comportamiento actual basado en el histórico de cambio producido exactamente TopeN días antes, el funcionamiento del algoritmo puede observarse gráficamente con mas detalle en la Figura 4-11. Presentamos el siguiente ejemplo mostrado en la Figura 4-10 (Medición tomada del portal web Universidad del Zulia http://www.uc.edu.ve/ durante el mes de marzo de 2014) , que muestra el volumen de cambios durante la observación del sitio con un intervalo de 12 años.

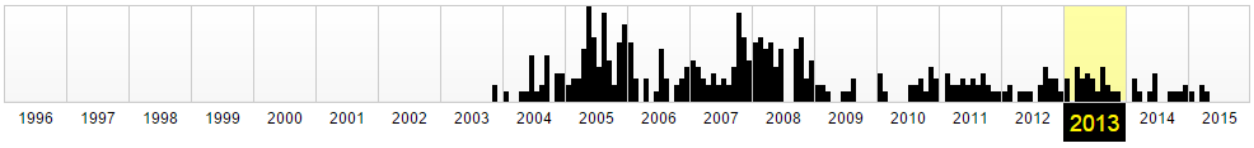


Figura 4-46 Volumen de cambios por años de la Universidad del Zulia http://www.luz.edu.ve/ [23]

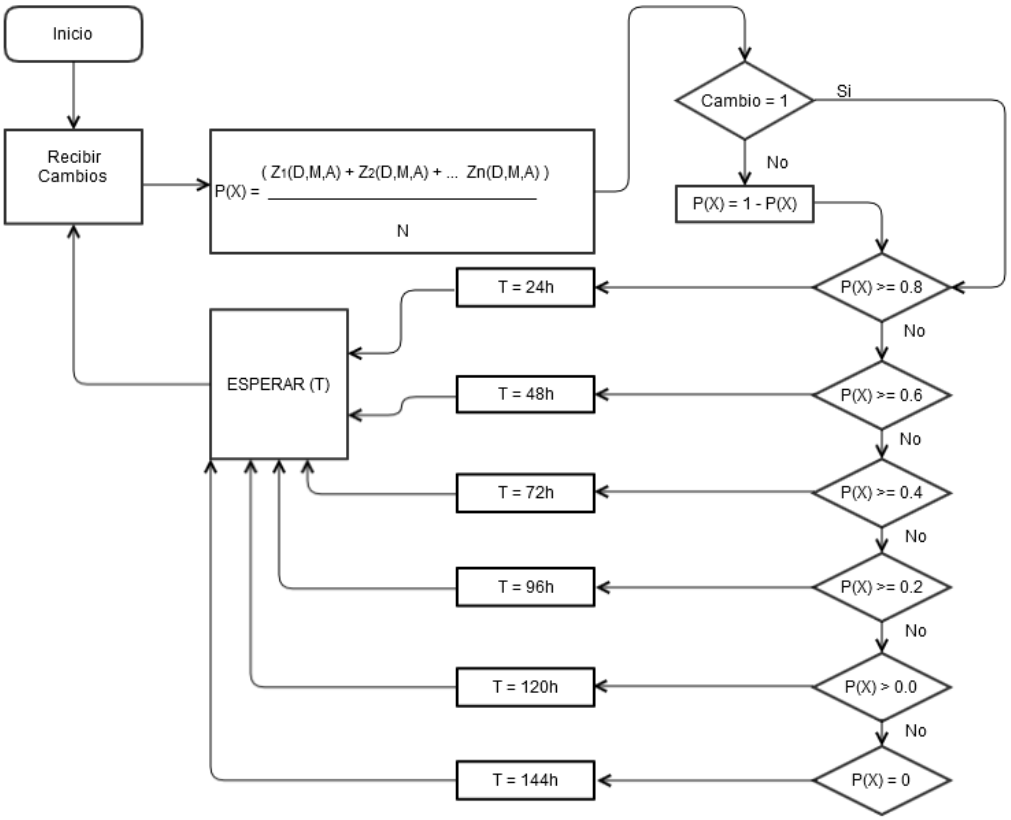


Figura 4-47 Algoritmo de predicción simple con históricos.

Se puede observar en la figura 4-39, un ejemplo del histórico de cambio de un sitio. En la figura 4-40, se observa el mismo sitio con un intervalo de observación diferente, apreciándose el funcionamiento del Algoritmo de predicción con históricos en un lapso real de observación.

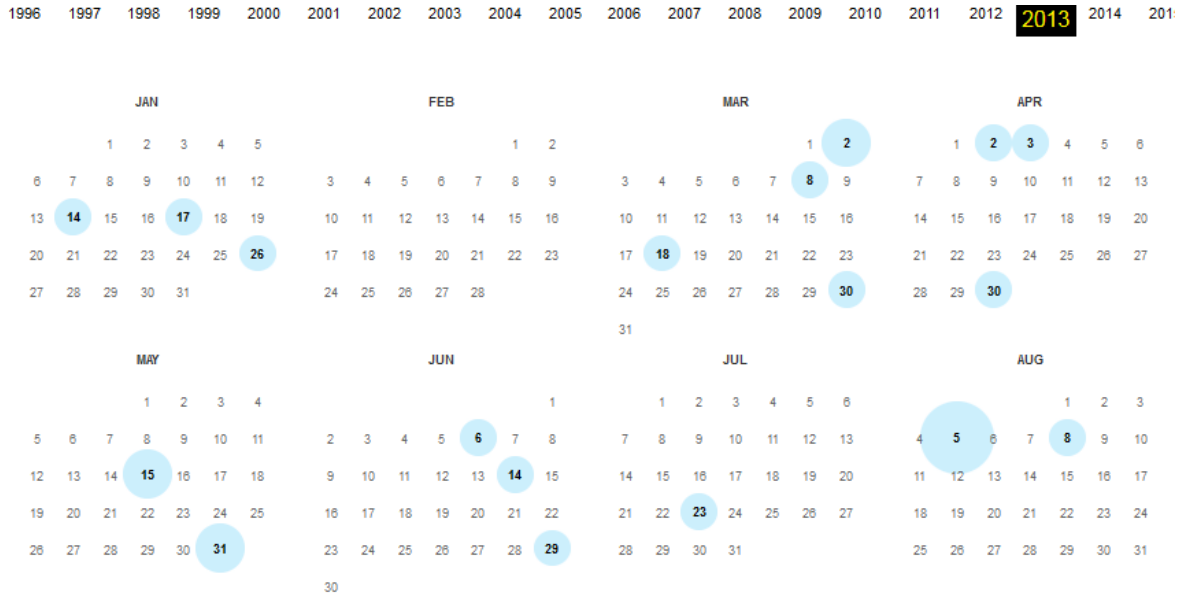


Figura 4-48 Histórico de cambio del año 2013 del sitio Universidad del Zulia http://www.luz.edu.ve/

Leyenda:

✖ Refleja que un cambio se ha perdido.

El algoritmo ejecuto la orden para rastrear el sitio. (Rastreos)

El algoritmo no recibió notificación para rastrear y almacenar el sitio. (Consultas)

Datos Generales:

Algoritmo implementado: Si / Rastreos: 7 / Total Cambios Perdidos: 0 / Consultas: 20/ MB ocupados: 151MB

Algoritmo implementado: No / Frecuencia-diaria: 5 / MB ocupados: 1041,6MB

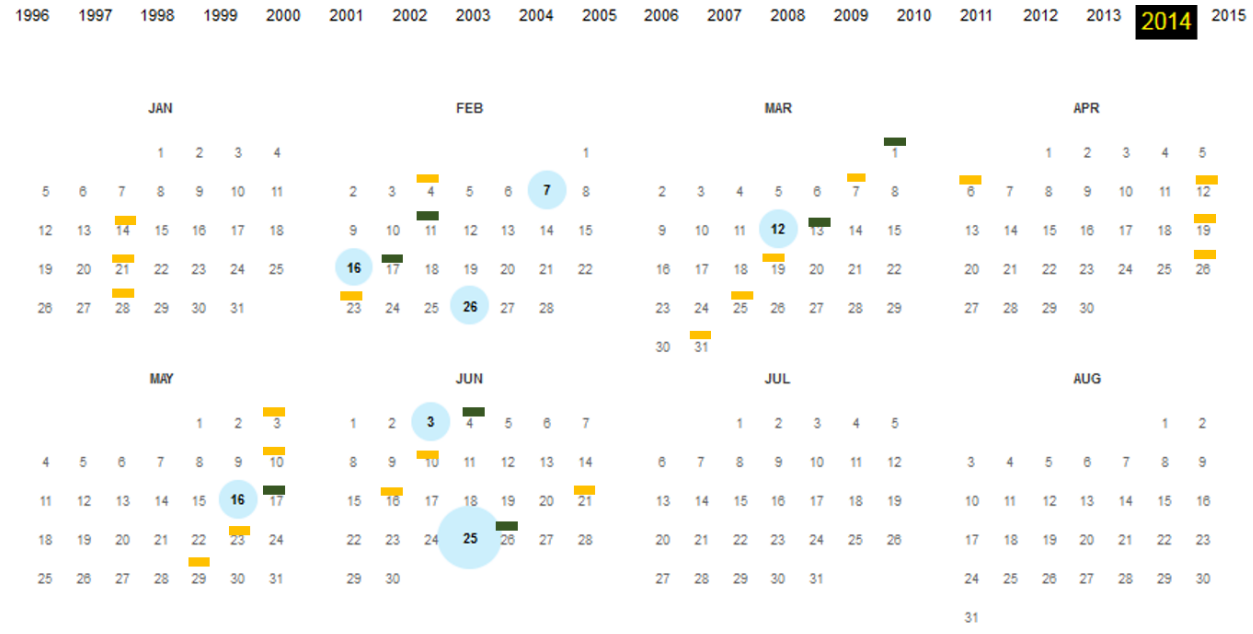


Figura 4-49 Algoritmo binomial con históricos aplicado al sitio de la Universidad del Zulia http://www.luz.edu.ve/ [23]

REFERENCIAS

[1]. **Patrimonio Cultural:** Recuperado en febrero de 2013. Disponible en: http://es.wikipedia.org/wiki/Patrimonio_culturalhttp://netpreserve.org/

[2]. **Internet Archive:** Consultado en febrero de 2013. Disponible en: http://www.archive.org/

[3]. **Patrimonio Cultural:** recuperado en febrero de 2013. Disponible en: http://es.wikipedia.org/wiki/Patrimonio_culturalhttp://netpreserve.org/

[4]. **AXES. (2012).** Recuperado el 20 de Octubre de 2013, de Acces to Audivisual Archive: http://www.axes-project.eu/?p=1032

[5]. **García, J. C. (2013).** Recuperado el 15 de Enero de 2013, Scalable Analytics Institute. Obtenido de http://oak.cs.ucla.edu/~cho/papers/cho-freq.pdf

[6]. **Mitra, Q. T. (2010).** Recuperado el 15 de Enero de 2015, *Penn State Personal Web Server*. Obtenido de Penn State: http://www.personal.psu.edu/pum10/tois-tan.pdf

[7]. **Bennett, K. R. (2013).** Recuperado el 16 de Enero de 2015 Microsoft Research. Obtenido de Microsoft: http://research.microsoft.com/en-us/um/people/pauben/papers/wsdm2013-change-radinsky-bennett.pdf

[8]. **Masanès, Julien(2006).** *Web Archive.* New York: Springer, 2006. ISBN-10 3-540-23338-5.

[9]. **UNESCO.(2003).** *DIRECTRICES PARA LA PRESERVACIÓN DEL PATRIMONIO DIGITAL*. Australia: Biblioteaca Nacional de Australia

[10]. **ISO**. (2009). ISO. 28500 information and documentation-WARC file format . Nueva Zelanda.

[11]. **George Canavos (1988).** Probabilidad y Estadística, aplicaciones y métodos . México: McGraw-Hill.

[12]. **Collins-Sussman, B. F. (2004).** *Version Control with Subversion.* N.Y: O'Reilly.

[13]. **Git (2015).** Recuperado el 11 de Enero de 2015, *Git*: http://git-scm.com/

[14]. **Github (2015).** Recuperado el 15 de Febrero de 2015, *Github*: https://github.com/

[15]. Python (2015). Recuperado el 2 de Febrero de 2015, *Python foundation*: https://www.python.org/

[16]. Escuelaweb (2014). Recuperado el 7 de Enero de 2015, *Php, Phython o Ruby Que son y para que sirve cada uno:* https:// blog.escuelaweb.net/php-python-o-ruby-para-que-sirve-cada-uno/

[17]. Scipy (2015). Recuperado el 7 de Marzo de 2015, *Scipi scientific library for python:* https:// www.scipy.org

[18]. Django (2015). Recuperado el 10 de Diciembre de 2014, *Django the web framework for python:* https://www.djangoproject.com/

[19]. Fielding, R. T. (2013). Obtenido de http://roy.gbiv.com/untangled/2008/rest-apis-must-be-hypertext-driven

[20]. The Agile Unified Process, recuperado en Febrero de 2014. Disponible en:http://netpreserve.org/http://www.ambysoft.com/unifiedprocess/agileUP.html

[21]. Ingeniería de software basada en componentes, recuperado en febrero de 2013. Disponible en:http://netpreserve.org/http://es.wikipedia.org/wiki/Ingenier%C3%ADa_de_software_basada_en_componentes**.**

[22]. A brief Comparation, recuperado en Noviembre de 2014. Disponible en: http://biz30.timedoctor.com/git-mecurial-and-cvs-comparison-of-svn-software/

[23]. Wayback Machine recuperado en Febrero de 2015. Disponible en: https://archive.org/web/

[24]. Simuladores, recuperado en Diciembre de 2014 Disponible en: http://www.prepa9.unam.mx/academia/cienciavirtual/simuladores.htm

[25]. GIt, recuperado en Diciembre de 2014 Disponible en: http://www.git-scm.com/

[26]. Real Academia Española, http://www.rae.es/ recuperado en Marzo de 2015 Disponible en: http://www.rae.es/

[27]. Guias Probabilidad y Estadisticas, http://www.rae.es/ recuperado en Enero de 2015 Disponible en: http://www.ciens.ucv.ve/portalasig/probabilidad_y_estadistica/2-2013/

ANEXO A

Instalación de Heritrix versión 3.1.1

Se descarga Heritrix de su página web:

http://builds.archive.org:8080/maven2/org/archive/heritrix/heritrix/3.1.1/heritrix-3.1.1-dist.zip. Una vez descargado se descomprime el archivo, usando el comando que puede apreciarse en la Figura 98.

```
1   unzip heritrix-3.1.1-dist.zip
```

Figura 98. Descomprimir Heritrix

- **JAVA_HOME**,: Debe apuntar al directorio de instalación de JRE 1.6, por ejemplo JAVA_HOME=/usr/local/java/jre
- **HERITRIX_HOME**: debe apuntar al directorio de instalación de heritrix, por ejemplo HERITRIX_HOME=/home/user/heritrix3.1.1
- **JAVA_OPTS**: Esta variable es usada para definir la cantidad de memoria asignada a heritrix, por ejemplo un valor igual a -Xmx1024M, indica que se está asignando 1Gb de memoria

Adicionalmente se le debe dar permiso de ejecución al archivo HERITRIX_HOME/bin/heritrix, como se muestra en la Figura 99.

```
1   chmod u+x $HERITRIX_HOME/bin/heritrix
```

Figura 99. Heritrix permisos

Ejecutar Heritrix

Para ejecutar Heritrix se usa el comando que se puede apreciar en la Figura 100.

```
1   $HERITRIX_HOME/bin/heritrix
```

Figura 100. Comando para ejecutar Heritrix

API REST

Heritrix usa REST para exponer sus funcionalidades, la implementación REST de heritrix está basada en Restlet que es un framework RestFul para java.

Heritrix expone esta API a través de HTTPS, a través de este protocolo se hacen peticiones para recuperar o modificar configuraciones y manejos de rastreos.

Cualquier cliente que soporte HTTPS puede ser usado para invocar el API REST, por ejemplo se puede usar el cliente de línea de comandos curl.

CrearJobs

Para crear un nuevo job se utiliza el comando que puede apreciarse en la Figura 101.

```
curl -v -d "createpath=myjob&action=create" -k -u <usuario>:<contraseña>
--anyauth  --location https://<HeritrixHost>:8443/engine
```

Figura 101. Crear Job

Dónde:

createpath: Nombre del job

<usuario>: usuario de heritrix

<contrasena>: contraseña del usuario heritrix

<HeritrixHost>: Host donde está corriendo heritrix

Construir el Job

Luego de que un job es creado se debe leer el archivo de configuración necesario para correr el rastreo, a partir de las especificaciones dictadas en este archivo se construye el job, el comando utilizado puede apreciarse en la Figura 102.

```
curl -v -d "action=build" -k -u <usuario>:<contraseña> --anyauth
--location https://<HeritrixHost>:8443/engine/job/<nombreJob>
```

Figura 102. Construir Job

Dónde:

<usuario>: usuario de heritrix

<contrasena>: contraseña del usuario heritrix

<HeritrixHost>: Host donde está corriendo heritrix

<nombreJob>: Nombre del job

Lanzar el job

Una vez el job ha sido construido, es necesario lanzarlo o ponerlo a correr, para ello se usa el comando que puede apreciarse en la Figura 103.

```
curl -v -d "action=launch" -k -u <usuario>:<contraseña> --anyauth
--location https://<HeritrixHost>:8443/engine/job/<nombreJob>
```

Figura 103. Lanzar Job

donde:

<usuario>: usuario de heritrix

<contrasena>: contraseña del usuario heritrix

<HeritrixHost>: Host donde está corriendo heritrix

<nombreJob> : Nombre del job

Quitar la pausa a un job

Por defecto el rastreo es lanzando en modo pausa, por lo que es necesario quitarle la pausa para que pueda comenzar a procesar, el comando usado puede apreciarse en la Figura 104.

```
curl -v -d "action=unpause" -k -u <usuario>:<contraseña> --anyauth
--location https://<HeritrixHost>:8443/engine/job/<nombreJob>
```

Figura 104. Despausar job

donde:

<usuario>: usuario de heritrix

<contrasena>: contraseña del usuario heritrix

<HeritrixHost>: Host donde está corriendo heritrix

<nombreJob>: Nombre del job

ANEXO B Instalación de Python

Se puede utilizar este comando para saber si tenemos instalado python y en que versión:

$ apt-cache search python | egrep "^python2.[0-9] " --color

Escribir el siguiente comando para instalar **python en linux** en la version 2.x:

$ sudo apt-get install python2.7

Escribir el siguiente comando para **instalar python en linux** en la version 3.x:

$ sudo apt-get install python3.1

Una nota para los usuarios de Red Hat / RHEL / CentOS:
Puede instalar python de la siguiente manera:

$ sudo yum install python

O instalar python de esta otra manera:

yum install python

Verificar la version de python que fue instalada

Escribir el siguiente comando:

$ python –version

python get-pip.py

Anexo C Instalación de Django mediante el gestor de paquetes de python pip

instalar pip ingresando el siguiente comando:

$ python get-pip.py

Para instalar django ingresar el siguiente comando:

$pip install django