Universidad de las Ciencias Informáticas

Facultad 3

"Biblioteca de Algoritmos para la Detección de Plagio"

Trabajo de Diploma para optar por el título de

Ingeniero en Ciencias Informáticas

Autores:

Giselle Pérez Carreras

Ivette María Suárez Muñoz

Tutor:

Lic. Rolan Rober Bullain Diéguez

Ciudad de la Habana, 2010

"Año 52 de la Revolución"

DECLARACIÓN DE AUTORÍA

Nosotras declaramos que somos las únicas autoras de este trabajo y autorizamos a la Universidad de las Ciencias Informáticas a hacer uso del mismo en su beneficio.

Para que así conste firmamos la presente a los ____ días del mes de _____ del año _____.

Autores:

Ivette María Suárez Muñoz

_____

Giselle Pérez Carreras

_____

Tutor:

Lic. Rolan Rober Bullain Diéguez

_____

"…aquel que ve cometer un fraude y no lo combate enérgica y revolucionariamente, comete una traición a sus compañeros una verdadera traición a sus compañeros, a su escuela y a la educación del país; comete una traición a sus principios. Porque no queremos formar solo sabios, aparte de que nunca los fraudulentos serán sabios; no queremos formar falsos sabios. Y, sobre todo, queremos formar hombres puros y limpios, hombres leales y honestos, hombres de principios."

Fidel Castro.

AGRADECIMIENTOS

A mis padres por su dedicación hacia mí y todo el cariño que me han dado siempre, gracias por formarme en la vida como la persona que soy y estoy muy orgullosa de ser tanto como ustedes, por marcar el camino correcto siempre a seguir, por confiar en mí, los quiero con la vida.

A mis abuelos que me han dado todo lo que tienen y lo que no, porque logre ser alguien en la vida, nunca sería capaz de defraudarlos, no sé qué haría sin ustedes, aunque esté lejos siempre están en mi corazón.

A mi hermano que siempre ha confiado en mí y será el segundo ingeniero de la familia porque me quiere tanto como yo a él y por eso no me va a defraudar.

A Yandy que ha sido de gran apoyo para mí durante estos tres años, gracias por tu dedicación y cariño, que me han ayudado a seguir adelante y romper cualquier obstáculo que se ha puesto en el camino, gracias por no dejarme ir, te quiero y sabes que ocupas un lugar especial en mi corazón.

A mi familia en general que me han apoyado para siempre seguir adelante.

A mis amigos que hacen tanta falta cuando la familia está tan lejos, sobre todo cuando son como Ana, que me ha tendido la mano incondicionalmente en todos los momentos difíciles.

A mi tutor Rolan he aprendido mucho de usted.

A la nueva familia que he adquirido hace tres años atrás por haberme ayudado tanto.

A este país que me ha dado la oportunidad de graduarme como lo que siempre quise ser, una profesional.

Ivette María Suárez Muñoz

A nuestro Comandante en Jefe y a la Revolución, por la oportunidad de convertirnos en profesionales.

A la Universidad de las Ciencias Informáticas, donde viví cinco inolvidables años de mi vida.

A mis padres que confiaron en mí siempre, que me han amado y guiado por el camino correcto, quienes dieron con amor todo cuanto pudieron para que lograra este éxito.

A mi tutor Rolan Rober Bullain Diéguez por la dedicación brindada para que este trabajo saliera adelante, por ser tan exigente y preocupado, por sus sugerencias, en fin por su gran apoyo a lo largo de todos estos meses de trabajo.

A Yandy por su ayuda incondicional, en todo momento.

A mi tía Pitiqui, que siempre ha estado ahí, apoyándome en todo como una madre.

A mis abuelos, a mis tíos, a mis primos, a mi novio, a mis compañeras de cuarto, a mis vecinos, a mis amigos.

En fin, les agradezco a todas las personas que de una forma u otra han hecho posible la realización de este sueño.

Giselle Pérez Carreras

DEDICATORIA

…A mis padres, mi hermano y mis abuelos…

Ivette María Suárez Muñoz

…A mi familia, y en especial a mi mamá y a mi papá por todo el apoyo, dedicación y confianza que siempre han depositado en mí. Sin ustedes no hubiera podido lograr lo que he sido hasta hoy…

Giselle Pérez Carreras

RESUMEN

El plagio se ha incrementado notablemente en los últimos años debido al gran desarrollo que ha alcanzado el hombre en la tecnología. Con el uso de Internet la tentación a copiar textos de otra persona y tomarlos como propios es muy usual principalmente en los estudiantes. Debido a esto toma gran relevancia el desarrollo de métodos que faciliten la detección de plagio.

En este trabajo se presenta un estudio de las causas que conllevan a los estudiantes a cometer plagio, así como el estado del arte de los métodos existentes para detectar plagio: herramientas y algoritmos. Se ha decidido implementar una biblioteca de algoritmos para la detección de plagio pues las herramientas existentes hasta el momento no satisfacen completamente las necesidades de los centros académicos.

Se han hecho evaluaciones para determinar la confiabilidad de cada uno de los algoritmos estudiados, seleccionando los más eficientes para conformar la biblioteca y se han implementado en el lenguaje de programación python. De esta manera se reduce las probabilidades que los estudiantes cometan plagio en trabajos orientados de corte científico y se hace más fácil la tarea de los profesores a la hora de detectarlo.

PALABRAS CLAVES: Plagio, Internet, Biblioteca de Algoritmos, Python.

TABLA DE CONTENIDOS

INTRODUCCIÓN

El hombre ha sido capaz mediante años de evolución y estudio de desarrollar la tecnología a tal punto de crear una red virtual, que ha revolucionado su vida y ha creado una nueva era: "La Era de Internet". Desde todas partes del mundo, cualquier persona puede acceder fácilmente a grandes volúmenes de información, servicios y comunicación usando computadoras conectadas entre sí.

En los últimos años, Internet ha alcanzado un gran auge en prácticamente todas las áreas del conocimiento humano. Ha despertado el interés de todo el mundo y cada día, más personas se conectan y lo hacen parte de su vida diaria, haciendo de éste definitivamente, el motor que mueve al planeta entero al compás de las evoluciones técnicas.

Internet, a modo de fuente de fuentes, tal como señalara Comas (Comas, 2007) ha expandido el número de recursos y contenidos a los que se tiene acceso gratuitamente e instantáneamente; este hecho constituye en sí mismo una revolución positiva sin posibilidad aparente de vuelta atrás y con claras implicaciones en términos del proceso educativo en general. Internet y sus peculiaridades han modificado la manera en que se accede, consulta y emplea la información en el ámbito educativo y académico. Se ha convertido en fuente de referencia primordial para la documentación académica por quienes forman parte de cualquier proceso de enseñanza-aprendizaje. La gran cantidad de recursos, materiales, artículos, multimedia, libros, revistas especializadas, bases de datos, portales temáticos, accesibles a través de Internet, la mayoría de los cuales no son accesibles desde otro espacio o canal, la facilidad y comodidad de acceso a los mismos y la gratuidad de la mayor parte de ellos, son sin dudas factores de gran ayuda para cualquiera que desee consultar información con finalidades académicas y en cierta manera, no se concibe hoy en día un trabajo académico, el planteamiento de una investigación, la ampliación de contenidos de una asignatura o el trabajo en un aula, sin contar con la ayuda de Internet como elemento de consulta.

Esta disponibilidad de las comunicaciones, unido al fácil manejo de la información y la llegada de los soportes de almacenamiento masivo portables, ha conllevado a un incremento de la actividad de "copia y pega", como la llamara Fernando Bordignon (Bordignon, y otros, 2005) y con esta la posibilidad de plagio.

El plagio es una falta grave que se puede ver reflejada en muchos ámbitos. Según la Real Academia Española: *Plagiar, copiar en lo sustancial obras ajenas, dándolas como propias.*

El plagio es uno de los grandes problemas que hoy en día enfrenta nuestra sociedad. Según La Torre (La Torre, 1994), este fenómeno constituye un atentado al derecho de autor, pues en esencia, significa desconocer la paternidad del autor y por consiguiente, la relación que le une con su obra, sustrayéndole todo conocimiento e ignorándole toda aportación creativa.

En el ámbito académico, el plagio no es un fenómeno nuevo, ha sido un problema clásico en las instituciones académicas, que antes de la llegada de Internet se realizaba utilizando como único recurso la información contenida en bibliotecas, diarios, periódicos y trabajos desarrollados por compañeros de estudio; aunque con el uso del Internet se ha incrementado considerablemente, debido a que como bien dijera Comas (Comas R., 2006), la mayor parte de investigaciones y análisis recientes que sobre el tema se han desarrollado señalan que el aumento acelerado en los índices de penetración de Internet en nuestra sociedad y la mayor facilidad de acceso a contenidos e información que se pueden encontrar en la Red, han provocado un auge en las prácticas de plagio entre el alumnado en los distintos niveles del sistema educativo.

Esto ha conllevado a que un grupo de universidades del mundo se hayan proclamado en contra de este fenómeno, como son:

- La Universidad Católica del Perú, que ha publicado diversos artículos como: “Por qué y cómo debemos combatir el plagio”. Publicado el 18/04/08.
- La Universidad Nacional de Colombia, el 4/09/2007 inició una cruzada contra el plagio.
- La Universidad Sergio Arboleda, que publicó la Primera Edición del material “Anotaciones sobre el plagio” en julio del 2008, por Sonia Jannett Girón Castro, Docente del Departamento de Gramática.
- La Universidad de Granada, la cual crea una plataforma contra el plagio. Esta iniciativa partió de la profesora: Rosa María Medina Doménech, del departamento de Anatomía Patológica e Historia de la Ciencia.

Como la profesora Rosa M. Medina, muchos son los que se han proyectado en contra de esta tendencia de “copia y pega”, de ahí que en los últimos años se han desarrollado diversas aplicaciones informáticas para la detección del plagio. Entre estas aplicaciones informáticas para detectar el plagio se encuentran: Approbo, Educared, Turnitin, entre otras que son abordadas en el Capítulo I.

En Cuba se han hecho campañas contra el fraude académico, el Comandante Fidel Castro en su discurso del día 4 de septiembre de 1978 en el Instituto Politécnico de la Salud de la provincia de Camagüey, hace un llamado de conciencia sobre la negatividad del fraude y la necesidad de luchar contra esto. El plagio es considerado un tipo de fraude.

Particularmente en la Universidad de las Ciencias Informáticas (UCI), donde se llevan a cabo un grupo de eventos de relevante connotación: como la Jornada Científica, Mi Web por Cuba, trabajos de diploma y maestría, incluso diariamente se desarrollan una serie de trabajos investigativos en el área docente orientados en clases, no se encuentra exenta de esta situación que está afectando a la sociedad mundial.

Datos del departamento de investigaciones de la UCI, arrojan que actualmente la universidad carece de una herramienta capaz de detectar el plagio en dichos trabajos investigativos y debido al gran cúmulo de trabajos realizados, a los profesores se les hace casi imposible revisar manualmente cada uno de dichos trabajos en la búsqueda de plagio, dando por resultado que se han presentado trabajos que han sido realizados bajo las tendencias del “copia y pega”.

De lo planteado anteriormente se define el siguiente **problema científico**: el proceso de identificación de fraude mediante el plagio de documentos se realiza de forma manual, lo que implica que sea poco confiable, no exhaustivo e impreciso.

Para dar solución al problema anterior se plantea como **objetivo general**: desarrollar una biblioteca de algoritmos para la detección de plagio.

Por tanto, el presente trabajo centra su **objeto de estudio** en: la recuperación de información y como **campo de acción** se tiene: los algoritmos y técnicas para la detección de plagio.

Para dar solución al problema científico se parte de la siguiente **hipótesis:** si se desarrolla una biblioteca de algoritmos capaz de detectar el plagio, posteriormente se podrá utilizar en la realización de un software que permitirá una mejor identificación de los trabajos plagiados, elevando la calidad y eficiencia de los trabajos de carácter científico y docente.

Los **objetivos específicos** que nos hemos propuesto desarrollar para dar cumplimiento al objetivo general son:

1. Identificar técnicas y algoritmos para la detección de plagio.
2. Definir las características deseables para la biblioteca de algoritmos de detección de plagio.
3. Reutilizar código libre.
4. Implementar la biblioteca de clases de detección de plagio.

Los **resultados esperados** del presente trabajo son que el desarrollo de una biblioteca de algoritmos permita:

1. Dado un documento de corte científico, determine si contiene plagio o no.

2. Identifique las partes del documento que contienen plagio y el documento original fuente del mismo.

El presente trabajo está estructurado en dos capítulos que abarcan toda la investigación realizada:

En el **Capítulo I:** Fundamentación teórica, se hace alusión a los conceptos y definiciones necesarias para llevar a cabo el desarrollo de la biblioteca así como la tecnología a utilizar.

En el **Capítulo II:** Desarrollo de la biblioteca y su validación, se identifican las funcionalidades a cumplir, se definen los algoritmos que componen la biblioteca y se valida la solución propuesta.

CAPÍTULO I: FUNDAMENTACIÓN TEÓRICA

1.1. Introducción

En nuestros días, los problemas concernientes al plagio, y en particular al plagio de textos, se han visto incrementados debido al fácil acceso a grandes fuentes de información a través de medios electrónicos tales como bibliotecas electrónicas y por supuesto, Internet. Desafortunadamente, su detección es prácticamente imposible de forma manual. Por ello, según un estudio realizado por Barrón Cedeño (Barrón Cedeño, 2008), es importante desarrollar mecanismos automatizados que permitan realizar la tarea de detección de plagio en un tiempo razonable. De esta manera, se puede combatir la enorme tentación de plagiar textos provenientes de cualquier lado. Por fortuna, como podrá observarse a través de la lectura de este trabajo, este problema que se ha visto beneficiado por el desarrollo de la tecnología informática, puede ser atacado por ella misma.

El proceso de recuperación de la información

El desarrollo de la biblioteca de algoritmos para la detección de plagio, ha llevado al estudio profundo de la recuperación de la información que es la disciplina de la Inteligencia Artificial que se ocupa de la gestión de grandes volúmenes de información.

Sistema de recuperación de información se puede definir como**:** proceso donde se accede a una información previamente almacenada, mediante herramientas informáticas que permiten establecer ecuaciones de búsquedas específicas. Dicha información ha debido de ser estructurada previamente a su almacenamiento. El proceso de recuperación se lleva a cabo mediante consultas a la base de datos donde se almacena la información, mediante un lenguaje de interrogación adecuado. Es necesario tener en cuenta los elementos claves que permiten hacer la búsqueda, determinando un mayor grado de pertinencia y precisión, como son: los índices, palabras clave, tesauros y los fenómenos que se pueden dar en el proceso como son el ruido y silencio documental. Uno de los problemas que surgen en la búsqueda de información es: si lo que se recupera es "mucho o poco" es decir, dependiendo del tipo de búsqueda se pueden recuperar multitud de documentos o simplemente un número muy reducido. A este fenómeno se denomina Silencio o Ruido documental (Pinto Molina).

- *Silencio documental*: Son aquellos documentos almacenados en la base de datos pero que no han sido recuperados, debido a que la estrategia de búsqueda ha sido demasiado

específica o que las palabras claves utilizadas no son las adecuadas para definir la búsqueda.

- *Ruido documental*: Son aquellos documentos recuperados por el sistema pero que no son relevantes. Esto suele ocurrir cuando la estrategia de búsqueda se ha definido demasiado genérica.

El proceso de indexación

Se ha estudiado además el proceso de indexación para obtener ideas sobre cómo ordenar la información que se tendrá disponible, para realizar la búsqueda de documentos originales que pueden ser la fuente de plagio de documentos sospechosos.

Indexar no es más que elaborar un índice que contenga de forma ordenada la información para obtener óptimos resultados al realizar una búsqueda. Los motores de búsqueda más importantes como Google, contienen la información de la web indexada, cuando es realizada una consulta, se dirige hacia los resultados precisos. Según la Real Academia Española: *Indexar, registrar ordenadamente datos e informaciones, para elaborar su índice.*

La indexación consta de los siguientes pasos (Fernández González):

- Recorrer todos los documentos sobre los que se desea buscar: éste puede ser un conjunto finito conocido o no.
- Procesar el documento a indexar: se descompone hasta obtener la lista de palabras que lo forman. Este proceso puede ser muy sencillo, como en los archivos de texto plano, que tiene todas las palabras separadas por espacios u otros caracteres, o muy complejo, como un documento PDF que debe ser decodificado, separando el formato y las imágenes, extrayendo solamente el texto plano.
- Con la lista de palabras de un documento, se crea el índice invertido: se guarda en una lista apuntando en qué documento se ha encontrado cada palabra. Habrá algunas palabras que solo aparezcan en un documento (la búsqueda de esa palabra proporciona un solo resultado), mientras que otras palabras más comunes aparecerán en muchos documentos.
- Opcionalmente se puede almacenar el documento, en su estado actual, en la propia base de datos.

1.2. El plagio de palabras

Existen muy diversos tipos de plagio: de música, de imágenes, de palabras e incluso de ideas. El tipo de plagio que nos interesa en este momento es el de palabras, en particular aquellas que se

encuentran en una realización escrita, es decir, el plagio en texto. Dentro de este ámbito, plagiar implica incluir fragmentos de texto que se encuentran en documentos escritos por otro autor en un documento propio sin incluir el crédito correspondiente (Barrón Cedeño, 2008).

1.2.1. Clasificaciones

Existen varios criterios en cuanto a la clasificación de los tipos de plagios existentes, dos de los cuales son planteados por Barrón Cedeño (Barrón Cedeño, 2008) en su investigación:

El Departamento de Química de la Universidad de Kentucky ha delimitado lo siguiente:

- El caso más sencillo se refiere a la copia directa de material original sin realizar alguna modificación.
- Otro de los casos, que a primera vista se refleja más complicado de analizar de manera automática es la reescritura. En este caso, en lugar de simplemente copiar algún fragmento de texto, éste se reescribe por medio del uso de sinónimos o del cambio en el orden de las palabras que lo conforman.

Otro criterio de clasificación ha destacado tres tipos de plagio:

- Palabra por palabra: se trata de la copia, incluso modificada, de fragmentos de texto sin incluir su fuente.
- De referencias: se da cuando una referencia es observada en un documento y se transcribe al que se está escribiendo sin haber leído realmente la fuente.
- De autoría: ocurre cuando un autor dice ser creador de un trabajo que fue sustancialmente realizado por otro.

Dentro de esta última clasificación, los plagios que se consideran factibles de ser detectados por medio del análisis de texto son: el plagio de palabra por palabra y el plagio de autoría. Esto se debe a que se ven reflejados en la lectura de un texto. Por su parte, el plagio de referencias es prácticamente imposible de detectar. ¿Si tanto la cita según Barrón Cedeño (Barrón Cedeño, 2008) como la referencia son incluidas correctamente, cómo podría detectarse un caso de plagio?

Un tercer criterio de clasificación del plagio es la de Comas (Comas, 2007), quien incluye ya el término ciber-plagio.

Ciber-plagio intencional:

- Comprar o descargar un trabajo, artículo o proyecto, desde una web dedicada a la venta y/o intercambio de trabajos académicos y presentar la obra como propia.
- Copiar un texto completo, desde una web o un archivo descargado de Internet, y presentarlo como propio sin citas ni referencias.
- Copiar partes o párrafos de distintos textos extraídos de Internet y presentarlos en un texto único como propios.
- Copiar de Internet y traducir un trabajo completo, partes del mismo o resultados de investigaciones.

Ciber-plagio accidental:

- Uso de parafraseados inadecuados.
- Mala citación de los recursos y bibliografía utilizados.

1.2.2. Causas asociadas al fenómeno

Se trata sin duda de una de las áreas más analizadas y sobre la que existe más literatura, aunque ésta, esté basada principalmente en apreciaciones e ideas devenidas de la experiencia de académicos que han abordado el tema y no en investigaciones rigurosas. Destaca de todas maneras un trabajo de investigación pionero en este campo en el cual se analizan las causas del ciber-plagio desde la óptica de los estudiantes universitarios y docentes. Según el estudio realizado por Comas (Comas, 2007), los estudiantes plagian, por:

- Intentar obtener mejores calificaciones y resultados académicos.
- Pereza y mala gestión del tiempo dedicado al estudio y elaboración de trabajos.
- Facilidad y comodidad de acceso a material vía Internet.
- Desconocimiento de las normas básicas a seguir para la elaboración de un trabajo académico.

A la anterior lista habría que añadir el tipo de metodología seguida en muchos casos por los docentes a la hora de plantear la asignatura y los trabajos académicos que se pide a los alumnos, en cierta medida podrían incitar a la adopción de prácticas tendente al plagio por parte del alumnado. Otro elemento a tener en cuenta que tiene que ver con la brecha digital asociada a la diferencia generacional estudiante-docente, pues provoca el enfrentamiento entre alumnos muy hábiles en el uso de las Tecnologías de la Información y la Comunicación (TIC) y docentes poco hábiles en el uso de las mismas. Este hecho provoca en los alumnos una sensación de confianza en no ser descubiertos por parte del profesor y aumenta su propensión al plagio (Comas, 2007).

1.2.3. Posibles soluciones

En el ámbito de las soluciones frente al ciber-plagio, se pueden separar en dos grandes campos, *Prevención* y *Detección* (Comas, 2007).

En cuanto a la *prevención* del plagio, existen básicamente tres líneas de trabajo, las cuales se exponen a continuación.

a) Regulación punitiva: muchas universidades y centros educativos de los EE.UU, Canadá, Reino Unido, Australia, Hong-Kong, Alemania, Cuba y países escandinavos han desarrollado estrictas reglamentaciones con vistas a rebajar la incidencia del plagio académico, la mayoría de estas regulaciones se inscriben en un marco general de compromiso ético y códigos de honor universitarios que los estudiantes deben seguir. Las consecuencias de actividades probadas de plagio pueden castigarse con medidas que van desde la realización del trabajo plagiado por segunda vez, hasta la expulsión del centro o la supresión de una titulación ya concedida, como sucedió en un conocido caso en la Universidad de Virginia en el 2002.
b) Información y concienciación: algunos centros intentan influir y concienciar a los estudiantes acerca del plagio académico mediante campañas de información acerca de, entre otras cosas, qué es el plagio, qué consecuencias suceden a su infracción y maneras de evitarlo. Estas campañas se desarrollan a través de información presente en la Web de la universidad; información suministrada a través de póster, trípticos; a través de información explícita sobre la regulación del centro adjunta al formulario de matriculación.
c) Formación dirigida al alumnado y profesorado: el tercero de los elementos apuntados tiene que ver con la formación de los alumnos en aspectos tales como: cómo escribir un trabajo académico; cómo buscar información y documentarse y cómo citar los recursos usados; cómo evaluar información extraída de Internet. También se trabaja en la formación del cuerpo docente en aspectos relativos a: maneras de detectar el plagio, metodologías innovadoras de trabajo.

Por lo que respecta a la *detección*, se habla principalmente de dos tipos de sistemas:

- Basada en la experiencia y análisis del docente.
- Mediante instrumentos tecnológicos.

Por lo que respecta al primer nivel de detección los docentes pueden intentar detectar el plagio en trabajos académicos prestando atención a aspectos como:

- Incongruencia entre ideas, teorías e hipótesis expresadas en el trabajo.

- Incongruencia en los estilos de escritura (narración, usos gramaticales) dentro del texto.
- Incongruencia en el desarrollo "lógico" del trabajo.
- Incongruencia entre las ideas, teorías, hipótesis, presentadas por el alumno y trabajos anteriores del mismo.
- Incongruencia en la bibliografía y fuentes citadas.
- Incongruencia en el formato de presentación, por ejemplo: existencia de saltos de página incorrectos; existencia de diferentes tipos de fuente, tamaño o existencia de números de página no consecutivos.
- Incongruencia en las citas.

Resulta evidente, de todas maneras, que detectar el plagio en un trabajo académico a simple vista, sobre todo si el alumno ha sido precavido y cuidadoso, es una tarea muy difícil dado que parece imposible controlar toda la bibliografía en formato digital existente, que no cesa de aumentar día a día. Ante ello, en los últimos años se han desarrollado, principalmente en los EEUU, programas informáticos de detección de plagio que suelen operar a través de una doble comparación:

- Comparan los trabajos con recursos accesibles en Internet a través de una búsqueda orientada en la Red.
- Comparan los trabajos con bases de datos propias de trabajos académicos presentados en años anteriores.

A partir de esta doble comparación, se establece si el trabajo presenta indicios de plagio o no. Muchas universidades de los EEUU, Reino Unido y Canadá han adoptado estos programas para la detección del plagio.

Otra posibilidad tecnológica, bastante más rudimentaria, menos rigurosa y que supone la dedicación de más tiempo, es la introducción de frases sueltas o párrafos del trabajo del alumno en un buscador generalista de Internet de cara a encontrar documentos, que hayan podido usarse para el plagio del trabajo.

1.3. Aplicaciones informáticas para la detección de plagio

Como vía de detección, en cuanto a las posibles soluciones se han desarrollado en el mundo una serie de aplicaciones informáticas antiplagio, las cuales han sido estudiadas, para determinar si alguna cumple con los objetivos propuestos.

1.3.1. Sherlock

Sherlock es un programa que encuentra similitudes entre los documentos textuales. Se utilizan firmas digitales para encontrar piezas similares de texto. Una firma digital es un número que está formado por varias palabras de inflexión en la entrada en una serie de bits y unirse a los bits en un número. Sherlock trabaja en los archivos de texto como el ensayo, el código fuente de los archivos de computadora y otras tareas en forma digital. Esta herramienta fue desarrollada por la Universidad de Sydney. Entre sus principales ventajas se tiene que es una herramienta de código abierto y trabaja con código escrito en los lenguajes Java, C y texto natural. Una de las desventajas que posee es que no busca similitudes en Internet, ya que solo trabaja con archivos locales. Además de que al no utilizar el documento en su totalidad, no se puede afirmar que sean completamente iguales (Diaz, y otros).

1.3.2. Simian

Simian, fue desarrollada por una consultora de Australia llamada REDHILL. Su principal ventaja es que identifica la duplicación de códigos escritos tanto en JAVA, como en C#, C, C++, JSP, ASP, HTML, XML, Visual Basic y texto natural. Aunque posee la desventaja de que el ingreso de los parámetros es a través de líneas de comando, ya que no cuenta con una interfaz gráfica. Además de que no busca similitudes en Internet, pues solo trabaja con archivos locales (Diaz, y otros).

1.3.3. Jplag

Jplag es una herramienta web programada en Java, que busca similitudes en el código fuente de los programas a comparar, esto lo hace no sólo buscando similitudes en el texto del código, sino que también revisa sintaxis y estructuras de programas, para aquellos casos donde se buscan disfrazar las copias. Fue desarrollada por la Universidad de Karlsruhe de Alemania. Dentro de los lenguajes de programación que soporta están: C, C++, Java y texto natural. Al igual que otras herramientas, este sistema sólo trabaja con archivos locales. Entre sus principales ventajas, se encuentra que se pueden hacer comparaciones de varios códigos a la vez, unidos a que su interfaz es a través de línea de comando o puede implementarse en cliente propio. Además que para el manejo de archivos enviados, puede utilizarse el cliente brindado por la universidad que lo desarrolló o bien crear un cliente propio como se mencionaba anteriormente. Posee una arquitectura de trabajo de Cliente/Servidor, que sólo permite la programación del cliente, pero el servidor es el que realiza las operaciones de comparación, se encuentra en la universidad donde se desarrolló, lo que constituye una de sus principales desventajas (Diaz, y otros).

1.3.4. Tester SIM

Tester SIM es una herramienta desarrollada por la Universidad de Ámsterdam. Trabaja con los lenguajes de programación: C, JAVA, Pascal y texto natural. Entre sus principales ventajas se encuentra que es de código abierto y que se puede procesar un conjunto de archivos viejos contra un conjunto de archivos nuevos. Como desventajas presenta que sólo trabaja con búsqueda de similitudes en archivos locales y que no cuenta con una interfaz gráfica, por lo que los distintos parámetros se ingresan por líneas de comandos (Diaz, y otros).

1.3.5. Turnitin

La herramienta Turnitin, es una solución antiplagio, creada por la empresa estadounidense iParadigms. Su principal ventaja es que rastrea para usuarios de los EE.UU. y Europa una enorme cantidad de sitios web, trabajos de estudiantes, diarios, revistas científicas y libros. Aunque tiene como desventaja que es un servicio pago, que oscila entre 800 y 1.000 euros (Turnitin).

1.3.6. Wcopyfind

Este programa examina una colección de archivos de documentos. Extrae las porciones de texto de esos documentos y se ve a través de ellos para hacer coincidir las palabras en frases de una duración mínima determinada. Cuando encuentra dos archivos que comparten palabras suficientes en esas frases, WCopyfind genera archivos de informe HTML. Estos informes contienen el texto del documento con las frases coincidentes subrayado. Entre sus principales ventajas se encuentra que, es un software libre, además de que puede manejar texto, HTML, y algunos archivos de procesador de textos (en particular, documentos de Microsoft Word en la antigua .doc). Posee algunas desventajas como el no poder manejar documentos de Microsoft Word modernos como .docx, ni archivos PDF. Además de que no puede buscar en la Web o en Internet para encontrar los documentos que se pongan en venta (Copyfind).

1.3.7. EducaRed Antiplagio

Es un software gratis que debe instalarse en la computadora. Efectúa un análisis textual entre documentos para detectar si existen plagios entre ellos. Analiza las fuentes externas más frecuentes (sitios web como el “rincón de vago” o portales como Enciclonet) y fuentes internas (documentos dados de alta dentro de la aplicación). Además posee la ventaja de analizar todo tipo de documentos cuyo texto se pueda seleccionar y copiar. Como desventaja tiene que solo puede ser usado en Windows (Educared).

1.3.8. Approbo

Es un servicio online gratis, que rastrea en los millones de sitios indexados en Google y otros buscadores, y marca qué frases de un documento se encuentran también en la Web. Entre otras de sus ventajas tiene que pueden verificarse documentos de texto, hojas de cálculo, o archivos de PDF. Su principal desventaja es que sólo puede usarse desde computadoras con acceso a Internet (Approbo).

1.4. Algoritmos

La detección de plagio, tarea que no implica únicamente el análisis de textos completos para determinar si han sido escritos por un autor determinado o no, sino que busca analizar un documento o uno de sus fragmentos para intentar determinar si realmente fue escrito por el autor que reclama haberlo hecho. Existen dos vertientes principales que buscan dar solución a este problema que, debido a su naturaleza, no son capaces de ofrecer el mismo tipo de información tras el análisis realizado. El primero de ellos es el conocido como análisis intrínseco de plagio, en el que el único recurso utilizado es el texto sospechoso por sí mismo. El segundo de ellos, es la detección de plagio con referencia, donde se requiere contar con un conjunto de documentos originales con el afán de buscar el origen de los fragmentos potencialmente plagiados dentro de un texto sospechoso. En el caso del análisis intrínseco de plagio, sólo es posible hallar fragmentos que son sospechosos de ser plagiados. Por medio de la detección de plagio con referencia, es posible obtener además el origen potencial de un fragmento de texto considerado como candidato a ser un caso de plagio (Barrón Cedeño, 2008).

Para la certera detección automática de plagio, es de vital importancia seleccionar un conjunto de características del texto que sean capaces de discriminar textos plagiados de originales. Luis Alberto Barrón Cedeño (Barrón Cedeño, 2008) ha delimitado un conjunto de características que pueden ser explotadas para localizar potenciales casos de plagio, aunque las orienta al ámbito académico, su aplicación se extiende de manera directa a cualquier otro. Las características son:

1. Vocabulario utilizado: analizar el vocabulario utilizado en alguna tarea con respecto a documentos escritos previamente por el mismo estudiante. La existencia de una alta cantidad de vocabulario nuevo podría ayudar a determinar si un estudiante realmente escribió un texto o no.
2. Cambios de vocabulario: si el vocabulario utilizado en un texto cambia significativamente a través de un documento.
3. Texto incoherente: si un texto fluye de manera inconsistente o confusa.

4. Puntuación: es muy poco probable que dos autores utilicen los signos de puntuación exactamente de la misma manera.
5. Cantidad de texto común entre documentos: es poco frecuente que dos documentos escritos de manera independiente compartan grandes cantidades de texto.
6. Errores en común: resulta muy improbable que dos textos independientes tengan los mismos errores de escritura.
7. Distribución de las palabras: es poco frecuente que la distribución en el uso de las palabras a través de textos escritos independientemente sea la misma.
8. Estructura sintáctica del texto: un indicador de plagio es que dos textos compartan una estructura sintáctica común.
9. Largas secuencias de texto en común: es poco probable que dos textos independientes (incluso cuando traten el mismo tema), compartan largas secuencias de caracteres o palabras consecutivas.
10. Orden de similitud entre textos: si existe un conjunto significativo de palabras o frases comunes en dos textos, puede haber un caso de plagio.
11. Dependencia entre ciertas palabras y frases: un autor tiene preferencias sobre el uso de ciertas palabras y frases, encontrarlas en un trabajo realizado por otro, debe ser considerado sospechoso.
12. Frecuencia de palabras: es poco común que las palabras halladas en dos textos independientes sean usadas con la misma frecuencia.
13. Preferencia por el uso de sentencias cortas o largas: los autores pueden tener una marcada preferencia sobre la longitud de las sentencias: dicha longitud podría ser poco usual en compañía de otras características.
14. Legibilidad del texto: resulta improbable que dos autores compartan las mismas medidas de legibilidad, tales como los índices de Gunning, Flesch o SMOG.
15. Referencias incongruentes: la aparición de referencias en el texto que no se encuentran en la bibliografía o viceversa son disparadores de un posible caso de plagio.

Las características señaladas en esta lista, en conjunto con algunas otras, han sido explotadas por diferentes enfoques para la detección de plagio. Se ha realizado una clasificación de métodos para la detección de plagio que está conformada por tres categorías principales (Barrón Cedeño, 2008):

- La comparación exhaustiva entre documentos sospechosos y documentos de referencia.
- La definición de un fragmento de texto característico en un documento sospechoso para buscarlo en la web.
- La realización de un análisis de estilo, el cual es conocido como estilometría.

El estudio realizado en la tesis de maestría de Luis Alberto Barrón Cedeño (Barrón Cedeño, 2008) hace énfasis en que esta clasificación lleva a dividir la tarea de detección automática de plagio en los dos conjuntos que se define previamente: análisis intrínseco de plagio y detección de plagio con referencia, de este estudio nos hemos apoyado para hacer una breve explicación a continuación.

1.4.1. Análisis intrínseco de plagio

Una de las mayores dificultades en la detección de textos plagiados es la enorme cantidad de documentos originales que deben ser considerados con el objetivo de determinar si pueden ser su fuente. Por ello, algunas investigaciones se han basado en el análisis de plagio sin tomar en cuenta ningún documento original con el cual hacer alguna comparación. Éste es el principio básico del análisis intrínseco de plagio. El análisis intrínseco de plagio se basa en un hecho muy común en el ámbito académico (e incluso fuera de él): una persona es capaz de detectar que un documento es irregular (sospechoso de plagio), por el simple hecho de leerlo.

La idea principal en el análisis intrínseco de plagio es precisamente capturar el estilo y la complejidad a través de un documento sospechoso con el afán de encontrar fragmentos inusuales que sean candidatos a ser casos de plagio. En esta investigación el estilo y complejidad de un texto son medidas con base en un conjunto de parámetros que intentan medir los aspectos anteriormente señalados. Dado un documento sospechoso s, los parámetros a considerar son:

1. Promedio de clases de palabras basado en la frecuencia. Cada palabra $w \in s$ es asignada a una clase denominada $c(w)$**.** La clase asignada a cada palabra depende de su frecuencia en el documento. La palabra (o conjunto de palabras) que presente la máxima frecuencia de aparición en s es denominada $w*$ y se asocia a la clase c_0**.** El resto de palabras en s es asignado a la clase cuyo subíndice se determina por $[\log_2 (f(w*) / f(w))]$, en donde $[x]$ es la función piso. Esta medida refleja la complejidad y el tamaño del vocabulario de un documento. Se utiliza debido a que suele ser bastante estable cuando se analiza un documento original, escrito por un único autor, sin importar su longitud.
2. Longitud de las sentencias. El promedio de la longitud de sentencias suele ser relativamente uniforme a través de un documento escrito por un autor.
3. Partes de la oración. Las categorías gramaticales utilizadas hablan del estilo de escritura de un autor.

4. Número promedio de palabras de paro. El uso de determinados artículos, preposiciones y otras palabras que no aportan demasiado significado a un texto puede ser completamente diferente de un autor a otro.
5. Índice de confusión de Gunning. Esta medida ha sido diseñada para determinar qué tan comprensible es un texto escrito. El valor obtenido por dicha medida es una aproximación al número de años de educación formal que una persona requiere para comprender un texto leyéndolo una sola vez.

   Dicho índice se calcula tomando una muestra de texto de alrededor de 100 palabras por medio de la siguiente ecuación:

   $$I_G = 0.4\left(\frac{|palabras|}{|sentencias|} + 100 * \frac{|palabras\ complejas|}{|palabras|}\right) \text{ (1)}$$

   Donde $|x|$ es el número de elementos x en la muestra de texto. Se considera que una palabra compleja contiene al menos tres sílabas.

   Por ejemplo, la revista Newsweek tiene un índice IG (Newsweek) = 10 mientras que un comic tiene aproximadamente IG (comic) = 6.
6. Índice de Flesch-Kincaid. Este índice es muy parecido al anterior y de nuevo intenta calcular los años de educación necesarios para comprender un documento. Su cálculo se hace por medio de la siguiente ecuación:

   $$IFK = 1.599\lambda - 1.015\beta - 31.517 \text{ (2)}$$

   donde **λ** es el número promedio de palabras de una sílaba por cada cien palabras y **β** es la longitud promedio de las sentencias en número de palabras.
7. Índice de Dale-Chall. Este índice fue diseñado en los años cuarenta para determinar de nuevo los años de estudio necesarios para leer un texto. La fórmula para su cálculo es:

   $$I_{DC} = 0.0496\beta + 0.1579\varphi + 3.6365 \text{ (3)}$$

   donde φ es el porcentaje de “palabras difíciles” en el texto (es necesario definir previamente un vocabulario con estas palabras).
8. Función R. Esta medida propuesta por Honore intenta capturar la variedad en el vocabulario de un autor. Se calcula por medio de la siguiente ecuación:

   $$R = \frac{100\log(M)}{M^2} \text{ (4)}$$

   donde M es el número de palabras en el texto analizado.
9. Función K. Esta función es en realidad una alternativa para el cálculo de la riqueza de vocabulario obtenida por medio de la función R. Se calcula de la siguiente manera:

   $$K = \frac{10^4\left(\sum_{i=1}^{\infty} i^2 V_i - M\right)}{M^2} \text{ (5)}$$

   donde V_i es el número de palabras que aparece i veces en el texto. M tiene el mismo significado que en el punto anterior.

Estas medidas son calculadas primero considerando el texto completo del documento sospechoso, luego se calculan para cada párrafo. Con los resultados obtenidos, se realiza una comparación para buscar las variaciones que puedan ser reflejo de fragmentos que no fueron escritos por el mismo autor, es decir, candidatos a estar plagiados.

Hay un aspecto importante en este enfoque al que se debe poner especial atención: de ninguna manera el análisis intrínseco de plagio es capaz de demostrar que un fragmento de texto está plagiado. La razón es simple, dado que por su esencia este enfoque no considera algún tipo de comparación de los documentos sospechosos con respecto a documentos originales, no es posible encontrar el potencial documento original que sirvió como fuente de un fragmento plagiado.

1.4.2. Detección de plagio con referencia

Una de las primeras ideas que pueden surgir cuando se busca resolver la tarea de detección de plagio, es realizar una comparación de un texto sospechoso con un conjunto de textos originales. Éste es precisamente el principio básico de esta aproximación a la detección de plagio: la detección de plagio con referencia. Dado un corpus D conformado por un conjunto de documentos originales y un documento sospechoso s, la tarea de detección de plagio puede reducirse a realizar una comparación exhaustiva del texto en s sobre el corpus D para responder a la pregunta: ¿Existe algún fragmento $s_i \in s$ que esté incluido en algún documento de D?

Análisis a nivel de documentos

El análisis se basa en definir un conjunto compuesto por aquellas palabras que muestren un número de ocurrencias similar en cada uno de los dos documentos analizados. Si dos documentos muestran frecuencias similares de ocurrencia de un conjunto de palabras, es muy probable que se trate de distintas versiones de un mismo texto.

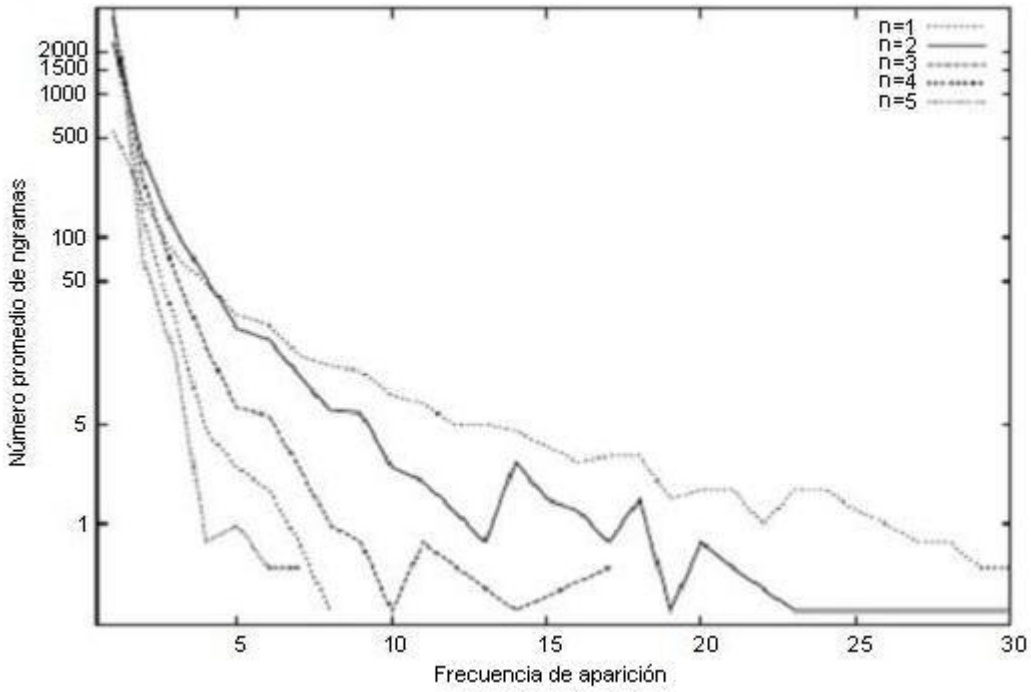
Este análisis se da principalmente a nivel de documento. Sin embargo, esto no es siempre suficiente. Para que exista un caso de plagio, no es necesario que se halle un duplicado de un documento entero, basta con que un fragmento de texto sea extraído de otro. Por ello, se han desarrollado otros enfoques que realizan un análisis a un nivel inferior, los cuales se abordan en las siguientes secciones, como ya se ha señalado, los fragmentos plagiados pueden aparecer mezclados entre texto nuevo e incluso modificados a distintos niveles.

Análisis basado en comparación de n-gramas

Para realizar una estrategia de búsqueda flexible, se ha basado la comparación de documentos en los n-gramas contenidos en ellos. La justificación para realizar esto es que dos textos independientes tienen un nivel muy bajo de n-gramas en común, siempre y cuando se considere un valor $N > 1$. De hecho, la frecuencia de aparición de n-gramas en un mismo documento suele ser muy baja.

A medida que el grado de los n-gramas considerados se eleva, la mayoría de ellos tiende a ser único. Por ende, la probabilidad de aparición de un n-grama en documentos distintos (incluso escritos por el mismo autor) es menor mientras el valor de n sea mayor. Resulta claro que la probabilidad de encontrar un bigrama en varios documentos es mucho más alta que la de encontrar un tetragrama.

Figura 1. Gráfica de distribución de n-gramas en un conjunto de textos sobre el mismo tema (n=grado del n-grama)



Fuente: (Barrón Cedeño, 2008).

Los mejores resultados se obtienen, como se muestra en la Figura 1, considerando trigramas. Así, tanto el documento sospechoso s como cada uno de los documentos de referencia $d \in D$ son codificados en forma de n-gramas para luego compararlos. Con el objetivo de determinar si el documento s puede estar plagiado del documento d se han propuesto dos medidas principales: semejanza (R) y contención (C) .

La medida de semejanza es útil cuando los conjuntos de n-gramas a comparar provienen de textos de longitud equiparable (comparaciones documento a documento, sentencia a sentencia). Considerando un documento de referencia d y uno sospechoso s, la semejanza se define por medio de la ecuación (6).

$$R(s|d) = \frac{|N(d) \cap N(s)|}{|N(d) \cup N(s)|} \quad (6)$$

donde $N(x)$ es el conjunto de n-gramas en x.

En caso de que los textos que se deseen comparar no tengan una longitud equiparable (comparaciones sentencia a documento, por ejemplo), la opción es la medida de contención, la cual se define en la ecuación (7).

$$C(s_i|d) = \frac{|N(s_i) \cap N(d)|}{|N(s_i)|} \quad (7)$$

donde s_i es alguno de los fragmentos en el documento sospechoso s.

Tanto la semejanza como la contención son valores dentro del intervalo $[0,1]$. Es necesario definir un umbral dentro de este intervalo tal que al ser superado, se considere que el texto sospechoso es un candidato a haber sido plagiado a partir del texto de referencia.

Análisis basado en la comparación a nivel de sentencia

El proceso de detección de plagio debe realizarse a nivel de sentencia (análisis de una sentencia sospechosa s_i con respecto a cada una de las sentencias de referencia d_j). Uno de los aspectos que diferencian esta investigación no sólo busca encontrar fragmentos plagiados, sino que intenta determinar de qué tipo de plagio se trata. En particular, se consideran los siguientes tipos de sentencia plagiada:

- Copia exacta: en este caso, s_i y d_j son idénticas.
- Copia con inserción de palabras: a la sentencia d_j se le han agregado palabras para generar s_i.
- Copia con eliminación de palabras: lo inverso al caso anterior, en lugar de agregar palabras, se eliminan.
- Reescritura: la sentencia si expresa exactamente lo mismo que d_j. Esto se debe al cambio de palabras por sus sinónimos o la sustitución de ciertas palabras (como preposiciones o artículos) por otras.

1.4.3. El problema del espacio de búsqueda

En las publicaciones sobre la tarea de detección de plagio a menudo se asume, que el espacio de búsqueda (que consiste en el conjunto de documentos de referencia D), es suficientemente pequeño como para que cualquier estrategia de búsqueda genere resultados aceptables en un corto tiempo. Sin embargo, en general esto no es verdad. Los corpus de referencia suelen estar compuestos por grandes cantidades de documentos originales, lo que afecta directamente al tiempo de procesamiento necesario para analizar un documento sospechoso. Por ello, se considera que antes de realizar cualquier tipo de búsqueda (como la descrita en la sección anterior), es necesario reducir tanto como sea posible el espacio de búsqueda representado por los documentos hallados en el corpus de referencia (Barrón Cedeño, 2008).

Dado el corpus de referencia D y un documento sospechoso s, los esfuerzos están ahora orientados a localizar de manera eficiente, un subconjunto D' de documentos de referencia tal que $|D'| \ll |D|$. El subconjunto D' debe contener aquellos documentos d con la más alta probabilidad de contener la fuente de los posibles fragmentos plagiados en s. Luego de esta reducción del corpus de referencia, puede realizarse un proceso exhaustivo de búsqueda de las sentencias de s sobre D' (Barrón Cedeño, 2008).

1.5. Herramientas de desarrollo

1.5.1. Metodologías de desarrollo de software

Un proceso de desarrollo de software define quién está haciendo qué, cuándo y cómo alcanzar un determinado objetivo; en la ingeniería de software el objetivo es construir un producto de software o mejorar uno existente con calidad y en el menor tiempo posible. Con el transcurso del tiempo y a medida que se revoluciona la producción de software a nivel mundial, se han ido creando metodologías y procesos que aceleran, fortalecen y mejoran la calidad de la producción. La metodología de desarrollo es un aspecto de gran importancia para el proceso de desarrollo del software, manifiesta más trascendencia si el proyecto a realizar es complejo, puesto que todo desarrollo de software es riesgoso y difícil de controlar. Si no se lleva una metodología adecuada, lo que se obtiene son clientes insatisfechos con el resultado y desarrolladores aún más insatisfechos. Las metodologías se clasifican en ágiles, flexibles y robustas.

Las metodologías robustas son más adaptables para proyectos a largo plazo, son basadas en la documentación. Cuentan con diferentes elementos de planificación (plan de desarrollo, plan de iteración, plan de calidad) con los que se controla el desarrollo del software. A través de un

esquema predefinido de escalabilidad y gestión de riesgos, se pueden reconocer previamente problemas y fallos de forma temprana y prevenirlos o corregirlos.

Las metodologías flexibles son adaptables a las otras.

Las metodologías ágiles, dan mayor valor al individuo, a la colaboración con el cliente y al desarrollo incremental del software con iteraciones muy cortas. Este enfoque muestra su efectividad en proyectos con requisitos poco definidos o muy cambiantes y cuando se exige reducir drásticamente los tiempos de desarrollo, pero manteniendo una alta calidad. Además de que se aplican bien en equipos pequeños que resuelven problemas concretos.

A continuación se analiza la Metodología Programación Extrema (XP, por sus siglas en inglés) y la Metodología OpenUp porque son unas de las principales metodologías ágiles que se utilizan actualmente, debido a que en éstas se valora más a las personas que a los procesos, se involucra al cliente y se realiza el desarrollo del producto basado en iteraciones.

Metodología OpenUp

Es una versión más ágil de lo que es el Proceso Unificado de Rational (RUP, por sus siglas en inglés), que aplica propuestas iterativas e incrementales dentro del ciclo de vida, tratando de ser manejable en relación con RUP. Plantea que se debe tener un software ya funcional o lo que es lo mismo, un proyecto ejecutable en un período de tiempo corto. Además, debe utilizar solo los procesos que sean necesarios y en este caso se necesita de personas y profesionales que sean capaces de distinguir entre lo necesario y lo que no es necesario para que el proyecto no tenga errores y con eso evitar entregar al usuario final un producto de mala calidad; aunque este tipo de método plantea además que no se deben utilizar demasiados artefactos y sobre todo que el proyecto debe acoplarse a las necesidades del usuario pudiendo ser este modificado, mejorado y extendido (Navarro, y otros, 2008).

Metodología XP

Es una metodología ágil centrada en potenciar las relaciones interpersonales como clave para el éxito en el desarrollo de software, promoviendo el trabajo en equipo, preocupándose por el aprendizaje de los desarrolladores, y propiciando un buen clima de trabajo. XP se basa en realimentación continua entre el cliente y el equipo de desarrollo, comunicación fluida entre todos los participantes, simplicidad en las soluciones implementadas y coraje para enfrentar los cambios. La principal suposición que se realiza en XP es la posibilidad de disminuir la mítica curva exponencial del costo del cambio a lo largo del proyecto, lo suficiente para que el diseño evolutivo funcione. Esto se consigue gracias a las tecnologías disponibles para ayudar en el

desarrollo de software y a la aplicación disciplinada de las siguientes prácticas (Isse Reyes, y otros, 2009).

- El juego de la planificación: Hay una comunicación frecuente entre el cliente y los programadores. El equipo técnico realiza una estimación del esfuerzo requerido para la implementación de las historias de usuario y los clientes deciden sobre el ámbito y tiempo de las entregas y de cada iteración. Entregas pequeñas: Producir rápidamente versiones del sistema que sean operativas, aunque no cuenten con toda la funcionalidad del sistema. Esta versión ya constituye un resultado de valor para el negocio. Una entrega no debería tardar más de 3 meses.
- Diseño simple: Se debe diseñar la solución más simple que pueda funcionar y ser implementada en un momento determinado del proyecto.
- Pruebas: La producción de código está dirigida por las pruebas unitarias. Éstas son establecidas por el cliente antes de escribirse el código y son ejecutadas constantemente ante cada modificación del sistema.
- Programación en parejas: Toda la producción de código debe realizarse con trabajo en parejas de programadores. Esto conlleva ventajas implícitas (menor tasa de errores, mejor diseño, mayor satisfacción de los programadores).
- Integración continua: Cada pieza de código es integrada en el sistema una vez que esté lista. Así, el sistema puede llegar a ser integrado y construido varias veces en un mismo día.

El ciclo de desarrollo consiste a grandes rasgos en los siguientes pasos:

1. El cliente define el valor del negocio a implementar.
2. El programador estima el esfuerzo necesario para su implementación.
3. El cliente selecciona qué es lo que se va a construir de acuerdo con sus prioridades y las restricciones de tiempo.
4. El programador construye ese valor de negocio.
5. Vuelve al paso uno.

El ciclo de vida ideal de XP se define en seis fases: Exploración, Planificación, Iteraciones, Producción, Mantenimiento y Muerte del Proyecto.

1.5.2. Lenguajes de programación

Perl

Perl es un lenguaje diseñado por Larry Wall en 1987. Fue creado con el fin de que fuera un lenguaje fácil de usar, completo y eficiente, en lugar de que fuera compacto y elegante. Además tiene como principales características la facilidad de uso, el soporte para diferentes tipos de programación como lo son la orientada a objetos, la estructural y la programación funcional, posee un robusto sistema de procesamiento de texto y una gran cantidad de módulos (lo que en otros lenguajes de programación como Java es llamado "bibliotecas"). También toma características de otros lenguajes de programación y su estructura está basada en bloques al estilo de C, lo que lo convierte en un lenguaje imperativo, con variables, expresiones, asignaciones, delimitación de bloques de código mediante llaves, estructuras de control y subrutinas (Pérez Hidalgo).

Perl es un lenguaje práctico, lo que quiere decir que no determina estrictamente una estructura a la hora de programar (incluye características a la hora de ser usadas, tiene tolerancia de excepciones, y utiliza heurística para resolver ambigüedades sintácticas). Sin embargo, esta teoría hace que muchas veces sea muy difícil la detección de errores (Pérez Hidalgo).

La principal desventaja de Perl se encuentra en el tiempo de ejecución de un programa, ya que es compilado cada vez que se ejecuta, por lo que puede resultar más lento que un programa similar escrito en otro lenguaje (Pérez Hidalgo).

Python

Python es un lenguaje de programación de propósito general, multiplataforma y orientado a objetos. Es un lenguaje interpretado, lo que ofrece ventajas como la rapidez de desarrollo, ya que no se necesita compilar el código fuente para poder ejecutarlo. En realidad sí se realiza una compilación, pero ésta se realiza de manera transparente para el programador. En ciertos casos, cuando se ejecuta por primera vez un código, se producen unos bytecodes que se guardan en el sistema y que sirven para acelerar la compilación implícita que realiza el intérprete cada vez que se ejecuta el mismo código. Python dispone de un intérprete por línea de comandos en el que se pueden introducir sentencias. Cada sentencia se ejecuta y produce un resultado visible, que puede ayudar a entender mejor el lenguaje y probar los resultados de la ejecución de porciones de código rápidamente. Python dispone también de muchas funciones incorporadas en el propio lenguaje, para el tratamiento de cadenas, números, archivos. (Alvarez).

Tiene eficaces estructuras de datos de alto nivel y una solución de programación orientada a objetos simple pero eficaz. Permite la declaración dinámica de variables, es decir, no es necesaria la declaración de variables, se pueden declarar variables a la vez, asignándoles algún tipo de dato. Dispone también de un gestor de memoria que se encargará de liberar la memoria de objetos no utilizados. Además de que se puede combinar con otros múltiples lenguajes de programación, como: Java (Jython), C o C++. Cuenta con una amplia biblioteca de módulos que permiten un desarrollo rápido y eficiente. Su sencillez también ayuda a que los programas escritos en este lenguaje sean muy sintéticos (Alvarez).

Realizar un programa bajo este lenguaje, seguramente costaría entre la mitad o la cuarta parte del tiempo que tardaría en desarrollar el mismo programa en C, C++ o Java esto hace que sea muy potente (Alvarez), aunque posee ciertas desventajas como (Arias Guerra, Julio del 2008):

- No es un lenguaje apropiado para la programación a bajo nivel (por ejemplo: kernels y drivers) ni tiene el control de la memoria, debido a sus características de lenguaje interpretado.

- Al ser un lenguaje interpretado, trae consigo una menor velocidad de procesamiento del código, ya que para ejecutar el código tendría que chequear los errores que puedan haber surgido dentro del código, no así en los demás lenguajes como C o C++ por solo citar dos, ya que estos sí que crean un ejecutable con códigos binarios gracias al compilador asociado a ellos.
- No es adecuado para aplicaciones que requieren una alta capacidad de cómputo, por ejemplo para el procesamiento de imágenes.

1.5.3. Entorno de desarrollo

Eclipse

Se ha decidido utilizar, para la codificación de la biblioteca el ambiente de desarrollo Eclipse, que fue fundado originalmente por la IBM y actualmente desarrollado por la fundación Eclipse, ya que el mismo es un ambiente de desarrollo de fuente abierta y típicamente ha sido usado para desarrollar entornos de desarrollo integrados. Algunas de sus características fundamentales: editor de texto, resaltado de sintaxis, compilación en tiempo real (Benavides Jorge, y otros, Julio 2007).

El Entorno Integrado de Desarrollo (IDE, por sus siglas en inglés) de Eclipse emplea módulos (plug-in) para proporcionar toda su funcionalidad. Este mecanismo de módulos es una

plataforma ligera para componentes de software, adicionalmente le permite a Eclipse extenderse usando otros lenguajes de programación como son C, C++ y Python. Se ha decidido utilizar el plug-in Pydev, ya que el mismo permite desarrollar programas en el lenguaje de programación Python (Benavides Jorge, y otros, Julio 2007).

1.6. Conclusiones Parciales

Como resultado del estudio realizado en este capítulo sobre las herramientas y algoritmos para la detección de plagio y las herramientas de desarrollo necesarias para la elaboración de la biblioteca de algoritmos de detección de plagio, se ha decidido implementar los algoritmos análisis intrínseco de plagio y detección de plagio con referencia, compuesto este último por el análisis a nivel de documento, la comparación a nivel de trigramas y el análisis a nivel de sentencias; ya que estos algoritmos satisfacen los objetivos propuestos en el trabajo.

Para la implementación de la biblioteca, se utilizará el lenguaje de programación Python, por ser un lenguaje multiplataforma, posee una sintaxis clara y elegante que se acerca al lenguaje natural y su biblioteca contiene un sinfín de módulos de utilidad.

El desarrollo de la biblioteca se realizará siguiendo los pasos propuestos por la metodología de desarrollo XP, debido a que es una metodología ágil, inicialmente creada para proyectos con cambios constantes en los requisitos y donde la aplicación se va revaluando en cortos períodos de tiempo.

CAPÍTULO II: DESARROLLO DE LA BIBLIOTECA

2.1. Introducción

Se ha realizado un estudio sobre el estado del arte del plagio, las herramientas y algoritmos que existen para detectarlo. En este capítulo se realiza el diseño, implementación y validación de la biblioteca de algoritmos, con los algoritmos anteriormente seleccionados. Para identificar si un documento tiene plagio o no aunque los algoritmos den resultados positivos, no es suficiente hay que comprobar si el documento sospechoso tiene la referencia del documento original correctamente, nuestra universidad se rige por la norma ISO-690 y es la que vamos a utilizar para hacer estas verificaciones en el documento sospechoso.

2.2. Fase de Exploración

La fase de exploración es la primera fase de la metodología XP. En esta fase el equipo de desarrollo se familiariza con las herramientas, tecnologías y prácticas que se utilizarán en el proyecto. Además se exploran las posibilidades de la arquitectura del sistema construyendo un prototipo. También se plantean a grandes rasgos las historias de usuario lo que posteriormente servirán para realizar una planificación del ciclo de vida del proyecto. Para realizar cada paso de cada una de las fases de esta metodología, nos hemos apoyado en la tesis de Roddany Olivera (Olivera, y otros, 2009).

2.2.1 Historias de Usuario

Las historias de usuario representan una breve descripción del comportamiento del sistema, o sea son la forma en XP de especificar los requisitos del sistema, emplea tecnología del cliente sin lenguaje técnico. Durante nuestra exploración hemos identificado las historias de usuario que se muestran a continuación.

Tabla 1: HU1- Identificar si un documento puede tener plagio o no.

Historia de Usuario	
Número: 1	Nombre: Identificar si un documento puede tener plagio o no.
Usuario: Cliente	Iteración Asignada: 1
Prioridad en el Negocio: Alta (Alta, Media, Baja)	Puntos Estimados: 2
Descripción: Analizar el documento sospechoso mediante los algoritmos de detección	

de plagio para descartar o no que contiene plagio.
Observaciones:

Tabla 2: HU2- Identificar si tiene la referencia del documento original.

Historia de Usuario	
Número: 2	Nombre: Identificar si tiene la referencia del documento original.
Usuario: Cliente	Iteración Asignada: 2
Prioridad en el Negocio: Alta (Alta, Media, Baja)	Puntos Estimados: 2
Descripción: Al haber encontrado que el documento sospechoso tiene una gran probabilidad de haber plagiado a un documento original localizado, verificar si el sospechoso tiene la referencia del documento original correctamente planteada.	
Observaciones: Solamente detecta las referencias escritas con la norma ISO-690.	

Tabla 3: HU3- Identificar las partes del documento que tienen plagio.

Historia de Usuario	
Número: 3	Nombre: Identificar las partes del documento que tienen plagio.
Usuario: Cliente	Iteración Asignada: 1
Prioridad en el Negocio: Alta (Alta, Media, Baja)	Puntos Estimados: 1
Descripción: Al comprobar que el documento sospechoso contiene plagio informar las partes que fueron plagiadas en oracioones, incluso puede ser el documento completo.	
Observaciones:	

Tabla 4: HU4- Reconocer de un conjunto de documentos de referencia el o los documentos originales que han sido fuente del plagiado.

Historia de Usuario	
Número: 4	Nombre: Reconocer de un conjunto de documentos de referencia el o los documentos originales que han sido fuente del plagiado.
Usuario: Cliente	Iteración Asignada:1
Prioridad en el Negocio: Alta (Alta, Media, Baja)	Puntos Estimados: 1
Descripción: Para definir si el documento sospechoso contiene plagio, se va a comparar con los documentos de referencia que van a estar almacenados para identificar cuál es el documento de referencia o los documentos que han sido	

plagiados.
Observaciones: Los documentos de referencia deben tener especificado el título y el autor.

Tabla 5: HU5- Calcular el porciento del documento que tiene fraude.

Historia de Usuario	
Número: 5	Nombre: Calcular el porciento del documento que tiene fraude.
Usuario: Cliente	Iteración Asignada: 2
Prioridad en el Negocio: (Alta, Media, Baja)	Puntos Estimados: 1
Descripción: Después de conocer las partes del documento que han sido plagiadas, informar el porciento que representa del documento sospechoso en general.	
Observaciones: El cálculo del porciento tiene en cuenta las palabras no las oraciones.	

2.3. Fase de Planificación

Durante la fase de planificación se priorizan las historias de usuarios dividiéndolas en iteraciones y los programadores estiman cuánto esfuerzo requiere cada una. Para estimar el esfuerzo se usan los puntos de estimación, los cuales son considerados como una semana de trabajo. Esta estimación incluye todo el esfuerzo asociado a la implementación de la historia de usuario, por ejemplo, las pruebas unitarias, la integración y refactorización del código, y la preparación y ejecución de las pruebas de aceptación (Olivera, y otros, 2009).

2.3.1. Estimación de esfuerzo por historias de usuario

Para el buen desarrollo de la biblioteca de algoritmos, se ha realizado una estimación para cada una de las historias de usuario identificadas, llegando a los resultados que se muestran a continuación:

Tabla 6: Estimación de esfuerzo por historias de usuario.

Historia de Usuario	**Puntos de estimación**
Identificar si un documento puede tener plagio o no.	2
Identificar si tiene la referencia del documento original.	2
Identificar las partes del documento que tienen plagio.	1
Reconocer de un conjunto de documentos de referencia el o los documentos originales que han sido fuente del plagiado.	1
Calcular el porciento del documento que tiene fraude.	1

2.3.2. Plan de iteraciones

Después de ser descritas e identificadas las historias de usuario y estimado el esfuerzo propuesto para la realización de cada una de ellas, se procede a la planificación de la etapa de implementación del sistema. Este plan especifica exactamente las historias de usuario que serán implementadas para cada iteración del sistema y las posibles fechas para estas liberaciones.

Para planificar la implementación de la biblioteca de algoritmos se decide hacer dos iteraciones:

Iteración 1: En esta iteración son implementadas las historias de usuario 1, 3, 4.

Iteración 2: En esta iteración son implementadas las historias de usuario 2, 5.

A continuación se muestra el plan de duración estimado para cada iteración con las historias de usuario en el orden en que serán implementadas

2.3.3. Plan de duración de las iteraciones.

Como parte del ciclo de vida de un proyecto utilizando la metodología XP se crea el plan de duración de cada una de las iteraciones, en este caso se hace para el único equipo de desarrollo con el cual se cuenta. Este plan tiene como finalidad mostrar la duración de cada iteración, así como el orden en que serán implementadas las historias de usuario en cada una de las mismas (Olivera, y otros, 2009).

Tabla 7: Plan de duración de las iteraciones.

Iteraciones	Orden de las Historias de Usuario a implementar	Duración total de las iteraciones
Iteración 1	Identificar si un documento puede tener plagio o no. Identificar las partes del documento que tienen plagio. Reconocer de un conjunto de documentos de referencia el o los documentos originales que han sido fuente del plagiado.	5 semanas
Iteración 2	Identificar si tiene la referencia del documento	4 semanas

	original. Calcular el porciento del documento que tiene fraude.	

2.3.4. Plan de Tareas

Este plan tiene como objetivo descomponer cada historia de usuario en tareas que serán desarrolladas por los programadores, las que proporcionan una guía para un mejor desarrollo y cumplimiento de cada historia.

Tabla 8: Plan de Tareas.

Historia de Usuario por Iteraciones	**Tareas**
Identificar si un documento puede tener plagio o no.	Implementar los algoritmos. Analizar el documento sospechoso con cada algoritmo de detección de plagio. Analizar los resultados obtenidos y determinar si tiene plagio o no.
Identificar las partes del documento que tienen plagio.	Identificar las oraciones que más probabilidades tienen de tener plagio. Mostrar las oraciones.
Reconocer de un conjunto de documentos de referencia el o los documentos originales que han sido fuente del plagiado.	Comprar el documento sospechoso con cada uno de los documentos de referencia. Determinar cuál es el documento de referencia que ha sido plagiado.
Identificar si tiene la referencia del documento original.	Buscar en el documento sospechoso la referencia del documento original que ha sido fuente del plagio.
Calcular el porciento del documento que tiene fraude.	Mostrar el porciento del documento sospechoso que contiene plagio.

Tabla 9: Tarea 1-Implementar los algoritmos.

Tarea

Número de Tarea: 1	Número de la HU: 1
Nombre de la Tarea: Implementar los algoritmos.	
Tipo de Tarea: Desarrollo	Puntos estimados:
Fecha Inicio:	Fecha Fin:
Programador Responsable: Ivette María Suárez Muñoz	
Descripción: Implementar los cuatro algoritmos de detección de plagio seleccionados agrupados en: Análisis Intrínseco de Plagio y Detección de Plagio con Referencia.	

Tabla 10: Tarea 2- Analizar el documento sospechoso con cada algoritmo de detección de plagio.

Tarea	
Número de Tarea: 2	Número de la HU: 1
Nombre de la Tarea: Analizar el documento sospechoso con cada algoritmo de detección de plagio.	
Tipo de Tarea: Desarrollo	Puntos estimados:
Fecha Inicio:	Fecha Fin:
Programador Responsable: Ivette María Suárez Muñoz	
Descripción: Ejecutar sobre el documento sospechoso los algoritmos implementados y obtener resultados.	

Tabla 11: Tarea 3- Analizar los resultados obtenidos y determinar si tiene plagio o no.

Tarea	
Número de Tarea: 3	Número de la HU: 1
Nombre de la Tarea: Analizar los resultados obtenidos y determinar si tiene plagio o no.	
Tipo de Tarea: Desarrollo	Puntos estimados:
Fecha Inicio:	Fecha Fin:
Programador Responsable: Giselle Pérez Carreras	
Descripción: Con los resultados de los algoritmos, comparar cada uno con el umbral seleccionado previamente para cada algoritmo y devolver una respuesta.	

Tabla 12: Tarea 4- Identificar las oraciones que más probabilidades tienen de tener plagio.

Tarea	
Número de Tarea: 4	Número de la HU: 2
Nombre de la Tarea: Identificar las oraciones que más probabilidades tienen de tener plagio.	
Tipo de Tarea: Desarrollo	Puntos estimados:
Fecha Inicio:	Fecha Fin:

Programador Responsable: Ivette María Suárez Muñoz
Descripción: Luego del análisis del documento sospechoso de cada algoritmo, identificar las oraciones que pueden contener plagio.

Tabla 13: Tarea 5- Mostrar las partes que tienen plagio.

Tarea	
Número de Tarea: 5	Número de la HU: 2
Nombre de la Tarea: Mostrar las oraciones.	
Tipo de Tarea: Desarrollo	Puntos estimados:
Fecha Inicio:	Fecha Fin:
Programador Responsable: Ivette María Suárez Muñoz	
Descripción: Mostrar cada oración del documento sospechoso que tiene plagio.	

Tabla 14: Tarea 6- Comprar el documento sospechoso con cada uno de los documentos de referencia.

Tarea	
Número de Tarea: 6	Número de la HU: 3
Nombre de la Tarea: Comparar el documento sospechoso con cada uno de los documentos de referencia.	
Tipo de Tarea: Desarrollo	Puntos estimados:
Fecha Inicio:	Fecha Fin:
Programador Responsable: Giselle Pérez Carreras	
Descripción: Los algoritmos de detección de plagio por referencia se basan en comparar el documento sospechoso con el original, se realizarán comparaciones con cada uno de los documentos originales que estén almacenados.	

Tabla 15: Tarea 7-Determinar cuál es el documento de referencia que ha sido plagiado.

Tarea	
Número de Tarea: 7	Número de la HU: 3
Nombre de la Tarea: Determinar cuál es el documento de referencia que ha sido plagiado.	
Tipo de Tarea: Desarrollo	Puntos estimados:
Fecha Inicio:	Fecha Fin:
Programador Responsable: Giselle Pérez Carreras	
Descripción: Al comparar el documento sospechoso con cada documento de referencia se obtendrá cuál o cuáles fueron los documentos originales plagiados, será mostrado su título.	

Tabla 16: Tarea 8- Verificar las referencias del documento sospechoso.

Tarea	
Número de Tarea: 8	Número de la HU: 4
Nombre de la Tarea: Verificar las referencias del documento sospechoso.	
Tipo de Tarea: Desarrollo	Puntos estimados:
Fecha Inicio:	Fecha Fin:
Programador Responsable: Ivette María Suárez Muñoz	
Descripción: Si el documento tiene posibilidades de contener plagio verificar si tiene referenciado el o los documentos originales.	

Tabla 17: Tarea 9- Mostrar el porciento del documento sospechoso que contiene plagio.

Tarea	
Número de Tarea: 9	Número de la HU: 5
Nombre de la Tarea: Mostrar el porciento del documento sospechoso que contiene plagio.	
Tipo de Tarea: Desarrollo	Puntos estimados:
Fecha Inicio:	Fecha Fin:
Programador Responsable: Giselle Pérez Carreras	
Descripción: Calcular y mostrar el porciento de plagio del documento.	

2.4. Fase de Producción

La metodología XP plantea que la implementación de un software debe realizarse de forma iterativa, obteniendo al culminar cada iteración un producto funcional que debe ser probado y mostrado al cliente para incrementar la visión de los desarrolladores con la opinión de éste.

2.4.1. Tarjetas CRC

Tarjetas Clase-Responsabilidad-Colaborador (CRC) se dividen en tres secciones que contienen la información del nombre de la clase, sus responsabilidades y sus colaboradores. Para la confección de estas tarjetas se definen las clases principales, luego sus responsabilidades que son sus atributos y métodos y sus colaboradores que son las demás clases que se relacionan con ella (Olivera, y otros, 2009).

Las clases principales que tenemos en nuestra biblioteca son:

1- **Analisis_Intrinseco:** es la que implementa el algoritmo análisis intrínseco de plagio
2- **Analisis_Nivel_Documento:** es la que implementa el algoritmo análisis a nivel de documento.

3- **Comparacion_Nivel_N-gramas:** es la que implementa el algoritmo análisis a nivel de n-gramas.
4- **Analisis_Nivel_Sentencias:** es la que implementa el algoritmo comparación a nivel de sentencias.
5- **Referencia:** es la que va a analizar si el documento sospechoso tiene la referencia correctamente elaborada.
6- **Biblioteca_de_Algoritmos***:* es la que va a permitir comparar el documento sospechoso con cada documento de referencia con cada uno de los algoritmos.

Tabla 18: Tarjeta CRC clase Analisis_Intrinseco

clase: Analisis_Intrinseco	
Responsabilidades:	**Colaboradores:**
- Dividir el texto en párrafos. - Calcular la variedad del documento. - Calcular la variedad de cada párrafo. - Comparar los resultados obtenidos de la complejidad del documento completo, con los de cada párrafo. - Devolver los párrafos que tienen posibilidad de tener plagio.	Biblioteca_de_Algoritmos

Tabla 19: Tarjeta CRC clase Analisis_Nivel_Documento

clase: Analisis_Nivel_Documento	
Responsabilidades:	**Colaboradores:**
- Calcular la frecuencia de cada palabra del documento sospechoso que se encuentra en el documento original. - Comparar las frecuencias de aparición de una misma palabra entre el documento original y el sospechoso. - Devolver la lista de palabras fraudulentas.	Biblioteca_de_Algoritmos

Tabla 20: Tarjeta CRC clase Comparacion_Nivel_n-gramas

clase: Comparacion_Nivel_N-gramas	
Responsabilidades:	**Colaboradores:**
- Agrupar los n-gramas en conjuntos.	Biblioteca_de_Algoritmos

- Calcular la semejanza o contención. - Comparar la semejanza o contención con el umbral determinado. - Devolver los n-gramas que han sido plagiados.	

Tabla 21: Tarjeta CRC clase Analisis_Nivel_Sentencias

clase: Analisis_Nivel_Sentencias	
Responsabilidades:	**Colaboradores:**
- Agrupar las sentencias del sospechoso en un conjunto y las del original en otro. - Devolver las sentencias que han sido plagiadas.	Biblioteca_de_Algoritmos

Tabla 22: Tarjeta CRC clase Referencia

clase: Referencia	
Responsabilidades:	**Colaboradores:**
- Buscar las referencias de documento sospechoso. - Identificar si las referencias corresponden con el documento original. - Devolver si están hechas correctamente las referencias en el documento sospechoso.	Biblioteca_de_Algoritmos

Tabla 23: Tarjeta CRC clase Biblioteca_de_Algoritmos

clase: Biblioteca_de_Algoritmos	
Responsabilidades:	**Colaboradores:**
- Mostrar si el documento sospechoso contiene plagio o no. - Mostrar el título del o los documentos que fueron plagiados. - Mostrar las partes que tienen plagio. - Mostrar el porciento del documento que contiene fraude.	Analisis_Intrinseco Analisis_Nivel_Documento Comparacion_Nivel_N-gramas Analisis_Nivel_Sentencias Referencia

2.4.2. Estándares de codificación

Un estándar de codificación completo comprende todos los aspectos de la generación de código. Debido a que la biblioteca de clases es implementada por dos programadores hemos decidido establecer un estándar para el código y así garantizar la homogeneidad y reflejar un estilo armonioso, como si un único programador hubiera escrito todo el código de una sola vez. Además, la biblioteca de clases puede estar sujeta a mejoras, por esto es necesario facilitar elementos para que otros desarrolladores entiendan el código fuente. El estilo de código utilizado se describe a continuación (Olivera, y otros, 2009):

Identación: se usan tres espacios por cada nivel de identación.

Líneas en blanco: se separan las importaciones y las definiciones de clases con dos líneas en blanco y las definiciones de métodos dentro de una misma clase se separan con una línea en blanco.

Importar: Los imports se colocan siempre en la parte superior del archivo, justo después de cualquier comentario o cadena de documentación del módulo, y antes de las variables globales y las constantes del módulo y en distintas líneas.

Espacios en blanco en expresiones y sentencias: Se coloca un espacio siempre antes y después de los siguientes operadores: asignación (=), asignación aumentada (+=, -=), comparación (==, <, >,!=, <>, <=, >=, in, not in, is, is not), booleanos (and, or, not), aritméticos(+, -, *, /).

Comentarios: Los comentarios se aplican al código que se encuentra a continuación, y se indentan al mismo nivel que dicho código. Cada línea de un comentario de bloque comienza con un # y un único espacio y la primera letra con mayúscula y al final punto.

Variables: Comienzan con minúscula y si están compuesta por más de una palabra se separan con guión bajo.

Métodos: Empiezan con minúsculas y si tienen más de una palabra, se escribe a partir de la segunda la primera letra con mayúscula.

2.4.3. Análisis Estadístico

Se ha realizado una evaluación de los algoritmos mediante la selección de veinte textos que son resúmenes de tesis: diez originales y diez con plagio de los primeros mencionados. Se han comparado usando cada uno de los algoritmos implementados por separado aún, para

determinar el umbral de cada algoritmo, o sea, desde donde el texto sospechoso contiene plagio o no. Los textos seleccionados se muestran en los anexos.

A continuación se muestra una tabla del porciento aproximado de plagio que tiene cada texto sospechoso del original.

Tabla 24: Porciento de plagio de textos sospechosos.

Texto_Original_1	Texto_Sospechoso _4	100%
Texto_Original_2	Texto_Sospechoso _5	78%
Texto_Original_3	Texto_Sospechoso _6	86%
Texto_Original_7	Texto_Sospechoso _14	73%
Texto_Original_8	Texto_Sospechoso _15	22%
Texto_Original_9	Texto_Sospechoso _16	15%
Texto_Original_10	Texto_Sospechoso _17	47%
Texto_Original_11	Texto_Sospechoso _18	59%
Texto_Original_12	Texto_Sospechoso _19	60%
Texto_Original_13	Texto_Sospechoso _20	58%

Para determinar este umbral en el caso del algoritmo análisis a nivel de documento, se obtuvo la frecuencia de aparición de cada una de las palabras del documento original y del documento sospechoso. Luego se calculó la diferencia entre ambas frecuencias y por último el promedio de las diferencias. En la figura 3 se muestra cada uno de los promedios de las frecuencias de aparición de las palabras del documento sospechoso que están en el documento original. En el caso del algoritmo de comparación a nivel de n-gramas se codificaron los documentos en secuencias de a 3 y posteriormente se calculó la semejanza o la contención según corresponda en cada caso. En el algoritmo análisis a nivel de sentencia, se calculó la suma de las oraciones que fueran iguales y similares en ambos documentos, además del conjunto reducido (las oraciones que aparecen iguales a las originales pero con palabras de menos) y el conjunto ampliado (que son las que son iguales que las oraciones originales pero con palabras de más).

Los resultados obtenidos de los algoritmos análisis a nivel de documento, comparación a nivel de trigramas y comparación a nivel de sentencia se muestran a continuación, los textos están detallados en los Anexos.

Tabla 25: Resultados del algoritmo Análisis a Nivel de Documento.

		Frecuencia
1/4.	100%	0.000750027

2/5.	78%	0.001825974
3/6.	86%	0.005288923
7/14.	73%	0.001706789
8/15.	22%	0.00768158
9/16.	15%	0.005602206
10/17.	47%	0.005134792
11/18.	59%	0.002400159
12/19.	60%	0.003733345
13/20.	58%	0.002885917
Promedio		0.003700971

Tabla 26: Resultados del algoritmo Comparación a Nivel de Trigramas

Semejanza o Contención		
1/4.	100%	0.68932039
2/5.	78%	0.46417445
3/6.	86%	0.70093458
7/14.	73%	0.72644377
8/15.	22%	0.17785235
9/16.	15%	0.10769231
10/17.	47%	0.46122449
11/18.	59%	0.43811881
12/19.	60%	0.52432432
13/20.	58%	0.49632353

Tabla 27: Resultados del algoritmo Comparación a Nivel de Sentencia.

		Total de Oraciones Sospechosas	Oraciones iguales	Oraciones Ampliadas	Oraciones Reducidas	Oraciones Similares	Oraciones con Plagio
1/4.	100%	5	2	0	1	2	5
2/5.	78%	10	3	1	0	4	8
3/6.	86%	4	0	0	1	2	3
7/14.	73%	11	8	0	0	0	8
8/15.	22%	7	2	0	0	0	2
9/16.	15%	8	2	0	0	0	2
10/17.	47%	7	3	0	1	0	4
11/18.	59%	9	5	0	0	0	5

12/19.	60%	5	4	0	0	0	4
13/20.	58%	8	4	0	0	0	4

Tabla 28: Tabla Resumen

		A nivel de documento	Comparación de 3-gramas	Sentencias
1/4.	100%	0.000750027	0.689320388	5
2/5.	78%	0.001825974	0.464174455	8
3/6.	86%	0.005288923	0.700934579	3
7/14.	73%	0.001706789	0.726443769	8
8/15.	22%	0.00768158	0.177852349	2
9/16.	15%	0.005602206	0.107692308	2
10/17.	47%	0.005134792	0.46122449	4
11/18.	59%	0.002400159	0.438118812	5
12/19.	60%	0.003733345	0.524324324	4
13/20.	58%	0.002885917	0.496323529	4
Promedio		0.003700971	0.4786409	4.5

Análisis Intrínseco

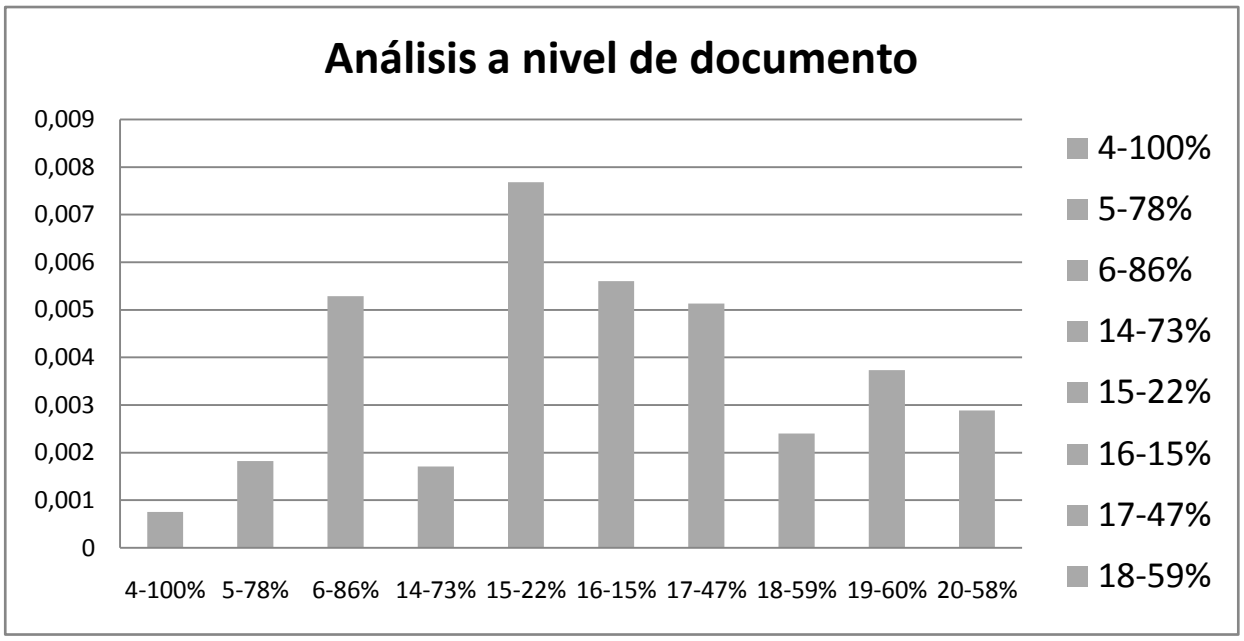
El algoritmo análisis intrínseco de plagio se ha usando para su implementación la medida propuesta por Honore, que devuelve la variedad del vocabulario de un autor por cada párrafo, luego hemos determinado que si la diferencia entre la variedad de cada párrafo y la del documento completo es mayor que 0.1 entonces ese párrafo puede contener plagio. Hemos analizado el algoritmo y es el que da resultados menos confiables por lo que hemos determinado darle prioridad 1 a la hora de determinar si un documento sospechoso tiene plagio o no.

Análisis a Nivel de Documentos

Este algoritmo como se ha explicado obtiene sus máximos resultados cuando el documento sospechoso es un duplicado del original. Aunque al evaluarlo con los textos se han obtenido buenos resultados a la hora de detectar plagio. Como se puede observar en la figura 7, a mayor diferencia de frecuencias de aparición de las palabras, generalmente corresponden los documentos que menos plagio tienen. El umbral más eficiente para el algoritmo se ha determinado que es 0.004 (el número se determinó promediando los resultados obtenidos en los experimentos realizados) se comporta en el algoritmos de la siguiente manera: si la diferencia entre la frecuencia de aparición de las palabras del documento sospecho que aparecen en el original y la frecuencia de aparición de las palabras del documento original que están en el

sospechoso es menor o igual que 0.004, entonces pasa a ser una palabra sospechosa, si las palabras sospechosas representan el 65 % del documento sospechoso en su totalidad o más, entonces se determina que es muy probable que el documento contenga plagio. Al algoritmo se le ha dado una prioridad de valor 2.

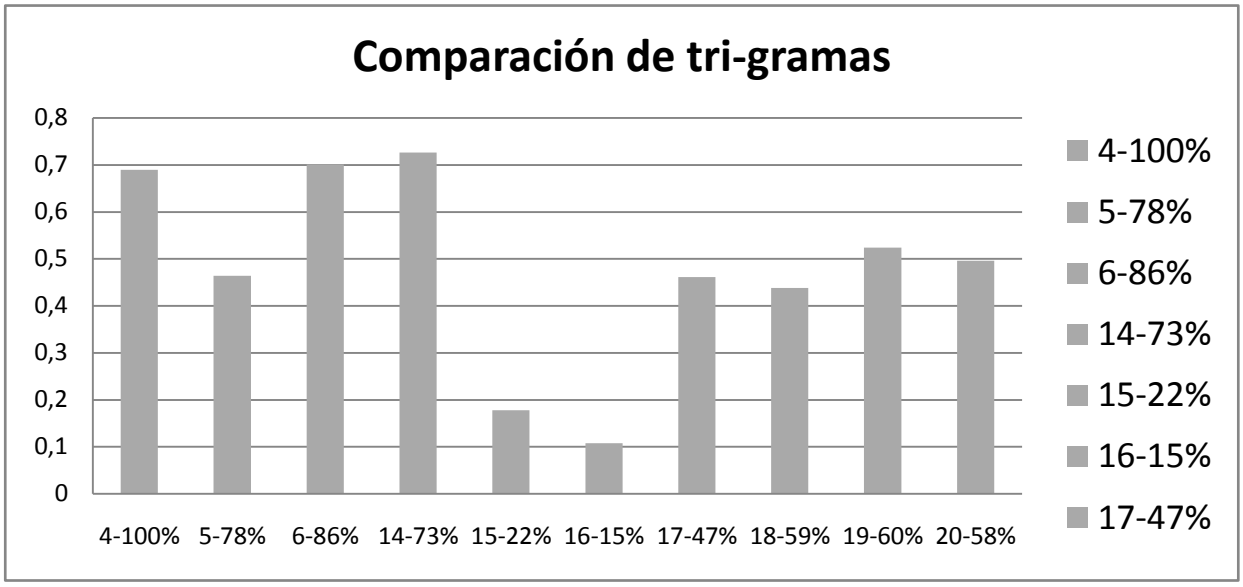
Figura 2: Gráfica algoritmo Análisis a Nivel de Documento.



Comparación a nivel de Trigramas

Este es uno de los algoritmos que mejores resultados ha arrojado. Se hicieron pruebas con el número 0.48 que es el promedio de los valores obtenidos de la contención (en caso de que las longitudes no sean equiparables) o la semejanza (en caso de que las longitudes sean equiparables) y de los experimentos realizados los resultados no fueron satisfactorios pues se va muy por encima de los casos reales de plagio, debido a que como se aprecia en la figura 8, a mayor promedio de plagio en el documento sospechoso mayor es el resultado obtenido, al hacer un promedio deja por debajo de las probabilidades de plagio a los documentos que menor porciento tienen. El objetivo es detectar de la manera más confiable posible el plagio en los documentos, por lo que se ha determinado tomar como umbral del algoritmo el número 0.25, tomando a consideración los menores resultados obtenidos, número que como se aprecia en la figura 8 implica a la mayoría de los documentos plagiados y con el cual las pruebas al algoritmo han dado buenos resultados. El algoritmo se comporta de la siguiente manera: si la semejanza o contención es mayor que 0.25, es muy probable que el documento contenga plagio. Al algoritmo se le ha dado una prioridad de valor 3.

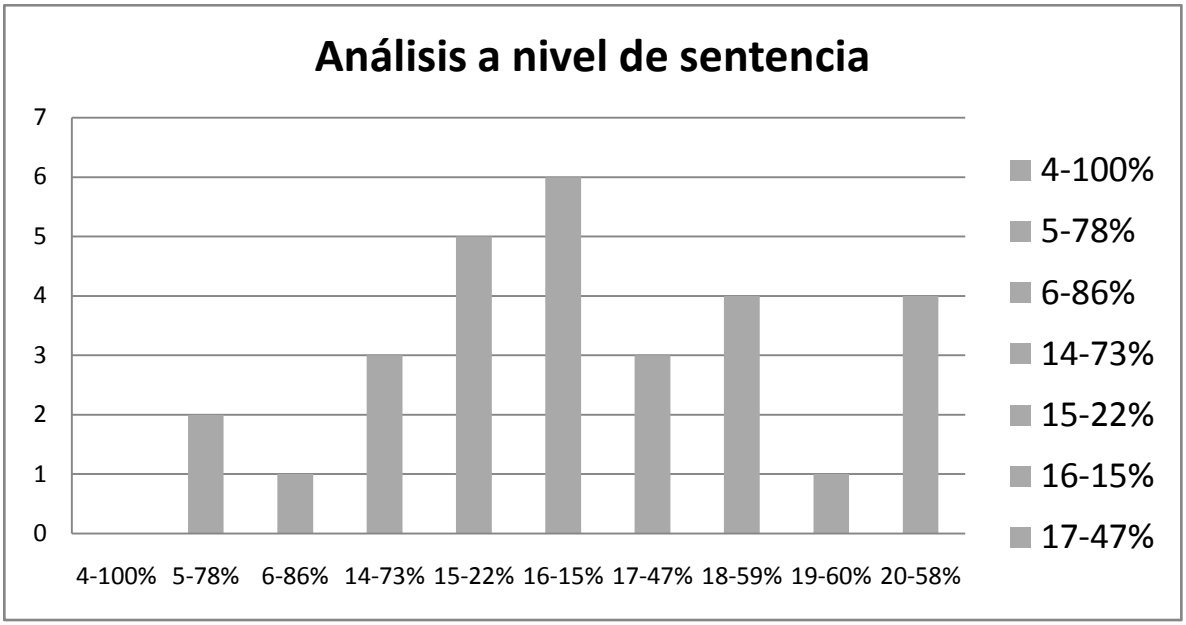
Figura 3: Gráfica algoritmo Comparación a Nivel de Trigramas.



Comparación a Nivel de Sentencias

Este algoritmo es el que mejores resultados ha obtenido, por eso le hemos dado la máxima prioridad con valor 4, se ha determinado que si las oraciones con plagio representan el 10% o más del documento sospechoso entonces el documento contiene plagio. Las oraciones con plagio se determinaron si: son iguales, son semejantes, se le han agregado palabras o se le han eliminado palabras. En la figura 9 se muestra el análisis realizado de los experimentos con este algoritmo, las barras muestran la diferencia entre las oraciones del texto sospechoso en general y las oraciones que tienen plagio, por lo que las barras de mayor longitud generalmente corresponden a los textos sospechosos que menos plagio tienen.

Figura 4: Gráfica algoritmo Comparación a Nivel de Sentencia.



Como se ha podido apreciar a cada algoritmo se le ha asignado una prioridad de acuerdo a su confiabilidad, si el valor de la prioridad es igual o mayor que 4 entonces se define finalmente que el documento tiene plagio.

2.4.4. Pruebas de Unitarias y de Aceptación

Las características del software que no pueden ser demostradas mediante pruebas simplemente no existen. Las pruebas dan la oportunidad de saber si lo implementado es lo que en realidad se tenía en mente. Se han ido haciendo pruebas constantes al código llamadas pruebas unitarias, que son pruebas de caja blanca, y al final de cada iteración se han hecho las pruebas de aceptación, son pruebas de caja negra que se hacen por cada historia de usuario de esa iteración destinadas a evaluar si se obtuvo la funcionalidad requerida, además de comprobar que dicha funcionalidad sea la esperada por el cliente (Olivera, y otros, 2009).

A lo largo de la implementación de los algoritmos, en cada clase se fueron comprobando todos los métodos, de esta manera se hace más fácil la corrección de errores. Se ha utilizado para el desarrollo de las pruebas unitarias el módulo unittest también llamado PyUnit, creado por Kent Beck, uno de los padres de la Programación Extrema (eXtreme Programming). En la medida que se fueron realizando las pruebas se fueron solucionando los errores detectados, lográndose así que cada clase de la biblioteca cumpla con las funcionalidades planificadas correctamente.

Para el desarrollo de las pruebas de aceptación, además de utilizar los módulos de los algoritmos, se han creado dos módulos más, **pruebas.py** que es el que tiene una ventana que

permite ubicar mediante la dirección al documento sospechoso, cargarlo y pasárselo a la biblioteca, además se ha creado el módulo **datos.py** que va a simular una base de datos, pues va a contener la dirección de los documentos originales con su respectivo título y autor. A continuación se muestran los resultados obtenidos.

Iteración 1

Identificar si un documento puede tener plagio o no.

Identificar las partes del documento que tienen plagio.

Reconocer de un conjunto de documentos de referencia el o los documentos originales que han sido fuente del plagiado.

Tabla 29: Caso de Prueba HU1_P1

Caso de Prueba de Aceptación	
Código: HU1_P1	Historia de Usuario: 1
Nombre: Identificar si un documento puede tener plagio o no.	
Descripción: El usuario entra la dirección del documento sospechoso y la biblioteca comprueba si contiene plagio o no y devuelve los resultados obtenidos.	
Condiciones de Ejecución: El usuario debe introducir la dirección del documento sospechoso.	
Entrada/Pasos de Ejecución: Se entra la dirección del documento y se intenta probar que lo carga bien y si muestra eficientemente los resultados de los algoritmos.	
Resultado Esperado: Los algoritmos muestren resultados bastante confiables.	
Evaluación de la Prueba: Prueba satisfactoria.	

Tabla 30: Caso de Prueba HU3_P2

Caso de Prueba de Aceptación	
Código: HU3_P2	Historia de Usuario:3
Nombre: Identificar las partes del documento que tienen plagio.	
Descripción: Cada algoritmo identifica y muestra las partes que contienen plagio.	
Condiciones de Ejecución: Los algoritmos han comprobado que el documento sospechoso contiene plagio.	
Entrada/Pasos de Ejecución: El análisis de los algoritmos ha dado que el documento contiene plagio, luego se muestran las partes del documento que contienen plagio.	
Resultado Esperado: Se muestre correctamente las partes del documento sospechoso que contiene plagio.	

Evaluación de la Prueba: Prueba satisfactoria.

Tabla 31: Caso de Prueba HU2_P3

Caso de Prueba de Aceptación	
Código: HU2_P3	Historia de Usuario: 4
Nombre: Reconocer de un conjunto de documentos de referencia el o los documentos original que ha sido fuente del plagiado.	
Descripción: El sistema debe comparar el documento sospechoso usando los algoritmos con cada documento original y definir de cuáles se ha cometido plagio mostrando el título del documento original que ha sido plagiado.	
Condiciones de Ejecución: Se ha cargado el documento sospechoso y se tiene una lista de documentos originales.	
Entrada/Pasos de Ejecución: El sistema carga el documento sospechoso, luego compara con cada uno de los originales usando los algoritmos, de cada documento original que los algoritmos devuelvan que tiene plagio se va a mostrar el título.	
Resultado Esperado: Se muestre el título del o los documentos que verdaderamente han sido plagiados.	
Evaluación de la Prueba: Prueba satisfactoria.	

Iteración 2:

Identificar si tiene la referencia del documento original.

Calcular el porciento del documento que tiene fraude.

Tabla 32: Caso de Prueba HU2_P4

Caso de Prueba de Aceptación	
Código: HU2_P4	Historia de Usuario:2
Nombre: Identificar si tiene la referencia del documento original.	
Descripción: El sistema debe revisar si el documento sospechoso tiene la referencia del documento original.	
Condiciones de Ejecución: Se haya detectado que es posible que el documento tenga plagio.	
Entrada/Pasos de Ejecución: Después que los algoritmos detectan plagio se va a verificar si tiene escrita el documento sospechoso, la referencia del documento original.	
Resultado Esperado: Identifique la referencia en el documento original correctamente	

si cumple con la norma ISO-690.
Evaluación de la Prueba: Prueba satisfactoria.

Tabla 33: Caso de Prueba HU5_P5

Caso de Prueba de Aceptación	
Código: HU5_P5	Historia de Usuario: 5
Nombre: Calcular el por ciento del documento que tiene fraude.	
Descripción: Al comprobar que existe plagio en el documento sospechoso se va a calcular el porciento que tiene fraude.	
Condiciones de Ejecución: Se haya definido que tiene plagio el documento sospechoso.	
Entrada/Pasos de Ejecución: Se comprueba que el documento sospechoso tiene plagio, luego el sistema calcula el porciento que representa la parte que tiene plagio sobre el documento completo y se muestra.	
Resultado Esperado: Se calcule el porciento de plagio correctamente.	
Evaluación de la Prueba: Prueba satisfactoria.	

2.4.5. Descripción de la biblioteca

Se han obtenido finalmente sietes módulos: **intrinseco.py**, en el que se encuentra la clase Analisis_Intrinseco que implementa el algoritmo análisis intrínseco de plagio; **documento.py**, en el que se encuentra la clase Analisis_Nivel_Documento que implementa el algoritmo análisis a nivel de documento; **ngramas.py**, en el que se encuentra la clase Comparacion_Nivel_Ngramas que implementa el algoritmo comparación a nivel de trigramas; **sentencias.py**, en el que se encuentra la clase Analisis_Nivel_Sentencia que implementa el algoritmo comparación a nivel de sentencias; **referencia.py**, que contiene la clase Referencia y detecta la referencia del documento original en el documento sospechoso; **útil.py** que va contener una serie de clases que serán usadas por las demás clases, para facilitar el uso de hilos en paralelo; por último el módulo **biblioteca.py**, en el que está la clase Biblioteca_de_Algoritmos, que une todos los algoritmos y devuelve un resultado final.

A continuación se describen las funcionalidades de las clases principales.

Análisis Intrínseco (class Analisis_Intrinseco)

1. Eliminar saltos de línea entre párrafos innecesarios.
2. Quitar signos de puntuación y las mayúsculas.
1. Obtener lista de palabras de texto sospechoso sin repetir.

2. Calcular la variedad del vocabulario del texto sospechoso mediante la fórmula:
   log10 (numero_palabras)*100/numero_palabras**2.
3. Obtener una lista de párrafos.
4. Calcular la variedad del vocabulario por cada párrafo del texto sospechoso.
5. Comparar la variedad de cada párrafo con la del texto.
6. Si la diferencia de la variedad del texto y la de cada párrafo es mayor que 0.1, se toma el párrafo como sospechoso a contener plagio.
7. Si la cantidad de párrafos sospechosos es igual o mayor que el 50% de los párrafos del texto sospechoso completo, entonces devuelve que tiene plagio.

Análisis a Nivel de Documento (class Analisis_Nivel_Documento)

1. Eliminar los saltos de línea de los dos textos.
2. Obtener dos listas de palabras sin los signos de puntuación y las mayúsculas de los textos.
3. Obtener una lista de palabras de cada texto sin repeticiones.
4. Contar cada palabra y obtener una lista de contadores de las palabras.
5. Obtener una lista con las palabras de texto sospechoso que están en el texto original.
6. Calcular la frecuencia de aparición de cada palabra de la lista obtenida en ambos textos.
7. Comparar la frecuencia de cada palabra en ambos textos.
8. Si la diferencia es menor o igual que 0.004, entonces se toma como una palabra sospechosa.
9. Si la cantidad de palabras que son tomadas como fraudulentas representan el 65% o más de la cantidad de palabras sin repetir del texto sospechoso, entonces se considera que el texto tiene plagio.

Comparación de Trigramas (class Comparacion_Nivel_Ngramas)

1. Eliminar todos los saltos de líneas de los dos textos.
2. Quitar signos de puntuación y mayúsculas.
3. Obtener una lista de palabras de cada texto.
4. De cada lista obtenida se crea un conjunto de trigramas.
5. Obtener la unión y la intercepción de cada conjunto.
6. Definir si los textos son equiparables, si el texto sospechoso representa más del 80% del original entonces son equiparables.
7. Calcular semejanza o contención si son equiparables o no.
8. Si el resultado es mayor que 0.25 entonces se toma el texto sospechoso contiene plagio.

Comparación a nivel de Sentencia (class Analisis_Nivel_Sentencia)

1. Cambiar los saltos de líneas por espacios.
2. Hacer una lista de oraciones del documento original y una del documento sospechoso.
3. Quitar los signos de puntuación y las mayúsculas.
4. Convertir cada oración en un conjunto, y crear una lista de conjuntos del texto original y una lista de conjuntos del texto sospechoso.
5. Comparar la lista de conjuntos de oraciones del documento original con la del sospechoso, buscando conjuntos iguales, ampliados, reducidos y similares, las cuales se van agregando a una lista de oraciones con plagio.
6. Si la lista de oraciones con plagio representa el 10% o más del documento sospechoso, entonces se considera que el documento tiene plagio.

Referencia (class Referencia)

1. Cambiar los saltos de líneas por espacios.
2. Quitar signos de puntuación y mayúsculas.
3. Crear una lista con los autores, que deben ser pasados por parámetro separados por coma.
4. Obtener todas las posibilidades existentes en que se haya escrito el nombre de los autores según la cantidad de palabras que lo compongan.
5. Buscar el nombre del autor en todas las posibilidades de encontrarlo existentes en todo el texto sospechoso.
6. Si lo encuentra, busca el título que debe ser pasado por parámetro, en un intervalo permisible contando la cantidad de palabras que tiene el título, cumpliendo con la norma ISO-690.

Biblioteca (class Biblioteca_de_Algoritmos)

1. Recibe por parámetros una lista que contiene el título, el o los autores y el texto plano de todos los documentos originales, además el texto sospechoso.
2. Obtener resultados de los algoritmos de la comparación del texto sospechoso con cada uno de los originales.
3. Si los algoritmos arrojan que tiene plagio y tiene la referencia pues simplemente se muestra que no tiene plagio.
4. Si no tiene la referencia, se muestra en cada iteración el título del texto original.
5. Luego se muestra finalmente las partes del documento sospechoso que contienen plagio.
6. Mostrar el porciento de plagio del texto sospechoso.

2.4.6. Complejidad de los algoritmos

Una vez que se dispone de un algoritmo que funcione correctamente, es necesario definir criterios para medir su comportamiento (rendimiento), esto está centrado fundamentalmente en la simplicidad y la utilización eficiente de los recursos. La complejidad temporal (el tiempo) de un algoritmo depende de diversos factores, como pueden ser: los datos de entrada suministrados, el lenguaje utilizado, la calidad del código generado por el compilador o la eficiencia del intérprete, la naturaleza y rapidez de las instrucciones de máquina del procesador que ejecute el programa, y la complejidad intrínseca del algoritmo. Existen dos posibles estudios sobre el tiempo de ejecución de un algoritmo (Introducción al Análisis de Algoritmos, 2009):

- Uno que proporcione una medida real (a posteriori), que consiste en medir el tiempo que se demora en ejecutar para una instancia de un problema (valores de entrada datos) y en un ordenador concreto.
- Uno que proporcione una medida teórica (a priori), que consiste en obtener una función que acote (superior o inferiormente) el tiempo de ejecución del algoritmo para cualquier instancia del problema.

El primer método empleado para medir el tiempo de ejecución de un algoritmo consiste en implementar éste en un lenguaje de programación, y medir cuánto se demora para distintos tamaños de la entrada utilizando un cronómetro, el tiempo medido de esta forma se dice que usa tiempo de reloj (Introducción al Análisis de Algoritmos, 2009)**.**

Para medir el tiempo de ejecución de los algoritmos implementados se ha usado el método de medida real a posteriori, se ha hecho un análisis de los datos que pudieran ser los más usados en la universidad, la computadora usada para las pruebas tiene las siguientes características: Pentium 4, velocidad del Microprocesador 2.40Ghz, memoria RAM 248 MB, Sistema Operativo Windows XP SP 3.

Como peor caso se ha seleccionado como texto original la tesis TD_1158_08 la cual consta de 136 páginas, longitud promedio con la que cuentan las tesis de la universidad, como texto sospechoso se ha seleccionado la misma tesis, los algoritmos arrojan 100% plagio.

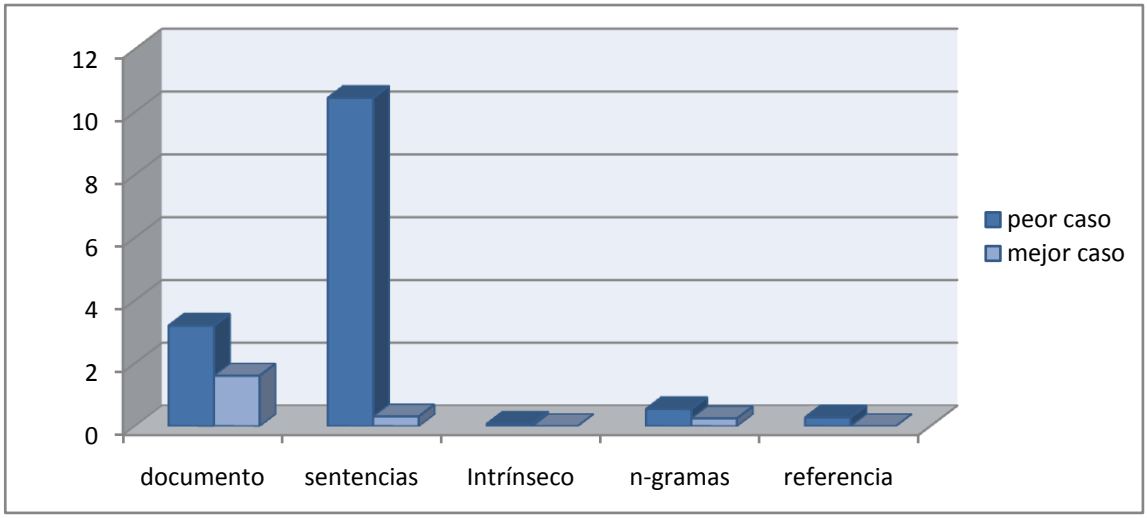
Para el mejor caso se ha seleccionado como texto original la tesis TD_1158_08 y como texto sospechoso el resumen de la tesis TD_2556_09 el cual consta de una página y los algoritmos arrojan que no tiene plagio.

A continuación los resultados de la medición del tiempo por cada algoritmo en segundos. Se ha usado la función **time** que mide el tiempo de las líneas de código seleccionadas en python, la cual devuelve los resultados en milisegundos.

Tabla 34: Medición del tiempo.

Algoritmos	Peor caso	Mejor caso
Análisis a Nivel de Documento	3.20300007	1.6099999
Análisis a Nivel de Sentencias	10.4530001	0.31200004
Análisis Intrínseco	0.07799983	0
Análisis a Nivel de Trigramas	0.54700017	0.25
Referencia	0.2809999	0

Figura 5: Gráfica de la complejidad algorítmica en cuanto al tiempo.



Como se puede observar en la gráfica, el algoritmo que más tiempo consume es el de análisis a nivel de sentencias aunque es el más confiable, y el que menos tiempo consume es el algoritmo análisis intrínseco que es el menos confiable.

2.5. Fase de Mantenimiento y Muerte

XP propone que durante la Fase de Mantenimiento pueden surgir nuevas funcionalidades por parte del cliente para el sistema. La fase de mantenimiento puede requerir la incorporación de nuevos integrantes y cambiar la estructura del equipo. Durante esta fase se han hecho los arreglos finales recomendados a la biblioteca de algoritmos.

Finalmente ya el proyecto alcanza la Fase de Muerte. Se ha decidido que la biblioteca no tiene más historias para ser incluidas y todas sus necesidades de rendimiento y confidencialidad han sido satisfactorias, por lo que no se realizarán más cambios en la arquitectura.

2.6. Conclusiones Parciales

Siguiendo los pasos propuestos por XP, guiado por las historias de usuario y partiendo de una óptima planificación y diseño, se obtuvo la biblioteca de algoritmos, la cual arroja el porciento de sospecha de plagio que puede tener un documento, aunque es al usuario, a quien le corresponde determinar si el documento fue plagiado o no.

El análisis estadístico realizado determinó por cada algoritmo, los intervalos para poder descartar o no el plagio en los documentos. Durante la implementación se realizaron pruebas unitarias y posteriormente pruebas de aceptación, las que han arrojado resultados satisfactorios, resaltando como algoritmos más confiables el análisis a nivel de sentencia y la comparación a nivel de n-gramas; y como algoritmo más eficiente en cuanto al tiempo de ejecución el análisis intrínseco de plagio y la comparación a nivel de trigramas.

CONCLUSIONES

Durante el desarrollo de este trabajo se ha realizado una amplia investigación del estado del arte de las diferentes herramientas y algoritmos para la detección de plagio en textos, existentes en la actualidad, seleccionando a partir de éstos los algoritmos que conforman la biblioteca.

Se ha realizado el diseño, la implementación y las pruebas del sistema cumpliéndose con el objetivo general propuesto, reafirmando así la utilidad y validez de emplear las tecnologías informáticas para apoyar las labores que se desarrollan en cualquier esfera del desarrollo social.

Como resultado de este trabajo la Universidad de las Ciencias Informáticas contará con una biblioteca de algoritmos para la detección de plagio, implementada en el lenguaje de programación Python, que podrá ser utilizada en una aplicación que facilitará la búsqueda de plagio en los documentos.

RECOMENDACIONES

Con la implementación de esta biblioteca de algoritmos para la detección de plagio se le dio cumplimiento a las tareas trazadas al inicio del trabajo, aunque no obstante hay que tener en cuenta una serie de recomendaciones para futuras aplicaciones de este mismo tema o el seguimiento de este producto en aras de ampliar su contenido, estas son:

- Que sea capaz de detectar referencias del documento sospechosos escritos en otras normas además de la ISO-690.
- Que posteriormente sea ampliada la biblioteca con algoritmos de detección de plagio translingües, o sea que detecte el plagio sin importar que los textos estén escritos en idiomas diferentes.
- Que sea utilizada la biblioteca de algoritmos en un sistema que compare el documento sospechoso con archivos locales y que realice búsquedas en Internet.
- Que en la aplicación para detectar el plagio que se realizará en la Universidad de las Ciencias Informáticas se utilice el algoritmo análisis basado en la comparación de trigramas, por haber resultado ser confiable y eficiente en cuanto al uso de recursos y el tiempo de ejecución.

GLOSARIO DE TÉRMINOS

Artefactos: Se entiende por artefacto cualquier obra manual realizada con un propósito o función técnica específica

Bases de datos: Una base de datos es un “almacén” que nos permite guardar grandes cantidades de información de forma organizada para que luego podamos encontrar y utilizar fácilmente.

Algoritmos: Conjunto ordenado y finito de pasos que permite hallar la solución de un problema.

Bit: Se llama a la estructura más pequeña de una computadora, podríamos definirlo también como la célula de la computadora.

Byte codes: Es un código intermedio más abstracto que el código máquina.

Concienciación: Acción y efecto de concienciar o concienciarse.

Heurística: En algunas ciencias, manera de buscar la solución de un problema mediante métodos no rigurosos, como por tanteo, reglas empíricas.

Incongruencia: Dicho o hecho faltos de sentido o de lógica.

Inflexión: Punto de una curva en que cambia de sentido su curvatura.

Inteligencia Artificial: Aquella inteligencia exhibida por artefactos creados por humanos.

Interfaz gráfica: Tipo de visualización que permite al usuario elegir comandos, iniciar programas y ver listas de archivos y otras opciones utilizando las representaciones visuales (iconos) y las listas de elementos del menú.

Iteración: Acción y efecto de iterar, repetir.

Lenguajes de programación: Los lenguajes de programación son herramientas que nos permiten crear programas.

Línea de comando: Tipo de interfaz para manipular un programa o sistema operativo con instrucciones escritas.

Métricas: Una métrica es una medida efectuada sobre algún aspecto del sistema en desarrollo o del proceso empleado que permite, previa comparación con unos valores (medidas) de referencia, obtener conclusiones sobre el aspecto medido con el fin de adoptar las decisiones necesarias.

Parámetros: Dato o factor que se toma como necesario para analizar o valorar una situación.

Paternidad del autor: Circunstancia de ser autor o creador de una obra.

Portales temáticos: Son los puentes o puertas de acceso a la información de modo temático. Son páginas que ofrecen conexiones a otras páginas.

Pruebas unitarias: Las pruebas unitarias son una de las formas que tenemos de probar pequeñas e individuales porciones de código.

Refactorización del código: Es una de las técnicas y herramientas para tratar de hacer el código más entendible, ahorrar tiempos y costos en el mantenimiento.

Scripting: Es el acto de hacer líneas de código funcionales para un programa o parte de este.

Strings: Es una cadena de caracteres, palabra, ristra de caracteres o frase.

Tokenización: Es dividir una cadena en elementos más pequeños, para poder trabajar con ellos por separado.

Tokens: Son los elementos en que el preprocesado desmenuza el código fuente.

BIBLIOGRAFÍA

Alvarez, Miguel Angel. DesarrolloWeb.com. [En línea] http://www.desarrolloWeb.com.

Approbo. *Approbo.* [En línea] http://www.approbo.com/.

Arias Guerra, Yulaine. Julio del 2008. *"Desarrollo de una Biblioteca de Estructuras de Datos Avanzadas (listas, pilas, colas, tablas hash y DCEL)".* Ciudad de la Habana : s.n., Julio del 2008.

Barrón Cedeño, Luis Alberto. 2008. *Detección automática de plagio en texto.* Valencia : s.n., 2008.

Benavides Jorge, Dailin y Fernández Rive, Yarisleidis. Julio 2007. *Biblioteca de Clases para la creación automática de resúmenes extractos. Dailin Benavides Jorge, Yarisleidis Fernández Rivera .* Ciudad de la Habana : s.n., Julio 2007.

Bordignon, Fernando R., y otros. 2005. Artículo,JEITICS 2005 - Primeras Jornadas de Educación en Informática y TICS en Argentina¨Primeras Experiencias en Detección de Plagio en el Ambiente Educativo¨. *Departamento de Ciencias e Ingeniería de la Computación.* [En línea] 2005. http://cs.uns.edu.ar/jeitics2005/Trabajos/pdf/19.pdf.

Capistrán, Marcos. 2009. *El lenguaje de programación Python.* 2009.

COMAS R., S. J. 2006. Ciber Sociedad. *¨Ciber-plagio académico: la generación de cortar y pegar¨.III Congreso On line.* [En línea] 2006. http://www.cibersociedad.net/congres2006/gts/gt.php?llegua=es&id=96.

COMAS, R.&. 2007. Cibersociedad. *Ciber-Plagio Académico. Una aproximación al estado de los conocimientos, Revista TEXTOS de la CiberSociedad,10.Temática Variada.* [En línea] 2007. http://cibersociedad.net.

CopyFind. VirusProt. *VirusProt.* [En línea] http://www.virusprot.com/Nt080823.html.

Diaz, F. Javier, Tzancoff Banchoff, Claudia M. y Rodríguez, Anahí S. 1900. *"Herramientas para la detección de plagio de software. Un caso de.* La Plata : s.n.

EducaRed. *EducaRed.* [En línea] http://www.educared.net/.

Fernández González, David. Arquitectura de base de datos para buscadores web. *Arquitectura de base de datos para buscadores web.* [En línea] http://www.telefonica.net/web2/basedatosbuscador/indexacion.htm.

Gamma, Erich. *Extreme Programming Explained.*

Hernández León, R. y Coello, G. 2002. *El Paradigma Cuantitativo De La Investigación Científica.* Ciudad de la Habana : Editorial Universitaria, 2002.

Introducción al Análisis de Algoritmos. **Departamento de Técnicas de Programación. 2009.** 2009.

Isse Reyes, Guillermo y Pérez Hernández, Héctor. 2009. *Titulo "Sistema de Gestión de la Dirección de Deporte.".* 2009.

La Torre, V. 1994. *Protección Penal del Derecho de Autor.* Valencia : Tirant lo Blanch, 1994.

León, Téllez Lisbeth y Meneses, Jiménez Alain. 2008. *¨BioSyS.Desarrollo de Portlests para la Gestión de la Información en la Simulación de Sistemas Biológicos¨.* Ciudad de la Habana : s.n., 2008.

Navarro, Bermúdez Mabel y Rodríguez, Aladana Yissel. 2008. *¨BioSyS:Implementación del Módulo de Simulacíón.¨.* Ciudad de la Habana : s.n., 2008.

Olivera Echevarría, Roddany y León Pérez, Ronny. 2009. *"Multimedia interactiva sobre las tecnologías para la Conservación de los carros".* Ciudad de la Habana : s.n., 2009.

Pérez Hidalgo, Andrey Francisco. *DESCRIPCIÓN DE LOS ASPECTOS FUNDAMENTALES DEL LENGUAJE DE PROGRAMACIÓN PERL.* San Jose, Costa Rica : s.n.

Pinto Molina, Maria. BÚSQUEDA Y RECUPERACIÓN DE INFORMACIÓN. *BÚSQUEDA Y RECUPERACIÓN DE INFORMACIÓN.* [En línea] http://www.mariapinto.es/e-coms/recu_infor.htm.

Python sp-ar:Introduccion. [En línea] http://www.swaroopch.com/notes/Python_sp-ar:Introduccion.

Turnitin. *Turnitin.* [En línea] http://www.turnitin.com.

ANEXOS

Textos Originales utilizados para la evaluación de los algoritmos.

Los textos originales escogidos para evaluar los algoritmos implementados son resúmenes de tesis desarrolladas en la UCI en años anteriores.

Texto Original 1(TO1)

Título: "Desarrollo de una Biblioteca de Estructuras de Datos Avanzadas (listas, pilas, colas, tablas hash y DCEL)".

Autor: Yulaine Arias Guerra.

Fecha: Julio del 2008.

El entorno donde coexiste el problema se da lugar en la Universidad de las Ciencias Informáticas donde es necesario utilizar estructuras de datos avanzadas para la solución de determinados problemas de la asignatura de Programación y en el desarrollo de los diferentes proyectos productivos. Pero desafortunadamente sobre este tipo de estructuras no se ha implementado aún una biblioteca que contenga agrupada una gran variedad de ellas y cuando es necesario utilizarlas, hay que implementar cada una por separado o usar las que existen aisladas.

Por tal motivo la tesis tiene como propósito el desarrollo de una Biblioteca de Estructuras de Datos Avanzadas elaborada en una plataforma de software libre, la cual utiliza UML como notación y sigue los pasos que propone el Proceso Unificado de Desarrollo de Software, utilizándose además otras herramientas complementarias.

El uso de estas estructuras de datos que se agrupan para conformar una biblioteca es una tendencia creciente del desarrollo del software con el fin de utilizarlas en la elaboración de diversas aplicaciones. Con el uso de esta biblioteca se espera optimizar el tiempo en la elaboración de los programas.

Texto Original 2 (TO2)

Título: Biblioteca genérica para la Búsqueda de Caminos.

Autor: Lecnier Sanz Friman

Fecha: 2009

La popularidad de las simulaciones de entornos virtuales en los últimos años ha crecido exponencialmente. En la parte de los videos juegos a muchas personas les gusta interactuar con personas virtuales conocidos también como agentes inteligentes. Se han creado un gran número de simuladores que ayudan a la formación de militares, al estudio del comportamiento de comunidades de especies, etc. Estas simulaciones que involucran el empleo de agentes inteligentes deben contar con un módulo encargado de manejar la Inteligencia Artificial (IA). Una de las ramas de la IA es la búsqueda de caminos. La responsabilidad de esta rama es la de mostrarle a los agentes inteligentes por donde deben desplazarse para llegar a sus destinos. El objetivo de este trabajo es la creación de una biblioteca que se encargue de la planificación y búsqueda de caminos que los agentes inteligentes puedan necesitar. Para desarrollar esta biblioteca se estudian los temas: planificador de camino, los distintos tipos de modelos cognitivos, los algoritmos de búsqueda más usados y las heurísticas que se emplean en estos algoritmos. Se exponen además las características técnicas que presentará el sistema como solución a los problemas planteados. Se analiza con profundidad el objeto de estudio, y a partir del mismo se confeccionan los distintos diagramas que posibilitarán un mayor entendimiento y proporcionarán la base para el diseño e implementación de la biblioteca.

Texto Original 3 (TO3)

Título: Biblioteca de Efectos de Post Procesamiento.

Autor: Yeisnier Domínguez Silva

Fecha: Junio 2009

En los últimos años el empleo de efectos de post procesamiento ha hecho posible el realismo gráfico presentado por los Sistemas de Realidad Virtual (SRV). Actualmente existen cientos de estos efectos, lo que conlleva a que la mayoría de las aplicaciones que contengan efectos de post procesamiento sean de gran aceptación por parte de los usuarios debido al nivel de realismo que alcanzan las mismas. A causa de esto las aplicaciones de Realidad Virtual (RV) son empleadas en diferentes ramas de la sociedad como la Medicina, Educación, Entretenimiento y la Defensa.

El presente trabajo abarca el diseño de una biblioteca de efectos de post procesamiento para que las aplicaciones de Realidad Virtual, simuladores y Video Juegos creados en la UCI tengan una mayor aceptación en el mercado actual. Para alcanzar este objetivo se analizarán y describirán los diferentes efectos de post procesamiento más usados actualmente. Se diseñará y

desarrollan algunos de estos efectos para incluir como ejemplos, pero a su vez se incluirá en la biblioteca la posibilidad de que se le adicionen nuevos efectos. De esta forma se obtendrá una biblioteca de efectos de post procesamiento de fácil acoplamiento en los diferentes proyectos de Realidad Virtual de la facultad 5 de la Universidad de la Ciencias Informáticas.

Texto Original 7 (TO7)

Título: Sistema de Inteligencia Artificial para el juego simulador de indicios aduaneros.

Autor: Dismel Echevarria Villafranca

Fecha: "junio del 2009"

Uno de los pilares en el desarrollo de juegos electrónicos es la Inteligencia Artificial (IA). Con la ayuda de IA se aumenta el grado de realismo de los juegos, lo que contribuye en gran medida al éxito de estos. Existen varios algoritmos y técnicas que se utilizan para el desarrollo de la IA, el hecho de escoger una u otra viene dado por la aplicación o el objetivo que vaya a tener el juego que se está desarrollando. El objetivo principal de esta investigación es desarrollar un sistema de IA para el juego simulador de indicios aduaneros; que intenta simular el flujo de pasajeros a través del Aeropuerto Nacional. Por lo que el sistema de IA pretende dotar a los NPCs (caracteres no personales por sus siglas en ingles) o caracteres del juego; con movimiento autónomo que necesitan estos para desplazarse por el entorno del juego. Para lograr este movimiento, a la hora del implementar el sistema se escogieron como algoritmos de movimiento los Steering Behaviors y como se pretende simular el movimiento de multitudes de personas también se hace uso de los comportamientos grupales.

Después de explicar brevemente el desarrollo de la IA en los juegos electrónicos a través de los años y algunas de las técnicas que utiliza la IA para proporcionar el realismo necesario para los juegos; se expone la propuesta para dar solución al objetivo planteado. Más adelante se muestra el diseño y la implementación de las clases que conforman el sistema de IA para el juego simulador de indicios aduaneros.

Texto Original 8 (TO8)

Título: "Multimedia interactiva sobre las tecnologías para la Conservación de los carros"

Autores: Roddany Olivera Echevarría Ronny León Pérez.

Fecha: 2009

El uso de las Nuevas Tecnologías de la Información y las Comunicaciones (NTIC) actualmente avanza impetuosamente, y sin lugar a dudas la sociedad se encuentra inmersa en la era de la información. Las NTIC se utilizan con diversos fines y uno de ellos es el desarrollo de software para resolver cualquier problemática. El presente trabajo propone desarrollar una Aplicación Informática haciendo uso de la Tecnología Multimedia, sobre las Tecnologías de Conservación de los Carros en las FAR. Nuestra Aplicación se propone recopilar y centralizar toda la información referente a estas técnicas para una mayor y mejor preparación y comprensión de estas. Con la implementación y utilización de esta Aplicación con Tecnología Multimedia se pretende hacer llegar a las FAR, un buen medio de consulta y auto preparación.

Texto Original 9 (TO9)

Título: Multimedia sobre el framework Symfony

Autores: Ainedy Domínguez López, Felipe Lastra Moreno

Fecha: junio de 2009

El desarrollo de las nuevas tecnologías de la informática y las comunicaciones ha significado un salto fundamental en la productividad de las actividades de la sociedad cubana. Estas son empleadas principalmente en el desarrollo de software, teniendo como objetivo resolver cualquier problemática. Es por ello que el presente trabajo está enmarcado en desarrollar una aplicación que brinde información sobre el framework Symfony a los estudiantes de la Universidad de las Ciencias Informáticas (UCI), principalmente a los estudiantes de la Facultad 8. Con este fin se realiza el levantamiento de los requerimientos, el análisis, diseño e implementación de la solución propuesta siguiendo la metodología RUP, modelando con ApEM-L. Para el desarrollo del producto se analizan las tendencias, tecnologías y las posibles herramientas y se realiza un estudio de factibilidad determinando entre otras cosas la cantidad de programadores y el tiempo que se emplea en la elaboración del mismo. Con este software se pretende aumentar la bibliografía existente y el interés de los estudiantes por el aprendizaje del framework Symfony, mostrando la información de una manera amena e interesante.

Texto Original 10(TO10)

Título: Subsistema de gestión de contenidos para aplicaciones con tecnología multimedia. Prima-Frame

Autores: Eric Manuel Cobas Peña. José Antonio Soto Pérez.

Fecha: Abril, 2009

En la Universidad de las Ciencias Informáticas (UCI), en el proceso de gestión de contenidos para aplicaciones con tecnología multimedia, no se usa ninguna herramienta que automatice el proceso, para ganar en tiempo y eficacia. El presente trabajo consiste en desarrollo de una herramienta libre para la gestión de los contenidos de las aplicaciones con tecnología multimedia, que agilice el desarrollo de los proyectos, disminuya el gasto de recursos humanos y mejore la calidad del producto final. El documento recoge, en un primer momento, el estudio del estado del arte y la investigación realizada. Luego detalla el proceso de desarrollo de software seguido para la creación de la solución y los aspectos técnicos de la creación del mismo.

Texto Original 11 (TO11)

Título: "Multimedia sobre los Métodos de Recubrimientos Galvánicos."

Autor: Janiel Javier Castro Rodríguez.

Fecha: Diciembre 2009.

El Centro de Enseñanza Militar "Antonio Maceo", es un instituto técnico militar para la formación y preparación de las nuevas generaciones de oficiales de las Fuerzas Armadas Revolucionarias. En el mismo se imparten diferentes materias que son de vital importancia para el desempeño futuro y profesional de sus oficiales. Dentro de éstas se encuentra la asignatura de Recuperación donde se imparten contenidos para la recuperación y reparación de piezas, como son los métodos de recubrimientos galvánicos, para su enseñanza el instituto cuenta con poca documentación, digitalizada en su mayoría aparejado al acceso limitado para la práctica del procedimiento, lo cual limita la preparación de instructores y estudiantes.

Para brindar apoyo a la asignatura, se propone como solución una aplicación con tecnología multimedia que proporcionará información sobre los métodos de recubrimientos galvánicos para erradicar los problemas anteriormente mencionados. En este trabajo se refleja el estudio realizado de las tendencias, tecnologías, herramientas a utilizar, así como un estudio de factibilidad para determinar el tiempo, costo y viabilidad de la multimedia, además del levantamiento de los requerimientos funcionales y no funcionales, el análisis, diseño e implementación de la solución propuesta siguiendo la metodología RUP (Proceso Unificado de Desarrollo de Software) utilizando UML (Lenguaje Unificado de Modelado) como soporte de la modelación apoyándose en la extensión OMMMA-L para lograr un modelado eficiente de la solución.

Para la implementación de la “Multimedia sobre los Métodos de Recubrimientos Galvánicos” se utilizó la herramienta Macromedia Flash 8, la cual cuenta con el potente lenguaje de programación orientado a objetos ActionScript 2.0 y el empleo de XML (Lenguaje de Marcas Extendido) para almacenar la información y cargarla de manera dinámica. Con este producto se obtendrá un medio eficiente y flexible que facilitará el aprendizaje de los métodos de recubrimientos galvánicos.

Texto Original 12 (TO12)

Título: Plug-in para Eclipse para la generación de elementos arquitectónicos personalizados

Autor: Héctor Luis López González

Fecha: Mayo 2009

En el presente trabajo de diploma se propone un nuevo plug-in para Eclipse denominado GEAP, plug-in que permite agilizar el desarrollo mediante la generación de código a partir de las especificaciones arquitectónicas. El plug-in consiste en varios componentes que permiten la generación de elementos arquitectónicos personalizados, e incluye además una ayuda integrada al Eclipse. El sistema pretende automatizar la creación de clases y de paquetes bien estructurados, a través de la creación de plantillas especificadas por la arquitectura. Se utilizó OpenUP como metodología por sus características ágiles y de orientación a proyectos y equipos pequeños. Como resultado de este trabajo se presenta la implementación del plug-in, así como la documentación necesaria para su uso y una validación de su utilidad en un ambiente real.

Texto Original 13 (TO13)

Título: Procedimiento de consultoría de software de gestión

Autores: Yadira Suárez Cruz Dariel Cardero Rodríguez

Fecha: 2009

Teniendo en cuenta los conflictos que pueda enfrentar la Universidad de las Ciencias Informáticas a la hora de brindar servicios de consultoría de software de gestión, se efectuó la investigación que se expone a continuación con el principal objetivo de elaborar un procedimiento que facilite la prestación de servicios de consultoría de software de gestión. Para el logro del objetivo propuesto se realizó una amplia revisión bibliográfica, lo que permitió elaborar y exponer la fundamentación teórica de la consultoría. Así mismo, se presentó el

procedimiento propuesto para la solución del problema, el cual explica de forma general qué hacer en cada fase y en qué momento deben ser aplicadas, proponiendo además algunos artefactos y el personal que debe realizarlas. El procedimiento propuesto tiene un carácter secuencial, que facilita la rectificación de defectos u omisiones encontrados, con el objetivo de lograr un cliente satisfecho. Con el propósito de obtener una valoración preliminar que permitiera, además, la mejora del procedimiento, incluso antes de su salida, se realizaron encuestas a especialistas en el tema, obteniéndose resultados importantes para su ejecución y eficiencia.

Textos Sospechosos utilizados para la evaluación de los algoritmos

Texto Sospechoso 4, tiene plagio del TO1 100%

El entorno donde coexiste el problema tiene lugar en la Universidad de las Ciencias Informáticas donde es necesario utilizar estructuras de datos avanzadas para la solución de ciertos problemas de la asignatura de Programación y en el desarrollo de los diferentes proyectos productivos. Desafortunadamente sobre este tipo de estructuras no se ha implementado aún una biblioteca que contenga una gran variedad de ellas y cuando es necesario utilizarlas, hay que implementar cada una por separado o usar las que existen aisladas.

Es por esto que la tesis tiene como objetivo el desarrollo de una biblioteca de Estructuras de Datos Avanzadas elaborada en una plataforma de software libre, en la cual se utilizará UML como notación y sigue los pasos que plantea el Proceso Unificado de Desarrollo de Software, utilizándose también otras herramientas complementarias.

El uso de estas estructuras de datos que se agrupan para conformar una biblioteca es una tendencia creciente del desarrollo del software con el fin de utilizarlas en la elaboración de diversas aplicaciones. Con el uso de esta biblioteca se espera optimizar el tiempo en la elaboración de los programas.

Texto Sospechoso 5, tiene plagio del TO2 78%

La popularidad de las simulaciones de entornos virtuales ha crecido exponencialmente en los últimos años. En el caso particular de los videos juegos, son muchas las personas a las que les agrada interactuar con personas virtuales denominados también como agentes inteligentes.

Se han desarrollado una serie de simuladores que ayudan a la formación de militares, al estudio del comportamiento de comunidades de especies, entre otras. Estas simulaciones que involucran el empleo de agentes inteligentes deben contar con un módulo encargado de manejar

la Inteligencia Artificial (IA). Una de las ramas de la IA es la búsqueda de caminos. El objetivo de esta rama es la de mostrarle a los agentes inteligentes por donde moverse para llegar a sus destinos. El propósito de este trabajo es desarrollar una biblioteca que se encargue de la planificación y búsqueda de caminos que los agentes inteligentes puedan necesitar.

Para el desarrollar de esta biblioteca se estudian temas como: planificador de camino, los distintos tipos de modelos cognitivos, los algoritmos de búsqueda más usados y las heurísticas que se emplean en estos algoritmos. Se presentan también las características técnicas que tendrá el sistema como solución a los problemas planteados. Se analizara con profundidad el objeto de estudio, y a partir del mismo se elaboraran los diferentes diagramas que posibilitarán una mayor comprensión y servirán de base para el diseño e implementación de la biblioteca.

Texto Sospechoso 6, tiene plagio del TO3 86 %

El empleo de efectos de post procesamiento ha hecho posible el realismo grafico presentado por los Sistemas de realidad Virtual (SRV). En la actualidad existen cientos de estos efectos, lo que ha traído consigo que la mayoría de las aplicaciones que contengan efectos de post procesamiento sean de gran aceptación por parte de los usuarios debido al nivel de realismo que las mismas alcanzan. Debido a esto las aplicaciones de Realidad Virtual (RV) son empleadas en diferentes ramas de la sociedad.

En el presente trabajo se pretende realizar un biblioteca de efectos de post procesamiento para que las aplicaciones de Realidad Virtual, simuladores y vides Juegos creados en la UCI tengan una mayor aceptación en el mercado actual.

Texto Sospechoso 14, tiene plagio del TO7 73%

Uno de los pilares en el desarrollo de juegos electrónicos es la Inteligencia Artificial (IA). Con la ayuda de IA se aumenta el grado de realismo de los juegos, lo que contribuye en gran medida al éxito de estos. Existen varios algoritmos y técnicas que se utilizan para el desarrollo de la IA, el hecho de escoger una u otra viene dado por la aplicación o el objetivo que vaya a tener el juego que se está desarrollando. El objetivo principal de esta investigación es desarrollar un sistema de IA para el juego simulador de indicios aduaneros; que intenta simular el flujo de pasajeros a través del Aeropuerto Nacional. Por lo que el sistema de IA pretende dotar a los NPCs (caracteres no personales por sus siglas en ingles) o caracteres del juego; con movimiento autónomo que necesitan estos para desplazarse por el entorno del juego. Para lograr este movimiento, a la hora del implementar el sistema se escogieron como algoritmos de movimiento

los Steering Behaviors y como se pretende simular el movimiento de multitudes de personas también se hace uso de los comportamientos grupales.

Después de explicar brevemente el desarrollo de la IA en los juegos electrónicos a través de los años y algunas de las técnicas que utiliza la IA para proporcionar el realismo necesario para los juegos; se expone la propuesta para dar solución al objetivo planteado. Más adelante se muestra el diseño y la implementación de las clases que conforman el sistema de IA para el juego simulador de indicios aduaneros.

Para lograrlo se analiza el flujo actual de los procesos presentes en la conformación de los Juegos Mella, los cuales poseen un grupo de características que permiten incluirlos dentro de los denominados eventos deportivos múltiples. A partir del análisis anterior se identifican los procesos candidatos a automatizar para finalmente conformar la propuesta de solución que garantice obtener el resultado esperado. Dicha propuesta se basa en la elaboración de un sistema informático desarrollado utilizando las denominadas tecnologías Web, cuyo título será: “Sistema para la Gestión de la Información de los Juegos Deportivos Interfacultades”.

Texto Sospechoso 15, tiene plagio del TO8 22%

El uso de las Nuevas Tecnologías de la Información y las Comunicaciones (NTIC) actualmente avanza impetuosamente, y sin lugar a dudas la sociedad se encuentra inmersa en la era de la información. Las NTIC se utilizan con diversos fines y uno de ellos es el desarrollo de software para resolver cualquier problemática.

En el presente trabajo de diploma se propone una multimedia interactiva para ser utilizada como material de apoyo para elevar los conocimientos de profesores, estudiantes y demás interesados en la escuela InterArmas Antonio Maceo, sobre Tecnologías existentes para la Diagnosis de Carros.

En el documento se recogen los resultados de un estudio realizado del estado del arte del desarrollo de software con tecnología multimedia, de los lenguajes de modelado, metodologías, herramientas y lenguajes de programación que se pueden utilizar para la elaboración del producto. Luego del análisis exhaustivo se determinó hacer uso de la metodología de desarrollo RUP y el lenguaje de modelado ApEM-L para llevar a cabo el levantamiento de requisitos, el análisis y diseño. El sistema fue implementado haciendo uso de la herramienta Macromedia Flash 8, el lenguaje de programación

ActionScript en su versión 2.0 y XML para el almacenamiento de la información y para cargar dinámicamente los contenidos.

Esta multimedia por su interactividad, resulta un material de estudio más atractivo y ameno que aquellos a los que los estudiantes y profesores de la EIFAR tienen alcance hasta este momento.

Texto Sospechoso 16, tiene plagio del TO9 15%

El desarrollo de las nuevas tecnologías de la informática y las comunicaciones ha significado un salto fundamental en la productividad de las actividades de la sociedad cubana. Estas son empleadas principalmente en el desarrollo de software, teniendo como objetivo resolver cualquier problemática.

Las Casas de Protocolo del Consejo de Estado de la República de Cuba tienen como principal objetivo brindar hospedaje a personalidades de Cuba y el resto del mundo, ofreciendo una atención de excelencia. Actualmente cuentan con una intranet que brinda información acerca de la institución y sus servicios. Dicha intranet no representa una solución acorde a los intereses y objetivos del lugar, ni logra estar a la altura de sus huéspedes, es por ello que se concibió una nueva propuesta de diseño a partir de las no conformidades requeridas. El presente trabajo de diploma tiene el propósito de diseñar una nueva intranet para las Residencias de Protocolo con el objetivo de satisfacer los requerimientos actuales en correspondencia con el nivel que se desea alcanzar en el lugar. Lo anterior deriva primeramente en la obtención de un mejor diseño gráfico que se identifique con sus instalaciones y resulte atractivo y práctico para sus huéspedes, y en segundo lugar ampliar y lograr una mejor ubicación de la información. Para cumplir con las expectativas propuestas de forma más eficiente se decidió hacer uso de un Sistema de Administración de Contenidos (CMS) que junto a las herramientas y tecnologías adecuadas permitieron modelar una aplicación web idónea a las Casas de Protocolo del Consejo de Estado de la República de Cuba sirviendo como guía y punto de partida para la implementación de un producto confiable y eficiente.

Texto Sospechoso 17, tiene plagio del TO10 47%

En la Universidad de las Ciencias Informáticas (UCI), en el proceso de gestión de contenidos para aplicaciones con tecnología multimedia, no se usa ninguna herramienta que automatice el proceso, para ganar en tiempo y eficacia. El presente trabajo consiste en desarrollo de una herramienta libre para la gestión de los contenidos de las aplicaciones con tecnología multimedia, que agilice el desarrollo de los proyectos, disminuya el gasto de recursos humanos y mejore la calidad del producto final. El documento recoge, en un primer momento, el estudio del estado del arte y la investigación realizada. Luego detalla el proceso de desarrollo de software seguido para la creación de la solución y los aspectos técnicos de la creación del mismo.

Entre los principales flujos de trabajo que se realizan en la Dirección Central de Calidad se encuentra el de comprobar la calidad de los productos software que se crean tanto en la Universidad de las Ciencias Informáticas (UCI), como en otras Empresas del país; al mismo se le denomina Proceso de Liberación de Productos Software (PLPS), actualmente este proceso se lleva a cabo de forma manual causando numerosos inconvenientes en la gestión del mismo. Este trabajo pretende modelar un Sistema Informático que solucione los problemas relacionados con la planificación, gestión y control del cúmulo de actividades, recursos e información, necesarios para garantizar el Proceso de Liberación de todos los productos software. Para alcanzar este objetivo se utilizó la metodología de desarrollo RUP, el lenguaje de modelado UML, y la herramienta CASE Visual Paradigm.

Texto Sospechoso 18, tiene plagio del TO11 59%

En el presente trabajo de diploma se abordarán los aspectos técnico-tácticos del aprendizaje del futsal para los estudiantes de la Universidad de las Ciencias Informáticas (UCI) a través de la realización de un producto multimedia interactivo. Para ello se plantea el problema a resolver como parte de la metodología de la investigación utilizada. Presenta el estado del arte actual que enmarca el desarrollo de dicho producto, así como las tendencias actuales y las tecnologías a considerar y se realiza un estudio de la factibilidad para la construcción de la aplicación. Este producto permite el aumento y la centralización de la información referente a los aspectos técnico-tácticos del futsal. Con el desarrollo de este trabajo se complementará y posibilitará el aumento del rendimiento técnico-táctico de los estudiantes de la UCI.

Para brindar apoyo a la asignatura, se propone como solución una aplicación con tecnología multimedia que proporcionará información sobre los métodos de recubrimientos galvánicos para erradicar los problemas anteriormente mencionados. En este trabajo se refleja el estudio realizado de las tendencias, tecnologías, herramientas a utilizar, así como un estudio de factibilidad para determinar el tiempo, costo y viabilidad de la multimedia, además del levantamiento de los requerimientos funcionales y no funcionales, el análisis, diseño e implementación de la solución propuesta siguiendo la metodología RUP (Proceso Unificado de Desarrollo de Software) utilizando UML (Lenguaje Unificado de Modelado) como soporte de la modelación apoyándose en la extensión OMMMA-L para lograr un modelado eficiente de la solución.

Para la implementación de la “Multimedia sobre los Métodos de Recubrimientos Galvánicos” se utilizó la herramienta Macromedia Flash 8, la cual cuenta con el potente lenguaje de programación orientado a objetos ActionScript 2.0 y el empleo de XML (Lenguaje de Marcas Extendido) para almacenar la información y cargarla de manera dinámica. Con este producto se

obtendrá un medio eficiente y flexible que facilitará el aprendizaje de los métodos de recubrimientos galvánicos.

Texto Sospechoso 19, tiene plagio del TO12 60%

En el presente trabajo de diploma se propone un nuevo plug-in para Eclipse denominado GEAP, plug-in que permite agilizar el desarrollo mediante la generación de código a partir de las especificaciones arquitectónicas. El plug-in consiste en varios componentes que permiten la generación de elementos arquitectónicos personalizados, e incluye además una ayuda integrada al Eclipse. El sistema pretende automatizar la creación de clases y de paquetes bien estructurados, a través de la creación de plantillas especificadas por la arquitectura. Se utilizó OpenUP como metodología por sus características ágiles y de orientación a proyectos y equipos pequeños.

Como resultado de esta investigación se presenta un prototipo completamente funcional del software que debe servir de base para profundizar en el funcionamiento de los Sistemas de Recuperación de Información, para incentivar la técnica de trabajo colaborativo en los proyectos de software libre de la Universidad y la posibilidad de poseer una herramienta operacional para acceder a la información web disponible en la universidad.

Texto Sospechoso 20, tiene plagio del TO13 58%

Este trabajo contiene la investigación y desarrollo de una aplicación web que gestione toda la información referente a la Dirección de Deporte en la Universidad de Ciencias Informáticas. El mismo pretende facilitar el trabajo con una eficiente informatización de los procesos existentes en los 3 departamentos con que esta constituido esta dirección, obteniendo un mejor desempeño en la misma.

Teniendo en cuenta los conflictos que pueda enfrentar la Universidad de las Ciencias Informáticas a la hora de brindar servicios de consultoría de software de gestión, se efectuó la investigación que se expone a continuación con el principal objetivo de elaborar un procedimiento que facilite la prestación de servicios de consultoría de software de gestión. Para el logro del objetivo propuesto se realizó una amplia revisión bibliográfica, lo que permitió elaborar y exponer la fundamentación teórica de la consultoría. Así mismo, se presentó el procedimiento propuesto para la solución del problema, el cual explica de forma general qué hacer en cada fase y en qué momento deben ser aplicadas, proponiendo además algunos artefactos y el personal que debe realizarlas. El procedimiento propuesto tiene un carácter

secuencial, que facilita la rectificación de defectos u omisiones encontrados, con el objetivo de lograr un cliente satisfecho.

El trabajo incluye un estudio del estado del arte, las herramientas utilizadas, características del sistema, planificación, implementación y la estrategia de pruebas que se llevo a cabo. Además un conjunto de conclusiones y recomendaciones que consideramos de gran importancia para el tema tratado.