Cooperativa CubaSWL

Cooperativa CubaSWL

Historias de Usuario

ToNgueLP, editor de corpus de textos para tareas NLP

Producto: ToNgueLP

Versión: 1.0

Autor: Ing. Abel Meneses Abad

Reglas de Confidencialidad

Clasificación: Software de licencia dual libre con fines de investigación, y comercial para aplicaciones a ámbitos con fines de lucro, factible de exportar por las partes desarrolladoras, de carácter nacional.

Este documento contiene información propietaria de **Abel Meneses Abad**, y es emitido confidencialmente para orientar al equipo de desarrollo de ToNgueLP.

El que recibe el documento asume la custodia y control, comprometiéndose a no reproducir, divulgar, difundir o de cualquier manera hacer de conocimientos público su contenido, excepto para cumplir el propósito para el cual se ha generado.

Estas reglas son aplicables a las 8 áginas de este documento.

Índice de contenido

Gestión del Proceso de Desarrollo

Flujo de Comunicación

- @ a Leonel con apuntes sobre el porqué el corpus tendrá que ser XML y el README de PAP donde explican todos los componentes del corpus. 10/5/14
- @ a Leonel con resumen de lo trabajado e indicaciones para lo demás. 21/7/14
- @ a Leonel con Parser 21/08/14 **0.5h**.
- @ a Leonel con los avances del proyecto. 21/08/14 **0.8h**.

Elaborado por el Ing Abel Meneses Abad
Basado en las planillas originales de la UCI, Copylefth Creative Common Share Alike Non-Comercial Use

Flujo de Configuración

R3.1_01 Historia de Usuario “*Configuración Base de ToNgueLP*” OK

Historia de Usuario	
Número: *R3.1-7*	**Nombre Historia de Usuario:** *Configuración Base de ToNgueLP*
Modificación de Historia de Usuario Número:	
Usuario: *Sistema*	**Iteración Asignada:** *1*
Programador responsable: *Abel Meneses Abad como Gerente y Gestor de Configuraciones*	
Prioridad en Negocio: *Baja*	**Puntos Estimados:** *2*
Riesgo en Desarrollo: *Bajo*	**Puntos Reales:** *2*
Descripción: *Elaborar el sistema de carpetas base del sistema. Fundamentalmente para la generación automática de la documentación y los tests.*	
Observaciones: • *Se utilizó la documentación de Sphinx y además las anotaciones personales de Abel.* • *Se logró configurar la carpeta para que en ~/ToNgueLP/html se genere la documentación y que los links de la documentación tanto dinámicos como estáticos funcionen al mover la carpeta a alguien que la quiera.* • *Se instaló Kate, editor de texto con soporte para HTML para generar una página web de los XML donde se vea la estructura del mismo y que pueda ser leído desde la documentación en el navegador web sin necesidad de otra herramienta.* *Tiempo de documentación: 10/05/14 1h.*	
Prototipo de interfase: *[Imagen de cada una de las interfaces relacionadas con la HU.]*	

Actividades de documentación:

- Configurar la carpeta de trabajo con Sphinx. **16h**
  - Diseñar y crear el árbol de carpetas del proyecto. Usando la que ya tenía de Sunshine, y haciendo configuraciones nuevas para este proyecto.
- LRP
  - Creando la LRP inicial. Utilizando la de Sunshine. Haciendo modificaciones básicas para que se parezca a lo más avanzado de mi experiencia con Sunshine, y mejorando algunos aspectos. **2h**.
  - Haciendo anotaciones antiguas de la LRP escritas en la libreta ahora en el digital. **2h**.
  - Adicionando funcionalidades escritas en hojas. 15/05/14 **2h**.
  - Acondicionando el documento con su índice, las secciones pertinentes como “Funcionalidades de los Módulos”, enlaces cruzados para mejorar la usabilidad del documento. 20/08/14 **2h**. 21/08/14 **1h**.
  - Definición de las funcionalidades de la Vista de Diccionarios, Definición de las funcionalidades de la Capa de Control. 21/08/14 **1.5h**.
  - 22/08/14 20:03:53 Revisiando y redefiniendo las funcionalidades de la Vista Principal. 22/08/14 21:31 **1.5h**.
- HU
  - Acondicionando el documento con su índice, las secciones pertinentes como “Flujo de Diseño”, enlaces cruzados para mejorar la usabilidad del documento. 20/08/14 **1h**. 21/08/14 **1h**.
  - HU-R1.5_01, HU-R1.5_02 -> Arreglada también la HU-R3.1-7. 19/08/14 **3h**.
  - Documentar la R8.1_01 Historia de Usuario “*Parser ToNgueLP-plagiarism-corpus-parser*” 21/08/14 1**.5h**.
  - 07/09/14 Documentando actividades del diseño de los diagramas de estado de la vista principal.
- Documentando actividades de diseño de las estructuras de los XMLs que responden a las funcionalidades del módulo 4. 22/08/14 15:11:02 - 16:05:41 **1h**.
- 29/08/14 14:11:59 Documentar el *Proceso de Normalización de Textos* provenientes de PDF → TXT.

Ir al Índice

Elaborado por el Ing Abel Meneses Abad
Basado en las planillas originales de la UCI, Copylefth Creative Common Share Alike Non-Comercial Use

15:12:41 **1h**.

- 29/08/14 15:13:12 Documentar “P*rocesos de la Vista Principal* “ 30/08/14 16:12. **1h**.
- 1/9/14 Documentar diagrama de estados de la Vista Principal
- 3/9/14 Documentar Arq_de_Informacion v0.3. Todos los elementos de la vista principal definidos y descritos. 04/09/14 **5h**.
- 07/09/14 Haciendo salvas de las cosas del doctorado en la memoria **0.5h**.
- Actualizando ToNgueLP.vue con sus colores actuales de los documentos.
-
- Documentar v0.1 del parser dentro de la documentación Sphinx. [P]
-

Flujo de Diseño

Diseño del XML “algorithmXXX-plag-report.xml”.

- *05/5/14* Documentando el suspXXX-srcYYY-plag-report.xml [1]
- Haciendo un ejemplo susp01243-src00225-plag-report.xml
- *05/5/14* Documentando el suspXXX-srcYYY-plag-report.xml
- . Haciendo un ejemplo susp01243-src00225-plag-report.xml
- 08/05/14 8:30AM - 14:42:15Estudiando XML para poder entender como hacer esto. Es mi primera vez con XML . Terminado el suspXXX-plag-report.xml.
- Estudiar el código del PAN para ver los parsers XMLs **6h**.
- *10/5/14* Revisando el corpus P4P. Objetivo: revisar el formato, y sí es o no .xml?
- *18/6/14* Generar html de suspXXX-plag-report.xml y agregar a la documentación.
-

Diseño del XML “ToNgueLP-plag-cases-corpus.xml”.

-
- *21/7/14* - * Estudiar P4P para terminar ToNgueLPyyy-plag-cases-corpus.XML
- - Escribir a Montse Nofre para dudas con P4P. [OK, contestó al día siguiente remitiéndome a Vila]
- *22/7/14* - @ a Vila con las dudas más explicadas y un PDF de la herramienta para manejar los corpus.
- - @ a Montse Nofre de agradecimiento.
- *25/7/14* - Estudiar P4P para terminar ToNgueLPyyy-plag-cases-corpus.XML -> Recordé los DTD de Meuschke de PubMed y fui a ver si había alguna idea.
- - Revisé superficialmente los .ent y XML de JAST Publishing <resultado> Son bastante complejos, a pesar de que en el sitio dicen que no son tan extensos como en el sitio oficial del estandar XML para documentos. => quise ver como implementaba ParCit este asunto.
- Creando carpeta de CbPD con los fuentes de Parcit, y además las URLs descargadas + los DTD del JAST Publishing.
- => Estudiando ParCit <resultado> Es algo complejo de instalar, dejo como nota los principales ficheros a leer luego. => volví al Chapter 15 del "How to Program with exercises" para ver la definición de los DTD, esto debe servirme para comprobar si el XML contiene la estructura correcta que el parser tiene implementada. <nota> Importante el fichero de como instalar y luego como usarlo.
- => Estudiando Chapter 15 del "How to Program with exercises" (cont) <resultado> Llegué hasta el final del capítulo. <tarea> Revisar el epígrafe de DTD cuando vaya a construir el validador del XML.
- Revisar el XML que se genera en la salida de los algoritmos del PAN. <resultado> Todas las etiquetas de aquí ya fueron cubiertas en el 'algorithmXXX-plag-report.xml'.
- Revisar el XML 'EScorpusYYY-plag-cases-corpus' vs P4P para ver si todas las etiquetas están cubiertas. -> Generado la primera versión Beta del corpus.

1 El nombre de los XMLs, sobre todo el reporte de plagio, cambiaron varias veces en la medida en que fui ordenando como ocurriría el proceso de generación de estos. Los nombres correctos son los que aparecen en la documentación en Sphinx.

Elaborado por el Ing Abel Meneses Abad
Basado en las planillas originales de la UCI, Copylefth Creative Common Share Alike Non-Comercial Use

- * No se usará la etiqueta <scope> pues aún no se entiende si esto hace referencia ideas semánticas, oraciones, chunks, u otro. <?>
- * Se usará el atributo 'projection' para declarar el efecto del fenómeno (Ej: adition) lingüístico en más de una idea sintácticamente independiente.
- * <fragmento> se cambió por <snippet> en todos los contextos.
- * el atributo 'source_description' se mantendrá para mapear el caso con otra BD.
- * Sustancialmente se cambian los atributos de <snippet> para lograr referenciar 'pares de fragmentos'(susp y src) y no fragmentos de un solo documento. Por lo tanto se convertirá en un elemento vacío, y no contendrá texto como en P4P. (esto contradice la estrategia usada en el 'algorithmXXX-plag-report.xml' de autocontener los textos sin necesidad de recurrir a los .txt susp y src, pero es solo una prueba hay que ver que tan eficiente es el acceso a texto si al final ToNgueLP visualizará uno a uno y contendrá algunos en memoria, y qué crea más problemas a la hora de programar el parser, o sea cual obliga a programar métodos más largos o complejos) <? >¿Autocontener los textos en los XMLs ahorrará tiempo y complejiza el XML o en definitiva no ahorra mucho tiempo, ni hace complejo el XML?</?>
- * Eliminados los demás tags que estaban puestos del 'algorithmXXX-plag-report.xml'., por el momento no se usarán <?>¿Harán falta estos tags en el futuro? </?>
- Generado la primera versión Beta del corpus.

- Hacer v0.1 del corpus.

Diseño del Proceso de Normalización de Textos provenientes del PDF →TXT.

- 26/8/14 Revisando el código del replace 1. Generando la documentación.

Diseño del Diagrama de Estados Vista Principal ToNgueLP.

- 01/09/14 Diseñando en papel el diagrama de estado. Verificando contra la v0.2 de la Vista Principal 04/09/14 **8h**.
- 04/09/14 10:49:38 Diseñando nuevos componentes de las “Barras de Herramientas” y “Barra de Edición de Casos”. 18:29:31 **1h**.
- 05/09/14 09:48:01 Estudiando en libro “*UML en 24 horas*”: Qué es un diagrama de estados?, para valorar si es correcto el diagrama que hice.
  - En el libro “UML y patrones”, pág. 381, recomienda el uso del diagrama de Estados para representar la inactivación de los widgets.11:38:51 **2h**.
- 05/09/14 11:39:11 Diseñando en digital Diagrama de Estados Vista Principal ToNgueLP. URL: ../ToNgueLP/doc/01_Ingenieria/1.2_Arquitectura_y_Design/Diagrams/Diagrama_de_Estados_Vista_Principal_ToNgueLP-Corpus_Process.dia 13:35:34 06/09/14 10:30 – 12:30 & 14:00 – 19:35. Total **9.5h**.
- 07/09/14 14:10 Diseñando en digital Diagrama de Estados Vista Principal ToNgueLP. URL: ../ToNgueLP/doc/01_Ingenieria/1.2_Arquitectura_y_Design/Diagrams/Diagrama_de_Estados_Vista_Principal_ToNgueLP-Case_Process.dia 22:33:35 Total **7h**.

R1.5_01 Historia de Usuario “*Prototipo Vista Principal*” OK

		Historia de Usuario
Número: *R1.5_01*	**Nombre Historia de Usuario:** Prototipo *Vista Principal*	
Modificación de Historia de Usuario Número:		
Usuario: *Lingüistas*		**Iteración Asignada:** *1*
Programador responsable: *Abel Meneses Abad como Arquitecto de Información*		
Prioridad en Negocio: *Muy-Alto*		**Puntos Estimados:** *1*
Riesgo en Desarrollo: *Bajo*		**Puntos Reales:** *1*
Descripción: *Diseñar el prototipo de la vista principal.*		
Observaciones: *La lista de los elementos a tener en cuenta para elaborar esta interfaz se encuentra en la versión de la*		

Elaborado por el Ing Abel Meneses Abad
Basado en las planillas originales de la UCI, Copylefth Creative Common Share Alike Non-Comercial Use

Arquitectura de Información del 17/7.
Tiempo de desarrollo: del 24/6/14 al 29/6/14
Desarrollado con la herramienta libre Evolus-Pencil.

Prototipo de interfase:

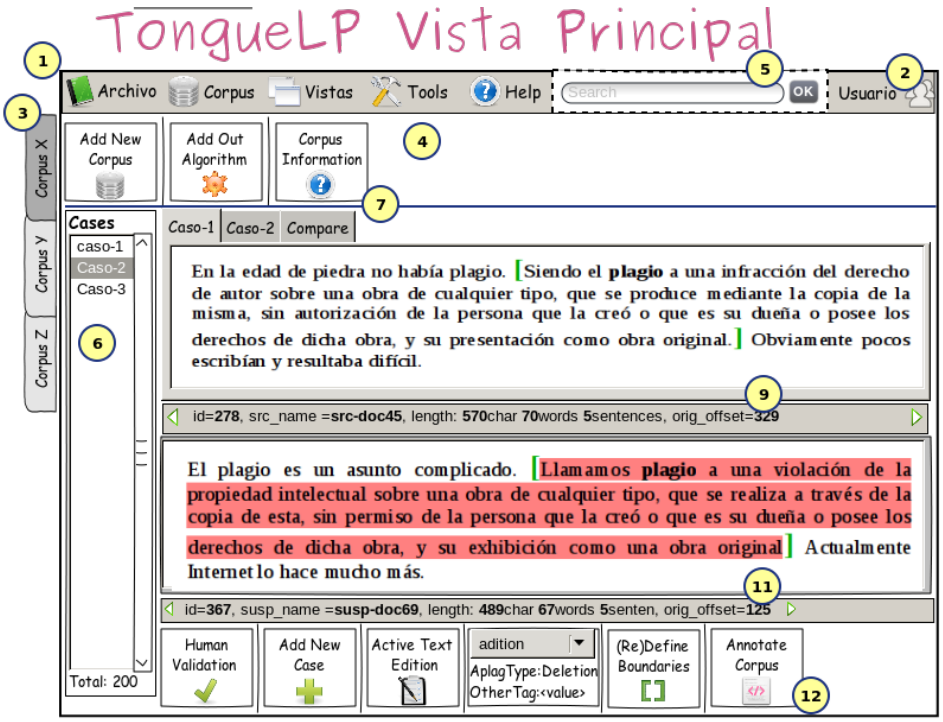

Fig. 1: Prototipo Vista Principal ToNgueLP, v0.2, elaborado en Pencil

Tareas de Ingeniería:

- *Para diseñar el prototipo hubo que leer el artículo de Laurence Antony "A critical look at software tools in corpus linguistics".*
- *Revisados otros softwares mencionados en el estudio de Antony, como: UAM CorpusTool, WordSmith Tools, RSTTools, Nooj. Objetivo extraer la primera versión de funcionalidades a partir de la observación y de las ideas concebidas durante la investigación de plagio.*
- *Adicionada las plantillas de Arq_de_Informacion.lyx y Levantamiento_de_Informacion.lyx elaborados por Enma para SXP 4.3*
- *Elaborado el Estudio de Homólogos.*
- *Diseñando en papel la herramienta.*
- *Elaborada la Arquitectura de Información con la primera lista de los elementos a tener en cuenta.*
- *Diseñando en Evolus-Pencil el prototipo.*
- *6/7/2014 Revisado con el desarrollador de UI todo el prototipo para su entendimiento.*

Flujo de Implementación

R1.5_02 Historia de Usuario “UI *Vista Principal*”

Historia de Usuario

Elaborado por el Ing Abel Meneses Abad
Basado en las planillas originales de la UCI, Copylefth Creative Common Share Alike Non-Comercial Use

Número: *R1.5_02*	**Nombre Historia de Usuario:** UI *Vista Principal*
Modificación de Historia de Usuario Número:	
Usuario: *Lingüistas*	**Iteración Asignada:** *3*
Programador responsable: *Leonel Salazar Videaux*	
Prioridad en Negocio: *Muy-Alto*	**Puntos Estimados:** *1*
Riesgo en Desarrollo: *Bajo*	**Puntos Reales:** *1*
Descripción: *Diseñar el UI en Qt de la vista principal.*	
Observaciones:	
Prototipo de interfase: (Aquí se debe poner el screenshot real hecho de la interfaz en Qt4)	
Tareas de Ingeniería: •	

R8.1_01 Historia de Usuario "*Parser ToNgueLP-plagiarism-corpus-parser.xml*"

	Historia de Usuario
Número: *R1.5_02*	**Nombre Historia de Usuario:** *Parser ToNgueLP-plagiarism-corpus.parser*
Modificación de Historia de Usuario Número:	
Usuario: *Sistema*	**Iteración Asignada:** *2*
Programador responsable: *Abel Meneses Abad*	
Prioridad en Negocio: *Muy-Alto*	**Puntos Estimados:** *1*
Riesgo en Desarrollo: *Bajo*	**Puntos Reales:** *1*
Descripción: *Parser de lectura y escritura para leer la estructura de ToNgueLP-plag-cases-corpus.xml*	
Observaciones: • *Disponible como módulo en: ../ToNgueLP/modules/ToNgueLP-plagiarism-corpus-parser* • *No posee, esto es un conjunto de clases y métodos.* • *Tiempo de desarrollo: del 25/7/14 al 31/7/14*	
Prototipo de interfase: No posee, esto es un conjunto de clases y métodos.	
Tareas de Ingeniería: • *25/7/14 Hacer el primer parser ejemplo.* • *Probar el parser ejemplo con ipython.* • *27/7/14 Comenzando el parser a partir de la versión del parser.read del PAN-PC-2012.* • *29/7/14 Terminar el parser.read v0.1 de ToNgueLP Corpus 001. => Seguir estudiando XML + SAX.* • *=> Estudiando XML con SAX (varios libros). <resultado> Utilicé el xml_indent.py del libro 'Python 2.6 Text Processing' y pude leer el etiquetado automáticamente, así como sus atributos.* • *=> <resultado> En el libro 'Python esential reference 4th edition' encontré los métodos de SAX para ver los nombres de los atributos, así como sus valores. Probé esto en el código xml_indent.py y funcionó a la perfección.* • *=> <resultado> La llamada que hacía al dtd en el XML funcionó durante el parseo, tuve que eliminarla porque no tengo hecho el dtd del corpus.* • *=> Revisé entonces los códigos que tenía hechos desde marzo, cuando dejé de trabajar en los códigos fuentes. <resultado>* ◦ ** Documenté las fechas en los métodos para el trabajo con n-gramas.* ◦ ** Hice algunos apuntes a los códigos de mi colección de métodos para NLPv0.2.* ◦ ** Solo había un fichero relacionado con el parseo XML contenía solo comentarios y ningún código reusable.* ◦ ** Borré este fichero para evitar confusiones futuras.*	

Elaborado por el Ing Abel Meneses Abad
Basado en las planillas originales de la UCI, Copylefth Creative Common Share Alike Non-Comercial Use

- *=> Probar la creación de un objeto de la clase IndentHandler en un fichero .py aparte para ver si puedo manipular los datos extraidos del XML. <resultado> Código XML_read.py v0.1 donde se separa la declaración de la clase IndentHandler.<nota> Esto puede simplificarse más aún.*
- *30/07/14 => Probar la lectura de atributos por separado. Ej corpus_name. <resultado> En el xml_indent se agregó un atributo a la clase IndentHandler llamado self.corpus_name, y a este fue asignado el valor del atributo name al encontrar el elemento 'corpus'. ¡Satisfactorio todo! <tarea> 01-29-07-2014 Estudiar la POO de python para programar esto como dios manda.</tarea>*
- *Terminar el parser.read v0.1 de ToNgueLP Corpus 001 => 01-29-07-2014 Estudiar la POO de python para programar esto como dios manda*
  - *Probar en ipython.[OK, v0.1]*
- *=> Estudiando programación orientada a objetos en el libro '2012-03_Book_Byte_of_Python' de la pág 80 - 90.<resultado> nada, todo muy simple, la inicialización de variables puede servir.*
- *=> Estudiando programación orientada a objetos en el libro 'OK_2002_Book_Downey_Allen_et_all-Aprenda_a_pensar_como_un_porgramador_con_Python' de la pag 157 - 195. <resultado> Entendí mejor la cosa, y al final debo trabajar con atributos de la clase digamos "corpus", que están fuera de todos los métodos para poderlos llamar desde cualquier método.*
- ** A esta clase debería programársele quizás un método getCaso que devuelva un caso con todas sus características, incluyendo el texto, que debe leer del fichero de texto.*
- ** Enviar a Leonel esta implementación para saber si en materia de POO estoy cometiendo algún error. </resultado> => Estudiar en algún otro libro para confirmar esta arquitectura del parser.*
- *=> Estudiando programación orientada a objetos en el libro '2013_Book_Dierbach_Charles-Introduction to Computer Science Using Python' de la pag 383 - XXX el capítulo 10 dedicado a la programación orientada a objetos. <resultado> En el libro el contenido de POO está muy bien explicado, confirmados algunos elementos del código.*
- *=> Empecé a implementar todos los atributos que contendrá la clase corpus, al llegar al 'creation_date' me doy cuenta de que <tarea>01-30-07-2014 hace falta manejar los datos de tipo date.</tarea>*
- *=> 01-30-07-2014 Estudiando manejo de Dates and Time values en el libro '2011_The python standard library by example' de la pag 173-196. <resultado> Hay que generar este valor en ToNgueLP utilizando la función time.ctime() que devuelve un string con todos los valores de la fecha.*
- *=> (cont) implementando los atributos de la clase corpus. <tare>02-30-07-2014 Hay que implementar tuplas dentro de los datos, pues 'snippet' contiene varios valores que están asociados a un mismo atributo. Ej susp_doc:susp-doc01765;src_doc:src-doc02276*
- *=> 02-30-07-2014 Recordando manejo de tuplas en el libro '2008_Book-Lie_Magnus-Beginning Python From Novice to Profesional' <resultado>*
  - ** Viendo las tuplas y no conveniente.*
  - ** Probando con diccionarios de listas. OK.*
  - ** Revisar el tema de los performances de los tipos de datos que están en este mismo documento, a ver si diccionario conviene. OK, diccionario es la segunda más rápida después de sets, pero en este caso no funciona.*
  - ** usar el index del diccionario como los 'id' de los pares. De manera que se pueda llamar a un elemento como Ej dict[id][3] esto es en el par [id] el elemento [3] obtener el valor.*
- *=> (cont) implementando los atributos de la clase corpus. Comencé a trabajar para leer los atributos múltiples. Encontré que devuelve el procesamiento un unicode. En el libro '2008_Book-Lie_Magnus-Beginning_Python_From_Novice_to_Profesional' en la pág 467 aparece una explicación sobre lo que pasa con esto específicamente en el procesamiento de los XMLs.<?> ¿Qué hago convertir estos unicodes a str?</?>*
- *=> Revisar los códigos del PAN para ver cómo lo hacen esta gente. <resultado> Los*

Elaborado por el Ing Abel Meneses Abad
Basado en las planillas originales de la UCI, Copylefth Creative Common Share Alike Non-Comercial Use

programadores de Webis trabajaron con el DOM.

- *31/7/14 => (cont) implementando la lectura de los atributos de la clase corpus. <resultado>*
  - ** logré convertir el unicode multiparámetro a string, con la funct 'str()'*
  - ** luego aplicando un ciclo doble dividí los parámetros para obtenerlos en un arreglo.*
  - ** asigné este arreglo al elemento[id] del diccionario creado para guardar los casos.*
  - ** @ a leonel con algunas dudas de programación sobre este código en particular.*
  - ** Logré visualizar el fragmento de texto de los documentos.*
- *Estudiando biblioteca de python de fuzzy string comparison. En este caso FuzzyWuzzy, que contiene también comparación de cadénas también por la fonética. Instalando el paquete.*
- *=> agregando un código al XML_read.py para ver en el caso incluído en el corpus cuanto dan todos los fuzzy_ratios. Todos dieron 100%.*
- *=> agregando un nuevo caso del PAN pero con obfuscation. => <resultado>*
  - ** Encontré un error en lo que había programado, una tontería. Arreglado.*
  - ** El resultado de la prueba evidencia que se demora fuzzywuzzy. (que además usa por debajo difflib), pero el ratio da 0.741908073425, 1% de similaridad al comparar solo los snippets(no los textos enteros como en mi algoritmo).*
  - ** Me dio por probar ambos textos con mi primer algoritmo de similaridad basado en lista de conjuntos. Tuve que recordar mis códigos NLP, etc. Tiempo: 3.74385690689 segundos, y 0% encontrado.*
- *=> Insertado un 3er caso del tipo "artificial low". Realizando el corrimiento del código y comparando con fuzzywuzzy. <resultado>*
  - ** Verificado manualmente que los fragmentos se parecen. Hay cambios de palabras por otras, errores ortográficos o sea cambios menores.*
  - ** El primer fuzz.ratio se demoró 0.0109889507294 encontrando un 23%, sin embargo el sort_ratio encontró un 95% de similaridad.*
  - ** Al correrlo por 2da vez para ver el tiempo del sort_ratio este fue de: 0.00577402114868*
- *Refinando el parser.<resultado>*
  - ** Cambiando nombre a la clase.*
  - ** Cambiando nombre al fichero que contiene la clase y sus métodos.*
  - ** Agregando método getCase.*
  - ** Incorporando en el XML read la posibilidad de visualizar cualquier caso, al entrarlo por la consola.*
  - ** Generizando la dirección del corpus en ./proyecto/data/<corpus_name>*

Elaborado por el Ing Abel Meneses Abad
Basado en las planillas originales de la UCI, Copylefth Creative Common Share Alike Non-Comercial Use