Detecció automàtica de còpies en un conjunt de documents MS Word

Jordi Cuenca i Ros
Enginyeria en Informàtica

Robert Clarisó Viladrosa
09 de gener de 2006

A l'Imma, per la paciència que ha tingut aquests quatre mesos

1. Resum

L'objectiu últim d'aquest projecte és l'estudi de la plagiabilitat dels lliuraments de les Proves d'Avaluació Continuada i pràctiques dels estudiants de la UOC així com l'estudi dels diferents mitjans per evitar-la.

Per al desenvolupament d'aquest projecte ha calgut fer tres estudis tècnics:

- Estudi del format .doc de Microsoft, amb l'objectiu d'avaluar-ne aquells punts que ens podrien ajudar a detectar la plagiabilitat
- Estudi de les diferents eines existents al mercat, amb l'objectiu de comprovar-ne les diferents funcionalitats, recollir-ne idees i estudiar-ne el seu disseny
- Estudi de la possibilitat d'automatitzar un registre dels diferents autors que han intervingut en l'edició d'un determinat document Word.

Un cop finalitzat aquest estudi s'ha procedit a desenvolupar un prototip d'eina automàtica d'avaluació de plagis mitjançant la implementació d'un dels algoritmes existents, en el nostre cas, el Greedy String Tiling.

Les conclusions que s'han extret d'aquest projecte són diverses:

- L'ús de les metadades existents en el format .doc pot ajudar a detectar possibles plagis
- Existeix una preocupació creixent en l'àmbit acadèmic pel que fa a la possibilitat de plagis als exercicis dels estudiants (sobretot, per la facilitat com es poden plagiar fonts d'Internet)
- Existeixen nombroses eines al mercat molt bones per a la detecció de plagis
- La detecció automàtica de plagis és un camp tècnicament complex, que ofereix moltes possibilitats

1.1. Paraules clau

plagi, document, autor, detecció, font, coincidència, metadades

1.2. Àrea del projecte

El projecte s'inscriu dins de l'Àrea d'E-learning de la UOC.

2. Índex de continguts

3. Índex de figures

4. Introducció

Abans de començar cap mena de projecte, és important conèixer-ne els objectius, així com decidir com se n'enfocarà el desenvolupament i tenir una planificació acurada i realista.

En els propers apartats, es desenvolupen aquests punts detalladament.

4.1. Justificació del projecte i context

La UOC ha apostat fortament per un sistema d'avaluació continuada dels seus estudiants. Aquest fet es reflexa en una gran quantitat d'exercicis (Proves d'Avaluació Continuada) que els professors i consultors han d'avaluar.

Davant d'aquest volum és molt complicat detectar plagis entre diferents solucions de forma manual. Per aquest motiu, la UOC desitja disposar d'una eina de detecció automàtica de plagis que permeti detectar-los de forma massiva.

Aquesta eina ha de ser àgil i pràctica. La sortida ha de permetre identificar fàcilment els plagis i ignorar els exercicis originals.

Les solucions a les PAC de la UOC es lliure molt sovint en documents .doc de Microsoft Word, d'aquí que el projecte s'hagi enfocat cap al tractament d'aquest tipus de documents. A més a més, els documents de Word inclouen, a part del contingut en text natural, una part sovint ignorada però que pot aportar informació clau a l'hora de detectar plagis, que són les metadades (per exemple, el nom de l'autor, el temps d'edició, etc.).

4.2. Objectius

Els objectius d'aquest projecte són:

- Estudi del problema de detecció automàtica de còpies en un context acadèmic.
- Avaluació de les eines existents de detecció de copies a l'actualitat de detecció de còpies
- Descripció de les particularitats de la comparació de textos en llenguatge natural
- Estudi del format dels documents .doc de Microsoft Word, tant pel que fa al contingut com a les metadades.
- Desenvolupament d'una eina automàtica de detecció de còpies que pugui ser utilitzada per l'equip docent de la UOC.
- Opcionalment, l'eina ha de ser portable.

4.3. Enfocament i mètode seguit

El projecte s'ha desenvolupat tenint molt present la planificació lliurada com a PAC1. El temps disponible per a desenvolupar-lo i documentar-lo és molt just i qualsevol desviació que es pugués produir era difícil de corregir. Per tant, s'ha intentat seguir al màxim la planificació decidida inicialment en tot moment.

No obstant, aquesta planificació s'ha seguit en forma de "grans objectius", és a dir, la distribució temporal mostrada arriba molt al detall de quins plaços té cada petita fita del projecte. Sovint, però, ens hem trobat que no podem continuar un dels punts indicats a la planificació en un determinat moment (s'està a l'espera d'alguna resposta en algun fòrum, per exemple) i s'ha invertit l'ordre d'execució d'algun dels punts. No obstant,

s'ha intentat en tot moment, mantenir la planificació "global" dels grans subapartats dels que està compost el projecte.

També cal destacar que, per evitar problemes de temporalització i acumulació de feines a les últimes etapes del projecte, s'ha anat redactant el present document durant tota la vida del projecte deixant-ne únicament una revisió final pel dies previs al lliurament.

4.4. Planificació del projecte

Per a dur a terme el projecte, s'han considerat les següents tasques:

1. Estudi preliminar del projecte
   a. Estudi de les diferents eines disponibles per a obrir, llegir i/o convertir fitxers en format .doc de Microsoft Word
   b. Estudi de les particularitats dels diferents llenguatges de programació: rendiment, llibreries estàndard, possibilitat d'ús de llibreries de manipulació de documents de Microsoft Word, tractament dels *strings* i *arrays*, etc.
   c. Revisió dels requeriments
   d. Revisió dels riscos
2. Recerca i estudi de les diferents eines d'avaluació automàtica de plagis existents actualment.
3. Recerca i estudi de les característiques del format .doc de Microsoft Word, detectant especialment aquelles que ens seran útils per a la detecció.
4. Recerca i estudi de les diferents opcions a l'hora d'enregistrar automàticament les diferents autories d'un document de Word.
5. Desenvolupament de l'eina
   a. Recollida i anàlisi de requeriments de l'eina a desenvolupar
   b. Mòdul d'entrada
      i. Disseny del sistema d'entrada (línia de comandes, visual, paràmetres necessaris, etc.)
      ii. Implementació d'un sistema de lectura del contingut de N fitxers Word: text i metadades (segons el decidit a l'apartat 3)
      iii. Test
   c. Mòdul de detecció de plagis
      i. Elecció d'un o més algorismes de detecció de plagis
      ii. Elecció dels paràmetres necessaris (tipus d'algorisme, tolerància, etc.)
      iii. Implementació dels algorismes
      iv. Test
   d. Mòdul de *reporting*
      i. Definició de plantilles de les dades a mostrar
      ii. Implementació
      iii. Test
6. Test global
7. Documentació preliminar
8. Lliurament preliminar

9. Revisió del desenvolupament
10. Test global definitiu
11. Redacció de la documentació definitiva
12. Lliurament

La distribució temporal és mostra a la següent taula:

Id	Tasca	Durada	Inici	Fi	Prec.
1	Redacció memòria	66 días	dj. 29/09/05	dj. 29/12/05	
2	Estudi preliminar	4 días	dj. 29/09/05	dt. 04/10/05	
3	Estudi eines tractament .doc	3 días	dj. 29/09/05	dl. 03/10/05	
4	Estudi llenguatges programació	1 día	dj. 29/09/05	dj. 29/09/05	
5	Revisió requeriments	3 días	dj. 29/09/05	dl. 03/10/05	
6	Revisió riscos	1 día	dt. 04/10/05	dt. 04/10/05	"3;4;5"
7	FITA: PAC1	0 días	dl. 03/10/05	dl. 03/10/05	
8	Estudi eines avaluació automàtica	3 días	dc. 05/10/05	dv. 07/10/05	2
9	Estudi característiques format .doc	10 días	dc. 05/10/05	dt. 18/10/05	2
10	Desenvolupament eina detecció	43 días	dc. 19/10/05	dl. 19/12/05	"8;9"
11	Recollida i anàlisi de requeriments	5 días	dc. 19/10/05	dt. 25/10/05	9
12	Mòdul d'entrada	8 días	dt. 25/10/05	dv. 04/11/05	11
13	Disseny sistema entrada	1 día	dc. 26/10/05	dc. 26/10/05	9
14	Implementació sistema d'entrada	5 días	dj. 27/10/05	dc. 02/11/05	13
15	FITA: PAC2	0 días	dt. 25/10/05	dt. 25/10/05	
16	Test i revisió	2 días	dj. 03/11/05	dv. 04/11/05	14
17	Mòdul de detecció de plagis	14 días	dl. 07/11/05	dj. 24/11/05	12
18	Elecció algorismes	2 días	dl. 07/11/05	dt. 08/11/05	
19	Elecció paràmetres	2 días	dl. 07/11/05	dt. 08/11/05	16
20	Implementació	10 días	dc. 09/11/05	dt. 22/11/05	"18;19"
21	Test i revisió	2 días	dc. 23/11/05	dj. 24/11/05	20
22	Mòdul reporting	16 días	dv. 25/11/05	dl. 19/12/05	
23	Definició plantilles	2 días	dv. 25/11/05	dl. 28/11/05	21
24	Implementació	10 días	dt. 29/11/05	dl. 12/12/05	23
25	FITA: PAC3	0 días	dl. 12/12/05	dl. 12/12/05	
26	Test i revisió	2 días	dt. 13/12/05	dc. 14/12/05	24
27	Test global	2 días	dl. 19/12/05	dt. 20/12/05	22
28	Estudi dels diferents mitjans d'enregistrament automàtic dels autors d'un document	2 días	dc. 08/12/05	Dj. 09/12/05	
29	Documentació preliminar	2 días	dv. 25/11/05	dl. 28/11/05	17
30	Lliurament preliminar	0 días	dt. 20/12/05	dt. 20/12/05	"27;28"
31	Revisió desenvolupament	5 días	dc. 21/12/05	dt. 27/12/05	29
32	Test global definitiu	2 días	dc. 28/12/05	dj. 29/12/05	30
33	Revisió documentació definitiva	2 días	dv. 30/12/05	dl. 02/01/06	31
34	Lliurament	0 días	dl. 02/01/06	dl. 02/01/06	32

4.5. Productes obtinguts

S'han obtingut tres productes del projecte.

- Plantilla de document Word on s'enregistra mitjançant l'ús de macros, els diferents autors d'un document (veure apartat 8.1)
- Document de Word on s'exemplifica l'ús del control de canvis per a registrar els diferents autors d'un document (veure apartat 8.2)
- Eina de detecció automàtica de plagis desenvolupada en Java (veure apartats 9 i 10)

4.6. Descripció de la memòria

La memòria s'ha redactat intentant ser coherent amb la planificació temporal del projecte per dos motius:

- Facilitar la tasca dels lectors tant per a la recerca de continguts com per la comprensió global del projecte i del seu desenvolupament
- Facilitar la tasca de redacció per tal d'acomplir els objectius mencionats a l'apartat *Enfocament i Mètode Seguit* sobre la redacció de la memòria de manera simultània al desenvolupament del projecte

Finalment, la memòria ha quedat composta d'una sèrie d'apartats i subapartats seguint, doncs, aquesta planificació temporal. No obstant, creiem que no és l'objectiu d'aquest apartat reproduir novament l'índex. Malgrat tot, sí que creiem oportú explicar, a grans trets, l'estructura de la memòria i resumir-ne el contingut.

L'apartat d'introducció (apartat 1) conté tota aquella informació referent a la justificació del projecte (4.1), els objectius (4.2), com s'ha dut a terme (4.3), quin mètode s'ha fet servir per desenvolupar-lo (4.3) i seguir la planificació (4.4), etc.

A l'apartat de l'estudi preliminar del projecte (5), es descriu com s'han dut a terme i a quines conclusions s'han arribat pel que fa a aspectes que calia conèixer i considerar abans de desenvolupar el projecte en si. Per exemple, quines eines ja existents al mercat podem fer servir i com (5.1), quins llenguatges de programació són més adients per al tipus de desenvolupament a fer (5.2), etc.

Seguidament, el document continua explicant els resultats i conclusions de la recerca de les diferents eines d'avaluació automàtica de detecció de plagis que existeixen actualment (6) i de les característiques dels documents .doc de Microsoft Word així com de les seves diferents versions (7).

A l'apartat sobre el desenvolupament (8), es descriu l'etapa de desenvolupament de l'eina amb les diferents decisions preses, els anàlisis funcionals (9.1), problemes trobats, etc.

Tot seguit, tenim un apartat dedicat a la instal·lació de l'aplicació desenvolupada així com dels diferents programes auxiliars que necessitem per a fer-la funcionar (10).

Finalment, a l'apartat de referències i bibliografia (13) s'hi detallen totes aquelles fonts que s'han consultat en algun moment. S'ha intentat classificar-les segons a quina part del projecte ha estat necessari consultar-les.

5. Estudi preliminar del projecte

Abans de començar el desenvolupament del projecte

Existeixen moltes opcions per a desenvolupar l'eina que se sol·licita: eines ja desenvolupades i a punt per a ser executades, llibreries, etc. i, a més a més, en diferents llenguatges de programació.

És per aquest motiu que, abans de desenvolupar l'eina ha calgut fer un anàlisi de les diferents opcions que tenim disponibles.

S'ha dividit aquest estudi preliminar en dues parts:

- Estudi de les diferents eines disponibles per obrir, llegir i/o convertir fitxers en format .doc de Microsoft Word
- Estudi de les particularitats dels diferents llenguatges de programació: rendiment, llibreries, tractament dels diferents tipus de dades, etc.

5.1. Eines de manipulació de fitxers .doc de Microsoft Word

El primer que veiem en estudiar aquestes eines és que tenim dues grans possibilitats:

- Desenvolupar un mòdul de lectura de fitxers .doc que afegirem a l'eina que hem de desenvolupar
- Fer servir una eina existent en l'actualitat que ens realitzi una conversió massiva dels documents .doc a un format fàcilment llegible: XML, text, HTML, etc.

Per a la primera possibilitat s'ha comprovat que existeixen nombroses llibreries en molts formats: DLL, OLE, etc. Algunes de les solucions trobades es mostren a la taula següent:

Nom	Fabricant/Desenvolupador	Llicència
downCast	Infinity-Loop GmbH	Freeware per ús personal
Microsoft Scripting Runtime	Microsoft	Freeware?

Figura 1 – Taula de llibreries de conversió de .doc a .xml o .txt

Els avantatges d'aquestes eines són:

- Permeten accedir fàcilment a les metadades del document
- Es poden integrar realment en l'eina desenvolupada

No obstant, aquestes eines tenen un seguit d'inconvenients:

- Són molt dependents de la plataforma on els vulguem executar (bàsicament, la majoria estan desenvolupades per a la plataforma Win32)
- Moltes són llibreries de pagament
- Impliquen molta feina per aprendre a fer-les servir. El desenvolupador no té gens d'experiència en desenvolupaments basats en llibreries externes d'aquest tipus i embarcar-se en aquest desenvolupament podia comprometre seriosament la finalització del projecte dins dels terminis previstos.

Existeixen nombroses solucions també per a la segona possibilitat. Com sempre, moltes d'elles són de pagament però n'existeixen algunes també gratuïtes. La següent taula mostra algunes de les solucions trobades (n'hi ha moltes més):

Nom	Fabricant/Desenvolupador	Llicència	Mode
Antiword	Adri van Os	Freeware	Comanda
CZ_Doc2Txt	Convert Zone	299 $	Comanda
Davisor Offisor	Davisor Ltd.	1100 €	Finestra
Doc2XML	Geoffrey Oakham	Freeware	Comanda
eXportXML	Schultz	149 $	Finestra (plug-in dins de Word)
ODC_ExpWordToXML	Microsoft	Freeware	Finestra
W2ML	Byte Ryte bv	¿?	Finestra
Wvware	SourceForge	Freeware	Comanda

Figura 2 – Taula d'aplicacions de conversió .doc a .xml o .txt en mode comanda

Finalment s'ha optat per l'aplicació Antiword. Aquesta aplicació té un seguit de característiques de han fet que l'hàgim considerat òptima per al projecte desenvolupat:

- S'executa en mode línia de comandes: això ens permet crear fàcilment un fitxer .bat que executi tota l'aplicació (conversió i anàlisi de possibles plagis) de manera que el procés de conversió es mostra totalment transparent a l'usuari.
- Antiword està disponible en totes les grans plataformes existents en l'actualitat i això ens ajudarà molt a poder garantir la portabilitat de l'eina. Està disponible en RISC OS, Linux/Unix, FreeBSD, BeOS, OS/2, Mac OS X, Amiga, VMS, NetWare, Plan9, EPOC, Zaurus, DOS i Windows.
- Ens permet extreure en diversos formats el text en format text pla.

5.2. Estudi dels diferents llenguatges de programació

A l'hora de decidir-nos per un llenguatge o un altre a l'hora de desenvolupar aquest projecte s'han tingut en compte els següents criteris:

- Nivell de coneixement del llenguatge: hi ha poc temps per a desenvolupar l'eina i hi ha poc marge per "aprendre" un llenguatge i entorns de desenvolupament nous.
- Portabilitat: si bé no és un requeriment obligatori, sí que es dóna com a requisit desitjable a l'enunciat la portabilitat de l'eina
- Rendiment: no s'han donat especificacions pel que fa als requeriments del rendiment de l'eina, no obstant, s'ha valorat aquest punt.
- Orientació del llenguatge al tractament de textos

5.2.1. Nivell de coneixement del llenguatge

L'autor coneix els següents llenguatges de programació: C, C++, Java i Visual Basic. No obstant, l'autor d'aquest projecte normalment no es dedica al desenvolupament de software i l'experiència amb els diferents llenguatges de programació és únicament aquella procedent de la realització de pràctiques a la universitat. El nivell de coneixement de cadascun dels llenguatges és:

- C: coneixements molt sòlids tant en el desenvolupament a alt nivell com a baix nivell

- C++, Java i Visual Basic: coneixements bàsics

5.2.2. Portabilitat

Pel que fa a la portabilitat, hem de descartar únicament Visual Basic perquè l'aplicació resultant seria únicament executable en entorns basats en Microsoft Windows. La resta de llenguatges esmentats no ofereixen problemes pel que fa a la portabilitat.

5.2.3. Rendiment

En aquest apartat qui millor puntuació obtindria seria la parella C/C++ i, en un segon lloc la parella Visual Basic - Java.

5.2.4. Orientació del llenguatge al tractament de textos

Cal tenir sempre present que aquest projecte requereix, per sobre de tot, tractar contínuament cadenes de caràcters. El tractament d'*strings* està molt ben tractat en Visual Basic i en Java però és bastant precari en C i C++.

Treballar doncs amb un llenguatge que ens permeti fàcilment concatenar, tallar, cercar, etc. *strings* ens pot reduir molt el temps de desenvolupament.

5.2.5. Conclusió

Finalment, s'ha optat pel llenguatge Java perquè és suficientment conegut per l'autor, és portable i, sobretot, perquè incorpora gran quantitat d'eines ja desenvolupades i un excel·lent tractament dels *strings* i s'ha considerat que el temps de desenvolupament és el factor més crític d'aquest projecte.

6. Eines d'avaluació automàtica de plagis

Actualment, existeixen al mercat nombroses eines de detecció de plagi. Bàsicament, existeixen dos tipus d'eines (algunes combinen ambdues funcionalitats): eines que comparen els documents amb multitud de fonts d'Internet i eines que comparen diferents documents entre ells.

A continuació, anem a analitzar algunes de les eines que s'han trobat.

6.1. Turnitin

El funcionament de *Turnitin* és realment senzill i el veurem a diferents eines (*Bibl. 23*). El professor (o la institució per la que treballa) ha d'adquirir una llicència del programa. Amb l'adquisició de la llicència, se li proporciona un identificador (*ID number*) i un password. El professor dóna aquest identificador als estudiants que han de crear-se una compta d'usuari a *Turnitin* sota aquell identificador.

Quan un estudiant ha de lliurar un document al professor, ho fa a través de la seva compta d'usuari de *Turnitin*. En lliurar el document, aquest és automàticament analitzat per l'eina.

Posteriorment, el professor pot connectar-se a *Turnitin* i revisar l'anàlisi de plagi realitzat.

Aquest anàlisi es dóna en forma de percentatge de coincidència global del document amb diferents fonts (ja siguin documents d'altres estudiants com fonts procedents d'Internet). La interfície on es mostren els resultats és molt visual i atractiva i sembla molt fàcil i intuïtiva[1]. Es pot navegar i consultar els paràgrafs sospitosos, veure'n quina part és la coincident, etc.

Turnitin inclou, a més a més, opcions molt interessants, com la possibilitat de comparar els documents tenint o no en compte la bibliografia consultada, tenint o no en compte la presència de l'enunciat de l'exercici, comparar amb tota la biblioteca de fonts de que *Turnitin* disposa o bé comparar amb una única font, etc.

Tot seguit, es mostren unes captures de pantalla que es troben a la web del fabricant:

[1] Malauradament, no s'inclou cap versió de prova i no s'ha pogut comprovar personalment el funcionament.

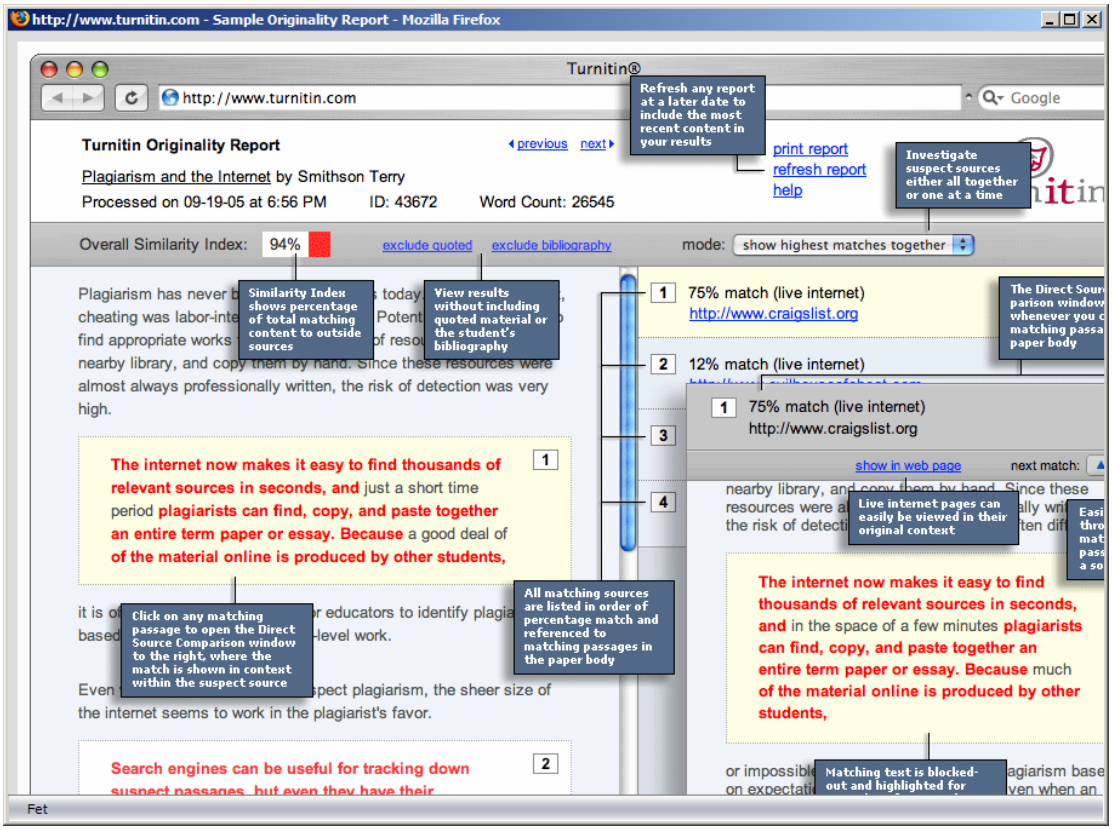


Figura 3 – Captura de pantalla del funcionament de *Turnitin* (I)

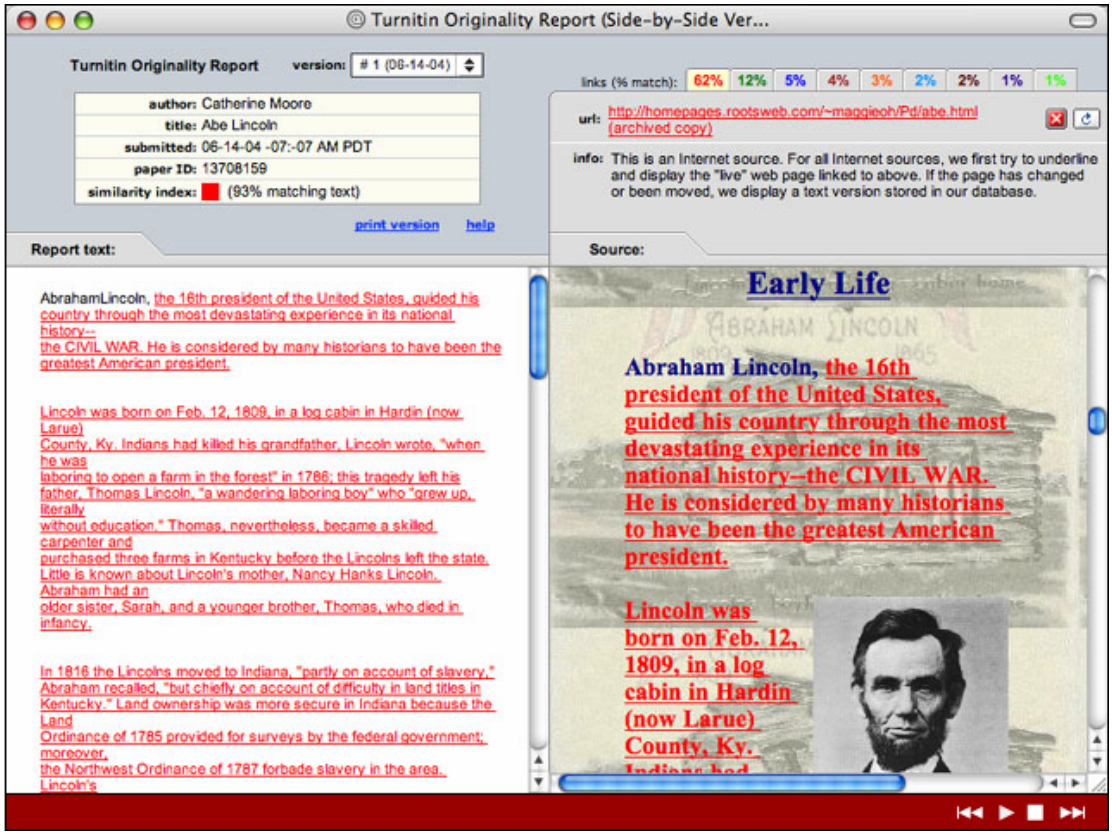


Figura 4 – Captura de pantalla del funcionament de *Turnitin* (II)

- **El millor**
  - ✓ és on-line (sempre està actualitzada)
  - ✓ no requereix instal·lació de software i, per tant, facilita la mobilitat del professor
  - ✓ compara els documents dels estudiants entre si
  - ✓ la interfície és molt agradable i marca clarament les zones sospitoses dels documents
- **El pitjor**
  - ✓ cal pagar per volum d'utilització[2]
  - ✓ les quotes d'utilització per llicència són molt baixes: una llicència serveix per a un instructor, 150 estudiants i 150 documents.
- **Més informació:** http://www.turnitin.com/static/home.html

6.2. Eve2 Plagiarism Detection for Teachers

El funcionament d'aquesta eina (*Bibl. 24*) és molt semblant al *Turnitin*. No obstant, hi ha certes diferències que mencionem tot seguit:

- No és una eina on-line sinó que es tracta d'un programa que s'instal·la a l'ordinador del professor
- No realitza comparacions entre els diferents documents dels estudiants sinó que únicament compara cadascun d'aquests amb les diferents fonts existents a Internet.

Eve2 ens permet escollir entre tres modes de recerca: ràpid (*quick*), mitjà (*medium*) o alt (*extra strength*) tot i que se'ns avisa a la web del fabricant que el mode *extra strength* requereix força temps de càlcul.

Una característica que ens crida molt l'atenció (negativament) és que, malgrat que accepta diversos formats d'entrada (Word, WordPerfect, etc.) únicament ens ofereix la informació completa si fem servir fitxers de text plans.

- **El millor**
  - ✓ El preu: 29.99 $ per una llicència il·limitada
- **El pitjor**
  - ✓ El *reporting* complet només està disponible si fem servir fitxers de text plans
  - ✓ S'ha d'instal·lar l'eina a l'ordinador i, per tant, cal que disposem d'un ordinador que la tingui instal·lada per a poder consultar els informes
  - ✓ No realitza comparacions entre els documents dels propis estudiants
- **Més informació:** http://www.canexus.com/

[2] No s'ha aconseguit descobrir el preu del producte ja que únicament es facilita en el moment de fer efectiva la compra.

6.3. OrCheck – Original Checker

Ens trobem novament davant d'una eina per a revisar si s'ha realitzat un plagi de les fonts disponibles a Internet (*Bibl. 25*). OrCheck empra la potència del motor de cerca de Google a partir de les paraules clau del document origen per tal de comparar-lo amb les web obtingudes amb el buscador.

La pròpia web del desenvolupador ja ens avisa que el procés de comparació serà llarg. Després del temps d'espera, obtenim una sortida en format gràfic semblant a aquesta:

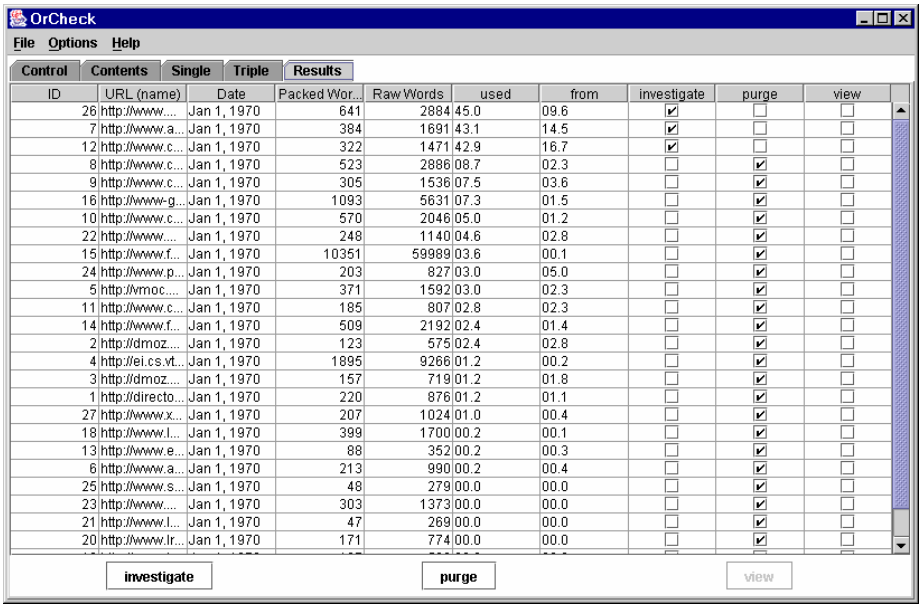


Figura 5 – Captura de pantalla de l'aplicació OrCheck

A la sortida obtindrem, de forma ordenada, les diferents URLs trobades a Google així com un índex de concordança (a la columna *Used* en forma de percentatge) i dos indicadors *Investigate* i *Purge* que ens indiquen, respectivament, si hi ha sospita de plagi procedent de la font o no.

Finalment, si fem clic a la casella *View* passarem al mode de visualització on se'ns mostraran les parts coincidents:

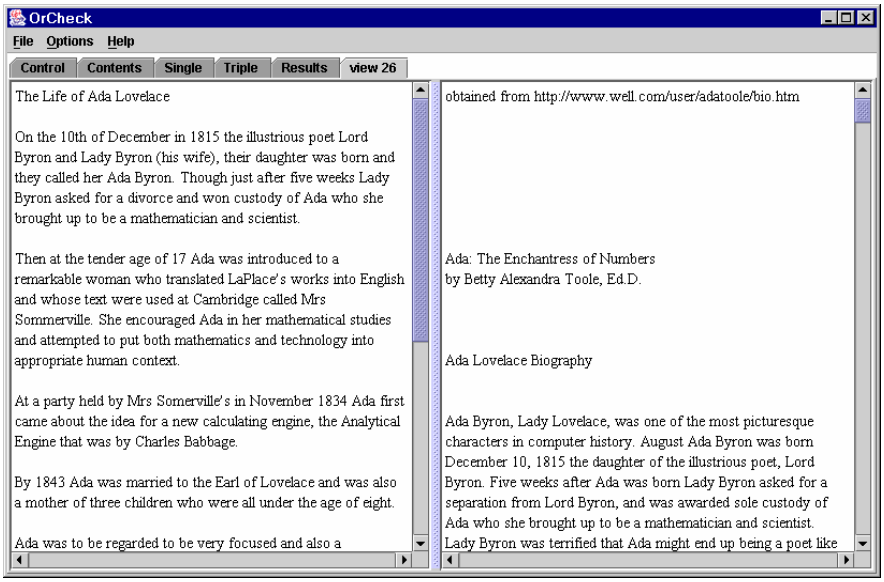


Figura 6 – Detall de la comparació de documents amb OrCheck

- **El millor:**
  - ✓ Desenvolupat amb Java i accessible via web, fet que n'augmenta la portabilitat
  - ✓ És gratuït
- **El pitjor:**
  - ✓ Requereix configurar força paràmetres de la configuració Java del nostre PC
  - ✓ Cal registrar-se a Google per obtenir una clau per a poder-lo executar
  - ✓ Només permet capturar fitxers en format text pla
- **Més informació:**
  - ✓ http://cise.sbu.ac.uk/orcheck/

6.4. VAST – Visual Analysis of Similarity Tool

VAST és una eina per a analitzar similituds entre dos documents, el que s'anomena *collusion* (*Bibl. 26*).

El funcionament de l'eina és molt senzill. Simplement disposem de dues àrees on enganxar el contingut dels dos documents que volem comparar, és a dir, primer els haurem d'obrir amb alguna eina que ens permeti copiar els documents al portapapers com, per exemple, el *Notepad* de Windows o qualsevol altre editor

Un cop omplertes les dues caselles, escollirem si volem tenir o no en compte l'ordre dels mots dins les frases (la documentació ja ens avisa que tenir-lo en compte augmenta molt el temps d'execució).

Com a resultat obtenim un mapa de col·lisió, de manera que, molt gràficament, veiem si s'han produït coincidències o no.

La següent figura mostra un mapa de col·lisió on hi ha certes coincidències:

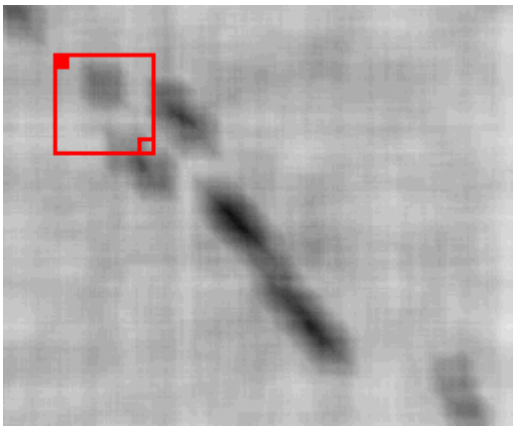


Figura 7 – Mapa de col·lisió amb coincidències entre dos documents

Un cop finalitzat l'anàlisi podem seleccionar les àrees sospitoses i visualitzar-ne el contingut d'ambdós documents tal i com es mostra en la següent captura de pantalla:

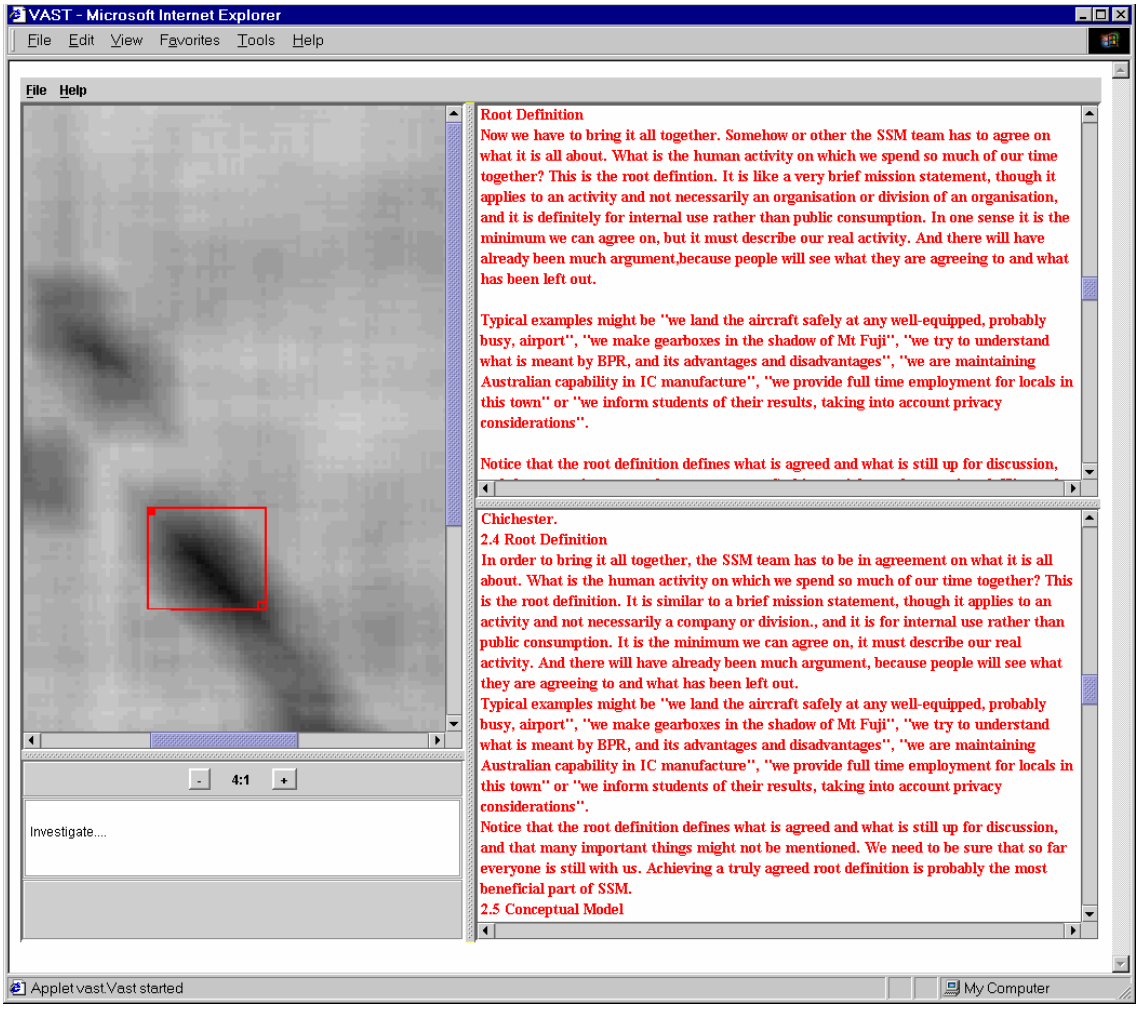


Figura 8 – Captura de pantalla de l'execució de l'eina VAST

- **El millor:**
  - ✓ Desenvolupat amb Java i accessible via web, fet que n'augmenta la portabilitat
  - ✓ És gratuït
- **El pitjor:**
  - ✓ Requereix configurar força paràmetres de la configuració Java del nostre PC
  - ✓ Només permet comparar dos documents cada vegada
  - ✓ El fet que s'hagi de fer *copy & paste* dels documents n'impossibilita l'ús massiu i automatitzat
- **Més informació:** http://cise.lsbu.ac.uk/orcheck/vast.html

6.5. MyDropBox

MyDropBox (*Bibl. 27*) és una altra eina que ens permet comparar de forma molt ràpida (segons la web del fabricant, necessita entre 1 a 2 minuts per realitzar un anàlisi complet d'un document) si s'ha plagiat alguna font disponible a Internet.

El funcionament consisteix, com en altres eines com *Turnitin*, en adquirir una llicència que ens donarà accés com a professors i permetrà que els alumnes puguin enviar-hi els documents dels treballs.

Un cop analitzats els documents, MyDropBox ens presenta els casos sospitosos de plagi d'una forma realment pràctica, mostrant-nos una capçalera amb les diferents URL d'on se sospita que s'ha pogut plagiar la informació i un detall on podem veure de forma molt gràfica quin/s fragment/s es correspon amb cada URL. Això ho fa marcant cada parella fragment-URL d'un color diferent. A la següent captura de pantalla es pot veure més clarament:

Figura 9 – *Report* obtingut amb l'eina MyDropBox

- **El millor:**
  - ✓ La velocitat de comparació
  - ✓ L'informe de la comparació generat
  - ✓ És on-line, per tant no requereix tenir cap software instal·lat
- **El pitjor:**
  - ✓ No és gratuït. No s'ha pogut consultar el preu de les llicències per a Departaments o Universitats però la llicència d'ús individual és de 89.90 $ anuals

- **Més informació:** http://www.mydropbox.com/

6.6. WCopyFind

WCopyFind (*Bibl. 28*) està dissenyat per a detectar plagis entre diferents documents (*peer review*) que s'indicaran al programa com a entrada i també inclou comparació amb diferents fonts d'Internet si bé, a diferència d'altre eines que s'han analitzat, aquesta es realitza de forma més manual (cal indicar manualment les URL que volem utilitzar com a fonts).

WCopyFind és una eina realment interessant perquè ens permet introduir com a entrades documents *nous* (*new document files*) i documents *vells* (*old document files*). Entendrem per document/s nou/s aquell/s que volem analitzar-ne la plagiabilitat. Per document/s vell/s entendrem aquells documents que sabem que poden haver servit de font per al plagi (per al cas de la UOC, per exemple, solucions a PACs o pràctiques de semestres anteriors).

El que fa WCopyFind és, per una banda, comparar els documents nous per parelles (tots contra tots) i, per l'altra banda, comparar cadascun dels documents nous amb cadascun dels documents vells (evidentment, no té sentit que es comparin els documents vells entre ells).

Vegem una captura de pantalla de la seva interfície:

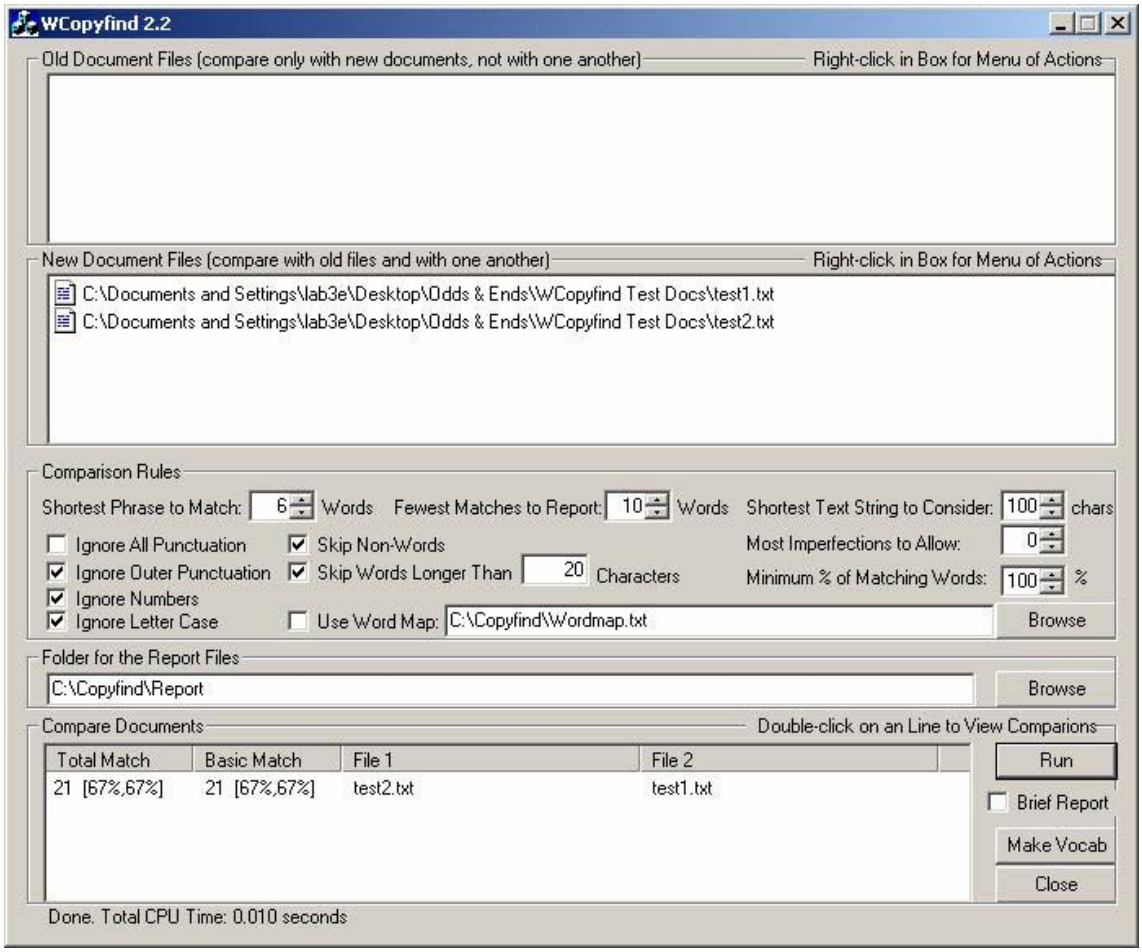


Figura 10 – Captura de pantalla de l'eina WCopyFind

WCopyFind, a més a més, aprofita el seu algoritme de comparació de textos per incorporar una eina d'ajuda a la redacció de documents per evitar les repeticions excessives de certes expressions típiques de cada autor (*pet expressions*).

Per a fer servir aquesta utilitat, primer hem de crear un document que contindrà les expressions que sabem que solem utilitzar en excés i, a continuació li diem a WCopyFind que analitzi el nostre document a la recerca de *pet expressions*.

- **El millor:**
  - ✓ És gratuït
  - ✓ Ràpid en l'anàlisi
  - ✓ Interfície senzilla i fàcil de ser servir
  - ✓ Està disponible en diverses plataformes
- **El pitjor**
  - ✓ L'informe presentat no és navegable
  - ✓ No sabem quina/es part/s del document són les sospitoses del plagi
- **Més informació:** http://plagiarism.phys.virginia.edu/Wsoftware.html

6.7. Conclusions

L'anàlisi de la plagiabilitat entre documents produïts per estudiants és un problema que preocupa bastant a professors d'arreu del món. Així ho demostren les nombres eines existents i projectes en desenvolupament que es poden trobar fàcilment a Internet (*Bibl. 33, 34* i *35*).

Les eines revisades se centren en dos grans focus de treball:

- ✓ anàlisi de la plagiabilitat entre estudiants, és a dir, la inclusió (cedida voluntàriament o no) de parts d'exercicis d'altres estudiants
- ✓ anàlisi de la plagiabilitat de fonts d'Internet, és a dir, la inclusió de fragments trobats en diferents fonts d'Internet (especialitzades en solucions d'exercicis o no)

De les eines analitzades, ens quedaríem amb Turnitin com a eina per a l'anàlisi de la plagiabilitat entre estudiants i Mydropbox per a l'anàlisi de la plagiabilitat procedent de fonts d'Internet. Ambdues eines, a part de disposar de molt bons motors de recerca i comparació, disposen d'unes interfícies d'usuari molt potents que permeten identificar molt fàcilment les parts susceptibles de ser plagiades.

7. Característiques del format .doc de Microsoft Word

El primer que s'ha pogut constatar en realitzar una recerca sobre el format .doc de Microsoft Word és que, en tractar-se d'un format propietari de Microsoft hi ha molt poca informació disponible (*Bibl. 5, 9, 10, 11 i 14*).

S'ha consultat exhaustivament diverses fonts a la recerca d'informació sobre aquest format: cercadors, fòrums, *news* d'UseNet, etc.

En principi, l'única manera d'obtenir informació sobre aquest format és participant en programes de *partnership* amb Microsoft. No obstant, s'han filtrat a Internet fragments d'informació sobre els diferents formats. En especial, existeix un document anomenat **wword8.html** que explica el funcionament del format .doc de Word 97[3] que podem trobar a la pàgina http://www.wotsit.org/download.asp?f=wword8.

7.1. Versions

Un primer problema que ens trobem només de començar la recerca és que el format .doc canvia amb cada versió de Microsoft Word que apareix al mercat. Aquest fet comporta diversos inconvenients:

1. Desenvolupar programari que llegeixi documents .doc implica actualitzar-lo periòdicament per adaptar-lo als nous formats
2. És difícil la recerca d'informació sobre el format
3. La conversió d'una versió més antiga a una de nova no sempre és possible realitzar-la amb exactitud. En aquest sentit, existeixen nombroses queixes expressades pels usuaris a la xarxa sobre diferències en els formats de fitxers convertits.

Per solucionar el primer punt, la millor opció és no desenvolupar el nostre software lector de documents .doc des de zero, és a dir, no implementar la part que llegeix el fitxer. És molt més aconsellable emprar les API ofertes per Microsoft i mantenir-nos actualitzats a mesura que apareixen nous formats de Word. Citant el propi document **wword8.html** llegim:

> *"Word 97 is an OLE 2.0 application. A Word binary file is a docfile and Word binary data is written into streams within the docfile using the OLE 2.0 docfile APIs. These streams are stored in the file as linked lists of file blocks and this data cannot be reliably accessed by using the operating system Open APIs. To access data within a Word binary file, the file must be opened using the OLE 2.0 docfile APIs, and it must be read with the appropriate docfile APIs."*

El segon i el tercer punt són de més difícil solució. Pel que fa al segon, és política de Microsoft l'hermetisme sobre els seus formats i únicament podem aconseguir-ne informació demanant-la directament a l'empresa. El tercer punt serà més o menys problemà-

[3] En realitat, tal i com es comenta més endavant en aquest mateix document, el format del fitxer .doc canvia pràcticament a cada versió de Word. No obstant, els termes generals es mantenen de versió en versió i el que es comenta sobre el format dels fitxers de Word 97 és vàlid, a grans trets, per als altres formats fins a la versió d'Office XP.

tic en funció de la comptabilitat entre els diferents formats que apareguin al mercat amb les noves versions de Microsoft Word.

Les versions dels diferents formats dels fitxers .doc[4] que han aparegut fins al moment són:

- Document de Microsoft Word 2.0
- Document de Microsoft Word 6.0 i Microsoft Word 95
- Document de Microsoft Word 97 i Microsoft Word 2002 (conegut com a Word XP)
- Document de Microsoft Word 2003
- Document de Microsoft Word 12

7.2. Format

El primer que observem si intentem editar un fitxer corresponent a un document de Word és que no és llegible. El document està guardat en format binari[5] i té una estructura certament complexa.

El documents de Word consten de diferents segments emmagatzemats dins del fitxer en forma de llistes enllaçades. La informació que s'emmagatzema dels documents, a alt nivell és:

- Metadades del document
- Text
- Imatges i objectes incrustats
- Informació de formats i estils
  - ✓ Format del text
  - ✓ Informació d'estils
  - ✓ Capçaleres i peus de pàgina
  - ✓ Notes de peu de pàgina i notes de final de document
  - ✓ Paginació del document
  - ✓ Seccions

7.2.1. Metadades del document

A part del contingut editat per l'usuari (text, imatges, formats, etc.) els documents de Word, en totes les seves versions, registren una sèrie d'informació de forma automàtica. Aquesta informació, anomenada **metadades del document** (en anglès *Summary Information Properties*) és una taula d'atributs que mostrem a continuació:

[4] Tenint en compte únicament les versions aparegudes per a Microsoft Windows

[5] El format del document Word és en format binari fins a la versió XP d'Office. A partir de la versió 2003, el format es canvia a format XML comprimit tal i com es pot veure més endavant en aquest document (*Bibl. 6 i 7*).

Element	Tipus	Descripció
Title	string	El títol del document. No ha de coincidir necessàriament amb el nom del fitxer. Es fa servir per fer cerques.
Subject	string	Una breu descripció del motiu del text
Author	string	Nom de la persona que va guardar el document per primera vegada
Company	String	Empresa on treballa l'autor
Keywords	string	Paraules clau
Comments	string	Comentaris de l'autor sobre el text
Template	String	Plantilla que s'ha fet servir per a crear el document i en el que aquest es basa
Last Saved By	string	Nom de l'última persona que ha guardat el document
Revision Number	integer	Nombre de vegades que s'ha guardat el document
Total Editing Time	integer	El temps d'edició del document
Last Printed	date/time**	Última vegada que s'ha imprès
Created Time/Date	date/time**	Data de creació
Last Saved Time/Date	date/time**	Data de l'última actualització
Pages	integer	Nombre de pàgines
Words	integer	Nombre de paraules
Characters	integer	Nombre de caràcters comptant espais
Thumbnail		
Name of Creating Application	string	Nom de l'aplicació que ha actualitzat el document per última vegada
Security		

7.2.2. Text

El text del document es guarda en un segment[6] anomenat *Main Stream*. El text es guarda com a ASCII (sense comprimir) únicament amb certes restriccions derivades de l'ús de caràcters de control: finalitzacions de paràgraf (ASCII 13), salts de pàgina (ASCII 12), tabulacions (ASCII 9), etc.

Dins del propi segment *Main Stream* és on s'emmagatzema també la informació de format del text. Més endavant, en aquest mateix document, veurem de quina manera (veure apartat 7.2.4).

7.2.3. Imatges i objectes incrustats

Les imatges i els objectes incrustats s'emmagatzemen dins d'un segment anomenat *Data Stream*.

Les imatges es guarden inserint dins del segment (després d'una capçalera de control) directament els *bytes* corresponents al fitxer original de la imatge, és a dir, els *bytes* del

[6] A l'original, *stream*

document .bmp, .jpg, etc. La major o menor compressió de la imatge inserida dependrà, doncs, del format original d'aquesta. No obstant, a partir de la versió 2002 d'Office, el Word incorpora utilitats per a comprimir les imatges inserides independentment del format d'origen.

De manera molt similar, es guarden els objectes incrustats.

7.2.4. Informació de formats i estils

Tal i com hem vist, la informació del format del text es guarda dins del mateix segment de dades (*Main Stream*).

Per tal de no haver de guardar tota la informació corresponent al format d'un determinat text, Word únicament en registra les diferències respecte a un format preestablert. D'aquesta manera, únicament cal registrar quin és el format base del nostre text i anar registrant l'inici i la finalització dels diferents atributs del text. Per exemple, el present text està escrit en tipus de lletra *Times New Roman* tamany 12. Aquest fet es registra una única vegada. Ara bé, quan introduïm alguna variació sobre aquesta tipografia, per exemple, escrivim un fragment en **negreta**, Word registra l'inici i la fi de la negreta i no registra tota la informació sencera del nou fragment.

Microsoft Word ens permet treballar també amb estils. Un estil és un conjunt predefinit d'atributs (poden ser atributs referents únicament al text o bé atributs referents al text i al paràgraf que el conté). D'aquesta manera tindrem, per exemple, un estil *normal* que serà la tipografia per defecte (en aquest document *Times New Roman*, tamany 12, justificat tant per la dreta com per l'esquerra, amb interlineat senzill); un altre estil podria ser el del títol d'un apartat (en aquest document *Arial*, tamany 16, en negreta i justificat tant per la dreta com per l'esquerra).

Aquest fet, Word l'enregistra mantenint a l'inici del fitxer una taula d'estils i la seva definició. Així, d'un determinat text, únicament cal que en registrem l'estil amb el que està redactat i no cal que en registrem cada vegada tota la informació de format. En fem prou en tenir-la registrada una única vegada.

7.3. Word 2003. Evolució a XML

A partir de la versió 2003 de la *suite* Office, Microsoft aposta per un canvi profund pel que fa al format .doc (i, de fet, .xls, .pps, etc.). A partir d'aquesta versió, Microsoft passa dels fitxers binaris als fitxers basats en l'especificació XML.

Aquest canvi aporta claredat al contingut físic del fitxer i obre la porta cap a una major facilitat per al coneixement del format així com al desenvolupament d'eines per a tractar-lo. Aquest trencament amb l'antiga filosofia basada en documents binaris encara és més palès amb l'arribada de la versió 12 (2005) de la *suite* Office. A partir d'aquesta versió es documenta extensivament el format del document. En paraules de Jean Paoli, *Senior Director and XML Architect* de Microsoft:

> *The new format will be fully documented and available ahead of the release date of "Office 12" so customers and partners can provide us early feedback.* (*Bibl. 6*)

A la versió 2003 del document, aquest era un únic document XML amb tot el contingut. A partir de la versió 12, el document constarà, en realitat, de diversos documents XML

més petits comprimits i empaquetats junts seguint l'algoritme estàndard de compressió ZIP (fet que, novament, veiem que està orientat a "obrir" el format del document ja que Microsoft ha optat per fer servir un algoritme de compressió conegut en lloc d'emprar un possible algoritme propietari), seguint així la línia seguida per altres eines semblants com, per exemple, Open Office.

7.3.1. Format del text

De manera molt semblant a com hem vist que es desava el format del text en els documents binaris, en el format XML també tenim dues maneres de fer-ho.

A la primera manera tenim que la informació del format del text es desa en forma de *tags* de tipus *run* on s'especifiquen les accions de format que cal executar sobre el text (per exemple, aplicar negreta, un tamany de text, etc.).

L'altra manera de que disposem és registrar la informació d'aquests *runs* dins d'un estil i aplicar els estils al text en comptes dels *runs*. També podem fer una combinació dels dos.

7.3.2. Metadades

Els documents de Word 2003 i Word 12 continuen guardant valuosa informació de metadades igual com passava amb els seus formats predecessors. En aquestes últimes versions, s'ha ampliat el conjunt de les metadades. A continuació, mostrem aquelles que no són presents als formats anteriors però que sí ho són a partir de la versió 2003:

Element	Tipus	Descripció
Category	string	Categoria del text. Serveix, per exemple, per a agrupar diferents documents en una mateixa categoria (per exemple: notes de premsa, articles, receptes de cuina, etc.)
Manager	string	Superior jeràrquic de l'autor del document
HyperlinkBase	string	Adreça base per als diferents *links* que hi hagi al document. Per exemple, podem posar-hi www.uoc.edu i així els *links* els podrem indicar com a /personal/jordi.html de manera que, en realitat, estaran apuntant a www.uoc.edu/personal/jordi.html
PresentationFormat	string	En quin format s'ha d'obrir el document: web, presentació preliminar, etc.
Guid	string	Número únic d'identificació del document
Pages	integer	Nombre aproximat de pàgines
Words	integer	Nombre aproximat de paraules
Characters	integer	Nombre aproximat de caràcters sense compta espais
CharactersWithSpaces	integer	Nombre aproximat de caràcters comptant els espais
Bytes	integer	Tamany aproximat del document
Lines	integer	Nombre aproximat de línies
Paragraphs	integer	Nombre aproximat de paràgrafs

Alguns dels atributs ja eren presents a les versions anteriors però se'ls ha canviat el nom:

Versions binaries	Versions XML
Last Saved By	LastAuthor
Comments	Description
Revision Number	Revision
Total Editing Time	TotalTime
Last Saved Time/Date	Last Saved
Name of Creating Application	AppName

8. Històric d'enregistraments d'un document

S'ha plantejat la necessitat d'emmagatzemar automàticament un històric de les persones que han guardat el document.

Existeixen dues alternatives que, lluny de ser excloents, es poden complementar.

La primera d'elles és l'ús de macros per enregistrar de forma automàtica (i oculta a l'estudiant) cadascun dels autors que ha anat enregistrant el document en forma de versions d'aquest.

L'altra consisteix en activar de forma permanent el control de canvis de Word (*Bibl. 21* i *22*).

8.1. Ús de macros per a registrar autors

Fent ús d'unes macros amb nom especial, podem aconseguir que Word registri certs events sobre un document. Aquests events són:

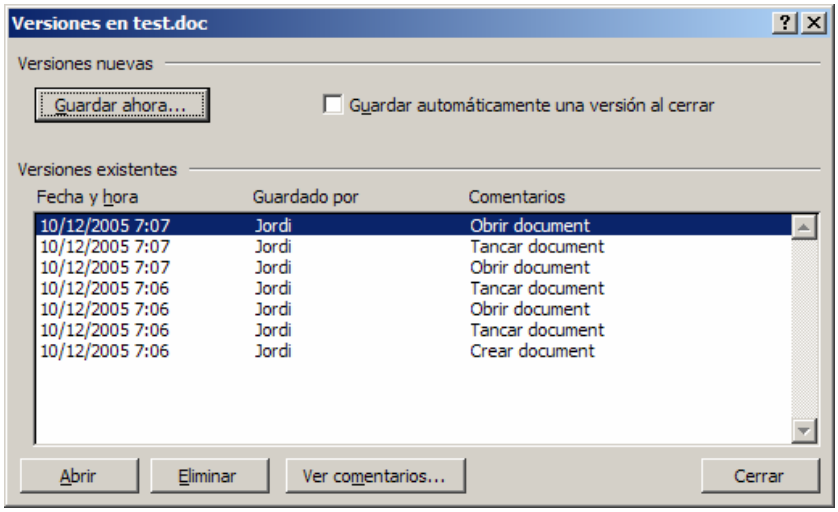
- Crear d'un document nou a partir d'una plantilla
- Obrir un document de Word
- Tancar un document de Word

Els noms de macros que haurem d'utilitzar, respectivament, per a controlar aquests events són: AutoNew, AutoOpen i AutoClose.

El contingut d'aquestes macros és molt senzill, ja que únicament el que farem serà registrar una nova versió del document, afegint-hi un comentari.

```
Sub AutoClose()
    ActiveDocument.Versions.Save Comment:="Tancar document"
End Sub
```

Tot seguit, podem veure un exemple d'execució d'aquestes tres macros, després de diversos accessos al document:



L'inconvenient d'aquest sistema és que un usuari malintencionat pot eliminar versions si ho desitja, esborrant, d'aquesta manera el rastre.

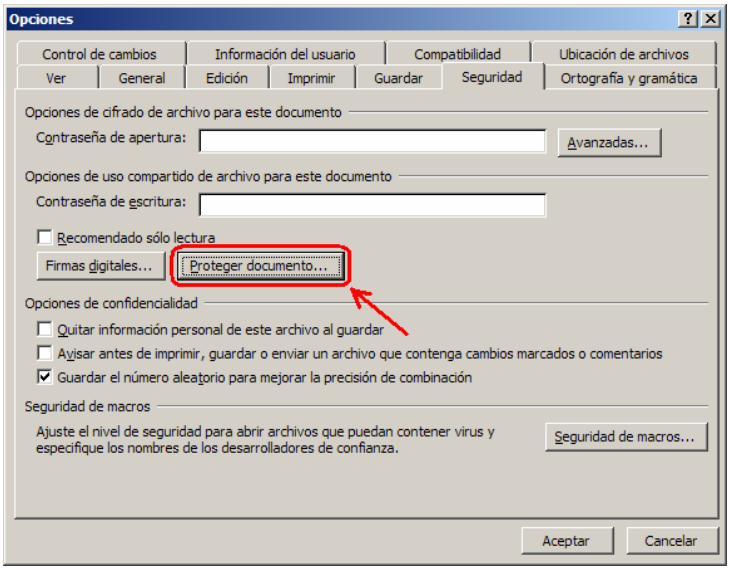
No obstant, cal dir que la majoria d'usuaris de Word no coneixen l'existència ni de les macros ni de les versions pel que aquest registre "silenciós" no seria detectat. Malauradament, no hi ha cap manera d'evitar la modificació del control de versions ni tampoc l'eliminació de les pròpies macros que el generen.

8.2. Ús del control de canvi per registrar autors

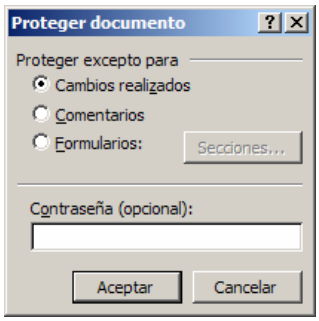
La segona opció de que disposem per a registrar els diversos autors que accedeixen i modifiquen un document és fer servir el control de canvis que incorpora Word.

Per a activar-lo, accedirem al menú **Herramientas → Opciones** i, a la pantalla que ens apareixerà, anirem a la pestanya **Seguridad**.

Un cop allà, premerem sobre el botó **Proteger el documento**.

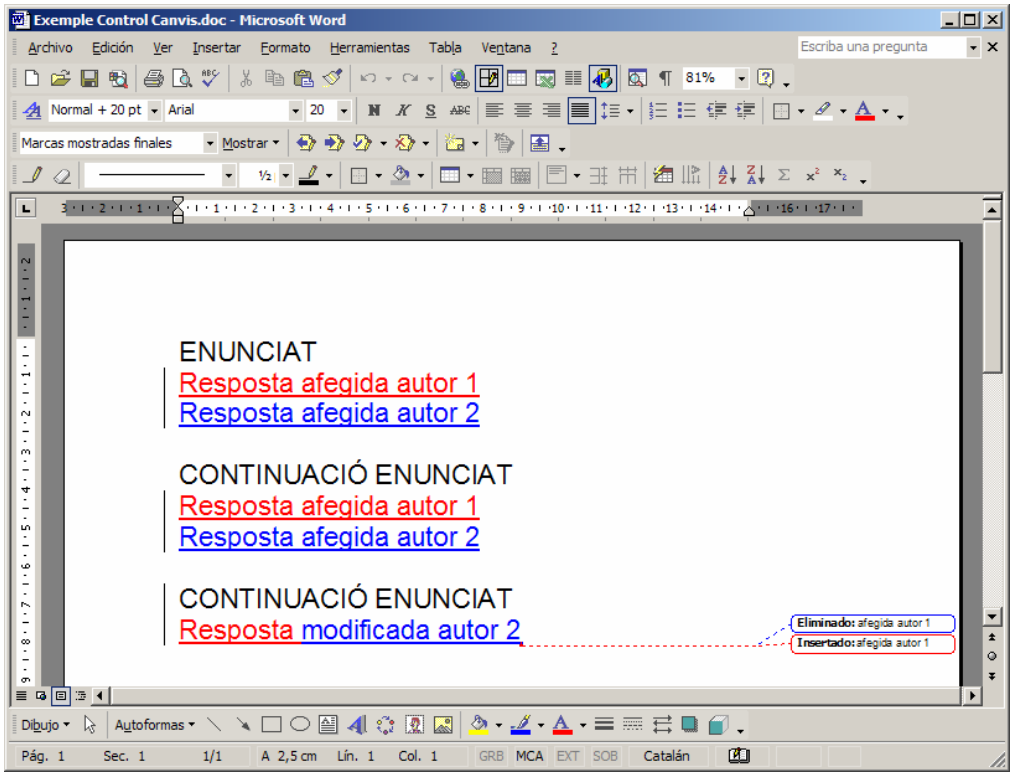


Ens apareixerà un quadre de diàleg on se'ns sol·licitarà quina mena de protecció volem. Escollirem **Proteger excepto para Cambios Realizados** i introduirem un password per a protegir el document:



Aquest sistema, a diferència de l'anterior, és molt més robust davant de possibles intents de deshabilitar-lo ja que es necessita conèixer el password per a fer-ho.

A continuació, es mostra un exemple de document manipulat per dos autors diferents, juntament amb l'enunciat proporcionat per la universitat.



8.3. Conclusions

Existeixen dues maneres complementàries de registrar l'autoria de diferents parts del document. Cadascuna d'elles té uns avantatges i uns inconvenients:

Registre mitjançant macros:

- Avantatges: és "silenciós" i, per tant, l'alumne, no sap que s'està registrant
- Inconvenients: el registre és modificable i les macros es poden deshabilitar/esborrar. No obstant, la deshabilitació de les macros o el fet d'esborrar-les també pot ser un indicatiu de frau.

Registre mitjançant el control de canvis:

- Avantatges: protegit per password, l'usuari no pot deshabilitar-lo.
- Inconvenients: l'usuari és conscient que l'estan registrant, per tant, pot anticipar-se al control i buscar-hi solucions alternatives. Per exemple, fent un copy & paste d'un document d'un altre company sobre el propi, no deixa constància de la còpia.

Així doncs, cap dels dos mètodes és infalible. De fet, ni tant sols la seva combinació ho és. No obstant, no tots els estudiants coneixen Word suficientment com per a poder evitar aquests controls.

Adjunt a aquest document, es proporcionen els documents **PlantillaMacros.dot** i **Exemple Control Canvis.doc** que exemplifiquen, respectivament els dos mètodes explicats.

9. Desenvolupament

Un cop analitzades els diferents algoritmes i eines de detecció de plagis en llenguatge natural i analitzat el format .doc de Microsoft Word on hem pogut detectar algunes característiques que ens podran ajudar a la detecció, s'ha passat a desenvolupar l'eina.

El següent esquema ens mostra, a alt nivell, el funcionament de l'eina:

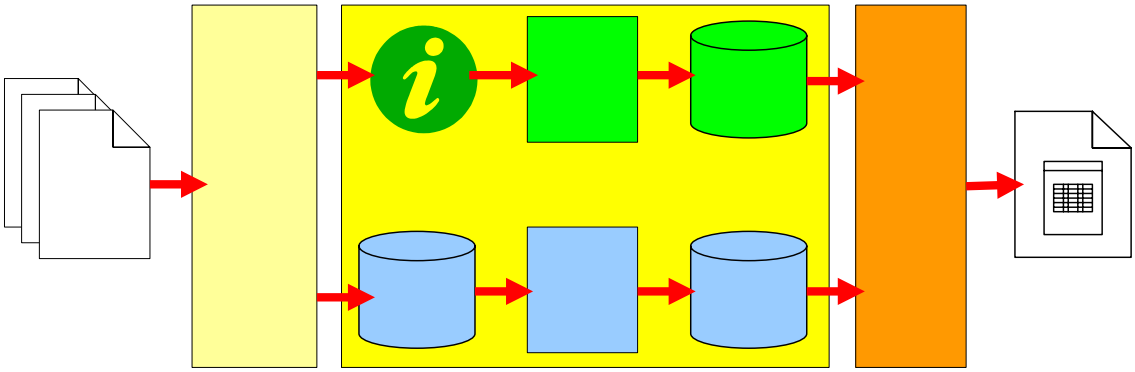


Figura 11 – Esquema general del funcionament de l'analitzador

Malauradament, durant la realització del projecte, es va comprovar que l'elecció d'AntiWord per a l'extracció del text dels documents de Word no ens permetria extreure la major part de les metadades interessants (*Bibl. 3 i 4*). A causa de la detecció d'aquest inconvenient en un estadi avançat del projecte, es va decidir no fer-les servir en la detecció i es va incorporar l'apartat dedicat a l'enregistrament d'un històric d'enregistraments d'un document (apartat 8).

9.1. Anàlisi funcional de l'eina

Per facilitar-ne el desenvolupament i, posteriorment, el manteniment, s'ha dividit l'eina en tres blocs funcionals:

- Mòdul d'entrada (preprocés)
- Mòdul de detecció (anàlisi)
- Mòdul de sortida (*reporting*)

Vegem, a continuació, cadascun dels blocs funcionals amb més detall:

9.1.1. Mòdul d'entrada

El mòdul d'entrada compleix dues funcions bàsiques i necessàries abans de procedir a l'anàlisi de la plagiabilitat dels documents:

- Obrir els documents en format .doc
- Preprocessar el contingut dels fitxers per tal de facilitar-ne el posterior anàlisi:
  - ✓ Eliminar contingut no processable: conjuncions, disjuncions, signes de puntuació, espais en blanc, mots buits…
  - ✓ Eliminar accents i altres signes de puntuació
  - ✓ Convertir el contingut del fitxer a majúscules

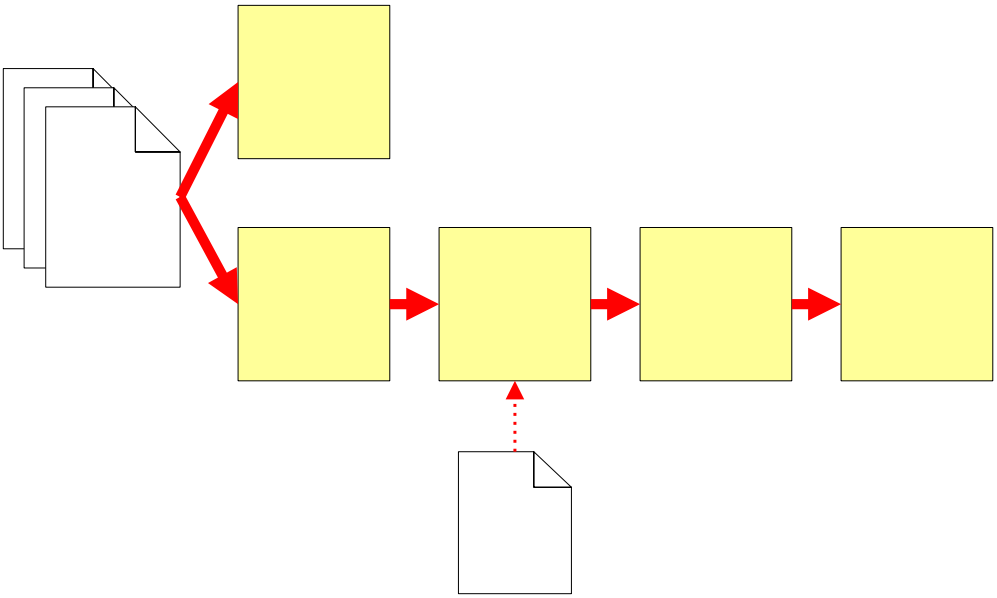


Figura 12 – Funcions del mòdul d'entrada

9.1.2. Mòdul de detecció

El mòdul de detecció és qui s'encarregarà d'aplicar l'algoritme de detecció de plagis (*Bibl. 30 i 32*) al text pretractat al mòdul d'entrada i a les metadades.

Hi ha dues estratègies per a analitzar similituds entre documents: *tots contra tots* o bé mitjançant la utilització d'un diccionari.

L'estratègia *tots contra tots*, com el seu nom indica, consisteix en comparar totes les possibles parelles de documents a analitzar:

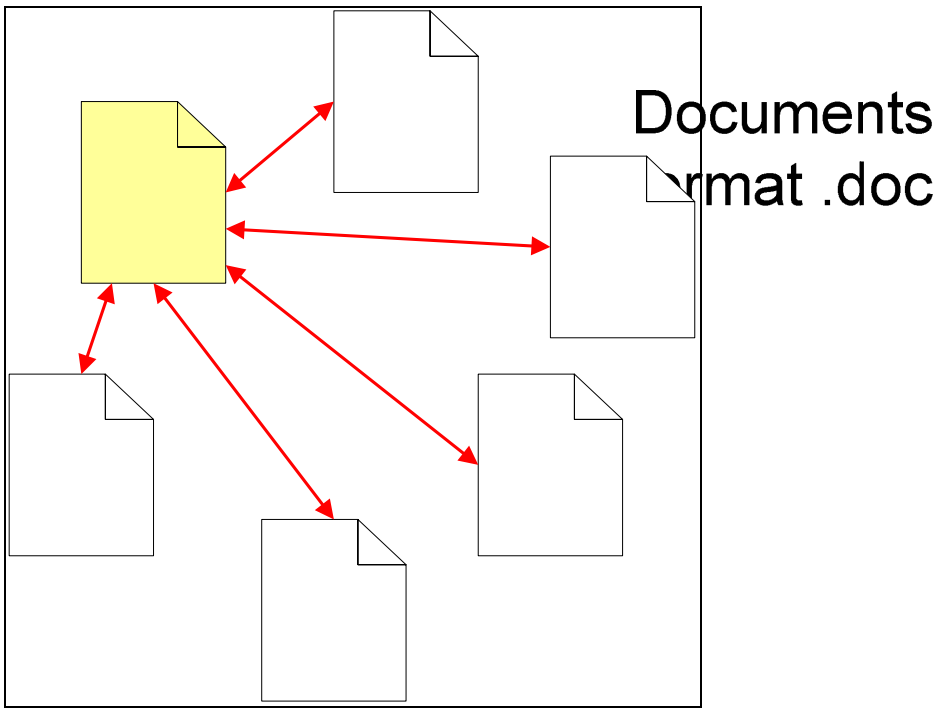


Figura 13 – Estratègia de comparació *tots contra tots*

El problema principal d'aquesta estratègia és el nombre de combinacions que cal fer per a fer un anàlisi. Es tracta de realitzar combinacions sense repetició de N documents prenent-los de dos en dos:

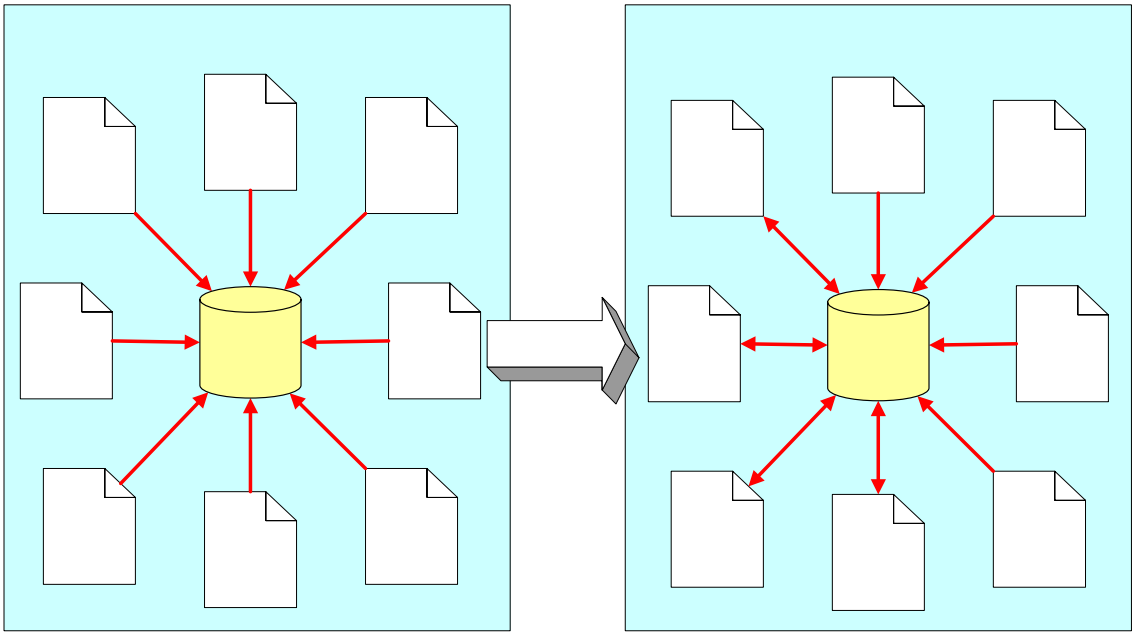
$$C_N^2 = \frac{N!}{2!(N-2)!}$$

A l'enunciat del projecte se'ns ha dit que cal tenir en compte el gran volum de documents que podem arribar a haver de comparar. Si calculem el nombre de combinacions a realitzar per a, per exemple, analitzar 300 documents obtindrem:

$$C_{300}^2 = \frac{300!}{2!(300-2)!} = \frac{300!}{2 \cdot 298!} = \frac{300 \cdot 299}{2} = 44850$$

És a dir, hem de realitzar 44.850 comparacions de documents sencers. Si suposem una mitjana de 10 pàgines per document, hem d'analitzar 448.500 pàgines!

L'alternativa a l'estratègia *tots contra tots* és la creació i ús d'un diccionari amb les diferents aparicions dels k-grames.



Finalment s'ha escollit el mètode de comparació *tots contra tots* per a la realització d'aquest projecte perquè és el més senzill d'implementar.

9.1.3. Mòdul de sortida (reporting)

Les especificacions del projecte ens diuen que la sortida ha de ser molt visual i que ens permeti, ràpidament, veure aquells k-grames detectats com a plagis. D'aquesta manera, la persona que utilitzi l'eina podrà jutjar si es tracta d'un plagi real o bé d'una coincidència (sovint, dos exercicis on la resposta ha de ser molt objectiva, poden donar peu a respostes pràcticament idèntiques).

La sortida que s'ha decidit implementar és en forma de document HTML amb la següent estructura:

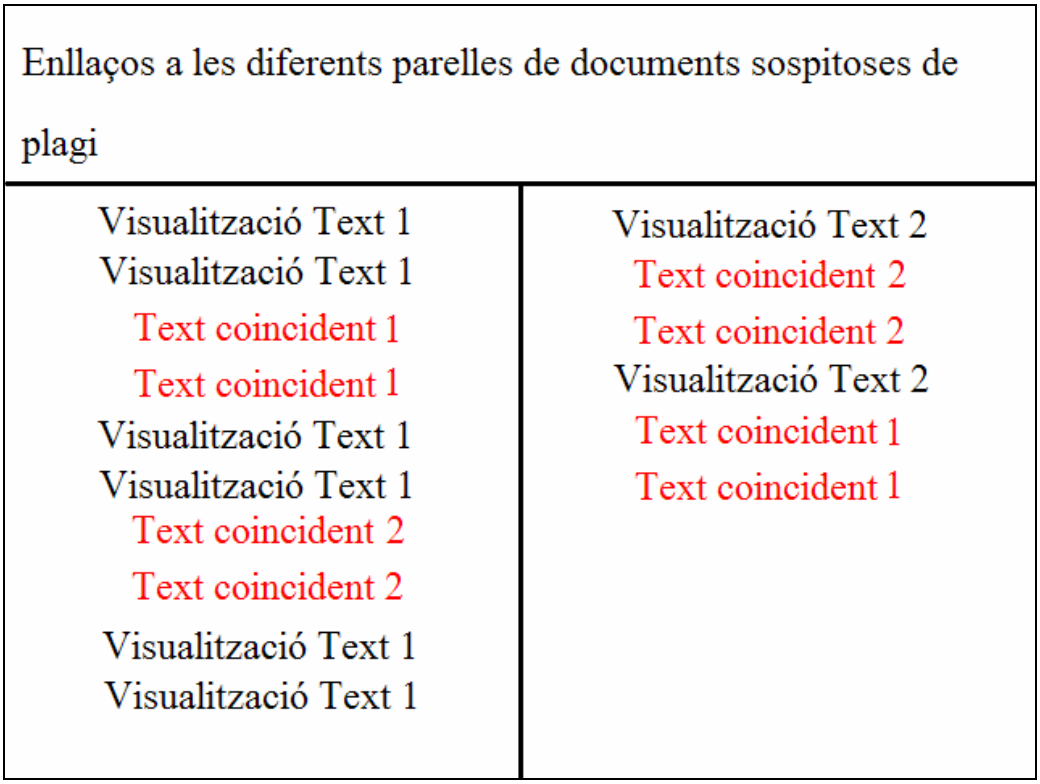


Figura 14 – Esquema de l'estructura del document de sortida

D'aquesta manera, al marc superior hi tindrem un seguit d'enllaços a les diferents parelles de documents. En escollir una determinada parella, es carregaran ambdós documents als marcs inferiors on es mostrarà en negre les parts de text no coincidents i en vermell les coincidents.

9.2. Desenvolupament de l'eina

Tal i com s'ha comentat a l'apartat 5 d'aquest projecte (estudi preliminar), el llenguatge escollit per al desenvolupament ha estat el Java per la seva vastíssima API i pel seu excel·lent tractament dels *strings*.

S'han desenvolupat diverses classes, algunes d'auxiliars (per a poder realitzar filtres del contingut d'un directori, etc.) i d'altres que són les que realment són interessants en aquest projecte. El disseny d'aquestes classes es comentarà en els apartats següents.

9.2.1. Mòdul d'entrada

Tal i com hem vist, el mòdul d'entrada, a part de llegir els fitxers a tractar, també realitza tasques de pretractament de la informació.

Una d'aquestes tasques és l'eliminació de signes de puntuació, accents, dièresis, excés de caràcters blancs, salts de línia i, en definitiva, qualsevol caràcter no alfanumèric que pogués entorpir l'execució de l'algoritme de detecció.

Una altra tasca que realitza el mòdul d'entrada és la conversió del text en *tokens*. Per a fer-ho, s'han creat dues classes: *Token* i *TokenizedFile*.

La classe *Token* conté informació sobre l'estat d'aquest (veure l'apartat 9.2.2. sobre l'algoritme *Greedy-String-Tiling*), informació necessària per al mòdul de sortida (*offset* del *token* dins del fitxer, per exemple) i, per a optimitzar les comparacions entre *tokens* (bàsicament, per no haver de comparar *strings*) un *hash* precalculat del *token*.

La classe *TokenizedFile* conté la conversió completa d'un fitxer d'entrada a *tokens*. Conté, a més a més, informació per relacionar-se amb el fitxer d'entrada (el nom, el *path*, etc.) i el nombre de *tokens* que conté. El constructor d'aquesta classe és qui fa la lectura del fitxer d'entrada i el pretractament. És, doncs, una classe clau en el projecte.

Com es comenta a l'apartat 9.2.2 d'aquest document, en el llenguatge natural existeixen una sèrie de paraules que no aporten informació però que podrien interferir en la detecció del plagi. Per aquest motiu, abans d'incloure un *token* dins del *tokenized file* es compara aquest amb una llista de mots buits. Si el *token* pertany a la llista, no s'inclou dins del *tokenized file*.

Finalment, per a poder representar el fitxer original ressaltant-ne les parts plagiades, cal mantenir, per cada *token* dos indicadors del seu *offset* dins del fitxer: un per a l'*offset* de l'inici del *token* i un altre per al final d'aquest (veure apartat 9.2.3).

A continuació, es mostra un esquema de com interactuen aquestes classes:

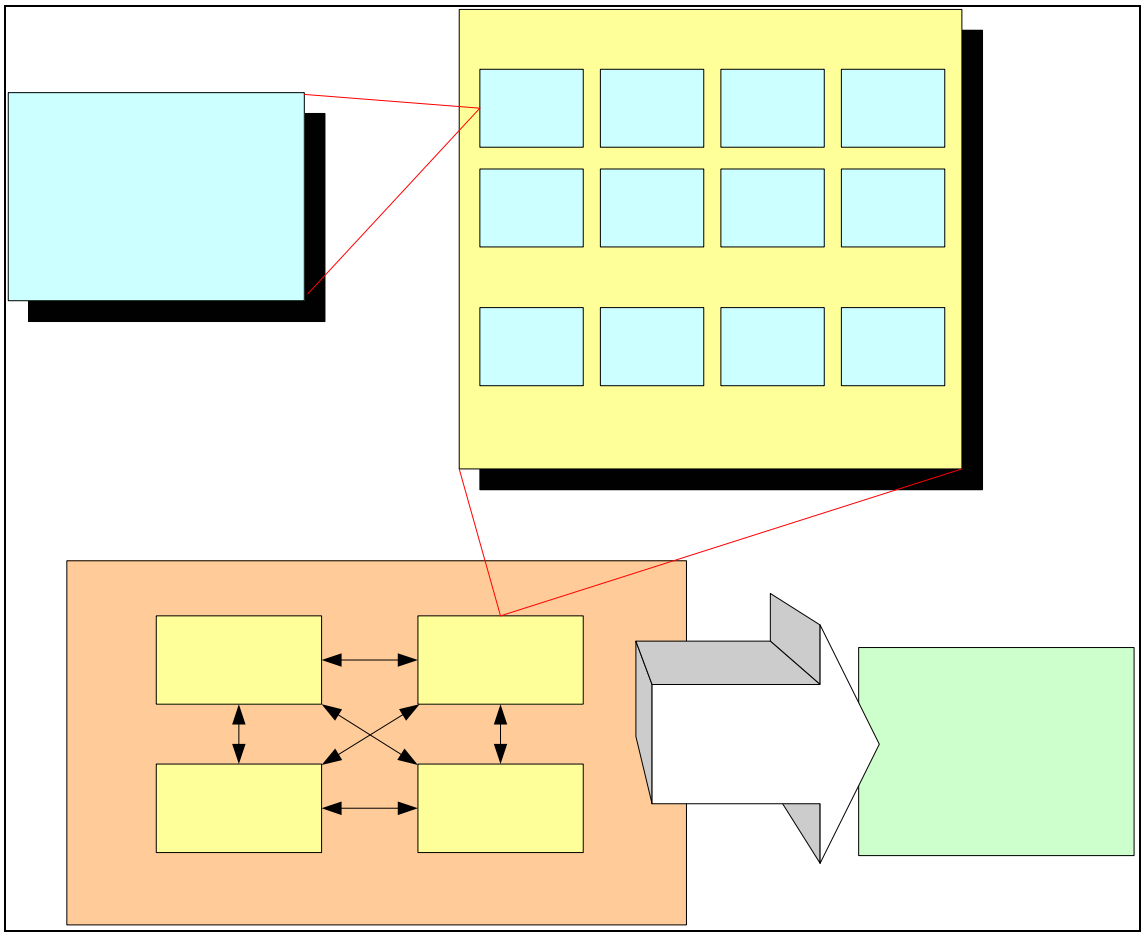


Figura 15 – Esquema de la interacció de les classes

9.2.2. Mòdul de detecció

L'algoritme *Greedy-String-Tiling*

Per a la implementació del mòdul de detecció de plagis calia escollir entre algun dels algoritmes existents a l'actualitat. En el nostre cas, s'ha escollit el *Greedy-String-Tiling* (d'ara endavant, GST) pel fet que és públic i la seva senzilla implementació. Aquest algoritme és el que utilitza el software de detecció de plagis en codi font JPlag (*Bibl. 30*).

El GST és un algoritme que detecta els majors k-grames idèntics possibles entre una parella de documents. D'aquesta manera, abans d'executar l'algoritme, decidirem la grandària mínima que ha de tenir cadascun dels k-grames d'una parella per a ser considerats una coincidència (és obvi que tamanys de k massa petits, produiran coincidències possiblement falses).

Per a poder emprar el GST, primer cal convertir el codi font a *tokens*. Aquests *tokens* podrien ser INICI_CLASSE, DEFINICIO_VARIABLE, ASSIGNACIO, etc. D'aquesta manera, el GST aconsegueix ser independent de les variacions més típiques a les que se sol incórrer per plagiar un codi: canvis de noms de classes, canvis de noms de variables, etc.

L'algoritme GST és el següent:

```
algoritme Greedy-String-Tiling (TokenizedFile A, TokenizedFile B) {
    tiles = {};
    repetir
        maxmatch = MinimumMatchLength;
        matches = {};
        per tots unmarked tokens Aa de A fer
            per tots unmarked tokens Bb de B fer
                j := 0;
                mentre (Aa+j == Bb+j i unmarked(Aa+j) i unmarked(Bb+j))
                    j:=j+1;
                fmentre
                si j==maxmatch llavors
                    matches ⊕ match(a,b,j);
                altrament si j>maxmatch llavors
                    matches = {match(a,b,j);
                fsi
                maxmatch:=j;
            fper
        fper
        per tot match(a,b,maxmatch) ∈ matches fer
            per j de 0 fins a (maxmatch – 1) fer
                marcar(Aa+j);
                marcar(Bb+j);
            pfer
            tiles = tiles ∪ match(a,b,maxmatch);
        fper
    fins que maxmatch > MinimumMatchLength;
    retorna tiles;
falgoritme;
```

Figura 16 – Algoritme *Greedy-String-Tiling*

La filosofia d'aquest algoritme és intentar obtenir els k-grames més grans possibles entre la parella de documents i que aquests siguin sempre més grans que un mínim determinat (*MinimumMaxLength*).

L'algoritme retorna un conjunt de *matchs* anomenat *tiles*. Un *match* no és res més que una tupla que conté l'*offset*[7] de cada *token* dins de cada *tokenized file* i una llargada màxima de la coincidència.

L'algoritme també ens garanteix que un determinat *token* només pot formar part d'un sol *match* i que, a més a més, si un *token* fos susceptible de formar part de dos *matchs* diferents, l'algoritme ens retornarà sempre el major d'ells dos. Aquesta part es fa marcant els *tokens* utilitzats i comprovant al bucle principal que els que intentem incorporar a un *match* no estiguin ja marcats.

Adaptació de l'algoritme al llenguatge natural

Com podem comprovar, cal adaptar aquest algoritme a la detecció en llenguatge natural (*Bibl. 33*). Per a fer-ho s'ha fet l'equivalència paraula-*token* on cada paraula diferent es converteix en un *token* diferent.

Ara bé, en el llenguatge natural hi ha un seguit de paraules que no aporten, en realitat, informació i que podrien produir-nos variacions dins d'un text que fessin que l'algoritme no detectés com a iguals dos determinats k-grames. Igualment, els idiomes llatins incorporen tota una sèrie de signes de puntuació que podrien confondre l'algoritme.

És per aquests dos motius que cal fer el pretractament que hem vist a l'apartat 9.2.1 d'aquest document.

No obstant, aquest pretractament no és suficient. Cal tenir en compte que la majoria de treballs lliurats per alumnes contenen una còpia de l'enunciat. Aquest enunciat sovint és més extens que les pròpies respostes. És evident, doncs, que si no tenim en compte aquest fet, detectarem falsos indicis de plagi que, malgrat un observador humà els descartarà de seguida, dificultaran la tasca d'aquest.

Per aquest motiu, el primer que fa el mòdul de detecció és "restar" l'enunciat de la llista de *tokens* a analitzar. L'algoritme de resta és, conceptualment, molt senzill:

```
Convertir enunciat a TokenizedFile
Per a cada TokenizedFile d'entrada i <> Enunciat fer
       Aplicar GreedyStringTiling (i,Enunciat)
       Marcar Tokens de i que hagin fet match com a no tractables
FPer
```

Figura 17 – Algoritme emprat per a restar l'enunciat

[7] Cal anar amb compte en no confondre aquest *offset* amb l'*offset* del *token* dins del fitxer original. Són dos conceptes totalment diferents. En el cas de l'algoritme és l'*offset* del *token* dins de la llista de *tokens* a analitzar. En el segon cas, estem parlant de la posició real en bytes de l'aparició del *token* dins del fitxer i és únicament necessària per a poder fer l'informe final.

Per tal de no haver de modificar l'algoritme GST el que s'ha fet ha estat modificar únicament el mètode ***getUnmarkedTokens*** de la classe *TokenizedFile* per tal que retorni com a no marcats únicament aquells que no han coincidit amb l'enunciat.

9.2.3. Mòdul de sortida

La sortida del programa és en format HTML. Tal i com s'ha vist a l'apartat 9.1.3, la sortida conté 3 marcs. En el marc superior hi ha els enllaços a les diferents parelles de documents que són susceptibles de contenir un plagi entre ells. A la part inferior, s'hi mostren els dos documents seleccionats indicant en negre el text no coincident i en vermell el text coincident.

Per a poder mostrar aquests marcs es generen una sèrie de fitxers .html.

- ✓ document principal: s'anomena amb la data i hora del sistema. És el document que ha de visualitzar l'usuari.
- ✓ document amb els enllaços: s'anomena **main.html**
- ✓ document amb la visualització d'un fitxer: s'anomenen seqüencialment segons l'ordre de generació: **file0.html, file1.html**, etc.

S'ha contemplat la possibilitat que l'usuari vulgui realitzar diferents comparacions amb diferents paràmetres (per exemple, amb diferents valors de longitud dels k-grames). Per aquest motiu, cada comparació genera un resultat que es registra en un directori diferent.

Els directoris generats s'anomenen mitjançant el dia i l'hora del moment de l'inici de la comparació. Així, per exemple, si la comparació s'inicia el dia 25/12/2005 a les 18:30:00 el directori generat serà **20051225 183000**. Igualment, es generarà un fitxer .html al mateix directori des d'on s'hagi executat l'aplicació amb el mateix nom (en l'exemple **20051225 183000.html**).

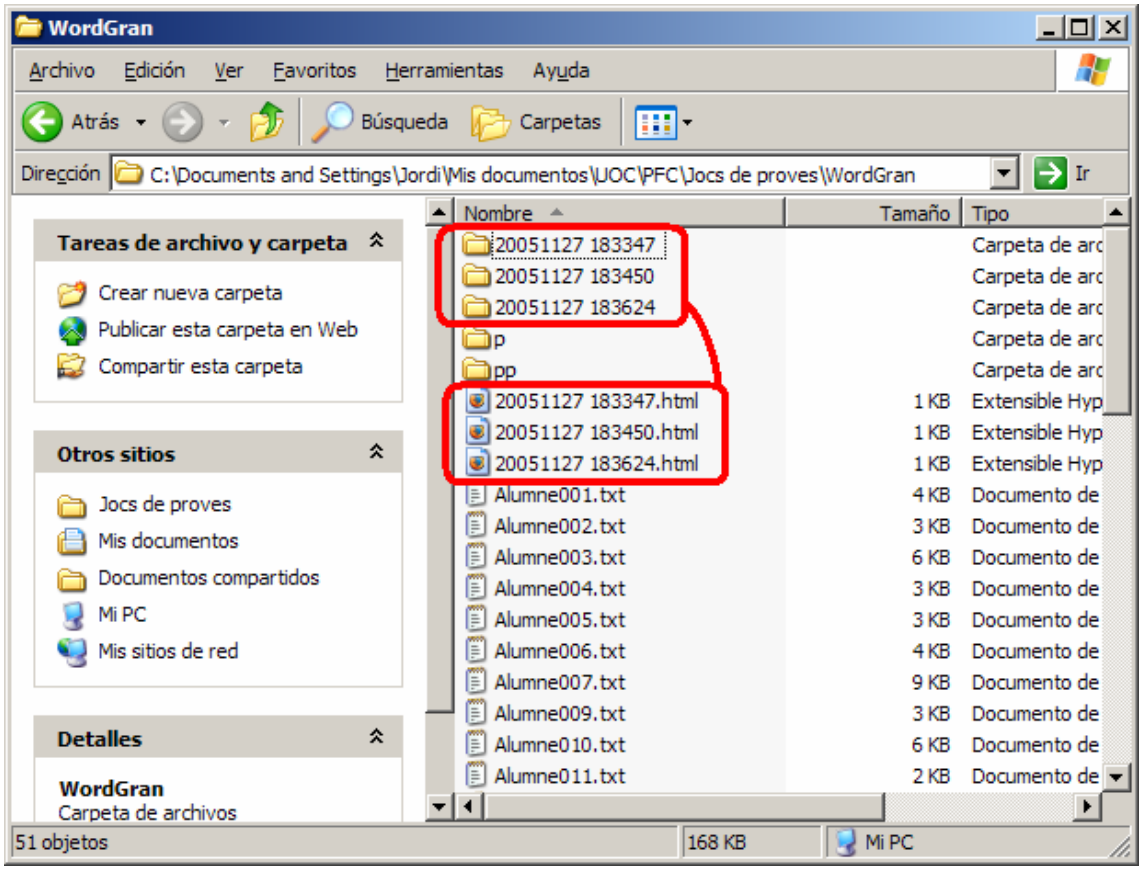


Figura 18 – Directoris i fitxers principals resultats de l'execució de l'analitzador

El principal problema de la realització del mòdul de sortida ha estat que tota l'execució es fa sobre *tokens* preprocessats mentre que la visualització cal fer-la sobre el fitxer original (recordem que s'han eliminat paraules, s'ha convertit tot el text a majúscules, s'han eliminat accents, etc.)

Per a solucionar aquest problema, la classe *Token* registra la posició (*offset*) de l'inici d'aquest dins del fitxer així com la posició del final. Amb aquesta informació, s'ha pogut aplicar el següent algoritme per a generar la visualització:

```
posicio_anterior := 1
per j de 1 fins a total_matchs fer
    per i de posicio_anterior fins a posicio_anterior + longitud(match) fer
        escriure_cararcter_en_negre(fitxer[i);
    fper
    per i de offset_ini_primer_token(match[j]) fins offset_fi_ultim_token(match[j]) fer
        escriure_caracter_en_vermell(fitxer[i]);
    fper
    posicio_anterior := i;
fper
```

Òbviament, les escriptures als fitxers ha calgut realitzar-les tenint en compte que havien de seguir l'estàndar HTML i s'han afegit els corresponents *tags* convenientment.

9.3. Execució

Per a executar l'aplicació, primer haurem de convertir els documents de Word a format pla de text. Això ho farem fent servir el fitxer **convert.bat** proporcionat:

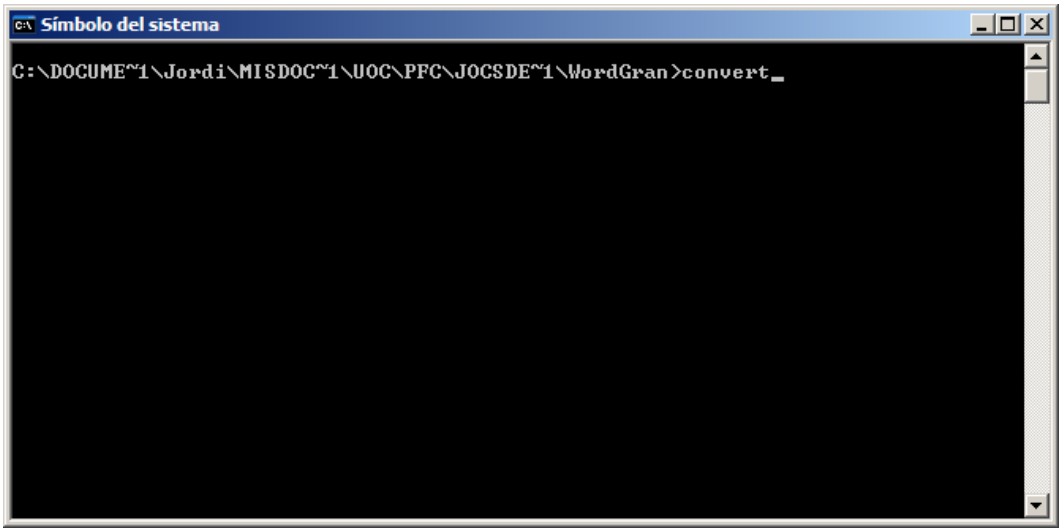


Figura 19 – Captura de pantalla: comanda per a convertir els fitxers de .doc a .txt

L'execució d'aquest fitxer (el que fa és un bucle que crida a l'Antiword per convertir cada fitxer) dóna un missatge d'error per cada conversió. S'han consultat diverses fonts a Internet per a poder eliminar aquest missatge sense èxit. No obstant, la conversió es realitza sense problemes:

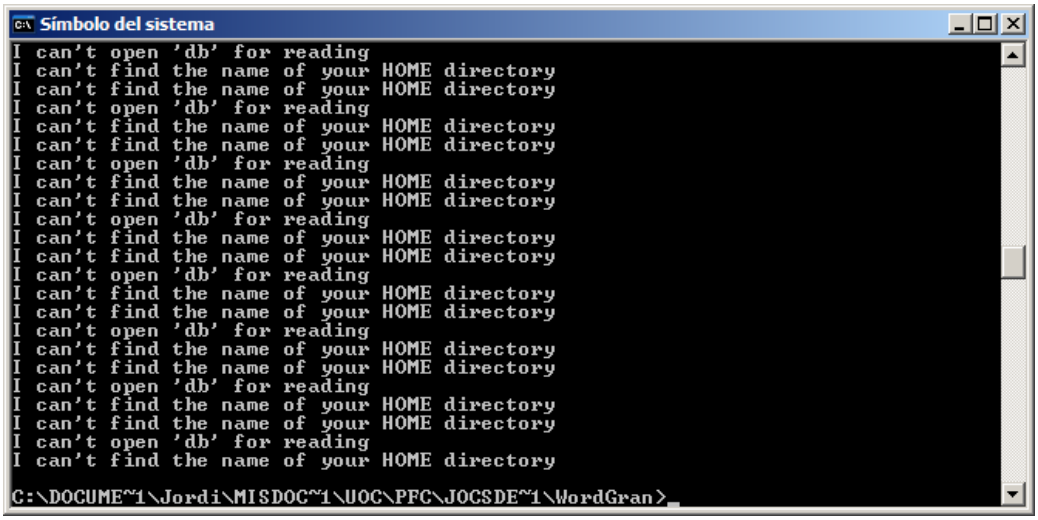


Figura 20 – Captura de pantalla: conversió de .doc a .txt

Un cop tenim els fitxers convertits a text, executarem l'eina d'anàlisi de plagis. Això ho farem mitjançant el comandament **java Analitzador**. L'eina disposa d'una sèrie de paràmetres:

- ✓ **-e enunciat:** aquest paràmetre és opcional. Indica que el fitxer **enunciat** conté l'enunciat dels exercicis a comparar. Activant aquesta opció, fem que l'analitzador "resti" l'enunciat per a evitar falsos positius.
- ✓ **-k longitud:** longitud mínima de coincidència de dos k-grames per a ser considerats susceptibles de plagi. Si s'obvia aquest paràmetre, la longitud per defecte és 50.
- ✓ **-d directori:** directori on es troben els fitxers. Si s'obvia aquest paràmetre, el programa tracta els fitxers que hi hagi al directori des d'on s'executa.

A continuació, es mostra un exemple d'execució, amb una longitud mínima de 20 i amb un enunciat contingut al fitxer **Enunciat.txt**.

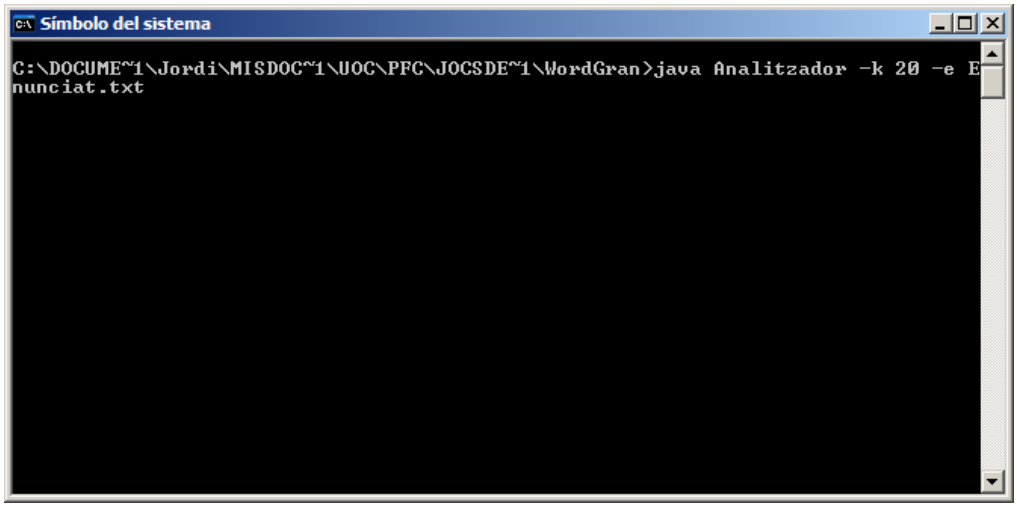


Figura 21 – Captura de pantalla: crida a l'aplicació d'anàlisi

L'execució passa per quatre fases:

Primer es llegeix el diccionari de mots buits que són aquells que no es tindran en compte a l'hora de fer la comparació.

Tot seguit es llegeixen els fitxers, es pretracten i es converteixen a *tokens*.

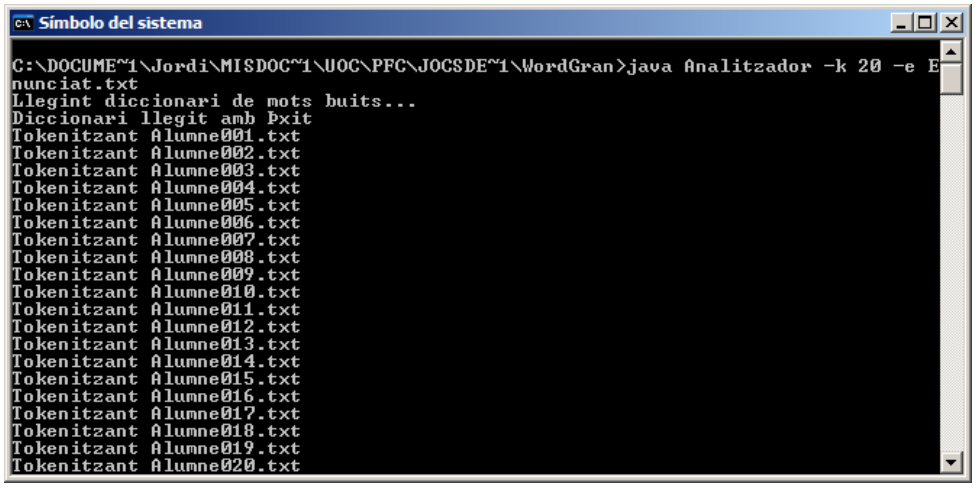


Figura 22 – Captura de pantalla: conversió dels fitxers a *tokens*

Un cop convertits els fitxers a *tokens*, se'ls resta l'enunciat, si n'hi ha:

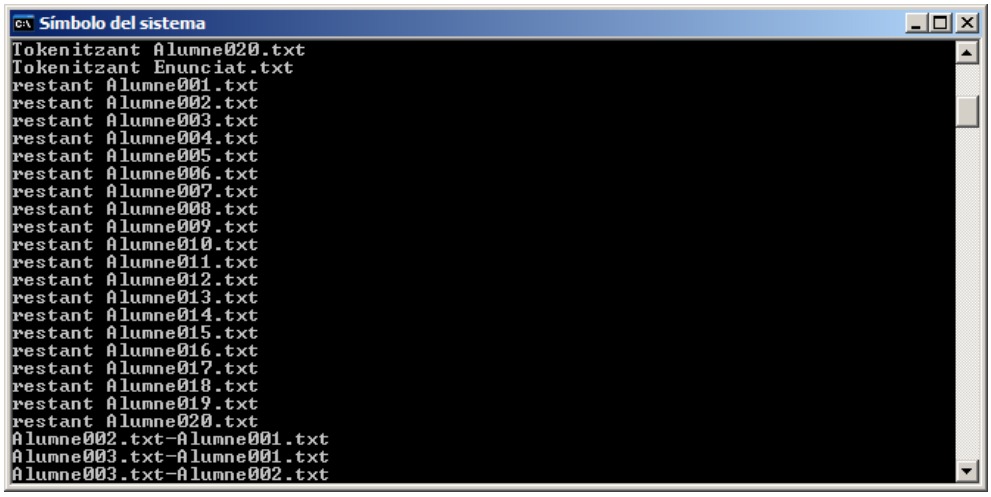


Figura 23 – Captura de pantalla: resta de l'enunciat de cadascun dels fitxers origen

Finalment, un cop restat l'enunciat, comença la comparació per parelles:

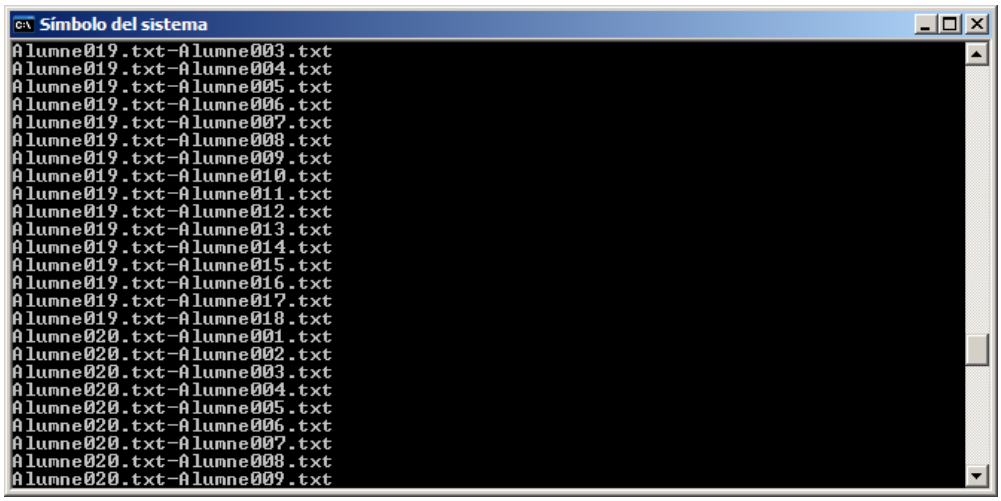


Figura 24 – Captura de pantalla: procés de comparació dels fitxers per parelles

Un cop finalitzada l'execució, podrem observar com se'ns ha generat un fitxer .html el nom del qual es correspon a la data i hora de l'inici de l'execució:

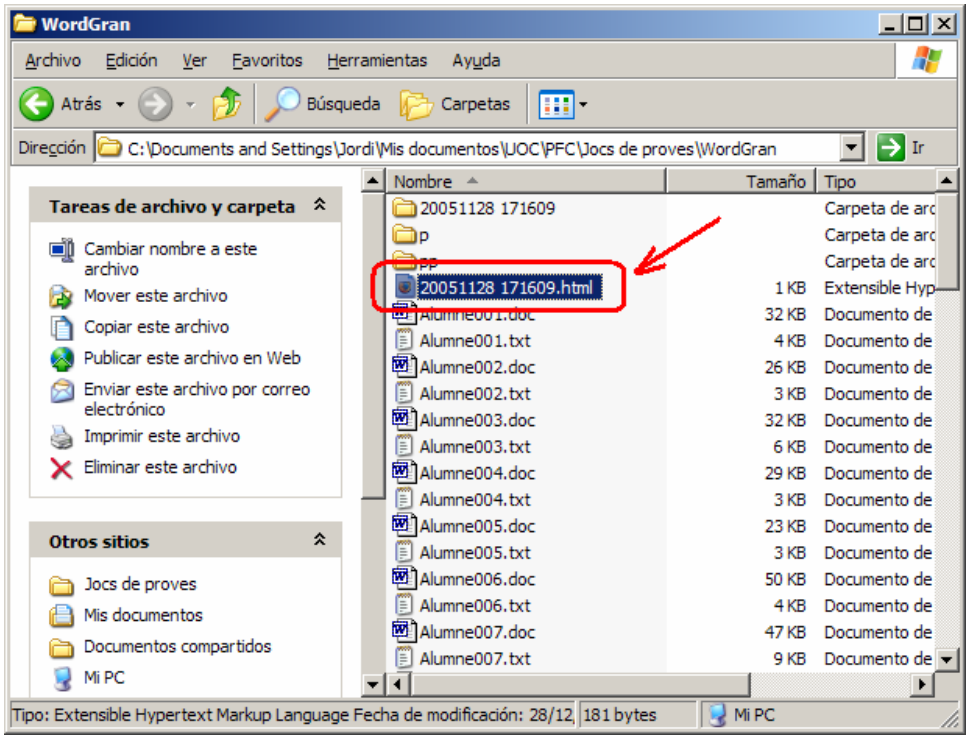


Figura 25 – Captura de pantalla: fitxer .html generat com a resultat de l'anàlisi

Si obrim aquest fitxer amb un navegador, obtindrem una pantalla com la que segueix:

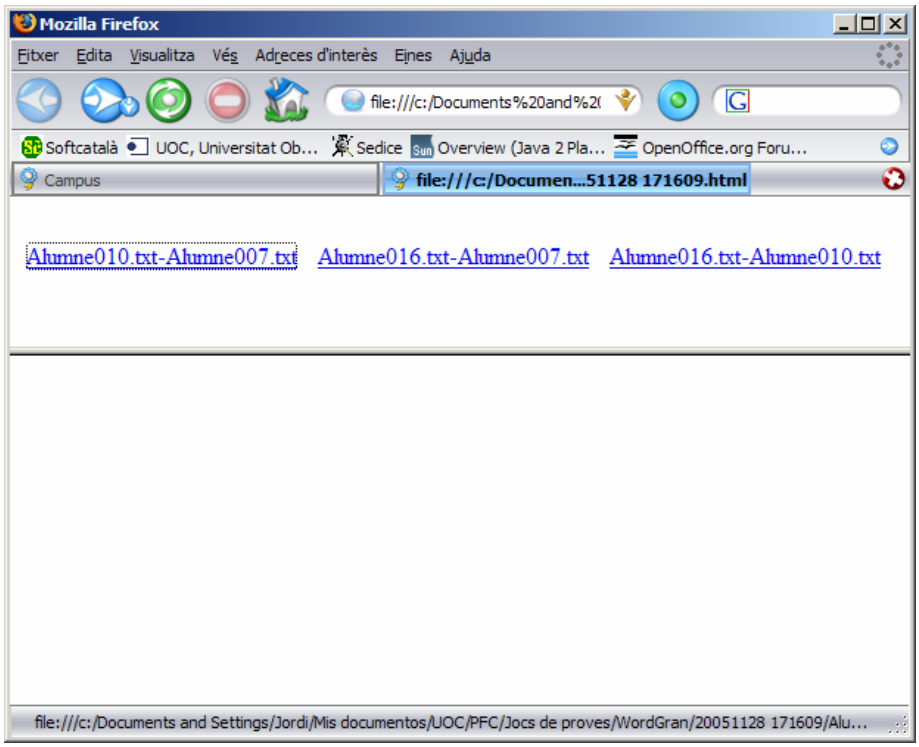


Figura 26 – Captura de pantalla: fitxer resultat de la comparació

Tot seguit, si fem clic sobre algun dels enllaços, podrem observar la comparació a la part inferior:

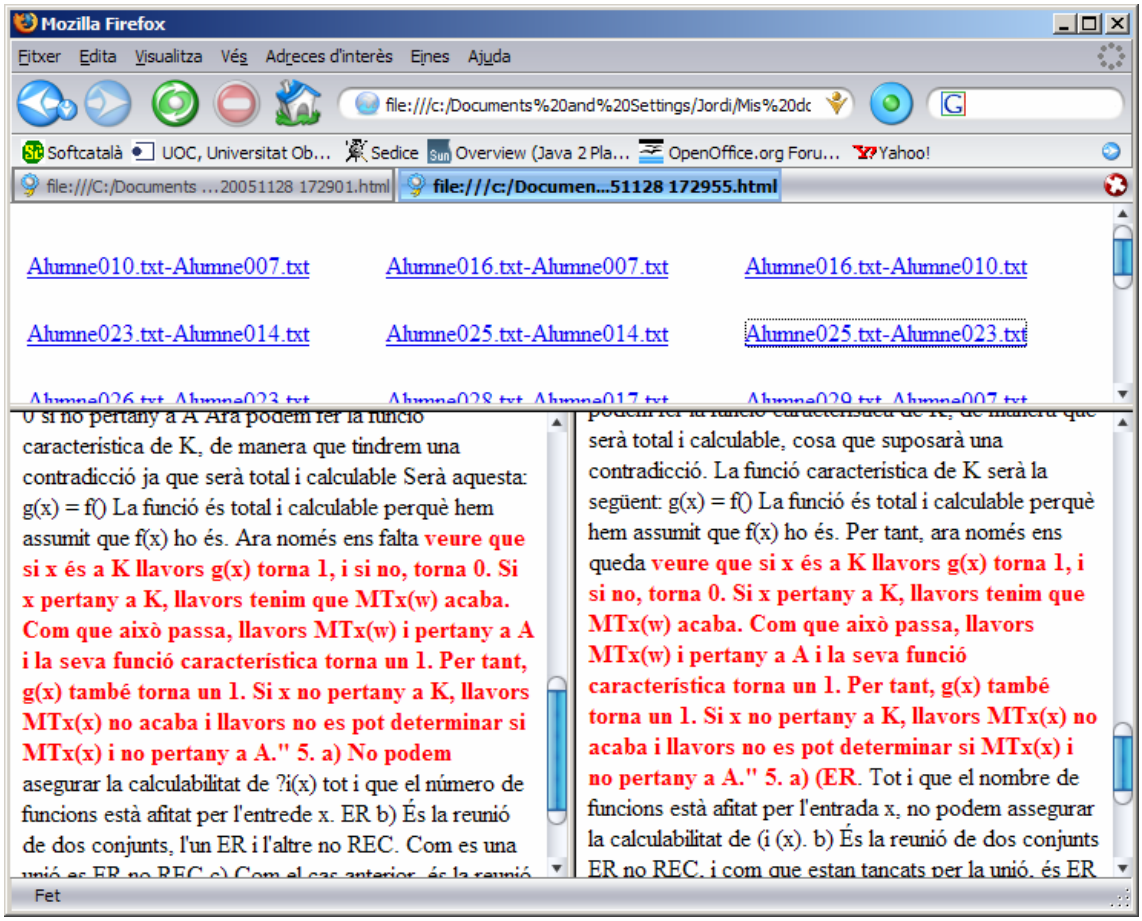


Figura 27 – Captura de pantalla: comparació de dos documents analitzats

9.4. Problemes coneguts

La manca de més temps ha fet que hagin quedat per resoldre alguns problemes malgrat ser coneguts:

- ✓ L'algoritme *Greedy-String-Tiling* només detecta còpies flagrants. Qualsevol canvi d'una sola paraula d'un k-grama fa que aquest ja no sigui detectat com a plagi.
- ✓ Degut a desplaçaments provocats per la lectura de caràcters de control als fitxers d'entrada original, es pot produir un petit decalatge entre dels *offset* dels *tokens* fent que a la sortida es visualitzin alguns caràcters com a positius sense ser-ho o alguns com a negatius sense ser-ho.

9.5. Anàlisi del rendiment

Per tal de comprovar el rendiment de l'eina se li han fet dues proves d'estrès diferents. La primera ha consistit en fer-li analitzar diferents còpies d'un mateix document molt extens (més de 20.000 paraules) i la segona en fer-li analitzar un gran nombre de còpies d'un fitxer més petit (menys de 500 paraules).

Les proves s'han realitzat sobre un ordinador PC amb les següents característiques:

- ✓ Processador AMD Athlon 2.2 GHz
- ✓ 1 Gb de RAM
- ✓ Windows XP Home Edition

Per a la comparació amb fitxers grans, els resultats obtinguts han estat:

Nombre Docs	Temps (minuts)	Temps (segons)	Nombre de Comparacions
2	1:05	65	1
3	2:41	161	3
4	5:08	308	6
5	8:26	506	10
10	37:48	2268	45

Figura 28 – Taula comparativa dels temps d'execució per a diferents nombres de còpies de fitxers grans (20.000 paraules)

Com podem observar, els fitxers grans penalitzen bastant el temps d'execució. Això és perquè el *Greedy-String-Tiling* té cost cúbic respecte el tamany de l'entrada. A continuació, es mostra un gràfic amb els temps d'execució:

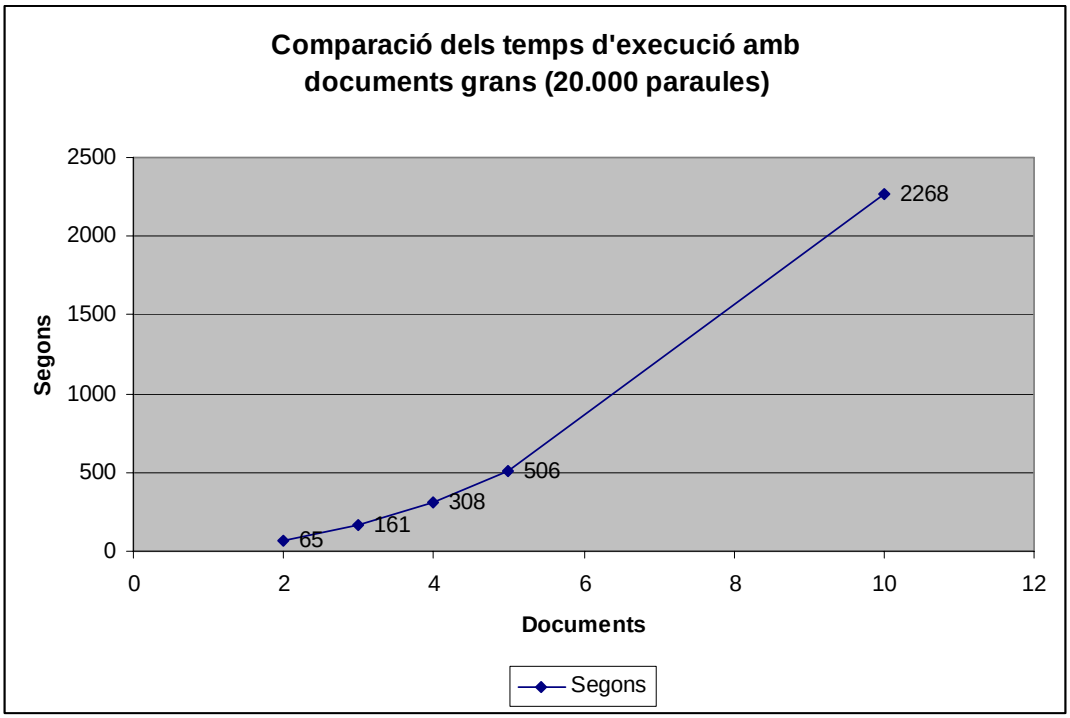


Figura 29 – Gràfic amb els temps d'execució per a la comparació de documents grans (20.000 paraules)

Per a la comparació amb fitxers petits, els resultats obtinguts han estat:

Nombre Docs	Temps (minuts)	Temps (segons)	Nombre de Comparacions
16	0:02	2	120
32	0:10	10	496
48	0:24	24	1128
96	1:41	101	4560
192	7:01	421	18836

Figura 30 – Taula comparativa dels temps d'execució per a diferents nombres de còpies de fitxers petits (500 paraules)

Com es pot comprovar, el sistema és força menys crític en temps d'execució per a fitxers petits. Podem arribar a comparar-ne prop de 200 en menys de 10 minuts.

A continuació, es mostra el gràfic amb els temps d'execució:

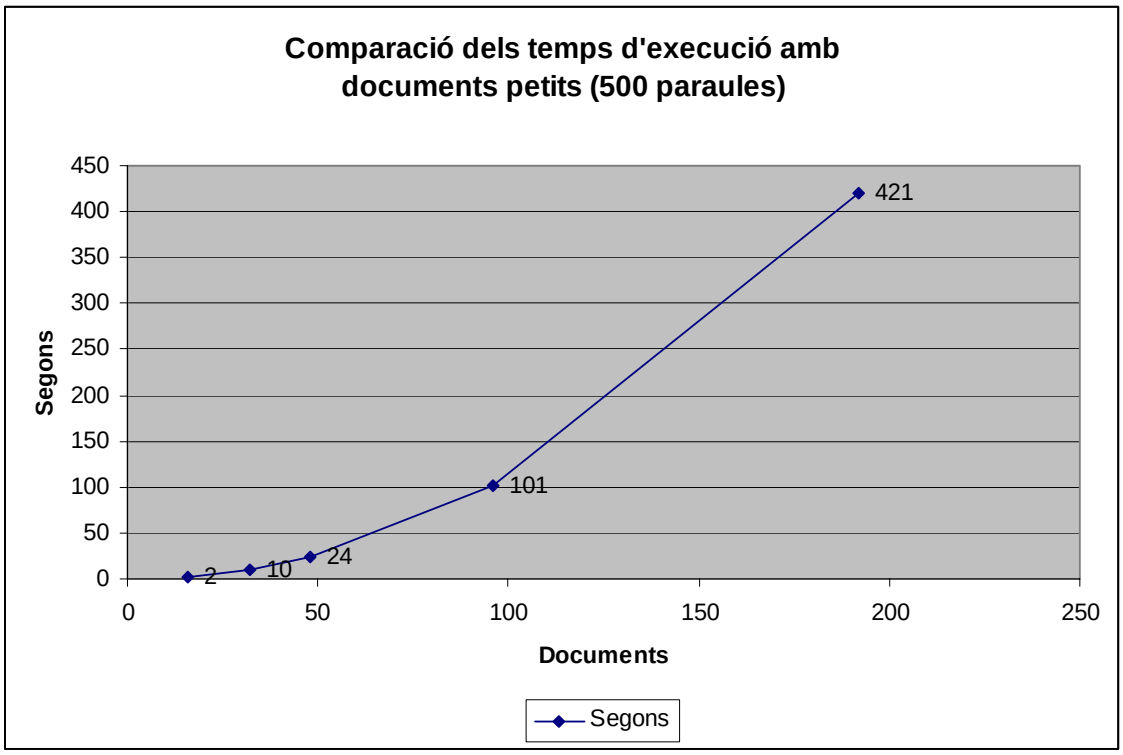


Figura 31 – Gràfic amb els temps d'execució per a la comparació de documents petits (500 paraules)

Per l'experiència de l'autor, una solució a una PAC real de la UOC conté entre 1000 i 2000 paraules. Ens trobaríem, doncs, prop de la situació de la figura 30. Així doncs, creiem que en una avaluació real a la recerca de plagi en una situació d'uns 300 estudiants amb PAC de mida mitjana, el temps de recerca no hauria de superar els 30 minuts. Veiem doncs, com en un temps raonable, es podria avaluar la plagiabilitat dels exercicis lliurats.

9.6. Línies futures

Hi ha diverses millores que es podrien aplicar al desenvolupament fet que no s'han implementat per limitacions de temps. Algunes d'elles podrien ser:

- ✓ **Millores en el rendiment de l'aplicació en temps** fent una anàlisi detallada dels punts on itera l'algoritme (bàsicament, l'algoritme de comparació) i estudiant diferents sistemes per a optimitzar-ne cada iteració. Una vegada realitzada aquesta anàlisi, seria bastant senzill dur-lo a terme gràcies a la modularitat de l'eina. Segurament, caldria optimitzar el procés de comparació de *tokens* de manera que només necessitaríem revisar els mètodes que fem servir per a aquesta comparació.
- ✓ **Millores en la gestió de memòria** carregant únicament allò que necessitem i deixant la resta a disc. Aquesta gestió, però, va frontalment en contra amb l'optimització en temps. Aquesta revisió implicaria canviar profundament l'eina i afectaria tant a la classe *Analitzador* com a la classe *TokenizedFile*.
- ✓ **Utilització d'algoritmes diferents al *Greedy-String-Tiling*.** Es podria intentar utilitzar un algoritme més orientat al llenguatge natural en comptes de fer-ne servir un orientat a codi. Aquesta modificació seria més o menys complexa depenent de si el nou algoritme també retornés *tiles* (coincidències que es poguessin emmagatzemar a la classe *Match* o no). En el cas que l'algoritme retornés *tiles*, n'hi hauria prou en revisar l'algoritme i procurar que els retornés en el mateix format que l'actual. En cas que l'algoritme fos radicalment diferent i no retornés *tiles*, caldria revisar profundament tota l'aplicació.
- ✓ **Revisió de sinòmins:** es podria mantenir un diccionari en memòria de sinònims i en el moment de fer la comparació, no només comparar cada *token* sinó els seus sinòmins per evitar l'ús d'aquests per simular un possible plagi. Aquesta modificació podria ser relativament senzilla de portar a terme si no tenim en compte com afectaria al rendiment. Només caldria modificar el mètode de comparació de la classe *Token* per comparar també els sinòmins dels *tokens*. Fer que aquesta revisió no fos excessivament costosa en temps, podria ser molt difícil d'aconseguir.
- ✓ **Revisió de les metadades:** com s'ha comentat, finalment no s'han emprat les metadades contingudes als fitxers de Word. No obstant, aquestes continuen oferint una bona font d'informació davant del plagi i podrien ser utilitzades.
- ✓ **Evitar l'ús de l'Antiword:** és a dir, importar directament els fitxers .doc de Microsoft Word. Per a dur a terme aquesta millora, caldria cercar (o desenvolupar) unes llibreries que permetessin obrir i llegir documents de Word. Si es pogués disposar d'aquestes llibreries, la modificació de l'eina seria tan senzilla com revisar el constructor de la classe *TokenizedFile*. No obstant, si no es disposés d'aquestes llibreries en Java (és segur que existeixen en .NET), caldria codificar de nou tota l'aplicació.

10. Instal·lació del software

10.1. Instal·lació de l'Antiword

Tal i com hem vist a l'apartat on parlàvem de les diferents eines existents per obrir documents Word, s'ha optat finalment per l'eina Antiword com a eina auxiliar que ens permeti fer una conversió prèvia dels fitxers a un format llegible.

Per a instal·lar Antiword, el primer que hem de fer és baixar-lo de la web de l'autor:

http://www.winfield.demon.nl/

Un cop dins la web, accedirem a la secció on hi hagi el programa per la plataforma que vulguem fer servir. En el nostre cas, a mode d'exemple, s'ha escollit la plataforma Windows, per ser la més utilitzada.

La instal·lació és molt senzilla. Un cop baixat el fitxer .zip corresponent (en el moment de redactar aquest document és **antiword-0_36-windows.zip**) el descomprimirem al directori arrel del disc dur. Automàticament, se'ns crearà un directori anomenat **c:\antiword**.

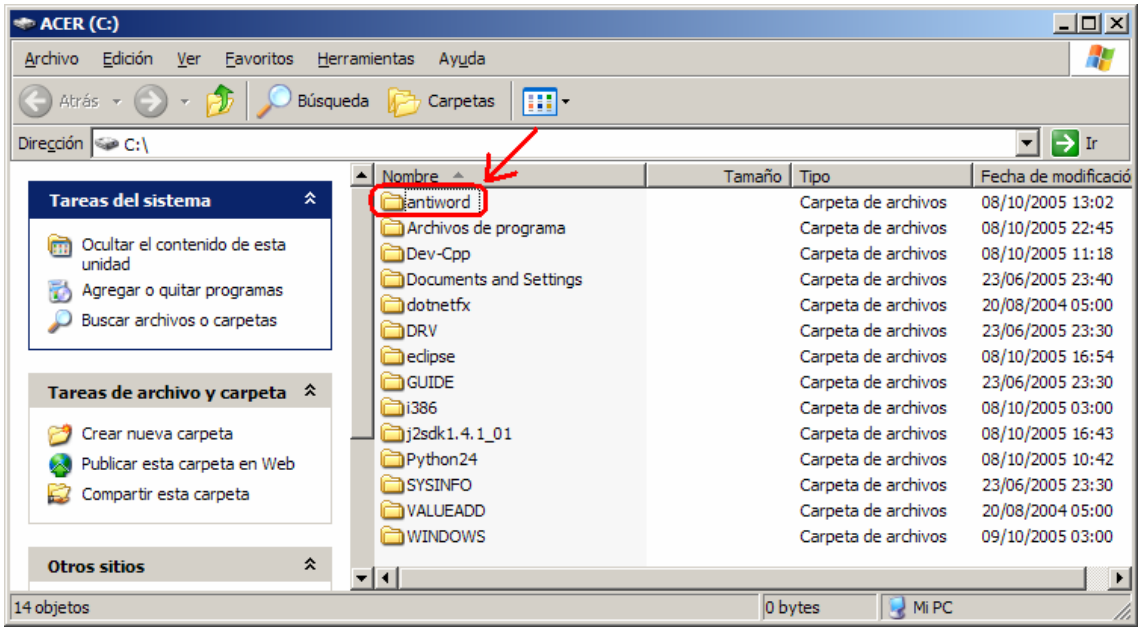


Figura 32 – Directori d'instal·lació del programa Antiword

És important respectar aquest directori de destí per a la instal·lació. S'ha comprovat que Antiword dóna algun problema de funcionament si no es troba instal·lat en aquest directori.

Tot seguit, modificarem el PATH del sistema per tal que contingui aquest directori.

10.2. Compilació de l'aplicació de detecció de plagis

Per tal de compilar l'aplicació, cal guardar en un mateix directori els següents fitxers .java:

- ✓ Analitzador.java
- ✓ Filtre.java
- ✓ Match.java
- ✓ Token.java
- ✓ TokenizedFile.java

Per a compilar-los n'hi ha prou amb executar des de la línia de comandes la sentència **javac *.java**. S'obtindran els següents fitxers:

- ✓ Analitzador.class
- ✓ Filtre.class
- ✓ Match.class
- ✓ Token.class
- ✓ TokenizedFile.class

Cal tenir en compte que el compilador de Java (javac) ha d'estar inclòs al *path* de l'ordinador des d'on es vulgui compilar així com que els *packages* **java.io** i **java.util** es trobin en algun dels directoris apuntats pel *classpath*.

10.3. Conversor massiu de .doc a .txt

Per convertir els fitxers origen en fitxers plans .txt per a poder ser tractats cal copiar els dos fitxers .bat que acompanyen aquesta documentació anomenats **convert.bat** i **dir-se.bat**. Ambdós fitxers han d'estar al mateix directori on hi hagi els fitxers .doc a convertir.

10.4. Llista de mots buits

Per tal de poder obviar aquells mots que no aporten informació, cal que existeixi al mateix directori on hi ha els fitxers a convertir, un fitxer anomenat **buits.lst** que contingui la llista de mots buits de l'idioma amb el qual estiguin escrits els documents originals.

Adjunt a aquesta documentació, es proporciona un fitxer **buits.lst** amb els mots buits del català.

10.5. Execució de l'analitzador

Per a executar l'analitzador, simplement cal iniciar la màquina virtual de Java en un directori on hi hagi les cinc classes (fitxers .class) obtinguts durant la compilació (veure apartat 9.3, sobre la compilació).

11. Conclusions

Una vegada finalitzat el projecte, podem arribar a una sèrie de conclusions pel que fa a l'abast de l'eina, a l'efectivitat de la lluita antiplagi i al propi desenvolupament.

11.1. Abast de l'eina de detecció

Durant la realització d'aquest projecte s'ha pogut anar prenent consciència de la problemàtica del plagi en l'àmbit universitari i, en especial, en un entorn d'ensenyament virtual com és la UOC, tal i com ho hem pogut constatar en l'estudi de les diferents eines existents a l'actualitat (apartat 6).

Malgrat que l'objectiu del projecte era obtenir una eina per a evitar els plagis, la primera conclusió –i la més important, segurament– a que s'ha arribat és que no és possible evitar-los ni detectar-los al 100%.

Són diversos els factors que ens han portat a aquesta conclusió. Per una banda, l'estudi de la problemàtica de tractar amb llenguatge natural, ens ha fet adonar que una eina infal·lible en la detecció de plagis requeriria d'una complexitat en el seu desenvolupament molt gran. Un autor malintencionat, coneixedor d'una eina de detecció pot emprar diferents tècniques per a burlar-la: ús de sinònmins, alteració de l'ordre de les paraules i/o de les frases o, fins i tot, la inclusió intencionada de faltes d'ortografia, de manera que, per exemple, *pla contable* es converteixi en una expressió totalment diferent –des del punt de vista del tractament automàtic– a *pla comptable*.

Si bé és cert que podem anar millorant i afinant l'eina de detecció per tal d'incloure-hi aquesta "base de dades de coneixement" sobre possibles trampes, també ho és que hi ha un factor que no podem menysprear: la complexitat algorítmica de l'anàlisi.

Per tant, no creiem que tingui sentit, incrementar indefinidament una eina, dedicant-hi esforços en temps i recursos humans al seu desenvolupament, augmentant-ne indefinidament el temps d'execució per a poder-ho *detectar tot*. Creiem que el millor és arribar a un equilibri i dedicar els esforços a la detecció d'allò que realment sigui habitual.

Pel que fa al rendiment obtingut pel prototipus, creiem que és molt bo (veure apartat 9.5 sobre el rendiment) i s'adequa a les necessitats (de volum i tamany dels fitxers) de la UOC.

Malgrat que, en un primer moment, com es mencionava a l'apartat 5.2., semblava que Java no havia de ser el llenguatge idoni per a la realització d'una eina on el temps d'execució ja sabíem, d'entrada, que seria crític, cal dir que el llenguatge escollit no ha estat, en cap moment, un problema. Encara que Java sigui un llenguatge interpretat, la seva API disposa de nombroses classes altament optimitzades que s'han pogut utilitzar.

11.2. Prevenció + detecció: la clau de l'èxit

Significa el que hem vist a l'apartat anterior que la detecció parcial és tot el que podem fer contra el plagi? La resposta és un contundent **no**.

El desenvolupament d'una eina de detecció no hauria de ser res més que el primer graó. Les vulnerabilitats de l'eina és quelcom que s'ha d'amagar tant com sigui possible. Al contrari, però, cal donar-la a conèixer, remarcar-ne les virtuts. D'aquesta manera, estarem fent política de prevenció. El fet que els estudiants de la UOC coneguin l'existència d'una eina de detecció evitarà, per si mateix, molts intents de còpia o plagi per part dels alumnes. En aquest sentit, ens atreviríem a dir, que aquest és el punt més fort de l'eina.

Així mateix, continuant amb aquesta política de prevenció, també es poden lliurar els enunciats als alumnes fent servir els sistemes de registre d'autoria explicats a l'apartat 8 d'aquest document. Novament, però, estarem davant de solucions tècnicament parcials ja que, com hem vist, aquestes també poden ser burlades pels estudiants. No obstant, el seu coneixement, juntament amb el coneixement de l'existència de l'eina de detecció de plagis pot fer renunciar a l'intent de còpia als estudiants.

Alguns dels "atacs" més simples que poden intentar dur a terme els estudiants sí que són, però, detectats per l'eina i no interfereixen en el seu funcionament:

- ✓ afegir tabuladors innecessaris
- ✓ afegir línies en blanc
- ✓ faltes d'ortografia realitzades expressament afegint o traient accents
- ✓ afegir espais en blanc
- ✓ canviar majúscules i minúscules

11.3. Sobre el desenvolupament

L'eina desenvolupada durant la realització d'aquest projecte és poc més que un prototipus del que podria –hauria– d'arribar a ser. El temps ha estat, en aquest sentit, un factor clau.

Si s'hagués de tornar a desenvolupar aquest projecte hi hauria diversos punts que podrien fer-se diferents i/o millorar-se. Un d'aquests seria l'algoritme de detecció escollit: el Greedy String Tiling.

Com he vist als apartats 9.1.2 i 9.2.2, aquest algoritme té el gran inconvenient que està orientat a la detecció de plagis de codi font. Per a fer-lo servir en la detecció en llenguatge natural, hem hagut de fer l'"assimilació" paraula=*token* que s'ha convertit en el punt feble de l'eina desenvolupada. Aquesta assimilació fa que l'eina tingui els següents punts febles (tots ells íntimament relacionats):

- És sensible a les incorreccions ortogràfiques.
- És sensible a les variacions sinonímiques (un estudiant en faria prou en canviar diverses paraules per d'altres de sinònimes i el sistema ignoraria els k-grames que les continguessin)
- És sensible a les variacions d'ordre de les paraules dins d'un k-grama.

12. Glossari

- **Autor**: qualsevol persona que modifica part d'un document
- **Coincidència**: part d'un document que s'assembla a una part d'un altre més del que determina un determinat llindar.
- **Control de canvis**: funcionalitat de Microsoft Word que permet visualitzar els canvis entre diferents versions d'un mateix document tot assenyalant-los en diferents colors.
- **Eina de detecció de plagis**: sistema automàtic d'avaluació de documents capaç de descobrir possibles plagis parcials o totals realitzats en la redacció d'un document.
- **Estil**: en un document Word, és un conjunt de característiques de format emmagatzemades sota un nom que serveix per identificar-les. Mitjançant estils poden aplicar-se, de cop, totes les característiques a un text prèviament seleccionat.
- **Índex de concordança**: valor numèric que ens indica el grau de coincidència entre dos fragments de dos documents.
- **Llenguatge natural**: llenguatge amb que ens expressem els humans. Es diferencia d'altres tipus de llenguatge, com per exemple, els llenguatges de programació en el fet que permet més variacions. L'anàlisi del plagi és més complicat en llenguatge natural que en altres tipus de llenguatge.
- **Macro**: conjunt de comandes i instruccions que s'agrupen en una sola comanda de forma que una determinada tasca pugui fer-se automàticament.
- **Mapa de col·lisió**: representació visual de l'índex de la coincidència entre dos documents mitjançant un mapa bidimensional on s'indiquen en un color fosc les parts que coincideixen i en un color clar les parts que no ho fan.
- **Metadades**: en un document de Word són el conjunt d'informacions, diferents del propi contingut del document, relatives a la seva edició: nombre de pàgines, autor, data de l'última actualització, etc.
- **Open Office**: versió de programari lliure del paquet ofimàtic Star Office propietat de Sun Microsystems. Open Office inclou un seguit d'aplicacions molt semblants a les de la *suite* Microsoft Office.
- **Pet expression**: frase o expressió característica d'un autor.
- **Plagi**: fet de publicar o de donar per pròpia l'obra literària, científica o artística d'una altra persona o bé inserir-ne una part dins l'obra pròpia sense citar-ne la font.
- **Versió:** funcionalitat de Microsoft Word per la qual es poden enregistrar diferents estats d'un mateix document en un sol fitxer físic.

13. Referències i bibliografia

13.1. Conversió de documents .doc a .txt

Pàgines web:

Ref. 1. XML.com: http://www.xml.com/pub/a/2003/12/31/qa.html
Ref. 2. XMLSoftware: http://www.xmlsoftware.com/convert.html

Grups d'Usenet:

Ref. 3. comp.programming
Ref. 4. comp.text.xml

13.2. Format dels documents Word

Pàgines web:

Ref. 5. Wikipèdia – Microsoft Word: http://en.wikipedia.org/wiki/Microsoft_Word
Ref. 6. Brian Jones Blog: http://blogs.msdn.com/brian_jones/default.aspx
Ref. 7. Office 2003: XML Reference Schemas (és un fitxer executable que es baixa de http://www.microsoft.com/downloads/details.aspx?FamilyId=FE118952-3547-420A-A412-00A2662442D9&displaylang=en)
Ref. 8. MSDN. Properties and Property Sets: http://msdn.microsoft.com/library/default.asp?url=/library/en-us/stg/stg/the_documentsummaryinformation_and_userdefined_property_sets.asp

Grups d'Usenet:

Ref. 9. Grups d'Usenet:
Ref. 10. microsoft.public.es.word
Ref. 11. microsoft.public.word
Ref. 12. microsoft.public.es.officedev
Ref. 13. microsoft.public.office.misc
Ref. 14. comp.programming

13.3. Sistemes d'enregistrament de l'autoria d'un document

Pàgines web:

Ref. 15. Solucionar problemas de inicio en Microsoft Word 2002: http://www.svetlian.com/msoffice/word2002_start.htm
Ref. 16. Manual básico e intermedio de Word 97: http://www.monografias.com/trabajos/word972/word972.shtml

Grups d'usenet:

Ref. 17. microsoft.public.es.word
Ref. 18. microsoft.public.word

Ref. 19. microsoft.public.es.officedev
Ref. 20. microsoft.public.office.misc

Paper:

Ref. 21. Microsoft Word 2000 Avançat – Centre de formació BRANCA, Recursos i Serveis
Ref. 22. Introducció a Word 2000. Jordi Cuenca i Ros, 2003-2005

13.4. Eines de detecció de plagi existents al mercat

Pàgines web:

Ref. 23. Turnitin: http://www.turnitin.com/static/home.html
Ref. 24. Eve2 Plagiarism Detection for Teachers: http://www.canexus.com/
Ref. 25. Original Checker: http://cise.sbu.ac.uk/orcheck/
Ref. 26. VAST: http://cise.lsbu.ac.uk/orcheck/vast.html
Ref. 27. MyDropBox: http://www.mydropbox.com/
Ref. 28. WCopyFind: http://plagiarism.phys.virginia.edu/Wsoftware.html

13.5. Desenvolupament de l'eina

Pàgines web:

Ref. 29. Plagiarism Detection – YAP: http://www.bio.cam.ac.uk/~mw263/YAP.html
Ref. 30. JPlag: Finding plagiarisms among a set of programs: http://page.mi.fu-berlin.de/~prechelt/Biblio/jplagTR.pdf
Ref. 31. A System for Detecting Software Plagiarism: http://www.cs.berkeley.edu/~aiken/moss.html
Ref. 32. Wikipèdia – Sting Searching Algorith: http://en.wikipedia.org/wiki/String_searching_algorithm

13.6. Problemàtica sobre plagis

Pàgines web:

Ref. 33. Sentence-based natural language plagiarism detection: http://portal.acm.org/citation.cfm?id=1086339.1086341
Ref. 34. Plagiarism, A Good Practice Guide By Jude Carroll and Jon Appleton May 2001: http://www.jisc.ac.uk/uploaded_documents/brookes.pdf
Ref. 35. Les webs explicades a l'apartat 13.4