ΑΛΕΞΑΝΔΡΕΙΟ Τ.Ε.Ι. ΘΕΣΣΑΛΟΝΙΚΗΣ

ΣΧΟΛΗ ΤΕΧΝΟΛΟΓΙΚΩΝ ΕΦΑΡΜΟΓΩΝ

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

Πτυχιακή εργασία

«ΜΕΛΕΤΗ ΚΑΙ ΧΡΗΣΗ ΤΟΥ TOOLKIT GATE (A GENERAL ARCHITECTURE FOR TEXT ENGINEERING)»

Του φοιτητή

Φώτιου Παρασχιάκου

Αρ. Μητρώου : 03/2161

Επιβλέπων καθηγητής

Μιχαήλ Σαλαμπάσης

ΠΙΝΑΚΑΣ ΠΕΡΙΕΧΟΜΕΝΩΝ

ΚΕΦΑΛΑΙΟ 1

ΕΙΣΑΓΩΓΗ

1.1 ΣΤΟΧΟΙ ΚΑΙ ΣΚΟΠΟΙ ΤΗΣ ΠΤΥΧΙΑΚΗΣ ΕΡΓΑΣΙΑΣ

Η πτυχιακή εργασία αυτή πραγματεύεται την μελέτη των σύγχρονων εφαρμογών της επεξεργασίας φυσικής γλώσσας (Natural Language Processing), και συγκεκριμένα του toolkit GATE. Το GATE είναι ένα open source περιβάλλον ανάπτυξης εφαρμογών Text Engineering το οποίο χρήζει ενδιαφέροντος καθώς είναι το αποδοτικό σχετικά με τα υπόλοιπα παρόμοια εργαλεία, εύχρηστο και εύκολα επεκτάσιμο, και συνεπώς μπορεί να χρησιμοποιηθεί σε ένα μεγάλο εύρος εφαρμογών υπολογιστικής γλωσσολογίας, Natural Language Processing και διάφορων άλλων παρεμφερών φύσεων.

Στόχος της παρούσας εργασίας είναι η μελέτη των δυνατοτήτων του GATE σε εφαρμογές Natural Language Processing καθώς και η ανάπτυξη μιας εφαρμογής η οποία θα χρησιμοποιεί το GATE για να λύσει προβλήματα πρακτικής φύσεως.

Η εφαρμογή αυτή θα υλοποιηθεί σε γλώσσα Java με τη βοήθεια των βιβλιοθηκών SWING για το γραφικό περιβάλλον του χρήστη και θα ενσωματώσει της λειτουργίες του GATE για την σημασιολογική ανάλυση εγγράφων τα οποία προέρχονται απο RSS Feeds.

1.2 ΔΟΜΗ ΤΗΣ ΠΤΥΧΙΑΚΗΣ ΕΡΓΑΣΙΑΣ

Η διαδικασία επίτευξης των στόχων της εργασίας παρουσιάζεται στα επιμέρους κεφάλαιά της. Ειδικότερα :

- Στο **κεφάλαιο 1** παρουσιάζονται εισαγωγικές πληροφορίες για την εργασία.
- Στο **κεφάλαιο 2** γίνεται μια περιγραφή της έννοιας της Επεξεργασίας Φυσικής Γλώσσας (Natural Language Processing ή NLP). Παρουσιάζεται η ιστορία του όρου, οι λειτουργίες και οι περιορισμοί της NLP καθώς και οι σύγχρονες τάσεις

και τα εργαλεία που χρησιμοποιούνται τη σημερινή ημέρα για τη λύση προβλημάτων τέτοιου τύπου.

- Στο **κεφάλαιο 3** παρουσιάζεται η διαδικασία της Κατανόησης Φυσικής Γλώσσας (Natural Language Understanding ή NLU). Η NLU είναι ένα υποπρόβλημα της NLP και αφορά την κατανόηση και την ανάλυση ενός κειμένου γραμμένο σε φυσική γλώσσα. Είναι μια πολύπλοκη διαδικασία η οποία με τη σειρά της αναλύεται σε υποδιαδικασίες οι οποίες θα παρουσιαστούν στο συγκεκριμένο κεφάλαιο.
- Στο **κεφάλαιο 4** παρουσιάζεται και μελετάται τεχνικά το GATE Framework, και παρουσιάζεται ο τρόπος με τον οποίο είναι δομημένη η αρχιτεκτονική του. Επίσης παρουσιάζονται εργαλεία benchmarking για την μέτρηση της απόδοσης του.
- Στο **κεφάλαιο 5** παρουσιάζεται το GATE Developer, το γραφικό περιβάλλον τοy GATE, και αναλύονται μερικές βασικές περιπτώσεις χρήσης του.
- Στο **κεφάλαιο 6** παρουσιάζεται το RSSGATE, το πρόγραμμα το οποίο δημιούργησα για την παρουσίαση των δυνατοτήτων του GATE, και γίνεται επίδειξη της χρησιμότητάς του σε μια πρακτική περίπτωση χρήσης.

ΚΕΦΑΛΑΙΟ 2

ΕΠΕΞΕΡΓΑΣΙΑ ΦΥΣΙΚΗΣ ΓΛΩΣΣΑΣ

(NATURAL LANGUAGE PROCCESSING)

Η επεξεργασία φυσικής γλώσσας (NLP) είναι ένας τομέας της πληροφορικής και της γλωσσολογίας ο οποίος ασχολείται με την αλληλεπίδραση μεταξύ των υπολογιστών και των ανθρώπινων (φυσικών) γλωσσών.

Η NLP είναι μια μέθοδος με την οποία μπορεί να επιτευχθεί η αλληλεπίδραση ανθρώπου με υπολόγιστή (human-computer interaction). Έχει αρκετά κοινά στοιχεία με την υπολογιστική γλωσσολογία, και θεωρείται επίσης υπο-τομέας της τεχνητής νοημοσύνης.

2.1 ΙΣΤΟΡΙΚΗ ΑΝΑΔΡΟΜΗ ΤΗΣ NLP

Ο όρος NLP πρωτοαναφέρεται στη δεκαετία του ’50, ωστόσο μπορούν να βρεθούν παραδείγματα χρήσης τέτοιας τεχνολογίας και νωρίτερα. ΄Το 1950, ο Alan Turing δημοσίευσε μια εργασία με τίτλο “Computing Machinery and Intelligence”, η οποία παρουσίασε το γνωστό πλέον Turing test ώς κριτήριο νοημοσύνης. Το κριτήριο βασίζεται στην δυνατότητα ενός υπολογιστικού συστήματος να υποδυθεί έναν άνθρωπο σε μια γραπτή συνομιλία πραγματικού χρόνου με έναν ανθρώπινο κριτή, αρκετά ικανοποιητικά έτσι ώστε ο κριτής να μην είναι ικανός να ξεχωρίσει με σιγουριά (βασισμένος αποκλειστικά στο περιεχόμενο της συνομιλίας) των άνθρωπο απο τον υπολογιστή. Στο πείραμα της Georgetown (1954), ένας υπολογιστής μετέφρασε αυτόματα 60 ολοκληρωμένες προτάσεις της Ρώσσικης γλώσσας στην Αγγλική. Οι επιστήμονες που διεξήγαγαν το πείραμα υποστήριξαν ότι μέσα σε τρία με πέντε χρόνια, το πρόβλημα της αυτόματης μετάφρασης (machine translation) θα έχει λυθεί.

Στη δεκαετία του 60, το SHRDLU, ένα σύστημα κατανόησης φυσικής γώσσας (NLU), οδήγησε τους ερευνητές σε πολύ αισιόδοξες προβλέψεις για το μέλλον του NLP.

Το σύστημα χρησιμοποιούσε τη μέθοδο «blocks worlds» με περιορισμένα λεξιλόγια, και είχε παρουσιάσει πολύ καλά αποτελέσματα στα εργαστήρια. Η πραγματική πρόοδος οστώσο ήταν πολύ πιο αργή, και έτσι, μετά την αναφορά ALPAC το 1966, η οποία συμπέρανε ότι τα 10 χρόνια της έρευνας δεν είχαν τα αναμενόμενα αποτελέσματα, η χρηματοδότηση μειώθηκε δραστικά.

Το ELIZA γράφτηκε απο τον Joseph Weizenbaum την περίοδο 1964 – 1966. Το πρόγραμμα προσωμείωνε έναν ψυχοθεραπευτή. Παρ όλο που δεν χρησιμοποιούσε σχεδόν καθόλου πληροφορίες σχετικά με το τον ανθρώπινο ψυχισμό, τις σκέψεις, τα συναισθήματα κλπ, το ELIZA πολλές φορές παρείχε μια εντυπωσιακά ανθρωπομορφική αλληλεπίδραση. Όταν η είσοδος που παρείχε ο «ασθενής» υπέρβαινε την μικρή βάση γνώσεών του, το ELIZA συνήθως έδινε μια αόριστη απάντηση π.χ. απαντώντας στη φράση «πονάει το κεφάλι μου» με την απάντηση : «Γιατί λες ότι πονάει το κεφάλι σου;»

Κατα τη διάρκεια της δεκαετίας του ’70, πολλοί προγραμματιστές ξεκίνησαν να γράφουν «εννοιολογικές οντολογίες», οι οποίες δομούσαν πληροφορίες του πραγματικού κόσμου σε δεδομένα τα οποία εύκολα μπορούσαν να κατανοηθούν απο υπολογιστή. Μερικά παραδείγματα : MARGIE (Schank, 1975), SAM (Cullingford, 1978), PAM (Wilensky, 1978), TaleSpin (Meehan, 1976), QUALM (Lehnert, 1977), Politics (Carbonell, 1979), Plot Units (Lehnert 1981). Εκείνη την περίοδο, γράφτηκαν επίσης πολλοί αυτόματοι συνομιλητές (chatterbots) π.χ. PARRY, Racter και Jabberwacky.

Στο τέλος της δεκαετίας του ’80, καθώς η υπολογιστική δύναμη αυξήθηκε και έγινε φθηνότερη, ξεκίνησε να δείχνεται περισσότερο ενδιαφέρον στα στατιστικά μοντέλα της αυτόματης μετάγφρασης (machine translation).

2.2 ΛΕΙΤΟΥΡΓΙΕΣ ΚΑΙ ΠΕΡΙΟΡΙΣΜΟΙ

. Τα συστήματα δημιουργίας φυσικής γλώσσας (natural language generation systems) μετατρέπουν πληροφορίες από βάσεις δεδομένων υπολογιστών σε αναγνώσιμη ανθρώπινη γλώσσα. Τα συστήματα κατανόησης φυσικής γλώσσας (Natural language understanding systems), μετατρέπουν κομμάτια ανθρώπινης γλώσσας σε πιο τυπικές αναπαραστάσεις (formal representations) οι οποίες καθιστούν πιο εύκολη την κατανόηση τους και τον χειρισμό τους απο ένα υπολογιστικό σύστημα.

Πολλά προβλήματα NLP εφαρμόζονται εξ ίσου στην δημιουργία και στην κατανόηση του κειμένου. Για παράδειγμα, ο υπολογιστής, για να είναι σε θέση να «κατανοήσει» μια πρόταση στην Ελληνική (ή οποιαδήποτε) γλώσσα, θα πρέπει να πρέπει να μπορεί να μοντελοποιήσει την μορφολογία (την δομή των λέξεων) της πρότασης αυτής. Όμως, ένα μορφολογικό μοντέλο χρειάζεται και για την δημιουργία μιας γραμματικά σωστής πρότασης στην Ελληνική γλώσσα.

2.3 ΥΠΟΠΡΟΒΛΗΜΑΤΑ

- **Τμηματοποιηση ομιλίας (Speech segmentation)**

Στις περισσότερες προφορικές γλώσσες, οι ήχοι που αναπαριστούν διαδοχικά γράμματα αναμιγνύωνται μεταξύ τους, με αποτέλεσμα η μετατροπή του αναλογικού σήματος σε διακριτους χαρακτήρες να είναι μια πολυ δύσκολη διαδικασία. Επίσης, στην φυσική ομιλία δεν υπάρχουν σχεδόν καθόλου παύσεις μεταξύ των διαδοχικών λέξεων, και έτσι πρέπει ο υπολογιστής να τις ξεχωρίσει μεταξύ τους. Αυτή η εύρεση των ορίων των λέξεων θα πρέπει να λαμβάνει υπ όψιν γραμματικές και σημασιολογικές σταθερές, καθώς και τα συμφραζόμενα κάθε πρότασης.

- **Τμηματοποίηση κειμένου (Text segmentation)**

Κάποιες γλώσσες όπως η Κινέζικη, η Ιαπωνική ή η Ταιλανδέζικη, δεν έχουν όρια μεταξύ κάθε λέξης ξεχωριστά, οπότε κάθε σημαντική ανάλυση κειμένου (parsing), συνήθως χρειάζεται να βρεθούν τα όρια των λέξεων.

- **Σημείωση μέρους λόγου (Part-of-speech tagging)**

Η ανάλυση της γλώσσας απο την πλευρά του συντακτικού περιλαμβάνει και αυτή την λειτουργία η οποία σημειώνει τι μέρος του λόγου είναι κάθε λέξη. Η λειτουργία αυτή θα αναφερθεί εκτενέστερα στο επόμενο κεφάλαιο.

- **Αποσαφήνιση εννοιών λέξεων (word sense disambiguation)**

Πολλές λέξεις έχουν πάνω απο μια έννοια. Θα πρέπει να βρεθεί η έννοια η οποία έχει περισσότερο νόημα σχετικά με τα συμφραζόμενα.

- **Συντακτική ασάφεια (Syntactic ambiguity)**

Η γραμματική των φυσικών γλωσσών είναι ασαφής. Παραδείγματος χάριν, συχνά,σε κάθε πρόταση υπάρχουν πολλά πιθανά δέντρα ερμηνείας (parse trees). Η

επιλογή του πιο ταιριαστού για κάθε περίπτωση απαιτεί πληροφορίες σχετικά με τα συμφραζόμενα και τη σημειολογία της πρότασης. Πολλά προβληματα συντακτικής ασάφειας απαιτούν αποσαφήνιση των ορίων μιας πρότασης (sentence boundary disambiguation).

- **Ατελής ή αφύσικη είσοδος (Imperfect or irregular input)**

Για παράδειγμα : Ξένες ή τοπικές προφορές και διάλεκτοι και διαταραχές της ομιλίας, ορθογραφικά λάθη καθώς και πιθανά λάθη της οπτικής ανάγνωσης χαρακτήρων (OCR) σε κείμενα.

2.4 ΣΤΑΤΙΣΤΙΚΗ NLP

Η στατιστική ανάλυση φυσικής γλώσσας χρησιμοποιεί στοχαστικές, πιθανοτικές και στατιστικές μεθόδους με σκοπό να λύσει κάποια απο τα προαναφερθέντα προβλήματα, ειδικά αυτά που προκύπτουν απο το μέγεθος της ασάφειας η οποία υπάρχει όταν επεξεργαζόμαστε μεγαλύτερες προτάσεις χρησιμοποιόντας ρεαλιστικές γραμματικές, με αποτέλεσμα να υπάρχουν χιλιάδες η και εκατομμύρια πιθανές αναλύσεις. Η αποσαφήνιση συνήθως γίνεται με την χρήση «Σωμάτων» (corpora) και μοντέλων Markov. Η στατιστική NLP αποτελείται από όλες τις ποσοτικές προσεγγίσεις στην αυτοματοποιημένη επεξεργασία γλώσσας, συμπεριλαμβανομένων και της πιθανοτικής μοντελοποίησης (probabilistic modeling), της θεωρίας των πληροφοριών (information theory) και της γραμμικής άλγεβρας.

Η τεχνολογία της στατιστικής NLP πηγάζει κυρίως απο την μηχανική μάθηση (machine learning) και την εξόρυξη δεδομένων (data mining). Οι παραπάνω τεχνολογίες είναι το πεδίο της τεχνητής νοημοσύνης το οποίο ασχολείται με την επεξεργασία δεδομένων και την μάθηση μέσα απο αυτά.

2.5 ΣΗΜΑΝΤΙΚΟΤΕΡΕΣ ΕΦΑΡΜΟΓΕΣ ΤΗΣ NLP

- Αυτόματη δημιουργία περίληψης
- Βοήθημα για την ανάγνωση και τη γραφή ξένων γλωσσών
- Εξόρυξη πληροφοριών
- Αυτόματη μετάφραση

- Αναγνώριση κύριων οντοτήτων (Σε ένα κομμάτι κειμένου φυσικής γλώσσας, η εύρεση των κομματιών που ταιριάζουν με κύρια ονόματα, π.χ ονόματα ανθρώπων ή τόπων. Σε γλώσσες όπως η Ελληνική και η Αγγλική τα κύρια ονόματα είναι γραμμένα με το πρώτο γράμμα κεφαλαίο οπότε η διαδικασία είναι σχετικά απλή, αντιθέτως με τα γερμανικά και πολλές άλλες γλώσσες στις οποίες δεν ισχύει κάτι τέτοιο).
- Δημιουργία φυσικής γλώσσας
- Κατανόηση φυσικής γλώσσας
- Αναζήτηση σε φυσική γλώσσα
- Αναγνώριση οπτικών χαρακτήρων (OCR)
- Απάντηση σε ερωτήσεις (Απάντηση σε φυσική γλώσσα σε ερωτήσεις που επίσης εκφράζονται σε φυσική γλώσσα, είτε κλειστού τύπου π.χ «Ποιά είναι η πρωτεύουσα της Ελλάδας είτε ανοιχτού π.χ. «Ποιό είναι το νόημα της ζωής»
- Αναγνώριση φωνής και μετατροπή σε κείμενο
- Μετατροπή κειμένου σε φωνή
- Επιμέλεια κειμένου (proof reading)

2.6 ΣΥΓΧΡΟΝΑ ΕΡΓΑΛΕΙΑ NLP

Ο ακόλουθος πίνακας παρουσιάζει τα δημοφιλέστερα εργαλεία (toolkits) που χρησιμοποιούνται για natural language processing. Είναι συλλογές απο βιβλιοθήκες, frameworks, και εφαρμογές συμβολικής και στατιστικής επεξεργασίας φυσικής γλώσσας και ομιλίας. Τα εργαλεία NLP συνήθως προσφέρουν :

- Ανίχνευση προτάσεων
- Δημιουργία tokens (tokenization)
- Σημείωση μέρους λόγου (Part of speech tagging)
- Μαζική ανάλυση κομματιού κειμένου (Text chunking)
- Λημματοποίηση (lemmatization)
- Ανάλυση διπλής αναφοράς (coreference analysis)

- Αναγνώριση κύριων οντοτήτων

Όνομα	Γλώσσα	Άδεια	Δημιουργοί	Website
AlchemyAPI	C, C++, C#, Java, Python, Perl, Ruby	Ελεύθερη ή εμπορική	Orchestr8	http://www.alchemyapi.com/
Cogito		Εμπορική	Expert System S.p.A.	http://www.expertsystem.net/page.asp?id=1521
Illinois NLP	Java, C++	Ανοιχτή για ερευνητές	Cognitive Computation Group at the University of Illinois at Urbana Champaign	http://l2r.cs.uiuc.edu/~cogcomp/software.php
Carabao Language Kit	Any COM+ compliant language. Customization is via data entry	Εμπορική με δωρεάν εργαλεία ανάπτυξης	Digital Sonata Pty Ltd	http://www.digitalsonata.com/default.aspx
Distinguo	C++	Εμπορική	Ultralingua Inc.	http://ultralingua.com/en/semantic-search.htm
Ellogon	C / C++	LGPL	Georgios Petasis	http://www.ellogon.org/
FreeLing	C++	GPL	Universitat	http://garraf.epsevg

			Politècnica de Catalunya	.upc.es/freeling/
General Architecture for Text Engineering	Java	LGPL	GATE open source communtiy	http://gate.ac.uk/
LingPipe	Java	Ελεύθερη ή εμπορική	Alias-i	http://alias-i.com/lingpipe/index.html
LinguaStream	Java	Ελεύθερη για έρευνα	University of Caen, France	http://www.linguastream.org/
Mallet	Java	Common Public License	University of Massachusetts Amherst	http://mallet.cs.umass.edu/
MII nlp toolkit	Java	LGPL	UCLA Medical Imaging Informatics (MII) Group	http://www.mii.ucla.edu/nlp
Modular Audio Recognition	Java	BSD	The MARF Research and Development Group,	http://marf.sf.net/
MontyLingua	Python, Java	Ελεύθερη για έρευνα	MIT	http://web.media.mit.edu/~hugo/montylingua/
Natural Language Toolkit (NLTK)	Python	Apache 2.0		http://www.nltk.org/Home

NooJ (based on INTEX)	.NET Framework-based	Ελεύθερη για έρευνα	University of Franche-Comté, France	http://www.nooj4nlp.net/
OpenNLP	Java	LGPL	Online κοινότητα	http://opennlp.sourceforge.net/
Rosette		Εμπορική	Basis Technology	http://rosette.basistech.com/
Stanford NLP	Java	GPL	The Stanford Natural Language Processing Group	http://nlp.stanford.edu/software/index.shtml
UIMA	Java / C++	Apache 2.0	Apache	http://incubator.apache.org/uima/index.html
WebLab	Java	LGPL	OW2	http://weblab-project.org/
UniteX	Java & C++	LGPL	Laboratoire d'Automatique Documentaire et Linguistique	http://www-igm.univ-mlv.fr/~unitex/

Σχήμα 2.1 : Πίνακας των σύγχρονων εργαλείων NLP

ΚΕΦΑΛΑΙΟ 3

ΚΑΤΑΝΟΗΣΗ ΦΥΣΙΚΗΣ ΓΛΩΣΣΑΣ

(NATURAL LANGUAGE UNDERSTANDING)

Μια από τις μεγάλες φιλοσοφικές διαμάχες κατά το πέρασμα των αιώνων έχει επικεντρωθεί γύρω από το ερώτημα του τι σημαίνει μια πρόταση. Είναι πολύ δύσκολο να δοθεί οριστική απάντηση. Αλλά εφόσον αντιληφθούμε ότι η κατανόηση μιας γλώσσας περιλαμβάνει αντιστοίχισή της σε κάποιες αναπαραστάσεις κατάλληλες σε μια συγκεκριμένη κατάσταση, γίνεται εύκολο να δούμε γιατί ερωτήματα όπως «τι είναι κατανόηση γλώσσας;» και «τι σημαίνει μια πρόταση;» έχουν αποδειχθεί τόσο δύσκολα να απαντηθούν. Χρησιμοποιούμε τη γλώσσα για να περιγράψουμε μια τόσο μεγάλη ποικιλία καταστάσεων που δε είναι δυνατόν να υπάρξει ένας μοναδικός ορισμός της κατανόησης. Καθώς κάποιος καταπιάνεται με το έργο κατασκευής προγραμμάτων που αντιλαμβάνονται τη φυσική γλώσσα, ένα από τα βασικά πράγματα που πρέπει να έχει υπ όψιν του είναι να ορίσει επακριβώς τον απώτερο στόχο καθώς επίσης και την τελική μορφή της εσωτερικής αναπαράστασης.

3.1 ΤΑ ΒΗΜΑΤΑ ΤΗΣ ΔΙΑΔΙΚΑΣΙΑΣ

Πριν εξεταστούν με λεπτομέρεια τα συστατικά της διαδικασίας κατανόησης φυσικής γλώσσας είναι χρήσιμο να ομαδοποιηθούν για να βρεθεί ο τρόπος με τον οποίον ταιριάζουν μεταξύ τους. Χονδρικά η διαδικασία αναλύεται στα ακόλουθα τμήματα:

- Μορφολογική Ανάλυση (Morphological Analysis) - Ξεχωριστές λέξεις αναλύονται στα συστατικά τους και τα μη λεκτικά σύμβολα όπως π.χ.τα σημεία στίξης, και χωρίζονται από τις λέξεις.
- Συντακτική Ανάλυση (Syntactic Analysis) - Γραμμικές ακολουθίες λέξεων μετατρέπονται σε δομές που απεικονίζουν τον τρόπο με τον οποίο οι λέξεις

συνδέονται η μια με την άλλη. Ορισμένες ακολουθίες λέξεων ίσως απορριφθούν αν παραβιάζουν τους κανόνες της γλώσσας, οι οποίοι ρυθμίζουν τον τρόπο με τον οποίο συνδυάζονται οι λέξεις. Για παράδειγμα ένας συντακτικός αναλυτής θα απέρριπτε την πρόταση «Το παιδί πήγε κατάστημα στο».

- Σημασιολογική Ανάλυση (Semantic Analysis) – Προσδίδει νόημα στις δομές που εξάγονται από το συντακτικό αναλυτή. Με άλλα λόγια γίνεται ένα είδος απεικόνισης ανάμεσα στις συντακτικές δομές και στα αντικείμενα του χώρου του προβλήματος. Οι δομές που δεν μπορούν να αντιστοιχισθούν θα απορριφθούν. Για παράδειγμα στις περισσότερες περιπτώσεις η πρόταση «Άχρωμες πράσινες ιδέες κοιμούνται με οργή» θα χαρακτηριζόταν ως σημασιολογικά ανώμαλη.

- Ανάλυση Διαλόγου (Discourse Integration) - Το νόημα μιας ξεχωριστής πρότασης ίσως βασίζεται στις προτάσεις που προηγούνται και μπορεί να επηρεάζει το νόημα των προτάσεων που έπονται. Για παράδειγμα η λέξη «το» της πρότασης «Ο Γιάννης το ήθελε» βασίζεται στο προηγούμενο νόημα ενώ η λέξη «Γιάννης» ίσως να επηρεάζει το νόημα των επόμενων προτάσεων. (Όπως η «Αυτός το έκανε πάντα».)

- Πραγματολογική Ανάλυση (Pragmatic Analysis) - Η δομή που αναπαριστά το τι έχει ειπωθεί επανερμηνεύεται για να καθορίσει το τι πραγματικά σημαίνει. Για παράδειγμα η πρόταση «Ξέρεις τι ώρα είναι;» θα πρέπει να ερμηνευτεί ως παράκληση για να μας πουν την ώρα.

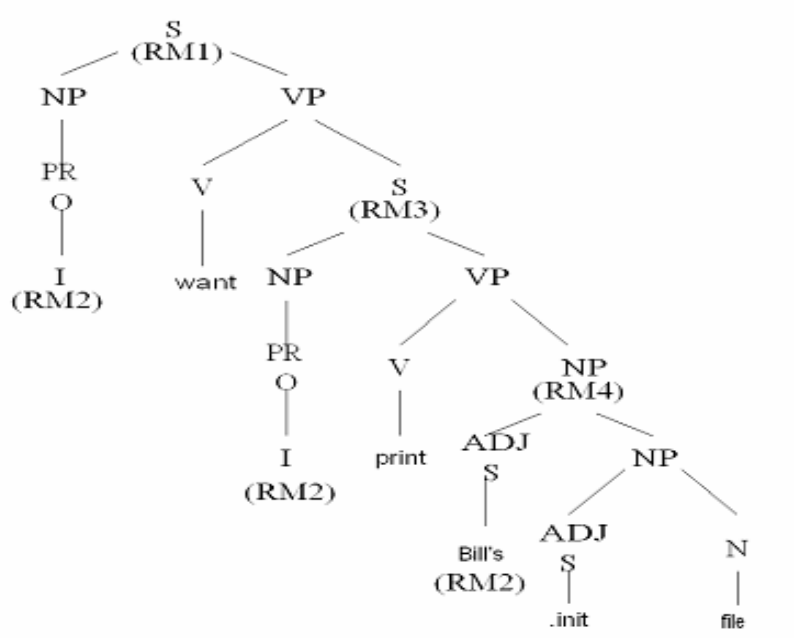
Τα όρια ανάμεσα στις πέντε αυτές φάσεις είναι συχνά ασαφή. Οι φάσεις άλλοτε εκτελούνται σειριακά και άλλοτε όλες μαζί. Στην πρώτη περίπτωση ίσως χρειαστεί η μία να καλέσει την άλλη. Για παράδειγμα, ένα τμήμα της συντακτικής ανάλυσης της πρότασης “Is the glass jar peanut butter?” αποφασίζει πώς να κατασκευάσει δύο ονοματικές φράσεις από τέσσερα ουσιαστικά στο τέλος της πρότασης (προτάσεις της μορφής “Is the x y?”). Κάθε μια από τις παραπάνω συνιστώσες είναι συντακτικά δυνατή: glass, glass jar, glass jar peanut, jar peanut butter, peanut butter, butter. Ένας συντακτικός επεξεργαστής πρέπει από μόνος του να διαλέξει μία από αυτές και κάθε απόφαση πρέπει να παρθεί συγκρίνοντας με κάποιο πραγματικό μοντέλο, κατά το οποίο κάποιες από αυτές τις φράσεις θα έχουν νόημα και κάποιες όχι. Έτσι καταλήγουμε σε μία συντακτική δομή με τις συνιστώσες “glass jar” και “peanut butter”.

Για αυτό το λόγο, παρόλο που συχνά είναι χρήσιμο να τις ξεχωρίζουμε, οι πέντε φάσεις επεξεργασίας μπορεί να αλληλεπιδρούν μεταξύ τους, πράγμα που κάνει τον πλήρη διαχωρισμό τους αδύνατο.

Πιο συγκεκριμένα, για να εντοπίσουμε με ακρίβεια το πρόβλημα της κατανόησης φυσικής γλώσσας, θα βοηθούσε ο παρακάτω διαχωρισμός της διαδικασίας αποσύνθεσης ενός προγράμματος σε δύο μέρη:

- Στη γνώση και στις διαδικασίες που απαιτούνται.
- Στη συνολική δομή ελέγχου που επιβάλλεται σε αυτές τις διαδικασίες.

Κάτω από αυτή την επισήμανση, εξετάζουμε το παρακάτω παράδειγμα, για να δούμε τον τρόπο με τον οποίο λειτουργούν οι ανεξάρτητες διαδικασίες. Θεωρούμε ότι για το παράδειγμά μας, οι διεργασίες συμβαίνουν σειριακά. Έστω ότι έχουμε ένα λειτουργικό σύστημα με μία διεπαφή (interface) στα αγγλικά, και τυπώνεται η ακόλουθη πρόταση: “I want to print Bill’s .init file”.



Σχήμα 3.2 : Αποτέλεσμα συντακτικής ανάλυσης της πρότασης : “I want to print Bill’s .init file”

3.1.1 ΜΟΡΦΟΛΟΓΙΚΗ ΑΝΑΛΥΣΗ

Η μορφολογική ανάλυση θα πρέπει να κάνει τα παρακάτω:

- Να ξεχωρίσει από τη λέξη “Bill’s” το ουσιαστικό “Bill” και το κτητικό επίθεμα “’s ”.

- Να αναγνωρίσει τη συμβολοσειρά ".init" ως κατάληξη ενός αρχείου του υπολογιστή και να αντιληφθεί ότι λειτουργεί σαν επίθετο στην πρόταση.

Επιπρόσθετα, η διαδικασία θα αναθέσει κάθε λέξη της πρότασης σε συντακτικές κατηγορίες. Αυτό γίνεται συνήθως σε αυτό το σημείο, διότι η ερμηνεία των επιθεμάτων ή των προθεμάτων ίσως βασιστεί στη συντακτική κατηγορία κάθε λέξης. Για παράδειγμα, ας θεωρήσουμε τη λέξη "prints".

Αυτή είναι είτε ουσιαστικό πληθυντικού αριθμού (με το "-s" να χαρακτηρίζει τον πληθυντικό), είτε ρήμα, στο τρίτο πρόσωπο του ενικού (όπως στο "he prints"), κατά το οποίο το "-s" δηλώνει πρόσωπο και αριθμό. Έτσι λοιπόν, αν αυτό το βήμα γίνει τώρα, τότε επιστρέφοντας στο παράδειγμά μας, θα εμφανισθεί ασάφεια αφού οι λέξεις "want", "print" και "file" μπορούν να ανήκουν σε παραπάνω από μία κατηγορία.

3.1.2 ΣΥΝΤΑΚΤΙΚΗ ΑΝΑΛΥΣΗ

Η συντακτική ανάλυση θα πρέπει να λάβει τα αποτελέσματα της μορφολογικής ανάλυσης για να κατασκευάσει μια δομημένη περιγραφή της πρότασης. Ο σκοπός αυτής της διαδικασίας (parsing), είναι να μετατρέψει μια γραμμική λίστα λέξεων που σχηματίζουν μια πρόταση σε μια δομή που ορίζει κάθε τμήμα που αναπαριστάται από αυτή τη λίστα. Για το παράδειγμά μας, το αποτέλεσμα της ανάλυσης φαίνεται στο Σχήμα 3.1 Οι λεπτομέρειες αυτής της αναπαράστασης δεν είναι ιδιαιτέρως σημαντικές. Αυτό που είναι σημαντικό, είναι το γεγονός ότι μια γραμμική πρόταση έχει μετατραπεί σε μια ιεραρχημένη δομή, και ότι αυτή η δομή αντιστοιχεί σε ενότητες προτάσεων, όπως οι ονοματικές φράσεις (noun phrases). Οι ενότητες αυτές θα αντιστοιχούν σε νοηματικές ενότητες κατά την εκτέλεση της σημασιολογικής ανάλυσης.

Κάτι χρήσιμο που κάνουμε, αν και δεν το κάνουν όλα τα συντακτικά συστήματα, είναι να δημιουργήσουμε ένα σύνολο οντοτήτων, τις οποίες αποκαλούμε σημεία παραπομπής (reference markers) και φαίνονται στις παρενθέσεις του συντακτικού δέντρου. Καθένα από αυτά αναφέρεται σε μια οντότητα που αναφέρεται στην πρόταση. Χρησιμεύουν αργότερα γιατί παρέχουν το μέρος όπου μπορούμε να συσσωρεύσουμε πληροφορία για τις οντότητες που συναντούμε. Συνεπώς, αν και μέχρι εδώ δεν έχουμε επιχειρήσει να κάνουμε

σημασιολογική ανάλυση, έχουμε σχεδιάσει τη διαδικασία συντακτικής ανάλυσης έτσι ώστε να βρίσκει συστατικά στα οποία μπορεί να αποδωθεί κάποιο νόημα.

3.1.3 ΣΗΜΑΣΙΟΛΟΓΙΚΗ ΑΝΑΛΥΣΗ

Η σημασιολογική ανάλυση πρέπει να κάνει δύο σημαντικά πράγματα:

- Πρέπει να απεικονίσει μεμονωμένες λέξεις στα κατάλληλα αντικείμενα της βάσης δεδομένων ή της βάσης γνώσεων.
- Πρέπει να δημιουργήσει τις σωστές δομές που ανταποκρίνονται στον τρόπο με τον οποίο συνδυάζονται τα νοήματα των μεμονωμένων λέξεων μεταξύ τους.

Για το ακόλουθο παράδειγμα ας υποθέσουμε ότι έχουμε μία βάση γνώσης, βασισμένη σε πλαίσια, που περιέχει τις οντότητες που απεικονίζονται στο Σχήμα 3.2 Τότε μπορούμε να δημιουργήσουμε ένα μερικό νόημα, όσον αφορά τη συγκεκριμένη βάση γνώσης, το οποίο απεικονίζεται στο Σχήμα 3.3. Το σημείο αναφοράς RM1 αντιστοιχεί στο συμβάν κορυφαίου επιπέδου της πρότασης. Είναι ένα συμβάν που δηλώνει επιθυμία σύμφωνα με το οποίο ο ομιλητής (υποδεικνύεται από το “I”) επιθυμεί από το σύστημα να του δώσει εκείνα τα αρχεία που έχουν κατάληξη “.init” και που ο ιδιοκτήτης τους είναι ο Bill.

3.1.4 ΑΝΑΛΥΣΗ ΔΙΑΛΟΓΟΥ

Σε αυτό το σημείο, έχουμε εξακριβώσει για ποια πράγματα μιλάει η πρόταση, αλλά δεν ξέρουμε ακόμη σε ποια συγκεκριμένα υποκείμενα αναφέρεται. Ειδικότερα, σε ποιόν αναφέρεται η αντωνυμία “I” ή το κύριο όνομα “Bill”. Για να επισημάνουμε αυτές τις αναφορές απαιτείται μια προσέγγιση σε ένα μοντέλο του τρέχοντος διαλόγου, από το οποίο μπορούμε να μάθουμε ότι ο τρέχων χρήστης (που τύπωσε τη λέξη “I”) είναι ο User068 και ότι το μόνο άτομο που ονομάζεται Bill για τον οποίο μπορούμε να μιλάμε είναι ο User073. Αφού εξακριβώσουμε τη σωστή αναφορά για τον Bill, μπορούμε να προσδιορίσουμε ακριβώς και σε ποιο αρχείο αναφέρεται: Το F1 είναι το μοναδικό αρχείο με κατάληξη “.init” που έχει ιδιοκτήτη τον Bill.

3.1.5 ΠΡΑΓΜΑΤΟΛΟΓΙΚΗ ΑΝΑΛΥΣΗ

Έως τώρα έχουμε μια σαφή περιγραφή για το τι έχει ειπωθεί, υπό τη μορφή όρων που παρέχονται από τη βάση γνώσεών μας. Το τελικό βήμα στην επίτευξη μιας αποτελεσματικής διαδικασίας κατανόησης είναι να αποφασίσουμε τι θα κάνουμε ως προς το αποτέλεσμα. Ένας πιθανός τρόπος είναι να καταγράψουμε αυτό που έχει ειπωθεί και να το δεχτούμε σαν ένα υπάρχον γεγονός.

Για εκείνες τις προτάσεις, των οποίων το επιδιωκόμενο αποτέλεσμα είναι ξεκάθαρα δηλωμένο, ο παραπάνω τρόπος είναι ό,τι ακριβώς χρειάζεται. Όμως για άλλες προτάσεις, συμπεριλαμβανομένης και αυτής του παραδείγματος, το επιδιωκόμενο αποτέλεσμα είναι διαφορετικό. Μπορούμε να το αποκαλύψουμε εφαρμόζοντας κανόνες που χαρακτηρίζουν ένα διάλογο. Στο παράδειγμά μας, εκμεταλλευόμαστε το γεγονός ότι όταν ο χρήστης ζητά κάτι που το σύστημα είναι ικανό να επεξεργαστεί, τότε το σύστημα πρέπει και να το πράξει. Έτσι παράγεται το τελικό νόημα του σχήματος 3.4.

Χρήστης	***Χρήστης068***
είναι : Άτομο	περίσταση : Χρήστης
* κωδικό-όνομα : πρέπει να είναι <συμβολοσειρά>	κωδικό-όνομα : Susan-Black
Χρήστης073	***F1***
περίσταση : Χρήστης	περίσταση : Δομή-Αρχείου
κωδικό-όνομα : Bill-Smith	όνομα : stuff
Δομή-Αρχείου	επέκταση : .init
είναι : Αντικείμενο-Πληροφορίας	κάτοχος : User073
Εκτύπωση	στον-κατάλογο : /wsmith/
είναι : Φυσικό-Γεγονός	***Επιθυμία***
* δράστης : πρέπει να είναι <έμψυχο ή πρόγραμμα>	είναι : Νοητικό-Γεγονός
* αντικείμενο : πρέπει να είναι <αντικείμενο-πληροφορίας>	* δράστης : πρέπει να είναι <έμψυχο>
Επιθυμία	* αντικείμενο : πρέπει να είναι <κατάσταση ή γεγονός>
είναι : Νοητικό-Γεγονός	**Διαταγή**
* δράστης : πρέπει να είναι <έμψυχο>	είναι : Νοητικό-Γεγονός
* αντικείμενο : πρέπει να είναι <κατάσταση ή γεγονός>	* δράστης : πρέπει να είναι <έμψυχο>
Αυτό-το-Σύστημα	* εκτελεστής : πρέπει να είναι <έμψυχο ή πρόγραμμα>
περίσταση : Πρόγραμμα	

Σχήμα 3.2 : Ένα τμήμα της βάσης γνώσης

Το τελικό βήμα στην πραγματολογική ανάλυση είναι ο μετασχηματισμός, όποτε είναι απαραίτητος, από την αναπαράσταση της βάσης γνώσης σε μία εντολή που θα εκτελεστεί από το σύστημα. Στην περίπτωσή μας, το παραπάνω βήμα είναι απαραίτητο και βλέπουμε ότι το τελικό αποτέλεσμα στη διαδικασία κατανόησης είναι το εξής:

RM1	***{ολόκληρη η πρόταση}***
περίσταση : Επιθυμία	
δράστης : RM2 {I}	
αντικείμενο : RM3 {ένα γεγονός εκτύπωσης}	
RM2	***{I}***
RM3	***{ένα γεγονός εκτύπωσης}***
περίσταση : Εκτύπωση	
δράστης : RM2 {1}	
αντικείμενο : RM4 {το αρχείο .init του Bill}	
RM4	***{το αρχείο .init του Bill}***
περίσταση : Δομή-Αρχείου	
επέκταση : .init	
κάτοχος : RM5 {Bill}	
RM5	***{Bill}***
περίσταση : Επιθυμία	
μικρό-όνομα : Bill	

Σχήμα 3.3 : Το μερικό νόημα μιας πρότασης

Ερμηνεία
περίσταση : Διαταγή
δράστης : Χρήστης068
εκτελεστής : Αυτό-το-Σύστημα
αντικείμενο : P27
P27
περίσταση : Εκτύπωση
δράστης : Αυτό-το-Σύστημα
αντικείμενο : F1

Σχήμα 3.4 : Αναπαράσταση της Επιθυμητής Ερμηνείας

3.2 ΣΥΝΤΑΚΤΙΚΗ ΕΠΕΞΕΡΓΑΣΙΑ (PARSING)

Η συντακτική επεξεργασία είναι το βήμα κατά το οποίο μία πρόταση εισόδου μετατρέπεται σε μία ιεραρχημένη δομή που ανταποκρίνεται στις νοηματικές μονάδες της πρότασης. Η όλη διαδικασία ονομάζεται αλλιώς και ανάλυση (parsing). Παρόλο που υπάρχουν συστήματα κατανόησης φυσικής γλώσσας που την παραλείπουν, η ανάλυση παίζει σημαντικό ρόλο για δύο λόγους:

1. Η σημασιολογική επεξεργασία πρέπει να χειρίζεται τα συστατικά της πρότασης. Αν δεν υπάρχει συντακτική ανάλυση, τότε το σύστημα που ασχολείται με τη σημασιολογία πρέπει να αποφασίσει βάσει των δικών του συστατικών. Όταν τελειώσει η ανάλυση, περιορίζει τον αριθμό των συστατικών που έχει ήδη θεωρήσει η σημασιολογία.
2. Η συντακτική ανάλυση απαιτεί λιγότερους υπολογισμούς από τη σημασιολογική (η οποία απαιτεί σημαντική προσπάθεια εξαγωγής συμπεράσματος). Για αυτό το λόγο παίζει σημαντικό ρόλο στο να μειώνει τη συνολική πολυπλοκότητα του συστήματος.

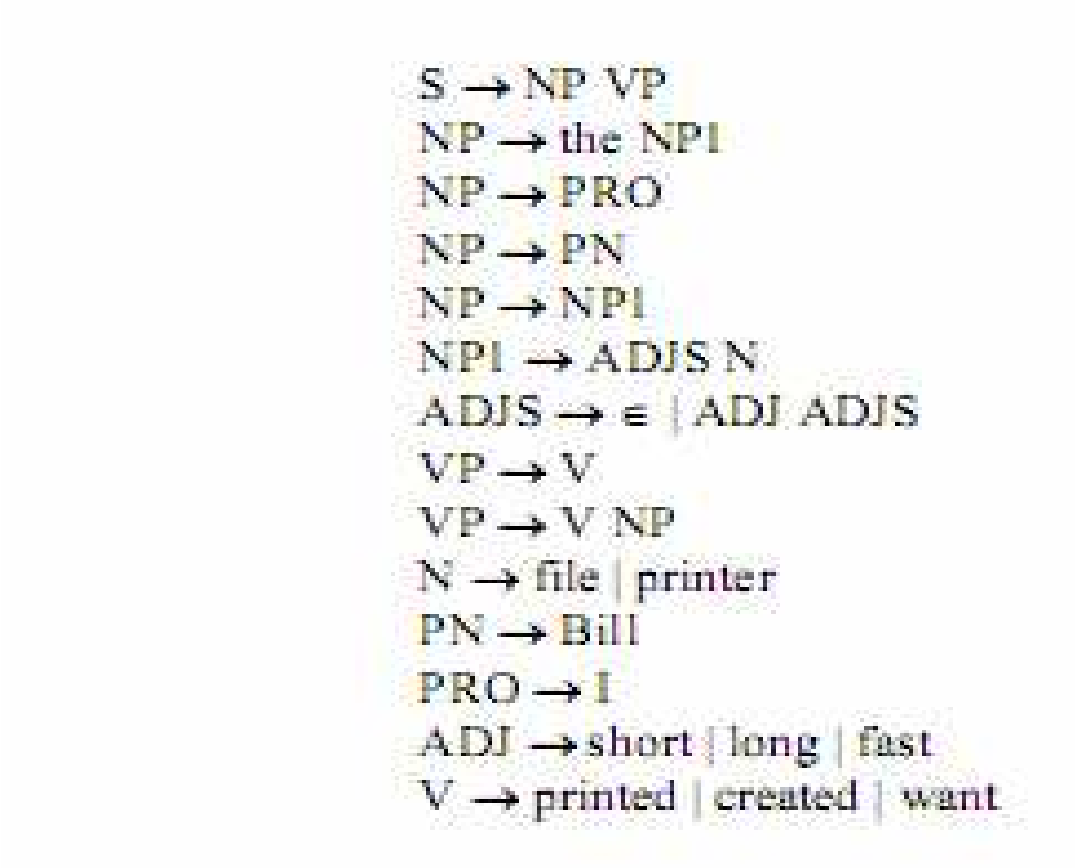
Παρόλο που είναι συχνά πιθανό να εξαχθεί το νόημα μιας πρότασης δίχως την εφαρμογή γραμματικών κανόνων, αυτό δεν είναι πάντοτε δυνατό. Θεωρήστε για παράδειγμα τις εξής προτάσεις:

- The satellite orbited Mars.
- Mars orbited the satellite.

Στη δεύτερη πρόταση, οι συντακτικοί κανόνες δίνουν μια ερμηνεία, κατά την οποία ένας πλανήτης (Mars) περιστρέφεται γύρω από ένα δορυφόρο, παρά το γεγονός ότι αυτό είναι ένα απίθανο σενάριο.

Αν και υπάρχουν πολλοί τρόποι να παραχθεί μια ανάλυση (parse), σχεδόν όλα τα χρησιμοποιούμενα συστήματα έχουν δυο κύρια συστατικά:

- Μια ρητή αναπαράσταση, που καλείται γραμματική (grammar), των συντακτικών κανόνων της γλώσσας.
- Μια διαδικασία, που καλείται αναλυτής (parser), που συγκρίνει τη γραμματική με προτάσεις εισόδου για να παράγει συντακτικές δομές.



Σχήμα 3.5 : Μια απλή γραμματική για ένα τμήμα της Αγγλικής γλώσσας

3.2.1 ΓΡΑΜΜΑΤΙΚΕΣ ΚΑΙ ΑΝΑΛΥΤΕΣ

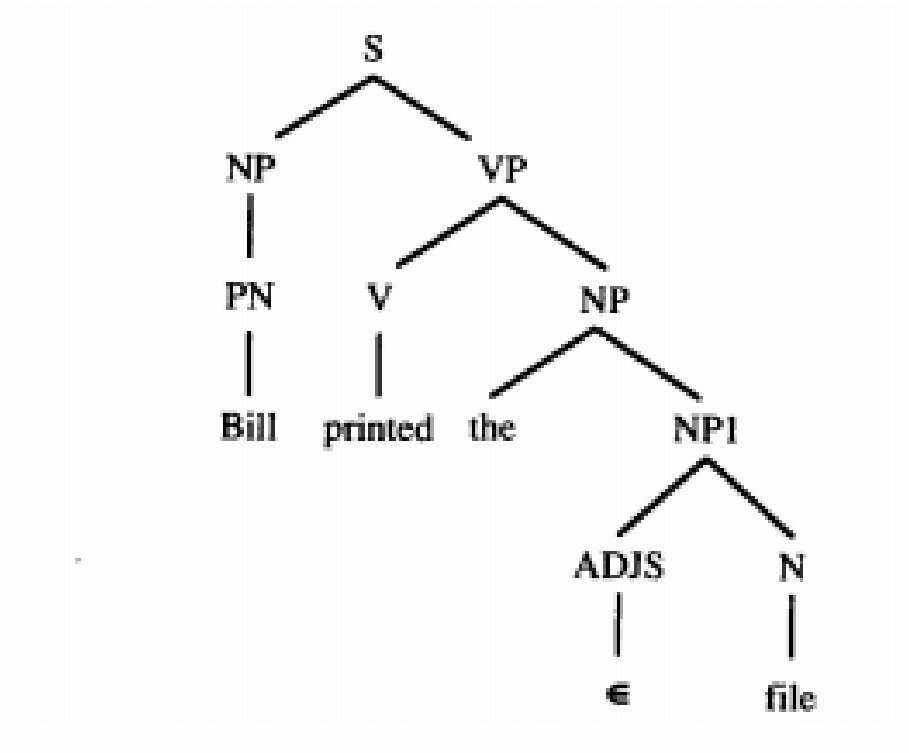
Ο πιο κοινός τρόπος να αναπαραστήσουμε τη γραμματική είναι ένα σύνολο κανόνων παραγωγής. Παρόλο που οι λεπτομέρειες των επιτρεπτών φορμών των κανόνων διαφέρουν, η βασική ιδέα παραμένει ίδια και φαίνεται στο Σχήμα 2.6,

που δείχνει μια απλή γραμματική ανεξάρτητη συμφραζομένων (context-free grammar) για την αγγλική.

Ο πρώτος κανόνας λέει ότι «Μια πρόταση αποτελείται από μια ονοματική φράση, ακολουθούμενη από μια ρηματική φράση». Στη γραμματική αυτή, η κάθετη μπάρα αντιπροσωπεύει το λογικό «ή». Το $\in$ δηλώνει την κενή συμβολοσειρά. Τα σύμβολα που αναπτύσσονται περισσότερο από τους κανόνες ονομάζονται μη-τερματικά σύμβολα (nonterminal symbols). Αυτά που αντιστοιχούν απ' ευθείας σε συμβολοσειρές μιας πρότασης εισόδου, λέγονται τερματικά σύμβολα (terminal symbols).

Γραμματικοί φορμαλισμοί όπως αυτός υπόκεινται σε πολλές γλωσσολογικές θεωρίες. Αυτές με τη σειρά τους παρέχουν τη βάση για πολλά συστήματα κατανόησης φυσικής γλώσσας. Πρέπει να επισημάνουμε ότι υπάρχει μια γενική συμφωνία ότι οι καθαρές context-free γραμματικές δεν είναι αποτελεσματικές στο να περιγράφουν φυσικές γλώσσες.

Ως αποτέλεσμα, τα συστήματα επεξεργασίας φυσικής γλώσσας έχουν λιγότερες ομοιότητες με τα συστήματα επεξεργασίας γλώσσας υπολογιστών από όσες ίσως θα περιμένατε. Ανεξάρτητα από τη θεωρητική βάση της γραμματικής, η διαδικασία της ανάλυσης παίρνει τους κανόνες της γραμματικής και τους συγκρίνει με την πρόταση εισόδου. Κάθε κανόνας που ταιριάζει προσθέτει κάτι για την ολοκλήρωση της δομής που σχηματίζει η πρόταση. Η πιο απλή δομή που μπορεί να δημιουργηθεί, είναι ένα συντακτικό δέντρο (parse tree), το οποίο καταγράφει τους κανόνες και το πώς συνταιριάζονται. Το Σχήμα 3.7 δείχνει το συντακτικό δέντρο που παράγεται από την πρόταση "Bill printed the file" χρησιμοποιώντας αυτή τη γραμματική. Το Σχήμα 3.1 περιείχε ένα άλλο παράδειγμα συντακτικού δέντρου, ωστόσο απαιτούνται κάποιες προσθήκες σε αυτή τη γραμματική για να παραχθεί.



Σχήμα 3.7 : Το συντακτικό δέντρο μιας πρότασης

Παρατηρήστε ότι κάθε κόμβος του συντακτικού δέντρου αντιστοιχεί είτε σε μια λέξη εισόδου, είτε σε ένα μη-τερματικό σύμβολο της γραμματικής. Κάθε επίπεδο του δέντρου αντιστοιχεί στην εφαρμογή ενός γραμματικού κανόνα. Συνεπώς, παρατηρούμε ότι μια γραμματική ορίζει δυο πράγματα για μια γλώσσα:

- Την ασθενή παραγωγική ικανότητά της, με την οποία εννοούμε ένα σύνολο από προτάσεις που περιέχονται μέσα στη γλώσσα. Το σύνολο αυτό (που ονομάζεται σύνολο γραμματικών προτάσεων), αποτελείται από εκείνες ακριβώς τις προτάσεις που μπορούν να ταιριάξουν με μια σειρά κανόνων της γραμματικής.
- Την ισχυρή παραγωγική ικανότητά της, με την οποία εννοούμε τη δομή (ή τις δομές) που προσαρτώνται σε κάθε γραμματικά-ορθή πρόταση της γλώσσας.

Έως τώρα, δείξαμε ότι τα αποτελέσματα της ανάλυσης είναι στην ουσία μία καταγραφή των κανόνων που εφαρμόστηκαν. Όμως αυτό δεν ισχύει πάντα. Ορισμένες γραμματικές περιέχουν επιπρόσθετη πληροφορία που περιγράφει το πώς πρέπει να φτιαχτεί η δομή.

Πρώτα όμως χρειάζεται να μελετηθούν δυο σημαντικά ζητήματα που ορίζουν το χώρο των δυνατών αναλυτών, που μπορούν να εκμεταλλευθούν τις γραμματικές που γράφουμε.

3.2.2 TOP-DOWN ΚΑΙ BOTTOM-UP ΑΝΑΛΥΣΗ

Για να αναλύσουμε μια πρόταση, χρειάζεται να βρούμε έναν τρόπο σχηματισμού της πρότασης από κάποιο αρχικό σύμβολο. Υπάρχουν δυο τρόποι για να γίνει αυτό:

- Top Down Ανάλυση (Από την Κορυφή προς τη Βάση) – Ξεκινώντας από ένα αρχικό σύμβολο, εφαρμόζουμε τους γραμματικούς κανόνες προς τα εμπρός, έως ότου τα σύμβολα στα τερματικά σημεία του δέντρου αντιστοιχισθούν με κατάλληλα μέρη της πρότασης.
- Bottom Up Ανάλυση (Από τη Βάση προς την Κορυφή) – Ξεκινώντας από την υπό ανάλυση πρόταση, εφαρμόζουμε τους γραμματικούς κανόνες αντίστροφα, έως ότου δημιουργηθεί ένα μοναδικό δέντρο, του οποίου τα τερματικά σημεία είναι λέξεις της πρότασης και η ρίζα του είναι το αρχικό σύμβολο.

Η επιλογή ανάμεσα στις δυο παραπάνω προσεγγίσεις, είναι παρόμοια με την επιλογή για ευθεία ή αντίστροφη συλλογιστική στην επίλυση προβλημάτων. Την απόφαση καθορίζει κυρίως ο παράγοντας διακλάδωσης. Είναι μεγαλύτερος όταν πηγαίνουμε προς τα εμπρός ή προς τα πίσω;

Ένα άλλο ζήτημα είναι το πόσο καλά ευρετικά στοιχεία (heuristics) έχουμε κατά τη διάρκεια της αξιολόγησης. Υπάρχει πληροφορία που να μπορεί να χρησιμοποιηθεί για τον αποκλεισμό κανόνων σχετικά νωρίς; Ορισμένες φορές οι παραπάνω μέθοδοι συνδυάζονται και σχηματίζουν τη μέθοδο που λέγεται «bottom-up ανάλυση με top-down φιλτράρισμα». Σε αυτήν, η ανάλυση προχωρά ουσιαστικά bottom-up (δηλαδή οι γραμματικοί κανόνες εφαρμόζονται αντίστροφα). Όμως, μερήση πινάκων που έχουν προϋπολογισθεί για μια συγκεκριμένη γραμματική, ο αναλυτής μπορεί άμεσα να αφαιρέσει συστατικά που δεν πρόκειται ποτέ να συνθέσουν μια χρήσιμη δομή υψηλότερου επιπέδου.

3.2.3 ΒΡΙΣΚΟΝΤΑΣ ΜΙΑ Η ΠΕΡΙΣΣΟΤΕΡΕΣ ΕΡΜΗΝΕΙΕΣ

Όπως έχουν δείξει αρκετά παραδείγματα, η διαδικασία κατανόησης μιας πρότασης είναι μια διαδικασία αναζήτησης κατά την οποία πρέπει να εξετασθεί ένας χώρος πιθανών ερμηνειών για να βρεθεί μια ερμηνεία που τηρεί τους περιορισμούς που επιβάλλει η πρόταση. Όπως και σε οποιαδήποτε διαδικασία αναζήτησης, πρέπει να αποφασισθεί αν θα ερευνηθούν όλα τα πιθανά μονοπάτια ή αν θα εξετασθεί ένα πιθανό μονοπάτι, το οποίο και θα εξαχθεί ως απάντηση.

Ας υποθέσουμε, για παράδειγμα, ότι ένας επεξεργαστής προτάσεων κοιτάζει τις λέξεις μιας πρότασης εισόδου, μία κάθε φορά, από τα αριστερά προς τα δεξιά και έστω ότι μέχρι στιγμής έχει δει τα εξής:

“Have the students who missed the exam –”

Υπάρχουν δυο μονοπάτια που μπορεί να ακολουθήσει ο επεξεργαστής:

- Το “have” είναι το κύριο ρήμα μιας πρότασης στην προστακτική, όπως η πρόταση “Have the students who missed the exam take it today.”
- Το “have” είναι βοηθητικό ρήμα μιας ερωτηματικής πρότασης όπως “Have the students who missed the exam taken it today?”

Υπάρχουν τέσσερις τρόποι χειρισμού τέτοιων προτάσεων:

- Όλα τα μονοπάτια – Ακολουθούμε όλα τα πιθανά μονοπάτια και δημιουργούμε όλα τα πιθανά ενδιάμεσα συστατικά. Πολλά από αυτά αργότερα θα αγνοηθούν επειδή δε θα εμφανισθούν οι απαραίτητες είσοδοι. Για παράδειγμα, αν έχει βρεθεί η ερμηνεία τουβοηθητικού ρήματος “have” στο προηγούμενο παράδειγμα, θα διαγραφεί αν δεν εμφανισθεί καμία μετοχή όπως ή “taken”. Το μεγαλύτερο μειονέκτημα αυτής της μεθόδου είναι ότι κατασκευάζονται παραπλανητικά συστατικά και ακολουθούνται πολλά αδιέξοδα μονοπάτια, με αποτέλεσμα να την καθιστούν αναποτελεσματική.
- Το καλύτερο μονοπάτι με οπισθοδρόμηση – Κάθε φορά ακολουθούμε μόνο ένα μονοπάτι αλλά καταγράφουμε σε κάθε σημείο απόφασης την απαραίτητη πληροφορία για να λάβουμε μια διαφορετική απόφαση όταν το επιλεγμένο μονοπάτι αποτύχει να μας οδηγήσει στην επιτυχή ερμηνεία της πρότασης. Στο παράδειγμά μας, αν είχε επιλεγεί η ερμηνεία του “have” σαν βοηθητικό ρήμα και δεν εμφανιζόταν το κύριο ρήμα στο τέλος της πρότασης, το σύστημα

θα ανίχνευε λάθος και θα επέστρεφε για να δοκιμάσει ένα άλλο μονοπάτι. Υπάρχουν δυο σημαντικοί ανασταλτικοί παράγοντες. Ο πρώτος είναι ότι χάνεται πολύτιμος χρόνος επειδή πρέπει να καταγράφονται οι καταστάσεις κάθε σημείου επιλογής, αν και τελικά η οπισθοδρόμηση θα εκτελεστεί μόνο σε μερικά από αυτά. Ο δεύτερος είναι ότι συχνά το ίδιο συστατικό αναλύεται πολλές φορές. Στο παράδειγμά μας, αν επιλεγεί η λανθασμένη ερμηνεία για τη λέξη “have”, αυτό δε θα αποκαλυφθεί παρά μόνο μετά την ερμηνεία του “the students who missed the exam”. Μόλις ανιχνευθεί το λάθος, ένας απλός μηχανισμός οπισθοδρόμησης θα αναιρέσει οτιδήποτε υπάρχει μετά το “have”, και η ουσιαστική φράση θα επανερμηνευθεί (αυτούσια) μετά από την επιλογή της δεύτερης ερμηνείας του “have”. Το πρόβλημα μπορεί να αντιμετωπισθεί αν χρησιμοποιήσουμε ένα είδος οπισθοδρόμησης καθοδηγούμενης από εξαρτήσεις. Τότε όμως, η υλοποίηση του αναλυτή θα είναι πιο πολύπλοκη.

- Καλύτερο μονοπάτι με patchup (διόρθωση) - Κάθε φορά ακολουθούμε μόνο ένα μονοπάτι αλλά όταν ανιχνευθεί ένα λάθος, μετατοπίζουμε τα συστατικά που έχουν ήδη δημιουργηθεί. Ξαναγυρνώντας στο παράδειγμα μας, αν είχε επιλεγεί η ερμηνεία του “have” ως βοηθητικό ρήμα, η ονοματική φράση “The students who missed the exam” θα ερμηνευόταν και θα καταχωρούταν ως το υποκείμενο της πρότασης. Αν η λέξη “taken” εμφανισθεί μετά, το μονοπάτι μπορεί να συνεχισθεί. Αν όμως εμφανισθεί το “take”, το σύστημα απλώς θα μετατοπίσει τα συστατικά σε διαφορετικές θέσεις. Το “have” γίνεται το κύριο ρήμα. Η ουσιαστική φράση που είχε χαρακτηρισθεί ως υποκείμενο της πρότασης γίνεται υποκείμενο της πρότασης “The students who missed the exam take it today”. Το υποκείμενο της κύριας πρότασης συμπληρώνεται με το “you”, το προκαθορισμένο υποκείμενο για προστακτικές προτάσεις. Αυτή η μέθοδος είναι συνήθως αποτελεσματικότερη των δυο προηγουμένων. Το κύριο μειονέκτημα είναι ότι απαιτεί αλληλεπίδραση των κανόνων της γραμματικής ώστε να γίνονται κανόνες για τη μετακίνηση των συστατικών από το ένα μέρος στο άλλο. Ο διερμηνευτής συχνά γίνεται επί τούτου (ad hoc) αντί να καθοδηγείται από τη γραμματική.
- Αναμονή και παρατήρηση (Wait and See) – Ακολουθούμε ένα μονοπάτι, αλλά αντί να λαμβάνουμε απόφαση τη στιγμή που συναντούμε ένα στοιχείο,

καθυστερούμε την απόφαση έως ότου υπάρχει επαρκής πληροφορία για να λάβουμε τη σωστή. Χρησιμοποιώντας αυτή τη μέθοδο, όταν συναντήσουμε τη λέξη “have” του παραδείγματός μας, καταγράφουμε ότι πρόκειται για κάποιο είδος ρήματος με άγνωστη έως τώρα λειτουργία. Η επόμενη ονοματική φράση, απλώς θα ερμηνευόταν και θα καταγραφόταν ως τέτοια. Έπειτα, όταν συναντήσουμε την επόμενη λέξη, μπορούμε να αποφασίσουμε για το πώς μπορούμε να συνδυάσουμε όλα τα στοιχεία που έχουμε βρει έως τώρα. Αν και αρκετοί αναλυτές κάνουν ένα είδος χρήσης της τεχνικής αναμονής και παρατήρησης, ένας μόνο, ο PARSIFAL [Marcus, 1980] βασίζεται σε αυτήν αποκλειστικά. Χρησιμοποιεί ένα μικρό, προκαθορισμένου μεγέθους ενταμιευτή (buffer) στον οποίο αποθηκεύονται τα στοιχεία μέχρι να αποφασισθεί η λειτουργικότητά τους. Η μέθοδος αυτή, είναι πολύ αποτελεσματική, αλλά έχει το εξής μειονέκτημα: Αν η απαιτούμενη πρόβλεψη ξεπερνά το μέγεθος του ενταμιευτή, ο ερμηνευτής αποτυγχάνει. Όμως, οι προτάσεις στις οποίες αποτυγχάνει είναι αυτές που και οι ίδιοι οι άνθρωποι έχουν πρόβλημα, προφανώς επειδή διαλέγουν λανθασμένη ερμηνεία. Κλασσικό παράδειγμα αυτού του φαινομένου, το οποίο ονομάζεται πρόταση των διακλαδωμένων μονοπατιών (garden path sentence), είναι η ακόλουθη πρόταση:

The horse raced past the barn fell down.

Αν και το πρόβλημα της απόφασης για το ποια μονοπάτια θα πρέπει να ακολουθηθούν και πώς πρέπει να χειριστούμε την οπισθοδρόμηση, είναι κοινό για όλες τις διαδικασίες αναζήτησης, περιπλέκεται στην περίπτωση της κατανόησης φυσικής γλώσσας, λόγω της παρουσίας ασαφών προτάσεων. Μια τέτοια πρόταση “They are flying planes” είδαμε σε προηγούμενο παράδειγμα. Αν ίναι σημαντικό να βρεθούν όλες οι πιθανές ερμηνείες και όχι μόνο μία, τότε θα πρέπει να ακολουθηθούν όλα τα πιθανά μονοπάτια (πράγμα το οποίο κοστίζει, αφού τα περισσότερα καταργούνται πριν από το τέλος των υπολογισμών). Πολλά συστήματα ικανοποιούνται αν βρουν μια εύλογη ερμηνεία. Αν αυτή απορριφθεί αργότερα, πιθανόν για σημασιολογικούς ή πραγματολογικούς λόγους, τότε επιχειρείται μια νέα προσπάθεια εύρεσης μίας διαφορετικής ερμηνείας.

3.2.4 ΑΝΑΣΚΟΠΗΣΗ ΑΝΑΛΥΤΩΝ

Όπως υπονοεί ο τίτλος αυτής της ενότητας, υπάρχουν πολλά είδη αναλυτών. Τρία από αυτά χρησιμοποιούνται ευρέως στα συστήματα κατανόησης φυσικής γλώσσας:

- Διαγραμματικοί αναλυτές (chart parsers) [Winograd, 1983], που παρέχουν έναν τρόπο αποφυγής δημιουργίας αντιγράφων με το να αποθηκεύουν ενδιάμεσα συστατικά ώστε να επαναχρησιμοποιηθούν σε εναλλακτικά μονοπάτια.
- Γραμματικές οριστικών προτάσεων (definite clause grammars) [Pereira και Warren, 1980], όπου οι γραμματικοί κανόνες είναι γραμμένοι ως προτάσεις PROLOG, και ο μεταφραστής της PROLOG κάνει top-down ανάλυση σε βάθος.
- Επαυξημένα δίκτυα μετάβασης (augmented transition networks) ή ΕΔΜ [Woods, 1970], στα οποία η διαδικασία ανάλυσης περιγράφεται ως η μετάβαση από μία αρχική κατάσταση σε μία τελική, μέσα σε ένα δίκτυο μεταβάσεων που αντιστοιχεί σε μια γραμματική της φυσικής γλώσσας.

ΚΕΦΑΛΑΙΟ 4

GATE FRAMEWORK

Το πλάισιο GATE (GATE Framework) είναι ένα πλαίσιο και γραφικό περιβάλλον ανάπτυξης το οποίο επιτρέπει στους χρήστες του να αναπύξουν πόρους και προγράμματα σχετικά με ανάπτυξη φυσικής γλώσσας (Natural language processing) και γλωσσικής μηχανικής (language engineering). Η αρχιτεκτονική GATE επιτρέπει όχι μόνο τη δημιουργία εφαρμογών για σχετικά με διάφορες λειτουργίες NLP (όπως την εξόρυξη πληροφοριών) αλλα και την δημιουργία και την σημείωση (annotation) σωμάτων κειμένου (corpora) και την αξιολόγηση των εφαρμογών που δημιουργήθηκαν. Το πλαίσιο μπορεί να χρησιμοποιηθεί για τη δημιουργία εφαρμογών και πόρων (resources) σε πολλές διαφορετικές γλώσσες, βασισμένο στην υποστήριξη που υπάρχει για το Unicode.

Η επιτυχημένη ανάπτυξη μερών εφαρμογών (components) με σκοπό την επεξεργασία της ανθρώπινης γλώσσας είναι μια πολύπλοκη διαδικασία όσον αφορά την δημιουργία τους. Το GATE βοηθάει με πολλούς διαφορετικούς τρόπους στον τομέα αυτό :

- Το σύστημα είναι σχεδιασμένο έτσι ώστε να ξεχωρίζει εμφανώς ανάμεσα σε εργασίες χαμηλού επιπέδου όπως αποθήκευση δεδομένων, οπτικοποίηση δεδομένων, εντοπισμό και φόρτωση components και εκτέλεση εφαρμογών από τις δομές δεδομένων και τους αλγόριθμους οι οποίοι πραγματικά επεξεργάζονται ανθρώπινη γλώσσα.
- Αυτόματη μέτρηση της απόδοσης των components της επεξεργασίας γλώσσας
- Μείωση των overheads της ενοποίησης προσφέροντας τυπικούς μηχανισμούς για την επικοινωνία μεταξύ components σχετικά με τη γλώσσα, και θέση ανοιχτών προτύπων όπως η Java και η XML ως υποπλατφόρμα.
- Παροχή ενός βασικού συνόλου components επεξεργασίας γλώσσας τα οποία θα μπορούν να επεκταθούν και να αντικατασταθούν από χρήστες κατά βούληση

Το GATE είναι :

- Αρχιτεκτονική (Architecture)
- Πλαίσιο (Framework)
- Περιβάλλον ανάπτυξης (Development environment)

Σκοπός του είναι να διεκπεραιώσει στόχους σχετικούς με την Μηχανική κειμένου (Language engineering). Ως αρχιτεκτονική, προσδιορίζει την οργάνωση ενός συστήματος μηχανικής κειμένου και την απόδοση αρμοδιοτήτων σε διάφορα στρώματα πρόσβασης (access layers) σε μια βάση δεδομένων. Εναλλακτικά, μπορεί να εφαρμόσει ολόκληρο το σύστημα.

4.1 ΑΛΓΟΡΙΘΜΟΙ, ΔΕΔΟΜΕΝΑ ΚΑΙ ΕΦΑΡΜΟΓΕΣ ΓΡΑΦΙΚΟΥ ΠΕΡΙΒΑΛΛΟΝΤΟΣ

Ένα απο τα πιο σημαντικά στοιχεία του GATE είναι ο διαχωρισμός που υπάρχει μεταξύ των δεδομένων, των αλγόριθμων και των τρόπων οπτικοποίησης τους. Με άλλα λόγια, τα components του GATE μπορεί να είναι τριών ειδών :

- **Γλωσσικοί πόροι (Language Resources ή LR).** Αναπαριστούν οντότητες όπως λεξικά, σώματα (corpora) ή οντολογίες.
- **Επεξεργαστικοί πόροι (Processing Resources ή PR)** αναπαριστούν οντότητες οι οποίες είναι πρωταρχικά αλγοριθμικές, όπως οι αναλυτές (parsers), οι γεννήτριες (generators) ή οι ngram modelers.
- **Οπτικοί πόροι (Visual resources ή VR)** οι οποίοι αναπαριστούν οπτικοποιήσεις ή άλλα οπτικά στοιχεία τα οποία συμμετέχουν στο γραφικό περιβάλλον χρήστη.

Αυτοί οι πόροι μπορούν να είναι τοπικοί και να υπάρχουν στο μηχάνημα του χρήστη ή να είναι απομακρυσμένοι και να χρησιμοποιούνται μέσω HTTP. Σε κάθε περίπτωση, όλοι μπορούνε να επεκταθούν απο τους χρήστες χωρίς να υπάρξει μεταβολή στο ίδιο το GATE.

Ένα απο τα κύρια πλεονεκτήματα που εχει ο διαχωρισμός των αλγορίθμων απο τα δεδομένα τα οποία απαιτούν είναι το γεγονός ότι μπορούν να αναπτυχθούν ανεξάρτητα μεταξύ τους απο επιστήμονες διαφορετικών ειδικοτήτων, π.χ.

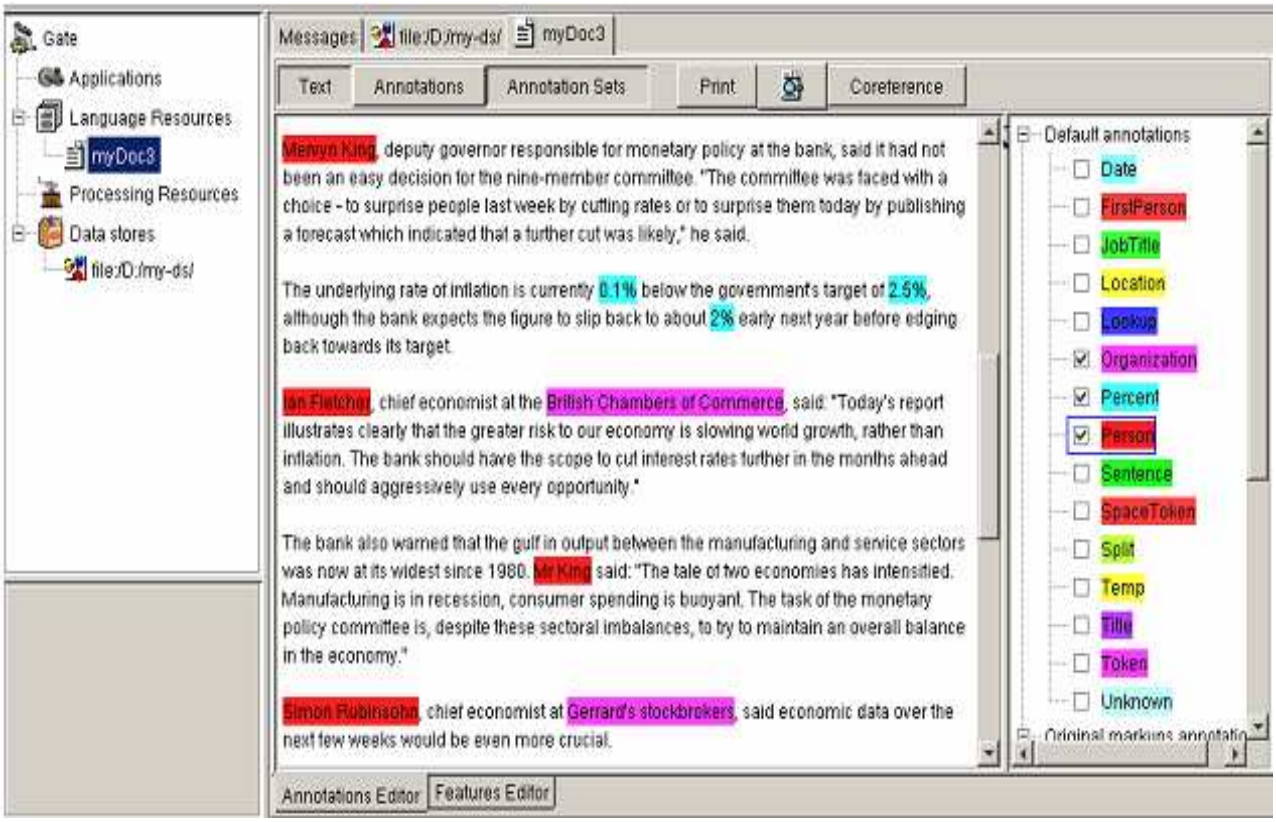
προγραμματιστές και γλωσσολόγους. Παρομοίως, ο διαχωρισμός των δεδομένων απο τα στοιχεία της οπτικοποίησής τους επιτρέπει στους χρήστες να αναπτύξουν εναλλακτικούς οπτικούς πόρους, χρησιμοποιώντας ένα στοιχείο γλωσσικού πόρου το οποίο παρέχεται ήδη απο το GATE.

Συνολικά, όλοι οι πόροι είναι γνωστοί και ώς CREOLE (A Collection of REusable Objects for Language Enginnering), και δηλώνονται σε ένα αρχείο XML, το οποίο περιγράφει το όνομα τους, την κλάση που ενσωματώνεται, τους παραμέτρους κλπ. Αυτή η αποθήκη εργαλείων χρησιμοποιείται απο το πλαίσιο με σκοπό τον εντοπισμό και τη χρήση καινούργιων πόρων.

Ένα tag “Parameters” περιγράφει τις παραμέτρους τις οποίες χρειάζεται κάθε πόρος όταν δημιουργείται ή εκτελείται. Οι παράμετροι μπορεί να είναι και προαιρετικοί π.χ. όταν δημιουργείται ένα corpus, άν παρέχεται μαζί ώς παράμετρος και μια λίστα εγγράφων, τότε τα έγγραφα αυτόματα θα γίνονται μέρος του corpus.

Όταν μια εφαρμογή αναπτύσσεται μέσα στο γραφικό περιβάλλον του GATE, ο χρήστης διαλέγει ποιούς επεξεργαστικούς πόρους θα χρησιμοποιήσει, με ποιά σειρά θα εκτελούνται, και πάνω σε ποιά δεδομένα (έγγραφο ή corpus).

Οι παράμετροι εκτέλεσης κάθε πόρου επίσης δηλώνονται εκεί π.χ. ένα έγγραφο δίνεται ώς παράμετρος σε κάθε PR. Όταν τρέχει η εφαρμογή, τα PR θα εκτελούνται στην προκαθορισμένη σειρά και στο δεδηλωμένο κείμενο. Μπορούμε να δούμε τα αποτελέσματα στον επεξεργαστή κειμένου του GATE (Σχήμα 4.1).



Σχήμα 4.1: Το γραφικό περιβάλλον GATE Developer

4.2 ΑΝΑΠΑΡΑΣΤΑΣΗ ΚΑΙ ΔΙΑΧΕΙΡΙΣΗ ΔΕΔΟΜΕΝΩΝ

Το GATE υποστηρίζει πολλούς τύπους αναπαράστασης δεδομένων, για παράδειγμα:

- XML
- RTF
- HTML
- SGML
- E-mail
- Απλό κείμενο

Σε κάθε περίπτωση, όταν ένα κείμενο δημιουργείται ή ανοίγεται στο GATE, ο τύπος του κειμένου αναλύεται και μετατρέπεται σε ένα απλό και ενοποιημένο μοντέλο σημειώσεων (annotation model). Ο τύπος των σημειώσεων είναι μια εξελιγμένη μορφή του προτύπου TIPSTER, το οποίο είναι επαρκώς συμβατό με το πρότυπο Atlas.

Οι σημειώσεις που σχετίζονται με κάθε κείμενο είναι μια δομή κεντρική στο GATE, επειδή κωδικοποιούνε την προσπέλαση γλωσσικών δεδομένων και παράγονται απο κάθε επεξεργαστικό πόρο.

Το GATE επίσης παρέχει δυνατότητα μόνιμης αποθήκευσης των γλωσσικών πόρων. Στην παρούσα φάση προσφέρει τρείς μηχανισμούς αποθήκευσης. Ο ένας χρησιμοποιεί σχεσιακές βάσεις δεδομένων (π.χ Oracle) και οι άλλοι 2 βασίζονται σε αρχεία και χρησιμοποιούν σειριοποίηση (serialization) τύπου java ή έναν εσωτερικό τύπο αποθήκευσης βασισμένο σε XML. Τα κείμενα GATE μπορούν επίσης να αποθηκευθούν και πάλι στη μορφή του αρχικού τους τύπου (π.χ. SGML, XML) και ο χρήστης μπορεί να επιλέξει αν θέλει να προστεθούν επιπλέον tags (για παράδειγμα πληροφορίες κύριων ονομάτων).

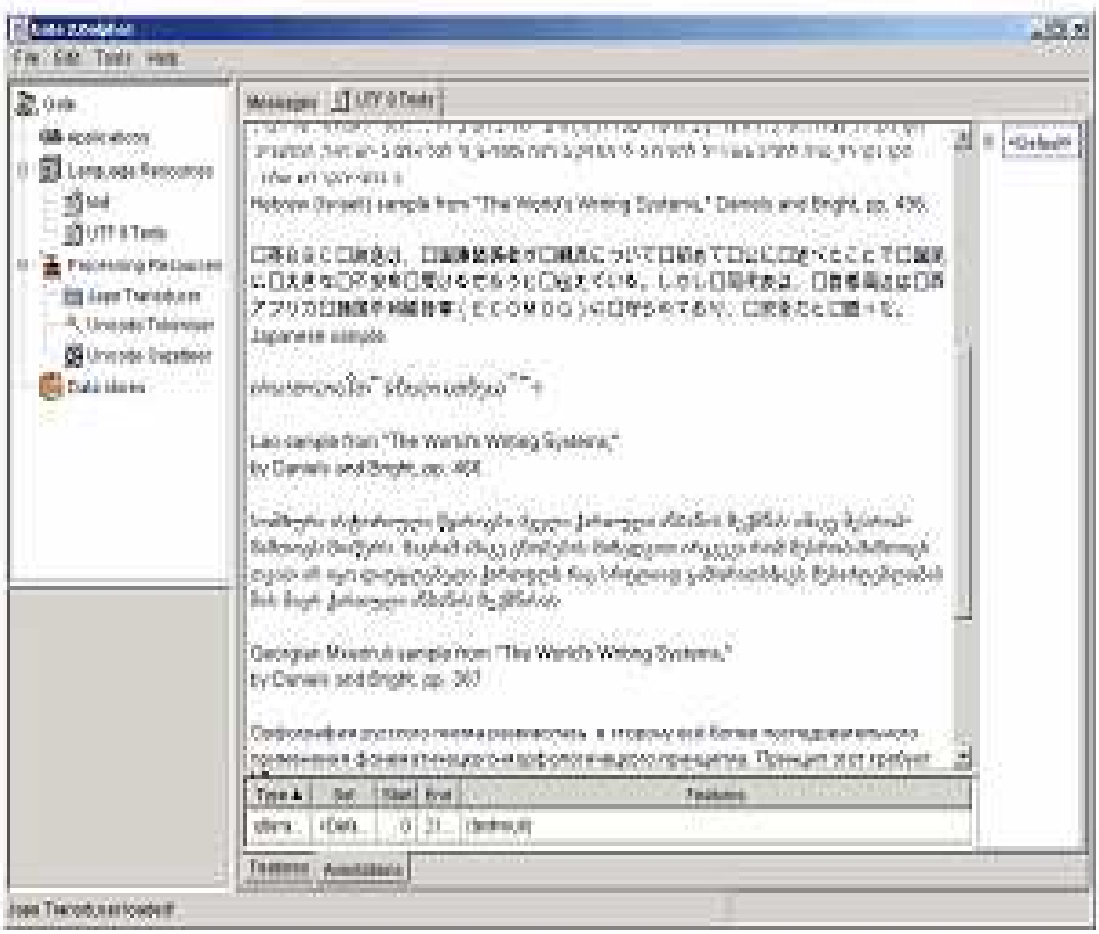
Έτσι συνοψίζοντας, βλέπουμε ότι η ύπαρξη μιας ενοποιημένης δομής δεδομένων εξασφαλίζει μια ομαλή επικοινωνία μεταξύ των μερών (components) του προγράμματος, ενώ η δυνατότητα εισαγωγών και εξαγωγών και μετατροπής δεδομένων σε διαφορετικούς τύπους (import-export) κάνουν εφικτή την απλουστευμένη επικοινωνία με τον «έξω κόσμο».

4.3 ΠΟΛΥΓΛΩΣΣΙΚΗ ΕΠΕΞΕΡΓΑΣΙΑ

Τα τελευταία χρόνια, έχει μεγαλώσει η ανάγκη στο λογισμικό για πολυγλωσσία (multilinguality) και έχει δοθεί μεγάλη έμφαση στην έρευνα σχετικά με το αντικείμενο. Πολύ σημαντική πρόοδος στον τομέα αυτό έχει γίνει με την επικράτηση της Unicode ώς επικρατέστερο σύστημα αναπαράστασης δεδομένων σε μορφή κειμένου.

Το GATE υποστηρίζει πολυγλωσσική επεξεργασία δεδομένων χρησιμοποιούντας την Unicode ώς προεπιλεγμένη μέθοδο κωδικοποίησης κειμένου. Επίσης είναι μια μέθοδος εισαγωγής κειμένου σε διάφορες γλώσσες, χρησιμοποιώντας εικονικα πληκτρολόγια όπου η γλώσσα δεν υποστηρίζεται απο την τρέχουσα υπολογιστική πλατφόρμα. Η Java υποστηρίζει χαρακτήρες σε μορφή Unicode, αλλά για πολλές γλώσσες που καλύπτονται απο το Unicode, η Java δεν υποστηρίζει εισαγωγή δεδομένων. Στο GATE, αυτή τη στιγμή υποστηρίζονται 28 γλώσσες, και σε επόμενες εκδόσεις σχεδιάζεται και επέκταση της υποστήριξης σε περισσότερες. Επειδή το GATE είναι ανοιχτής αρχιτεκτονικής, καινούργιου τύπου εικονικά πληκτρολόγια μπορούν να δημιουργηθούν και να χρησιμοποιηθούν απο το σύστημα για την κάλυψη

αναγκών. Για την προβολή του κειμένου, το GATE βασίζεται στις ευκολίες απόδοσης (rendering) για την εκάστοτε πλατφόρμα. Το σχήμα 4.2 δείχνει ένα παράδειγμα κειμένου σε δίαφορες γλώσσες στο GATE.



Σχήμα 4.2 : Κείμενο Unicode στο GATE

Η δυνατότητα διαχείρισης δεδομένων Unicode και ο διαχωρισμός μεταξύ δεδομένων και προγραμματισμού, επιτρέπει στα συστήματα διαχείρισης γλώσσας βασισμένα στο GATE να μεταφερθούν σε επιπλέον γλώσσες χωρίς επιπλέον επιφόρτωση εργασίας, εκτός απο την ανάπτυξη των πόρων που χρειάζονται για τη συγκεκριμένη γλώσσα. Τα χαρακτηριστικά αυτά αναπτύχθηκαν ώς επι το πλείστον ώς μέρος του προγράμματος EMILE, το οποίο εστιάζει στη δημιουργία ενός ηλεκτρονικού λεξικού 63 εκατομμυρίων λέξεως των γλωσσών της νότιας Ασίας.

4.4 ΕΦΑΡΜΟΓΕΣ ΤΟΥ GATE

Ένα απο τα χαρακτηριστικά του GATE, είναι και το γεγονός ότι είναι αρκετά ευέλικτο αλλα και σταθερό ώστε να υπάρχει δυνατότητα ανάπτυξης ενός μεγάλου εύρους εφαρμογών μέσα στο πλαίσιο του. Κάποιες εφαρμογές NLP που έχουν αναπτυχθεί στα πλαίσια της αρχιτεκτονικής GATE είναι οι εξής :

4.4.1 MUSE

Το σύστημα MUSE είναι ένα σύστημα αναγνώρισης λέξεων (Named entity recognition) με πολλαπλές χρήσεις. Είναι ικανό να επεξεργαστεί κείμενα πολύ διαφορετικών ειδών, με σκοπό τη μείωση της ανάγκης για ακριβή και χρονοβόρα κατανομή υπαρχόντων πόρων σε νέες εφαρμογές και πεδία. Το σύστημα στοχεύει στην ταυτοποίηση των παραμέτρων σχετικών με την δημιουργία ενός συστήματος αναγνώρισης ονομάτων μεταξύ διάφορων τύπων κειμένου. Για παράδειγμα, κείμενα λιγότερο τυπικά δεν τηρούν τους τυπικούς κανόνες χρήσης κεφαλαίων και μικρών (capitalization), σημείων στίξης και γραμματικής. Το γεγονός αυτό δημιουργεί προβλήματα στην εφαρμογή πολλών συστημάτων αναγνώρισης λέξεων. Οι εκτιμήσεις πιστότητας του συστήματος αυτού δείχνουν περίπου 93% ακρίβεια σε ένα δείγμα πολλών διαφορετικών τύπων κειμένου.

4.4.1 ACE

Το σύστημα MUSE έχει επίσης προσαρμοστεί ωστε να αποτελεί μέρος του τρέχοντος συστήματος ACE (Automatic Content Extraction) που χρησιμοποιείται απο το NIST (National Institute of Standards and Technology). Το ACE περιλαμβάνει αναγνώριση προτύπων (pattern recognition) ανάμεσα σε τρεις διαφορετικούς τύπους «καθαρών» κειμένων ειδήσεων (δελτία τύπου, ηλεκτρονικά μέσα ενημέρωσης και εφημερίδες) και δυο τύπους «αλλοιωμένων» ειδήσεων (έξοδος OCR και συστημάτων speech-to-text).

4.4.2 MUMIS

Το σύστημα MUMIS (Multimedia Indexing and Searching environment), χρησιμοποιεί τεχνολογία εξόρυξης πληροφοριών η οποία αναπτύχθηκε μέσα απο το

GATE με σκοπο τη δημιουργία τυπικών σημειώσεων σχετικά με τα σημαντικά γεγονότα που συμβαίνουν σε έναν ποδοσφαιρικό αγώνα.

Το σύστημα περιλαμβάνει κομμάτια του GATE όπως tokenization, ανίχνευση προτάσεων, POS-tagging και semantic tagging που έχουν αναπτυχθεί ώς δομικοί λίθοι του GATE. Επίσης όμως περιέχει διαδικασίες μορφολογικής και πλήρους συντακτικής ανάλυσης. Το module της σημειολογικής σημείωσης, στην παρούσα φάση πετυχαίνει περίπου 91% ακρίβεια, κάτι το οποίο είναι μια σημαντική βελτίωση σε σχέση με το βασικό σύστημα αναγνώρισης οντοτήτων.

4.5 PROCESSING RESOURCES

Μαζί με το GATE παρέχεται και ένα σύνολο απο επαναχρησιμοποιήσιμα processing resources για την επίτευξη κοινών εργασιών NLP. Δεν είναι αναντικατάστατοι, ο χρήστης μπορεί να αντικαταστήσει ή να επεκτείνει οποιονδήποτε απο αυτούς αν το κρίνει σκόπιμο. Στο σύνολο τους αποτελούν το ANNIE (A Nearly New Information Extraction system). Επίσης, μπορούν να χρησιμοποιηθούν και ο καθένας ξεχωριστά. Για παράδεγμα, πολλές άλλες εργασίες NLP μπορεί να χρειάζονται έναν sentence splitter και έναν POS tagger, αλλα δε χρειάζονται απαραίτητα πόρους πιο ειδικευμένους όπως έναν named entity transducer. Το σύστημα χρησιμοποιείται για μια πλειάδα απο εργασίες εξόρυξης πληροφοριών και άλλης φύσεως.

Τα κύρια processing resources του GATE είναι τα ακόλουθα:

- Tokenizer
- Sentence splitter
- POS tagger
- Gazetteer
- Orthomancer

Οι πόροι επικοινωνούν μέσω του annotation API του GATE, το οποίο είναι ένα στοχευμένο γράφημα απο ακμές που αντιπροσωπεύουν δεδομένα, και κόμβους που συνδέουν τα δεδομένα και το περιεχόμενο του κειμένου.

Ο **tokenizer** χωριζει το κείμενο σε tokens (αντικείμενα) όπως αριθμοί, σημεία στίξης, σύμβολα, και λέξεις διάφορων ειδών (π.χ. με το πρώτο γράμμα κεφαλαίο, όλα

κεφαλαία κλπ.). Ο στόχος είναι ο περιορισμός του φόρτου εργασίας του tokenizer με σκοπό τη μεγιστοποίηση της αποδοτικότητας, και η μεγαλύτερη ευελιξία. Αυτό επιτυγχάνεται με την μεταφορά της εργασίας της ανάλυσης στις γραμματικές. Αυτό σημαίνει ότι ο tokenizer δε χρειάζεται να διαφοροποιηθεί για διαφορετικές εφαρμογές ή τύπους κειμένων.

Ο **sentence splitter** είναι μια σειρά απο finite state transducers, οι οποίοι χωρίζουν το κείμενο σε προτάσεις. Χρησιμοποιεί μια λίστα απο συντομογραφίες τύπου gazetteer με τα οποία ξεχωρίζει τις τελείες οι οποίες τελειώνουν τις προτάσεις απο άλλου είδους τελείες.

Ο **POS Tagger** διαχωρίζει κάθε λέξη και σύμβολο δίνοντας πληροφορίες αναλογικά με ποίο μέρος του λόγου αντιτοιχεί σε αυτό. Το προεπιλεγμένο λεξικό του tagger και το σύνολο κανόνων (ruleset) τους οποίους ακολουθεί είναι αποτέλεσμα εξάσκησης σε ένα μεγάλο corpus που προέρχεται απο το Wall Street Journal. Και τα δυο μπορούν να τροποποιηθούν χειροκίνητα αν είναι απαραίτητο.

Ο ρόλος του **gazetteer** είναι εντοπίζει ονόματα οντοτήτων στο κείμενο παρμένα μέσα απο λίστες. Οι λίστες που χρησιμοποιεί είναι απλά αρχεία κειμένου, με μια περίπτωση ανα γραμμή. Κάθε λίστα αναπαριστά ένα διαφορετικό σύνολο ονομάτων, όπως ονόματα απο πόλεις, οργανισμούς, μέρες της εβδομάδας κλπ.

Ο **orthomancer** προσθέτει σχέσεις ταυτότητας σε named entities που βρίσκει ο semantic tagger, και τα συγκρίνει μεταξύ τους (coreference). Δεν βρίσκει καινούργια named entities απο μόνος του, αλλα μπορεί να δώσει έναν τύπο σε ένα κύριο όνομα που δεν έχει ακόμα ταυτοποιηθεί, χρησιμοποιόντας κάποιο που του ταιριάζει. Η παραπάνω μέθοδος χρησιμοποιείται μόνο αν τα ονόματα που γίνονται αντικείμενο σύγκρισης είναι όλα του ίδιου τύπου, π.χ. καί τα δυο οργανισμοί, ή αν ένα απο αυτά κατηγοριοποιείται ώς άγνωστος τύπος. Αυτό γίνεται για να περιοριστεί η επαναλαμβανόμενη κατηγοριοποίηση ενός ήδη κατηγοριοποιημένου ονόματος.

4.6 ΕΦΑΡΜΟΓΗ ΤΩΝ ΕΠΕΞΕΡΓΑΣΤΙΚΩΝ ΠΟΡΩΝ

Η εφαρμογή των επεξεργαστικών πόρων επικεντρώνεται στην χρηστικότητα, την σταθερότητα και τον ξεκάθαρο διαχωρισμό μεταξύ της δηλωτικής αναπαράστασης δεδομένων (declarative data representation) και τους αλγόριθμους πεπερασμένης κατάστασης (finite state algorithms). Η συμπεριφορά όλων των επεξεργαστών

ελέγχεται εξ ολοκλήρου απο τους εξωτερικούς πόρους, συγκεκριμένα τις γραμματικές και τα σύνολα κανόνων. Έτσι μπορούν εύκολα να προσαρμοστούν απο τους χρήστες, χωρίς να είναι η απαραίτητη η γνώση γλωσσών προγραμματισμού.

Το γεγονός ότι όλοι οι επεξεργαστικοί πόροι χρησιμοποιούν τεχνολογία finite state transducer, αυξάνει την απόδοσή τους όσον αφορά τον χρόνο εκτέλεσης. Πειράματα έδειξαν ότι ένα ολοκληρωμένο σύστημα αναγνώρισης οντοτήτων είναι ικανό να επεξεργαστεί περίπου 2.5 KB/s σε έναν Pentium III με 256MB RAM (ανεξάρτητα από το μέγεθος του αρχείου εισόδου. Οι επεξεργαστικές απαιτήσεις είναι γραμμικές σχετικά με το μέγεθος του κειμένου).

4.7 ΔΗΜΙΟΥΡΓΙΑ ΓΛΩΣΣΙΚΩΝ ΠΟΡΩΝ

Επείδή πολλοί αλγόριθμοι NLP χρειάζονται σώματα (corpora) με annotations για να «εξασκηθούν», το περιβάλλον ανάπτυξης του GATE παρέχει αρκετά εύχρηστα και εξελίξιμα εργαλεία σχετικά με το annotation κειμένου. Για τον έλεγχο της χρησιμότητας του στην πράξη, αυτά τα εργαλεία χρησιμοποιούνται για την δημιοθργία σωμάτων απο annotated κείμενα για τις εφαρμογές MUSE, ACE και MUMIS.

Το annotation μπορεί να γίνει χειροκίνητα απο τον χρήστη ή ημιαυτόματα, εφαρμόζοντας κάποιους επεξεργαστικούς πόρους πάνω στο σώμα και μετά προσθέτοντας ή αφαιρόντας καινούργια annotations χειροκίνητα. Ανάλογα με τις πληροφορίες που χρειάζεται να γίνουν annotated, κάποια κομμάτια του ANNIE μπορούν να χρησιμοποιηθούν ή να προσαρμοστούν για να διευκολύνουν το έργο του annotation του σώματος. Για παράδειγμα, σε μια εργασία NLP, δημιουργήθηκε χειροκίνητα ένας κατάλογος απο ονόματα απο τοποθεσίες του 18^{ου} αιώνα στο Λονδίνο. Αυτός ο κατάλογος, σε συνδυασμό με τον gazetteer του ANNIE, βοήθησε στο αυτόματο annotation των γεωγραφικών πληροφοριών σε μια μεγάλη συλλογή απο αναφορές που προέρχονταν απο τα δικαστήρια του Λονδίνου και εκδόθηκαν τον 18^{ο} αιώνα.

Επειδή το χειροκίνητο annotation είναι μια εργασία δύσκολη και μπορούν πολύ εύκολα να γίνουν λάθη, το GATE στοχεύει στο να κάνει την διαδικασία εύχρηστη και ταυτόχρονα και ευέλικτη. Για να προσθέσει ο χρήστης καινούργιο annotation, απλά θα πρέπει να επιλέξει το κείμενο με το ποντίκι και μετά να κάνει click στον επιθυμητο τύπο annotation, ο οποίος φαίνεται στην λίστα των τύπων annotations στην δεξιά πλευρά τις

επιφάνειας εργασίας του γραφικού περιβάλλοντος του GATE (Σχήμα 3.1). Οστόσω, άν ο επιθυμητός τύπος annotation δε βρίσκεται εκεί ή ο χρήστης επιθυμεί να ενσωματόσει πιο λεπτομερείς πληροφορίες στο annotation (και όχι μόνο το όνομά του), τότε μπορεί να χρησιμοποιηθεί ένας διάλογος επεξεργασίας annotation.

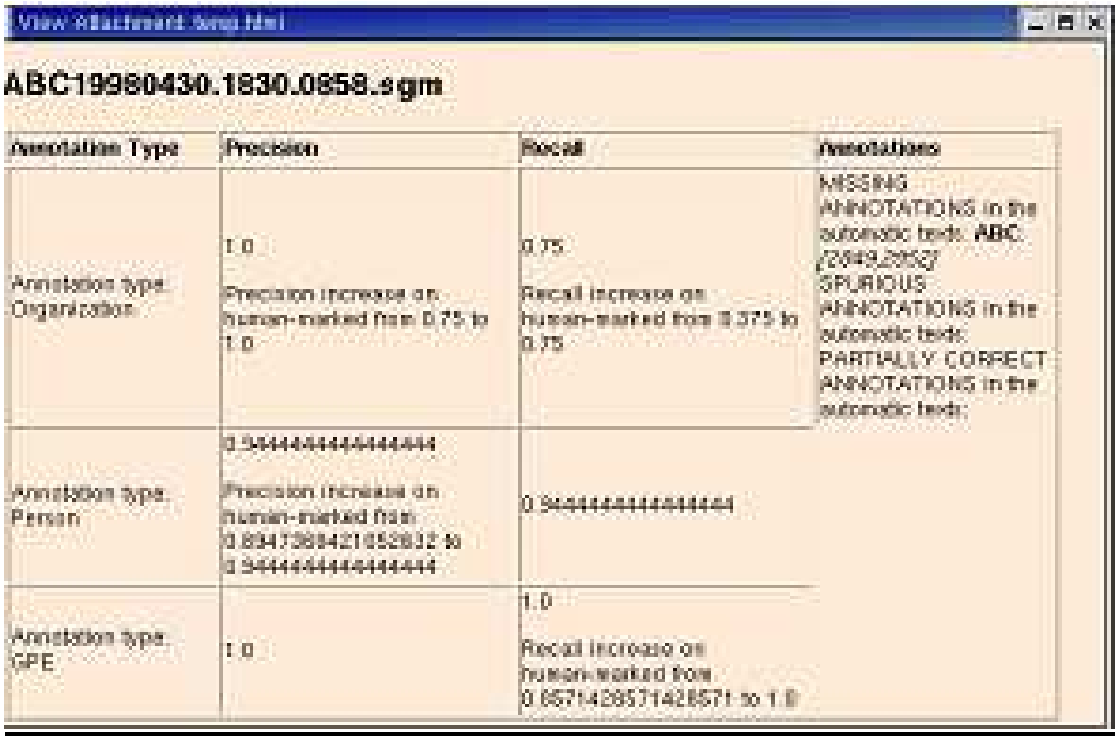
4.8 ΕΚΤΙΜΗΣΗ ΑΠΟΔΟΣΗΣ

Ένα πολύ σημαντικό κομμάτι κάθε εφαρμογης text engineering είναι και η εκτίμηση της απόδησης της, και κάθε περιβάλλον ανάπτυξης αυτού του τομέα δε θα ήταν ολοκληρωμένο αν δεν παρείχε μηχανισμούς για την μέτρηση της σε διάφορες περιπτώσεις ελέγχου. Το GATE παρέχει δυο τέτοιους μηχανισμούς : Ένα εργαλείο εκτίμησης (AnnottationDiff) το οποίο παρέχει αυτόματη μέτρηση απόδοσης και οπτικοποίηση των αποτελεσμάτων, και ένα εργαλείο συγκριτικής αξιολόγησης (benchmarking) το οποίο επιτρέπει την παρακολούθηση της προόδου του συστήματος και τον διευκολύνει τον έλεγχο με οπισθοδρόμηση.

4.8.1 ΤΟ ΕΡΓΑΛΕΙΟ ANNOTATIONDIFF

Το εργαλείο AnnotationDiff συγκρίνει δυο ομάδες απο annotations, με απώτερο στόχο είτε να συγκρίνει ένα κείμενο το οποίο έχει γίνει αυτόματα annotated (μέσα απο το σύστημα) με το ίδιο κείμενο αλλά χρησιμοποιώντας «χειροκίνητο» annotation, είτε να συγκρίνει την έξοδο δυο διαφορετικών συστημάτων αυτόματου annotation (ή δυο διαφορετικών παραλλαγών του ίδιου συστήματος). Για κάθε τύπο annotation, δημιουργούνται στατιστικά όσον αφορά την ακρίβεια, την ανάκληση, το F-measure και τα ψευδή θετικά.

Ο viewer του AnnotationDIff δείχνει τις δυο ομάδες από annotations, έχοντας το καθένα διαφορετικά χρώματα (παρόμοια με τα «οπτικά diff» MKS Toolkit και TkDiff). Τα annotations στην κύρια ομάδα (key set) μπορούν να έχουν δυο πιθανά χρώματα ανάλογα με την κατάστασή τους : λευκό για annotations που έχουν ένα συμβατό (ή μερικώς συμβατό) annotations στην απέναντι ομάδα, ή πορτοκαλί αν δεν υπάρχει αντίστοιχο. Τα annotations στην απέναντι ομάδα (response set) έχουν τρια πιθανά χρώματα. Πράσινο αν είναι συμβατά με τα αντίστοιχα της κύριας ομάδας, μπλέ αν είναι μερικώς συμβατά και κόκκινο αν είναι ασύμβατα.



Σχήμα 4.3 : Τμήμα αποτελεσμάτων απο το εργαλείο συγκριτικής αξιολόγησης

4.8.1 ΤΟ ΕΡΓΑΛΕΙΟ ΣΥΓΚΡΙΤΙΚΗΣ ΑΞΙΟΛΟΓΗΣΗΣ

Το εργαλείο συγκριτικής αξιολόγησης (benchmarking) του GATE διαφέρει απο το AnnotationDiff με την έννοια ότι με το συγκεκριμένο εργαλείο, η εκτίμηση μπορεί να εκτελεστεί πάνω σε ένα ολόκληρο corpus αντί για ένα μόνο έγγραφο. Επίσης, η απόδοση του συστήματος μπορεί να μετρηθεί και στην πάροδο του χρόνου.

Το εργαλείο χρειάζεται ένα καθαρό corpus (χωρίς annotations) και ένα ακόμα corpus με annotations (reference set). Σε πρώτη φάση, το εργαλείο εκτελείται σε generation mode με σκοπό να παραχτεί ένα σύνολο κειμένων τα οποία έχουν γίνει annotated απο το σύστημα. Αυτά τα κείμενα αποθηκεύονται για μελλοντική χρήση. Έπειτα, το εργαλείο μπορεί να εκτελεστεί με τρεις τρόπους :

- Σύγκριση του annotated set με το reference set
- Σύγκριση του annotated set με ένα άλλο set που παράγεται απο μια πιο πρόσφατη έκδοση των επεξεργαστικών πόρων του συστήματος (latest set)
- Σύγκριση του latest set με το reference set

Στην κάθε περίπτωση, θα παραχθούν στατιστικά σχετικά με την απόδοση κάθε κειμένου στο set, και γενικά στατιστικά για ολόκληρο το set σε σύγκριση με το

reference set . Το annotated set μπορεί να ανανεωθεί κάθε στιγμή αν ξαναεκτελέσουμε το εργαλείο σε generation mode, όπου για κάθε αποτέλεσμα μέτρησης κάτω απο κάποιο όριο (ορισμένο απο τον χρήστη), θα μπορούμε να δούμε στην οθόνη τα annotations που δεν συμφωνούνε (και το κείμενο στο οποίο δείχνουν). Η έξοδος του εργαλείου γράφεται σε ένα αρχείο HTML, σε πίνακα, όπως φαίνεται στο σχήμα 3.3.

Οι σύγχρονες εκτιμήσεις για το σύστημα MUSE Language Engineering παράγουν μέσες τιμές 90-95% σε θέματα ακρίβειας και ανάκλησης, σε μια γκάμα διαφορετικών τύπων κειμένου (κείμενο απο αναγνώριση φωνής, email κλπ.) Το προεπιλεγμένο σύστημα στο ANNIE παράγει 80-90% ακρίβεια και ανάκληση σε ειδησεογραφικά κείμενα. Αυτή η τιμή είναι χαμηλότερη στο σύστημα MUSE, επειδή οι πόροι δεν έχουν προσαρμοστεί σε έναν συγκεκριμένο τύπο κειμένου ή annotation, αλλά προσαρμόζονται κατα βούληση ανάλογα με τις εκάστοτε ανάγκες.

ΚΕΦΑΛΑΙΟ 5

GATE DEVELOPER

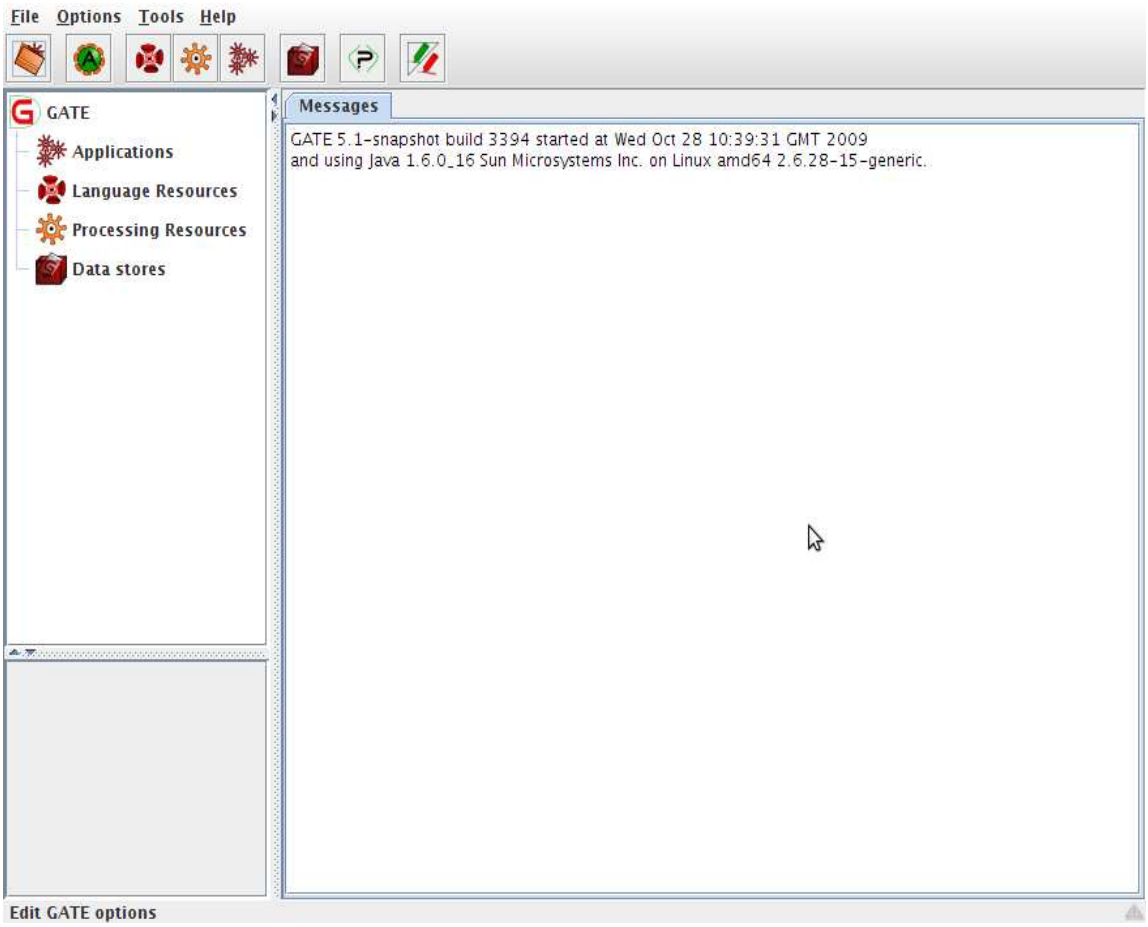
Το GATE Developer είναι το γραφικό περιβάλλον χρήστη (GUI) του GATE. Είναι ανάλογο με συστήματα όπως το mathematica για μαθηματικούς, ή το eclipse για προγραμματιστές Java, αφού παρέχει ένα εύχρηστο γραφικό περιβάλλον για την ανάπτυξη και την έρευνα λογισμικού επεξεργασίας γλώσσας. Εκτός απο αυτόνομη χρήση, μπορεί να συνδυαστεί εύκολα και αποδοτικά και με το GATE Embedded (Το API του GATE με τη χρήση του οποίου μπορούν κάποια κομμάτια του να χρησιμοποιηθούν μέσα σε ανεξάρτητες εφαρμογές). Για παράδειγμα, το GATE Developer μπορεί να χρησιμοποιηθεί για να δημιουργηθούν εφαρμογές οι οποίες έπειτα μπορούν να ενσωματωθούν μέσω του API.

Η βασική λειτουργία του GATE είναι το annotation εγγράφων. Σημαντικά στοιχεία του είναι :

- Τα έγγραφα (documents) τα οποία θα κάνει annotate
- Τα σώματα εγγράφων (corpora), ομάδες εγγράφων συγκεντρωμένες έτσι ώστε να εκτελούνται ομοιόμορφα οι διαδικασίες πάνω τους
- Τα annotations που δημιουργούνται πάνω στα κείμενα
- annotation types όπως όνομα και ημερομηνία.
- annotation sets δηλαδή ομάδες απο annotations
- processing resources τα οποία επεξεργάζονται και δημιουργούν annotations
- applications, που αποτελούνται απο μια σειρά processing resources που εφαρμόζονται σε ένα κείμενο ή ένα corpus.

Το τελικό αποτέλεσμα μιας διαδικασίας τέτοιου τύπου ποικίλει. Στις περισσότερες περιπτώσεις, η έξοδος είναι στη μορφή ενός annotated κειμένου ή ενός corpus.

5.1 ΤΟ MAIN WINDOW ΤΟΥ GATE



Σχήμα 5.1 : Το main window του GATE Developer

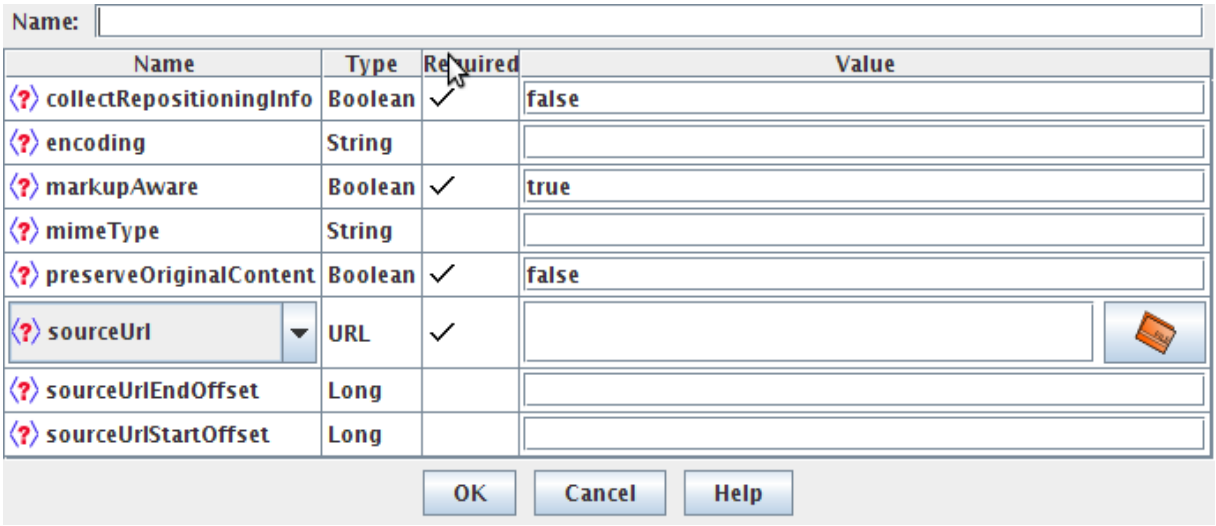
Το πρώτο παράθυρο που βλέπουμε όταν εκτελούμε το GATE είναι το main window. Υπάρχουν 5 περιοχές :

1. Στην κορυφή, τα menu bars και toolbars με τα menu “File”, “Options”, “Tools” και “Help” και τα εικονίδια για τις σημαντικότερες εντολές.
2. Στην αριστερή πλευρά, ένα δέντρο (resource tree) που ξεκινάτει απο το “GATE” και περιέχει τους πόρους του προγράμματος.
3. Στην κάτω αριστερή πλευρά ένα ορθογώνιο (small resource viewer)
4. Στο κέντρο είναι ο main resource viewer, που περιέχει tabs με τα ανοιχτά resources και το tab “Messages”.
5. Στην κάτω μεριά της οθόνης, ένα messages bar.

Το resource tree και το resource viewer συνεργάζονται έτσι ώστε να παρέχουν στο πρόγραμμα τη δυνατότητα οπτικοποίησης των resources με διάφορους τρόπους.

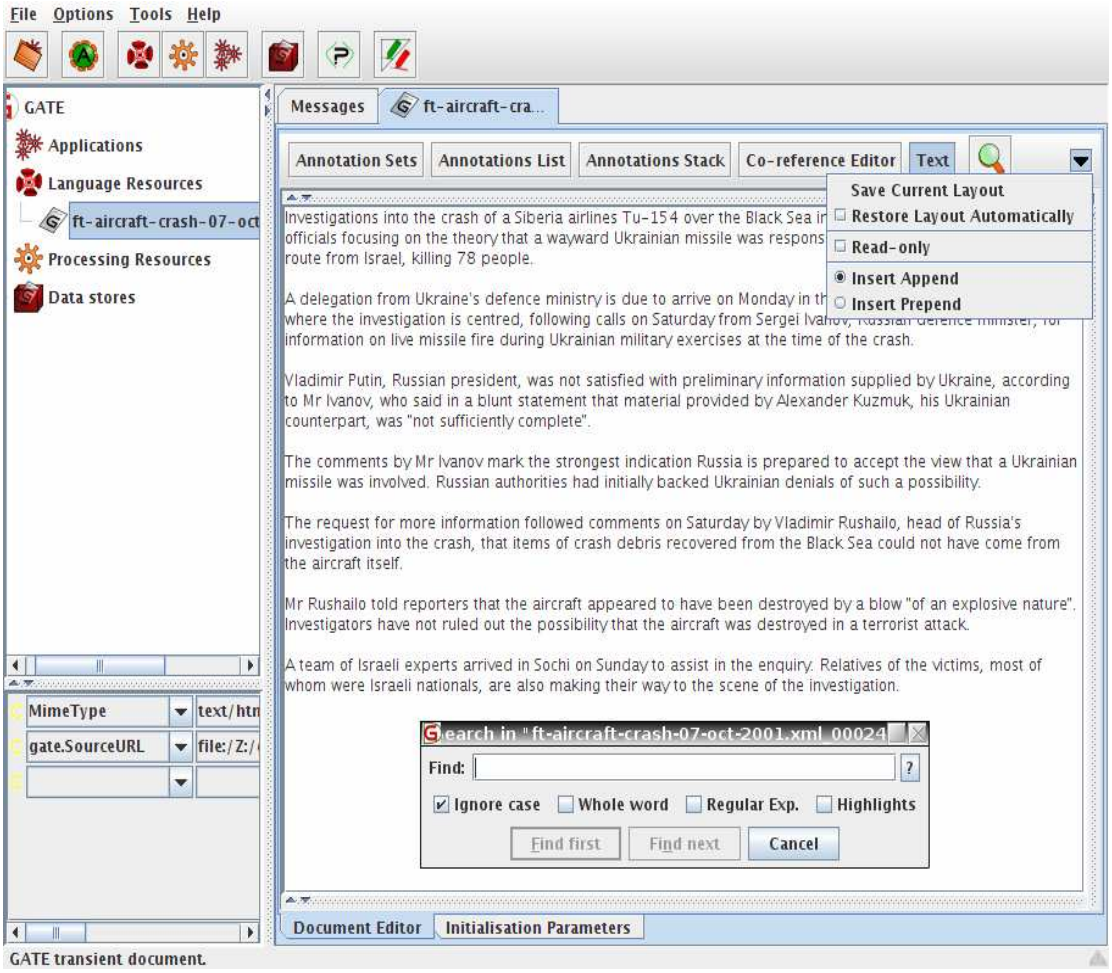
Διαλέγοντας tab, μπορούν να εμφανιστούν διαφορετικές πληροφορίες στο main viewer. Αν υπάρχει κάποιο σφάλμα, το messages tab θα αλλάξει χρώμα σε κόκκινο, και σε κάποιες περιπτώσεις θα υπάρξει και pop-up error message. Επίσης υπάρχει η δυνατότητα απο το menu Options να επιλέξουμε αν θέλουμε να συνδέσουμε την επιλογή στο resource tree με το τί εμφανίζεται στο main view.

5.2 ΦΟΡΤΩΣΗ ΕΓΓΡΑΦΩΝ



Σχήμα 5.2 : Δημιουργώντας ένα καινούργιο document

Με δεξί κλίκ στο «language resources» sto resource tree και επιλογή New > GATE Document , ανοίγει το παράθυρο “Parameters for the new GATE Document”, όπως φαίνεται στο σχήμα 5.2 . Στο παράθυρο αυτό μπορούμε να παραμετροποιήσουμε το έγγραφο GATE που θέλουμε να δημιουργήσουμε. Τα υποχρεωτικά πεδία σημειώνονται με tick. Αν δεν συμπληρωθεί το όνομα του εγγράφου, θα δωθεί ένα προεπιλεγμένο. Στο sourceURL συμπληρώνουμε το URL της πηγής του εγγράφου. Για παράδειγμα μπορούμε να δώσουμε ένα URL ιστοσελίδας ή ένα path ενός αρχείου κειμένου ή XML αποθηκευμένο στον δίσκο.

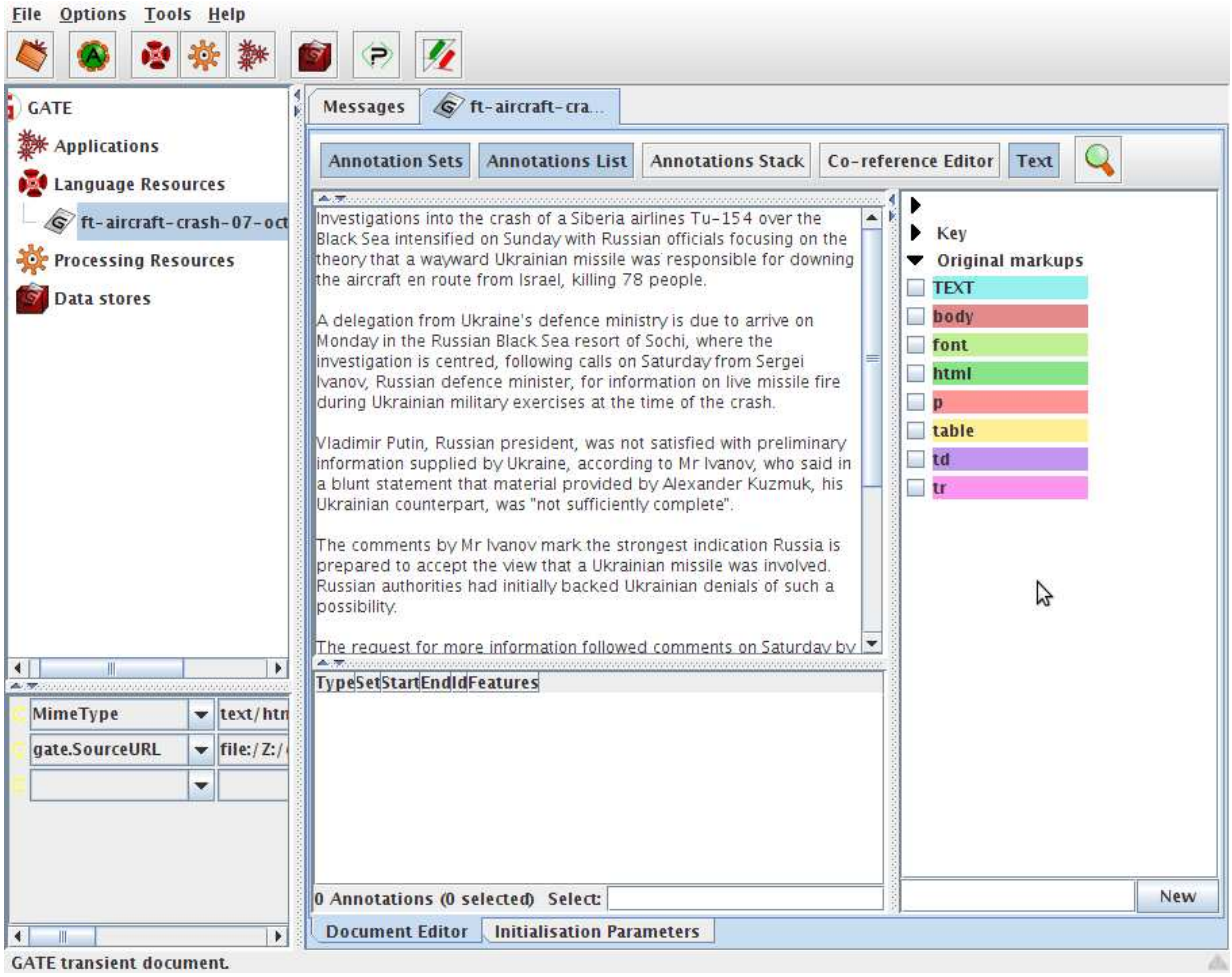


Σχήμα 5.3 : Ο επεξεργαστής εγγράφου

Ο επεξεργαστής εγγάφου βρίσκεται στο κεντρικό παράθυρο του GATE Developer, και εμφανίζεται με διπλό κλικ στο έγγραφο στο resource tree . Αποτελείται από το panel στην κορυφή, το οποίο διαχειρίζεται κάποιες διαφορετικές όψεις και το search box. Αρχικά, ο χρήστης βλέπει μόνο το κείμενο του εγγράφου, όπως φαίνεται στο σχήμα 5.3. Αν ενεργοποιήσουμε τις επιλογές «Annotation Sets» και «Annotations List», μπορούμε να δούμε στα δεξιά μας τα annotation sets και στο κάτω μέρος του editor την λίστα των annotations, παρομοίως με το σχήμα 5.4. Στο κάτω μέρος αντί για την λίστα θα μπορούσε να υπάρχει και το Annotation Stack, και στα δεξιά αντί για τα sets, ο co-reference editor. Επίσης υπάρχει η δυνατότητα save και load layout και κάποιες άλλες επιλογές στο τρίγωνο πάνω δεξιά.

Όταν φορτώνεται ένα έγγραφο, υπάρχει η δυνατότητα επεξεργασίας του κειμένου του. Παρέχονται οι κλασσικές εντολές copy, paste, cut και τα keyboard

shortcuts ανάλογα με το λειτοθργικό σύστημα. Το εικονίδιο του μεγεθυντικού φακού ενεργοποιεί την λειτουργία αναζήτησης. Για να μην εμφανίζονται συνέχεια παράθυρα με πληροφορίες απο annotations κάθε φορά που επιλέγεται κείμενο, ο χρήστις πρέπει να έχει πατημένο και το CTRL. Εναλλακτικά, αν κλείσει το annotation sets view, το έγγραφο χάνει τα highlights του.



Σχήμα 5.4 : Ο document editor με annotation sets και annotation list.

5.3 ΔΗΜΙΟΥΡΓΙΑ ΚΑΙ ΕΠΕΞΕΡΓΑΣΙΑ CORPORA

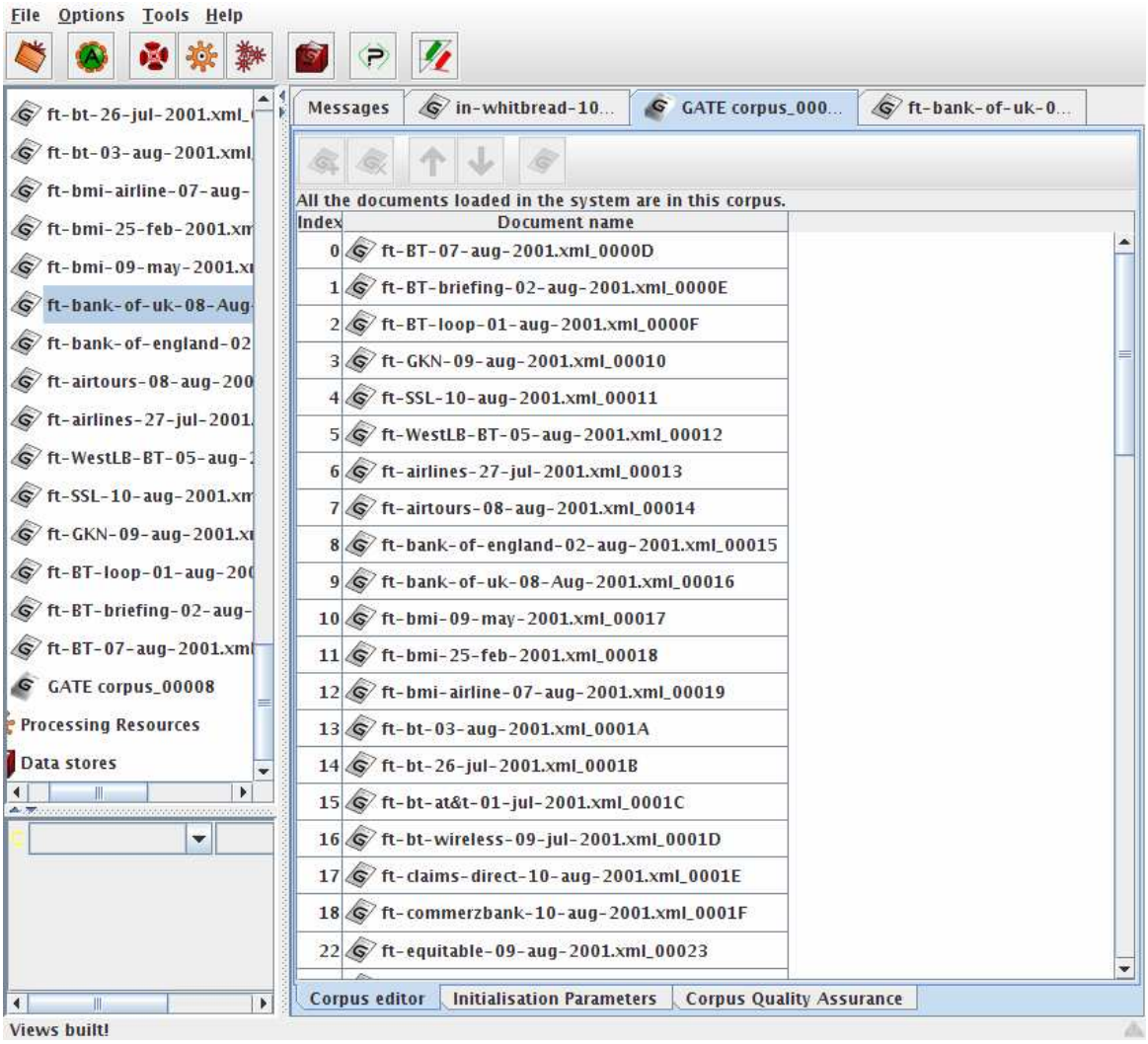
Παρομοίως με την δημιουργία εγγράφων, υπάρχει και η δυνατότητα δημιουργίας corpora. Με δεξί κλικ στο Language resources > New > GATE Corpus, εμφανίζεται ένα πλαίσιο διαλόγου, στο οποίο ο χρήστης μπορεί να δώσει ένα όνομα για το corpus (αν αφεθεί κενό, θα δωθεί ένα default όνομα). Επίσης μπορεί να προσθέσει έγγραφα στο corpus, τα οποία έγγραφα όμως θα πρέπει ήδη να είναι φορτωμένα στο GATE.

Με διπλό κλικ στο corpus στο resource view, εμφανίζεται ο επεξεργαστής του (corpus editor) όπως βλέπουμε στο σχήμα 4.5. Στο κεντρικό παράθυρο υπάρχει ένας κατάλογος των εγγράφων που είναι μέσα στο corpus.

Στο πάνω αριστερά μέρος του corpus editor, υπάρχουν κουμπιά με τα σύμβολα σύν και πλήν. Αυτά επιτρέπουν στον χρήστη να προσθέσει ή να αφαιρέσει έγγραφα από το corpus.

Τα κουμπία που δείχνουν πάνω και κάτω, επιτρέπουν την επαναδιοργάνωση των εγγράφων στο corpus. Το κουμπί στα δεξιά, ανοίγει το έγγραφο στον document editor.

Στο κάτω μέρος, εκτός απο το corpus editor tab που απεικονίζεται, υπάρχουν τα tabs «Initialization Parameters» και «Corpus Quality Assurance». Με κλικ στο πρώτο, βλέπουμε τους παραμέτρους της αρχικοποίησης του συγκεκριμένου corpus. Στο δεύτερο βλέπουμε κάποιες μετρήσεις που υπολογίζονται (Agreement measures) σχετικά με το corpus.



Σχήμα 5.5 : Ο document editor με annotation sets και annotation list.

5.4 ΕΠΕΞΕΡΓΑΣΙΑ ANNOTATIONS

Εκτός απο την αυτόματη δημιουργία annotations η οποία πετυχαίνεται με την χρήση processing resources στο GATE, υπάρχει και η δυνατότητα χειροκίνητης δημιουργίας και επεξεργασίας τους, είτε τελείως χειροκίνητα απο τον χρήστη είτε ημιαυτόματα, τρέχοντας μια εφαρμογή πάνω απο το corpus και έπειτα διορθώνοντας τα annotations και προσθέτοντας καινούργια χειροκίνητα.

5.4.1 ΤΟ ANNOTATION SETS VIEW

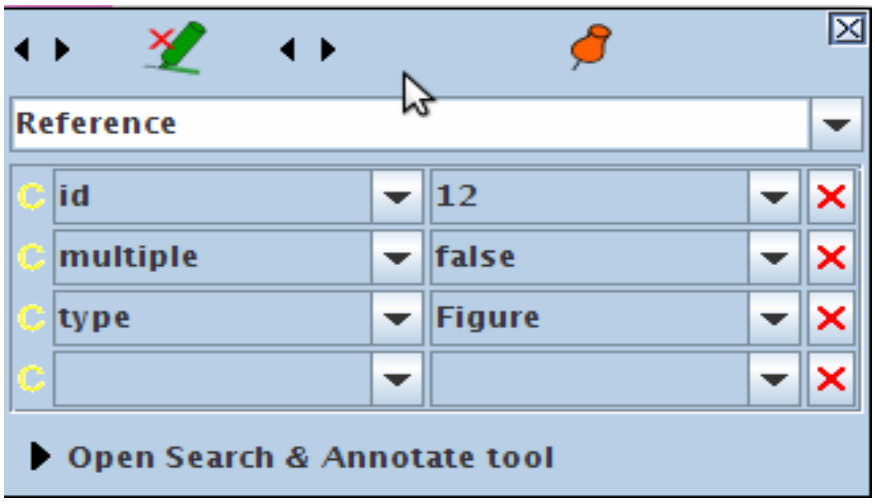
Τα annotation sets εμφανίζονται με δεξί κλικ στο κουμπί “annotation Sets” στην κορυφή του επεξεργαστή εγγράφου, ή με το κουμπί F3. Θα εμφανιστεί ο annotation

sets viewer, μέσω του οποίου εμφανίζονται τα διαθέσιμα annotation sets και οι αντίστοιχοι τύποι annotation.

Το annotation sets view εμφανίζεται στο αριστερό μέρος του επεξεργαστή εγγράφου. Η μορφή του είναι δενδροειδής με μια ρίζα για κάθε annotation set. Το πρώτο annotation set στη λίστα είναι πάντα ανώνυμο (το default annotation set). Στο σχήμα 5.4 φαίνεται σαν ένα drop-down arrow χωρίς όνομα δίπλα του. Άλλα annotation Sets που βλέπουμε στο 5.4 είναι το “Key” και το “Original Markups”. Επειδή το πρωτότυπο κείμενο είναι της μορφής XML, η αυθεντική περιγραφή (markup) XML του εγγράφου διατηρείται με μορφή annotation set, το οποίο έχει επεκταθεί και περιέχει επίσης annotations τύπου “TEXT”, “body”, “font”, “html”, “p“, “table”, “td” kai “tr”.

Με το τσεκάρισμα του checkbox δίπλα στο annotation, εμφανίζονται χρωματισμένα (highlighted) στο main window όλα τα κομμάτια του κειμένου στα οποία εμφανίζεται το συγκεκριμένο annotation. Υπάρχει επίσης δυνατότητα διαγραφής ενός τύπου annotation με το delete ή αλλαγή χρώματος με το Enter. Όλες αυτές οι διεργασίες εμφανίζονται και σε ένα context menu me δεξί κλικ σε έναν τύπο annotation ή ένα annotation set.

Αν υπάρχουν annotations στο έγγραφο, με δεξί κλικ ή απλα με hover του δείκτη του mouse πάνω σε ένα annotation, εμφανίζεται η λίστα των annotations τα οποία σχετίζονται με το συγκεκριμένο κομμάτι κειμένου, απο τα οποία επιλέγοντας ένα εμφανίζεται ο annotation editor (Σχήμα 5.6). Αν υπάρχει μόνο ένα, τότε ο editor εμφανίζεται απευθείας.



Σχήμα 5.6 : Ο Annotation Editor.

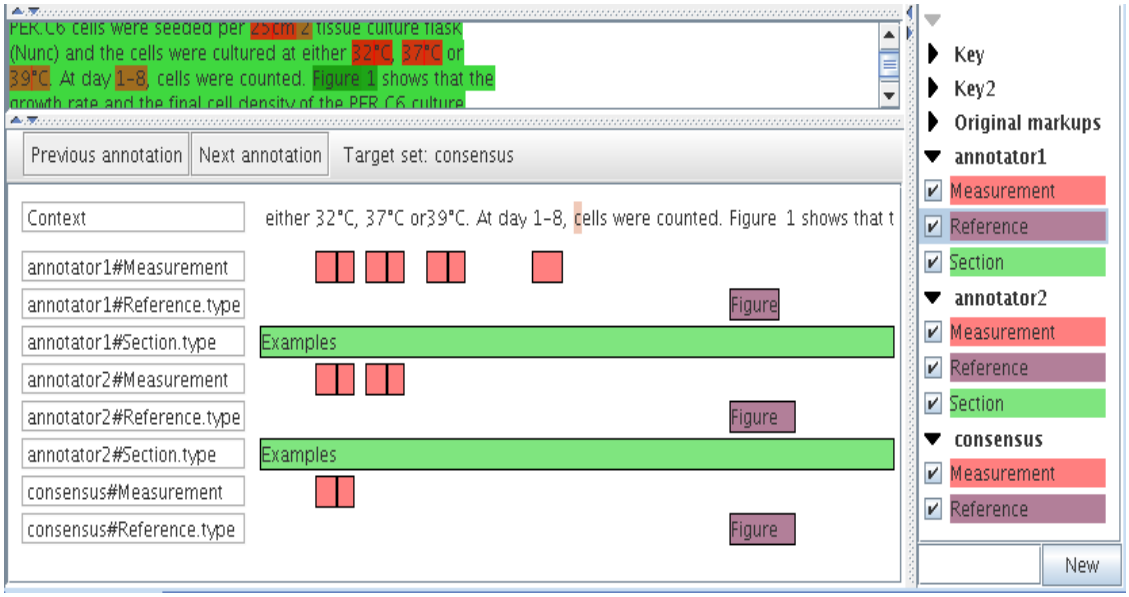
5.4.2 ΤΟ ANNOTATION LIST VIEW

Το annotation list view εμφανίζεται κάτω απο το main text με κλικ στο κουμπι “annotation list” ή πατώντας το F4. Περιέχει τα annotations που επιλέχθηκαν απο τα Annotation Sets. Υπάρχει δυνατότητα αλφαβητικής ταξινόμησης για κάθε στήλη, με κλικ στο όνομα της στήλης. Με δεξί κλικ στο όνομα, η στήλη κρύβεται.

5.4.3 ΤΟ ANNOTATION STACK VIEW

Το view αυτό δείχνει τα annotations μαζί με κάποιο context πρίν και μετά το επιλεγμένο γράμμα του κέρσορα. Έτσι, ο χρήστης μπορεί να την ελέγξει απλά με κίνηση του κέρσορα. Με τα πλήκτρα δεξί και αριστερό βέλος, η όψη μετακινείται ανα γράμμα και με ctrl+δεξιά ή αριστερά μετακινείται ανα λέξη. Με το πάνω και κάτω αλλάζει η γραμμή.

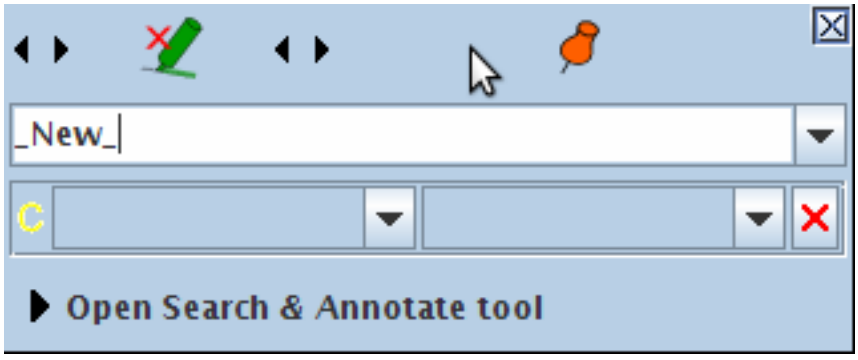
Οι τύποι annotation που φαίνονται είναι ανάλογοι με τις επιλογές που έχουν γίνει στο annotation sets view. Με δεξί κλικ σε οποιοδήποτε annotation στο annotation stack υπάρχει η επιλογή για επεξεργασία του annotation.



Σχήμα 5.7 Το annotation Stack View

5.4.4 ΔΗΜΙΟΥΡΓΙΑ ANNOTATIONS

Για χειροκίνητη δημιουργία annotations, ο χρήστης επιλέγει το κείμενο το οποίο θέλει να γίνει annotated, και αφήνει (hover) το ποντίκι πάνω του ή χρησιμοποιεί το συνδυασμό ctrl+E. Θα εμφανιστεί ένα παράθυρο popup, με τη βοήθεια του οποίου δημιουργείται ένα καινούργιο annotation, όπως φαίνεται και στην εικόνα 5.8 :



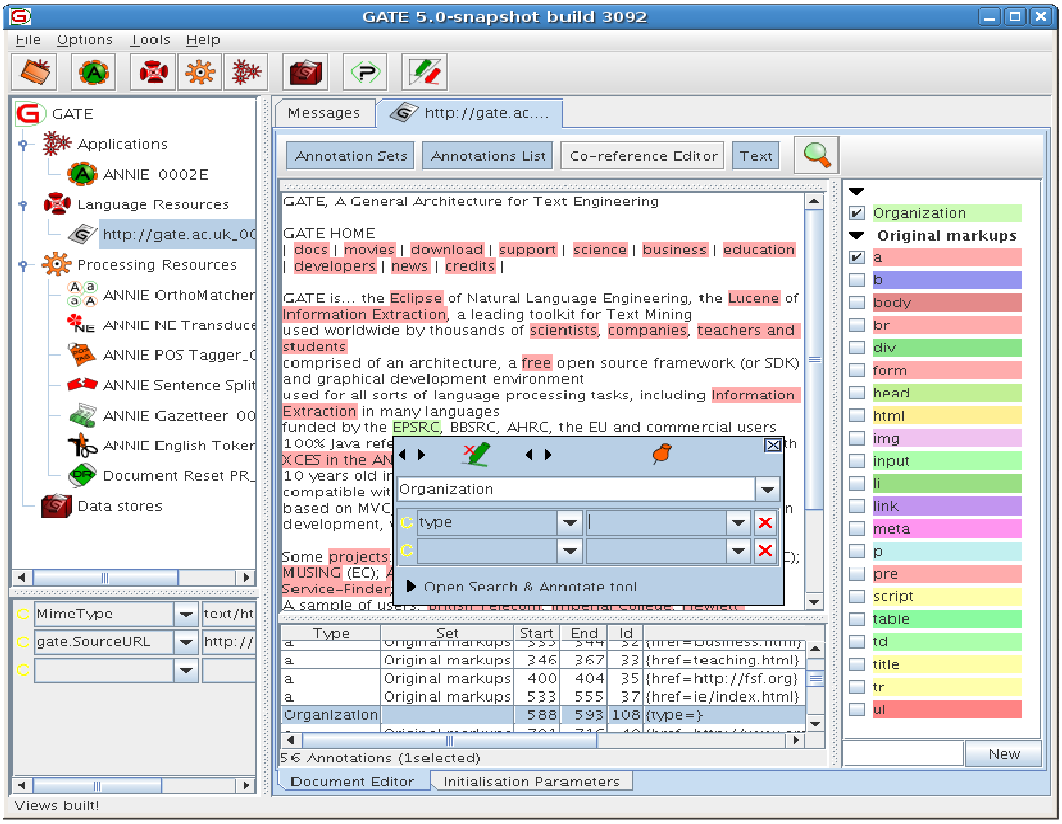
Σχήμα 5.8 : Δημιουργόντας ένα καινούργιο annotation

Ο προεπιλεγμένος τύπος του annotation, θα είναι ο ίδιος με τον αντίστοιχο του τελευταίου annotation που έχει δημιουργηθεί, εκτός αν δεν ισχύει αυτή η περίπτωση, οπότε θα είναι τύπου “_New_”. Μπορεί να χρησιμοποιηθεί οποιοδήποτε όνομα annotation στο text box. Από τα drop down menus κάτω απο το όνομα προστίθενται και αφαιρούνται χαρακτηριστικά (features) και οι αντίστοιχες τιμές τους.

Η διαγραφή ενός annotation επιτυγχάνεται πατώντας το κόκκινο κουμπί X στην κορυφή του popup window. Επίσης το εικονίδιο της πινέζας κάνει pin το παράθυρο με σκοπό να παραμείνει εκεί που είναι. Το pinning πρακτικά σημαίνει ότι ακόμα κι αν ο χρήστης επιλέξει ένα άλλο annotation, το παράθυρο θα μείνει στο ίδιο μέρος.

Αν δημιουργηθεί καινούργιο annotation, θα τοποθετηθεί στο annotation set το οποίο έχει επιλεχθεί (highlighted) απο τον χρήστη. Για να δημιουργηθεί ένα καινούργιο annotation set, απλά ο χρήστης πρέπει να γράψει το όνομα του στο κουτί κάτω από τη λίστα των annotation sets και να κάνει κλικ στο “New”.

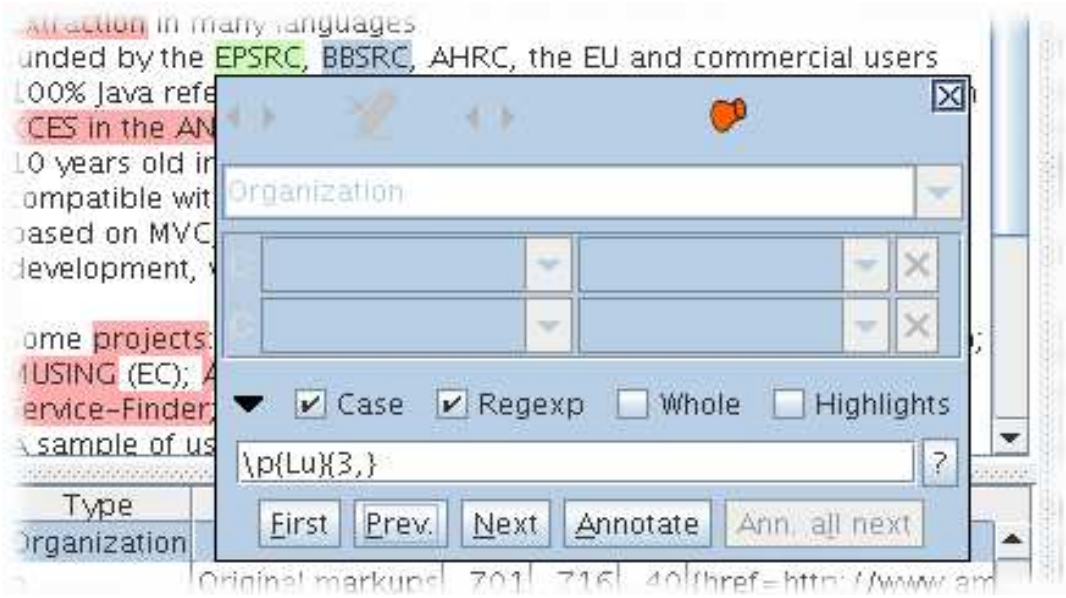
Στην εικόνα 5.9, φαίνεται η διαδικασία δημιουργίας καινούργιου annotation με όνομα “Organization” για το string “EPSRC” το οποίο annotation ανήκει στο default annotation set (φαίνεται με κενό όνομα στο annotation set view δεξιά) και ένός feature με όνομα “type” και χωρίς ακόμα τιμή.



Σχήμα 5.9 : Προσθέτωντας το annotation “Orgaizaton” στο Default Annotation Set

Για αυτόματη αναζήτηση και annotation του κειμένου, υπάρχει η επιλογή “search and annotate”. Όπως βλέπουμε στο σχήμα 5.10 ο χρήστης :

- Δημιουργεί ή επιλέγει ένα annotation το οποίο θα χρησιμοποιηθεί σαν μοντέλο.
- Ανοίγει το panel στο κάτω μέρος του παραθύρου annotation editor.
- Αλλάζει την πρόταση την οποία αναζητούμε αν είναι αναγκαίο.
- Με το κουμπί “First” ή το Enter διαλέγει την πρώτη πρόταση που θέλει να κάνει annotate.
- Αν η επιλογή είναι σωστή, το κουμπί “Annotate” κάνει το annotation. Αλλιώς με το κουμπι “Next”, το πόγραμμα περνάει στο επίμενο. Μετά απο κάποια “Annotate” και “Search”, για τα υπόλοιπα μπορεί να χρησιμοποιήσει το κουμπί “Ann. all next”



Σχήμα 5.10 : Η διαδικασία “Search and Annotate”

5.5 ΧΡΗΣΗ CREOLE PLUGINS

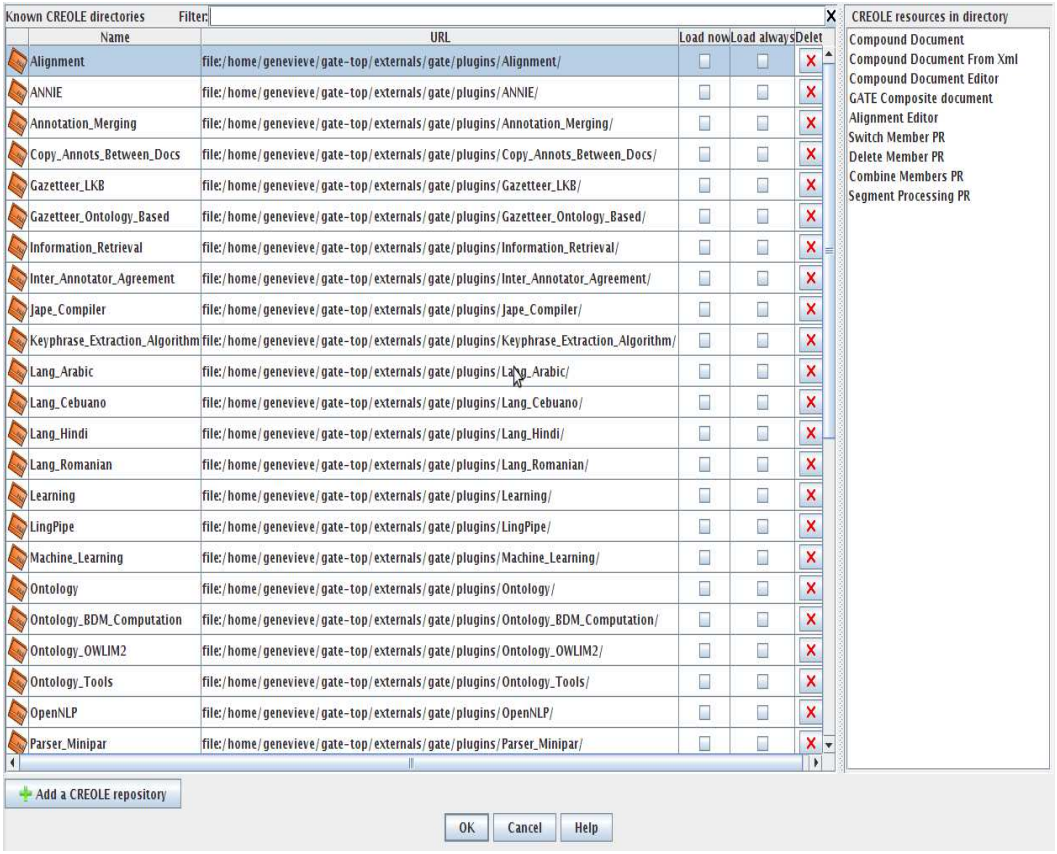
Στο GATE, τα processing resources χρησιμοποιούνται για την αυτόματη δημιουργία και επεξεργασία των annotations σε έγγραφα. Στην επεξεργασία των processing resources θα αναφερθεί ο τομέας 5.6. Προαπαιτούμενο για αυτό όμως είναι η αναφορά στα CREOLE Plugins. Ο λόγος είναι ότι για να χρησιμοποιηθεί ένα οποιοδήποτε processing resource (όπως και αρκετά language resources) θα πρέπει να φορτωθεί πρώτα το CREOLE Plugin το οποίο το περιέχει.

Τα CREOLE resources περιέχονται σε CREOLE directories (φάκελοι οι οποίοι περιγράφουν τους πόρους, το java archive που περιέχει τον compiled executable κώδικα, και οι βιβλιοθήκες που χρειάζονται από τα resources.

Από την έκδοση 3 του GATE και μετά, τα CREOLE directories ονομάζονται “CREOLE Plugins” ή απλά “Plugins”. Σε προηγούμενες εκδόσεις, τα CREOLE Resources που περιέχει το GATE υπήρχαν μέσα σε ένα μονολιθικό αρχείο με όνομα gate.jar. Το γεγονός ότι η έκδοση 3 έχει κάθε plugin σε ξεχωριστό φάκελο επιτρέπει την εύκολη πρόσβαση στα αρχεία χωρίς να χρειάζεται ο χρήστης να κάνει unpack το αρχείο gate.jar.

Ο χρήστης μπορεί να διαχειριστεί τα CREOLE plugins μέσω του GUI τους, το οποίο ενεργοποιείται με την επιλογή File > Manage Creole Plugins. Για κάθε plugin υπάρχουν δυο επιλογές με τη μορφή check boxes. Η μία (“Load now”) φορτώνει το

plugin στη μνήμη, και η άλλη (“Load Always”) βάζει το plugin σε λίστα απο εγκατεστημένα auto-loadable plugins. Επίσης υπάρχει και ένα κουμπί “Delete” το οποίο διαγράφει το plugin απο τη λίστα των γνωστών plugins, αλλα δεν διαγράφει και το directory του. Τα εγκατεστημένα plugins φορτώνονται αυτόματα κάθε φορά που ανοίγει το GATE. Αν ένα εγκατεστημένο plugin σβηστεί απο τη λίστα, θα ξαναεμφανιστεί την επόμενη φορά που θα ξεκινήσει το GATE.



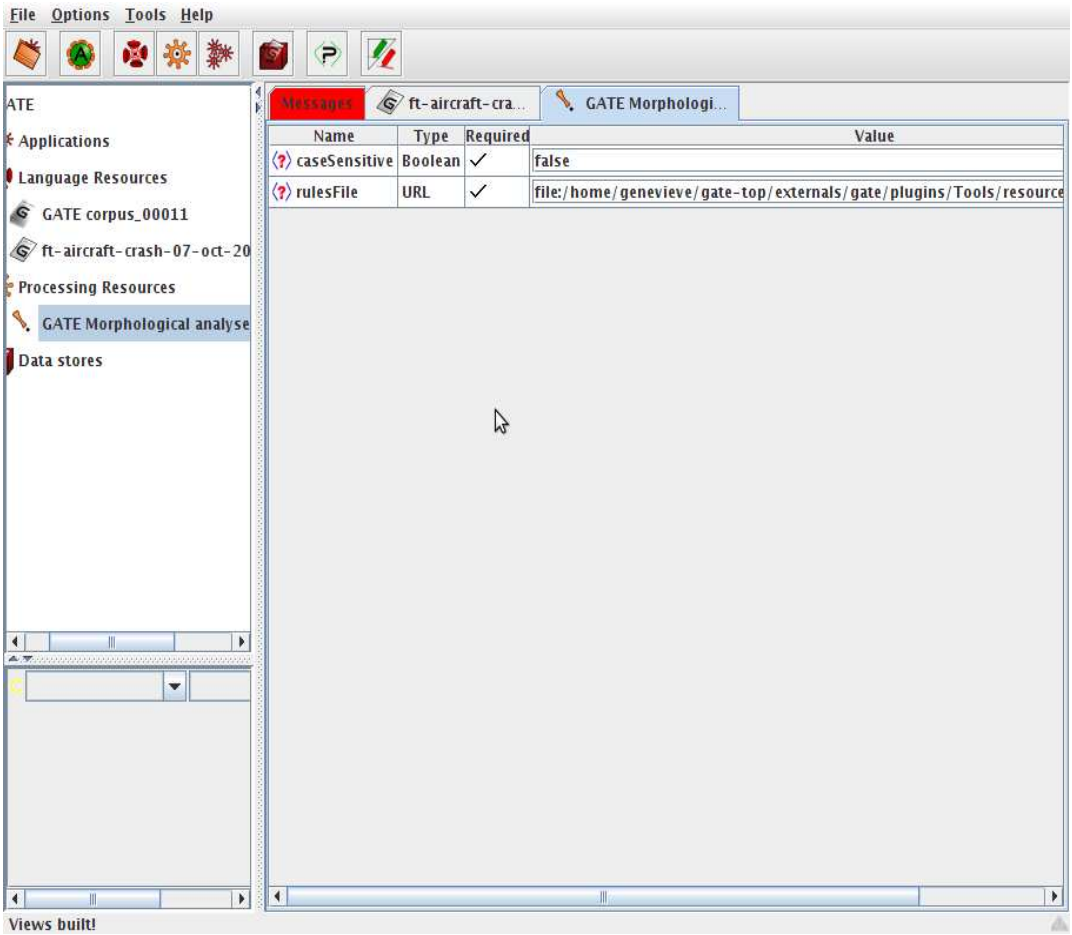
Σχήμα 5.11 : Κονσόλα επεξεργασίας plugins

Από τη στιγμή που τα plugins φορτωθούν, οι πόροι που περιέχουν θα είναι έτοιμοι για χρήση. Τις περισσότερες φορές, αυτό σημαίνει ότι θα εμφανίζονται στο μενού “New” όταν ο χρήστης κάνει δεξί κλικ στο “Processing Resources”, στο resource pane. Οστώσο, κάποια συγκεκριμένα plugins έχουν διαφορετικές λειτουργίες (Για παράδειγμα ο Schema_Annotation_Editor).

5.6 ΧΡΗΣΗ PROCCESSING RESOURCES

Το πρώτο βήμα για την χρήση των processing resources είναι η φόρτωση των αντίστοιχων plugins. Αν το resource που απαιτείται δεν υπάρχει στη λίστα των processing resources, τότε πιθανόν να μην είναι φορτωμένο το plugin που απαιτείται. Τα resources φορτώνονται με απο την λίστα με τον τρόπο που προαναφέρθηκε στο προηγούμενο κεφάλαιο.

Για παράδειγμα, έστω οτι είναι φορτωμένο το Plugin “Tools”. Με δεξι κλικ στο Processing resources > New, υπάρχει η δυνατότητα να δημιουργηθούν όλα τα processing resources που παρέχει το Plugin. Αν επιλέξει ο χρήστης να δημιουργήσει το “GATE Morphological Analyser” με τις default παραμέτρους, ένα instance του εμφανίζεται κάτω απο το “Proccessing resources”, έτοιμο για χρήση. Με διπλό κλίκ πάνω του εμφανίζονται οι παράμετροί του (σχήμα 5.12).



Σχήμα 5.12 : Παράμετροι του GATE Morphological Analyser

5.7 ΔΗΜΙΟΥΡΓΙΑ ΚΑΙ ΕΚΤΕΛΕΣΗ ΕΦΑΡΜΟΓΩΝ

Όταν έχουν φορτωθεί όλα τα resources, μπορεί να δημιουργηθεί μια καινούργια εφαρμογή για να εκτελεστεί στο corpus της επιλογής του χρήστη. Αυτό γίνεται με δεκί κλικ στο “Applications” > “New” και μετά υπάρχουν οι επιλογές “Corpus pipeline” και “Pipeline”. Ένα pipeline μπορεί να εκτελεστεί μόνο πάνω σε ένα έγγραφο, ενώ ένα corpus pipeline σε ένα ολόκληρο corpus.

Το επόμενο βήμα είναι ο χρήστης να κάνει διπλο κλικ στο pipeline και να επιλέξει τους πόρους που χρειάζονται για την εκτέλεση της εφαρμογής, και να μεταφέρει τα απαιτούμενα components απο την αριστερή πλευρά του main window στη δεξιά, με διπλο κλικ πάνω τους. Επίσης σημαντική είναι και η σειρά με την οποία θα τοποθετηθούν τα components, η οποία αλλάζει με το πάνω και κάτω βέλος.

Τέλος, εφ όσον οι παράμετροι του κάθε resource είναι οι σωστοί, και έχει επιλεχθεί το σωστό corpus στην περίπτωση του corpus pipeline, η εφαρμογή εκτελείται με το κουμπί “Run”.

ΚΕΦΑΛΑΙΟ 6

RSSGATE

Κατα τη διάρκεια της έρευνας σχετικά με τις δυνατότητες του GATE η οποία έγινε στα πλαίσια της παρούσας εργασίας, παρουσιάστηκε η ανάγκη να προστεθούν λειτουργίες του GATE σε εξωτερικές εφαρμογές. Η αιτία είναι κάποιες συγκεκριμένες αδυναμίες τις οποίες έχει το GATE Developer και το καθιστάν μη αποδοτικό σε αρκετές περιπτώσεις.

Τα δυο κυριότερα απο αυτά τα προβλήματα είναι τα εξής :

- Το corpus το οποίο αναλύεται θα πρέπει να δημιουργηθεί χειροκίνητα, χωρίς καμια δυνατότητα αυτοματοποίησης, και πρέπει οι πηγές απο τις οποίες θα δημιουργούνται τα έγγραφα να είναι στατικά αρχεία κειμένου. Έτσι, δεν υπάρχει η δυνατότητα annotation εγγράφων απο κάποια δυναμική πηγή, ή από κάποιο stream οποιουδήποτε είδους.
- Δεν υπάρχει δυνατήτητα αυτόματης επεξεργασίας των αποτελεσμάτων κάθε διαδικασίας ανάλυσης ενός corpus.

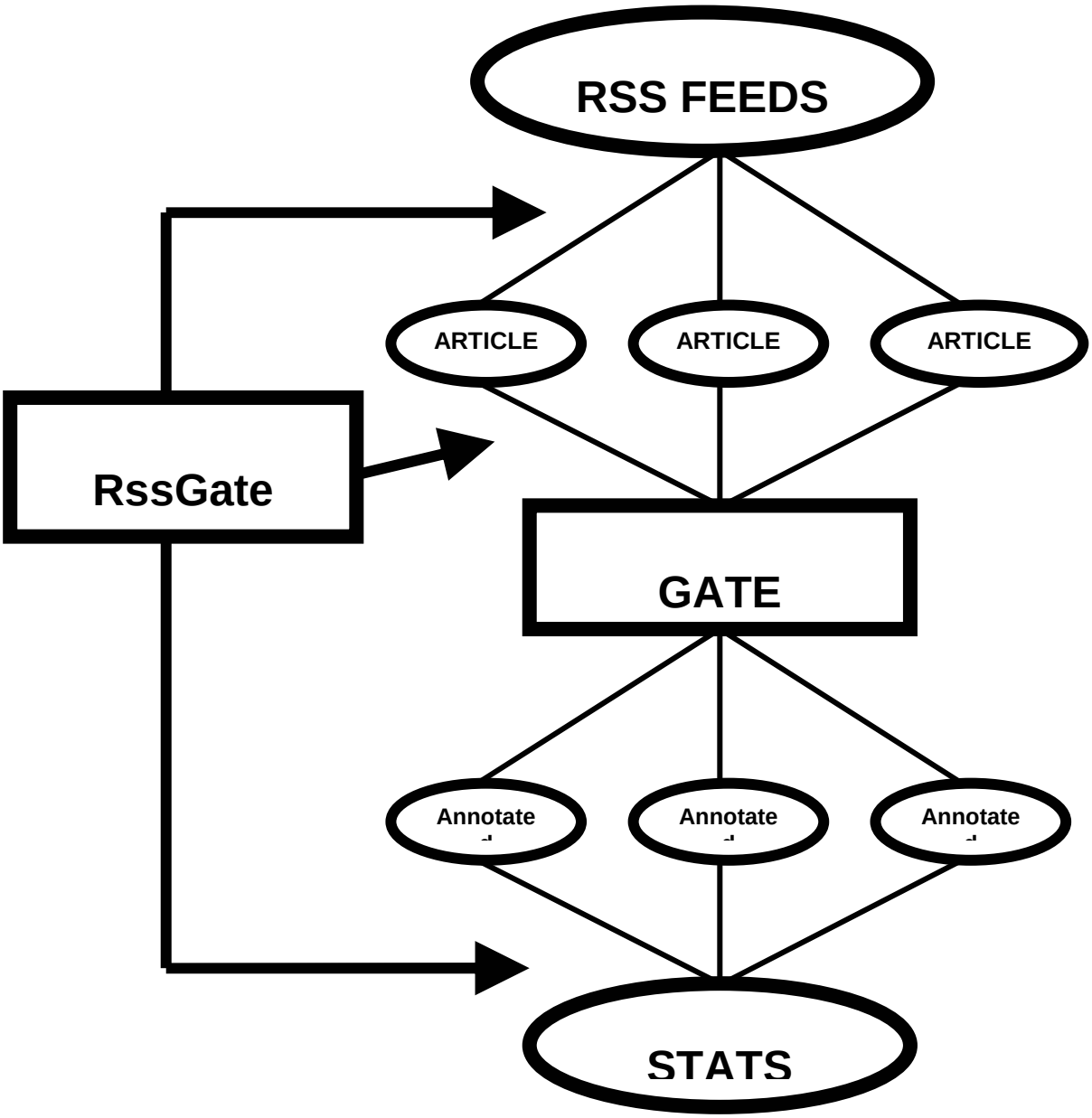
Έτσι, χρειάστηκε να δημιουργηθεί ένα πρόγραμμα, το οποίο να χρησιμοποιεί το GATE Embedded για να καλύψει συγκεκριμένες ανάγκες. Αν η διαδικασία του text engineering θεωρηθεί σαν σύστημα, αυτό ουσιαστικά που στοχεύει να παρέχει το πρόγραμμα είναι η αυτοματοποίηση και η ευκολία στην είσοδο και στην έξοδο.

6.1 ΔΟΜΗ ΤΟΥ RSSGATE

Όπως φαίνεται και απο το σχήμα 6.1, τον ρόλο της επεξεργασίας στο σύστημα μας τον παίζει το GATE, και συγκεκριμένα τα default processing resources του ANNIE, τα οποία είναι διαμορφωμένα σε ένα pipeline του οποίου τελικό αποτέλεσμα είναι το πλήρες annotation του κάθε εγγράφου.

Η είσοδος είναι άρθρα τα οποία το RSSGATE αυτόματα κάνει download από τα feeds που έχει επιλέξει ο χρήστης. Φυσικά, το RSSGATE παρέχει και την δυνατότητα ανάγνωσης των feeds, οπότε μπορεί να χρησιμοποιθεί και σαν ένας αυτόνομος RSS Reader.

Η έξοδος είναι τα ποσοτικά δεδομένα τα οποία προκύπτουν από τις αναλύσεις των εγγράφων σε βάθος χρόνου. Επίσης, υπάρχει η δυνατήτητα εμφάνισης των annotated άρθρων με παρόμοιο τρόπο με το GATE Developer.



Σχήμα 6.1 : Η διαδικασία του RssGate

6.2 ΔΙΑΔΙΚΑΣΙΑ ΧΡΗΣΗΣ ΤΟΥ RSSGATE

Βλέποντας το σχήμα 6.1, μπορούμε να συμπεράνουμε ότι η τυπική διαδικασία περίπτωσης χρήσης του RSSGATE περνάει από τα εξής στάδια :

1. Καταχώρηση των URL των RSS Feeds που χρειάζονται από τον χρήστη. Επιλογή των τύπων annotations που θέλει να χρησιμοποιηθούν.
2. Αυτόματο download όλων των άρθρων που προσφέρει το feed, μετατροπή τους σε plain text language resources έτοιμα για χρήση από τα processing resources του ANNIE (αποκοπή των HTML tags και των εικόνων) και πέρασμά τους από τα processing resources του ANNIE με σκοπό την πλήρη ανάλυση του κειμένου. Έπειτα, δημιουργείται ένα περιβάλλον παρόμοιο με το GATE Developer στο οποίο κάθε annotation που υπάρχει στα articles είναι χρωματισμένο, με διαφορετικό χρωματισμό ανάλογα με τον τύπο του και δυνατότητα αναζήτησης λέξεων, instances των annotations ή και συνδυασμό μεταξύ τους.
3. Δυνατότητα αναζήτησης μακράς διάρκειας. Αυτή η διαδικασία είναι παρόμοια με την αναζήτηση annotations στο προηγούμενο βήμα, με τη διαφορά ότι ο χρήστης παρέχει μια φορά τους κανόνες της και γίνεται αυτόματα σε τακτά χρονικά διαστήματα, με σκοπό την συγκομιδή μακροπρόθεσμων αποτελεσμάτων.
4. Παρουσίαση των μακροπρόθεσμων αποτελεσμάτων της αναζήτησης σε μορφή πίνακα ή γραφικής παράστασης.

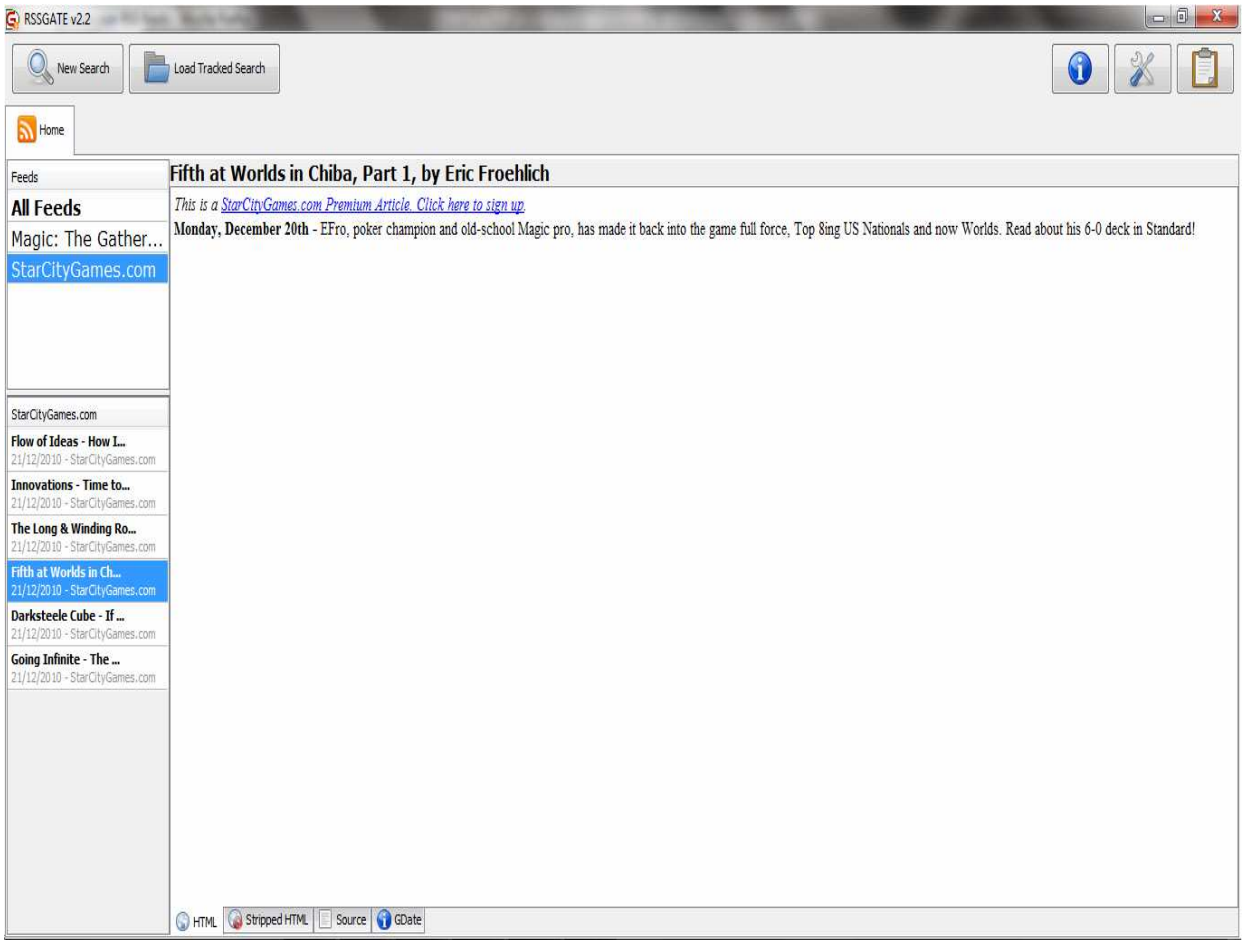
6.3 ΤΟ ΓΡΑΦΙΚΟ ΠΕΡΙΒΑΛΛΟΝ ΤΟΥ RSSGATE

Το graphical user interface του RSSGATE γράφτηκε για Windows, και η γλώσσα που χρησιμοποιήθηκε είναι η java, με τη βοήθεια των libraries του swing.

6.3.1 ΤΟ MAIN WINDOW ΤΟΥ RSSGATE

Το main window (Σχήμα 6.2) είναι το πρώτο παράθυρο που εμφανίζεται από τη στιγμή της εκτέλεσης του RSSGATE. Στην κορυφή του παραθύρου υπάρχουν τα κουμπιά “New Search” το οποίο ανοίγει ένα search tab, για την πραγματοποίηση

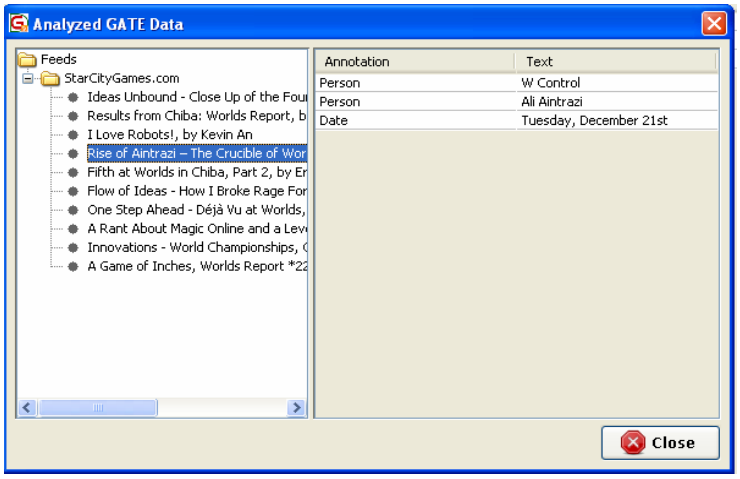
αναζήτησης σε όλα τα articles των feeds που είναι φορτωμένα στο πρόγραμμα. Η λειτουργία του Search θα αναλυθεί περισσότερο στο κεφάλαιο 6.3.3



Σχήμα 6.2 : Το main window του RSSGATE

Δεξιά του υπάρχει το πλήκτρο “Load Tracked Search”, το οποίο ανοίγει ένα μενού (Σχήμα 6.3) με τα tracked searches (μακροπρόθεσμες αναζητήσεις) που έχει αποθηκεύσει ο χρήστης. Με την επιλογή του και το πλήκτρο load, εμφανίζεται το παράθυρο στατιστικών στοιχείων της αποθηκευμένης αναζήτησης. Επίσης με το πλήκτρο delete υπάρχει η δυνατότητα διαγραφής κάποιου tracked search. Δεξιά από αυτά υπάρχουν τα πλήκτρα “View Analyzed GATE data”, “Configure” και “GATE Log”, από αριστερά προς τα δεξιά, τα οποία πηγαίνουν τον χρήστη σε δευτερεύοντα βοηθητικά μενού για τον χρήστη.

Το “View Analyzed GATE Data” ανοίγει ένα παράθυρο το οποίο περιέχει σε ένα δεντρο συγκεντρωμένα όλα τα annotations που κάθε article περιέχει (Σχήμα 6.4)



Σχήμα 6.4 : Το παράθυρο "Analyzed GATE Data"

Το "Configure" ανοίγει το configuration window, στο οποίο ο χρήστης επιλέγει ποια feeds θα χρησιμοποιήσει. Με το πλήκτρο add, ο χρήστης μπορεί να δώσει το url του feed :

Σχήμα 6.5 : Το Configure μενού.

Τέλος, το "GATE Log" ανοίγει ένα text box το οποίο περιέχει τα logs που δημιουργεί το GATE στις αναλύσεις του κειμένου που έχει κάνει μέχρι τώρα, πράγμα χρήσιμο για το debugging του RSSGATE.

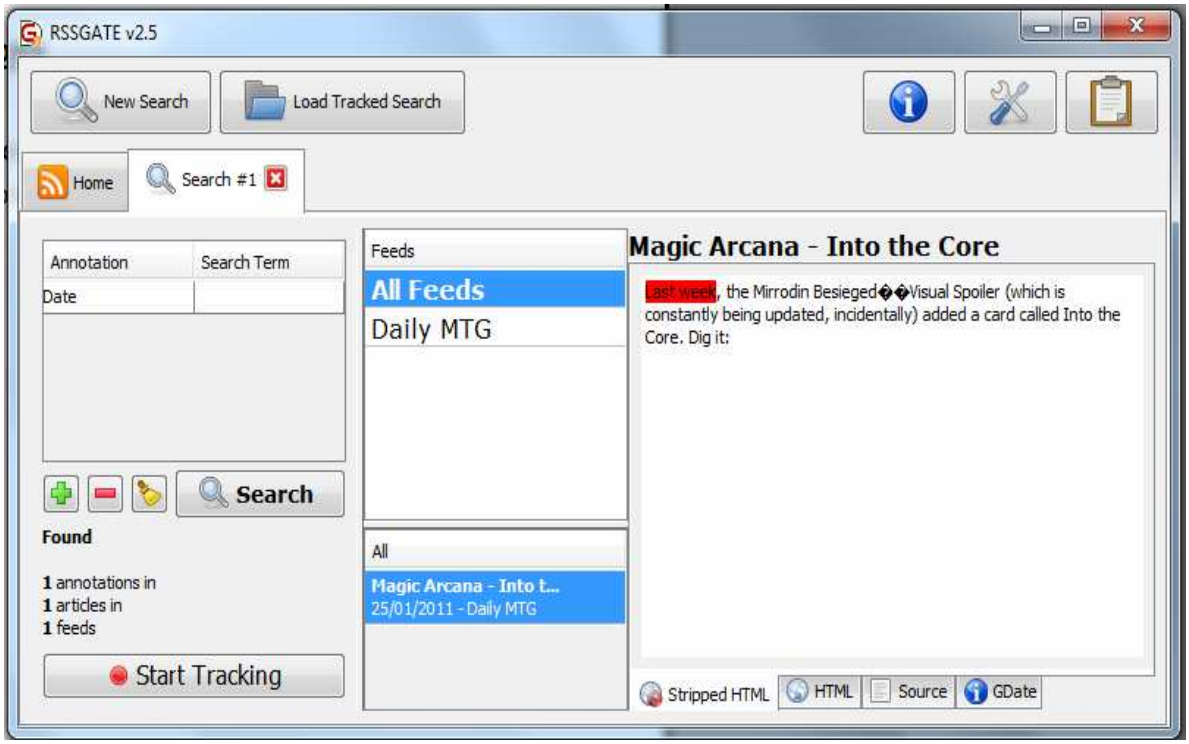
Στο Home tab, υπάρχει αριστερά μια λίστα με feeds και ακριβώς από κάτω

μια λίστα με articles του επιλεγμένου feed, καθώς και η επιλογή “All Feeds” η οποία δείχνει όλα τα άρθρα ανεξαρτήτως feed. Δεξιά είναι το κυρίως περιεχόμενο του feed, το οποίο χρησιμοποιώντας τα tabs στην κάτω μεριά του text box, εμφανίζεται ως :

- Η αυθεντική HTML μορφή του article όπως έρχεται από το feed, και χωρίς καμια επεξεργασία
- Stripped HTML χωρίς κανένα html tag, στην οποία μορφή μετατρέπεται κάθε έγγραφο αφού για να γίνει annotated θα πρέπει να είναι καθαρά σε μορφή text. Σε αντίθετη περίπτωση, τα HTML tags γίνονται annotated και δε βγαίνουν σωστά αποτελέσματα στην ανάλυσή του.
- Το HTML source του article
- Επίσης υπάρχει και ένας πίνακας GDate ο οποίος είναι ανάλογος με αυτόν στην εικόνα 6.4 αλλα ισχύει για το κάθε article ξεχωριστά.

6.3.2 ΑΝΑΖΗΤΗΣΗ ANNOTATIONS

Το κουμπί “New Search” ανοίγει ένα καινούργιο search tab στο οποίο ο χρήστης μπορεί να αναζητήσει annotations.



Σχήμα 6.6 : Το tab “Search”

Στα αριστερά υπάρχει ένα pull-down menu με τη βοήθεια του οποίου ο χρήστης επιλέγει τα κριτήρια αναζήτησης που θα χρησιμοποιηθούν, συγκεκριμένα τον τύπο annotation και την τιμή του. Αν το πεδίο Search Term είναι άδειο, τότε το RSSGATE θα εμφανίσει όλα άρθρα που περιέχουν τα annotations του τύπου αυτού. Επίσης, με το πλήκτρο “+” μπορεί ο χρήστης να προσθέσει και επιπλέον ζευγάρια τύπου Annotation-Search Term με σκοπό να περιοριστεί ο αριθμός των άρθρων που εμφανίζονται σε αυτά που τηρούν όλες τις προυποθέσεις.

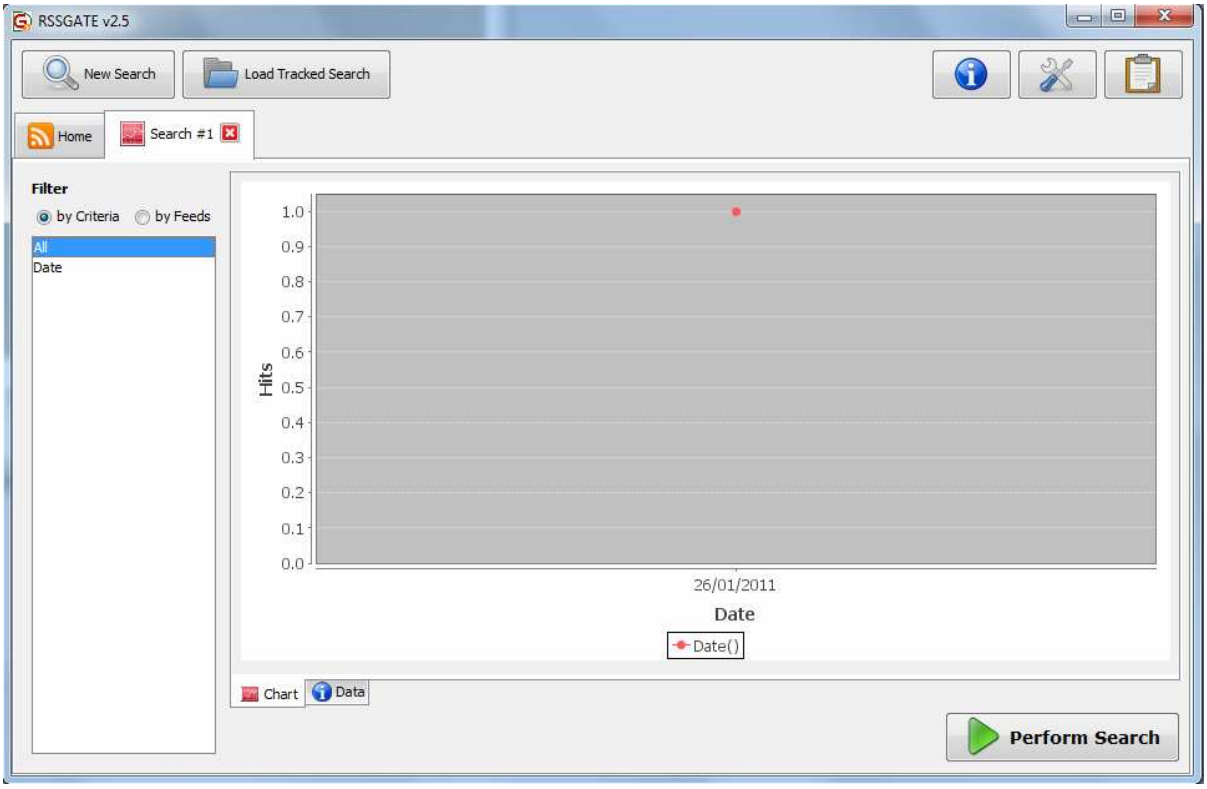
Με το πλήκτρο “-“ αφαιρείται το επιλεγμένο κριτήριο ενώ με το εικονίδιο της σκούπας αφαιρούνται όλα τα κριτήρια. Πατώντας το Search ξεκινάει η αναζήτηση.

Δεξιά απο τα κριτήρια υπάρχει ένας κατάλογος των φορτωμένων feeds και ένας viewer ανάλογος του αντίστοιχου στο κύριο μενου, με τη διαφορά ότι τα αποτελέσματα χρωματίζονται με διαφορετικά χρώματα, ανάλογα του τύπου annotation.

Τέλως, πατώντας το πλήκτρο Start Tracking, ο χρήστης ξεκινάει την διαδικασία tracking του συγκεκριμένου search, δίνοντας του όνομα και αποθηκεύοντάς το.

6.3.3 SEARCH TRACKING

Με την πίεση του πλήκτρου Load Tracked Search στην κεντρική οθόνη, ο χρήστης επιλέγει ένα ήδη tracked search, και ανοίγει ένα tab το οποίο παρουσιάζει στατιστικα στοιχεία σχετικά με την αναζήτηση σε βάθος χρόνου (Σχήμα 6.7).



Σχημα 6.7 : Το tab Tracked Search.

Αριστερά υπάρχει μια λίστα με τα κριτήρια της αναζήτησης, και η επιλογή να φιλτραριστούν τα αποτελέσματα ανα feed και ανα κριτήριο. Στα δεξιά παρουσιάζονται τα αποτελέσματα της μακροχρόνιας αναζήτησης, και υπάρχει η επιλογή για την οπτικοποίηση τους είτε σε μορφή διαγράμματος είτε σε πίνακα, παρουσιάζοντας για κάθε αυτόματη εκτέλεση της αναζήτησης τον αριθμό και τη μορφή των αποτελεσμάτων ανα ημερομηνία.

6.4 ΠΑΡΑΔΕΙΓΜΑ ΧΡΗΣΗΣ ΤΟΥ RSSGATE

Με τις δυνατότητες που παρέχει το RSSGATE, και οι οποίες περιγράφηκαν στα προηγούμενα κεφάλαια, υπάρχει η δυνατότητα για ενδελεχή έρευνα πάνω σε διάφορους τομείς. Καθώς υπάρχει η δυνατότητα αυτόματης ανάλυσης κειμένου μέσα απο RSS Feeds και μάλιστα σε βάθος χρόνου, το μόνο που χρειάζεται είναι κάποια RSS Feeds τα οποία να παρέχουν χρήσιμα δεδομένα μετά την ανάλυση των άρθρων σε annotations και ταυτόχρονα να ανανεώνονται καθημερινά. Δυστυχώς τα περισσότερα RSS Feeds δεν πληρούν αυτές τις προυποθέσεις.

Ένας τύπος feeds ο οποίος είναι αρκετά τυποποιημένος, παρέχει έυχρηστα και parsable δεδομένα και ανανεώνεται συνεχώς, είναι οι αγγελίες real estate (listings).

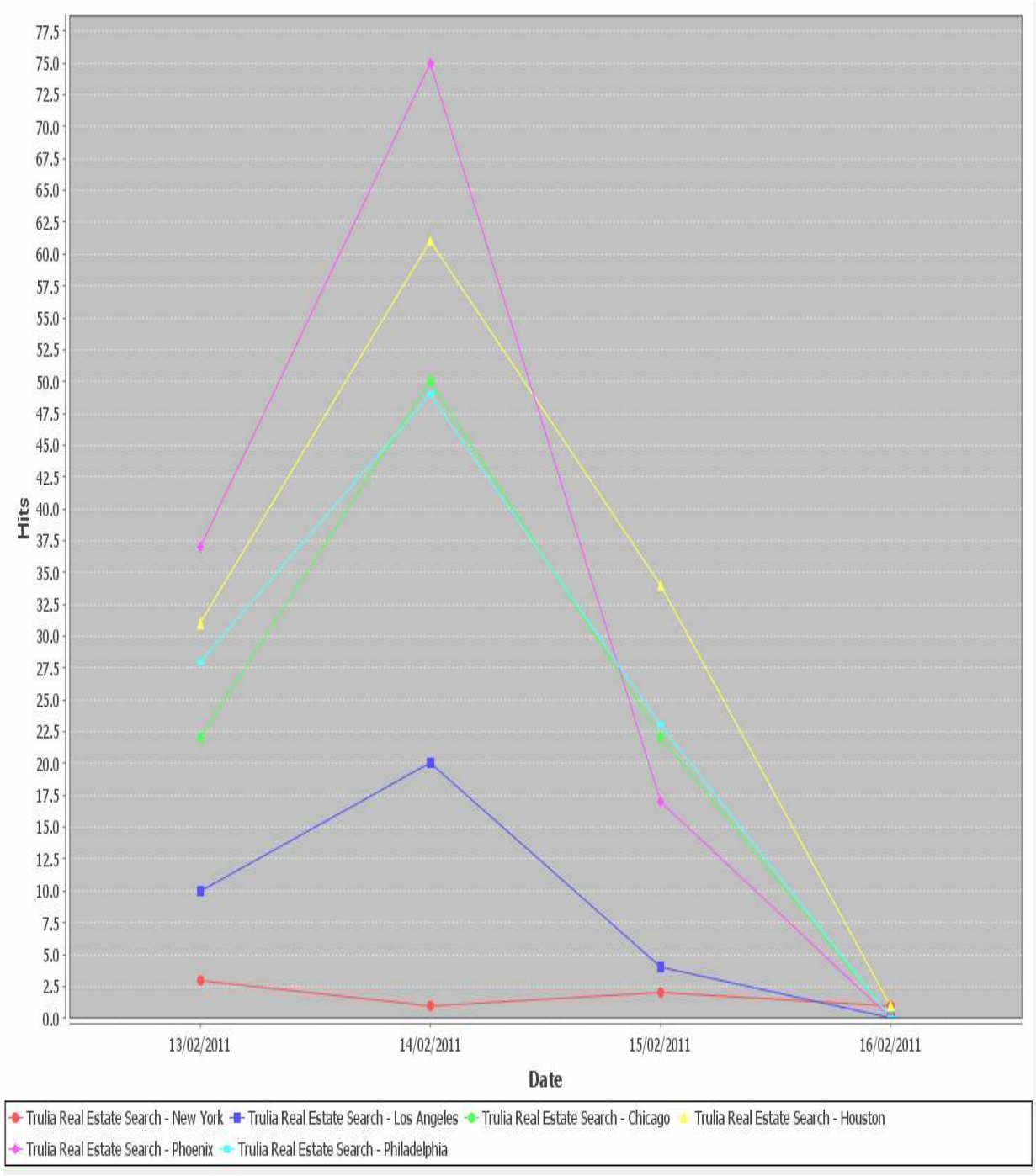
Έτσι, στο παρόν παράδειγμα, το RSSGATE έχει φορτωμένα 6 feeds, απο τον ίδιο aggregator, τα οποία αντιπροσωπεύουν το καθένα μια από τις 6 μεγαλύτερες σε πληθυσμό πόλεις των Ηνωμένων Πολιτειών, και περιέχουν αγγελίες για σπίτια προς πώληση.

Το RSSGATE τρέχει συνεχώς στο background και περιέχει 3 διαφορετικά tracked searches τα οποία τρέχουν και στα 6 feeds ταυτόχρονα και ελέγχουν τις τιμές των σπιτιών στα άρθρα. Τα terms είναι τα εξής (με επεξήγηση του expression που δώθηκε στα πεδία Annotation και Search Term) :

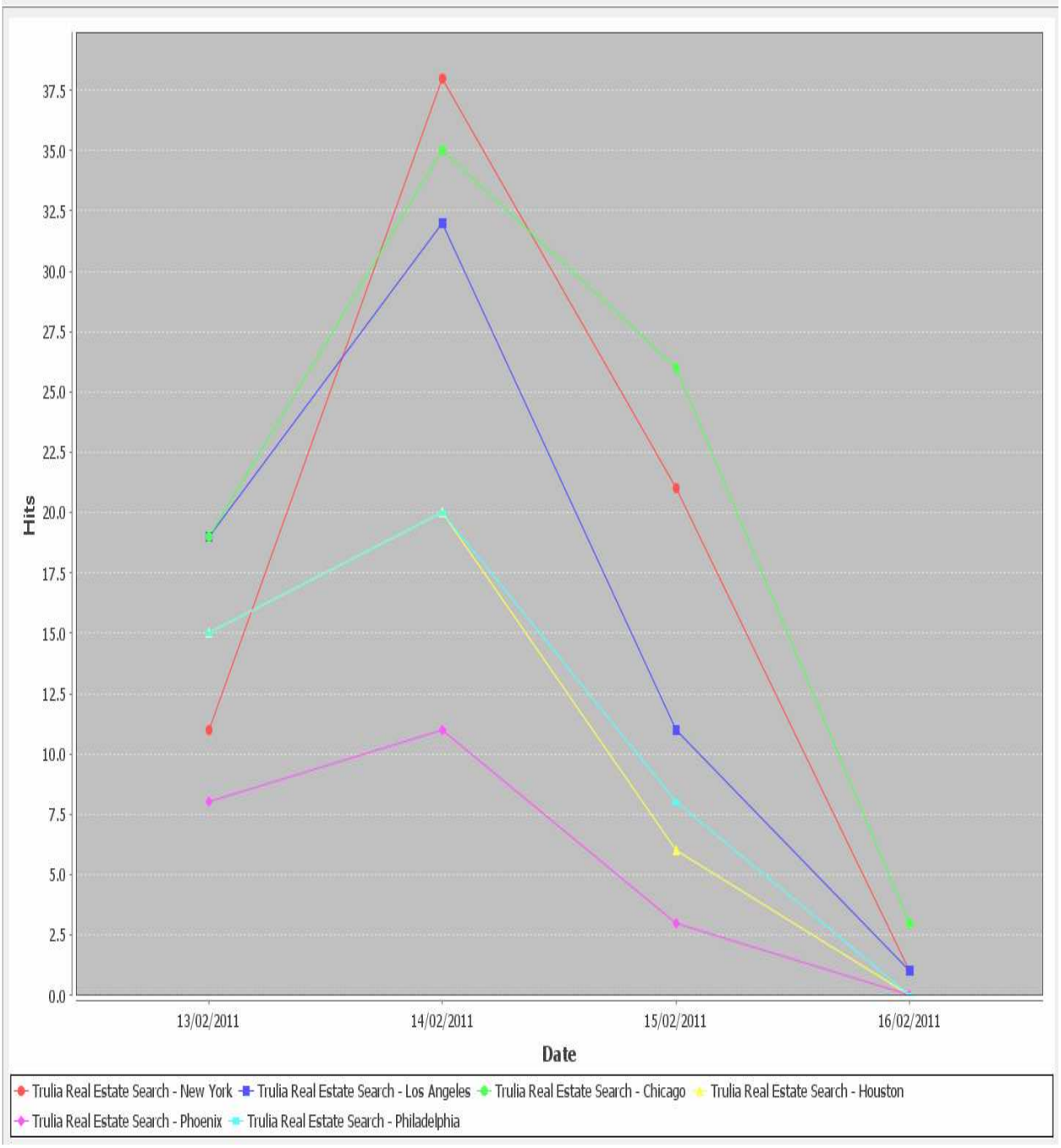
- Τιμή μικρότερη από 250.000$ (Annotation Money, Search Term : <250000).
- Τιμή ανάμεσα σε 250.000$ και 600.000$ (Annotation Money, Search Term : >250000&<600000).
- Τιμή μεγαλύτερη απο 600.000$ (Annotation Money, Search Term : >600000).

Μετά απο τέσσερις ημέρες, τα αποτελέσματα είναι τα εξής:

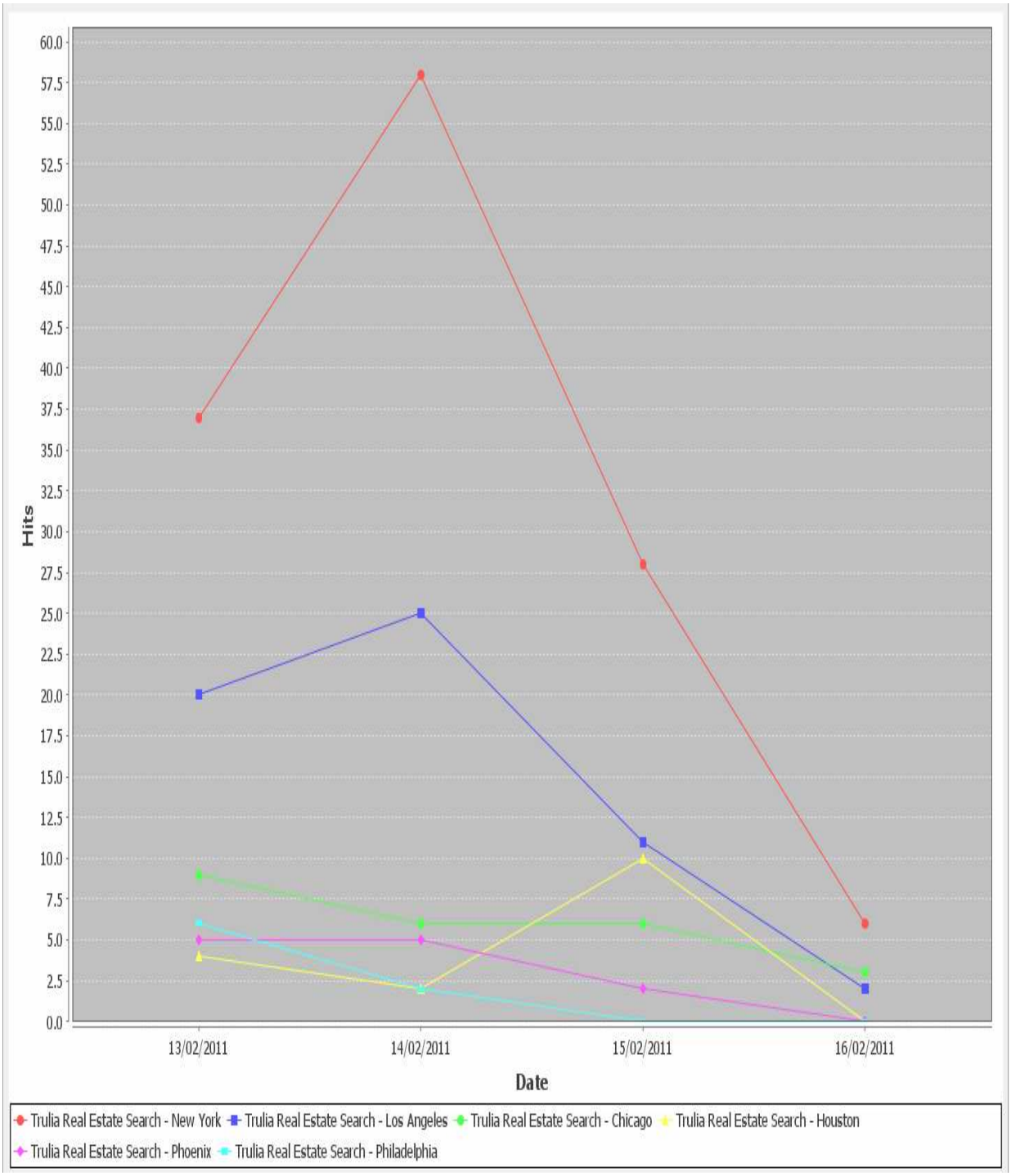
- Για σπίτια με τιμή μικρότερη των 250.000$:



- Για σπίτια με τιμή μεταξύ 250.000$ και 600.000$:



- Για σπίτια με τιμή μεγαλύτερη απο 600.000$:



ΚΕΦΑΛΑΙΟ 7

ΕΠΙΛΟΓΟΣ

7.1 ΔΥΣΚΟΛΙΕΣ ΚΑΙ ΠΡΟΒΛΗΜΑΤΑ

Κατα τη διάρκεια της πτυχιακής εργασίας, παρουσιάστηκαν αρκετές δυσκολίες, οι περισσότερες απο τις οποίες ήταν απλά προβλήματα τα οποία ξεπεράστηκαν. Κάποια απο αυτά όμως, των οποίων η λύση υπερέβαινε τη σκοπιά της παρούσας εργασίας, επηρέασαν τον σχεδιασμό της και την άλλαξαν αρκετά σχετικά με την αρχική ιδέα που είχα για την εργασία. Για αυτό το λόγο, είναι ενδιαφέρον να αναφερθούν.

Το σημαντικότερο πρόβλημα που αντιμετώπισα ήταν η σχεδόν ολοκληρωτική έλλειψη βιβλιογραφίας σχετικά με την NLP στην ελληνική γλώσσα. Σε άλλους τομείς μπορεί αυτό να μην αποτελεί πρόβλημα, αλλά στον συγκεκριμένο, λόγω του οτι ασχολείται με την φυσική γλώσσα, είναι σημαντικό εμπόδιο. Όλη η έρευνα (και η συγγραφή της εργασίας), έπρεπε να γίνει με τα δεδομένα της NLP της Αγγλικής γλώσσας, η οποία έχει διαφορές απο την Ελληνική.

Παραπλήσιο πρόβλημα ήταν και το γεγονός ότι το GATE δεν παρέχει βιβλιοθήκες processing resources (π.χ. το CREOLE) για την Ελληνιική γλώσσα, και έτσι δε μπορεί να γίνει ανάλυση κειμένων γραμμένων στη γλώσσα αυτή, ούτε στο GATE Developer αλλά ούτε και στο GATE Embedded, και συνεπώς και στο πρόγραμμα RSSGATE χωρίς να γραφτούν καινούργια εργαλεία παραπλήσια του ANNIE στην Ελληνική γλώσσα, κάτι για το οποίο ίσως να μην έχει γίνει καν η απαραίτητη γλωσσολογική έρευνα στην Ελλάδα για να επιτευχθεί.

Τέλος, κάποια προβλήματα υπήρξαν και στην ανάπτυξη του RSSGATE, καθώς υπήρξαν ζητήματα ασυμβατότητας μεταξύ των βιβλιοθηκών του GATE και της Java, καθώς και ζητήματα διαχείρησης μνήμης, τα οποία όμως λύθηκαν χωρις ιδιαίτερο κόπο.

7.2 ΕΠΕΚΤΑΣΕΙΣ/ΣΥΝΕΧΙΣΗ ΤΗΣ ΕΡΕΥΝΑΣ

Η παρούσα πτυχιακή είναι μια ώς επι το πλείστον ολοκληρωμένη και αυτοτελής έρευνα πάνω στο GATE Framework και γενικά στην NLP. Υπάρχουν αρκετοί τομείς στους οποίους αυτή η έρευνα θα μπορούσε να ειδικευτεί σε κάποια ενδεχόμενη συνέχειά της.

Η σημαντικότερη επέκταση (και κάτι που σκοπεύω να κάνω στο πλαίσιο μεταπτυχιακών σπουδών) είναι η μελέτη των εφαρμογών NLP στην Ελληνική γλώσσα. Συγκεκριμένα, το πλαίσιο GATE είναι πλήρως επεκτάσιμο και εκτιμώ ότι είναι σχετικά απλό να δεχτεί αλφάβητα άλλα εκτός του λατινικού.

Ωστόσο, για να μπορεί το GATE εκτελέσει εργασίες NLP στην Ελληνική γλώσσα (η και σε κάθε άλλη γλώσσα φυσικα) θα πρέπει να δημιουργηθούν processing resources (Λεξικά, γραμματικοί κανόνες κλπ) απο την αρχή, κάτι το οποίο είναι εξαιρετικά δύσκολο και όπως προανέφερα προυποθέτει εκτεταμένες μελέτες όχι μόνο στο GATE και στο NLP και NLU της Ελληνικής γλώσσας, κάτι το οποίο προυποθέτει βαθιές γνώσεις γλωσσολογίας.

Φυσικά, τα ωφέλη μιας τέτοιας εργασίας θα είναι πολύ σημαντικά, αφού κάτι τέτοιο δεν έχει γίνει ποτε (τουλάχιστον σε ακαδημαικό επίπεδο), και είναι το πρώτο (και απαραίτητο) βήμα για την υλοποίηση σημειολογικής αναζήτησης (semantic search) στην Ελληνική φυσική γλώσσα.

ΒΙΒΛΙΟΓΡΑΦΙΚΕΣ ΑΝΑΦΟΡΕΣ

- *H. Cunningham, D. Maynard, K. Bontcheva, V. Tablan. GATE: A Framework and Graphical Development Environment for Robust NLP Tools and Applications. Proceedings of the 40th Anniversary Meeting of the Association for Computational Linguistics (ACL'02). Philadelphia, July 2002.*
- *K. Bontcheva, V. Tablan, D. Maynard, H. Cunningham. Evolving GATE to Meet New Challenges in Language Engineering.*
- *M. Dowman, V. Tablan, H. Cunningham and B. Popov. Web-Assisted Annotation, Semantic Indexing and Search of Television and Radio News. 14th International World Wide Web Conference. Chiba, Japan, 2005.*
- *H. Cunningham. Software Architecture for Language Engineering. PhD Thesis, University of Sheffield, 2000.*
- Manu Conchady . Building Search Applications with Lucene, LingPipe and Gate. Mustru Publishing, 2008.
- Graham Wilcock . Linguistic Annotation and Analysis. 2009.
- The GATE User Guide, The GATE Team, 2010.
- H. Cunningham. Software Architecture for Language Engineering. *PhD Thesis*, University of Sheffield, 2000.
- http://gate.ac.uk