ΠΑΝΕΠΙΣΤΗΜΙΟ ΚΡΗΤΗΣ
ΣΧΟΛΗ ΘΕΤΙΚΩΝ ΕΠΙΣΤΗΜΩΝ
ΤΜΗΜΑ ΕΠΙΣΤΗΜΗΣ ΥΠΟΛΟΓΙΣΤΩΝ

Σύστημα Υπομνηματισμού Ηλεκτρονικών Εγγράφων στο Διαδίκτυο

Μανόλης Τζομπανάκης

Μεταπτυχιακή εργασία

Ηράκλειο, Φεβρουάριος 2003

ΠΑΝΕΠΙΣΤΗΜΙΟ ΚΡΗΤΗΣ
ΣΧΟΛΗ ΘΕΤΙΚΩΝ ΕΠΙΣΤΗΜΩΝ
ΤΜΗΜΑ ΕΠΙΣΤΗΜΗΣ ΥΠΟΛΟΓΙΣΤΩΝ

«Σύστημα Υπομνηματισμού Ηλεκτρονικών Εγγράφων στο Διαδίκτυο»

Μεταπτυχιακή εργασία
Υποβλήθηκε από τον
Μανόλη Σ. Τζομπανάκη
ως μερική απαίτηση για την απόκτηση Μεταπτυχιακού Διπλώματος Ειδίκευσης

Συγγραφέας:

_____
Μανόλης Τζομπανάκης
Τμήμα Επιστήμης Υπολογιστών

Εισηγητική Επιτροπή:

_____
Πάνος Κωνσταντόπουλος, Καθηγητής, Επόπτης

_____
Δημήτρης Πλεξουσάκης, Αναπληρωτής Καθηγητής, Μέλος

_____
Βασίλης Χριστοφίδης, Επίκουρος Καθηγητής Μέλος

Δεκτή:

_____
Πάνος Κωνσταντόπουλος, Καθηγητής,
Πρόεδρος Επιτροπής Μεταπτυχιακών Σπουδών

Σύστημα Υπομνηματισμού Ηλεκτρονικών Εγγράφων στο Διαδίκτυο

Μανόλης Τζομπανάκης

Μεταπτυχιακή Εργασία

Τμήμα Επιστήμης Υπολογιστών
Πανεπιστήμιο Κρήτης

Περίληψη

Μια ψηφιακή βιβλιοθήκη παρέχει στους χρήστες της υπηρεσίες οι οποίες αφορούν την πρόσβαση σε ηλεκτρονικά έγγραφα. Ταυτόχρονα, εάν γίνει χρήση του WWW επιτυγχάνεται η προσπέλαση των εγγράφων από οποιοδήποτε κόμβο του Internet. Το γεγονός όμως ότι χάνεται η επαφή με το χαρτί ως φυσικό μέσο αποτύπωσης των εγγράφων καθιστά αναγκαία την εύρεση μιας νέας διαδικασίας για την συγγραφή σημειώσεων πάνω σε αυτά ώστε τα ηλεκτρονικά έγγραφα να μην χάνουν μέρος της λειτουργικότητας, που παρέχουν τα έγγραφα εκείνα, τα οποία έχουν μέσο αποθήκευσης το χαρτί.

Η εργασία μας ασχολείται ακριβώς με τη δημιουργία ενός συστήματος υπομνηματισμού ηλεκτρονικών εγγράφων. Δηλαδή ο σκοπός μας είναι η διαχείριση (δημιουργία, αποθήκευση, ανάκτηση) σημειώσεων πάνω σε ηλεκτρονικά έγγραφα καθώς και η αναζήτηση εγγράφων με βάση αυτές. Γίνεται χρήση του Διαδικτύου ώστε να μπορεί να χρησιμοποιείται από ομάδες που συνεργάζονται στο ίδιο θέμα και βρίσκονται σε διαφορετικούς χώρους διευκολύνοντας έτσι την πρόσβαση στις σημειώσεις μεταξύ των μελών των ομάδων.

Καταλήξαμε σε τύπους σημειώσεων που παρουσιάζουν τη γνώμη του αναγνώστη σε σχέση με ένα έγγραφο ή με ένα μέρος του εγγράφου. Ακόμη δημιουργήσαμε τύπους σημειώσεων οι οποίοι φανερώνουν τη σχέση που έχουν τα έγγραφα μεταξύ τους, όπως καταγράφεται μέσα σε αυτά και όχι απλά τη προσωπική γνώμη του συγγραφέα των σημειώσεων για τα έγγραφα. Το μοντέλο των σημειώσεων το παραστήσαμε με τη γλώσσα TELOS και για τα πεδία των σημειώσεων κάναμε χρήση επιλεγμένων Dublin Core metadata. Με τη χρήση του πρωτοκολλου

HTTP κάναμε δυνατή την εύκολη χρησιμοποίηση του συστήματος μας από το Διαδίκτυο. Για την υλοποίηση χρησιμοποιήσαμε τη γλώσσα προγραμματισμού Java (httpServlets) καθώς επίσης κάναμε χρήση του Web server Apache Tomcat. Τέλος χρησιμοποιήσαμε ως βάση αποθήκευσης των σημειώσεων το SIS κατασκευασμένο στο Ινστιτούτο Πληροφορικής του ΙΤΕ.

Επόπτης:: Πάνος Κωνσταντόπουλος, Καθηγητής
Τμήμα Επιστήμης Υπολογιστών, Πανεπιστήμιο Κρήτης

“An Annotation System for digital documents”

Manolis Tzobanakis
Master of Science Thesis

Department of Computer Science
University of Crete

Abstract

A digital library provides services that enable access to digital documents. At the same time the use of WWW allows users to access the documents from any node on the Internet. The fact that the paper is no longer used as the mean of storing the documents necessitates the device of a new procedure for the creation of annotations on the digital documents.

This thesis deals with the creation of an annotation system of digital documents. Our goal is the management (creation, storage and retrieval) of annotations on digital documents as well as the process of retrieving a document based on the annotations. Internet is used in order to support networked scholarship communities and collaboration activities between groups of scientists located at different places making easier for the group members to have access to the annotations of the other members of the same group.

Many annotation types are provided. There are two main categories of annotations: those concerning the annotations assigned on one document, and those that are related to more than one documents. The representation language TELOS is used in order to create the information model. For the annotation record fields the Dublin Core Metadata have been used. Also, by using the HTTP protocol, the easy use of our system over the Internet was made possible. The application was implemented in Java and also the Apache Web Server was used. Finally as the storage base for our application the Semantic Index System was used.

Supervisor: Panos Constantopoulos,

ProfessorDepartment of Computer Science, University of Crete

Περιεχόμενα

Κατάλογος Πινάκων

Πίνακας Σχημάτων

1. Εισαγωγή

1.1. Γενικό πρόβλημα – πλαίσιο

Παλαιότερα η διακίνηση των πληροφοριών γινόταν αποκλειστικά με έντυπο υλικό. Το χαρτί όμως το οποίο αποτελούσε την αποθήκη των πληροφοριών γινόταν η αιτία για την συχνή απώλεια χρήσιμης πληροφορίας καθώς ο χρόνος και η συχνή χρήση των συγγραμμάτων το έφθειραν αρκετά καθώς είναι ευαίσθητο υλικό. Στη σύγχρονη εποχή είναι πλέον διαδεδομένη η χρήση ψηφιακών μέσων για την αποθήκευση εγγράφων.

Έχουν αναπτυχθεί λοιπόν οι λεγόμενες ψηφιακές βιβλιοθήκες οι οποίες παρέχουν στους χρήστες τους υπηρεσίες αποθήκευσης, αναζήτησης και ανάκτησης των εγγράφων που περιέχουν. Μάλιστα τα έγγραφα πλέον δεν είναι απλό κείμενο αλλά μπορεί να έχουν ήχο και εικόνα παρέχοντας με τον τρόπο αυτό εναλλακτικούς τρόπους αναπαράστασης της πληροφορίας που περιέχουν. Ακόμη τα έγγραφα μπορεί να έχουν μία διακεκριμένη δομή η οποία να βοηθά στην καλύτερη επεξεργασία και διαχείριση της πληροφορίας που περιέχουν αυτά. Επίσης μέσω των δικτύων τα ψηφιακά έγγραφα πλέον είναι προσβάσιμα από οποιοδήποτε σημείο όπως πχ από ενδιαφερόμενους που βρίσκονται σε απομακρυσμένα μέρη.

Παρόλα τα πλεονεκτήματα που προσφέρουν οι συλλογές από ηλεκτρονικά έγγραφα παρουσιάζουν ορισμένα μειονεκτήματα σε σχέση με τις κλασσικές βιβλιοθήκες συγγραμμάτων. Όταν το υλικό αποθήκευσης της πληροφορίας ήταν το χαρτί, ήταν πολύ απλή υπόθεση η συγγραφή σημειώσεων από τους αναγνώστες των εγγράφων είτε πάνω στα συγγράμματα είτε συνηθέστερα σε ξεχωριστές καρτέλες οι οποίες με τον καιρό αποτελούσαν πολύ σημαντική πηγή γνώσης για τους αναγνώστες των εγγράφων. Επίσης η γνώση που αποτυπώνεται σε αυτές τις σημειώσεις, η οποία αποτελεί την προσωπική γνώμη του αναγνώστη θα αποτελέσει σίγουρα αντικείμενο μελέτης του ιδίου ή ακόμη και άλλων αναγνωστών κάποια στιγμή στο μέλλον.

Ζητούμενο λοιπόν από τις οργανωμένες συλλογές ηλεκτρονικών εγγράφων είναι η δυνατότητα να παρέχουν μία υπηρεσία υπομνηματισμού. Δηλαδή μία υπηρεσία για την δημιουργία, αποθήκευση και ανάκτηση σημειώσεων στα ηλεκτρονικά έγγραφα που περιέχονται σε αυτές ώστε να καλύψουν αυτό το κενό που παρουσιάζεται με τα νέα μέσα αποθήκευσης των ψηφιακών πλέον εγγράφων.

1.2 Περιγραφή της εργασίας

Στην εργασία μας αυτή σχεδιάσαμε και υλοποιήσαμε ένα σύστημα υπομνηματισμού σε ηλεκτρονικά έγγραφα. Με το σύστημα αυτό παρέχουμε τη δυνατότητα για δημιουργία, αποθήκευση και ανάκτηση σημειώσεων σε ηλεκτρονικά έγγραφα. Μας απασχόλησαν θέματα που σχετίζονται

α) με τη δημιουργία νέων τύπων σημειώσεων καθώς εκείνοι που παρέχονταν από τα ήδη υπάρχοντα συστήματα δεν κάλυπταν όλες τις ανάγκες μας

β) την περιγραφή της σημείωσης με ένα σύνολο από ευρέως αποδεκτά πεδία τα οποία να είναι ικανοποιητικά για την περιγραφή των σημειώσεων

γ) την παράσταση του περιεχομένου της σημείωσης με δομημένη μορφή ώστε να είναι δυνατή η αποτελεσματική παρουσίαση της και η δυνατότητα επεξεργασίας της

δ) την δυνατότητα χρήσης του συστήματος μας μέσω του διαδικτύου ώστε να παρέχουμε τη δυνατότητα υποσημειώσεων σε απομακρυσμένους χρήστες και

τέλος ε) την προσαρμογή του συστήματος μας με τρόπο ώστε να λειτουργήσει ως μία υπηρεσία που παρέχεται σε συνδιασμό με ένα σύνολο άλλων υπηρεσιών στα πλαίσια μίας ψηφιακής βιβλιοθήκης.

1.3 Τα μέρη του συστήματος

Για την μοντελοποίηση των σημειώσεων χρησιμοποιήσαμε τη γλώσσα TELOS. Για την αποθήκευση των σημειώσεων κάναμε χρήση του συστήματος σημασιολογικού ευρετηριασμού (SIS)[SIS1]. Το σύστημα μας πρόκειται να χρησιμοποιηθεί ως βάση για την αποθήκευση των σημειώσεων που αναφέρονται σε έγγραφα τα οποία βρίσκονται κατανεμημένα σε διαφορετικές βάσεις στο διαδίκτυο. Για το λόγο αυτό προέκυψε η ανάγκη για το σχεδιασμό ενός προτοκόλου σύμφωνα με το οποίο θα γίνεται η επικοινωνία μεταξύ των clients και καθενός server. Το πρωτόκολλο αυτό είναι το OLP [OLP] και έχει σχεδιαστεί έτσι ώστε να επιτρέπει σε διάφορες εφαρμογές – υπηρεσίες του διαδικτύου να επικοινωνούν μεταξύ τους με τρόπο ώστε να επιτυγχάνεται η εύκολη προσθήκη νέων υπηρεσιών χωρίς να επηρεάζεται η λειτουργία των υπολοίπων.

1.4 Τα προβλήματα που ανέκυψαν

Στην πορεία του σχεδιασμού δημιουργήθηκαν προβληματισμοί που αφορούσαν τόσο τις λειτουργίες όσο και ζητήματα αρχιτεκτονικής αλλά και διαχείρισης του συστήματος. Ένα πρόβλημα που μας απασχόλησε ήταν οι τύποι των σημειώσεων. Ήταν δεδομένη η σκέψη για δημιουργία σημειώσεων που απλά καταγράφουν την σκέψη του αναγνώστη σχετικά με κάποιο έγγραφο. Όλα τα συστήματα που είχαμε εξετάσει όριζαν τέτοιους τύπους σημειώσεων. Εκείνο που βελτιώσαμε σε αυτήν την κατηγορία των σημειώσεων είναι η καλύτερη οργάνωση των σημειώσεων καθώς και η προσθήκη περισσότερων νέων τύπων σημειώσεων. Επίσης δημιουργήσαμε μία κατηγορία σημειώσεων η οποία να συσχετίζει έγγραφα μεταξύ τους και όχι απλά να αντιστοιχεί τη γνώμη του αναγνώστη σε κάποιο από αυτά.

Άλλο ένα θέμα αφορούσε την αποθήκευση των σημειώσεων. Το ερώτημα είναι εάν οι σημειώσεις πρέπει να αποθηκεύονται στην ίδια βάση που αποθηκεύονται τα έγγραφα ή εάν θα έπρεπε να αποθηκεύονται σε διαφορετική βάση. Εμείς θεωρήσαμε τις σημειώσεις σαν διαφορετικά αυτόνομα έγγραφα και τα χειρισθήκαμε με ανάλογο τρόπο εξασφαλίζοντας τους διαφορετική βάση δεδομένων από εκείνη των εγγράφων. Με τον τρόπο αυτό δεν επιβαρύνουμε τον server που διατηρεί τα έγγραφα. Επίσης δεν μας ενδιαφέρει το DBMS που διατηρεί τα έγγραφα το οποίο μπορεί να το διαχειρίζεται μία διαφορετική από τη δική μας υπηρεσία. Γενικά η ύπαρξη διαφορετικής βάσης για την φύλαξη των σημειώσεων από εκείνη που χρησιμοποιούμε για την αποθήκευση των εγγράφων μας δίδει μια ανεξαρτησία – ευελιξία.

Ένα άλλο ζήτημα που μας απασχόλησε ήταν η επιλογή της τεχνολογίας που θα χρησιμοποιούσαμε για την υλοποίηση της εφαρμογής. Επιλέξαμε την Java η οποία μας παρέχει ανεξαρτησία από τα διαφορετικά λειτουργικά συστήματα που ενδεχομένως συναντάμε στο web καθώς και πολλά εργαλεία (web servers, XML parsers κ.λπ.) τα οποία μας διεκόλυναν στην υλοποίηση. Επίσης με τη χρήση της κλάσης HttpServlets κατορθώσαμε να «προσωμοιώσουμε» το HTTP χωρίς ιδιαίτερες δυσκολίες υλοποιώντας μάλιστα εύκολα το Open Library Protocol όπως αναφέρουμε σε επόμενη παράγραφο κατορθώνοντας έτσι να επικοινωνούμε με άλλες ενεξάρτητες εφαρμογές και όλες μαζί να παρέχουμε υπηρεσίες ψηφιακή βιβλιοθήκης στον χρήστη μέσω του web.

2. Υπομνηματισμός σε έγγραφα

2.1. Προηγούμενα συστήματα

Στα συστήματα που εξετάσαμε διαπιστώσαμε διαφορετικές προσεγγίσεις. Σε άλλα συστήματα οι σημειώσεις αφορούν μέρος του κειμένου, σε άλλα αφορούν ολόκληρο το κείμενο ή σε άλλα ακόμη και τα δύο(Amaya)[AMAYA]. Στο σύστημα CritSuit [CRIT] **το τέλος του εγγράφου** υπάρχουν σύνδεσμοι προς HTML σελίδες που έχουν σαν περιεχόμενο τις σημειώσεις αυτές. Είδαμε σημειώσεις να γίνονται **σε σημεία που ορίζονται από τον συγγραφέα** του εγγράφου (σύστημα CoNote) [CONOTE]όμως σε άλλα συστήματα οι σημειώσεις γίνονται σε **τυχαία σημεία** μέσα στο κείμενο (Amaya, ComMentor [COMENTOR]κ.α.). Στην προσέγγιση αυτή βέβαια υπάρχουν προβλήματα που αφορούν την εγκυρότητα της σημείωσης σε περίπτωση που το κείμενο υποστεί μεταβολές από την κατάσταση στην οποία βρισκόταν όταν έγινε η σημείωση.

Τύποι και είδη σημειώσεων.

- **Είδη σημειώσεων**

Λέγοντας είδος σημείωσης εννούμε τον τρόπο που γίνεται η σημείωση στο έγγραφο[Τούλης94]. Τα διάφορα είδη που έχουμε δεί μπορούν να καταταχθούν στις εξής κατηγορίες :

1. Εικονίδιο

Στο Amaya και στο CoMentor υπάρχουν εικονίδια που συμβολίζουν ακριβώς ότι υπάρχει σημείωση από κάποιον αναγνώστη του εγγράφου στο σημείο αυτό. Τα εικονίδια μπορεί είναι ίδια κάθε φορά (Amaya) ή να είναι διαφορετικά ανάλογα με τον τύπο της σημείωσης (CoMentor). Μάλιστα στο τελευταίο, υπάρχει η δυνατότητα το εικονίδιο να είναι μία μικρή φωτογραφία (16x22 pixels) του συντάκτη της σημείωσης. Πατώντας το εικονίδιο εμφανίζεται ένα νέο παράθυρο με την σημείωση και τα στοιχεία εκείνου που την έχει κάνει.

2. Σύνδεσμος (link)

Υπάρχει η δυνατότητα μέσω συνδέσμων που υπάρχουν σε σημεία του εγγράφου να οδηγηθούμε στις σημειώσεις οι οποίες είναι HTML σελίδες (CritSuit).

3. Υπογράμμιση

Οι λέξεις ή φράσεις που σχολιάζονται είναι υπογραμμισμένες με χαρακτηριστικό χρώμα. Στο ThirdVoice υπάρχει μια πορτοκαλί γραμμή κάτω από το μέρος του εγγράφου που έχει σχολιαστεί.

4. Ηχητική ένδειξη

Στη βιβλιογραφία διαπιστώσαμε ότι υπάρχουν και άλλα είδη σημειώσεων όπως ένας όρος ενός λεξιλογίου που σημαίνει κάτι ξεχωριστό σε όσους γνωρίζουν το λεξιλόγιο αυτό.

- **Τύποι σημειώσεων**

Ο τύπος σημείωσης σημαίνει εκείνο που εκφράζει η σημείωση, δηλαδή το νόημα της.
Έχουμε διαπιστώσει ότι σχεδόν κάθε Annotation System έχει ορίσει δικούς του τύπους σημείωσης.
Στο Amaya δίδεται η δυνατότητα στον χρήστη να διαλέξει ανάμεσα σε αρκετούς διαθέσιμους τύπους ποιος από από όλους είναι κατάλληλος για τη σημείωση που καταχωρεί στο έγγραφο ("Annotation", "Comment", "Query" κλπ). Στο HyperNews χρησιμοποιείται η «Απάντηση», η «ερώτηση», η «συμφωνία», η «διαφωνία» και το «γενικό σχόλιο». Στην Principia Cybernetica [CYBERNETICA] προτείνονται οι τύποι «Απόρριψη», «συμφωνία», «διόρθωση», «εικονογράφηση» και «σχόλιο».

Τα συστήματα που εξετάσαμε λειτουργούν στο περιβάλλον του WWW. Το τεράστιο πλεονέκτημα που έχουν έναντι εκείνων των συστημάτων που δεν λειτουργούν μέσω δικτύου είναι η δυνατότητα που υπάρχει για τους χρήστες να διαβάσουν τις σημειώσεις που έχουν κάνει άλλοι πριν από αυτούς και έχουν σχέση με το έγγραφο που τους ενδιαφέρει. Για να γίνει αυτό εφικτό χρησιμοποιούνται διάφορες προσεγγίσεις σε σχέση με τον server που θα εξυπηρετεί την ανάκτηση των σημειώσεων. Η γενική ιδέα πάντως σε κάθε περίπτωση είναι ότι οι σημειώσεις είναι ανεξάρτητες από το αρχικό κείμενο και οτιδήποτε τις αφορά δεν συμβαίνει πάνω σε αυτό. Δηλαδή είναι εξωτερικές και μπορούν να αποθηκευτούν είτε τοπικά είτε σε κάποιο μηχάνημα που θα λειτουργεί σαν Annotation Server. Στο σημείο αυτό μπορούν να τεθούν μερικά ερωτήματα που σχετίζονται με τον τρόπο αποθήκευσης των σημειώσεων. Σε ποιο μηχάνημα θα αποθηκεύονται οι σημειώσεις; α) Είναι καλύτερα να αποθηκεύονται στο μηχάνημα που βρίσκεται το έγγραφο το οποίο σχολιάζεται; β) Είναι καλύτερα να αποθηκεύονται στο μηχάνημα

εκείνου που κάνει τη σημείωση; γ) είναι καλύτερα να υπάρχουν μηχανήματα που θα λειτουργούν σαν Annotation Servers και θα είναι ανεξάρτητα από τα μηχανήματα στα οποία βρίσκονται τα έγγραφα αλλά και από το μηχάνημα εκείνου που κάνει τη σημείωση;

Γενικά οι clients δεν είναι δυνατό να γνωρίζουν όλους τους servers που έχουν αποθηκεύσει σημειώσεις. Έτσι ο client θα πρέπει να επικοινωνεί με έναν server κάθε φορά (τον ίδιο πάντα) και στη συνέχεια οι servers μεταξύ τους θα μπορούν να ανταλλάσουν πληροφορίες και να ενημερώνουν ο ένας τον άλλο για την σωστή πληροφόρηση του client. το Amaya δεν ακολουθείται αυτή η πρακτική αλλά ο χρήστης καλείται να δηλώσει ο ίδιος τους servers από τους οποίους θα παίρνει τις σημειώσεις γεγονός που περιορίζει τη λειτουργικότητα.

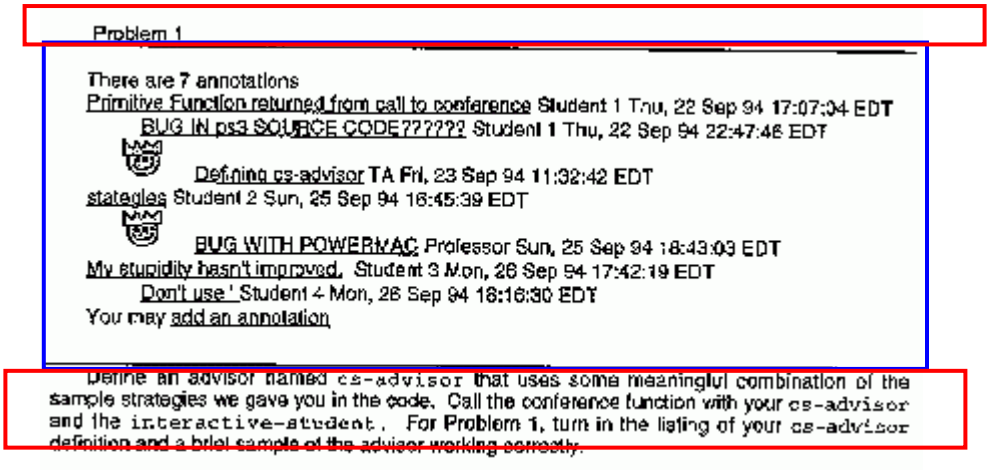
Στην αναζήτηση μας για Annotation Systems περιοριζόμαστε σε εκείνα που έχουν δυνατότητα χρήσης στο www. Χωρίσαμε τα συστήματα που βρήκαμε σε δύο κύριες κατηγορίες. Στα συστήματα εκείνα που έχουν προέλθει από ερευνητικές δραστηριότητες (CoNet, Amaya, ComMentor, CritSuit, FutPlex κ.α.) και σε αυτά που βρήκαμε σαν εμπορικά πακέτα (ThirdVoice, JotBot κ.α.). Στη βιβλιογραφία και στο Internet βρήκαμε αρκετές αναφορές ακτός από τα παραπάνω και σε διάφορα άλλα Annotation tools για τα οποία όμως δεν μπορέσαμε να δούμε παρά μόνο μερικές γραμμές από σχόλια και απλά τα αναφέρουμε. Θελήσαμε να τα εξετάσουμε ως προς τις **λειτουργίες** που προσφέρουν, την **αρχιτεκτονική** και το **σχήμα** που περιγράφει την δομή της βάσης. Πλήρης περιγραφή και για τις τρείς συνιστώσες βρήκαμε μόνο για το ComMentor και εν μέρει για το Amaya και το CoNote. Για το λόγο αυτό τα περιγράφουμε πιο αναλυτικά από τα υπόλοιπα. Να σημειώσουμε ότι δεν μπορέσουμε να τρέξουμε όλα τα συστήματα που αναφέρουμε στον συγκριτικό πίνακα έτσι βασιστήκαμε σε αναφορές άλλων οι οποίες όμως περιορίζονται κυρίως στις λειτουργίες των συστημάτων και όχι στην αρχιτεκτονική και στο σχήμα.

CoNote

Δημιουργήθηκε στο Cornell University. Πρόκειται για ένα «καθαρό» annotation system το οποίο μάλιστα δοκιμάστηκε με θετικά αποτελέσματα στο τμήμα Επιστήμης Υπολογιστών του Cornell ενώ αξιολογήθηκε θετικά και από μία ομάδα της ευρωπαϊκής επιτροπής[IEP99]. Το URL που θα μας έδινε πρόσβαση στο CoNote [CONOTE] δεν λειτουργεί και έτσι δεν μπορέσαμε να το δούμε να δουλεύει..

Το CoNote απαιτεί από τον χρήστη να δώσει το σωστό login και password, στη συνέχεια να κάνει register στον server και από το σημείο αυτό και μετά μπορεί να κάνει σημειώσεις σε

κείμενα που τον ενδιαφέρουν καθώς και να βλέπει τις σημειώσεις άλλων στα κείμενα αυτά. Για να δημιουργήσει μία νέα σημείωση, ο χρήστης πρέπει να συμπληρώσει μία φόρμα η οποία έχει συγκεκριμένο format. Το πρώτο πεδίο έχει τον **τίτλο** της σημείωσης. Υπάρχει πεδίο για **keywords** έτσι ώστε ο υπόλοιπος κόσμος να μπορεί όταν ψάχνει με βάση τις λέξεις αυτές να βρίσκει το κείμενο μας ενώ το τρίτο κομμάτι της φόρμας αφορά την ίδια τη σημείωση η οποία μπορεί να είναι είτε σε μορφή ASCII είτε σε μορφή HTML. Οι σημειώσεις δεν εμφανίζονται οι ίδιες στο κείμενο αλλά εμφανίζονται links σε αυτές in-line στο κείμενο και ο χρήστης απλά πλοηγείται σε αυτές. Κάθε link δείχνει τον τίτλο, το όνομα εκείνου που το έκανε και την ημερομηνία που έγινε ο σχολιασμός. Όπως φαίνεται στο σχήμα παρακάτω οι σημειώσεις διατάσσονται σε δενδρική διάταξη δίνοντας τη δυνατότητα στ ους χρήστες να βλέπουν τις απαντήσεις στα σχόλια που κάνουν οι ίδιοι ή κάποιοι άλλοι με αποτέλεσμα να δημιουργείται η εντύπωση ότι γίνεται συζήτηση μεταξύ των χρηστών.



Σχήμα 1: ***Annotations στο CoNote***

Οι χρήστες έχουν δυνατότητα να κάνουν αναζήτηση στις σημειώσεις με βάση κάποιο χαρακτηριστικό όπως για παράδειγμα το χρόνο που έγινε η σημείωση, σε ποιο έγγραφο έγινε η σημείωση ποιος έκανε τη σημείωση κλπ. Στα πλαίσια του "document group" που ανήκει κάθε χρήστης του δίνεται η δυνατότητα να έχει κάποιους ρόλους (ο ρόλος παίζει τον ρόλο των permissions). Οι ρόλοι αυτοί είναι viewer, reader, user και author. Ο viewer δεν μπορεί να δεί ούτε να κάνει σημειώσεις πάνω στο έγγραφο. Ο reader μπορεί να διαβάσει σημειώσεις του εγγράφου αλλά όχι να κάνει σημειώσεις σ'αυτό. Ο user μπορεί να κάνει σημειώσεις στο έγγραφο όμως δεν μπορεί να σβήσει κάποια σημείωση από αυτό. Τέλος ο author μπορεί να κάνει όλες τις

λειτουργίες πάνω στο έγγραφο και στις σημειώσεις ακόμα και να σβήσει κάποια από αυτές. Κάθε μέλος του "document group" είναι δυνατό να έχει διαφορετικούς ρόλους σε κάποιο έγγραφο.

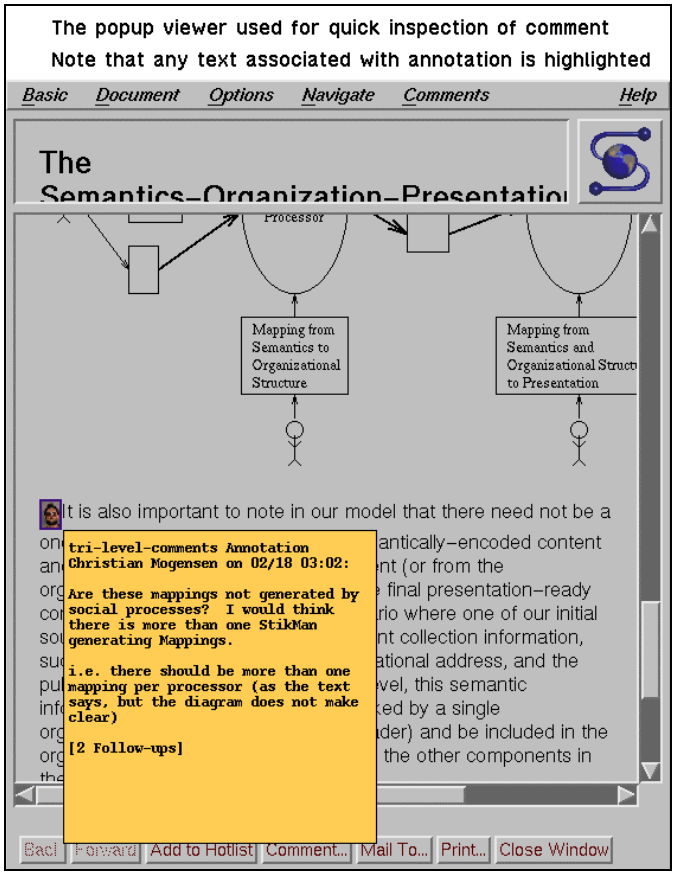
Όπως φαίνεται στο σχήμα 1 οι σημειώσεις εμφανίζονται in-line μέσα στο κείμενο. Αυτό βέβαια δε σημαίνει ότι αλλιώνουν την αρχική μορφή του. Οι σημειώσεις αποθηκεύονται σε ξεχωριστή βάση και απλά εμφανίζονται σαν να είναι κομμάτι του εγγράφου όταν ο χρήστης ζητήσει να διαβάσει το έγγραφο. Στο σχήμα 1 το κομμάτι του κειμένου που είναι σε κόκκινο πλάισιο είναι πραγματικό κείμενο ενώ στο μπλέ πλαίσιο είναι οι σημειώσεις.

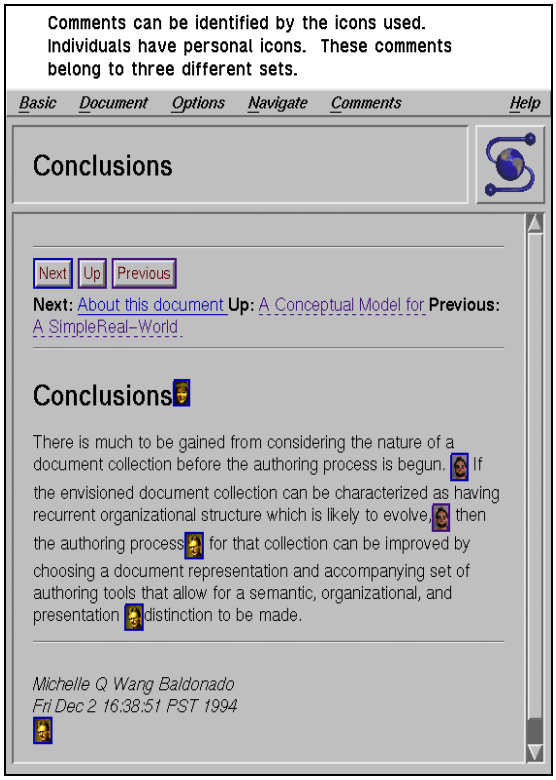
Για την αρχιτεκτονική του CoNote δεν βρήκαμε λεπτομέρειες. Ακολουθεί την ιδέα του caching Web proxy Server. Γίνεται επεξεργασία στα αυθεντικά έγγραφα ώστε να εισαχθεί το link στη σημείωση πριν εμφανιστεί στο χρήστη το κείμενο. Στη συνέχεια τα αρχεία που προκύπτουν από την επεξεργασία αυτή αποθηκεύονται για να είναι γρηγορότερη η αποστολή τους στον client εάν εκείνος ζητήσει να τα δεί ξανά. Όποτε εισαχθεί νέα σημείωση σε κάποιο κείμενο τότε ελέγχεται εάν έχει γίνει προσωρινά αποθηκευμένο (cached) και εαν είναι πράγματι έτσι τότε η προσωρινή αποθήκευση παύει να είναι έγκυρη. Στην βιβλιογραφία εμφανίζεται άλλη μία υλοποίηση του CoNote στην οποία δεν χρησιμοποιήται η προηγούμενη αρχιτεκτονική αλλά γίνεται χρήση του Jigsaw/SLK server με servlets η οποία έχει καλύτερη απόδοση από την αρχική και όπως αναφέρεται στα σχετικά άρθρα απλοποιεί την υλοποίηση.

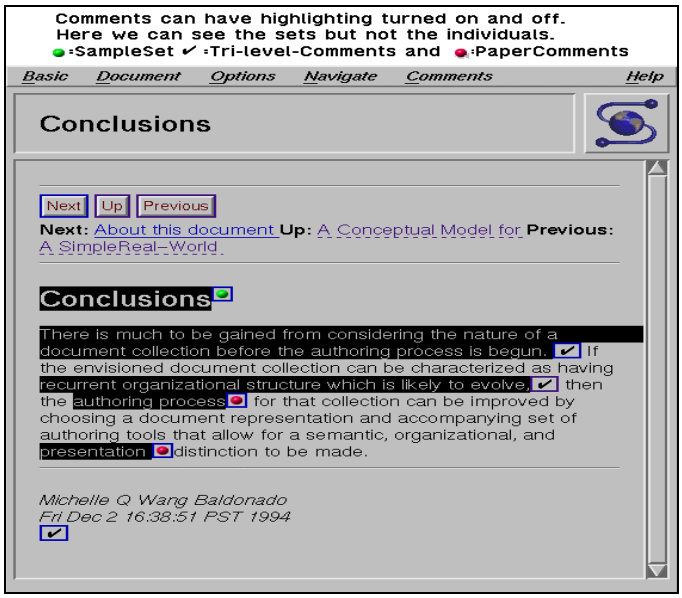
Για το CoNote μπορούμε να σχολιάσουμε ότι δεν προσφέρει δυνατότητα σημειώσεων σε άλλα μέσα παρά μόνο σε text.

ComMentor

Το ComMentor [RMW94] υλοποιήθηκε στο Πανεπιστήμιο Stanford. Στην ουσία πρόκειται για έναν www browser ο οποίος παρέχει στο χρήστη τη δυνατότητα να κάνει σημειώσεις σε HTML σελίδες. Οι σημειώσεις μπορούν να γίνουν σε οποιοδήποτε σημείο της HTML σελίδας. Το μέρος του κειμένου το οποίο έχει επιλεχτεί να σχολιαστεί έχει διαφορετικό χρώμα και υπάρχει ένα εικονίδιο στο τέλος του. Το εικονίδιο μπορεί να προεπιλεγεί από τον χρήστη και μάλιστα μπορεί να είναι ακόμη και μία μικρή φωτογραφία του (16x22 pixels). Για το εικονίδιο αυτό υπάρχει μία ακόμη λειτουργία. Όταν το μεσαίο πλήκτρο του mouse επιλέξει το εικονίδιο και μένει πατημένο σε αυτό τότε εμφανίζεται ένα μικρό παράθυρο (μοιάζει με PostIt παρόμοιο με εκείνο του MS Word) με το κείμενο της σημείωσης και εξαφανίζεται όταν το μεσαίο πλήκτρο απελευθερωθεί.







Στα διπλανά σχήματα βλέπουμε παραδείγματα χρήσης του συστήματος ComMentor.

Βλέπουμε τα εικονίδια με τις φωτογραφίες των συγγραφέων των σημειώσεων, η σύντομη παρουσίαση της σημείωσης σαν *"post-it"*

Για να εισάγει νέα σημείωση ο χρήστης μαρκάρει το κείμενο που θέλει και κάνει την επιλογή annotate. Όταν γίνει αυτό εμφανίζεται μία φόρμα που ζητά από τον χρήστη τον τίτλο και το

κείμενο της σημείωσης. Μπορεί επίσης να επιλέξει εάν η σημείωση είναι private ή public ή στα πλαίσια ενός group χρηστών. Χαρακτηριστικό του συστήματος είναι ακόμη ότι ο χρήστης μπορεί να κάνει σημείωση στις σημειώσεις όμως δεν μπορεί να καταλάβει ότι μία σημείωση έχει άλλες σημειώσεις παρά μόνο εάν την ανοίξει για ανάγνωση. Επίσης, σε μία σημείωση δεν επιτρέπεται να αλλάξει το περιεχόμενο της. Δηλαδή από τη στιγμή που γράφεται δεν μπορεί να γίνει ξανά edit παρά μόνο να γίνει νέα σημείωση που θα συμπληρώνει την προηγούμενη.

Για τη διαδικασία ανάκτησης των σημειώσεων μπορούμε να πούμε τα παρακάτω. Όταν ένας χρήστης θελήσει να δει μία σελίδα τότε ο browser στέλνει μία ερώτηση στον Annotation server ζητώντας να μάθει εάν για τη συγκεκριμένη σελίδα (URL) υπάρχουν σημείωσεις. Ο Annotation server στέλνει πίσω ένα string με πληροφορίες (meta-information) τις οποίες ο client χρησιμοποιεί για να επιστρέψει το έγγραφο με τις σημειώσεις ορατές πάνω σε αυτό με τον τρόπο που περιγράψαμε πριν (εικονίδια κ.λπ). Οι πληροφορίες αυτές προσδιορίζουν μοναδικά τη θέση της σημείωσης στο έγγραφο. Ακόμη ο χρήστης μπορεί να επιλέξει σημειώσεις προσδιορίζοντας το χρόνο που έχουν γίνει. Πχ μπορεί να επιλέξει τις σημειώσεις που έγιναν την τελευταία ώρα, ημέρα, μήνα κ.λπ.

Για να έχει δικαίωμα να διαβάσει τις σημειώσεις σε ένα έγγραφο, ο χρήστης πρέπει να ανήκει σε ένα group χρηστών με πρόσβαση σε αυτό. Η διαδικασία για να γίνει μέλος του group ο χρήστης θυμίζει εκείνη που απαιτείται για να γίνει μέλος μιας οποιασδήποτε mailing list.

FutPlex

Από το Eindhoven University of Technology το FutPlex [FUTPLEX] είναι ένα annotation system που επιτρέπει την εισαγωγή σημειώσεων σε οποιοδήποτε σημείο της HTML σελίδας. Οι σελίδες που σχολιάζονται δεν είναι public HTML σελίδες αλλά εισάγονται και διαγράφονται σε μία δενδρική μορφή. Το σύστημα δεν περιέχει μηχανισμούς που να επιτρέπουν στους χρήστες να σβήνουν σελίδες. Οι χρήστες μπορούν να σβήνουν μόνο links σε σελίδες.
Οι σημειώσεις μπορούν να γίνουν edited από εκείνον που τις έγραψε πρώτη φορά αλλά μπορούν να σβηστούν από οποιονδήποτε χρήστη.

Το σύστημα αυτό τρέχει σε UNIX και έχει υλοποιηθεί με AWK και C.

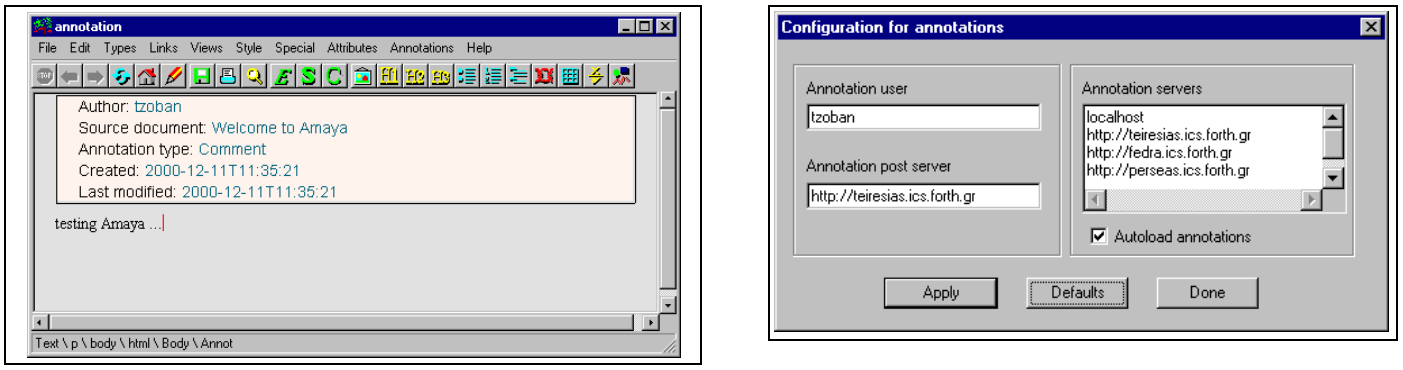
Amaya

Το Amaya [AMAYA] είναι στην ουσία ένας ολοκληρωμένος browser (όπως ο Internet Explorer, ο Netscape Navigator κ.α.) οποίος διαθέτει την λειτουργία Annotation στις διαθέσιμες λειτουργίες του. Στο Amaya οι σημειώσεις αποθηκεύονται τοπικά είτε σε έναν ή περισσότερους Annotation Servers. Όταν κάθε σελίδα έρχεται στον browser (Amaya) τότε εκείνος κάνει queries στους Annotation Servers που έχει δηλώσει ο χρήστης αναζητώντας τις σημειώσεις που σχετίζονται με τη σελίδα.

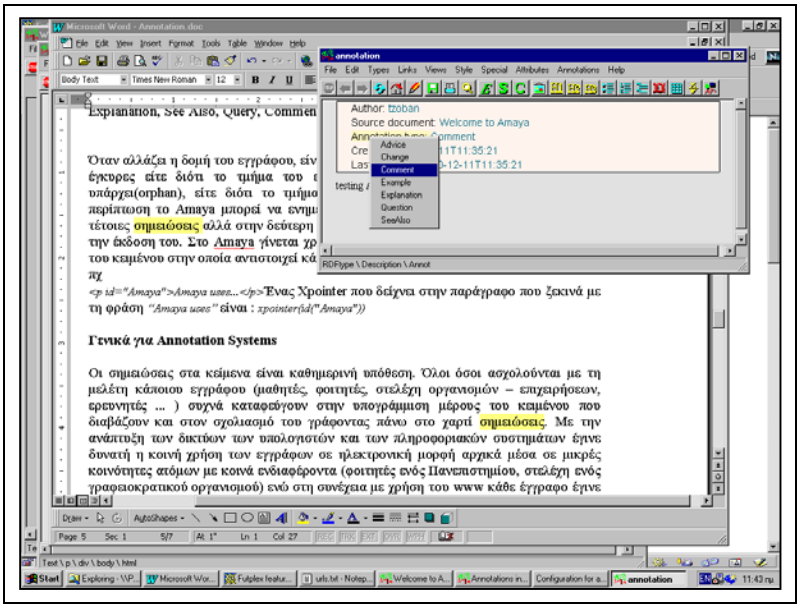
Οι σημειώσεις εμφανίζονται με ένα εικονίδιο που συμβολίζει ένα μολύβι να προηγείται από το κείμενο που έχει επιλεγεί για να σχολιαστεί. Κάθε χρήστης μπορεί να έχει τοπικά τις δικές του σημειώσεις χωρίς τη χρήση www server αλλά μπορεί να τις κάνει public απλά εγκαθιστώντας έναν server στο μηχάνημα του (στο site του Amaya προτείνονται μερικοί πχ Apache, AnalogX SimpleServer κ.α.) και ακολουθώντας τα βήματα στις οδηγίες του δίνονται.

Χαρακτηριστικό του Amaya είναι ότι οι Servers τους οποίους «ακούει» ο browser για να πάρει τις σημειώσεις, ορίζονται από τον χρήστη. Επίσης ο χρήστης ορίζει τον server στον οποίο οι σημειώσεις που κάνει ο ίδιος είναι διαθέσιμες στο κοινό (που θα τις βρούν οι άλλοι χρήστες).

Στο Amaya μπορεί να σχολιαστεί είτε μέρος του κειμένου είτε όλοκληρο το κείμενο. Για το σχολιασμό μέρους του κειμένου απλά επιλέγουμε με το mouse το κείμενο και στη συνέχεια με την επιλογή “Annotate Selection” κάνουμε το σχόλιο στο κείμενο που επιλέξαμε. Επίσης επιλέγοντας “Annotate Document” το σχόλιο (το εικονίδιο με το μολύβι) εμφανίζεται στην αρχή του κειμένου. Σε κάθε μία από τις δύο περιπτώσεις η φόρμα που πρέπει να συμπληρωθεί είναι κοινή και περιλαμβάνει δύο τμήματα. Πρώτα τα στοιχεία του σχολιαστή, το χρόνο, τον τύπο της σημείωσης κ.λπ. και άλλο ένα κομμάτι που αφορά το μέρος της σημείωσης (σχόλιο). Οι τύποι σημειώσεων που υποστηρίζει το Amaya είναι Annotation, Query, Comment κα. Όταν αλλάζει η δομή του εγγράφου, είναι πιθανό μερικές σημειώσεις να μην είναι πλέον “valid” είτε διότι το τμήμα του εγγράφου στο οποίο έδειχναν τώρα πια δεν υπάρχει(orphan), είτε διότι το τμήμα αυτό άλλαξε θέση(misleading). Στην πρώτη περίπτωση το Amaya μπορεί να ενημερώσει τον χρήστη ότι στο κείμενο βρίσκονται τέτοιες σημειώσεις αλλά στην δεύτερη περίπτωση δεν έχει τρόπο ενημέρωσης σε αυτήν την έκδοση του.



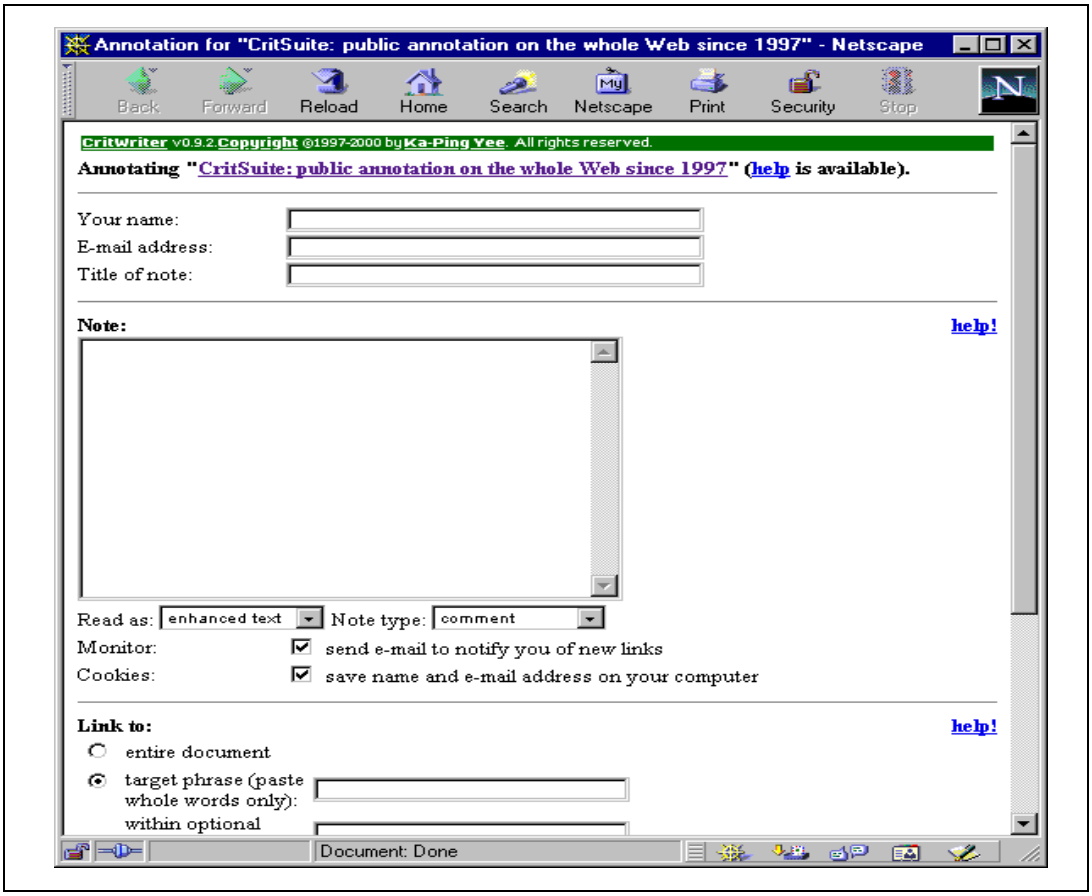
Σχήμα 2: ***editor για τη δημιουργία σημειώσεων στο Amaya και επιλογή των Annotation Servers από τον χρήστη.***



Σχήμα 3: ***Δημιουργία σημειώσεων στο Amaya.***

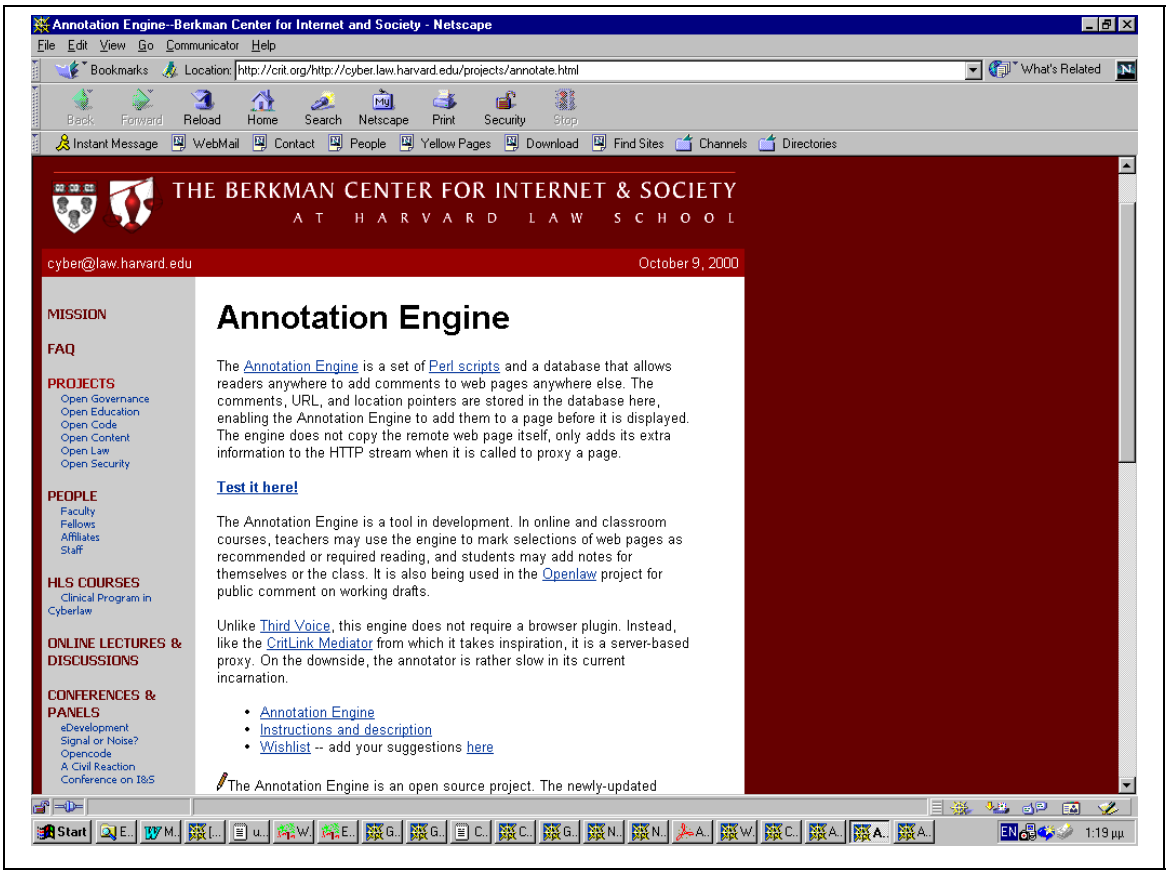
CritSuit

Στο CritSuit [CRIT] ο χρήστης αποκτά λογαριασμό και μπορεί να δημιουργήσει σημειώσεις σε ιστοσελίδες αφού προηγουμένως έχει συνδεθεί στον server που παρέχει τη λειτουργία υπομνηματισμού. Στο σύστημα αυτό οι σημειώσεις άναφέρονται σαν σύνδεσμοι προς HTML σελίδες στο τέλος του εγράφου. Η φόρμα που δημιουργεί τις σημειώσεις είναι εκείνη που φαίνεται στο παρακάτω σχήμα.



Σχήμα 4: ***Φόρμα δημιουργίας σημειώσεων στο CritSuit***

Το σύστημα έχει βρει εφαρμογή και στην νομική σχολή του Harvard University. Η αρχική σελίδα του συστήματος φαίνεται παρακάτω.



Σχήμα 5: ***Αρχική σελίδα του συστήματος υπομνηματισμού της νομικής σχολής του Πανεπιστημίου του Harvard***

BSCW

Το Διαδίκτυο έχει ορισμένα χαρακτηριστικά, τα οποία μπορούν να στηρίξουν συστήματα συνεργασίας για ανταλλαγή πληροφοριών.

- Οι browsers είναι διαθέσιμοι για όλες τις σημαντικές πλατφόρμες και παρέχουν πρόσβαση σε πληροφορίες ανεξάρτητα από το σύστημα στο οποίο ανήκουν.
- Οι browsers παρέχουν ένα απλό και σχετικά αμετάβλητο GUI σε κάθε έκδοση, διευκολύνοντας τους χρήστες
- Οι browsers επίσης αποτελούν μέρος του υπολογιστικού συστήματος στους περισσότερους οργανισμούς
- Οι περισσότεροι οργανισμοί έχουν εγκαταστήσει τους δικούς τους web servers και γνωρίζουν τον τρόπο για την συντήρηση τους.

Με τον όρο *shared workspace* εννοούμε έναν κοινό χώρο που χρησιμοποιείται για την αποθήκευση πληροφορίας, στον οποίο έχουν πρόσβαση άτομα ή ομάδες. Οι χρήστες του BSCW έχουν πρόσβαση σε τέτοιους κοινούς χώρους με χρήση των συνηθισμένων web – browsers χωρίς να είναι απαραίτητη η εγκατάσταση νέου λογισμικού. Το BSCW (Basic Support for Cooperative Work) είναι ένα εργαλείο CSCW (Computer Support for Cooperative Work) που αναπτύχθηκε στο ερευνητικό κέντρο GMD [GMD].

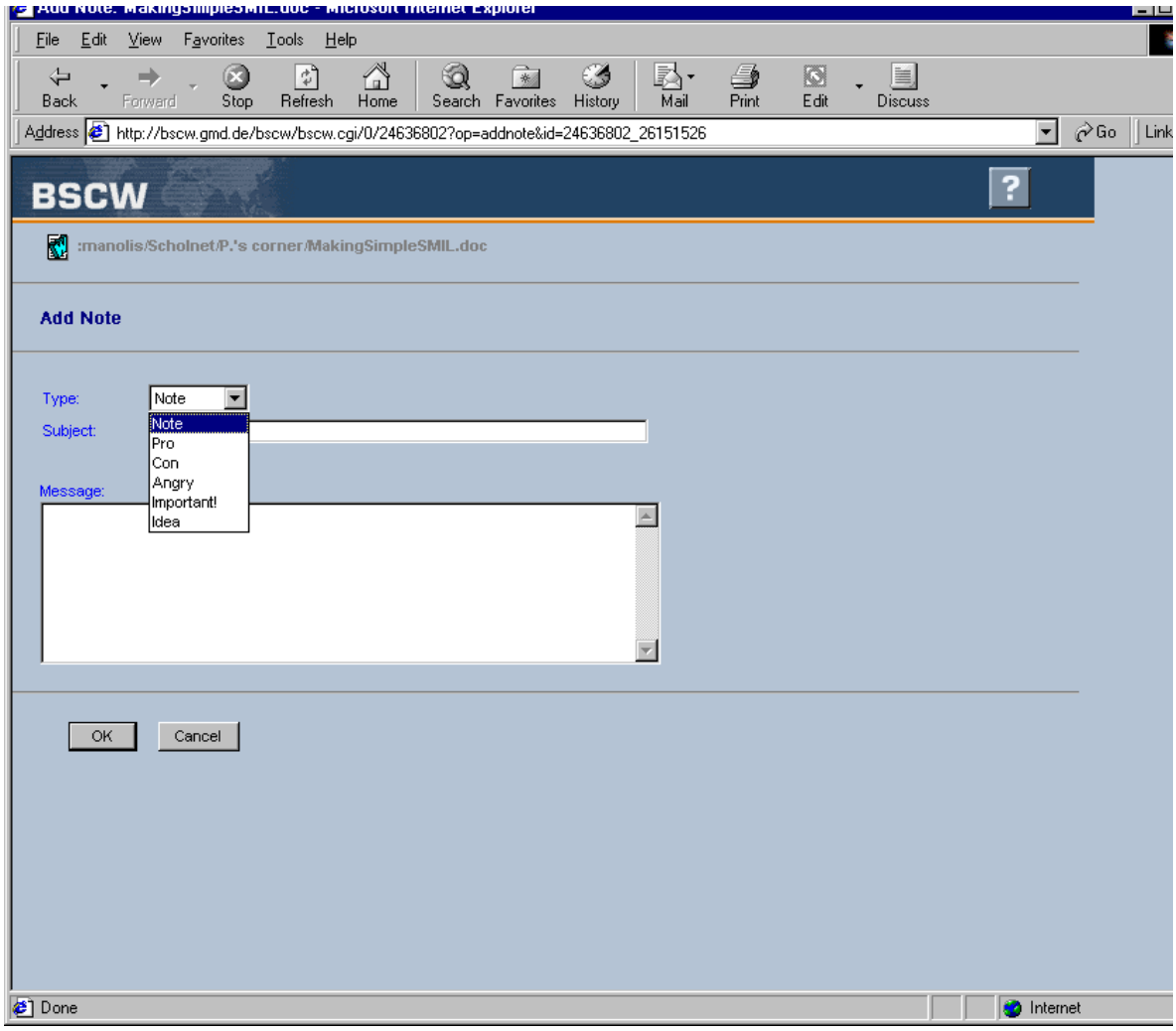
Ορισμένα από τα βασικά χαρακτηριστικά του συστήματος είναι τα εξής :

1. Διαχείριση εκδόσεων και «κλείδωμα» κατά τη διάρκεια των περιόδων εγγραφών
2. Επιβεβαίωση χρηστών
3. Λίστα συζητήσεων
4. Δικαιώματα πρόσβασης
5. Δυνατότητα αναζήτησης με βάση το όνομα, το περιεχόμενο ή συγκεκριμένη ιδιότητα κάποιου εγγράφου πχ την ημερομηνία μεταβολής του εγγράφου
6. Ταξινόμηση των εγγράφων με βάση τις απαιτήσεις των χρηστών πχ με βάση την ημερομηνία ή με βάση τον τύπο του εγγράφου.
7. Δημιουργία σημειώσεων και βαθμολόγηση εγγράφων σχετικά με την ποιότητα τους.
8. Προφίλ χρηστών – Beginner, advanced και Expert.
9. Υπηρεσία ηλεκτρονικού ταχυδρομίου για επικοινωνία μεταξύ των χρηστών του συστήματος.
10. Καλάθι αχρήστων, βιβλίο διευθύνσεων για καταχώρηση στοιχείων χρηστών, ημερολόγιο για σημαντικές ημερομηνίες. Αυτές οι λειτουργίες αφορούν προσωπική χρήση κάθε χρήστη.
11. Υποστήριξη περισσότερων από μία γλώσσα.

Υπάρχει ένας κεντρικός server για χρήση από απλούς χρήστες. Σε αυτόν υπάρχει δυνατότητα για δημιουργία ενός νέου shared workspace και τη φιλοξενία σε αυτό εγγράφων. Αν όμως κάποιος χρήστης επιθυμεί να έχει τον δικό του BSCW server μπορεί να τον "κατεβάσει" [BSCW2] από το Internet και να τον εγκαταστήσει στο μηχάνημα του. Ο server έχει υλοποιηθεί σε python και μπορεί να τρέξει σε Unix ή σε WindowsNT.

Ιδιαίτερα για την υπηρεσία υπομνηματισμού που προσφέρει το BSCW μπορούμε να σχολιάσουμε ότι είναι μία πολύ ευέλικτη εφαρμογή που υποστηρίζεται από οποιοδήποτε browser έχοντας πολύ ένχρηστη διεπιφάνεια χρήσης. Το BSCW προσφέρει στους χρήστες

του τη δυνατότητα δημιουργίας σημειώσεων καλύπτοντας τους παρακάτω τύπους Note, Angry, Important, idea, Pro, Con. Απλά συμπληρώνουμε στην παρακάτω φόρμα τον τύπο, το θέμα και το μύνημα του χρήστη στα κατάλληλα πεδία και επιλέγουμε "OK"



Σχήμα 6 ***Η φόρμα δημιουργίας σημειώσεων στο BSCW***

Σύστημα υπομνηματισμού εγγράφων Σαράντη Τούλη

Στην μεταπτυχιακή εργασία του Σαράντη Τούλη[Τούλης94] σχεδιάστηκε και υλοποιήθηκε ένα συστημα υπομνηματισμού εγγράφων το οποίο όπως και το δικό μας, είχε χρησιμοποιήσει ως σύστημα διαχείρισης το SIS. Εκείνο το μοντέλο των σημειώσεων προοριζόταν για να περιγράφει τις σημειώσεις που αντιστοιχούσαν σε άρθρα και βιβλία, η παράσταση των οποίων γινόταν σύμφωνα με το πρότυπο ISO 12983. Το σύστημα εκείνο αναπαριστούσε τις σημειώσεις με την SGML. Για την εισαγωγή των στοιχείων στη βάση είχαν χρησιμοποιηθεί τα «δελτία εισαγωγής στοιχείων» (data entry forms). Η χρήση του συστήματος όμως ήταν μόνο τοπική. Το σύστημα εκείνο παρείχε πολύ ικανοποιητικές κατηγορίες σημειώσεων. Υποστήριζε τύπους σημειώσεων που αφορούσαν μόνο ένα έγγραφο και την γνώμη τοιυ αναγνώστη για αυτό, καθώς και τύπους σημειώσεων που συσχέτιζαν περισσότερα από ένα έγγραφα.

Στο σύστημα μας το μοντέλο των σημειώσεων περιγράφεται με στοιχεία που ανήκουν στο σύνολο μεταδεδομένων Dublin Core. Το σύστημα μας λειτουργεί στο Διαδίκτυο κάνοντας χρήση της τεχνολογίας Java, ώστε να εξασφαλιστεί η ανεξαρτησία του από τις διαφορετικές πλατφόρμες που χρησιμοποιούν οι διαφορετικοί χρήστες. Χρησιμοποιούμε έναν web server για να εξυπηρετήσουμε τις διάφορες κλήσεις από remote clients. Επίσης κάνουμε χρήση του Java API για την κλήση των ρουτινών του C++ API από τους clients ώστε να επιτευχθεί η επικοινωνία με το SIS. Εξακολουθούμε να παρέχουμε τόσο τύπους σημειώσεων που αφορούν την γνώμη του αναγνωστη όσο και από εκείνη την κατηγορία των τύπων που συσχετίζουν έγγραφα μεταξύ τους. Οι τύποι, οι κατηγορίες και οι διαδικασίες που αφορούν τις σημειώσεις είναι κοινές για τις διαφορετικές ομάδες συνεργατών έτσι ώστε οι ομάδες των μελετητών να μπορούν να διευκολυνθούν στην-εργασία τους. Σε σχέση λοιπόν με το προηγούμενο σύστημα ο σχεδιασμός της εφαρμογής μας και η τεχνολογία που χρησιμοποιήσαμε μας επιτρέπει να συμπεριλάβουμε το σύστημα μας στα πλαίσια ενός συνόλου δικτυακών εφαρμογών οι οποίες χρησιμοποιούν κοινά πρωτόκολα επικοινωνίας επιτυγχάνοντας έτσι την προσαρμογή του συστήματος υπομνηματισμού ως μία υπηρεσία που παρέχεται από μία ψηφιακή βιβλιοθήκη μέσω του WWW. Με το σύστημα εκείνο ο χρήστης μπορούσε να κατατάξει ένα έγγραφο σε μία από τις εξής κατηγορίες:

1. is_annotated_by_classification, να το ταξινομίσει με βάση κάποιον όρο ενός ελεγχόμενου λεξιλογίου.
2. is_annotated_by_work_theme, να το σχολιάσει σε σχέση με μία εργασία του.
3. is_annotated_by_main_point, σημειώσεις που εκφράζουν τα κύρια σημεία ενός εγγράφου
4. is_annotated_by_existing_document, συσχέτιση με άλλο έγγραφο

Εμείς ορίσαμε και παρουσιάσαμε μία νέα, διαφορετική κατηγοριοποίηση των τύπων των σημειώσεων ενώ παρέχουμε λειτουργίες, που βοηθούν στην αναζήτηση σημειώσεων και εγγράφων. Βασική διαφορά από το προηγούμενο σύστημα αποτελεί η δυνατότητα για ύπαρξη πολλών «διαστάσεων» σε κάθε σημείωση. Αυτός είναι ένας λόγος για τον οποίο οι τύποι των σημειώσεων, που έχουμε ορίσει διαφοροποιούνται από εκείνους που είχαν οριστεί σε εκείνο το σύστημα. Στο σύστημα μας χωρίσαμε τους τύπους των σημειώσεων με βάση τις παρακάτω κατηγορίες

1. evaluates
2. criticizes
3. records
4. is related with

Η τελευταία κατηγορία μπορεί να θεωρήσουμε ότι αντιστοιχεί με την κατηγορία is_annotated_by_existing_document του προηγούμενου συστήματος. Οι υπόλοιπες τρεις κατηγορίες έχουν δημιουργηθεί για να κατηγοριοποιήσουν εννοιολογικά τους τύπους, στους οποίους εμείς καταλήξαμε.

Βασικό σημείο είναι το γεγονός ότι σε όλες τις κατηγορίες μας περιέχονται τύποι ελεγχόμενου λεξιλογίου. Όμως εμείς προτιμήσαμε να μην δημιουργήσουμε μία νέα κατηγορία σημειώσεων για να τους εντάξουμε (όπως συνέβει στο προηγούμενο σύστημα) αλλά να κατατάξουμε τους τύπους αυτούς με βάση την έννοια τους σε κάποια από τις 1, 2, 3 κατηγορίες. Μας ενδιαφέρει να εκφράσουμε περισσότερο το λόγο, που κάνει τη σημείωση ο σχολιαστής παρά τον τρόπο με τον οποίο την κάνει. Αυτό εξάλλου μας οδήγησε να ονομάσουμε τις κατηγορίες στις οποίες εντάξαμε τις σημειώσεις με τα ονόματα “criticises”, “evaluates”, “records” θεωρώντας ότι με αυτόν τον τρόπο

δηλώνουμε την πρόθεση του σχολιαστή να ασκήσει κριτική, να κάνει μια εκτίμηση ή να καταγράψει κάποιο ξεχωριστό σημείο του εγγράφου.

2.3 Αξιολόγηση των προηγουμένων συστημάτων

Στον παρακάτω πίνακα εμφανίζονται μερικά από τα συστήματα που μπορέσαμε να βρούμε στο διαδίκτυο και στη βιβλιογραφία.. Τα εξετάσαμε ως προς την αρχιτεκτονική, το μοντέλο, το σύστημα διαχείρισης δεδομένων που χρησιμοποιούν καθώς και τα queries που υποστηρίζουν. Επίσης αναφέρουμε κάποια επιπλέον χαρακτηριστικά τα οποία χαρακτηρίζουν κάθε ένα από τα συστήματα όπως πχ γλώσσα υλοποίησης .

Πίνακας 1 ***Συγκριτικός πίνακας συστημάτων υπομνηματισμού***

Εφαρμογή	Αρχιτεκτονική	Μοντέλο	Αποθήκευση	Τύποι επερωτήσεων	Τύποι σημειώσεων	Επιπλέον Χαρακτηριστικά
Amaya	Οι private σημειώσεις αποθηκεύονται τοπικά και οι public σε κατανεμημένους servers	Υλοποιημένο σε RDF	Δεν αναφέρεται DBMS	Δεν υποστηρίζει customized queries – υπάρχει πρόβλεψη για αργότερα	Advice, change, comment, example, explanation, question, see also	
CoNote			Δεν αναφέρεται DBMS	υποστηρίζει customized queries		Ύπαρξη user – groups, εμφάνιση σε δενδρική δομή και πλοήγηση με links. Χαρακτηριστικά εικονίδια για τις σημειώσεις που έχει γράψει ο author του εγγράφου
Futplex			Δεν αναφέρεται DBMS			
Critlink	Κατανεμημένη		Δεν αναφέρεται DBMS		Query, comment, support, issue	
Principia Cybernetica	Κεντρικοποιημένος server		Δεν αναφέρεται DBMS		Comment, refutation, confirmation, correction, illustration	

HyperNews			Δεν αναφέρεται DBMS	Δεν υποστηρίζει επερωτήσεις, απλά εμφανίζεται μία δενδρική διάταξη με τα σχόλια	Question, note, warning, feedback, idea, more, news, ok, sad, angry, agree, disagree	Υλοποίηση με java, ύπαρξη εικονιδίων
JotBot			Δεν υπάρχει DBMS παρά μόνο αρχεία		Δεν γίνεται αναφορά σε τύπους σημειώσεων	Χρήση HTML για την εμφάνιση των αποτελεσμάτων. Μελλοντική χρήση XML

Δυστυχώς δεν μπορέσαμε να βρούμε πληροφορίες για όλα τα συστήματα που αναφέρονται στον πίνακα.. Για όσα από αυτά βρήκαμε στοιχεία, παρατηρούμε από τον παραπάνω πίνακα ότι :

- Τα περισσότερα από αυτά δεν κάνουν χρήση κάποιου DBMS.
- Χρησιμοποιούν λίγους τύπους σημειώσεων οι οποίοι δεν καλύπτουν πλήρως τις απαιτήσεις των χρηστών.
- Οι τύποι των σημειώσεων που έχουν δημιουργηθεί αφορούν αυστηρά μόνο ένα έγγραφο. Δηλαδή δεν υπάρχουν τύποι σημειώσεων οι οποίοι να συσχετίζουν δύο ή περισσότερα έγγραφα μεταξύ τους..
- Πολλά από αυτά τα συστήματα δεν υποστηρίζουν τη δημιουργία σημειώσεων σε άλλες σημείωσεις. Βέβαια μερικά συστήματα έχουν τέτοια δυνατότητα και έτσι δημιουργούνται δίκτυα από σημειώσεις.
- Ορισμένα κάνουν σημειώσεις μόνο σε ολόκληρο το έγγραφο ενώ δεν υποστηρίζουν σημειώσεις σε μέρη του εγγράφου.
- Μόνο ένα από αυτά κάνει χρήση της RDF για την περιγραφή των σημειώσεων.
- Ελάχιστα συστήματα κάνουν χρήση της Java στην υλοποίηση τους, με αποτέλεσμα να υπάρχει πρόβλημα για χρήστες που χρησιμοποιούν διαφορετικές πλατφόρμες.

3. Η πρόταση μας

3.1. Ανάγκες - Περιβάλλον

3.1.1 Υπομνηματισμός στις Ψηφιακές Βιβλιοθήκες

Σε μια εποχή, που η εξάπλωση των δικτύων έχει επιτρέψει την πλοήγηση στο www ακόμη και σε απομακρυσμένες περιοχές γίνεται ολοένα και πιο φανερή η ανάγκη για πρόσβαση σε ψηφιακές βιβλιοθήκες έτσι ώστε να προαχθεί η ενημέρωση και η διακίνηση της γνώσης σε όσους το επιθυμούν. Η απόσταση μεταξύ του ενδαφερόμενου και της πληροφορίας φαίνεται ότι έχει πάψει πλέον να αποτελεί το σοβαρότερο εμπόδιο στην απόκτηση της και πλέον η προσοχή έχει στραφεί σε νέες κατευθύνσεις όπως για παράδειγμα η αλληλεπίδραση του ενδιαφερομένου με την πληροφορία καθώς και η όσο το δυνατό καλύτερη απεικόνιση της πληροφορίας στον ενδιαφερόμενο.

Η αλληλεπίδραση του ενδιαφερόμενου χρήστη με τις υπάρχουσες πληροφορίες στις ψηφιακές βιβλιοθήκες πρέπει να αποβλέπει στην εξαγωγή νέας χρήσιμης πληροφορίας. Μία τέτοιου είδους αλληλεπίδραση χρήστη – ψηφιακής βιβλιοθήκης είναι ο σχολιασμός – υπομνηματισμός στα ηλεκτρονικά έγγραφα που την αποτελούν. Ο αναγνώστης ενός κειμένου έχοντας μελετήσει ένα έγγραφο είναι σε θέση να εξάγει συμπεράσματα και σκέψεις οι οποίες θα τον βοηθήσουν σε μελλοντικές ενέργειες του. Η καταγραφή αυτών των συμπερασμάτων μπορεί, εάν γίνει σε μία οργανωμένη βάση δεδομένων με δυνατότητες διαχείρισης (δημιουργίας, ανάκτησης, διαγραφής, ενημέρωσης σημειώσεων) να επιτρέψει τα εξής :

- τη δημιουργία νέας ανεξάρτητης βάσης δεδομένων με πληροφορίες χρήσιμες στον κάθε αναγνώστη.
- τις επιπλέον συσχετίσεις των ηλεκτρονικών εγγράφων της ψηφιακής βιβλιοθήκης τις οποίες δημιούργησε ο αναγνώστης όπως φανερώνονται από τις σημειώσεις.
- τη θεώρηση των σημειώσεων ως έγγραφα και την παραπέρα συσχέτιση μεταξύ τους καθώς και τη συσχέτιση μεταξύ αυτών και των εγγράφων της ψηφιακής βιβλιοθήκης.

Ζητήματα που πρέπει να συζητηθούν παραπέρα είναι :

a. η διαφορετικότητα των χρηστών και η μετάδοση της πληροφορίας με κατάλληλο τρόπο σε αυτούς.
b. η εύρεση κοινά αποδεκτών χαρακτηριστικών για την περιγραφή των σημειώσεων.
c. η δημιουργία τύπων σημειώσεων ικανών να εκφράσουν τις σχέσεις μεταξύ των εγγράφων όπως εμείς θέλουμε.
d. η πρόταση μιας αρχιτεκτονικής η οποία να μπορεί να συνδέσει το σύστημα υπομνηματισμού με τις υπόλοιπες υπηρεσίες της ψηφιακής βιβλιοθήκης.
e. ποια μέσα πρόκειται να χρησιμοποιήσουμε για να επιτύχουμε μια εφαρμογή όσο το δυνατόν πιο συμβατή με τα ευρέως χρησιμοποιούμενα τεχνολογικά standards παρουσιάζοντας όμως ταυτόχρονα μία πρωτότυπη εργασία.

3.1.2 Παράσταση πληροφορίας με XML

Υπάρχουν πολλοί τρόποι με τους οποίους μπορούμε να αναπαραστήσουμε πληροφορία. Σχήματα όπως XML[XML1], RDF [Brickley] μπορούν να χρησιμοποιηθούν παρέχοντας κατάλληλες δομές (attributes, elements) ώστε να γίνει δυνατή η παράσταση της πληροφορίας. Δηλαδή τα σχήματα αυτά παρέχουν ένα σύνολο από κανόνες – οδηγίες – περιορισμούς για τον σχεδιασμό συγκεκριμένου format αρχείων επιτρέποντας τη διακίνηση τους μέσα στο δίκτυο σε διαφορετικές πλατφόρμες επιλύοντας ζητήματα επεκτασιμότητας όπως επίσης internationalization/localization κ.α.

Στην εργασία μας επιλέξαμε την XML για την περιγραφή των σημειώσεων. Όπως η HTML έτσι και η XML κάνει χρήση των tags και των attributes [XML2]. Όμως ενώ η HTML προσδιορίζει τη χρήση κάθενος από τα tags και τα attributes που χρησιμοποιούνται στα HTML αρχεία, η XML με τα tags και τα attributes απλά προσδιορίζει – διαχωρίζει τα τμήματα της πληροφορίας και αφήνει την λειτουργία κάθενος από τα τμήματα στην εφαρμογή η οποία ασχολείται με την ανάγνωση του XML αρχείου. Για παράδειγμα το tag <p> της HTML σημαίνει κάτι διαφορετικό σε αρχείο τύπου XML απ'ότι σημαίνει στα HTML αρχεία. Μπορούμε δηλαδή να παρουσιάσουμε σε XML αρχείο την πληροφορία που βρίσκεται αποθηκευμένη στη βάση μας και να ονομάσουμε κάθε tag όπως εμείς κρίνουμε σκόπιμο ανάλογα με την έννοια του περιεχομένου του.

Τα αρχεία τύπου XML είναι αρχεία κειμένου τα οποία όμως συνήθως δεν προορίζονται για να διαβαστούν από ανθρώπους αλλά να γίνουν αντικείμενο επεξεργασίας από ειδικά προγράμματα. Σκοπός των προγραμμάτων αυτών είναι να κάνουν parsing τα tags των XML αρχείων και να απομονώσουν τις τιμές των πεδίων καθώς και των γνωρισμάτων τους (attributes). Με τον τρόπο αυτό μπορούμε να διακινήσουμε την πληροφορία η οποία βρίσκεται στη βάση μας με τρόπο ανεξάρτητο από τη φύση του client που ζητά αυτήν την πληροφορία αρκεί να έχει εγκατεστημένο το πρόγραμμα που θα του επιτρέψει να απομονώσει τις τιμές των πεδίων και των γνωρισμάτων τους ώστε να προχωρήσει στην επεξεργασία τους.

3.1.3 Dublin Core Metadata για την περιγραφή των σημειώσεων

Γενικά για τα μετα – δεδομένα

Με τον όρο **Metadata (**μετα – δεδομένα**)** εννοούμε δεδομένα σχετικά με άλλα δεδομένα *"Data about data"*. Ειδικότερα μπορούμε να πούμε ότι μετα – δεδομένα αποτελούν όλες οι αναγνωρίσιμες από τους υπολογιστές πληροφορίες οι οποίες σχετίζονται με άλλες πληροφορίες. Στην περίπτωση μας αναφερόμαστε σε πληροφορίες η οποίες μπορούν να γίνουν αντικείμενο επεξεργασίας από μηχανές και τελικά να μας κάνουν πιο εύκολα τη δουλειά μας. Μπορούμε δηλαδή να σχεδιάσουμε κάποιο πρόγραμμα το οποίο να αξιοποιεί τις πληροφορίες για τις πληροφορίες που μας ενδιαφέρουν.

Ένα παράδειγμα είναι το εξής. Με χρήση του HTTP ανακτούμε ένα object από το www, τότε με το πρωτόκολλο αυτό μας παίρνουμε πληροφορίες σχετικά με τον *author*, την *ημερομηνία δημιουργίας* του αντικειμένου κ.α. Έτσι το web γίνεται μια δεξαμενή πληροφοριών πολλές από τις οποίες αποτελούν metadata καθώς αφορούν άλλες πληροφορίες. Ασφαλώς τα μέτα – δεδομένα αποτελούν κι εκείνα πληροφορίες και για το λόγο αυτό μπορούμε να τα διαχειριστούμε με παρόμοιο τρόπο που διαχειριζόμαστε κάθε είδους πληροφορία. Αυτό σημαίνει ότι μπορούμε να τα αποθηκεύσουμε σε μία βάση δεδομένων και με την ύπαρξη του κατάλληλου συστήματος διαχείρισης βάσεως δεδομένων να κάνουμε Queries αποσκοπώντας σε αποτελέσματα που θα διευκολύνουν τη δουλειά μας. Ώστε τα μέτα – δεδομένα για μία πληροφορία μπορούν να αποθηκεύονται σε διαφορετικό χώρο από τις πληροφορίες στις οποίες αναφέρονται. Τα μέτα – δεδομένα αποτελούν ισχυρισμούς σχετικά με τα δεδομένα και

μπορούν να παρουσιαστούν σαν σύνολο από ανεξάρτητα δεδομένα που υπακούουν σε συγκεκριμένους κανόνες σχετικά με τον τρόπο αποθήκευσης τους και τον τρόπο παρουσίασης τους στο χρήστη .

Dublin Core Metadata

Το σύνολο μεταδεδομένων Dublin Core [DC4] είναι ένα σύνολο από ιδιότητες και χαρακτηριστικά που περιγράφουν οποιοδήποτε είδος πληροφορίας. Αυτό το σύνολο των ιδιοτήτων βοηθά στην οργάνωση και κατάταξη της πληροφορίας ανεξάρτητα από το γνωστικό αντικείμενο της. Αποτελείται από 15 ορισμούς [DC1] που οργανώνουν και περιγράφουν εννοιολογικά οποιαδήποτε πληροφορία ανεξάρτητα από τον τύπο της και την κατηγορία στην οποία ανήκει. Όλοι μπορούν να χρησιμοποιήσουν Dublin Core metadata για να περιγράψουν τις **resources**[1] σε ένα οποιοδήποτε πληροφοριακό σύστημα. Ιδιαίτερη χρήση βρίσκουμε στις web σελίδες με τη βοήθεια των HTML tags.
Τα Dublin Core metadata χρησιμοποιούνται ως βάση για συστήματα που περιγράφουν πληροφορίες σε διάφορες κοινότητες όπως

- Ερευνητές
- Εκπαιδευτικούς οργανισμούς
- Βιβλιοθήκες
- Σχεδιαστές web σελίδων
- Εμπορικούς – γραφειοκρατικούς οργανισμούς

Πρόκειται δηλαδή για ένα ευρέως αποδεκτό σύνολο μεταδεδομένων στο οποίο μπορούμε να βασιστούμε για την περιγραφή των σημειώσεων. Ώστε με χρήση των DC μεταδεδομένων δημιουργούμε ένα record το οποίο περιγράφει επαρκώς την πληροφορία των σημειώσεων. Δηλαδή κάθε ένα από τα πεδία του annotation record αντιστοιχεί σε κάποιο από τα DC μεταδεδομένα.. Πρόκειται λοιπόν για μία ευρέως αποδεκτή αντιστοιχία καθώς βασίζεται στα Dublin Core μεταδεδομένα. Στον επόμενο πίνακα φαίνεται η αντιστοιχία αυτή.

[1] Λέγοντας **resource** όταν αναφερόμαστε για το Web, εννοούμε οτιδήποτε στο οποίο μπορεί να αποδωθεί ένα URL.

Πίνακας 2*: Αντιστοιχία Dublin Core Metadata Record – Annotation Record*

Dublin Core	Annotation Record
Creator	Author
Subject	Subject
Description	The Text of each Annotation Type
Date.Modified	Date
Type	Annotation type: "text"
Format	Το DTD που θα οριστεί για τις σημειώσεις
Identifier	Identifier – annotation name
Relation.references	Program, Links.toValue
Rights	Group, Project

3.1.4 Επεξήγηση της αντιστοιχίας των πεδίων DC record – annotation record

Παρακάτω εμφανίζουμε τα πεδία του πεδίου των σημειώσεων καθώς και μία επεξήγηση για κάθε ένα από αυτά.

- **Author:** ο συγγραφέας της σημείωσης. Στο μοντέλο των σημειώσεων υπάρχει κλάση που προσδιορίζει τον συγγραφέα της σημείωσης. Το πεδίο αυτό του Annotation Record αντιστοιχεί στο element "*Creator"* των Dublin Core Metadata. Πρόκειται για μία οντότητα που είναι υπεύθυνη για την δημιουργία του περιεχομένου της σημείωσης.
- **Subject:** Το θέμα του περιεχομένου της σημείωσης. Συνήθως εκφράζεται σαν keywords, key phrases ή όρους που περιγράφουν το θέμα της σημείωσης. Ο χρήστης μπορεί να ορίσει δικά του *Subjects.*
- **The text of each Annotation type:** Πρόκειται για το κομμάτι της σημείωσης που περιέχει το σχόλιο του συγγραφέα σχετικά με το έγγραφο που σημειώνει. Μπορεί να περιλαμβάνει απλό κείμενο-σχόλιο. Βέβαια έχουμε ορίσει κάποιους τύπους σημειώσεων οι οποίοι δεν υποστηρίζουν κάποιο κείμενο.
- **Date:** Η ημερομηνία δημιουργίας της σημείωσης η οποία φυσικά δεν μεταβαλλεται και παραμένει η ίδια για όλη τη διάρκεια ζωής της σημείωσης. Το format που προτείνουμε να ακολουθείται είναι YYYY-MM-DD.
- **Annotation Type:** Η φύση του περιεχομένου της σημείωσης. Υποστηρίζουμε μόνο σημειώσεις σε μορφή text για τα κείμενα. Το σύστημα μας δεν υποστηρίζει εικόνες, ήχο ή άλλου τύπου σημειώσεις παρά μόνο απλό κείμενο.

- **Format:** Με το Format προσδιορίζεται το software ή οποιοσδήποτε άλλος εξοπλισμός απαιτείται για την αναπαράσταση της σημείωσης. Στην περίπτωση μας το Format ισοδυναμεί με το DTD που έχει οριστεί για τα XML αρχεία που περιγράφουν τις σημειώσεις.
- **Identifier:** Προσδιορίζει μοναδικά τη σημείωση. Ο προσδιορισμός της σημείωσης γίνεται με το όνομα του κόμβου με το οποίο έχει καταχωρηθεί στο SIS.
- **Program:** Τα προγράμματα στα οποία ανήκει η σημείωση. Όσοι ασχολούνται με αυτά τα προγράμματα έχουν πρόσβαση στη σημείωση.
- **Group:** Τα user groups τα οποία έχουν πρόσβαση στη σημείωση. Όσοι χρήστες ανήκουν στα groups έχουν πρόσβαση στη σημείωση.
- **Links.toValue:** Ο κόμβος στον οποίο καταλήγει κάποιος από τους συνδέσμους του annotation Node οι οποίοι αφορούν συσχέτιση με άλλα έγγραφα ή σημειώσεις.

3.2. Εννοιολογική Σχεδίαση

3.2.1 Γενικά

Σκοπός μας είναι η δημιουργία ενός συστήματος διαχείρισης σημειώσεων στα ηλεκτρονικά έγγραφα μιας ψηφιακής βιβλιοθήκης. Οι λειτουργίες που πρέπει να παρέχουμε αφορούν την **αποθήκευση σημειώσεων**, την **ανάκτηση σημειώσεων με βάση κριτήρια** τα οποία θέτει ο χρήστης καθώς και τη **διαγραφή σημειώσεων**. Η αποθήκευση των σημειώσεων γίνεται σε μία οντοκεντρική βάση δεδομένων στην οποία έχουμε δημιουργήσει ένα μοντέλο που περιγράφει τις σημειώσεις. Το μοντέλο έχει γραφεί με γλώσσα TELOS. Η ανάκτηση των σημειώσεων γίνεται με queries πάνω σε αυτό το μοντέλο. Τα queries γράφονται σε γλώσσα Java και κάθε απομακρυσμένος client είναι σε θέση να τα καλέσει μέσω του διαδικτύου. Τα queries στη βάση είναι προκαθορισμένα και όχι δυναμικά. Αυτό σημαίνει ότι έχουν καθοριστεί οι ανάγκες των χρηστών και έχουμε γράψει τα queries ώστε να καλύψουμε αυτές τις ανάγκες. Προγραμματιστικά τα queries είναι γραμμένα σε Java κώδικα μέσα από τον οποίο καλούνται C++ ρουτίνες ενός API με το οποίο μπορούμε να κάνουμε queries στο μοντέλο των σημειώσεων.

Στο μοντέλο σημειώσεων στο οποίο έχουμε δημιουργήσει περιγράφονται δύο βασικές κατηγορίες σημειώσεων. Η πρώτη κατηγορία είναι εκείνη η οποία συνδέει την σημείωση απεύθειας με το έγγραφο στο οποίο αναφέρεται. Θεωρήσαμε όμως επίσης ότι πρέπει να δημιουργήσουμε τύπους σημειώσεων οι οποίοι να συσχετίζουν τα έγγραφα

μεταξύ τους και όχι απλά να δημιουργούμε σχέσεις μεταξύ των εγγράφων και της προσωπικής γνώμης των αναγνωστών τους (σημείωση) όπως συνέβαινε σε προηγούμενα συστήματα υπομνηματισμού. Στο μοντέλο επομένως έχουμε δημιουργήσει σχέσεις που συνδέουν τους κόμβους που αναπαριστούν τα έγγραφα. Όμως υπάρχουν ορισμένα είδη συνδέσμων οι οποίοι συνδέουν τις σχέσεις μεταξύ των κόμβων που αναπαριστούν τα έγγραφα με κάποια σημείωση ενός συγγραφέα. Έτσι μπορέσαμε να αναπαριστήσουμε τη σχέση των εγγράφων μεταξύ τους όπως υποστηρίζει ο αναγνώστης τους ότι ισχύει και να την αντιστοιχίσουμε με κάποια σημείωση.

3.2.2 Οι τύποι των σημειώσεων στα έγγραφα

Το κύριο μέσο για αποθήκευση των σημειώσεων εξακολουθεί φυσικά να παραμένει το χαρτί. Σε αυτό το μέσο ο αναγνώστης μπορεί να καταγράφει τις παρατηρήσεις του, τις απορίες ή τις απόψεις του για εκείνα που αναφέρονται στο έγγραφο που μελετά. Εμείς σαν σχεδιαστές του συστήματος υπομνηματισμού έπρεπε να πάρουμε αποφάσεις σχετικά με τους τύπους των σημειώσεων που θα υποστήριζε το σύστημα μας ώστε να καλύψει όλες τις ανάγκες κάθε κατηγορίας χρηστών. Μελετώντας τις ανάγκες των μελλοντικών χρηστών του συστήματος καταλήξαμε στο συμπέρασμα ότι αυτοί αναζητούν ένα σύστημα υπομνηματισμού το οποίο να τους παρέχει τη δυνατότητα να δημιουργήσουν και να αντιστοιχήσουν σημειώσεις άμεσα στα ηλεκτρονικά έγγραφα που μελετούν όπως θα έκαναν εάν σημείωναν πάνω στο χαρτί. Εμείς βέβαια αξιοποιήσαμε το γεγονός ότι το μέσο στο οποίο καταγράφουμε τις σημειώσεις παρέχει πολλές άλλες δυνατότητες εκτός από μια απλή καταγραφή. Δημιουργήσαμε λοιπόν μία δεύτερη κατηγορία σημειώσεων η οποία συχετίζει τα έγγραφα μεταξύ τους με βάση εκείνα που αναφέρουν στο περιεχόμενο τους το ένα για το άλλο και όχι απλά με βάση την προσωπική γνώμη του αναγνώστη.

Στην πρώτη κατηγορία ομαδοποιούνται οι σημειώσεις εκείνες που αφορούν την άμεση συσχέτιση της προσωπικής γνώμης του αναγνώστη – συγγραφέα της σημείωσης με κάποιο έγγραφο της ψηφιακής βιβλιοθήκης. Στη δεύτερη κατηγορία κατατάσσονται οι σημειώσεις εκείνες που συσχετίζουν έγγραφα μεταξύ τους. Σε αυτήν την κατηγορία οι σημειώσεις περιέχουν την γνώμη του συγγραφέα της σημείωσης η οποία αφορά όμως τη σχέση μεταξύ των εγγράφων και δεν αφορά τη γνώμη του για ένα μόνο έγγραφο. Ο συγγραφέας δηλαδή αφού έχει πρώτα μελετήσει τα έγγραφα στα οποία αναφέρεται και έχει καταλήξει στη συσχέτιση μεταξύ των εγγράφων αυτών, τότε επιλέγει να

δημιουργήσει κάποια σημείωση η οποία να φανερώνει ακριβώς αυτή τη συσχέτιση. Οι σημειώσεις της δεύτερης κατηγορίας δεν εμφανίζονται σε κάποιο από τα άλλα συστήματα που είχαμε μελετήσει.

Θα ξεκινήσουμε με τους τύπους της πρώτης κατηγορίας, δηλαδή εκείνους τους τύπους των σημειώσεων οι οποίοι αφορούν μόνο ένα έγγραφο και προσδιορίζουν τη γνώμη του αναγνώστη γι' αυτό. Εννοιολογικά οι τύποι αυτοί μπορούν να ταξινομηθούν σε τρεις κατηγορίες. Ορίσαμε τις κατηγορίες *Evaluates, Criticizes, Records* για να κατατάξουμε όλους τους τύπους σημειώσεων που ορίσαμε.

Πίνακας 3 : ***Συγκεντρωτικός Πίνακας της πρώτης κατηγορίας των σημειώσεων***

Evaluates	Criticizes	Records
Needs	Agrees	Has document nature
Remark	Query	Has document subject
Writing	Explains	Has figure
Clarity	Summarize	Has table
Suitable Reader	Inspire	Excerpt
Value	Observe	Genre
Originality	Is Interested	
Contribute		
Opinion		

Παρακάτω θα εξηγήσουμε τί εκφράζει κάθε σημείωση που δημιουργούμε με χρήση καθενός από τους παραπάνω τύπους.

1. **Needs** : Με αυτόν τον τύπο σημείωσης ο χρήστης καλείται να επιλέξει μεταξύ μερικών επιλογών που προσδιορίζουν τί ελλείψεις έχει το έγγραφο. Τα προβλήματα που πιθανά έχει το έγγραφο αφορούν τυχόν διορθώσεις – επιλογή needs correction ή προσθήκες επιπλέον πληροφορίας για να συμπληρωθεί το περιεχόμενο ώστε να καλύψει κενά που υπάρχουν σε αυτό – επιλογή needs more information .
2. **Remark :** Με αυτόν τον τύπο σημείωσης ο χρήστης καταγράφει κάποια επισήμανση – σχόλιο που έχει να κάνει σχετικά με το έγγραφο. Ο συγκεκριμένος τύπος συναντάται επίσης σε άλλα συστήματα υπομνηματισμού.

3. **Writing :** Με αυτόν τον τύπο σημείωσης ο χρήστης καταγράφει τη γνώμη του για το έγγραφο βλέποντας το από την σκοπιά του περιεχομένου και του τρόπου γραφής. Εδώ δηλαδή κρίνουμε εάν η ανάλυση του προβλήματος και η ανάπτυξη του θέματος έχει γίνει ικανοποιητικά. Ο χρήστης έχει τις παρακάτω επιλογές.
    a. *Adequate analysis of the problem*, εάν το πρόβλημα αναλύεται επαρκώς
    b. *Background information adequately presented*, εάν οι «προαπαιτούμενες» πληροφορίες που αφορούν το πρόβλημα έχουν παρουσιαστεί επαρκώς.
    c. *There are enough evidence to support the conclusions*, εάν τα επιχειρήματα που προβάλονται στο έγγραφο είναι αρκετά πειστικά.
    d. *Provoke discussion*, εάν το έγγραφο είναι σε θέση να προκαλέσει συχητήσεις – σχόλια για το περιχόμενο του.
    e. *Logically arranged and organized*, εάν το έγγραφο είναι οργανωμένο με μία «σωστή» λογική και είναι εύκολο για τον αναγνώστη να το παρακολουθήσει.
4. **Clarity :** Με αυτόν τον τύπο σημείωσης ο χρήστης καταγράφει τη γνώμη του σχετικά με την σαφήνεια του εγγράφου. Πιθανές επιλογές είναι :
    a. *The paper is clearly written*, όταν το έγγραφο έχει απόλυτη σαφήνεια και το νόημα είναι ξεκάθαρο.
    b. *There should be additional illustrations to clarify the meaning*, εάν ο χρήστης θεωρεί ότι χρειάζεται επιπλέον γραφικές αναπαραστάσεις πληροφορίας για να καταγράψει πληροφορία που δεν είναι φανερή.
5. **Suitable Reader :** Με αυτόν τον τύπο σημείωσης ο χρήστης καταγράφει τη γνώμη του σχετικά με το επίπεδο γνώσεων που απαιτείται να έχει κάποιος που θέλει να μελετήσει το έγγραφο. Πιθανές επιλογές είναι οι παρακάτω :
    a. *Expert*, για χρήστες με βαθειά γνώση στο αντικείμενο που διαπραγματεύεται το έγγραφο.
    b. *Novice*, για χρήστες βασικές γνώσεις σχετικά με το αντικείμενο του εγγράφου.
    c. *Amateur*, για χρήστες χωρίς ιδιαίτερη γνώση για το αντικείμενο. Δηλαδή για όσους ασχολούνται πρώτη φορά με αυτό.
6. **Value :** Με αυτόν τον τύπο σημείωσης ο χρήστης καταγράφει τη γνώμη του σχετικά με την αξία του εγγράφου στο χρόνο. Το ζήτημα εδώ είναι εάν το έγγραφο παρέχει πληροφορίες διαχρονικής αξίας σχετικά με βασικές αρχές μιας

τεχνολογίας ή μιας επιστήμης ή απλά περιορισμένης αξίας πληροφορίες σχετικά με ένα προσωρινό πρόβλημα. Πιθανές επιλογές είναι οι παρακάτω : *permanent value*, *Limited value* αντίστοιχα.

7. **Originality :** Με αυτόν τον τύπο σημείωσης ο χρήστης καταγράφει το γεγονός ότι κατά τη γνώμη του ο συγγραφέας του εγγράφου διαπραγματεύεται ένα θέμα το οποίο είναι πρωτότυπο. Η τιμή που μπορεί να δώσει ο χρήστης είναι *Authentic*
8. **Contribute :** Με αυτόν τον τύπο σημείωσης ο χρήστης καταγράφει την προσωπική του συνεισφορά στο θέμα που αναφέρεται στο έγγραφο. Υπάρχει δηλαδή κάποιο πεδίο το οποίο ο χρήστης του συστήματος συμπληρώνει και το κείμενο που γράφει θεωρείται ως η συνεισφορά του στο θέμα.
9. **Opinion :** Με αυτόν τον τύπο σημείωσης ο χρήστης καταγράφει την προσωπική του γνώμη για το έγγραφο συνολικά. Εδώ περιμένουμε κριτική ώς προς το εάν αποδεχόμαστε ή όχι το έγγραφο ή σε πόσο μεγάλο βαθμό γίνεται αποδεκτό αυτό κ.λπ. Τιμές που αποδίδουμε στο έγγραφο είναι *Outstanding, Excellent, Very Good, Accepted, Not Accepted, Meaningless*.
10. **Agrees :** Με αυτόν τον τύπο σημείωσης ο χρήστης καταγράφει την συμφωνία του ή την διαφωνία του με το έγγραφο. Έχουμε δημιουργήσει διάφορα επίπεδα συμφωνίας.
    - ***a.*** *Strong Agreement,* δεν επιδέχεται αμφισβήτηση η γνώμη που εκφράζεται στο έγγραφο.
    - **b.** *Moderate Agreement,* δεχόμαστε τη γνώμη του εγγράφου έχοντας πειστεί από τις αποδείξεις που αναφέρονται σε αυτό.
    - **c.** *Weak Agreement,* δεν διαφωνούμε με τη γνώμη που εκφράζεται στο έγγραφο αλλά δεν μας έχει πείσει απόλυτα ο συγγραφέας με ισχυρά επιχειρήματα για την ορθότητα της γνώμης του.
    - **d.** *Disagreement*, πλήρης διαφωνία με τη γνώμη που εκφράζεται στο έγγραφο.

11. **Query :** Με αυτόν τον τύπο σημείωσης ο χρήστης καταγράφει ένα ερώτημα που του προξένησε κάτι που καταγράφεται στο έγγραφο.
12. **Explains :** Με αυτόν τον τύπο σημείωσης ο χρήστης καταγράφει μία παρατήρηση – επεξήγηση για κάτι που αναφέρεται στο έγγραφο. Ο τύπος αυτός είναι αντίστοιχος με μια απλή σημείωση στο περιθώριο ενός βιβλίου η οποία να εξηγεί κάποια παράγραφο – φράση που υπάρχει στο κείμενο.

13. **Summarize :** Με αυτόν τον τύπο σημείωσης ο χρήστης καταγράφει μία περίληψη ενός εγγράφου ή μέρους του εγγράφου.
14. **Inspire :** Με αυτόν τον τύπο σημείωσης ο χρήστης καταγράφει μία νέα ιδέα που του δημιουργήθηκε όταν μελέτησε το έγγραφο ή μέρος του εγγράφου ώστε να αποτελέσει την λύση σε κάποιο πρόβλημα του.
15. **Observe :** Με αυτόν τον τύπο σημείωσης ο χρήστης καταγράφει μία παρατήρηση – συμπληρωματική αναφορά σε σχέση με το περιεχόμενο του εγγράφου ή του μέρους του εγγράφου που μελετά.
16. **Is interesting :** Με αυτόν τον τύπο σημείωσης ο χρήστης δηλώνει πόσο ενδιαφέρον παρουσιάζει το έγγραφο που μελέτησε. Διακρίνουμε μία κλίμακα τεσσάρων επιπέδων. *Strong Interest, Moderate Interest, Weak Interest, No Interest.*
17. **Document nature :** Με αυτόν τον τύπο σημείωσης ο χρήστης επιλέγει την φύση του εγγράφου μεταξύ των εξής επιλογών που προσφέρουμε : *Θεωρητικό, σχεδιαστικό, τεχνικό* κλπ
18. **Document subject :** Με αυτόν τον τύπο σημείωσης ο χρήστης επιλέγει το θέμα του εγγράφου.
19. **Genre :** Με αυτόν τον τύπο σημείωσης ο χρήστης επιλέγει τον τύπο – είδος του εγγράφου μεταξύ των εξής επιλογών που προσφέρουμε : *review, paper, survey, MSc, PhD* κ.λπ.
20. **Table :** Ο χρήστης δηλώνει την παρουσία ενός πίνακα σε συγκεκριμένο σημείο του εγγράφου.
21. **Figure :** Ο χρήστης δηλώνει την παρουσία ενός γραφήματος (εικόνας, γράφου, γραφικής παράστασης κ.λπ.) σε συγκεκριμένο σημείο του εγγράφου.
22. **Excerpt :** Τμήμα του εγγράφου π.χ. κάποιο πλαίσιο κειμένου ή μία εξίσωση.

3.2.3 Το μοντέλο των σημειώσεων σε ηλεκτρονικά έγγραφα

Καταλήξαμε στο σχεδιασμό του δικού μας μοντέλου με βάση το γεγονός ότι υποστηρίζουμε δύο κατηγορίες σημειώσεων όπως προαναφέραμε. Βασικό χαρακτηριστικό του μοντέλου μας είναι το γεγονός ότι επιτρέπει τη δημιουργία σημειώσεων σε μέρη του εγγράφου και όχι μόνο σημειώσεις που αφορούν ολόκληρο το έγγραφο. Αυτό ήταν ένα χαρακτηριστικό το οποίο δεν εμφανίζεται σε πολλά από τα προηγούμενα συστήματα που εξετασάμε. Επιπλέον

στοιχείο που δεν εμφανιζόταν σε πολλά από τα συστήματα υπομνηματισμού που εξετάσαμε είναι η δυνατότητα να αντιμετωπίζουμε την σημείωση ως ένα νέο ξεχωριστό έγγραφο. Αυτό βέβαια μας δίνει την ευκαιρία να δημιουργήσουμε σημειώσεις πάνω σε άλλες σημειώσεις, να δημιουργήσουμε queries τα οποία συσχετίζουν τα χαρακτηριστικά των σημειώσεων κ.α.

Όπως ήδη έχουμε αναφέρει η περιγραφή του μοντέλου των σημειώσεων έγινε με χρήση της γλώσσας TELOS. Κάναμε χρήση τριών επιπέδων αφαίρεσης στην αναπαράσταση των σημειώσεων στο μοντέλο μας.

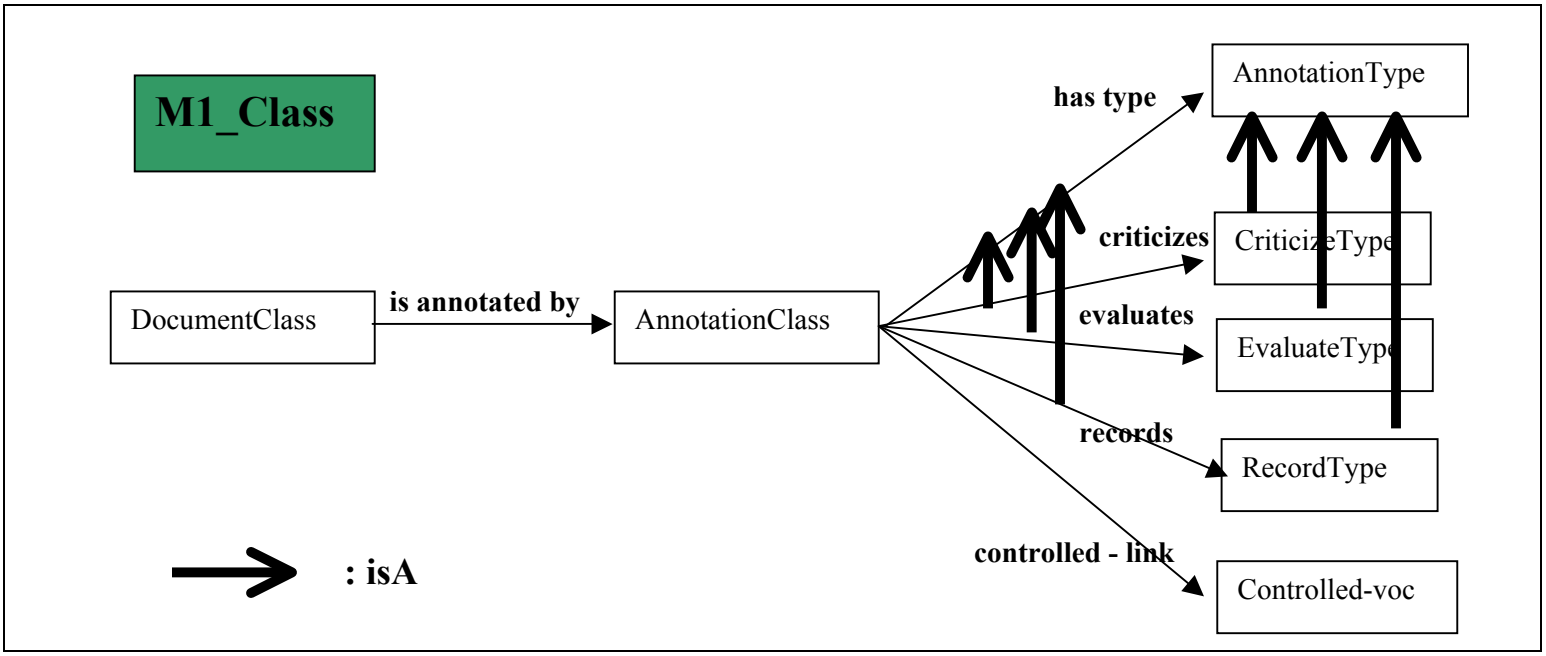
Στο επίπεδο M1_Class διακρίνουμε τις εξής κλάσεις.

1. DocumentClass
2. AnnotationClass
3. CriticizeType
4. EvaluateType
5. RecordType
6. AnnotationType

Η **DocumentClass** κλάση περιλαμβάνει όλα τα αντικείμενα στα οποία μπορούμε να αντιστοιχίσουμε μία σημείωση. Η κλάση **AnnotationClass** συνοψίζει τις διαστάσεις – χαρακτηριστικά μίας σημείωσης. Έχουμε ορίσει τρεις μετα-categories γνωρισμάτων, τις *criticizes, evaluates, records* οι οποίες καταλήγουν αντίστοιχα σε τρεις meta-categories **CriticizeType**, **EvaluateType**, **RecordType** που συμβολίζουν τις τρεις κατηγορίες – τύπους των σημειωσεων όπως τις παρουσιάσαμε σε προηγούμενο σημείο. Οι τρεις αυτές μετα – κατηγορίες είναι υποκλάσεις της κλάσης AnnotationType.

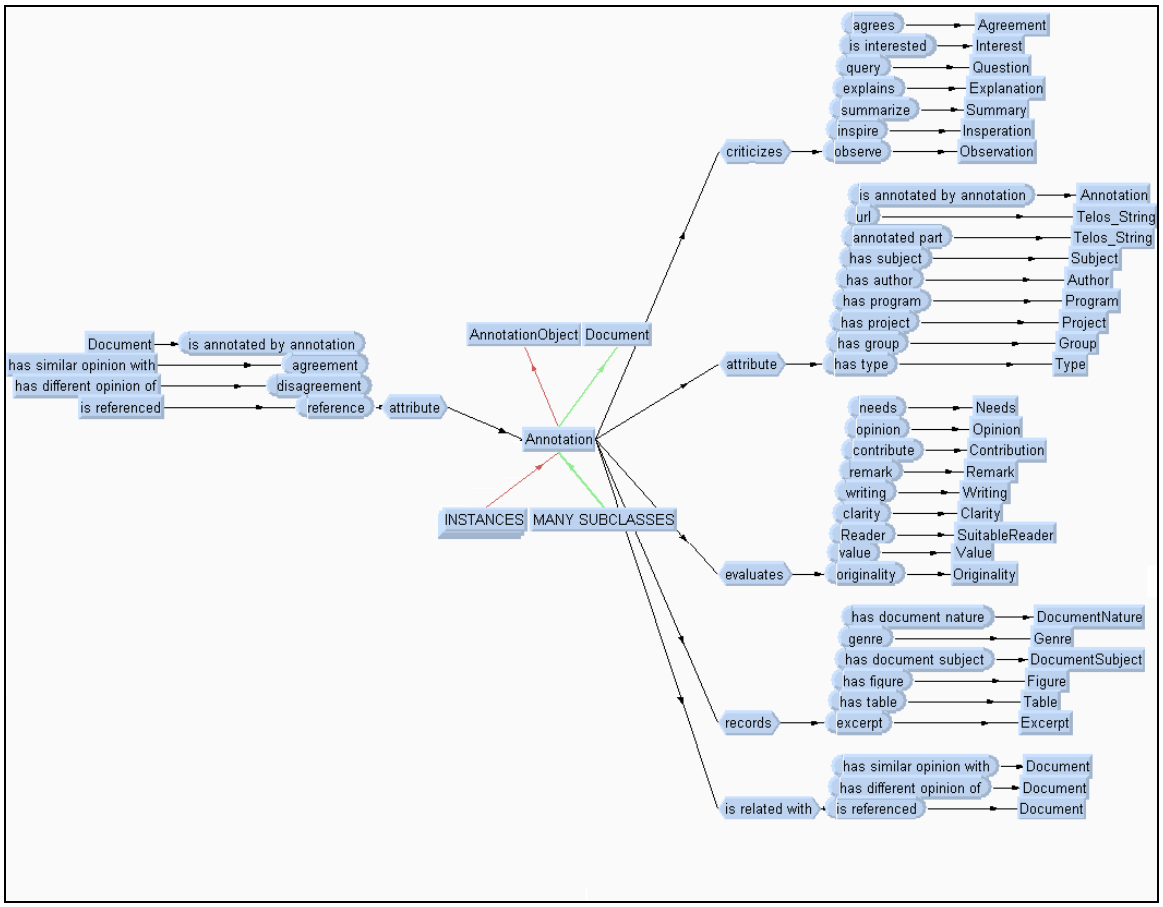
Στο ίδιο επίπεδο αφαίρεσης έχουμε ορίσει την κλάση **controlled-voc** καθώς και την κλάση γνωρισμάτων controlled-link. Όπως αναφέραμε σε προηγούμενη παράγραφο, ο χρήστης του συστήματος μας μπορεί να επιλέξει μια τιμή ανάμεσα σε ένα πλήθος προκαθορισμένων τιμών. Ο σκοπός της δημιουργίας αυτής της κλάσης είναι να ομαδοποιήσει όσους τύπους σημειώσεων επιβάλλουν στον χρήστη την επιλογή μεταξύ προκαθορισμένων τιμών για τη σημείωση. Η κλάση γνωρισμάτων controlled-link

συνοψίζει όσα links ξεκινούν από την κλάση Annotation στο S_Class και καταλήγουν σε κάποιο από τους τύπους σημειώσεων που είναι instances της κλάσης controlled-Voc.



Σχήμα 7 : M1_Class

Τα instances της κλάσης controlled-link στο S_Class επίπεδο είναι τα εξής: *agrees, is interested, needs, opinion, writing, clarity, reader, value, originality, has document nature, genre.* Αντίστοιχα τα instances της κλάσης controlled-voc στα οποία αυτά καταλήγουν είναι τα εξής: *Agreement, Interest, Needs, Opinion, Writing, Clarity, SuitableReader, Value, Originality, DocumentNature, Genre.* Κάθε ένας από αυτούς τους τύπους ανήκει εκτός από την κλάση controlled-voc σε κάποια άλλη από τις κλάσεις C*riticizeType, EvaluateType, RecordType* στο M1_Class.



Σχήμα 8 : *S_Class*

Όλες αυτές οι κλάσεις είναι instances σε κάποια από τις κλάσεις **CriticizeType**, **EvaluateType**, **RecordType** του M1 επιπέδου ανάλογα με τον τύπο της. Ο διαχωρισμός των διαστάσεων της σημείωσης μπορεί να γίνει όπως φαίνεται από τα links τα οποία ξεκινάνε από τον κόμβο Annotation και άποτελούν instances κάποιας από τις meta-categories γνωρισμάτων του M1 επιπέδου.

Κάθε ένας από τους τύπους σημειώσεων στο S_Class επίπεδο, αποτελεί όπως είπαμε instance μίας από τις κλάσεις **CriticizeType**, **EvaluateType**, **RecordType** του M1 επιπέδου. Ορισμένοι από τους τύπους σημειώσεων που έχουμε δημιουργήσει είναι απαριθμητοί. Αυτό σημαίνει ότι προτείνονται στον χρήστη ορισμένες προκαθορισμένες επιλογές και εκείνος μπορεί να επιλέξει κάποια ανάμεσα από αυτές σαν «τιμή» στην σημείωση. Ασφαλώς υπάρχουν αμοιβαία αποκλειόμενες τιμές ο συνδυασμός των οποίων δεν μπορεί να γίνει αποδεκτός από το σύστημα. Για να εξασφαλίσουμε ότι ο χρήστης δεν

πρόκειται να επιλέξει κάποιο μη επιτρεπτό συνδυασμό τιμών κάνουμε χρήση μιας βοηθητικής κλάσης στο S_Class επίπεδο και μιας βοηθητικής κλάσης στο M1_Class επίπεδο (πρόκειται για τις κλάσεις **FreeTextCategory** και **controlled-Voc**) οι οποίες ομαδοποιούν τους τύπους των σημειώσεων με τον εξής τρόπο:

FreeTextCategory

Στην κατηγορία αυτή ανήκουν οι τύποι εκείνοι των σημειώσεων οι οποίοι απαιτούν από το χρήστη να πληκτρολογήσει μια τιμή σε κάποιο πεδίο (στο *Graphical User Interface*) ως τιμή της σημείωσης ώστε αυτή να καταχωρηθεί στη βάση. Οι κλάσεις που δεν ανήκουν σε αυτή την κατηγορία έχουν χαρακτηριστικό ότι προτείνουν στους χρήστες να επιλέξουν κάποιες τιμές οι οποίες έχουν ήδη οριστεί από εμάς και δεν τους δίνουν τη δυνατότητα να πληκτρολογήσουν εκείνοι κάποια συγκεκριμένη τιμή. Η δημιουργία αυτής της κλάσης οφείλεται στην ανάγκη να γνωρίζει η υπηρεσία, που δημιουργεί το Graphical User Interface τις κλάσεις εκείνες οι οποίες απαιτούν τη συμπλήρωση από μέρους του χρήστη κάποιου πεδίου Text Field με δυναμικό τρόπο ώστε να τους παρέχει ένα τέτοιο πεδίο. Η γνώση αυτή προκύπτει με την εκτέλεση ενός Query στη βάση και τη δημιουργία κατάλληλων tags στο XML έγγραφο το οποίο μεταβιβάζεται στην υπηρεσία Graphical User Interface ώστε να γνωρίζει εκείνη σε ποιες κατηγορίες να προσθέσει επιπλέον πεδία και σε ποιες απλά να δημιουργήσει λίστα από ήδη ορισμένες επιλογές. Οι τύποι που ανήκουν σε αυτή την υποκλάση είναι οι *Remark, Question, Summary, Inspiration, Explanation, Observation και Contribution*.

controlled-Voc

Είναι η κλάση εκείνη στην οποία ανήκουν οι τύποι των σημειώσεων, οι οποίοι ζητούν από τους χρήστες να επιλέξουν κάποια από τις προκαθορισμένες τιμές, οι οποίες βρίσκονται αποθηκευμένες στη βάση. Όταν κατασκευάζεται η φόρμα, με την οποία ο χρήστης δημιουργεί τη σημείωση, οι τιμές αυτές ανακαλούνται, από τη βάση και τοποθετούνται μέσα σε κατάλληλα tags ώστε να ομαδοποιούνται. Το σύστημα επιτρέπει στον χρήστη να επιλέξει μόνο μία από τις τιμές που προτείνονται σε κάθε ομάδα.

Στην κατηγορία αυτή ανήκουν όσες κλάσεις δεν ανήκουν στην προηγούμενη κατηγορία. Παράδειγμα XML εγγράφου που κάνει χρήση αυτής της κατηγορίας των σημειώσεων είναι το παρακάτω

```
<value class="Opinion">
  <controlled-voc>Outstanding</controlled-voc>
  <controlled-voc>Excellent</controlled-voc>
  <controlled-voc>Very Good</controlled-voc>
  <controlled-voc>Accepted</controlled-voc>
  <controlled-voc>Not Accepted</controlled-voc>
  <controlled-voc>Meaningless</controlled-voc>
</value>
```

Στο κομμάτι αυτό του XML εγγράφου εμφανίζονται οι προκαθορισμένες επιλογές για την υποκλάση *Opinion* οι οποίες γράφονται μέσα στα tags *</controlled-voc>* . Το User Interface είναι υπεύθυνο

α) για την εμφάνιση στον χρήστη των επιλογών αυτών όταν εκείνος ζητήσει να κάνει μία σημείωση τύπου *Opinion.*

β) να εξασφαλίσει ότι ο χρήστης θα επιλέξει ακριβώς μία από τις επιλογές που του προτείνονται. Βέβαια ο τρόπος που ομαδοποιεί τις σημειώσεις το GUI γίνεται μετά από επικοινωνία με το SIS με κατάλληλο Query.

3.2.5 Integrity Constraints

Χαρακτηριστικό του συστήματος μας είναι η δυνατότητα που δίνει στον χρήστη να επιλέγει μία επιθυμητή τιμή για κάποιους τύπους σημειώσεων μέσα από ένα πλήθος προκαθορισμένων τιμών. Αυτή η επιλογή μεταξύ προκαθορισμένων τιμών διευκολύνει την δημιουργία σημειώσεων αλλά χρειάζεται προσοχή στην επιλογή των τιμών που πρόκειται να αποθηκευτούν στη βάση για κάθε σημείωση ώστε οι σημειώσεις που δημιουργούνται να έχουν νόημα.

Βασική προϋπόθεση για τη σωστή χρήση αυτών των τύπων των σημειώσεων είναι η επιλογή μιας μοναδικής τιμής για κάθε σημείωση που δημιουργείται. Αυτό σημαίνει ότι εάν σε κάποιο σημείο του εγγράφου δημιουργήσουμε μια σημείωση και αντιστοιχίσουμε σε αυτήν μία συμφωνία με τιμή "strong agreement" τότε *στην ίδια σημείωση* δεν μπορούμε να αντιστοιχίσουμε μία συμφωνία με διαφορετική τιμή πχ "weak agreement".

Ένα ακόμη πρόβλημα είναι ότι στο ίδιο σημείο ενός εγγράφου είναι πιθανή η δημιουργία *διαφορετικών σημειώσεων στις οποίες να αντιστοιχούν αντικρουόμενοι τύποι σημειώσεων από τον ίδιο χρήστη*. Αυτή η πιθανότητα πρέπει να αξαλειφθεί. Αυτό σημαίνει ότι σε κάποιο τμήμα ενός εγγράφου δεν είναι σωστό να υπάρχουν δύο διαφορετικές σημειώσεις που να έχουν αντικρουόμενη σημασία. Σαν παράδειγμα αναφέρουμε τον τύπο «συμφωνία». Προφανώς είναι λάθος να υπάρχει στο ίδιο σημείο ενός εγγράφου μία συμφωνία με τιμή "strong agreement" και ταυτόχρονα μία άλλη συμφωνία από τον ίδιο χρήστη με τιμή "weak agreement". Για να φροντίσουμε την εξάλειψη αυτών των καταστάσεων δημιουργήσαμε έναν αριθμό από βοηθητικές κλάσεις που εξυπηρετούν ακριβώς αυτό το σκοπό. Παρακάτω περιγράφουμε πως λειτουργεί αυτό το κομμάτι της εφαρμογής μας.

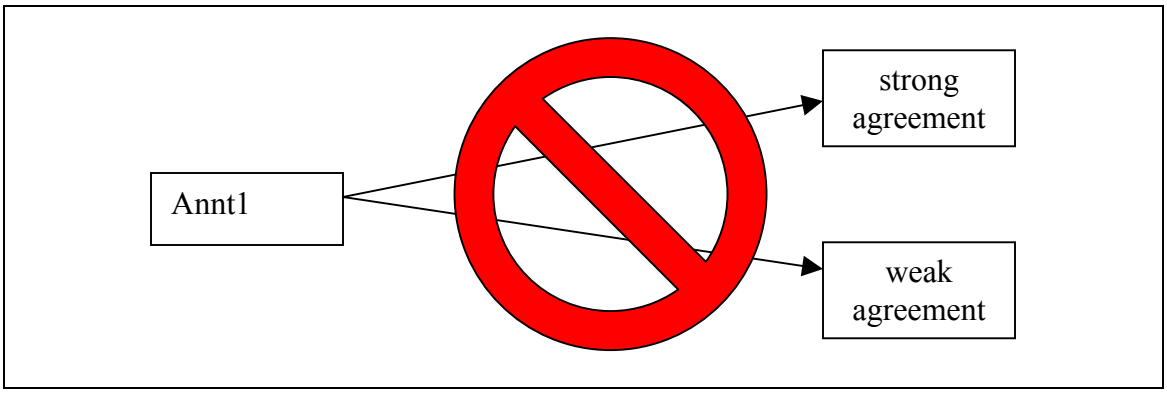
Τα instances της κλάσης controlled-link στο S_Class επίπεδο είναι τα εξής: *agrees, is interested, needs, opinion, writing, clarity, reader, value, originality, has document nature, genre.* Αντίστοιχα τα instances της κλάσης controlled-voc στα οποία αυτά καταλήγουν είναι τα εξής: *Agreement, Interest, Needs, Opinion, Writing, Clarity, SuitableReader, Value, Originality, DocumentNature, Genre.* Ο τρόπος για να εξασφαλίσουμε κάθε φορά που δημιουργείται μία σημείωση ότι τηρούνται οι

περιορισμοί που έχουμε θέσει είναι με τους περιορισμούς που θέτει το user Interface. Παρακάτω περιγράφουμε με χρήση λογικής τους περιορισμούς ακεραιότητας.

1ος περιορισμός

Στην ίδια σημείωση επιτρέπεται να υπάρχει μόνο μία τιμή για κάθε έναν τύπο σημειώσεων από αυτούς που είναι instance της κλάσης controlled-voc. Πχ δεν επιτρέπεται η ύπαρξη του τύπου συμφωνία με τιμή "strong agreement" και ταυτόχρονα "weak agreement" για την ίδια σημείωση.

For all u in AnnotationObject!controlled-link p in u ^q in u ^ a in Annotation
from(a) = a ^ from(q) = a → to(q) = to(p)

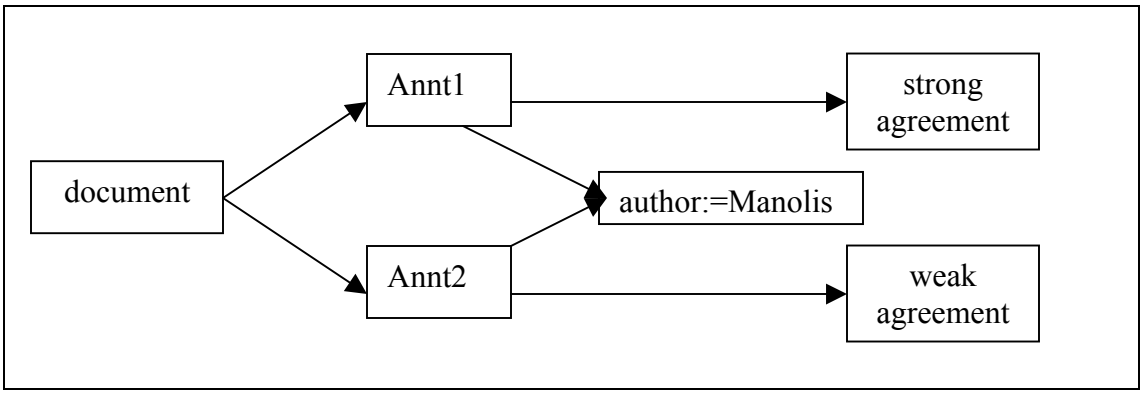


Σχήμα 9 ***1ος Περιορισμός ακεραιότητας***

Παρατήρηση

Μία σκέψη που είχαμε αρχικά ήταν να μην επιτρέψουμε στο ίδιο σημείο ενός εγγράφου να υπάρχουν *διαφορετικές σημείωσεις* από τον ίδιο author στις οποίες να αντιστοιχούν διαφορετικές τιμές ενός τύπου – instance της controlled–voc. Πχ να μην επιτρέπεται σε ένα σημείο του εγγράφου να υπάρχει ο χαρακτηρισμός "very interesting" και ταυτόχρονα για το ϊδιο σημείο σε άλλη σημείωση να υπάρχει ο χαρακτηρισμός "not interesting" από τον ίδιο author. Στην συνέχεια όμως κάναμε την σκέψη ότι πιθανά ο σχολιαστής να αλλάξει τη γνώμη του για το έγγραφο και να δημιουργήσει μία νέα σημείωση, η οποία θα εκφράζει διαφορετική γνώμη από αυτήν που είχε αρχικά. Για το λόγο αυτό επιτρέπουμε

στο σύστημα μας να δέχεται σημειώσεις που εκφράζουν διαφορετικές απόψεις του ίδιου author για το ίδιο έγγραφο όπως φαίνεται στο παρακάτω σχήμα.



Σχήμα 10 ***Σημειώσεις διαφορετικού τύπου ενός author στο ίδιο έγγραφο***

2ος περιορισμός

Για σημειώσεις που αφορούν περισσότερα από ένα έγγραφα. Αν δύο έγγραφα συσχετίζονται σε κάποια σημεία τους με σχέση agreement τότε στα ίδια σημεία δεν επιτρέπεται να συσχετιστούν με τη σχέση disagreement σε καμμία σημείωση από τον ίδιοauthor.

(For all D_0, D_1 in Document ∧ part, part' in annotatedPart ∧ a, a' in Annotation ∧ sim in Document!(has similar opinion with) ∧ agr in (has similar opinion with)!agreement ∧ sim = from(agr) ∧ dif in Document!(has different opinion of) ∧ dis in (has different opinion of)!disagreement ∧ dif = from(dis))
[D_0.(has similar opinion with) = sim ∧ to(sim) = D_1 ∧ D_0. annotatedPart = part ∧ D_1. annotatedPart = part' ∧ to(agr) = a ∧ D_0.(has different opinion of) = dif ∧ to(dif) = D_1 ∧ to(dis) = a' → a ≠ a' ∧ a.author ≠ a'.author]

3.3 Αρχιτεκτονική

3.3.1 Ανεξάρτητες Υπηρεσίες – Υπηρεσία Υπομνηματισμού στις ψηφιακές βιβλιοθήκες

Μία ψηφιακή βιβλιοθήκη προσφέρει στους χρήστες υπηρεσίες σχετικές με τη διαχείριση ηλεκτρονικών εγγράφων. Για παράδειγμα εάν κάποιος ζητήσει πρόσβαση στα έγγραφα της ψηφιακής βιβλιοθήκης είναι υποχρεωμένος να κάνει login δίνοντας κάποιο password. Μερικά από τα έγγραφα της ψηφιακής βιβλιοθήκης είναι σε θέση να τα δει ενώ σε άλλα δεν του επιτρέπεται η πρόσβαση. Ο χρήστης μιας ψηφιακής βιβλιοθήκης μπορεί - εάν έχει τη σχετική άδεια - μετά από κατάλληλες επερωτήσεις στη βάση και να ανακτήσει ένα πλήθος εγγράφων. Το σύστημα διαχείρισης της ψηφιακής βιβλιοθήκης είναι υπεύθυνο για όλες τις λειτουργίες που προαναφέραμε (εισαγωγή στη ψηφιακή βιβλιοθήκη, έλεγχο πρόσβασης σε αρχεία, εκτέλεση Queries κ.α). Η ιδέα μας είναι η υπηρεσία υπομνηματισμού να αποτελεί άλλη μια από τις λειτουργίες που προσφέρει η ψηφιακή βιβλιοθήκη. Αυτό σημαίνει ότι θα πρέπει να επικοινωνεί – συνεργάζεται με τις ήδη υπάρχουσες λειτουργίες της ψηφιακής βιβλιοθήκης και να είναι σε θέση να προσαρμοστεί με εκείνες τις υπηρεσίες που θα προστεθούν στο μέλλον σε αυτήν.

Το Scholnet [SCHOLNET1] [SCHOLNET2] αποτελεί ένα project που σκοπό έχει να επεκτείνει τις βασικές υπηρεσίες που προσφέρει το ERCIM Technical Reference Digital Library με εργαλεία, που υλοποιούν νέες υπηρεσίες ώστε να χειρίζονται πολυμέσα και να παρέχουν ικανοποιητικό περιβάλλον για συνεργασία μεταξύ ερευνητών.

Στα σχέδια μας είναι να μπορέσουμε να εντάξουμε την εφαρμογή μας στην ψηφιακή βιβλιοθήκη Scholnet, η οποία μιλάμε πρόκειται να είναι μία κατανεμημένη ψηφιακή βιβλιοθήκη. Αυτό σημαίνει ότι έχει δημιουργηθεί από αυτόνομες ανεξάρτητες υπηρεσίες οι οποίες μπορούν να παρέχονται από οποιοδήποτε σημείο του Internet. Όταν όλες αυτές οι υπηρεσίες συνδυαστούν τότε δημιουργούν μία κατανεμημένη ψηφιακή βιβλιοθήκη. Οι υπηρεσίες που παρέχει το Scholnet στους χρήστες του περιλαμβάνουν αποθήκευση και πρόσβαση σε multimedia, multilingual resources, σημειώσεις σε ψηφιακά έγγραφα (text-only), user registration κ.α.

Οι επικοινωνία μεταξύ αυτών των υπηρεσιών γίνεται μέσω ενός προκαθορισμένου πρωτοκόλλου κάνοντας χρήση του HTTP protocol. Ενδιαφέρον παρουσιάζει το γεγονός ότι μερικές από τις υπηρεσίες πρόκειται να είναι κατανεμημένες σε διαφορετικούς servers, άλλες

θα είναι replicated ενώ κάποιες άλλες θα είναι κεντρικοποιημένες υπηρεσίες.. Η κατανομή των υπηρεσιών σε σχέση με τον παραπάνω διαχωρισμό γίνεται ώς εξής:

Έχουν οριστεί τρεις κλάσεις

a. Distributed
b. Replicated
c. Centralized

Κάθε μία από αυτές τις υποκλάσεις αντιπροσωπεύει αντίστοιχα τις κατανεμημένες, replicated και κεντρικοποιημένες υπηρεσίεςτης της ψηφιακής βιβλιοθήκης. Στην πρώτη κατηγορία ανήκουν οι υπηρεσίες repository, Library management, Index, Multimedia Storage, Annotation. Στη δεύτερη κατηγορία ανήκουν οι υπηρεσίες Meta Service, Collection Service, User Interface, Browse, Query Mediator. Στην τρίτη κατηγορία ανήκουν οι υπηρεσίες registry, Personalized Dissemination και Multilingual Thesaurus. Πρώτα θα δούμε ποιες είναι οι υπηρεσίες της ψηφιακής βιβλιοθήκης για την οποία μιλάμε καθώς και τον τρόπο με τον οποίο αυτές συνεργάζονται μεταξύ τους και στη συνέχεια θα δούμε ποια είναι η σχέση κάθε μιας από αυτές με την υπηρεσία υπομνηματισμού.

1. Repository

   Αποθηκεύει τα έγγραφα της ψηφιακής βιβλιοθήκης Είναι κατανεμημένη υπηρεσία διότι μπορεί να υπάρχουν πολλοί servers οι οποίοι να έχουν έγγραφα σε διαφορτικά σημεία στο Internet – σε διαφορετικά authorities όπως λέμε.

2. Multimedia Storage

   Αποθηκεύει video και υποστηρίζει την προώθηση τους στους χρήστης είτε ως ολόκληρα είτε ως τμήματα του video (scenes, shots, frames). Είναι μία ακόμη κατανεμημένη υπηρεσία καθώς διαφορετικά authorities μπορούν να έχουν το καθενα το δικό του server για αποθήκευση πολυμέσων.

3. Library management

   Είναι μία υπηρεσία η οποία υποστηρίζει την αποστολή, ανάκτηση και αντικατάσταση των εγγράφων της ψηφιακής βιβλιοθήκης. Ανήκει στην κατηγορία των κατανεμημένων υπηρεσιών διότι κάθε authority έχει το δικό του server που αναλαμβάνει τη διαχείριση των εγγράφων.

4. Index

Δέχεται queries και δίνει σαν αποτέλεσμα μία λίστα από document identifiers που ικανοποιούν αυτά τα queries. Η υπηρεσία αυτή μπορεί να είναι replicated αλλά και κατανεμημένη σε πολλούς servers.

5. Query Mediator

Διανύμει τα queries στους κατάλληλους index – servers.

6. Browse

Διαχειρίζεται βιβλιογραφικές εγγραφές (bibliographic records) σε συνάρτηση με κάποιο συγκεκριμένη μορφοποίηση μετα – δεδομένων.

7. Registry

Υποστηρίζει την αποθήκευση και την πρόσβαση σε στοιχεία που αφορούν τους αποδεκτούς χρήστες των εγγράφων και τα αποδεκτά user groups που επιτρέπεται να έχουν πρόσβαση ή δικαιώματα αλλαγής στα έγγραφα της ψηφιακής βιβλιοθήκης. Η υπηρεσία αυτή παρέχεται μόνο από έναν server.

8. Personalized Dissemination

Υπηρεσία που ενημερώνει τους χρήστες για την εισαγωγή στην ψηφιακή βιβλιοθήκη νέων εγγράφων σχετικών με τα ενδαιφέροντα τους.

9. Multilingual Thesaurus

Η υπηρεσία αυτή φροντίζει να διατηρεί πολυγλωσσικούς θησαυρούς σε έναν κεντρικό server.

10. User Interface

Μεσολαβεί μεταξύ των χρηστών και των υπόλοιπων υπηρεσιών.

11. Meta Service

Η υπηρεσία αυτή διαχειρίζεται τις μετα-πληροφορίες που επιτρέπουν την κατανομή των υπηρεσιών σε πολλούς servers. Συλλέγει πληροφορίες από όλους τους servers και τις οργανώνει σε έναν χάρτη τον οποίο στη συνέχεια παρέχει σε όσους τον

ζητήσουν ώστε όλοι οι servers να γνωρίζουν τις πληροφορίες οι οποίες είναι απαραίτητες για την εκτέλεση των λειτουργιών τους όταν αλληλεπιδρούν με κάποιες από τις υπόλοιπες υπηρεσίες.

12. Annotation

Η υπηρεσία υπομνηματισμού (annotation service) του Scholnet αποτελεί μία κατανεμημένη υπηρεσία. Οι annotation servers αποθηκεύουν σημειώσεις σε έγγραφα και τις κάνουν προσβάσιμες στους χρήστες. Είναι κατανεμημένη υπηρεσία καθώς κάθε repository server έχει έναν annotation server για τις σημειώσεις των εγγράφων που περιέχει.

Οι υπηρεσίες που αναφέραμε προφανώς πρέπει να επικοινωνούν με κάποιο κοινά αποδεκτά από τους κατασκευαστές τρόπο. Ο τρόπος που επιλέξαμε είναι το πρωτόκολλο OLP (Open Library Protocol). Το πρωτόκολλο αυτό είναι αρκετά ευέλικτο ώστε να επιτρέπει την προσθήκη νέων υπηρεσιών. Βασίζεται στο ευρέως διαδεδομένο HTTP.

3.3.2 Αλληλεπίδραση του συστήματος υπομνηματισμού με τις υπόλοιπες υπηρεσίες της Ψηφιακής Βιβλιοθήκης Scholnet

Η υπηρεσία υπομνηματισμού αλληλεπιδρά με τις εξής υπηρεσίες του Scholnet :

A) Meta Service

B) Registry

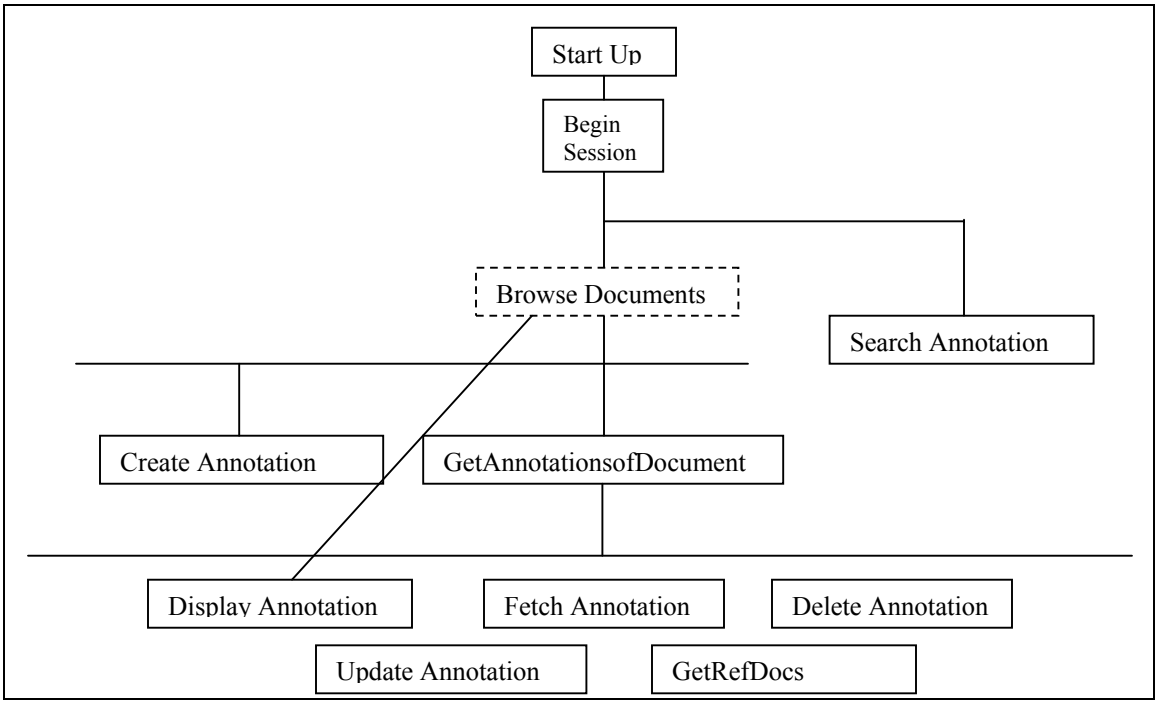
C) Repository

Η λειτουργία της ψηφιακής βιβλιοθήκης προυποθέτει μερικούς κανόνες όπως για παράδειγμα το γεγονός ότι πρέπει να γίνονται κάποιοι έλεγχοι πριν την εκτέλεση οποιουδήποτε query, επίσης είναι απαραίτητη η ανάθεση κάποιου id σε κάθε νέο χρήστη καθώς και η δημιουργία δομών με χρήσιμα στοιχεία για τα queries που πρόκειται να ακολουθήσουν όπως διευθύνσεις, ports κ.λπ. Για να επιτευχθεί η δημιουργία των δομών αυτών είναι απαραίτητη η εκτέλεση κάποιων λειτουργιών για ορισμένες από τις υπηρεσίες. Επομένως η εκτέλεση ορισμένων λειτουργιών προηγείται κάποιων άλλων. Παρακάτω εξηγούμε ποια είναι η αλληλουχία των λειτουργιών κάθε μιας από τις υπηρεσίες ώστε να αποκτήσουμε τα επιθυμητά στοιχεία.

Θα περιοριστούμε στην υπηρεσία υπομνηματισμού καθώς στην υλοποίηση αυτής της υπηρεσίας έχουμε συμβάλει. Αρχικά είναι απαραίτητη η εγκατάσταση καθενός από τους annotation servers. Κατά τη διαδικασία της εγκατάστασης γίνεται γνωστό σε κάθε έναν από αυτούς η διεύθυνση του meta server. Αυτό πρέπει να γίνει ώστε να ενημερώσουν τον meta server για την ύπαρξη τους. Στη συνέχεια έχοντας ξεκινήσει τη λειτουργία του καθένας από τους annotation servers αποκτά από τον meta server ορισμένα από τα στοιχεία τα οποία του είναι απαραίτητα για τις λειτουργίες που μπορεί να εκτελέσει (όπως πχ οι διευθύνσεις των repository server).

Πριν κάποιος χρήστης ζητήσει να εκτελέσει κάποια από τις λειτουργίες του συστήματος υπομνηματισμού είναι υποχρεωμένος να αποκτήσει ένα έγκυρο session id το οποίο θα του επιτρέψει να χρησιμοποιήσει το OLP. Αυτό το κάνει καλώντας την λειτουργία BeginSession η οποία επιστρέφει ένα έγκυρο id το οποίο αναθέτει στη συνέχεια στον χρήστη ο οποίος έκανε την αίτηση και παραμένει έγκυρο μέχρι να κληθεί η εντολή EndSession ή μέχρι να περάσει ορίσμένο χρονικό διάστημα στο οποίο ο χρήστης δεν θα απασχολήσει την υπηρεσία υπομνηματισμού.

Σε ορισμένες από τις λειτουργίες του συστήματος υπομνηματισμού πρέπει να παρέχουμε το annotation – handle μιας σημείωσης ή ακόμη τα document – handles ορισμένων εγγράφων. Αυτό μπορεί να γίνει με δύο τρόπους. Ο πρώτος είναι ο χρήστης να γνωρίζει το annotation – handle ή τα document – handles και να τα πληκτρολογήσει ο ίδιος και ο δεύτερος είναι να κάνει χρήση ορισμένων από τις λειτουργίες που παρέχει το σύστημα υπομνηματισμού ώστε να πάρει ως αποτέλεσμα τα επιθυμητά handles. Παρακάτω περιγράφουμε σχηματικά την αλληλουχία των λειτουργιών για το σύστημα υπομνηματισμού.

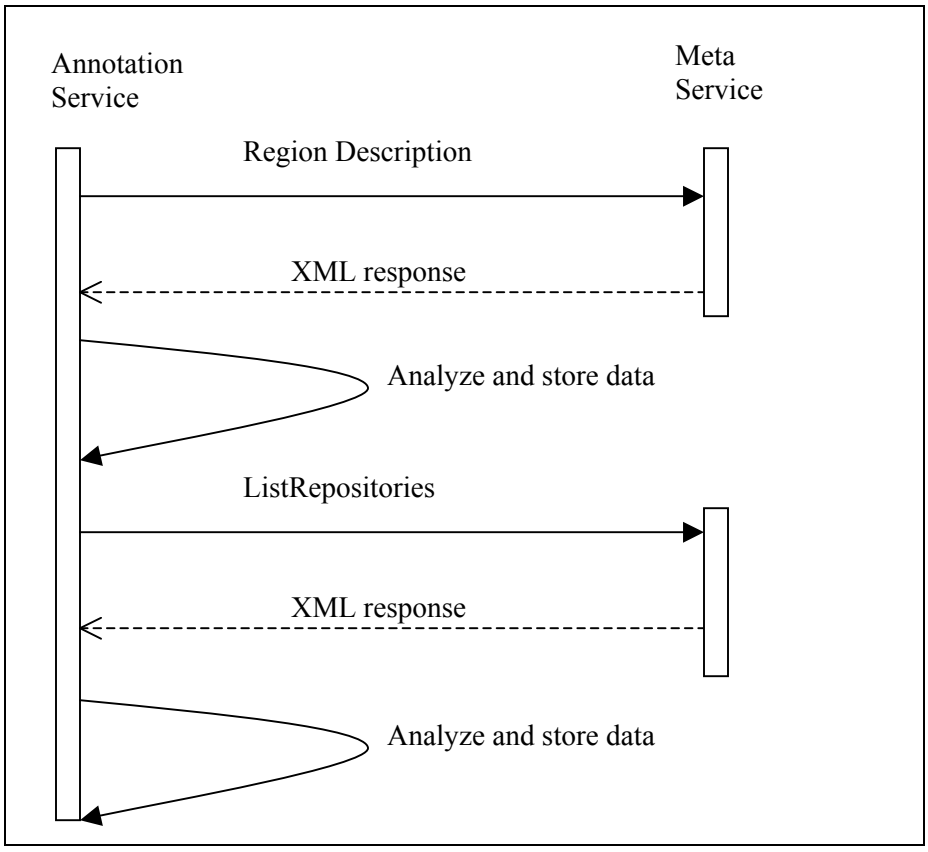


Σχήμα 11: ***Διάγραμμα λειτουργιών στην υπηρεσία Υπομνηματισμού***

Κάποιος χρήστης μπορεί να αποκτήσει δυνατότητα χρήσης του συστήματος μετά την κλήση της *BeginSession* (καλείται μόνη της). Στη συνέχεια ο χρήστης μπορεί να κάνει χρήστη της *SearchAnnotation* ή να πλοηγηθεί με το GUI στα έγγραφα της ψηφιακής βιβλιοθήκης και εάν καταλήξει σε κάποιο έγγραφο τότε να κάνει χρήση των λειτουργιών *CreateAnnotation* και *GetAnnotationsofDocument.* Εάν κάνει χρήση της τελευταίας τότε έχει μια λίστα από σημειώσεις στις οποίες μπορεί να εφαρμόσει τις λειτουργίες *DisplayAnnotation, FetchAnnotation, DeleteAnnotation, UpdateAnnotation, GetRefDocs.*

Θα εξηγήσουμε στη συνέχεια με τη βοήθεια διαγραμμάτων πως γίνεται η αλληλεπίδραση μεταξύ του συστήματος υπομνηματισμού και των υπόλοιπων υπηρεσιών της ψηφιακής βιβλιοθήκης για κάθε μία από τις λειτουργίες του συστήματος υπομνηματισμού.

1) **Annotation Service Start – up**

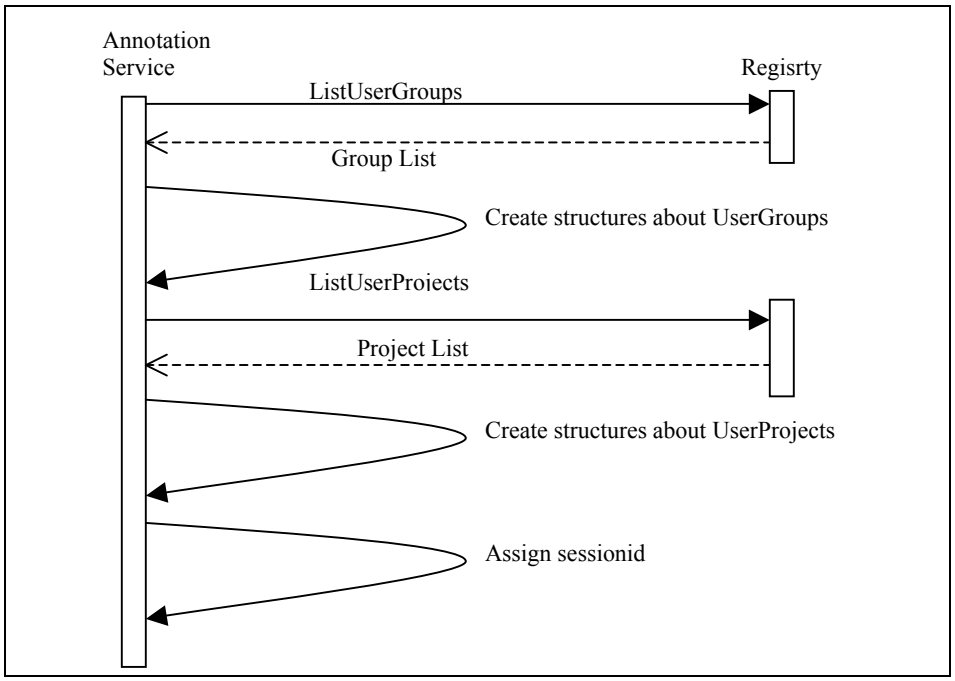


Σχήμα 12: ***Annotation Service start up***

Κατά την εκκίνηση της λειτουργίας annotation απαιτείται η επικοινωνία κάθε annotation server με τον Meta Server ώστε να ενημερωθεί αυτός για όσα στοιχεία είναι απαραίτητα να γνωρίζουν οι άλλες υπηρεσίες για τους annotation servers. Το URL του Meta server είναι γνωστό κατά τη διαδικασία εγκατάστασης του annotation server και έτσι μπορεί να γίνει η αποστολή των στοιχείων αυτών. Επίσης ο Meta Server επιστρέφει στους annotation servers στοιχεία απαραίτητα για την παραπέρα λειτουργία τους όπως μία λίστα με όλα τα Repositories (δηλαδή τα URL στα οποία βρίσκονται τα ψηφιακά έγγραφα.) για τα queries που αυτοί εκτελούν.

2) **Begin Session**

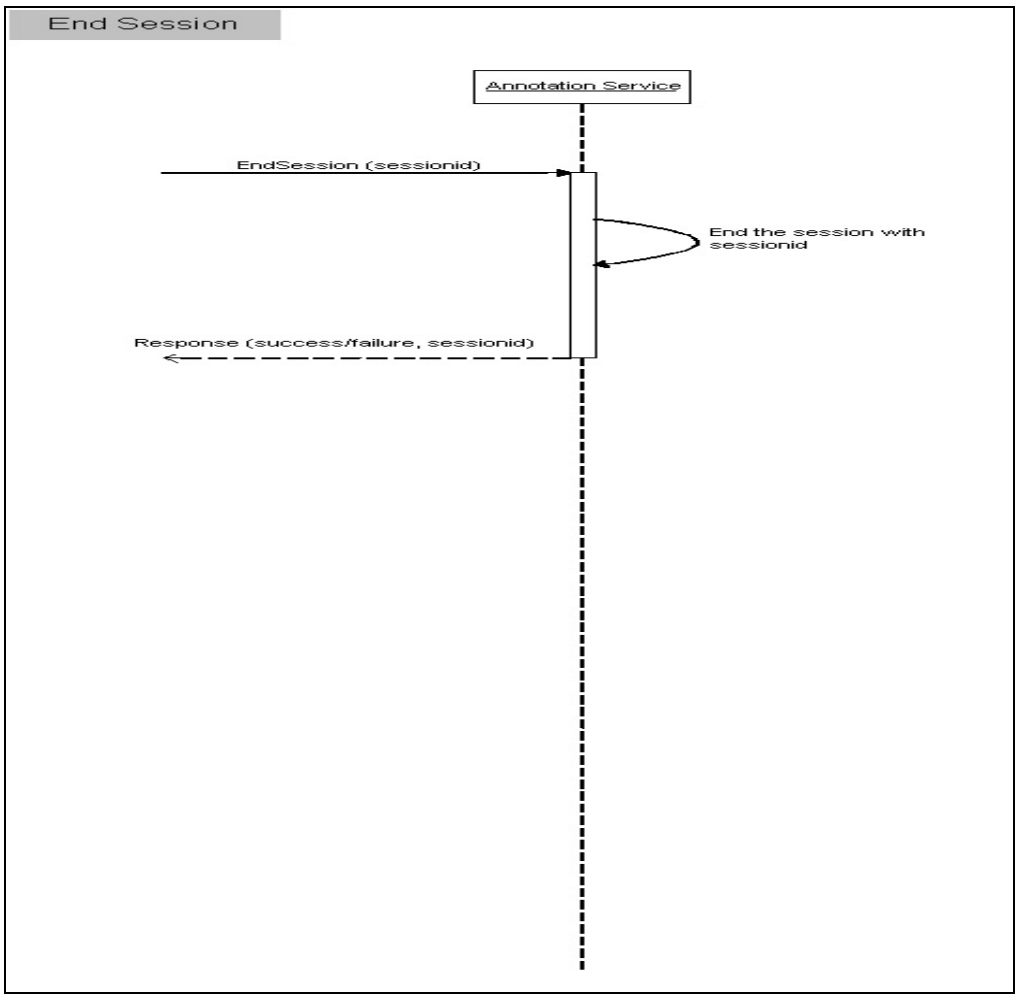
Είναι η λειτουργία η οποία γίνεται στην αρχή χρήσης του συστήματος υπομνηματισμού από κάθε user. Ο χρήστης δίνοντας το login του σαν όρισμα καλεί τη λειτουργία BeginSession με τον τρόπο που ορίζει το OLP και περιμένει να του επιτραπεί η χρήση της υπηρεσίας υπομνηματισμού. Ο annotation server στον οποίο έχει γίνει η αίτηση για δημιουργία session δημιουργεί με τη βοήθεια του Registry μία λίστα με όσα groups , projects ο user έχει πρόσβαση ώστε να μπορεί να ελέγχει (filtering) τα αποτελέσματα των queries που αυτός εκτελεί. Τέλος αποδίδει στον χρήστη ένα session id το οποίο είναι μοναδικό και θα πάψει να είναι έγκυρο μόλις ο χρήστης διακόψει τη λειτουργία annotation.



Σχήμα 13: ***verb BeginSession***

3) **End Session**

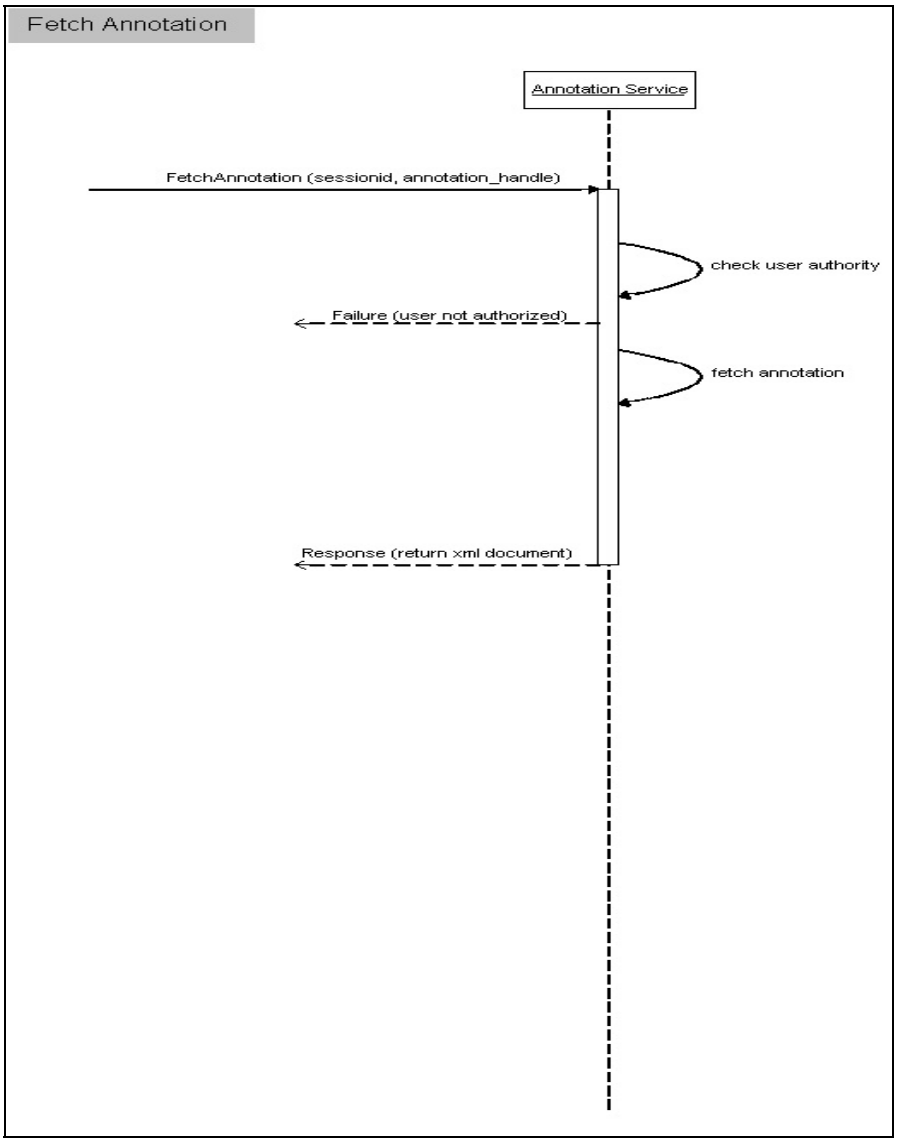
Λειτουργία που κλείνει το session το οποίο έχει δημιουργήσει ο χρήστης για την εκτέλεση queries στον annotation server.



Σχήμα 14: ***verb EndSession***

4) **Fetch Annotation**

Λειτουργία που σκοπό έχει την ανάκτηση πληροφορίας από τη βάση για μία συγκεκριμένη σημείωση. Ο χρήστης απλά παρέχει το id της σημείωσης (annotation-handle). Στη συνέχεια γίνεται έλεγχος από τον annotation server εάν ο χρήστης έχει permissions για την συγκεκριμένη σημείωση. Εάν όλα πάνε καλά τότε εξάγεται ως αποτέλεσμα ένα XML έγγραφο το οποίο περιγράφει την σημείωση όπως έχει αποθηκευτεί στη βάση. Το annotation – handle ο χρήστης μπορεί να το ανακτήσει καλώντας κάποιο από τα άλλα verbs ή να το συμπληρώσει μόνος του.

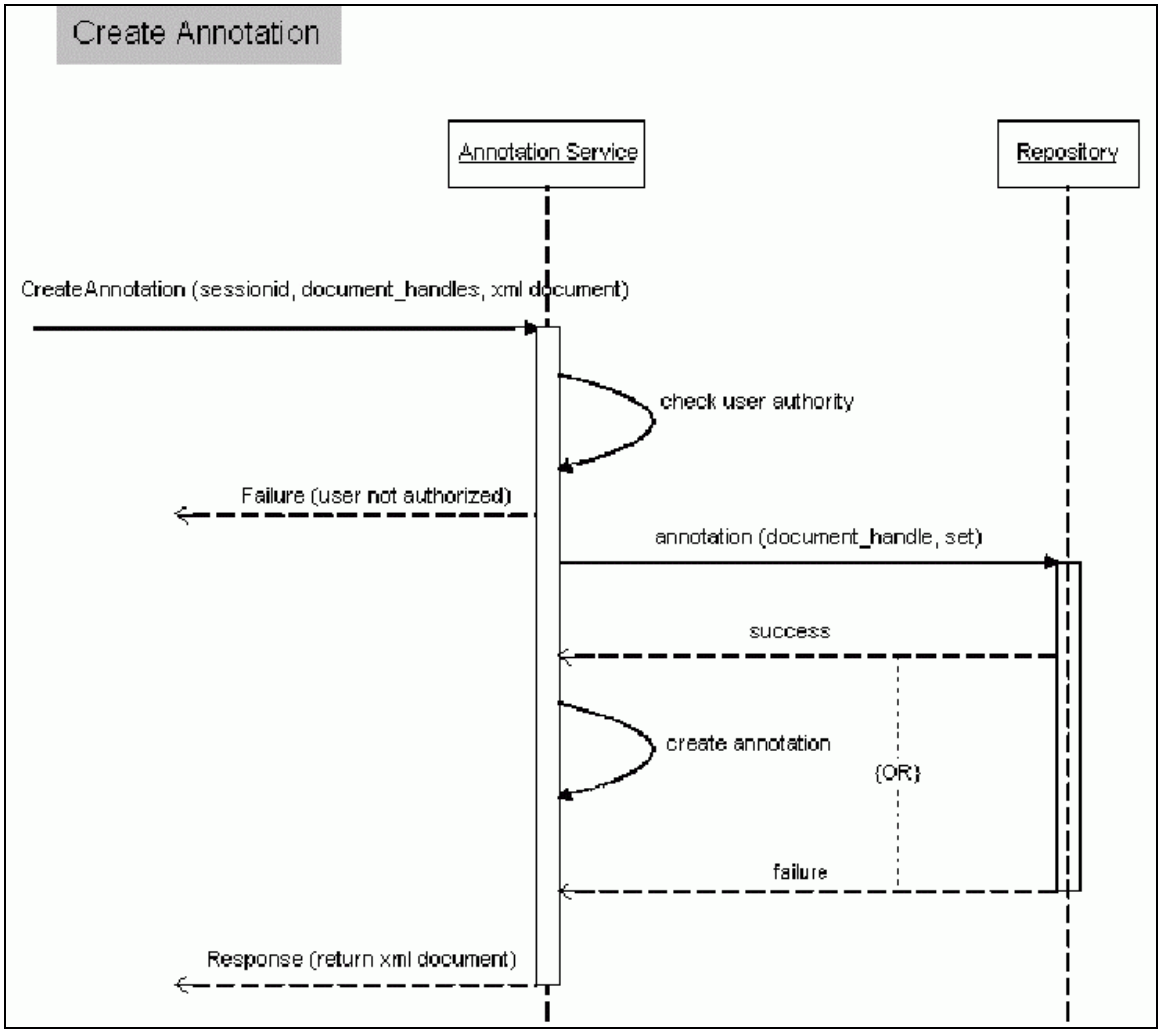


Σχήμα 15: ***verb FetchAnnotation***

5) **Create Annotation**

Λειτουργία που σκοπό έχει την αποθήκευση πληροφορίας στη βάση για μία συγκεκριμένη σημείωση. Ο χρήστης πρέπει να παρέχει στο σύστημα ένα XML έγγραφο το οποίο να περιγράφει την σημείωση. Το έγγραφο αυτό το δημιουργεί με χρήση της υπηρεσίας User Interface και μεταφέρεται στον annotation server με post request όπως ορίζει το HTTP. Εάν όλα πάνε καλά και η σημείωση καταχωρηθεί σωστά τότε εμφανίζεται μύνημα επιτυχούς δημιουργίας διαφορετικά εμφανίζεται κάποιο μύνημα που εξηγεί ότι κάτι δεν λειτούργησε σωστά κατά την διαδικασία που προηγήθηκε. Εάν κάποιο από τα έγγραφα που συσχετίζονται στην σημείωση

εμπλέκονται για πρώτη φορά σε διαδικασία υποσημείωσης, τότε πρέπει να ενημερωθεί ο repository server στον οποίο αντιστοιχεί και να θέσει την τιμή true στο annotated – flag.

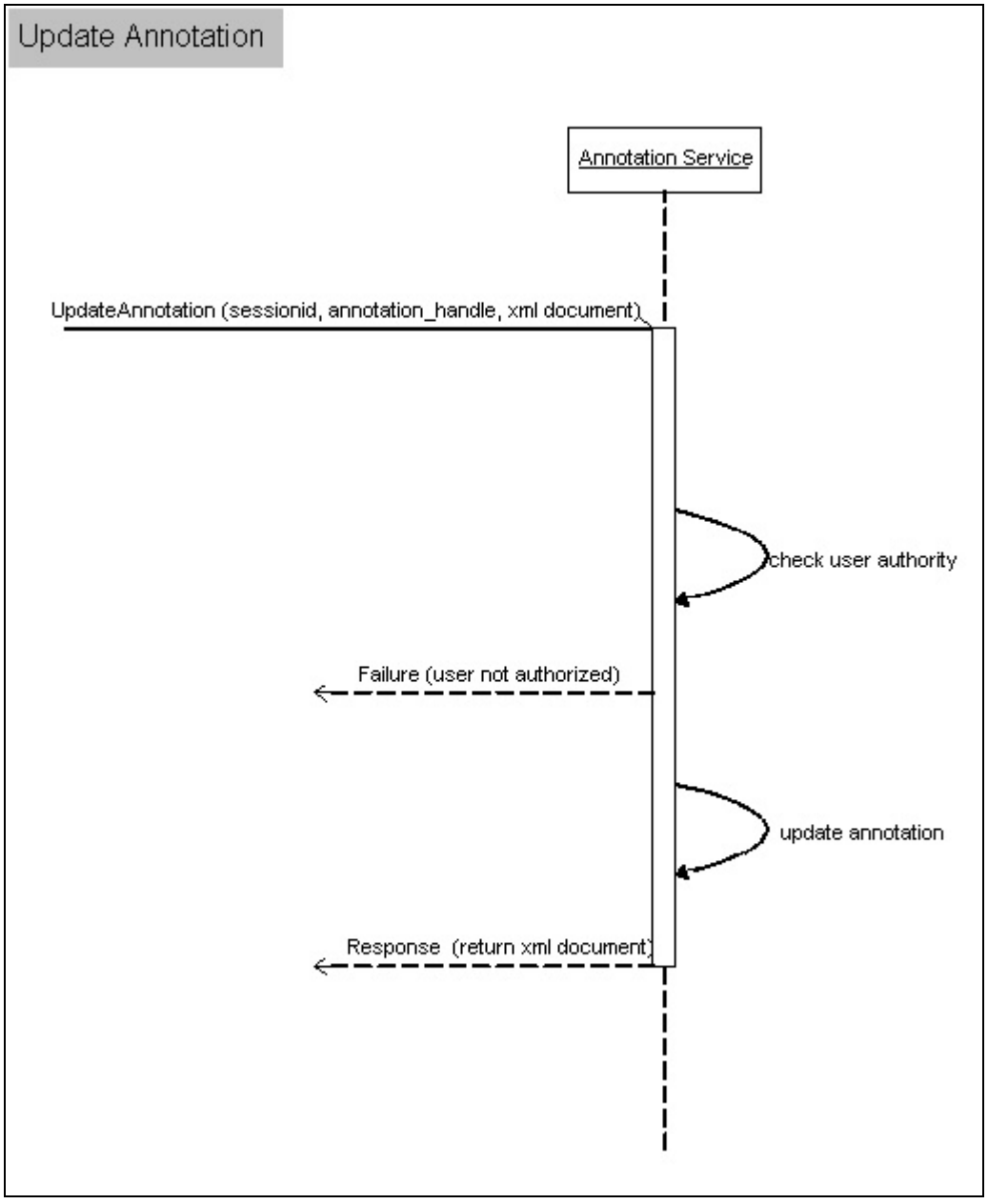


Σχήμα 16: ***verb CreateAnnotation***

6) **Update Annotation**

Λειτουργία που σκοπό έχει την ενημέρωση της αποθηκευμένης πληροφορίας στη βάση για μία συγκεκριμένη σημείωση. Ο χρήστης πρέπει να παρέχει στο σύστημα ένα XML έγγραφο το οποίο να περιγράφει την σημείωση. Το έγγραφο αυτό το δημιουργεί με χρήση της υπηρεσίας User Interface και μεταφέρεται στον annotation server με post request όπως ορίζει το HTTP. Εάν όλα εξελιχθούν ομαλά και η σημείωση ενημερωθεί σωστά τότε εμφανίζεται μύνημα επιτυχούς ενημέρωσης

διαφορετικά εμφανίζεται κάποιο μύνημα που εξηγεί ότι κάτι δεν πήγε καλά κατά την διαδικασία που προηγήθηκε.

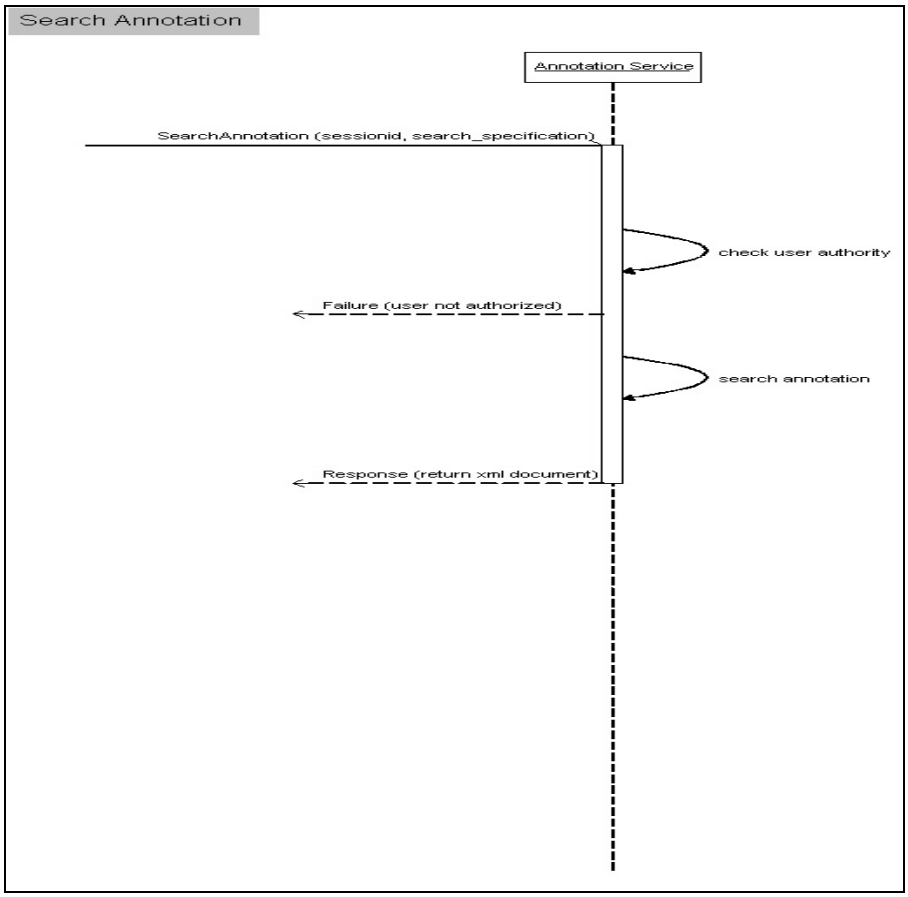


Σχήμα 17: ***verb UpdateAnnotation***

7) **Search Annotation**

Λειτουργία που σκοπό έχει την ανάκτηση αποθηκευμένης πληροφορίας στη βάση με βάση κάποιους περιορισμούς τους οποίους έχει ορίσει ο χρήστης. Ο χρήστης

παρέχει στο σύστημα ένα XML έγγραφο το οποίο να περιγράφει χαρακτηριστικά των σημειώσεων που αναζητά. Για παράδειγμα μπορεί να αναζητήσει όλες τις σημειώσεις στις οποίες συγγραφέας είναι κάποιος συγκεκριμένος user. Μπορεί επίσης να κάνει συνδιαστικές ερωτήσεις αναζήτησης, όπως να βρεθούν οι σημειώσεις που έχει συγκεκριμένος χρήστης και αφορούν συγκεκριμένο project ή ακόμη να βρεθούν εκείνες οι οποίες συσχετίζουν δύο συγκεκριμένα έγγραφα μεταξύ τους κ.α. Το XML έγγραφο που περιγράφει τα χαρακτηριστικά των σημειώσεων που αναζητά το δημιουργεί με χρήση της υπηρεσίας User Interface και μεταφέρεται στον annotation server με post request όπως ορίζει το HTTP. Στη συνέχεια εάν ο χρήστης έχει δικαίωμα πρόσβασης στα στοιχεία της βάσης εκτελείται το query και τα αποτελέσματα μεταφέρονται σαν XML αρχείο στο User Interface.



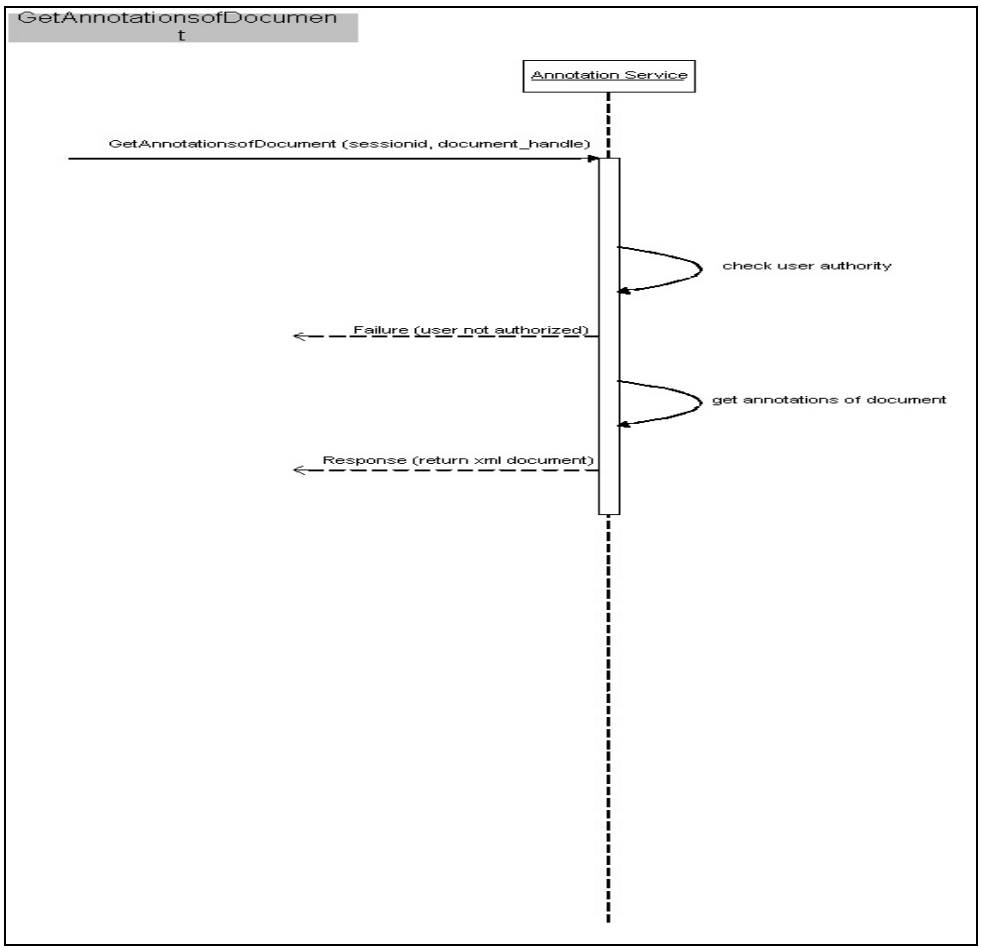
Σχήμα 18: ***verb SearchAnnotation***

8) **GetRefDocuments**

Λειτουργία που σκοπό έχει την ανάκτηση όσων εγγράφων συσχετίζονται με μία σημείωση. Δίνουμε σαν όρισμα το annotation – handle και εάν ο χρήστης έχει δικαίωμα πρόσβασης στη βάση τότε τα document – handles των εγγράφων καθώς και λίγη πληροφορία που σχετίζεται με αυτά είναι διαθέσιμη στον χρήστη με τη μορφή XML εγγράφου.

9) **GetAnnotationsofDocument**

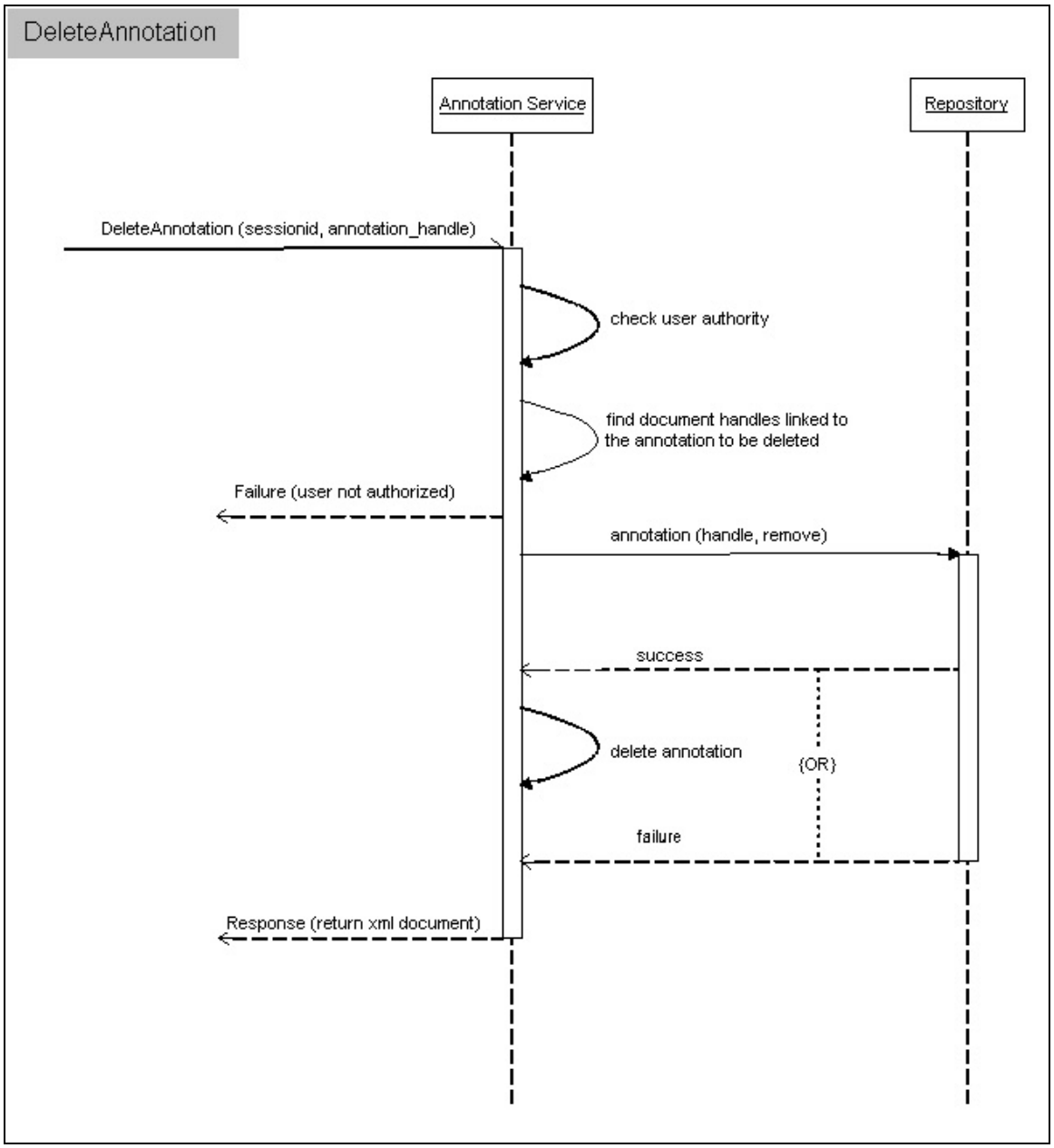
Λειτουργία που σκοπό έχει την ανάκτηση όσων σημειώσεων συσχετίζονται με ένα έγγραφο. Δίνουμε σαν όρισμα το document – handle και εάν ο χρήστης έχει δικαίωμα πρόσβασης στη βάση τότε τα annotation – handles των σημειώσεων που σχετίζονται με το έγγραφο είναι διαθέσιμα στον χρήστη με τη μορφή XML εγγράφου.



Σχήμα 19: ***verb GetAnnotationsofDocument***

10) **Delete Annotation**

Λειτουργία που σκοπό έχει την διαγραφή από τη βάση μίας συγκεκριμένης σημείωσης. Ο χρήστης πρέπει να παρέχει στο σύστημα το annotation – handle της σημείωσης.. Εάν όλα πάνε καλά και η σημείωση διαγραφεί σωστά τότε εμφανίζεται μύνημα επιτυχούς διαγραφής διαφορετικά εμφανίζεται κάποιο μύνημα που εξηγεί ότι κάτι δεν έχει παέι καλά κατά την διαδικασία που προηγήθηκε. Εάν κάποιο από τα έγγραφα, τα οποία συσχετίζονται με τη σημείωση, που μόλις διαγράφηκε δεν εμπλέκονται σε άλλη σημείωση, τότε πρέπει να ενημερωθεί ο repository server στον οποίο αντιστοιχεί και να θέσει την τιμή false στο annotated – flag.

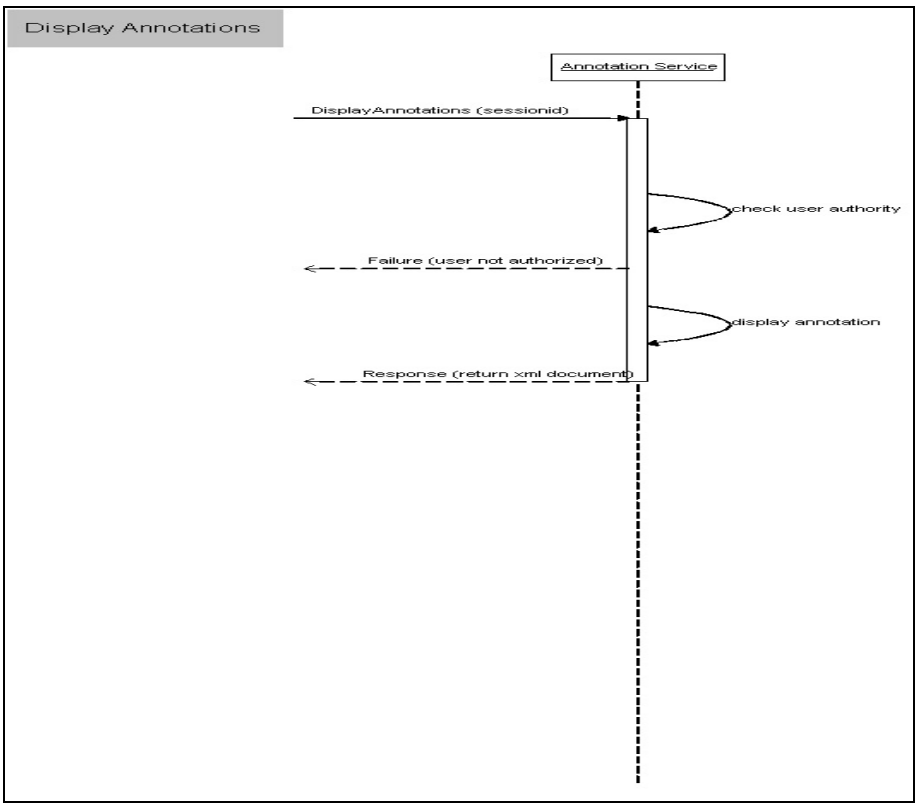


Σχήμα 20: ***verb DeleteAnnotation***

11) **Display Annotation**

Λειτουργία που σκοπό έχει την ανάκτηση όλων των σημειώσεων που σχετίζονται με το όρισμα που δίνει ο χρήστης σαν input. Το όρισμα αυτό μπορεί να είναι είτε annotation – handle είτε document – handle. Εάν δωθεί κάποιο annotation – handle τότε το output της λειτουργίας είναι όλες οι σημειώσεις οι οποίες αντιστοιχούν στην σημείωση που δώθηκε σαν όρισμα. Εάν δωθεί σαν όρισμα ένα document – handle

τότε το output της λειτουργίας είναι όλες οι σημειώσεις οι οποίες αντιστοιχούν στο έγγραφο που δώθηκε.



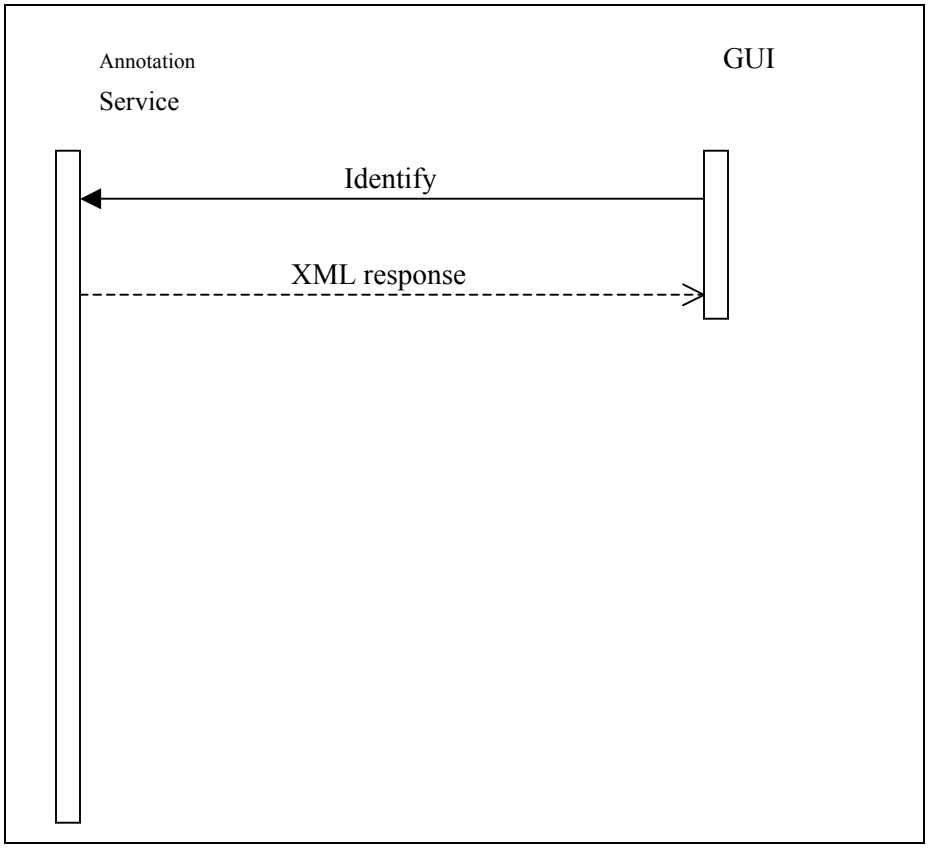
Σχήμα 21: ***verb DisplayAnnotation***

Εκτός από τα παραπάνω verbs κρίθηκε απαραίτητη η δημιουργία ορισμένων ακόμη τα οποία να χρησιμοποιούνται από τις υπόλοιπες υπηρεσίες της ψηφιακής βιβλιοθήκης έτσι ώστε να παρέχουν πληροφορίες σχετικά με τις λειτουργίες της δικής μας υπηρεσίας. Αυτά τα verbs είναι τα εξής : Identify, DescribeVerb και ListVerbs. Κάθε μία από τις ανεξάρτητες υπηρεσίες της Ψηφιακής Βιβλιοθήκης έχει αναπτύξει verbs με αυτά τα ονόματα. Οι λειτουργίες που επιτελούν αυτά για την υπηρεσία Υπομνηματισμού περιγράφονται παρακάτω.

12) **Identify**

Η λειτουργία του verb Identify για την υπηρεσία Annotation, έχει σκοπό να δώσει σαν έξοδο ένα XML αρχείο το οποίο να περιγράφει το όνομα της υπηρεσίας καθώς και άλλα στοιχεία όπως το URL του server, το email του administrator κ.λπ. Ιδιαίτερα για τη δική

μας υπηρεσία στην οποία θέλαμε να δώσουμε ευελιξία σε ό,τι έχει να κάνει με την δυναμική δημιουργία XML εγγράφων τα οποία περιγράφουν τις σημειώσεις, η χρήση της Identify συνδιάστηκε με ένα Query στην βάση το αποτέλεσμα του οποίου είναι η κατασκευή της φόρμας την οποία το User Interface εμφανίζει στον χρήστη για την δημιουργία της σημείωσης.



Σχήμα 22: ***verb Identify***

13) **DescribeVerb**

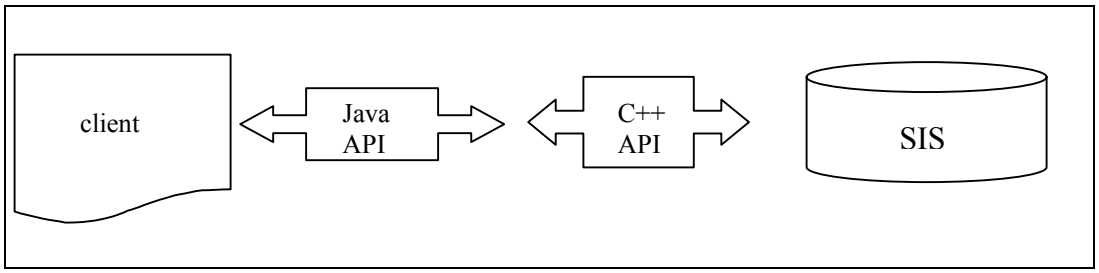
Κάθε φορά που καλείται η λειτουργία αυτή αποστέλεται σαν απάντηση ένα XML έγγραφο το οποίο περιέχει πληροφορίες και μια μικρή περιγραφή για το verb του οποίου η περιγραφή ζητάται από τον χρήστη.

14) **ListVerb**

Κάθε φορά που καλείται η λειτουργία αυτή αποστέλεται σαν απάντηση ένα XML έγγραφο το οποίο περιέχει λίστα με όλα τα Verbs τα οποία υποστηρίζει η υπηρεσία υπομνηματισμού.

3.3.3 Annotation server

Με τον όρο annotation server εννούμε το σύνολο των προγραμμάτων τα οποία απαιτούνται για να επιτευχθεί ο στόχος, ο οποίος είναι η δημιουργία ενός συστήματος υπομνηματισμού για τα ηλεκτρονικά έγγραφα της ψηφιακής βιβλιοθήκης. Το ιδιαίτερο χαρακτηριστικό της βάσης, που χρησιμοποιούμε για αποθήκευση των σημειώσεων είναι το γεγονός ότι πρόκειται για μία οντοκεντρική βάση και ότι για να κάνουμε queries προς αυτήν πρέπει να γραφούν σε γλώσσα C++. Το γεγονός όμως ότι οι clients είναι κατανεμημένοι μας οδηγεί στη χρήση Java ώστε να υποστηρίξουμε τα ετερογενή συστήματα τα οποία πιθανά ζητήσουν πρόσβαση στη βάση μας. Αυτό έχει ως αποτέλεσμα την ανάγκη για την ύπαρξη ενός ενδιάμεσου προγράμματος το οποίο να μετατρέπει την Java σε C++ από τους clients προς τον server και αντίστροφα από C++ σε Java από τον server προς τους clients. Αυτό το πρόγραμμα υπάρχει και το έχουμε ονομάσει Java API. Στην ουσία πρόκειται για Java native methods οι οποίες καλούν τις ρουτίνες του C++ API το οποίο με τη σειρά του εκτελεί queries σε γλώσσα TELOS στο SIS.



Σχήμα 23: ***Αρχιτεκτονική συστήματος – Java API, C++ API***

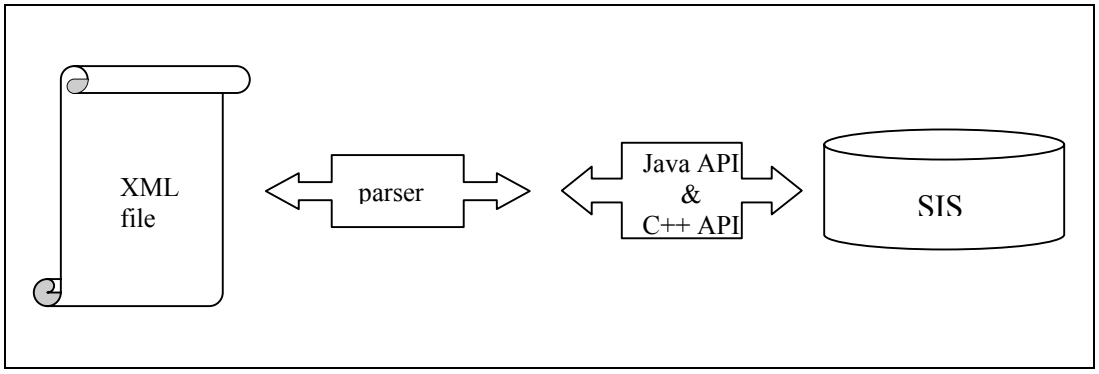
Σε προηγούμενη παράγραφο αναφέραμε ότι με τη χρήση της XML διακινούμε την αποθηκευμένη στη βάση πληροφορία μεταξύ ετερογενών συστημάτων (clients) στο διαδίκτυο.

Στη βάση μας βέβαια δεν αποθηκεύουμε ολόκληρο το XML αρχείο. Εάν το κάναμε αυτό θα επιβαρύναμε τη βάση με τα tags που αντιστοιχούν στο annotation record τόσες φορές όσες είναι και οι σημειώσεις οι οποίες βρίσκονται αποθηκευμένες σε αυτή και επίσης δε θα μπορούσαμε να κάνουμε με αποδοτικό τρόπο ερωτήσεις στη βάση. Για το λόγο αυτό αποθηκεύουμε μόνο τις τιμές των tags και των attributes που αυτά έχουν. Το μοντέλο δηλαδή που έχουμε δημιουργήσει στη βάση αποθηκεύει τους όρους οι οποίοι

όλοι μαζί αποτελούν το record μιας σημείωσης. Ήταν απαραίτητη λοιπόν η δημιουργία ενός προγράμματος το οποίο να έχει τις εξής λειτουργίες :

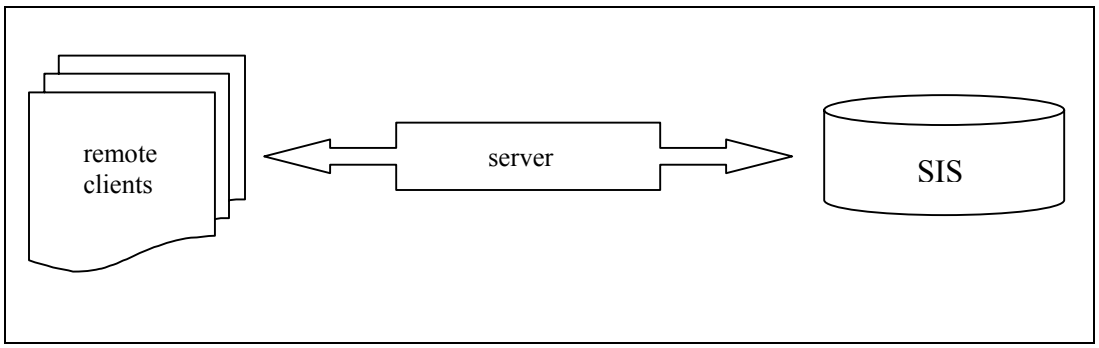
Α) κάνει parsing το XML αρχείο και απομονώνει τις τιμές των πεδίων (όρους) που έχει σε κάθε tag. Στη συνέχεια καλεί τις μεθόδους του Java API δίνοντας σ'αυτές τις μεθόδους ως ορίσματα τις τιμές που έχει απομονώσει. Οι μέθοδοι αυτές όμως καλούν με τη σειρά τους ρουτίνες του C++ API έτσι ώστε να γίνει δυνατή η επικοινωνία με το SIS και να ολοκληρωθεί η ανάκτηση ή η εισαγωγή των όρων από και προς τη βάση.

Β) Εκτελεί την αντίστροφη διαδικασία. Δηλαδή έχοντας ανακτήσει τους όρους από τη βάση το πρόγραμμα είναι υπεύθυνο για τη δημιουργία ενός XML αρχείου ώστε να γίνει δυνατή η περιγραφή και η μεταφορά της πληροφορίας που υπάρχει στη βάση στον χρήστη. Το πρόγραμμα αυτό είναι ο parser όπως φαίνεται στο παρακάτω σχήμα και επικοινωνεί τόσο με τους κατανεμημένους clients όσο και με τον server.



Σχήμα 24 : ***Γενική αρχιτεκτονική του συστήματος***

Για να επιτευχθεί αυτό πρέπει να υπάρχει ένας ενδιάμεσος server μεταξύ του client και της βάσης ώστε να εξυπηρετεί τις αιτήσεις για Queries στο SIS. Ο server αυτός αναλαμβάνει να κάνει το parsing των XML εγγράφων και επίσης να επικοινωνήσει μέσω των API's με τη βάση. Τις λειτουργίες αυτές τις κάνει για κάθε ένα client.



Σχήμα 25: ***Εξυπηρέτηση πολλών απομακρυσμένων clients***

4. Υλοποίηση

4.1 Μέσα υλοποίησης

4.1.1 Το πρωτόκολλο OLP

Η επικοινωνία μεταξύ των ανεξάρτητων υπηρεσιών του Scholnet, υλοποιείται μέσω του *OpenLib Protocol.* Αυτό κληρονομεί από το *Dienst* τους βασικούς κανόνες και τις βασικές συμβάσεις. Σε αυτό το τμήμα της εργασίας παρουσιάζουμε τους βασικούς κανόνες του OLP [OLP].

Verbs και Versions

Οι αιτήσεις του OLP ονομάζονται "verbs". Κάθε μια από τις υπηρεσίες υποστηρίζει ένα σύνολο από verbs. Κάθε υπηρεσία είναι πιθανό να υποστηρίζει περισσότερες από μία εκδόσεις ενός verb, κάθε version ίσως διαφέρει από τις υπόλοιπες στο συντακτικό ή ακόμη και εννοιολογικά. Μία έκδοση σχηματίζεται από δύο ακεραίους οι οποίοι διαχωρίζονται με μία τελεία. Π.χ *version 1.0.* Η version αυτή αφορά μόνο την υπηρεσία και δεν αφορά το πρωτόκολλο.

Είναι πιθανό ένας server που υποστηρίζει υποστηρίζει το OLP να υποστηρίζει verbs σε διάφορες εκδόσεις. Εάν μία υπηρεσία λάβει μύνημα με version number μικρότερο από το δικό της πρέπει να απαντήσει με το συντακτικό της προηγούμενης έκδοσης ή να επιστρέψει μήνυμα λάθους.

Οι κλήσεις με βάση το OLP protocol εκφράζονται σαν URL's όπως συμβαίνει με τις HTTP requests. Μια τυπική υλοποίηση χρησιμοποιεί έναν συνηθισμένο web server όπως π.χ. ο Apache, ο οποίος έχει διαμορφωθεί ώστε κάθε OLP URL να αντιστοιχεί σε μία OLP υπηρεσία.

Message Format

Όλα τα μυνήματα κωδικοποιούνται σε URL's. Το path του URL αποτελείται από τα εξής πεδία:

`OLP`*: το πεδίο αυτό εμφανίζεται λεξικογραφικά στο URL*

`Service Name`*: Το όνομα της υπηρεσίας που είναι σε θέση να εξυπηρετήσει την αίτηση, π.χ. HAS (annotation system)*

Version: *Η έκδοση του verb που καλείται*

Verb: *Το όνομα του verb, το οποίο είναι μοναδικό για κάθε υπηρεσία*

Fixed Arguments: *Όσα από τα γνωρίσματα είναι υποχρεωτικά για τη λειτουργία κάθενος verb*

KeyWord Arguments: *Το συντακτικό για τα γνωρίσματα έχουν τη μορφή key = value. Εάν υπάρχουν περισσότερα από ένα keywords τότε αυτά διαχωρίζονται με ένα ερωτηματικό. Τα γνωρίσματα εμφανίζονται με τυχαία σειρά. Γενικά είναι προεραιτικά.*

Το διαχωριστικό των τμημάτων του path είναι το slash '/' εκτός από το διαχωριστικό πριν από το τμήμα των keywords το οποίο είναι το αγγλικό ερωτηματικό '?'.

Παράδειγμα

Παρακάτω έχουμε μια κλήση για την έκδοση 1.0 της υπηρεσίας *Repository* με ένα *fixed argument* και δύο *optional keywords(version and view).*

OLP/Repository/1.0/Structure/handlecorp/docid?version=2&view=book

Το πλήρες URL για αυτήν την κλήση προς έναν web server είναι:

http://xx/OLP/Repository/1.0/Structure/handlecorp/docid?version=2&view=book

Ειδικοί χαρακτήρες

Το συντακτικό των URI's δίνει σε ορισμένους χαρακτήρες ιδιαίτερους ρόλους. Εάν αυτοί οι χαρακτήρες πρέπει να χρησιμοποιηθούν με κάποιο άλλο τρόπο τότε αυτοί πρέπει να γραφούν με άλλο τρόπο. Ένα '%' ακολουθούμενο από έναν δεκαεξαδικό αριθμό. Οι δεσμευμένοι χαρακτήρες είναι οι εξής.

Πίνακας 4: ***Δεσμευμένοι χαρακτήρες – escape sequences***

Character	Role	Escape Sequence
/	Path component seperator	%2F
?	Query component separator	%3F
#	Fragment Identifier	%23
=	Name/Value separator	%3D
&	Argument Seperator in Query Component	%26
:	Host Port separator	%3A
;	Authority/Set separator	%3B

Ο χαρακτήρας κενό ‘ ’ μπορεί να μην εμφανίζεται πουθενά μέσα στο URL. Μπορεί να γραφεί με ένα ‘+’ (ή με ένα %20).

Message Responses

Οι απαντήσεις σε διάφορες αιτήσεις παίρνουν το formatt που παίρνουν οι HTTP responses με συμπληρωμένα τα κατάλληλα HTTP header πεδία. Ο τύπος επιστροφής που προσδιορίζεται για κάθε μύνημα είναι MIME type και συμπεριλαμβάνεται στο HTTP – content type header πεδίο.

MIME types

Οι ακόλουθοι MIME types μπορεί να αποτελέσουν responses σε OLP protocol requests:

- *Text/plain* για αιτήσεις που αφορούν μη δομημένη πληροφορία
- *Text/XML* για αιτήσεις που αφορούν δομημένη πληροφορία πχ ένα verb που ζητά την εσωτερική δομή ενός digital object. Αυτό το έγγραφο καταγράφει το DTD (Document Type Definition) για κάθε verb που έχει text/xml responce. Όλες οι XML responses του OLP πρωτοκόλλου έχουν τα ακόλουθα χαρακτηριστικά.

  Το πρώτο tag είναι ένας XML ορισμός στον οποίο η version είναι πάντα 1.0 και η κωδικοποίηση είναι πάντα UTF-8. Το υπόλοιπο περιεχόμενο του εγγράφου περικλύεται σε ένα root element και έχει το ίδιο όνομα όπως το verb της request. Αυτό το root element έχει ένα απλό γνώρισμα που λέγεται version το οποίο έχει τιμή η οποία είναι η version του verb που κλήθηκε.

Status Codes

Οι κωδικοί που επιστρέφονται και αφορούν τα λάθη και την κατάσταση της response είναι οι ίδιοι όπως εκείνοι που ορίζονται για το HTTP πρωτόκολλο. Μία φυσιολογική απόκριση του συστήματος για ένα OLP μύνημα στο HTTP σηματοδοτείται με ένα κωδικό 200. Αποκρίσεις που δηλώνουν λάθος παίρνουν κωδικούς 4xx 5xx όπως ακριβώς συμβαίνει στο HTTP.

400 – Το OLP request έχει σχηματιστεί με λάθος τρόπο. Πχ λάθος arguments ή λάθος σειρά στα arguments

401 – Εάν ο client δεν έχει τα δικαιώματα που απαιτούνται ώστε να κάνει την αίτηση

404 – Εάν ένα έγγραφο που ζητάται στο request δεν υπάρχει στο repository

415 – Εάν το format, η κωδικοποίηση κλπ που ζητάται στην αίτηση δεν είναι διαθέσιμα ή δεν υφίστανται

501 – Εάν η υπηρεσία OLP που ζητάται δεν υποστξρίζεται από τον συγκεκριμένο server

503 – Εάν ο Server είναι σε θέση να απαντήσει αλλά δεν μπορεί να κρατήσει την σύνδεση ανοικτή για τόσο χρόνο όσο απαιτεί η απάντηση.

4.1.2 Java HttpServlets

Το σύστημα μας προφανώς έχει ιδιαίτερη αξία διότι χρησιμοποιείται μέσω του web. Αυτό βέβαια έχει το σημαντικό μειονέκτημα ότι δεν μπορούμε να γνωρίζουμε τί είδους clients πρόκειται να κάνουν queries στη βάση μας. Ζητούμενο ήταν να είμαστε σε θέση να ικανοποιήσουμε κάθε αίτηση για χρήση της εφαρμογής μας ανεξάρτητα από την πλατφόρμα του client που έκανε την αίτηση. Για να μπορέσουμε να δώσουμε στην εφαρμογή μας ένα τέτοιο χαρακτηριστικό επιλέξαμε ως μέσο υλοποίησης τη γλώσσα προγραμματισμού Java [JAVA1], η οποία χαρακτηρίζεται ως *platform – independent.*

Οι παραδοσιακοί compilers μετατρέπουν τον source code σε εντολές γλώσσας μηχανής. Δηλαδή ο εκτελέσιμος κώδικας αφορά το μηχάνημα στο οποίο έγινε compile ο source κώδικας. Για το λόγο αυτό σε αυτού του είδους τους compilers υπάρχει εξάρτηση από το μηχάνημα στο οποίο έγινε compiled ο source κώδικας. Ο compiler της Java όμως λειτουργεί διαφορετικά. Μετατρέπει τον Java source code σε ενδιάμεσο κώδικα ο οποίος μεταφράζεται *"at run time"* σε έντολές μηχανής από την Java Virtual Machine. Έτσι ο ενδιάμεσος κώδικας μπορεί να παραχθεί σε ένα προγραμματιστικό περιβάλλον και να εκτελεστεί σε κάποιο άλλο το οποίο να διαθέτει μία JVM η οποία να μεταγλωτίσει τον ενδιάμεσο κώδικα σε εντολές γλώσσας μηχανής για το μηχάνημα στο οποίο πρόκειται να τρέξει το πρόγραμμα. Επιτυγχάνουμε δηλαδή τη δημιουργία κώδικα οποίος μπορεί να τρέξει σε οποιαδήποτε πλατφόρμα διαθέτει μία JVM.

Η γλώσσα προγραμματισμού Java στηρίζεται σε κλάσεις και αντικείμενα τα οποία κληρονομούν τις ιδιότητες των κλάσεων στις οποίες ανήκουν. Η Java διαθέτει ένα τεράστιο πλήθος από κλάσεις κάθε μία από τις οποίες επιτελεί ένα διαφορετικό σκοπό. Υπάρχουν κλάσεις για τη δημιουργία γραφικών, για τη δημιουργία σύνθετων δομών, για την επικοινωνία με σχεσιακές βάσεις δεδομένων, για την δημιουργία εφαρμογών αρχιτεκτονικής client – server κ.α. Μέσα από αυτό το πλήθος των κλάσεων βρίκαμε ότι η Java παρέχει μία κλάση η οποία προσωμοιώνει το πρωτόκολλο HTTP στην ουσία καθώς μέσα από τις μεθόδους της επιτυγχάνει την υλοποίηση των λειτουργιών GET, POST, SERVICE κ.λπ. όπως συμβαίνει στο HTTP protocol [HTTP]. Χρησιμοποιούμε την κλάση αυτή, η οποία είναι η *HttpServlet* έτσι ώστε να επιτύχουμε την υλοποίηση του OLP το οποίο όπως είδαμε έχει τη βάση του στο HTTP.

Τα Servlets [SERVLETS3] είναι μία κλάση της Java η οποία προσθέτει λειτουργικότητα σε έναν web server με παρόμοιο τρόπο όπως τα applets προσθέτουν λειτουργικότητα σε έναν browser. Τα servlets έχουν σχεδιαστεί με τρόπο ώστε να υποστηρίζουν ένα μοντέλο request/response το οποίο είναι συνηθισμένο στους web servers. Σε ένα τέτοιο μοντέλο ένας client στέλνει μία αίτηση – *request* σε έναν server και ο server αποκρίνεται με την αποστολή μίας απόκρισης – *response*. Από το Java Servlet Development Kit (JSDK), χρησιμοποιήσαμε το Java Servlet API [SERVLETS1] για να δημιουργήσουμε servlets ώστε να είμαστε σε θέση να λάβουμε requests από clients και να δημιουργήσουμε responses προς αυτούς.

Ο ρόλος των Servlets στην αρχιτεκονική συστημάτων

Τα servlets μπορούν να παίξουν ιδιαίτερα σημαντικό ρόλο στην αρχιτεκτονική συστημάτων εξαιτίας της δύναμης και της ευελιξίας που διαθέτουν. Μπορούν να ασκήσουν την επεξεργασία που αποδίδεται στο μεσαίο επίπεδο μιας εφαρμογής (middle tier), να λειτουργήσουν ως ένας proxy για κάποιους clients, ακόμη και να επαυξήσουν τα χαρακτηριστικά του middle tier με την ενίσχυση του προσθέτοντας ισχύ νέων πρωτοκόλλων ή άλλων χαρακτηριστικών.

Το μεσσαίο επίπεδο δρα ως ο application server στα επονομαζόμενα client/server συστήματα τριών επιπέδων, θέτοντας τον εαυτό του ανάμεσα σε έναν client τύπου web browser και στην βάση δεδομένων. Με τη χρήση του επιπέδου αυτού αρκετή από την επεξεργασία, που θα έπρεπε να κάνουν οι clients ή ο DB server αποδίδεται σε κάποιον ενδιάμεσο. Με τον τρόπο αυτό όμως οι clients γίνονται πιο γρήγοροι και πιο «ελαφροί» ενώ τα μηχανήματα στα οποία τρέχουν οι DB servers έχουν αποκλειστική αρμοδιότητα την κάλυψη των αναγκών της βάσης δεδομένων και δεν σπαταλούν άδικα τους υπολογιστικούς πόρους σε άσκοπες συνδέσεις με 100δες ή ακόμη και 1000δες clients γεγονός που θα τους έκανε πιο αργούς και ευάλωτους σε ζητήματα ασφάλειας κ.α. Οι ρόλοι που μπορεί να παίξει ένα τέτοιο επίπεδο υλοποιημένο με τεχνολογία Java περιλαμβάνει τους εξής:

- Διαχείριση των transaction στη βάση δεδομένων.
- Αντιστοίχιση των clients προς εξυπηρέτηση σε ένα τυχαίο server επιλεγμένο μέσα από ένα δοσμένο σύνολο από servers με γνωστή αρχιτεκτονική.
- Υποστήριξη διαφορετικών τύπων clients εφόσον αυτοί είναι εφοδιασμένοι με μία έκδοση της Java Virtual Machine.

Τα servlets δεν τρέχουν με τον ίδιο τρόπο όπως τα applets ή οι εφαρμογές Java (applications). Απαραίτητη προυπόθεση για να τρέξει ένα servlet είναι η εγκατάσταση σε κάποιο μηχάνημα ενός server οποίος να υποστηρίζει τα servlets. Στη συνέχεια απαιτούνται δύο βήματα.

1. εγκατάσταση του servlet στο μηχάνημα που είναι εγκατεστημένος ο server σύμφωνα με τις οδηγίες που παρέχεται στο εγχειρίδιο του server.
2. Δημιουργία αίτησης προς το servlet μέσω ενός client π.χ. Internet Explorer.

Τα Servlets τρέχουν στο μηχάνημα που τρέχει και ο web server σαν τμήμα της ίδιας διαδικασίας σαν να ήταν ένας web server τα ίδια. Ο web server είναι υπεύθυνος για την αρχικοποίηση, την κλήση, και την καταστροφή καθενός από τα instances του servlet.

Ένας web server επικοινωνεί με ένα servlet μέσω ενός απλού interface. Το interface αυτό αποτελείται από τρεις κύριες μεθόδους:

- init()
- service()
- destroy()

Η μέθοδος init()

Όταν ένα servlet φορτώνεται για πρώτη φορά, η μέθοδος του init() καλείται. Αυτή επιτρέπει στο servlet να ασκήσει οποιαδήποτε διαδικασία set – up όπως για παράδειγμα να ανοίξει συνδέσεις σε κάποιους servers ή να διαβάσει κάποια αρχεία. Εάν ένα servlet έχει εγκατασταθεί μόνιμα σε έναν server, τότε αυτό φορτώνεται όταν ο server ξεκινά να τρέχει. Διαφορετικά, ο server ενεργοποιεί το servlet όταν εκείνος λάβει την πρώτη άιτηση από έναν client για κάποια υπηρεσία που παρέχει το servlet.
Η μέθοδος init() πρόκειται να έχει ολοκληρωθεί πρίν οποιαδήποτε άλλη κλήση γίνουν προς το servlet—όπως πχ στην μέθοδο service(). Ας προσέξουμε ότι η μέθοδος init() θα κληθεί μόνο μία φορά, δεν θα κληθεί ξανά εκτός εάν το servlet έχει γίνει unloaded και μετα reloaded ξανά από τον server.
Η μέθοδος init() έχει ένα argument, μία αναφορά προς ένα αντικείμενο ServletConfig το οποίο παρέχει ορίσματα αρχικοποίησης για το servlet. Αυτό το αντικείμενο έχει μία

μέθοδο getServletContext() η οποία επιστρέφει ένα αντικείμενο ServletContext το οποίο περιέχει πληροφορίες σχετικά με το περιβάλλον στο οποίο τρέχει το servlet.

Η μέθοδος service()

Η μέθοδος service() είναι η καρδιά του servlet. Κάθε request από έναν client έχει ως αποτέλεσμα μία κλήση στην μέθοδο service() του servlet. Η μέθοδος αυτή διαβάζει τις αιτήσεις και παράγει μία response με δύο παραμέτρους:

1. Ένα αντικείμενο ServletRequest με πληροφορίες από τον client. Οι πληροφορίες αυτές αποτελούνται από ζεύγη name/value των παραμέτρων και ένα InputStream. Πολλές μέθοδοι παρέχονται οι οποίοι επιστρέφουν τις πληροφορίες των παραμέτρων. Το InputStream από τον client μπορεί να αποκτηθεί με τη μέθοδο getInputStream(). Αυτή η μέθοδος επιστρέφει ένα ServletInputStream, το οποίο μπορεί να χρησιμοποιηθεί ώστε να αποκτηθούν επιπλέον δεδομένα από τον client.

2. Ένα αντικείμενο ServletResponse αναπαριστά την απόκριση του servlet προς τον client. Όταν προετοιμάζει μία απόκριση η μέθοδος setContentType() καλείται και θέτει τον τύπο MIME για την απόκριση. Στη συνέχεια η μέθοδος getOutputStream() ή η μέθοδος getWriter() μπορεί να χρησιμοποιηθεί ώστε να αποκτηθεί ένα ServletOutputStream ή ένας PrintWriter, ώστε να αποστείλει τα δεδομένα πίσω στον client.

Όπως μπορούμε να δούμε υπάρχουν δύο τρόποι για έναν client να στείλει πληροφορίες στο servlet. Ο πρώτος είναι να στείλει τις τιμές των παραμέτρων μέσω του URL και ο δεύτερος είναι να στείλει πληροφορίες μέσω του InputStream (ή του Reader).

Η μέθοδος destroy()

Η μέθοδος destroy() καλείται για να επιτρέψει στο servlet να καθαρίσει όσες πηγές πληροφορίας παραμένουν ανοικτές και πλέον πρέπει να κλήσουν (όπως ανοικτά αρχεία, ή συνδέσεις με βάσεις δεδομένων) πριν το servlet γίνει unloaded. Εάν δεν απαιτείς οποιαδήποτε clean-up λειτουργία τότε μπορεί να είναι μία άδεια μέθοδος. Ο server περιμένει να καλέσει την μέθοδο destroy() όταν όλες οι υπόλοιπες κλήσεις έχουν εξυπηρετηθεί, είτε όταν ένα ορισμένο χρονικό διάστημα περάσει. Αυτό σημαίνει ότι η

μέθοδος destroy() μπορεί να κληθεί ενώ ακόμη κάποια άλλη μέθοδος service() ακόμη τρέχει.

HTTP Support

Τα Servlets που κάνουν χρήση του HTTP protocol είναι πολύ συνηθησμένα Υπάρχει μάλιστα ιδιαίτερη βοήθεια για όσους γράφουν κώδικα για servlets και κάνουν χρήση του πρωροκόλλου αυτού. Υποστήριξη για ζητήματα που σχετίζονται με HTTP protocol παρέχεται στο package javax.servlet.http. Το HTTP είναι αρχικά των λέξεων *HyperText Transfer Protocol*. Ορίζει ένα πρωτόκολλο για χρήση από web browsers και servers για να επικοινωνούν ο ένας με τον άλλο. Το πρωτόκολο ορίζει ένα set από requests βασισμένες σε text messages οι οποίες ονομάζονται *HTTP methods*[2]. Οι HTTP methods περιλαμβάνουν τις παρακάτω:

GET, HEAD, POST, PUT, DELETE, TRACE, CONNECT, OPTIONS

Θα αναφερθούμε στις δύο πιο σημαντικές από αυτές τις μεθόδους

1) GET

Η μέθοδος GET είναι της μορφής

GET URL <http version>
Host: <target host>

Επιπρόσθετα υπάρχουν κι άλλες πληροφορίες

Για παράδειγμα το ακόλουθο HTTP GET message ζητά την home page μου από το πανεπιστήμιο:

GET csd.uch.gr/~tzoban/index.html
Connection: Keep-Alive
User-Agent: Mozilla/4.0 (compatible; MSIE 4.01; Windows NT)
Host: csd.uch.gr
Accept: image/gif, image/x-xbitmap, image/jpeg, image/pjpeg

2) POST

Μία HTTP POST request επιτρέπει στον client να στέλνει δεδομένα στον server. Αυτή η λειτουργία μπορεί να φανεί χρήσιμη σε πολλούς σκοπούς όπως :

- Αποστολή πληροφοριών σε ένα newsgroup

[2] ***Σημείωση:*** Οι προδιαγραφές του HTTP τις ονομάζει *HTTP methods.* Δεν πρέπει να τις συγχέουμε με τις Java methods.

- Να περάσουμε περισσότερη πληροφορία από όση μας επιτρέπει η GET request

Είναι ιδιαίτερα σημαντικό το τελευταίο σημείο. Η HTTP GET request παιρνά όλα τα arguments σαν μέρος του URL. Πολλοί Web servers έχουν ένα όριο στο πόσα πολλά δεδομένα μπορούν να δεκτούν σαν μέρος του URL. Η POST method περνά όλες τις μεταβλητές της σε ένα input stream, εξαλείφοντας αυτό το όριο.

Μια τυπική POST request είναι ως ακολούθως:

```
POST /servlet/MyServlet HTTP/1.1
User-Agent: Mozilla/4.0 (   compatible;    MSIE 4.01;    Windows NT)
Host: www.magelang.com
Accept: image/gif, image/x-xbitmap, image/jpeg, image/pjpeg, */
Content-type: application/x-www-form-urlencoded
Content-length: 39


name=Scott&company=MageLang%20Institute
```

Παρατηρήστε την κενή γραμμή πριν εκείνη που ξεκινά με το name. Αυτή σηματοδοτεί το τέλος των header της POST request και την αρχή των επιπλέον πληροφοριών.

4.1.3 SIS – Οντοκεντρικά μοντέλα

Το Σημασιολογικό Σύστημα Ευρετηριασμού (Semantic Index System ή SIS) [SIS1], [SIS2] είναι ένα εργαλείο για περιγραφή και τεκμηρίωση μεγάλου πληθυσμού, ιδιόμορφων, εξελισσόμενων και πολλαπλώς συνδεδεμένων δεδομένων. Για το λόγο αυτό είναι κατάλληλο για την παράσταση επιστημονικής, σχεδιαστικής και κατασκευαστικής γνώσης. Το SIS αποτελείται από ένα μηχανισμό αποθήκευσης, που βασίζεται σε ένα συνδυασμό οντοκεντρικού μοντέλου και σημασιολογικού δικτύου, και από μία γενική διαλογική επαφή χρήσης για την εισαγωγή και ανάκτηση (retrieval) πληροφοριών. Το μοντέλο των δεδομένων περιγράφεται με την γλώσσα παράστασης γνώσης Telos. Το SIS και η ενσωματωμένη σε αυτό υλοποίηση του δομικού μέρους της γλώσσας Telos έχουν αναπτυχθεί από την Ομάδα Πληροφοριακών Συστημάτων και Τεχνολογίας Λογισμικού του Ινστιτούτου Πληροφορικής του Ιδρύματος Τεχνολογίας Έρευνας (Ι.Τ.Ε.)

Σημαντικά χαρακτηριστικά της Telos

Η Telos είναι μια γλώσσα παράστασης γνώσης η οποία υποστηρίζει ένα οντοκεντρικό μοντέλο δεδομένων. Προσφέρει τους εξής στατικούς εκφραστικούς

μηχανισμούς: μη-φραγμένη ιεραρχία ταξινόμησης, πολλαπλή και αυστηρή κληρονομικότητα και πλειότιμα γνωρίσματα τα οποία μπορούν να έχουν τα δικά τους γνωρίσματα. Επίσης παρέχει μηχανισμούς εξαγωγής συμπερασμάτων και χρονικής λογικής.Η γλώσσα Telos παρέχει πολλές στάθμες αφαίρεσης, άρα προσφέρει η δυνατότητα οργάνωσης του σχήματος σύμφωνα με ένα μετασχήμα, το οποίο περιγράφει πιο αφηρημένες έννοιες και ιδιότητες, επιτρέποντας έτσι την διατύπωση και απάντηση ερωτήσεων που βασίζονται σε αφηρημένες ιδιότητες. Με τη σειρά τους, οι έννοιες του μετασχήματος μπορούν να υπάγονται σε ένα μετα-μετασχήμα, κ.ο.κ. Έτσι η Telos παρέχει ομοιόμορφη διαχείριση δεδομένων και σχήματος τόσο στην εισαγωγή στοιχείων όσο και στον μηχανισμό ερωτήσεων. Αυτό επιτρέπει δυναμικό ορισμό και ενημέρωση του σχήματος της βάσης χωρίς χρονική επιβάρυνση.

Οι οντότητες και τα γνωρίσματα θεωρούνται και χρησιμοποιούνται σαν ισότιμα αντικείμενα, τα οποία μπορούν να έχουν λογικά ονόματα. Τα λογικά ονόματα παρέχουν επιπλέον πληροφορίες για τα αντικείμενα, κάνοντας έτσι τα περιεχόμενα μιας βάσης πιο κατανοητά στο χρήστη. Επίσης αποτελούν έναν άμεσο και κατανοητό τρόπο αναφοράς των αντικειμένων της βάσης.

Στην υλοποίηση της Telos που χρησιμοποιήθηκε, οι σύνδεσμοι (ταξινόμησης, γενίκευσης, γνωρισμάτων) αποθηκεύονται ως σύνδεσμοι διπλής κατεύθυνσης προσφέροντας καλές επιδόσεις στις διασχίσεις και ερωτήσεις, μεγαλώνοντας όμως το μέγεθος της βάσης. Παρακάτω παρουσιάζονται συνοπτικά οι βασικοί εκφραστικοί μηχανισμοί που χρησιμοποιεί η γλώσσα Telos για την δημιουργία των μοντέλων παράστασης γνώσεων.

- **Ονοματοδοσία**

  Κάθε οντότητα έχει ένα εσωτερικό, παραγόμενο από το σύστημα, αναγνωριστικό όνομα. Επιπλέον ο χρήστης έχει τη δυνατότητα να ονομάσει ο ίδιος μια οντότητα με ένα *λογικό όνομα*. Το λογικό όνομα ενός γνωρίσματος είναι της μορφής x.y, όπου x είναι το λογικό όνομα της οντότητας της οποίας αποτελεί γνώρισμα και y είναι το όνομα του ίδιου.

- **Ταξινόμηση**

  Με το μηχανισμό αυτό μια ατομική οντότητα περιγράφεται ως μέλος (περίπτωση) μιας κλάσης, της οποίας κληρονομεί τα γνωρίσματα. Μια κλάση είναι και αυτή με τη σειρά της μια οντότητα, άρα μπορεί να είναι περίπτωση μιας άλλης κλάσης. Η Telos απαιτεί κάθε οντότητα να αποτελεί περίπτωση μιας κλάσης. Έτσι δημιουργείται μια μη φραγμένη ιεραρχία από κλάσεις.

Στο κατώτερο επίπεδο τοποθετούνται οι ατομικέςοντότητες ή Tokens, κατόπιν υπάρχουν οι απλές κλάσεις που αποτελούνται από ατομικές οντότητες, μετά οι μετακλάσεις που αποτελούνται από απλές κλάσεις, οι μετα-μετακλάσεις, κοκ.
Η μη φραγμένη ιεραρχία είναι άπειρη αλλά όχι υποχρεωτικά. Η απαίτηση κάθε οντότητα να ανήκει σε μια κλάση εκπληρώνεται με την ύπαρξη ειδικών κλάσεων του συστήματος ή ω-κλάσεων.
Οι κλάσεις του συστήματος δεν μπορούν να αλλαχτούν από τον χρήστη. Αποτελούν τον αρχικό πληθυσμό της βάσης και οποιαδήποτε δεδομένα που θα εισάγει στο σύστημα ο χρήστης πρέπει να συνδεθούν έμμεσα ή άμεσα με αυτές.

Όλες οι οντότητες που περιγράφονται με την Telos ταξινομούνται στην κλάση του συστήματος Object. Υποκλάσεις της Object είναι οι κλάσεις Individual και Attribute. Στην κλάση Individual ταξινομούνται οι οντότητες, οι κλάσεις από οντότητες, οι κλάσεις από κλάσεις από οντότητες, κοκ. Στην κλάση Attribute ταξινομούνται οι σχέσεις που έχουν οι οντότητες μεταξύ τους , οι κλάσεις σχέσεων, οι κλάσεις από κλάσεις σχέσεων, κοκ. Οι κλάσεις που ορίζει ο χρήστης ταξινομούνται στην κλάση του συστήματος Class. Τα Tokens ταξινομούνται στην κλάση Token. Οι απλές κλάσεις ταξινομούνται στην κλάση S_Class , οι μετακλάσεις στην M1_Class, οι μετα-μετακλάσεις στην M2_Class, κοκ. Επίσης υπάρχουν οι πρωτογενείς τιμές Integer, Real, String. Οι πρωτογενείς τιμές δεν μπορούν να δημιουργηθούν ή να καταστραφούν αλλά μόνο να γίνει αναφορά σε αυτές.

Μία οντότητα μπορεί να ανήκει σε παραπάνω από μία κλάσεις. Με αυτόν τον τρόπο μπορεί η ταξινόμηση να υποκαταστήσει την απόδοση γνωρισμάτων (δυνατότητα που χρησιμοποιείται και στο μοντέλο που δημιουργήσαμε). Με αυτόν τον τρόπο όμως πρέπει να αποδίδονται μόνο εγγενείς ιδιότητες μιας οντότητας ενώ ο μηχανισμός απόδοσης γνωρίσματος είναι πιο κατάλληλος για την περιγραφή των επιφανειακών γνωρισμάτων που πιθανόν να έχει η οντότητα.

- **Απόδοση γνωρίσματος**

Με το μηχανισμό αυτό αποδίδονται γνωρίσματα στις οντότητες. Τα γνωρίσματα αυτά μπορούν να θεωρηθούν ως σχέσεις μεταξύ οντοτήτων μιας και έχουν ένα πεδίο ορισμού (την οντότητα στην οποία αποδίδονται) και ένα πεδίο τιμών (την οντότητα που προσδιορίζει τί τύπου είναι η τιμή του γνωρίσματος). Κάθε γνώρισμα μπορεί να έχει παραπάνω από μια ή και καμία τιμή. Επίσης πορεί να έχει και αυτό γνωρίσματα

(πράγμα που απορρέει από την ισότιμη εταχείριση οντοτήτων και γνωρισμάτων από την Telos).

- **Περιορισμός στην ταξινόμηση των γνωρισμάτων**

Αν ένα γνώρισμα είναι περίπτωση μιας κατηγορίας γνωρισμάτων ονομάζεται μια κλάση γνωρισμάτων. Συνήθως με αυτόν τον όρο αναφερόμαστε σε κλάσεις γνωρισμάτων στο μετα-επίπεδο, αλλά και ένα γνώρισμα σε επίπεδο S_Class αποτελεί κατηγορία γνωρισμάτων για τις περιπτώσεις του σε επίπεδο Token , τότε το πεδίο ορισμού και το πεδίο τιμών του πρέπει να είναι περιπτώσεις των πεδίων ορισμού και τιμών της κλάσης γνωρισμάτων στην οποία ανήκει.

- **Γενίκευση**

Ο μηχανισμός αυτός ισχύει μόνο για κλάσεις (όχι δηλαδή για άτομα) που βρίσκονται στο ίδιο επίπεδο ταξινόμησης. Ορίζει μια σχέση υποσυνόλου μεταξύ των κλάσεων που ονομάζεται *isA*. Αν A isA B (A και B κλάσεις), τότε η A ονομάζεται υποκλάση της B και η B υπερκλάση της A. Η A κληρονομεί όλα τα γνωρίσματα της B και είτε έχει επιπλέον γνωρίσματα είτε περιορίζει το σύνολο τιμών των γνωρισμάτων που κληρονομεί από την B. Μια κλάση μπορεί να έχει παραπάνω από μια υπερκλάσεις. Έτσι η σχέση isA υποστηρίζει πολλαπλή και αυστηρή κληρονόμηση.

Ο μηχανισμός γενίκευσης/εξειδίκευσης επιτρέπει την οργάνωση των κλάσεων σε διάφορες ιεραρχίες γενίκευσης οι οποίες προσδίδουν οικονομία και συνέπεια στο μοντέλο, αφού δεν χρειάζεται να επαναληφθεί ο ορισμός ενός γνωρίσματος που έχει ήδη αποδοθεί σε μια γνωστή υπερκλάση.

- **Περιορισμός στις υποκλάσεις των γνωρισμάτων**

Έστω AC1 και AC2 κατηγορίες γνωρισμάτων. Αν AC1 *isA* AC2, τότε τα πεδία ορισμού και τιμών της AC1 είναι υποκλάσεις των αντίστοιχων της AC2.

Το σύστημα Σημασιολογικού Ευρετηριασμού

Το Σημασιολογικό Σύστημα Ευρετηριασμού (Semantic Index System ή SIS) είναι ένα εργαλείο για περιγραφή και τεκμηρίωση μεγάλου πληθυσμού, ιδιόμορφων, εξελισσόμενων και πολλαπλώς συνδεδεμένων δεδομένων sis). Για το λόγο αυτό είναι κατάλληλο για την παράσταση επιστημονικής, σχεδιαστικής και κατασκευαστικής γνώσης. Το SIS αποτελείται από ένα μηχανισμό αποθήκευσης, που βασίζεται σε ένα

συνδυασμό οντοκεντρικού μοντέλου και σημασιολογικού δικτύου, και από μια γενική διαλογική επαφή χρήσης για την εισαγωγή και ανάκτηση (retrieval) πληροφοριών. Το μοντέλο των δεδομένων περιγράφεται με την γλώσσα παράστασης γνώσης Telos.Ο μηχανισμός αποθήκευσης των αντικειμένων της Telos, υποστηρίζει δοσοληψίες και ταυτόχρονη πρόσβαση από πολλούς χρήστες. Το SIS συνοδεύεται από ένα μηχανισμό ερωτήσεων, ο οποίος παρέχει ένα σύνολο από ερωτηματικές εντολές, με τις οποίες είναι δυνατή η πλοήγηση στη βάση καθώς και η διατύπωση αναδρομικών ερωτήσεων με πολλαπλά κριτήρια. Οι ερωτήσεις μπορούν να γίνουν με ένα διαλογικό εργαλείο, τον answerer, ή μέσα από άλλες εφαρμογές με τη βοήθεια του programmatic query interface.

Η επικοινωνία του SIS με το χρήστη υποστηρίζεται από ένα αποδοτικό και προσαρμόσιμο εργαλείο το οποίο επιτρέπει διερεύνηση σε όλα τα επίπεδα αφαίρεσης και στα δεδομένα καθώς και παρουσιάσεις πολύμορφων δεδομένων (multimedia) εξωτερικώς αποθηκευμένων. Υπάρχει η δυνατότητα ορισμού και ενεργοποίησης προκαθορισμένων ερωτήσεων. Αυτές οι ερωτήσεις διακρίνονται σε δύο κατηγορίες: α) σε αυτές που το αποτέλεσμα τους είναι ένας γράφος και β) σε αυτές που το αποτέλεσμα τους είναι κείμενο. Οι ερωτήσεις γίνονται για μία οντότητα της βάσης, ενώ υπάρχει η δυνατότητα για διατύπωση ερωτήσεων με δελτία, δηλαδή παίρνοντας υπ' όψην δύο ή περισσότερες οντότητες. Η εισαγωγή δεδομένων γίνεται μαζικά από τον μεταφραστή της Telos ή διαλογικά από ένα Δελτίο Εισαγωγής Δεδομένων, του οποίου η λειτουργία μπορεί να προσαρμοστεί ανάλογα με τις ανάγκες κάθε εφαρμογής. Η διασύνδεση και επικοινωνία του SIS με εξωτερικά εργαλεία και εφαρμογές υποστηρίζονται από ειδικούς μηχανισμούς που έχουν αναπτυχθεί.
Το SIS αποτελεί το κύριο εργαλείο για την υλοποίηση της εργασίας και έχει ήδη χρησιμοποιηθεί σε πλήθος άλλων εφαρμογών.

4.2 Παραδείγματα χρήσης του συστήματος

Παρακάτω έχουμε μια σειρά από παραδείγματα τα οποία φανερώνουν τη δύναμη του συστήματος που έχουμε σχεδιάσει και υλοποιήσει. Αρχικά εμφανίζουμε 2 παραδείγματα που αφορούν την λειτουργία *CreateAnnotation.*

1) Σε αυτό το παράδειγμα ο χρήστης *Franco Niccoluci* κάνει μια σημείωση σε μέρος του έγγραφου «*caa2002*» το οποίο βρίσκεται στην ψηφιακή βιβλιοθήκη. Ο μοναδικός τύπος που επέλεξε είναι απλά μία παρατήρηση *"Remark"*. Το XML αρχείο που δημιουργείται και αποστέλεται στον AnnotationServer φαίνεται παρακάτω.

```
<?xml version="1.0" encoding="UTF-8" ?>
- <annotation>
    <annt-handle>ics.forth.gr/caa2002RP80-1p2c2h</annt-handle>
  - <doc-info>
      <doc-handle>ics.forth.gr/caa2002P80</doc-handle>
      <annotated-part>Page 2, column 2 half</annotated-part>
    </doc-info>
  - <description>
      <subject>Review of CAA2002 paper</subject>
      <project>CAA2002</project>
      <date>22 July 2002</date>
    </description>
  - <user-info>
      <author>Franco Nicolucci</author>
      <group>CAA2002Reviewer</group>
    </user-info>
  - <links>
    - <link-category name="evaluates">
      - <link>
          <link-name>remark</link-name>
        - <value class="Remark">
            <string-val>Page 2, column 2 half: I should
              reformulate the sentence as follows ... a function
              m (termed the membership function) whose
              values (called memberships) are between 0 and 1.
              As it is now, m is undefined and "between"
              incorrectly precedes the interval, where perhaps
              also the extremes 0 and 1 should be acceptable (0
              £ m £ 1).</string-val>
          </value>
        </link>
      </link-category>
    </links>
</annotation>
```

Παρατηρούμε ότι έχει κάνει χρήση του τύπου Remark και στην ουσία η «τιμή» της παρατήρησης βρίσκεται στο tag *string-val*.

2) Ένα πιο ενδιαφέρον παράδειγμα έχει γίνει από τον ίδιο χρήστη και αφορά μία σημείωση η οποία έχει περισσότερες από μία συνιστώσες – δηλαδή στην ίδια σημείωση αντιστοιχούν περισσότεροι του ενός τύποι. Επιπλέον κάνει χρήση ενός τύπου σημειώσεων ο οποίος συσχετίζει δύο έγγραφα μεταξύ τους. Αυτή τη φορά δηλαδή δεν κάνει απλή αναφορά σε προσωπική γνώμη όπως στα προηγούμενα παραδείγματα αλλά συσχετίζει το περιεχόμενο των εγγράφων.

```
<?xml version="1.0" encoding="UTF-8" ?>
- <annotation>
    <annt-handle>ics.forth.gr/caa2002RP80-1</annt-handle>
  - <doc-info>
      <doc-handle>ics.forth.gr/caa2002P80</doc-handle>
    </doc-info>
  - <description>
      <subject>Review of CAA2002 paper</subject>
      <project>CAA2002</project>
      <date>22 July 2002</date>
    </description>
  - <user-info>
      <author>Franco Nicolucci</author>
      <group>CAA2002Reviewer</group>
    </user-info>
  - <links>
    - <link-category name="evaluates">
      - <link>
          <link-name>opinion</link-name>
        - <value class="Opinion">
            <controlled-voc>Accepted</controlled-voc>
          </value>
        </link>
      - <link>
          <link-name>remark</link-name>
        - <value class="Remark">
            <string-val>The title incorrectly ends with a fullstop.
              Mathematical variables should always be written
              in italics (e. g. x : see Table 2, in which moreover
              X and x should be used coherently and not
              casually intermixed as they are). Reviewer's
              comment The paper is very nice and I enclose a
              list of minor annotations.</string-val>
          </value>
        </link>
      </link-category>
    - <link-category name="criticizes">
      - <link>
          <link-name>observe</link-name>
        - <value class="Observation">
            <string-val>The authors should be careful when using
              the term "prediction", especially in conjunction
              with "environmental", to avoid being charged of
```

```
                "environmental determinism", a suspicion which
                is always present about GIS use in archaeology; of
                course the authors are aware of this issue, well
                represented e.g. by van Leusen's and Gaffney's
                paper in "Archaeology and GIS" by Locke and
                Stancic.</string-val>
            </value>
        </link>
    </link-category>
  - <link-category name="records">
    - <link>
        <link-name>genre</link-name>
      - <value class="Genre">
          <controlled-voc>Paper</controlled-voc>
        </value>
      </link>
    </link-category>
  - <link-category name="is related with">
    - <link>
        <link-name link="agreement">has similar opinion
          with</link-name>
      - <value class="Doc">
        - <doc-info>
            <doc-handle>ics.forth.gr/HarrisLock1995</doc-
              handle>
            <annotated-part>page357</annotated-part>
          </doc-info>
        </value>
      </link>
    - <link>
        <link-name link="agreement">has similar opinion
          with</link-name>
      - <value class="Doc">
        - <doc-info>
            <doc-
              handle>ics.forth.gr/CrescioliD'AndreaNicolucc
              i2000</doc-handle>
          </doc-info>
        </value>
      </link>
    </link-category>
  </links>
</annotation>
```


Ο χρήστης σε αυτήν την σημείωση έκανε ένα σχόλιο (Remark) για το έγγραφο αλλά επίσης έκρινε ότι το έγγραφο είναι αποδεκτό (Opinion = accepted). Επίσης έκανε μία παρατήρηση (Observation) ενώ κατέγραψε το γεγονός ότι το έγγραφο ανήκει στην κατηγορία paper (Genre = parer). Το νέο στοιχείο εδώ είναι ότι καταγράφει μία σχέση συμφωνίας μεταξύ του εγγράφου **ics.forth.gr/caa2002P80** (στο οποίο αντιστοιχεί η σημείωση**)** και της σελίδας 357 του εγγράφου **ics.forth.gr/HarrisLock1995.**

Στην συνέχεια παραθέτουμε ένα παράδειγμα για την λειτουργία *SearchAnnotation*. Σε αυτό το παράδειγμα υποθέτουμε ότι ο χρήστης έχει δώσει ως input ένα XML αρχείο σαν αυτό που προκύπτει ως έξοδος από τη λειτουργία *Identify*. Ορισμένα από τα πεδία αυτού του αρχείου ο χρήστης καλείται να συμπληρώσει επιλεκτικά (όσων την τιμή τον ενδιαφέρουν) και κατόπιν να το στείλει στον server με μία post request. Όταν το αρχείο φθάσει στον server γίνεται ο διαχωρισμός των τιμών από τα διάφορα πεδία (tags) του αρχείου και καλείται η μέθοδος που κάνει την ανάκτηση από τη βάση. Το αρχείο που προκύπτει από αυτή τη λειτουργία είναι το εξής:

```
<annotation>
      <annt-handle>ics.forth.gr/Scholnet_Specifications_V1P135</annt-handle>
      <links>
            <link>
                  <link-name>has subject</link-name>
                  <value>Annotation Systems</value>
            </link>
            <link>
                  <link-name>has project</link-name>
                  <value>Scholnet</value>
            </link>
            <link>
                  <link-name>has author</link-name>
                  <value>Manolis</value>
            </link>
            <link>
                  <link-name>has group</link-name>
                  <value>ISL.ICS.FORTH</value>
            </link>
      </links>
</annotation>

<annotation>
      <annt-handle>ics.forth.gr/Scholnet_Specifications_V1P138</annt-handle>
      <links>
            <link>
                  <link-name>has subject</link-name>
                  <value>Information Systems</value>
            </link>
            <link>
                  <link-name>has project</link-name>
                  <value>Scholnet</value>
            </link>
            <link>
                  <link-name>has author</link-name>
                  <value>Manolis</value>
            </link>
            <link>
                  <link-name>has group</link-name>
                  <value>ISL.ICS.FORTH</value>
            </link>
      </links>
```

```
</annotation>
```

Στο συγκεκριμένο παράδειγμα το αποτέλεσμα είναι η ανάκτηση από τη βάση δύο σημειώσεων των

```
ics.forth.gr/Scholnet_Specifications_V1P138
ics.forth.gr/Scholnet_Specifications_V1P135
```

Στο XML output file που σχηματίζεται εμφανίζονται τα annotation – handles των σημειώσεων αυτών καθώς επίσης και οι τιμές των attributes που έχουν συμφωνηθεί στα specifications. Δεν είναι απαραίτητη η παρουσία όλων των attributes του annotation record παρά μόνο ορισμένων πεδίων τα οποία πρόκειται να μας υποδείξουν τα βασικά χαρακτηριστικά της σημείωσης που αναφέρεται πχ author, subject, project, group από τα οποία ο χρήστης μπορεί να διαπιστώσει εάν η σημείωση είναι κάποια από αυτές που τον ενδιαφέρουν και στη συνέχεια με την λειτουργία *FetchAnnotation* να φέρει όλα τα πεδία του Annotation Record.

Στην συνέχεια παραθέτουμε ένα παράδειγμα για την λειτουργία *DisplayAnnotation.* Σε αυτό το παράδειγμα υποθέτουμε ότι ο χρήστης έχει δώσει ως input το document-handle από ένα έγγραφο και περιμένει σαν output όσες σημειώσεις έχουν σχέση με το έγγραφο αυτό. Φροντίσαμε να έχουμε ως output αυτής της λειτουργίας ορισμένα μόνο από τα στοιχεία των σημειώσεων που ανακτώνται και όχι όλα τα στοιχεία τους. Για να δεί κάποιος χρήστης όλα τα στοιχεία της σημείωσης (το πλήρες XML που αφορά την σημείωση) θα πρέπει να καλέσει την *FetchAnnotation* με όρισμα το annotation-handle της σημείωσης.

```
<DisplayAnnotation>

    <AnnotatedHandle>ics.forth.gr/Scholnet_Specifications_V1</AnnotatedHandle>
        <Annotation>
               <handle> ics.forth.gr/Scholnet_Specifications_V1P233</handle>
                 <author>Manolis</author>
                 <project>Scholnet</project>
                 <group>ISL.ICS.FORTH</group>
                 <subject>Information Systems</subject>
                 <date>27 June 2002</date>
                 <type>evaluates</type>
                 <type>critisizes</type>
      </Annotation>

        <Annotation>
                 <handle>ics.forth.gr/Scholnet_Specifications_V1P235</handle>
                 <author>Manolis</author>
                 <project>Scholnet</project>
                 <group>ISL.ICS.FORTH</group>
                 <subject></subject>
                 <date>26 June 2002</date>
                 <type>critisizes</type>
                 <link-name>has similar opinion with</link-name>
      </Annotation>
```

```
</DisplayAnnotation>
```

Εδώ ανακτήθηκε πληροφορία για 3 σημειώσεις οι οποίες αφορούν το document του οποίου τις σημειώσεις ζήτησε να δει ο χρήστης. Όπως στην λειτουργία *SearchAnnotation* έτσι και σε αυτό το XML output file που σχηματίζεται εμφανίζονται τα annotation – handles των σημειώσεων αυτών καθώς επίσης και οι τιμές των attributes που έχουν συμφωνηθεί στα specifications. Δεν είναι απαραίτητη η παρουσία όλων των attributes του annotation record παρά μόνο ορισμένων πεδίων τα οποία πρόκειται να μας υποδείξουν τα βασικά χαρακτηριστικά της σημείωσης που αναφέρεται πχ author, subject, project, group από τα οποία ο χρήστης μπορεί να διαπιστώσει εάν η σημείωση είναι κάποια από αυτές που τον ενδιαφέρουν και στη συνέχεια με την λειτουργία *FetchAnnotation* να φέρει όλα τα πεδία του Annotation Record. Με τον τρόπο αυτό γλιτώνουμε περιττά calls στο SIS αλλά επιπλέον το XML output file είναι σχετικά μικρό και ευανάγνωστο.

5. Επίλογος

Πρώτα κάνουμε μία ανασκόπηση των κύριων χαρακτηριστικών που προσφέρει η εφαρμογή μας. Στη συνέχεια θα δούμε ποιες λειτουργίες δεν έχουμε υλοποιήσει ενώ πιθανά θα ήταν επιθυμητές.

Κύρια χαρακτηριστικά του συστήματος

Α) Προσφέρουμε τη δυνατότητα δημιουργίας και αντιστοίχισης μιας σημείωσης σε ένα κάποιο ψηφιακό έγγραφο **ή σε μέρος ενός εγγράφου**. Η σημείωση αυτή έχει τη μορφή XML εγγράφου και είναι δομημένη με βάση συγκεκριμένα DTD όπως ορίζονται στις προδιαγραφές της επικοινωνίας των διαφόρων υπηρεσιών του Scholnet. Η σημείωση αυτή μπορεί να εμπλέκει μόνο ένα έγγραφο αλλά επίσης μπορεί να εμπλέκει και περισσότερα από ένα.

Β) Ανάμεσα στους τύπους που έχει να επιλέξει ο χρήστης περιλαμβάνονται τύποι σημειώσεων με συγκεκριμένες προεπιλεγμένες τιμές και ο χρήστης απλά επιλέγει μία από αυτές, ενώ υπάρχουν τύποι σημειώσεων στους οποίους ο χρήστης καλείται να συμπληρώσει ο ίδιος κάποιο πεδίο text το οποίο στη συνέχεια αποθηκεύεται στη βάση.

Γ) Το σύστημα μας έχει βάση αποθήκευσης των σημειώσεων το SIS. Παρέχεται λοιπόν μία αξιόπιστη λύση για την φύλαξή των δεδομένων και δυνατότητες αναζήτησης και μοντελοποίησης της πληροφορίας. Παρέχουμε προκαθορισμένα Queries στους χρήστες για τη διευκόλυνση τους στην αναζήτηση των σημειώσεων και των εγγράφων, στα οποία αυτές αντιστοιχούν. Οι χρήστες είναι οργανωμένοι σε groups και projects δημιουργώντας επιπλέον βαθμίδες ασφάλειας για την πρόσβαση στις σημειώσεις.

Δ) Η εφαρμογή μας είναι μία δικτυακή εφαρμογή (κάνουμε χρήση του Internet) αρχιτεκτονικής client – server. Δημιουργούνται λοιπόν προβλήματα που έχουν να κάνουν με τη φύση των clients οι οποίοι ζητάνε να εξυπηρετηθούν από τον Annotation server. Το γεγονός ότι χρησιμοποιούμε την γλώσσα προγραμματισμού Java για την υλοποίηση του συστηματός μας προσφέρει ανεξαρτησία από τις διαφορετικές πλατφόρμες που χρησιμοποιούν οι διαφορετικοί clients στο διαδίκτυο. Επίσης η χρήση του OLP κάνει δυνατή την επικοινωνία του συστήματος μας με άλλες ανεξάρτητες εφαρμογές οι οποίες και εκείνες χρησιμοποιούν το OLP ως πρωτόκολλο επικοινωνίας.

Ο παρακάτω πίνακας δείχνει ότι σε σχέση με προυπάρχουσες εφαρμογές, η δική μας προσφέρει πιο ολοκληρωμένες λυσεις υπομνηματισμού.

Πίνακας 5*: Συγκριτικός πίνακας με ήδη υπάρχοντα συστήματα*

	Storage base	Queries	User Groups	Επικοινωνία με άλλες εφαρμογές	Structure
Amaya	OXI	OXI	Private-public	OXI	RDF
Conote	OXI	NAI	Private-public		Text
Comentor	OXI		Private-public		Text
Hypernews	OXI	OXI			Text
JotBot	OXI	OXI			Text
Principia Cybernetica	OXI			OXI	Text
CritLink				OXI	Text
Futplex	OXI	OXI			Text
Εργασία Σαράντη Τούλη	SIS	NAI	OXI	OXI	SGML
Scholnet	SIS	Προκαθορισμένα	Groups και Projects	NAI	XML

Περιορισμοί του συστήματος

Να μιλήσουμε τώρα για τους περιορισμούς που έχει το σύστημα μας. Βασικό περιορισμό αποτελεί το γεγονός ότι οι σημειώσεις μας είναι απλό κείμενο text και ένα σύνολο από προκαθορισμένες τιμές από τις οποίες ο χρήστης καλείται να επιλέξει. Το κείμενο του χρήστη και οι επιλεγμένες τιμές αποκτούν μία δομή XML αρχείου. Δηλαδή δεν υποστηρίζουμε non – textual σημειώσεις. Όμως εάν κάποιος επιθυμεί να εισαγάγει σαν σημείωση μία εικόνα, έναν γράφο ή έναν ήχο, το σύστημα μας δεν μπορεί να καλύψει αυτήν την ανάγκη. Ασφαλώς όμως μεταβάλλοντας το σχήμα της βάσης που περιγράφει τις σημειώσεις καθώς και τα XML έγγραφα, που περιγράφουν την αποθηκευμένη πληροφορία μπορούμε να δημιουργήσουμε νέα tags τα οποία να μπορούν να προσδιορίσουν αυτούς τους τύπους σημειώσεων που σε αυτή τη φάση δεν υποστηρίζουμε. Άλλος ένας βασικός περιορισμός αποτελεί το γεγονός ότι τα Queries στη βάση είναι προκαθορισμένα. Στην ουσία, ο,τιδήποτε μπορεί να κάνει ο χρήστης στο σύστημα μας είναι καθορισμένο από πριν. Αυτό σημαίνει πχ ότι κάποιος χρήστης δεν μπορεί να σχηματίσει και να εκτελέσει κάποιο query το οποίο δεν είναι υλοποιημένο από εμάς (δεν υπάρχει δυνατότητα χρήσης SQL εδώ!). Βέβαια φροντίσαμε, όταν σχεδιάζαμε την εφαρμογή, να καλύψουμε όσες περιπτώσεις φάνηκε ότι έπρεπε να καλυφθούν από

εμάς στη φάση αυτή. Νέες ανάγκες μπορούν να οδηγήσουν στη σχεδίαση και υλοποίηση καινούργιων Verbs και στον εμπλουτισμό του συστήματος με νέες λειτουργίες. Ακόμη κατά την δημιουργία μιας νέας έκδοσης ενός εγγράφου, οι σημειώσεις, που αφορούσαν τμήμα του εγγράφου που δεν έχει υποστεί μεταβολές, δεν μεταφέρονται στο καινούργιο έγγραφο. Έτσι ο αναγνώστης πρέπει να καταφύγει στην παλιά έκδοση του εγγράφου για την πρόσβαση στις σημειώσεις αυτές. Αυτό βέβαια οφείλεται στο γεγονός ότι το νέο έγγραφο αποτελεί νέα οντότητα στην ψηφιακή βιβλιοθήκη. Εάν οι διορθώσεις γίνουν στο έγγραφο το οποίο έχει ήδη σημειωθεί τότε προφανώς εξακολουθούν να είναι έγκυρες.

Βιβλιογραφία

[AMAYA] W3C's Editor/Browser
http://www.w3.org/Amaya/

[Brickley] Dan Brickley, University of Bristol R.V. Guha. Epinions Resource Description Framework (RDF) Schema Specification1
http://www.w3.org/TR/2000/CR-rdf-schema-20000327/#refs

[Bush45] Vannevar Bush. “As we may think”. 1945
http://www.theatlantic.com/unbound/flashbks/computer/Bushf.htm

[BSCW] Fraunhofer FIT and OrbiTeam Software GmbH. BSCW
http://bscw.gmd.de/

[BSCW2] Download BSCW server software
http://bscw.gmd.de/DownloadServer.html [CRIT]

[CONOTE] Jim Davis “CoNote”, 1994
http://www.cs.cornell.edu/home/dph/annotation/annotations.html

[CRIT] Ka-Ping Yee: CritLink Mediator
http://crit.org

[CYBERNETICA] F. Heylingen : About the Principia Cybernetica Server, Mar 15, 2002
http://pespmc1.vub.ac.be/SERVER.html

[Darby98] Chád Darby. Applet and Servlet Communication.
Java Developer's Journal, September 1998
http://www.j-nine.com/pubs/applet2servlet/Applet2Servlet.html

[Davis95] Jim Davis. "Annotation Systems", 1995
http://dri.cornell.edu/pub/davis/Annotation/others.html

[DC1] Dublin Core Qualifiers
http://www.dublincore.org/documents/dcmes-qualifiers/

[DC2] Andy Powell. Dublin Core metadata editor
http://www.ukoln.ac.uk/metadata/dcdot/

[DC3] Metadata Resources, Dublin Core-DC
http://www.ukoln.ac.uk/metadata/resources/dc/

[DC4] DCMI Frequently Asked Questions (FAQ)
http://dublincore.org/resources/faq/

[DC5] Dianne Hillman. Using Dublin Core
http://dublincore.org/documents/usageguide/

[GMD] http://www.gmd.de

[HTML] Hypertext Transfer Protocol -- HTTP/1.0
http://www.ics.uci.edu/pub/ietf/http/rfc1945.html

[HTTP] HTTP - Hypertext Transfer Protocol
http://www.w3.org/Protocols/

[HYPER1] Dimitri Konstantas, Jean Henry Morin: HyperNews Hypermedia Newspaper
http://cui.unige.ch/OSG/projects/media/hypernews.html

[HYPER2] Discussion of OO-software issues, thoughts and ideas
http://www-numi.fnal.gov:8875/HyperNews/get/software_discussion.html

[HYPER3] HyperNews Relation Icons Preview
http://www-numi.fnal.gov:8875/HyperNews/display_Icons.pl

[IEP99] Interactive Electronic Publishing, "Web based tools for document annotation"
http://www.elpub.org

[JAVA1] The Source for Java™ Technology
http://www.javasoft.com

[Lassila] Ora Lassila, Ralph R. Swick. Resource Description Framework (RDF) Model and Syntax Specification
http://www.w3.org/TR/1999/REC-rdf-syntax-19990222/

[Mylopoylos] J. Mylopoulos , "Conceptual Modeling and Telos", in Loucopoulos, P. And Zicari R. (eds), Conceptual Modeling, Databases and CASE: An Integrated View of Information on System Development, Wiley 1992

[OLP] Information Society RTD Standards Implementation Report, November 2001
http://www.diffuse.org/0111-rtd.html

[Plexousakis93] Dimitris Plexousakis, "Integrity Constraint and Rule Maintenance in Temporal Deductive Knowledge Bases", Proceedings of the 19th VLDB Conference Dublin, Ireland 1993

[Pierce] Andy Pierce. The Power parallel user group Homepage
http://spud-web.tc.cornell.edu/HyperNews/get/SPUserGroup.html

[RMW94] Roichester, Martin and Christian Mogensen, Terry Winogrand (1994)"Shared web Anotations As a platform for third party value added information providers: Architecture, Protocols and user examples", Techincal Report CSDTR/DLTR 1994

http://bscw.gmd.de/DownloadServer.html

[SCHOLNET1] Scholnet: A Digital Library Testbed to Support Networked Scholarly Communities
http://www.ics.forth.gr/isl/projects/projects_individual.jsp?ProjectID=4

[SCHOLNET2] Scholnet : Developing a Digital Library Testbed to Support Networked Scholarly Communities
http://www.ercim.org/scholnet/

[SERVLETS1] Java™ 2 Platform, Enterprise Edition, v 1.3 API Specification
http://java.sun.com/j2ee/sdk_1.3/techdocs/api/index.html

[SERVLETS2] Servlets FAQ Home Page
http://www.jguru.com/faq/home.jsp?topic=Servlets

[SERVLETS3] Using Servlets and JavaServer Pages
http://servlet.java.sun.com/manual/servlets/1-intro.htm#529506

[SIS] The Semantic Index System - SIS
http://www.ics.forth.gr/isl/r-d-activities/sis.html

[Τούλης94] Σαράντης Τούλης: Σχεδίαση και υλοποίηση ενός συστήματος υπομνηματισμού ηλεκτρονικών εγγράφων. Πανεπιστήμιο Κρήτης Τμήμα Επιστήμης Υπολογιστών 1994
http://www.ics.forth.gr/isl/publications/by_type.html#msc

[XML1] The Tutorial for the Java™/ Api for XML Parsing (JAXP) version 1.1
http://java.sun.com/xml/tutorial_intro.html

[XML2] XML Tutorial DevelopMentor,
http://www.develop.com/tutorials/xml/xml-overview.asp

[XML3] XML in Netscape and in Explorer
http://www.w3schools.com/xml/xml_browsers.asp

[XML4] Edd Dumbill. Distributed XML The role played by XML in the next-generation Web. September 06, 2000
http://www.xml.com/pub/a/2000/09/06/distributed.html

[XML5] John Perkins. RDF, XML and the OAI Protocol Avoiding Mixing Apples and Oranges.
http://www.cimi.org/wg/oai/RDF_XML_Logoze_0601.html

[XML6] Tim Berners- Lee. Why XML model is different than RDF model, September 1998
http://www.w3.org/DesignIssues/RDF-XML.html

[Zeiger99] Stefan Zeiger. Servlets Essentials. November 4 1999
http://www.novocode.com/doc/servlet-essentials/