ΠΑΝΕΠΙΣΤΗΜΙΟ ΚΡΗΤΗΣ
ΣΧΟΛΗ ΘΕΤΙΚΩΝ ΕΠΙΣΤΗΜΩΝ
ΤΜΗΜΑ ΕΠΙΣΤΗΜΗΣ ΥΠΟΛΟΓΙΣΤΩΝ

# ΚΑΤΑΝΕΜΗΜΕΝΟΙ ΜΗΧΑΝΙΣΜΟΙ ΕΥΡΕΤΗΡΙΑΣΜΟΥ ΚΑΙ ΑΝΑΖΗΤΗΣΗΣ

Αντώνης Σιδηρόπουλος

Μεταπτυχιακή Εργασία

Ηράκλειο, Φεβρουάριος 1999

ΠΑΝΕΠΙΣΤΗΜΙΟ ΚΡΗΤΗΣ
ΣΧΟΛΗ ΘΕΤΙΚΩΝ ΕΠΙΣΤΗΜΩΝ
ΤΜΗΜΑ ΕΠΙΣΤΗΜΗΣ ΥΠΟΛΟΓΙΣΤΩΝ

# ΚΑΤΑΝΕΜΗΜΕΝΟΙ ΜΗΧΑΝΙΣΜΟΙ ΕΥΡΕΤΗΡΙΑΣΜΟΥ ΚΑΙ ΑΝΑΖΗΤΗΣΗΣ

Εργασία που υποβλήθηκε από τον
Αντώνη Σιδηρόπουλο
ως μερική εκπλήρωση των απαιτήσεων για την
απόκτηση

Μεταπτυχιακού Διπλώματος Ειδίκευσης στην
Επιστήμη Υπολογιστών

Συγγραφέας: _____

Αντώνης Σιδηρόπουλος
Τμήμα Επιστήμης Υπολογιστών

Εισηγητική Επιτροπή: _____

Χρήστος Νικολάου, Καθηγητής, Επόπτης

_____

Καίτη Χούστη, Αναπληρώτρια Καθηγήτρια, Μέλος

_____

Ευάγγελος Μαρκάτος, Επίκουρος Καθηγητής, Μέλος

Δεκτή:

_____

Πάνος Κωνσταντόπουλος, Καθηγητής
Πρόεδρος Επιτροπής Μεταπτυχιακών Σπουδών

Ηράκλειο, Φεβρουάριος 1999

# ΚΑΤΑΝΕΜΗΜΕΝΟΙ ΜΗΧΑΝΙΣΜΟΙ ΕΥΡΕΤΗΡΙΑΣΜΟΥ ΚΑΙ ΑΝΑΖΗΤΗΣΗΣ

Αντώνης Σιδηρόπουλος

Μεταπτυχιακή Εργασία

Τμήμα Επιστήμης Υπολογιστών
Πανεπιστήμιο Κρήτης

## ΠΕΡΙΛΗΨΗ

Ο ερευνητικός τομέας των «Ψηφιακών Βιβλιοθηκών» ασχολείται με θέματα ταξινόμησης και ανεύρεσης ηλεκτρονικών πληροφοριών με χρήση του διαδικτύου. Μια αρχιτεκτονική που σχεδιάστηκε και αναπτύχθηκε για αυτόν τον σκοπό είναι το dienst. Το dienst χρησιμοποιείται στην συλλογή NCSTRL-ERCIM με τεχνικές αναφορές επιστήμης υπολογιστών.

Στην παρούσα εργασία παρουσιάζουμε επεκτάσεις που έγιναν στο σύστημα dienst καθώς και έναν μηχανισμό παρακολούθησης και περισυλλογής πληροφοριών από την συλλογή NCSTRL-ERCIM, τον dienst spider.

Οι επεκτάσεις στο dienst χωρίζονται σε 3 τμήματα: Επεκτάσεις που έγιναν για χρήση στην παρούσα συλλογή κειμένων της NCSTRL-ERCIM, όπως ιεραρχικοποίηση των κόμβων του συστήματος, πρόσθεση πληροφορίας κόστους αντικειμένων και html μορφή αντικειμένων. Επεκτάσεις ειδικά για την χρήση του dienst στο ΙΤΕ-ΙΠ, όπως αναζήτηση στα Ελληνικά, αυτόματη διαχείριση των εξυπηρετητών του ΙΤΕ/ΙΠ. Και επεκτάσεις ώστε να διευρύνουμε την λειτουργικότητα του dienst ώστε να μπορεί να χρησιμοποιηθεί και για άλλου τύπου συλλογές, όπως γεωγραφικά δεδομένα.

Παράλληλα, αναπτύξαμε το σύστημα “dienst spider”. Το σύστημα αυτό έχει την δυνατότητα να μαζεύει βιβλιογραφικές πληροφορίες από την συλλογή NCSTRL-ERCIM (ή κάποια άλλη συλλογή), να δημιουργεί αρχεία ευρετηρίων για τα δεδομένα αυτά, να δέχεται αιτήσεις αναζήτησης από χρήστες του διαδικτύου και να τους προωθεί στους αντίστοιχους εξυπηρετητές χώρων αποθήκευσης του συστήματος dienst. Η συγκέντρωση της παραπάνω πληροφορίας, μας επιτρέπει να βελτιώσουμε την αναζήτηση σε σχέση με τον μηχανισμό αναζήτησης του dienst, και να έχουμε πολύ καλύτερη απόδοση στις αναζητήσεις.

Επόπτης: Χρήστος Νικολάου
Καθηγητής Επιστήμης Υπολογιστών
Πανεπιστήμιο Κρήτης

# DISTRIBUTED INDEXING AND SEARCHING MECHANISMS

Antonis Sidiropoulos

Master of Science Thesis

Department of Computer Science
University of Crete

## ABSTRACT

The "Digital Libraries" research area is studying the problem of indexing and searching for digital information through the internet. An architecture that was designed and developed for this reason is the Dienst. Dienst system is currently used by the NCSTRL-ERCIM computer science technical report collection.

In this paper, we present the extensions we made for Dienst system and a mechanism for collecting bibliographic records from the NCSTRL-ERCIM document collection, the "dienst spider".

The extensions made on the dienst system, can be grouped in 3 main categories: Extensions for use in the current NCSTRL-ERCIM document collection, Such as hierarchical organization of the system nodes, addition of objects' pricing information and html objects format. Extensions specially made for use of the dienst system at ICS-FORTH, such as searching in Greek, automated administration of the dienst server installed at ICS-FORTH. And finally, extensions to increase the dienst functionality so that it can handle several collection types, such as geographical data.

We also, developed the system called "dienst spider". This system is able to collect bibliographic records from the NCSTRL-ERCIM collection (or any other collection), index these records, accept search requests from internet users, and redirect them to the appropriate dienst repository servers. The collection of the above information, gives us the ability to improve the search and have better performance than the dienst system itself.

Supervisor: Christos Nikolaou
Professor of Computer Science
University of Crete

# ΠΕΡΙΕΧΟΜΕΝΑ

# ΣΧΗΜΑΤΑ

# ΠΙΝΑΚΕΣ

# ΓΡΑΦΗΜΑΤΑ

# ΕΥΧΑΡΙΣΤΙΕΣ

Η παρούσα εργασία δεν θα είχε πραγματοποιηθεί χωρίς την βοήθεια ορισμένων ατόμων στα διάφορα στάδια της πορείας της, τους οποίους και θα ήθελα να ευχαριστήσω.

Αρχικά θα ήθελα να ευχαριστήσω τον επόπτη καθηγητή της εργασίας, κ. Χρήστο Νικολάου, ο οποίος μου έδωσε την δυνατότητα υλοποίησης της συγκεκριμένης εργασίας καθώς και την δυνατότητα συνεργασίας μου και με άλλους ερευνητές του τομέα αυτού. Τον κ. Σαράντο Καπιδάκη, ο οποίος ήταν επιβλέπων καθηγητής της εργασίας αυτής στο μεγαλύτερο τμήμα της υλοποίησης της καθώς και για τις συμβουλές του σε τμήματα που έπρεπε να γίνουν σημαντικές επιλογές. Την κ. Κατερίνα Χούστη και τον κ. Βαγγέλη Μαρκάτο, μέλη της εισηγητικής επιτροπής, για την συμβολή τους στην ολοκλήρωση της εργασίας κάνοντας εκτενή και ουσιώδη σχολιασμό της.

Επίσης για την σημαντική βοήθειά τους θα ήθελα να ευχαριστήσω:

Τον καθ. Νικόλαο Πατρικαλάκη, ο οποίος μου έδωσε την δυνατότητα να εργαστώ μαζί του στο Design Laboratory του MIT πάνω στο αντικείμενο της εργασίας αυτής, και να έρθω σε επαφή με άλλους ερευνητές με ενδιαφέροντα στις ψηφιακές βιβλιοθήκες με γεωγραφικά δεδομένα. Καθώς και για την σημαντική εμπειρία που απέκτησα δουλεύοντας στο συγκεκριμένο εργαστήριο του MIT. Επίσης ευχαριστώ και τους υπόλοιπους εργαζόμενους στο παραπάνω εργαστήριο για την πολύ καλή συνεργασία που είχαμε.

Τον Jakka Sairamesh, ο οποίος είχε την αρχική ιδέα υλοποίησης του dienst spider, και για την συμβολή του στην αρχική σχεδίαση του.

Όλα τα υπόλοιπα μέλη της ομάδας των ΠΛΕΙΑΔΩΝ για το καλό κλίμα συνεργασίας που υπήρχε και ειδικότερα τα μέλη που απασχολούνταν στον τομέα των ψηφιακών βιβλιοθηκών – Σ. Τερζή, Α. Χατζησταματίου, Γ. Καρβουναράκη, Π. Αλεξάκο, Γ. Μαυροειδή, Ι. Μαυροειδή για την συνεργασία τους σε διάφορα τμήματα της εργασίας αυτής.

Επίσης, την Πηνελόπη Κωνσταντά, για την πολύτιμη βοήθειά της στην συγγραφή και διόρθωση του παρόντος κειμένου.

Τέλος, το Τμήμα Επιστήμης Υπολογιστών του Πανεπιστημίου Κρήτης και το Ινστιτούτο Πληροφορικής του Ιδρύματος Τεχνολογίας και Έρευνας για την υλικοτεχνική υποστήριξη που μου παρείχαν.

# Μέρος 1ο:
# Εισαγωγή στις Ψηφιακές Βιβλιοθήκες και στο σύστημα dienst

# *Κεφάλαιο 1*

## 1. Εισαγωγή

Οι βιβλιοθήκες από αρχαιοτάτων χρόνων, αποτελούν πόλους συγκέντρωσης και αναζήτησης πληροφοριών. Με την ανάπτυξη της τεχνολογίας ορισμένες υπηρεσίες ταξινόμησης και αναζήτησης πληροφοριών (ως επί το πλείστον βιβλίων) γίνεται μέσω υπολογιστών. Η χρήση όμως των υπολογιστών στις σύγχρονες βιβλιοθήκες περιορίζεται στην χωροταξική ταξινόμηση και αναζήτηση τίτλων βιβλίων, περιορίζοντας την αναζήτηση πληροφορίας στην αναφορά της φυσικής θέσης ενός βιβλίου (εντύπου) στους κτιριακούς χώρους της βιβλιοθήκης.

Η ραγδαία ανάπτυξη, από την άλλη μεριά, των τηλεπικοινωνιών και του διαδικτυακού ιστού είχαν σαν συνέπεια την άναρχη αύξηση των πόλων συγκέντρωσης των πληροφοριών, την μη ταξινόμησή τους και κατά συνέπεια την ολοένα και δυσκολότερη ανεύρεσή τους. Ο ερευνητικός τομέας των «Ψηφιακών Βιβλιοθηκών» που μόλις πρόσφατα έχει αρχίσει να δραστηριοποιείται, ασχολείται με θέματα ταξινόμησης και ανεύρεσης ηλεκτρονικών πληροφοριών με χρήση του www.

### 1.1. Παραδοσιακές Βιβλιοθήκες

Στις σημερινές σύγχρονες βιβλιοθήκες, η ταξινόμηση και αναζήτηση αντικειμένων (βιβλίων, ταινιών, μουσικών δίσκων κτλ.) γίνεται μέσω υπολογιστή. Η πληροφορία για κάθε αντικείμενο περιορίζεται συνήθως στο όνομα του συγγραφέα, τίτλο, αριθμό ταξινόμησης στη βιβλιοθήκη, θέση (ράφι, δωμάτιο, κτίριο κτλ.). Έτσι ο χρήστης χρησιμοποιεί το υπολογιστικό σύστημα, κάνει την ερώτηση και του επιστρέφονται αναφορές του τόπου (ράφια), που υπάρχουν τα βιβλία που ικανοποιούν την ερώτηση. Έπειτα αναλαμβάνει ο ίδιος να περιηγηθεί στα ράφια με την λίστα των αναφορών, και να ελέγξει ο ίδιος αν πράγματι τον ικανοποιούν τα αποτελέσματα. Αν όχι τότε επαναλαμβάνει την διαδικασία. Βέβαια αυτό που περιγράψαμε είναι η βέλτιστη περίπτωση διότι υπάρχει η περίπτωση κάποιο από τα αποτελέσματα της αναζήτησης να υπάρχει σε διαφορετικό φυσικό χώρο (π.χ. διπλανό κτίριο ή μια άλλη βιβλιοθήκη), οπότε και είναι αναγκασμένος να κάνει μεγάλες μετακινήσεις. Επίσης πάντα υπάρχει η περίπτωση η βιβλιοθήκη να μην είναι καλά οργανωμένη, οπότε και να μην είναι διαθέσιμο υπολογιστικό σύστημα για αναζητήσεις, συνεπώς είναι αναγκασμένος να γνωρίζει περισσότερα για αυτό που αναζητά, ακριβή τίτλο, συγγραφέα κτλ..

## 1.2. Ψηφιακές Βιβλιοθήκες

Με την ανάπτυξη της τεχνολογίας και την δημιουργία ψηφιακών πλέον αντικειμένων (κείμενο, video κτλ.) που περιέχουν πληροφορίες, άρχισε και η οργάνωσή τους σε ηλεκτρονικές συλλογές, ψηφιακές βιβλιοθήκες[1], που είναι ως επί το πλείστον κατανεμημένες (τα αντικείμενα δεν είναι αποθηκευμένα στον ίδιο χώρο).

Στόχος των ψηφιακών βιβλιοθηκών είναι η οργάνωση της πληροφορίας με τρόπο που να είναι εύκολη η αναζήτησή του και η πρόσβαση στα ψηφιακά αντικείμενα της συλλογής.

## 1.3. Ψηφιακές Βιβλιοθήκες και το WWW

Σαν επακόλουθο της ανάπτυξης των τηλεπικοινωνιών και της εμφάνισης του www, ήταν η αναζήτηση τρόπων πρόσβασης μέσω του www στις ψηφιακές βιβλιοθήκες. Έτσι ο χρήστης μπορεί ακόμη και από το σπίτι του, εύκολα και γρήγορα να αναζητήσει και να βρει κάτι που τον ενδιαφέρει. Καθώς όμως ο όγκος πληροφορίας αυξάνεται, γίνεται όλο και περισσότερο δύσκολο για έναν χρήστη να βρει εύκολα χρήσιμη πληροφορία στο διαδίκτυο. Αυτό συμβαίνει διότι στις περισσότερες περιπτώσεις ο όγκος αυτός της πληροφορίας είναι ανοργάνωτος. Ακόμη και αν τα κείμενα μέσα σε έναν κόμβο (site) ακολουθούν κάποια οργάνωση, από τους χιλιάδες κόμβους που υπάρχουν στο διαδίκτυο κανείς δεν χρησιμοποιεί κοινή δομή με κάποιον άλλον. Και ακόμη περισσότερο κανείς δεν προσφέρει έναν ενιαίο τρόπο αναζήτησης με κάποιον άλλον.

### *1.3.1. Spiders και Search Engines του WWW*

Για να αντιμετωπιστεί αυτό το πρόβλημα, σε πρώτη φάση δημιουργήθηκαν οι spiders και οι μηχανές αναζήτησης του WWW. Ο τρόπος λειτουργίας αυτών των συστημάτων είναι πολύ απλός. Οι spiders αναλαμβάνουν να περιηγηθούν στο διαδίκτυο, να μαζέψουν όσα περισσότερα κείμενα μπορούν (ακολουθώντας συνδέσμους από κείμενο σε κείμενο), και να δημιουργήσουν ευρετήρια γι' αυτά. Αυτό βελτίωσε κάπως την κατάσταση, εφόσον οι χρήστες εύκολα ή δύσκολα, μπορούν να βρουν κάποια σχετικά κείμενα με το αντικείμενο που αναζητούν. Στην προσπάθεια αυτή αναπτύχθηκαν πολλές αξιόλογες μηχανές αναζήτησης του διαδικτύου ([I] [II] [III] [IV] [V]). Δεν λύθηκε όμως το πρόβλημα. Η αναζήτηση γίνεται σχεδόν αποκλειστικά με βάση λέξεις κλειδιά που δίνει ο χρήστης, και όχι με νοηματικές ερωτήσεις. Αυτό σε συνδυασμό με το ότι κάποιο κείμενο περιέχει λέξεις όχι μόνο σχετικές με το αντικείμενο του

[1] Αρχικά είχε χρησιμοποιηθεί ο όρος electronic library. Την περίοδο 1991-93 παρατηρείται η τάση να προτιμάται ο όρος Digital Library, πιθανόν ακολουθώντας το αυξημένο ενδιαφέρον της εποχής για digital networks, digital audio, digital video [12].

εγγράφου αλλά και άλλες, έχει ως αποτέλεσμα να επιστρέφονται στον χρήστη χιλιάδες κείμενα, από τα οποία το μεγαλύτερο ποσοστό δεν ενδιαφέρουν τον χρήστη. Αυτό οδηγεί τον χρήστη:

- να δοκιμάζει αρκετά διαφορετικές ερωτήσεις ώστε να μειωθεί όσο γίνεται περισσότερο το πλήθος των κειμένων,

- και να αναγκάζεται να περιηγηθεί σε πολλά άσχετα με την ερώτησή του κείμενα.

Συνεπώς η αναζήτηση γίνεται επίπονη και απαιτεί πολύ περισσότερο χρόνο από το επιθυμητό.

### *1.3.2. Η Συλλογή NCSTRL-ERCIM*

Στον χώρο του διαδικτύου, υπάρχει πληθώρα Ψηφιακών Βιβλιοθηκών για διάφορους τύπους δεδομένων. Περισσότερο διαδεδομένες αυτή τη στιγμή είναι βιβλιοθήκες με κείμενα. Μια από τις πρώτες ψηφιακές συλλογές ήταν αυτή του NCSTRL (Networked Computer Science Technical Report Library). Στην συλλογή αυτή για την ενοποίησή της, χρησιμοποιείται η αρχιτεκτονική Dienst. Η αρχιτεκτονική αυτή και το πρωτόκολλο, σχεδιάστηκε στο Πανεπιστήμιο του Cornell. Τα κυριότερα χαρακτηριστικά αυτής της συλλογής (και συγχρόνως πλεονεκτήματα) είναι ότι:

- Περιέχει τεχνικές αναφορές επιστήμης υπολογιστών
- Βασίζεται πάνω σε ένα πολύ καλά σχεδιασμένο πρωτόκολλο
- Δίνει πρόσβαση στα δεδομένα μέσω www
- Μπορεί να συμμετέχει στην συλλογή οποιοσδήποτε ερευνητικός οργανισμός ή πανεπιστήμιο.
- είναι κατανεμημένη
- Ο κάθε συμμετέχοντας οργανισμός μπορεί να κάνει τις δικές του επεκτάσεις στο σύστημα, ώστε να υποστηρίζει τυχών ιδιαιτερότητες που έχουν τα δεδομένα του. Οι επεκτάσεις αυτές γίνονται τοπικά και δεν επηρεάζουν την ομαλή λειτουργία των υπόλοιπων κόμβων.
- Το λογισμικό είναι γραμμένο σε perl, και άρα μπορεί να λειτουργήσει σε οποιοδήποτε λειτουργικό σύστημα UNIX (χωρίς να χρειάζονται ιδιαίτερες μετατροπές για το κάθε σύστημα).
- η αναζήτηση βασίζεται στην αναζήτηση πάνω σε metadata – άρα είναι γρήγορη.

Πολύ σύντομα, δεδομένου του ρυθμού αύξησης του μεγέθους της συλλογής, στην συλλογή αυτή συμμετείχε και το ERCIM (European Research Consortium for Informatics and Mathematics) με το ΙΤΕ/ΙΠ. Έτσι το 1996, στην συλλογή συμμετείχαν περίπου 20 οργανισμοί (και το ΙΤΕ/ΙΠ), και σήμερα συμμετέχουν 130 με συνεχόμενη αύξηση του

αριθμού. Έτσι πλέον η συλλογή είναι τόσο πλούσια, ώστε όταν κάποιος χρήστης αναζητά δημοσιεύσεις επιστήμης υπολογιστών, θα βρει αυτό που αναζητά σε κάποιον κόμβο του NCSTRL.

Αν και η αρχική χρήση του dienst, είναι για Ψηφιακή Βιβλιοθήκη κειμένων, είναι δυνατή η μετατροπή του και για άλλους τύπους δεδομένων, αρκεί αυτά τα δεδομένα να είναι δομικά απλά. Στο ΙΤΕ/ΙΠ και στην ομάδα των κατανεμημένων συστημάτων, έχει γίνει πολύ αξιόλογη δουλειά, ώστε να είναι δυνατή η χρησιμοποίηση του dienst και για άλλων τύπων δεδομένων όπως: ιατρικές εικόνες, μουσικές συλλογές και βίντεο.

## 1.4. Η χρήση των meta data στις Ψηφιακές Βιβλιοθήκες

Ο τρόπος με τον οποίο γίνεται η αναζήτηση στα δεδομένα μιας συλλογής, είναι ένα άλλο μεγάλο τμήμα έρευνας του χώρου των Ψηφιακών Βιβλιοθηκών. Η αναζήτηση στα ίδια τα δεδομένα θα ήταν πάρα πολύ αργή αλλά και όχι ακριβής. Ένα κείμενο μπορεί να περιέχει λέξεις οι οποίες δεν ανήκουν αναγκαστικά στην ίδια θεματική περιοχή που ανήκει το κείμενο το ίδιο. Συνεπώς μια αναζήτηση με μια λέξη κλειδί, θα επέστρεφε πολλά κείμενα άσχετα με την θεματική περιοχή που ενδιαφέρει τον χρήστη. Για να μειώσουμε στο ελάχιστο αυτήν την περίπτωση, χρησιμοποιούνται τα Meta Data (δεδομένα για τα δεδομένα). Τα Meta Data περιέχουν την περιγραφή του κειμένου (ή γενικότερα του αντικειμένου). Έτσι όλες οι αναζητήσεις γίνονται με βάση τα Meta Data των αντικειμένων που ανήκουν στην εκάστοτε συλλογή. Για παράδειγμα, στην περίπτωση κειμένων τα Meta Data συνήθως περιέχουν όλες τις βιβλιογραφικές πληροφορίες για το κείμενο, δηλαδή τίτλο, συγγραφείς, εκδότη, ημερομηνία δημοσίευσης κ.τ.λ.. Πολλές φορές περιέχουν και μια περίληψη του κειμένου. Έτσι ο χρήστης μπορεί να κάνει την αναζήτησή του με βάση οποιοδήποτε από τα παραπάνω κλειδιά.

## 1.5. Το θέμα της εργασίας

Το επίκεντρο της παρούσας εργασίας είναι το σύστημα dienst και γενικότερα οι ψηφιακές βιβλιοθήκες. Έχοντας λοιπόν το dienst, ως ένα σύστημα ψηφιακής βιβλιοθήκης για τεχνικές αναφορές, προσπαθούμε να το επεκτείνουμε ώστε αυτό να γίνει πληρέστερο και να έχει εφαρμογή και σε άλλους τομείς. Επίσης αναπτύσσουμε παράλληλα άλλα βοηθητικά συστήματα για να πετύχουμε την καλύτερη λειτουργία του. Μια από αυτές τις επεκτάσεις είναι το dienst spider, το οποίο αποτελεί και το μεγαλύτερο τμήμα αυτής της εργασίας.

## 1.6. Η οργάνωση του κειμένου

Το κείμενο, μετά το παρών εισαγωγικό κεφάλαιο χωρίζεται σε 4 τμήματα:

1. **Ψηφιακές Βιβλιοθήκες και το σύστημα dienst:** επισκόπηση ψηφιακών βιβλιοθηκών που έχουν υλοποιηθεί και με μεγαλύτερη λεπτομέρεια εξετάζουμε το σύστημα dienst.

2. **Επεκτάσεις του ΙΤΕ/ΙΠ στο σύστημα dienst:**
   - **i.** γενικές επεκτάσεις-διορθώσεις.
   - **ii.** επεκτάσεις ειδικά για την χρήση του dienst στο ΙΤΕ-ΙΠ.
   - **iii.** επεκτάσεις για χρήση του dienst για γεωγραφικές πληροφορίες.
3. **Το σύστημα dienst spider:**
   - **i.** το σύστημα dienst spider
   - **ii.** αρχιτεκτονική
   - **iii.** θέματα υλοποίησης
   - **iv.** μετρήσεις απόδοσή του
4. **Παραρτήματα:** λεπτομέρειες υλοποίησης και χρήσης τμημάτων της προαναφερθείσης εργασίας.

# *Κεφάλαιο 2*

## 2. Ψηφιακές Βιβλιοθήκες: Επισκόπηση

Στο πρώτο μέρος του κεφαλαίου αυτού γίνεται μια προσπάθεια για παρουσίαση ορισμένων πλευρών του θέματος των Ψηφιακών Βιβλιοθηκών, όπως έχουν αρχίσει να διαμορφώνονται μέσα από συζητήσεις των τελευταίων χρόνων. Από τη συζήτηση θα φανεί ότι μέχρι σήμερα δεν έχει παγιωθεί ένας συγκεκριμένος ορισμός για τις ψηφιακές βιβλιοθήκες, αλλά το θέμα είναι ακόμα ανοιχτό σε απόψεις και ιδέες, και αντλεί εμπειρία από την περιοχή των Παραδοσιακών Βιβλιοθηκών, καθώς και από πολλές περιοχές της επιστήμης υπολογιστών όπως το Information Retrieval, τα User Interfaces κ.α.

Στο κεφάλαιο 2.1.1 θα παρουσιάσουμε ένα μοντέλο του πληροφοριακού περιβάλλοντος των Παραδοσιακών Βιβλιοθηκών, που θα μας δώσει μια εικόνα του παραδείγματος στο οποίο βασίζονται οι Ψηφιακές Βιβλιοθήκες. Στο 2.1.2 θα δούμε μερικές τρέχουσες υποθέσεις για τις Ψηφιακές Βιβλιοθήκες και πώς αυτές πιθανώς περιορίζουν άσκοπα το εύρος των Ψηφιακών Βιβλιοθηκών. Στο 2.1.3 θα εξετάσουμε αν το WWW είναι μια ψηφιακή βιβλιοθήκη, και στο επόμενο κεφάλαιο θα συγκρίνουμε τις παραδοσιακές και ψηφιακές βιβλιοθήκες ως προς τις υπηρεσίες που παρέχει η κάθε μια.

Στο δεύτερο μέρος 2.2, παρουσιάζουμε τις κυριότερες εργασίες σε ψηφιακές βιβλιοθήκες που υπάρχουν στις Ηνωμένες Πολιτείες Αμερικής.

### 2.1. Ψηφιακές Βιβλιοθήκες

#### *2.1.1. Μοντέλο του Πληροφοριακού Περιβάλλοντος των Παραδοσιακών Βιβλιοθηκών*

Οι τέσσερις βασικές δραστηριότητες που λαμβάνουν χώρα σε μια παραδοσιακή βιβλιοθήκη, όπως φαίνονται από την πλευρά του χρήστη, είναι οι εξής: ερωτήσεις στο βιβλιοθηκάριο, ψάξιμο στους διαθέσιμους καταλόγους, αναζήτηση στην ανοιχτή συλλογή, και τέλος επεξεργασία των πληροφοριών που αποκτήθηκαν.

Ας υποθέσουμε ότι ο χρήστης αρχίζει την αναζήτησή του συμβουλευόμενος το βιβλιοθηκάριο. Μπορούμε να περιγράψουμε τη διαδικασία που συμβαίνει προκειμένου να μπορέσει ο βιβλιοθηκάριος να οδηγήσει το χρήστη στις κατάλληλες πηγές ως μια διαδικασία σχηματισμού και χρήσης γνωσιακών μοντέλων. Έτσι, ο βιβλιοθηκάριος πρέπει κατ' αρχήν να σχηματίσει ένα

μοντέλο του χρήστη, π.χ. ποιο είναι το επίπεδο γνώσης του σχετικά με τη βιβλιοθήκη, ή κατά πόσο καταλαβαίνει τη χρήση των θεματικών καταλόγων.

Κατόπιν, πρέπει να σχηματίσει ένα μοντέλο των πληροφοριακών αναγκών του χρήστη, και πιθανόν των δυνατοτήτων του για επεξεργασία των πληροφοριών (π.χ. μπορεί να έχει πολύ καλή γνώση μιας θεματικής περιοχής και να αναζητεί ένα συγκεκριμένο σημείο της).

Συγκρίνοντας αυτά τα μοντέλα με το μοντέλο των resources της βιβλιοθήκης, θα μπορέσει να καθορίσει ένα σύνολο από ενέργειες που θα οδηγήσουν το χρήστη στην κατάλληλη πληροφορία.

### *2.1.2. Τρέχουσες υποθέσεις-περιορισμοί για τις Ψηφιακές Βιβλιοθήκες*

Η εμπειρία που αντλείται από τις παραδοσιακές βιβλιοθήκες εκτός από καθοδηγητική, μπορεί να αποβεί πολλές φορές περιοριστική. Παρακάτω παρατίθενται τρεις τέτοιες υποθέσεις για τις ψηφιακές βιβλιοθήκες, που, όπως έχει δείξει η τρέχουσα πρακτική, έχουν περιορίσει τις δυνατότητες των συστημάτων ψηφιακών βιβλιοθηκών, σε σχέση με αυτές που θα μπορούσαν ιδανικά να παρέχουν.

***1η Υπόθεση:*** *Οι βιβλιοθήκες περιέχουν στατικά και μόνιμα έγγραφα.*

Δύο σημαντικά χαρακτηριστικά των εγγράφων ή των συλλογών εγγράφων είναι

I. η συχνότητα αλλαγής τους: ένα κείμενο μπορεί να αλλάζει με το χρόνο, π.χ. οι τηλεφωνικοί κατάλογοι

II. η διάρκειά τους: ένα κείμενο είναι χρήσιμο για ένα μεγάλο ή μικρό χρονικό διάστημα. Τα βιβλία λ.χ., έχουν μεγάλη διάρκεια ζωής, ενώ ένα άρθρο για τα κοσμικά ίσως μόνο μερικές μέρες.

Οι δύο αυτές διαστάσεις είναι ανεξάρτητες μεταξύ τους. Για παράδειγμα, το άρθρο μπορεί στη διάρκεια της μικρής ζωής του να μην αλλάζει καθόλου, ενώ ένα βιβλίο με συνταγές μαγειρικής να υπόκειται σε συνεχείς αλλαγές.

Παρόλο που η εικόνα της βιβλιοθήκης που έχουμε οι περισσότεροι παραπέμπει σε στατικά και μόνιμα έγγραφα, μια τέτοια υπόθεση είναι αναληθής στις ψηφιακές βιβλιοθήκες. Η ευκολία που παρέχει η ψηφιακή τεχνολογία για αλλαγές στα κείμενα, υπαγορεύει μάλλον το αντίθετο.

***2η Υπόθεση:*** *Οι ψηφιακές βιβλιοθήκες βασίζονται αποκλειστικά στην ψηφιακή τεχνολογία.*

Αν φανταστούμε τις ψηφιακές βιβλιοθήκες του μέλλοντος όπου τα πάντα είναι ψηφιοποιημένα, ίσως αυτή η υπόθεση να μην είναι άστοχη. Όμως είναι πολύ πιθανόν, αν όχι σίγουρο, ότι για να μπορέσουν οι υπάρχουσες βιβλιοθήκες να μεταφέρουν όλο τους το υλικό σε ψηφιακή μορφή θα περάσει ακόμα αρκετός καιρός. Σε αυτό το διάστημα, θα πρέπει είτε να δεχτούμε την ύπαρξη από τη μια ψηφιακών βιβλιοθηκών και από την άλλη των παραδοσιακών, ως δύο διαφορετικούς κόσμους, είτε θα πρέπει να δούμε τις βιβλιοθήκες υπό μια διαφορετική οπτική γωνία, όπου ψηφιακά και μη δεδομένα θα συνυπάρχουν, και στο χρήστη θα δίνεται ένας κατά το δυνατόν ομοιόμορφος τρόπος πρόσβασης στα περιεχόμενα.

***3η Υπόθεση:*** *Η ψηφιακή βιβλιοθήκη χρησιμοποιείται από άτομα που δουλεύουν μόνα τους.*

Στην πραγματικότητα, ούτε ο βιβλιοθηκάριος είναι ο μόνος υπεύθυνος για την επιλογή του υλικού και τη συγκρότηση της βιβλιοθήκης, ούτε ο ερευνητής απομονώνεται σε ένα γραφείο για να περιοριστεί στις πηγές που θα βρει ο ίδιος. Η αναζήτηση πληροφορίας περιλαμβάνει συνεργασία πολλών ατόμων, αν και πολλές φορές παραλείπουμε να το αναγνωρίσουμε.

Τα παραπάνω υποδεικνύουν ότι η υποστήριξη για επικοινωνία και συνεργασία είναι πολύ σημαντική, για ένα περιβάλλον αναζήτησης πληροφορίας, όπως είναι οι βιβλιοθήκες. Αν η θεαματική εξάπλωση της χρήσης του διαδικτύου έχει δείξει κάτι, είναι ότι οι χρήστες χρειάζονται επικοινωνία τουλάχιστον όσο χρειάζονται πρόσβαση σε πληροφορία [13].

### *2.1.3. Το WWW ως ψηφιακή βιβλιοθήκη*

*Μπορούμε να υποστηρίξουμε ότι το WWW λειτουργεί ως μια ψηφιακή βιβλιοθήκη, χωρίς μάλιστα να έχει τους παραπάνω περιορισμούς;*

Μάλλον το WWW και το διαδίκτυο αποτελούν μια υποδομή από τεχνικές για την ανάπτυξη ψηφιακών βιβλιοθηκών, αλλά δεν αποτελούν από μόνα τους μια ψηφιακή βιβλιοθήκη. Ένα σημείο στο οποίο διαφέρει, είναι ότι το υλικό που υπάρχει στο Web δεν αποτελεί μια συλλογή, τουλάχιστον με την έννοια ενός επιλεγμένου συνόλου αντικειμένων, οργανωμένου για μια συγκεκριμένη χρήση. Επιπλέον, λείπουν οι κρίσιμες υπηρεσίες, όπως η καταλογογράφηση, με την οποία οι συλλογές σταθεροποιούνται και γίνονται συνεχώς και πιο εύχρηστες.

Τι επιπλέον χρειάζεται για να προσεγγίσει το Web την έννοια της ψηφιακής βιβλιοθήκης; Παρακάτω αναφέρουμε τρία σημαντικά ζητήματα:

1. Versioning. Εάν θεωρήσουμε σαν δεδομένο ότι οι μελλοντικές συλλογές κειμένων θα περιέχουν υλικό το οποίο θα αλλάζει μορφή με το χρόνο, θα χρειαστούμε περισσότερο αναπτυγμένες τεχνικές για ονοματολογία, εύρεση και χειρισμό εκδόσεων (versions).

2. Εργαλεία για επικοινωνία και συνεργασία. Μερικές από τις υπηρεσίες που θα πρέπει να υποστηρίζονται είναι διαμοιραζόμενες σημειώσεις στα κείμενα, συντήρηση τοπικών υποσυλλογών, όπως και υποστήριξη για την απαραίτητη επικοινωνία ανάμεσα στους ανθρώπους στα πλαίσια μιας εργασίας.

3. Υπηρεσίες. Εκτός από νέες τεχνολογίες, νέες πρακτικές είναι πολύ πιθανόν να χρειαστούν για την ανάπτυξη ψηφιακών βιβλιοθηκών. Ήδη έρευνες ασχολούνται με το κατά πόσον οι υπάρχουσες πρακτικές των παραδοσιακών βιβλιοθηκών είναι επαρκείς, ή νέες είναι απαραίτητες για την ταξινόμηση και συντήρηση ψηφιακού υλικού. [19]

### *2.1.4. Σύγκριση Παραδοσιακών και Ψηφιακών Βιβλιοθηκών με βάση τις υπηρεσίες*

#### *2.1.4.1. Υπηρεσίες που χάνονται.*

Ήδη είναι δυνατόν για οποιονδήποτε συγγραφέα να τοποθετήσει τη δουλειά του στο Web, παρακάμπτοντας τον εκδότη και χρονοβόρες διαδικασίες που μπορούν να διαρκέσουν μήνες. Το γεγονός αυτό όμως εισάγει νέα ζητήματα, όπως αυτό του ελέγχου του λεξιλογίου (vocabulary control), ή της αξιοπιστίας του υλικού που κυκλοφορεί. Από την άλλη μεριά, χάνει τα συγγραφικά δικαιώματα (copyright).

Άλλη υπηρεσία που παρουσιάζει αρνητική δυναμική, είναι αυτή του volume editor. Η κατηγοριοποίηση κειμένων και δημιουργία συλλογών από σχετικά μεταξύ τους άρθρα ήδη σε κάποιον βαθμό επιτυγχάνεται από αυτόματα συστήματα, όπως το Harvest και το Aliweb. [18]

#### *2.1.4.2. Νέες υπηρεσίες*

Για την υποστήριξη των νέων συστημάτων βιβλιοθήκης, απαραίτητη είναι η χρέωση. Νέες μορφές on-line χρέωσης είναι απαραίτητες, που θα υποστηρίζουν τη λειτουργία και ανάπτυξη των υπηρεσιών παροχής πληροφοριών, αλλά και θα παρέχουν στους συγγραφείς την κατάλληλη αμοιβή για την εργασία τους. Ένα μεγάλο ζήτημα που ανακύπτει είναι αυτό της προστασίας της πνευματικής ιδιοκτησίας, στα πλαίσια ενός τέτοιου περιβάλλοντος. Ήδη έχει αρχίσει να γίνεται λόγος για επαναπροσδιορισμό του όρου και των ιδιοτήτων του, ώστε να ανταποκρίνεται στις ανάγκες νέας πραγματικότητας.

Μια ιδιαίτερη παράσταση πληροφορίας αποτελούν οι εικόνες, είτε αυτές είναι φωτογραφίες, χάρτες, γράφοι κλπ. Η προσέγγιση που έχει ακολουθηθεί μέχρι σήμερα είναι να γίνεται αναζήτηση μιας εικόνας μέσο μετα-πληροφορίας, όπως είναι ο τίτλος, η ημερομηνία, ή το θέμα της εικόνας. Με την διαθεσιμότητα νέων εργαλείων, αρχίζει να γίνεται εφικτή η αναζήτηση εικόνων με βάση το περιεχόμενό τους. Ιδανικά, θα θέλαμε αφ' ενός να μπορούν οι εικόνες να αναλύονται, ώστε να αναγνωρίζουμε αντικείμενα που περιέχουν, και αφ' ετέρου να μπορεί ο

χρήστης να περιγράφει με γραφικό τρόπο μια εικόνα, και το σύστημα να επιστρέφει όσες μοιάζουν.

Τέλος, αρχίζει να γίνεται εφικτή η χρήση και υποστήριξη "δυναμικών βιβλίων". Πολλά βιβλία γράφονται από πολλούς συγγραφείς. Άλλα πάλι αποτελούνται από ξεχωριστά κεφάλαια και είναι οργανωμένα έτσι ώστε ένας καθηγητής ή σπουδαστής να μπορεί να παραλείπει όσα θεωρεί άσχετα. Εάν διευθετήσουμε το ζήτημα του copyright, είναι δυνατόν να προχωρήσουμε στη χρήση βιβλίων με έναν ή περισσότερους προδιαγεγραμμένους ή όχι σκελετούς, το περιεχόμενο των οποίων θα παράγεται δυναμικά, ώστε σε κάθε στιγμή να περιλαμβάνουν την τελευταία έκδοση, με τις τελευταίες πληροφορίες.

## 2.2. Υπάρχουσες Ψηφιακές Βιβλιοθήκες

Παρακάτω παρουσιάζουμε τις κυριότερες προσπάθειες για ανάπτυξη ψηφιακών βιβλιοθηκών, και τα σημεία στα οποία η κάθε μια επικεντρώνεται.

### *2.2.1. Informedia Digital Video Library*

http://fuzine.mt.cs.cmu.edu/im/informedia.html *[XIII]*

Το project έχει ως σκοπό τη ανάπτυξη νέων τεχνικών για ανάκτηση δεδομένων από βιβλιοθήκες ψηφιακών video, με βάση το περιεχόμενο (full-content search and retrieval). Αναπτύσσεται από την WQED Pittsburgh, με τη συνεργασία πολλών εταιριών. Χρησιμοποιούν τεχνικές αναγνώρισης φωνής (Sphinx-II speech recognizer) για τη δημιουργία *περιγραφών* σε σκηνές, οι οποίες χρησιμοποιούνται κατά την ανάκτηση.

Επιπλέον, χρησιμοποιούν εργαλεία για τμηματοποίηση video clips με βάση το περιεχόμενο, ώστε αυτόματα να χωρίζουν διαφορετικές σκηνές του video.

### *2.2.2. Stanford Digital Library*

http://www.diglib.stanford.edu/diglib *[XII]*

Στο Stanford ασχολούνται με την ιδέα της ενοποιημένης ψηφιακής βιβλιοθήκης (integrated digital library), που θα παρέχει δυνατότητες πρόσβασης σε δεδομένα που θα κυμαίνονται από προσωπικές σημειώσεις ως μεγάλες θεματικές βιβλιοθήκες. Για το σκοπό αυτό αναπτύσσουν υψηλού επιπέδου πρωτόκολλα.

Το σύστημα όπως σχεδιάζεται στο υψηλότερο επίπεδο ονομάζεται InfoBus, και φιλοδοξεί να αποτελέσει ένα ευρέως αποδεκτό "μέσο" σύνδεσης όλων των προσπαθειών για ψηφιακές βιβλιοθήκες.

### *2.2.3. UC Berkeley Digital Library*

http://http.cs.berkeley.edu/~wilensky *[XI]*

Στο Berkeley γίνεται δουλειά πάνω στην υποδομή που είναι απαραίτητη για μια ψηφιακή βιβλιοθήκη. Ασχολούνται με αυτοματοποιημένο ευρετηριασμό και "έξυπνη" ανάκτηση δεδομένων, με χρήση μηχανικής όρασης (OCR), και τεχνικές αναγνώρισης φυσικής γλώσσας (NLP). Επενδύουν στην τεχνολογία των κατανεμημένων βάσεων δεδομένων, σε client/server αρχιτεκτονικές βασισμένες στο Z39.50 και την SQL3 (ZQL), καθώς και σε θέματα user interfaces, δουλεύοντας πάνω σε μια βιβλιοθήκη με περιβαλλοντολογικές αναφορές από την περιοχή της California (The California Environment).

### *2.2.4. Alexandria Digital Library*

http://alexandria.sdc.ucsb.edu *[X]*

Το project αυτό έχει ως σκοπό την παροχή ενός κατανεμημένου συστήματος βιβλιοθήκης το οποίο θα περιέχει γεωγραφικά δεδομένα. Αναπτύσσονται τεχνικές για ευρετηριασμό δεδομένων με βάση τις χωρικές τους ιδιότητες, και για την ολοκλήρωση του συστήματος χρησιμοποιούνται γνωστές τεχνικές για user interfaces και αποθήκευση δεδομένων, καθώς και εμπορικά συστήματα DB (Ingest), ως υπόβαθρο.

### *2.2.5. Illinois Digital Library*

http://www.grainger.uiuc.edu/dli *[IX]*

Στο Illinois αναπτύσσουν ένα πρωτότυπο σύστημα ψηφιακής βιβλιοθήκης το οποίο θα χρησιμοποιείται αρχικά στα πλαίσια του πανεπιστημίου, και έχει ως σκοπό να αποτελέσει μέσο μελέτης των αναγκών για μια ψηφιακή βιβλιοθήκη, όσον αφορά τις ιδιαιτερότητές της και τα μοντέλα των χρηστών που τη χρησιμοποιούν. Έρευνα γίνεται πάνω σε θέματα σημασιολογικής ταξινόμησης με χρήση θησαυρών.

### *2.2.6. University of Michigan Digital Library Project*

http://www.sils.umich.edu/UMDL/HomePage.html *[VIII]*

Στο Michigan, όπως και στο Illinois, χρησιμοποιούν ένα ακόμα αναπτυσσόμενο κατανεμημένο σύστημα ψηφιακής βιβλιοθήκης, για μελέτη των αναγκών που προκύπτουν.

Το σύστημα είναι βασισμένο στην ιδέα των software agents, και αποτελείται από UI agents (για διάλογο με το χρήστη και καθορισμό των περιοχών αναζήτησης, οι οποίες τον ενδιαφέρουν), mediation agents οι οποίοι διευθύνουν την αναζήτηση στα κατανεμημένα τμήματα του συστήματος, και collection agents, οι οποίοι είναι υπεύθυνοι για αναζήτηση μέσα σε μια συγκεκριμένη αυτόνομη συλλογή.

Η πρωτοτυπία του συστήματος βασίζεται στο DIRECT, ένα interface για αναζήτηση και περιήγηση σε υλικό 40 επιστημονικών περιοδικών.

### *2.2.7. MEL*

Master Environmental Library

*http://mel.dmso.mil/* [XVII]

Ο σκοπός του συστήματος αυτού είναι να παρέχει έναν ενιαίο τρόπο αναζήτησης σε περιβαλλοντικά δεδομένα περιλαμβανομένων γεωγραφικών δεδομένων, μοντέλων, αλγορίθμων και κειμένων. Στην παρούσα φάση, το σύστημα παρέχει στον χρήστη την δυνατότητα αναζήτησης στα meta data διαφόρων συλλογών με ενιαίο τρόπο μέσω του διαδικτύου και την προώθησή του στην πηγή από όπου μπορεί να αποκτήσει αυτά τα δεδομένα.

### *2.2.8. EOSDIS*

Earth Observing System (EOS) Data and Information System (EOSDIS)

*http://spsosun.gsfc.nasa.gov/* [XVIII]

Το Σύστημα παρατήρησης της γης δεδομένων και πληροφοριών (EOSDIS), έχει αναπτυχθεί από την NASA ώστε να γίνουν διαθέσιμα στο ευρύ κοινό, εύκολα, δεδομένα που έχουν σχέση με την γη και την παρατήρηση αυτής. Το σύστημα βρίσκεται ακόμη υπό ανάπτυξη με την αρχική έκδοση 0 σε λειτουργία.

Αυτή τη στιγμή, το EOSDIS, περιέχει παλαιότερα και νέα δεδομένα από τις παρατηρήσεις της NASA. Κατά την διάρκεια της περιόδου EOS, η οποία άρχισε με την εκτόξευση ενός TRMM δορυφόρου το 1997, το EOSDIS (θα) ελέγχει δορυφόρους και εργαλεία μετρήσεων και θα κάνει διαθέσιμα τα δεδομένα που προκύπτουν, για παράδειγμα μοντέλα κλίματος μετά από προσομοιώσεις.

Το EOSDIS περιέχει ένα ευρύ φάσμα πληροφοριών και δεδομένων ώστε να υποστηρίξει την αμερικανική και την παγκόσμια επιστημονική κοινότητα. Επίσης παρέχει πληθώρα υπηρεσιών, οι οποίες απευθύνονται και σε διαφορετικούς τύπους χρηστών (απλοί χρήστες, επιστήμονες, ή επιστήμονες συνεργαζόμενοι με την NASA

Η επικοινωνία του χρήστη με το σύστημα μπορεί να γίνει με δύο τρόπους: μέσω web ή χρησιμοποιώντας το πρόγραμμα[2] του EOSDIS για απευθείας πρόσβαση στις βάσεις δεδομένων.

### *2.2.9. DOD*

*http://dods.gso.uri.edu/DODS* [XIX]

Πρόκειται για ένα τελείως διαφορετικό σύστημα από τα υπόλοιπα που εξετάζουμε σε αυτό το κεφάλαιο, αλλά το αναφέρουμε διότι και αυτό έχει άμεση σχέση με κατανεμημένα συστήματα δεδομένων. Ουσιαστικά πρόκειται για μια βιβλιοθήκη για την υλοποίηση ενός remote file system για χρήση από το πακέτο MathLab.

Είναι γνωστό ότι στην επιστημονική κοινότητα χρησιμοποιούνται διάφορα προγράμματα (και κυρίως το MathLab) για την υλοποίηση προσομοιώσεων και υπολογισμών. Για τους υπολογισμούς αυτούς χρησιμοποιούνται τεράστιες ποσότητες δεδομένων (εφόσον στόχος είναι η προσομοίωση, όσο γίνεται πλησιέστερα στην πραγματικότητα). Όταν ένα σύνολο δεδομένων είναι διαθέσιμο «κάπου», αλλά θέλει να το χρησιμοποιήσει κάποιος χρήστης από έναν απομακρυσμένο σταθμό, τότε αυτός είναι υποχρεωμένος να αντιγράψει τα δεδομένα στον δικό του υπολογιστή. Αν μάλιστα τα δεδομένα αυτά ανανεώνονται, πρέπει να επαναλαμβάνει την ίδια διαδικασία. Έτσι, για να διευκολυνθεί η κατάσταση, έγινε η υλοποίηση μιας βιβλιοθήκης για την υποστήριξη ενός απομακρυσμένου συστήματος αρχείων (remote file system) μέσω του διαδικτύου. Έτσι, ο χρήστης που θέλει να κάνει τους υπολογισμούς, απλά ορίζει «που» βρίσκονται τα δεδομένα που χρειάζεται, και το σύστημα αναλαμβάνει να μεταφέρει τα απαραίτητα κομμάτια για τον υπολογισμό.

### *2.2.10. Unidata*

Άλλη μια ψηφιακή βιβλιοθήκη για γεωγραφικά δεδομένα. Βέβαια το συγκεκριμένο σύστημα, στερείται βασικών λειτουργιών μιας ψηφιακής βιβλιοθήκης, εφόσον δεν υποστηρίζει ούτε αναζήτηση ούτε περιήγηση στα δεδομένα (που είναι βασικές λειτουργίες μιας ψηφιακής βιβλιοθήκης).

[2] Το πρόγραμμα αυτό εγκαταστάθηκε πειραματικά στο deslab.mit.edu.

Σε αυτό το σύστημα, ο χρήστης ζητά πρόσβαση σε ένα συγκεκριμένο τύπο δεδομένων, και μετά από αυτό δημιουργείται ένα data stream στο οποίο έχει πρόσβαση να διαβάσει τα δεδομένα αυτά. Τα περισσότερα δεδομένα αυτού του συστήματος είναι real-time μετρήσεις.

# *Κεφάλαιο 3*

## 3. Το Σύστημα Dienst

Το σύστημα Dienst χρησιμοποιείται αυτήν την στιγμή από την συλλογή NCSTRL (Networked Computer Science Technical Report Library) ως το σύστημα για την ενοποίηση των βιβλιοθηκών των εκδοτών που συμπεριλαμβάνονται σε αυτή τη συλλογή. Στην ίδια συλλογή στο διαδίκτυο συμμετέχει και το ERCIM (European Research Consortium for Informatics and Mathematics) .Έτσι αυτή τη στιγμή στην κοινή συλλογή NCSTRL-ERCIM για τεχνικές αναφορές επιστήμης υπολογιστών συμμετέχουν πάνω από 120 εκδότες ή ερευνητικά ινστιτούτα, ανάμεσα στα οποία είναι και το Ινστιτούτο Πληροφορικής (ΙΠ) του Ιδρύματος Τεχνολογίας και Έρευνας (ΙΤΕ)[3].

Παρόλη την ευρεία αποδοχή του dienst, δεν μπορούμε να πούμε ότι από μόνο του αποτελεί μια ψηφιακή βιβλιοθήκη. Το dienst απλά παρέχει ένα κοινό πρωτόκολλο επικοινωνίας των κατά τόπους συλλογών με τεχνικές αναφορές. Έτσι είναι δυνατή η εμφάνιση προς τον χρήστη μιας ενιαίας συλλογής. Αυτός ακριβώς είναι ο στόχος του dienst, να ενοποιήσει συλλογές, οργανωμένες σε ετερογενή συστήματα με ένα κοινό πρωτόκολλο επικοινωνίας.

Για να γίνει αυτό σαν βάση του πρωτοκόλλου χρησιμοποιείται το Web, το οποίο είναι εδώ και λίγα χρόνια ευρέως αποδεκτό. Έτσι το πρωτόκολλο του dienst υλοποιείται κάτω από τα URLs (Uniform Resource Locators) (Βλέπε Παράρτημα ΙΙΙ. )

Επιγραμματικά, οι στόχοι για την δημιουργία του dienst και συγχρόνως τα πλεονεκτήματα που μας προσφέρει είναι τα παρακάτω [9]:

- Έχουμε την υλοποίηση μας κατανεμημένης ψηφιακής βιβλιοθήκης στο WWW.
- Οι χρήστες (κυρίως ερευνητές) μπορούν να χρησιμοποιούν γνωστά εργαλεία (Web Browsers) για να αναζητήσουν, να περιηγηθούν, να διαβάσουν ή να μεταφέρουν στον υπολογιστή τους τεχνικές αναφορές.
- Οι συγγραφείς μπορούν πολύ εύκολα να δημοσιεύσουν τα συγγράμματά τους και να αποκτήσουν ένα ευρύ αναγνωστικό κοινό.

[3] Μάλιστα το ΙΤΕ/ΙΠ ήταν ένα από τα πρώτα ινστιτούτα που συμμετείχαν στην συλλογή.

- Οι εκδότες (Πανεπιστήμια, Ινστιτούτα, κ.τ.λ.) αποκτούν ένα αποτελεσματικό σύστημα για να δημοσιεύουν τις τεχνικές αναφορές.

## 3.1. Η Συλλογή NCSTRL – ERCIM

Όπως αναφέρθηκε πριν, το σύστημα dienst χρησιμοποιείται για την συλλογή των τεχνικών αναφορών επιστήμης υπολογιστών που έχει δημιουργήσει η NCSTRL και το ERCIM. 130 εκδότες ή ερευνητικά ινστιτούτα[4] (Βλέπε Παράρτημα ΙΙΙ. ) σε Η.Π.Α. και Ευρώπη χρησιμοποιούν το dienst (ή και το dienst) για την δημοσίευση των τεχνικών αναφορών. Το σύστημα έχει πλέον γίνει ευρείας αποδοχής αφού γίνονται στο σύστημα περίπου 5.000 έως 10.000 αναζητήσεις ημερησίως.

## 3.2. Αρχιτεκτονική του Συστήματος Dienst

Το πρωτόκολλο επικοινωνίας του συστήματος είναι βασισμένο στην μορφή URL (Uniform Resource Locator) . (Βλέπε Παράρτημα Ι). Έτσι κάθε επικοινωνία με το σύστημα γίνεται με την μορφή αιτήσεων HTTP. Η κάθε αίτηση προς το σύστημα παριστάνεται με ένα URL. Για καλύτερη οργάνωση του πρωτοκόλλου, οι υπηρεσίες που προσφέρονται έχουν ιεραρχηθεί με βάση την υπηρεσία στην οποία ανήκουν. Έτσι έχουμε τις παρακάτω υπηρεσίες:

- Repository Service
- Index Service
- UI Service
- Meta Service

Κάθε υπηρεσία από τις παραπάνω προσφέρει ένα αριθμό από εντολές που αντιστοιχούν στον τύπο της υπηρεσίας. Έτσι ο τρόπος με τον οποίο κατασκευάζονται οι αιτήσεις (υπό μορφή URL) είναι:

| *http://dienst_host:dienst_port/Dienst/{Service}/{Version}/{Command}/{Args}?{Optional args}* |
|---|

Πίνακας 1. URL αίτησης στο dienst

[4] Την 18η Αυγούστου 1998.

dienst_host: Το όνομα του υπολογιστή στον οποίο λειτουργεί ο dienst server που απευθύνεται η αίτηση.

dienst_port: Το port στο οποίο δέχεται αιτήσεις ο dienst server

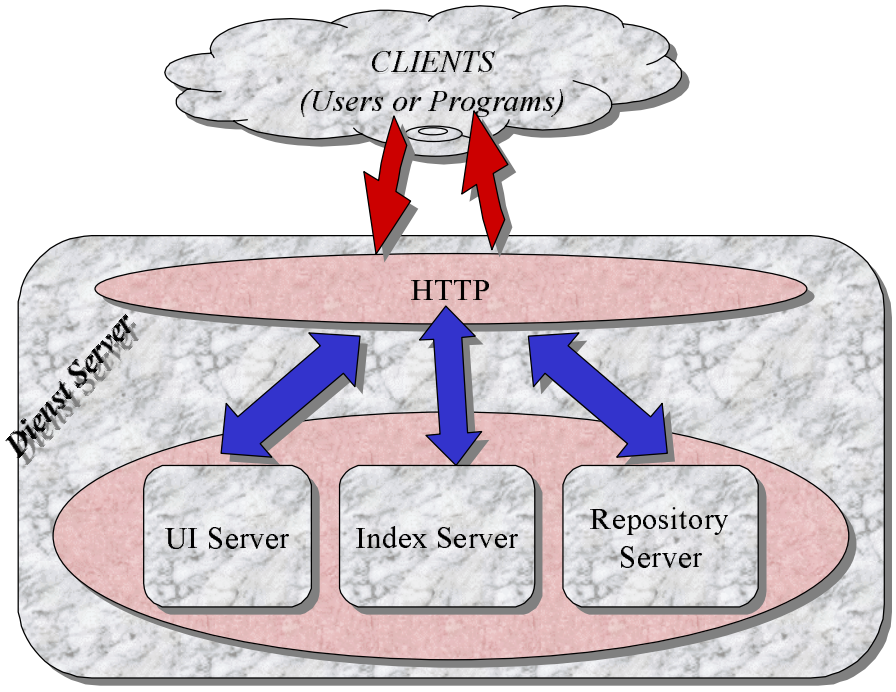
Service: Η υπηρεσία στην οποία απευθύνεται η αίτηση

Command: Η εντολή προς εκτέλεση

Args, Optional Args: Παράμετροι για την εντολή προς εκτέλεση

### *3.2.1. Υπηρεσίες του Συστήματος*

Κάθε υπηρεσία του συστήματος, υλοποιείται από διαφορετικό εξυπηρετητή (Σχήμα 1). Αν και στην υλοποίηση του συστήματος όλοι οι servers τρέχουν στον ίδιο υπολογιστή σαν τμήματα μιας και μόνο διεργασίας (process), ο σχεδιασμός είναι τέτοιος ώστε να μπορούν να λειτουργούν και σε διαφορετικούς υπολογιστές.

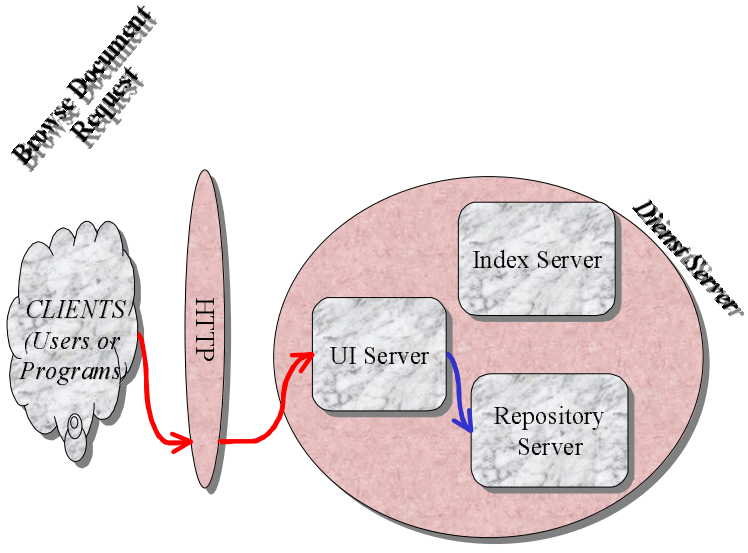



Σχήμα 1. Διανομή των αιτήσεων στον αντίστοιχο εξυπηρετητή.

Οι κύριοι εξυπηρετητές του συστήματος (οι οποίοι αντιστοιχούν και στην ανάλογη υπηρεσία) είναι οι παρακάτω:

- **Repository Server**: Εδώ αποθηκεύονται τα αντικείμενα, και προς αυτόν τον server κάνουμε αίτηση ώστε να πάρουμε ένα αντικείμενο σε κάποια μορφή (π.χ. το postscript αρχείο ενός κειμένου) ή κάποια τμήματα του αντικειμένου (π.χ. μια σελίδα από ένα κείμενο)
- **Index Server:** Περιέχει τα αρχεία ευρετηριασμού (index files) για τα αντικείμενα του αντίστοιχου Repository Server. Επίσης μπορεί να εξυπηρετήσει αιτήσεις που αναπαριστούν επερωτήσεις για αναζήτηση με βάση λέξεις κλειδιά.
- **UI Server:** Μορφοποιεί τα δεδομένα και τα διανέμει στους χρήστες υπό μορφή HTML, ώστε να είναι εύκολα αναγνώσιμα μέσα από τα γνωστά εργαλεία περιήγησης του διαδικτύου.
- **Meta Server:** Παρέχει έναν κατάλογο με τους υπάρχοντες εξυπηρετητές στην συλλογή, και όλα τα απαραίτητα στοιχεία για επικοινωνία με αυτούς. Σε όλη την συλλογή, υπάρχει μόνο ένας κύριος Meta Server (master meta server), ο οποίος είναι εγκατεστημένος στο Cornell (το οποίο μπορούμε να πούμε ότι είναι το κέντρο ελέγχου του συστήματος). Επίσης για να αποφεύγεται η συχνή επικοινωνία όλων των εξυπηρετητών dienst με αυτόν του Cornell, υπάρχουν και οι meta servers περιοχών.

Όλες οι αιτήσεις των χρηστών γίνονται μέσω του UI Server (εκτός των αιτήσεων για download ολόκληρων κειμένων). Βλέπε Σχήμα 2, Σχήμα 4, Σχήμα 5, Σχήμα 6.




Σχήμα 2. Αίτηση περιήγησης σε κείμενο από χρήστη.

### *3.2.2. Λειτουργία του Συστήματος*

Το σύστημα, όπως φαίνεται και στο Σχήμα 2, λειτουργεί σε συνεργασία με ένα web server. Λεπτομέρειες για το θέμα αυτό βρίσκονται στην παράγραφο IV.

### *3.2.3. Meta Data του Dienst*

Τα meta data του dienst βασίζονται στο RFC-1357. Το RFC-1357 είναι ένα πρωτόκολλο που σχεδιάστηκε αρχικά για μεταφορά βιβλιογραφικών πληροφοριών μέσω e-mail, αλλά τελικώς, όπως βλέπουμε βρήκε και άλλες χρήσεις, όπως αυτή του dienst για την αποθήκευση των meta data αρχείων.

Πρόκειται για μια πολύ απλή μορφή. Τα στοιχεία είναι απλά πεδία (tagged fields), και δεν επιτρέπονται φωλιασμένες (nested) αναφορές. Τα πεδία που χρησιμοποιούνται συνήθως από το dienst είναι τα εξής:

| | | | |
|---|---|---|---|
| <M> | BIB-VERSION:: | CS-TR_v2.0 | |
| <M> | ID:: | ercim.ics//TR96-128 | (docid) |
| <M> | ENTRY:: | January 15, 1996 | (month day, year) |
| <O> | ORGANIZATION:: | ICS - FORTH | (free-text) |
| <M> | TITLE:: | A Spider for Dienst | (free-text) |
| <O> | AUTHOR:: | Kapidakis, Sarantos | (last, first) |
| <O> | AUTHOR:: | Sidiropoulos, Antonis | |
| <O> | DATE:: | January 1996 | (month year) or (month day, year) |
| <O> | PAGES:: | 30 | (number) |
| <O> | LANGUAGE:: | Greek | (free-text) |
| <O> | ABSTRACT:: | xxxx……xxxx | (free-text) |
| | | xxxx……xxxx | |
| | | xxxx……xxxx | |
| <M> | END:: | ercim.ics//TR96-128 | (docid) |

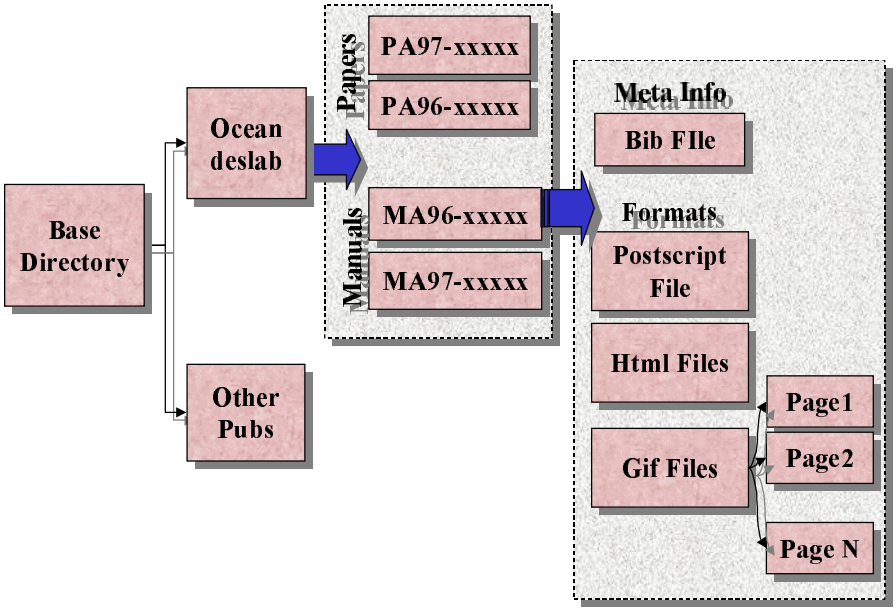
Πίνακας 2. RFC-1357 του dienst

## 3.3. Repository Server

### *3.3.1. Η Βάση Δεδομένων του Συστήματος*

Ένα από τα ενδιαφέροντα τμήματα του dienst είναι η βάση δεδομένων του, η απλότητά της και η ευελιξία που έχει για αποθήκευση διάφορων τύπων αντικειμένων. Στο Σχήμα 3 φαίνεται η δομή της βάσης δεδομένων, η οποία βασίζεται σε μια ιεραρχική οργάνωση του συστήματος

αρχείων. Για κάθε κείμενο (αντικείμενο) υπάρχει οπωσδήποτε ένα αρχείο με βιβλιογραφικές πληροφορίες[5] (meta info). Κάθε κείμενο έχει ένα αναγνωριστικό κειμένου (document id), το οποίο είναι μοναδικό στο σύνολο των κειμένων του αντίστοιχου εκδότη. Στο αναγνωριστικό κειμένου, προστίθεται και το αναγνωριστικό του εκδότη ({authority}:TR{year}-{number}), οπότε και δημιουργείται ένα παγκοσμίως μοναδικό αναγνωριστικό (unique id).




Σχήμα 3. Η δομή της Βάσης Δεδομένων του dienst

Οι προκαθορισμένοι τύποι αντικειμένων, χωρισμένοι σε 3 μεγάλες κατηγορίες, είναι:

- Summary
  - Thumbnail
- Inline images
  - GIF (zoom capability)
  - TIFF
  - Every type supported by MIME-format
- Document
  - Postscript (capability for page selection)

[5] Στο dienst αναφέρεται σαν bib file.

- OCR
- plain text

Στην παράγραφο 4.4.1 θα δούμε πώς ακριβώς μπορεί ο υπεύθυνος του συστήματος να προσθέσει νέους τύπους δεδομένων.

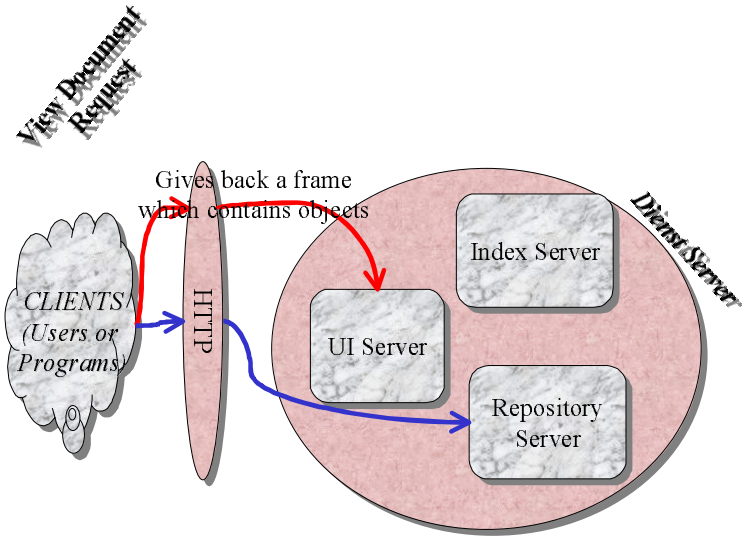
### *3.3.2. Repository Service*

Το Repository Service έχει την ευθύνη να μοιράζει τα κείμενα που του ζητούνται. Συνήθως αυτό γίνεται σε συνεργασία με το UI service. Δηλαδή ο χρήστης ζητά από το UI να του εμφανιστεί η σελίδα «χ» του «ψ» κειμένου. Ο UI server, επιστρέφει ένα HTML αρχείο με τα απαραίτητα links, και ένα link σε inline αντικείμενο, το οποίο είναι η σελίδα που ζητείται, με το τρόπο (Σχήμα 4):

```
<img src=http://repository_server/Dienst/Repository/2.0/Page/«χ»/«ψ»>
```

Και ο browser κάνει αυτόματα την μεταφορά της σελίδας από τον Repository Server.

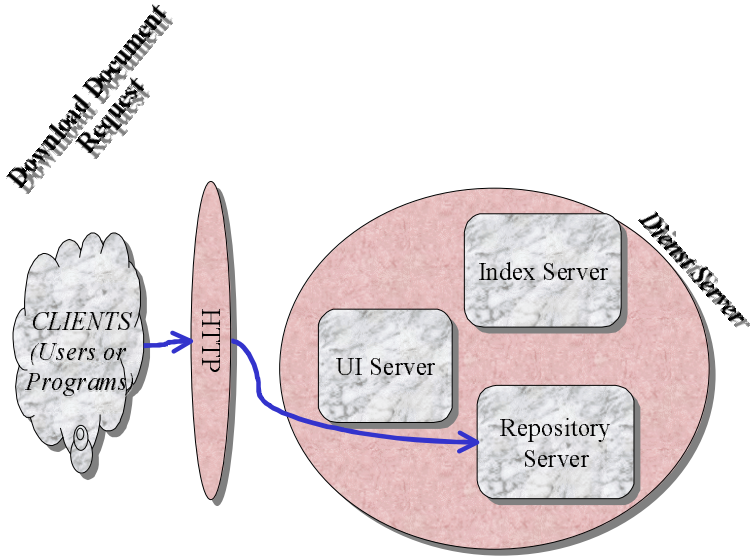
Έτσι τελικώς έχουμε ένα πραγματικά κατανεμημένο σύστημα, ακόμη και ανάμεσα στα διαφορετικά services του ίδιου dienst server.




Σχήμα 4. Αίτηση εμφάνισης κειμένου από χρήστη.

Βέβαια υπάρχει και η περίπτωση, να επικοινωνήσει ο χρήστης απευθείας με τον Repository Server (Σχήμα 5). Αυτό γίνεται μόνο στην περίπτωση που ο χρήστης κάνει «download» έναν

τύπο στον οποίο είναι αποθηκευμένο το κείμενο, για παράδειγμα το postscript αρχείο (βεβαίως αυτό συμβαίνει μετά την καθοδήγηση του UI server προς τον χρήστη με τα αντίστοιχα links).




Σχήμα 5. Αίτηση για download κειμένου.

## 3.4. Index Server του συστήματος

Ο Index Server του συστήματος χρησιμοποιεί «μόνο» τα meta data των κειμένων ώστε να μπορεί να κάνει αναζήτηση σε αυτά. Μπορούμε να μπούμε ότι χωρίζεται σε 3 τμήματα:

- το τμήμα που προετοιμάζει τα indices (ο index builder ή indexer)
- το τμήμα που εκτελεί τις αναζητήσεις (ο server)
- και ένα ακόμη τμήμα για άλλες λειτουργίες, όπως μετατροπή άλλων μορφών meta data στην μορφή RFC-1357.

### *3.4.1. Δημιουργός Αρχείων Ευρετηριασμού (Index builder)*

Αυτό το τμήμα αναλαμβάνει να οργανώσει τα αρχεία ευρετηριασμού του συστήματος. Τα αρχεία αυτά χτίζονται με βάση τα meta data μόνο, και είναι απλές αντεστραμμένες λίστες (inverted files).

Δημιουργούνται 3 κύριες λίστες ευρετηριασμού για:

- Συγγραφείς (author words)
- λέξεις στις περιλήψεις (abstract words)
- λέξεις στους τίτλους (title words)

Για παράδειγμα το αρχείο με τις λέξεις στις περιλήψεις είναι της μορφής:

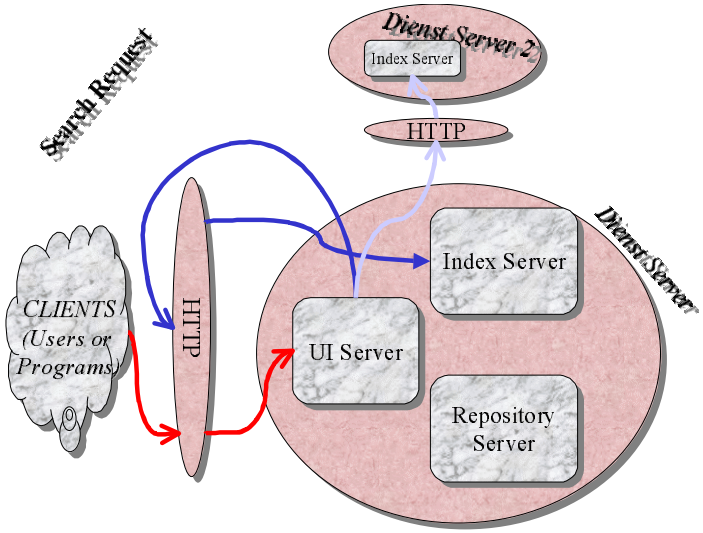
| | |
|---|---|
| network | ercim.forth.ics:TR97-1988 ercim.forth.ics:TR97-1970 |
| networks | ercim.forth.ics:TR97-1988 ercim.forth.ics:TR96-1890 |
| negative | ercim.forth.ics:TR97-1890 ercim.forth.ics:TR97-1970 |

Στην πρώτη στήλη έχουμε μια λέξη, και στις υπόλοιπες, τα αντικείμενα τα οποία περιέχουν την συγκεκριμένη λέξη. Για παράδειγμα η λέξη "negative" περιέχεται στα αντικείμενα "ercim.forth.ics:TR97-1890" και "ercim.forth.ics:TR97-1970".

### *3.4.2. Εξυπηρετητής Ευρετηριασμού (Index Server)*

Ο Index server χρησιμοποιεί τα αρχεία ευρετηριασμού (Βλέπε 3.4.1) ώστε να εξυπηρετήσει τις αιτήσεις αναζήτησης που δέχεται. Όπως και στην περίπτωση του Repository Server, ο χρήστης δεν κάνει απευθείας αιτήσεις προς τον index server, αλλά μέσω του UI server (Βλέπε Σχήμα 6).

Όταν ο UI server δέχεται μια αίτηση για αναζήτηση, τότε ελέγχει σε ποιους εκδότες απευθύνεται η ερώτηση, και προωθεί την ερώτηση για κάθε εκδότη στον αντίστοιχο index server. Επειδή τα κείμενα ενός εκδότη μπορεί να γίνονται index από πολλούς servers, για κάθε εκδότη, επιλέγεται ο κοντινότερος index server (της περιοχής) που περιέχει τα αρχεία ευρετηριασμού αυτού του εκδότη. Όταν




Σχήμα 6. Αίτηση αναζήτησης στο dienst από χρήστη.

περιέχεται ο τοπικός εκδότης, τότε η αίτηση σε αυτόν προωθείται ακριβώς με τον ίδιο τρόπο όπως και στους απομακρυσμένους. Αυτό σημαίνει ότι η αίτηση αναζήτησης στον τοπικό εκδότη ακολουθεί το μονοπάτι: user browser → http server → cgi-script → dienst server (UI module) → http server → cgi-script → dienst server (index module) (Παράρτημα IV. ).

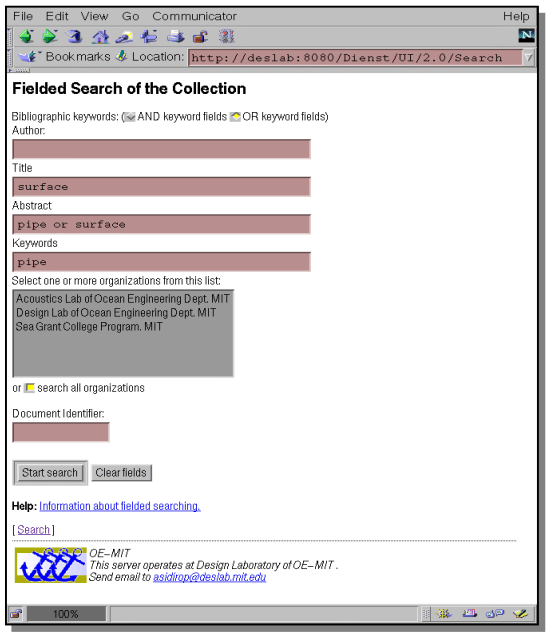
## 3.5. User Interface Server του συστήματος

Όπως έχει αναφερθεί και πριν, η πρόσβαση στο σύστημα γίνεται χρησιμοποιώντας έναν web browser. Ο User Interface server του συστήματος, δημιουργεί τα κατάλληλα html αρχεία, τα οποία στέλνει στον browser, τα οποία αρχεία περιέχουν τις κατάλληλες εντολές για την μορφοποίηση της διεπιφάνειας χρήσης.

Οι κύριες λειτουργίες που μπορεί να κάνει ο χρήστης, είναι αναζήτηση και περιήγηση στα δεδομένα και εμφάνιση ή αντιγραφή (download) ενός αντικειμένου της συλλογής.

Παρακάτω, παραθέτουμε δύο φωτογραφικά στιγμιότυπα της φόρμας αναζήτησης (Σχήμα 7) και της διαδικασίας περιήγησης (Σχήμα 8).

Στο παράρτημα V. εξετάζεται με μεγαλύτερη λεπτομέρεια η διεπιφάνεια χρήσης του dienst.




Σχήμα 7. Φόρμα αναζήτησης του dienst


Σχήμα 8. Περιήγηση με βάση συγγραφέα ή χρονολογία

# Μέρος 2ο:
# Επεκτάσεις στο σύστημα dienst

# *Κεφάλαιο 4*

## 4. Επεκτάσεις στο σύστημα Dienst

Σε αυτό το κεφάλαιο θα δούμε επεκτάσεις που έγιναν στα πλαίσια αυτής της εργασίας, στο σύστημα dienst, και εργαλεία που αναπτύχθηκαν για την μελέτη του. Οι επεκτάσεις αυτές είναι για την υπάρχουσα συλλογή τεχνικών αναφορών.

1. Η πρώτη επέκταση που έγινε έχει να κάνει με το debugging του συστήματος και την μελέτη του πώς ακριβώς δουλεύει αυτό ώστε να μελετήσουμε την οργάνωσή του και την ροή των πληροφοριών.

2. Η δεύτερη επέκταση είναι η πρόσθεση πληροφορίας και υπολογισμού κόστους ενός αντικειμένου. Είναι προφανές, ότι από κάποια χρονική στιγμή και μετά, οι περισσότερες, αν όχι όλες, παροχές πληροφοριών μέσω του διαδικτύου θα έχουν κάποιο κόστος για αυτόν που τις ζητάει. Αυτό διότι δεν είναι δυνατόν οι εκδότες, ή ο οποιοσδήποτε κάνει διαθέσιμη πληροφορία, συλλογές κειμένων, εικόνων κτλ., να το κάνει χωρίς να έχει κάποιο οικονομικό αντίκρισμα, ώστε τουλάχιστον να καλύπτει τα έξοδα που έχει κάνει για να δώσει αυτήν την παροχή υπηρεσιών.
   Άρα, τα συστήματα ψηφιακών βιβλιοθηκών, πρέπει να είναι προετοιμασμένα ώστε να μπορούν να υποστηρίξουν κάτι τέτοιο, διότι σε αντίθετη περίπτωση, αν δηλαδή οι εκδότες που κάνουν διαθέσιμη πληροφορία, δεν έχουν κάποιο οικονομικό όφελος από την χρήση μιας ψηφιακής βιβλιοθήκης, η πληροφορία που θα γίνεται διαθέσιμη θα είναι είτε χαμηλής ποιότητας, είτε θα είναι παλιά-και άρα σχεδόν άχρηστη.
   Ένα παράδειγμα σε αυτήν την περίπτωση είναι μια συλλογή ημερήσιου τύπου. Αν οι εκδότες δεν μπορούν να έχουν οικονομικό όφελος από την δημοσίευση αυτής της συλλογής, τότε θα κάνουν διαθέσιμα άρθρα *μόνο* παλαιότερων ημερών. Αν όμως υπάρχει η δυνατότητα για χρέωση, μπορεί ο χρήστης να αγοράζει την εφημερίδα της ημέρας από το διαδίκτυο, ή να μπορεί να δει με χαμηλότερο κόστος τις δημοσιεύσεις της τελευταίας εβδομάδας, ή να μπορεί να δει δωρεάν παλαιότερες δημοσιεύσεις.
   Έτσι, κάναμε μια προσπάθεια ενσωμάτωσης πληροφορίας κόστους στο σύστημα dienst. Βέβαια, αυτήν την στιγμή το dienst χρησιμοποιείται σχεδόν αποκλειστικά από πανεπιστήμια και ερευνητικούς οργανισμούς, που διαθέτουν δωρεάν τις τεχνικές αναφορές μέσω διαδικτύου. Όμως δεν μπορούμε να αποκλείσουμε πιθανές χρεώσεις στο μέλλον (π.χ. χρέωση στους ιδιώτες), και πολύ περισσότερο δεν μπορούμε να αγνοούμε τους υπόλοιπους

(δηλαδή τους μη ερευνητικούς οργανισμούς) που θα ήθελαν να δημιουργήσουν μια συλλογή με μια ψηφιακή βιβλιοθήκη.

3. Η τρίτη επέκταση, αφορά την καλύτερη οργάνωση των εκδοτών σε δύο κατευθύνσεις. Η πρώτη όσον αφορά την ονοματολογία και η δεύτερη όσον αφορά την καλύτερη παρουσίαση των εκδοτών (οι οποίοι ξεπερνούν πλέον τους 100), προς τον χρήστη. Σε αυτό το σημείο παρουσιάζουμε και κάποιες ιδέες για μελλοντικές επεκτάσεις.

4. Η τέταρτη επέκταση, αφορά τους τύπους των δεδομένων που μπορεί να αναγνωρίσει το dienst. Η αναγνώριση τύπων, οι οποίοι περιέχουν περισσότερα από ένα αρχεία με εσωτερικούς συνδέσμους από το ένα στο άλλο ή να περιέχει το ένα το άλλο, είναι ιδιαίτερη περίπτωση για το dienst. Μια τέτοια περίπτωση είναι η html μορφή η οποία αποτελείται από πολλά αρχεία κειμένων και εικόνων.

## 4.1. Dienst Debugging

Πριν κάνουμε οποιαδήποτε επέκταση σε ένα σύστημα, πρέπει να το μελετήσουμε εξαντλητικά. Να δούμε πώς ακριβώς δουλεύει, να μελετήσουμε την λειτουργικότητά του και να κάνουμε μια αξιολόγηση, ώστε να δούμε προς ποια κατεύθυνση θα κάνουμε τις επεκτάσεις και τις βελτιώσεις.

### *4.1.1. Flow Monitor*

Έτσι, σε πρώτη φάση, αναπτύχθηκε ένα βοηθητικό τμήμα το οποίο μπορούμε να το ονομάσουμε "flow monitor". Σε αυτήν την φάση προστέθηκε σε κάθε σημείο του dienst η ιδιότητα να καταγράφει σε html αρχείο όλα τα βήματα που έχουν γίνει, όλες τις κλήσεις «σημαντικών» ρουτινών και όλες τις αναπαραγωγές διεργασιών. Ένα παράδειγμα αυτού του αρχείου, φαίνεται στο Σχήμα 9. Επίσης καταγράφεται ο χρόνος που δαπανήθηκε σε κάθε ρουτίνα. Στο παράδειγμα φαίνεται ότι η Search_Remote διαρκεί 22 δεύτερα και όλη η διαδικασία της αναζήτησης 24 δεύτερα. [XIV]

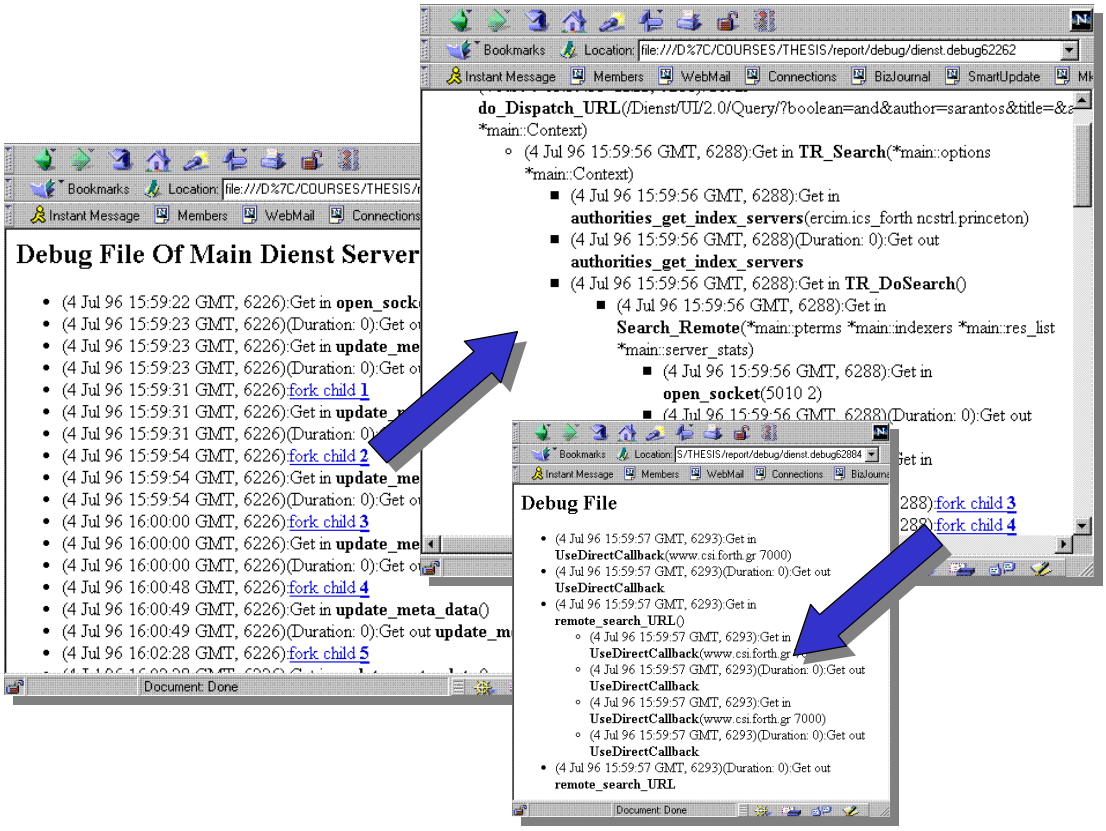
Οι στόχοι κατά την υλοποίηση του παραπάνω ήταν να αποκτήσουμε μια πρώτη οπτική εντύπωση για το πώς δουλεύει το dienst εσωτερικά και αργότερα να αναπτυχθεί περισσότερο το "flow monitor" ώστε:

- Να έχουμε μεγαλύτερη ακρίβεια στις μετρήσεις[6]

[6] Η πρώτη αυτή έκδοση, ήταν εξολοκλήρου ενσωματωμένη στον κώδικα του dienst, και άρα γραμμένη σε perl. Με την perl όμως δεν μπορείς να μετρήσεις χρόνο μικρότερο από την τάξη του δευτερολέπτου.

- Να είμαστε σε θέση να ορίσουμε κάποιες «στατιστικές» μεταβλητές για το dienst (όσον αφορά την απόδοση και την ταχύτητα)

- Και να επεκτείνουμε το πρωτόκολλο του dienst έτσι ώστε να μπορούμε να «βλέπουμε» πληροφορίες για την απόδοση και τον φόρτο του κάθε dienst server κατά την διάρκεια λειτουργίας του, και να έχουμε μια αποδοτική εξισορρόπηση φορτίου.




Σχήμα 9. Αρχεία του flow monitor.

## *4.1.2. Performance Monitor*

Δεδομένου ότι τα αποτελέσματα από την πρώτη «εξερεύνηση» του συστήματος ήταν ενδιαφέροντα, κρίθηκε απαραίτητο να αποκτήσουμε μεγαλύτερη ακρίβεια στις μετρήσεις. Έτσι υλοποιήθηκε ένα ξεχωριστό τμήμα[7], το οποίο έτρεχε ανεξάρτητα από το dienst και δημιουργούσε αρχεία της παραπάνω μορφής. Το τμήμα αυτό υλοποιήθηκε στην γλώσσα

[7] Το οποίο υλοποίησε ο Αντώνης Χατζησταματίου

προγραμματισμού C, οπότε και ήταν δυνατό να έχουμε μεγαλύτερη ακρίβεια στις μετρήσεις από την προηγούμενη έκδοση. [21]

Το dienst, κάθε φορά που ήθελε να γράψει κάτι σε αρχείο, αντί να το γράψει το ίδιο, έστελνε ένα μήνυμα στο performance monitor, το οποίο αφού υπολόγιζε τον χρόνο, έκανε την καταγραφή στο αρχείο. Σε μεταγενέστερες εκδόσεις προστέθηκε ο υπολογισμός μέσων όρων χρόνων και διάφορων άλλων πληροφοριών απόδοσης.

Για να φτάσουμε τέλος σε μια ολοκληρωμένη έκδοση ενός performance monitor[8] [20].

## 4.2. Πληροφορία Κόστους (Pricing Information)

Με τα συστήματα Ψηφιακών Βιβλιοθηκών, πλέον ο κάθε χρήστης είναι σε θέση να μπορεί να αποκτήσει τις πληροφορίες που επιθυμεί. Δεν είναι όμως δυνατόν κάποιος εκδότης ή ερευνητικό ινστιτούτο, να μοιράζει πάντα πληροφορίες στο κοινό χωρίς το αντίστοιχο οικονομικό αντίτιμο. Έτσι προχωρούμε προς την επόμενη γενιά των Ψηφιακών Βιβλιοθηκών, οι οποίες λαμβάνουν υπ' όψιν τους το κόστος της κάθε πληροφορίας καθώς και την ποιότητα της παροχής υπηρεσιών. Έτσι επεκτείναμε το dienst[9], ώστε να περιέχει και πληροφορίες κοστολόγησης του κάθε αντικειμένου που περιέχει και που είναι δυνατό να αντιγράψει ο χρήστης.

Το κόστος κάθε μετακίνησης πληροφορίας είναι συνάρτηση ενός πλήθους παραγόντων, όπως το κόστος του ίδιου του αντικειμένου προς μεταφορά, η ποιότητα αποθήκευσης της πληροφορίας[10] καθώς και ο φόρτος του συστήματος την στιγμή της συναλλαγής.

### *4.2.1. Επεκτάσεις στο σύστημα*

#### *4.2.1.1. Επεκτάσεις στα meta data*

Για να γίνει αυτό, το πρώτο πράγμα που χρειαζόμαστε στο σύστημα είναι η πληροφορία του κόστους του αντικειμένου. Έτσι επεκτάθηκαν τα meta data του dienst, προσθέτοντας ένα

[8] Η οποία υλοποιήθηκε από τον Σωτήρη Τερζή σαν τμήμα της μεταπτυχιακής του εργασίας [20] και τον Αντώνη Χατζησταματίου ο οποίος υλοποίησε ένα «real-time Viewer» των παραμέτρων απόδοσης.

[9] Η υλοποίηση έγινε από τους Αντώνη Σιδηρόπουλο (προσαρμογή του data base module και price computation), Αντώνη Χατζησταματίου (προσαρμογή του User Interface) και Σωτήρη Τερζή (distributed name server module).

[10] Αυτό είναι περισσότερο προφανές στις περιπτώσεις όπου το αντικείμενο μπορεί να υπάρχει σε διαφορετικές ποιότητες. Δηλαδή στις περιπτώσεις ήχου ή βίντεο, όπου το αντικείμενο μπορεί να διατεθεί με διάφορες ποιότητες, ή στις περιπτώσεις εικόνας, όπου πάλι μπορούμε να έχουμε διαφορετικές ποιότητες στο αντικείμενο (πχ. Μεγάλη ή μικρή ανάλυση, πλήθος χρωμάτων κτλ.)

ακόμη πεδίο με την πληροφορία του κόστους. Έτσι η μορφή του meta data αρχείου έχει αυτήν που φαίνεται στον παρακάτω πίνακα (Πίνακας 3).

Δεδομένου ότι η ανάπτυξη του παραπάνω έγινε κυρίως για σκοπούς επίδειξης, δεν υπήρχε μονάδα κόστους σε κάποιο από τα γνωστά νομίσματα (π.χ. USD), ώστε να μπορούν να γίνονται και οι αντίστοιχες μετατροπές, αλλά απλά ένας απόλυτος αριθμός. Επίσης το κόστος κάποιου αντικειμένου περιέχεται στα ίδια τα meta data.

| | | | |
|---|---|---|---|
| <M> | BIB-VERSION:: | CS-TR_v2.0 | |
| <M> | ID:: | ercim.ics//TR96-128 | (docid) |
| <M> | ENTRY:: | January 15, 1996 | (month day, year) |
| <O> | ORGANIZATION:: | ICS - FORTH | (free-text) |
| <M> | TITLE:: | A Spider for Dienst | (free-text) |
| <O> | AUTHOR:: | Kapidakis, Sarantos | (last, first) |
| <O> | AUTHOR:: | Sidiropoulos, Antonis | |
| <O> | DATE:: | January 1996 | (month year) or (month day, year) |
| <M> | PRICE:: | 100 | (number) |
| <O> | PAGES:: | 30 | (number) |
| <O> | LANGUAGE:: | Greek | (free-text) |
| <O> | ABSTRACT:: | xxxx……xxxx | (free-text) |
| | | xxxx……xxxx | |
| | | xxxx……xxxx | |
| <M> | END:: | ercim.ics//TR96-128 | (docid) |

Πίνακας 3. Meta data του dienst με την πληροφορία κόστους

*4.2.1.2. Ποιότητα Υπηρεσίας*

Ο παράγοντας αυτός υπολογισμού του κόστους, ουσιαστικά αγνοήθηκε, αφού σε όλα τα αντικείμενα και σε όλους τους τύπους αντικειμένων (και για λόγους απλότητας) δόθηκε ο ίδιος συντελεστής ποιότητας.

*4.2.1.3. Φόρτος Εξυπηρετητή*

Ο συντελεστής υπολογίζεται με βάση το ρυθμό αιτήσεων που έρχονται στον εκάστοτε εξυπηρετητή. Ο ρυθμός αιτήσεων υπολογίζεται από:

$$r = \text{ρυθμός αιτήσεων} = \frac{\text{πλήθος αιτήσεων στον χρόνο t}}{t}$$ [11]

Έτσι γίνεται υπολογισμός του επιπλέον κόστους πρόσβασης (ac) σε κάποιον server.

$ac = r * k$, όπου k ένας συντελεστής βαρύτητας, ο οποίος εξαρτάται από την υπολογιστική ισχύ του εξυπηρετητή, το κόστος συντήρησής του. Η μονάδα μέτρησής του είναι (χρηματική μονάδα / χρόνο, ώστε ο συντελεστής ac να αναπαριστά χρηματικές μονάδες.

Το *ac* στέλλεται σε ένα distributed name server (dns). Έτσι ανά πάσα στιγμή ο οποιοσδήποτε server μπορεί να μάθει για το *ac* οποιουδήποτε άλλου, και να ενημερώνει τον χρήστη.

### *4.2.2. Τελικό Κόστος*

Έτσι το τελικό κόστος ενός αντικειμένου είναι:

final cost = ac + c * q ( c το καθαρό κόστος του αντικειμένου, q ο συντελεστής ποιότητας τον οποίο θέσαμε 1 και ac το κόστος πρόσβασης στον server).

## 4.3. Ιεραρχία Εκδοτών

Αρχικά το authorityID (αναγνωριστικό εκδότη), είχε σαν στόχο να δώσει έναν μοναδικό κωδικό στον κάθε εκδότη. Αργότερα, όταν οι εκδότες πολλαπλασιάστηκαν, αποφασίστηκε να τηρείται μια ιεραρχική ονοματολογία στους κωδικούς των εκδοτών[12]. Έτσι ορίστηκε ότι ο τύπος για τα αναγνωριστικά εκδοτών θα είναι:

```
[organization that belongs to].[authority].[sub-authority]…
```

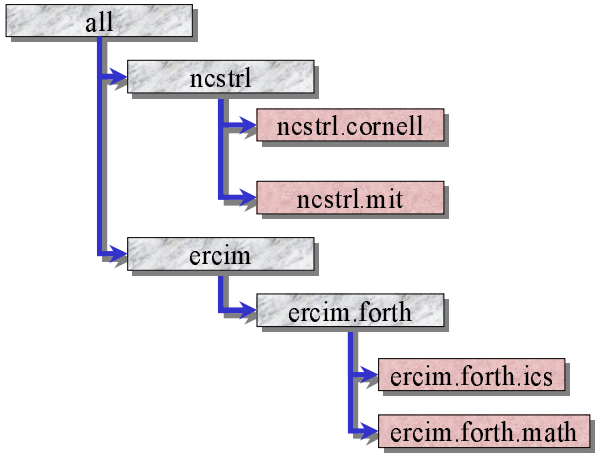
Το πρώτο τμήμα αυτή τη στιγμή μπορεί να είναι είτε ncstrl είτε ercim. Το δεύτερο τμήμα δηλώνει την κωδική ονομασία του Ιδρύματος ή του πανεπιστήμιου, και το τρίτο τμήμα το ινστιτούτο ή το τμήμα (department) του πανεπιστημίου. Για παράδειγμα το ΙΤΕ/ΙΠ έχει την κωδική ονομασία ercim.forth.ics. Με αυτόν τον τρόπο είναι απολύτως ξεκάθαρο ποιος υπάγεται που.

Σε αυτό το σημείο έχει γίνει και μια αλλαγή στην διεπιφάνεια χρήσης του dienst, ώστε η πληροφορία που έχουμε για την ιεραρχία των εκδοτών, να είναι εμφανής και στον απλό

[11] Έτσι αναγκαστικά για κάθε αίτηση που φτάνει στον εξυπηρετητή αποθηκεύεται και μια χρονοσφραγίδα.

[12] Όπως αυτή που χρησιμοποιείται στα domain names.

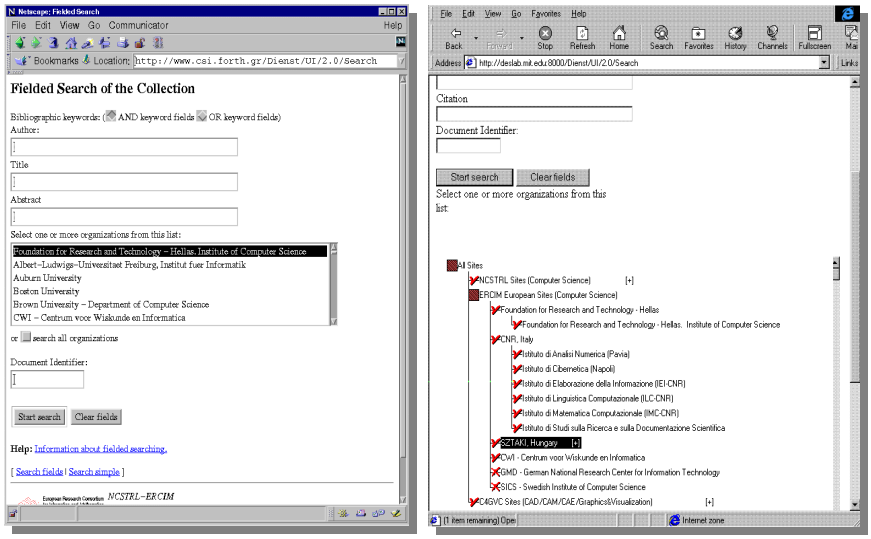
χρήστη. Έτσι η φόρμα επιλογής εκδοτών από απλή λίστα, μετατρέπεται σε ένα δέντρο που έχει την μορφή, όπως φαίνεται στο Σχήμα 10.




Σχήμα 10. Δεντροειδής μορφή οργάνωσης των εκδοτών.

Με την υπάρχουσα οργάνωση του dienst, υπάρχει κάποιο πρόβλημα που πρέπει να επιλύσουμε. Είναι προφανές ότι όταν τυπώνουμε το δέντρο των εκδοτών, δεν τυπώνουμε τα αναγνωριστικά τους, αλλά τα πλήρη ονόματά τους. Έτσι ξέρουμε (από την λίστα των εκδοτών) το πλήρες όνομα του ercim.forth.ics, αλλά δεν ξέρουμε το πλήρες όνομα για τον κόμβο ercim.forth. Για να επιλύσουμε αυτό το πρόβλημα, θα πρέπει η πληροφορία για την ονοματολογία των κόμβων (και όχι μόνο των φύλλων) να περιέχεται στην λίστα των εκδοτών που παρέχουν οι meta servers (με την αίτηση /Dienst/Meta/2.0/Publishers). Στο ΙΤΕ/ΙΠ ακολουθήσαμε μια δεύτερη εναλλακτική λύση: στον meta server που χρησιμοποιούμε (Βλέπε και παράγραφο 5.3) υλοποιήθηκε ακόμη ένας τύπος αίτησης (αντίστοιχος με τους Publishers, Repositories, Indices), ο οποίος επιστρέφει την ονοματολογία των κόμβων του δέντρου των εκδοτών. Έτσι και η αίτηση για τα ονόματα των «Publishers» παραμένει ως έχει, δηλαδή να επιστρέφει μόνο τα φύλλα του δέντρου των εκδοτών.

Το τελικό αποτέλεσμα αυτής της ιεραρχικοποίησης των εκδοτών φαίνεται στο Σχήμα 11.




Default dienst publishers' selection

Hierarchical dienst publishers' selection

Σχήμα 11. Ιεραρχική και μη, επιλογή των εκδοτών.

### *4.3.1. Προτάσεις για μελλοντικές επεκτάσεις*

Αυτήν την στιγμή, η συλλογή ncstrl-ercim, περιέχει σχεδόν αποκλειστικά τεχνικές αναφορές επιστήμης υπολογιστών. Μελλοντικά όμως θα μπορούσε να εμπλουτιστεί με την είσοδο και άλλων θεματικών περιοχών όπως μαθηματικών, βιολογίας κτλ., καθώς και οργανισμών, Θα πρέπει λοιπόν να υπάρχει κάποιος εύκολος τρόπος ταξινόμησης των εκδοτών με βάση το περιεχόμενο της συλλογής τους. Έτσι μια ιδέα είναι η πρόσθεση στο τέλος του authorityID και ενός αναγνωριστικού του τύπου της συλλογής. Για παράδειγμα συλλογές με τεχνικές αναφορές επιστήμης υπολογιστών θα μπορούσαν να έχουν την κατάληξη cstr (Computer Science Technical Reports), συλλογές με κείμενα βιολογίας την κατάληξη btr (Biology Technical Reports) κ.ο.κ..

Έτσι, ο χρήστης θα μπορεί να επιλέγει να δει διαφορετικές όψεις της ιεραρχίας των εκδοτών, είτε με βάση σε ποιο ινστιτούτο ανήκουν, είτε με βάση τον τύπο των δεδομένων. Για παράδειγμα,, αρχικά ο διαχωρισμός σε συλλογές με κείμενα, εικόνες, ήχο, έπειτα διαχωρισμός με βάση τον επιστημονικό τομέα που αυτά αφορούν , είτε εναλλακτικά αντίστροφη ιεράρχηση (πρώτα ο επιστημονικός τομέας και έπειτα με βάση τον τύπο των δεδομένων), είτε με οποιαδήποτε άλλη σειρά.

Η κεντρική ιδέα είναι, να δίνουμε την δυνατότητα στον χρήστη να κάνει την εναλλαγή στον τρόπο εμφάνισης της ιεραρχίας των εκδοτών, με δυναμικό τρόπο.

## 4.4. HTML μορφή αντικειμένων της βάσης του dienst

### *4.4.1. Μέθοδος Πρόσθεσης νέων τύπων αντικειμένων*

Ο τρόπος με τον οποίο δηλώνονται νέοι τύποι αντικειμένων στο dienst, είναι δηλώνοντας την κατάληξη του αρχείου και τον τύπο MIME:

```
def_format("Type", "MIME-Type", $code, "Full Name",
           "File Pattern", file-pattern-args)
```

Πίνακας 4. Τρόπος δήλωσης νέων τύπων αντικειμένων

Αυτού του είδους οι δηλώσεις, γίνονται στο αρχείο custom.pl, το οποίο περιέχει τις δηλώσεις των μορφών των δεδομένων. Για παράδειγμα για να δηλωθεί ο τύπος postscript χρειάζεται η εξής εντολή μέσα στο προηγούμενο αρχείο (Περισσότερα στις οδηγίες εγκατάστασης του συστήματος [XX][XXI]) :

```
def_format("postscript", "application/postscript", 0, "Postscript", "%s.ps",
"name")
```

Βέβαια, για να δηλωθούν περισσότερο πολύπλοκοι τύποι, για παράδειγμα αυτοί που περιλαμβάνουν περισσότερα από ένα αρχεία, δεν είναι τόσο τετριμμένη η δήλωσή τους.

### *4.4.2. Πρόσθεση HTML τύπου δεδομένων*

Όπως αναφέρθηκε πριν, η πρόσθεση ενός νέου τύπου δεδομένων, ο οποίος αποτελείται από ένα και μόνο αρχείο (όπως postscript, pdf, κτλ.) είναι τετριμμένη. Όμως η αναγνώριση και παρουσίαση τύπων αρχείων, τα οποία αποτελούνται από ένα σύνολο ανομοιογενών αρχείων, είναι τελείως διαφορετική περίπτωση για το dienst.

Αυτό διότι, όλα τα αρχεία, στην πραγματικότητα είναι τμήματα μόνο ενός τύπου δεδομένων, και μπορεί να περιέχουν συνδέσμους από το ένα τμήμα στο άλλο ή να περιέχει το ένα τμήμα κάποιο άλλο. Στην περίπτωση του html τύπου, ένα κείμενο αποτελείται από πολλά αρχεία κειμένου (π.χ. το κάθε κεφάλαιο μπορεί να είναι ένα διαφορετικό html αρχείο), τα αρχεία περιέχουν συνδέσμους μεταξύ τους (π.χ. από τον πίνακα περιεχομένων σύνδεσμοι προς όλα τα κεφάλαια ή από ένα κεφάλαιο στα υποκεφάλαια), και το κάθε αρχείο κειμένου μπορεί να εμπεριέχει και άλλα αρχεία (πχ. ένα κεφάλαιο περιέχει εικόνες οι οποίες είναι διαφορετικά

αρχεία). Έτσι πρέπει (με κατάλληλη επέκταση του πρωτοκόλλου) τα αρχεία αυτά να παραμένουν ανεξάρτητα αλλά να είναι εμφανές ότι περιέχονται σε ένα αντικείμενο.

Το πρώτο πρόβλημα που έχουμε να αντιμετωπίσουμε είναι πώς θα αποθηκεύσουμε αυτά τα αρχεία στην βάση δεδομένων διατηρώντας και την υπάρχουσα δομή της βάσης. Σε αυτήν την περίπτωση επιλέξαμε να περιλάβουμε όλα τα αρχεία της html μορφής σε ένα tar αρχείο[13]. Η αρχική σελίδα του κειμένου πρέπει να είναι αποθηκευμένη με το όνομα index.html (όπως έχει καθιερωθεί στην περίπτωση των html κειμένων). Έτσι η html μορφή, αφού έχει μετατραπεί σε ένα tar αρχείο, αποθηκεύεται στην βάση όπως και οι υπόλοιπες απλές μορφές.

Το δεύτερο πρόβλημα, είναι πώς δίνουμε πρόσβαση στον χρήστη σε αυτήν την μορφή. Σε αυτό το σημείο θα πρέπει να υπενθυμίσουμε το dienst πρωτόκολλο για πρόσβαση σε μια μορφή ενός κειμένου (βλέπε [14]). Για παράδειγμα η αίτηση για να δούμε την postscript μορφή της τεχνικής αναφοράς ercim.forth.ics/TR98-0139 είναι το παρακάτω URL:

*http://dienst.csi.forth.gr/Dienst/Repository/2.0/Body/postscript/ercim.forth.ics/TR98-0139*

Αντίστοιχα, στην html μορφή η αίτηση:

*http://dienst.csi.forth.gr/Dienst/Repository/2.0/Body/html/ercim.forth.ics/TR98-0139*

θα πρέπει να επιστρέφει το αρχικό html αρχείο (το index.html).

Για να πάρουμε κάποιο άλλο αρχείο που περιέχεται στην html μορφή, πρέπει να όνομα αυτού του αρχείου να περιέχεται ως όρισμα στο παραπάνω URL. Έτσι αν υποθέσουμε ότι στο παραπάνω TR98-0139, το κεφάλαιο 2 είναι αποθηκευμένο με το όνομα chapter2.html, το URL γίνεται:

*http://dienst.csi.forth.gr/Dienst/Repository/2.0/Body/html/ercim.forth.ics/TR98-0139?file=chaper2.html*

Αντίστοιχα, στην περίπτωση εικόνων, περιέχεται το όνομα του αρχείου της εικόνας στο όρισμα "file" στο URL της αίτησης:

[13] Εναλλακτικά θα μπορούσε να είχε χρησιμοποιηθεί και η μορφή zip. Σε αυτήν την περίπτωση όμως, χάνουμε σε χρόνο απόκρισης, διότι το αρχείο πρέπει να αποσυμπιέζεται κάθε φορά που ζητούμε κάποιο από τα περιεχόμενά του. Από την άλλη δεν κερδίζουμε σημαντικά σε χώρο στον δίσκο, διότι ο κύριος όγκος ενός html κειμένου είναι οι εικόνες που αυτό περιέχει, και όχι το ίδιο το κείμενο. Οι εικόνες όμως είναι σε gif ή jpeg μορφή, η οποία είναι ήδη συμπιεσμένη και δεν συμπιέζεται περισσότερο.

http://dienst.csi.forth.gr/Dienst/Repository/2.0/Body/html/ercim.forth.ics/TR98-0139?file=image35.gif

Για να γίνει αυτό, έχει οριστεί στο dienst ότι τις αιτήσεις για παρουσίαση html μορφής, θα τις διαχειρίζεται μια ρουτίνα, η οποία διαβάζει τα ορίσματα, και συγκεκριμένα το όρισμα "file" και εξάγει από το tar αρχείο το αρχείο με το όνομα που ζητήθηκε. Αν έχει παραληφθεί το όρισμα "file" τότε εννοείται το αρχικό αρχείο, δηλαδή το "index.html".

Επίσης, πριν από την δημιουργία του tar αρχείου, πρέπει τα html αρχεία να υποστούν κάποια επεξεργασία. Αυτή η επεξεργασία, είναι η αντικατάσταση των συνδέσμων, με συνδέσμους της μορφής των παραπάνω URLs. Έτσι όταν ένα html αρχείο (π.χ. το index.html) περιέχει ένα link στο "chapter2.html" τότε το link <a href="chapter2.html"> πρέπει να μετατραπεί σε <a href="TR98-0139?file=chapter2.html">. Έχει υλοποιηθεί ένα script, το οποίο κάνει ακριβώς αυτήν την δουλειά[14].

### *4.4.3. Περιορισμοί*

Οι σύνδεσμοι που περιέχονται σε ένα html αρχείο, μπορεί να είναι:

- είτε πλήρη URLs (<a href="http://www.csi.forth.gr/~asidirop">),
- είτε full paths στον παρών server (<a href="/proj/pleiades">),
- είτε relative paths από το παρών directory (<a href="./main.html">).

Στην περίπτωσή μας, όλοι οι σύνδεσμοι μεταξύ των διαφόρων τμημάτων ενός κειμένου πρέπει να είναι relative paths. Σε αντίθετη περίπτωση, θεωρείται ότι ο σύνδεσμος είναι προς κάποιο αρχείο εκτός του συγκεκριμένου κειμένου.

### *4.4.4. Εναλλακτικές λύσεις*

Μια άλλη εναλλακτική λύση, για την υλοποίηση των παραπάνω, όσον αφορά την πρόσβαση σε ένα κείμενο θα ήταν να χρησιμοποιήσουμε την δυνατότητα του HTTP πρωτοκόλλου, για αποστολή πολλών αρχείων με μια αίτηση. Έτσι θα αρκούσε μόνο μια αίτηση ώστε να πάρει ο χρήστης όλα τα τμήματα του html κειμένου. Κάτι τέτοιο όμως θα απαιτούσε την μεταφορά όλων των τμημάτων/ αρχείων του κειμένου, δηλαδή θα αναγκάζαμε τον χρήστη να μεταφέρει

[14] Το script παίρνει σαν όρισμα το όνομα (ID) του κειμένου (π.χ. TR98-0139) και το directory στο οποίο περιέχονται τα html αρχεία. Το script, αφού επεξεργαστεί τα links, βάζει τα αρχεία σε ένα tar αρχείο, έτοιμο προς χρήση από την βάση του dienst.

ολόκληρο το κείμενο, προκειμένου να διαβάσει ένα κεφάλαιο από αυτό. Η συγκεκριμένη λύση θα ήταν κατάλληλη μόνο για την περίπτωση που ο χρήστης θέλει πραγματικά να μεταφέρει όλο το κείμενο για να το αποθηκεύσει τοπικά και να το εκτυπώσει ώστε να το διαβάσει ευκολότερα. Για αυτήν την περίπτωση όμως, ενδείκνυται η postscript ή η pdf μορφή ενός κειμένου, η οποία είναι ένα αρχείο ειδικό για εκτύπωση, και όχι η html. Όλα τα κείμενα στην βάση του dienst, τα οποία προήλθαν από ηλεκτρονική μορφή κειμένου (και όχι από scan εκτύπωσης), υπάρχουν σε postscript ή σε pdf μορφή.

Άρα τελικώς, η υποστήριξη μόνο αυτής της λύσης, είναι ελλιπείς και περιοριστική. Ενώ η υποστήριξη της, σε συνδυασμό με αυτήν που έχει υλοποιηθεί είναι πλεονασμός (αλλά ίσως κάποια στιγμή χρήσιμη).

# *Κεφάλαιο 5*

## 5. Επεκτάσεις στο dienst ειδικά για το ΙΤΕ/ΙΠ

Σε αυτό το κεφάλαιο θα δούμε κάποιες επεκτάσεις που έγιναν στο dienst. Οι επεκτάσεις αυτές έγιναν προκειμένου να πετύχουμε την καλή λειτουργία του στον server του ΙΤΕ, και να διευκολυνθεί η εργασία της ομάδας των ΠΛΕΙΑΔΩΝ για την ανάπτυξη λογισμικού για το dienst. Έτσι, μετατράπηκε το dienst ώστε να είναι δυνατός ο ευρετηριασμός των ελληνικών τεχνικών αναφορών που υπάρχουν στο ΙΤΕ. Επίσης για να έχουμε κάποιον έλεγχο στους πολλούς dienst servers που έχουν εγκατασταθεί στα συστήματα των ΠΛΕΙΑΔΩΝ και στο ΙΤΕ-ΙΠ γενικότερα, δημιουργήσαμε έναν διαχειριστή των συστημάτων dienst, ο οποίος όμως μπορεί να χρησιμοποιηθεί και για οποιοδήποτε σύστημα από servers. Τέλος φτιάξαμε meta servers για το dienst, ώστε να μπορούμε να έχουμε διάφορες ανεξάρτητες ή όχι, ομάδες από servers.

### 5.1. Αναζήτηση στα Ελληνικά μέσω του dienst

Δεδομένου ότι η ευρέως αποδεκτή γλώσσα στον χώρο του internet είναι τα αγγλικά και το ότι το dienst δημιουργήθηκε στις Η.Π.Α., το τμήμα δημιουργίας των αρχείων ευρετηριασμού (indexer) του dienst αγνοεί τελείως κείμενα με ελληνικές λέξεις (δηλαδή τους ελληνικούς χαρακτήρες). Συνεπώς έγινε μια διόρθωσή του, ώστε να αναγνωρίζει ως λέξεις και αυτές που αποτελούνται από ελληνικούς χαρακτήρες.

Έτσι, μετά από αυτήν την διόρθωση, ο χρήστης ήταν δυνατόν να δώσει μια ερώτηση στα ελληνικά και να γίνει η αναζήτηση. Ακόμη όμως και σε αυτήν την περίπτωση υπάρχει κάποιο πρόβλημα. Τι γίνεται αν ο χρήστης θέλει να του επιστραφούν κείμενα που είναι είτε στα ελληνικά είτε στα αγγλικά; Αυτό είναι περισσότερο εμφανές όταν η ερώτηση βασίζεται σε όνομα συγγραφέα. Δηλαδή μια ερώτηση του χρήστη για "Nikolaou" θα πρέπει να επιστρέψει και τα κείμενα που έχουν συγγραφέα «Nikolaou» και αυτά που έχουν «Νικολάου». Μια λύση είναι να χρησιμοποιήσουμε έναν μεταφραστή-λεξικό για όλες τις γλώσσες. Στον server του ΙΤΕ/ΙΠ όμως, σαν πρώτη προσέγγιση κάναμε το εξής:

Όταν εισάγεται μια ελληνική λέξη στο αρχείο ευρετηριασμού, μετατρέπεται από ελληνικούς χαρακτήρες σε λατινικούς (π.χ. «Νικολάου» → «Nikolaou», «υπολογιστής» → «ypologisths» ), έχοντας ορίσει μια αντιστοιχία ένα προς ένα των ελληνικών με αγγλικούς χαρακτήρες. Έτσι αν ο χρήστης δώσει την ερώτηση «Nikolaou» θα του επιστραφούν «όλα» τα κείμενα του

«Nikolaou» και «Νικολάου»[15]. Αν ο χρήστης δώσει την ερώτηση στα ελληνικά, τότε μετατρέπεται η ερώτηση (με την ίδια αντιστοιχία που έχουμε ορίσει) σε λατινικούς χαρακτήρες, οπότε και η ερώτηση «Νικολάου» θα δώσει ακριβώς τα ίδια αποτελέσματα.

Αργότερα αναπτύχθηκε μια πλήρως πολυγλωσσική έκδοση του dienst από τον Ιάκωβο Μαυροειδή.

## 5.2. Αυτόματη διαχείριση dienst server

Όταν για ερευνητικούς σκοπούς χρησιμοποιούμε πολλούς servers, όπως γίνεται στην περίπτωση του dienst στο ΙΤΕ/ΙΠ, χρειαζόμαστε έναν κεντρικό μηχανισμό ώστε να παίρνουμε πληροφορίες για τους server αλλά και να μπορούμε να τους διαχειριστούμε. Έτσι αναπτύχθηκε ένα εργαλείο, το «dienstadmin», με το οποίο μπορούμε να διαχειριστούμε τους servers που έχουμε.

Προφανώς για να είναι χρησιμοποιήσιμο το εργαλείο αυτό, πρέπει αρχικά να εισάγουμε τις πληροφορίες για κάθε dienst server που έχουμε. Όλες αυτές οι πληροφορίες εισάγονται σε ένα αρχείο, του οποίου το format δεν είναι απαραίτητο να δούμε σε αυτήν την εργασία, αλλά μπορούμε να αναφέρουμε τους τύπους των πληροφοριών που αποθηκεύονται στο αρχείο αυτό για κάθε dienst server (ή για κάθε ομάδα από dienst servers η οποία έχει τα ίδια settings, απλά οι server-processes τρέχουν σε διαφορετικά μηχανήματα). Στον παρακάτω πίνακα (Πίνακας 5) φαίνεται μια τέτοια περίπτωση.

Έχοντας όλες τις πληροφορίες που φαίνονται στον παρακάτω πίνακα, ξέρουμε τα πάντα για τους servers, σε ποια μηχανήματα τρέχουν, ποια ports χρησιμοποιούν, πώς μπορούμε να τους ξεκινήσουμε ή να τους σταματήσουμε κτλ..

Στο Παράρτημα Ι δίνεται αναλυτικότερη περιγραφή αυτού του προγράμματος καθώς και οδηγίες χρήσης.

[15] Αυτό έγινε και για έναν ακόμη λόγο. Συνήθως στα μηχανήματα με λειτουργικό UNIX (σε αντίθεση με τα MS windows) είναι δύσκολο ο χρήστης να πληκτρολογήσει ελληνικούς χαρακτήρες. Έτσι ένα πολύ μεγάλο ποσοστό χρηστών δεν θα μπορούσε να κάνει μια αναζήτηση στα ελληνικά. Με αυτόν τον τρόπο μπορεί, γράφοντας απλά την ερώτηση στα ελληνικά αλλά με λατινικούς χαρακτήρες.

| | | |
|---|---|---|
| **Server Name** | DIENST4.1.2.music.2 | |
| **Server Info** | Dienst Server version 4.1.2 for alpha with music library v2 and Greek search support This is a group of servers. | |
| **Home Page** | *http://calliope.csi.forth.gr:1500/Dienst/htdocs/index.html* | |
| **Dienst Server Args** | useraccount | jalexak |
| | host | calliope,aspasia |
| | path | /proj/samos/dienst_music |
| | version | DIENST4.1.2.music.2 |
| | system | sunos |
| | httpserver | yes |
| | tostartup | /dienst_ro/bin/RestartDienstAlpha /proj/samos/dienst_music/DIENST4.1.2.music.2/dienst/logs_HOST/dienst-pid /proj/samos/dienst_music/DIENST4.1.2.music.2/dienst jalexak |
| | pidpath | /proj/samos/dienst_music/DIENST4.1.2.music.2/dienst/logs_HOST |
| **Http Server Args** | useraccount | jalexak |
| | host | aspasia,calliope |
| | path | /proj/samos/dienst_music/web_server1500_HOST/logs |
| | pidfile | httpd.pid |
| | system | sunos |
| | tostartup | /proj/samos/dienst_music/web_server1500_HOST/httpd_start |
| **Server Ports** | server_port | 1501 |
| | callback_port | 1502 |
| | http_port | 1500 |
| | hosts | aspasia,calliope |

Πίνακας 5. Στοιχεία του dienstadmin για ένα group από dienst servers

## 5.3. dienst meta server

### *5.3.1. meta server - έκδοση 0*

Για την εγκατάσταση και σωστή λειτουργία μιας ομάδας από dienst servers, είναι απαραίτητη η ύπαρξη ενός meta server, ο οποίος θα "συντονίζει" τους dienst servers που ανήκουν στην ομάδα. Αρχικά, όταν οι απαιτήσεις από έναν meta server ήταν σχετικώς μικρές, υλοποιήθηκε ένας εικονικός meta server. Δηλαδή απλά έγινε setup ένας web server, ο οποίος απαντούσε στις αιτήσεις τύπου: /Dienst/Meta/2.0/{Repositories, Publishers, Indices}. Η απάντηση σε αυτές τις αιτήσεις ήταν προϋπολογισμένη, και αυτό που έκανε απλά ο web server, ήταν να επιστρέψει απλά το αντίστοιχο αρχείο που περιείχε την απάντηση στην αίτηση.

### *5.3.2. meta server - έκδοση 1*

Η έκδοση αυτή, υλοποιήθηκε για την συλλογή C4GVC[16]. Σε αυτήν την συλλογή, οι απαιτήσεις μας από τον meta server ήταν οι εξής:

- Να ενσωματώνονται οι εκδότες της συλλογής ncstrl-ercim, ώστε ένας dienst server που ανήκει στην συλλογή C4GVC να μπορεί να κάνει αιτήσεις σε κάποιον από την NCSTRL.

- Να διατηρούνται τα regional settings, δεδομένου ότι έχουμε μόνο έναν meta server και όχι διάφορους regional meta servers.

- Να εξυπηρετείται αίτηση της μορφής /Dienst/Meta/2.0/Domains η οποία χρησιμοποιείται από την έκδοση του dienst που παρουσιάζει την ιεραρχική δομή των εκδοτών.

Η υλοποίηση της 3ης απαίτησης είναι απλή, αφού απλά ορίζεται τοπικά η απάντηση σε αυτήν την αίτηση.

Για την υλοποίηση της 1ης και δεύτερης απαίτησης γίνανε τα εξής:

Πρώτον προστέθηκε στην αίτηση το όρισμα της περιοχής, δηλαδή ένας dienst server που ανήκει στην περιοχή EU-CRETE, θα κάνει την αίτηση:
*/Dienst/Meta/2.0/Indices?region=EU-CRETE.*
Έτσι ο meta server μπορεί να αναγνωρίσει από ποια περιοχή προέρχεται η αίτηση.

Δεύτερον, ο meta server, όταν "μαζεύει" τις πληροφορίες για τα sites της ncstrl κάνει το εξής:

- Αίτηση στον master meta server (ή σε οποιονδήποτε meta server) για να πάρει την λίστα "Publishers". Αυτή η λίστα είναι πάντα κοινή για όλες τις περιοχές.

- Αίτηση στον master meta server (ή σε οποιονδήποτε meta server) για να πάρει την λίστα "Repositories". Και αυτή η λίστα είναι πάντα κοινή για όλες τις περιοχές.

- Αίτηση στον master meta server για να μάθει τους regional meta servers για κάθε περιοχή. Έπειτα γίνεται αίτηση σε κάθε regional meta server για να πάρουμε την λίστα "Indices", η οποία για κάθε περιοχή είναι διαφορετική.

[16] Η υλοποίηση αυτή έγινε στο Design Laboratory – MIT, για την εξυπηρέτηση της συλλογής C4GVC που δημιουργήθηκε εκεί.

- Γίνεται ενσωμάτωση της τοπικής πληροφορίας με αυτήν που μαζεύτηκε από την συλλογή ncstrl, και ο meta server είναι έτοιμος να εξυπηρετήσει αιτήσεις.

Για την καλύτερη λειτουργία του server, αποθηκεύεται σε μια cache με όλες τις αιτήσεις που έχει κάνει ο server. Έτσι αν κάποια στιγμή, η επικοινωνία με κάποιον server, στον οποίο θέλουμε να κάνουμε αίτηση, είναι ανέφικτη, τότε διαβάζονται τα περιεχόμενα της cache. Αντιθέτως, αν η αίτηση πετύχει, τότε ενημερώνονται τα περιεχόμενα της cache.

### *5.3.3. meta server - έκδοση 2*

Η τελευταία έκδοση συνδυάζει τις λειτουργίες της προηγούμενης έκδοσης με μερικές διαφοροποιήσεις και προσθέσεις. Σε αυτήν λοιπόν την έκδοση, οι αρχικές απαιτήσεις είναι οι εξής:

- Δυνατότητα εξυπηρέτησης της αίτησης "Domains" (όπως και στην προηγούμενη έκδοση.

- Δυνατότητα εξυπηρέτησης της αίτησης "Collections". Αυτή η αίτηση χρησιμοποιείται από το παραμετροποιημένο dienst (υλοποίηση του Παναγιώτη Αλεξάκου για την πτυχιακή του εργασία), και δίνει πληροφορίες για τα πεδία που περιέχουν τα meta data αρχεία του κάθε εκδότη.

- Δυνατότητα ενσωμάτωσης λίστας εκδοτών μιας (ή περισσοτέρων) εξωτερικών συλλογών, όπως αυτή της ncstrl ή οποιαδήποτε άλλης.

Η λειτουργία αυτής της έκδοσης, είναι λίγο διαφορετική από την προηγούμενη όσον αφορά την αντιμετώπιση των περιοχών. Σε αυτήν την έκδοση δεν μας ενδιαφέρουν τα regional settings. Απλά όταν ο υπεύθυνος του συστήματος ορίζει meta server μιας εξωτερικής συλλογής, θα πρέπει να ορίσει τον αντίστοιχο meta server που εξυπηρετεί την συλλογή της περιοχής του. Έτσι για παράδειγμα, για την περίπτωση ενός meta server που βρίσκεται στην Ελλάδα και θέλουμε να ενσωματώσουμε και την συλλογή ncstrl, θα πρέπει ο υπεύθυνος του συστήματος να ορίσει ως meta server τον regional meta server της περιοχής, δηλαδή τον dienst.csi.forth.gr και όχι τον ncstrl.cs.cornell.edu.

Περισσότερα για την χρήση του meta server παρατίθενται στο παράρτημα VII. .

# *Κεφάλαιο 6*

## 6. Ψηφιακές Βιβλιοθήκες με Γεωγραφικά Δεδομένα

Ο χώρος των ψηφιακών βιβλιοθηκών δεν περιορίζεται μόνο σε απλά κείμενα ή ακόμη και σε απλές συλλογές εικόνων. Έχουν ήδη αναπτυχθεί (ή έχουν αρχίσει να αναπτύσσονται) ψηφιακές βιβλιοθήκες με γεωγραφικά δεδομένα. Σε αυτήν την περίπτωση τα πράγματα είναι αρκετά διαφορετικά σε σχέση με την περίπτωση των κειμένων, και θα μπορούσαμε να πούμε αρκετά πιο πολύπλοκα.

Σε αυτό το κεφάλαιο θα δούμε την προσαρμογή που έχει γίνει στο dienst για την περίπτωση γεωγραφικών δεδομένων. Επίσης στο δεύτερο τμήμα αυτού του κεφαλαίου προτείνουμε μια νέα αρχιτεκτονική για την διαχείριση γεωγραφικών δεδομένων χρησιμοποιώντας μια καθαρά αντικειμενοστραφή σχεδίαση έχοντας σαν βάση την CORBA.

Σε αυτό το σημείο θα ήταν σκόπιμο να αναφερθεί ότι οι μελέτες που αναφέρονται σε αυτό το κεφάλαιο έχουν γίνει κατά την παραμονή του γράφοντος στο εργαστήριο σχεδίασης του τμήματος Ωκεανογραφίας του Τεχνολογικού Ινστιτούτου της Μασαχουσέτης (Design Laboratory - Ocean Engineering Department – MIT), σε συνεργασία με τους καθ. Νίκο Πατρικαλάκη και καθ. Χρυσόστομο Χρυσοστομίδη.

### 6.1. Το Dienst με Γεωγραφικά Δεδομένα

Η εργασία αυτή έγινε στο τμήμα Ωκεανογραφίας του M.I.T. και περιλαμβάνει 2 τμήματα. Το πρώτο τμήμα είναι η μετατροπή της βάσης, του ευρετηρίου και του Interface του dienst, ώστε να μπορεί να διαχειριστεί γεωγραφικά δεδομένα και κυρίως (γεωγραφικές) εικόνες. Το δεύτερο τμήμα είναι η υλοποίηση αναζήτησης και σε άλλα συστήματα γεωγραφικών δεδομένων μέσα από το dienst.

#### *6.1.1. Επέκταση του Dienst για Γεωγραφικά Δεδομένα*

Για να γίνει αυτό απαιτείται ένα πλήθος αλλαγών στον μηχανισμό του dienst και κυρίως στα meta data και στο πώς γίνεται η αναζήτηση σε αυτά, άρα στο τρόπο ευρετηριασμού. Στο User Interface οι αλλαγές είναι πολύ μικρές, δεδομένου ότι τα data είναι κυρίως εικόνες (GIF).

Ας κάνουμε αρχικά μια ανάλυση των αναγκών που προκύπτουν και μετά θα δούμε πώς υλοποιήθηκαν οι απαραίτητες αλλαγές.

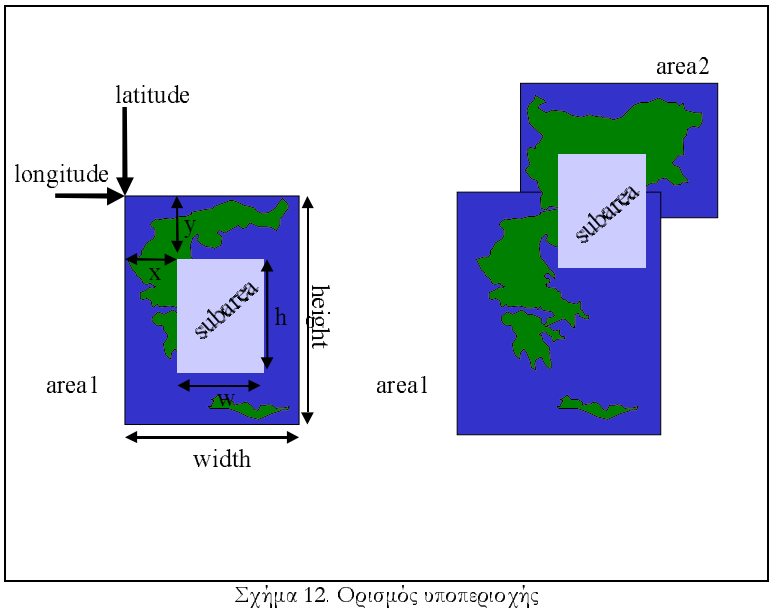
Είναι προφανές ότι κάθε δεδομένο σε ένα γεωγραφικό σύστημα έχει και την ιδιότητα της γεωγραφικής θέσης. Έτσι πρέπει να προστεθεί αυτή η πληροφορία στα meta data και η δυνατότητα αναζήτησης με βάση αυτή την ιδιότητα. Επίσης κάθε δεδομένο έχει μια χρονοσφραγίδα (ή μια χρονοσφραγίδα περιόδου). Άρα ο χρήστης πρέπει να έχει την δυνατότητα να κάνει αναζήτηση με βάση το χρόνο. Ακόμη, πρέπει να μπορεί να επιλέξει την σειρά με την οποία θα του εμφανίζονται τα δεδομένα σε μια λίστα, δηλαδή αν πρώτο κριτήριο θα είναι ο χρόνος ή η γεωγραφική θέση κτλ..

Επίσης, μια άλλη χρήσιμη λειτουργία θα ήταν η άντληση μερικής πληροφορίας. Τα περισσότερα στοιχεία της βάσης, όπως είπαμε και πριν, έχουν την ιδιότητα της γεωγραφικής θέσης, δηλαδή γεωγραφικό πλάτος και μήκος της αρχής και το μέγεθος της περιοχής, π.χ. (latitude, longitude, width, height). Ο χρήστης όμως, πολλές φορές, θέλει πληροφορίες για μια υποπεριοχή ή για μια περιοχή που συνδυάζει δεδομένα από 2 (ή περισσότερες) γειτονικές περιοχές με την ίδια χρονοσφραγίδα (Σχήμα 12). Θα θέλαμε λοιπόν από ένα σύστημα να είναι σε θέση να κάνει μια τέτοια λειτουργία[17] ώστε κάθε φορά ο χρήστης να μεταφέρει ακριβώς (ή την μικρότερη δυνατή) ποσότητα δεδομένων που χρειάζεται, και όχι υπερσύνολο αυτής.

*6.1.1.1. Αλλαγές στα meta data – meta data descriptor*

Για να έχουμε την δυνατότητα να αποθηκεύσουμε τις γεωγραφικές πληροφορίες, έγιναν κάποιες αλλαγές στα meta data. Προκειμένου το σύστημα να είναι ευέλικτο και να μπορούν να γίνουν και άλλες προσθέσεις πεδίων στα meta data, ορίστηκε μια μορφή αρχείου για την περιγραφή των meta data – metadata description file. Το αρχείο αυτό περιέχει την περιγραφή των meta data, δηλαδή: ποια πεδία υπάρχουν, ποια από αυτά τα πεδία χρησιμοποιούνται στην αναζήτηση, ποια χρησιμοποιούνται στην περιήγηση, τι τύπου είναι το καθένα, δηλαδή πώς κάνουμε μια σύγκριση μεταξύ πεδίων αυτού του τύπου, και αν χρειάζεται το πεδίο περαιτέρω επεξεργασία (σε ένα πεδίο με κείμενο πρέπει να αναγνωρίσουμε τις λέξεις, σε ένα πεδίο με ημερομηνία πρέπει να βρούμε την ημερομηνία αυτή κτλ.).

[17] Κάτι τέτοιο δεν υλοποιήθηκε τελικώς για το dienst στο MIT. Έτσι περιοριζόμαστε σε περιοχές όπως ακριβώς αυτές υπάρχουν στην βάση.




Σχήμα 12. Ορισμός υποπεριοχής

Έτσι ο υπεύθυνος του συστήματος μπορεί να ορίσει και νέους τύπους πεδίων και το πως γίνονται συγκρίσεις πάνω σε αυτούς, καθώς και διάφορες άλλες πράξεις που μπορούν να γίνουν[18].

*6.1.1.2. Αλλαγές στο User Interface*

Επίσης έχουν γίνει αλλαγές και στο Interface του συστήματος, όσον αφορά τον τρόπο με τον οποίο εισάγονται οι ερωτήσεις των χρηστών για τα επιπλέον δεδομένα. Περισσότερα, για αυτές τις επεκτάσεις αναφέρονται στην παράγραφο 7.3.

Όταν τα δεδομένα είναι αποθηκευμένα υπό την μορφή εικόνας, τότε ο χρήστης μπορεί να δει απευθείας (με μια αίτηση στον Repository Server) τις εικόνες (Σχήμα 13). Αν τα δεδομένα είναι σε άλλη μορφή[19] τότε απλά θα τα κάνει download. Άρα σε αυτήν την περίπτωση, για να δώσουμε στον χρήστη την δυνατότητα να δει γρήγορα τα δεδομένα υπό μορφή εικόνας, θα πρέπει να μετατρέπονται από το σύστημα με το αντίστοιχο φίλτρο (εάν βεβαίως είναι δυνατή μια τέτοια αναπαράσταση των δεδομένων).

[18] Όπως στο παράδειγμα στην παράγραφο 7.2, όπου ορίζονται οι πράξεις "inside area" κτλ.

[19] δηλαδή μορφή εικόνας μη αναγνωρίσιμη από web browsers.


Σχήμα 13. Εμφάνιση αντικειμένου-εικόνας.

### *6.1.2. Επικοινωνία του Dienst με άλλους Servers Γεωγραφικών Δεδομένων*

Κατά την διάρκεια της παραπάνω εργασίας, προκειμένου τα αποτελέσματα αναζητήσεων στο dienst να περιέχουν ικανοποιητικό πλήθος δεδομένων, επιχειρήθηκε να υλοποιηθεί επικοινωνία του dienst με άλλους servers αντίστοιχων δεδομένων (federated search). Μια τέτοια συνεργασία με άλλα συστήματα ήταν εφικτή μόνο για την περίπτωση του συστήματος MEL, το οποίο και βασίζεται στην κλασσική φιλοσοφία των μηχανισμών αναζήτησης του Web[20].

Κατά την διαδικασία της κατανεμημένης αναζήτησης, Η αίτηση προς τον server MEL, στέλλεται όπως ακριβώς στέλνονται και οι αιτήσεις στους κατανεμημένους dienst servers, αλλά αλλάζει το πρωτόκολλο επικοινωνίας, δηλαδή ο τρόπος με τον οποίο δίνονται τα ορίσματα της αίτησης και η δομή της απάντησης. Υπάρχει δηλαδή ένα ενδιάμεσο στάδιο στην αποστολή της αίτησης, το οποίο αναλαμβάνει να μετατρέψει την αίτηση στην μορφή που είναι αναγνωρίσιμη από τον MEL server, και να μετατρέψει την απάντηση του MEL στην μορφή που είναι αναγνωρίσιμη από τον dienst server.

[20] Δηλαδή ο χρήστης κάνει «ένα» request και του επιστρέφονται τα αποτελέσματα σε μια HTML (ή γενικότερα formatted) σελίδα.

Ουσιαστικά έχουμε την έννοια του wrapper[21]. Σε ένα περισσότερο σύνθετο σύστημα, οι wrappers αποτελούν ξεχωριστές οντότητες του συστήματος. Αντιθέτως, σε αυτή την περίπτωση, τον wrapper τον ενσωματώσαμε στο ίδιο το dienst, προκειμένου να έχουμε καλύτερη απόδοση και να αποφύγουμε επικοινωνία με άλλα τμήματα.

## 6.2. Μια πρόταση για μια Ψ.Β. με Ωκεανογραφικά Δεδομένα

### *6.2.1. Η Ανάγκη δημιουργίας μιας νέας Αρχιτεκτονικής*

Στην πραγματικότητα, αυτά που ζητούμε από μια ψηφιακή βιβλιοθήκη με γεωγραφικά δεδομένα, είναι πολύ περισσότερα από αυτά που μπορούμε να πετύχουμε χρησιμοποιώντας το ίδιο το dienst και επεκτείνοντας το. Αν και η αρχιτεκτονική του dienst, μοιάζει αρκετά με αντικειμενοστραφής, στην πραγματικότητα δεν είναι. Αυτό διότι δεν μπορούμε να αποθηκεύσουμε μαζί με κάποιο αντικείμενο (μια εικόνα, ένα σύνολο από δεδομένα) και τις μεθόδους του, οι οποίες αναλαμβάνουν να κάνουν τις βασικές πράξεις / λειτουργίες πάνω σε αυτό. Η πιο κοινή και χρήσιμη μέθοδος στην πρώτη φάση είναι αυτή της εμφάνισης. Έτσι, δεδομένης της ποικιλότητας των μορφών δεδομένων, ιδίως στον χώρο των γεωγραφικών δεδομένων, υπάρχουν προβλήματα μετάφρασης και κατανόησης του τύπου με τον οποίο είναι αποθηκευμένα τα δεδομένα. Το πρόβλημα βέβαια δεν υφίσταται, αν όλοι γνωρίζουν όλους τους τύπους δεδομένων. Κάτι τέτοιο όμως, είναι ουσιαστικά ανέφικτο. Ιδίως όταν πρόκειται για χρήση από την επιστημονική κοινότητα, όπου η εμφάνιση και η χρήση ενός νέου τύπου δεδομένων είναι πολύ συχνή.

Έτσι, απλά και μόνο η ονομασία του τύπου των δεδομένων με το MIME-Type, δεν είναι αρκετή για την κατανόηση του τύπου του ίδιου. Θα ήταν αρκετή, αν υπήρχε ένας παγκόσμιος ορισμός από plug-ins, που περιέχουν διάφορες μεθόδους για τον κάθε τύπο. Κάτι τέτοιο όμως, αυτή τη στιγμή δεν υπάρχει, και είναι πολύ δύσκολο να υπάρξει.

Άρα αναγκαστικά, σε μια ψηφιακή βιβλιοθήκη, θέλουμε να μπορούμε να βρίσκουμε μεθόδους για την ανάλυση του κάθε τύπου δεδομένων. Συνεπώς, οδηγούμαστε σε κάποιο αντικειμενοστεφές μοντέλο.

---

[21] Συνήθως ως wrapper αναφέρουμε ένα ανεξάρτητο τμήμα ή διεργασία η οποία κάνει μετατροπή στο format δεδομένων. Και συνήθως θέλουμε να μπορούμε να προσθέτουμε wrappers στο σύστημα, είτε προσθέτοντας νέα προγράμματα που κάνουν άλλες μετατροπές, είτε προσθέτοντας μια νέα γραμματική στα δεδομένα που αναγνωρίζει ο wrapper. Η δεύτερη περίπτωση προτιμήθηκε για την υλοποίηση της μεταπτυχιακής εργασία ς του Pubudu Wariyapola στο Design Laboratory του MIT.

*6.2.1.1. Αντικειμενοστραφής Ψηφιακές Βιβλιοθήκες*

Η φύση του αντικειμένου των Ψηφιακών Βιβλιοθηκών είναι αντικειμενοστραφείς. Η ιδέα είναι ότι ο χρήστης πρέπει να μπορεί να ζητήσει από το σύστημα ένα αντικείμενο που είναι βιβλίο και να είναι δυνατό να το διαβάσει, ή μια εικόνα και να είναι δυνατό να την δει.

Βέβαια, αυτό, από πλευράς υλοποίησης είναι πολύ δύσκολο να γίνει, γιατί τότε τα αντικείμενα θα έπρεπε να περιλαμβάνουν μεθόδους εμφάνισής τους για όλα τα λειτουργικά συστήματα. Έτσι επινοήθηκε η πλάγια οδός, μέχρι τώρα χρησιμοποιούνται τα MIME-Types. Ένα αντικείμενο (αρχείο) είναι ενός τύπου MIME. Ο χρήστης είναι υπεύθυνος να εγκαταστήσει στο μηχάνημά του την εφαρμογή (plug-in) που μπορεί να εμφανίσει ένα αντικείμενο ενός συγκεκριμένου τύπου MIME.

Στην περίπτωση όμως των γεωγραφικών δεδομένων, η ποικιλομορφία και η ιδιαιτερότητα των μορφών αποθήκευσης κάνει δύσκολη την χρήση μόνο των τύπων MIME. Άλλωστε αυτοί προορίζονται κυρίως για να βοηθήσουν την εμφάνιση ενός αντικειμένου στον χρήστη, αλλά όχι την επεξεργασία του

*6.2.1.2. Χρήση της CORBA*

Αυτή τη στιγμή, αυτό που μπορεί να ικανοποιήσει τις απαιτήσεις μας είναι η χρήση της CORBA.

## *6.2.2. Η Αρχιτεκτονική του Συστήματος*

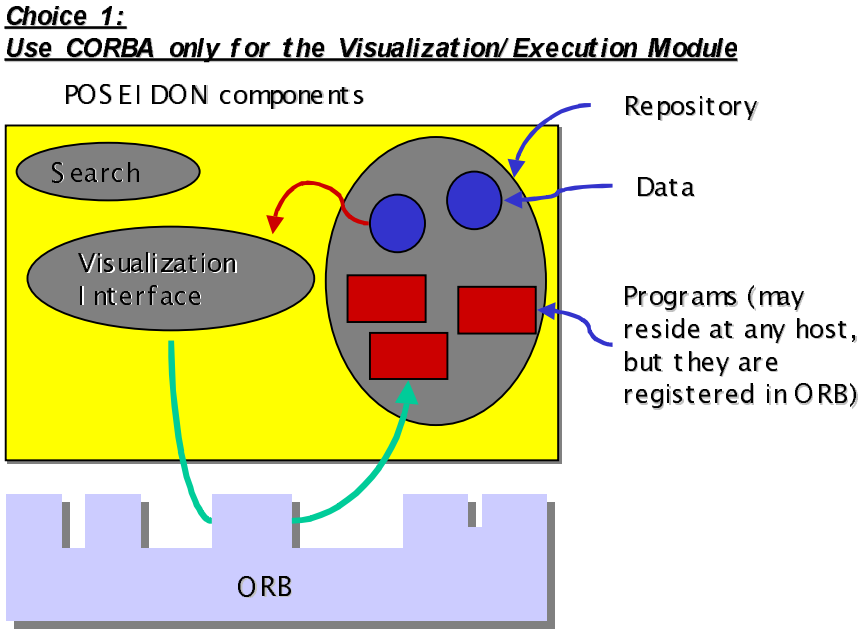
Σαν οδηγός για την σχεδίαση της αρχιτεκτονικής ενός τέτοιου συστήματος χρησιμοποιήθηκε το Poseidon Project ([XXIV], [XXV], [XXVI], [XXVII]) . Από μια πρώτη μελέτη, προέκυψε το συμπέρασμα ότι έχουμε 3 εναλλακτικές λύσεις, τουλάχιστον όσον αφορά το εύρος της χρήσης της CORBA (δηλαδή ποια τμήματα του συστήματος θα είναι objects). Παρακάτω θα παρουσιάσουμε μια σχεδίαση ενός τέτοιου συστήματος, σε πολύ υψηλό επίπεδο.

Κατ' αρχάς να αναφέρουμε ποια είναι τα βασικά τμήματα (modules) έχουμε στο σύστημα Poseidon:

- Search Engine: Η μηχανή αναζήτησης δεδομένων ή προγραμμάτων ή αλγορίθμων.
- Repositories: Τα φυσικά σημεία όπου τα δεδομένα /προγράμματα είναι αποθηκευμένα
- Visualization Interface: Το τμήμα μέσω του οποίου ο χρήστης μπορεί να οργανώσει ένα πείραμα συνδυάζοντας δεδομένα και προγράμματα που είναι αποθηκευμένα σε διάφορα μέρη. Με αυτό το τμήμα ο χρήστης μπορεί να ορίσει ότι θα πάρει τα δεδομένα χ από τον

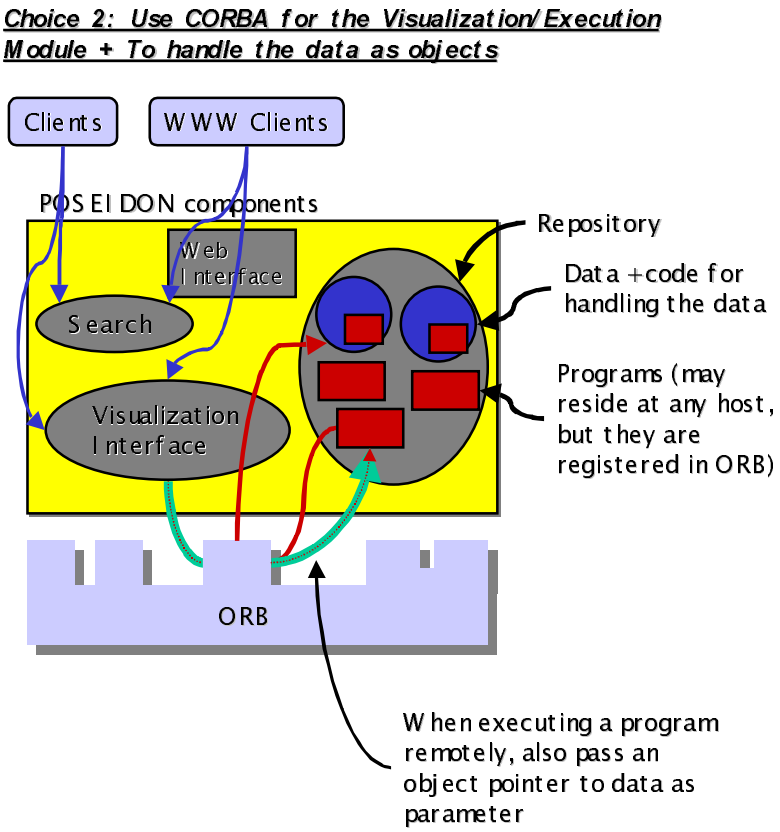
ψ server, τα οποία θα χρησιμοποιήσει ως είσοδο στο πρόγραμμα A, του οποίου την έξοδο θα την μετατρέψει στο format M, το οποίο θα δώσει ως είσοδο στο πρόγραμμα B.

- Visualization/Execution: Το τμήμα που αναλαμβάνει την εκτέλεση του παραπάνω διαγράμματος/ γραφήματος.



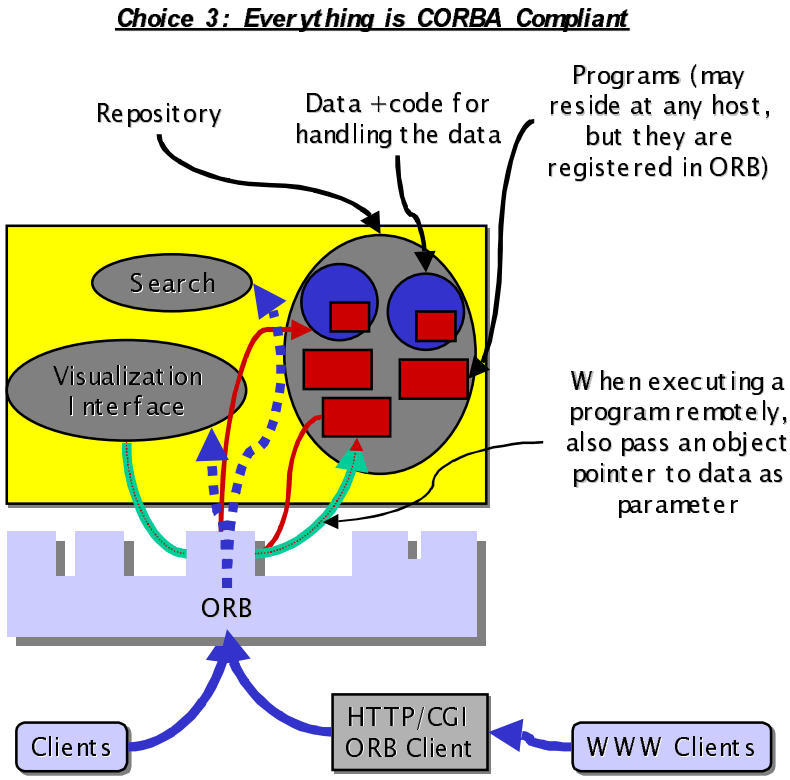

Σχήμα 14. Τα προγράμματα /αλγόριθμοι ως αντικείμενα CORBA.

Στο Σχήμα 14, βλέπουμε ένα διάγραμμα, στο οποίο θεωρούμε ότι τα προγράμματα, τα οποία είναι αποθηκευμένα στους Repository Servers, είναι δηλωμένα στο ORB ως CORBA objects. Έτσι, όταν μέσα από το Visualization τμήμα θέλουμε να εκτελέσουμε έναν αλγόριθμο με κάποια είσοδο, κάνουμε την αντίστοιχη κλήση προς την υπηρεσία που υποστηρίζει το αντικείμενο CORBA με τα κατάλληλα ορίσματα (π.χ. ένα όρισμα πρέπει να είναι οπωσδήποτε το που βρίσκονται τα αρχεία εισόδου). Σε αυτήν την περίπτωση πρέπει τα δεδομένα της εισόδου να έχουν ακριβώς την ίδια μορφή με αυτήν που απαιτεί ο εξυπηρετητής.




Σχήμα 15. Τα προγράμματα και τα δεδομένα ως αντικείμενα CORBA.

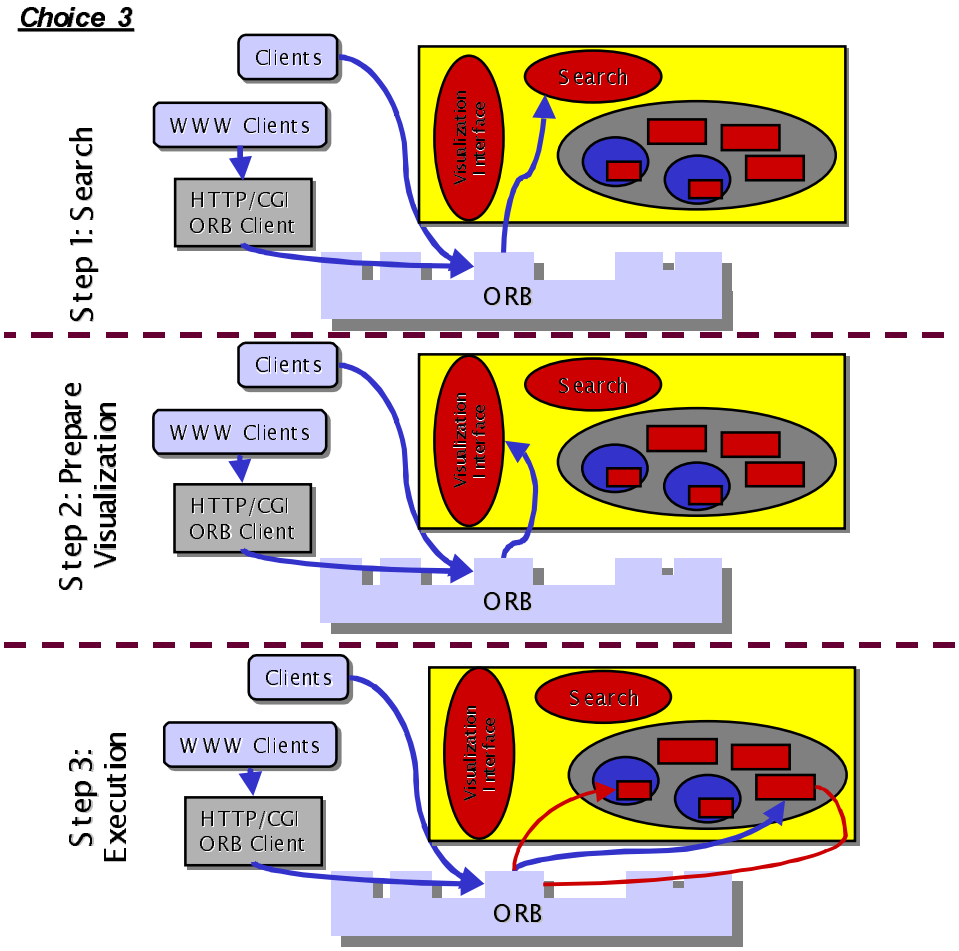
Στην δεύτερη περίπτωση, στο σύστημα είναι δηλωμένα ως αντικείμενα και τα δεδομένα. Έτσι τα ίδια τα δεδομένα-αντικείμενα περιλαμβάνουν μεθόδους πρόσβασης και ερμηνείας των ίδιων (δηλαδή μετατροπής του εαυτού τους), ώστε να μπορεί αυτός που τα διαβάζει να τα ερμηνεύσει σωστά. Έτσι όλη η διαδικασία του visualization γίνεται μέσω της CORBA. Ουσιαστικά, δηλαδή, οι wrappers της προηγούμενης περίπτωσης, ενσωματώνονται μέσα στα ίδια τα δεδομένα. Σε αυτήν την περίπτωση, οι εκδότες που κάνουν διαθέσιμα δεδομένα στο σύστημα, είναι υποχρεωμένοι να τα "εξοπλίζουν" με τις κατάλληλες μεθόδους ώστε να είναι αναγνώσιμα και αναγνωρίσιμα από τα υπόλοιπα τμήματα του συστήματος.




Σχήμα 16. Οι πελάτες (clients) επικοινωνούν με το σύστημα μέσω του ORB.

Στις 2 προηγούμενες περιπτώσεις, ο χρήστης-πελάτης επικοινωνούσε με κάποιο τμήμα του συστήματος (π.χ. search ή visualization UI), και αυτό το τμήμα επικοινωνούσε με τα υπόλοιπα μέσω του ORB. Σε αυτήν την περίπτωση, ακόμη και η πρόσβαση από κάποιον εξωτερικό πελάτη μπορεί να γίνει δια μέσου του ORB. Έτσι, σε αυτήν την περίπτωση, ακόμη και η αναζήτηση δεδομένων, μπορεί να γίνει μέσω της CORBA, δηλαδή μέσω των διαφόρων εργαλείων που έχουν αναπτυχθεί για αναζήτηση CORBA αντικειμένων. Ένα τέτοιο παράδειγμα είναι το VisiBroker Smart Agent, που συνοδεύει προαιρετικά τον ORB που έχει αναπτυχθεί από την Visigenic, και ο οποίος περιέχει τμήμα για αναζήτηση δεδομένων-αντικειμένων με βάση τις ιδιότητες τους.

Στο Σχήμα 17 φαίνονται πώς οι 3 βασικές λειτουργίες του συστήματος, η αναζήτηση, η προετοιμασία του visualization και η εκτέλεση του visualization, γίνονται μέσω του ORB.




Σχήμα 17. Παράδειγμα χρήσης του συστήματος της 3ης περίπτωσης

# *Κεφάλαιο 7*

## 7. Meta-Meta files

Προκειμένου να κάνουμε τις επεκτάσεις που αναφέρονται στο κεφάλαιο 6, επιλέξαμε να μην κάνουμε απλά αλλαγές ώστε να προσαρμόσουμε το dienst μόνο για γεωγραφικά δεδομένα, αλλά να υλοποιήσουμε μια γενικότερη έκδοση, η οποία με το κατάλληλο configuration να προσαρμόζεται σε οποιοδήποτε τύπου συλλογή. Σε αυτό το κεφάλαιο θα δούμε τις αλλαγές που έγιναν στο dienst, και πώς μπορούμε να ορίσουμε έναν νέο τύπο στοιχείων.

### 7.1. Αρχεία Παραμέτρων

Στους παρακάτω πίνακες φαίνονται τα στοιχεία που ορίζουμε για την παραμετροποίηση του dienst server. Στον πρώτο πίνακα (Πίνακας 6) ορίζονται οι τύποι των αντικειμένων που διαχειρίζεται ο server. Στην πρώτη υλοποίηση ορίσαμε έναν τύπο (με το όνομα data) για όλους τους τύπους των δεδομένων, αλλά είναι δυνατό διάφορα δεδομένα να έχουν τόσο διαφορετικές ιδιότητες, ώστε να είμαστε αναγκασμένοι να ορίσουμε νέο τύπο.

| *@collections* |
|---|
| "data" |
| "programs" |
| "documents" |

Πίνακας 6. Τύποι αντικειμένων

Στον παρακάτω πίνακα (Πίνακας 7), ορίζονται οι τύποι όλων των πεδίων που μπορεί να αναγνωρίσει ο server. Αν κάποιο πεδίο δεν είναι ορισμένο στον παρακάτω πίνακα, τότε αυτό αγνοείται κατά την διαδικασία της αναζήτησης και της περιήγησης. Εμφανίζεται όμως στα στοιχεία του αντικειμένου που το περιέχει σαν απλό πεδίο κειμένου.

Σε αυτό το σημείο θα μπορούμε να αναφέρουμε για ποιον λόγο χρησιμοποιήθηκαν οι τύποι "name", "text" και "words", εφόσον ουσιαστικά είναι όλοι του τύπου string. Στην περίπτωση του συγγραφέα (author) ο τύπος είναι "name" διότι στην περίπτωση ονομάτων, συνήθως θέλουμε να ξεχωρίζουμε το μικρό όνομα από το επώνυμο. Αυτό γίνεται και στην κανονική έκδοση του dienst όταν το όνομα του πεδίου είναι "author", προκειμένου να δημιουργείται λίστα των συγγραφέων (αλφαβητική) με βάση το επώνυμο. Με την νέα ρύθμιση μπορεί πολύ εύκολα να δημιουργείται λίστα των προγραμματιστών (programmers) ή δημιουργών (creators) ή οτιδήποτε έχει να κάνει με όνομα. Για τις περιπτώσεις "words" και "text" η διαφορά είναι μικρή, και δεν είναι ουσιαστική χρησιμοποιώντας τον παρών μηχανισμό ευρετηριασμού. Η

ιδέα είναι ότι στην περίπτωση του κειμένου (text), μας ενδιαφέρει (αν ο μηχανισμός ευρετηριασμού είναι έξυπνος) ακόμη και η σειρά των λέξεων ή η απόσταση μεταξύ τους. Έτσι μπορεί να γίνει καλύτερη κατηγοριοποίηση του κειμένου ή εύρεση της θεματικής περιοχής. Στην περίπτωση των λέξεων (words) δεν μας ενδιαφέρει κάτι τέτοιο, διότι σε αυτήν την περίπτωση οι λέξεις είναι ισοδύναμες μεταξύ τους (π.χ. οι λέξεις κλειδιά-keywords).

| *%data_types* | |
|---|---|
| "author" | "name" |
| "data" | "text" |
| "type" | "words" |
| "language" | "words" |
| "title" | "text" |
| "keywords" | "words" |
| "programmer" | "name" |
| "abstract" | "text" |
| "date" | "date" |
| "entry" | "date" |
| "organization" | "text" |
| "region" | "text" |
| "signature" | "text" |
| "latitude_min" | "int" |
| "latitude_max" | "int" |
| "longtitude_min" | "int" |
| "longtitude_max" | "int" |
| "coords" | "vector" |
| "entry" | "date" |
| "month" | "date" |

Πίνακας 7. Τύποι πεδίων

Άλλες πληροφορίες που χρειάζεται το σύστημα, είναι: ποια πεδία χρησιμοποιούνται από κάθε συλλογή, με βάση ποια πεδία μπορεί ο χρήστης να κάνει αναζήτηση (searchable) και με βάση ποια πεδία μπορεί να γίνει περιήγηση (browsable). Αυτές οι πληροφορίες ορίζονται στους δυο επόμενους πίνακες (Πίνακας 8, Πίνακας 9). Τα πεδία που περιέχονται στον πίνακα «search_fields» είναι αυτά που θα εμφανίζονται στην φόρμα αναζήτησης, ενώ τα πεδία που περιέχονται στον πίνακα «browse_collection_fields» είναι αυτά με βάση τα οποία ο χρήστης θα μπορεί να κάνει περιήγηση στην συλλογή.

| *%search_fields* | |
|---|---|
| 'data' | 'title:region:date:latitude_min:latitude_max:longtitude_min:longtitude_max:coords' |
| 'programs' | 'input:output:language:programmer:organization:entry' |
| 'documents' | 'author:title:abstract:date:keywords:entry' |

Πίνακας 8. Πεδία με βάση των οποίων μπορεί να γίνει αναζήτηση

| *%browse_collection_fields* | |
|---|---|
| 'data' | 'title:region:date' |
| 'programs' | 'input:output:programmer:organization:entry:date' |
| 'documents' | 'author:date' |

Πίνακας 9. Πεδία με βάση των οποίων μπορεί να περιηγηθεί ο χρήστης στην συλλογή

| %indexer_info | |
|---|---|
| "words" | "split" |
| "text" | "split" |
| "name" | "split" |
| "date" | "nosplit" |
| "vector" | "nosplit" |

Πίνακας 10. Πληροφορίες για τα σύνθετα πεδία

Επίσης, δίνεται η πληροφορία στον indexer, πως θα διαχειριστεί τα σύνθετα πεδία. Έτσι με βάση τον παραπάνω πίνακα (Πίνακας 10), ο indexer θα διαχωρίσει τις λέξεις για τα πεδία "words", "text", και "name" ενώ όχι για τα πεδία τύπου "date" και "vector".

## 7.2. Index Server

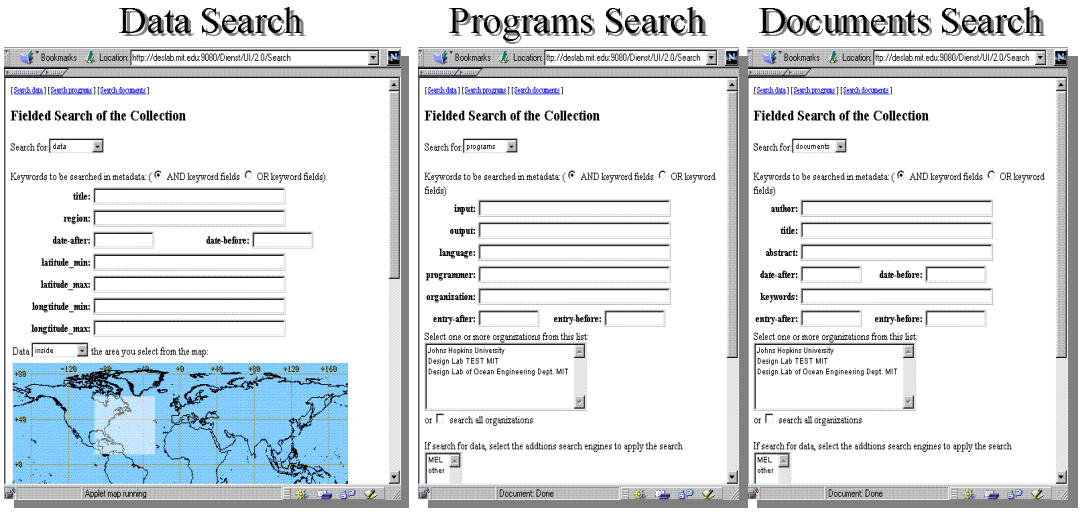
Με βάση τα παραπάνω δημιουργούνται τα αρχεία ευρετηριασμού (index files). Τα αρχεία ευρετηριασμού είναι αντεστραμμένες λίστες (inverted files), όπως ήταν και στο «παραδοσιακό» dienst για την συλλογή κειμένων. Βέβαια, πλέον τα πεδία δεν ερμηνεύονται σαν απλά πεδία κειμένου, όπως γινόταν, αλλά με βάση τον τύπο τους (int, date κτλ.). Προφανώς, για κάθε νέο τύπο που ορίζεται, πρέπει να οριστούν και οι πράξεις που μπορούν να γίνουν με βάση αυτόν, και καταρχάς ο έλεγχος ισότητας ή ανισότητας. Έτσι μπορούμε να κάνουμε αναζήτηση με βάση κάποιο στοιχείο (π.χ. Ημερομηνία) και να ζητήσουμε δεδομένα μετά ή πριν από κάποια ημερομηνία. Επίσης για το πεδίο "coords" (coordinates – συντεταγμένες), έχουν οριστεί οι πράξεις "inside area" και "intersection with area". Στην πρώτη περίπτωση ψάχνουμε για περιοχές που βρίσκονται μέσα στην περιοχή που ορίζουμε, ενώ στην δεύτερη για περιοχές που έχουν μη μηδενική τομή με την περιοχή που ορίζουμε. Προφανώς, υπάρχει η δυνατότητα από τον υπεύθυνο του συστήματος, να ορίσει οποιονδήποτε τύπο δεδομένων και οποιαδήποτε πράξη πάνω σε αυτόν.

## 7.3. User Interface

Πλέον υπάρχει η ανάγκη για την δημιουργία ενός δυναμικού interface, το οποίο αναπροσαρμόζεται ανάλογα με τον τύπο αντικειμένων που ψάχνουμε.

Ένα σημείο στο οποίο επιδρά αναγκαστικά η αλλαγή των meta-αρχείων, είναι η φόρμα αναζήτησης σε έναν τέτοιο server, η οποία πρέπει να προσαρμόζεται ανάλογα με τον τύπο του αντικειμένου που ψάχνει ο χρήστης. Στο παράδειγμα της προηγούμενης παραγράφου, ορίσαμε τριών τύπων δεδομένα. Οι αντίστοιχες φόρμες αναζήτησης φαίνονται στο Σχήμα 18. Ο χρήστης έχει την δυνατότητα να αλλάξει τον τύπο δεδομένων που ψάχνει χρησιμοποιώντας το πρώτο πεδίο της φόρμας.

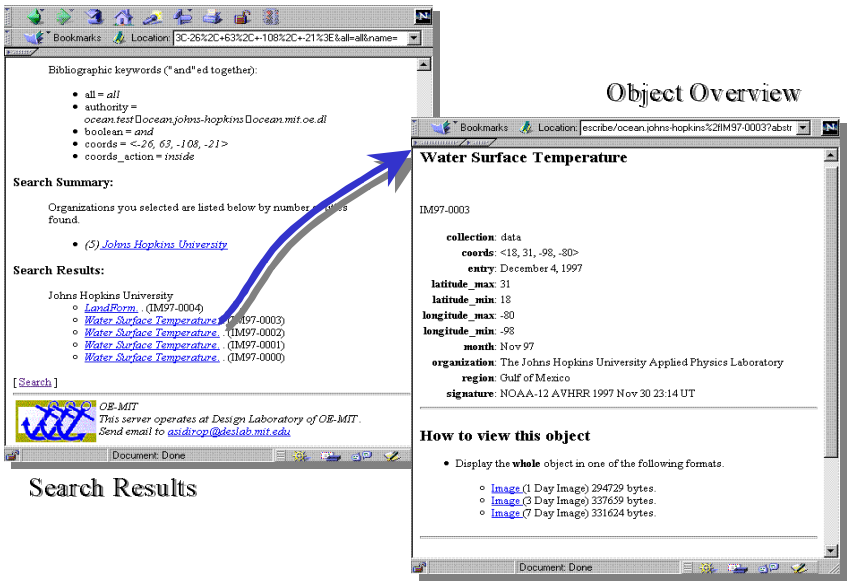
Βλέπουμε επίσης, ότι αν ο τύπος δεδομένων που αναζητούμε έχει την ιδιότητα της τοποθεσίας, τότε εμφανίζεται και ένας χάρτης, στο οποίο ο χρήστης μπορεί να ορίσει μια περιοχή και να αναζητήσει δεδομένα-αντικείμενα τα οποία περιέχονται σε αυτήν την περιοχή ή έχουν μη μηδενική τομή.



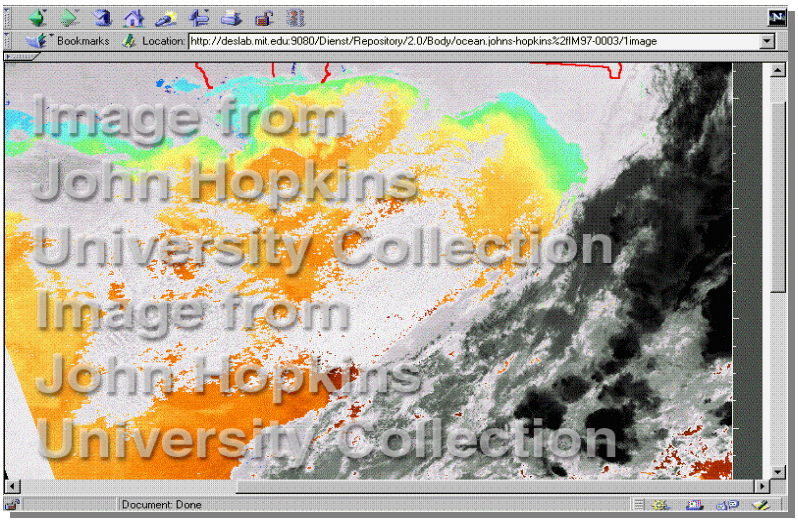

Σχήμα 18. Αναζήτηση στα διάφορα τμήματα της συλλογής

Στο Σχήμα 19 φαίνεται η σελίδα αποτελεσμάτων μιας αναζήτησης και η σελίδα εμφάνισης περίληψης για ένα από τα αντικείμενα που περιέχονται στην λίστα. Απ' ότι φαίνεται η σελίδα εμφάνισης των αποτελεσμάτων δεν έχει καμιά διαφορά από αυτή του αρχικού dienst. Η σελίδα όμως εμφάνισης πληροφοριών για ένα αντικείμενο, έχει αλλάξει, αφού αυτή πρέπει να περιέχει όλα τα πεδία των meta data που υπάρχουν για αυτό το αντικείμενο (και όχι μόνο τίτλο, συγγραφείς και περίληψη).

Στο παραπάνω παράδειγμα, όλες οι μορφές στις οποίες υπάρχει το συγκεκριμένο αντικείμενο, είναι απλές εικόνες (GIF ή JPEG, mimes-types: image/gif, image/jpeg), οπότε και η εμφάνισή τους είναι τετριμμένη (Σχήμα 20). Αν ο τύπος είναι διαφορετικός, τότε ο χρήστης θα πρέπει να αντιγράψει (download) το αντίστοιχο αρχείο και να το επεξεργαστεί είτε με κάποιο plug-in του browser, είτε με κάποιο άλλο πρόγραμμα που μπορεί να αναγνωρίσει την αντίστοιχη μορφή.




Σχήμα 19. Αποτελέσματα αναζήτησης γεωγραφικών δεδομένων.




Σχήμα 20. Εμφάνιση αντικειμένου - εικόνας[22]

[22] Το υδατογράφημα που εμφανίζεται στο Σχήμα 20, δεν είναι τμήμα της εικόνας ή του συστήματος. Χρησιμοποιείται μόνο στην παρούσα δημοσίευση.

# Μέρος 3ο:
# Το σύστημα dienst Spider

# *Κεφάλαιο 8*

# 8. Dienst Spider

## 8.1. Εισαγωγικά

Κύριος στόχος μας σε αυτήν την περίπτωση, είναι να φτιάξουμε έναν "spider", ειδικά προσαρμοσμένο στις ανάγκες του dienst, και έναν μηχανισμό αναζήτησης ο οποίος θα αναλαμβάνει την αναζήτηση στα meta data. Τα αρχεία ευρετηριασμού, είναι αποθηκευμένα τοπικά, συνεπώς μπορούμε να έχουμε πάρα πολύ καλή απόδοση στην ταχύτητα απόκρισης. Έτσι σε αυτό το τμήμα χρησιμοποιούνται 2 κυρίως τεχνολογίες, αυτή των "spiders" και αυτή των μηχανισμών ευρετηριασμού.

Σε αυτό το κεφάλαιο θα παρουσιάσουμε την σχεδίαση της αρχιτεκτονικής.

### *8.1.1. Spiders Γενικά*

Τους spiders μπορούμε να τους κατατάξουμε σε 2 κύριες κατηγορίες: Στους spiders του web οι οποίοι που περιηγούνται στον χώρο του διαδικτύου και μαζεύουν html κείμενα, και στους εξειδικευμένους spiders οι οποίοι συλλέγουν συγκεκριμένες πληροφορίες ή αρχεία., π.χ. να δουν ποιοι είναι οι περισσότερο χρησιμοποιούμενοι http servers ή να συλλέξουν άλλου είδους πληροφορίες ή να συλλέξουν συγκεκριμένους τύπους αρχείων[23] ή να δημιουργήσουν ευρετήρια για ftp sites ή για home pages χρηστών.

Σε αυτήν την κατηγορία (στους εξειδικευμένους spiders) μπορούμε να κατατάξουμε και τον spider του dienst.

Στην διεύθυνση: *http://info.webcrawler.com/mak/projects/robots/active/html/type.html* ([XXII]) υπάρχει μια εκτενής λίστα με διάφορους spiders που είναι ενεργοί ή που βρίσκονται υπό ανάπτυξη.

[23] Πχ. Ένας spider μπορεί να συλλέγει εικόνες (που ικανοποιούν κάποιες προδιαγραφές) ή μουσικά κομμάτια που βρίσκει στο web.

### *8.1.2. Μηχανισμοί Αναζήτησης*

Οι spiders που συλλέγουν html κείμενα χρησιμοποιούνται, σχεδόν αποκλειστικά, σε συνεργασία με κάποιον μηχανισμό αναζήτησης, ώστε να δημιουργηθούν ευρετήρια για τα κείμενα που συλλέχθηκαν και να επιτρέπεται αναζήτηση σε αυτά. Υπάρχουν πολλά συστήματα που δημιουργούν ευρετήρια για html σελίδες. Τα περισσότερο διαδεδομένα από αυτά, είναι τα freeWAIS ([XXX], [XXIX], [XXVIII]), glimpse ([XXXI]), excite ([IV]), και φυσικά οι μηχανισμοί ευρετηριασμού που χρησιμοποιούν και τα περισσότερο γνωστά sites του internet όπως Altavista ([I]), lycos ([V]), κ.τ.λ.

Τα περισσότερα από αυτά τα συστήματα είναι ικανά να αναγνωρίσουν μόνο html κείμενα (διότι για αυτόν τον λόγο δημιουργήθηκαν). Βέβαια, υπάρχουν και μηχανισμοί που αναγνωρίζουν οποιαδήποτε δομημένη μορφή αρχείων την οποία την καθορίζει ο υπεύθυνος του συστήματος[24] (π.χ. glimpse[25]).

Τα περισσότερα από αυτά τα συστήματα χρησιμοποιούν ως τρόπο ευρετηριασμού την μέθοδο των αντεστραμμένων αρχείων (inverted files ή inverted lists).

### *8.1.3. Αντίστοιχα Συστήματα*

Κατά την συγγραφή του παρόντος μας ήταν γνωστό μόνο ένα σύστημα που συνδυάζει την τεχνολογία των εξειδικευμένων spiders με ευρετηριαστή προκαθορισμένης δομημένης μορφής αρχείων: το UCSTRI (Unified Computer Science TR Index).

Η λειτουργία του UCSTRI είναι αντίστοιχη με αυτήν του dienst spider, δηλαδή δημιουργία ευρετηρίων για τεχνικές αναφορές επιστήμης υπολογιστών. Κατά την δημιουργία του UCSTRI (1993), ο μόνος τρόπος ηλεκτρονικής δημοσίευσης των τεχνικών αναφορών, ήταν η εισαγωγή τους σε κάποιο ftp site. Έτσι το UCSTRI περιηγείται στα ftp sites και ψάχνει για αρχεία που είναι τεχνικές αναφορές. Έτσι είναι δυνατό για το σύστημα να βρει τα TRs, οπότε και δημιουργεί ευρετήρια για αυτά χρησιμοποιώντας τα αντίστοιχα αρχεία βιβλιογραφικών αναφορών που περιέχονται μαζί με τα ίδια τα κείμενα. (http://www.cs.indiana.edu:800/cstr/ [XXIII]).

---

[24] Για τον λόγο αυτό (η γενικότητα της δυνατής χρήσης), οδηγεί στο να πάσχουν αυτά τα συστήματα όσον αφορά την απόδοση (ταχύτητα αναζήτησης).

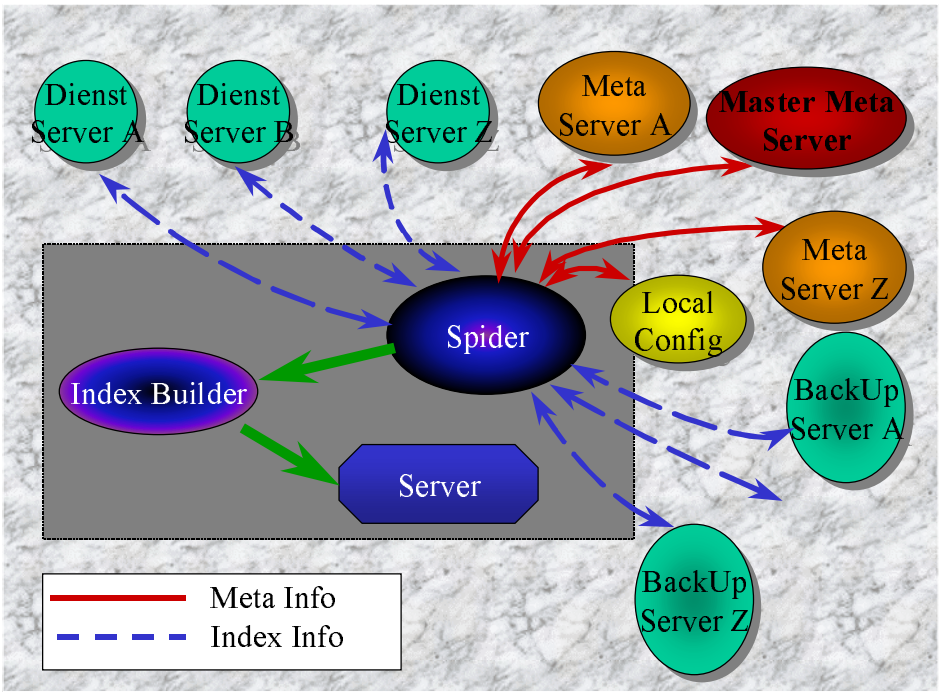
[25] Στις συγκριτικές δοκιμές, οι οποίες αναφέρονται στο κεφάλαιο 11, συγκρίνεται ο μηχανισμός αναζήτησης που αναπτύξαμε με το dienst που χρησιμοποιεί ως μηχανισμό αναζήτησης το glimpse.

## 8.2. Τμήματα του συστήματος

Το σύστημα αποτελείται από τα εξής τμήματα:

- Spider module
- Indexer Module
- Index Server/Web Server
- MetaData Database

Στο παρακάτω σχήμα (Σχήμα 21) φαίνονται τα τμήματα του συστήματος και πως αυτά επικοινωνούν μεταξύ τους. Στα επόμενα υποκεφάλαια θα δούμε περισσότερο αναλυτικά το κάθε τμήμα και τις λειτουργίες του.




Σχήμα 21. Τα τμήματα του spider

### *8.2.1. Τμήμα του Spider (Spider Module)*

Αυτό το τμήμα αναλαμβάνει να μαζεύει από την συλλογή NCSTRL (και κατ' επέκταση οποιαδήποτε συλλογή βασισμένη στην αρχιτεκτονική dienst) τα meta data των αντικειμένων (κειμένων) που είναι αποθηκευμένα στην συλλογή.

Αυτή τη στιγμή ο spider είναι ικανός να επικοινωνήσει με οποιοδήποτε dienst server που υποστηρίζει την 4η έκδοση (version 4.0) του συστήματος και έκδοση 2.0 του πρωτοκόλλου επικοινωνίας ή μεγαλύτερη. Για τους servers που υποστηρίζουν μόνο την έκδοση 3 του συστήματος, ο spider βρίσκει κάποιον άλλον server που έχει τα ίδια δεδομένα και υποστηρίζει την έκδοση 4 [26](π.χ. οι regional index servers). Είναι βέβαια δυνατή η επέκταση του συστήματος ώστε να υποστηρίζει, όχι μόνο κάποια άλλη έκδοση του dienst, αλλά και κάποιο άλλο πρωτόκολλο, αρκεί ο τύπος των meta data να μοιάζουν με αυτόν του dienst.

Επίσης το τμήμα αυτό, περιλαμβάνει ένα υποτμήμα αποθήκευσης στατιστικών στοιχείων, όσον αφορά τον χρόνο απόκρισης των servers, από τους οποίους παίρνει δεδομένα, και τον ρυθμό μεταφοράς των δεδομένων. Τα στατιστικά αυτά χρησιμοποιούνται στις περιπτώσεις όπου ο spider, προκειμένου να συλλέξει meta data από κάποιον εκδότη, έχει την ευχέρεια να επιλέξει από ένα σύνολο από servers (τον server του ίδιου του εκδότη - Repository server, ή κάποιον από τους servers τήρησης αντιγράφων περιοχής - Regional Backup Server). Χρησιμοποιώντας τα παραπάνω στατιστικά στοιχεία ο spider επικοινωνεί με τον server που ανταποκρίνεται γρηγορότερα, και αυτό γίνεται την ώρα της ημέρας κατά την οποία συνήθως επιτυγχάνονται οι μέγιστες ταχύτητες σύμφωνα με τα στατιστικά.

Ο spider επίσης, μπορεί να περιλάβει και ένα τμήμα υπολογισμού ελλιπής μέτα-πληροφορίας (όπου αυτό είναι δυνατό). Τέτοιου είδους μετα-πληροφορία είναι για παράδειγμα το πλήθος των σελίδων του κειμένου ή ο καθορισμός της γλώσσας στην οποία είναι γραμμένο το κείμενο (στην περίπτωση που αυτές οι πληροφορίες έχουν παραληφθεί από τα meta data, και είναι δυνατό να υπολογιστούν).

### *8.2.2. Ευρετηριαστής (Indexer Module)*

Έχει γίνει υλοποίηση ενός δημιουργού αρχείων ευρετηριασμού (indexer), χρησιμοποιώντας αντίστροφες λίστες (inverted files). Ο indexer είναι βελτιστοποιημένος όσον αφορά την απόδοση όταν χρησιμοποιούνται meta data για τεχνικές αναφορές του τύπου του dienst.

---

[26] Αυτό επιλέχθηκε , ώστε να μην μεγαλώσουμε το σύστημα, απλά για να έχουμε συμβατότητα προς τα πίσω, δεδομένου ότι είναι δυνατή η λειτουργία του συστήματος και χωρίς αυτήν.

### *8.2.3. Εξυπηρετητής (Index Server/Web Server Module)*

Έχει γίνει υλοποίηση ενός index server ο οποίος χρησιμοποιεί τις παραπάνω αντίστροφες λίστες. Επίσης περιλαμβάνεται:

- Η σχεδίαση του πρωτοκόλλου των αιτήσεων προς τον server, το οποίο βασίζεται στο πρωτόκολλο του dienst (URL based protocol, dienst like architecture) με αρκετές βελτιώσεις.
- Βελτιστοποίηση του index server όσον αφορά την απόδοσή του κατά την διαδικασία της αναζήτησης, σε σχέση με τα παρακάτω.
  - χρόνος αναζήτησης,
  - ελάχιστη πρόσβαση στον δίσκο κατά την αναζήτηση,
  - ελάχιστη χρήση της εικονικής μνήμης (virtual memory/swap file) του συστήματος
  - ελάχιστη καθυστέρηση από τον web server. Για να επιτευχθεί αυτό υποστηρίζεται το πρωτόκολλο HTTP/1.0 από τον ίδιο τον index server ώστε να αποφεύγεται το overhead επικοινωνίας «httpd↔cgi-bin↔index server», καθώς και το overhead που προσθέτει ο ίδιος ο httpd.

  Η ανάγκη για τις παραπάνω βελτιστοποιήσεις έχει προκύψει από την μεταπτυχιακή εργασία του Σωτήρη Τερζή καθώς και από την εμπειρία μας στην χρήση ενός συστήματος για ευρετηριασμό όπως το dienst.
- Υλοποίηση της αναζήτησης που περιλαμβάνει μερικές προσθέσεις στην δυνατότητα του dienst για αναζήτηση, όπως:
  - Αναζήτηση με λέξεις κλειδιά και περισσότερο ευέλικτες λογικές πράξεις (and/or/not ) απ' ότι το dienst.
  - Υπολογισμός του σκορ κάθε αποτελέσματος σε κάποια ερώτηση (πόσο πιθανόν είναι το κείμενο που επιστράφηκε να ανταποκρίνεται στην ερώτηση του χρήστη). Έτσι γίνεται και μια ταξινόμηση των αποτελεσμάτων.
  - Δυνατότητα για αναζήτηση σχετικών κειμένων. (Ο χρήστης έχει βρει ένα κείμενο που τον ενδιαφέρει και ψάχνει σχετικά κείμενα με αυτό).

Από τις πρώτες εκδόσεις του συστήματος οι αναζητήσεις στα ίδια δεδομένα με την ίδια ερώτηση στον Index Server του spider και στον Index Server του dienst, στον ίδιο σταθμό εργασίας (estia.csi.forth.gr / pleiades.csi.forth.gr), έδειξαν ότι η ταχύτητα αναζήτησης στον νέο server είναι τάξεις μεγαλύτερη από ότι στο dienst. (Χωρίς να έχουν γίνει ακόμη μετρήσεις η διαφορά στην ταχύτητα αναζήτησης είναι αισθητή στον χρήστη). Περισσότερα για αυτό το θέμα στο κεφάλαιο 11.

## 8.3. Λειτουργία του συστήματος

Σε αυτή τη παράγραφο θα παρουσιάσουμε την λειτουργία του συστήματος του dienst spider για κάθε τμήμα ξεχωριστά.

### *8.3.1. Λειτουργία του τμήματος spider*

#### *8.3.1.1. Έναρξη Λειτουργίας*

Αρχικά, το τμήμα αυτό, πρέπει να γνωρίζει κάποιες βασικές πληροφορίες για την λειτουργία του. Πρέπει να του έχει οριστεί ποιος είναι ο master meta server της συλλογής για την οποία θα δημιουργήσει τα αρχεία ευρετηριασμού, και προαιρετικά κάποιους meta servers (regional). Από αυτούς του servers, ο spider θα μάθει τις πληροφορίες για τον «κόσμο» που τον ενδιαφέρει. Επίσης ο υπεύθυνος του συστήματος να συμπεριλάβει οποιονδήποτε άλλον dienst server (που δεν ανήκει στην συλλογή), και οποιονδήποτε άλλον meta server (όχι όμως master meta server).

Αρχικά λοιπόν, ο spider επικοινωνεί με τον master meta server, από τον οποίο μαθαίνει ποιοι είναι οι υπόλοιποι (regional) meta servers. Έπειτα θα επικοινωνήσει με όλους τους (regional) meta servers για τους οποίους γνωρίζει (ή με όσους μπορέσει), για να μάθει όσο γίνεται για περισσότερους index servers (απλούς ή backup index servers). Εδώ θα πρέπει να σημειώσουμε, ότι επειδή ο υπεύθυνος του συστήματος μπορεί να ορίσει και αυτός ένα σύνολο από meta servers, ο spider θα επικοινωνήσει και με αυτούς. Άρα μπορούμε να δούμε και εξωτερικές συλλογές από αυτήν της ncstrl - ercim.

#### *8.3.1.2. Λειτουργία*

Ο spider χωρίζεται σε 2 τμήματα. Το πρώτο τμήμα, βρίσκεται σε έναν βρόγχο στον οποίο:

- Ελέγχει για κάθε εκδότη ποια κείμενα υπάρχουν στην συλλογή του (με την αίτηση /Dienst/Info/2.0/Contents).
    - Αν υπάρχουν νέα κείμενα, τα οποία δεν τα έχει «μαζέψει» ο spider, τότε δημιουργείται μια λίστα με τα αναγνωριστικά των κειμένων αυτών, η οποία και αποθηκεύεται.

Το δεύτερο τμήμα, είναι αυτό που ελέγχει την λίστα με τα αναγνωριστικά των κειμένων που πρέπει να «μαζέψει» ο spider.

1. Αρχικά, αναγνωρίζονται οι εκδότες στους οποίους ανήκουν τα κείμενα που πρέπει να «πάρει» ο spider.

2. Για κάθε εκδότη, ελέγχεται από ποιους servers μπορεί να πάρει τα δεδομένα, από αυτούς, ανάλογα με τα στατιστικά στοιχεία, επιλέγεται ο καλύτερος για την αντίστοιχη στιγμή.

2.1. Γίνεται προσπάθεια για επικοινωνία με τον καλύτερο server

2.2. Αν τα δεδομένα μεταφερθούν επιτυχώς τότε προχωράμε στο βήμα 3.

2.3. Αν όχι, τότε επιλέγουμε τον επόμενο καλύτερο server, και επαναλαμβάνεται το βήμα 2.1. Αν δεν υπάρχει άλλος server, τότε πάμε στο βήμα 2 για τον επόμενο εκδότη[27].

3. Αφού έχουν μεταφερθεί τα δεδομένα, τότε υπολογίζονται τα συμπληρωματικά δεδομένα (αν η επιλογή είναι ενεργοποιημένη).

4. Δημιουργείται μια λίστα με τα νέα κείμενα που έχουν προστεθεί στην βάση, η οποία και αποθηκεύεται.

5. Αφαιρούνται τα αντίστοιχα κείμενα, από την λίστα των εκκρεμών κειμένων για μεταφορά.

6. Επαναλαμβάνουμε το βήμα 2 για τον επόμενο εκδότη, αν δεν υπάρχει επόμενος τότε:

6.1. αν η λίστα των εκκρεμών κειμένων για μεταφορά δεν είναι κενή, πάμε στο βήμα 1

6.2. αν είναι κενή τότε περιμένουμε μέχρι να μπει κάποιο στοιχείο στην λίστα. Όταν μπει κάποιο στοιχείο τότε πάμε στο βήμα 1.

### *8.3.2. Λειτουργία του δημιουργού ευρετηρίου (Index Builder)*

Αυτό το τμήμα χρησιμοποιείται για την δημιουργία των αρχείων ευρετηριασμού, τα οποία βασίζονται στην μέθοδο των αντεστραμμένων λιστών.

Αρχικά να αναφέρουμε ότι μπορεί να λειτουργήσει με 2 τρόπους. Ο πρώτος είναι δημιουργία των αρχείων ευρετηριασμού από την αρχή, και ο δεύτερος είναι επέκταση των αρχείων ευρετηριασμού, δηλαδή πρόσθεση μόνο των νέων αντικειμένων που εισαχθήκανε στην βάση.

[27] Στην περίπτωση αυτή, τα κείμενα θα παραμείνουν στην λίστα εκκρεμών κειμένων, οπότε και θα ξαναγίνει προσπάθεια για μεταφορά τους αργότερα.

Για την περιγραφή της λειτουργίας του, ας πάρουμε την πρώτη περίπτωση. Η δεύτερη περίπτωση είναι αντίστοιχη, μόνο που φορτώνονται τα αρχεία ευρετηριασμού που υπάρχουν ήδη.

- Για κάθε αντικείμενο που υπάρχει στην βάση:
    - Γίνεται ανάγνωση του αρχείου με τα meta data, αναγνώριση των διαφόρων πεδίων.
    - Διαχωρισμός των λέξεων των κειμένων
    - Μέτρηση του πλήθους εμφάνισης της κάθε λέξης.
    - Υπολογισμός των ριζών των λέξεων (stemming)[28]
    - Μέτρηση του πλήθους εμφάνισης της κάθε ρίζας λέξης
    - Πρόσθεση της λέξης στην λίστα των λέξεων που έχουν εμφανιστεί
    - Πρόσθεση της ρίζας της λέξης στην αντίστοιχη λίστα ριζών λέξεων
    - Πρόσθεση του αναγνωριστικού του κειμένου και του πλήθους εμφανίσεων στις δυο παραπάνω λίστες αντίστοιχα. Στην πληροφορία πλήθους εμφανίσεων, στην πραγματικότητα, κρατάμε 4 νούμερα: το πρώτο είναι το συνολικό πλήθος εμφάνισης, το δεύτερο είναι το πλήθος εμφάνισης στον τίτλο του κειμένου, το τρίτο στους συγγραφείς των κειμένων, το τέταρτο στην περίληψη του κειμένου και το τέταρτο στα υπόλοιπα πεδία. Αυτό διότι είναι προφανές ότι άλλο βάρος έχει μια λέξη αν εμφανίζεται στον τίτλο, και άλλο αν εμφανίζεται σε κάποιο άλλο πεδίο.
    - Επίσης, υπολογίζεται για κάθε ρίζα λέξης, πόσες φορές συναντάται αυτή συνολικά (σε όλη τη συλλογή). Αυτό θα χρησιμοποιηθεί για τον υπολογισμό του βαθμού ομοιότητας (βλέπε παράγραφο 8.7).
- Αφού ολοκληρώσουμε για όλα τα κείμενα, τότε αποθηκεύονται τα αρχεία ευρετηριασμού. Αφού γίνει αυτό, τότε πρέπει να στείλουμε ένα σήμα στον server, ώστε να ξαναφορτώσει τα αρχεία ευρετηριασμού.

[28] Το stemming γίνεται μόνο για τις αγγλικές λέξεις, και ο αλγόριθμος που χρησιμοποιείται είναι ο ίδιος που χρησιμοποιείται από τον μηχανισμό ευρετηριασμού "free-WAIS".

### *8.3.3. Λειτουργία του εξυπηρετητή (server)*

Ο εξυπηρετητής είναι το τμήμα το οποίο είναι συνεχώς ενεργό, και αναλαμβάνει να εξυπηρετεί τις αιτήσεις (που περιέχουν επερωτήσεις) των χρηστών του συστήματος. Στην παρακάτω παράγραφο (8.4) θα εξετάσουμε πώς γίνεται η επικοινωνία με αυτό το τμήμα του συστήματος, ενώ στις παραγράφους 8.5 και 8.6 θα δούμε αναλυτικά τους τύπους των επερωτήσεων που υποστηρίζονται.

Σε αυτήν την παράγραφο θα εξετάσουμε γενικά την λειτουργία του εξυπηρετητή. Όταν ξεκινά η διεργασία του εξυπηρετητή (server), διαβάζεται η λίστα των εκδοτών που συμμετέχουν στην συλλογή, και έπειτα τα αρχεία ευρετηριασμού. Αφού ολοκληρωθεί αυτή η διαδικασία ανάγνωσης όλων των απαραίτητων δεδομένων από τον δίσκο, τότε η διεργασία κάνει ένα αντίγραφο του εαυτού της. Συνεπώς έχουμε διαθέσιμους δύο εξυπηρετητές. Αυτό μπορεί να αλλάξει από τον υπεύθυνο του συστήματος, ανάλογα με το σύστημα στο οποίο λειτουργεί ο εξυπηρετητής. Αν για παράδειγμα, ο σταθμός εργασίας έχει λίγη διαθέσιμη μνήμη, τότε θα θέλαμε μόνο μια διεργασία. Χρησιμοποιώντας όμως, πάνω από μια διεργασίες έχουμε τα εξής πλεονεκτήματα: πρώτον δεν βαρύνουμε πολύ την κάθε διεργασία με πολλές ίνες (threads) και δεύτερον, με τον τρόπο βελτιώνεται η ανοχή του συστήματος σε βλάβες (fault tolerance)[29].

Όλες οι διεργασίες δέχονται αιτήσεις στο ίδιο "port" του σταθμού εργασίας. Όταν ο χρήστης στέλνει μια αίτηση στον εξυπηρετητή, η επιλογή της διεργασίας η οποία θα δεχτεί και θα εξυπηρετήσει την αίτηση, γίνεται από τον λειτουργικό σύστημα με τυχαίο τρόπο, αλλά συνήθως είναι αυτή που είναι λιγότερο απασχολημένη.

Όταν μια διεργασία δεχτεί μια αίτηση, τότε δημιουργεί μια ίνα (thread), η οποία θα αναλάβει την εξυπηρέτησή της. Το μέγιστο πλήθος των ενεργών ινών μιας διεργασίας, έχει οριστεί στις τριάντα, αλλά προφανώς είναι πολύ εύκολο για τον υπεύθυνο του συστήματος να αλλάξει το πλήθος αυτό.

Επίσης, ο εξυπηρετητής, όταν διαβάζει δεδομένα (συγγραφέα, τίτλο, ημερομηνία κ.τ.λ.) για ένα αντικείμενο από την βάση δεδομένων, κρατά μερικά από αυτά από αυτά τα δεδομένα σε μια εσωτερική cache. Το ποια πεδία είναι αυτά που θα παραμείνουν στην cache, ορίζεται από την μεταβλητή «MEMORY_RESIDENT_FIELDS», που είναι όρισμα στο server. Ανά τακτά χρονικά διαστήματα, δημιουργείται μια ίνα (thread), η οποία καθαρίζει την cache. Ο αλγόριθμος που χρησιμοποιήθηκε για αυτήν την εργασία είναι ο FIFO. Δηλαδή τα κείμενα που γράφτηκαν πρώτα στην cache, θα διαγραφούν πρώτα.

[29] Αν κάποια από τις διεργασίες έχει πρόβλημα λειτουργίας, τότε το σύστημα συνεχίζει να λειτουργεί κανονικά, δεδομένου ότι η άλλη διεργασία (ή οι άλλες διεργασίες) εξακολουθεί να δέχεται αιτήσεις από τους χρήστες (εφόσον οι διεργασίες στο UNIX είναι ανεξάρτητες).

## 8.4. Πρωτόκολλο επικοινωνίας με τον server

Σε αυτό το κεφάλαιο, θα δούμε το πρωτόκολλο επικοινωνίας με τον εξυπηρετητή. Τα υπόλοιπα τμήμα δεν έχουν κάποιο πρωτόκολλο επικοινωνίας, με την έννοια ότι δεν εξυπηρετούν αιτήσεις, αλλά κάθε φορά χρησιμοποιούν το πρωτόκολλο επικοινωνίας του εξυπηρετητή με τον οποίο επικοινωνούν.

Η επικοινωνία με τον εξυπηρετητή, βασίζεται στο πρωτόκολλο HTTP/1.0. Ο εξυπηρετητής είναι ικανός να δέχεται απευθείας αιτήσεις από browsers, χωρίς την διαμεσολάβηση ενός HTTP server. Αυτό έγινε ώστε να αποφύγουμε τον επιπλέον φόρτο επικοινωνίας με έναν http server . Στην δεύτερη περίπτωση για την εξυπηρέτηση μιας αίτησης θα γίνονταν τα παρακάτω βήματα:

1. Επικοινωνία του browser με τον http server
2. εκκίνηση ενός cgi-script
3. μεταφορά της αίτησης στον εξυπηρετητή
4. εξυπηρέτηση της αίτησης
5. μεταφορά των δεδομένων από τον εξυπηρετητή στο cgi-script
6. μεταφορά των δεδομένων από το cgi-script στον http server
7. μεταφορά των δεδομένων από τον http server στον browser του χρήστη.

Αντιθέτως, στην περίπτωση που έχει υλοποιηθεί, ακολουθούνται τα παρακάτω βήματα:

1. Επικοινωνία του browser με τον εξυπηρετητή
2. εξυπηρέτηση της αίτησης
3. μεταφορά των δεδομένων από τον εξυπηρετητή στον browser του χρήστη.

Θα περιγράψουμε παρακάτω, πώς οι αιτήσεις στον εξυπηρετητή, κωδικοποιούνται κάτω από το πρωτόκολλο HTTP, χρησιμοποιώντας URLs.

## 8.5. Τύποι Αιτήσεων προς τον server

Η μορφή του URL των αιτήσεων, έχει οριστεί ως εξής:

| *http://host:port/Spider/{protocol_version}/{Html\|Text}/{Command}?Arguments* |
|---|

Πίνακας 11. Μορφή URL αίτησης προς τον spider server.

Το {protocol_version} αυτήν την στιγμή είναι 1.0. Το δεύτερο τμήμα, είναι είτε Html είτε Text, και δηλώνει την μορφή της απάντησης του server. Η μορφή text, μπορεί να χρησιμοποιηθεί για αποστολή αιτήσεων προς τον server, είτε από άλλους dienst spiders, είτε από άλλες μηχανές αναζήτησης που κάνουν αιτήσεις προς τον dienst spider, ώστε να είναι ευκολότερη η ερμηνεία της απάντησης. Η εντολή που περιέχεται στην αίτηση αντιστοιχεί στο {command}, και ακολουθούν τα ορίσματα στην εντολή αυτή. Παρακάτω θα δούμε τις εντολές που υποστηρίζονται, καθώς και τα δυνατά ορίσματα για κάθε μια από αυτές.

### *8.5.1. Info*

Η εντολή "Info" δίνει πληροφορίες για την κατάσταση του server. Ανάμεσα στις πληροφορίες περιλαμβάνονται το πλήθος των αντικειμένων που περιέχονται στην βάση, το μέγεθος του αρχείου ευρετηριασμού, κτλ..

### *8.5.2. Search*

Η εντολή «search» είναι η βασική εντολή προς τον spider server. Τα ορίσματα σε αυτήν την εντολή είναι τα ορίσματα που είναι κοινά και με τις υπόλοιπες εντολές, που έχουν να κάνουν με τα περιεχόμενα της σελίδας αποτελεσμάτων, οι λέξεις κλειδιά, καθώς και ο τύπος της αναζήτησης. Οι τύποι αναζήτησης περιγράφονται παρακάτω:

- Exact keyword: Εδώ η αναζήτηση γίνεται με βάση το ακριβές ταίριασμα των λέξεων κλειδιών.

- Substring keyword: αν η αναζήτηση είναι αυτού του τύπου, τότε για να πετύχει το ταίριασμα, αρκεί η λέξη κλειδί να περιέχεται σαν υπο-λέξη σε κάποια λέξη στο αρχείο ευρετηριασμού. Π.χ. η λέξη κλειδί "net" θα ταιριάσει με τις λέξεις "net", "network", κ.τ.λ..

- Stemmed keyword: Εδώ η αναζήτηση γίνεται με βάση την ρίζα της λέξης. Π.χ. η λέξη κλειδί "indexing" θα ταιριάσει με τις λέξεις "indexer", "indexing", "indexed", "indices" κ.τ.λ..

- Category: Σε αυτήν την περίπτωση, θεωρείται ότι η επερώτηση περιέχει μια θεματική κατηγορία. Σε αυτήν την περίπτωση η αναζήτηση γίνεται με βάση τις ρίζες των λέξεων, και επιπλέον υπολογίζεται και κάποιο σκορ. Περισσότερα για τον τρόπο υπολογισμού του, αναφέρονται στην παράγραφο 8.7.

### *8.5.3. Search like document*

Με αυτήν την εντολή, ο χρήστης μπορεί να ψάξει για κείμενα που μοιάζουν με κάποιο άλλο, του οποίου το αναγνωριστικό δίνεται σαν όρισμα. Ο χρήστης μπορεί να δώσει αυτήν την εντολή, ακολουθώντας το link: "More Like This", που υπάρχει στην σελίδα εμφάνισης αποτελεσμάτων αναζήτησης δίπλα από κάθε κείμενο.

Επειδή αυτού του τύπου η αναζήτηση είναι χρονοβόρα, (πρέπει να συγκριθεί το κείμενο που δίνεται ως όρισμα με όλα τα υπόλοιπα κείμενα και να υπολογιστεί το ποσοστό ομοιότητας), επιλεκτικά, ο υπεύθυνος του συστήματος μπορεί να ορίσει να προϋπολογίζονται τα ποσοστά ομοιότητας κατά την διαδικασία της δημιουργίας των αρχείων ευρετηριασμού, και να αποθηκεύονται σε αρχείο. Ακόμη, επειδή τα «παρόμοια» κείμενα μπορεί να είναι πάρα πολλά, στο αρχείο αποθηκεύονται μόνο τα 200 «παρόμοια» κείμενα με το μεγαλύτερο ποσοστό ομοιότητας.

### *8.5.4. Search page (start page)*

Αν και η αρχική σελίδα, είναι στην πραγματικότητα στατική, και αυτή δημιουργείται αυτόματα με την κενή αίτηση προς τον server ή την αίτηση για το αρχικό αρχείο (http://sappho.csi.forth.gr:22000/) . Αυτό για την μελλοντική περίπτωση που θα θέλαμε η σελίδα αυτή να είναι δυναμική (με βάση τα περιεχόμενα της βάσης).

## 8.6. Ορίσματα αιτήσεων αναζήτησης προς τον server

Σε όλες τις αιτήσεις αναζήτησης, υπάρχουν κάποια κοινά ορίσματα. Τα ορίσματα με τα οποία ο χρήστης ορίζει πώς τι θα περιέχεται στην σελίδα των αποτελεσμάτων, και τα ορίσματα για τον τύπο της αναζήτησης.

### *8.6.1. Ορίσματα για εμφάνιση αποτελεσμάτων*

Αυτά τα ορίσματα είναι τα παρακάτω:

- *Number*: Το πλήθος των επιτυχιών που θα εμφανιστούν.
- *Publisher*: Ο εκδότης του κειμένου (π.χ. ΙΤΕ/ΙΠ).
- *Author*: Οι συγγραφείς του κειμένου.
- *Date*: Η ημερομηνία έκδοσης του κειμένου.
- *Morelike*: Ένας σύνδεσμος (link) για τα κείμενα που μοιάζουν με το παρών κείμενο.
- *Stats*: Στατιστικά, όπως το πλήθος εμφάνισης της λέξης μέσα στο κείμενο, ή το σκορ ομοιότητας.

Η τιμή του "number" είναι ένας αριθμός, ο οποίος δηλώνει το πλήθος των επιτυχιών που θα εμφανιστούν στην σελίδα αποτελεσμάτων. Όλες οι υπόλοιπες τιμές μπορεί να είναι "yes" ή "no". Αν παραληφτεί κάποιο, θεωρείται ότι είναι "no". Το "yes" δηλώνει ότι ο χρήστης θέλει να εμφανίζεται το αντίστοιχο πεδίο στην σελίδα αποτελεσμάτων, και σε αντίθετη περίπτωση δεν εμφανίζεται.

Ένα παράδειγμα αίτησης είναι το παρακάτω:

*http://sappho.csi.forth.gr:22000/Spider/0.1/Html/Search?keywords=spiders&Search_Type=category&number=10&mtitle=yes&mauthor=yes&mabstract=yes&mother=yes&publisher=yes&author=yes&date=yes&morelike=yes&stats=yes*

*http://sappho.csi.forth.gr:22000/Spider/0.1/Text/Search?keywords=spiders&Search_Type=category&number=10&mtitle=yes&mauthor=yes&mabstract=yes&mother=yes&publisher=yes&author=yes&date=yes&morelike=yes&stats=yes*

Πίνακας 12. Παραδείγματα αίτησης αναζήτησης προς τον spider server.

Σε αυτό το παράδειγμα, οι δύο αιτήσεις είναι απολύτως ισοδύναμες, μόνο που η πρώτη δημιουργεί έξοδο σε html μορφή, ενώ η δεύτερη σε απλή μορφή κειμένου.

### *8.6.2. Ορίσματα για την αναζήτηση*

Τα ορίσματα για την αναζήτηση είναι τα παρακάτω:

- *Keywords:* Οι λέξεις κλειδιά για την αναζήτηση
- *Search_type:* Ο τύπος της αναζήτησης (exact string, sub-string, stem match, category)
- *M(atch)title:* Οι τιμές είναι ναι ή όχι (yes/no), αντίστοιχα αν η αναζήτηση θα γίνει και στο πεδίο «τίτλος» ή όχι.
- *M(atch)author:* Ομοίως αν η αναζήτηση θα γίνει και στο πεδίο «συγγραφέας» ή όχι.
- *M(atch)abstract:* Ομοίως αν η αναζήτηση θα γίνει και στο πεδίο «περίληψη» ή όχι.
- *M(atch)other:* Ομοίως αν η αναζήτηση θα γίνει και στα υπόλοιπα πεδία ή όχι.

## 8.7. Αναζήτηση κατηγορίας

Η αναζήτηση με βάση λέξεις κλειδιά, είναι τετριμμένη. Αντιθέτως, ιδιαίτερο ενδιαφέρον παρουσιάζει η αναζήτηση με βάση θεματική κατηγορία. Παρακάτω θα εξηγήσουμε πως λειτουργεί ο αλγόριθμος αναζήτησης σε αυτήν την περίπτωση.

Έστω, ότι ο χρήστης δίνει k λέξεις κλειδιά που δηλώνουν μια θεματική περιοχή, τα $key_1,\ldots, key_k$. Για κάθε ένα από αυτά τα κλειδιά, βρίσκουμε τα κείμενα στα οποία ανήκει. Έστω $doc\_set_1,\ldots,doc\_set_k$ είναι τα σύνολα των κειμένων που περιέχουν τις αντίστοιχες λέξεις κλειδιά. Αρχικά δίνουμε από κάποιο βάρος σε κάθε λέξη κλειδί. Αυτό το κάνουμε για να εντοπίσουμε τις «σημαντικές» λέξεις κλειδιά από λέξεις οι οποίες χρησιμοποιούνται συχνά. Έτσι το βάρος κάθε λέξης προκύπτει από τον τύπο:

$$term\_weight_i = -\log\left(\frac{number\_of\_documents\_that\_this\_term\_belongs}{total\_number\_of\_documents}\right)$$ [30]

Με αυτό τον τρόπο, στις λέξεις που χρησιμοποιούνται συχνά, δίνουμε πολύ μικρό βάρος.

Για κάθε κείμενο j που ανήκει στο σύνολο $doc\_set_i$ , υπολογίζουμε το σκορ:

$$score_{ij} = \left(\frac{a\_f * a\_c_{ij} + t\_f * t\_c_{ij} + au\_f * au\_c_{ij} + o\_f * o\_c_{ij}}{total\_count_j}\right) * term\_weight_i$$

$$total\_count_j = a\_f * a\_w_j + t\_f * t\_w_j + au\_f * au\_w_j + o\_f * o\_w_j$$

Αν το κείμενο j, δεν ανήκει στο σύνολο $doc\_set_i$ , τότε θέτουμε $score_{ij}=0$.

Οι μεταβλητές *a_f (abstract factor), t_f (title factor), au_f (author factor), o_f (other factor)* είναι οι συντελεστές βαρύτητας των πεδίων περίληψης (abstract), τίτλου (title), συγγραφέα (author), όλα τα υπόλοιπα (other). Αυτό διότι είναι προφανές, ότι έχει μεγαλύτερο συντελεστή βάρους μια λέξη στον τίτλο ενός κειμένου από μια άλλη στην περίληψη ή σε κάποιο άλλο πεδίο. Οι παραπάνω συντελεστές βαρύτητας έχουν οριστεί ως εξής: ABSTRACT_FACTOR=1, TITLE_FACTOR=4, AUTHOR_FACTOR=2, OTHER_FACTOR=0. Για όλα τα υπόλοιπα πεδία (other factor), έχουμε δώσει τον συντελεστή 0, διότι στην περίπτωση των τεχνικών αναφορών, όλη η ζητούμενη πληροφορία περιέχεται στα πεδία: τίτλος, συγγραφείς και περίληψη.

Οι μεταβλητές $a\_c_{ij}$ *(abstract count),* $t\_c_{ij}$ *(title count),* $au\_c_{ij}$ *(author count),* $o\_c_{ij}$ *(other_count)* είναι το πλήθος των φορών που υπάρχει η λέξη κλειδί i στο κείμενο j στο αντίστοιχο πεδίο.

Οι μεταβλητές $a\_w_j$ *(abstract words),* $t\_w_j$ *(title words),* $au\_w_j$ *(author words),* $o\_w_j$ *(other words)* είναι το πλήθος των λέξεων που περιέχει το αντίστοιχο πεδίο του κείμενο j.

[30] Υπάρχουν πολλές άλλες δυνατές επιλογές όσον αφορά τον υπολογισμό του βάρους. Βλέπε παράρτημα VIII.

Το συνολικό σκορ ενός κειμένου j υπολογίζεται ως:

$$score_j = 1 - \prod_{i=1}^{k} (1 - score_{ij})$$

Τον οποίο μπορούμε να τον υλοποιήσουμε με αναδρομικό τρόπο ανά δύο, ως εξής:

$$score_{(1..i)j} = 1 - (1 - score_{ij}) * (1 - score_{(1..i-1)j})$$

Το $score_{(1..i)j}$ , είναι το σκορ του κειμένου j, για τις λέξεις κλειδιά 1 έως i. Όταν το i γίνει ίσο με το k, τότε έχουμε υπολογίσει το $score_j$ .

Εάν, το τελικό $score_j$, το διαιρέσουμε με το άθροισμα $\sum_{i=1}^{k} term\_weight_i$ , τότε παίρνουμε την πιθανότητα το κείμενο j να ανήκει την ζητούμενη κατηγορία.

## 8.8. Γραμματική εισαγωγής επερωτήσεων

Η γραμματική εισαγωγής των επερωτήσεων από τον χρήστη, είναι αρκετά απλή, και αντίστοιχη με τις γραμματικές που χρησιμοποιούν και άλλες γνωστές μηχανές αναζήτησης (όπως η altavista), ώστε ο χρήστης να χρησιμοποιεί ήδη γνωστό τρόπο εισαγωγής επερωτήσεων.

Έτσι η γραμματική είναι η εξής:

```
query: subquery
subquery: subquery | subquery searchterm
searchterm: operator keyword
operator: + | - | e
```

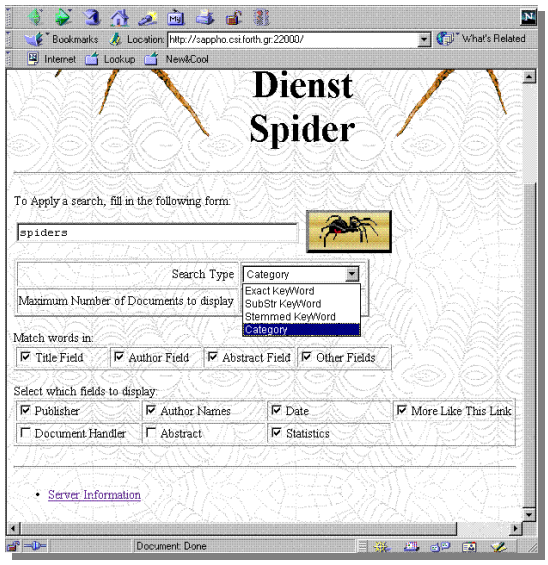
Η φιλοσοφία είναι η εξής: αν ο χρήστης δεν βάλει κανέναν "operator" ενδιάμεσα στις λέξεις κλειδιά, τότε εννοείται η λογική πράξη «ή» ("or") μεταξύ των λέξεων κλειδιών. Άρα για να ικανοποιεί ένα κείμενο την επερώτηση πρέπει να περιέχει κάποια από τις λέξεις κλειδιά. Με το σύμβολο «+» δηλώνεται το «και» ("and"), και όταν προηγείται από κάποια λέξη κλειδί σημαίνει ότι ο χρήστης θέλει να περιέχεται οπωσδήποτε η λέξη κλειδί που ακολουθεί μέσα στο κείμενο. Αντίστοιχα το σύμβολο «-» δηλώνει το «όχι» ("not"), σημαίνει ότι η λέξη κλειδί που ακολουθεί δεν πρέπει να περιέχεται στο κείμενο.

# Κεφάλαιο 9

## 9. Χρήση – Διεπιφάνεια Χρήσης

### 9.1. Αρχική φόρμα αναζήτησης

Στην αρχική σελίδα του συστήματος, εμφανίζεται η φόρμα αναζήτησης (Σχήμα 22). Εδώ ο χρήστης, εκτός από την εισαγωγή της επερώτησης (query), μπορεί να ορίσει 3 κατηγορίες παραμέτρων που αφορούν την αναζήτηση και την εμφάνιση των αποτελεσμάτων:




Σχήμα 22. Φόρμα αναζήτησης του spider server.

- Τον τύπο της επερώτησης (exact, sub-string, stemmed keyword, category), καθώς και το πλήθος των επιτυχιών που θα περιέχει η σελίδα αποτελεσμάτων.

- Τα πεδία τα οποία θέλει να περιέχουν τις λέξεις κλειδιά (μόνο στις περιπτώσεις του keyword search).

- Και τέλος, τα πεδία των αντικειμένων που ικανοποιούν την επερώτηση, τα οποία θέλει να εμφανίζονται στην σελίδα των αποτελεσμάτων. Έτσι μπορεί ο χρήστης να επιλέξει να βλέπει μόνο τους τίτλους των κειμένων, και να έχει μια πολύ συμπυκνωμένη εμφάνιση (compact layout).

## 9.2. Σελίδα πληροφοριών για τον server

Εδώ, εμφανίζονται κάποιες πληροφορίες για την διαμόρφωση (configuration) του εξυπηρετητή (Σχήμα 23). Οι πληροφορίες που εμφανίζονται είναι κάποιες βασικές πληροφορίες για το μέγεθος της βάσης δεδομένων και των αρχείων ευρετηριασμού που έχει δημιουργήσει, καθώς επίσης και ποιος είναι ο "master meta server" της συλλογής από την οποία παίρνει δεδομένα ο spider. Στην περίπτωση αυτής της εφαρμογής του spider, η συλλογή είναι η ncstrl-ercim, οπότε και ο master meta server είναι αυτός του Cornell.

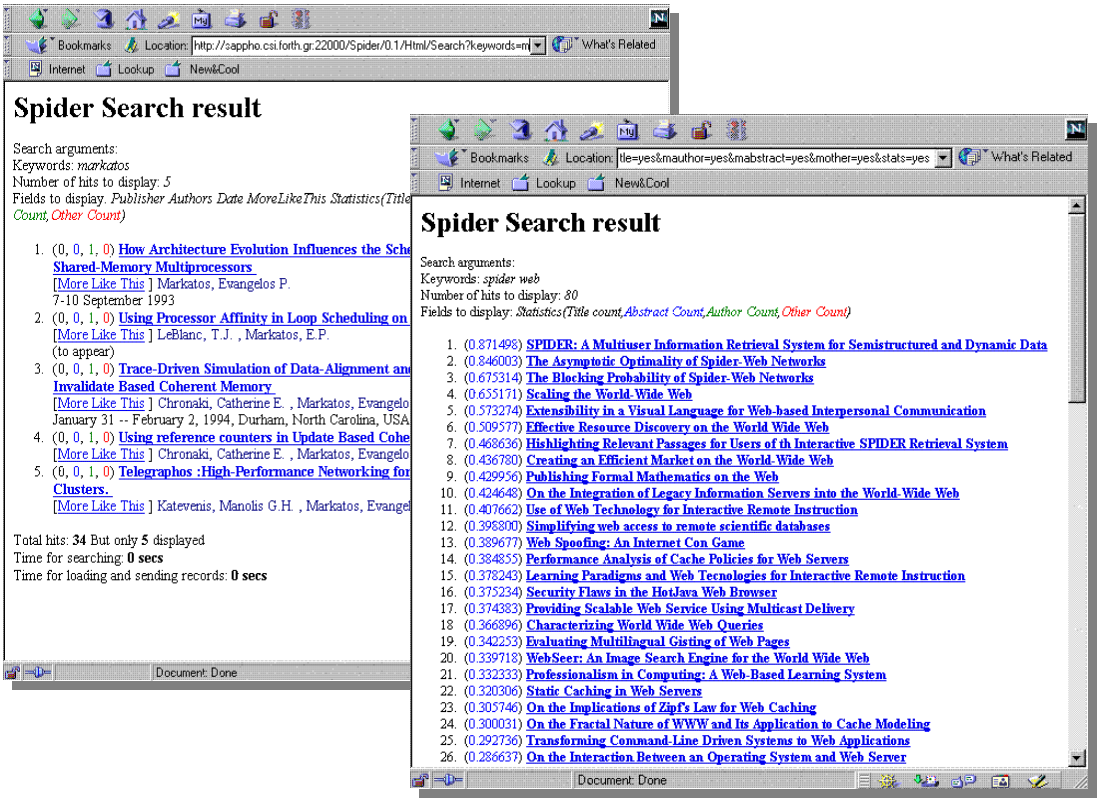

Σχήμα 23. Σελίδα πληροφοριών για τον server

## 9.3. Παρουσίαση αποτελεσμάτων αναζήτησης

Στις σελίδες παρουσίασης των αποτελεσμάτων αναζήτησης, παρουσιάζεται μια απλή λίστα με τα κείμενα που ικανοποιούν τις συνθήκες της αναζήτησης. Τα ακριβή περιεχόμενα του κάθε στοιχείου της λίστας (όπως περιγράφεται και στην παράγραφο 8.6.1) τα ορίζει ο ίδιος ο χρήστης. Σίγουρα περιλαμβάνεται ο σύνδεσμος (link) προς την τοποθεσία από την οποία ο χρήστης μπορεί να πάρει το κείμενο. Εναλλακτικά ο χρήστης μπορεί να επιλέξει να βλέπει την ημερομηνία δημοσίευσης του κειμένου, τους συγγραφείς ή το αναγνωριστικό του κειμένου (docid). Στο παρακάτω σχήμα (Σχήμα 24) παρουσιάζεται μια τυπική σελίδα αποτελεσμάτων και μια σελίδα με αποτελέσματα σε συμπιεσμένη μορφή.

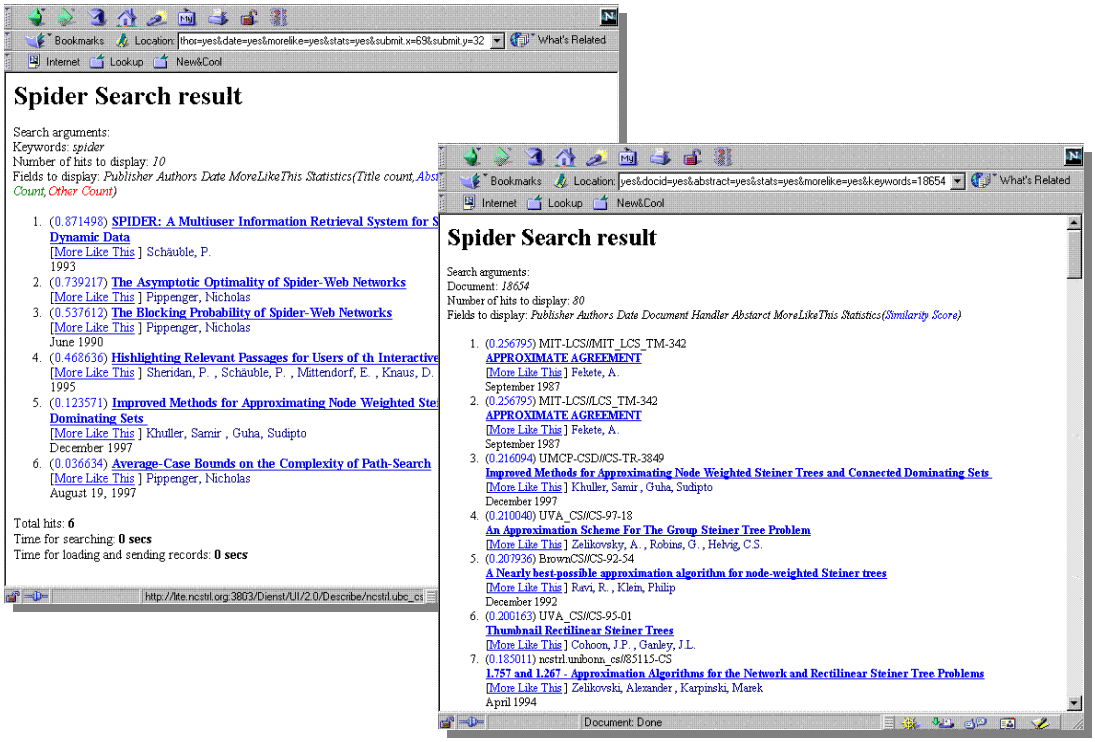
Επίσης, αν έχει ενεργοποιηθεί η αντίστοιχη επιλογή, για κάθε κείμενο, εμφανίζεται ένας σύνδεσμος «more like this», ο οποίος οδηγεί σε αναζήτηση παρόμοιων κειμένων με αυτό στο οποίο αντιστοιχεί ο σύνδεσμος (Σχήμα 25β).

Τέλος, στην αρχή του κάθε πεδίου της λίστας, εμφανίζονται τα στατιστικά αποτελέσματα της αναζήτησης. Αν η αναζήτηση είναι απλό ταίριασμα λέξεων κλειδιών, τότε εμφανίζονται 4 νούμερα τα οποία είναι πόσες φορές περιέχεται η λέξη κλειδί στα πεδία τίτλο, περίληψη, συγγραφείς και στα υπόλοιπα πεδία. Αν η αναζήτηση είναι αναζήτηση κατηγορίας, τότε εμφανίζεται ένα νούμερο, το οποίο είναι το σκορ που δίνει το κάθε κείμενο για την συγκεκριμένη κατηγορία. Τέλος, αν η αναζήτηση είναι αναζήτηση ομοιότητας με κάποιο άλλο κείμενο, τότε εμφανίζεται ο συντελεστής ομοιότητας με το συγκεκριμένο κείμενο.




Σχήμα 24. Αποτελέσματα αναζήτησης (α) Κανονική εμφάνιση (β) Εμφάνιση σε συμπιεσμένη μορφή

Στο κάτω τμήμα κάθε σελίδας, εμφανίζονται κάποια απλά στοιχεία για την αναζήτηση τα οποία είναι πόσα συνολικά κείμενα ικανοποιούν την αναζήτηση και πόσα τελικά επιστράφηκαν, καθώς και τον χρόνο που έκανε ο εξυπηρετητής να εκτελέσει την αναζήτηση και τον χρόνο που του πήρε για να στείλει τα αποτελέσματα.




Σχήμα 25. Παρουσίαση αποτελεσμάτων αναζήτησης (α) αναζήτηση κατηγορίας (β) αναζήτηση παρόμοιων κειμένων

# *Κεφάλαιο 10*

# 10. Θέματα Υλοποίησης

## 10.1. Σχεδίαση και περιεχόμενα της Βάσης Δεδομένων

### *10.1.1. Περιεχόμενα της βάσης*

Τα περιεχόμενα της βάσης είναι βιβλιογραφικές εγγραφές των κειμένων. Δηλαδή τα ίδια δεδομένα που χρησιμοποιεί και το dienst για να κάνει αναζήτηση. Το dienst αποθηκεύει και μεταφέρει, τις βιβλιογραφικές εγγραφές, χρησιμοποιώντας το στάνταρ RFC-1357. Βλέπε παράγραφο 3.2.3.

Για την αποθήκευση των βιβλιογραφικών εγγραφών, μπορούν να χρησιμοποιηθούν διάφοροι εναλλακτικοί τρόποι, ο καθένας από αυτούς έχει πλεονεκτήματα και μειονεκτήματα.

- Μια επιλογή, είναι η χρησιμοποίηση μιας σχεσιακής (ή οντοκεντρικής) βάσης δεδομένων. Σε αυτήν την περίπτωση μπορούμε να κάνουμε μια σειρά από επιλογές, που θα έχουν αντίχτυπο στην απόδοση και στην διαχείριση χώρου στο δίσκο.

    - Η μια επιλογή είναι να "σπάσουμε" μια εγγραφή σε πολλές πλειάδες. Κάθε πλειάδα θα περιέχει ένα πεδίο της εγγραφής, το ID της εγγραφής, και το ID του τίτλου του πεδίου.
      Σε αυτήν την περίπτωση έχουμε δραματική αύξηση του χρόνου πρόσβασης σε μια εγγραφή ή στα πεδία της διότι για την ανάγνωση μιας εγγραφής, χρειάζεται να διαβάζονται πολλές πλειάδες από την βάση, οι οποίες – εφόσον δεν μπορούμε να έχουμε τον έλεγχο της λειτουργίας της βάσης – μπορεί να είναι αποθηκευμένες σε μακρινά σημεία του δίσκου. Έχουμε όμως το πλεονέκτημα ότι μπορούμε να διαβάσουμε οποιοδήποτε πεδίο μιας εγγραφής, χωρίς να μας απασχολούν τα υπόλοιπα πεδία.

    - Η άλλη επιλογή είναι να υπάρχει μια πλειάδα για κάθε εγγραφή, η οποία θα περιέχει σε ένα πεδίο το ID της βιβλιογραφικής εγγραφής, και στο δεύτερο (που θα είναι τύπου κειμένου μεταβλητού μεγέθους) όλο το κείμενο (text) της εγγραφής, όπως αυτό ορίζεται από το RFC-1357. Σε αυτήν την περίπτωση, πρόσβαση σε ένα πεδίο της εγγραφής, απαιτεί "parsing" όλης της εγγραφής (όπως και αν χρησιμοποιούσαμε αρχεία). Αυτή μπορούμε να πούμε ότι είναι η χειρότερη

επιλογή για χρήση μιας βάσης δεδομένων, διότι απορρίπτουμε όλα τα πλεονεκτήματα που μπορεί να μας προσφέρει αυτή, αλλά επιπλέον έχουμε το κόστος χρήσης μιας βάσης δεδομένων[31].

- Ή, τέλος, μια σχεσιακή βάση σχεδιασμένη για τα δεδομένα που έχουμε. Τότε όμως έχουμε τους περιορισμούς του σχήματος της βάσης για ότι αφορά τα περιεχόμενα των δεδομένων. Για παράδειγμα, αν το σχήμα δεν περιλάμβανε το πεδίο "references", τότε αν βρισκόταν ένα αντικείμενο με τέτοιο πεδίο, θα χάναμε την πληροφορία που περιλαμβάνει. Αν η μορφή των meta data των δεδομένων, είναι αυστηρώς προκαθορισμένη, τότε θα μας συνέφερε να υλοποιήσουμε αυτή τη μέθοδο, στην περίπτωσή μας όμως αυτό δεν ισχύει.

Χρησιμοποιώντας μια βάση δεδομένων έχουμε το κέρδος της επιπλέον λειτουργικότητας και της ευκολίας υλοποίησης. Επίσης μεταφέρουμε το όλο θέμα της απόδοσης στην ανάγνωση και αναζήτηση στον μηχανισμό της βάσης, στον οποίο όμως δεν μπορούμε να έχουμε καμία ουσιαστική παρέμβαση για να κάνουμε οποιαδήποτε βελτίωση. Τέλος, έχουμε το επιπλέον κόστος επικοινωνίας με τον εξυπηρετητή της βάσης (data base server).

- Μια άλλη επιλογή είναι η χρήση απευθείας του συστήματος αρχείων (file system). Και σε αυτήν την περίπτωση μπορούμε να ακολουθήσουμε διάφορους τρόπους υλοποίησης:

  - Ο πρώτος, ο οποίος είναι αρκετά οικονομικός από άποψη χώρου στον δίσκο, είναι η αποθήκευση όλων των εγγραφών σε ένα αρχείο, ή όλων των εγγραφών ενός εκδότη σε ένα αρχείο. Έτσι εξοικονομείται χώρος, αλλά η αλλαγή των περιεχομένων είναι δαπανηρή. Όμως, η ανάγνωση των δεδομένων είναι γρήγορη, δεδομένου ότι το αρχείο παραμένει ανοιχτό, και κάθε φορά που πρέπει να διαβαστεί μια εγγραφή, μεταφέρεται ο δείκτης στο αρχείο στο αντίστοιχο σημείο που είναι αποθηκευμένη η εγγραφή (με seek). Για να είναι αυτό δυνατό πρέπει να αποθηκεύονται (στο ίδιο αρχείο ή σε άλλο), οι θέσεις μέσα στο αρχείο στις οποίες είναι αποθηκευμένη η κάθε εγγραφή. Αυτό, αυξάνει την πολυπλοκότητα του συστήματος ακόμη και για την ανάγνωση δεδομένων. Επίσης ακόμη περισσότερο αυξάνεται η πολυπλοκότητα του συστήματος όταν η διεργασία που κάνει πρόσβαση στο αρχείο αποτελείται από πολλές ίνες (multithreaded), διότι η διεργασία πρέπει να διατηρεί τόσους δείκτες στο αρχείο αυτό, όσες είναι και οι ίνες που κάνουν ανάγνωση δεδομένων[32]. Οπότε, τελικώς πρέπει να

[31] Το overhead της επικοινωνίας με την βάση δεδομένων, και το overhead που προσθέτει η ίδια η βάση λόγω υποστήριξης δοσοληψιών (transactions).

[32] Από την άλλη, αν διατηρούμε μόνο έναν δείκτη στο κάθε αρχείο, δεν μπορούμε να έχουμε παραλληλισμό στην ανάγνωσή του.

υλοποιηθεί κατά κάποιον τρόπο ένα εσωτερικό σύστημα αρχείων της διεργασίας που ουσιαστικά θα μετατρέπει το απλό αρχείο σε ένα εικονικό σύστημα αρχείων (για την διεργασία εσωτερικά)[33].

- Ο δεύτερος τρόπος, αυτός που χρησιμοποιείται και από το dienst, είναι για κάθε εγγραφή ένα ξεχωριστό αρχείο με όνομα που προκύπτει από το "ID" της. Σε αυτήν την περίπτωση, η ανάγνωση είναι λίγο περισσότερο αργή απ' ότι στην προηγούμενη, διότι κάθε φορά που ζητείται μια εγγραφή πρέπει να αναζητηθεί το αρχείο, δηλαδή να ελεγχθεί από το λειτουργικό, το σύστημα αρχείων (file system) για να βρεθεί η θέση του αρχείου στον δίσκο, να κληθούν τουλάχιστον 2 system calls (open και close στο τέλος) και να δημιουργηθεί ο αντίστοιχος δείκτης στο αρχείο (file handle). Από την άλλη όμως μπορούμε να έχουμε πλήρη παραλληλισμό εσωτερικά μιας διεργασίας (στις ίνες), χωρίς να χρειάζεται να κάνουμε τίποτα επιπλέον, διότι έτσι κι αλλιώς για την ανάγνωση μιας νέας εγγραφής είναι απαραίτητη η εκ νέου δημιουργία του δείκτη στο αντίστοιχο αρχείο (file handle).

Και στις 2 πάνω περιπτώσεις τον ευρετηριασμό τον αναλαμβάνει ένας μηχανισμός, ο οποίος δημιουργεί αντίστροφες λίστες με τις λέξεις κλειδιά[34].

Έτσι σε σχέση με την χρήση μιας βάσης δεδομένων, χρησιμοποιώντας ένα απλό σύστημα αρχείων, έχουμε πλήρη έλεγχο του συστήματος αποθήκευσης των δεδομένων, και επιπλέον δεν χρειαζόμαστε εσωτερική επικοινωνία με κάποιο άλλο σύστημα.

Δεδομένου λοιπόν ότι δεν χρειαζόμαστε (για το συγκεκριμένο σύστημα) τις επιπλέον λειτουργίες μιας βάσης δεδομένων, είναι προτιμότερη η λύση της χρήσης ενός απλού συστήματος αρχείων. Από τις επιλογές, όσον αφορά τον τρόπο αποθήκευσης με την χρήση απευθείας του συστήματος αρχείων, επιλέξαμε την δεύτερη επιλογή (χρήση ενός αρχείου για κάθε εγγραφή), αν και ίσως πιο αργό απ' ότι η πρώτη επιλογή, διότι είναι περισσότερο κατάλληλη για την συγκεκριμένη περίπτωση, και επιπλέον χρησιμοποιούμε την τεχνογνωσία που αποκτήθηκε κατά την εργασία μας πάνω στο σύστημα dienst. Επίσης, δεδομένου ότι μπορούμε να έχουμε πλήρη έλεγχο του συστήματος, υλοποιήσαμε μια εσωτερική cache, η οποία κρατά από τις εγγραφές, τα πεδία που χρησιμοποιούνται περισσότερο συχνά (Βλέπε 8.3.3, 10.2).

---

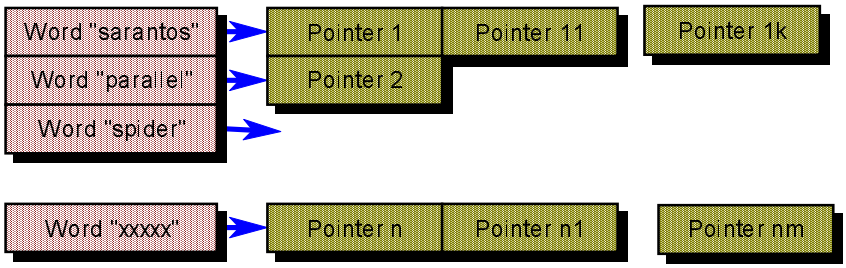
[33] Το οποίο δεν αποτελεί στόχο αυτής της εργασίας και επιπλέον μια απλή υλοποίησή του θα μπορούσε να κάνει το σύστημα πολύ πιο αργό, σε σχέση με την δεύτερη επιλογή χρήσης απλού αρχείου ανά εγγραφή.

[34] Το οποίο μπορεί να υλοποιηθεί και χρησιμοποιώντας μια βάση δεδομένων.

### *10.1.2. Ευρετηριασμός βάσης - αντίστροφη λίστα*

Για τον ευρετηριασμό των εγγράφων που περιέχονται στην βάση, χρησιμοποιείται αντίστροφη λίστα (Inverted list).

Μια απλή μορφή αντίστροφης λίστας, φαίνεται στο Σχήμα 26. Η αντίστροφη λίστα περιέχει μια λίστα από λέξεις κλειδιά, και για κάθε κλειδί ακολουθεί μια σειρά από δείκτες στα κείμενα που περιέχουν την αντίστοιχη λέξη.

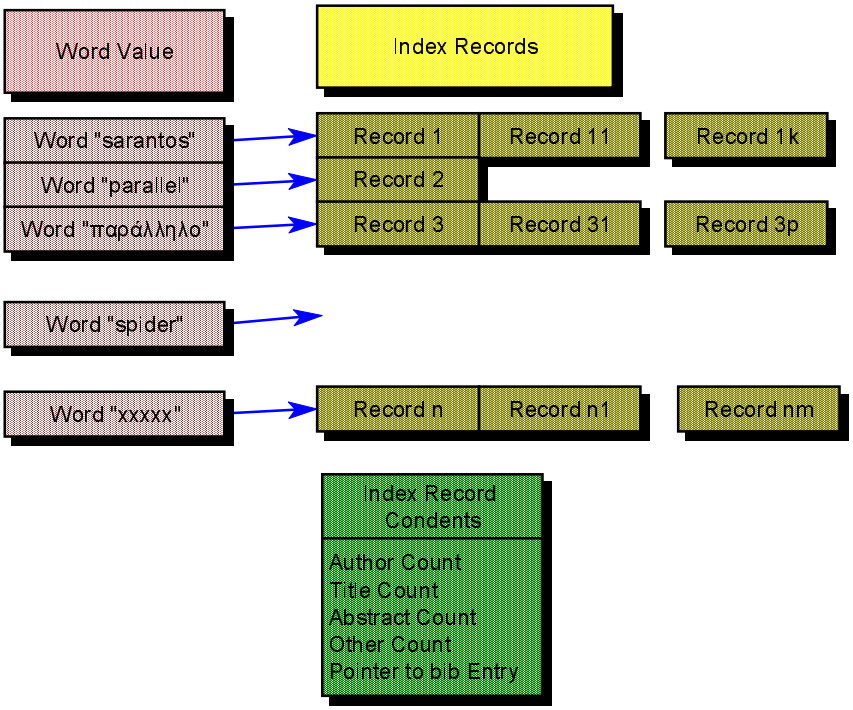



Σχήμα 26. Κλασική δομή αντίστροφης λίστας

Η παραπάνω δομή, δεν είναι αρκετή για την αποθήκευση των πληροφοριών που χρειάζεται το σύστημα. Έτσι χρησιμοποιείται μια μικρή παραλλαγή της παραπάνω μορφής. Τα επιπλέον στοιχεία που εισάγονται είναι, για κάθε λέξη είναι:

1. πόσα είναι τα κείμενα που περιέχουν την λέξη κλειδί.
2. πόσες φορές παρουσιάζεται η λέξη στα παραπάνω κείμενα.
3. το κάθε στοιχείο της λίστας, δεν είναι απλά ένας δείκτης στο αντίστοιχο κείμενο, αλλά μια δομή που περιέχει, εκτός του δείκτη στο κείμενο και πόσες φορές παρουσιάζεται η λέξη σε κάθε τμήμα του κειμένου (στον τίτλο, στο abstract κτλ.).

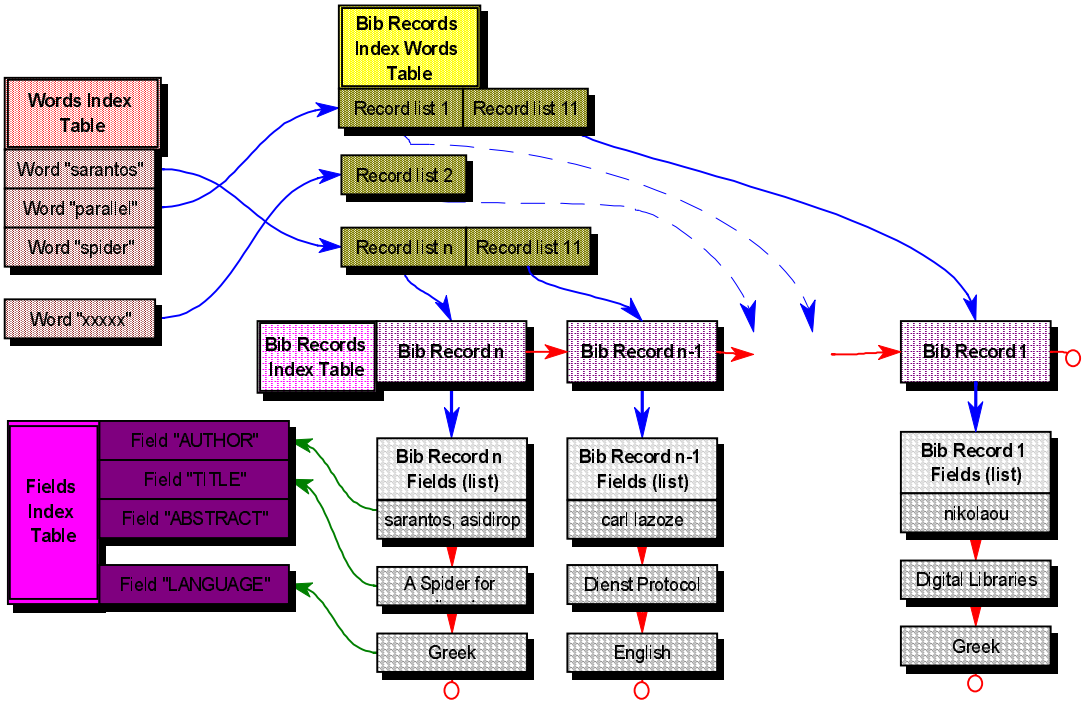
Ο λόγος για τον οποίο συμπεριλαμβάνονται και οι επιπλέον πληροφορίες, δεν είναι απλά στατιστικός, αλλά είναι απαραίτητα στην περίπτωση που θέλουμε να υπολογίσουμε ποσοστό ομοιότητας. Έτσι αποφεύγονται υπολογισμοί και μετρήσεις την ώρα της εκτέλεσης μιας αναζήτησης, αφού η πληροφορία μπορεί να είναι διαθέσιμη από πριν.




Σχήμα 27. Δομή αντίστροφης λίστας που χρησιμοποιείται

Στο επόμενο σχήμα (Σχήμα 28), φαίνεται τι περιέχει η μνήμη του spider (σε σχέση με τα αρχεία ευρετηριασμού) κατά την διάρκεια λειτουργίας του. Εδώ, να σημειώσουμε, ότι προκειμένου να έχουμε καλή απόδοση κατά την αναζήτηση προτιμούμε δεδομένα που θα διαβαστούν συγχρόνως να τα αποθηκεύουμε σε συνεχόμενες θέσεις μνήμης. Έτσι το "words_index_table"(Σχήμα 28), είναι ένας δυναμικός πίνακας, ταξινομημένος, στον οποίο η αναζήτηση γίνεται δυαδικά. Ομοίως και οι πίνακες "bib_records_index_words_table". Έτσι, κατά την αναζήτηση, ακόμη και αν η διεργασία βρισκόταν στην εικονική μνήμη του συστήματος (swap memory), το σύστημα θα ανακτήσει από αυτήν μόνο τον πίνακα "words_index_table" και τις αντίστοιχες γραμμές από τον "bib_records_index_words_table" τα οποία βρίσκονται σε κοντινές θέσεις μνήμης[35].

[35] Έτσι, κατά την δοκιμή του συστήματος σε work station με λίγη κύρια μνήμη (pleiades.csi.forth.gr-64MB RAM), για την εξυπηρέτηση μιας αίτησης, αρκούσε να ανακληθούν από την εικονική μνήμη μόνο 2MB από τα 60MB που καταλάμβανε η διεργασία . Αντιθέτως, στον ευρετηριαστή του dienst, για να εξυπηρετηθεί μια αίτηση αναζήτησης, πρέπει να ανακληθούν στην κύρια μνήμη πάνω την μισή ποσότητα μνήμης από αυτήν που καταλαμβάνει (περίπου 30MB από τα 55MB).




Σχήμα 28. Δομές Δεδομένων

## 10.2. Παράμετροι για τον εξυπηρετητή

Όπως αναφέρθηκε και πριν, το σύστημα είναι παραμετροποιημένο. Προκειμένου να είναι ευκολότερος ο ορισμός παραμέτρων, αυτές δίνονται σαν μεταβλητές περιβάλλοντος (environment variables) πριν την εκτέλεση του προγράμματος. Έτσι αποφεύγονται οι ορισμοί πολλών παραμέτρων στην γραμμή εντολής εκτέλεσης, οι οποίοι είναι κουραστικοί, και πολλές φορές οδηγείται ο υπεύθυνος στην παράληψη κάποιας ή κάποιων από αυτές.

Οι βασικές παράμετροι είναι οι εξής:

- *MASTER_META_SERVER_HOST, MASTER_META_SERVER_PORT:* πληροφορίες για την συλλογή.
- *SPIDER_PATH:* Η θέση που είναι εγκατεστημένο το σύστημα.
- *SPIDER_PORT:* Το port στο οποίο δέχεται αιτήσεις ο εξυπηρετητής.
- *MEMORY_RESIDENT_FIELDS:* Τα πεδία τα οποία θα αποθηκεύονται στην εσωτερική cache για γρηγορότερη πρόσβαση μετά την πρώτη ανάγνωσή τους. Η προκαθορισμένη τιμή του είναι "AUTHOR:TITLE:DATE".

Επίσης, κατά την διάρκεια την μετάφρασης του προγράμματος, μπορούν να οριστούν κάποιες άλλες παράμετροι, όπως ο τρόπος αποθήκευσης των αρχείων ευρετηριασμού. Αυτός μπορεί να είναι απλό κείμενο (text όπως στο dienst) το οποίο είναι αναγνώσιμο με το μάτι, ή δυαδικός (binary) με τον οποίον επιταχύνεται η διαδικασία ανάγνωσης-γραφής στο αρχείο καθώς και μειώνεται σημαντικά το μέγεθος των αρχείων με τα ευρετήρια.

# *Κεφάλαιο 11*

# 11. Dienst Spider: Στατιστικά – Απόδοση

## 11.1. Μετρήσεις Απόδοσης

Προκειμένου να κάνουμε μια αξιολόγηση της ταχύτητας του μηχανισμού αναζήτησης το συγκρίναμε με τον αντίστοιχο μηχανισμό του dienst. Για να γίνει αυτό υποστηρίζονται από τον spider οι αιτήσεις του dienst για αναζήτηση. Πιο συγκεκριμένα υλοποιήσαμε της αιτήσεις:

- /Dienst/Index/2.0/SeachBoolean
- /Dienst/Index/2.0/SearchBooleanCallback

Έτσι συγκρίνουμε τον εξυπηρετητή του spider με το κομμάτι του dienst που κάνει την αναζήτηση σε *έναν μόνο κόμβο*[36].

Έτσι τα συστήματα, όσον αφορά τις αιτήσεις για αναζήτηση, είναι απολύτως ισοδύναμα. Αν και αρχικά το σύστημα δεν ήταν σχεδιασμένο ώστε να δέχεται τέτοιου είδους αιτήσεις, και άρα δεν είχε βελτιστοποιηθεί για κάτι τέτοιο. Παρόλα αυτά όμως τα αποτελέσματα ήταν εντυπωσιακά.

Επίσης, είναι προφανές, ότι εφόσον ο server του dienst spider μπορεί να εξυπηρετήσει τέτοιου είδους αιτήσεις, μπορεί να αντικαταστήσει τον index server του dienst (σε έναν απλό κόμβο ή σε έναν κόμβο εξυπηρέτησης περιοχής - MIS).

Για τις μετρήσεις ακολουθήσαμε την εξής στρατηγική:

Πρώτον εγκαταστήσαμε τα συστήματα που θέλουμε να συγκρίνουμε στον ίδιο σταθμό εργασίας. Η βάση με τις τεχνικές αναφορές εγκαταστάθηκε στο ίδιο δίσκο, ώστε όλοι οι servers να έχουν την ίδια βάση και η καθυστέρηση λόγο πρόσβασης στον δίσκο να είναι η ίδια.

Δεύτερον, συγκρίναμε 3 συστήματα:

1. Τον server του dienst spider

[36] Στο dienst η κατανεμημένη αναζήτηση γίνεται μέσω του User Interface server. Όλες οι αιτήσεις προς έναν Index server του dienst εξυπηρετούνται τοπικά.

2. Τον Index Server του dienst με τον βασικό μηχανισμό ευρετηριασμού και αναζήτησης που τον συνοδεύει.

3. Τον Index Server του dienst με τον μηχανισμό ευρετηριασμού και αναζήτησης του glimpse.

Τον server του dienst spider, τον μετρήσαμε για 4 είδη αιτήσεων:

1. Την αίτηση τύπου dienst, όπου επιστρέφονται όλα τα αντικείμενα που ικανοποιούν την επερώτηση που περιέχεται στην αίτηση

2. Αίτηση της μορφής: /Spider/1.0/Html/Search με επιστροφή / εμφάνιση τις 20 πρώτες επιτυχίες

3. Αίτηση της παραπάνω μορφής με επιστροφή τις 40 πρώτες επιτυχίες

4. Και ομοίως με επιστροφή τις 80 πρώτες επιτυχίες.

Σε αυτό το σημείο θα πρέπει να επισημάνουμε ότι οι αιτήσεις του τύπου /Dienst/Index/2.0/SearchBoolean, δεν είναι αυτές που δίνει απευθείας ο χρήστης από τον browser. Στο σενάριο αναζήτησης από χρήστη, ο χρήστης επικοινωνεί με τον UI server του dienst, και ο UI server στέλνει την αίτηση SearchBoolean στον αντίστοιχο Index server. Αυτό σημαίνει ότι οι χρόνοι που θα μετρήσουμε στην περίπτωση μέτρησης αυτού του τύπου αίτησης του dienst, δεν είναι αυτοί που θα «δει» ο τελικός χρήστης. Ο χρήστης θα έχει μια επιπλέον καθυστέρηση στην εμφάνιση της απάντησης, αυτήν που θα προσθέσει η επικοινωνία με τον UI server και της μετατροπής της απάντησης σε html μορφή.

Αντιθέτως, στην περίπτωση των αιτήσεων της μορφής του spider (/Spider/Html/1.0/Search), το αποτέλεσμα μετατρέπεται απευθείας σε html μορφή, και άρα δεν θα έχουμε την επιπλέον καθυστέρηση (overhead) ενός UI server.

Τρίτον, για τις μετρήσεις χρησιμοποιήσαμε τα log files του εξυπηρετητή περιοχής νοτιοανατολικής Ευρώπης, που είναι εγκατεστημένος στο ΙΤΕ/ΙΠ[37].

Κάναμε 2 ειδών μετρήσεις. Η πρώτη είναι για να μετρήσουμε τον χρόνο απόκρισης. Η δεύτερη είναι για να μετρήσουμε το throughput του κάθε server σε ρυθμό εξυπηρετήσεων ανά λεπτό.

[37] Του Φεβρουάριου και Μαρτίου 1998.

### *11.1.1. Configuration των servers*

Προκειμένου να κάνουμε τις μετρήσεις ορίσαμε τους servers ώστε να έχουν τους ίδιους περιορισμούς, δηλαδή:

Από πλευράς dienst, ορίσαμε για τον web server, και για τον server του dienst, ένα μέγιστο πλήθος πελατών (συγχρόνως) ίσο με 30. Ομοίως και για τον spider, ορίσαμε το ίδιο μέγιστο.

### *11.1.2. Η βάση δεδομένων*

Χρησιμοποιήσαμε την ίδια βάση για όλους τους servers. Η βάση είναι όλες οι τεχνικές αναφορές της συλλογής ncstrl, η οποία αποτελείται περίπου από 25000 τεχνικές αναφορές, οι οποίες περιέχουν περίπου 55000 διαφορετικές λέξεις στα metadata αρχεία.

### *11.1.3. Ο σταθμός εργασίας με τους servers*

Όπως αναφέρθηκε και πριν οι μετρήσεις γίνανε στον ίδιο σταθμό εργασίας, περίπου την ίδια ώρα. Έτσι η καθυστέρηση εξυπηρέτησης λόγω φόρτου του μηχανήματος (από άλλα προγράμματα είναι κοινή).

Το μηχάνημα στο οποίο έγιναν οι μετρήσεις είναι ένα Ultra Sparc (sappho.csi.forth.gr) με 512MB RAM και 2GB swap space[38].

### *11.1.4. Ο πελάτης*

Για κάθε αίτηση που στέλνει ο πελάτης, μετράει 4 χρόνους. Ο πρώτος είναι ο χρόνος που χρειάζεται ο πελάτης να συνδεθεί στον εξυπηρετητή και να στείλει την αίτηση (connect time). Ο δεύτερος είναι ο χρόνος που μεσολαβεί ώστε να αρχίσει ο server να απαντάει στην αίτηση (response time). Είναι ο χρόνος από την στιγμή που στείλαμε την αίτηση μέχρι να πάρουμε το πρώτο byte απάντησης. Αυτός είναι ουσιαστικά ο χρόνος επεξεργασίας της αίτησης. Ο τρίτος χρόνος, είναι ο χρόνος μέχρι να ολοκληρωθεί η παραλαβή της απάντησης (transfer time). Και ο τέταρτος είναι το άθροισμα των τριών προηγούμενων (total time).

Το timeout του πελάτη ορίστηκε στα 20sec (δεδομένου ότι πελάτης και εξυπηρετητής βρίσκονται στο ίδιο τοπικό δίκτυο).

---

[38] Αρχικά έγινε προσπάθεια οι μετρήσεις να γίνουν στο σταθμό εργασίας crete.csi.forth.gr, ο οποίος είναι Ultra sparc με 256MB RAM και 500MB swap space. Όμως σε αυτό το μηχάνημα, αν και ο spider-server μπορούσε να λειτουργήσει κανονικά, δεν μπορούσε να λειτουργήσει ο dienst server για πολλούς πελάτες συγχρόνως. Στον dienst server, για σύγχρονη εξυπηρέτηση 10 αιτήσεων, απαιτούνται 10*55MB=550MB μνήμης, τα οποία, λόγω φόρτου του σταθμού εργασίας, ήταν αδύνατο να δεσμευτούν. Άρα είχαμε ένα πάρα πολύ μεγάλο ποσοστό άρνησης του dienst server για την εξυπηρέτηση της επερώτησης.

- Δηλαδή ο πελάτης προσπαθεί να συνδεθεί στον server για 6 δεύτερα. Αν δεν το καταφέρει μέσα σε 6 δεύτερα, τότε αποτυγχάνει.

- Έπειτα περιμένει για t δεύτερα το πολύ να αρχίσει να έρχεται η απάντηση. Αν μέσα σε t δεύτερα δεν αρχίσει να έρχεται η απάντηση, τότε πάλι αποτυγχάνει. Το t στην περίπτωση των πολλών σύγχρονων πελατών είναι ίσο με 20, ενώ στην περίπτωση του ενός πελάτη είναι ίσο με 60[39].

- Καθώς παραλαμβάνει την απάντηση (η οποία στις περιπτώσεις αιτήσεων τύπου dienst είναι μεγάλη), αποτυγχάνει αν για 6 δευτερόλεπτα δεν παραλάβει κανένα byte από τον server (stalled).

### *11.1.5. Μέτρηση του χρόνου απόκρισης*

Για κάθε αίτηση, για την οποία θέλουμε να μετρήσουμε την ταχύτητα εξυπηρέτησης, στέλνουμε στον κάθε server 3 συνεχόμενες αιτήσεις (τις ίδιες). Αυτό διότι είναι πολύ πιθανό, λόγω φόρτου του συστήματος, η διεργασία του εξυπηρετητή πριν την πρώτη αίτηση να έχει φύγει από την κύρια μνήμη του συστήματος (να έχει αποθηκευτεί στο swap space). Άρα η εξυπηρέτηση της πρώτης αίτησης θα είναι περισσότερο αργή. Κατά την 2η και 3η αίτηση (οι οποίες γίνονται αμέσως μετά), η διεργασία έχει μεταφερθεί στην κύρια μνήμη, οπότε και η εξυπηρέτησή της είναι περισσότερο γρήγορη. Επιπλέον κατά την εξυπηρέτηση της 3ης αίτησης, είναι πιθανόν τα στοιχεία που πρόκειται να διαβαστούν από τον δίσκο να έχουν γίνει cached. Για τις μετρήσεις κρατούμε 4 νούμερα: τον χρόνο για την εξυπηρέτηση της 1ης, 2ης και 3ης αίτησης, και τον μέσο όρο των τριών παραπάνω χρόνων.

Επαναλαμβάνουμε την ίδια διαδικασία και για τους υπόλοιπους servers για τους οποίους μετρούμε την απόδοση.

### *11.1.6. Μέτρηση του throughput*

Για να μετρήσουμε το throughput, στέλνουμε σε κάθε server, για 5 λεπτά αιτήσεις, από 10 πελάτες συγχρόνως. Ο αλγόριθμος έχει ως εξής:

Ο συντονιστής των πελατών διαβάζει τα διαθέσιμα log files. Για κάθε log file, το οποίο αντιστοιχεί στις αιτήσεις μιας μέρας, δημιουργεί αντίγραφο του εαυτού του, το οποίο αναλαμβάνει να κάνει τις αιτήσεις αυτές. Το πλήθος των αιτήσεων μιας μέρας μπορεί να είναι είτε μικρό π.χ. 2-3 αιτήσεις, είτε μεγαλύτερο (περισσότερες από 40).

Ανά πάσα στιγμή, ο συντονιστής έχει 5 αντίγραφα του εαυτού του, τα οποία στέλνουν αιτήσεις, οι οποίες αντιστοιχούν σε διαφορετικά αρχεία. Όταν κάποιο από τα αντίγραφα, τελειώσει με τις αιτήσεις που του έχουν ανατεθεί, τότε αποθηκεύει σε ένα αρχείο, το χρόνο που του πήρε να στείλει τις αιτήσεις, το πλήθος των αιτήσεων, και το πλήθος των αιτήσεων που εξυπηρετήθηκαν επιτυχώς. Ο συντονιστής, από την μεριά του ελέγχει αν κάποιο από τα αντίγραφα που δημιούργησε τελείωσε την δουλειά που του είχε ανατεθεί, αν ναι, τότε δημιουργείται ένα άλλο αντίγραφο στην θέση του, και διαβάζονται τα αποτελέσματα του προηγούμενου αντιγράφου από το αρχείο που αυτό είχε δημιουργήσει.

Όταν τελειώσει ο προβλεπόμενος χρόνος (στην περίπτωσή μας 5 λεπτά), τότε ο συντονιστής στέλνει σήμα στα αντίγραφα που είναι ενεργά να σταματήσουν να στέλνουν αιτήσεις. Το κάθε αντίγραφο, αφού πάρει αυτό το σήμα, περιμένει μέχρι να εξυπηρετηθεί η τελευταία εκκρεμής αίτηση που έστειλε, και αφού λάβει απάντηση για αυτήν (ή κάνει timeout), τότε γράφει τους χρόνος στο αρχείο και τελειώνει. Ο συντονιστής περιμένει μέχρι να τελειώσουν όλα τα αντίγραφα που είχε δημιουργήσει και διαβάζει τα αντίστοιχα αρχεία. Ο χρόνος από την στιγμή που ο συντονιστής έστειλε το σήμα, μέχρι να τελειώσουν όλα τα αντίγραφα, είναι περίπου 2-3 δευτερόλεπτα, αν δεν έχει «κολλήσει» κάποιο από τα αντίγραφα.

Η ίδια διαδικασία επαναλαμβάνεται και για τους επόμενους servers.

### *11.1.7. Αναμενόμενα αποτελέσματα*

Σύμφωνα με την σχεδίαση του κάθε server, περιμένουμε να επιβεβαιωθούν κάποιες υποθέσεις που έχουμε κάνει. Αυτές είναι:

- Περιμένουμε το response time του dienst, να είναι μεγαλύτερο από το αντίστοιχο του spider, για 2 λόγους:
    - στην περίπτωση του dienst έχουμε ένα δεδομένο overhead, αυτό της εκκίνησης του cgi-script και της μεταφοράς της αίτησης από το cgi-script στον dienst server.
    - Ο μηχανισμός αναζήτησης του spider, που είναι υλοποιημένος σε C, πρέπει να είναι γρηγορότερος από αυτόν σε perl.
- Περιμένουμε το transfer time, να είναι περίπου το ίδιο για το dienst και spider για αιτήσεις της μορφής του dienst, διότι τα στοιχεία που επιστρέφονται διαβάζονται από τον δίσκο.

---

[39] Αρχικά, το T, είχε οριστεί και αυτό στα 6secs, αλλά το dienst δεν προλάβαινε στην πλειοψηφία των περιπτώσεων να απαντήσει και γινόταν timeout.

- Για ερωτήσεις της μορφής του spider (με 20,40 και 80 επιτυχίες στην απάντηση), περιμένουμε το transfer time να είναι μικρότερο από το αντίστοιχο για αίτηση μορφής dienst. Αυτό διότι θα μεταφερθούν πολύ λιγότερα δεδομένα. Επίσης περιμένουμε το transfer time να είναι περίπου ανάλογο του πλήθους των επιτυχιών που επιστρέφονται (δηλαδή 20/40/80).

Επίσης, ξέρουμε για το dienst, ότι το πλήθος των αιτήσεων που μπορεί να εξυπηρετήσει συγχρόνως είναι αντίστοιχο της διαθέσιμης μνήμης στον σταθμό εργασίας που είναι εγκατεστημένο. Έτσι αν ο σταθμός εργασίας έχει 100MB διαθέσιμη μνήμη, τότε μπορούμε να εξυπηρετήσουμε το πολύ 2 αιτήσεις συγχρόνως. Και άρα αν στέλνουμε αιτήσεις από 10 διαφορετικούς πελάτες, οι 8 στις 10 θα αποτύχουν[40].

## *11.1.8. Αποτελέσματα*

### *11.1.8.1. Μέτρηση του χρόνου απόκρισης*

Αυτό που δείχνουμε στα γραφήματα, είναι το ποσοστό επί τις εκατό των αιτήσεων που εξυπηρετήθηκαν σε χρόνο μικρότερο του t (άξονας χ). Η μέτρηση, λοιπόν, των χρόνων απόκρισης, έδωσε τα παρακάτω αποτελέσματα (Γράφημα 1).

Στους παρακάτω πίνακες (Πίνακας 13 και Πίνακας 14), φαίνονται οι μέσοι χρόνοι απόκρισης των συστημάτων που συγκρίνουμε. Στον πρώτο πίνακα από τους δύο βλέπουμε την ανάλυση των μέσων χρόνων στους επιμέρους, και στον δεύτερο την ανάλυση του μέσου χρόνου στις «πρώτες», «δεύτερες» και «τρίτες» αιτήσεις που τους στάλθηκαν.

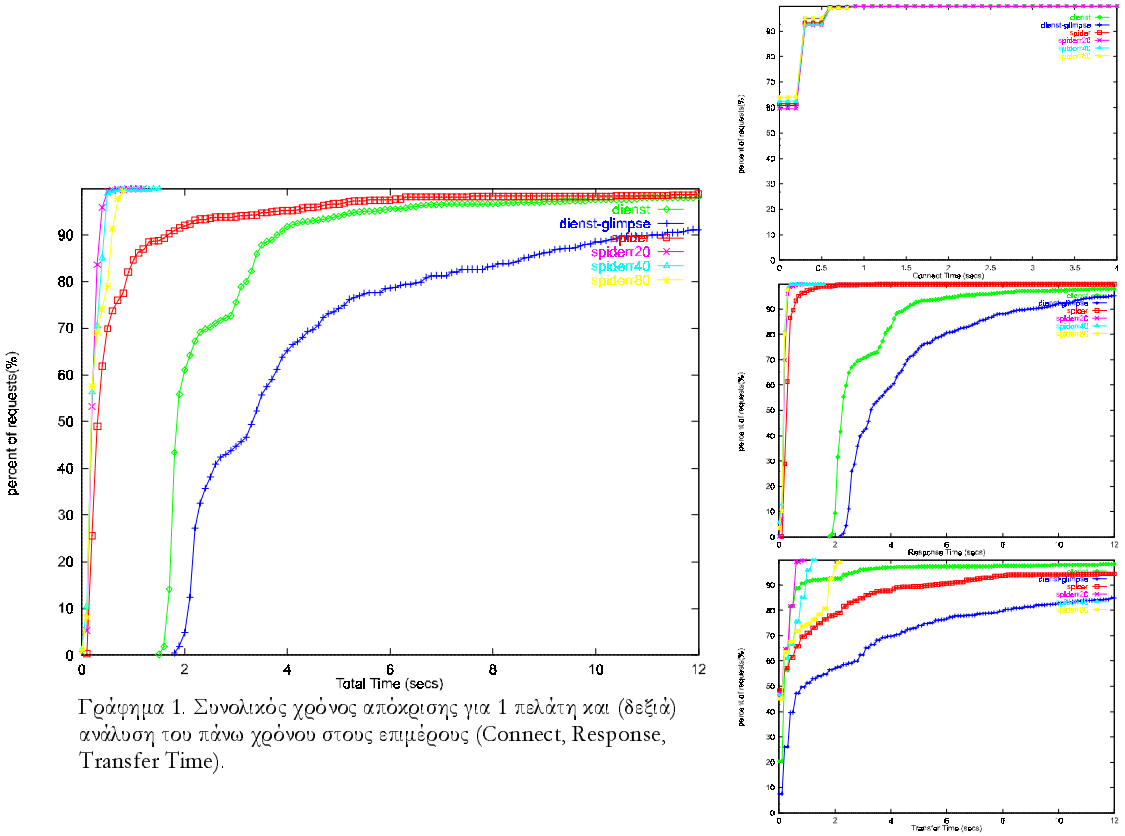
[40] Αυτό ακριβώς συνέβηκε κατά την προσπάθεια μέτρησης στον σταθμό εργασίας crete, διότι η διαθέσιμη μνήμη ήταν περίπου 100MB.

| | Total | Connect | Response | Transfer |
|---|---|---|---|---|
| **dienst** | 2,678025 | 0,025031 | 2,481661 | 0,171333 |
| | | 0,93% | 92,67% | 6,40% |
| **dienst-glimpse** | 5,623744 | 0,020028 | 3,874519 | 1,729197 |
| | | 0,36% | 68,90% | 30,75% |
| **spider** | 0,899388 | 0,020715 | 0,280663 | 0,59801 |
| | | 2,30% | 31,21% | 66,49% |
| **spiderr20** | 0,213411 | 0,023592 | 0,142058 | 0,04776 |
| | | 11,05% | 66,57% | 22,38% |
| **spiderr40** | 0,223624 | 0,020715 | 0,126737 | 0,076172 |
| | | 9,26% | 56,67% | 34,06% |
| **spiderr80** | 0,272176 | 0,018629 | 0,127528 | 0,126018 |
| | | 6,84% | 46,86% | 46,30% |

Πίνακας 13. Ανάλυση των μέσων χρόνων απόκρισης

| | total | 1rt req. | 2nd req. | 3rd req. |
|---|---|---|---|---|
| dienst | **2,67056** | 3,46683 | 2,24393 | 2,30091 |
| dienst-glimpse | **5,65129** | 6,16911 | 5,41490 | 5,36987 |
| spider | **0,91055** | 1,10022 | 0,82497 | 0,80647 |
| spider20 | **0,21545** | 0,25478 | 0,19759 | 0,19396 |
| spider40 | **0,22462** | 0,22533 | 0,22469 | 0,22384 |
| spider80 | **0,27277** | 0,27484 | 0,26950 | 0,27398 |

Πίνακας 14. Μέσοι χρόνοι απόκρισης



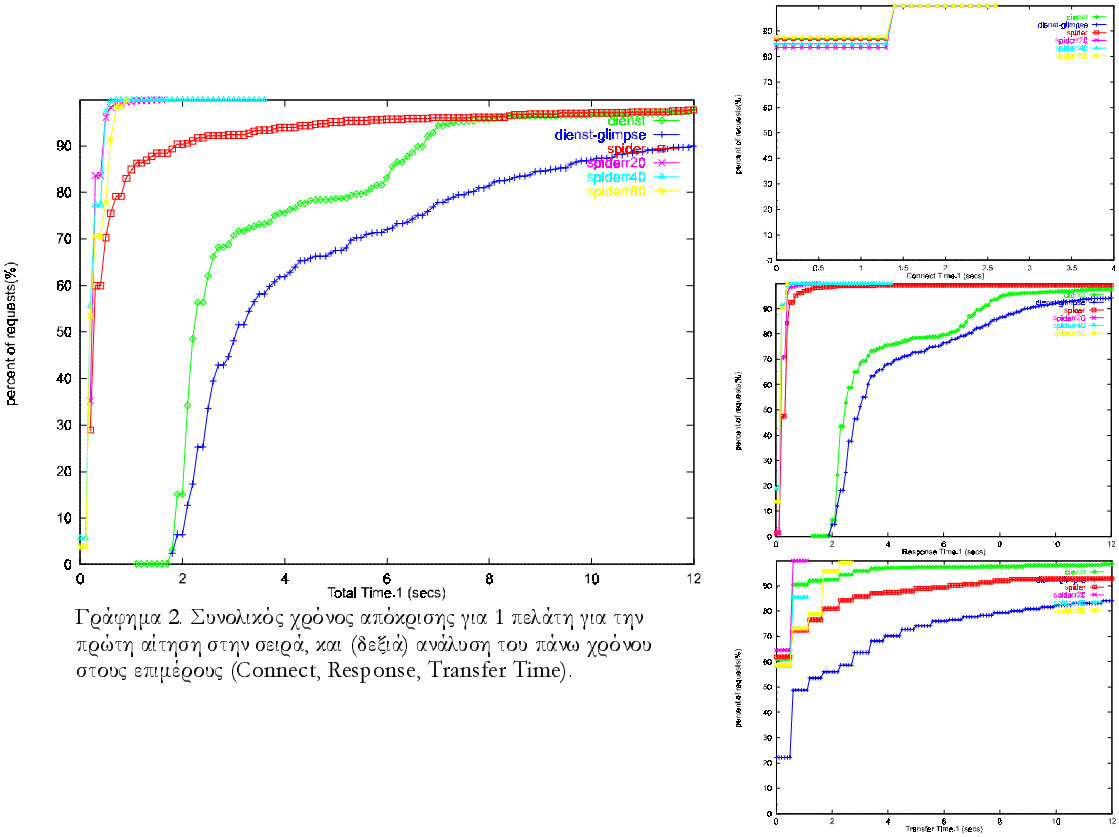

Γράφημα 1. Συνολικός χρόνος απόκρισης για 1 πελάτη και (δεξιά) ανάλυση του πάνω χρόνου στους επιμέρους (Connect, Response, Transfer Time).

Στο Γράφημα 1, βλέπουμε τον συνολικό χρόνο απόκρισης, για κάθε ένα από τα συστήματα που μετρούμε. Ακριβώς δίπλα (όπως και σε όλα τα επόμενα γραφήματα), υπάρχει η ανάλυση του συνολικού χρόνου στους επιμέρους χρόνους: χρόνος σύνδεσης, χρόνος έναρξης απόκρισης, χρόνος μεταφοράς απάντησης.

Είναι προφανές από αυτό το γράφημα ότι ο συνολικός χρόνος εξυπηρέτησης των αιτήσεων προς τον spider, είναι πολύ μικρότερος από τον αντίστοιχο του dienst. Επίσης φαίνεται ότι οι αιτήσεις που περιλαμβάνουν το μέγιστο πλήθος των επιτυχιών που θα επιστραφούν, είναι πολύ γρηγορότερες από τις υπόλοιπες.

Επίσης, φαίνεται ότι αυτό από το οποίο «πάσχει» το dienst, είναι ο χρόνος έναρξης της απόκρισης. Βλέπουμε ότι σχεδόν στο 100% των αιτήσεων που έγιναν προς το dienst, η απάντηση άρχισε να μεταφέρεται μετά από περίπου 2 δεύτερα. Αυτό συμβαίνει διότι υπάρχει μια ελάχιστη καθυστέρηση στην μεταφορά της αίτησης στον http server, έναρξη του cgi-script,

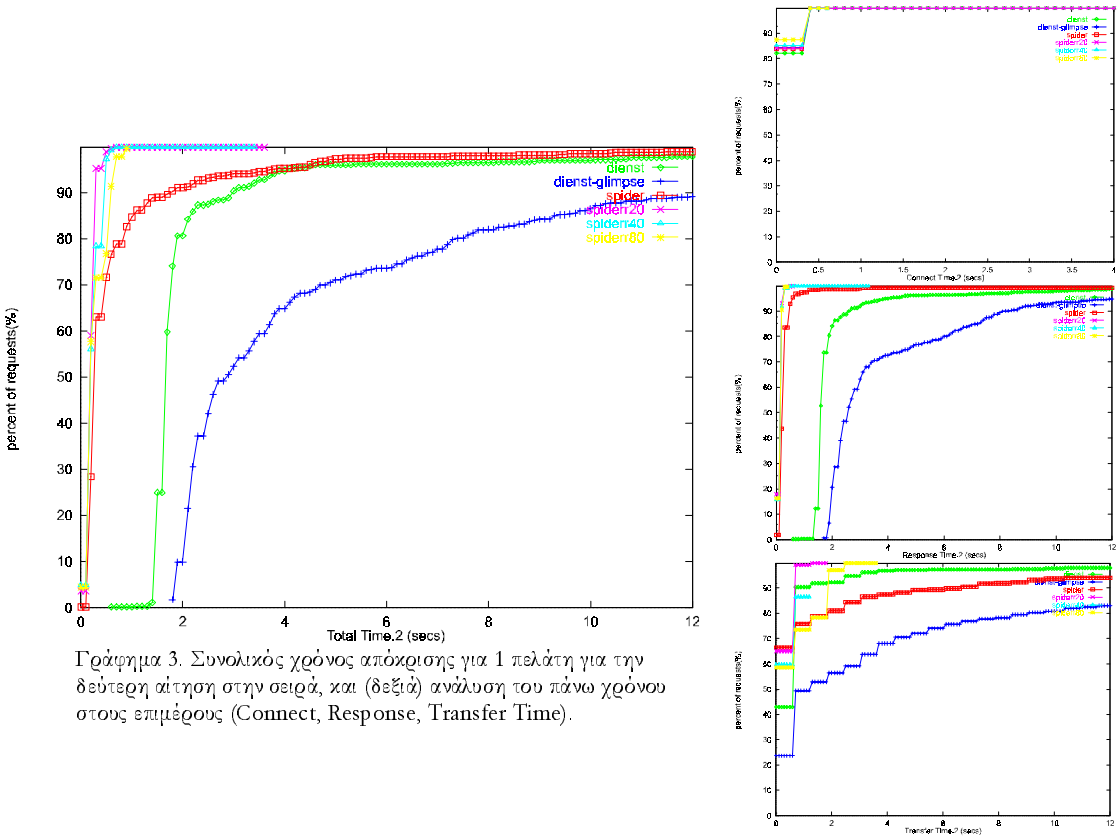
μεταφορά της αίτησης στο cgi-script και από εκεί στον server. Επίσης υπάρχει και μια καθυστέρηση λόγο της δημιουργίας αντιγράφου της διεργασίας του εξυπηρετητή.




Γράφημα 2. Συνολικός χρόνος απόκρισης για 1 πελάτη για την πρώτη αίτηση στην σειρά, και (δεξιά) ανάλυση του πάνω χρόνου στους επιμέρους (Connect, Response, Transfer Time).

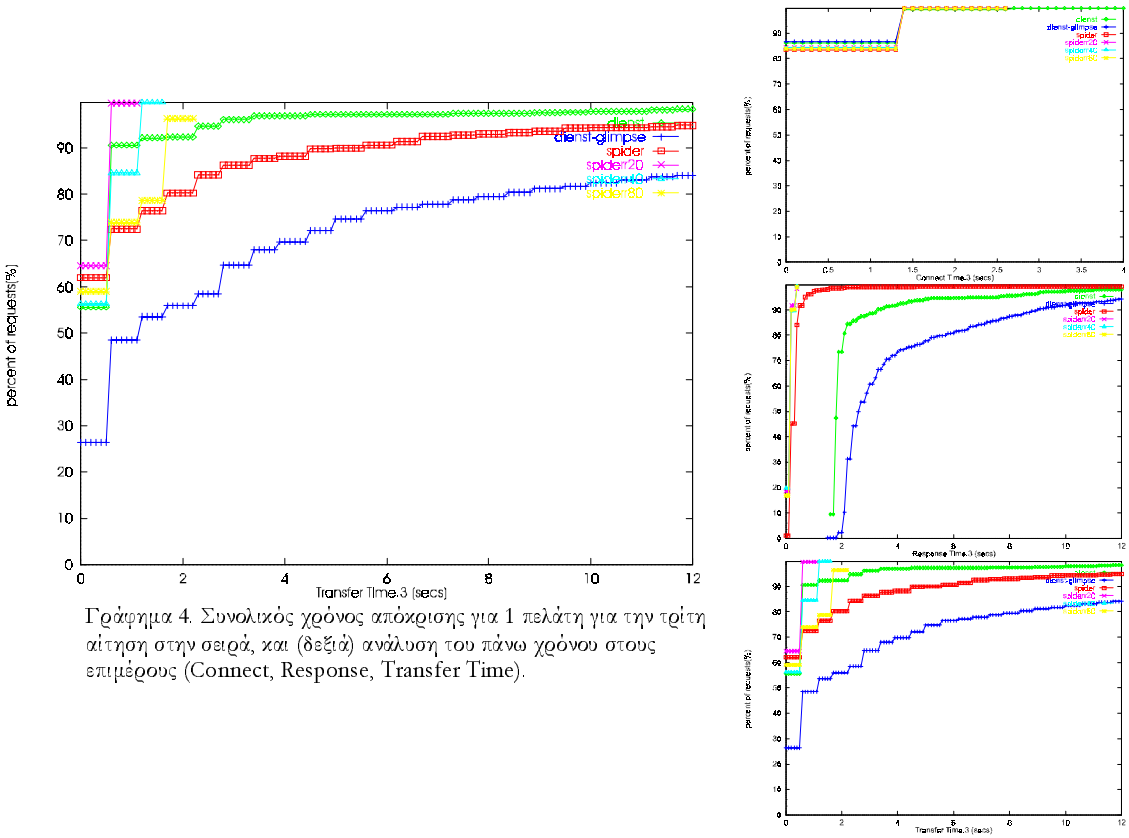
Στα υπόλοιπα γραφήματα (Γράφημα 2, Γράφημα 3, Γράφημα 4), βλέπουμε τους χρόνους απόκρισης για τις αιτήσεις υπ' αριθμόν 1,2 και 3, δηλαδή τους χρόνους για να εξυπηρετηθεί η πρώτη αίτηση που φτάνει στον εξυπηρετητή, η 2η αίτηση (αμέσως μετά), και η 3η αίτηση.

Όπως έχει αναφερθεί και παραπάνω, κατά την πρώτη αίτηση, είναι πολύ πιθανόν η διεργασία του εξυπηρετητή να έχει μεταφερθεί στην εικονική μνήμη του συστήματος. Άρα, για να εξυπηρετήσει μια αίτηση, πρέπει κάποια τμήματά της να μεταφερθούν στην κύρια μνήμη. Όσο μικρότερα είναι αυτά τα τμήματα, και όσο περισσότερο κοντινά είναι, τόσο γρηγορότερα θα γίνει αυτή η μεταφορά, και άρα θα αρχίσει να εξυπηρετείται η αίτηση.




Γράφημα 3. Συνολικός χρόνος απόκρισης για 1 πελάτη για την δεύτερη αίτηση στην σειρά, και (δεξιά) ανάλυση του πάνω χρόνου στους επιμέρους (Connect, Response, Transfer Time).

Από όλα τα γραφήματα, βλέπουμε ότι η απόδοση του spider, είναι σχεδόν σταθερή, ενώ για το dienst, είναι χειρότερη στην πρώτη αίτηση, και βελτιώνεται στις επόμενες. Αυτό συμβαίνει διότι η ποσότητα των δεδομένων που πρέπει να μεταφερθεί στην κύρια μνήμη από την εικονική, είναι μεγάλο στην περίπτωση του dienst. Και άρα, όταν η διεργασία είναι στην εικονική μνήμη αργεί να μεταφερθεί στην κύρια. Αντιθέτως, ο spider, για να εξυπηρετήσει μια αίτηση χρειάζεται λίγα δεδομένα, και επιπλέον αυτή τα δεδομένα βρίσκονται σε συνεχόμενες θέσεις μνήμης (και άρα σε λίγες σελίδες μνήμης – pages).




Γράφημα 4. Συνολικός χρόνος απόκρισης για 1 πελάτη για την τρίτη αίτηση στην σειρά, και (δεξιά) ανάλυση του πάνω χρόνου στους επιμέρους (Connect, Response, Transfer Time).

*11.1.8.2. Μέτρηση του throughput*

Σε αυτήν την παράγραφο παρουσιάζουμε τα αποτελέσματα μέτρησης του throughput. Παράλληλα όμως μετρήσαμε και τους χρόνους απόκρισης για κάθε επιτυχημένη αίτηση προς τον αντίστοιχο server.

Στον επόμενο πίνακα (Πίνακας 15), φαίνονται τα αποτελέσματα μέτρησης της απόδοσης των συστημάτων. Τα νούμερα που αναφέρονται σε αυτόν τον πίνακα είναι οι μέσοι χρόνοι για 5 δοκιμές (από 13 έως 17 Νοεμβρίου 1998 στο σταθμό εργασίας sappho, με τους πελάτες στον σταθμό εργασίας estia). Σε αυτό το σημείο πρέπει να αναφέρουμε ότι κατά την διάρκεια των μετρήσεων, ο σταθμός εργασίας ήταν φορτωμένος και με άλλες διεργασίες. Όλες όμως οι δοκιμές γίνανε με περίπου τον ίδιο βαθμό φόρτου του σταθμού εργασίας.

Είναι προφανές από τον πίνακα, η καλή συμπεριφορά του spider server ιδίως σε σχέση με το dienst. Για τις ίδιες ακριβώς αιτήσεις με το dienst, μπορεί να εξυπηρετήσει τριπλάσιο πλήθος αιτήσεων στον ίδιο χρόνο.

Επίσης, όταν η απάντηση προς την αίτηση περιέχει μόνο τις 20 (ή 40 ή 80) πρώτες επιτυχίες, η απόδοση υπερδιπλασιάζεται.

| | dienst | dienst glimpse | spider | spider20 | spider40 | spider80 |
|---|---|---|---|---|---|---|
| **χρόνος αιτήσεων** | 180,00 | 180,00 | 180,00 | 180,00 | 180,00 | 180,00 |
| **συνολικός χρόνος αιτήσεων** | 196,00 | 209,00 | 248,00 | 196,00 | 196,00 | 197,00 |
| **αιτήσεις που στάλθηκαν** | 173 | 61 | 483 | 972 | 863 | 710 |
| **αιτήσεις που εξυπηρετήθηκαν επιτυχώς** | 155 | 48 | 480 | 972 | 863 | 710 |
| **throughput (αιτήσεις/λεπτό)** | 51,666667 | 16 | 160 | 324 | 287,66667 | 236,66667 |
| **ποσοστό επιτυχούς απάντησης** | 89,6% | 78,7% | 99,4% | 100,0% | 100,0% | 100,0% |

Πίνακας 15. Απόδοση των συστημάτων για 5 πελάτες συγχρόνως.

| | |
|---|---|
| dienst | **1,2271335** |
| dienst-glimpse | **4,2974878** |
| spider | **0,6526752** |
| spider20 | **0,1622589** |
| spider40 | **0,1822563** |
| spider80 | **0,2439301** |

Πίνακας 16. Μέσοι χρόνοι απόκρισης για 5 πελάτες.

Στο παρακάτω γράφημα (Γράφημα 5), βλέπουμε τους χρόνους απόκρισης. Σε αυτό το γράφημα, φαίνεται ότι και στην περίπτωση των παράλληλων αιτήσεων, το spider έχει καλύτερη απόδοση απ' ότι ο μηχανισμός αναζήτησης του dienst. Μάλιστα, το dienst με μηχανισμό ευρετηριασμού του glimpse, έχει πολύ κακή απόδοση.

Κάτι άλλο που μπορούμε να παρατηρήσουμε στο Γράφημα 5, είναι ότι το dienst έχει βελτιώσει σε αυτήν την περίπτωση τον χρόνο έναρξης απόκρισης. Αυτό, λόγω του ότι η διεργασία είναι συνεχώς ολόκληρη στην κύρια μνήμη, άρα η δημιουργία αντιγράφου της διεργασίας και η αναζήτηση η ίδια είναι περισσότερο γρήγορη. Εδώ βέβαια, πρέπει να υπενθυμίσουμε 2 σημεία:

1. Το παρακάτω γράφημα, περιέχει μόνο τους χρόνους για τις αιτήσεις που τελικώς εξυπηρετήθηκαν επιτυχώς.

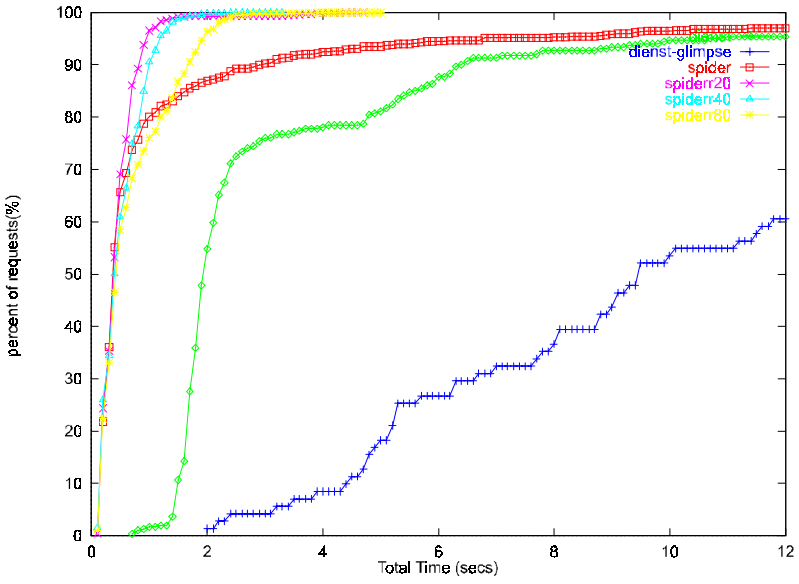
2. Πάντα στον μηχανισμό του dienst, υπάρχει ο περιορισμός των ενεργών διεργασιών (και άρα παράλληλων αιτήσεων), με βάση την ποσότητα μνήμης που διαθέτει ο σταθμός εργασίας στον οποίο λειτουργεί ο εξυπηρετητής.

Επίσης φαίνεται από το γράφημα, ότι αδύνατο σημείο στον spider είναι η μεταφορά των δεδομένων. Αυτό βέβαια δεν μας ενοχλεί διότι:

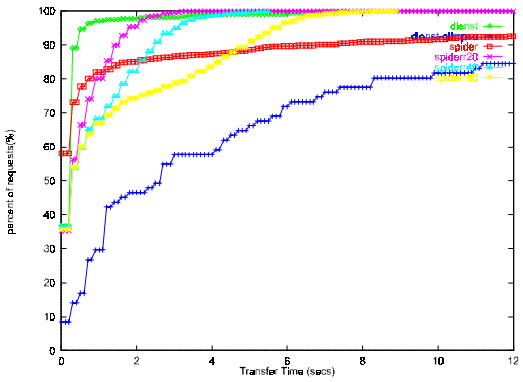
1. Η μεταφορά των δεδομένων προς τον χρήστη μέσω του διαδικτύου, είναι έτσι κι αλλιώς αργή διαδικασία (σε σχέση με τις υπόλοιπες).

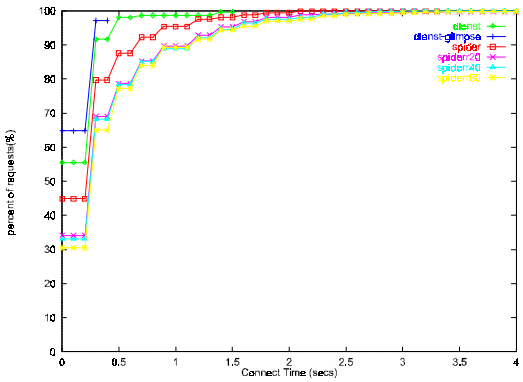
2. Ο συνολικός χρόνος ολοκλήρωσης της εξυπηρέτησης της αίτησης, εξακολουθεί να είναι κατά πολύ μικρότερος από ότι του dienst.

Από την άλλη, αν δούμε τους χρόνους, όχι με ποσοστά, αλλά με απόλυτα μεγέθη, θα δούμε ότι πάλι είναι καλύτεροι οι χρόνοι μεταφοράς των δεδομένων του spider. Δηλαδή αναλυτικά έχουμε: για το 90% των αιτήσεων στον dienst, τα δεδομένα μεταφέρθηκαν σε λιγότερο από 0,75 δεύτερα για κάθε αίτηση, ενώ το 77% των αιτήσεων στον spider (για αίτηση τύπου dienst όπου και μεταφέρονται ακριβώς η ίδια ποσότητα δεδομένων), τα δεδομένα μεταφέρθηκαν σε λιγότερο από 0,75 δεύτερα για κάθε αίτηση. Αλλά, μέσα σε 1 λεπτό, το dienst εξυπηρέτησε 51 αιτήσεις, ενώ ο spider 160. Άρα το πλήθος των αιτήσεων (μέσα σε ένα λεπτό) των οποίων τα δεδομένα μεταφέρθηκαν σε λιγότερο από 0,75 δεύτερα, είναι 46 (=0,9*51) αιτήσεις για το dienst, ενώ για τον spider 123 (=0,77*160) αιτήσεις.

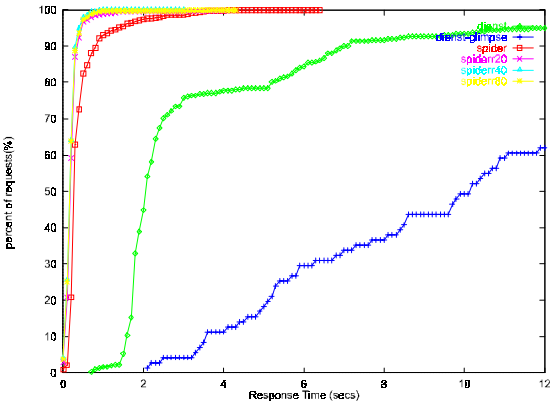



Γράφημα 5. Συνολικός χρόνος απόκρισης για 5 πελάτες συγχρόνως.




Γράφημα 6. Ανάλυση του πάνω χρόνου (Γράφημα 5) στους επιμέρους (Connect, Response, Transfer Time).



# Μέρος 4ο:
# Παραρτήματα

# ΠΑΡΑΡΤΗΜΑΤΑ

# *Παράρτημα Ι*

## I. Uniform Resource Locators – URLs

Το URL μας δίνουν έναν ενιαίο τρόπο ονομασίας και αναφοράς στους πόρους του διαδικτύου (internet resources). Η σύμβαση για τον τρόπο ονομασίας έχει ως εξής:

*{protocol}://{host}:{port}/{path}?{arguments}#{reference}*

Το πρώτο τμήμα του ονόματος περιλαμβάνει το πρωτόκολλο (protocol). Το περισσότερο συνηθισμένο είναι το http, το οποίο πλέον έχει γίνει το βασικό πρωτόκολλο επικοινωνίας του διαδικτύου. Άλλα σημαντικά πρωτόκολλα είναι τα: shttp, ftp, ghofer, mailto, WAIS, telnet, tn3270, news.

Το δεύτερο τμήμα της ονομασίας περιέχει το όνομα που έχει ο υπολογιστής στο διαδίκτυο (hostname) ή την διεύθυνση του υπολογιστή (IP address) στον οποίο είναι αποθηκευμένος ο πόρος στον οποίο ζητούμε πρόσβαση.

Το τρίτο τμήμα (το οποίο είναι προαιρετικό) περιέχει την θύρα (port) του υπολογιστή από το οποίο είναι διαθέσιμος ο πόρος στον οποίο ζητούμε πρόσβαση. Όταν αυτό το τμήμα παραλείπεται, εννοείται η προκαθορισμένη (default) θύρα η οποίο αντιστοιχεί στο δοθέν πρωτόκολλο. Για παράδειγμα το http έχει σαν προκαθορισμένη θύρα το νούμερο 80, το ftp το νούμερο 21 κ.ο.κ.[41], οπότε και δεν χρειάζεται να γράψουμε: *http://www.csi.forth.gr:80/* αλλά αρκεί το *http://www.csi.forth.gr/*.

Το επόμενο τμήμα περιέχει την θέση του πόρου και το όνομα του. Συνήθως ο πόρος είναι κάποιο αρχείο (ιδίως στα πρωτόκολλα ftp και http), οπότε εδώ έχουμε το path στο οποίο βρίσκεται το αρχείο και το όνομα αυτού.

Μετά το διαχωριστικό «?» ακολουθούν οι παράμετροι για τον πόρο. Φυσικά κάτι τέτοιο δεν έχει νόημα όταν ο πόρος είναι αρχείο. Μπορεί όμως να μην είναι αρχείο, αλλά να είναι κάποιο πρόγραμμα που κάνει κάποιον υπολογισμό. Στο πρωτόκολλο http είναι συνηθισμένη αυτή η

[41] Μπορούμε πολύ εύκολα να βρούμε ποια είναι τα default ports για κάθε πρωτόκολλο. Αυτά περιέχονται στο αρχείο /etc/services σε ένα UNIX σύστημα ή στο αρχείο {windows path}\Services (συνήθως C:\Windows\Services) σε ένα σύστημα με Microsoft Windows.

περίπτωση (και το πρόγραμμα λέγεται cgi). Τα ορίσματα μεταξύ τους διαχωρίζονται με τον χαρακτήρα «&». Για να υπάρχει σαφήνεια στα ορίσματα, ο τύπος που χρησιμοποιείται έχει τη μορφή:

?{arg1_name}={arg1_value}&{arg2_name}={arg2_value}…

Δεδομένου ότι υπάρχουν πολλοί δεσμευμένοι χαρακτήρες (?, =, &, /, #) , είναι προφανές ότι αυτοί δεν μπορούν να χρησιμοποιηθούν για ονόματα αρχείων ή για ονόματα και τιμές παραμέτρων. Έτσι όταν πρόκειται να χρησιμοποιηθεί κάποιος από αυτούς τους χαρακτήρες, χρησιμοποιείται ο αντίστοιχος δεκαεξαδικός αριθμός. Επίσης για να μην υπάρχει πρόβλημα με το http πρωτόκολλο υπάρχουν και άλλοι δεσμευμένοι χαρακτήρες, οι οποίοι είναι διαχωριστικά του HTTP/1.x. Δηλαδή:

- **« » (κενό-space):** Αντικαθίσταται με τον χαρακτήρα «+» (είναι διαχωριστικό στο πρωτόκολλο http)..
- **«+» (συν-plus):** Αντικαθίσταται με τον αντίστοιχο δεκαεξαδικό (όταν δεν παριστάνει κενό).
- **«\n» (νέα γραμμή-return):** Αντικαθίσταται με τον αντίστοιχο δεκαεξαδικό (είναι διαχωριστικό στο πρωτόκολλο http).
- **«\t» (νέα στήλη-tab):** Αντικαθίσταται με τον αντίστοιχο δεκαεξαδικό (είναι διαχωριστικό στο πρωτόκολλο http).
- **«/» (κάθετος-slash):** Αντικαθίσταται με τον αντίστοιχο δεκαεξαδικό (είναι διαχωριστικό στο URL).
- **«?» (ερωτηματικό-question mark):** Αντικαθίσταται με τον αντίστοιχο δεκαεξαδικό (διαχωρίζει στο URL το όνομα του πόρου από τις παραμέτρους).
- **«#» (δίεση-**): Αντικαθίσταται με τον αντίστοιχο δεκαεξαδικό (διαχωρίζει στο URL την αναφορά σε κάποιο σημείο του κειμένου).
- **«=» (ίσον-equal):** Αντικαθίσταται με τον αντίστοιχο δεκαεξαδικό (διαχωρίζει στο URL τα ονόματα των παραμέτρων από τις τιμές).

- **«&» (και-)**: Αντικαθίσταται με τον αντίστοιχο δεκαεξαδικό (διαχωρίζει στο URL τις παραμέτρους).

# *Παράρτημα ΙΙ*

## ΙΙ. Το πρωτόκολλο HTTP (HyperText Transfer Protocol)

Για την αναφορά των θέσεων των πόρων του διαδικτύου, χρησιμοποιείται η μέθοδος ονομασίας URL (Uniform Resource Locator). Όταν το πρωτόκολλο που ορίζεται σε ένα URL είναι το http, τότε μιλάμε για πόρους στους οποίους μπορούμε να έχουμε πρόσβαση μόνο με το HTTP πρωτόκολλο.

Το HTTP πρωτόκολλο, αν και πλέον χρησιμοποιείται για την μεταφορά οποιουδήποτε τύπου αρχείου από τον εξυπηρετητή στον χρήστη, έχει σχεδιαστεί ειδικά για την μεταφορά HTML (HyperText Markup Language) αρχείων.

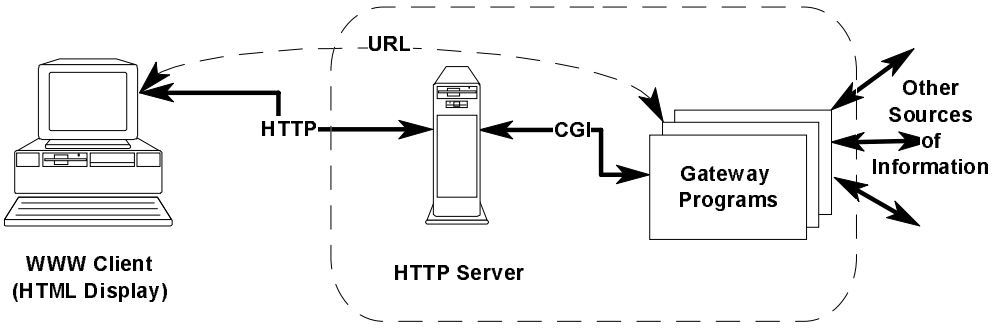
Εδώ να επισημάνουμε ότι η HTML γλώσσα:

- Περιλαμβάνει μεθόδους για την εισαγωγή στοιχείων-δεδομένων από τον χρήστη και την μεταφορά τους στον εξυπηρετητή.

- Δείχνει τον τρόπο εμφάνισης ενός κειμένου στον web browser.

Επίσης, με το HTTP πρωτόκολλο, δίνεται πρόσβαση σε CGI (Common Gateway Interface) προγράμματα. Τα CGI προγράμματα:

Είναι συνήθως προγράμματα που χρησιμοποιούνται για την δυναμική παραγωγή HTML σελίδων μετά από επικοινωνία με άλλες πηγές πληροφοριών (όπως βάσεις δεδομένων). Έτσι μπορεί να επιτευχθεί επικοινωνία του χρήστη με κάποια βάση δεδομένων μέσω web interface. Στο παρακάτω σχήμα (Σχήμα 29) φαίνεται το σενάριο επικοινωνίας του χρήστη με κάποια πηγή πληροφοριών, με τον HTTP πρωτόκολλο χρησιμοποιώντας CGI προγράμματα.




Σχήμα 29. Σενάριο επικοινωνίας χρήστη με CGI πρόγραμμα

## II.i. Μέθοδος επικοινωνίας με το HTTP πρωτόκολλο

Η επικοινωνία με έναν εξυπηρετητή HTTP πρωτοκόλλου γίνεται χρησιμοποιώντας ένα socket stream κάτω από TCP/IP. Στην πρώτη φάση της επικοινωνίας, ο πελάτης (client) στέλνει την αίτηση προς τον εξυπηρετητή. Η αίτηση αυτή περιέχει:

- Τον τύπο της αίτησης, ο οποίος μπορεί να είναι GET, HEAD ή POST.
- Το όνομα του πόρου που ζητάει.
- Την ακριβή έκδοση του πρωτοκόλλου επικοινωνίας που υποστηρίζει (HTTP/1.0 ή HTTP/1.1)
- Κάποιες πληροφορίες για τον τύπο του (του πελάτη).
- Και τέλος τα ορίσματα-δεδομένα που θα δοθούν σαν είσοδος στον πόρο στον οποίο ζητά πρόσβαση.

Αφού τελειώσει η αποστολή της αίτησης, τότε αναμένει την απάντηση του εξυπηρετητή από το ίδιο stream. Η απάντηση περιέχει:

- Τον κωδικό αποτελέσματος εξυπηρέτησης της αίτησης (άν πέτυχε ή όχι, ή αν ο πόρος έχει μεταφερθεί κάπου αλλού, κ.τ.λ.)
- ένα κομμάτι πληροφοριών για τα δεδομένα που ακολουθούν (τύπος δεδομένων/MIME-type, μέγεθος, κ.τ.λ.)

- Τα ίδια τα δεδομένα

Η διαδικασία αυτή τελειώνει όταν κάποιος από τους δύο κλείσει την επικοινωνία. Αν η όλη διαδικασία είναι επιτυχής, την επικοινωνία θα την κλείσει σίγουρα πρώτος ο εξυπηρετητής, όταν τελειώσει την αποστολή των δεδομένων.

# *Παράρτημα ΙΙΙ*

## III. Ινστιτούτα που συμμετέχουν στην συλλογή NCSTRL-ERCIM

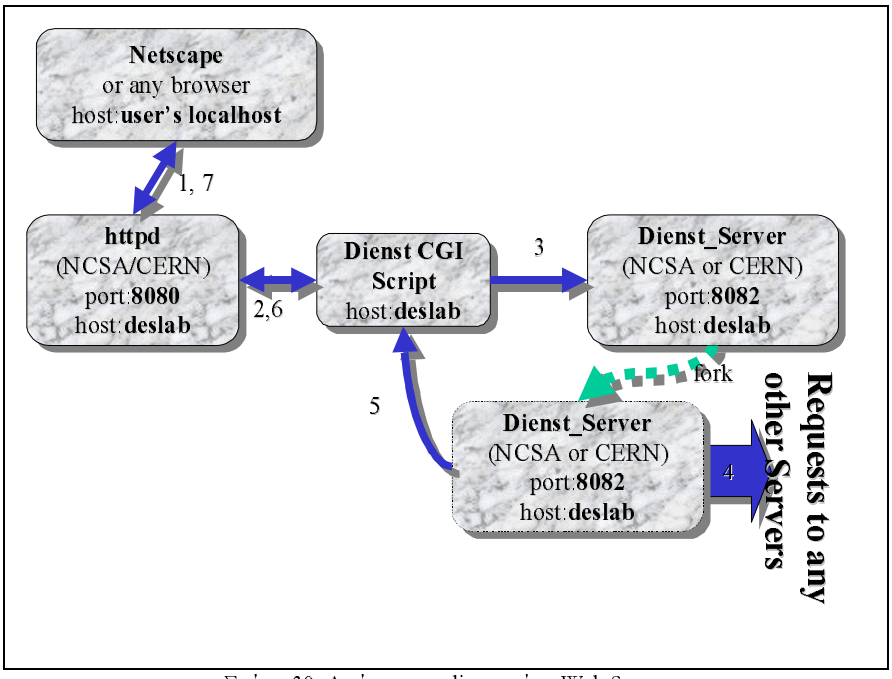
1. Auburn University
2. Boston University
3. Brown University - Department of Computer Science
4. California Institute of Technology
5. University of Chicago
6. Chicago Journal of Theoretical Computer Science
7. Carnegie Mellon University
8. Cornell University, Computer Science
9. Chinese University of Hong Kong
10. Erasmus University, Rotterdam
11. George Mason University
12. Iowa State University
13. Johns Hopkins University
14. McGill Centre for Intelligent Machines
15. MIT-AILab
16. MIT-LCS
17. North Carolina State University
18. Old Dominion University
19. Oregon State University
20. Princeton University
21. Stanford University
22. SUNY at Buffalo
23. TUB-FB13
24. University of California, Berkeley
25. University of California, Irvine
26. University of California, San Diego
27. UMCP-CSD
28. Universite de Montreal
29. University of North Carolina, Chapel Hill
30. University of Toronto
31. UT-HARBURG
32. University of Texas, Austin
33. University of Virginia
34. Xerox, Design Research Institute
35. D-Lib Magazine
36. DIMACS
37. CNR - Area della Ricerca (Palermo)
38. CNR - Istituto CNUCE (Pisa)
39. CNR - Istituto per le Applicazioni della Metamatica e dell'Informatica (Milano)
40. CNR - Istituto di Analisi Numerica (Pavia)
41. CNR - Istituto di Cibernetica (Napoli)
42. CNR - Istituto di Elaborazione della Informazione (Pisa)
43. CNR - Istituto di Fisica Cosmica con Applicazioni all'Informatica (Palermo)
44. CNR - Istituto di Linguistica Computazionale (Pisa)
45. CNR - Instituto per la Matematica Applicata (Genova)
46. CNR - Istituto di Matematica Computazionale (Pisa)
47. CNR - Istituto di Studi sulla Ricerca e sulla Documentazione Scientifica (Roma)
48. CNR - Istituto di Tecnologie Didattiche e Formative (Palermo)
49. CWI - Centrum voor Wiskunde en Informatica
50. Foundation for Research and Technology - Hellas. Institute of Computer Science
51. GMD - German National Research Center for Information Technology
52. SICS - Swedish Institute of Computer Science
53. SZTAKI, The Computer and Automation Research Institute
54. State University of New York at Albany
55. University of Arizona, Department of Computer Science
56. University of Bristol, UK. Computer Science
57. Oxford Brookes University

58. CaberNet Technical Report and Abstracts Service
59. Progetto Mediterraneo, CNR-Pisa
60. Colorado State University
61. Cornell University, Theory Center
62. University of Colorado, Denver - Center for Computational Mathematics
63. Dartmouth College
64. Duke University
65. Swiss Federal Institute of Technology, Lausanne
66. Swiss Federal Institute of Technology, Zurich
67. Universidade de Lisboa
68. Georgia Institute of Technology. College of Computing
69. Georgia Institute of Technology. Graphics, Visualization and Usability Center
70. George Mason University, Computer Science
71. ICASE
72. Albert-Ludwigs-Universitaet Freiburg, Institut fuer Informatik
73. University of Ljubljana, Jozef Stefan Institute
74. Jet Propulsion Laboratory, California Institute of Technology
75. Mississippi State University
76. Miami University, Oxford, Ohio. Systems Analysis.
77. National Centre for Scientific Research Demokritos
78. North Dakota State University. Computer Science and Operations Research
79. Navy Center for Applied Research in Artificial Intelligence
80. New York University
81. Oregon Graduate Institute of Science and Technology
82. Odense University, Denmark
83. Universidad Catolica de Chile. Departmento de Ciencia de la Computacion.
84. Queensland University of Technology. Faculty of Information Technology
85. Rice University
86. University of Rochester. Computer Science.
87. Texas A&M University
88. University of Dublin, Trinity College. Computer Science Department.
89. Technical University of Chemnitz-Zwickau
90. Dresden University of Technology
91. Technical University of Munich
92. Turku Centre for Computer Science, Finland
93. University of British Columbia. Computer Science
94. University of Karlsruhe, Germany, Computer Science
95. University of Central Florida
96. University of California, Los Angeles
97. University of California, Santa Cruz, Jack Baskin School of Engineering
98. University of Hamburg, Germany
99. University of Illinois at Urbana-Champaign
100. University of Kentucky
101. Universitaet Leipzig. Institute fuer Informatik
102. Johannes Kepler University Linz, Austria
103. University of Magdeburg. Computer Science.
104. University of Mannhein, Germany
105. University of Massachusetts, Amherst
106. University of Michigan Department of Electrical Engineering and Computer Science
107. University of Bonn, Department of Computer Science
108. University of Piraeus, Greece. Computer Science
109. Pisa University, Italy
110. University of Zurich, Switzerland
111. National and Capodistrian University of Athens. Department of Informatics.
112. University of Saskatchewan
113. Universitaet Stuttgart, Germany. Fakultaet Informatik
114. University of Utah, Department of Computer Science
115. University of Tennessee, Knoxville
116. University of Western Australia. Department of Electrical and Electronic Engineering.
117. University of Washington
118. University of Wisconsin, Madison
119. Virginia Polytechnic Inst. and State University
120. University of Warwick
121. Weizmann Institute of Science, Faculty of Mathematical Sciences

# *Παράρτημα IV*

## IV. Λειτουργία του συστήματος dienst

Σε αυτό το παράρτημα θα δούμε το σενάριο συνεργασίας του dienst με τον WWW server και πώς αυτός δέχεται εξωτερικές αιτήσεις (από χρήστες ή από άλλους dienst servers). Στο Σχήμα 30 φαίνεται η ροή των μηνυμάτων για μια αίτηση στο dienst.




Σχήμα 30. Αιτήσεις στο dienst μέσω Web Server

- Netscape process(User)➔httpd process (HTTP): Αρχικά έχουμε την αίτηση του εξωτερικού παράγοντα (π.χ. χρήστη) προς τον web server.

- httpd process ➔dienst script (CGI-BIN): Ο web server αφού αναγνωρίσει ότι η αίτηση απευθύνεται στο dienst, εκτελεί ένα ειδικό πρόγραμμα (CGI-script) το οποίο θα προωθήσει την αίτηση στον dienst server.

- dienst script➔dienst server (dienst protocol): Το CGI-script προωθεί την αίτηση στον dienst server.

- fork & service: Ο dienst server κάνει ένα αντίγραφο του εαυτού του, το οποίο είναι υπεύθυνο να εξυπηρετήσει την συγκεκριμένη αίτηση.

- dienst server (child)➔ dienst script ➔ httpd process: Αφού επεξεργαστεί η αίτηση και είναι έτοιμα τα αποτελέσματα, η απάντηση στέλνεται πίσω στον χρήστη, ακολουθώντας το ίδιο μονοπάτι με το οποίο προωθήθηκε, από τον dienst server (το αντίγραφο), στο CGI script, από εκεί στον web server και πίσω στον Browser του χρήστη.

# *Παράρτημα V*

## V. Dienst User Interface και κάποιες προτάσεις για βελτιώσεις

Σε αυτό το κεφάλαιο θα ασχοληθούμε κυρίως με το User Interface του dienst. Θα δούμε το interface που χρησιμοποιείται στην τρέχουσα έκδοση (v4.1.9), στην επόμενη που θα βγει (v5.0.0), και θα κάνουμε μια πρόταση για το πώς μπορεί να βελτιωθεί αυτό.

Είναι προφανές ότι το User Interface, εξαρτάται και από την λειτουργικότητα του μηχανισμού του προγράμματος το οποίο το χρησιμοποιεί. Εμείς σε αυτή την εργασία θα θεωρήσουμε αρχικά την λειτουργικότητα δεδομένη, και θα προτείνουμε βελτιώσεις δεδομένης αυτής της λειτουργικότητας. Αλλά θα προτείνουμε και κάποιες επιπλέον λειτουργίες που ενδεχομένως θα έπρεπε να έχει το dienst, οι οποίες χρειάζονται ώστε να γίνει το user interface περισσότερο φιλικό και ο μηχανισμός αναζήτησης και περιήγησης του χρήστη περισσότερο αποτελεσματικός.

### V.i. Dienst: User Interface

Το dienst, είναι σχεδιασμένο ειδικά για να λειτουργεί κάτω από το πρωτόκολλο του WWW, το HTTP. Άρα είναι προφανές ότι και η διεπιφάνεια χρήσης είναι σχεδιασμένη έτσι ώστε να συμβαδίζει με τις απαιτήσεις του Web. Επίσης πρέπει να έχουμε υπόψη μας ότι, τουλάχιστον αυτή τη στιγμή, το dienst χρησιμοποιείται μόνο για την συλλογή τεχνικών αναφορών επιστήμης υπολογιστών. Άρα απευθύνεται σε χρήστες με εμπειρία χρήσης υπολογιστών. Συνεπώς το ζητούμενο από την διεπιφάνεια χρήσης δεν είναι fancy user interface με τις πολλές εικόνες και τα πολλά βήματα για να κάνεις κάτι, αλλά να δίνει την δυνατότητα να μπορεί ο χρήστης να κάνει εύκολα και γρήγορα αυτό που θέλει χωρίς να χρειάζεται να ακολουθεί πολλά απλά βήματα αλλά λίγα, και να έχει όσο το δυνατό περισσότερες δυνατότητες στο κάθε βήμα. Επίσης συνήθως υπάρχει η απαίτηση σε μια "οθόνη" να μπορούμε να δούμε όσο γίνεται περισσότερη πληροφορία.

### V.i.i. Task Analysis

Πριν προτείνουμε οποιαδήποτε αλλαγή στο user interface του dienst, είναι σκόπιμο να δούμε την σχεδίαση του user interface της παρούσας έκδοσης και της επόμενης που είναι έτοιμη να παρουσιαστεί. 'ν και το σύστημα dienst έχει σχεδιαστεί εδώ και πολύ καιρό, δεν υπάρχει κάποια μελέτη από την σκοπιά του user interface. Η έμφαση η οποία δίνεται μέχρι στιγμής είναι ο σχεδιασμός της αρχιτεκτονικής και των κατανεμημένων μηχανισμών λειτουργίας του συστήματος. Παρόλα αυτά η παρουσίαση του ήδη υπάρχων task analysis και dialog design θα γίνει περιληπτικά.

#### Λειτουργίες

Οι λειτουργίες οι οποίες υποστηρίζονται από την παρούσα έκδοση είναι οι παρακάτω:

- **Αναζήτηση:** Η κύρια λειτουργία που υποστηρίζεται από το σύστημα είναι αυτή της αναζήτησης εγγράφων. Αυτή χωρίζεται σε 2 τμήματα:
  - Απλή Αναζήτηση, στην οποία ο χρήστης δίνει λέξεις κλειδιά για τις οποίες ενδιαφέρεται, χωρίς όμως να μπορεί να ορίσει κάτι άλλο.
  - Αναζήτηση με βάση πεδία. Σε αυτού του τύπου την αναζήτηση ο χρήστης ορίζει λέξεις κλειδιά με βάση ένα από τα παρακάτω δυνατά πεδία: Συγγραφέας, Τίτλος, Περίληψη, κωδικός εγγράφου. Στην ίδια σελίδα ο χρήστης μπορεί να επιλέξει τους κόμβους της συλλογής στους οποίους θα σταλεί η ερώτηση.
- **Περιήγηση**
  - Περιήγηση με βάση Συγγραφέα: Δεδομένου ενός κόμβου του συστήματος, ο χρήστης μπορεί να επιλέξει να δει τα έγγραφα που περιέχει αυτός ο κόμβος ταξινομημένα με βάση το όνομα του συγγραφέα. Ακόμη μπορεί να επιλέξει μόνο τους συγγραφείς που το όνομά τους αρχίζει από ένα γράμμα, ή από κάποιο προκαθορισμένο σύνολο γραμμάτων.
  - Περιήγηση με βάση το έτος: Ομοίως, δεδομένου ενός κόμβου του συστήματος, ο χρήστης μπορεί να επιλέξει να δει τα έγγραφα που περιέχει αυτός ο κόμβος ταξινομημένα με βάση το έτος δημοσίευσης τους. Ακόμη μπορεί να επιλέξει μόνο ένα έτος, ή ένα σύνολο από έτη των 10.
- **Παρουσίαση εγγράφου:** Εδώ παρουσιάζονται κάποιες περιληπτικές πληροφορίες για το έγγραφο: τίτλος, συγγραφέα, ημερομηνία, περίληψη, οι μορφές στις οποίες αυτό είναι διαθέσιμο και σύνδεσμοι προς αυτά.

Στην έκδοση 5.0.0 η οποία είναι στο στάδιο πριν την τελική έκδοση, στα παραπάνω έχει προστεθεί:

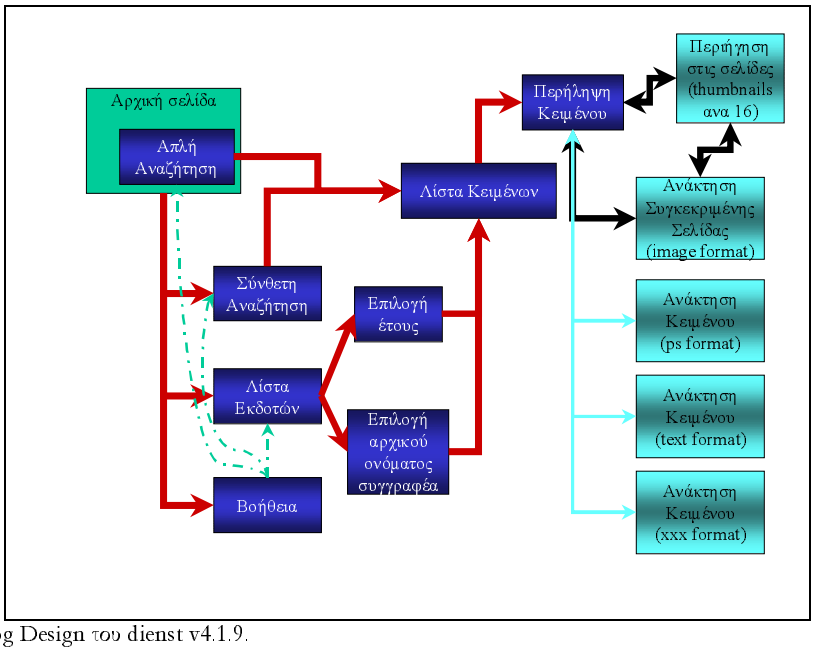
### 1. User Profiles

Ορισμός: Ο χρήστης που χρησιμοποιεί για πρώτη φορά την επιλογή αυτή ζητείται να εισάγει τα στοιχεία του (κυρίως το e-mail address, userid και password)

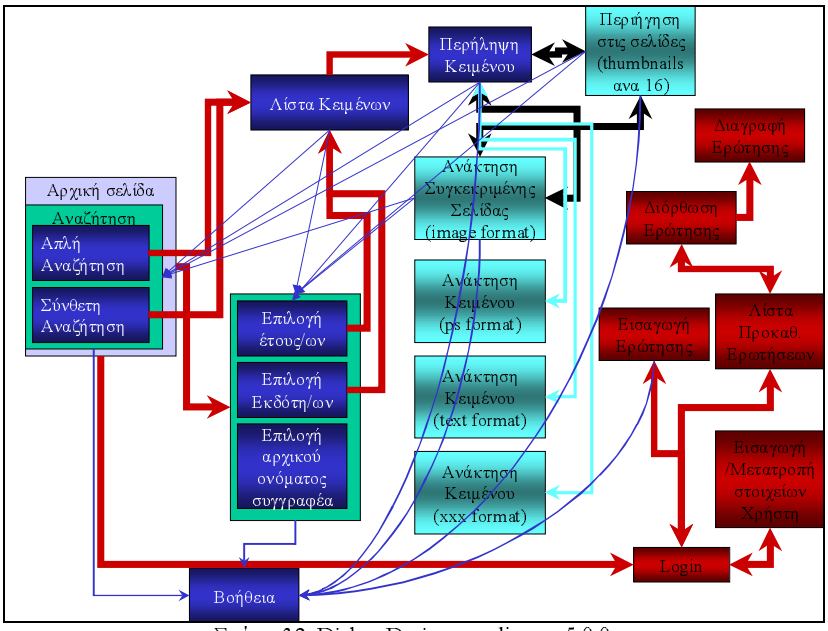
Μετατροπή: Ο χρήστης μπορεί να μεταβάλει οποιοδήποτε από τα στοιχεία που έχει ορίσει για τον εαυτό του.

- **Προκαθορισμένες Ερωτήσεις Χρήστη**
  - Ορισμός: Ο ορισμός της ερώτησης, γίνεται ακριβώς με τον ίδιο τρόπο που εισάγει ο χρήστης ερωτήσεις (queries), μόνο που σε αυτήν την περίπτωση, η ερώτηση δεν εκτελείται αμέσως. Τα στοιχεία κρατούνται εσωτερικά στο σύστημα, και η ερώτηση εκτελείται κάθε μέρα ή κάθε βδομάδα ή κάθε μήνα (ανάλογα με το τι έχει ορίσει ο χρήστης) και τα αποτελέσματα αποστέλλονται με e-mail στην διεύθυνση που έχει ορίσει.
  - Λίστα προκαθορισμένων Ερωτήσεων: Ο χρήστης μπορεί να δει ποιες ερωτήσεις έχει ορίσει, και να επιλέξει να διαγράψει κάποια ή να την μετατρέψει.
  - Μετατροπή: Εδώ ο χρήστης οδηγείται από την προηγούμενη λίστα για να αλλάξει κάποια στοιχεία μιας προκαθορισμένης ερώτησης.

## V.i.ii. Dialog Design




Σχήμα 31. Dialog Design του dienst v4.1.9.



Σχήμα 32. Dialog Design του dienst v5.0.0.

## V.ii. UI

### V.ii.i. Αρχική σελίδα

Αρχική σελίδα. Εδώ ο χρήστης έχει την επιλογή να ακολουθήσει διάφορα links ή να δώσει απευθείας μια απλή ερώτηση.

### V.ii.ii. Φόρμα σύνθετης αναζήτησης

Φόρμα σύνθετης αναζήτησης. Αν και σύνθετη η αναζήτηση, είναι αρκετά απλή. Ο χρήστης δεν μπορεί να ορίσει την ερώτηση χρησιμοποιώντας κανονικές εκφράσεις ούτε να δώσει προτεραιότητα εκτέλεσης των συνθηκών (χρησιμοποιώντας παρενθέσεις). Η σύνταξη μέσα στα πεδία είναι της μορφής (Πίνακας 17):

```
Query-author: expresion|null
expresion: term operator expresion|term
term: keyword
operator: and|or
```

Πίνακας 17. Γραμματική εισαγωγής επερωτήσεων στο dienst

Ομοίως και για τα υπόλοιπα πεδία.

Επίσης δεν μπορεί να ορίσει πολλαπλά and ή or ανάμεσα στα πεδία. Αν από το click-box, επιλέξει "and" τότε η ερώτηση θα είναι: Query-author and Query-title and Query-abstract, ενώ αν επιλέξει "or" τότε: Query-author or Query-title or Query-abstract.

Ο χρήστης μπορεί επίσης να επιλέξει τους κόμβους στους οποίους θα απευθυνθεί η ερώτηση.

Στην έκδοση 5.0.0, παρατηρούμε μια συμπίεση της οθόνης. Η απλή αναζήτηση και η σύνθετη έχουν μπει σε μία οθόνη. Επίσης έχει αφαιρεθεί η δυνατότητα επιλογής κόμβων. Έχει προστεθεί όμως η επιλογή της σειράς με την οποία θα εμφανιστούν τα αποτελέσματα.

### V.ii.iii. Σελίδα αποτελεσμάτων αναζήτησης

Σελίδα αποτελεσμάτων αναζήτησης: Στην σελίδα αποτελεσμάτων, τα links στα κείμενα είναι οργανωμένα σύμφωνα με τον εκδότη (κόμβο) στον οποίο ανήκουν τα κείμενα. Επίσης εμφανίζονται περιληπτικά τα στοιχεία της ερώτησης του χρήστη και κάποια στοιχεία για τυχόν λάθη που έχουν γίνει (πχ. timeout κόμβου ή άλλα λάθη).

Στην έκδοση 5.0.0 εμφανίζονται τα ίδια στοιχεία, αλλά επιπλέον:

1. links στα ονόματα των συγγραφέων για εκτέλεση ερώτησης με βάση αυτά τα ονόματα
2. Τα κείμενα μπορεί να είναι ταξινομημένα με βάση τον εκδότη, το έτος ή τη βαθμολογία σχετικότητας με την ερώτηση.

### V.ii.iv. Viewing Documents - Περίληψη Κειμένου

Εδώ εμφανίζονται οι περιληπτικές πληροφορίες για το κείμενο. Βλέποντας μια πολύ μικρή αλλαγή ανάμεσα στις 2 εκδόσεις, βλέπουμε ότι μια μικρή αλλαγή έχει βελτιώσει κατά πολύ την διεπιφάνεια χρήσης. Στην πρώτη περίπτωση ο χρήστης εφόσον επιλέξει ένα format, χάνει από

την οθόνη τις υπόλοιπες επιλογές, ενώ στην επόμενη έκδοση, με την πρόσθεση ενός frame, εξακολουθεί να έχει την επιλογή να δει απευθείας ένα άλλο format, χωρίς να χρειάζεται να επιστρέψει στην προηγούμενη σελίδα.

## V.iii. Προτάσεις για περαιτέρω βελτίωση της διεπιφάνειας χρήσης

### V.iii.i. Search

Εκτός από τις λειτουργίες που προσφέρει αυτή τη στιγμή η επιλογή για αναζήτηση, προτείνουμε τις εξής διαφοροποιήσεις:

- Αρχικά, επαναφορά της δυνατότητας επιλογής από τον χρήστη των κόμβων στους οποίους απευθύνεται η ερώτηση. Βέβαια, αυτή η επιλογή δεν ήταν αρκετά βολική να την χρησιμοποιήσει ο χρήστης, διότι εμφανίζονταν μια "ξερή" λίστα από εκδότες και ο χρήστης έπρεπε να ξέρει τι ακριβώς ζητάει. Σε αυτήν την περίπτωση προτείνουμε μια ιεραρχική δόμηση της λίστας των εκδοτών σύμφωνα (αρχικά) με τον οργανισμό στον οποίο ανήκουν, χώρα, πανεπιστήμιο, τμήμα κ.τ.λ.. Αντίστοιχη υλοποίηση έχει γίνει στο CSI, στην οποία η ιεραρχική δομή προκύπτει από το publisherID το οποίο είναι της μορφής: ercim.forth.ics, ncstrl.cornell.cs e.t.c. Επίσης χρήση και βελτίωση αυτής της δόμησης έχει γίνει στην έκδοση που έχει χρησιμοποιηθεί από το Design Lab - MIT (asidirop) για την συλλογή εγγράφων CAD/CAM/CAE. Εδώ η ιδέα είναι ότι έχουμε μια ξεχωριστή συλλογή από έγγραφα μη επιστήμης Υπολογιστών. Όμως και στην συλλογή NCSTRL, υπάρχουν σχετικές δημοσιεύσεις. 'ρα θέλουμε να δώσουμε την δυνατότητα στον χρήστη να μπορεί να επιλέξει μια άλλη συλλογή, ή ένα άλλο σύνολο από εκδότες, χωρίς να τον μπερδέψουμε για το ποιος ανήκει πού.
  Ενδεχομένως μπορούν να προστεθούν και άλλα κριτήρια ομαδοποίησης, π.χ. ίσως ακόμη και η γεωγραφική τοποθεσία του εκδότη, και ο χρήστης να επιλέγει "on the fly" την μορφή με την οποία του εμφανίζεται το δέντρο.
- Πρέπει να προστεθεί η δυνατότητα του τελεστή "not" για κάποια λέξη κλειδί.
- Επίσης πρέπει ο χρήστης να έχει την δυνατότητα να μπορεί να δώσει περισσότερο πολύπλοκες ερωτήσεις. Κάτι τέτοιο το πρωτόκολλο επικοινωνίας του dienst δεν το υποστηρίζει. Ακόμη και στην έκδοση 5.0.0 που ο μηχανισμός αναζήτησης έχει αντικατασταθεί με τον WAIS (ο οποίος υποστηρίζει κάτι τέτοιο), για λόγους συμβατότητας με την παλαιότερη έκδοση το πρωτόκολλο επικοινωνίας έχει παραμείνει το ίδιο και άρα, παρότι ο μηχανισμός μας δίνει μια τέτοια δυνατότητα, την χάνουμε.

- Δυνατότητα επιλογής της γλώσσας στην οποία είναι τα αποτελέσματα. Και εδώ υπάρχει ένα μικρό πρόβλημα στην υλοποίηση κάτι τέτοιου, διότι για ελάχιστα κείμενα υπάρχει η πληροφορία για την γλώσσα στην οποία είναι γραμμένα, αν και το format στο οποίο αποθηκεύονται οι βιβλιογραφικές πληροφορίες περιέχει αντίστοιχο πεδίο (LANGUAGE), κανείς από τους εκδότες δεν το συμπληρώνει στα έγγραφά του.
- Δυνατότητα επιλογής του πλήθους των search hits που θα υπάρχουν σε κάθε σελίδα αποτελεσμάτων.

### V.iii.ii. Search Results

Στην έκδοση 4.1.9, στην σελίδα αποτελεσμάτων αναζήτησης, τα αποτελέσματα είναι ταξινομημένα σύμφωνα με τον εκδότη στον οποίο ανήκουν. Στην αρχή της σελίδας υπάρχουν links προς τις αντίστοιχες "ομάδες" παρακάτω στην ίδια σελίδα. Επίσης οι πληροφορίες για το κάθε κείμενο είναι στοιχεία μιας λίστας HTML (UL)

Στην έκδοση 5.0.0 έχει επιλέξει από πριν ο χρήστης τον τρόπο ταξινόμησης. Οι πληροφορίες για κάθε κείμενο είναι στοιχεία ενός cell πίνακα (με διαφορετικό χρώμα). Έτσι διακρίνονται καλύτερα από την προηγούμενη έκδοση, αλλά αυτό έχει σαν συνέπεια την δημιουργία πολύ μακριών σελίδων. 'ρα ο χρήστης πρέπει να κάνει αρκετές φορές scroll, για να δει τα αποτελέσματα.

Για αυτήν την περίπτωση σαν βελτιώσεις προτείνουμε:

1. Περισσότερο συμπιεσμένο layout. Δηλαδή, ο χρήστης αρχικά μπορεί να βλέπει μόνο τους τίτλους από τα κείμενα. Αλλά να έχει την δυνατότητα, στην ίδια σελίδα να ορίσει ποια άλλα πεδία θέλει να βλέπει.
2. Αν ο χρήστης μετακινήσει το ποντίκι πάνω από ένα link, να του εμφανίζονται όλες οι πληροφορίες για το κείμενο.
3. Στην ίδια σελίδα ο χρήστης πρέπει να μπορεί να αλλάξει τα keywords για την αναζήτησή του, χωρίς να χρειάζεται να κάνει "back", στην περίπτωση που διαπιστώσει ότι αυτά δεν είναι αρκετά.
4. Ακριβώς κάτω από τα στοιχεία της αναζήτησης, μπορεί να υπάρχει μια λίστα με τα 10 περισσότερο σχετικές λέξεις κλειδιά, τις οποίες θα μπορεί να προσθέσει (+), να αφαιρέσει (-) ή να αγνοήσει (.) στην αναζήτηση.
5. Επιλογή του πλήθους των search hits που εμφανίζονται στην σελίδα.
6. Links με buttons προς την επόμενη σελίδα αποτελεσμάτων ή την προηγούμενη.
7. Επιλογή του τρόπου ταξινόμησης των αποτελεσμάτων.

### V.iii.iii. Browsing the Collection

Και αυτό το τμήμα έχει αλλάξει πολύ από την έκδοση 4.1.9 στην 5.0.0. Συμπιέζοντας τις φόρμες, έχει γίνει περισσότερο βολικό για τον χρήστη. Παρόλα αυτά έχουμε να προτείνουμε κάποιες διαφοροποιήσεις:

1. Δεν υπάρχει νόημα να κάνει κάποιος browse σε όλους τους εκδότες, αλλά σε έναν, ή σε ένα σύνολο που έχουν κάποια κοινά χαρακτηριστικά. Έτσι προτείνουμε να αφαιρεθεί η by default επιλογή όλων των εκδοτών. Άλλωστε αυτό παραπλανεί τον χρήστη, διότι είναι μια λειτουργία η οποία 1) δεν είναι πλήρως υλοποιημένη και 2) Χρειάζεται πάρα πολύ χρόνο για να εκτελεστεί. Θα μπορούσε να είναι "by default" επιλεγμένος ο εκδότης στον οποίο ανήκει ο κόμβος που έχει επισκεφθεί ο χρήστης.
2. Πρόσθεση της δυνατότητας να μπορεί ο χρήστης να επιλέξει οποιοδήποτε σύνολο από έτη ή αρχικά ονομάτων (π.χ. "1991,1994-1996").
3. Αύξηση της ακρίβειας στην χρονολογία με την προσθήκη (προαιρετικά) και του μήνα (π.χ. Feb 1998 - now)

### V.iii.iv. Viewing documents

Αυτή η λειτουργία είναι αρκετά ιδιαίτερη διότι εκτός από το ότι το κάθε κείμενο συνήθως δεν είναι διαθέσιμο σε όλα τα προκαθορισμένα formats, επιπλέον και ο κάθε εκδότης μπορεί να ορίσει επιπλέον formats στα οποία είναι διαθέσιμα τα δικά του κείμενα. Παρόλα αυτά όμως μπορούμε να θεωρήσουμε την ύπαρξη του postscript format βασική.

# *Παράρτημα VI*

## VI. Το πρόγραμμα «dienstadmin»

### VI.i. Περιγραφή

Από τις pleiades, μπορούν οι χρήστες που ανήκουν στην ομάδα του dienst, να διαχειριστούν τους dienst servers που τρέχουν στο ICS. Για να μπορεί να γίνει αυτό είναι προφανές, ότι πρέπει να υπάρχουν (στο script) κάποιες πληροφορίες σχετικά με το πού τρέχει ο server, την version κτλ (αυτό το αναλλαμβάνει ο administrator του dienst).

Το πρόγραμμα καλείται με:

```
/usr/admin/bin/super dienstadmin "args"
```

στο prompt του unix στις pleiades.

Το πρόγραμμα είναι δυνατόν να εκτελεστεί και παρακάμπτοντας την χρήση του super (/usr/lbin/dienst-admin), από όλους τους χρήστες, αλλά τότε δεν έχει την δυνατότητα να κάνει startup ή shutdown κανέναν server παρά μόνο να δώσει πληροφορίες.

### VI.ii. Ορίσματα

Αν δεν δοθεί κανένα όρισμα στο πρόγραμμα τότε τυπώνεται η σελίδα βοήθειας.

- -i
  Τυπώνονται πληροφορίες για τους server που μπορεί να διαχειριστεί το πρόγραμμα.
  Πεδία:
  Server Name: Το όνομα του server. Αν μέσα από το πρόγραμμα (χρησιμοποιώντας τα υπόλοιπα ορίσματα) θέλουμε να αναφερθούμε σε αυτόν τον server, πρέπει να αναφερθούμε σε αυτό το όνομα.
  Server Info: Κάποιες πληροφορίες για την έκδοση του server.
  Home Page: Link στο home page του server.

Dienst Server Args: Πληροφορίες εσωτερικής χρήσης.
Http Server Args: Πληροφορίες εσωτερικής χρήσης.
Server Ports: Πληροφορίες για το ποια ports και σε ποια μηχανήματα χρησιμοποιεί ο server.

- -s "server"
  Γίνεται ShutDown ο server που ορίζεται. Σε κάθε τρέξιμο του προγράμματος, μπορεί να οριστεί μόνο ένας server για shutdown.

- -c "server" ή -c all
  Τυπώνονται πληροφορίες για τις processes των servers (πχ. άν είναι running).

- -f "server" ή -f all
  Τυπώνονται πληροφορίες για τις processes των servers (πχ. άν είναι running) και οι πληροφορίες που θα τυπώνονταν με το όρισμα -i.

- -v "server" ή -v all
  Τυπώνονται πληροφορίες για τις processes των servers (πχ. αν είναι running) και άν ΔΕΝ είναι running ξεκινούν.

- -p
  Τυπώνονται πληροφορίες για τα ports των dienst-http servers.

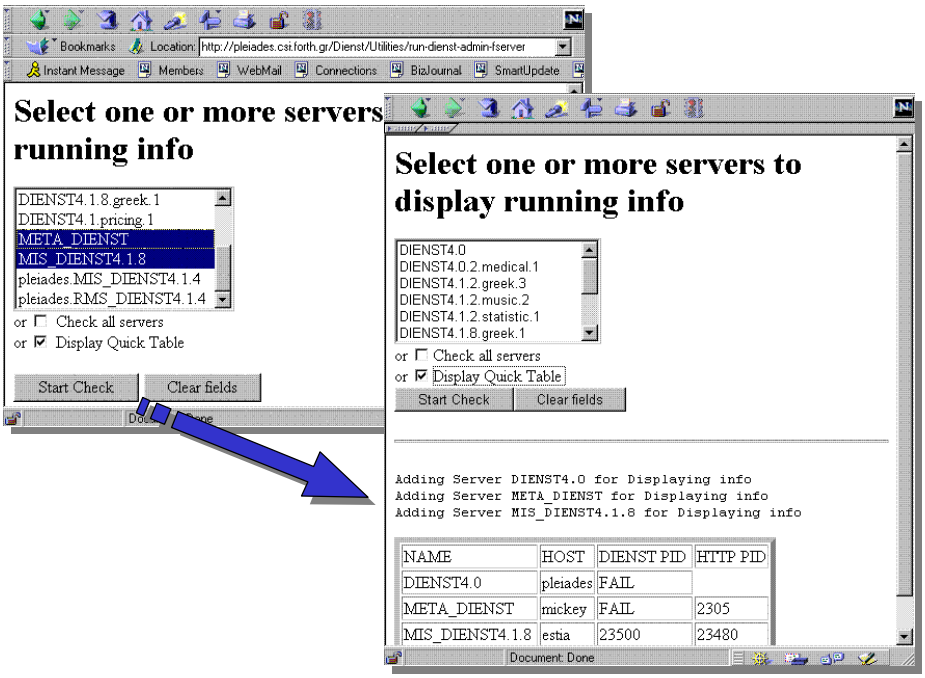
- -quick
  Τυπώνονται σύντομες πληροφορίες για τους servers (που έχουν προστεθεί στο -f ή -c πεδίο) σε μορφή πίνακα. Τα πεδία που υπάρχουν στον πίνακα είναι: NAME, HOST, dienst process id, http process id. Χρησιμεύει στο να πάρουμε μια γρήγορη εκτίμηση για το ποιοι servers είναι active και ποιοι όχι.

- -html
  Σε συνδυασμό με το -quick option, τυπώνει τον πίνακα (που προκύπτει από το -quick) σε html μορφή.

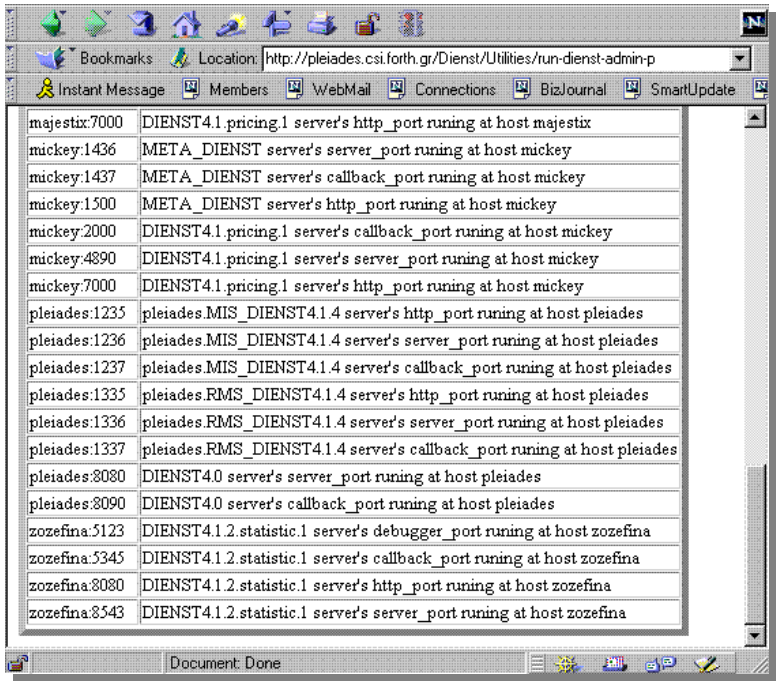
## VI.iii. Web Interface

Για το ίδιο πρόγραμμα, έχει υλοποιηθεί και ένα interface μέσω web, το οποίο μας βοηθάει να έχουμε μια συνοπτική εικόνα για τους servers που υπάρχουν, ποιοι είναι ενεργοί κτλ.. Σε αυτό μπορούμε να έχουμε πρόσβαση στο *http://www.csi.forth.gr/~dienst/dienstadmin.html* [XVI].

Στο Σχήμα 33 φαίνεται η σελίδα επιλογής των server και στο Σχήμα 34 οι πληροφορίες για τα ports που καταλαμβάνονται σε κάθε μηχάνημα.




Σχήμα 33. Επιλογή των server (ή των groups από servers) για εκτύπωση πληροφοριών γι' αυτούς.




| | |
|---|---|
| majestix:7000 | DIENST4.1.pricing.1 server's http_port runing at host majestix |
| mickey:1436 | META_DIENST server's server_port runing at host mickey |
| mickey:1437 | META_DIENST server's callback_port runing at host mickey |
| mickey:1500 | META_DIENST server's http_port runing at host mickey |
| mickey:2000 | DIENST4.1.pricing.1 server's callback_port runing at host mickey |
| mickey:4890 | DIENST4.1.pricing.1 server's server_port runing at host mickey |
| mickey:7000 | DIENST4.1.pricing.1 server's http_port runing at host mickey |
| pleiades:1235 | pleiades.MIS_DIENST4.1.4 server's http_port runing at host pleiades |
| pleiades:1236 | pleiades.MIS_DIENST4.1.4 server's server_port runing at host pleiades |
| pleiades:1237 | pleiades.MIS_DIENST4.1.4 server's callback_port runing at host pleiades |
| pleiades:1335 | pleiades.RMS_DIENST4.1.4 server's http_port runing at host pleiades |
| pleiades:1336 | pleiades.RMS_DIENST4.1.4 server's server_port runing at host pleiades |
| pleiades:1337 | pleiades.RMS_DIENST4.1.4 server's callback_port runing at host pleiades |
| pleiades:8080 | DIENST4.0 server's server_port runing at host pleiades |
| pleiades:8090 | DIENST4.0 server's callback_port runing at host pleiades |
| zozefina:5123 | DIENST4.1.2.statistic.1 server's debugger_port runing at host zozefina |
| zozefina:5345 | DIENST4.1.2.statistic.1 server's callback_port runing at host zozefina |
| zozefina:8080 | DIENST4.1.2.statistic.1 server's http_port runing at host zozefina |
| zozefina:8543 | DIENST4.1.2.statistic.1 server's server_port runing at host zozefina |

Σχήμα 34. Πληροφορίες του dienstadmin για τα ports που καταλαμβάνονται σε κάθε μηχάνημα.

# *Παράρτημα VII*

# VII. Meta server

## VII.i. Introduction

This server, is a demon that serves dienst Meta requests. In order to simplify the installation, and in order to be able to run it at a same host with a dienst server, you don't need to install a web server to communicate through. This server itself, is able to parse and serve requests made with the HTTP/1.0 protocol.

With this server, you can organize your own dienst collection, and if you want, you can set it up in an asymmetric mode (which means that your dienst servers with be able to "see" dienst servers of other collections, e.g. the ncstrl one).

## VII.ii. Installation

First untar the file: merged_meta_server.tar.gz A directory META will be created as is shown in the README.file file. Then, before running it, you must configure it.

## VII.iii. Configuration

1. Edit the **config.pl** file: You may need to change the following variables: $server_port, @meta_servers, $SOCK_STREAM, $source_path
   **Variables:**
   $server_port: is the port that the server will be listening to, for requests. If you don't run the server as the root user, the server_port must be greater than 1024.
   @meta_servers: Is an array with the external meta servers, that this meta server will grab information. If you just want to set up your own collection, and you do not want to be able to "see" external collections, you must set this variable to: @meta_servers = (); Otherwise, if you want to communicate with ncstrl collection, you can add any regional

meta server (it is preferable to add the meta server of your region (see ncstrl documentation for the regional meta servers or http://www.ics.forth.gr/pleiades/projects/dlib/metaservers.html ) Or/And, of course, you can add any other meta server you want. e.g.: @meta_servers = ("dienst.csi.forth.gr:80", "second.meta.server:port", "third.meta.server:port");
$SOCK_STREAM: The SOCKSTREAM variable. This can be 1 or 2, depending on the operating system (for sunos, osf is 1, for solaris,irix is 2)
$source_path: This must be the full path, where you have untar the files.

2. Edit the file: **metaserver.pl** : In this file you may need to change the perl path, which is in the first line of the file as shown below:
#!/usr/local/bin/perl
Usually, the perl binary is placed in the above location, but it is possible in your machine to be anywhere else. For determining the perl path, you can use the command at the unix prompt: "which perl"

## VII.iv. Configuration of the dienst servers

For each dienst server, you want to get meta from this meta server, you must set-up the variables: $regional_meta_server (with the host name which this meta server is running), and $regional_meta_server_port (with the port that you setup this meta server to listen to). You can change these variable either in the dienst file: dienst/Config/config_constants.pl or in the file dienst/Config/install.config. In the second case you must rerun the dienst/Config/auto_config.pl utility. (See dienst installation instructions for further info).

## VII.v. Running the meta server

The meta server program can take only one argument. This could be none, stop or reload.

- Running the command: "./metaserver.pl" with no arguments will start up the meta server. If the server is already running, this will fail and you will get the message:
"bind: Address already in use at ./metaserver.pl line 115.".
In success you will get the message:
"Waiting for connection at port xxxx"

- Running the command: "./metaserver.pl stop" will shutdown the server.

- The command: "./metaserver.pl reload" will cause the server to reload itself, so all the config files in the "META/data" directory are re-read. Use this command when you have add or remove or change something in the configuration files. Alternatively, you can shutdown the server, and restart it.

# *Παράρτημα VIII*

## VIII. Υπολογισμός βάρους λέξης κλειδί

Τα παρακάτω αναφέρονται στο [22] (σελ. 64-66). Σε αυτό το παράρτημα αναφέρουμε μόνο ενδεικτικά μερικούς από τους τύπους που χρησιμοποιούνται για τον υπολογισμό του βάρους της λέξης κλειδί.

$f_{in}$ = frequency of term i in document n
$t_n$ = number of types (unique terms) in document n
$d_i$= number of postings of term i
$k_n$ = number of tokens (term occurrences) in document n $\left(=\sum_{i=1}^{M} f_{in}\right)$
$F_i$ = frequency of term I in database $\left(=\sum_{n=1}^{N} f_{in}\right)$
$N$ = number of documents in database
$M$ = number of terms in dictionary
$D$ = number of postings in database $\left(=\sum_{i=1}^{M} d_i\right)$
$K$ = number of tokens in database $\left(=\sum_{n=1}^{M}\sum_{i=1}^{M} f_{in}\right)$

| **Formula ($W_{in}$=)** | **Comments** |
|---|---|
| (1) 1 | 'Unweighted'. Simplest and most commonly used method (Sager, 1976) |
| (2) $\frac{1}{t_n}$ | D. Simplest consideration of document length (Sager, 1976) |
| *(6)* $f_{in}$ | F. Simplest consideration of with-in document frequency (Sager, 1976) |
| (7) $\log f_{in}$ | F. Diminishes effect of many occurrences in a document (Sparck Jones, 1973) |
| (8) $\frac{f_{in}}{k_n}$ | FD. Simplest consideration of with-in document frequency and document length (using frequency information) (Sager, 1976) |

| | |
|---|---|
| (9) $\frac{f_{in}}{\log k_n}$ | FD. Like No. (8) but diminishes impact of document length |
| (12) $\frac{1}{d_i}$ | C. Simplest consideration of collection frequency (Sager, 1976) |
| (14) $\log \frac{N}{d_i}$ | C. Based on the information content of a term about a document. Salton, Wong and Yang |
| (18) $\frac{f_{in}}{F_i}$ | FC. No. (12) with frequency information (Sparck Jones, 1973) |

# *Παράρτημα IX*

## IX. Στατιστική ανάλυση της συλλογής NCSTRL-ERCIM

Εδώ παρουσιάζουμε μια στατιστική μελέτη του ρυθμού αύξησης των βάσεων δεδομένων του dienst και αύξησης μεγέθους των αρχείων δεικτοδότησης (index files). Πριν προχωρήσουμε, θα πρέπει να αναφερθεί ότι η οργάνωση των αρχείων δεικτοδότησης που χρησιμοποιήθηκε, δεν είναι ακριβώς ίδια με αυτή του dienst, αλλά αρκετά παρόμοια. Παρόλα αυτά μπορούμε να βγάλουμε συμπεράσματα και για την βάση του dienst.

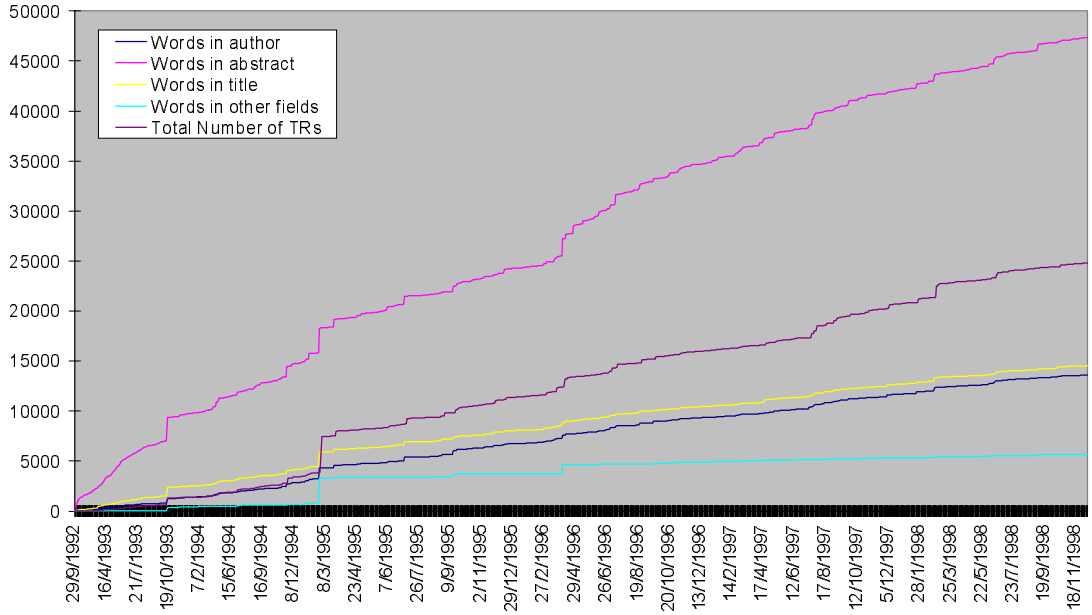
Στα παρακάτω διαγράμματα, ο άξονας Χ παριστάνει την ημερομηνία την οποία κάποιο έγγραφο εισήχθηκε στην βάση του dienst. Αυτή η ημερομηνία, είναι διαφορετική από την ημερομηνία δημοσίευσης του κειμένου. Χρησιμοποιήθηκε αυτή η ημερομηνία (το πεδίο "ENTRY" στα βιβλιογραφικά αρχεία) αντί της ημερομηνίας δημοσίευσης ("DATE") του κειμένου, για τους παρακάτω τρεις απλούς λόγους:

3. Το πεδίο "DATE" δεν υπάρχει σε αρκετά έγγραφα.
4. Οι εκδότες δεν χρησιμοποιούν τον ίδιο τρόπο αναπαράστασης ημερομηνίας σε αυτό το πεδίο, και συνεπώς η ανάλυση αυτού του πεδίου θα ήταν περισσότερο πολύπλοκη (σε αρκετά έγγραφα αναφέρεται μόνο έτος, ενώ στα περισσότερα μήνας-έτος).
5. Και τέλος, δεν μας αφορά ο ρυθμός δημοσίευσης των εγγράφων, αλλά ο ρυθμός με τον οποίο αυτά εισάγονται στην βάση.

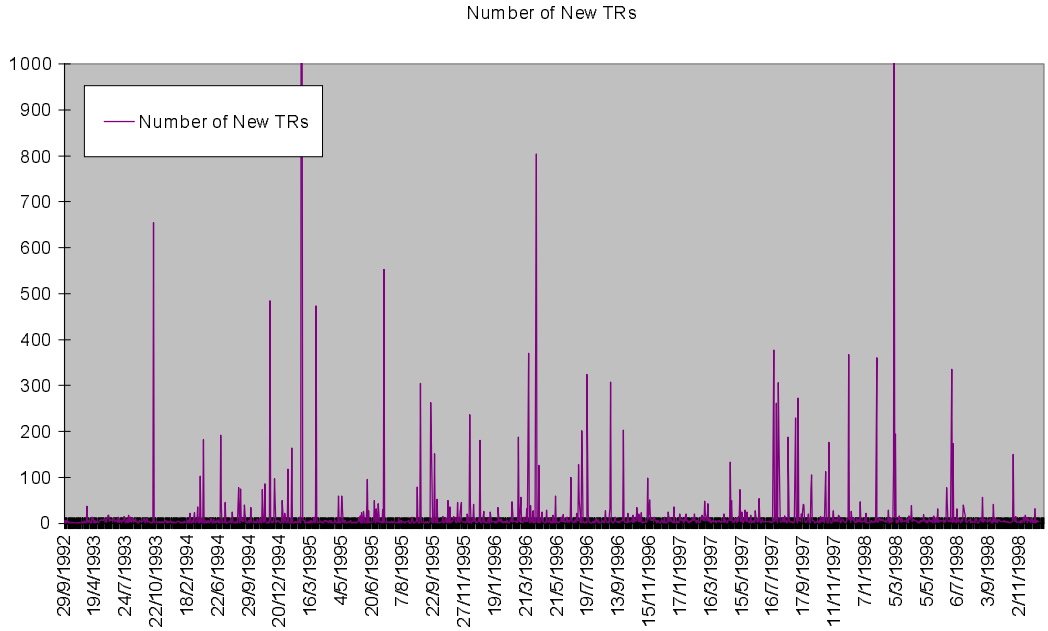
Από όλα τα διαγράμματα, είναι φανερό ότι υπάρχουν κάποιες ημερομηνίες, κατά τις οποίες εισήχθηκαν στην βάση πολλά κείμενα. Είναι οι ημερομηνίες που οι "εκδότες" εγκατέστησαν το dienst, και έτσι βλέπουμε κάποια σημεία στα οποία οι γραφικές παραστάσεις αυξάνουν απότομα. Αυτή η συμπεριφορά της βάσης δεν χαλάει την γενικότητα των στατιστικών στοιχείων, γιατί ακριβώς τα ίδια φαινόμενα είναι αναμενόμενα και στο μέλλον, όταν νέοι "εκδότες" αποφασίσουν να εγκαταστήσουν το dienst.

Όπως και είναι προφανές, τα αποτελέσματα αυτής της μελέτης, ήταν αναμενόμενα, δηλαδή η συνεχής αυξητική πορεία των τιμών (πλήθος κειμένων, πλήθος λέξεων, κτλ.). Θα θέλαμε όμως να δούμε τα ακριβή μεγέθη των τιμών αυτών, καθώς και τον ρυθμό αύξησης. Επίσης το πολύ ενδιαφέρον στατιστικό που προέκυψε είναι το πλήθος των νέων λέξεων που εισάγονται στα αρχεία ευρετηρίου για κάθε νέο κείμενο που εισάγεται στην βάση. Έτσι, όπως φαίνεται στο Γράφημα 10 και Γράφημα 11, η εισαγωγή ενός νέου κειμένου στην βάση, οδηγεί στην

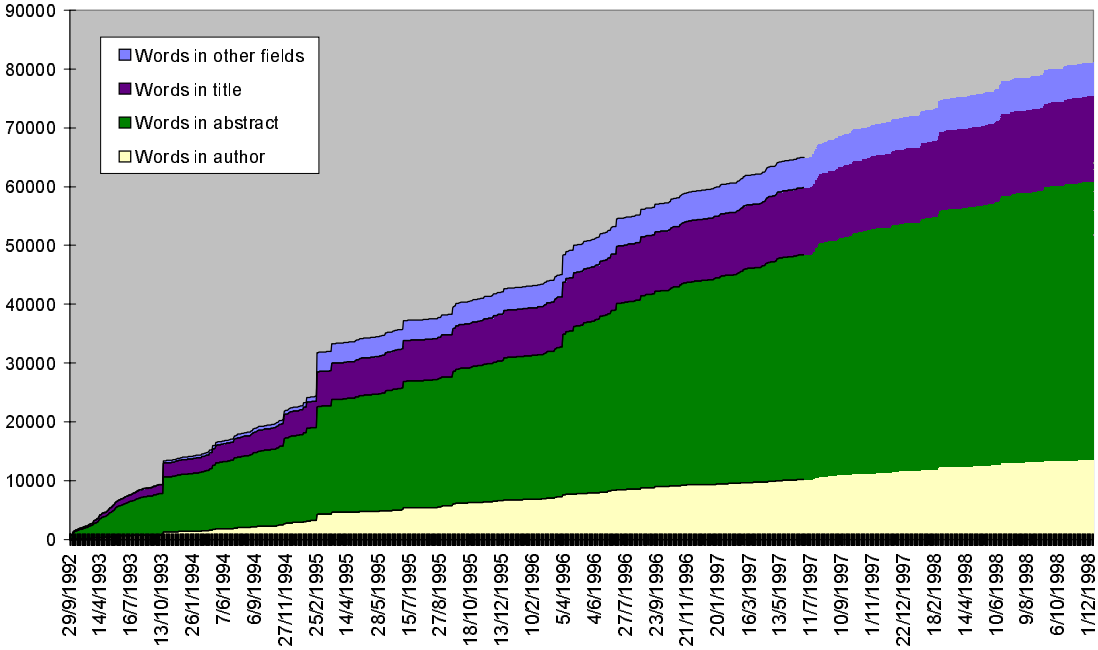
εισαγωγή 3 περίπου λέξεων στα αρχεία ευρετηρίων. Η τιμή αυτή έχει σταθεροποιηθεί στο προηγούμενο νούμερο από τον Μάρτιο του 1995 και παραμένει σταθερή.



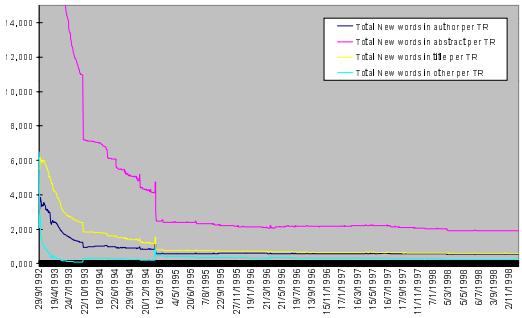

Γράφημα 7. Ρυθμός αύξησης των Technical Reports και του πλήθους των λέξεων (ανά πεδίο) στα αρχεία ευρετηρίου



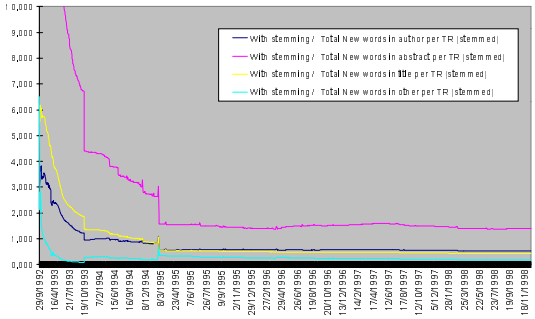

Γράφημα 8. Πλήθος των TRs που εισήχθηκαν ανά μέρα στην βάση.




Γράφημα 9. Συνολικό πλήθος λέξεων ανά πεδίο.



Γράφημα 10. Νέες λέξεις κάθε πεδίου ανά κείμενο



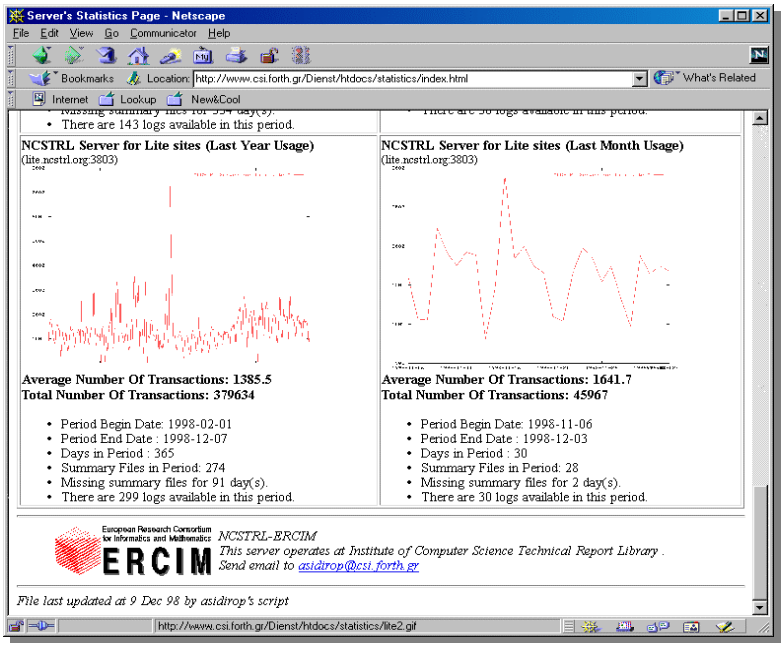
Γράφημα 11. Νέες ρίζες λέξεων κάθε πεδίου ανά κείμενο

# *Παράρτημα X*

## X. Στατιστική ανάλυση χρήσης της συλλογής NCSTRL-ERCIM

Το πρωτόκολλο dienst μας επιτρέπει να ζητούμε και να βλέπουμε τα αρχεία καταγραφής (log files) κάποιου dienst server. Επίσης είναι δυνατό να πάρουμε στατιστικές πληροφορίες για την χρήση του server, όπως πόσες αιτήσεις χρηστών γίνανε κάποια μέρα, ποια κείμενα είναι περισσότερο δημοφιλή κτλ. (Βλέπε [14]).

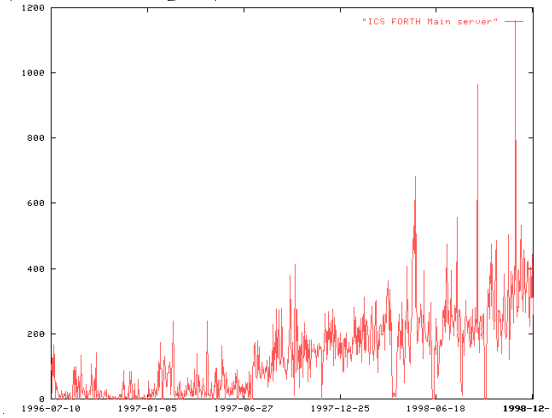
Στο ΙΤΕ/ΙΠ, προκειμένου να αποκτήσουμε μια συνολική εικόνα της χρήσης της συλλογής, δημιουργήσαμε έναν ακόμη "spider", ο οποίος μαζεύει στατιστικά στοιχεία από επιλεγμένους εξυπηρετητές, μορφοποιεί τα στοιχεία αυτά σε γραφήματα (έτσι ώστε να παρουσιάζεται μια γενική εικόνα χρήσης με μια μόνο ματιά). Η όλη αυτή η διαδικασία γίνεται αυτόματα, και παρουσιάζονται τα στοιχεία συνεχώς στην ιστοσελίδα:
*http://dienst.csi.forth.gr/Dienst/htdocs/statistics/index.html* , όπως φαίνεται στο παρακάτω σχήμα (Σχήμα 35).




Σχήμα 35. Παρουσίαση της χρήσης της συλλογής NCSTRL-ERCIM.

Στην παραπάνω σελίδα, για κάθε εξυπηρετητή που έχουμε ορίσει, εμφανίζονται 2 γραφήματα. Το πρώτο δείχνει την χρήση του εξυπηρετητή κατά την διάρκεια ολόκληρου του τελευταίου έτους, και το δεύτερο, την χρήση του κατά την διάρκεια του τελευταίου μήνα. Έτσι η χρήση του dienst server του ΙΤΕ/ΙΠ είναι:

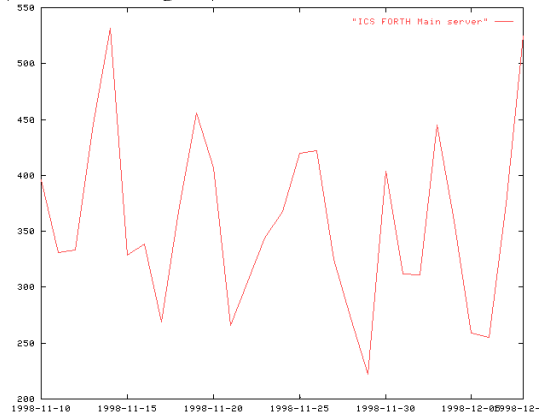
ICS_FORTH Main server (Total Usage)
(dienst.csi.forth.gr:80)




Average Number Of Transactions: 131.7
Total Number Of Transactions: 111715

- Period Begin Date: 1996-07-10
- Period End Date : 1998-12-06
- Days in Period : 879
- Summary Files in Period: 848
- Missing summary files for 31 day(s).

ICS_FORTH Main server (Last Month Usage)
(dienst.csi.forth.gr:80)




Average Number Of Transactions: 356.5
Total Number Of Transactions: 10695

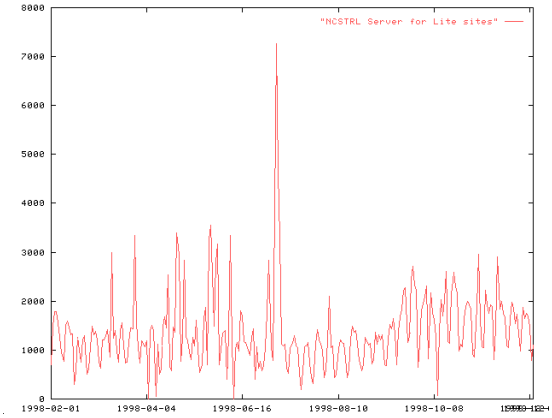
- Period Begin Date: 1998-11-10
- Period End Date : 1998-12-09
- Days in Period : 30

Πίνακας 18. Χρήση του εξυπηρετητή του ΙΤΕ/ΙΠ

Στο παραπάνω πίνακα βλέπουμε, ότι η χρήση του εξυπηρετητή στο ΙΤΕ/ΙΠ έχει μια αύξουσα πορεία από την ημερομηνία εγκατάστασής του[42]. Επίσης φαίνεται ότι τον τελευταίο μήνα (10/11/98 έως 9/12/98), η μέση χρήση του συστήματος ήταν 356 αιτήσεις ημερησίως.

[42] Για τους τοπικούς εξυπηρετητές στο ΙΠΕ/ΙΠ, παρουσιάζουμε, αντί τα στατιστικά της τελευταίας χρονιάς, στατιστικά από την ημερομηνία εγκατάστασης του συστήματος.

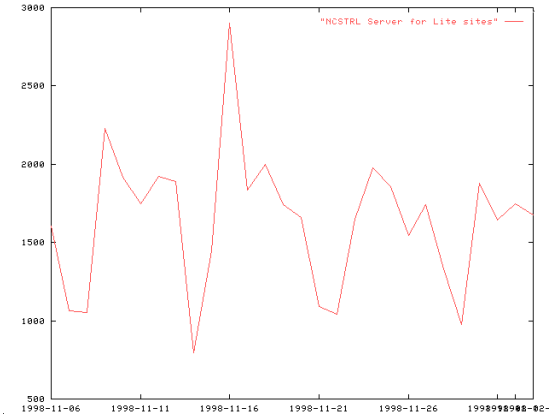
NCSTRL Server for Lite sites (Last Year Usage)
(lite.ncstrl.org:3803)




Average Number Of Transactions: 1385.5
Total Number Of Transactions: 379634

- Period Begin Date: 1998-02-01
- Period End Date : 1998-12-07
- Days in Period : 365
- Summary Files in Period: 274
- Missing summary files for 91 day(s).

NCSTRL Server for Lite sites (Last Month Usage)
(lite.ncstrl.org:3803)




Average Number Of Transactions: 1641.7
Total Number Of Transactions: 45967

- Period Begin Date: 1998-11-06
- Period End Date : 1998-12-03
- Days in Period : 30
- Summary Files in Period: 28
- Missing summary files for 2 day(s).

Πίνακας 19. Χρήση του εξυπηρετητή για τα NCSTRL lite sites.

# ΒΙΒΛΙΟΓΡΑΦΙΑ


[1] Carl Lagoze and James R. Davis. *Dienst: An Architecture for Distributed Document Libraries*. Communications of the ACM, April 1995, Vol. 38, No 4, pg. 47.

[2] David M. Levy and Catherine C. Marshall. *Going Digital: A look at Assumptions Underlying Digital Libraries*. Communications of the ACM, April 1995, Vol. 38, No 4, pg. 77-83.

[3] Edward A. Fox. World-Wide Web and Computer Science Reports. Communications of the ACM, April 1995, Vol. 38, No 4, pg. 43-44.

[4] Gary Marchionini and Hermann Maurer. *The Roles of Digital Libraries In Teaching and Learning*. Communications of the ACM, April 1995, Vol. 38, No 4, pg. 67-75.

[5] Gerald Salton. *Introduction to Modern Information Retrieval*. McGraw-Hill, 1993.

[6] Gio Wiederhold. *Digital Libraries, Value, and Productivity*. Communications of the ACM, April 1995, Vol. 38, No 4, pg. 85-95.

[7] James C. French, Edward A. Fox, Kurt Maly and Alan L. Selman. *Wide Area Technical Report Service: Technical Reports Online*. Communications of the ACM, April 1995, Vol. 38, No 4, pg. 45.

[8] James R. Davis, Carl Lagoze. *"Drop-in" publishing with the World Wide Web*. Dienst technical report server. Technical Report TR-95-1614, Xerox Inc and Cornell University, 1995.

[9] James R. Davis, Carl Lagoze. *A protocol and server for a distributed digital technical report library*. Dienst technical report server. Technical Report TR-94-1514, Cornell University, 1994.

[10] Ramana Rao, Jan O. Pedersen, Marti A. Hearst, Jock D. Mackinlay, Stuart K. Card, Larry Masinter, Per-Kristian Halvorsen and George G. Robertson. Rich *Interaction in the Digital Library*. Communications of the ACM, April 1995, Vol. 38, No 4, pg. 29-39.

[11] Tim Berners-Bee, Robert Cailliau, Jean-Francois Groff, and Berd Pollermann. World-wide web: The information universe. *Electronic Networking: Research, Applications and Policy*, 2(1):52-58, 1992.

[12] Edward A. Fox, Robert M. Akscyn, Richard K. Fururta and John J. Leggett. *Introduction to Digital Libraries*. Communications of the ACM, April 1995, Vol. 38, No 4, pg. 22-28.

[13] Piller, C. Dreamnet. *Macworld*. Oct. 1994, 96-105.

[14] Carl Lagoze, Erin Shaw, James R. Davis and Dean B. Krafft, *Dienst: Implementation Reference Manual*. TR95-1514, Cornell University, Computer Science, http://cs-tr.cs.cornell.edu/Dienst/UI/2.0/Describe/ncstrl.cornell/TR95-1514

[15] James R. Davis and Carl Lagoze. *The Networked Computer Science Technical Report Library*. TR96-1595, Cornell University, http://cs-

tr.cs.cornell.edu/Dienst/UI/2.0/Describe/ncstrl.cornell/TR96-1595

[16] Carl Lagoze and David Ely. *Implementation Issues in an Open Architectural Framework for Digital Object Services*. TR95-1540, Cornell University, http://cs-tr.cs.cornell.edu/Dienst/UI/2.0/Describe/ncstrl.cornell/TR95-1540

[17] James R. Davis and Carl Lagoze. *A protocol and server for a distributed digital technical report library*. TR94-1418, http://cs-tr.cs.cornell.edu/Dienst/UI/2.0/Describe/ncstrl.cornell/TR94-1418

[18] Bowman, C. M., Danzig, P. B., Hardy, D. R., Manber, U. and Schwartz, M. F. *The HARVEST information discovery and access system*. In Proceedings of the Second International WWW Conference (Chicago, Illinois, Oct. 1994), pp. 763-771, ftp://ftp.cs.colorado.edu/pub/cs/techreports/schwartz/Harvest.Conf.ps.Z

[19] Horowitz, *L.R. CETH workshop on documenting electronic texts, Tech. Rep. 2,* Center for Electronic Texts in the Humanities, New Brunswick, N.J., 1994

[20] Σωτήρης Τερζής,, *Εποπτεία επίδοσης σε Συστήματα Ψηφιακών Βιβλιοθηκών*. Master Thesis, Πανεπιστήμιο Κρήτης, Σεπ. 1997.

[21] J. Sairamesh, S. Kapidakis, S. Terzis, A. Sidiropolous, A. Hatzstamatiou, C. Nikolaou, *Measuring and Monitoring DIENST Behavior: A Performance Management Architecture*. ICS-FORTH and University of Crete, http://www.csi.forth.gr/~dienst/docs/

[22] R. N. Oddy, S. E. Robertson, C. J. van Rijsbergen, P. W. Williams, *Information Retrieval Research*, Butterworths.

# ΑΝΑΦΟΡΕΣ ΣΤΟ ΔΙΑΔΙΚΤΥΟ


[I] http://www.altavista.digital.com , Altavista Search Engine

[II] http://www.hotbot.com , Hotbot Search Engine

[III] http://www.metacrawler.com , Meta Crawler Search Engine

[IV] http://www.excite.com , Excite Search Engine

[V] http://www.lycos.com , Lycos Search Engine

[VI] http://cs-tr.cs.cornell.edu , NCSTRL Collection (entry point)

[VII] http://dienst.csi.forth.gr/TR , NCSTRL Collection (entry point)

[VIII] http://www.sils.umich.edu/UMDL/homePage.html , University of Michigan Digital Library

[IX] http://www.grainger.uiuc.edu/dli , Illinios Digital Library

[X] http://alexandria.sdc.ucsb.edu/ , Alexandria Digital Library

[XI] http://http.cs.berkeley.edu/~wilensky , UC Berkeley's Digital Library

[XII] http://www-diglib.stanford.edu/diglib , The Stanford Digital Library Project

[XIII] http://fuzine.mt.cs.cmu.edu/im/informedia.html , Informedia Digital Library

[XIV] http://www.csi.forth.gr/~asidirop/DIENST.DEBUG/index.html , Dienst Debugging and profiling

[XV] http://www.csi.forth.gr/~dienst , Dienst Home Page.

[XVI] http://www.csi.forth.gr/~dienst/dienstadmin.html , dienst administration interface and instructions

[XVII] http://mel.dmso.mil/ , MEL Home page

[XVIII] http://spsosun.gsfc.nasa.gov/ , EOSDIS Home Page

[XIX] http://dods.gso.uri.edu/DODS , DOD Home Page

[XX] http://cs-tr.cs.cornell.edu/Dienst/htdocs/dienst_install/ , Dienst installation instructions

[XXI] http://dienst.csi.forth.gr/Dienst/htdocs/dienst_install/ , Dienst installation instructions

[XXII] http://info.webcrawler.com/mak/projects/robots/active/html/type.html , The Web Robots Pages

[XXIII] http://www.cs.indiana.edu:800/cstr/ , UCSTRI Search Page

[XXIV] http://www.fgdc.gov/Metadata/Metadata.html , FGDC, Content Standard for Digital Geosptial Metadata

[XXV] http://seagrant.mit.edu/~auvlab/loops.html , LOOPS, A National Littoral Ocean Observing and Predictive System

[XXVI] http://www.ics.forth.gr/pleiades/THETIS/thetis.html , THETIS, A Data Management and Visualization System for Coastal Zone Management of the Mediterranean Sea

[XXVII] http://deslab.mit.edu/ , MIT, Department of Ocean Engineering Design Laboratory

[XXVIII] http://oasis.leidenuniv.nl/general/index-search/choices/choices.htm , CHOICE: (R)DBMS or WAIS / freeWAIS-sf ?

[XXIX] http://ls6-www.informatik.uni-dortmund.de/freeWAIS-sf/ , Pfeifer, U: FreeWAIS-sf

[XXX] http://www.igd.fhg.de/www/www95/papers/47/fwsf/fwsf.html , Searching Structured Documents with the Enhanced Retrieval Functionality of freeWAIS-sf and Sfgate

[XXXI] http://glimpse.cs.arizona.edu/ , GLIMPSE , A tool to search entire file systems

# ΛΕΞΙΚΟ ΤΕΧΝΙΚΩΝ ΟΡΩΝ

## ΕΛΛΗΝΙΚΑ → ΑΓΓΛΙΚΑ

| | |
|---|---|
| **Α** | |
| Αίτηση | Request |
| Αλληλεπιδραστικός | Interactive |
| Αναγνωριστικό Διεργασίας | Process id |
| Αναγνωριστικό Εγγράφου | Document Handle |
| Αναδιφώ/ Αναδίφηση | Browse |
| Ανάκληση | Retrieval |
| Αναπαραγωγή Διεργασίας | Fork |
| Ανοχή στις αποτυχίες | Fault-tolerance |
| Αξιοπιστία | Dependability |
| Απευθείας Απάντηση | Direct Callback |
| Αποσυνθέσεις Εγγράφων | Document Decompositions |
| Αποσφαλμάτωση | Debugging |
| Αρχείο εικονικής μνήμης | Swap file |
| Αρχείο Καταγραφής | Log File |
| Αρχή Έκδοσης | Authority [Publishing] |
| Αρχή Έκδοσης | Publishing Authority |
| Αρχιτεκτονική Περιοχών | Regional Architecture |
| Ασφάλεια | Security/Safety |
| **Β** | |
| Βιβλιογραφική Εγγραφή | Bibliographic Record |
| **Γ** | |
| Γλώσσα Σημείωσης Υπερκειμένου | Hypertext Markup Language-HTML |
| **Δ** | |
| Δεδομένα για τα Δεδομένα | Meta Data |
| Διαδικασία | Procedure |
| Διαδικασία Διαχείρισης της Βάσης των | Database Management Process |
| Διαδικασία Μέτρησης | Measurement Process |
| Διαδίκτυο | Internet |
| Διεπιφάνεια Χρήσης | User Interface |
| Διεργασία | Process |
| Δοσοληψία | Transaction |
| **Ε** | |
| Εικονική μνήμη | Virtual memory |
| Εν Λειτουργία Επόπτης Επίδοσης | On-line Performance Monitor |
| Ενήμερος | Up to date |
| Εξέταση | Probing |
| Εξισορρόπηση Φόρτου | Load Balancing |
| Εξυπηρετητής | Server |
| Εξυπηρετητής Αναγνωριστικών | Handle Server |
| Εξυπηρετητής Ευρετηριασμού | Index Server |
| Εξυπηρετητής Περιοχής | Regional Server |
| Εξυπηρετητής Συγχωνευμένου Ευρετηρίου | Merged Index Server |
| Εξυπηρετητής Υποστήριξης | Backup Server |
| Επανεκκίνηση | Reset, Restart |
| Επερώτηση | Query |
| Επίπεδο Μεταφοράς | Transport Layer |
| Ευρετήριο | Index |
| **Ζ** | |
| **Η** | |
| **Θ** | |
| **Ι** | |
| **Κ** | |
| Καθυστέρηση Δικτύου | Network Latency |
| Κατασκευή Προφίλ Προγράμματος | Program Profiling |
| Κόμβος | Node |
| Κωδικός Εγγράφου | Docid |
| **Λ** | |
| Λέξη Κλειδί | Keyword |
| Λογαριασμός Χρήσης | Account |
| **Μ** | |
| Μεταφερσιμότητα | Portability |
| Μηχανή Αναζήτησης | Search Engine |
| Μοντέλο Πελάτη-Εξυπηρετητή | Client-Server Model |
| **Ν** | |
| **Ξ** | |
| **Ο** | |
| Ομοιόμορφος Προσδιοριστής Πόρων | Uniform Resource Locator–URL |
| Ορμαθός Χαρακτήρων | String |

| | |
|---|---|
| **Π** | |
| Παγκόσμιο Δίκτυο | Internet |
| Παγκόσμιος [Δικτυακός] Ιστός | World Wide Web |
| Πελάτης Παγκόσμιου Ιστού | Web Client |
| Πίνακας Κατακερματισμού | Hash Table |
| Ποιότητα Υπηρεσίας | Quality of Service (QoS) |
| Πόρος | Resource |
| Πράκτορας | Agent |
| Πρότυπο Πρόσβασης | Access Pattern |
| **Ρ** | |
| **Σ** | |
| Σημείο Αναφοράς | Named Anchor |
| Συμβάν | Event |
| Συνεδρία | Session |
| Σύστημα Αναγνωριστικών Εγγράφων | Handle System |
| **Τ** | |
| Τμήμα Εκτέλεσης | Module |
| Τοποθεσία, Κόμβος | Site |
| **Υ** | |
| Υπερκείμενο | Hypertext |
| Υπηρεσία Αποθήκευσης | Repository Service |
| Υπηρεσία Ευρετηριασμού | Index Service |
| Υπηρεσία Ονοματοδοσίας | Name Service |
| **Φ** | |
| **Χ** | |
| Χρήσιμος Χρόνος Συστήματος | Utilization |
| Χρονικό Περιθώριο | Timeout |
| Χρονοσφραγίδα | Timestamp |
| Χώρος Αποθήκευσης | Repository |
| **Ψ** | |
| Ψηφιακές Βιβλιοθήκες | Digital Libraries |
| **Ω** | |

# ΛΕΞΙΚΟ ΤΕΧΝΙΚΩΝ ΟΡΩΝ

## ΑΓΓΛΙΚΑ → ΕΛΛΗΝΙΚΑ

| | |
|---|---|
| **A** | |
| Access Pattern | Πρότυπο Πρόσβασης |
| Account | Λογαριασμός Χρήσης |
| Agent | Πράκτορας |
| Authority [Publishing] | Αρχή Έκδοσης |
| **B** | |
| Backup Server | Εξυπηρετητής Υποστήριξης |
| Bibliographic Record | Βιβλιογραφική Εγγραφή |
| Browse | Αναδιφώ/ Αναδίφηση |
| **C** | |
| Client-Server Model | Μοντέλο Πελάτη-Εξυπηρετητή |
| **D** | |
| Database Management Process | Διαδικασία Διαχείρισης της Βάσης των |
| Debugging | Αποσφαλμάτωση |
| Dependability | Αξιοπιστία |
| Digital Libraries | Ψηφιακές Βιβλιοθήκες |
| Direct Callback | Απευθείας Απάντηση |
| Docid | Κωδικός Εγγράφου |
| Document Decompositions | Αποσυνθέσεις Εγγράφων |
| Document Handle | Αναγνωριστικό Εγγράφου |
| **E** | |
| Event | Συμβάν |
| **F** | |
| Fault-tolerance | Ανοχή στις αποτυχίες |
| Fork | Αναπαραγωγή Διεργασίας |
| **G** | |
| **H** | |
| Handle Server | Εξυπηρετητής Αναγνωριστικών |
| Handle System | Σύστημα Αναγνωριστικών Εγγράφων |
| Hash Table | Πίνακας Κατακερματισμού |
| Hypertext | Υπερκείμενο |
| Hypertext Markup Language-HTML | Γλώσσα Σημείωσης Υπερκειμένου |
| **I** | |
| Index | Ευρετήριο |
| Index Server | Εξυπηρετητής Ευρετηριασμού |
| Index Service | Υπηρεσία Ευρετηριασμού |
| Interactive | Αλληλεπιδραστικός |
| Internet | Διαδίκτυο |
| Internet | Παγκόσμιο Δίκτυο |
| **J** | |
| **K** | |
| Keyword | Λέξη Κλειδί |
| **L** | |
| Load Balancing | Εξισορρόπηση Φόρτου |
| Log File | Αρχείο Καταγραφής |
| **M** | |
| Measurement Process | Διαδικασία Μέτρησης |
| Merged Index Server | Εξυπηρετητής Συγχωνευμένου Ευρετηρίου |
| Meta Data | Δεδομένα για τα Δεδομένα |
| Module | Τμήμα Εκτέλεσης |
| **N** | |
| Name Service | Υπηρεσία Ονοματοδοσίας |
| Named Anchor | Σημείο Αναφοράς |
| Network Latency | Καθυστέρηση Δικτύου |
| Node | Κόμβος |
| **O** | |
| On-line Performance Monitor | Εν Λειτουργία Επόπτης Επίδοσης |
| **P** | |
| Portability | Μεταφερσιμότητα |
| Probing | Εξέταση |
| Procedure | Διαδικασία |
| Process | Διεργασία |
| Process id | Αναγνωριστικό Διεργασίας |
| Program Profiling | Κατασκευή Προφίλ Προγράμματος |
| Publishing Authority | Αρχή Έκδοσης |
| **Q** | |
| Quality of Service (QoS) | Ποιότητα Υπηρεσίας |
| Query | Επερώτηση |
| **R** | |

| | |
|---|---|
| Regional Architecture | Αρχιτεκτονική Περιοχών |
| Regional Server | Εξυπηρετητής Περιοχής |
| Repository | Χώρος Αποθήκευσης |
| Repository Service | Υπηρεσία Αποθήκευσης |
| Request | Αίτηση |
| Reset | Επανεκκίνηση |
| Resource | Πόρος |
| Retrieval | Ανάκληση |
| **S** | |
| Search Engine | Μηχανή Αναζήτησης |
| Security/Safety | Ασφάλεια |
| Server | Εξυπηρετητής |
| Session | Συνεδρία |
| Site | Τοποθεσία, Κόμβος |
| String | Ορμαθός Χαρακτήρων |
| Swap file | Αρχείο εικονικής μνήμης |
| **T** | |
| Timeout | [Μέγιστο] Χρονικό Περιθώριο |
| Timestamp | Χρονοσφραγίδα |
| Transaction | Δοσοληψία |
| Transport Layer | Επίπεδο Μεταφοράς |
| **U** | |
| Uniform Resource Locator–URL | Ομοιόμορφος Προσδιοριστής Πόρων |
| Up to date | Ενήμερος |
| User Interface | Διεπιφάνεια Χρήσης |
| Utilization | Χρήσιμος Χρόνος Συστήματος |
| **V** | |
| Virtual memory | εικονική μνήμη |
| **W** | |
| Web Client | Πελάτης Παγκόσμιου Ιστού |
| World Wide Web | Παγκόσμιος [Δικτυακός] Ιστός |
| **X** | |
| **Y** | |
| **Z** | |

# ΑΚΡΩΝΥΜΙΑ

**C4GVC.** CAD/CAM/CAE/CAGD/Graphics and Visualization DIGITAL LIBRARY

**CGI**. Common Gateway Interface

**DNS.** Distributed Name Server

**ERCIM**. European Research Consortium for Informatics and Mathematics

**FIFO**. First In – First Out

**FTP.** File Transfer Protocol

**HTML**. HyperText Markup Language

**HTTP.** HyperText Transfer Protocol

**MIT** Massachusetts Institute of Technology

**NCSTRL**. Networked Computer Science Technical Report Library

**OCR**. Optical Character Recognition

**RFC.** Request For Comments

**SHTTP.** Secure HyperText Transfer Protocol

**SQL**. Structured Query Language

**UCSTRI.** Unified Computer Science TR Index

**UI**. User Interface

**URL**. Uniform Resource Locator

**WAIS.** Wide Area Information Service

**WWW**. World Wide Web

**ΙΤΕ/ΙΠ.** Ίδρυμα Τεχνολογίας Έρευνας / Ινστιτούτο Πληροφορικής

# ΕΥΡΕΤΗΡΙΟ

## *W*



## *A*



## *B*



## *Γ*



## *Δ*



## *Z*



## *Θ*



## *I*



## *K*



## *M*



## *O*



## *Π*



## *Σ*