ΑΛΕΞΑΝΔΡΕΙΟ ΤΕΧΝΟΛΟΓΙΚΟ ΕΚΠΑΙΔΕΥΤΙΚΟ ΙΔΡΥΜΑ ΘΕΣΣΑΛΟΝΙΚΗΣ

ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ

ΠΡΟΓΡΑΜΜΑ ΜΕΤΑΠΤΥΧΙΑΚΩΝ ΣΠΟΥΔΩΝ

ΕΥΦΥΕΙΣ ΤΕΧΝΟΛΟΓΙΕΣ ΔΙΑΔΙΚΤΥΟΥ – WEB INTELLIGENCE

Ανάπτυξη και αξιολόγηση web-based συστημάτων για αναζήτηση σε ημι-δομημένη πληροφορία με χρήση faceted search

ΜΕΤΑΠΤΥΧΙΑΚΗ ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

του

ΑΠΟΣΤΟΛΟΣ ΜΑΒΙΔΗΣ

Επιβλέπων : ΜΙΧΑΗΛ ΣΑΛΑΜΠΑΣΗΣ
Καθηγητής, ΑΤΕΙΘ

Θεσσαλονίκη, Φεβρουάριος 2016

Η σελίδα αυτή είναι σκόπιμα λευκή.

ΑΛΕΞΑΝΔΡΕΙΟ ΤΕΧΝΟΛΟΓΙΚΟ ΕΚΠΑΙΔΕΥΤΙΚΟ ΙΔΡΥΜΑ ΘΕΣΣΑΛΟΝΙΚΗΣ
ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ
ΠΡΟΓΡΑΜΜΑ ΜΕΤΑΠΤΥΧΙΑΚΩΝ ΣΠΟΥΔΩΝ
ΕΥΦΥΕΙΣ ΤΕΧΝΟΛΟΓΙΕΣ ΔΙΑΔΙΚΤΥΟΥ – WEB INTELLIGENCE

Ανάπτυξη και αξιολόγηση web-based συστημάτων για αναζήτηση σε ημι-δομημένη πληροφορία με χρήση faceted search

ΜΕΤΑΠΤΥΧΙΑΚΗ ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

του

ΑΠΟΣΤΟΛΟΣ ΜΑΒΙΔΗΣ

Επιβλέπων : ΜΙΧΑΗΛ ΣΑΛΑΜΠΑΣΗΣ
Καθηγητής, ΑΤΕΙΘ

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή στις 3 Φεβρουαρίου 2016.

(Υπογραφή)	*(Υπογραφή)*	*(Υπογραφή)*
.....	.....	.....
Μιχαήλ Σαλαμπάσης Καθηγητής Α.Τ.Ε.Ι.Θ.	Παναγιώτης Σφέτσος Αναπληρωτής Καθηγητής Α.Τ.Ε.Ι.Θ.	Αντώνιος Σιδηρόπουλος Καθηγητής Εφαρμογών / Λέκτορας Α.Τ.Ε.Ι.Θ.

Θεσσαλονίκη, Φεβρουάριος 2016

(Υπογραφή)

.....

Απόστολος Μαβίδης

Μηχανικός Πληροφορικής ΑΤΕΙΘ

«Αφιερωμένο στην κόρη μου Κωνσταντίνα.»

Η σελίδα αυτή είναι σκόπιμα λευκή.

Περίληψη

Από τις πρώτες διαδικασίες αναζήτησης που συνήθως αναζητούσαν φυσικά έγγραφα στο περιβάλλον μιας βιβλιοθήκης αλλά και τις μετέπειτα ηλεκτρονικές αναζητήσεις δεδομένων που βρίσκονται αποθηκευμένα σε ηλεκτρονικά μέσα, σε δομημένη ή όχι μορφή, οι διαδικασίες, οι μέθοδοι και τα συστήματα αναζήτησης πληροφορίας έχουν αλλάξει σημαντικά. Έχουν εμπλουτισθεί με εργαλεία, τεχνικές και επηρεάσθηκαν ακόμα και από εξωγενείς παράγοντες όπως η ανάπτυξη των ασύρματων δικτύων. Το άμεσο αποτέλεσμα αυτής της ανάπτυξης είναι η αναζήτηση να αποκτήσει τον χαρακτήρα μιας πιο σύνθετης διεργασίας έχοντας σαν στόχο να ικανοποιήσει πολύπλευρα τον χρήστη σε ένα εξαιρετικά πιο σύνθετο περιβάλλον.

Η αναζήτηση πληροφορίας μπορεί να στηρίζεται σε πολλές διαφορετικές μεθόδους ωστόσο ο βασικός σκοπός της παραμένει ο ίδιος. Να καταστήσει εύκολη τη πρόσβαση και την περεταίρω επεξεργασία σε πληροφορίες που βρίσκονται μεταξύ άλλων για να χρησιμοποιηθεί για ανάκτηση, παραγωγή ή σύνθεση γνώσης. Αν αναλογιστούμε το γεγονός ότι η ποσότητα των διαθέσιμων πληροφοριών είναι συνήθως τεράστια μέσα στον τεράστιο όγκο εγγράφων, φωτογραφιών, βίντεο κλπ, η διαδικασία της πρόσβασης στις πληροφορίες δεν είναι τελικά απλή υπόθεση.

Τα τελευταία χρόνια μία μέθοδος η «αναζήτηση με χρήση όψεων» (faceted search) δίνει τη δυνατότητα σε μη εκπαιδευμένους χρήστες να οπτικοποιήσουν δυναμικά τη διαδικασία αναζήτησης καλύτερα και εν τέλει να αναζητήσουν ευκολότερα και πιθανόν πιο αποτελεσματικά τις επιθυμητές πληροφορίες. Η faceted search ή faceted navigation ή faceted browsing είναι εξορισμού η πλέον αποδεκτή μέθοδος σε περιπτώσεις όπως η αναζήτηση προϊόντων σε ηλεκτρονικά καταστήματα που πληρούν κάποια επιθυμητά κριτήρια. Υποστηρίζει τόσο την απευθείας αναζήτηση (query-based analytical search) όσο και για καθοδηγούμενη εξερεύνηση (explorative search) σύνθετης πληροφορίας.

Σκοπός της παρούσας διπλωματικής εργασίας είναι η παρουσίαση της τεχνολογίας faceted search και των βασικών χαρακτηριστικών της, τις ιδιαιτερότητές της και τα πλεονεκτήματα ή μειονεκτήματά της. Θα μελετήσουμε και αξιολογήσουμε λύσεις που έχουν ενσωματώσει την τεχνολογία αυτή όπως και τα διαθέσιμα εργαλεία για ανάπτυξη των δικών μας λύσεων.

Θα παρουσιάσουμε την ανάπτυξη μίας διαδικτυακής εφαρμογής που ενσωματώνει faceted search και φιλοδοξεί να δώσει τη δυνατότητα ενσωμάτωσης των δυνατοτήτων της σε άλλες εφαρμογές που χρειάζονται τέτοια χαρακτηριστικά αλλά έως τώρα δεν τα διαθέτουν.

Θα αναλύσουμε επίσης στοιχεία που επηρεάζουν την σχεδίαση εφαρμογών αναζήτησης στο web και ειδικά εφαρμογών με faceted search και θα προτείνουμε καλές πρακτικές σχεδίασης.

Λέξεις Κλειδιά: «Faceted search, Faceted Navigation, Faceted Browsing, Information Retrieval, Lucene.net, Apache Solr, dynamic taxonomy»

Η σελίδα αυτή είναι σκόπιμα λευκή.

Abstract

Information seeking tasks usually included a search of physical documents inside a public library. This information seeking process has seriously changed with the advent of computerized systems and digitally stored documents on hard drives and so on, being stored in structured or unstructured format. It was enriched with tools, techniques and sometimes influenced by external parameters such as the development of wireless networks. The result is that this information seeking process has become more complex, having its primary goal to versatile satisfy the user.

Though, Information Retrieval is based on a lot of different technologies it's basic objective remains the same. That is to provide an efficient and effective way of accessing relevant information stored among others. Considering the fact that amount of available information is huge, such a task is not very easy.

In recent years a method called "faceted search" provides the functionality to less experienced users to visualize and search effectively information spaces that could partially possess taxonomic data or these can by dynamically extracted. By now Faceted Search (faceted navigation, faceted browsing) is de facto solution for search subsystems applied in e-commerce applications. It supports both direct analytical search and also a more guided and explorative search of data.

The goal of the current thesis is to present faceted search technology along with its basic characteristics, its advantages and disadvantages. In addition, we will present the development of a new application with the ambition of providing faceted search characteristics to other applications in need, but with no such functionality. At the end, we discuss several matters that influence the design of web information retrieval systems, especially applications using faceted search and we present some good design practices.

Keywords: «Faceted search, Faceted Navigation, Faceted Browsing, Information Retrieval, Lucene.net, Apache Solr, dynamic taxonomy»

Η σελίδα αυτή είναι σκόπιμα λευκή.

Πίνακας περιεχομένων

1

Εισαγωγή

1.1 Information seeking

Η σημερινή εποχή που χαρακτηρίζεται από την τεράστια έκρηξη της ποσότητας της διαθέσιμης πληροφορίας που παράγεται παγκοσμίως σε πολλές διαφορετικές μορφές (κείμενο, βίντεο, εικόνες) κατέστησε περισσότερο από ποτέ επιτακτική την ανάγκη για προηγμένα συστήματα αναζήτησης πληροφορίας. Καταλυτικός παράγοντας για την ανάπτυξη νέων τεχνικών και συστημάτων αναζήτησης υπήρξε η ραγδαία διάδοση του διαδικτύου όπως και η πρόσφατη επικράτηση μικρών ισχυρών φορητών συσκευών με μόνιμη σύνδεση στο διαδίκτυο. Όλα αυτά οδήγησαν στην εξέλιξη του πεδίου αναζήτησης πληροφορίας (information retrieval, IR). Ήδη από τις προηγούμενες δεκαετίες είχαμε την εξέλιξη διαφόρων συστημάτων με πιο γνωστή την ανάπτυξη των διαφόρων μηχανών αναζήτησης στο διαδίκτυο. Σήμερα όλοι σχεδόν έχουμε αναζητήσει κάτι, είτε συνειδητά, δηλαδή χρησιμοποιώντας μία μηχανή αναζήτησης, είτε χωρίς να το γνωρίζουμε, όπως για παράδειγμα μέσω ενός συστήματος χαρτών.

Ωστόσο, ακόμα και τα σημερινά συστήματα αναζήτησης πάσχουν από το ίδιο σύμπτωμα των παλιών συστημάτων αναζήτησης σε βιβλιοθήκες, το λεγόμενο known-item-search (Spencer, 2006), είναι δηλαδή προσανατολισμένα στην αναζήτηση αντικειμένων τα οποία γνωρίζουμε από πριν και είμαστε σχεδόν βέβαιοι ότι υπάρχουν στη συλλογή των εγγράφων που αναζητούμε (π.χ. αναζήτηση στο εθνικό τυπογραφείο για ένα ΦΕΚ) ή στο Web. Όμως αυτή η

συνθήκη και παραδοχή δεν είναι πάντα η πλέον κατάλληλη για να περιγράψει και ικανοποιήσει τις ανάγκες μας.

Πιο συγκεκριμένα συχνά ψάχνουμε για κάτι το οποίο δεν γνωρίζουμε από πριν, αλλά έχουμε προσδιορίσει μόνο την ανάγκη μας. Για παράδειγμα, η αναζήτηση κατάλληλου εξοπλισμού για μια χειμερινή εξόρμηση μπορεί να καταλήξει στην επιλογή εξοπλισμού σκι ή ρούχων. Έτσι, εμπλεκόμαστε σε μία διαδικασία συνεχούς αναζήτησης εξερευνώντας και προχωρώντας βαθύτερα στα αποτελέσματα διαδοχικών αναζητήσεων.

Στο μοντέλο του κλασικού information retrieval ο χρήστης υποβάλει ένα ερώτημα, συνήθως με λέξεις κλειδιά, προς το σύστημα αναζήτησης, και αυτό ανταποκρίνεται με ένα πλήθος αποτελεσμάτων τα οποία ο χρήστης εξετάζει για να ικανοποιήσει την ανάγκη πληροφόρησής του. Η διαδικασία συνεχούς έρευνας και διαδοχικών αναζητήσεων ξεπερνάει το κλασικό information retrieval και εμπλέκει όλο και περισσότερο εμάς τους χρήστες, την τρέχουσα κατάστασή μας την κάθε στιγμή της αναζήτησης και τις επιλογές μας. Αυτό το διαδραστικό ταξίδι μεταξύ πληροφοριακής ανάγκης (information need) και γνώσης αναφέρεται ως αναζήτηση πληροφορίας (information seeking) και είναι ζητούμενο για την κάλυψη πολλών πληροφοριακών αναγκών σε σύγχρονα συστήματα στο διαδίκτυο όπου η αναζήτηση πληροφορίας είναι βασικό συστατικό στοιχείο (π.χ. ψηφιακές βιβλιοθήκες, e-shop κάθε είδους, αναζήτηση για παραγωγή γνώσης, αδόμητη πληροφορία με ταξινομικές ιδιότητες, π.χ. αγγελίες).

Ένα τέτοιο σύστημα αναζήτησης πληροφορίας πρέπει να δίνει τη δυνατότητα να διερευνήσουμε τις ταξινομικές επιλογές που έχουμε σε κάθε βήμα της διαδικασίας αναζήτησης αφού προηγουμένως τις έχει αξιολογήσει και τις έχει ομαδοποιήσει κατάλληλα, παρουσιάζοντάς τις σαν μία λίστα επιλογών. Αυτή η δυνατότητα δεν υπήρχε στα παλιότερα συστήματα αναζήτησης με αποτέλεσμα ο χρήστης να αναγκάζεται σε διαδοχικές αναζητήσεις, συχνά σε μεγάλο πλήθος αποτελεσμάτων και τελικά να έχει φτωχά αποτελέσματα.

Η διαπίστωση αυτής της αδυναμίας των συστημάτων αναζήτησης οδήγησε στην ανάπτυξη της τεχνολογίας του Faceted Search

1.2 Faceted Search

Στην ακαδημαϊκή κοινότητα ο όρος “faceted search” αρχικά εμφανίστηκε στην επιστήμη της βιβλιοθηκονομίας σύντομα όμως επεκτάθηκε στην πληροφορική και ειδικότερα στο πεδίο της ανάκτησης πληροφοριών. Με απλά λόγια πρόκειται για μία τεχνική που επιτρέπει την

πλοήγηση σε μία συλλογή πληροφοριών μέσω της εφαρμογής πολλαπλών ταξινομικών οργανώσεων και κατατμήσεων της συλλογής που αναζητούμε. Η πληροφορία είναι οργανωμένη σύμφωνα με ένα πολύπλευρο (faceted) σύστημα ταξινόμησης, το οποίο ταξινομεί ή κατατάσσει τα δεδομένα σε διάφορες σαφείς και διακριτές διαστάσεις, τα facets. Με αυτόν τον τρόπο οι ταξινομήσεις καθίστανται προσβάσιμες με πολλούς διαφορετικούς τρόπους αντί για έναν μέσω μίας προκαθορισμένης διάταξης (Tunkelang Daniel, 2009).

Το faceted search (η αλλιώς faceted navigation) είναι μια τεχνική με την οποία μπορούμε να έχουμε πρόσβαση στα αποτελέσματα μίας αρχικής αναζήτησης τα οποία προηγουμένως έχουν ταξινομηθεί και καταταχθεί ταυτόχρονα σε πολλές διαφορετικές κλάσεις ή αλλιώς όψεις, που εδώ βρίσκουν εφαρμογή με τη μορφή των facets. Για παράδειγμα, η αναζήτηση προϊόντων σε ένα e-commerce παράγει μία λίστα αποτελεσμάτων όπου κάθε προϊόν ταξινομείται ανάλογα με κάποια χαρακτηριστικά του όπως το χρώμα του, τον τύπο του ή τον κατασκευαστή του. Φυσικά αυτό προϋποθέτει ότι το σύστημα αναζήτησης διαθέτει ή ότι μπορεί δυναμικά να παράγει τέτοια facets για κάθε έγγραφο (information item) που διαθέτει προς αναζήτησης π.χ. ένα facet για το χρώμα, ένα για τον τύπο και ένα για τον κατασκευαστή.

Σε κάθε βήμα της διαδικασίας ο χρήστης μπορεί να επιλέξει την ομαδοποίηση (facet) που επιθυμεί, ένα ή περισσότερα συνδυαστικά, περιορίζοντας έτσι τα αποτελέσματα της αρχικής αναζήτησης και ταυτόχρονα εξερευνώντας τις νέες διαθέσιμες επιλογές του. Ξεκινάει έτσι μία συνεχής διαδικασία drill-down στα δεδομένα με σκοπό την ικανοποίηση της αρχικής ανάγκης ή όπως αυτή μεταβάλλεται ή εξελίσσεται κατά τη διάρκεια της αναζήτησης.

Το κάθε facet αναπαρίσταται με το όνομά του και με το πλήθος των εγγράφων που συμφωνούν με την επιλογή του (hitCount) ενώ μετά από κάθε ενέργεια τα ίδια τα facets και οι τιμές αυτές επαναϋπολογίζονται ώστε ο χρήστης να έχει στη διάθεσή μόνο τις επιλογές που θα τον οδηγήσουν σε νέα αποτελέσματα μηδενίζοντας έτσι την πιθανότητα για κενά αποτελέσματα (dead ends) στις αναζητήσεις.

Τα facets επιλέγονται κατάλληλα ώστε να βρουν εφαρμογή ανάλογα με τα έγγραφα της συλλογής. Οι τιμές των facet προέρχονται από τις ιδιότητες των εγγράφων που συχνά αναφέρονται ως μεταδεδομένα (metadata). Η διαδικασία αυτή μπορεί να εμπλέκει ανάλυση των κειμένων των εγγράφων αξιοποιώντας τεχνικές ανάκτησης οντοτήτων. Εάν τα έγγραφα βρίσκονται σε βάσεις δεδομένων η επιλογή γίνεται μεταξύ των πεδίων των πινάκων. Με τον τρόπο αυτό, ιστοσελίδες, online συλλογές από άρθρα ή προϊόντα ενός συστήματος e-commerce μπορούν να χρησιμοποιηθούν σαν facets.

Η αξία της faceted αναζήτησης εντοπίζεται στη δυνατότητα που παρέχει στο χρήστη να προσπελάσει ένα πλήθος επιλογών που σχηματίζονται δυναμικά και να καθορίσει μόνος του

τις παραμέτρους πλοήγησης, συνδυάζοντας τις διάφορες οπτικές που τον ενδιαφέρουν

Η αναζήτηση με όψεις η αλλιώς Faceted Search είναι μια σημαντική εξέλιξη και αποτελεί σήμερα κεντρικό στοιχείο συστημάτων όπως για παράδειγμα το ηλεκτρονικό εμπόριο. Το faceted search παρέχει μεγάλη ευελιξία (Ferre S., Hermann A., 2012) και είναι κατάλληλο τόσο για εφαρμογές query-based analytical search όσο και για εφαρμογές exploratory search. Ενεργεί προτείνοντας στον χρήστη περιορισμούς ή facets για τα τρέχοντα αποτελέσματα ώστε να φτάσει στο επιθυμητό αποτέλεσμα. Το Faceted Search είναι εύκολο στη χρήση του και ταυτόχρονα ασφαλές, υπό την έννοια ότι ο κάθε περιορισμός προτείνεται μόνο εφόσον έχει έστω και ένα πιθανό αποτέλεσμα αποκλείοντας έτσι τα κενά αποτελέσματα.

Στην διπλωματική αυτή εργασία θα ερευνήσουμε την τεχνική του faceted search, εξετάζοντας διαφορετικές όψεις του πεδίου όπως :

1. Την εξέλιξή του από τα παλιότερα συστήματα
2. Τη σχετική βιβλιογραφική έρευνα
3. Τα χαρακτηριστικά του συστήματος
4. Το μοντέλο δεδομένων
5. Το μοντέλο πλοήγησης
6. Το μοντέλο αλληλεπίδρασης με τον χρήστη
7. Τις επιλογές μας κατά την υλοποίηση του συστήματος
8. Τα προβλήματα κατά την εφαρμογή του
9. Τις προκλήσεις κατά τη σχεδίαση του συστήματος
10. Τη σύγκριση με άλλες σχετικές τεχνολογίες
11. Τα διαθέσιμα εργαλεία προγραμματισμού ή πλατφόρμες

Θα εξετάσουμε επίσης την αποτελεσματική σχεδίαση συστημάτων web διερευνώντας θέματα όπως:

1. Τους παράγοντες που επηρεάζουν την διαδικασία αναζήτησης
2. Τους διαφορετικούς τύπους χρηστών
3. Τα μοντέλα αναζήτησης πληροφορίας από τους χρήστες
4. Τα προβλήματα, τις καλές πρακτικές και τα συμπεράσματα για τη σχεδίαση των συστημάτων αναζήτησης

Τέλος, θα παρουσιάσουμε την ανάπτυξη μίας εφαρμογής που ενσωματώνει δυναμικά χαρακτηριστικά του faceted search με σκοπό να έχουμε ένα παράδειγμα υλοποίησης ενός συστήματος.

1.3 Αντικείμενο διπλωματικής

Ο σκοπός της παρούσης διπλωματικής εργασίας είναι η εξερεύνηση της τεχνολογίας του Faceted Search. Αρχόμενοι από τα προβλήματα των κλασικών συστημάτων ανάκτησης πληροφορίας οδηγούμαστε στα συστήματα αναζήτησης γνώσης. Εκεί εντάσσεται και το faceted search όντας μία διαδεδομένη τεχνική που έχει αποδείξει την αξία της σε πολλές εγκαταστάσεις από μικρές λύσεις όπως ένα e-commerce έως πολύ μεγάλες εφαρμογές όπως social networks.

Επιθυμούμε ο αναγνώστης να έχει κατανοήσει την τεχνική αυτή και να έχει αποκτήσει την ικανότητα να διακρίνει συστήματα που εφαρμόζουν το faceted search σε σχέση με άλλες αντίστοιχες τεχνικές όπως directory navigation, parametric search filtered search.

Επιχειρούμε επίσης να βάλουμε σε μία σειρά τις έννοιες που αφορούν στο information seeking η αλλιώς στην αλληλεπίδραση του χρήστη με ένα σύστημα αναζήτησης γνώσης. Η επιρροή του χρήστη στο σύστημα είναι ένας πολύ σημαντικός παράγοντας που συχνά δεν λαμβάνεται υπόψη. Οι παράγοντες που επηρεάζουν την αλληλεπίδραση ταξινομούνται και παρουσιάζονται σε σχέση με το Information retrieval στο web και το faceted search.

Προσπαθούμε επίσης να παρουσιάσουμε ταξινομημένα τα διαθέσιμα εργαλεία για την ανάπτυξη σχετικών εφαρμογών ή τις διαθέσιμες υπηρεσίες που προσφέρονται από τρίτους και υλοποιούν την τεχνικής του faceted search. Αναπτύσσουμε μια demo web εφαρμογή. Σκοπός της εφαρμογής είναι να δώσουμε ένα πρακτικό παράδειγμα αξιοποιώντας κάποια από τα εργαλεία που αναφέραμε στο προηγούμενο στάδιο

1.4 Τεχνολογίες που χρησιμοποιήθηκαν

Μία σημαντική απόφαση που έπρεπε να ληφθεί είναι η τεχνολογία στην οποία θα στηριχθεί η εφαρμογή μας. Αποφασίσθηκε να χρησιμοποιήσουμε την πλατφόρμα .Net της Microsoft διότι παρέχει ένα εξαιρετικό προγραμματιστικό περιβάλλον και μια πλήρη βιβλιοθήκη που ενσωματώνει χαρακτηριστικά ευελιξίας, επεκτασιμότητας αλλά και ασφάλειας. Άλλη μια απόφαση ήταν το είδος της εφαρμογής ως παραγόμενο αποτέλεσμα. Θέσαμε εξ αρχής ως πεδίο εφαρμογής το web.

Ένας βασικός πυλώνας της ανάπτυξης υπήρξε και η επιλογή του συστήματος information retrieval το οποίο θα έπρεπε να χρησιμοποιήσουμε. Αφού εξετάσαμε διάφορα συστήματα καταλήξαμε στην επιλογή του Lucene.Net ενός Port της βιβλιοθήκης αναζήτησης πληροφορίας Lucene, γραμμένο σε C#. Η επιλογή της C# αφορά σε developers που αξιοποιούν το περιβάλλον .Net.

Για την απόδειξη της καλής λειτουργίας της εφαρμογής μας εκτελέσαμε μία αξιολόγηση τύπου Formative evaluation με χρήστες που έχουν προγραμματιστικές γνώσεις από ανάπτυξης εφαρμογών για web με Visual Studio.Net.

Τέλος, επιλέξαμε η εφαρμογή να χρησιμοποιεί δεδομένα που βρίσκονται σε παραδοσιακά συστήματα διαχείρισης βάσεων δεδομένων (ΣΔΒΔ). Η επιλογή έγινε για δύο λόγους: Ένας αφορά στην απλότητα που εστιάζει στην πραγματική λειτουργικότητα που απαιτείται και ένας δεύτερος έχει να κάνει με το γεγονός ότι οι περισσότερες έτοιμες εφαρμογές που θα ήταν δεκτικές να αξιοποιήσουν το δικό μας σύστημα χρησιμοποιούν και αυτές ΣΔΒΔ. Στην πράξη η επιλογή αυτή δεν αφαιρεί κανένα ουσιαστικό χαρακτηριστικό γιατί το σύστημα ανάκτησης πληροφορίας Lucene.Net που επιλέξαμε μπορεί να αξιοποιήσει σαν πηγή πολλών ειδών δεδομένα που βρίσκονται σε δομημένη ή ημι-δομημένη μορφή.

1.5 Συνεισφορά

Η συνεισφορά της διπλωματικής αφορά στα εξής:

1. Φιλοδοξούμε να βάλλουμε σε μία τάξη τις βασικές έννοιες που αφορούν στην βαθιά κατανόηση που σχετίζονται με την διαδικασία της αναζήτησης πληροφορίας.
2. Διερευνήσαμε και παρουσιάζουμε αναλυτικά το μοντέλο λειτουργίας του faceted search.
3. Διερευνήσαμε τους παράγοντες που επηρεάζουν την διαδικασία της αναζήτησης πληροφορίας ειδικά στο web και κατά πόσο η σχεδίαση της εφαρμογής μπορεί να βοηθήσει αποτελεσματικά τον χρήστη.
4. Ερευνήσαμε τις εναλλακτικές τεχνολογικές λύσεις για την αναζήτηση πληροφορίας που χρησιμοποιούνται σήμερα και τις συγκρίναμε μεταξύ τους.
5. Διερευνήσαμε τα λογισμικά που διατίθενται σήμερα με ενσωματωμένη την αντίστοιχη λειτουργικότητα.
6. Αναπτύξαμε μία πειραματική εφαρμογή που έχει κατά τη γνώμη μας προοπτικές περαιτέρω ανάπτυξης διότι δεν υπάρχει αντίστοιχο προϊόν ευρέως διαθέσιμο στην αγορά.

7. Αξιολογήσαμε την λειτουργικότητα των faceted search συστημάτων σε διάφορες περιπτώσεις.

1.6 Οργάνωση κειμένου

Για την επίτευξη του στόχου η εργασία μας περιλαμβάνει τα εξής βασικά μέρη:

- Ένα εισαγωγικό μέρος με την τοποθέτηση του πεδίου που μας απασχολεί και την εισαγωγική παρουσίαση του faceted search.
- Στο πρώτο μέρος (κεφάλαιο 2) κάνουμε μία ιστορική προσέγγιση και βιβλιογραφική έρευνα στο πεδίο της αναζήτησης πληροφορίας σε συνδυασμό με το faceted search. Αναφέρουμε τις σημαντικότερες ερευνητικές προσπάθειες και μερικά από τα ακαδημαϊκά project που υλοποίησαν την τεχνική έχοντας διάφορες επιδιώξεις. Αναφέρουμε επίσης υπάρχουσες εφαρμογές προς απόδειξη της αξίας της τεχνικής
- Στο επόμενο κεφάλαιο εξετάζουμε αρχικά τις βασικές έννοιες της ανάκτησης πληροφορίας και εξερευνούμε τις προσπάθειες και τις πρώιμες τεχνικές πριν από το faceted search. Στη συνέχεια αναλύουμε το faceted search με τα κύρια χαρακτηριστικά του αλλά και επιχειρούμε να αποδώσουμε τον πυρήνα της τεχνικής, το θεωρητικό υπόβαθρο, να ξεκαθαρίσουμε τις διαφορετικές τεχνικές που προσεγγίζουν το ζήτημα και συχνά λανθασμένα παρουσιάζονται σαν faceted search.
- Στο κεφάλαιο 4 κάνουμε μια προσπάθεια καταγραφής, ανάλυσης και ταξινόμησης των κύριων ζητημάτων που αφορούν στην σχεδίαση ενός συστήματος information retrieval (IR) στο web. Μελετάμε τις βασικές έννοιες για το IR και άλλα θέματα όπως για παράδειγμα το μοντέλο αναζήτησης από τους χρήστες. Ένα σημαντικό μέρος της εργασίας αφορά στην κατάλληλη σχεδίαση ενός συστήματος αναζήτησης και στην ενσωμάτωση των χαρακτηριστικών του faceted search
- Στο κεφάλαιο 5 παρουσιάζουμε τα διαθέσιμα προγραμματιστικά εργαλεία και πλατφόρμες για την ανάπτυξη διαδικτυακών εφαρμογών αναζήτησης με faceted search.
- Στο κεφάλαιο 6 παρουσιάζουμε τις βασικές παραμέτρους ανάπτυξης της demo εφαρμογής όπως τις λειτουργικές απαιτήσεις της, τα προγραμματιστικά εργαλεία που επιλέχθηκαν και κομμάτια από τον κώδικα που αναπτύχθηκε.
- Στο κεφάλαιο 7 αναφέρουμε τα αποτελέσματα της αξιολόγησης καθώς και χρήσιμα συμπεράσματα από τη διαδικασία.
- Στο κεφάλαιο 8 καταλήγουμε σε συμπεράσματα και διατυπώνουμε προβληματισμούς για την μελλοντική χρήση της τεχνικής του faceted search.

2

Σχετικές εργασίες

2.1 Ιστορικά στοιχεία και έρευνα στο πεδίο του faceted search

Ο αρχαίος Έλληνας φιλόσοφος και πολυεπιστήμονας Αριστοτέλης άφησε πίσω του ένα πλούσιο συγγραφικό έργο που αναφέρεται σε διάφορα αντικείμενα όπως φιλοσοφία, βιολογία, ζωολογία, θέατρο, μουσική ρητορική και άλλα. Σε κάποια από αυτά (“Περί ζώων ιστορίας” και “Περί ζώων γενέσεως”) επιχειρεί να κατατάξει τον εν ζωή φυσικό κόσμο αναπτύσσοντας ένα δικό του σύστημα κατάταξης, “το δέντρο του Αριστοτέλη”, χωρίζοντας τους ζωντανούς οργανισμούς σε ομάδες όπως φυτά και ζώα και σε άλλες όπως αυτά που γεννούν αυγά και αυτά που γεννούν ζωντανά. Ο Αριστοτέλης υπήρξε λοιπόν ο πρώτος “ταξινομιστής” και η εργασία του υπήρξε παρόμοια με τη σημερινή ταξινόμηση δηλαδή την κατάταξη σε ομάδες και ιεραρχίες.

Σήμερα η ταξινόμηση αναφέρεται σε οποιοδήποτε ιεραρχικό σχήμα κατάταξης και στις βασικές έννοιες που διέπουν την διαδικασία. Η δενδροειδής μορφή είναι η συνηθέστερη απεικόνιση με γονείς και παιδιά ενώ υπάρχουν και πιο πολύπλοκα σχήματα όπου ένα παιδί έχει πολλούς γονείς (πολύ-ιεραρχικό σχήμα). Οι κόμβοι είναι οι διαφορετικές περιπτώσεις εννοιών που κατατάσσονται στην ιεραρχία. Όλο το οικοδόμημα έχει νόημα γιατί αναπαριστά μία λογική σειρά στη ανθρώπινη γνώση την ταξινομεί ώστε να υπάρχει πάντα τουλάχιστον ένα μονοπάτι από τον κόμβο ρίζα (root) προς κάθε άλλον κόμβο.

Το σύστημα ταξινόμησης του Αριστοτέλη κράτησε σχεδόν 2000 χρόνια έως ότου ο Melvil Dewey ανάπτυξε το Dewey Decimal Classification (DDC) που ακόμα και τώρα χρησιμοποιείται σε βιβλιοθήκες. Το σύστημα αυτό χρησιμοποιούσε ένα πλήθος από δυαδικά bits που το κάθε ένα αναφερόταν σε κλαδιά στην ιεραρχία. Το σύστημα αυτό όμως έχει πολλά προβλήματα αντιστοίχισης στον πραγματικό κόσμο. Για παράδειγμα η ίδια έννοια συναντάται σε πολλά κλαδιά, επίσης ο μόνος τρόπος προσθήκης νέας γνώσης είναι η προσθήκη κάτω από κάποια υπάρχουσα. Τα προβλήματα αυτά εντόπισε ο Ινδός μαθηματικός S.R. Ranganathan στο έργο του για την επιστήμη της βιβλιοθηκονομίας "Philosophy of Library Classification" και προτείνοντας την παρεμβολή και επέκταση ως βασικές ενέργειες στις ιεραρχίες γνώσης.

Μια πιο σύγχρονη προσέγγιση στρέφεται στη χρήση οντολογιών αντί των κλασικών ιεραρχιών γνώσης. Η οντολογία είναι η μελέτη της ύπαρξης για οτιδήποτε αλλά στην πληροφορική έχει πάρει μια πιο πρακτική έννοια και αναφέρεται στην αναπαράσταση οποιασδήποτε έννοιες και στις σχέσεις που έχει με άλλες οντολογίες. Στην πραγματικότητα οι οντολογίες μπορεί να συνδέονται με πιο σύνθετες σχέσεις που εμπλέκουν περισσότερες από δύο οντολογίες ταυτόχρονα όπως για παράδειγμα στη φράση "Ο Οδυσσέας Ελύτης διακρίθηκε με το Νόμπελ λογοτεχνίας το 1979". Το 2001 το W3C δημιούργησε την ομάδα Web Ontology Working Group με σκοπό την ανάπτυξη της OWL, μιας περιγραφικής γλώσσας για οντολογίες. Η ταξινομία (taxonomy) είναι μια ειδική περίπτωση οντολογιών στην οποία οι σχέσεις είναι αποκλειστικά τύπου "is-a" και συνδέουν γονείς με παιδιά.

Η ακαδημαϊκή έρευνα στον τομέα ξεκινάει αρκετά παλιά και είναι κάπως δύσκολο να εντοπίσει κανείς τις πρώτες δημοσιεύσεις. Σίγουρα μία από τις αυτές αφορούσε στο αντικείμενο του faceted navigation και ήταν αυτή του University Of Maryland "Dynamic Queries for Visual Information Seeking" (Shneiderman Ben, 1994). Σε αυτή ορίζονται τα δυναμικά ερωτήματα ως μέρος του διαλόγου του χρήστη με ένα σύστημα αναζήτησης και τονίζεται η διαφορά ενός τέτοιου συστήματος από τα έως τότε υπάρχοντα βασιζόμενα σε SQL. Τα οπτικοποιημένα χαρακτηριστικά του βοηθούν τον άπειρο χρήστη ειδικά σε σχέση με τα συστήματα command line και επιτρέπουν την άμεση ανατροφοδότηση των αποτελεσμάτων.

Η εργασία των Pollitt, A. Steven; Smith, Martin P.; Treglown, Mark; Braekevelt, Patrick με τίτλο "View-Based Searching Systems--Progress Towards Effective Disintermediation" (Pollitt, A. Steven; Smith, Martin P.; Treglown, Mark; Braekevelt, Patrick, 1996) πρότεινε ένα σύστημα για αποτελεσματική αναζήτηση χρηστών σε συστήματα βάσεων δεδομένων με την χρήση ενός συστήματος κατάταξης (classification) που παρήγαγε συνδέσμους για την αναζήτηση του χρήστη κάνοντας εφαρμογή σε μεταδεδομένα που υπάρχουν σε βιοιατρικά έγγραφα.

Στο University of North Carolina αναπτύχθηκε ένα project με όνομα Relation Browser με σκοπό την βελτίωση των μηχανισμών αναζήτησης και πλοήγησης χρησιμοποιώντας ένα σύστημα preview που επιτρέπει στους χρήστες την εξερεύνηση ενός χώρου εγγράφων με δυναμικά ερωτήματα και facets.

Ένα ακόμα ακαδημαϊκό Project υπήρξε το Parallax του MIT και μετέπειτα της MetaWeb[1], το οποίο ήταν ένα σύστημα ακαδημαϊκής ανταλλαγής γνώσης. Το σύστημα επέκτεινε το faceted search επιτρέποντας την μετάβαση σε διαφορετικές όψεις αξιοποιώντας μια σημασιολογική προσέγγιση. Το 2010 η MetaWeb απορροφήθηκε από την Google.

Από τις πρώτες σημαντικές εργασίες είναι αυτή των Hearst et al (Hearst et al, 2002) με τίτλο "Finding the Flow in Web Site Search". Όρισαν το faceted search και προχώρησαν στον ορισμό των hierarchical faceted metadata και στη χρήση τους για την κατάταξη των δεδομένων σε τάξεις με πολλές διαστάσεις.

Μία πολύ σημαντική ερευνητική εργασία έγινε από τους Sacco G, Tzitzikas Y., Ferre S (Sacco G., Tzitzikas Y., 2009) με τίτλο "Dynamic Taxonomies and Faceted Search". Στην εργασία αυτή γίνεται παρουσίαση του σημερινού μοντέλου αναζήτησης στον ιστό, παρουσιάζεται η έννοια του multidimensional taxonomy και ενός faceted search system και γίνεται σύγκριση με άλλες αντίστοιχες τεχνολογίες αναζήτησης αλλά και σχεδιαστικές αναζητήσεις-προτάσεις για την ανάπτυξη συστημάτων. Σε αυτήν παρουσιάζεται το θεωρητικό μαθηματικό μοντέλο αλλά και προτάσεις για την υλοποίηση ενός συστήματος faceted search.

Το 2009 επίσης δημοσιεύτηκε η εργασία του Daniel Tunkelang με τίτλο "Synthesis Lectures on Information Concepts, Retrieval, and Services" (Tunkelang, 2009) η οποία συνοψίζει πολύ αποτελεσματικά την εισαγωγή στο faceted search και εντοπίζει και επεξεργάζεται ζητήματα όπως σχεδιαστικά αλλά και απόδοσης του συστήματος.

Την αμεσότητα και την ευελιξία ενός συστήματος faceted search και την εκφραστικότητα των γλωσσών σημασιολογικού ιστού όπως η OWL επιχειρεί να συμβιβάσει η έρευνα των Ferre, Herman (Ferre S., Herman A., 2012) οι οποίοι πρότειναν ένα μοντέλο αναζήτησης σε σημασιολογικά δεδομένα. Το σύστημά τους με ονομασία Query-Based Faceted Search (QFS) προτείνεται ως εύκολο στη χρήση και εξίσου ασφαλές στα αποτελέσματα όπως τα faceted navigation συστήματα, αλλά ταυτόχρονα εκφραστικό όπως η SPARQL και η OWL-QL. Σε συνέχεια αυτής της εργασίας το 2013 δημιουργήθηκε από την ομάδα του Ferre S. το SPARKLIS. Πρόκειται για ένα σύστημα ερωτήσεων και απαντήσεων που ενσωματώνει τα χαρακτηριστικά του QFS και αντλεί τα δεδομένα του από την DBpedia.

[1] Η MetaWeb ανέπτυξε το FreeBase

Μία ακόμα ενδιαφέρουσα δημοσίευση αφορά στη ανάπτυξη ενός συστήματος αναζήτησης Entities το οποίο αντί να επιστρέφει συνδέσμους για ιστοσελίδες επιστρέφει entities όπως και τις συσχετίσεις μεταξύ τους (Fafalios, P., Kitsos, I., Marketakis, Y., Baldassarre, C., Salampasis, M., & Tzitzikas, Y., 2012). Σύμφωνα με τους συγγραφείς ένα τέτοιο σύστημα μπορεί επίσης να αντλεί τα μεταδεδομένα του που χρησιμοποιούνται για faceted navigation σε πραγματικό χρόνο είτε από τα snippets αποτελεσμάτων που επιστρέφει ένα κλασσικό σύστημα αναζήτησης είτε από τα πραγματικά έγγραφα δεδομένων. Η έρευνα οδήγησε στην ανάπτυξη ενός πειραματικού συστήματος αναζήτησης με τον τίτλο X-Search[2] που υλοποιεί ανάκτηση πραγματικού χρόνου σε entities.

Μια ολοκληρωμένη εργασία, είναι η έκδοση των Tony Russell και Tyler Tate. Στην εργασία αυτή κεντρικό ρόλο έχουν οι χρήστες και οι ιδιαιτερότητες του κάθε ενός. Πρόκειται για ένα σημαντικό έργο το οποίο παρουσιάζει τα διάφορα μοντέλα αναζήτησης και τις ιδιαιτερότητές τους και προτείνει αναλυτικά σχεδιαστικές λύσεις για τα σύγχρονα συστήματα αναζήτησης. Σε αυτή γίνεται επίσης ιδιαίτερη παρουσίαση για τη σχεδίαση συστημάτων με χαρακτηριστικά faceted search και ακόμα θέματα όπως η κατάλληλη σχεδίαση για κινητές συσκευές αλλά και αναζητήσεις σε κοινωνικά δίκτυα.

Μία ακόμα ενδιαφέρουσα ερευνητική εργασία (Gomadam, K., Ranabahu, A., Nagarajan, M., Sheth, A. P., & Verma, K., 2008) αφορά στην αναζήτηση και ταξινόμηση των αποτελεσμάτων σε Web APIs. Η χρήση των APIs είναι κοινή πρακτική σε web εφαρμογές ενώ σύγχρονες εφαρμογές έχουν πρόσβαση σε δύο η περισσότερα services (mashup apps). Υπάρχουν διαθέσιμα API directory όπως το ProgrammableWeb, ωστόσο δεν υπάρχουν εξειδικευμένα εργαλεία αναζήτησης σε αυτά και συνήθως χρησιμοποιούνται οι κλασικές μηχανές αναζήτησης. Στην εργασία προτείνεται ένα μοντέλο faceted search που κατηγοριοποιεί τα Web APIs ανάλογα με την περιγραφή τους. Η υλοποίηση περιλαμβάνει τέσσερα στάδια, Defining Facets for Web API search, Classification of APIs using Facets, Searching και Ranking

Ένα ακόμα ακαδημαϊκό project το Serendipity[3] που αναπτύχθηκε από το Universidad Politécnica de Madrid και το Universidad Técnica Particular de Loja (Ecuador) χρησιμοποιεί open source linked data και υλοποιεί ένα σύστημα αναζήτησης δεδομένων με faceted search για την ανεύρεση ανοικτού ακαδημαϊκού υλικού (opencourseware).

[2] http://www.ics.forth.gr/isl/X-Search/

[3] http://serendipity.utpl.edu.ec/index.html

2.2 Χρήση του faceted search σε εφαρμογές Web

Η αξία του faceted search και οι σημαντικές προοπτικές του στο πεδίο της αναζήτησης πληροφορίας αναγνωρίστηκαν γρήγορα από τη βιομηχανία πληροφορικής. Πολλές εφαρμογές ενσωμάτωσαν τα χαρακτηριστικά του και η εμπορική αξιοποίηση του ξεκίνησε αρκετά νωρίς. Σήμερα πολλές μεγάλες εγκαταστάσεις αξιοποιούν την τεχνική αυτή ενώ ακόμα και για μικρότερες όπως το e-commerce, το faceted search είναι η de facto λύση.

Αναφέρουμε μερικά παραδείγματα:

2.2.1 Endeca

Μία από τις πρώτες εφαρμογές που ενσωμάτωσε το faceted search ήταν αυτή της Endeca. Η εταιρεία, παρείχε υπηρεσίες εφαρμογών αναζήτησης στους πελάτες της που κυρίως ήταν πάροχοι ηλεκτρονικού εμπορίου. Το προϊόν της με εμπορικό όνομα "Guided Navigation" ενσωμάτωνε χαρακτηριστικά faceted search. Η σημαντική ειδίκευση της Endeca ήταν η διαχείριση δεδομένων σε αδόμητη μορφή στα οποία εφάρμοζε τεχνικές entity extraction. Από το 2011 η Endeca απορροφήθηκε από την Oracle.

2.2.2 Amazon

Το Amazon ξεκίνησε την πειραματική εφαρμογή faceted search μέσω ενός δικού του project από το 2002 με το όνομα "Project Ruby" το οποίο επέτρεπε την επισκόπηση προϊόντων βάση χαρακτηριστικών, κατηγορίας κτλ. Δημιούργησε την αρχική έκδοση του συστήματος faceted search και έκτοτε το εξέλιξε και το συμπεριέλαβε σαν βασική τεχνική για την αναζήτηση προϊόντων από τους χρήστες του. Το Amazon επίσης προσφέρει το Amazon CloudSearch, μία cloud based υπηρεσία αναζήτησης.

2.2.3 Ebay

Το eBay ενσωμάτωσε χαρακτηριστικά faceted search από ένα παλιότερο δικό του project που αναπτύχθηκε το 2006 με τίτλο "Ebay Express" που είχε στόχο την εφαρμογή ηλεκτρονικού εμπορίου. Σήμερα χρησιμοποιεί εκτενώς χαρακτηριστικά της τεχνικής αυτής για τις πολυπληθείς αναζητήσεις των χρηστών.

2.2.4 CNET

Το CNET είναι μία ιδιαίτερη περίπτωση γιατί όχι απλά εφάρμοσε faceted search στην αναζήτησή του αλλά επίσης από το 2006 συνέβαλε στην ανάπτυξη του Solr. Αυτό το ανοικτό λογισμικό (open source) βελτίωνε αρκετά χαρακτηριστικά του open source συστήματος

αναζήτησης Lucene και σήμερα είναι ευρέως διαδεδομένο. Μία από τις βελτιώσεις του ήταν η εφαρμογή faceted search χαρακτηριστικών.

2.2.5 LinkedIn

Το Linkedin είναι το μεγαλύτερο κοινωνικό δίκτυο προσανατολισμένο σε επαγγελματίες. Τα στατιστικά του είναι εντυπωσιακά[4]. Τον Νοέμβριο του 2015 είχε περίπου 400 εκατομμύρια χρήστες, με ρυθμό αύξησης δύο χρήστες ανά δευτερόλεπτο. Μια βασική λειτουργία της υπηρεσίας είναι η ανεύρεση χρηστών που ταιριάζουν σε κάποιο προφίλ. Ωστόσο, η αναζήτηση στο linkedin δεν είναι απλή υπόθεση: ο κάθε χρήστης μπορεί να υποβάλει το ερώτημά του στο οποίο πρέπει να συμπεριλάβει τα δικά του ιδιαίτερα χαρακτηριστικά βασιζόμενα στο προφίλ του τα οποία θα επηρεάσουν και την κατάταξη (scoring) των αποτελεσμάτων. Ο τεράστιος αριθμός χρηστών έκανε επιτακτική την ανάγκη ανάπτυξης κάποιας σύγχρονης λύσης. Η λύση αυτή ήρθε με την υλοποίηση μίας εφαρμογής που στηρίζεται σε 3 αντικείμενα:

- Στη μηχανή αναζήτησης Lucene
- Στη βιβλιοθήκη BoboBrowse
- Στη βιβλιοθήκη Zoie

Το Linkedin χρησιμοποιεί facets για την αναζήτηση προφίλ χρηστών το στην αρχική του σελίδα. Η αναζήτηση περιλαμβάνει facets για τον τύπο της αναζήτησης, τοποθεσία, εταιρία, βιομηχανία, εμπειρία και είναι πλήρως προσαρμόσιμη.

[4] http://expandedramblings.com/index.php/by-the-numbers-a-few-important-linkedin-stats/

3

Ανάκτηση πληροφορίας και faceted search

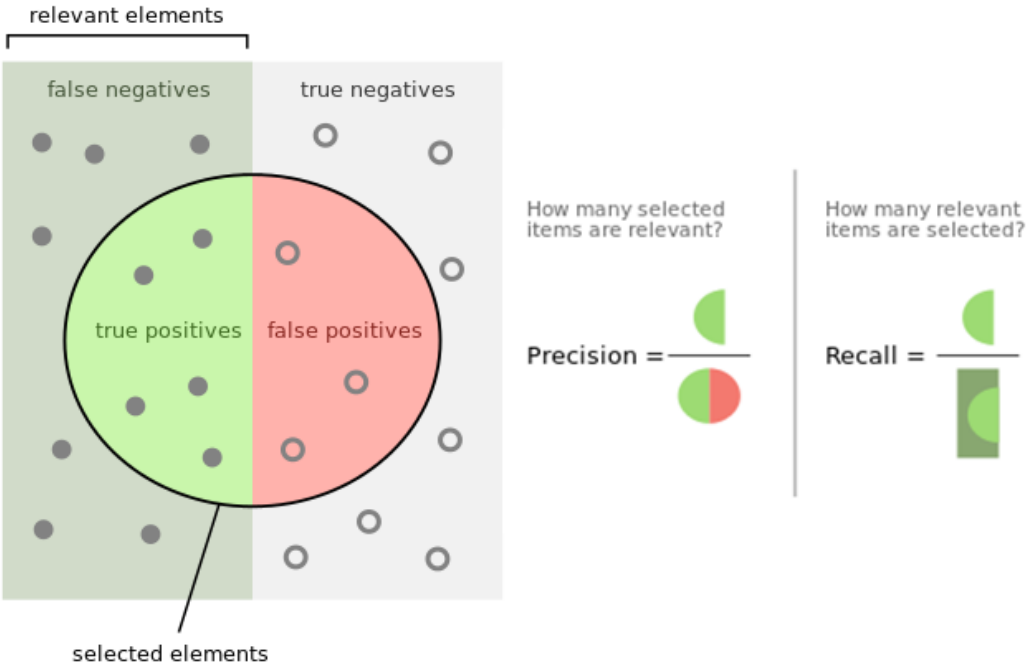
3.1 Βασικές έννοιες για την Ανάκτηση Πληροφορίας

Η ανάκτηση πληροφορίας (information retrieval ή IR) είναι η διαδικασία ανάκτησης πόρων (εγγράφων) σχετικών με κάποια ανάγκη πληροφόρησης του χρήστη. Οι πόροι συλλέγονται από μία συλλογή διαθέσιμων εγγράφων (collection) ή από το web και μπορεί να βρίσκονται σε δομημένη ή, συνηθέστερα, σε ημι-δομημένη μορφή.

Η πιο συνηθισμένη εικόνα για την αναζήτηση πληροφορίας είναι ο εντοπισμός σχετικών με το ερώτημά μας εγγράφων σε μία συλλογή πολλών εγγράφων. Η σχετικότητα είναι η πιο γνωστή μετρική που χρησιμοποιούμε για να συλλέξουμε τα αποτελέσματα ωστόσο δεν είναι εύκολο να δοθεί σαφής ορισμός. Θα λέγαμε ότι η σχετικότητα ενός εγγράφου προς ένα ερώτημα είναι η απαραίτητη πληροφορία που αντιστοιχεί το έγγραφο αυτό στο ερώτημα του χρήστη τουλάχιστον ως προς ένα βαθμό. Η συσχέτιση μπορεί να είναι χρηστο-κεντρική ή μια διαδικασία τύπου benchmark με διαφορές στην αξιολόγηση των συστημάτων αναζήτησης.

Δύο σημαντικές μετρικές είναι η ακρίβεια (**precision)** και η ανάκληση (**recall)**. Ας δούμε τους όρους αυτούς με ένα παράδειγμα: Έστω ότι έχουμε ένα δεδομένο ερώτημα και θα αναζητήσουμε σε μία σε μια δεδομένη συλλογή. Η αναζήτηση σχετικών εγγράφων στην συλλογή μας επιστρέφει κάποια έγγραφα που είναι πράγματι σχετικά (true positive) και κάποια που αξιολογήθηκαν σχετικά αλλά στην πραγματικότητα δεν είναι (false positive).

Επίσης κάποια έγγραφα αξιολογήθηκαν ορθά σαν μη σχετικά (true negative) ενώ κάποια άλλα λανθασμένα σαν μη σχετικά (false negative).



Εικόνα 1: Precision και Recall

Έτσι, για δεδομένο ερώτημα το precision είναι το κλάσμα των ανακτηθέντων εγγράφων σε σχέση με τα έγγραφα που αξιολογήθηκαν σαν σχετικά δηλαδή TP / (TP + FP) εκφράζει δηλαδή την ακρίβεια των αποτελεσμάτων. Εάν δεν έχουμε κανένα FP το precision είναι 1.

Αντίστοιχα το recall είναι το κλάσμα των ανακτηθέντων εγγράφων σε σχέση με τα πραγματικά σχετικά έγγραφα της συλλογής μας δηλαδή TP / (TP+FN). Εκφράζει την δυνατότητα του συστήματος να συμπεριλάβει όλα τα ορθά σχετικά έγγραφα. Αν δεν έχουμε κανένα FN το recall θα είναι 1.

Στην ιδανική περίπτωση ένα σύστημα ανάκτησης θα διατηρεί υψηλά τα επίπεδα ανάκτησης και ακρίβειας αλλά αυτό δεν είναι πάντα εύκολο ειδικά στα Boolean συστήματα ανάκτησης. Στην πράξη τα precision και recall αποτυγχάνουν να μας δώσουν σωστή εικόνα για τα αποτελέσματα όταν δεν χρησιμοποιούνται μαζί. Εναλλακτικά το F-Score δηλαδή ο αρμονικός μέσος δίνει καλύτερα αποτελέσματα. Το F-Score είναι το κλάσμα 2*(precision*recall)/(precision+recall) και συνδυάζει σε μία μετρική τα αποτελέσματα των άλλων δύο.

Ένα μοντέλο αναζήτησης που συσχετίζει τα έγγραφα με το ερώτημα χρησιμοποιώντας δυαδικές πράξεις είναι το Boolean Search μοντέλο αναζήτησης. Στην βασική μορφή του τα αποτελέσματα δεν κατατάσσονται (ranking) και είναι δύσκολο να αξιοποιηθεί σε σύγχρονα

συστήματα αναζήτησης γιατί σε αυτό δεν ισορροπούν σωστά μεταξύ precision και recall. Ωστόσο, ακόμα και σήμερα χρησιμοποιείται συμπληρωματικά, όταν σε ένα έτοιμο σετ αποτελεσμάτων θέλουμε να περιορίσουμε τα αποτελέσματα ακόμα περισσότερο.

$$Precision = \frac{TruePositive}{TruePositive + FalsePositive}$$

$$Recall = \frac{TruePositive}{TruePositive + FalseNegative}$$

$$F - Score = 2 \times \frac{Precision \times Recall}{Precision + Recall}$$

Εικόνα 2: Precision, Recall και Fscore

Η εναλλακτική πρόταση στο boolean μοντέλο είναι το ranked retrieval. Σε αυτό το μοντέλο χρησιμοποιούνται λέξεις κλειδιά που εξάγονται από ένα ελεύθερο κείμενο ερωτήματος αντί δυαδικών τελεστών, ενώ παράλληλα, επιχειρείται να δοθεί βαρύτητα στα αποτελέσματα με σκοπό την κατάταξή τους βάσει της ορθότητάς τους. Αυτό έχει θετικά αποτελέσματα στη λειτουργία του συστήματος ειδικά για τους απλούς χρήστες. Στον αντίποδα, ένα τέτοιο σύστημα μπορεί να επιστρέφει υπερβολικά μεγάλο αριθμό εγγράφων που πρέπει να εξετασθούν. Όμως η κατάταξη βοηθάει τη διαδικασία γιατί τα πρώτα αποτελέσματα είναι πολύ πιο σχετικά από τα υπόλοιπα.

Εύλογα το ερώτημα που προκύπτει είναι πως ένα σύστημα αποδίδει ορθά την βαρύτητα στα αποτελέσματα. Η ανάλυση των μετρικών συσχέτισης ξεπερνάει τους σκοπούς του παρόντος εγγράφου αλλά ο όρος TF-IDF (Term Frequency-Inverse Document Frequency) που διατυπώθηκε από την Karen Spärck Jones (Karen Spärck Jones, 1973) είναι μάλλον η πλέον κατάλληλη μετρική για την απόδοση βαρύτητας. Στην πράξη είναι μία στατιστική προσέγγιση για την απόδοση βαρύτητας στην οποία το πρώτο μέρος (TF) αφορά στην συχνότητα εμφάνισης ενός όρου στη συλλογή μας ενώ το δεύτερο μέρος (IDF) αποδίδει υψηλότερη βαρύτητα σε σπάνιους όρους σε σχέση με αυτούς που εμφανίζονται συχνά.

Το Πρόγραμμα Text Retrieval Conference (TREC)[5] έχει ως σκοπό την υποστήριξη της έρευνας στο πεδίο των συστημάτων αναζήτησης πληροφορίας σε μεγάλες βάσεις

[5] http://trec.nist.gov/

πληροφοριών. Ειδικοί σκοποί του είναι η αύξηση της επικοινωνίας μεταξύ των εμπλεκομένων μερών όπως δημόσιοι οργανισμοί και επιχειρήσεις, αλλά και η επιτάχυνση της μεταφοράς τεχνογνωσίας. Το TREC έχει δημοσιεύσει αρκετές συλλογές ελέγχου (test collections) εγγράφων με σκοπό την εφαρμογή σε αυτές και την αξιολόγηση διαφορετικών συστημάτων αναζήτησης.

Η ανάκτηση πληροφορίας πρέπει να αντιμετωπισθεί σαν ένα μέρος της γενικότερης διαδικασίας που λέγεται αναζήτηση πληροφορίας (Information Seeking). Είναι μια γενικότερη δυναμική διαδικασία που περιλαμβάνει πολλά στάδια αλλά σημαντικός παράγοντας είναι ο ίδιος ο χρήστης. Η αίσθηση που αποκομίζει ο χρήστης κατά τη διαδικασία, η ανατροφοδότησή του οι αποφάσεις και οι διαδοχικές ενέργειές τους προκειμένου να φτάσει στο αποτέλεσμα είναι σημαντικά στοιχεία του information seeking. Διαφορετικά, είναι μια διαδικασία που περιλαμβάνει όλες τις παραπάνω ενέργειες αλλά όχι μόνο από τεχνική πλευρά.

3.2 To exploratory search

Θα λέγαμε ότι έχουμε γενικά δύο τύπους αναζήτησης:

Ο πρώτος τύπος αφορά στην αναζήτηση ανάκτησης (analytical query-based search) όπου ο χρήστης αναζητά πληροφορίες σχετικές με λέξεις κλειδιά ενώ το αντικείμενο της αναζήτησης είναι συνήθως γνωστό (known item search). Κλασικό παράδειγμα είναι η λειτουργία των γνωστών σελίδων αναζήτησης στο διαδίκτυο όπως τα Bing και Google.

Στον δεύτερο τύπο αναζήτησης που αναφέρεται ως αναζήτηση εξερεύνησης (exploratory search) οι χρήστες δεν είναι απαραίτητο να έχουν εκ των προτέρων γνώση για αυτό που αναζητούν αλλά το σύστημα επιτρέπει την διαδοχική πλοήγηση στα αποτελέσματα. Λειτουργεί παρόμοια με το web όπου οι χρήστες πλοηγούνται από σελίδα σε σελίδα μέσω συνδέσμων και προελαύνουν την πληροφορία για να ικανοποιήσουν κάποια ανάγκη τους. Εδώ, οι στόχοι του χρήστη αλλά και τα διαθέσιμα μέσα δεν είναι σίγουρο ότι έχουν ορισθεί με ακρίβεια. Αντί αυτών, το σύστημα αναλαμβάνει να καθοδηγήσει τον χρήστη βοηθώντας τον με την συνεχή αποκάλυψη των διαθέσιμων δεδομένων.

Ίσως το πιο κατάλληλο παράδειγμα για exploratory search είναι αναζητήσεις του τύπου "βρες το κατάλληλο αντικείμενο" οι οποίες είναι πολύ συνηθισμένες σε εφαρμογές e-commerce. Σε αυτές πολλά αντικείμενα κάνουν για την δουλειά αλλά χρειάζεται να διαλέξουμε το πιο κατάλληλο βάση των χαρακτηριστικών του. Ο χρήστης πρέπει να μπορεί να βρει σχετικά

εύκολα όλα τα διαθέσιμα χαρακτηριστικά, να εστιάσει σε αυτά που θεωρεί πιο σημαντικά και να εξερευνήσει τις επιλογές που έχει.

Για να είναι αποτελεσματική η διαδικασία αναζήτησης ο χρήστης πρέπει να μπορεί (Sacco G, Tzitzikas Y., 2009):

- Να εντοπίσει γρήγορα όλα τα σχετικά με την αναζήτηση αντικείμενα. Το πλήθος τους ίσως είναι απαγορευτικό για αναλυτική εξέταση και θα πρέπει να οργανωθούν συστηματικά. Η ταξινόμησή τους είναι απαραίτητη.
- Να εστιάσει ελεύθερα στα πιο σημαντικά για αυτόν χαρακτηριστικά. Μπορεί να είναι ένα τεχνικό χαρακτηριστικό ενός προϊόντος ή η τιμή του.
- Να εξερευνήσει όλα τα χαρακτηριστικά που σχετίζονται με την επιλογή του. Αν το σύστημα δεν τα δίνει αυτόματα θα πρέπει να τα καταγράψει μόνος του. Αν τα έχει στη διάθεσή του μπορεί να επιλέξει το επόμενο σημαντικό χαρακτηριστικό ώστε να περιορίσει περισσότερο τα αποτελέσματά του.

Βέβαια δεν είναι σε όλες τις διαδικασίες exploratory search τόσο σημαντική η επιλογή χαρακτηριστικών. Σε κάποιες άλλες όπως για παράδειγμα στην αναζήτηση καλών πρακτικών για λήψη φωτογραφιών η εξερεύνηση σχετίζεται περισσότερο με την αναζήτηση γνώσης στην οποία η διαδικασία είναι πιο σημαντική από το τελικό αποτέλεσμα. Το θέμα αυτό προσέγγισαν οι White and Roth (White and Roth, 2009) οι οποίοι αντιγράφοντας τον Καβάφη υποστήριξαν ότι "στο ταξίδι αναζήτησης της γνώσης το ταξίδι είναι εξίσου σημαντικό όσο και το αποτέλεσμα".

3.3 Πριν από το faceted search

Μέχρι τώρα έχουμε δώσει μια εικόνα για το τι θα πρέπει να περιλαμβάνει ένα σύστημα faceted search. Πριν μελετήσουμε αναλυτικά τα χαρακτηριστικά του βλέπουμε σε συντομία διαφορετικές τεχνικές που προσεγγίζουν την ίδια λογική για exploratory search οι οποίες όμως δεν πρέπει να θεωρούνται σαν πλήρη faceted search συστήματα. Συγκεκριμένα θα εξετάσουμε την πλοήγηση σε στατικούς καταλόγους (directory navigation) που ήταν η πρώτη προσπάθεια για απλοϊκή ταξινόμηση των εγγράφων της συλλογής μας, την παραμετρική αναζήτηση (parametric search) που δίνει βαρύτητα σε αναζήτηση με Boolean τελεστές και το πρώιμο faceted navigation (Tunkelang Daniel, 2009).

3.3.1 Directory Navigation

Μια από τις πιο απλές τεχνικές για την οργάνωση των εγγράφων είναι η χρήση ιεραρχιών αποτελούμενων από κατηγορίες δομημένες μεταξύ τους έτσι ώστε να σχηματίζεται ένας

κατάλογος. Μια τέτοια κατηγοριοποίηση αναφέρεται και σαν ταξινομία (taxonomy). Με αυτή την οργάνωση οι χρήστες μπορούν σχετικά εύκολα να προσδιορίσουν το πεδίο ενδιαφέροντός τους και να εντοπίσουν σχετικά έγγραφα. Ενδιαφέρον σημείο αυτής της τεχνικής είναι ο συνδυασμός της με αναζήτηση ελεύθερου κειμένου (free text) διότι επιτρέπει στον χρήστη την αναζήτηση όχι στη πλήρη συλλογή αλλά σε υποσύνολα αυτής. Το Yahoo Directory ήταν μία σχετική υλοποίηση. Το προβληματικό στοιχείο σε αυτή την υλοποίηση είναι ότι οι κατάλογοι είναι στατικοί και ο χρήστης σπάνια είχε την ίδια εικόνα κατάταξης όπως αυτή του καταλόγου.

Οι κατάλογοι εξακολουθούν να χρησιμοποιούνται συνήθως όμως συμπληρωματικά με άλλες τεχνικές και μπορεί να είναι χρήσιμη βοήθεια ειδικά στην έναρξη της αναζήτησης από τον χρήστη

3.3.2 Parametric search

Ένα σύστημα παραμετρικής αναζήτησης είναι αυτό που επιτρέπει την boolean αναζήτηση χρησιμοποιώντας μια διεπαφή με τον χρήστη. Τα ερωτήματα επαναδιατυπώνονται με ένα πλήθος από περιορισμούς που εφαρμόζονται ή καταργούνται. Τα facets είναι οι περιορισμοί που εφαρμόζονται σε κάθε χαρακτηριστικό και επιλέγονται από μία λίστα. Για αυτά που έχουν διακριτές τιμές είναι εύκολος ο τρόπος επιλογής τους με μία λίστα με επιλογές. Για αυτά με συνεχείς τιμές χρησιμοποιείται συνήθως ένα range selector ή κάτι αντίστοιχο. Το σημαντικότερο στοιχείο είναι ότι στα παραμετρικά συστήματα αναζήτησης υλοποιείται boolean αναζήτηση με τη μορφή facets σε έγγραφα κειμένου. Τα συστήματα αυτά πάσχουν από εκφραστικότητα και είτε επιστρέφουν υπερβολικά πολλά αποτελέσματα είτε κανένα (“million or none”).

3.3.3 Early faceted navigation

Το πρώιμο faceted navigation συμπληρώνει ένα κομμάτι που λείπει από τα παραμετρικά ερωτήματα. Αυτό είναι η καθοδήγηση. Μετά από κάθε βήμα αναζήτησης το σύστημα δίνει στον χρήστη τις πιθανές επιλογές του με τη μορφή ενεργών facets. Ενεργεί σαν ένα σύστημα επαναδιατύπωσης των ερωτημάτων μετά από κάθε ενέργεια μηδενίζοντας όμως τα κενά αποτελέσματα γιατί έχει απενεργοποιήσει τις επιλογές που δεν είναι συμβατές μεταξύ τους. Γενικά, το faceted navigation έχει αξία όταν η συνεχείς επιλογές από facets έχουν μία λογική σειρά. Αν και σύγχρονες τεχνολογίες ανάπτυξης συστημάτων (κυρίως web) επιτρέπουν την σωστή επιλογή ακόμα και συνεχών τιμών, το πρόβλημα που παραμένει έχει να κάνει με την σωστή διαχείριση ερωτημάτων κειμένου. Την απάντηση σε αυτό δίνει η τεχνική του faceted search όπως την εξετάζουμε παρακάτω.

3.4 Faceted search

Η αναζήτηση πληροφοριών είναι ένας διάλογος μεταξύ του χρήστη και του συστήματος προκειμένου ο πρώτος να καλύψει την ανάγκη για κάποιου είδους πληροφόρηση. Ο πυρήνας ενός τέτοιου συστήματος αναζήτησης η σωστή σχεδίαση των ερωτημάτων και η σωστή διαχείριση των αποτελεσμάτων. Προχωρώντας ένα βήμα παραπάνω, συναντάμε το faceted search έναν πιο δυναμικό τρόπο να διαχειριστούμε τον διάλογο χρήστη - συστήματος. Οι Hearst et al (Hearst et al, 2002) προτείνουν ότι τα δεδομένα χαρακτηρίζονται από μεταδεδομένα και με αυτά μπορούμε να τα ταξινομήσουμε δηλαδή να τα οργανώσουμε σε πολυδιάστατες ταξινομήσεις. Τα μεταδεδομένα μπορεί να είναι faceted δηλαδή ένα σετ από κατηγορίες και επίσης ιεραρχημένα. Μπορεί να είναι single valued ή multi valued δηλαδή να έχουν μια μόνο τιμή κάθε φορά ή πολλές διαφορετικές τιμές ταυτόχρονα. Το άμεσο πρόβλημα είναι το κατά πόσο είναι εύκολο να αποφασίσουμε ποια είναι τα χαρακτηριστικά που θα χτίσουν αυτά τα μεταδεδομένα.

Στην πράξη, οι περισσότερες συλλογές εγγράφων που μας ενδιαφέρουν βρίσκονται σε ημιδομημένη μορφή. Για παράδειγμα ένα έγγραφο κειμένου, μπορεί να έχει κάποιο βαθμό δόμησης όπως τίτλο, εισαγωγή ή επίλογο αλλά δεν ακολουθεί κάποια αυστηρή δομή όπως τα δεδομένα στις βάσεις δεδομένων. Συνήθως εξάγονται μεταδεδομένα για τη δομή του εγγράφου δηλαδή χαρακτηριστικά με κάποια λογική. Σε αυτά τα ημι-δομημένα έγγραφα μπορούμε να εφαρμόσουμε text search και σε συνδυασμό με το faceted navigation έχουμε τον πυρήνα του faceted search. Ο χρήστης ξεκινάει με μια αναζήτηση free text και εφαρμόζει facets ώστε να περιορίσει τα αποτελέσματα ενώ τα μηδενικά αποτελέσματα (dead ends) είναι πρακτικά ανύπαρκτα.

3.4.1 Τα κύρια χαρακτηριστικά του faceted search

Το faceted search έχει τη δυνατότητα να μεταβάλει με έναν μοναδικό τρόπο την εμπειρία αλληλεπίδρασης του χρήστη με το σύστημα αναζήτησης. Παρέχει έναν ευέλικτο πλαίσιο επιτρέποντας στους χρήστες να ικανοποιούν ένα εύρος αναγκών πληροφόρησης ξεκινώντας από απλές περιπτώσεις ως πολύ σύνθετες. Σε συνδυασμό με το free text search και τις κατάλληλες λέξεις κλειδιά η τεχνική γίνεται τόσο δυναμική και αποδοτική που σήμερα είναι η de facto λύση σε πολλά συστήματα όπως τα e-commerce (Tunkelang, 2009).

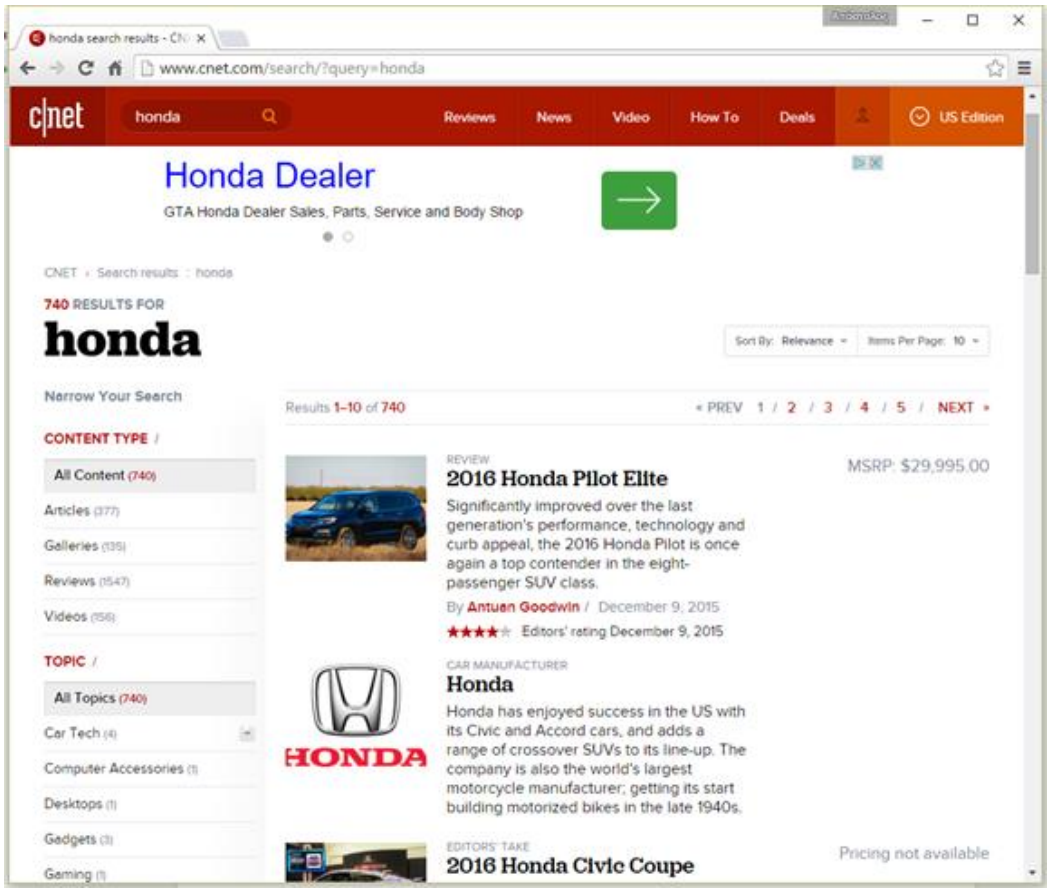
Συγκεντρωτικά τα κύρια χαρακτηριστικά ενός τέτοιου συστήματος παρουσιάστηκαν από τους Russel και Tyler (Russel and Tyler, 2012) ως εξής:

- Τα facets είναι ανεξάρτητα χαρακτηριστικά ή διαστάσεις (dimensions) με τα οποία μπορούμε να κατατάξουμε κάποιο αντικείμενο, δηλαδή ένα έγγραφο της συλλογής

μας σε κάποια ταξινομία.

- Επιλέγουμε facets (selections) αλλάζοντας τις τιμές τους και διαπιστώνοντας τα αποτελέσματα που σχετίζονται με αυτά. Το σετ των δυνατών επιλογών στα facets είναι το navigational context.
- Το faceted search σχεδόν μηδενίζει τη περίπτωση μηδενικών αποτελεσμάτων διότι οι επιτρεπόμενες επιλογές ορίζονται από τα πραγματικά δεδομένα.
- Τα facets μπορούν να είναι single-selected ή multi-selected. Τα πρώτα είναι αμοιβαία αποκλειόμενα.
- Τα multi-select facets επιτρέπουν λογικές ενέργειες OR ή AND.
- Συνήθως οι τιμές που επιλέγονται σε διαφορετικά facets εφαρμόζονται προσθετικά (AND) ενώ αυτές που επιλέγονται στο ίδιο facets εφαρμόζονται διαζευκτικά (OR).

Μερικά παραδείγματα dimensions για Facets είναι το είδος ενός βιβλίου, η ημερομηνία έκδοσης και ο συγγραφέας. Το faceted search επιτρέπει στους χρήστες να επαναδιατυπώνουν το ερώτημά τους επιλέγοντας τις τιμές των facets και αλλάζοντας την προτίμησή τους σε κάποια ή κάποιες από τις διαστάσεις.



Εικόνα 3: Faceted Search στο CNET

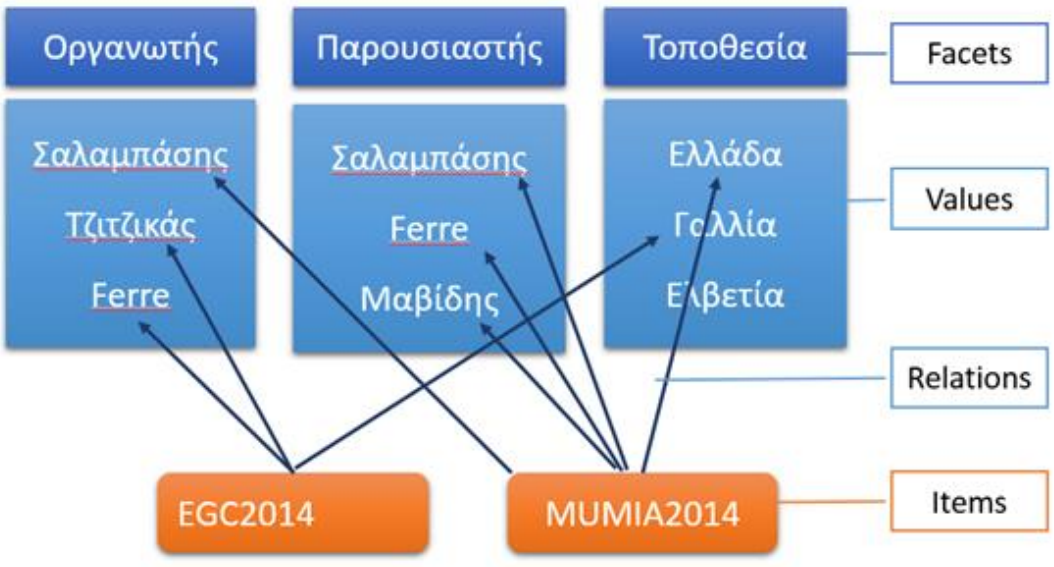
3.4.2 Το μοντέλο δεδομένων

Το μοντέλο δεδομένων στο οποίο στηρίζεται το faceted search είναι σχετικά απλό (Ferre S., Hermann A., 2012). Κάθε στοιχείο (item) περιγράφεται ταυτόχρονα από ένα πλήθος facets ή dimensions ενώ κάθε facet έχει ένα εύρος τιμών. Έτσι κάθε στοιχείο περιγράφεται από ένα πλήθος ζευγαριών facets-values τα οποία καλούμε χαρακτηριστικά. Κάθε χαρακτηριστικό έχει ένα σετ από στοιχεία-κόμβους (nodes). Ένα facet δεν περιγράφεται υποχρεωτικά με όλα τα στοιχεία.

Σε κάθε βήμα πλοήγησης η τρέχουσα επιλογή ορίζεται σαν ένα σετ από στοιχεία ενώ η αρχική επιλογή είναι όλα τα στοιχεία. Στην τρέχουσα επιλογή ένα σετ από περιορισμοί είναι διαθέσιμοι στον χρήστη. Ο κάθε περιορισμός είναι ένα χαρακτηριστικό που καταλήγει σε τουλάχιστον ένα στοιχείο στην τρέχουσα επιλογή και συνήθως συνοδεύεται από το πλήθος των στοιχείων που συμφωνούν με αυτόν. Κάθε περιορισμός παρουσιάζεται σαν ένα facet και είναι ταυτόχρονα σύνοψη για την τρέχουσα συλλογή αλλά και ο δρόμος για υποσύνολα στοιχείων.

Η αναζήτηση στο faceted search βασίζεται σε τρέχουσες επιλογές και περιορισμούς και μετά από κάθε βήμα πλοήγησης πρέπει να επαναϋπολογισθούν και να παρουσιαστούν κατάλληλα στον χρήστη. Γενικά οι περιορισμοί θα πρέπει να είναι διαθέσιμοι στον χρήστη και η από-επιλογή κάποιου να οδηγεί σε περισσότερα αποτελέσματα (zoom out).

Το παρακάτω σχήμα απεικονίζει το μοντέλο δεδομένων στο faceted search. Σε αυτό περιλαμβάνονται facets (π.χ. χρώμα προϊόντος), τιμές (π.χ. μπλε και κόκκινο), items δηλαδή τα έγγραφα της συλλογής μας που έχουν ταξινομηθεί (π.χ. προϊόν) και σχέσεις που προκύπτουν (π.χ. το προϊόν είναι κόκκινο).



Εικόνα 4: Μοντέλο δεδομένων στο faceted search

Στο μοντέλο δεδομένων του Faceted search ισχύουν τα εξής:

- Πολλά facets Μπορεί να έχουν τις ίδιες τιμές
- Ένα στοιχείο δηλαδή ένα έγγραφο μπορεί να σχετίζεται με πολλές τιμές του ίδιου facets
- Οι τιμές των facets μπορεί να περιλαμβάνουν κάποια ιεραρχία

Οι παρακάτω τυπικοί συμβολισμοί περιγράφουν το μοντέλο:

- I: Στοιχεία
- F: Facets
- $\{V_f\}_{F \in V}$: Πεδίο τιμών για κάθε facets
- $R \subseteq I * F * V$: Συσχέτιση μεταξύ των στοιχείων, facets και τιμών, όπου το R είναι μια τριπλέτα (item, facet, value)

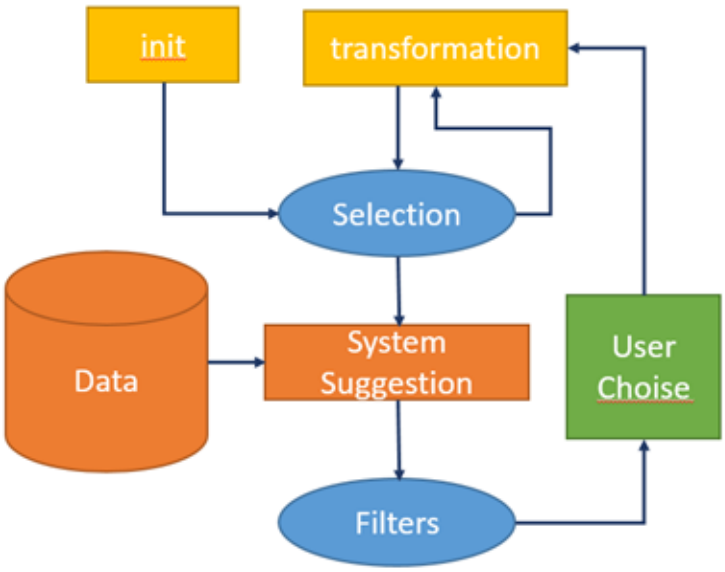
Περιγραφή των συσχετίσεων:

- $R \in 2^{I \times F \times V}$: Το R είναι ένα σετ από τριπλέτες που ανήκει σε μία από τις $2^{I \times F \times V}$
- $R \in (I \rightarrow 2^{F \times V})$: Τα items περιγράφονται από ζευγάρια τιμών (facet, value)
- $R \in (I \times F \rightarrow 2^{V})$: Σε κάθε ζευγάρι από items και facets αντιστοιχεί ένας πίνακας τιμών για τα οποία χτίζεται Index. Τα ενεργά facets είναι αυτά στα οποία ο πίνακας έχει τουλάχιστον ένα μέλος
- $R \in (F \times V \rightarrow 2^{I})$: Ένας index στα items που έχουν τα ζευγάρια (facets, value)

3.4.3 Το μοντέλο αλληλεπίδρασης με τον χρήστη

Το μοντέλο αλληλεπίδρασης με τον χρήστη (user interaction model) αναπαριστά τη ροή επιλογών μεταξύ χρήστη και συστήματος. Σε αυτό το μοντέλο, ο χρήστης εκτελεί κάποια αρχική διαδικασία αναζήτησης, συνήθως με λέξεις κλειδιά, και παίρνει τα αρχικά αποτελέσματα από τα οποία το σύστημα χτίζει τα facets, υπολογίζει τις τιμές τους και τα εμφανίζει κατάλληλα μέσω του interface. Ο χρήστης προκειμένου να περιορίσει τα αποτελέσματα, επιλέγει κάποιο ή κάποια από τα facets αυτά με αποτέλεσμα να περιορισθούν τα αποτελέσματα, και να αλλάξουν οι διαθέσιμες επιλογές του, δηλαδή τα facets να επαναϋπολογισθούν.

Οι επιλογή των facets σε κάθε στάδιο μπορεί να γίνεται αυτοματοποιημένα από το σύστημα το οποίο τα επιλέγει και υπολογίζει τις τιμές βάσει των δεδομένων της συλλογής εγγράφων προς αναζήτηση. Ο χρήστης παίρνει αποτελέσματα τα οποία μπορεί να φιλτράρει ακόμα και στατικά και να διαλέξει από αυτά. Η κάθε επιλογή του χρήστη οδηγεί σε επαναϋπολογισμό των facets.



Εικόνα 5: Το μοντέλο αλληλεπίδρασης με τον χρήστη (Ferre S., 2014)

3.4.4 Navigation modes στο faceted search

Οι επιλογές πλοήγησης του χρήστη (navigation modes) είναι οι ενέργειες που είναι διαθέσιμες από το σύστημα ανά πάσα στιγμή και καθορίζουν τον διαθέσιμο χώρο πληροφόρησης του χρήστη. Υπό κάποια έννοια η πλοήγηση του χρήστη είναι μια διαδικασία συνεχούς υποβολής νέων ερωτημάτων και η μετάβαση στα αποτελέσματά τους. Τα ερωτήματα αυτά διαμορφώνονται από το interface και παρουσιάζονται ως σύνδεσμοι, controls (checkboxes, range controls, radio buttons κτλ.) και πλαίσια αναζήτησης.

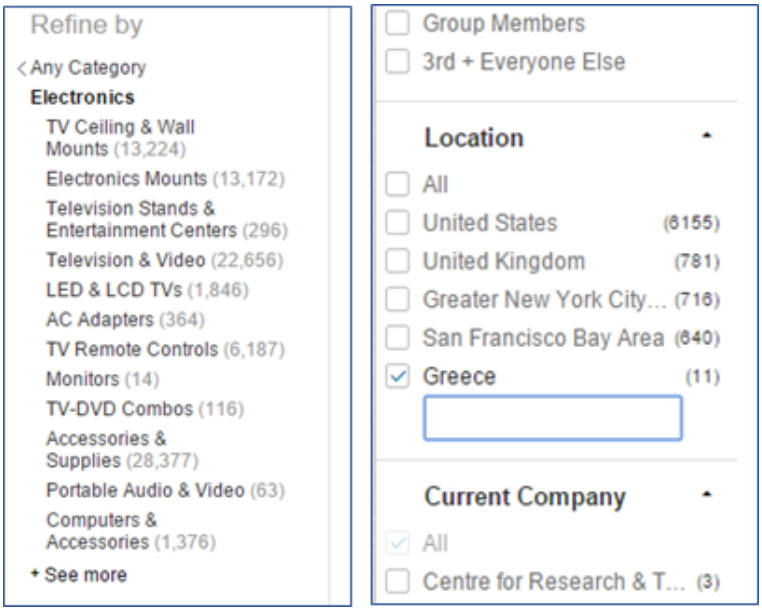
Οι διαθέσιμες επιλογές πλοήγησης συνοψίζονται στις εξής (Sacco G., Tzitzikas Y., 2009):

1. zoom-in: ο χρήστης επαναδιατυπώνει το ερώτημα με μεγαλύτερη ακρίβεια αναζητώντας ένα πιο στενό εύρος αποτελεσμάτων μέσα στα ήδη υπάρχοντα. Στο faceted search αυτό σημαίνει ότι έχει επιλέξει ένα facet και οδηγείται σε ένα πιο στενό πλαίσιο αποτελεσμάτων.
2. zoom-out: ο χρήστης κάνει το ερώτημα πιο γενικό συνήθως καταργώντας κάποιον περιορισμό. Τα αναμενόμενα αποτελέσματα είναι πιο πολλά από αυτά που έχει ήδη. Όπως και πριν, στο faceted search αυτό σημαίνει ότι έχει από-επιλέξει ένα facet και οδηγείται σε ένα ευρύτερο πλαίσιο αποτελεσμάτων.
3. shift: Αντικαθίσταται ένα μέρος του ερωτήματος με κάτι σχετικό. Αυτό μπορεί να είναι το αποτέλεσμα της αντικατάστασης της επιλογής ενός facet με κάποιο άλλο γιατί ο χρήστης κρίνει ορθότερη την επιλογή του δεύτερου.
4. pivot: Αντικαθίσταται όλο το ερώτημα με κάποιο άλλο σχετικό ερώτημα. Είναι ο τρόπος με τον οποίο επαναδιατυπώνεται όλο το ερώτημα.
5. slice and dice: επιτρέπει την διαζευκτική επιλογή πολλών τιμών για ένα facet.

6. range selection: επιτρέπει στον χρήστη να δώσει όρια στο ερώτημα για facets με συνεχείς ή πραγματικές τιμές.
7. Query by examples: Μια ειδική περίπτωση πλοήγησης είναι όταν ο χρήστης μέσα στα υπάρχοντα αποτελέσματα διακρίνει ενδιαφέροντα χαρακτηριστικά και επαναδιατυπώνει το ερώτημά του σε σχέση με αυτά τα χαρακτηριστικά

Στα υπάρχοντα συστήματα faceted search υλοποιούνται σίγουρα τα zoom-in και zoom-out και οι ενέργειες συχνά ονομάζονται **drill-down, drill-up**. Αμφίβολο είναι εάν ένα σύστημα υλοποιεί το shift που συχνά ονομάζεται **drill-sideways**. Αν πρέπει να αναλογιστούμε τι κάνει το drill-sideways, ας πάρουμε το παράδειγμα όπου ένας χρήστης επιλέγει ένα facet όπως τη μάρκα μίας τηλεόρασης. Αυτόματα τα αποτελέσματα από τις υπόλοιπες μάρκες εξαφανίζονται και τα άλλα facets επαναϋπολογίζονται για την συγκεκριμένη μάρκα. Ωστόσο, ο χρήστης ίσως θέλει να μπορεί να επιλέξει εναλλακτικές για τη μάρκα, αλλά η επιλογή αυτή δεν είναι πια διαθέσιμη. Αυτό οδηγεί σε φτωχό interface και συνήθως αντιμετωπίζεται με χρήση της ενέργειας Back του Browser για να γίνει μετά νέα επιλογή.

Μερικά συστήματα υλοποιούν το drill-sideways ως εξής: η επιλογή ενός facet δεν αποκρύπτει τις άλλες τιμές του και ο χρήστης έχει τη δυνατότητα να επιλέξει σε αυτές κάτι άλλο. Αυτό κάνει για παράδειγμα το Amazon. Το LinkedIn προχωράει ένα ακόμα επίπεδο, προσφέροντας στον χρήστη την επιλογή να προσθέσει δυναμικά την τιμή που επιθυμεί για το facet και να πάρει τα ανάλογα αποτελέσματα.



Εικόνα 6: Διαδικασία drill-sideways από το amazon (αριστερά) και το linkedin (δεξιά)

Το προβληματικό σημείο με το drill-sideways έχει να κάνει με τον επιπλέον φόρτο που πρέπει να επωμισθεί το σύστημα ώστε να υπολογίσει τις επιπλέον τιμές. Ας πάρουμε το παράδειγμα ενός συστήματος με ένα Facet F και τιμές τα Α, Β, Γ με count 10,20,30

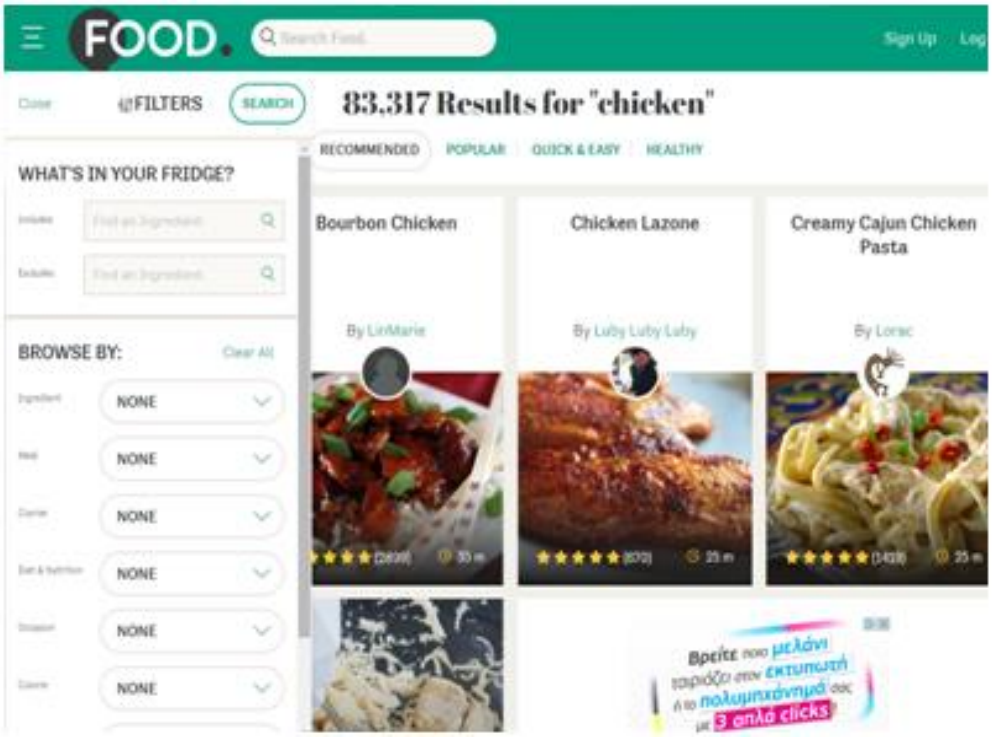
αντίστοιχα. αν ο χρήστης επιλέξει το Α θέλουμε όλα τα υπόλοιπα facets να υπολογίσουν τις τιμές τους σύμφωνα με τη νέα επιλογή, αλλά το F να κρατήσει τις τιμές του για τα Β και Γ. Αυτό μπορεί να γίνει με διάφορους τρόπους αλλά όλοι απαιτούν επιπλέον υπολογιστικό κόστος γιατί απαιτούν την εκτέλεση πολλών queries αντί ενός. Μια εύκολη λύση κατάλληλη για μικρά συστήματα θα ήταν το σύστημα σε κάθε βήμα να κρατάει στην cache του τις προηγούμενες τιμές και να τις παρουσιάζει στον χρήστη. Σε κάθε περίπτωση το drill-sideways είναι μία σχετικά πολύπλοκη διαδικασία που επιβαρύνει το σύστημα αλλά ευνοεί την αλληλεπίδραση με τον χρήστη.

3.5 Faceted Search ή Filtered Search;

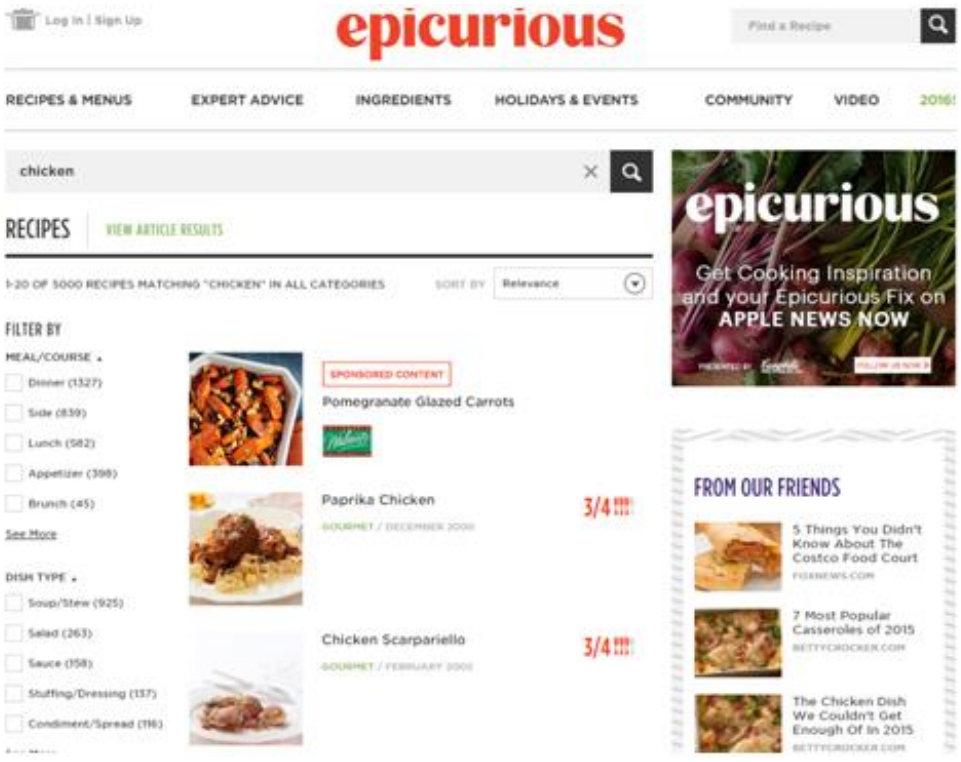
Οι όροι filtered navigation και faceted navigation εναλλάσσονται συχνά και πολλές φορές αποδίδονται λανθασμένα σε περιγραφές συστημάτων αναζήτησης δεδομένων. Η βασική ιδέα παραμένει η ίδια, δηλαδή αφού κάνουμε μια αρχική free text αναζήτηση, θέλουμε να περιορίσουμε τα αποτελέσματα σε αυτά που ικανοποιούν κάποια κριτήρια. Όπως είναι αναμενόμενο υπάρχει μεγάλη αλληλοεπικάλυψη μεταξύ των δύο συστημάτων αλλά δεν είναι το ίδιο. Τα facets περιγράφουν τα δεδομένα από πολλές διαφορετικές όψεις δίνοντας έτσι μεγαλύτερη ευελιξία στον χρήστη.

Τα φίλτρα είναι συνήθως στατικά και η εφαρμογή τους μπορεί να μας οδηγήσει σε κενά αποτελέσματα (dead ends). Για παράδειγμα στο δικτυακό τόπο Food.com η αναζήτηση για τον όρο chicken οδήγησε σε ένα πλήθος αποτελεσμάτων στα οποία μπορούμε να εφαρμόσουμε στατικά φίλτρα όπως φαίνεται στην εικόνα 7. Ο χρήστης δεν είναι σε θέση αν γνωρίζει από πριν αν η εφαρμογή των φίλτρων θα οδηγήσει σε αποτελέσματα.

Από την άλλη τα Facets είναι δυναμικές όψεις που δημιουργούνται αυτόματα ή ημιαυτόματα από το σύστημα. Σε κάθε βήμα της αναζήτησης επαναϋπολογίζονται και παρουσιάζονται στον χρήστη μόνο αυτά που δίνουν αποτελέσματα μηδενίζοντας πρακτικά τα κενά αποτελέσματα παρέχοντας την δυνατότητα στον χρήστη να γνωρίζει πριν την επιλογή ενός facet πόσα θα είναι τα αποτελέσματα που περιμένει. Μετά την εφαρμογή του facet θα έχει στη διάθεσή του άλλο σετ επιλογών. Η δυναμική αυτή διαδικασία βοηθάει τους χρήστες στο να εστιάσουν στα αποτελέσματα ιδιαίτερα αυτούς που δεν γνωρίζουν καλά το πεδία αναζήτησης. Για παράδειγμα, στο δικτυακό τόπο Epicurious.com η αναζήτηση για τον όρο chicken οδήγησε πάλι σε κάποια αποτελέσματα (Εικόνα 8). Σε αυτή την περίπτωση ο χρήστης έχει στη διάθεσή του ένα σύνολο από facets που υπολογίζονται δυναμικά για το συγκεκριμένο ερώτημα. Για κάθε facet αναγράφεται το σύνολο των αποτελεσμάτων που είναι σύμφωνα με αυτό.



Εικόνα 7: Εφαρμογή στατικών φίλτρων στα αποτελέσματα της αναζήτησης



Εικόνα 8: Επιλογές από facets στα αποτελέσματα της αναζήτησης

Υπάρχει μία ακόμα πολύ σημαντική διαφορά μεταξύ των δύο τεχνικών. Στη διαδικασία αναζήτησης με facets μετά από κάθε βήμα αποδίδεται νέα βαρύτητα στα αποτελέσματα και μάλλον τα πιο σημαντικά από αυτά βρίσκονται πάλι στην κορυφή της λίστας αποτελεσμάτων

βοηθώντας τον χρήστη τα προσπελάσει όπως επιθυμεί. Σε αντίθεση με την εφαρμογή των φίλτρων όπου η βαρύτητα αποδίδεται μόνο στην αρχική αναζήτηση και παραμένει η ίδια σε κάθε επόμενο στάδιο.

3.6 Προβλήματα στην εφαρμογή του faceted search

Από την περιγραφή ενός συστήματος μέχρι την επιτυχημένη εφαρμογή του μπορεί να προκύψουν διάφορα προβλήματα τα οποία πρέπει να αντιμετωπίσουμε. Ο Tunkelang (Tunkelang D, 2009) κατέγραψε ως προβλήματα υλοποίησης ενός συστήματος faceted search τα εξής:

- Scale
- Efficiency
- Information overload
- Availability of metadata
- The Vocabulary problem
- Multiple Entity Types

3.6.1 Scale

Το μέγεθος της συλλογής εγγράφων είναι μια ένδειξη για την ποσότητα της πληροφορίας και άρα την αξία της. Το faceted search κάνει πιο εύχρηστη την εξερεύνησης και την πρόσβαση σε μεγαλύτερο μέρος της ενώ το μέγεθος αναφέρεται στο πλήθος των εγγράφων της συλλογής, στον αριθμό των τιμών των facet και στο μέγεθος του κειμένου ανά έγγραφο που είναι διαθέσιμο για αναζήτηση. Το πρόβλημα αφορά στις απαιτήσεις του hardware και την πολυπλοκότητα των ενεργειών. Για όλα τα έγγραφα απαιτείται έναν Inverted index και ένας Index με την αντιστοιχία facets ανά document. Σε μεγάλες συλλογές η αποθήκευση των δομών στην κύρια μνήμη είναι δαπανηρή. Μια λύση θα ήταν η χρήση της δευτερεύουσας μνήμης ή η χρήση πολλών server ταυτόχρονα που διαμοιράζονται την εφαρμογή.

3.6.2 Efficiency

Οι ενέργειες που απαιτούνται σε ένα faceted search σύστημα είναι πολύπλοκες. Το σύστημα επεξεργάζεται το ερώτημα σαν οποιοδήποτε άλλο σύστημα αναζήτησης αλλά σε δεύτερο χρόνο υπολογίζει τις διαθέσιμες τιμές των facets και τις εμφανίζει στον χρήστη. Συχνά το πλήθος των αποτελεσμάτων για κάθε τιμή facet πρέπει να εμφανίζεται δίπλα στις τιμές. Οι δύο τεχνικές που θα μπορούσαμε να χρησιμοποιήσουμε είναι εξίσου υπολογιστικά δαπανηρές. Η μία top-down τακτική διατρέχει τον Index και υπολογίζει εξαρχής τις τιμές των

facets για όλα τα έγγραφα. Η άλλη τακτική bottom-up ελέγχει όλα τα έγγραφα στα αποτελέσματα και αποδίδει τιμές στα facets για αυτά. Το κόστος υλοποίησης σχετίζεται με τις δομές δεδομένων που επιλέγονται. Για παράδειγμα το Solr συνδυάζει και τις δύο διαφορετικές τακτικές. Σε κάθε περίπτωση η απόδοση του συστήματος μπορεί να βελτιωθεί με καλύτερο υλικό ή με κατανομή της επεξεργασίας σε πολλά συστήματα

3.6.3 Information overload

Με το πλήθος των διαθέσιμων επιλογών, ερωτημάτων και τιμών από facets να αυξάνουν την πολυπλοκότητα του Interface προκύπτει το πρόβλημα της προσοχής του χρήστη. Σε κάποιο βαθμό η υπερβολική αύξηση της πληροφορίας είναι εις βάρος της διεπαφής. Δύο είναι οι παράγοντες που υπερφορτώνουν τις επιλογές μας. Το πλήθος των facets και το πλήθος τιμών ανά facet. Γενικά, όταν τα facets είναι πολλά δεν πρέπει να έχουμε όλες τις επιλογές άμεσα διαθέσιμες αλλά τα ποιο σημαντικά από αυτά. Υπάρχουν όμως και άλλοι παράγοντες όπως η συσχέτιση των facets μεταξύ τους. Αντί της υπερ-πληροφόρησης του χρήστη επιλέγουμε να εμφανίσουμε τις πιο σημαντικές τιμές των facets ως εξής:

- Επιλέγουμε αυτά με τη μεγαλύτερη κάλυψη στη συλλογή.
- Ενισχύουμε την υψηλής εντροπίας κατανομή τιμών ώστε να καλύψουν καλύτερα τα έγγραφά μας.
- Συγχωνεύουμε facets τα οποία περιέχουν αλληλεπικαλυπτόμενες τιμές.

Το μεγάλο πλήθος τιμών μπορεί να παρουσιάζεται προοδευτικά μέσω του "δέντρου" ιεραρχίας των facets, ωστόσο η ιεράρχηση αυτή δεν είναι πάντα εφικτή. Επίσης σε κάποια πεδία όπως τα ονόματα τα facets θα παίρνουν υποχρεωτικά πολλές τιμές και η ιεραρχία δεν έχει νόημα.

3.6.4 Availability of metadata

Το faceted search προϋποθέτει την ύπαρξη μεταδεδομένων για αξιοποίηση από το σύστημα κατάταξης. Διαφορετικά η αναζήτηση καταλήγει να είναι η κλασική text search. Η διαθεσιμότητα των μεταδεδομένων είναι λοιπόν κρίσιμο στοιχείο της διαδικασίας. Η διαδικασία είναι πιο απλή για δεδομένα που βρίσκονται σε βάσεις δεδομένων λόγω της αυστηρής δομής που επιτρέπει τον ορισμό των facets. Σε άλλες περιπτώσεις αξιοποιούνται μεταδεδομένα όπως ημερομηνίες, εκδότης ενός άρθρου, κατηγοριοποίηση κειμένου όπως νέα, δραστηριότητες κτλ. Μία πρόσφατη τεχνική, η καταγραφή των hashtags σε άρθρα, εικόνες, tweets, ειδήσεις δίνει τα απαραίτητα μεταδεδομένα για την ταξινόμησή τους.

Σε κάθε περίπτωση, η μορφή των εγγράφων σπάνια είναι εντελώς αδόμητη. Η αυτοματοποιημένη ανάκτηση τιμών από μη ημιδομημενα έγγραφα είναι επιθυμητή και

συνήθως εφικτή αλλά σε κάποιο σημείο μάλλον θα χρειαστεί η επέμβαση του χρήστη.

Στην πράξη υπάρχουν διάφορες τεχνικές, για να εμπλουτίσουμε το μη δομημένο κείμενο έτσι ώστε να περιλάβει μεταδεδομένα. Μερικές είναι οι:

- Εντοπίζουμε και ορίζουμε μεταδεδομένα όπως πηγή, τόπος και μήκος εγγράφου. Αυτά συνήθως είναι άμεσα διαθέσιμα.
- Ορίζουμε κανόνες, βάση των οποίων και κάνουμε μια πρώτη ταξινόμηση των εγγράφων.
- Με τη χρήση των τεχνικών εξαγωγής όρων (term extraction) ανακτούμε τους όρους και δημιουργούμε ένα "λεξικό" για να χρησιμοποιηθεί σαν τιμές στα Facets.

3.6.5 The Vocabulary problem

Το πρόβλημα του λεξικού είναι ταυτόχρονα πρόβλημα και κίνητρο για το faceted search. οι επιλογές των τιμών για τα facets μπορεί να προέρχονται από την επέμβαση του χρήστη, ωστόσο θα θέλαμε οι τιμές αυτές να εμπλουτίζονται με τις προτιμήσεις των χρηστών ώστε να είναι οι πιο κατανοητές σε αυτούς. Το πρόβλημα εντοπίσθηκε από το 1987 στην δημοσίευση "The Vocabulary Problem in Human-System Communication". Σε αυτή τονίστηκε ότι οι λέξεις κλειδιά που χρησιμοποιούν οι χρήστες δεν είναι οι ίδιες με αυτές που περιλαμβάνονται στις βάσεις δεδομένων και προτάθηκε μία προσαρμοστική λύση που στηρίζεται σε λεξικά για να εντοπίζουν τη συνάφεια λέξεων και όρων.

3.6.6 Multiple entity types

Ένα τελευταίο πρόβλημα αφορά στην παρουσία υπερβολικά πολλών τιμών για ένα facet. Για παράδειγμα σε μία βιβλιοθήκη ο συγγραφέας είναι μέρος του εγγράφου αλλά θα μπορούσε να είναι και τιμή για κάποιο facet. Οι τιμές όμως μπορεί να είναι υπερβολικά πολλές και επίσης κάποιο document μπορεί να έχει περισσότερους από έναν συγγραφείς. Μια προσέγγιση είναι ένα μερικώς αποκανονικοποιημένο μοντέλο, όπου όλα είναι documents ή facets αλλά όχι και τα δύο. Αν κάτι είναι στο έγγραφο δεν θα είναι στα facets. Όμως με αυτό το μοντέλο δεν μπορούμε να απαντήσουμε σε ερωτήματα όπως αν έχουμε συγγραφείς από μια συγκεκριμένη χώρα γιατί δεν υπάρχει η σύνδεση των facets της χώρας με τους συγγραφείς. Η λύση είναι σύνθετη και πολύπλοκη στην υλοποίηση. Τόσο οι συγγραφείς όσο και τα έγγραφα έχουν τα δικά τους ανεξάρτητα facets και αυτά σχετίζονται μεταξύ τους.

3.7 Προκλήσεις κατά την επιλογή και σχεδίαση των facets

Μία απλή εφαρμογή με facet search έχει σίγουρα δύο περιοχές αλληλεπίδρασης με τον

χρήστη. Αυτή που παρουσιάζονται τα facets με τις τιμές τους και αυτή που παρουσιάζονται τα αποτελέσματα. Επιπλέον στοιχεία μπορεί να περιλαμβάνουν αναλυτική ή συγκεντρωτική όψη για επιλεγμένους πόρους, λωρίδα breadcrumb για το ιστορικό πλοήγησης του χρήστη. Στο στάδιο της σχεδίασης αρκετές αποφάσεις πρέπει να ληφθούν υπόψη όπως οι εξής (Sacco G., Tzitzikas Y., 2009):

- Ποιος είναι ο κατάλληλος τύπος δεδομένων για τα διάφορα facets ανάλογα με το αν περιλαμβάνουν διακριτές τιμές, ιεραρχία, συνεχείς τιμές κτλ.
- Ποιες τιμές facets παρουσιάζονται στον χρήστη; Θα πρέπει να είναι όλες οι τιμές ορατές;
- Μπορεί ο χρήστης να επιλέξει πολλαπλές τιμές για ένα facet ή το facet πρέπει να υποστηρίζει μοναδική επιλογή; Αν μπορεί να επιλέξει πολλαπλές τιμές, προκαλεί σύζευξη ή διάζευξη στα αποτελέσματα;

Με βάση αυτές τις προκλήσεις κατά το στήσιμο ενός faceted navigation σχήματος και κατά τη σχεδίαση του συστήματος η προσοχή μας στρέφεται στα εξής:

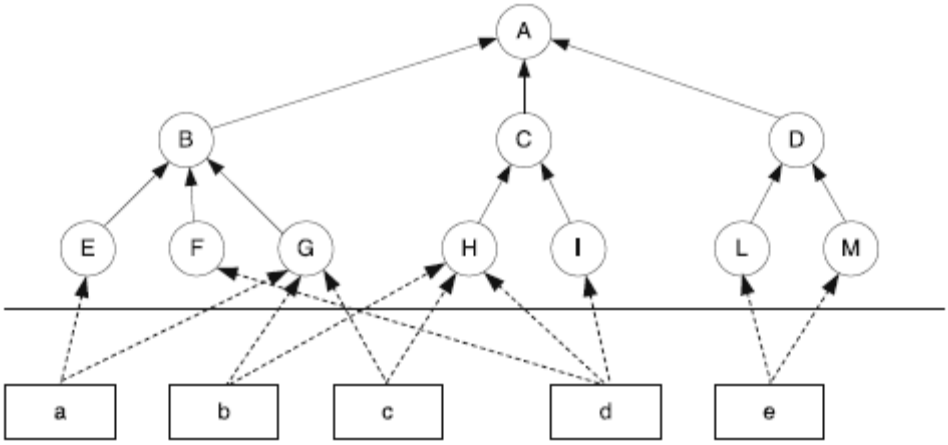
- Boolean query logic: Η επιλογή πολλών διαφορετικών facets σημαίνει σύζευξη (AND) και λειτουργεί σαν προσθετικά φίλτρα. Η επιλογή όμως πολλών τιμών από το ίδιο facet συνήθως σημαίνει διάζευξη (OR) και επιτρέπει αποτελέσματα από όλες τις τιμές.
- Ακατάστατο interface: Η επιλογή του να έχουμε όλα τα controls διαθέσιμα και ορατά στον χρήστη δεν είναι καλή ειδικά όταν οι επιλογές είναι πολλές ή έχουμε περιορισμό στον χώρο της οθόνης. Αντίθετα είναι απαραίτητη η σωστή τοποθέτηση, ιεραρχικά και δομημένα ώστε ο χρήστης να επεκτείνει μόνο τα controls που επιθυμεί.
- Ενσωμάτωση αναζήτησης με λέξεις κλειδιά: Επιπρόσθετα με τις διαθέσιμες επιλογές του χρήστη είναι απαραίτητη η δυνατότητα αναζήτησης με λέξεις κλειδιά σε κάθε στάδιο της εξερεύνησης αποτελεσμάτων. Η δυνατότητα αυτή μπορεί να λειτουργεί προσθετικά στις ήδη επιλεγμένες τιμές των facets η και να οδηγεί στην αρχική επαναδιατύπωση του ερωτήματος. Ο διαχωρισμός μεταξύ των δύο λειτουργιών πρέπει να γίνεται ξεκάθαρα.

3.8 Dynamic taxonomies και Faceted Search

Τα facets βασίζονται στην ταξινομία (taxonomy) δηλαδή στην πρακτική και στην επιστήμη της κατάταξης. Αρχικά, μια taxonomy αναφέρονταν στην κατάταξη οργανισμών σε είδη ενώ στην ευρύτερη έννοιά της αναφέρεται στην κατάταξη αντικειμένων ή εννοιών σε κλάσεις. Τα αντικείμενα και οι κλάσεις μπορεί να σχετίζονται μεταξύ τους με τη μορφή γονέας-παιδί

σχηματίζοντας έτσι ιεραρχικές μορφές δικτύων κατάταξης. Στην σύγχρονη θεωρία γνώσης η ταξινομία θεωρείται στενότερη από τις οντολογίες οι οποίες καλύπτουν περισσότερες μορφές σχέσεων μεταξύ των αντικειμένων. Στα μαθηματικά, η ιεραρχική ταξινόμηση είναι μια δομή τύπου δέντρου και οι κλάσεις είναι οι κόμβοι στο δέντρο. Στην κορυφή υπάρχει ένας κόμβος ρίζα και κάθε ένας στο κατώτερο επίπεδο περιλαμβάνει ένα υποσύνολο από τα αντικείμενα με σχέσεις μεταξύ των κόμβων τις "IS-A" και "HAS-A". Το μοντέλο αυτό ενσωματώθηκε σε πολλές εφαρμογές όπως το ηλεκτρονικό εμπόριο στο οποίο είναι ζητούμενο η κατάταξη των προϊόντων σε κατηγορίες ώστε να είναι πρακτικά εφικτή η αναζήτησή τους από τον χρήστη.

Οι Dynamic Taxonomies η αλλιώς faceted search systems είναι γενικά ένα μοντέλο διαχείρισης της γνώσης βασιζόμενο σε μία πολυδιάστατη κατάταξη (**multidimensional classification**) ετερογενώς αντικειμένων δεδομένων (Sacco G, Tzitzikas Y., 2009). Σε αυτές τα αντικείμενα κατατάσσονται σε μία αλλά ίσως και σε περισσότερες τάξεις σχηματίζοντας την πολυδιάστατη κατάταξη που είναι πιο κοντά στις πραγματικές συνθήκες. Για παράδειγμα σε ένα e-commerce ένα προϊόν μπορεί να καταταχθεί σε πολλές κατηγορίες και να περιγραφεί από πολλά διαφορετικά χαρακτηριστικά (facets). Το exploratory search είναι μια συνεχής διαδικασία zoom in και zoom out στην πολυδιάστατη αυτή δομή. Στην εικόνα 9 εμφανίζεται μία δυναμική ταξινόμηση όπου οι γραμμές είναι υποκατηγορίες ενώ οι διακεκομμένες γραμμές κατατάξεις.

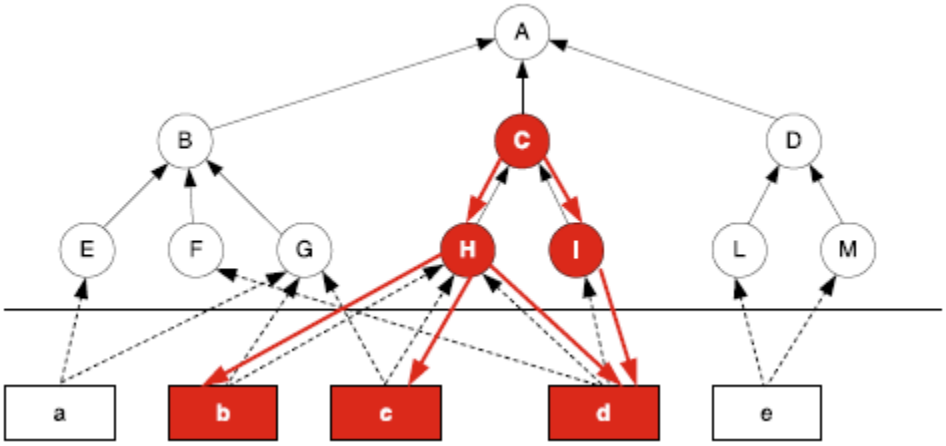


Εικόνα 9: Dynamic Taxonomy.

Στην επόμενη εικόνα 10, ο χρήστης έχει κάνει zoom-in στην υποκατηγορία C οπότε υπολογίζεται δυναμικά η κατάταξη μόνο για αυτή.

Τα πλεονεκτήματα των δυναμικών ταξινομήσεων σχετίζονται με την αλληλεπίδραση του χρήστη που είναι απλή και πιο φυσική, την διαδικασία εξερεύνησης που αναπαριστούν, την εύκολη σχεδίαση του σχήματος για την δομή του faceted search και την αποτελεσματικότητα της αναζήτησης διότι οδηγούν σε μικρά σετ αποτελεσμάτων που διαχειρίζονται εύκολα. Το

πεδίο εφαρμογής των δυναμικών ταξινομήσεων περιλαμβάνει φυσικά το e-commerce αλλά και τα e-auctions, e-catalogs, e-government, portals, multimedia κτλ.



Εικόνα 10: Dynamic Taxonomy, zoom-in C

Το σύστημα το οποίο υλοποιεί δυναμική ταξινόμηση και λειτουργεί κατατάσσοντας τα αντικείμενα με τη χρήση faceted classification είναι ένα faceted search system.

3.9 Καθορισμός των facet για τα συστήματα αναζήτησης

Φαίνεται λοιπόν το faceted search είναι μία τεχνολογία ώριμη και ευέλικτη, έτοιμη για ενσωμάτωση σε οποιοδήποτε σύστημα αναζήτησης εάν απαιτηθεί. Ωστόσο, ένα λεπτό σημείο που χρήζει περαιτέρω εξερεύνησης είναι ο ορισμός των facet και των ιεραρχικών σχέσεων μεταξύ τους που θα χρησιμοποιηθούν. πως δηλαδή θα επιλέξουμε ανάμεσα σε δεκάδες ίσως υποψήφια facets για το σύστημά μας. Το θέμα αυτό προσεγγίζει το πρόβλημα που τέθηκε από τον Tunkenlang (Tunkenlang, 2009) «Availability of metadata» όπως αναφέρθηκε νωρίτερα.

Διαπιστώνουμε ότι υπάρχουν τρεις προτάσεις, η μία είναι τα facets να ορίζονται χειρωνακτικά, η άλλη να υπολογίζονται από το περιεχόμενο των εγγράφων και η τρίτη να αξιοποιούνται τα Linked Open data.

3.9.1 Manual definitions

Η πρακτική των περισσότερων συστημάτων σήμερα είναι τα facets να καθορίζονται χειρωνακτικά. Για παράδειγμα στο Bobo Search (άρα στο Lucene) καθορίζουμε τα πεδία που θα χρησιμοποιηθούν σαν facets είτε μέσω το Bobo.Spring File, είτε προγραμματιστικά χρησιμοποιώντας τις κατάλληλες εντολές πριν την εκτέλεση του ερωτήματος ανάκτησης. Σε κάθε περίπτωση το σύστημα θα αναλάβει να επιστρέψει τις τιμές των facets που συνήθως είναι ένα περιορισμένο πλήθος με την υψηλότερη αξιολόγηση ή διασπορά. Η πρακτική αυτή

έχει βέβαια πολύ καλά αποτελέσματα αλλά σαν αντίβαρο έχει το χρόνο της προετοιμασίας που απαιτείται. Έχει και ένα ακόμα μειονέκτημα, αν απαιτείται το σύστημα να εναλλάσσει την συλλογή εγγράφων δυναμικά, δεν είναι σε θέση να επιλέξει από μόνο του τα facets.

3.9.2 Υπολογισμός των facets από αυτοματοποιημένη διαδικασία

Δεν είναι δύσκολο να φανταστούμε ένα σύστημα το οποίο αφού διαβάσει τα πεδία της βάσης δεδομένων να προτείνει στον χρήστη τα facets ή ακόμα να επιλέξει από μόνο του κάποια με τα ίδια κριτήρια που θα διαλέγαμε κι εμείς. Σε δεύτερο χρόνο θα χτίσει τις τιμές όπως και στα υπάρχοντα συστήματα. Το πρόβλημα που έχουμε είναι πως το σύστημα θα επιλέξει τιμές σε περιπτώσεις ημιδομημένων εγγράφων ή όταν έχουμε πεδία κειμένου. Για παράδειγμα, η ανάλυση ενός πεδίου κειμένου θα οδηγήσει σε δεκάδες όρους υποψήφιους για τιμές των facets συνήθως χωρίς πραγματική αξία.

Διάφορες προσεγγίσεις έχουν προταθεί και σε όλες υπάρχει ένας κοινός παράγοντας που είναι η ύπαρξη ενός εξωτερικού λεξικού για ανάλυση των υποψήφιων όρων και κατάταξή τους σε ιεραρχίες. Οι Dakka και Ipeirotis, (Dakka, W., & Ipeirotis, P. G, 2008) παρατήρησαν ότι η ορθή ονομασία του όρου σπάνια περιλαμβάνεται στο κείμενο του εγγράφου. Για παράδειγμα, αν στο κείμενο υπάρχουν οι λέξεις “Jacques Chirac” δεν περιμένουμε να περιλαμβάνονται και οι όροι “European -> Political Leader” ή ακόμα “Region -> Europe -> France” που θα ήταν κάποιες από τις σωστές ιεραρχίες από facets. Αντί αυτού θα μπορούσαμε να χρησιμοποιήσουμε εξωτερικούς πόρους όπως το WordNet, Wikipedia κτλ. για να αναζητήσουμε εκεί τους όρους και με βάση τα αποτελέσματα της αναζήτησης να τους κατατάξουμε σε ιεραρχίες από facets.

Οι λύσεις με αυτοματοποιημένο υπολογισμό των facets είναι ακόμα σε ερευνητικό στάδιο και δεν περιλαμβάνονται στα υπάρχοντα συστήματα.

3.9.3 Αξιοποίηση των Linked Open data

Το entity search (αναζήτηση οντοτήτων) είναι η διαδικασία αυτοματοποιημένης ανάκτησης των οντοτήτων και των σχέσεων που τις διέπουν από ένα σύνολο δεδομένων. Σε αντίθεση με την κλασική αναζήτηση στο web όπου τα αποτελέσματα είναι απλοί σύνδεσμοι εμπλουτισμένοι με μια περιγραφή (snippet) και άλλα πληροφοριακά στοιχεία, στο entity search τα αποτελέσματα προέρχονται από μια διαδικασία που εμπλέκει το information retrieval και τα linked data δηλαδή δεδομένα σε μορφή κατανοητή από τεχνολογίες του σημασιολογικού ιστού.

Γενικά υπήρξαν διάφορες προσπάθειες για την ανάκτηση entities από Linked data σε πραγματικό χρόνο. Κάποιες από αυτές υλοποιήθηκαν σε project όπως το EntityCube της

Microsoft [6] και το MediaFaces που ενσωματώθηκε στις υπηρεσίες του Yahoo. Στην πράξη το πρόβλημα που αντιμετωπίζουν αυτά τα συστήματα είναι ο μεγάλος επιπλέον υπολογιστικός φόρτος που προκαλεί καθυστέρηση και καθιστά τις μεθόδους μη ρεαλιστικές για εμπορική χρήση.

Ένα τέτοιο σύστημα θα είχε θετική επίδραση στη διαδικασία της αναζήτησης γιατί ο χρήστης δεν θα χρειαζόταν πια να αναζητήσει την πληροφορία στα επιστρεφόμενα αποτελέσματα ανοίγοντας διαδοχικά σελίδες αποτελεσμάτων. Αντί αυτού, θα είναι σε θέση να αναγνωρίσει τις κατηγορίες και τις ονομασίες των οντοτήτων γιατί εύκολα τις συσχετίζει με τον πραγματικό κόσμο και την εμπειρία του. Επιπλέον οι οντότητες που θα επέστρεφε το σύστημα θα ήταν για τον χρήστη μία ένδειξη καλής ή κακής διατύπωσης του ερωτήματός του.

Όμως για να ήταν χρήσιμο ένα τέτοιο σύστημα θα έπρεπε να αποδίδει σωστά σε δύο σημεία:

- Να λειτουργεί σε πραγματικό χρόνο. Ο χρόνος εκτέλεσης και παρουσίασης των αποτελεσμάτων είναι σήμερα ένα κρίσιμο χαρακτηριστικό όλων των συστημάτων αναζήτησης. Ένα τέτοιο σύστημα στην παραγωγή θα πρέπει να μπορεί να χαρακτηριστεί real-time έχοντας άμεση απόκριση στα ερωτήματα του χρήστη.
- Να γίνεται κατάλληλη επιλογή των οντοτήτων και να προσδιορισθούν κριτήρια για την ορθή κατάταξη (ranking) των οντοτήτων.

Οι Fafalios, Kitsos, Marketakis, Baldassarre, Salampasis, & Tzitzikas (Fafalios, P., Kitsos, I., Marketakis, Y., Baldassarre, C., Salampasis, M., & Tzitzikas, Y.,2012) προτείναν ένα υβριδικό σύστημα στο οποίο το ερώτημα του χρήστη θα εκτελείται αρχικά από μία μηχανή αναζήτησης και από τα αποτελέσματα θα διαλέγονται τα πρώτα snippets τα οποία θα αποτελούν τα έγγραφα της ανάλυσης σε συνδυασμό με τα linked data.

Τα πειράματα της ομάδας έδειξαν ότι η λύση αυτή έχει το πλεονέκτημα ότι αποδίδει ικανοποιητικά όσον αφορά τον χρόνο της εκτέλεσης. Για ένα ερώτημα σε 50 snippets χρειάστηκε συνολικά περίπου 1,5 δευτερόλεπτα εκτέλεσης. Αντίθετα με αυτό, για ένα ερώτημα στα 50 πλήρη έγγραφα απαιτήθηκαν περίπου 78 δευτερόλεπτα, χρόνος μάλλον απαγορευτικός για ρεαλιστική χρήση.

Το δεύτερο κριτήριο είναι ο αριθμός των οντοτήτων που επιστρέφει ένα ερώτημα στα snippets και κατά πόσο προσεγγίζει τις οντότητες του πλήρους εγγράφου. Όπως είναι αναμενόμενο τα snippets είναι υποσύνολο του εγγράφου άρα και οι οντότητες σε αυτά θα είναι υποσύνολο των οντοτήτων του εγγράφου. Όμως διαπιστώθηκε πολύ μεγάλη απόκλιση

[6] http://entitycube.research.microsoft.com/

μεταξύ τους, όπως σε μία περίπτωση που το ερώτημα στα snippets επέστρεψε 18 οντότητες ενώ στα πλήρη έγγραφα 527!

Συμπερασματικά οι μέθοδοι αυτόματης ανάκτησης των οντοτήτων σε συνδυασμό με δεδομένα σημασιολογικού ιστού έχουν μεγάλες προοπτικές, ωστόσο δεν είναι ακόμα μία ώριμη λύση.

3.10 Σύγκριση με άλλες τεχνολογίες

Η σημερινή κατάσταση για την ανάκτηση πληροφορίας χαρακτηρίζεται από πολλές και διαφορετικές τεχνολογίες οι οποίες ενσωματώθηκαν στα διάφορα συστήματα αναζήτησης προκειμένου να ικανοποιήσουν την ανάγκη πληροφόρησης. Θα μπορούσαμε να συνοψίσουμε τη μορφή των συστημάτων αναζήτησης στα εξής:

- **Συστήματα με χρήση γλώσσας ερωτημάτων** (**query languages**). Στα συστήματα με χρήση γλώσσας ερωτημάτων ο χρήστης παρέχει κάποιου είδους ερώτημα και το σύστημα απαντάει σε αυτό επιστρέφοντας ένας πλήθος αποτελεσμάτων. Σε αυτή την κατηγορία εντάσσεται το κλασικό Information retrieval. Τα ερωτήματα μπορεί να διατυπώνονται σε ελεύθερο κείμενο ή με λέξεις κλειδιά και να απευθύνονται σε δομημένα ή ημιδομημένα ή εντελώς αδόμητα δεδομένα. Επίσης σε αυτά περιλαμβάνονται συστήματα με χρήση SQL και natural language interface (NLI).
- **Συστήματα με κάποια δομή πλοήγησης** (**navigations structures**). Στα συστήματα αυτά ο χρήστης πλοηγείται από σημείο σε σημείο χρησιμοποιώντας συνδέσμους όπως στον παγκόσμιο ιστό. Συνήθως υπάρχει κάποια εσωτερική ιεραρχία ενώ τα δεδομένα μπορεί να έχουν από πριν καταταχθεί σε ενότητες, σύνολα και υποσύνολα ώστε να επιτρέπεται η αποτελεσματική πλοήγηση του χρήστη. Τέτοια συστήματα είναι οι κατάλογοι (directories), ιεραρχικά συστήματα όπως το σύστημα αρχείων, συστήματα γράφων όπως είναι ο παγκόσμιος ιστός και μεγάλες συγκεντρώσεις πληροφορίας όπως για παράδειγμα το Wikipedia.
- **Διαδραστικές όψεις** (**interactive views**). Στα συστήματα αυτά επιτρέπεται στον χρήστη να συνδιαλλάσσεται με το σύστημα χρησιμοποιώντας όψεις των δεδομένων. Οι όψεις αυτές παρέχουν έναν πολυδιάστατο τρόπο απεικόνισης της επιθυμητής πληροφορίας. Συστήματα που παρέχουν αυτή τη δυνατότητα είναι τα OLAP που προέρχονται από τον χώρο του data mining και τα συστήματα Faceted Search που επιτρέπουν την δημιουργία συλλογών αντικειμένων και το φιλτράρισμα σε αυτά.
- **Δέντρα αποφάσεων** (**decision trees**). Στα συστήματα αυτά χτίζεται μια δενδροειδής ιεραρχική μορφή στην οποία τα αντικείμενα κατατάσσονται με βάση τα

χαρακτηριστικά τους. Η κατάταξη είναι συνήθως μια αυτοματοποιημένη διαδικασία με βάση κάποιον αλγόριθμο.

- **Σημασιολογικός ιστός και Περιγραφική λογική** (**Semantic Web and Description Logic**). Ο σημασιολογικό ιστός κατά κάποιο τρόπο επεκτείνει το σημερινό web με τρόπο ώστε να είναι κατανοητός και αναγνώσιμος όχι μόνο από ανθρώπους αλλά και μηχανές. Οι περιγραφικές λογικές αναπτύχθηκαν για να καταγράψουν γνώση και να αποδώσουν τυπική σημασιολογία στις γλώσσες καταγραφής του σημασιολογικού ιστού. Γλώσσες όπως η SPARQL χρησιμοποιούν ερωτήματα για να ανακτήσουν πληροφορίες με σημασιολογικό περιεχόμενο.

Σε όλα τα παραπάνω συστήματα ανάκτησης γνώσης διακρίνουμε διαφορές τόσο σε λειτουργικό επίπεδο όσο και σε απόδοση. Αυτές αφορούν στα εξής:

- **Expressivity** (εκφραστικότητα). Ποιο εύρος ερωτημάτων δύναται να απαντηθεί; Με πιθανές τιμές:
  - Atom: ατομικές τιμές, αποτελέσματα
  - List: λίστα αποτελεσμάτων και κατάλληλοι τελεστές (UNION, Intersection)
  - Table: αποτέλεσμα σε μορφή πίνακα
  - Join: σχεσιακά join ή σχεσιακή άλγεβρα
- **Guidance** (καθοδήγηση). Οι χρήστες καθοδηγούνται κατά τη διαδικασία και προστατεύονται από dead-ends δηλαδή από κενά αποτελέσματα;
  - Με πιθανές τιμές NO/YES
- **Readability** (Αναγνωσιμότητα). Κατά πόσο είναι εύκολο στους χρήστες να κατανοήσουν εύκολα και να αλληλοεπιδράσουν με το σύστημα;
  - Με πιθανές τιμές technicists, specialists (domain knowledge), laymen (βασικές γνώσεις χρήσης υπολογιστών)
- **Scalability** (επεκτασιμότητα). Ποιο είναι το πλήθος των δεδομένων που διαχειρίζεται το σύστημα σε αποδεκτό χρόνο απόκρισης; Με πιθανές τιμές:
  - database: τα δεδομένα βρίσκονται σε έναν server
  - web: τα δεδομένα βρίσκονται διάσπαρτα παντού

Τα Guidance και Readability συμβάλουν στην χρηστικότητα (**Usability**) του συστήματος. Τα faceted search systems συγκρινόμενα με άλλες τεχνολογίες αναζήτησης έχουν αυξημένα όλα τα παραπάνω χαρακτηριστικά. Αυτοπεριγράφονται διότι παρουσιάζουν σε κάθε βήμα τις διαθέσιμες επιλογές προς τον χρήστη και προεπισκόπηση πληροφορίας σε συγκεντρωτική μορφή όπως το πλήθος αποτελεσμάτων για κάθε τιμή facet. Έχουν μεγαλύτερη ευελιξία από αυστηρά συστήματα που βασίζονται σε στατικές δομές ή σε συστήματα OLAP. Οι χρήστες αλληλοεπιδρούν πολύ εύκολα με το σύστημα σαν μία διαρκή διαδικασία επιλογών που

επαναϋπολογίζουν τα αποτελέσματα. Είναι ίσως ο καλύτερος τρόπος να διαχειριστούμε πολύ μεγάλα σετ δεδομένων γιατί έχουμε ταυτόχρονα τη δυνατότητα διαρκούς περιορισμού των αποτελεσμάτων ώστε να ικανοποιήσουμε την ανάγκη πληροφόρησης του χρήστη αλλά επίσης μηδενισμού των κενών αποτελεσμάτων γιατί οι διαθέσιμες επιλογές που παρουσιάζονται στον χρήστη έχουν τουλάχιστον ένα αποτέλεσμα. Στον πίνακα 1 συνοψίζονται τα χαρακτηριστικά:

approach	expressivity	guidance	readability	scalability
QL/SQL	table+join	no	technicists	database
QL/NLI	list+join	no	specialist	database
QL/keywords	list	no	laymen	Web
NS	atom	yes	laymen	Web
IV/OLAP	table	yes	specialist	database
IV/FS	list	yes	laymen	database

Πίνακας 1 : Σύγκριση διαφόρων τεχνικών αναζήτησης

4

Αποτελεσματική σχεδίαση web συστημάτων αναζήτησης

4.1 Παράγοντες που επηρεάζουν την διαδικασία της αναζήτησης πληροφορίας

Η σημερινή πραγματικότητα με τα πολλά εξειδικευμένα και μη συστήματα αναζήτησης μας προϊδεάζει σχετικά με την πολυπλοκότητα των συστημάτων αναζήτησης. Από τη μία μηχανές εξειδικευμένης αναζήτησης όπως η βιβλιοθήκη της IEEE και από την άλλη μηχανές γενικής αναζήτησης όπως η Google. Η πρώτη με ένα εξαρχής σύνθετο περιβάλλον διαχείρισης ενώ η άλλη με μια τάση απλούστευσης των διαδικασιών. Αν αναλογιστούμε τους λόγους της τόσο διαφορετικής προσέγγισης της κάθε μίας εύκολα θα καταλήξουμε σε δύο με τρεις διαπιστώσεις.

Διαπίστωση πρώτη: Η Google έχει δυνητικό χρήστη τον κάθε χρήστη του διαδικτύου, έμπειρο η μη, πιθανά με μικρή ή και καμία ικανότητα αξιολόγησης του αποτελέσματος της αναζήτησης. Η IEEE έχει τον ερευνητή, όχι απαραίτητα αλλά συνήθως με τεχνικές γνώσεις, ο οποίος πιθανότατα έχει τη ικανότητα αξιολόγησης του αποτελέσματος.

Διαπίστωση δεύτερη: Η IEEE ψάχνει ερευνητικά έγγραφα, βιβλία ή δημοσιεύσεις. Η Google ψάχνει τα πάντα. Έγγραφα, ιστοσελίδες, εικόνες είναι γνωστά παραδείγματα.

Διαπίστωση τρίτη: Η διαδικασία της αναζήτησης ενώ στην IEEE γίνεται μάλλον συνειδητά, στην Google ίσως γίνεται συγκαλυμμένα εντός κάποιας άλλη διαδικασίας όπως για παράδειγμα πλοήγηση σε χάρτη.

Τελικά, ακόμα και με την πρώτη ματιά αποκαλύπτονται πολλοί παράγοντες που επηρεάζουν το θέμα της σχεδίασης ενός συστήματος αναζήτησης. Η σωστή σχεδίαση είναι βέβαια υποκειμενική ωστόσο η διαδικασία σαφώς εμπλέκει πολλά στάδια τα οποία αναλύουμε παρακάτω. Οι Tony Russell Rose και Tyler Tate (Russell Tate, 2012) υποθέτουν ότι η αναζήτηση μπορεί να ταξινομηθεί και να κατανοηθεί καλύτερα μέσω ενός εύρους υποθέσεων όπως ποιος αναζητά, τι αναζητά, τι θέλει να επιτύχει και ποιες είναι οι περιστάσεις που επηρεάζουν τη διαδικασία. Τα ορίζουν ως τις τέσσερις διαστάσεις της αναζήτησης, που είναι οι εξής: Ο τύπος του χρήστη, Ο σκοπός της αναζήτησης, Το πλαίσιο της αναζήτησης και Ο τρόπος που αυτό υλοποιείται.

4.1.1 Ο τύπος του χρήστη

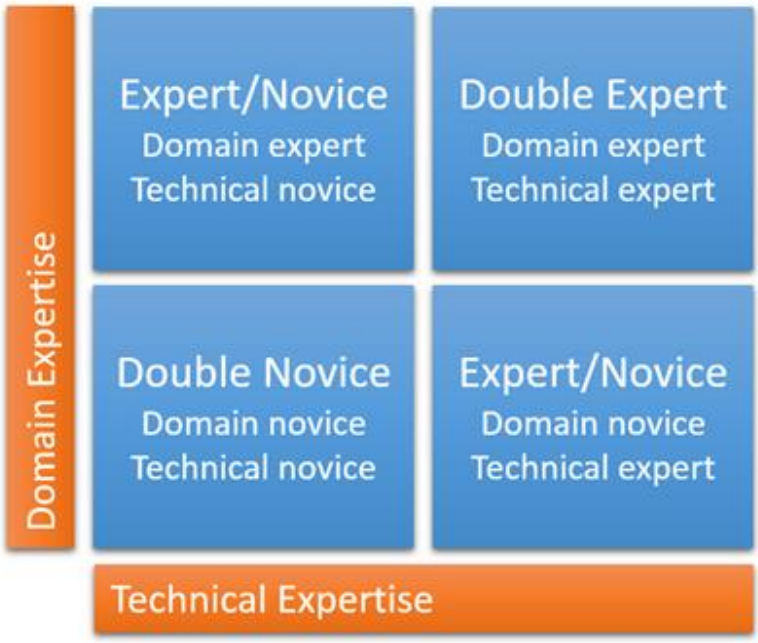
Συνήθως κατατάσσουμε την εμπειρία σε μία κατηγορία, ωστόσο στην αναζήτηση της πληροφορίας μας ενδιαφέρουν δύο διαφορετικοί τύπο εμπειρίας. Η μία αφορά στην εμπειρία του περιεχομένου και την γενικότερη εξοικείωση του χρήστη σε ένα αντικείμενο. Για παράδειγμα έναν γιατρός έχει την εμπειρία του περιεχομένου όταν αναζητά πληροφορίες για νέες μεθόδους της ειδικότητάς του. Ονομάζουμε την εμπειρία αυτή **Domain Expertise**. Η δεύτερη έχει να κάνει με την τεχνική εμπειρία όσον αφορά στην εξοικείωση χρήσης των εργαλείων αναζήτησης, υπολογιστών και άλλων. Ο ίδιος γιατρός ίσως δεν είναι εξοικειωμένος με το περιβάλλον αναζήτησης όπως ίντερνετ διαχείριση αρχείων σελιδοδεικτών και άλλα. Ονομάζουμε την εμπειρία αυτή Technical Expertise.

Η διαδικασία της αναζήτησης γίνεται ευκολότερη όταν και οι δύο εμπειρίες είναι σε υψηλό βαθμό αλλά με βάση αυτά έχουμε τέσσερις τύπους χρηστών:

- **Double experts**: Είναι οι πλέον έμπειροι χρήστες και "βουτάνε" άμεσα στην πληροφορία. Κύριο χαρακτηριστικό τους είναι η εξέταση περισσότερων σελίδων σε ένα αποτέλεσμα, η σε βάθος εξέταση χωρίς να επιστρέφουν στην αρχική σελίδα αναζήτησης και ο μικρότερος απαιτούμενος χρόνος.
- **Domain novice/technical novice:** Είναι οι πλέον άπειροι χρήστες που δεν έχουν ούτε τεχνική ευχέρεια με τους υπολογιστές ούτε την γνώση του προς αναζήτηση αντικείμενο. Κύριο χαρακτηριστικό τους είναι η ανάγκη να γυρίζουν στην αρχική σελίδα της αναζήτησης αναδιατυπώνοντας το αρχικό ερώτημα. Οι αλλαγές που κάνουν είναι συχνά μικρές λόγω του ότι φοβούνται τα λογικά άλματα που περιλαμβάνει η πλήρης αναδιατύπωση ενός ερωτήματος. Ξοδεύουν περισσότερο

χρόνο γιατί δεν έχουν την δυνατότητα σωστής αξιολόγησης των αποτελεσμάτων και δεν μπορούν να διαχειριστούν κατάλληλα τα υπολογιστικά εργαλεία.

- **Domain expert/technical novice**: Έχει τη δυνατότητα χρήσης προχωρημένης ορολογίας στον τομέα του και να αξιολογεί επιτυχώς τα αποτελέσματα της αναζήτησης. Δεν έχει τη δυνατότητα καλής χρήσης των υπολογιστών και εύκολα θα οδηγηθεί στην αρχική σελίδα αναζήτησης.
- **Domain novice/technical expert**: Αξιοποιεί σωστά τα εργαλεία αναζήτησης με προχωρημένες τεχνικές και προχωρά με εμπιστοσύνη στα αποτελέσματα. Δεν έχει όμως τη δυνατότητα σωστής αξιολόγησής τους.

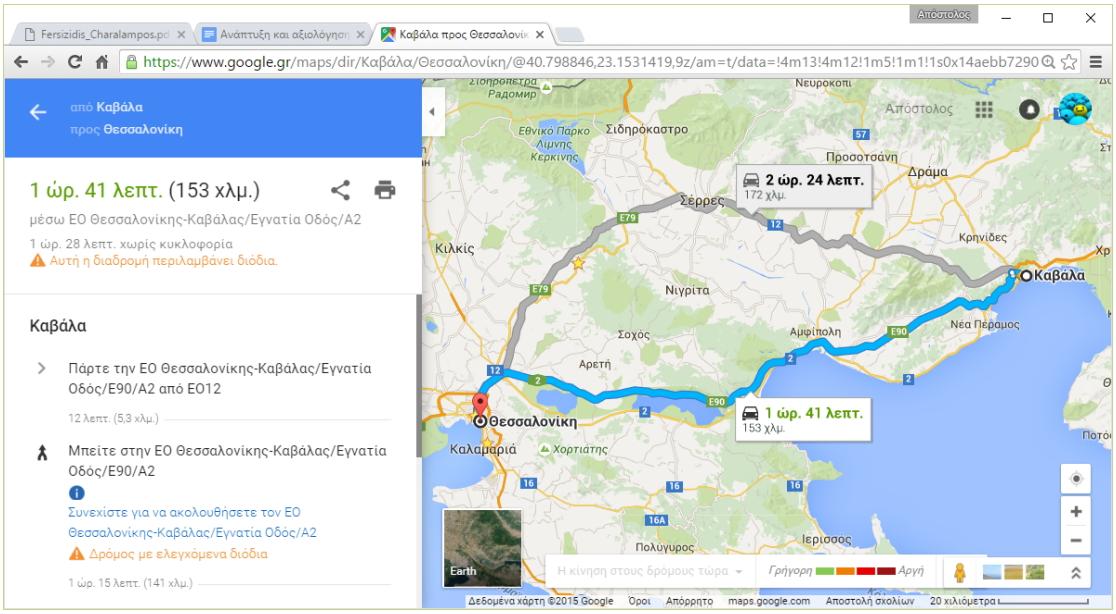


Εικόνα 11:Δύο είδη εμπειρίας του χρήστη

Μία ακόμα διάκριση αφορά στον χρήστη και έχει να κάνει με την **εσωτερική διαδικασία μάθησης**. Υπάρχουν αυτοί που ακολουθούν τα πάντα βήμα προς βήμα (serial thinkers) και αυτοί που επιμένουν να βλέπουν τη γενικότερη εικόνα (holistic thinkers) κάνοντας λογικά άλματα προς τα συμπεράσματα. Η δεύτερη κατηγορία έχει καλύτερες πιθανότητες επιτυχίας στην αναζήτηση.

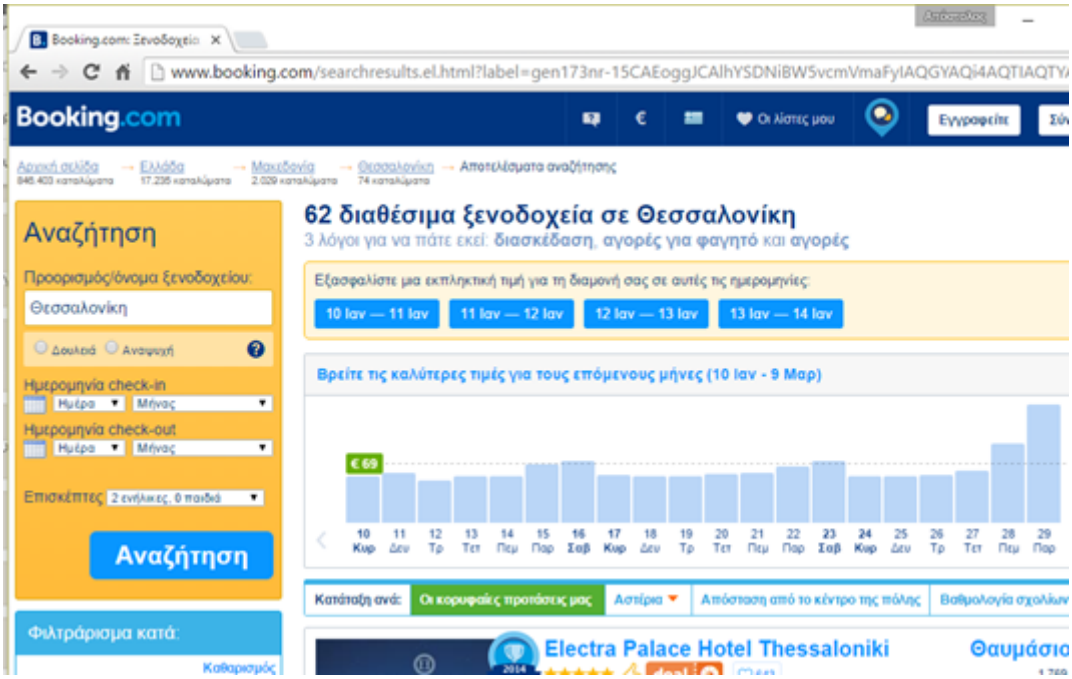
Οι χρήστες επίσης διαφοροποιούνται ως προς τον **τρόπο μάθησης**. Κάποιοι προτιμούν τον λεκτικό και κάποιο τον οπτικό. Οι χρήστες θα βρουν χρήσιμη την ιδέα το σύστημα να κάνει την μετατροπή κειμένων σε εικόνες και το αντίστροφο (Dual coding Theory). Για παράδειγμα, ένα σύστημα πλοήγησης είναι χρήσιμο να έχουμε τόσο τον χάρτη όσο και τα βήματα της διαδρομής σε κείμενο. Είναι μία διαδικασία που οι χρήστες την κάνουν από μόνοι τους εσωτερικά και προφανώς η κατάλληλη σχεδίαση επιταχύνει τη διαδικασία. Η σχεδίαση

των συστημάτων αναζήτησης θα πρέπει να έχει την τάση να ενοποιεί τις δύο αυτές κατηγορίες ενισχύοντας την μάθηση ως αποτέλεσμα της αναζήτησης.



Εικόνα 12: Dual Coding Theory στο Google Maps

Άλλος τρόπος είναι η σχεδίαση με overviews και reviews. Για παράδειγμα οι οπτικοί τύποι θα βρουν πολύ χρήσιμα τα previews προϊόντων σε ένα e-commerce με εικόνες και σύντομη παρουσίαση των προϊόντων. Τα overviews είναι λεκτικές ή οπτικές αναπαραστάσεις συγκεντρωτικής πληροφορίας που μπορούν να κατευθύνουν τον χρήστη ή να του δώσουν μια ιδέα κατά πόσο κινείται στη σωστή κατεύθυνση. Τέτοια μπορεί να είναι ένα διάγραμμα τιμών για μία αεροπορική γραμμή ή για κρατήσεις ξενοδοχείων.

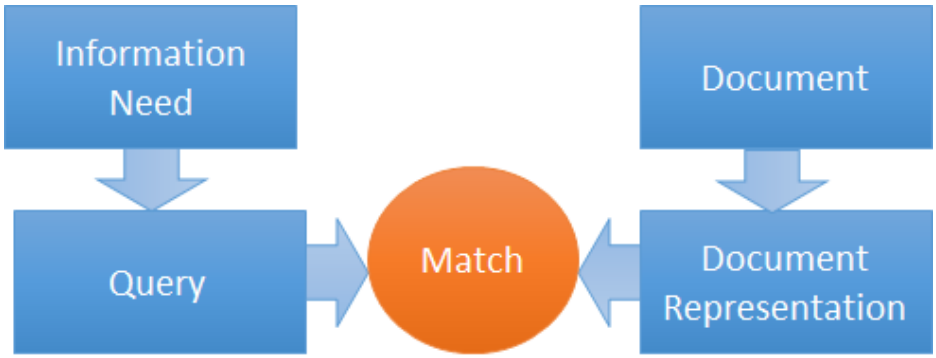


Εικόνα 13: Overviews αναζήτησης στο Booking.com

4.1.2 Ο σκοπός της αναζήτησης πληροφορίας

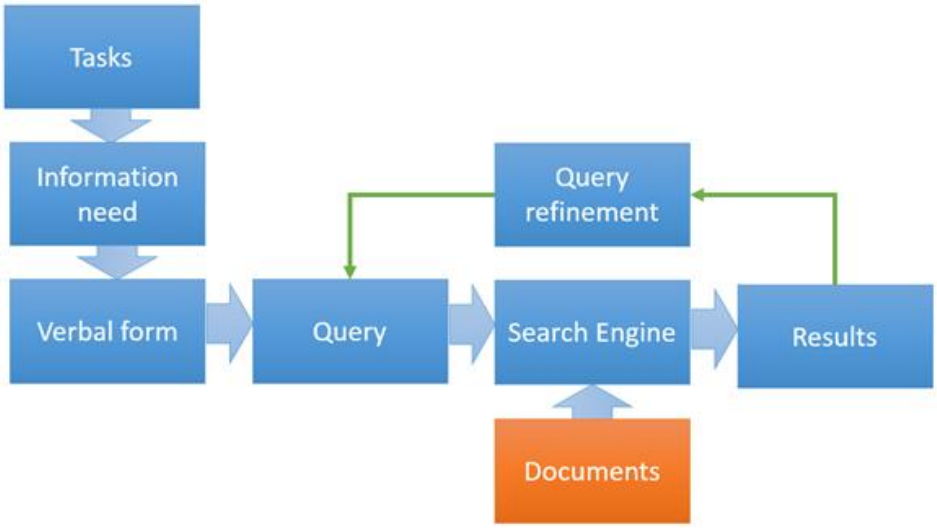
Η ανάκτηση της πληροφορίας είναι συνήθως ένα μικρό τεχνικό μέρος μιας γενικότερης διαδικασίας. Περιλαμβάνει συλλογές εγγράφων, τεχνικές αναζήτησης και αξιολόγησης αποτελεσμάτων και πολλά άλλα. Όμως, πάντα προηγείται η ανάγκη του χρήστη και η ικανοποίηση της ανάγκης αυτής μέσω της διαδικασίας ανάκτησης. Η ανάγκη γεννά ερωτήματα, ανακτά αποτελέσματα, αξιολογεί να δημιουργεί νέα ερωτήματα η αναδιατυπώνει τα προηγούμενα ώστε να ικανοποιήσουν με καλύτερο τρόπο την ανάγκη μας. Αυτό το ταξίδι μεταξύ ανάγκης και ικανοποίησης είναι η αναζήτηση πληροφορίες Information Seeking. Πέντε μοντέλα έχουν προταθεί κατά καιρούς για να περιγράψουν τη διαδικασία αυτή:

Το **classic model**: Από τα πρώτα που προτάθηκαν και χρησιμοποιήθηκαν ευρέως το οποίο όμως αγνοεί έναν σημαντικό παράγοντα, τον χρήστη.



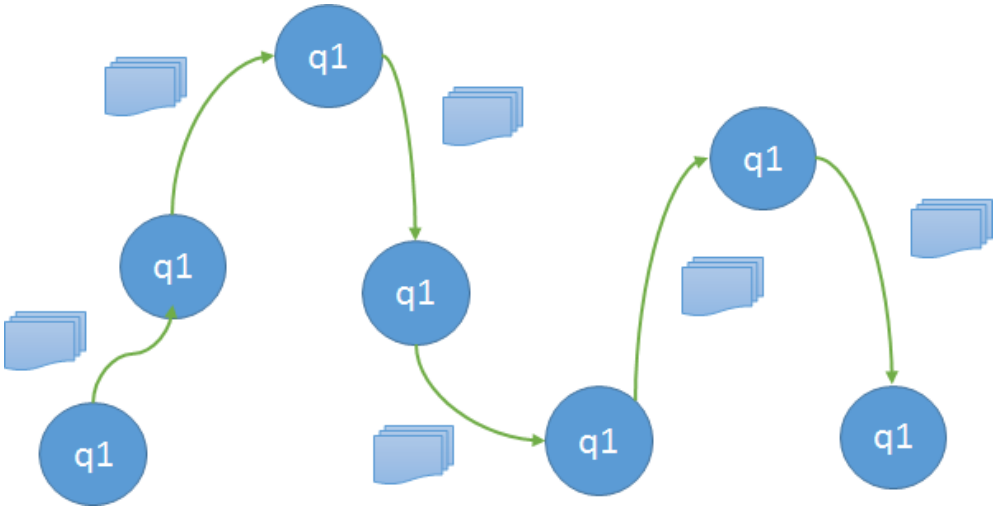
Εικόνα 14: Το classic model

Το **standard model**: Σε αντίθεση με το κλασικό μοντέλο δίνει έμφαση στον χρήστη παρομοιάζοντας την αναζήτηση πληροφορίας με ένα μοντέλο λύσης προβλήματος. Το ερώτημα επαναδιατυπώνεται μετά από αξιολόγηση των αποτελεσμάτων.



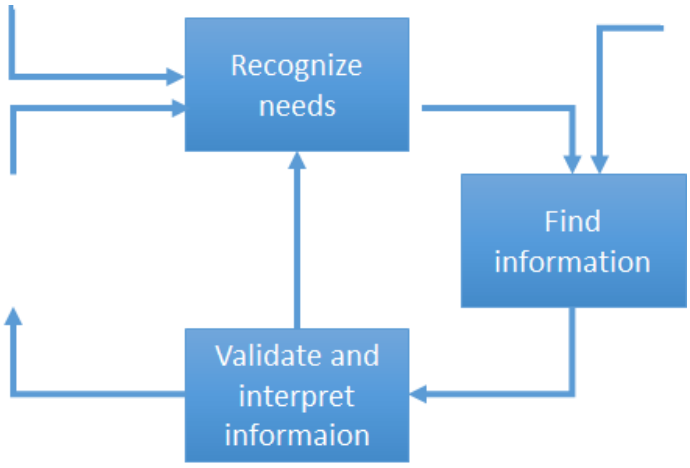
Εικόνα 15: Το standard model

Το **dynamic model**: Το δυναμικό μοντέλο βλέπει την διαδικασία της αναζήτησης πληροφορίας όχι απλά ως αναδιατύπωση των ερωτημάτων αλλά ως μια δυναμική διαδικασία κατά την οποία οι ανάγκες του χρήστη μεταβάλλονται όσο αυτός προχωράει προς τους στόχους του.



Εικόνα 16: Το δυναμικό μοντέλο

Το **information journey model**: άρχεται από τις ανάγκες του χρήστη και καταγράφει την συμπεριφορά του. Οι κύριες ενέργειες είναι τέσσερις και μεταξύ άλλων αναγνωρίζει την τυχαιότητα όπως και την πιθανή ξαφνική αλλαγή στη διαδικασία.



Εικόνα 17: Το information journey model

Όλα τα πιο πάνω μοντέλα επιχείρησαν να καταγράψουν τη διαδικασία της αναζήτησης πληροφορίας και τις εμπλεκόμενες ενέργειες. Η καταγραφή αυτή είτε απλοϊκή είτε πιο σύνθετη μας οδηγεί σε χρήσιμα συμπεράσματα για αποτελεσματικότερη σχεδίαση των web συστημάτων αναζήτησης όπως είναι τα εξής:

- Τίτλοι που ορθά περιγράφουν το περιεχόμενο. Οι τίτλοι πρέπει να είναι σχετικοί αλλά να διευκολύνουν τον χρήστη. Σε έρευνά τους οι (Spool et al, 2004) έδειξαν ότι η διαισθητική αναζήτηση των χρηστών είναι καλύτερη όταν οι τίτλοι είναι σωστά διατυπωμένοι χωρίς διαφημιστικά μηνύματα.
- Οι εμφατικοί (highlighted) σύνδεσμοι βοηθούν τους χρήστες να έχουν μια εικόνα του ιστορικού ειδικά όταν επιστρέφουν στην αρχική σελίδα αναζήτησης.
- Τα ξεκάθαρα μηνύματα στις ετικέτες επιτρέπουν στους χρήστες να αξιολογήσουν τα έγγραφα που τους αφορούν εστιάζοντας καλύτερα στις ανάγκες τους.

Η εύρεση σχετικών εγγράφων είναι ένα μέρος της αναζήτησης. Το άλλο μέρος αφορά στην αξιολόγησή τους και στην σύνδεσή τους με την αρχική ανάγκη του χρήστη. Ένα σημαντικό στοιχείο είναι η μνήμη των χρηστών και η σημασιολογία των εννοιών που παρουσιάζονται. Σε αυτό το στάδιο η συνολική αίσθηση του χρήστη (**sense making**) παίζει σημαντικό ρόλο βοηθώντας τον χρήστη να φιλτράρει τα αποτελέσματα που συνήθως είναι υπερβολικά πολλά και να καταλήξει σε λίγα σημαντικά.

Η διαδικασία περιλαμβάνει τέσσερα στάδια: την αναζήτηση, την εξαγωγή πληροφορίας, την σύνδεση των εννοιών και την ανάλυσή τους. Πάλι η κατάλληλη σχεδίαση μας βοηθάει τον χρήστη. Οι Pirolli και Card (Pirolli and Card, 2005) παρατήρησαν τρεις κοινές πρακτικές:

- **Shoebox.** Τα υποτιθέμενα σχετικά έγγραφα μαζεύονται σε μια κοινή συλλογή με σκοπό την περαιτέρω ανάλυσή του.
- **Evidence file.** Από την εξέταση των εγγράφων εντός του shoebox προκύπτουν τα στοιχεία που με μεγαλύτερη λεπτομέρεια σχετίζονται με την ανάγκη του χρήστη. Αυτά μπορεί να είναι μια πρόταση η ένα κείμενο κτλ.
- **Το schema**. Το εξωτερικό σχήμα συσχετίζει ακόμα καλύτερα τα ανακτηθέντα έγγραφα με τις ανάγκες μας. Μπορεί να τα συσχετίζει με πρόσωπα, περιοχές κτλ. χτίζοντας έτσι πρότυπα σύγκρισης για περαιτέρω ανάλυση.

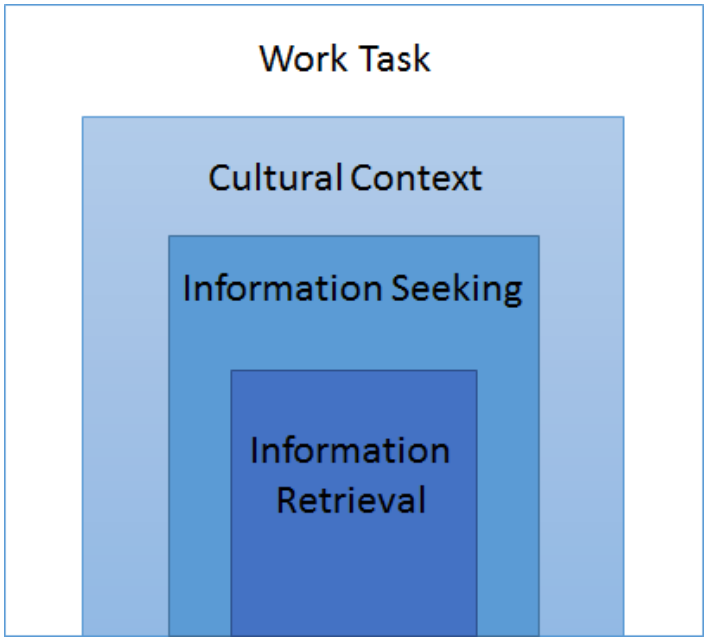
4.1.3 Το πλαίσιο της αναζήτησης

Η αναζήτηση πληροφορία είναι κάτι σαν ένα διάλογος μεταξύ του συστήματος και του χρήστη. Όπως όλοι οι διάλογοι βγάζει νόημα όταν έχουμε περιγράψει με σαφήνεια το πλαίσιο των αναγκών μας. Το πλαίσιο είναι "ποιοι είμαστε, με ποιους είμαστε και τι πόρους έχουμε στη διάθεσή μας". Διαφορετικά, το πλαίσιο περιγράφεται και ως "οποιαδήποτε γνώση χαρακτηρίζει μια κατάσταση στην οποία βρίσκεται κάποια οντότητα".

Εάν για παράδειγμα κάνουμε αναζήτηση της λέξης "Java" τι ψάχνουμε; Το νησί, τον καφέ η την γλώσσα προγραμματισμού; Εάν πάλι ψάχνουμε τις λέξεις "Ολυμπιακοί αγώνες", τι ακριβώς αναζητούμε; Πληροφορίες για την επόμενη διοργάνωση; Ιστορικά στοιχεία ή κάτι

άλλο; Γνωρίζοντας τον ρόλο, την τοποθεσία ή την πρότερη διαδικασία μπορούμε να στοχεύσουμε καλύτερα στις ανάγκες του χρήστη. Το πλαίσιο είναι το κλειδί για την επιτυχημένη αναζήτηση.

Οι Jarvelin και Ingwersen (Jarvelin και Ingwersen, 2004) πρότειναν ένα σετ από τέσσερα επίπεδα που προσδιορίζουν το πλαίσιο:



Εικόνα 18: Τα επίπεδα προσδιορισμού του πλαισίου αναζήτησης.

Στο επίπεδο information retrieval εστιάζουμε στην εύρεση εγγράφων σχετικών με ένα ερώτημα. Συνήθως αναζητούμε κάτι συγκεκριμένο (known item search). Η αξιολόγηση της διαδικασίας γίνεται με μετρικές όπως το precision και το recall και είναι καθαρά τεχνική διαδικασία.

Το επίπεδο Information seeking αφορά γενικότερες και πιο σύνθετες διαδικασίες. Για παράδειγμα όταν ένας χρήστης αναζητά ρούχα κατάλληλα για ένα δείπνο, η προς αναζήτηση πληροφορία δεν είναι ήδη γνωστή.

Στο επίπεδο work task γίνεται συσχέτιση της ανάγκης του χρήστη με το περιβάλλον της εργασίας του ή με προσωπικά κίνητρα. Η διαδικασία αυτή μπορεί να εξετάζει πρακτικές του χώρου εργασίας, περιορισμούς και διαθέσιμους πόρους.

Το επίπεδο Cultural επηρεάζει την γενικότερη διαδικασία και τις απαιτήσεις του χρήστη. Για παράδειγμα η ίδια διεργασία αναζήτησης αντιμετωπίζεται με άλλον τρόπο στα πλαίσια μια μικρής ή μιας μεγάλης επιχείρησης.

Τα παραπάνω επίπεδα μας δίνουν ένα πρίσμα το οποίο μπορεί να εκμεταλλευτούμε κατά τη σχεδίαση ενός συστήματος αναζήτησης. Όσο προχωράμε στην αναζήτηση το Information seeking αναλύεται σε information retrieval tasks. Για παράδειγμα ο γνωστός δικτυακός τόπος

ebay στην αρχική του σελίδα παρουσιάζει προτάσεις για διάφορες κατηγορίες προϊόντων οι οποίες διαμορφώνονται δυναμικά βάσει των προτιμήσεων των χρηστών. Αυτό ίσως βοηθάει τον χρήστη στην αρχική του αναζήτηση. Σε δεύτερο χρόνο ο χρήστης μάλλον θα αναζητήσει προϊόντα συγκεκριμένα ή μη από την κατηγορία που τον ενδιαφέρει. Σε αυτό το στάδιο προχωράει στη ανάκτηση της πληροφορίας.

Τα τελευταία χρόνια με την ανάπτυξη των ασύρματων δικτύων και συσκευών έχουν προκύψει νέες απαιτήσεις. Για παράδειγμα η πλειοψηφία των αιτημάτων προς το Google Maps γίνεται ποια από κινητές συσκευές (Hughes, 2011). Η συμπεριφορά αυτή έχει συνέπειες στη διαδικασία της αναζήτησης. Ο χρόνος των χρηστών είναι περιορισμένος και δεν είναι εύκολο να διατυπωθούν σύνθετα ερωτήματα. Η "φυσική παρουσία" του χρήστη είναι πια πολύ σημαντική. Ολοένα και περισσότερο επιθυμούμε οι προτάσεις ενός συστήματος αναζήτησης να λαμβάνουν υπόψη τη γεωγραφική παρουσία του χρήστη. Η απόσταση είναι σημαντική και μπορεί να οριστεί ως η γεωγραφική απόσταση ή αλλιώς η ποσότητα του χρόνου που απαιτείται για την πρόσβαση στην θέση της πληροφορίας. Ο χρόνος που παρήχθησαν τα αποτελέσματα είναι επίσης σημαντικός. Η πιο πρόσφατη πληροφορία είναι μάλλον η πιο σχετική. Αρκετά συστήματα υιοθέτησαν έναν νέο τρόπο επικοινωνίας με τον χρήστη. Η πληροφορία αποστέλλεται όχι κατά άμεσα απαίτηση του χρήστη αλλά σύμφωνα με κάποια χαρακτηριστικά φυσικής παρουσίας σε συνδυασμό με κοινωνικά δίκτυα. Η υπηρεσία foursquare για παράδειγμα πληροφορεί τον χρήστη για τοποθεσίες check-in φίλων κτλ.

Επίσης, είναι γνωστό ότι το μεγαλύτερο μέρος της πληροφορίας βρίσκεται σε αδόμητες ή ημιδομημένες μορφές. Συστήματα που στηρίζουν την αναζήτηση σε δομημένα μοντέλα δεν ικανοποιούν τις ανάγκες μας και πρέπει να εκσυγχρονισθούν.

4.1.4 Ο τρόπος που υλοποιείται η αναζήτηση

Τελικά η αναζήτηση εμπλέκει πολύ περισσότερα από την εύρεση πληροφορίας και περιλαμβάνει την εξερεύνηση, την μορφοποίηση και συλλογή πληροφοριών. Για παράδειγμα κάποιος που ψάχνει ρούχα να ταιριάξει με ένα κοινωνικό γεγονός έχει πολλά περισσότερα από την αναζήτηση.

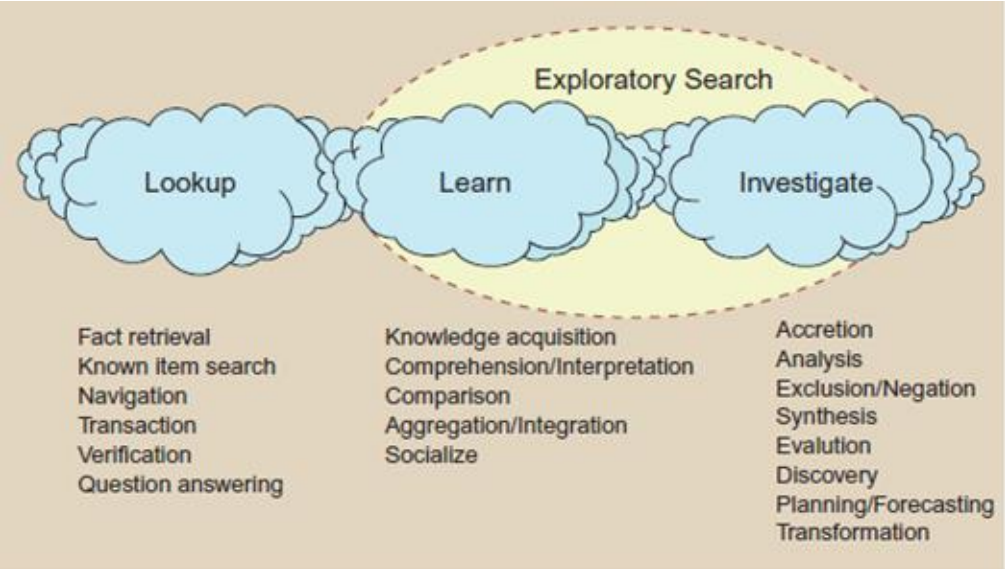
Η Donna Spencer (Spencer, 2006) πρότεινε τέσσερα στάδια αναζήτησης

- Known item. Ξέρουμε τι αναζητάμε και ψάχνουμε έγγραφα που να σχετίζονται με αυτό.
- Exploratory. Έχουμε μια ιδέα του τι ψάχνουμε αλλά όχι συνολική η ακριβής.
- Don't know what you need to know. Έχουμε ένα στόχο στο μυαλό μας αλλά στη διαδικασία ίσως τον αντικαταστήσουμε με άλλον.

- Refinding. Αν βρούμε κάτι το αναγνωρίζουμε αλλά δεν γνωρίζουμε που να ψάξουμε για αυτό.

Ο Marchionini (Marchionini, 2006) πρότεινε τρία συγκεντρωτικά στάδια στα οποία υλοποιείται η αναζήτηση.

- Lookup (Locate, verify, monitor)
- Learn (Compare, Comprehend, Explore)
- Investigate (Analyze, Evaluate, Synthesize



Εικόνα 19: Τρία στάδια υλοποίησης της αναζήτησης κατά Marchionini

Τα στάδια υλοποιούνται στη σχεδίαση των συστημάτων αναζήτησης όπως βλέπουμε στα παραδείγματα:

Για το στάδιο του Lookup, η άμεση συμπλήρωση ερωτημάτων της Google επιβεβαιώνει ένα χρήστη για το ερώτημά του. Επίσης η προκαταρκτική λίστα αποτελεσμάτων από ένα ερώτημα μπορεί να παρουσιάζει συγκεντρωτικές πληροφορίες στον χρήστη. Για παράδειγμα, η αναζήτηση σε ένα e-shop επιστρέφει προϊόντα με μια αρχική εικόνα και σύντομες πληροφορίες.

Στο στάδιο του Learn είναι σημαντική η αναζήτηση γενικότερης πληροφόρησης που θα εμπλουτίσει τη γνώση του χρήστη. Το σύστημα πρέπει να δίνει στον χρήστη την γενική εικόνα του που βρίσκεται. Οι χρωματισμένοι σύνδεσμοι βοηθάνε όπως επίσης και το ιστορικό από πρόσφατες αναζητήσεις και η χρήση προτάσεων όπως “See also”. Σημαντική βοήθεια μπορεί να παρέχει το faceted search. Μια τεχνική όπου βασικά χαρακτηριστικά των αποτελεσμάτων σχηματίζουν μενού και ο χρήστης μπορεί να περιορίσει περισσότερο τα αποτελέσματά του.

4.2 Η κατάλληλη σχεδίαση των web συστημάτων αναζήτησης

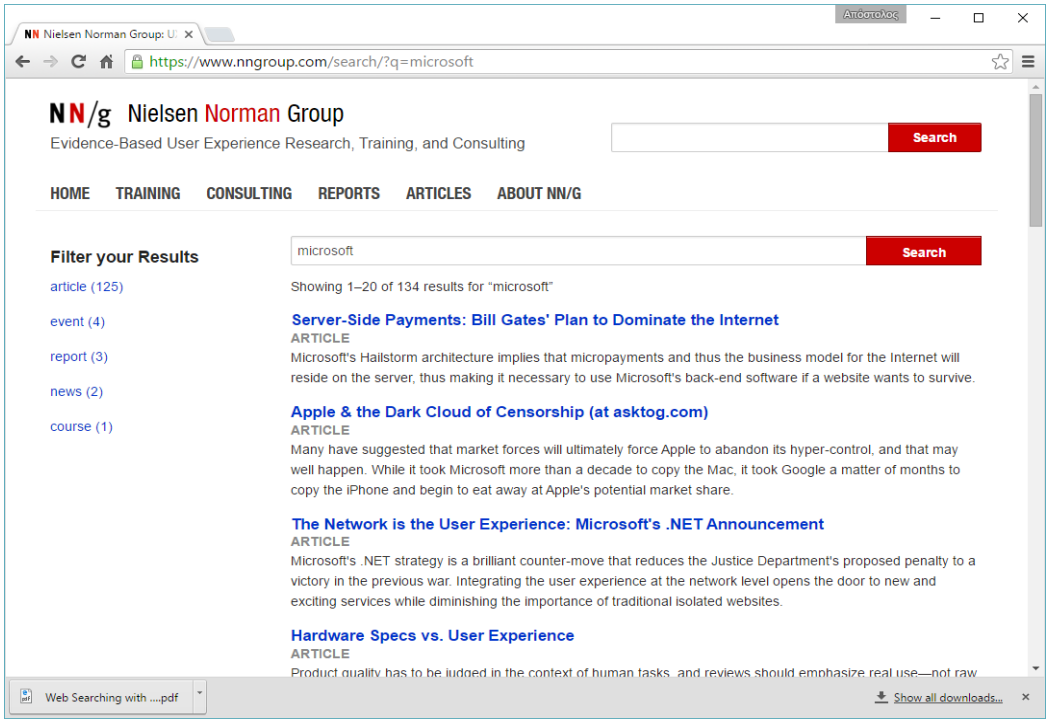
4.2.1 Διατυπώνοντας τα ερωτήματα

Η αναζήτηση είναι μία ολιστική εμπειρία με δυναμικό περιεχόμενο και έτσι πρέπει να την αντιμετωπίζουμε κατά τη σχεδίαση ενός συστήματος αναζήτησης. Σε κάθε ένα από τα μοντέλα αναζήτησης πληροφορίας σημείο έναρξης είναι η ανάγκη του χρήστη η οποία διατυπώνεται με κάποιο ερώτημα ή κάποια φόρμα. Τα διάφορα στοιχεία σε ένα σύστημα αναζήτησης πρέπει να αυτοπροσδιορίζονται κατάλληλα, δηλαδή να υλοποιούν αυτό που ο χρήστης αντιλαμβάνεται εξαρχής και όχι κάτι διαφορετικό ώστε να διευκολύνουν την εξοικείωση του χρήστη.

Το πλέον κοινό σημείο έναρξης αλληλεπίδρασης με τον χρήστη είναι ένα **search box**. Η χρήση του είναι πολύ διαδεδομένη ακριβώς λόγω της απλότητας που παρέχει ενώ η σχεδίασή του πρέπει να υπονοεί ξεκάθαρα τη λειτουργία του και να είναι αρκετά μεγάλο ώστε ένα μέτριο ερώτημα να διατυπώνεται με ευχέρεια. Τα περισσότερα συστήματα διευκολύνουν τη χρήση του τοποθετώντας αυτόματα τον κέρσορα σε αυτό και έχοντας ένα κουμπί ή γραφιστικό έναρξης της αναζήτησης δίπλα σε αυτό που συνήθως ενεργοποιείται με απλό enter. Μετά την έναρξη της αναζήτησης το ερώτημα παραμένει στο search box ώστε να είναι εύκολη η τροποποίησή του. Σε περιπτώσεις όπως σελίδες e-commerce όπου η αναζήτηση στοχεύει σε συγκεκριμένο περιεχόμενο και πρέπει να περιορίζεται σε αυτό, το search box είναι ένα εξαιρετικό σημείο εκκίνησης αλλά και παροχής υποστήριξης για τους χρήστες. Για παράδειγμα μπορεί να προτείνονται κατηγορίες προϊόντων ή να εμφανίζονται mega-menu με κατηγορίες-υποκατηγορίες που προτείνονται αντί της κλασικής αναζήτησης.

Οι περισσότεροι χρήστες μετά την αρχική αναζήτηση ψάχνουν στα σχετικά αποτελέσματα την πληροφορία που του ικανοποιεί. Η πρακτική αυτή να κάνουν αναζήτηση σε στάδια, οδήγησε στην λειτουργία τύπου **search within**. Σε αυτή ένα δεύτερο search box εμφανίζεται ξεκάθαρα σε διαφορετικό χώρο από το αρχικό και δίνει τη δυνατότητα αναζήτησης εντός των αρχικών αποτελεσμάτων. Ο χώρος συνήθως είναι ένα faceted navigation menu.

Αν και η ενοποίηση της αίσθησης του χρήστη εντός του search box είναι ζητούμενο, ωστόσο σε αρκετές εφαρμογές είναι εξαρχής σημαντικός ένα διαχωρισμός. Για παράδειγμα η αναζήτηση Java στον χώρο της πληροφορικής ή στον χώρο της γεωγραφίας. Η δυνατότητα αυτή που αναφέρεται ως **scoped search** μπορεί να είναι ενσωματωμένη στο search box. το οποίο δίνει τη δυνατότητα επιλογής κατηγορίας αποτελεσμάτων ανάλογα με τον χρήστη ή την περίπτωση.



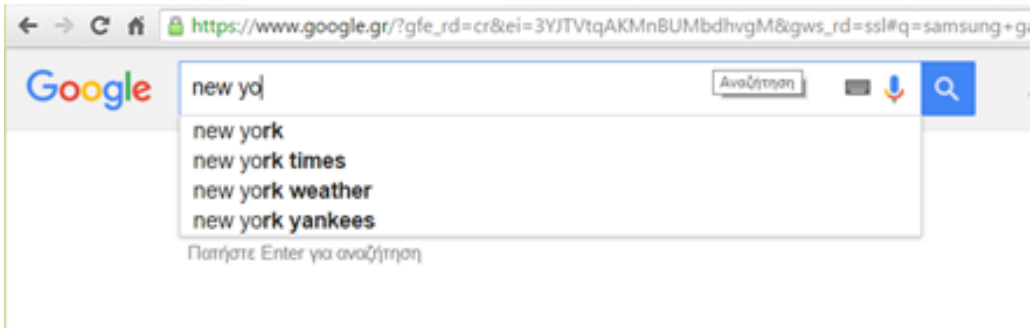
Εικόνα 20: Εφαρμογή search within στην αναζήτηση της NNG

Μια ακόμη δυνατότητα είναι το **advanced search** δηλαδή η χρήση μίας φόρμας επιλογών από τον χρήση που κατευθύνει το σύστημα και περιορίζει κατάλληλα τα αποτελέσματα πριν την αναζήτηση. Την δυνατότητα αυτή χρησιμοποιούν ακόμα και τώρα πολλές μηχανές αναζήτησης. Όμως ή λιγότερο φυσική διατύπωση των ερωτημάτων και η εξέλιξη των μηχανών αναζήτησης μείωσε σημαντικά την αξία του. Δεν πρέπει να ξεχνάμε ότι θέλουμε το σύστημα αναζήτησης να συνδιαλέγεται με τον χρήστη με τον πλέον φυσικό τρόπο.

Όπως είναι αναμενόμενο, η πρώτη εμπειρία με κάποιο σύστημα αναζήτηση περιλαμβάνει λέξεις κλειδιά και σύνθετες επιλογές, ωστόσο δεν είναι πάντα η πλέον κατάλληλη πρακτική. Κάποιες μηχανές αναζήτησης επιχείρησαν να αξιοποιήσουν την εξέλιξη των τεχνικών μηχανικής μάθησης και της Natural Language Processing (NLP) να χρησιμοποιήσουν όχι λέξεις κλειδιά αλλά φυσική γλώσσα ερωτημάτων, καμιά φορά σαν μια σειρά από ερωτήματα. Η χρήση τους όμως δεν είχε τα αναμενόμενα αποτελέσματα σε αντίθεση με άλλες τακτικές όπως το faceted search. Δεν θα πρέπει όμως να αποκλείσουμε την NLP σαν μια μορφή αλληλεπίδρασης με ένα σύστημα αναζήτησης αλλά να τις αξιοποιήσουμε στην σωστή τους μορφή. Σε επιβεβαίωση της διαπίστωσης αυτής, σχετικά πρόσφατες εξελίξεις επιτρέπουν την αλληλεπίδραση με δεδομένα σε μορφές διαφορετικές από κείμενα. Σημαντικό παράδειγμα είναι η αναζήτηση εικόνων της google που επιτρέπει την αναζήτηση μέσω μορφών και προτύπων. Ακόμα πιο εντυπωσιακή είναι η χρήση υπηρεσιών όπως το Shazam που επιτρέπουν την αναζήτηση μουσικής σε μία βάση δεδομένων της υπηρεσίας.

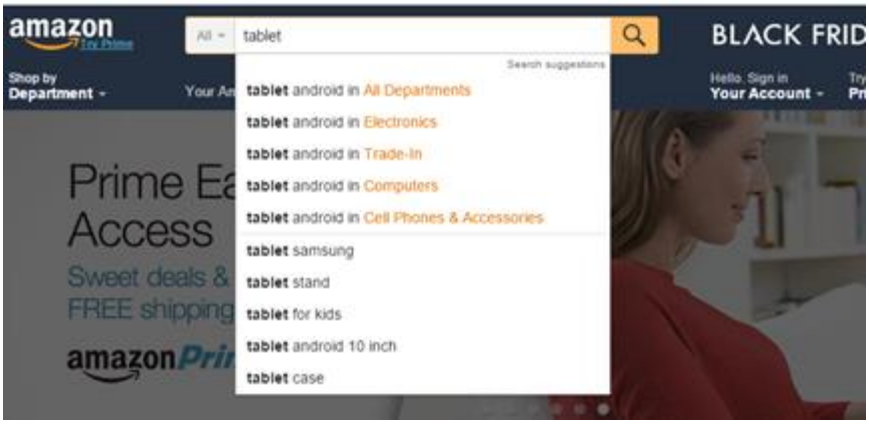
4.2.2 Επαναδιατυπώνοντας τα ερωτήματα

Η Google προσφέρει τη δυνατότητα στον χρήστη να πάει απευθείας στην χρήσιμη πληροφορία παρά να την ψάξει σε μία τεράστια λίστα αποτελεσμάτων. Η δυνατότητα αυτή προσφέρεται μέσω του κουμπιού "I'm feeling lucky" αλλά η επιτυχία του δεν είναι δεδομένη. Οι χρήστες συνήθως βλέποντας τα πρώτα αποτελέσματα αναδιατυπώνουν τα ερωτήματά τους ή αντιλαμβάνονται ότι ψάχνουν προς τη λάθος κατεύθυνση. Διάφορες τεχνικές επιτρέπουν ακριβώς αυτό, δηλαδή την καθοδήγηση του χρήστη προς τη σωστή πλευρά. Το autocomplete είναι μία από αυτές. Καθόσον ο χρήστης γράφει το ερώτημα, το σύστημα το συμπληρώνει λαμβάνοντας υπόψη συνήθως τις ενέργειες άλλων χρηστών. Το autocomplete μετατρέπει ένα μνημονικό σύστημα σε ένα σύστημα αναγνώρισης. Ο χρήστης δεν είναι υποχρεωμένος να θυμάται πολύπλοκες λέξεις κλειδιά γιατί το σύστημα θα τις συμπληρώσει αυτόματα στοχεύοντας στην ταχύτητα και την ακρίβεια των ερωτημάτων του χρήστη.



Εικόνα 21: Autocomplete στο Google

Ένα χαρακτηριστικό που μοιάζει με το auto complete είναι το auto suggest. Το auto suggest σκοπεύει να αναγνωρίσει λέξεις και προτάσεις και να προτείνει στον χρήστη όχι απλά ερωτήματα που ταιριάζουν λεκτικά με αυτό που γράφει αλλά και σχετικές φράσεις που αντλούνται από τις συνήθειες των χρηστών.



Εικόνα 22: Auto Suggest στο Amazon

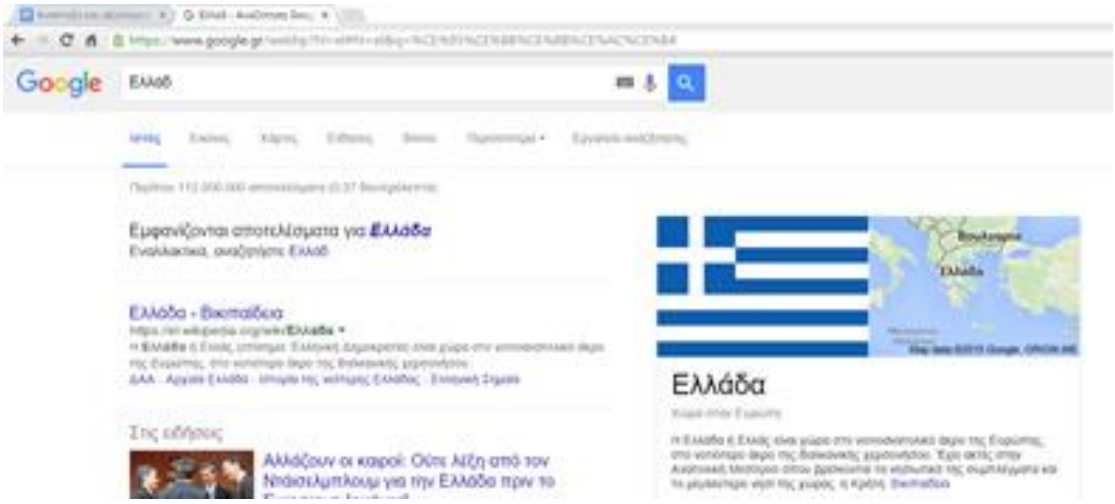
Μια πρόσφατη εξέλιξη επιτρέπει στο σύστημα να φέρνει άμεσα αποτελέσματα προς τον χρήστη καθόσον αυτός διατυπώνει το ερώτημα. Φέρνοντας άμεσα αποτελέσματα του

επιτρέπει να ελέγξει και την ακρίβεια του ερωτήματός του. Η τεχνική αυτή **instant results** και χρησιμοποιείται τόσο σε μηχανές αναζήτησης web αλλά και στην ενσωματωμένη αναζήτηση των λειτουργικών συστημάτων όπως στα windows 7.

Δύο ακόμα τεχνικές βοηθούν τον χρήστη στην κατάλληλη διατύπωση αλλά και στην διόρθωση των ερωτημάτων του.

Did you mean: Το σύστημα αναζήτησης χρησιμοποιώντας αλγόριθμους ελέγχου ορθογραφίας αναγνωρίζει σφάλματα και προτείνει την διόρθωσή τους. Ο χρήστης έχει τη δυνατότητα να επιλέξει την προτεινόμενη διόρθωση.

Autocorrect: Το σύστημα αναζήτησης αναγνωρίζει ένα ερώτημα ως λανθασμένο και το μεταβάλει αυτόματα. Σε αντίθεση με το did you mean η τεχνική αυτή επιλέγει την αυτόματα διορθωμένη έκφραση.



Εικόνα 23: AutoCorrect στο Google

4.2.3 Η παρουσίαση και η διαχείριση των αποτελεσμάτων

Η κατάλληλη παρουσίαση των αποτελεσμάτων της αναζήτησης είναι ένα επίσης σημαντικό μέρος της διαδικασίας. Η συνηθισμένη πρακτική είναι η χρήση κάποιας σελίδας αποτελεσμάτων με την οποία συνδιαλέγονται οι χρήστες κατά την διάρκεια της αναζήτησης της πληροφορίας. Με την κατάλληλη δομή και λειτουργία οι σελίδες αποτελεσμάτων μπορούν να υποστηρίξουν πολλές μορφές αναζήτησης. Ακόμα και όταν δεν υπάρχουν αποτελέσματα οι σελίδες αυτές μπορούν να έχουν ένα ενεργητικό ρόλο βοηθώντας τον χρήστη να επαναδιατυπώσει κατάλληλα το ερώτημά του. Οι ίδιες σελίδες μπορούν να καθοδηγούν τον χρήστη να επιλέξει ξεκάθαρα το είδος των αποτελεσμάτων που επιθυμεί όταν υπάρχει δυσκολία από το σύστημα αναζήτησης. Αυτές τις τακτικές θα εξετάσουμε σε συντομία παρακάτω.

Η βασική ιδέα μια σελίδας αποτελεσμάτων είναι να παρουσιάσει ένα μέρος από κάθε σχετικό έγγραφο ως αποτέλεσμα της αναζήτησης από τον χρήστη. Σε αυτό ακριβώς το σημείο εισέρχεται ο πρώτος προβληματισμός μας. Αν συμπεριλάβουμε υπερβολικά λίγη λεπτομέρεια σε κάθε αποτέλεσμα ο χρήστης αναγκάζεται να το ανοίξει και να το εξετάσει αναλυτικά. Αν πάλι συμπεριλάβουμε υπερβολικά πολύ λεπτομέρεια, δαπανούμε χώρο και ίσως μπερδεύουμε τον χρήστη με άχρηστη πληροφορία. Αυτό που θέλουμε είναι η ποσότητα της πληροφορίας που ενισχύει την κρίση του χρήστη ώστε να οδηγηθεί γρηγορότερα στην επιθυμητή πληροφορία.

Η δομή της σελίδας αποτελεσμάτων είναι σημαντική τόσο όσο και η παρουσίασή τους. Εξ ορισμού σε κάθε αποτέλεσμα περιλαμβάνεται το URL της πηγής και μια πληροφοριακή περίληψη (**snippet**). Την τακτική αυτή ακολουθούν όλες οι διαδεδομένες μηχανές αναζήτησης. Επιπρόσθετα, η επισήμανση των λέξεων κλειδιών από αυτές που συμπεριλάβαμε στην αναζήτηση τραβάει την προσοχή των χρηστών. Στην πραγματικότητα, η περίληψη που θα συμπεριλάβουμε μπορεί να διαφέρει ανάλογα με την εφαρμογή. Για παράδειγμα σε e-commerce θα χρειαστεί να συμπεριλάβουμε εικόνες των προϊόντων ενώ για ένα σύστημα αναζήτησης ειδήσεων, η ημερομηνία δημοσίευσης είναι πολύ σημαντική. Σε όλα αυτά να προσθέσουμε ότι τα αποτελέσματα μπορεί να μην είναι κείμενα. Για παράδειγμα η αναζήτηση εικόνων ή η αναζήτηση σε χάρτες. Οι προεπισκοπήσεις (previews) εμφανίζουν στα αποτελέσματα μια προεπισκόπηση του πραγματικού αποτελέσματος δηλαδή του σχετικού εγγράφου ενώ παράλληλα έχουν εφαρμογή σε αναζήτηση σε κοινωνικά δίκτυα.

Μερικές εφαρμογές βάζουν στα αποτελέσματα συνδέσμους που οδηγούν απευθείας σε ενέργειες του χρήστη. Για παράδειγμα, σε εφαρμογές e-commerce περιλαμβάνεται shortcut στην προσθήκη του προϊόντος απευθείας από τη σελίδα των αποτελεσμάτων.

4.2.3.1 Η σχεδίαση των σελίδων αποτελεσμάτων

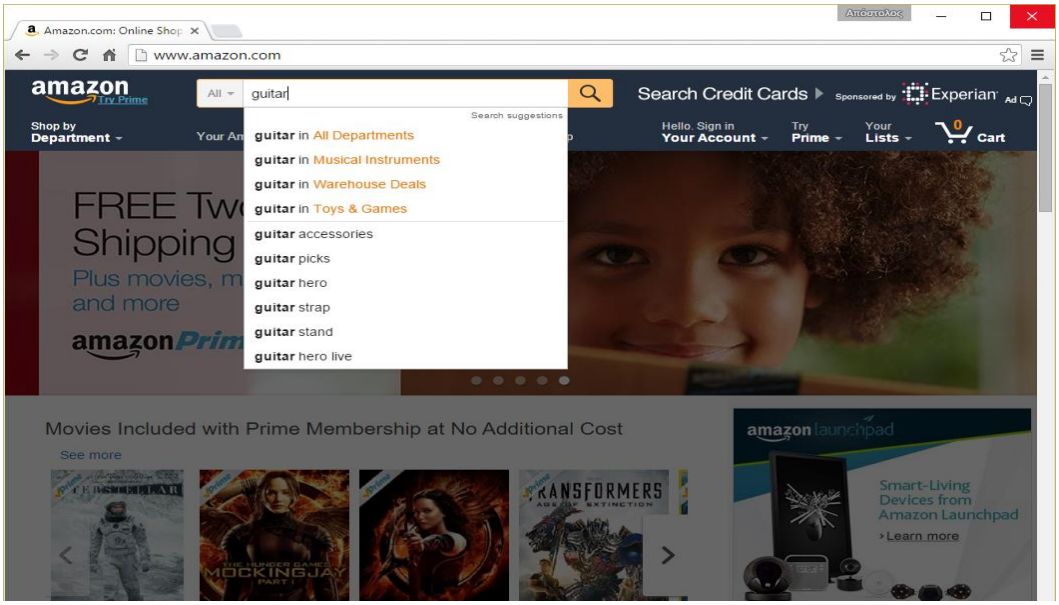
Η σελίδα με τα αποτελέσματα συνήθως έχει κάποια δομή (Wilson, 2011) που περιλαμβάνει τα εξής:

- Είσοδο με κάποιο Search box και κατά ακολουθία auto-suggest και άμεσα αποτελέσματα
- Περιγραφικές πληροφορίες για τα αποτελέσματα όπως, το πλήθος των αποτελεσμάτων και υποστήριξη για επαναδιατύπωσή τους.
- Τα ίδια τα αποτελέσματα που περιλαμβάνουν συνδέσμους για την πηγή του, previews, σχετικές με αυτά πληροφορίες όπως citations, παρόμοια άρθρα,
- Στοιχεία ελέγχου όπως facets, σελιδοποίηση
- Προσωποποιημένα στοιχεία όπως profile.

Είναι σημαντικό η σελίδα να παρουσιάζει την τρέχουσα κατάσταση όπως για παράδειγμα το query ή τα facets που επιλέχτηκαν και να επιτρέπει την αλλαγή σε αυτά. Το πλήθος των αποτελεσμάτων είναι μια σημαντική μετρική και βοηθάει τον χρήστη να έχει μία αρχική εκτίμηση για την επιτυχία της αναζήτησης όπως και στην επαναδιατύπωση του ερωτήματος.

Ίσως το πλέον γνωστό χαρακτηριστικό στις σελίδες των αποτελεσμάτων είναι η σειρά με την οποία εμφανίζονται τα αποτελέσματα. Είναι αναμενόμενο να έχει δοθεί μεγάλο βάρος σε αυτό γιατί έρευνες (Joachim et al, 2005) έχουν δείξει ότι οι χρήστες σπάνια εξετάζουν περισσότερα από τα πρώτα αποτελέσματα και ακόμα πιο σπάνια οδηγούνται στις επόμενες σελίδες. Λογικά η καλύτερη προσέγγιση θα ήταν "τα καλύτερα πρώτα". Η ποιοτική κατάταξη των αποτελεσμάτων εξαρτάται από πολλά κριτήρια όπως τα links άλλων σε αυτά, χαρακτηριστικά των εγγράφων και η τάση των άλλων χρηστών.

Μία πρόκληση είναι ο προβολή αποτελεσμάτων για όρους που δεν διακρίνεται το πεδίο ορισμού τους. Σε αναζητήσεις με λέξεις κλειδιά όπως "Java" το σύστημα δεν είναι εύκολα σε θέση να διακρίνει αν αναζητούμε πληροφορίες για το νησί ή για την γλώσσα προγραμματισμού. Σε τέτοιες περιπτώσεις θα μπορούσαμε να έχουμε "τα καλύτερα πρώτα" από κάθε πιθανή μετάφραση και οι τομείς των αποτελεσμάτων δημιουργούνται δυναμικά. Θα μπορούσαμε επίσης να ζητήσουμε από τον χρήστη να επιλέξει την κατηγορία που τον ενδιαφέρει.



Εικόνα 24: Εμφάνιση κατηγοριών στο search box του Amazon

Σε κάποιες περιπτώσεις αυτό δεν είναι η κατάλληλη τεχνική, αλλά προτιμάται η παρουσίαση αποτελεσμάτων ομαδοποιημένα κατά κατηγορία. Το google για παράδειγμα ομαδοποιεί κατά σελίδες, βίντεο, εικόνες κτλ. Τα αποτελέσματα περιλαμβάνουν δεδομένα με τα οποία μπορούμε να τα ομαδοποιήσουμε αξιοποιώντας τα facets σχηματίζοντας έτσι δισδιάστατους

εννοιολογικούς χώρους. Το faceted search είναι η τεχνική που χρησιμοποιεί αυτή την προσέγγιση φιλτραρίσματος και βελτίωσης των αποτελεσμάτων.

Η τακτική των περισσότερων μηχανών αναζήτησης είναι να παρουσιάζουν τα αποτελέσματα σε σελίδες με κατακόρυφη διάταξη δηλαδή το ένα κάτω από το άλλο αν και μερικές φορές ταιριάζει καλύτερα η οριζόντια παρουσίαση. Επίσης, σε περιπτώσεις όπως αναζήτηση προϊόντων e-commerce βολεύει να έχουμε ένα δισδιάστατο πλέγμα αποτελεσμάτων που συνδυάζουν εικόνες και προεπισκόπηση κειμένων. Τι συμβαίνει όμως όταν δεν έχουμε καθόλου αποτελέσματα; Αν και μια άδεια σελίδα αποτελεσμάτων είναι ένδειξη για μια κακή αναζήτηση ωστόσο είναι ένα ενδιαφέρον σημείο για να βοηθήσουμε τον χρήστη να επαναδιατυπώσει το ερώτημά του με τεχνικές όπως "did you mean …".

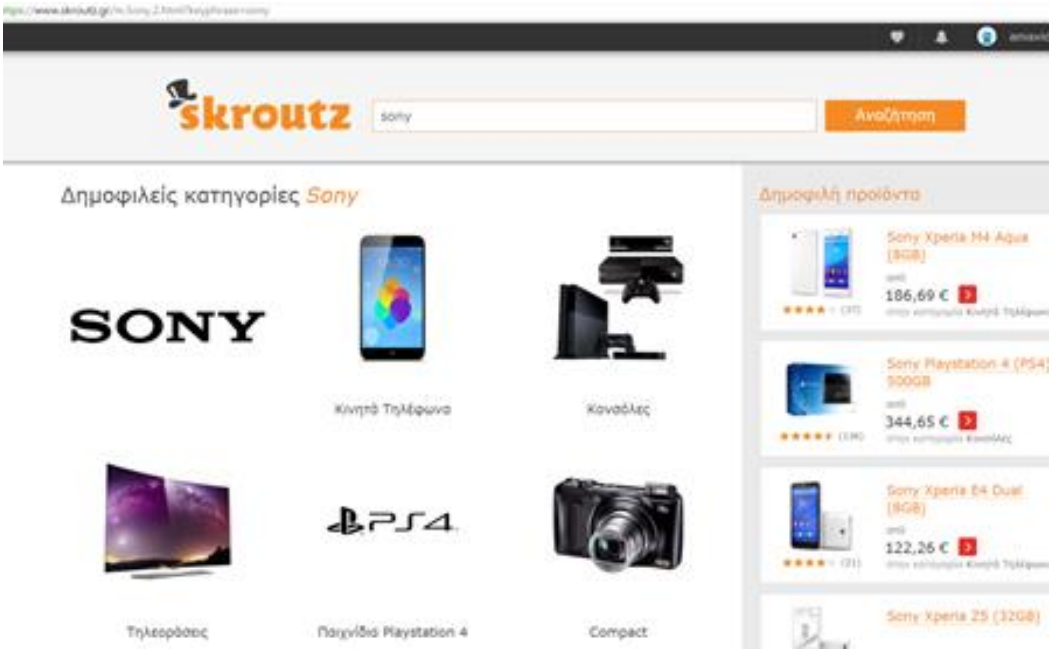
4.2.3.2 Διαχείριση των αποτελεσμάτων

Όπως συζητήθηκε παραπάνω, η αναζήτηση δεν είναι απλά εύρεση εγγράφων και μπορεί να περιλαμβάνει μια σειρά από επαναλαμβανόμενα βήματα όπως η ανάλυση, η σύγκριση, η αξιολόγηση κτλ. Σε κάθε περίπτωση πρέπει να διαχειριστούμε κατάλληλα τον μεγάλο όγκο αποτελεσμάτων.

Η σελιδοποίηση είναι ένας συνηθισμένος τρόπος αντιμετώπισης του υπερβολικά μεγάλου όγκου. αποτελεσμάτων. Προσφέρει δύο πλεονεκτήματα, μειώνει τον χρόνο φόρτωσης των αποτελεσμάτων και παρέχει ένα πρακτικό μέτρο εκτίμησης του όγκου. Σε αντίθεση με τη σελιδοποίηση το συνεχόμενο scroll των αποτελεσμάτων δεν είναι πρακτικό ειδικά σε φορητές συσκευές και δεν επιτρέπει τα bookmarks από τον χρήστη.

Ο δεύτερος τρόπος αντιμετώπισης του φόρτου είναι το φιλτράρισμα και η ταξινόμηση των αποτελεσμάτων. Η διάταξη λογικά σχετίζεται με την ταύτιση των αποτελεσμάτων ως προς το ερώτημα. Θέλουμε να μπορούμε να την αλλάξουμε για να προβάλλουμε τα αποτελέσματα σε άλλη διάσταση όπως π.χ. τα πιο φτηνά ή τα πιο νέο κτλ. Επιπρόσθετα το φιλτράρισμα επιτρέπει στον χρήστη να εστιάσει καλύτερα στις ανάγκες του. Η σελιδοποίηση κατά κάποιο τρόπο λειτουργεί ως ένα de facto φίλτρο στα πρώτα καλύτερα αποτελέσματα. Μια υβριδική προσέγγιση περιλαμβάνει ταξινόμηση μαζί με φιλτράρισμα όπως αυτή που ακολουθείται στα App Store με τις επιλογές "Top paid", "Top free" κτλ. Ο πλέον αποτελεσματικός τρόπος φιλτραρίσματος είναι η αξιοποίηση των facets που θα εξετάσουμε παρακάτω.

Ένας άλλος τρόπο αντιμετώπισης του όγκου είναι η προσπάθεια προσέγγισης των απαιτήσεων του χρήστη κατά τη διάρκεια του ερωτήματος η οποία βελτιώνει την απόδοση του συστήματος αναζήτησης (Tunkelang, 2009). Υπάρχουν διάφοροι τρόποι όπως η εμφάνιση κατηγοριών στο search box, ή μια πρώτη σελίδα αποτελεσμάτων που εμφανίζει αποκλειστικά κατηγορίες τις οποίες επιλέγει ο χρήστης.



Εικόνα 25: Σελίδα αποτελεσμάτων με προτάσεις κατηγοριών στο Skroutz.gr

Το τελευταίο χαρακτηριστικό που δίνει στον χρήστη τη δυνατότητα καλύτερης προσέγγισης στα επιθυμητά αποτελέσματα είναι η δυνατότητα σύγκρισης των αποτελεσμάτων. Η σύγκριση για παράδειγμα προϊόντων επιτρέπει την άμεση αντιπαράθεση των χαρακτηριστικών τους, όμως όπως είναι αναμενόμενο το πλήθος των αποτελεσμάτων στη σύγκριση πρέπει να είναι πολύ περιορισμένο.

4.3 Σχεδίαση συστημάτων αναζήτησης με faceted search

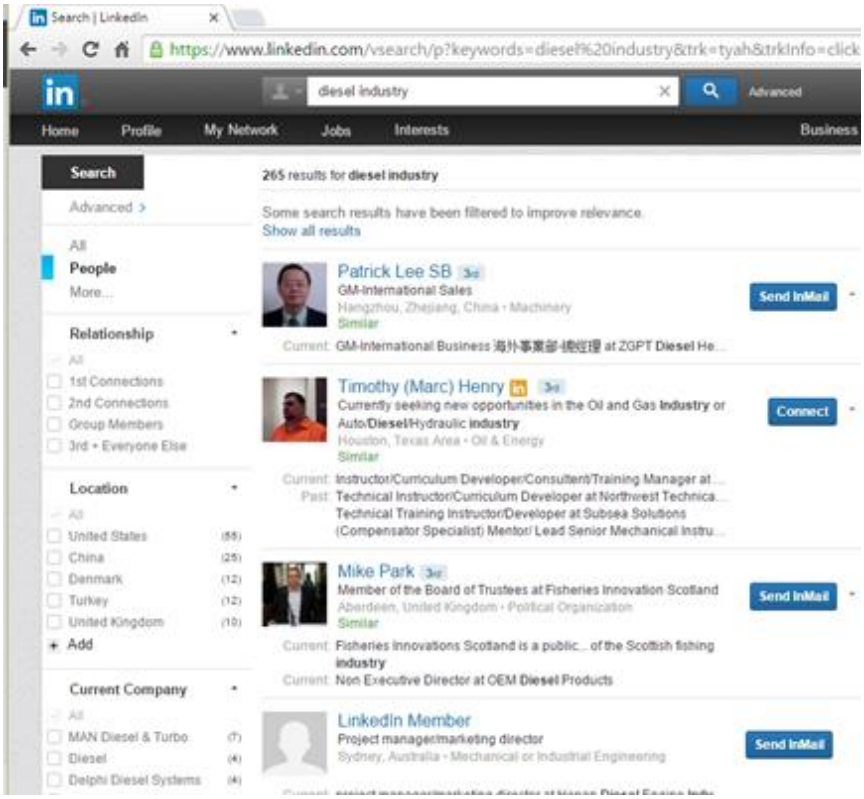
Στο προηγούμενο μέρος μελετήσαμε κάποια σχεδιαστικά ζητήματα που αφορούν στην υλοποίηση web συστημάτων αναζήτησης. Στο κεφάλαιο αυτό θα προσθέσουμε τον παράγοντα faceted search στον σχεδιασμό των συστημάτων μας. Το faceted search παρέχει έναν μοναδικό τρόπο να μεταμορφώσουμε συνολικά την εμπειρία του χρήστη με το σύστημα αναζήτησης. Παρέχει ένα ευέλικτο πλαίσιο ικανοποίησης ενός μεγάλου εύρους απαιτήσεων του χρήστη τόσο σε συνθήκες exploratory search όσο και σε σύνθετα προβλήματα ανακάλυψης νέας γνώσης. Ειδικά εάν το συνδυάσουμε με τις γνωστές πρακτικές αναζήτησης με λέξεις κλειδιά έχουμε ένα πολύ ισχυρό εργαλείο.

4.3.1 Διάταξη layout

Το faceted search υλοποιείται με κάποιο αντικείμενο στη διεπαφή του χρήστη που ονομάζουμε faceted navigation menu. Η διάταξή του μπορεί να είναι κατακόρυφη, οριζόντια και υβριδική.

Η **κατακόρυφη διάταξη** είναι η πλέον συνηθισμένη. Το πλεονέκτημα με αυτή τη διάταξη είναι ότι είναι κατάλληλη ακόμα και για μεγάλο αριθμό facets αλλά και να είναι συμβατή με την συνηθέστερη διάταξη των αποτελεσμάτων που είναι επίσης η κατακόρυφη παράθεση. Βέβαια, ο υπερβολικά μεγάλος αριθμός των facet αναγκάζει τον χρήστη σε συνεχή scroll και έχει ανεπιθύμητα αποτελέσματα στην αλληλεπίδραση με το σύστημα. Δεν πρέπει να ξεχνάμε ότι τα Facets είναι μέρος ενός συστήματος αναζήτησης. Όπως και στα αποτελέσματα όπου ο χρήστης πρακτικά ψάχνει μόνο στα πρώτα με το υψηλότερο rank, έτσι και στα facets εάν είναι υπερβολικά πολλά κανείς δεν θα δώσει σημασία στα τελευταία στη λίστα.

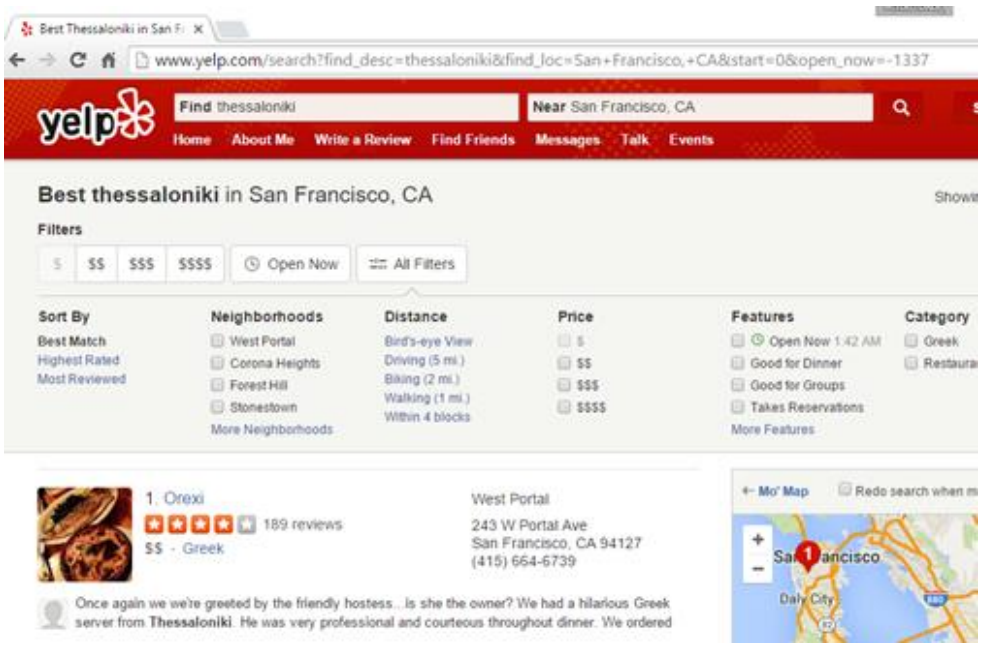
Η τοποθέτηση του κατακόρυφου μενού στα δεξιά δεν επηρεάζεται και από τις αυξομειώσεις των παραθύρων ανάγνωσης.



Εικόνα 26: Κατακόρυφα facets στο Linkedin.com

Η **οριζόντια διάταξη** συνήθως τοποθετεί τα facets στην κορυφή της σελίδας ακριβώς πάνω από τα αποτελέσματα και προκαλεί τον χρήστη να διαλέξει τις κατάλληλες τιμές. Έχει όμως το πρόβλημα της μη καλής προσαρμογής στην αλλαγή μεγέθους το χρήστη και στο πλήθος των facets που μπορεί να φιλοξενήσει.

Ενώ οι περισσότερες υλοποιήσεις χρησιμοποιούν κατακόρυφη ή οριζόντια διάταξη για το facet navigation menu, ωστόσο υπάρχουν και **υβριδικές υλοποιήσεις**. Σε αυτές υπάρχει ένα κατακόρυφο μενού από facets και σε κάποιες περιπτώσεις εμφανίζεται και οριζόντιο το οποίο λειτουργεί συμπληρωματικά.



Εικόνα 27: Οριζόντιο facet layout στο Yelp.com

4.3.2 Η default κατάσταση των facets και η εμφάνιση επιπλέον τιμών

Όσον αφορά στην default κατάσταση των facets έχουμε 2 επιλογές. Να είναι **κλειστά η ανοικτά**. Η πρώτη επιλογή εξοικονομεί χώρο στην αρχική οθόνη, ειδικά όταν το πλήθος των facets είναι μεγάλο. Το μειονέκτημα είναι ότι δεν επικοινωνεί το ίδιο καλά την λειτουργία των facets με τον χρήστη. Η δεύτερη επιλογή είναι να βρίσκονται ανοικτά επιτρέποντας τον χρήστη να έχει καλύτερη αντίληψη για την λειτουργικότητά τους αλλά δαπανά περισσότερο χώρο. Μια χρήσιμη λειτουργία με μόνιμα ανοικτά facets είναι τα "**smart dead ends**" δηλαδή η κατάλληλη υπόδειξη για τιμές που οδηγούν σε μηδενικά αποτελέσματα αναζήτησης όπως ανενεργές επιλογές. Φυσικά, υπάρχουν και ενδιάμεσες καταστάσεις στις οποίες κάποια από τα facets χαρακτηρίζονται σαν τα κυρίαρχα για τις επιλογές του χρήστη και είναι εξορισμού ανοικτά και κάποια άλλα χαρακτηρίζονται σα δευτερεύοντα και είναι κλειστά.

Η έλευση των γραφικών διεπαφών του χρήστη προσφέρει πολλές επιλογές για τα controls που χρησιμοποιούμε στα μενού. Ομοίως στα facet navigation menus έχουμε τη δυνατότητα να χρησιμοποιήσουμε διάφορα controls όπως hyperlinks, Checkboxes, range sliders κτλ. Τα hyperlinks είναι η πιο συνηθισμένη μορφή κουμπιών επιλογών για τα facets και σε συνδυασμό με τον πλήθος των αποτελεσμάτων με τα οποία ταυτίζονται είναι ένας απλός και αξιόπιστος τρόπος επικοινωνίας. Σε αυτή τη μορφή τα single selection facets εξαφανίζονται αφού έχουν επιλεχθεί. Τα hyperlinks δεν θεωρούνται κατάλληλα για multi selections όχι για κάποιο τεχνικό λόγο αλλά γιατί είναι συνυφασμένα με Single selections και επίσης γιατί υπάρχουν άλλα πιο κατάλληλα controls όπως είναι τα check boxes. Η κακή χρήση των controls μπορεί να οδηγήσει σε κακή εμπειρία του χρήστη, δυσανάλογη με τις προσδοκίες του. Τα range sliders, είναι μία ευέλικτη και εντυπωσιακή μέθοδος για επιλογές εύρους τιμών

όπως για παράδειγμα το εύρος τιμών μεταχειρισμένων αυτοκινήτων. Η χρήση τους είναι πλέον κατάλληλη όταν έχουμε ποσοτικές τιμές στην συγκεκριμένη διάσταση. Τα Input boxes είναι επίσης κατάλληλα για ποσοτικές διαστάσεις και μπορεί να είναι πιο εύχρηστα από τα sliders ειδικά σε περιπτώσεις που απαιτείται ακρίβεια στις τιμές. Άλλα controls μπορεί να προσφέρουν καλύτερη διεπαφή με τον χρήστη σε συγκεκριμένες περιπτώσεις, όπως είναι το color picker που επιλέγεται χρώμα αντί να καταγράφεται αλλά και αρκετά πιο σύνθετα controls που οπτικοποιούν τα δεδομένα όπως το διάγραμμα τιμών εισιτηρίων για ένα συγκεκριμένο χρονικό διάστημα από ένα πρακτορείο ταξιδιών. Σε τέτοια controls ο ρόλος του facets είναι όχι τόσο να βρεθεί άμεσα ένα αποτέλεσμα αλλά να ομαλοποιήσει τα βήματα που απαιτούνται προς αυτό.

Η βασική επιλογή ενός συστήματος αναζήτησης που χρησιμοποιεί facets είναι η απόκρυψη των τιμών που δεν επιστρέφουν αποτελέσματα. Όμως στην περίπτωση που ο χρήστης θέλει να δει την πλήρη λίστα τιμών αυτό μπορεί να είναι δυνατό με κάποιους τρόπους:

- Ένας τρόπος είναι να εμφανίζεται από την αρχή όλη η λίστα τιμών με τα facets. Είναι ο πιο απλός τρόπος όμως προϋποθέτει ότι η λίστα δεν θα έχει υπερβολικά μεγάλο μέγεθος.
- Ο δεύτερος τρόπος είναι η χρήση περιοχών που επεκτείνονται ώστε να εμφανίζουν τις τιμές κατά απαίτηση όπως για παράδειγμα με ένα κουμπί ”show more”.
- Η τρίτη επιλογή μας είναι οι επιπλέον τιμές να εμφανίζονται σε κάποια ξεχωριστή περιοχή που εμφανίζεται όταν το θελήσει ο χρήστης. Στην πιο απλή έκδοσή της η περιοχή αυτή υπερκαλύπτει τα άλλα περιεχόμενα της σελίδας αναζήτησης ενώ σε άλλες περιπτώσεις εμφανίζεται ένα modal διάλογος με τις πλήρης τιμές των facets.

4.3.3 Αλληλεπίδραση με τον χρήστη

Βασική επιδίωξη των facets είναι η αποφυγή κενών σελίδων αποτελεσμάτων αλλά και η δυναμική επαναδιατύπωση των ερωτημάτων. Όταν για παράδειγμα έχουμε λίγες τιμές στα αποτελέσματα, μπορούμε να συμπεριλάβουμε και άλλες τιμές στα facets διευρύνοντας την αναζήτησή μας. Το ζήτημα είναι εάν η μεταβολή στις επιλογές των facets έχει άμεση μεταβολή στα αποτελέσματα ή αυτά αλλάζουν σε δεύτερο χρόνο με κάποιο πλήκτρο εφαρμογής. Αυτή η τακτική είναι συνεπής με τη λογική των facets και έχει την αποδοχή των χρηστών. Σε άλλες περιπτώσεις θα ήταν προτιμότερο ο χρήστης να ολοκληρώσει μια επιλογή όπως ημερομηνίες ή τα δύο άκρα ενός range slider πριν γίνει οποιαδήποτε αλλαγή στα αποτελέσματα ώστε να έχει νόημα η αλλαγή αυτή. Υπάρχουν και άλλες υλοποιήσεις που ο χρήστης επιλέγει μαζικά τις τιμές όλων των facets και επιλέγει κάποιο κουμπί εκκίνησης πριν εφαρμοστούν οι αλλαγές αυτές. Σε αυτή την μορφή έχουμε μια επικοινωνία δύο σταδίων. Αυτό που επιλέγονται οι τιμές και αυτό που εμφανίζονται τα αποτελέσματα. Χρειάζεται

μεγάλη προσοχή διότι μεταξύ των σταδίων μπορεί να έχουμε ασυνέπεια στις τιμές και στα αποτελέσματα. Ο χρήστης πρέπει να είναι ενήμερος για το ποιες επιλογές έχουν ήδη εφαρμοστεί και ποιες όχι. Εάν οι τιμές των facets είναι συζευκτικές τότε το instant update model είναι προτιμότερο γιατί δεν επιτρέπει επιλογές που θα οδηγήσουν σε κενά αποτελέσματα.

Είναι σημαντικό κατά τη διάρκεια της αναζήτησης ο χρήστης να γνωρίζει σε ποιο σημείο βρίσκεται και ποιες είναι οι διαθέσιμες επιλογές του. Η συνηθισμένη κατάσταση στο web είναι τα **breadcrumbs** που επικοινωνούν την τρέχουσα κατάσταση. Όμως η λογική των facets είναι να μπορούμε να έχουμε άμεση επιλογή σε πολλές διαφορετικές διαστάσεις και όχι σε μια αυστηρή ιεραρχία. Μια λύση είναι τα inline breadcrumbs όπως αυτά που χρησιμοποιεί το Amazon. Για κάθε κατηγορία (διάσταση) εμφανίζεται το όνομά της και οι τρέχουσες επιλογές του χρήστη. Η λύση αυτή έχει το πρόβλημα του μεγάλου μεγέθους εάν οι επιλογές μας είναι πολλές και επίσης δεν είναι κατάλληλη για Multi-select facets. Μια άλλη λύση είναι οι επιλεγμένες μόνο τιμές των facets να εμφανίζονται συγκεντρωμένες σε ένα ξεχωριστό χώρο ο οποίος αναφέρεται ως breadbox. Το πλεονέκτημα της μεθόδου είναι ότι οι επιλογές είναι μόνιμα εμφανείς σε ένα αποκλειστικό σημείο. Το μειονέκτημα είναι ότι η σχέση τιμών και facets είναι λιγότερο εμφανής και καμιά φορά δεν διακρίνεται εύκολα. Για παράδειγμα, αναζητώντας μητρική για έναν Η/Υ έχουμε στο breadcrumb τη λέξη 2 Gbyte. Όμως σε τι αναφέρεται η λέξη; Στο μέγεθος της μνήμης RAM ή στο buffer της κάρτας γραφικών; Σε αυτές τις περιπτώσεις το breadcrumb μπορεί να αναφέρει όχι μόνο την τιμή αλλά και τη διάσταση στην οποία αναφέρεται.

Με τις σημερινές τεχνολογίες ιστού έχουμε μια πληθώρα από διαθέσιμα controls. Ωστόσο δεν είναι όλα κατάλληλα για την απεικόνιση ενός facet. Στην συνήθη περίπτωση που έχουμε facets που λειτουργούν διαζευτικά πλέον κατάλληλα είναι τα radio buttons ή dropdown lists που επιτρέπουν την επιλογή μόνο μιας τιμής. Αν θέλουμε να έχουμε τη δυνατότητα επιλογής πολλών τιμών από το ίδιο facet θα προτιμούσαμε ένα checkbox. Για την επιλογή εύρους τιμών κατάλληλα είναι τα slider controls που παρέχουν κάτω και άνω όριο τιμών. Για facets που ενσωματώνουν κάποια ιεραρχία θα μπορούσαμε να χρησιμοποιήσουμε explorer tree controls ή κάτι παρόμοιο. Η εξοικονόμηση χώρου στην οθόνη μπορεί να γίνει χρησιμοποιώντας panels που ανοίγουν/κλείνουν ή στην λογική του accordion control. Μια καλή πρακτική είναι να χρησιμοποιείται κάποιου είδος animation για τις μεταβολές στις τιμές των facet ώστε ο χρήστης να μην χάσει οπτικά κάποια αλλαγή. Επιπρόσθετα η εξαφάνιση κάποιων facet απλά με την επιλογή τους δίνει στον χρήστη την αίσθηση ανασφάλειας για την σωστή ή όχι επιλογή τιμών.

5

Πλατφόρμες και Εργαλεία προγραμματισμού για information retrieval

5.1 Σύγχρονα Εργαλεία information retrieval

Σήμερα ένα πλήθος από εργαλεία που υλοποιούν information retrieval είναι διαθέσιμα. Αρκετά από αυτά είναι κλειστού κώδικα ή προσφέρονται σαν υπηρεσία από τις εταιρείες που τα αναπτύσσουν. Στη δική μας εργασία ήταν απαραίτητο να εξετάσουμε τις επιλογές μας στα εργαλεία ανοικτού κώδικα ώστε να μπορέσουμε να αναπτύξουμε το δικό μας προϊόν. Τα περισσότερα από τα εργαλεία που βρήκαμε είναι projects ή ports χτισμένα πάνω στο Apache Lucene. Επίσης κάποια εργαλεία χτίζουν την δική τους βάση δεδομένων ή χρησιμοποιούν κάποια έτοιμη λύση.

5.1.1 Apache Lucene

Το Apache Lucene [7] [8] είναι το σημαντικότερο και πιο διαδεδομένο εργαλείο information retrieval. Υλοποιεί μία βιβλιοθήκη με όλα τα απαραίτητα χαρακτηριστικά και είναι εργαλείο ανοικτού κώδικα γραμμένο σε Java. Τα χαρακτηριστικά του είναι:

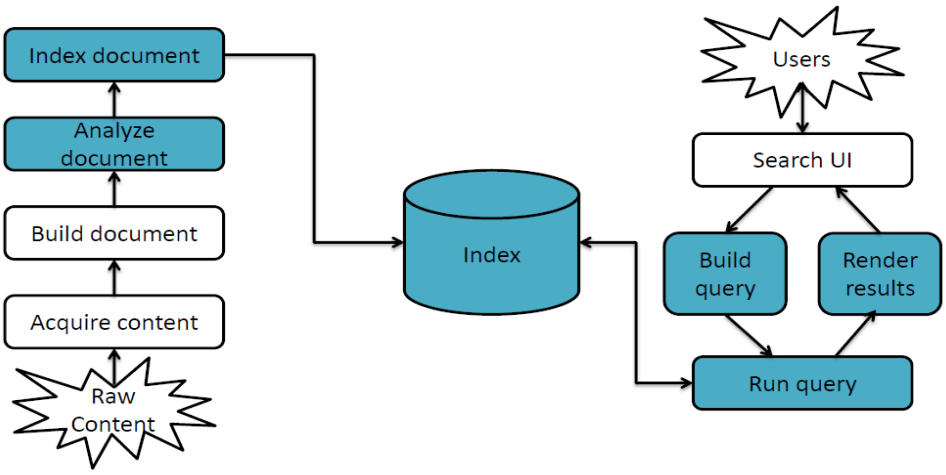
[7] https://lucene.apache.org/core/

[8] https://en.wikipedia.org/wiki/Lucene

- Scalable, High-Performance Indexing
- ranked searching
- fielded searching
- pluggable ranking models
- allows simultaneous update and searching
- flexible faceting, highlighting, joins and result grouping

Πρόκειται για μία βιβλιοθήκη και όχι για ένα έτοιμο προϊόν για χρήση. Πρέπει να ενσωματωθεί σε κάποια εφαρμογή που θα το καλεί υλοποιώντας ένα επίπεδο Data Import χρησιμοποιώντας XML και επίσης θα υλοποιεί και το user interface. Μεγάλες εφαρμογές στηρίζονται σε αυτό όπως τα Twitter, LinkedIn, CNET, AOL, Eclipse (IDE), AOL, Disney κτλ.

Σε όλες τις διαφορετικές εκδόσεις του Lucene η διαδικασία αξιοποίησής του περιλαμβάνει κάποια βασικά στάδια. Τα αρχικά έγγραφα αναλύονται και χτίζεται ο αρχικός index. Ο τελικός χρήσης μέσω του UI δίνει ένα ερώτημα το οποίο μετατρέπεται στο τελικό ερώτημα προς τον index ενώ τα παραγόμενα αποτελέσματα δίνονται στον χρήστη.



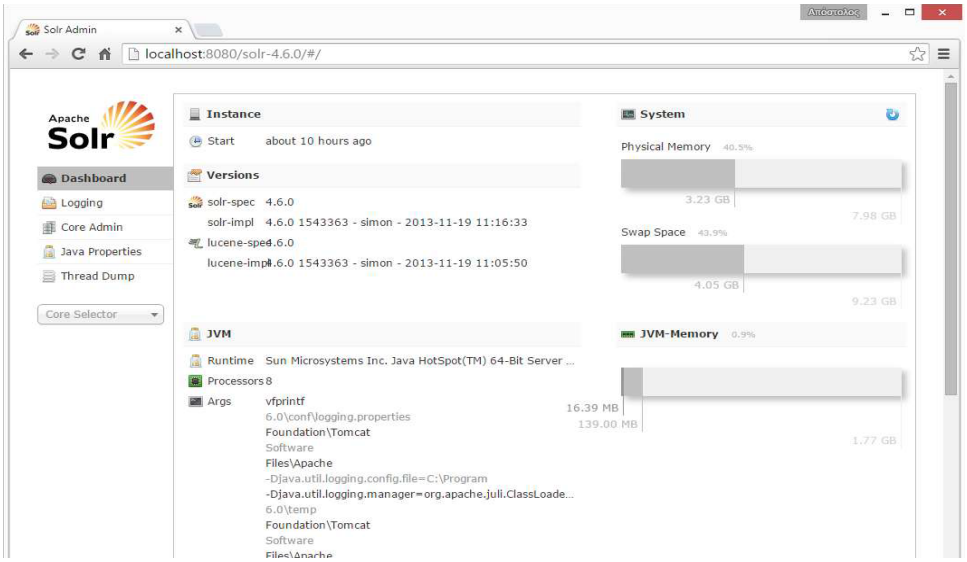
Εικόνα 28: Διαδικασία χρήσης του Lucene

Το Lucene ενσωματώνει πολύ δυναμικά χαρακτηριστικά. Φυσικά ένα από αυτά είναι το faceted search. Υπάρχει μάλιστα η δυνατότητα τα facets να χτίζονται όχι από το σύνολο των εγγράφων της συλλογής μας αλλά από ένα δείγμα (sample) αυτών. Εφόσον τα facets φτιαχτούν, μετά χρησιμοποιούνται τα ίδια για όλη τη συλλογή.

Άλλο χαρακτηριστικό αφορά στην ταυτόχρονη αναζήτηση και χτίσιμο του index. Αυτό είναι δυνατό γιατί το lucene είναι multiprocess και υποστηρίζει ταυτόχρονες διαδικασίες και στηρίζεται τα point of time (POT) δηλαδή όταν ο χρήστης ανοίγει τον Index παίρνει ένα view του συστήματος όπως ήταν εκείνη τη στιγμή. Αν κατά τη διάρκεια της χρήσης ο Index αλλάξει, το συγκεκριμένο session δεν θα το αντιληφθεί.

5.1.2 Solr

Το Solr [9](προφέρεται "ΣΟΛΑΡ") είναι μία έτοιμη πλατφόρμα information retrieval χτισμένη πάνω στο Lucene. Είναι open-source, γραμμένη σε Java, με πολύ δυναμικά χαρακτηριστικά όπως free-text search, hit highlighting, faceted search, real-time indexing, dynamic clustering, database integration, NoSQL και υποστήριξη για rich documents (Word, PDF). Είναι ένας έτοιμος Standalone server που αρχικά αναπτύχθηκε από το CNET για εσωτερική λειτουργεία αλλά από το 2010 είναι κοινό προϊόν με το Lucene. Βασίζεται σε ανοικτά πρότυπα όπως XML, HTTP, Json και διαθέτει το δικό του Web Interface.



Εικόνα 29: Το web interface του Apache Solr

5.1.3 Lucene.Net

Το Lucene.Net [10]είναι ένα port για το Apache Lucene γραμμένο σε C# το οποίο αξιοποιεί τις βιβλιοθήκες του .Net και έχει σαν βασικό στόχο την διατήρηση της πλήρης λειτουργικότητας του Lucene και την μεγιστοποίηση της ευχρηστία σε συνδυασμό με το .Net runtime. Υπήρξε σημαντική εξέλιξη στο project Lucene διότι έδωσε τη δυνατότητα ενσωμάτωσης των χαρακτηριστικών του σε εφαρμογές που αναπτύσσονται με τα εργαλεία της Microsoft όπως το Visual Studio [11].

Διάφορα project συνεισφέρουν στο Lucene.Net επεκτείνοντας τα χαρακτηριστικά του ή απλοποιώντας τη χρήση του. Παρακάτω αναφέρουμε κάποια από αυτά.

[9] http://lucene.apache.org/solr/

[10] https://lucenenet.apache.org/

[11] Ένα πολύ καλό βοήθημα είναι το βιβλίο "Lucene in action" (εκδόσεις Manning)

Ίσως το πιο σημαντικό από αυτά είναι το **Bobo Search Engine**. Είναι ένα software wrapper που αναπτύχθηκε αρχικά σε Java και αργότερα σε C# αξιοποιώντας χαρακτηριστικά όπως το faceted search. Η δική μας demo εφαρμογή που περιγράφουμε στο κεφάλαιο 6, χρησιμοποιεί το BoBo Search. Εκεί θα κάνουμε ιδιαίτερη αναφορά στα προγραμματιστικά αντικείμενά του.

Άλλο σημαντικό project είναι το **NHibernate-Search** το οποίο διευκολύνει το mapping δεδομένων από βάσεις δεδομένων σε αντικείμενα.

Τέλος τα **SimpleFacetedSearch** και το **MultiFacetedSearch** διευκολύνουν τη χρήση των αντικειμένων του Lucene από τους developers ειδικά στη χρήση του faceted search. Όλα τα παραπάνω project ειναι διαθέσιμα μέσω GitHub και επίσης σαν πακέτα εγκαταστασης NuGet.

5.1.4 Elastic Search

Το ElasticSearch [12] είναι μία ακόμα βιβλιοθήκη της οικογένειας του Lucene η οποία ανταγωνίζεται το Solr. Παρέχει ένα κατανεμημένο περιβάλλον με web interface και Json δεδομένα.Φτιάχτηκε με σκοπό να παρέχει εύκολη εγκατάσταση, χωρίς την απαίτηση για Schema δεδομένων σε ένα scalable περιβάλλον αναζήτησης και σήμερα είναι η δεύτερη πιο διαδεδομένη λύση μετά το Solr και χρησιμοποιείται από αρκετές μεγάλες εγκαταστάσεις όπως το Facebook, ING, NetFlix, The New York Times, Microsoft, Mozzila, Ebay, Adobe κτλ.

5.1.5 Sphinx

Ένα άλλο framework είναι το Sphinx [13]. Ιδιαίτερο χαρακτηριστικό του είναι η στενή διασύνδεσή του με σχεσιακές βάσεις δεδομένων όπως η MySQL, ενώ φαίνεται να είναι πιο γρήγορο από το Solr σε κάποιες διεργασίες όπως το χτίσιμο του index. Οι δυνατότητές του για faceted search είναι μεγάλες αλλά μόνο αφού έχει γίνει χειροκίνητα προεργασία. Γραμμένο σε C++ χρησιμοποιείται επίσης από μεγάλες εγκαταστάσεις όπως η Groupon.

5.1.6 Lemur Project

Από τη συνεργασία των University of Massachusetts Amherst, Language Technologies Institute, Carnegie Mellon University προέκυψε το Lemur Project [14] ένα προϊόν που περιλαμβάνει τα Lemur Toolkit και Indri Search Engine. Το Lemur Toolkit είναι ένα open-

[12] https://www.elastic.co/

[13] http://www.sphinx-doc.org/en/stable/

[14] http://www.lemurproject.org/

source software framework για information retrieval και ανάπτυξη εργαλείων ανάλυσης κειμένων, toolbars για browsers κτλ. Το Indri επιτρέπει την δημιουργία Index για δεδομένα μέσω απλών εντολών command line και την συνεργασία με εφαρμογές. Είναι κατάλληλο για μεγάλο όγκο δεδομένων και διαχειρίζεται αρχεία XML.

5.1.7 Terrier

Πρόκειται για ένα εργαλείο ανοικτού λογισμικού που αναπτύχθηκε στο πανεπιστήμιο της Γλασκώβης [15]. Γραμμένο σε Java είναι σχετικά εύκολο στη χρήση του με προσανατολισμό κυρίως στην ερευνητική κοινότητα. Βασικά χαρακτηριστικά του είναι τα εξής:

- Αποτελεσματικό
- Υλοποίηση πολλών μοντέλων αναζήτησης
- Επεκτάσιμο: Νέες τεχνικές αναζήτησης, νέα χαρακτηριστικά ranking
- Διαδραστικό μέσω desktop, web app

Τέλος, υπάρχει ανεπτυγμένη μεγάλη γνωσιακή βάση.

5.1.8 Drupal

Το Drupal [16] ένα Open source λογισμικό, είναι η πρώτη και η μόνη ως τώρα πλατφόρμα CMS που ενσωματώνει εσωτερικά δυνατότητα faceted search. Η δυνατότητα περιλαμβάνεται σε ένα σετ από Modules το καθένα από τα οποία υλοποιεί ένα μέρος του συστήματος. Για παράδειγμα το "Faceted Search" υλοποιεί την αναζήτηση, το ""Faceted Search UI" υλοποιεί την διεπαφή με τον χρήστη, το "Author Facet" ορίζει και διαχειρίζεται τα facets κτλ. Η ανάπτυξη του συστήματος γίνεται από το Drupal και όχι από τρίτους κατασκευαστές. Ενδιαφέρον έχει η προειδοποίηση ότι το σύστημα απασχολεί ιδιαίτερα την βάση δεδομένων και έχει επιπτώσεις στην απόδοση ειδικά σε περιπτώσεις μειωμένων πόρων υλικού ή δικτύου. Αρκετές εγκαταστάσεις που βασίζονται στο drupal χρησιμοποιούν facets.

5.1.9 Joomla και Wordpress

Τα πολύ διαδεδομένα εργαλεία CMS Joomla και Wordpress δεν διαθέτουν χαρακτηριστικά faceted search, ωστόσο μπορεί κανείς να βρεις κατάλληλα extensions και plugins. Ένα από αυτά είναι το wordpress plugin WPSOLR που στηρίζεται στην ενσωμάτωση του Solr σε μία εγκατάσταση wordpress. Άλλο ένα είναι το InstandSearch+ είναι ένα cloud based plugin που ενσωματώνει δυναμικά χαρακτηριστικά αναζήτησης στο wordpress με εγκατεστημένο το

[15] http://terrier.org/

[16] https://www.drupal.org/

woocomerce,plugin που χτίζει ένα e-commerce πάνω στο wordpress. Στο Joomla, modules όπως το JA K2 Filter and Search Module και το XTDir υλοποιούν faceted search.

5.2 Search services

Εκτός από τα παραπάνω εργαλεία με τα οποία μπορούμε να χτίσουμε δικές μας εφαρμογές information retrieval με faceted search υπάρχουν και έτοιμα λογισμικά που προσφέρονται συνήθως στο cloud σαν υπηρεσίες (software as a service). Αναφέρουμε μερικά από τα πιο διαδεδομένα:

5.2.1 Azure Search Service

Η μεγάλη εξάπλωση του cloud computing όπως ήταν αναμενόμενο έφερε νέα εργαλεία και στο πεδίου του information retrieval. Μία σχετικά πρόσφατη εξέλιξη είναι το Microsoft Azure Search Service [17]. Πρόκειται για ένα λογισμικό που προσφέρεται σαν υπηρεσία (Software as a Service) από το Azure που είναι το γενικό όνομα για όλες τις υπηρεσίες Cloud της Microsoft. Τα βασικά χαρακτηριστικά του είναι :

- Αυτόματη δημιουργία και χρήση των index
- Δυνατότητα επέκτασης (scale up) και μείωσης (scale down) κατά απίτηση του χρήστη
- Επεξεργασία φυσικής γλώσσας (Natural Language Processing)
- Αυτόματη φόρτωση και ανανέωση των index
- Ενσωμάτωση Geo-spatial δεδομένων στην εφαρμογή μας
- Ενσωμάτωση των χαρακτηριστικών του σε οποιαδήποτε εφαρμογή web και mobile που μπορεί να καλεί το Azure
- Όπως κάθε S.A.A.S. μειώνει την πολυπλοκότητα του συστήματος καθώς οι λεπτομέρειες υλοποίησης και διαχείρισης παραμένουν ευθύνη της υπηρεσίας
- Η βιβλιοθήκη είναι διαθέσιμη μέσω του Nuget [18]

5.2.2 Amazon CloudSearch

Μία από τις cloud υπηρεσίες του Amazon (Amazon Cloud Services) είναι το CloudSearch [19]. Πρόκειται για μία ακόμα υπηρεσία τύπου Software as a Service που προσφέρει υπηρεσίες

[17] Δείτε περισσότερα στο https://azure.microsoft.com/en-us/services/search/

[18] https://www.nuget.org/profiles/azure-sdk

[19] https://aws.amazon.com/cloudsearch/

search για τα δεδομένα των πελατών του. Το Amazon CloudSearch μπορεί να διαχειριστεί δεδομένα όπως αρχεία, ιστοσελίδες, databases και προσφέρει εξελιγμένα χαρακτηριστικά όπως free text Search, επεξεργασία φυσικής γλώσσας, faceted search, boolean search (filters), ranging, highlights και autocomplete. Οι ενέργειες του χτισίματος, διαχείρισης του Index και της αναζήτησης αναλαμβάνονται από το σύστημα και όπως σε άλλες αντίστοιχες υπηρεσίες το scale up είναι εύκολη υπόθεση.

5.2.3 SearchBlox

Η εταιρεία Searchblox [20] ειδικεύεται στην παροχή υπηρεσιών enterprise search. Προσφέρει εργαλεία με τα οποία οι πελάτες της μπορούν να χτίσουν ένα search service για τα δεδομένα της επιχείρησής τους που μπορεί να είναι websites, databases, file system κτλ. Η υπηρεσία είναι cloud based και ενσωματώνεται εύκολα στις εφαρμογές των πελατών. Υποστηρίζεται website crawling, indexing, faceted search, e-commerce search, φιλτράρισμα σε προϊόντα ή κατηγορίες και intranet search. Στην Ελλάδα ένας πελάτης της SearchBlox ειναι η Cosmote.

5.3 Τα κύρια προγραμματιστικά αντικείμενα του Lucene.Net

Όπως αναφέραμε ήδη η ανάπτυξη της εφαρμογής μας βασίζεται στο Lucene.Net, ίσως τη μοναδική λύση λογισμικού βιβλιοθήκης αναζήτησης για εργασία σε περιβάλλον .Net. Είναι πολύ πιθανό η ανάπτυξη μίας εφαρμογής να χρησιμοποιεί επιπλέον βιβλιοθήκες που συνεισφέρουν στον project ωστόσο υπάρχουν κάποια βασικά αντικείμενα που πρέπει να χρησιμοποιήσει μία εφαρμογή βασισμένη στο Lucene τα οποία είναι τα ίδια στην Java έκδοση όσο και στο .Net. Σε αυτό το στάδιο θα περιγράψουμε μερικά από αυτά.

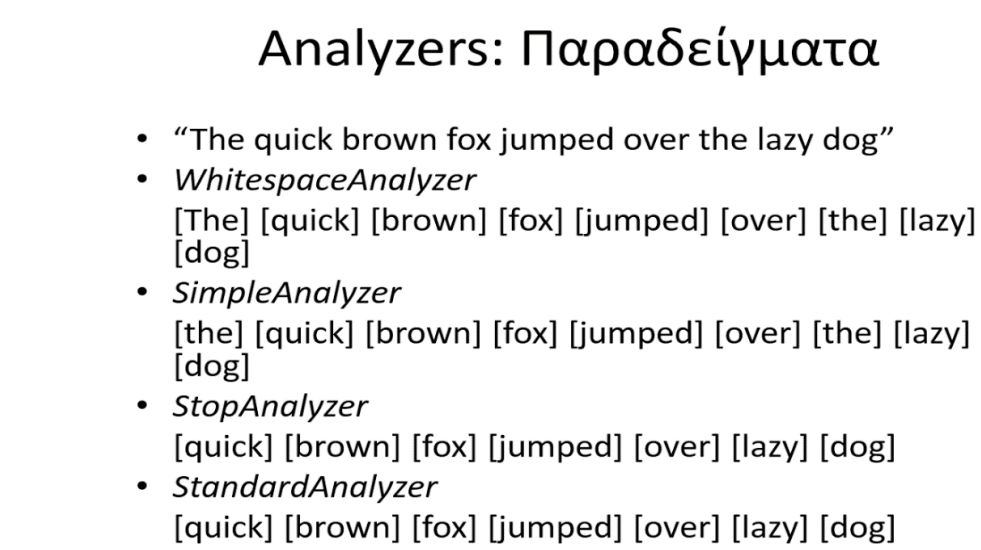
Directory: Είναι ο χώρος στον οποίο αποθηκεύεται ο index. Μπορεί να είναι ένας φάκελος στον σύστημα αρχείων (FSDirectory) ή ένας χώρος στη μνήμη (RAMDirectory). Ο Index είναι συμβατός με όλες τις εκδόσεις του Lucene ώστε να χρησιμοποιείται ταυρόχρονα από εφαρμογές .Net ή Java. Σε κάποιες περιπτώσεις προτιμάται η χρήση RamDirectory λόγω της ταχύτητας που προσφέρει η λύση, ωστόσο υπάρχει περιορισμός λόγω χώρου και δεν είναι κατάλληλο για εφαρμογές που εξυπηρετούν πολλά δεδομένα. Σε αυτές τις περιπτώσεις άλλες τεχνικές όπως οι distributed indexes και το clustering μπορεί να δώσουν λύσει.

IndexWriter: Είναι το αντικείμενο που διαχειρίζεται τον index και τα έγγραφα που προσθέτουμε σε αυτόν. Τον δημιουργεί και προσθέτει σε αυτόν έγγραφα προς indexing.

[20] http://www.searchblox.com/

Analyzer: Η αρχική δουλειά του analyzer είναι η εξαγωγή χρήσιμων όρων για την αναζήτηση. Διάφορες υλοποιήσεις περιλαμβάνονται στην βασική έκδοση το Lucene. Ο StandardAnalyzer κάνει Tokenizing χρησιμοποιώντας λεξικά εργαλεία και αφαιρώντας τις μη χρήσιμες λέξεις (stopwords). Ο SnowballAnalyzer κάνει την αντίστοιχη δουλειά σε δύο στάδια. Στο πρώτο κάνει Tokenize και στο δεύτερο (Stemming) αποδίδει τις λέξεις στην ρίζα τους. Για παράδειγμα οι λέξεις developer, develop, development καταλήγουν στην ρίζα τους develop. ο Whitespace Analyzer κάνει απλά διαχωρισμό των λέξεων. Ο Simple analyzer κάνει διαχωρισμό των λέξεων, μετατρέπει όλους τους χαρακτήρες σε πεζούς και αφαιρεί τα απλά γράμματα.

Πρέπει να σημειώσουμε ότι ο οι ενέργειες του Analyzer χρειάζονται τόσο στο χτίσιμο του Index όσο και στο query parsing.



Εικόνα 30: Παραδείγματα λειτουργίας των lucene analyzers

Document: Είναι η μονάδα αναζήτησης η οποία αναλύεται και καταγράφεται στον index. Τα documents μπορεί να περιλαμβάνουν πεδία όπως στις βάσεις δεδομένων, να είναι δομημένα, ημιδομημένα η αδόμητα ενώ η τεχνικές για την ανάλυσή τους μπορεί να διαφοροποιούνται ακόμα και από πεδίο σε πεδίο ανάλογα με τις απαιτήσεις. Κατά την δημιουργία του Index έχουμε τη δυνατότητα να αναλύσουμε τα πεδία κατάλληλα με τις ιδιότητες:

- Field.Store:
    - True / False. Επιτρέπει την αποθήκευση των περιεχομένων του πεδίου εντός του index. Σε διαφορετική περίπτωση αποθηκεύονται μόνο οι δυαδικές τιμές που εξετάζονται κατά την αναζήτηση. Αν χρειαστεί η ανάκτηση του πεδίου θα γίνει από τα πραγματικά έγγραφα.
- Field.Index:
    - Analyzed: Γίνεται ανάλυση του πεδίου και σπάσιμο αυτού σε tokens

- NOT_Analyzed: Δεν γίνεται ανάλυση του πεδίου και καταγράφεται στον index όπως είναι.
- NO: Το πεδίο δεν συμπεριλαμβάνεται στον index

Searcher and Index Searcher: Είναι το αντικείμενο που αναλαμβάνει την αναζήτηση των όρων του ερωτήματος εντός του index.

QueryParser: Είναι υπεύθυνος για την ανάλυση του ερωτήματος που συνήθως διατυπώνεται σε ελεύθερο κείμενο ώστε να προκύψουν τα αντικείμενα της αναζήτησης.

Το αποτέλεσμα μίας αναζήτησης στο lucene προσπελαύνεται μέσω των αντικείμενων TopDocs και ScoreDocs. Το TopDocs επιλέγει τα έγγραφα που ταιριάζουν στην αναζήτησή μας. Το ScoreDocs υλοποιεί το Ranking

Η διαδικασία που ακολουθείται για το χτίσιμο μίας εφαρμογής περιλαμβάνει τα εξής βήματα:

- Αρχικοποίηση του directory και του indexwriter
- Προσθήκη εγγράφων στον index
- Χτίσιμο του ερωτήματος του χρήστη
- Αναζήτηση στον index από τον indexsearcher
- Επαναληπτική επισκόπηση των αποτελεσμάτων
- Αποδέσμευση όλων των αντικειμένων

```
//Initialize directory, analyzer, indexwriter
Directory directory = FSDirectory.GetDirectory("LuceneIndex");
Analyzer analyzer = new StandardAnalyzer();
IndexWriter writer = new IndexWriter(directory, analyzer);
IndexWriter writer = new IndexWriter("LuceneIndex", analyzer);

//add documents
Document doc = new Document();
doc.Add(new Field("id", i.ToString(), Field.Store.YES, Field.Index.NO));
doc.Add(new Field("postBody", text, Field.Store.YES, Field.Index.TOKENIZED));
writer.AddDocument(doc);
writer.Optimize();

//Close the writer
writer.Flush();
writer.Close();

//Parse the query
QueryParser parser = new QueryParser("postBody", analyzer);
Query query = parser.Parse("text");

//Setup searcher and Start searching
IndexSearcher searcher = new IndexSearcher(directory);
Hits hits = searcher.Search(query);int results = hits.Length();
Console.WriteLine("Found {0} results", results);

//Loop through results
for (int i = 0; i < results; i++)
{
    Document doc = hits.Doc(i);
    float score = hits.Score(i);
    Console.WriteLine("Result num {0}, score {1}", i+1,score);
    Console.WriteLine("ID: {0}", doc.Get("id"));
    Console.WriteLine("Text found: {0}" + Environment.NewLine, doc.Get("postBody"));
```

Εικόνα 31: demo code για χρήση του Lucene.Net

5.4 Η βιβλιοθήκη Bobo Browse

Όπως αναφέραμε νωρίτερα η βιβλιοθήκη BoBo Browse είναι ένα Open source project που αναπτύχθηκε σαν ένα extension του Lucene με σκοπό την απλοποίηση της χρήσης του Lucene, την βελτίωση κάποιων χαρακτηριστικών και την επέκταση των δυνατοτήτων του. Μπορεί να χρησιμοποιήσει οποιονδήποτε Index που φτιάχτηκε για το Lucene με την προσθήκη κάποιων πληροφοριών για το faceted search που είναι τα faceted πεδία και ο τύπος των facets για το κάθε πεδίο. Στο Bobo Browse η δήλωση των facets γίνεται με τη μορφή ενός **Spring File**[21] δηλαδή ενός configuration file σε μορφή XML. Εναλλακτικά ο developer μπορεί να ορίσει προγραμματιστικά όλες τις απαραίτητες πληροφορίες. Η αρχιτεκτονική αυτή επιλέχθηκε ώστε να μπορεί ο κάθε Index να συμπεριλάβει facets χωρίς να χρειαστεί να χτιστεί από την αρχή. Κάθε πεδίο ορίζεται από έναν **facet Handler**, αντικείμενο που παρέχει προεπισκόπηση των δεδομένων του (όπως Facet HitCount) και το Bobo Browse χτρησιμοποιεί τους facet handlers για count, sort και field retrieval.

Η απλοποιημένη διαδικασία από το άνοιγμα ενός Index μέχρι το διάβασμα επιστρεφόμενων τιμών και facets περιλαμβάνει τα εξής στάδια:

- Διαβάζουμε έναν index αρχικά ανοίγοντάς τον όπως θα κάναμε με το lucene και μετά χτίζοντας ένα αντικείμενο **BoBoIndexReader**.
- Δημιουργούμε ένα αντικείμενο BrowseRequest το οποίο δένει τον index με το query του χρήστη σε συνδυασμό με την λίστα των facet hanlders που θα παρέχουμε.
- Το αντικείμενο **BrowseRequest** χρειάζεται μια βασική παράμετρο που είναι το query, αντικείμενο του lucene.
- Το Bobobrowse υπολογίζει τις τιμές των facets και τις διαβάζουμε σαν μία λίστα από **facetspecs** που επιστρέφει το BrowseRequest. Σε αυτά μπορούμε να θέσουμε τις προτιμήσεις μας όπως, το μέγιστο πλήθος facets για κάθε πεδίο, την διάταξη (sort) των facets που μπορεί να είναι αλφαβητική ή φθίνουσα κατά hitcount και το ελάχιστο όριο hitCount που πρέπει να ικανοποιεί ένα facet για να συμπεριληφθεί στη λίστα.
- Τώρα μπορούμε να διαβάσουμε τιμές δημιουργώντας ένα αντικείμενο **BoboBrowser** που κληρονομεί από την κλάση indexsearcher του Lucene. Η μέθοδος browse του BoboBrowser επιστρέφει το αντικείμενο **result** το οποίο ανοίγουμε και διαβάζουμε με μια επαναληπτική διαδικασία.
- Το result περιλαμβάνει τη μέθοδο GetHIts που επιστρέφει την λίστα από **BrowseHit**, αντίστοιχο αντικείμενο με το scoredocs του Lucene το οποίο όμως εκτός από τα

[21] http://spring.io/

έγγραφα που κάνουν match περιλαμβάνει πληροφορίες και για τα facets στη μορφή ζευγαριών (name, value) **Facetaccesible**.

Σε δοκιμές που έγιναν από την ομάδα ανάπτυξης αποδείχτηκε ότι η προσπέλαση του Index με τους facetHanlders του bobobrowse αντί της κλασικής μεθόδου του lucene δηλαδή του αντικείμενου document αυξάνει την απόδοση του συστήματος κατά 30 φορές!

```
IndexReader reader = getLuceneIndexReader();
BoboIndexReader boboReader = BoboIndexReader.getInstance(reader);
BrowseRequest br = new BrowseRequest();
br.setCount(10);
br.setOffset(0);
QueryParser queryParser;
Query query = queryParser.parse("position_description:(software OR engineer)");
br.setQuery(query);
FacetSpec companyNameSpec = new FacetSpec();
companyNameSpec.setOrderBy(FacetSortSpec.OrderHitsDesc);
companyNameSpec.setMaxCount(20);
br.setFacetSpec("company_name", companyNameSpec);
Browseable browser = new BoboBrowser(boboReader);
BrowseResult result = browser.browse(br);
int totalHits = result.getNumHits();
BrowseHit[] hits = result.getHits();
```

Εικόνα 32: Κώδικας ανοίγματος index και υποβολής ερωτήματος με τη βιβλιοθήκη BoboBrowse

5.5 Αναζήτηση σε πραγματικό χρόνο

Ένα σύστημα αναζήτησης πραγματικού χρόνου είναι αυτό που σε κάθε αναζήτηση που γίνεται εντοπίζει όλες τις αλλαγές που έχουν γίνει ως εκείνη τη χρονική στιγμή στη συλλογή εγγράφων. Αν για παράδειγμα προστεθεί ένα έγγραφο το περιεχόμενό του θα πρέπει να έχει αναλυθεί και καταγραφεί στον index του συστήματος αναζήτησης ώστε όλες οι μετέπειτα αναζητήσεις να περιλαμβάνουν την αλλαγή αυτή. Αν και το θέμα αυτό του real time search αφορά στην διαδικασία της αναζήτησης πριν ακόμα χτιστούν τα facets, είναι εξίσου σημαντικό για το faceted search γιατί δίνει τη δυνατότητα στα facets να υπολογίζονται με τα δεδομένα που υπάρχουν ακριβώς εκείνη τη στιγμή στην βάση μας και όχι με αυτά που υπήρχαν όταν φτιάχτηκε ο index.

Το Lucene δίνει τη δυνατότητα να χτίζουμε τον index σταδιακά (incremental) και όχι κάθε φορά από την αρχή. Ωστόσο, η απλοϊκή λύση του να χτίζουμε έστω σταδιακά τον Index μετά από κάθε αλλαγή παρουσιάζει κάποια προβλήματα όπως:

- Πολύ υψηλό φόρτο και καθυστέρηση λόγω εγγραφής του index στον δίσκο.
- Μεγάλος κατακερματισμός του αρχείου του index λόγω συχνών αλλαγών.
- Ίσως αχρείαστος όγκος indexing λόγω συχνών και άσκοπων αλλαγών των δεδομένων από κάποιον χρήστη.
- Η ανάγκη για συχνή φόρτωση του IndexReader καθιστά το σύστημα πολύ αργό.

Η εναλλακτική λύση θα ήταν ο Index να διατηρείται στη μνήμη του συστήματος μέσω ενός RAMDirectory αντικειμένου. Όμως η λύση αυτή δεν αντιμετωπίζει όλα τα προβλήματα ενώ προσθέτει και το θέμα της συντήρησής του σε κατανεμημένα συστήματα.

5.5.1 Το Near Real Time Search του Lucene

Το Lucene παρέχει τη δυνατότητα Near Real Time Search (NRT) μέσω των νέων μεθόδων getReader και SetIndexReaderWarmer του αντικειμένου IndexWriter. Η ιδέα είναι η μέθοδος getReader να επιστρέφει τον Indexreader με τα έγγραφα που έχει καταγράψει αν και δεν έχει ακόμα καλέσει τη μέθοδο IndexWriter.Commit. Η δεύτερη μέθοδος setIndexREaderWarmer είναι να ενσωματώνει στον IndexReader τα νέα τμήματα που αφορούν στα έγγραφα για τα οποία δεν έγινε ακόμα Commit (warm task).

5.5.2 Το σύστημα Zoie

Η ομάδα ανάπτυξης της βιβλιοθήκης BoboBrowse ανέπτυξε για το LinkedIn ένα σύστημα αναζήτησης πραγματικού χρόνου με το οποίο κάθε προφίλ χρήστη που προστίθεται είναι άμεσα διαθέσιμο για αναζήτηση. Το σύστημα ονομάστηκε Zoie[22], είναι λογισμικό ανοικτού κώδικα και λειτουργεί σαν ενδιάμεσος μεταξύ των αιτημάτων των χρηστών και του Lucene με σκοπό τον συγχρονισμό των νέων εγγράφων που προστίθενται στην συλλογή με τον index του συστήματος. Το Zoie βασίζει τη λειτουργία του στον συνδυασμό δύο RAMDirectories και ενός FileDirectory. Το FileDirectory είναι ο κεντρικός index του συστήματος που εξυπηρετεί τα ερωτήματα των χρηστών. Ο πρώτος RAMDirectory καταγράφει τις νέες αλλαγές στα δεδομένα μας και ανά διαστήματα (που ρυθμίζονται προγραμματιστικά) μεταφέρει μαζικά τα νέα δεδομένα στον κεντρικό Index. Όταν γίνεται αυτό, την καταγραφή των νέων δεδομένων αναλαμβάνει ο δεύτερος RAMDirectory. Εκείνη τη στιγμή τα ερωτήματα των χρηστών εξυπηρετούνται από όλους τους Index. Όταν τελειώσει η μεταφορά του πρώτου RAMDirectory στον FileDirectory, αυτός αδειάζει και αλλάζει ρόλο με τον δεύτερο.

Το σύστημα αποδείχτηκε πρακτικά πολύ αποτελεσματικό με τη χρήση του στο LinkedIn.

[22] http://sna-projects.com/zoie)

5.6 Προγραμματισμός με Visual Studio

Αν και η παρουσίαση ενός γενικού περιβάλλοντος προγραμματισμού ξεφεύγει από τους σκοπούς του κειμένου, ωστόσο θα κάνουμε μια σύντομη αναφορά στο Visual Studio με το οποίο αναπτύξαμε την εφαρμογή μας. Πρόκειται για ένα ιδιαίτερα εξελιγμένο περιβάλλον προγραμματισμού εφαρμογών κάθε είδους όπως windows εφαρμογές, web εφαρμογές, εφαρμογές για κινητές συσκευές, βιβλιοθήκες αντικειμένων κάθε είδους, εφαρμογές που συνεργάζονται με βάσεις δεδομένων. Επίσης εκτός από εφαρμογές για Microsoft Windows δίνεται η δυνατότητα ανάπτυξης εφαρμογών για Android, IOS. Υποστηρίζεται η ομαδική εργασία, η εφαρμογή Test Units και υποστηρίζονται πολλές γλώσσες όπως οι C#, C++, Visual Basic, Python κτλ. Ένα πρόσφατο χαρακτηριστικό είναι η πλήρης υποστήριξη για ανάπτυξη εφαρμογών που εκτελούνται στο cloud.

Με το Visual Studio έχουμε τη δυνατότητα ανάπτυξης εφαρμογών για κάθε πλατφόρμα ωστόσο ο βασικός προσανατολισμός παραμένει η αξιοποίηση της βιβλιοθήκης .Net[23]. Πρόκειται για μία πολύ ισχυρή και πλούσια βιβλιοθήκη που δίνει τη δυνατότητα ανάπτυξης σύγχρονων εφαρμογών με την χρήση των αντικειμένων που παρέχει. Το .Net αποτελείται από ένα περιβάλλον εκτέλεσης εφαρμογών το οποίο συνεργάζεται στενά με το λειτουργικό σύστημα των windows και παρέχει υπηρεσίες διαχείρισης της εκτέλεσης των εφαρμογών έτσι ώστε οι εφαρμογές που χρησιμοποιούν το .Net[24] να είναι απαλλαγμένες από τις λεπτομέρειες διαχείρισης και εκτέλεσης του κώδικα. Για παράδειγμα μια εφαρμογή δεν χρειάζεται να καταστρέφει τα αντικείμενα που έχει αποδεσμεύσει γιατί θα το αναλάβει το .Net framework. Επιπλέον, η εκτεταμένη βιβλιοθήκη δίνει στον προγραμματιστή τη δυνατότητα να εκμεταλλευτεί τον έτοιμο, αξιόπιστο και ανθεκτικό σε σφάλματα κώδικα που καλύπτει πολλές σημαντικές περιοχές της ανάπτυξης εφαρμογών. Ένα παράδειγμα αποτελεί η ασφάλεια που υλοποιείται μέσω του .Net framework.

Τα βασικά χαρακτηριστικά του .Net framework είναι τα εξής:

- Το .Net framework αναλαμβάνει τη διαχείριση της μνήμης κατά την εκτέλεση των εφαρμογών.
- Οι γλώσσες προγραμματισμού χρησιμοποιούν τους κοινούς τύπους δεδομένων με αποτέλεσμα να μην υπάρχουν προβλήματα ασυμβατότητας τύπων δεδομένων μεταξύ εφαρμογών που γράφτηκαν σε διαφορετικές γλώσσες προγραμματισμού.

[23] Προφέρεται Dot Net

[24] Οι εφαρμογές που εκτελούνται από το .Net αναφέρονται ως managed applications

- Η εκτεταμένη βιβλιοθήκη επιτρέπει την επαναχρησιμοποίηση έτοιμου κώδικα αντί να πρέπει ο χρήστης να τον γράψει ο ίδιος.
- Παρέχει διάφορες τεχνολογίες όπως το ASP.Net για εκτέλεση κώδικα σε ιστοσελίδες ή το ADO.Net για πρόσβαση σε βάσεις δεδομένων.
- Οι compilers που αξιοποιούν το .Net παράγουν έναν ενδιάμεσο κώδικα (MSIL) ο οποίος είναι ο ίδιος ανεξάρτητα από τη γλώσσα που γράφτηκε ή την πλατφόρμα που εκτελείται.
- Είναι δυνατή η συνύπαρξη και εκτέλεση του ίδιου κώδικα σε διαφορετικές εκδόσεις, ακόμα και στην ίδια έκδοση για διαφορετικές γλώσσες. Αυτό έχει σαν αποτέλεσμα την ελαχιστοποίηση ασυμβατότητας μεταξύ παλιότερων και νεότερων εφαρμογών.
- Με τη βιβλιοθήκη .Net framework είναι δυνατή η ανάπτυξη εφαρμογών για διάφορες πλατφόρμες όπως windows 7, windows 8, windows 10 κτλ.

Για την ανάπτυξη της εφαρμογής μας χρησιμοποιήσαμε σαν περιβάλλον ανάπτυξης το Visual Studio 2013 με κώδικα ASP.Net και webforms, ενώ η Visual Basic ήταν η βασική γλώσσα προγραμματισμού[25].

[25] Κατά το χρόνο συγγραφής της διπλωματικής εργασίας προσφέρεται δωρεάν η έκδοση Visual Studio Community

6

Ανάλυση του προβλήματός μας και υλοποίηση της εφαρμογής

6.1 Κίνητρα και γενική περιγραφή

Μια από τις προκλήσεις ενός συστήματος είναι προφανώς η υλοποίησή του. Στο προηγούμενο κεφάλαιο εξετάσαμε αρκετά προγραμματιστικά εργαλεία και πλατφόρμες που δίνουν την δυνατότητα να κατασκευάσουμε την δική μας εφαρμογή στην οποία θα ενσωματώσουμε χαρακτηριστικά Information Retrieval και Faceted Search. Θέλουμε να δείξουμε ένα παράδειγμα ενσωμάτωσης αυτής της τεχνολογίας σε μία demo εφαρμογή αποδεικνύοντας έτσι τις δυνατότητες και την ευελιξία που προσφέρει το faceted search.

Όμως, το σημαντικότερο κίνητρο υπήρξε η διαπίστωση ότι παρόλο που υπάρχουν τόσα εργαλεία και υπηρεσίες διαθέσιμα, σήμερα αρκετές λύσεις συστημάτων αναζήτησης πληροφοριών στερούνται των δυνατοτήτων που προσφέρει το faceted search. Αν πάμε ένα βήμα πιο πίσω θα διαπιστώσουμε ότι πολλές εφαρμογές στερούνται μια αξιόπιστης, γρήγορης και προσαρμόσιμης λύσης για αναζήτηση δεδομένων και χρησιμοποιούν μηχανισμούς αναζήτησης με μεγάλα προβλήματα όσον αφορά στο scaling, στον τύπο εγγράφων, στην ευελιξία επιλογής πηγών δεδομένων και άλλα χαρακτηριστικά που βρίσκει κανείς στο Lucene και άλλα τέτοια χαρακτηριστικά. Για παράδειγμα, μία εφαρμογή που αρχειοθετεί νομικά έγγραφα έχει μεγάλες ανάγκες αναζήτησης βάσει λέξεων κλειδιών ή βάσει ταξινομήσεων των

εγγράφων ανάλογα με τα περιεχόμενά τους. Το δικό μας παράδειγμα έχει σκοπό να δείξει πόσο εφικτό είναι με προγραμματιστικά εργαλεία ελεύθερα διαθέσιμα να υποστηρίξουμε τέτοιες εφαρμογές.

Στο δικό μας παράδειγμα τα δεδομένα μας προέρχονται από κάποια βάση δεδομένων δηλαδή τα έγγραφα της συλλογής μας είναι οι εγγραφές της βάσης δεδομένων. Εύκολα μπορούμε να φανταστούμε εφαρμογές που οι εγγραφές βρίσκονται σε βάσεις δεδομένων και χρειάζεται να έχουμε χαρακτηριστικά αναζήτησης. Για παράδειγμα ένα σύστημα άρθρων που δημοσιεύονται μέσω κάποιου Content Management System στα οποία απαιτείται η αναζήτηση στα περιεχόμενα των κειμένων. Η χρήση μίας δομημένης δομής εγγράφων όπως μία βάση δεδομένων εξυπηρετεί την απλότητα του παραδείγματός μας και σε καμία περίπτωση δεν διαφοροποιεί τον σκοπό μας ή τα διαθέσιμα εργαλεία.

Για την ακρίβεια χρησιμοποιήθηκαν δύο διαφορετικές βάσεις δεδομένων, μία βασισμένη σε SQL Server και μία σε MySQL. Ο λόγος της χρήσης των δύο συστημάτων διαχείρισης βάσεων δεδομένων είναι η απόδειξη της ευελιξίας του συστήματος που μπορεί να προσαρμόζεται ικανοποιητικά και να εξυπηρετεί διαφορετικές πηγές δεδομένων κατά τον χρόνο εκτέλεσης.

Προφανώς, βασικός στόχος μας ήταν να εξετάσουμε κατά πόσο είναι εύκολη η υλοποίηση ενός συστήματος με χαρακτηριστικά αναζήτησης το οποίο να βασίζεται σε ευρέως διαθέσιμα εργαλεία για information retrieval και όχι σε κλειστά συστήματα ή συνδρομητικές λύσεις παροχής ανάλογων υπηρεσιών.

6.2 Τα εργαλεία μας

Όπως αναφέραμε νωρίτερα, η εφαρμογή μας αναπτύχθηκε σαν μία web εφαρμογή με τη χρήση του Microsoft Visual Studio 2013. Χρησιμοποιήθηκαν Web Forms με server side κώδικα κυρίως σε VB.NET.

Αξιοποιήσαμε σχεδιαστικά εργαλεία όπως master pages για την ομοιόμορφη εμφάνιση των σελίδων μας, ενώ σε κάποια σημεία χρειάστηκαν μικρές επεμβάσεις στα αρχεία CSS για την προσαρμογή του γραφιστικού συστήματος.

Η βάσεις δεδομένων που χρησιμοποιήθηκαν ήταν δύο:

- Μία MS SQL με δεδομένα από ένα σύστημα αγγελιών. Ο βασικός πίνακας στον οποίο κάνουμε αναζήτηση περιλαμβάνει 5803 αγγελίες με κατηγορίες όπως τύπος αγγελίας, περιοχή αναζήτησης, κατηγορία αγγελίας κτλ.

- Ο δεύτερος πίνακας προερχόταν από την sample βάση δεδομένων Adventureworks2014[26] της Microsoft. Περιλαμβάνει χιλιάδες εγγραφές μίας υποθετικής εταιρείας εμπορίας ποδηλάτων. Η βάση έτρεχε πάνω σε MS SQL Server 2014 express edition.

Στην πράξη η εφαρμογή μπορεί να χρησιμοποιήσει σχεδόν οποιαδήποτε άλλη βάση δεδομένων την οποία ο χρήστης καθορίζει ως προφίλ σύνδεσης με την πηγή δεδομένων σε ένα αρχείο ρυθμίσεων το "datasources.xml".

Για λόγους απλότητας αλλά και ευκολίας παρέμβασης στις ρυθμίσεις του συστήματος χρησιμοποιήθηκαν αρχεία ρυθμίσεων XML. Η πρακτική αυτή δίνει τη δυνατότητα οι ρυθμίσεις να γράφονται και να διαβάζονται είτε αυτοματοποιημένα μέσω κάποιου περιβάλλοντος διαχείρισης, όσο και χειρωνακτικά με παρέμβαση του χρήστη. Τα αρχεία XML είναι τόσο machine όσο και human readable.

Το σημαντικότερο εργαλείο που χρησιμοποιήσαμε ήταν η βιβλιοθήκη αναζήτησης Lucene.Net (για την οποία κάναμε αναφορά στην παράγραφο 5.1.3). Η βιβλιοθήκη είναι διαθέσιμη μέσω του Nuget Package Manager [2] και η εγκατάστασή στην εφαρμογή της είναι ζήτημα δευτερολέπτων.

Με τη βιβλιοθήκη Lucene.Net χρησιμοποιήσαμε το BoboBrowse, έναν wrapper που προσθέτει χαρακτηριστικά faceted search και σχετική απλότητα στα αντικείμενα. Πρέπει να αναφέρουμε ότι τα εν γένει αντικείμενα του Lucene.NET υποστηρίζουν faceted search αλλά η χρήση τους προϋποθέτει πολύ πιο σύνθετο κώδικα.

6.3 Αρχεία ρυθμίσεων

Ήδη αναφέραμε ότι η εφαρμογή χρησιμοποιεί αρχεία ρυθμίσεων σε μορφή XML. Η χρήση αρχείων ρυθμίσεων δεν αποτελεί την βέλτιστη επιλογή για μια σύγχρονη εφαρμογή για μια σειρά από λόγους η σημαντικότερη των οποίων είναι η ασφάλεια. Μας βολεύει όμως γιατί παρέχει την απαραίτητη διαφάνεια προς τον χρήστη που καταγράφει τις επιλογές του και τις βλέπει να υλοποιούνται από το σύστημα.

Με τη χρήση των αρχείων ρυθμίσεων, φορτώνεται το προφίλ των δεδομένων του συστήματος και ρυθμίζονται διάφορες παράμετροι που αφορούν στο indexing και στο χτίσιμο των facets.

Μία γενική παρατήρηση σχετικά με τα αρχεία ρυθμίσεων είναι ότι είναι καθολικά. Δηλαδή οι ρυθμίσεις αφορούν σε όλους τους χρήστες της εφαρμογής. Αυτό όμως δεν αποτελεί

[26] Κατεβάστε τη ΒΔ σε μορφή εγκατάστασης στο http://msftdbprodsamples.codeplex.com/

πρόβλημα για δύο λόγους. Αρχικά η διαχειριστικές σελίδες που δημιουργούν και αλλάζουν τα αρχεία ρυθμίσεων θα πρέπει να βρίσκονται στη διάθεση μόνο του διαχειριστή Εάν οι λειτουργικές απαιτήσεις της εφαρμογής επιτρέπουν στους χρήστες να φτιάχνουν τα δικά τους αρχεία ρυθμίσεων αυτά μπορεί να αποθηκεύονται σε βάση δεδομένων πάντα στο security context του κάθε χρήστη. Οι δύο παραπάνω προσεγγίσεις μπορούν εύκολα να υλοποιηθούν.

Αναφέρουμε παρακάτω τα αρχεία ρυθμίσεων και την χρήση του καθενός.

6.3.1 Αρχείο datasources.xml

Για κάθε βάση δεδομένων περιγράφει σε elements <conf_table> σχετικές πληροφορίες όπως τον τύπο της βάσης, το όνομά της, το connectionstring δηλαδή το αλφαριθμητικό που χρησιμοποιείται για να συνδεθεί η εφαρμογή με τις βάσεις δεδομένων και τον φάκελο indexfolder. Για κάθε μία σύνδεση ορίζουμε διαφορετικό φάκελο μέσα στον οποίο η εφαρμογή θα χτίσει και μετά θα χρησιμοποιήσει τον index.

```
<?xml version="1.0" standalone="yes"?>
<ConfigData xmlns="http://tempuri.org/LuceneIndexConfigDataSet.xsd">
  <conf_table>
    <id>0</id>
    <dbtype>mysql</dbtype>
    <dbname>Ads</dbname>
    <indexfolder>indexfolder1</indexfolder>
    <connectionstring>server=db33.grserver.gr;user id=apostolos;password=X;persistsecurityinfo=True;database=adsdesktop
    </connectionstring>
  </conf_table>
  <conf_table>
    <id>1</id>
    <dbtype>ms sql</dbtype>
    <dbname>Adventureworks</dbname>
    <indexfolder>indexfolder2</indexfolder>
    <connectionstring>Data Source=APO-VAIO\SQLEXPRESS;Initial Catalog=AdventureWorks2014;Integrated Security=True
    </connectionstring>
  </conf_table>
</ConfigData>
```

Εικόνα 33: Αρχείο datasoucre.XML

6.3.2 Αρχείο configData.XML

Μέσα σε κάθε φάκελο indexfolder υπάρχει το XML αρχείο configData.XML. Πρακτικά ορίζει τον πίνακα και τα πεδία της βάσης δεδομένων που θα χρησιμοποιηθούν για το χτίσιμο του Index. Για κάθε πεδίο ορίζει εάν είναι το κλειδί του πίνακα, εάν πρέπει να γίνει ανάλυση σε όρους (λέξεις κλειδιά) ή όχι, και ένα θα πρέπει να αποθηκευτεί η τιμή του πεδίου στον index, διαφορετικά αποθηκεύονται δυαδικές τιμές για να χρησιμοποιηθούν στην αναζήτηση.

Γενικά ο Index θα πρέπει να είναι σε θέση να ικανοποιήσει τα ερωτήματα σε κάποια πεδία και να μας επιστρέψει μόνο τα κλειδιά των εγγραφών που κάνουν match με το ερώτημά μας. Η ανάκτηση της εγγραφής θα πρέπει να γίνεται από τη βάση. Σε μερικές περιπτώσεις όμως, η αποθήκευση μέσα στον index έχει νόημα ώστε να μπορούμε να διαβάσουμε κάποιες τιμές απευθείας από τον Index χωρίς να χρειάζεται να ανατρέξουμε στην αρχική εγγραφή για λόγους performance.

Στην παράγραφο 3.9 αναφέραμε τους τρεις διαφορετικούς τρόπους με τους οποίους ένα σύστημα μπορεί να επιλέγει τα facets. Το δικό μας demo σύστημα ανήκει στην πρώτη κατηγορία δηλαδή τα facets προεπιλέγονται χειρωνακτικά από τον χρήση. Αυτό απαιτεί τα πεδία να έχουν προετοιμασθεί κατάλληλα δηλαδή να έχουμε εμπλουτίσει τον index με τα απαραίτητα μεταδεδομένα .

Για να αξιοποιήσουμε έναν πίνακα για faceted search πρέπει να περιλαμβάνει όλα τα επιπλέον χαρακτηριστικά (metadata) για κάθε εγγραφή που θα αποτελέσουν τις τιμές των facets. Αυτά θα πρέπει να βρίσκονται εντός του ίδιου του πίνακα, με άλλα λόγια ο πίνακας γίνεται flattened με τις επιπλέον τιμές από άλλου συνδεδεμένους πίνακας. Αυτό μπορεί να γίνει εύκολα με ένα View και αυτή τη λύση ακολουθήσαμε και εμείς. Γενικά τα δεδομένα μας πρέπει να ακολουθήσουν μία διαδικασία αποκανονικοποίησης για να είναι έτοιμα για facets. Εφόσον γίνει αυτό, ο υπολογισμός των τιμών των facet καθώς και το rank τους για κάθε ερώτημα γίνεται αυτόματα από το σύστημα.

Μία καλή πρακτική είναι τα πεδία που θα χρησιμοποιήσουμε για facet όπως για παράδειγμα η κατηγορίας ενός προϊόντος, να μην περιλαμβάνουν τιμές null. Η αποφυγή των null τιμών ή η αντικατάστασή τους με κάποιο string (π.χ. “Χωρίς Κατηγορία”) θα γίνει στο χτίσιμο του view.

```
<?xml version="1.0" standalone="yes"?>
<LuceneIndexConfigDataSet xmlns="http://tempuri.org/LuceneIndexConfigDataSet.xsd'
  <conf_table>
    <id>1</id>
    <itemtype>table</itemtype>
    <itemname>ProductsFacets</itemname>
    <analyzed>false</analyzed>
    <facetfield>false</facetfield>
    <idfield>false</idfield>
    <textfield>false</textfield>
    <parent>0</parent>
  </conf_table>
  <conf_table>
    <id>2</id>
    <itemtype>field</itemtype>
    <itemname>ProductID</itemname>
    <analyzed>False</analyzed>
    <facetfield>false</facetfield>
    <idfield>true</idfield>
    <textfield>false</textfield>
    <parent>1</parent>
  </conf_table>
  <conf_table>
    <id>3</id>
    <itemtype>field</itemtype>
    <itemname>Name</itemname>
    <analyzed>true</analyzed>
    <facetfield>false</facetfield>
    <idfield>false</idfield>
```

Εικόνα 34:Αρχείο configData.XML

6.3.3 *Αρχείο Bobo.spring*

Κατά τη διάρκεια της ανάκτησης δεδομένων και εφόσον βέβαια ο index υπάρχει ήδη, απαιτείται να ορίσουμε ποια θα είναι τα facets για τα δεδομένα μας όπως για παράδειγμα η περιοχή για τις αγγελίες ή το χρώμα για τα προϊόντα. Ο ορισμός των facet μπορεί να γίνει προγραμματιστικά δηλαδή να συμπεριληφθεί κώδικας για αυτό ή εναλλακτικά μπορεί να χρησιμοποιηθεί ένα αρχείο Bobo.Spring. Τα Spring files [27] είναι XML αρχεία με συγκεκριμένο schema τα οποία εφόσον βρίσκονται μέσα στον φάκελο με τον Index η βιβλιοθήκη Bobobrowse τα φορτώνει αυτόματα διαβάζοντας τα facets που θα πρέπει να χρησιμοποιήσει.

Αυτό είναι μία ευέλικτη αρχιτεκτονική επιλογή γιατί βάζοντας τα Bobo.Spring αρχεία μέσα σε κάθε φάκελο που έχουμε τον Index μας μπορούμε για κάθε Index να ορίσουμε ανεξάρτητο σετ από facets που θα αξιοποιηθούν στο επόμενο στάδιο που είναι οι αναζητήσεις στον συγκεκριμένο index.

```
<?xml version="1.0" encoding="UTF-8"?>
<objects xmlns="http://www.springframework.net"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xsi:schemaLocation="http://www.springframework.net http://www.springframework.net/xsd/spring-objects.xsd" >
  <object id="Color" type="BoboBrowse.Net.Facets.Impl.SimpleFacetHandler, BoboBrowse.Net">
   <constructor-arg value="Color" />
  </object>
  <object id="ListPrice" type="BoboBrowse.Net.Facets.Impl.RangeFacetHandler, BoboBrowse.Net">
    <constructor-arg name="name" value="ListPrice"/>
    <constructor-arg name="termListFactory">
      <object type="BoboBrowse.Net.Facets.Data.PredefinedTermListFactory, BoboBrowse.Net">
        <constructor-arg value="System.Int32"/>
        <constructor-arg value="00000000000000000000" />
      </object>
    </constructor-arg>
    <constructor-arg name="predefinedRanges">
      <list element-type="string">
        <value>[0 TO 100]</value>
        <value>[101 TO 500]</value>
        <value>[501 TO 1000]</value>
        <value>[1001 TO 2000]</value>
        <value>[2001 TO 9999]</value>
      </list>
    </constructor-arg>
  </object>
  <object id="CategoryName" type="BoboBrowse.Net.Facets.Impl.SimpleFacetHandler, BoboBrowse.Net">
    <constructor-arg value="CategoryName" />
  </object>
</objects>
```

Εικόνα 35:Spring file

6.4 *Λειτουργικότητα της εφαρμογής*

Στο παρακάτω σχήμα φαίνεται διαγραμματικά η λειτουργικότητα της εφαρμογής. Υπάρχει μια αρχική οθόνη υποδοχής (Intro Page) μέσω της οποίας είναι διαθέσιμες οι υπόλοιπες λειτουργίες της εφαρμογής.

[27] http://www.springframework.net/

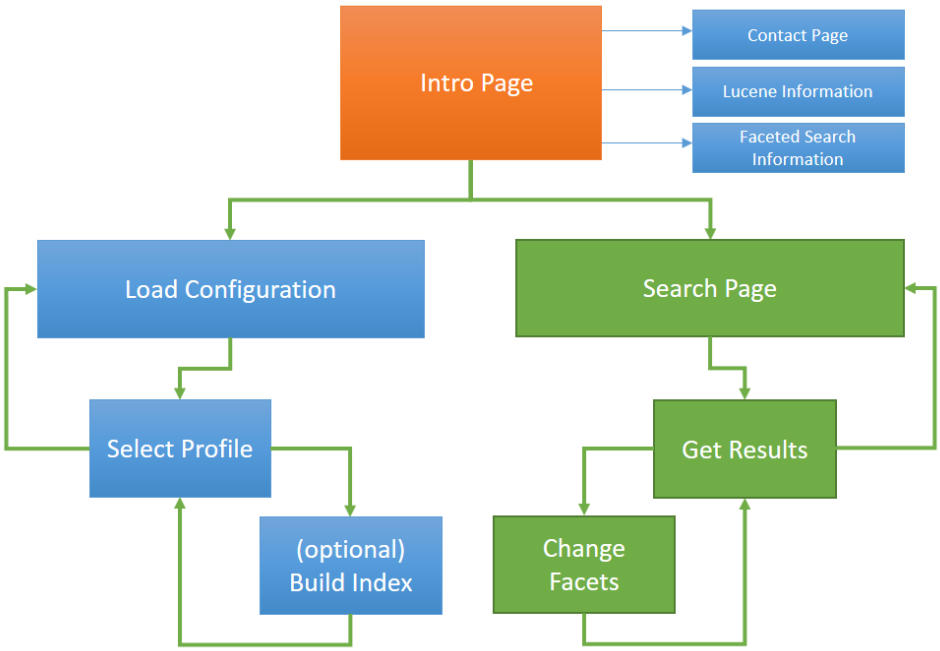
Αρχικά, από την αρχική σελίδα μπορούμε να κατευθυνθούμε σε τρεις πληροφοριακές σελίδες, μία για επικοινωνία και πληροφορίες της εφαρμογής, μίας σχετική με το Lucene και μία για το Faceted Search. Οι σελίδες αυτές βρίσκονται στην demo εφαρμογή για να εμπλουτίσουν την εμπειρία με τον χρήστη χωρίς καμία ιδιαίτερη λειτουργικότητα.

Οι δύο επόμενες σελίδες περιλαμβάνουν τις σημαντικές λειτουργίες της εφαρμογής.

Πρώτη από αυτές είναι μία διαχειριστική σελίδα, η Load Configuration. Η σελίδα αυτή υλοποιεί ένα υποτυπώδες περιβάλλον διαχείρισης για τον χρήστη από το οποίο μπορεί:

- Από εδώ μπορεί να επιλέξει και να φορτώσει ένα από τα διαθέσιμα προφίλ. Τα προφίλ διαβάζονται δυναμικά από το αρχείο datasources.xml.
- Αν δεν υπάρχει ο index για το επιλεγμένο προφίλ μπορεί να τον δημιουργήσει. Όμοια, μπορεί να τον χτίσει εκ νέου για οποιοδήποτε λόγω όπως γιατί άλλαξαν τα δεδομένα στη βάση δεδομένων ή γιατί έχει αλλάξει η επιλογή των πεδίων.

Από την αρχική σελίδα επίσης μπορούμε να μεταβούμε στην κεντρική σελίδα της αναζήτησης. Η αναζήτηση περιλαμβάνει ένα απλό πεδίο κειμένου και ένα κουμπί εντολής υλοποιώντας ένα βασικό search button από το οποίο ξεκινάει κάθε διαδικασία αναζήτησης. Η αρχική αναζήτηση λειτουργεί ψάχνοντας εντός του index για τους όρους που σχηματίζονται από το κείμενο του χρήστη. Θυμίζουμε ότι στη διαδικασία εμπλέκονται τα αντικείμενα Analyzers του Lucene. Εμείς επιλέξαμε να χρησιμοποιήσουμε τον SimpleAnalyzer για λόγους απλότητας.



Εικόνα 36:Διάγραμμα λειτουργιών

Μετά την αρχική αναζήτηση, υπολογίζονται τα facets τα οποία εμφανίζονται σε αριστερό sidebar σαν ένα πλήθος επιλογών. Ο υπολογισμός των facet γίνεται αυτόματα από το lucene το οποίο υπολογίζει επίσης κάτι πολύ σημαντικό για κάθε facet. Το hitCount. Η εφαρμογή εμφανίζει το hitCount δίπλα στο όνομα κάθε facet γιατί όπως προκύπτει από την ανάλυση της σχεδίασης αυτό είναι ο πιο σημαντικός παράγοντας preview για τον χρήστη. Γνωρίζει δηλαδή πόσα έγγραφα (η εγγραφές της βάσης δεδομένων) θα επιστρέψει το σύστημά μας αν επιλέξει το συγκεκριμένο facet. Το hitCount λειτουργεί και διαφορετικά δίνοντας στον χρήστη την αίσθηση της επιτυχίας και της σωστής κατεύθυνσης των ερωτημάτων του.

Η επιλογή ενός από τα facets υλοποιεί ενέργεια zoom-in (drill-down) και μετά από αυτή τα αποτελέσματα περιορίζονται, αλλά επίσης επίσης τα facets και οι τιμές τους επαναϋπολογίζονται.

Στην εφαρμογή μας, μετά από κάθε αναζήτηση τα αποτελέσματα εμφανίζονται σε ένα gridview το οποίο έχει σελιδοποίηση και υλοποιεί την υποτιθέμενη διεπαφή με τον χρήστη. Μπρούμε να πούμε ότι τιμές του gridview είναι τα snippets των αποτελεσμάτων μας. Σε αυτά μπορούμε να περιλάβουμε διάφορα πληροφοριακά δεδομένα όπως κατηγοριοποίηση, τιμές rank κτλ.

Στο επόμενο στάδιο η εφαρμογή πρέπει να επεξεργάζεται το κλικ του χρήστη σε κάποια εγγραφή και να τον οδηγεί στην αρχική εγγραφή της βάσης δεδομένων ή κάτι αντίστοιχο (έγγραφο, αρχείο, XML κτλ) αλλά το χτίσιμο των περαιτέρω λειτουργιών δεν περιλαμβάνεται στους σκοπούς μας.

Στο παράρτημα παρατίθενται κομμάτια από τον κώδικα των λειτουργιών αλλά επίσης και screenshots από τις οθόνες της εφαρμογής.

7

Αξιολόγηση του συστήματος

7.1 Οργάνωση πειράματος

Ο σκοπός της εργασίας μας υπήρξε η μελέτη των συστημάτων information retrieval σε συνδυασμό με την τεχνική του faceted search όπως επίσης η ανασκόπηση των σχεδιαστικών προβληματισμών και των διαθέσιμων προγραμματιστικών εργαλείων.

Η αξιολόγησή του συστήματος έχει ως σκοπό να αποδείξει την προσαρμοστικότητα ενός περιβάλλοντος αναζήτηση με facets και την βελτίωση της εμπειρίας με τον χρήστη κατά τη διαδικασία της εξερεύνησης. Η αξιολόγησή μας περιελάβανε ένα ερωτηματολόγιο στο οποίο καταγράφονταν τέσσερις απλές διεργασίες που έπρεπε να φέρουν εις πέρας οι συμμετέχοντες στη διαδικασία της αξιολόγησης.

Το σύστημα αναζήτησης το οποίο χρησιμοποίησαν ήταν η demo εφαρμογή που χτίσαμε και τα ερωτήματα απευθυνόταν στην demo βάση δεδομένων AdventureWorks2014.

Οι συμμετέχοντας στη διαδικασία της αξιολόγησης ήταν δώδεκα διοικητικοί υπάλληλοι της δευτεροβάθμιας εκπαίδευσης δυτικής Θεσσαλονίκης. Σε όλους εξηγήθηκε ο σκοπός της πειραματικής εφαρμογής και το περιβάλλον της εφαρμογής σε συντομία, ενώ παράλληλα έγινε απλή επίδειξη της χρήσης της εφαρμογής χρησιμοποιώντας την δεύτερη βάση δεδομένων με τις αγγελίες. Στο κύριο μέρος της διαδικασίας, τους ζητήσαμε να υλοποιήσουν τις διεργασίες του ερωτηματολόγιου, που πρακτικά ήταν ερωτήματα επί της βάσης δεδομένων, και να απαντήσουν σε ένα πλήθος ερωτήσεων. Στο επόμενο στάδιο

επεξεργαστήκαμε τις απαντήσεις των συμμετεχόντων χρησιμοποιώντας το Excel και διαξάγαμε τα αποτελέσματα τα οποία και παρουσιάζουμε παρακάτω.

Τα ερωτήματα τα οποία έφεραν εις πέρας ήταν τα εξής:

- Εντοπίστε τα ακριβότερα προϊόντα
- Εντοπίστε τα φτηνότερα προϊόντα από τα ”mountain bikes”
- Εντοπίστε τα προϊόντα με όνομα κάτι σαν “Mountain”και της κατηγορίας “Wheel”
- Εντοπίστε τα χρώματα από τα ακριβότερα ποδήλατα της κατηγορίας “touring Bikes”

Το ερωτηματολόγιο περιελάβανε 14 ερωτήσεις τις οποίες βαθμολόγησαν στην κλίμακα από 1 έως 5. Σε όλες τις ερωτήσεις η κλίμακα 5 αντιστοιχεί στην πλέον θετική όψη της ερώτησης και η κλίμακα 1 στην λιγότερο θετική. Για παράδειγμα στις ερωτήσεις που αναφέρονται στην κάλυψη των αναγκών βαθμολογούνται με 5 οι απαντήσεις που κάλυψαν πολύ ικανοποιητικά τις ανάγκες των χρηστών και με 1 αυτά που δεν τις κάλυψαν καθόλου. Σε αυτές που αναφέρονται στον χρόνο αναζήτησης στα αποτελέσματα, η κλίμακα 5 αντιστοιχεί σε μικρό χρόνο αναζήτησης, δηλαδή γρήγορη κάλυψη της ανάγκης μας και η κλίμακα 1 σε μεγαλύτερο χρόνο αναζήτησης, δηλαδή σε σχετικά αργή κάλυψη της ανάγκης μας.

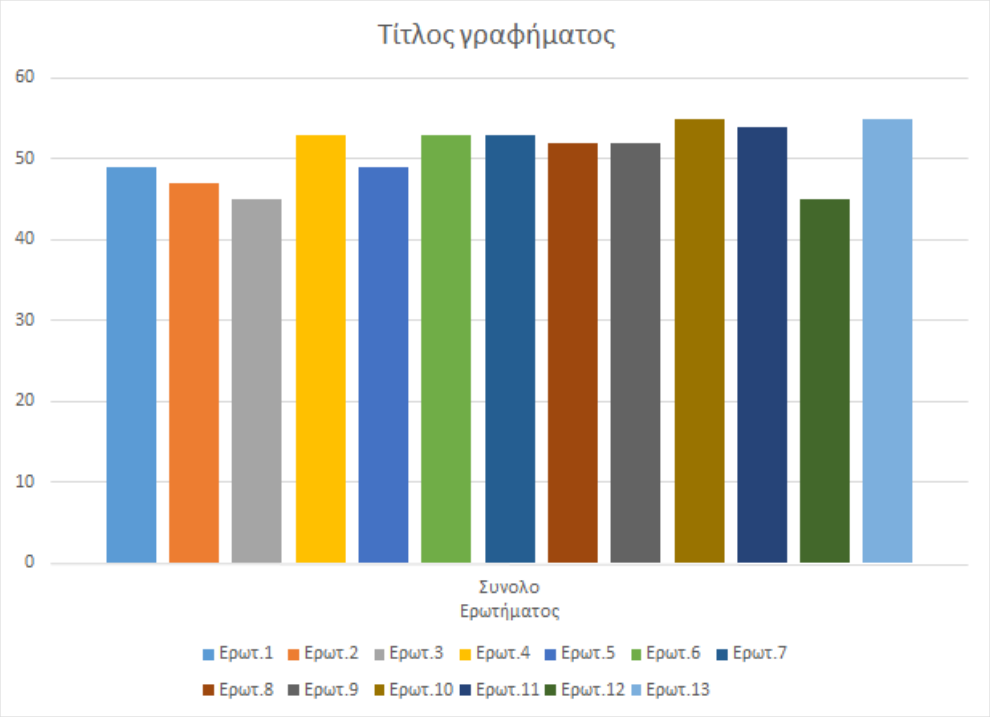
Οι ερωτήσεις ήταν οι εξής:

1. Πόσο εξοικειωμένος είστε με τη χρήση Η/Υ;
2. Πόσο εξοικειωμένοι είστε με τη χρήση μηχανών αναζήτησης (όπως Google, Bing)
3. Πόσο εξοικειωμένοι είστε με την αναζήτηση σε web site τύπου e-commerce.
4. Πόσο δύσκολο ήταν να φέρετε εις πέρας το ερώτημα 1;
5. Σε ποιο βαθμό πιστεύετε ότι καλύψατε το σύνολο του ερωτήματος 1;
6. Πόσο δύσκολο ήταν να φέρετε εις πέρας το ερώτημα 2;
7. Σε ποιο βαθμό πιστεύετε ότι καλύψατε το σύνολο του ερωτήματος 2;
8. Πόσο δύσκολο ήταν να φέρετε εις πέρας το ερώτημα 3;
9. Σε ποιο βαθμό πιστεύετε ότι καλύψατε το σύνολο του ερωτήματος 3;
10. Πόσο δύσκολο ήταν να φέρετε εις πέρας το ερώτημα 4;
11. Σε ποιο βαθμό πιστεύετε ότι καλύψατε το σύνολο του ερωτήματος 4;
12. Πόσο δύσκολο ήταν να φέρετε εις πέρας το ερώτημα 5;
13. Σε ποιο βαθμό πιστεύετε ότι καλύψατε το σύνολο του ερωτήματος 5;
14. Από το χρόνο που απαιτήθηκε συνολικά για την εκπλήρωση των ασκήσεων, ποιο ήταν το ποσοστό του χρόνου μέσα στον οποίο χρειάστηκε να αναζητήσετε μέσα στη λίστα αποτελεσμάτων;
    a. λιγότερο από 25%,
    b. μεταξύ 25% και 50%
    c. μεταξύ 50% και 75%

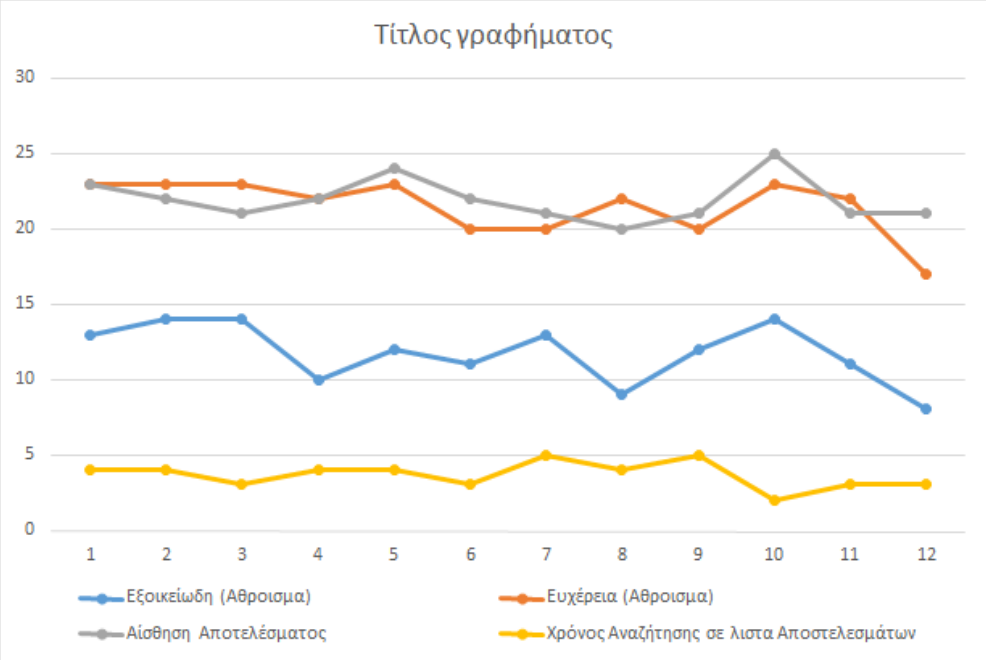
d. Πάνω από 75%

7.2 Επεξεργασία αποτελεσμάτων

Από τα αποτελέσματα των ερωτηματολογίων και για την αξιολόγηση του συστήματός μας κατασκευάσαμε τα εξής γραφήματα :



Εικόνα 37: Άθροισμα Βαθμολογίας ανά ερώτημα



Εικόνα 38:Σύγκριση εξοικείωσης, Ευχέρειας, Αίσθησης αποτελέσματος και Χρόνου αναζήτησης

Στο γράφημα 01 εμφανίζονται τα αθροίσματα όλων των βαθμολογιών για όλες τις ερωτήσεις από όλους του χρήστες. Παρατηρούμε ότι δεν υπάρχουν μεγάλες διαφορές μεταξύ των απαντήσεων για τις διαφορετικές ερωτήσεις και επίσης ότι τα αποτελέσματα είναι σχετικά υψηλά κάτι που αποδεικνύει την αποδοχή του συστήματος.

Στο γράφημα 02 εμφανίζονται συγκριτικά γραφήματα για τα εξής:

- Εξοικείωση με συστήματα αναζήτησης (ερωτήσεις 1,2,3)
- Ευχέρεια στην χρήση του συστήματος (ερωτήσεις 4,6,8,10,12)
- Αίσθηση εκπλήρωσης αποτελέσματος (ερωτήσεις 5,7,9,11,13)
- Χρόνος αναζήτησης (ερώτηση 14)

Αυτό που παρατηρούμε είναι ότι τα διαγράμματα κινούνται αναλογικά. Δηλαδή, τα ερωτήματα που δυσκόλεψαν λίγο περισσότερο τους χρήστες είχαν αντίστοιχο αποτέλεσμα στην αίσθηση αποτελέσματος και στον χρόνο της αναζήτησης.

8

Κεφάλαιο 8: Επίλογος, συμπεράσματα,

Σκοπός τη εργασίας μας ήταν η διερεύνηση και αξιολόγηση της τεχνολογίας του **faceted search**. Αφορμή υπήρξε η διαπίστωση της τρέχουσας κατάστασης στο πεδίο των συστημάτων αναζήτησης. Οι σημερινές ανάγκες για αναζήτηση πληροφορίας έχουν αλλάξει δραματικά σε σχέση με το πρόσφατο παρελθόν. Ο όγκος της πληροφορίας που πρέπει να διαχειριστούμε είναι πλέον τεράστιος και τα διαθέσιμα μέσα για την προσπέλαση της πληροφορίες έχουν αλλάξει. Εκτός από μικροϋπολογιστές και μεγαλύτερα συστήματα έχουν προστεθεί μικρές φορητές συσκευές όπως smartphones, tablet, μηχανισμοί που συνδέονται στο διαδίκτυο και παράγουν πληροφορίες όπως αισθητήρες, συστήματα αυτοματισμού κτλ. Όλο και περισσότερο, η αναζήτηση της πληροφορίας γίνεται όχι απλά με τους κλασικούς μηχανισμούς αναζήτησης που γνωρίζαμε αλλά και με παρασκηνιακές διαδικασίες που δεν είναι τόσο διαφανείς στον χρήστη, όπως για παράδειγμα, η πλοήγηση σε ένα σύστημα με χάρτες. Ο παράγοντας χρόνος γίνεται ολοένα και πιο πιεστικός καθώς το περιβάλλον της αναζήτησης της πληροφορίας είναι ολοένα και πιο ανταγωνιστικό. Για παράδειγμα, κανένας δεν θα χρησιμοποιήσει ένα e-commerce στο οποίο η αναζήτηση προϊόντων καθυστερεί πολύ ή δεν καλύπτει τις ανάγκες μας, όπως επίσης κανένας δεν θα χρησιμοποιήσει ένα σύστημα αναγνώρισης μουσικής (όπως για παράδειγμα το Shazam) το οποίο ανταποκρίνεται μετά από μερικά λεπτά.

Όλα αυτά έχουν δημιουργήσει ένα εκρηκτικό περιβάλλον στο οποίο η άμεση προσπέλαση στην πληροφορία είναι εξαιρετικά κρίσιμη.

Όμως εκτός από τις τεχνικές δυσκολίες που αφορούν στην υλοποίηση των σύγχρονων συστημάτων αναζήτησης η άποψή μας είναι ότι ένας πολύ σημαντικός παράγοντας είναι το μοντέλο αλληλεπίδρασης του χρήστη με τα συστήματα αναζήτησης. Η σύγχρονη πρακτική μας δείχνει ότι το μοντέλο αυτό έχει διαφοροποιηθεί και απομακρύνεται από το κλασικό μοντέλο του information retrieval όπου ο χρήστης υποβάλλει μία σειρά από λέξεις κλειδιά προς ένα σύστημα αναζήτησης και περιμένει αποτελέσματα τα οποία θα εξετάσει. Η σύγχρονη τάση οδηγεί προς μία πιο διαδραστική διαδικασία κατά την οποία ο χρήστης εξερευνεί σε κάθε στάδιο τις επιλογές του και βήμα προς βήμα προχωρά στην αναζήτηση της γνώσης ή αλλιώς **information seeking**. Οι παλιότερες εφαρμογές συστημάτων αναζήτησης δεν ταιριάζουν στο μοντέλο του Information seeking και αποτυγχάνουν να ικανοποιήσουν τις ανάγκες των χρηστών. Αυτός υπήρξε και ο κύριος παράγοντας αποτυχίας και ανάγκης εκσυγχρονισμού τους.

Καταλήξαμε λοιπόν στο συμπέρασμα ότι η ανάγκη του χρήστη ικανοποιείται μέσω μιας εξερευνητικής διαδικασίας στην οποία ο χρήστης ίσως δεν γνωρίζει αρκετά ούτε το πεδίο αναζήτησης ούτε και τι είναι αυτό που ψάχνει. Αναφερόμαστε σε αυτό το μοντέλο με τον όρο εξερευνητική διαδικασία αναζήτησης (exploratory search). Στο παρελθόν, αρκετές προσπάθειες έχουν γίνει για να ικανοποιήσουν το μοντέλο της εξερευνητικής αναζήτησης (exploratory search) όπως οι στατικοί κατάλογοι και τα παραμετρικά ερωτήματα, οι οποίες όμως δεν είχαν τα αναμενόμενα αποτελέσματα,

Τα τελευταία χρόνια το **faceted search** είναι η τεχνολογία που έχει καθιερωθεί σαν η de facto λύση για συστήματα όπως τα e-commerce. Τα χαρακτηριστικά του faceted search το καθιστούν κατάλληλο για τα περισσότερα συστήματα αναζήτησης καθώς φαίνεται να είναι το πλέον κατάλληλο για εξερευνητικές διαδικασίες αναζήτησης ενώ παράλληλα δίνει τέλος στις αδιέξοδες αναζητήσεις, στα κενά αποτελέσματα και στη συνεχή επαναδιατύπωση των ερωτημάτων από τους χρήστες. Το σημαντικό με αυτήν την τεχνική είναι ότι ταιριάζει καλύτερα στον τρόπο με τον οποίο αλληλοεπιδρά ο χρήστης με ένα σύγχρονο σύστημα αναζήτησης με αποτέλεσμα να βελτιώνει όλες τις μετρικές για τα συστήματα αναζήτησης καθοδηγώντας τον χρήστη και επιτρέποντάς τον να εξερευνεί ελεύθερα όλες τις διαθέσιμες επιλογές του. Η αξία του faceted search γίνεται εμφανής εάν αναλογιστούμε σε πόσες μεγάλες εγκαταστάσεις είναι σήμερα διαθέσιμο.

Στο παρόν κείμενο κάναμε μία εκτενή παρουσίαση του **faceted search**. Μελετήσαμε τα χαρακτηριστικά του, το θεωρητικό μοντέλο στο οποίο στηρίζεται, το μοντέλο αλληλεπίδρασης με τον χρήστη, τις προκλήσεις για την επιλογή των facets και τους διαφορετικούς τρόπους με τους οποίους μπορεί να γίνεται η επιλογή αυτή. Μιλήσαμε για τις δυναμικές ταξινομίες που αποτελούν τη βάση του συστήματος και κάναμε μία εκτενή αναφορά στα προβλήματα κατά την εφαρμογή του faceted search.

Εξερευνήσαμε επίσης τα διαθέσιμα εργαλεία με τα οποία μπορούμε να αναπτύξουμε μία διαδικτυακή εφαρμογή αναζήτησης με χαρακτηριστικά faceted search. Αυτά μπορεί να είναι έτοιμες πλατφόρμες όπως το drupal, ή υπηρεσίες αναζήτησης «out of the box» όπως το Amazon Cloudsearch, η αμιγώς λογισμικά αναζήτησης που μπορούμε να τα ενσωματώσουμε στις εφαρμογές μας όπως το lucene.

Πιστεύουμε ότι στη διαδικασία υλοποίησης ενός τέτοιου συστήματος αναζήτησης ένα σημαντικό στοιχείο του όλου συστήματος είναι η κατάλληλη σχεδίαση. Για το λόγο αυτό αναλύσαμε τους παράγοντες που επηρεάζουν την διαδικασία σε συνδυασμό με το πώς θα τους ικανοποιήσουμε με την κατάλληλη σχεδίαση. Προχωρώντας ένα βήμα ακόμα, αναλύσαμε ειδικά το θέμα της σχεδίασης ενός faceted search συστήματος εξετάζοντας και αξιολογώντας τις προτεινόμενες λύσεις.

Ωστόσο, η εφαρμογή του faceted search εγείρει μια σειρά από δευτερεύοντα ζητήματα:

- Η πολυπλοκότητα του συστήματος το οποίο ταυτόχρονα με τις διεργασίες της αναζήτησης πρέπει να ενημερώνει και να παρουσιάζει στον χρήστη τα facets και τις τιμές τους είναι ένα σημαντικό ζήτημα το οποίο πρέπει να αντιμετωπισθεί. Σε πολλές περιπτώσεις ο φόρτος είναι υπερβολικά μεγάλος.
- Η αυτόματη επιλογή των facets ειδικά όταν αυτά προκύπτουν από συλλογές που δεν είναι αυστηρά δομημένες, είναι επίσης ένα σημαντικό πεδίο έρευνας. Οι συλλογές αυτές που βρίσκονται σε αδόμητη ή ημιδομημένη κατάσταση, όπως για παράδειγμα τα κείμενα των εγγράφων μίας βιβλιοθήκης, είναι μάλλον οι πλέον συνηθισμένες.
- Θα θέλαμε το faceted search να αποκτήσει την εκφραστικότητα άλλων συστημάτων όπως τα ερωτήματα σε δεδομένα σημασιολογικού ιστού για παράδειγμα με χρήση της SPARL και της OWL τα οποία είναι πολύπλοκα και προς το παρόν δεν είναι κατάλληλα για άμεση χρήση από μη εξειδικευμένους χρήστες. Εκτός αυτού, η διασύνδεση ενός συστήματος αναζήτησης με faceted search με άλλες τεχνολογίες του σημασιολογικού ιστού είναι επίσης σημαντικό πεδίο έρευνας.
- Το faceted search βασίζεται σε ένα σχετικά απλό μοντέλο το οποίο ταυτόχρονα λειτουργεί περιοριστικά. Δηλαδή μπορούμε να εξερευνήσουμε τις επιλογές μας για τις τιμές των facets που έχουμε επιλέξει για τα ερωτήματά μας. Θα μπορούσαμε να επεκτείνουμε το εύρος των ερωτημάτων μας αν οι τιμές των facets (values) είχαν τα δικά τους facets στα οποία θα μπορούσαμε να επεκταθούμε. Για παράδειγμα η Χώρας κατασκευής ενός προϊόντος μπορεί να έχει την τιμή «Ελλάδα» και η Ελλάδα να έχει τα δικά της facets όπως για παράδειγμα κατηγορίες προϊόντων που κατασκευάζονται εκεί. Αυτή η αυθαίρετη εναλλαγή μεταξύ αντικειμένων και facets όπως και η ανακάλυψη της κρυμμένης συσχέτισης μεταξύ των διαφόρων αντικειμένων της συλλογής μας, θα έδινε στα ερωτήματά μας πολύ μεγαλύτερη εκφραστικότητα. Ίσως η λύση στο πρόβλημα να

είναι η ενσωμάτωση χαρακτηριστικών σημασιολογικού ιστού στα ερωτήματά μας, αλλά αυτό είναι μία σημαντική διαφοροποίηση στο μοντέλο του faceted search που πρέπει να διερευνηθεί.

- Διαπιστώσαμε ότι το faceted search μπορεί να λειτουργεί σωστά σε συστήματα αναζήτησης στα οποία τα ερωτήματα διατυπώνονται όχι απλά με κείμενο αλλά και με άλλους τρόπους όπως μουσική, εικόνες κτλ. Σε όλα αυτά όμως η ύπαρξη μεταδεδομένων για την ταξινόμηση των δεδομένων είναι σημαντική. Πρέπει να διερευνηθεί περισσότερο η δυνατότητα αυτόματης άντλησης των μεταδεδομένων από τα δεδομένα μας σε όποια μορφή και αν βρίσκονται αυτά.

Όλα αυτά αποτελούν σημαντικά σημεία εξέλιξης και ήδη απασχολούν την ακαδημαϊκή ερευνητική κοινότητα. Προτείνουμε ότι θα πρέπει να μελετηθούν περισσότερο διότι θα μας απασχολήσουν στο άμεσο μέλλον στις νέες μορφές των συστημάτων αναζήτησης.

Βιβλιογραφία

Alsallakh, B., Miksch, S., & Rauber, A. (2014). Towards a Visualization of Multi-faceted Search Results. In Proceedings of the DL2014 Workshop on Knowledge Maps and Information Retrieval (KMIR), the ACM/IEEE Joint Conference on Digital Libraries.

Basu Roy, S., Wang, H., Das, G., Nambiar, U., & Mohania, M. (2008, October). Minimum-effort driven dynamic faceted search in structured databases. InProceedings of the 17th ACM conference on Information and knowledge management (pp. 13-22). ACM.

Ben-Yitzhak, O., Golbandi, N., Har'El, N., Lempel, R., Neumann, A., Ofek-Koifman, S., ... & Yogev, S. (2008, February). Beyond basic faceted search. InProceedings of the 2008 International Conference on Web Search and Data Mining (pp. 33-44). ACM.

Dakka, W., & Ipeirotis, P. G. (2008, April). Automatic extraction of useful facet hierarchies from text databases. In Data Engineering, 2008. ICDE 2008. IEEE 24th International Conference on (pp. 466-475). IEEE.

Dash, D., Rao, J., Megiddo, N., Ailamaki, A., & Lohman, G. (2008, October). Dynamic faceted search for discovery-driven analysis. In Proceedings of the 17th ACM conference on Information and knowledge management (pp. 3-12). ACM.

Elsweiler, D., Wilson, M. L., & Lunn, B. K. (2011). Understanding casual-leisure information behaviour. Library and Information Science, 211, 241.

Fafalios, P., Kitsos, I., Marketakis, Y., Baldassarre, C., Salampasis, M., & Tzitzikas, Y. (2012). Web searching with entity mining at query time. InMultidisciplinary Information Retrieval (pp. 73-88). Springer Berlin Heidelberg.

Fafalios, P., Papadakos, P., & Tzitzikas, Y. (2014). Enriching textual search results at query time using entity mining, linked data and link analysis.International Journal of Semantic Computing, 8(04), 515-544.

Ferré, S., & Hermann, A. (2012). Reconciling faceted search and query languages for the semantic web. International Journal of Metadata, Semantics and Ontologies, 7(1), 37-54.

Ferré, S.. (2014).Query-based Faceted Search: Expressive and Guided Information Access to the Semantic Web, 3rd MUMIA Training School

Gomadam, K., Ranabahu, A., Nagarajan, M., Sheth, A. P., & Verma, K. (2008, September). A faceted classification based approach to search and rank web apis. In Web Services, 2008. ICWS'08. IEEE International Conference on (pp. 177-184). IEEE.

Hatcher, E., Gospodnetic, O., & McCandless, M. (2004). Lucene in action.

Hearst, M. (2006, August). Design recommendations for hierarchical faceted search interfaces. In ACM SIGIR workshop on faceted search (pp. 1-5).

Hearst, M., Elliott, A., English, J., Sinha, R., Swearingen, K., & Yee, K. P. (2002). Finding the flow in web site search. Communications of the ACM, 45(9), 42-49.

Hughes, R. (2011). Mobile Google map queries to overtake those from desktop PCs. Clickroutes. Retrieved Decemper 2015 from <http://www.clickroutes.com/2011/05/ mobile-google-map-queries-to-overtake-those-from-desktop-pcs/>.

Järvelin, K., & Ingwersen, P. (2004). Information seeking research needs extension towards tasks and technology. Information Research, 10(1).

Joachims, T., Granka, L., Pan, B., Hembrooke, H., & Gay, G. (2005, August). Accurately interpreting clickthrough data as implicit feedback. In Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval (pp. 154-161). ACM.

Jones, K. S. (1973). Index term weighting. Information storage and retrieval,9(11), 619-633.

Marchionini, G. (2006). Exploratory search: from finding to understanding.Communications of the ACM, 49(4), 41-46.

Pirolli, P., & Card, S. (1999). Information foraging. Psychological review, 106(4), 643.

Pollitt, A. S., Smith, M. P., Treglown, M., & Braekevelt, P. (1996). View-Based Searching Systems--Progress Towards Effective Disintermediation.

Russell-Rose, T., & Tate, T. (2012). Designing the search experience: The information architecture of discovery. Newnes.

S. Ferré, A. Hermann, 2012. Reconciling faceted search and query languages for the Semantic Web. Int. J. Metadata, Semantics and Ontologies 7(1).

Sacco, G. M., & Tzitzikas, Y. (Eds.). (2009). Dynamic taxonomies and faceted search: theory, practice, and experience (Vol. 25). Springer Science & Business Media.

Shneiderman, B. (1994). Dynamic queries for visual information seeking.Software, IEEE, 11(6), 70-77.

Spencer, D. (2006). Four modes of seeking information and how to design for them. Boxes and Arrows (http://www.boxesandarrows.com/view/four_modes_of_seeking_information_and_how_to_design_for_them).

Spool, J. M., Perfetti, C., & Brittan, D. (2004). Designing for the scent of information. User Interface Engineering.

Tunkelang, D. (2009). Faceted search. Synthesis lectures on information concepts, retrieval, and services, 1(1), 1-80.

White, R. W. and R. A. Roth (2009). "Exploratory search: Beyond the query-response paradigm." In: Synthesis Lectures on Information Concepts, Retrieval, and Services (p. 2, 3, 69).

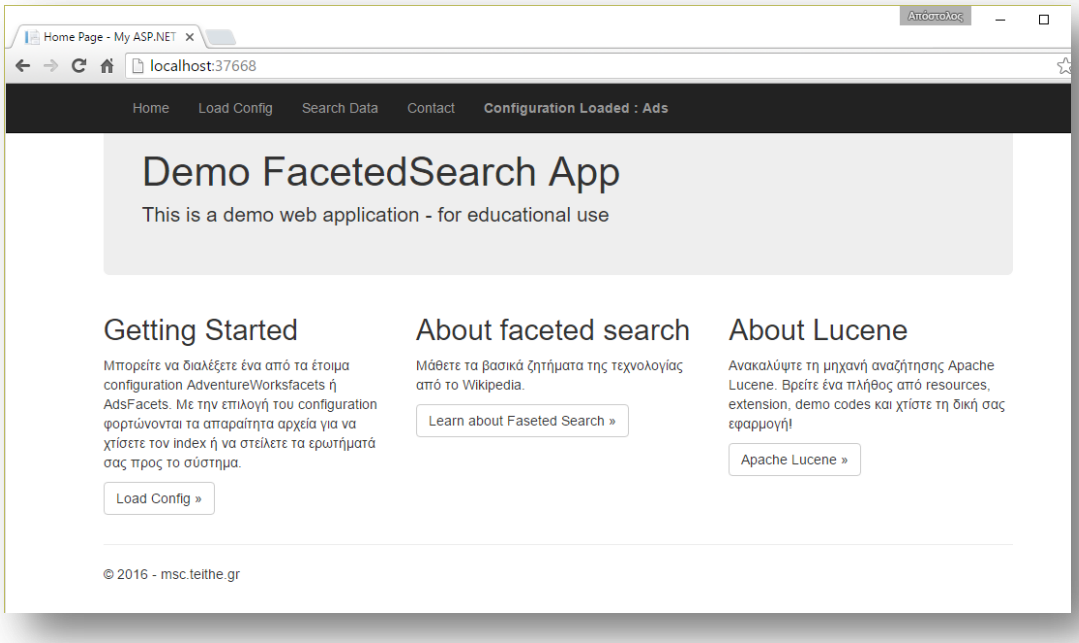
Wikipedia, Faceted search. https://en.wikipedia.org/wiki/Faceted_search [Accessed 12/ 2015].

9

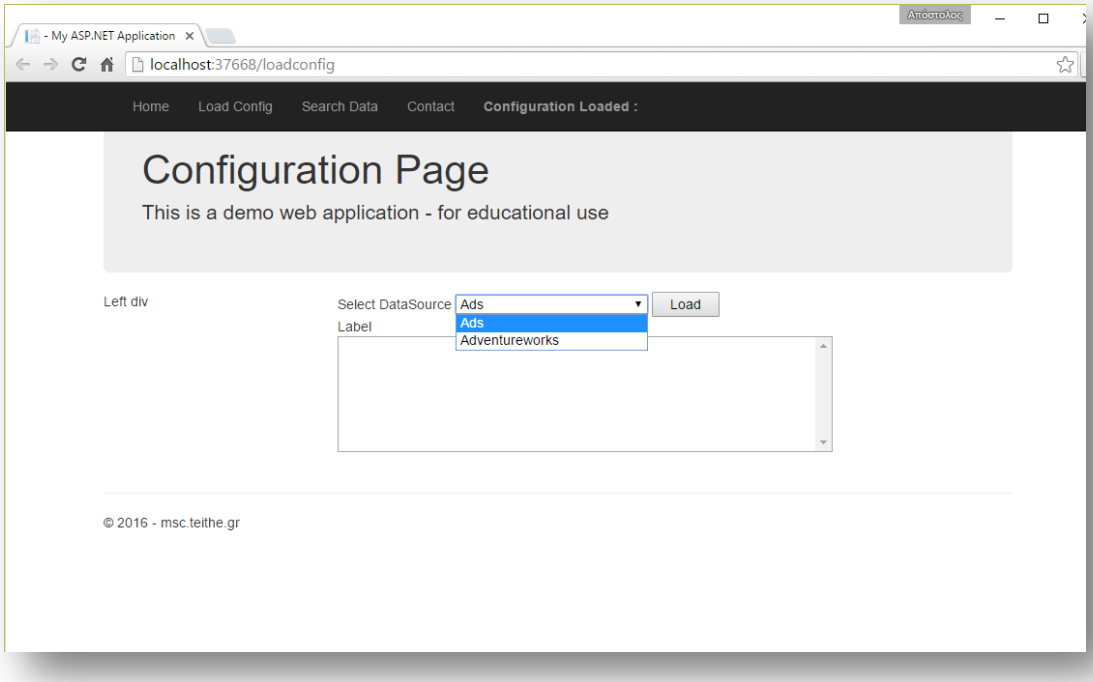
Παράρτημα

9.1 *Screenshots από την Demo εφαρμογή*

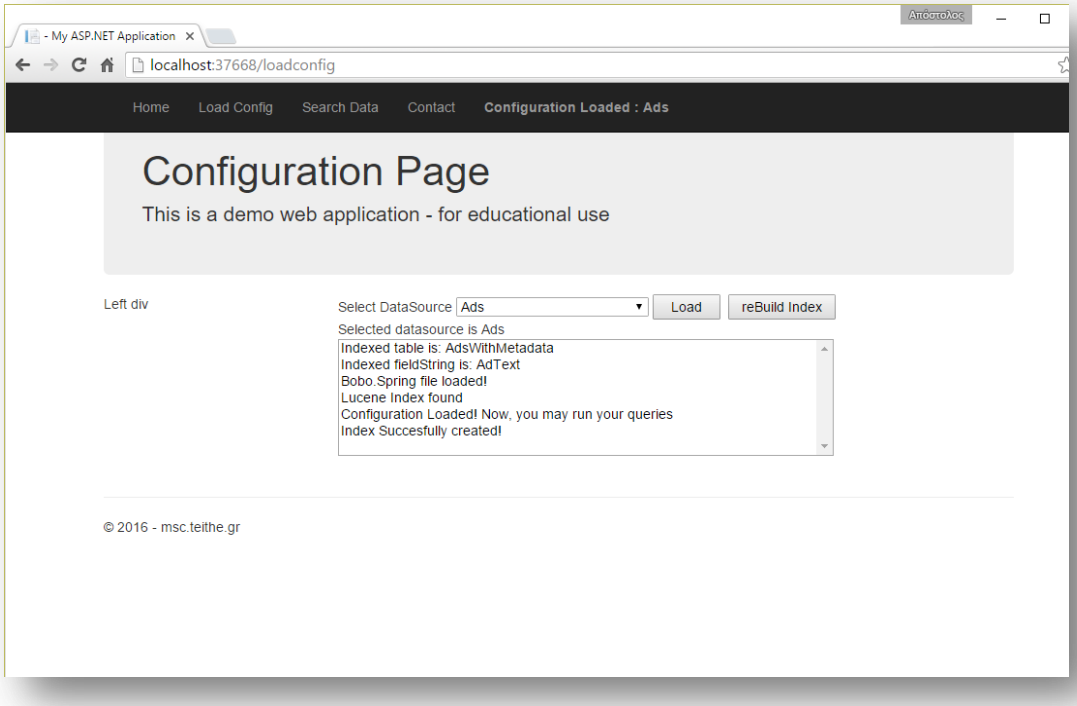
Η αρχική σελίδα



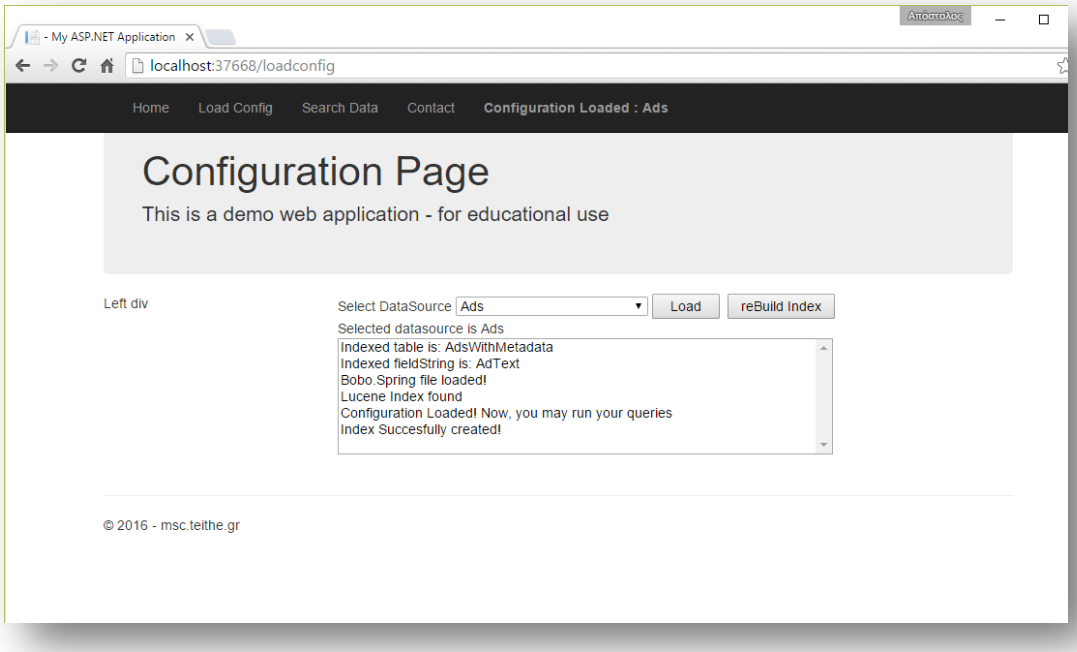
Η σελίδα των ρυθμίσεων: Επιλογή προφίλ



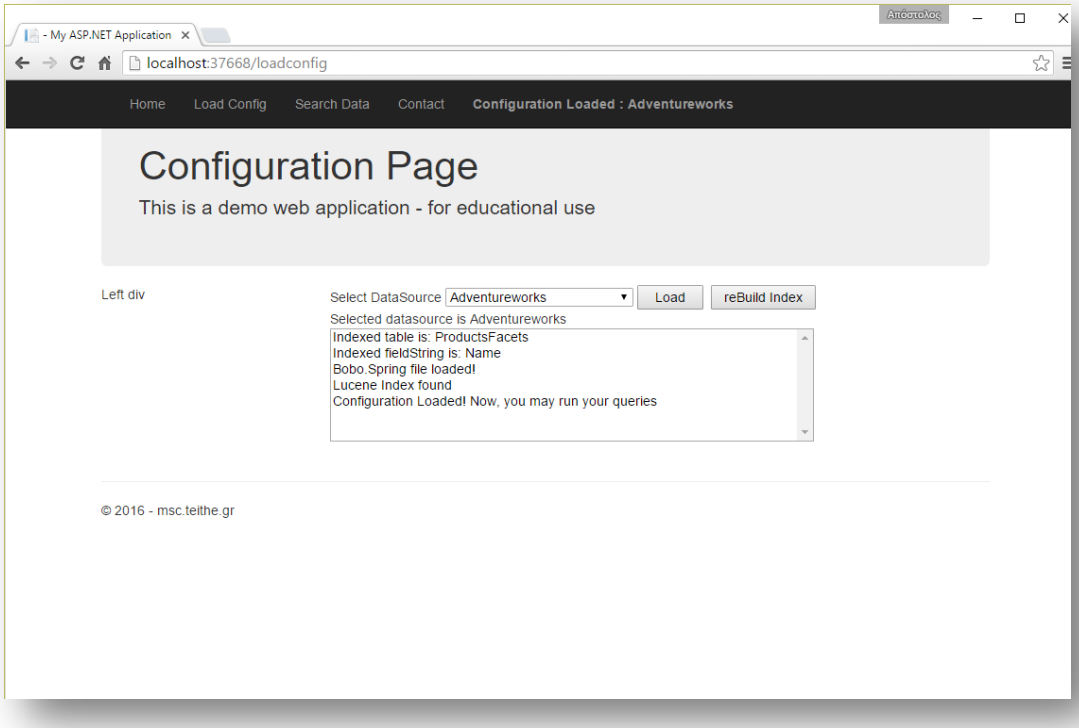
Σελίδα ρυθμίσεων: Φόρτωση προφίλ βάσης δεδομένων Αγγελιών



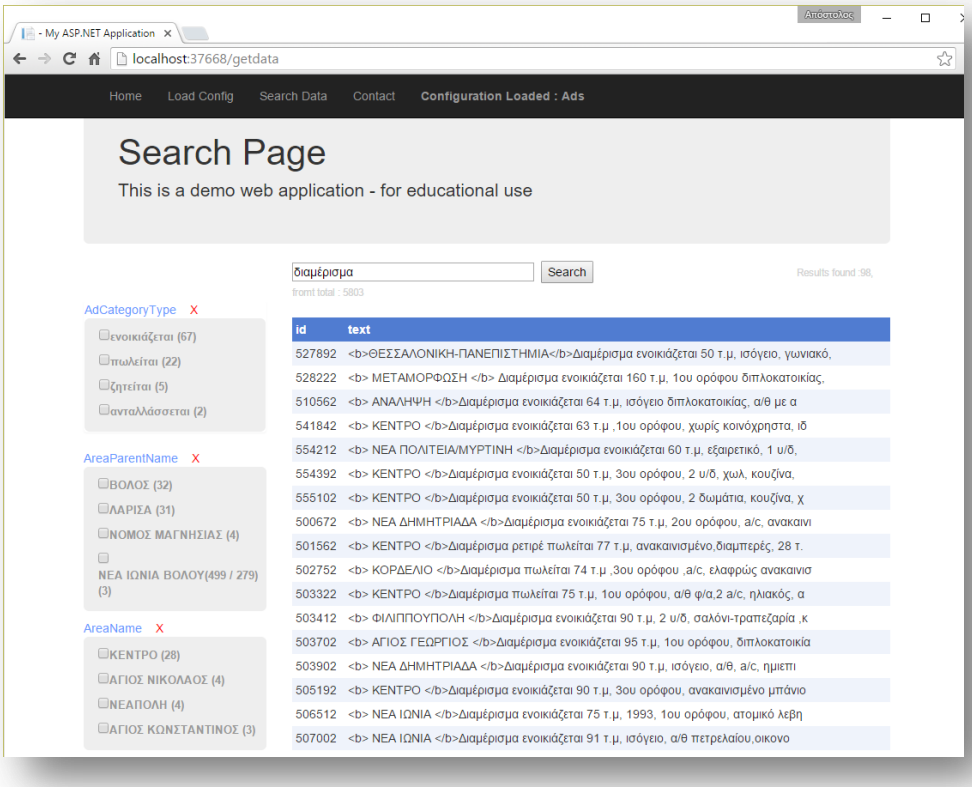
Σελίδα ρυθμίσεων: Δημιουργία Index του πίνακα με τις αγγελίες



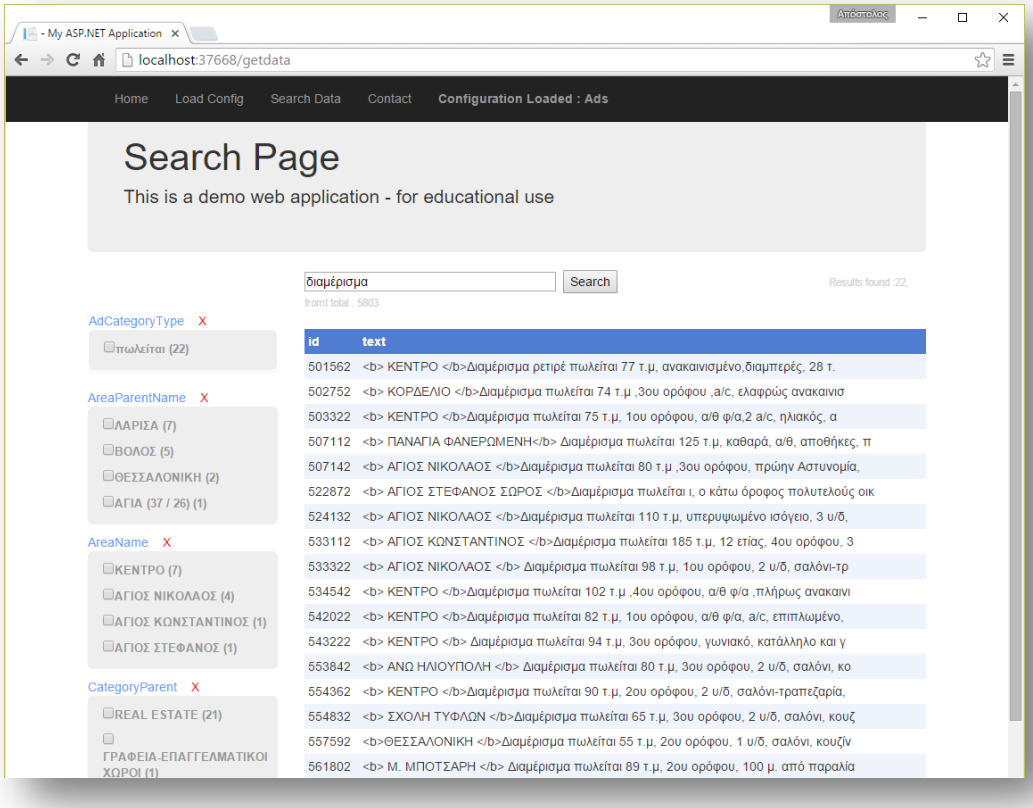
Φόρτωση προφίλ βάσης δεδομένων AdventureWorks



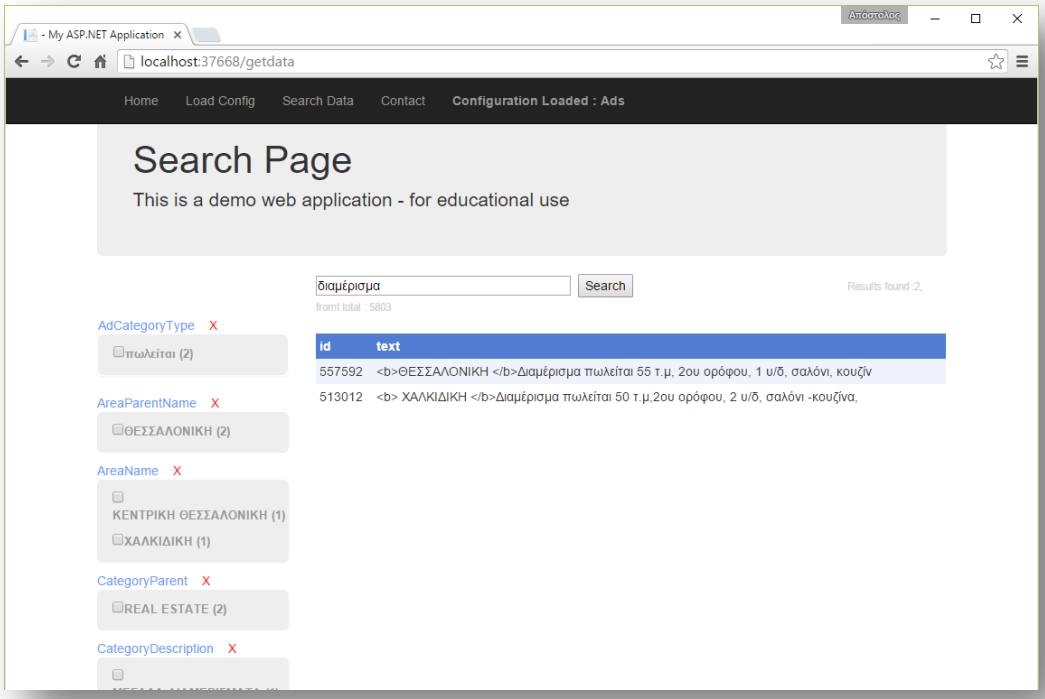
Σελίδα αποτελεσμάτων στον πίνακα Αγγελίες



Σελίδα αποτελεσμάτων: Εφαρμογή Facet πεδίου AdCategoryType



Επιλογή και δεύτερου facet AreaParantName



9.2 *Φύλλο συμμετοχής – ερωτηματολόγιο*

Αλεξάνδρειο Τεχνολογικό Εκπαιδευτικό Ίδρυμα

Μεταπτυχιακό Πρόγραμμα Σπουδών «**Ευφυείς Τεχνολογίες Διαδικτύου**»

Διπλωματική εργασία του «Απόστολος Μαβίδης»

Φύλλο συμμετοχής στην πειραματική διαδικασία

Πρώτο μέρος: Εκτέλεση ερωτημάτων στην demo εφαρμογή

Χρησιμοποιώντας την demo εφαρμογή αναζήτησης που σας έχουμε επιδείξει και επιλέγοντας το προφίλ της βάσης δεδομένων AdventureWorks2014, παρακαλούμε να εκτελέσετε τις παρακάτω ενέργειες αναζήτησης. Για κάθε ενέργεια μπορείτε να σταματήσετε όταν αισθάνεστε ότι εκπληρώσατε τον στόχο της:

αα	Περιγραφή ενέργειας	Διεκπεραίωση (ναι/όχι)
1	Εντοπίστε τα ακριβότερα προϊόντα	
2	Εντοπίστε τα φτηνότερα προϊόντα από τα "mountain bikes"	
3	Εντοπίστε τα προϊόντα με όνομα κάτι σαν "Mountain"και της κατηγορίας "Wheel"	
4	Εντοπίστε τα χρώματα από τα ακριβότερα ποδήλατα της κατηγορίας "touring Bikes"	

Δεύτερο μέρος: Συμπλήρωση Ερωτηματολογίου

Σημειώστε τις απαντήσεις σας στις παρακάτω ερωτήσεις, βαθμολογώντας με την κλίμακα από 1 έως 5 (1 =σχεδόν καθόλου, 2=λίγο, 3=μέτρια, 4=ικανοποιητικά, 5=πολύ)

αα	ερώτηση	Βαθμός (1 -5)
1	Πόσο εξοικειωμένος είστε με τη χρήση Η/Υ;	
2	Πόσο εξοικειωμένος είστε με τη χρήση μηχανών αναζήτησης (όπως Google, Bing)	
3	Πόσο εξοικειωμένος είστε με την αναζήτηση σε web site τύπου ecommerce.	
4	Πόσο δύσκολο ήταν να φέρετε εις πέρας το ερώτημα 1;	
5	Σε ποιο βαθμό πιστεύετε ότι καλύψατε το σύνολο του ερωτήματος 1;	
6	Πόσο δύσκολο ήταν να φέρετε εις πέρας το ερώτημα 2;	
7	Σε ποιο βαθμό πιστεύετε ότι καλύψατε το σύνολο του ερωτήματος 2;	
8	Πόσο δύσκολο ήταν να φέρετε εις πέρας το ερώτημα 3;	
9	Σε ποιο βαθμό πιστεύετε ότι καλύψατε το σύνολο του ερωτήματος 3;	
10	Πόσο δύσκολο ήταν να φέρετε εις πέρας το ερώτημα 4;	
11	Σε ποιο βαθμό πιστεύετε ότι καλύψατε το σύνολο του ερωτήματος 4;	
12	Πόσο δύσκολο ήταν να φέρετε εις πέρας το ερώτημα 5;	
13	Σε ποιο βαθμό πιστεύετε ότι καλύψατε το σύνολο του ερωτήματος 5;	
14	Από το χρόνο που απαιτήθηκε συνολικά για την εκπλήρωση των ασκήσεων, ποιο ήταν το ποσοστό του χρόνου μέσα στον οποίο χρειάστηκε να αναζητήσετε μέσα στη λίστα αποτελεσμάτων; 1. λιγότερο από 25%, 2. μεταξύ 25% και 50% 3. μεταξύ 50% και 75% 4. Πάνω από 75%	

Ευχαριστούμε για τη συμμετοχή σας στη διαδικασία.