ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ

ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

ΤΟΜΕΑΣ ΤΕΧΝΟΛΟΓΙΑΣ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΥΠΟΛΟΓΙΣΤΩΝ

Δυναμική Παρουσίαση και Διασύστημα Ενημέρωσης Χρηστών Φυλλομετρητών με Μετεωρολογικά Δεδομένα

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

της

ΟΥΡΑΝΙΑΣ ΛΕΛΙΟΥ

Επιβλέπων : Τίμος Σελλής
Καθηγητής Ε.Μ.Π.

Αθήνα, Ιούλιος 2008

ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ
ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
ΤΟΜΕΑΣ ΤΕΧΝΟΛΟΓΙΑΣ ΠΛΗΡΟΦΟΡΙΚΗΣ
ΚΑΙ ΥΠΟΛΟΓΙΣΤΩΝ

Δυναμική Παρουσίαση και Διασύστημα Ενημέρωσης Χρηστών Φυλλομετρητών με Μετεωρολογικά Δεδομένα

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

της

ΟΥΡΑΝΙΑΣ ΛΕΛΙΟΥ

Επιβλέπων : Τίμος Σελλής
Καθηγητής Ε.Μ.Π.

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την 10η Ιουλίου 2008.

.....	.....	.....
Τίμος Σελλής	Ιωάννης Βασιλείου	Νεκτάριος Κοζύρης
Καθηγητής Ε.Μ.Π.	Καθηγητής Ε.Μ.Π.	Αναπλ. Καθηγητής Ε.Μ.Π.

Αθήνα, Ιούλιος 2008

.....

ΛΕΛΙΟΥ ΟΥΡΑΝΙΑ

Διπλωματούχος Ηλεκτρολόγος Μηχανικός και Μηχανικός Υπολογιστών Ε.Μ.Π.

Περίληψη

Στην παρούσα διπλωματική εργασία περιγράφεται η υλοποίηση του συστήματος UranuS, ενός συστήματος τύπου mozilla extension για το φυλλομετρητή Firefox Mozilla, το οποίο ενημερώνει το χρήστη για μετεωρολογικά δεδομένα που αφορούν την περιοχή της επιλογής του. Ο χρήστης ορίζει ο ίδιος την πηγή της πληροφόρησης (συγκεκριμένη ιστοσελίδα) και επιλέγει τα χαρακτηριστικά του καιρού που τον ενδιαφέρουν και τον τρόπο παρουσίασης των δεδομένων. Οι πληροφορίες εξάγονται από την εκάστοτε ιστοσελίδα με τη βοήθεια κανονικών εκφράσεων, ενώ δίνεται η δυνατότητα στο χρήστη να ορίσει και να δοκιμάσει και τις δικές του κανονικές εκφράσεις. Τέλος, παρέχεται ένα ήδη έτοιμο προφίλ χρήστη το οποίο λαμβάνει μετεωρολογικές προβλέψεις από τον δικτυακό τόπο http://freemeteo.com, δίνοντας έτσι τη δυνατότητα για λήψη ακόμη περισσότερων πληροφοριών.

Λέξεις Κλειδιά: UranuS, Mozilla Firefox, extension, μετεωρολογικές προγνώσεις, εξαγωγή πληροφορίας, wrapper, κανονικές εκφράσεις.

Abstract

In the present diploma thesis I describe the implementation of the UranuS system, a Mozilla extension system for the web browser Firefox Mozilla which informs the user of weather data regarding the region of his choice. The user defines the source of the information (a specific website) himself, and chooses the weather characteristics that interest him and the way in which the data will be presented to him. The information is extracted from the website with the help of regular expressions, while it is possible for the user to define and use his own regular expressions. Finally, a default profile is provided, which receives weather forecasts from the website http://freemeteo.com thus giving the opportunity to receive even more detailed information.

Keywords: UranuS, Mozilla Firefox, extension, weather forecasts, information extraction, wrapper, regular expressions.

Πίνακας περιεχομένων

1 *Εισαγωγή*

1.1 Εξαγωγή πληροφορίας από διαδικτυακές πηγές και ανάπτυξη επεκτάσεων φυλλομετρητών

Τα τελευταία χρόνια έχει παρατηρηθεί μια συνεχής ανάπτυξη του Διαδικτύου και μια δραματική αύξηση τόσο των χρηστών του όσο και του όγκου των πληροφοριών που διοχετεύονται σε αυτό. Καθημερινά, εκατομμύρια χρηστών σε όλο τον κόσμο συνδέονται στο διαδίκτυο, αναζητώντας κάθε είδους πληροφορία από έναν τεράστιο αριθμό ετερόκλητων πηγών, το περιεχόμενο των οποίων μεταβάλλεται συνεχώς. Η αυξανόμενη αυτή ανάγκη για πληροφόρηση έχει οδηγήσει στην ανάπτυξη αυτοματοποιημένων συστημάτων τα οποία μπορούν να εντοπίσουν και να εξάγουν δεδομένα από διάφορες πηγές και να τα οργανώσουν σε δομημένες πληροφορίες, έτοιμες προς παρουσίαση στο χρήστη ή προς χρήση από άλλες εφαρμογές. Παράλληλα, οι απαιτήσεις για εφαρμογές που θα διευκολύνουν το χρήστη του διαδικτύου υλοποιώντας υπηρεσίες που θα είναι προσβάσιμες από το παράθυρο του περιηγητή του μόνο με το πάτημα ενός πλήκτρου η με μια ματιά στη γραμμή κατάστασης, έχουν οδηγήσει στην ανάπτυξη εκατοντάδων προγραμμάτων – επεκτάσεων των περιηγητών του Παγκόσμιου Ιστού.

Ωστόσο, πολύ λίγες είναι οι προσεγγίσεις που έχουν γίνει προς την κατεύθυνση του συνδυασμού αυτών των δύο λειτουργιών, δηλαδή της εξαγωγής της πληροφορίας ακριβώς που ενδιαφέρει το χρήστη, και μάλιστα από την πηγή που έχει καθορίσει ο ίδιος ανεξάρτητα με τον τρόπο που παρέχεται η πληροφορία από την πηγή, και της παρουσίασής της ακριβώς όπως έχει επιλέξει εκείνος, με τρόπο άμεσα προσβάσιμο ώστε να είναι συνεχής και άμεση η ενημέρωσή του. Μια εφαρμογή τέτοιου είδους θα πρέπει να είναι σε θέση επομένως να συνδυάσει αποτελεσματικά διάφορες σύγχρονες τεχνικές, εξυπηρετώντας καλύτερα το χρήστη του διαδικτύου και ικανοποιώντας με τον πιο ολοκληρωμένο τρόπο τη βασική του ανάγκη, την ανάγκη για ενημέρωση.

1.2 Αντικείμενο διπλωματικής

Ένας από τους τομείς που συγκεντρώνουν ιδιαίτερο ενδιαφέρον από τη μεριά των χρηστών του διαδικτύου όσον αφορά στην ενημέρωση, είναι αυτός των μετεωρολογικών προγνώσεων. Αντικείμενο της παρούσας διπλωματικής εργασίας είναι ο σχεδιασμός και η υλοποίηση του συστήματος UranuS, μιας επέκτασης για τον περιηγητή Mozilla Firefox, η οποία λαμβάνει ανά τακτά χρονικά διαστήματα μετεωρολογικά δεδομένα από μια πηγή στο διαδίκτυο που έχει επιλέξει ο χρήστης και τα παρουσιάζει στη γραμμή κατάστασης στο παράθυρο του φυλλομετρητή του, σύμφωνα με τον τρόπο που επιθυμεί εκείνος.

Πολλοί είναι οι χρήστες που επισκέπτονται μια τοποθεσία με μετεωρολογικές προγνώσεις προκειμένου να ενημερωθούν για τον καιρό σε κάποια πόλη στην οποία πρόκειται να προγραμματίσουν κάποιο ταξίδι, αρκετοί είναι όμως και αυτοί που επισκέπτονται συχνά (για παράδειγμα καθημερινά) τέτοιες τοποθεσίες, προκειμένου να είναι πλήρως ενημερωμένοι για τις συνθήκες του καιρού. Μάλιστα, στο μεγαλύτερο μέρος τους οι χρήστες της δεύτερης κατηγορίας προτιμούν να επισκέπτονται μία συγκεκριμένη τοποθεσία (ή έναν περιορισμένο αριθμό τοποθεσιών) την οποία εμπιστεύονται για το σκοπό αυτό, και να αναζητούν πληροφορίες που αφορούν μια συγκεκριμένη περιοχή, όπως για παράδειγμα τον τόπο κατοικίας τους. Στην περίπτωση αυτή, ένας χρήστης συνήθως προσθέτει τη συγκεκριμένη σελίδα με τις πληροφορίες που τον ενδιαφέρει στους σελιδοδείκτες (bookmarks) του περιηγητή του και να ανοίγει σε ένα νέο παράθυρο ή μια νέα καρτέλα κάθε φορά που θέλει να ενημερωθεί σχετικά. Εναλλακτικά, μπορεί να εγκαταστήσει κάποια από τις εφαρμογές που υπάρχουν διαθέσιμες στο διαδίκτυο και οι οποίες αναλαμβάνουν να ενημερώνονται για την πρόγνωση του καιρού για κάποια περιοχή και να του την παρουσιάζουν. Οι εφαρμογές αυτές ωστόσο, λαμβάνουν πληροφορίες από μία συγκεκριμένη και προκαθορισμένη πηγή, την οποία ο χρήστης μπορεί να κρίνει ως αναξιόπιστη. Στο σύστημα UranuS, ο χρήστης μπορεί να λαμβάνει μετεωρολογικές προγνώσεις από μια τοποθεσία που ο ίδιος εμπιστεύεται, χωρίς όμως να χρειάζεται να την επισκέπτεται κάθε τόσο και ενδεχομένως να πραγματοποιεί κάθε φορά από την αρχή την αναζήτηση για την τοποθεσία του ενδιαφέροντός του, ή την επιλογή του συστήματος μέτρησης και των επιθυμητών ημερών πρόγνωσης. Έχει ακόμη τη δυνατότητα να αποθηκεύσει διαφορετικά προφίλ χρήσης, έχοντας έτσι γρήγορη πρόσβαση σε προγνώσεις για διαφορετικές τοποθεσίες ή από διαφορετικές πηγές γλιτώνοντας και πάλι το χρόνο και τον κόπο της επαναλαμβανόμενης αναζήτησης στο διαδίκτυο.

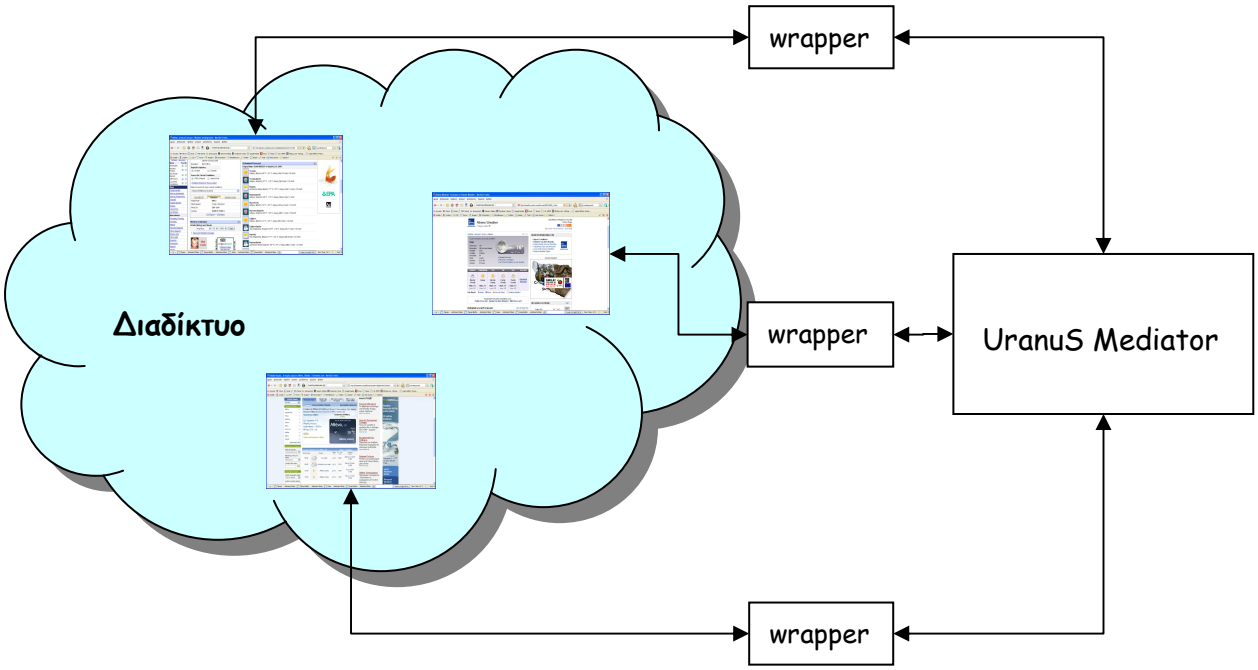
Όπως αναφέρθηκε και προηγουμένως, το σύστημα αποτελεί μια επέκταση (extension) για το φυλλομετρητή Mozilla Firefox. Μια επέκταση είναι ένα πρόγραμμα το οποίο προσθέτει νέα χαρακτηριστικά ή λειτουργίες στον περιηγητή, ή βελτιώνει τα ήδη υπάρχοντα. Στο σύστημα UranuS τα μετεωρολογικά δεδομένα εμφανίζονται στη γραμμή κατάστασης του περιηγητή και επομένως είναι μονίμως ορατά στο χρήστη. ενώ επιπλέον, ενημερώνονται ανά τακτά χρονικά διαστήματα ούτως ώστε

αυτός να μη χρειάζεται να ανοίγει ένα νέο παράθυρο, ή να εναλλάσσεται μεταξύ παραθύρων και να πατά το πλήκτρο της ανανέωσης της ιστοσελίδας, αναβάλλοντας πιθανώς μια εργασία με την οποία ασχολείται τη δεδομένη στιγμή.

Αξίζει να αναφερθεί ότι οι ιστοσελίδες με τις μετεωρολογικές προγνώσεις, χαρακτηρίζονται από μεγάλη ετερογένεια μεταξύ τους, ενώ και για κάθε σελίδα ξεχωριστά μπορεί κάθε τόσο να αλλάζει η δομή ή ο τρόπος παρουσίασης των περιεχομένων της. Έτσι, απαιτείται το σύστημα UranuS να μπορεί να τις αντιμετωπίζει με ενιαίο τρόπο και να εντοπίζει τα μετεωρολογικά δεδομένα ανεξάρτητα με τον τρόπο που είναι οργανωμένα αυτά. Το σύστημα επομένως λειτουργεί ουσιαστικά ως wrapper. Wrappers ονομάζονται τα προγράμματα τα οποία παρέχουν μια ενιαία διεπαφή για την προσπέλαση διαφορετικών πηγών από ένα σύστημα. Πιο συγκεκριμένα, ένας wrapper περιβάλλει (περιτυλίσσει) μια πηγή πληροφορίας, επιτρέποντας έτσι στο σύστημα να έχει πρόσβαση σε αυτή χωρίς να χρειάζεται να τροποποιηθεί ο βασικός του κώδικας. Με τον τρόπο αυτό είναι δυνατόν να προστίθενται νέες πηγές ή να τροποποιούνται οι ήδη υπάρχουσες πηγές, ενώ ο κώδικας-πυρήνας του συστήματος παραμένει ο ίδιος. Οι λειτουργίες που επιτελούνται από έναν wrapper στην περίπτωση που η πηγή είναι ένας web server, είναι:

1. σύνδεση με τον web server και ανάκτηση των κατάλληλων ιστοσελίδων μέσω του πρωτοκόλλου HTTP
2. εξαγωγή της επιθυμητής πληροφορίας από τις ιστοσελίδες.
3. ενοποίηση των αποτελεσμάτων με τα αποτελέσματα από άλλες πηγές.

Στο παρακάτω σχήμα φαίνεται η λειτουργία του συστήματος ως wrapper:



Εικόνα 1.2-1

Πρέπει να σημειωθεί ωστόσο ότι η εξαγωγή των επιθυμητών πληροφοριών από μια οποιαδήποτε σελίδα μετεωρολογικών προγνώσεων δεν είναι πάντα εύκολη, καθώς οι τρόποι αναπαράστασης των χαρακτηριστικών του καιρού ποικίλλουν από πηγή σε πηγή, ενώ πολλές φορές μπορεί σε μία σελίδα να βρίσκονται πολλά παρόμοια δεδομένα (για παράδειγμα οι θερμοκρασίες για τις σημαντικότερες πόλεις μιας χώρας ή οι θερμοκρασίες μιας ολόκληρης βδομάδας για μία συγκεκριμένη πόλη), ώστε να είναι δύσκολη η διάκριση μεταξύ τους και η επιλογή της σωστής πληροφορίας. Το σύστημα UranuS προσπαθεί να αντιμετωπίσει τα προβλήματα αυτά με τον καλύτερο δυνατό τρόπο, επιτρέποντας στο χρήστη να συμμετέχει στη διαδικασία εντοπισμού των επιθυμητών δεδομένων κατά τον ορισμό ενός νέου προφίλ χρήσης και να δηλώνει τις προτιμήσεις του, ώστε αυτές να αξιοποιηθούν στη συνέχεια για την εξαγωγή της σωστής πληροφορίας.

1.2.1 Συνεισφορά

Η συνεισφορά της διπλωματικής συνοψίζεται ως εξής:

1. Μελετήσαμε συστήματα που πραγματοποιούν εξαγωγή διαδικτυακής πληροφορίας.
2. Μελετήσαμε τον τρόπο δημιουργίας και λειτουργίας ενός Mozilla extension.
3. Αναπτύξαμε ένα μοντέλο οργάνωσης μετεωρολογικών δεδομένων, βάσει μελέτης που κάναμε σε πολλούς ιστότοπους παρουσίασης καιρού.
4. Αναλύσαμε, σχεδιάσαμε και υλοποιήσαμε επέκταση του Mozilla Firefox για τη λήψη και παρουσίαση στο χρήστη μετεωρολογικών δεδομένων.
5. Το σύστημα σχεδιάστηκε με βάση την αρχιτεκτονική mediator-wrapper. Έχει τη δυνατότητα εξαγωγής και επεξεργασίας πληροφοριών από πολλές πηγές δεδομένων, παρά το γεγονός ότι κάθε πηγή έχει το δικό της τρόπο οργάνωσης των δεδομένων αυτών.

1.3 Οργάνωση κειμένου

Στο Κεφάλαιο 2 παρουσιάζονται εργασίες σχετικές με το αντικείμενο της διπλωματικής. Το Κεφάλαιο 3 περιέχει το θεωρητικό υπόβαθρο που είναι απαραίτητο για την κατανόηση της παρούσας εργασίας. Στο Κεφάλαιο 4 γίνεται παρουσίαση της ανάλυσης απαιτήσεων του συστήματος. Τα θέματα σχεδίασης του συστήματος αναπτύσσονται στο κεφάλαιο 5, ενώ στο Κεφάλαιο 6 συζητούνται θέματα υλοποίησης. Στο Κεφάλαιο 7 παρουσιάζεται η διαδικασία ελέγχου της συμπεριφοράς του συστήματος. Τέλος, στο Κεφάλαιο 8 δίνονται τα συμπεράσματα της διπλωματικής εργασίας και στο Κεφάλαιο 9 παρουσιάζεται η βιβλιογραφία που χρησιμοποιήθηκε.

2 *Σχετικές Εργασίες*

Στο κεφάλαιο αυτό γίνεται μια αναφορά σε εργασίες οι οποίες θεωρούνται σχετικές με την παρούσα εργασία. Καταρχάς, γίνεται μια παρουσίαση άλλων επεκτάσεων του φυλλομετρητή Mozilla Firefox που σαν αντικείμενό τους έχουν την ανάκτηση και παρουσίαση στο χρήστη μετεωρολογικών προγνώσεων. Στη συνέχεια γίνεται μια εκτενής αναφορά στα συστήματα εξαγωγής πληροφορίας από το διαδίκτυο, και μια παρουσίαση και κατηγοριοποίηση των βασικότερων από αυτά. Η περιοχή αυτή σχετίζεται άμεσα με την παρούσα διπλωματική, καθώς η βασική λειτουργία του συστήματος UranuS είναι η εξαγωγή πληροφορίας από το διαδίκτυο και παρουσίασή της στο χρήστη. Ακολουθεί η παρουσίαση της τεχνολογίας των RSS feeds, η οποία θα μπορούσε να έχει χρησιμοποιηθεί εναλλακτικά από το σύστημα για την πραγματοποίηση της εξαγωγής πληροφορίας, χρησιμοποιώντας ως είσοδο τα RSS feeds των ιστοσελίδων-πηγών αντί για τις ίδιες τις ιστοσελίδες. Τέλος, παρουσιάζεται μια σχετικά νέα μεθοδολογία για πραγματοποίηση εξαγωγής πληροφορίας από διάφορες πηγές και συνδυασμού των αποτελεσμάτων, το web scraping.

2.1 Mozilla Firefox forecast extensions

Παρακάτω παρουσιάζονται μερικά προγράμματα-επεκτάσεις για τον φυλλομετρητή Mozilla Firefox, τα οποία σχετίζονται με μετεωρολογικά δεδομένα και προγνώσεις για τον καιρό. Η διαφορά των συστημάτων αυτών από το σύστημα UranuS είναι ότι η πηγή των δεδομένων τους είναι συγκεκριμένη για κάθε σύστημα, ενώ η λήψη των ενημερώσεων γίνεται συνήθως κατευθείαν τη βάση δεδομένων της πηγής ή από την ιστοσελίδα της, η οργάνωση της οποίας όμως είναι γνωστή από πριν.

2.1.1 Forecastfox – Forecastfox Enhanced

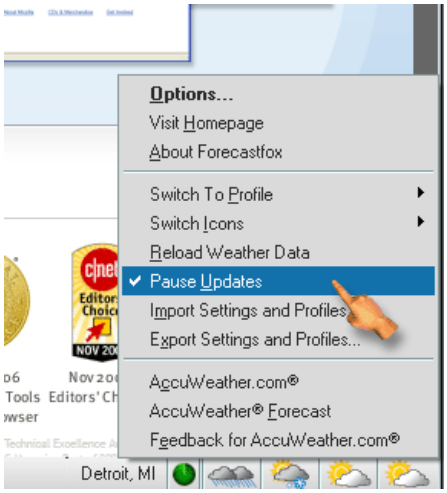
Το Forecastfox είναι μια από τις δημοφιλέστερες επεκτάσεις του Mozilla Firefox σχετικά με την πρόβλεψη του καιρού, η οποία έχει δημιουργηθεί από τους Jon Stritar και Richard Klein. Η επέκταση

αυτή εμφανίζει τις τρέχουσες καιρικές συνθήκες αλλά και μετεωρολογικές προβλέψεις για όλο τον κόσμο από τον ιστότοπο AccuWeather.com στην γραμμή κατάστασης του Firefox ή σε όποια από τις γραμμές εργαλείων επιθυμεί ο χρήστης. Ο χρήστης μπορεί να επιλέξει την περιοχή που τον ενδιαφέρει εισάγοντας τον ταχυδρομικό κώδικα, αν είναι περιοχή των Η.Π.Α, ή χρησιμοποιώντας το εργαλείο αναζήτησης κωδικών περιοχών που διαθέτει η επέκταση. Κάθε φορά που ανανεώνονται οι τρέχουσες πληροφορίες, εμφανίζεται ένα μικρό κυλιόμενο πλαίσιο με τα νέα δεδομένα, το οποίο χάνεται μετά την πάροδο μερικών δευτερολέπτων, ενώ οι διάφορες πληροφορίες εμφανίζονται σταθερά στη θέση που έχει ορίσει ο χρήστης. Επιλέγοντας με το ποντίκι τις πληροφορίες αυτές μάλιστα, ο χρήστης μπορεί να μεταβεί στην αντίστοιχη ιστοσελίδα της τοποθεσίας AccuWeather.com ώστε να έχει μια λεπτομερέστερη πρόβλεψη. Ο χρήστης του Forecastfox μπορεί να ρυθμίσει επίσης τον αριθμό των ημερών για τις οποίες θα έχει πρόβλεψη, το ποιες πληροφορίες θέλει να βλέπει το σύστημα μονάδων που θα χρησιμοποιείται και αν θα βλέπει μόνο τα δεδομένα ή και τις επιγραφές τους. Επίσης, έχει τη δυνατότητα να δημιουργήσει και να αποθηκεύσει διαφορετικά προφίλ, ώστε να αλλάζει εύκολα μεταξύ διαφορετικών τοποθεσιών ή επιλογών εμφάνισης.

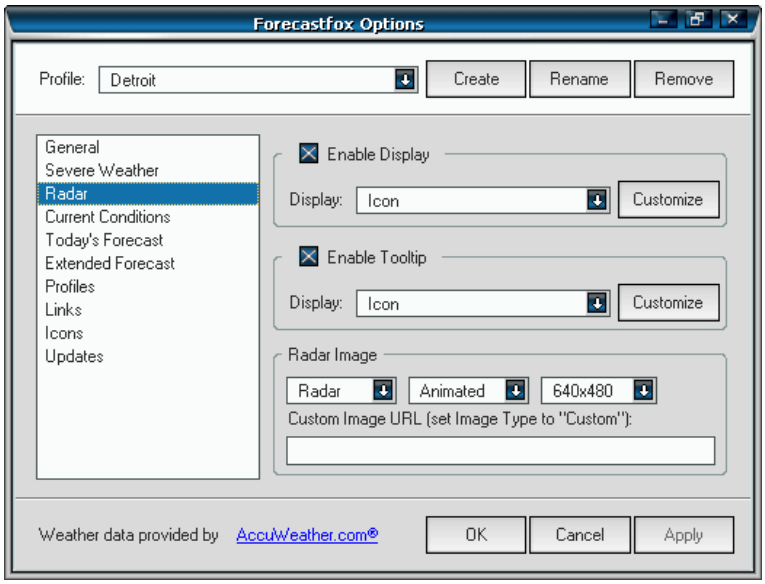
Εικόνα 2.1-1 Forecastfox screenshot 1

Εικόνα 2.1-2 Forecastfox screenshot 2

Το Forecastfox enhanced αποτελεί μια βελτιωμένη εκδοχή του Forecastfox, η οποία προσθέτει βελτιωμένες εικόνες ραντάρ για 12 τοποθεσίες στις ΗΠΑ και αρκετές ακόμη για διεθνείς τοποθεσίες, καθώς και τη δυνατότητα να παρέχει ο χρήστης το url της τοποθεσίας από την οποία θέλει να λαμβάνονται οι εικόνες του ραντάρ που χρησιμοποιούνται στην πρόβλεψη. Επιτρέπει επίσης την παύση ή επανεκκίνηση των αυτόματων ενημερώσεων και τη ρύθμιση της συχνότητας αυτών, όπως και την απενεργοποίηση του μετρητή προόδου κατά την ανανέωση των δεδομένων. Η επέκταση αυτή δημιουργήθηκε από τον Aaron Sarna. αλλά βασίζεται στην εκδοχή των Stritar και Klein.



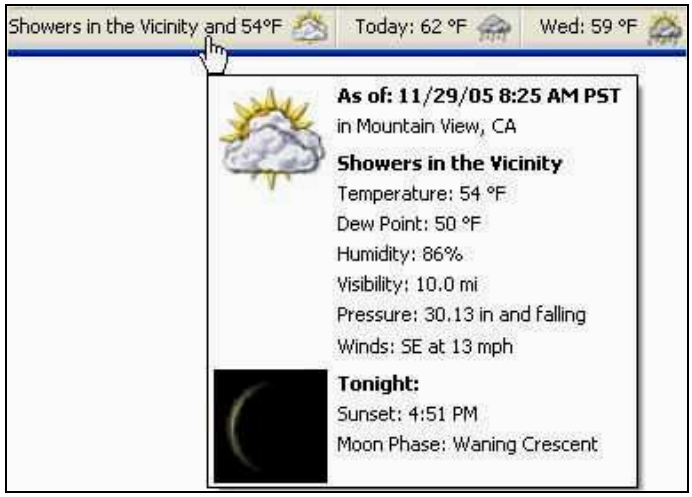
Εικόνα 2.1-3 Forecastfox Enhanced screenshot 1



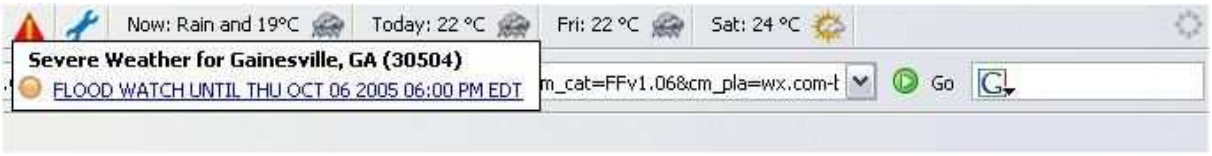
Εικόνα 2.1-4 Forecastfox Enhanced screenshot 2

2.1.2 1-ClickWeather

Το 1-ClickWeather είναι μια επέκταση παρόμοιων δυνατοτήτων με το Forecastfox, που αναπτύχθηκε από ομάδα της τοποθεσίας weather.com, η οποία αποτελεί και την πηγή των μετεωρολογικών δεδομένων και των προβλέψεων. Πέρα από τις προβλέψεις παρέχονται ακόμη προειδοποιήσεις για έκτακτα καιρικά φαινόμενα κωδικοποιημένες χρωματικά, χάρτες ραντάρ Doppler, δορυφορικοί χάρτες και βίντεο καιρικών συνθηκών κατ' απαίτηση. Ακόμη, παρέχεται η δυνατότητα μετάβασης με ένα κλικ σε ένα προκαθορισμένο από το χρήστη σύνολο συνδέσμων στην τοποθεσία weather.com, σχετικών με τοπικές προβλέψεις 36 ημερών ή 10ημερών, ή ακόμη και με προβλέψεις για τις ημέρες αγώνων baseball.



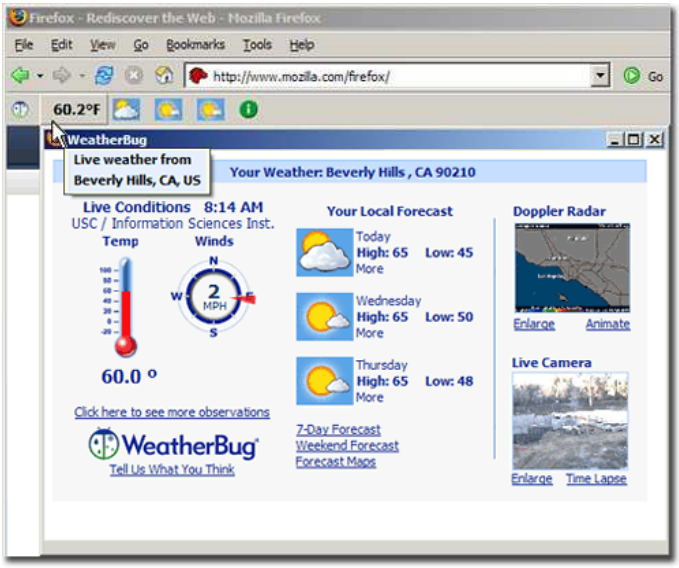
Εικόνα 2.1-5 1-ClickWeather screenshot 1



Εικόνα 2.1-6 1-ClickWeather screenshot 2

2.1.3 WeatherBug

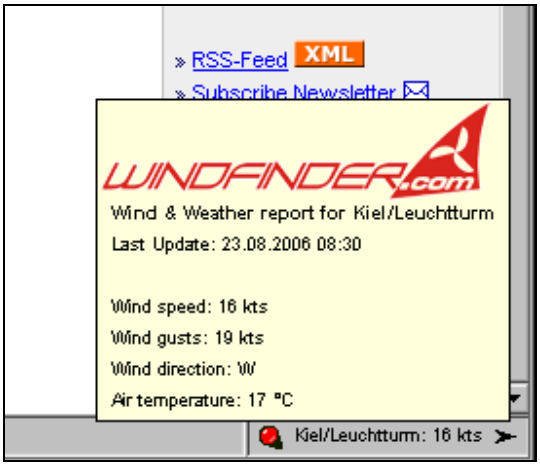
Μια ακόμη αντίστοιχη επέκταση είναι το WeatherBug Extension, το οποίο έχει αναπτυχθεί από τον Justin Dearing. Και εδώ, παρέχονται πληροφορίες για τις τρέχουσες καιρικές συνθήκες, έκτακτες μετεωρολογικές προειδοποιήσεις και εικόνες από ραντάρ, όλα από τον ιστότοπο WeatherBug.com.



Εικόνα 2.1-7 WeatherBug

2.1.4 WindFox

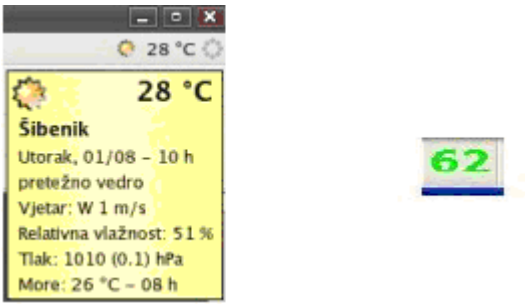
Το WindFox είναι μια επέκταση που απευθύνεται κατά βάση στους λάτρεις των θαλάσσιων σπορ (windsurfing, kitesurfing, sailing, surfing) και εμφανίζει στη γραμμή κατάστασης του Firefox την ταχύτητα και την κατεύθυνση του ανέμου καθώς και τη θερμοκρασία του αέρα για κάποια θαλάσσια τοποθεσία που επιλέγει ο χρήστης. Δημιουργοί της επέκτασης αυτής είναι οι Jonas Kaufmann και Johannes Bönniger, ενώ πηγή των δεδομένων είναι η τοποθεσία Windfinder.com.



Εικόνα 2.1-8 WindFox screenshot

2.1.5 HR Weather - Fast Weather – Yandex.Bar

Οι επεκτάσεις αυτές λιγότερο διαδεδομένες καθώς έχουν δημιουργηθεί για να παρέχουν πληροφορίες που εξυπηρετούν συγκεκριμένες περιοχές. Η πρώτη από αυτές, το HR Weather, εμφανίζει τις τρέχουσες καιρικές συνθήκες για κάποιες περιοχές της Κροατίας ενώ η δεύτερη, το Fast Weather, εμφανίζει στη γραμμή κατάστασης την θερμοκρασία για την περιοχή Boulder ή για το πάρκο Estes στο Colorado, όπως αυτή παρέχεται από τα NCAR Foothills Laboratory, NCAR Table Mesa Laboratory. Τέλος, το Yandex.Bar είναι μια προηγμένη γραμμή εργαλείων για τον Firefox που αναπτύχθηκε από τον Danil Ivanov και επιτρέπει στους χρήστες να πραγματοποιήσουν αναζήτηση στις υπηρεσίες της ρωσικής τοποθεσίας yandex.ru, και να λάβουν πληροφορίες και ειδοποιήσεις σχετικά με μη αναγνωσμένα μηνύματα (Yandex.Mail) και τροφοδοτήσεις (Yandex.Lenta), να λάβουν πληροφορίες σχετικά με το χρηματιστήριο (Yandex.Money) και τον καιρό (Yandex.Weather) και να προσθέσουν σελιδοδείκτες στο Yandex.Zakladki.



Εικόνα 2.1-9 HR Weather και Fast Weather screenshots

2.2 Συστήματα Εξαγωγής Διαδικτυακής Πληροφορίας (Web Information Extraction Systems)

2.2.1 Τι είναι η Εξαγωγή Διαδικτυακής Πληροφορίας

Η τεράστια ανάπτυξη και εξάπλωση του διαδικτύου, σε συνδυασμό με την ετερογένεια των πηγών πληροφορίας σε αυτό και την έλλειψη συγκεκριμένης δομής, έχει δημιουργήσει την ανάγκη για αυτοματοποιημένα εργαλεία που θα επιτρέπουν την εξαγωγή πληροφορίας από διάφορες πηγές και την οργάνωσή της σε δομές δεδομένων. Σε αντίθεση με την *Αναζήτηση Πληροφορίας* (ΑΠ), η οποία αφορά στην αναγνώριση των εγγράφων από μια συλλογή τα οποία σχετίζονται μεταξύ τους, η *Εξαγωγή Πληροφορίας* (ΕΠ) παράγει δομημένα δεδομένα, έτοιμα για μετα-επεξεργασία, κάτι ιδιαίτερα σημαντικό για αρκετές εφαρμογές εξόρυξης δεδομένων (web mining) και για εργαλεία αναζήτησης.

Τα προγράμματα τα οποία κάνουν την εξαγωγή της πληροφορίας ονομάζονται *extractors* ή πολλές φορές καλούνται, λανθασμένα, και *wrappers*. Αρχικά, ο wrapper είχε την έννοια μιας ψηφίδας σε ένα σύστημα ενοποίησης πληροφορίας, το οποίο στοχεύει στην παροχή μιας ενιαίας διεπαφής ερωτημάτων για την πρόσβαση σε πολλαπλές πηγές πληροφορίας. Γενικά, σε ένα σύστημα ενοποίησης πληροφορίας, ο wrapper είναι ένα πρόγραμμα που «περιβάλλει» μια πηγή πληροφορίας (για παράδειγμα έναν database server ή έναν web server), με τέτοιο τρόπο ώστε το σύστημα να μπορεί να προσπελάσει αυτή την πηγή χωρίς να χρειαστεί να αλλάξει τον εσωτερικό μηχανισμό απάντησης ερωτημάτων που διαθέτει. Στην περίπτωση που η πηγή είναι ένας web server, ο wrapper πρέπει να υποβάλλει ερωτήματα στον web server για να συλλέξει τις κατάλληλες σελίδες μέσω των πρωτοκόλλων HTTP, να κάνει εξαγωγή πληροφορίας για να εξάγει τα περιεχόμενα των HTML εγγράφων και τελικά να τα ενοποιήσει με τα αποτελέσματα από άλλες πηγές δεδομένων. Από τις τρεις αυτές λειτουργίες, η εξαγωγή πληροφορίας έχει τραβήξει περισσότερο την προσοχή, με αποτέλεσμα αρκετοί να χρησιμοποιούν τον όρο “wrappers” για να υποδηλώσουν τα προγράμματα εξαγωγής πληροφορίας.

Τα *συστήματα εξαγωγής πληροφορίας* (information extraction – IE) ή *συστήματα παραγωγής wrappers* (wrapper induction – WI) είναι εργαλεία λογισμικού τα οποία έχουν σχεδιαστεί για να παράγουν wrappers. Συνήθως ένας wrapper εφαρμόζει μια διαδικασία *ταιριάσματος μοτίβου* (pattern matching), η οποία βασίζεται σε ένα σύνολο κανόνων εξαγωγής. Η εργασία της προσαρμογής ενός τέτοιου συστήματος σε νέες απαιτήσεις ποικίλλει σε δυσκολία, ανάλογα με τον τύπο και την προέλευση του κειμένου και το συγκεκριμένο σενάριο. Για τη μεγιστοποίηση της επαναχρησιμοποίησης και την ελαχιστοποίηση του κόστους συντήρησης, η σχεδίαση ενός εκπαιδευόμενου συστήματος αποτελεί σημαντικό θέμα έρευνας στα πεδία της κατανόησης μηνυμάτων, της μηχανικής μάθησης, της εξόρυξης δεδομένων κλπ.

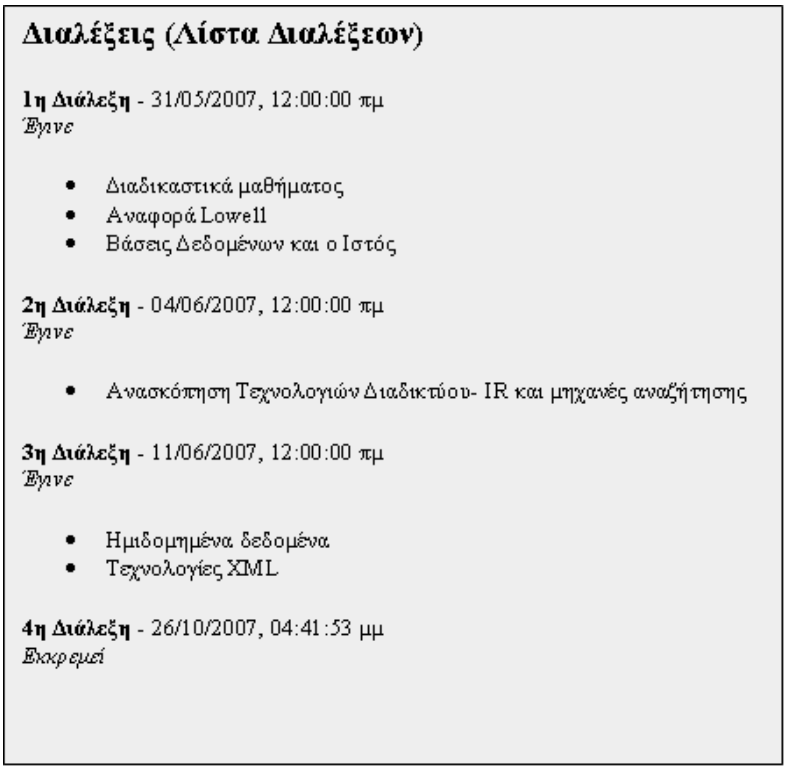
Η διαφορά των συστημάτων εξαγωγής διαδικτυακής πληροφορίας (συστήματα ΕΔΠ) από τα παραδοσιακά συστήματα εξαγωγής πληροφορίας είναι ότι, ενώ τα παραδοσιακά συστήματα στοχεύουν στην εξαγωγή δεδομένων από πλήρως αδόμητα, ελεύθερα κείμενα, γραμμένα σε φυσική γλώσσα, τα συστήματα ΕΔΠ επεξεργάζονται έγγραφα στο διαδίκτυο που είναι ημι-δομημένα και συνήθως παράγονται αυτόματα από προγράμματα εφαρμογών server-side. Ως εκ τούτου, η παραδοσιακή ΕΠ συνήθως εκμεταλλεύεται τεχνικές NLP (Natural Language Processing – Επεξεργασία φυσικής γλώσσας), όπως λεξιλόγια και γραμματικές, ενώ η ΕΔΠ συνήθως εφαρμόζει τεχνικές μηχανικής μάθησης και εξόρυξης μοτίβων για να εκμεταλλευτεί τα συντακτικά μοτίβα στη δομή διάταξης των εγγράφων , τα οποία είναι βασισμένα σε συγκεκριμένα πρότυπα (templates).

2.2.2 Βασικά χαρακτηριστικά των Συστημάτων ΕΔΠ

Γενικά, μια εργασία εξαγωγής πληροφορίας καθορίζεται από την είσοδο της και το αντικείμενο (στόχο) της εξαγωγής. Η είσοδος μπορεί να είναι μη-δομημένα, ημι-δομημένα ή και δομημένα έγγραφα. Μη-δομημένα θεωρούνται τα ελεύθερα κείμενα, τα οποία γράφονται σε φυσική γλώσσα, ενώ δομημένα μπορούν να θεωρηθούν τα XML αρχεία, για τα οποία υπάρχει το DTD ή το XML schema, το οποίο περιγράφει τα δεδομένα. Τέλος, ημι-δομημένα έγγραφα θεωρούνται οι περισσότερες από τις HTML σελίδες που συναντά κανείς στο διαδίκτυο, καθώς τα δεδομένα που περιέχουν περιγράφονται σε μεγάλο βαθμό μέσω των HTML tags. Γενικά, ημι-δομημένες είσοδοι θεωρούνται τα έγγραφα που έχουν μια αρκετά καλή δομή και μπορεί να διαθέτουν ή και να μη διαθέτουν μορφοποίηση HTML. Τα περισσότερα από αυτά πάντως, προέρχονται από το διαδίκτυο και μπορεί να είναι είτε δυναμικές ιστοσελίδες που παράγονται από δομημένες βάσεις δεδομένων με βάση κάποιο template (όπως για παράδειγμα οι σελίδες που περιγράφουν τα βιβλία στο Amazon, οι οποίες έχουν ακριβώς την ίδια διάταξη για τον συγγραφέα, τον τίτλο, την τιμή κλπ.) είτε ακόμη σελίδες γραμμένες στο χέρι (όπως οι λίστες δημοσιεύσεων στις προσωπικές ιστοσελίδες διαφόρων ερευνητών που, αν και έχουν δημιουργηθεί από διαφορετικά άτομα, διαθέτουν όλες έναν τίτλο και τις πηγές για κάθε καταχώρηση). Σε πολλές εργασίες εξαγωγής πληροφορίας, οι σελίδες εισόδου ανήκουν όλες στην ίδια «κλάση» σελίδων, υπάρχουν όμως και περιπτώσεις που η εξαγωγή γίνεται από αρκετές, διαφορετικές, σελίδες στο διαδίκτυο.

Το αντικείμενο της εξαγωγής μπορεί να είναι μια σχεσιακή k-πλειάδα (όπου k είναι ο αριθμός των γνωρισμάτων-στηλών σε μια εγγραφή) ή ένα σύνθετο αντικείμενο, με ιεραρχικά οργανωμένα δεδομένα. Σε μερικές περιπτώσεις, ένα γνώρισμα μπορεί να έχει 0 ή πολλαπλές τιμές σε μία εγγραφή, ενώ η διαδικασία της εξαγωγής πληροφορίας μπορεί να περιπλεχθεί ακόμη περισσότερο όταν υπάρχουν μεταθέσεις γνωρισμάτων ή τυπογραφικά λάθη στα έγγραφα εισόδου. Με βάση το αντικείμενό εξαγωγής, οι διάφορες εργασίες εξαγωγής πληροφορίας μπορεί να ταξινομηθούν σε εργασίες *επιπέδου εγγραφής*, *επιπέδου σελίδας* ή *επιπέδου ιστότοπου*. Οι wrappers που ανήκουν στην πρώτη κατηγορία ανακαλύπτουν τα όρια μια εγγραφής και στη συνέχεια τη διαιρούν στα γνωρίσματά της, αυτοί της δεύτερης κατηγορίας εξάγουν όλα τα δεδομένα που βρίσκονται ενσωματωμένα σε μία συγκεκριμένη ιστοσελίδα, ενώ τέλος οι wrappers επιπέδου ιστότοπου εισάγουν δεδομένα σε μια βάση δεδομένων από διάφορες σελίδες ενός συγκεκριμένου ιστότοπου. Οι ακαδημαϊκοί ερευνητές έχουν αφοσιωθεί περισσότερο στην ανάπτυξη wrappers επιπέδου εγγραφής και επιπέδου σελίδας , ενώ οι βιομηχανικοί ερευνητές ενδιαφέρονται περισσότερο για πλήρεις εφαρμογές που υποστηρίζουν εξαγωγή επιπέδου ιστότοπου. Για την περιγραφή του αντικειμένου εξαγωγής χρησιμοποιείται συνήθως ένα ιεραρχικό δέντρο, όπου τα φύλλα είναι βασικοί τύποι, ενώ οι εσωτερικοί κόμβοι είναι λίστες πλειάδων. Ένα αντικείμενο δεδομένων μπορεί να έχει απλή ή σύνθετη δομή. Στην πρώτη περίπτωση, υπάρχει μόνο ένας εσωτερικός κόμβος, η ρίζα, ενώ στη δεύτερη περίπτωση έχουμε περισσότερα από 2 επίπεδα.

Για να γίνουν περισσότερο κατανοητά τα παραπάνω, ακολουθεί ένα παράδειγμα μιας εργασίας εξαγωγής πληροφορίας από μια σελίδα στο διαδίκτυο. Στις Εικόνες 2.2-1 και 2.2-2 φαίνεται μια απλοποιημένη σελίδα με τη λίστα των διαλέξεων για ένα μάθημα σε μια πανεπιστημιακή σχολή, και ο κώδικάς της σε HTML αντίστοιχα. Συνεπώς, η είσοδος της εργασίας ΕΔΠ είναι η συγκεκριμένη HTML σελίδα, η οποία αποτελεί ημι-δομημένο έγγραφο. Παρόμοιες σελίδες, με την ίδια ακριβώς δομή, μπορεί να υπάρχουν για όλα τα μαθήματα της συγκεκριμένης σχολή, αποτελώντας μια κλάση σελίδων. Όπως φαίνεται στην εικόνα, στη σελίδα δίνονται για κάθε διάλεξη η ημερομηνία διεξαγωγής της, η «κατάστασή» της («Έγινε» ή «Εκκρεμεί») , καθώς και οι ενότητες που παραδόθηκαν κατά τη διάρκειά της. Έτσι, μπορούμε να θεωρήσουμε ότι το αντικείμενο της εξαγωγής θα είναι μια σχεσιακή 4-πλειάδα με τα γνωρίσματα «Αριθμός διάλεξης», «Ημ/νία» «Κατάσταση διάλεξης», «Παραδοθέντα κεφάλαια».



Εικόνα 2.2-1

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
  <title>
       Διαλέξεις (Λίστα Διαλέξεων)
  </title>
</head>
<body>
  <h2>
    Διαλέξεις (Λίστα Διαλέξεων)
  </h2>
  <b>1η Διάλεξη</b> - 31/05/2007, 12:00:00 πμ<br/>
  <i>Έγινε</i>
  <ul>
       <li>Διαδικαστικά μαθήματος</li>
       <li>Αναφορά Lowell</li>
       <li>Βάσεις Δεδομένων και ο Ιστός</li>
  </ul>
  <b>2η Διάλεξη</b> - 04/06/2007, 12:00:00 πμ<br/>
  <i>Έγινε</i>
  <ul>
       <li>Ανασκόπηση Τεχνολογιών Διαδικτύου- IR και μηχανές
αναζήτησης</li>
  </ul>
  <b>3η Διάλεξη</b> - 11/06/2007, 12:00:00 πμ<br/>
  <i>Έγινε</i>
  <ul>
       <li>Ημιδομημένα δεδομένα</li>
       <li>Τεχνολογίες XML</li>
  </ul>
  <b>4η Διάλεξη</b> - 26/10/2007,  04:41:53 μμ<br/>
  <i>Εκκρεμεί</i>
  <ul>
  </ul>
</body>
</html>
```

Εικόνα 2.2-2

Σε σχέση με τον τρόπο παρουσίασης ή το σύνολο των γνωρισμάτων που αποτελούν ένα αντικείμενο εξαγωγής τώρα, μπορεί να έχουμε τις εξής παραλλαγές:

- Ένα γνώρισμα μπορεί να έχει μηδέν ή περισσότερες τιμές. Στην πρώτη περίπτωση, το γνώρισμα καλείται *απόν γνώρισμα (missing attribute)*, ενώ στη δεύτερη καλείται *γνώρισμα πολλαπλών τιμών*. Για παράδειγμα, στην εργασία ΕΔΠ που περιγράφηκε προηγουμένως, το γνώρισμα «Παραδοθέντα κεφάλαια» μπορεί να λαμβάνει περισσότερες από μία τιμές, όπως συμβαίνει στην $1^{η}$ Διάλεξη, ή ακόμα και να μην έχει καμία τιμή, όπως στην $4^{η}$ Διάλεξη, η οποία δεν έχει πραγματοποιηθεί ακόμη.
- Το σύνολο των γνωρισμάτων (A_1, A_2, ..., A_k) μπορεί να έχει πολλαπλές διατάξεις, πράγμα που σημαίνει ότι ένα γνώρισμα A_i μπορεί να απαντάται σε διαφορετικές θέσεις σε διάφορα στιγμιότυπα ενός αντικειμένου δεδομένων. Αυτά τα γνωρίσματα καλούνται *γνωρίσματα πολλαπλής διάταξης*. Για παράδειγμα, σε ένα site για κινηματογραφικές ταινίες μπορεί η ημερομηνία κυκλοφορίας μιας ταινίας, να εμφανίζεται πριν τον τίτλο για ταινίες πριν το 1999 και μετά τον τίτλο για μεταγενέστερες ταινίες. Αντίστοιχα, στο παράδειγμα της κλάσης σελίδων της Εικόνας 2.2-2, θα μπορούσαμε στις σελίδες κάποιων από τα μαθήματα να συναντάμε την κατάσταση των διαλέξεων πριν από την ημερομηνία διεξαγωγής τους.
- Ένα γνώρισμα μπορεί να έχει ποικίλες μορφοποιήσεις σε διαφορετικά στιγμιότυπα ενός αντικειμένου δεδομένων. Αν η μορφοποίηση ενός γνωρίσματος δεν είναι συγκεκριμένη, μπορεί να χρειαστούμε κανόνες διάζευξης για να καλύψουμε όλες τις περιπτώσεις. Από την άλλη, διαφορετικά γνωρίσματα σε ένα αντικείμενο μπορεί να έχουν την ίδια μορφοποίηση, κυρίως στην αναπαράσταση σε πίνακα, όπου τα διάφορα γνωρίσματα παρουσιάζονται όλα μέσα στα tags <TD> και </TD>. Σε τέτοιες περιπτώσεις, το κλειδί για να διαχωριστούν τα διάφορα γνωρίσματα μεταξύ τους είναι η σειρά διάταξής τους. Αν όμως υπάρχουν απόντα γνωρίσματα ή έχουμε πολλαπλή διάταξη, οι κανόνες εξαγωγής για τα γνωρίσματα αυτά πρέπει να αναθεωρηθούν.
- Τα περισσότερα συστήματα εξαγωγής πληροφορίας χειρίζονται τα έγγραφα εισόδου σαν συμβολοσειρές από λεκτικές μονάδες (tokens) αντί για συμβολοσειρές από χαρακτήρες, καθώς έτσι είναι πιο εύκολο να τα επεξεργαστούν. Ανάλογα και με τη μέθοδο της λεκτικής ανάλυσης (tokenization), μερικές φορές ένα γνώρισμα δεν μπορεί να διασπαστεί σε ξεχωριστές λεκτικές μονάδες. Για παράδειγμα, στον κατάλογο των μαθημάτων σε ένα πανεπιστήμιο, ο κωδικός του τμήματος δεν έχει κάποιο διαχωριστικό χαρακτήρα για να το ξεχωρίζει από το νούμερο του μαθήματος (π.χ. COMP4016, GEOL2001). Ο βαθμός διάσπασης του αντικειμένου εξαγωγής επηρεάζει την επιλογή των σχημάτων λεκτικής ανάλυσης σε ένα σύστημα εξαγωγής πληροφορίας.

Όπως φαίνεται από τα παραπάνω, ο συνδυασμός διαφόρων εγγράφων εισόδου και διαφόρων αντικειμένων εξαγωγής δημιουργεί διαφορετικούς βαθμούς δυσκολίας στην εξαγωγή πληροφορίας.

Σε σχέση με τις τεχνικές που χρησιμοποιούνται από τα συστήματα εξαγωγής πληροφορίας, τα χαρακτηριστικά που μας ενδιαφέρουν είναι περισσότερο τα σχήματα λεκτικής ανάλυσης και κωδικοποίησης, ο τύπος των κανόνων εξαγωγής, ο αλγόριθμος εκμάθησης, κ.α. Για παράδειγμα, για τη λεκτική ανάλυση της εισόδου, υπάρχουν διάφοροι βαθμοί διάσπασης: στην *κωδικοποίηση επιπέδου επιγραφής*, κάθε HTML tag μεταφράζεται σε μία λεκτική μονάδα, και οτιδήποτε κείμενο βρίσκεται μεταξύ δύο tags μεταφράζεται σε μια ειδική λεκτική μονάδα, ενώ στην *κωδικοποίηση επιπέδου λέξης* κάθε λέξη στο έγγραφο θεωρείται λεκτική μονάδα. Οι κανόνες εξαγωγής μπορεί να παράγονται από γενίκευση από-πάνω-προς-τα-κάτω ή από-κάτω-προς-τα-πάνω, από εξόρυξη μοτίβων ή από λογικό προγραμματισμό, ενώ ο τύπος τους μπορεί να εκφράζεται μέσω κανονικών γραμματικών ή λογικών κανόνων.

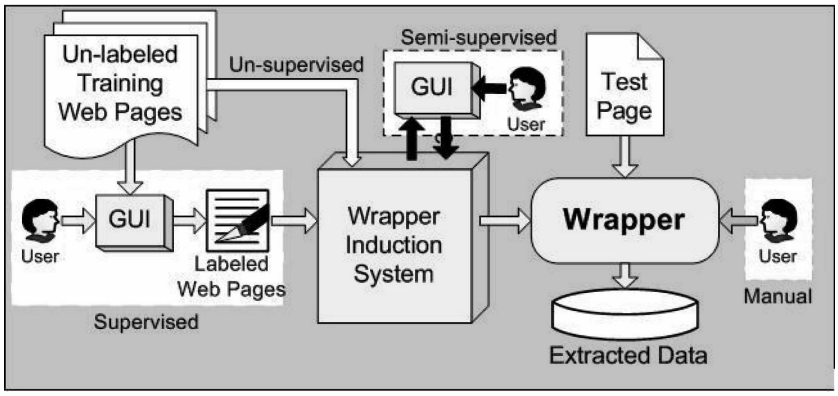
Τέλος, ένα άλλο χαρακτηριστικό που μας ενδιαφέρει για τη μελέτη και σύγκριση των διαφόρων wrappers είναι ο βαθμός αυτοματοποίησής τους. Γενικά, υπάρχουν διάφορες φάσεις από τις οποίες πρέπει να περάσει ένας wrapper: η συλλογή σελίδων για εκπαίδευση του προγράμματος, η τοποθέτηση ετικετών στα παραδείγματα εκπαίδευσης (labeling), η γενίκευση των κανόνων εξαγωγής, η εξαγωγή των κατάλληλων δεδομένων και η παρουσίαση του αποτελέσματος στην κατάλληλη μορφή. Η φάση του labeling γενικά καθορίζει την έξοδο της εργασίας εξαγωγής δεδομένων και απαιτεί τη συμμετοχή των χρηστών. Από την άλλη, μερικά συστήματα WI δεν απαιτούν να γίνει η ενέργεια αυτή στα παραδείγματα πριν το στάδιο της εκμάθησης, αλλά αντιθέτως μπορεί να γίνει η τοποθέτηση ετικετών ή σχολίων μετά την παραγωγή των κανόνων εξαγωγής (με ή χωρίς τη συμμετοχή του χρήστη). Έτσι σε μερικά συστήματα απαιτείται μεγαλύτερη εμπειρία από το χρήστη για την τοποθέτηση των ετικετών στα δεδομένα και την παραγωγή των κανόνων εξαγωγής, ενώ σε άλλα ο χρήστης απλά περιμένει το σύστημα να κάνει την εξαγωγή των δεδομένων. Φυσικά, όσο πιο αυτοματοποιημένο είναι ένα σύστημα, τόσο λιγότερο εφαρμόσιμη είναι η συγκεκριμένη μέθοδος σε κάποιο άλλο πεδίο εργασίας, ενώ μερικές φορές υπάρχει περιορισμός ακόμα και στον αριθμό των σελίδων εισόδου.

2.2.3 Επισκόπηση των βασικότερων συστημάτων WI

Τα παλαιότερα συστήματα WI έχουν σχεδιαστεί έτσι ώστε να διευκολύνουν τους προγραμματιστές στη συγγραφή κανόνων εξαγωγής. Αντίθετα, τα πιο πρόσφατα συστήματα χρησιμοποιούν μηχανική μάθηση για την αυτόματη παραγωγή κανόνων. Έτσι, πλέον η συμμετοχή του χρήστη αφορά στην τοποθέτηση ετικετών στα δεδομένα που έχουν εξαχθεί. Τελευταία μάλιστα, όλο και περισσότερες προσπάθειες γίνονται για να περιοριστεί η διαδικασία του labeling και να δημιουργούνται συστήματα με παραδείγματα για εκπαίδευση χωρίς ετικέτες. Με βάση τις τάσεις αυτές, μπορούμε να διακρίνουμε τα συστήματα WI σε 4 κατηγορίες :

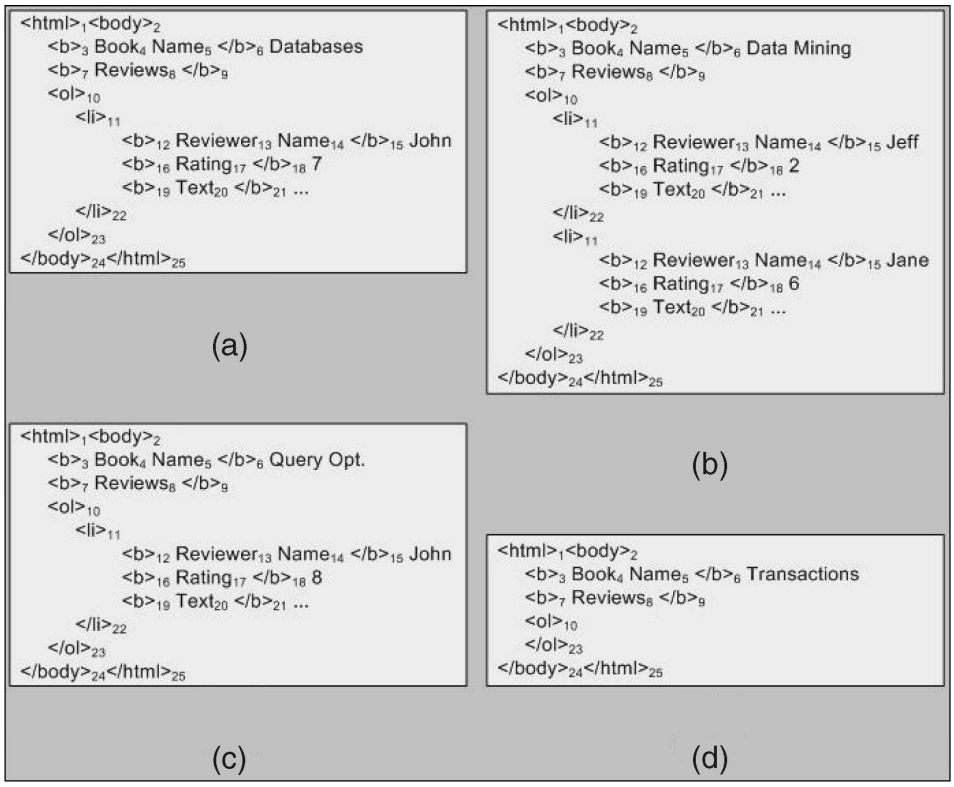
- Συστήματα γραμμένα στο χέρι (χειρωνακτικά συστήματα). Στα συστήματα αυτά ο χρήστης – προγραμματιστής γράφει ο ίδιος τους wrappers για κάθε διαδικτυακή τοποθεσία, με χρήση κάποιας γλώσσας προγραμματισμού.
- Επιβλεπόμενα συστήματα. Σε αυτά τα συστήματα ο χρήστης τοποθετεί ετικέτες σε ένα σύνολο σελίδων με παραδείγματα δεδομένων προς εξαγωγή και στη συνέχεια οι σελίδες αυτές δίνονται ως είσοδος στο σύστημα, το οποίο παράγει στην έξοδο τον κατάλληλο wrapper.
- Ημι-επιβλεπόμενα συστήματα. Τα συστήματα της κατηγορίας αυτής δέχονται ως είσοδο ένα σύνολο πιο γενικών παραδειγμάτων (σελίδες χωρίς ετικέτες), πραγματοποιούν την εξαγωγή της πληροφορίας και στη συνέχεια ο χρήστης αναλαμβάνει να επιλέξει τα επιθυμητά μοτίβα και τα αντικείμενα εξαγωγής.
- Μη-επιβλεπόμενα συστήματα. Εδώ ο χρήστης δεν αλληλεπιδρά καθόλου με το σύστημα. Στην είσοδο δίνονται σελίδες χωρίς καθόλου ετικέτες και το σύστημα αναλαμβάνει από μόνο του να επιλέξει τα αντικείμενα της εξαγωγής.

Τα τέσσερα αυτά είδη συστημάτων, καθώς και ο τρόπος με τον οποίο αλληλεπιδρά ο χρήστης, φαίνονται στην παρακάτω εικόνα :



Εικόνα 2.2-3

Στη συνέχεια, παρουσιάζονται σύμφωνα με αυτή την κατηγοριοποίηση οι πιο σημαντικοί και πιο σύγχρονοι wrappers. Για την καλύτερη κατανόηση του τρόπου λειτουργίας τους, δίνεται μια εργασία εξαγωγής πληροφορίας (Εικόνα 2.2-4) και για κάθε προσέγγιση περιγράφεται ο wrapper που παράγεται για την εξαγωγή των επιθυμητών πληροφοριών. Η εργασία ΕΔΠ αφορά σε 4 σελίδες που περιέχουν η καθεμία κριτικές για ένα βιβλίο. Οι πληροφορίες προς εξαγωγή είναι ο τίτλος του βιβλίου και οι αντίστοιχες κριτικές, οι οποίες περιλαμβάνουν το όνομα του βαθμολογητή, τη βαθμολογία και τα σχόλια.



Εικόνα 2.2-4

Όπως φαίνεται και στην εικόνα, οι 4 σελίδες ακολουθούν όλες το ίδιο μοντέλο, παρουσιάζουν όμως παραλλαγές όπως πολλαπλές κριτικές ή απουσία κριτικών, δίνοντάς έτσι την ευκαιρία να παρουσιαστούν οι δυνατότητες του κάθε συστήματος ως προς αυτές τις ιδιομορφίες.

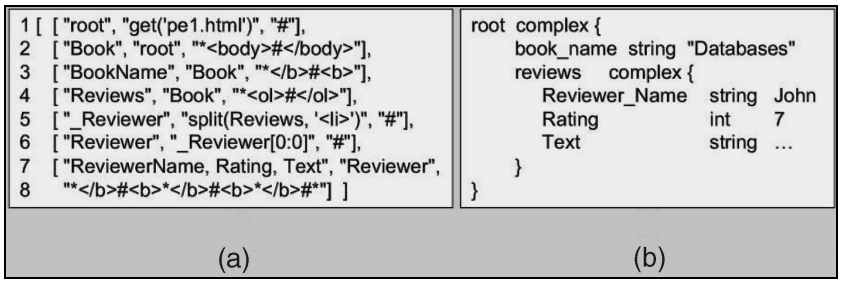
2.2.3.1 Χειρωνακτικά συστήματα WI

Σε αυτού του είδους τα συστήματα, οι χρήστες γράφουν οι ίδιοι έναν wrapper για κάθε ιστότοπο, χρησιμοποιώντας γενικές γλώσσες προγραμματισμού όπως η Perl, ή και ειδικά σχεδιασμένες γλώσσες. Αυτού του είδους τα εργαλεία απαιτούν από το χρήστη να έχει σημαντική εμπειρία πάνω στους υπολογιστές και τον προγραμματισμό, και γι' αυτό το λόγο είναι ακριβά. Τέτοια συστήματα είναι τα : TSIMMIS [HHM97], Minerva [CM98], WebOQL [AM98], W4F [SA01] και το σύστημα ΕΔΠ του συστήματος «Νέστωρ» [KVSP01].

2.2.3.1.1 TSIMMIS

Το TSIMMIS αποτελεί μια από τις πρώτες προσεγγίσεις οι οποίες θέτουν ένα πλαίσιο στο χειρονακτικό «χτίσιμο» των Web wrappers [HHM97]. Το βασικό συστατικό εδώ είναι ένας wrapper που δέχεται ως είσοδο ένα προσδιοριστικό αρχείο όπου δηλώνονται σαφώς (από μια ακολουθία εντολών που δίνονται από τους προγραμματιστές) το πού βρίσκονται τα δεδομένα που ενδιαφέρουν μέσα στις σελίδες, και το

πώς πρέπει να οργανωθούν τα δεδομένα σε αντικείμενα. Για παράδειγμα, η Εικόνα 2.2-5a δείχνει το προσδιοριστικό αρχείο για την εργασία ΕΠ της Εικόνας 2.2-4. Κάθε εντολή έχει τη μορφή [*μεταβλητές, πηγή, μοτίβο*], όπου η *πηγή* καθορίζει το κείμενο εισόδου το οποίο πρέπει να ληφθεί υπόψη, το *μοτίβο* καθορίζει πώς θα βρεθεί το κείμενο που ενδιαφέρει μέσα στο πηγαίο κείμενο και οι *μεταβλητές* είναι μια λίστα μεταβλητών όπου κρατούνται τα αποτελέσματα της εξαγωγής. Το ειδικό σύμβολο "*" σε ένα μοτίβο σημαίνει απόρριψη και το "#" σημαίνει αποθήκευση στις μεταβλητές. Έτσι, στο παράδειγμά μας, η 1η γραμμή του προσδιοριστικού αρχείου ορίζει την αποθήκευση ολόκληρου του HTML αρχείου της σελίδας pe1 στη μεταβλητή root και στη συνέχεια, στη 2η γραμμή γίνεται αποθήκευση μόνο του τμήματος του σώματος της σελίδας στη μεταβλητή Book. Έτσι προχωρά το πρόγραμμα σταδιακά. Το TSIMMIS παρέχει δύο σημαντικούς τελεστές: τον τελεστή *split* και τον τελεστή *case*. Ο πρώτος χρησιμοποιείται για να διαιρέσει την λίστα στοιχείων της εισόδου σε μεμονωμένα στοιχεία, όπως συμβαίνει στη γραμμή 5 του παραδείγματος, όπου η μεταβλητή Reviews «σπάει» σε κομμάτια που αντιστοιχούν στα στοιχεία της λίστας με τις κριτικές και τα οποία αποθηκεύονται στον πίνακα Reviewer, για να χρησιμοποιηθεί στη συνέχεια το πρώτο στοιχείο του πίνακα αυτού στην επόμενη γραμμή. Ο δεύτερος τελεστής (τελεστής case) επιτρέπει το χειρισμό των ανωμαλιών στη δομή των σελίδων εισόδου. Το TSIMMIS τελικά εξάγει τα δεδομένα στο Object Exchange Model (Εικόνα 2.2-5b), όπου περιέχονται τα επιθυμητά δεδομένα μαζί με πληροφορία σχετικά με τη δομή και τα περιεχόμενα του αποτελέσματος.



Εικόνα 2.2-5

2.2.3.1.2 Minerva

Το Minerva επιχειρεί να συνδυάσει τα πλεονεκτήματα μιας δηλωτικής προσέγγισης, βασισμένης σε μια γραμματική με την ευελιξία που προσφέρει ο διαδικαστικός προγραμματισμός στο χειρισμό των ετερογενειών και τον εξαιρέσεων [CM98]. Αυτό επιτυγχάνεται ενσωματώνοντας ένα μηχανισμό ρητών χειρισμών εξαιρέσεων σε μια κανονική γραμματική. Οι διαδικασίες χειρισμού εξαιρέσεων γράφονται στο Minerva σε μια ειδική γλώσσα που καλείται Editor. Η γραμματική που χρησιμοποιείται καθορίζεται με ένα συμβολισμό τύπου EBNF, όπου ορίζεται ένα σύνολο παραγωγών, και κάθε κανόνας παραγωγής καθορίζει τη δομή ενός μη τερματικού συμβόλου (το οποίο ακολουθεί πάντα το σύμβολο "$") της γραμματικής. Για παράδειγμα, στην Εικόνα 2.2-6 φαίνεται το σύνολο παραγωγών που μπορεί να χρησιμοποιηθεί για την εξαγωγή (αλλά επίσης και για την εισαγωγή σε βάση δεδομένων) των επιθυμητών γνωρισμάτων για τη δοσμένη εργασία ΕΠ (Εικόνα 2.2-4). Όπως συνηθίζεται στο

συμβολισμό EBNF, η έκφραση [p] υποδηλώνει ένα προαιρετικό μοτίβο p, ενώ η έκφραση (p)* υποδηλώνει ότι το p μπορεί να επαναλαμβάνεται 0 ή περισσότερες φορές (βλ. γραμμή 2). Οι μη τερματικές παραγωγές $bName, $rName, $rate και $text δίνονται στις γραμμές 3-6, αμέσως μετά τη χρήση τους στον ορισμό του $Book. Έτσι, για παράδειγμα, το όνομα του βιβλίου καταλαβαίνουμε ότι βρίσκεται μεταξύ των "Book Name" και "", όπως δείχνει το μοτίβο "*(?)" της γραμμής 3, το οποίο επιτυγχάνει ταίριασμα με κάθε συμβολοακολουθία πριν το tag . Τέλος, η τελευταία παραγωγή της εικόνας (γραμμές 7-12) ορίζει ένα ειδικό μη τερματικό σύμβολο $TP (Tuple Production), το οποίο χρησιμοποιείται για την εισαγωγή μιας πλειάδας στη βάση δεδομένων μετά από την συντακτική ανάλυση κάθε βιβλίου. Για κάθε κανόνα παραγωγής, είναι δυνατόν να εισάγουμε έναν χειριστή εξαιρέσεων (exception handler) που θα περιλαμβάνει το κομμάτι του κώδικα σε Editor που μπορεί να χειριστεί τις ανωμαλίες που απαντώνται στα δεδομένα. Κάθε φορά που αποτυγχάνει η ανάλυση του συγκεκριμένου κανόνα παραγωγής, εγείρεται μια εξαίρεση και εκτελείται ο αντίστοιχος exception handler.

```
Page Book_Reviews
01: $Book_Reviews: <html><body>$Book </body></html>;
02: $Book:  <b>Book Name </b> $bName <b>Reviews </b>
        [<ol> ( <li><b> Reviewer Name </b> $rName <b>
                Rating </b>$rate <b> Text </b> $text $TP )* </ol>];
03: $bName:              *(?<b>);
04: $rName:              *(?<b>);
05: $rate:               *(?<b>);
06: $text:               *(?</li>);

07: $TP:                 {
08:                       $bName, $rName
09:                       $rate
10:                       $text
11:             }
12: END
```

Εικόνα 2.2-6

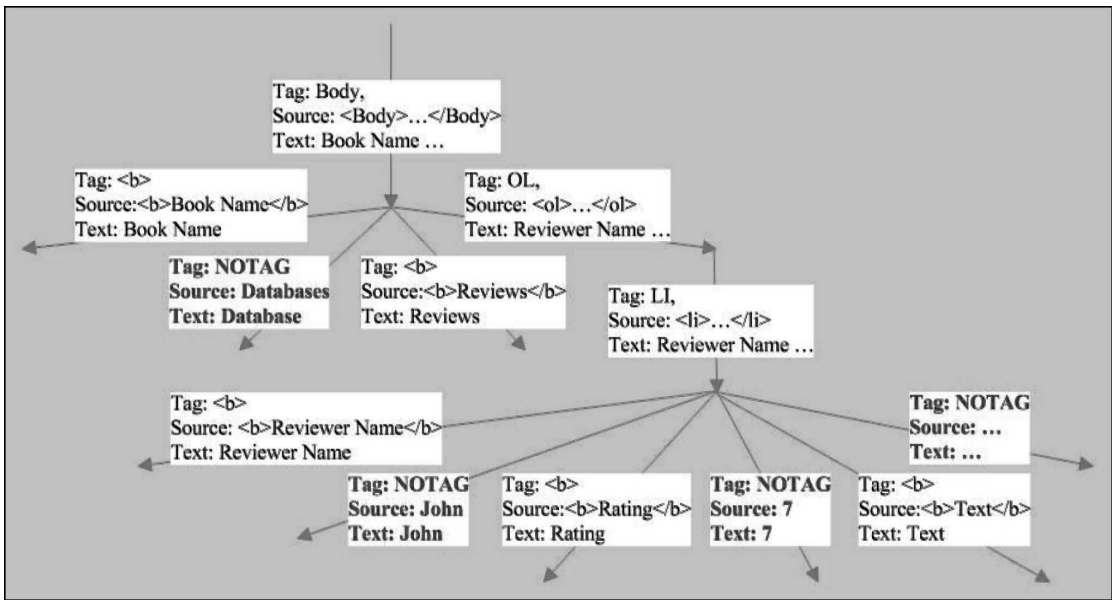
2.2.3.1.3 WebOQL

Η WebOQL είναι μια συναρτησιακή γλώσσα που μπορεί να χρησιμοποιηθεί ως γλώσσα ερωτημάτων για το διαδίκτυο, για ημι-δομημένα δεδομένα και για ανακατασκευή ιστότοπων [AM98]. Η βασική δομή δεδομένων που παρέχει είναι το *υπερδέντρο*. Τα υπερδέντρα είναι ταξινομημένα δέντρα με ετικέτες στις ακμές, που μπορούν να χρησιμοποιηθούν για να μοντελοποιήσουν ένα σχεσιακό πίνακα, ένα αρχείο Bibtex, μια ιεραρχία καταλόγων κ.τ.λ. Το επίπεδο αφαίρεσης του μοντέλου δεδομένων μπορεί να υποστηρίξει συλλογές, εμφωλευμένες δομές και ταξινομήσεις. Η Εικόνα 2.2-7 δείχνει το υπερδέντρο για την σελίδα p_{e1} του παραδείγματος εργασίας που δόθηκε. Όπως φαίνεται στην εικόνα, η δομή του δέντρου είναι παρόμοια με τη δομή ενός δέντρου DOM, όπου οι ακμές φέρουν ετικέτες με εγγραφές με 3

γνωρίσματα, Tag, Source και Text, που αντιστοιχούν στο όνομα του tag, το κομμάτι του κώδικα HTML και στο κείμενο χωρίς τη σήμανση αντίστοιχα. Η βασική κατασκευή που παρέχει η WebOQL είναι η γνωστή select-from-where. Η γλώσσα έχει τη δυνατότητα να προσομοιάσει όλες τις λειτουργίες στην εμφωλευμένη σχεσιακή άλγεβρα και να υπολογίσει το μεταβατικό κλείσιμο σε μια αυθαίρετη δυαδική σχέση. Σαν παράδειγμα, το παρακάτω ερώτημα εξάγει τα ονόματα “Jeff” και “Jane” από τη σελίδα pe2, όπου οι τόνοι και τα θαυμαστικά υποδηλώνουν το πρώτο υποδέντρο και το τελευταίο υποδεντρο αντίστοιχα. Οι μεταβλητές, ανάλογα και με την περίπτωση, κάνουν επαναλήψεις πάνω στα απλά δέντρα ή στα τελευταία δέντρα του υποδέντρου που προσδιορίζεται μετά από τον τελεστή “in”.

Select [Z!΄.Text]
From x in **browse** (“pe2.html”)΄, y in x΄, Z in y΄
Where x.Tag = “ol” and Z.Text=“Reviewer Name”

Επιπλέον του χειρισμού των δεδομένων μέσω των υπερδέντρων, η γλώσσα μπορεί να χρησιμοποιηθεί επίσης για ανακατασκευή στο διαδίκτυο, κάνοντας διαθέσιμο το αποτέλεσμα του ερωτήματος σε άλλες εφαρμογές.



Εικόνα 2.2-7

2.2.3.1.4 W4F

Το W4F (Wysiwyg Web Wrapper Factory) είναι μια εργαλειοθήκη της Java για την παραγωγή διαδικτυακών wrappers [SA01]. Η διαδικασία της ανάπτυξης των wrappers αποτελείται από τρία ανεξάρτητα στρώματα : *το στρώμα ανάκτησης*, *το στρώμα εξαγωγής* και *το στρώμα απεικόνισης*. Στο στρώμα ανάκτησης, γίνεται ανάκτηση του εγγράφου προς επεξεργασία (από το διαδίκτυο, μέσω του πρωτοκόλλου HTTP), το οποίο στη συνέχεια καθαρίζεται και οδηγείται σε έναν HTML parser που κατασκευάζει το αντίστοιχο parse tree σύμφωνα με το μοντέλο DOM. Στο στρώμα εξαγωγής, εφαρμόζονται κανόνες εξαγωγής στο parse tree και οι πληροφορίες που εξάγονται αποθηκεύονται σε ένα

εσωτερικό σχήμα του W4F, που καλείται Nested String List (NSL). Στο στρώμα απεικόνισης τέλος, οι δομές NSL εξάγονται προς την εφαρμογή ανώτερου επιπέδου. σύμφωνα με τους κανόνες απεικόνισης. Οι κανόνες εξαγωγής εκφράζονται μέσω της HEL (HTML Extraction Language), η οποία χρησιμοποιεί το μονοπάτι του δέντρου συντακτικής ανάλυσης της HTML (δέντρο DOM) για να προσδιορίσει τα δεδομένα που πρέπει να εντοπιστούν. Για παράδειγμα, για να προσδιορίσουμε τα ονόματα των βαθμολογητών "Jeff" και "Jane" όπως και πριν από τη σελίδα pe2, μπορούμε να χρησιμοποιήσουμε την έκφραση

html.body.ol[0].li[*].pcdata[0].txt,

όπου το σύμβολο [*] αντιστοιχεί σε οποιονδήποτε αριθμό (στην περίπτωσή μας 0 και 1). Η γλώσσα προσφέρει επίσης κανονικές εκφράσεις και περιορισμούς για να προσδιορίσει μικρότερα τμήματα των δεδομένων. Για παράδειγμα, οι χρήστες μπορούν να χρησιμοποιήσουν μια κανονική έκφραση για να πετύχουν ταίριασμα ή να χωρίσουν (*match* και *split* αντίστοιχα, σύμφωνα με τη σύνταξη σε Perl) το string που λαμβάνεται από το μονοπάτι του DOM δέντρου. Τέλος, ο τελεστής *fork* επιτρέπει την κατασκευή μιας εμφωλευμένης λίστας από string, ακολουθώντας πολλαπλά υπο-μονοπάτια ταυτοχρόνως. Για να διευκολύνει το χρήστη να προσδιορίσει το μονοπάτι του DOM δέντρου, η εργαλειοθήκη έχει σχεδιαστεί να υποστηρίζει wysiwyg μέσω έξυπνων wizards.

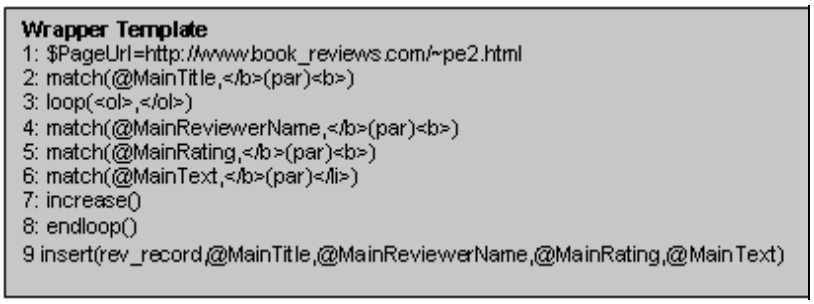
2.2.3.1.5 Σύστημα Εξαγωγής Πληροφορίας του συστήματος Νέστωρ

Μια ακόμα περίπτωση που παρουσιάζει ενδιαφέρον είναι η γλώσσα WTL (Wrapper Template Language) [KVSP01], μια scripting γλώσσα υψηλού επιπέδου για τον ορισμό μοντέλων εξαγωγής πληροφορίας, που αναπτύχθηκε στο εργαστήριο Συστημάτων Γνώσεων και Βάσεων Δεδομένων του Ε.Μ.Π. στα πλαίσια του έργου «Νέστωρ», ενός διαδικτυακού συστήματος που υποστηρίζει τη διαχείριση πηγών όπως βιβλία, πρακτικά συνεδρίων, περιοδικά, papers κ.α.. Σκοπός των wrappers του «Νέστωρ» είναι η αυτόματη ανάκτηση του πίνακα περιεχομένων για πρακτικά συνεδρίων και περιοδικά από τις αντίστοιχες ιστοσελίδες των εκδοτών. Αντί να δημιουργηθεί ένα σύνολο από προγράμματα, ένα για κάθε διαφορετική πηγή, αναπτύχθηκε ένας ευέλικτος και ισχυρός μηχανισμός για να ορίζονται και να παράγονται δυναμικά μοντέλα για wrappers. Παρόλο που η WTL είναι ενσωματωμένη στο Νέστορα και χρησιμοποιεί τη δική του βάση δεδομένων, δεν κάνει καμία παραδοχή για το σχήμα της βάσης δεδομένων, τη μορφή της πηγής των δεδομένων και τη φύση της εφαρμογής που τη φιλοξενεί, πράγμα που την κάνει κατάλληλη για οποιαδήποτε εφαρμογή στοχεύει στην εξαγωγή πληροφοριών από οποιοδήποτε ημι-δομημένο έγγραφο ή ακόμη και από ελεύθερο κείμενο, και στην αποθήκευσή τους σε μια σχεσιακή βάση δεδομένων σε πλειάδες. Η βασική ιδέα της WTL είναι η δυναμική παραγωγή wrappers μέσα από κώδικα υψηλού επιπέδου που ορίζει κάποια μοντέλα χωρίς να χρειάζεται ο προγραμματιστής να απασχολείται με τον ακριβή κώδικα του κάθε προγράμματος, αποφεύγοντας έτσι την επανάληψη παρόμοιων εργασιών.

Η WTL χρησιμοποιεί έναν σύνολο από δεσμευμένες μεταβλητές και εντολές. Η βασικότερη δεσμευμένη μεταβλητή, είναι η μεταβλητή $PageUrl, στην οποία φυλάσσεται το url της ιστοσελίδας που πρόκειται να αναλυθεί, και η οποία ορίζεται υποχρεωτικά στην αρχή του προγράμματος. Για να μπορεί να οριστεί δυναμικά η τιμή του url, η τιμή της $ PageUrl μπορεί να ανατεθεί με τη βοήθεια ειδικών μεταβλητών εισαγωγής, των $FormVar1, ..., $FormVarN, οι οποίες χρησιμοποιούνται για πέρασμα ορισμάτων. Η εξαγωγή των επιθυμητών δεδομένων βασίζεται στα HTML tags της σελίδας. Οι κανόνες εξαγωγής έχουν τη μορφή:

$$\text{match}(\mu\varepsilon\tau\alpha\beta\lambda\eta\tau\acute{\eta}, \chi\alpha\rho\alpha\kappa\tau\_\alpha\rho\chi\acute{\eta}\varsigma(\text{par})\chi\alpha\rho\alpha\kappa\tau\_\tau\acute{\varepsilon}\lambda ous),$$

όπου *μεταβλητή* είναι το όνομα της μεταβλητής στην οποία θέλουμε να αποθηκευθούν τα δεδομένα, *χαρακτ_αρχής* είναι το σύνολο τον χαρακτήρων στην αρχή της συμβολοακολουθίας που μας ενδιαφέρει, *χαρακτ_τέλους* είναι το σύνολο το χαρακτήρων στο τέλος αυτής, ο μετα-χαρακτήρας par υποδηλώνει οποιονδήποτε χαρακτήρα, και οι παρενθέσεις δηλώνουν το τμήμα πληροφορίας που θέλουμε να εξαχθεί. Αν θέλουμε δηλαδή να εξαχθούν και οι χαρακτήρες αρχής και τέλους, πρέπει να εσωκλείονται και αυτοί στις παρενθέσεις. Για να δημιουργηθούν μεταβλητές συναφείς με τη δομή του πίνακα της βάσης που θέλουμε να αποθηκευθούν τα δεδομένα, χρησιμοποιείται το πρόθεμα “@Main”, ακολουθούμενο από οποιοδήποτε όνομα. Οι μεταβλητές *@Main* είναι δυναμικοί πίνακες που μπορούν να αποθηκεύσουν περισσότερες από μία τιμές. Η αύξηση του δείκτη όλων των *@Main* μεταβλητών γίνεται με την εντολή *increase*. Αφού έχουν αποθηκευτεί οι επιθυμητές τιμές σε μια μεταβλητή *@Main* μπορούν στη συνέχεια να εισαχθούν στη βάση δεδομένων σε ένα βήμα, με την εντολή *insert*, η οποία δέχεται σαν ορίσματα το όνομα του πίνακα στον οποίο θα γίνει η εισαγωγή των δεδομένων και τα ονόματα των *@Main* μεταβλητών που περιέχουν τα δεδομένα κάθε στήλης του πίνακα, με την κατάλληλη πάντα σειρά. Αν πρέπει να γίνουν κάποιοι χειρισμοί στα δεδομένα πριν την αποθήκευσή τους στη βάση, τότε μπορούν να χρησιμοποιηθούν οι μεταβλητές *$Temp*, οι οποίες είναι προσωρινές μεταβλητές που δηλώνονται με το πρόθεμα “$Temp” ακολουθούμενο από οποιοδήποτε όνομα. Ο χειρισμός των δεδομένων που είναι αποθηκευμένα σε τέτοιες μεταβλητές γίνεται με τη χρήση της εντολής *perl*, ακολουθούμενης από μια εντολή σε γλώσσα Perl. Τα δεδομένα που βρίσκονται σε προσωρινές μεταβλητές μπορούν να αποθηκευθούν σε *@Main* μεταβλητές με την εντολή *bind*.Η γλώσσα παρέχει ακόμα τις εντολές *convert* και *loop*. Η πρώτη μπορεί να χρησιμοποιηθεί για την μετατροπή σε απόλυτη μορφή συνδέσμων που είναι αποθηκευμένοι σε σχετική μορφή, ενώ η δεύτερη εκτελεί τις εντολές που περιλαμβάνονται στο σώμα της επαναληπτικά σε όλο το κομμάτι που προσδιορίζεται από τα ορίσματα που της δίνονται. Επομένως, το μοντέλο του wrapper για την σελίδα p_{e2} του παραδείγματός μας, είναι αυτό που φαίνεται στην εικόνα:



Εικόνα 2.2-8

2.2.3.2 Επιβλεπόμενα Συστήματα WI

Όπως φαίνεται και στην Εικόνα 2.2-3, τα επιβλεπόμενα συστήματα WI δέχονται στην είσοδο ένα σύνολο ιστοσελίδων όπου έχουν σημειωθεί παραδείγματα των προς εξαγωγή δεδομένων και δίνουν ως έξοδο έναν wrapper. Ο χρήστης παρέχει ένα αρχικό σύνολο παραδειγμάτων με ετικέτες και το σύστημα, (μέσω ενός GUI) μπορεί να προτείνει επιπλέον σελίδες για να τοποθετήσει ετικέτες ο χρήστης. Σε τέτοιου είδους συστήματα, αντί για προγραμματιστές μπορούν να χρησιμοποιηθούν γενικοί χρήστες, οι οποίοι θα εκπαιδευτούν στη χρήση των GUI τοποθέτησης ετικετών, μειώνοντας έτσι το κόστος παραγωγής του wrapper. Τέτοια συστήματα είναι τα SRV [F98], RAPIER [CM98] και STALKER [MMK99].

2.2.3.2.1 SRV

Ο SRV είναι ένας σχεσιακός αλγόριθμος από-πάνω-προς-τα-κάτω που παράγει κανόνες παραγωγής single-slot [F98]. Αντιμετωπίζει την εξαγωγή πληροφορίας σαν ένα πρόβλημα ταξινόμησης. Τα έγγραφα εισόδου διασπώνται σε λεκτικές μονάδες και όλες οι υπο-συμβολοσειρές από συνεχόμενες λεκτικές μονάδες (text fragments) χαρακτηρίζονται είτε ως αντικείμενο εξαγωγής (θετικά παραδείγματα) είτε ως μη-αντικείμενο εξαγωγής (αρνητικά παραδείγματα). Οι κανόνες που παράγονται από τον SRV είναι λογικοί κανόνες που βασίζονται σε ένα σύνολο χαρακτηριστικών προσανατολισμένων στις λεκτικές μονάδες (κατηγορήματα). Αυτά τα χαρακτηριστικά είναι δύο ειδών: απλά και σχεσιακά. Ένα απλό χαρακτηριστικό είναι μια συνάρτηση που αντιστοιχεί μια λεκτική μονάδα σε μια διακριτή τιμή όπως το μήκος, ο τύπος του χαρακτήρα (π.χ. αριθμητικός), η ορθογραφία (π.χ. με κεφαλαία) και το μέρος του λόγου (π.χ. ρήμα). Ένα σχεσιακό χαρακτηριστικό αντιστοιχεί μια λεκτική μονάδα σε μια άλλη λεκτική μονάδα, π.χ. αντιστοίχιση στις συναφείς (προηγούμενες ή επόμενες) λεκτικές μονάδες της μονάδας εισόδου. Ο αλγόριθμος εκμάθησης συνεχίζει όπως ο αλγόριθμος FOIL, ξεκινώντας από ολόκληρο το σύνολο των παραδειγμάτων και προσθέτοντας κατηγορήματα με άπληστο τρόπο για να καλύψει όσο το δυνατόν περισσότερα θετικά παραδείγματα και όσο το δυνατόν λιγότερα αρνητικά. Για παράδειγμα, για την εξαγωγή της βαθμολογίας της κριτικής στην περίπτωση που μελετάμε, ο SRV μπορεί να επιστρέψει τον εξής κανόνα :

length(=1)

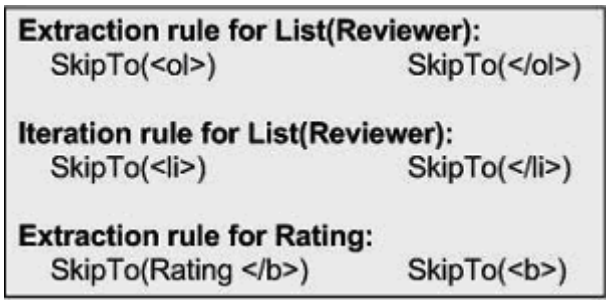
every(numeric true),

every(in_list true)

Ο κανόνας αυτός δηλώνει ότι η βαθμολογία είναι μια απλή αριθμητική λέξη και προκύπτει μέσα σε ένα HTML list tag.

2.2.3.2.2 RAPIER

Το RAPIER εστιάζει και αυτό στην εξαγωγή σε επίπεδο πεδίου, αλλά χρησιμοποιεί έναν σχεσιακό αλγόριθμό εκμάθησης από-κάτω-προς-τα-πάνω βασισμένο στη συμπίεση [CM98]. Αυτό σημαίνει ότι ξεκινά με τους περισσότερο συγκεκριμένους κανόνες και στη συνέχεια τους αντικαθιστά με περισσότερο γενικούς. Το RAPIER μαθαίνει μοτίβα single slot που χρησιμοποιούν συντακτική και σημασιολογική πληροφορία, όπως έναν tagger για τα μέρη του λόγου ή ένα λεξικό (WordNet). Οι κανόνες εξαγωγής αποτελούνται από τρία διακριτά μοτίβα. Το πρώτο είναι το μοτίβο του prefiller, το οποίο επιτυγχάνει ταίριασμα με κείμενο που προηγείται ακριβώς του κειμένου της σχισμής (filler), το δεύτερο είναι το μοτίβο που επιτυγχάνει ταίριασμα με το ίδιο το κείμενο της σχισμής, και το τελευταίο είναι το μοτίβο postfiller που επιτυγχάνει ταίριασμα με το κείμενο που ακολουθεί ακριβώς το κείμενο της σχισμής. Για παράδειγμα, η Εικόνα 2.2-9 δείχνει τον κανόνα εξαγωγής για τον τίτλο του βιβλίου, του οποίου προηγούνται ακριβώς οι λέξεις “Book”, “Name” και “” και το οποίο ακολουθεί αμέσως η λέξη “”. Το μοτίβο της σχισμής (“Filler pattern”) προσδιορίζει ότι ο τίτλος αποτελείται από το πολύ δύο λέξεις, στις οποίες έχει τοποθετηθεί η ετικέτα “nn” ή “nns” από τον POS tagger (υπονοώντας ένα ή δύο κοινά ουσιαστικά στον ενικό ή πληθυντικό αριθμό).

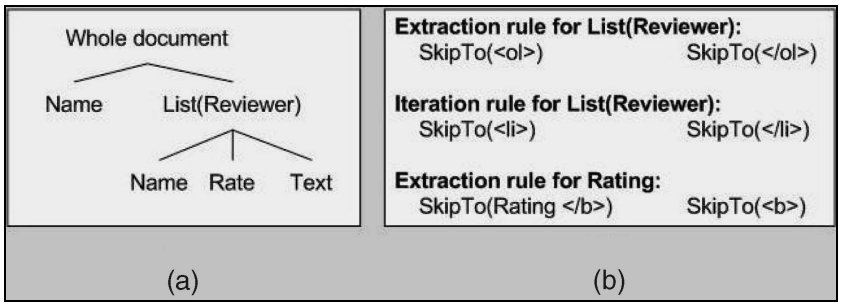


Εικόνα 2.2-9

2.2.3.2.3 STALKER

Το STALKER είναι ένα σύστημα WI το οποίο πραγματοποιεί εξαγωγή δεδομένων με ιεραρχικό τρόπο [MMK99]. Εισάγει την έννοια του φορμαλισμού του ενσωματωμένου καταλόγου (embedded catalog – EC) για να περιγράψει τη δομή μιας ευρείας γκάμας ημι-δομημένων εγγράφων. Η περιγραφή EC για μια σελίδα είναι μια δομή που έχει τη μορφή δέντρου, στην οποία τα φύλλα είναι τα προς εξαγωγή γνωρίσματα και οι εσωτερικοί κόμβοι είναι λίστες από πλειάδες. Για κάθε κόμβο του δέντρου, ο wrapper

χρειάζεται έναν κανόνα για να εξάγει αυτόν τον κόμβο από τον γονέα του. Επιπλέον, για κάθε κόμβο-λίστα, ο wrapper απαιτεί έναν κανόνα επανάληψης που αποσυνθέτει τη λίστα σε μεμονωμένες πλειάδες. Επομένως, το STALKER μετατρέπει το δύσκολο πρόβλημα της εξαγωγής δεδομένων από ένα αυθαίρετο σύνθετο έγγραφο, σε μια σειρά από ευκολότερες εργασίες εξαγωγής από το ανώτερο στο χαμηλότερο επίπεδο. Ακόμη, ο εξαγωγέας χρησιμοποιεί ανάλυση πολλαπλών περασμάτων για να διαχειριστεί τα απόντα γνωρίσματα και τις πολλαπλές μεταθέσεις. Οι κανόνες εξαγωγής παράγονται με τη χρήση ενός ακολουθιακού αλγορίθμου κάλυψης, ο οποίος ξεκινά από γραμμικά αυτόματα αναφοράς που να καλύπτουν όσο το δυνατόν περισσότερα θετικά παραδείγματα, και στη συνέχεια προσπαθεί να παράγει νέα αυτόματα για τα υπόλοιπα παραδείγματα. Ένα δέντρο EC του STALKER που περιγράφει τη δομή δεδομένων του δοσμένου παραδείγματος φαίνεται στην Εικόνα 2.2-10a, ενώ κάποιοι από τους κανόνες παραγωγής φαίνονται στην Εικόνα 2.2-10b. Για παράδειγμα, οι βαθμολογίες μπορεί να εξαχθούν εφαρμόζοντας αρχικά τον κανόνα εξαγωγής List(Reviewer), (ο οποίος ξεκινάει με “
” και τελειώνει με “”) σε ολόκληρο το έγγραφο και στη συνέχεια τον κανόνα εξαγωγής Rating σε κάθε έναν από τους βαθμολογητές, οι οποίοι έχει προκύψει από την εφαρμογή του κανόνα επανάληψης στο List(Reviewer).



Εικόνα 2.2-10

2.2.3.3 Ημι-επιβλεπόμενα συστήματα WI

Τα συστήματα που ανήκουν σε αυτήν την κατηγορία περιλαμβάνουν τα IEPAD [HCHLC05], OLERA [CK04] και Thresher [HK05]. Σε αντίθεση με την επιβλεπόμενη προσέγγιση, τα συστήματα της κατηγορίας αυτής δέχονται στην είσοδο ένα χοντρικό (αντί για ολοκληρωμένο και ακριβές) παράδειγμα από τους χρήστες για την παραγωγή των κανόνων εξαγωγής, και για το λόγο αυτό καλούνται ημι-επιβλεπόμενα. Στο IEPAD, παρόλο που δεν απαιτούνται σελίδες εκμάθησης με ετικέτες, απαιτείται προσπάθεια από το χρήστη μετά την εξαγωγή για να επιλέξει το μοτίβο-στόχο και να επισημάνει τα δεδομένα προς εξαγωγή. Αυτά τα συστήματα έχουν ως στόχο εργασίες εξαγωγής επιπέδου εγγραφής. Από τη στιγμή που δεν καθορίζονται από την αρχή τα αντικείμενα της εξαγωγής, απαιτείται ένα GUI για τον καθορισμό των αντικειμένων της εξαγωγής από τους χρήστες μετά τη φάση της εκμάθησης. Έτσι, απαιτείται η επίβλεψη του χρήστη και εδώ.

2.2.3.3.1 IEPAD

Το IEPAD είναι ένα από τα πρώτα συστήματα ΕΔ που παράγουν μοτίβα εξαγωγής από ιστοσελίδες χωρίς ετικέτες [HCHLC05]. Αυτή η μέθοδος εκμεταλλεύεται το γεγονός ότι αν μια ιστοσελίδα περιέχει πολλαπλές (ομογενείς) εγγραφές δεδομένων προς εξαγωγή, τότε τις χειρίζεται με ενιαίο τρόπο, χρησιμοποιώντας το ίδιο πρότυπο για να πετύχει καλή οπτικοποίηση. Έτσι, μπορούμε να ανακαλύψουμε επαναλαμβανόμενα μοτίβα αν η σελίδα είναι καλά κωδικοποιημένη. Επομένως, μπορούν να δημιουργηθούν εκπαιδευόμενοι wrappers από την ανακάλυψη επαναλαμβανόμενων μοτίβων. Το IEPAD χρησιμοποιεί για το σκοπό αυτό μια δομή δεδομένων, το PAT δέντρο, το οποίο είναι ένα δυαδικό δέντρο καταλήξεων. Καθώς μια τέτοια δομή δεδομένων καταγράφει μόνο το ακριβές ταίριασμα των καταλήξεων, το IEPAD επιπλέον εφαρμόζει τον κεντρικό αλγόριθμο αστέρα για να ευθυγραμμίσει πολλαπλά strings που ξεκινούν σε κάθε εμφάνιση μιας επανάληψης και τελειώνουν πριν την έναρξη της επόμενης . Τέλος, χρησιμοποιείται μια χαρακτηριστική αναπαράσταση για να υποδηλώσει το πρότυπο για την κατανόηση όλων των εγγραφών. Για το παράδειγμά μας, μόνο η σελίδα p_{e2} μπορεί να χρησιμοποιηθεί ως είσοδος για το IEPAD. Κωδικοποιώντας κάθε tag σαν μια ξεχωριστή λεκτική μονάδα και κάθε κείμενο μεταξύ δύο γειτονικών tags ως μια ειδική λεκτική μονάδα “T”, το IEPAD ανακαλύπτει το μοτίβο

“
TTTTTT”

το οποίο εμφανίζεται δύο φορές. Ο χρήστης μετά πρέπει να προσδιορίσει, για παράδειγμα, την δεύτερη, τέταρτη και έκτη λεκτική μονάδα “T”, ως τα σχετικά δεδομένα (που υποδηλώνουν το όνομα του κριτικού, τη βαθμολογία και το σχόλιο αντίστοιχα).

2.2.3.3.2 OLERA

Το OLERA είναι ένα ημι-επιβλεπόμενο σύστημα ΕΠ που δέχεται ένα χονδρικό παράδειγμα από το χρήστη για την παραγωγή κανόνων εξαγωγής [CK04]. Το OLERA μπορεί να μάθει κανόνες εξαγωγής για σελίδες που περιέχουν ακόμη και μόνο μία εγγραφή δεδομένων, μια περίπτωση όπου το IEPAD αποτυγχάνει. Το OLERA αποτελείται από τρεις βασικές λειτουργίες: 1)*Περιορισμός ενός τμήματος ενδιαφέροντος (block πληροφοριών)*: όπου ο χρήστης μαρκάρει ένα block πληροφοριών που περιέχει μια εγγραφή προς εξαγωγή, ώστε το OLERA να ανακαλύψει άλλα αντίστοιχα blocks (με τη χρήση μιας τεχνικής προσεγγιστικού ταιριάσματος) και να τα εκγενικεύσει σε ένα μοτίβο εξαγωγής (με τη χρήση μιας τεχνικής *ευθυγράμμισης πολλαπλών strings*). 2)*Εφαρμογή Drill-down ή Roll-up σε ένα slot πληροφορίας*: το drill-down επιτρέπει στο χρήστη να μεταφερθεί από ένα τμήμα κειμένου σε πιο λεπτομερή συνθετικά στοιχεία αυτού, ενώ το roll-up συνδυάζει αρκετά slots για να σχηματίσει μια μονάδα πληροφορίας με κάποιο νόημα. 3)*Καθορισμός σχετικών slots πληροφοριών* για τον προσδιορισμό ενός σχήματος, όπως στο IEPAD.

2.2.3.3.3 Thresher

Το Thresher [HK05] είναι επίσης μια ημι-επιβλεπόμενη προσέγγιση, που μοιάζει με το OLERA. Το GUI για το Thresher βρίσκεται ενσωματωμένο στον φυλλομετρητή Haystack, που επιτρέπει στους χρήστες να καθορίσουν παραδείγματα ή σημασιολογικό περιεχόμενο τονίζοντάς τα και περιγράφοντας τη σημασία τους (τοποθετώντας ετικέτες). Παρόλα αυτά, χρησιμοποιεί την απόσταση μέσα στο δέντρο (αντί για την απόσταση μέσα στο string όπως συμβαίνει στο OLERA) μεταξύ των υποδέντρων του DOM στα παραδείγματα αυτά για να δημιουργήσει έναν wrapper. Μετά, επιτρέπει στο χρήστη να εφαρμόσει τις κλάσεις και τα κατηγορήματα της σημασιολογικής διαδικτυακής γλώσσας RDF (Resource Description Framework) στους κόμβους των wrappers αυτών.

2.2.3.4 Μη-επιβλεπόμενα Συστήματα WI

Όπως φαίνεται στην Εικόνα 2.2-3 , τα μη-επιβλεπόμενα συστήματα δεν χρησιμοποιούν καθόλου παραδείγματα εκμάθησης με ετικέτες και δεν δέχονται καμία αλληλεπίδραση από το χρήστη στην παραγωγή του wrapper. Τα συστήματα RoadRunner [CMM01] και EXALG [AM03] που περιγράφονται παρακάτω είναι σχεδιασμένα για εργασίες εξαγωγής επιπέδου σελίδας, ενώ το DeLa [WL02], [WL03] έχει σχεδιαστεί για εργασίες εξαγωγής επιπέδου εγγραφής. Σε αντίθεση με τα επιβλεπόμενα συστήματα όπου τα αντικείμενα της εξαγωγής καθορίζονται από τους χρήστες, εδώ τα αντικείμενα της εξαγωγής ορίζονται ως τα δεδομένα που χρησιμοποιούνται για την δημιουργία της σελίδας ή τα κείμενα χωρίς tags σε περιοχές της σελίδας εισόδου πλούσιες σε κείμενο. Σε μερικές περιπτώσεις, μπορεί να ταιριάζουν διάφορα σχήματα στις σελίδες εκμάθησης εξαιτίας της παρουσίας γνωρισμάτων χωρίς δεδομένα, τα οποία οδηγούν σε αμφιβολία. Η επιλογή του σωστού σχήματος αφήνεται στους χρήστες. Παρομοίως, αν δεν χρειάζονται όλα τα δεδομένα, μπορεί να απαιτείται μετα-επεξεργασία, ώστε ο χρήστης να επιλέξει τα σχετικά δεδομένα και να δώσει σε κάθε κομμάτι δεδομένων ένα κατάλληλο όνομα.

2.2.3.4.1 DeLa

Σαν μια επέκταση του IEPAD, το DeLa καταργεί την αλληλεπίδραση του χρήστη στη γενίκευση των κανόνων εξαγωγής και ασχολείται με την εξαγωγή εμφωλευμένων αντικειμένων [WL02], [WL03]. Η διαδικασία παραγωγής του wrapper στο DeLa γίνεται σε δύο συνεχόμενα βήματα. Πρώτα, εφαρμόζεται ένας αλγόριθμος εξαγωγής τμήματος πλούσιου σε κείμενο (*Data-rich Section Extraction – DSE*), ο οποίος έχει σχεδιαστεί για να εξάγει περιοχές των ιστοσελίδων που είναι πλούσιες σε κείμενο συγκρίνοντας τα DOM δέντρα δύο ιστοσελίδων (από τον ίδιο ιστότοπο), και απορρίπτοντας κόμβους που έχουν πανομοιότυπα υποδέντρα. Δεύτερον, χρησιμοποιείται ένας *εξαγωγέας μοτίβων* για να ανακαλύψει συνεχώς επαναλαμβανόμενα μοτίβα (continuously repeated – C-repeated) χρησιμοποιώντας δέντρα καταλήξεων. Διατηρώντας την τελευταία εμφάνιση κάθε μοτίβου, ανακαλύπτονται επαναληπτικά

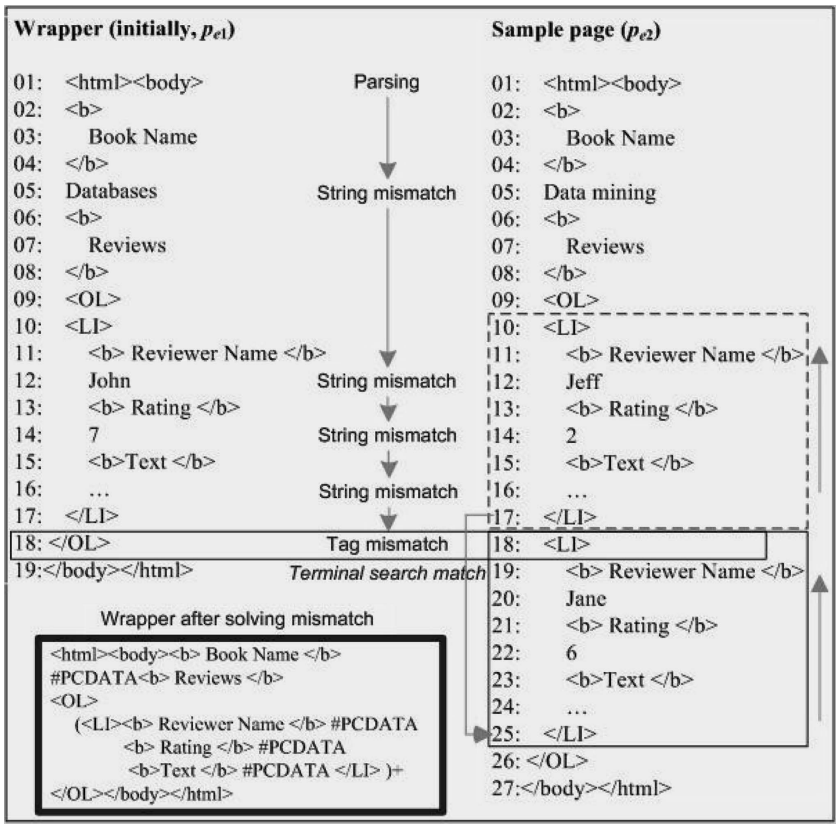
νέα επαναλαμβανόμενα μοτίβα από τη νέα ακολουθία, δημιουργώντας μια εμφωλευμένη δομή. Για παράδειγμα, δεδομένης της συμβολοακολουθίας

“<P><A>T</A><A>T</A>T</P><P><A>T</A>T</P>”,

το DeLa θα ανακαλύψει το μοτίβο “<P><A>T</A>T</P>” από την άμεση ακολουθία “<P><A>T</A>T</P><P><A>T</A>T</P>” και θα επιστρέψει το μοτίβο σε παρένθεση “(<P>(<A>T</A>)*T<P>)*” για να υποδηλώσει την εμφωλευμένη δομή. Καθώς ένα ανακαλυφθέν μοτίβο μπορεί να ξεπεράσει τα σύνορα ενός αντικειμένου δεδομένων, το DeLa δοκιμάζει K σελίδες και επιλέγει τη μία με τη μεγαλύτερη υποστήριξη σελίδας. Και πάλι, κάθε εμφάνιση της κανονικής έκφρασης αναπαριστά ένα αντικείμενο δεδομένων. Τα αντικείμενα δεδομένων μετατρέπονται στη συνέχεια σε ένα σχεσιακό πίνακα όπου πολλαπλές τιμές ενός γνωρίσματος μοιράζονται σε πολλαπλές σειρές του πίνακα. Τέλος, τοποθετούνται επιγραφές στις στήλες του πίνακα δεδομένων από τέσσερις ευριστικές συναρτήσεις, οι οποίες περιλαμβάνουν τις επιγραφές διαφόρων στοιχείων σε μια φόρμα αναζήτησης ή σε πίνακες της σελίδας, και το μέγιστο πρόθεμα και τη μέγιστη κατάληξη που είναι κοινά για όλα τα κελιά της στήλης.

2.2.3.4.2 RoadRunner

Το RoadRunner θεωρεί τη διαδικασία δημιουργίας της ιστοσελίδας σαν μια κωδικοποίηση των περιεχομένων της αρχικής βάσης δεδομένων σε συμβολοακολουθίες κώδικα HTML [CMM01]. Συνεπώς, η εξαγωγή δεδομένων θεωρείται μια διαδικασία αποκωδικοποίησης. Για το λόγο αυτό, η δημιουργία ενός wrapper για ένα σύνολο HTML σελίδων αντιστοιχεί στη συναγωγή μιας γραμματικής για τον κώδικα HTML. Το σύστημα χρησιμοποιεί την τεχνική ταιριάσματος ACME για να συγκρίνει HTML σελίδες της ίδιας τάξης και να δημιουργήσει έναν wrapper βασισμένο στις ομοιότητες και τις διαφορές τους. Ξεκινά με τη σύγκριση δύο σελίδων, χρησιμοποιώντας την τεχνική ACME για να ευθυγραμμίσει τις λεκτικές μονάδες που ταιριάζουν μεταξύ τους και σταματά όταν βρίσκει λεκτικές μονάδες που δεν ταιριάζουν. Υπάρχουν δύο είδη μη-ταιριάσματος: *το μη-ταίριασμα συμβολοακολουθιών* που χρησιμεύει στην ανεύρεση γνωρισμάτων (#PCDATA) και το *μη-ταίριασμα επιγραφών* που χρησιμεύει στην ανεύρεση μετρητών επανάληψης (+) και προαιρετικών (?) μοτίβων. Η Εικόνα 2.2-11 δείχνει ένα παράδειγμα ταιριάσματος για τις πρώτες δύο σελίδες της δοσμένης εργασίας ΕΔ και τον wrapper που παράγεται σαν αποτέλεσμα. Καθώς μπορεί να υπάρξουν αρκετές ευθυγραμμίσεις, το RoagRunner υιοθετεί το UFRE (union-free regular expression) για να μειώσει την πολυπλοκότητα. Το αποτέλεσμα της ευθυγράμμισης για τις δύο πρώτες σελίδες συγκρίνεται στη συνέχεια με την τρίτη σελίδα από την κλάση των σελίδων. Εκτός από την υπομονάδα για την συναγωγή των προτύπων, το RoadRunner περιέχει ακόμη δύο υπομονάδες, για να διευκολύνει την κατασκευή των wrappers, τον Classifier και τον Labeler. Η πρώτη μονάδα, ο *Classifier*, αναλύει σελίδες και τις οργανώνει σε ομάδες με ομογενή δομή, ώστε δηλαδή σελίδες με το ίδιο πρότυπο να ανήκουν στην ίδια ομάδα. Η δεύτερη μονάδα, ο *Labeler*, ανακαλύπτει τα ονόματα των γνωρισμάτων για κάθε κλάση σελίδων.



Εικόνα 2.2-11

2.2.3.4.3 EXALG

Οι Arasu και Molina παρουσίασαν μια αποτελεσματική διατύπωση του προβλήματος της εξαγωγής δεδομένων από σελίδες στο διαδίκτυο [AM03]. Η είσοδος στο EXALG είναι ένα σύνολο σελίδων που δημιουργούνται από το άγνωστο πρότυπο *T* και τις προς εξαγωγή τιμές. Το EXALG συνάγει το πρότυπο *T* και το χρησιμοποιεί για να εξάγει το σύνολο των τιμών από τις κωδικοποιημένες σελίδες στην έξοδο. Το σύστημα εντοπίζει το άγνωστο πρότυπο με τη χρήση δύο τεχνικών, των *διαφοροποιημένων ρόλων* και των *κλάσεων ισοδυναμίας* (EC). Σύμφωνα με την πρώτη τεχνική, δυο εμφανίσεις μιας συγκεκριμένης λεκτικής μονάδας σε δύο διαφορετικά μονοπάτια θεωρείται ότι έχουν διαφορετικούς ρόλους, π.χ. στο παράδειγμά μας ο ρόλος της λέξης “Name” όταν συναντάται μέσα στο “Book Name” είναι διαφορετικός από το ρόλο του όταν συναντάται στο “Reviewer Name”. Σύμφωνα με τη δεύτερη τεχνική, μια κλάση ισοδυναμίας είναι ένα μέγιστο σύνολο από λεκτικές μονάδες που έχουν την ίδια συχνότητα εμφάνισης στις σελίδες εκμάθησης (*διάνυσμα εμφάνισης*). Για παράδειγμα, στην Εικόνα 2.2-4 οι δύο λεκτικές μονάδες <html>1 και <body>2 έχουν το ίδιο διάνυσμα εμφάνισης (<1, 1, 1, 1>), οπότε ανήκουν στην ίδια κλάση ισοδυναμίας. Η ιδέα είναι ότι λεκτικές μονάδες ενός προτύπου που περικλείουν μια πλειάδα δεδομένων έχουν το ίδιο διάνυσμα εμφάνισης και σχηματίζουν μια κλάση ισοδυναμίας. Παρόλα αυτά, για την αποφυγή του τυχαίου σχηματισμού κλάσεων ισοδυναμίας από λεκτικές μονάδες, οι κλάσεις με ανεπαρκή υποστήριξη (μικρός αριθμός σελίδων που περιέχουν τις λεκτικές μονάδες) και με ανεπαρκές

μέγεθος (μικρός αριθμός των μονάδων σε μια κλάση ισοδυναμίας) φιλτράρονται. Επιπλέον, για τη συμμόρφωση με την ιεραρχική δομή του σχήματος δεδομένων, οι κλάσεις ισοδυναμίας πρέπει να είναι από κοινού εμφωλευμένες και οι λεκτικές μονάδες σε κάθε κλάση πρέπει να είναι διατεταγμένες. Αυτές οι έγκυρες κλάσεις ισοδυναμίας χρησιμοποιούνται στη συνέχεια για την κατασκευή του αρχικού προτύπου.

2.2.4 Συμπεράσματα

Παρακάτω παρατίθενται μερικά επιπλέον σχόλια και συμπεράσματα σχετικά με τα συστήματα ΕΔΠ που παρουσιάστηκαν, αναφορικά και με κάποια κριτήρια σύγκρισης που ενδιαφέρουν και που παρουσιάστηκαν στην αρχή.

2.2.4.1 Τύπος σελίδας

Όπως αναφέρθηκε και προηγουμένως, οι ιστοσελίδες μπορεί να είναι δομημένες, ημι-δομημένες ή ελεύθερο κείμενο. Για παράδειγμα, τα επιβλεπόμενα συστήματα ΕΔ και τα χειρωνακτικά συστήματα, σχεδιάζονται να εξάγουν πληροφορίες από διάφορες σελίδες στο διαδίκτυο (π.χ. δεδομένα για καθηγητές από διάφορα πανεπιστήμια), ενώ τα ημι-επιβλεπόμενα και τα μη-επιβλεπόμενα σχεδιάζονται κατά βάση για να εξάγουν δεδομένα από το σελίδες που ακολουθούν κάποιο πρότυπο. Έτσι, τα ημι-επιβλεπόμενα και μη-επιβλεπόμενα συστήματα εξαρτώνται σε μεγάλο βαθμό από το κοινό πρότυπο που χρησιμοποιείται για την παραγωγή των ιστοσελίδων, ενώ τα άλλα περιλαμβάνουν περισσότερα χαρακτηριστικά των λεκτικών μονάδων (όπως ο αριθμός των χαρακτήρων, ο λόγος των χαρακτήρων σε κεφαλαία κ.τ.λ.) για τη συναγωγή των κανόνων εξαγωγής. Ενσωματώνοντας περισσότερα χαρακτηριστικά των σελίδων με πρότυπα, τα μη-επιβλεπόμενα συστήματα παρουσιάζουν μεγάλο βαθμό αυτοματοποίησης στην παραγωγή των κανόνων εξαγωγής, αλλά από την άλλη η δυνατότητα επέκτασής τους σε σελίδες που δεν ακολουθούν πρότυπα είναι πολύ περιορισμένη.

2.2.4.2 Υποστήριξη μη-HTML εγγράφων (Non-HTML Support – NHS)

Ένα άλλο στοιχείο που ενδιαφέρει σχετικά με τα συστήματα παραγωγής wrappers είναι η υποστήριξη σελίδων που δεν είναι γραμμένες σε HTML , κάτι που εξαρτάται από το γνωστικό υπόβαθρο που χρησιμοποιείται από τα συστήματα αυτά. Έτσι, όταν ένα σύστημα αποτυγχάνει στην παραγωγή κανόνων για μια εργασία ΕΠ, οι προγραμματιστές γνωρίζουν πώς να ρυθμίσουν το σύστημα για μια τέτοια εργασία. Τα περισσότερα επιβλεπόμενα συστήματα μπορούν να υποστηρίξουν μη-HTML έγγραφα αλλάζοντας την ιεραρχία παραγωγής κανόνων, ή προσθέτοντας νέα χαρακτηριστικά λεκτικών μονάδων (όπως στο SRV). Τα χειρωνακτικά συστήματα όπως το Minerva και το TSIMMIS και το σύστημα του

«Νέστωρ", όπου οι κανόνες εξαγωγής γράφονται με το χέρι, μπορούν να προσαρμοστούν από τον προγραμματιστή του wrapper για τον χειρισμό τέτοιων εγγράφων. Κάποιοι wrappers όμως, όπως οι WebOQL και W4F βασίζονται έντονα στη χρήση των πληροφοριών των DOM δέντρων στα συστήματά τους, οπότε και δεν μπορούν να υποστηρίξουν μη-HTML έγγραφα, ενώ οι προσεγγίσεις που βασίζονται σε ακολουθίες, όπως τα IEPAD, OLERA, RoadRunner και DeLa μπορούν να ρυθμιστούν για να διαχειρίζονται τα έγγραφα αυτά με την προσθήκη κατάλληλων σχημάτων κωδικοποίησης. Η τεχνολογία των κλάσεων ισοδυναμίας του EXALG τέλος, υποστηρίζει και αυτή μη-HTML έγγραφα, αλλά η επιτυχία εξαρτάται από τη διαφοροποίηση των ρόλων των λεκτικών μονάδων.

2.2.4.3 Επίπεδο εξαγωγής

Σύμφωνα με την κατηγοριοποίηση των εργασιών ΕΠ σε εργασίες επιπέδου πεδίου, εγγραφής, σελίδας και ιστότοπου, τα Rapier και SRV κατατάσσονται στην πρώτη κατηγορία, τα EXALG και RoadRunner ανήκουν στην τρίτη και όλα τα υπόλοιπα ανήκουν στη δεύτερη. Μέχρι τώρα, δεν υπάρχουν wrappers επιπέδου ιστότοπου.

2.2.4.4 Παραλλαγές στα αντικείμενα της εξαγωγής

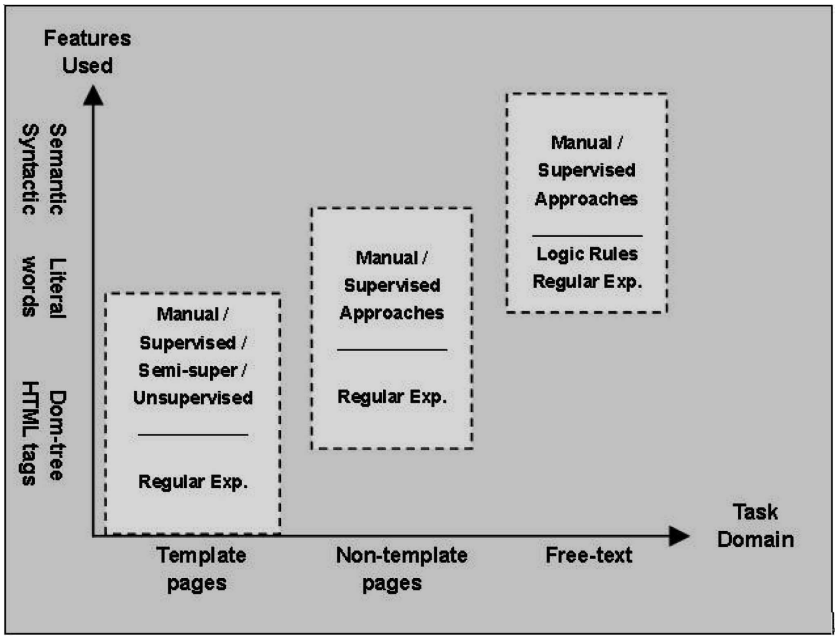
Τα περισσότερα συστήματα υποστηρίζουν απόντα γνωρίσματα και γνωρίσματα πολλαπλών τιμών. Πολλά από αυτά όμως δεν υποστηρίζουν γνωρίσματα πολλαπλής διάταξης, καθώς οι κανόνες εξαγωγής τους βασίζονται στη θέση των πεδίων μέσα σε μια εγγραφή. Από τα συστήματα που αναφέραμε μάλιστα, τα Stalker, IEPAD και OLERA μπορούν να διαχειριστούν και εμφωλευμένα αντικείμενα δεδομένων. Το ίδιο συμβαίνει και με τα RoadRunner και EXALG, τα οποία όμως δεν μπορούν να διαχειριστούν γνωρίσματα πολλαπλής διάταξης.

2.2.4.5 Συνολική Σύγκριση - Εφαρμοσιμότητα

Ανάμεσα στα κριτήρια που παρουσιάστηκαν, υπάρχουν αρκετοί συσχετισμοί. Για παράδειγμα, οι σελίδες που βασίζονται σε κάποιο πρότυπο (template) έχουν μεγαλύτερο βαθμό αυτοματοποίησης από αυτές που δεν βασίζονται σε πρότυπα και από τα έγγραφα με ελεύθερο κείμενο, καθώς οι είσοδοι στην πρώτη περίπτωση παρουσιάζουν ένα δομημένο πλαίσιο εργασίας, το οποίο μπορούν να το ανακαλύψουν οι μη-επιβλεπόμενες προσεγγίσεις. Το γεγονός αυτό όμως δε σημαίνει ότι η εξαγωγή από σελίδες με πρότυπο είναι ευκολότερη από ότι σε άλλες σελίδες. Αντίθετα, στις περιπτώσεις αυτές εμφανίζονται νέα προβλήματα, όπως ο διαχωρισμός μεταξύ των λεκτικών μονάδων που ανήκουν στο πρότυπο και αυτών που αποτελούν δεδομένα, και η τοποθέτηση ετικετών στα δεδομένα.

Όπως φαίνεται στην Εικόνα 2.2-12, τα χειρωνακτικά συστήματα ΕΠ μπορούν να εφαρμοστούν σε όλα τα είδη εισόδων, με την προϋπόθεση ότι παρέχονται από τα συστήματα τα κατάλληλα χαρακτηριστικά, αν και η σύνθεση των κανόνων εξαγωγής εξαρτάται αποκλειστικά από τις τεχνικές του προγραμματιστή. Τα ημι-επιβλεπόμενα και μη-επιβλεπόμενα συστήματα μπορούν να εφαρμοστούν μόνο σε σελίδες βασισμένες σε κάποιο πρότυπο, καθώς η επιτυχία τους εξαρτάται από την ύπαρξη προτύπου. Επιπλέον, φαίνεται ότι τα μη-επιβλεπόμενα συστήματα συνήθως εφαρμόζουν πιο επιφανειακά γνωρίσματα, όπως τα HTML tags, για τη δημιουργία των κανονικών εκφράσεών τους, καθώς απευθύνονται σε σελίδες βασισμένες σε πρότυπα. Για την εξαγωγή πληροφορίας από πολλές διαφορετικές σελίδες του διαδικτύου όμως, απαιτούνται σημασιολογικά χαρακτηριστικά (π.χ. ορθογραφία, μήκος λεκτικής μονάδας κλπ.) καθώς δεν υπάρχουν τόσες κοινές επιγραφές και λέξεις μεταξύ των εγγράφων εισόδου.

Σε σχέση λοιπόν με την εφαρμοσιμότητα των διαφόρων συστημάτων ΕΠ, μπορούμε να πούμε ότι γενικά τα ημι-επιβλεπόμενα και τα μη-επιβλεπόμενα συστήματα χρησιμοποιούν ευριστικές τεχνικές που προέρχονται από παρατήρηση των σελίδων βασισμένων σε πρότυπα, όπως συναφείς και μη συναφείς εγγραφές δεδομένων, και εμφωλευμένα αντικείμενα. Λόγω της μεγάλης ποικιλίας εγγράφων στο διαδίκτυο όμως, δεν υπάρχει εγγύηση για το ποιες τεχνικές είναι περισσότερο αποτελεσματικές, παρόλο που φαίνεται πως οι νεότερες τεχνικές μπορούν να εφαρμοστούν σε περισσότερες σελίδες από ότι οι παλαιότερες. Από την άλλη, οι επιβλεπόμενες τεχνικές παρουσιάζουν πολύ μεγαλύτερη εφαρμοσιμότητα, καθώς τα προς εξαγωγή δεδομένα σημαδεύονται από τους χρήστες, χωρίς όμως και πάλι να υπάρχει κάποια εγγύηση για την επιτυχία των κανόνων εξαγωγής τους.



Εικόνα 2.2-12

Παρακάτω παρουσιάζονται τρεις πίνακες όπου αναφέρονται συνοπτικά τα χαρακτηριστικά όλων των συστημάτων που αναφέρθηκαν στην ενότητα αυτή, αναφορικά με το πεδίο του χώρου εργασίας, τις χρησιμοποιούμενες τεχνικές και το βαθμό αυτοματοποίησής τους.

Πίνακας 2.2-1 Κατάταξη με βάση το πεδίου του χώρου εργασίας

	Τύπος		Επίπεδο	Παραλλαγές αντικειμ. Εξαγωγής		
Σύστημα	σελίδας	NHS	εξαγωγής	ΑΓ/ΓΠΤ	ΓΠΔ	Εμφωλευμένα
Minerva	ημι-δομ.	ΝΑΙ	εγγραφής	ΝΑΙ	ΝΑΙ	ΝΑΙ
TSIMMIS	ημι-δομ.	ΝΑΙ	εγγραφής	ΝΑΙ	ΟΧΙ	ΝΑΙ
WebOQL	ημι-δομ.	ΟΧΙ	εγγραφής	ΝΑΙ	ΝΑΙ	ΝΑΙ
W4F	template	ΟΧΙ	εγγραφής	ΝΑΙ	ΝΑΙ	ΝΑΙ
WTL	ελεύθ.κειμ.	ΝΑΙ	εγγραφής	ΝΑΙ	ΟΧΙ	ΝΑΙ
RAPIER	ελεύθ.κειμ.	ΝΑΙ	πεδίου	ΝΑΙ	-	-
SRV	ελεύθ.κειμ.	ΝΑΙ	πεδίου	ΝΑΙ	-	-
STALKER	ημι-δομ.	ΝΑΙ	εγγραφής	ΝΑΙ	ΝΑΙ	ΝΑΙ
IEPAD	template	Περιορ.	εγγραφής	ΝΑΙ	Περιορ.	Περιορ.
OLERA	template	Περιορ.	εγγραφής	ΝΑΙ	Περιορ.	Περιορ.
DeLa	template	Περιορ.	εγγραφής	ΝΑΙ	Περιορ.	ΝΑΙ
RoadRunner	template	Περιορ.	σελίδας	ΝΑΙ	ΟΧΙ	ΝΑΙ
EXALG	template	Περιορ.	σελίδας	ΝΑΙ	ΟΧΙ	ΝΑΙ

Πίνακας 2.2-2 Κατάταξη με βάση τις χρησιμοποιούμενες τεχνικές

	Περάσματα	Τύπος κανόνα	Χαρακτηριστικά	Κωδικοποίηση
Σύστημα	σελίδας	εξαγωγής		λεκτ.ανάλυσης
Minerva	απλό	κανονικές εκφρ.	HTML tags/λέξεις κειμένου	με το χέρι
TSIMMIS	απλό	κανονικές εκφρ.	HTML tags/λέξεις κειμένου	με το χέρι
WebOQL	απλό	κανονικές εκφρ.	Υπερδέντρο	με το χέρι
W4F	απλό	κανονικές εκφρ.	διευθ.μονοπατιού DOM	επιπέδου επιγραφής
WTL	απλό	κανονικές εκφρ.	HTML tags/λέξεις κειμένου	με το χέρι
RAPIER	πολλαπλό	κανονικές εκφρ.	Συντακτικά/σημασιολογικά	επιπέδου λέξης
SRV	πολλαπλό	κανονικές εκφρ.	Συντακτικά/σημασιολογικά	επιπέδου λέξης
STALKER	πολλαπλό	κανονικές εκφρ.	HTML tags/λέξεις κειμένου	επιπέδου λέξης
IEPAD	απλό	κανονικές εκφρ.	HTML tags	πολλαπλών επιπέδων
OLERA	απλό	κανονικές εκφρ.	HTML tags	πολλαπλών επιπέδων
DeLa	απλό	κανονικές εκφρ.	HTML tags	επιπέδου επιγραφής
RoadRunner	απλό	κανονικές εκφρ.	HTML tags	επιπέδου επιγραφής
EXALG	απλό	κανονικές εκφρ.	HTML tags/λέξεις κειμένου	ΝΑΙ

Πίνακας 2.2-3 Κατάταξη με βάση το βαθμό αυτοματοποίησης

Σύστημα	Εμπειρία χρήστη	Ανάκτηση σελίδων	Μορφή εξόδου	Εφαρμοσ/τα	Περιορισμοί
Minerva	προγραμματισμός	ΟΧΙ	XML	υψηλή	ΟΧΙ
TSIMMIS	προγραμματισμός	ΟΧΙ	κείμενο	υψηλή	ΟΧΙ
WebOQL	προγραμματισμός	ΟΧΙ	κείμενο	υψηλή	ΟΧΙ
W4F	προγραμματισμός	ΝΑΙ	XML	μέτρια	ΟΧΙ
WTL	προγραμματισμός	ΝΑΙ	πλειάδες	υψηλή	ΟΧΙ
RAPIER	τοποθ. ετικετών	ΟΧΙ	κείμενο	μέτρια	ΟΧΙ
SRV	τοποθ. ετικετών	ΟΧΙ	κείμενο	μέτρια	ΟΧΙ
STALKER	τοποθ. ετικετών	ΟΧΙ	κείμενο	μέτρια	ΟΧΙ
IEPAD	τοποθ. ετικετών μετά,	ΟΧΙ	κείμενο	χαμηλή	σελίδα πολλαπλών
	επιλογή μοτίβων				εγγραφών
OLERA	μερική τοπ. ετικετών	ΟΧΙ	XML	χαμηλή	ΟΧΙ
DeLa	επιλογή μοτίβων	ΝΑΙ	κείμενο	χαμηλή	σελίδα πολλαπλών
					εγγραφών, >1 σελίδες
RoadRunner	επιλογή μοτίβων	ΝΑΙ	XML	χαμηλή	>1 σελίδες
EXALG	επιλογή μοτίβων	ΟΧΙ	κείμενο	χαμηλή	>1 σελίδες

2.3 RSS Feeds

Το RSS είναι μια σχετικά νέα μέθοδος που χρησιμοποιείται στο διαδίκτυο για τη δημοσίευση περιεχομένου το οποίο ανανεώνεται συχνά, όπως είναι τα άρθρα ενός blog, οι τίτλοι ειδήσεων και τα podcasts, από μία τοποθεσία σε πολλές άλλες. Τα αρχικά RSS σημαίνουν ***R**eally **S**imple **S**yndication*, δηλαδή «Πραγματικά Απλό Συνδικάτο», καθώς ουσιαστικά το RSS αποτελεί μια οικογένεια τυποποιήσεων για το σχήμα αυτό των δεδομένων το οποίο δίνει τη δυνατότητα στους χρήστες του διαδικτύου να έχουν πρόσβαση με εύκολο και γρήγορο τρόπο σε όλες τις πληροφορίες που τους ενδιαφέρουν. Τα έγγραφα RSS, τα οποία καλούνται "*feeds*" ή "*web feeds*" ή "*channels*", περιέχουν είτε μια περίληψη του περιεχομένου ενός ιστότοπου ή ακόμη και το πλήρες κείμενο. Ουσιαστικά, είναι έγγραφα μορφοποίησης XML, όπως XML/RDF ή Atom. Ένα παράδειγμα ενός πολύ απλού RSS εγγράφου είναι αυτό που φαίνεται στην παρακάτω εικόνα:

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<rss version="2.0">
<channel>
  <title>W3Schools Home Page</title>
  <link>http://www.w3schools.com</link>
  <description>Free web building tutorials</description>
  <item>
    <title>RSS Tutorial</title>
    <link>http://www.w3schools.com/rss</link>
    <description>New RSS tutorial on W3Schools</description>
  </item>
  <item>
    <title>XML Tutorial</title>
    <link>http://www.w3schools.com/xml</link>
    <description>New XML tutorial on W3Schools</description>
  </item>
</channel>
</rss>
```

Εικόνα 2.3-1 Παράδειγμα RSS εγγράφου (πηγή: δικτυακός τόπος w3schools.com - http://www.w3schools.com/rss/rss_syntax.asp).

Το περιεχόμενο RSS μπορεί να αναγνωστεί με τη βοήθεια κατάλληλου λογισμικού, του *προγράμματος ανάγνωσης RSS* (“*RSS aggregator*” ή “*RSS reader*” ή “*feed reader*”). Ο χρήστης εισάγει το σύνδεσμο για κάθε RSS feed που τον ενδιαφέρει στον aggregator (ή πιο εύκολα πατά πάνω στο εικονίδιο του RSS από τη σελίδα του ενδιαφέροντός του – βλ. Εικόνα 2.3-2), και γίνεται έτσι «συνδρομητής» στο περιεχόμενο της αντίστοιχης σελίδας. Από το σημείο αυτό και μετά, ο aggregator αναλαμβάνει να ανανεώνεται κατά τακτικά χρονικά διαστήματα με το νέο περιεχόμενο των feeds του χρήστη, ώστε πλέον ο χρήστης να μπορεί απλά ανοίγοντας το πρόγραμμα ανάγνωσης να βλέπει τις τελευταίες καταχωρήσεις των σελίδων του ενδιαφέροντός του.

Εικόνα 2.3-2 Το εικονίδιο του RSS

Το RSS γεννήθηκε το 1997, με τη δημιουργία του scriptingNews από τον Dave Winer. Ήδη, το 1999 αναπτύχθηκε το RSS 0.90 από τη Netscape, το οποίο ακολούθησε το RSS 0.91 την ίδια χρονιά, και στη συνέχεια το 2000 τα RSS 1.0, από μια ομάδα στο O’Reilly με επικεφαλής τον Rael Dornfest και RSS 0.92, από τον Dave Winer. Τέλος το 2003 κυκλοφόρησαν οι επίσημές προδιαγραφές του RSS 2.0, πάλι από τον Dave Winer. Τα αρχικά RSS χρησιμοποιούνται ως αναφορά στα παρακάτω:

- Really Simple Syndication (RSS 2.0)
- RDF Site Summary (RSS 1.0 RSS 0.90)
- Rich Site Summary (RSS 0.91)

Τελευταία, όλο και περισσότερες τοποθεσίες στο διαδίκτυο χρησιμοποιούν το RSS και παράλληλα όλο και περισσότεροι χρήστες εκτιμούν το είδος της υπηρεσίας αυτής. Έτσι, πρακτορεία ειδήσεων, blogs, σελίδες ανακοινώσεων Πανεπιστημιακών σχολών, εξυπηρετητές ηλεκτρονικού ταχυδρομείου αλλά και τοποθεσίες με μετεωρολογικές προβλέψεις ή με νέα για το χρηματιστήριο διαθέτουν πλέον τα κατάλληλα feeds (πολλές φορές και περισσότερα από ένα για κάθε τοποθεσία, ανάλογα με τις διαφορετικές κατηγορίες ενδιαφέροντος που περιέχουν), στα οποία μπορούν πολύ εύκολα να γίνουν συνδρομητές οι χρήστες.

Ένα ακόμη βασικό πλεονέκτημα των RSS εγγράφων είναι και η ευκολία με την οποία μπορούν να υποστούν λεκτική και συντακτική ανάλυση, εξαιτίας της XML δομής τους, διευκολύνοντας έτσι ιδιαίτερα την εξαγωγή πληροφορίας από αυτά. Έτσι, πολλές φορές είναι σκόπιμο ως είσοδος σε ένα σύστημα ΕΔΠ να δοθεί το RSS feed μιας ιστοσελίδας παρά ο ίδιος ο HTML κώδικας της σελίδας αυτής.

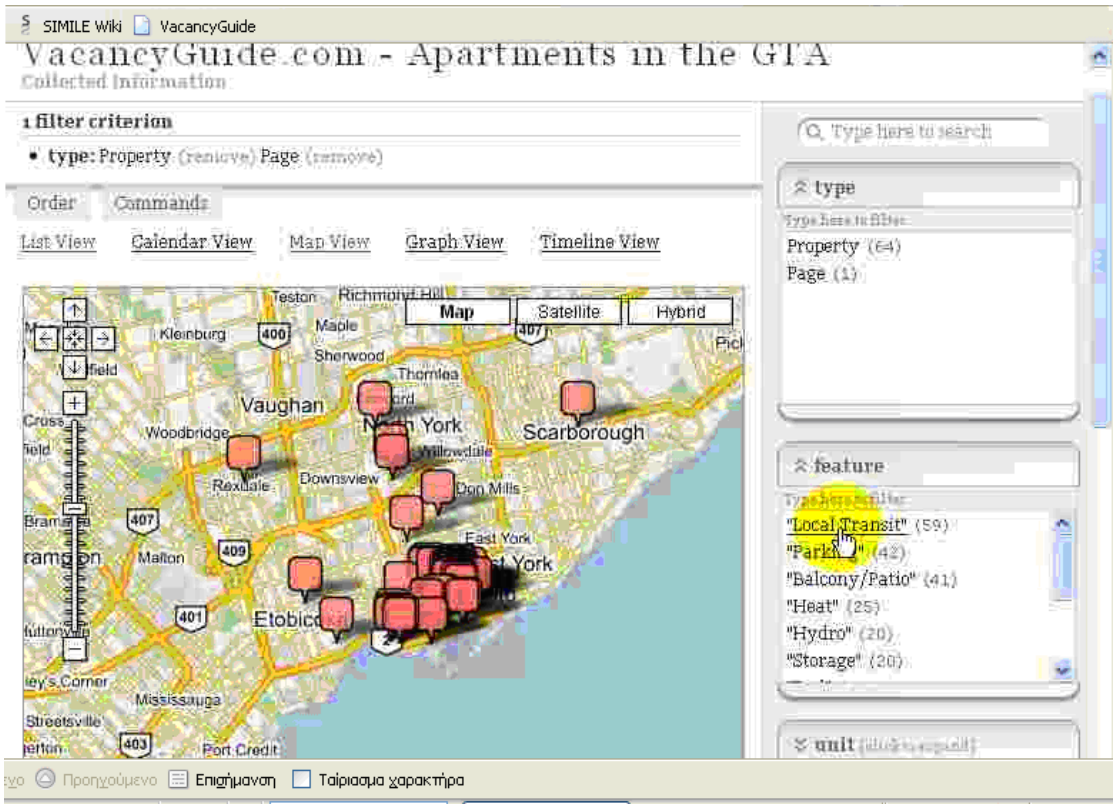
2.4 Web scraping

Η ταχεία και ευρεία διάδοση της XML (Extended Markup Language) και των διαδικτυακών υπηρεσιών (web services) έχει οδηγήσει στη δημιουργία τεχνολογιών που βελτιώνουν τη διαδικασία της εξαγωγής δεδομένων «φιλικών προς τις μηχανές» από ιστοσελίδες. Μάλιστα, ένας από τους βασικότερους στόχους του *Σημασιολογικού Ιστού* (Semantic Web) είναι να επιτρέψει τη δημιουργία εγγράφων τα οποία θα είναι εύκολο να διαβαστούν τόσο από ανθρώπους όσο και από μηχανές. Η διαδικτυακή απόξεση (*web scraping* ή *harvesting*, όπως αποκαλείται καμιά φορά) εξυπηρετεί ακριβώς αυτό το σκοπό, καθώς αποτελεί ένα σύνολο μεθόδων εξαγωγής περιεχομένου από έναν ιστότοπο μέσω του πρωτοκόλλου HTTP με σκοπό τη μετατροπή του σε μια άλλη μορφή, κατάλληλη να χρησιμοποιηθεί σε κάποιο άλλο πλαίσιο εργασίας. Μια τυπική εφαρμογή του web scraping αποτελούν οι web crawlers, οι οποίοι αντιγράφουν περιεχόμενο από μία ή περισσότερες υπάρχουσες ιστοσελίδες και το ενσωματώνουν σε μια άλλη τοποθεσία, η οποία χαρακτηρίζεται *scraper site*. Αυτό μπορεί να γίνει είτε με «αγνές προθέσεις», όπως για διαλογή μόνο της χρήσιμης πληροφορίας από κάποια τοποθεσία και συνάθροισή της με αντίστοιχες πληροφορίες από άλλες τοποθεσίες, ή και κακόβουλα (λογοκλοπή), όπου αντιγράφεται σκόπιμα περιεχόμενο από διάφορες ιστοσελίδες, πολλές φορές χωρίς καμία λογική συνάφεια, με σκοπό την αύξηση της επισκεψιμότητας μιας ιστοθέσης (μέσω των διαδικτυακών μηχανών αναζήτησης) και άρα της αύξησης των εσόδων της μέσω των διαφημίσεων. Το web scraping συχνά αναφέρεται και ως data ή page ή HTML scraping ή ακόμα και screen scraping. Από τους όρους αυτούς, ο τελευταίος είναι γενικότερος και αρχικά είχε την έννοια μιας τεχνικής για την πραγματοποίηση από κάποιο πρόγραμμα υπολογιστή (του screen scraper) εξαγωγής δεδομένων από την οθόνη εξόδου ενός άλλου προγράμματος. Στην πραγματικότητα, διαφέρει από το web scraping με την έννοια ότι το πρώτο εργάζεται πάνω στην οπτική απεικόνιση των δεδομένων, ενώ το δεύτερο ασχολείται με τη δομή των αντικειμένων (μοντέλο DOM) της HTML και της JavaScript.

Πολλά από τα σύγχρονα προγραμματιστικά εργαλεία έχουν πλέον ενσωματωμένα χαρακτηριστικά και βιβλιοθήκες που επιτρέπουν τη δημιουργία εφαρμογών που πραγματοποιούν web scraping. Μεταξύ αυτών είναι οι γλώσσες Perl, PHP, Java, Python και Ruby, η υλοποίηση των διαδικτυακών υπηρεσιών από τη Microsoft όπως και διάφορα περιβάλλοντα Unix.

Μια πολύ ενδιαφέρουσα εφαρμογή που χρησιμοποιεί web scraping είναι η επέκταση “Piggy Bank” για τον Mozilla Firefox. Σκοπός της επέκτασης αυτής είναι να δώσει λύσει σε κάποιους περιορισμούς που θέτει το διαδίκτυο, το οποίο δε δίνει τη δυνατότητα συγχώνευσης πληροφοριών από διαφορετικές τοποθεσίες χωρίς να πρέπει ο χρήστης να τις αντιγράψει σε κάποια άλλη εφαρμογή, πέρα βέβαια από το εργαλείο των σελιδοδεικτών, στο οποίο όμως αποθηκεύονται απλώς οι τίτλοι και οι διευθύνσεις των τοποθεσιών. Το Piggy Bank ακολουθεί το πνεύμα της πρωτοβουλίας του Σημασιολογικού Ιστού, η οποία έχει καθορίσει ένα μοντέλο για την αναπαράσταση των δεδομένων στο διαδίκτυο που επιτρέπουν την ευκολότερη «ανάμειξή» αυτών και την επαναχρησιμοποίησή τους με βάση τις ανάγκες του κάθε χρήστη, άσχετα με την προέλευσή τους ή την αρχική μορφοποίησή τους. Το μοντέλο αυτό είναι το *RDF (Resource Description Framework)* και τα δεδομένα που αναπαριστώνται με τη βοήθειά του χαρακτηρίζονται ως «*καθαρά*» δεδομένα. Συγκεκριμένα, το Piggy Bank εξάγει πληροφορία από υπάρχουσες ιστοσελίδες και την αποθηκεύει σε μορφή RDF, είτε απευθείας- αν η σελίδα συνδέεται με πληροφορία RDF- είτε με τη βοήθεια κάποιου web scraper που αναλαμβάνει να ξεχωρίσει την «καθαρή πληροφορία» από τη σελίδα. Στη συνέχεια, ο χρήστης μπορεί μέσα από το φυλλομετρητή του και από τη διεπαφή χρήστη της επέκτασης να έχει πρόσβαση στην πληροφορία που έχει εξάγει, ανεξάρτητα από την αρχική σελίδα. Ακόμη, το Piggy Bank μπορεί να προσφέρει μια ενοποιημένη όψη της «καθαρής» πληροφορίας από πολλές διαφορετικές πηγές. Μπορεί ακόμη και να χρησιμοποιήσει την υπηρεσία χαρτών Google Maps για την αναπαράσταση γεωγραφικής πληροφορίας, ακόμη και αν καμία τέτοια δυνατότητα δεν προσφερόταν από τις αρχικές ιστοσελίδες. Ένα τέτοιο παράδειγμα χρήσης του Piggy Bank, για την εύρεση κατοικιών στην περιοχή του Toronto και για την εμφάνισή τους σε χάρτη ανάλογα και με τα χαρακτηριστικά που παρουσιάζουν, φαίνεται στην Εικόνα 2.4-1.

Οι screen scrapers (όπως αποκαλούνται οι web scrapers στο Piggy Bank) που χρησιμοποιούνται είναι διαφορετικοί για τις διαφορετικές ιστοσελίδες και ο χρήστης μπορεί είτε να φορτώσει στο σύστημά του κάποιον υπάρχοντα scraper είτε να δημιουργήσει ο ίδιος έναν. Μάλιστα το Piggy Bank υποτίθεται ότι είναι και πιο ανθεκτικό στις συνεχείς αλλαγές των ιστοσελίδων, καθώς η εξαγωγή των δεδομένων δε γίνεται στο συντακτικό επίπεδο με την εφαρμογή κανονικών εκφράσεων στη σειριακή αναπαράσταση του DOM δέντρου της HTML σελίδας, όπως γίνεται συνήθως, αλλά στο επίπεδο μοντέλου, επιτρέποντας την εφαρμογή εκφράσεων xpath.


SIMILE Wiki
VacancyGuide
VacancyGuide.com - Apartments in the GTA
Collected Information
1 filter criterion
type: Property (remove) Page (remove)
Order
Commands
List View
Calendar View
Map View
Graph View
Timeline View
Map
Satellite
Hybrid
Vaughan
Scarborough
Woodbridge
Maple
Sherwood
Thornlea
Downsview
Don Mills
East York
Mississauga
Port Credit
Type here to search
type
Property (64)
Page (1)
feature
"Local Transit" (59)
"Balcony/Patio" (41)
"Heat" (25)
"Hydro" (20)
"Storage" (20)
unit
Προηγούμενο
Επιφάνεια
Ταίριασμα χαρακτήρα

Εικόνα 2.4-1

3 *Θεωρητικό Υπόβαθρο*

Στο παρόν κεφάλαιο αναφέρονται κάποια μοντέλα, συστήματα εργαλεία και γλώσσες προγραμματισμού που χρησιμοποιήθηκαν στην ανάπτυξη του συστήματος, και που η κατανόησή τους από τον αναγνώστη κρίνεται χρήσιμη πριν την παρουσίαση της ανάλυσης και της σχεδίασης του συστήματος. Αρχικά γίνεται μια αναφορά στους φυλλομετρητές (web browsers) και συγκεκριμένα στον Mozilla Firefox, του οποίου επέκταση αποτελεί το σύστημά μας. Ακολουθεί μια συνοπτική παρουσίαση του τρόπου οργάνωσης των αρχείων μέσα σε μία επέκταση (extension) του Firefox και του ρόλου κάθε αρχείου. Στη συνέχεια δίνεται μια σύντομη περιγραφή της γλώσσας σήμανσης XUL και της scripting γλώσσας JavaScript που χρησιμοποιήθηκαν σε μεγάλο βαθμό για την ανάπτυξη του συστήματος. Ακολούθως, γίνεται αναφορά στο μοντέλο DOM (Document Object Model) στο οποίο βασίζεται η οργάνωση των βασικών στοιχείων παρουσίασης του συστήματος, όπως και στα CSS, που αποτελούν την γλώσσα περιγραφής του τρόπου παρουσίασης των στοιχείων αυτών. Τέλος, γίνεται αναφορά στις Κανονικές Εκφράσεις, οι οποίες αποτελούν βασικό εργαλείο του συστήματος μας.

3.1 Φυλλομετρητές – Mozilla Firefox

3.1.1 Εισαγωγή

Ένας *φυλλομετρητής ιστοσελίδων* ή αλλιώς *πρόγραμμα περιήγησης Ίντερνετ* ή *περιήγησης Ιστού* (*web browser*) είναι μια εφαρμογή η οποία παρέχει στο χρήστη εύκολη και γρήγορη πρόσβαση σε ιστοσελίδες που βρίσκονται αναρτημένες στο διαδίκτυο. Οι ιστοσελίδες που μπορεί να παρουσιάσει ο φυλλομετρητής αποτελούν υπερκείμενα μορφοποιημένα σύμφωνα με τη γλώσσα μορφοποίησης HTML ή XHTML και μπορεί να ενσωματώνουν εικόνες (στατικές ή κινούμενες), ήχο και άλλα στοιχεία, αν μπορεί να τα παρουσιάσει ο web browser. Επιπλέον, το κείμενο και οι εικόνες μπορεί να περιέχουν υπερσυνδέσμους προς άλλες ιστοσελίδες στην ίδια ή σε διαφορετικές τοποθεσίες. Έτσι, «διασχίζοντας» τους συνδέσμους

αυτούς, οι web browsers επιτρέπουν στο χρήστη να έχει πρόσβαση με εύκολο τρόπο σε πληροφορία που παρέχεται από πολλές ιστοσελίδες σε πολλούς ιστότοπους. Οι web browsers ουσιαστικά αποτελούν λογισμικό πελάτη του πρωτοκόλλου HTTP (HyperText Transfer Protocol).

Μερικοί από τους φυλλομετρητές που είναι διαθέσιμοι για τους προσωπικούς υπολογιστές είναι οι Internet Explorer, Mozilla Firefox, Apple Safari και Opera, με φθίνουσα σειρά δημοτικότητας (κατά το Νοέμβριο του 2007). Παρόλο που οι browsers τυπικά χρησιμοποιούνται για την πρόσβαση στον Παγκόσμιο Ιστό, μπορούν να χρησιμοποιηθούν επίσης για πρόσβαση σε πληροφορία που παρέχεται από εξυπηρετητές διαδικτύου (web servers) σε ιδιωτικά δίκτυα ή συστήματα αρχείων.

Εικόνα 3.1-1 Internet Explorer, Mozilla Firefox, Apple Safari, Opera.

3.1.2 Χρησιμοποιούμενα Πρωτόκολλα

Οι web browsers επικοινωνούν με τους web servers πρωταρχικά μέσω της χρήσης του πρωτοκόλλου HTTP για την ανάκτηση των ιστοσελίδων. Το πρωτόκολλο αυτό επιτρέπει επίσης στους φυλλομετρητές και την υποβολή πληροφορίας στους web servers. Οι σελίδες εντοπίζονται μέσω ενός URL (Uniform Resource Locator), το οποίο χρησιμοποιείται ως διεύθυνση και το οποίο ξεκινά με “http:” για πρόσβαση μέσω του HTTP. Πολλοί από τους browsers υποστηρίζουν αρκετούς ακόμη τύπους URL και τα αντίστοιχα πρωτόκολλα, όπως “gopher:” για το πρωτόκολλο Gopher (ένα ιεραρχικό πρωτόκολλο υπερσυνδέσμων), “ftp:” για το πρωτόκολλο FTP (File Tranfer Protocol), “rtsp:” για το πρωτόκολλο RTSP (Real-Time Streaming Protocol) και “https:” για το πρωτόκολλο HTTPS (μια κρυπτογραφημένη εκδοχή του HTTP). Μερικοί από τους πιο διάσημους φυλλομετρητές υποστηρίζουν και επιπλέον πρωτόκολλα, όπως τα NTTP (Network News Transfer Protocol), SMTP (Simple Mail Transfer Protocol), IMAP (Internet Message Access Protocol) και POP (Post Office Protocol). Αυτοί οι browsers συχνά αναφέρονται και ως Internet suites ή application suites. Για πολλούς web browsers διατίθενται, επίσης, και αρκετά πρόσθετα (add-ons ή extensions), με στόχο την επαύξηση των δυνατοτήτων τους, τη βελτίωση της χρηστικότητάς τους καθώς και την προστασία του χρήστη σε θέματα ασφάλειας.

3.1.3 Ο Mozilla Firefox

Ο *Mozilla Firefox* (συντομευμένα *Fx* και, πιο ανεπίσημα, *FF*) είναι ένας web browser ο οποίος προέρχεται από το Πακέτο Εφαρμογών Mozilla του Οργανισμού Mozilla. Τον Ιανουάριο του 2008, ο

Firefox είχε ποσοστό χρήσης περίπου 15% σε σχέση με τους άλλους web browsers, και ήταν δεύτερος σε δημοτικότητα, μετά τον Internet Explorer της Microsoft. Ο Firefox χρησιμοποιεί τη μηχανή ανοιχτού κώδικά Gecko, η οποία υλοποιεί κάποια πρόσφατα πρότυπα του διαδικτύου, καθώς και κάποια χαρακτηριστικά που στοχεύουν στην επίσπευση πιθανών επιπλέον προσθηκών στα πρότυπα. Τα χαρακτηριστικά του Firefox περιλαμβάνουν την περιήγηση με καρτέλες (tabs), ένα εργαλείο ορθογραφικού ελέγχου, αυξητική αναζήτηση, απευθείας δημιουργία σελιδοδεικτών (live bookmarking), έναν download manager και ένα σύστημα αναζήτησης που χρησιμοποιεί το Google. Επιπλέον λειτουργίες μπορεί να προστεθούν μέσα από περισσότερα από 2000 πρόσθετα που έχουν αναπτυχθεί από τρίτους. Ο Firefox μπορεί να «τρέξει» σε διάφορες εκδόσεις των λειτουργικών συστημάτων Windows, Mac OS X, Linux και σε πολλά άλλα λειτουργικά συστήματα της μορφής Unix. Η τελευταία του «σταθερή» εκδοχή που κυκλοφορεί είναι η 2.0.0.13 (κυκλοφόρησε το Μάρτιο του 2008), ενώ από τις 10 Μαρτίου 2008 κυκλοφορεί και η τέταρτη δοκιμαστική έκδοση του Firefox 3. Ο πηγαίος κώδικας του Firefox αποτελεί ελεύθερο λογισμικό.

3.2 Mozilla Firefox Extensions

Τα πρόσθετα ή add-ons είναι μικρά προγράμματα που μπορεί να εγκαταστήσει ο χρήστης για να προσθέσει χαρακτηριστικά ή να βελτιστοποιήσει τον Firefox. Μπορούν να προσθέσουν νέες μηχανές αναζήτησης ή ξενόγλωσσα λεξικά, να αλλάξουν την εμφάνιση του Firefox ή να πραγματοποιήσουν πολλές άλλες λειτουργίες. Τα πρόσθετα χωρίζονται σε πέντε υπο-κατηγορίες: τις επεκτάσεις (extensions), τα θέματα (themes), τα λεξικά, τις μηχανές αναζήτησης και τα πρόσθετα προγράμματα (plugins). Από αυτά, τα πρόσθετα προγράμματα βοηθούν τον περιηγητή να εκτελέσει συγκεκριμένες λειτουργίες όπως την εμφάνιση ειδικών τύπων γραφικών ή την αναπαραγωγή αρχείων πολυμέσων ενώ οι επεκτάσεις τροποποιούν ή προσθέτουν χαρακτηριστικά. Οι επεκτάσεις δηλαδή, προσθέτουν νέα λειτουργικότητα στον Firefox, ή οποία μπορεί να είναι ένα απλό κουμπί εργαλειοθήκης ως και ένα εντελώς νέο χαρακτηριστικό. Επιτρέπουν με αυτόν τον τρόπο στο χρήστη να προσαρμόσει τον φυλλομετρητή ανάλογα με τις δικές του ανάγκες και προτιμήσεις.

Οι επεκτάσεις συσκευάζονται και διανέμονται μέσω αρχείων ZIP, ή αλλιώς Bundles, με την κατάληξη xpi (προφέρεται “zippy”). Η δομή του περιεχομένου του XPI αρχείου έχει ως εξής:

```
extension.xpi:
              /install.rdf
              /components/*
              /components/cmdline.js
```

```
/defaults/
/defaults/preferences/*.js
/chrome.manifest
/chrome/icons/default/*
/chrome/
/chrome/content/
```

3.2.1 Αρχείο Install.rdf

Το αρχείο install.rdf είναι ένα αρχείο κειμένου το οποίο περιέχει μετα-δεδομένα (δεδομένα που δίνουν πληροφορίες στην εφαρμογή στην οποία απευθύνεται η επέκταση – στην περίπτωσή μας στο Mozilla Firefox) σχετικά με την επέκταση. Το δεδομένα αυτά ανήκουν σε 3 κατηγορίες:

- Πληροφορίες που χρησιμεύουν στο μοναδικό προσδιορισμό της επέκτασης από την εφαρμογή (όπως μια μοναδική προσδιοριστική συμβολοακολουθία - ID- και η έκδοσή της επέκτασης),
- Πληροφορίες χρήσιμες στο χρήστη (όπως το όνομά, μια περιγραφή της επέκτασης και οι συγγραφείς της), και τέλος
- Πληροφορίες συμβατότητας, (η εφαρμογή στην οποία απευθύνεται και ελάχιστη και μέγιστη έκδοση της εφαρμογής με την οποία είναι συμβατή η επέκταση).

3.2.2 Αρχείο chrome.manifest

Το αρχείο chrome.manifest είναι ένα αρχείο που ενημερώνει την εφαρμογή-στόχο για το πού βρίσκεται το πακέτο συσκευασίας των αρχείων chrome για τη συγκεκριμένη επέκταση. Το *chrome* είναι το σύνολο των στοιχείων της διεπαφής χρήστη τα οποία βρίσκονται έξω από την περιοχή του περιεχομένου του παραθύρου της εφαρμογής, όπως είναι οι γραμμές εργαλείων, οι γραμμές κατάστασης, οι γραμμές των μενού κλπ. Μάλιστα, οι προγραμματιστές της Mozilla έχουν δημιουργήσει μια νέα κατηγορία URI, τα “chrome://” URI, μέσω των οποίων μπορεί να φορτώνονται εύκολα τα αρχεία chrome, έτσι ώστε η εγκατεστημένη εφαρμογή να γνωρίζει πώς να βρει αυτά τα αρχεία και πώς να τα χειριστεί.

Πιο συγκεκριμένα, στο αρχείο chrome.manifest καταχωρούνται πληροφορίες πέντε ειδών:

- overlay: Ένα overlay («επίστρωμα») επιτρέπει την προσθήκη νέου περιεχομένου σε ένα ήδη υπάρχον έγγραφο της εφαρμογής, όπως για παράδειγμα στο αρχείο ορισμού του βασικού παραθύρου του Firefox. Με τον τρόπο αυτόν λοιπόν δηλώνονται τα έγγραφα που πρόκειται να τροποποιηθούν, και τα έγγραφα της επέκτασης που περιέχουν τις τροποποιήσεις που θα γίνουν (αρχεία XUL)
- content: Πρόκειται για τον κατάλογο όπου περιέχονται όλα τα XUL αρχεία που ορίζουν τα περιεχόμενα των παραθύρων και των διαλόγων της επέκτασης, τα JavaScript αρχεία που καθορίζουν τη λειτουργικότητα της διεπαφής χρήστη, όπως και τα XBL αρχεία, αν υπάρχουν, τα οποία καθορίζουν τη συμπεριφορά και την εμφάνιση των XUL στοιχείων.

- locale: Εδώ προσδιορίζεται ο κατάλογος που περιέχει όλα τα αρχεία (αρχεία .dtd και .properties) για τον «εν-τοπισμό» της επέκτασης, δηλαδή αρχεία όπου περιέχονται οι μεταφράσεις των μηνυμάτων που απευθύνονται προς το χρήστη σε διάφορες γλώσσες (για τις γλώσσες που υποστηρίζονται από την επέκταση, αν υποστηρίζονται)
- skin: Εδώ προσδιορίζεται ο κατάλογος των διάφορων εικόνων που χρησιμοποιούνται και των αρχείων που περιέχουν όλους τους προσδιορισμούς σχετικά με την εμφάνιση των διαφόρων στοιχείων της διεπαφής χρήστη (αρχεία css).
- override: Με τον τρόπο αυτό ορίζεται η αντικατάσταση ενός αρχείου chrome της εφαρμογής με ένα νέο αρχείο.

3.2.3 Κατάλογος αρχείων chrome

Ο κατάλογος αυτός αποτελείται από τρεις υποκαταλόγους, content, locale και skin, οι οποίοι αντιστοιχούν στα είδη των αρχείων που περιγράφηκαν παραπάνω, στις αντίστοιχες καταχωρήσεις στο manifest αρχείο. Από αυτούς, μόνο ο υποκατάλογος content είναι υποχρεωτικό να υπάρχει, καθώς περιέχει όλο το περιεχόμενο της επέκτασης (αρχεία chrome).

3.2.4 Κατάλογος αρχείων default

Εδώ περιέχονται τα αρχεία που ορίζουν και αρχικοποιούν τις μεταβλητές των προτιμήσεων του χρήστη (preferences). Συγκεκριμένα, οι αρχικοποιήσεις αυτές γίνονται σε ένα αρχείο JavaScript (κατάληξη .js) ώστε να φορτωθούν αυτόματα από το σύστημα προτιμήσεων του Firefox όταν αυτός ξεκινήσει. Η ύπαρξη αυτού του καταλόγου είναι προαιρετική.

3.2.5 Κατάλογος αρχείων components

Εδώ περιέχονται .js (JavaScript) και .dll αρχεία στα οποία ορίζονται στοιχεία XPCOM (Cross Platform Component Object Model). Το XPCOM είναι ένα μοντέλο αντικειμένων ανεξάρτητο από πλατφόρμες που επιτρέπει τη δημιουργία νέων αντικειμένων από το χρήστη, τα οποία μπορούν να επικοινωνούν με άλλα αντικείμενα. Η δημιουργία και η διαχείριση των αντικειμένων μπορεί να γίνει σε JavaScript, Java, Python ή C++.

3.3 Η γλώσσα XUL

Η XUL (XML-based User Interface Language) είναι μια γλώσσα η οποία έχει αναπτυχθεί από τους προγραμματιστές της Mozilla και είναι βασισμένη στη δομή της XML. Η XUL (προφέρεται “zool”) χαρακτηρίζεται «γλώσσα διεπαφής χρήστη», γιατί επιτρέπει τη δημιουργία εφαρμογών πλούσιων σε χαρακτηριστικά και ανεξάρτητων από πλατφόρμες, οι οποίες μπορούν να τρέχουν είτε υπάρχει σύνδεση

με το διαδίκτυο είτε όχι. Όλα τα στοιχεία που αποτελούν τη διεπαφή χρήστη στον Firefox και τις επεκτάσεις του (παράθυρα, διάλογοι, γραμμές εργαλείων, μενού αλλά και τα επιμέρους στοιχεία αυτών όπως κουμπιά,, λίστες, ετικέτες κλπ), όπως και σε άλλες εφαρμογές της Mozilla, είναι γραμμένα σε XUL

Το μεγάλο πλεονέκτημα της XUL είναι ότι είναι ανεξάρτητη από την πλατφόρμα (λειτουργικό σύστημα) που χρησιμοποιείται, γεγονός που κάνει τις εφαρμογές που γράφονται σε αυτή μεταφέρσιμες σε οποιοδήποτε σύστημα. Επίσης, η XUL έχει όλα τα πλεονεκτήματα των γλωσσών XML. Για παράδειγμα μπορούν να εμφωλευτούν σε αυτήν στοιχεία από άλλες γλώσσες XML όπως XHTML, MathML ή SVG. Ακόμη, ισχύει η ιεραρχία DOM και γι' αυτό μπορούν εύκολα να δημιουργηθούν και να τροποποιηθούν δυναμικά στοιχεία XUL με τη βοήθεια αρχείων JavaScript ή ακόμα να γίνει συντακτική ανάλυση (parsing) των .xul αρχείων. Τέλος, το κείμενο το οποίο εμφανίζεται με τη βοήθεια της XUL μπορεί αρκετά εύκολα να μεταφερθεί σε άλλες γλώσσες.

Η XUL ορίζει μια ευρεία γκάμα στοιχείων, που μπορούν χονδρικά να κατηγοριοποιηθούν ως εξής:

- Στοιχεία ανώτερου επιπέδου, π.χ. window, page, dialog, wizard κλπ.
- Widgets, π.χ. label, button, textbox, listbox, menulist, menupopup, radiogroup, checkbox, tree, menu, toolbar, groupbox, tabbox, colorpicker, spacer, splitter κλπ.
- Box Model, π.χ. box, grid, stack, deck κ.λ.π.
- Events και Scripts, π.χ. script, command, key, broadcaster, observer κ.λ.π.
- Πηγές δεδομένων, π.χ. template, rule κ.λ.π.
- Άλλα, π.χ. overlay, iframe, browser, editor, κ.λ.π.

3.4 Η γλώσσα JavaScript

Η JavaScript είναι μια ελαφριά, αντικειμενοστραφής scripting γλώσσα, ανεξάρτητη από την χρησιμοποιούμενη πλατφόρμα, η οποία συνήθως χρησιμοποιείται για ανάπτυξη client-side εφαρμογών. Το επίσημό της όνομα είναι ECMAScript. Η Javascript έχει επηρεαστεί από πολλές γλώσσες προγραμματισμού και έχει σχεδιαστεί να μοιάζει αρκετά με τη Java, αλλά να είναι πιο εύκολο να εργαστούν μαζί της οι προγραμματιστές. Αρχικά αναπτύχθηκε από τον Brendan Eich στη Netscape με το όνομα Mocha, στη συνέχεια μετονομάστηκε σε LiveScript και τελικά σε JavaScript. Για πρώτη φορά παρουσιάστηκε και ενσωματώθηκε στον περιηγητή της Netscape το 1995. Η τελευταία έκδοση της JavaScript που κυκλοφορεί είναι η 1.7

Η JavaScript υποστηρίζει τη σύνταξη δομημένου προγραμματισμού της C (π.χ. εντολές if, switch, βρόγχοι while κλπ.). Όπως συμβαίνει με τις περισσότερες scripting γλώσσες, οι μεταβλητές της δε δεσμεύονται σε συγκεκριμένους τύπους (κάτι τέτοιο συμβαίνει μόνο με τις τιμές). Ακόμη, τα αντικείμενα

σε αυτή αντιμετωπίζονται ως συσχετιζόμενοι πίνακες, έτσι ώστε τα ονόματα των γνωρισμάτων των αντικειμένων να αποτελούν κλειδιά για τους πίνακες. Άλλα χαρακτηριστικά της JavaScript είναι ότι τα αντικείμενά της δε βασίζονται σε κλάσεις αλλά σε «πρωτότυπα», ότι δε γίνεται διαχωρισμός μεταξύ των συναρτήσεων και των μεθόδων (όπως συμβαίνει συνήθως στις αντικειμενοστραφείς γλώσσες) παρά μόνο κατά την κλήση μιας συνάρτησης, ότι υποστηρίζει κανονικές εκφράσεις, και ότι χρησιμοποιεί μεταφραστή (ο οποίος αποκαλείται «μηχανή») αντί για μεταγλωττιστή.

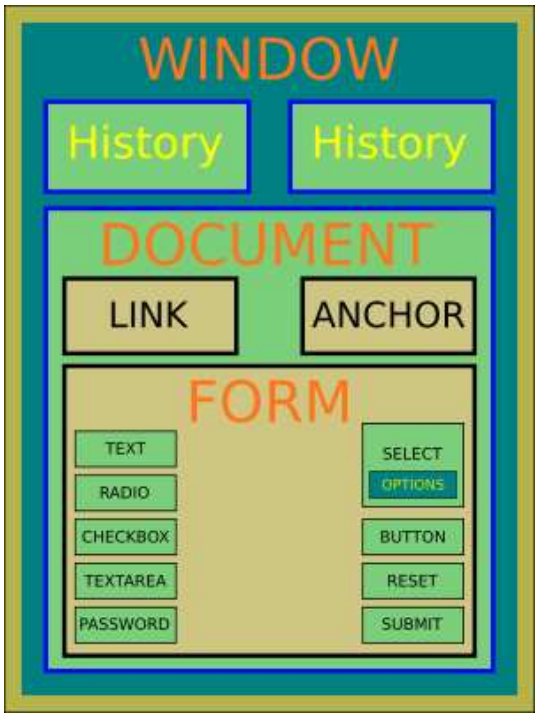
Όπως αναφέρθηκε, η βασικότερη χρήση της JavaScript είναι στη συγγραφή συναρτήσεων που βρίσκονται ενσωματωμένες σε HTML σελίδες και αλληλεπιδρούν με το μοντέλο DOM της σελίδας. Επειδή μάλιστα ο κώδικάς της μπορεί να τρέχει τοπικά στον browser του χρήστη (παρά σε κάποιον απομακρυσμένο εξυπηρετητή), ανταποκρίνεται πολύ γρήγορα στις ενέργειες του χρήστη, κάνοντας την εκάστοτε εφαρμογή να φαίνεται ιδιαίτερα αποκρίσιμη. Επιπλέον, ο κώδικας της JavaScript μπορεί να ανιχνεύσει τις ενέργειες του χρήστη, όπως τις μεμονωμένες πιέσεις πλήκτρων, κάτι που η HTML δεν μπορεί να κάνει από μόνη της. Αυτό το γεγονός το εκμεταλλεύεται αρκετά και η σύγχρονη τάση προγραμματισμού σε Ajax.

Η JavaScript χρησιμοποιείται στις επεκτάσεις του Mozilla Firefox για να παρέχει λειτουργικότητα κατά το «χρόνο εκτέλεσης» και αλληλεπίδραση με το χρήστη. Χρησιμοποιείται επίσης για την αποθήκευση των προτιμήσεων του χρήστη (preferences) και για τη δημιουργία στοιχείων XPCOM.

3.5 Document Object Model (DOM)

Το Document Object Model (DOM) είναι πρότυπο μοντέλο αντικειμένων για την αναπαράσταση της HTML, της XML και άλλων σχετικών γλωσσών, το οποίο είναι ανεξάρτητο της γλώσσας προγραμματισμού και της πλατφόρμας που χρησιμοποιείται. Ένας web browser δεν είναι υποχρεωτικό να χρησιμοποιήσει το DOM για να αποδώσει ένα HTML έγγραφο. Παρόλα αυτά, το DOM είναι απαραίτητο στα scipts της JavaScript όπου γίνεται δυναμική επιθεώρηση ή τροποποίηση μιας ιστοσελίδας. Με λίγα λόγια το DOM είναι ο τρόπος που βλέπει η JavaScript τη σελίδα HTML που την περιέχει και την κατάσταση του browser. Στην Εικόνα 3.5-1 φαίνεται η ιεραρχία των αντικειμένων για ένα παράδειγμα HTML DOM.Η ανάπτυξη του Document Object Model ξεκίνησε στα μέσα της δεκαετίας του ’90 από το World Wide Web Consortium (WWWC – W3C), τον κύριο διεθνή οργανισμό προτύπων για τον Παγκόσμιο Ιστό. Η τρέχουσα έκδοση του DOM είναι η DOM 3, η οποία κυκλοφορεί από τον Απρίλιο του 2004. Η υποστήριξη ενός κοινού σχήματος DOM από τους διάφορους web browsers έχει λύσει πολλά προβλήματα διαλειτουργικότητας που υπήρχαν παλαιότερα σε σχέση με τις διαφορετικές ιστοσελίδες στο διαδίκτυο.

Η ιεραρχία DOM χρησιμοποιείται στην ανάπτυξη των επεκτάσεων του Firefox για τη «δόμηση» των στοιχείων που συνθέτουν τον περιηγητή μέσω της XUL, καθώς επίσης για την προσπέλαση των στοιχείων αυτών μέσω της JavaScript.



Εικόνα 3.5-1

3.6 CSS Stylesheets

Τα *Cascading Style Sheets (CSS)* είναι μια γλώσσα μορφοποίησης σελίδων στο διαδίκτυο, η οποία χρησιμοποιείται για την παρουσίαση ενός εγγράφου γραμμένου σε μια γλώσσα σήμανσης (markup language), όπως είναι η XML, η XUL, η HTML ή η XHTML. Με τη βοήθεια των CSS μπορούν να οριστούν τα χρώματα, οι γραμματοσειρές και η δομή του εγγράφου, όπως και άλλα χαρακτηριστικά του γενικότερου στυλ του. Ο κύριος λόγος που αναπτύχθηκε η γλώσσα αυτή ήταν για να διαχωριστεί το περιεχόμενο ενός εγγράφου, το οποίο γράφεται σε HTML ή σε μια παρόμοια γλώσσα σήμανσης, από τον τρόπο παρουσίασής του, που γράφεται σε CSS. Ο διαχωρισμός αυτός μπορεί να βελτιώσει την προσβασιμότητα του περιεχομένου, να παρέχει μεγαλύτερη ευλυγισία και έλεγχο στον προσδιορισμό των χαρακτηριστικών παρουσίασης, και να μειώσει την πολυπλοκότητα και την επανάληψη στο δομημένο περιεχόμενο, καθώς οι κανόνες παρουσίασης χρειάζεται να γραφτούν μόνο μία φορά για κάθε επιθυμητή ομάδα στοιχείων με κοινό στυλ, ακόμα και για πολλαπλές σελίδες κάποιας δικτυακής τοποθεσίας (όταν τα stylesheets γράφονται σε εξωτερικά .css αρχεία, με τα οποία συνδέονται όλες οι σελίδες). Τα CSS επιτρέπουν ακόμη την παρουσίαση της ίδιας markup σελίδας με διαφορετικά στυλ για διαφορετικές μεθόδους απόδοσης του περιεχομένου, όπως εμφάνιση στην οθόνη, εκτύπωση σε χαρτί, απαγγελία (απαγγελία του περιεχομένου της σελίδας από έναν περιηγητή βασισμένο στην ομιλία ή από ένα πρόγραμμα ανάγνωσης της οθόνης) και παρουσίαση σε συσκευές ψηλάφησης, τύπου Braille. Τα CSS ορίζουν ένα σχήμα προτεραιότητας για να καθορίσουν ποιοι κανόνες μορφοποίησης εφαρμόζονται όταν σε ένα στοιχείο ταιριάζουν περισσότεροι από έναν κανόνες. Σε αυτή τη διαδοχική ακολουθία (*cascade*,

όπως αποκαλείται), υπολογίζονται οι προτεραιότητες ή αλλιώς βάρη και ανατίθενται στους κανόνες, έτσι ώστε τα αποτελέσματα να είναι προβλέψιμα. Οι προδιαγραφές για τα CSS έχουν οριστεί από το W3C, ενώ ο τύπος πολυμέσων διαδικτύου (MIME type) “text/css” έχει καταχωρηθεί για χρήση από τα Cascading Style Sheets.

Η σύνταξη των CSS είναι απλή και χρησιμοποιεί ένα σύνολο λέξεων-κλειδιά στα αγγλικά για να προσδιορίσει τα ονόματα διαφόρων γνωρισμάτων μορφοποίησης. Ένα φύλλο μορφοποίησης (*stylesheet*) αποτελείται από ένα σύνολο *κανόνων*. Κάθε κανόνας ή σύνολο κανόνων αποτελείται από έναν ή περισσότερους *επιλογείς* (*selectors*) και ένα *μπλοκ δηλώσεων*. Παρακάτω φαίνεται η γενική μορφή ενός κανόνα:

selectors

{

property1 : *value1*;

.....

propertyN: *valueN*;

}

Οι επιλογείς είναι κριτήρια σύμφωνα με τα οποία επιλέγονται τα στοιχεία στα οποία εφαρμόζεται ένας κανόνας. Ένας επιλογέας μπορεί να είναι το όνομα ενός τύπου στοιχείων (π.χ. το στοιχείο “img” της HTML), ή η τιμή για ένα συγκεκριμένο γνώρισμα (attribute) κάποιων στοιχείων, η τιμή του ειδικού γνωρίσματος “class” που χρησιμοποιείται για την εφαρμογή διαφορετικών στυλ ακόμη και σε στοιχεία του ίδιου τύπου, η τιμή του ειδικού γνωρίσματος “id” που χρησιμοποιείται για το μοναδικό προσδιορισμό ενός συγκεκριμένου στοιχείου, ή κάποια ειδικά προσδιοριστικά που σχετίζονται με τη σχετική θέση των στοιχείων στο έγγραφο ή τον τρόπο με τον οποίο βρίσκονται αυτά εμφωλευμένα στο Document Object Model. Τέλος, ως επιλογέας μπορεί να χρησιμοποιηθεί και ένας συνδυασμός κάποιων από τα παραπάνω χαρακτηριστικά. Επιπλέον όλων αυτών, μπορεί να χρησιμοποιηθεί και ένα σύνολο ψευδο-κλάσεων για να καθορίσει επιπλέον συμπεριφορές, όπως τον τρόπο παρουσίασης στοιχείων στα οποία «δείχνει» ο χρήστης με το ποντίκι του, ή στοιχείων-συνδέσμων τους οποίους έχει προηγουμένως επισκεφθεί ο χρήστης.

Τα CSS χρησιμοποιούνται και στις επεκτάσεις του φυλλομετρητή Mozilla Firefox, για να ορίσουν τους κανόνες παρουσίασης για τα αντικείμενα που ορίζονται στα .xul και .js αρχεία (μενού, παράθυρα, κουμπιά, ετικέτες κλπ.).

3.7 Κανονικές Εκφράσεις

Οι *Κανονικές Εκφράσεις (Regular Expressions)* αποτελούν έναν συνοπτικό και ευέλικτο τρόπο για τον προσδιορισμό συμβολοακολουθιών που ενδιαφέρουν, όπως συγκεκριμένους χαρακτήρες, λέξεις ή μοτίβα

χαρακτήρων. Οι Κανονικές εκφράσεις ήταν πάντα ένα ισχυρό εργαλείο στα χέρια των διαχειριστών συστημάτων Unix και στη συνέχεια των προγραμματιστών της γλώσσας Perl, ενώ πρόσφατα, οι δυνατότητες αυτές ενσωματώθηκαν και στη JavaScript, και κατ' επέκταση στους web browsers Netscape Navigator και Internet Explorer, και αργότερα στο Mozilla Firefox.

Οι κανονικές εκφράσεις (για συντομία *regexes* ή *regexps* ή *regexen*) γράφονται σε μια επίσημη γλώσσα η οποία μπορεί να διερμηνευθεί από έναν *επεξεργαστή κανονικών εκφράσεων*, ένα πρόγραμμα που είτε χρησιμεύει ως γεννήτρια λεκτικών αναλυτών (parsers) είτε εξετάζει ένα κείμενο και προσδιορίζει τα τμήματά του που ταιριάζουν με τις προδιαγραφές που έχουν δοθεί. Πέρα από τον προσδιορισμό του αν μια συμβολοακολουθία (string) ταιριάζει με ένα δοσμένο μοτίβο, με τη βοήθεια των κανονικών εκφράσεων μπορεί επίσης να προσδιοριστεί η θέση της υπο-συμβολοακολουθίας που πετυχαίνει το ταίριασμα, μέσα στο αρχικό string ή να γίνει αντικατάσταση του τμήματος που ταίριαξε, με κάποιο άλλο string, ή ακόμα να γίνει διαχωρισμός της αρχικής συμβολοακολουθίας σε τμήματα, τα οποία διαχωρίζονται μεταξύ τους από τα susbstrings που πέτυχαν ταίριασμα με το μοτίβο. Κάθε κανονική έκφραση ορίζει ένα μοτίβο, το οποίο μπορεί να αποτελείται από σύνολα επιθυμητών (ή και ανεπιθύμητων) χαρακτήρων, από ακολουθίες χαρακτήρων, από ποσοτικοποιητές (που υποδηλώνουν πόσες φορές θέλουμε να επαναληφθεί κάποιο υπο-μοτίβο) και άλλους ειδικούς χαρακτήρες που δηλώνουν αν κάποιο μοτίβο θέλουμε να βρίσκεται στην αρχή ή στο τέλος της συμβολοακολουθίας εισόδου, να βρίσκεται στο όριο μιας λέξης ή μιας γραμμής , να συνοδεύει (ή να μη συνοδεύει) υποχρεωτικά κάποιο άλλο μοτίβο κ.α.

Οι κανονικές εκφράσεις χρησιμεύουν στην παραγωγή συστημάτων highlighting (επεξεργαστές κειμένου που μορφοποιούν διαφορετικά τις διάφορες κατηγορίες όρων σε ένα κώδικα γραμμένο σε κάποια γλώσσα προγραμματισμού), στην επικύρωση δεδομένων και ιδιαίτερα στην εξαγωγή πληροφορίας. Στο σύστημα UranuS, οι κανονικές εκφράσεις έπαιξαν σημαντικό ρόλο, ως κομμάτι της JavaScript, για την εξαγωγή των επιθυμητών μετεωρολογικών δεδομένων από τις ιστοσελίδες-πηγές.

4

Ανάλυση Απαιτήσεων Συστήματος

Το σύστημα UranuS αποτελεί μια επέκταση του φυλλομετρητή Mozilla Firefox για τη δυναμική ενημέρωση του χρήστη για μετεωρολογικά δεδομένα και προγνώσεις καιρού για κάποια περιοχή της επιλογής του. Η ενημέρωση του χρήστη θα γίνεται μέσω ενός «*παραθύρου πρόγνωσης*» το οποίο θα μπορεί να εμφανίσει ο ίδιος όποια στιγμή το επιθυμεί με την κατάλληλη εντολή του μενού, αλλά και μέσω της γραμμής κατάστασης του παραθύρου του περιηγητή, όπου θα εμφανίζεται διαρκώς ένα μέρος των δεδομένων. Τα δεδομένα θα πρέπει να ανανεώνονται κατά τακτά χρονικά διαστήματα, ενώ ο χρήστης θα πρέπει να μπορεί να απαιτήσει και την ενημέρωση των δεδομένων όποια στιγμή το επιθυμήσει αυτός. Το σύστημα θα πρέπει να επιτρέπει στο χρήστη να δημιουργήσει και να αποθηκεύσει διαφορετικά προφίλ χρήσης. Για κάθε προφίλ, ο χρήστης θα μπορεί να ορίσει την πηγή των δεδομένων (συγκεκριμένη ιστοσελίδα), να επιλέξει τα χαρακτηριστικά του καιρού που τον ενδιαφέρουν, και να διαμορφώσει τον ακριβή τρόπο παρουσίασης των δεδομένων σε αυτόν. Θα μπορεί ακόμη, αν το επιθυμεί, να ορίσει τον τρόπο εξαγωγής των δεδομένων, δηλαδή τις συγκεκριμένες κανονικές εκφράσεις που θα χρησιμοποιηθούν για την εξαγωγή της επιθυμητής πληροφορίας. Τέλος, θα πρέπει δίνεται η δυνατότητα στο χρήστη να χρησιμοποιήσει για δική του διευκόλυνση ένα προκαθορισμένο προφίλ το οποίο θα χρησιμοποιεί σταθερά ως πηγή δεδομένων το δικτυακό τόπο freemeteo.com, και για το οποίο θα χρειάζεται να ορίσει μόνο την περιοχή, τα χαρακτηριστικά του καιρού και κάποια στοιχεία παρουσίασης των δεδομένων, όλα μέσα από ένα προκαθορισμένο σύνολο επιλογών. Όλες αυτές οι επιλογές που περιγράφηκαν θα πρέπει να προσφέρονται στο χρήστη μέσα από το βασικό μενού της επέκτασης, καθώς επίσης και μέσα από ένα μενού προσαρτημένο στη γραμμή κατάστασης, για γρηγορότερη πρόσβαση.

Στο κεφάλαιο αυτό ακολουθεί η ανάλυση απαιτήσεων για τη λειτουργία του συστήματος και η περιγραφή της αρχιτεκτονικής του.

4.1 Μετεωρολογικό Μοντέλο Συστήματος

Το μετεωρολογικό μοντέλο για τη συγκεκριμένη εφαρμογή περιλαμβάνει τα εξής στοιχεία :

4.1.1 Πόλη – Περιοχή

Η περιοχή ή η πόλη δίνεται με ένα όνομα (πχ. *Θεσσαλονίκη, Λάρισα, Άμστερνταμ, Μακεδονία, Νότιο Αιγαίο* κ.α.). Φυσικά, ο αριθμός των πόλεων και των περιοχών είναι άπειρος, γι' αυτό και θα πρέπει κάθε φορά να προσδιορίζει ο χρήστης ποια πόλη - περιοχή τον ενδιαφέρει

4.1.2 Θερμοκρασία

Η θερμοκρασία παρουσιάζεται συνήθως ως ένας αριθμός με ένα ή δύο ακέραια ψηφία και, ενδεχομένως, ένα δεκαδικό ψηφίο, συνοδευόμενος από το σύμβολο *°C ή °F* ή σκέτο *F* ή *C*. Μπορεί επίσης να δίνεται και ένα εύρος θερμοκρασιών με τη μορφή *XX.X-YY.Y* και στη συνέχεια η αντίστοιχη μονάδα μέτρησης. Επειδή μπορεί να υπάρξουν και αρνητικές θερμοκρασίες, μπροστά από την τιμή της μπορεί να συναντήσουμε και το σύμβολο «-» (και μερικές φορές και το σύμβολο +).

Η θερμοκρασία μπορεί να δίνεται ανά πόλη, ανά ημέρα, ανά διάστημα της μέρας (ημέρα-νύχτα), ανά ένα χρονικό διάστημα κάποιων ωρών (πχ κάθε 6 ώρες), ή και σε συνδυασμούς των παραπάνω. Μπορεί επίσης να δίνεται και με τη μορφή «Υψηλή – Χαμηλή».

4.1.3 Άνεμος

Ο Άνεμος χαρακτηρίζεται από την ένταση και τη διεύθυνση. Η διεύθυνση του ανέμου δίνεται συνήθως με έναν από τους χαρακτηρισμούς *Β, Ν, Α, Δ, ΒΑ, ΒΔ, ΝΑ, ΝΔ, ΒΒΑ, ΒΒΔ, ΝΝΑ, ΝΝΔ, ΑΒΑ, ΑΝΑ, ΔΒΔ, ΔΝΔ* (και αντίστοιχα στα λατινικά *N, S, E, W, NE, NW, SE, SW, NNE, NNW, SSE, SSW, ENE, ESE, WNW, WSW*). Η ένταση δίνεται με έναν από τους ακόλουθους τρόπους

- δύο ακέραια ψηφία (και ενδεχομένως ένα δεκαδικό ψηφίο) συνοδευόμενα από τη μονάδα *km/h* ή *kph* ή σπανιότερα *χμω*
- δύο ακέραια ψηφία συνοδευόμενα από τη μονάδα *mph*
- ένα ακέραιο ψηφίο συνοδευόμενο από τη μονάδα *μποφόρ* ή *μποφώρ* (αντίστοιχα *beaufort*)

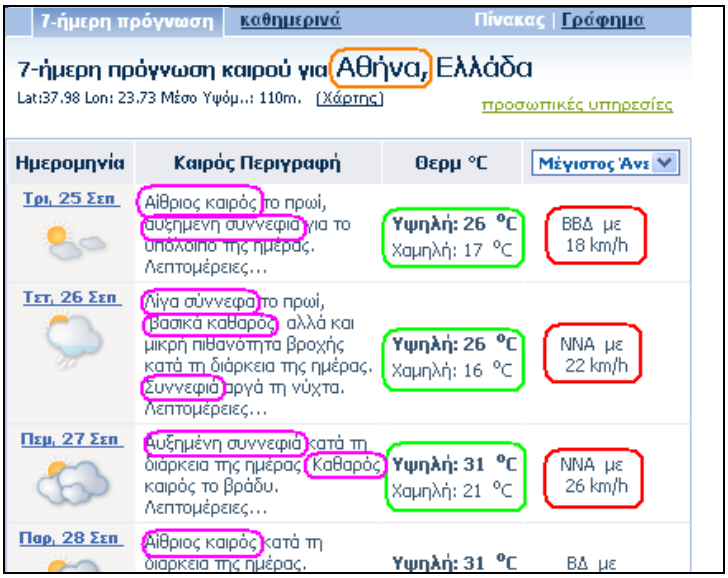
Ο άνεμος μπορεί να δίνεται ανά πόλη, ανά ημέρα, ανά διάστημα της μέρας (ημέρα-νύχτα), ανά ένα χρονικό διάστημα κάποιων ωρών (πχ κάθε 6 ώρες), ή και σε συνδυασμούς των παραπάνω.

4.1.4 Γενικός Χαρακτηρισμός καιρού

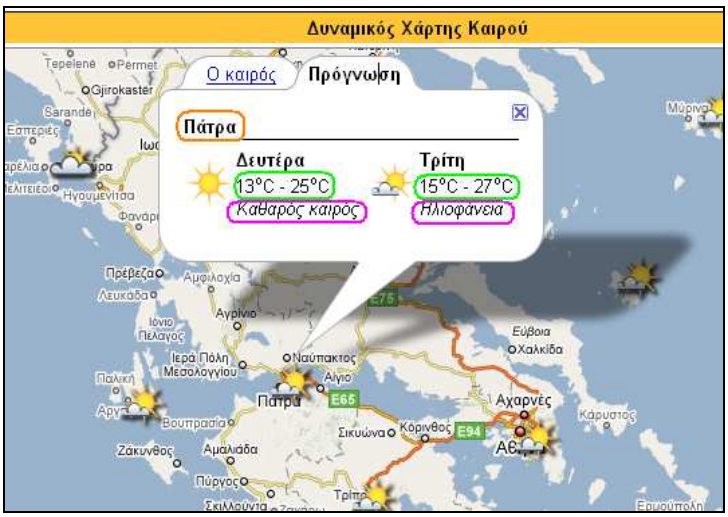
Ο γενικός χαρακτηρισμός του καιρού δίνεται με εκφράσεις όπως «αίθριος», «καθαρός», «λίγα σύννεφα», «συννεφιά», «ασθενείς βροχοπτώσεις», «βροχές», «καταιγίδες», «χαλαζόπτωση», «χιόνια» κλπ. Επειδή οι ακριβείς εκφράσεις του καιρού ποικίλλουν ανάλογα με την πηγή της μετεωρολογικής πρόγνωσης (δηλ. τη συγκεκριμένη ιστοσελίδα) είναι δύσκολο να καταγραφούν στο σύνολό τους.

Ο γενικός χαρακτηρισμός μπορεί να δίνεται ανά πόλη, ανά ημέρα, ανά διάστημα ημέρας (ημέρα- νύχτα) ή και σε συνδυασμούς των παραπάνω.

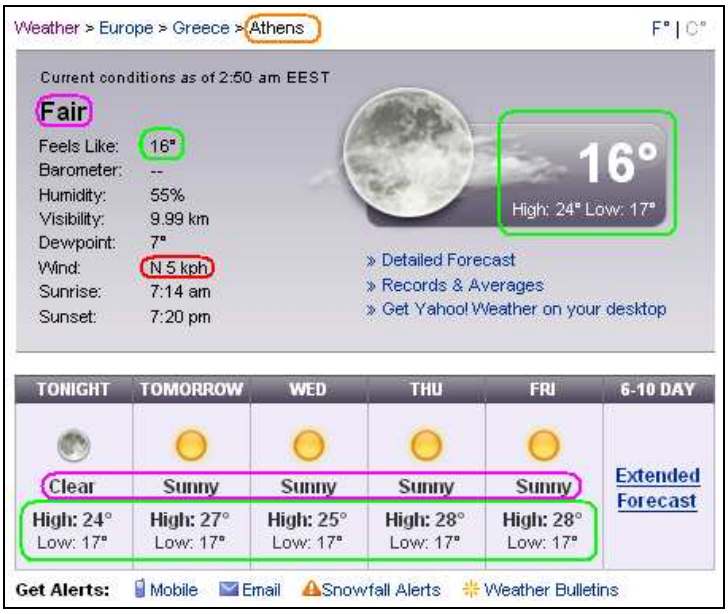
Στις Εικόνες 4.1-1 έως 4.1-5 παρουσιάζονται μερικά στιγμιότυπα από ιστοσελίδες πρόγνωσης καιρού, όπου σημειώνονται τα χαρακτηριστικά που αναφέρθηκαν παραπάνω. Συγκεκριμένα, η περιοχή παρουσιάζεται με πορτοκαλί χρώμα, η θερμοκρασία με πράσινο, ο άνεμος με κόκκινο και ο γενικός χαρακτηρισμός με μωβ χρώμα.



Εικόνα 4.1-1 Ο καιρός από το freemeteo.com

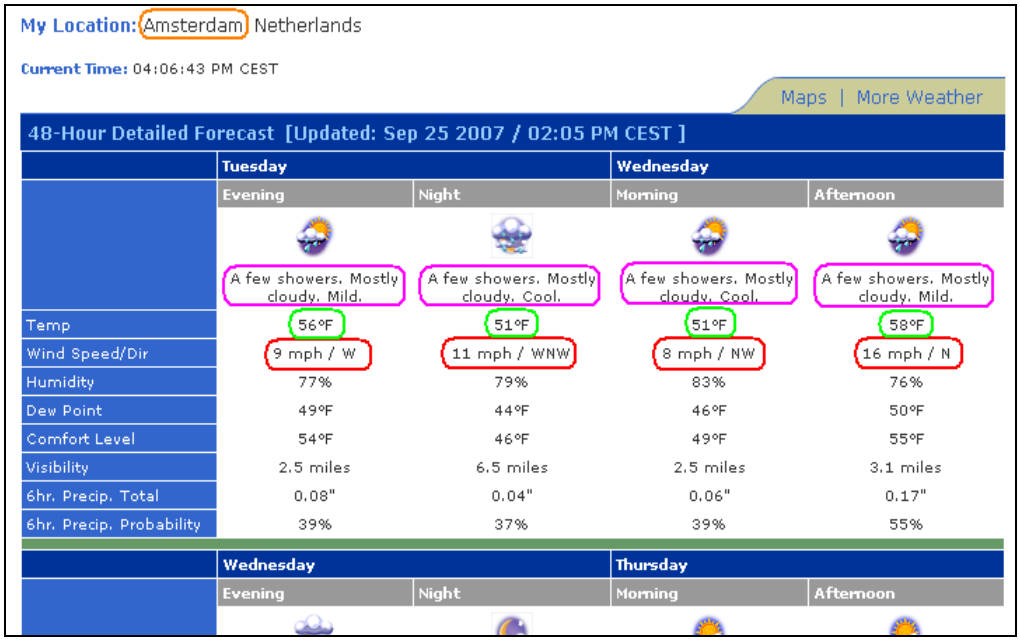


Εικόνα 4.1-2 Ο καιρός από το weather.robby.gr



Εικόνα 4.1-3 Ο καιρός από το weather.yahoo.com

Εικόνα 4.1-4 Ο καιρός από το www.meteo.gr



My Location: Amsterdam Netherlands

Current Time: 04:06:43 PM CEST

Maps | More Weather

48-Hour Detailed Forecast [Updated: Sep 25 2007 / 02:05 PM CEST]

	Tuesday		Wednesday	
	Evening	Night	Morning	Afternoon
	A few showers. Mostly cloudy. Mild.	A few showers. Mostly cloudy. Cool.	A few showers. Mostly cloudy. Cool.	A few showers. Mostly cloudy. Mild.
Temp	56°F	51°F	51°F	58°F
Wind Speed/Dir	9 mph / W	11 mph / WNW	8 mph / NW	16 mph / N
Humidity	77%	79%	83%	76%
Dew Point	49°F	44°F	46°F	50°F
Comfort Level	54°F	46°F	49°F	55°F
Visibility	2.5 miles	6.5 miles	2.5 miles	3.1 miles
6hr. Precip. Total	0.08"	0.04"	0.06"	0.17"
6hr. Precip. Probability	39%	37%	39%	55%

	Wednesday		Thursday	
	Evening	Night	Morning	Afternoon

Εικόνα 4.1-5 Ο καιρός από το www.myforecast.com

Επιπλέον των παραπάνω, στο ήδη έτοιμο προφίλ χρήστη που θα υπάρχει στο σύστημα, για λήψη μετεωρολογικών προγνώσεων από το freemeteo.com, θα περιλαμβάνονται δύο ακόμη στοιχεία :

4.1.5 Υγρασία

Η υγρασία παρουσιάζεται συνήθως ως ένας αριθμός με ένα ή δύο ακέραια ψηφία συνοδευόμενα από το σύμβολο %. Μπορεί επίσης να δίνεται και ένα εύρος υγρασίας με τη μορφή *XY-ZW%*.

4.1.6 Συνολικός Υετός

Ο υετός είναι ένα μέγεθος που εκφράζει κάθε πτώση ή εναπόθεση στο έδαφος προϊόντων του ύδατος (σε υγρή ή στερεά μορφή, επιμερισμένη) τα οποία προέρχονται από συμπύκνωση των υδρατμών της ατμόσφαιρας (βροχή, χιονόνερο, χαλάζι, χιόνι, ή πάγος που δημιουργείται όμως στο έδαφος). Οι παραπάνω μορφές ονομάζονται και (ατμοσφαιρικά) υδατώδη μετεωρολογικά κατακρημνίσματα, ή απλά κατακρημνίσματα, όταν αναφέρονται στη μετεωρολογία, καθώς ακόμη και υδρομετέωρα. Ο συνολικός υετός (total precipitation στα αγγλικά) μετράται με ένα ειδικό όργανο, το βροχόμετρο. Η τιμή του υετού δίνεται από έναν αριθμό με ένα ή δύο ακέραια και ένα δεκαδικό ψηφίο, συνοδευόμενα από τη μονάδα μέτρησης mm ή “″” ή “inches”, ενώ συχνά αναφέρεται ως ύψος βροχόπτωσης.

4.2 Χρήστης Συστήματος

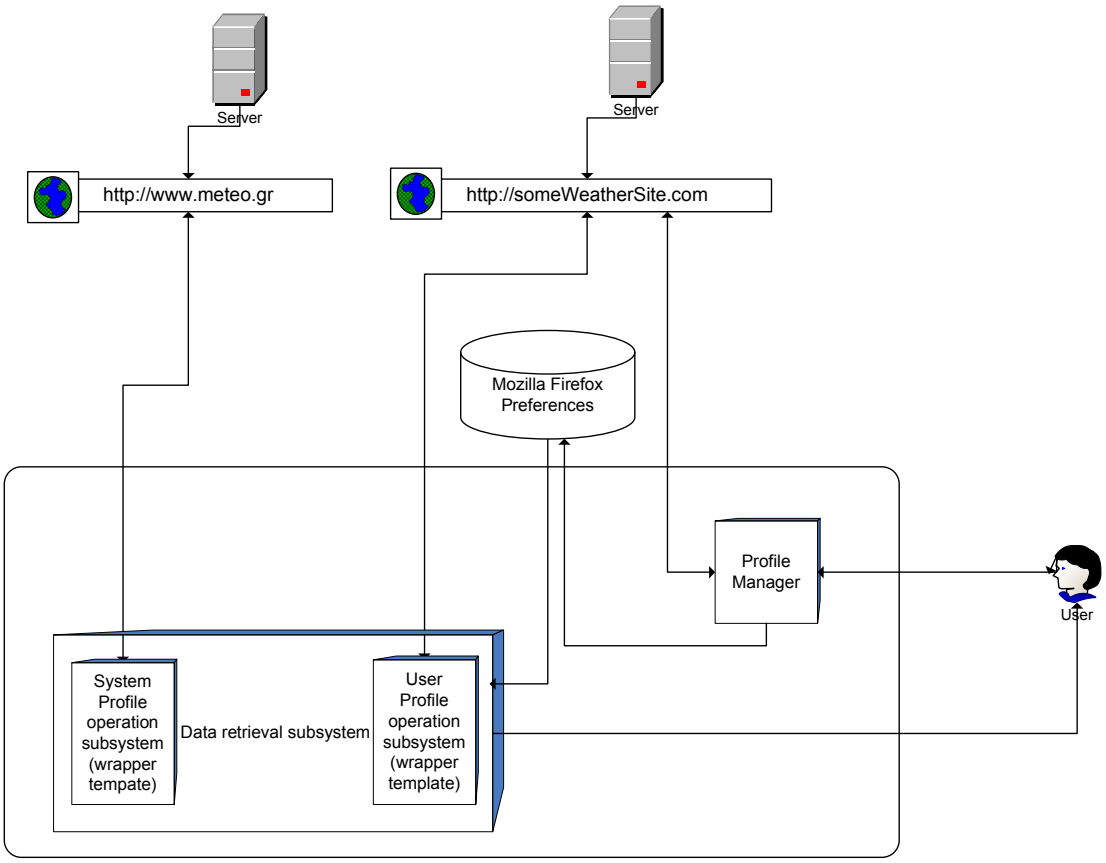
Ο χρήστης του συστήματος είναι ένας χρήστης του διαδικτύου, ο οποίος χρησιμοποιεί για την περιήγησή του στο διαδίκτυο το φυλλομετρητή Mozilla Firefox και ο οποίος έχει εγκαταστήσει το σύστημα στον υπολογιστή του.

4.3 Αρχιτεκτονική

Το σύστημά μας μπορούμε να θεωρήσουμε ότι αποτελείται από δύο υποσυστήματα:

- **Υποσύστημα διαχείρισης προφίλ:** Το υποσύστημα αυτό θα ασχολείται με τη δημιουργία νέων προφίλ και την επεξεργασία των ήδη υπαρχόντων από το χρήστη.
- **Υποσύστημα ανάκτησης δεδομένων:** Αυτό είναι το υποσύστημα που θα αναλαμβάνει να λάβει τα μετεωρολογικά δεδομένα και να τα παρουσιάσει στο χρήστη. Ουσιαστικά θα επιλέγει τον wrapper που θα χρησιμοποιηθεί για την εξαγωγή των δεδομένων και θα τον ενεργοποιεί. Αποτελείται από δύο επιμέρους υποσυστήματα:
    - **Υποσύστημα λειτουργίας προφίλ Χρήστη:** Κάνει την εξαγωγή των δεδομένων σύμφωνα με τις προτιμήσεις που έχουν οριστεί στο τρέχον προφίλ, όταν αυτό είναι κάποιο από τα προφίλ χρήστη. Περιλαμβάνει ένα πρότυπο wrapper (*wrapper template*) το οποίο προσαρμόζεται ανάλογα με τις προτιμήσεις του επιλεγμένου προφίλ
    - **Υποσύστημα λειτουργίας προφίλ συστήματος:** Κάνει την εξαγωγή των δεδομένων από την τοποθεσία freemeteo.com σύμφωνα με τις προτιμήσεις που έχουν οριστεί από το χρήστη για το προφίλ "*freemeteo*", όταν αυτό έχει επιλεγεί προς χρήση. Όπως και το υποσύστημα λειτουργίας προφίλ Χρήστη, περιλαμβάνει και αυτό ένα *wrapper template*.

Στα επόμενα, το προεπιλεγμένο προφίλ θα αναφέρεται και ως *«προφίλ συστήματος»*, ενώ τα ορισμένα από το χρήστη προφίλ θα αναφέρονται ως *«προφίλ χρήστη»*. Ο τρόπος με τον οποίο επικοινωνούν τα παραπάνω υποσυστήματα μεταξύ τους φαίνεται στο παρακάτω διάγραμμα:



4.4 *Περιγραφή Λειτουργιών*

Ακολουθεί η περιγραφή των λειτουργιών κάθε υποσυστήματος χωριστά.

4.4.1 Υποσύστημα διαχείρισης προφίλ χρήσης

Το υποσύστημα αυτό θα ενεργοποιείται όταν ο χρήστης ανοίξει το βασικό μενού του συστήματος ή όταν επιλέξει τις αντίστοιχες εντολές από το μενού της γραμμής κατάστασης. Οι λειτουργίες του θα είναι οι εξής:

- Δημιουργία νέου προφίλ χρήστη
- Τροποποίηση ήδη υπάρχοντος προφίλ χρήστη.
- Τροποποίηση προφίλ συστήματος "*freemeteo*".
- Επιλογή προφίλ προς χρήση.

Η τελευταία από αυτές είναι απλή και δεν αναλύεται περαιτέρω. Ακολουθεί αναλυτικότερη παρουσίαση των υπόλοιπων λειτουργιών με τη μορφή περιγραφών (σεναρίων) χρήσης.

4.4.1.1 Διαδικασία δημιουργίας νέου προφίλ χρήστη

Παρακάτω περιγράφεται η διαδικασία με την οποία ο χρήστης του συστήματος θα δημιουργεί ένα νέο προφίλ. Για τη δημιουργία του προφίλ θα πρέπει να χρησιμοποιείται ένα είδος «βοηθού», που θα αποτελείται από μία σειρά βημάτων. Κάθε φορά που ο χρήστης θα πατά "*Επόμενο*" θα μεταβαίνει στο επόμενο βήμα, ενώ πατώντας "*Προηγούμενο*" θα μπορεί να επιστρέψει στο αμέσως προηγούμενο βήμα και να αλλάξει τις προτιμήσεις του. Η διαδικασία θα ξεκινά όταν ο χρήστης επιλέξει από το βασικό μενού ή από το μενού της γραμμής κατάστασης να ορίσει ένα νέο προφίλ και θα αποτελείται από τα εξής βασικά βήματα:.

Βήμα 1

Αρχικά ο χρήστης θα καλείται να ανοίξει στον Mozilla Firefox τη συγκεκριμένη ιστοσελίδα από την οποία θέλει να λαμβάνει πληροφορίες.

Βήμα 2

Στη συνέχεια, θα λαμβάνεται το URL της σελίδας που άνοιξε ο χρήστης και θα εμφανίζεται σε ένα ειδικό πεδίο. Εδώ, θα πρέπει να δίνεται η δυνατότητα στο χρήστη να χρησιμοποιήσει τελικά ένα διαφορετικό URL φορτώνοντας μια νέα σελίδα στον περιηγητή και δίνοντας μια εντολή ανανέωσης.

Βήμα 3

Ακολούθως θα γίνεται ο προσδιορισμός των επιθυμητών χαρακτηριστικών καιρού και των προτιμήσεων της λεκτικής ανάλυσης της σελίδας, μέσα από ένα σύνολο επιλογών και ρυθμίσεων:

- Καταρχάς, ο χρήστης θα πρέπει να δώσει το όνομα της πόλης ή της περιοχής για την οποία θέλει να λαμβάνει μετεωρολογικές προγνώσεις, όπως θέλει να εμφανίζεται αυτό κατά την παρουσίαση των δεδομένων.
- Από τη λίστα των διαθέσιμων μετεωρολογικών χαρακτηριστικών (*θερμοκρασία*, *άνεμος*, *γενικός χαρακτηρισμός καιρού*) ο χρήστης θα μπορεί να επιλέξει αυτές που τον ενδιαφέρουν. Θα πρέπει μάλιστα να του δίνεται η δυνατότητα να δει τις διαθέσιμες κανονικές εκφράσεις για την εξαγωγή της πληροφορίας που αφορά καθένα από αυτά τα χαρακτηριστικά, ή να δώσει ενδεχομένως ο ίδιος την κατάλληλη κανονική έκφραση. Για το σκοπό αυτό απαιτείται να υπάρχουν τρεις ειδικές επιλογές, μία για κάθε χαρακτηριστικό του καιρού (βλ. βήμα 3α).
- Ο χρήστης θα μπορεί επίσης να επιλέξει την επιλογή "*split-by*", που θα του επιτρέπει να ορίσει το χωρισμό του κειμένου της ιστοσελίδας σε τμήματα (fragments) με βάση κάποια έκφραση που ο ίδιος θα προσδιορίζει. Η δυνατότητα αυτή είναι χρήσιμη γιατί διευκολύνει στον εντοπισμό των συγκεκριμένων πληροφοριών που ενδιαφέρουν το χρήστη (π.χ. στην περίπτωση που σε μια

σελίδα δίνονται οι θερμοκρασίες για την πόλη του ενδιαφέροντος του αλλά και για κάποια άλλη πόλη)

- Για να μπορέσει ο χρήστης να κατανοήσει το νόημα κάθε πληροφορίας που εξάγεται από την ιστοσελίδα-πηγή και να επιλέξει τα επιθυμητά δεδομένα, απαιτείται να παρουσιάζεται σε αυτόν ένα τμήμα του κειμένου που προηγείται και έπεται κάθε αποτελέσματος. Έτσι ,στο βήμα αυτό θα πρέπει να μπορεί ο χρήστης να προσδιορίσει το βαθμό της λεπτομέρειας με την οποία θέλει να του παρουσιαστεί στη συνέχεια το κείμενο αυτό (πολύ χαμηλή, χαμηλή, μεσαία, υψηλή, πολύ υψηλή).
- Τέλος, ο χρήστης θα πρέπει να μπορεί να επιλέξει αν θέλει να αγνοηθούν ή όχι από το σύστημα τα tags της γλώσσας HTML (συμπεριλαμβανομένων των σχολίων και των τμημάτων κώδικα - *scripts*) κατά τη λεκτική και συντακτική ανάλυση (parsing) του κειμένου και την εξαγωγή πληροφορίας..

Βήμα 3α (προαιρετικό)

Όπως αναφέραμε και προηγουμένως, ο χρήστης μέσω της κατάλληλης επιλογής θα μπορεί να ξεκινήσει μια διαδικασία ορισμού των κανονικών εκφράσεων. Ο χρήστης θα βλέπει για το επιθυμητό χαρακτηριστικό του καιρού τις κανονικές εκφράσεις που χρησιμοποιούνται για την εξαγωγή πληροφορίας και θα μπορεί να επιλέξει κάποια από αυτές, να την τροποποιήσει, να τη διαγράψει οριστικά ή και να εισάγει μια νέα έκφραση η οποία θέλει να χρησιμοποιηθεί. Θα παρέχεται μάλιστα και μια επιλογή «δοκιμής» ώστε να μπορεί ο χρήστης να τρέξει επιτόπου την επιλεγμένη έκφραση στη σελίδα που έχει ορίσει και να ελέγξει τα αποτελέσματά της. Για τη δημιουργία νέων κανονικών εκφράσεων ή την τροποποίηση των υπαρχόντων θα δίνεται η δυνατότητα στο χρήστη να πληκτρολογήσει την κανονική έκφραση ή τις αλλαγές που θέλει να κάνει. Επιπλέον όμως, πρέπει να προσφέρεται ένα σύνολο βοηθητικών επιλογών για την αυτόματη προσθήκη στην κανονική έκφραση φράσεων ή μονάδων μέτρησης που αντιστοιχούν στο συγκεκριμένο χαρακτηριστικό του καιρού (π.χ. «αίθριος» ή «άστατος» για τη γενική περιγραφή του καιρού, «km/h» ή «μποφόρ» για τον άνεμο) ή άλλων εκφράσεων, όπως για παράδειγμα αριθμούς (διψήφιους ακεραίους, δεκαδικούς αριθμούς κ.α.), ποσοτικοποιητές (για επανάληψη μιας φράσης ν φορές), κενά, τελεστές διάζευξης κ.τ.λ. Με τον τρόπο αυτό διευκολύνεται ο χρήστης, ο οποίος δε χρειάζεται να πληκτρολογήσει ολόκληρη την έκφραση.

Βήμα 4

Στη συνέχεια, εφαρμογή θα κάνει parsing της συγκεκριμένης σελίδας που έδωσε ο χρήστης, αναζητώντας τις πληροφορίες που του ζήτησε και θα παρουσιάζει τα αποτελέσματα σε μία λίστα. Σε κάθε γραμμή θα εμφανίζεται το κείμενο που έχει εξαχθεί από τη σελίδα μαζί με το γειτονικό του κείμενο (χωρίς τα tags της γλώσσας HTML ώστε να είναι πιο κατανοητό), ενώ θα πρέπει να προσφέρεται και η δυνατότητα να

χαρακτηριστεί η συγκεκριμένη πληροφορία ως επιθυμητή, Μάλιστα, για να περιγραφεί το νόημα κάθε πληροφορίας, θα πρέπει να μπορεί ο χρήστης να τοποθετεί κάποιες επιγραφές σε αυτές. Οι επιγραφές αυτές θα είναι δύο ειδών: ετικέτες (*labels*) που περιγράφουν το «τι είναι η συγκεκριμένη πληροφορία» (π.χ. θερμοκρασία) και προσδιοριστικά περιεχομένου (*context tags*) τα οποία δείχνουν το γενικότερο νοηματικό πλαίσιο της πληροφορίας (π.χ. «Ο καιρός απόψε») και σύμφωνα με τα οποία μπορεί να γίνεται ομαδοποίηση των πληροφοριών.

Στην περίπτωση που δεν έγινε δυνατή η εξαγωγή καμίας πληροφορίας, πρέπει να εμφανίζεται στο χρήστη το κατάλληλο μήνυμα. Το ίδιο ισχύει και στην περίπτωση που δεν ήταν εφικτή η σύνδεση με τη συγκεκριμένη ιστοσελίδα. Αν, αντίθετα, βρέθηκαν πάνω από 10 πληροφορίες που να ταιριάζουν σε κάθε επιλογή του χρήστη, τότε θα εμφανίζονται μόνο οι 10 πρώτες και ο χρήστης θα ενημερώνεται με κατάλληλο μήνυμα ώστε να τροποποιήσει ενδεχομένως τις ρυθμίσεις του.

Όσο ο χρήστης δεν είναι ευχαριστημένος από το αποτέλεσμα, θα μπορεί είτε να επιστρέψει στο βήμα 3 και να διορθώσει τις επιλογές του, είτε να τροποποιήσει τις κανονικές εκφράσεις, μέσω της ειδικής επιλογής που απαιτείται να υπάρχει και σε αυτό το βήμα, και στη συνέχεια να ξανατρέξει τη σελίδα. Αν είναι ευχαριστημένος, τότε θα μπορεί να προχωρήσει στο επόμενο βήμα, αφού σημειώσει τις πληροφορίες που τον ενδιαφέρουν και συμπληρώσει τα labels και τα context tags, όπου επιθυμεί.

Βήμα 5

Στο επόμενο βήμα θα γίνεται ο καθορισμός των επιλογών εμφάνισης της πρόγνωσης σε δύο διαδοχικά στάδια:

- Αρχικά, θα εμφανίζονται όλες οι επιλογές του χρήστη, κατηγοριοποιημένες σύμφωνα με τα context tags. Ο χρήστης θα μπορεί να ρυθμίσει τη σειρά εμφάνισης κάθε ομάδας στην πρόγνωση (π.χ. πρώτα η ομάδα «Σήμερα» και μετά η ομάδα «Αύριο»), καθώς και τη σειρά εμφάνισης των στοιχείων μέσα σε κάθε ομάδα (π.χ. στο «Σήμερα» να εμφανίζεται πρώτα η «θερμοκρασία» και μετά η «υγρασία»).
- Στη συνέχεια θα εμφανίζονται όλες οι ομάδες με τα στοιχεία τους με τη σειρά που έχει οριστεί, και για κάθε στοιχείο θα δίνεται η δυνατότητα στο χρήστη να επιλέξει αν θα εμφανίζεται εκτός από το κεντρικό παράθυρο της πρόγνωσης και στην πρόγνωση της γραμμής κατάστασης και αν, στην περίπτωση αυτή, θα αποτελεί στοιχείο της γραμμής κατάστασης ή στοιχείο του κυλιόμενου μηνύματος (tooltip) που θα εμφανίζεται όταν το ποντίκι του χρήστη δείχνει πάνω στην πρόγνωση.

Στην περίπτωση που ο χρήστης έχει επιλέξει στο Βήμα 2 το χαρακτηριστικό "*γενικός χαρακτηρισμός καιρού*" τότε μετά το Βήμα 5 θα μεταβαίνει στο Βήμα 6, διαφορετικά θα πηγαίνει κατευθείαν στο Βήμα 7.

Βήμα 6

Στο βήμα αυτό θα καθορίζονται οι προτιμήσεις του χρήστη σχετικά με την χρήση εικονιδίων πρόγνωσης. Σε σχέση με τη χρήση των εικονιδίων πρέπει να παρατηρηθούν τα εξής: κάθε εικονίδιο θα πρέπει να μπορεί να συνδέεται με ένα σύνολο φράσεων (π.χ. «συννεφιά», «νεφελώδης», «σύννεφα»), ώστε να χρησιμοποιείται όταν και μόνο όταν στην πρόγνωση περιέχονται οι φράσεις αυτές. Για το σκοπό αυτό θα πρέπει να χρησιμοποιηθεί και πάλι ο μηχανισμός των κανονικών εκφράσεων. Όπως είναι λογικό όμως, δεν μπορεί να γίνει απευθείας σύνδεση των εικονιδίων με την κανονική έκφραση του χαρακτηριστικού "γενική περιγραφή καιρού", καθώς η έκφραση αυτή πιθανότατα περιλαμβάνει όλους τους πιθανούς χαρακτηρισμούς του καιρού, και όχι μόνο αυτούς που αντιστοιχούν σε μία εικόνα. Έτσι, στο βήμα αυτό απαιτείται να δώσει ο χρήστης μία κανονική έκφραση για κάθε εικόνα που επιθυμεί να χρησιμοποιεί στην παρουσίαση των δεδομένων. Η έκφραση αυτή θα μπορεί να δίνεται είτε απευθείας, είτε ως ένα σύνολο φράσεων, οι οποίες θα συνδυάζονται στην πορεία για να δώσουν την απαιτούμενη κανονική έκφραση. Για τη διευκόλυνσή του μάλιστα στη δεύτερη περίπτωση είναι αναγκαίο να παρέχεται ένας μηχανισμός αυτόματης εισαγωγής φράσεων.

Πιο συγκεκριμένα επομένως, σε αυτό το βήμα θα πρέπει να υπάρχει μια επιλογή την οποία θα μπορεί να «τσεκάρει» ο χρήστης αν θέλει να εμφανίζονται εικονίδια στην πρόγνωσή του, οπότε και θα ενεργοποιείται μια σειρά επιπλέον επιλογών που θα του επιτρέπει:

- Να προσθέσει ένα εικονίδιο το οποίο επιθυμεί να χρησιμοποιείται, φορτώνοντάς το από κάποια τοποθεσία στον υπολογιστή του. Όταν ο χρήστης επιλέξει το εικονίδιο θα πρέπει να καθορίσει την αντίστοιχη κανονική έκφραση μέσα από ένα ενδιάμεσο βήμα και σύμφωνα με τα όσα αναφέρθηκαν παραπάνω.
- Να διαγράψει ένα εικονίδιο από τη λίστα των εικονιδίων που έχουν επιλεγεί προς χρήση.
- Να τροποποιήσει την κανονική έκφραση που έχει συνδέσει με κάποιο από τα εικονίδια
- Να φορτώσει τα προεπιλεγμένα εικονίδια του συστήματος. Τα εικονίδια αυτά θα είναι ήδη συνδεδεμένα με συγκεκριμένες κανονικές εκφράσεις, τις οποίες όμως θα μπορεί να τροποποιήσει ο χρήστης όπως και τις υπόλοιπες, ενώ τα εικονίδια θα μπορούν να χρησιμοποιηθούν και σε συνδυασμό με άλλα.
- Να καθαρίσει τη λίστα από όλα τα εικονίδια που έχει εισάγει.

Βήμα 7

Τέλος, ο χρήστης θα καλείται να δώσει ένα όνομα στο προφίλ που δημιούργησε και να επιλέξει αν θα χρησιμοποιηθεί αυτό το προφίλ για την ενημέρωσή του. Στην περίπτωση που υπάρχει ήδη προφίλ με το όνομα που έδωσε, ή που χρησιμοποιήθηκαν μη επιτρεπτοί χαρακτήρες το σύστημα θα πρέπει να τον ενημερώνει και να τον προτρέπει να δώσει ένα διαφορετικό όνομα. Με την ολοκλήρωση του βήματος

αυτού θα ολοκληρώνεται και ο βοηθός δημιουργίας νέου προφίλ και θα αποθηκεύονται οι προτιμήσεις του χρήστη για το νέο προφίλ στο σύστημα προτιμήσεων του Mozilla Firefox.

4.4.1.2 Διαδικασία τροποποίησης προφίλ χρήστη

Εδώ περιγράφεται η διαδικασία με την οποία ο χρήστης του συστήματος θα μπορεί να τροποποιήσει ένα ήδη υπάρχον προφίλ, όταν θα κάνει την αντίστοιχη επιλογή από το βασικό μενού ή από το μενού της γραμμής κατάστασης. Για το σκοπό αυτό θα χρησιμοποιείται και πάλι ένας βοηθός, αντίστοιχος με αυτόν που περιγράφηκε παραπάνω. Η διαδικασία θα είναι εντελώς παρόμοια με τη δημιουργία νέου προφίλ, με τη διαφορά ότι ο βοηθός θα ξεκινά από το Βήμα 2, έχοντας ήδη φορτώσει το URL που έχει αποθηκευθεί ως πηγή της πρόγνωσης, ενώ αυτόματα με τη μετάβαση στο επόμενο βήμα θα ανοίγεται στον περιηγητή η αντίστοιχη σελίδα. Ακόμη, σε κάθε βήμα ο χρήστης θα βλέπει ήδη επιλεγμένες τις προτιμήσεις του όπως έχουν αποθηκευθεί στο συγκεκριμένο προφίλ, εκτός αν σε προηγούμενο βήμα έχει τροποποιήσει κάποιες από τις επιλογές του. Όπως είναι αυτονόητο, αν στο Βήμα 3 ο χρήστης δεν τροποποιήσει καμία από τις επιλογές του, θα παρακάμπτεται το Βήμα 4 καθώς δεν υπάρχει λόγος να γίνει εκ νέου τρέξιμο της σελίδας και θα εμφανίζονται κατευθείαν η επιλογές του χρήστη σχετικά με τη σειρά εμφάνισης των διαφόρων ομάδων και των στοιχείων σε αυτές (Βήμα 5). Για την περίπτωση όμως που ο χρήστης επιθυμεί να ξαναγίνει το parsing της σελίδας ούτως ή άλλως, θα προσφέρεται η αντίστοιχη επιλογή στο Βήμα 3.

4.4.1.3 Διαδικασία τροποποίησης προτιμήσεων για το προφίλ συστήματος

Όταν από το βασικό μενού ο χρήστης επιλέξει το προφίλ "*freemeteo*" και πατήσει την επιλογή που αντιστοιχεί σε τροποποίηση ενός προφίλ, η διαδικασία που θα ακολουθείται θα είναι διαφορετική απ' ότι για τα προφίλ χρήστη, καθότι στην περίπτωση αυτή οι επιλογές είναι πιο συγκεκριμένες. Η διαδικασία αυτή (για την οποία μπορεί να χρησιμοποιηθεί ένα τυπικό παράθυρο ορισμού προτιμήσεων του Mozilla Firefox), θα αποτελείται από 2 βήματα:

- Το πρώτο βήμα αφορά στα χαρακτηριστικά του καιρού τα οποία θέλει να βλέπει ο χρήστης στην πρόγνωσή του. Συγκεκριμένα εδώ ο χρήστης θα μπορεί να επιλέξει την περιοχή για την οποία θα λαμβάνει πρόγνωση από μία προκαθορισμένη λίστα πόλεων και να σημειώσει ποιο από τα χαρακτηριστικά "*γενική περιγραφή καιρού*", "*θερμοκρασία*", "*άνεμος*", "*υγρασία*", "*συνολικός υετός*" τον ενδιαφέρει να βλέπει.
- Το δεύτερο βήμα αφορά στον τρόπο εμφάνισης των μετεωρολογικών προγνώσεων. Καταρχάς πρέπει να υπάρχει επιλογή για τον αριθμό των ημερών (από 1 μέχρι 6 ημέρες) που θα εμφανίζονται στο παράθυρο προγνώσεων, καθώς και για τον αριθμό των ημερών που θα

εμφανίζονται στη γραμμή κατάστασης (το πολύ μέχρι 3 ημέρες για να μην καταλαμβάνεται όλος ο χώρος). Ο χρήστης θα μπορεί ακόμη να επιλέξει αν στη γραμμή κατάστασης θα εμφανίζονται μόνο οι προγνώσεις για την ημέρα (πρωί) ή και οι προγνώσεις για τη νύχτα για τις αντίστοιχες πάντα ημέρες. Τέλος, θα δίνεται η δυνατότητα επιλογής για χρήση εικονιδίων στην πρόγνωση ή όχι.

4.4.2 Υποσύστημα ανάκτησης δεδομένων

Το υποσύστημα αυτό θα τίθεται σε λειτουργία από τη στιγμή που ο χρήστης θα ανοίξει τον περιηγητή του. Οι βασικές του λειτουργίες θα είναι:

- Επιλογή του κατάλληλου προγράμματος εξαγωγής δεδομένων (κατάλληλο wrapper template), ανάλογα με το επιλεγμένο προφίλ.
- Σύνδεση με την αντίστοιχη τοποθεσία και εξαγωγή των δεδομένων.
- Παρουσίαση των δεδομένων στο χρήστη.
- Έλεγχος για αλλαγές των προτιμήσεων χρήστη (αλλαγές στο τρέχον προφίλ, ή επιλογή διαφορετικού προφίλ προς χρήση)
- Διαχείριση των εντολών που δίνονται από το χρήστη, μέσω του μενού γραμμής κατάστασης.

Η πρώτη από αυτές τις λειτουργίες αφορά στον έλεγχο εκ μέρους του υποσυστήματος του τρέχοντος προφίλ που έχει επιλέξει να χρησιμοποιεί ο χρήστης. Ανάλογα με το αν πρόκειται για προφίλ χρήστη ή για το προφίλ συστήματος θα ενεργοποιείται το κατάλληλο υποσύστημα (Υποσύστημα λειτουργίας προφίλ χρήστη ή Υποσύστημα λειτουργίας προφίλ συστήματος). Στη συνέχεια θα λαμβάνονται οι προτιμήσεις του χρήστη για το επιλεγμένο προφίλ. Οι τέσσερις άλλες λειτουργίες θα εκτελούνται πλέον από το ενεργοποιημένο υποσύστημα. Πρέπει να σημειωθεί εδώ ότι τα δύο υποσυστήματα είναι εντελώς αντίστοιχα, αν και η εξαγωγή και η παρουσίαση των δεδομένων θα γίνεται με διαφορετικό τρόπο, γεγονός που οφείλεται στο ότι οι διαδικασίες είναι πιο καθορισμένες και πιο άμεσες στην περίπτωση του προφίλ “*freemeteo*”. Θα υπάρχουν δηλαδή δύο διαφορετικά πρότυπα wrapper, ένα για κάθε υποσύστημα, τα οποία θα πραγματοποιούν την εξαγωγή της επιθυμητής πληροφορίας χρησιμοποιώντας τις προτιμήσεις του χρήστη για το εκάστοτε χρησιμοποιούμενο προφίλ.

Κατά τη δεύτερη λειτουργία, θα πραγματοποιείται ένα αίτημα για λήψη της ιστοσελίδας-πηγής των μετεωρολογικών δεδομένων και στη συνέχεια θα γίνεται η εξαγωγή των δεδομένων από την ιστοσελίδα αυτή με τη χρήση του wrapper template. Τα δεδομένα που έχουν εξαχθεί θα τοποθετούνται στο παράθυρο πρόγνωσης αλλά και στη γραμμή κατάστασης, σύμφωνα με τις ρυθμίσεις του τρέχοντος προφίλ. Τα δεδομένα της γραμμής κατάστασης θα είναι μονίμως φανερά στο χρήστη, ενημερώνοντάς τον άμεσα για τις πληροφορίες που έχει κρίνει αυτός ως βασικές. Αντίθετα, το παράθυρο πρόγνωσης θα περιέχει πιο αναλυτικές πληροφορίες, τις οποίες θα μπορεί να εμφανίσει ο χρήστης μέσω της κατάλληλης εντολής του

μενού (λειτουργία παρουσίασης δεδομένων στο χρήστη). Η διαδικασία της σύνδεσης με την πηγή της πρόγνωσης και τις εξαγωγής των δεδομένων πρέπει να εκτελείται ανά τακτά χρονικά διαστήματα, ώστε να είναι πάντα ενημερωμένα τα δεδομένα που παρουσιάζονται στο χρήστη.

Η λειτουργία της παρουσίασης των δεδομένων θα εκτελείται όταν ο χρήστης επιλέξει από το μενού της γραμμής κατάστασης την αντίστοιχη εντολή, οπότε θα εμφανίζεται ένα παράθυρο που θα περιέχει όλα τα δεδομένα που έχουν εξαχθεί από την επιλεγμένη πηγή πρόγνωσης, σύμφωνα με τον τρόπο παρουσίασης που έχει οριστεί στο τρέχον προφίλ.

Τέλος, η λειτουργία του ελέγχου για αλλαγές των προτιμήσεων χρήστη είναι αυτή που θα αναλαμβάνει να κάνει τις κατάλληλες ρυθμίσεις όταν υπάρξει αλλαγή σε κάποια προτίμηση του χρήστη. Τέτοιες ενέργειες για παράδειγμα μπορεί να είναι η λήψη των νέων προτιμήσεων, η ενεργοποίηση διαφορετικού υποσυστήματος λειτουργίας αν αυτό είναι απαραίτητο, και η επανεκκίνηση της διαδικασίας λήψης και εξαγωγής των δεδομένων. Η λειτουργία αυτή είναι απαραίτητο να εκτελείται ασταμάτητα στο παρασκήνιο.

5 *Σχεδίαση Συστήματος*

Στην ενότητα αυτή παρουσιάζεται η σχεδίαση του συστήματος. Στην πρώτη παράγραφο δίνονται τα επιμέρους τμήματα από τα οποία έχει δομηθεί ο κώδικάς του συστήματος (κλάσεις του προγράμματος), ενώ στην επόμενη παράγραφο ακολουθεί η συνοπτική περιγραφή των λειτουργιών-μεθόδων της κάθε κλάσης ξεχωριστά.

5.1 Αρχιτεκτονική

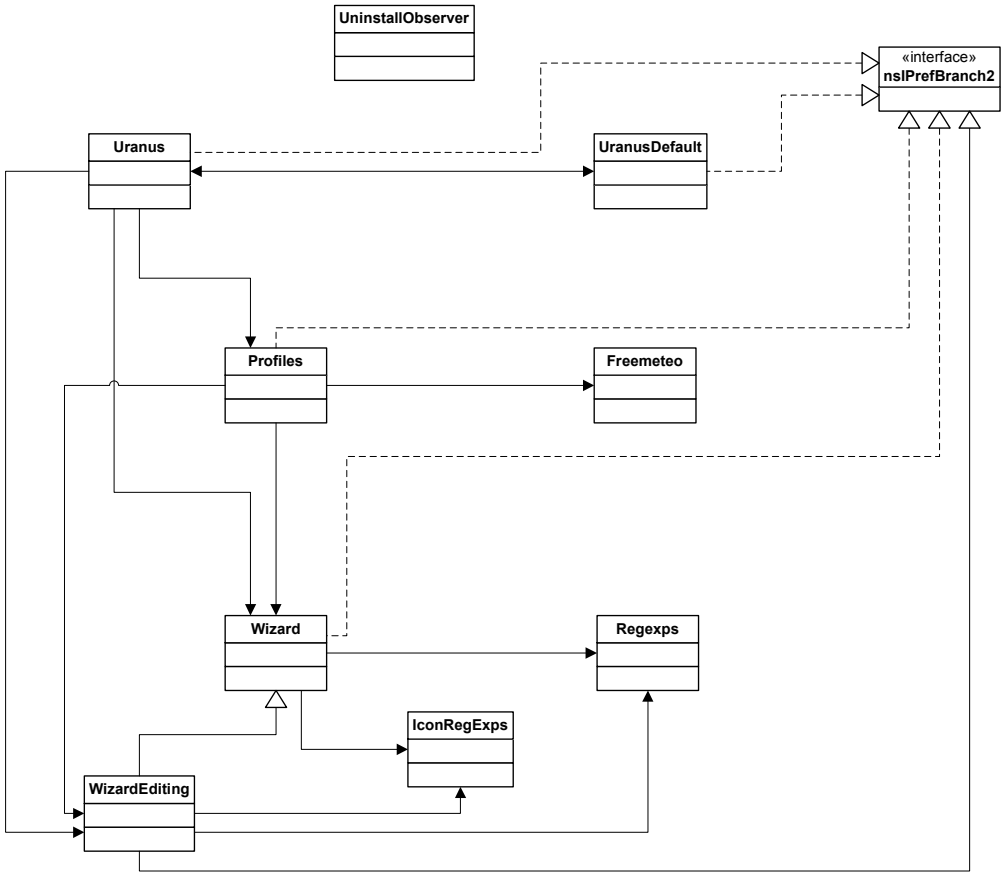
Τα επιμέρους τμήματα- κλάσεις του κώδικα του συστήματος UranuS είναι τα εξής:

- ***Κλάση Uranus:*** η κλάση αυτή αναλαμβάνει τον έλεγχο για το χρησιμοποιούμενο προφίλ και πραγματοποιεί την εξαγωγή και παρουσίαση των δεδομένων στην περίπτωση που χρησιμοποιείται ένα από τα προφίλ χρήστη. Επίσης διαχειρίζεται τις εντολές που δίνονται από το χρήστη μέσω του μενού της γραμμής κατάστασης.
- ***Κλάση UranusDef:*** πραγματοποιεί την εξαγωγή και παρουσίαση των δεδομένων στην περίπτωση που χρησιμοποιείται το προφίλ συστήματος. Διαχειρίζεται και αυτή τις εντολές του χρήστη που δίνονται μέσα από το μενού της γραμμής κατάστασης.
- ***Κλάση Profiles:*** αναλαμβάνει τη βασική διαχείριση των προφίλ (παρουσίαση των προφίλ στο χρήστη και κλήση της αντίστοιχης κλάσης ανάλογα με την ενέργεια που επιλέγει να κάνει αυτός)
- ***Κλάση Freemeteo:*** αναλαμβάνει την επεξεργασία του προφίλ συστήματος.
- ***Κλάση Wizard:*** χρησιμοποιείται για τη δημιουργία νέου προφίλ από το χρήστη.
- ***Κλάση WizardEditing:*** χρησιμοποιείται για την τροποποίηση ήδη υπάρχοντος προφίλ χρήστη.
- ***Κλάση RegExps:*** είναι υπεύθυνη για τη διαχείριση των κανονικών εκφράσεων που θα χρησιμοποιηθούν για την εξαγωγή πληροφορίας σε ένα προφίλ.

- ***Κλάση IconsRegExps:*** είναι υπεύθυνη για τη διαχείριση των κανονικών εκφράσεων που συνδέονται με κάποιο εικονίδιο.
- ***Κλάση UninstallObserver:*** είναι υπεύθυνη για τον έλεγχο αιτήματος για απεγκατάσταση της επέκτασης από το χρήστη, ώστε να εκτελέσει κάποιες απαραίτητες ενέργειες στην περίπτωση αυτή (διαγραφή των προτιμήσεων χρήστη)

Από τις κλάσεις αυτές, οι *Uranus, UranusDefault, Profiles, Wizard* και *WizardEditing* υλοποιούν τη διεπαφή (interface) *nsIPrefBranch2* του Mozilla, η οποία σχετίζεται με τη διαχείριση των προτιμήσεων του χρήστη (λήψη ή ορισμός τιμών προτιμήσεων, παρατήρηση αλλαγών κλπ.).

Ακολουθεί το διάγραμμα κλάσεων του συστήματος:



5.2 Περιγραφή Κλάσεων

Παρακάτω περιγράφονται συνοπτικά οι μέθοδοι των κλάσεων που αναφέρθηκαν προηγουμένως.

5.2.1 Κλάση Uranus

Οι βασικές λειτουργίες/μέθοδοι της κλάσης είναι οι εξής:

1. ***startup*:** λαμβάνει τις τιμές των προτιμήσεων (preferences) του χρήστη και αρχικοποιεί τις διάφορες παραμέτρους της κλάσης. Επίσης δημιουργεί έναν «παρατηρητή» (observer), επιφορτισμένο με τον έλεγχο για τυχόν αλλαγές στις προτιμήσεις του χρήστη. Τέλος, καλεί τη μέθοδο *refreshInformation* που περιγράφεται πιο κάτω.
2. ***observe*:** ορίζει τις ενέργειες που πρέπει να γίνουν στην περίπτωση που υπάρξει αλλαγή σε κάποιες από τις προτιμήσεις του χρήστη (π.χ. ενημέρωση παραμέτρων της κλάσης, επανεκκίνηση της διαδικασίας λήψης δεδομένων, κλήση της κλάσης UranusDefault στην περίπτωση που επελέγη να χρησιμοποιηθεί το προφίλ συστήματος κ.τ.λ.)
3. ***refreshInformation*:** ανανεώνει τα μετεωρολογικά δεδομένα που παρουσιάζονται στο χρήστη. Χρησιμοποιεί τις εξής μεθόδους:
    - ***sendRequest*:** στέλνει ένα αίτημα στο διαδίκτυο μέσω του πρωτοκόλλου http (*httpRequest*) για την HTML σελίδα που της δίνεται ως παράμετρος. Καλείται από τη μέθοδο *refreshInformation* μία φορά, με παράμετρο το URL που έχει οριστεί ως πηγή των δεδομένων από το χρήστη.
    - ***handleRequest*:** διαχειρίζεται την πληροφορία που λαμβάνεται ως απόκριση στο *httpRequest*. Συγκεκριμένα, εφαρμόζει τις κανονικές εκφράσεις που ορίζονται από το τρέχον προφίλ για να εξάγει την επιθυμητή πληροφορία και στη συνέχεια ενσωματώνει την πληροφορία αυτή στο παράθυρο πρόγνωσης και στη γραμμή κατάστασης σύμφωνα με τον τρόπο που έχει ορίσει ο χρήστης στο τρέχον προφίλ.
    - ***timeElapsed*:** στην περίπτωση αποτυχίας λήψης απόκρισης από την επιθυμητή τοποθεσία μέσα σε κάποιο προκαθορισμένο διάστημα, εμφανίζει το σχετικό μήνυμα στη γραμμή κατάστασης.
4. ***openOptions*:** ανοίγει το παράθυρο της διαχείρισης των προφίλ όταν ο χρήστης πατήσει την αντίστοιχη εντολή στο μενού γραμμής κατάστασης.
5. ***openWizard*:** ανοίγει το βοηθό (wizard) για τη δημιουργία νέου προφίλ όταν ο χρήστης επιλέξει την αντίστοιχη εντολή από το μενού γραμμής κατάστασης.
6. ***editProfile*:** ανοίγει το βοηθό επεξεργασίας προφίλ για το τρέχον προφίλ, όταν ο χρήστης επιλέξει να κάνει αυτή την ενέργεια μέσω του μενού γραμμής κατάστασης.
7. ***shutdown*:** τερματίζει τον observer που είχε δημιουργήσει η μέθοδος *startup*.

5.2.2 Κλάση UranusDefault

Οι βασικές μέθοδοι της κλάσης είναι οι εξής:

1. ***startup*:** λαμβάνει τις τιμές των προτιμήσεων του χρήστη για το προφίλ συστήματος και αρχικοποιεί τις διάφορες παραμέτρους της κλάσης. Επίσης δημιουργεί έναν observer, επιφορτισμένο με τον έλεγχο για τυχόν αλλαγές στις προτιμήσεις του χρήστη και καλεί τη μέθοδο *refreshInformation* της κλάσης.
2. ***observe*:** ορίζει τις ενέργειες που πρέπει να γίνουν στην περίπτωση που υπάρξει αλλαγή σε κάποιες από τις προτιμήσεις του χρήστη (π.χ. ενημέρωση παραμέτρων της κλάσης, επανεκκίνηση της διαδικασίας λήψης δεδομένων, τερματισμός της κλάσης και κλήση της κλάσης Uranus στην περίπτωση που επελέγη να χρησιμοποιηθεί ένα προφίλ χρήστη κ.τ.λ.)
3. ***refreshInformation*:** ανανεώνει τα μετεωρολογικά δεδομένα που παρουσιάζονται στο χρήστη. Χρησιμοποιεί τις εξής μεθόδους:
    - ***sendRequest*:** καλείται από την πιο πάνω μέθοδο μία φορά για κάθε ημέρα που περιλαμβάνεται στην πρόγνωση με βάση τις προτιμήσεις του χρήστη. Η μέθοδος αυτή στέλνει ένα αίτημα *httpRequest* για την HTML σελίδα που της δίνεται ως παράμετρος.
    - ***handleRequest*:** διαχειρίζεται την πληροφορία που λαμβάνεται ως απόκριση σε ένα *httpRequest*. Συγκεκριμένα, εφαρμόζει τις κανονικές εκφράσεις του προφίλ συστήματος για να εξάγει την επιθυμητή πληροφορία και στη συνέχεια ενσωματώνει την πληροφορία αυτή στο παράθυρο πρόγνωσης και στη γραμμή κατάστασης, στην κατάλληλη θέση (ανάλογα και με την ημέρα στην οποία αντιστοιχεί η απόκριση που ελήφθη).
    - ***timeElapsed*:** στην περίπτωση αποτυχίας λήψης απόκρισης από την επιθυμητή τοποθεσία μέσα σε κάποιο προκαθορισμένο διάστημα, εμφανίζει το σχετικό μήνυμα στη γραμμή κατάστασης.
4. ***changeSize*:** αλλάζει την τιμή της προτίμησης του χρήστη που αφορά στον αριθμό των ημερών της πρόγνωσης που εμφανίζονται στη γραμμή κατάστασης., όταν ο χρήστης επιλέξει μια νέα τιμή από το μενού της γραμμής κατάστασης.
5. ***saveForecast*:** αποθηκεύει το τροποποιημένο XUL αρχείο που ορίζει το περιεχόμενο του παραθύρου πρόγνωσης.
6. ***shutdown*:** τερματίζει τον observer που έχει δημιουργήσει η μέθοδος *startup*.

5.2.3 Κλάση Profiles

Η κλάση αυτή σχετίζεται με το παράθυρο του βασικού μενού και οι λειτουργίες της αντιστοιχούν στην πλειοψηφία τους στις επιλογές που παρέχονται στο χρήστη για τη διαχείριση των προφίλ (δημιουργία, διαγραφή, επεξεργασία κ.τ.λ.).

1. ***startup*:** ενημερώνεται για το χρησιμοποιούμενο προφίλ και τα ονόματα των διαφόρων άλλων προφίλ που έχει δημιουργήσει ο χρήστης και τα παρουσιάζει στο χρήστη σε μία λίστα. Επίσης δημιουργεί έναν observer, επιφορτισμένο με τον έλεγχο για τυχόν αλλαγή του χρησιμοποιούμενου προφίλ (τρέχον προφίλ) ή για διαγραφή κάποιου προφίλ.
2. ***observe*:** ορίζει τις ενέργειες που πρέπει να γίνουν στην περίπτωση που επιλεγεί νέο προφίλ προς χρήση ή διαγραφεί ή προστεθεί κάποιο προφίλ. Σε κάθε περίπτωση πρέπει να αλλάξουν τα στοιχεία που παρουσιάζονται στο χρήστη στο παράθυρο του βασικού μενού, ενώ στην περίπτωση διαγραφής προφίλ γίνεται και έλεγχος για το αν το προφίλ αυτό ήταν το χρησιμοποιούμενο, ώστε να ειδοποιηθεί σχετικά ο χρήστης και να ορίσει νέο τρέχον προφίλ.
3. ***checkActions*:** ελέγχει την ενεργοποίηση και την απενεργοποίηση των κουμπιών του βασικού μενού ανάλογα με τον αριθμό των προφίλ που έχει επιλέξει ο χρήστης από την αντίστοιχη λίστα (π.χ. αν ο χρήστης δεν έχει επιλέξει ένα προφίλ δεν μπορεί να πατήσει την επιλογή "*Delete Profile*").
4. ***selectProfile*:** αποθηκεύει ως τρέχον προφίλ το προφίλ που έχει επιλέξει ο χρήστης στη λίστα.
5. ***deleteProfile*:** διαγράφει το προφίλ που είναι επιλεγμένο στη λίστα και ενημερώνει την κατάλληλη παράμετρο της κλάσης σχετικά με το αν το προφίλ που διεγράφη ήταν το τρέχον προφίλ.
6. ***createNew*:** ανοίγει το βοηθό δημιουργίας νέου προφίλ χρήστη (ή αν ήταν ήδη ανοιχτός τον επαναφέρει στο προσκήνιο).
7. ***editProfile*:** ανοίγει το βοηθό επεξεργασίας προφίλ χρήστη (ή τον επαναφέρει στο προσκήνιο, αντίστοιχα με πριν).
8. ***doSave*:** όταν ο χρήστης ζητήσει να κλείσει το παράθυρο του βασικού μενού ελέγχει αν έχει διαγραφεί το χρησιμοποιούμενο προφίλ και αν ναι απαιτεί από το χρήστη να ορίσει νέο τρέχον προφίλ.

5.2.4 Κλάση Freemeteo

Η κλάση αυτή σχετίζεται με το παράθυρο τροποποίησης του προφίλ συστήματος. Το παράθυρο αυτό είναι ένα τυπικό παράθυρο της κατηγορίας "*prefwindow*" του Mozilla το οποίο με αυτόματο τρόπο αναλαμβάνει να λάβει και να παρουσιάσει στο χρήστη τις αντίστοιχες προτιμήσεις του και στη συνέχει

να αποθηκεύσει τις τυχόν αλλαγές που ζήτησε ο αυτός. Επομένως η κλάση αυτή έχει πολύ μικρό ρόλο, ο οποίος περιορίζεται στις παρακάτω λειτουργίες.

1. ***adjustS_Bsize*:** όταν ο χρήστης τροποποιήσει την τιμή της προτίμησης που αφορά στον συνολικό αριθμό ημερών που περιλαμβάνονται στην πρόγνωση (έστω n), η μέθοδος αυτή αναλαμβάνει να τροποποιήσει αντίστοιχα τη λίστα με τις δυνατές τιμές για τον αριθμό των ημερών προς εμφάνιση στη γραμμή κατάστασης, ώστε να επιτρέπεται η εμφάνιση το πολύ m ημερών, όπου m = min (3,n).
2. ***checkValues*:** κατά το κλείσιμο του αντίστοιχου παραθύρου ελέγχει αν όντως ο αριθμός των ημερών που έχει επιλεγεί να εμφανίζονται στη γραμμή κατάστασης είναι μικρότερος από το συνολικό αριθμό ημερών και διαφορετικά εμφανίζει το κατάλληλο μήνυμα στο χρήστη.

5.2.5 Κλάση Wizard

Ακολουθούν οι βασικές λειτουργίες της κλάσης:

1. ***startup*:** αρχικοποιεί τις παραμέτρους της κλάσης.
2. ***loadPage:*** φορτώνει στον wizard το URL της σελίδας που έχει ανοιχτή ο χρήστης στον Mozilla Firefox.
3. ***Λειτουργίες βήματος ορισμού των επιθυμητών χαρακτηριστικών καιρού (Βήμα 4):***
    - ***enableButtons*:** ενεργοποιεί τα κουμπιά για τον ορισμό των κανονικών εκφράσεων για ένα χαρακτηριστικό καιρού, όταν ο χρήστης επιλέξει να ενημερώνεται για το συγκεκριμένο χαρακτηριστικό.
    - ***enableSplit:*** ενεργοποιεί το πεδίο ορισμού της έκφρασης με βάση την οποία θα διαχωριστεί σε τμήματα η απόκριση από την πηγή των δεδομένων, όταν ο χρήστης έχει επιλέξει να γίνεται η ενέργεια αυτή (*split*).
    - ***openRegExpWin:*** ανοίγει το παράθυρο διαλόγου των κανονικών εκφράσεων για το κατάλληλο χαρακτηριστικό καιρού, όταν ο χρήστης έχει πατήσει το αντίστοιχο κουμπί. Όταν ο χρήστης επιστρέψει από το διάλογο αυτό, αποθηκεύει τις κανονικές εκφράσεις στις κατάλληλες μεταβλητές (παραμέτρους) της κλάσης.
4. ***parsePage:*** κατά την ολοκλήρωση του Βήματος 4, ελέγχει ότι ο χρήστης έχει συμπληρώσει όλα τα απαραίτητα πεδία στη σελίδα ορισμού των επιθυμητών χαρακτηριστικών καιρού και στη συνέχει συνδέεται με τη σελίδα που έχει ζητήσει ο χρήστης, πραγματοποιεί την εξαγωγή των δεδομένων και παρουσιάζει τα αποτελέσματα στο χρήστη. Χρησιμοποιεί τις εξής μεθόδους:
    - ***sendRequest:*** αντίστοιχη με τις μεθόδους *sendRequest* των άλλων κλάσεων.

- ***handleRequest:*** επίσης αντίστοιχη με τις άλλες μεθόδους. Εφαρμόζει τις κανονικές εκφράσεις (όπως παρέχονται από το σύστημα ή όπως τις όρισε ο χρήστης στο παράθυρο των κανονικών εκφράσεων) στην απόκριση του αιτήματος *httpRequest* και εμφανίζει όλα τα αποτελέσματα της εξαγωγής στο χρήστη, μαζί με τα απαραίτητα πεδία "*label*", "*context tag*", και "*adjacent text*" όπως έχουν καθοριστεί στην ανάλυση απαιτήσεων του συστήματος.
- ***timeElapsed:*** παρόμοια με τις αντίστοιχες μεθόδους των άλλων κλάσεων.

5. ***enableTags:*** στο βήμα επιλογής των επιθυμητών αποτελεσμάτων (Βήμα 5) ενεργοποιεί και απενεργοποιεί τα πεδία *label* και *context tag* ενός στοιχείου ανάλογα με το αν ο χρήστης επέλεξε αυτό το στοιχείο ως επιθυμητό.
6. ***collectChoices:*** κατά την ολοκλήρωση του Βήματος 5 συλλέγει τις επιλογές του χρήστη από τη σελίδα των αποτελεσμάτων (πληροφορίες που θέλει να βλέπει και *labels* και *context tags* που αντιστοιχούν σε αυτές), τις ομαδοποιεί με βάση το πεδίο *context tag* και τις εμφανίζει στην επόμενη σελίδα του wizard.
7. ***moveGroup, moveItem:*** στο βήμα ορισμού της σειράς παρουσίασης των επιλεγμένων από το χρήστη στοιχείων (Βήμα 6a), μετακινούν μία ομάδα περιεχομένου, ή ένα στοιχείο μέσα σε μια ομάδα περιεχομένου αντίστοιχα, κατά μία θέση προς τα πάνω ή προς τα κάτω στην αντίστοιχη λίστα, όταν ο χρήστης πατά τα κατάλληλα πλήκτρα.
8. ***checkOrder:*** δημιουργεί τις νέες ομάδες περιεχομένου σύμφωνα με τη σειρά που έχει ορίσει ο χρήστης στο Βήμα 6a και τις τοποθετεί με τη δοσμένη σειρά στην επόμενη σελίδα ώστε να γίνει η επιλογή του τρόπου παρουσίασης κάθε στοιχείου.
9. ***enableFields:*** στο βήμα ορισμού του τρόπου παρουσίασης των δεδομένων (Βήμα 6b), αν ο χρήστης επιλέξει την εμφάνιση ενός στοιχείου στη γραμμή κατάστασης εμφανίζει τις επιλογές για το ρόλο που θα έχει (στοιχείο της γραμμής κατάστασης ή στοιχείο του κυλιόμενου κειμένου αυτής) ενώ διαφορετικά τις αποκρύπτει.
10. ***saveOptions:*** αποθηκεύει και τις προτιμήσεις παρουσίασης του χρήστη και στη συνέχεια, ανάλογα με το αν ο χρήστης έχει επιλέξει να ενημερώνεται για το χαρακτηριστικό "*γενική περιγραφή καιρού*" ή όχι, προχωρά στο βήμα ορισμού των χρησιμοποιούμενων εικονιδίων, ή το παρακάμπτει.
11. ***Λειτουργίες βήματος ορισμού των χρησιμοποιούμενων εικόνων (Βήμα 7):***
    - ***enableIcons:*** ενεργοποιεί τη λίστα και τα πλήκτρα με τις διάφορες ενέργειες (προσθήκη νέας εικόνας, τροποποίηση κανονικής έκφρασης κλπ) όταν ο χρήστης επιλέξει να χρησιμοποιούνται εικονίδια στην πρόβλεψη.
    - ***openIconsPicker:*** όταν επιλέγεται η προσθήκη νέου εικονιδίου, ανοίγει το παράθυρο αναζήτησης αρχείου και μετά την επιλογή κάποιας εικόνας τοποθετεί στη λίστα το

μονοπάτι αυτής στον υπολογιστή του χρήστη και ανοίγει το παράθυρο αντιστοίχισης κανονικής έκφρασης.

- ***ckeckButtons:*** ενεργοποιεί και απενεργοποιεί τις επιλογές "*remove*" και "*edit*" ανάλογα με το αν έχει επιλεγεί κάποιο εικονίδιο από τη λίστα ή όχι.
- ***changeIcon:*** όταν ο χρήστης επιλέξει την αλλαγή κάποιας χρησιμοποιούμενης εικόνας, ανοίγει πάλι το παράθυρο αναζήτησης αρχείου, ξεκινώντας από τον κατάλογο όπου βρισκόταν η εικόνα αυτή.
- ***removeIcon:*** διαγράφει την επιλεγμένη εικόνα από τη λίστα.
- ***editExpr:*** ανοίγει πάλι το παράθυρο αντιστοίχισης κανονικής έκφρασης σε εικόνα όταν ο χρήστης επιλέξει την τροποποίησης της κανονικής έκφρασης.
- ***loadDefaultIcons:*** τοποθετεί στη λίστα όλες τις προεπιλεγμένες εικόνες του συστήματος με τις αντίστοιχες κανονικές εκφράσεις τους.
- ***clearIcons:*** αδειάζει τη λίστα των εικόνων.

12. ***savePreferences:*** αποθηκεύει τις προτιμήσεις του χρήστη σε ένα νέο προφίλ, αφού πρώτα διενεργήσει τους κατάλληλους ελέγχους για την εγκυρότητα του ονόματος του προφίλ και για τη μη χρησιμοποίησή του από άλλο προφίλ (στη δεύτερη περίπτωση, αν ο χρήστης το επιλέξει τροποποιείται το ήδη υπάρχον προφίλ με βάση τις νέες επιλογές).

5.2.6 Κλάση WizardEditing

Οι μέθοδοι της κλάσης αυτής είναι στην πλειοψηφία τους ίδιες ή παρόμοιες με τις μεθόδους της κλάσης *Wizard*. Για το λόγο αυτό θα αναφερθούν μόνο οι μέθοδοι που παρουσιάζουν διαφορές ή που είναι εντελώς νέες.

1. ***startup*****:** λαμβάνει τις προτιμήσεις του χρήστη για το συγκεκριμένο προφίλ και αρχικοποιεί κατάλληλα το βοηθό τοποθετώντας τις τιμές αυτές στα διάφορα πεδία και «τσεκάροντας» τις αντίστοιχες επιλογές. Ακόμη, αρχικοποιεί τις διάφορες παραμέτρους της κλάσης.
2. ***sameRegexps:*** ελέγχει αν δύο πίνακες που δίνονται ως παράμετροι περιλαμβάνουν τις ίδιες ακριβώς κανονικές εκφράσεις. Χρησιμοποιείται από τη *openRegExpWin* όταν επιστρέφει ο χρήστης από το παράθυρο διαλόγου για να ελέγξει αν υπήρξαν αλλαγές στις κανονικές εκφράσεις και να ενημερώσει την αντίστοιχη μεταβλητή ελέγχου.
3. ***openPage:*** ανοίγει την HTML σελίδα που ορίζεται από το αποθηκευμένο URL του προφίλ, όταν ο χρήστης έχει τσεκάρει την αντίστοιχη επιλογή στο Βήμα 1.
4. ***checkParsePage:*** ελέγχει αν υπήρξε αλλαγή των προτιμήσεων του χρήστη σχετικά με την ιστοσελίδα-πηγή, τα επιθυμητά χαρακτηριστικά του καιρού ή τις αντίστοιχες κανονικές εκφράσεις. Αν οι επιλογές του είναι αμετάβλητες και δεν έχει επιλεχθεί να γίνει parsing ούτως ή

άλλως, τότε παρακάμπτεται το επόμενο βήμα, διαφορετικά καλείται η μέθοδος *parsePage* και γίνεται εκ νέου εξαγωγή πληροφορίας από την ιστοσελίδα για την παρουσίαση των αποτελεσμάτων στο χρήστη και την επιλογή της επιθυμητής πληροφορίας.

5. ***checkOrder:*** επιπλέον των λειτουργιών της αντίστοιχης μεθόδου της κλάσης *Wizard* ελέγχει αν οι επιλογές του χρήστη για τις ομάδες περιεχομένου και τη σειρά παρουσίασης αυτών και των στοιχείων τους είναι ακριβώς οι ίδιες με αυτές που είχαν αρχικά οριστεί στο προφίλ. Στην περίπτωση αυτή, στο επόμενο βήμα (όπου ορίζεται ο τρόπος παρουσίασης των στοιχείων) τοποθετούνται και οι αρχικές επιλογές του χρήστη.
6. ***savePreferences:*** επιπλέον των όσων αναφέρθηκαν στην περιγραφή της αντίστοιχης μεθόδου της κλάσης *Wizard*, ελέγχει αν το όνομα που δόθηκε για το προφίλ είναι εντελώς νέο, ώστε να αποθηκευθούν τελικά οι προτιμήσεις σε νέο προφίλ, ενώ στην περίπτωση που το όνομα παραμένει το ίδιο, ζητά επιβεβαίωση πριν πραγματοποιήσει τις ενημερώσεις του προφίλ. (Ο χρήστης έχει τη δυνατότητα να απενεργοποιήσει δια παντός αυτή την προτροπή για επιβεβαίωση, επιλέγοντας το αντίστοιχο πεδίο στο μήνυμα που του εμφανίζεται).

5.2.7 Κλάση RegExps

Οι βασικές μέθοδοι της κλάσης, οι οποίες στην πλειοψηφία τους αντιστοιχούν στις διάφορες επιλογές που έχει ο χρήστης στο παράθυρο των κανονικών εκφράσεων, είναι οι παρακάτω:

1. ***startup:*** αρχικοποιεί τις παραμέτρους της κλάσης, τοποθετεί τις κατάλληλες κανονικές εκφράσεις στη λίστα των εκφράσεων και κατασκευάζει τα μενού δημιουργίας εκφράσεων (μενού με έτοιμες εκφράσεις), ανάλογα με το χαρακτηριστικό του καιρού στο οποίο αντιστοιχεί το παράθυρο.
2. ***checkButtons*****:** Ελέγχει την ενεργοποίηση και την απενεργοποίηση των πλήκτρων (*πλήκτρα μενού*) που σχετίζονται με τις διάφορες ενέργειες επί των εκφράσεων (τροποποίηση, διαγραφή, δημιουργία νέας, δοκιμή κ.τ.λ.), ανάλογα με τον αριθμό των εκφράσεων που έχουν επιλεγεί από τη λίστα (0, 1 ή πολλές).
3. ***selectAll*****:** επιλέγει όλες τις κανονικές εκφράσεις της λίστας ταυτόχρονα
4. ***clearAll*****:** από-επιλέγει όλες τις κανονικές εκφράσεις της λίστας.
5. ***disableAll:*** απενεργοποιεί όλα τα πλήκτρα-μενού αλλά και τα πλήκτρα επιλογής και από-επιλογής όλων, και την ίδια τη λίστα των κανονικών εκφράσεων. Χρησιμοποιείται όταν έχει εμφανιστεί το ειδικό πεδίο δημιουργίας ή τροποποίησης κανονικής έκφρασης ή όταν δοκιμάζονται κάποιες κανονικές εκφράσεις στην ιστοσελίδα-πηγή (parsing σελίδας).
6. ***enableAll:*** ενεργοποιεί όλα τα παραπάνω.
7. ***enableBasic:*** ενεργοποιεί τη λίστα των εκφράσεων, τα πλήκτρα δημιουργίας νέας έκφρασης και φόρτωσης των προκαθορισμένων εκφράσεων, και τα πλήκτρα επιλογής και από-επιλογής όλων.

8. ***Λειτουργίες επί των εκφράσεων:***

- ***createNew:*** εμφανίζει το πεδίο δημιουργίας/τροποποίησης κανονικής έκφρασης και τα αντίστοιχα μενού (πλήκτρα "*add*", "*clear*", μενού αυτόματης εισαγωγής φράσης κ.τ.λ.).
- ***addToRegxp:*** προσθέτει το στοιχείο που δίνεται ως παράμετρος (στοιχείο των μενού αυτόματης εισαγωγής φράσης) στην κανονική έκφραση που βρίσκεται στο πεδίο δημιουργίας-τροποποίησης κανονικής έκφρασης.
- ***addExpr:*** προσθέτει στη λίστα των κανονικών εκφράσεων το περιεχόμενο του πεδίου δημιουργίας/τροποποίησης κανονικής έκφρασης και στη συνέχεια αποκρύπτει τα μενού δημιουργίας/τροποποίησης, και ενεργοποιεί το βασικό μενού. Αν υπάρχει λάθος στην κανονική έκφραση δεν είναι δυνατή η πραγματοποίηση αυτής της ενέργειας και ο χρήστης ειδοποιείται σχετικά.
- ***cancelExpr:*** ακυρώνει τη δημιουργία/τροποποίηση της κανονικής έκφρασης και αποκρύπτει τα αντίστοιχα μενού, ενεργοποιώντας και πάλι το βασικό μενού.
- ***deleteExpr:*** διαγράφει την επιλεγμένη κανονική έκφραση από τη λίστα, αφού πρώτα ζητήσει επιβεβαίωση από το χρήστη.
- ***editExpr:*** εμφανίζει το πεδίο δημιουργίας/τροποποίησης κανονικής έκφρασης και τα αντίστοιχα μενού και απενεργοποιεί το βασικό μενού.
- ***alterExpr:*** αντικαθιστά την επιλεγμένη κανονική έκφραση της λίστας με το περιεχόμενο του πεδίου δημιουργίας/τροποποίησης έκφρασης (αν αυτό φυσικά συνιστά ορθή κανονική έκφραση).
- ***restoreDefaults:*** καθαρίζει τη λίστα των κανονικών εκφράσεων και επαναφέρει τις προεπιλεγμένες κανονικές εκφράσεις του συστήματος για το συγκεκριμένο χαρακτηριστικό του καιρού.
- ***tryExpr:*** στέλνει ένα αίτημα *httpRequest* για την HTML σελίδα που έχει οριστεί στον wizard ως πηγή, πραγματοποιεί εξαγωγή δεδομένων με χρήση των επιλεγμένων κανονικών εκφράσεων από τη λίστα και παρουσιάζει τα αποτελέσματα στο χρήστη. Χρησιμοποιεί μια μέθοδο *handleResponse* αντίστοιχη με αυτές που έχουν χρησιμοποιηθεί σε άλλες κλάσεις, με τη διαφορά ότι στα αποτελέσματα παρουσιάζονται μόνο οι εξαγμένες πληροφορίες και το γειτονικό τους κείμενο (adjacent text). Χρησιμοποιεί επίσης τη μέθοδο *timeElapsed* κατά τα γνωστά.
- ***hideResults:*** αποκρύπτει τα αποτελέσματα του parsing της ιστοσελίδας-πηγής από το παράθυρο των κανονικών εκφράσεων.
- ***doSave:*** επιστρέφει στην καλούσα κλάση τις κανονικές εκφράσεις της λίστας.

5.2.8 Κλάση IconsRegExps

Η κλάση αυτή χειρίζεται τις διάφορες λειτουργίες του παραθύρου ορισμού των εκφράσεων που συνδέονται με κάποιο εικονίδιο που πρόκειται να χρησιμοποιηθεί στην πρόβλεψη. Όπως έχει αναφερθεί στην ανάλυση απαιτήσεων του συστήματος, οι επιλογές που δίνονται στο χρήστη είναι να δώσει μια πλήρη κανονική έκφραση, ή να δώσει φράσεις χωρισμένες με κόμμα, οι οποίες θα χρησιμοποιηθούν από το πρόγραμμα για την κατασκευή της κανονικής έκφρασης. Στη δεύτερη περίπτωση μάλιστα υπάρχει και ένα σύνολο ήδη έτοιμων φράσεων, τις οποίες μπορεί να επιλέξει να χρησιμοποιήσει ο χρήστης. Για ευκολία, παρακάτω η πρώτη περίπτωση θα χαρακτηρίζεται «εισαγωγή μέσω κανονικής έκφρασης» και η δεύτερη «εισαγωγή μέσω φράσεων»

1. ***startup:*** αρχικοποιεί τα στοιχεία του παραθύρου με το μονοπάτι της εικόνας η οποία έχει επιλεγεί να χρησιμοποιηθεί και με την οποία πρόκειται να συνδέσουμε την κανονική έκφραση, ή ακόμη και με την κανονική έκφραση ή το σύνολο των φράσεων που έχουν επιλεχθεί, όταν γίνεται τροποποίηση της έκφρασης.
2. ***doSave*****:** Αν έχει επιλεγεί εισαγωγή μέσω φράσεων, κατασκευάζει την αντίστοιχη κανονική έκφραση και την επιστρέφει στην καλούσα κλάση. Διαφορετικά, ελέγχει την ορθότητα της κανονικής έκφρασης που έχει πληκτρολογήσει ο χρήστης και, αν δεν υπάρχουν προβλήματα, την επιστρέφει.
3. ***selectPhrases:*** Ενεργοποιεί την εισαγωγή μέσω φράσεων.
4. ***selectRegExpr:*** Ενεργοποιεί την εισαγωγή μέσω κανονικής έκφρασης.
5. ***addToExpr:*** Στην εισαγωγή μέσω φράσεων, προσθέτει την έκφραση που έχει επιλέξει ο χρήστης από τη λίστα με τις διαθέσιμες εκφράσεις του συστήματος στο αντίστοιχο πεδίο εισαγωγής.
6. ***doCancel:*** ακυρώνει τη διαδικασία εισαγωγής φράσεων ή κανονικής έκφρασης και άρα τη χρήση της συγκεκριμένης εικόνας στην πρόγνωση.

5.2.9 Κλάση UninstallObserver

1. ***register*****:** δημιουργεί έναν observer για την παρατήρηση τυχόν ενεργειών που έχει απαιτήσει ο χρήστης από τον περιηγητή ή της εξόδου του από την εφαρμογή (κλείσιμο του browser).
2. ***unregister*****:** κατά το κλείσιμο του παραθύρου του περιηγητή διαγράφει τον observer που δημιούργησε η register.
3. ***observe*****:** ενεργοποιείται όταν γίνεται μία από τις αλλαγές που αναφέρθηκαν προηγουμένως και τροποποιεί κάποιες παραμέτρους ώστε αν ο χρήστης έχει ζητήσει την απεγκατάσταση της επέκτασης UranuS και πατήσει κλείσιμο του παραθύρου του περιηγητή, να κληθεί η *uninstallPreferences*.

4. ***uninstallPreferences*:** διαγράφει όλες τις προτιμήσεις του Mozilla Firefox που σχετίζονται με το σύστημα UranuS.

6 *Υλοποίηση*

Στην ενότητα αυτή συζητούνται λεπτομερώς θέματα υλοποίησης του συστήματος. Στην πρώτη παράγραφο αναλύονται θέματα της διπλωματικής που παρουσιάζουν ιδιαίτερο ενδιαφέρον, ενώ στη δεύτερη παράγραφο παρουσιάζονται οι τεχνολογίες και τα προγραμματιστικά εργαλεία που χρησιμοποιήθηκαν για την υλοποίηση, οι πλατφόρμες εκτέλεσης και η διαδικασία εγκατάστασής του.

6.1 Λεπτομέρειες υλοποίησης

6.1.1 Διαχείριση των προτιμήσεων χρήστη

6.1.1.1 Γενικά για τις προτιμήσεις στον Mozilla Firefox

Κατά την υλοποίηση του συστήματος UranuS, για την αποθήκευση των προτιμήσεων του χρήστη χρησιμοποιήθηκε η Διεπαφή Προγραμματισμού Εφαρμογών του Mozilla Firefox για τις προτιμήσεις (Preferences Application Programming Interface – Preferences API). Το Preferences API είναι αρκετά εύκολο στη χρήση του και παρέχει ένα σύνολο χρήσιμων χαρακτηριστικών για χρήστες και προγραμματιστές, όπως την υποστήριξη των προκαθορισμένων (default) προτιμήσεων, των καθορισμένων από το χρήστη προτιμήσεων και του κλειδώματος προτιμήσεων. Πιο συγκεκριμένα, υπάρχουν οι εξής κατηγορίες για τις τιμές των προτιμήσεων:

- **Προκαθορισμένες Τιμές:** αυτές δηλώνονται σε αρχεία .js (JavaScript αρχεία) αποθηκευμένα στον κατάλογο defaults/pref μέσα στον κατάλογο αρχείων της εφαρμογής Mozilla ή στους καταλόγους defaults/preferences των διαφόρων επεκτάσεων. Εκεί δηλαδή δηλώνονται προκαθορισμένες τιμές για ορισμένες προτιμήσεις, οι οποίες χρησιμοποιούνται στην περίπτωση που δεν έχουν δοθεί τιμές από το χρήστη.

- **Τρέχουσες Τιμές:** βρίσκονται αποθηκευμένες στο αρχείο prefs.js στον κατάλογο του συγκεκριμένου προφίλ του περιηγητή Firefox.

Το Preferences API προσφέρει επίσης ένα σύνολο XPCOM διεπαφών για τη διαχείριση των προτιμήσεων, από τις οποίες οι σημαντικότερες είναι η *nsIPrefService*, η *nsIPrefBranch* και η *nsIPrefBranch2*, η οποία επεκτείνει την *nsIPrefBranch*. Από αυτές, η πρώτη ουσιαστικά αποτελεί υπηρεσία του Mozilla, η οποία χρησιμεύει ως βασικό σημείο εισόδου στη βιβλιοθήκη διαχείρισης προτιμήσεων. Μέσω αυτής γίνεται η πρόσβαση στις άλλες δύο διεπαφές, οι οποίες επιτρέπουν την άμεση διαχείριση των προτιμήσεων

Οι προτιμήσεις προσδιορίζονται από ένα όνομα, το οποίο αποτελείται από συμβολοακολουθιες, χωρισμένες με τελείες. Οι συμβολοακολουθίες και ο τρόπος που χωρίζονται από τις τελείες δεν έχουν κάποιο ιδιαίτερο νόημα, παρά μόνο για την οργάνωση των προτιμήσεων σε κλάδους (branches). Για παράδειγμα, οι προτιμήσεις “uranus.wizard.selectedProfile” και “uranus.wizard.maxResults” μπορούν να προσπελαστούν η καθεμιά με το όνομά της, ή με ενιαίο τρόπο μέσω του κλάδου “uranus.wizard.”. Έτσι, συνήθως οι προτιμήσεις που έχουν κάποια σχέση μεταξύ τους έχουν κοινό πρόθεμα. Ο βασικός κλάδος (ρίζα) των προτιμήσεων λαμβάνεται μέσω της μεθόδου *QueryInterface* της υπηρεσίας *nsIPrefService*, ενώ οι διάφοροι υπο-κλάδοι λαμβάνονται καλώντας τη μέθοδο *getBranch*.

Υπάρχουν τέσσερις τύποι προτιμήσεων:

- συμβολοακολουθίες (strings)
- ακέραιες αριθμητικές τιμές
- Boolean τιμές
- τιμές σύνθετου τύπου

Οι τελευταίες χρησιμοποιούνται για τη διαχείριση Unicode συμβολοακολουθιών, δηλαδή ακολουθιών που μπορεί να περιέχουν non-ASCII χαρακτήρες, και για την αποθήκευση μονοπατιών αρχείων.

Στο σύστημα UranuS, οι προτιμήσεις που αποθηκεύονται, χωρίζονται σε δύο κατηγορίες: στις προτιμήσεις *freemeteo* και στις προτιμήσεις *wizard*. Από αυτές, οι πρώτες σχετίζονται με το προφίλ συστήματος και περιλαμβάνουν:

- τις Boolean προτιμήσεις που αφορούν στην εμφάνιση της γενικής περιγραφής, της θερμοκρασίας, του ανέμου, του συνολικού υετού και της υγρασίας στην πρόγνωση.
- τις Boolean προτιμήσεις που αφορούν στη χρήση εικονιδίων και στην εμφάνιση των νυχτερινών προγνώσεων στη γραμμή κατάστασης.
- τις αριθμητικές προτιμήσεις που αφορούν στο συνολικό αριθμό των ημερών πρόγνωσης και στον αριθμό των ημερών πρόγνωσης της γραμμής κατάστασης.

- την αριθμητική προτίμηση που περιλαμβάνει τον κωδικό αριθμό της πόλης για την οποία λαμβάνονται οι προγνώσεις (κωδικός gid, με βάση την κωδικοποίηση των πόλεων στην τοποθεσία freemeteo.com).

Από τις προτιμήσεις της κατηγορίας *wizard*, ένα μέρος σχετίζεται με γενικές λειτουργίες του προγράμματος (όπως ο χρόνος που μεσολαβεί μέχρι να γίνει ανανέωση των μετεωρολογικών δεδομένων, το όνομα του επιλεγμένου προφίλ, ο μέγιστος αριθμός αποτελεσμάτων που μπορούν να παρουσιαστούν στο χρήστη κατά το τρέξιμο μιας σελίδας για τη δημιουργία ή τροποποίηση ενός προφίλ κ.α.). Στη μεγάλη τους πλειοψηφία όμως σχετίζονται με προτιμήσεις των διαφόρων προφίλ χρήστη.

6.1.1.2 Προκαθορισμένες τιμές προτιμήσεων

Παρακάτω παρατίθεται το αρχείο uranus.js του καταλόγου defaults/preferences της επέκτασης, στο οποίο βρίσκονται αποθηκευμένες οι προκαθορισμένες τιμές των προτιμήσεων.

```
//-----------------------------wizard prefs-------------------------------//
//general prefs
pref("uranus.wizard.iconsBool",false);      //icons changed in wizardEditing
pref("uranus.wizard.regexpsBool",false);    //regxps changed in wizardEditing

pref("uranus.wizard.maxresults", 10);       //maximum number of results shown
pref("uranus.wizard.timeout", 60);          //timeout for httpRequest
pref("uranus.wizard.selectedProfile", "freemeteo profile");
pref("uranus.wizard.overwriteProfiles", false);   //overwrite without asking

//-------------------------freemeteo prefs-------------------------------//
pref("uranus.freemeteo.regexps.date",
   "/ (Δευτέρα|Τρίτη|Τετάρτη|Πέμπτη|Παρασκευή|Σάββατο|Κυριακή)( \d{1,2}
    \D{3}) /gi");
pref("uranus.freemeteo.regexps.descr",
   "/(αίθριος καιρός|βασικά καθαρός καιρός|βασικά καθαρός ουρανός|αρκετά
    σύννεφα|Αυξημένη συννεφιά|αραιή συννεφιά|συννεφιά|πιθανή βροχή|ασθενής
    βροχή|ασθενή βροχή|ισχυρή βροχή|βροχή κατά περιόδους|βροχές|βροχή|πιθανή
    καταιγίδα|καταιγίδα|πιθανή ισχυρή καταιγίδα|ισχυρή καταιγίδα|χιονόνερο|
    χιόνι|παγετός|παγετό|χαλάζι|ομίχλη|διαστήματα με καθαρό καιρό| ή | και |
    με | αλλά και )+/gi");
pref("uranus.freemeteo.regexps.temp", "/(-?\d{1,2})(?=&deg;C)/gi");
pref("uranus.freemeteo.regexps.wind",
   "/(Β|Ν|Α|Δ|ΒΑ|ΒΔ|ΝΑ|ΝΔ|ΒΒΑ|ΒΒΔ|ΝΝΑ|ΝΝΔ|ΔΒΔ|ΔΝΔ|ΑΒΑ|ΑΝΑ) στα (\d{1,2})
    (Km\/h)/gi");
pref("uranus.freemeteo.regexps.humidity", "/(\d{1,2})% - (\d{1,2})%/gi");
pref("uranus.freemeteo.regexps.yetos", "/(\d{1,2}) mm/gi");

pref("uranus.freemeteo.night_on_bar", true);
pref("uranus.freemeteo.size", 2);
pref("uranus.freemeteo.size_on_bar", 2);
pref("uranus.freemeteo.iconsBool", true);
pref("uranus.freemeteo.gid",264371);
pref("uranus.freemeteo.description", true);
pref("uranus.freemeteo.temperature", true);
pref("uranus.freemeteo.wind", true);
pref("uranus.freemeteo.yetos", true);
pref("uranus.freemeteo.humidity", true);
```

Όπως είναι φανερό, η αποθήκευση των προτιμήσεων γίνεται με την εντολή pref(pref_name, pref_value). Ο τύπος της προτίμησης δε δηλώνεται πουθενά άμεσα, παρά μόνο μέσω του τύπου της τιμής που δίνεται. Σημειώνεται ότι αν θέλαμε στην προκαθορισμένη τιμή μιας προτίμησης να χρησιμοποιήσουμε μη ASCII χαρακτήρες, τότε στο παραπάνω αρχείο ως τιμή της προτίμησης θα έπρεπε να δώσουμε το chrome μονοπάτι όπου θα βρισκόταν το εν-τοπισμένο αρχείο default.properties, το οποίο θα περιείχε την αντίστοιχη τιμή προτίμησης στην κατάλληλη κάθε φορά γλώσσα. Για παράδειγμα αυτό το μονοπάτι για το σύστημά μας θα ήταν "chrome://uranus/locale/defaults.properties".

6.1.1.3 Τρέχουσες τιμές προτιμήσεων

Οι μέθοδοι για τη λήψη και τον ορισμό των τιμών των προτιμήσεων για κάθε τύπο είναι οι εξής: *getCharPref*, *setCharPref*, *getIntPref*, *setIntPref*, *getBoolPref*, *setBoolPref*, *getComplexValue*, *setComplexValue*. Από αυτές, οι 6 πρώτες είναι απλές, δεχόμενες ως ορίσματα μόνο το όνομα της προτίμησης (οι μέθοδοι λήψης) ή το όνομα της προτίμησης και την αντίστοιχη τιμή (οι μέθοδοι ορισμού τιμών). Η *getComplexValue* από την άλλη δέχεται ως παραμέτρους το όνομα της προτίμησης και τον τύπο των σύνθετων δεδομένων, ενώ η αντίστοιχη συμβολοακολουθία περιλαμβάνεται στο γνώρισμα data της επιστρεφόμενης τιμής. Η *setComplexValue* δέχεται τα ίδια ορίσματα, και επιπλέον ένα ειδικό αντικείμενο, στο γνώρισμα data του οποίου βρίσκεται αποθηκευμένη η επιθυμητή συμβολοακολουθία. Σημειώνεται ότι στο σύστημά μας χρησιμοποιήθηκε μόνο ο σύνθετος τύπος *nsIPrefLocalizedString* που χρησιμεύει στην αποθήκευση strings με μη ASCII χαρακτήρες των οποίων οι προκαθορισμένες τιμές, βρίσκονται σε κάποιο εν-τοπισμένο αρχείο.

Άλλες χρήσιμες μέθοδοι για τη διαχείριση των προτιμήσεων είναι οι *lockPref*, *getChildList*, *prefHasUserValue*, *clearUserPref*, *deleteBranch* και *addObserver*, με λειτουργίες η οποία μπορούν να συναχθούν εύκολα από το όνομά τους. Συγκεκριμένα, η τελευταία μέθοδος παρουσιάζει ιδιαίτερο ενδιαφέρον και χρησιμοποιήθηκε κατά κόρον στο σύστημά μας, αφού βοηθά στην άμεση ενημέρωση των κλάσεων που είναι ενεργοποιημένες την τρέχουσα χρονική στιγμή για τις αλλαγές που γίνονται στις τιμές των προτιμήσεων, και την κατάλληλη τροποποίηση των ρυθμίσεων και των παραμέτρων τους με τη βοήθεια της ειδικής μεθόδου *observe*.

6.1.1.4 Διαχείριση των προτιμήσεων στο σύστημα **UranuS**

Ακολουθούν δύο παραδείγματα διαχείρισης των προτιμήσεων μέσα από τον κώδικα του συστήματος UranuS. Το πρώτο είναι η μέθοδος *startup* της κλάσης UranuS, στην οποία όπως αναφέρθηκε στο προηγούμενο κεφάλαιο γίνεται λήψη των προτιμήσεων χρήστη και αρχικοποιούνται οι παράμετροι της κλάσης. Ως παράδειγμα αποθήκευσης προτιμήσεων δίνεται απόσπασμα από τη μέθοδο *savePrefernces* της κλάσης *Wizard*.

Μέθοδος startup

```
1  startup: function()
2  {
3      document.getElementById("freemeteoMenu").hidden = true;
4      this.prefSvc =
5                  Components.classes["@mozilla.org/preferences-service;1"]
6                   .getService(Components.interfaces.nsIPrefService);
7      this.prefs = Uranus.prefSvc.getBranch("uranus.wizard.");
9      Uranus.prefs.QueryInterface(Components.interfaces.nsIPrefBranch2);
9      Uranus.prefs.addObserver("", this, false);
10
11     /*get the profile*/
12     Uranus.profile = Uranus.prefs.getCharPref("selectedProfile");
13     try
14     {
15         document.getElementById("edit_prof").label =
16                                     "Edit Profile '" + Uranus.profile + "'";
17     }
18     catch(error)
19     {
20         document.getElementById("edit_prof").label = "Edit Profile";
21     }
22     if (Uranus.profile == "freemeteo profile")
23     {
24          usermode = false;
25          UranusDef.startup();
26          Uranus.shutdown();
27          return;
28      }
29      usermode = true;
30      this.timeoutValue = this.prefs.getIntPref("timeout");
31      Uranus.profileBranch = Uranus.prefSvc.getBranch("uranus.wizard." +
32                                                      this.profile + ".");
33
34      /*get the weather characteristics and icons preferences*/
35      Uranus.genBool = Uranus.profileBranch.getBoolPref("genBool");
36      Uranus.tempBool = Uranus.profileBranch.getBoolPref("tempBool");
37      Uranus.windBool = Uranus.profileBranch.getBoolPref("windBool");
38      Uranus.excludeTags= Uranus.profileBranch.getBoolPref("excludeTags");
39
40      /*get the 'split', 'split_expr', 'url', and 'region' preferences*/
41      Uranus.url = Uranus.profileBranch.getCharPref("url");
42      Uranus.split = Uranus.profileBranch.getBoolPref("split");
43      Uranus.split_expr =
44                  eval(Uranus.profileBranch.getComplexValue("split_expr",
45              Components.interfaces.nsIPrefLocalizedString).data);
46      Uranus.region = Uranus.profileBranch.getComplexValue("region",
47                  Components.interfaces.nsIPrefLocalizedString).data;
48
49      /*get the Regular Expressions for each weather characteristic*/
50      Uranus.getExpressions();
51      Uranus.getIcons();
52      Uranus.refreshInformation();
53      var interval =
54              self.setInterval("Uranus.refreshInformation()", 12*60*1000);
55  }
```

Όπως βλέπουμε, στις γραμμές 4-9 του κώδικα γίνεται λήψη της υπηρεσίας nsIPrefService και μέσω αυτής του κλάδου “uranus.wizard” των προτιμήσεων χρήστη , στον οποίο εφαρμόζεται η επαφή *nsIPrefService* και συνδέεται ένας observer για την παρατήρηση τυχόν αλλαγών στις τιμές των προτιμήσεων του κλάδου και ειδοποίηση της κλάσης. Στη συνέχεια, στις γραμμές 12 και 30 γίνεται λήψη 2 προτιμήσεων, μιας συμβολοακολουθίας και μίας ακεραίας τιμής. Στις γραμμές 31-32 εκχωρείται σε μία

μεταβλητή ο κλάδος που αντιστοιχεί στο τρέχον προφίλ και στη συνέχεια στις γραμμές 34-52 λαμβάνονται όλες οι προτιμήσεις για το προφίλ αυτό (οι δύο βοηθητικές μέθοδοι *getExpressions* και *getIcons* περιλαμβάνουν εντολές που πραγματοποιούν λήψη όλων των κανονικών εκφράσεων και όλων των ρυθμίσεων εικονιδίων αντίστοιχα). Ιδιαίτερο ενδιαφέρον παρουσιάζει ο τρόπος χρήσης της μεθόδου *getComplexValue* στις γραμμές 44-45 και 46-47 για τη λήψη των τιμών των προτιμήσεων *split_expr* και *region*. Για τις προτιμήσεις αυτές επιλέχθηκε ο σύνθετος τύπος *nsIPrefLocalizedString* επειδή είναι πολύ πιθανό ο χρήστης να θελήσει να χρησιμοποιήσει ελληνικούς χαρακτήρες για τις τιμές τους.

Απόσπασμα από τη μέθοδο savePreferences

```
 1  var groups = this.prefSvc.getBranch("uranus.wizard."+
 2                                                  prof_name + ".group");
 3  groups.deleteBranch("");
 4  var pls = Components.classes["@mozilla.org/pref-localizedstring;1"]
 5            .createInstance(Components.interfaces.nsIPrefLocalizedString);
 6  for (var i=0; i<this.gen.length; i++)
 7  {
 8        pls.data = this.gen[i].source;
 9        this.prefs.setComplexValue(prof_name + ".gen." + i,
10                        Components.interfaces.nsIPrefLocalizedString, pls);
11  }
12  for (var i=0; i<this.temp.length; i++)
13  {
14        pls.data = this.temp[i].source;
15        this.prefs.setComplexValue(prof_name + ".temp." + i,
16                        Components.interfaces.nsIPrefLocalizedString, pls);
17  }
18  for (var i=0; i<this.wind.length; i++)
19  {
20        pls.data = this.wind[i].source;
21        this.prefs.setComplexValue(prof_name + ".wind." + i,
22                        Components.interfaces.nsIPrefLocalizedString, pls);
23  }
24  this.prefs.setCharPref(prof_name + ".url",url);
25  this.prefs.setBoolPref(prof_name + ".split",split);
26  pls.data = split_xpr;
27  this.prefs.setComplexValue(prof_name + ".split_expr",
28                        Components.interfaces.nsIPrefLocalizedString, pls);
29  pls.data = region;
30  this.prefs.setComplexValue(prof_name + ".region",
31                        Components.interfaces.nsIPrefLocalizedString, pls);
32  this.prefs.setBoolPref(prof_name + ".genBool",gen);
33  this.prefs.setBoolPref(prof_name + ".tempBool",temp);
34  this.prefs.setBoolPref(prof_name + ".windBool",wind);
35  this.prefs.setBoolPref(prof_name + ".excludeTags",excludeTags);
36
37  if (useThis)
38  {
39        this.prefs.setCharPref("selectedProfile", prof_name);
40        this.prefs.setBoolPref("userDefined", true);
41  }
```

Η μέθοδος *savePreferences* δε δόθηκε ολόκληρη γιατί είναι μεγάλη σε έκταση θεωρήθηκε σκόπιμο να παρατεθούν μόνο τα τμήματά της που παρουσιάζουν ενδιαφέρον σε σχέση με την αποθήκευση τιμών προτιμήσεων. Θεωρούμε λοιπόν ότι οι διάφορες μεταβλητές που υπάρχουν στον παραπάνω κώδικα έχουν

λάβει τις τιμές τους σε κάποιο κομμάτι αρχικοποίησης της μεθόδου. Παρατηρούμε ότι στις γραμμές 1-3 γίνεται λήψη και διαγραφή του κλάδου που αντιστοιχεί στο προφίλ του οποίου το όνομα δόθηκε από το χρήστη (μέθοδος *deleteBranch*). Στη συνέχεια, στη γραμμή 4 δημιουργείται το ειδικό αντικείμενο *pls* τύπου *nsIPrefLocalizedString*, το οποίο θα χρησιμοποιηθεί από τις κλήσεις της μεθόδου setComplexValue. Η μέθοδος αυτή καλείται διαδοχικά στις γραμμές 6-11, 12-17 και 18-23 για την αποθήκευση των κανονικών εκφράσεων που αντιστοιχούν σε κάθε χαρακτηριστικό του καιρού (βρίσκονται αποθηκευμένες στους πίνακες *gen*, *temp* και *wind* της κλάσης). Κάθε φορά, αποθηκεύεται στο *pls.data* η αντίστοιχη συμβολοακολουθία (το περιεχόμενο της κανονικής έκφρασης, το οποίο λαμβάνεται με το γνώρισμα *source* της κανονικής έκφρασης) και στη συνέχεια καλείται η setComplexValue, όπως περιγράφηκε νωρίτερα. Δύο ακόμη κλήσεις της ίδιας μεθόδου έχουμε στις γραμμές 27-28 και 30-31. Στο υπόλοιπο τμήμα του κώδικα που παρατίθεται βλέπουμε κλήσεις των μεθόδων *setBoolPref* και *setCharPref*.

6.1.1.5 Τρόπος αποθήκευσης των προτιμήσεων εξαγωγής & παρουσίασης πληροφορίας για κάθε προφίλ χρήστη

Ένα σημαντικό θέμα κατά την υλοποίηση του συστήματος ήταν να βρεθεί ένας αποδοτικός τρόπος για να αποθηκεύονται στις προτιμήσεις του χρήστη όλες οι απαραίτητες πληροφορίες που αφορούν στην εξαγωγή και παρουσίαση των δεδομένων για ένα συγκεκριμένο προφίλ Οι έννοιες που μας ενδιαφέρουν σχετικά με μια πληροφορία που έχει επιλεχθεί από το χρήστη ως επιθυμητή είναι οι εξής:

- σε ποιο χαρακτηριστικό του καιρού (“*γενική περιγραφή*”, “*θερμοκρασία*” ή “*άνεμος*”) ανήκει
- σε ποιο από τα τμήματα (*fragments*) του αρχικού κειμένου της σελίδας βρίσκεται (όταν είναι ενεργοποιημένη η προτίμηση “*split-by*”)
- ποιο ταίριασμα (*match*) στη σειρά αντιστοιχεί σε αυτή την πληροφορία (π.χ. το 1ο, το 4ο κλπ.). Το ταίριασμα αναφέρεται στην εφαρμογή όλων των κανονικών εκφράσεων που έχουν συνδεθεί με το συγκεκριμένο χαρακτηριστικού καιρού στο προφίλ, με τη σειρά που αυτές είναι δοσμένες
- ποια είναι η ετικέτα (*label*) της πληροφορίας
- σε ποια ομάδα περιεχομένου (*context group*) ανήκει
- πώς θα παρουσιάζεται στο χρήστη (ως στοιχείο της γραμμής κατάστασης, ως στοιχείο κυλιόμενου κειμένου ή μόνο στο παράθυρο πρόγνωσης)
- ποια θα είναι η σειρά εμφάνισής της μέσα στην ομάδα περιεχομένου στην οποία ανήκει, και φυσικά
- ποια θα είναι η σειρά εμφάνισης της εν λόγω ομάδας περιεχομένου σε σχέση με τις υπόλοιπες.

Όπως είναι προφανές, τα στοιχεία που σχετίζονται με κάθε πληροφορία είναι αρκετά και η ξεχωριστή τους αποθήκευση θα οδηγούσε σε τεράστιο όγκο δεδομένων και αντίστοιχα σε μεγάλο αριθμό χρησιμοποιούμενων μεταβλητών άλλα και διαδοχικών εντολών πρόσβασης στις προτιμήσεις. Ακόμη, κάποια από τα στοιχεία αυτά, όπως η ομάδα περιεχομένου και η σειρά εμφάνισής της, είναι κοινά μεταξύ διαφορετικών πληροφοριών, γεγονός που μπορεί να οδηγήσει σε επανάληψη πληροφορίας. Έτσι, αναζητήθηκε ένας τρόπος οργάνωσης όλων των παραπάνω στοιχείων για κάθε πληροφορία, ώστε να γίνεται εύκολη αποθήκευση και ανάκτηση των προτιμήσεων και παράλληλα η χρησιμοποίησή τους από το πρόγραμμα εξαγωγής πληροφορίας.

Ο λύση που επιλέχθηκε τελικά ήταν η οργάνωση και αποθήκευση των πληροφοριών ανά ομάδες περιεχομένου. Για κάθε ομάδα περιεχομένου αποθηκεύονται δύο τιμές προτιμήσεων: μία τιμή τύπου string στην οποία κρατείται το όνομα της ομάδας, και μία τιμή επίσης τύπου string στην οποία κρατείται το περιεχόμενο της ομάδας, δηλαδή όλες οι πληροφορίες και οι σχετικές με αυτές επιλογές. Συγκεκριμένα, το περιεχόμενο κάθε ομάδας αποθηκεύεται ως γραμμικός πίνακας που αποτελείται από υπο-πίνακες (πληροφορίες ομάδας), ως εξής:

$$\left[\left[match1, label1, status\_bar1\right], ..., \left[matchN, labelN, status\_barN\right]\right]$$

Όπως φαίνεται παραπάνω, κάθε πληροφορία αποτελείται από τρία στοιχεία. Το πρώτο είναι μια σύνθετη τιμή που απαρτίζεται από ένα προσδιοριστικό του χαρακτηριστικού του καιρού στο οποίο αναφέρεται η πληροφορία (χρησιμοποιούνται τα αυστηρά τα προσδιοριστικά “*gen*”, “*temp*” και “*wind*”, για τη γενική περιγραφή του καιρού, τη θερμοκρασία και τον άνεμο αντίστοιχα), τον αριθμό του fragment του κειμένου στο οποίο βρίσκεται αυτή (δίνεται η τιμή 1 στην περίπτωση που δεν έχει επιλεγεί διαχωρισμός του κειμένου με βάση κάποια έκφραση) και τη σειρά του ταιριάσματος για το χαρακτηριστικό αυτό. Οι τρείς αυτές πληροφορίες διαχωρίζονται μεταξύ τους από τον χαρακτήρα underscore (“_”). Το δεύτερο στοιχείο της πληροφορίας είναι η ετικέτα που της έχει δοθεί από το χρήστη, ενώ τέλος το τρίτο είναι ο τρόπος παρουσίασής της (χρησιμοποιείται μία από τις τρεις επιλογές “*tooltip*”, *status-bar*” ή “*null*” για την εμφάνιση στη γραμμή κατάστασης μόνο ως κυλιόμενο κείμενο, την εμφάνιση ως στοιχείο της γραμμής ή την εμφάνιση μόνο στο παράθυρο πρόγνωσης και όχι στη γραμμή κατάστασης).

Η σειρά εμφάνισης των πληροφοριών μέσα σε μια ομάδα περιεχομένου καθορίζεται από τη σειρά με την οποία δίνονται στον αντίστοιχο πίνακα στις προτιμήσεις. Από την άλλη, η σειρά μεταξύ των ομάδων καθορίζεται από το όνομα που τους δίνεται. Συγκεκριμένα, το όνομα κάθε προτίμησης έχει τη μορφή

uranus.wizard.(profileName).group.(groupSeq),

όπου *(profileName)* είναι το όνομα του προφίλ χρήστη και *(groupSeq)* είναι η σειρά εμφάνισης της ομάδας. Για την αποθήκευση του ονόματος της ομάδας χρησιμοποιείται η επέκταση “*tag*” και για το περιεχόμενό της η επέκταση “*array*”.

Πρέπει να σημειωθεί τέλος ότι, καθώς δεν είναι δυνατή η αποθήκευση πινάκων στη διεπαφή προτιμήσεων του Mozilla, οι πίνακες αποθηκεύονται ως συμβολοακολουθίες και κατά την ανάκτησή τους από το πρόγραμμα, γίνεται η μετατροπή τους σε πίνακες από μια βοηθητική μέθοδο της κλάσης *Uranus*. Συγκεκριμένα, η μέθοδος αυτή δημιουργεί έναν πίνακα με στοιχεία όχι πίνακες, αλλά ειδικά αντικείμενα, τύπου *GroupItem*, που έχουν οριστεί κατάλληλα, ώστε τα στοιχεία τους *match*, *label* και *status_bar* να προσπελαύνονται τοποθετώντας την αντίστοιχη κατάληξη (π.χ. για τη μεταβλητή *item* τύπου *GroupItem* η προσπέλαση της ετικέτας του γίνεται με την εντολή "*item.label*")

6.1.2 Εξαγωγή πληροφορίας από την ιστοσελίδα-πηγή

6.1.2.1 Ένα απλό παράδειγμα χρήσης κανονικών εκφράσεων στη JavaScript

Όπως έχει ήδη αναφερθεί, η πιο ουσιαστική λειτουργία της εφαρμογής είναι η εξαγωγή των πληροφοριών που έχει χαρακτηρίσει ο χρήστης ως επιθυμητών, από την ιστοσελίδα-πηγή. Η εξαγωγή γίνεται με τη χρήση κανονικών εκφράσεων, γραμμένων στη γλώσσα JavaScript. Παρακάτω δίνεται ένα παράδειγμα για τον ορισμό κανονικών εκφράσεων σε κώδικα JavaScript και την εφαρμογή τους στο κείμενο μιας ιστοσελίδας. Ας θεωρήσουμε για παράδειγμα το εξής τμήμα μιας ιστοσελίδας από την τοποθεσία freemeteo.com:

Θα ορίσουμε μια κανονική έκφραση για την περιγραφή του ανέμου και μία για την περιγραφή της θερμοκρασίας και θα δώσουμε τα αποτελέσματα της εξαγωγής πληροφορίας από το παραπάνω τμήμα της ιστοσελίδας. Έχουμε λοιπόν τον εξής κώδικα:

```
 1  handlePage(response)
 2  {
 3        var temp_regexp = /(\d{1,2})(?=&deg;C)/gi;
 4        var wind_regexp = new RegExp("(Β|Ν|Α|Δ|ΒΑ|ΒΔ|ΝΑ|ΝΔ|ΒΒΑ|
 5                         ΒΒΔ|ΝΝΑ|ΝΝΔ) με (\\d{1,2}) Km/h","gi");
 6
 7        var temp_array = new Array();
 8        var wind_array = new Array();
 9        var temp_results;
10        var wind_results;
11
12        temp_results = response.match(temp_regexp);
13        if (temp_results)
14        {
15                temp_array[0] = RegExp.lastMatch;
16                temp_array [1] = RegExp.$1;
17        }
18
19        var     wind_results = response.exec(wind_regexp);
20        if (wind_results)
21        {
22                wind_array [0] = RegExp.lastMatch;
23                wind_array [1] = RegExp.$1;
24                wind_array [2] = RegExp.$2;
25        }
26        alert("temp_results        : " + temp_results
27          + "\ntemp_array           : " + temp
28          + "\nwind_results         : " + wind_results
29          + "\nwind_array           : " + wind);
30  }
```

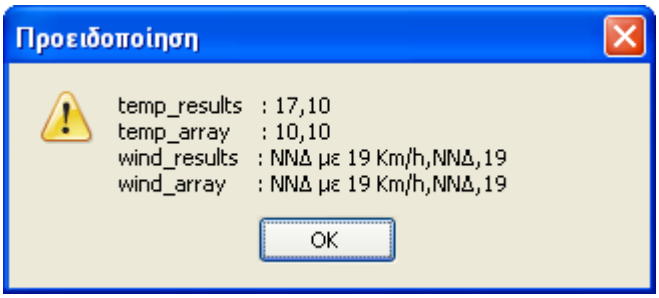
Καταρχάς, στις γραμμές 3-5 γίνεται η δήλωση των κανονικών εκφράσεων. Μια κανονική έκφραση μπορεί να οριστεί στη γλώσσα JavaScript με δύο τρόπους. Ο πρώτος είναι αυτός που φαίνεται στη γραμμή 3 για την temp_regexp, δηλαδή η παράθεση της έκφρασης ανάμεσα σε δύο χαρακτήρες “/”, ακολουθούμενη ενδεχομένως από ειδικούς χαρακτήρες. Η έκφραση που έχει δοθεί δηλώνει ότι αναζητείται μια φράση που αποτελείται από 1 ή 2 συνεχόμενα ακέραια ψηφία, ακολουθούμενα από ένα κενό και την παράσταση “&degC;”, η οποία χρησιμοποιείται για να αναπαραστήσει το σύμβολο “°” στη γλώσσα HTML. Μάλιστα, η πρώτη παρένθεση δηλώνει ότι θέλουμε το πρόγραμμα να θυμάται ξεχωριστά το κομμάτι της εισόδου που ταιριάζει με το περιεχόμενό της, ενώ η δεύτερη, επειδή ξεκινά με τα σύμβολα “?=” δηλώνει ότι δε θέλουμε να θυμάται το αντίστοιχο κομμάτι. Ο ειδικός χαρακτήρας “i” που δίνεται στο τέλος της έκφρασης δηλώνει ότι η αναζήτηση θέλουμε να είναι “case indifferent” δηλαδή να μην διακρίνει τα πεζά από τα κεφαλαία γράμματα. Ο δεύτερος ειδικός χαρακτήρας, ο χαρακτήρας “g”, δηλώνει ότι η αναζήτηση θέλουμε να είναι “global”, δηλαδή να μη σταματά μόλις γίνει ένα ταίριασμα αλλά να συνεχίζει μέχρι να φτάσει στο τέλος του κειμένου εισόδου.

Ο δεύτερος τρόπος για να δηλωθεί μια κανονική έκφραση είναι αυτός που φαίνεται στην 4η και 5η γραμμή. Σε αυτή την περίπτωση η έκφραση δημιουργείται ως ένα νέο αντικείμενο RegExp. Το πρώτο όρισμα που δίνεται είναι το περιεχόμενο της κανονικής έκφρασης σε μορφή string, ενώ το δεύτερο είναι οι ειδικοί χαρακτήρες, πάλι σε μορφή string. Η έκφραση που δίνεται για την περιγραφή του ανέμου δηλώνει ότι αναζητείται μια από τις συμβολοακολουθίες “Β”, “Ν”, “Α”, “Δ”, “ΒΒΑ” κ.ο.κ. το ταίριασμα των οποίων μάλιστα θέλουμε να θυμόμαστε, ακολουθούμενο από τη φράση “ με ”, ένα ή δύο ακέραια ψηφία (τα οποία επίσης θέλουμε να θυμόμαστε), και τη φράση “ km/h”. Αξίζει να σημειωθεί ότι σε αυτή την περίπτωση, για τη δήλωση των δεκαδικών ψηφίων χρησιμοποιείται διπλός χαρακτήρας “\” εξαιτίας του γεγονότος ότι έχουμε να κάνουμε με string.

Ο τρόπος κλήσης μιας κανονικής έκφρασης στη JavaScript μπορεί και πάλι να γίνει με δύο τρόπους, ανεξάρτητα από τον τρόπο με τον οποίο δηλώθηκε αυτή. Ο πρώτος τρόπος είναι μέσω της μεθόδου *match()* των αντικειμένων τύπου sting και ο δεύτερος μέσω της μεθόδου *exec()* του ειδικού αντικειμένου RegExp. Και οι δύο μέθοδοι στην περίπτωση που επιτύχουν ταίριασμα επιστρέφουν ένα πίνακα με τα αποτελέσματα και ενημερώνουν τα γνωρίσματα του αντικειμένου RegExp, ενώ αν αποτύχουν επιστρέφουν null. Τα πιο βασικά γνωρίσματα του αντικειμένου RegExp είναι το “*lastMatch*” και τα γνωρίσματα που αποτελούνται από το χαρακτήρα του δολαρίου (“*$*”), ακολουθούμενο από ένα δεκαδικό ψηφίο (από 1 έως 9). Το *lastMatch* περιέχει τους χαρακτήρες του τελευταίου ταιριάσματος που έγινε, ενώ τα άλλα γνωρίσματα αποτελούν αναφορές στους χαρακτήρες που αντιστοιχούν σε κάθε παρένθεση της κανονικής έκφρασης (ο αριθμός δηλώνει τη σειρά της παρένθεσης). Οι αναφορές αυτές χαρακτηρίζονται ως “backreferences”. Οι διαφορά μεταξύ των δύο μεθόδων κλήσης μιας κανονικής έκφρασης είναι στον πίνακα που επιστρέφουν. Η πρώτη επιστρέφει έναν πίνακα με όλα τα ταιριάσματα που πραγματοποιήθηκαν, ενώ ο πίνακας της δεύτερης περιέχει το τελευταίο μόνο ταίριασμα και όλα τα backreferences.

Στον παραπάνω κώδικα βλέπουμε ότι έχουμε την κλήση της πρώτης κανονικής έκφρασης μέσω της μεθόδου *match* στις γραμμές 12-17, και την κλήση της δεύτερης μέσω της *exec* στις γραμμές 20-25. Και στις δύο περιπτώσεις καλείται πρώτα η αντίστοιχη μέθοδος και αν είναι επιτυχής, εκχωρούνται στις θέσεις ενός πίνακα (temp_array ή wind_array αντίστοιχα) το γνώρισμα *lastMatch* του αντικειμένου RegExp και τα backreferences που αντιστοιχούν στην τρέχουσα κανονική έκφραση.

Τέλος, εμφανίζεται ένα μήνυμα από το πρόγραμμα που εμφανίζει για κάθε έκφραση τον πίνακα της κλήσης της (temp_results και wind_results αντίστοιχα) και τον πίνακα _array αυτής. Το αποτέλεσμα της κλήσης της μεθόδου *handlePage* στο απόσπασμα της ιστοσελίδας που παραθέσαμε παραπάνω είναι το εξής:



Βλέπουμε ότι στους πίνακες _array τόσο της θερμοκρασίας όσο και του ανέμου περιέχεται το τελευταίο ταίριασμα που πραγματοποιήθηκε και οι αναφορές των παρενθέσεων της κανονικής έκφρασης, όπως ακριβώς το είχαμε ορίσει. Στον πίνακα των αποτελεσμάτων της θερμοκρασίας όμως περιέχονται όλα τα ταιριάσματα που έγιναν, ενώ σε αυτόν του ανέμου περιέχονται τα ίδια ακριβώς δεδομένα με τον πίνακα _array.

6.1.2.2 Η εξαγωγή πληροφορίας στο σύστημα UranuS

Παρακάτω δίνεται ο κώδικας της εφαρμογής που υλοποιεί τη λειτουργία της εξαγωγής πληροφορίας από την ιστοσελίδα-πηγή. Συγκεκριμένα δίνεται μόνο το κομμάτι της μεθόδου *handleRequest* της κλάσης *Uranus* στο οποίο γίνεται η εξαγωγή της πληροφορίας και η βοηθητική μέθοδος *findMatch* η οποία καλείται εσωτερικά από την *handleRequest*. Θεωρείται ότι το κείμενο που έχει ληφθεί ως απόκριση στο *httpRequest* (δηλ. η HTML σελίδα) έχει αποθηκευθεί στη μεταβλητή *response*, οι γενικές προτιμήσεις του χρήστη για το συγκεκριμένο προφίλ έχουν αποθηκευθεί στα αντίστοιχα γνωρίσματα της κλάσης, ενώ οι προτιμήσεις του σχετικά με τις πληροφορίες προς εξαγωγή βρίσκονται αποθηκευμένες στον πίνακα *groups*. Ο πίνακας αυτός έχει ως στοιχεία τους πίνακες για κάθε ομάδα περιεχομένου, δηλ. στη θέση 0 βρίσκεται ολόκληρος ο πίνακας της 1ης ομάδας, στη θέση 1 ο πίνακας για τη δεύτερη ομάδα κ.ο.κ. Σημειώνεται ότι από τον κώδικα που παρατίθεται έχουν παραληφθεί οι εντολές που σχετίζονται με την παρουσίαση των πληροφοριών στο χρήστη.

Απόσπασμα από τη μέθοδο handleRequest

```
/*for each group*/
for (var i=0; i<groups.length; i++)
{
     /*for each groupItem*/
     for (var j=0; j<groups[i].length; j++)
     {
          var arr = groups[i][j].match_id.split("_");
          char_id = arr[0];
          fragment = arr[1];
          num = arr[2];
          var match = null;

          if (eval("this." + char_id + "Bool"))
          {
```

```
                if (this.split && this.split_expr != "")
                {
                     var response2 = response.split(this.split_expr);
                     match = Uranus.findMatch(char_id,num,response2[fragment]);
                }
                else
                {
                     match = Uranus.findMatch(char_id, num, response);
                }
           }
     }
}
```

Όπως φαίνεται παραπάνω, ακολουθείται η ίδια διαδικασία για κάθε στοιχείο κάθε ομάδας περιεχομένου. Αρχικά γίνεται διάσπαση της πληροφορίας *match_id* στα τρία στοιχεία *char_id*, *fragment* και *num* που αντιστοιχούν στο χαρακτηριστικό του καιρού, τον αριθμό του fragment και τον αριθμό του ταιριάσματος, κατά τα όσα αναφέρθηκαν στην προηγούμενη παράγραφο. Στη συνέχεια, ελέγχεται για σιγουριά ότι ο χρήστης έχει επιλέξει να ενημερώνεται για το συγκεκριμένο χαρακτηριστικό (έλεγχος των boolean γνωρισμάτων της κλάσης *genBool*, *tempBool* ή *windBool* ανάλογα την περίπτωση). Αν όντως ο χρήστης έχει περιλάβει το αντίστοιχο χαρακτηριστικό στην πρόγνωσή του, τότε στην περίπτωση που δεν έχει ζητηθεί να γίνει διαχωρισμός του κειμένου καλείται η μέθοδος *findMatch* για ολόκληρο το κείμενο της μεταβλητής *response*. Διαφορετικά, γίνεται διάσπαση του *response* σε fragments με βάση την έκφραση που είναι αποθηκευμένη στο γνώρισμα *split_expr* και καλείται η *findMatch* για το κατάλληλο fragment.

Μέθοδος findMatch

```
findMatch: function findMatch(char_id, num, response)
{
/*do the same thing for each regular expression for this characteristic*/
        var expression = eval("this." + char_id);
        var total_matches = 0;

        var l=0;
        while(l<expression.length && total_matches<num)
        {
                expression[l].lastIndex=0;
                var lastindex = response.length
                while(lastindex != 0 && total_matches < num)
                {
                        match = expression[l].exec(response);
                        lastindex = expression[l].lastIndex;
                        if (match)
                        {
                                total_matches++;
                        }
                }
                l++;
        }
        if (total_matches == num)
        {
                return match[0];
        }
        else
        {
                return null;
        }
}
```

Στη μέθοδο *findMatch* γίνεται ο εντοπισμός μιας επιθυμητής πληροφορίας. Ως παράμετροι δίνονται το προσδιοριστικό του χαρακτηριστικού του καιρού, ο αριθμός του ταιριάσματος (έστω ν) και το κείμενο στο οποίο θα γίνει η αναζήτηση. Αρχικά λαμβάνεται στη μεταβλητή *expression* ο πίνακας κανονικών εκφράσεων που αντιστοιχεί στο χαρακτηριστικό του καιρού. Στα γνωρίσματα της κλάσης *Uranus* έχουν αποθηκευθεί τρεις τέτοιοι πίνακες, *gen*, *temp* και *wind*, οι οποίοι αποτελούνται όλοι από αντικείμενα τύπου RegExp (είναι το αντικείμενο της JavaScript για τις κανονικές εκφράσεις), οπότε εδώ γίνεται εκχώρηση στην *expression* του κατάλληλου από τους τρεις πίνακες. Ακόμη, αρχικοποιείται στο 0 η μεταβλητή *total_matches* στην οποία θα κρατείται ο αριθμός των επιτυχών ταιριασμάτων. Ακολούθως, για κάθε έκφραση που ανήκει στον πίνακα *expression* (ή μέχρι να βρούμε την επιθυμητή πληροφορία) γίνεται αναζήτησή στο κείμενο ώσπου να φτάσουμε στο τέλος του, ή να βρεθεί το ν-στο ταίριασμα συνολικά. Για να εντοπιστεί πότε φτάσαμε στο τέλος του κειμένου χρησιμοποιείται η μεταβλητή *lastIndex*, η οποία αρχικοποιείται κάθε φορά στη θέση του τελευταίου χαρακτήρα του κειμένου για να ξεκινήσει η αναζήτηση από την αρχή και όχι από εκεί που είχε σταματήσει η προηγούμενη κανονική έκφραση,. Ακόμη, το γνώρισμα *lastIndex* της εκάστοτε κανονικής έκφρασης, που δηλώνει τη θέση που έγινε ταίριασμα τελευταία, αρχικοποιείται στο 0, ώστε να μην κρατείται στη μνήμη ένα πιθανό προηγούμενο ταίριασμα.

Η αναζήτηση για ταίριασμα γίνεται μέσω της μεθόδου *exec* της κανονικής έκφρασης, η οποία αν είναι επιτυχής επιστρέφει έναν πίνακα με 1ο στοιχείο την πληροφορία που εντοπίστηκε, ενώ διαφορετικά επιστρέφει null. Κάθε φορά, το αποτέλεσμα της exec εκχωρείται στη μεταβλητή *match*, ενώ η μεταβλητή *lastindex* παίρνει την τιμή της θέσης όπου έγινε το τελευταίο ταίριασμα (στην περίπτωση που δεν έγινε κανένα νέο ταίριασμα η τιμή αυτή γίνεται 0). Αν μάλιστα η *exec* ήταν επιτυχής, η μεταβλητή total_matches αυξάνεται κατά1. Όταν ολοκληρωθεί η διαδικασία της αναζήτησης συνολικά τότε στην περίπτωση που βρέθηκε το ν-οστό ταίριασμα επιστρέφεται η τιμή του πρώτου στοιχείου της *match* (επιθυμητή πληροφορία) ενώ διαφορετικά επιστρέφεται η τιμή null.

6.2 Πλατφόρμες και προγραμματιστικά εργαλεία

6.2.1 Χρησιμοποιούμενες τεχνικές και προγραμματιστικά εργαλεία

Για την ανάπτυξη των στοιχείων διεπαφής χρήστη της εφαρμογής χρησιμοποιήθηκαν οι γλώσσες προγραμματισμού XUL και JavaScript, ακολουθώντας τη λογική της ανάπτυξης όλων των επεκτάσεων του φυλλομετρητή Mozilla Firefox. Η πρώτη από αυτές χρησιμοποιήθηκε για τη δημιουργία των στοιχείων της διεπαφής χρήστη (παράθυρα, διάλογοι, μενού, κουμπιά κ.τ.λ.) και η δεύτερη για να προσδώσει λειτουργικότητα σε αυτά τα στοιχεία. Χρησιμοποιήθηκε ακόμη ένα σύνολο τεχνολογιών, οι

οποίες έχουν εισαχθεί από το Mozilla και συμπληρώνουν την XUL στην κατασκευή εφαρμογών ανεξάρτητων της πλατφόρμας:

- **Overlays**

τα Overlays είναι αρχεία XUL τα οποία χρησιμοποιούνται για να περιγράψουν επιπλέον περιεχόμενο για τη διεπαφή χρήστη. Με τη βοήθεια των Overlays μπορεί να γίνει προσαρμογή και επέκταση ήδη υπαρχόντων εφαρμογών, χωρίς να χρειάζεται να παρέχεται εκ νέου όλο το UI, αλλά απλά υπερθέτοντας τα νέα στοιχεία και ενδεχομένως παρακάμπτοντας κάποια τμήματα του αρχικού UI. Τα Overlays μπορούν ακόμη να λειτουργήσουν με τρόπο αντίστοιχο με τα αρχεία “include” άλλων γλωσσών προγραμματισμού, καθώς μια εφαρμογή μπορεί να καθορίσει να συμπεριληφθεί κάποιο Overlay στον ορισμό της.

- **XPCOM/XPConnect**

Τα XPCOM και XPConnect είναι συμπληρωματικές τεχνολογίες που επιτρέπουν την ενοποίηση εξωτερικών βιβλιοθηκών με εφαρμογές XUL. Όπως έχει ήδη αναφερθεί, το XPCOM (Cross Platform Component Object Model) είναι ένα framework για τη συγγραφή αρθρωτού λογισμικού ανεξάρτητου από πλατφόρμες, και το οποίο επιτρέπει τη δημιουργία αντικειμένων τα οποία επικοινωνούν με άλλα αντικείμενα. Τα XPCOM στοιχεία μπορεί να είναι γραμμένα σε C, C++ ή JavaScript και μπορούν να χρησιμοποιούνται από C, C++, JavaScript, Python, Java και Perl. Το XPConnect είναι μια τεχνολογία που επιτρέπει τη διαλειτουργικότητα μεταξύ των XPCOM και της JavaScript. Συγκεκριμένα, επιτρέπει σε αντικείμενα της JavaScript να έχουν πρόσβαση και να διαχειρίζονται διαφανώς τα αντικείμενα XPCOM. Στο σύστημα UranuS δεν κατασκευάστηκαν νέα αντικείμενα XPCOM, αλλά έγινε χρήση ήδη έτοιμων αντικειμένων XPCOM του Mozilla.

- **XPInstall**

Το XPInstall, (Cross Platform Install) είναι η τεχνολογία εγκατάστασης του Mozilla η οποία είναι ανεξάρτητη της πλατφόρμας. Παρέχει έναν καθορισμένο τρόπο για τη συσκευασία των στοιχείων εφαρμογής XUL, μαζί με ένα script εγκατάστασης το οποίο μπορεί να κατεβάσει και να εκτελέσει ο Mozilla. Έτσι, επιτρέπει στους χρήστες να εγκαθιστούν χωρίς κόπο νέες εφαρμογές XUL από το διαδίκτυο ή από εταιρικούς εξυπηρετητές intranet. Για την εγκατάσταση μιας νέας εφαρμογής, ο χρήστης χρειάζεται μόνο να πατήσει με το ποντίκι του σε έναν σύνδεσμο υπερκειμένου σε κάποια ιστοσελίδα ή σε ένα μήνυμα ηλεκτρονικού ταχυδρομείου και να αποδεχθεί το νέο πακέτο συσκευασίας μέσω από έναν διάλογο εγκατάστασης του Mozilla.

Όπως είναι αυτονόητο, η επιλογή της XUL για την κατασκευή της διεπαφής χρήστη ήταν αναγκαία καθώς το σύστημα UranuS αποτελεί επέκταση του Mozilla Firefox και επομένως έπρεπε να χτιστεί πάνω στην ίδια πλατφόρμα. Αξίζει να αναφερθούν ωστόσο οι λόγοι για τους οποίους αναπτύχθηκε η XUL και για τους οποίους βασίστηκε πάνω της η κατασκευή της πλατφόρμας του Mozilla:

1. *Αποτελεί μια ισχυρή markup γλώσσα, βασισμένη στα στοιχεία εφαρμογής (widgets)*

Στόχος της XUL είναι η κατασκευή εφαρμογών ανεξάρτητων της πλατφόρμας, σε αντίθεση για παράδειγμα με τη DHTML που σα στόχο έχει την ανάπτυξη ιστοσελίδων. Για το λόγο αυτό, η XUL είναι προσανατολισμένη στα widgets, όπως τα παράθυρα, οι επιγραφές και τα κουμπιά, αντί για τις σελίδες, τα επίπεδα επικεφαλίδων και τους υπερσυνδέσμους. Αρκετοί προγραμματιστές καταβάλλουν υπερπροσπάθεια για να πετύχουν αντίστοιχα αποτελέσματα στις DHTML web εφαρμογές τους, έχοντας όμως επιπτώσεις στην πολυπλοκότητα και την απόδοση και χωρίς να έχουν υποστήριξη από πρότυπα (standards).

2. *Είναι βασισμένη σε ήδη υπάρχοντα πρότυπα*

Η XUL είναι μια XML γλώσσα, βασισμένη στο πρότυπο XML 1.0 του W3C. Οι εφαρμογές που γράφονται σε αυτή βασίζονται επιπλέον σε άλλες προτυποποιημένες από τον W3C τεχνολογίες όπως την HTML 4.0, τα CSS 1 και 2, τα επίπεδα DOM 1 και 2, τη JavaScript 1.5, συμπεριλαμβανομένης της ECMA-262 Edition 3 (ECMAscript) και την XML 1.0.

3. *Φορητότητα Πλατφόρμας*

Όπως η HTML, η XUL είναι σχεδιασμένη για να είναι ουδέτερη στις πλατφόρμες, κάνοντας έτσι εύκολη τη μεταφορά των εφαρμογών σε όλα τα λειτουργικά συστήματα στα οποία τρέχει ο Mozilla. Αν αναλογιστεί κανείς το μεγάλο εύρος πλατφόρμων που υποστηρίζουν το Mozilla αυτή τη στιγμή, αυτό είναι ένα από τα πιο ελκυστικά χαρακτηριστικά της XUL ως τεχνολογίας κατασκευής εφαρμογών. Καθώς η XUL προσφέρει μια αφαίρεση των στοιχείων διεπαφής χρήστη, εξυπηρετεί τη λογική “write-once, run-anywhere” («το γράφεις μια φορά, το τρέχεις οπουδήποτε»).

4. *Διαχωρισμός της παρουσίασης από την λογική της εφαρμογής*

Ένα από τα μεγαλύτερα μειονεκτήματα των περισσότερων διαδικτυακών εφαρμογών είναι η ισχυρή σύζευξη των στοιχείων της διεπαφής χρήστη με την λογική της εφαρμογής πελάτη, καθώς συνήθως τα δύο αυτά κομμάτια αναπτύσσονται από διαφορετικά άτομα. Η XUL παρέχει έναν καθαρό διαχωρισμό μεταξύ του ορισμού και της προγραμματιστικής λογικής της εφαρμογής πελάτη (του “content”, που αποτελείται από την XUL, την XBL και τη JavaScript), της παρουσίασης (του “skin”, που αποτελείται από τα CSS και από εικόνες) και των επιγραφών σε συγκεκριμένες γλώσσες (του “locale”, που αποτελείται από .dtd και .properties αρχεία). Έτσι, η δομή και εμφάνιση των εφαρμογών XUL μπορούν να τροποποιηθούν ανεξάρτητα από τον ορισμό και τη λογική της εφαρμογής. Επιπλέον, η εφαρμογή μπορεί να μεταφερθεί σε διαφορετικές γλώσσες, ανεξάρτητα από τη λογική ή τον τρόπο παρουσίασής της. Αυτός ο βαθμός διαχωρισμού οδηγεί σε εφαρμογές που είναι ευκολότερο να συντηρηθούν από προγραμματιστές και να προσαρμοστούν στις διάφορες απαιτήσεις από τους σχεδιαστές και τους

μεταφραστές γλώσσας. Η ροή εργασίας σε αυτές τις ανεξάρτητες εφαρμογές συντονίζεται πολύ ευκολότερα απ' ότι στις εφαρμογές που βασίζονται στην HTML, και με μικρότερη επίδραση στη συνολική σταθερότητα και την ποιότητα του συστήματος.

5. *Εύκολη προσαρμογή στις απαιτήσεις, στον εν-τοπισμό ή τη σήμανση*

Ένα άλλο ιδιαίτερα πρακτικό πλεονέκτημα του διαχωρισμού που προσφέρει η XUL μεταξύ της λογικής της εφαρμογής, της παρουσίασης και του γλωσσικού κειμένου είναι η ευκολία προσαρμογής της εφαρμογής για διαφορετικούς πελάτες ή ομάδες χρηστών. Ο προγραμματιστής μπορεί να διατηρεί τον βασικό κώδικα της εφαρμογής και να προσαρμόζει το λογότυπο και τη σήμανση για κάθε πελάτη του, παρέχοντας διαφορετικά skins. Ακόμη, μια εφαρμογή που είναι γραμμένη και αναπτυγμένη με μια διεπαφή χρήστη στην Αγγλική γλώσσα μπορεί να μεταφραστεί σε άλλη γλώσσα για τον ίδιο πελάτη. Ενώ τέτοιου είδους αλλαγές είναι εκτεταμένες και επηρεάζουν σε μεγάλο βαθμό την εφαρμογή, ταυτόχρονα είναι απομονωμένες η μία από την άλλη, επιτρέποντας στον πυρήνα του ορισμού και της λογικής της εφαρμογής να μοιράζεται μεταξύ των διαφόρων εκδόσεων.

6.2.2 Απαιτήσεις Συστήματος

Το σύστημα UranuS μπορεί να εγκατασταθεί σε υπολογιστή με λειτουργικό σύστημα Windows (Windows 98, XP ή Vista), όπου είναι εγκατεστημένος ο φυλλομετρητής Mozilla Firefox. Μπορεί ωστόσο να εγκατασταθεί και να λειτουργήσει και σε περιβάλλον Linux. Η ελάχιστη έκδοση του Firefox που απαιτείται είναι η 1.5, ενώ υποστηρίζονται όλες οι εκδόσεις της σειράς 2.0.0.* . Η υποστήριξη της έκδοσης 3, η οποία βρίσκεται ακόμη σε δοκιμαστική κυκλοφορία, δεν έχει ελεγχθεί.

6.2.3 Διαδικασία Εγκατάστασης της Εφαρμογής

Για την εγκατάσταση της εφαρμογής, ο ενδιαφερόμενος πρέπει να ανοίξει τον Mozilla Firefox, να μεταβεί στην ιστοσελίδα http://users.ntua.gr/el02226/UranuS_extension.html και να πατήσει με το ποντίκι του στον υπερσύνδεσμο που εμφανίζεται. Στη συνέχεια, θα παρουσιαστεί ένα παράθυρο διαλόγου που θα του ζητά να επιβεβαιώσει ότι επιθυμεί την εγκατάσταση του λογισμικού. Μετά την αποδοχή από το χρήστη, εμφανίζεται το παράθυρο για τα *Πρόσθετα* του Firefox όπου υπάρχει μια μπάρα προόδου που δείχνει την πορεία της εγκατάστασης. Αφού γεμίσει η μπάρα προόδου, επισημαίνεται στο χρήστη ότι για να ολοκληρωθεί η εγκατάσταση πρέπει να γίνει επανεκκίνηση του Firefox. Αν ο χρήστης πατήσει το ειδικό κουμπί που υπάρχει στο παράθυρο για την επανεκκίνηση του περιηγητή, τότε η επανεκκίνηση γίνεται αυτόματα και φορτώνονται ξανά όλες οι ιστοσελίδες που είχε προηγουμένως ανοιχτέ, ενώ στο κάτω μέρος του παραθύρου του Firefox εμφανίζονται οι ενδείξεις του συστήματος UranuS. Από τη

στιγμή αυτή, στο παράθυρο "*Πρόσθετα*" ("*Add-ons*") του Firefox, το οποίο μπορεί να βρει κανείς επιλέγοντας "*Εργαλεία*"→"*Πρόσθετα*" υπάρχει, μεταξύ των άλλων, μία καταχώρηση για το σύστημα UranuS. Ο χρήστης μπορεί πλέον να χρησιμοποιήσει κανονικά την εφαρμογή, ορίζοντας νέα προφίλ, αλλάζοντας το προφίλ χρήσης και τροποποιώντας τα ήδη υπάρχοντα προφίλ. Σημειώνεται ότι αν ο χρήστης κατά την εγκατάσταση επιλέξει να μην επανεκκινήσει αμέσως τον υπολογιστή του από την αντίστοιχη επιλογή, αλλά κλείσει κανονικά το παράθυρο "*Πρόσθετα*", η εγκατάσταση θα ολοκληρωθεί μόνο όταν τερματιστεί η λειτουργία του Firefox και εκκινηθεί ξανά.

7 Έλεγχος

Στο κεφάλαιο αυτό γίνεται η αξιολόγηση του συστήματος UranuS.

7.1 Μεθοδολογία ελέγχου

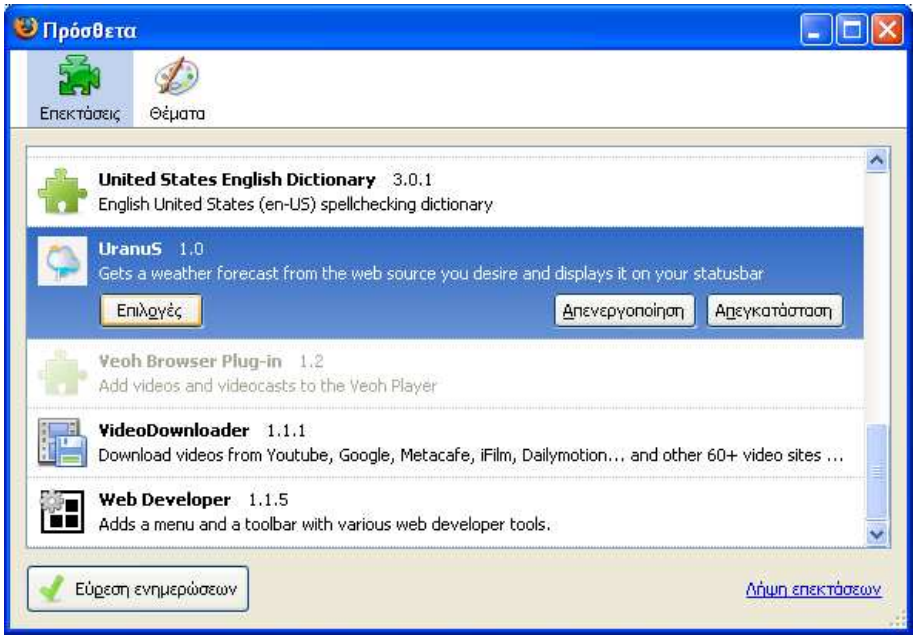
Ο έλεγχος πραγματοποιήθηκε με τη χρήση ενός σεναρίου λειτουργίας για τη δημιουργία ενός νέου προφίλ από το χρήστη και τη χρήση του από το σύστημα. Το σενάριο έχει ως εξής:

1. Ο χρήστης ανοίγει τον περιηγητή Moxilla Firefox και επιλέγει από το μενού των πρόσθετων το βασικό μενού της επέκτασης UranuS.
2. Από το βασικό μενού επιλέγει τη δημιουργία νέου προφίλ, οπότε και ανοίγει ο βοηθός (wizard) που αναφέρθηκε στο κεφάλαιο της Ανάλυσης Απαιτήσεων του συστήματος.
3. Ο χρήστης ακολουθεί τα βήματα που περιγράφηκαν στην Ανάλυση Απαιτήσεων καθοδηγούμενος από τον wizard και ολοκληρώνει τη δημιουργία του προφίλ, ορίζοντας τη χρησιμοποίηση του νέου προφίλ ως τρέχοντος.
4. Αμέσως μετά, η εφαρμογή λαμβάνει και εμφανίζει στη γραμμή κατάστασης τα μετεωρολογικά δεδομένα, ενώ ο χρήστης μπορεί να τα δει σε μεγαλύτερη λεπτομέρεια ανοίγοντας το παράθυρο πρόβλεψης.

7.2 *Αναλυτική παρουσίαση ελέγχου*

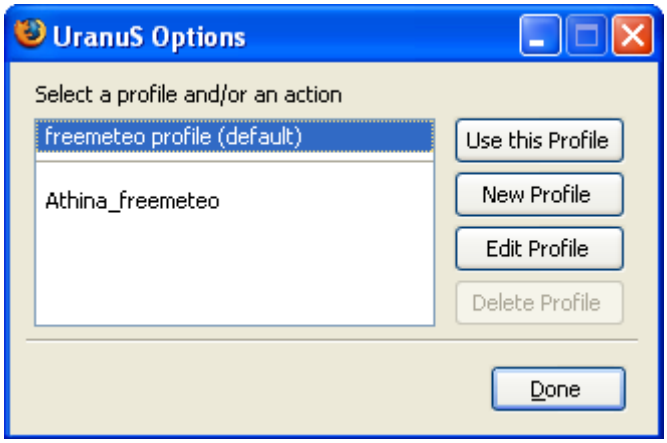
Στην ενότητα αυτή παρουσιάζεται αναλυτικά ο έλεγχος του συστήματος UranuS σύμφωνα με το σενάριο που περιγράφηκε στην προηγούμενη ενότητα.

1. Ο χρήστης ανοίγει τον περιηγητή του και επιλέγει από τη γραμμή μενού "*Εργαλεία→Πρόσθετα*". Από το παράθυρο που εμφανίζεται, επιλέγει την εντολή "*Επιλογές*" της επέκτασης UranuS.



Εικόνα 7.2-1 Το παράθυρο "*Πρόσθετα*" του Mozilla Firefox

2. Από το βασικό μενού που παρουσιάζεται, ο χρήστης επιλέγει "*New Profile*".

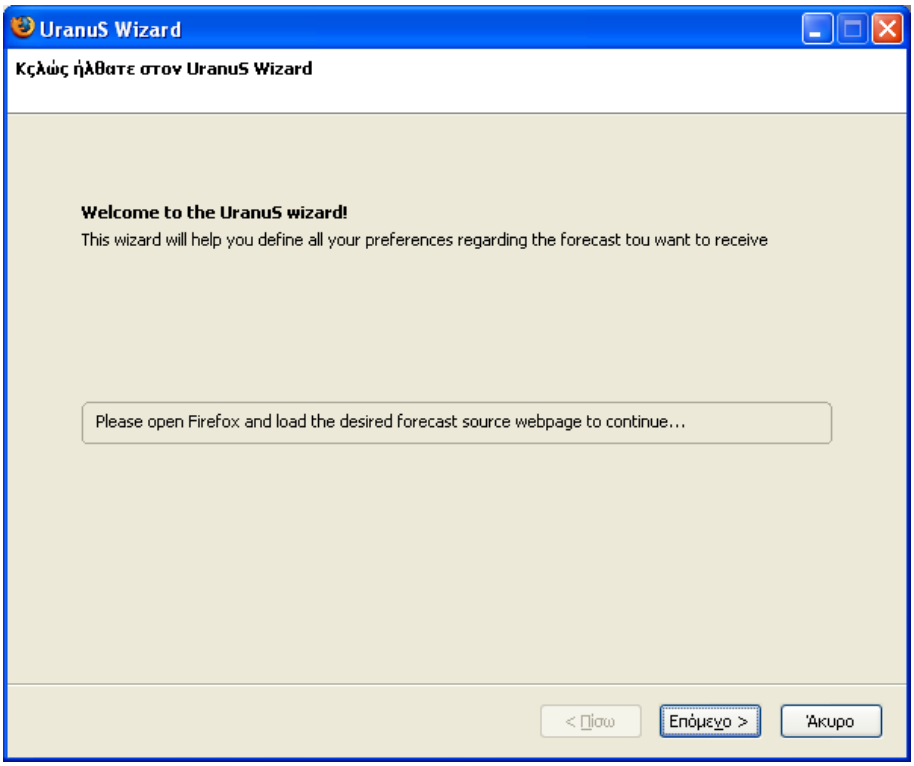


Εικόνα 7.2-2 Το βασικό μενού επιλογών της επέκτασης UranuS

3. Ανοίγει ο βοηθός δημιουργίας νέου προφίλ χρήστη. Ακολουθούν τα εξής βήματα:
    a. Ο χρήστης καλωσορίζεται στο βοηθό. Αναζητά και βρίσκει στον περιηγητή του την ιστοσελίδα

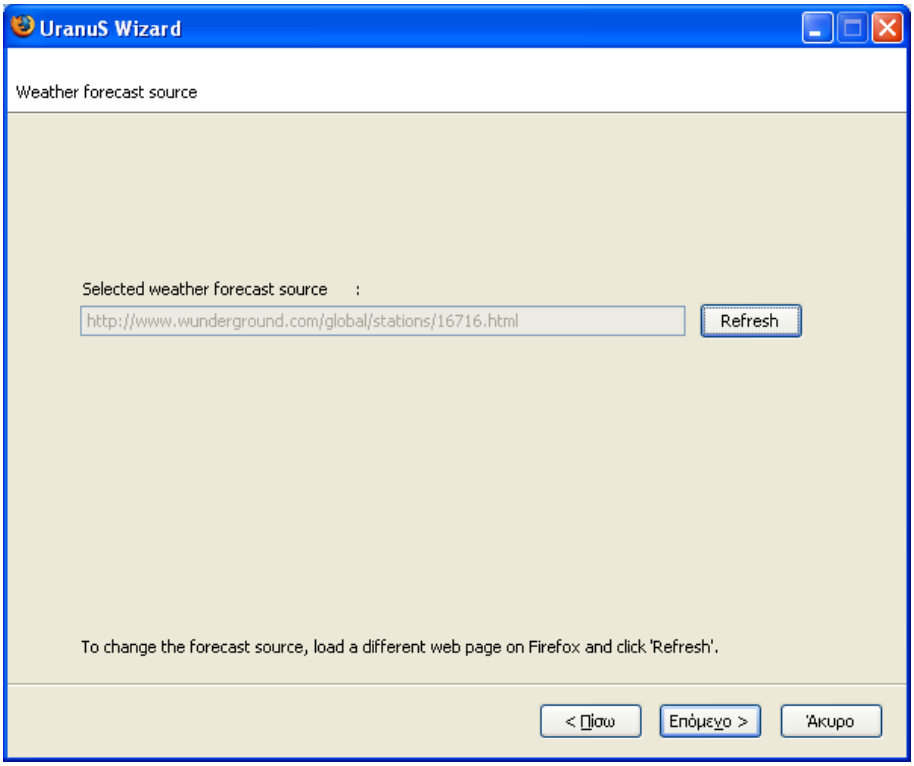
        http://www.wunderground.com/global/stations/16716.html

    η οποία περιέχει μετεωρολογικές προγνώσεις για την πόλη της Αθήνας και προχωρά στο επόμενο βήμα.



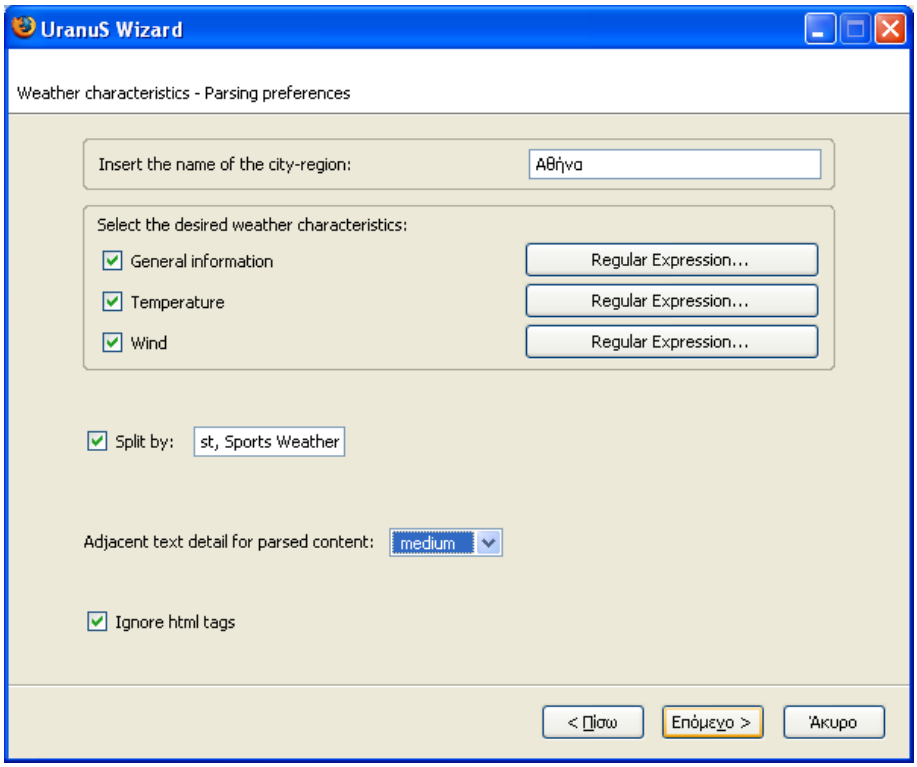
Εικόνα 7.2-3 Αρχική σελίδα του wizard

b. Το URL της ιστοσελίδας-πηγής φορτώνεται στο ειδικό πλαίσιο στα παράθυρο του wizard.

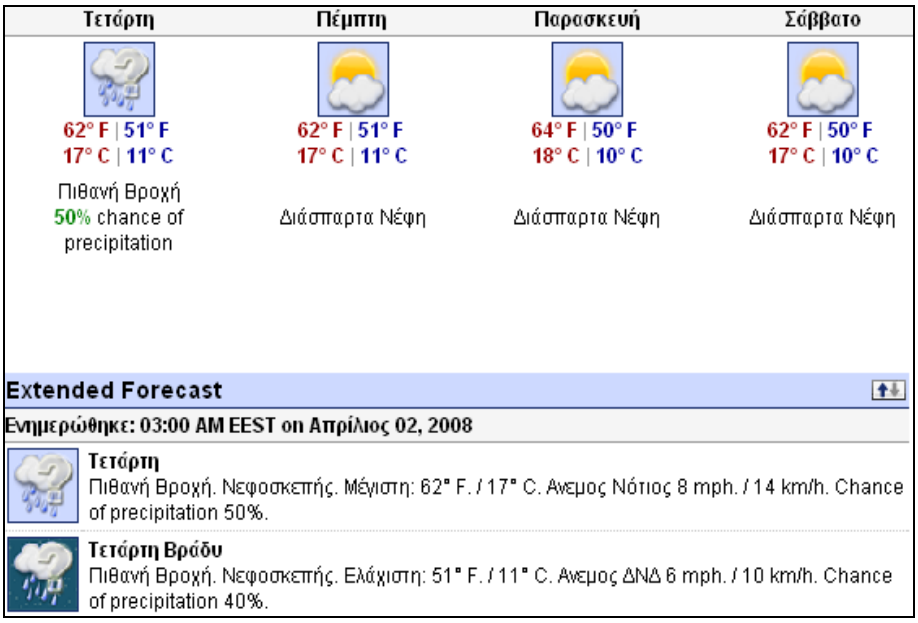


Εικόνα 7.2-4 Σελίδα καθορισμού της ιστοσελίδα-πηγής

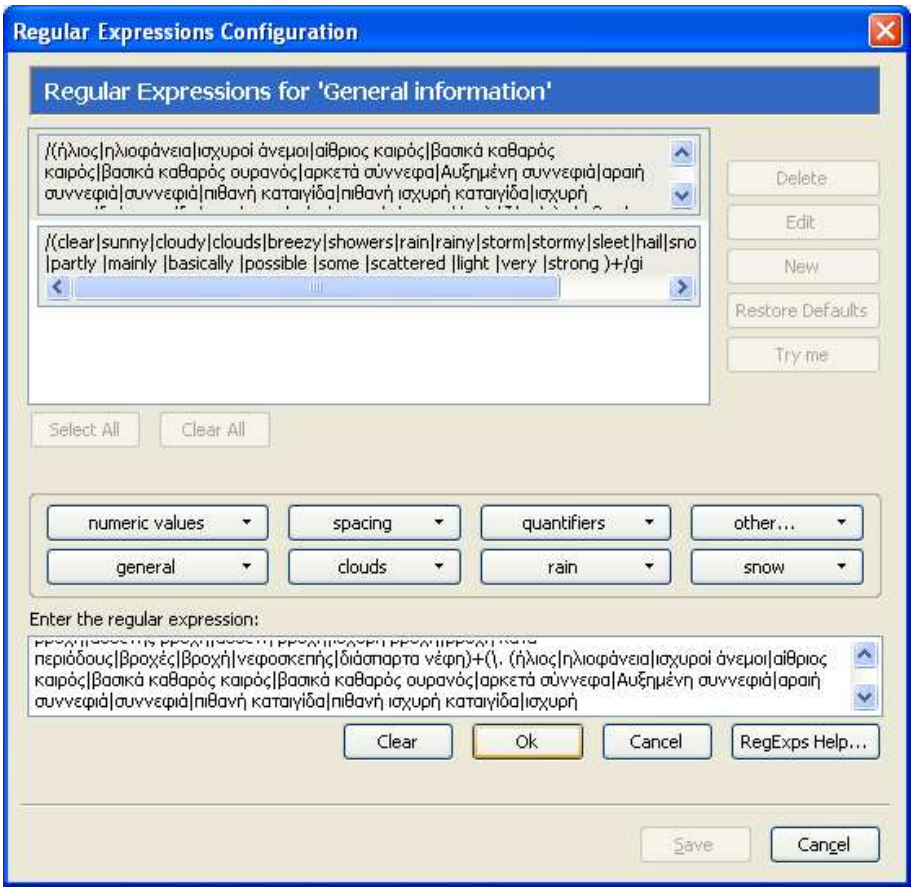
c. Στη νέα οθόνη, ο χρήστης συμπληρώνει το πεδίο της περιοχής και τσεκάρει τις επιλογές "*General information*", "*Temperature*" και "*Wind*", αφού θέλει να ενημερώνεται για τόσο για τη γενική περιγραφή του καιρού, όσο και για τη θερμοκρασία και τον καιρό. Παρατηρώντας ότι οι πληροφορίες για τον άνεμο στην ιστοσελίδα δίνονται μόνο στην ενότητα "Extended Forecast", η οποία ακολουθείται αμέσως από την ενότητα "Sports Weather" επιλέγει να γίνει ο διαχωρισμός της ιστοσελίδας με βάση τις εκφράσεις αυτές, τσεκάροντας την επιλογή "*split-by*" και συμπληρώνοντας το αντίστοιχο πεδίο. Επιλέγει ακόμη να αγνοηθούν τα HTML tags της ιστοσελίδας κατά τη λεκτική και συντακτική ανάλυση, και τον μεσαίο βαθμό λεπτομέρειας για την παρουσίαση του γειτονικού κειμένου στο επόμενο βήμα. Τέλος, ελέγχει τις κανονικές εκφράσεις για κάθε χαρακτηριστικό καιρού και τις διαμορφώνει έτσι ώστε να ταιριάζουν στα χαρακτηριστικά της σελίδας (για παράδειγμα, παρατηρεί ότι χρησιμοποιείται η ασυνήθιστη φράση «Νεφοσκεπής» όπως και η φράση «Διάσπαρτα νέφη» για την περιγραφή του καιρού, τις οποίες και συμπληρώνει στην αντίστοιχη κανονική έκφραση).



Εικόνα 7.2-5 Σελίδα καθορισμού των χαρακτηριστικών καιρού και των προτιμήσεων λεκτικής και συντακτικής ανάλυσης.

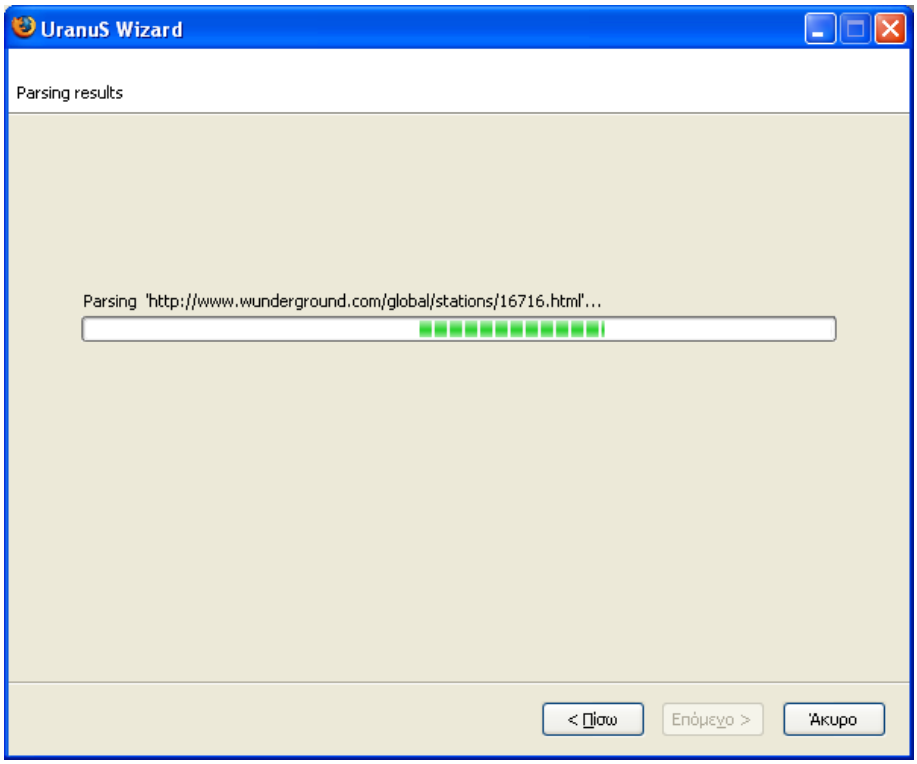


Εικόνα 7.2-6 Απόσπασμα από την ιστοσελίδα – πηγή της πρόγνωσης στον ιστότοπο wunderground.com

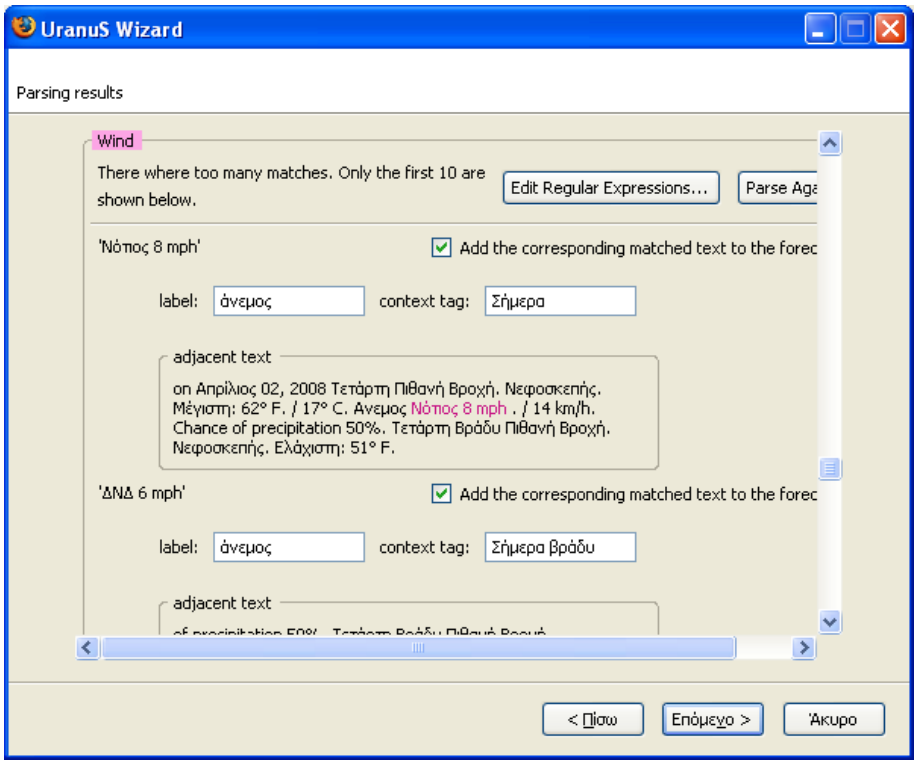


Εικόνα 7.2-7 Παράθυρο ορισμού των κανονικών εκφράσεων

d. Στη συνέχεια γίνεται parsing της ιστοσελίδας και παρουσιάζονται τα αποτελέσματα στο χρήστη, κατά τον τρόπο που έχει περιγραφεί στην Ανάλυση Απαιτήσεων. Ο χρήστης εντοπίζει το δεύτερο τμήμα κειμένου (*Fragment 2*) και από εκείνο τις πληροφορίες που αντιστοιχούν στις πληροφορίες που τον ενδιαφέρουν, με βάση τη σειρά παρουσίασης των αποτελεσμάτων και το γειτονικό τους κείμενο. Συγκεκριμένα, αν και στο συγκεκριμένο τμήμα της σελίδας περιλαμβάνονται οι προγνώσεις για 6 συνεχείς ημέρες τόσο για τη διάρκεια της μέρας όσο και για τη διάρκεια της νύχτας, ο χρήστης ενδιαφέρεται μόνο για τις 3 πρώτες ημέρες. Έτσι, συμπληρώνει τα πεδία label και context tag με τις κατάλληλες τιμές για όλες τις σχετικές πληροφορίες και προχωρά στο επόμενο βήμα.

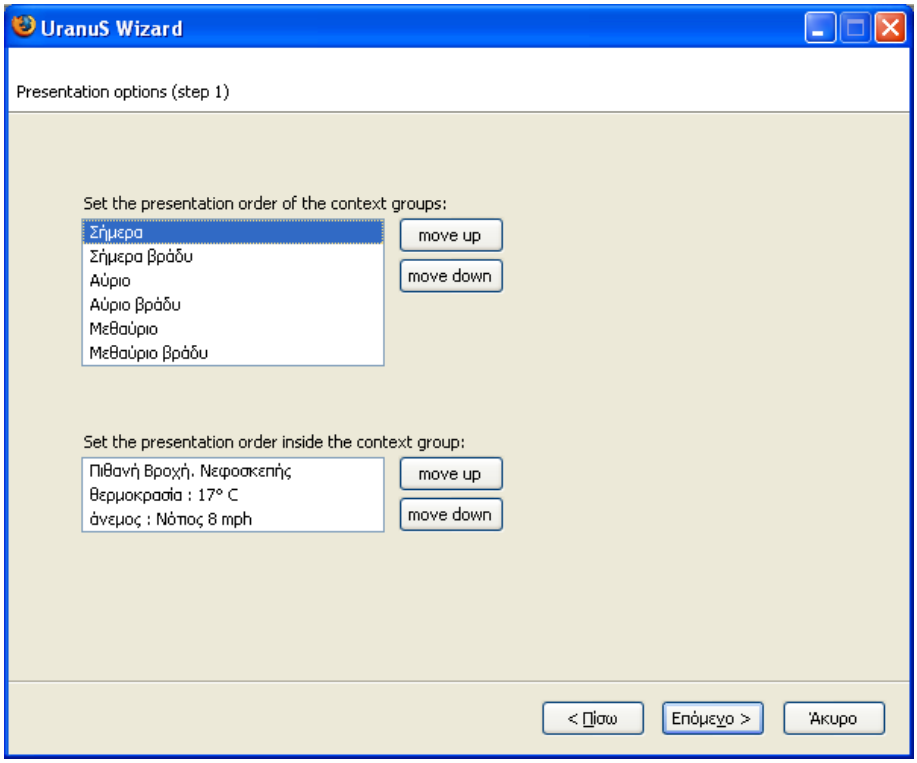


Εικόνα 7.2-8 Parsing της ιστοσελίδας - πηγής



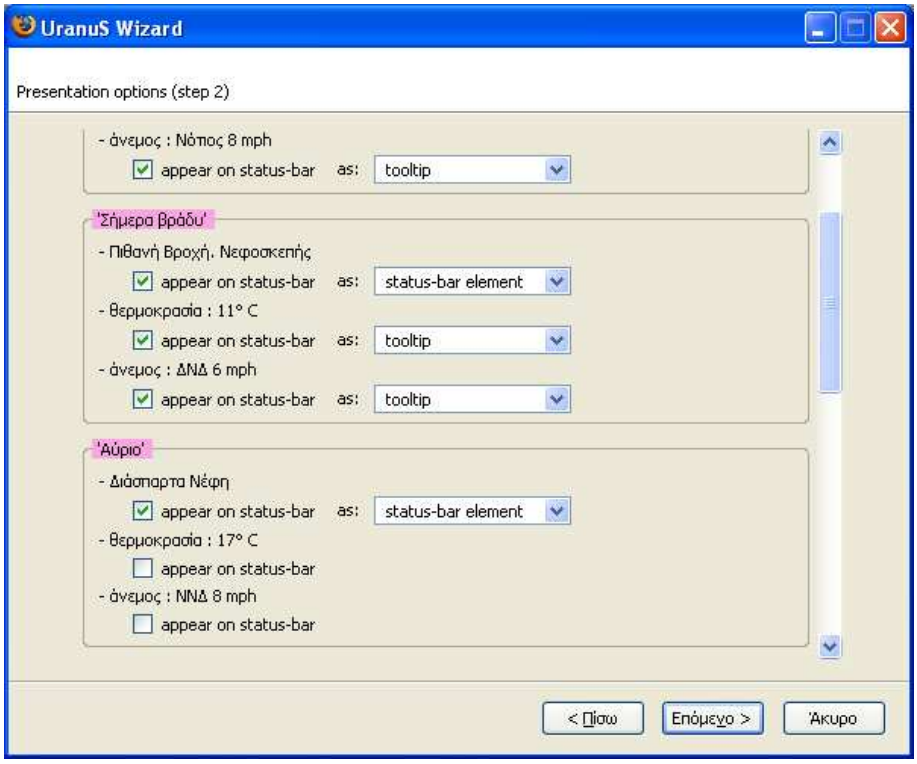
Εικόνα 7.2-9 Παρουσίαση των αποτελεσμάτων στον χρήστη και επιλογή των επιθυμητών πληροφοριών

e. Στην επόμενη οθόνη παρουσιάζονται οι ομάδες πληροφοριών που δημιουργήθηκαν. Από τον τρόπο που ορίστηκαν οι πληροφορίες δεν απαιτείται καμία αλλαγή στη σειρά παρουσίασης των ομάδων, ούτε στη σειρά των στοιχείων σε καθεμία από αυτές.



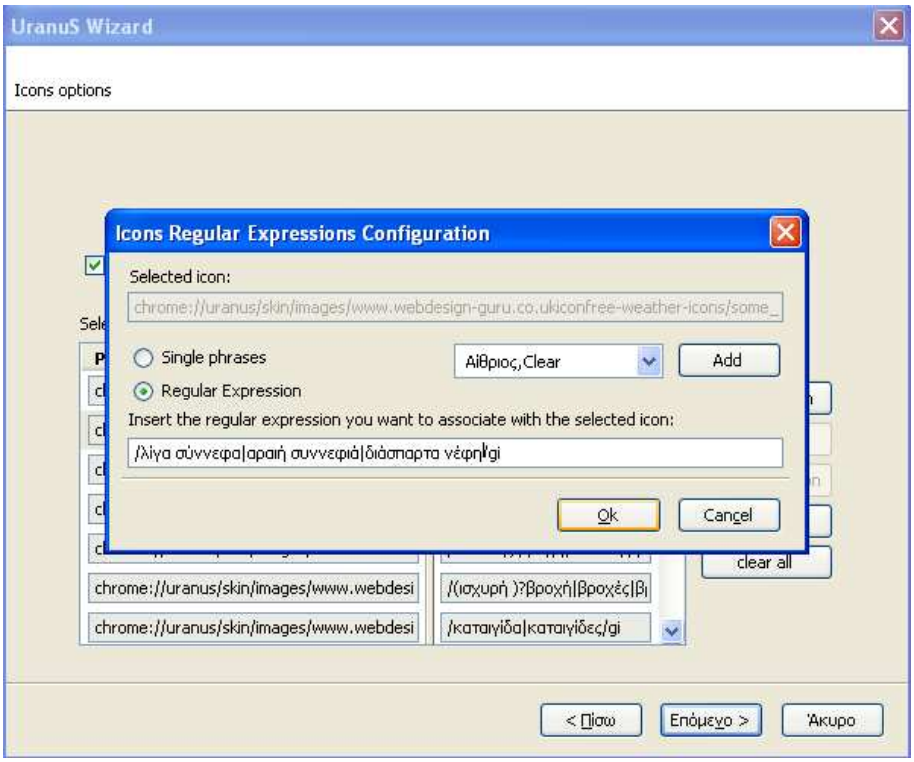
Εικόνα 7.2-10 Σελίδα καθορισμού της σειράς εμφάνισης των πληροφοριών

f. Ακολούθως, εμφανίζονται όλες οι πληροφορίες με τη σωστή σειρά και ο χρήστης ορίζει πώς θέλει να εμφανίζονται. Συγκεκριμένα, επιλέγει να εμφανίσει τις γενικές πληροφορίες του καιρού για τις ομάδες «Σήμερα», «Σήμερα βράδυ», «Αύριο» και «Αύριο βράδυ» στη γραμμή κατάστασης, τις πιο αναλυτικές πληροφορίες των ομάδων αυτών στο κυλιόμενο κείμενο της γραμμής κατάστασης (μαζί και με τις γενικές πληροφορίες φυσικά) και τέλος όλες τις πληροφορίες για τις ομάδες «Μεθαύριο» και «Μεθαύριο βράδυ» μόνο στο παράθυρο πρόγνωσης, μαζί και με όλες τις προηγούμενες πληροφορίες.

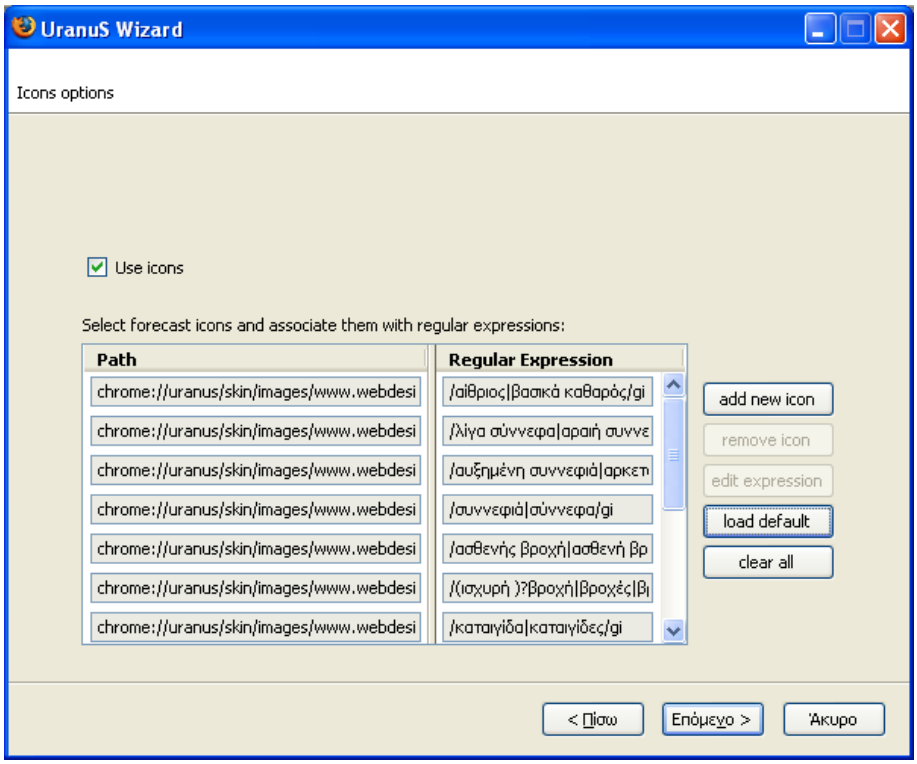


Εικόνα 7.2-11 Σελίδα καθορισμού τρόπου παρουσίασης των πληροφοριών

g. Στο επόμενο βήμα ο χρήστης ορίζει ότι θέλει να χρησιμοποιούνται εικονίδια στις προγνώσεις του. Για το σκοπό αυτό φορτώνει τα προκαθορισμένα εικονίδια του συστήματος και τροποποιεί τις κανονικές εκφράσεις για τα εικονίδια που αντιστοιχούν στην αραιή συννεφιά και την πυκνή συννεφιά ώστε να περιλαμβάνουν τις εκφράσεις «διάσπαρτα νέφη» και «νεφοσκεπής» που παρατήρησε ότι χρησιμοποιούνται από την ιστοσελίδα.

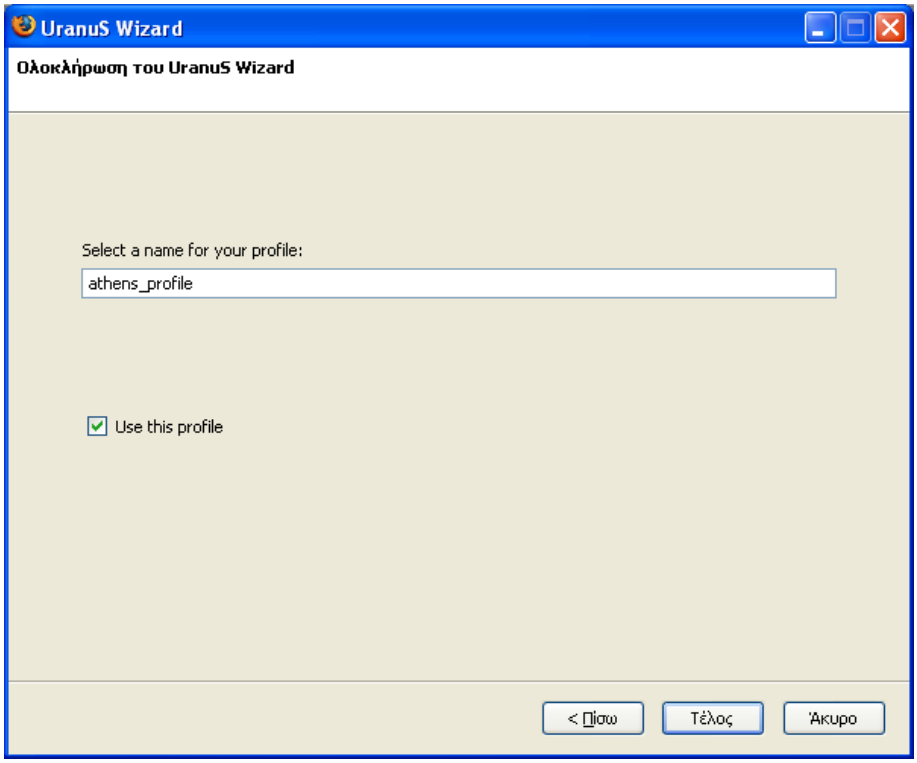


Εικόνα 7.2-12



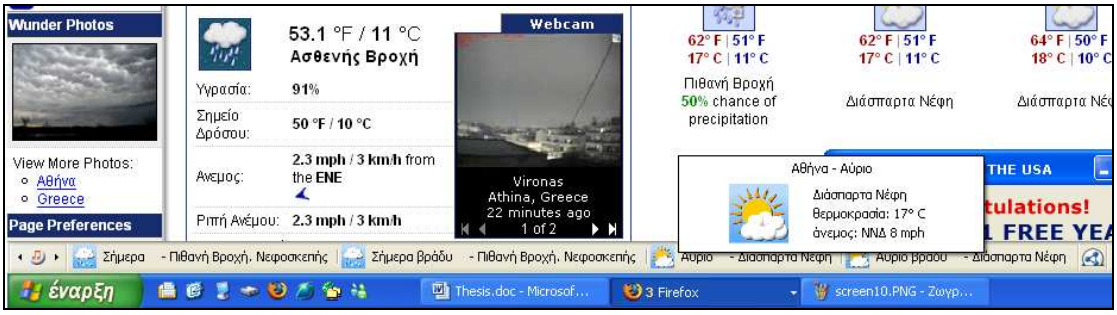
Εικόνα 7.2-13 Σελίδα ορισμού των χρησιμοποιούμενων εικονιδίων

h. Τέλος, ο χρήστης ορίζει το όνομα του προφίλ και επιλέγει να χρησιμοποιείται αυτό το προφίλ από εδώ και στο εξής.



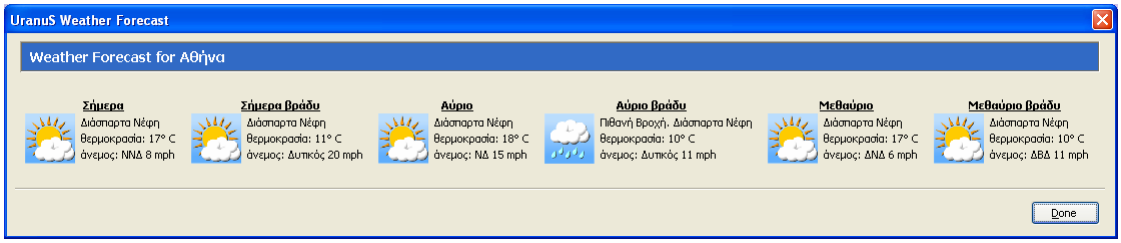
Εικόνα 7.2-14 Ολοκλήρωση του wizard

4. Αμέσως, στη γραμμή κατάστασης, στη θέση εμφάνισης της επέκτασης UranuS, εμφανίζεται μια γραμμή προόδου, η οποία σύντομα αντικαθίσταται από τις ληφθείσες πληροφορίες που αντιστοιχούν στο νέο προφίλ.



Εικόνα 7.2-15 Εμφάνιση της πρόγνωσης στη γραμμή κατάστασης του Mozilla Firefox

5. Ο χρήστης, επιθυμώντας να δει όλες τις πληροφορίες της πρόγνωσης ανοίγει με δεξί κλικ πάνω στην πρόγνωση το μενού της γραμμής κατάστασης για την επέκταση UranuS, από το οποίο επιλέγει “See weather forecast”. Έτσι, ανοίγει το παράθυρο πρόγνωσης με συγκεντρωμένα όλα τα επιθυμητά δεδομένα.



Εικόνα 7.2-16 Παράθυρο πρόγνωσης

8 *Επίλογος*

Στο κεφάλαιο αυτό συνοψίζεται η παρουσίαση της διπλωματικής εργασίας.

8.1 Σύνοψη και συμπεράσματα

Το σύστημα που υλοποιήθηκε στην παρούσα διπλωματική παρέχει μια χρήσιμη υπηρεσία στο χρήστη, αφού του προσφέρει ενημέρωση με έναν άμεσο τρόπο, προσαρμοζόμενο στις δικές του απαιτήσεις και ταυτόχρονα απαλλάσσοντάς τον από καθημερινά επαναλαμβανόμενες ενέργειες. Έτσι ο χρήστης αποκτά απόλυτη ελευθερία κινήσεων.

Βασικό πλεονέκτημα της εφαρμογής είναι ότι εξαγωγή των δεδομένων μπορεί θεωρητικά να γίνει από οποιονδήποτε τύπο σελίδας, όπως από δομημένες σελίδες που χρησιμοποιούν πίνακες για την παρουσίαση των δεδομένων ή από σελίδες με προγνώσεις γραμμένες σε φυσική γλώσσα, ενώ δεν επηρεάζεται και από τυχόν αλλαγές στη δομή μιας σελίδας. Ο κώδικας δηλαδή της εφαρμογής παραμένει ο ίδιος ακόμη και όταν ο χρήστης προσθέτει μια νέα πηγή πληροφορίας ή όταν μια ήδη υπάρχουσα πηγή αλλάζει τον τρόπο παρουσίασης των δεδομένων της. Το μόνο που τροποποιείται είναι οι προτιμήσεις του χρήστη, η οποίες χρησιμοποιούνται ως παράμετροι από το πρόγραμμα για την εξαγωγή της επιθυμητής πληροφορίας.

Ένα άλλο σημαντικό πλεονέκτημα είναι ότι το σύστημα, έχοντας τη δυνατότητα να προσαρμοστεί στις απαιτήσεις του χρήστη, μπορεί να προσφέρει αρκετά διαφορετικούς τρόπους ενημέρωσης και άρα να ικανοποιήσει διαφορετικούς χρήστες, ακόμη και με ιδιαίτερα απαιτητικές προτιμήσεις Για παράδειγμα, ένας χρήστης μπορεί να επιλέξει να βλέπει τις τρέχουσες καιρικές συνθήκες στις σημαντικότερες πόλεις της Ελλάδας, άλλος να λαμβάνει την πρόγνωση των 3 επόμενων ημερών για την πόλη του και άλλος να ενημερώνεται για τη θερμοκρασία και το ύψος της χιονόπτωσης σε διαφορετικά χιονοδρομικά κέντρα της Ελλάδας. Από την άλλη, το σύστημα μπορεί να χρησιμοποιηθεί εύκολα και από μη έμπειρους χρήστες ή

από χρήστες που επιζητούν μια απλή και γρήγορη λύση, καθώς μέσω του προκαθορισμένου προφίλ που προσφέρει μπορεί να λειτουργήσει και ως μια οποιοδήποτε απλή επέκταση για μετεωρολογικές προγνώσεις.

Ένα αδύνατο σημείο ωστόσο της εφαρμογής είναι ότι, εξαιτίας της τεράστιας ανομοιογένειας στον τρόπο παρουσίασης των μετεωρολογικών δεδομένων από τις διάφορες πηγές, η εξαγωγή των επιθυμητών πληροφοριών βασίζεται στη σειρά με την οποία συναντά αντίστοιχα δεδομένα (σειρά ταιριάσματος της κανονικής έκφρασης) και άρα απαιτεί ένα σχετικά σταθερό μοντέλο σελίδας. Έτσι, υπάρχουν περιπτώσεις που μπορεί να αστοχήσει, αν το περιεχόμενο της ιστοσελίδας μεταβάλλεται με ανομοιόμορφο τρόπο, όπως για παράδειγμα στην περίπτωση που μία σελίδα που παρέχει προβλέψεις για την τρέχουσα ημέρα τόσο για τη διάρκεια της μέρας όσο και για τη διάρκεια της νύχτας, απομακρύνει την πρόγνωση της μέρας μετά το πέρας κάποιας απογευματινής ώρας. Παρόλα αυτά, ο τρόπος με τον οποίο έχει σχεδιαστεί το σύστημα παρέχει τέτοια ευλυγισία, ώστε ο χρήστης να μπορεί με κατάλληλες προσαρμογές να επιλύσει τα περισσότερα προβλήματα που μπορεί να συναντήσει.

Με βάση τα παραπάνω επιβεβαιώνεται λοιπόν η συνεισφορά της διπλωματικής στα προβλήματα που αναφέρθηκαν στην εισαγωγή.

8.2 Μελλοντικές επεκτάσεις

Κάποιες πιθανές επεκτάσεις της διπλωματικής στη μορφή που έχει τώρα είναι:

- Εισαγωγή επιλογής στο χρήστη σε σχέση με το σύστημα μέτρησης που θα χρησιμοποιηθεί στην παρουσίαση των δεδομένων (μετατροπή μεταξύ συστημάτων κ.τ.λ.),
- Εισαγωγή επιπλέον επιλογών όπως ρύθμιση του χρόνου ανανέωσης των δεδομένων ή του συνδυασμού δεδομένων από διαφορετικές πηγές.
- Εισαγωγή επιπλέον χαρακτηριστικών στο μετεωρολογικό μοντέλο (π.χ. υγρασία, ύψος βροχόπτωσης, πίεση κ.τ.λ.).

Μια πιο ουσιαστική ωστόσο επέκταση της διπλωματικής θα ήταν η προσαρμογή του συστήματος ώστε να μπορεί να πραγματοποιεί λήψη όχι αποκλειστικά μετεωρολογικών δεδομένων, αλλά κάθε είδους πληροφορίας που μπορεί να ενδιαφέρει το χρήστη (για παράδειγμα χρηματιστηριακά, δεδομένα). Μια τέτοια επέκταση είναι καθ' όλα εφικτή καθώς, με τη δυνατότητα που έχει δοθεί στο χρήστη να καθορίζει ο ίδιος αν το επιθυμεί τις κανονικές εκφράσεις για κάθε χαρακτηριστικό του καιρού αλλά και να διασαφηνίζει τη φύση των αποτελεσμάτων εξαγωγής δίνοντάς τους τον προσδιορισμό με τον οποίο θέλει να του παρουσιάζονται, μπορεί τελικά να επιτύχει την εξαγωγή κάθε είδους πληροφορίας.

9 *Βιβλιογραφία*

[DG] DevGuru – JavaScript OBJECT:RegExp, <http://www.devguru.com/Technologies/ecmaScript/quickref/regexp.html>

[HEV] Hevanet Communications – XUL Periodic Table, <http://www.hevanet.com/acorbin/xul/top.xul>

[MDC1] Mozilla Developer Center – Preferences API, <http://developer.mozilla.org/en/docs/Preferences_API>

[MDC2] Mozilla Developer Center – The Joy of XUL, <http://developer.mozilla.org/en/docs/The_Joy_of_XUL>

[MDC3] Mozilla Developer Center – XUL Tutorial, <http://developer.mozilla.org/en/docs/XUL_Tutorial>

[MDC4] Mozilla Developer Center – Code snippets:Preferences, <http://developer.mozilla.org/en/docs/Code_snippets:Preferences>

[MDC5] Mozilla Developer Center – Extensions, <http://developer.mozilla.org/en/docs/Extensions>

[MDC6] Mozilla Developer Center – Building an Extension, <http://developer.mozilla.org/en/docs/Building_an_Extension>

[MDC7] Mozilla Developer Center – Install Manifests, <http://developer.mozilla.org/en/docs/Install_Manifests#aboutURL>

[MDC8] Mozilla Developer Center – Installing Extensions and Themes from Web pages, <http://developer.mozilla.org/en/docs/Installing_Extensions_and_Themes_From_Web_Pages#Web_Script_Example>

[MDC9] Mozilla Developer Center – Parsing and Serializing XML, <http://developer.mozilla.org/en/docs/Parsing_and_serializing_XML>

[MDC10] Mozilla Developer Center – Core JavaScript 1.5 Guide:Regular Expressions, <http://developer.mozilla.org/en/docs/Core_JavaScript_1.5_Guide:Regular_Expressions>

[MO] Mozilla – Firefox Add-ons, <https://addons.mozilla.org/el/firefox/search?q=forecast&cat=all>

[PG] Piggy Bank – PiggyBank Introduction - SIMILE, <http://simile.mit.edu/wiki/Piggy_Bank_Introduction>

[WP1] Wikipedia, The Free Encyclopedia – Web Browsers, <http://en.wikipedia.org/wiki/Web_browsers>

[WP2] Wikipedia, The Free Encyclopedia – Web Scraping, <http://en.wikipedia.org/wiki/Web_scraping>

[WP3] Wikipedia, The Free Encyclopedia – Regular Expression, <http://en.wikipedia.org/wiki/Regular_expression>

[WP4] Wikipedia, The Free Encyclopedia – Cascading Style Sheets, <http://en.wikipedia.org/wiki/Cascading_Style_Sheets>

[WP5] Wikipedia, The Free Encyclopedia – JavaScript, <http://en.wikipedia.org/wiki/JavaScript>

[WP6] Wikipedia, The Free Encyclopedia – XUL, <http://en.wikipedia.org/wiki/Xul>

[WP7] Wikipedia, The Free Encyclopedia – DOM, <http://en.wikipedia.org/wiki/Document_Object_Model>

[WP8] Wikipedia, The Free Encyclopedia – Mozilla Firefox, <http://en.wikipedia.org/wiki/Mozilla_Firefox>

[WP9] Wikipedia, The Free Encyclopedia – RSS, <http://en.wikipedia.org/wiki/RSS>

[WR] WebReference.com – Regular Expressions, <http://www.webreference.com/js/column5/>

[W3S1] W3 Schools – CSS2 Reference, <http://www.w3schools.com/css/css_reference.asp#font>

[W3S2] W3 Schools – JavaScript Tutorial, <http://www.w3schools.com/js/default.asp>

[W3S3] W3 Schools – RSS Tutorial, <http://www.w3schools.com/rss/rss_intro.asp>

[XP1] XUL Planet – Interface Reference-nsIPrefBranch, <http://www.xulplanet.com/references/xpcomref/ifaces/nsIPrefBranch.html >

[XP2] XUL Planet – Interface Reference-nsIPrefBranch2, <http://www.xulplanet.com/references/xpcomref/ifaces/nsIPrefBranch2.html>

[XP3] XUL Planet – Interface Reference-nsIPrefService, <http://www.xulplanet.com/references/xpcomref/ifaces/nsIPrefService.

html>

[XP4] XUL Planet – XMLHttpRequest, <http://xulplanet.com/references/objref/XMLHttpRequest.html>

[AGM03] A. Arasu and H. Garcia-Molina, “Extracting Structured Data from Web Pages,” Proc. ACM SIGMOD Int’l Conf. Management of Data, pp. 337-348, 2003.

[AM98] G.O. Arocena and A.O. Mendelzon, “WebOQL: Restructuring Documents, Databases, and Webs,” Proc. 14th IEEE Int’l Conf. Data Eng. (ICDE), pp. 24-33, 1998.

[CK04] C.-H. Chang and S.-C. Kuo, “OLERA: A Semisupervised Approach for Web Data Extraction with Visual Support,” IEEE Intelligent Systems, vol. 19, no. 6, pp. 56-64, 2004.

[CKGS06] C.-H. Chang, M. Kayed, M.R. Girgis, K.F. Shaalan, “A Survey of Web Information Extraction Systems”, IEEE transactions on knowledge and data engineering (IEEE trans. knowl. data eng.), vol. 18, no. 10, pp. 1411-1428, 2006

[CM98] M. Califf and R. Mooney, “Relational Learning of Pattern-Match Rules for Information Extraction,” Proc. AAAI Spring Symp. Applying Machine Learning to Discourse Processing Mar. 1998.

[CMM01] V. Crescenzi, G. Mecca, and P. Merialdo, “RoadRunner: Towards-Automatic Data Extraction from Large Web Sites,” Proc. the 26th Int’l Conf. Very Large Database Systems (VLDB), pp. 109-118, 2001.

[CRM98] V. Crescenzi and G. Mecca, “Grammars Have Exceptions,” Information Systems, vol. 23, no. 8, pp. 539-565, 1998.

[F98] D. Freitag, “Information Extraction from HTML: Application of a General Learning Approach,” Proc. 15th Conf. Artificial Intelligence (AAAI ’98), 1998.

[HCHLC05] C.-N. Hsu, C.-H. Chang, C.-H. Hsieh, J.-J. Lu, and C.-C. Chang, “Reconfigurable Web Wrapper Agents for Biological Information Integration,” J. Am. Soc. for Information Science and Technology, special issue on bioinformatics, vol. 56, no. 5, pp. 505-517, 2005.

[HHM97] J. Hammer, J. McHugh, and H. Garcia-Molina, “Semistructured Data: The TSIMMIS Experience,” Proc. First East-European Symp. Advances in Databases and Information Systems (ADBIS), pp. 1-8, 1997.

[HK05] A. Hogue and D. Karger, “Thresher: Automating the Unwrapping of Semantic Content from the World Wide,” Proc. 14th Int’l Conf. World Wide Web (WWW), pp. 86-95, 2005.

[KVSP01] K. Karageorgos, C. Valakas, Y. Stavrakas, A. Polyrakis, “Designing and Implementing a Wrapper Specification Language for Web Information Sources”, Proc. 8th Panhellenic Conference on Informatics, 2001

[MMK99] I. Muslea, S. Minton, and C. Knoblock, “A Hierarchical Approach to Wrapper Induction,” Proc. Third Int’l Conf. Autonomous Agents (AA ’99), 1999.

[SA01] A. Saiiuguet and F. Azavant, “Building Intelligent Web Applications Using Lightweight Wrappers,” Data and Knowledge Eng., vol. 36, no. 3, 283-316, 2001.

[WL02] J. Wang and F.H. Lochovsky, “Wrapper Induction Based on Nested Pattern Discovery,” Technical Report HKUST-CS-27-02, Dept. of Computer Science, Hong Kong, Univ. of Science & Technology, 2002.

[WL03] J. Wang and F.H. Lochovsky, “Data Extraction and Label Assignment for Web Databases,” Proc. 12th Int’l Conf. World Wide Web (WWW), pp. 187-196, 2003.