ΤΕΧΝΟΛΟΓΙΚΟ ΕΚΠΑΙΔΕΥΤΙΚΟ ΙΔΡΥΜΑ ΣΕΡΡΩΝ
ΣΧΟΛΗ ΤΕΧΝΟΛΟΓΙΚΩΝ ΕΦΑΡΜΟΓΩΝ
ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΕΠΙΚΟΙΝΩΝΙΩΝ

ΚΑΤΑΝΟΗΣΗ ΔΙΑΔΙΚΤΥΑΚΩΝ ΕΦΑΡΜΟΓΩΝ ΓΙΑ ΣΧΕΔΙΑΣΜΟ ΣΗΜΑΣΙΟΛΟΓΙΚΟΥ ΔΙΑΔΙΚΤΥΟΥ

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ ΤΟΥ

ΔΗΜΗΤΡΗ ΛΙΑΣΟΠΟΥΛΟΥ (Α.Μ. 378)

ΕΠΙΒΛΕΠΩΝ : Δρ. ΘΕΟΔΩΡΟΣ ΛΑΝΤΖΟΣ ΕΠΙΣΤΗΜΟΝΙΚΟΣ ΣΥΝΕΡΓΑΤΗΣ

ΤΕΧΝΟΛΟΓΙΚΟ ΕΚΠΑΙΔΕΥΤΙΚΟ ΙΔΡΥΜΑ ΣΕΡΡΩΝ
ΣΧΟΛΗ ΤΕΧΝΟΛΟΓΙΚΩΝ ΕΦΑΡΜΟΓΩΝ
ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΕΠΙΚΟΙΝΩΝΙΩΝ

ΚΑΤΑΝΟΗΣΗ ΔΙΑΔΙΚΤΥΑΚΩΝ ΕΦΑΡΜΟΓΩΝ ΓΙΑ ΣΧΕΔΙΑΣΜΟ ΣΗΜΑΣΙΟΛΟΓΙΚΟΥ ΔΙΑΔΙΚΤΥΟΥ

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ ΤΟΥ

ΔΗΜΗΤΡΗ ΛΙΑΣΟΠΟΥΛΟΥ (Α.Μ. 378)

ΕΠΙΒΛΕΠΩΝ : Δρ. ΘΕΟΔΩΡΟΣ ΛΑΝΤΖΟΣ ΕΠΙΣΤΗΜΟΝΙΚΟΣ ΣΥΝΕΡΓΑΤΗΣ

ΣΕΡΡΕΣ 2009

ΠΕΡΙΛΗΨΗ

Στα σπάργανα βρίσκεται τα τελευταία χρόνια ένας νέος Ιστός, ο Σημασιολογικός Ιστός. Ο Σημασιολογικός Ιστός (Semantic Web) είναι στην ουσία μια εξέλιξη του ήδη υπάρχοντα Ιστού παρά ένας νέος Ιστός και ο εμπνευστής του, Tim Berners-Lee , είχε το όραμα ενός Ιστού δεδομένων αυτόματα επεξεργάσιμων από τις εφαρμογές βάσει του νοήματος (σημασίας) και όχι της μορφής της πληροφορίας. Ο Σημασιολογικός Ιστός έχει σαν βάση την XML. Τα έγγραφα XML είναι αρχεία κειμένου και η ανάκτηση των δεδομένων μέσω της σειριακής προσπέλασης αρχείων δεν είναι αποτελεσματική. Το κενό αυτό έρχεται να καλύψει η διαδικασία της λεκτικής ανάλυσης η οποία αναλύει το κείμενο στα επιμέρους στοιχεία του (αρχή ετικέτας, τέλος ετικέτας, κείμενο, ιδιότητες). Η διαδικασία της λεκτικής ανάλυσης γίνεται με ένα λεκτικό αναλυτή. Ένας από τους πιο γνωστούς και αυτός που ασχοληθήκαμε εμείς στην παρούσα πτυχιακή είναι ο DOM (Document Object Model). Το DOM έχει την δυνατότητα να αποθηκεύει τα δεδομένα εγγράφου ως δομές δένδρων στην μνήμη. Το DOM αναπαριστά κάθε στοιχείο του εγγράφου XML ως κόμβο και δίνει την δυνατότητα στο χρήστη να δημιουργήσει ένα XML κείμενο, να πλοηγηθεί μέσα σε αυτό να προσθέσει, να μεταβάλει και να αφαιρέσει στοιχεία του. Υπάρχουν πολλά προγράμματα στην αγορά τα οποία αναλύουν το DOM και δίνουν την δυνατότητα στον χρήστη να πλοηγηθεί σε αυτό κάνοντας ότι αλλαγή θέλει.

Σκοπός αυτής της πτυχιακής είναι να καταλάβουμε τι είναι τελικά αυτός ο Σημασιολογικός Ιστός. Να καταλάβουμε την δομή του, τα πλεονεκτήματα που μας προσφέρει και φυσικά να μελετήσει την δομή των νέων ιστοσελίδων και τον τρόπο κατανόησης τους με στόχο τη υποστήριξη σχεδιασμού σημασιολογικού δικτύου. Επίσης στη εργασία αυτή παρουσιάζεται τι είναι οι Διαδικτυακές Υπηρεσίες και πως θα μετέχουν αυτές στον Σημασιολογικό Ιστό. Δώσαμε έμφαση στο Μοντέλο Αντικειμένου Εγγράφου (DOM) το οποίο φροντίζει για την ανάγνωση αλλά και την τροποποίηση των εγγράφων XML. Αναλύσαμε θεωρητικά το DOM και προσπαθήσαμε, με κάποια έτοιμα προγράμματα τα οποία αναλύουν το DOM της κάθε σελίδας, να πλοηγηθούμε σε αυτό και να το τροποποιήσουμε.

ΠΕΡΙΕΧΟΜΕΝΑ

Ευρετήριο Σχημάτων

Ευρετήριο Πινάκων

ΚΕΦΑΛΑΙΟ 1

Σημασιολογικός Ιστός (Semantic Web)

ΕΙΣΑΓΩΓΗ

Σύμφωνα με τα λόγια πολλών ειδικών που ασχολούνται με τις τελευταίες τάσεις τις τεχνολογίας του Internet, συμβαίνει μια επανάσταση στο παγκόσμιο ιστό. Η επανάσταση αυτή έχει σαν σκοπό να δώσει στον ιστό νόημα και θα προσπαθήσει να μας μεταφέρει σε έναν κόσμο όπου τα πάντα θα βασίζονται στη γνώση. Όλα αυτά βέβαια θα πρέπει να είναι κατανοητά και επεξεργάσιμα από τους υπολογιστές. Η εξέλιξη αυτή του Παγκόσμιου ιστού ονομάζεται Σημασιολογικός Ιστός (Semantic Web).

Ο σημερινός παγκόσμιος ιστός (World Wide Web) έχει σχεδιαστεί κυρίως για την χρήση από ανθρώπους. Τελευταία όμως παρατηρείται μια αυξημένη αυτοματοποίηση διαδικασιών μέσω της χρησιμοποίησης Διαδικτυακών Υπηρεσιών (Web Services) κυρίως σε εφαρμογές ηλεκτρονικού εμπορίου (e-commerce) και σε εφαρμογές B2B (Business-to-Business). Προς το παρόν μια τέτοια λειτουργικότητα επιτυγχάνεται μέσω διεπαφών (API) οι οποίες ανιχνεύουν και εξάγουν πληροφορία από HTML σελίδες. Υπάρχει όμως ένα πρόβλημα οποιαδήποτε αλλαγή στην δομή τους συνεπάγεται και προσαρμογή των διεπαφών αυτών ώστε να λειτουργήσουν και πάλι με τα νέα δεδομένα. (π.χ. εργαλεία τα οποία ψάχνουν μεταξύ ηλεκτρονικών καταστημάτων να βρουν ένα συγκεκριμένο προϊόν με την χαμηλότερη τιμή).

Με λίγα λόγια θα μπορούσαμε να πούμε ότι ο Σημασιολογικός Ιστός ή Σημασιολογικό διαδίκτυο (Semantic Web) αποτελεί τη μεγαλύτερη προσπάθεια αυτόματης ενοποίησης συστημάτων ώστε να συνεργάζονται διαλειτουργικά σε παγκόσμιο επίπεδο.

Ο Tim Berners-Lee, ο οποίος επινόησε το WWW (World Wide Web) το 1989, είχε το όραμα ενός ιστού δεδομένων αυτόματα επεξεργάσιμων από τις εφαρμογές, βάσει του νοήματος και όχι της μορφής της πληροφορίας.

Το κέντρο βάρους των περιεχομένων του διαδικτύου μετατοπίζεται συνεχώς από το ελεύθερο κείμενο που είναι πλήρως κατανοητό μόνο από τον άνθρωπο, προς την ημιδομημένη πληροφορία ή και πλήρως δομημένη πληροφορία η οποία μπορεί να γίνει αυτόματα κατανοητή από διαδικτυακές εφαρμογές (π.χ. διαδικτυακές υπηρεσίες, ευφυείς πράκτορες κ.α.). Για να φτάσει το διαδίκτυο στο μέγιστο των δυνατοτήτων του, πρέπει να εξελιχθεί σε μια τέτοια μορφή στην οποία να παρέχει μια παγκοσμίως προσβάσιμη πλατφόρμα που να επιτρέπει σε αυτοματοποιήμενα εργαλεία (πράκτορες) να διαμοιράζονται και να επεξεργάζονται πληροφορίες και δεδομένα για λογαριασμό ανθρώπων-χρηστών τους.

1.1 Γενικά

Είναι ξεκάθαρο σε όλους πλέον ότι ο Παγκόσμιος Ιστός έχει επηρεάσει σε μεγάλο βαθμό τον τρόπο που ενεργούμε επιχειρηματικά. Πολλές γραμμές σε πολλά βιβλία έχουν γραφτεί για τις ευκαιρίες που ανοίγει το Web στις εταιρίες να επεκταθούν σε παγκόσμιο επίπεδο. Όμως κάτι λείπει από την συνολική εικόνα. Πλέον έχουμε συλλέξει περισσότερα δεδομένα από ποτέ, διαφόρων ειδών δεδομένα που μπορεί να είναι δεδομένα πωλήσεων μέχρι μηχανικά σχέδια και τα έχουμε διαθέσιμα σε όλους μέσω του διαδικτύου. Έτσι η αναζήτηση που πραγματοποιούμε κάθε φορά γίνεται όλο και πιο δύσκολη.

Ο Παγκόσμιος Ιστός αυτήν την στιγμή περιέχει πάνω από 3 δισεκατομμύρια στατικά έγγραφα, τα οποία είναι προσβάσιμα πάνω από 500 εκατομμύρια χρήστες σε παγκόσμια βάση. Επιπρόσθετα, αυτό το τεράστιο ποσό πληροφορίας είναι εξαιρετικά δύσκολο να αναβρεθεί, προσβαστεί, παρουσιαστεί και συντηρηθεί διότι το περιεχόμενο της πληροφορίας αυτής παρουσιάζεται σε φυσική γλώσσα η οποία κατανοείται αποκλειστικά από τους ανθρώπους. Κάτι τέτοιο έχει σαν αποτέλεσμα να έχει δημιουργηθεί ένα μεγάλο κενό μεταξύ της πληροφορίας που είναι διαθέσιμη σε εργαλεία όπως μηχανές αναζήτησης και της πληροφορίας η οποία συντηρείται σε μορφή κατανοητή από τον άνθρωπο. Όπως αντιλαμβανόμαστε αυτό είναι ένα σημαντικό πρόβλημα για τον κάθε άνθρωπο.

Η πληροφορία που διακινείται μέσω του Παγκόσμιου Ιστού είναι πολύτιμη για εκατομμύρια χρήστες. Υπάρχουν δύο βασικά μοντέλα αναζήτησης πληροφορίας στον Παγκόσμιο Ιστό (WWW) η αναζήτηση με την χρήση ερωτημάτων (search by query) και η αναζήτηση με την χρήση πλοήγησης (search by browsing), τα οποία είναι ευρέως γνωστά ως αναζήτηση και πλοήγηση. Σύμφωνα με το μοντέλο της αναζήτησης, η πρόσβαση στις πηγές πληροφόρησης του Παγκόσμιου Ιστού πραγματοποιείται μέσω δικτυακών μηχανών αναζήτησης (web search engines), ενώ σύμφωνα με το μοντέλο πλοήγησης, η πρόσβαση στην πληροφορία του Παγκόσμιου Ιστού πραγματοποιείται μέσω δικτυακών καταλόγων (web directories). Και τα δύο μοντέλα αναζήτησης έχουν αποτελέσει αντικείμενο διερεύνησης πολλών εργασιών, με στόχο να μελετηθούν τόσο οι γνωστικές διεργασίες που συντελούνται από τους χρήστες κάθε φορά που υιοθετούν ένα από τα δύο μοντέλα, όσο και οι συνθήκες υπό τις οποίες οι χρήστες επιλέγουν ένα μοντέλο έναντι του άλλου κατά την επικοινωνία τους με το διαδικτύο.

Πιο συγκεκριμένα, κατά την αναζήτηση, οι χρήστες καταφεύγουν σε κάποια δικτυακή μηχανή αναζήτησης π.χ. Google (www.google.com) , η οποία παρέχει μέσω μιας διεπαφής χρήστη (interface) ένα πεδίο διατύπωσης ερωτημάτων (search box) , όπου οι χρήστες πληκτρολογούν το πληροφοριακό τους αίτημα (search query) σε φυσική γλώσσα, υπό την μορφή λέξεων-κλειδιών (keywords). Στη συνέχεια η μηχανή αναζήτησης κάνοντας χρήση τεχνικών επεξεργασίας των ερωτημάτων ανατρέχει στο ευρετήριο ιστοσελίδων (index) που διατηρεί και εντοπίζει στο σύνολο των ευρετηριοποιημένων ιστοσελίδων, τις σελίδες εκείνες που περιέχουν τους ίδιους όρους με τους όρους του ερωτήματος του χρήστη. Οι σελίδες αυτές επιστρέφονται στο χρήστη ταξινομημένες ανάλογα με τον βαθμό στον οποίο απαντούν το πληροφοριακό του αίτημα.

Αντίθετα κατά την πλοήγηση οι χρήστες ανατρέχουν σε κάποιο από τους διαθέσιμους δικτυακούς καταλόγους π.χ. yahoo directory (www.yahoo.com) dmoz directory (http://dmoz.org) google directory (http://dir.google.com) οι οποίοι τους παρέχουν μέσω μιας διεπαφής ένα σύνολο ιεραρχικά δομημένων θεματικών κατηγοριών, τη λεγόμενη θεματική ιεραρχία (thematic/subject hierarchy). Οι χρήστες εξετάζουν τις παρεχόμενες θεματικές κατηγορίες, επιλέγουν την θεματική κατηγορία που τους ενδιαφέρει και φυλλομετρούν (browse) τις ιστοσελίδες που έχουν ανατεθεί

σε κάθε κατηγορία μέχρι να εντοπίσουν τις ιστοσελίδες που ικανοποιούν το πληροφοριακό τους αίτημα. Οι θεματικές κατηγορίες των καταλόγων Διαδικτύου ορίζονται σύμφωνα με ένα σχήμα ιεραρχικής οργάνωσης, το οποίο επιτρέπει την αποτελεσματική πρόσβαση στα δεδομένα του Παγκόσμιου Ιστού, μέσω μιας διαδικασίας ιεραρχικής πλοήγησης (hierarchy navigation process) που συντελείται μέσω της απλής ακολουθίας συνδέσμων.

Παρά τις εγγενείς διαφοροποιήσεις που παρουσιάζουν μεταξύ τους τα μοντέλα της αναζήτησης και της πλοήγησης, εμφανίζουν συμπληρωματικά πλεονεκτήματα για τους χρήστες που επιχειρούν να ικανοποιήσουν πολύπλοκα πληροφοριακά αιτήματα στον Παγκόσμιο Ιστό. Συγκεκριμένα, το μοντέλο της αναζήτησης υιοθετείται συνήθως όταν ο χρήστης έχει μια αποκρυσταλλωμένη εικόνα για το πληροφοριακό του αίτημα, το οποίο είναι σε θέση να διατυπώσει ρητά μέσω επιλεγμένων λέξεων-κλειδιών που απευθύνει στις δικτυακές μηχανές αναζήτησης. Εναλλακτικά, ο χρήστης υιοθετεί το μοντέλο της πλοήγησης είτε όταν δεν έχει αποκρυσταλλώσει την ανάγκη του για πληροφόρηση και συνεπώς αδυνατεί να την διατυπώσει, είτε όταν η πληροφοριακή του ανάγκη είναι αρκετά γενικευμένη και ικανοποιείται από ένα σύνολο πηγών πληροφόρησης που σχετίζονται με την ανάγκη αυτή. Βασική προϋπόθεση για την ικανοποιητική απόδοση και των δύο μοντέλων αναζήτησης είναι η ανάπτυξη ευέλικτων και εύρωστων μηχανισμών διαχείρισης του πλήθους των δεδομένων του Παγκόσμιου Ιστού, οι οποίοι επιτυγχάνουν την ποιοτική οργάνωση των ιστοσελίδων που δεικτοδοτούνται στα ευρετήρια μηχανών αναζήτησης και δικτυακών καταλόγων προς όφελος των τελικών χρηστών που καταφεύγουν σε αυτά για την ικανοποίηση των πληροφοριακών τους αναγκών.

Για την ικανοποίηση των πληροφοριακών αιτημάτων που απευθύνονται σε δικτυακές μηχανές αναζήτησης έχουν αναπτυχθεί διάφορες αυτοματοποιημένες τεχνικές που στοχεύουν στη βελτίωση της εμπειρίας του χρήστη κατά την αναζήτηση πληροφορίας από τον Παγκόσμιο Ιστό. Οι τεχνικές αυτές αφορούν στην προσκομιδή, δεικτοδότηση και επεξεργασία ιστοσελίδων, στον εντοπισμό των ιστοσελίδων που είναι σχετικές με συγκεκριμένα πληροφοριακά αιτήματα, στη μέτρηση της σπουδαιότητας των ιστοσελίδων, καθώς και στη σειρά ταξινόμησής τους κατά την παρουσίασή τους στους χρήστες.

Αντίθετα, η πρόοδος που έχει συντελεστεί για τη βελτίωση της εμπειρίας των χρηστών που ανατρέχουν στους δικτυακούς καταλόγους για την ικανοποίηση των πληροφοριακών τους αιτημάτων, είναι σημαντικά δυσανάλογη με αυτήν της αναζήτησης. Ο περιορισμένος ρυθμός εξέλιξης που χαρακτηρίζει τους καταλόγους του Παγκόσμιου Ιστού, πηγάζει από την περιορισμένη δυνατότητα παροχής αποδοτικών και αυτοματοποιημένων τεχνικών γλωσσικής επεξεργασίας. Πιο αναλυτικά, εφόσον η δημιουργία, η οργάνωση και η συντήρηση δικτυακών καταλόγων πραγματοποιείται με βάση το σημασιολογικό περιεχόμενο των ιστοσελίδων που φιλοξενούνται σε αυτούς, καθίσταται προφανές πως οι περιορισμένες δυνατότητες των διαθέσιμων τεχνικών σημασιολογικής επεξεργασίας, εισάγουν περιορισμούς στη δυνατότητα αυτοματοποιημένης κατασκευής δικτυακών καταλόγων. Τούτο έχει ως αποτέλεσμα η δημιουργία καταλόγων Διαδικτύου να στηρίζεται στην αναντικατάστατη ανθρώπινη προσπάθεια. Όμως ο ρυθμός αύξησης και μεταβολής των δεδομένων που διακινούνται στον Παγκόσμιο Ιστό καθιστά τη χειρωνακτική επεξεργασία των ιστοσελίδων μια εξαιρετικά χρονοβόρα, επίπονη και κυρίως αναποτελεσματική διαδικασία, εφόσον ο αριθμός των ατόμων που επιμελείται της επεξεργασίας ιστοσελίδων είναι σημαντικά δυσανάλογος του αριθμού των τελευταίων. Αρκεί να αναφέρουμε πως ο δικτυακός κατάλογος με τον μεγαλύτερο γνωστό αριθμό επιμελητών Διαδικτύου (web editors), το Dmoz Directory

(http://dmoz.org), απασχολεί σήμερα περισσότερους από 70.000 εθελοντές, ενώ ο αριθμός των ιστοσελίδων που μας είναι γνωστές ανέρχεται περίπου στις 9.000.000.000 και διαρκώς αυξάνεται.

Η λύση σε αυτό το πρόβλημα που αναλύσαμε διεξοδικά θα μπορούσε να είναι η δημιουργία ενός νέου Web βασιζόμενου στο συντακτικό και τις έννοιες τις γλώσσας μας αλλά να είναι ουσιαστικά αόρατος στους ανθρώπους. Το όραμα αυτό ονομάζεται Σημασιολογικός Ιστός (Semantic Web) και προορίζεται αυτή την φορά για τις μηχανές και όχι για τους ανθρώπους όπως το συμβατικό Web. Ο στόχος είναι να επιτρέψει στους υπολογιστές όχι απλά να επεξεργάζονται και να μεταφέρουν δεδομένα αλλά να κατανοούν το νόημά τους.

1.2 Δομή Σημασιολογικού ιστού

Στην πραγματικότητα ο Σημασιολογικός Ιστός είναι περισσότερο μια επέκταση του σημερινού Παγκόσμιου Ιστού παρά ένας τελείως καινούργιος Ιστός. Προτάθηκε από τον πνευματικό πατέρα του Παγκόσμιου Ιστού, τον Tim Berners-Lee, και η πιο πιθανή μορφή του είναι η προσθήκη εξειδικευμένων tags μέσα σε ένα έγγραφο HTML, που εκτός από να προσδιορίζει την ιστοσελίδα, θα βοηθάει τους υπολογιστές να κατανοούν το περιεχόμενο της. Το όραμα του Tim Berners-Lee χωρίζεται σε δύο μέρη. Το πρώτο μέρος έχει στόχο να κάνει το Web ένα πιο συνεργάσιμο μέσο. Το δεύτερο είναι αυτό που ασχολείται με το να μετατρέψει το Web σε μια οντότητα επεξεργάσιμη και κατανοήσιμη από τους υπολογιστές.

Θα μπορούσαμε να αναρωτηθούμε γιατί δεν αρκούν οι τρέχουσες τεχνικές ευρετηριοποίησης και ανάκτησης δεδομένων που χρησιμοποιούν οι δημοφιλείς μηχανές αναζήτησης. Οι μηχανές αναζήτησης του σήμερα (Google, Yahoo, AltaVista) αποτελούν τα κυριότερα εργαλεία χρήσης του Web και όπως ξέρουμε η λειτουργία τους βασίζεται σε λέξεις κλειδιά (keywords). Είναι ξεκάθαρο ότι η τεράστια επιτυχία του Web δεν θα ήταν τόσο μεγάλη χωρίς την συνδρομή των μηχανών αναζήτησης. Παρ' όλ' αυτά, υπάρχουν σοβαρά προβλήματα-μειονεκτήματα που σχετίζονται με την χρήση τους.

Υψηλή ανάκληση(recall), χαμηλή ακρίβεια(precision). Ακόμα και εάν οι κύριες σχετικές σελίδες ανακτηθούν σωστά, δεν θα μπορούν να χρησιμοποιηθούν εάν υπάρχουν ακόμα 50000 ελάχιστα σχετικές ή άσχετες με το αποτέλεσμα.

Πολύ χαμηλή ή μηδενική ανάκληση. Αυτό το πρόβλημα είναι λιγότερο σύνηθες αλλά δεν παύει να ισχύει. Συνήθως συμβαίνει να μην λαμβάνουμε απάντηση στο αίτημα μας ή οι πραγματικές σελίδες δεν ανακτώνται.

Τα αποτελέσματα είναι πολύ ευαίσθητα στο λεξιλόγιο. Συχνά οι αρχικές λέξεις κλειδιά δεν φέρνουν τα αποτελέσματα που επιθυμούμε. Αυτό συμβαίνει επειδή τα έγγραφα που περιέχουν την πληροφορία που θέλουμε περιέχουν διαφορετική ορολογία από την δική μας. Κάτι τέτοιο δεν είναι ικανοποιητικό διότι ερωτήματα που είναι σημασιολογικά παρόμοια θα έπρεπε να επιστρέφουν παρόμοια αποτελέσματα.

Τα αποτελέσματα που επιστρέφονται είναι μοναδικές ιστοσελίδες. Εάν χρειαζόμαστε πληροφορία που βρίσκεται διασκορπισμένη σε πολλές ιστοσελίδες, πρέπει να εκτελέσουμε πάνω από ένα ερωτήματα και να συλλέξουμε όλα τα έγγραφα χειροκίνητα. Στην συνέχεια θα πρέπει πάλι χειροκίνητα να εξάγουμε την πληροφορία που θέλουμε από κάθε έγγραφο και να την ενοποιήσουμε σε ένα.

Αν και οι τεχνολογίες των μηχανών αναναζήτησης βελτιώνονται σημαντικά, (γίνονται γρηγορότερες και καλύπτουν όλο και μεγαλύτερο μέρος του Παγκόσμιου Ιστού), τα παραπάνω προβλήματα παραμένουν τα ίδια. Φαίνεται πως ο ρυθμός

γιγάντωσης του Παγκόσμιου Ιστού ξεπερνά το ρυθμό της τεχνολογικής προόδου. Αλλά ακόμα και αν οι αναζητήσεις γίνουν επιτυχείς, ο άνθρωπος-χρήστης είναι αυτός που πρέπει να αναζητήσει μέσα στα επιλεγμένα-ανακτημένα έγγραφα για να βρει την πληροφορία που θέλει. Με λίγα λόγια, δεν υπάρχει ακόμα η υποστήριξη για την εύρεση της πληροφορίας μέσα σε ένα έγγραφο, παρά μόνο ανακάλυψη της συγκεκριμένης τοποθεσίας όπου βρίσκεται το έγγραφο. Τέλος, τα αποτελέσματα της αναζήτησης τείνουν να είναι αρκετά απομονωμένες εφαρμογές.

Για να γίνει εφικτή η επεξεργασία της γνώσης που περιέχεται στον Παγκόσμιο Ιστό από τους υπολογιστές, θα πρέπει να επέλθει μια ριζική αλλαγή στον τρόπο με τον οποίο σκεφτόμαστε τα δεδομένα. Αρχικά, τα δεδομένα θεωρούταν μικρότερης σημασίας από την επεξεργασία των δεδομένων. Σύντομα κάτι τέτοιο φάνηκε ότι είναι μια λανθασμένη άποψη και έχει αποδειχθεί στην πράξη ότι η χρησιμότητα του λογισμικού είναι καθολικά εξαρτημένη από την ποιότητα των δεδομένων στα οποία βασίζεται(π.χ. GIGO-Garbage in, Garbage out). Κατά συνέπεια, οι σχεδιαστές συστημάτων, οι γλώσσες προγραμματισμού και γενικά η επιστήμη των υπολογιστών με τις ραγδαία εξελισσόμενες τεχνολογίες βάσεων δεδομένων και αναπαράστασης τους, έχουν δώσει στα δεδομένα ναι αρκετά σημαντική θέση. Με τις νέες τεχνολογίες XML, τον Παγκόσμιο Ιστό και τον ανερχόμενο Σημασιολογικό Ιστό, η προσοχή μεταφέρεται όλο και περισσότερο από τις εφαρμογές στα δεδομένα. Για να μπορέσει όμως ένας υπολογιστής να κατανοήσει το νόημα των δεδομένων, αυτά θα πρέπει να γίνουν πιο «έξυπνα».

Ας προσπαθήσουμε να ταξινομήσουμε τα δεδομένα με βάσει την «εξυπνάδα» τους.

Αρχεία κειμένου και εγγραφές βάσεων δεδομένων (Text document and database records): Αποτελούν την αρχική μορφή κατά την οποία το μεγαλύτερο μέρος των δεδομένων ανήκει σε μια εφαρμογή. Κατά συνέπεια η «εξυπνάδα» βρίσκεται στην εφαρμογή και όχι στα δεδομένα.

Αρχεία XML για ένα γνωστικό πεδίο (domain): Τα δεδομένα πλέον είναι ανεξάρτητα από τις εφαρμογές και κινούνται μέσα σε ένα συγκεκριμένο γνωστικό πεδίο (problem domain). Τα δεδομένα είναι πλέον αρκετά ευφυή ώστε να μπορούν να μετακινούνται μεταξύ εφαρμογών του ίδιου γνωστικού πεδίου. Παράδειγμα αποτελούν τα XML Standards για την ιατρική επιστήμη ή το Real Estate.

Ιεραρχίες (taxonomies) και έγγραφα με αναμεμειγμένα λεξικά (vocabularies): Στο στάδιο αυτό είναι δυνατή η σύνθεση δεδομένων από διαφορετικά γνωστικά πεδία και η ταξινόμηση τους σε μια ιεραρχία. Στην πραγματικότητα, κάτι τέτοιο χρησιμοποιείται για την ανακάλυψη των δεδομένων. Πραγματοποιούνται απλές συσχετίσεις μεταξύ των δεδομένων ώστε να ανακαλύπτονται εύκολα και να συνδυάζονται λογικά με άλλα δεδομένα.

Οντολογίες και κανόνες: Στο σημείο αυτό, υπάρχει η δυνατότητα να συμπεράνουμε νέα δεδομένα από ήδη υπάρχοντα ακολουθώντας λογικούς κανόνες. Στην ουσία, τα δεδομένα είναι αρκετά έξυπνα ώστε να περιγράφουν με συγκεκριμένες σχέσεις (concrete relationships) και εξελιγμένους φορμαλισμούς και δίνουν την δυνατότητα για λογικούς υπολογισμούς. Με άλλα λόγια, δίνεται η δυνατότητα εξελιγμένου συνδυασμού των δεδομένων όπως για παράδειγμα η μετάφραση ενός εγγράφου από το ένα γνωστικό πεδίο στο άλλο.

1.3 Χρησιμότητα του Σημασιολογικού Ιστού

Ο Σημασιολογικός Ιστός εκπροσωπεί μια ομάδα τεχνολογιών που θα λειτουργήσουν εξίσου καλά και στον Παγκόσμιο Ιστό αλλά και στα εταιρικά δίκτυα (intranets). Για αυτό το λόγο, υπόσχεται να λύσει ορισμένα προβλήματα-κλειδιά που αντιμετωπίζουν οι τρέχουσες πληροφοριακές υποδομές.

Υπερφόρτωση Πληροφορίας (Information Overload)

Αποτελεί το πιο προφανές πρόβλημα που χρειάζεται να λύσει ο Σημασιολογικός Ιστός καθώς οι ειδικοί της τεχνολογίας μας προειδοποιούν εδώ και 50 περίπου χρόνια. Σύμφωνα με τον Paul Krill, «Η κατάσταση αυτή είναι το αποτέλεσμα της ραγδαίας αύξησης στην διαθέσιμη πληροφορία, ενώ οι ημέρες παραμένουν 24 ώρες και το ανθρώπινο μυαλό παραμένει στην ίδια κατάσταση εξέλιξης με άνθρωπο των σπηλαίων». Μπορούμε να φανταστούμε πόσο γιγαντώνεται το πρόβλημα αυτό με την παγκόσμια διάδοση του Internet και του email.

Stovepipe Systems

Τα Stovepipe Systems είναι συστήματα όπου όλα τα συστατικά τους είναι φτιαγμένα έτσι ώστε να λειτουργούν μόνο μεταξύ τους. Η πληροφορία παραμένει εγκλωβισμένη σε ένα τέτοιο σύστημα και δεν μπορεί να χρησιμοποιηθεί από άλλους οργανισμούς ή συστήματα που τη χρειάζονται.

Ανεπαρκής συγκέντρωση περιεχομένου (content aggregation)

Η συγκέντρωση πληροφορίας από διαφορετικές πηγές είναι ένα πρόβλημα το οποίο συναντάται πολλές φορές σε διαφορετικές περιοχές. Δυστυχώς, οι τεχνικές που χρησιμοποιούνται σήμερα είναι ανεπαρκείς, λύνουν μόνο βραχυπρόθεσμα τα προβλήματα και δεν εστιάζουν στο νόημα των εγγράφων και χρειάζονται συνεχή και επίπονη συντήρηση.

1.4. Από τον σημερινό Παγκόσμιο Ιστό στον Σημασιολογικό Ιστό

Στο περιβάλλον του σημερινού Παγκόσμιου Ιστού γίνονται αρκετές προσπάθειες ώστε η πληροφορία και η επεξεργασία της να είναι πιο «έξυπνη». Μπορεί ο Σημασιολογικός Ιστός να καλύψει τα κενά; Στις παρακάτω ενότητες δίνονται ορισμένα παραδείγματα.

1.4.1 Διαχείριση Γνώσης

Η διαχείριση γνώσης (knowledge management) ασχολείται με την απόκτηση, πρόσβαση και διαχείριση της γνώσης μέσα σε έναν οργανισμό. Έχει αναδειχθεί σαν μια αρκετά βασική λειτουργία μια μεγάλης επιχείρησης διότι κάθε μεγάλη επιχείρηση βλέπει την γνώση σαν ένα ανεκτίμητο διανοητικό κεκτημένο (intellectual asset) από το οποίο μπορούν να αντλήσουν μεγαλύτερη παραγωγικότητα, να δημιουργήσουν νέα

αξία και να αυξήσουν την ανταγωνιστικότητα. Η διαχείριση της γνώσης είναι ιδιαίτερα σημαντική για παγκόσμιους οργανισμούς με γεωγραφικά διασκορπισμένα τμήματα.

1.4.2. Ηλεκτρονικό Εμπόριο B2C (Business to Consumer)

Το ηλεκτρονικό εμπόριο B2C είναι η κυρίαρχη εμπορική πρακτική για τους χρήστες του παγκόσμιου ιστού. Ένα τυπικό σενάριο περιλαμβάνει την επίσκεψη ενός χρήστη σε έναν αριθμό από ηλεκτρονικά καταστήματα, την περιήγηση στις προσφορές και τα προϊόντα τους και τέλος την επιλογή και παραγγελία ενός ή περισσοτέρων από αυτά. Σε μια ιδανική περίπτωση ένας χρήστης θα συνέλεγε πληροφορίες σχετικά με τις τιμές, τους όρους και την διαθεσιμότητα όλων ή των περισσοτέρων σχετικών προϊόντων διαθέσιμων από αρκετά ηλεκτρονικά καταστήματα και μετά θα επέλεγε την καλύτερη προσφορά. Όμως, η χειροκίνητη αναζήτηση αποδεικνύεται πολύ χρονοβόρα και μάλιστα σε ένα τέτοιο επίπεδο. Συνήθως ο χρήστης θα επισκεφτεί μικρό αριθμό καταστημάτων πριν κάνει την τελική του απόφαση.

Υπάρχουν εργαλεία τα οποία λειτουργούν ως Agents οι οποίοι επισκέπτονται τα ηλεκτρονικά καταστήματα και δημιουργούν μια επισκόπηση της αγοράς. Η λειτουργία τους όμως εξαρτάται από wrappers οι οποίοι θα πρέπει να έχουν φτιαχτεί για κάθε κατάστημα. Η προσέγγιση αυτή έχει ορισμένα μειονεκτήματα- το σημαντικότερο βρίσκεται στην διαδικασία της εξαγωγής της πληροφορίας. Αυτή βασίζεται στην ανάλυση κειμένου και στην έρευνα μέσω λέξεων-κλειδιά όπου λαμβάνονται κάποιες παραδοχές για το πόσο κοντά βρίσκεται η μια πληροφορία στην άλλη, για παράδειγμα ότι η τιμή ενός προϊόντος βρίσκεται μετά από την λέξη price και το σήμα $. Αυτή η ευριστική προσέγγιση είναι ευαίσθητη σε λάθη και δεν εγγυάται ότι θα δουλεύει πάντοτε σωστά. Ο Σημασιολογικός Ιστός θα επιτρέψει την δημιουργία software agents οι οποίοι θα μπορούν να ερμηνεύσουν όλες τις πληροφορίες ενός προϊόντος αλλά και τους όρους παροχής μιας υπηρεσίας.

1.4.3 Ηλεκτρονικό εμπόριο B2B (Business to Business)

Οι περισσότεροι συνδέουν το εμπορικό κομμάτι του Web με το B2C αλλά η μεγαλύτερη οικονομική υπόσχεση από όλες τις άλλες δικτυακές τεχνολογίες βρίσκεται στο ηλεκτρονικό εμπόριο B2B. Παραδοσιακά οι επιχειρήσεις αντάλλασσαν δεδομένα χρησιμοποιώντας την EDI προσέγγιση η οποία όμως είναι αρκετά πολύπλοκη και κατανοητή μόνο από εξειδικευμένα στελέχη. Είναι γενικά δύσκολη στον προγραμματισμό και την συντήρηση και είναι και αυτή ευαίσθητη σε λάθη. Επιπλέον, είναι μια τεχνολογία απομονωμένη καθώς τα δεδομένα που ανταλλάσσονται μεταξύ δύο επιχειρήσεων δεν είναι προσβάσιμα από άλλες. Και σε αυτό το στάδιο υπάρχουν ήδη εργαλεία που διευκολύνουν το B2B όπως τα B2B Portals και τα B2B Marketplaces. Όμως, η HTML είναι πολύ αδύναμη ώστε να μπορέσει να υποστηρίξει τέτοιες λειτουργίες.

Η έλευση του Σημασιολογικού Ιστού θα επιτρέψει στις επιχειρήσεις να δημιουργούν συνεργασίες χωρίς μεγάλο κόστος. Οι διαφορές στην ορολογία θα επιλύονται άμεσα χρησιμοποιώντας αφαιρετικά μοντέλα για κάθε γνωστικό πεδίο (abstract domain models) και τα δεδομένα θα ανταλλάσσονται μεταξύ των επιχειρήσεων χρησιμοποιώντας τις υπηρεσίες μετάφρασης. Δημοπρασίες,

διαπραγματεύσεις και άλλες συναλλαγές θα μπορούν να πραγματοποιούνται αυτόματα ή ημιαυτόματα μέσω των software agents.

1.5 Το παρόν του Σημασιολογικού Ιστού

Είναι προφανές ότι οι εταιρίες θα μπορούσαν να χρησιμοποιήσουν έναν πιο αποτελεσματικό τρόπο να δομήσουν τα δεδομένα και να βρίσκουν την σχετική πληροφορία οπουδήποτε είναι αποθηκευμένη. Αλλά ο χαρακτηρισμός αδόμητων δεδομένων είναι μόνο το πρώτο βήμα, όπως έχουμε δει. Η πραγματική δύναμη έρχεται αργότερα, όταν οι υπολογιστές θα μπορούν να «δουν» τα δεδομένα και να τα κατανοήσουν, όταν δηλαδή τα δεδομένα φτάσουν στο ανώτερο στάδιο «εξυπνάδας».

Φυσικά, υπάρχουν λόγοι για τους οποίους αρκετά έξυπνες εταιρίες όπως η Microsoft περιμένουν την εξέλιξη του Σημασιολογικού Ιστού και δεν τον υιοθετούν. Ο Σημασιολογικός Ιστός είναι ακόμα μακριά από το σημείο όπου θα είναι έτοιμο για ευρεία χρήση και πρέπει να ξεπεραστούν σημαντικά εμπόδια πριν εμφανιστούν τα πραγματικά τεχνολογικά οφέλη. Όπως κάθε τεχνολογικό υπολογιστικό περιβάλλον, για να πετύχει, θα πρέπει ένα μεγάλο ποσοστό χρηστών και εταιρειών να συμφωνήσουν σε πρωτόκολλα και standards που θα πρέπει να υιοθετούν για την λειτουργία του. Η XML είναι μια γλώσσα περιγραφής δεδομένων με την οποία όλοι έχουν εξοικειωθεί, και αποτελεί το πρώτο βήμα για την αποδοχή του Σημασιολογικού Ιστού, όπως και άλλες τεχνολογίες (Διαδικτυακές Υπηρεσίες-Web Services) οι οποίες θα αποτελέσουν και τα βασικά δομικά του υλικά.

1.5.1 Σημασιολογικός Ιστός και σχετικές τεχνολογίες

Ο Σημασιολογικός Ιστός αποτελεί για πολλούς μια επέκταση του υπάρχοντος Παγκόσμιου Ιστού, με έμφαση όχι πλέον στην παρουσίαση, αλλά στην ανταλλαγή πληροφορίας κατανοητής από τους υπολογιστές (machine readable content). Αν και εισάγονται νέες γλώσσες και τεχνολογίες που θα βοηθήσουν στην πραγματοποίηση του οράματος του Σημασιολογικού Ιστού, βασικά δομικά υλικά του αποτελούν τεχνολογίες οι οποίες περιλαμβάνονται ήδη στον συμβατικό Παγκόσμιο Ιστό.

1.5.2. Η XML και ο Σημασιολογικός Ιστός

Η γλώσσα XML αποτελεί την βάση για τον Σημασιολογικό Ιστό. Όλες οι τεχνολογίες που θα παρέχουν τα χαρακτηριστικά για τον Σημασιολογικό Ιστό θα βασίζονται πάνω στην XML. Με αυτό τον τρόπο θα είναι εγγυημένο ένα αρχικό επίπεδο συνεργασιμότητας (interoperability). Οι τεχνολογίες στις οποίες βασίζεται η XML είναι χαρακτήρες Unicode και οι Uniform Resource Identifiers (URI). Τα URI χρησιμοποιούνται σαν μοναδικοί προσδιοριστές γνωστικών πεδίων στον Σημασιολογικό Ιστό. Η XML όμως παρέχει μόνο συντακτική συνεργασιμότητα – η ανταλλαγή ενός εγγράφου XML μεταξύ δύο μερών έχει νόημα μόνο εάν και τα δύο μέρη γνωρίζουν και καταλαβαίνουν τα ονόματα των στοιχείων. Π.χ. εάν μία εταιρία έχει το στοιχείο `<price>$12,00</price>` στο τιμολόγιο ενώ η άλλη το `<cost>$12,00</cost>`, δεν υπάρχει τρόπος ένας υπολογιστής να γνωρίζει ότι τα

δύο στοιχεία σημαίνουν το ίδιο εκτός εάν εφαρμοστούν οι τεχνολογίες του Σημασιολογικού Ιστού.

1.5.3 Διαδικτυακές Υπηρεσίες και Σημασιολογικός Ιστός

Οι διαδικτυακές υπηρεσίες είναι λογισμικό το οποίο αναγνωρίζεται από ένα URI και περιγράφεται, ανακαλύπτεται και είναι προσβάσιμο μέσων των πρωτοκόλλων Web. Το σημαντικότερο σημείο των διαδικτυακών υπηρεσιών είναι ότι μπορούν να εκμεταλλεύονται αλλά και να επιστρέφουν ως αποτέλεσμα έγγραφα XML. Η εξέλιξη των διαδικτυακών υπηρεσιών είναι ραγδαία, και αρχίζουν να παίρνουν την μορφή ιστοσελίδων, με την διαφορά ότι ακόμα είναι πολύ δύσκολο να ανακαλυφθούν από τους χρήστες. Η χρήση τεχνολογιών του Σημασιολογικού Ιστού θα είναι απαραίτητες για να λυθεί το πρόβλημα ανακάλυψης των διαδικτυακών υπηρεσιών –τότε θα ονομάζονται Σημασιολογικές Διαδικτυακές Υπηρεσίες (Semantic Web Services).

1.6 Τεχνολογίες που εισάγονται με την έλευση του Σημασιολογικού Ιστού

Το όραμα του Σημασιολογικού Ιστού ενδυναμώνεται συνεχώς από την γέννηση νέων τεχνολογιών και εργαλείων όπως το RDF και η OWL που βοηθούν στην αναπαράσταση της πληροφορίας σε μορφή εύκολα κατανοητή και επεξεργάσιμη από τους πράκτορες του Σημασιολογικού Ιστού. Αν και η αναπαράσταση της πληροφορίας αποτελεί προγενέστερο θέμα μελέτης της Τεχνητής Νοημοσύνης, στην περίπτωση του Σημασιολογικού Ιστού η προσπάθεια η οποία γίνεται εδώ, είναι και προς την κατεύθυνση της σωστής και επιτυχημένης δια-συνεργασίας των δομικών στοιχείων του.

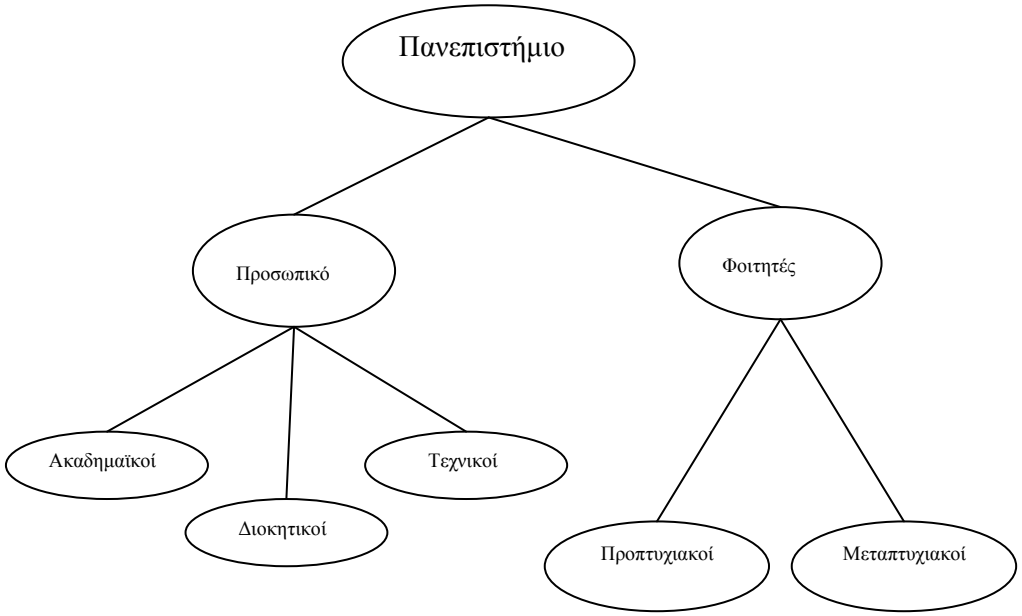
1.6.1 Ρητά Μεταδεδομένα (Explicit Metadata)

Ο Σημασιολογικός Ιστός χρησιμοποιεί εκτενώς την XML για την περιγραφή των περιεχομένων μιας ιστοσελίδας ή ενός ιστοτόπου. Η αναπαράσταση με XML έχει το πλεονέκτημα από την HTML στο ότι είναι πολύ πιο επεξεργάσιμη από τους υπολογιστές. Με άλλα λόγια, ενώ η HTML προσφέρεται για την οπτική αναπαράσταση στους Web Browsers και κατά συνέπεια στους χρήστες, η XML αναπαριστά τα μετά-δεδομένα δηλαδή τα δεδομένα για τα δεδομένα. Τα μεταδεδομένα αναπαριστούν επιτυχώς ένα μέρος της έννοιας που έχουν τα δεδομένα και επιπλέον είναι επεξεργάσιμα από τους υπολογιστές.

1.6.2 Οντολογίες

Στην επιστήμη των υπολογιστών, ο όρος Οντολογία έχει καθιερωθεί ως η επίσημη περιγραφή μιας έννοιας. Τυπικά περιέχει έναν πεπερασμένο αριθμό όρων και των σχέσεων μεταξύ τους. Οι όροι αναπαριστούν σημαντικές έννοιες (κλάσεις). Π.χ. στο περιβάλλον ενός πανεπιστημίου, σημαντικές έννοιες είναι το προσωπικό, οι

φοιτητές, τα μαθήματα, οι αίθουσες. Στις σχέσεις συνήθως αναπαριστώνται οι ιεραρχίες των κλάσεων αυτών. Για το παράδειγμα μας, μια πιθανή ιεραρχία φαίνεται στο παρακάτω σχήμα .



σχήμα 1) Μία ιεραρχία

Εκτός όμως από την ιεραρχία οι οντολογίες μπορούν να περιέχουν πληροφορίες όπως

- Ιδιότητες(ο Χ διδάσκει το Υ)
- Περιορισμούς τιμών (Μόνο οι ακαδημαϊκοί διδάσκουν μαθήματα)
- Διαχωρισμοί (Το διοικητικό προσωπικό και το Ακαδημαϊκό είναι χωριστοί)
- Ορισμός λογικών σχέσεων μεταξύ αντικειμένων (κάθε τμήμα πρέπει να περιέχει τουλάχιστον 50 φοιτητές)

Οι οντολογίες παρέχουν μια κοινή κατανόηση ενός γνωστικού πεδίου. Μια τέτοια προσέγγιση μπορεί να καταργήσει τις διαφορές στην ορολογία ανάμεσα στους φορείς στο ίδιο γνωστικό πεδίο. Επίσης μπορεί να λύσει το πρόβλημα που υπάρχει όταν δύο φορείς έχουν την ίδια ορολογία για διαφορετικές έννοιες. Αυτό μπορεί να γίνει εφικτό εάν η γνώση απεικονιστεί σε μια κοινή οντολογία και οριστούν αντιστοιχίες της οντολογίας αυτής με την ορολογία του φορέα. Η ανάπτυξη γλωσσών για οντολογίες είναι εδώ και αρκετό καιρό αντικείμενο έρευνας της Τεχνητής Νοημοσύνης και αποτελεί θεμελιώδη τεχνολογία στην οποία θα «χτιστεί» ο Σημασιολογικός Ιστός. Οι σημαντικότερες γλώσσες οντολογίας για τον Ιστό είναι οι παρακάτω:

1. Η XML παρέχει το βασικό συντακτικό για δομημένα έγγραφα αλλά δεν βάζει σημασιολογικούς περιορισμούς στους όρους του εγγράφου.
2. Η XML Schema είναι μια γλώσσα η οποία επιβάλλει περιορισμούς στα έγγραφα XML
3. Το RDF είναι ένα μοντέλο δεδομένων για αντικείμενα και σχέσεις μεταξύ των αντικειμένων. Παρέχει βασική σημασιολογική λειτουργία και το μοντέλο αναπαριστάται σε σύνταξη XML.

4. Το RDF Schema είναι μια γλώσσα περιγραφής λεξικού που περιγράφει ιδιότητες και κλάσεις των αντικειμένων RDF, με σημασιολογική αναπαράσταση σχέσεων γενίκευσης (generalization).
5. Η QWL είναι μια πιο πλούσια γλώσσα περιγραφής λεξικού από το RDF Schema, στο ότι επιτρέπει αναπαράσταση του cardinality, πιο πλούσιο ορισμό των ιδιοτήτων και των χαρακτηριστικών τους κ.λ.π.

1.6.3.Λογική

Η λογική στην Τεχνητή Νοημοσύνη προσφέρει επίσημες γλώσσες για την αναπαράσταση της γνώσης και κατανοητή σημασιολογική πληροφορία. Κάτι τέτοιο βοηθάει σε αυτοματοποιημένους εξαγωγείς γνώσεις να βγάζουν νέα επαγωγικά συμπεράσματα από την ήδη υπάρχουσα γνώση. Το σημαντικό πλεονέκτημα της λογικής είναι ότι μπορεί να παρέχει εξηγήσεις για το συμπέρασμα που βγάζει, εάν κάποιος ακολουθήσει τα βήματα που εκτελέστηκαν ώστε να βγεί το τελικό αποτέλεσμα. Οι εξηγήσεις αυτές είναι σημαντικές για τον Σημασιολογικό Ιστό, διότι μπορούν να αυξήσουν την εμπιστοσύνη των χρηστών στους πράκτορες του Σημασιολογικού Ιστού.

1.6.4 Πράκτορες (Agents)

Οι πράκτορες αποτελούν κομμάτια λογισμικού τα οποία λειτουργούν αυτόματα. Σαν έννοια προέρχονται από τις αντίστοιχες έννοιες του αντικειμενοστραφούς προγραμματισμού και την ανάπτυξη λογισμικού βασισμένη σε components. Ένας «προσωπικός» πράκτορας, δηλαδή ενός χρήστη του Σημασιολογικού Ιστού, θα λάβει κάποια καθήκοντα και προτιμήσεις από τον χρήστη, θα αναζητήσει πληροφορία από πηγές στον Παγκόσμιο Ιστό, θα επικοινωνήσει με άλλους πράκτορες, θα συγκρίνει τις πληροφορίες με τις προτιμήσεις και τις απαιτήσεις του χρήστη και τελικά αφού κάνει κάποιες επιλογές, θα δώσει τις τελικές απαντήσεις στον χρήστη.

Οι πράκτορες στο Σημασιολογικό Ιστό θα κάνουν χρήση όλων των τεχνολογιών που προαναφέρθηκαν ως εξής:

- Τα Μεταδεδομένα πρόκειται να χρησιμοποιηθούν στον εντοπισμό και εξαγωγή πληροφορίας από πηγές του Παγκόσμιου Ιστού
- Οι Οντολογίες θα χρησιμοποιηθούν ώστε να γίνουν πιο εύκολες οι αναζητήσεις στον Παγκόσμιο Ιστό και να γίνει εφικτή η επικοινωνία με άλλους πράκτορες
- Η Λογική θα χρησιμοποιηθεί για την επεξεργασία της πληροφορίας που βρέθηκε και για την εξαγωγή συμπερασμάτων

Ας δούμε ένα παράδειγμα για να καταλάβουμε καλύτερα τους πράκτορες.

Έστω κάποια έγκυος γυναίκα Χ που επισκέπτεται το γυναικολόγο στα πλαίσια της τακτικής μηνιαίας εξέτασης. Ο γυναικολόγος διακρίνει κάτι το ανησυχητικό στο υπερηχογράφημα, το οποίο όμως δεν έχει αρκετά μεγάλη διακριτική ικανότητα ώστε να μπορέσει να κάνει διάγνωση, και προτείνει εξέταση από κάποιο διαγνωστικό κέντρο με καλύτερο εξοπλισμό. Η Χ αναθέτει στο διαδικτυακό πράκτορα της να βρει

κάποιο τέτοιο διαγνωστικό κέντρο. Θα ήταν επιθυμητό το διαγνωστικό κέντρο να συνεργάζεται με τον ασφαλιστικό της φορεά ώστε να μην χρειαστεί να πληρώσει.

Ο πράκτορας:

Ενημερώνεται από τον πράκτορα του γυναικολόγου για το είδος της εξέτασης που πρέπει να γίνει.

Συνδέεται με την υπηρεσία του ασφαλιστικού ταμείου για να βρει ποια διαγνωστικά κέντρα που συνεργάζονται με το ταμείο μπορούν να πραγματοποιήσουν την εξέταση.

Επιλέγει διαγνωστικά κέντρα που βρίσκονται κοντινότερα στη δουλειά ή στο σπίτι.

Ελέγχει την ποιότητα του καθενός χρησιμοποιώντας μια ανεξάρτητη υπηρεσία (εξουσιοδοτημένη από το Υπουργείο Υγείας να αξιολογεί διαγνωστικά κέντρα, νοσοκομεία κ.λ.π.).

Στη συνέχεια ο πράκτορας:

Επικοινώνησε με τους πράκτορες όσων διαγνωστικών κέντρων ικανοποιούσαν τα κριτήρια, ώστε να βρεθούν οι διαθέσιμες ώρες για την εξέταση σύμφωνα με το πρόγραμμα της Χ

Παρουσίασε δύο επιλογές.

Καμία όμως δεν ήταν ικανοποιητική.

Στην πρώτη επιλογή το διαθέσιμο ραντεβού ήταν σε δύο εβδομάδες, ενώ η εξέταση είναι επείγουσα.

Η δεύτερη επιλογή απαιτούσε την παραμονή της Χ στο κέντρο της πόλης 3 ώρες μετά την λήξη του ωραρίου εργασίας της.

Η Χ αποφάσισε να επαναλάβει το αίτημα στον πράκτορα θέτοντας αυστηρότερα χρονικά όρια.

Ο πράκτορας βρήκε μία νέα λύση.

Βρισκόταν μέσα στα χρονικά όρια που τέθηκαν.

Δεν ικανοποιούσε μερικά από τα αρχικά κριτήρια.

Βρέθηκε ένα διαγνωστικό κέντρο στην περιοχή της κατοικίας της Χ

Είχε διαθέσιμο ραντεβού την επόμενη ημέρα.

Δε συνεργάζεται με το ασφαλιστικό ταμείο γιατί χρεώνει μεγαλύτερο ποσό από το προβλεπόμενο.

Ο πράκτορας έλεγξε ότι το ταμείο θα επέστρεφε το μέγιστο ποσό που προβλεπόταν για την εξέταση αυτή, συνεπώς η Χ θα πλήρωνε μόνο την διαφορά.

Το διαγνωστικό κέντρο συγκαταλέγεται μέσα στα καλύτερα σύμφωνα με μια ανεξάρτητη αλλά πιστοποιημένη υπηρεσία αξιολόγησης τους, από την οποία βρήκε το διαγνωστικό κέντρο ο πράκτορας.

Ένα ακόμα πρόβλημα που πρέπει να επιλυθεί είναι ότι τι ραντεβού είναι ακριβώς την ώρα που αρχίζει η Χ δουλειά.

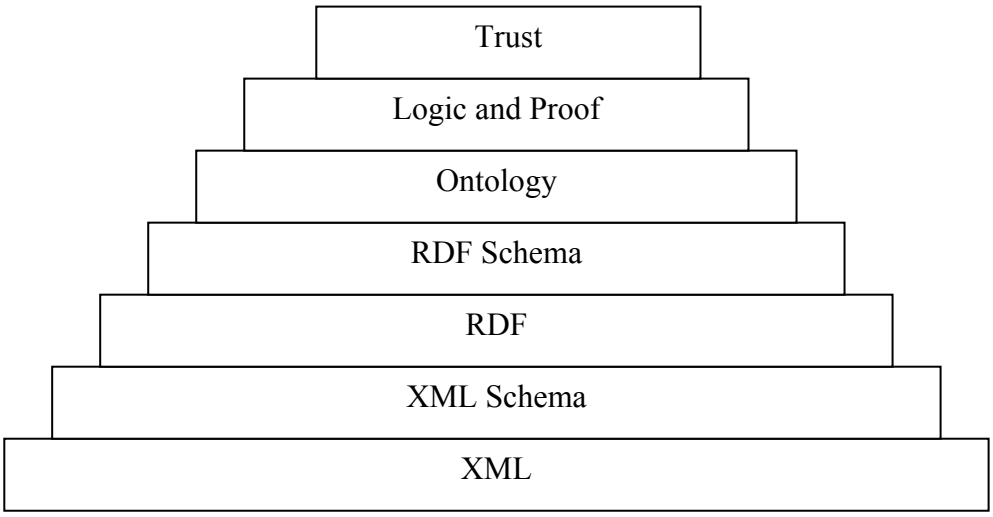
Θα πρέπει να ειδοποιηθεί ο πράκτορας του εργοδότη για την καθυστερημένη άφιξή της Χ την επόμενη ημέρα.

Ο πράκτορας προσφέρθηκε να διεκπεραιώσει την εργασία εφόσον η Χ ενέκρινε τη λύση.

Η Χ είδε πως τα κριτήρια που δεν ικανοποιεί αυτή η λύση δεν είναι και τόσο ανελαστικά, συνεπώς αποφάσισε να την υιοθετήσει.

1.7 Η δομή και τα επίπεδα του Σημασιολογικού Ιστού

Το World Wide Web Consortium (W3C) είναι ηγέτης στην ανάπτυξη τεχνολογιών για τον Παγκόσμιο Ιστό. Ηγέτης του είναι ο Tim Berners-Lee που εκτός από το μεγάλο κατόρθωμα να κάνει το Web παγκόσμιο πρότυπο, οραματίστηκε τον Σημασιολογικό Ιστό και προωθεί πλέον την ανάπτυξη του. Η προσέγγιση του W3C στην εξέλιξη του Σημασιολογικού Ιστού είναι η λεγόμενη αρχιτεκτονική του, βάση της οποίας ο Σημασιολογικός Ιστός αποτελείται από διαφορετικές στρώσεις (layers) των οποίων η ύπαρξη υποδηλώνει ότι το ανώτερο επίπεδο εξαρτάται πλήρως και «χτίζεται» πάνω στο κατώτερο. Η αρχιτεκτονική αυτή φαίνεται στο παρακάτω σχήμα



σχήμα 2 : Η αρχιτεκτονική του Σημασιολογικού Ιστού σύμφωνα με το W3C

Η αρχιτεκτονική αυτή, την οποία ονομάζουν και «Semantic Web Cake», αποτελείται από τα παρακάτω επίπεδα, ξεκινώντας από το κατώτατο προς το ανώτερο.

1. XML: Η γλώσσα η οποία χρησιμοποιείται από το 1998 ώστε να ορίζει όλες τις νέες γλώσσες οι οποίες θα χρησιμοποιηθούν για την ανταλλαγή δεδομένων στον Παγκόσμιο Ιστό
2. XML Schema: Όπως αναφερθήκαμε και προηγουμένως, ορίζει αυστηρά τη δομή των εγγράφων XML
3. RDF: Όπως και παραπάνω, η περιγραφή πληροφορίας σε μορφή μετά-δεδομένων
4. RDF Schema: Ορισμός του λεξικού που θα χρησιμοποιεί η γλώσσα RDF
5. Οντολογίες: Ο κοινός ορισμός εννοιών ενός γνωστικού πεδίου
6. Λογική και Απόδειξη: Η λογική για την οποία μιλήσαμε παραπάνω, θα βοηθήσει στην εξαγωγή αληθινών συμπερασμάτων
7. Εμπιστοσύνη: Μέσα για την παροχή αυθεντικοποίησης της ταυτότητας και απόδειξη της αξιοπιστίας των δεδομένων, των υπηρεσιών και των πρακτόρων. Στο επίπεδο αυτό αντιμετωπίζονται ζητήματα εμπιστοσύνης όπως ποιο είναι το κοινά αποδεκτό βιομηχανικό πρότυπο για οντολογίες σε κάποιο γνωστικό

πεδίο ,όπως π.χ. στην βιοϊατρική. Η εμπιστοσύνη μπορεί να είναι εφικτή μέσω των ήδη διαδεδομένων ψηφιακών υπογραφών.

Η πρακτική αυτή, δηλαδή της αρχιτεκτονικής σε επίπεδα, κάνει πιο εύκολη την σύμπνοια για τις τεχνολογίες που θα πρέπει να υιοθετηθούν, η οποία πραγματοποιείται σε μικρά βήματα. Γενικά, για το «χτίσιμο» ενός επιπέδου πάνω στο άλλο θα πρέπει να ακολοθούνται δύο αρχές οι οποίες είναι οι παρακάτω:

1. Συμβατότητα με τα κατώτερα επίπεδα: Οι πράκτορες οι οποίοι θα γνωρίζουν πλήρως ένα επίπεδο θα πρέπει να ερμηνεύουν και να χρησιμοποιούν πληροφορία που προέρχεται από τα κατώτερα επίπεδα.
2. Μερική κατανόηση των ανωτέρων επιπέδων: Από την άλλη πλευρά, οι ίδιοι πράκτορες θα πρέπει να εκμεταλλεύονται τουλάχιστον σε κάποιο μέρος την πληροφορία ανωτέρων επιπέδων.

1.8. Ο ρόλος των Διαδικτυακών Υπηρεσιών

Αν και οι διαδικτυακές υπηρεσίες δεν αναφέρονται ρητά στην αρχιτεκτονική σου Σημασιολογικού Ιστού, ο συνδυασμός τους με τον Σημασιολογικό Ιστό μπορεί να βοηθήσει σημαντικά στην αυτοματοποίηση της ανακάλυψης, σύνθεσης, εκτέλεσης και ελέγχου των υπηρεσιών που παρέχονται στο διαδίκτυο αρκεί να χρησιμοποιηθούν κατάλληλες περιγραφές, ώστε να είναι αναγνωρίσιμες από τους πράκτορες.

Εκτός από την καθαρή πληροφορία που βρίσκεται στο διαδίκτυο, πολλές επιχειρήσεις παρέχουν μορφή παροχής υπηρεσιών μέσω αυτού. Οι διαδικτυακές υπηρεσίες είναι η νέα ανερχόμενη μορφή παροχής υπηρεσιών η οποία θα επιτρέψει επιπλέον την αλληλεπίδραση του διαδικτύου με παλιά, legacy συστήματα που τις παρείχαν ως τώρα. Για να γίνει εύκολη η ανακάλυψη τέτοιων υπηρεσιών, θα πρέπει να γίνει σωστή και δυναμική χρήση των οντολογιών. Εάν οι οντολογίες που δημιουργηθούν είναι κατανοητές από πράκτορες που θα τις εντοπίζουν αυτόματα, τότε θα δοθεί μια επιπλέον ώθηση στο ηλεκτρονικό εμπόριο όπου οι χρήστες του διαδικτύου εκτός από προϊόντα θα μπορούν να αγοράζουν και υπηρεσίες μέσω διαδικτύου αλλά και να τους παρέχονται οι υπηρεσίες αυτές μέσα από το διαδίκτυο.

1.9 Η γλώσσα XML ως θεμέλιο του Σημασιολογικού Ιστού

Χωρίς αμφιβολία η γλώσσα XML εμφανίζεται από τους περισσότερους ειδικούς της πληροφορικής ως η νέα τεχνολογική τάση. Είναι μια νέα γλώσσα η οποία περιγράφει τα δεδομένα και όχι την παρουσίαση τους, όπως η αδελφή HTML. Μπορεί να περιγράψει ένα οποιοδήποτε είδος εγγράφου σε μορφή κατανοητή και ανταλλάξιμη από τους υπολογιστές, ενώ με απλούς μετασχηματισμούς όπως το XSLT, μπορεί να μετατραπεί σε οποιονδήποτε κοινό τύπο δεδομένων για χρήση ακόμα και από παλαιότερα συστήματα που δεν την υποστηρίζουν.

1.9.1 Η επιτυχία της XML

Η XML πλέον έχει περάσει από την φάση της αρχικής έγκρισης και υιοθεσίας στην καθολική αποδοχή. Σήμερα, η κύρια χρήση της XML είναι η ανταλλαγή δεδομένων μεταξύ εσωτερικών και εξωτερικών οργανισμών. Αποτελεί δηλαδή έναν κοινά αποδεκτό μηχανισμό συνεργασίας. Καθώς οι τεχνολογίες XQuery και XMLSchema ωριμάζουν και υιοθετούνται ακόμα περισσότερο, η XML προβλέπεται να γίνει το βασικό συντακτικό για όλα τα δεδομένα μιας επιχείρησης. Η επιτυχία της και η ευρεία αποδοχή της οφείλεται κυρίως στα παρακάτω χαρακτηριστικά:

1. Δημιουργία εγγράφων και δεδομένων ανεξαρτήτων από τις εφαρμογές
2. Πρότυπο συντακτικό για μεταδεδομένα
3. Πρότυπη δομή για έγγραφα και δεδομένα
4. Τεχνολογία που βρίσκεται σε ισχύ εδώ και αρκετό καιρό
5. Υπάρχει πλέον η υπολογιστική ισχύς για να εκμεταλλευτούμε τα προτερήματα της XML
6. Είναι αναγνώσιμη και από τον άνθρωπο αλλά και από τους υπολογιστές

Η XML στην πραγματικότητα δεν είναι γλώσσα προγραμματισμού όπως η Java. Είναι επιπλέον ένα σύνολο συντακτικών κανόνων για την δημιουργία γλωσσών σήμανσης (markup languages) σε κάποιο συγκεκριμένο γνωστικό πεδίο. Με άλλα λόγια, χρησιμοποιούμε την XML για να φτιάξουμε νέες γλώσσες, όπως για παράδειγμα η MathMl που αποτελεί εφαρμογή της XML στο γνωστικό πεδίο των μαθηματικών. Η γλώσσα σήμανσης έχει σαν βασικό σκοπό να προσθέσει πληροφορία και νόημα στο ακατέργαστο περιεχόμενο ενός εγγράφου. Είναι σημαντικό όμως να πούμε ότι η σημασιολογική πληροφορία που μπορεί να προσδώσει η XML μπορεί να επισυναφθεί σε ένα έγγραφο.

1.9.2 Σύνταξη της XML

Για να καταλάβουμε πως συντάσσεται ένα έγγραφο XML, θα εξετάσουμε ένα παράδειγμα και θα δούμε τις ομοιότητες με την γλώσσα HTML που όλοι γνωρίζουμε.

```
<?xml version=”1.0”?>
<book type=”Computer Science”>
     <title>
         System Analysis and Design with Uml 2.0: An
Object-Oriented Approach
     </title>
     <author>A.Dennis</author>
     <author>B.Wixom</author>
     <author>D.Tegarden</author>
     <publisher>John Wiley And Sons</publisher>
     <year>2005</year>
     <ISBN>0471659207</ISBN>
</book>
```

Η XML επιτρέπει στον χρήστη να ονομάσει τα στοιχεία όπως θέλει, σε αντίθεση με την HTML η οποία περιορίζει τον χρήστη σε συγκεκριμένες ετικέτες. Δεν ενδιαφέρει την XML η χρήση του εγγράφου, ο τρόπος με τον οποίο θα

παρουσιαστεί ούτε καν τι σημαίνουν τα στοιχεία. Αρκεί να ακολουθούνται οι βασικοί κανόνες για την σήμανση. Συνήθως όμως χρησιμοποιούνται ονόματα στα στοιχεία τα οποία είναι σχετικά με την πληροφορία που περιγράφει το έγγραφο.

Το παράδειγμα αυτό, όπως και όλα τα έγγραφα XML, αποτελείται από τα σύμβολα σήμανσης. Ονομάζουμε ετικέτες τα ονόματα τα οποία περικλείονται από τα σύμβολα <>. Οι ετικέτες βοηθούν στην αναγνώριση και τον ορισμό της δομής του εγγράφου. Ετικέτες και κείμενο δημιουργούν μια οντότητα πληροφορίας όπως π.χ. το όνομα ή το ISBN του βιβλίου στο παράδειγμα.

1.9.3 Ο πρόλογος του εγγράφου XML

Η αρχική γραμμή του εγγράφου είναι ο πρόλογος του εγγράφου και στην πιο απλή του μορφή υποδηλώνει ότι το έγγραφο μας είναι XML, όπως στο παράδειγμα μας με το βιβλίο, αλλά μπορεί να κρατάει και άλλες πληροφορίες όπως την δήλωση τύπου εγγράφου (DTD), την κωδικοποίηση του κειμένου και κάποιες οδηγίες προς τους επεξεργαστές XML. Ο πρόλογος μπορεί να αποτελείται από παραπάνω από δύο γραμμές, όπως στο παράδειγμα που ακολουθεί:

```
1:   <?xml version="1.0" encoding="utf-8"?>
2:   <!DOCTYPE time-o-gram
3:        PUBLIC "-//LordsOfTime//DTD TimeOGram 1.8//EN"
4:        "htto://www.lordsoftime.org/DTDs/timeogram.dtd"
5:   [
6:        <!ENTITY sj "Sarah Jane">
7:        <!ENTITY me "Doctor Who"
8:   ]>
```

Μελετώντας των κώδικα XML γραμμή προς γραμμή, βλέπουμε ότι οι γραμμές του παραπάνω παραδείγματος δηλώνουν τα παρακάτω:

1: Η γραμμή αυτή δηλώνει στον αναγνώστη (υπολογιστή) ότι θα χρειαστεί έναν XML Parser για να μπορέσει να ερμηνεύσει το έγγραφο αυτό.
2: Στην γραμμή αυτή περιγράφεται ο τύπος του στοιχείου-ρίζας (root element) το οποίο στην περίπτωση αυτή είναι το `<time-o-gram>`.
3: Ο δημόσιος προσδιοριστής (public identifier) προσδιορίζει το DTD που θα χρησιμοποιηθεί. Το DTD είναι μια συλλογή παραμέτρων περιγραφής ενός τύπου εγγράφου βάσει της οποίας ο XML Parser θα πρέπει να εξετάσει εάν το έγγραφο μας συμφωνεί με τις παραμέτρους αυτές. Έτσι το έγγραφο XML θα χαρακτηριστεί ως έγκυρο (valid) οπότε και θα αποτελεί ένα στιγμιότυπο (instance) του τύπου αυτού.
4: Στην γραμμή αυτή ορίζεται ο προσδιοριστής συστήματος ο οποίος παρέχει την τοποθεσία του αρχείου DTD. Στην περίπτωση αυτή ο «system identifier» είναι ένα URL.
5: Η αρχή ενός εσωτερικού υποσυνόλου που επιτρέπει ορισμένες ειδικές δηλώσεις
6-7: Μέσα στον χώρο αυτό πραγματοποιούνται δύο δηλώσεις οντοτήτων.
8: Κλείνει ο χώρος ορισμού καθώς και ο πρόλογος με τα σημεία] και >.

Ιδιαίτερη σημασία έχει το δεύτερο μέρος του προλόγου το οποίο ασχολείται με την δήλωση του τύπου του εγγράφου. Στο σημείο αυτό ορίζεται μια αναφορά σε

ένα DTD, με το οποίο δίνουμε εντολή στον XML Parser να συγκρίνει το έγγραφο XML μας με το πρότυπο αυτό – μια διαδικασία που ονομάζεται έλεγχος εγκυρότητας (validity checking). Αν και ο έλεγχος αυτός δεν είναι υποχρεωτικός, για τον Σημασιολογικό Ιστό είναι σημαντικό τα έγγραφα XML εκτός από έγκυρα θα πρέπει να είναι και σωστά μορφοποιημένα (Well Formed).

1.9.4 Τα στοιχεία (elements) ως δομικά στοιχεία της XML

Τα στοιχεία είναι πράγματι τα δομικά στοιχεία ενός εγγράφου XML, καθώς αναπαριστούν τις έννοιες με τις οποίες ασχολείται ένα έγγραφο XML όπως για παράδειγμα το βιβλίο στο παράδειγμά μας. Ένα στοιχείο αποτελείται από την ετικέτα ανοίγματος, το κείμενο περιεχομένου και την ετικέτα κλεισίματος. Για παράδειγμα

```
<author>A.Dennis</author>
```

Υπάρχουν ορισμένου περιορισμοί στην χρήση ονομάτων για τις ετικέτες. Ο πρώτος χαρακτήρας πρέπει να είναι γράμμα, ή ένας από τους χαρακτήρες _και : ενώ δεν θα πρέπει να ξεκινούν με την ονομασία xml. Εάν δεν υπάρχουν περιεχόμενα στο στοιχείο τότε το στοιχείο αποκαλείται κενό και απεικονίζεται με έναν από τους δύο τρόπους.

```
<lecturer></lecturer> ή < lecturer />
```

1.9.5 Χαρακτηριστικά (attributes)

Τα χαρακτηριστικά ενός στοιχείου απεικονίζονται με τον τρόπο αυτό: όνομα= “τιμή” και τοποθετούνται στην ετικέτα ανοίγματος του στοιχείου. Παρακάτω φαίνεται ένα στοιχείο με χαρακτηριστικά:

```
<order    orderNo=“3451000587”    customerID    =“3000147”
deliveryDate=“15/7/2006”>
     <item>
          <articleID>6583</articleID>
          <quantity>5</quantity>
     </item>
     <item>
          <articleID>6541</articleID>
          <quantity>8</quantity>
     </item>
</order>
```

1.9.6 Σωστά μορφοποιημένα έγγραφα (Well Formed Documents)

Ένα έγγραφο XML χαρακτηρίζεται well formed εάν είναι συντακτικά σωστό. Μερικοί συντακτικοί κανόνες είναι οι παρακάτω

1. Υπάρχει μόνο ένα ανώτερο στοιχείο στο έγγραφο το οποίο ονομάζεται στοιχείο ρίζα.
2. Κάθε στοιχείο περιέχει μια ετικέτα ανοίγματος και την αντίστοιχη κλεισίματος
3. Δεν γίνεται να υπάρχει επικάλυψη ετικετών
4. Τα χαρακτηριστικά μέσα σε ένα στοιχείο πρέπει να είναι μοναδικά

1.9.7 Δόμηση Εγγράφων XML

Οι κανόνες μορφοποίησης ενός εγγράφου XML δεν επιβάλλουν κάποιους περιορισμούς σχετικά με την δομή του εγγράφου και εάν αυτή είναι σωστή. Για τον λόγο αυτό χρειάζεται να υπάρχει ένα λεξικό που να ορίζει όλα τα στοιχεία και τα χαρακτηριστικά που θα μπορούν να χρησιμοποιούνται σε ένα έγγραφο XML που θα θέλει να είναι σύμφωνο με κάποιο συγκεκριμένο τύπο. Επιπλέον, χρειάζεται να οριστούν οι τιμές που θα μπορεί να πάρει ένα χαρακτηριστικό, ποια στοιχεία θα μπορούν να εμφανιστούν εμφωλιασμένα μέσα σε ένα άλλο στοιχείο σε τι αριθμό κ.λ.π.

Υπάρχουν δύο τρόποι ορισμού της δομής των εγγράφων XML: τα DTD και το XML Schema. Καθώς το XML Schema αποτελεί και τον δεύτερο layer της αρχιτεκτονικής του Σημασιολογικού Ιστού, και επιπλέον το DTD θεωρείται ως πιο παλιό αυτό με τις πιο περιορισμένες δυνατότητες, θα εξετάσουμε την τεχνική δόμησης XML Schema. Προχωρώντας πέρα από την δυσνόητη και όχι ευέλικτη σύνταξη των DTD, το XML Schema ακολουθεί και αυτό σύνταξη well formed XML (σε αντίθεση με τα DTD) ενώ παρέχει επιπλέον επιπρόσθετο έλεγχο στους τύπους των δεδομένων (στοιχεία και χαρακτηριστικά). Για τον λόγο αυτό έχει γίνει πιο δημοφιλής για αυστηρούς ελέγχους εγκυρότητας δεδομένων.

Ο ρόλος που μπορεί να παίξει το XML Schema εκτός από έλεγχο εγκυρότητας είναι επιπλέον, να χρησιμοποιηθεί ως template για την δημιουργία νέων εγγράφων ως στιγμιότυπα του τύπου εγγράφου που περιγράφει. Σε μια αναλογία με τις βάσεις δεδομένων, το XML Schema ορίζει τις στήλες και τους τύπους δεδομένων ενός πίνακα, ενώ τα έγκυρα στιγμιότυπα-έγγραφα είναι οι εγγραφές του πίνακα. Ένα XML Schema ξεκινάει συνήθως με την παρακάτω γραμμή:

```
<xsd:schema xmlns:xsd= "http¨//w3.org/2000/10/XMLSchema"
version= "1.0">
```

Όπου το XML Schema στην ιστοσελίδα του W3C χρησιμοποιείται σαν θεμέλιο για την δημιουργία νέων XML Schema. Τα πιο σημαντικά στοιχεία των XML Schema είναι ο ορισμός των στοιχείων και των χαρακτηριστικών.

Ο τύπος και η μορφή των στοιχείων ορίζονται ως εξής:

```
<element name= "…" />
```

Επιπλέον χαρακτηρισμοί μπορούν να υπάρχουν, όπως ο τύπος και ο ελάχιστος και ο μέγιστος αριθμός εμφάνισης:

```
<element name = "…" type= "…" minOccurs= "x" maxOccurs=
"y" />
```

π.χ.

```
<element name="FullName" type="CDATA" minOccurs="1"
maxOccurs="1" />
```

Παρομοίως συντάσσεται και ο τύπος του χαρακτηριστικού.

```
<attribute name="…" type="…" use="optional/required"
value="x/default/fixed"/>
```

Σημειώνουμε ότι όπου βλέπουμε type= "x" όπου x μπορούμε να συμπληρώσουμε έναν από τους πιθανούς τύπους δεδομένων που μας προσφέρει το XML Schema (Αριθμητικοί: integer, Short, Byte, Long, Float, Decimal – Αλφαριθμητικοί: string, ID, IDREF, CDATA, Language – Ημερολογιακοί: time, Date, Month, Year). Είναι δυνατή η δημιουργία πολύπλοκων τύπων δεδομένων από τον χρήστη, χρησιμοποιώντας τους απλούς τύπους που αναφέραμε προηγουμένως με την ετικέτα complexType. Χρησιμοποιούμε επίσης τους παρακάτω ορισμούς

sequence: Εάν χρειάζεται να ορίσουμε την σειρά με την οποία θα εμφανίζονται τα στοιχεία μέσα στον νέο τύπο δεδομένων

all: Εάν δεν μας ενδιαφέρει η σειρά αλλά πρέπει να εμφανίζονται όλα τα στοιχεία

choice: Εάν χρειάζεται να επιλεγεί ένα από τα διαθέσιμα στοιχεία.

1.9.8 XML Namespaces

Ένα XML namespace μπορεί απλά να περιγραφεί ως ένα όνομα το οποίο προσδιορίζει μοναδικά ένα στοιχείο XML (tag) και τις αντίστοιχες ιδιότητες του (attributes). Συνήθως ένα namespace συνδέεται με την τοποθεσία ενός εγγράφου XML Schema που περιέχει τους κανόνες εγκυρότητας που διέπει τα στοιχεία και τις ιδιότητες που ανήκουν σε αυτό. Το prefix που ορίζεται σε ένα ορισμό namespace παρέχει μια βοηθητική ονομασία (alias) η οποία θα χρησιμοποιείται σε ένα έγγραφο XML για την παραπομπή στους κανόνες που ορίζονται σε ένα χωριστό έγγραφο XML Schema.

Πρακτικά ένα namespace μπορεί να διαχωρίσει δυο διαφορετικά tags που έχουν το ίδιο όνομα. Για παράδειγμα, εάν δύο έγγραφα XML Schema ορίζουν το καθένα διαφορετικά την ετικέτα `<car>`, τότε ένα έγγραφο XML μπορεί να αναφερθεί σε ένα συγκεκριμένο με την χρήση του namespace prefix. Αν υποθέσουμε ότι έχουμε ορίσει το `xmlns:ford= "http://www.ford.com/cars.ford-car"` τότε με την δήλωση `<ford:car>` θα μπορεί το έγγραφο XML να προσδιορίσει μοναδικά το συγκεκριμένο tag.

1.9.9 Αξίζει ο έλεγχος εγκυρότητας των εγγράφων XML

Ο έλεγχος εγκυρότητας είναι μια κρίσιμη διαδικασία στην προσπάθεια για διαχείριση των δεδομένων ενός οργανισμού. Ομοίως και στην XML, είναι τεράστιας σημασίας καθώς τα έγγραφα XML προορίζονται για μια πλειάδα διαφορετικών εφαρμογών. Οπότε, πιθανά λάθη σε ένα έγγραφο XML μπορούν να μεταδοθούν σε

πολλούς αποδέκτες και το κόστος ενός λάθους μπορεί να πολλαπλασιαστεί (π.χ. λάθη στην παραγγελία να μεταφερθούν στην αλυσίδα τροφοδοσίας). Για αυτό είναι ιδιαίτερα σημαντικό κάθε στιγμιότυπο ενός τύπου εγγράφου XML να ελέγχεται κατά την δημιουργία του ώστε να είναι εγγυημένη η εγκυρότητα των δεδομένων και να είναι δυνατή η ανταλλαγή τους. Αν σκεφτούμε ότι στην κορυφή της πυραμίδας βρίσκεται η εμπιστοσύνη, μπορούμε να φανταστούμε τις επιπτώσεις σε πιθανά λάθη στην βάση της.

ΚΕΦΑΛΑΙΟ 2

Διαδικτυακές Υπηρεσίες (Web Services)

2.1. Γενικά

Η υπόσχεση των διαδικτυακών υπηρεσιών και γενικά των αρχιτεκτονικών που βασίζονται σε υπηρεσίες είναι να δημιουργήσουν ένα κατανεμημένο περιβάλλον στο οποίο οι εφαρμογές και τα συστατικά τους, συνεργάζονται με διάφανο τρόπο, μέσα στον ίδιο οργανισμό ή και μεταξύ διαφορετικών οργανισμών, ανεξάρτητα από την πλατφόρμα στην οποία αναπτύχθηκαν. Μια διαδικτυακή υπηρεσία είναι στην ουσία ένα κομμάτι επιχειρησιακής λογικής το οποίο βρίσκεται σε κάποιο σημείο του Internet, και είναι διαθέσιμο μέσω των πρωτοκόλλων του Internet όπως το HTTP ή το SMTP. Η χρήση μιας διαδικτυακής υπηρεσίας μπορεί να είναι είτε απλή όπως π.χ. η είσοδος ενός χρήστη σε ένα δικτυακό τοπίο, είτε πολύπλοκη όπως π.χ. μια διαπραγμάτευση μεταξύ οργανισμών-εταιριών.

2.2 Τι είναι τα Web Services

Έτοιμες υπηρεσίες στο Internet οι οποίες μπορούν να χρησιμοποιηθούν από τις ΜΜΕ. Τα web services είναι μια καινοτομική αρχιτεκτονική με την οποία παρέχεται η δυνατότητα δημιουργίας και χρήσης ηλεκτρονικών υπηρεσιών στο διαδίκτυο με απλό και οικονομικό τρόπο. Μέχρι πρόσφατα η δημιουργία και η παροχή υπηρεσιών από επιχειρήσεις στο Internet γίνονταν με ακαθόριστο τρόπο ο οποίος διέφερε από επιχείρηση σε επιχείρηση. Έτσι, ενώ υπήρχε ένα αρκετά μεγάλο σύνολο από παρεχόμενες υπηρεσίες στο Internet, για να μπορούσε κάποιος να τις χρησιμοποιήσει θα έπρεπε για κάθε μία υπηρεσία να μελετήσει τον τρόπο με τον οποίο θα την καλέσει, να ελέγξει αν χρησιμοποιούν το ίδιο πρωτόκολλο επικοινωνίας (TCP/IP, Http, κλπ) και γενικά να προσαρμόσει όλο το σύστημά του έτσι ώστε να γίνει συμβατό με αυτό του παροχέα της υπηρεσίας.

Για παράδειγμα ας υποθέσουμε ότι κάποια επιχείρηση ενδιαφερόταν να χρησιμοποιήσει μία υποτιθέμενη υπηρεσία που παρείχε το Εθνικό Κέντρο Βιβλίου και η οποία παρουσίαζε όλες τις συνοδευτικές πληροφορίες (τίτλο, εκδοτικό οίκο, τιμή κλπ) για κάποιο βιβλίο δοθέντος του κωδικού του (ISBN). Σε αυτή την περίπτωση ο προγραμματιστής της επιχείρησης θα έπρεπε στην ουσία να δημιουργήσει ένα σύστημα συμβατό με αυτό του Εθνικού Κέντρου Βιβλίου και ως προς το πρωτόκολλο επικοινωνίας αλλά και ως προς τον τρόπο κλήσης των ερωτημάτων και κατόπιν να το προσαρμόσει στις ανάγκες του συστήματος της επιχείρησης. Πολλές φορές αυτό ήταν πολύ δύσκολο – αν όχι ακατόρθωτο – και ακόμα περισσότερες φορές οι επιχειρήσεις σχεδίαζαν τα συστήματά τους έτσι ώστε να αποφεύγουν τέτοιου είδους συνεργασίες με ξένες πηγές για λόγους

πολυπλοκότητας και γενικότερα για λόγους κόστους.Τα πράγματα όμως τα τελευταία τρία χρόνια φαίνεται να παίρνουν διαφορετική τροπή αφού πλέον σχεδόν όλες οι επιχειρήσεις που δημιουργούν υπηρεσίες στο Internet βασίζονται σε μία κοινή αρχιτεκτονική ανάπτυξης, δημοσίευσης και εκμετάλλευσης των υπηρεσιών τους, όπως αυτή καθορίζεται από το W3C και που ορίζεται ως η αρχιτεκτονική των web services. Μια διαδικτυακή υπηρεσία αποτελείται από πολλές συσχετιζόμενες τεχνολογίες που τοποθετούνται σε διαφορετικό επίπεδο. Ξεκινώντας από κάτω προς τα πάνω, αναφέρουμε τα πρότυπα που χρησιμοποιούνται και τα αναλύουμε εκτενέστερα μαζί με τα θέματα ασφάλειας και διαχείρισης στις επόμενες ενότητες. Αρχικά, απαιτείται ένα πρωτόκολλο για μεταφορά πληροφοριών μέσω δικτύου, όπως το http (HyperText Transfer Protocol), το SMTP (Simple Mail Transport Protocol) ή το FTP (File Transfer Protocol). Κάθε κλήση και απόκριση της υπηρεσίας θα πρέπει να «συσκευαστεί» σε ένα μήνυμα SOAP (Simple Object Access Protocol), το οποίο μπορεί να υφίσταται επεξεργασία από κάποιες επεκτάσεις SOAP (SOAP Extensions) πριν αποσταλεί από τον αιτούντα υπηρεσία (request agent) και παραδοθεί στον πάροχο υπηρεσίας (provider agent) και αντίστροφα. Τα μηνύματα που ανταλλάσσονται και ο τρόπος που γίνεται η ανταλλαγή περιγράφονται λεπτομερώς στο αρχείο WSDL (Web Services Description Language). Επόμενο βήμα είναι η ανακάλυψη των υπηρεσιών, για την οποία υπάρχουν τρεις προσεγγίσεις : η ύπαρξη ενός καταλόγου καταγραφής και δημοσίευσης των υπηρεσιών με τη μορφή υπηρεσίας καταγραφής (registry), όπως το UDDI (Universal Description, Definition and Integration) και το DISCO (Microsoft Discovery), που αποτελεί την πιο διαδεδομένη λύση, η ύπαρξη ιστοσελίδων παραπομπής σε υπηρεσίες, τύπου http://www.amazon.com και πρότυπα SOAP, WSDL, UDDI και DISCO έχουν γραφεί σε γλώσσα επισήμανσης XML (eXtensible Markup Language) και ενδεχομένως υπακούουν σε κάποιο έγγραφο DTD (Document Type Definition) ή XML Schema. Η τεχνολογία των Web Services αποτελεί την πιο εξελίξιμη και νεωτεριστική τεχνολογία του μέλλοντος, παγκοσμίως. Ο προσδιορισμός της αποτελεί προϊόν συνεργασίας κολοσσών της πληροφορικής, όπως η Microsoft, η Sun Microsystems και η IBM και αποτελεί την ιδανική λύση για διεπιχειρησιακές εφαρμογές και συνεργασίες (Business-to-Business, B2B).

Τα Web Services στηρίζονται σε ευρέως διαδεδομένα πρότυπα της βιομηχανίας της πληροφορικής και δυνητικά συνιστούν τον βέλτιστο τρόπο για την διασύνδεση των επιχειρησιακών εφαρμογών τόσο μέσα στο εταιρικό δίκτυο (ενδοδίκτυο), αλλά και στο διαδίκτυο. Με τα Web Services μπορούν να διασυνδεθούν και να επικοινωνήσουν εσωτερικές εφαρμογές όπως π.χ. το Λογιστικό πακέτο με το πακέτο Διαχείρισης Πελατειακών Σχέσεων μέσω του ενδοδικτύου. Επίσης, μπορούν να χρησιμοποιηθούν για την διασύνδεση, την συνεργασία και την ανταλλαγή πληροφοριών μεταξύ ενδο-επιχειρησιακών εφαρμογών και εφαρμογών τρίτων, όπως π.χ. των συνεργατών και προμηθευτών σας μέσω του διαδικτύου. Στηρίζονται στην XML και άλλες καινοτόμες τεχνολογίες και προσφέρουν ένα απλούστερο τρόπο για την επίτευξη κατανεμημένων εφαρμογών (distributed computing), διευκολύνοντας τα διασυνδεδεμένα συστήματα να ανταλλάξουν πληροφορίες και να συνδιαλλαγούν μεταξύ τους. Η Forward e-business αξιοποιώντας την εφαρμοσμένη εμπειρία της σε Java και XML εισέρχεται δυναμικά στον χώρο των Web Services, προσφέροντάς οικονομικές, ποιοτικές και ευέλικτες λύσεις, που θα οδηγήσουν οποιαδήποτε επιχείρηση στον δρόμο των παγκόσμιων εξελίξεων.

2.3 Χαρακτηριστικά-Πλεονεκτήματα των Web Services

Η αρχιτεκτονική των web services παρέχει αρκετά πλεονεκτήματα μερικά από τα οποία αναφέρονται παρακάτω:

Διαλειτουργικότητα:

Ένα web service παρέχει ανεξαρτησία τόσο από λειτουργικό σύστημα όσο και από το hardware. Οποιοδήποτε πρόγραμμα που συμβαδίζει με αυτή τη τεχνολογία μπορεί πολύ εύκολα να προσπελάσει μία τέτοια υπηρεσία.

Ενσωμάτωση:

Σε ένα υπάρχον λογισμικό σύστημα που λειτουργεί μέσα στο Internet η δημιουργία ενός web service δεν απαιτεί αλλαγές στον μηχανισμό του συστήματος.

Διαθεσιμότητα και δημοσίευση:

Οι πληροφορίες για τα web services δημοσιεύονται οπότε η εύρεση και η χρήση τους μπορεί να είναι ταχύτατες.

Επέκταση:

Ένα έτοιμο web service είναι δυνατό να ανανεωθεί με εύκολο τρόπο παρέχοντας έτσι επιπρόσθετες υπηρεσίες στους χρήστες του.

Μικρό κόστος δημιουργίας και χρήσης:

Εφόσον σε ένα λογισμικό σύστημα υπάρχει ήδη κάποια διαδικασία που χρειάζεται να επεκταθεί σε on-line υπηρεσία, η δημιουργία του web service κοστίζει ελάχιστα. Επίσης το κόστος ενσωμάτωσης ενός web service σε κάποιο website ή σε δικτυακή εφαρμογή είναι πάρα πολύ μικρό. Ακόμα και στις περιπτώσεις που η χρήση κάποιου web service γίνεται με ενοικίαση σίγουρα το συνολικό κόστος της χρήσης είναι αρκετά πιο μικρό από το κόστος δημιουργίας της υπηρεσίας αυτής.

Χρήση λογισμικών συστημάτων:

Όλα τα λογισμικά συστήματα και ειδικότερα τα websites που χρησιμοποιούν έτοιμες υπηρεσίες γίνονται πιο λειτουργικά και πιο φιλικά αφού παρέχουν περισσότερες υπηρεσίες στους χρήστες.

Η νέα γενιά διαδικτυακών υπηρεσιών και οι τεχνολογίες τους βασίζονται στην XML και υποστηρίζονται σε παγκόσμιο επίπεδο από τις περισσότερες μεγάλες τεχνολογικές εταιρίες. Όπως είδαμε η XML παρέχει ένα τρόπο αναπαράστασης δεδομένων ανεξαρτήτως πλατφόρμας και γλώσσας προγραμματισμού. Στον επιχειρησιακό τομέα, οι ευκαιρίες για παγκόσμια συνεργασία και ενσωμάτωση λογισμικού είναι τρομακτικά μεγάλες. Μια διαδικτυακή υπηρεσία (web service) έχει ορισμένα ξεχωριστά χαρακτηριστικά.

Βασίζεται σε XML

Χρησιμοποιώντας τη XML για αναπαράσταση δεδομένων σε όλα τα πρωτόκολλα και τις τεχνολογίες διαδικτυακών υπηρεσιών, οι τεχνολογίες μπορούν να συνεργάζονται μεταξύ τους σε βασικό επίπεδο. Εάν χρησιμοποιηθεί για την μεταφορά δεδομένων, η XML εξαφανίζει κάθε στενό δεσμό που μπορεί να έχει ένα πρωτόκολλο με το δίκτυο ή το λειτουργικό σύστημα στο οποίο βρίσκεται.

Έχει χαλαρούς δεσμούς

Όποιος χρησιμοποιεί μια διαδικτυακή υπηρεσία δεν είναι άμεσα συνδεδεμένος με αυτήν –η υπηρεσία αυτή μπορεί να αλλάζει με το χρόνο χωρίς να υποβαθμίζεται η δυνατότητα του πελάτη της να αλληλεπιδρά με αυτή. Σε αντίθετη περίπτωση εάν ο server και ο client έχουν στενούς δεσμούς τότε πιθανή αλλαγή στο ένα άκρο θα απαιτούσε και αντίστοιχη αλλαγή και στο άλλο. Οι χαλαροί δεσμοί επιτρέπουν εύκολη ολοκλήρωση και επικοινωνία μεταξύ διαφορετικών συστημάτων.

Σύγχρονη και ασύγχρονη λειτουργία

Η σύγχρονη λειτουργία αναφέρεται στην δυνατότητα σύνδεσης του πελάτη με την εκτέλεση της υπηρεσίας, δηλαδή ο πελάτης περιμένει την υπηρεσία να ολοκληρώσει την λειτουργία της πριν συνεχίσει την εργασία του. Η ασύγχρονη λειτουργία επιτρέπει στον πελάτη να καλέσει την υπηρεσία και μετά να εκτελέσει άλλες λειτουργίες. Οι ασύγχρονοι πελάτες λαμβάνουν το αποτέλεσμα που ζήτησαν προηγουμένως μόλις ολοκληρωθεί η εκτέλεση της υπηρεσίας.

Υποστήριξη απομακρυσμένων κλήσεων διαδικασιών (Remote Procedure Call)

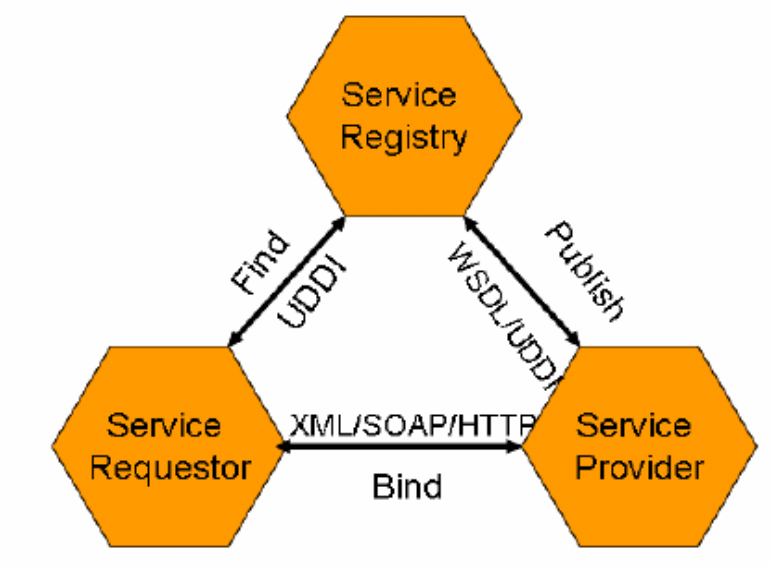
Οι διαδικτυακές υπηρεσίες επιτρέπουν στους πελάτες να εκτελούν διαδικασίες, συναρτήσεις και μεθόδους σε απομακρυσμένα αντικείμενα χρησιμοποιώντας ένα πρωτόκολλο που βασίζεται στην γλώσσα XML.

Υποστήριξη ανταλλαγής πληροφοριών

Ένα από τα βασικότερα πλεονεκτήματα της XML είναι ο απλός τρόπος αναπαράστασης όχι μόνο δεδομένων αλλά και πολύπλοκων εγγράφων. Τα έγγραφα αυτά μπορούν να είναι απλά, όπως π.χ. όταν αναπαριστώνται διευθύνσεις αλλά και αρκετά πιο πολύπλοκα όπως π.χ. ένα ολόκληρο βιβλίο. Οι διαδικτυακές υπηρεσίες υποστηρίζουν την διάφανη ανταλλαγή εγγράφων ώστε να υποστηρίξουν επιχειρησιακή ολοκλήρωση.

2.4 Υπηρεσιοκεντρική Αρχιτεκτονική (Service Oriented Architecture – SOA)

Το μοντέλο που χρησιμοποιείται στις μέρες μας για την περιγραφή των «Υπηρεσιών Διαδικτύου» βασίζεται κυρίως στο κλασικό επιχειρησιακό μοντέλο, το οποίο έχουν υιοθετήσει οι διάφοροι ιδιωτικοί οργανισμοί και απεικονίζεται στο παρακάτω σχήμα.



Σχήμα 3) Αρχιτεκτονική γύρω από την υπηρεσία

Από το παραπάνω σχήμα μπορούμε να διακρίνουμε τις οντότητες που απαρτίζουν την τρέχουσα αρχιτεκτονική των υπηρεσιών διαδικτύου. Αυτές είναι η οντότητα που ζητάει την υπηρεσία (Service Requestor), η οντότητα που παρέχει την υπηρεσία (Service Provider) και τέλος η οντότητα του καταλόγου υπηρεσιών (Service Registry). Με περισσότερη λεπτομέρεια, κάθε μία οντότητα έχει τους ακόλουθους ρόλους: Η οντότητα, η οποία ζητάει την υπηρεσία (Service Requestor), είναι στην ουσία ο «αιτούμενος» (Client) της υπηρεσίας και πυροδοτεί την έναρξη της όλης διαδικασίας. Αρχικά επιφορτίζεται με το έργο της αναζήτησης της κατάλληλης περιγραφής μιας υπηρεσίας τις καταχωρήσεις της υπηρεσίας καταλόγου. Στη συνέχεια, και ενώ έχει εντοπίσει τον επιθυμητό παροχέα, εγκαθιδρύει σύνδεση με αυτόν, καλεί την υπηρεσία και λαμβάνει τα αποτελέσματα.

Η οντότητα που παρέχει την υπηρεσία (Service Provider), δημοσιοποιεί την περιγραφή της υπηρεσίας ιστού στην οντότητα του καταλόγου υπηρεσιών (Service Registry) και ουσιαστικά παρέχει την υπηρεσία στο διαδίκτυο. Επιπλέον, της ανατίθεται ο ρόλος της λήψης των μηνυμάτων κλήσης για την υπηρεσία από έναν ή περισσότερους αιτούμενους και της παροχής των απαραίτητων αποτελεσμάτων.

Τέλος, η οντότητα του καταλόγου υπηρεσιών (Service Registry), περιέχει καταχωρήσεις με τις περιγραφές των ήδη δημοσιοποιηθέντων υπηρεσιών διαδικτύου. Για τις υπηρεσίες αυτές παρέχονται τρόποι αναζήτησης ανάμεσα στις διάφορες περιγραφές.

Οι λειτουργίες που παρέχονται μέσω αυτής της αρχιτεκτονικής είναι οι ακόλουθες:

Η καταχώρηση των απαραίτητων πληροφοριών για μια υπηρεσία από τη μεριά του παρόχου (publishing), η οποία επιτυγχάνεται μέσω της χρήσης της γλώσσας περιγραφής WSDL.

Η αναζήτηση και η εύρεση στους καταλόγους της κατάλληλης περιγραφής μιας «Υπηρεσίας Διαδικτύου» (finding). Η επικρατούσα τεχνολογία καταλόγου ονομάζεται UDDI.

Τέλος, έχουμε την εγκαθίδρυση σύνδεσης μεταξύ αιτούμενου και παροχέα (binding), κλήση της κατάλληλης υπηρεσίας και αποστολή των αποτελεσμάτων. Όλα τα παραπάνω επιτυγχάνονται με την ανταλλαγή μηνυμάτων SOAP μεταξύ των τριών εμπλεκόμενων οντοτήτων, η οποία βασίζεται στη γλώσσα μεταδεδομένων XML. Το κλειδί της παραπάνω αρχιτεκτονικής είναι η περιγραφή της «Υπηρεσίας

Διαδικτύου». Η περιγραφή είναι εκείνη που δημοσιοποιείται από τον παροχέα της υπηρεσίας στον κατάλογο υπηρεσιών. Η περιγραφή είναι το αποτέλεσμα της αναζήτησης του αιτούμενου στον κατάλογο υπηρεσιών. Η περιγραφή είναι εκείνη που λέει στον αιτούμενο όσα ακριβώς πρέπει να γνωρίζει προκειμένου να κάνει κλήση της υπηρεσίας που τον ενδιαφέρει. Τέλος, η περιγραφή της υπηρεσίας μπορεί να περιέχει πληροφορία για το τι είδους είναι το αποτέλεσμα που αναμένεται να επιστραφεί ύστερα από την κλήση της υπηρεσίας. Με την παραπάνω αρχιτεκτονική ήδη υπάρχουσες εφαρμογές μπορούν εύκολα να μετασχηματισθούν σε υπηρεσίες που να εξυπηρετούν άλλες υπάρχουσες – ή και νέες – εφαρμογές. Πολλές υπηρεσίες μπορούν να συνδυαστούν και να συνεργασθούν για την παραγωγή ενός αποτελέσματος με τρόπο διαφανή προς τον τελικό χρήστη, στον οποίο δίνεται η αίσθηση ότι κλήθηκε μία και μόνο υπηρεσία. Επιπρόσθετα, οργανισμοί και επιχειρήσεις μπορούν ευκολότερα να κατασκευάσουν λογισμικό, που να αλληλεπιδρά με επιχειρησιακές διαδικασίες και να ανταποκρίνεται γρήγορα στις αλλαγές του επιχειρησιακού περιβάλλοντος.

2.5 Τεχνολογίες που χρησιμοποιούν τα Web Services

Οι τεχνολογίες που χρησιμοποιούν τα Web Services συμπεριλαμβάνουν: XML, που περιλαμβάνει βασική XML, XML schemas και XML parsers.

SOAP (Simple Object Access Protocol), που αποτελεί ένα πρωτόκολλο επικοινωνίας εφαρμογών βασισμένο σε XML.

WSDL (Web Services Description Languages), που είναι ένα XML schema για περιγραφή των μηνυμάτων, λειτουργιών και αντιστοιχήσεις πρωτοκόλλων των υπηρεσιών διαδικτύου.

UDDI (Universal Description Discovery and Integration), που είναι ο χώρος αποθήκευσης για καταχώρηση και αναζήτηση περιγραφών υπηρεσιών διαδικτύου.

Η ανάγκη για χρήση των υπηρεσιών διαδικτύου ανακύπτει από την απαίτηση των χρηστών να μπορούν να έχουν εύκολη πρόσβαση σε πληροφορία που μπορεί να δημοσιευτεί σε οποιοδήποτε μέρος του διαδικτύου. Η υπάρχουσα τεχνολογική υποδομή του παγκοσμίου ιστού αν και έχει διευκολύνει τον κόσμο των επιχειρήσεων έχει μερικούς περιορισμούς. Δεν καλύπτει την ανάγκη αυτόματης αλληλεπίδρασης μεταξύ εφαρμογών. Σήμερα οι εφαρμογές πρέπει να εκτελεστούν «με το χέρι» χρησιμοποιώντας έναν φυλλομετρητή. Επίσης χρειάζεται ένας καλύτερος μηχανισμός για την αναζήτηση πληροφορίας στο διαδίκτυο από αυτόν που χρησιμοποιείται σήμερα και βασίζεται στην «σάρωση» HTML σελίδων προκειμένου να βρεθεί το ζητούμενο αλφαριθμητικό ή ομάδα αλφαριθμητικών. Οι υπηρεσίες διαδικτύου έρχονται να καλύψουν τέτοιου είδους κενά εκμεταλλευόμενες την κατανεμημένη μορφή του διαδικτύου και παρέχοντας ένα νέο μοντέλο ανταλλαγής της πληροφορίας.

2.6 HTTP (HyperText Transfer Protocol)

Το πρωτόκολλο Μεταφοράς Υπερ-κειμένου HTTP (HyperText Transfer Protocol) είναι το στάνταρ πρωτόκολλο μεταφοράς στον Παγκόσμιο Ιστό. Κάθε ανταλλαγή πληροφορίας συνίσταται από μια αίτηση ASCII, ακολουθούμενη από μία απόκριση RFC 822 τύπου MIME, η οποία αποτελεί το de facto πρότυπο μορφής μηνύματος για το ηλεκτρονικό ταχυδρομείο. Παρά το γεγονός ότι είναι σύνηθες να

χρησιμοποιείται το πρωτόκολλο Ελέγχου Μεταφοράς TCP (Transmission Control Protocol) για τη σύνδεση μεταφοράς, δεν απαιτείται επίσημα από το πρότυπο.

2.7. Πρωτόκολλο XML (eXtensible Markup Language)

Η XML (eXtensible Markup Language) αναπτύχθηκε από το W3C's XML Working Group το 1996 και συνδυάζει την ισχύ και την επεκτασιμότητα της SGML (Standard Generalized Markup Language), από την οποία προέρχεται, με την απλότητα που απαιτεί η κοινότητα του Διαδικτύου. Είναι μια μεταφερτή, ευρέως υποστηριζόμενη, ανοικτή τεχνολογία για την περιγραφή δεδομένων. Η XML περιγράφει μια κατηγορία αντικειμένων δεδομένων που ονομάζονται XML έγγραφα και εν μέρει τη συμπεριφορά των προγραμμάτων υπολογιστών που τα επεξεργάζονται. Τα έγγραφα XML αποτελούνται από μονάδες αποθήκευσης, τις οντότητες (entities), οι οποίες περιέχουν αναλυμένα λεκτικά (parsed) ή μη αναλυμένα λεκτικά (unparsed) δεδομένα. Τα αναλυμένα λεκτικά δεδομένα (parsed) συνίστανται από χαρακτήρες, ορισμένοι από τους οποίους αποτελούν δεδομένα χαρακτήρων και κάποιοι άλλοι δημιουργούν markup, δηλαδή κωδικοποιούν μια περιγραφή της διάταξης και της λογικής δομής του εγγράφου. Για τα μη αναλυμένα λεκτικά δεδομένα (unparsed) δεν είναι γνωστό εάν είναι κείμενο ή όχι, αλλά ακόμη κι αν είναι κείμενο, ενδέχεται να μην είναι XML, οπότε αγνοούνται κατά τη διαδικασία της ανάλυσης λεκτικών δεδομένων (parsing), στην οποία θα αναφερθούμε εκτενέστερα στη συνέχεια, και διοχετεύονται όπου προβλέπει η εφαρμογή.

Η XML αποτελεί σήμερα το πρότυπο για την αποθήκευση δεδομένων που ανταλλάσσονται μεταξύ των εφαρμογών χάρη στα ακόλουθα χαρακτηριστικά που παρουσιάζει :

• Υποστηρίζει ανεξαρτησία από τα δεδομένα και διαχωρίζει τα περιεχόμενα από τον τρόπο εμφάνισής τους και τον χειρισμό τους, οπότε διευκολύνεται η λεκτική ανάλυσή τους (parsing).

• Διατίθενται έτοιμοι τρόποι σύνδεσης των κειμένων XML με τα πλέον σύγχρονα προγραμματιστικά περιβάλλοντα, όπως το Document Object Model (DOM) και το Simple API for XML (SAX).

• Είναι επεκτάσιμη και ανεξάρτητη από πλατφόρμες, γεγονός που την καθιστά απρόσβλητη σε τεχνολογικές αλλαγές.

• Τα έγγραφα XML είναι αναγνώσιμα από ανθρώπους και μηχανές και παρότι δεν προορίζονται για ανάγνωση προσφέρουν αυτή τη δυνατότητα στο χρήστη εάν κριθεί αναγκαίο.

• Είναι πλήρως συμβατή με Unicode, οπότε μπορεί να χειριστεί την πληροφορία που έχει γραφεί σε οποιαδήποτε ανθρώπινη γλώσσα. Παράλληλα, υποστηρίζει διεθνείς και τοπικές προσαρμογές.

• Χρησιμοποιώντας την XML, οι δημιουργοί των εγγράφων μπορούν να αναπαραστήσουν πολύπλοκες δομές δεδομένων, όπως λίστες, εγγραφές και δένδρα και να περιγράψουν οποιονδήποτε τύπο δεδομένων, συμπεριλαμβάνοντας μαθηματικούς τύπους, οδηγίες διαμόρφωσης λογισμικού, μουσική, αποδείξεις και οικονομικές αναφορές.

Ενδεικτικά αναφέρουμε ότι η XML αποτελεί τη βάση του RDF (Resource Description Framework), του Σημασιολογικού Ιστού (Semantic Web), της WML (Wireless Markup Language), της MathML και της GML (Geography Markup Language).

```
<?xml version="1.0" ?>
<!--
A first program using XML
-->
<holidays xmlns:altgr="http://www.alternativegreece.gr"
xmlns:ginn="http://www.guestinn.com">
<cruises>
<voyage type="on a large ship" price="normal"
destination="scheduled"/>
<voyage type="on a private yacht" price="high"
destination="anywhere"/>
</cruises>
<altgr:alternative_tourism>
<altgr type="rafting" />
<altgr type="ski" />
<participation price="20-25 euro">There are special
prices for
students</participation>
<ages type="young" />
</altgr:alternative_tourism>
<ginn:alternative_tourism>
<ginn type="residence in traditional villages" />
<participation price="50-60 euro" />
<ages type="all ages" />
</ginn:alternative_tourism>
</holidays>
```

Το έγγραφο αρχίζει με μια προαιρετική δήλωση XML, που το προσδιορίζει ως έγγραφο XML. Η παράμετρος πληροφοριών version καθορίζει την έκδοση της XML που χρησιμοποιείται στο έγγραφο. Τα σχόλια XML αρχίζουν με <!- - και τελειώνουν με - -> και μπορούν να τοποθετηθούν σχεδόν οπουδήποτε σε ένα έγγραφο XML.

Στην XML, τα δεδομένα επισημαίνονται χρησιμοποιώντας ετικέτες (tags), που είναι ονόματα που περιβάλλονται από τα σύμβολα <>. Οι ετικέτες χρησιμοποιούνται σε ζεύγη για να οροθετήσουν τους χαρακτήρες δεδομένων (π.χ. There are special prices for students). Με τις ετικέτες επιτυγχάνουμε το λεγόμενο “markup” και μια ετικέτα που αρχίζει “markup” (δηλαδή δεδομένα XML) διαφέρει από μια ετικέτα που τερματίζει την επισήμανση κατά την κεκλιμένη / που ακολουθεί τον χαρακτήρα <. Οι ετικέτες προσδιορίζονται ελεύθερα από το χρήστη, ο οποίος μπορεί να τους δίνει όποια σημασία αυτός ορίσει. Ένα έγγραφο XML περιλαμβάνει ένα στοιχείο, το στοιχείο ρίζας, που περιλαμβάνει κάθε άλλο στοιχείο και είναι το πρώτο μετά τη δήλωση XML. Τα στοιχεία εντίθενται το ένα μέσα στο άλλο για να σχηματίσουν ιεραρχίες – με το στοιχείο ρίζας στην κορυφή της ιεραρχίας. Με την ιδιότητα αυτή της ενθυλάκωσης (nesting) παρέχεται η δυνατότητα να κατασκευαστούν στοιχεία με πολύπλοκη εσωτερική δομή, για παράδειγμα το στοιχείο cruises.

Κάθε στοιχείο (element) μπορεί να περιέχει κάποιες ιδιότητες (attributes), που

είναι ζεύγη ονόματος – τιμής στις ετικέτες αρχής, όπως το στοιχείο voyage που ένα από τα χαρακτηριστικά του είναι το type με τιμή χαρακτηριστικού "on a large ship" που δηλώνει ότι αυτό το στοιχείο voyage επισημαίνει πληροφορίες για τα ταξίδια με μεγάλα πλοία.

Με τους χώρους ονομάτων (namespaces) της XML παρέχεται ένα μέσο μοναδικού προσδιορισμού των στοιχείων XML και αποφεύγονται οι διενέξεις ονομάτων εάν λόγου χάρη ένα στοιχείο υπάρχει σε δύο διαφορετικά κείμενα με το ίδιο όνομα και αυτά τα κείμενα πρόκειται να συγχωνευτούν. Τα στοιχεία διαφοροποιούνται μέσω των προθεμάτων χώρων ονομάτων, που προσδιορίζουν τον χώρο ονομάτων στον οποίο ανήκει ένα στοιχείο, και μιας σύμβασης, που συνήθως είναι ένας καθολικός δείκτης πόρων (Universal Resource Locator – URL) και προσδιορίζει πού έχει οριστεί ο χώρος ονομάτων. Τα URLs δεν χρειάζεται να αναφέρονται σε πραγματικές ιστοσελίδες ή να είναι διαμορφωμένα κατάλληλα, απλώς αναπαριστούν μια ακολουθία χαρακτήρων για να διαφοροποιήσουν τα ονόματα. Για παράδειγμα, το `<altgr:alternative_tourism>` προσδιορίζει το στοιχείο alternative_tourism με το πρόθεμα χώρου ονομάτων altgr και URL το http://www.alternativegreece.gr .

Ορθότητα XML εγγράφου

Για να είναι ένα έγγραφο XML σωστό θα πρέπει να είναι συγχρόνως :

• Καλοσχηματισμένο (well-formed), δηλαδή να υπακούει σε όλους τους συντακτικούς κανόνες της XML. Εάν ένα έγγραφο δεν είναι συντακτικά σωστό, τότε δεν θεωρείται έγγραφο XML, η ανάλυση (parsing) σταματά και ο αναλυτής (parser) δίνει ένα σφάλμα.

• Έγκυρο (valid), δηλαδή να ικανοποιεί ένα σύνολο κανόνων που έχει ορίσει ο χρήστης και περιλαμβάνεται σε προαιρετικά έγγραφα XML, όπως τους Ορισμούς Τύπων Εγγράφων (Document Type Definitions – DTDs) και τα Σχήματα XML (XML Schemas).

2.8 DTD

Η XML χρησιμοποιεί ένα μοντέλο για την περιγραφή των εγγράφων XML. Ένα τέτοιο μοντέλο ονομάζεται Document Type Definition (DTD). Η XML το χρησιμοποιεί για να περιγράψει τα δεδομένα. Το DTD είναι ένας μηχανισμός που κληρονομήθηκε από την γλώσσα SGML και χρησιμοποιείται για την περιγραφή κάθε αντικειμένου σε ένα έγγραφο XML, όπως είναι τα στοιχεία και τα χαρακτηριστικά. Αποτελεί ένα σύνολο κανόνων που αφορούν στα tags της γλώσσας XML. Πιο συγκεκριμένα ορίζει κανόνες για το ποια είναι τα επιτρεπτά ονόματα στοιχείων και τα επιτρεπτά υποστοιχεία (subelements) και χαρακτηριστικά (attributes) για κάθε συγκεκριμένο στοιχείο. Ακολουθεί ένα έγγραφο DTD. Να αναφέρουμε ότι στα έγγραφα με μορφοποίηση XML τα σχόλια περιλαμβάνονται μεταξύ των <!—και -->.

```
<!Elements name (#PCDATA | fname | lname )*>
<!Elements fname (#PCDATA )> <!—Η δεσμευμένη λέξη PCDATA σημαίνει parsed character data και σημαίνει ότι τα συγκεκριμένο στοιχείο μπορεί να περιέχει κείμενο. Χρησιμοποιείτε συνήθως για τα στοιχεία «φύλλα».-- >
<!Elements lname (#PCDATA )>
<!Elements adddress (street, region; , country)>
<!ATTLIST adddress preferred (true | false) ‘false’>
<!Elements street (#PCDATA)> <!—Το στοιχείο street είναι ένα στοιχείο <<φύλλο>>
διότι δεν έχει παιδιά -->
<!Elements region (#PCDATA)>
<!Elements country (#PCDATA)>
<!Elements tel (#PCDATA)>
<!ATTLIST tel preferred (true | false) ‘false’>
<!Elements email (#PCDATA)>
<!ATTLIST email href CDATA #REQUIRED
preferred (true | false) ‘false’>
```

Το παραπάνω DTD έγγραφο μπορεί να συνοδεύει το ακόλουθο έγγραφο XML:

```
<?xml version=”1.0”?>
<!DOCTYPE address-book SYSTEM “address-book. dtd”>
< address-book >
<entry>
<name>Γιάννης Παπαδόπουλος </name>
<address>
<street>Σταμπουλή 82</street>
<region>Φλώρινα</region>
<country>Ελλάδα</country>
</address>
<tel preferred=”true”>2385023364</tel>
<email href=jimlias@yahoo.gr/>
</entry>
</address-book>
```

Να κάνουμε τις εξής παρατηρήσεις σχετικά με τη δομή και τη σύνταξη των εγγράφων DTD:

• Όλα τα ονόματα των αντικειμένων ξεκινούν με γράμμα και δεν επιτρέπεται ο χαρακτήρας κενού διαστήματος. Στην θέση του χρησιμοποιείται το ενωτικό.

• Η XML υποστηρίζει πλήρως το πρότυπο κωδικοποίησης χαρακτήρων Unicode, συνεπώς μπορούμε ελεύθερα να χρησιμοποιήσουμε ελληνικούς χαρακτήρες σαν έγκυρους χαρακτήρες ονομάτων για τα στοιχεία του εγγράφου μας.

• Οι χαρακτήρες *, ?, και + ονομάζονται χαρακτήρες ύπαρξης και υποδεικνύουν αν και πώς τα στοιχεία στην λίστα των παιδιών επαναλαμβάνονται. Αν δεν υπάρχει δείκτης ύπαρξης σε ένα στοιχείο αυτό σημαίνει ότι το συγκεκριμένο στοιχείο εμφανίζεται μια φορά. Στοιχείο με τον χαρακτήρα + πρέπει να εμφανίζεται μια τουλάχιστον φορά. Με τον χαρακτήρα * καμία ή περισσότερες φορές. Με τον χαρακτήρα ? σημαίνει ότι το στοιχείο μπορεί να εμφανιστεί το πολύ μια φορά.

• Οι χαρακτήρες , και | ονομάζονται συζευκτήρες. Βοηθούν στον διαχωρισμό των παιδιών ενός στοιχείου μέσα στο μοντέλο περιεχομένου. Ο χαρακτήρας , σημαίνει ότι τα δύο στοιχεία αριστερά και δεξιά του κόμματος πρέπει να

εμφανίζονται με την ίδια σειρά μέσα στο έγγραφο. Ενώ ο χαρακτήρας | σημαίνει ότι μόνο το ένα από τα στοιχεία πρέπει να εμφανίζεται στο έγγραφο.
Ένα έγγραφο XML μπορεί να ανήκει σε μια από τις εξής δύο βασικές κατηγορίες:
• Έγκυρο (valid) , δηλαδή να διαθέτει ένα DTD που το συνοδεύει
• Σωστά μορφοποιημένο (Well-formed) που δεν διαθέτει κάποιο συνοδευτικό DTD, απλά υποχρεούνται να εφαρμόζει όλους τους κανόνες σύνταξης της XML.
Κάθε διαφορετικός τύπος έγκυρου εγγράφου XML χρησιμοποιεί το δικό του DTD. Για να είναι εφικτή η επαλήθευση της εγκυρότητας ενός XML για συμμόρφωση με τους κανόνες που ορίζει το συνοδευτικό DTD. Με τα ίδια προγράμματα πρέπει να ελέγχονται και τα ίδια τα έγγραφα DTD για συνέπεια με τους κανόνες που ορίζονται στο πρότυπο DTD. Τα γνωστότερα προγράμματα validators είναι τα εξής:
• Γραμμής εντολών, όπως για παράδειγμα το XSV
• Αυτά που διαθέτουν API για τους προγραμματιστές όπως είναι τα
Apache Xerces
IBM Schema Quality Checker
Microsoft MSXML4.0
• Validators με γραφικό περιβάλλον
XML Spy
Turbo XML

2.9. XML Schemas

Τα XML Schemas αποτελούν σύσταση του οργανισμού W3C και ουσιαστικά είναι έγγραφα XML που καθορίζουν πώς πρέπει να δομούνται κάποια άλλα έγγραφα XML. Το πλεονέκτημά τους σε σχέση με τα DTDs (Document Type Definitions) είναι ότι μπορούν να υποστούν χειρισμούς, παραδείγματος χάρη μπορούν να προστεθούν ή να διαγραφούν στοιχεία, όπως σε οποιοδήποτε άλλο έγγραφο XML Τα XML Schemas εγγυώνται ότι ο σωστός τύπος δεδομένων αποθηκεύεται σε κάθε στοιχείο, για παράδειγμα το περιεχόμενο του στοιχείου Date_of_birth θα είναι τύπου date και όχι ακέραιος ή κάποια ακολουθία χαρακτήρων (string). Επίσης, επιτρέπουν στον χρήστη να ορίσει δικούς του τύπους δεδομένων, επιπλέον αυτών που ήδη υπάρχουν στο σύστημα τύπων XML Schema Definition (XSD). Διευκολύνουν τη μετάβαση ανάμεσα στις πλατφόρμες διότι το σύστημα τύπων XML Schema Definition (XSD) είναι ανεξάρτητο από πλατφόρμες, οπότε δεν δημιουργείται πρόβλημα εάν ένας .NET εξυπηρετητής στείλει μια τιμή ακεραίου (integer) μήκους 32 bit σε έναν Visual Basic 6.0 πελάτη, για τον οποίο ο τύπος ακέραιος (integer) έχει μήκος 16 bit.

Ακολουθεί ένα έγγραφο XML με το αντίστοιχο XML Schema.

```
Έγγραφο XML book.xml
<?xml version="1.0" ?>
<!--
αρχείο XML που επισημαίνει δεδομένα βιβλίου
-->
<book>
<title>C# How to program</title>
<serial_no>123456789</serial_no>
```

```
</book>
XML Schema bookSchema.xsd
<?xml version="1.0" encoding="utf-8" ?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
<xs:element name="book">
<xs:complexType>
<xs:sequence>
<xs:element name="title" type="xs:string"/>
<xs:element name="serial_no" type="xs:integer"/>
</xs:sequence>
</xs:complexType>
</xs:element>
</xs:schema>
```

Τα DTD είναι ένα χαρακτηριστικό της XML που κληρονομήθηκε από τον πρόγονο της γλώσσας, την SMGML. Για τον λόγο αυτό τα DTD δεν υποστηρίζουν ενδογενώς τους χώρους ονοματοδοσίας. Εξαιτίας του παραπάνω γεγονότος αλλά και πολλών περιορισμών που θέτει το DTD λόγω της συσχέτισης του με την SMGL, το W3C εξετάζει την αντικατάσταση των DTD με τη νέα σύσταση των XML Schemas. Ένα XML Schema είναι ένα XML έγγραφο με επέκταση αρχείου .xsd το οποίο εκφράζει τους κανόνες που πρέπει να ακολουθεί ένα XML έγγραφο. Ανάλογα με την λειτουργία των DTD, κάθε καλά έγκυρο έγγραφο XML μπορεί να συνοδεύεται εναλλακτικά από ένα XML Schema. Λόγω των πολλών πλεονεκτημάτων του XML Schema έναντι του DTD, πολλές φορές είναι χρήσιμο να μετατρέπουμε τα διαθέσιμα .dtd αρχεία .xsd. Τα βασικότερα πλεονεκτήματα που προσφέρουν τα XML Schemas έναντι των DTD είναι τα εξής:

• Η γλώσσα που χρησιμοποιείτε για την σύνταξη των .xsd αρχείων ακολουθεί σύνταξη ανάλογη με την σύνταξη των .xml εγγράφων. Εν αντιθέσει τα έγγραφα .dtd χρησιμοποιούν σύνταξη ασύμβατη με αυτή των .xml εγγράφων.

• Τα XML Schemas υποστηρίζουν τους χώρους ονοματοδοσίας (namespaces). Ένα namespaces ορίζει το σύνολο στο οποίο ανήκουν τα ονόματα των στοιχείων σε ένα έγγραφο XML.

• Υποστηρίζουν ενδογενώς περισσότερους τύπους δεδομένων από ότι τα DTD. Ενώ τα DTD υποστηρίζουν 10 τύπος δεδομένων, τα Schemas υποστηρίζουν τουλάχιστον 44 τύπους. Επίσης τα XML Schemas επιτρέπουν τον ορισμό νέων σύνθετων τύπων και έτσι αυξάνονται κατά πολύ οι δυνατότητες περιγραφής δεδομένων και κανόνων που τα διέπουν.

• Υποστηρίζουν κανονικές εκφράσεις (regular expressions). Ειδικά κατά τον ορισμό νέων σύνθετων τύπων η δυνατότητα αυτή είναι πάρα πολύ χρήσιμη.

• Τα Schemas προσφέρουν δυνατότητες συμπύκνωσης των tags (inlined element declarations), με αποτέλεσμα την μείωση της πολυπλοκότητας και την ευκολότερη σάρωση και ανάγνωση του εγγράφου κανόνων .xsd.

• Στα Schemas μπορούν να οριστούν στοιχεία με μηδενικό (null) περιεχόμενο ή ακόμα και δύο ή περισσότερα στοιχεία που έχουν το ίδιο όνομα , αλλά διαφορετικό περιεχόμενο.

• Μπορούν να χρησιμοποιηθούν σε ένα έγγραφο περισσότερα του ενός Schemas , αρκεί όλα να ανήκουν στο ίδιο namespace.

• Ένα Schema με το στοιχείο `<include>` μπορεί να χρησιμοποιεί στοιχεία που είναι δηλωμένα σε ένα άλλο Schema και με το στοιχείο `<redefine>` μπορεί επίσης να επανακαθορίζει τα στοιχεία αυτά.

• Η επεκτασιμότητα ενός Schema αυξάνεται αν ενσωματωθούν στο αρχείο .xsd οι τεχνολογίες XSLT/Xpath και Schematron [XFR02].
• Επιτρέπεται στα XML έγγραφα να χρησιμοποιούν πρόσθετα στοιχεία, πέραν αυτών που ορίζονται στο XML Schema, με βάση τις ανάγκες του δημιουργού περιεχομένου. Στην περίπτωση αυτή δηλώνονται τα στοιχεία `<any>` και `<anyAttribute>` στο .xsd αρχείο για να δηλώσουν την επεκτασιμότητα που περιγράφτηκε.

2.10 XML (XML Parsers)

Αν και τα έγγραφα XML είναι αρχεία κειμένου, η ανάκτηση δεδομένων από αυτά μέσω των τεχνικών σειριακής προσπέλασης αρχείων δεν είναι ούτε πρακτική, ούτε αποτελεσματική, ειδικά σε περιπτώσεις όπου τα δεδομένα πρέπει να προστεθούν ή να αφαιρεθούν δυναμικά. Το κενό αυτό έρχεται να καλύψει η διαδικασία της λεκτικής ανάλυσης (parsing), η οποία αναλύει το κείμενο στα επιμέρους στοιχεία του, στα οποία συμπεριλαμβάνονται αρχή ετικέτας, τέλος ετικέτας, κείμενο, ιδιότητες.
Στην διαδικασία της λεκτικής ανάλυσης εμπλέκεται πάντα ένας λεκτικός αναλυτής (parser).Ο λεκτικός αναλυτής κατέχει κεντρικό ρόλο στην αντιστοίχηση και αλληλοσύνδεση της δενδρικής δομής του κειμένου XML με την αντίστοιχη (συνήθως αντικειμενοστραφή) δομή στην χρησιμοποιούμενη γλώσσα, δεδομένου ότι το κείμενο XML είναι ένας συρμός χαρακτήρων που ανταλλάσσεται μέσω μίας επικοινωνιακής σύνδεσης, ενός αρχείου στον δίσκο, κλπ. Οι πιο γνωστοί λεκτικοί αναλυτές είναι ο DOM (Document Object Model) και ο SAX (Simple API for XML).

Το Μοντέλο Αντικειμένου Εγγράφου (Document Object Model - DOM) αποθηκεύει τα δεδομένα εγγράφου ως δομές δένδρων στη μνήμη με αυτόματο τρόπο, χωρίς την παρέμβαση του χρήστη κατά την εξέλιξή της ανάλυσης. Βοηθά τον χρήστη να θεωρήσει την δενδρική μορφή του XML κειμένου με αντικειμενοστραφή τρόπο, καθώς το δένδρο DOM αναπαριστά κάθε στοιχείο του εγγράφου XML (στοιχείο, ιδιότητα, κλπ.) ως ένα κόμβο στο δένδρο, και να επέμβει σε αυτό με συγκεκριμένες διεπαφές. Με το XML DOM παρέχεται, λοιπόν, η δυνατότητα στον χρήστη να δημιουργήσει ένα XML κείμενο, να πλοηγηθεί μέσα σ΄αυτό και να προσθέσει, να μεταβάλει και να αφαιρέσει στοιχεία του.

Το Simple API for XML (SAX) μοιάζει με σειριακό αναγνώστη (SAX reader) του κειμένου XML, χαρακτήρα προς χαρακτήρα, ο οποίος αναγνωρίζοντας τα διάφορα μέρη του αρχείου, δημιουργεί αντίστοιχα συμβάντα (events), στα οποία το πρόγραμμα του χρήστη οφείλει να ανταποκριθεί σύμφωνα με τις επιθυμίες του. Μόλις ο χειριστής συμβάντων (event handler) που έχει δημιουργήσει ο χρήστης αντιμετωπίσει το συμβάν, ο λεκτικός αναλυτής συνεχίζει μέχρι το επόμενο σημείο. Με τον λεκτικό αναλυτή SAX, λοιπόν, ο χρήστης βρίσκεται στο κατώτερο προγραμματιστικά περιβάλλον που του δίνει πλήρη πρόσβαση στο έγγραφο XML. Η ένταξη των επιθυμιών του (μέσω μιας εφαρμογής) είναι πιο επίπονη σε σχέση με το DOM, αλλά οι διαθέσιμες δυνατότητες και η ταχύτητα υπερτερούν. Ο χρήστης δεν περιμένει να δημιουργηθεί και να του προσφερθεί η πλήρης δομή του XML σύμφωνα με το DOM μοντέλο, αλλά ειδοποιείται και μπορεί να ανταποκριθεί στα σχετικά συμβάντα “on the fly”, καθώς διαβάζεται το XML κείμενο και χωρίς να περιμένει το τέλος αυτής της ανάγνωσης.

2.11 Το μέλλον των διαδικτυακών υπηρεσιών.

2.11.1 Ενορχήστρωση των διαδικτυακών υπηρεσιών

Για πολλούς ειδικούς το μέλλον των διαδικτυακών υπηρεσιών και το σημείο όπου θα δείξουν την πραγματική τους δύναμη, είναι η ενορχήστρωσή τους, δηλαδή ο συνδυασμός πολλών απλών υπηρεσιών για την δημιουργία ενός πολύπλοκου έργου. Ο σχεδιασμός μιας τέτοιας αρχιτεκτονικής μπορεί να περιλαμβάνει και μια ή περισσότερρες εφαρμογές που δεν είναι διαδικτυακές υπηρεσίες, με τελικό σκοπό τον σχεδιασμό ενός workflow.
Ένα παράδειγμα μιας τέτοιας ενορχήστρωσης θα ήταν οι υπηρεσίες ενός πρακτορείου διακοπών να χρησιμοποιεί τις παρακάτω διαδικτυακές υπηρεσίες.

- Υπηρεσία εύρεσης ξενοδοχείου (τοποθεσία ξενοδοχείου, τιμές δωματίων, διαθεσιμότητα, προσφερόμενες ανέσεις)
- Υπηρεσία οδικών οδηγιών
- Υπηρεσία εύρεσης και κράτησης αεροπορικών εισητηρίων
- Υπηρεσία ενοικίασης αυτοκινήτων
- Υπηρεσία εύρεσης κοινωνικών γεγονότων όπως συναυλίες, εκδηλώσεις, προβολών κινηματογράφων, θεάτρου κ.λ.π.

Η εφαρμογή του πρακτορείου, βάση κάποιων κανόνων-περιορισμών που θα απέστειλε στις διαδικτυακές υπηρεσίες, θα συνέλεγε τις πληροφορίες που επέστρεφαν και θα παρήγαγε μια τελική αναφορά με τις διαφορετικές επιλογές που θα μπορούσε να επιλέξει ο πελάτης βάσει του κόστους και των χρημάτων που θα ήταν διατεθειμένος να πληρώσει.

2.11.2 Πλέγμα (Grid) από διαδικτυακές υπηρεσίες

Η τεχνολογία Grid Computing αποτελεί ένα μοντέλο το οποίο μπορεί να επιτύχει ασφαλή, ευέλικτη και οργανωμένη κοινή χρήση υπολογιστικών πόρων ανάμεσα σε μια συνεχώς μεταβαλλόμενη ομάδα χρηστών. Μια δημοφιλής αναλογία της τεχνολογίας αυτής είναι το δίκτυο ηλεκτρικής ενέργειας. Χαρακτηριστικό του δικτύου ρεύματος είναι ότι ο χρήστης του συνδέει στην παροχή μια ηλεκτρική συσκευή και απολαμβάνει τα αγαθά της παρεχόμενης υπηρεσίας χωρίς να τον απασχολεί από πού προέρχεται ούτε πως παράγεται, ενώ χρεώνεται για το ακριβές ποσό ενέργειας που χρησιμοποιεί μέσω του μετρητή που είναι εγκατεστημένος στο σπίτι του.

Αντίστοιχα στο περιβάλλον του Grid Computing, ένας χρήστης ή μια εφαρμογή μπορεί να συνδεθεί στο δίκτυο με μια απλή διεπαφή και να έχει άμεσα πρόσβαση σε πόρους χωρίς να γνωρίζει την τοποθεσία τους. Οι διαδικτυακές υπηρεσίες και οι τεχνολογίες τους σε συνδυασμό με ένα τέτοιο περιβάλλον, θα επιτρέψουν σε οργανισμούς να παρέχουν στους πελάτες υπηρεσίες σε μεγάλη κλίμακα λύνοντας προβλήματα όπως η αντοχή σε σφάλματα (fault tolerance), ο διαμοιρασμός του φόρτου εργασίας (load balancing) και η διαθεσιμότητα (availability).

2.11.3 Ένας Σημασιολογικός Ιστός από διαδικτυακές υπηρεσίες

Η εξέλιξη του σημασιολογικού ιστού και των διαδικτυακών υπηρεσιών είναι παράλληλη. Η γλώσσα XML και μόνο δεν αρκεί, καθώς υπάρχει ανάγκη για αυτοματοποιημένο χειρισμό των αμέτρητων ειδικευμένων τύπων δεδομένων που υπάρχουν (και θα συνεχίσουν να αυξάνονται). Στα επόμενα χρόνια θα δούμε τα σημασιολογικά δεδομένα να περιγράφουν προβλήματα και επιχειρηματικές διαδικασίες σε ειδικευμένες εφαρμογές και τομείς. Οι οντολογίες θα παίξουν βασικό ρόλο στον σκοπό αυτό που θα οδηγήσει στον σημασιολογικό ιστό ο οποίος θα κατορθώσει τον συνυφασμό της ανθρώπινης νόησης με την αυτοματοποιημένη επεξεργασία δεδομένων από τους υπολογιστές.

ΚΕΦΑΛΑΙΟ 3

DOM (Document Object Model)

3.1. Γενικά

Για την ανάγνωση και επεξεργασία των εγγράφων XML μέσα από μια εφαρμογή έχουν αναπτυχθεί δύο API (Application Program Interfaces). Το DOM έχει συσταθεί και υποστηρίζεται σαν πρότυπο από το W3C (World Wide Web Consortium). Διαθέτει επίσης ευρεία υποστήριξη από πολλά προγράμματα φυλλομέτρησης του Ιστού (Web Browsers). Το DOM εκτός από ανάγνωση προσφέρει δυνατότητες και για τροποποίηση υπαρχόντων ή για δημιουργία νέων XML εγγράφων. Το δεύτερο API ονομάζεται SAX (Simple API for XML). Το SAX βασίζεται σε μια λογική συμβάντων (events) και χειριστών συμβάντων (events handlers) και παρέχει στον προγραμματιστή λειτουργίες χαμηλότερου επιπέδου. Το SAX, σε αντίθεση με το DOM, δεν υποστηρίζεται από καμία επίσημη τυποποίηση, αλλά χρησιμοποιείται ευρύτατα και θεωρείται ως μια de facto τυποποίηση. Εμείς στην παρούσα πτυχιακή θα ασχοληθούμε με το DOM και θα αναλύσουμε περαιτέρω τα χαρακτηριστικά του.

3.2 Δομή του DOM

3.2.1 Το Web έγγραφο με δομή DOM

Το Document Object Model αναπαριστά το Html έγγραφο με την δομή ενός δέντρου. Κάθε κόμβος του δέντρου δηλώνει μία Html ετικέτα (tag) ή κάποια οντότητα κειμένου (textual entry) σαν περιεχόμενο Html ετικέτας (tag). Όλο το Html έγγραφο περιγράφεται μέσα από την δομή του δένδρου και των τύπων των σχέσεων μεταξύ των κόμβων μέσα ως προς την ιεραρχία.

- Γονέας κόμβος (parent)
- Παιδί κόμβος (child) – Αδελφός κόμβος (sibling)

Με το DOM έχουμε την δυνατότητα να πλοηγούμε μέσα στο έγγραφο HTML και να προσπελάζουμε οποιαδήποτε ετικέτα σήμανσης καθώς και το κείμενο που μπορεί να περιέχει, είτε με την μέθοδο της ιεραρχικής διαδρομής των κόμβων (walk method) είτε με την μέθοδο της άμεσης προσπέλασης (by tag or id name) με αναφορά στο όνομα της συγκεκριμένης ετικέτας (tag name) ή στο αναγνωριστικό της όνομα (id).

Η εφαρμογή στην συνέχεια των ιδιοτήτων και των μεθόδων είναι κοινή σε κάθε κόμβο-ετικέτα του εγγράφου, ανάλογα με τον τύπο του κόμβου (node type).

Οι δυνατότητες που έχουμε με το Dom, ως προς την εισαγωγή νέων κόμβων στο έγγραφο HTML, διαγραφής υπαρχόντων κόμβων και ενημέρωσης των περιεχομένων των κόμβων, καθιστούν το DOM το αδιαμφισβήτητο σύστημα διαχείρισης του εγγράφου.

Το DOM, αντιμετωπίζοντας το έγγραφο HTML, σαν ένα αρχείο αντικειμένων με συγκεκριμένες πληροφορίες και σήμανση, το διαχειρίζεται με την JavaScript, μέσα από τις βασικές αρχές του αντικειμενοστραφούς προγραμματισμού και τις μεθόδους διαχείρισης αρχείων όπως εισαγωγή κόμβων (input), ενημέρωση κόμβων (update) και διαγραφή κόμβων (delete).

3.2.2 Η ιδιότητα id

Η προδιαγραφή του DOM, απαιτεί όπως κάθε ετικέτα (tag) του HTML-XHTML ή XML εγγράφου, να χαρακτηρίζεται με ένα μοναδικό αναγνωριστικό όνομα, σαν περιεχόμενο της ιδιότητας id για το συγκεκριμένο στοιχείο που θα δηλώνει η ετικέτα αυτή.

Με το όνομα αυτό αναγνωρίζεται ο κάθε ένας από τους κόμβους του εγγράφου μέσα στο DOM. Στο παράδειγμα που ακολουθεί το στοιχείο της συγκεκριμένης παραγράφου του εγγράφου, έχει αναγνωριστικό όνομα “par1”.

```
<p id=”par1”>…</p>
```

Η Netscape και η Microsoft, στην προσπάθεια τους, να αντικειμενοποιήσουν τα στοιχεία των HTML εγγράφων, με τους πρώτους browsers που κυκλοφόρησαν στην αγορά, παρουσίασαν τα δικά τους μοντέλα αντικειμενοποίησης εγγράφου, έτσι ώστε η ασθενής γλώσσα σήμανσης με το όνομα HTML, να μπορεί να λειτουργεί έστω και παραμετρικά μέσω της JavaScript σαν αντικειμενοστραφής γλώσσα.

3.2.3 W3C & DOM

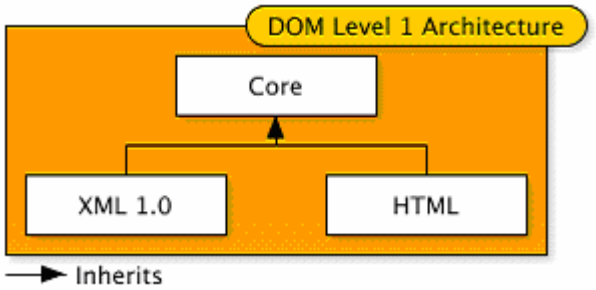
Το DOM (Μοντέλο αντικειμένου εγγράφου) έγινε αρχικά από τον οργανισμό προτυποποίησης W3C με σκοπό την συμβατότητα των browsers και κατά κύριο λόγο του Netscape Navigator 3 και του Internet Explorer 3.

Η σύσταση του DOM, υποστηρίζει έγγραφα και HTML και XML. Το DOM επιτρέπει στους προγραμματιστές

1. την προσπέλαση και πρόσθεση
2. την προσπέλαση και διαγραφή
3. την προσπέλαση και χειρισμό
4. όλων των στυλ μορφοποίησης (styles)
5. όλων των στοιχείων (elements)
6. όλων των ιδιοτήτων (attributes) του εγγράφου (document)

Το έγγραφο μπορεί να προσπελαθεί με το μοντέλο DOM, μέσα από διαφορετικές γλώσσες προγραμματισμού που υποστηρίζει ο browser, όπως α)Java, JavaServer Pages β)JavaScript/ECMAScript/Jscript γ) VBScript(MSIE).

Το Dom 1 (level 1), καθορίζει την πλήρη τυποποίηση του εγγράφου ως προς τα αντικείμενα, πως δηλαδή είναι δομημένα τα αντικείμενα που το απαρτίζουν και πως μπορούμε να αναφερόμαστε σε αυτά.



Σχήμα 4) Το σχεδιάγραμμα του πυρήνα DOM Level 1

Το DOM, μας παρέχει έναν τυπικό τρόπο πρόσβασης στα αντικείμενα του εγγράφου, ανεξάρτητα από την γλώσσα προγραμματισμού ή τα προγράμματα ανάγνωσης του εγγράφου (browsers) και μας δίνει την δυνατότητα να αναπτύσσουμε εφαρμογές DOM χρησιμοποιώντας την JavaScript ή άλλες γλώσσες προγραμματισμού.

Η σύνδεση του W3C DOM με την ECMAScript, που αποτελεί την τυποποίηση της JavaScript, μας επιτρέπει να παράγουμε εφαρμογές που υποστηρίζουν όσους περισσότερους browsers είναι δυνατόν.

3.2.4 Η Σύσταση W3C DOM (Level 2)

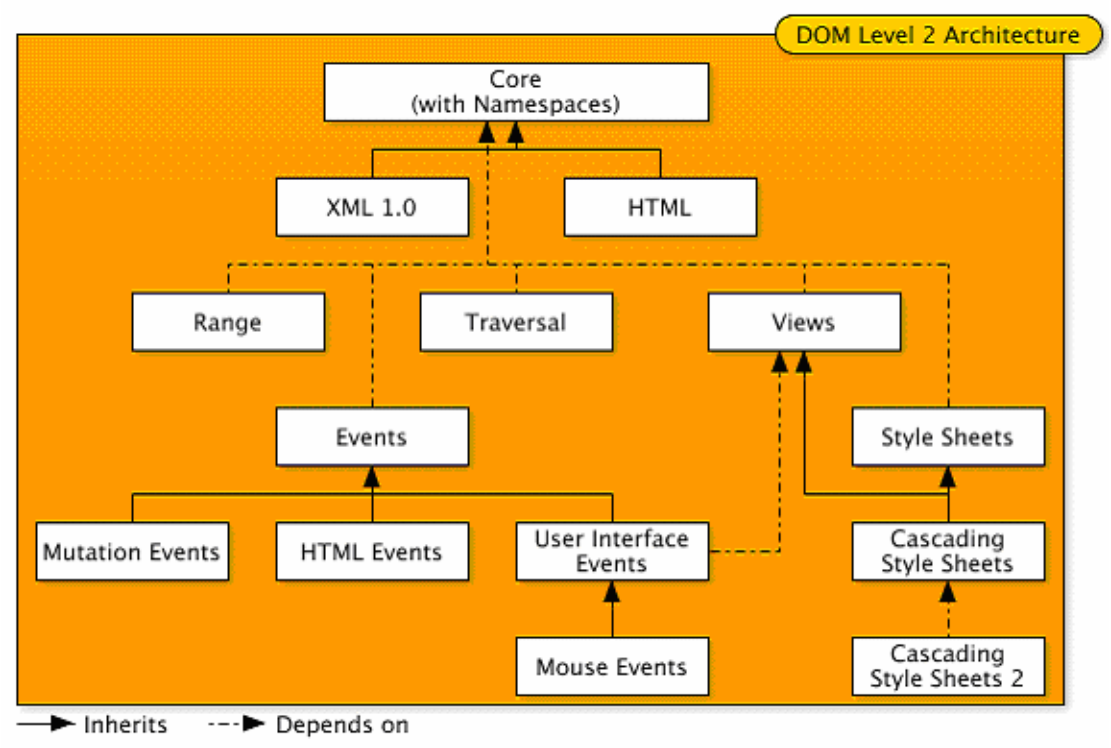
Το DOM Level 2, επιπρόσθετα καθορίζει:

1. Την εφαρμογή του μοντέλου (Namespaces) Δημιουργίας Χώρων Ονομάτων για τα στοιχεία του εγγράφου (prefix-elements handling)
2. Τις εφαρμογές των μοντέλων (Range & Traversal)

(Range) Χειρισμού του περιεχομένου του εγγράφου συγκεκριμένης τάξης (range) με καθορισμένα όρια,

(Traversal) Δημιουργίας υπό-δέντρου (sub-tree) λίστας κόμβων (nodes) του εγγράφου και μετακίνησης εντός αυτής με σκοπό την διαχείριση της.

3. Την εφαρμογή του μοντέλου (Event Model) χειρισμού των συμβάντων (Event handling)
4. Την εφαρμογή του μοντέλου (Style Sheets) χειρισμού των φύλλων Στυλ (style handling)



Σχήμα 5)
Το Σχεδιάγραμμα Αρχιτεκτονικής του πυρήνα DOM Level 2- είναι του W3C

Οι browsers που υποστηρίζουν το DOM, είναι ο Internet Explorer 5+, και ο Netscape 6+ και μας δίνουν πλέον την δυνατότητα να έχουμε πρόσβαση σε οποιοδήποτε κώδικα, στοιχείο (element) ή ιδιότητα (attribute) στοιχείου HTML.

Η υποστήριξη ταυτόχρονα πολλαπλών browsers (cross-browser), από το DOM, έχει εφαρμογή μόνο στις εκδόσεις των δύο κυριοτέρων browsers της αγοράς. Ο έλεγχος της συμβατότητας προς τα πίσω, που αφορά τους παλαιότερους browsers που δεν υποστηρίζουν το DOM, παραμένει έργο των προγραμματιστών, είτε μέσα από την ανίχνευση του Browser (Browser Detection), είτε μέσα από τα αντικείμενα που αυτός υποστηρίζει (Object Detection).

3.2.5 Ομάδα Διαχείρισης Αντικειμένων & IDL (Interface Definition Language)

Το Μοντέλο Αντικειμένου Εγγράφου-DOM, δεν έχει κατασκευαστεί μόνον για την αγορά των browsers, ώστε να περιορίζεται στην JavaScript, αλλά σαν εφαρμογή (API-Application Program Interface) για πολλαπλές πλατφόρμες (multiplatform) και πολλαπλές γλώσσες προγραμματισμού (Multilanguage interface).

Η ομάδα διαχείρισης αντικειμένων OMG (Object Management Group) του W3C έχει δημιουργήσει την γλώσσα καθορισμού διασύνδεσης –IDL (Interface Definition Language) μία ειδική γλώσσα διασύνδεσης αντικειμένων (object interfaces).

Η OMG IDL, χρησιμοποιείται όχι για να περιγράψει τα αντικείμενα, τι ακριβώς δηλαδή κάνουν, αλλά τι ιδιότητες και τι μεθόδους έχουν αυτά. Με τον τρόπο αυτό η IDL, χαρτογραφεί τα στοιχεία-αντικείμενα μίας αντικειμενοστραφούς γλώσσας προγραμματισμού-OOP (Object Oriented Programming) και την

συγκεκριμένη πλέον προδιαγραφή DOM σε IDL, που κατασκευάζει για την γλώσσα αυτή, επιτυγχάνει πολλαπλή υποστήριξη γλωσσών (cross-language).

Μέχρι αυτή την στιγμή, υπάρχει χαρτογράφηση σε IDL, των Java, C++, Ada, Smalltalk και Cobol καθώς και διαθέσιμες οι αντίστοιχες προδιαγραφές DOM.

3.2.6 Το μοντέλο αντικειμένων της Netscape

Η Netscape, στο δικό της μοντέλο αντικειμένου εγγράφου, πριν τις εκδόσεις NS 6+, χρησιμοποιεί σαν βασικό αντικείμενο του εγγράφου (root element) το document. Για κάθε είδος στοιχείου του εγγράφου δημιουργεί τον αντίστοιχο πίνακα, όπου και αποθηκεύει όλα τα όμοια στοιχεία με το διάβασμα του εγγράφου.

// Επιστρέφει ένα πίνακα με όλες τις εικόνες του εγγράφου

```
document.images
```

// Επιστρέφει ένα πίνακα με όλες τις φόρμες του εγγράφου

```
document.forms
```

// Επιστρέφει ένα πίνακα με όλους τους συνδέσμους του εγγράφου.

```
document.links
```

Η προσπέλαση του αντικειμένου της παραγράφου

```
<p id="part1">…</p>
```

εάν υποθέσουμε πως περιέχεται σε μία από τις φόρμες του εγγράφου με αναγνωριστικό όνομα `id="formaB"` είναι :

```
document.forms[formaB].part1
```

3.2.7 Το μοντέλο Αντικειμένων της Microsoft

Η Microsoft, με το δικό της μοντέλο αντικειμένου εγγράφου, στις εκδόσεις πριν τον IE 5+, χρησιμοποιεί σαν βασικό αντικείμενο του εγγράφου (root element) το document.

Σαν δεύτερο αντικείμενο στην ιεραρχία των αντικειμένων, χρησιμοποιεί τον πίνακα all, στον οποίο αποθηκεύει όλες τις ετικέτες του εγγράφου.

Ο Internet Explorer, εξάγοντας όλα τα στοιχεία ενός HTML εγγράφου σαν αντικείμενα, απαιτεί ήδη, σύνταξη καθορισμού διευθύνσεων (addressing) των ετικετών, δηλαδή αναφορά της ιδιότητας id, σε κάθε ετικέτα. Όλες τις ετικέτες ενός εγγράφου HTML, τις τοποθετεί στον πίνακα αντικείμενο all

```
document.all
```

Η προσπέλαση λοιπόν της συγκεκριμένης ετικέτας-αντικειμένου με τον Internet Explorer γίνεται :

// Με πλήρη διαδρομή της ιεραρχίας αντικειμένων

```
var object = document.all.part1maB].part1
```

// Και με άμεση προσπέλαση του ονόματος id.

```
var object = part1
```

3.3 DOM Κόμβοι (Nodes)

3.3.1. Οι Κόμβοι (nodes) και το Οικογενειακό Δέντρο

Το DOM σαν μία εφαρμογή διασύνδεσης (API), χρησιμοποιεί την τεχνολογία των κόμβων για να περιγράψει την δομή και το περιεχόμενο του εγγράφου. Κάθε στοιχείο του εγγράφου HTML καθώς και το περιεχόμενο του αναπαρίστανται από ένα ξεχωριστό κόμβο DOM. Δείτε την δομή του εγγράφου HTML που ακολουθεί.

Έγγραφο: dom10.html

```
<html>
    <head>
          <title> Αρχική Σελίδα DOM </title>
    </head>
    <body>
        <p id="part1">
       Το DOM είναι
        <b id="bold1"> το web πρότυπο </β>
      </p>
      </body>
  </html>
```

Κάθε στοιχείο (element) του εγγράφου HTML, αποτελεί ένα κόμβο-στοιχείου στην δομή DOM και καθορίζεται από μία ετικέτα αρχής και μία ετικέτα τέλους. Κάθε κείμενο (text) που περιέχεται εντός HTML στοιχείου, αποτελεί και αυτό ένα κόμβο-κειμένου (text-node).

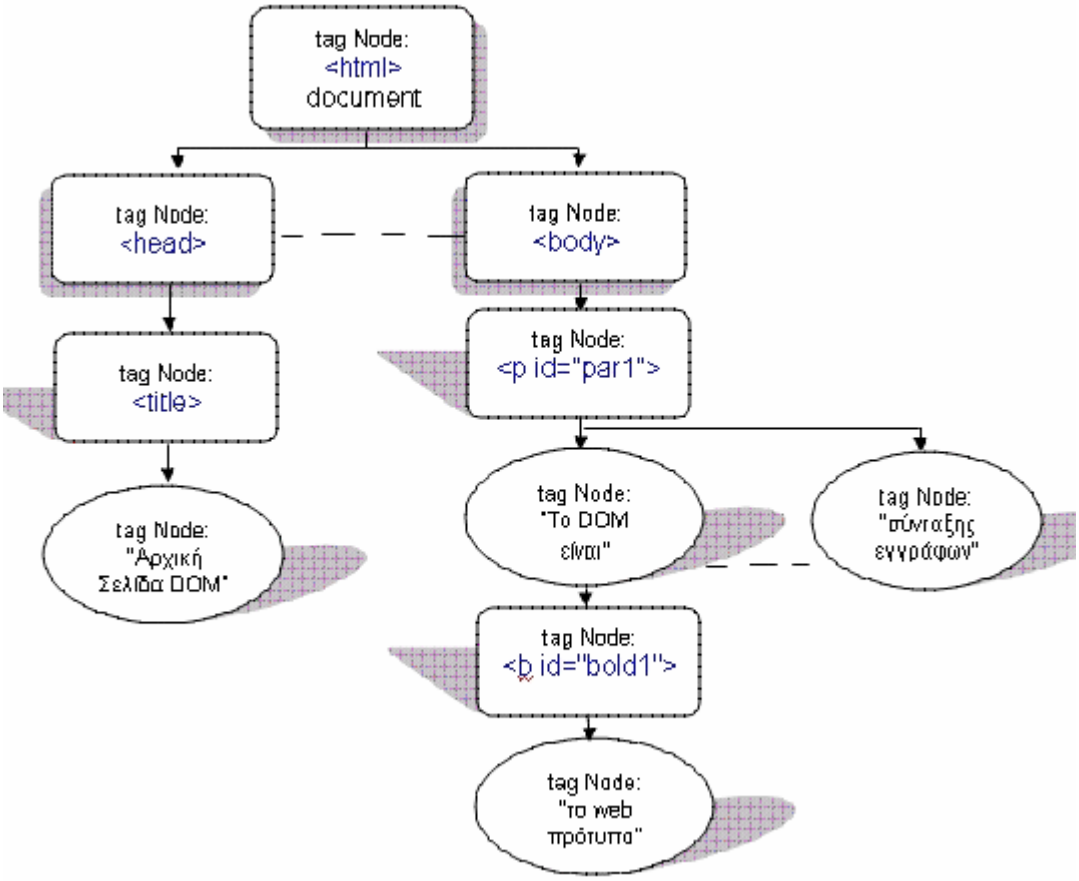
Εάν στο έγγραφο περιέχονται ετικέτες μόνον αρχής όπως
, <img>, <hr> κ.λ.π. θεωρούνται και αυτές επίσης κόμβοι που στέκονται μόνοι τους στα κλαδιά (branches) της ιεραρχικής δομής του δέντρου και οι οποίοι είναι ικανοί να γεννήσουν άλλους κόμβους.

Το στοιχείο HTML ορίζει τον βασικό κόμβο του εγγράφου (document node), αφού αυτός περιέχει εντός του όλα τα άλλα στοιχεία μαζί και τα περιεχόμενα τους. Ο κόμβος αυτός είναι ο γονέας (parent) των κόμβων-παιδιά (children) HEAD and BODY, που μεταξύ τους είναι αδέλφια (siblings).

Τα επίπεδα ιεραρχίας των κόμβων καθορίζονται από το πόσοι κόμβοι περιέχονται εντός άλλων κόμβων κ.ο.έ.

Ας δούμε το διάγραμμα του DOM του HTML εγγράφου του παραδείγματος.

Διάγραμμα: dom10.html



Σχήμα 6) Διάγραμμα του DOM του εγγράφου dom10.html

Οι διακεκομμένες γραμμές δείχνουν το ίδιο επίπεδο ιεραρχίας και πως οι συγκεκριμένοι κόμβοι έχουν τον ίδιο γονέα (parent) είναι δηλαδή μεταξύ τους αδέλφια (siblings).

3.3.2 Προσπέλαση εγγράφου με το DOM-W3C

Το W3C-DOM, δίνει την δυνατότητα στους browsers που είναι συμβατοί με αυτό, μέσω της μεθόδου getElemtsByTagName () να επιστρέφουν ένα πίνακα κάθε φορά, με όλες τις ετικέτες που περιέχονται στο HTML έγγραφο και οι οποίες έχουν

ίδιο όνομα ετικέτας με αυτό της παραμέτρου μέσα στην παρένθεση. Το όνομα της ετικέτας θα πρέπει να περικλείεται σε διπλά εισαγωγικά ως αλφαριθμητικό (“string”).

Μπορούμε να έχουμε σαν επιστροφή εντός του πίνακα με την εν λόγω μέθοδο, όλες τις ετικέτες του εγγράφου εάν σαν παράμετρο δηλώσουμε τον αστερίσκο(“*”).

```
If (document.getElementsByTagName(“*”))
        {
                var alltags=document.getElementsByTagName(“*”)
        }
```

Με το ανωτέρω script εκχωρήσαμε όλες τις ετικέτες (tags) των στοιχείων του εγγράφου στο αντικείμενο-μεταβλητή alltags. Στην συνέχεια, μπορούμε να προσπελάσουμε οποιαδήποτε ετικέτα (tag) του εγγράφου νε το αναγνωριστικό της όνομα id στην περίπτωση του παραδείγματος

```
<p id=”par1”>…</p>>
```

θα έχουμε:

Με την πλήρη διαδρομή της ιεραρχίας

```
Var object=document.alltags.par1
```

Με άμεση προσπέλαση id

```
Var object=document.getElementbyId(“par1”)
```

Στην προδιαγραφή του DOM επιπέδου (Level 2) οι ιδιότητες των φύλλων Στυλ (Style Sheets), μπορούν να προσπεάθούν μέσα από την ιδιότητα style κάθε στοιχείο (element) HTML. Έστω, ότι θέλουμε να αλλάξουμε την γραμματοσειρά για το περιεχόμενο κείμενο της παραγράφου

```
<p id=”par1”>…</p>
```

από Ariel σε Verdana, η σύνταξη της πρότασης DOM, θα είναι

```
document.alltags.par1.style.font = “verdana”
```

Είναι ομολογουμένως εκπληκτικές οι δυνατότητες προσπέλασης τόσο των στοιχείων-κόμβων του εγγράφου, όσο και των ιδιοτήτων τους με το Document Object Model.

Η τεχνική αυτή δεν μπορεί να έχει εφαρμογή στον Internet Explorer 5, επειδή δεν είναι πλήρως συμβατός (full compatible) με την προδιαγραφή DOM (Level 1). Αυτό έχει σαν αποτέλεσμα να μην υποστηρίζει την συλλογή όλων των ετικετών του εγγράφου με τιμή παραμέτρου(“*”), σε πίνακα μέσω της συγκεκριμένης ιδιότητας,

```
var alltags = document.getElementsByTagName(“*”)
```

με αποτέλεσμα να επιστρέφει τιμή μηδέν (zero).

3.3.3 Ο Κόμβος είναι ο πυρήνας του DOM

Ο πυρήνας αντικείμενο (core object) της δομής DOM, είναι ο Κόμβος (Node).

- Κάθε ετικέτα (tag) στοιχείου (element) του εγγράφου, αποτελεί ένα κόμβο (node object)
- Τα αντικείμενα του εγγράφου είναι οργανωμένα σε δομή δέντρου (tree structure).
- Πρώτο στην ιεραρχία και στην ρίζα του δέντρου βρίσκεται το αντικείμενο document.
- Τα αντικείμενα-κόμβοι (node objects) του DOM, κληρονομούν (inheritance) τις τιμές των ιδιοτήτων τους από ανώτερους ιεραρχικά κόμβους και μπορούν να τις μεταβιβάζουν σε κατώτερους ιεραρχικά κόμβους.

Η προσπέλαση στα αντικείμενα-κόμβους (node objects) του DOM, γίνεται είτε με την ιεραρχική αναφορά της διαδρομής των αντικειμένων του δέντρου μέχρι το αντικείμενο-κόμβο που ζητούμε, είτε με αναφορά του ονόματος (ID="…") που έχουμε εκχωρήσει ως ταυτοτικό όνομα στο αντικείμενο-κόμβο, δηλαδή μέσω των ιδιοτήτων του εγγράφου (attributes).

Όλη η ουσία του DOM είναι η μοντελοποίηση των εγγράφων, έτσι ώστε κάθε στοιχείο τους (element) να μπορεί να χρησιμοποιηθεί σαν αντικείμενο (object), ανεξάρτητα από την δομή της οργάνωσης που αυτά έχουν μέσα στο έγγραφο. Οι κόμβοι δεν αναπαριστούν την δομή οργάνωσης του εγγράφου, αλλά τα στοιχεία του, σαν αντικείμενα (objects) με συγκεκριμένες ιδιότητες, τιμές και μεθόδους αναφοράς σε αυτές. Στις γλώσσες που είναι προσανατολισμένες προς τον αντικειμενοστραφή προγραμματισμό, τα δεδομένα (data) βρίσκονται εγκλεισμένα (encapsulated) εντός αντικειμένων και προσπελαύνονται μόνο μέσα από τις μεθόδους (methods) των συγκεκριμένων αντικειμένων που τα περιέχουν. Η προστασία, από την άμεση εξαγωγή και τον χειρισμό των δεδομένων (data), που παρέχεται με τον εγκλεισμό τους μέσα στα αντικείμενα, είναι μέρος του μοντέλου αντικειμένων (object model) που υποστηρίζει το DOM.

Ως μοντέλο αντικειμένου (object model) το DOM παρέχει:

- Διεπαφές (interfaces) και Αντικείμενα (objects), έτσι ώστε να αναπαριστούν και να χειρίζονται ένα έγγραφο.
- Την εννοιολογική σημασία των διεπαφών και των αντικειμένων μαζί με την συμπεριφορά και τις ιδιότητές τους.
- Τις σχέσεις και τις συνεργασίες ανάμεσα στις διεπαφές και τα αντικείμενα.

Κάθε έγγραφο μπορεί είτε να περιέχει είτε όχι, έναν κόμβο που καθορίζει τον τύπο του (doctype).

- Στο DOM τα έγγραφα αναπαρίστανται με την λογική δομή (structure model) οργάνωσης του δέντρου (tree structure).
- Υπάρχει ένας κόμβος ρίζας (root element node), που είναι η ρίζα του δέντρου των στοιχείων. (element tree) του εγγράφου.
- Μέσω της δενδροειδούς δομής του εγγράφου, μπορούμε να προσπελάσουμε στοιχεία του, με την μέθοδο (tree walking), η

οποία προσδιορίζει ιεραρχικά την διαδρομή, χωρίς να αναφερθούμε σε ιδιότητες.

Το DOM παρέχει δομή ισομορφισμού (structural isomorphism) :

- Εάν για την αναπαράσταση του ίδιου εγγράφου, έχουν χρησιμοποιηθεί δύο εφαρμογές DOM, θα δημιουργηθεί το ίδιο μοντέλο, σύμφωνα με τις προδιαγραφές της XML.
- Κάποιες μικρές διαφορές, εάν παρουσιαστούν, αυτές θα οφείλονται στον αναλυτή (parser) του εγγράφου που θα διαβάσει το έγγραφο και από αυτό θα δημιουργήσει το αντίστοιχο μοντέλο αντικειμένων DOM.

Σαν παράδειγμα μπορούμε να αναφέρουμε τα λευκά διαστήματα (whitespaces) τα οποία μπορεί να μην δέχεται σαν περιεχόμενο (content) στοιχείων του εγγράφου του DOM, εάν τα έχει αποκόψει ο αναλυτής (parser) του εγγράφου.

Το μοντέλο αντικειμένου εγγράφου, αναπαριστά τα XML και HTML έγγραφα σε δομές δέντρων αντικειμένων, έτσι ώστε αυτά να μπορούν να χρησιμοποιηθούν μέσα από τον αντικειμενοστραφή προγραμματισμό. Το DOM, είναι απλά ένα σύνολο εφαρμογών προγραμμάτων διασύνδεσης (APIs-Application Program Interfaces), και αντικειμένων με σκοπό την διαχείριση εγγράφων XML και HTML.

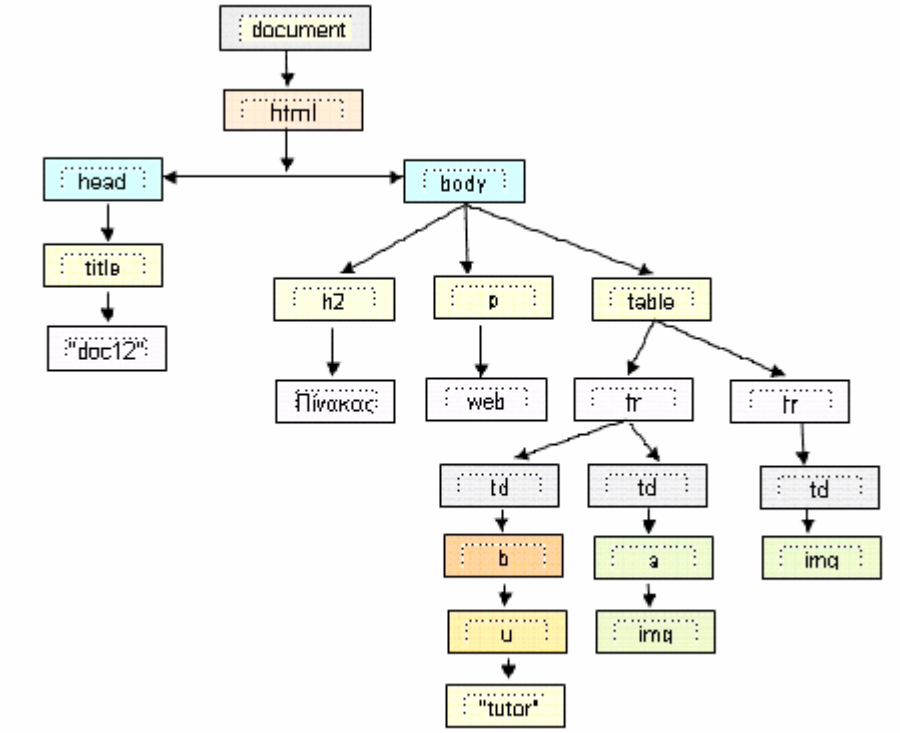
3.3.4 Οι κόμβοι του HTML-XHTML εγγράφου

Όλες οι ετικέτες των στοιχείων HTML-XHTML αποτελούν κόμβους, του εγγράφου με δομή DOM, καθώς επίσης και τα κείμενα (text) μεταξύ αυτών όπου αυτά υπάρχουν. Ας δούμε το έγγραφο που παραθέτουμε πιο κάτω.

Έγγραφο:domHTML.html

```
<html>
   <head>
      <title>doc12</title>
   </head>
   <body>
      <h2>Πίνακας</h2>
       <p>web</p>
       <table width="100%">
       <tr>
     <td><b><u>tutor</u></b><td>
     <td><a href="starweb.html">
          <img src="srat.gif"border="0" />
          </a>
     </td>
         </tr>
         <tr>
         <td>
          <img src="sky.gif" border="0" />
         </td>
          </tr>
     </table>
      </body>
</html>
```

Το διάγραμμα που ακολουθεί αναπαριστά την δομή του DOM του εγγράφου καθώς και τις σχέσεις των επί μέρους στοιχείων με βάσει την διάκριση τους σε επίπεδα ιεραρχίας που αποτελούν τα κλαδιά



Σχήμα 7) Διάγραμμα της δομής του DOM του εγγράφου domHTML.html

Η ιεραρχία στη δομή των αντικειμένων παρουσιάζεται ανά επίπεδο με διαφορετικά χρώματα. Το DOM απαιτεί όλα τα στοιχεία του εγγράφου να υποχρεωτικά να ολοκληρώνουν την σήμανσή τους με ετικέτα τέλους. Οι διακεκομμένες γραμμές δείχνουν το ίδιο επίπεδο ιεραρχίας και πως οι συγκεκριμένοι κόμβοι έχουν τον ίδιο γονέα (parent) είναι δηλαδή μεταξύ τους αδέλφια (siblings).

3.3.5. Οι κόμβοι του XML εγγράφου

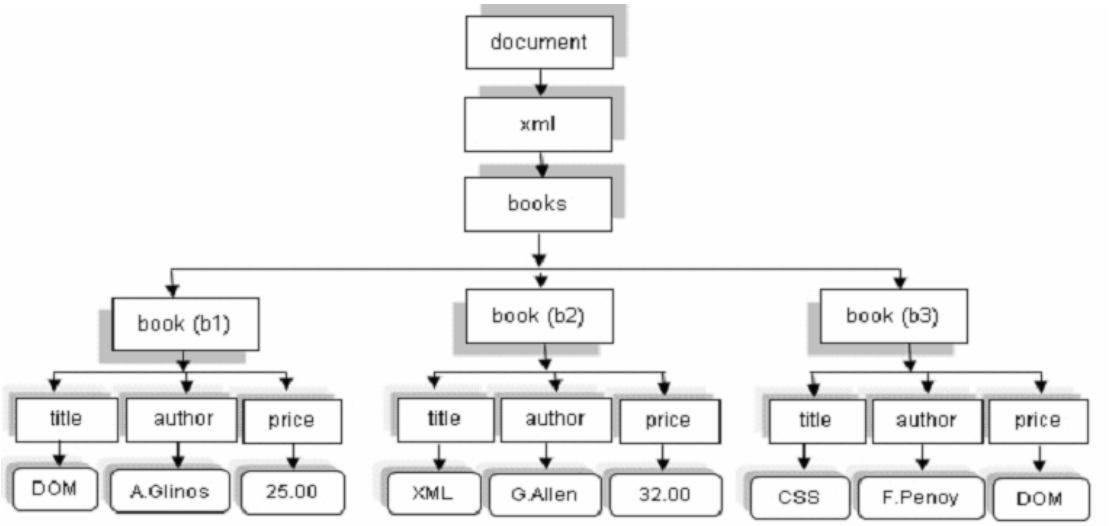
Όλες οι ετικέτες των στοιχείων του εγγράφου XML αποτελούν κόμβους DOM, καθώς επίσης και τα κείμενα (text) μεταξύ αυτών όπου υπάρχουν. Ας παρατηρήσουμε το παρακάτω XML έγγραφο.

Έγγραφο : domXML.xml

```
<?xml version="1.0"?>
     <books>
          <book id="b1">
               <title>DOM</title>
               <author>A.Glinos</author>
               <price>25.00</price>
          </book>
          <book id="b2">
```

```
                    <title>XML</title>
                    <author>G.Allen</author>
                    <price>32.00</price>
               </book>
               <book id="b3">
                    <title>CSS</title>
                    <author>F.Penoy</author>
                    <price>22.00</price>
               </book>
          </books>
```

Στο διάγραμμα που ακολουθεί, αναπαριστούμε το ανωτέρω έγγραφο XML σε μοντέλο DOM. Ο κόμβος που αφορά την ετικέτα του εγγράφου doctype (πως είναι έγγραφο XML) είναι ο κόμβος XML, αμέσως μετά τον κόμβο document. Ο κόμβος ρίζας του εγγράφου root element node, είναι ο κόμβος books.



Σχήμα 8) Διάγραμμα αναπαράστασης του XML εγγράφου domXML.xml σε μοντέλο DOM

3.3.6 Κόμβοι Γονείς, Παιδιά και αδέλφια

Στην JavaScript τα αντικείμενα που εμπεριέχουν άλλα αντικείμενα, καλούνται γονείς (parents) αυτών και αυτά που εμπεριέχονται παιδιά τους (children). Το μοντέλο αντικειμένου εγγράφου-DOM του W3C, ακολουθεί την ίδια ορολογία έτσι ώστε να αναφέρεται σε κόμβους που εμπεριέχουν άλλους κόμβους και αυτοί με την σειρά τους άλλους κ.ο.ε.

Επιπρόσθετα το DOM, χαρακτηρίζει σαν αδέλφια (siblings), τους κόμβους-αντικείμενα που βρίσκονται στο ίδιο επίπεδο ιεραρχίας και εμπεριέχονται στον ίδιο κόμβο-αντικείμενο, έχουν δηλαδή τον ίδιο γονέα. Στο διάγραμμα domHTML, οι κόμβο-αντικείμενα h2,p και table, είναι αδέλφια, αφού βρίσκονται στο ίδιο επίπεδο ιεραρχίας της δομής του δέντρου του εγγράφου και παράλληλα έχουν τον ίδιο γονέα, εμπεριέχονται δηλαδή στον κόμβο-αντικείμενο body.

Αντίθετα οι κόμβοι-αντικείμενα "πίνακας","web", tr και tr ενώ βρίσκονται στο ίδιο επίπεδο ιεραρχίας δεν είναι όλα αδέλφια μεταξύ τους, αλλά μόνο τα tr και tr,

επειδή αυτά έχουν τον ίδιο γονέα, τον κόμβο-αντικείμενο table. Τα περιεχόμενα κείμενα εντός στοιχείων του εγγράφου, αποτελούν κόμβους-αντικείμενα κειμένου (text) και ως εκ τούτου είναι παιδιά των κόμβων-αντικειμένων που τα εμπεριέχουν.

Τα κείμενα "Πίνακας" και "web", είναι παιδιά των γονέων κόμβων-αντικειμένων h2 και p αντίστοιχα. Όπως ακριβώς και ο κόμβος αντικείμενο "doc12", έχει κόμβο-αντικείμενο γονέα το στοιχείο title.

Στο διάγραμμα domXML, το οποίο αναπαριστά το μοντέλο αντικειμένων DOM για το XML έγγραφο, ακολουθώντας την ιεραρχία της δομής δέντρου έχουμε σαν παιδιά αντικείμενα που περιέχονται σε άλλα αντικείμενα, όπως τα book(b1), book(b2) και book(b3) που είναι παιδιά του κόμβου books και με την σειρά τους είναι γονείς των κόμβων title, author και price το καθένα ξεχωριστά.

Τα παιδιά του κάθε ενός από τους κόμβους book(b1), book(b2) και book(b3) είναι αδέλφια χωρίς όμως να είναι αδέλφια και με τα παιδιά των άλλων κόμβων.

3.3.7 Κατηγορίες Κόμβων

Το DOM, οργανώνει το κάθε τι μέσα στο έγγραφο, ως κόμβο-αντικείμενο μέρος της δομής ενός δέντρου. Οι κόμβοι σαν γενικά αντικείμενα (generic objects) κατηγοριοποιούνται ανάλογα με το τι ακριβώς δηλώνουν μέσα στο έγγραφο, όπως για παράδειγμα κόμβοι στοιχείων (elements nodes), κόμβοι ιδιοτήτων στοιχείων (attributes nodes), κόμβοι κειμένου (text nodes), κόμβοι σχολίων (comment nodes) κ.ο.ε. Στον πίνακα που παραθέτουμε δίνεται η κωδικοποίηση που μπορεί να χαρακτηρίζει ένα κόμβο στο DOM.

Πίνακας 1) Κατηγορίες Κόμβων

Κωδικός Node	Τύπος Κόμβου Node Type	Όνομα Κόμβου Node Name	Τιμή Κόμβου Node Value	Ιδιότητες AttributesI
1	Στοιχείο (Element)	tagName	κενό (null)	NamedNodeMap
2	Ιδιότητα (Attr)	Όνομα ιδιότητας-attr	Τιμή ιδιότητας κόμβου	κενό (null)
3	Κείμενο (Text)	#text	Περιεχόμενο κόμβου κειμένου	κενό (null)
4	Ενότητα πληροφοριών (CDATA)	#cdata-section	Περιεχόμενο της δήλωσης CDATA	κενό (null)
5	Αναφορά οντότητας (Entiry Reference)	Όνομα entiry referenced	κενό (null)	κενό (null)
6	Δήλωση Οντότητας στο DTD (Eentiry)	Όνομα entiry	κενό (null)	κενό (null)
7	Εντολή Επεξεργασίας (Processing Instruction)	Target	Περιεχόμενο που αφορά τον στόχο	κενό (null)
8	Σχόλιο (Comment)	#comment	Περιεχόμενο που αφορά το	κενό (null)

			σχόλιο	
9	Έγγραφο (Document)	#document	κενό (null)	κενό (null)
10	Τύπος Εγγράφου (Document Type)	Όνομα document type	κενό (null)	κενό (null)
11	Απόσπασμα Εγγράφου(Document Fragment)	#document-fragment	κενό (null)	κενό (null)
12	Δήλωση Σήμανσης DTD (Notation)	Όνομα-notation	κενό (null)	κενό (null)

Όλοι οι ανωτέρω τύποι κόμβων αποτελούν αντικείμενα DOM, αφού θεωρούνται συστατικά που μπορούν να περιέχονται σε έγγραφα HTML και XML. Η ιδιότητα Attr, δηλώνει κόμβο-ιδιότητα και αναπαριστά μία ιδιότητα ενός κόμβου-στοιχείου (element). Δεν είναι κόμβος-παιδί (child) του κόμβου-στοιχείου (element) που περιγράφει και για τον λόγο αυτό το DOM δεν την θεωρεί μέρος της δομής του δέντρου του εγγράφου (document tree). Έτσι οι ιδιότητες που αφορούν τις οικογενειακές σχέσεις κόμβων (parentNode, previousSibling, nextSibling) δεν έχουν εφαρμογή σε κόμβους-Attr και επιστρέφουν σαν τιμή κενό (null).

Η εντολή επεξεργασίας (processing instruction), μπορεί να την τοποθετεί σε οποιοδήποτε σημείο του εγγράφου αρκεί να μην βρίσκεται εντός άλλης σήμανσης. Η σήμανση σε έγγραφα XML, αρχίζει με <? και κλείνει με ?>.

```
<? Xml-stylesheet type=”text/css” href=”books.css” ?>
```

Η ανωτέρω εντολή επεξεργασίας λέει στον browser να χρησιμοποιήσει το φύλλο επάλληλων στυλ που βρίσκεται στο αρχείο “books.css”

Η ενότητα CDATA, δημιουργεί σήμανση σε έγγραφα XML με την ετικέτα

<![CDATA[Με την πληκτρολόγηση οποιουδήποτε περιεχομένου και χαρακτήρων εκτός των δύο δεξιών αγκύλων και του συμβόλου “μεγαλύτερο από”]]>

Η Οντότητα (Entity) μπορεί να είναι ένα εξωτερικό αρχείο του οποίου τα δεδομένα θέλουμε να συνδέσουμε με το έγγραφό μας, είτε να είναι ένα μπλοκ κειμένου που ορίζουμε εμείς σαν οντότητα επειδή το χρησιμοποιούμε συχνα-πυκνά μέσα στο έγγραφο.

```
<! ENTITY title “Εισαγωγή στην XML “>
```

3.4 DOM Ιδιότητες-Μέθοδοι

3.4.1 Βασικές Ιδιότητες Κόμβων

Οι ιδιότητες των κόμβων DOM, αναφέρονται και έχουν εφαρμογή στα αντίστοιχα στοιχεία-αντικειμένου του εγγράφου. Κάθε κόμβος μπορεί να

προσπελαστεί μέσω της JavaScript. Όπως στην JavaScript, όσον αφορά την σύνταξη των αντικειμένων, των ιδιοτήτων και των μεθόδων τους έχουμε ευαισθησία στην διάκριση μεταξύ πεζών και κεφαλαίων χαρακτήρων, έτσι και στο μοντέλο DOM έχουμε την ίδια ευαισθησία στην διάκριση. Οι βασικές ιδιότητες που αφορούν κάθε κόμβο στην δομή του DOM είναι οι ακόλουθες :

nodeType

Δίνει ένα κωδικό αριθμό που αναπαριστά τον τύπο του αντικειμένου-κόμβου (π.χ. 1=κόμβος στοιχείου-element, 3=κόμβος κειμένου-text, 9=κόμβος εγγράφου-documemt)

```
Typoskomvou = Node.nodeType;
```

nodeName

Δίνει το όνομα του κόμβου και όχι το αναγνωριστικό όνομα id (π.χ. για τους κόμβους στοιχείων το όνομα της ετικέτας τους –tag name, όπως p, body, form, για τους κόμβους εγγράφων το κωδικό όνομα #document, για τους κόμβους κειμένων το κωδικό όνομα #text).

```
NameKomvou = Element.nodeName;
```

nodeValue

Δίνει το περιεχόμενο κείμενο των κόμβων κειμένου, null να δεν έχει τιμή (δεν έχει εφαρμογή σε κόμβους που δεν σχετίζονται με κείμενο).

```
AttributeAxia = Atrribute.nodeValue;
  ElementAxia = Element.nodeValue
```

;

3.4.2. Βασική ιδιότητα attributes

Η βασική ιδιότητα attributes, δημιουργεί ένα αντικείμενο λίστα πίνακα (NamedNodeMap) με όλους τους κόμβους-ιδιοτήτων (attribute nodes) του συγκεκριμένου κόμβου)

Attributes

```
NamedNodeMap = Document.documentElement.clildNodes
                    (0).attributes
```

Με την ανωτέρω πρόταση JavaScript, δημιουργούμε την μεταβλητή πίνακα NamedNodeMap, στην οποία εκχωρούμε και συλλέγουμε όλες τις ιδιότητες του πρώτου κόμβου που περιέχεται στο πρώτο πεδίο (0), του πίνακα childNodes() των θυγατρικών κόμβων του βασικού στοιχείου (root element) του εγγράφου, documentElement. Όλες οι ιδιότητες του συγκεκριμένου κόμβου, συλλέγονται στην NamedNodeMap, σαν κόμβοι ιδιότητες (κωδικός τύπου κόμβου-nodeType=2).

Με την ιδιότητα length του αντικειμένου NamedNodeMap, έχουμε τον αριθμό των κόμβων-ιδιοτήτων που περιέχει ο πίνακας.

```
AttrNumber=Element.attributes.length;
```

Μπορούμε χρησιμοποιώντας την ιδιότητα length, να δημιουργήσουμε ένα βρόχο-for και να προσπελάσουμε όλους τους κόμβους της συλλογής attributes:

```
     NamedNodeMap                                                    =
Document.documentElement.childNodes(0).attributes;

     For (i=0; i<NamedNodeMap; i++)
     {
          alert     ("     Όνομα     ιδιότητας     Κόμβου:
"+NamedNodeMAp(i).nodeName +"\n"+ "τιμή ιδιότητας : "+
NamedNodeMap(i).nodeValue);
     }
```

Με την μέθοδο reset(), τοποθετείται ο δείκτης στον κόμβο που βρίσκεται πριν από τον πρώτο κόμβο της `NamedNodeMap`, έτσι ώστε η κλίση με την μέθοδο `nextNode()` να επιστρέφει τον πρώτο κόμβο της συλλογής.

```
Element.attributes.reset();
```

Με την μέθοδο `nextNode()`, επιστρέφει τον επόμενο κόμβο ιδιότητας της συλλογής από αυτόν στον οποίο είναι τοποθετημένος ο δείκτης.

```
     Element.attributes.reset();
FirstAttribute=Element.attributes.nextNode();
```

Με την μέθοδο `getNamedItem`(όνομα κόμβου-ιδιότητας), επιστρέφει τον κόμβο ιδιότητα με το συγκεκριμένο όνομα.

```
AttrKomvos = Element.attributes.getNamedItem ("AttributeName");
```

3.4.3 Η βασική ιδιότητα text

Η βασική ιδιότητα text, δίνει το πλήρες περιεχόμενο του συγκεκριμένου κόμβου και όλων των θυγατρικών κόμβων Element που περιέχει.

Text

```
AllText = Element.text;
```

Με την ιδιότητα length παίρνουμε τον αριθμό των χαρακτήρων του κειμένου που περιέχει ο κόμβος.

```
CharNumber = Text.length;
```

Οι μέθοδοι που αναφέρουμε αμέσως μετά έχουν την εφαρμογή μόνο σε κόμβους-κειμένου (text-nodes)

Με την μέθοδο `splitText()`
χωρίζουμε το περιεχόμενο ενός κόμβου-κειμένου σε δύο μέρη
Αφήνουμε ένα μέρος σαν περιεχόμενο στον τρέχοντα κόμβο και εκχωρούμε το υπόλοιπο μέρος σε μία νέα μεταβλητή string που δημιουργείται αυτόματα.

```
newSubText = komvosText.splitText(8);
```

Αν το περιεχόμενο του κόμβου `komvosText = "W3C-DOM` Το πρότυπο του Web"
Η αριθμητική παράμετρος , δηλώνει πάντα ένα ακέραιο αριθμό (integer), ο οποίος αφορά τον αριθμό των χαρακτήρων που θα παραμείνουν σαν περιεχόμενο στον τρέχοντα κόμβο.
Στο παράδειγμά μας μετά την εκτέλεση της μεθόδου-συνάρτησης θα έχουμε :

```
komvosText = "W3C-DOM " και
newSubText = "Το πρότυπο του Web"
```

Με την μέθοδο `insertData()`

παρεμβάλλουμε κείμενο ή αλφαριθμητική μεταβλητή εντός συγκεκριμένου σημείου στο περιεχόμενο ενός κόμβου-κειμένου (text node).

```
komvosText.insertData(8, "* hello DOM *");
```

Η αριθμητική παράμετρος , δηλώνει πάντα ένα ακέραιο αριθμό (integer), ο οποίος λειτουργεί σαν δείκτης (pointer) και δείχνει από ποιο χαρακτήρα και μετά του υπάρχοντος περιεχομένου θα προστεθεί το νέο περιεχόμενο string ή της αλφαριθμητικής μεταβλητής που εμφανίζεται σαν δεύτερη παράμετρος της μεθόδου.
Στο ανωτέρω παράδειγμα μετά την εφαρμογή της μεθόδου θα έχουμε :

```
komvosText = "W3C-DOM * hello DOM * Το πρότυπο του Web"
```

Με την μέθοδο `appendData()`,
προσθέτουμε κείμενο με μορφή αλφαριθμητικής σταθεράς (literal) ή με την δήλωση κάποιας μεταβλητής string, τέλος του περιεχομένου ενός συγκεκριμένου κόμβου-κειμένου (text node).

```
komvosText.appendData(" είναι το μέλλον !");
```

Η παράμετρός θα μπορούσε να περιέχει κάποια μεταβλητή string.
Μετά την εφαρμογή της μεθόδου επί του κόμβου komvosText θα έχουμε :

```
komvosText = "W3C-DOM Το πρότυπο του Web είναι το μέλλον ! "
```

Με την μέθοδο `replaceData()`,
έχουμε την διπλή δυνατότητα αφ΄ ενός να αφαιρέσουμε ένα τμήμα από το περιεχόμενο ενός κόμβου-κειμένου και αφ΄ετέρου στην θέση του να προσθέσουμε ένα νέο κείμενο ανεξαρτήτως μεγέθους.
Το νέο κείμενο-string ουσιαστικά αντικαθιστά το αφαιρούμενο τμήμα καλύπτοντας το κενό.

```
komvosText.replaceData(8, 18, ", είναι το τέλειο ! ");
```

Η πρώτη αριθμητική παράμετρος, δηλώνει πάντοτε έναν ακέραιο αριθμό (integer), ο οποίος λειτουργεί σαν δείκτης (pointer) και δείχνει από ποιο χαρακτήρα και μετά του υπάρχοντος περιεχομένου θα αρχίσει η διαγραφή και για όσους χαρακτήρες δηλώνει η δεύτερη αριθμητική παράμετρός που και αυτή θα πρέπει πάντα να είναι ένας ακέραιος αριθμός (integer).

Η τρίτη παράμετρος είναι κάποια αλφαριθμητική σταθερά (literal) ή κάποια μεταβλητή string, που θα αντικαταστήσει το αφαιρούμενο τμήμα.
Εάν στο παράδειγμα εφαρμογής της μεθόδου, ο κόμβος
```
komvosText = "W3C-DOM  Το πρότυπο του Web"
```

μετά την εφαρμογή της, θα έχουμε :
```
komvosText="W3C-DOM,είναι το τέλειο !"
```

Με την μέθοδο `deleteData()`,
διαγράφουμε συγκεκριμένο μέρος (substring) του περιεχομένου ενός κόμβου-κειμένου, ορίζοντας το σύνολο των χαρακτήρων που πρόκειται να διαγραφούν με αριθμητικές παραμέτρους - δείκτες.
```
komvosText.deleteData(8,23);
```

Η πρώτη αριθμητική παράμετρος, δηλώνει πάντοτε έναν ακέραιο αριθμό (integer), ο οποίος λειτουργεί σαν δείκτης (pointer) και μας δείχνει από ποιο χαρακτήρα και μετά του υπάρχοντος περιεχομένου θα αρχίσει η διαγραφή.
Η δεύτερη αριθμητική παράμετρος, που και αυτή θα πρέπει πάντα να είναι ένας ακέραιος αριθμό (integer), δηλώνει το σύνολο των χαρακτήρων που θα διαγραφούν σειριακά με πρώτο τον αμέσως επόμενο από εκείνον που δείχνει η πρώτη παράμετρος (pointer).
Στο παράδειγμά μας, με τον κόμβο `komvosText` έχει περιεχόμενο,
`komvosText = "W3C-DOM` το μοντέλο συμβατότητας, είναι το μέλλον ! "
μετά την εφαρμογή της μεθόδου, θα έχουμε :
```
komvosText = "W3C-DOM , είναι το μέλλον ! "
```

3.4.4 Οι ιδιότητες σχέσεων κόμβων

Οι Ιδιότητες Σχέσεων Κόμβων DOM `parentNode & childNodes`
Οι Ιδιότητες Σχέσεων περιγράφουν τις σχέσεις που μπορεί να έχει ένας κόμβος με άλλους κόμβους μέσα στην δομή του δέντρου DOM.
Η ιεραρχία στην δομή DOM καθορίζει και την κληρονομικότητα των κόμβων, την μεταφορά δηλαδή των ιδιοτήτων τους
σε περιεχόμενους κόμβους κατωτέρου επιπέδου.
Εάν μία ιδιότητα δεν έχει εφαρμογή σε ένα συγκεκριμένο κόμβο παίρνει την τιμή null.
Σαν παράδειγμα μπορούμε να αναφέρουμε ένα κόμβο που δεν έχει θυγατρικούς κόμβους, κόμβους παιδιά, για τον κόμβο αυτόν η ιδιότητα `firstChild` θα είναι null.
Οι Ιδιότητες Σχέσεων είναι οι ακόλουθες :
`parentNode`
Δίνει τον γονέα (parent) του τρέχοντος κόμβου, εάν υπάρχει., δεν ισχύει για κόμβο attribute.

```
GoneasKomvos = Element.parentNode;
```
Η ιδιότητα `childNodes` δημιουργεί μία Λίστα-Πίνακα (array), που περιέχει όλους τους κόμβους - παιδιά ενός κόμβου.

Το αντικείμενο Λίστα-Πίνακα που δημιουργεί η ιδιότητα αυτή των κόμβων που έχουν θυγατρικά στοιχεία, είναι γνωστό σαν αντικείμενο NodeList.
childNodes
Με την ιδιότητα length, έχουμε τον αριθμό των θυγατρικών κόμβων που περιέχει η λίστα.

```
Kom.childNodes.length;
```

Μπορούμε χρησιμοποιώντας την ιδιότητα length, να δημιουργήσουμε ένα βρόχο-for και να προσπελάσουμε όλους τους θυγατρικούς κόμβους της λίστας :

```
for (i=0;i<element.childNodes.length;++i)
 {
    if (  element.childNodes(i).nodeType =  3 )



        text += element.childNodes(i).data;
    }
```

Με άμεση προσπέλαση στον πίνακα childNodes έχουμε την σύνταξη :
FirstKomvos = Kom.childNodes(0); // Δίνει τον πρώτο κόμβο παιδί, του Kom.
LastKomvos = Kom.childNodes(i-1); // Δίνει τον τελευταίο κόμβο παιδί, του Kom.

Με την μέθοδο reset(), τοποθετείται ο δείκτης στον κόμβο που βρίσκεται πριν από τον πρώτο κόμβο της childNodes, έτσι ώστε η κλίση με την μέθοδο nextNode( ) να επιστρέφει τον πρώτο κόμβο της συλλογής.

```
Kom.childNodes.reset();
```

Με την μέθοδο nextNode(), επιστρέφει τον επόμενο κόμβο της συλλογής των θυγατρικών κόμβων από αυτόν στον οποίο είναι τοποθετημένος ο εσωτερικός δείκτης.

```
Kom.childNodes.reset();
FirstKomvos = Kom.childNodes.nextNode();
```

Οι ιδιότητες firstChild, lastChild, previousSibling και nextSibling
Οι Ιδιότητες Σχέσεων Κόμβων,
childNodes , firstChild και lastChild έχουν εφαρμογή σε θυγατρικούς κόμβους με εξαίρεση τους θυγατρικούς κόμβους των Ιδιοτήτων (attributes) των κόμβων, για τις περιπτώσεις αυτές χρησιμοποιούμε την ιδιότητα attributes του κόμβου Element.

Με τις ιδιότητες previousSibling και nextSibling,

έχουμε την δυνατότητα να προσπελάσουμε οποιονδήποτε τύπο κόμβου ιδίου επιπέδου.

`firstChild`
Δίνει τον πρώτο κόμβο παιδί του τρέχοντος κόμβου.

```
FirstChildKomvos = Element.firstChild;
```

`lastChild`
Δίνει τον τελευταίο κόμβο παιδί του τρέχοντος κόμβου.

```
LastChildKomvos = Element.lastChild;
```

`previousSibling`
Δίνει τον κόμβο που βρίσκεται στο ίδιο επίπεδο με τον τρέχοντα κόμβο και προηγείται αυτού.
(είναι ο προηγούμενος αδελφός κόμβος από τον τρέχοντα αδελφό κόμβο)

```
AdelfosKomvosPrevious = Element.previousSibling;
```

`nextSibling`
Δίνει τον κόμβο που βρίσκεται στο ίδιο επίπεδο με τον τρέχοντα κόμβο και έπεται αυτού.
(είναι ο επόμενος αδελφός κόμβος από τον τρέχοντα αδελφό κόμβο)

```
AdelfosKomvosNext = Element.nextSibling;
```

3.4.5 Γενικές μέθοδοι Σχέσεων κόμβων

Κάθε κόμβος document,
διαθέτει μεθόδους προσπέλασης των αντικειμένων - κόμβων που περιέχει είτε με αναφορά στις ιδιότητές τους (attributes), είτε με αναφορά στο όνομα της ετικέτας (tag name) του συγκεκριμένου κόμβου.
Η θέση του κόμβου μέσα στην δομή του μοντέλου DOM, μας επιτρέπει και την χρήση της ανάλογης μεθόδου, δηλαδή της μεθόδου που μπορεί να εφαρμοστεί για τον τρέχοντα κόμβο στην συγκεκριμένη θέση.
Η θέση του κόμβου, καθορίζει την σχέση του ως προς τους άλλους κόμβους του δέντρου της ιεραρχίας των αντικειμένων και κατά συνέπεια τον τρόπο που επικοινωνεί με αυτούς.

`appendChild(child)`
Προσθέτει το νέο κόμβο παιδί στο τέλος της λίστας των υπαρχόντων κόμβων παιδιών.

`insertBefore(child,beforechild)`
Προσθέτει το νέο κόμβο παιδί με το όνομα `child`, πριν από το υπάρχον κόμβο παιδί, με το όνομα `beforechild` .

`replaceChild(child,old)`
Αντικαθιστά με το νέο κόμβο παιδί με το όνομα child, τον κόμβο παιδί με το όνομα old.

`removeChild(child)`

Αφαιρεί τον κόμβο παιδί με το όνομα child, από το σύνολο των κόμβων παιδιών του τρέχοντος κόμβου.

`cloneNode(deep)`

Δημιουργεί ένα αντίγραφο κόμβο του τρέχοντος κόμβου και εάν η τιμή της παραμέτρου deep είναι αληθής (true) δημιουργεί αντίγραφα και για όλα τα παιδιά του τρέχοντος κόμβου, εφ όσον έχει.

`hasChildNodes()`

Επιστρέφει τιμή αληθές (true) εάν ο κόμβος έχει κόμβους παιδιά και ψευδές (false) εάν δεν έχει κανένα.

`getNamedItem()`

Επιστρέφει τον συγκεκριμένο κόμβο με το καθορισμένο όνομα παραμέτρου.

`setNamedItem()`

Αποθέτει τον συγκεκριμένο κόμβο με το καθορισμένο όνομα παραμέτρου.

`removeNamedItem()`

Μετακινεί τον συγκεκριμένο κόμβο με το καθορισμένο όνομα παραμέτρου.

3.4.6 Μέθοδοι Ιδιοτήτων Κόμβων Element

Οι Μέθοδοι των Κόμβων τύπου Element,
επενεργούν επί των Ιδιοτήτων των κόμβων στοιχείων - τύπου Element είτε για προσθέσουν ή να αφαιρέσουν ιδιότητες. είτε για να επεξεργαστούν το περιεχόμενό τους.
Ας μην ξεχνάμε πως κάθε ιδιότητα κόμβου του εγγράφου, είναι μία μεταβλητή - αντικείμενο με συγκεκριμένο περιεχόμενο.
Δείτε αναλυτικά τις μεθόδους DOM, που μπορείτε να εφαρμόσετε σε ιδιότητες στοιχείων - Elements του εγγράφου.

`setAttribute(name,value)`

(επί των ιδιοτήτων των κόμβων στοιχείων)
Δημιουργεί μία ιδιότητα (attribute) με το όνομα name="born" για τον τρέχοντα κόμβο-element link και με περιεχόμενο την δεύτερη παράμετρο value='mydesmos'

```
link.setAttribute('born','mydesmos');
```

`setAttribute(attr)`

(επί των ιδιοτήτων των κόμβων στοιχείων)
Προσθέτει το born αντικείμενο-object σαν ιδιότητα στο τρέχον στοιχείο - Element.

```
AttrNew = Element.setAttribute ("born");
```

`getAttribute(name)`

(επί των ιδιοτήτων των κόμβων στοιχείων)
Επιστρέφει το περιεχόμενο της ιδιότητας με το όνομα born.

```
AttrAxia = Element.getAttribute ("born");
```

`removeAtrribute(name)`

(επί των ιδιοτήτων των κόμβων στοιχείων)
Αφαιρεί την ιδιότητα με το όνομα name από το κόμβο στοιχείο-Element.

```
AttrEnd = Element.removeAttribute ("born");
```

`setAttributeNode('name','value')`

(επί των ιδιοτήτων των κόμβων ιδιοτήτων)

Δημιουργεί ένα κόμβο ιδιότητας (attribute node) με το όνομα name και αποθηκεύει σε αυτόν την τιμή value.(Στο παράδειγμα, δημιουργούμε ένα νέο κόμβο-στοιχείο αγκύρωσης (a) και στην συνέχεια του προσδίδουμε την ιδιότητα href με τιμή-περιεχόμενο mypage.htm)

```
var link = document.createElement("a");
    link.setAttribute("href", "ypage.htm");
```

`getAttributeNode(name)`

(επί των ιδιοτήτων των κόμβων ιδιοτήτων)

Επιστρέφει τον κόμβο ιδιότητας born του στοιχείου με το όνομα Element

```
Attribute = Element.getAttributeNode ("born");
```

`removeAttributeNode(attr)`

(επί των ιδιοτήτων των κόμβων ιδιοτήτων)

Αφαιρεί την ιδιότητα με το όνομα born από τον τρέχοντα κόμβο ιδιότητα (attribute node) του στοιχείου Element.

```
AttrNodeEnd = Element.removeAttributeNode ("born");
```

3.5 DOM Σχέσεις Κόμβων

3.5.1 Σχέσεις & Διαχείριση Κόμβων XML

Κάθε κόμβος του HTML εγγράφου έχει ένα γονέα κόμβο (parent), εκτός του κόμβου εγγράφου (document node) που αφορά όλο το έγγραφο με όλα τα συστατικά του μέρη και είναι ουσιαστικά η ρίζα του δέντρου (root of the tree). Οι σχέσεις μεταξύ των κόμβων καθορίζονται από την οργάνωση και την σύνταξη των HTML στοιχείων (elements) καθώς και των κειμένων (text) που αυτά περιέχουν.Οι τύποι κόμβων που χρησιμοποιούνται περισσότερο στα έγγραφα HTML είναι οι κόμβοι - στοιχείων (node elements) και οι κόμβοι - κειμένου (node text). Κάθε κόμβος, μπορεί να μην έχει κανένα ή να έχει ένα και περισσότερα αδέλφια - κόμβους (siblings), που είναι κόμβοι που βρίσκονται στο ίδιο επίπεδο με αυτόν και έχουν τον ίδιο γονέα (parent), καθώς και κανένα ή ένα και περισσότερα παιδιά - κόμβους (children), κόμβους δηλαδή, που περιέχονται στον συγκεκριμένο τρέχοντα κόμβο.

Δείτε το έγγραφο dom10.html

```
<html>
  <head>
    <title> Αρχική Σελίδα DOM </title>
  </head>
  <body>
    <p id="par1">
       Το DOM είναι
       <b id="bold1"> το web πρότυπο</b>
       σύνταξης εγγράφων.
    </p>
  </body>
 </html>
```

Ο κόμβος html είναι ο βασικός κόμβος - στοιχείο (basic node element) του εγγράφου, χωρίς αδέλφια - κόμβους αλλά με δύο κόμβους - παιδιά, τους head και body.
Η σχέση των head και body είναι αδελφική, είναι δηλαδή κόμβοι - αδέλφια (siblings) αφού βρίσκονται στο ίδιο επίπεδο ιεραρχίας και έχουν τον ίδιο πατέρα - κόμβο (parent node), εν προκειμένω τον κόμβος html.
Ο κόμβος body περιέχει με την σειρά του ένα κόμβο - παιδί (child), τον κόμβο - στοιχείου p (element).
Ο κόμβος p περιέχει τρις κόμβους - παιδιά (children), εκ των οποίων μόνον ένας, ο κόμβος b είναι τύπου - στοιχείου (element node type) ενώ οι άλλοι δύο είναι τύπου - κειμένου (text nodes).

Ο κόμβος b περιέχει ένα κόμβο τύπου - κειμένου (text node).

3.5.2 Πίνακας Σχέσων Κόμβων DOM

Στον Πίνακα Σχέσων Κόμβων DOM,
που ακολουθεί, δείχνουμε τις σχέσεις μεταξύ των κόμβων με βάσει την δομή DOM του HTML εγγράφου του παραδειγματος.

```
<html>
  <head>
    <title> Αρχική Σελίδα DOM </title>
  </head>
  <body>
    <p id="par1">
       Το DOM είναι
       <b id="bold1"> το web πρότυπο</b>
       σύνταξης εγγράφων.
    </p>
  </body>
 </html>
```

Πίνακας 2) Εφαρμογής Ιδιοτήτων Σχέσεων Κόμβων DOM - Εγγράφου HTML

	Κόμβοι Εγγράφου				
Ιδιότητες DOM	document	HTML	title	p	b
NodeType	9	1	1	1	3
NodeName	#document	html	title	p	#text
NodeValue	κενό	κενό	κενό	κενό	"το web πρότυπο"
ParentNode	κενό	document	head	body	b
PreviousSibling	κενό	κενό	κενό	κενό	κενό
NextSibling	κενό	κενό	κενό	κενό	κενό

ChildNodes	html	head body	"Αρχική Σελίδα DOM"	"Το DOM είναι" b "σύνταξης εγγράφων"	κενό
FirstChild	html	head	"Αρχική Σελίδα DOM"	"Το DOM είναι"	κενό
LastChild	html	boby	"Αρχική Σελίδα DOM"	"σύνταξης εγγράφων"	κενό

Όταν ο IE 5, ανοίγει μία σελίδα HTML, δημιουργεί το δικό του αντικείμενο προγραμματισμού, γνωστό ως Πηγαίο Αντικείμενο Δεδομένων - DSO (Data Source Object) στο οποίο και αποθηκεύει ιεραρχικά τα στοιχεία και παρέχει πρόσβαση σε αυτά.

Το DSO, αποθηκεύει τα στοιχεία της HTML σελίδας, ως ένα σύνολο εγγραφών (recordset), μία συλλογή (collection) δηλαδή από εγγραφές μαζί με τα πεδία τους που αντιπροσωπεύουν τις ιδιότητες τους.
Όμως ο IE 5, όσον αφορά την συμβατότητά του με την προδιαγραφή του DOM, δεν θεωρεί το ίδιο έγγραφο HTML σαν κόμβο (document node) στην δομή δέντρου του εγγράφου.
Κατά συνέπεια οι ιδιότητες DOM που αναφέρονται στον κόμβο document, του ανωτέρω πίνακα (5), δεν έχουν εφαρμογή στον IE 5 και επιστρέφουν απροσδιοριστία (undefined), εκτός της ιδιότητας document.childNodes.
Η ιδιότητα nodeValue εφαρμόζεται επί των κόμβων-κειμένου (text nodes) και επιστρέφει αυτό καθ΄ αυτό το κείμενο που περιέχουν.

Η εφαρμογή της σε κόμβους-στοιχείων (element nodes), έχει σαν αποτέλεσμα επιστροφή τιμής πάντοτε το κενό (null).
Οι ιδιότητες innerHTML,innerText, outerHTML και outerText
είναι ιδιότητες του Internet Explorer και δεν αποτελούν μέρος του DOM.
Η τροποποίηση των περιεχομένων (contents) των κόμβων-στοιχείων HTML (element nodes) γίνεται με τις μεθόδους των κόμβων-αντικειμένων DOM.

3.5.3 Η Δημιουργία Νέου Κόμβου

Η δημιουργία ενός νέου κόμβου,
στην δομή DOM του εγγράφου HTML, γίνεται με την μέθοδο createElement("tag") του αντικειμένου document .
Το string που δηλώνεται σαν παράμετρος περιέχει την ετικέτα του στοιχείου που θέλουμε να δημιουργήσουμε και στην συνέχεια να προσθέσουμε στην δομή DOM του εγγράφου.

Η σύνταξη της δημιουργίας νέου κόμβου είναι :

```
varNewNode = document.createElement(“P”);
```

Η ανωτέρω πρόταση δημιουργεί ένα νέο αντικείμενο με το όνομα newNode στην μνήμη του Η/Υ χωρίς αυτό να περνά ακόμα στην δομή DOM του εγγράφου HTML και να αποκτά σχέσεις θυγατρικές, αδελφικές και γονικές με άλλους κόμβους της ιεραρχίας του κομβικού δέντρου.
Ακόμα, στο αντικείμενο - κόμβος newNode, δεν έχουμε προσδώσει καμία ιδιότητα (attribute) καθώς και κανένα περιεχόμενο, ενέργειες που ακολουθούν αμέσως μετά.

3.5.4 Προσθήκη ιδιότητας id στον Νέο Κόμβο

Υπάρχουν δύο τρόποι ενσωμάτωσης της ιδιότητας id σε κόμβους DOM,

- Ο απλός τρόπος της εκχώρησης αναγνωριστικού ονόματος στην ιδιότητα id του κόμβου - αντικειμένου :

```
newNode.id = "neoPar";
```

- και ο τρόπος μέσω της μεθόδου setAttribute() επί των στοιχείων - κόμβων DOM :

```
newNode.setAttribute("id","neoPar");
```

Και στις δύο ανωτέρω περιπτώσεις για το αντικείμενο - κόμβο newNode, καθορίσαμε την ιδιότητα id, με το αναγνωριστικό όνομα "neoPar",που σημαίνει ότι μέσω ενός script, στην συνέχεια θα μπορούμε να έχουμε άμεση προσπέλαση στον κόμβο αυτό με αναφορά στο αναγνωριστικό του όνομα "neoPar".

3.5.5 Περιεχόμενο για τον Νέο Κόμβο

Η δημιουργία περιεχομένου κειμένου (text),
εντός του νέου κόμβου newNode, απαιτεί την δημιουργία ενός κόμβου - κειμένου (TextNode) στον οποίο θα εκχωρηθεί το επιθυμητό κείμενο.
Αυτό γίνεται μέσω της μεθόδου createTextNode() του κόμβου αντικειμένου document, η οποία δημιουργεί ένα κόμβο που θα περιέχει κείμενο (text), με την σύνταξη που ακολουθεί :
δημιουργία ενός κόμβου - κειμένου (TextNode)

```
var newText=document.createTextNode();
```

εκχώρηση του κειμένου στην ιδιότητα περιεχόμενο - κόμβου (NodeValue)

```
newText.NodeValue="Το DOM είναι ασυναγώνιστο ..." ;
```

Ή με την ταυτόχρονη δημιουργία κόμβου - κειμένου (TextNode) και εκχώρηση σε αυτόν της τιμής (NodeValue)

```
var newText=document.createTextNode("Το τέλειο DOM");
```

3.5.6 Ενσωμάτωση στο στοιχείο body

Το επίπεδο ιεραρχίας,
των κόμβων του εγγράφου (document), στο οποίο θέλουμε να προστεθεί ο νέος κόμβος καθορίζει και τις σχέσεις του νέου κόμβου τόσο με τους κόμβους του αμέσως ανώτερου επιπέδου, όσο και με τους κόμβους του αυτού επιπέδου καθώς και κατώτερων από αυτό.
Η μέθοδος ενσωμάτωσης είναι η appendChild() και το επίπεδο ιεραρχίας στο οποίο θα ενσωματώσουμε τον νέο κόμβο είναι εντός του σώματος του εγγράφου, εντός δηλαδή της

```
<body> ... </body>
```

Αυτό σημαίνει πως ο νέος κόμβος newNode, θα έχει γονέα-κόμβο (parent node) τον κόμβο <body> και αδελφό-κόμβο (sibling node) τον κόμβο

```
<p id="par1">... </p>
```

Η σύνταξη της ενσωμάτωσης εντός του αντικείμενου - κόμβου είναι :

```
document.body.appendChild(newNode)
```

Η διαδρομή της δομής της ιεραρχίας DOM του εγγράφου πρέπει να ακολουθείται σχολαστικά, έτσι ώστε να προστίθεται το νέο αντικείμενο-κόμβος σαν θυγατρικός κόμβος εντός του συγκεκριμένου κόμβου που στοχεύουμε.
Η αναπαράσταση εμπεριεχομένων επιπέδων (nested) , στο οικογενειακό δέντρο (family tree) των κόμβων μετά την ενσωμάτωση και του νέου κόμβου newNode, θα είναι :
document

```
<html>
  <head>
     < title>
        "Αρχική Σελίδα DOM"
  <body>
     <p id="par1">
        "Το DOM είναι"
           <b id="bold1">
               "το web πρότυπο"
       "σύνταξης εγγράφων"
          <p id="neoPar">
          "Το DOM είναι ασυναγώνιστο ... "
```

3.5.7 Αντικατάσταση ενός κόμβου με νέο

Η αντικατάσταση ενός κόμβου,

στην δομή DOM του εγγράφου γίνεται με την μέθοδο replaceChild(node).
Πριν φυσικά εφαρμόσουμε την μέθοδο θα πρέπει να προσπελάσουμε τον κόμβο που θέλουμε να αντικαταστήσουμε,

- είτε με άμεση προσπέλαση με την μέθοδο λήψης κόμβου με το αναγνωριστικό του όνομα,

```
document.getElementById("N1").replaceChild(newNode,old)
```

- είτε με τον προσδιορισμό του μέσω της ιεραρχικής διαδρομής των κόμβων της δομής DOM,

```
document.body.N1.replaceChild(newNode,old)
```

Και στις δύο περιπτώσεις πρέπει πρώτα να έχουμε δημιουργήσει τον νέο κόμβο newNode και να έχουμε προσδιορίσει την θέση του προς αντικατάσταση κόμβου old στην δομή της ιεραρχίας.

Αυτό γίνεται με την δημιουργία των παραμέτρων - μεταβλητών, με τις προτάσεις :

```
Var newNode=document.createTextNode("Το τέλειο DOM")
var old=document.getElementById("N1").childNodes[0]
```

Με την εφαρμογή της μεθόδου replaceChild(newNode, old), η πρώτη μεταβλητή - παράμετρος newNode δημιουργεί ένα κόμβο-κειμένου και του εκχωρεί ταυτόχρονα και την τιμή του string "Το τέλειο DOM", ενώ η δεύτερη μεταβλητή - παράμετρος old προσπελάζει και φέρνει στην μνήμη τον κόμβο που πρόκειται να αντικατασταθεί από τον newNode .
Υπάρχει ακόμα και η πλέον απλή τεχνική της αντικατάστασης ενός κόμβου-κειμένου (node text) που περιέχεται εντός κάποιου άλλου κόμβου της δομής του δέντρου, με εκχώρηση μίας νέας αλφαριθμητικής τιμής string, στην ιδιότητα nodeValue του συγκεκριμένου κόμβο, όπως η πρόταση

```
document.getElementById("N1").childNodes[0].nodeValue =
                    "Το τέλειο DOM"
```

Υπενθυμίζουμε πως η ιδιότητα nodeValue, έχει εφαρμογή μόνο επί κόμβων κειμένου (text nodes).

3.5.8 Διαγραφή ενός υπάρχοντος Κόμβου

Η διαγραφή ενός κόμβου,
από την δομή DOM του εγγράφου, γίνεται με την μέθοδο removeChild(Node).
Εφαρμόζουμε την μέθοδο της διαγραφής για τον κόμβο που θέλουμε να διαγράψουμε, αφού πρώτα τον προσπελάσουμε και τον εκχωρήσουμε σε μία βοηθητική μεταβλητή στην μνήμη, όπως :

```
var old=document.getElementById("neoPar").childNodes[0]
```

Στην συνέχεια εφαρμόζουμε για την συγκεκριμένη μεταβλητή την μέθοδο της διαγραφής removeChild(old).:

- είτε με άμεση προσπέλαση κόμβου με το αναγνωριστικό του όνομα,

```
document.getElementById("neoPar").removeChild(old)
```

- είτε με τον προσδιορισμό του κόμβου μέσω της ιεραρχικής διαδρομής των κόμβων της δομής DOM,

```
document.body.neoPar.removeChild(old)
```

3.6 DOM Συμβάντα

3.6.1 Συμβάντα & Χειριστές Συμβάντων DOM

Συμβάντα (events) και Χειριστές Συμβάντων (event handlers)

Ένα συμβάν (event)
είναι η πραγματοποίηση ενός γεγονότος που πυροδοτήθηκε και έλαβε χώρα μέσα από κάποια ενέργεια του χρήστη, όπως για παράδειγμα το κλικ με το ποντίκι του πάνω σε μία εικόνα - σύνδεσμο (image link) ή σ΄ ένα κουμπί (button) του εγγράφου

```
<a id="a1"  href="http:www.w3c.org">
    Με click δημιουργείτε ένα συμβάν !
 </a>
```

Ακόμα ένα συμβάν μπορεί να προγραμματιστεί να λάβει χώρα μέσα από τον κώδικα του εγγράφου και σε συγκεκριμένο σημείο του, χωρίς καμία ενέργεια από την μεριά του χρήστη.

Ο στόχος ενός συμβάντος (event target)
είναι το αντικείμενο - κόμβος του εγγράφου επί του οποίου πρόκειται να λάβει χώρα το συμβάν. Το DOM, όπως ήδη γνωρίζουμε, τα στοιχεία (elements) των HTML και XML εγγράφων, τα μετατρέπει σε αντικείμενα και αυτά σε κόμβους με οργάνωση την ιεραρχική δομή του οικογενειακού δέντρου (family tree).
Το αντικείμενο στόχος του συμβάντος click του προηγούμενου παραδείγματος είναι ο κόμβος κειμένου "Κάνοντας click δημιουργείτε ένα συμβάν ..."

Ένας χειριστής συμβάντος (event handler)
είναι μία συνάρτηση σεναρίου (script) σε κώδικα που γράφουμε σε JavaScript, με σκοπό να εκτελεστεί σαν απόκριση-απάντηση σε συγκεκριμένο συμβάν (event), όταν ο χρήστης με ενέργειά του το κάνει να λάβει χώρα, δηλαδή να πραγματοποιηθεί.
Επειδή το συμβάν click, είναι συμβάν που δημιουργείται με το στοιχείο-αντικείμενο <a>, εντός της ετικέτας του συγκεκριμένου στοιχείου <a> θα

προσθέσουμε σαν ιδιότητά του τον αντίστοιχο χειριστή συμβάντος onclick, εφ΄ όσον θέλουμε να υπάρχει απόκριση όταν θα λαμβάνει χώρα το συμβάν, ήτοι :

```
<a id="a1"  href="http:www.w3c.org"
    onclick = "alert('Η σύνδεση σας οδηγεί στον W3C')">
    Κάνοντας click δημιουργείτε ένα συμβάν ...
 </a>
```

Συμβάν Αντικείμενο (event object)
είναι το ειδικό αντικείμενο (event object) που δημιουργείται με κάποια ενέργεια του χρήστη πάνω στο έγγραφο και το οποίο κρατά μέσα από τις ιδιότητές του πληροφορίες για το συμβάν αυτό.
Στο ανωτέρω παράδειγμα το αντικείμενο με το αναγνωριστικό όνομα "a1" είναι το αντικείμενο-κόμβος του οποίου ο θυγατρικός κόμβος "Κάνοντας click δημιουργείτε ένα συμβάν ..." που είναι κόμβος κειμένου (node text) είναι το αντικείμενο στόχος (event target) του συμβάντος click.
Κάθε συμβάν που δημιουργείται, αυτόματα ορίζεται σαν συμβάν αντικείμενο (event object) με το όνομα e.

3.6.2 Ο Κύκλος Ζωής ενός Συμβάντος

Ο κύκλος ζωής ενός συμβάντος (event's life cycle)
αρχίζει με την ενέργεια, είτε εξωγενώς από τον χρήστη, είτε ενδογενώς από τον κώδικα του εγγράφου και τελειώνει με το κάλεσμα και την εκτέλεση του script μέσα από τον αντίστοιχο με το συμβάν, χειριστή του (event handler).

1. Η ενέργεια του χρήστη ή η συνδεδεμένη συνθήκη με το συμβάν, το πυροδοτούν.

2. Το αντικείμενο event αυτόματα ενημερώνεται με όλες τις πληροφορίες που αφορούν το συμβάν.

3. Το συμβάν λαμβάνει χώρα.

4. Ο χειριστής συμβάντος που συνδέεται με το στοιχείο-αντικείμενο επί του οποίου λαμβάνει χώρα το συμβάν αποκρίνεται εκτελώντας κάποια συνάρτηση JavaScript.

5. Το συμβάν περνά στο στοιχείο-αντικείμενο του επόμενου επιπέδου ιεραρχίας και εφ΄ όσον υπάρχει και σε αυτό συνδεδεμένος χειριστής συμβάντος καλείται και εκτελείται.

   Το βήμα αυτό επαναλαμβάνεται για τον internet Explorer, μέχρι να φτάσει το συμβάν στο στοιχείο window που είναι ουσιαστικά το ανώτατο επίπεδο ιεραρχίας (browser window) και εκεί να συνεχίσει να αναπαράγεται κοχλάζοντας (bubblling),
   ενώ για τον Netscape επαναλαμβάνεται από το σημείο (spot) της πυροδότησης του πάνω στην επιφάνεια του εγγράφου - document σε κάθε επίπεδο της ιεραρχίας των αντικειμένων κατά την πορεία του προς το στοιχείο-

αντικείμενο στόχο με σκοπό την σύλληψή του (capture).

Η επάλληλη ενέργεια ενός συμβάντος στα επόμενα επίπεδα της ιεραρχίας του βήματος 5, λαμβάνει χώρα εξ΄ ορισμού (by default) εκτός και εάν οριστεί διαφορετικά ή με κάποιον χειριστή συμβάντος σε κάποιο από τα επίπεδα διακοπεί.

3.6.3 Συμβάντα & Χειριστές του DOM

Όλα τα στοιχεία της HTML δεν υποστηρίζουν όλα τα συμβάντα,
και κατ΄ ακολουθία και τους αντίστοιχους χειριστές του καταλόγου.
Τα συμβάντα διακρίνονται σε συμβάντα ποντικού (mouse events), που λαμβάνουν χώρα επί στοιχείων που δέχονται ενέργειες του χρήστη με το ποντίκι.

Σε συμβάντα πληκτρολόγιου (keyboards events), που λαμβάνουν χώρα επί στοιχείων που δέχονται ενέργειες του χρήστη με το πληκτρολόγιο.

Σε συμβάντα εστίασης και επιλογής (focus & selection events), που λαμβάνουν χώρα επί στοιχείων που δέχονται ενέργειες του χρήστη με το ποντίκι ή το πληκτρολόγιο και αφορούν την εστίαση σε κάποιο σημείο του εγγράφου ή την μετακίνηση κάποιου στοιχείου του σε άλλο μέρος.

Σε συμβάντα που αφορούν την φόρτωση και την απαξίωση (από - φόρτωση) του εγγράφου (load events).
Καθώς επίσης και σε συμβάντα κόμβων (node events), που λαμβάνουν χώρα επί στοιχείων που δέχονται ενέργειες του χρήστη με σκοπό την εισαγωγή ή διαγραφή κόμβων ή ιδιοτήτων τους από την δομή του εγγράφου DOM.
Υπενθυμίζουμε πως οι χειριστές συμβάντων (event handlers), εφαρμόζονται επί των αντικειμένων συμβάντων (event objects) σαν ιδιότητες (attributes) με την δυνατότητα σε ορισμένους από αυτούς να μπορούμε να αναγνώσουμε και να αλλάξουμε τα περιεχόμενά τους (read & write), ενώ σε άλλους μόνο να τα αναγνώσουμε (read only).

3.6.4 Πίνακας Συμβάντων & Χειριστών DOM2

Ολοκληρωμένος και πλήρως συμβατός με το μοντέλο συμβάντων του DOM Level 2,
είναι ο κατάλογος που ακολουθεί.
Χαρακτηρίζουμε με (read only) εκείνους τους χειριστές συμβάντων του DOM, που μόνο μας επιστρέφουν το υπάρχον περιεχόμενό τους.

Πίνακας 3) Συμβάντων και Χειριστών Συμβάντων DOM2		
Σ υ μ β ά ν	Χειριστής	Συμβάντος

• mousedown Πλήκτρο του ποντικού πατημένο	(ro)	onmousedown
• mouseup Απελευθέρωση πλήκτρου ποντικού	(ro)	onmouseup
• click Όταν γίνεται με το ποντίκι κλικ	(ro)	onclick
• dblclick Όταν γίνεται με το ποντίκι διπλό κλικ	(ro)	ondblclick
• mouseover Όταν μετακινείται ο δείκτης του ποντικού πάνω στο αντικείμενο - στόχο	(ro)	onmouseover
• mouseout Όταν ο δείκτης του ποντικού εγκαταλείπει το αντικείμενο - στόχο	(ro)	onmouseout
• mousemove Όταν ο δείκτης του ποντικού κινείται	(ro)	onmousemove
• contextmenu Όταν δημιουργείται μενού περιεχομένων		oncontextmenu
• keydown Όταν ένα πλήκτρο του πληκτρολογίου κρατείται πατημένο	(ro)	onkeydown
• keyup Όταν ένα πατημένο πλήκτρο απελευθερώνεται	(ro)	onkeyup
• keypress Όταν ένα πλήκτρο του πληκτρολογίου πατηθεί	(ro)	onkeypress

• focus Όταν η εστίαση τίθεται στο αντικείμενο - στόχο	(ro)	onfocus
• blur Όταν η εστίαση μετακινείται μακριά από το αντικείμενο - στόχο	(ro)	onblur
• load Όταν φορτώνεται το έγγραφο - document στο παράθυρο του browser - window		onload
• unload Όταν απο-φορτώνεται το έγγραφο - document από το παράθυρο του browser - window		onunload
• abort Όταν διακόπτεται μία ενέργεια		onabort
• error Όταν παρουσιάζεται ένα σφάλμα (error)		onerror
• submit Όταν αποστέλλεται μία φόρμα		onsubmit
• reset Όταν ακυρώνεται η αποστολή μίας φόρμας		onreset
• change Όταν η τιμή σε ένα στοιχείο της φόρμας αλλάζει		onchange
• select Όταν ένα στοιχείο έχει επιλεγεί		onselect
• input Όταν εισάγεται τιμή σε ένα στοιχείο φόρμας		oninput

• paint		onpaint
• text		ontext
• popupShowing		onpopupShowing
• popupShown		onpopupShown
• popupHiding		onpopupHiding
• popupHidden		onpopupHidden
• close Όταν το παράθυρο ενός πλαισίου (frame) κλείνει		onclose
• command Όταν το στοιχείο - αντικείμενο στόχος έχει υποστεί το συμβάν π.χ. έχει γίνει κλικ σε σύνδεσμο (clicked) ή έχει επιλεγεί κάποιο στοιχείο (selected)		oncommand
• broadcast		Onbroadcast
• commandupdate		Oncommandupdate
• dragenter		Ondragenter
• dragover Όταν κάποιο στοιχείο του εγγράφου σύρεται πάνω στο αντικείμενο - στόχο του συμβάντος		ondragover
• dragexit Όταν κάποιο στοιχείο του εγγράφου αποσύρεται πάνω από το αντικείμενο - στόχο του συμβάντος		Ondragexit
• dragdrop		ondragdrop

Όταν ένα στοιχείο του εγγράφου έχει συρθεί και αφεθεί πάνω στο αντικείμενο - στόχο του συμβάντος.		
• draggesture		Ondraggesture
• resize Όταν μετασχηματίζεται το παράθυρο του browser	(ro)	onresize
• scroll Όταν το παράθυρο ενός πλαισίου (frame) υφίσταται κύλιση.		onscroll
• overflow Όταν το κείμενο υπερχειλίζει το παράθυρο ενός πλαισίου (frame) σε σχέση με τις διαστάσεις του.		onoverflow
• underflow Όταν το κείμενο ενός πλαισίου (frame) περιέχεται σε αυτό χωρίς να το υπερχειλίζει.		onunderflow
• overflowchanged Όταν αλλάζουν οι διατάσεις ενός πλαισίου (frame) έτσι ώστε να χωρά σε αυτό το κείμενο που υπερχειλίζει.		onoverflowchanged
• subtreemodified Όταν τροποποιείται ένα τμήμα (subtree) της δομής του τρέχοντος εγγράφου που επεξεργάζεται ο browser.		onsubtreemodified
• nodeinserted Όταν ένας κόμβος εισέρχεται στην δομή του εγγράφου DOM		onnodeinserted
• noderemoved Όταν ένας κόμβος μετακινείται από την δομή του εγγράφου.		onnoderemoved

• noderemovedfromdocument Όταν ένας κόμβος έχει μετακινηθεί από την δομή του εγγράφου.		onnoderemovedfromdocument
• nodeinsertedintodocument Όταν ένας νέος κόμβος εισέρχεται στην δομή του εγγράφου DOM		onnodeinsertedintodocument
• attrmodified Όταν τροποποιείται ένας κόμβος ιδιότητας DOM		onattrmodified
• characterdatamodified Όταν δεδομένα χαρακτήρων τροποποιούνται		oncharacterdatamodified

3.6.5 Αντικείμενο event - Ιδιότητες & Μέθοδοι

Το αντικείμενο event (event object) στο DOM,
δημιουργείται κάθε φορά που ένα συμβάν λαμβάνει χώρα σε ένα web έγγραφο.

Το αντικείμενο αυτό κρατά πληροφορίες για το συγκεκριμένο συμβάν (event) που λαμβάνει χώρα, έτσι ώστε ο αντίστοιχος χειριστής συμβάντος (event handlers) που θα αποκριθεί με κάποιο κώδικα script σε αυτό, να μπορεί να τις μεταχειριστεί.

Οι πληροφορίες αυτές μπορεί να αφορούν, την θέση του δείκτη του ποντικού (mouse), την κατάσταση των πλήκτρων του πληκτρολογίου, το στοιχείο του HTML εγγράφου επί του οποίου έλαβε χώρα το συμβάν κ.λ.π.

Οι ιδιότητες του αντικειμένου event είναι :

1. type : καθορίζει ένα string που περιέχει τον τύπο του συμβάντος που λαμβάνει χώρα.

2. detail : επιστρέφει πληροφορίες για το συμβάν σχετικές με τον τύπο του.

3. target : είναι η αναφορά στο αντικείμενο, που στοχεύει και επί του οποίου θα λάβει χώρα το συμβάν.

4. currentTarget : είναι η αναφορά στο αντικείμενο, του οποίου ο ακροατής συμβάντος (event listener) καλεί προς εκτέλεση την συνάρτηση που το αφορά (listener's function).

Η αναφορά στο τρέχον αντικείμενο - στόχο με την εν λόγω ιδιότητα μας παρέχει πληροφορίες σχετικά με την θέση του συγκεκριμένου αντικειμένου στην ιεραρχία του εγγράφου.

5. relatedTarget : καθορίζει έναν δευτερεύον στόχο - αντικείμενο για το συγκεκριμένο συμβάν.

6. button : μας δίνει έναν αριθμό που δηλώνει εάν το κουμπί του ποντικού (mouse button) πατήθηκε (pressed) ή ελευθερώθηκε (released).

7. eventPhase :μας δηλώνει με έναν αριθμό την τρέχουσα φάση στην οποία βρίσκεται το συμβάν.

    - Με τιμή μηδέν (0), έχουμε την πραγματοποίηση του συμβάντος από έξω προς τα μέσα (capture)
    - Με τιμή ένα (1), έχουμε την στιγμή κατά την οποία το συμβάν βρίσκεται στο αντικείμενο - στόχο (target)
    - Με τιμή δύο (2), έχουμε την πραγματοποίηση του συμβάντος από μέσα προς τα έξω (bubbling)

8. cancelBubble : επιστρέφει τιμή μεταβλητής Boole (αληθές ή ψευδές) που καθορίζει εάν ή όχι το συμβάν θα συνεχίσει να λαμβάνει χώρα και στο επόμενο στοιχείο - αντικείμενο της ιεραρχίας του εγγράφου.
   Η προεπιλεγμένη τιμή, είναι συνήθως false, μολονότι κάποια συμβάντα (events) δεν συνεχίζουν επαναληπτικά και στα επόμενα στοιχεία του εγγράφου εξ΄ ορισμού.
   Στην περίπτωση που συνεχίζει το συγκεκριμένο συμβάν να λαμβάνει χώρα και στα επόμενα επίπεδα της ιεραρχίας και επιθυμούμε να σταματήσουμε το συμβάν, θα πρέπει με την εν λόγω ιδιότητα και με τιμή true μέσα τον ακροατή του συγκεκριμένου συμβάντος (event listener) να το δηλώσουμε.
   Έτσι η αλυσίδα των επάλληλων πραγματώσεων του συγκεκριμένου συμβάντος θα σταματήσει στο στοιχείο-αντικείμενο του οποίου ο ακροατής συμβάντος έκανε χρήση της ιδιότητας αυτής με τιμή true .

9. bubbles : επιστρέφει τιμή μεταβλητής Boole , όπου εκτός από την τιμή 1 και με οποιαδήποτε άλλη τιμή, δηλώνει για το συγκεκριμένο συμβάν εξ΄ ορισμού, συνεχή πραγμάτωση σε κάθε στοιχείο του επομένου επιπέδου ιεραρχίας.

10. cancelable - αφορά τιμή μεταβλητής Boole που δείχνει εάν το συμβάν μπορεί να διαγραφεί.
    Εκ του γεγονότος πως υπάρχουν συμβάντα που δεν διαγράφονται (events not cancelable).

11. keyCode : εάν το συμβάν έχει προέλθει από το πάτημα ενός πλήκτρου η εν λόγω ιδιότητα επιστρέφει τον αριθμητικό κώδικα για το πλήκτρο αυτό.

12. altKey : επιστέφει μία λογική τιμή boole, που μας δείχνει εάν το πλήκτρο

    <alt> ήταν πατημένο, όταν πυροδοτήθηκε και έλαβε χώρα το συμβάν.

13. shiftKey : επιστέφει μία λογική τιμή boole, που μας δείχνει εάν το πλήκτρο <shift> ήταν πατημένο, όταν πυροδοτήθηκε και έλαβε χώρα το συμβάν.

14. ctrlKey : επιστέφει μία λογική τιμή boole, που μας δείχνει εάν το πλήκτρο <ctrl> ήταν πατημένο, όταν πυροδοτήθηκε και έλαβε χώρα το συμβάν.

15. metaKey : επιστέφει μία λογική τιμή boole, που μας δείχνει εάν το κλειδί meta ήταν πατημένο, όταν πυροδοτήθηκε και έλαβε χώρα το συμβάν.

16. charCode : επιστρέφει έναν αριθμό που αντιπροσωπεύει τον χαρακτήρα που ήταν πατημένος, όταν ένα συμβάν πληκτρολογίου έλαβε χώρα.

17. isChar : επιστέφει μία λογική τιμή boole, που μας δείχνει εάν το συμβάν παράγει ένα χαρακτήρα πληκτρολογίου ή όχι.

18. pageX και pageY : επιστρέφει την οριζόντια (X) ή κάθετη (Y) συντεταγμένη του συμβάντος της ιστοσελίδας (page) - του html ή xml εγγράφου (document).

19. layerX και layerY : επιστρέφει την οριζόντια (X) ή κάθετη (Y) συντεταγμένη του συμβάντος που είναι σχετικό με την τρέχουσα θέση (current layer) του στοιχείου (element) επί του οποίου λαμβάνει χώρα το συμβάν.

20. clientX και clientY : επιστρέφει την οριζόντια (X) ή κάθετη (Y) συντεταγμένη του συμβάντος.

21. screenX και screenY : επιστρέφει την οριζόντια (X) ή κάθετη (Y) συντεταγμένη του συμβάντος της οθόνης.

22. timeStamp : επιστρέφει τον χρόνο στον οποίο δημιουργήθηκε το συμβάν.

Οι Μέθοδοι του αντικειμένου event είναι :

- initEvent : καθορίζει αρχική τιμή σ΄ ένα συμβάν που έχει δημιουργηθεί μέσω της διασύνδεσης - documentEvent
- initMouseEvent : Αρχικοποιεί (δίνει αρχική τιμή) ένα συμβάν ποντικού (mouse event) κατά την δημιουργία του.
- initUIEvent : Αρχικοποιεί ένα συμβάν UI (user interface) κατά την δημιουργία του.
- preventDefault : Διαγράφει ένα συμβάν, εάν η ιδιότητα cancelable επιστρέφει τιμή αληθές (true), που σημαίνει πως αυτό το συμβάν μπορεί

να διαγραφεί.

- StopPropagation : Σταματά την περαιτέρω αναπαραγωγή του συμβάντος εντός της δομής του DOM.

3.7 DOM Διασύνδεση

3.7.1 Η Διασύνδεση Συμβάντων με το DOM

Όταν ένα συμβάν λαμβάνει χώρα,
όπως ήδη έχουμε αναφέρει, δεν παραμένει στο ίδιο σημείο (spot) του εγγράφου που πυροδοτήθηκε, αλλά ταξιδεύει ιεραρχικά δια μέσω των στοιχείων του εγγράφου μέχρι, είτε να διαγραφεί από την δομή DOM, με την με μέθοδο preventDefault , είτε να σταματήσει η αναπαραγωγή του, με την μέθοδο StopPropagation.
Η κατεύθυνση της πορείας του συμβάντος, είτε από μέσα προς τα έξω (event bubbling), προς την επιφάνεια δηλαδή του εγγράφου, είτε από έξω προς τα μέσα (event capture), από την επιφάνεια δηλαδή του εγγράφου προς το αντικείμενο συμβάντος (event object), εξαρτάται από το μοντέλο συμβάντων που υποστηρίζει ο browser του χρήστη.

Η τυπική σύνταξη της διασύνδεσης ενός αντικειμένου συμβάντος με το DOM είναι :

```
objectEvent.onEvent = functionName
```

όπου,
objectEvent : το αντικείμενο επί του οποίου λαμβάνει χώρα το συμβάν
onEvent : ο αντίστοιχος χειριστής για το συγκεκριμένο συμβάν και
functionName : το όνομα της συνάρτησης που εκτελείται σαν απόκριση στο συμβάν.
Η ανωτέρω σύνταξη της σύστασης του μοντέλου συμβάντων του DOM, είναι αυθεντική αναφορά στην σύνταξη της JavaScript 1.2, με την οποία μπορούμε να προσδώσουμε σαν ιδιότητα σε ένα αντικείμενο συμβάντος (event object) ένα χειριστήριο συμβάντος (event handler) στο οποίο στην συνέχεια να εκχωρήσουμε σαν τιμή μία συνάρτηση (script function), η οποία και εκτελείται κάθε φορά που καλούμε το συγκεκριμένο αντικείμενο με την ιδιότητα αυτή.
Στις επόμενες ενότητες θα αναφερθούμε αναλυτικά στις δομές αυτές, της εγγραφής και της επεξεργασίας συμβάντων πάνω στα αντικείμενα - κόμβους της δομής του DOM.

3.7.2 Μοντέλο Συμβάντων της Microsoft

Στο μοντέλο συμβάντων του Internet Explorer,
το αντικείμενο event είναι ιδιότητα του αντικειμένου window και έχει το όνομα window.event

Από την στιγμή που πυροδοτείται και λαμβάνει χώρα ένα συμβάν (event) στον Internet Explorer, συνεχίζει να ενεργεί σε κάθε γονικό στοιχείο ανωτέρου ιεραρχικά επιπέδου μέχρι το ανώτατο ιεραρχικά επίπεδο της δομής των αντικειμένων που είναι το παράθυρο του IE (browser window).
Ο Internet Explorer, ακολουθώντας την ανωτέρω διαδρομή (path) καθιέρωσε την έννοια του event bubbling.
Κάθε συμβάν πυροδοτούμενο στο στοιχείο-αντικείμενο στόχο ταξιδεύει μέχρι την επιφάνεια του εγγράφου - document (browser window).
Διαδρομή event bubbling.
text ---> tagObject ---> document
Το μοντέλο συμβάντων του Internet Explorer, μας δίνει έτσι την δυνατότητα να σταματάμε και να συλλαμβάνουμε τα συμβάντα κατά την διάρκεια της πορείας τους προς την επιφάνεια του εγγράφου.
Φτάνοντας δε στην επιφάνεια του εγγράφου κάθε συμβάν - σαν μία φουσκάλα που κοχλάζει (bubbling), έχει ποιο άμεση σχέση με αυτό καθ΄ εαυτό το έγγραφο - document και το αντικείμενο window.event που περιέχει τις πληροφορίες που το αφορούν και που είναι χρήσιμες για την αντίστοιχη συνάρτηση χειρισμού του (event handler).
Δείτε το ακόλουθο παράδειγμα :
Στο απλό html έγγραφο που παραθέτουμε, έχουμε σαν στόχο - αντικείμενο για το συμβάν click το στοιχείο body.
Σε οποιοδήποτε σημείο λοιπόν της επιφάνειας του εγγράφου, όταν ο χρήστης κάνει με το ποντίκι του κλικ, ο χειριστής του αντίστοιχου συμβάντος onclick εντός της ετικέτας body, θα καλέσει και θα εκτελέσει την συνάρτηση της JavaScript με το όνομα event_life( ).
Η απόκριση της συνάρτησης είναι ένα μήνυμα συνοδευόμενο από τις τιμές του αντικειμένου event, για τον τύπο του συμβάντος και για το όνομα του στοιχείου επί του οποίου έλαβε χώρα το συμβάν.
Στην περίπτωση του IE, όπου αναφερόμαστε το αντικείμενο event, που δημιουργείται με κάθε κλικ του χρήστη το αντιπροσωπεύει η ιδιότητα window.event του αντικειμένου window.
Η ιδιότητα srcElement.tagName αφορά ιδιότητα του αντικειμένου window.event του μοντέλου συμβάντων του IE και δεν υποστηρίζεται από την σύσταση του DOM Event Model του W3C..

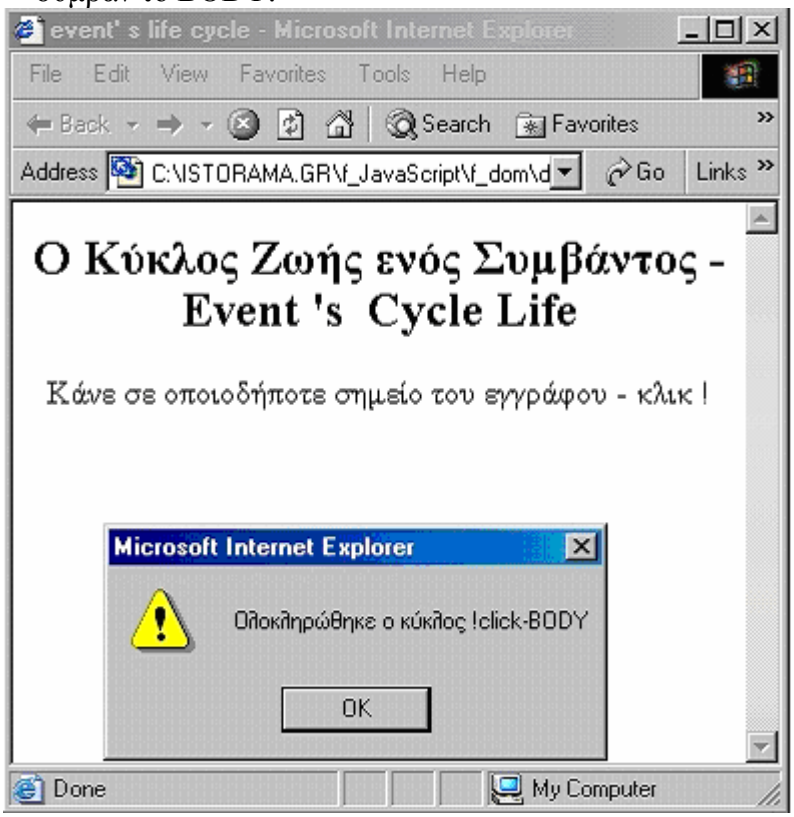
Έγγραφο : dom4_IE_event_model.html

Έγγραφο : dom4_IE_event_model.html

```
<html>
 <head>
  <title>event' s life cycle</title>
  <script languge="JavaScript">
  <!-- //
  function event_life()
  {
  alert("Ολοκληρώθηκε ο κύκλος !" + window.event.type + "-
" +
        window.event.srcElement.tagName);
```

```
  }
  // --></script>
</head>
<body onclick="event_life()">
  <center>
  <h2>Ο Κύκλος Ζωής ενός Συμβάντος - Event ' s Cycle Life
</h2>
  <p>Κάνε σε οποιοδήποτε σημείο του εγγράφου - κλικ !</p>
 </center>
</body>
</html>
```

Πατώντας το κουμπί της προεπισκόπησης μπορείτε να τρέξετε ζωντανά το πρόγραμμα με τον IE 5+.
Η απόκριση του browser θα πρέπει να σας δώσει την ακόλουθη οθόνη με τιμές, σαν τύπο συμβάντος το click και σαν στοιχείο επί του οποίου λαμβάνει χώρα το συμβάν το BODY.



Σχήμα 9) Ο κύκλος ζωής ενός Συμβάντος

3.7.3 Μοντέλο Συμβάντων της Netscape

Στο μοντέλο συμβάντων της Netscape,
το αντικείμενο event, δηλώνεται με τον πεζό χαρακτήρα e.

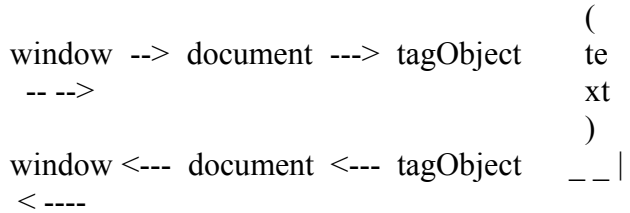
Με το μοντέλο συμβάντων της, η Netscape εισήγαγε την έννοια της σύλληψης συμβάντος (event capture), καθ΄ όλη την διάρκεια της πορείας του ταξιδιού του, από την στιγμή που αυτό λαμβάνει χώρα και ξεκινά την διαδρομή του, από έξω προς τα μέσα (inward), από την επιφάνεια δηλαδή του εγγράφου προς το αντικείμενο συμβάντος (event object).
Η διαδρομή της επάλληλης αναπαραγωγής συμβάντος στο μοντέλο Netscape, είναι ακριβώς η αντίθετη από αυτήν του IE.
Κάθε συμβάν πυροδοτούμενο, από την συγκεκριμένη θέση (spot) της επιφάνειας του εγγράφου (document - browser window) ταξιδεύει μέχρι το συγκεκριμένο στοιχείο-αντικείμενο (event object) που στοχεύει και επενεργεί επί του περιεχομένου του.
Διαδρομή event capture

document ---> tagObject ---> text

Ο browser της Netscape μας επιτρέπει να συλλάβουμε οποιοδήποτε συμβάν πέσει μέσα στο έγγραφο και αρχίσει πυροδοτούμενο να ταξιδεύει προς το στοιχείο - αντικείμενο στόχο (event object).
Στους browsers από την 6η γενιά και μετά η Netscape παρέχει τον συνδυασμό και των δύο μοντέλων συμβάντων όσον αφορά την εκκίνηση της διαδρομής τους και τέλος τους.

Αρχίζει δηλαδή να παρακολουθεί την πορεία ενός συμβάντος με την πυροδότηση του στο παράθυρο του browser μέχρι αυτό να φτάσει στο συγκεκριμένο στοιχείο-αντικείμενο στόχο (event capture) και στην συνέχεια συνεχίζει να παρακολουθεί την επιστροφή του μέσα από την ίδια διαδρομή μέχρι να φτάσει ξανά πίσω στο παράθυρο του browser και στο σημείο (spot) απ΄ όπου ξεκίνησε (event bubbling).

Διαδρομή event capture & bubbling



Το μοντέλο συμβάντων αυτό, που υποστηρίζεται από τους Netscape 6+ και Mozilla browsers, μας επιτρέπει την σύλληψη συμβάντων σε οποιαδήποτε φάση του κύκλου ζωής τους βρίσκονται.

Εύκολα μπορούμε να διακρίνουμε τις τρις φάσεις της εξέλιξης της ζωής ενός συμβάντος στο μοντέλο αυτό, ήτοι :

- Την φάση της πορείας του συμβάντος προς το στοιχείο-αντικείμενο στόχο (event capture)
- Την φάση, όπου το συμβάν έχει φτάσει στο στοιχείο-αντικείμενο στόχο και λαμβάνει χώρα.
- Την φάση, όπου το συμβάν επιστρέφει στο παράθυρο του browser ταξιδεύοντας προς τα πίσω (event bubble).

3.7.4 Μοντέλο Συμβάντων DOM του W3C

Η σύσταση του W3C-DOM μοντέλου συμβάντων,
πρωτοπαρουσιάστηκε με την ολοκλήρωση από τον W3C του DOM - Level 2, σαν μέρος του μαζί με το μέρος που αφορά την μορφοποίηση του εγγράφου με τα Επάλληλα Φύλα Στυλ το DOM CSS (Cascading Style sheets), τον Νοέμβριο του 2000.
Το μοντέλο αυτό έγινε με σκοπό την υποστήριξη των συμβάντων εκείνων που κυρίως προέρχονται από ενέργειες του χρήστη επί του εγγράφου (user events interface) αλλά και για τον χειρισμό των συμβάντων σε όλη την δομή της ιεραρχίας του εγγράφου (tree manipulation events).
Το αντικείμενο event του DOM, όπως ήδη αναφέραμε είναι η κύρια διασύνδεση (interface) με το μοντέλο συμβάντων του.
Οι δομές DOM διασύνδεουν τα αντικείμενα συμβάντα (event objects) με τα συμβάντα (events) που επενεργούν επί αυτών είναι :

- των χειριστών συμβάντων (event handlers)

```
objectEvent.onEvent  =  functionName
```

Στην κλασική περίπτωση της δομής διασύνδεσης της JavaScript, όπου ορίζουμε τα χειριστήρια συμβάντων ως ιδιότητες των αντικειμένων συμβάντων (event objects) με περιεχόμενο script συναρτήσεις σαν αποκρίσεις στα συγκεκριμένα συμβάντα, έχουμε μόνο μία συνάρτηση-απόκριση για κάθε ένα συγκεκριμένο συμβάν του ίδιου στοιχείου-αντικειμένου του εγγράφου.
Για παράδειγμα το χειριστήριο συμβάντος onClick, σαν ιδιότητα του στοιχείου-αντικειμένου ... μπορεί να λάβει μόνο μία συνάρτηση σαν περιεχόμενο, ήτοι : link.onClick = funcA
.

- των ακροατών συμβάντων (event liesteners)

object (eventType, functionName, useCapture)

Όπου, σε κάθε αντικείμενο-κόμβο της δομής του εγγράφου DOM, προσθέτουμε όσους ακροατές συμβάντων θέλουμε, οι οποίοι μπορούν να αποκρίνονται στο ίδιο συμβάν που αφορά το ίδιο αντικείμενο-κόμβο με τις ανάλογες συναρτήσεις.
Μπορούμε για παράδειγμα να έχουμε πέντε διαφορετικές μεθόδους απόκρισης για το συμβάν click για το ίδιο στοιχείο link, προσθέτοντας πέντε διαφορετικούς ακροατές συμβάντων με διαφορετικές συναρτήσεις απόκρισης για το συγκεκριμένο συμβάν.
Ουσιαστικά, οι ακροατές συμβάντων (event liesteners) είναι ειδικά αντικείμενα (special objects), τα οποία προσθέτουμε στα στοιχεία-αντικείμενα του εγγράφου

μας με σκοπό να ακροαστούν το συγκεκριμένο γεγονός, όταν λάβει χώρα και να αποκριθούν σε αυτό.

3.7.5 Αναπαραγωγή Συμβάντος (propagation)

Από την στιγμή που πυροδοτείται ένα συμβάν,
αναπαράγεται (propagate) σε κάθε επίπεδο της ιεραρχικής δομής των αντικειμένων του εγγράφου, ανάλογα με το μοντέλο συμβάντων που υποστηρίζει ο browser που διαβάζει το συγκεκριμένο έγγραφο.
Είδαμε στο προηγούμενο κεφάλαιο, τα μοντέλα συμβάντων του IE και του Netscape και τις μεθόδους αναπαραγωγής συμβάντων που υποστηρίζουν.
Ο W3C, στην προσπάθειά του να εξισορροπήσει τις διαφορές και να πετύχει την μέγιστη δυνατή συμβατότητα υποστηρίζει και τις δύο μεθόδους αναπαραγωγής των συμβάντων ήτοι την μέθοδο της σύλληψης (capture) του συμβάντος του Netscape και την μέθοδο του κοχλασμού του συμβάντος (bubble) πάνω στην επιφάνεια του εγγράφου του IE.
Το DOM μοντέλο συμβάντων, μας δίνει την δυνατότητα να καθορίσουμε είτε την μία είτε την άλλη μέθοδο σαν διαδρομή ενός συμβάντος, με αναφορά στις τιμές boole (αληθές = true ή ψευδές = falshe) της useCapture που είναι παράμετρος του ακροατή του συγκεκριμένου συμβάντος (event listener).
Στην περίπτωση που επιλέγουμε την μέθοδο capure η τιμή της παραμέτρου useCapture πρέπει να οριστεί να είναι αληθής = true.
Εάν επιλέξουμε την μέθοδο bubble η τιμή της παραμέτρου useCapture πρέπει να είναι ψευδής = false.
Όπως είδαμε, το μοντέλο συμβάντων DOM, μας δίνει την δυνατότητα να χρησιμοποιήσουμε περισσότερους του ενός ακροατές συμβάντων (event listeners) για το ίδιο συμβάν και να τους προγραμματίσουμε έτσι ώστε να αποκριθούν ο κάθε ένας με τον τρόπο του στο συγκεκριμένο συμβάν.
Μέσα από την δυνατότητα αυτή, μπορούμε να χρησιμοποιήσουμε έναν ακροατή συμβάντος (event listener) για την διαδρομή του συμβάντος προς το αντικείμενο - στόχο (capure) και έναν για την διαδρομή της επιστροφής του συμβάντος από το αντικείμενο συμβάντος (event object) προς την επιφάνεια του εγγράφου - document, που είναι το παράθυρο του browser (bubble).
Μέσα από την ιδιότητα eventPhase, του αντικείμενου event του DOM, μπορούμε να γνωρίζουμε την φάση στην οποία βρίσκεται κάθε στιγμή το συμβάν που λαμβάνει χώρα και να εκτελούμε τον συγκεκριμένο ακροατή συμβάντος, δηλαδή αφού πρώτα διαπιστώσουμε την πορεία του του συμβάντος στην ιεραρχία της δομής των αντικειμένων του εγγράφου,

- Με

  eventPhase = 0, έχουμε την πραγματοποίηση του συμβάντος από έξω προς τα μέσα (capture)

- Με

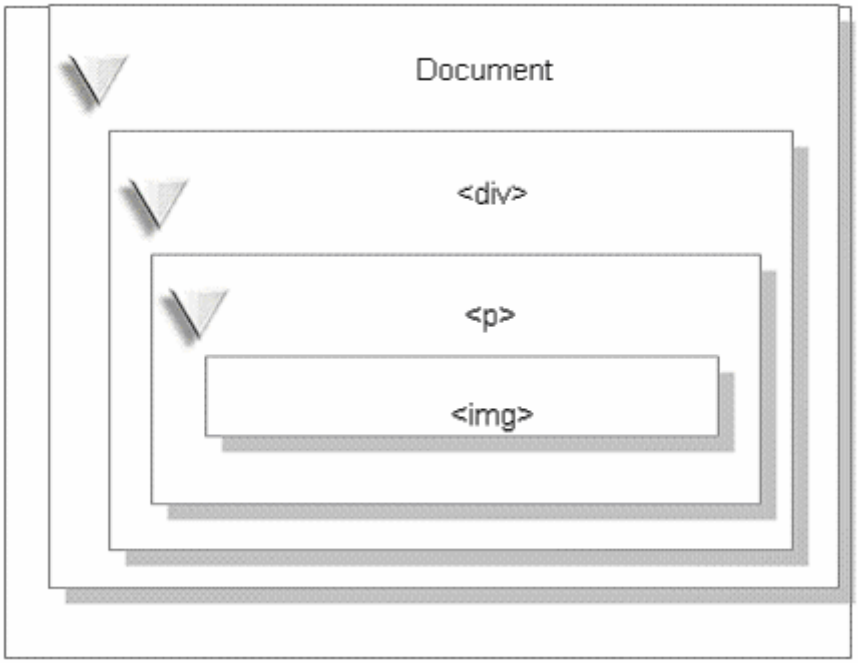
  eventPhase = 1, όταν το συμβάν βρίσκεται και λαμβάνει χώρα στο αντικείμενο - στόχο (target)

- Με

  eventPhase = 2, έχουμε την πραγματοποίηση του συμβάντος από μέσα προς τα έξω (bubbling)

3.7.6 Μέθοδος capture με τιμή αληθές

Η πορεία που ακολουθεί ένα συμβάν προς το αντικείμενο - στόχο,
από την στιγμή που αυτό πυροδοτείται στην επιφάνεια του εγγράφου-document (browser window), περιγράφεται με την μέθοδο capture.
Εάν υποθέσουμε πως έχουμε ένα έγγραφο HTML, με μία ορισμένη περιοχή , της οποίας η παράγραφος <p id="parA1"> περιέχει ένα στόχο-αντικείμενο εικόνα <img id="iconA1> για ένα συμβάν ποντικού (mouse event) click και προσθέσουμε ένα ακροατή συμβάντος (event listener) για το εν λόγω συμβάν σε κάθε ένα από τα ανωτέρω στοιχεία-αντικείμενα του εγγράφου ορίζοντας σαν μέθοδο αναπαραγωγής του συμβάντος την μέθοδο capture, διαγραμματικά θα έχουμε την διαδρομή.



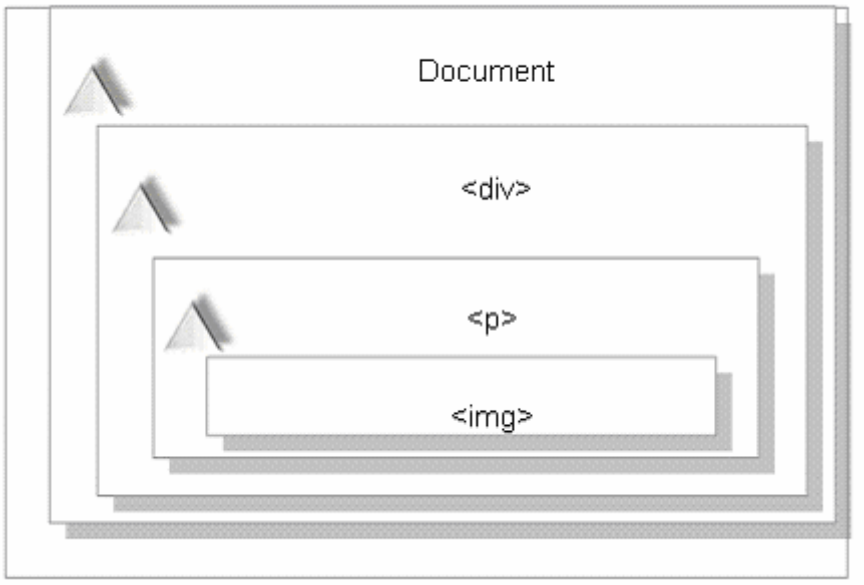
Σχήμα 10) Αναπαραγωγή συμβάντος capture

Όταν ο χρήστης κάνει κλικ στην συγκεκριμένη εικόνα του εγγράφου, το συμβάν click θα πυροδοτηθεί, όμως δεν θα το λάβει πρώτα η εικόνα στόχος κι ας έγινε κλικ πάνω σε αυτή, αλλά ο ακροατής συμβάντος (event listener) που είναι συνδεδεμένος με το έγγραφο- document.
Εξ΄ αυτού και ο όρος της σύλληψης συμβάντος (event capture), πρώτα από το ανώτατο αντικείμενο στην ιεραρχία των αντικειμένων της δομής DOM, που είναι το αυτό καθ΄ εαυτό το έγγραφο - document, ανεξαρτήτως της θέσης που λαμβάνει χώρα το συμβάν.
Θα λέγαμε πως ο ακροατής συμβάντος που είναι συνδεδεμένος με το document ακροάται στην γέννηση του το συμβάν και στην κυριολεξία το αρπάζει ή το συλλαμβάνει πραγματοποιώντας την σχετική επεξεργασία.

Στην συνέχεια του παραδείγματός μας, το συγκεκριμένο συμβάν περνά στον ακροατή συμβάντος (event listener) του αντικείμενου και μετά στο αντικείμενο <p> και τέλος στο αντικείμενο-στόχο εικόνα <img>.
Η όλη διαδρομή ακολουθεί την ιεραρχία της δομής των κόμβων του οικογενειακού δέντρου οργάνωσης, έτσι ώστε όλοι οι κόμβοι πρόγονοι του στόχου-αντικείμενου να συλλαμβάνουν και να αποκρίνονται ανάλογα στο συμβάν.

3.7.7 Μέθοδος κοχλασμού-bubble με τιμή ψευδές

Όταν χρησιμοποιούμε την μέθοδο του κοχλασμού (bubble),
Έχουμε αποφασίσει να έχουμε προσπέλαση σε συμβάντα που ακολουθούν διαδρομή από μέσα προς τα έξω, δηλαδή από συγκεκριμένα σημεία ή θέσεις του εγγράφου όπου λαμβάνουν χώρα με τελικό προορισμό την επιφάνεια του εγγράφου-document.
Στο προηγούμενο παράδειγμα, του συμβάντος του ποντικού (mouse event) click , εάν προσθέσουμε ένα ακροατή συμβάντος (event listener) για το εν λόγω συμβάν σε κάθε ένα από τα ανωτέρω στοιχεία-αντικείμενα του εγγράφου ορίζοντας σαν μέθοδο αναπαραγωγής του συμβάντος την μέθοδο bubble, διαγραμματικά θα έχουμε την διαδρομή.



Σχήμα 11) Αναπαραγωγή συμβάντος bubbling

Όταν ο χρήστης κάνει κλικ στο συγκεκριμένο αντικείμενο-στόχο, εν προκειμένω στο αντικείμενο <img>, ο ακροατής συμβάντος (event listener) που είναι συνδεδεμένος με αυτό θα αποκριθεί πρώτος και θα επεξεργαστεί το συμβάν.
Στην συνέχεια το συμβάν θα περάσει στο γονικό και στην συνέχεια στα προγονικά στοιχεία-αντικείμενα σύμφωνα με την ιεραρχία, μέχρι να επιστρέψει στον κόμβο-αντικείμενο ρίζα (root node) που είναι το document.

3.8 DOM Ακροατές

3.8.1 Ακροατής Συμβάντος & Διασύνδεση DOM

Ο ακρατής συμβάντος (event listener) είναι μία συνάρτηση,
ή ένα ειδικό αντικείμενο (special object), το οποίο καθορίζει την διασύνδεση (interface) ενός αντικειμένου συμβάντος με τον τύπου του συμβάντος (eventType), την απόκριση σε αυτό (functionName) και την μέθοδο αναπαραγωγής του (propagation).
Η γενική μορφή της συνάρτησης διασύνδεσης ενός ακροατή συμβάντος (event listener interface), είναι :

object(eventType,functionName,useBoolean)

Με τις ανεξάρτητες μεταβλητές ή τις παραμέτρους της συνάρτησης, eventType, δηλώνουμε σε αλφαριθμητική σταθερά (string), τον τύπο του συμβάντος που θέλουμε να ακροάται ο συγκεκριμένος ακροατής (listener).

Έγκυρες δηλώσεις είναι, "click" , "mouseover", "mouseout", "focus", κ.λ.π. Δηλώσεις που αφορούν χειριστές συμβάντων με το πρόθεμα on, όπως "onclick" δεν είναι αποδεκτές.

functionName, δηλώνουμε την συνάρτηση - κώδικα script - που θέλουμε να εκτελεστεί σαν απόκριση στο συγκεκριμένο συμβάν όταν αυτό πυροδοτηθεί και λάβει χώρα.

Δεν πρέπει να χρησιμοποιούνται εισαγωγικά, διότι δεν είναι αλφαριθμητική σταθερά - string, το όνομα μιας συνάρτησης που αφορά χειριστή συμβάντος (event handler)

useBoolean (true ή false), δηλώνουμε μία μεταβλητή Boole, που λέει στον ακροατή πότε θα εκτελέσει τον χειριστή συμβάντος, την συνάρτηση δηλαδή που έχουμε δηλώσει με την μεταβλητή functionName.

Εάν η τιμή είναι αληθές (true), η συνάρτηση εκτελείται όταν το συμβάν έχει πορεία από την επιφάνεια του εγγράφου προς το αντικείμενο - στόχο (capture phase).

Εάν η τιμή είναι ψευδές (false), η συνάρτηση εκτελείται όταν το συμβάν έχει πορεία ταξιδεύοντας προς τα πίσω από το αντικείμενο - στόχο προς την επιφάνεια του εγγράφου (bubbling phase) .

ΚΕΦΑΛΑΙΟ 4

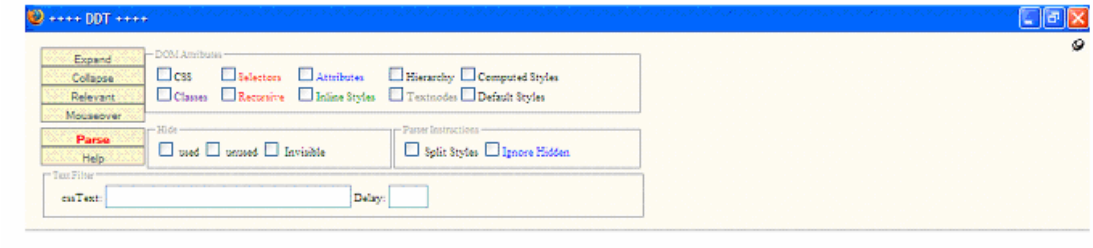
Εργαλεία DOM (DOM Tools)

4.1 Γενικά

Για να κατανοήσουμε παραπάνω το DOM και τις δυνατότητες του θα μελετήσουμε κάποια εργαλεία. Συγκεκριμένα θα δούμε κάποιες από τις δυνατότητες του DOM Inspector αλλά και του Dynamic Dom Tools. Αυτά τα δύο εργαλεία είναι ελεύθερα στην χρήση και μπορούν να επεμβαίνουν στο DOM της κάθε σελίδας. Στην ουσία έχουν την δυνατότητα να διαβάζουν τον DOM κάθε σελίδας. Με λίγα λόγια μπορούμε να προσθέτουμε ή να αφαιρούμε κόμβους, ετικέτες, στοιχεία, χαρακτηριστικά κ.α.

4.2. Dynamic Dom Tools (DDT)

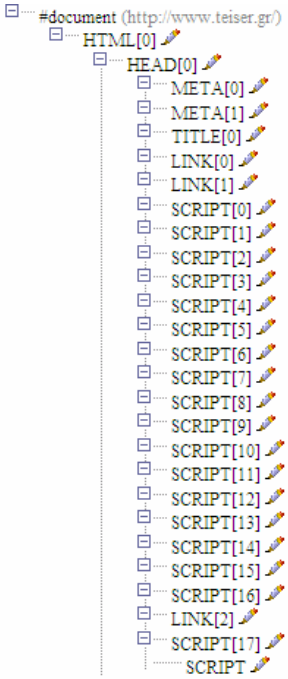
Ή εικόνα που βλέπουμε μόλις τρέξουμε το DDT φαίνεται παρακάτω



Σχήμα 12) Η εικόνα του DDT

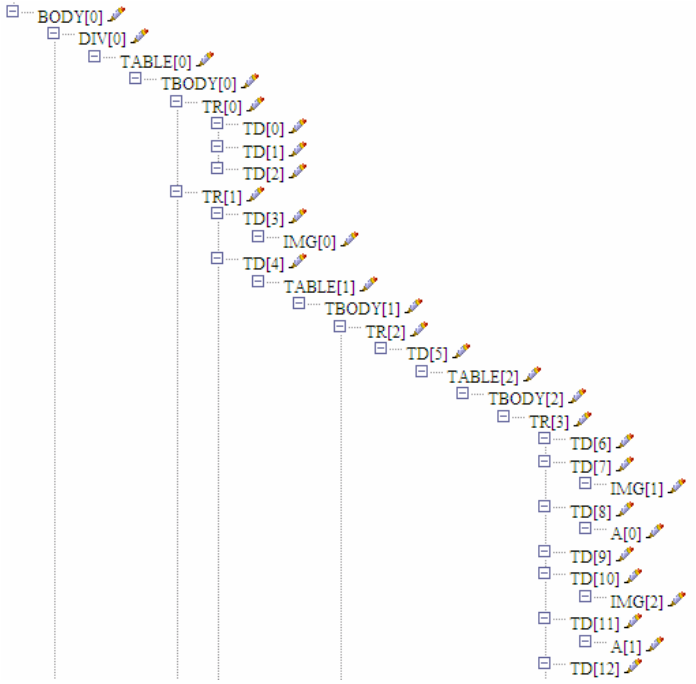
Όπως παρατηρούμε το DDT έχει κάποια φίλτρα με τα οποία μπορούμε να κατανοήσουμε καλύτερα την δενδροειδή μορφή του DOM της κάθε σελίδας και να επέμβουμε σε αυτήν.

Για παράδειγμα εάν θέλαμε να αναλύσουμε τον DOM του www.teiser.gr θα ανοίγαμε την ιστοσελίδα του TEI, μετά θα τρέχαμε το DDT και θα πατούσαμε το κουμπί Parse. Το αποτέλεσμα θα ήταν το παρακάτω



Σχήμα 13)Το DOM του www.teiser.gr (HEAD)

Και ένα μέρος από το BODY (για ευνόητους λόγους)

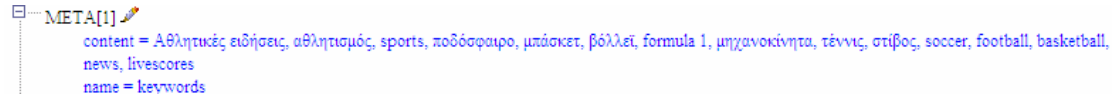


Σχήμα 14) Το DOM του www.teiser.gr (BODY)

Μπορούμε βέβαια να επιλέξουμε το κουτάκι Attributes και το DDT να μας δώσει τα χαρακτηριστικά των στοιχείων με μπλέ χρώμα. Ας το δούμε σε μια νέα ιστοσελίδα. Π.χ στο www.contra.gr το αποτέλεσμα θα ήταν

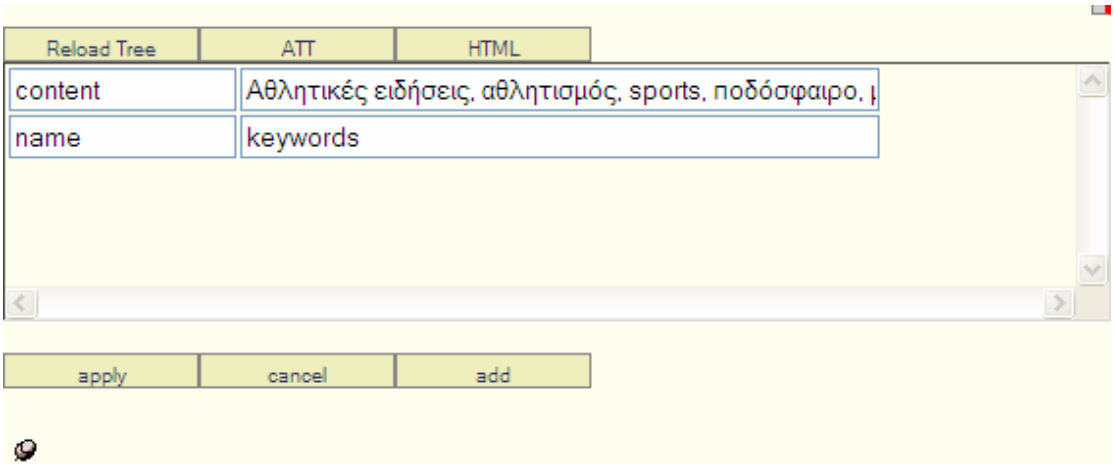
Σχήμα 15) Attributes στο www.contra.gr

Όπως παρατηρούμε δίπλα από κάθε στοιχείο-κόμβο υπάρχει ένα στυλό . Εάν πατήσουμε το στυλό θα εμφανιστεί ένα νέο παράθυρο με τα χαρακτηριστικά του στοιχείου στο οποίο μπορούμε να τα επεξεργαστούμε. Για παράδειγμα έστω ότι θέλουμε να προσθέσουμε μια ετικέτα (tag) στις ήδη υπάρχουσες. Θα πηγαίναμε στο στοιχείο που αναφέρεται στις λέξεις κλειδιά και θα πατούσαμε το στυλό.



Σχήμα 16) Επιλογή κόμβου στο DDT για τροποποίηση χαρακτηριστικών

Στην συνέχεια θα μας άνοιγε ένα νέο παράθυρο



Σχήμα 17) Παράθυρο τροποποίησης χαρακτηριστικών στο DDT

Τώρα μπορούμε να πάμε στην περιοχή content και να προσθέσουμε ή να αφαιρέσουμε λέξεις κλειδιά. Αν πατήσουμε Reload Tree και μετά Apply οι αλλαγές μας θα έχουν καταχωρηθεί. Με τον ίδιο τρόπο μπορούμε να αλλάξουμε ότι χαρακτηριστικό θέλουμε. Εαν επιθυμούσαμε θα μπορούσαμε να διαλέξουμε την ετικέτα HTML και να δούμε τον κώδικα σε HTML.

Αν διαλέξουμε τώρα την επιλογή CSS και Parse στο www.teiser.gr

```
==== CSS StyleSheet List: ====
*** http://www.teiser.gr/teiser.css ***
body
{
 background: rgb(226, 214, 181) url(images/background.gif) repeat-x scroll 0% 0%;
 margin-top: 10px;
 -moz-background-clip: -moz-initial;
 -moz-background-origin: -moz-initial;
 -moz-background-inline-policy: -moz-initial;
}
h3
{
 font-size: 13px;
}
h4
{
 font-size: 11px;
}
a
{
 color: rgb(153, 0, 0);
 text-decoration: none;
}
a:hover
{
 color: rgb(204, 0, 0);
}
```

Σχήμα 18) Τα CSS στο www.teiser.gr

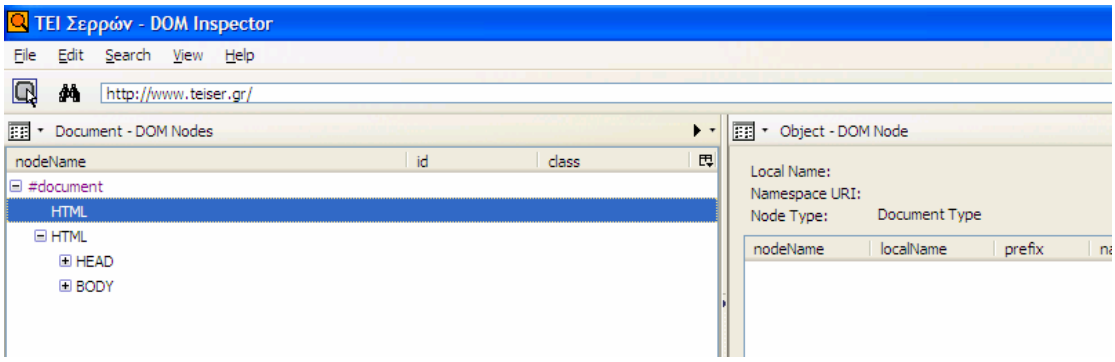
Στην αρχή για κάθε stylesheet έχει ένα σύνδεσμο που εάν τον επιλέξουμε μας δείχνει το πηγαίο αρχείο με τον κώδικα. Στην συνέχεια εμφανίζονται οι πληροφορίες που έχουν φορτωθεί στο DOM. Εμείς έχουμε την δυνατότητα να παρέμβουμε είτε αλλάζοντας κάποια στοιχεία είτε πατώντας το να δούμε την ιστοσελίδα χωρίς τις συγκεκριμένες πληροφορίες.

4.3 Dom Inspector

Ο Dom Inspector είναι άλλο ένα εργαλείο το οποίο διαβάζει το DOM της ιστοσελίδας που επιθυμούμε και μας δίνει την δυνατότητα να επεμβαίνουμε σε αυτό. Αποτελείται από δύο κυρίως παράθυρα. Το ένα μας δείχνει τους κόμβους του DOM και στη διπλανή έχουμε τα στοιχεία του αντικείμενου που επιλέγουμε.

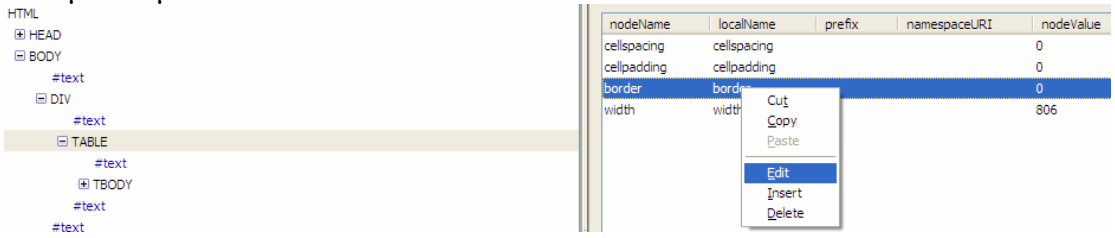
Πρώτο βήμα μας είναι να επιλέξουμε την τοποθεσία που θέλουμε να εξερευνήσουμε. Πηγαίνουμε File->Inspect a Url και στην συνέχεια γράφουμε την ιστοσελίδα που επιθυμούμε π.χ. www.teiser.gr

Το αποτέλεσμα φαίνεται παρακάτω



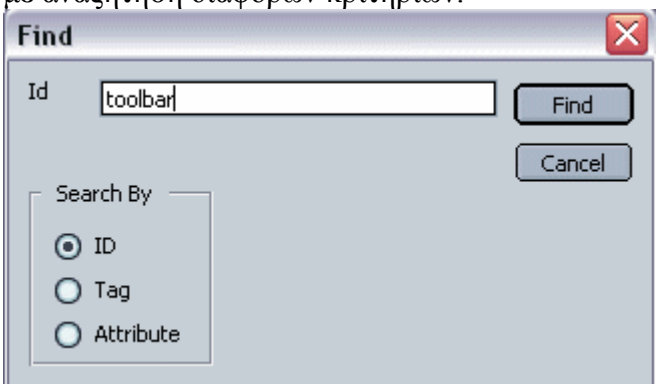
Σχήμα 19) Το DOM του www.teiser.gr μέσω του DOM Inspector

Στο αριστερό παράθυρο έχουμε τους κόμβους του DOM και στο δεξί παράθυρο έχουμε τα χαρακτηριστικά του αντικειμένου που επιλέγουμε στο αριστερό. Μπορούμε να αλλάξουμε όποια χαρακτηριστικά θέλουμε εμείς με απλό τρόπο όπως κάναμε και με το DDT.



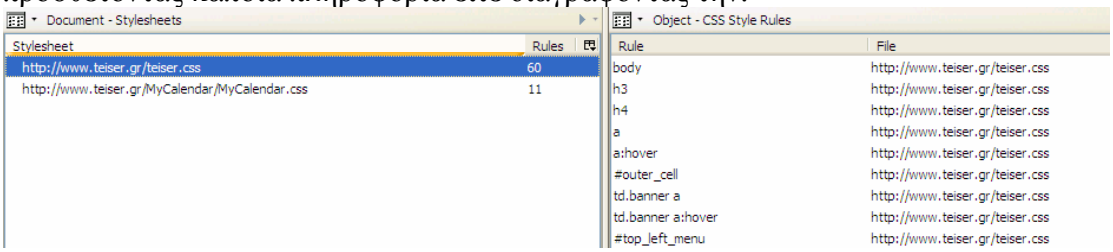
Σχήμα 20) Τροποποίηση χαρακτηριστικών στον DOM Inspector

Επίσης έχουμε την δυνατότητα να ψάξουμε τον κόμβο-στοιχείο που εμείς επιθυμούμε με αναζήτηση διαφόρων κριτηρίων.



Σχήμα 21) Αναζήτηση με κριτήρια στον DOM Inspector

Επίσης μπορούμε να δούμε τα stylesheet και να επέμβουμε σε αυτά είτε προσθέτοντας κάποια πληροφορία είτε διαγράφοντας την.



Σχήμα 22) Τροποποίηση stylesheet στον DOM Inspector

Μια σημαντική λειτουργία που μας προσφέρει το Dom Inspector είναι ότι μπορούμε να αποθηκεύσουμε τις αλλαγές που έχουμε κάνει στο DOM σε ένα αρχείο HTML ή XML.

ΕΠΙΛΟΓΟΣ

Πολλοί λένε ότι το Semantic Web (Σημασιολογικό Ιστός) θα είναι το κύριο χαρακτηριστικό του Web 3.0. Εύκολο να το λέμε αλλά στην πράξη δεν είναι τόσο απλό. Αυτή τη στιγμή υπάρχουν κενά τεχνολογίας και τεχνογνωσίας που πρέπει να καλυφθούν για να γίνει κάτι τέτοιο μαζική πραγματικότητα όπως το Web 2.0. Από την άλλη πλευρά το Web 2.0 είναι κάτι που είδαμε χρησιμοποιήσαμε και φτάνει σιγά σιγά σε έναν μικρό κορεσμό. Φωτογραφίες, Bookmarks, Video, Voip, Blogging και άλλα παρόμοια τα οποία σε μεγάλο βαθμό έχουν καλυφθεί από τις δυνατότητες και την καινοτομία που είχε να προσφέρει το Web 2.0.

Χαρακτηριστικό παράδειγμα, όταν πρωτοπαρουσιάστηκε το Σημασιολογικό Διαδίκτυο, που αναφέρθηκε ήταν ότι οι σημερινές μηχανές αναζήτησης όταν γράφουμε την λέξη Paris δεν μπορούν να καταλάβουν εάν εννοούμε το Παρίσι ή την Paris Hilton. Ο Σημασιολογικός Ιστός με λίγα λόγια θα μπορεί να καταλαβαίνει τι θέλουμε από την έννοια της λέξης.

Στην Ελλάδα το Web 2.0 προχωράει αλλά ο μέσος Έλληνας θα σε κοιτάξει στραβά αν του αναφέρεις αυτόν τον όρο. Πόσο μάλλον αν αρχίσεις να του λες και Semantic. Η μετάβαση λοιπόν προς το παρόν έχει ανάγκη από πολύ έρευνα και δεν αναμένετε στα επόμενα χρόνια να δούμε ένα μαζικό semantic web.

Φυσικά και έχουν γίνει κάποιες προσπάθειες πάνω στον Σημασιολογικό Ιστό και τις δυνατότητες του με μερικές να παίρνουν ιδιαίτερη δημοσιότητα. Χαρακτηριστικό παράδειγμα η ιστοσελίδα της Wolfram Alpha η οποία μπορεί να απαντήσει σε διάφορα ερωτήματα (π.χ. ποιο είναι το ύψος ενός βουνού, πόσος ο πληθυσμός μιας πόλης και άλλα. Λόγω της ικανότητάς της να καταλαβαίνει τη φυσική ανθρώπινη γλώσσα (και όχι απλώς λέξεις-κλειδιά όπως η Google και οι άλλες μηχανές αναζήτησης), η Wolfram Alpha πιθανότατα θα ενσωματωθεί στο Σημασιολογικό Διαδίκτυο.

Η μετάβαση από το Web 2.0 στο Web 3.0 θα γίνει και γίνεται σταδιακά και το κύριο χαρακτηριστικό του θα είναι ο Σημασιολογικός Ιστός.

ΒΙΒΛΙΟΓΡΑΦΙΑ

1. Vladimir Geroimenko, Chaomei Chen. Visualizing the Semantic Web. XML based Internet and information visualization , 3rd printing, 2004
2. Bhavani Thuraisingham. Xml databases and the Semantic Web, 2002
3. Giorgos Stamou, Stefanos Kolias. Multimedia content and the semantic web : methods, standards and Tools, 2005
4. John Davies, Dieter Fensel, Frank Van Harmelen. Toward the Semantic Web Ontology-Driven Knoledge Management, 2003
5. Βασιλειάδης Νικόλαος, Βλαχάβας Ιωάννης, Κεφάλας Πέτρος, Κόκκορας Φώτης, Σακελλαρίου Ηλίας. Τεχνητή Νοημοσύνη, Έκδοση Γ΄, 2006
6. Πληροφορίες από την ηλεκτρονική διεύθυνση www.istorama.gr
7. Πληροφορίες από την ηλεκτρονική διεύθυνση www.w3schools.com
8. Πληροφορίες από την ηλεκτρονική διεύθυνση www.headwork.net