

Τεχνολογικό Εκπαιδευτικό Ίδρυμα Κρήτης

Σχολή Τεχνολογικών Εφαρμογών

Τμήμα Μηχανικών Πληροφορικής



Πτυχιακή Εργασία

Αξιοποίηση δημόσιων API για την προώθηση περιεχομένου στο Διαδίκτυο.

Φερετζάκης Γεώργιος (Α.Μ 3619)

Επιβλέπων καθηγητής : Ακουμιανάκης Δημοσθένης

ΗΡΑΚΛΕΙΟ

2018

Ευχαριστίες

Αρχικά, θα ήθελα να εκφράσω τις θερμότερες ευχαριστίες μου στον Δρ. Δημοσθένη Ακουμιανάκη για την καθοδήγηση του ως επιβλέπων καθηγητής στην πτυχιακή μου εργασία και την ευκαιρία που μου έδωσε να ενταχθώ στο ερευνητικό Εργαστήριο Τεχνολογίας & Συστημάτων Λογισμικού (iSTLab). Επίσης θα ήθελα να ευχαριστήσω τον Δρ. Βιδάκη Νικόλαο και την εξαιρετική ομάδα του εργαστηρίου για την πολύτιμη βοήθεια και την άψογη συνεργασία που είχαμε. Τέλος, θέλω να ευχαριστήσω την οικογένεια μου για τη συνεχή συμπαράσταση και υποστήριξη που μου παρείχε.

Abstract

This thesis attempts to implement an advanced web-based resource manager of a group shared repository in the cloud. The system is based on modern internet technologies and adopts an architectural development approach that allows the exploitation of already existing cloud services and applications. In addition, this "mashup" web application, supported by various public APIs, and particularly those that follow the REST (REpresentational State Transfer) architecture style, allows users to add value to the digital content with metadata and to share this on other applications in the context of the involved online workgroup. Specifically, users access the system using their Google account credentials and then can insert/remove and browse content, search content by multiple criteria and metadata, tag and like content. Automatic updates are supported through online services such as Google Mail and shared calendars with digital material attachment support. Lastly, we developed a mechanism to detect possible changes of resource tags by iSTLab Publications Calendar users, to update the file(s) metadata that the application website make use of, for faster information retrieval. As a use case for the above, the system utilised the data on iSTLab team publications that have been stored on shared Google Drive, from which the necessary data will be retrieved.

Σύνοψη

Η παρούσα πτυχιακή εργασία επιχειρεί να υλοποιήσει ένα προηγμένο διαδικτυακό σύστημα διαχείρισης ψηφιακών πόρων σε ένα κοινόχρηστο αποθετήριο μιας ομάδας στο νέφος. Το σύστημα είναι βασισμένο σε σύγχρονες τεχνολογίες του διαδικτύου και υιοθετεί μία αρχιτεκτονική προσέγγιση ανάπτυξης με την οποία αξιοποιούνται ήδη υπάρχουσες εφαρμογές και υπηρεσίες νέφους. Επιπλέον, αυτό η "mashup" διαδικτυακή εφαρμογή που υποστηρίζεται από διάφορες δημόσιες διεπαφές προγραμματισμού εφαρμογών (public APIs), και ειδικότερα αυτές που ακολουθούν την αρχιτεκτονική REST (REpresentational State Transfer), επιτρέπει στους χρήστες να προσθέσουν αξία στο περιεχόμενο με μεταδεδομένα και να το διαμοιράσουν σε άλλες εφαρμογές στο πλαίσιο της εμπλεκόμενης διαδικτυακής ομάδας. Πιο συγκεκριμένα, οι χρήστες αποκτούν πρόσβαση στο σύστημα μέσω πιστοποίησης του Google λογαριασμού τους και έπειτα έχουν τη δυνατότητα προσθαφαίρεσης και επισκόπησης περιεχομένου, αναζήτησης βάσει πολλαπλών κριτηρίων και μεταδεδομένων, επισήμανσης με ετικέτες και δήλωση αρέσκειας. Επίσης υποστηρίζονται αυτόματες ενημερώσεις περιεχομένου μέσω ηλεκτρονικών υπηρεσιών, όπως Google Mail και διαμοιραζόμενα ηλεκτρονικά ημερολόγια με δυνατότητες αρχειοθέτησης ψηφιακού υλικού (iSTLab Calendar). Τέλος, αναπτύχθηκε ένας μηχανισμός παρακολούθησης για την ανίχνευση πιθανών αλλαγών στις ετικέτες των αρχείων από τους χρήστες του iSTLab Calendar, ώστε να ενημερώνονται και τα μεταδεδομένα των αρχείων που αξιοποιεί η εφαρμογή για ταχύτερη άντληση πληροφοριών. Ως μελέτη περίπτωσης των παραπάνω αξιοποιήθηκαν τα δεδομένα που αφορούν δημοσιεύσεις της ομάδας iSTLab τα οποία βρίσκονται αρχειοθετημένα σε κοινόχρηστο Google Drive από το οποίο θα ανακτώνται και θα προστίθενται κατάλληλα τα απαραίτητα δεδομένα.

Πίνακας Περιεχομένων

Ευχαριστίες ii
Abstract iii
Σύνοψη iv
1 Εισαγωγή 1
2 Τρέχουσα Τεχνολογική Στάθμιση – State of the Art 2
2.1 Παγκόσμιος ιστός και η εξέλιξή του 2
2.2 Διαδικτυακές εφαρμογές 5
2.2.1 Διαδικτυακές εφαρμογές σε κινητές συσκευές 7
2.2.2 AJAX 7
2.3 Διαδικτυακές υπηρεσίες και διαλειτουργικότητα 8
2.3.1 Διαδικτυακές υπηρεσίες 8
2.3.2 Αρχιτεκτονική Υπηρεσιών 9
2.3.3 Πλεονεκτήματα μοντέλου Διαδικτυακών Υπηρεσιών 10
2.3.4 Βασικές Τεχνολογίες των Διαδικτυακών Υπηρεσιών 11
2.3.5 Αρχιτεκτονική REST 15
2.3.6 SOAP vs REST 16
2.3.7 Διαλειτουργικότητα Υπηρεσιών 17
2.4 Μεταδεδομένα και APIs 21
2.4.1 Χρησιμότητα Μεταδεδομένων 22
2.4.2 APIs 22
2.5 Mashups και η λογική τους 25
2.5.1 Περίπτωση Χρήσης Web 2.0 25
2.5.2 Περίπτωση Χρήσης Enterprise 2.0 26
2.5.3 Mashups vs Portals 26
2.5.4 Δομικά Χαρακτηριστικά Mashups 26
2.5.5 Αρχιτεκτονική Των Mashups 27
2.5.6 Screen-Scraping 28
2.5.7 Σημασιολογικός Ιστός, RDF και Microformats 29
2.5.8 RSS και ATOM 32
2.6 Tagging και Folksonomies 33
2.6.1 Κριτήρια αποδοτικού tagging 34
2.6.2 Tag Clouds 35
2.6.3 Σημασιολογικές προσεγγίσεις των folksonomies 36

2.7 Πλατφόρμες συνεργασίας 38
2.7.1 Αρχιτεκτονική Groupware 40
2.7.2 Κοινόχρηστοι χώροι εργασίας (shared workspaces) 41
2.7.3 Κοινόχρηστα αποθετήρια (shared repositories) 42
2.7.4 Υπηρεσίες υπολογιστικού νέφους και συνεργασία 43
3 Σενάριο χρήσης & προσέγγιση 46
3.1 iSTLab Calendar 46
3.2 Εργαλεία που αξιοποιήθηκαν 47
3.2.1 Google Sign-In 47
3.2.2 Google Drive API 48
3.2.3 Gmail API 49
3.2.4 Google Contacts API 50
3.2.5 LinkedIn REST API 50
3.2.6 Google App Engine 51
3.2.7 Materialize 51
3.2.8 jQuery 52
3.3 Λογική σχεδίαση 53
3.3.1 Διάγραμμα περιπτώσεων χρήσης 53
3.3.2 Διάγραμμα κλάσεων και ανάλυση λειτουργικότητας κλάσεων 55
4 Υλοποίηση του σεναρίου για το web 62
4.1 Επισκόπηση περιεχομένου 62
4.2 Εισαγωγή νέου περιεχομένου 65
4.3 Γραφική αναπαράσταση 71
4.4 Κοινοποίηση περιεχομένου 73
4.5 Μηχανισμός ενημερώσεων (email) 74
4.6 Μηχανισμός παρακολούθησης στο iSTLab Calendar 75
4.7 Διαμοιρασμός κοινόχρηστου αποθετηρίου 76
5 Κατακλείδα & Συμπεράσματα 78
5.1 Μελλοντική εργασία και επεκτάσεις 78
Βιβλιογραφία 79

Πίνακας Εικόνων

Εικόνα 1: Evolution of World Wide Web. 4
Εικόνα 2: Επικοινωνία πελάτη – διακομιστή 9
Εικόνα 3: Ρόλοι και λειτουργίες των υπηρεσιών ιστού. 10
Εικόνα 4: Υπηρεσίες ιστού και στοίβα πρωτοκόλλων SOA. 11
Εικόνα 5: Συστατικά περιγραφής των υπηρεσιών με WSDL 13
Εικόνα 6: Παράδειγμα WSDL εγγράφου. Με κόκκινα βέλη υποδεικνύονται τα στοιχεία του εγγράφου 14
Εικόνα 7: Επίπεδα ολοκλήρωσης υπηρεσίας 17
Εικόνα 8: BPEL Process για οργάνωση ταξιδιού 19
Εικόνα 9: Κατηγορίες μεταδεδομένων. 22
Εικόνα 10: Web APIs 24
Εικόνα 11: Επικεφαλίδα HTTP αιτήματος για κλήση του Google Calendar feed service (Calendar API) 24
Εικόνα 12: Αρχιτεκτονική mashups 28
Εικόνα 13: Κατευθυνόμενος γράφος των συστατικών 29
Εικόνα 14: Παράδειγμα RDF/XML κωδικοποίησης. 30
Εικόνα 15: Ο γράφος που προκύπτει από την RDF/XML αναπαράσταση (Εικόνα 14) 30
Εικόνα 16: Δεδομένα που εξάγονται από microformats 31
Εικόνα 17: Παραδείγματα microformats 32
Εικόνα 18: RSS 2.0 έγγραφο ηλεκτρονικού βιβλιοπωλείου 32
Εικόνα 19: Atom 1.0 έγγραφο ηλεκτρονικού βιβλιοπωλείου 33
Εικόνα 20: Tag Cloud με όρους σχετικούς προς το Web 2.0 36
Εικόνα 21: Οντολογία επισήμανσης MOAT 38
Εικόνα 22: Χρονική/Χωρική κατηγοριοποίηση του groupware 39
Εικόνα 23: Γενική αρχιτεκτονική 40
Εικόνα 24: Κεντροποιημένη αρχιτεκτονική 40
Εικόνα 25: Αρχιτεκτονική δημιουργίας αντιγράφων 41
Εικόνα 26: Αρχιτεκτονική iSTLab Calendar 47
Εικόνα 27: Χορήγηση δικαιωμάτων στην εφαρμογή 48
Εικόνα 28: Σε αυτό το τμήμα κώδικα το αντικείμενο που επιστρέφεται από την ταυτοποίηση του λογαριασμού Google του χρήστη, αν περιέχει κωδικό πιστοποίησης (authorization code) θα σταλεί στο backend για την ανταλλαγή του με access και refresh tokens και στη συνέχεια θα καλέσει το API για την άντληση των μεταδεδομένων των αρχείων της εφαρμογής. 48
Εικόνα 29: Μέθοδος Drive API για την άντληση των μεταδεδομένων των αρχείων της εφαρμογής 49
Εικόνα 30: Μέθοδοι κοινοποίησης περιεχομένου στο LinkedIn του object LinkedInAction 51
Εικόνα 31: Διάγραμμα περιπτώσεων χρήσης 54
Εικόνα 32:Δημιουργία credential με κωδικό πιστοποίησης 56
Εικόνα 33: Αποσπάσματα κώδικα της κλάσης Servlet 56
Εικόνα 34: Η μέθοδος που εισάγει custom properties στα νέα αρχεία 57
Εικόνα 35 : Τμήμα του διαγράμματος κλάσεων. Οι κλάσεις AuthServlet, MyListener και CredentialsCron είναι επιφορτισμένες με τη δημιουργία και διαχείριση των Google credentials και χρησιμοποιούν στελέχη της Constants 59

Εικόνα 36 : Τμήμα του διαγράμματος κλάσεων. Αναπαρίστανται οι βοηθητικές κλάσεις GDrive, SocialDatastore, Gmail , η κλάση/servlet Servlet η οποία είναι το βασικό servlet της εφαρμογής, και η Constants. 60
Εικόνα 37: Τμήμα του διαγράμματος κλάσεων. Οι τρεις κλάσεις στο κάτω μέρος περιέχουν τη λειτουργικότητα της παρακολούθησης 61
Εικόνα 38: Λογότυπο και κουμπί Sign-In της εφαρμογής 62
Εικόνα 39: Περιεχόμενο σε κάρτες (card-material components) 63
Εικόνα 40: Ταξινόμηση και φιλτράρισμα περιεχομένου 64
Εικόνα 41: Αυτόματη συμπλήρωση στο πλαίσιο αναζήτησης 64
Εικόνα 42: Αυτόματη συμπλήρωση στο πλαίσιο αναζήτησης με πληκτρολόγηση του χρήστη 65
Εικόνα 43: Δομή JSON νέας δημοσίευσης 66
Εικόνα 44: Δομή JSON ενημέρωσης δημοσίευσης 67
Εικόνα 45: Καρτέλα EVENT με τα πεδία δεδομένων του event 68
Εικόνα 46: Καρτέλα RESOURCES με τα πεδία δεδομένων του αρχείου προς μεταφόρτωση 68
Εικόνα 47: Καρτέλα TAGS με τα πεδία δεδομένων που χαρακτηρίζουν το αρχείο. 69
Εικόνα 48: Το autocomplete πεδίο αναζήτησης ενημερώθηκε με τα μεταδεδομένα που εισαγάγαμε στο νέο αρχείο 69
Εικόνα 49: Στο modal επεξεργασίας της δημοσίευσης μπορούμε να δούμε (από όλους) και να επεξεργαστούμε (από εμάς) τα μεταδεδομένα που έχουν καταχωρηθεί. 69
Εικόνα 50: Στο modal επεξεργασίας της δημοσίευσης μπορούμε να δούμε (από όλους) και να επεξεργαστούμε (από εμάς) τα μεταδεδομένα που έχουν καταχωρηθεί. 70
Εικόνα 51: Στο modal επεξεργασίας της δημοσίευσης μπορούμε να δούμε (από όλους) και να επεξεργαστούμε (από εμάς) τις ετικέτες που έχουν καταχωρηθεί 70
Εικόνα 52: Από το λογαριασμό Google του εργαστηρίου προστέθηκε η ετικέτα 'istlab'. Η ετικέτα 'calendaring' μπορεί να αφαιρεθεί μόνο από τον ιδιοκτήτη-tagger της. 70
Εικόνα 53: Το σύνολο των ετικετών της δημοσίευσης 70
Εικόνα 54: Επισκόπηση νέου περιεχομένου και likes. 71
Εικόνα 55: Δήλωση like 71
Εικόνα 56: Στη λίστα resources του Datastore παρατηρούμε το entity του αρχείου με το id του (κλειδί) και τα properties του, τα οποία είναι λίστες με tags από τον κάθε χρήστη και μία λίστα με όλους τους χρήστες στους οποίους αρέσει η δημοσίευση 71
Εικόνα 57: Σύννεφο ετικετών (tag cloud) με ετικέτες από τα μεταδεδομένα των αρχείων των δημοσιεύσεων (λέξεις-κλειδιά) 72
Εικόνα 58: Σύννεφο ετικετών (tag cloud) με ετικέτες από τα μεταδεδομένα των αρχείων των δημοσιεύσεων (ονόματα συγγραφέων) 72
Εικόνα 59: Το αποτέλεσμα του φιλτραρίσματος με τα tag clouds 73
Εικόνα 60: Κοινοποίηση στο LinkedIn 73
Εικόνα 61: Το company update που δημιουργήθηκε στο LinkedIn page του εργαστηρίου 74
Εικόνα 62: Ενημέρωση με email που στάλθηκε από την εφαρμογή στους εταίρους 74
Εικόνα 63: Event στο iSTLab Calendar 75
Εικόνα 64: Στιγμιότυπα του log monitor 75
Εικόνα 65: Στιγμιότυπα του log monitor 75
Εικόνα 66: Το νέο αρχείο του event 76
Εικόνα 67: Οι λέξεις κλειδιά του αρχείου όπως προστέθηκαν στο calendar. 76

Εικόνα 68: Λίστα αυτόματης συμπλήρωσης των Google Contacts του χρήστη.....77
Εικόνα 69: Το modal "Share Repository" με τις επιλεγμένες επαφές. 77

Εισαγωγή

Τα τελευταία χρόνια, η αλματώδης εξάπλωση του Διαδικτύου και των τεχνολογιών του «νέου» Παγκόσμιου Ιστού (Web 2.0), έχει δημιουργήσει νέα δεδομένα στην ανάπτυξη διαδικτυακών εφαρμογών, ενώ έχει εισάγει και μία νέα διάσταση στη δημιουργία και τη διαχείριση του ψηφιακού περιεχομένου. Στον «διασυνδεδεμένο» κόσμο του Διαδικτύου, οι χρήστες μπορούν εύκολα να ανταλλάσσουν πληροφορίες, να εκτελέσουν συνεργατικά καθήκοντα, να σχηματίσουν εικονικές κοινότητες χρηστών σύμφωνα με κοινά ενδιαφέροντα και να παράγουν πλούσιο και δυναμικό περιεχόμενο.

Στόχος της παρούσας πτυχιακής εργασίας είναι η μελέτη των διαδικτυακών δημόσιων διεπαφών προγραμματισμού εφαρμογών (public APIs) και η αξιοποίηση τους για ανάπτυξη mashup διαδικτυακών εφαρμογών. Ακόμα, καταπιάνεται και με τις πρακτικές των χρηστών που προσθέτουν αξία στο κοινόχρηστο περιεχόμενο (metadata), δημιουργώντας ένα σύστημα κατηγοριοποίησής του με τη χρήση ετικετών (tags). Στοχεύει, παράλληλα, στην υλοποίηση μιας διαδικτυακής εφαρμογής , με τη χρήση βασικών τεχνολογιών της τρέχουσας τεχνολογικής στάθμισης, που θα διαχειρίζεται το ψηφιακό περιεχόμενο μιας ομάδας χρηστών. Πιο συγκεκριμένα, το σύστημα αντλεί τα μεταδεδομένα των αρχείων που βρίσκονται αποθηκευμένα στο νέφος παρέχοντας στο χρήστη δυνατότητες κατηγοριοποίησης, χαρακτηρισμού και διαμοιρασμού τους.

Η πτυχιακή εργασία αποτελείται από πέντε (5) κεφάλαια, στα οποία αρχικά κάνουμε μια μικρή ιστορική αναδρομή για τον Παγκόσμιο Ιστό. Στην συνέχεια συνδυάζουμε το διαδίκτυο και τις τεχνολογίες του με τον Παγκόσμιο Ιστό, αναδεικνύοντας την χρησιμότητα τους και την επίδραση στο είδος της εφαρμογής που αναπτύχθηκε. Αναλύουμε την υλοποίηση και λειτουργία της εφαρμογής μας, παραθέτοντας στιγμιότυπα και κατάλληλα παραδείγματα. Τέλος, επιχειρούμε μία αποτίμηση της εκπόνησης της εργασίας και προτείνουμε μελλοντικές επεκτάσεις που θα μπορούσαν να υλοποιηθούν για τη βελτίωση επιμέρους λειτουργικών συστατικών.

2 Τρέχουσα Τεχνολογική Στάθμιση – State of the Art

2.1 Παγκόσμιος ιστός και η εξέλιξή του

Ο παγκόσμιος ιστός είναι ένα σύστημα διασυνδεδεμένων εγγράφων υπερκειμένου τα οποία είναι προσβάσιμα μέσω του Διαδικτύου. Παγκόσμιος ιστός και Διαδίκτυο συχνά θεωρούνται ταυτόσημα πράγματα. Η αντίληψη αυτή είναι λανθασμένη καθώς ο ιστός αποτελεί μία εφαρμογή του Διαδικτύου, και μάλιστα, την πιο δημοφιλή. Η τεχνολογία του ιστού δημιουργήθηκε το 1989 από τον Βρετανό Tim Berners-Lee. Το όνομα που έδωσε στην εφεύρεσή του είναι World Wide Web (www). Αυτό που οδήγησε τον Lee στην εφεύρεση του παγκοσμίου ιστού ήταν το όραμά του για ένα κόσμο που ο καθένας θα μπορούσε να ανταλλάσσει πληροφορίες και ιδέες άμεσα προσβάσιμες από τους υπολοίπους. Καθιστά δυνατή τη δημιουργία υπερκειμένων, μία διασύνδεση πάρα πολλών μη ιεραρχημένων στοιχείων που παλιότερα ήταν απομονωμένα. Τα στοιχεία αυτά μπορούν να πάρουν και άλλες μορφές πέραν της μορφής του γραπτού κειμένου, όπως εικόνας και ήχου.

Έως τα Χριστούγεννα του 1990, ο Berners-Lee είχε αναπτύξει όλα τα απαραίτητα εργαλεία για τη λειτουργία του ιστού: το πρωτόκολλο HTTP, τη γλώσσα υπερκειμένου HTML, τον πρώτο περιηγητή ιστού WorldWideWeb, το πρώτο λογισμικό διακομιστή με HTTP, τον πρώτο διακομιστή ιστού και τις πρώτες ιστοσελίδες όπου περιέγραφε το εγχείρημά του. Πεντακόσιοι διακομιστές ιστού υπήρχαν σε όλο τον κόσμο έως τον Οκτώβριο του 1993.

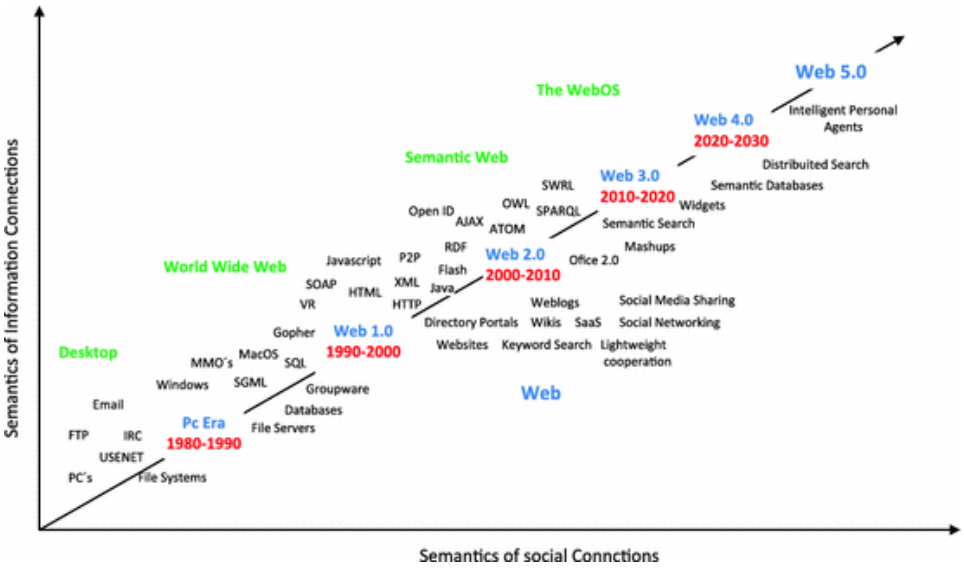
Έως το 1996 έγινε προφανές στις εισηγμένες στο χρηματιστήριο εταιρείες, ότι μία δημόσια παρουσία στο Διαδίκτυο δεν ήταν, πλέον, προαιρετική. Αν και αρχικά οι άνθρωποι εξέτασαν δυνατότητες για ελεύθερη δημοσίευση και πληροφόρηση από τον παγκόσμιο ιστό, η εξοικείωση με την αμφίδρομη επικοινωνία οδήγησε στη δυνατότητα ηλεκτρονικού εμπορίου.

Πριν το 1999 ο παγκόσμιος ιστός αποκαλούταν από τους ειδικούς ως «read-only» web (μόνο-για-ανάγνωση ιστός). Συνεπώς, αυτό περιόριζε τους χρήστες του διαδικτύου στο να αναζητούν και βλέπουν την πληροφορία που χωρίς τη δυνατότητα αμφίδρομης ροής πληροφοριών ή επικοινωνίας. Η απουσία της αλληλεπίδρασης του χρήστη με το web περιεχόμενο οδήγησε στη δημιουργία μιας νέας μορφής του παγκοσμίου ιστού, το Web 2.0. (1)

Το Web 2.0 ή "social web" όπως αποκαλούταν προσφέρει στους χρήστες διαμοιρασμό πληροφοριών και συνεργασία μέσω υπολογιστή μεταξύ άλλων. (2)Πλέον κάθε κοινός χρήστης μπορεί να έχει ενεργό ρόλο και να προσθέσει περιεχόμενο (προώθηση περιεχομένου) κάνοντας χρήση των blogging πλατφόρμων της εποχής (1999). Στην αρχή του 21^{ου} αιώνα το τοπίο στο διαδίκτυο και τον παγκόσμιο ιστό άλλαξε δραματικά και παρείχε εργαλεία όπως wiki, blog, tagging (folksonomies), feeds, social networking sites (π.χ. Facebook), video sharing websites (π.χ. YouTube) και online web services (π.χ. eBay) κάνοντας το διαδίκτυο μια πλατφόρμα συνεργασίας και δικτύωσης και δημιουργώντας μεγάλη αξία (social value). Επίσης, ένα πολύ σημαντικό χαρακτηριστικό του Web 2.0 αποτελεί η τάση των κινητών συσκευών και του mobile internet.

Τα κύρια χαρακτηριστικά του Web 2.0 είναι τα εξής: (3)

- Το διαδίκτυο και όλες οι συσκευές που είναι συνδεδεμένες σε αυτό, αποτελούν μια παγκόσμια πλατφόρμα επαναχρησιμοποιούμενων υπηρεσιών και δεδομένων, τα οποία προέρχονται κυρίως από τους ίδιους τους χρήστες και στις περισσότερες περιπτώσεις διακινούνται ελεύθερα.
- Αρκεί ένας browser, ώστε να χρησιμοποιείται σαν interface με αυτή την πλατφόρμα, η οποία λειτουργεί ανεξαρτήτως συσκευής πρόσβασης (π.χ. Η/Υ, PDA, κινητό τηλέφωνο) και λειτουργικού συστήματος. Μόνη προϋπόθεση είναι η ύπαρξη σύνδεσης στο διαδίκτυο.
- Λογισμικό, περιεχόμενο και εφαρμογές ανοιχτού κώδικα (open source).
- Χρήση κυρίως «ελαφριάς» τεχνολογίας σε ότι αφορά τα πρωτόκολλα, τις γλώσσες προγραμματισμού, τα interfaces, ενώ διαπιστώνεται και μια τάση για απλότητα στον προγραμματιστικό σχεδιασμό τους.
- Πλούσια και διαδραστικά interfaces χρηστών (Rich Internet Applications - RIA), δυναμικό περιεχόμενο, ιστοσελίδες που ανανεώνουν μόνο όποιο περιεχόμενό τους αλλάζει (τεχνολογία AJAX).
- Συνεχής και άμεση ανανέωση των δεδομένων και του λογισμικού.
- Προώθηση του δημοκρατικού χαρακτήρα του διαδικτύου, με τους χρήστες να έχουν τον πρωταγωνιστικό ρόλο.
- Υιοθέτηση της τάσης για αποκέντρωση των δεδομένων, υπηρεσιών και προτύπων.
- Δυνατότητα κατηγοριοποίησης και αξιολόγησης του περιεχομένου από το χρήστη με σημασιολογικές έννοιες για ευκολότερη αναζήτηση της πληροφορίας.
- Δυνατότητα για ανοιχτή επικοινωνία, ανάδραση, διάχυση πληροφοριών, άμεση συγκέντρωση και εκμετάλλευση της γνώσης των χρηστών για διάφορα ζητήματα.
- Αμφίδρομη επικοινωνία του χρήστη με επιχειρήσεις ή οργανισμούς που μπορεί να έχει σαν αποτέλεσμα την επίδρασή του στην υιοθέτηση κατευθύνσεων και στη λήψη αποφάσεων.



Εικόνα 1: Evolution of World Wide Web.

Βασιζόμενοι στον όρο «read-only» web της πρώτης μορφής του παγκοσμίου ιστού, το Web 3.0 θα μπορούσε να χαρακτηριστεί ως «read-write-execute» web. Το Web 3.0 είναι ο λεγόμενος σημασιολογικός ιστός, μια επέκταση του σημερινού Ιστού , που θα φέρει λογική δόμηση και σημασιολογία στο περιεχόμενο των ιστοσελίδων. Η λογική πίσω από αυτό είναι ότι η δημοσιευμένη πληροφορία θα περιέχει μεταδεδομένα, τα οποία θα είναι κοινά για όλους, θα μπορούν να «κατανοούνται» και από μηχανές, οι οποίες θα βοηθήσουν στην καλύτερη συλλογή και επεξεργασία τους. Αποτελεί ένα νέο κόσμο όπου θα υπάρχει η δυνατότητα ανάλυσης όλων των πληροφοριών που υπάρχουν με στόχο να αντιμετωπίσει το μεγάλο όγκο των δεδομένων και να γίνει πιο εύχρηστο για την καλύτερη εξυπηρέτηση των χρηστών.

Ο σημασιολογικός ιστός βασίζεται σε δύο πράγματα: το semantic markup (σημασιολογική σήμανση) και τα web services (υπηρεσίες Διαδικτύου). Η σημασιολογική σήμανση συνεισφέρει στην επικοινωνία ανθρώπου-υπολογιστικού συστήματος. Η αναπαράσταση πληροφορίας στον Παγκόσμιο Ιστό δε συνυπήρχε με τη συσχέτιση των δεδομένων σύμφωνα με ένα εννοιολογικό πλαίσιο και έτσι ήταν αδύνατο για τις διαδικτυακές εφαρμογές να «κατανοήσουν» κάποιου είδους συσχέτιση ανάμεσα στα δεδομένα. Η ιδέα της τροποποίησης των δεδομένων σε κατανοητή μορφή για μια διαδικτυακή εφαρμογή επιβεβαιώνει το "execute" μέρος του χαρακτηρισμού του Web 3.0 και σχετίζεται με το έτερο βασικό στοιχείο του σημασιολογικού ιστού, τα web services (υπηρεσίες Διαδικτύου). Οι διαδικτυακές υπηρεσίες είναι συστήματα λογισμικού που επιτρέπουν στις εφαρμογές να επικοινωνούν μεταξύ τους ανεξαρτήτως πλατφόρμας και γλώσσας προγραμματισμού. Συνδυάζοντας τα δύο αυτά θεμελιώδη στοιχεία του Web 3.0 υπάρχει η δυνατότητα επικοινωνίας των εφαρμογών και αναζήτησης πληροφοριών με καλύτερα αποτελέσματα.

Οι τεχνολογίες που βασίζεται ο σημασιολογικός ιστός είναι οι υφιστάμενες URI και XML αλλά και οι νέες τεχνολογίες RDF, RDFS, OWL, κα., οι οποίες αναπτύσσονται με την βοήθεια της κοινότητας. Δεδομένου ότι ο νέος Ιστός σκοπεύει να είναι μια μεγάλη βάση όπου δεδομένα από διαφορετικά πεδία θα συνδέονται μεταξύ τους, αναμένεται να παίξει μεγάλο ρόλο στη ζωή μας.

Το Web 3.0 αποτελεί τη μεγαλύτερη προσπάθεια αυτόματης ενοποίησης των συστημάτων ώστε να συνεργάζονται διαλειτουργικά σε παγκόσμιο επίπεδο. Στηρίζεται δηλαδή στην ιδέα της οργάνωσης και διασύνδεσης της πληροφορίας που υπάρχει στο διαδίκτυο, ώστε να χρησιμοποιηθεί πιο αποτελεσματικά

για την ανακάλυψη, αυτοματοποίηση και επαναχρησιμοποίησή της από διαφορετικές μεταξύ τους εφαρμογές. Δίνεται ιδιαίτερη έμφαση στις προτιμήσεις κάθε χρήστη (εξατομίκευση) και είναι δυνατή η εξαγωγή χρήσιμων πληροφοριών για τον χρήστη μέσα από τη διαδικτυακή του συμπεριφορά. Επίσης, χαρακτηριστικό του Web 3.0 είναι η φορητότητα. Ο χρήστης μπορεί να έχει πρόσβαση από οποιονδήποτε χώρο οποιαδήποτε στιγμή.

Μια -όχι νέα- αλλά εναλλακτική εκδοχή του υφιστάμενου παγκοσμίου ιστού είναι το Web 4.0, το λεγόμενο mobile web (κινητός παγκόσμιος ιστός) που βασίζεται στην ασύρματη επικοινωνία συνδέοντας ανθρώπους και συσκευές οπουδήποτε και οποτεδήποτε στον πραγματικό ή εικονικό κόσμο σε πραγματικό χρόνο. Ο «επόμενος» παγκόσμιος ιστός θα έχει ως κύριο θέμα τη φυσική αλληλεπίδραση ανθρώπου-υπολογιστή. Πρόκειται για το «emotional web» που θα αλληλοεπιδρά καθημερινά με τους φυσικούς χρήστες π.χ. μέσω αναγνώρισης προσώπου.

Το Web 5.0 θα είναι η (συναισθηματική) αλληλεπίδραση μεταξύ των ανθρώπων και των υπολογιστών. Η αλληλεπίδραση θα γίνει μια καθημερινή συνήθεια για πολλούς ανθρώπους με βάση τη Νευροτεχνολογία. Προς το παρόν διαδίκτυο είναι "συναισθηματικά" ουδέτερο, πράγμα που σημαίνει ότι δεν αντιλαμβάνεται πως οι χρήστες αισθάνονται και νιώθουν. Αυτό θα αλλάξει με το web 5.0 - συναισθηματικό web. Ένα παράδειγμα αυτού είναι www.wefeelfine.org, όπου χαρτογραφούνται συναισθήματα ανθρώπων. Με τα ακουστικά, οι χρήστες θα αλληλεπιδρούν με το περιεχόμενο που αλληλεπιδρά με τα συναισθήματα ή τις αλλαγές στην αναγνώριση του προσώπου τους.

2.2 Διαδικτυακές εφαρμογές

Μία εφαρμογή (application ή application software/program) είναι ένα πρόγραμμα υπολογιστή, tablet ή smartphone σχεδιασμένο για να εκτελεί μία ομάδα συντονισμένων λειτουργιών, έργων ή δραστηριοτήτων προς όφελος του χρήστη. Παραδείγματα εφαρμογών περιλαμβάνουν επεξεργαστές κειμένου, λογιστικά φύλλα, μια λογιστική εφαρμογή, ένα web browser, ένα media player, έναν αεροναυτικό προσομοιωτή πτήσης, μια κονσόλα παιχνιδιών ή ένα πρόγραμμα επεξεργασίας φωτογραφιών. (4)

Μια διαδικτυακή εφαρμογή είναι μία client-server (αρχιτεκτονική) εφαρμογή η οποία χρησιμοποιεί ένα browser ως περιβάλλον «φιλοξενίας» για το client μέρος-λειτουργία της. Παραδείγματα web application είναι webmail,electronic commerce και online chat applications.Η διάκριση ανάμεσα σε μία διαδικτυακή εφαρμογή και σε μια ιστοσελίδα δυναμικού περιεχομένου είναι ασαφής και αντικειμενική. Κατά γενική ομολογία μία web app καθορίζεται από την αλληλεπίδραση των χρηστών και την επεξεργασία δεδομένων που λαμβάνει χώρα στο server «κομμάτι» ενώ το περιεχόμενο καθορίζει μία ιστοσελίδα. Αυτό δεν αποκλείει την ύπαρξη ενός προηγμένου backend για μία ιστοσελίδα δυναμικού περιεχομένου. Για παράδειγμα μία σελίδα ειδήσεων (ιστοσελίδα) που παρουσιάζει το ίδιο περιεχόμενο σε όλους τους επισκέπτες διαφέρει από ένα συνεργατικό ημερολόγιο (web app) το οποίο επεξεργάζεται δεδομένα για τον κάθε χρήστη ατομικά.

Οι single-page applications συμπεριφέρονται περισσότερο ως εφαρμογές επειδή δεν ακολουθούν τη κλασική περιήγηση σε ξεχωριστές σελίδες (web documents) με τη χρήση διαφορετικών URLs. Η AngularJS είναι ένα εξαιρετικό πλαίσιο εργασίας (framework) για την ανάπτυξη τέτοιων εφαρμογών.[]

Σε πρωτύτερες αρχιτεκτονικές όπως η client-server, μια εφαρμογή αποτελούταν από ένα client side πρόγραμμα (user interface) το οποίο ήταν εγκατεστημένο σε κάθε υπολογιστή χρήστη τοπικά και το server-side μέρος. Όταν ο server-side κώδικας αναβαθμιζόταν ο κώδικας στα client μηχανήματα απαιτούσε αλλαγές οι οποίες αύξαναν το κόστος υποστήριξης και μείωναν την παραγωγικότητα. Σε αυτό

έρχονταν να προστεθούν και περιορισμοί όσον αφορά την αλλαγή περιβάλλοντος εκτέλεσης (hardware και λειτουργικά συστήματα) των εφαρμογών.

Οι διαδικτυακές εφαρμογές χρησιμοποιούν αρχεία (web documents) συγκεκριμένου format (HTML, Javascript, JSON, XML) . Ο browser «τρέχει» τα αρχεία αυτά δημιουργώντας το γραφικό περιβάλλον μέσω του οποίου μπορεί πλέον να αλληλεπιδρά με το σύστημα.

Κατά τη συνεδρία (επίσκεψη/λήψη της σελίδας) ο web browser εκτελεί και εμφανίζει τις σελίδες λειτουργώντας ως ένας παγκόσμιος client για οποιαδήποτε διαδικτυακή εφαρμογή σε οποιοδήποτε μηχάνημα στον κόσμο.

Στα αρχικά στάδια του παγκόσμιου ιστού, κάθε σελίδα έφτανε στο χρήστη ως ένα στατικό έγγραφο (αρχείο υπερκειμένου). Ακόμα κι έτσι ο χρήστης είχε μια εμπειρία αλληλεπίδρασης, μέσω των συνεχών λήψεων των εγγράφων (web documents) που περιείχαν τις καταχωρήσεις του. Όμως κάθε αλλαγή του χρήστη απαιτούσε ένα ταξίδι μετ' επιστροφής στο server έτσι ώστε να ανανεωθεί η σελίδα και να υπάρχει η αλληλεπίδραση αυτή.

Το 1995 το Netscape εισήγαγε στο παγκόσμιο ιστό τη JavaScript, μια γλώσσα που έδινε τη δυνατότητα αλλαγής του περιβάλλοντος χρήστη (user interface) σε μια συγκεκριμένη σελίδα μέσω ενεργειών του χρήστη με το ποντίκι ή το πληκτρολόγιο ή σε καθορισμένα χρονικά διαστήματα (client-side scripting). Οι προγραμματιστές πρόσθεταν στη σελίδα δυναμικά στοιχεία, με τα οποία γινόταν επαλήθευση στις καταχωρήσεις των χρηστών ή προσθαφαίρεση στοιχείων της σελίδας χωρίς την ανάγκη αποστολής δεδομένων στο server για να δημιουργήσει και να αποστείλει τη σελίδα στον client εκ νέου.

Το 1996, η Macromedia παρουσίασε το Flash μία πρόσθετη λειτουργία για τους browsers (plug-in) η οποία επέτρεπε την αναπαραγωγή κινούμενων εικόνων από υπολογιστή (vector animation player) που υπήρχαν ενσωματωμένες σε μία ιστοσελίδα. Μέσω μιας γλώσσας προγραμματισμού σεναρίων (scripting language) αυτοματοποιούνταν οι αλληλεπιδράσεις και οι αλλαγές στη μεριά του client χωρίς επικοινωνία με το server.

Το 1999, η ιδέα του "web application" παρουσιάστηκε στη Java (Java Servlet). Ως εκείνο το σημείο η JavaScript και το XML είχαν αναπτυχθεί αλλά η τεχνολογία Ajax δεν είχε δημιουργηθεί και το XMLHttpRequest (an API in the form of an object whose methods transfer data between a web browser and a web server) αντικείμενο είχε πρόσφατα εισαχθεί στον Internet Explorer 5 ως ActiveX αντικείμενο.

Το 2005, με τη δημιουργία του Ajax, εφαρμογές όπως το Gmail ολοένα και περισσότερο γίνονταν πιο διαδραστικές. Πλέον το script που βρίσκεται σε μια σελίδα δίνει τη δυνατότητα επικοινωνίας του client με το server για να αποθηκεύσει ή να ανακτήσει δεδομένα χωρίς να «κατεβάζει» κάθε φορά ολόκληρη τη σελίδα από αυτόν.

Το 2011, η ολοκληρώθηκε η HTML5 που παρείχε πολυμεσικές δυνατότητες χωρίς την ανάγκη πρόσθετων από την πλευρά του χρήστη.Η HTML5 επίσης εμπλούτισε το σημασιολογικό περιεχόμενο των web documents. Τα APIs και το document object model (DOM) είναι πλέον θεμελιώδη χαρακτηριστικά της HTML5. Το WebGL API έδειξε το δρόμο για προηγμένα τρισδιάστατα γραφικά στο HTML5 canvas (στοιχείο της HTML5 που χρησιμοποιείται για την απεικόνιση γραφικών στις ιστοσελίδες) με τη χρήση JavaScript. Όλα αυτά έχουν ιδιαίτερη σημασία για τη δημιουργία πραγματικών, ανεξαρτήτων browser rich web applications (εφαρμογών που υποστηρίζουν υψηλό επίπεδο διαλειτουργικότητας και υποστήριξη όλων των ειδών των περιεχομένων που μπορούν να μεταδοθούν μέσω του διαδικτύου).

2.2.1 Διαδικτυακές εφαρμογές σε κινητές συσκευές

Υπάρχουν αρκετές μέθοδοι ανάπτυξης διαδικτυακών εφαρμογών όταν αυτές σκοπεύουν σε περιβάλλοντα smartphones και tablets:

- Το Responsive Design χρησιμοποιείται για να κάνει μία ιστοσελίδα να προσαρμόζεται δυναμικά και «on the fly» (άμεσα) σε κάθε διάσταση και προσανατολισμό (orientation) οθόνης. Αυτός ο σχεδιασμός ευνοεί και τη χρήση touchscreens.
- Progressive Web Apps είναι εφαρμογές οι οποίες είναι ένας συνδυασμός (υβρίδιο) από μία ιστοσελίδα και μια mobile εφαρμογή. Αυτός ο νέος τύπος εφαρμογών συνδυάζει τις δυνατότητες που προσφέρουν οι σύγχρονοι browsers με τα πλεονεκτήματα της εμπειρίας χρήσης σαν μια native mobile εφαρμογή. Επομένως πρόκειται για web apps που ανοίγουν από τον browser σας σε μια ξεχωριστή σελίδα, έχοντας μια αντίστοιχη συμπεριφορά με μια εγκατεστημένη native app, ενσωματώνοντας πλήρη χαρακτηριστικά όπως οι ειδοποιήσεις (notifications). Αυτό συνεπάγεται και εξοικονόμηση χώρου στις κινητές συσκεύες και γλιτώνει το χρήστη από τη διαδικασία της εγκατάστασης από το app store.

2.2.2 AJAX

Η ανάπτυξη διαδικτυακών εφαρμογών εξελίσσεται τα τελευταία χρόνια και οι σύγχρονες ανάγκες επικεντρώνονται σε πλήρως δυναμικές εφαρμογές ιστού, οι οποίες πρέπει να ενσωματώνουν λειτουργικά συστατικά πραγματικού χρόνου.

Ajax (asynchronous JavaScript and XML) είναι ένα σύνολο τεχνικών και τεχνολογιών που χρησιμοποιείται στον πελάτη (client) μέρος μιας διαδικτυακής εφαρμογής και μπορεί να ανταλλάξει δεδομένα με ένα διακομιστή (server) ασύγχρονα, ενημερώνοντας μία σελίδα ή ένα μέρος της δυναμικά, χωρίς να είναι απαραίτητη η επαναφόρτωση της (web document). Η XML σήμανση που υπάρχει στο ακρωνύμιο τις περισσότερες φορές «αντικαθίσταται» από το JSON ένα άλλο data format, πιο κατάλληλο για ανταλλαγή δεδομένων.

Με την έννοια Reverse Ajax εννοούμε την ικανότητα αποστολής δεδομένων από τον διακομιστή στον πελάτη. Για να επιτευχθεί αυτό μπορούν να χρησιμοποιηθούν οι παρακάτω τεχνικές:

HTTP polling και JSONP polling. Το polling συνεπάγεται ένα απλό HTTP Ajax αίτημα από τον πελάτη στον διακομιστή για να λάβει κάποια δεδομένα. Τα αιτήματα γίνονται σύμφωνα με ένα συγκεκριμένο χρονικό μεσοδιάστημα (interval). Το μειονέκτημα έγκειται στο γεγονός ότι η δημιουργία συχνών αιτημάτων στον διακομιστή, για το βέλτιστο δυνατό χρόνο απόκρισής του, καταναλώνει άσκοπα εύρος ζώνης και επεξεργαστικούς πόρους. Το JSONP polling είναι ουσιαστικά το ίδιο, με τη διαφορά ότι μπορούν να γίνουν αιτήματα προς διαφορετικούς τομείς (cross-domain requests). Ένα JSONP αίτημα έχει ως παράμετρο (προαιρετικά) ένα callback με το περιεχόμενο της απάντησης (JSON), ένα κομμάτι κώδικα (συνάρτηση, function) που εκτελείται με την απόκριση του διακομιστή.

Piggyback polling. Πρόκειται για μία πολύ πιο έξυπνη μέθοδο από το απλό polling καθώς καταργεί τα άχρηστα αιτήματα (που δεν επιστρέφουν δεδομένα). Τα αιτήματα στέλνονται όποτε ο πελάτης θέλει να λάβει δεδομένα από τον διακομιστή και όχι ανά χρονικό μεσοδιάστημα. Η διαφορά έγκειται στην απάντηση η οποία χωρίζεται στα αιτούμενα δεδομένα από τον πελάτη και σε τυχόν ενημερώσεις/συμβάντα. Η μη άσκοπη αποστολή αιτήσεων στο server εξοικονομεί πόρους, όμως η απαίτηση αιτήματος από την πλευρά του πελάτη για αποστολή τυχόν ενημερώσεων από το server σε αυτόν.

Comet model. Στο μοντέλο εφαρμογών ιστού Comet στέλνεται ένα αίτημα στον διακομιστή, το οποίο διατηρείται «ενεργό» για ένα μεγάλο χρονικό διάστημα μέχρι να έρθει η χρονοπρογραμματισμένη λήξη του (timeout) ή να υπάρξει συμβάν στον διακομιστή (άρα αποστολή ενημέρωσης στον πελάτη). Μετά τη λήξη, ένα νέο αίτημα Ajax αποστέλλεται για τυχόν επόμενες ενημερώσεις/συμβάντα από το server. Με το Comet κάθε πελάτης έχει πάντα έναν ανοικτό δίαυλο επικοινωνίας με τον διακομιστή. Λόγω του χρόνου που παραμένουν ανοιχτά τα αιτήματα, απαιτούνται ειδικές πρόσθετες δυνατότητες στον διακομιστή (σε σχέση με το polling) για το χειρισμό τους.

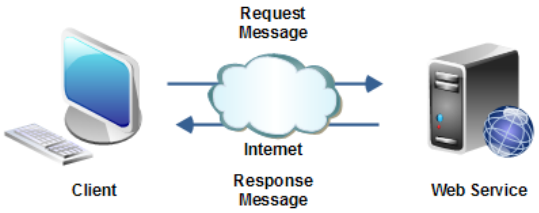
Στο Comet σε streaming mode (σε λειτουργία μετάδοσης σε πραγματικό χρόνο), χρησιμοποιείται ένα μόνο αίτημα «μακράς διάρκειας», χρησιμοποιώντας την ίδια σύνδεση. Δύο συνήθεις τεχνικές για comet με streaming αποτελούν τα κρυφά iframes (Forever Iframes) και η multi-part τύπου περιεχομένου αιτήματος του XMLHttpRequest αντικειμένου. Η πρώτη μέθοδος είναι απλή και στηρίζεται σε HTML Tags και συγκεκριμένα iframes τα οποία είναι συνδεδεμένα με το μονοπάτι/διεύθυνση του server, μέσω του attribute src (source), το οποίο επιστρέφει τυχόν συμβάντα. Δεν προσφέρει αξιόπιστο έλεγχο σφαλμάτων ή δυνατότητα παρακολούθησης της σύνδεσης γιατί ο browser χειρίζεται τη σύνδεση και τα δεδομένα μέσω ενός html tag (iframe). Η δεύτερη μέθοδος είναι πιο αξιόπιστη και οι ενημερώσεις/απαντήσεις από τα συμβάντα στον διακομιστή είναι υποχρεωτικά multi-part τύπου περιεχομένου. Κάθε φορά που υπάρχει συμβάν στον διακομιστή στέλνεται multi-part τύπου περιεχομένου απάντηση, μέσω της ίδιας σύνδεσης. Στο server, το multi-part αίτημα μπαίνει σε ασύγχρονη λειτουργία και έτσι διακόπτεται η σύνδεση και η ασύγχρονη «μορφή» του αιτήματος καταχωρείται σε μία δομή δεδομένων (π.χ. ουρά). Όταν υπάρξει κάποιο συμβάν, στέλνονται απαντήσεις/ενημερώσεις μέσω των «αδρανών» συνδέσεων σε όλους τους πελάτες που έχουν αιτηθεί να είναι ενήμεροι για τα συμβάντα στον διακομιστή.

Στο Comet σε HTTP long polling mode η σύνδεση διατηρείται ανοικτή από τον διακομιστή και αμέσως μετά από ένα συμβάν η απάντηση στέλνεται στον πελάτη, η σύνδεση κλείνει και μία νέα ανοίγει από τον πελάτη για επόμενες ενημερώσεις/συμβάντα. Για την υλοποίηση της συγκεκριμένης τεχνικής μπορούν να χρησιμοποιηθούν script tags ή ένα XMLHttpRequest αντικείμενο. Ομοίως με τα κρυφά iframes η επικοινωνία με script tags επιτυγχάνεται μέσω ενός <script> (HTML script tag) στη σελίδα. Ο διακομιστής θα διακόψει τη σύνδεση μέχρι κάποιο συμβάν, θα στείλει το περιεχόμενο του script πίσω στη μεριά του πελάτη και εκείνος θα ανοίξει ένα νέο <script> για τις επόμενες ενημερώσεις/συμβάντα. Ναι μεν είναι απλό στην υλοποίηση αλλά αντιμετωπίζει παρόμοια προβλήματα με τα Forever Iframes. Στο long polling με XMLHttpRequest ο διακομιστής θέτει σε αδράνεια τα αιτήματα, όπως και στο streaming mode, και μόλις υπάρξει συμβάν στέλνει τις ενημερώσεις μέσω των «αδρανών» συνδέσεων και ακολούθως τις τερματίζει. Ο πελάτης με τη σειρά του, λαμβάνει την απάντηση και στέλνει νέο αίτημα στον διακομιστή.

2.3 Διαδικτυακές υπηρεσίες και διαλειτουργικότητα

2.3.1 Διαδικτυακές υπηρεσίες

Μία διαδικτυακή υπηρεσία ή υπηρεσία ιστού (web service) είναι ένα σύνολο συνδεόμενων λειτουργιών ενός συστήματος λογισμικού, το οποίο καλείται «προγραμματιστικά» για την ανταλλαγή δεδομένων ανάμεσα σε εφαρμογές μέσω ενός δικτύου, όπως το διαδίκτυο, ανεξαρτήτως λειτουργικού συστήματος και γλώσσας προγραμματισμού. Μπορεί να χρησιμοποιεί REST (representational state transfer) ή/και πρότυπα μηνύματα XML για την επικοινωνία και οι περιγράφεται από ένα XML πρότυπο, το WSDL (Web Services Description Language).



Εικόνα 2: Επικοινωνία πελάτη – διακομιστή

Οι υπηρεσίες ιστού συνδυάζουν την ισχύ των δύο ευρέως διαδεδομένων τεχνολογιών: του XML, γενική γλώσσα περιγραφής δεδομένων και του HTTP, το δημοφιλέστερο πρωτόκολλο μεταφοράς που υποστηρίζεται από όλους τους περιηγητές και διακομιστές διαδικτύου (Εικόνα 2). Η βασική διαφορά των υπηρεσιών διαδικτύου με παλιότερες τεχνολογίες (J2EE, Corba κ.α.) είναι η τυποποίηση που τις χαρακτηρίζει. Η τεχνολογία των υπηρεσιών ιστού βασίζεται στο πρότυπο XML που υποστηρίζεται από την πλειοψηφία των «μεγάλων παικτών» ευνοώντας την επικοινωνία ανάμεσα στις εταιρείες με ένα ενιαίο πρότυπο. Μία διαδικτυακή υπηρεσία έχει τα εξής χαρακτηριστικά:

- Είναι διαθέσιμη στο διαδίκτυο ή σε ιδιωτικά δίκτυα
- Χρησιμοποιεί τυποποιημένο XML σύστημα επικοινωνίας
- Είναι ανεξάρτητη γλώσσας, πλατφόρμας και διαλειτουργική
- Αυτοπεριγράφεται με κοινή γραμματική XML
- Είναι εντοπίσιμη μέσω ενός μηχανισμού αναζήτησης καταλόγων υπηρεσιών
- Σχετίζεται χαλαρά με τον καταναλωτή όσον αφορά τις αλλαγές στη λειτουργικότητά της και το χρόνο απόκρισης της.

2.3.2 Αρχιτεκτονική Υπηρεσιών

Οι επιχειρήσεις έχουν επενδύσει σημαντικά σε εφαρμογές και συστήματα λογισμικού για την εύρυθμη λειτουργία τους. Συχνά υπάρχουν ετερογένειες στις υποδομές τους, όσον αφορά τα συστήματα λογισμικού, οι οποίες οφείλονται στα διαφορετικά λειτουργικά συστήματα και γλώσσες προγραμματισμού.

Η μεταβλητότητα είναι ένα ακόμα θέμα που έχουν να αντιμετωπίσουν οι επαγγελματίες της πληροφορικής και οι επιχειρήσεις. Η παγκοσμιοποίηση, το ηλεκτρονικό επιχειρείν και το ηλεκτρονικό εμπόριο φέρνουν ραγδαίες αλλαγές και οδηγούν σε σκληρό ανταγωνισμό ανάμεσα στις επιχειρήσεις. Νέες υπηρεσίες και προϊόντα δημιουργούνται και διαδέχονται άλλα με πολύ ταχείς ρυθμούς. Οι αυξανόμενες απαιτήσεις των καταναλωτών που προκύπτουν σε αυτό το δυναμικό και συνεχώς αναπτυσσόμενο περιβάλλον ενισχύονται και από τις βελτιώσεις των τεχνολογιών.

Επομένως, οι επιχειρήσεις πρέπει να εναρμονίζονται με την τεχνολογία, να είναι ευέλικτες στις αλλαγές του επιχειρηματικού περιβάλλοντος και να παρέχουν νέες και βελτιωμένες υπηρεσίες, αξιοποιώντας τις ήδη υπάρχουσες υποδομές.

Οι διαδικτυακές υπηρεσίες και η SOA προσέγγιση προσφέρουν καλύτερες ευκαιρίες για την ανάπτυξη και ολοκλήρωση επιχειρησιακών εφαρμογών με πρόσθετα οφέλη όπως τα μειωμένα κόστη ανάπτυξης, επαναχρησιμοποίηση πόρων και διαλειτουργικότητα.

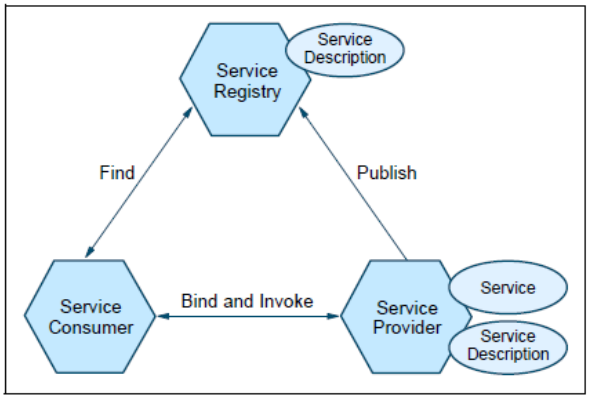
Το μοντέλο των διαδικτυακών υπηρεσιών βασίζεται στην υπηρεσιοστρεφή αρχιτεκτονική (SOA, service-oriented architecture). Αντιπροσωπεύει την εξέλιξη των αρχιτεκτονικών πελάτη-διακομιστή (client

server). Σε αυτές τις αρχιτεκτονικές οι λειτουργίες της επιχειρηματικής λογικής , της διαχείρισης δεδομένων και της διεπαφής χρήστη είναι ξεχωριστές, δίνοντας τη δυνατότητα σε κάθε μία λειτουργία να υλοποιείται με τις βέλτιστες τεχνολογίες. (5)

Η υπηρεσιοστρεφής προσέγγιση διασπά την επιχειρηματική λογική (application/business logic) ή άλλες λειτουργίες (modularization) και τις παρουσιάζει ως υπηρεσίες ιστού για τους πελάτες. Το κλειδί σε αυτές τις υπηρεσίες είναι η χαλαρή σύζευξή τους (loose coupling). Η διεπαφή της υπηρεσίας είναι ανεξάρτητη με την υλοποίηση της, επομένως οι προγραμματιστές και οι υπεύθυνοι ολοκλήρωσης λογισμικού μπορούν να δημιουργήσουν εφαρμογές χρησιμοποιώντας μία σύνθεση υπηρεσιών χωρίς να γνωρίζουν τεχνικές λεπτομέρειες της υλοποίησης τους. (6)

Το SOA βασίζεται στις αλληλεπιδράσεις τριών οντοτήτων (Εικόνα 3). Οι πάροχοι των υπηρεσιών(service providers) δημοσιεύουν τις υπηρεσίες (publishing) σε ένα κατάλογο υπηρεσιών (service registry/broker). Μετά ο αιτών της υπηρεσίας (service requester) ή καταναλωτής της υπηρεσίας (service consumer), δηλαδή ο πελάτης μπορεί να βρει τις υπηρεσίες (finding) που καλύπτουν τις ανάγκες του, αναζητώντας τις περιγραφές τους στον κατάλογο. Αφού ο consumer βρει την υπηρεσία που τον ενδιαφέρει, δεσμεύεται (binding) με αυτήν στον provider και καλεί την υπηρεσία (invocation).

(7)



Εικόνα 3: Ρόλοι και λειτουργίες των υπηρεσιών ιστού.

Ο πάροχος θεωρείται ο ιδιοκτήτης της υπηρεσίας. Υλοποιεί τις εφαρμογές τις οποίες προσφέρει με τη μορφή διεπαφών των υπηρεσιών και μπορεί να δημοσιεύσει τις παρεχόμενες υπηρεσίες στο μητρώο υπηρεσιών. Το μητρώο υπηρεσιών είναι ένας κατάλογος που περιέχει πληροφορίες για τη χρήση των υπηρεσιών. Οι πάροχοι περιγράφουν τις υπηρεσίες τους και οι πελάτες αναζητούν στον κατάλογο αυτό τις υπηρεσίες που χρειάζονται. Οι καταναλωτές της υπηρεσίας μπορούν να ανακτήσουν και επιπλέον πληροφορίες που αφορούν την προγραμματιστική διεπαφή της υπηρεσίας για τη δέσμευση σε αυτήν και τελικά την επίκλησή της.

2.3.3 Πλεονεκτήματα μοντέλου Διαδικτυακών Υπηρεσιών

Με το μοντέλο SOA μπορεί να επιτευχθεί η διαλειτουργικότητα σε υπηρεσίες και εφαρμογές, καθώς και η προσαρμοστικότητα σε αλλαγές τεχνολογιών και απαιτήσεων. Σε σχέση με παλαιότερες τεχνολογίες και πρακτικές οι υπηρεσίες διαδικτύου προσφέρουν:

- Χαλαρά συνδεδεμένες εφαρμογές και διαφάνεια τοποθεσίας
- Δυνατότητα σύνδεσης της εφαρμογής με ετερογενείς πλατφόρμες και γλώσσες προγραμματισμού
- Ευθυγράμμιση της πληροφορικής με τις ανάγκες των επιχειρήσεων
- Επαναχρησιμοποίηση υφιστάμενων υποδομών και εφαρμογών

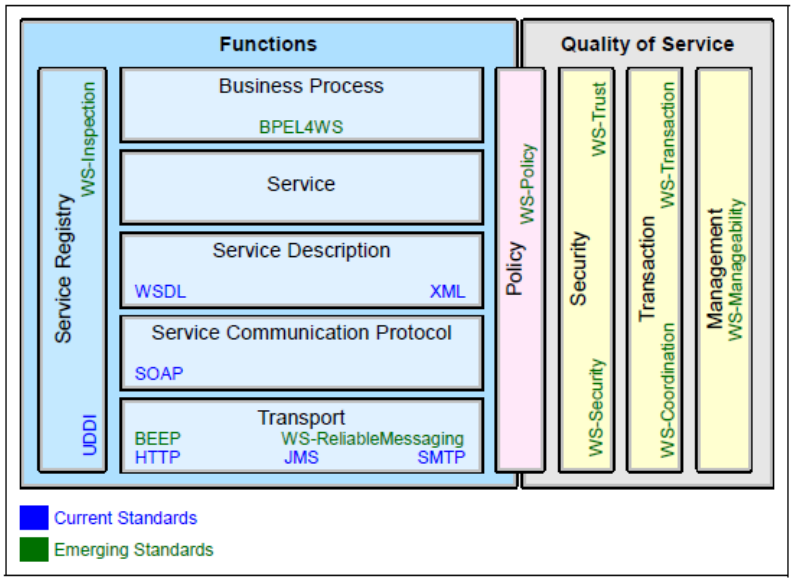
- Μεγαλύτερη επεκτασιμότητα και ευελιξία εφαρμογών και συστημάτων
- Ευκολία στην επικοινωνία και απλότητα πρωτοκόλλων
- Ευκολότερη συντήρηση
- Παράλληλη και μη εξαρτώμενη ανάπτυξη εφαρμογών
- Μειωμένα κόστη ανάπτυξης και ολοκλήρωσης εφαρμογών
- Λιγότερες εξαρτώμενες από συγκεκριμένους προμηθευτές εταιρείες (vendor lock-ins)

2.3.4 Βασικές Τεχνολογίες των Διαδικτυακών Υπηρεσιών

Οι διαδικτυακές υπηρεσίες προσφέρουν μία προσέγγιση κατανεμημένων συστήματος για την ενοποίηση ετερογενών εφαρμογών στο Διαδίκτυο. Οι προδιαγραφές τους είναι εντελώς ανεξάρτητες λογισμικού και έτσι προσδίδουν χαλαρή σύζευξη ανάμεσα στον καταναλωτή και τον πάροχο. Οι ανοικτές τεχνολογίες (πρότυπα) που βασίζονται τα web services είναι:

- eXtensible Markup Language (XML)
- Simple Object Access Protocol (SOAP)
- Universal Description, Discovery and Integration (UDDI)
- Web Services Description Language (WSDL)

Χρησιμοποιώντας ανοικτά πρότυπα οι εταιρείες μπορούν να υλοποιούν υπηρεσίες χωρίς να γνωρίζουν για τους καταναλωτές και διασφαλίζεται διαλειτουργικότητα των υπηρεσιών.



Εικόνα 4: Υπηρεσίες ιστού και στοίβα πρωτοκόλλων SOA.

Το επίπεδο μεταφοράς στην δομή του SOA (Εικόνα 4) είναι υπεύθυνο για την ανταλλαγή μηνυμάτων ανάμεσα στις εφαρμογές. Πιο συγκεκριμένα, στη μεταφορά αιτήσεων από τον καταναλωτή στον πάροχο και τις απαντήσεις από τον πάροχο στον καταναλωτή. Το πιο διαδεδομένο πρωτόκολλο που χρησιμοποιείται είναι το HTTP (hypertext transfer protocol).

2.3.4.1 SOAP

Το SOAP (Simple Object Access Protocol) είναι ένα πρωτόκολλο που βασίζεται σε XML αρχεία για την ανταλλαγή μηνυμάτων μέσω HTTP (Εικόνα 4). Το SOAP δίνει τη δυνατότητα σε ένα client application να τρέξει απομακρυσμένα μεθόδους μια υπηρεσίας. Το πρωτόκολλο SOAP έχει τα εξής χαρακτηριστικά:

- Το SOAP είναι απλό. Άρα το κόστος και η πολυπλοκότητα υλοποίησης μειώνονται αισθητά.
- Το SOAP είναι ανεξάρτητο από πλατφόρμα και γλώσσα προγραμματισμού οπότε μπορεί να χρησιμοποιηθεί για επικοινωνία μεταξύ εφαρμογών γραμμένων για διαφορετικές πλατφόρμες και σε διαφορετικές γλώσσες προγραμματισμού.
- Το SOAP είναι ευέλικτο. Χρησιμοποιεί πρότυπα πρωτόκολλα όπως το HTTP και το SMTP ως μέσα μεταφοράς οπότε μπορεί να χρησιμοποιηθεί στο διαδίκτυο και να διαπερνά τύχη προστασίας χωρίς συμβιβασμούς στην ασφάλεια της υποδομής μιας επιχείρησης. Αυτό αυτομάτως μειώνει και σε ορισμένες περιπτώσεις εξαλήφει το κόστος υποδομής αφού οι περισσότερες επιχειρήσεις σήμερα έχουν και τον εξοπλισμό και την τεχνογνωσία για τη χρήση του διαδικτύου.
- Το SOAP είναι επεκτάσιμο. Αν και δεν προσφέρει τόσες πολλές λειτουργίες όσο άλλες τεχνολογίες όπως το CORBA και το DCOM επιτρέπει σε άλλα πρότυπα να το επεκτείνουν παρέχοντας υπηρεσίες που λείπουν από αυτό. Αυτό το χαρακτηριστικό αποδείχθηκε ίσως το σημαντικότερο γιατί επάνω του βασίζονται πολλές αναπτυσσόμενες τεχνολογίες των web services που προσφέρουν υπηρεσίες όπως «αξιοπιστία» (reliability), «δρομολόγηση» (routing) και «ασφάλεια» (security). (8)

2.3.4.2 Περιγραφή υπηρεσίας: WSDL

Η Web Services Description Language (WSDL) αντιπροσωπεύει το επίπεδο της περιγραφής της υπηρεσίας στη στοίβα πρωτοκόλλων των διαδικτυακών υπηρεσιών (Εικόνα 4).

Μία διαδικτυακή υπηρεσία μπορεί να θεωρηθεί ως ένα «σύνολο τελικών σημείων» τα οποία ανταλλάσσουν και επεξεργάζονται μηνύματα που περιέχουν πληροφορία προσανατολισμένη στσ έγγραφα ή στις διαδικασίες (που καλούνται απομακρυσμένα). (7)

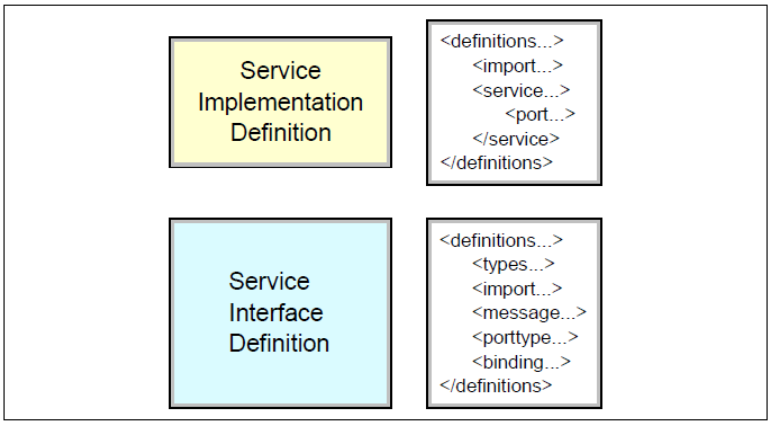
Περιγράφει το τι κάνει μία υπηρεσία, το πως καλούνται οι διαδικασίες της και που μπορεί ο ενδιαφερόμενος να τη βρει. Έχει δημιουργήσει ξεχωριστούς ορισμούς και ορολογία για τον προσδιορισμό μιας υπηρεσίας, το τελικό σημείο επικοινωνίας της υπηρεσίας, την απαιτούμενη μορφή των μηνυμάτων που λαμβάνει και στέλνει η υπηρεσία και έναν αφηρημένο τρόπο για να δηλώνει μια σύνδεση (binding) σε ένα συγκεκριμένο πρωτόκολλο και μορφή δεδομένων. (9)

Ό,τι ορίζεται στο WSDL έγγραφο περιγράφει απλά παραμέτρους και περιορισμούς για την επικοινωνία κατά το χρόνο εκτέλεσης (ορισμός διεπαφής – service interface definition). Οι περιγραφές αυτές δεσμεύονται σε ένα συγκεκριμένο πρωτόκολλο και μορφή δεδομένων για τη δημιουργία ενός τελικού σημείου επικοινωνίας της υπηρεσίας (ορισμός υλοποίησης της υπηρεσίας – service implementation definition). Παρέχεται η δυνατότητα ορισμού μιας σύνδεσης (binding) η οποία «δένει» ένα σύνολο ορισμών για μηνύματα σε ένα πρωτόκολλο ή μορφή δεδομένων. Η WSDL επεξηγεί επεκτάσεις σύνδεσης για χρήση των SOAP 1.1, HTTP GET, HTTP POST και MIME.

Η WSDL χρησιμοποιεί τα παρακάτω στοιχεία για τον ορισμό της υπηρεσίας: (8)(Εικόνα 4)

- ***Types*** - ένα περίβλημα για ορισμούς τύπων δεδομένων χρησιμοποιώντας ένα σύστημα τύπων (όπως για παράδειγμα το XML Schema).

- ***Message*** - ένας περιγραφικός ορισμός των δεδομένων που ανταλλάσσονται.
- ***Operation*** - μία περιγραφή μίας λειτουργίας που υποστηρίζεται από μία υπηρεσία
- ***Port Type*** - ένα περιγραφικό σύνολο από λειτουργίες που υποστηρίζονται από ένα ή περισσότερα τελικά σημεία.
- ***Binding*** - ένα συγκεκριμένο πρωτόκολλο και μορφή δεδομένων για ένα συγκεκριμένο τύπο τελικών σημείων (port type).
- ***Port*** - ένα μοναδικό τελικό σημείο που ορίζεται σαν συνδυασμός μίας σύνδεσης (binding) και μιας διεύθυνσης δικτύου.
- ***Service*** - μία συλλογή από σχετικά τελικά σημεία.



Εικόνα 5: Συστατικά περιγραφής των υπηρεσιών με WSDL

Η σήμανση που χρησιμοποιείται σε ένα αρχείο WSDL για να περιγράψει μορφές μηνυμάτων βασίζεται στο πρότυπο του XML Schema το οποίο σημαίνει ότι είναι ταυτόχρονα ανεξάρτητη από γλώσσα προγραμματισμού και βασισμένη σε πρότυπα. Αυτό το γεγονός την κάνει κατάλληλη για να περιγράψει διεπαφές web services οι οποίες είναι προσβάσιμες από μία μεγάλη ποικιλία πλατφορμών και γλωσσών προγραμματισμού. Επιπλέον, εκτός του ότι περιγράφει τα περιεχόμενα των μηνυμάτων, η WSDL ορίζει πού είναι διαθέσιμη μία υπηρεσία και ποια πρωτόκολλα επικοινωνίας χρησιμοποιούνται για να επικοινωνήσουμε με αυτή την υπηρεσία. Αυτό σημαίνει ότι ένα αρχείο WSDL ορίζει όλα όσα χρειάζονται για να γράψουμε ένα πρόγραμμα το οποίο να λειτουργεί με ένα web service. (8)(Εικόνα 6)



Εικόνα 6: Παράδειγμα WSDL εγγράφου. Με κόκκινα βέλη υποδεικνύονται τα στοιχεία του εγγράφου

2.3.4.3 Ανακάλυψη της υπηρεσίας: UDDI

Το UDDI (Universal Description, Discovery and Integration) αντιπροσωπεύει το επίπεδο της ανακάλυψης/εύρεσης της υπηρεσίας από τον ενδιαφερόμενο, στη στοίβα πρωτοκόλλων των διαδικτυακών υπηρεσιών. Ορίζει έναν τρόπο δημοσίευσης και ανακάλυψης/εύρεσης πληροφοριών για τις υπηρεσίες ιστού.

Είναι ένα πρωτόκολλο βασισμένο στο SOAP που ορίζει τους τρόπους επικοινωνίας πελατών – UDDI μητρώων. Παρέχει APIs για την αναζήτηση και δημοσίευση πληροφοριών σε ένα μητρώο, καθώς και για την αναπαραγωγή των δεδομένων ανάμεσα στους κόμβους. Επίσης ορίζει έγγραφα που περιγράφουν τις επιχειρήσεις και τις υπηρεσίες (σε XML μορφή).

Το UDDI Business Registry (UBR), είναι ένα «σύννεφο» (cloud) παγκόσμιας κατανομής, και αποτελείται από πολλούς κόμβους, των οποίων τα δεδομένα συγχρονίζονται μέσω της αναπαραγωγής τους (δημιουργία αντιγράφων - replications) σε άλλους κόμβους. Κάθε χειριστής κόμβος παρέχει τις ίδιες ακριβώς πληροφορίες και ποιότητα υπηρεσίας με οποιοδήποτε άλλο χειριστή κόμβο. Ο πρώτος κόμβος στον οποίο θα εισαχθεί περιεχόμενο γίνεται και ο κύριος κάτοχός του. Οποιαδήποτε πιθανή αλλαγή των δεδομένων πρέπει να λάβει χώρα στον κύριο κάτοχο – κόμβο στον οποίο τα δεδομένα εισήχθησαν. Επιπλέον μία εταιρεία μπορεί να διαθέτει ιδιωτικούς κόμβους, των οποίων τα δεδομένα δε θα έχουν καμία σχέση με το UBR, άρα και κανένα συγχρονισμό με τους κόμβους του. Όμως, μπορεί να υπάρξει συγχρονισμός του περιεχομένου ανάμεσα σε ιδιωτικούς χειριστές κόμβους

μέσα στο «ιδιωτικό σύννεφο» της εταιρείας και δυνατότητα δημοσίευσης της υπηρεσίας και δοκιμών σε ασφαλές περιβάλλον.

Ο ορισμός που δίνει το OASIS για το Universal Description, Discovery and Integration είναι ο παρακάτω:

"Τα web services έχουν νόημα μόνο όταν δυνητικοί χρήστες μπορούν να βρουν πληροφορίες ικανές ώστε να επιτρέψουν την εκτέλεσή τους. Το Universal Description Discovery & Integration (UDDI) εστιάζει στον καθορισμό ενός συνόλου από υπηρεσίες που θα υποστηρίζουν την περιγραφή και την ανακάλυψη:

1. Των εταιριών, των οργανισμών και άλλων παρόχων web services
2. Των web services, που είναι διαθέσιμες
3. Και των τεχνικών διεπαφών οι οποίες μπορούν να χρησιμοποιηθούν ώστε να έχει κάποιος χρήστης πρόσβαση σε αυτές τις υπηρεσίες.

Βασισμένο σε ένα κοινό σύνολο από βιομηχανικά πρότυπα, συμπεριλαμβανομένων των HTTP, XML, XML Schema και SOAP το UDDI παρέχει μία διαλειτουργική, θεμελιώδη υποδομή για ένα περιβάλλον λογισμικού προσανατολισμένο στις υπηρεσίες τόσο για δημόσια διαθέσιμες υπηρεσιές όσο και για υπηρεσίες που εκτίθενται μόνο εσωτερικά ενός οργανισμού." (8)

Τα δεδομένα που συλλέγονται σε αυτό χωρίζονται σε λευκές, κίτρινες και πράσινες σελίδες (pages). Οι λευκές σελίδες περιλαμβάνουν γενικές πληροφορίες μιας εταιρείας (όνομα, περιγραφή, διεύθυνση κ.α.). Οι κίτρινες σελίδες περιλαμβάνουν δεδομένα κατηγοριοποίησης είτε για την εταιρεία είτε για την υπηρεσία που προσφέρει (τομέας βιομηχανίας κ.α.) και οι πράσινες τεχνικές πληροφορίες για μία υπηρεσία (μία παραπομπή μίας εξωτερικής προδιαγραφής και μία διεύθυνση για την κλήση της υπηρεσίας).

Έχει πολλές διαφορετικές χρήσεις, ανάλογα με το ποιος το χρησιμοποιεί. Από την οπτική ενός επιχειρηματικού αναλυτή, το UDDI είναι παρόμοιο με μία μηχανή αναζήτησης για στο διαδίκτυο, που απευθύνεται σε επιχειρηματικές διαδικασίες. Ένας επιχειρηματικός αναλυτής μπορεί να δει σε έναν ή περισσότερους UDDI καταλόγους τις επιχειρήσεις που εκθέτουν υπηρεσίες και τις προδιαγραφές των υπηρεσιών αυτών. Οι προγραμματιστές χρησιμοποιούν το UDDI API για την έκδοση μιας υπηρεσίας και για να υποβάλλουν ερωτήσεις στον κατάλογο με διάφορα κριτήρια για την ανακάλυψη/εύρεση υπηρεσιών. (9)

Ένας κατάλογος UDDI, μπορεί να παρομοιαστεί με ένα «χρυσό οδηγό» και έχει δύο ειδών πελάτες: επιχειρήσεις που θέλουν να δημοσιεύσουν μια υπηρεσία (και τις διεπαφές της), και πελάτες που θέλουν να χρησιμοποιήσουν συγκεκριμένες υπηρεσίες και συνδέονται προγραμματιστικά με αυτές.

2.3.5 Αρχιτεκτονική REST

Η αρχιτεκτονική REST (Representational State Transfer) συστήθηκε από τον Roy Fielding το 2000 στη διδακτορική διατριβή του. Είναι μία μέθοδος σχεδίασης κατανεμημένων συστημάτων. Δεν αποτελεί πρότυπο αλλά μία περιγραφή μίας stateless αρχιτεκτονικής πελάτη – διακομιστή η οποία παρέχει μία απλή διεπαφή επικοινωνίας με χρήση XML και HTTP. Κάθε πόρος προσδιορίζεται με ένα URI (Uniform Resource Identifiers) και η χρήση του HTTP επιτρέπει την επικοινωνία μέσω εντολών GET, POST, PUT και DELETE. (10)

Η REST έχει κερδίσει την ευρεία αποδοχή στο διαδίκτυο, ως εναλλακτική του SOAP και WSDL μοντέλου. Η καλύτερη απόδειξη της αλλαγής αυτής είναι η υιοθέτηση της REST αρχιτεκτονικής από

τους ισχυρότερους παρόχους υπηρεσιών Web 2.0, όπως Yahoo, Google, Facebook, οι οποίες απέρριψαν τις διεπαφές που βασίζονταν σε αυτά τα πρότυπα για χάρη ενός απλότερου -όσον αφορά τη χρήση και προσανατολισμένου στους πόρους μοντέλου για τις υπηρεσίες τους. (11)

Η αρχιτεκτονική REST βασίζεται στις εξής σχεδιαστικές αρχές:

Αναγνώριση πόρων μέσω URIs. Μία RESTful διαδικτυακή υπηρεσία εκθέτει ένα σύνολο πόρων που προσδιορίζουν τους στόχους της αλληλεπίδρασης (επικοινωνίας) με του πελάτες της. Οι πόροι ταυτοποιούνται μέσω των URIs, τα οποία παρέχουν μία παγκόσμιας εμβέλειας διεύθυνση των δεδομένων και εντοπισμό της υπηρεσίας. Αυτό το μοναδικό URI στο οποίο ο πόρος είναι συνδεδεμένος αποτελεί το αναγνωριστικό του. Ο ρόλος τους είναι παρόμοιος με τη δομή καταλόγου αρχείων σε διάφορα λειτουργικά συστήματα να δείχνουν τον τρέχων κατάλογο και τον κατάλογο-γονέα. (12)

Ομοιομορφία διεπαφής. Οι πόροι διακινούνται χρησιμοποιώντας ένα σταθερό σύνολο HTTP μεθόδων (PUT, GET, POST, DELETE). Η PUT δημιουργεί έναν νέο πόρο, ο οποίος μπορεί να διαγραφεί αργότερα χρησιμοποιώντας το DELETE. Η GET ανακτά την τρέχουσα κατάσταση ενός πόρου και η POST μεταφέρει μία νέα κατάσταση σε αυτόν.

Αυτοπεριγραφικά μηνύματα. Οι πόροι είναι ανεξάρτητοι της αναπαράστασής τους έτσι ώστε το περιεχόμενό τους να είναι προσβάσιμο σε ποικιλία μορφών δεδομένων (HTML, XML, plain text, PDF, JPEG κ.α.). Ο πόρος διαθέτει μεταδεδομένα τα οποία χρησιμοποιούνται για έλεγχο caching (αποθήκευση δεδομένων σε κρυφή μνήμη), ανίχνευση σφαλμάτων μετάδοσης, διαπραγμάτευση της κατάλληλης μορφής μηνύματος και επαλήθευση ταυτότητας/έλεγχο πρόσβασης.

Αλληλεπιδράσεις με επίβλεψη κατάστασης (stateful) μέσω υπερσυνδέσμων. Κάθε αλληλεπίδραση με έναν πόρο είναι χωρίς επίβλεψη κατάστασης (stateless). Υπάρχουν αρκετές τεχνικές για ανταλλαγή κατάστασης, π.χ. επαναδιατύπωση URI, cookies, και κρυφά πεδία μίας φόρμας συμπλήρωσης. Η κατάσταση μπορεί να ενσωματωθεί σε μηνύματα απόκρισης για την υπόδειξη μελλοντικών καταστάσεων της αλληλεπίδρασης. (13)

2.3.6 SOAP vs REST

Το SOAP παρέχει σε σχέση με τη REST τα εξής πλεονεκτήματα:

- Ανεξαρτησία από γλώσσα προγραμματισμού, πλατφόρμα και πρωτοκόλλου μεταφοράς μηνυμάτων (η REST απαιτεί HTTP)
- Αποδίδει καλά σε κατανεμημένα επιχειρησιακά περιβάλλοντα (η REST βασίζεται σε επικοινωνία από σε σημείο σε σημείο)
- Προτυποποίηση
- Παρέχει σημαντική επεκτασιμότητα με τα WS πρότυπα (π.χ. WS-Security για ασφάλεια)
- Ενσωματωμένος χειρισμός σφαλμάτων (σε περίπτωση σφάλματος αναφέρεται και ο κωδικός σφάλματος βάσει του οποίου μπορεί να γίνει αυτόματος χειρισμός)
- Αυτοματισμός διαδικασιών όταν χρησιμοποιείται με ορισμένες γλώσσες (π.χ. .NET)

Η REST είναι πιο απλό στη χρήση και πιο ευέλικτο. Τα πλεονεκτήματα του σε σχέση με το SOAP είναι τα εξής:

- Δεν απαιτείται ακριβός εξοπλισμός για τη χρήση της υπηρεσίας (επικοινωνία με την υπηρεσία)
- Ευκολότερη στη χρήση από τεχνικής άποψης

- Αποδοτικότερη (μπορεί να χρησιμοποιεί μικρότερες μορφές για τα μηνύματα, σε σχέση με το SOAP, το οποίο χρησιμοποιεί XML για όλα τα μηνύματα)
- Ταχύτερη, καθώς δεν απαιτείται εκτεταμένη επεξεργασία
- Πιο ευθυγραμμισμένη με άλλες τεχνολογίες ιστού από σχεδιαστικής άποψης

2.3.7 Διαλειτουργικότητα Υπηρεσιών

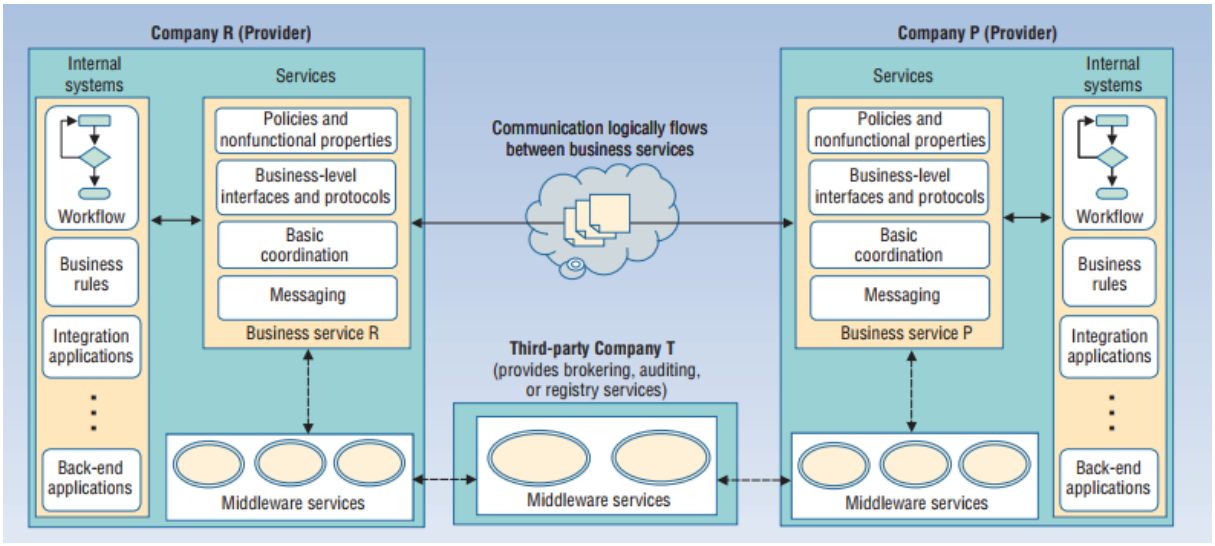
Η διαλειτουργικότητα των υπηρεσιών που εκτελούνται σε διαφορετικές πλατφόρμες και είναι υλοποιημένες σε διαφορετικές προγραμματιστικές γλώσσες αποτελεί την υπόσχεση των υπηρεσιών ιστού. Για παράδειγμα, μία υπηρεσία που υλοποιήθηκε σε .NET, μπορεί να κληθεί από οποιαδήποτε client εφαρμογή που έχει υλοποιηθεί σε κάποια άλλη γλώσσα όπως η Java. Οι υπηρεσίες ιστού, μέσω των διεπαφών τους, «περιτυλίγουν» τις υπάρχουσες εφαρμογές έτσι ώστε να έχουν πρόσβαση οι προγραμματιστές, με τυποποιημένες γλώσσες και πρωτόκολλα.

Η τυποποίηση απλουστεύει τη διαλειτουργικότητα. Οι εφαρμογές, αντί να αλληλεπιδρούν με ετερογενή συστήματα, με διαφορετικές προδιαγραφές (πρωτόκολλο μεταφοράς, μορφή δεδομένων κλπ.) το καθένα, αλληλεπιδρούν με περισσότερο ομοιογενή συστήματα. Ειδικότερα, τα πρότυπα των υπηρεσιών ιστού προάγουν την υποστήριξη αποκεντρωμένων και χαλαρά συνδεδεμένων αλληλεπιδράσεων. (14)

Τα ζητήματα διαλειτουργικότητας των υπηρεσιών είναι πολύπλευρα και επομένως μπορούν να προσεγγιστούν διαφορετικά.

2.3.7.1 Επίπεδα ολοκλήρωσης

Οι ομοιότητες των δικτύων υπολογιστών με τις υπηρεσίες ιστού καθιστούν χρήσιμη την διαστρωματική («κατακόρυφη» στοίβα από επίπεδα) προσέγγιση των θεμάτων διαλειτουργικότητας των υπηρεσιών, ώστε να αποθέσουμε τα μέρη του προβλήματος σε διαφορετικά επίπεδα αφαίρεσης. Η τυπική δομή ξεκινά από το κατώτερο στρώμα-το οποίο είναι απαραίτητο από τις περισσότερες λειτουργίες-και κατευθύνεται στα ανώτερα στρώματα τα οποία χρειάζονται ανάλογα με την εφαρμογή. (14) Στην παρακάτω εικόνα βλέπουμε τα επίπεδα. (Εικόνα 7)



Εικόνα 7: Επίπεδα ολοκλήρωσης υπηρεσίας

Η μηνυματοδότηση (messaging). Βασική παραδοχή διαλειτουργικότητας αποτελεί ο καθορισμός ενός πρωτοκόλλου μεταφοράς της πληροφορίας, ανεξάρτητα από το περιεχόμενο και τη δομή της. Το SOAP (Simple Object Access Protocol) είναι το πιο διαδεδομένο πρωτόκολλο στο επίπεδο της ανταλλαγής μηνυμάτων. Επιπλέον προδιαγραφές του πρωτοκόλλου είναι το WS-Reliability και WS-Security τα οποία παρέχουν επιπλέον ιδιότητες (αξιοπιστία, ασφάλεια κ.α.) και εγγυήσεις στο messaging.

Το επίπεδο του συντονισμού (basic coordination). Καθορίζει τις απαιτήσεις και τις ιδιότητες σχετικά με ένα σύνολο ανταλασσόμενων μηνυμάτων μεταξύ δύο ή παραπάνω εταίρων (για τη διεκπεραίωση μιας συναλλαγής για παράδειγμα). Για παράδειγμα, δύο υπηρεσίες ενδέχεται να χρειαστούν συντονισμό όταν θέλουν να παρέχουν ατομικότητα (atomicity), δηλαδή μία κατανεμημένη συναλλαγή/καθήκον που δε μπορεί να υποδιαιρεθεί, και πρέπει να επεξεργάζεται ολόκληρη ή καθόλου. (15) Βασίζεται στο two-phase commit, ένα μηχανισμό που διασφαλίζει την ακεραιότητα των δεδομένων σε μία κατανεμημένη συναλλαγή. Ο συντονιστής κόμβος «ζητάει» από όλους τους συμμετέχοντες κόμβους να εφαρμόσουν μία αλλαγή (την ίδια) σε μία βάση δεδομένων. Εάν αυτό δεν είναι δυνατό ζητείται αναίρεση των αλλαγών. (16) Άλλες προδιαγραφές του επιπέδου αυτού περιλαμβάνουν κεντρικοποιημένα συστήματα ταυτοποίησης (Federated Identity Management) όπως τα WS-Federation και Liberty Alliance.

Το στρώμα διεπαφών και πρωτοκόλλων επιχειρηματικού-επιπέδου (business-level interfaces and protocols). Είναι επιφορτισμένο με τον καθορισμό της λειτουργικότητας μιας υπηρεσίας. Η διεπαφή (interface) μιας υπηρεσίας αποτελείται από το σύνολο των διαδικασιών που υποστηρίζει και τα μηνύματα που ανταλλάσσουν, λαμβάνουν και δημιουργούν αυτές. Η γλώσσα περιγραφής των υπηρεσιών (Web Services Definition Language) είναι η πιο διαδεδομένη γλώσσα για τον ορισμό των διεπαφών διαδικτυακών υπηρεσιών (web service interfaces). Σε αυτό το επίπεδο μπορούν να υπάρξουν οι εξής ετερογένειες:

- Συντακτικές (syntactical)-διαφορές ανάμεσα σε υπογραφές της διαδικασίας-δηλαδή ονόματα διαδικασιών και ονόματα εισερχόμενων/εξερχόμενων μηνυμάτων.
- Δομικές (structural)-διαφορές στους τύπους ή στις τιμές των αναμενόμενων εισερχόμενων/εξερχόμενων μηνυμάτων.
- Σημασιολογικές (semantic)-διαφορές στην ερμηνεία του νοήματος ενός στοιχείου δεδομένων ή της λειτουργίας μιας διαδικασίας. (14)

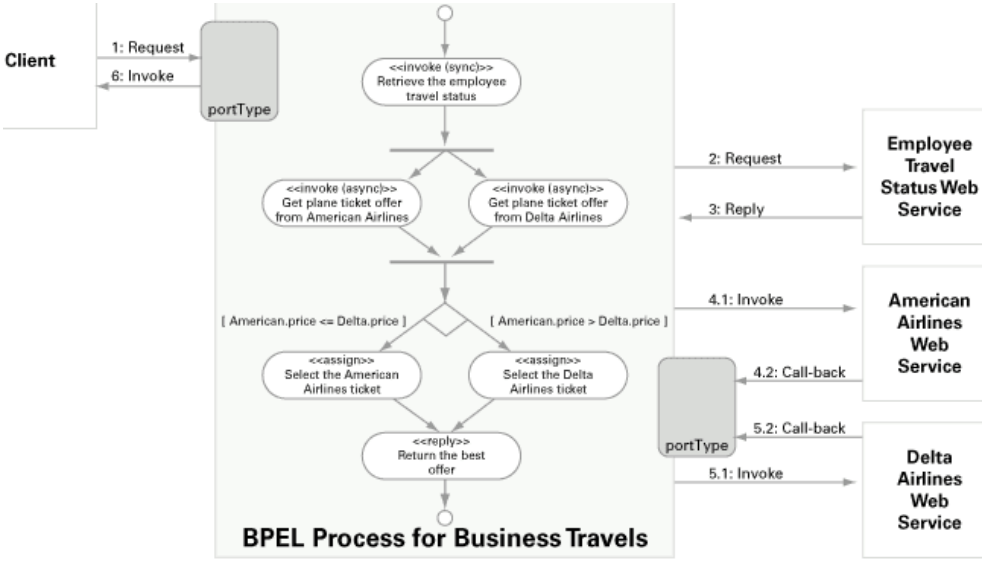
Το επιχειρηματικό πρωτόκολλο καθορίζει τις επιτρεπτές ακολουθίες κλήσεων των διαδικασιών (ανταλλαγή μηνυμάτων). Μπορούν να προκύψουν ετερογένειες μεταξύ των επιχειρηματικών πρωτοκόλλων λόγω διαφορετικών περιορισμών στη σειρά των μηνυμάτων ή επειδή μια υπηρεσία αναμένει (ή στέλνει) ένα μήνυμα όπου ο εταίρος δεν είναι έτοιμος να στείλει (ή να λάβει). Για παράδειγμα ένας αποστολέας μπορεί να περιμένει μια επιβεβαίωση για το μήνυμα που έστειλε όμως ο πάροχος της υπηρεσίας να μην εκδίδει τέτοιου είδους μηνύματα. Η WS-Business Process Language (BPEL) είναι η τρέχουσα κορυφαία γλώσσα για τον καθορισμό επιχειρηματικών πρωτοκόλλων. Βασίζεται στο XML και καθορίζει τον τρόπο που μια επιχειρηματική διαδικασία, η οποία αλληλεπιδρά με άλλες υπηρεσίες, θα εκτελεστεί. (Εικόνα 8)

Πολιτικές και μη-λειτουργικές ιδιότητες (policies and nonfunctional properties). Η περιγραφή μια υπηρεσίας είναι δυνατόν να περιλαμβάνει πολιτικές (π.χ. πολιτικές απορρήτου) κι άλλες μη λειτουργικές ιδιότητες (π.χ. περιγραφές για την ποιότητα της υπηρεσίας - QoS, όπως ο χρόνος απόκρισης) όπου οι εφαρμογές-πελάτες που αλληλεπιδρούν με την υπηρεσία πρέπει να γνωρίζουν. Για παράδειγμα το WS-Policy είναι ένα πρότυπο που επιτρέπει στις διαδικτυακές υπηρεσίες να εκφράζουν (γνωστοποιούν) τις πολιτικές τους (ασφαλείας, ποιότητας της υπηρεσίας κ.α.). Αυτές οι προδιαγραφές εξυπηρετούν επίσης τη διαλειτουργικότητα όμως είναι πιο χρήσιμες στις διαδικασίες ανακάλυψης και σύνδεσης της

υπηρεσίας (web service discover/binding, εύρεση της υπηρεσίας και σύνδεση-αλληλεπίδρασή της με εφαρμογές) παρά (κατά τη διάρκεια) το χρόνο αλληλεπίδρασης (interaction time). Παρόμοιες ετερογένειες με αυτές του επιπέδου επιχειρηματικών διεπαφών και πρωτοκόλλων (business-level interfaces and protocols layer) μπορούν να προκύψουν κι εδώ.

Γενικά, η διαλειτουργικότητα σχετίζεται και με επιχειρηματικές αλλά και με ενδιάμεσες υπηρεσίες διαδικτύου (middleware web services). Οι υπηρεσίες που έχουν να κάνουν με την υποστήριξη της επιχείρησης, (π.χ. διαχείριση τιμολογίων) είναι οι βασικοί αποστολείς ή δέκτες μηνυμάτων της υπηρεσίας. Η επικοινωνία ανάμεσα στις επιχειρηματικές υπηρεσίες πολλές φορές απαιτεί εξωτερικές (third-party) ή ενδιάμεσες υπηρεσίες που παρέχουν ανακάλυψη υπηρεσιών, διαχείριση, συντονισμό κ.α. Οι ενδιάμεσες υπηρεσίες παρεμβάλλονται ανάμεσα στις επιχειρηματικές υπηρεσίες, συμβάλλοντας στη λειτουργία-επικοινωνία τους. Ορισμένα πρωτόκολλα είναι έτσι σχεδιασμένα ώστε να χρησιμοποιούνται αποκλειστικά από τις ενδιάμεσες υπηρεσίες. Για παράδειγμα το SOAP και οι βιβλιοθήκες που ενσωματώνει υποστηρίζουν την αποστολή και λήψη μηνυμάτων, οπότε οι προγραμματιστές δε χρειάζεται να γνωρίζουν τις λεπτομέρειες αυτές για την υλοποίηση web services.

Στην την άλλη πλευρά, έχουμε προδιαγραφές και γλώσσες των υψηλότερων επιπέδων αφαίρεσης. Για παράδειγμα, οι προγραμματιστές που θέλουν να αναπτύξουν μια υπηρεσία για την εξυπηρέτηση μίας επιχειρηματικής διαδικασίας, χρησιμοποιούν γλώσσες επιχειρηματικού πρωτοκόλλου για να δημιουργήσουν προδιαγραφές σύμφωνα με τις περιγραφές της υπηρεσίας. Επιπλέον, πρέπει να γνωρίζουν προδιαγραφές πρωτοκόλλων άλλων web services, ώστε το business service τους να συνεργάζεται επιτυχώς με τα αυτά (web services). Γενικά, λίγες προδιαγραφές απευθύνονται αποκλειστικά σε προγραμματιστές επιχειρηματικών υπηρεσιών.



Εικόνα 8: BPEL Process για οργάνωση ταξιδιού

Η διάκριση μεταξύ γλωσσών και προδιαγραφών είναι προφανής. Όμως, σε κάποιες περιπτώσεις η ενιαιοποίηση (γενίκευση) προδιαγραφών με γλώσσες στοχεύει στην δημιουργία μιας γλώσσας με την οποία οι προγραμματιστές θα μπορούν να ορίζουν προδιαγραφές. Για παράδειγμα, προγραμματιστές ή κοινοπραξίες τυποποίησης μπορούν να χρησιμοποιήσουν πρωτόκολλα όπως WSDL, WS-BPEL και WS-Policy για να παρέχουν προδιαγραφές σε διεπαφές, επιχειρηματικά πρωτόκολλα και πολιτικές. Σε άλλες περιπτώσεις, οι προσπάθειες προτυποποίησης δημιουργούν άμεσα προδιαγραφές. Για παράδειγμα,

πρωτόκολλα όπως τα SOAP και WS-Transactions είναι προδιαγραφές, και όχι γλώσσες που χρησιμοποιούνται για τη δημιουργία άλλων προδιαγραφών.

Όπως στα περισσότερα πλαίσια εργασίας που υποστηρίζεται διαλειτουργικότητα, οι διαδικτυακές υπηρεσίες παρέχουν προδιαγραφές για τα χαμηλότερα επίπεδα, και γλώσσες για τα υψηλότερα επίπεδά της. Για αυτό το λόγο παρατηρείται ότι οι προδιαγραφές των υψηλότερων στρωμάτων είναι πιο εξειδικευμένες με το service (service-specific). Οι προγραμματιστές μπορούν να περιγράψουν μια υπηρεσία συσχετίζοντάς την με ένα σύνολο προδιαγραφών σε διαφορετικά επίπεδα. Οι κοινοπραξίες τυποποίησης παρέχουν άμεσα ορισμένες από αυτές τις προδιαγραφές, ενώ οι κοινοπραξίες συγκεκριμένων τομέων, όπως οι κατασκευαστές αυτοκινήτων, αλλά και προγραμματιστές υπηρεσιών, δημιουργούν τις άλλες. *

Υπάρχουν δύο ειδών καταναλωτές της πληροφορίας της διαλειτουργικότητας μιας υπηρεσίας: οι φυσικοί χρήστες (άνθρωποι) και τα προγράμματα εφαρμογών. Σε γενικές γραμμές οι φυσικοί χρήστες χρειάζονται τις πληροφορίες της διαλειτουργικότητας, δηλαδή την περιγραφή μιας υπηρεσίας, για να αποφασίσουν αν θα χρησιμοποιήσουν την υπηρεσία και να σχεδιάσουν πως θα αναπτύξουν client λογισμικό που θα μπορεί να επικοινωνεί απρόσκοπτα με την υπηρεσία. Μπορούν να αξιοποιούν επίσημες και ανεπίσημες περιγραφές. Συχνά η περιγραφή μιας υπηρεσίας περιέχει ένα μέρος της πληροφορίας σε φυσική γλώσσα. Το WDSL περιλαμβάνει μια τυπική περιγραφή της μορφής των μηνυμάτων και των ονομάτων των λειτουργιών του και πληροφορίες που περιγράφουν το ρόλο κάθε λειτουργίας.

Οι εφαρμογές «διαβάζουν» τις προδιαγραφές της υπηρεσίας οι οποίες παρέχονται σε κάποια τυποποιημένη γλώσσα όπως η WS-BPEL (Εικόνα 8). Γενικά οι εφαρμογές μπορούν να χρησιμοποιούν αποκλειστικά επίσημες προδιαγραφές. Συγκεκριμένα σημεία του κύκλου ανάπτυξης μιας εφαρμογής μπορούν να αυτοματοποιηθούν, με την πρόσβαση πλατφορμών/εφαρμογών ανάπτυξης υπηρεσιών (π.χ. Eclipse) σε επίσημες προδιαγραφές, όπως για παράδειγμα στη δημιουργία σκελετών/στελεχών (skeleton/stubs) από τις προδιαγραφές του WSDL εγγράφου.

Αντίθετα, εφαρμογές που υποστηρίζουν σύνδεση (binding), και ειδικότερα δυναμική σύνδεση, δεν «διαβάζουν» τις προδιαγραφές άρα δεν ενδιαφέρονται για το βαθμό τυπικότητάς τους αλλά για το αν πράγματι μια υπηρεσία τηρεί μια συγκεκριμένη προδιαγραφή. Για παράδειγμα, το client πρόγραμμα μιας ναυτιλιακής εταιρείας χρειάζεται να ξέρει αν μία υπηρεσία υποστηρίζει το SOAP 1.2 και μία συγκεκριμένη τυποποιημένη διεπαφή που έχει αναπτυχθεί από την κοινοπραξία τυποποίησης των ναυτιλιακών εταιρειών. Η ανάγνωση και η ανάλυση του WSDL εγγράφου δεν συμβαίνει κατά το χρόνο εκτέλεσης. Εφόσον μία υπηρεσία δεν υποστηρίζει την ίδια διεπαφή και τα πρωτόκολλα τότε δεν έχει νόημα η δυναμική σύνδεση με αυτά, και συνεπώς η πρόσβαση στο WSDL έγγραφο.

Τρέχουσες ερευνητικές προσπάθειες στοχεύουν στην αυτοματοποίηση της διαλειτουργικότητας μια υπηρεσίας. Για παράδειγμα, η client εφαρμογή να επιτρέπεται να «διαβάζει» τα χαρακτηριστικά του παρόχου τη στιγμή της σύνδεσης κι αν διαπιστώσει ότι η υπηρεσία που εντοπίστηκε δεν είναι συμβατή, να έχει τη δυνατότητα να επιλύει τις διαφορές.

Οι βασικές λειτουργίες του κύκλου ζωής των υπηρεσιών ιστού διακρίνονται σε τρεις φάσεις. Στο πρώτο στάδιο, τη σύνδεση (binding), μια εφαρμογή αναλύει περιγραφές «υποψήφιων» υπηρεσιών για να αποφασίσει αν μπορεί να αλληλεπιδράσει με κάποια από αυτές (interaction). Γενικά ο πελάτης (developer ή application) χρειάζεται όλες τις προδιαγραφές για να ξέρει αν η υπηρεσία μπορεί να υποστηρίξει μία συγκεκριμένη προδιαγραφή. Επίσης κατά τη φάση του binding ελέγχονται και οι μη-λειτουργικές ιδιότητες. Οι πολιτικές των υπηρεσιών ,χρησιμοποιώντας το WS-Policy, συνήθως δηλώνουν τις πληροφορίες αυτές μαζί με υποστήριξη συγκεκριμένων προδιαγραφών και μη-λειτουργικών ιδιοτήτων.

Μετά το στάδιο του binding, η φάση της αλληλεπίδρασης (interaction) περιλαμβάνει όλες τις λειτουργίες για την επίκληση της υπηρεσίας (service invocation), την ανταλλαγή (και επαλήθευση) μηνυμάτων και τον συντονισμό ενός συνόλου συσχετιζόμενων μηνυμάτων που αφορούν την αξιοπιστία, την ασφάλεια και τις συναλλαγές.

Η λειτουργία της διαχείρισης (management) αφορά την παρακολούθηση (monitoring) των αλληλεπιδράσεων μίας υπηρεσίας, την παροχή της, και τη διαχείριση των σχέσεων και των παραμέτρων της. Προδιαγραφές διαχείρισης, όπως η WSDM (Web Services Distributed Management), προσφέρουν τον ορισμό ορατών διεπαφών, όπως για παράδειγμα έναν ψηφιακό πίνακα ελέγχου (dashboard) για την παρακολούθηση της κατάστασης ή της απόδοσης υπηρεσιών και ενδεχομένως την επανεκκίνηση των υπηρεσιών σε περίπτωση αποτυχίας. (17)

2.4 Μεταδεδομένα και APIs

Ο όρος μεταδεδομένα (metadata) αναφέρεται σε «δεδομένα που παρέχουν πληροφορίες για άλλα δεδομένα» (18). Χρησιμοποιείται για τον εμπλουτισμό υπαρχόντων δεδομένων και αποτελούν ετικέτες για πληροφοριακούς πόρους παρέχοντας σημαντικές πληροφορίες για αυτούς. Τα μεταδεδομένα δεν είναι χαρακτηριστικά μόνο ψηφιακών πόρων αλλά μπορούν να αφορούν κι άλλα είδη πόρων, π.χ. ο συγγραφέας και ο τίτλος ενός βιβλίου είναι μεταδεδομένα για το βιβλίο.

Υπάρχουν τρία βασικές κατηγορίες μεταδεδομένων: (19)(Εικόνα 9)

- *Περιγραφικά (descriptive).* Αποτυπώνουν περιγραφικά στοιχεία που είναι καταγεγραμμένα πάνω στο τεκμήριο και περιλαμβάνουν στοιχεία όπως τον τίτλο, το συγγραφέα, την περίληψη και τις λέξεις κλειδιά. Αποτελούν τα βασικά πεδία καταγραφής και για την εγγραφή τους χρησιμοποιούνται οδηγίες και κανόνες (AACR2) ή αντλούν από προτυποποιημένες λίστες ονομάτων φυσικών προσώπων, οργανισμών, συλλογικών οργάνων, γεωγραφικών όρων και θεμάτων, τα λεγόμενα καθιερωμένα αρχεία.
- *Δομικά (structural).* Αποτυπώνουν στοιχεία που αφορούν τη φυσική ή λογική δομή του πληροφοριακού πόρου. Για παράδειγμα η δόμηση ενός βιβλίου σε κεφάλαια, οι πίνακες περιεχομένων, τα έγγραφα ενός αρχειακού φακέλου κ.λπ.. παρουσιάζουν τη σειρά των θεμάτων ή και φυσικών στοιχείων μέσα σε ένα τεκμήριο και τη σχέση τους με άλλους πληροφοριακούς πόρους. Επίσης, συνδέουν ένα τεκμήριο με άλλα τεκμήρια ώστε να αποτελούν λογικές μονάδες (π.χ. πληροφορίες που συνδέουν την κάθε εικόνα που υπάρχει σε ένα βιβλίο με τις υπόλοιπες εικόνες του βιβλίου).
- *Διαχειριστικά (administrative)* Αποτυπώνουν στοιχεία που χρησιμοποιούνται για τη διαχείριση του τεκμηρίου ή τον έλεγχο πρόσβασης σε αυτό. Μπορεί να περιλαμβάνουν πληροφορίες για το πώς ψηφιοποιήθηκε ένα τεκμήριο, το μορφότυπο φύλαξης, τα δικαιώματα πνευματικής ιδιοκτησίας, και πληροφορίες για τη μακροπρόθεσμη συντήρηση των ψηφιακών αρχείων. Υποενότητες των διαχειριστικών μεταδεδομένων αποτελούν:
    - Τα πεδία που αφορούν τα πνευματικά δικαιώματα και τα δικαιώματα χρήσης του τεκμηρίου.
    - Τα πεδία που αφορούν τη συντήρηση και διατήρηση του τεκμηρίου συμπεριλαμβανομένης και της ψηφιακής διατήρησης του πληροφοριακού πόρου.

Κατηγορία μεταδεδομένων	**Παραδείγματα**
Περιγραφικά: αναζήτηση-εύρεση	Συγγραφέας, τίτλος, λέξεις-κλειδιά, έκδοση, θέματα

Δομικά: αποθήκευση-παρουσίαση	Συστατικά μέρη, σχέσεις συστατικών, πλοήγηση, είδη αντικειμένων κ.λπ.
Διαχειριστικά: διοίκηση-διαφύλαξη	Ημερομηνία δημιουργίας, τελευταία τροποποίηση, υπεύθυνος, οδηγίες χειρισμού
*Όροι και προϋποθέσεις *υποκατηγορία	Δικαιώματα χρήσης, πρόσβασης, εκτύπωσης, αντιγραφής, επαναχρησιμοποίησης, αναδημοσίευσης, σχετικό κόστος

Εικόνα 9: Κατηγορίες μεταδεδομένων.

2.4.1 Χρησιμότητα Μεταδεδομένων

Τα μεταδεδομένα μεταφέρουν πληροφορία για τεκμήρια σε δομημένη μορφή και μπορούν να χρησιμοποιηθούν σε απαιτητικές υπολογιστικές λειτουργίες, όπως η προηγμένη αναζήτηση. Για παράδειγμα ένας χρήστης μπορεί να αναζητήσει και να ανακτήσει τεκμήρια ως προς ένα κοινό χαρακτηριστικό (εννοιολογικό) που καταγράφεται στα μεταδεδομένα τους. Με τον τρόπο αυτό βελτιστοποιείται ο χρόνος απόκρισης των μηχανών αναζήτησης αφού η αναζήτηση γίνεται στα μεταδεδομένα των τεκμηρίων, που έχει ευρετηριάσει (indexing) η μηχανή. Η προσθήκη μεταδεδομένων στα τεκμήρια καθιστά την ανάκτηση των αποτελεσμάτων ακριβέστερη και πιο αυτοματοποιημένη. Η αναζήτηση με βάση μία δομή μεταδεδομένων επιτρέπει την αναζήτηση συγκεκριμένων όρων σε διακριτά στοιχεία και γλιτώνει τον χρήστη από περίπλοκες και χρονοβόρες χειροκίνητες διαδικασίες φιλτραρίσματος πληροφοριών.

Η διαλειτουργικότητα αποτελεί τη βασική αρχή για την παραγωγή μεταδεδομένων στα πληροφοριακά συστήματα. Για αυτό το λόγο αξιοποιούνται ανοιχτά κοινά πρότυπα που μπορούν να επικοινωνήσουν και να λειτουργήσουν με διαφορετικά λειτουργικά συστήματα και περιηγητές διαδικτύου (browsers).

Η διαδικασία του εμπλουτισμού του περιεχομένου (τεκμήρια) με μεταδεδομένα καλείται τεκμηρίωση. Για την τεκμηρίωση των τεκμηρίων χρησιμοποιείται ένα πρότυπο ή σχήμα, δηλαδή μία δομή περιγραφής των τεκμηρίων με τα μεταδεδομένα. Σε γενικές γραμμές, η ποικιλία και οι ιδιαιτερότητες των πληροφοριακών πόρων δυσχεραίνει την ανάπτυξη προτύπων τεκμηρίωσης ευρείας αποδοχής. Παρόλα αυτά, υιοθετούνται όσο το δυνατόν πιο αποδεκτά και χρησιμοποιούμενα πρότυπα τεκμηρίωσης, διευκολύνοντας την διαθεσιμότητα των πόρων σε όλο και περισσότερες, πιο ετερογενείς και απαιτητικότερες κοινότητες χρηστών.

Ένα από τα πιο διαδεδομένα πρότυπα σήμερα, είναι το Dublin Core, ένα γενικό περιγραφικό σύστημα τεκμηρίωσης για τη διαχείριση ηλεκτρονικών αντικειμένων. Άλλα παρεμφερή με αυτό έχουν είναι τα EAD, MARC και το TEI. Κοινό και πολύ σημαντικό χαρακτηριστικό των διαδεδομένων προτύπων τεκμηρίωσης είναι η άμεση συσχέτισή τους μέσω των κοινών στοιχείων (elements) που περιλαμβάνει η δομή δεδομένων τους, όπως π.χ. ο δημιουργός ή το όνομα του αντικειμένου. (20) Με αυτό τον τρόπο μπορεί να διαμοιραστούν ηλεκτρονικά αντικείμενα, ετερογενώς τεκμηριωμένων ψηφιακών συλλογών (20), αντιστοιχίζοντας τα στοιχεία του ενός σχήματος με τα στοιχεία του άλλου σχήματος (schema crosswalk).

2.4.2 APIs

Μία διεπαφή προγραμματισμού εφαρμογών (API) είναι ένα σύνολο διαδικασιών (ρουτινών), πρωτοκόλλων και εργαλείων για την ανάπτυξη εφαρμογών λογισμικού. Ένα API, ουσιαστικά, καθορίζει τον τρόπο με τον οποίο τα συστατικά λογισμικού αλληλεπιδρούν. Επιπλέον, τα APIs χρησιμοποιούνται κατά τον προγραμματισμό των γραφικών στοιχείων διεπαφής χρήστη (GUI). Ένα API διευκολύνει την

ανάπτυξη ενός προγράμματος παρέχοντας όλα τα δομικά στοιχεία στον προγραμματιστή. (21) Το απαιτούμενο συντακτικό των API κλήσεων περιγράφεται στο αντίστοιχο documentation.

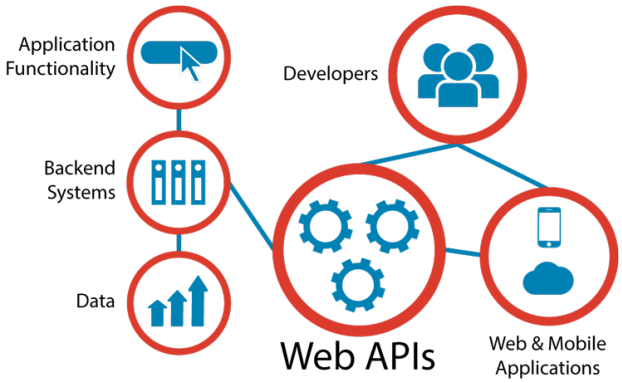
Οι δημόσιες ή ελεύθερες διεπαφές προγραμματισμού εφαρμογών (public/open APIs) καθιστούν ελεύθερη την πρόσβαση στις αντίστοιχες υπηρεσίες από τους καταναλωτές της (προγραμματιστές). Τα περισσότερα ακολουθούν την αρχιτεκτονική REST εκθέτοντας συγκεκριμένα λειτουργικά χαρακτηριστικά, προστατεύοντας την υπόλοιπη εφαρμογή. Τα δημόσια APIs είναι ελεύθερα προσβάσιμα από οπουδήποτε στο Διαδίκτυο. Αντίθετα, τα κλειστά ή ιδιωτικά APIs (private/closed APIs) τα οποία δεν είναι ελεύθερα προσβάσιμα στο Διαδίκτυο, βρίσκονται σε ασφαλή περιορισμένα περιβάλλοντα (κλήσεις μέσω firewalls ή VPN) και συχνά χρησιμοποιούνται για εσωτερικές υπηρεσίες και διαδικασίες.

Για μεγάλο χρονικό διάστημα, τα περισσότερα λογισμικά υπολογιστών που παρείχαν δεδομένα ή υπηρεσίες όπως π.χ. επεξεργασία εικόνας, σχεδιάζονταν και απευθύνονταν αποκλειστικά στον άνθρωπο. Ο άνθρωπος ως τελικός χρήστης κατανάλωνε την πληροφορία μέσω μιας διεπαφής χρήστη (UI) εύκολα και ευχάριστα. Ένα API (application program interface) έχει παρόμοιο ρόλο με μια διεπαφή χρήστη-υπολογιστή με τη διαφορά ότι προορίζεται για κατανάλωση από λογισμικό. Αποτελεί μία ενδιάμεση οντότητα μέσω της οποίας εφαρμογές αλληλεπιδρούν και διαμοιράζονται δεδομένα.

Οι προγραμματιστές εφαρμογών σπάνια ξεκινούν την ανάπτυξη «από το μηδέν». Για παράδειγμα μία εταιρεία λογισμικού που θέλει να δημιουργήσει μία εφαρμογή επεξεργασίας κειμένου για Microsoft Windows, θα χρησιμοποιήσει το παρεχόμενο από τη Microsoft, API αντί να γράψει κώδικα για π.χ. τα γραφικά συστατικά της εφαρμογής και την λειτουργικότητά τους. Τα APIs αποκρύπτουν την πολυπλοκότητα και τη λογική που απαιτείται για μία εργασία, όπως μετάφραση κειμένου, αποθήκευση δεδομένων και στυλιστική διαμόρφωση γραφικών στοιχείων της διεπαφής χρήστη. Υιοθετώντας το μοντέλο αυτό, οι προγραμματιστές αυξάνουν σημαντικά την παραγωγικότητά τους και δε χρειάζεται να «επαναφεύρουν τον τροχό». (22) Εφαρμογές που κάποτε χρειάζονταν μήνες ή χρόνια ανάπτυξης τώρα χρειάζονται ημέρες ή και ώρες. Έτσι μειώνεται ο κύκλος ανάπτυξης των εφαρμογών και οι καταναλωτές λαμβάνουν ενημερώσεις με νέα χαρακτηριστικά συχνότερα.

Ο επαρκής διαχωρισμός της εφαρμογής που διατίθεται για κατανάλωση, με την τεχνική υποδομή από την οποία παρέχεται μία υπηρεσία, χαρίζει τεράστια ευελιξία στον τρόπο με τον οποίο η υπηρεσία προσφέρεται. Για παράδειγμα, οι διακομιστές (υλικό) σε ένα datacenter, μπορεί να μετακινηθούν σε εικονικούς διακομιστές που τρέχουν σε σύννεφο (cloud) εταιρειών όπως η Amazon και το λογισμικό αν είναι γραμμένο, για παράδειγμα, σε Java, μπορεί να μετακινηθεί σε Node.js στο Windows Azure. Όσο η υπηρεσία που προσφέρεται, παραδίδεται στο τελικό σημείο όπως έχει συμφωνηθεί από τα δύο μέρη («API contract»), οι αλλαγές στην υποδομή υποστήριξης περνούν απαρατήρητες από τους χρήστες.

Τα δημόσια τελικά σημεία (endpoints) των APIs σε ένα καθορισμένο σύστημα αιτήματος-ανταπόκρισης (request-response), που έχουν συγκεκριμένη διεύθυνση στον Παγκόσμιο Ιστό και ακολουθούν τη δομή εντολών HTTP (Εικόνα 11) καλούνται συχνά διαδικτυακά (Web) APIs. Ο Παγκόσμιος ιστός, χάρη στις διαδικτυακές είτε mobile εφαρμογές που χτίζονται μέσω κλήσεων (API calls) σε πολλά διαδικτυακά (Web) APIs, έχει μετατραπεί σε ισχυρή προγραμματιστική πλατφόρμα (Εικόνα 10). Οι «σύνθετες» εφαρμογές που δημιουργούνται με διάφορα APIs καλούνται «mashups». Τα τελευταία χρόνια, σημαντικές διαδικτυακές υπηρεσίες «άνοιξαν» τα συστήματά τους στον έξω κόσμο με την υλοποίηση δημόσιων διαδικτυακών APIs (public). Αυτό έχει ως αποτέλεσμα την ενίσχυση της τάσης της ανάπτυξης mashup διαδικτυακών εφαρμογών στον τομέα των προγραμματιστών.



Εικόνα 10: Web APIs

Τα τελικά σημεία (Εικόνα 11) αποτελούν σημαντική πτυχή της αλληλεπίδρασης των εφαρμογών με τα server-side Web APIs καθώς διευκρινίζουν το που βρίσκονται οι προσβάσιμοι πόροι για το λογισμικό τρίτων. Τα τελικά σημεία πρέπει να έχουν στατική διεύθυνση για να εγγυάται η απρόσκοπτη λειτουργία των εφαρμογών που αλληλεπιδρούν με το API.

```
GET /calendar/feeds/default/private/full HTTP/1.1
Content-Type: application/x-www-form-urlencoded
Authorization: AuthSub token="GD32CMCL25aZ-v____8B"
User-Agent: Java/1.5.0_06
Host: www.google.com
Accept: text/html, image/gif, image/jpeg, *; q=.2, */*; q=.2
Connection: keep-alive
```

Εικόνα 11: Επικεφαλίδα HTTP αιτήματος για κλήση του Google Calendar feed service (Calendar API)

2.4.2.1 Web Services vs Web APIs/REST

Αν και τα διαδικτυακά APIs θεωρούνται συνώνυμα των διαδικτυακών υπηρεσιών (web services), πολλές Web 2.0 εφαρμογές έχουν αφήσει πίσω τους τα services βασισμένα σε SOAP και στρέφονται σε RESTful λύσεις.

Τα web services βασίζονται στην XML και κάθε πτυχή τους είναι πολύ καλά καθορισμένη. Υπάρχουν πρότυπα για τις παραμέτρους κλήσης, τις απαντήσεις και την εύρεση της υπηρεσίας. Τα RESTful APIs είναι διαθέσιμα μέσω HTTP μεθόδων και χρησιμοποιούν τη μορφή ανταλλαγής δεδομένων JSON. Σε αντίθεση με τα web services, δεν έχουν προτυποποιημένη λειτουργία. Η ανάπτυξη και η λειτουργία των διαδικτυακών υπηρεσιών είναι περισσότερο πολύπλοκη και απαιτητική σε υλικούς πόρους εξαιτίας των χρονοβόρων μετατροπών που απαιτούνται στα μορφότυπα των δεδομένων (REQUEST->XML->DATA->RESPONSE->XML->VALIDATION->CONVERSION->DATA). (23)

Μεγάλες εταιρείες όπως η Google και η Yahoo έχουν στραφεί και βασιστεί στο REST για τη διάθεση των υπηρεσιών τους.

2.4.2.2 Παραδείγματα διαδεδομένων APIs

Υπάρχουν πλήθος APIs τα οποία προσφέρονται από κατασκευαστές λογισμικού και υπηρεσιών του διαδικτύου. Ενδεικτικά, το **Google Maps APIs** επιτρέπει στους προγραμματιστές να ενσωματώνουν τους χάρτες της Google και τα χαρακτηριστικά (λειτουργικότητα) τους σε ιστοσελίδες και εφαρμογές είτε σε mobile είτε desktop περιβάλλον. Το **YouTube API** επιτρέπει στους προγραμματιστές

να προσθέσουν στις εφαρμογές τους πλήθος χαρακτηριστικών του YouTube, όπως μεταφόρτωση video, διαχείριση λιστών αναπαραγωγής και συνδρομών και άλλα. Επιπλέον προσφέρει προηγμένη αναζήτηση περιεχομένου με συγκεκριμένους όρους αναζήτησης. Το **Flickr API** χρησιμοποιείται από τους προγραμματιστές για να έχουν πρόσβαση στα δεδομένα του υλικού που φιλοξενείται από την υπηρεσία Flickr καθώς και για ενσωμάτωση άλλων δυνατοτήτων, όπως μεταφόρτωση φωτογραφιών και video. Αποτελείται από ένα σύνολο καλούμενων μεθόδων και μερικά τελικά σημεία (endpoints). Το **Twitter** προσφέρει δύο APIs – το REST API επιτρέπει στους προγραμματιστές να έχουν πρόσβαση σε βασικά δεδομένα που περιλαμβάνουν πληροφορίες χρηστών, δεδομένα καταστάσεων και ενημέρωση χρονολογίων και το Search API το οποίο προσφέρει μεθόδους για αλληλεπίδραση με δεδομένα από το Twitter Search και τις τάσεις. Τέλος, το **Amazon's Product Advertising API** χορηγεί στους προγραμματιστές πρόσβαση στη λειτουργικότητα της επιλογής και εύρεσης προϊόντων του Amazon για να τα διαφημίσει και να δημιουργήσει έσοδα από έναν ιστότοπο.

2.5 Mashups και η λογική τους

Το Web 2.0 υποστηρίζει τη συνεργασία των χρηστών καθώς και τη χρήση του ιστού ως μία ενιαία πλατφόρμα που διευκολύνει τις δραστηριότητες των οργανισμών για συνεργασία ανά πάσα στιγμή και οπουδήποτε. Οι προσπάθειες συγκέντρωσης περιεχομένου σε μία ενιαία ιστοσελίδα αναφέρονται με τον όρο «Mashup». Ένα mashup web application αποτελεί μία εφαρμογή που συνδυάζει πολλαπλά σύνολα ροών δεδομένων σε μία ενιαία εμπειρία αλληλεπίδρασης για το χρήστη. (24)

Καθώς ο όγκος και η ποικιλία των πληροφοριών μεγαλώνουν, οι χρήστες απαιτούν ενοποιημένα περιβάλλοντα πρόσβασης, διαχείρισης και επεξεργασίας του περιεχομένου από υπηρεσίες. Τα mashups προσφέρουν τη δυνατότητα ενεργής συμμετοχής στην εύρεση της πληροφορίας και στην προβολή της ώστε να είναι κατανοητή και εύκολα διαχειρίσιμη. Σε αυτό συμβάλλει καθοριστικά η ανάπτυξη των τεχνολογιών του Web 2.0 οι οποίες έβαλαν τέλος στις στατικές ιστοσελίδες και προσφέρουν δυναμικό περιεχόμενο, το οποίο τις περισσότερες φορές καθορίζεται από το χρήστη.

Οι πληροφορίες προέρχονται από πολλές πηγές μέσω των δημόσιων διαδικτυακών APIs/υπηρεσιών τους, RSS feeds και άλλα μέσα. Οι προγραμματιστές χρησιμοποιούν ποικιλία από Web APIs για την ανάπτυξη mashup εφαρμογών ιστού συνδυάζοντας χάρτες, εύρεση γεωγραφικής τοποθεσίας, φωτογραφίες και βίντεο, μετεωρολογικά δεδομένα και ειδήσεις, λίστες προϊόντων και υπηρεσιών κ.ά.

Αντίθετα με τα πρότυπα και τα πρωτόκολλα του Διαδικτύου, τα mashups δεν δημιουργήθηκαν από κάποια συγκεκριμένη σχεδιαστική διαδικασία και επομένως δεν ακολουθούν αυστηρή δόμηση και αρχιτεκτονική.

Για την απεικόνιση του παραδείγματος του "Programmable Web", δηλαδή του παγκοσμίου ιστού ως προγραμματιστική πλατφόρμα, θα χρησιμοποιήσουμε δύο περιπτώσεις χρήσης: Το Web 2.0 και το Enterprise 2.0 που είναι μέρος του Web 2.0 και χρησιμοποιεί εργαλεία και υπηρεσίες με τα χαρακτηριστικά του Web 2.0 σε επιχειρηματικό πλαίσιο (επιχειρήσεις και πελάτες) (25).

2.5.1 Περίπτωση Χρήσης Web 2.0

Στο «κοινωνικό» Web οι τελικοί χρήστες συνεισφέρουν και συνεργάζονται για την παραγωγή περιεχομένου, δημιουργούν αξία σχολιάζοντας, αξιολογώντας και τοποθετώντας ετικέτες και συγκεντρώνουν περιεχόμενο το οποίο διατίθεται από διάφορες πηγές. Ένα απλό παράδειγμα mashup θα ήταν η δημιουργία μιας τελικής διεπαφής χρήστη για την εύρεση γεγονότων χρησιμοποιώντας την τοποθεσία και την κατηγορία του γεγονότος που θα εισάγει ο χρήστης, όπως π.χ. «Αθήνα» και

«Παρουσίαση Βιβλίου». Μέσω ενός API όπως το Eventful API, μπορούν να ανακτηθούν events που περιέχουν τις ετικέτες (tags), τον τίτλο και την τοποθεσία που επιλέχθηκε. Με τα δεδομένα αυτά μπορούν να αντληθούν και σχετικές εικόνες, ειδήσεις και άλλο περιεχόμενο από το Flickr.com και το Digg.com με τα αντίστοιχα APIs.

2.5.2 Περίπτωση Χρήσης Enterprise 2.0

Στο πλαίσιο των επιχειρήσεων τα mashups προσφέρουν ένα εξαιρετικό πεδίο για τη δημιουργία εφαρμογών υποστήριξης των καθημερινών δραστηριοτήτων για συγκεκριμένες, επαναλαμβανόμενες εργασίες. Για παράδειγμα, στα ερευνητικά εργαστήρια της IBM, οι εργαζόμενοι ενημερώνουν τακτικά ημερολόγια με συζητήσεις διεθνών εργαστηρίων και δημόσιες ομιλίες που λαμβάνουν χώρα σε πανεπιστήμια και άλλους οργανισμούς. Επομένως, ένα mashup θα μπορούσε να συγκεντρώνει Atom feeds από διάφορες ομιλίες και να προσφέρει στους εργαζόμενους ένα κοινό τόπο ενημέρωσης των καταλόγων τους με μία σειρά από χαρακτηριστικά και μηχανισμούς διαχείρισης των ενημερώσεων. (25)

2.5.3 Mashups vs Portals

Τα portals είναι παλαιότερη τεχνολογία και επέκταση του κλασικού μοντέλου διακομιστή ιστοσελίδων (δυναμικό περιεχόμενο), στην οποία η διαδικασία μετατροπής των δεδομένων σε διαδικτυακό περιεχόμενο χωρίζεται σε δύο φάσεις: Τη δημιουργία των τμημάτων σήμανσης από τα portlets και τη συγκέντρωση τους σε μία ενιαία ιστοσελίδα (portal). Τα portlets μπορούν να φιλοξενούνται στον portal διακομιστή ή σε έναν απομακρυσμένο. Η συγκέντρωση του περιεχομένου πραγματοποιείται στο διακομιστή ενώ η mashup προσέγγιση επιτρέπει τη συγκέντρωση του περιεχομένου και στη μεριά του χρήστη. Τα portals αποτελούνται από τμήματα σήμανσης όπως HTML, WML, VoiceXML (presentation-oriented) κ.ά. Από την άλλη πλευρά, τα mashups βασίζονται σε «καθαρό» (plain) XML περιεχόμενο και HTML (presentation-oriented). Οι λειτουργίες read και update (συμβάντα ανάγνωσης και ενημέρωσης) σε ένα portal καθορίζονται με συγκεκριμένο API ενώ οι CRUD (Create, Read, Update, Delete) λειτουργίες των mashups βασίζονται στο REST και όχι σε επίσημο API.

2.5.4 Δομικά Χαρακτηριστικά Mashups

Τα mashups είναι απλά ένα νέο είδος διαδικτυακής εφαρμογής. Για αυτό το λόγο τα καλά δομημένα mashups υπόκεινται ισοδύναμα στο τρίπτυχο -μοντέλα δεδομένων, όψεις και ελεγκτές αλληλεπίδρασης, γνωστό ως MVC-της σχεδίασης μίας διαδικτυακής εφαρμογής. Η «μίξη» διαφορετικών Web APIs συνεπάγεται τις εξής λειτουργίες:

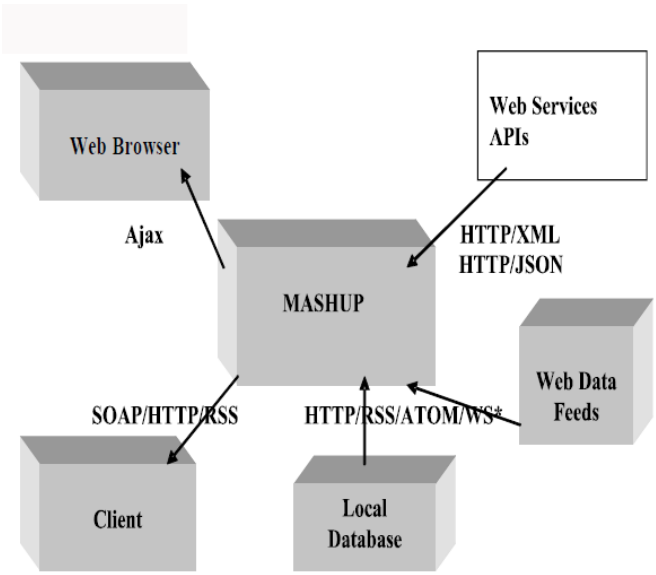
- *Μεσολάβηση δεδομένων*. Λαμβάνουν μέρος μετατροπές δεδομένων και συνδυασμός στοιχείων των δεδομένων για την εξυπηρέτηση της λογικής της mashup εφαρμογής. Έτσι μπορεί να δημιουργηθεί νέο στοιχείο δεδομένων ή απλά να κληθεί μία μέθοδος του API με τα κατάλληλα ορίσματα.
- *Μεσολάβηση διαδικασιών*. Κατευθύνει τη δημιουργία μίας νέας διαδικασίας από τα διαφορετικά APIs, και αναπαριστά τα σημεία αλληλεπίδρασης στο χρήστη.
- *Προσαρμογή της διεπαφής χρήστη*. Δέχεται τις εισαγόμενες από το χρήστη πληροφορίες και τον ενημερώνει σχετικά με τις διαδικασίες.

2.5.5 Αρχιτεκτονική Των Mashups

Το νέο αυτό είδος εφαρμογής ιστού, αποτελείται αρχιτεκτονικά, από τρεις φυσικά-διαχωρισμένους συμμετέχοντες: τους παρόχους των APIs, την ιστοσελίδα της mashup εφαρμογής και τον περιηγητή ιστού στην μεριά του χρήστη. Αναλυτικότερα: (26)

- *Πάροχοι APIs.* Η διάθεση των δεδομένων γίνεται με web πρωτόκολλα όπως REST, Web Services, και λίστες RSS/Atom. Καλούνται "mashup enablers" καθώς διαθέτουν τα δομικά υλικά για την ανάπτυξη μιας τέτοιας εφαρμογής. Το περιεχόμενο προέρχεται επίσης από μία τεχνική που καλείται «screen-scraping» και συλλέγει αξιοποιήσιμες πληροφορίες από τους παρόχους για το mashup.
- *Ιστοσελίδα εφαρμογής mashup.* Ο τόπος όπου φιλοξενείται η εφαρμογή. Σε αυτόν υπάρχει η λογική του mashup χωρίς να εκτελούνται απαραίτητα εδώ και οι αντίστοιχες διαδικασίες. Η υλοποίηση ενός mashup μπορεί να ακολουθεί το παράδειγμα των κλασικών εφαρμογών ιστού και την δυναμική παραγωγή περιεχομένου στο διακομιστή (server-side mashup). Εναλλακτικά, ο συνδυασμός του περιεχομένου μπορεί να γίνει απευθείας στον ιστότοπο του mashup (client-side mashup), στο browser του χρήστη, με JavaScript ή κάποιο applet. Με τον τρόπο αυτό αποφορτίζεται ο διακομιστής της εφαρμογής και βελτιώνεται η εμπειρία για το χρήστη επειδή το περιεχόμενο ανανεώνεται τμηματικά χωρίς την απαίτηση επαναφόρτωσης του ιστότοπου. Σύμφωνα με την παραπάνω προσέγγιση, ένα mashup αποτελεί μία Rich Internet Application. Συχνά χρησιμοποιείται λογική και από τις δύο πλευρές (διακομιστή-χρήστη) για τη συγκέντρωση των δεδομένων. Επιπλεόν, η εκτέλεση περίπλοκων ερωτημάτων σε δεδομένα από πολλαπλές πηγές δεν είναι εφικτή στον browser του χρήστη.
- *Πρόγραμμα περιήγησης ιστού.* Στον περιηγητή ιστού γίνεται η γραφική σύνθεση της εφαρμογής κι αποτελεί το μέσο αλληλεπίδρασης χρήστη-εφαρμογής. Τα περισσότερα προγράμματα περιήγησης μπορούν να εγκατασταθούν με επιτυχία σε διαφορετικά λειτουργικά συστήματα και να λειτουργούν ως εναλλακτικές μορφές περιήγησης στο Διαδίκτυο. Έτσι δεν απαιτείται η χρήση εξειδικευμένου προγράμματος για την υλοποίηση του mashup, ο χρήστης είναι εξοικειωμένος με τη χρήση του περιηγητή λόγω καθημερινής χρήσης και υπάρχει συμβατότητα με όλα τα λειτουργικά συστήματα. Όλα τα παραπάνω αποτελούν τους κύριους λόγους που χρησιμοποιείται μία τέτοια διεπαφή για τη φιλοξενία των mashup εφαρμογών.

Στην παρακάτω εικόνα (Εικόνα 12) βλέπουμε τα συστατικά που αναφέραμε παραπάνω: το πρόγραμμα περιήγησης ιστού, τους παρόχους (web services SPIs, web data feeds), ιστοσελίδα εφαρμογής (client, local database).



Εικόνα 12: Αρχιτεκτονική mashups

2.5.6 Screen-Scraping

Τα δημόσια APIs (public APIs) είναι κατάλληλα σχεδιασμένα κανάλια πρόσβασης σε δεδομένα και υπηρεσίες του Παγκόσμιου Ιστού. Σε κάποιες περιπτώσεις, θέλουμε να συμπεριλάβουμε περιεχόμενο, σε ένα mashup web application, για το οποίο δεν υπάρχει αντίστοιχο public API. Ακόμα κι αν υπάρχει, ενδέχεται να μην καλύπτει πλήρως τις απαιτήσεις της εφαρμογής. Άρα λοιπόν, είναι χρήσιμο να ψάχνουμε πέρα από το διαθέσιμο API και να καταφεύγουμε σε άλλες μεθόδους απόκτησης των δεδομένων.

Το screen-scraping είναι μία τεχνική απόσπασης δεδομένων από έναν πάροχο πληροφορίας ο οποίος δε διαθέτει τα κατάλληλα APIs για τη διανομή τους. Τα δεδομένα αποκτούνται από διεπαφές, σχεδιασμένες για ανθρώπους-χρήστες.

Ακριβώς επειδή η συγκεκριμένη πληροφορία δεν έχει σχεδιαστεί ώστε να είναι διαθέσιμη σε άλλες εφαρμογές, το screen-scraping μπορεί να θεωρηθεί ως ανεπιθύμητη ενέργεια εξαιτίας, για παράδειγμα, της επιβάρυνσης του συστήματος απ' όπου αντλείται η πληροφορία, ενδεχόμενη απώλεια εισοδήματος από διαφήμιση ή την απώλεια ελέγχου του περιεχομένου της πληροφορίας σχετικά με τη χρήση του. Από την πλευρά της mashup εφαρμογής απαιτούνται αρκετά πολύπλοκοι μηχανισμοί για την ανάλυση της ιστοσελίδας, την απομόνωση και τελικά, τη μεταφορά της πληροφορίας. Οι μηχανισμοί αυτοί διαφοροποιούνται, ανάλογα με τη δομή του περιεχομένου της εκάστοτε ιστοσελίδας. (27) Υπάρχουν πολλοί ταυτόσημοι όροι για τη συγκεκριμένη τεχνική, όπως data scraping, data extraction, web scraping, page scraping και HTML scraping.

Τα bots (web bots, internet bots, ή διαδικτυακά ρομπότ) και τα spiders (web crawlers ή προγράμματα ανίχνευσης του Web) είναι προγράμματα που τρέχουν αυτόματα μέσω του διαδικτύου και χρησιμοποιούν τεχνικές screen-scraping. Υπάρχουν πολλά είδη bots με διάφορους σκοπούς: Από bots που δημιουργούν και στέλνουν ανεπιθύμητα μηνύματα μέχρι bots που απαντούν με άμεσα μηνύματα, εξυπηρετώντας πελάτες. Τα web spiders «περιπλανώνται» στον παγκόσμιο ιστό συλλέγοντας τεράστιο όγκο ιστοσελίδων, οι οποίες χρησιμοποιούνται από τις μηχανές αναζήτησης για ευρετηρίαση (indexing).

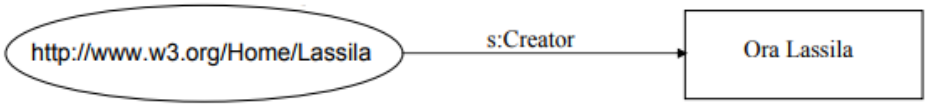
Σε γενικές γραμμές, το screen-scraping είναι μία άκομψη, ad-hoc τεχνική που συνήθως χρησιμοποιείται όταν δεν είναι διαθέσιμος κάποιος «επίσημος» μηχανισμός για πρόσβαση στα δεδομένα, όπως τα δημόσια APIs.

2.5.7 Σημασιολογικός Ιστός, RDF και Microformats

Μία από τις μεγαλύτερες προκλήσεις που αντιμετωπίζει η επιστημονική κοινότητα του Παγκόσμιου Ιστού είναι ο ορισμός πλαισίων και μεθόδων για την απόκτηση κοινού κώδικα επικοινωνίας ανθρώπου-υπολογιστή. Ο σημασιολογικός ιστός (semantic web) είναι μία επέκταση του Παγκόσμιου Ιστού που προσδίδει δομή στο περιεχόμενο που είναι κατανοητό στον άνθρωπο, ώστε να είναι «κατανοητό» και κατάλληλο για χειρισμό, και από τους υπολογιστές. Αποδίδεται στην πληροφορία σαφώς καθορισμένη σημασία που επιτρέπει τη συνεργασία ανάμεσα σε ανθρώπους και υπολογιστές. Έχει στόχο τη δημιουργία ενός περιβάλλοντος όπου πράκτορες λογισμικού (software agents), περιπλανώμενοι σε σελίδες, εκτελούν απαιτητικές εργασίες για τους χρήστες. (28) Για παράδειγμα, ένας τέτοιος πράκτορας λογισμικού που θα επισκεφτεί την ιστοσελίδα μίας ιατρικής κλινικής θα γνωρίζει, εκτός τις λέξεις-κλειδιά «θεραπεία, φαρμακευτική αγωγή», τα ωράρια των ιατρών, τον τρόπο κλεισίματος ραντεβού και άλλες χρήσιμες πληροφορίες.

Μία σημαντική τεχνολογία για την ανάπτυξη του σημασιολογικού ιστού είναι το RDF (Resource Description Framework). Βασίζεται στην XML και παρέχει μεθοδολογίες (26) για τη δημιουργία συντακτικών δομών που περιγράφουν τα δεδομένα ώστε να είναι δυνατή η κατανόησή τους από υπολογιστές (machine-understandable). Η XML επιτρέπει την αυθαίρετη δόμηση (όχι σαφώς καθορισμένη δομή) του εγγράφου δεδομένων, η οποία οδηγεί σε πολλές πιθανές κωδικοποιήσεις των ίδιων δεδομένων, αλλά δεν περιγράφει την εννοιολογία της δομής. Η εννοιολογία εκφράζεται από το RDF, το οποίο την κωδικοποιεί σε τριάδες. Στην αναπαράσταση με τριάδες ορίζονται τα συστατικά μέρη subject-predicate-object (υποκείμενο-κατηγόρημα-αντικείμενο). Οι τριάδες μπορούν να γραφούν με τη χρήση XML tags και εναλλακτικά μπορούν να μοντελοποιηθούν σε ένα κατευθυνόμενο γράφο. Σε ένα RDF έγγραφο εκφράζονται ισχυρισμοί ότι συγκεκριμένες οντότητες (άνθρωποι, ιστοσελίδες κ.ά.) έχουν ιδιότητες με συγκεκριμένες τιμές (άνθρωποι, έγγραφα, ιστοσελίδες, ημερομηνίες) στη τριαδική μορφή που εξηγήσαμε παραπάνω. Το υποκείμενο και το αντικείμενο μπορούν να οριστούν μέσα στο έγγραφο ή να αναφέρεται μέσω ενός URI σε έναν άλλο πόρο στον Παγκόσμιο Ιστό. Το κατηγόρημα μπορεί να είναι οποιοδήποτε όνομα από το χώρο των ειδικών ονομάτων XML (XML qualified).

Για παράδειγμα, (29)(Εικόνα 13) στην δήλωση "Ora Lassila is the creator of the resource http://www.w3.org/Home/Lassila" έχουμε το resource (subject) "http://www.w3.org/Home/Lassila" το property (predicate) "http://www.schema.org/#Creator" και το value (object) "Ora Lassila".



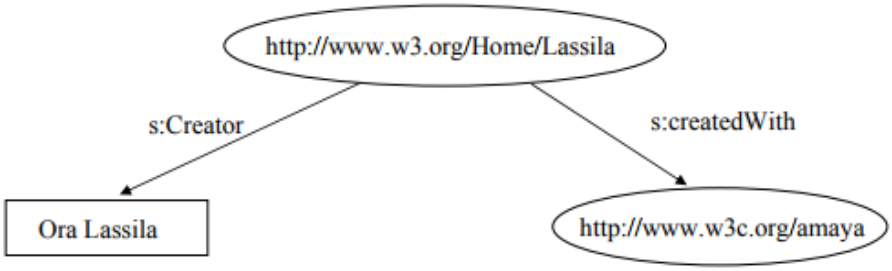
Εικόνα 13: Κατευθυνόμενος γράφος των συστατικών

Το RDF καθορίζει μία σύνταξη που καλείται RDF/XML (Εικόνα 14) για την αναπαράσταση RDF πληροφοριών και τη μετάδοσή τους ανάμεσα σε υπολογιστές. Τα XML tags εκφράζουν την εννοιολογία του εκάστοτε περιεχομένου, στα προγράμματα υπολογιστών.

```
<?xml version="1.0"?>
<rdf:RDF>
  <rdf:Description about="http://www.w3.org/Home/Lassila">
```

```
      <s:Creator>Ora Lassila</s:Creator>
      <s:createdWith rdf:resource="http://www.w3c.org/amaya"/>
   </rdf:Description>
</rdf:RDF>
```

Εικόνα 14: Παράδειγμα RDF/XML κωδικοποίησης.

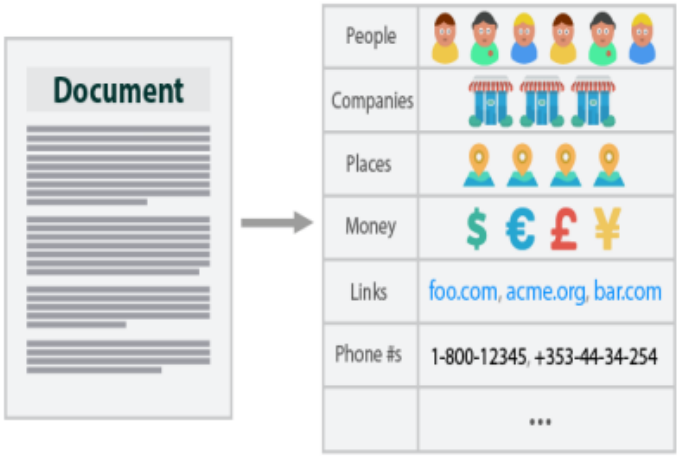


Εικόνα 15: Ο γράφος που προκύπτει από την RDF/XML αναπαράσταση (Εικόνα 14)

Ένα σύνολο από RDF ισχυρισμούς χρησιμοποιεί συγκεκριμένους όρους για τον καθορισμό των ιδιοτήτων (κατηγορημάτων) και των τύπων δεδομένων που έχουν σημασία για τις εφαρμογές. Ένα τέτοιο RDF «λεξιλόγιο» μπορεί να οριστεί το RDF-Schema. Το RDF-S χρησιμοποιεί ιεραρχία κλάσεων/υποκλάσεων για τους πόρους (υποκείμενα). Η σχέση ανάμεσα σε ένα στιγμιότυπο της κλάσης και την κλάση καθορίζεται με την ιδιότητα **type**. Type περιορισμοί (περιορισμοί σχέσης) στα υποκείμενα και τα αντικείμενα μπορούν να καθοριστούν με τα **domain** και **range**. Ένα παράδειγμα (30) περιορισμού **domain** είναι ότι σε έναν ισχυρισμό, τα υποκείμενα και τα αντικείμενα που έχουν την ιδιότητα http://www.example.org/friendOf σε μία τριάδα πρέπει να είναι της κλάσης http://www.example.org/Person.

Τα RDF λεξιλόγια υιοθετούνται από ένα μεγάλο εύρος εφαρμογών συμπεριλαμβανομένων των εφαρμογών κοινωνικής δικτύωσης (“Friend of friend” ή FOAF) και της αναμετάδοσης περιεχομένου (RSS).

Τα Microformats είναι μικρά, δομημένα κομμάτια δεδομένων που είναι ενσωματωμένα στις ιστοσελίδες (HTML/XHTML) και σε άλλα έγγραφα, όπως XML και Atom/RSS feeds. Η HTML/XHTML έχει σχεδιαστεί κυρίως για να παρέχει στο χρήστη διεπαφές αλληλεπίδρασης με το περιεχόμενο μέσω ενός περιηγητή ιστού. Ωστόσο, ακολουθώντας συγκεκριμένες ιδιότητες σήμανσης και χρησιμοποιώντας πρότυπα περιγραφικά ονόματα κλάσεων στις HTML/XHTML ετικέτες, μπορούμε να δημιουργήσουμε καθαρά εξαγόμενα δεδομένα. Έτσι, μπορούμε σχετικά εύκολα, να αναπτύξουμε προγράμματα που θα εξάγουν δεδομένα (όπως πληροφορίες επικοινωνίας, γεωγραφικές συντεταγμένες, διευθύνσεις, ημερολογιακά συμβάντα, κοινωνικές σχέσεις κ.ά.) από τα microformats και θα τα επαναχρησιμοποιούν mashups, μηχανές αναζήτησης και άλλες εφαρμογές. (Εικόνα 16) Τα microformats πρόσθεσαν στις ετικέτες σήμανσης του Παγκόσμιου Ιστού σημασιολογία (περιγραφή) της πληροφορίας που περιέχουν. Έτσι, αποφεύγονται άκομψες και περίπλοκες μέθοδοι όπως το screen-scraping και η επεξεργασία φυσικής γλώσσας. Η χρήση, η υιοθέτηση και η επεξεργασία των microformats επιτρέπει την ευρετηρίαση, την αναζήτηση, την αποθήκευση ή τη διασταύρωση των στοιχείων δεδομένων, ώστε οι πληροφορίες να μπορούν να επαναχρησιμοποιηθούν ή να συνδυαστούν. (31)



Εικόνα 16: Δεδομένα που εξάγονται από microformats

Στην παραπάνω εικόνα (Εικόνα 16) δίνεται μία οπτική επεξήγηση της χρησιμότητας των Microformats. Ένα πρόγραμμα υπολογιστή μπορεί να αναλύσει το έγγραφο HTML/XHTML, και με τη χρήση των microformats να εξάγει χρήσιμα δεδομένα που αφορούν σε ανθρώπους, οργανισμούς, τοποθεσίες, προϊόντα κ.ά., σε πάρα πολύ μικρό χρόνο.

2.5.7.1 Τύποι Microformats

Έχουν αναπτυχθεί αρκετά microformats για τη σημασιολογική σήμανση συγκεκριμένων ειδών πληροφορίας. Τα μόνα επιβεβαιωμένα είναι τα hCard και hCalendar:

- hCard – Αντιπροσωπεύει ένα φυσικό πρόσωπο ή έναν οργανισμό και συγκεκριμένα πληροφορίες επικοινωνίας. Περιλαμβάνει
  - adr – Για διευθύνσεις
  - geo – Για γεωγραφικές συντεταγμένες (Εικόνα 17)
- hCalendar – Αντιπροσωπεύει πληροφορίες της μορφής iCalendar για ένα γεγονός
- hAtom – Αντιπροσωπεύει Atom feeds μέσα στην HTML
- hNews – Αντιπροσωπεύει ειδησεογραφικό περιεχόμενο
- hProduct – Αντιπροσωπεύει προϊόντα
- hRecipes – Αντιπροσωπεύει συνταγές και φαγητό
- hResume – Αντιπροσωπεύει βιογραφικά σημειώματα
- rel-tag – Συσχετίζει μία HTML ετικέτα (<a>) με ένα URL. (Εικόνα 17)
- rel-license – Συσχετίζει μία HTML ετικέτα (<a>) με το URL της άδειας που χρησιμοποιεί η ιστοσελίδα
- xoxo – Αντιπροσωπεύει ιεραρχική διάρθρωση (δηλαδή εμφωλευμένες λίστες, όπως παραπάνω στο hCard)

```
<a href="/tag/mashup/" rel="tag" class="category">mashup</a>

The birds roosted at
   <span class="geo">
     <span class="latitude">52.48</span>,
```

```
    <span class="longitude">-1.89</span>
  </span>
```

Εικόνα 17: Παραδείγματα microformats

2.5.8 RSS και ATOM

Οι RSS feeds (ροές) είναι μία οικογένεια από τύπους δεδομένων βασισμένες στην XML. Η RSS χρησιμοποιείται για την μετάδοση περιεχομένου (ενημερώσεις) όπως άρθρα ειδήσεων, δημοσιεύσεις logs κ.ά. Οι εν λόγω ενημερώσεις περιέχουν τα πλήρη δεδομένα, σύνοψη των δεδομένων, σχετικά μεταδεδομένα, ημερομηνία έκδοσης κλπ, ενώ αποστέλλονται αυτομάτως στον συνδρομητή μέσω Διαδικτύου. Τέτοια προγράμματα που έχουν κατασκευαστεί ώστε να μπορούν να διαβάζουν τις ροές δεδομένων ονομάζονται αναγνώστες rss ή aggregators. Οι προγραμματιστές μπορούν να έχουν πρόσβαση στις ροές δεδομένων μέσω των μοναδικών URI τους, αφού ουσιαστικά πρόκειται για έγγραφα XML. Σε αντίθεση με το RSS 1.0, το RSS 2.0 (Εικόνα 18) δε χρησιμοποιεί την προδιαγραφή RDF.

Το ATOM είναι ένα νεότερο αλλά όμοιο πρωτόκολλο μετάδοσης ενημερώσεων βασισμένο στην XML και αποτελεί προτεινόμενο πρότυπο του IETF. Δημιουργήθηκε για να παρέχει περισσότερη πληροφορία από το RSS, έχει παρόμοια σημασιολογία αλλά χρησιμοποιεί διαφορετικές ονομασίες (Εικόνα 19).

```
<?xml version="1.0" encoding="UTF-8"?>
<rss version="2.0">
   <channel>
      <title>Apress</title>
      <link>http://www.apress.com/</link>
      <description>Welcome to Apress.com.</description>
      <item>
         <title>Excel 2007: Beyond the Manual</title>

<link>http://www.apress.com/book/bookDisplay.html?bID=10232</link>
         <description>...</description>
      </item>
      <item>
         <title>Word 2007: Beyond the Manual</title>

<link>http://www.apress.com/book/bookDisplay.html?bID=10249</link>
         <description>...</description>
      </item>
   </channel>
</rss>
```

Εικόνα 18: RSS 2.0 έγγραφο ηλεκτρονικού βιβλιοπωλείου.

```
<?xml version="1.0" encoding="UTF-8"?>
<feed xmlns="http://www.w3.org/2005/Atom">
   <title>Apress</title>
   <subtitle>Welcome to Apress.com.</subtitle>
   <link           rel="alternate"           type="text/html"
href="http://www.apress.com/" />
   <link                                          rel="self"
href="http://examples.mashupguide.net/ch04/Atom1.0_Apress.xml"
```

```
/>
   <updated>2007-07-25T12:57:02Z</updated>
   <author>
      <name>Apress, Inc.</name>
      <email>support@apress.com</email>
   </author>
   <id>http://apress.com/</id>
   <entry>
      <title>Excel 2007: Beyond the Manual</title>
      <link
href="http://www.apress.com/book/bookDisplay.html?bID=10232"
/>

<id>http://www.apress.com/book/bookDisplay.html?bID=10232</id>
      <updated>2007-07-25T12:57:02Z</updated>
      <summary type="html">...</summary>
   </entry>
   <entry>
      <title>Word 2007: Beyond the Manual</title>
      <link
href="http://www.apress.com/book/bookDisplay.html?bID=10249"
/>

<id>http://www.apress.com/book/bookDisplay.html?bID=10249</id>
      <updated>2007-07-25T12:57:10Z</updated>
      <summary type="html">...</summary>
   </entry>
</feed>
```

Εικόνα 19: Atom 1.0 έγγραφο ηλεκτρονικού βιβλιοπωλείου

2.6 Tagging και Folksonomies

Η οργάνωση του ψηφιακού περιεχομένου αποτελεί μία μεγάλη πρόκληση της κοινότητας του Παγκόσμιου Ιστού. Οι ετικέτες (tags) είναι ένας από τους πιο δημοφιλείς μηχανισμούς που χρησιμοποιούνται σήμερα, για την οργάνωσή και κατηγοριοποίηση του περιεχομένου από συμμετέχοντες χρήστες σε ένα online διαμοιραζόμενο περιβάλλον. Μία ετικέτα είναι ένα κείμενο (λέξη ή φράση) που μπορεί να προστεθεί σε ένα τμήμα περιεχομένου και να συνεισφέρει στην περιγραφή του.

Η εκχώρηση ετικετών σε διαδικτυακά αντικείμενα (tagging) είναι μία μέθοδος ευρετηρίασης (indexing) και κατηγοριοποίησης των πληροφοριών. Εφαρμόζεται σε μία ποικιλία ψηφιακών πόρων όπως φωτογραφίες, URLs, μουσικά κομμάτια κ.ά. Οι ετικέτες και οι πόροι εκτίθενται στους συμμετέχοντες χρήστες και αυτό δημιουργεί κοινωνικό κίνητρο στο tagging. Οι χρήστες εκχωρούν ετικέτες σε διαδικτυακά αντικείμενα στοχεύοντας στην κοινοποίηση, την ανακάλυψη και την ανάκτηση αυτών. Η ανακάλυψη χαρίζει τη δυνατότητα εύρεσης νέου περιεχομένου το οποίο έχει κοινοποιηθεί από άλλους χρήστες και μας ενδιαφέρει. Η ανάκτηση μας επιτρέπει να ανακαλούμε τα διαδικτυακά αντικείμενα που έχουν ανακαλυφθεί νωρίτερα. Με την προσθήκη μίας ετικέτας σε ένα άρθρο ή ένα βίντεο που μας ενδιαφέρει βοηθάμε άλλους χρήστες με παρόμοια ενδιαφέροντα να το βρουν. Συχνά συναντάμε τη διαδικασία του tagging (επισήμανση με ετικέτα) ως social/collaborative (κοινωνικό/συνεργατικό) tagging η οποία επικεντρώνεται σε κοινωνικά δίκτυα και εφαρμογές.

Το σύνολο των ετικετών που εκχωρείται από τους συμμετέχοντες χρήστες σχηματίζει μία "folksonomy". Αποτελεί ένα «συλλογικό λεξιλόγιο» που προκύπτει από τη διαδικασία του tagging. Folksonomy είναι ένα σύστημα κατηγοριοποίησης που βασίζεται στην πρακτική συλλογικής δημιουργίας ετικετών και σκοπεύει να καταστήσει την πληροφορία απλή στην πρόσβαση και την αναζήτηση. Οι συλλογές folksonomies αυτές έγιναν γνωστές στον Παγκόσμιο Ιστό με εφαρμογές «κοινωνικού» χαρακτήρα όπως συλλογικοί σελιδοδείκτες (social bookmarking applications), διαμοιρασμός φωτογραφιών (photo sharing applications) και προσωπικά ιστολόγια (weblogs).

Οι υποστηρικτές των συστημάτων tagging προτείνουν την κοινωνικότητα του tagging για ένα είδος μηχανισμού ανατροφοδότησης. Αν ένας χρήστης ακολουθήσει τους «κανόνες» των ετικετών που έχουν προστεθεί σε ένα πόρο από άλλους συμμετέχοντες, αυτοί είναι πιο πιθανό να υιοθετήσουν παρόμοιο λεξιλόγιο για την περιγραφή σχετικών πόρων. (32) Έτσι η folksonomy μετατρέπεται σε λεξιλόγιο που χρησιμοποιούν συμμετέχοντες με κοινά ενδιαφέροντα.

Οι folksonomies προσφέρουν ευελιξία, συνεργατικότητα και ελευθερία. Οι χρήστες μπορούν να χρησιμοποιούν οποιοδήποτε όρο για την πρόσθεση μια ετικέτας χωρίς να είναι απαραίτητο να γνωρίζουν ακριβώς τη σημασία του.

Παρόλα αυτά, η απλότητα χρήσης του tagging και η έλλειψη συγκεκριμένης ομοιόμορφης δομής και εννοιολογικής αναπαράστασης στα συστήματα δημιουργεί προβλήματα τα οποία χωρίζονται σε δύο κατηγορίες: (33)

- *Τοπικές διαφοροποιήσεις.* Οι ετικέτες έχουν μικρή σημασιολογία και πολλές εκδοχές. Ακόμα κι αν μεμονωμένες ετικέτες μπορούν να θεωρηθούν ως αποτέλεσμα μελέτης του χρήστη, το σύνολο αυτών δεν αντιπροσωπεύει πάντα με ακρίβεια τη διανοητική ικανότητά του.
- *Κατανεμημένες διαφοροποιήσεις.* Τα περισσότερα συστήματα tagging έχουν δικούς τους συγκεκριμένους τρόπους για την επεξεργασία και ερμηνεία των ετικετών. Επομένως, αν θέλουμε να συγκεντρώσουμε δεδομένα tagging από διαφορετικές πηγές, είναι πολύ δύσκολο να μάθουμε τις έννοιες και τις συσχετίσεις μεταξύ των ετικετών.

Τα συστήματα tagging επιλέγουν τα μεταδεδομένα που προέρχονται από τους χρήστες των συστημάτων. Οι χρήστες είναι αυτοί που προσθέτουν ετικέτες στους πόρους και όχι οι διαχειριστές συστημάτων και οι δημιουργοί περιεχομένου. Αυτό συμβαίνει, διότι υπάρχει η αντίληψη ότι οι δημιουργοί του περιεχομένου δεν έχουν κίνητρα για την παροχή ποιοτικών μεταδεδομένων εκτός αν πρόκειται για την αυτοπροβολή τους. Επομένως τα μεταδεδομένα που θα προκύπταν θα ήταν αναξιόπιστα. Οι χρήστες δημιουργούν ετικέτες για την εξυπηρέτηση των ατομικών τους αναγκών και με αυτόν τον τρόπο παράγεται ένα συναφές λεξιλόγιο. (32)

2.6.1 Κριτήρια αποδοτικού tagging

Η επισήμανση με «καλές» ετικέτες βοηθάει στην ανακάλυψη και ανάκτηση του περιεχομένου. Ένα αξιόλογο σύνολο ετικετών πρέπει να διαθέτει τις παρακάτω ιδιότητες:

- *Υψηλή κάλυψη πολλαπλών όψεων.* Ένας καλός συνδυασμός ετικετών πρέπει να περιλαμβάνει πολλαπλές όψεις των αντικειμένων. Για παράδειγμα, οι ετικέτες ενός URL ταξιδιωτικής ιστοσελίδας μπορούν να περιλαμβάνουν γενικές ετικέτες όπως κατηγορία (travel), τοποθεσία (Crete), ημερομηνία (2016), συγκεκριμένη (Knossos Palace) και υποκειμενική (interesting) ετικέτα. Όσο πιο πολλές πτυχές καλύπτονται, τόσο πιο πιθανό είναι ένας χρήστης να ανακτήσει το περιεχόμενο.

- *Υψηλή δημοτικότητα.* Αν ένα σύνολο ετικετών χρησιμοποιείται από πολλούς χρήστες για ένα συγκεκριμένο αντικείμενο, τότε είναι λιγότερο πιθανό να πρόκειται για spam. Είναι πιο πιθανό να προσδιορίζουν μοναδικά το περιεχόμενο που χαρακτηρίζουν και ακόμη πιο πιθανό να χρησιμοποιηθούν από ένα νέο χρήστη για το συγκεκριμένο αντικείμενο.
- *Ελάχιστη προσπάθεια.* Το πλήθος των ετικετών που χαρακτηρίζουν ένα αντικείμενο πρέπει να είναι το ελάχιστο δυνατό και ο αριθμός των αντικειμένων που αντιστοιχούν στο συνδυασμό (σύνολο) των ετικετών πρέπει να είναι μικρός. Ως αποτέλεσμα, ένας χρήστης μπορεί να βρίσκει αντικείμενα μέσω σύντομης περιήγησης στις ετικέτες τους.
- *Ομοιογένεια.* Δεδομένου ότι δεν υπάρχει καθολική οντολογία, οι ετικέτες μπορεί να αποκλίνουν δραματικά. Διαφορετικοί άνθρωποι μπορούν να χρησιμοποιήσουν διαφορετικούς όρους για την ίδια έννοια. Γενικά παρατηρούνται δύο τύποι αποκλίσεων: αυτές που οφείλονται σε συντακτικές διακυμάνσεις, π.χ. blog, blogging και bog, και εκείνες που οφείλονται σε συνωνυμίες, π.χ. cell-phone, mobile phone, δηλαδή διαφορετικοί συντακτικά, όροι, που αναφέρονται στην ίδια έννοια. Αν και οι αποκλίσεις φέρνουν ανισορροπία στα συστήματα tagging, μπορούν να αυξήσουν την δυνατότητα ανάκλησης αντικειμένων. Αυτό που πρέπει να γίνει είναι να επιτρέπεται στο χρήστη να χρησιμοποιεί οποιαδήποτε συντακτική μορφή αλλά να συμπτύσσονται οι παραλλαγές σε μία αναγνωρισμένη από το σύστημα αναπαράσταση.
- *Αποκλεισμός ορισμένων τύπων ετικετών.* Οι προσωπικές οργανωτικές ετικέτες είναι λιγότερο πιθανό να κοινοποιούνται από τους χρήστες. Επομένως πρέπει να αποκλείονται από τη δημόσια χρήση.

2.6.2 Tag Clouds

Η οπτικοποίηση των folksonomies γίνεται με τη χρήση «σύννεφων ετικετών». Τα σύννεφα ετικετών αποτελούν στοιχεία διεπαφής τα οποία απεικονίζουν τα δεδομένα μιας folksonomy. Στο σύννεφο εμφανίζονται οι ετικέτες του λεξιλογίου ως απλές λέξεις και συνήθως κατηγοριοποιούνται αλφαβητικά. Η στατιστική ανάλυση των ετικετών που περιέχει ένας δικτυακός τόπος περιγράφεται με αλλαγές στο χρώμα ή στη γραμματοσειρά λέξης. Με αυτόν τον τρόπο είναι δυνατή η εύρεση μιας ετικέτας είτε αλφαβητικά είτε μέσω της δημοτικότητάς της. Οι ετικέτες είναι συνήθως υπερσύνδεσμοι που οδηγούν σε συλλογές θεμάτων σχετικών με την ετικέτα. Επιλέγοντας μία ετικέτα εμφανίζεται το περιεχόμενο το οποίο έχει επισημανθεί με αυτή.

Τα tag clouds έχουν χαρακτηριστεί έξυπνα (Jeffrey Zeldman) αλλά επίσης έχουν επικριθεί πολλές φορές. Οι περισσότερες αρνητικές κριτικές αφορούσαν όχι τόσο στα tag clouds αλλά τις folksonomies γενικότερα. Αρκετοί έσπευσαν να υπερασπιστούν την προσέγγιση των σύννεφων (N. Jon , Tag Clouds: a Response, 2005) χαρακτηρίζοντας τα ως βοηθήματα πλοήγησης. Ο Zeldman ισχυρίστηκε ότι η προσαρμογή των ετικετών ανάλογα με τη δημοφιλία τους εμποδίζει την πρόσβαση σε χρήσιμα αντικείμενα που δεν έχουν επισημανθεί με δημοφιλείς ετικέτες. Ο Tomas Vander Wal ο άνθρωπος που επινόησε τον όρο "folksonomy" περιγράφει τα σύννεφα ετικετών ως «συμπαθητικά αλλά προσφέρουν λίγη αξία». Οι επικριτές της ιδέας υποστηρίζουν ακόμα ότι η ελευθερία που προσδίδεται στο χρήστη μπορεί να οδηγήσει στην εισαγωγή αναξιόπιστου όρου που θα έχει ασυνεπή αποτελέσματα. Επιπλέον, μπορεί να χρησιμοποιηθούν διαφορετικοί όροι για την περιγραφή ενός αντικειμένου με πιθανό αποτέλεσμα να ανακύπτουν εννοιολογικές συνωνυμίες και πολυσημίες. Η αυθαίρετη και κατά βούληση του χρήστη προσθήκη ετικετών μπορεί να οδηγήσει σε αναποτελεσματικές προσπάθειες κατηγοριοποίησης και αναζήτησης πληροφοριών.

Στα πλαίσια μίας έρευνας (32) για το «αν τα tag clouds παρέχουν αξία στους χρήστες που αναζητούν πληροφορίες σε ένα folksonomy σύνολο δεδομένων», οι συμμετέχοντες έχοντας ως επιλογές μέσων

αναζήτησης ένα σύννεφο ετικετών και ένα πλαίσιο αναζήτησης, απάντησαν περισσότερες ερωτήσεις με τη χρήση του σύννεφου. Αυτό δείχνει ότι τα σύννεφα ετικετών ωφελούν την πλοήγηση σε μία folksonomy συλλογή δεδομένων.



Εικόνα 20: Tag Cloud με όρους σχετικούς προς το Web 2.0

2.6.3 Σημασιολογικές προσεγγίσεις των folksonomies

Έχουν γίνει σημαντικές απόπειρες για την προσθήκη δομής και σημασιολογίας στα συστήματα tagging. Οι προσεγγίσεις στο tagging εστιάζουν στην στατιστική ανάλυση των προτύπων χρήσης ετικετών (Golder & Herberman, 2006), την ανάκτηση πληροφοριών και πλοήγηση (Halpin et al., 2006; Jäschke, 2008), και στην ανάλυση και ομαδοποίηση κοινωνικών δικτύων με βάση τα δεδομένα tagging (Mika, 2005; Brooks et al., 2006). Ο Golder και ο Herbermann συνέλεξαν δεδομένα από το del.icio.us (υπηρεσία συλλογικών σελιδοδεικτών) και ανέλυσαν τη δομή και τα πρότυπα χρήσης των συστημάτων tagging. Η εργασία τους πραγματεύεται τη διάκριση ανάμεσα στο συνεργατικό tagging και στα taxonomies. Παρά το γεγονός ότι το συνεργατικό tagging έχει περιορισμούς στη δομή και τη σημασιολογία του, παρέχει δυνατότητα μάθησης με το διαμοιρασμό και την οργάνωση της πληροφορίας. Ο Marlow (2006) παρατήρησε ότι με την πάροδο του χρόνου, ο αριθμός των ετικετών για συγκεκριμένους χρήστες, μπορεί να σταθεροποιηθεί, ενώ για άλλους αυξάνεται. Σύμφωνα με μελέτη που συμμετείχε ο Cattuto (2007) οι folksonomies παρουσιάζουν δομή μικρόκοσμου (small-world) όπως τα κοινωνικά δίκτυα. Έχουν εισαχθεί έννοιες ομαδοποίησης (clustering) και μέσης συντομότερης διαδρομής ανάμεσα σε δύο κόμβους (characteristic path) για την περιγραφή των συσχετίσεων δύο άγνωστων κόμβων στον μικρόκοσμο των ετικετών. Μία ακόμα μελέτη (Mika, 2005) περιγράφει τα social tagging συστήματα ως τριμερείς γράφους συμπεριλαμβάνοντας τους χρήστες, τις ετικέτες και τους πόρους. Δημιουργείται ένα δίκτυο συνεμφανίσεων (co-occurrence network) οντολογιών που δημιουργούνται αυτόματα από τις ετικέτες. Η πρόταση επέκτασης του τριμερούς μοντέλου έγινε από τον Gruber ο οποίος πρόσθεσε το χώρο επισήμανσης (π.χ. social bookmarking site) και την παράμετρο θετικό/αρνητικό η οποία για την αναπαράσταση ετικετών από spammers. (33)

Researchers	**Suggested model of tagging activities**
Mika, 2005; Halpin, 2006; Cattuto, 2007	Tagging: (*U*, *T*, *R*)
Gruber (2005)	Tagging: (*object*, *tag*, *tagger*, *source*, + or -)

Οι folksonomies δεν δημιουργούνται ξεχωριστά, αλλά από τους συμμετέχοντες χρήστες κατά τη διαδικασία του συνεργατικού tagging. Επομένως το μοντέλο αναπαράστασης των folksonomies πρέπει να θεωρηθεί ως μία συλλογή στιγμιότυπων του tagging model. Ένα μοντέλο αναπαράστασης folksonomy, εκτός από τα αντικείμενα που σχετίζονται με τις διαδικασίες tagging πρέπει να καλύπτει όλες τις πτυχές της συνεργασίας και τις σχέσεις. Επιπλέον, οι folksonomies αποτελούν μία πρακτική απόκτησης γνώσης από τις συνεργατικές διαδικασίες χαρακτηρισμού των πόρων. Άρα λοιπόν, το folksonomy model πρέπει να περιλαμβάνει και τις διαδικασίες tagging, όπως παρουσιάζεται παρακάτω:

Folksonomy: (*tag set, user group, source, occurrence, Tagging*†)

Όπου η τελευταία παράμετρος αποδίδει το σύνολο των διαδικασιών tagging που διεξήχθησαν από τους χρήστες και όπου κάθε διαδικασία αναπαρίσταται από το μοντέλο:

Tagging: (*object, tag, tagger*)

Όπου οι παράμετροι έχουν την ίδια σημασιολογία με το μοντέλο που προτάθηκε από τον Gruber (παραπάνω).

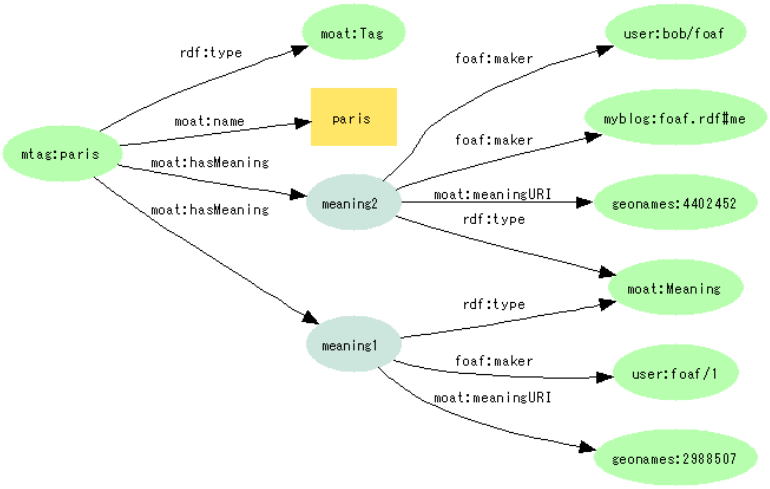
Υπάρχουν αρκετές συζητήσεις σχετικά με τα πλεονεκτήματα των folksonomies σε σύγκριση με τις οντολογίες και άλλα δομημένα λεξιλόγια και συστήματα κατηγοριοποίησης. Παρά τις διαφορές που υπάρχουν οι τεχνολογίες Σημασιολογικού ιστού μπορούν να θεωρηθούν ως συμπλήρωμα στις folksonomies (Shirky, 2005; Hendler, 2007). Όπως αναφέρθηκε ήδη, οι ετικέτες περιέχουν «ελεύθερο» κείμενο και επομένως δεν έχουν σαφείς σημασίες και είναι επιρρεπείς σε συνωνυμίες, πολυσημίες και ανθρώπινα σφάλματα. Ενώ οι χρήστες μπορούν να ερμηνεύσουν τη σημασιολογία μίας ετικέτας μέσω της ανάγνωσης ή της χρήσης της, οι υπολογιστές δεν μπορούν να την «κατανοήσουν», δεδομένου ότι δεν ορίζεται σε μορφή αναγνώσιμη από υπολογιστή (Passant,2008). Τα συστήματα folksonomies δεν παρέχουν ομοιόμορφο τρόπο διαμοιρασμού, ανταλλαγής και επαναχρησιμοποίησης των δεδομένων tagging ανάμεσα σε χρήστες ή κοινότητες (Kim et al., 2007). Οι Specia και Motta (2007) πρότειναν ως λύση την ενιαιοποίηση folksonomies και οντολογιών για τον εμπλουτισμό της εννοιολογίας των ετικετών. Σε γενικές γραμμές οι οντολογίες επισημάνσεων μπορούν να συμβάλλουν τους ακόλουθους τομείς: (33)

- *Προηγμένη Αναπαράσταση Γνώσης.* Μία οντολογία επισήμανσης μπορεί να αναπαραστήσει σταθερά τις οντότητες και τις συσχετίσεις από τις οποίες αποτελούνται οι επισημάνσεις. Η σαφής δομή γνώσης των δεδομένων επισήμανσης προετοιμάζει το έδαφος για τα διασυνδεδεμένα δεδομένα (Linked Data) των δεδομένων επισήμανσης στον Παγκόσμιο Ιστό.
- *Υποστήριξη Ανταλλαγής Γνώσης.* Οι οντολογίες παρέχουν επαναχρησιμοποιήσιμες δομές που επιτρέπουν την ανταλλαγή γνώσης ανάμεσα σε διαφορετικούς χρήστες και εφαρμογές. Μία οντολογία επισήμανσης μπορεί να χρησιμοποιηθεί για επισημάνσεις σε διαφορετικές πλατφόρμες.
- *Επεξεργασία Από Υπολογιστή.* Οι οντολογίες όπως και οι τεχνολογίες σημασιολογικού ιστού γενικότερα, μετασχηματίζουν την ανθρώπινη γνώση σε «κατανοητή» για τους υπολογιστές μορφή ώστε να πραγματοποιείται αυτόματα διασύνδεση δεδομένων από διαφορετικές πηγές και ενοποίηση των δεδομένων επισήμανσης.

Παρακάτω βλέπουμε (Πίνακας) ένα παράδειγμα περιγραφής ετικετών οντολογίας (ετικετών) MOAT η οποία στηρίζεται στη μορφή OWL. Στην Εικόνα 21 αναπαρίσταται το rdf schema της οντολογίας.

```
<moat:Tag rdf:about="http://tags.moat-project.org/tag/paris">
 <moat:name><![CDATA[paris]]></moat:name>
 <moat:hasMeaning>
```

```
  <moat:Meaning>
   <moat:meaningURI rdf:resource="http://sws.geonames.org/2988507/"/>
   <foaf:maker rdf:resource="http://example.org/user/foaf/1"/>
  </moat:Meaning>
 </moat:hasMeaning>
 <moat:hasMeaning>
  <moat:Meaning>
   <moat:meaningURI rdf:resource="http://sws.geonames.org/4402452/"/>
   <foaf:maker rdf:resource="http://example.org/user/bob/foaf"/>
   <foaf:maker rdf:resource="http://somwhere.net/myblog/foaf.rdf#me"/>
  </moat:Meaning>
 </moat:hasMeaning>
</moat:Tag>
```



Εικόνα 21: Οντολογία επισήμανσης MOAT

2.7 Πλατφόρμες συνεργασίας

Το groupware/collaborative software (ή πλατφόρμες συνεργασίας/συνεργατικό λογισμικό) είναι μία κατηγορία προγραμμάτων ηλεκτρονικών υπολογιστών που επιτρέπει στους χρήστες τους να συνεργάζονται σε εργασίες με κοινό στόχο, βρισκόμενοι σε γεωγραφικά διασκορπισμένες τοποθεσίες, μέσω κοινών διεπαφών στο Διαδίκτυο, ως μέσο επικοινωνίας εντός της ομάδας. Επίσης μπορεί να περιλαμβάνει συστήματα απομακρυσμένης πρόσβασης για την αρχειοθέτηση αρχειοθέτηση αρχείων δεδομένων που χρησιμοποιούνται συχνά. Τα μέλη της ομάδας εργασίας μπορούν να έχουν πρόσβαση στα αρχεία αυτά, να τα τροποποιήσουν και να τα ανακτήσουν.

Τα συστήματα groupware ταξινομούνται με βάση τις λειτουργίες τους, συγκεκριμένα:

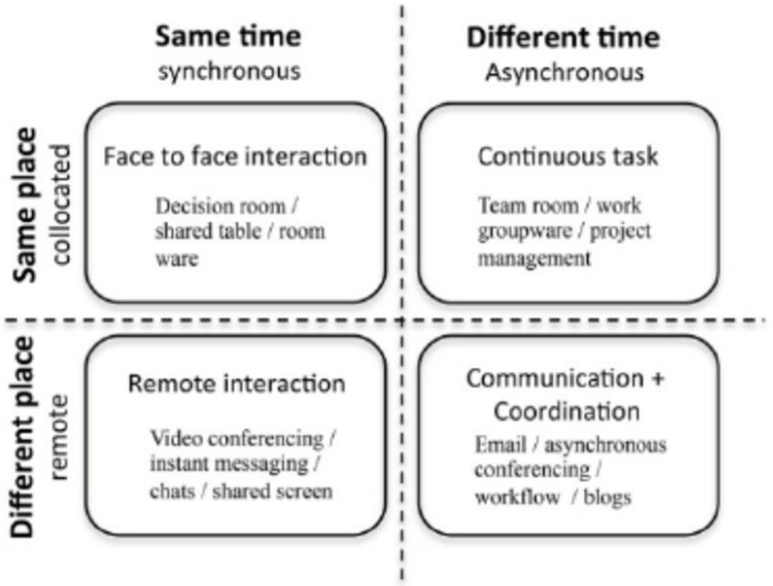
- Επικοινωνία μέσω υπολογιστή που υποστηρίζει άμεσο συμμετέχοντα.
- Συστήματα συναντήσεων και αποφάσεων που καταγράφουν την κοινή αντίληψη των συμμετεχόντων.

- Κοινόχρηστες εφαρμογές.
- Τεχνητά αντικείμενα που υποστηρίζουν την αλληλεπίδραση των συμμετεχόντων μέσω κοινόχρηστων αντικειμένων εργασίας

Το groupware είναι είτε σύγχρονο είτε ασύγχρονο (Εικόνα 22). Το σύγχρονο groupware είναι μια κατηγορία εφαρμογών που επιτρέπει σε μια ομάδα ατόμων που είναι φυσικά διαχωρισμένα να αλληλεπιδρούν μεταξύ τους χρησιμοποιώντας κοινόχρηστα υπολογιστικά αντικείμενα σε πραγματικό χρόνο. Η βασική απαίτηση του σύγχρονου groupware είναι ο συντονισμός σε πραγματικό χρόνο μεταξύ των πελατών. Οι διεπαφές χρήστη υποστηρίζουν την αίσθηση της ενότητας. Απαιτούν κοινό κανάλι ήχου για επικοινωνία.

Το ασύγχρονο συνεργατικό λογισμικό χρησιμοποιεί e-mail, δομημένα μηνύματα, πράκτορες, ροή εργασίας, πράκτορες διάσκεψης μέσω υπολογιστή, συστήματα κοινής χρήσης αρχείων και συνεργατικά συγγραφικά συστήματα, μεταξύ άλλων. Οι ασύγχρονες συνεργασίες μεταξύ των χρηστών διατηρούνται μόνο υπάρχει η δυνατότητα να συνεισφέρουν χωρίς περιορισμούς. Αυτό μπορεί να επιτευχθεί μέσω συστημάτων διαχείρισης αντιγράφων των δεδομένων με πρόσβαση ανάγνωσης ή εγγραφής των δεδομένων. Οι χρήστες μπορούν να εκτελούν ταυτόχρονες ενημερώσεις στα δεδομένα.

Η εκτεταμένη χρήση του groupware στο Διαδίκτυο συνείσφερε στην ανάπτυξη του Web 2.0, το οποίο χρησιμοποιεί την ανταλλαγή άμεσων μηνυμάτων, τη διάσκεψη στο Web, τα ημερολόγια ομάδων, την κοινή χρήση εγγράφων κ.λπ.



Εικόνα 22: Χρονική/Χωρική κατηγοριοποίηση του groupware

Τα συστήματα που υλοποιούν τις λειτουργίες του groupware είναι ανά κατηγορίες τα εξής:(Εικόνα 22)

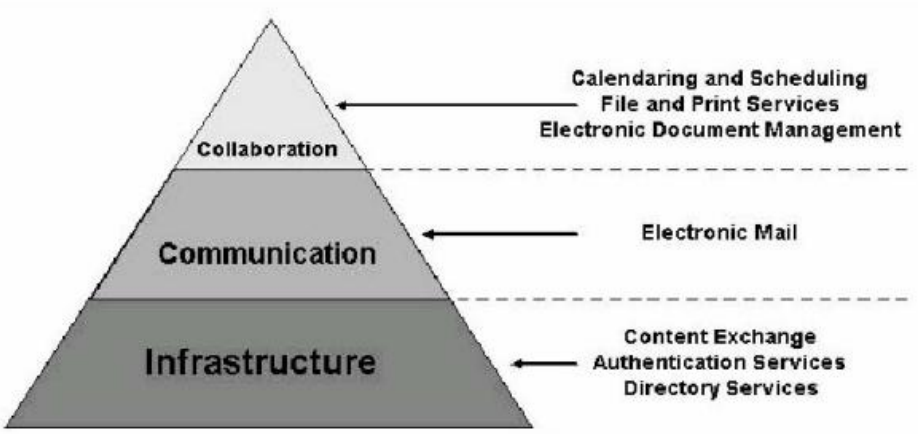
- Διαφορετικός χρόνος και τόπος
    - Email
    - Calendaring Scheduling
    - Workflow
- Ίδιος χρόνος και διαφορετικός τόπος
    - Chat room
    - Gaming

- Video Conferencing
- Shared Workspace

- Υβριδικά
  - Bulletin boards
  - Content exchange
  - Online Classrooms
  - Group Decision Support Systems and Electronic meeting rooms

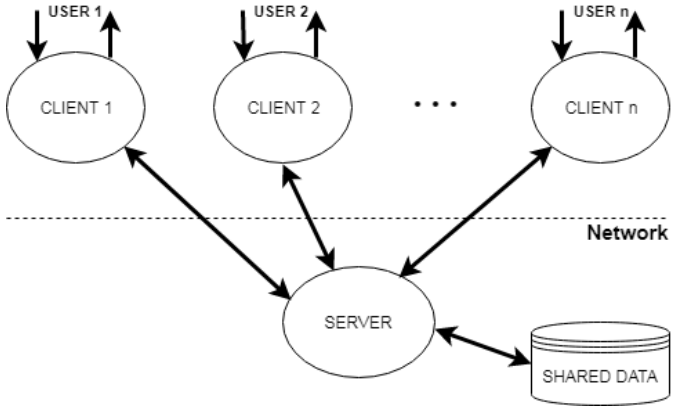
2.7.1 Αρχιτεκτονική Groupware

Όλα τα συστήματα groupware έχουν κοινά αρχιτεκτονικά στοιχεία στο front-end/client και στο back-end/server μέρος. (Εικόνα 23)



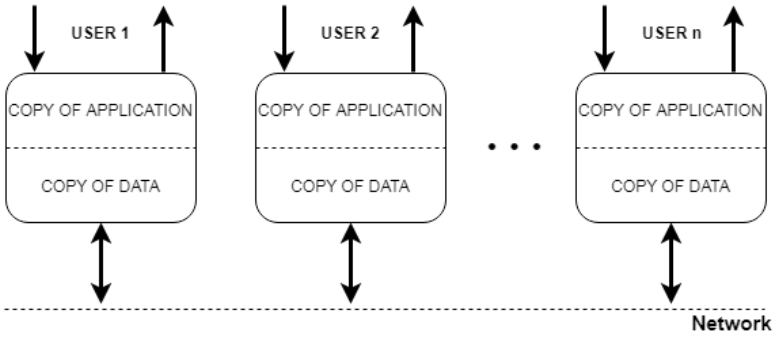
Εικόνα 23: Γενική αρχιτεκτονική

Παρακάτω (Εικόνα 24) βλέπουμε το μοντέλο της κεντροποιημένης (centralized) αρχιτεκτονικής groupware συστημάτων. Υπάρχει ένα μοναδικό αντίγραφο της και ακολουθείται η απλή περίπτωση αρχιτεκτονικής διακομιστή-πελάτη.



Εικόνα 24: Κεντροποιημένη αρχιτεκτονική

Παρακάτω (Εικόνα 25) βλέπουμε το μοντέλο της αρχιτεκτονικής δημιουργίας αντιγράφων (replicated) groupware συστημάτων, γνωστή και ως peer-peer. Υπάρχει ένα αντίγραφο της εφαρμογής και ένα αντίγραφο των δεδομένων σε κάθε σταθμό εργασίας. Αυτή η προσέγγιση δημιουργεί θέματα συνοχής.



Εικόνα 25: Αρχιτεκτονική δημιουργίας αντιγράφων

Η τρίτη αρχιτεκτονική καλείται υβριδική (Hybrid) γιατί συνδυάζει χαρακτηριστικά από τις άλλες δύο που αναφέραμε παραπάνω. Σε ένα hybrid groupware σύστημα υπάρχει αντίγραφο της εφαρμογής τοπικά στο σταθμό εργασίας καθώς και κρυφή μνήμη (cache) για γρήγορη ανταπόκριση. Συνεπώς η βάση δεδομένων βρίσκεται κεντρικά και το ρητό ή υπονοούμενο κλείδωμα (explicit or implicit locking) της βάσης δεδομένων και η διάδοση ενημερώσεων (updates propagation), σχετικά με την εισαγωγή, διαγραφή και τροποποίηση δεδομένων, αποτελούν κεντροποιημένες διαδικασίες.

2.7.2 Κοινόχρηστοι χώροι εργασίας (shared workspaces)

Μία σημαντική κατηγορία groupware συστημάτων είναι αυτή των εφαρμογών που υποστηρίζουν απομακρυσμένη συνεργασία μέσω ενός διαμοιραζόμενου χώρου εργασίας, μία επιφάνεια εργασίας όπου οι εταίροι συνεργάζονται με το χειρισμό εργαλείων και αντικειμένων εργασίας. Στην κατηγορία αυτή, συμπεριλαμβάνονται εφαρμογές όπως διαμοιραζόμενοι επεξεργαστές κειμένου, προγράμματα σχεδίασης και multi-player παιχνίδια.

Τα ομαδικά καθήκοντα σε αυτά τα συστήματα κατηγοριοποιούνται σε μερικούς τύπους. Σύμφωνα με τον McGrath, τα καθήκοντα σε κοινόχρηστους χώρους εργασίας συνήθως περιλαμβάνουν:

- δημιουργία νέων αντικειμένων,
- οργάνωση των υφιστάμενων αντικειμένων,
- εξερεύνηση του χώρου ή ενός συνόλου αντικειμένων
- κατασκευή μεγαλύτερων αντικειμένων από τα επιμέρους στοιχεία, ή
- τη διαχείριση ενός αυτόνομου συστήματος που παρουσιάζεται στον χώρο εργασίας

Για ένα έργο συνεργασίας που πρέπει να επιτευχθεί σε ένα κοινό χώρο εργασίας, πρέπει να συμβούν διάφορες δραστηριότητες. Κατ' αρχάς, πρέπει να εκτελεστεί η βασική εργασία. Αυτό το μέρος της εργασίας δεν είναι διαφορετικό για μια ομάδα από ό, τι για ένα άτομο, δεδομένου ότι οι ίδιες ενέργειες πρέπει να γίνουν για να φτάσουμε στην ολοκλήρωση της εργασίας. Ωστόσο οι περισσότερες ομάδες εργασίας περιλαμβάνουν ένα άλλο σύνολο εντελώς διαφορετικών δραστηριοτήτων. Αν θεωρήσουμε την εκτέλεση της εργασίας ως το κύριο καθήκον, τότε αυτό το σύνολο δραστηριοτήτων είναι η ομαδική εργασία. Ένα σύστημα groupware πρέπει να υποστηρίζει και την εργασία για το βασικό καθήκον και την ομαδική εργασία. προκειμένου να είναι πραγματικά χρήσιμο. Η ομαδική εργασία μπορεί να χωριστεί σε δύο τομείς: τα κοινωνικά και συναισθηματικά στοιχεία που αποτελούν την ομάδα και τη μηχανική της συνεργασίας. Ο τομέας που μας ενδιαφέρει είναι η μηχανική της συνεργασίας, δηλαδή οι ομαδικές

ενέργειες που πρέπει να συμβούν, επί πρόσθετα στις ατομικές, για να ολοκληρωθεί ένα καθήκον. Η μηχανική της συνεργασίας αποτελείται από τις ακόλουθες κύριες δραστηριότητες:

- *Αυθεντική επικοινωνία.* Η σκόπιμη ανταλλαγή μηνυμάτων μεταξύ των μελών της ομάδας αποτελεί ακρογωνιαίο λίθο της συνεργασίας.
- *Συνεπαγόμενη επικοινωνία.* Εκτός από την αυθεντική επικοινωνία, οι άνθρωποι μπορούν να λάβουν σημαντικές πληροφορίες οι οποίες εκπέμπονται μη ηθελημένα από τις δραστηριότητες τους. Αυτού του είδους η πληροφορία είναι σημαντική για την ομαλή λειτουργία της ομάδας. Οι δύο κύριοι τύποι συνεπαγόμενης επικοινωνίας περιλαμβάνουν πληροφορίες που προκύπτουν από τα αντικείμενα που χειρίζονται οι χρήστες του συστήματος ατομικά, και πληροφορίες που προκύπτουν από τις χαρακτηριστικές ενέργειες της ενσωμάτωσης ενός ατόμου μέσα στο χώρο εργασίας.
- *Συντονισμός δράσης.* Οι άνθρωποι οργανώνουν τις ενέργειές τους σε κοινό χώρο εργασίας ώστε να μην έρχονται σε αντιπαράθεση με άλλους. Οι κοινόχρηστοι πόροι και εργαλεία απαιτούν προγραμματισμένη χρήση. Ο κακός συντονισμός έχει ως αποτέλεσμα αντιπαραθέσεις χρηστών, διπλότυπες περιττές ενέργειες που έχουν ολοκληρωθεί από άλλους.
- *Σχεδίαση.* Ορισμένες δραστηριότητες σχεδιασμού είναι πολύ υψηλού επιπέδου για να θεωρηθούν ότι περιέχονται στη μηχανική της συνεργασίας, αλλά άλλες συμβαίνουν επανειλημμένα μέσα στο κοινόχρηστο χώρο εργασίας. Για παράδειγμα, οι άνθρωποι διανέμουν και επαναδιανέμουν το καθήκον όσο προχωρούν μαζί, δεσμεύουν περιοχές του χώρου εργασίας για δικιά τους χρήση ή εξετάστε διάφορους τρόπους ενεργειών με προσομοίωση τους στο χώρο εργασίας (π.χ. υποδεικνύοντας μια διαδρομή με δείκτη πριν αρχίσει η κατασκευή).
- *Παρακολούθηση.* Πολλές από τις ενέργειες της μηχανικής της συνεργασίας βασίζονται στην δυνατότητα παρακολούθησης και συλλογής πληροφοριών για τους άλλους στο χώρο εργασίας. Πολλές από τις πληροφορίες είναι απλά πληροφορίες ενημερότητας του χώρου εργασίας (workspace awareness). Δηλαδή πληροφορίες για την κατανόηση των ενεργειών των άλλων συμμετεχόντων καθώς και την επίγνωση των συμμετεχόντων. Υπάρχουν περιπτώσεις χρήσης όπου χρήστες παρακολουθούν τις ενέργειες άλλων πιο εμφανώς όπως π.χ. στην περίπτωση επίβλεψης ενός μαθητευόμενου.
- *Συνδρομή.* Τα μέλη μιας ομάδας παρέχουν βοήθεια σε ένα άλλο όταν χρειάζεται. Η βοήθεια μπορεί να είναι άτυπη και ευκαιριακή, ή μπορεί να ζητηθεί λεπτομερώς. Σε κάθε περίπτωση η κατάλληλη βοήθεια απαιτεί την επίγνωση του τι κάνουν και του σημείου που βρίσκονται στα καθήκοντα τους οι εταίροι που δέχονται τη βοήθεια.
- *Προστασία.* Ένας κίνδυνος στην ομαδική εργασία είναι ότι οι εταίροι μπορεί ακούσια να αλλάξουν ή να καταστρέψουν την εργασία που άλλοι έχουν κάνει. Οι εταίροι πρέπει επομένως να προσέχουν τα δικά τους έργα, παρατηρώντας τα αποτελέσματα που οι ενέργειες των άλλων θα μπορούσαν να φέρουν και να αναλάβουν δράση για να αποτρέψουν ορισμένα είδη δραστηριοτήτων.

2.7.3 Κοινόχρηστα αποθετήρια (shared repositories)

Κοινόχρηστα αποθετήρια καλούνται οι δεξαμενές δεδομένων στις οποίες εισάγει και αντλεί δεδομένα μία ομάδα στο διαδίκτυο. Μία τέτοια δεξαμενή θα μπορούσε να είναι ένας κοινόχρηστος φάκελος σε μια υπηρεσία νέφους για διαμοιρασμό αρχείων.

Μια δημοφιλής εκδοχή κοινόχρηστων αποθετηρίων, είναι τα αποθετήρια ελέγχου εκδόσεων. Τα αποθετήρια αυτά αναφέρονται σε δομές δεδομένων που βρίσκονται αποθηκευμένες σε κάποιον server και

περιέχουν, μεταξύ άλλων, το σύνολο των αρχείων και των φακέλων, ιστορικό σχετικά με αλλαγές που έχουν γίνει, αντικείμενα προς δέσμευση και πληροφορίες σχετικά με τα αντικείμενα αυτά.

Σε αυτήν την πτυχιακή εργασία, ως κοινόχρηστο αποθετήριο θεωρούμε το σύνολο του χώρου στον οποίο βρίσκεται το ψηφιακό περιεχόμενο, και αφορά υπηρεσίες νέφους διαμοιρασμού ψηφιακών αρχείων και εγγράφων (Google Drive. Τα μέλη της ομάδας βλέπουν αυτό το σύνολο ως μια ενοποιημένη δεξαμενή πληροφοριών, στην οποία μπορούν να εισάγουν, να τροποποιήσουν και να ανακτήσουν υλικό.

2.7.4 Υπηρεσίες υπολογιστικού νέφους και συνεργασία

Εννοιολογικά, η λέξη νέφος χρησιμοποιείται ως μεταφορά του Διαδικτύου και αναφέρεται σε όλη την υποδομή και το λογισμικό που καθιστά εφικτή τη λειτουργία του δικτύου και κατ' επέκταση του διαδικτύου. Ουσιαστικά το νέφος είναι το σύνολο των υποδομών, δικτύων, αποθηκευτικών μέσων, υπηρεσιών και διεπαφών, μέσω των οποίων είναι δυνατή η παροχή επεξεργαστικών και άλλων πόρων ως υπηρεσιών. Η παρεχόμενη υπηρεσία ενδέχεται να περιλαμβάνει λογισμικό, υποδομές, αποθηκευτικό περιεχόμενο ή και όλα αυτά ως ολοκληρωμένη πλατφόρμα μέσω του Διαδικτύου.

Στο υπολογιστικό νέφος η αποθήκευση, η επεξεργασία και η χρήση δεδομένων, λογισμικού και υπηρεσιών γίνεται διαδικτυακά, μέσω απομακρυσμένων υπολογιστών σε κεντρικά data center. Υπηρεσίες όπως η κατ' αίτηση παροχή εικονικών μηχανών, το διαδικτυακό ηλεκτρονικό ταχυδρομείο ή τα κοινωνικά δίκτυα συχνά βασίζονται στην τεχνολογία του υπολογιστικού νέφους. Οι καταναλωτές εξοικονομούν πόρους από την αγορά και συντήρηση λογισμικού, τη συντήρηση ακριβών εξυπηρετητών και εγκαταστάσεων αποθήκευσης δεδομένων.

Σύμφωνα με το Εθνικό Ινστιτούτο Προτύπων και Τεχνολογίας των Η.Π.Α., «το υπολογιστικό νέφος είναι ένα μοντέλο που καθιστά εφικτή την εύκολη και κατ' απαίτηση πρόσβαση μέσω δικτύου σε μία κοινόχρηστη δεξαμενή προσαρμόσιμων υπολογιστικών πόρων (π.χ. δίκτυα, διακομιστές, αποθηκευτικός χώρος, εφαρμογές και υπηρεσίες) που μπορούν να τροφοδοτηθούν/αναδιανεμηθούν και να τεθούν σε λειτουργία γρήγορα, με ελάχιστο κόστος και γνώσεις διαχείρισης ή αλληλεπίδραση με τον πάροχο της υπηρεσίας. Αυτό το μοντέλο αποτελείται από πέντε βασικά χαρακτηριστικά, τρία μοντέλα παροχής υπηρεσιών και τέσσερα μοντέλα ανάπτυξης» (34)

Τα κύρια χαρακτηριστικά, τα οποία είναι κοινά μεταξύ όλων των υπηρεσιών υπολογιστικού νέφους, είναι:

- *Αυτοεξυπηρετούμενη κατ' απαίτηση.* Ο καταναλωτής μπορεί να προμηθευτεί υπολογιστικούς πόρους όπως χρόνο στον διακομιστή και χώρο αποθήκευσης στο δίκτυο, όποτε το χρειαστεί αυτομάτως, χωρίς να απαιτείται κάποια ανθρώπινη ενέργεια με τον εκάστοτε πάροχο.
- *Ευρεία πρόσβαση μέσω δικτύου.* Οι δυνατότητες που προσφέρονται είναι διαθέσιμες σε όλο το δίκτυο και προσβάσιμες από συσκευές και πλατφόρμες διαφορετικού τύπου. (πχ κινητά, ταμπλέτες, υπολογιστές τσέπης, σταθεροί υπολογιστές κτλ.).
- *Διαθεσιμότητα πόρων.* Οι υπολογιστικοί πόροι του παρόχου διατίθενται με σκοπό να εξυπηρετήσουν πολλαπλούς χρήστες, με τους διαφορετικούς φυσικούς και εικονικούς πόρους να ανατίθενται σύμφωνα με τις απαιτήσεις του καταναλωτή
- *Ελαστικότητα.* Ένα από τα σημαντικότερα χαρακτηριστικά του υπολογιστικού νέφους σε σύγκριση με προγενέστερα υπολογιστικά συστήματα είναι η ελαστικότητα. Η ελαστικότητα αφορά στην αυξομείωση των υπολογιστικών πόρων του εκάστοτε συστήματος, ανάλογα με τις ανάγκες του για μία συγκεκριμένη χρονική περίοδο. Αυτό εξαλείφει την ανάγκη και το ρίσκο για

επενδύσεις σε τοπικές υλικές υποδομές. Η παραπάνω διαδικασία μπορεί να γίνει αυτόματα σε ορισμένες περιπτώσεις χωρίς την παρέμβαση του χρήστη.

- *Πληρωμή με βάση τη χρήση.* Οι υπολογιστικοί πόροι μετριούνται αναλυτικά, επιτρέποντας στους χρήστες να πληρώνουν ακριβώς μόνο για τους πόρους και το φόρτο εργασίας που χρησιμοποιούν.

2.7.4.1 Κατηγορίες υπηρεσιών νέφους

Software as a Service (SaaS). Το μοντέλο SaaS ορίζεται απλά ως εφαρμογές που βασίζονται (φιλοξενούνται)στο νέφος και είναι προσβάσιμες μέσω διαδικτύου ή ενός API. Ο πάροχος συγκροτεί κεντρικά εφαρμογές, οι οποίες είναι τυπικά προσβάσιμες μέσω διεπαφών διαδικτύου όπως ένας περιηγητής δικτύου σε συσκευές που καλύπτουν ένα ευρύ φάσμα όπως προσωπικούς υπολογιστές, tablets, smartphones και άλλες φορητές συσκευές. Ο χρήστης δεν μπορεί να επηρεάσει το διαδίκτυο, τους διακομιστές, τα λειτουργικά συστήματα ή τους αποθηκευτικούς χώρους και στις περισσότερες περιπτώσεις δεν έχει καθόλου (ή έχει περιορισμένο) έλεγχο πάνω στην ίδια την εφαρμογή.

Platform as a Services (PaaS). Στο μοντέλο PaaS, ο πάροχος διαθέτει στους χρήστες πλατφόρμες ανάπτυξης εφαρμογών και εργαλεία, για να αναπτύξουν τις δικές τους εφαρμογές σε μία υποδομή νέφους, χρησιμοποιώντας προγραμματιστικές γλώσσες και εργαλεία που έχουν δημιουργηθεί και υποστηρίζονται απ' αυτόν. Ο καταναλωτής δεν διαχειρίζεται ούτε ελέγχει το υπάρχον δίκτυο, τους διακομιστές, τα λειτουργικά συστήματα ή τους αποθηκευτικούς χώρους, αλλά μπορεί να ελέγξει τις ίδιες τις εφαρμογές και σε μερικές περιπτώσεις το περιβάλλον των εφαρμογών. Αντιπροσωπευτικά παραδείγματα για το PaaS είναι το Google App Engine και το Amazon Web Services.

Infrastructure as a Service (IaaS). Το IaaS είναι το μοντέλο στο οποίο ο πάροχος προσφέρει διακομιστές, αποθηκευτικό χώρο και άλλους εικονικούς υπολογιστικούς πόρους στο νέφος, στους καταναλωτές μέσω του διαδικτύου. Ο καταναλωτής δεν διαχειρίζεται ούτε ελέγχει την υφιστάμενη υποδομή του νέφους αλλά ελέγχει τα λειτουργικά συστήματα, την αποθήκευση, τις εφαρμογές ανάπτυξης και πιθανώς ελέγχει σε μικρότερο βαθμό κάποια στοιχεία όπως το τείχος προστασίας και την εξισορρόπηση φόρτου.

2.7.4.2 Μοντέλα ανάπτυξης

Δημόσιο υπολογιστικό νέφος. Ένα υπολογιστικό νέφος το οποίο διατίθεται ελεύθερα, προς χρήση, λέγεται δημόσιο. Οι υπηρεσίες του δημόσιου νέφους (εφαρμογές, χώρος αποθήκευσης και άλλοι πόροι) μπορούν να είναι δωρεάν. Γενικότερα, οι πάροχοι των υπηρεσιών δημόσιου υπολογιστικού νέφους κατέχουν και λειτουργούν την υποδομή σε δικά τους data centers και η πρόσβαση σε αυτές πραγματοποιείται μέσω Διαδικτύου. Δημοφιλείς υπηρεσίες δημοσίων νεφών αποτελούν οι Google Compute Engine (GCE), Google App Engine (GAE), Amazon Elastic Compute Cloud (EC2), Amazon Web Services (AWS), Microsoft Azure και Heroku.

Ιδιωτικό υπολογιστικό νέφος. Το ιδιωτικό υπολογιστικό νέφος είναι μία υποδομή υπολογιστικού νέφους που λειτουργεί αποκλειστικά για έναν μοναδικό οργανισμό. Η κατοχή και διαχείριση της υποδομής μπορεί να πραγματοποιείται εσωτερικά από τον οργανισμό ή εξωτερικά από τρίτους. Τα ιδιόκτητα data centers είναι εν γένει απαιτητικά σε κεφάλαια, καθώς απαιτούν κατανομή χώρων, υλικού και περιβαλλοντικό έλεγχο. Το ιδιωτικά υπολογιστικά νέφη έχουν επικριθεί, επειδή οι χρήστες πρέπει να αγοράζουν, να στήνουν και να διαχειρίζονται την υποδομή και ουσιαστικά δεν εκμεταλλεύονται τα οικονομικά πλεονεκτήματα που καθιστούν το υπολογιστικό νέφος μία τόσο ενδιαφέρουσα ιδέα.

Κοινοτικό υπολογιστικό νέφος. Ως κοινοτικό, χαρακτηρίζεται το υπολογιστικό νέφος που υλοποιείται προκειμένου να χρησιμοποιηθεί αποκλειστικά από μία συγκεκριμένη κοινότητα χρηστών από

οργανισμούς με καθορισμένο και κοινό σκοπό. Η υποδομή ενός κοινοτικού νέφους μπορεί να βρίσκεται στην κατοχή και διαχείριση ενός ή περισσοτέρων συμμετεχόντων οργανισμών, σε κάποιο εξωτερικό πάροχο ή ακόμα και σε συνδυασμό αυτών.

Υβριδικό υπολογιστικό νέφος. Το υβριδικό υπολογιστικό νέφος αποτελεί έναν συνδυασμό από δύο ή περισσότερες υποδομές νέφους (ιδιωτική, κοινοτική ή δημόσια), οι οποίες παραμένουν διακριτές οντότητες αλλά ταυτόχρονα συνδέονται μεταξύ τους, προσφέροντας τα οφέλη πολλαπλών μοντέλων ανάπτυξης. Η υποδομή του νέφους επιτρέπει την επέκταση των διαδικασιών μίας εφαρμογής νέφους σε συμπληρωματικά ανεξάρτητα νέφη όταν προκύψουν αυξημένες ανάγκες αποθήκευσης ή υπολογιστικής ισχύος (cloud bursting).

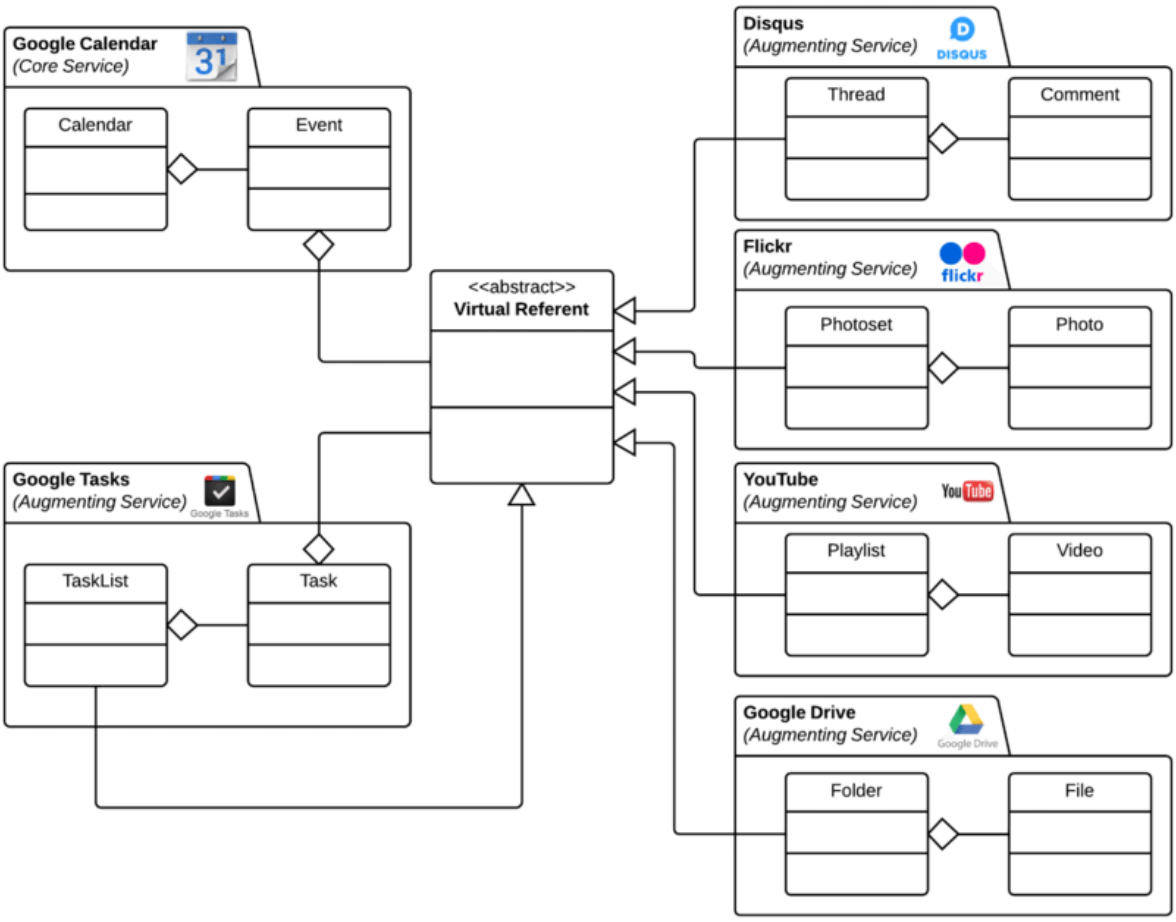
Σενάριο χρήσης & προσέγγιση

3.1 iSTLab Calendar

Οι εφαρμογές ηλεκτρονικού ημερολόγιου (online calendar) χρησιμοποιούνται κυρίως για χρονικό προσανατολισμό (προσαρμογή στα χρονικά περιθώρια ενός γεγονότος) και υπενθύμιση γεγονότων. Παρόλα αυτά η χρήση ενός online calendar μπορεί να απαιτεί χαρακτηριστικά, όπως προγραμματισμός και παρακολούθηση εργασιών, αρχειοθέτηση/ανάκληση και ανάκτηση καθηκόντων, τα οποία δεν υποστηρίζονται επαρκώς από τα διαθέσιμα συστήματα.

Το iSTLab Calendar είναι μία υπηρεσία online ημερολόγιου που βασίζεται στην υπηρεσία Google Calendar μέσω του αντίστοιχου API. Ενσωματώνει δυνατότητες για προσθήκη ξεχωριστού ψηφιακού υλικού στα ημερολογιακά γεγονότα. (Εικόνα 26) Υποστηρίζει διαμοιρασμό αρχείων μέσω της υπηρεσίας Google Drive (public API), φωτογραφιών και βίντεο μέσω της υπηρεσίας Flickr (public API), ηλεκτρονικές συζητήσεις μέσω της υπηρεσίας Disqus (public API), διαχείριση έργου/καθήκοντος μέσω της υπηρεσίας Asana (public API) και συλλογικούς σελιδοδείκτες. Η νέα υπηρεσία αποτελεί μία σύνθεση ψηφιακού υλικού το οποίο προσαρμόζεται και εξελίσσεται σταδιακά για να εξυπηρετήσει τις ανάγκες της εργασίας που σχεδιάζεται. Η σύνθετη υπηρεσία ημερολογίου βασίζεται στην αλληλεξαρτώμενη λειτουργία των επιμέρους διαφορετικών υπηρεσιών για να δημιουργήσει ένα ενοποιημένο σύστημα (οικοσύστημα) με «εξωτερικά» ξεχωριστά ψηφιακά αρχεία περιεχομένου. Αυτό έχει ως αποτέλεσμα τη συγκέντρωση συνόλων δεδομένων τα οποία μπορούν να ευθυγραμμίσουν όχι μόνο καθημερινές δραστηριότητες αλλά και οργανωτικές διαδικασίες. Τα σύνθετα ψηφιακά αντικείμενα στο iSTLab Calendar ενθαρρύνουν την ευέλικτη και υπεύθυνη συνεργατική εργασία, με την εκμετάλλευση κατανεμημένων πόρων.

Για τις ανάγκες της εφαρμογής που αναπτύχθηκε στο πλαίσιο της παρούσας πτυχιακής εργασίας, χρησιμοποιήθηκε το publications calendar (API), το ημερολόγιο το οποίο ενημερώνεται με τις δημοσιεύσεις ακαδημαϊκών, ερευνητικών συνεργατών και φοιτητών του εργαστηρίου iSTLab. Στο ημερολόγιο δημιουργούνται «πλούσια» γεγονότα τα οποία εκτός τα προφανή στοιχεία (ημερομηνία, ώρα, τίτλος κλπ.) μπορεί να φέρει ετικέτες χαρακτηρισμού του γεγονότος (event tags) ετικέτες που περιγράφουν το αντικείμενο της δημοσίευσης (λέξεις-κλειδιά). Ο δημιουργός του γεγονότος μπορεί να επισυνάψει το αρχείο της δημοσίευσης και άλλο υλικό μέσω του Google Drive. Επιπλέον, η εφαρμογή επιτρέπει την ενσωμάτωση βίντεο μέσω του YouTube, φωτογραφιών-βίντεο μέσω του Flickr και καθηκόντων μέσω του Asana.



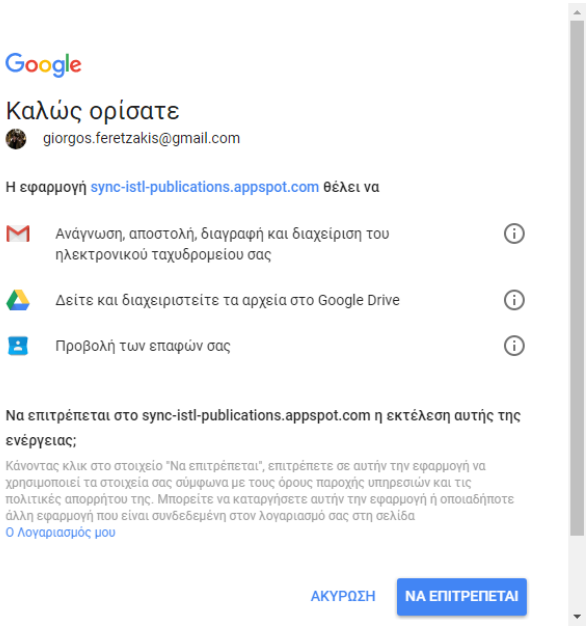
Εικόνα 26: Αρχιτεκτονική iSTLab Calendar

3.2 Εργαλεία που αξιοποιήθηκαν

3.2.1 Google Sign-In

Η διαδικασία Google Sign-In χρησιμοποιεί το πρωτόκολλο OAuth 2.0 για την ενσωμάτωση μιας εφαρμογής με τα Google APIs. Για την ταυτοποίηση των χρηστών απαιτείται το client ID της εφαρμογής και έπειτα, ένας λογαριασμός Google. Το authorization request (αίτημα ταυτοποίησης) αποστέλλεται στον Google OAuth εξυπηρετητή μέσω του Sign-In κουμπιού που ενσωματώνει ο προγραμματιστής στην ιστοσελίδα της εφαρμογής του. Στη συνέχεια αν ο χρήστης δεν έχει χορηγήσει δικαιώματα χρήσης στην εφαρμογή, εμφανίζεται ένα παράθυρο διαλόγου στο οποίο του ζητείται η χορήγηση δικαιωμάτων (Εικόνα 27) στα προσωπικά του δεδομένα ανάλογα με τις ανάγκες αυτής (αλληλογραφία Gmail, Google Docs, αρχεία στο Drive, επαφές Google κ.ά.). Έπειτα ο χρήστης λαμβάνει ένα token (**access_token**) το οποίο του δίνει του δίνει πρόσβαση στα APIs στα οποία προηγουμένως χορήγησε δικαιώματα. Μεταξύ άλλων, λαμβάνει και το authorization code το οποίο το στέλνει στο backend εξυπηρετητή της εφαρμογής (Εικόνα 28) και αυτός με τη σειρά του, το ανταλλάσσει με το εξυπηρετητή του API (Google API Server) για ένα άλλο token (**access_token**) το οποίο χρησιμοποιείται από το backend για κλήσεις στο API εκ μέρους του

χρήστη (πιστοποίηση του χρήστη στο backend) και μπορεί να ανανεώνεται με το **refresh token,** το οποίο επιστρέφεται από τον εξυπηρετητή του API μαζί με το access_token. (35)



Εικόνα 27: Χορήγηση δικαιωμάτων στην εφαρμογή

```
    else if (googleUser.code) {
        var auth2 = gapi.auth2.getAuthInstance();
        auth2.isSignedIn.listen(function (a) {
            email = getUserInformation(auth2.currentUser.get());
            sendToAuthServlet(googleUser.code, "false", email);
            makeApiCall();
        });
    }
```

Εικόνα 28: Σε αυτό το τμήμα κώδικα το αντικείμενο που επιστρέφεται από την ταυτοποίηση του λογαριασμού Google του χρήστη, αν περιέχει κωδικό πιστοποίησης (authorization code) θα σταλεί στο backend για την ανταλλαγή του με access και refresh tokens και στη συνέχεια θα καλέσει το API για την άντληση των μεταδεδομένων των αρχείων της εφαρμογής.

3.2.2 Google Drive API

Η πλατφόρμα του Drive προσφέρει APIs και βιβλιοθήκες, καθώς και documentation για την ανάπτυξη εφαρμογών που ενσωματώνουν τα χαρακτηριστικά του. Η βασική λειτουργικότητα περιλαμβάνει μεταφόρτωση και λήψη αρχείων στο σύννεφο. Όμως, το Drive προσφέρει πολλά περισσότερα από απλή αποθήκευση αρχείων:

- *Αναζήτηση αρχείων.* Χρησιμοποιώντας τη μέθοδο **files.list** του Drive API μπορούμε να αναζητήσουμε και να ανακτήσουμε συγκεκριμένα αρχεία, με κριτήρια που αφορούν στα μεταδεδομένα των αρχείων.(Εικόνα 29)
- *Διανομή και προώθηση.* Οι εφαρμογές που ενσωματώνουν τα χαρακτηριστικά του Drive μπορούν να είναι διαθέσιμες μέσω του Connected Apps to Drive σε σχετική ενότητα στην ιστοσελίδα του Google Drive.
- *Διαμοιρασμός και συνεργασία.* Οι εφαρμογές που αναπτύσσονται με τη χρήση του Drive API μπορούν να διαχειριστούν δικαιώματα χρήσης και ρυθμίσεις διαμοιρασμού των αρχείων.

Επιπλέον, παρέχεται δυνατότητα σχολιασμού και συζητήσεων για τα αρχεία, επιτρέποντας τη συνεργατική εργασία στους χρήστες που τα μοιράζονται.

- *Άνοιγμα και δημιουργία αρχείων με το Google Picker.* Τα αρχεία του Drive μπορούν να ανοίξουν μέσω του Google Picker, το οποίο φέρει τη μορφή και τα χαρακτηριστικά του Drive όμως είναι ανεξάρτητο από το UI του Drive.
- *Χρήση συντομεύσεων.* Μπορούν να δημιουργηθούν συντομεύσεις για αρχεία που βρίσκονται εκτός Drive, σε διαφορετικό τόπο φιλοξενίας, σε κάποιο datastore ή υπηρεσία σύννεφου. Αυτές οι συντομεύσεις μπορούν να έχουν τα χαρακτηριστικά των αρχείων του Drive όπως διαμοιρασμός, άνοιγμα, δημιουργία και ευρετηρίαση στην αναζήτηση.
- *Εξαγωγή και μετατροπή Google-αρχείων.* Η πλατφόρμα του Drive επιτρέπει στους προγραμματιστές άνοιγμα, εισαγωγή και εξαγωγή Google Docs όπως Google Spreadsheets, Presentations, Documents, and Drawings. Για παράδειγμα, αν μία εφαρμογή προορίζεται για άνοιγμα PDF αρχείων, οι χρήστες θα μπορούν να ανοίγουν Google Documents επειδή τα εν λόγω έγγραφα μπορούν να εξαχθούν σε PDF. (36)

```
function retrieveAllFiles(callback) {
    var retrievePageOfFiles = function (request, result) {
        request.execute(function (resp) {
            result = result.concat(resp.items);
            var nextPageToken = resp.nextPageToken;
            if (nextPageToken) {
                request = gapi.client.drive.files.list({
                    'orderBy': 'folder, createdDate desc',
                    'q': "properties has { key='publicationResource' and
value='true' and visibility='PUBLIC' }",
                    'pageToken': nextPageToken
                });
                retrievePageOfFiles(request, result);
            } else {
                callback(result);
            }
        });
    }
    var initialRequest = gapi.client.drive.files.list({
        'orderBy': 'folder, createdDate desc',
        'q': "properties has { key='publicationResource' and value='true'
and visibility='PUBLIC' }"
    });
    retrievePageOfFiles(initialRequest, []);
}
```

Εικόνα 29: Μέθοδος Drive API για την άντληση των μεταδεδομένων των αρχείων της εφαρμογής

3.2.3 Gmail API

Το Gmail API είναι ένα RESTful API το οποίο μπορεί να χρησιμοποιηθεί για πρόσβαση στην αλληλογραφία Gmail και αποστολή αλληλογραφίας. Χαρίζει ευελιξία, RESTful πρόσβαση στα εισερχόμενα μηνύματα του χρήστη παρέχοντας φυσικές διεπαφές για Threads (Νήματα), Messages (Κύρια), Drafts (Πρόχειρα), Labels (Ετικέτες), History (Ιστορικό), και Settings (Ρυθμίσεις). Το API μπορεί να προσθέσει σε εφαρμογές τα εξής χαρακτηριστικά του Gmail:

- Ανάγνωση μηνυμάτων από το Gmail
- Αποστολή μηνυμάτων

- Τροποποίηση ετικετών σε μηνύματα και νήματα
- Αναζήτηση συγκεκριμένων μηνυμάτων και νημάτων
- Δημιουργία φίλτρων για αυτόματη επισήμανση με ετικέτες, προώθηση ή αρχειοθέτηση μηνυμάτων.

Τυπικές περιπτώσεις χρήσης περιλαμβάνουν διάφορες εφαρμογές που στοχεύουν σε:

- Εξαγωγή, ευρετηρίαση και δημιουργία αντιγράφων ασφαλείας για αλληλογραφία.
- Διαχείριση ετικετών (προσθήκη/αφαίρεση)
- Αυτόματη αποστολή μηνυμάτων
- Μετεγκατάσταση λογαριασμών ταχυδρομείου σε άλλους παρόχους
- Ορισμό τυποποιημένων υπογραφών ηλεκτρονικού ταχυδρομείου σε έναν ιστότοπο. (37)

3.2.4 Google Contacts API

Το Google Contacts API δίνει τη δυνατότητα επισκόπησης και διαχείρισης των επαφών ενός χρήστη σε εφαρμογές-πελάτες. Οι επαφές είναι αποθηκευμένες στον Google λογαριασμό του χρήστη. Ο client κώδικας μια εφαρμογής μπορεί να δημιουργεί νέες επαφές, να επεξεργάζεται ή να διαγράφει υπάρχουσες και να ζητάει επαφές σύμφωνα με συγκεκριμένα κριτήρια. (38)

3.2.5 LinkedIn REST API

Το REST API του LinkedIn είναι ο πυρήνας όλων των προγραμματιστικών ενεργειών μέσω του LinkedIn. Τα JavaScript και Mobile SDKs είναι απλά μεσολαβητές για το REST API, που παρέχουν ένα επίπεδο ευκολίας και οικειότητας για τους προγραμματιστές.

Μία εφαρμογή που χρησιμοποιεί το LinkedIn API για να έχει πρόσβαση σε δεδομένα χρηστών και να στέλνει δεδομένα ενεργώντας εκ μέρους του χρήστη, πρέπει να ελέγχεται για την ταυτότητα αυτού. Το LinkedIn βασίζεται στο πρωτόκολλο OAuth 2.0 για τη χορήγηση δικαιωμάτων πρόσβασης, το οποίο χαρακτηρίζεται από απλότητα και ευκολία χρήσης.

Το αίτημα για δεδομένα στο API επιστρέφει τις πληροφορίες σε XML (by default) ή JSON μορφή. Μερικά API calls απαιτούν την αποστολή δεδομένων (όπως ο διαμοιρασμός περιεχομένου στο LinkedIn profile/page), η οποία μπορεί επίσης να γίνει σε XM ή JSON μορφή. (39)(Εικόνα 30)

```
    shareContent: function () {
        var payload = {
            "comment": "Check an iSTLab related publication. *Shared by
iSTLab Publications Manager  " + this.title + ' : ' + this.description + ' '
+ this.link,
            "visibility": {
                "code": "anyone"
            }
        };
        IN.API.Raw("/people/~/shares?format=json")
                .method("POST")
                .body(JSON.stringify(payload))
                .result(this.onSuccess1)
                .error(this.onError1);
    },
    shareContentCompany: function () {
        var payload = {
```

```
            "comment": "Check an iSTLab related publication. *Shared by
iSTLab Publications Manager  " + this.title + ' : ' + this.description + ' '
+ this.link,
            "visibility": {
                "code": "anyone"
            }
        };
        var cpnyID = "15189510";
        IN.API.Raw("/companies/" + cpnyID + "/shares?format=json")
                .method("POST")
                .body(JSON.stringify(payload))
                .result(this.onSuccess1)
                .error(this.onError1);
    }
```

Εικόνα 30: Μέθοδοι κοινοποίησης περιεχομένου στο LinkedIn του object LinkedInAction

3.2.6 Google App Engine

Το Google App Engine είναι διαδικτυακό πλαίσιο εργασίας (web framework) και πλατφόρμα cloud computing για ανάπτυξη και φιλοξενία διαδικτυακών εφαρμογών στα datacenters της Google.

3.2.7 Materialize

Το Materialize είναι ένα front-end πλαίσιο εργασίας (css framework) το οποίο βασίζεται στο Material Design της Google. (40)Διαθέτει εύχρηστα συστατικά γραφικής διεπαφής χρήστη (UI components) και παρέχει κλάσεις μορφοποίησης (CSS style classes), animations και πλέγματα διάταξης (grid layouts) για την κατασκευή καλαίσθητων και responsive ιστοσελίδων. Δημιουργήθηκε με HTML, CSS και JavaScript. Γενικά, τα συνήθη συστατικά των front-end πλαισίων εργασίας είναι:

- CSS κώδικας για τη δημιουργία ενός πλέγματος διάταξης των γραφικών στοιχείων που απαρτίζουν την διεπαφή. Επιτρέπει στον προγραμματιστή να τοποθετήσει τα διάφορα στοιχεία σε μία απλή και προσαρμόσιμη δομή.
- Τυπογραφικά σχέδια για HTML στοιχεία
- Λύσεις για ασυμβατότητα του προγράμματος πλοήγησης.
- Δημιουργία CSS κλάσεων για τη μορφοποίηση των γραφικών συστατικών της διεπαφής.

Τέλος, παρέχουν λύσεις για γραφική προσαρμογή της διεπαφής σε οποιαδήποτε desktop ή mobile συσκευή ακολουθώντας responsive σχεδίαση.

3.2.7.1 Material Design

Το Material Design είναι ένα σύνολο σχεδιαστικών κανόνων (design language) που δημιουργήθηκε από τη Google και ανακοινώθηκε για πρώτη φορά στο συνέδριο Google I/O στις 25 Ιουνίου, 2014. Επεκτείνει το μοντέλο των "καρτών" που είναι σχεδιασμός, βασισμένος σε διάταξη πίνακα (grid layout), με κινούμενα σχέδια και μεταβάσεις που ανταποκρίνονται στις κινήσεις των χρηστών, χρήση padding και εφέ βάθους με κατάλληλο φωτισμό και σκίαση των αντικειμένων. Ο σχεδιαστής Matías Duarte εξήγησε ότι " σε αντίθεση με το πραγματικό χαρτί, το ψηφιακό υλικό μπορεί να επεκταθεί και να μορφοποιηθεί έξυπνα. Το material έχει φυσικές επιφάνειες και άκρα. Οι σκιές δίνουν νόημα σε αυτό που αγγίζεις.". Η Google υποστηρίζει ότι η καινούρια σχεδιαστική τους γλώσσα βασίζεται στο χαρτί και το μολύβι.

Το Material Design μπορεί να χρησιμοποιηθεί στο Android από την έκδοση 2.1 και πάνω, μέσω της βιβλιοθήκης v7 appcompat, που χρησιμοποιείται σε όλες σχεδόν τις συσκευές Android που φτιάχτηκαν από το 2009 και μετά. Το Material Design σταδιακά θα επεκταθεί σε όλα τα προϊόντα της Google τόσο στο web όσο και στο mobile, προσφέροντας ομαλή εμπειρία σε όλες τις πλατφόρμες και εφαρμογές της. Η Google έχει κυκλοφορήσει επίσης και APIs (application programming interfaces) για τους προγραμματιστές που θέλουν να ενσωματώσουν την σχεδιαστική αυτή γλώσσα στις εφαρμογές τους.

Από το 2015, το Material Design ενσωματώθηκε στις περισσότερες εφαρμογές κινητών τηλεφώνων της Google στο Android όπως τα Gmail, Youtube, Google Drive, Google Docs, Sheets και Slides, Google Maps, Inbox και σε μικρότερο βαθμό στον Chrome browser και το Google Keep. Επίσης έχει ενσωματωθεί και στις web εφαρμογές για desktop στα Google Drive, Docs, Sheets, Slides και Inbox.

Η εφαρμογή των κανόνων του Material Design για UIs (User Interfaces) web εφαρμογών ονομάζεται polymer paper elements. Αποτελείται από τη βιβλιοθήκη Polymer, η οποία παρέχει ένα Web Components API για browsers που δεν εφαρμόζουν τοπικά τους κανόνες του material. (41)

3.2.8 jQuery

Η jQuery είναι μια ελαφριά, cross-browser JavaScript βιβλιοθήκη που δίνει έμφαση στην αλληλεπίδραση μεταξύ JavaScript και HTML. Κυκλοφόρησε τον Ιανουάριο του2006 από τον John Resig και πλέον χρησιμοποιείται σε περισσότερο από το 27% των 10.000 δημοφιλέστερων διαδικτυακών τόπων – καθιστώντας την ως την πιο δημοφιλή βιβλιοθήκη JavaScript σήμερα. Η βιβλιοθήκη jQuery είναι δωρεάν, ανοιχτού κώδικα λογισμικό υπό διπλή άδεια χρήσης MIT License και GNU General Public Lisence, Version 2. Η σύνταξη της έχει ως στόχο να την εύκολη πλοήγηση κόμβων ενός εγγράφου, επιλέγοντας στοιχεία του DOM, δημιουργώντας animations, διαχειρίζοντας γεγονότα (events), και αναπτύσσοντας διαδικτυακές εφαρμογές AJAX. Η jQuery παρέχει επίσης την δυνατότητα στους προγραμματιστές να δημιουργήσουν plugins στην ίδια την βιβλιοθήκη, με τέτοιο τρόπο ώστε να την επεκτείνουν. Με την παροχή αυτής της δυνατότητας, οι προγραμματιστές μπορούν να δημιουργήσουν χαμηλού επιπέδου αλληλεπιδράσεις και animations, προχωρημένα εφέ και υψηλού επιπέδου widgets. Το παραπάνω συμβάλλει δραστικά στην δημιουργία ισχυρών και δυναμικών ιστοσελίδων στα πρότυπα του Web 2.0. (42)

3.2.8.1 jQuery Plugins

Για την ανάπτυξη του front-end της εφαρμογής χρησιμοποιήθηκαν plugins που βασίζονται στη jQuery και προσθέτουν νέες λειτουργίες. Χρησιμοποιήθηκαν τα εξής:

- MixItUp. Το MixItUp είναι ένα jQuery plugin που παρέχει μηχανισμούς φιλτραρίσματος και ταξινόμησης με animation. Είναι ιδανικό για τη διαχείριση κάθε κατηγοριοποιημένου ή να διατεταγμένου περιεχόμενου, όπως portfolios, galleries και blogs, αλλά μπορεί επίσης να λειτουργήσει ως ένα ισχυρό εργαλείο το οποίο εμπλέκει UI εφαρμογές με την οπτικοποίησης δεδομένων.
- DevBridge Autocomplete. Επιτρέπει στους χρήστες να βρίσκουν και να επιλέγουν γρήγορα από μια προ συμπληρωμένη λίστα τιμών, καθώς πληκτρολογούν, αξιοποιώντας την αναζήτηση και το φιλτράρισμα.
- jQCloud. Δημιουργεί με απλή HTM και CSS καλαίσθητα σύννεφα ετικετών (tag clouds).

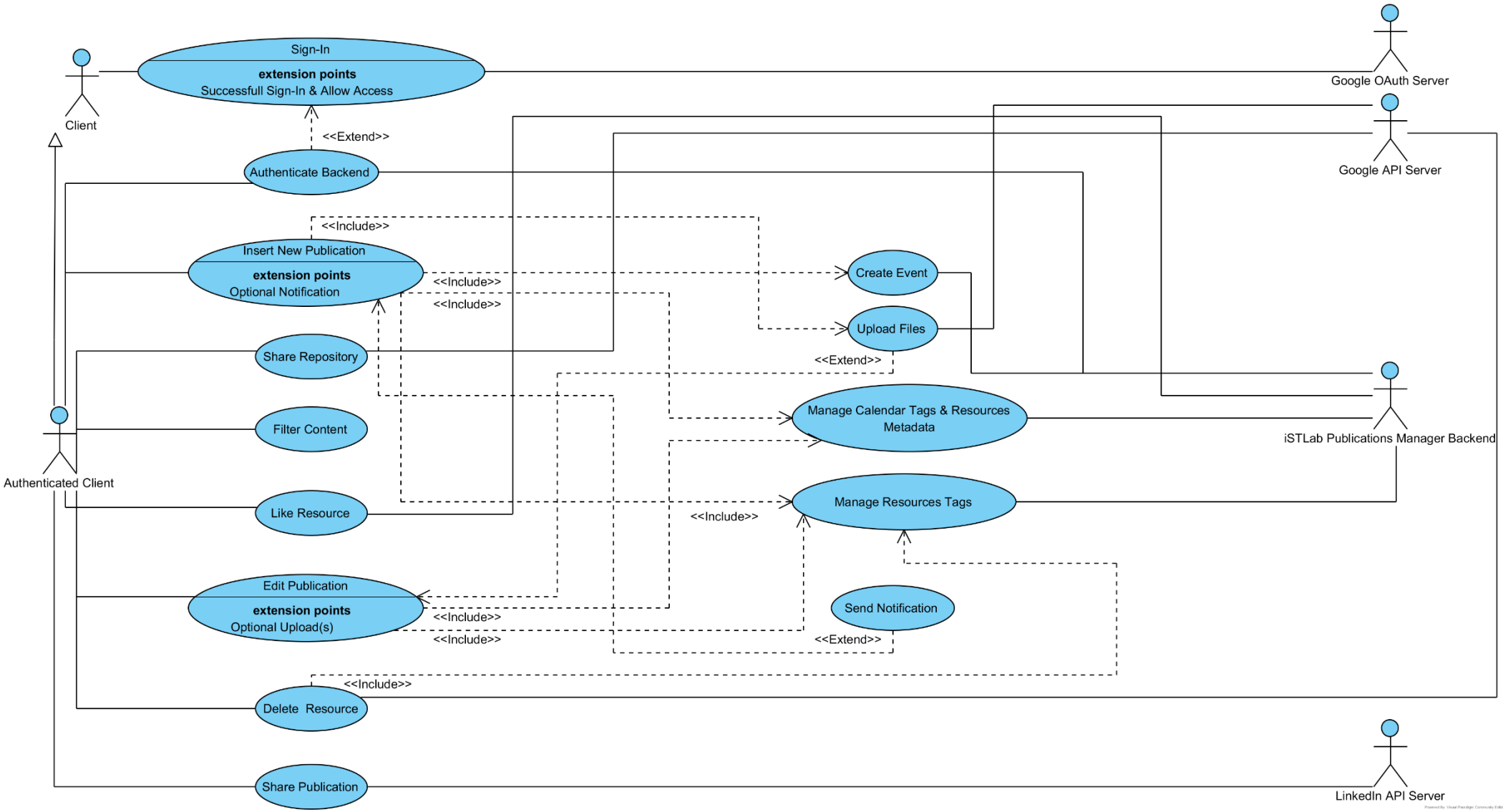
3.3 Λογική σχεδίαση

3.3.1 Διάγραμμα περιπτώσεων χρήσης

Το διάγραμμα περιπτώσεων χρήσης μοντελοποιεί την αλληλεπίδραση του χρήστη με ένα σύστημα υποδεικνύοντας τις σχέσεις ανάμεσα στο χρήστη και τις περιπτώσεις χρήσης που εμπλέκεται, καθώς και τις υπόλοιπες σχέσεις των στοιχείων του συστήματος. Επιπλέον το διάγραμμα περιπτώσεων χρήσης εκφράζει τη λειτουργικότητα και τις προδιαγραφές του συστήματος. Περιέχει τέσσερα συστατικά στοιχεία: Το πεδίο συστήματος (boundary) το οποίο ορίζει ένα ξεχωριστό σύστημα και περικλείει περιπτώσεις χρήσης, τα άτομα (actors) τα οποία σχετίζονται με τις περιπτώσεις χρήσεις μέσω ενός συγκεκριμένου ρόλου, τις περιπτώσεις χρήσης (use cases) οι οποίες είναι ενέργειες που καθορίζουν τις αλληλεπιδράσεις μεταξύ ενός ατόμου και ενός συστήματος για την επίτευξη ενός στόχου.

Το διάγραμμα περιπτώσεων χρήσης στην Εικόνα 31 που ακολουθεί αναπαριστά τα σενάρια χρήσης της εφαρμογής:

- Ο χρήστης ταυτοποιείται με τον Google λογαριασμό του.
- Αμέσως μετά, ο επικυρωμένος χρήστης πιστοποιείται και στο backend της εφαρμογής για κλήσεις από το backend στα Google APIs εκ μέρους του χρήστη.
- Ο χρήστης αφού ολοκληρώσει τη διαδικασία του authentication και μπορεί να εισάγει μία νέα δημοσίευση η οποία περιλαμβάνει τη δημιουργία του event στο Calendar, τη μεταφόρτωση των αρχείων της δημοσίευσης, την προσθήκη μεταδεδομένων στο γεγονός και τα τεκμήρια του Calendar καθώς και την επισήμανση με ετικέτες των τεκμηρίων στο Datastore. Εφόσον ο χρήστης επιθυμεί, ενημερώνονται με email οι εταίροι του διαμοιραζόμενου αποθετηρίου.
- Ο χρήστης μπορεί να φιλτράρει το περιεχόμενο βάσει κριτηρίων που αφορούν στα μεταδεδομένα του.
- Ο χρήστης μπορεί να δηλώσει ότι του αρέσει μία δημοσίευση.
- Ο χρήστης μπορεί να επεξεργαστεί τα μεταδεδομένα και τις ετικέτες, το event και να προσθέσει τεκμήρια στη δημοσίευση.
- Ο χρήστης μπορεί να διαγράψει τεκμήρια.
- Ο χρήστης μπορεί να κοινοποιήσει τη δημοσίευση στο LinkedIn profile του ή και στη LinkedIn σελίδα του εργαστηρίου, εφόσον είναι ένας από τους διαχειριστές της σελίδας.



Εικόνα 31: Διάγραμμα περιπτώσεων χρήσης

3.3.2 Διάγραμμα κλάσεων και ανάλυση λειτουργικότητας κλάσεων

Τα δομικά στοιχεία ενός αντικειμενοστραφούς συστήματος αποτελούνται από του τις κλάσεις και τις σχέσεις μεταξύ των κλάσεων, οι οποίες επιτρέπουν τη συνεργασία αντικειμένων που δημιουργούνται ως στιγμιότυπα των κλάσεων ή με άμεση πρόσβαση στα στατικά στοιχεία και τα χαρακτηριστικά τους. Το διάγραμμα κλάσεων αποτελείται από τις κλάσεις του συστήματος και τις μεταξύ τους συσχετίσεις, περιγράφοντας με αυτό τον τρόπο τη στατική δομή του συστήματος. Το backend της εφαρμογής είναι γραμμένο σε Java με χρήση του Google App Engine framework το οποίο βασίζεται στο Jetty (Java Servlet container & Java HTTP Web Server). Το backend αποτελείται από πέντε (4) servlets (κλάσεις AuthServlet, Servlet, WatchFolder, Channel και CredentialsCron). Το Java Source Package "com.istl" συμπληρώνουν οι κλάσεις GDrive, Gmail, SocialDatastore, CheckCalendar, MyListener και η final Constant. Οι AuthServlet και Servlet είναι οι διεπαφές αλληλεπίδρασης με το client μέρος. Τα υπόλοιπα servlets χρησιμεύουν ως cron services του Google App Engine, στην ανακάλυψη νέων αρχείων σε events και tags που προστέθηκαν από την εφαρμογή του iSTLab Calendar και παρέχουν μεθόδους αλληλεπίδρασης με τα APIs. Τα crons εκτελούν χρονοπρογραμματισμένες εργασίες για την παρακολούθηση των πόρων του αποθετηρίου στο Google Drive καθώς και την αρχικοποίηση και διαχείριση των Google Credentials, των αντικειμένων που αποτελούν τα πιστοποιητικά της εφαρμογής και του χρήστη για οποιαδήποτε αλληλεπίδραση με τα Google APIs. κάθε servlet εξυπηρετεί HTTP αιτήματα και υλοποιεί τη διεπαφή (interface) **javax.servlet.Servlet.** Επεκτείνει την **javax.servlet.http.HttpServlet**, μία αφηρημένη κλάση η οποία υλοποιεί τη Servlet διεπαφή και έχει τις κατάλληλες μεθόδους για το χειρισμό HTTP αιτημάτων. Αναλυτικότερα οι κλάσεις του συστήματος:

- Constants (final): Περιέχει τις σταθερές του συστήματος και όλα τα πεδία της είναι static. Επιπλέον, περιέχει μεθόδους για συνδιαλλαγές με το Datastore. Τα στελέχη της χρησιμοποιούνται από τις κλάσεις AuthServlet, MyListener, CredentialsCron, (Εικόνα 35) Servlet, GDrive και SocialDatastore. (Εικόνα 36)
- AuthServlet: Περιέχει μεθόδους για την πιστοποίηση του χρήστη στο backend και τη δημιουργία credentials με τον κωδικό πιστοποίησης (ανταλλαγή authorization code, Εικόνα 32) ή το αποθηκευμένο refresh token. Χρησιμοποιεί όλα τα στελέχη της Constants εκτός τη μέθοδο getEntities. (Εικόνα 35)
- MyListener: Η μέθοδος contextInitialized εκτελείται με warmup αίτημα στην αρχικοποίηση του instance της εφαρμογής στο App Engine και δημιουργεί το Google credential της εφαρμογής (service account credential) πριν από οποιαδήποτε κλήση servlet του συστήματος. (Εικόνα 35)
- CredentialsCron: Αποτελεί κλάση-χειριστή για cron job. Δέχεται αίτημα κάθε είκοσι λεπτά για να ανανεώσει τα credentials των χρηστών και της εφαρμογής (service account credential) που πλησιάζουν στη λήξη. Τα credentials των χρηστών δομούνται στο HashMap CREDENTIALS_CACHE της Constants. (Εικόνα 35)
- Servlet: Περιλαμβάνει τις μεθόδους που χρησιμοποιούνται για την εισαγωγή νέου περιεχομένου και πιο συγκεκριμένα για τη δημιουργία event και τη προσθήκη μεταδεδομένων/ετικετών. Περιέχει ακόμη, βοηθητικές μεθόδους που αντλούν δεδομένα από το iSTLab Calendar και μετατροπή JSON αντικειμένων σε λίστες. Χρησιμοποιεί μεθόδους (Εικόνα 36) από όλες τις βοηθητικές κλάσεις (για αποστολή ενημέρωσης, διαχείριση likes) και τη στατική μέθοδο datastoreTransactions της Constants για τις συνδιαλλαγές με το datastore.

```
    private static void exchangeAuthCodeForCredential(String code) throws IOException {
        String accessToken, refreshToken;
        HttpTransport HTTP_TRANSPORT = null;
        JsonFactory JSON_FACTORY = null;
        GoogleClientSecrets clientSecret
                =     GoogleClientSecrets.load(JSON_FACTORY,     new     InputStreamReader(new
FileInputStream(
                                        new File("WEB-INF/client_secret.json"))));
        GoogleAuthorizationCodeFlow gflow = new GoogleAuthorizationCodeFlow.Builder(
                HTTP_TRANSPORT, JSON_FACTORY, clientSecret, Constants.SCOPES)
                .setAccessType("offline")
                .setApprovalPrompt("force")
                .build();
        GoogleTokenResponse resp = gflow
                .newTokenRequest(code)
                .setRedirectUri(Constants.REDIRECT_URI)
                .execute();
        GoogleIdToken idToken = resp.parseIdToken();
        GoogleIdToken.Payload payload = idToken.getPayload();
        String email = payload.getEmail();
        accessToken = resp.getAccessToken();
        refreshToken = resp.getRefreshToken();
        //StaticVariables.credential = new GoogleCredential().setFromTokenResponse(resp);
        Constants.credential = new GoogleCredential.Builder().setTransport(HTTP_TRANSPORT)
                .setJsonFactory(JSON_FACTORY)
                .setClientSecrets(Constants.client_id, Constants.client_secret)
                .build()
                .setFromTokenResponse(resp);
        Constants.CREDENTIALS_CACHE.put(email, Constants.credential);
    }
```

Εικόνα 32:Δημιουργία credential με κωδικό πιστοποίησης

```
 for (Permission permission : GDrive.retrievePermissions(GDrive.serviceDriveBuilder("service"),
Constants.SHARED_FOLDER)) {
       recipients.add(permission.getEmailAddress());
 }
 try {
       Gmail.sendMessage(Gmail.serviceGmailBuilder(gmail), gmail,
Gmail.createEmail(recipients, gmail, jObject.getString("state") + "Publication Event", "Check
the new publication at " + title + "."));
       } catch (MessagingException ex) {
                   Logger.getLogger(Servlet.class.getName()).log(Level.SEVERE, null, ex);
          }
  //Κλήση μεθόδου για αποστολή ενημέρωσης

Constants.datastoreTransactions("resources", resourceId, keyMail,
convertToArrayList(jCoderTags.getJSONArray(resourceIndex)).toString().replace("[",
"").replace("]", ""), "storeList");
 //Κλήση μεθόδου για αποθήκευση ετικετών στο Datastore
```

Εικόνα 33: Αποσπάσματα κώδικα της κλάσης Servlet

- GDrive: Παρέχει μεθόδους αλληλεπίδρασης με το Google Drive API. Πιο συγκεκριμένα, χρησιμοποιείται για την άντληση και εισαγωγή μεταδεδομένων, παρακολούθηση φακέλων του αποθετηρίου (WatchFolder), άντληση διαχειριστών αποθετηρίου κ.ά. Χρησιμοποιεί στελέχη της Constants (SHARED_FOLDER_ID, CREDENTIALS_CACHE). (Εικόνα 36)

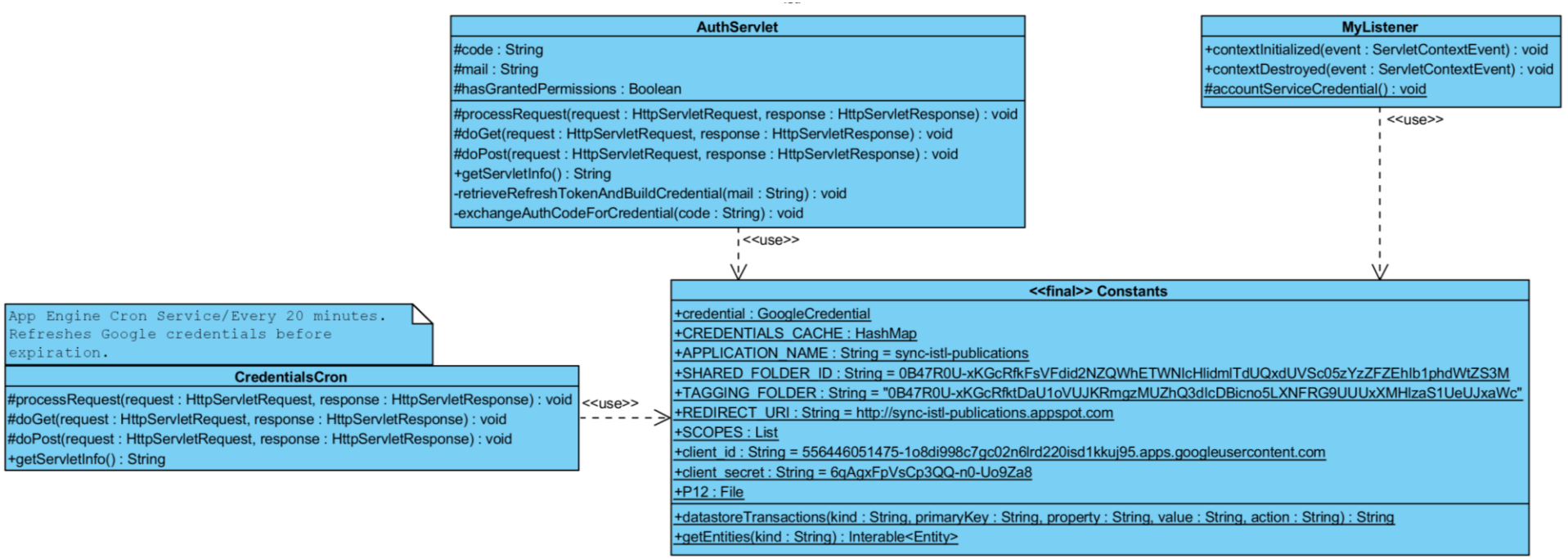
- Gmail: Παρέχει μεθόδους αλληλεπίδρασης με το Gmail API. Πιο συγκεκριμένα, χρησιμοποιείται για την αποστολή ενημερώσεων μέσω email στους εταίρους/διαχειριστές εκ μέρους του χρήστη που εισάγει νέο περιεχόμενο. (Εικόνα 36)
- SocialDatastore: Παρέχει μεθόδους αλληλεπίδρασης με το Datastore του Google App Engine. Πιο συγκεκριμένα, διαχειρίζεται τις «κοινωνικές» λειτουργίες της εφαρμογής (tagging, liking) και χρησιμοποιεί το Datastore για καταχώρηση και ανάκτηση δεδομένων. Όλες οι συνδιαλλαγές με το Datastore πραγματοποιούνται με τη μέθοδο datastoreTransactions της κλάσης Constants. (Εικόνα 36)
- WatchFolder: Αποτελεί κλάση-χειριστή για cron job. Δέχεται αίτημα κάθε μία ώρα, δηλαδή το χρονικό όριο λήξης της εγγραφής στο κανάλι παρακολούθησης. Χρησιμοποιεί τη στατική μέθοδο watchFile της GDrive για να παρακολουθεί φακέλους του αποθετηρίου στο Drive , όπου αποθηκεύονται οι ετικέτες του iSTLab Calendar, για πιθανές αλλαγές από χρήστες του calendar. (Εικόνα 37) Ο αποδέκτης των ενημερώσεων είναι το Channel servlet.
- Channel: Δέχεται τις ενημερώσεις (requests) του Drive API Server για αλλαγές στο directory που βρίσκονται αποθηκευμένες οι ετικέτες του iSTLab Calendar. Περιέχει μεθόδους που συλλέγουν και συγκρίνουν τα event ids/custom properties (custom properties των αρχείων που εισάγονται από την εφαρμογή) με τα event ids του iSTLab Calendar. Στη συνέχεια ελέγχονται για την ύπαρξη «παιδιών», οι φάκελοι των events στο Drive των οποίων τα ids δεν περιέχονται στη λίστα των event ids/custom properties από τα μεταδεδομένα των αρχείων. Επομένως, αναζητούνται αρχεία νέων events τα οποία δημιουργήθηκαν από χρήστες του calendar. Στα αρχεία αυτά εισάγονται τα «αναγνωριστικά» custom properties της εφαρμογής (Εικόνα 34). Χρησιμοποιεί στατικές μεθόδους της GDrive για ανάκτηση αρχείων και καλεί τη μέθοδο initCheckCalendar της CheckCalendar η οποία αναζητά πιθανές αλλαγές στις ετικέτες του iSTLab Calendar που αφορούν την εφαρμογή, ώστε να ενημερώσει τα custom properties στα μεταδεδομένα των αρχείων. (Εικόνα 37)

```
   protected void checkResourcesForRequiredProperties(ChildList driveResources, String title,
String eventTag, String htmlLink, String calId, String year) throws IOException {
        for (ChildReference child : driveResources.getItems()) {
            File                                    file                                    =
GDrive.serviceDriveBuilder("service").files().get(child.getId()).execute();
            if (file.getProperties() == null) {
                GDrive.insertProperty(GDrive.serviceDriveBuilder("service"),     file.getId(),
"eventTitle", title, "public");
                GDrive.insertProperty(GDrive.serviceDriveBuilder("service"),     file.getId(),
"publicationResource", "true", "public");
                GDrive.insertProperty(GDrive.serviceDriveBuilder("service"),     file.getId(),
"tagEvent", eventTag, "public");
                GDrive.insertProperty(GDrive.serviceDriveBuilder("service"),     file.getId(),
"htmlLink", htmlLink.substring(htmlLink.indexOf("=") + 1), "public");
                GDrive.insertProperty(GDrive.serviceDriveBuilder("service"),     file.getId(),
"istCalendar", calId, "public");
                GDrive.insertProperty(GDrive.serviceDriveBuilder("service"),     file.getId(),
"year", year, "public");
            }        }    }
```

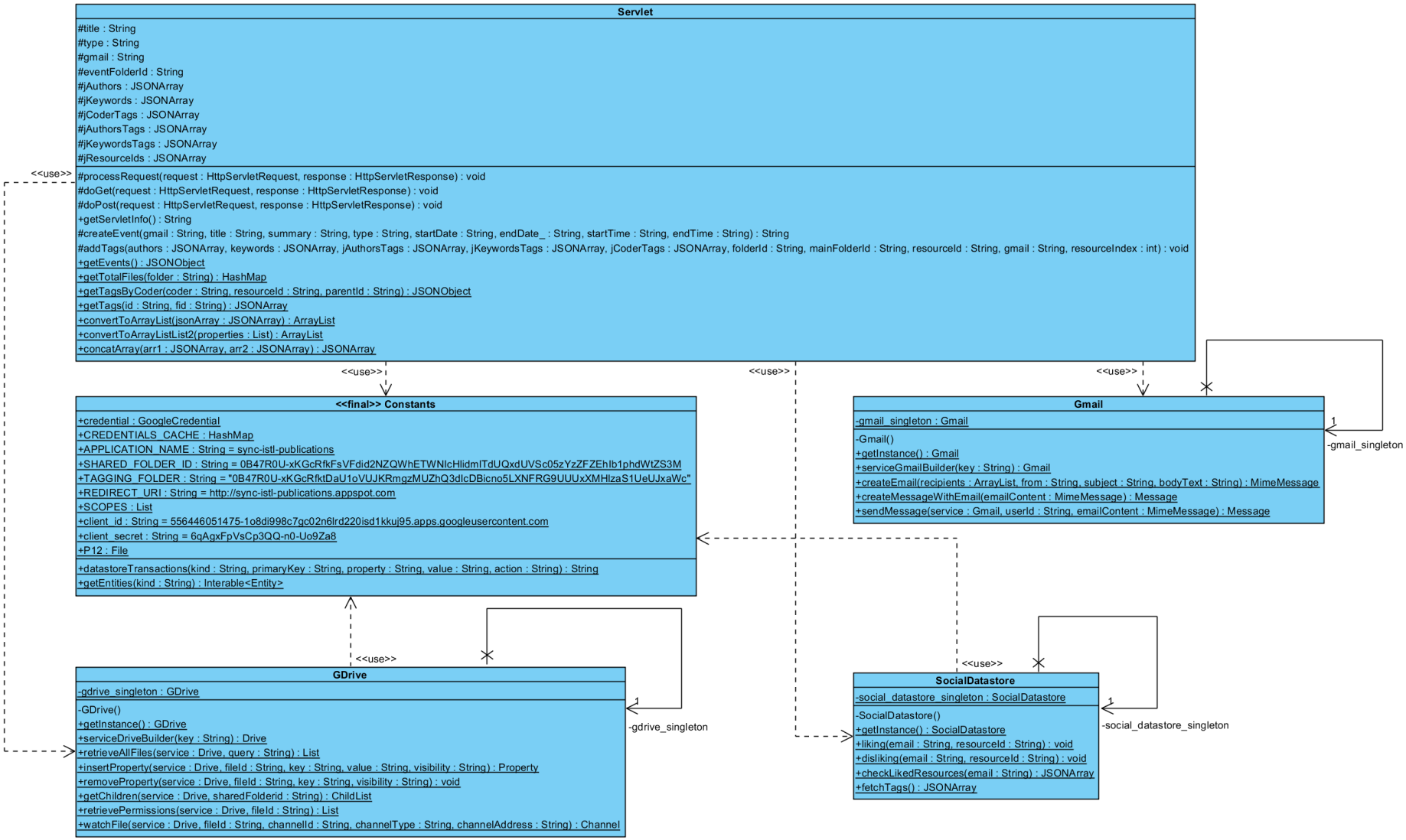
Εικόνα 34: Η μέθοδος που εισάγει custom properties στα νέα αρχεία

- CheckCalendar: Περιέχει μεθόδους για τη συλλογή και σύγκριση των μεταδεδομένων των αρχείων (custom properties) του αποθετηρίου με τις ετικέτες των αρχείων στα events του iSTLab Calendar. Με τη σύγκριση αυτή εντοπίζονται πιθανές αλλαγές (προσθήκη/διαγραφή ετικετών) από τους χρήστες του iSTLab Calendar οι οποίες εφαρμόζονται στα μεταδεδομένα των αρχείων

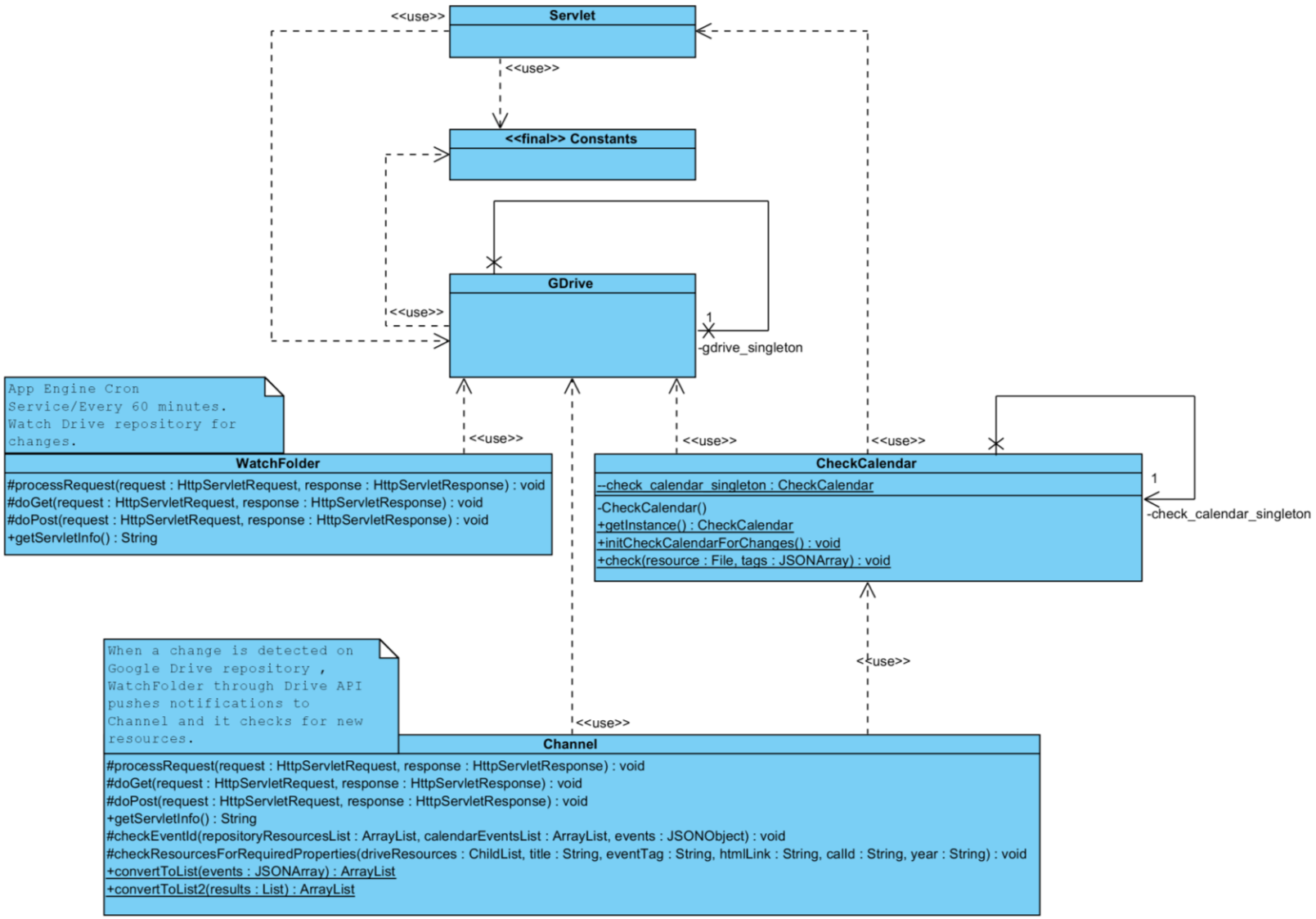
(custom properties) του αποθετηρίου. Χρησιμοποιεί βοηθητικές στατικές μεθόδους της Servlet για άντληση δεδομένων (ετικετών) από το iSTLab Calendar και μετατροπή JSON αντικειμένων σε λίστες για πιο εύκολη σύγκριση των δεδομένων. (Εικόνα 37)



Εικόνα 35 : Τμήμα του διαγράμματος κλάσεων. Οι κλάσεις AuthServlet, MyListener και CredentialsCron είναι επιφορτισμένες με τη δημιουργία και διαχείριση των Google credentials και χρησιμοποιούν στελέχη της Constants



Εικόνα 36 : Τμήμα του διαγράμματος κλάσεων. Αναπαρίστανται οι βοηθητικές κλάσεις GDrive, SocialDatastore, Gmail , η κλάση/servlet Servlet η οποία είναι το βασικό servlet της εφαρμογής, και η Constants.



Εικόνα 37: Τμήμα του διαγράμματος κλάσεων. Οι τρεις κλάσεις στο κάτω μέρος περιέχουν τη λειτουργικότητα της παρακολούθησης

Υλοποίηση του σεναρίου για το web



Εικόνα 38: Λογότυπο και κουμπί Sign-In της εφαρμογής

Στην ενότητα αυτή θα μελετήσουμε την περίπτωση της αξιοποίησης δημόσιων APIs για τη δημιουργία μίας mashup διαδικτυακής εφαρμογής η οποία οργανώνει το περιεχόμενο μιας εικονικής ομάδας (μέλη εργαστηρίου) και το αποτυπώνει σε διαφορετικές διεπαφές που σχετίζονται με την ομάδα αυτή. Πιο συγκεκριμένα, το ψηφιακό περιεχόμενο αποτυπώνεται σε ένα προηγμένο ηλεκτρονικό ημερολόγιο με βάση το event στο οποίο δημοσιεύεται και σε επαγγελματική σελίδα κοινωνικής δικτύωσης ως κοινοποίηση (post). Οι εταίροι του αποθετηρίου στο σύννεφο έχουν τη δυνατότητα προσθήκης ετικετών στο περιεχόμενο για την καλύτερη οργάνωση, αναζήτησή του.

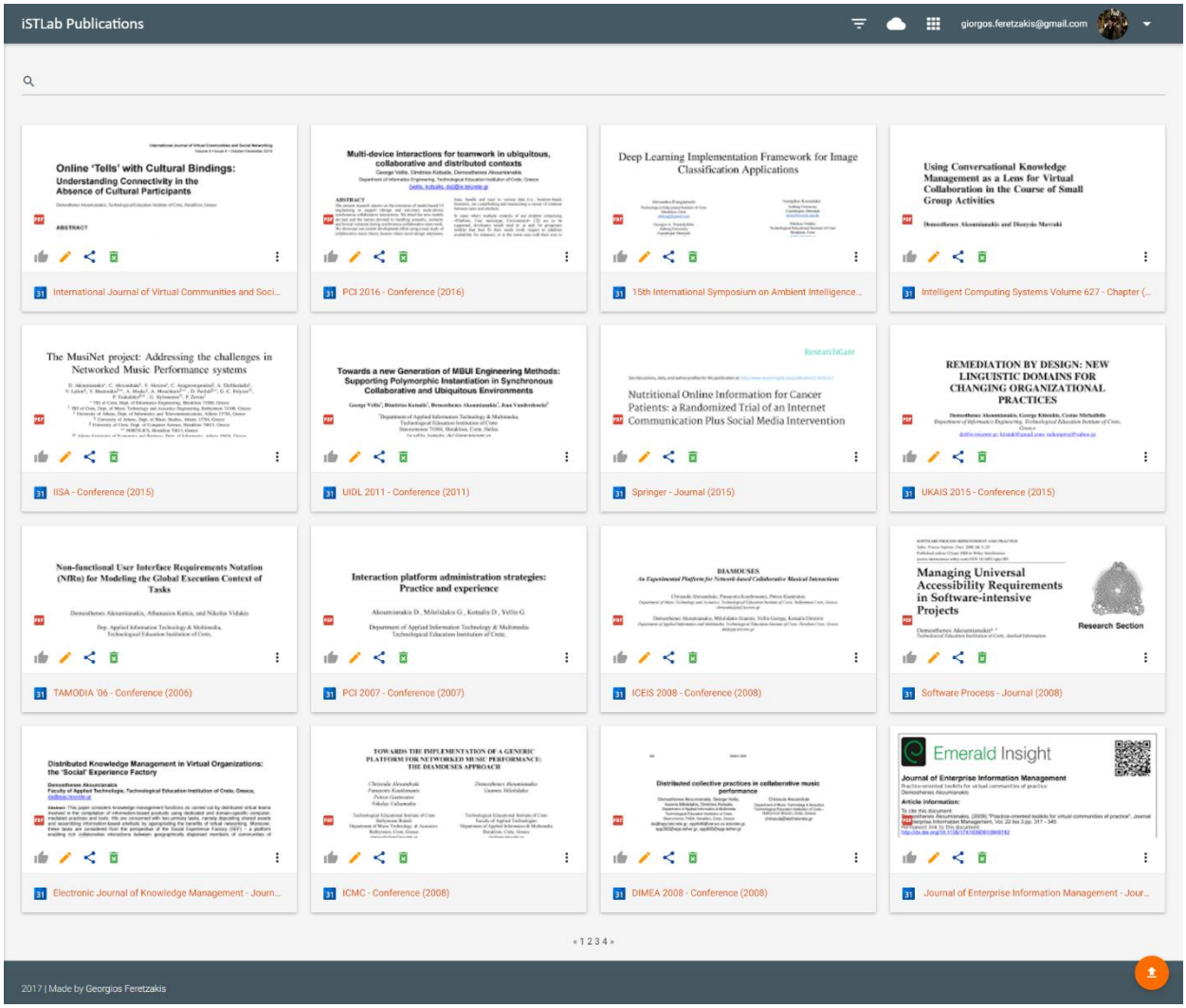
Η εφαρμογή αναπτύχθηκε με τη χρήση Java Servlet (Jetty) στο Google App Engine και ως βάση δεδομένων χρησιμοποιήθηκε το Datastore της ίδιας πλατφόρμας. Το Drive API είναι το κύριο εργαλείο που αξιοποιείται για τη συγκέντρωση, τη διαχείριση και την προσθήκη περιεχομένου αλλά και μεταδεδομένων στο περιεχόμενο. Ακόμη αξιοποιήθηκε το iSTLab Publications Calendar REST API για την άντληση δεδομένων που απαιτούνται για τη σύνθεση των ψηφιακών αντικειμένων, τον έλεγχο για νέα events και ψηφιακά έγγραφα, και τη δημιουργία νέων events που αφορούν δημοσιεύσεις. Τέλος, με το LinkedIn API γίνεται η κοινοποίηση του περιεχομένου στη σελίδα του εργαστηρίου, με το Gmail API η αποστολή ενημέρωσης στους εταίρους για την προσθήκη νέας δημοσίευσης και με το Google Contacts API η ανάκτηση των επαφών του χρήστη για το διαμοιρασμό του Drive αποθετηρίου. Τα client-side εργαλεία που χρησιμοποιήθηκαν είναι τα AJAX, jQuery και Materialize.

4.1 Επισκόπηση περιεχομένου

Με την επιτυχία της εισόδου του χρήστη με τον Google λογαριασμό του, η σκυτάλη παραδίδεται (client-side) στη συνάρτηση η οποία πραγματοποιεί την πρώτη κλήση στο Drive API (API call). Η κλήση γίνεται μέσω της μεθόδου list στην οποία ορίζουμε τις κατάλληλες παραμέτρους για να επιστρέψει τη λίστα με τα αρχεία των δημοσιεύσεων βάσει κριτηρίων ταξινόμησης (by folder, by creationDate desc). Τα αρχεία αυτά, βρίσκονται σε διαφορετικούς υπο-φακέλους του κύριου φακέλου “iSTLab Publications”

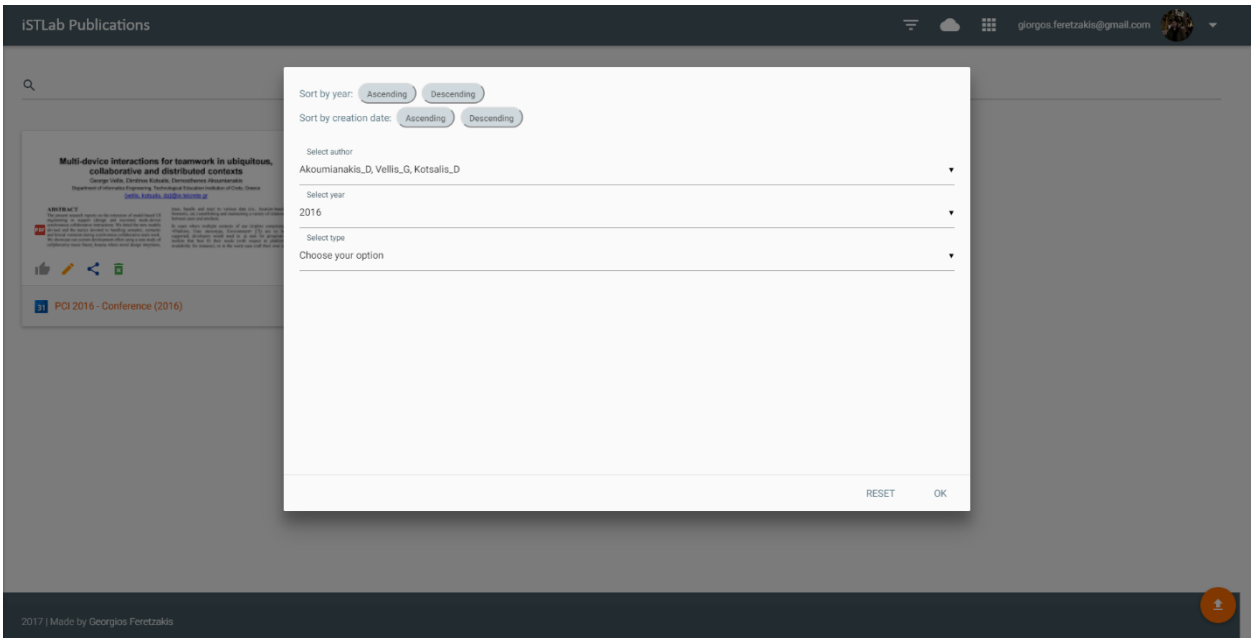
του Google Drive, ανά event και πρέπει να έχουν ως custom property το "publicationResource": "true" για να εντοπίζονται και να εμφανίζονται από την εφαρμογή. Το συγκεκριμένο custom property εισάγεται μαζί με το περιεχόμενο από την εφαρμογή και για αυτό το λόγο, η δημιουργία ενός event δημοσίευσης με προσθήκη αρχείων στον calendar γίνεται αντιληπτή στην εφαρμογή μέσω της παρακολούθησης του φακέλου του συστήματος tagging. Εναλλακτικά, γνωρίζοντας το id του root φακέλου του αποθετηρίου στο Drive, τα αρχεία θα μπορούσαν να αναζητούνται αναδρομικά (τα «παιδιά των παιδιών των παιδιών» του φακέλου) αλλά ο χρόνος απόκρισης θα ήταν απογοητευτικός για το χρήστη.

Κάθε drive αρχείο δημοσίευσης που μεταφορτώνεται, εμπλουτίζεται από την εφαρμογή με περαιτέρω μεταδεδομένα που αφορούν το event (title, date, id, type, status, description), τους φάκελους-γονείς που βρίσκονται και τις ετικέτες με τους συγγραφείς και τις λέξεις-κλειδιά των δημοσιεύσεων. Όλα αυτά τα μεταδεδομένα ενσωματώνονται ως data-attributes στην ιστοσελίδα της εφαρμογής και αξιοποιούνται στην οργάνωση, την αναζήτηση και το φιλτράρισμα του περιεχομένου. Οι δημοσιεύσεις αναπαρίστανται γραφικά με κάρτες Materialize (cards) , όπως στην παρακάτω εικόνα (Εικόνα 39), οι οποίες παρέχουν πολλές πληροφορίες (data-attributes) και δυνατότητα ανάγνωσης/επισκόπησης του αρχείου, δήλωσης "Like", επεξεργασίας ετικετών, κοινοποίησης και προσθήκης/αφαίρεσης αρχείων της δημοσίευσης.

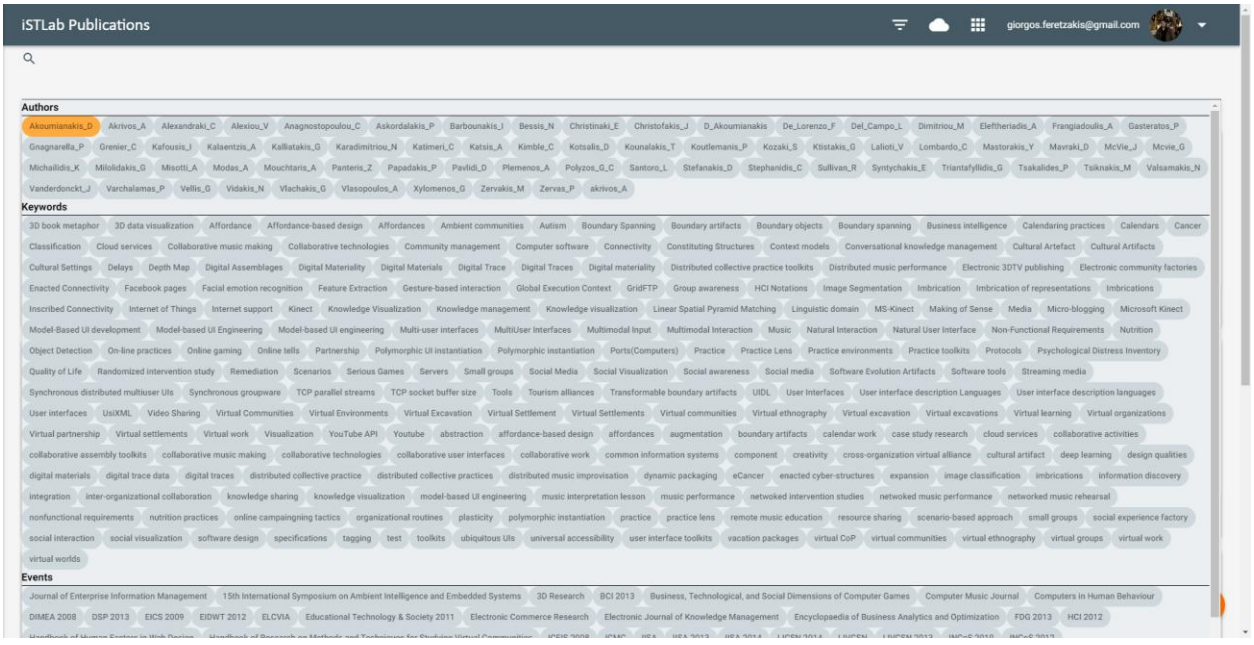


Εικόνα 39: Περιεχόμενο σε κάρτες (card-material components)

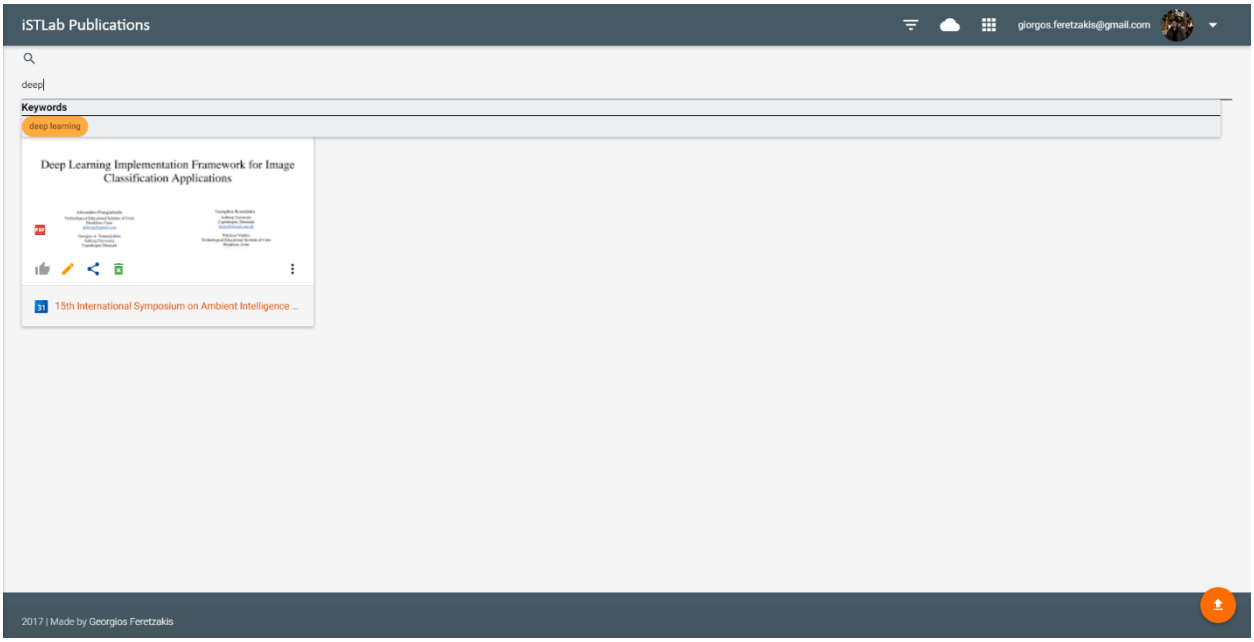
Όπως βλέπουμε παρακάτω (Εικόνα 40), ο χρήστης μπορεί να ταξινομήσει το περιεχόμενο και να το φιλτράρει, επιλέγοντας όσους συγγραφείς επιθυμεί, το έτος, και το είδος του event όπου δημοσιεύτηκε το paper. Επίσης, μπορεί να αναζητήσει περιεχόμενο στο σχετικό πλαίσιο αναζήτησης όπου υποστηρίζεται αυτόματη συμπλήρωση (autocomplete) για την πρόταση κλειδιών αναζήτησης. (Εικόνα 41,Εικόνα 42)



Εικόνα 40: Ταξινόμηση και φιλτράρισμα περιεχομένου



Εικόνα 41: Αυτόματη συμπλήρωση στο πλαίσιο αναζήτησης



Εικόνα 42: Αυτόματη συμπλήρωση στο πλαίσιο αναζήτησης με πληκτρολόγηση του χρήστη

4.2 Εισαγωγή νέου περιεχομένου

Ο χρήστης πατώντας το floating κουμπί πρέπει να συμπληρώσει τη φόρμα για την καταχώρηση νέας δημοσίευσης. Η φόρμα αποτελείται από τρεις καρτέλες (tabs). Η πρώτη ("EVENT") (Εικόνα 45) περιέχει πεδία συμπλήρωσης για τις πληροφορίες του event στο calendar. Πιο συγκεκριμένα περιέχει τα πεδία:

- Title (τίτλος)
- Start Date – Time (ημερομηνία και ώρα έναρξης). Ορίζονται μέσω pickers.
- End Date – Time (ημερομηνία και ώρα τέλους). Ορίζονται μέσω pickers.
- Type (είδος)
- Notification (ενημέρωση). Ορίζεται μέσω on/off κουμπιού.

Τα πεδία ελέγχονται για την εγκυρότητά τους (validation) με τη χρήση της μεθόδου jQuery validate() αλλά και custom jQuery μεθόδων που γράφτηκαν και προστέθηκαν (με jQuery.validator.addMethod()).

Στην καρτέλα "RESOURCES" (Εικόνα 46) ο χρήστης συμπληρώνει την περιγραφή (description) και με το κουμπί "FILE" διαλέγει το αρχείο που θα τεκμηριώνει το event από τον τοπικό χώρο αποθήκευσης (local directory). Ακόμη έχει τη δυνατότητα προσθήκης και άλλων αρχείων και των περιγραφών τους.

Στην καρτέλα "TAGS" (Εικόνα 47) ο χρήστης εισάγει τους συγγραφείς και τις λέξεις κλειδιά του paper ως μεταδεδομένα στο αρχείο. Για κάθε αρχείο που επιλέχτηκε για μεταφόρτωση δημιουργείται ένα ξεχωριστό τμήμα με πεδία. Τέλος, μπορούν να επισημανθούν με ετικέτες για περαιτέρω χαρακτηρισμό από τους εταίρους.

Τα δεδομένα της φόρμας δομούνται σε ένα JSON (Εικόνα 43) το οποίο αποστέλλεται στο Servlet. Αρχικά, δημιουργείται το event, στη συνέχεια ανεβαίνουν τα -σχετικά με το event- αρχεία στο Drive και τέλος γίνεται η προσθήκη μεταδεδομένων/ετικετών και η αποστολή ενημέρωσης.

```
      "resourcesNumber": 2,
      "title": "Title",
      "description": ["First Resource Description", "Second Resource
Description],
      "sDate": "2017-12-01",
      "eDate": "2017-12-01",
      "sTime": "10:00",
      "eTime": "12:00",
      "authors": [
            [{
                  "body": "author:First_Author",
                  "source": "event"
            }, {
                  "body": "author:Second_Author",
                  "source": "event"
            }],
            [{
                  "body": "author:First_Author1",
                  "source": "event"
            }, {
                  "body": "author:Second_Author1",
                  "source": "event"
            }]
      ],
      "keys": [
            [{
                  "body": "first keyword",
                  "source": "event"
            }, {
                  "body": "second keyword",
                  "source": "event"
            }],
            [{
                  "body": "first keyword1",
                  "source": "event"
            }, {
                  "body": "second keyword1",
                  "source": "event"
            }]
      ],
      "authorsTags": [
            ["First_Author", "Second_Author"],
            ["First_Author1", "Second_Author1"]
      ],
      "keywordsTags": [
            ["first keyword", "second keyword"],
            ["first keyword1", "second keyword1"]
      ],
      "tags": [
            ["first tag on Datastore", "second tag on Datastore"],
            ["first tag on Datastore1", "second tag on Datastore1"]
      ],
      "type": "Conference",
      "filename": ["thesis.docx", "Presentation.ppt"]
      "mime":                        ["application/vnd.openxmlformats-
officedocument.wordprocessingml.document",       "application/vnd.ms-
powerpoint"],
      "notification": true,
      "state": "Completed",
      "eventId": "aiul7od9dmukb0v15588n6agu8",
      "htmlLink":
"https://www.google.com/calendar/event?eid=YWl1bDdvZDlkbXVrYjB2MTU1OD
huNmFndTggdTZiajJsOGV0aWhxbWdyMnZocDFzbWFlaThAZw",
      "mainFolderId": "1r8M4HtG5HEOoqaqzClkiA8M1J3Dqb0V2",
      "eventFolderId": "16-Dl9MrliRJUmGWfrR--usUh58UGTDLG"
}
```

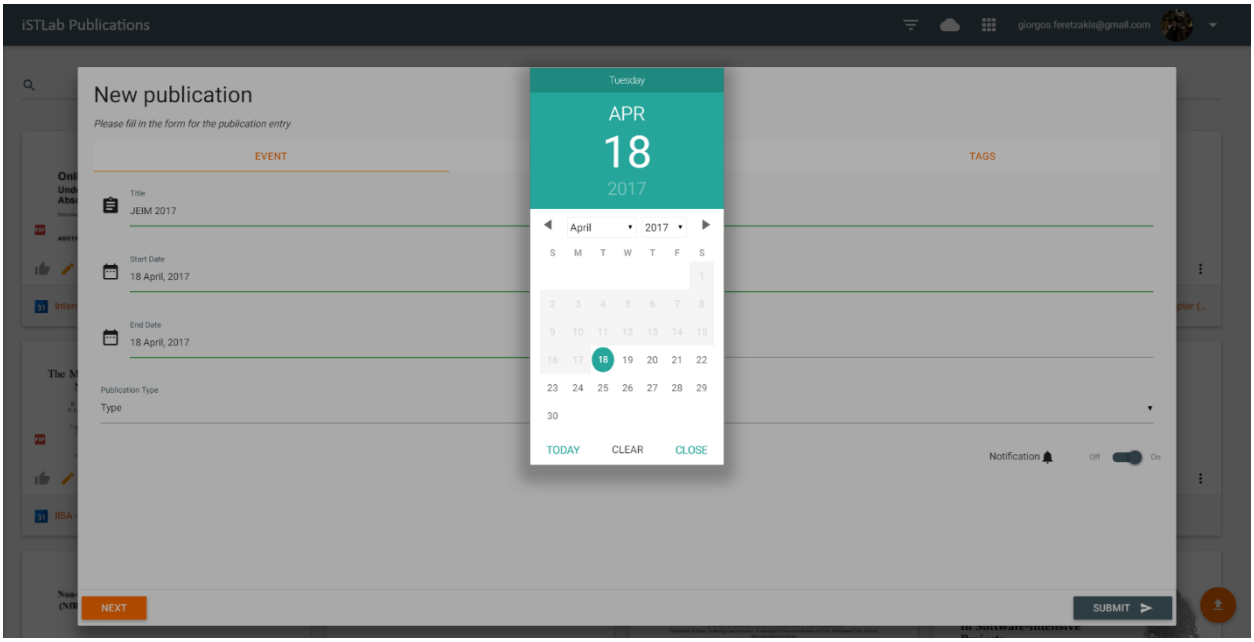

Εικόνα 43: Δομή JSON νέας δημοσίευσης

Μετά το πέρας της διαδικασίας της εισαγωγής νέου περιεχομένου , το "container" component του DOM ανανεώνει τις κάρτες, προσθέτοντας την νέα/τις νέες, και τις δομές δεδομένων από τις οποίες ενημερώνονται τα στοιχεία των χαρακτηριστικών αναζήτησης και φιλτραρίσματος της εφαρμογής (autocomplete, tag clouds, πεδία select-dropdown). (Εικόνα 48)

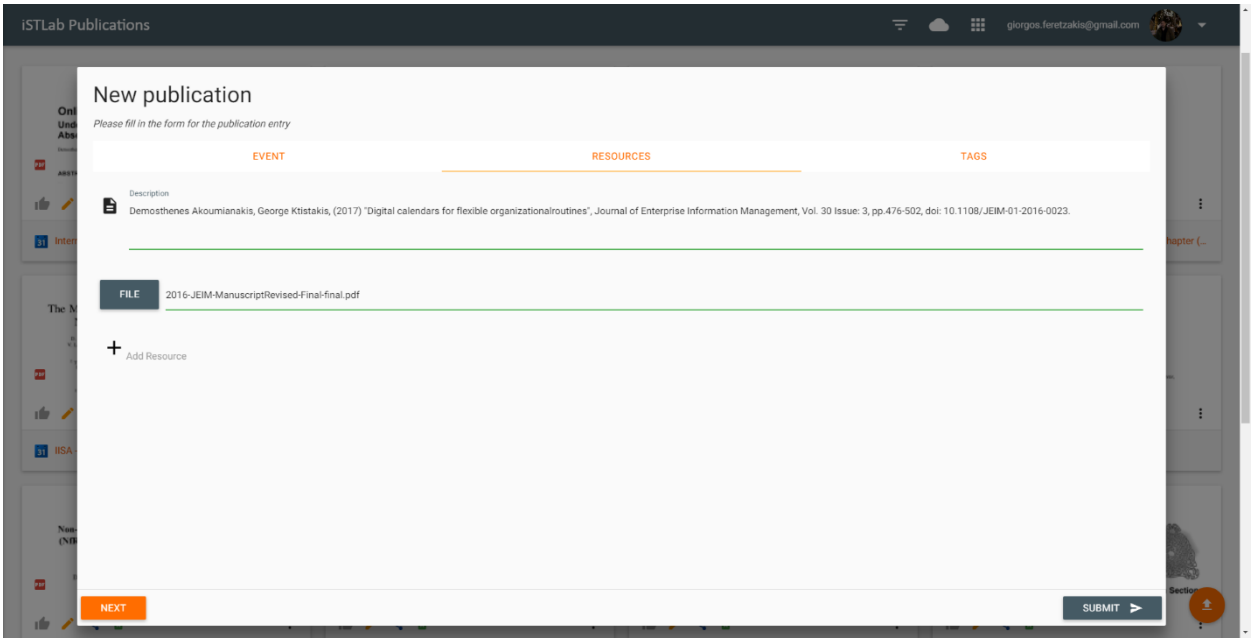
Παρόμοια διαδικασία ακολουθείται και για την εφαρμογή αλλαγών στις ετικέτες και τα μεταδεδομένα ή/και την προσθήκη νέου περιεχομένου σε ένα υπάρχον publication event. Τα δεδομένα που χαρακτηρίζουν το αρχείο δομούνται σε ένα JSON με τον παρακάτω τρόπο. (Εικόνα 44)

```
{ //"Deleting" internal arrays contain data to be deleted. These data are
removed from custom properties (Drive files metadata).
      "deleting": [
            ["Third_Author "],
            ["first keyword", "second keyword"]
      ],
      "tags": [                    //New list of tags to store on Datastore
(composed of one existed and two new)
            ["first tag on Datastore", "first new tag", "second new tag",]
      ],
      "authors": [{      //Tags to store on calendar.
            "body": "author:First_Author",
            "source": "event"
      }, {
            "body": "author:Second_Author",
            "source": "event"
      }, {
            "body": "author:Fourth_Author",
            "source": "event"
      }],
      "keys": [{    //Tags to store on calendar. One new keyword (tag) is
going to be stored.
            "body": "first new keyword",
            "source": "event"
      }],
      "keywordsTags": [    //Custom properties to be inserted
            ["first new keyword"]
      ],
      "authorsTags": [       //Custom properties to be inserted
            ["First_Author", "Second_Author", "Fourth_Author"]
      ]
      "eventId": "d5pmkdsdjv0758itqkahu5ao7g",
      "resourceIds": ["0B3RrZ8IPJtrpQmxmd0hYUjRKeEE"],
      "type": "Conference",
      "eventFolderId": "0B47R0U-xKGcRRXVxUGNBSnpidTQ"
}
```

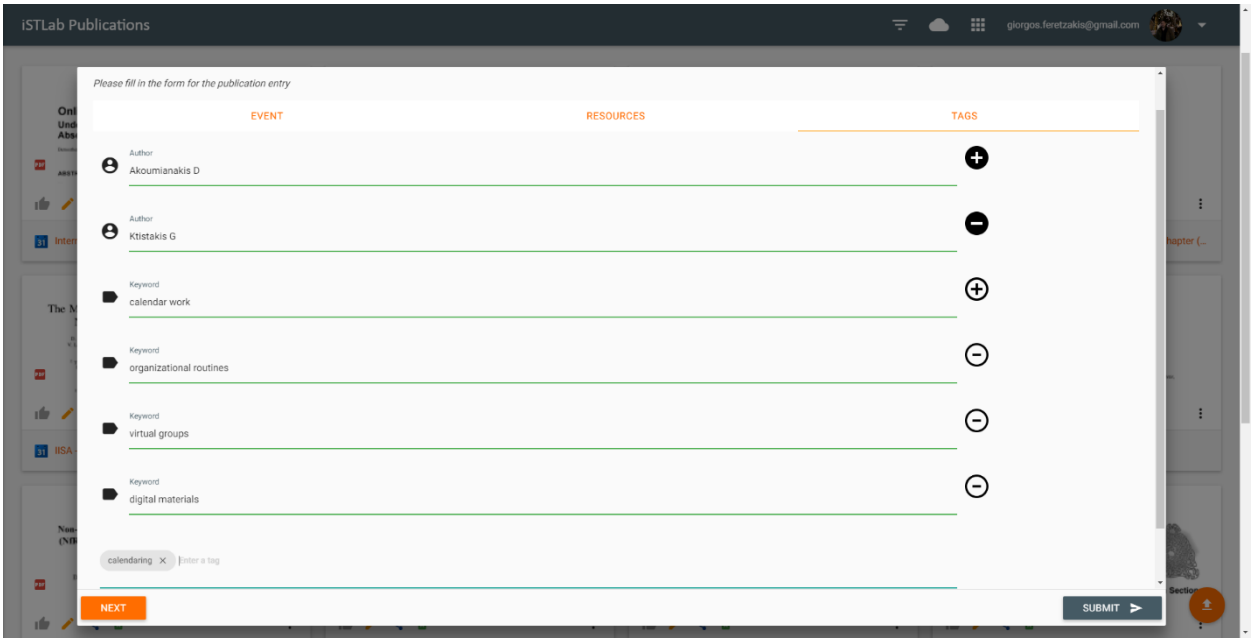
Εικόνα 44: Δομή JSON ενημέρωσης δημοσίευσης



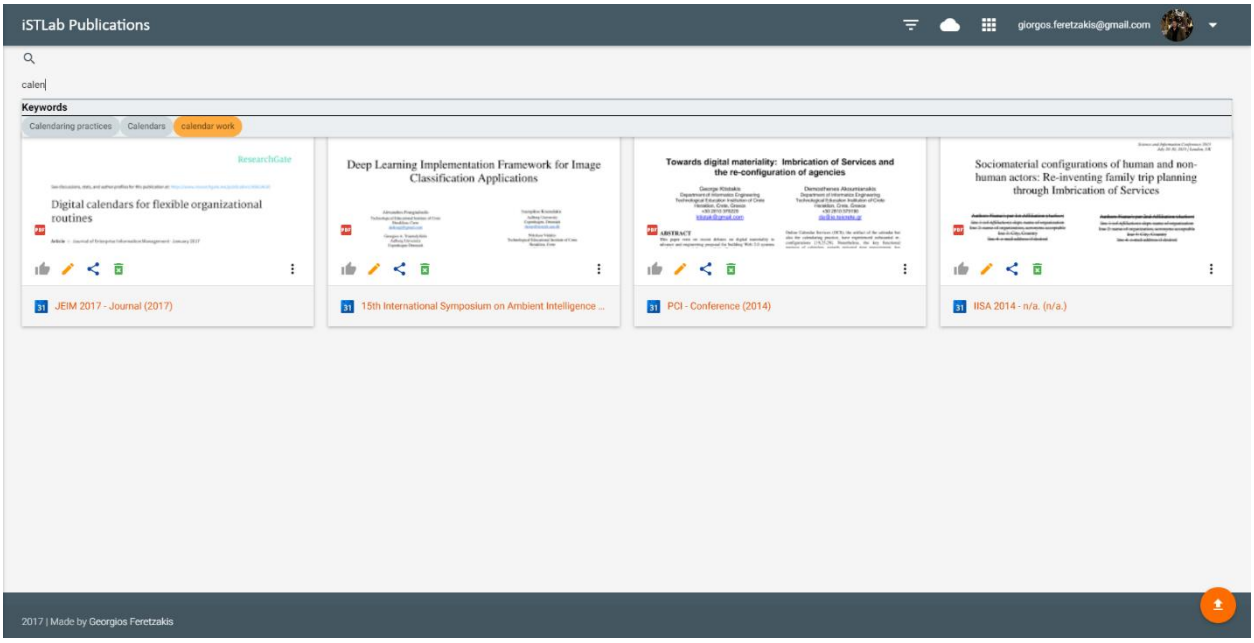
Εικόνα 45: Καρτέλα EVENT με τα πεδία δεδομένων του event



Εικόνα 46: Καρτέλα RESOURCES με τα πεδία δεδομένων του αρχείου προς μεταφόρτωση



Εικόνα 47: Καρτέλα TAGS με τα πεδία δεδομένων που χαρακτηρίζουν το αρχείο.



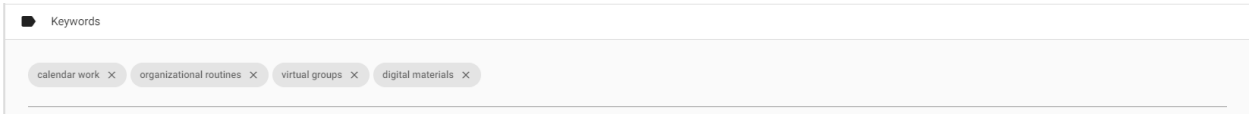
Εικόνα 48: Το autocomplete πεδίο αναζήτησης ενημερώθηκε με τα μεταδεδομένα που εισαγάγαμε στο νέο αρχείο

Στην παρακάτω εικόνα (Εικόνα 49) παρατηρούμε τα μεταδεδομένα (συγγραφείς) που καταχωρήσαμε κατά την εισαγωγή του νέου περιεχομένου.



Εικόνα 49: Στο modal επεξεργασίας της δημοσίευσης μπορούμε να δούμε (από όλους) και να επεξεργαστούμε (από εμάς) τα μεταδεδομένα που έχουν καταχωρηθεί.

Στην παρακάτω εικόνα (Εικόνα 50) παρατηρούμε τα μεταδεδομένα (λέξεις-κλειδιά) που καταχωρήσαμε κατά την εισαγωγή του νέου περιεχομένου.

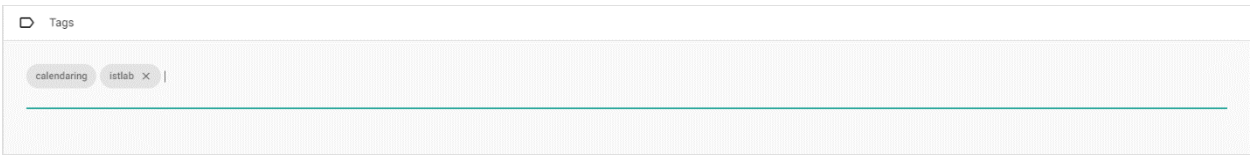


Εικόνα 50: Στο modal επεξεργασίας της δημοσίευσης μπορούμε να δούμε (από όλους) και να επεξεργαστούμε (από εμάς) τα μεταδεδομένα που έχουν καταχωρηθεί.

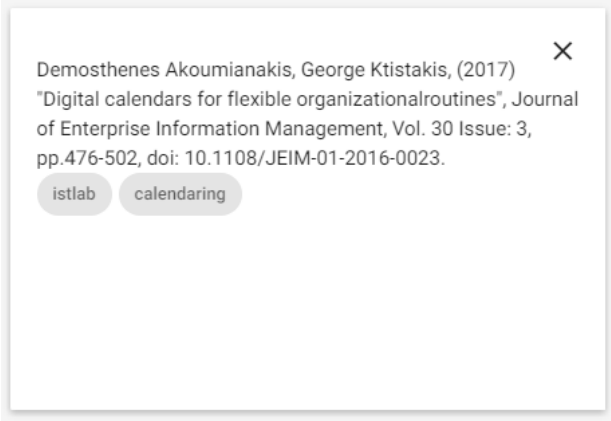
Στην παρακάτω εικόνα (Εικόνα 51) παρατηρούμε την ετικέτα που καταχωρήσαμε κατά την εισαγωγή του νέου περιεχομένου, οι οποίες κρατούνται στο Datastore της εφαρμογής. Στην επόμενη εικόνα (Εικόνα 52, Εικόνα 53) παρατηρούμε την ετικέτα που καταχωρήσαμε κατά την εισαγωγή του νέου περιεχομένου, και την ετικέτα που καταχωρήσαμε με τον δεύτερο Google λογαριασμό κατά την επεξεργασία του περιεχομένου, στο Datastore της εφαρμογής.



Εικόνα 51: Στο modal επεξεργασίας της δημοσίευσης μπορούμε να δούμε (από όλους) και να επεξεργαστούμε (από εμάς) τις ετικέτες που έχουν καταχωρηθεί.

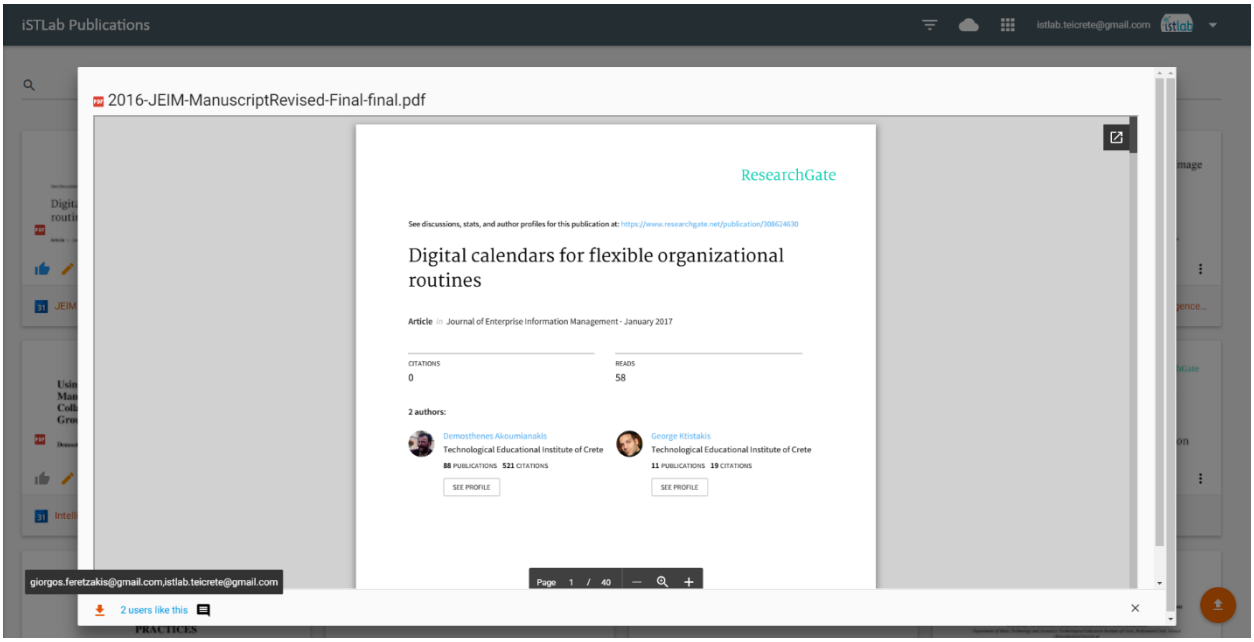


Εικόνα 52: Από το λογαριασμό Google του εργαστηρίου προστέθηκε η ετικέτα 'istlab'. Η ετικέτα 'calendaring' μπορεί να αφαιρεθεί μόνο από τον ιδιοκτήτη-tagger της.



Εικόνα 53: Το σύνολο των ετικετών της δημοσίευσης

Στις παρακάτω εικόνες (Εικόνα 54, Εικόνα 55) παρατηρούμε την επισκόπηση του νέου περιεχομένου, το σύνολο των likes που έχει συγκεντρώσει και την μπλε ένδειξη αρέσκειας.



Εικόνα 54: Επισκόπηση νέου περιεχομένου και likes.



Εικόνα 55: Δήλωση like

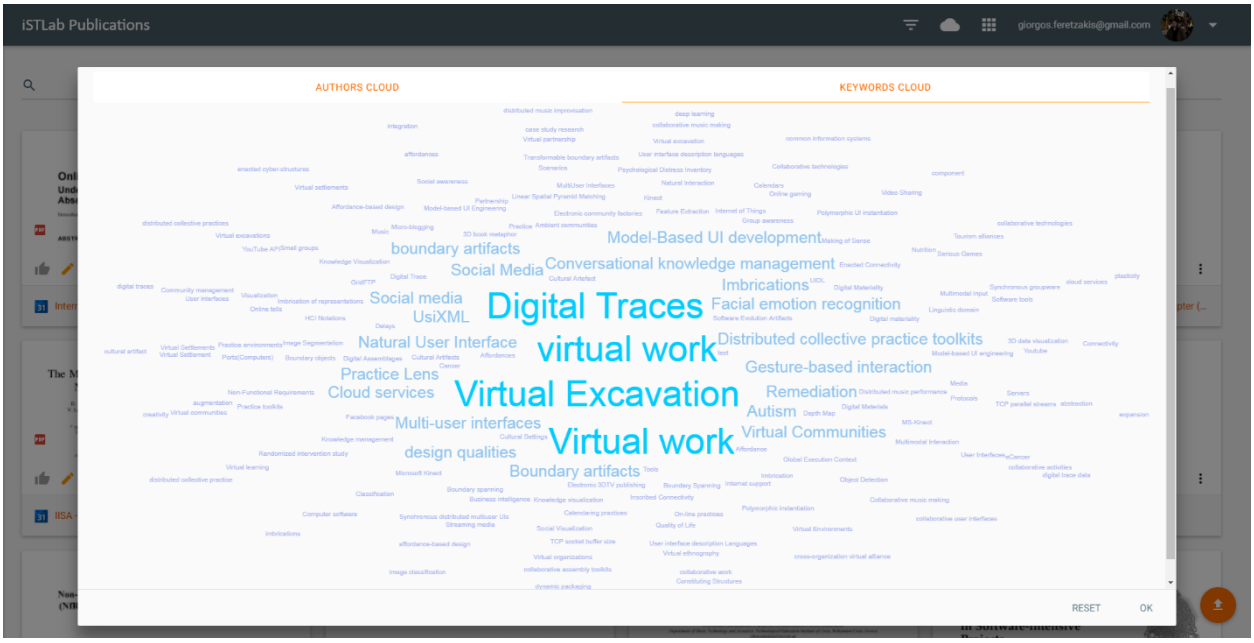
Τα likes του αρχείου που αφορά τη δημοσίευση αποθηκεύονται στο Datastore με την ακόλουθη δομή (Εικόνα 56)

name=1m-RGfb7MRJWjDk252daUqxZIwpDWFGaQ	["calendaring"]	["istlab"]	["giorgos.feretzakis@gmail.com","istlab.teicrete@gmail.com"]

Εικόνα 56: Στη λίστα resources του Datastore παρατηρούμε το entity του αρχείου με το id του (κλειδί) και τα properties του, τα οποία είναι λίστες με tags από τον κάθε χρήστη και μία λίστα με όλους τους χρήστες στους οποίους αρέσει η δημοσίευση

4.3 Γραφική αναπαράσταση

Κατά την ανανέωση του περιεχομένου και την αρχική φόρτωση της σελίδας δημιουργούνται δύο σύννεφα ετικετών με τα μεταδεδομένα που αντλούνται από τα αρχεία. Το πρώτο αποτελείται από τους authors (Εικόνα 58) και το δεύτερο από keywords (Εικόνα 57).

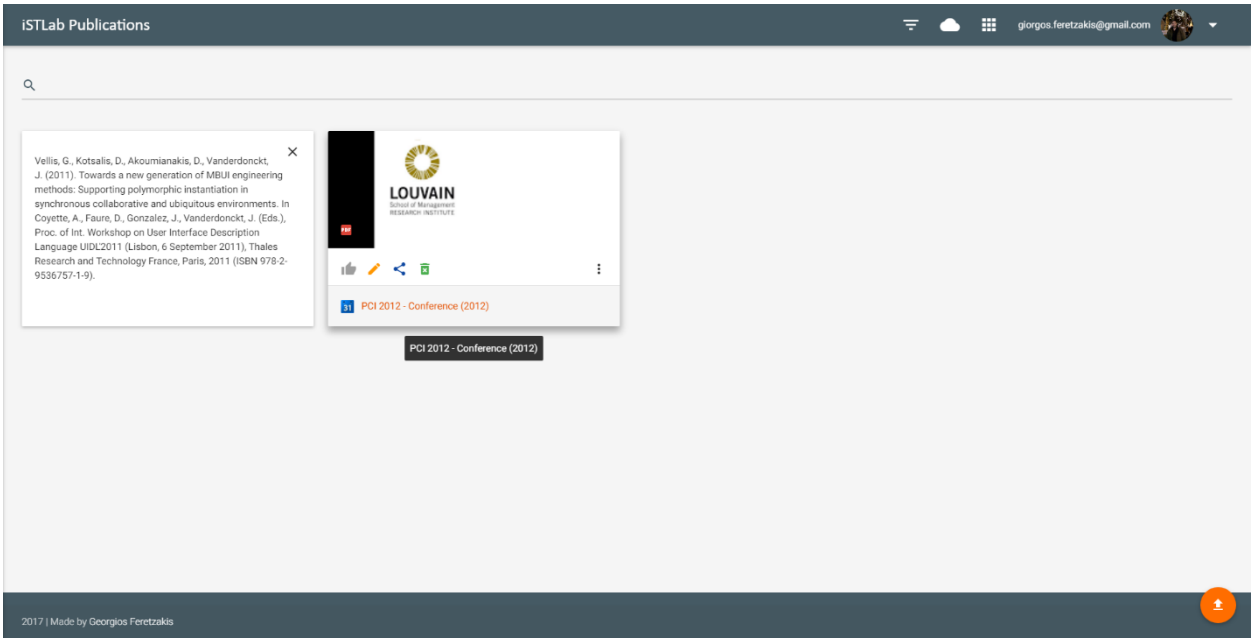


Εικόνα 57: Σύννεφο ετικετών (tag cloud) με ετικέτες από τα μεταδεδομένα των αρχείων των δημοσιεύσεων (λέξεις-κλειδιά)



Εικόνα 58: Σύννεφο ετικετών (tag cloud) με ετικέτες από τα μεταδεδομένα των αρχείων των δημοσιεύσεων (ονόματα συγγραφέων)

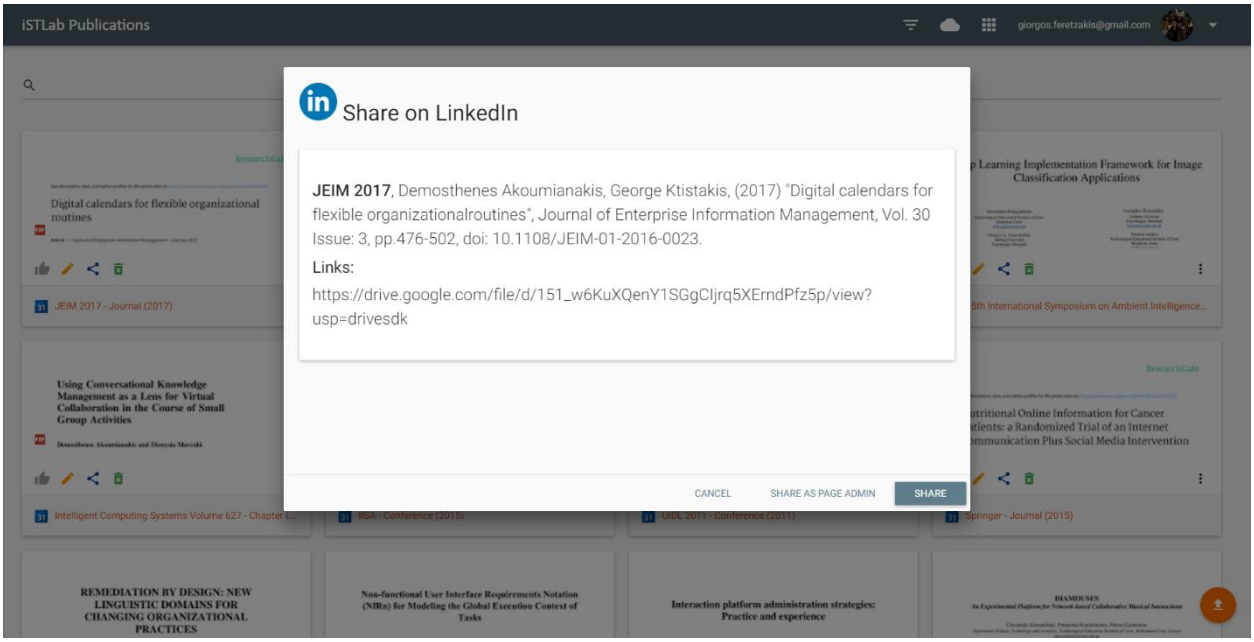
Το περιεχόμενο μπορεί να φιλτραριστεί με συνδυασμό ετικετών και από τα δύο σύννεφα. Το αποτέλεσμα του φιλτραρίσματος παρουσιάζεται στην Εικόνα 59.



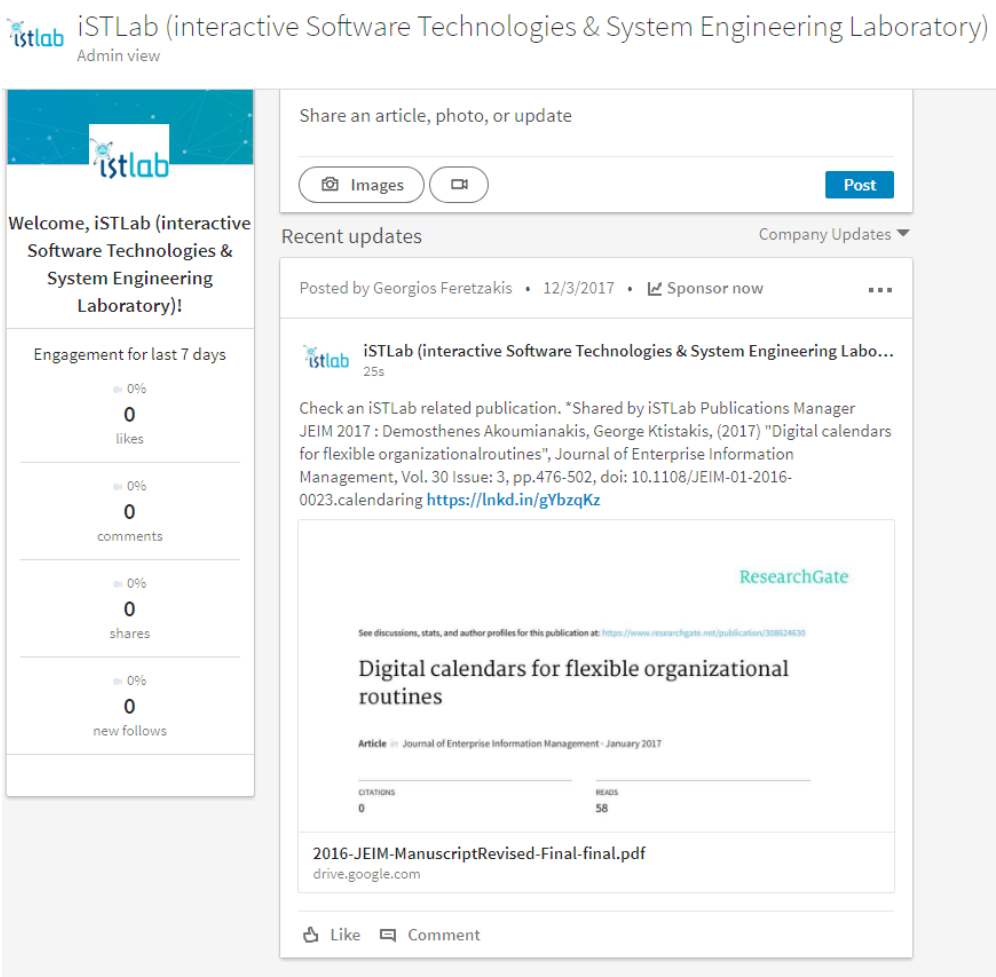
Εικόνα 59: Το αποτέλεσμα του φιλτραρίσματος με τα tag clouds

4.4 Κοινοποίηση περιεχομένου

Το περιεχόμενο μπορεί να κοινοποιηθεί από τους χρήστες στη σελίδα επαγγελματικής κοινωνικής δικτύωσης LinkedIn. Οι χρήστες που έχουν οριστεί διαχειριστές της σελίδας του εργαστηρίου στο LinkedIn, έχουν τη δυνατότητα κοινοποίησης σε αυτήν. Στις παρακάτω εικόνες (Εικόνα 60, Εικόνα 61) παρατηρούμε τα στιγμιότυπα της λειτουργίας της κοινοποίησης του περιεχομένου.



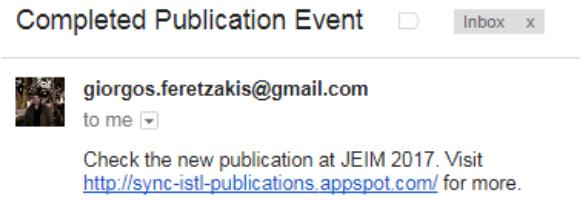
Εικόνα 60: Κοινοποίηση στο LinkedIn



Εικόνα 61: Το company update που δημιουργήθηκε στο LinkedIn page του εργαστηρίου

4.5 Μηχανισμός ενημερώσεων (email)

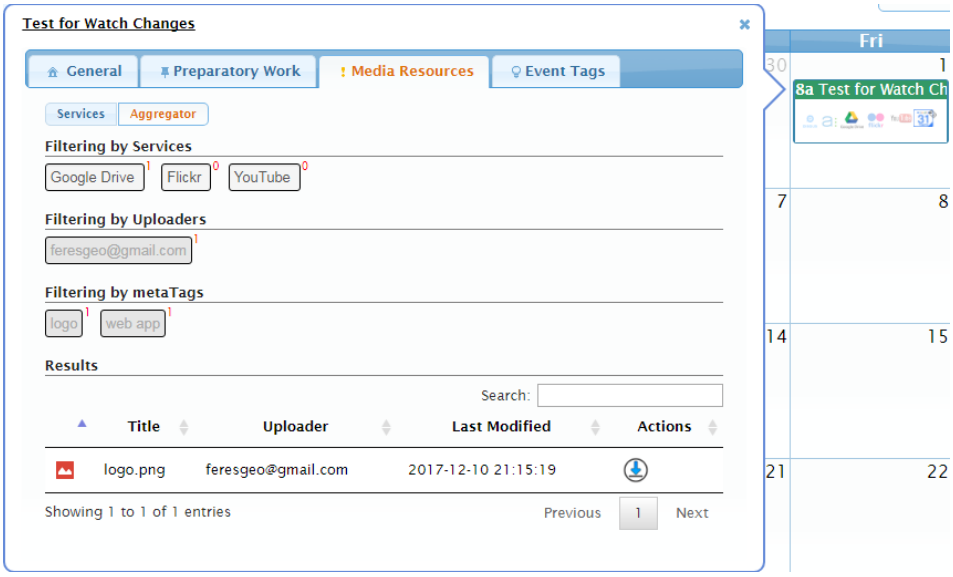
Ο χρήστης ενημερώνει με email μέσω του συστήματος τους εταίρους για τη νέα δημοσίευση, εφόσον το έχει επιλέξει στη φόρμα καταχώρησης της δημοσίευσης. Η αποστολή της ενημέρωσης πραγματοποιείται με το Gmail API στο backend της εφαρμογής εκ μέρους του χρήστη, όπως μπορούμε να δούμε παρακάτω (Εικόνα 62).



Εικόνα 62: Ενημέρωση με email που στάλθηκε από την εφαρμογή στους εταίρους

4.6 Μηχανισμός παρακολούθησης στο iSTLab Calendar

Στο σύστημα ενσωματώθηκε ένα cron job το οποίο «ανανεώνει» κάθε μία ώρα την εγγραφή στα κανάλια/φακέλους για τη λήψη ενημερώσεων σχετικά με αλλαγές στα δεδομένα (ετικέτες) του iSTLab Calendar. Παρακάτω μπορούμε να δούμε ένα νέο event που δημιουργήσαμε από τη διεπαφή του calendar και το αρχείο που επισυνάφτηκε με τις ετικέτες του. (Εικόνα 63)



Εικόνα 63: Event στο iSTLab Calendar

Στο log monitor της εφαρμογής στο GAE Console βλέπουμε τμήμα της εξόδου του Channel servlet (Εικόνα 64). Συγκρίνονται τα events του calendar με τα events που υπάρχουν ως μεταδεδομένα στα αρχεία του αποθετηρίου. Αν ένα event id του Calendar δεν περιέχεται στη λίστα των event ids των αρχείων της εφαρμογής (‘DOES NOT CONTAIN “event id”’), τότε ελέγχονται τα παιδιά-αρχεία του φακέλου που έχει δημιουργηθεί για τη φιλοξενία των αρχείων του event. Εφόσον υπάρχουν αρχεία στο φάκελο, εισάγονται σε αυτά, τα μεταδεδομένα (custom properties) που απαιτούνται (“publicationResource”:”true”), για την εμφάνισή τους στο front-end της εφαρμογής.

```
i 21:16:10.238 [s~sync-istl-publications/1.406124743309997158].<stdout>: Inserted properties for: Test for Watch Changes
i 21:16:10.238 [s~sync-istl-publications/1.406124743309997158].<stdout>: NOT CONTAINS g0cuc4q5rchejfss4b5lu1vmuo
```

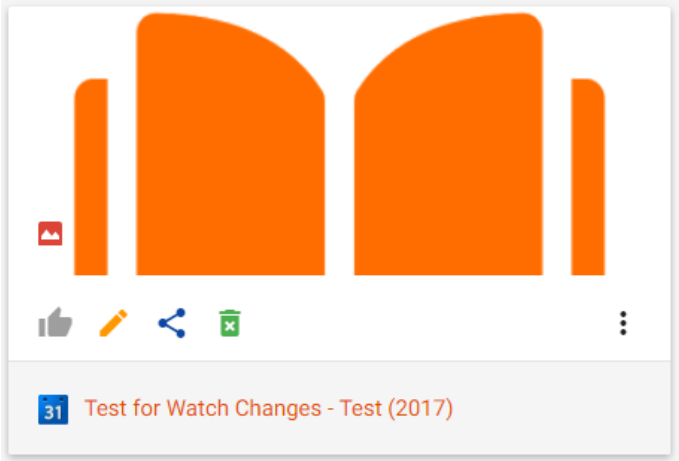
Εικόνα 64: Στιγμιότυπα του log monitor

Στο log monitor της εφαρμογής βλέπουμε τμήμα της εξόδου του Channel sevlet (Εικόνα 65) στο οποίο εντοπίζονται τα tags του αρχείου στο event του calendar, τα οποία εισάγονται στα μεταδεδομένα του αρχείου ως custom properties.

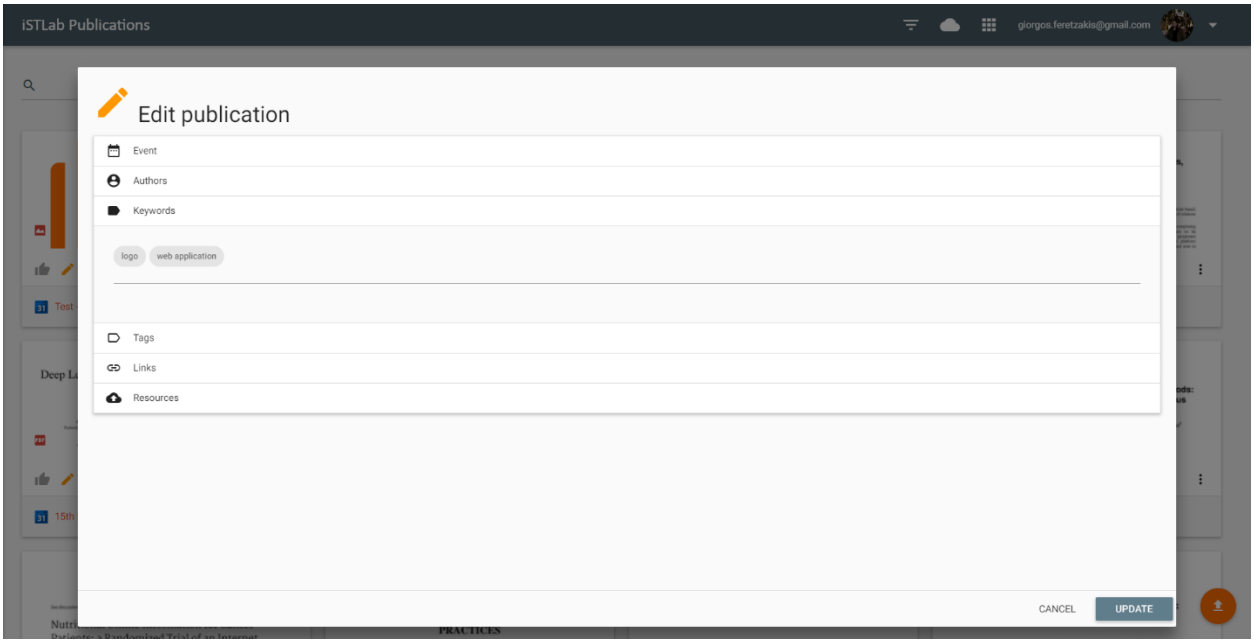
```
i 21:16:55.780 [s~sync-istl-publications/1.406124743309997158].<stdout>: Tag not included in properties: logo
i 21:16:56.151 [s~sync-istl-publications/1.406124743309997158].<stdout>: Tag not included in properties: web app
```

Εικόνα 65: Στιγμιότυπα του log monitor

Με την εισαγωγή των αναγνωριστικών μεταδεδομένων τα νέα αρχεία των νέων ή παλιών events εμφανίζονται στην ιστοσελίδα της εφαρμογής. (Εικόνα 66 & Εικόνα 67)



Εικόνα 66: Το νέο αρχείο του event



Εικόνα 67: Οι λέξεις κλειδιά του αρχείου όπως προστέθηκαν στο calendar.

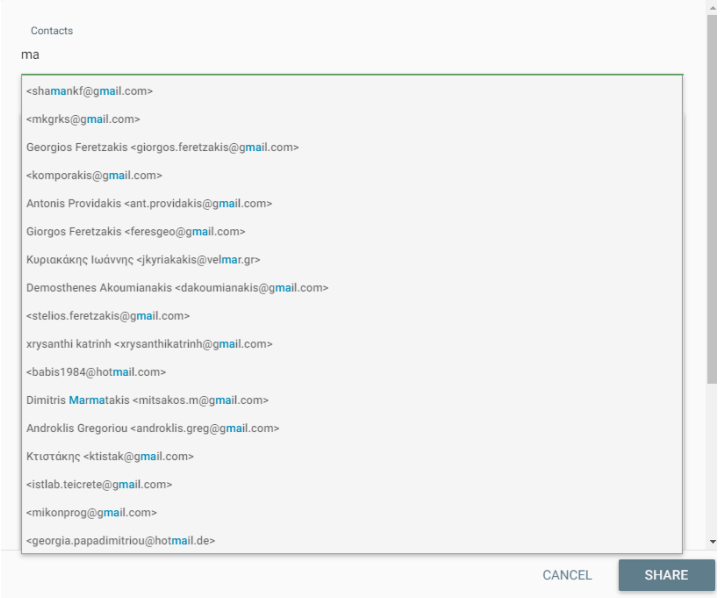
4.7 Διαμοιρασμός κοινόχρηστου αποθετηρίου

Στην πιο πρόσφατη Version της εφαρμογής υπάρχει η δυνατότητα διαμοιρασμού του αποθετηρίου με άλλους χρήστες οι οποίοι μπορούν να επιλεχθούν από τη λίστα αυτόματης συμπλήρωσης των Google επαφών τους. Παρακάτω βλέπουμε το αίτημα Ajax της εφαρμογής μας (client) στο url του API για την ανάκτηση επαφών του χρήστη. Η παράμετρος "q" του αιτήματος ορίζεται με το κείμενο στο πεδίο (input field) αναζήτησης κάθε φορά που ο χρήστης πληκτρολογεί.

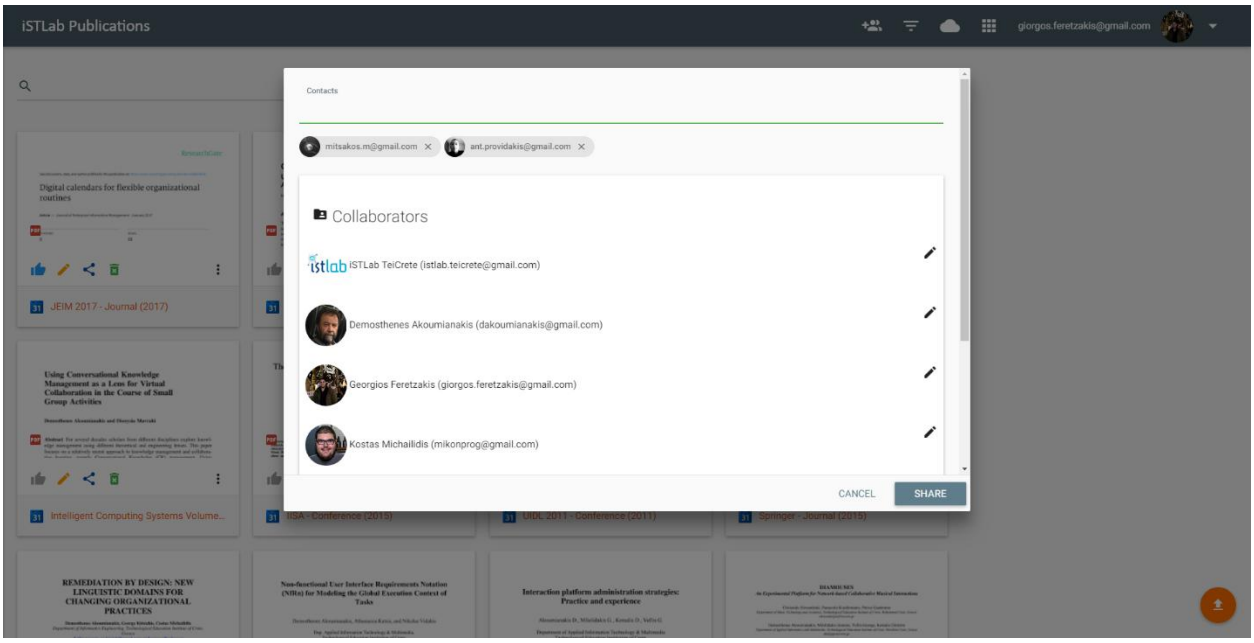
```
$.get("https://www.google.com/m8/feeds/contacts/default/full?alt=json&access_token="
+ getCookie("accessToken") + "&q=" + query + "&max-results=500&v=3.0", function
```

```
(response) {…});
```

Στις παρακάτω (Εικόνα 68 & Εικόνα 69) εικόνες παρατηρούμε το πεδίο autocomplete εύρεσης επαφών του χρήστη, τους επιλεγμένους για χορήγηση δικαιωμάτων διαχείρισης χρήστες και τους χρήστες που ήδη έχουν δικαιώματα διαχείρισης.



Εικόνα 68: Λίστα αυτόματης συμπλήρωσης των Google Contacts του χρήστη



Εικόνα 69: Το modal "Share Repository" με τις επιλεγμένες επαφές.

Κατακλείδα & Συμπεράσματα

Η παρούσα εργασία επέτρεψε στο συγγραφέα, αφενός να μελετήσει θέματα που αφορούν τη mashup προσέγγιση ανάπτυξης εφαρμογών, τα δημόσια APIs και τον διαμοιρασμό περιεχομένου στο Διαδίκτυο, και τις πρακτικές αλληλεπίδρασης των χρηστών με το περιεχόμενο στα πλαίσια ενός διαμοιραζόμενου αποθετηρίου μιας ομάδας, και αφετέρου να εξοικειωθεί με τεχνολογίες και εργαλεία της τρέχουσας τεχνολογικής στάθμισης. Κατά την εκπόνηση της εργασίας, εκτός από προγραμματιστικές δεξιότητες, αποκτήθηκαν επίσης, γνώσεις που βοήθησαν το συγγραφέα να κατανοήσει εις βάθος τη συνεργατικότητα που αναπτύσσεται σε ένα online κοινόχρηστο χώρο. Μέσα από τη συνεργασία μπορεί να παραχθεί γνώση και ενημέρωση προς όφελος των συμμετεχόντων χρηστών.

5.1 Μελλοντική εργασία και επεκτάσεις

Με τη χρήση σύγχρονων τεχνολογιών και εργαλείων Web 2.0 η εφαρμογή δίνει τη δυνατότητα στους χρήστες να εισάγουν, να προβάλουν, να χαρακτηρίσουν και να διαμοιράσουν το ψηφιακό περιεχόμενο σε άλλες διεπαφές όπως Google Mail, LinkedIn, iSTLab Calendar. Σε μία πιο εξελιγμένη εκδοχή, το σύστημα θα μπορούσε να ενσωματώνει περισσότερα χαρακτηριστικά λειτουργίας της υπηρεσίας Google Drive (Comments, Folders) και του iSTLab Calendar (Flickr, Asana, Disqus, Youtube). Με τη δυνατότητα καταχώρησης comments/replies που προσφέρει το Drive API, θα μπορούσαν να προστεθούν σχόλια πάνω σε συγκεκριμένα σημεία των κατά την προβολή των αρχείων. Επιπλέον, θα ήταν χρήσιμο να ενσωματωθεί σε κάθε κάρτα ένα κουμπί λήψης ή αντιγραφής στο clipboard της παράθεσης (citation) της κάθε δημοσίευσης. Μία ακόμη ενδιαφέρουσα επέκταση στα παραπάνω, θα ήταν η δυνατότητα ταξινόμησης αρχείων κατά σειρά βάσει του αριθμού των εμφανίσεων (occurrences) αλλά και των συνεμφανίσεων (co-occurrences) των ετικετών των αρχείων που ο χρήστης συνηθίζει να προβάλει. Σύμφωνα με το παραπάνω κριτήριο, το σύστημα θα μπορούσε να στέλνει ενημερώσεις για δραστηριότητες χρηστών σε αρχεία, που πιθανόν να ενδιαφέρουν τους χρήστες. Τέλος, το συγκεκριμένο σύστημα θα μπορούσε να χρησιμοποιηθεί και ως βοηθητικό εργαλείο από διδάσκοντες και φοιτητές σε μία τάξη για την οργάνωση της ύλης και των εργασιών του μαθήματος.

Βιβλιογραφία

1. **Wikipedia.** Wikipedia. [Ηλεκτρονικό] https://en.wikipedia.org/wiki/World_Wide_Web.

2. ***Web 2.0 and Emerging Learning Technologies/Future.***

3. Καλτσογιάννης. 2007.

4. Wikipedia. Application software. [Ηλεκτρονικό] https://en.wikipedia.org/wiki/Application_software.

5. Exforsys. [Ηλεκτρονικό] 2007. http://www.exforsys.com/tutorials/soa/soa-architecture.html.

6. Mahmood, Zaigham. ***Service Oriented Architecture: Potential Benefits and Challenges.*** **Agios Nikolaos, Crete Island, Greece : s.n., 2007.**

7. IBM Redbooks. Patterns: Service Oriented Architecture And Web Services. 2004.

8. Κωνσταντίνος Μαργαρίτης, Πανεπιστήμιο Μακεδονίας, Τμήμα Εφαρμοσμένης Πληροφορικής. Web Services και SOAP. [Ηλεκτρονικό] 2007. http://pdplab.it.uom.gr/project/soap/Theory/.

9. Chappell, David και Jewell, Tyler. ***Java Web Services.***

10. Ζωγόπουλος και Αθανίτης. Εξέλιξη των Τεχνολογιών και Υπηρεσιών του Παγκόσμιου Ιστού και Εφαρμογές στην Εκπαίδευση. ***6ο Πανελλήνιο Συνέδριο των Εκπαιδευτικών για τις ΤΠΕ Αξιοποίηση των Τεχνολογιών της.*** **2011.**

11. Rodriguez, Alex. RESTful Web services: The basics. 2008.

12. Berners-Lee. Uniform Resource Identifier (URI): Generic Syntax. 2005.

13. ***RESTful Web Services vs. "Big" Web Services: Making the Right Architectural Decision.*** **Pautasso, Cesare, Zimmermann, Olaf και Leymann, Frank . Beijing, China : s.n.**

14. Hamid R. , Motahari Nezhad, Casati, Fabio και Toumani, Farouk. Web Services Interoperability Specifications.

15. Webopedia. [Ηλεκτρονικό]

16. Oracle. Oracle Database Online Documentation . [Ηλεκτρονικό]

17. OASIS. [Ηλεκτρονικό]

18. Merriam-Webster. [Ηλεκτρονικό] https://www.merriam-webster.com/dictionary/metadata.

19. Κυριάκη-Μάνεση, Δάφνη. Μεταδεδομένα.

20. Διαμαντοπούλου. ***Μελέτη προτύπων μεταδεδομένων καταλογοποίησης πολιτιστικών και περιβαλλοντικών πηγών για την υποστήριξη της διαλειτουργικής ψηφιακής επεξεργασίας των σχετικών δεδομένων.*** **s.l. : Πανεπιστήμιο Πατρών, 2011.**

21. Webopedia. [Ηλεκτρονικό] https://www.webopedia.com/TERM/A/API.html.

22. Berlind, David. ProgrammableWeb. [Ηλεκτρονικό] 2015. https://www.programmableweb.com.

23. Software Engineering Stack Exchange. Software Engineering Stack Exchange. ***https://softwareengineering.stackexchange.com.*** **[Ηλεκτρονικό] 2011. https://softwareengineering.stackexchange.com/questions/38691/difference-between-web-api-and-web-service.**

24. ***Mashups: An Emerging Content Aggregation Web2.0 Paradigm.*** **Marjit, Ujjal και Jana, Shibsankar.**

25. ***An Online Platform for Web APIs and Service Mashups.*** **Maximilien, Michael, Ranabahu, Ajith και Gomadam, Karthik.**

26. Merrill, Duane. Mashups: The new breed of Web app. 2009.

27. Νησιώτης, Αντώνιος. ***Σχεδίαση εφαρμογής τύπου "Mashup" βασισμένη στην τεχνολογία Google maps για την απεικόνιση σημαντικών σημείων ενδιαφέροντος της.***

28. Berners-Lee, Tim, Hendler, James και Lassila, Ora. The Semantic Web. ***Scientific American.*** **2001.**

29. Staab, Steffen και Tausch, Bernhard . Semantic Web - RDF. [Ηλεκτρονικό] https://pdfs.semanticscholar.org/1352/ac4ce8b01bd35bbfd54415b7ef04830bec5f.pdf.

30. W3C. ***RDF 1.1 Primer.*** **2014.**

31. Knowledge@Wharton. [Ηλεκτρονικό] 2005. http://knowledge.wharton.upenn.edu/article/whats-the-next-big-thing-on-the-web-it-may-be-a-small-simple-thing-microformats/.

32. ***The folksonomy tag cloud: when is it useful?*** **Sinclair, James και Cardew-Hall, Michael . Department of Engineering, The Australian National University, Canberra, ACT 0200, Australia : s.n.**

33. ***The State of the Art in Tag Ontologies: A Semantic Model for Tagging and Folksonomies.*** **Kim, Hak Lae , και συν. s.l. : 2008 Proc. Int'l Conf. on Dublin Core and Metadata Applications.**

34. Mell, P. και Grance, T. The NIST definition of cloud computing. 2011.

35. ***Google Sign-In for Websites.*** **[Ηλεκτρονικό] https://developers.google.com/identity/sign-in/web/server-side-flow.**

36. Google Drive API. [Ηλεκτρονικό] https://developers.google.com/drive/v3/web/about-sdk.

37. Gmail API. [Ηλεκτρονικό] https://developers.google.com/gmail/api/guides/.

38. Google Contacts API. [Ηλεκτρονικό] https://developers.google.com/google-apps/contacts/v3/.

39. LinkedIn Developers. [Ηλεκτρονικό] https://developer.linkedin.com/docs/rest-api.

40. Prabhu, Anirudh και Shenoy, Aravind. *Introducing Materialize.*

41. Wikipedia. [Ηλεκτρονικό] https://el.wikipedia.org/wiki/Material_design.

42. Wikipedia. [Ηλεκτρονικό] https://en.wikipedia.org/wiki/JQuery.

43. Eclipse. *Eclipse Neon.* [Ηλεκτρονικό] https://help.eclipse.org/neon/index.jsp.

44. ServiceOrientation.com. [Ηλεκτρονικό] http://serviceorientation.com/serviceorientation/service_orientation_and_interoperability.

45. Wikipedia. [Ηλεκτρονικό] http://en.wikipedia.org/wiki/WS-*.

46. *Understanding Metadata.* NISO. 2004.

47. *Free Software Foundation Europe. 2015.*

48. Yee, Raymond. Pro Web 2.0 Mashups: Remixing Data and Web Services . 2008.

49. Wikipedia. Mashup (web application hybrid). [Ηλεκτρονικό] https://en.wikipedia.org/wiki/Mashup_(web_application_hybrid).