Μελέτη και Ανάπτυξη Δικτυακών Εφαρμογών Ασύγχρονης Επικοινωνίας

Ανδρέου Χαράλαμπος

bcand@phys.uoa.gr

Εθνικό και Καποδιστριακό Πανεπιστήμιο Αθηνών, Τμήματα Φυσικής - Πληροφορικής και Τηλεπικοινωνιών, Πανεπιστημιούπολη, Ιλίσια, 15784, Αθήνα, Ελλάς.

ΠΕΡΙΛΗΨΗ

Στη παρούσα διπλωματική εργασία, μελετήθηκε η δημιουργία κατανεμημένων εφαρμογών για την ασύγχρονη επικοινωνία αυτών, με σκοπό το σχεδιασμό και την υλοποίηση πληροφοριακών συστημάτων που επικοινωνούν πάνω από δίκτυα κινητής τηλεφωνίας. Για το σκοπό αυτό χρησιμοποιήθηκαν διαδικασίες εκπροσώπησης κατάστασης μεταφοράς (REST), οι οποίες ενδείκνυνται για ασταθή και αναξιόπιστα δίκτυα μεταφοράς, όπως είναι το δίκτυο των κινητών επικοινωνιών.

Λέξεις κλειδιά: Server – Client, REST, MDA, IPhone, Spring.

ΕΠΙΒΛΕΠΩΝ

Ρεΐσης Διονύσιος, Επίκουρος Καθηγητής, Τμήμα Φυσικής, Σχολή Θετικών Επιστημών, Εθνικό και Καποδιστριακό Πανεπιστήμιο Αθηνών.

1. ΕΙΣΑΓΩΓΗ

Οι απαιτήσεις επικοινωνίας στη σύγχρονη εποχή έχουν αυξηθεί τόσο από την εξέλιξη της τεχνολογίας όσο και από την ανάγκη του ανθρώπου για επικοινωνία σε όλα τα επίπεδα. Εφαρμογές κοινωνικοποίησης αλλά και εφαρμογές που βρίσκουν χρήση σε διάφορους εργασιακούς τομείς έχουν κατακλύσει τη ζωή μας με σκοπό την επικοινωνία ανθρώπου με άνθρωπο, ανθρώπου με επιχείρηση, ακόμα και πολίτη με τις δημόσιες υπηρεσίες. Δεδομένης της ανάπτυξης των δικτύων κινητής επικοινωνίας επισπεύσθηκε η ανάγκη δημιουργίας εύχρηστων, γρήγορων και αποτελεσματικών εφαρμογών για τη χρήση τους πάνω στις νέες έξυπνες συσκευές κινητής τηλεφωνίας, που συνεχώς παράγονται με μεγαλύτερους ρυθμού και με χαμηλότερο κόστος.

Για τους παραπάνω λόγους τα τελευταία χρόνια χρησιμοποιείται ευρέως η αρχιτεκτονική REST[2]. Το REST ορίζει ένα σύνολο αρχιτεκτονικών αρχών με τις οποίες μπορούν να σχεδιαστούν υπηρεσίες Web 2.0, που εστιάζουν στους πόρους ενός συστήματος, συμπεριλαμβανομένου του τρόπου με τον οποίο οι πόροι διεθυνσιοδοτούνται και μεταφέρονται μέσω HTTP από ένα ευρύ φάσμα πελατών, γραμμένοι σε διαφορετικές γλώσσες (php, javascript κ.λ.π.). Αν μετρήσουμε τον αριθμό των Web υπηρεσιών που το χρησιμοποιούν, το REST τα τελευταία χρόνια έχει αναδειχθεί ως κυρίαρχο μοντέλο των υπηρεσιών web αρχιτεκτονικής[1]. Στην πραγματικότητα, τοREST είχε τόσο μεγάλο αντίκτυπο στο διαδίκτυο ότι έχει εκτοπίσει ως επί το πλείστον το SOAP και το WSDL, διότι είναι ένα πολύ απλό στη χρήση.

Το REST ακολουθεί τέσσερις βασικές αρχές σχεδιασμού. Τη χρήση μεθόδων HTTP ρητά (GET, POST, PUT, DELETE), την απουσία κατάστασης συνόδου, εμφανίζουν δομή καταλόγου που μοιάζει με URIs και τη μεταφορά με XML, JavaScript Object Notation (JSON), ή και τα δύο.

Τόσο ο αυξανόμενος ρυθμός ανανεώσεων των εργαλείων υλοποίησης, όσο και ο πολλαπλάσια αυξανόμενος ρυθμός νέων χρηστών οδήγησε την τεχνολογία υλοποίησης εφαρμογών στην αρχιτεκτονική MDA (*Model*

Driven Architecture). Ο κύριος στόχος του MDA είναι να διαχωρίσει το σχεδιασμό από την αρχιτεκτονική της εφαρμογής, δίνοντας την ευκαιρία στον προγραμματιστή, να ασχοληθεί περισσότερο με τη λειτουργία και τη λειτουργικότητα της εφαρμογής και όχι τόσο με την παραμετροποίηση των εργαλείων αρχιτεκτονικής, που θα χρησιμοποιήσει.

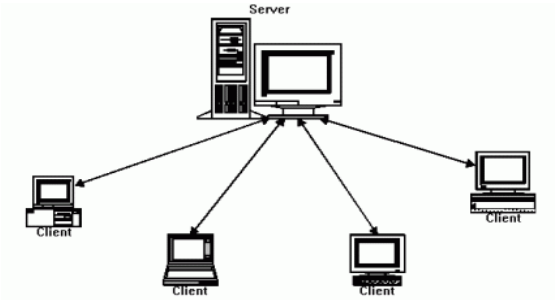
Με γνώμονα τα παραπάνω μελετήθηκε και υλοποιήθηκε μια εφαρμογή για μεταφορά μηνυμάτων πολυμεσικού περιεχομένου πάνω από ασύρματο δίκτυο δεδομένων κινητής τηλεφωνίας.

Στο Κεφάλαιο 2 περιγράφεται η Μεθοδολογία που ακολουθήθηκε στην παρούσα μελέτη. Στο Κεφάλαιο 3 αναλύονται τα εργαλεία που χρησιμοποιήθηκαν. Στο Κεφάλαιο 4 αναλύεται η υλοποίηση της εφαρμογής. Και στο Κεφάλαιο 5 αναλύονται τα τελικά συμπεράσματα.

2. ΜΕΘΟΔΟΛΟΓΙΑ

2.1 Αρχιτεκτονική Πελάτη – Εξυπηρετητή

Το μοντέλο αρχιτεκτονικής Πελάτη-Εξυπηρετητή χρησιμοποιείται στις κατανεμημένες δικτυακές εφαρμογές, για να χωρίσει τα καθήκοντα ή το φόρτο εργασίας μεταξύ των παροχών ενός πόρου ή μιας υπηρεσίας των αντίστοιχων εξυπηρετητών (servers) και των αιτούντων υπηρεσιών, που ονομάζονται πελάτες (clients).

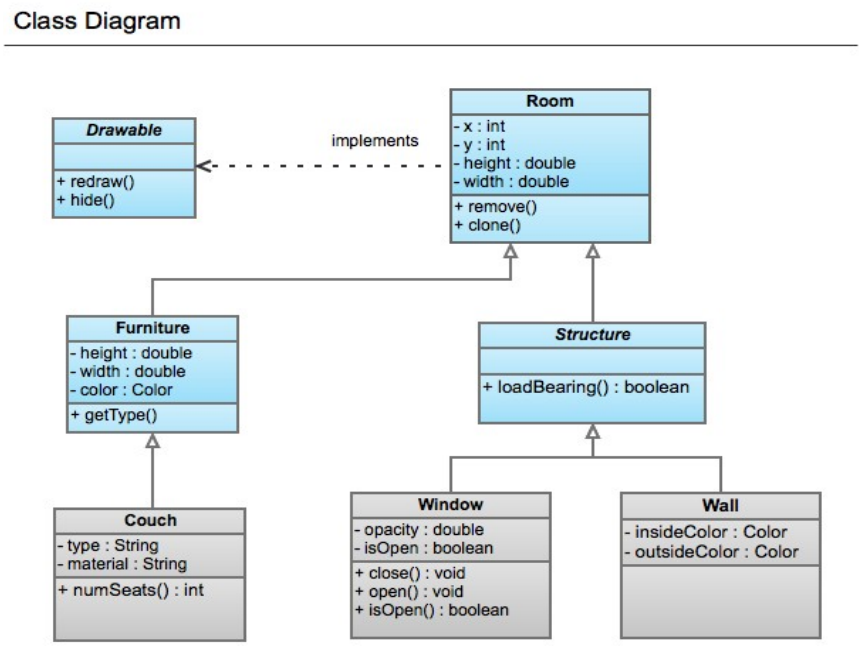


Εικόνα 1: Αρχιτεκτονική Πελάτης-Εξυπηρετητής

2.2 Αρχιτεκτονική MDA

Το MDA[9] (model driven architecture) είναι μια προσέγγιση σχεδιασμού λογισμικού για την ανάπτυξη συστημάτων λογισμικού. Παρέχει ένα σύνολο κατευθυντήριων γραμμών για τη διάρθρωση των προδιαγραφών, οι οποίες εκφράζονται ως μοντέλα.

Το MDA επικεντρώνεται στην παραγωγή κώδικα υπό αφηρημένη (abstract) μορφή, με διαγράμματα μοντελοποίησης (π.χ. διαγράμματα κλάσεων). Τα διαγράμματα αυτά περιγράφονται με την χρήση της γλώσσας μοντελοποίησης UML (Unified Modeling Language).



Εικόνα 2: Αρχιτεκτονική MDA

3. ΕΡΓΑΛΕΙΑ

Για την καλύτερη κατανόηση του παρόντος μοντέλου εξυπηρετητή – πελάτη (server – client) θα εξετάσουμε ένα προς ένα τα εργαλεία (frameworks) που χρησιμοποιήθηκαν.

3.1 ΕΦΑΡΜΟΓΗ ΠΕΛΑΤΗ (client side – iPhone RestKit)

Το RestKit[3] είναι ένα πλαίσιο (framework) το οποίο χρησημοποιήται στις συσκεύες κινητής τηλεφωνίας της Apple (iPhone) καθώς και στους υπολογιστές με λειτουργικό Mac OS X. Παρέχει πρόσβαση στο δίκτυο για Restful HTTP υπηρεσίες διαδικτύου, μια ισχυρή μηχανή χαρτογράφησης αντικειμένων, και την ενσωμάτωση με το πλαίσιο πυρήνα δεδομένων (core framework) της Apple για την τοπική διατήρηση των αντικειμένων που υπάρχουν σε ένα απομακρυσμένο σύστημα .

- Απλή, υψηλού επιπέδου HTTP αίτηση / απόκριση του συστήματος
- Πλαίσιο στήριξη για την αλλαγή server και περιβάλλοντος
- Σύστημα χαρτογράφησης αντικειμένων
- Υποστήριξη του πυρήνα δεδομένων

3.2 ΕΦΑΡΜΟΓΗ ΕΞΥΠΗΡΕΤΗΤΗ (server side – Java Server)

Η σύγχρονη αρχιτεκτονική υλοποίησης λογισμικού απαιτεί την χρήση της αρχιτεκτονικής MVC (Model-View-Controller)[4]. Το Model, δηλαδή το μοντέλο, αντιστοιχεί στα δεδομένα της υλοποίησης, οπως αυτά αποθηκεύονται σε βάσεις δεδομένων ή σε άλλα συστήματα αρχειοθέτησης και διαχείρισης δεδομένων. Το View, δηλαδή η παρουσίαση, αντιστοιχεί στην εμφάνιση της εφαρμογής στις τερματικές συσκευές και ο Controller, δηλαδή ο ελεγκτής, αντιστοιχεί στον έλεγχο ροής και διαχείρισης των δεδομένων.

3.2.1 STRUTS

Το πλαίσιο του Struts[5] παρέχει τις παρακάτω τρεις βασικές λειτουργίες:

- Διαχειριστή αιτημάτων για συγκεκριμένο URI.
- Διαχειριστή αποκρίσεων.
- Βιβλιοθήκη tag για τη βοήθεια του προγραμματιστή στη δημιουργία διαδραστικών εφαρμογών.
- Model-View-Controller: ένα HTTP και servlet-based πλαίσιο που παρέχει συνδέσεις για την επέκταση και προσαρμογή των εφαρμογών web, παρέχοντας και τη χρήση των Restful υπηρεσιών δικτύου.

Το Struts δουλεύει καλά με συμβατικές εφαρμογές REST αρχιτεκτονικής, όπως και με τεχνολογίες SOAP και AJAX.

3.2.2 SPRING

Το πλαίσιο εργασιών SPRING[6] περιλαμβάνει διάφορες ενότητες για την παρακάτω σειρά υπηρεσιών:

- Αντιστροφή του Control Container: Διαμόρφωση των στοιχείων της εφαρμογής και διαχείριση του κύκλου ζωής των Java αντικειμένων.
- Θεματοστρεφής προγραμματισμός: ο οποίος επιτρέπει την εφαρμογή cross-cutting ρουτινών.
- Πρόσβαση δεδομένων: Με χρήση των σχεσιακών συστημάτων διαχείρισης βάσεων δεδομένων και των αντικείμενο-σχεσιακών εργαλείων χαρτογράφησης (object-relational mapping tools), χρησιμοποιώντας τον οδηγό JDBC. Όπως και των πρόσφατων μη σχεσιακών βάσεων δεδομένων (Non Relational Databases) με την χρήση αντίστοιχων οδηγών.
- Διαχείρισης συναλλαγών: ενοποιεί διάφορα API διαχείρισης συναλλαγών και συντονίζει συναλλαγές για τα Java αντικείμενα.
- Πλαίσιο απομακρυσμένη πρόσβασης: διαμορφώσιμο RPC-style (Remote Procedure Call) για την έξοδο και είσοδο των Java αντικειμένων πάνω από δίκτυα που υποστηρίζουν RMI (*Remote Method Invocation*), CORBA (Common Object Request Broker Architecture) και HTTP-based πρωτόκολλα, συμπεριλαμβανομένων των υπηρεσιών web (SOAP-Simple Object Access Protocol)

3.2.3 HIBEPNATE

Το hibernate[7] είναι μια βιβλιοθήκη αντικείμενο-σχεσιακής χαρτογράφησης (ORM) για τη γλώσσα Java, παρέχοντας ένα πλαίσιο για τη χαρτογράφηση ενός αντικειμενοστραφούς μοντέλου σε μια παραδοσιακή σχεσιακή βάση δεδομένων.

3.2.4 APACHE MAVEN

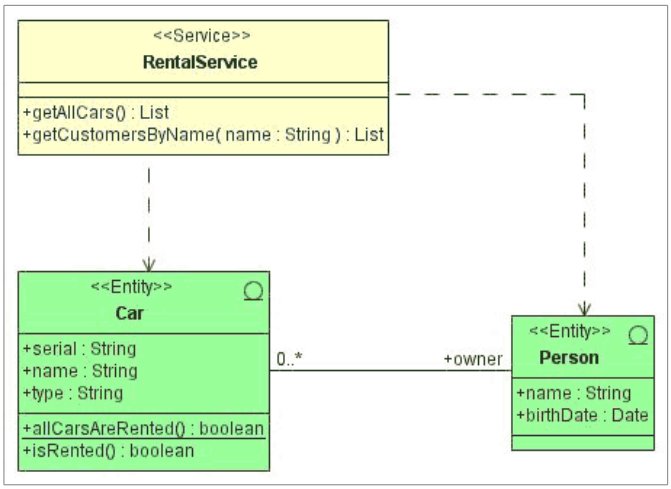
Το Maven[8] είναι ένα εργαλείο αυτοματισμού παραγωγής εκτελέσιμων προγραμμάτων που χρησιμοποιείτε σε Java έργα. Το Maven είναι παρόμοιο με το εργαλείο Apache Ant, αλλά βασίζεται σε διαφορετικές έννοιες και δουλεύει με ένα ριζικά διαφορετικό τρόπο. Μπορεί επίσης να χρησιμοποιηθεί για την κατασκευή και διαχείριση έργων γραμμένων σε C #, Ruby, Scala, και σε άλλες γλώσσες. Το Maven φιλοξενείται από την Apache Software Foundation, όπου ήταν στο παρελθόν μέρος του έργου Jakarta.

Το Maven χρησιμοποιεί ένα αρχείο XML για να περιγράψει το λογισμικό, που παράγει το εκτελέσιμο πρόγραμμα, τις εξαρτήσεις του σε άλλες εξωτερικές μονάδες και τα συστατικά αυτών, τους καταλόγους, τα απαιτούμενα plug-ins και τη σειρά εκτέλεσης.

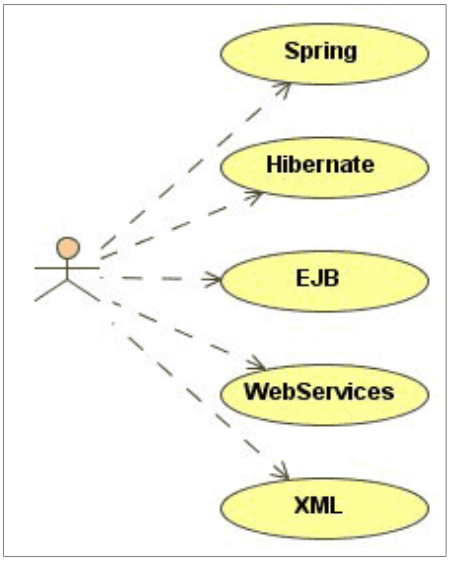
Έχει προκαθορισμένους στόχους για την εκτέλεση σαφώς καθορισμένων καθηκόντων, όπως η μεταγλώττιση κώδικα και packaging αυτού. Το Maven δυναμικά λαμβάνει τις Java βιβλιοθήκες και τα Maven plug-ins από ένα ή περισσότερα αποθετήρια (repositories), όπως το Maven 2 Repository Central.

3.2.5 ANDROMDA

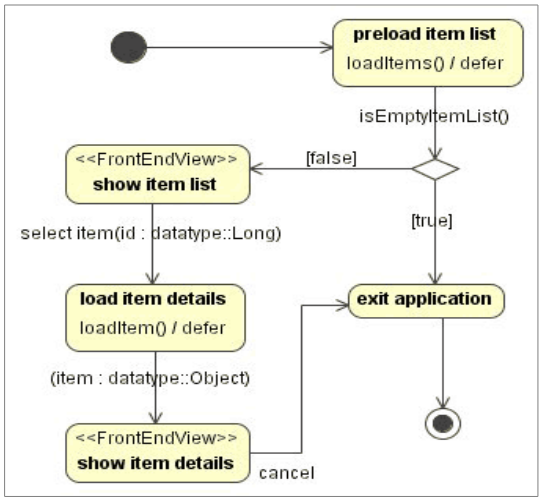
Το AndroMDA[10] είναι ένα MDA πλαίσιο ανοικτού κώδικα που χρησιμοποιώντας μοντέλα (συνήθως UML μοντέλα που αποθηκεύονται σε XMI μορφή), σε συνδυασμό με κατάλληλα AndroMDA plugins παράγει τα επιθυμητά στοιχεία λογισμικού. Μπορούμε να δημιουργήσουμε στοιχεία για κάθε γλώσσα που θέλουμε, Java, .Net, HTML, PHP, μπορούμε απλά να γράψουμε (ή να προσαρμόσουμε υπάρχοντα) plugins για να το υποστηρίξει. Με το AndroMDA μπορούμε να στήσουμε ένα νέο έργο J2EE από το μηδέν, στο οποίο ο κώδικας δημιουργείται από ένα μοντέλο UML. Μπορούμε να επιλέξουμε να δημιουργήσουμε κώδικα για Hibernate, EJB, Spring, WebServices και Struts.



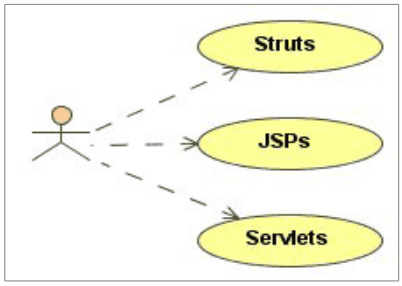
Εικόνα 4: Παράδειγμα κλάσεων



Εικόνα 5: Έξοδος κλάσεων



Εικόνα 6: Παράδειγμα διαγράμματος δραστηριότητας



Εικόνα 7: Έξοδος διαγράμματος δραστηριότητας

Επίπεδο Παρουσίασης: Το AndroMDA αυτή τη στιγμή προσφέρει δύο τεχνολογίες ως επιλογή για την κατασκευή του επίπεδου παρουσίασης: το Struts και το JSF. Δέχεται UML διαγράμματα δραστηριότητας ως είσοδο για να καθοριστούν οι ροές της σελίδα και παράγει εξαρτήματα του Struts ή του JSF πλαισίου.
Επιχειρησιακό Επίπεδο: Το Επιχειρησιακό Επίπεδο που δημιουργείται από το AndroMDA αποτελείται κυρίως από υπηρεσίες του Spring. Τις υπηρεσίες αυτές τις υλοποιούμε με το χέρι στις κενές μεθόδους που έχουν ήδη δημιουργηθεί από το AndroMDA. Οι υπηρεσίες μπορούν να υλοποιηθούν με χρήση EJBs (Enterprise JavaBeans), περίπτωση κατά την οποία οι υπηρεσίες θα πρέπει να τοποθετηθούν σε ένα container για EJBs (π.χ., JBoss, Tomcat). Οι υπηρεσίες μπορούν επίσης να εκτίθενται ως υπηρεσίες Web, παρέχοντας μια πλατφόρμα πρόσβασης στις λειτουργίες τους, για τους πελάτες, με ανεξάρτητο τρόπο.

Επίπεδο Πρόσβασης στα Δεδομένα: Το AndroMDA αξιοποιεί το δημοφιλές αντικείμενο-σχεσιακό εργαλείο χαρτογράφησης που ονομάζεται Hibernate για να δημιουργήσει το επίπεδο πρόσβασης δεδομένων στις εφαρμογές. Δημιουργεί αντικείμενα πρόσβασης δεδομένων (DAOs) για τις οντότητες που ορίζονται στο μοντέλο UML. Αυτά τα αντικείμενα πρόσβασης δεδομένων χρησιμοποιούν το API του Hibernate για να μετατραπούν αρχεία της βάσης δεδομένων σε αντικείμενα και το αντίστροφο.

Αποθήκευση Δεδομένων: Από το AndroMDA δημιουργούνται εφαρμογές, που χρησιμοποιούν το Hibernate, για να έχουν πρόσβαση στα δεδομένα, έτσι μπορούμε να χρησιμοποιήσουμε οποιαδήποτε από τις βάσεις δεδομένων που υποστηρίζονται από αυτό.

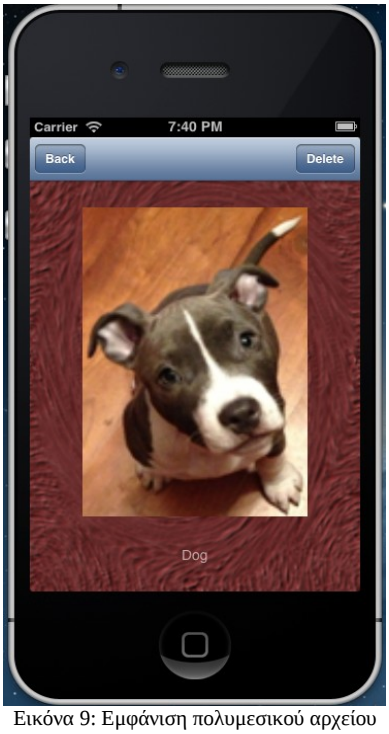
4. ΥΛΟΠΟΙΗΣΗ ΕΦΑΡΜΟΓΗΣ

Στη παρούσα υλοποίηση δημιουργήθηκε ένα σύστημα διαχείρισης πολυμεσικών αρχείων, τα οποία μεταφέρουν οι χρήστες μέσα από το πελάτη (client – iPhone) και πάνω απο το το ασύρματο τηλεφωνικό δίκτυο με τη μορφή JSON στον εξυπηρετητή (server - Java).

Από τη πλευρά του πελάτη (client), ο οποίος αναγνωρίζεται μέσω του user id, η εφαρμογή εμφανίζει ως πρώτη οθόνη μια λίστα με τα πολυμεσικά μηνύματα, που έχει ήδη τοποθετήσει ο χρήστης στον εξυπηρετητή.

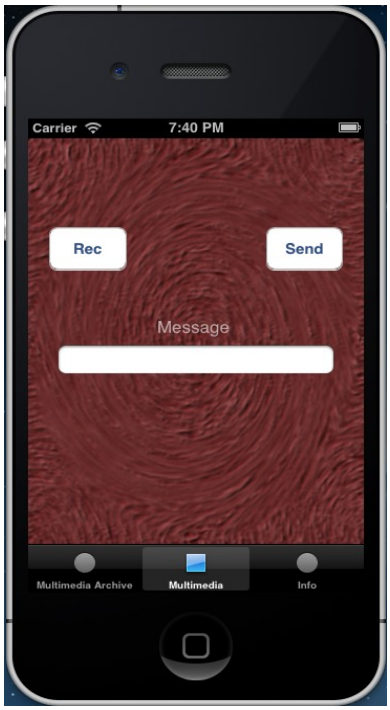
Όπου με την επιλογή ενός στοιχείου της παρακάτω λίστας (*Εικόνα 8*) εμφανίζεται το περιεχόμενο του πολυμεσικού μηνύματος (κείμενο και εικόνα), ενώ με το επάνω δεξιά κουμπί δίνεται η δυνατότητα διαγραφής του μηνύματος.

Εικόνα 8: Λίστα πολυμεσικών μηνυμάτων συνδεδεμένου χρήστη

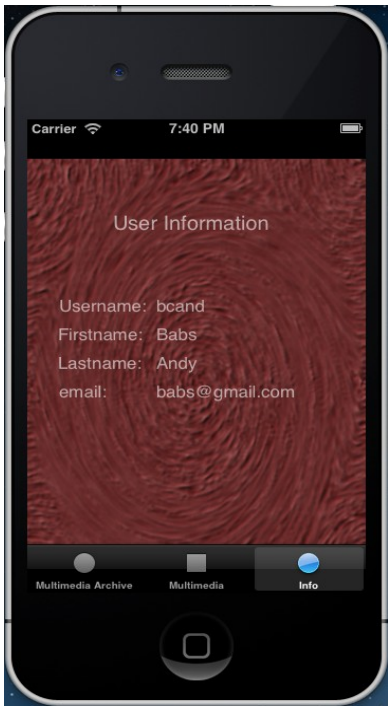


Εικόνα 9: Εμφάνιση πολυμεσικού αρχείου

Στη δεύτερη οθόνη ο χρήστης μπορεί να εγγράψει καθώς και να στείλει το πολυμεσικό του μήνυμα για αποθήκευση στον εξυπηρετητή.



Εικόνα 10: Αποστολή πολυμεσικού αρχείου



Εικόνα 11: Εμφάνιση στοιχείων χρήστη

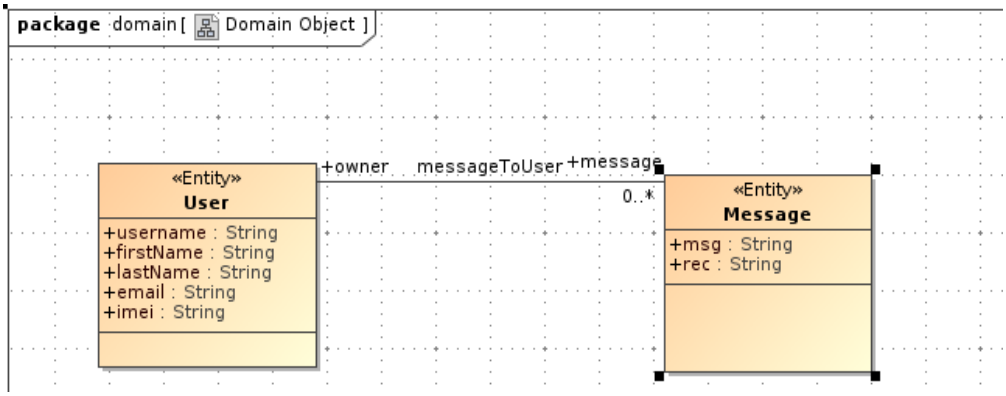
Στη τρίτη και τελευταία οθόνη ο χρήστης μπορεί να δει κάποια από τα στοιχεία του που είναι αποθηκευμένα στον εξυπηρετητή.

Για την κωδικοποίηση των πολυμεσικών μηνυμάτων, έτσι ώστε να μπορούν να μεταφερθούν πάνω στο JSON, χρησιμοποιήθηκε η κωδικοποίηση Base64.

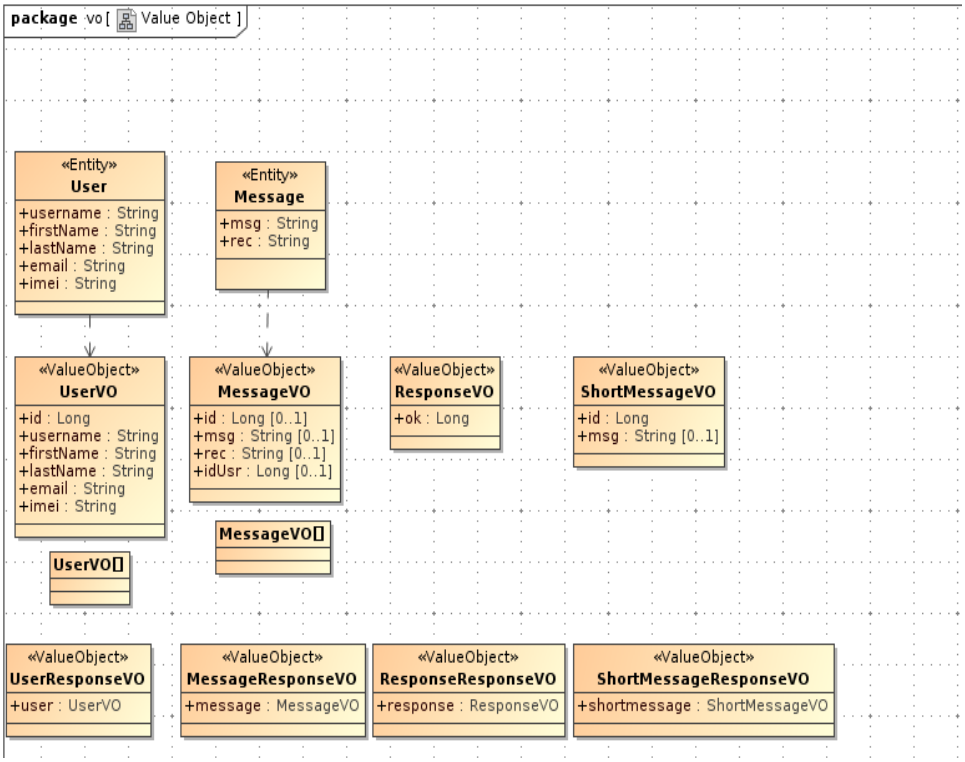
Από την πλευρά του εξυπηρετητή χρησιμοποιήθηκε Magic Draw για την δημιουργία του UML μοντέλου. Όπου το μοντέλο χωρίστηκε στις παρακάτω υποενότητες σχεδιαγραμμάτων για την καλύτερη κατανόηση και διαχείριση των περιγραφών:

- DomainObjects
- ValueObjects
- UserServices
- MessageServices

Αναλυτικότερα, το DomainObjects περιεχέι τις οντότητες της εφαρμογής (*Εικόνα 12*). Οι οντότητες είναι συσχετισμένες μεταξύ τους με σχεσιακή λογική ένα σε πολλά (one to many). Το ValueObjects περιέχει τα αντικείμενα που διαχειρίζετε ο εξυπηρετητής (*Εικόνα 13*). Το UserServices περιέχει τις υπηρεσίες του χρήστη (*Εικόνα 14*). Και τέλος το MessageServices περιέχει τις υπηρεσίες μηνυμάτων (*Εικόνα 15*). Τα PeopleServices και MessageServices αποτελούν την επικοινωνία του server με τον έξω κόσμο.



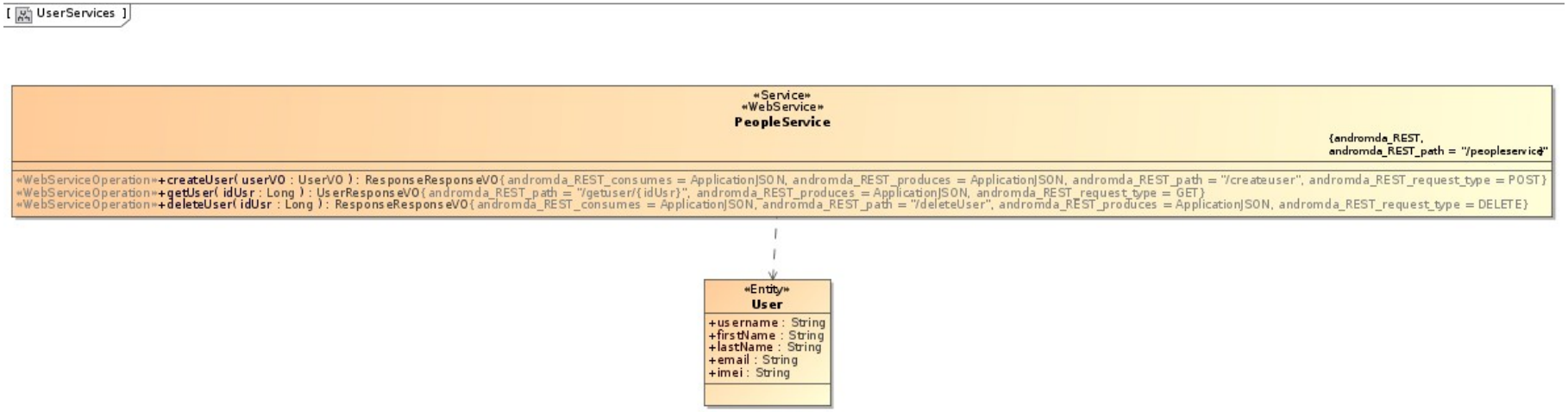
Εικόνα 12: Οντότητες του Domain Objects



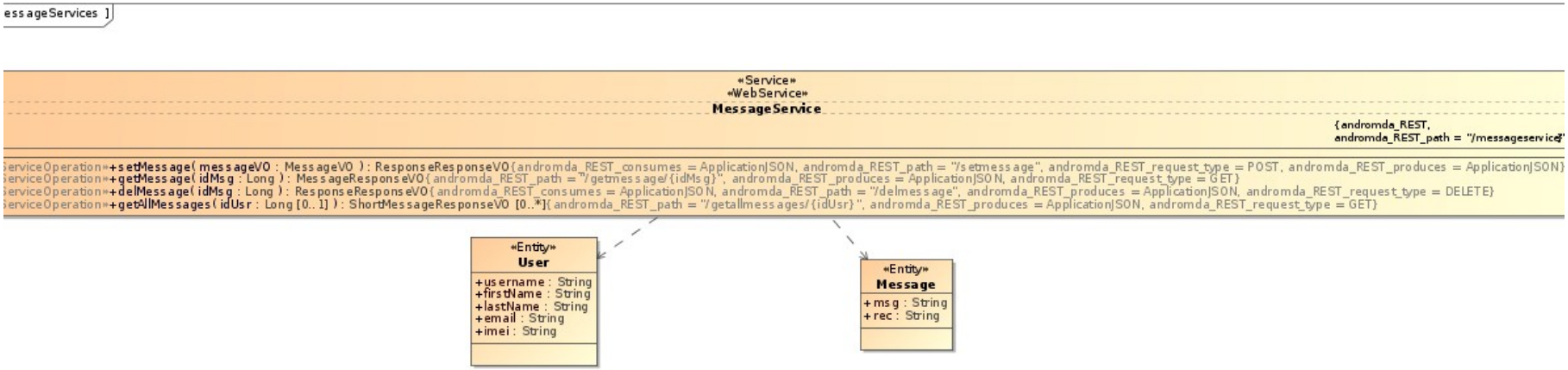
Εικόνα 13: Value Objects που περιέχει τα αντικείμενα που διαχειρίζεται ο εξυπηρετητής

Στην υλοποίηση του εξυπηρετητή χρησιμοποιήθηκαν EJBs με container τον Tomcat.

και από τα δύο Services: PeopleService και MessageService τα οποία αποτελούν και την επικοινωνία του με τον έξω κόσμο.



Εικόνα 14: PeopleService



Εικόνα 15: MessageService

5. ΣΥΜΠΕΡΑΣΜΑΤΑ

Η χρήση του παρόντος μοντέλου υλοποίησης εφαρμογών συνδύασε με επιτυχία ταχύτητα υλοποίησης εφαρμογής, ανάθεση κυρίως του σχεδιαστικού τμήματος στο προγραμματιστή, καλύτερη κατανόηση του μοντέλου της εφαρμογής και της χρήσης, μέσα από την σχηματική αναπαράσταση του μοντέλου, αλλά και την ελεύθερη πρόσβαση στο πηγαίο κώδικα. Το REST επιτυγχάνει να προσπεράσει τα κενά κάλυψης της κινητής τηλεφωνίας με την εκ νέου αποστολή των αποτυχημένων μηνυμάτων. Και τέλος είναι λογισμικό ανοικτού κώδικα με άδεια BSD, συνεπώς έχει μηδενικό κόστος αγοράς και χρήσης.

Ωστόσο, σημαντικό μειονέκτημα κατά την χρήση του μοντέλου αποτελεί η παραμετροποίηση του, τόσο κατά επιλογή των εργαλείων, καθώς απαιτείται να έχει υλοποιηθεί ο κατάλληλος οδηγός του πλαισίου εργαλείων, όσο και κατά την αλλαγής και ανανέωση των τεχνολογιών ή εργαλείων. Ως εκ τούτου, είναι επιβεβλημένη η ανάγκη προχωρημένης γνώσης των εργαλείων αρχιτεκτονικής, όπως και η συνεχής ανανέωση των βιβλιοθηκών του μοντέλου υλοποίησης androMDA.

Τέλος, όσο αναφορά τον προγραμματισμό στην συσκευή IPhone της Apple, εκτός κάποιων προβλημάτων στη διαχείριση μνήμης, είναι απλός και διασκεδαστικός.

6. ΑΝΑΦΟΡΕΣ

[1] Web Services http://en.wikipedia.org/wiki/Web_service
[2] RestFul Web Services http://en.wikipedia.org/wiki/Representational_state_transfer
[3] RestKit Framework for iPhone https://github.com/RestKit/RestKit/wiki
[4] Model view controller http://en.wikipedia.org/wiki/Model-view-controller
[5] Apache Struts Framework http://struts.apache.org
[6] Spring Framework http://en.wikipedia.org/wiki/Spring_Framework
[7] Hibernate Framework http://en.wikipedia.org/wiki/Hibernate_(Java)
[8] Apache Maven Framework http://en.wikipedia.org/wiki/Apache_Maven
[9] Model Driven Architecture (MDA) http://en.wikipedia.org/wiki/Model-driven_architecture
[10] AndroMDA Framework http://www.andromda.org/docs/index.html