Τεχνολογικό Εκπαιδευτικό Ίδρυμα Κρήτης

Σχολή Τεχνολογικών Εφαρμογών

Τμήμα Μηχανικών Πληροφορικής

Πτυχιακή Εργασία

Τίτλος: Συνεργατική εκτέλεση καθηκόντων με απαιτήσεις διαμοιρασμού χωρικών δεδομένων

Δημήτριος Μαρματάκης

Α.Μ 3245

Επιβλέπων καθηγητής : Δημοσθένης Ακουμιανάκης

ΗΡΑΚΛΕΙΟ
2017

Ευχαριστίες

Θα ήθελα να ευχαριστήσω την οικογένεια μου για την υποστήριξη που μου παρείχε κατά την διάρκεια των σπουδών μου. Επίσης θερμές ευχαριστίες θα ήθελα να εκφράσω στον κ. Δημοσθένη Ακουμιανάκη και τον κ. Νικόλαο Βιδάκη για την ευκαιρία που μου δώσανε να εκπονήσω την πτυχιακή και πρακτική μου εργασία στο iSTLab. Τέλος ευχαριστώ το Βελλή Γιώργο για την βοήθεια και την καθοδήγηση του στο πλαίσιο της πρακτικής μου άσκησης αλλά και της παρούσας πτυχιακής εργασίας, καθώς και τον Αντώνη και τον Μιχάλη για την υπομονή και την υποστήριξη που δείχνουν όλα αυτά τα χρόνια της συνεργασία μας.

Abstract

The goal of this thesis is to develop a system that support's synchronous collaborative tasks/work, which is specified from the relative location of each group member around the globe. Specifically this system will support multiple UIs in desktop (web-based app) and mobile (android-based app) environments with spatial information capabilities, group awareness and task synchronization. To achieve this goal we examine what collaborative systems are, the supported types of communication (synchronous, asynchronous), what issues arise (awareness, coordination, translucence) and what requirements arise from the development of collaborative UIs in multiple environments. For the implementation of our system we use the most suitable toolkit to track the location of group members, which allows remotely controlling the progress of a task and finally our system support's chat communication to group members who perform a task.

Σύνοψη

Στην εν λόγο πτυχιακή εργασία αναπτύχθηκε ένα σύστημα για την υποστήριξη σύγχρονων συνεργατικών καθηκόντων τα οποία καθορίζονται, μεταξύ άλλων και από τη σχετική θέση των μελών της ομάδας στο φυσικό περιβάλλον/χώρο. Πιο συγκεκριμένα το σύστημα που αναπτύχθηκε έχει τη δυνατότητα υποστήριξης πολλαπλών διεπαφών σε περιβάλλοντα επιτραπέζιου υπολογιστή (web-based app) και κινητής συσκευής (mobile-based app), με δυνατότητες διαμοιρασμού χωρικής πληροφορίας, ομαδικής αφύπνισης και συγχρονισμό καθηκόντων. Για την επίτευξη του στόχου αυτού μελετήσαμε τι είναι τα συνεργατικά συστήματα και καθήκοντα, τα ζητήματα που προκύπτουν (ενημερότητα, συντονισμός και διαφάνεια) στην ανάπτυξη τέτοιου είδους συστημάτων, καθώς και τις απαιτήσεις που προκύπτουν από την ανάπτυξη διεπαφών συνεργατικών καθηκόντων σε πολλαπλά περιβάλλοντα χρήσης. Όσον αφορά την υλοποίηση του συστήματος έγινε χρήση κατάλληλων εργαλειοθηκών για την ανίχνευση της θέσης μελών της ομάδας στο χώρο, την ανάπτυξη εναλλακτικών αλληλεπιδραστικών αναπαραστάσεων που θα επιτρέπουν τον εξ αποστάσεως έλεγχο της προόδου ενός καθήκοντος, καθώς και την επικοινωνία των μελών της ομάδας που εκτελούν το καθήκον.

Πίνακας Περιεχομένων

Πίνακας Εικόνων

1 Εισαγωγή

1.1 Σύγχρονα συνεργατικά συστήματα

Διανύεται μια περίοδος όπου η τεχνολογία και η πληροφορική αναπτύσσονται συνεχώς φέρνοντας στο προσκήνιο νέες καινοτόμες υπολογιστικές πλατφόρμες και συσκευές. Οι δυνατότητες που προσφέρουν αυξάνονται με εκθετικό ρυθμό μέρα με τη μέρα συνεισφέροντας στην απλοποίηση διαφόρων εφαρμογών σε προσωπικό και επαγγελματικό επίπεδο. Μαζί όμως με την ανάπτυξη εμφανίζονται νέες προκλήσεις, όπως η ανάγκη υποστήριξης σύγχρονων συνεργατικών συστημάτων. Τα σύγχρονα συνεργατικά συστήματα είναι μια νέα και αρκετά ενδιαφέρουσα τεχνολογία η οποία επιτρέπει στους χρήστες να ενεργούν ταυτόχρονα επάνω στην ίδια εφαρμογή, για παράδειγμα επιτρέπουν την ταυτόχρονη επεξεργασία κειμένου από μεγάλο αριθμό χρηστών. Ακόμη ανάλογα με την υλοποίηση του συστήματος, οι χρήστες έχουν την δυνατότητα να ενεργούν πάνω στην ίδια εφαρμογή είτε από τον ίδιο χώρο είτε αν είναι γεωγραφικά κατανεμημένοι. Μια ακόμα πρόκληση που θα πρέπει να λάβουμε υπόψιν, είναι η ανομοιογένεια των εφαρμογών, λόγο του πλήθους νέων υπολογιστικών πλατφορμών, που έχουν έρθει στο προσκήνιο. Λόγο αυτής της πρόκλησης η ανάπτυξη σύγχρονων συνεργατικών συστημάτων γίνετε ολοένα και πιο σύνθετη, καθώς πέρα από τα ζητήματα επικοινωνίας, οι προγραμματιστές θα πρέπει να λύσουν και θέματα που αφορούν την ανομοιογένεια σε επίπεδο πλατφορμών.

1.2 Στόχος πτυχιακής εργασίας

Ο κεντρικός στόχος της παρούσας πτυχιακής εργασίας είναι, η ανάδειξη των δυνατοτήτων ενός σύγχρονου συνεργατικού συστήματος με έμφαση στα χωρικά δεδομένα. Για την επίτευξη αυτού του στόχου έγινε η ανάπτυξη ενός σύγχρονου συνεργατικού συστήματος κατανεμημένων κινητών χρηστών, που εστιάζει στην αξιοποίηση και εκμετάλλευση χωρικών δεδομένων που παρέχονται από τους αυτούς. Στα πλαίσια αυτού του συστήματος αναπτύχθηκε μια εφαρμογή για τις πλατφόρμες Web και Android, η οποία αξιοποιεί τρέχοντες τεχνολογίες που αφορούν την επικοινωνία – ανταλλαγή μηνυμάτων μεταξύ των χρηστών, αλλά και για τη γραφική αναπαράσταση των χωρικών δεδομένων. Στα κεφάλαια που ακολουθούν γίνετε μια διεξοδική ανάλυση της υλοποίηση, των ρόλων και των δυνατοτήτων που έχει η εφαρμογή σύμφωνα με την πλατφόρμα που έχει υλοποιηθεί.

1.3 Οργάνωση αναφοράς

Η αναφορά περιλαμβάνει έξι κεφάλαια. Το κεφάλαιο 1 παρουσιάζει συνοπτικά το πεδίο εφαρμογής και το στόχο της εργασίας. Στο κεφάλαιο 2 αναλύονται βασικές έννοιες που αφορούν τα συνεργατικά συστήματα, την επικοινωνία και τα APIs που συναντάμε σε κάθε πλατφόρμα υλοποίησης. Το κεφάλαιο 3 παρουσιάζει τις περιπτώσεις χρήσης, τις απαιτήσεις που αφορούν την υλοποίηση και τον συγχρονισμό, καθώς και τις πλατφόρμες υλοποίησης. Το κεφάλαιο 4 παρουσιάζει την προσέγγιση, αρχιτεκτονική και την υλοποίηση της εφαρμογής. Στο κεφάλαιο 5 παρουσιάζεται το σενάριο επίδειξης μαζί με. Τέλος στο κεφάλαιο 6 παρουσιάζονται τα συμπεράσματα και η μελλοντικές επεκτάσεις.

2 Διεπαφές συνεργατικών καθηκόντων

Στο κεφάλαιο θα επιχειρήσουμε μια συνοπτική επισκόπηση της έννοιας της συνεργατικής διεπαφής και ακολούθως θα αναφερθούμε σε σχεδιαστικής φύσης ζητήματα καθώς και σε τεχνολογίες και εργαλεία που αφορούν την υπολογιστική υλοποίησή τους.

2.1 Ορισμός συνεργατικών καθηκόντων

Ο όρος συνεργατικό καθήκον, στη φυσική μας ζωή, υπονοεί καθήκοντα που αναλαμβάνονται από μια ομάδα ανθρώπων που συνεργάζονται μεταξύ τους με σκοπό να συνεισφέρουν στην επίτευξη κοινά αποδεκτών στόχων. Όταν τα μέλη της ομάδας είναι διασκορπισμένα και η συνεργασία υλοποιείται με τη διαμεσολάβηση Η/Υ, απαιτείται ειδικό λογισμικό ικανό να διεκπεραιώσει τη συναναστροφή των εμπλεκομένων και να εγγυηθεί κατάλληλη κατανομή του φόρτου εργασίας μεταξύ των μελών της ομάδας. Από τεχνολογικής σκοπιάς αυτό επιβάλει τη χρήση εξειδικευμένων εργαλείων (APIs) και αρχιτεκτονικών (replicated, centralized, hybrid) που να υποστηρίξουν την ανταλλαγή δεδομένων και να διασφαλίζουν τη συνοχή και συνέπεια αυτής μεταξύ των κατανεμημένων κόμβων. Επιπλέον, απαιτείται ειδική μέριμνα ούτως ώστε η απαραίτητη πληροφορία να μπορεί να διαχέεται κατάλληλα και απρόσκοπτα διαμέσου των συνεργατικών τεχνουργημάτων και του διαμοιραζόμενου συνεργατικού χώρου εργασίας με στόχο την υποστήριξη ενημερότητας και συντονισμού μεταξύ των κατανεμημένων συνεργαζόμενων χρηστών. Αυτού του είδους η πληροφορία κωδικοποιείται υπό το πρίσμα σειράς εννοιών οι οποίες συνιστούν τα σχεδιαστικά ζητήματα των συστημάτων που προορίζονται για τη συνεργασία εταίρων.

2.2 Σχεδιαστικά ζητήματα

Κύρια σχεδιαστική ζητήματα που αφορούν την αποτελεσματική υπολογιστική υποστήριξη της εκτέλεσης σύγχρονων συνεργατικών καθηκόντων, αφορούν στην: α) ενημερότητα (awareness), β) συντονισμό (coordination) και γ) διαφάνεια (translucence). Ακολούθως αναλύεται κάθε ένα από αυτά.

2.2.1 Ενημερότητα (Awareness)

Ως ενημερότητα (awareness) ορίζουμε τη καταγραφή, πληροφορίων που ανταλλάσσουν/γνωρίζουν τα μέλη μιας ομάδας στην πραγματική ζωή (π.χ. με ποια

δραστηριότητα ασχολείται κάποιο από τα μέλη της ομάδας, αν είναι διαθέσιμο κάποιο άλλο μέλος κοκ.) και η χρήση τους, έτσι ώστε το υπολογιστικό σύστημα να είναι σε θέση να παρουσιάζει σε κάθε μέλος όλα όσα χρειάζεται για την ολοκλήρωση της δραστηριότητας που εκτελεί [1]. Ανάλογα με το είδος της συνεργασίας, ενημερότητα που απαιτείται να υποστηρίζεται μπορεί να είναι της κατηγορίας:

- '*Group Awareness*' όταν γνωρίζουμε με ποιο/α από τα μέλη της ομάδας θα συνεργαστούμε, για την ολοκλήρωση μιας δραστηριότητας, τις ενέργειες που εκτελούν, καθώς και με ποιον τρόπο αυτές θα επηρεάσουν τη δική μας εργασία και το αντίστροφο [2] [3].
- *'Social Awareness*' όταν η ενημέρωση αφορά τις καθημερινές ασχολίες και συνήθειες μιας ομάδας ατόμων που συνεργάζονται το εργασιακό του περιβάλλον. Πιο συγκεκριμένα για να υπάρχει όσο το δυνατόν πιο καλή συνεργασία, θα πρέπει να γνωρίζουμε για παράδειγμα με τι ασχολείται το άτομο που συνεργαζόμαστε, με ποιόν μιλάει, πότε μπορούμε να τον απασχολήσουμε κτλ. [4] [5].
- '*Location Awareness*': έχουμε όταν γνωρίζουμε τη γεωγραφική θέση των συνεργατών μας, αλλά έχουμε και όταν γνωρίζουμε τη θέση του, μέσα στο χώρο εργασίας του [6] [7].

2.2.2 Συντονισμός (Coordination)

Σύμφυτος της εκτέλεσης συνεργατικών καθηκόντων είναι ο κατακερματισμός του συνολικού έργου σε συγκεκριμένους χρήστες ή ομάδες χρηστών υπό την έννοια μικρότερων υποδιαιρεμένων μονάδων/καθηκόντων εργασίας. Δεδομένου ότι η είσοδος ενός χρήστη (το αποτέλεσμα της εργασίας) μπορεί να αποτελεί είσοδο για την εργασία ενός - ή ένδειξη για τη βελτίωση του τρέχοντος καθήκοντος ενός - άλλου χρήστη, είναι σημαντικό να μπορεί να υποστηριχθεί ο λεγόμενος συντονισμός μεταξύ των εμπλεκομένων κάθε φορά μερών.

2.2.3 Διαφάνεια (Translucence)

Στόχος της κοινωνικής διαύγειας (social translucence) είναι η κατανόηση, καταγραφή και μεταφορά στον ψηφιακό κόσμο, ιδιότητες από τον φυσικό κόσμο. Όπως τον τρόπο που επικοινωνούν δύο άνθρωποι μεταξύ τους, τις αλληλεπιδράσεις που γίνονται ανάμεσα σε καταστάσεις πρόσωπο με πρόσωπο κ.ά. Για την επίτευξη της μεταφοράς αυτής θα πρέπει να λάβουμε υπόψιν μας τα εξής: ορατότητα/διαφάνεια (visibility), επίγνωση εταίρων(awareness), υπευθυνότητα(accountability) [8].

Με τον όρο διαφάνεια εννοούμε την ύπαρξη μηχανισμού ο οποίος θα αποκρύπτει προσωπικές πληροφορίες αλλά θα φέρνει στο φως πληροφορίες που έχουν να κάνουν με τη κοινωνική συμπεριφορά του ατόμου/εταίρου.

Η επίγνωση εταίρων τώρα είναι φυσικό επακόλουθο, καθώς υπάρχει η ανάγκη να έχουμε μια σαφή εικόνα των δραστηριοτήτων που εκτελεί ένα άτομο στα πλαίσια μιας ομάδας, όπως αναφέρετε και παραπάνω.

Οι πληροφορίες τώρα που συλλέγονται από τα awareness και visibility βοηθούν το σχεδιαστή του συστήματος να σχεδιάσει ένα σύστημα έτσι ώστε οι χρήστες να είναι ενήμεροί για το τι κάνει ποιος, με ποιόν το κάνει κτλ., ενώ παράλληλα να δίνει στο χρήστη την αίσθηση της υπευθυνότητας για τα υπόλοιπα μέλη της ομάδας. Όπως αναφέρουν οι Erickson και ο Kellogg "η κοινωνική διαύγεια είναι ένα θεμελιώδεις απαίτηση για την υποστήριξη όλων των τύπων επικοινωνίας και συνεργασίας" [8].

2.3 Τεχνολογίες και εργαλεία

Στην συγκεκριμένη υπο-ενότητα θα εξετάσουμε βασικές τεχνολογίες και εργαλεία που χρησιμοποιούνται, για τη δημιουργία σύγχρονων κατανεμημένων συνεργατικών συστημάτων.

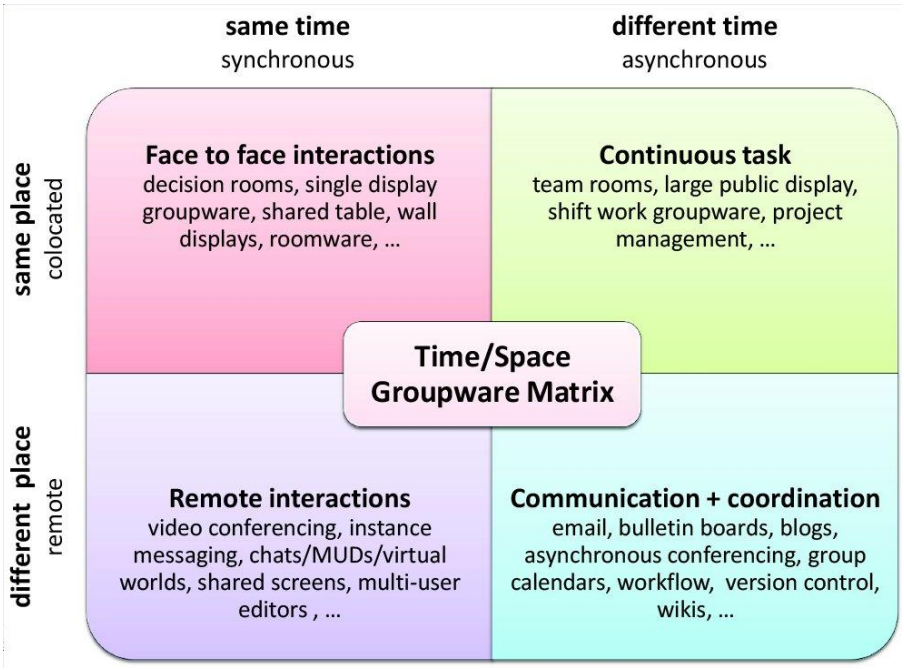
- Κλασικά εργαλεία ασύγχρονης επικοινωνίας π.χ email
- Σύγχρονη επικοινωνία π.χ online chat
- Multiuser widgets
- Εργαλειοθήκες διαδραστικών αντικειμένων
- desktop vs mobile settings
- Frameworks
- Περιβάλλοντα & συνεργατικών καθηκοντων π.χ google app engine

2.3.1.1 Ασύγχρονη επικοινωνία

Ένα ασύγχρονο συνεργατικό σύστημα χαρακτηρίζεται από ένα βαθμό ανεξαρτησίας ενός εταίρου από τον άλλο. Πιο συγκεκριμένα οι συνεργάτες που δουλεύουν ασύγχρονα, έχουν τυπικά μικρή ανάγκη για συχνό και καλά προσεγμένο συντονισμό ο ένας με τον άλλο. Ακόμα δεν υπάρχει ανάγκη άμεσης ενημέρωσης των συνεργατών, για αλλαγές που έχουν γίνει από άλλους πάνω σε κάποια διαμοιραζόμενα αντικείμενα που δουλεύουν μαζί [9]. Στη βιβλιογραφία το συναντάμε και ως "What You See Now Is What I See Then (WYSNIWIST)" [10]. Ένα κλασικό εργαλείο για ασύγχρονη και γεωγραφικά κατανεμημένη επικοινωνία είναι το email.

2.3.1.2 Σύγχρονη επικοινωνία

Στα σύγχρονα(πραγματικού χρόνου) συνεργατικά συστήματα τώρα, τα μέλη μιας ομάδας μπορούν να εκτελούν μια δραστηριότητα την ίδια στιγμή. Στα συστήματα αυτά υπάρχουν υψηλές απαιτήσεις τόσο σε awareness, όσο και σε συντονισμό, για να μπορεί το σύστημα να κρατήσει συγχρονισμένους τους χρήστες. Ένα ευρέος γνωστό παράδειγμα είναι το online chat. Στη βιβλιογραφία το συναντάμε ως "What You See Is What I See (WYSIWIS)" [11]. Στην παρακάτω εικόνα μπορούμε να δούμε κάποιες ενδεικτικές εφαρμογές/εργαλεία χωρισμένες σε χώρο και χρόνο.



Εικόνα 1: CSCW Matrix

2.3.2 Εργαλεία

Σε επίπεδο εργαλείων τώρα υπάρχουν αρκετές δουλείες πάνω σε widgets και interfaces πολλαπλών χρηστών (multi-user), σε συνεργατικές εργαλειοθήκες (groupware toolkits), frameworks και σε περιβάλλοντα συνεργατικών καθηκόντων. Παρακάτω θα δούμε τι είναι το καθένα από αυτά και κάποιες δουλείες που έχουν γίνει.

2.3.2.1 Multi-user Widgets

Widget μπορούμε να πούμε με πολύ απλά λόγια ότι είναι ένα μια πολύ απλή εφαρμογή ή ένα component που εκτελεί μια συγκεκριμένη εργασία. Η διεπαφή χρήστη (User Interface ή UI)

είναι η συλλογή από widgets ή components για τη δημιουργία μιας πιο σύνθετης εφαρμογής που θα είναι εύχρηστη και με συγκεκριμένες λειτουργίες. Όταν μιλάμε για widgets, components ή interfaces που υποστηρίζουν πολλαπλούς χρήστες, θα πρέπει να έχουμε στο μυαλό μας widgets/interfaces τα οποία υποστηρίζουν κάποια μορφή awareness. Ακόμα θα πρέπει να παρέχουν/υποστηρίζουν και κάποιον μηχανισμό για την επίτευξη συνεργασίας με σύγχρονο ή ασύγχρονο τρόπο.

2.3.2.2 Collaborative Toolkits

Με τον όρο εργαλειοθήκη(toolkit) εννοούμε τη συλλογή βιβλιοθηκών, που περιέχουν γραφικά ή και λειτουργικά widgets/components, που βοηθούν το προγραμματιστή στη γρήγορη και εύκολη ανάπτυξη διεπαφών. Σε μια συνεργατική βιβλιοθήκη θα πρέπει να επαυξήσουμε τα ήδη υπάρχοντα διαδραστικά αντικείμενα ενός ήδη υπάρχον toolkit (π.χ. Java Swing) ή τη δημιουργία τους από την αρχή, έτσι ώστε να μπορούν να υποστηρίξουν πολλαπλούς χρήστες. Κοιτάζοντας τα πιο δημοφιλή groupware toolkits GroupKit [12] και MAUI [13] για σύγχρονη ή real-time συνεργασίας, βλέπουμε την ανάγκη για συγχρονισμό και την επικοινωνία των χρηστών. Επίσης στο MAUI έχουμε υποστήριξη για group και workspace awareness, δηλαδή ο χρήστης έχει τη δυνατότητα να βλέπει τη δραστηριότητες εκτελούν οι συνεργάτες τους, καθώς και ποιο τμήμα της εφαρμογής βλέπουν. Η δημιουργία συνεργατικών εφαρμογών με τη χρήση toolkits, μας περιορίζει ως προς το περιβάλλον/πλατφόρμα που έχει υλοποιηθεί η εργαλειοθήκη, καθώς και στο ότι δεν υποστηρίζεται ή είναι αρκετά δύσκολο να υλοποιηθούν custom widgets, τα widgets που υποστηρίζονται συνήθως είναι form based(κουμπιά, διάλογοι, λίστες κτλ.).

2.3.2.3 Frameworks

Framework είναι ένα περιβάλλον χτισμένο πάνω σε κάποια πλατφόρμα (π.χ. Java, C++, android, κτλ.). Στόχος ενός framework είναι να παρέχει συγκεκριμένου σκοπού λειτουργικότητα (API) στους προγραμματιστές, διευκολύνοντας την ανάπτυξη εφαρμογών. Όταν ένα framework εστιάζει στη παροχή service για τη διαχείριση των συσκευών εισόδου/εξόδου, καθώς και για την επικοινωνία το αποκαλούμε με τον όρο middleware ή middleware framework. Όσον αφορά τώρα τα συνεργατικά συστήματα, μπορούμε να βρούμε αρκετά frameworks και middlewares, που προσπαθούν να επιλύσουν προβλήματα που έχουν να κάνουν με διάφορες μορφές ετερογένειας, όπως χρηστών (π.χ. κινητοί χρήστες), περιβάλλον (desktop, web ή mobile), συγχρονισμού (σύγχρονα ή ασύγχρονα συστήματα), κτλ. Παρακάτω θα γίνει αναφορά σε

κάποια frameworks και middlewares, καθώς και πως προσεγγίζουν τα προαναφερθέν προβλήματα.

Με βάση τα frameworks που μελετήσαμε (Artefact [14], COPSE-Web [15], Promondia [16], Agilo [17] και Pocket DreamTeam [18]), παρατηρούμε ότι οι περισσότερες δουλειές, έχουν να κάνουν με web-based συνεργατικά συστήματα(Artefact, COPSE και Promondia), μια έχει να κάνει με mobile users (Pocket DreamTeam) και μια πιο γενικού σκοπού (Agilo java-based platform). Ακόμα βλέπουμε ότι όλα τα frameworks εκτός το Artefact, δεν έχουν τι δυνατότητα επέκτασης του API τους με custom widgets. Παρατηρούμε ακόμα ότι μόνο το COPSE παρέχει API, για τη διαχείριση των λειτουργιών που έχουν να κάνουν με επικοινωνία και συγχρονισμό, πιο συγκεκριμένα παρέχει ένα server-side service-driven API. Από πλευρά awareness τώρα βλέπουμε ότι μόνο το COPSE, δίνει τη δυνατότητα χρήσης awareness components πάνω σε διαμοιραζόμενο περιβάλλον εργασίας (workspace). Τέλος όλα τα προαναφερθέν frameworks υποστηρίζουν, τη δημιουργία σύγχρονων συνεργατικών συστημάτων, καθώς και όλα εκτός το Agilo παρέχουν μηχανισμό για την διαχείριση sessions.

2.3.2.4 Middleware

Καθώς προχωράει η τεχνολογία προσφέροντας στους χρήστες τη δυνατότητα, να έχουν πρόσβαση στο internet από παντού και με διάφορους τρόπους (π.χ. Wi-Fi, δίκτυο κινητής τηλεφωνίας, κτλ.), η ανάγκη για συνεργατικά συστήματα που υποστηρίζουν κινητούς χρήστες αυξάνονται. Γι' αυτό το λόγο παρατηρούμε ότι οι περισσότερες δουλειές, πάνω σε middlewares που έχουν να κάνουν με το CSCW προσπαθούν και προτείνουν λύσεις πάνω στο πρόβλημα αυτό. Τα MoCA [19], U-Media [20], MUIT [21], SYNG [22] και SOMU [23], είναι κάποια από τα middlewares που μελετήσαμε. Παρακάτω θα δούμε της δυνατότητες τους. Το MoCA προσφέρει στον προγραμματιστή ευελιξία, σε θέματα που έχουν να κάνουν με επικοινωνία και συγχρονισμό(σύγχρονη ή ασύγχρονη, message ή sharing oriented), αυτό επιτυγχάνετε με τη παροχή API για τον Client και για τον Server. Ακόμα παρέχει μηχανισμούς για τη διαχείριση sessions. Επίσης το "location interface service" μας επιτρέπει να έχουμε location awareness. Με τα U-Media και MUIT τώρα μπορούμε να αναπτύξουμε μόνο ασύγχρονα συνεργατικά συστήματα, σε σχέση με την ευελιξία του MoCA. Για την επίτευξη της συνεργασίας στο U-Media γίνετε χρήση ενός event-based API βασισμένο στο android, ενώ το MUIT δεν προσφέρει κάποιο API. Ως προς το awareness τώρα βλέπουμε ότι και τα δύο υποστηρίζουν κάποια μορφή context awareness, στο U-Media έχει να κάνει με τη γεωγραφική θέση του χρήστη (location)

καθώς και πληροφορίες και εν δήξεις αισθητήρων που παρέχει η εκάστοτε συσκευή, ενώ στο MUIT έχει να κάνει με την κινητικότητα(mobility). Το U-Media είναι το μόνο middleware από αυτά που μελετήσαμε, που προσφέρει τη δυνατότητα προσαρμογής του UI. Η προσαρμογή αυτή γίνετε ανάλογα με το περιβάλλον/θέση που βρίσκεται ο χρήστης καθώς και με το αν κινείται ή όχι. Το MUIT από την άλλη είναι το μόνο middleware όπου ο προγραμματιστής έχει τη δυνατότητα να προσθέσει καινούργια ή να επαυξήσει υπάρχον widgets. Το SYNG είναι ένα middleware για σύγχρονα συνεργατικά συστήματα, με τη δυνατότητα για client-server ή peer-to-peer επικοινωνία, με τη χρήση του API που παρέχει από τη πλευρά του client/peer. Επίσης διαθέτη μηχανισμό για τη διαχείριση sessions. Τέλος στο SOMU παρατηρούμε ότι η επικοινωνία καθώς και το συνεργατικό σύστημα βασίζεται στα δίκτυα τύπο MANET(Mobile Ad-Hoc Networks), όπου ο χρήστης είναι και client και server. Η επικοινωνία, ο συγχρονισμός και ο διαμοιρασμός αρχείων γίνεται με την χρήση του API που παρέχει και βασίζεται στα web-service.

2.3.2.5 Περιβάλλοντα συνεργατικών καθηκόντων

Το περιβάλλον (development environment) είναι ένα υπολογιστικό σύστημα, στο οποίο ο προγραμματιστής έχει τη δυνατότητα να αναπτύξει και να εκτελέσει ένα πρόγραμμα/εφαρμογή. Υπάρχουν διάφοροι τύποι περιβαλλόντων (local, development, integration, test/QA, stage/production), ενδιαφέρον όμως δείχνουν τα περιβάλλοντα βασισμένα σε τεχνολογίες νέφους (cloud-based environments), για τις υπηρεσίες που παρέχουν στη δημιουργία συνεργατικών συστημάτων. Παρακάτω θα γίνει αναφορά σε κάποια από αυτά, μαζί με κάποιες από τις δυνατότητες που παρέχουν.

Τα περιβάλλοντα που θα περιγράψουμε τώρα, βασίζονται στη real-time επικοινωνία, παρέχοντας ένα αξιόπιστο API. Το Pubnub και το pusher είναι κάποια από αυτά, και τα δύο υποστηρίζουν την ανάπτυξη εφαρμογών πάνω στις πιο γνωστές μοντέρνες πλατφόρμες. Επίσης και τα δύο παρέχουν μηχανισμούς για ειδοποιήσεις για κινητές συσκευές(android, iOS). Μηχανισμούς για ειδοποιήσεις παρέχει και το firebase, ένα περιβάλλον για τη δημιουργία βάσεων δεδομένων πραγματικού χρόνου. Τέλος το app engine της google δίνει τη δυνατότητα στους προγραμματιστές τη δημιουργία ασύγχρονων εφαρμογών. Για real-time εφαρμογές η google παρέχει το real-time API.

2.4 Απαιτήσεις συνεργατικών καθηκόντων στο χώρο

Σε αυτό το σημείο θα αναφερθούμε σε βιβλιοθήκες-APIs και εργαλεία που έχουν να κάνουν με χάρτες, την παρουσίαση και διαχείριση χωρικών δεδομένων πάνω στις εξής πλατφόρμες web (JavaScript), android και desktop (java). Πριν ξεκινήσουμε με τα APIs θα γίνει μια αναφορά σε όρους που αφορούν τα χωρικά δεδομένα και που θα τους συναντήσουμε και παρακάτω. Συγκεκριμένα, θα εξηγήσουμε τι είναι το γεωγραφικό σύστημα συντεταγμένων (coordinate system), το Map layer ή layer, το WMS (Web Map Service) και τέλος τι είναι το WFS (Web Feature Service).

Το γεωγραφικό σύστημα συντεταγμένων είναι ένα σύστημα συντεταγμένων που δίνει τη δυνατότητα κάθε σημείο της γης να προσδιοριστεί από ένα σύνολο από αριθμούς, γράμματα ή σύμβολα. Οι συντεταγμένες συχνά επιλέγονται έτσι ώστε ένας από τους αριθμούς να αντιπροσωπεύει τη κατακόρυφη θέση και δύο ή τρεις από τους αριθμούς την οριζόντια θέση. Η κοινή επιλογή συντεταγμένων είναι το γεωγραφικό πλάτος (latitude), το γεωγραφικό μήκος (longitude) και το υψόμετρο (elevation ή altitude). Map Layer είναι ο μηχανισμός που χρησιμοποιείτε για την αναπαράσταση/προβολή γεωγραφικών δεδομένων. Ένα layer αναφέρεται σε συγκεκριμένα δεδομένα, καθορίζοντας τον τρόπο με τον οποίο θα προβάλλονται χρησιμοποιώντας σύμβολα και ετικέτες κειμένου. Το WMS είναι ένα standard πρωτόκολλο για την παροχή georeferenced εικόνες χαρτών, που έχουν δημιουργηθεί από ένα MapServer που κάνει χρήση δεδομένων από μια GIS (Geographic Information System) βάση δεδομένων. Με τον όρο georeferenced εννοούμε ότι το εσωτερικό σύστημα συντεταγμένων του χάρτη ή κάποιας αεροφωτογραφίας μπορεί να συσχετιστεί με το σύστημα εδάφους των γεωγραφικών συντεταγμένων. Οι αντίστοιχοι μετασχηματισμοί συντεταγμένων συνήθως αποθηκεύονται στο αρχείο της εικόνας (π.χ. GeoPDF και GeoTIFF), αν και υπάρχουν πολλοί πιθανοί μηχανισμοί για την εφαρμογή του. Το WFS είναι ένα πρότυπο που παρέχει μια διεπαφή που επιτρέπει τα αιτήματα που αφορούν γεωγραφικά/χωρικά χαρακτηριστικά από το web να εκτελούν ανεξαρτήτου πλατφόρμας κλήσης. Τα γεωγραφικά χαρακτηριστικά θεωρούνται και ως ο πηγαίος κώδικας πίσω από ένα χάρτη, σε σχέση με το WMS το οποίο επιστρέφει κάθε φορά μόνο μια εικόνα ως αποτέλεσμα. Με τη χρήση του WFS έχουμε τις εξής δυνατότητες:

- Να λάβουμε ή να κάνουμε ερώτημα (query) χαρακτηριστικών (features) με βάση χωρικών και μη περιορισμών
- Να δημιουργήσουμε ένα καινούργιο feature

- Να διαγράψουμε κάποιο feature
- Να ενημερώσουμε κάποιο feature

2.4.1 Βιβλιοθήκες για το Web

Οι βιβλιοθήκες-APIs που μελετήσαμε είναι η εξής:

- Google Maps
- Leaflet
- MapBox
- MapQuest
- OpenLayers

Τα κοινά χαρακτηριστικά που έχουν οι παραπάνω βιβλιοθήκες έχουν να κάνουν με: α) τη δυνατότητα τύπωσης σχημάτων πάνω στο χάρτη (γραμμές, πολύγωνα, σημεία, κτλ.), β) τη χρήση απλών ή συγκεκριμένου σκοπού popups, γ) υποστήριξη των πιο δημοφιλών τύπων γεωγραφικών πληροφοριών/δεδομένων (KML, GeoJSON, GML, κτλ.). Οι βιβλιοθήκες google maps, mapbox και MapQuest παρέχουν επίσης service όπως routing/directions, geocoding και distance matrix.

Για την ανάπτυξη της εφαρμογής που θα περιγράψουμε παρακάτω, στη πλατφόρμα του web επιλέξαμε τη χρήση του API της βιβλιοθήκης OpenLayers 3. Οι λόγοι που την επιλέξαμε έναντι των άλλων είναι: οι Google Maps και MapQuest δεν υποστηρίζουν τη χρήση WMS (Web Map Service, παρακάτω θα δούμε τι είναι και πως μπορούμε να δημιουργήσουμε ένα) layers, για να χρησιμοποιήσουμε/εκμεταλλευτούμε της δυνατότητες της MapBox θα έπρεπε να προχωρήσουμε σε πληρωμή, καθώς το free/open-source κομμάτι του API της δεν κάλυπτε της ανάγκες μας, και τέλος τη leaflet την αφήσαμε επειδή η OpenLayers έχει παραπάνω δυνατότητες.

Οι δυνατότητες που έχουμε από τη χρήση του API της OpenLayers είναι οι παρακάτω:

- Προβολή/παρουσίαση map tiles, διανυσματικά δεδομένα (vector data) και markers από οποιαδήποτε πηγή
- Χρήση γεωγραφική πληροφορίας κάθε τύπου
- Χρησιμοποιεί WebGL, Canvas 2D και HTML5 για την εύκολη προσαρμογή του χάρτη και της πληροφορίας που θέλουμε να προσθέσουμε
- Χρήση της CSS για τον έλεγχο του style που θα έχουν τα γραφικά αντικείμενα ελέγχου του χάρτη (π.χ. layer switcher, context menu κτλ.).

2.4.2 Βιβλιοθήκες για Android

Οι βιβλιοθήκες που εξετάσαμε για την πλατφόρμα Android, είναι η Nutiteq και η Glob3Mobile. Την nutiteq την αποκλείσαμε γρήγορα καθώς το open-source κομμάτι του API της δεν επαρκούσε για της ανάγκες της εφαρμογής μας. Από την άλλη τώρα η Glob3Mobile είναι μια open-source βιβλιοθήκη/SDK για τη δημιουργίας 2D, 2.5D και 3D χαρτών. Κάποιες από τις δυνατότητες που μας προσφέρει είναι οι εξής:

- Χρήση δεδομένων πραγματικού χρόνο(real-time)
- Ενσωμάτωση/χρήση κάθε είδους format και μεγέθους
- Υποστήριξη κάθε τύπου μοντέλων για τη 3D αναπαράσταση του εδάφους (π.χ. bil)
- Raster layers (WMS, MapBox, CortoDB, MapQuest, Bing, κτλ.)
- Κάθε τύπο διανυσματικής προβολής πληροφορίας (GeoJSON, BSON, shp, KML, κτλ.)
- Ακόμα μπορούμε να προσθέσουμε/ζωγραφίσουμε σημεία (markers), γραμμές, κτλ πάνω στο χάρτη.

2.4.3 Βιβλιοθήκες για Desktop

Ως προς τη πλατφόρμα Desktop (Java), είδαμε τις βιβλιοθήκες JXMapKit και Nasa World Wind. Στη JXMapKit είδαμε/δουλέψαμε με τους χάρτες που παρέχει το Open Street Map, φορτώνοντας κομμάτια του χάρτη είτε από το κεντρικό τους server είτε τοπικά, χρησιμοποιώντας τα jTiledownloader (δεν υποστηρίζετε πια από τους χάρτες του OSM) και TileDownloader. Χρησιμοποιήσαμε ακόμα το service που παρέχει το OSM για routing με τη βοήθεια της βιβλιοθήκης osm2po, επίσης χρησιμοποιήσαμε και τη βιβλιοθήκη metadata extractor της apache για να προσθέσουμε geotagged εικόνες πάνω στο χάρτη και το αντίστροφο.

Το World Wind είναι μια βιβλιοθήκη για 3D χάρτες, με δυνατότητες όμοιες με εκείνες του OpenLayers. Από τις δυνατότητες που έχει εμείς είδαμε τη χρήση των markers, πως μπορούμε να ζωγραφίσουμε σχήματα, καθώς και πως υποστηρίζει/υλοποιεί τη χρήση των WMS layers. Αν και αρχικά είχαμε ξεκινήσει την υλοποίηση της εφαρμογής, και με τη χρήση αυτής της βιβλιοθήκης (μαζί με την android και την web), αλλά την αφήσαμε λόγο της πολυπλοκότητας του API της, σε σχέση με εκείνο της OpenLayers και για το λόγο ότι είχαμε κάποιες παραπάνω δυνατότητες στη πλατφόρμα του web έναντι της Desktop, όσο αφορά την ενσωμάτωση visualizations στο ίδιο παράθυρο με τον χάρτη.

2.5 Servers

Σε επίπεδο Server, χρησιμοποιήσαμε τον apache tomcat για την επίτευξη του συγχρονισμού, καθώς και τον GeoServer για τη δημιουργία και διαμοιρασμό χωρικών δεδομένων. Παρακάτω θα δούμε τι είναι ο κάθε server καθώς και τον τρόπο/λόγο που τον χρησιμοποιήσαμε στα πλαίσια της εν λόγο πτυχιακής.

2.5.1 Tomcat

Ο apache tomcat ή tomcat είναι ένας open-source web server πάνω στον οποίο έχουμε τη δυνατότητα να τρέχουμε Java EE (servlets, jsp, java EL και web sockets) εφαρμογές. Στα πλαίσια αυτής της εργασίας χρησιμοποιήσαμε κάποια service (στα οποία θα αναφερθούμε παρακάτω) για την επίτευξη και διατήρηση του συγχρονισμού.

2.5.2 GeoServer

Ο GeoServer είναι ένα Java-based λογισμικό server, που επιτρέπει στους χρήστες να δουν αλλά και να επεξεργαστούν χωρικά δεδομένα. Χρησιμοποίει/βασίζετε σε ανοιχτά πρότυπα που ορίζονται από το Open Geospatial Consortium (OGC). Ο GeoServer προσφέρει μεγαλύτερη ευελιξία στη δημιουργία χαρτών και στην ανταλλαγή δεδομένων. Επίσης μας δίνει τη δυνατότητα δημιουργίας και προβολής/διαμοιρασμού των δικών μας χωρικών δεδομένων/πληροφορίων. Αξιοποιώντας το πρότυπο WMS έχουμε τη δυνατότητα δημιουργίας των δικών μας layers, παρακάτω θα δούμε πως μπορούμε να δημιουργήσουμε το δικό μας WMS. Ακόμα ο GeoServer είναι σύμφωνος με την υπηρεσία Web Feature Service (WFS) επιτρέποντας την πραγματική ανταλλαγή και επεξεργασία των δεδομένων που χρησιμοποιούνται για τη δημιουργία των χαρτών.

3 Περιπτώσεις χρήσης

Το πεδίο εφαρμογής αυτής της πτυχιακής αφορά την ανάπτυξη ενός σύγχρονου συνεργατικού συστήματος για κατανεμημένους χρήστες, με έμφαση στο διαμοιρασμό/αξιοποίηση χωρικών δεδομένων. Αυτό είναι χρήσιμο μιας και μπορεί να εφαρμοστεί σε σειρά πεδίων εφαρμογής/να καλύψει σειρά περιπτώσεων χρήσης όπως αυτές που θα δούμε παρακάτω.

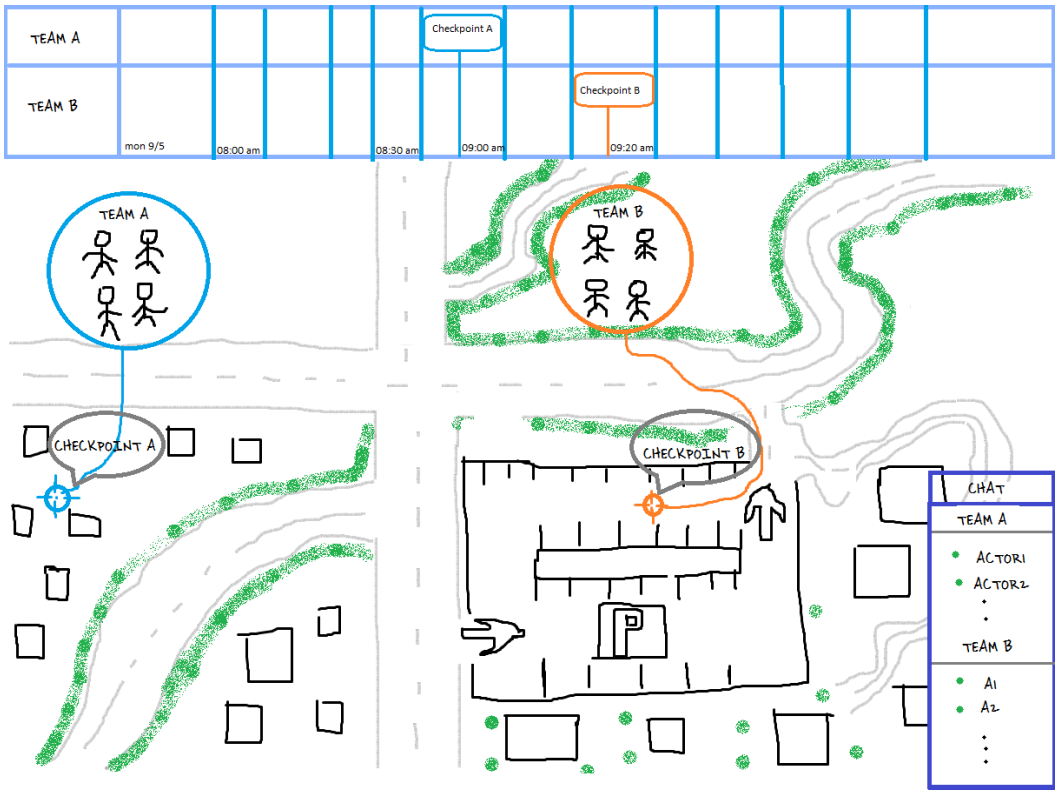
3.1 Απαιτήσεις και πρωτότυπα

Το βασικό σενάριο στο οποίο εστιάζουμε αφορά ένα multi-platform multi-user σύγχρονο σύστημα με έμφαση στα χωρικά δεδομένα, για τη διαχείριση και εν εκτελέσει εποπτεία τακτικών αποστολών. Η διευκόλυνση της εποπτείας τακτικών αποστολών προσφέρει στον παρατηρητή/συντονιστή μεγαλύτερη αίσθηση ως προς τις κινήσεις των εποπτευόμενων, καθώς και για την εξέλιξη/πρόοδο των καθηκόντων που εμπλέκονται οι εποπτευόμενοι. Στα πλαίσια αυτού οι χρήστες του συστήματος μπορεί να εμπλακούν χρησιμοποιώντας διαφορετικές τερματικές συσκευές και πλατφόρμες. Παρόλα αυτά, η εστίαση της παρούσας πτυχιακής αφορά στην υποστήριξη χρηστών που δρουν διαμέσου ενός web-based setting καθώς και ενός mobile-based setting.

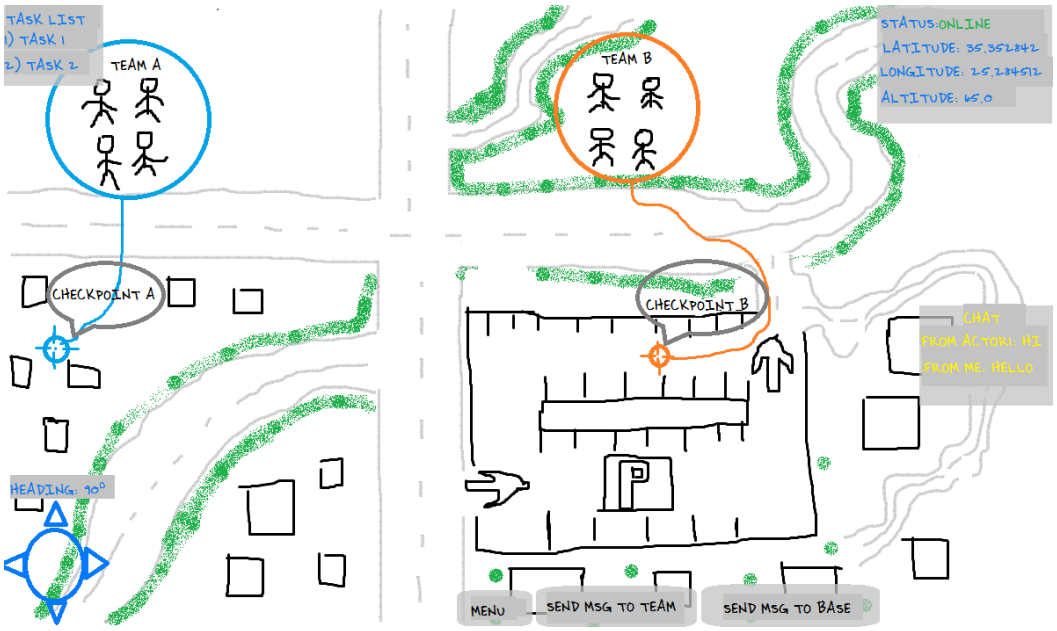
Στην Εικόνα 2 παρουσιάζεται το σκαρίφημα της διεπαφής του σεναρίου, την οποία μπορούμε να την χωρίσουμε σε 3 τμήματα: α) το χρονοδιάγραμμα (πάνω τμήμα της εικόνας) πάνω στο οποίο παρουσιάζονται τα καθήκοντα που προσθέτει ο χρήστης του web σύμφωνα με την ημερομηνία και ώρα που τους έχει οριστεί, β) το τμήμα που περιέχει τον χάρτη και την παρουσίαση/προβολή των χωρικών δεδομένων (android-users και τα καθήκοντα) και γ) το τμήμα του chat στο οποίο βλέπουμε τους χρήστες που είναι διαθέσιμοι για chat. Αντίστοιχα για το χρήστη του Android το σκαρίφημα της διεπαφής εμφανίζεται στην Εικόνα **3**, και την οποία μπορούμε να την χωρίσομε στα εξής τμήματα: α) λίστα καθηκόντων του χρήστη (πάνω αριστερό τμήμα της εικόνας), β) στοιχεία του χρήστη (πάνω δεξιό τμήμα), όπως η κατάσταση της σύνδεσης του και η θέση του σε συντεταγμένες, γ) το τμήμα που περιέχει τον χάρτη (όπως και στο σκαρίφημα της web εφαρμογής), δ) το τμήμα με την τρέχουσα συνομιλία του χρήστη με κάποιον άλλο χρήστη (το τμήμα με τα κίτρινα γράμματα στην αριστερή πλευρά), ε) πυξίδα (κάτω αριστερό τμήμα) και στ) στο τμήμα αυτό (κάτω μέρος της εικόνας) έχουμε ένα κουμπί το οποίο εμφανίζει ένα popup

menu, και δύο κουμπιά για την αποστολή “voice to text” μηνύματος στους χρήστης της ομάδας του και στον χρήστη του web “base”.

Στο κεφάλαιο 5 θα γίνει μια πιο αναλυτική περιγραφή στον τρόπο λειτουργίας των τμημάτων των δύο σκαριφημάτων.



Εικόνα 2: webMap mockup



Εικόνα 3: androidMap mockup

Το σύστημα υιοθετεί τους εξής κανόνες: α) την ύπαρξη ενός χρήστη που θα κατέχει το ρόλο του επόπτη/συντονιστή (υλοποίηση/υποστήριξη από τη web εφαρμογή), με δυνατότητες επιτήρησης και συντονισμού των υπάρχων ομάδων και β) την ύπαρξη ν-χρηστών που θα έχουν το ρόλο του εποπτευόμενου μέλους (υλοποίηση/υποστήριξη από την android εφαρμογή) με δυνατότητα δημιουργίας ομάδων, όπου θα ακολουθούν τις οδηγίες του χρήστη-συντονιστή. Παρακάτω αναλύεται ο ρόλος των αντικειμένων πεδίου που χρειάζεται να επιστρατευτούν.

3.1.1 Χάρτης

Ο χάρτης αφορά ένα domain-specific container που αποτυπώνει τόσο τη θέση των χρηστών στο χώρο όσο και τα καθήκοντα τους. Ανάλογα με το σενάριο μπορεί να αποδοθεί ως ένα γραφικό δισδιάστατο ή και τρισδιάστατο χάρτη, που ως εκ τούτου διακρίνεται από το γεωγραφικό μήκος και πλάτος, καθώς και το υψόμετρο (αν μιλάμε για τρισδιάστατη απεικόνιση του χάρτη).

3.1.2 Actor

Ο actor αφορά ένα αντικείμενο το οποίο χειρίζονται οι χρήστες του συστήματος που μετακινούνται (mobile). Το αντικείμενο αποτυπώνεται στο πλαίσιο του ‘χάρτη’ και φέρει συγκεκριμένα χαρακτηριστικά. Συγκεκριμένα, το γεωγραφικό μήκος και πλάτος (longitude και latitude αντίστοιχα) προσδιορίζουν τη πραγματική τρέχουσα θέση ενός χρήστη, μια ετικέτα κειμένου (text label) υποδεικνύει το όνομα και την ομάδα που ανήκει ο κάθε χρήστης. Η θέση

του χρήστη-actor ενημερώνεται από το GPS που διαθέτει η συσκευή του χρήστη (συσκευές που δεν διαθέτουν GPS δεν είναι κατάλληλες για τη mobile εφαρμογή του συστήματος μας).

3.1.3 Καθήκον

Το καθήκον ή αλλιώς 'task' αφορά 'εργασία' που έχει να εκτελέσει μια ομάδα. Η υλοποίηση του σηματοδοτείται από ένα γραφικό δισδιάστατο αντικείμενο (δηλ. γεωγραφικό μήκος και πλάτος), του οποίου η κύρια απεικόνιση γίνετε με ένα εικονίδιο πάνω στο χάρτη και στις δύο πλατφόρμες. Για την εφαρμογή στο web τα καθήκοντα απεικονίζονται πάνω σε ένα timeline από την πλευρά του web, ενώ στο android το συναντάμε ως μια εγγραφή στη λίστα καθηκόντων του χρήστη. Τα χαρακτηριστικά που μπορεί να έχει ένα καθήκον είναι μια περιγραφή, την ομάδα που της έχει ανατεθεί ως καθήκον και τέλος την τρέχουσα κατάσταση του (επιτυχές, απέτυχε και εκκρεμεί).

3.1.4 Ρόλοι χρηστών

Ο ρόλος που μπορεί να έχει ένας χρήστης της εφαρμογής μας, εξαρτάτε εν μέρη από την πλατφόρμα υλοποίησης (android και web). Ο χρήστης μπορεί να έχει έναν από τους εξής ρόλους: α) απλός χρήστης (σε android), β) αρχηγός/επικεφαλής (σε android) και γ) επόπτης/supervisor (σε web). Παρακάτω θα γίνει αναφορά στις δυνατότητες που έχει ο κάθε χρήστης σύμφωνα με τον ρόλο του.

Όπως αναφέρθηκε και παραπάνω ένας χρήστης android συσκευής, μπορεί να είναι είτε απλός χρήστης είτε επικεφαλής της ομάδας του, παρακάτω θα γίνει αναφορά στις δυνατότητες που έχει, ανάλογα με τον ρόλο που έχει. Οι δυνατότητες καθώς και οι ρόλοι μπορεί να επεκταθούν, ανάλογα με της απαιτήσεις της εφαρμογής.

Οι δυνατότητες που έχει ένας 'απλός χρήστης' είναι οι εξής:

- Προβολή των καθηκόντων που έχουν ανατεθεί στην η ομάδα του
- Προβολή της θέσης των υπόλοιπων χρηστών (android)
- Chat με τα μέλη της ομάδας και με τον χρήστη της web εφαρμογής

Οι δυνατότητες του 'απλού χρήστη' μπορούμε να πούμε ότι είναι οι ελάχιστες δυνατές ή οι βασικές δυνατότητες που μπορεί να προσφέρει το σύστημα μας. Ο 'επικεφαλής/αρχηγός ομάδας' πέρα από τις βασικές δυνατότητες, μπορεί να αλλάξει την τρέχουσα κατάσταση ενός καθήκοντος (succeed, failed, pending).

Ο ρόλος του ‘επόπτη’ από την άλλη βασίζεται στην εποπτεία, αλλά και στο συντονισμό των κινητών χρηστών (android users). Γνωρίζοντας αυτά μπορούμε να πούμε ότι οι βασικές δυνατότητες ενός web user είναι οι εξής:

- Προβολή όλων των ομάδων (θέση όλων των χρηστών πάνω στο χάρτη)
- Προβολή των καθηκόντων κάθε ομάδας
- Προσθήκη/ανάθεση καθήκοντος σε ομάδα
- Ενημέρωση τρέχουσας κατάστασης κάποιου καθήκοντος
- Chat με όλους τους χρήστες ανεξαρτήτου ομάδας

3.2 Πλατφόρμες υλοποίησης

Οι πλατφόρμες που επιλέχθηκαν για την υλοποίηση της εφαρμογής που περιγράφηκε προηγουμένως, είναι η android και η web (browser-based apps). Το android το επιλέξαμε για το λόγο, ότι είναι μια από της πιο δημοφιλής πλατφόρμα για κινητούς χρήστες, αλλά και για την ποικιλία σε open-source βιβλιοθήκες και toolkits, για την ανάπτυξη εφαρμογών, καθώς και για την υποστήριξη κάθε τύπου επικοινωνίας. Από την άλλη η επιλογή του web στηρίχθηκε, στο ότι γίνετε ολοένα και πιο δημοφιλής, καθώς και για την παροχή υψηλού επιπέδου APIs για τη χρήση χωρικών δεδομένων (π.χ. google maps, openlayers, κτλ.) και όχι μόνο.

Οι επιπλοκές/δυσκολίες που αντιμετωπίσαμε στην ανάπτυξη της εφαρμογής, βρίσκονται κυρίως στην ανομοιογένεια των βιβλιοθηκών και των toolkits, στην ανάπτυξη της εφαρμογής (χρήση 3D χαρτών στο android, ενώ 2D στο web). Επιπλοκές είχαμε ακόμα και στον τρόπο θα γίνετε η ανταλλαγή μηνυμάτων για την επίτευξη του συγχρονισμού και της επικοινωνίας γενικότερα λόγο της ετερογένειας στις γλώσσες προγραμματισμού (το android στηρίζεται στη java, ενώ στο web έχουμε JavaScript).

3.3 Απαιτήσεις συγχρονισμού

Οι πληροφορίες που ανταλλάσσονται δια μέσου του συστήματος που υλοποιήσαμε, έχουν να κάνουν με τη μετάδοση της θέσης ενός χρήση ή καθήκοντος (δηλαδή συντεταγμένες), καθώς και πληροφορίες που έχουν να κάνουν με το chat (από, προς, κείμενο). Λόγω του ότι οι πληροφορίες αυτές είναι θεμελιώδεις για τέτοιου είδους συστήματα, μπορούμε να πούμε ότι δεν υπάρχει διαφοροποίηση ανάλογα την περίπτωση. Η δομή που υιοθετήσαμε για την ανταλλαγή της εκάστοτε πληροφορίας στηρίζεται, στη μετάδοση XML-Based μηνυμάτων.

Οι τεχνολογικής φύσης δεσμεύσεις που προκύπτουν, έχουν να κάνουν με τη δομή/τρόπο που αποστέλλεται η πληροφορία, καθώς και στην ετερογένεια των πλατφορμών υλοποίησης. Έχοντας ως γνώμονα τις δεσμεύσεις αυτές, επιλέξαμε το HTTP Socket API το οποίο στηρίζετε στην ανταλλαγή μηνυμάτων/δομημένου κειμένου (message-sharing), για το συγχρονισμό έναντι των CORBA (Common Object Request Broker Architecture), RMI (Remote Method Invocation) και RPC (Remote Procedure Call). Για το λόγο ότι το CORBA δεν υπάρχει υποστήριξη για τη πλατφόρμα του Web, ενώ το RMI γιατί η μετάδοση της πληροφορίας στηρίζεται στην ανταλλαγή Java-Objects, ενώ στο RPC χρειάζεται/απαιτείται η χρήση Interface Description Language (IDL), για να είναι σε θέση να υποστηρίξει την επικοινωνία ανάμεσα σε διαφορετικές πλατφόρμες.

4 Προσέγγιση, αρχιτεκτονική, υλοποίηση

Στην παρούσα πτυχιακή εργασία υλοποιήθηκε ένα σύγχρονο συνεργατικό σύστημα, βασισμένο στο μοντέλο client-server, αξιοποιώντας μια προσέγγιση ανταλλαγής μηνυμάτων (message-sharing approach). Παρακάτω θα γίνει αναφορά σε τεχνολογίες και πρωτόκολλά που χρησιμοποιήθηκαν, από την πλευρά του Server και του client.

4.1 Servers

Ο server που χρησιμοποιήσαμε για τον συγχρονισμό είναι ο apache tomcat, πάνω στον οποίο υλοποιήθηκαν τα εξής service: signin, joinCollaborativeSession, getUpdates και commitUpdates, για την επικοινωνία και των συγχρονισμό της εφαρμογής μας. Ο server αξιοποιεί το πρωτόκολλο HTTP, για την διαχείριση και επικοινωνία με τους χρήστες. Ο τρόπος υλοποίησης του server δεν δημιουργεί κανένα περιορισμό στο τύπο των διεπαφών που έχουμε από την πλευρά του client. Τα ορίσματα που παίρνουν τα παραπάνω service έχουν ως εξής:

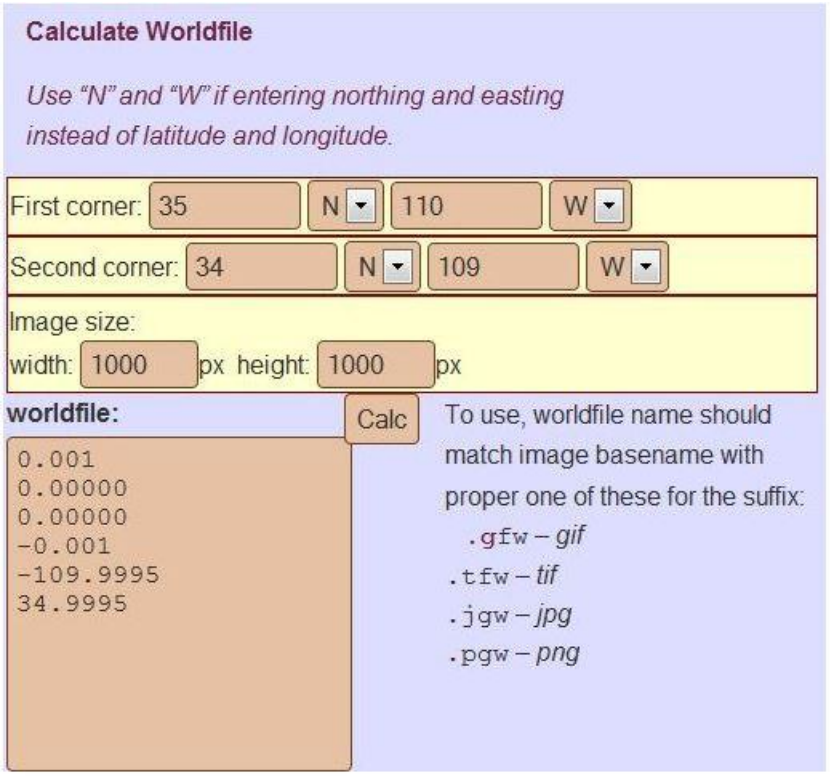
- signin(username, password)
- joinCollaborativeSession(sessionId)
- getUpdates(sessionId)
- commitUpdates(sessionId, className, propertyName, propertyValue, util_field)

Επίσης χρησιμοποιήσαμε τον GeoServer για τη δημιουργία και χρήση στην εφαρμογή μας τριών WMS layer, παρακάτω θα δούμε τη διαδικασία που ακολουθήσαμε για τη δημιουργία ενός WMS layer.

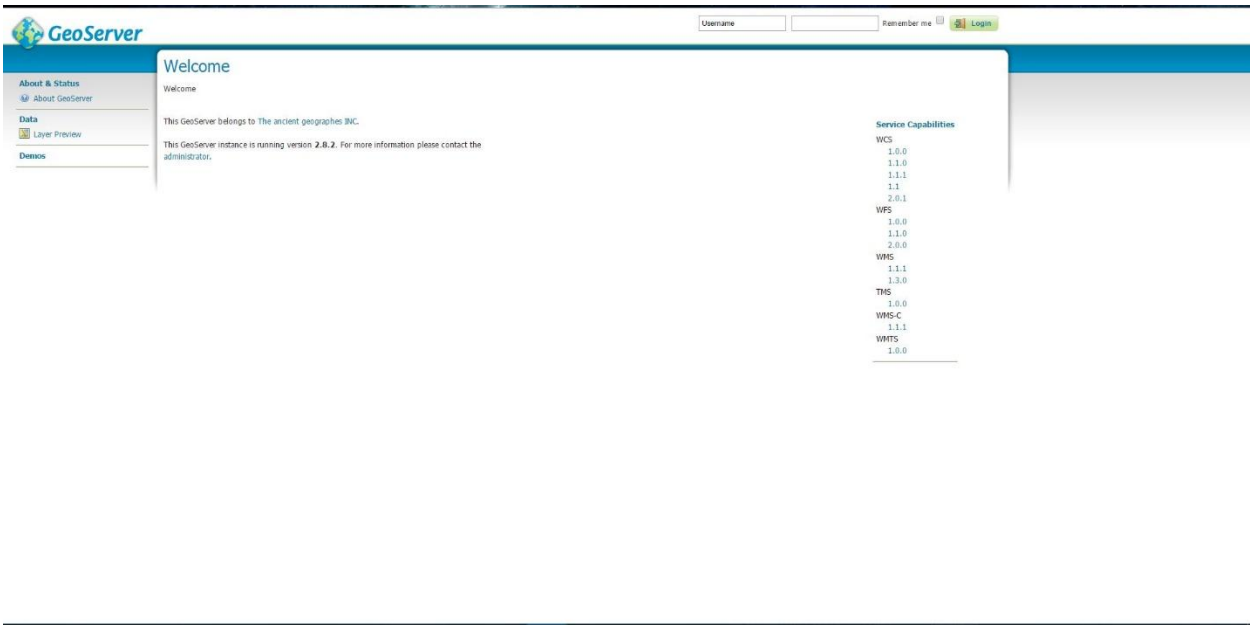
Για τη δημιουργία ενός WMS που θα εμφανίζει μια εικόνα στη θέση που επιθυμούμε θα πρέπει πρώτα να μετατρέψουμε την εικόνα μας σε worldimage, για να γίνει αυτό θα πρέπει να υπολογίσουμε και στη συνέχεια να δημιουργήσουμε ένα αρχείο τύπο ".jgw" (αν η εικόνα μας είναι jpeg, αν δεν είναι τότε θα πρέπει να επιλέξουμε τη σωστή κατάληξη μέσα από τη σελίδα που θα υπολογίσουμε τις τιμές του worldfile), για τον υπολογισμό των τιμών του αρχείου .jgw θα πάμε στο εξής site: http://freegeographytools.com/2009/online-worldfile-calculator (βλ. Εικόνα 4). Όσον αφορά τον calculator, στο πεδίο "first corner" βάζουμε τις συντεταγμένες της κάτω αριστερής γωνίας του σημείου/πλαισίου που θα εμφανίζετε η εικόνα μας, στο πεδίο "second corner" βάζουμε τις συντεταγμένες της πάνω δεξιά γωνίας, στη συνέχεια

συμπληρώνουμε τις διαστάσεις της εικόνας μας (σε pixel) και πατάμε calc, μόλις ολοκληρωθεί ο υπολογισμός αντιγράφουμε το αποτέλεσμα που βρίσκετε στο πεδίο "worldfile" στο αρχείο που φτιάξαμε προηγουμένως.

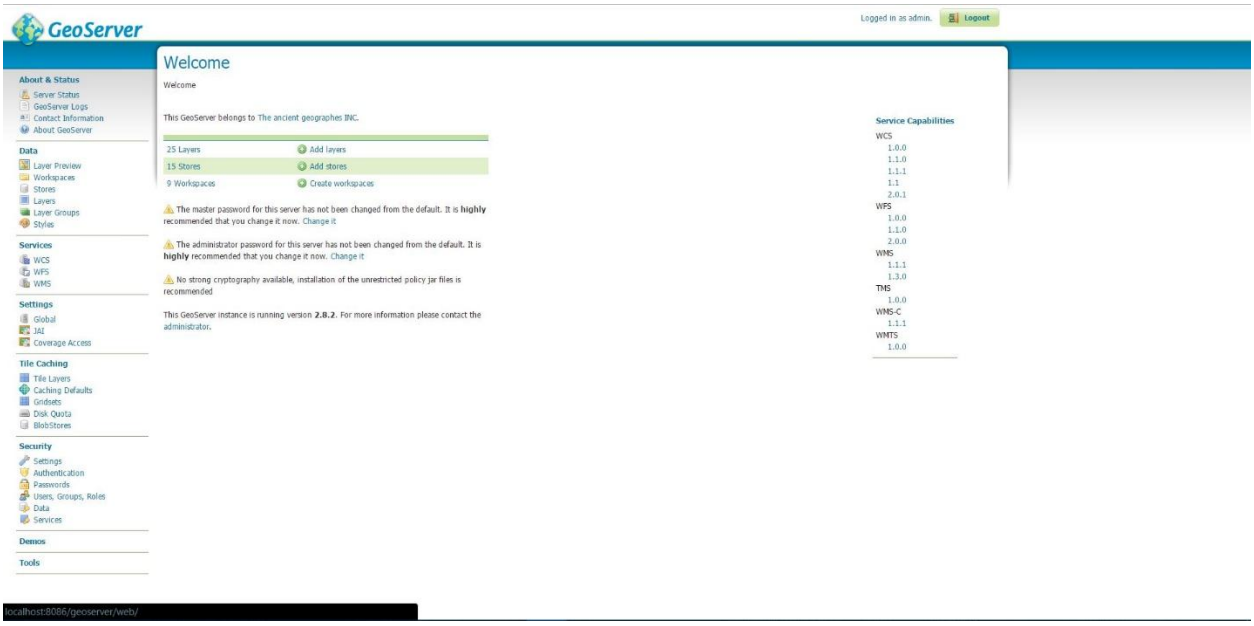
Από την πλευρά του server τώρα μόλις ξεκινήσει, μπορούμε να δούμε την αρχική του σελίδα πληκτρολογώντας στο browser http://localhost:8086/geoserver/web/ (η επιλογή της πόρτας γίνετε κατά τη διάρκεια της εγκατάστασης), μόλις πληκτρολογήσουμε τη διεύθυνση και πατήσουμε "enter" θα μας ανοίξει τη παρακάτω σελίδα (βλ. Εικόνα 5). Κάνοντας login ως admin θα δούμε στα αριστερά τη μπάρα με τις επιλογές που έχουμε (βλ. Εικόνα 6).



Εικόνα 4: worldfile calculator



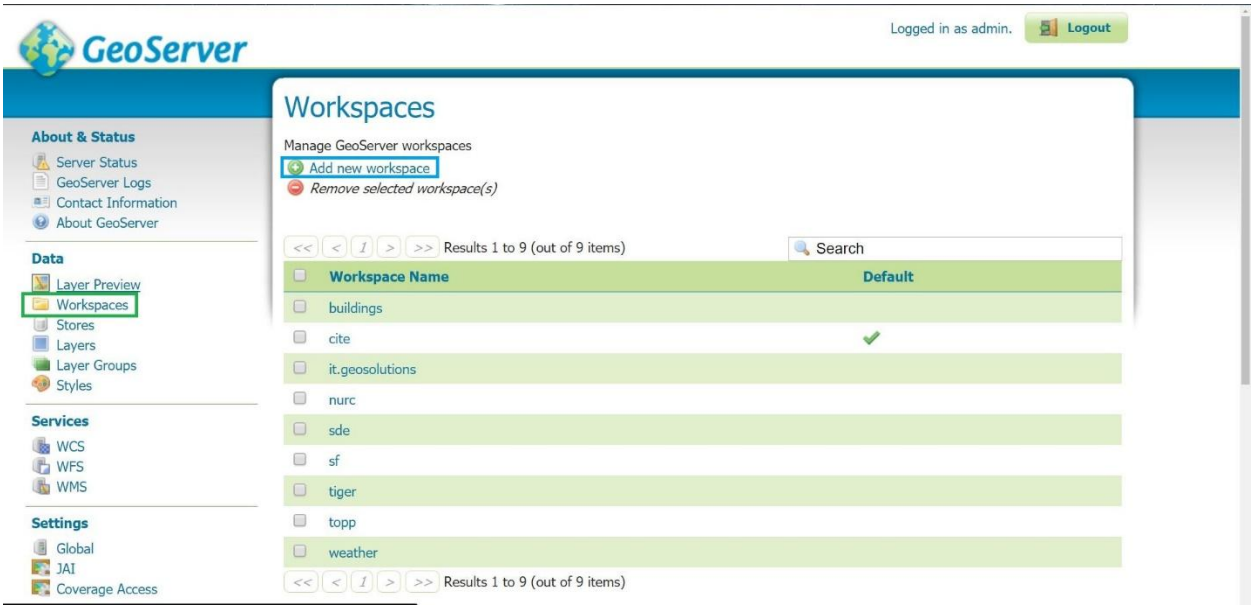
Εικόνα 5: GeoServer home page



Εικόνα 6: GeoServer logged in as admin

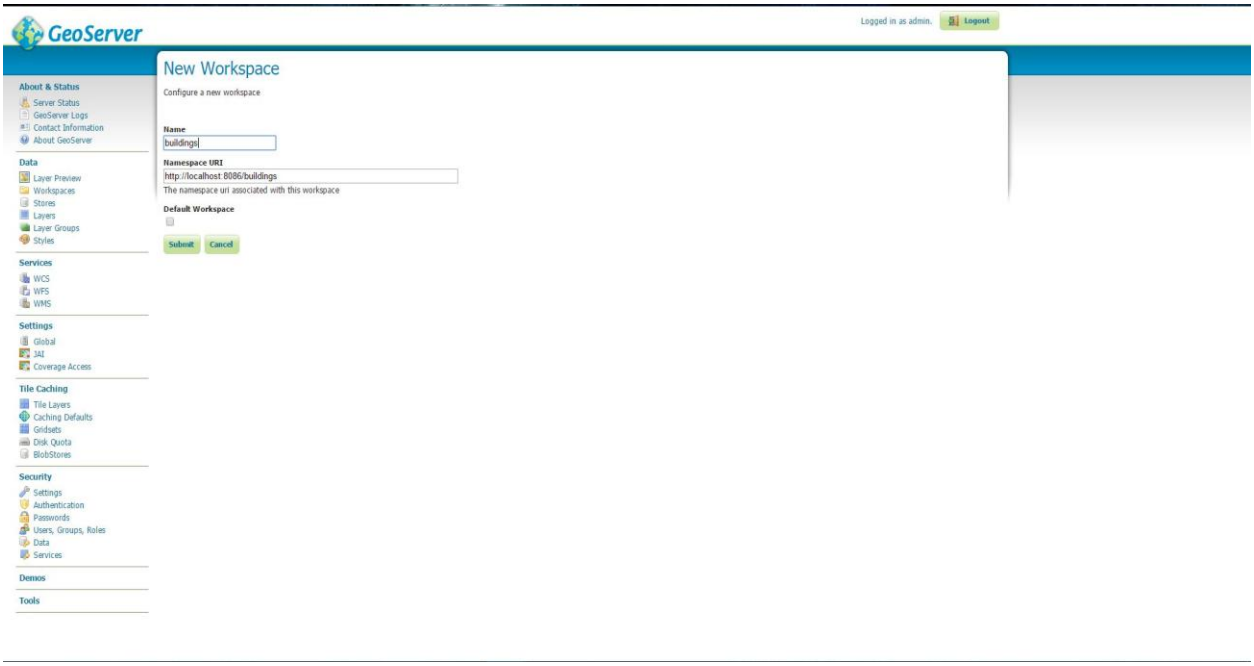
Αφού συνδεθούμε για να δημιουργήσουμε ένα νέο WMS (που θα εμφανίζει την εικόνα που επιλέξαμε πάνω στο χάρτη στη θέση που έχουμε επιλέξει), θα πρέπει να έχουμε δημιουργήσει το worldfile της εικόνας μας (όπως περιγράψαμε παραπάνω) και να ακολουθήσουμε τα παρακάτω βήματα:

1) Αρχικά θα δημιουργήσουμε ένα καινούργιο workspace. Ο ρόλος του οποίου είναι να ομαδοποιεί layers ίδιου τύπου. Το συναντάμε και όταν θέλουμε να αναφερθούμε σε κάποιο layer με την εξής δομή: όνομα workspace + άνω-κάτω τελεία + όνομα layer (π.χ. topp:states). Για να δημιουργήσουμε ένα νέο workspace, πάμε στο μενού που έχουμε στα αριστερά μας και πατάμε πάνω στο workspaces, και στη συνέχεια "add new workspace", (βλ. Εικόνα 7), πατώντας στην επιλογή που είναι στο πράσινο πλαίσιο βλέπουμε τα διαθέσιμα workspaces, καθώς και τις επιλογές για προσθήκη νέου (με το μπλε πλαίσιο) και διαγραφή επιλεγμένου.



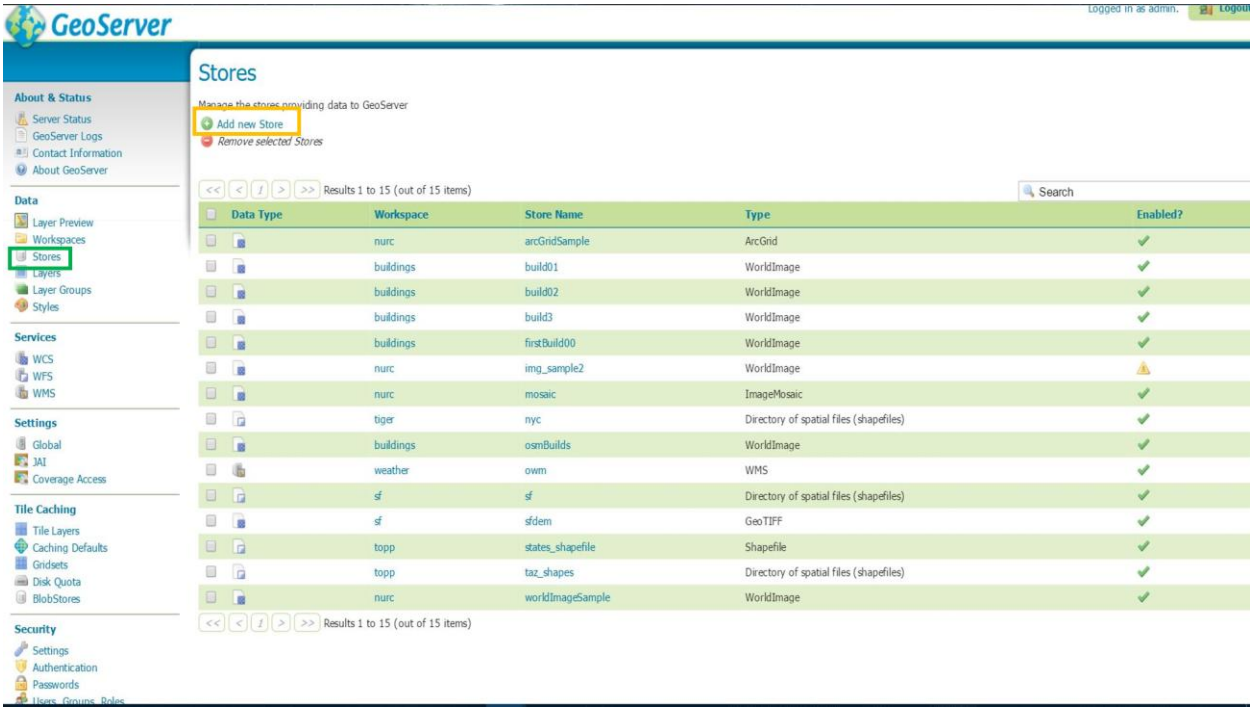
Εικόνα 7: add new workspace

Αφού επιλέξουμε το "add new workspace" θα εμφανιστεί μια φόρμα, στην οποία θα πρέπει να συμπληρώσουμε το όνομα του workspace μας και το URI που θα σχετίζετε με το συγκεκριμένο workspace. Το uri μπορεί να είναι κάποιος εξωτερικός σύνδεσμος προς το site από το οποίο παίρνουμε τα δεδομένα ή αν δεν υπάρχει κάποιο τότε μπορούμε να χρησιμοποιήσουμε το url του server μας + '/όνομα workspace' όπως μπορούμε να δούμε και στο παρακάτω screenshot (βλ. Εικόνα 8), στο οποίο δημιουργούμε ένα workspace με όνομα 'buildings' και με uri 'http://localhost:8086/buildings'.

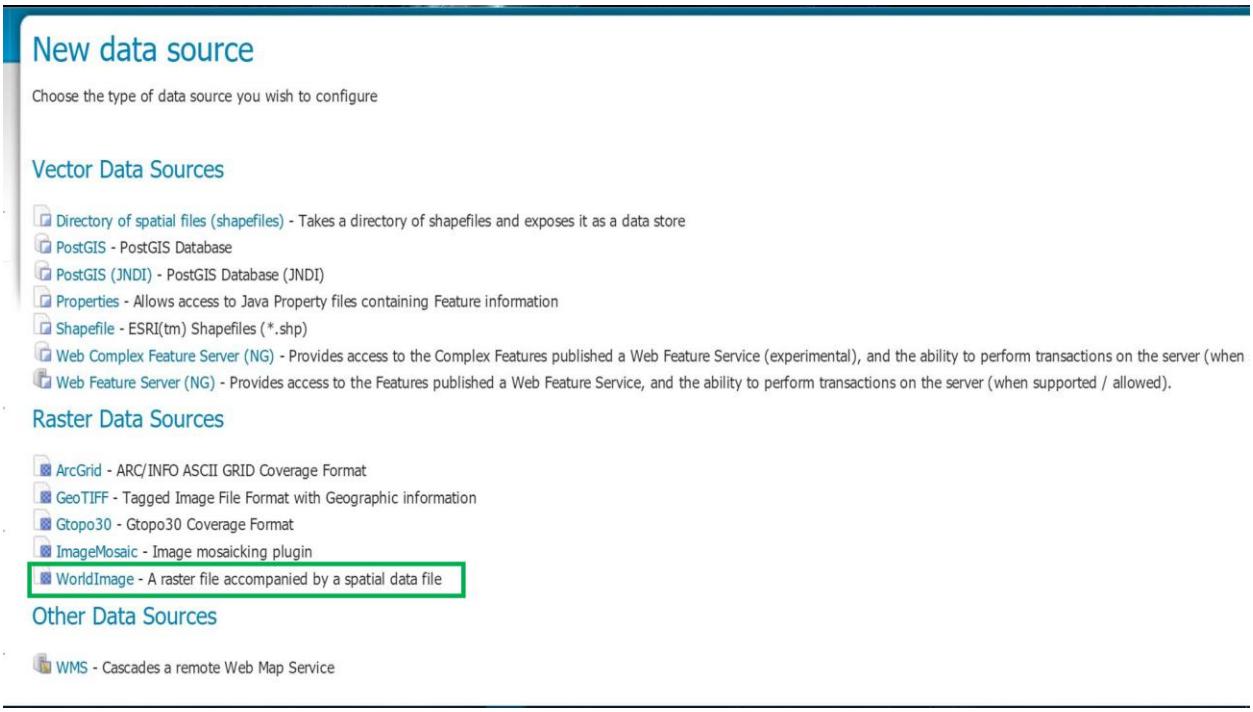


Εικόνα 8: new workspace

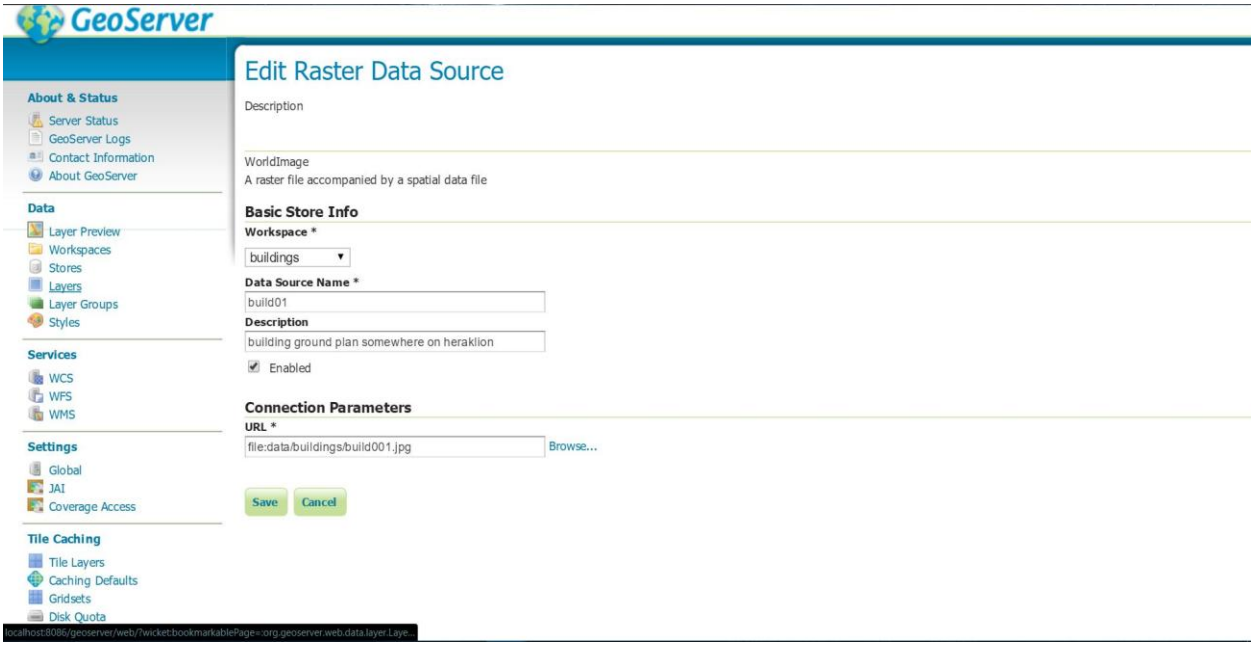
2) Στη συνέχεια θα πρέπει να φτιάξουμε ένα ‘store’, στο οποίο θα αποθηκευτούν όλες οι πληροφορίες που αφορούν το layer που πρόκειται να φτιάξουμε (όπως συντεταγμένες, τύπος, Συστήματα αναφοράς συντεταγμένων, κτλ.). Για να δημιουργήσουμε ένα νέο store θα πρέπει να πάμε από το μενού stores και στη συνέχεια Add new store (βλ. Εικόνα 9, stores πράσινο πλαίσιο, add new store πορτοκαλί πλαίσιο). Μόλις επιλέξουμε το add new store, θα μας ανοίξει μια λίστα με τους διαθέσιμους τύπους δεδομένων (βλ. Εικόνα 10). Όπως αναφέραμε και προηγουμένως ο τύπος της εικόνας που φτιάξαμε είναι worldImage, οπότε και εμείς θα επιλέξουμε τον τύπο worldImage (βλ. Εικόνα 10, πράσινο πλαίσιο), για την αποθήκευση των πληροφορίων του layer που πρόκειται να δημιουργήσουμε. Αφού επιλέξαμε τον τύπο του store, προχωράμε στη συμπλήρωση των πληροφοριών που αφορούν το συγκεκριμένο store. Οι πληροφορίες αυτές περιλαμβάνουν το workspace που θα χρησιμοποιήσουμε, το όνομα της πηγής μας, μια περιγραφή (προαιρετικό), αν θα είναι ενεργό και τέλος το μονοπάτι που έχουμε αποθηκεύσει την εικόνα και το worldfile, όπως μπορούμε να δούμε και στην Εικόνα 11.



Εικόνα 9: Add new store menu

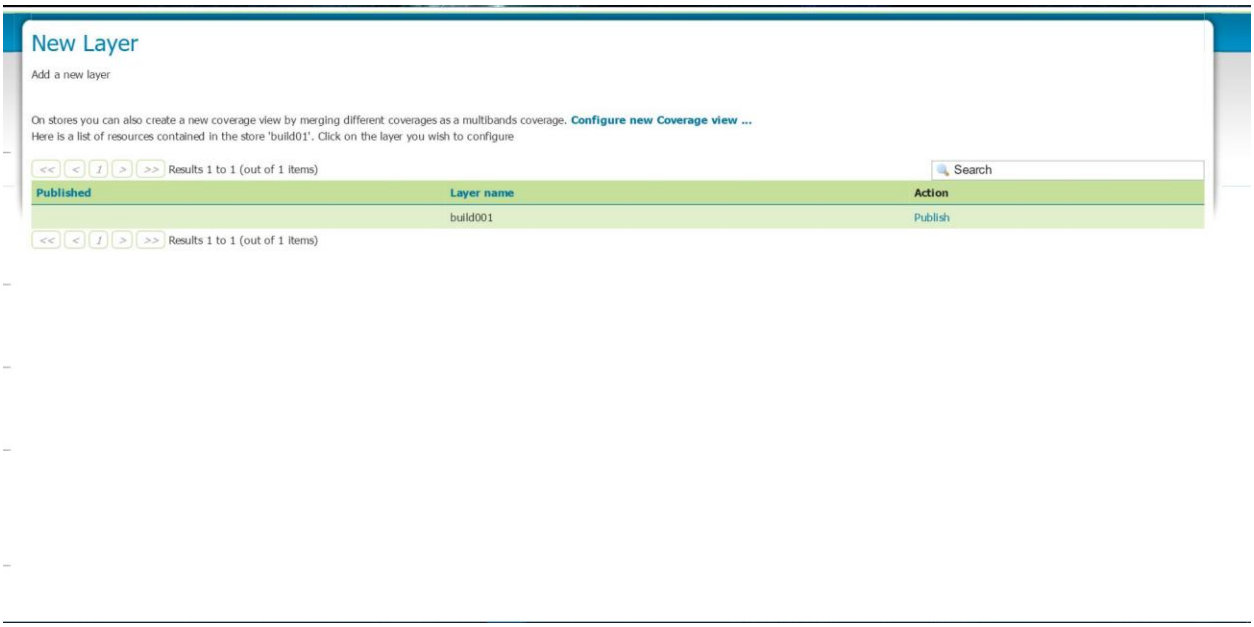


Εικόνα 10: data sources (add new store)

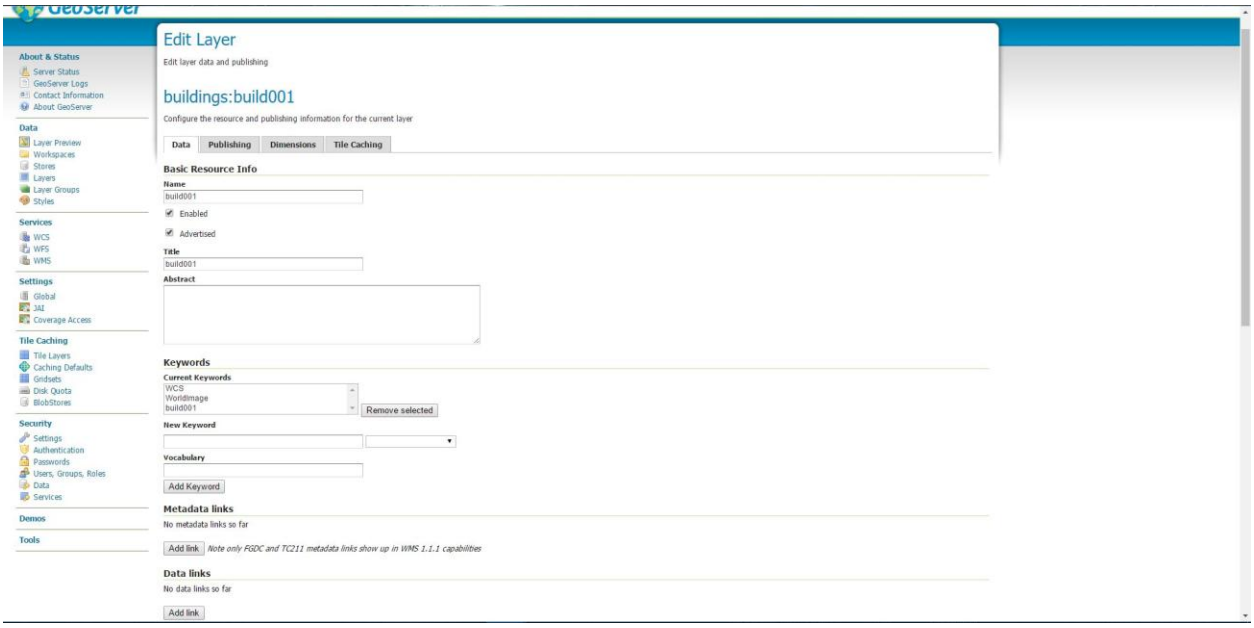


Εικόνα 11: data source info (new store)

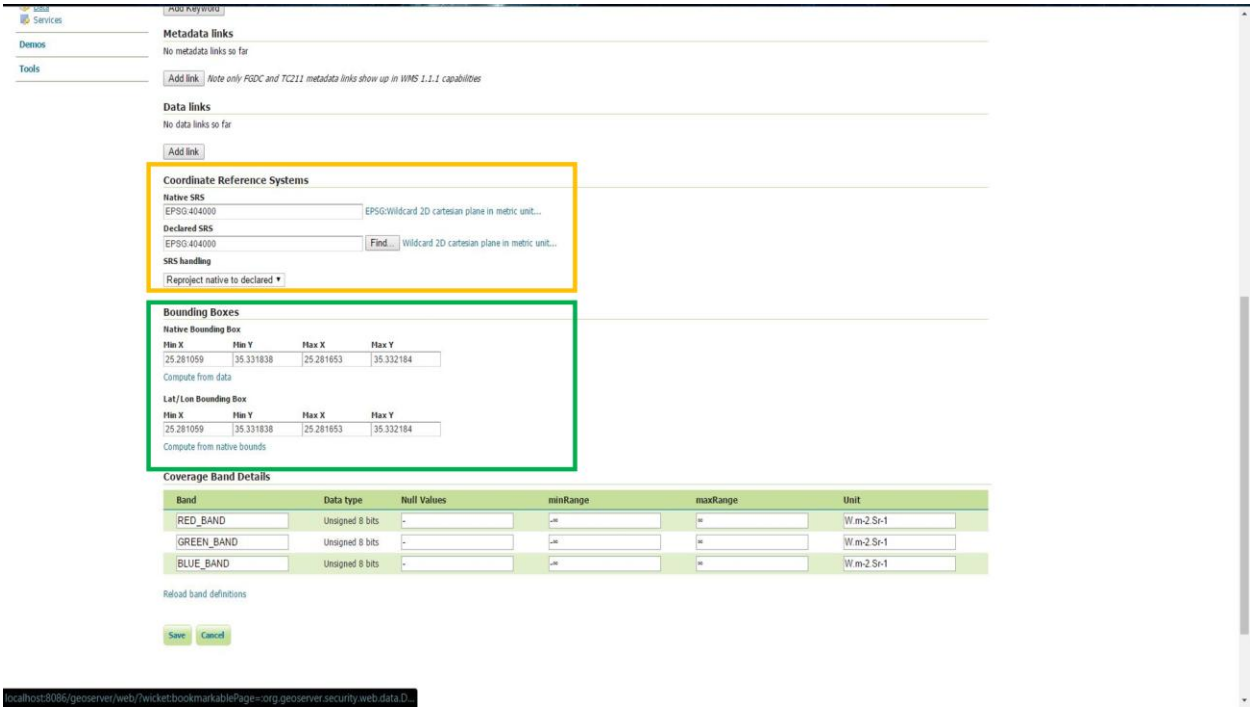
3) Μετά τη συμπλήρωση των πληροφοριών και την αποθήκευση, είμαστε έτοιμοι για τη δημοσίευση του layer που φτιάξαμε. Στην Εικόνα 12 βλέπουμε τα διαθέσιμα layers (που στη περίπτωση μας είναι ένα), για να προχωρήσουμε στη δημοσίευση του layer θα πρέπει να πατήσουμε το 'publish' που βρίσκετε κάτω από τη στήλη 'Action'. Πριν γίνει publish το layer που φτιάξαμε θα πρέπει πρώτα να συμπληρώσουμε κάποιες πληροφορίες στη σελίδα Edit layer (βλ. Εικόνα 13 - Εικόνα 14) που μας ανοίξει όταν πατήσουμε το publish. Από όλα τα πεδία που βλέπουμε στις εικόνες 13 και 14, αυτά που χρειάζεται να συμπληρώσουμε για να μπορέσουμε να δούμε το layer, είναι τα bounding boxes (βλ. Εικόνα 14 πράσινο πλαίσιο) τα οποία συμπληρώνονται αυτόματα πατώντας στις επιλογές 'compute from data' και ' compute from native bounds', τέλος θα πρέπει να επιλέξουμε το κατάλληλο σύστημα συντεταγμένων (βλ. Εικόνα 14 πορτοκαλί πλαίσιο) σύμφωνα με εκείνο που υποστηρίζει ο χάρτης που θα το προσθέσουμε πάνω.



Εικόνα 12: publish new layer

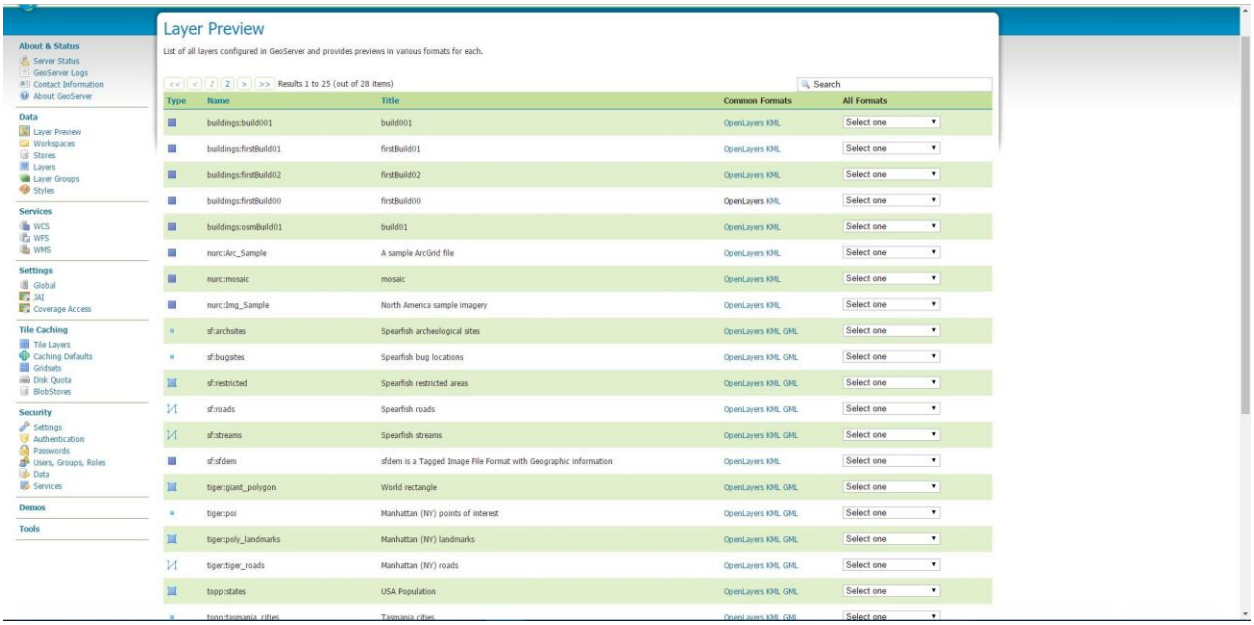


Εικόνα 13: edit Layer part 1

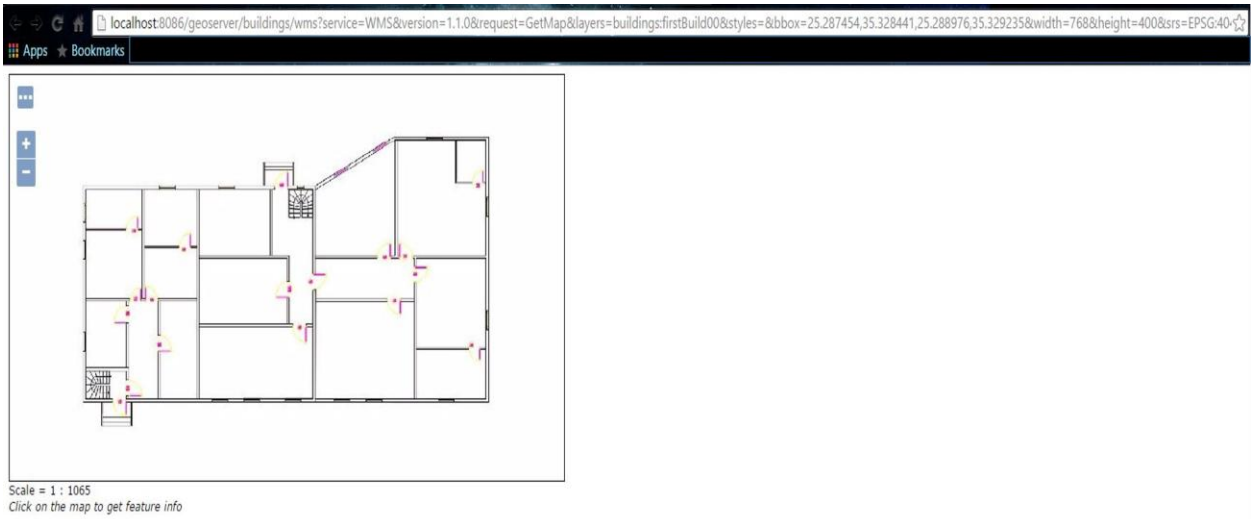


Εικόνα 14: edit Layer part 2

4) Τέλος μόλις πατήσουμε αποθήκευση θα πάμε στη σελίδα layer preview, όπου εκεί θα βρούμε το layer που φτιάξαμε καθώς και τα demo layers που έχει ο GeoServer από μόνος του (βλ. Εικόνα 15). Πηγαίνοντας τώρα στη γραμμή που είναι το layer που φτιάξαμε και στη στήλη common formats, και επιλέξουμε OpenLayers. Τέλος στην Εικόνα 16 βλέπουμε την εικόνα του layer που φτιάξαμε, στα screenshots του 5^{ου} κεφαλαίου φαίνεται το layer που φτιάξαμε πάνω στο χάρτη (και στην υλοποίηση στο web αλλά και στο android).



Εικόνα 15: layer preview page



Εικόνα 16: this is the preview of the our new layer

4.2 Android Client

Για την υλοποίηση της εφαρμογής στο android, αξιοποιήσαμε το API που παρέχει η βιβλιοθήκη HttpClient της Apache, για το συγχρονισμό και την ανταλλαγή μηνυμάτων. Από αρχιτεκτονική άποψη τώρα, για την επίτευξη της επικοινωνίας δημιουργήσαμε δύο κλάσεις τύπου interface (SyncDataListener και ChatListener) που χρησιμοποιούνται ως custom listeners, για τη καταγραφή και διαχείριση πληροφορίας που έχει να κάνει με τη τρέχουσα θέση του χρήστη, την αλλαγή κατάστασης κάποιου καθήκοντος, καθώς και για την αποστολή chat μηνυμάτων. Επίσης φτιάξαμε τη κλάση Communicator για την επικοινωνία με τον server, και τη κλάση AndroidMapPropagator ρόλος της οποίας είναι η λήψη των πληροφοριών που παρέχουν τα interfaces που αναφέραμε προηγουμένως, την αποστολή των πληροφοριών στον server και τη διαχείριση της πληροφορίας που στέλνει ο server.

Ως προς τη διεπαφή και τα χωρικά δεδομένα βασιστήκαμε στο API του Glob3Mobile SDK, μια open source βιβλιοθήκη με δυνατότητες, χρήσης αλλά και την αναπαράστασης/παρουσίασης (σε 2D ή και σε 3D) κάθε τύπου χωρικών δεδομένων. Την διεπαφή της εφαρμογής μας μπορούμε να πούμε ότι βασίζετε στις κλάσεις Map, Actor και ActorTask. Η κλάση Map είναι η βάση για την υλοποίηση της εφαρμογής, καθώς είναι υπεύθυνη για την γραφική απεικόνιση των χωρικών/γεωγραφικών πληροφοριών που χρειαζόμαστε, όπως τη τρέχουσα θέση των χρηστών ή και την θέση των καθηκόντων που έχουν αυτοί. Η κλάση Actor περιέχει όλες τις πληροφορίες που έχουν να κάνουν με τον χρήστη, όπως το όνομα, τη τρέχουσα θέση του, σε ποια ομάδα ανήκει κ.τ.λ. Τέλος η κλάση ActorTask περιέχει πληροφορίες που αφορούν ένα καθήκον που έχει ένας χρήστης, όπως περιγραφή, θέση, τρέχουσα κατάσταση και το όνομα της ομάδας που της έχει ανατεθεί το συγκεκριμένο καθήκον.

4.3 Web Client

Για την υλοποίηση της εφαρμογής στο Web αξιοποιήσαμε το API της βιβλιοθήκης OpenLayers, για την απεικόνιση των γεωγραφικών/χωρικών δεδομένων και τη vis για τη γραφική απεικόνιση των καθηκόντων, πάνω σε ένα χρονοδιάγραμμα (timeline). Για το συγχρονισμό έχουμε τις κλάσεις WebMapCommunicator και WebMapPropagator, οι οποίες έχουν την ίδια λειτουργικότητα με της Communicator και Propagator αντίστοιχα, με τη διαφορά ότι για την ανταλλαγή των μηνυμάτων γίνετε χρήση του API που παρέχει η AJAX διαμέσου της JQuery.

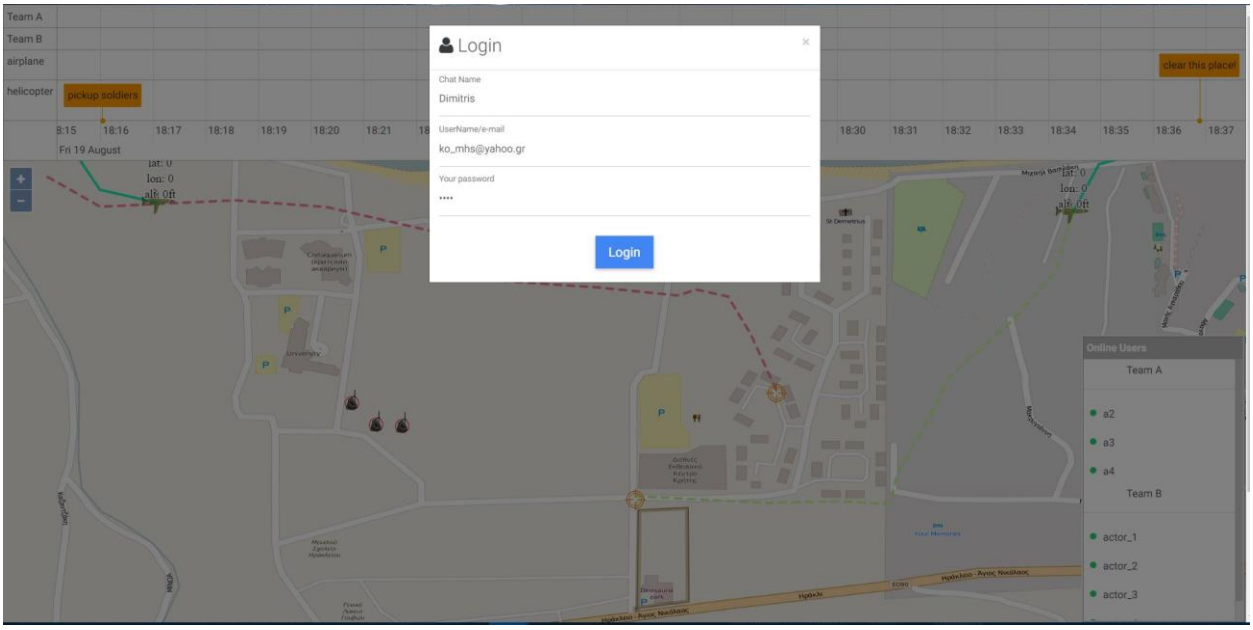
Ως προς τη διεπαφή παρατηρούμε, ότι οι κλάσεις (WebMapField, WebMapActor και Task) που έχουν να κάνουν με την γραφική απεικόνιση των χωρικών δεδομένων δεν διαφέρουν ως προς τη λειτουργικότητα, με τις αντίστοιχες του Android. Η μόνη διαφορά είναι ότι εδώ έχουμε 2D αναπαράσταση των χωρικών δεδομένων, ενώ στο android έχουμε 3D. Στην υλοποίηση μας στο web έχουμε προσθέσει και ένα χρονοδιάγραμμα, στο οποίο μπορούμε να δούμε τη σειρά με την οποία θα πρέπει να εκτελεστούν τα καθήκοντα, που έχουν ανατεθεί σε κάθε χρήση/ομάδα, καθώς και να ενημερώσουμε την τρέχουσα κατάσταση σε κάποιο task. Η κλάση που είναι υπεύθυνη για τη διαχείριση του timeline είναι η TaskManager.

5 Σενάριο επίδειξης

Το σενάριο επίδειξης που επιλέξαμε αφορά, την εκτέλεση/προσομοίωση μιας στρατιωτικής άσκησης. Πιο συγκεκριμένα θα δούμε πως το σύστημα που υλοποιήσαμε θα μπορούσε να διευκολύνει το συντονισμό, αλλά και την επιτήρηση οποιασδήποτε άσκησης. Αρχικά θα δούμε της δυνατότητες που έχει ο συντονιστής (web εφαρμογή) και στη συνέχεια τις δυνατότητες ενός κινητού χρήστη (Android εφαρμογή) ανάλογα με το ρόλο του (αρχηγός ομάδας ή απλός στρατιώτης στη περίπτωση μας), και τέλος θα δείξουμε τις εφαρμογές συγχρονισμένες.

5.1 Υλοποίηση στο Web

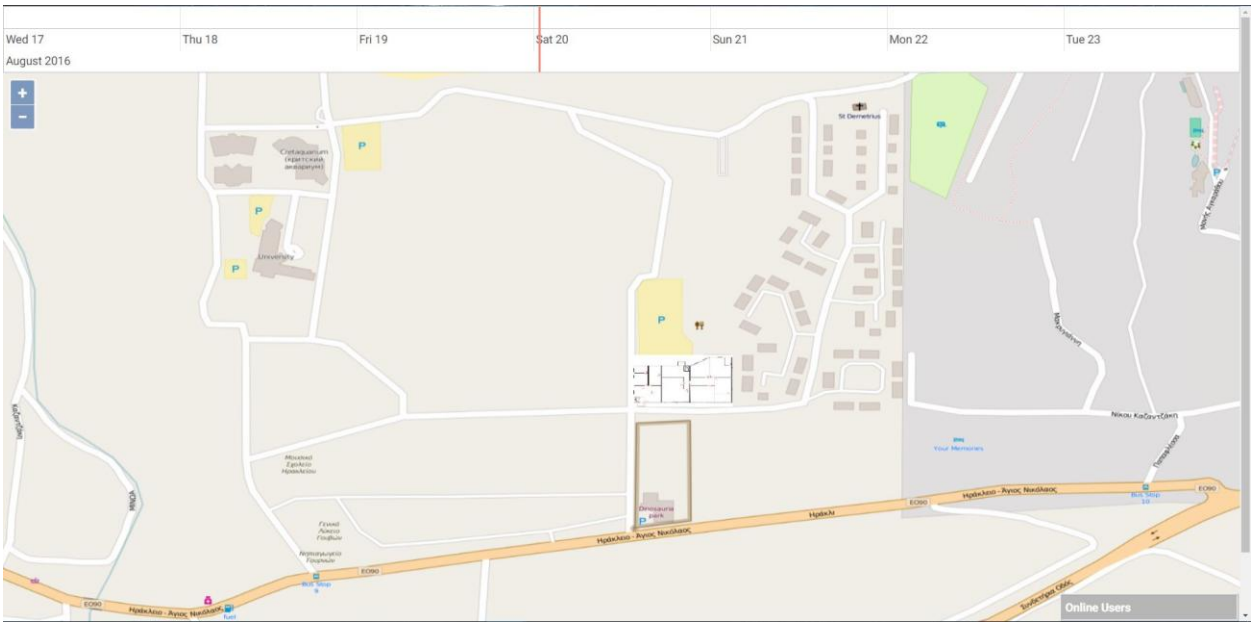
Ξεκινώντας την εφαρμογή στο web, το πρώτο που θα δούμε είναι ο διάλογος για την σύνδεση με το server και τον προσδιορισμό του ονόματος που θα βλέπουν οι χρήστες του android, όπως φαίνεται παρακάτω στην Εικόνα 17.



Εικόνα 17: login dialog

Μόλις γίνει η σύνδεση προχωράμε στη κεντρική οθόνη της εφαρμογής, που μπορούμε να τη χωρίσουμε στα εξής 3 μέρη: 1) το χρονοδιάγραμμα (timeline) για την προβολή των καθηκόντων που έχει η κάθε ομάδα, 2) ο χάρτης που είναι το κεντρικό κομμάτι της εφαρμογής μας, καθώς

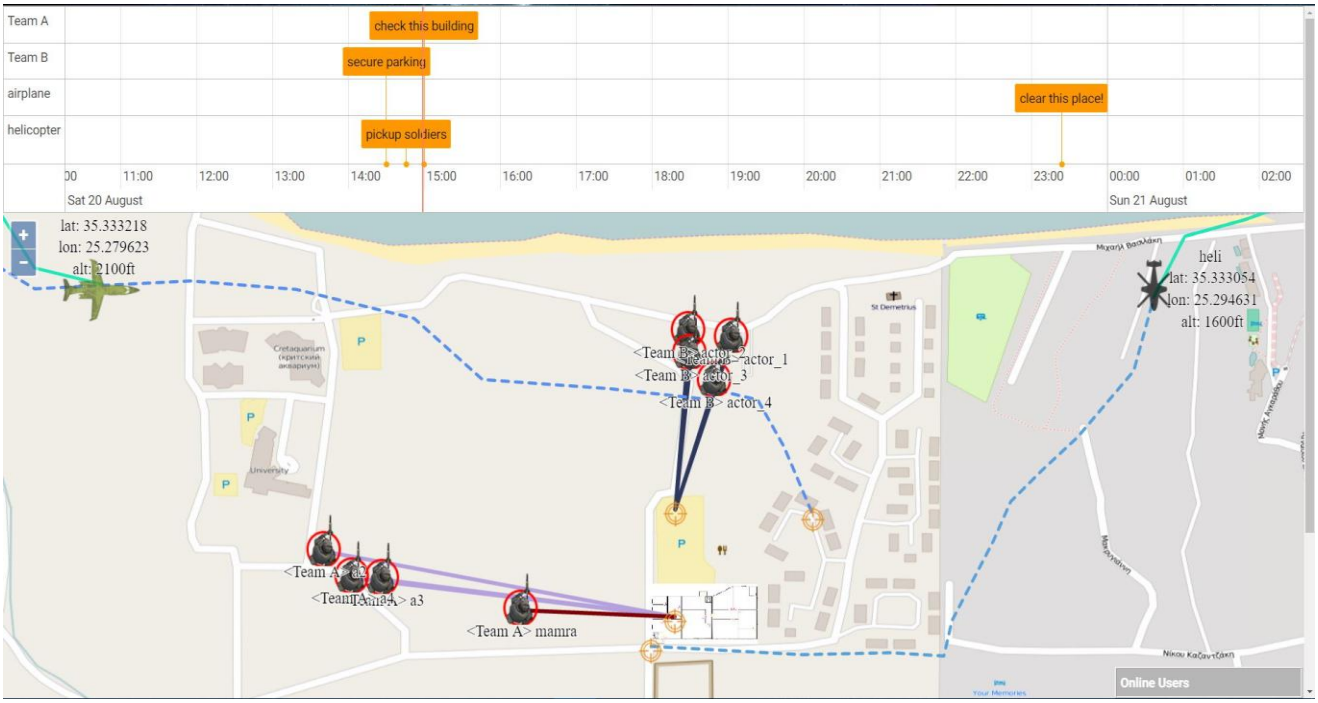
πάνω σε αυτόν γίνετε η προβολή της θέσης κάθε χρήστη και η προσθήκη/προβολή καθηκόντων και 3) η διεπαφή του chat. Βλ. Εικόνα 18 και Εικόνα 19.



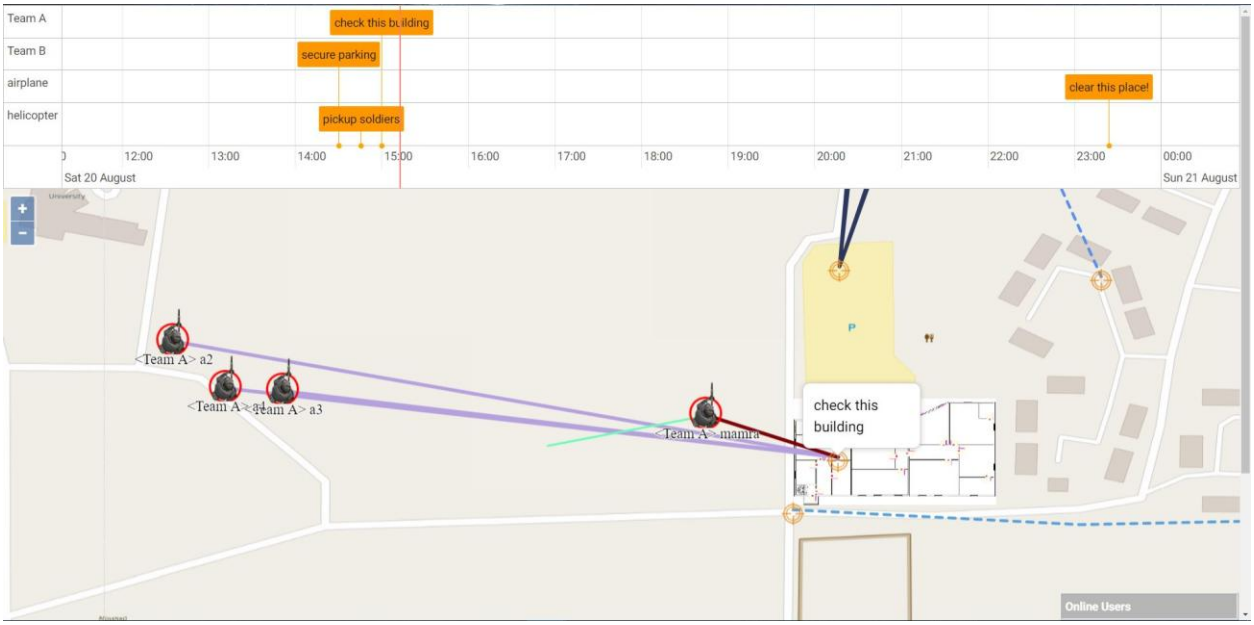
Εικόνα 18: main screen

Εικόνα 19: demo scenario preview

Στην Εικόνα 19 βλέπουμε ενδεικτικά κάποιους χρήστες χωρισμένους σε δύο ομάδες, ένα καθήκον (task) για κάθε ομάδα. Έχουμε προσθέσει ακόμα δύο χρήστες τύπου οχήματος με ένα task στο καθένα, που στόχο έχει την ανάδειξη της επεκτασιμότητας και μορφής που μπορεί να υποστηρίξει η εφαρμογή ως προς τον τύπο του χρήστη. Τα καθήκοντα που έχουμε τοποθετήσει πάνω στο χάρτη έχουν ως εικονίδιο ένα στόχο, ενώ το χρώμα του προσδιορίζει την τρέχουσα κατάσταση του καθήκοντος (πορτοκαλί εκκρεμεί, πράσινο επιτυχές, κόκκινο απέτυχε και μπλε επιλεγμένο, θα τα δούμε πιο αναλυτικά παρακάτω). Οι συνεχόμενες γραμμές καθώς και οι διακεκομμένες(από τον χρήστη προς το καθήκον) δείχνουν που είναι το (τρέχον) καθήκον που τους έχει ανατεθεί.



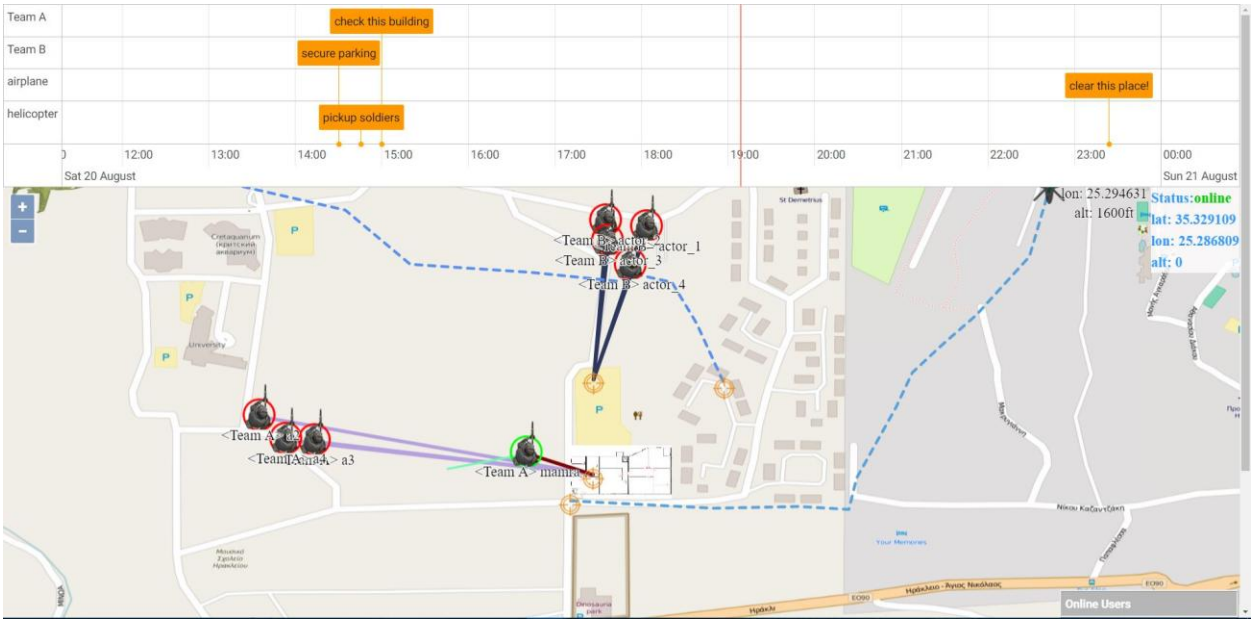
Εικόνα 20: new android actor logged in



Εικόνα 21: android actor moved

Στην Εικόνα 20 βλέπουμε τη σύνδεση ενός android χρήστη (με το όνομα mamra) στο server μας και την άμεση προβολή της θέσης του πάνω στον χάρτη της εφαρμογής μας. Στην Εικόνα 21 βλέπουμε ότι ο χρήστης mamra έχει κινηθεί προς το καθήκον που του έχει ανατεθεί, με τίτλο/περιγραφή "check this building" όπως φαίνεται και στο popup (το popup με τη περιγραφή ενός task εμφανίζεται όταν περνάμε το ποντίκι πάνω από το εικονίδιο του συγκεκριμένου task.) που βρίσκεται πάνω από το συγκεκριμένο task. Ακόμα βλέπουμε μια ευθεία γραμμή (που ακολουθεί τον χρήστη), που σκοπό έχει τη καταγραφή της πορείας που ακολουθεί ο χρήστης.

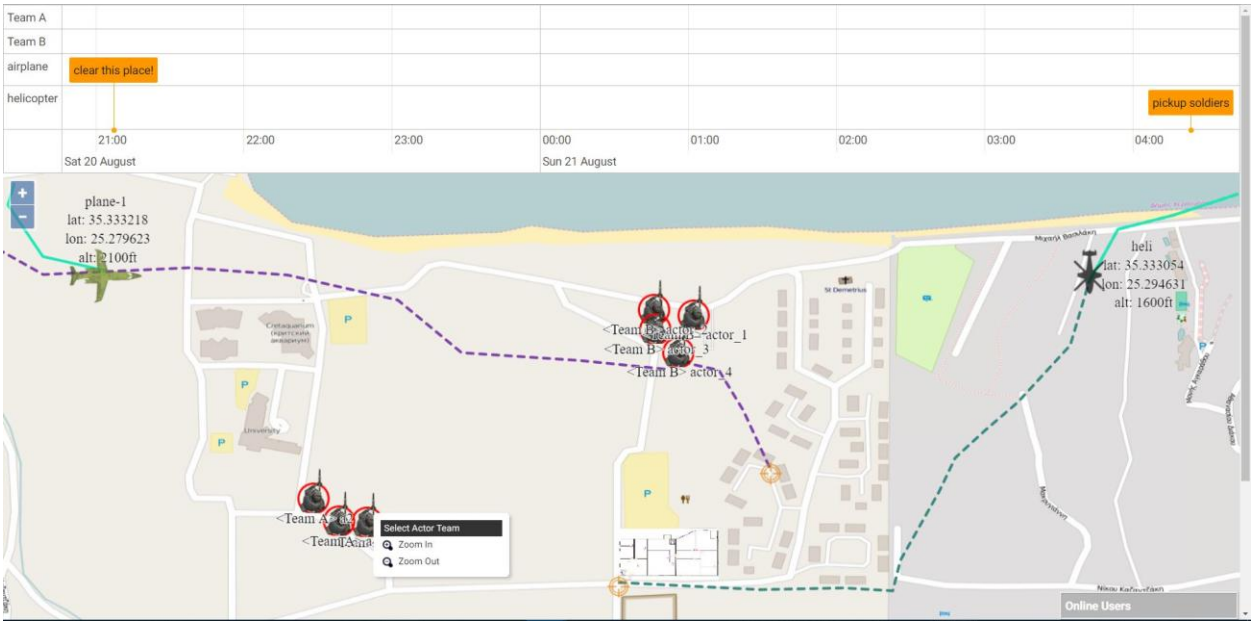
Κάνοντας κλικ πάνω σε έναν χρήστη, ο χρήστης αυτός επιλέγετε και ταυτόχρονα εμφανίζετε, στη πάνω δεξιά γωνία ένα πλαίσιο με πληροφορίες που αφορούν, τη τρέχουσα θέση του επιλεγμένου χρήστη (σε συντεταγμένες της μορφής γεωγραφικού μήκους και πλάτους), καθώς και τη κατάσταση της σύνδεσης του (ενεργός, ανενεργός, κτλ.), όπως φαίνεται και στη παρακάτω εικόνα (βλ. Εικόνα 22).



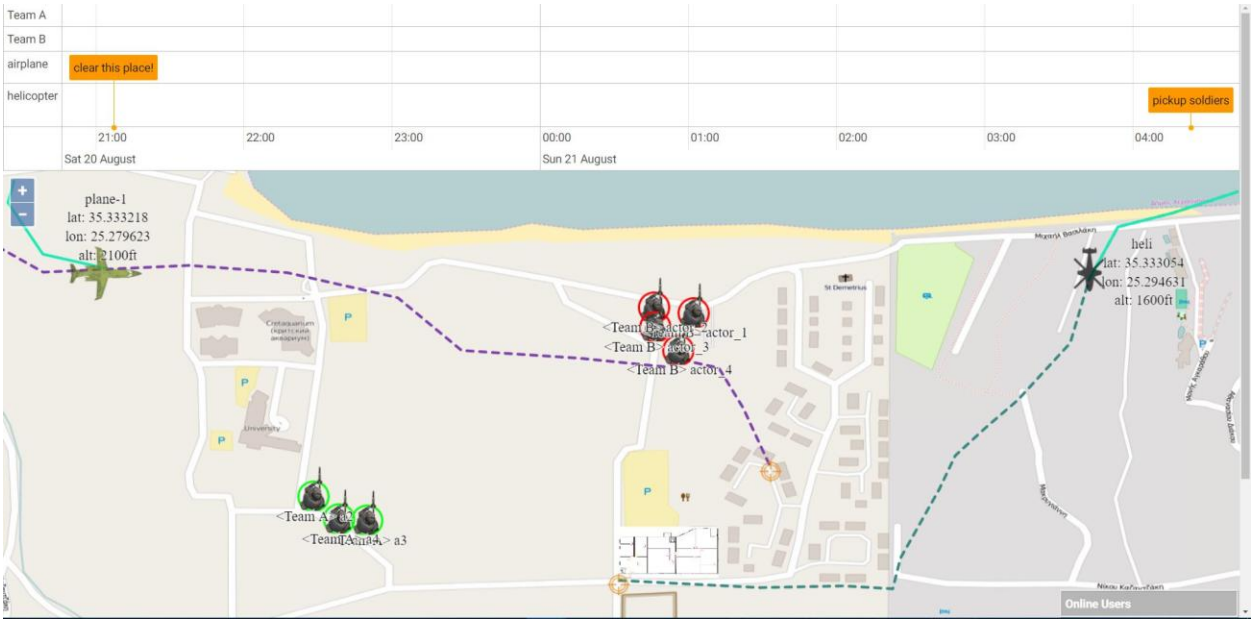
Εικόνα 22: selected actor

Ως προς τα καθήκοντα τώρα, από την πλευρά του web οι δυνατότητες που έχουμε είναι οι εξής: α) ανάθεση καθήκοντος σε κάποια ομάδα και β) ενημέρωση κατάστασης επιλεγμένου καθήκοντος. Στις εικόνες/screenshots που θα ακολουθήσουν θα δούμε από την αρχή τη διαδικασία που ακολουθείτε για τη δημιουργία και ανάθεση ενός καθήκοντος σε μια ομάδα, καθώς και πως γίνετε η ενημέρωση της κατάστασης του.

Για να δημιουργήσουμε και να αναθέσουμε ένα task σε μια ομάδα, θα πρέπει πρώτα να την επιλέξουμε. Για να γίνει η επιλογή θα πρέπει να τοποθετήσουμε τον δείκτη πάνω από ένα χρήστη που ανήκει στην ομάδα που επιθυμούμε, και στη συνέχεια να κάνουμε δεξί κλικ, στο μενού που θα εμφανιστεί (βλ. Εικόνα 23) επιλέγουμε το “Select Actor Team”. Μόλις γίνει αυτό ο κύκλος που περιέχει το εικονίδιο του στρατιώτη θα γίνει πράσινο (βλ. Εικόνα 24).



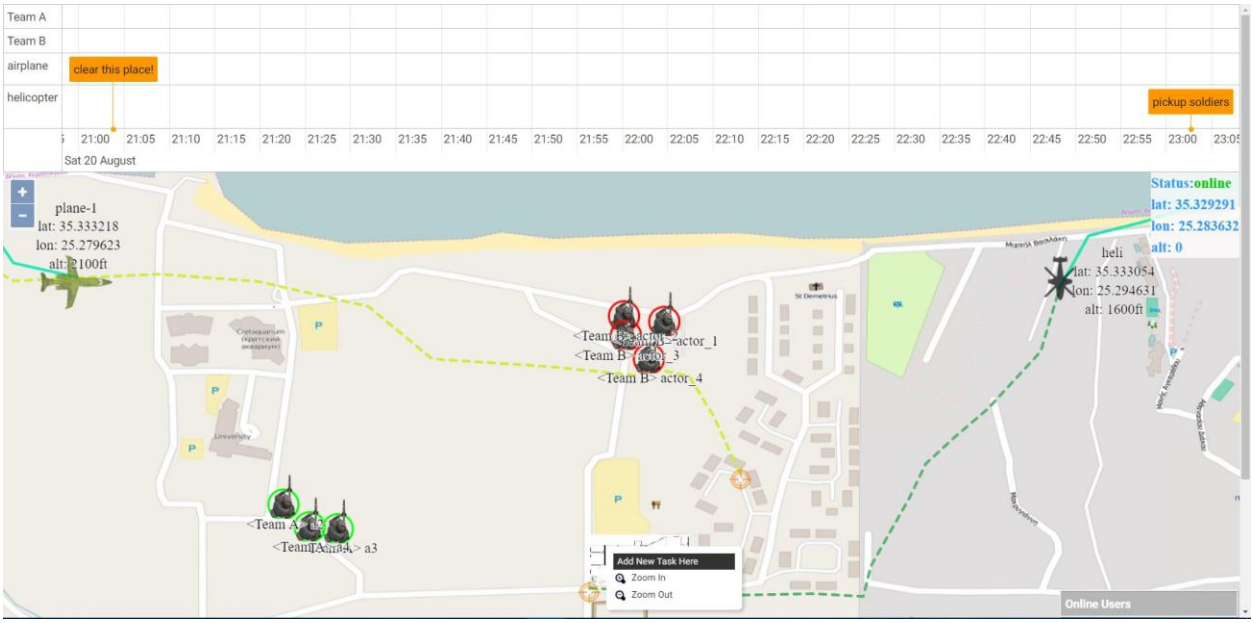
Εικόνα 23: select team menu



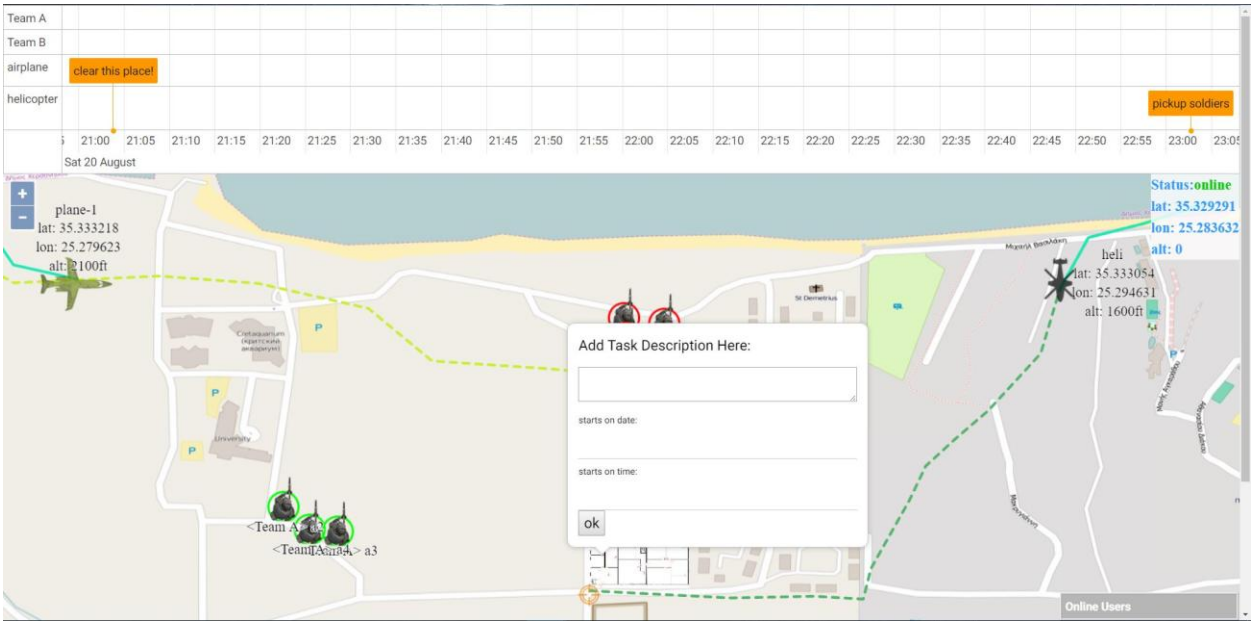
Εικόνα 24: team selected

Στη συνέχει τοποθετούμε τον δείκτη στη θέση που επιθυμούμε να δημιουργήσουμε ένα task και πατάμε δεξί κλικ, στο μενού που εμφανίζετε (βλ. Εικόνα 25) επιλέγουμε το "Add New Task Here", μόλις κάνουμε κλικ θα εμφανιστεί ένα popup (βλ. Εικόνα 26), στο οποίο θα πρέπει να συμπληρώσουμε την περιγραφή/τίτλο του task (βλ. Εικόνα 27), καθώς και να επιλέξουμε ημέρα (βλ. Εικόνα 28) και ώρα (βλ. Εικόνα 29). Η συμπλήρωση της φόρμας θα πρέπει να μοιάζει με

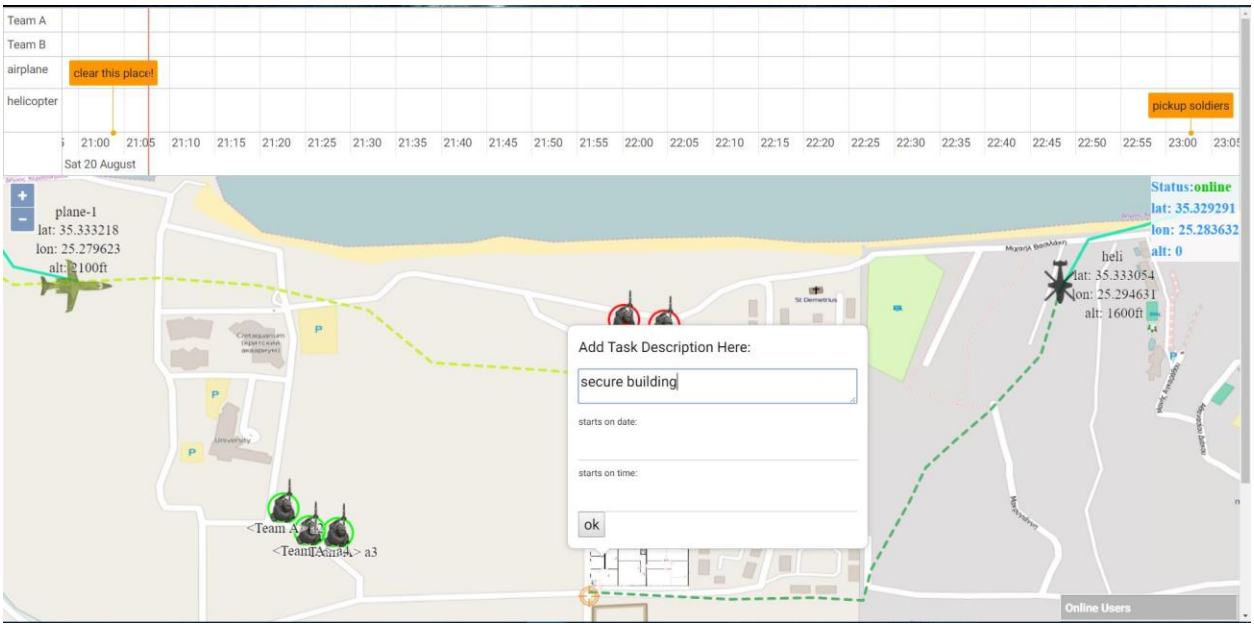
εκείνη της Εικόνα 30 και για να ολοκληρωθεί η διαδικασία πατάμε το κουμπί “ok”. Στην Εικόνα 31 βλέπουμε το task που δημιουργήσαμε πάνω στο χάρτη αλλά και στο timeline.



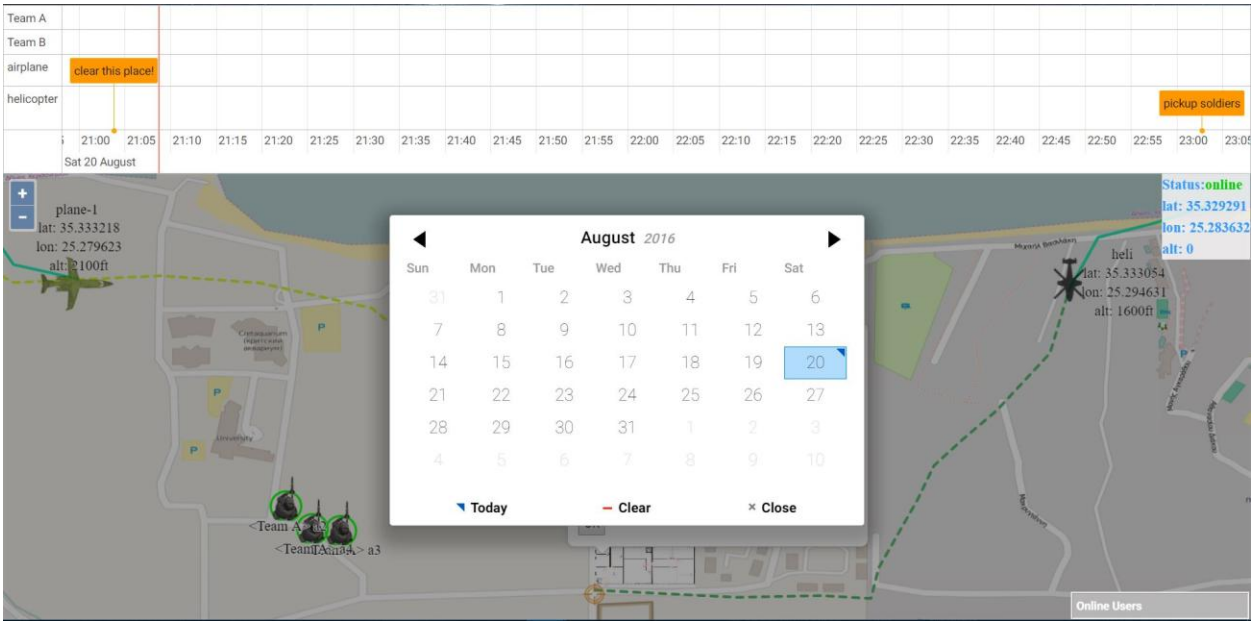
Εικόνα 25: add new task



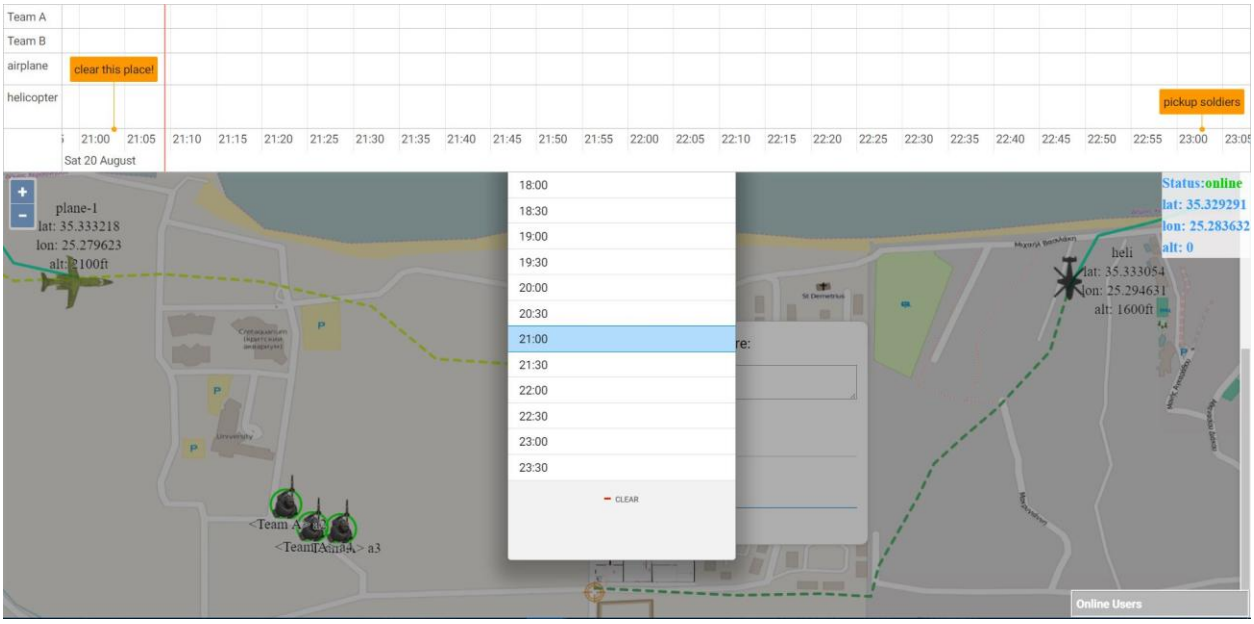
Εικόνα 26: new task form



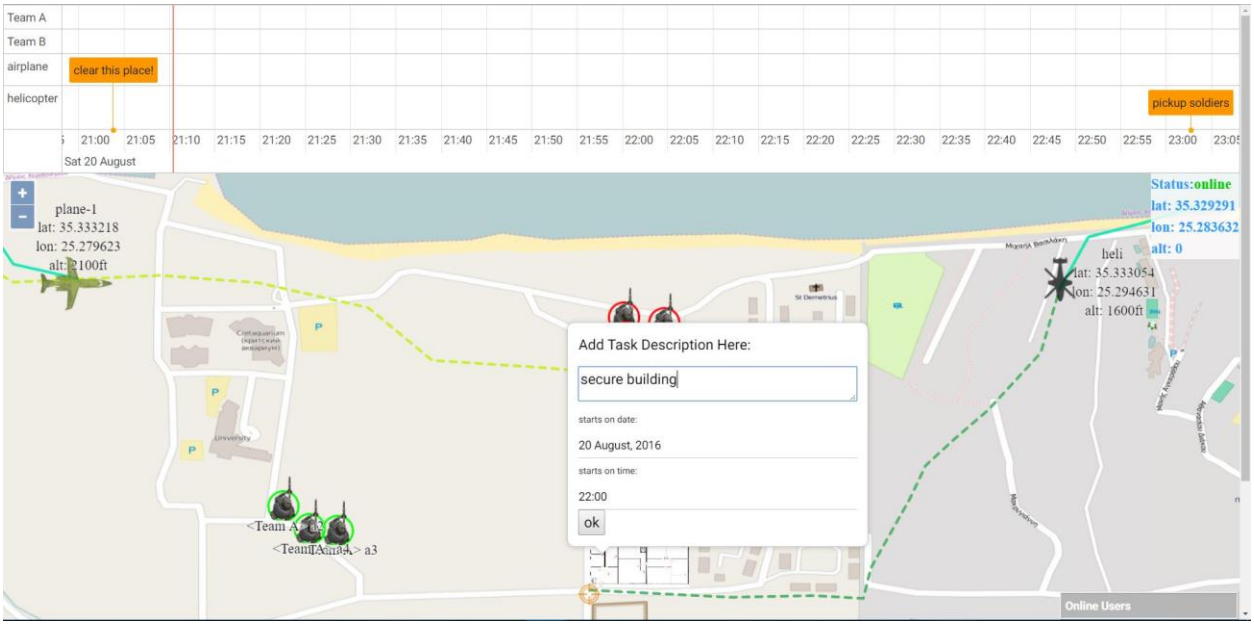
Εικόνα 27: new task (add description)



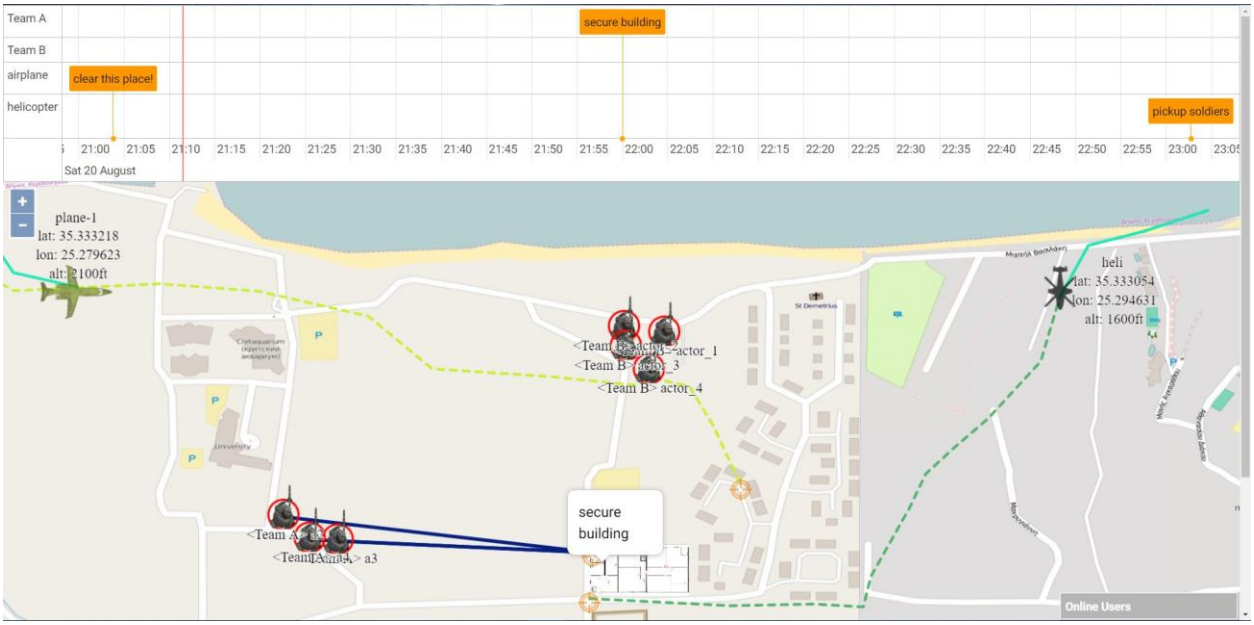
Εικόνα 28: new task (select date)



Εικόνα 29: new task (select time)

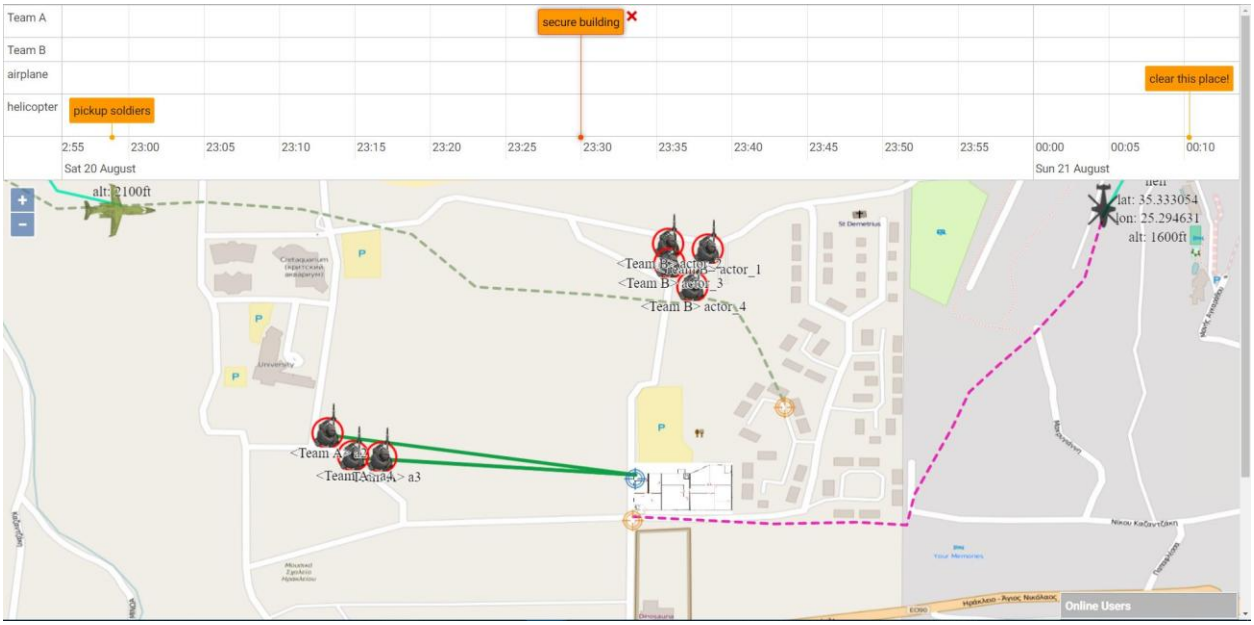


Εικόνα 30: new task form complete



Εικόνα 31: new task preview

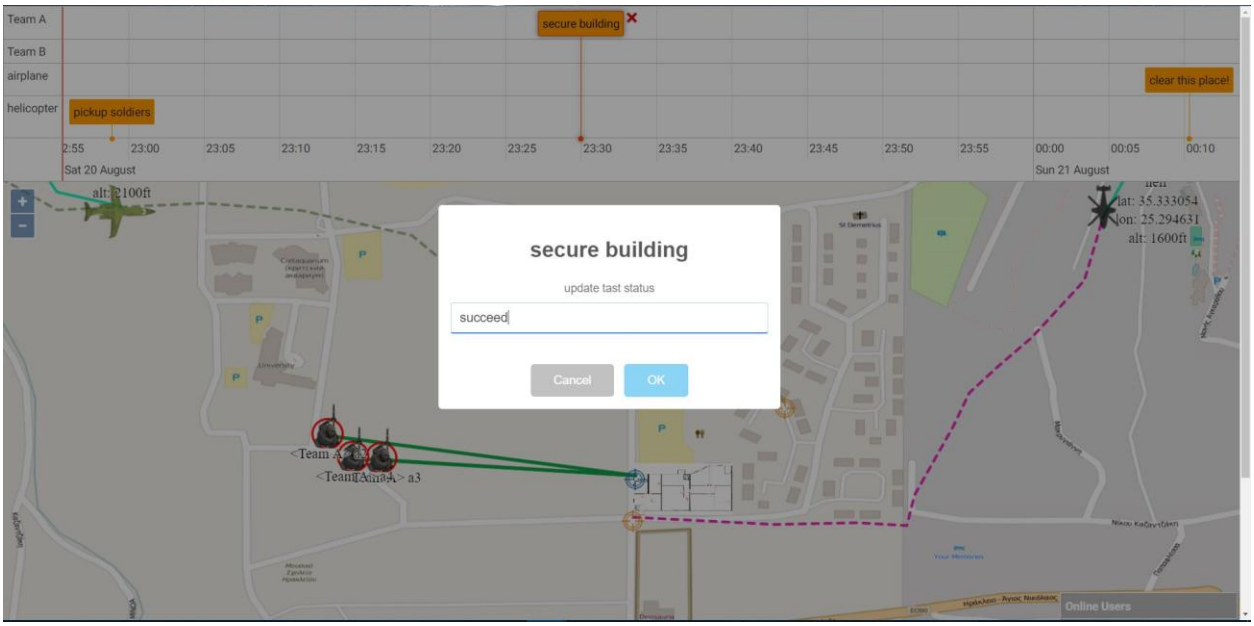
Για να επιλέξουμε ένα task, το μόνο που έχουμε να κάνουμε είναι να κλικ στο task που επιθυμούμε πάνω στο timeline (βλ. Εικόνα 32). Όταν επιλέγουμε ένα task το εικονίδιο που του αντιστοιχεί πάνω στον χάρτη γίνετε ανοιχτό μπλε, ενώ στο timeline σκουραίνει λίγο το χρώμα του και εμφανίζεται το εικονίδιο του κλεισίματος.



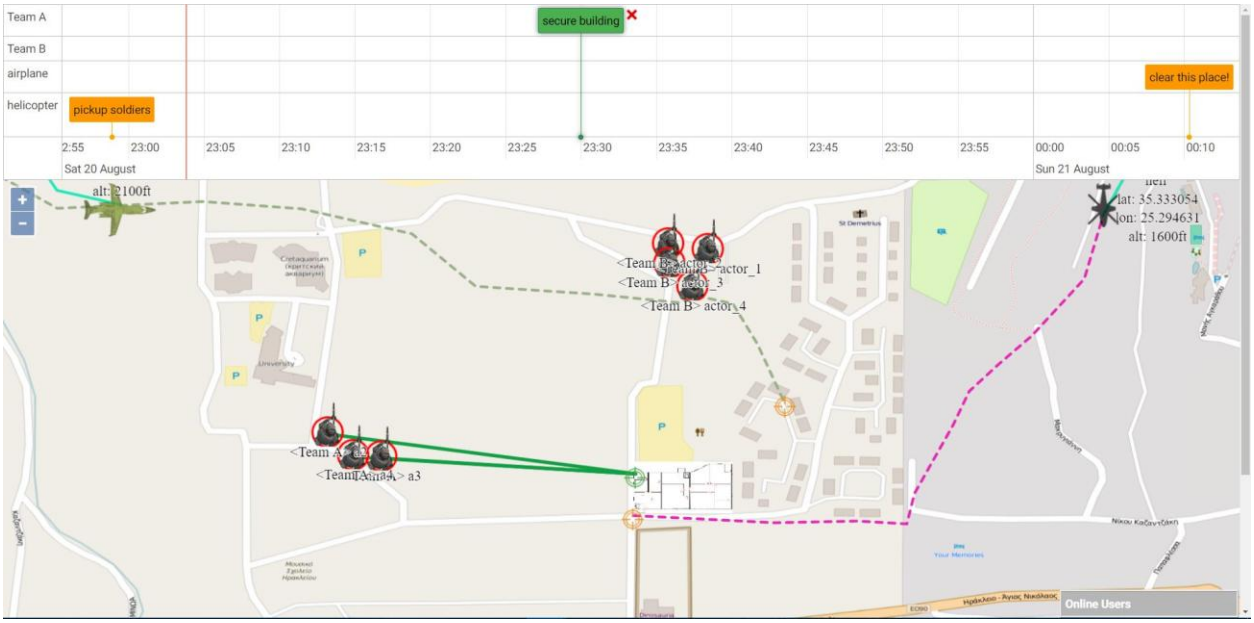
Εικόνα 32: select task

Για να ενημερώσουμε την κατάσταση ενός task, θα πρέπει να κάνουμε διπλό κλικ στο task που θέλουμε ή που έχουμε επιλέξει, για να μας εμφανιστή ο διάλογος που φαίνεται στην Εικόνα 33.

Στο πεδίο κειμένου του διαλόγου γράφουμε τη νέα κατάσταση του task (οι καταστάσεις είναι: pending, succeed και failed. Με αρχική κατάσταση την pending). Μετά την ενημέρωση της κατάστασης το χρώμα του καθήκοντος θα αλλάξει ανάλογα με την κατάσταση που έχουμε επιλέξει (βλ. Εικόνα 34).



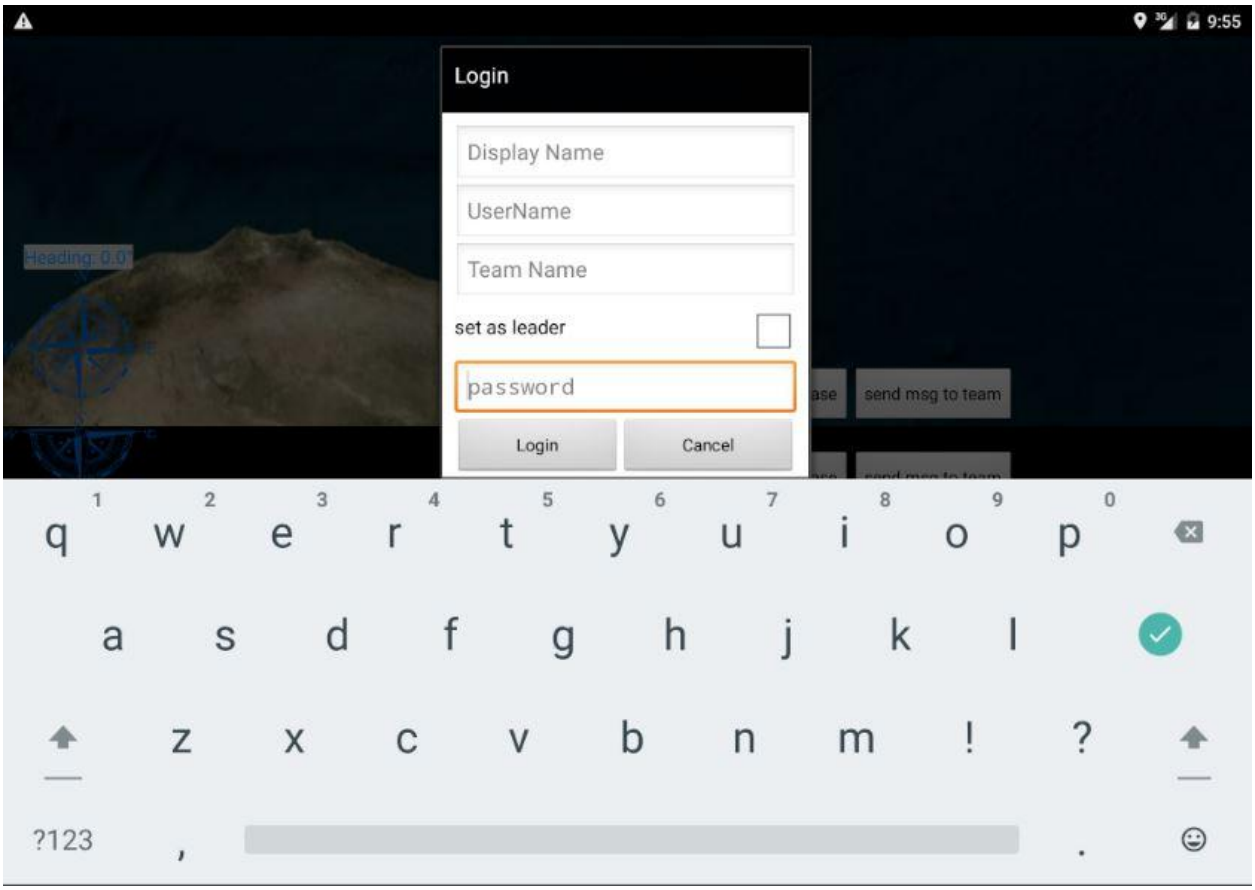
Εικόνα 33: update task status dialog



Εικόνα 34: task status updated

5.2 Υλοποίηση στο Android

Όπως στην υλοποίηση στο web έτσι και στο android, με το ξεκίνημα της εφαρμογής εμφανίζετε ένας διάλογος, στον οποίο θα πρέπει να συμπληρώσουμε τα στοιχειά που χρειάζονται για να συνδεθούμε στο server, καθώς και να προσδιορίσουμε το όνομα που θα έχουμε στην εφαρμογή, το όνομα της ομάδας μας και αν είμαστε αρχηγός ομάδας, όπως βλέπουμε στον παρακάτω screenshot (βλ. Εικόνα 35).



Εικόνα 35: android login dialog

Μετά τη συμπλήρωση του διαλόγου αν δεν επιλέξουμε να είμαστε "αρχηγός ομάδας", θα πάμε στην οθόνη που φαίνεται στο παρακάτω screenshot (βλ. Εικόνα 36). Αν επιλέξουμε την αρχηγία τότε θα πάμε στην οθόνη που φαίνεται στην Εικόνα 37.

Εικόνα 36: android simple actor GUI

Εικόνα 37: android team-leader actor

Η βασική διαφορά που εντοπίζουμε έχει να κάνει, με της δυνατότητες που έχει ο κάθε χρήστης σύμφωνα με τον ρόλο του. Πιο συγκεκριμένα η διεπαφή του αρχηγού ομάδας μας δίνει τη δυνατότητα να επιλέξουμε κάποιο task (καις στη συνέχεια να ενημερώσουμε την τρέχουσα κατάσταση του, όπως θα δούμε παρακάτω). Στα κοινά σημεία τώρα βλέπουμε την ύπαρξη μιας λίστας με τα καθήκοντα που έχουμε (θα την γεμίσουμε παρακάτω), μια μπάρα με πληροφορίες σχετικά με την κατάσταση της σύνδεση μας και τη θέση μας, μια πυξίδα, μπάρα με την τρέχουσα θερμοκρασία (αν η συσκευή έχει τον κατάλληλο αισθητήρα), μπάρα για τα μηνύματα από το chat, μια μπάρα που ακολουθεί τον χρήστη (εμάς) και μας παρέχει πληροφορίες σχετικά με το πολυμεσικό υλικό (εικόνες και βίντεο που έχουμε τραβήξει με τη χρήση της κάμερα της συσκευής μας) που έχουμε μαζέψει. Επίσης τα κουμπιά που βλέπουμε έχουν να κάνουμε με την αποστολή μηνύματος στα μέλη της ομάδας (send msg to team) ή στη βάση (send msg to base), τον εντοπισμό της θέσης μιας ομάδας (locate team), τέλος στο μενού έχουμε τις επιλογές για να τραβήξουμε φωτογραφία ή βίντεο.

Ως προς τα καθήκοντα τώρα, κάθε φόρα ανατίθεται κάποιο task στην ομάδα που ανήκουμε από τον συντονιστή (web app), θα εμφανίζετε η περιγραφή του στη λίστα των καθηκόντων μας, καθώς και η θέση του πάνω στο χάρτη με τη μορφή ενός στόχο (όπως και στην υλοποίηση στο web) (βλ. Εικόνα 38). Επειδή έχουμε συνδεθεί ως ‘αρχηγός ομάδας’ έχουμε τη δυνατότητα να ενημερώσουμε τη κατάσταση του task που μας ανατέθηκε. Για να γίνει αυτό θα πρέπει πρώτα να επιλέξουμε το συγκεκριμένο καθήκον (βλ. Εικόνα 39-Εικόνα 40), και στη συνέχεια να επιλέξουμε μια από τις διαθέσιμες καταστάσεις (pending, succeed, failed), που εμφανίζονται όταν πατάμε στο “Update Task Status" , στην Εικόνα 41 βλέπουμε ότι η κατάσταση του καθήκοντος άλλαξε από “pending” (στόχος με πορτοκαλί χρώμα) σε “failed” (στόχος με κόκκινο χρώμα).

Εικόνα 38: new task added

Εικόνα 39: select task

Εικόνα 40: task selected

Εικόνα 41: task failed

Όπως στην υλοποίηση στο web έτσι κα εδώ ο χρήστης έχει τη δυνατότητα, να παρακολουθεί πως κινούνται τα υπόλοιπα μέλη της ομάδας του, αλλά και των υπόλοιπων ομάδων (στα πλαίσια του σεναρίου μας). Όπως βλέπουμε και στο παρακάτω screenshot (βλ. Εικόνα 42) υπάρχει μια γραμμή για κάθε χρήστη που ξεκινά από τη θέση που έκανε ο χρήστης login στο σύστημα και τον ακολουθεί όσο αυτός κινείται.

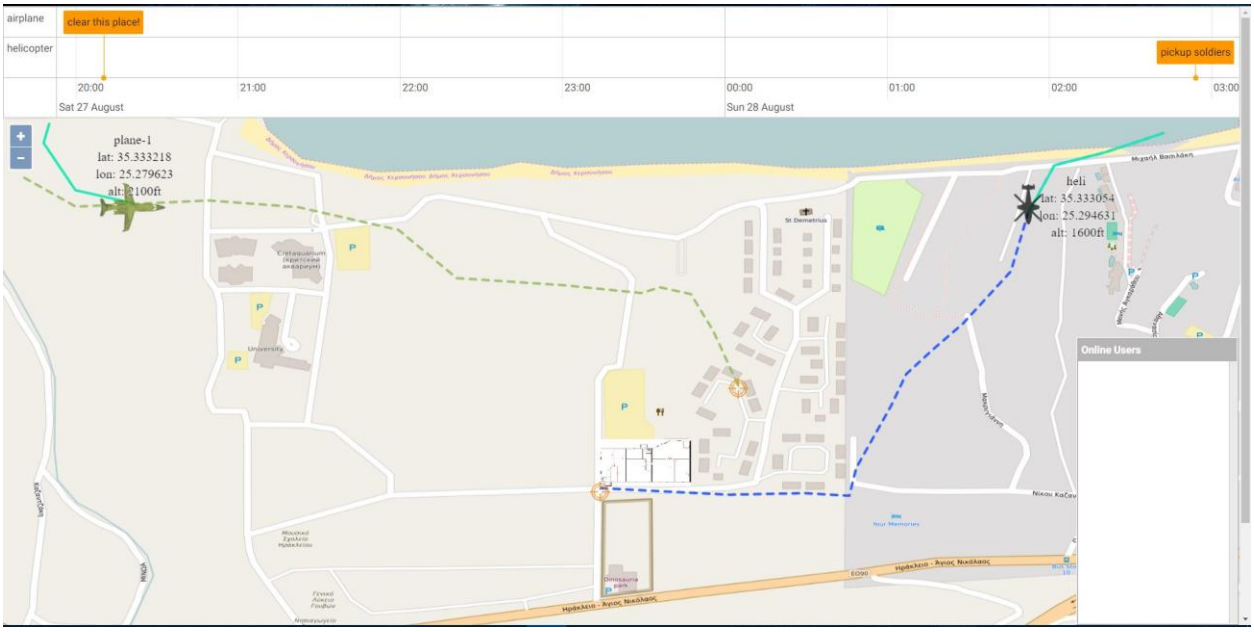
Εικόνα 42: actors movement

5.3 Επίδειξη σεναρίου

Αφού παρουσιάσαμε και εξηγήσαμε πως λειτουργεί η εφαρμογή μας στις πλατφόρμες υλοποίησης (web, android), μπορούμε να παρουσιάσουμε το σενάριο που αναφέραμε στην αρχή του κεφαλαίου. Στο σενάριο που θα δούμε παρακάτω θα έχουμε α) δύο ομάδες από 5 χρήστες, β) θα αναθέσουμε σε κάθε ομάδα από ένα task, γ) θα δείξουμε ενδεικτικά την κίνηση των χρηστών προς τα καθήκοντα που έχουν, δ) θα γίνει ενημέρωση της κατάστασης των καθηκόντων (μια από την υλοποίησης στο web και μια από το android), ε) τέλος θα δείξουμε/χρησιμοποιήσουμε και θα περιγράψουμε πως λειτουργεί το chat για την επικοινωνία/ανταλλαγή μηνυμάτων μεταξύ των χρηστών στο android και στο web.

Σημείωση: για το λόγο ότι έχουμε στη διάθεση μας μια συσκευή android, προσθέσαμε στην υλοποίηση στο web τη δυνατότητα να προσθέτει καινούργιους χρήστες, και να τους μετακινεί. Επίσης χρησιμοποιούμε την εφαρμογή "Fake GPS" στο android για να φαίνεται πως η θέση μας είναι στην τοποθεσία που συμφωνήσαμε για τη διεξαγωγή του σεναρίου μας.

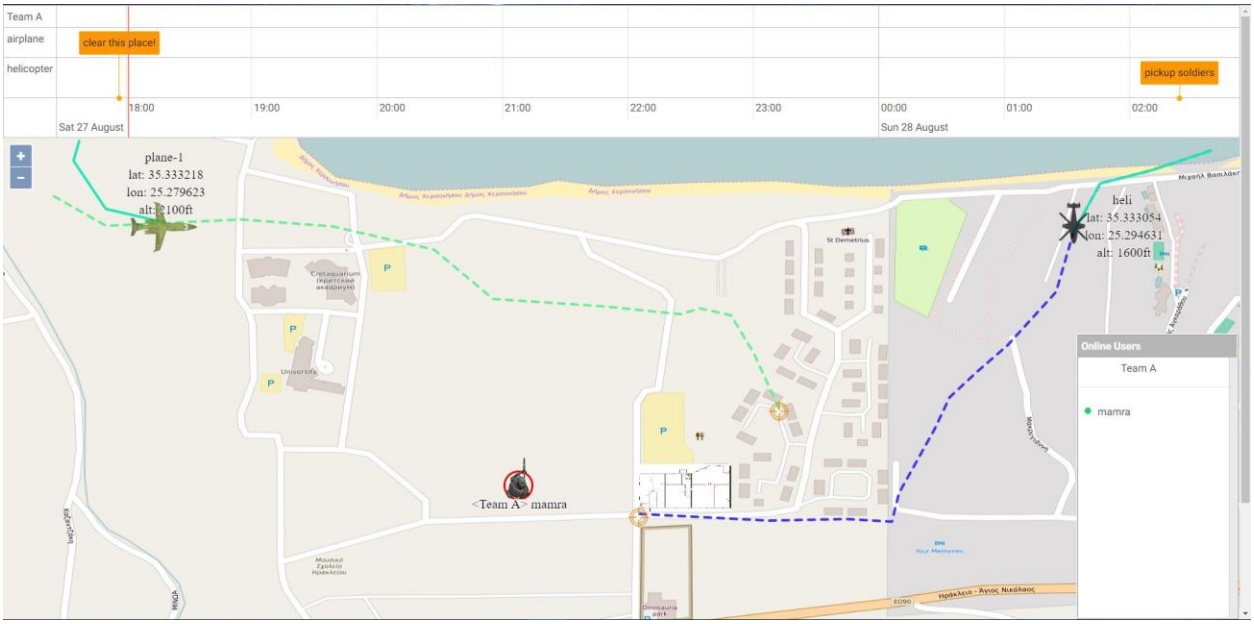
Ξεκινώντας με την υλοποίηση στο web, μόλις ολοκληρώσουμε το login θα δούμε τον χάρτη και το timeline χωρίς actors και tasks (πέρα από τους χρήστης τύπου οχήματος και τα καθήκοντα που έχουν), όπως φαίνεται και στο παρακάτω screenshot (βλ. Εικόνα 43).



Εικόνα 43: web app started

Μετά την ολοκλήρωση της σύνδεσης του επιτηρητή (web app) στο server μας, προχωράμε με τη σύνδεση του android χρήστη, ο οποίος θα είναι και αρχηγός της ομάδας του (για να έχουμε τη δυνατότητα αργότερα να ενημερώσουμε την κατάσταση του task που έχει ανατεθεί στην ομάδα του), με το που συνδεθούμε θα δούμε την θέση μας πάνω στον χάρτη, όπως μπορούμε να δούμε και στο παρακάτω screenshot (βλ. Εικόνα 44). Με το που συνδεθεί ο χρήστης του android στον server μας, ο συντονιστής/επιτηρητής θα είναι σε θέσει να δει τη θέση του πάνω στον χάρτη. Επίσης στην μπάρα του chat της υλοποίησης στο web θα δούμε μια εγγραφή με το όνομα της ομάδας και το όνομα του χρήστη (βλ. Εικόνα 45).

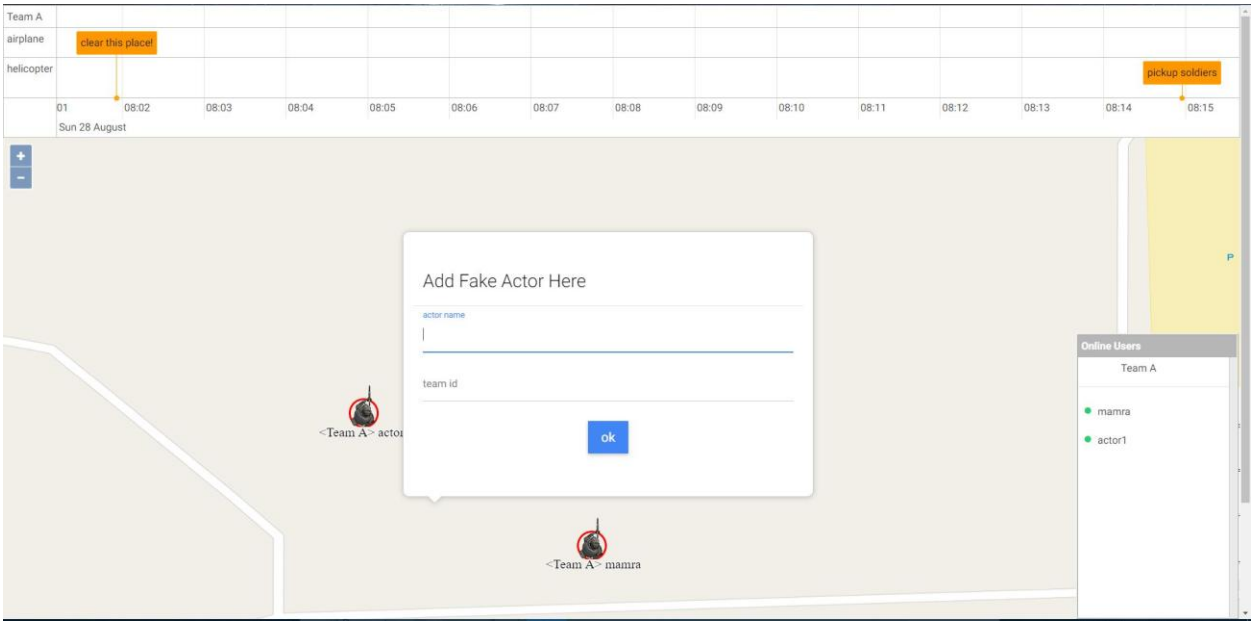
Εικόνα 44: android app started



Εικόνα 45: new android actor logged in (web app)

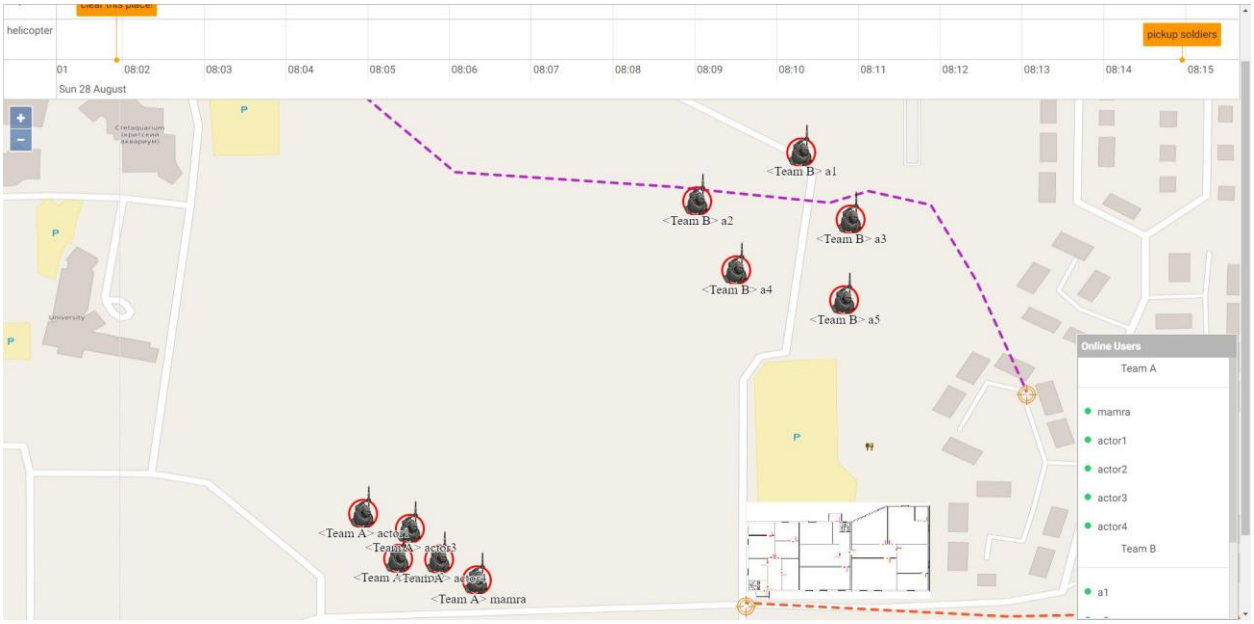
Αφού ολοκληρώσαμε τη σύνδεση των χρηστών μας, το επόμενο βήμα είναι η προσθήκη και των υπόλοιπων android actors. Για να προσθέσουμε έναν νέο χρήστη το μόνο που έχουμε να κάνουμε είναι να κάνουμε δεξί κλικ στη θέση που επιθυμούμε και στη συνέχεια, από το μενού που θα ανοίξει να επιλέξουμε την επιλογή "Add Actor Here" (χωρίς να έχουμε επιλέξει κάποιον άλλο χρήστη) (βλ. Εικόνα 46), τέλος συμπληρώνουμε τον διάλογο με το όνομα του χρήστη και την ομάδα στην οποία ανήκει (βλ. Εικόνα 47). Κάθε χρήστη που προσθέτουμε από την web εφαρμογή, προστίθεται αυτόματα και στην εφαρμογή στο android (σαν να συνδέεται ένας πραγματικός android χρήστης) (βλ. Εικόνα 48). Μόλις ολοκληρώσουμε τη διαδικασία προσθήκης 'ψεύτικων' χρηστών, θα μπορούμε να δούμε τις δύο ομάδες χρηστών που αναφέραμε παραπάνω και στις δύο εφαρμογές (βλ. Εικόνα 49-Εικόνα 50).

Εικόνα 46: add fake actor



Εικόνα 47: add fake actor dialog

Εικόνα 48: fake actors added (android app)

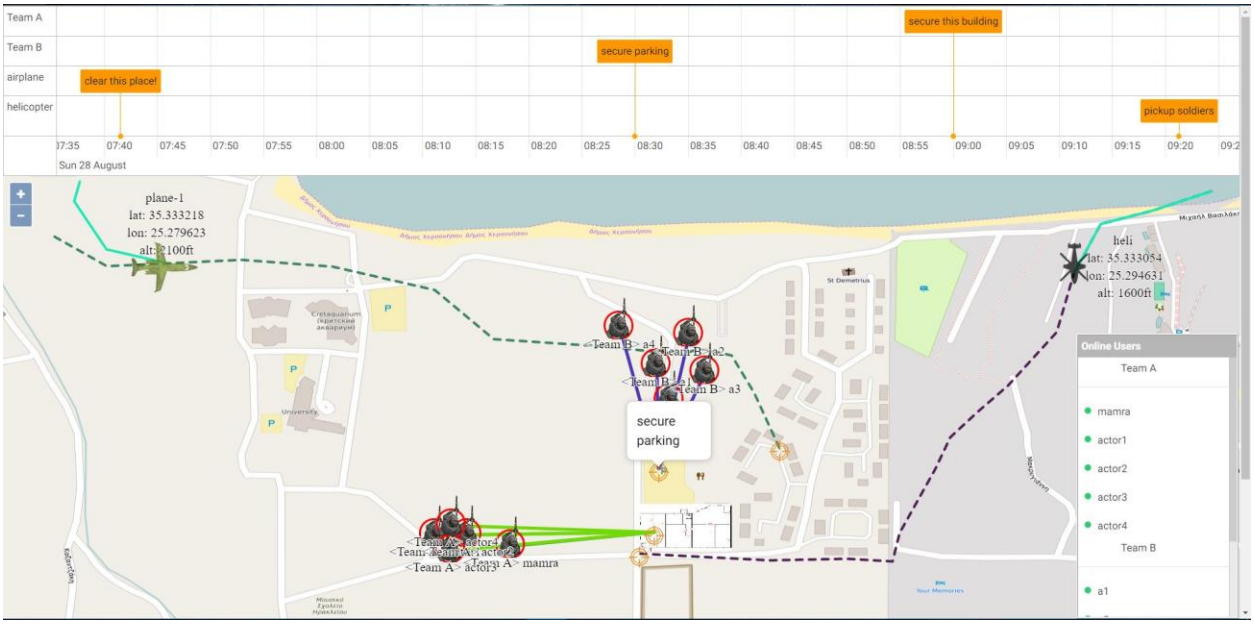


Εικόνα 49: teams ready (web app)

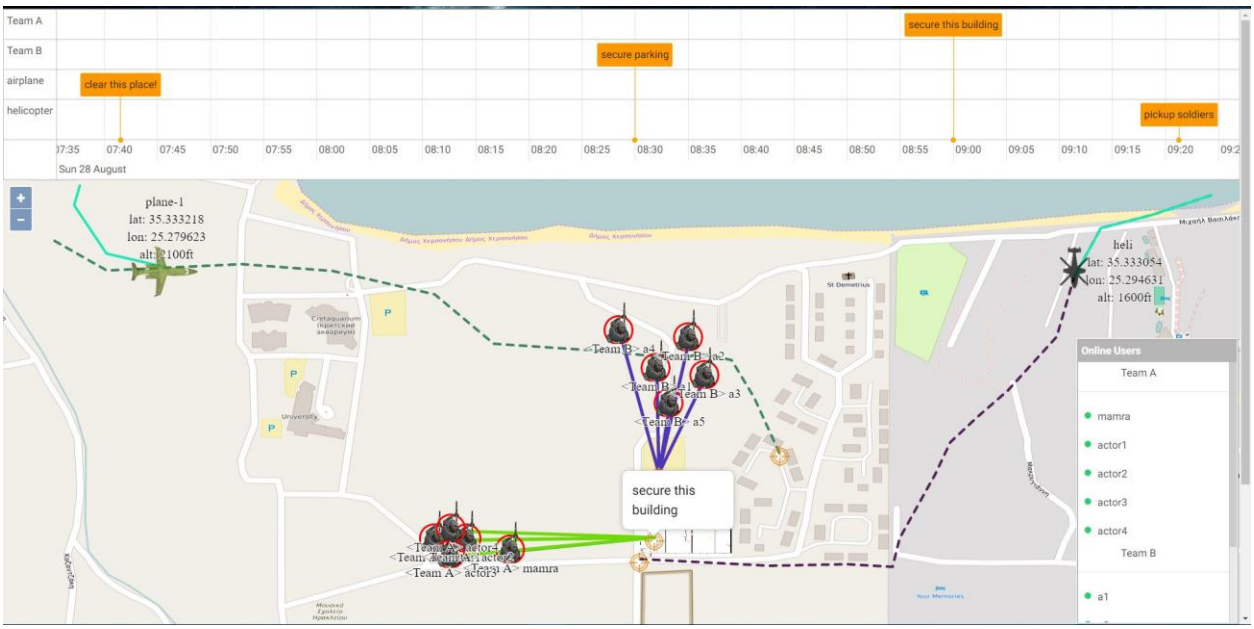
Εικόνα 50: teams ready (android app)

Προχωρώντας το επόμενο βήμα είναι η προσθήκη των καθηκόντων σε κάθε ομάδα. Ακολουθώντας τη διαδικασία που περιγράψαμε παραπάνω για την δημιουργία/προσθήκη ενός

task, θα προσθέσουμε/αναθέσουμε στην ομάδα με όνομα “Team A” το καθήκον με τίτλο/περιγραφή “secure this building” και στην “Team B” το “secure parking”, όπως θα δούμε και στα screenshots που ακολουθούν. Όπως αναφέραμε και προηγουμένως στην υλοποίηση στο android, η λίστα των καθηκόντων περιέχει μόνο το task που ανατέθηκε στην ομάδα μας (Team A).



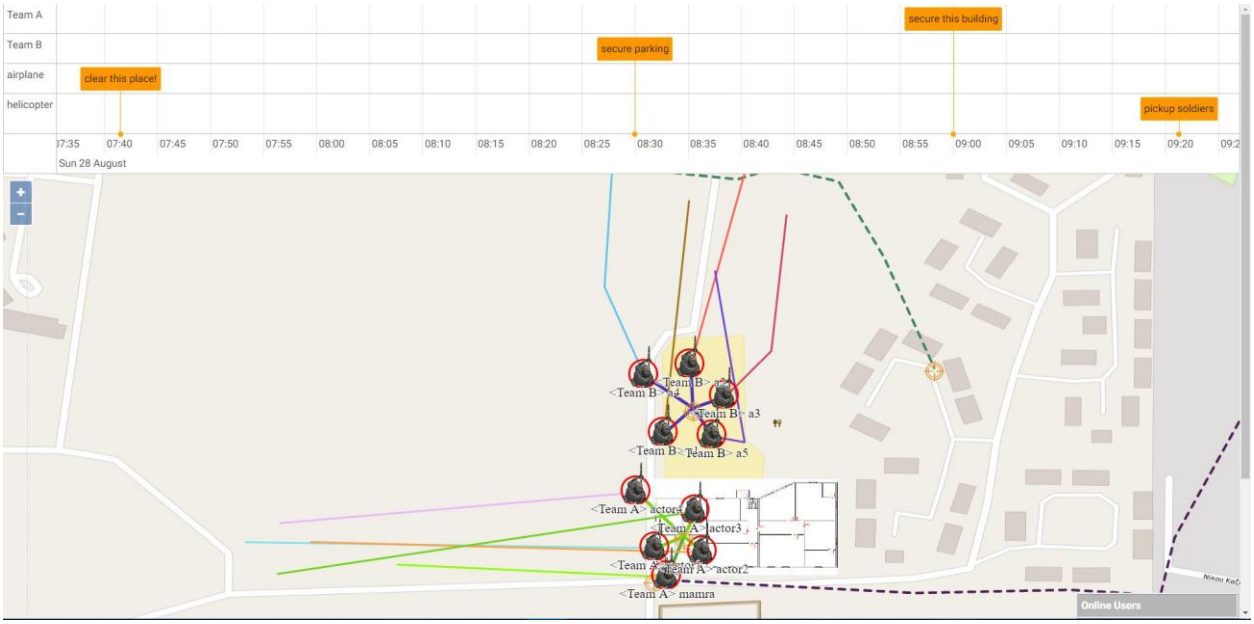
Εικόνα 51: Team B task (web app)



Εικόνα 52: Team A task (web app)

Εικόνα 53: teams tasks added (android app)

Λίγοι ώρα αργότερα όπως θα δούμε και στα screenshots που ακολουθούν, βλέπουμε τις ομάδες να έχουν προσεγγίσει το καθήκον που τους έχει ανατεθεί (βλ. Εικόνα 54 - Εικόνα 55).

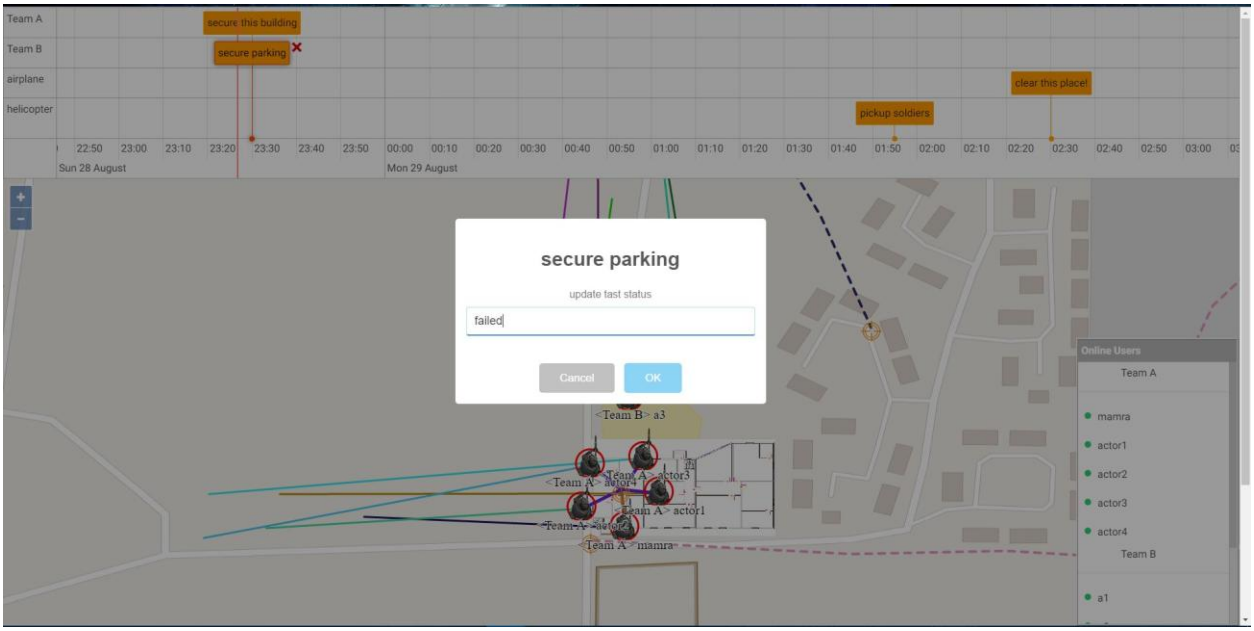


Εικόνα 54: actors moved to tasks (web app)

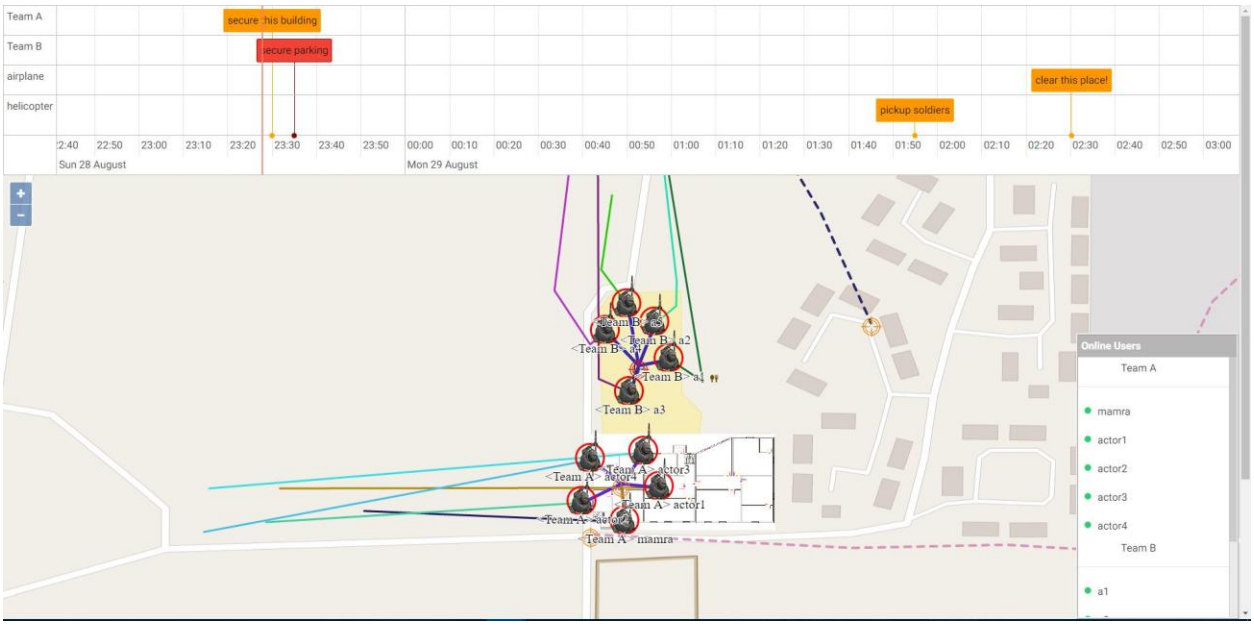
Εικόνα 55: actors moved to tasks (android app)

Έχοντας προσεγγίσει οι ομάδες μας το καθήκον που τους έχουμε αναθέσει (και αφού τους δώσαμε χρόνο για να το ολοκληρώσουν), είμαστε σε θέση να ενημερώσουμε την κατάσταση

τους. Η ενημέρωση της κατάστασης του καθήκοντος της “Team B” θα πραγματοποιηθεί από την web εφαρμογή, ενώ της “Team A” από την android. Έχοντας επιλέξει το task που αντιστοιχεί στην ομάδα B ή Team B, ακολουθούμε τα βήματα που αναφέραμε προηγουμένως για να ενημερώσουμε την κατάσταση του καθήκοντος σε “failed”, όπως μπορούμε να δούμε και στα παρακάτω screenshots (βλ. Εικόνα 56 - Εικόνα 57). Στην Εικόνα 58 βλέπουμε την ενημέρωση του task στο android (αν και δεν φαίνεται και πολύ καλά λόγο των γραμμών από τους χρήστες προς το task).



Εικόνα 56: update task status dialog (web app)



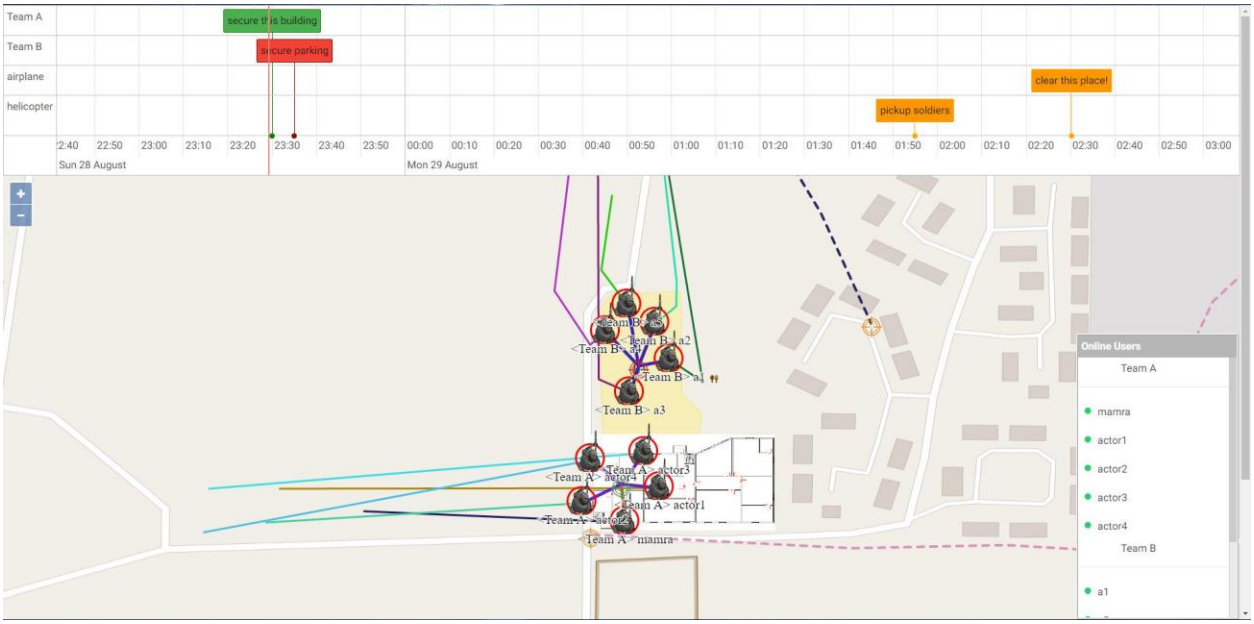
Εικόνα 57: Team B task failed (web app)

Εικόνα 58: Team B task failed (android app)

Ακολουθώντας τώρα τα βήματα που περιγράψαμε παραπάνω για το πώς θα ενημερώσουμε την κατάσταση ενός task, θα ενημερώσουμε και την κατάσταση του task της ομάδας Α ή Team A σε "succeed" (βλ. Εικόνα 59 – Εικόνα 60). Στην Εικόνα 61 μπορούμε να δούμε (πιο καθαρά από ότι στο android) την αλλαγή της κατάστασης από το timeline, αλλά και πάνω στο χάρτη.

Εικόνα 59: Team A task selected (android app)

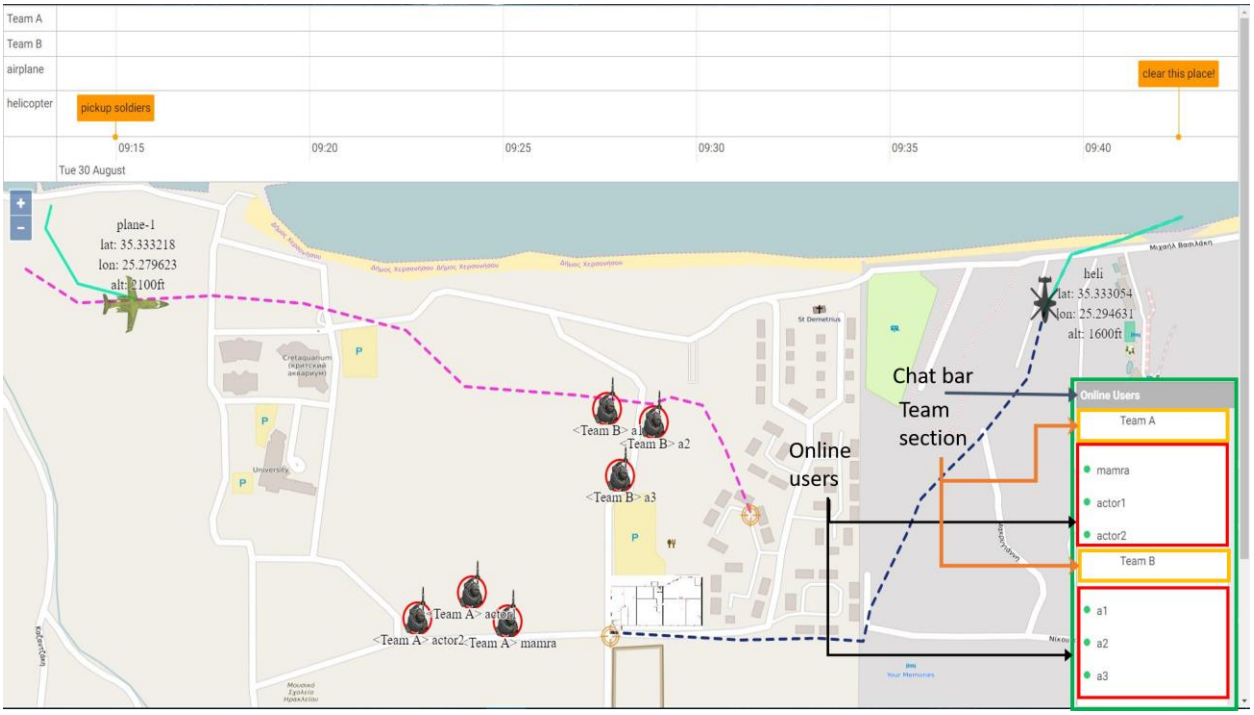
Εικόνα 60: Team A task succeed (android app)



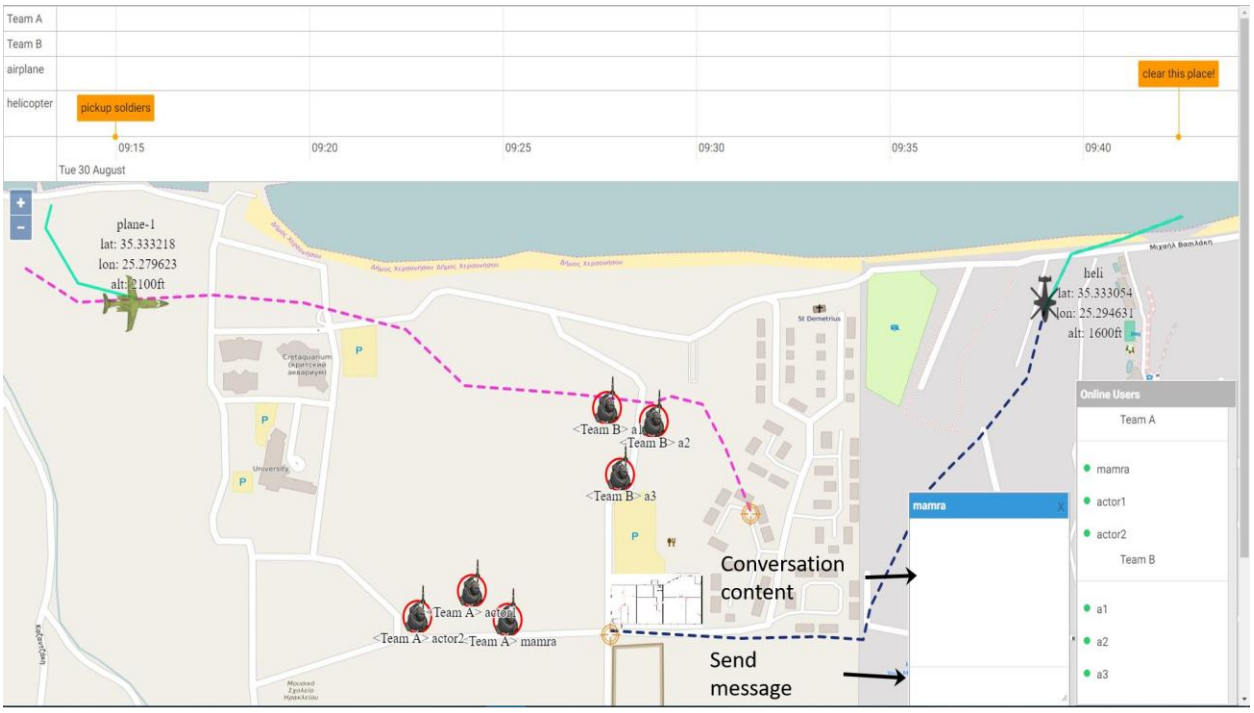
Εικόνα 61: Team A task succeed (web app)

Ως προς το chat τώρα θα αρχίσουμε με την περιγραφή της διεπαφής στο web και στο android, αρχίζοντας από το web.

Η διεπαφή του chat στο web αποτελείτε από μια μπάρα (στην δεξιά πλευρά της οθόνης), μέσα στην οποία εμφανίζονται οι συνδεμένοι χρήστες κάτω από το τμήμα που γράφει το όνομα της ομάδας τους όπως φαίνεται και στην παρακάτω εικόνα (βλ. Εικόνα 62). Για την προβολή συζήτησης ή το ξεκίνημα μιας, θα πρέπει να κάνουμε κλικ πάνω στο όνομα του χρήστη που επιθυμούμε, για να ανοίξει το πλαίσιο που περιέχει τη συζήτηση (αν υπάρχει ιστορικό μηνυμάτων) (βλ. Εικόνα 63).

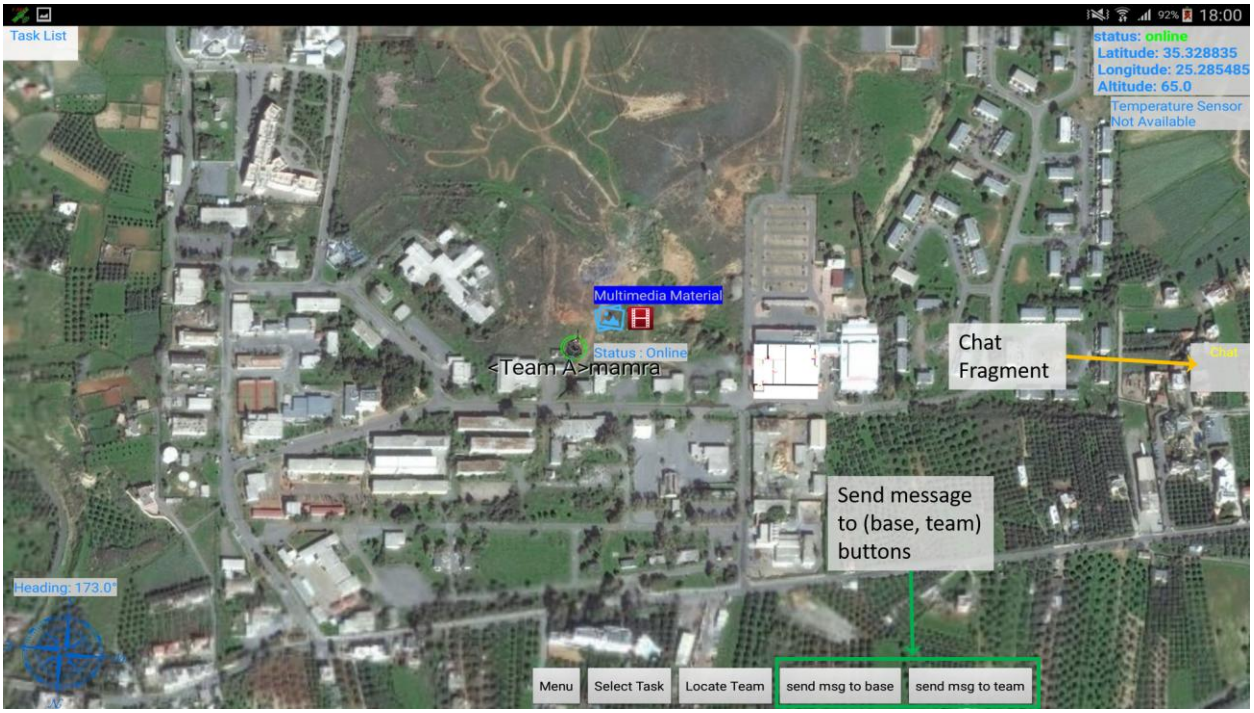


Εικόνα 62: chat GUI (web app)

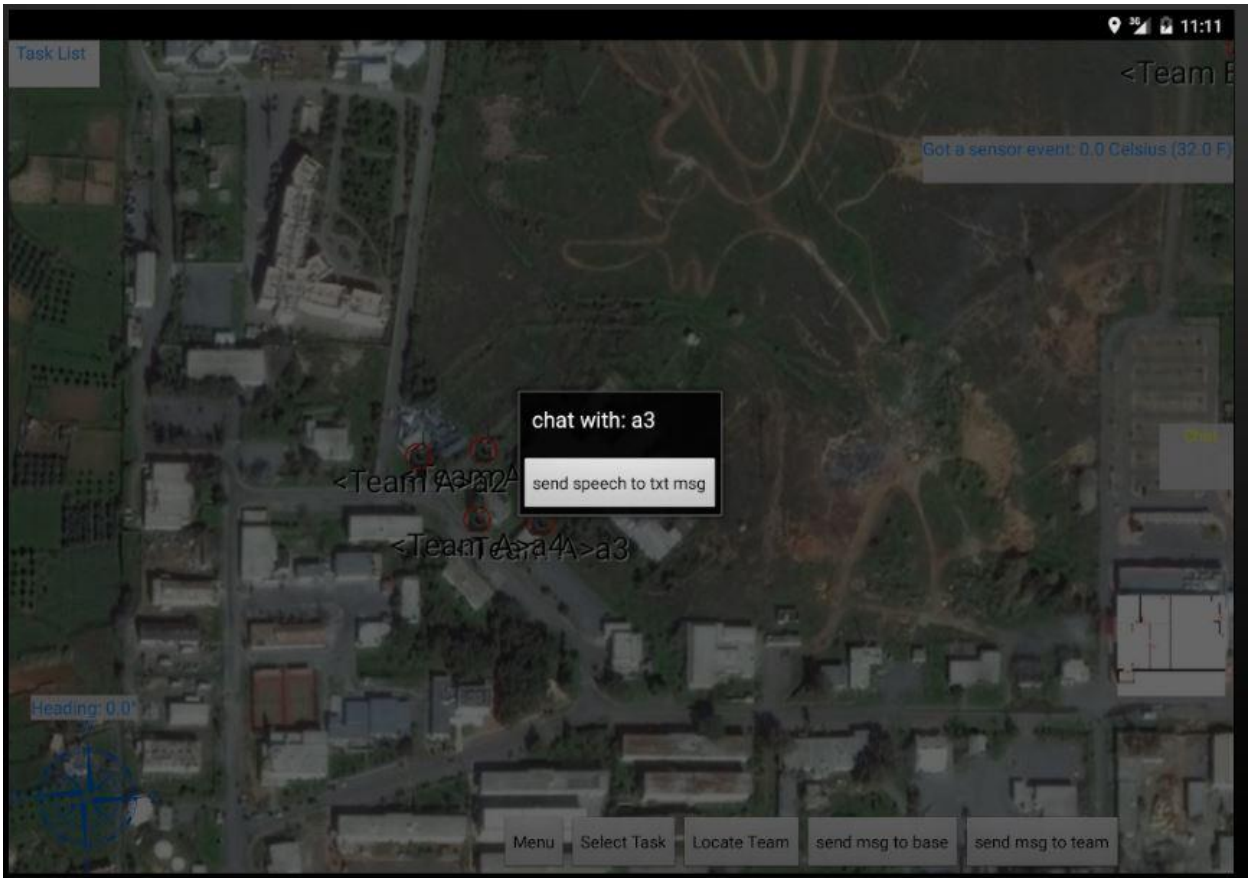


Εικόνα 63: start conversation with (web app)

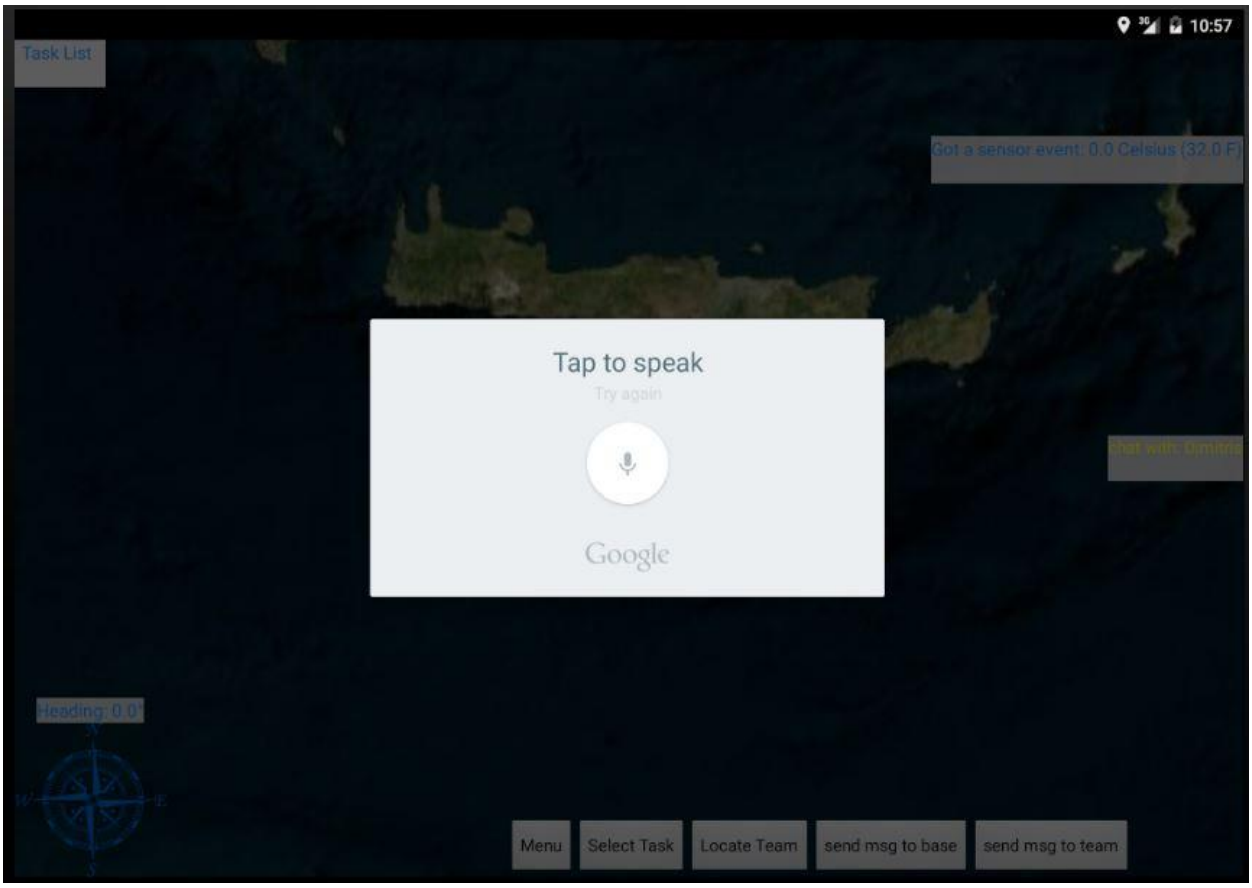
Στο android από την άλλη έχουμε μια διαφορετική προσέγγιση στον τρόπο όπου ο χρήστης στέλνει ένα μήνυμα, καθώς εκμεταλλευόμαστε τη δυνατότητα που μας παρέχει το android για μετατροπή ομιλίας σε κείμενο, καθώς ο server μας υποστηρίζει την ανταλλαγή μηνυμάτων κειμένου για συγχρονισμό, γι' αυτό το λόγο όπως θα δούμε και στα screenshots που ακολουθούν, η διεπαφή του chat μας αποτελείτε από: 1) chat fragment στο οποίο εμφανίζεται η τρέχουσα συζήτηση που έχουμε, 2) τα κουμπιά για την αποστολή μηνύματος προς την βάση (web app) ή και προς τα υπόλοιπα μέλη της ομάδας και 3) τον διάλογο που εμφανίζεται πατώντας πάνω σε κάποιο από τα μέλη της ομάδας μας.



Εικόνα 64: chat preview (android app)

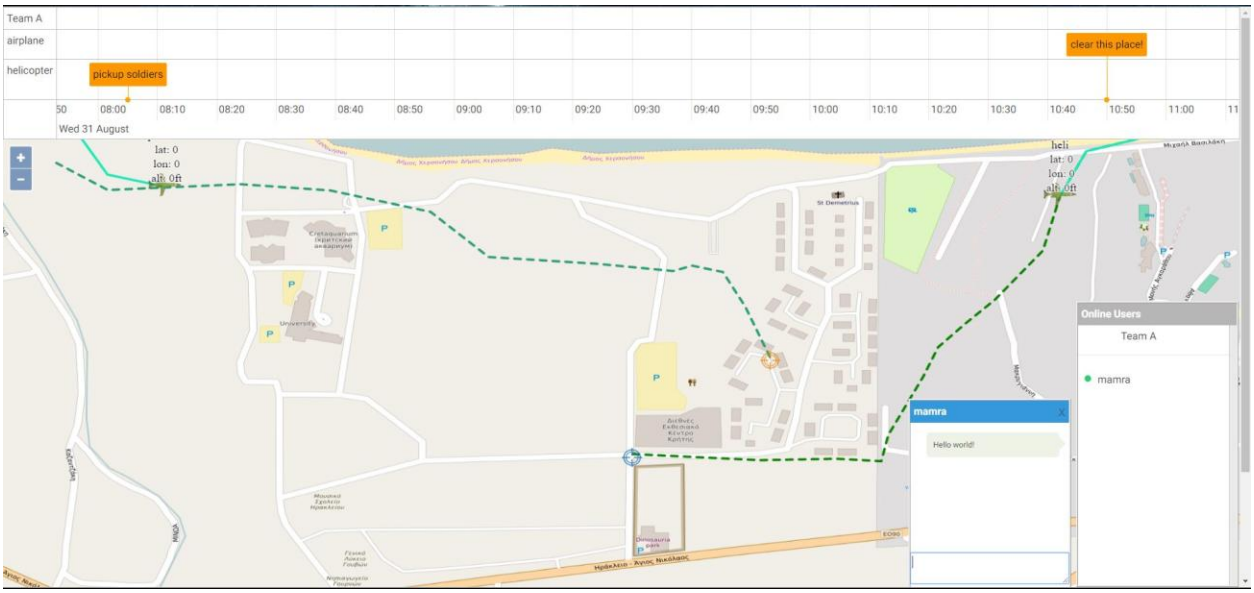


Εικόνα 65: send message to user (android app)



Εικόνα 66: android speech to text app

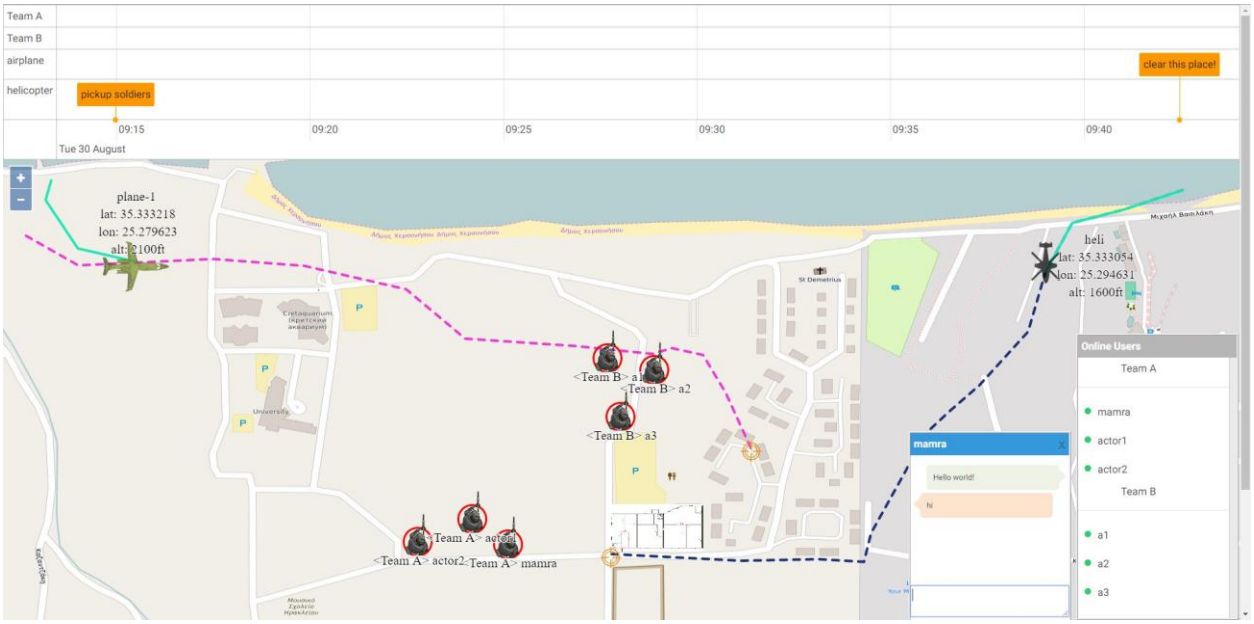
Στα παρακάτω screenshots θα δούμε τη συνομιλία ανάμεσα στον χρήστη στο android και τον χρήστη στο web.



Εικόνα 67: message sent (web app)

Εικόνα 68: message received (android app)

Εικόνα 69: message sent (android app)



Εικόνα 70: message received (web app)

6 Συμπεράσματα και μελλοντικές επεκτάσεις

Η εφαρμογή που αναπτύχθηκε, από την πλευρά του επιτηρητή/συντονιστή(υλοποίηση στο Web) μας δίνει τη δυνατότητα προβολής της σχετικής θέσης κάθε κινητού χρήστη πάνω σε ένα χάρτη, την ανάθεση καθηκόντων σε ομάδες κινητών χρηστών και την επιτήρηση της τρέχουσας κατάστασης κάθε καθήκοντος. Από την πλευρά του κινητού χρήστη (υλοποίηση στο Android) δίνει την δυνατότητα προβολής της λίστας των καθηκόντων της ομάδας του, τη σχετική θέση των μελών της ομάδας του, καθώς και των καθηκόντων. Τέλος η εφαρμογή παρέχει και τη δυνατότητα επικοινωνίας των χρηστών δια μέσου των υπηρεσιών chat που βρίσκουμε και στις δύο υλοποιήσεις.

Όσον αφορά, τα μελλοντικά σχέδια και τις δυνατότητες επέκτασης της εφαρμογής, μια πιθανή επέκταση είναι η προσθήκη μηχανισμού με τον οποίο ο χρήστης της κινητής συσκευής, θα έχει τη δυνατότητα διαμοιρασμού πολυμεσικού υλικού (εικόνας και βίντεο). Επίσης θα μπορούσαμε να επεκτείνουμε – βελτιώσουμε την υλοποίηση του chat έτσι ώστε οι χρήστες να έχουν τι δυνατότητα ανταλλαγής φωνητικών μηνυμάτων.

Αναφορές

[1] p. Dourish και S. Bly, «Portholes: supporting awareness in a distributed work group,» σε *Human Factors in Computing Systems*, New York, NY, USA, 1992.

[2] P. Dourish και V. Bellotti, «Awareness and Coordination in Shared Workspaces,» σε *Computer-supported cooperative work*, New York, NY, USA, 1992.

[3] C. Gutwin, R. Penner και K. Schneider, «Group Awareness in Distributed Software Development,» σε *ACM conference on Computer supported cooperative work*, Chicago, Illinois, 2004.

[4] k. Tolmar, O. Sandor και A. Schömer, «Supporting social awareness @ work design and experience,» σε *ACM conference on Computer supported cooperative work*, 1996.

[5] S. Bødker και E. Christiansen, «Computer Supported Cooperative Work (CSCW),» *Computer Support for Social Awareness in Flexible Work*, τόμ. 15, αρ. 1, pp. 1-28, 2006.

[6] A. Ranganathan, J. Al-Muhtadi, S. Chetan, R. Campbell και D. M. Mickunas, «MiddleWhere: a middleware for location awareness in ubiquitous computing applications,» σε *5th ACM/IFIP/USENIX international conference on Middleware*, 2004.

[7] J. F. McCarthy και E. S. Meidel, «ActiveMap: A Visualization Tool for Location Awareness to Support Informal Interactions,» σε *Handheld and Ubiquitous Computing*, Berlin Heidelberg, Springer, 2001, pp. 158-170.

[8] T. Erickson και W. A. Kelligg, «Social translucence: an approach to designing systems that support social processes,» *ACM Transactions on Computer-Human Interaction (TOCHI)*, τόμ. 7, αρ. 1, pp. 59-83, 2000.

[9] W. K. Edwards, E. D. Mynatt, k. Petersen, M. J. Spreitzer, D. B. Terry και M. M. Theimer, «Designing and implementing asynchronous collaborative applications with Bayou,» σε *User Interface Software and Technology*, 1997.

[10] N. R. Manihar και A. Prakah, «Replay by Re-execution: A Paradigm for Asynchronous Collaboration via Record and Replay of Interactive Multimedia Sessions,» 1994.

[11] M. Stefic, D. G. Bobrow, S. Lanning, D. Tatar και G. Foster, «WYSIWIS revised: early experiences with multi-user interfaces,» σε *ACM conference on Computer-supported cooperative work*, 1986.

[12] M. Roseman και S. Greenberg, «GROUPKIT: a groupware toolkit for building real-time conferencing applications,» σε *CSCW '92 Proceedings of the1992 ACM conference on Computer-supported cooperative work,* 1992.

[13] J. Hill και C. Gutwin, «The MAUI Toolkit: Groupware Widgets for Group Awareness,» *Computer Supported Cooperative Work (CSCW) ,* τόμ. 13, αρ. 5-6, pp. 539-571, 2004.

[14] J. Brandenburg, B. Byerly, T. Dobridge, J. Lin, D. Rajan και T. Roscoe, «Artefact: a framework for low-overhead Web-based collaborative systems,» σε *CSCW '98 Proceedings of the 1998 ACM conference on Computer supported cooperative work,* 1998.

[15] J. M. N. David και M. R. Borges, «COPSE-Web: An Infrastructure for Developing Web-Based Groupware Applications,» σε *Groupware: Design, Implementation, and Use*, Springer Berlin Heidelberg, 2002, pp. 275-284.

[16] U. Gall και F. J. Hauck, «Promondia: a Java-based framework for real-time group communication in the Web,» *Computer Networks and ISDN Systems,* τόμ. 29, αρ. 8-13, pp. 917-926, 1997.

[17] A. Guicking, P. Tandler και P. Avgeriou, «Agilo: A Highly Flexible Groupware Framework,» σε *Groupware: Design, Implementation, and Use,* Springer Berlin Heidelberg, 2005, pp. 49-56.

[18] J. Roth, «Mobility Support for Replicated Real-Time Applications,» σε *Innovative Internet Computing Systems*, Springer Berlin Heidelberg, 2002, pp. 181-192.

[19] V. Sacramento , M. Endler, H. K. Rubinsztejn, L. S. Lima, K. Goncalves, F. N. Nascimento και G. A. Bueno, «MoCA: A Middleware for Developing Collaborative Applications for Mobile Users,» *IEEE Distributed Systems Online,* τόμ. 5, αρ. 10, p. 2, 2004.

[20] D. Lincoln, M. Endler, S. D. J. Barbosa και J. V. Fiho, «Middleware Support for Context-Aware Mobile Applications with Adaptive Multimodal User Interfaces,» σε *Ubi-Media Computing (U-Media), 2011 4th International Conference*, Sao Paulo, 2011.

[21] L. Xuanzhe, X. Mengwei, G. Huang, T. Teng, Z. Zheng και M. Hong, «MUIT: A Middleware for Adaptive Mobile Web-based User Interfaces in WS-BPEL».

[22] M. Ionescu και I. Marsic, «SYNG: A middleware for statefull groupware in mobile environments,» σε *Collaborative Computing: Networking, Applications and Worksharing, 2007. CollaborateCom 2007. International Conference*, New York, NY, 2007.

[23] A. Neyem, S. F. Ochoa και J. A. Pino, «Supporting Mobile Collaboration with Service-Oriented Mobile Units,» σε *Groupware: Design, Implementation, and Use*, Springer Berlin Heidelberg, 2006, pp. 228-245.

[24] P. Wilson, Computer Supported Cooperative Work: An Introduction, Springer Netherlands, 1991.

[25] T. W. Malone and K. Crowston, "What is coordination theory and how can it help design cooperative work systems?," in *CSCW '90*, 1990.

[26] T. W. Malone και K. Crowston, «The interdisciplinary study of coordination,» *ACM Computing Surveys* , τόμ. 26, αρ. 1, pp. 87-119, 1994.