ΕΘΝΙΚΟ ΚΑΙ ΚΑΠΟΔΙΣΤΡΙΑΚΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΩΝ

ΣΧΟΛΗ ΘΕΤΙΚΩΝ ΕΠΙΣΤΗΜΩΝ
ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ

ΠΡΟΓΡΑΜΜΑ ΜΕΤΑΠΤΥΧΙΑΚΩΝ ΣΠΟΥΔΩΝ

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Sensor Web Enablement
Διαλειτουργικότητα των Έξυπνων Αισθητήρων με τις τεχνολογίες του Παγκόσμιου Ιστού

Μάρκος – Στυλιανός Ν. Πιτσιλός

Επιβλέπων: **Ευστάθιος Χατζηευθυμιάδης,** Επίκουρος Καθηγητής

ΑΘΗΝΑ

ΜΑΪΟΣ 2010

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Sensor Web Enablement
Διαλειτουργικότητα των Έξυπνων Αισθητήρων με τις τεχνολογίες του Παγκόσμιου Ιστού

Μάρκος – Στυλιανός Ν. Πιτσιλός

Α.Μ.: Μ0890

ΕΠΙΒΛΕΠΩΝ: **Ευστάθιος Χατζηευθυμιάδης,** Επίκουρος Καθηγητής

ΕΞΕΤΑΣΤΙΚΗ ΕΠΙΤΡΟΠΗ: **Λάζαρος Μεράκος,** Καθηγητής

Μάιος 2010

ΠΕΡΙΛΗΨΗ

Στα πλαίσια αυτής της μεταπτυχιακής εργασίας, εξετάστηκε η σουίτα πρωτοκόλλων SWE (Sensor Web Enablement) που αναπτύχθηκε από το OGC (Open Geospatial Consortium) για την προτυποποίησης της διαλειτουργικότητας των υποδομών αισθητήρων με τον Παγκόσμιο Ιστό. Το SWE περιλαμβάνει τόσο την περιγραφή νέων υπηρεσιών ιστού (Web Services) που θα παρέχουν τις απαραίτητες λειτουργίες, όσο και των γλωσσών σήμανσης (markup languages) για τα μηνύματα που αυτές θα επεξεργάζονται. Προβλέπονται υπηρεσίες για την συλλογή και ανάκτηση μετρήσεων μέσω επερώτησης (query), οι ειδοποιήσεις κατόπιν συνδρομής (subscribe – notify), όσο και προγραμματισμένες ενέργειες (tasking). Στο δεύτερο σκέλος της εργασίας αναλύθηκε μια μελέτη περίπτωσης (case study), που εστιάζει στην Υπηρεσία Παρατηρήσεων Αισθητήρων (SOS) και την Υπηρεσία Συναγερμών Αισθητήρων (SAS). Με χρήση των πρότυπων υλοποιήσεων SOS και SAS της 52°North και της πλατφόρμας ασύρματων αισθητήρων SunSPOT της Sun Microsystems, αναπτύχθηκε σε Java ένα σύστημα παρακολούθησης θερμοκρασίας σε υπό επίβλεψη χώρο, με τη δυνατότητα λήψης τόσο μεμονωμένων μετρήσεων όσο και σε εύρος χρονικού διαστήματος, τη γραφική απεικόνιση αυτών και τέλος την επιλογή αίτησης συνδρομής για ειδοποιήσεις όταν η θερμοκρασία αποκλίνει από τα επιθυμητά όρια.

ΘΕΜΑΤΙΚΗ ΠΕΡΙΟΧΗ: Δίκτυα Αισθητήρων

ΛΕΞΕΙΣ ΚΛΕΙΔΙΑ: αυτο-περιγραφή, παρατήρηση, δημοσίευση, συνδρομή, ειδοποίηση

ABSTRACT

In this Master thesis, the SWE (Sensor Web Enablement) protocol suite, developed by the OGC (Open Geospatial Consortium) for standardizing the interoperability of sensor infrastructure with the World Wide Web, was examined. SWE incorporates a description of Web Services that provide the necessary functionality, as well the definition of markup languages for the messages that these services process. Services include observation collection and querying, alerting based on the subscribe – notify model and tasking. In the second part of the thesis, a case study is presented, focusing on the Sensor Observation Service and the Sensor Alert Service. Utilizing the reference SOS and SAS implementations by 52°North and the SunSPOT wireless sensor platform by Sun Microsystems, a system was developed in Java for the purpose of monitoring temperature levels in a supervised space. The provided capabilities are retrieval of individual observations or observation collections in a specified time range, graphical representation thereof, and finally, the option of subscribing to notifications when temperature exceeds desired limits.

SUBJECT AREA: Sensor Networks

KEYWORDS: self-discription, observation, publish, subscribe, alert

ΠΕΡΙΕΧΟΜΕΝΑ

ΚΑΤΑΛΟΓΟΣ ΣΧΗΜΑΤΩΝ

ΚΑΤΑΛΟΓΟΣ ΕΙΚΟΝΩΝ

ΚΑΤΑΛΟΓΟΣ ΠΙΝΑΚΩΝ

ΠΡΟΛΟΓΟΣ

Η παρούσα εργασία εκπονήθηκε υπό την επίβλεψη του επίκουρου καθηγητή Ευστάθιου Χατζηευθυμιάδη, τον οποίο θα ήθέλε ο γράφων να ευχαριστήσει για τη συνεργασία και για την ενδιαφέρουσα θεματολογία, καθώς και του καθηγητή Λάζαρου Μεράκου. Αποτέλεσε ευκαιρία για μια πρώτη ενασχόληση του Πανεπιστημίου Αθηνών με το αντικείμενο του Sensor Web Enablement και αφορμή για μια υλοποίηση με το χαρακτήρα του proof of concept. Η εκπόνηση της έγινε εξ ολοκλήρου στην Αθήνα και έδωσε την αφορμή για εξοικείωση του γράφοντα με τεχνολογίες όπως το SunSPOT API της Sun Microsystems όσο και με τις Υπηρεσίες Ιστού (Web Services) και το πρωτόκολλο XMPP, αγγίζοντας έτσι ένα εύρος θεμάτων και δίνοντας σφαιρική εικόνα των αλληλεπιδράσεων που περιλαμβάνονται στη σουίτα πρωτοκόλλων του OGC για την διαλειτουργικότητα των Έξυπνων Αισθητήρων με τις Τεχνολογίες του Παγκοσμίου Ιστού.

1. ΕΙΣΑΓΩΓΗ

Οι αισθητήρες αποτελούν κτήμα της ανθρωπότητας εδώ και αρκετές δεκαετίες. Παρακολουθούν την ποιότητα του μίγματος στους κινητήρες των αυτοκινήτων μας, μετρούν τη θερμοκρασία στους καταψύκτες μας, ελέγχουν τη στάθμη των υδάτων στις δεξαμενές μας και πολλά άλλα. Το σίγουρο είναι ότι καθημερινά περιβαλλόμαστε από εκατοντάδες από αυτούς, χωρίς συχνά να συνειδητοποιούμε την ύπαρξη τους και το γεγονός ότι συμβάλλουν στη διαμόρφωση της ποιότητας ζωής μας όπως την έχουμε συνηθίσει.

Με την ανάπτυξη της Ηλεκτρονικής και της Πληροφορικής, τα αξιόλογα επίπεδα σμίκρυνσης (micronization) έχουν επιτρέψει την ύπαρξη σημαντικής υπολογιστικής ισχύος σε ολοένα και μικρότερες συσκευές. Πλεόν έχουμε στη διάθεση μας έξυπνους αισθητήρες (smart sensors) με πιο ισχυρούς επεξεργαστές και περισσότερη μνήμη από ότι είχαν προσωπικοί υπολογιστές παλαιότερων γενεών, και αυτό σε έναν όγκο όσο ένα κουτί σπίρτα. Οι αισθητήρες αυτοί μπορούν να υλοποιούν πολύπλοκα πρωτόκολλα, να επεξεργάζονται οι ίδιοι μερικώς ή ολικώς τις μετρήσεις που λαμβάνουν από το περιβάλλον, έως και να φιλοξενούν ένα περιορισμένο λειτουργικό σύστημα.

Η ενσωμάτωση δυνατοτήτων ασύρματης δικτύωσης επιτρέπει τη κατανομή τους σε πιο ευρείες περιοχές, σχηματίζοντας ασύρματα δίκτυα αισθητήρων (WSN – Wireless Sensor Networks). Ταυτόχρονα βέβαια εισάγει και νέες προκλήσεις, όπως π.χ. την ανάγκη εξοικονόμησης ενέργειας για απομεμακρυσμένους κόμβους του δικτύου που δεν έχουν τη δυνατότητα να επαναφορτίζονται συχνά ή και καθόλου. Ο αυξημένος χώρος διευθύνσεων (address space) που εισάγει το IPv6, πολλές τάξεις μεγέθους μεγαλύτερος από του IPv4, καθιστά θεωρητικά δυνατό να διευθυνσιοδοτηθεί κάθε κόμβος κάθε δικτύου αισθητήρων παγκοσμίως, πλησιάζοντας έτσι το όραμα του «Internet των πραγμάτων» (“Internet of things”) [1], όπου κάθε σημαντική ή ασήμαντη συσκευή είναι εν δυνάμει προσπελάσιμη μέσω διαδικτύου.

Το αυξημένο ενδιαφέρον για την διαλειτουργικότητα μεταξύ δικτύων αισθητήρων και της πλούσιας προϋπάρχουσας υποδομής (δια)δικτύων τύπου TCP/IP και δη της σημαντικότερης ίσως εφαρμογής αυτών, τον Παγκόσμιο Ιστό (WWW - World Wide Web), μας φέρνει στην έννοια του Sensor Web, ενός δικτύου διασυνδεδεμένων υποδομών αισθητήρων για την υποστήριξη γεωγραφικών υπηρεσιών πληροφόρησης (GIS – Geographical Information Systems) και βασισμένων στη θέση υπηρεσιών (LBS – Location Bases Services).

Σε ένα δίκτυο αισθητήρων, αναφερόμαστε σε κάθε μεμονωμένη δικτυωμένη συσκευή ως κόμβο (node / mote). Κάθε κόμβος είναι εφοδιασμένος με πομποδέκτη, κάποια διεπαφή δικτύου (network interface), επεξεργαστή ή ελεγκτή, τροφοδοσία ρεύματος, και φυσικά ένα ή περισσότερα είδη ψηφιακών ή αναλογικών αισθητήρων, καθώς και μετατροπείς αναλογικού σε ψηφιακό σήμα (ADC) ή το ανάποδο (DAC).

2. SENSOR WEB ENABLEMENT

Με τον όρο Sensor Web Enablement (SWE) αναφερόμαστε στην οργανωμένη προσπάθεια του Open Geospatial Consortium (OGC) για δημιουργία ανοιχτών προτύπων διαλειτουργικότητας (interoperability) για τους συνδεδεμένους στο διαδίκτυο αισθητήρες. Η προτυποίηση περιλαμβάνει την καθιέρωση διεπαφών τέτοιων ώστε οι ετερογενείς αισθητήρες ή σύνθετες αρχιτεκτονικές αυτών να μπορούν με ομοιόμορφο τρόπο να δημοσιεύουν (publish) τις μετρήσεις τους σε υπηρεσίες του Ιστού (web services), από όπου θα είναι προσπελάσιμες με χρήση διαδεδομένων πρωτοκόλλων όπως το HTTP και το SOAP, μέσα από μηνύματα κωδικοποιημένα σε XML.

Το SWE περιλαμβάνει, μεταξύ άλλων, γλώσσες σήμανσης (markup) για την περιγραφή των μετρήσεων και συμβάντων που σχετίζονται με τους αισθητήρες και τα υπό παρατήρηση φαινόμενα, υπηρεσίες Ιστού για τη συλλογή και παρακολούθηση της ροής δεδομένων, καθώς και τις αντίστοιχες προγραμματιστικές διεπαφές εφαρμογών (API - Application Programming Interface), που καθιστούν δυνατή την διάδραση με υπάρχουσες ή νέες πλατφόρμες λογισμικού.

Το SWE στοχεύει στη δημιουργία ενός επιπέδου αφαίρεσης (abstraction) των εφαρμογών που συνεργάζονται με αισθητήρες από την υποδομή αισθητήρων καθαυτή. Μέσα από τα πρωτόκολλα που εισάγει, οι αισθητήρες πρέπει να μπορούν να αυτοπεριγραφούν (self-description) δημοσιοποιώντας τα μεταδεδομένα (metadata) τους, να ανακαλυφθούν (discovery) από ενδιαφερόμενους χρήστες και διεργασίες, να εντολοδοτηθούν (tasking), να συλλέξουν δεδομένα σε πραγματικό χρόνο (real-time) ή σωρευτικά (time series) και να υποστηρίζουν εγγραφή σε μηχανισμούς ειδοποίησης όταν πληρούνται συνθήκες «συναγερμού» (subscribe – alert).

2.1 Γλώσσες σήμανσης

Οι γλώσσες σήμανσης που χρησιμοποιούνται στα πλαίσια του SWE είναι η O&M, η SensorML και η TransducerML.

Η γλώσσα O&M (Observations and Measurements) [2] χρησιμοποιείται για την περιγραφή των παρατηρήσεων και μετρήσεων, καθώς και της ακρίβειας αυτών. Οι μετρήσεις και παρατηρήσεις μπορούν να είναι τόσο ποσότικες, όσο και ποιοτικές. Μια ποσοτική μέτρηση θα μπορούσε να είναι π.χ. μια στάθμη θερμοκρασίας, ενώ μια ποιοτική παρατήρηση θα μπορούσε να είναι μια εκτίμηση της κατάστασης του ουρανού, π.χ. «συννεφώδης».

Τα μεταδεδομένα ενός αισθητήρα, όπως οι δυνατότητες μέτρησης, επεξεργασίας και προγραμματισμού που διαθέτει, η θέση του στο χώρο, κ.ά. περιγράφονται στη γλώσσα SensorML (Sensor Model Language) [3].

Η γλώσσα TransducerML ή TML (Transducer Markup Language) [4] μπορεί επίσης να χρησιμοποιηθεί για την περιγραφή των αισθητήριων μονάδων, καθώς και για τη μετάδοση δεδομένων από και προς αυτούς με συνεχή τρόπο (streaming).

Οι τρέχουσες εκδόσεις των προτύπων για τις άνωθι γλώσσες είναι:

- **O&M:** 1.0
- **SensorML:** 1.0.0
- **TML:** 1.0.0

2.2 Υπηρεσίες του SWE

Το SWE καταμερίζει τη λειτουργικότητα του σε τέσσερις επιμέρους υπηρεσίες Ιστού (web services): το SOS, SAS, SPS και WNS.

Η Υπηρεσία Παρατηρήσεων Αισθητήρων SOS (Sensor Observation Service) [5] παρέχει μια διεπαφή για την προσπέλαση σε μετρήσεις και παρατηρήσεις αισθητήρων, το φιλτράρισμα αυτών σύμφωνα με κάποια κριτήρια, καθώς και την πρόσβαση στα μεταδεδομένα των ίδιων των αισθητήρων. Ουσιαστικά, αποτελεί τον ενδιάμεσο μεταξύ του καταναλωτή γεωγραφικών δεδομένων και της αποθήκης δεδομένων παρατήρησης (observation repository) ή της (σχεδόν) πραγματικού χρόνου ροής δεδομένων από τις αισθητήριες μονάδες (near real-time sensor channel). Από την άποψη του καταναλωτή-πελάτη, η πρωτοβουλία για την ανάκτηση μετρήσεων έγκειται στον ίδιον, οπότε πρόκειται για μια υλοποίηση υπηρεσίας κατά το Data-Pull model.

Η Υπηρεσία Συναγερμών Αισθητήρων SAS (Sensor Alert Service) [5] παρέχει τη δυνατότητα σε αρχιτεκτονικές αισθητήρων να «διαφημίζουν» τα προς μέτρηση φαινόμενα που υποστηρίζουν, και να δημοσιεύουν (publish) τις μετρήσεις τους με σκοπό να συγκριθούν με κριτήρια ειδοποίησης που μπορεί να θέσει ο καταναλωτής. Αν τα κριτήρια πληρούνται, για παράδειγμα ξεπεραστεί κάποια τιμή κατωφλίου, ο πελάτης που εκδήλωσε ενδιαφέρον μέσα από συνδρομητική αίτηση (subscription), ειδοποιείται (alert) για αυτό το συμβάν. Από την οπτική γωνία του καταναλωτή-πελάτη, αν εξαιρεθεί η διαδικασία συνδρομητικής αίτησης, η πρωτοβουλία προώθησης μετρήσεων προέρχεται από το Sensor Web, οπότε έχουμε μια περίπτωση υπηρεσίας που ακολουθεί το Data-Push model.

Η Υπηρεσία Προγραμματισμού Αισθητήρων SPS (Sensor Planning Service) [7] επιτρέπει σε αισθητήρες να δηλώσουν κατά πόσον υποστηρίζουν εντολοδότηση και σύνθετες αποστολές (tasking), και στους πελάτες του πληροφοριακού συστήματος να αναθέσουν σε μια υποδομή αισθητήρων μια προσχεδιασμένη ακολουθία ενεργειών, σε καθορισμένο χρονικό διάστημα και επιθυμητό πλήθος επαναλήψεων. Πριν την ανάθεση ελέγχεται η πρακτική επιτευξιμότητα (feasibility) της προγραμματισμένης συλλογής δεδομένων.

Τέλος, η Υπηρεσία Ειδοποιήσεων Ιστού WNS (Web Notification Service) [8] επιτρέπει στο SAS και SPS να αποστέλλουν τις ειδοποιήσεις τους με ασύγχρονο τρόπο μέσω SMS, email ή αλλων μεθόδων.

Οι τρέχουσες εκδόσεις των προτύπων και συστάσεων για τις παραπάνω υπηρεσίες είναι:

- **SOS:** 1.0.0
- **SAS:** 0.9.0
- **SPS:** 1.0.0
- **WNS:** 0.0.9

Πρέπει να σημειώσουμε ότι το SOS και SPS έχουν σταθεροποιηθεί και βρίσκονται σε ώριμο στάδιο (δικαιολογώντας το χαρακτηρισμό «έκδοση 1.0.0»), ενώ το SAS παρέμεινε σε επίπεδο σύστασης (best practice) και αναμένεται να αντικατασταθεί από την Υπηρεσία Συμβάντων Αισθητήρων SES (Sensor Event Service) [9], που θα καλύπτει τις αδυναμίες του. Το WNS βρίσκεται επίσης σε επίπεδο σύστασης.

2.3 Ορολογία GIS / OGC και WSN

Πρωτού εμβαθύνουμε στην έννοια του Sensor Web Enablement, είναι χρήσιμο να εξοικειωθούμε με βασικούς όρους από το χώρο των Γεωγραφικών Πληροφοριακών Συστημάτων και των δικτύων αισθητήρων, ειδικότερα των ασύρματων.

Γεωγραφικό Γνώρισμα (Feature of interest)

Ένα γεωγραφικό γνώρισμα είναι μια οντότητα του πραγματικού κόσμου, π.χ. μια γέρφυρα, μια αποθήκη, ένα κτίριο, που χαρακτηρίζεται από μια δεδομένη θέση (location) στο χώρο.

Βάση Χωρικών Δεδομένων (Spatial Database)

Μια βάση χωρικών δεδομένων είναι μια βάση δεδομένων βελτιστοποιημένη για την αποθήκευση των χωρικών δεδομένων που περιγράφουν γεωγραφικά γνωρίσματα. Τα χωρικά δεδομένα αυτά είναι ως επί το πλείστον συλλογές σημείων (points), γραμμών (lines) και πολυγώνων (polygons), που προσεγγίζουν τη θέση και μορφή της οντοτήτας στο χώρο, στο βαθμό ακρίβειας που απαιτείται για μια δεδομένη εφαρμογή.

Φαινόμενο (Phenomenon)

Ως φαινόμενο καλείται οτιδήποτε μπορεί να παρατηρηθεί, όπως η βαρύτητα, η θερμότητα, η χημική σύσταση, η ηλεκτρική φόρτιση κτλ.

Παρατηρούμενη Ιδιότητα (Observed Property)

Οι παρατηρούμενες ιδιότητες ενός γεωγραφικού γνωρίσματος είναι τα διαθέσιμα προς παρατήρηση φαινόμενα στην περιοχή του γνωρίσματος αυτού. Παραδείγματος χάρη, το φαινόμενο «στάθμη ύδατος» κάτω από μια γέφυρα, αποτελεί παρατηρούμενη ιδιότητα του γνωρίσματος «γέφυρα».

Παρατήρηση (Observation) και Μέτρηση (Measurement)

Η παρατήρηση αποτελεί την ενέργεια της παρακολούθησης ενός φαινομένου με σκοπό την παραγωγή μιας εκτίμησης για τιμή μιας παρατηρούμενης ιδιότητας αυτού. Ένα συγκεκριμένο συμβάν ή στιγμιότυπο της ενέργειας αυτής καλείται επίσης μια παρατήρηση (observation). Μια παρατήρηση περιλαμβάνει ένα αποτέλεσμα (result) που αποτελεί την τιμή (value), ποιοτική ή ποσοτική, που εξήχθη.

Ειδικότερα, μέτρηση (measurement) είναι μια παρατήρηση της οποίας η τιμή είναι βαθμωτή (scalar) σε κάποιο σύστημα αναφοράς, και με κάποια μονάδα μέτρησης.

Μια ομάδα από συναφών ειδών παρατηρήσεων που προσφέρονται από ένα σύστημα ονομάζεται προσφορά παρατηρήσεων (observation offering).

Διαδικασία (Procedure)

Με τον όρο διαδικασία αναφερόμαστε σε μια μέθοδο, αλγόριθμο, όργανο ή συνδυασμό αυτών, που συμμετέχει στην ενέργεια της συλλογής παρατηρήσεων.

Αισθητήρας (Sensor)

Οι αισθητήρες υπάγονται στην κατηγορία των μορφοτροπέων (transducers), δηλαδή συσκευών που μετατρέπουν ενέργεια από μία μορφή σε άλλη, π.χ. ηλεκτρική, μηχανική, ηλεκτρομαγνητική, κ.ά. Οι αισθητήρες συγκεκριμένα χαρακτηρίζονται από τη δυνατότητα παρατήρησης ενός φαινομένου μετατρέποντας εξωτερικά ερεθίσματα του περιβάλλοντος, όπως θερμότητα, κίνηση, επίπεδο υγρασίας, φωτεινή ακτινοβολία, ακουστικά κύματα κτλ. σε ηλεκτρικό, ως επί το πλείστον, σήμα, είτε αναλογικό, είτε ψηφιακό, μέσω αναλογικοψηφιακής μετατροπής (ADC - Analog to Digital Conversion).

Συσκευές που πραγματοποιούν την αντίθετη μετατροπή, π.χ. από ηλεκτρικό σήμα σε κίνηση καλούνται ενεργοποιητές (actuators).

Ανάλογα με το αν έχουν δυνατότητα κίνησης (mobility), κατατάσσονται σε στατικούς (in situ) και κινητούς (mobile). Με κριτήριο τη διεπαφή με το κεντρικό σημείο συλλογής δεδομένων ή και τη συνδεσιμότητα με μια πηγή ρεύματος, οι αισθητήρες, όπως και όλες οι συσκευές, χωρίζονται σε ενσύρματους και ενσύρματους.

Ασύρματος Αισθητήρας (Wireless Sensor)

Για λόγους οικονομικής ή πρακτικής δυσκολία ενσύρματης δικτύωσης και τροφοδοσίας ρεύματος, οι ασύρματοι αισθητήρες έχουν αποκτήσει ιδιάζουσα θέση στον κόσμο των γεωγραφικών συστημάτων πληροφόρησης.

Με τους ασύρματους αισθητήρες μπορούν να αποφευχθούν πολλά προβλήματα χωροταξίας που θα προκαλλούνταν από την ύπαρξη καλωδίων, ενώ επίσης μπορούν να τοποθετηθούν σε περιοχές δύσβατες για τον άνθρωπο και με τρόπους που είναι πρακτικά ή οικονομικά ανέφικτοι για ενσύρματες διατάξεις, όπως π.χ. εναέρια ρίψη. Ας μην ξεχνάμε ότι συχνά σε δίκτυα το κόστος της καλωδίωσης μπορεί συχνά να κυριαρχεί στον οικονομικό προϋπολογισμό.

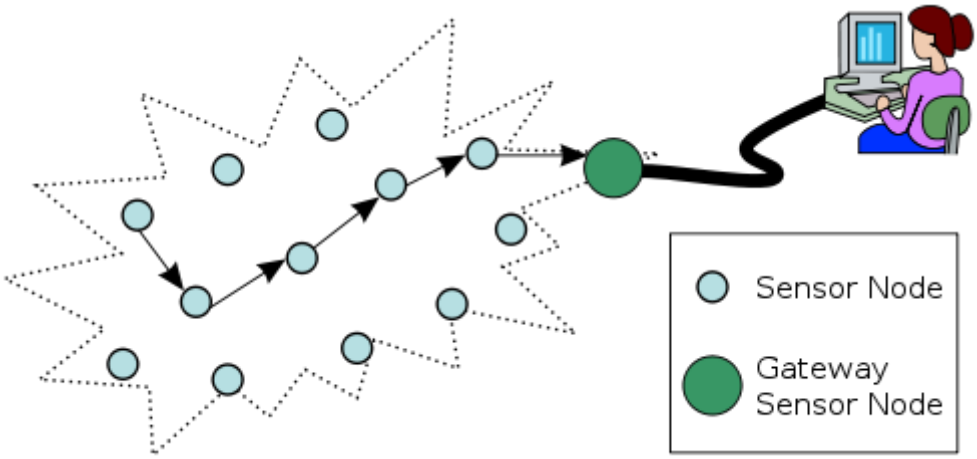
Η χρήση της εναέριας διεπαφής (aerial interface) ασφαλώς εισάγει όλες τις επιπτώσεις των φυσικών νόμων που διέπουν τις ασύρματες μεταδόσεις, όπως παρεμβολές, εξασθένηση (path loss), αυξημένες ενεργειακές δαπάνες, κ.ά. Οι μεμονωμένοι κόμβοι πρέπει να είναι ως επί το πλείστον μικροί σε μέγεθος, παράγοντας ο οποίος περιορίζει τη χωρητικότητα του συσσωρευτή (μπαταρίας) που μπορούν να φέρουν, συνεπώς και τα ενεργειακά τους αποθέματα.

Εξαιτίας της περιορισμένης ενέργειας, αλλά και της ανάγκης για μείωση των παρεμβολών, η ισχύς εκπομπής δεν μπορεί να είναι πολύ μεγάλη. Κατά συνέπεια, η εμβέλεια των εκπομπών είναι εν γένει μικρή, οπότε κάθε κόμβος συνήθως είναι αδύνατον να έχει την πλήρη εποπτεία του δικτύου στο οποίο ανήκει. Επίσης, ο ενεργειακός παράγοντας, σε συνδυασμό με την ανάγκη για λειτουργία των αισθητήρων για διάρκεια μηνών ή ετών, ενδεχομένως χωρίς επαναφόρτιση, θέτει ένα άνω φράγμα στην υπολογιστική ισχύ που μπορεί να φιλοξενείται.

Δημιουργείται έτσι η πρακτική ανάγκη για συσσώρευση και συνάθροιση δεδομένων (data aggregation) και μερική ή ολική επεξεργασία (processing) αυτών από περιορισμένο υποσύνολο των κόμβων του δικτύου, και όχι όλων καθώς αυτό είναι υπολογιστικά ασύμφορο. Επίσης γίνεται κατανομή ακριβών ή εξειδικευμένων αισθητήρων, όπως πχ. δέκτη GPS, σε λίγους ή μόνο έναν κόμβο ή σε διατάξεις που δεν είναι απαραίτητα ασύρματοι αισθητήρες όπως οι υπόλοιποι (π.χ. υπολογιστές, ελεγκτές κτλ.). Τα σημεία συνάθροισης δεδομένων λέγονται συγκεντρωτές (sinks). Δεδομένα μπορεί να συμψηφιστούν ή να υπολογιστεί η μέση τιμή τους ώστε να προωθηθεί αυτή παραπέρα, για να μειωθεί ο όγκος της πληροφορίας προς επεξεργασία και επαναμετάδοση. Είναι κοινή πρακτική ο συγκεντρωτής (ή οι συγκεντρωτές) του δικτύου

ασύρματων αισθητήρων να αναλαμβάνει και το ρόλο της πύλης (gateway) προς ετερογενή δίκτυα, π.χ. TCP/IP.

Όλα τα παραπάνω έχουν δώσει ώθηση για την ανάπτυξη ειδικών πρωτοκόλλων για τη δρομολόγηση (routing) των ληφθέντων μετρήσεων, τη γενικότερη διασπορά της πληροφορίας (information dissemination) και τη ρύθμιση της ενεργής κατάστασης ή μη των κόμβων στο δίκτυο. Τα πρωτόκολλα κάνουν χρήση διάφορων μετρικών και λαμβάνουν υπόψη το γεγονός ότι οι κόμβοι συχνά δεν μπορούν να επικοινωνήσουν όλοι μεταξύ τους, ενώ πιθανότατα είναι αναγκασμένοι να απενεργοποιούν τακτικά τους πομποδέκτες ή και περισσότερα τμήματα του υλικού (hardware) τους για διατήρηση ενέργειας (power conservation).



Σχήμα 1: Αναπαράσταση ενός δικτύου ασύρματων αισθητήρων

Αξίζει να σημειωθεί ότι στα πλαίσια του OGC / SWE, δεν υπάρχει εστίαση σε κάποια κατηγορία αισθητήρων ειδικότερα, ενσύρματων ή ασύρματων. Οι προδιαγραφές καλύπτουν όλα τα είδη αισθητήρων μέσω ενός επιπέδου αφαίρεσης, αντιμετωπίζοντας τους ως διαδικασίες, καθότι ο λογικός ρόλος τους είναι να συλλέγουν παρατηρήσεις. Τα ειδικά χαρακτηριστικά κάθε αισθητήρα μπορούν βέβαια να αποτυπωθούν σε μεταδεδομένα (metadata), στο βαθμό που αυτό εξυπηρετεί τις ανάγκες του SWE.

Στην παρούσα εργασία, θα δοθεί βάση κυρίως στους ασύρματους αισθητήρες, καθώς η τεχνολογία SunSPOTs της Sun Microsystems που αποτελεί γόνιμο έδαφος πειραματισμού για την ακαδημαϊκή κοινότητα, υπάγεται στην κατηγορία αυτή. Να σημειωθεί ότι στην πράξη η έκφραση «ασύρματος αισθητήρας» χρησιμοποιείται καταχρηστικά για όλο τον ασύρματο κόμβο, παρά το γεγονός ότι αυτός μπορεί να φέρει πολλούς αισθητήρες. Ένας πιο δόκιμος όρος για να αναφερθούμε σε έναν κόμβο εφοδιασμένο με αισθητήρες είναι «αισθητήριος κόμβος».

2.4 Διαγράμματα ροής μηνυμάτων

Στη συνέχεια θα εξοικειωθούμε με τις βασικές ροές μηνυμάτων για τις Push και Pull-based υπηρεσίες του SWE, δηλαδή το SOS και το SAS. Πρόκειται για απλά πρωτόκολλα αίτησης-απάντησης (request – response).

Για κάθε είδος υπηρεσίας θα διαχωρίσουμε την ανάλυση μας στα σκέλη που αφορούν στον παραγωγό παρατηρήσεων / μετρήσεων (sensor data) και τον καταναλωτή, αντίστοιχα. Θα εστιάσουμε στις πιο απαραίτητες λειτουργίες, αναφέροντας τις δευτερεύουσες πιο συνοπτικά.

2.4.1 Data pull – SOS

Το SOS, όπως και οι άλλες υπηρεσίες του SWE περιλαμβάνουν έναν αριθμό από ενέργειες που είναι υποχρεωτικό (mandatory) να υποστηρίζονται από τις υλοποιήσεις των υπηρεσιών αυτών και από τις συνεργαζόμενες εφαρμογές, και από μερικές προαιρετικές (optional) λειτουργίες.

Οι λειτουργίες που θα μας απασχολήσουν είναι οι ακόλουθες:

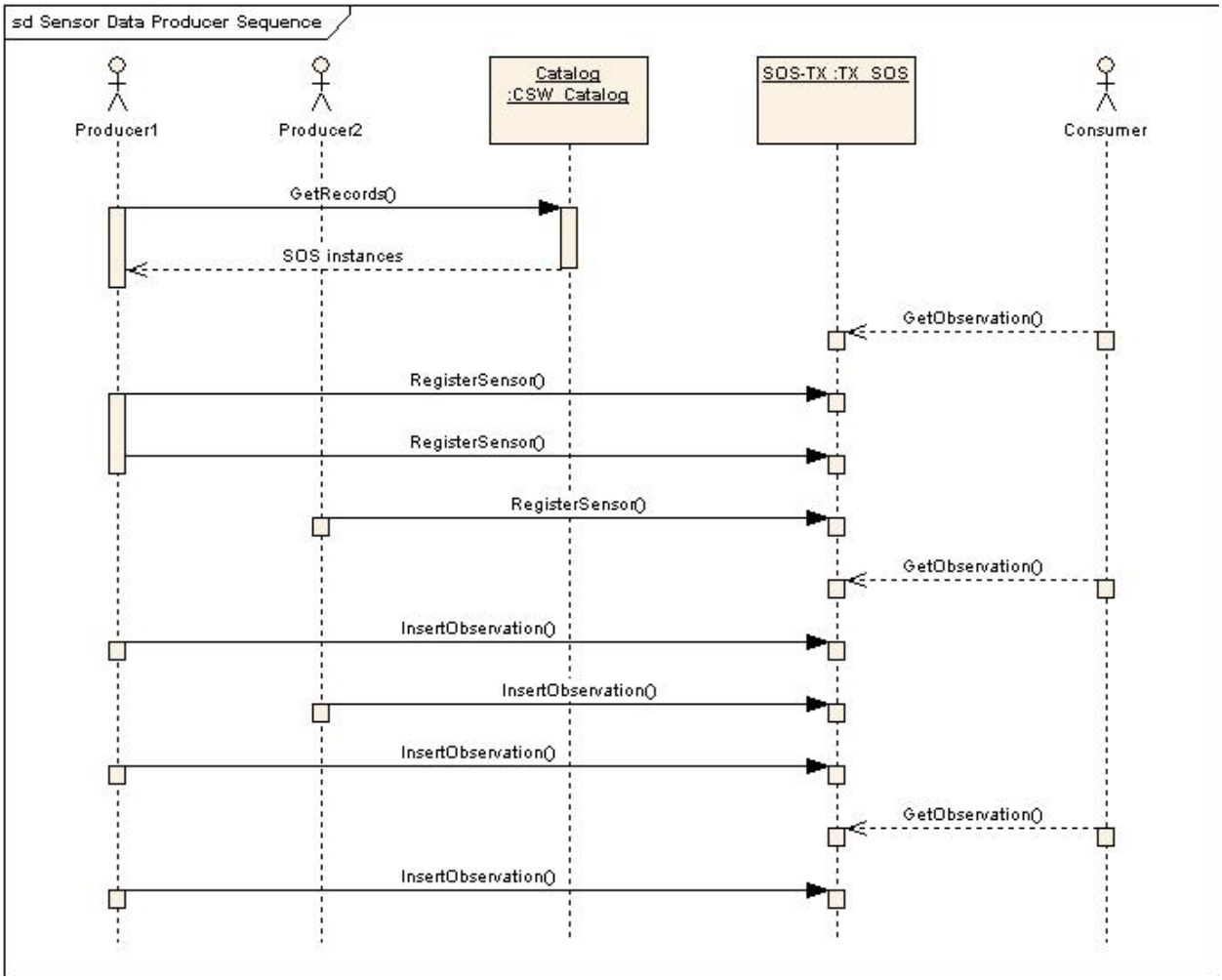
- **RegisterSensor** (προαιρετική): Για καταχώρηση ενός αισθητήρα στο μητρώο του SOS.
- **InsertObservation** (προαιρετική): Εισαγωγή μιας παρατήρησης / μέτρησης στη βάση δεδομένων του SOS.
- **GetCapabilities** (υποχρεωτική): Επερώτηση για τις δυνατότητες του SOS.
- **GetObservation** (υποχρεωτική): Επερώτηση για παρατηρήσεις αποθηκευμένες στη βάση δεδομένων του SOS.

2.4.1.1 Από την πλευρά του παραγωγού

Όσο και αν φαίνεται περίεργο, οι βασικές λειτουργίες για τον παραγωγό δεδομένων στο SOS τυπικά θεωρούνται προαιρετικές. Οι προδιαγραφές του SOS δεν απαγορεύουν στην υποδομή αισθητήρων να ενημερώνει το SOS web service μέσω κάποιας αυθαίρετης (custom) διεπαφής. Σε αυτήν την περίπτωση ο πάροχος του SOS θα πρέπει να ταυτίζεται με τον πάροχο των δεδομένων ή θα πρέπει να προσυμφωνείται μεταξύ τους η συγκεκριμένη διεπαφή.

Για την πιο γενική περίπτωση που ο πάροχος του SOS και ο παραγωγός δεδομένων δεν ταυτίζονται, το πρότυπο του SOS περιλαμβάνει και μια επέκταση που υποστηρίζει την εγγραφή αισθητήρων και την καταχώρηση μετρήσεων με αυστηρώς ορισμένα μηνύματα. Η επέκταση αυτή ονομάζεται SOS-T Profile (SOS Transactional) και περιέχει τις αιτήσεις RegisterSensor και InsertObservation.

Η διάδραση του παραγωγού δεδομένων με το SOS φαίνεται στο παρακάτω διάγραμμα:



Σχήμα 2: Διάγραμμα ακολουθίας μηνυμάτων μεταξύ παραγωγού δεδομένων και SOS

Εν συντομία, ο παραγωγός δεδομένων Producer1 συμβουλεύεται έναν κατάλογο (Catalog / Directory) για να ανακαλύψει το SOS, κατά τη συνήθη διαδικασία ανακάλυψης υπηρεσιών Ιστού. Κατόπιν, αποστέλλει στο SOS αιτήσεις εγγραφής για τους αισθητήρες που διαχειρίζεται μέσω της λειτουργίας RegisterSensor. Εφόσον αυτές πραγματοποιηθούν επιτυχώς, τότε ο Producer1 προχωρά στην αποστολή παρατηρήσεων εμφωλευμένων σε InsertObservation αιτήσεων. Στο εν λόγω διάγραμμα δεν φαίνονται οι απαντήσεις, αλλά κάθε επιτυχημένη αίτηση RegisterSensor συνοδεύεται ακολουθείται από μια απάντηση RegisterSensorResponse που περιέχει το αναγνωριστικό (ID / URI) με το οποίο έχει καταχωρηθεί ο αισθητήρας στη βάση του SOS. Παρομοίως, για κάθε επιτυχημένη καταχώρηση παρατήρησης επιστρέφεται το αναγνωριστικό της (ID) μέσα σε ένα InsertObservationResponse. Αποτυχημένες αιτήσεις λαμβάνουν ως απάντηση ένα μήνυμα εξαίρεσης (exception) με ένα σχετικό κωδικό ή περιγραφή σφάλματος.

Ο παραγωγός Producer2 τυχαίνει να γνωρίζει τη θέση (URL) του SOS εκ των προτέρων, οπότε μπορεί να ξεκινήσει απευθείας με την εγγραφή των αισθητήρων του και την αποστολή μετρήσεων. Στο δεξί μέρος της εικόνας φαίνεται ένας ενδιαφερόμενος πελάτης που αιτείται μετρήσεων από το SOS μέσω της GetObservation μεθόδου. Η πλευρά του καταναλωτή θα αναλυθεί περισσότερο στη συνέχεια, αλλά πρώτα θα δούμε σε μεγαλύτερο βάθος τις λειτουργίες RegisterSensor και InsertObservation.

2.4.1.1.1 RegisterSensor

Όπως φάνηκε από τα παραπάνω, η λειτουργία RegisterSensor επιτρέπει σε έναν παραγωγό δεδομένων να καταχωρήσει έναν αισθητήρα ή γενικότερα μια διαδικασία (procedure) σε ένα SOS. Η αίτηση περίεχει ως επί το πλείστον μεταδεδομένα του αισθητήρα, την τρέχουσα του θέση, την πληροφορία αν είναι κινητός ή όχι, ποια φαινόμενα μπορεί να παρατηρήσει (phenomenon, observed property) και για ποια γεωγραφικά γνωρίσματα (feature of interest).

Η καταχώρηση αυτή είναι απαραίτητη, γιατί το SOS δεν μπορεί να δεχτεί μέσω GetObservation μετρήσεις από αισθητήρα που δεν είναι εγγεγραμμένος στο μητρώο του. Τα μεταδεδομένα είναι προσπελάσιμα από τον πελάτη μέσω μιας επερώτησης GetCapabilities, όπου οι δυνατότητες (capabilities) του SOS ουσιαστικά προκύπτουν από το συνόλο των δυνατοτήτων των αισθητήρων που είναι εγγεγραμμένοι σε αυτό. Τα επιμέρους τμήματα μιας RegisterSensor αίτησης είναι χρήσιμο να τα δούμε σε ένα πρακτικό παράδειγμα.

Ένα δείγμα RegisterSensor σε XML είναι το εξής:

```
<?xml version="1.0" encoding="UTF-8"?>
<RegisterSensor service="SOS" version="1.0.0"
    xmlns="http://www.opengis.net/sos/1.0"
    xmlns:swe="http://www.opengis.net/swe/1.0.1"
    xmlns:ows="http://www.opengeospatial.net/ows"
    xmlns:xlink="http://www.w3.org/1999/xlink"
    xmlns:gml="http://www.opengis.net/gml"
    xmlns:ogc="http://www.opengis.net/ogc"
    xmlns:om="http://www.opengis.net/om/1.0"
    xmlns:sml="http://www.opengis.net/sensorML/1.0.1"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://www.opengis.net/sos/1.0
    http://schemas.opengis.net/sos/1.0.0/sosRegisterSensor.xsd
    http://www.opengis.net/om/1.0

http://schemas.opengis.net/om/1.0.0/extensions/observationSpecialization_over
ride.xsd">

    <!-- Sensor Description parameter; Currently, this has to be a sml:System
-->
    <SensorDescription>

        <sml:SensorML version="1.0.1">
        <sml:member>
        <sml:System xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"    >

            <!--sml:identification element must contain the ID of the sensor-
->
```

```
            <sml:identification>
                <sml:IdentifierList>
                    <sml:identifier>
                        <sml:Term
definition="urn:ogc:def:identifier:OGC:uniqueID">

<sml:value>urn:ogc:object:feature:Sensor:IFGI:ifgi-sensor-1</sml:value>
                        </sml:Term>
                    </sml:identifier>
                  </sml:IdentifierList>
            </sml:identification>


            <!-- sml:capabilities element has to contain status and mobility
information -->
            <sml:capabilities>
                <swe:SimpleDataRecord>
                    <!-- status indicates, whether sensor is collecting data
at the moment (true) or not (false) -->
                    <swe:field name="status">
                        <swe:Boolean>
                            <swe:value>true</swe:value>
                        </swe:Boolean>
                    </swe:field>
                    <!-- status indicates, whether sensor is mobile (true) or
fixed (false) -->
                    <swe:field name="mobile">
                        <swe:Boolean>
                            <swe:value>false</swe:value>
                        </swe:Boolean>
                    </swe:field>
                </swe:SimpleDataRecord>
            </sml:capabilities>


            <!-- last measured position of sensor -->
            <sml:position name="sensorPosition">
                <swe:Position referenceFrame="urn:ogc:def:crs:EPSG:4326">
                    <swe:location>
                        <swe:Vector gml:id="STATION_LOCATION">
                            <swe:coordinate name="easting">
                                <swe:Quantity axisID="x">
                                    <swe:uom code="degree"/>
                                    <swe:value>7.52</swe:value>
                                </swe:Quantity>
                            </swe:coordinate>
```

```
                        <swe:coordinate name="northing">
                            <swe:Quantity axisID="y">
                                <swe:uom code="degree"/>
                                <swe:value>52.90</swe:value>
                            </swe:Quantity>
                        </swe:coordinate>
                        <swe:coordinate name="altitude">
                            <swe:Quantity axisID="z">
                                <swe:uom code="m"/>
                                <swe:value>52.0</swe:value>
                            </swe:Quantity>
                        </swe:coordinate>
                    </swe:Vector>
                </swe:location>
            </swe:Position>
        </sml:position>

            <!-- list containing the input phenomena for this sensor system
-->
            <sml:inputs>
                <sml:InputList>
                    <sml:input name="waterlevel">
                        <swe:ObservableProperty
definition="urn:ogc:def:phenomenon:OGC:1.0.30:waterlevel"/>
                    </sml:input>
                </sml:InputList>
            </sml:inputs>

            <!-- list containing the output phenomena of this sensor
system; ATTENTION: these phenomena are parsed and inserted into the database;
they have to contain offering elements to determine the correct offering for
the sensors and measured phenomena -->
            <sml:outputs>
                <sml:OutputList>
                    <sml:output name="waterlevel">
                        <swe:Quantity
definition="urn:ogc:def:phenomenon:OGC:1.0.30:waterlevel">
                            <gml:metaDataProperty>
                                <offering>
                                    <id>GAUGE_HEIGHT</id>
                                    <name>gauge height in
Muenster</name>
                                </offering>
                            </gml:metaDataProperty>
```

```
                                        <swe:uom code="cm"/>
                                    </swe:Quantity>
                                </sml:output>
                            </sml:OutputList>
                        </sml:outputs>

                        <!-- description of components of this sensor system; these are
currently not used by the 52N SOS -->
                        <sml:components>
                            <sml:ComponentList>
                                <sml:component name="gaugeSensor">
                                    <sml:Component>
                                        <sml:identification>
                                            <sml:IdentifierList>
                                                <sml:identifier>
                                                    <sml:Term
definition="urn:ogc:def:identifier:OGC:uniqueID">

<sml:value>urn:ogc:object:feature:Sensor:water_level_sensor</sml:value>
                                                    </sml:Term>
                                                </sml:identifier>
                                            </sml:IdentifierList>
                                        </sml:identification>
                                        <sml:inputs>
                                            <sml:InputList>
                                                <sml:input name="gaugeHeight">
                                                    <swe:ObservableProperty
definition="urn:ogc:def:phenomenon:OGC:1.0.30:waterlevel"/>
                                                </sml:input>
                                            </sml:InputList>
                                        </sml:inputs>
                                        <sml:outputs>
                                            <sml:OutputList>
                                                <sml:output name="gaugeHeight">
                                                    <swe:Quantity
definition="urn:ogc:def:phenomenon:OGC:1.0.30:waterlevel">
                                                        <swe:uom code="cm"/>
                                                    </swe:Quantity>
                                                </sml:output>
                                            </sml:OutputList>
                                        </sml:outputs>
                                    </sml:Component>
                                </sml:component>
                            </sml:ComponentList>
```

```
                </sml:components>
            </sml:System>
            </sml:member>
            </sml:SensorML>
        </SensorDescription>

        <!-- ObservationTemplate parameter; this has to be an empty measurement
at the moment, as the 52N SOS only supports Measurements to be inserted -->
        <ObservationTemplate>
            <om:Measurement>
                <om:samplingTime/>
                <om:procedure/>
                <om:observedProperty/>
                <om:featureOfInterest></om:featureOfInterest>
                <om:result uom=""></om:result>
            </om:Measurement>
        </ObservationTemplate>

</RegisterSensor>
```

Εν ολίγοις, το παραπάνω XML έγγραφο δηλώνει ότι:

- Επιθυμείται η εγγραφή ενός αισθητήρα με το αναγνωριστικό (sensor ID / unique ID / URI) urn:ogc:object:feature:Sensor:IFGI:ifgi-sensor-1.
- Ο αισθητήρας είναι ενεργός (active) και κινητός (mobile).
- Η τρέχουσα του θέση (sensorPosition) είναι 7.52 μοίρες Ανατολικά, 52.9 μοίρες Βόρεια και σε υψόμετρο 52 μέτρων.
- Σαν είσοδος του αισθητήρα (input phenomenon / observable property) λαμβάνεται η στάθμη του νερού (water level).
- Σαν έξοδο ο αισθητήρας προσφέρει (offering) τη μέτρηση της στάθμης νερού (GAUGE_HEIGHT) με μονάδα μέτρησης τα εκατοστόμετρα (cm).
- Ο κόμβος περιέχει προφανώς ως συστατικό (component) από πλευράς υλικού (hardware) έναν αισθητήρα νερού (water_level_sensor).
- Οι μετρήσεις που αποστέλλει ο αισθητήρας αναμένονται σε μια φόρμα που ορίζεται από δεδομένο πρότυπο (template), η οποία περιέχει τα εξής πεδία:
    - Χρονική στιγμή δειγματοληψίας (sampling time)
    - Όνομα διαδικασίας (procedure), με άλλα λόγια αναγνωριστικό του αισθητήρα
    - Παρατηρούμενη ιδιότητα (observed property)
    - Γνώρισμα ενδιαφέροντος (feature of interest)
    - Αποτέλεσμα παρατήρησης (result), δηλαδή η μέτρηση καθαυτή.

Η απάντηση στο αίτημα αυτό μπορεί να έχει την ακόλουθη μορφή:

```
<sos:RegisterSensorResponse
xsi:schemaLocation="http://www.opengis.net/sos/1.0
http://schemas.opengis.net/sos/1.0.0/sosAll.xsd">
      <sos:AssignedSensorId>urn:ogc:object:feature:Sensor:IFGI:ifgi-sensor-
1</sos:AssignedSensorId>
</sos:RegisterSensorResponse>
```

Η πληροφορία που περιέχεται στην απάντηση είναι ότι ο αισθητήρας καταχωρήθηκε με το αναγνωριστικό urn:ogc:object:feature:Sensor:IFGI:ifgi-sensor-1.

Αν η ενέργεια είναι αποτυχής, η απάντηση μπορεί να είναι μια αναφορά λάθους (exception report), που περιέχει το συγκεκριμένο λόγο αποτυχίας. Για παράδειγμα αν ο αισθητήρας έχει ήδη εγγραφεί στο μητρώο του SOS προηγουμένως, μια διπλότυπη εγγραφή θα απορριφθεί με ένα μήνυμα σαν το ακόλουθο:

```
<ows:ExceptionReport version="1.0.0"
xsi:schemaLocation="http://schemas.opengis.net/ows/1.1.0/owsExceptionReport.x
sd">
    <ows:Exception exceptionCode="NoApplicableCode">
        <ows:ExceptionText>
            Sensor with ID: 'urn:ogc:object:feature:Sensor:IFGI:ifgi-sensor-
1' is already registered at this SOS!
        </ows:ExceptionText>
    </ows:Exception>
</ows:ExceptionReport>
```

Όπως φαίνεται, η εγγραφή απορρίφθηκε γιατί αισθητήρας με το ίδιο αναγνωριστικό υπάρχει ήδη στη βάση δεδομένων.

2.4.1.1.2 InsertObservation

Ένα παράδειγμα αίτησης InsertObervation φαίνεται στη συνέχεια:

```
<InsertObservation xmlns="http://www.opengis.net/sos/1.0"
    xmlns:ows="http://www.opengis.net/ows/1.1"
    xmlns:ogc="http://www.opengis.net/ogc"
    xmlns:om="http://www.opengis.net/om/1.0"
    xmlns:sos="http://www.opengis.net/sos/1.0"
    xmlns:sa="http://www.opengis.net/sampling/1.0"
    xmlns:gml="http://www.opengis.net/gml"
    xmlns:swe="http://www.opengis.net/swe/1.0.1"
    xmlns:xlink="http://www.w3.org/1999/xlink"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
```

```
    xsi:schemaLocation="http://www.opengis.net/sos/1.0
    http://schemas.opengis.net/sos/1.0.0/sosInsert.xsd
    http://www.opengis.net/sampling/1.0
    http://schemas.opengis.net/sampling/1.0.0/sampling.xsd
    http://www.opengis.net/om/1.0

http://schemas.opengis.net/om/1.0.0/extensions/observationSpecialization_override.xsd"
    service="SOS" version="1.0.0">

    <AssignedSensorId>urn:ogc:object:feature:Sensor:IFGI:ifgi-sensor-1</AssignedSensorId>

    <om:Measurement>

        <om:samplingTime>
            <gml:TimeInstant>
                <gml:timePosition>2008-04-01T17:44:15+00</gml:timePosition>
            </gml:TimeInstant>
        </om:samplingTime>

        <om:procedure xlink:href="urn:ogc:object:feature:Sensor:IFGI:ifgi-sensor-1"/>
        <om:observedProperty xlink:href="urn:ogc:def:phenomenon:OGC:1.0.30:waterlevel"/>

        <om:featureOfInterest>
            <sa:SamplingPoint gml:id="foi_1001">
                <gml:name>SamplingPoint 1</gml:name>
                <sa:sampledFeature xlink:href=""/>
                <sa:position>
                  <gml:Point>
                    <gml:pos srsName="urn:ogc:def:crs:EPSG:4326">7.52 52.90</gml:pos>
                  </gml:Point>
                </sa:position>
              </sa:SamplingPoint>
        </om:featureOfInterest>

        <om:result uom="cm">10.0</om:result>
    </om:Measurement>

</InsertObservation>
```

Το άνωθι XML κείμενο δηλώνει τα εξής:

- Η μέτρηση προέρχεται από τη διαδικασία / αισθητήρα με αναγνωριστικό (assigned sensor ID / procedure) urn:ogc:object:feature:Sensor:IFGI:ifgi-sensor-1.
- Η χρονική στιγμή δειγματοληψίας (sampling time) κατά ISO8601 είναι 2008-04-01T17:44:15+00.
- Η παρατηρούμενη ιδιότητα (observed property) είναι η στάθμη ύδατος (water level).
- Το γνώρισμα ενδιαφέροντος (feature of interest) είναι το Sampling Point 1.
- Το αναγνωριστικό (ID) που αντιστοιχίζεται στο γνώρισμα είναι foi_1001.
- Η θέση του γνωρίσματος είναι στο γεωγραφικό μήκος / πλάτος (7.52, 52.9).
- Το αποτέλεσμα (result) της μέτρησης είναι στάθμη ύδατος 10.0 εκαστόμετρων.

Παρατηρούμε ότι το κείμενο της InsertObservation συμμορφώνεται, καθώς οφείλει, με το πρότυπο (template) της παρατήρησης που υπεβλήθη για το δεδομένο αισθητήρα κατά την εγγραφή του με την RegisterSensor.

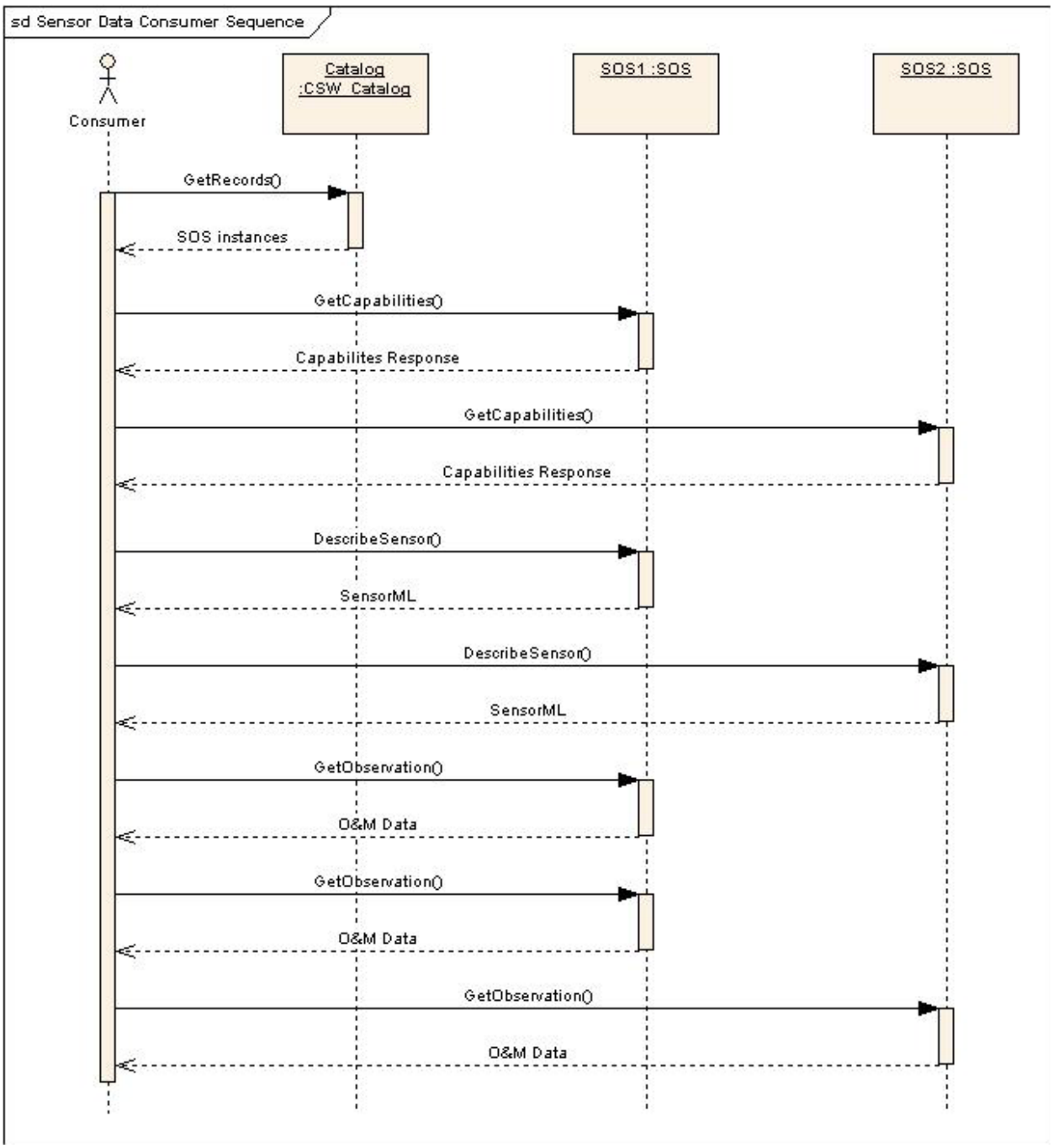
Σε περίπτωση επιτυχούς καταχώρησης της μέτρησης ή παρατήρησης, αναμένεται μια απάντηση InsertObservationResponse) σαν την ακόλουθη:

```
<sos:InsertObservationResponse>
    <sos:AssignedObservationId>o_126</sos:AssignedObservationId>
</sos:InsertObservationResponse>
```

Το SOS έχει συνεπώς αντιστοιχίσει στη μέτρηση το αναγνωριστικό (assigned observation ID) o_126. Η τιμή αυτή μπορεί να χρησιμοποιηθεί σε μια εξειδίκευση της GetObservation, την GetObservationById, η οποία επιστρέφει τη μέτρηση που αντιστοιχεί στο αναγνωριστικό που παρέχεται σαν είσοδος. Η παραλλαγή αυτή δε θα μας απασχολήσει περαιτέρω.

2.4.1.2 Από την πλευρά του πελάτη – καταναλωτή

Η οπτική γωνία του πελάτη ενός SOS (ή περισσότερων) φαίνεται στο παρακάτω σχήμα:



Σχήμα 3: Διάγραμμα ακολουθίας μηνυμάτων μεταξύ πελάτη και SOS

Εφόσον ο πελάτης δε γνωρίζει δια άλλων οδών τα URL των SOS που μπορούν να τον εξυπηρετήσουν μπορεί να υποβάλει μια επερώτηση σε δικτυακό κατάλογο SWE υπηρεσιών για να τα ανακαλύψει.

Ακολουθεί επερώτηση σε συγκεκριμένο SOS για προσδιορισμό των δυνατοτήτων του μέσω μιας αίτησης GetCapabilities. Με αυτόν τον τρόπο ο πελάτης μπορεί να μάθει ποια φαινόμενα είναι διαθέσιμα προς παρατήρηση, πόσοι και ποιοι αισθητήρες είναι

εγγεγραμμένοι στο SOS, καθώς και το τι δυνατότητες φιλτραρίσματος διαθέτει. Τα στοιχεία αυτά περιέχονται στο GetCapabilitiesResponse.

Ανάλογα με τη γενικότητα της υλοποίησης του πελάτη, αυτός μπορεί να έχει υποδομή για πολλά διαφορετικά είδη μετρήσεων. Ειδάλλως μπορεί απλά να διαχωρίσει αν το SOS προφέρει το είδος μετρήσεων που αυτός υποστηρίζει. Αν είναι απαραίτητο, μπορεί να προσπελάσει περισσότερες πληροφορίες για επιμέρους αισθητήρες μέσω μιας αίτησης DescribeSensor, παρέχοντας σαν είσοδο το αναγνωριστικό (ID) κάποιου δεδομένου αισθητήρα. Η συγκεκριμένη αίτηση δε θα μας απασχολήσει ιδιαίτερα στη συνέχεια.

Μετά την αναγκαία συλλογή μεταδεδομένων, ο πελάτης μπορεί να αιτηθεί μετρήσεων μέσω της λειτουργίας GetObservation, προσδιορίζοντας τα κριτήρια που πρέπει να πληρούνται για την ανάκτηση τους από τη βάση του SOS. Τα κριτήρια αυτά μπορεί να είναι χρονικά, χωρικά κτλ. Η χρήση των δεδομένων από την εφαρμογή, είτε περιλαμβάνει επεξεργασία, αποθήκευση είτε παρουσίαση στο χρήστη εξαρτάται από τη συγκεκριμένη περίπτωση. Η απάντηση σε μια αίτηση GetObservation περιέχει μια συλλογή από μηδέν ή περισσότερες μετρήσεις, κωδικοποιημένες στη γλώσσα σήμανσης O&M (Observations & Measurements).

Στη συνέχεια θα αναλύσουμε περισσότερα τα βασικά πεδία των κύριων μηνυμάτων που χρησιμοποιούνται από τον καταναλωτή δεδομένων.

2.4.1.2.1 GetCapabilities

Αν υποτεθεί ότι ο πελάτης δε γνωρίζει εκ των προτέρων για την ύπαρξη ενός στιγμιοτύπου (instance) SOS, η λειτουργία GetCapabilities είναι η πρώτη αίτηση που πρέπει να αποστείλει προς αυτό για την ανακάλυψη των μεταδεδομένων της ίδιας της υπηρεσίας και των αισθητήρων που αντιπροσωπεύει. Ειδάλλως δεν μπορεί να ανακτήσει τις απαραίτητες παραμέτρους για τις μετέπειτα αιτήσεις.

Ένα παράδειγμα αίτησης GetCapabilities είναι το εξής:

```
<?xml version="1.0" encoding="UTF-8"?>
<GetCapabilities xmlns="http://www.opengis.net/sos/1.0"
    xmlns:ows="http://www.opengis.net/ows/1.1"
    xmlns:ogc="http://www.opengis.net/ogc"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://www.opengis.net/sos/1.0
    http://schemas.opengis.net/sos/1.0.0/sosGetCapabilities.xsd"
    service="SOS">

    <ows:AcceptVersions>
        <ows:Version>1.0.0</ows:Version>
    </ows:AcceptVersions>

    <ows:Sections>
        <ows:Section>OperationsMetadata</ows:Section>
        <ows:Section>ServiceIdentification</ows:Section>
```

```
        <ows:Section>ServiceProvider</ows:Section>
        <ows:Section>Filter_Capabilities</ows:Section>
        <ows:Section>Contents</ows:Section>
    </ows:Sections>

</GetCapabilities>
```

Εν ολίγοις ο πελάτης ζητά από το SOS να του παρέχει τα μεταδεδομένα του για έναν αριθμό προσδιορισθέντων τμημάτων (sections), όπως τα μεταδεδομένα των ενεργειών / αιτήσεων που υποστηρίζει (operations metadata), πληροφορίες για την ταυτοποίηση της υπηρεσίας (service identification) και του παρόχου (service provider), τις δυνατότητες φιλτραρίσματος αποτελεσμάτων (filter capabilities) που διαθέτει και τα είδη των παρατηρήσεων που προσφέρονται.

Χάριν συντομίας δε θα παραθέσουμε την απάντηση σε μια τέτοια αίτηση, μιας και είναι αρκετά μακροσκελής. Μπορούμε όμως να πούμε ότι οι δυνατότητες του SOS - κατά τμήματα - περιλαμβάνουν:

- Ταυτοποίηση υπηρεσίας
    - Τίτλος
    - Έκδοση
- Διαχειριστής του SOS
    - Όνομα
    - Δικτυακό τόπο
    - Στοιχεία επικοινωνίας
- Μεταδεδομένα ενεργειών
    - Ονόματα υποστηριζόμενων αιτήσεων
    - Επιτρεπτές παράμετροι αιτήσεων
- Δυνατότητες Φιλτραρίσματος
    - Χωρικοί τελεστές (BBOX, Contains, Intersects, Overlaps)
    - Χρονικοί τελεστές (During, Equals, After, Before)
    - Συγκριτικοί τελεστές αποτελέσματος (Between, EqualTo, NotEqualTo, LessThan, LessThanEqualTo, GreaterThan, GreaterThanEqualTo, Like)
- Περιεχόμενα
    - Λίστα προσφερόμενων ειδών παρατηρήσεων (observation offering list)
        - Φαινόμενα προς παρατήρηση
        - Παρατηρούμενες ιδιότητες
        - Γνωρίσματα προς παρατήρηση
        - Αναγνωριστικά εγγεγραμμένων διαδικασιών / αισθητήρων

2.4.1.2.2 GetObservation

Με την επερώτηση GetObservation ο πελάτης μπορεί να ανακτήσει από τη βάση δεδομένων του SOS μετρήσεις που πληρούν κάποια δεδομένα κριτήρια:

- Προσφοράς (offering)
- Γνωρίσματος ενδιαφέροντος (feature of interest)
- Συγκεκριμένου αισθητήρα / διαδικασίας (procedure)
- Γεωγραφικής περιοχής βάσει συντεταγμένων
- Χρονικής περιόδου (time period) ή χρονικής στιγμής (time instant) λήψης
- Φίλτρου αποτελεσμάτων (μεγαλύτερο / μικρότερο από Χ κτλ.)

Ένα παράδειγμα XML εγγράφου που περιέχει μια GetObservation επερώτηση είναι:

```
<?xml version="1.0" encoding="UTF-8"?>
<GetObservation xmlns="http://www.opengis.net/sos/1.0"
    xmlns:ows="http://www.opengis.net/ows/1.1"
    xmlns:gml="http://www.opengis.net/gml"
    xmlns:ogc="http://www.opengis.net/ogc"
    xmlns:om="http://www.opengis.net/om/1.0"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://www.opengis.net/sos/1.0
    http://schemas.opengis.net/sos/1.0.0/sosGetObservation.xsd"
    service="SOS" version="1.0.0" srsName="urn:ogc:def:crs:EPSG:4326">

    <offering>GAUGE_HEIGHT</offering>

    <eventTime>
        <ogc:TM_During>
            <ogc:PropertyName>urn:ogc:data:time:iso8601</ogc:PropertyName>
            <gml:TimePeriod>
                <gml:beginPosition>2008-03-01T17:44:15+00</gml:beginPosition>
                <gml:endPosition>2008-05-01T17:44:15+00</gml:endPosition>
            </gml:TimePeriod>
        </ogc:TM_During>
    </eventTime>

          <observedProperty>urn:ogc:def:phenomenon:OGC:1.0.30:waterlevel
      </observedProperty>
    <responseFormat>text/xml;subtype=&amp;quot;om/1.0.0&amp;quot;</responseFormat>

</GetObservation>
```

Όπως φαίνεται στο παράδειγμα, ζητείται η ανάκτηση των μετρήσεων που περιλαμβάνονται στην προσφορά (offering) ΥΨΟΣ_ΜΕΤΡΗΤΗ (GAUGE_HEIGHT), για την παρατηρούμενη ιδιότητα «στάθμη ύδατος» (water level), για τη χρονική περίοδο 2008-03-01T17:44:15+00 έως 2008-05-01T17:44:15+00.

Εφόσον υπάρχουν μία ή περισσότερες μετρήσεις ή παρατηρήσεις που ανταποκρίνονται στα κριτήρια της επερώτησης, αναμένεται μια απάντηση σαν την εξής:

```
<?xml version="1.0" encoding="UTF-8"?>
<om:ObservationCollection xmlns:om="http://www.opengis.net/om/1.0"
xmlns:gml="http://www.opengis.net/gml"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xlink="http://www.w3.org/1999/xlink"
xmlns:swe="http://www.opengis.net/swe/1.0.1"
xmlns:sa="http://www.opengis.net/sampling/1.0" gml:id="oc_0"
xsi:schemaLocation="http://www.opengis.net/om/1.0
http://schemas.opengis.net/om/1.0.0/om.xsd
http://www.opengis.net/sampling/1.0
http://schemas.opengis.net/sampling/1.0.0/sampling.xsd">
  <gml:boundedBy>
    <gml:Envelope srsName="urn:ogc:def:crs:EPSG:4326">
      <gml:lowerCorner>7.52 52.9</gml:lowerCorner>
      <gml:upperCorner>7.52 52.9</gml:upperCorner>
    </gml:Envelope>
  </gml:boundedBy>
  <om:member>

    <om:Observation gml:id="ot_126">
      <om:samplingTime>
        <gml:TimePeriod xsi:type="gml:TimePeriodType">
          <gml:beginPosition>2008-04-
01T20:44:15.000+03:00</gml:beginPosition>
          <gml:endPosition>2008-04-01T20:44:15.000+03:00</gml:endPosition>
        </gml:TimePeriod>
      </om:samplingTime>
      <om:procedure xlink:href="urn:ogc:object:feature:Sensor:IFGI:ifgi-
sensor-1"/>

      <om:observedProperty>
        <swe:CompositePhenomenon gml:id="cpid0" dimension="1">
          <gml:name>resultComponents</gml:name>
          <swe:component xlink:href="urn:ogc:data:time:iso8601"/>
          <swe:component
xlink:href="urn:ogc:def:phenomenon:OGC:1.0.30:waterlevel"/>
        </swe:CompositePhenomenon>
```

```
      </om:observedProperty>
      <om:featureOfInterest>

        <gml:FeatureCollection>
          <gml:featureMember>
            <sa:SamplingPoint gml:id="foi_1001">
              <gml:name>SamplingPoint 1</gml:name>
              <sa:position>
                <gml:Point>
                  <gml:pos srsName="urn:ogc:def:crs:EPSG:4326">7.52
52.9</gml:pos>
                </gml:Point>

              </sa:position>
            </sa:SamplingPoint>
          </gml:featureMember>
        </gml:FeatureCollection>
      </om:featureOfInterest>
      <om:result>
        <swe:DataArray>
          <swe:elementCount>
            <swe:Count>

              <swe:value>1</swe:value>
            </swe:Count>
          </swe:elementCount>
          <swe:elementType name="Components">
            <swe:SimpleDataRecord>
              <swe:field name="Time">
                <swe:Time definition="urn:ogc:data:time:iso8601"/>
              </swe:field>

              <swe:field name="feature">
                <swe:Text definition="urn:ogc:data:feature"/>
              </swe:field>
              <swe:field name="waterlevel">
                <swe:Quantity
definition="urn:ogc:def:phenomenon:OGC:1.0.30:waterlevel">
                  <swe:uom code="cm"/>
                </swe:Quantity>
              </swe:field>
            </swe:SimpleDataRecord>
```

```
          </swe:elementType>
          <swe:encoding>
            <swe:TextBlock decimalSeparator="." tokenSeparator="," 
blockSeparator=";"/>
          </swe:encoding>
          <swe:values>2008-04-
01T20:44:15.000+03:00,foi_1001,10.0;</swe:values>
        </swe:DataArray>
      </om:result>
    </om:Observation>


  </om:member>
</om:ObservationCollection>
```

Η απάντηση περιέχει ένα στοιχείο (element) του τύπου ObservationCollection, δηλαδή μιας συλλογής παρατηρήσεων κωδικοποιημένων σε O&M, η οποία στο παράδειγμά μας περιέχει μία ακριβώς μέτρηση, με χρόνο δειγματοληψίας 2008-04-01T20:44:15.000+03:00 και αποτέλεσμα μέτρησης 10 εκατοστά στάθμη ύδατος στην περιοχή του γνωρίσματος SamplingPoint 1.

Αν δε βρεθούν μετρήσεις που να πληρούν τα κριτήρια της επερώτησης, επιστρέφεται κενό (nil) σύνολο αποτελεσμάτων:

```
<om:ObservationCollection xmlns:om="http://www.opengis.net/om/1.0"
xmlns:gml="http://www.opengis.net/gml"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xlink="http://www.w3.org/1999/xlink" gml:id="oc_0"
xsi:schemaLocation="http://www.opengis.net/om/1.0
http://schemas.opengis.net/om/1.0.0/om.xsd
http://www.opengis.net/sampling/1.0
http://schemas.opengis.net/sampling/1.0.0/sampling.xsd">
  <om:member xlink:href="urn:ogc:def:nil:OGC:inapplicable"/>
</om:ObservationCollection>
```

2.4.1.3 Λοιπές λειτουργίες και σχόλια επί των λειτουργιών του SOS

Παρέχονται επίσης από τις προδιαγραφές του SOS οι ακόλουθες ενέργειες, τις οποίες τις αναφέρουμε χωρίς παραιτέρω εμβάθυνση:

- **DescribeSensor** (υποχρεωτική): Περιγραφή ενός αισθητήρα μέσα από τα μεταδεδομένα του, δεδομένου του αναγνωριστικού του (ID / URI).
- **GetObservationById** (προαιρετική): Ανάκτηση συγκεκριμένης παρατήρησης με βάση το αναγνωριστικό της (ID).
- **GetResult** (προαιρετική): Επιτρέπει την τακτική ανάκτηση δεδομένων από συγκεκριμένους αισθητήρες, χωρίς να απαιτεί επαναλαμβανόμενες πανομοιότυπες επερωτήσεις.

- **GetFeatureIfInterest** (προαιρετική): Ανάκτηση μεταδεδομένων για συγκεκριμένο γνώρισμα ενδιαφέροντος.
- **GetFeatureOfInterestTime** (προαιρετική): Ανάκτηση της χρονικής περιόδου για την οποία υπάρχουν αποτελέσματα μετρήσεων για συγκεκριμένο γεωγραφικό γνώρισμα.
- **DescribeFeatureType** (προαιρετική): Ανάκτηση του XML σχήματος (schema) που προσδιορίζει ένα είδος γνωρίσματος ενδιαφέροντος.
- **DescribeObservationType** (προαιρετική): Ανάκτηση του XML σχήματος (schema) που περιγράφει έναν τύπο παρατήρησης.
- **DescribeResultModel** (προαιρετική): Ανάκτηση του XML σχήματος (schema) με το οποίο συμμορφώνονται τα αποτελέσματα (results) των παρατηρήσεων, π.χ. βαθμωτού τύπου ή μη, μονάδα μέτρησης κτλ.

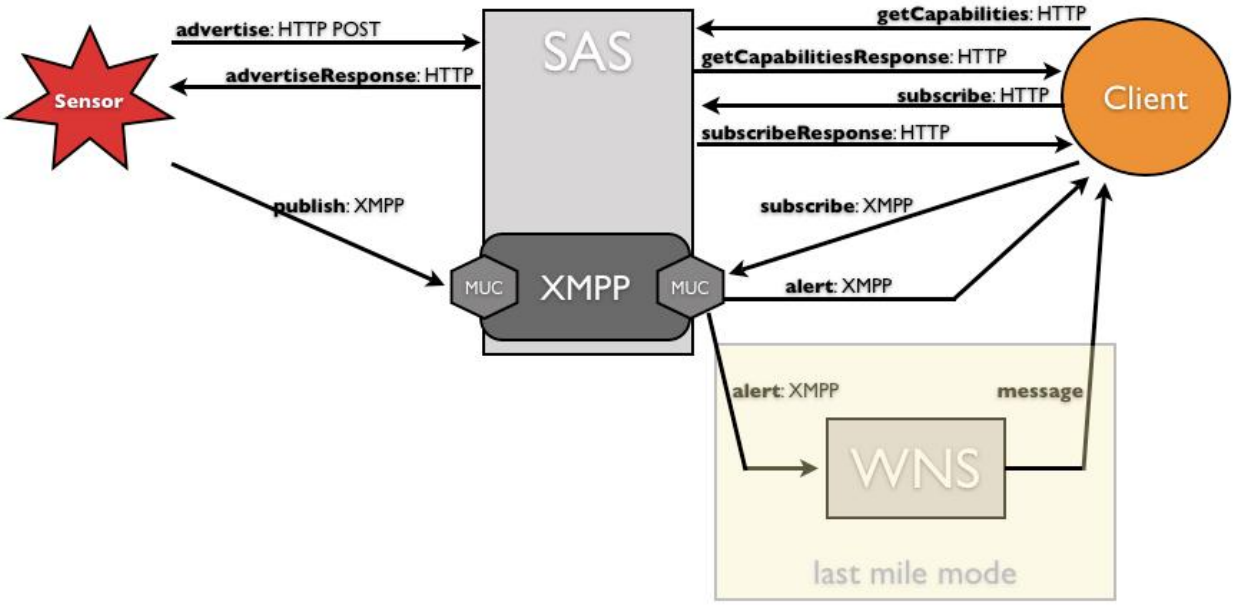
Από τις άνωθι λειτουργίες, οι προαιρετικές και πιο εξειδικευμένες αποτελούν το Enhanced Profile του SOS. Να σημειωθεί πως το πρωτόκολλο παρουσιάζει κάποιες ελλείψεις, όπως π.χ. η ανυπαρξία κάποιας αίτησης για αναίρεση της εγγραφής ενός αισθητήρα (DeRegisterSensor). Αυτό μπορεί να οδηγήσει σε απόρριψη αιτήσεων εγγραφής από αισθητήρες που είχαν καταχωρηθεί παλιότερα και που επανέρχονται κάποια στιγμή σε ενεργή κατάσταση. Συνεπώς το λογισμικό του παραγωγού πρέπει να λαμβάνει υπόψη αυτό το πρόβλημα και να συγκρατεί το ίδιο ποιοι αισθητήρες είναι εγγεγραμμένοι ή να θεωρεί την απόρριψη διπλότυπης εγγραφής σαν καταφατική απάντηση και να προχωράει στη συλλογή δεδομένων.

Μια δευτερεύουσα έλλειψη είναι η αδυναμία αφαίρεσης παρατηρήσεων που έχουν καταχωρηθεί, αν και αυτό μπορεί εν τέλει να μην κρίνεται και απαραίτητο ή επιθυμητό γιατί μπορεί να είναι σκόπιμη η διατήρηση όλων των μετρήσεων για λόγους αρχειοθέτησης (archiving), καθώς επίσης και για την αποφυγή κακόβουλης διαγραφής δεδομένων.

2.4.2 Data push – SAS

Η υλοποίηση Push-based υπηρεσιών κατά τη σύσταση του SAS είναι πιο περίπλοκη από το απλό Pull-based σχήμα που εξετάσαμε προηγουμένως. Πέρα από τον παραγωγό, τον πελάτη, το SAS και τη βάση δεδομένων του, απαιτεί και τη συμμετοχή ενός διακομιστή πρωτοκόλλου XMPP για αποστολή ειδοποιήσεων σε πραγματικό χρόνο μέσω άμεσης μηνυματοδοσίας (IM - Instant Messaging). Αρχικά το σκεπτικό του OGC ως προς ποιο πρωτόκολλο θα χρησιμοποιείται για την ειδοποίηση του πελάτη προσανατολιζόταν προς ανεξαρτησία από συγκεκριμένα πρωτόκολλα. Εν τέλει δόθηκε προτίμηση στο XMPP, παλιότερα γνωστό ως Jabber και ευρέως διαδεδομένο στο χώρο.

Το παρακάτω σχήμα συνοψίζει την αλληλεπίδραση του παραγωγού και καταναλωτή δεδομένων με την υπηρεσία SAS:



Σχήμα 4: Αναπαράσταση της αλληλεπίδρασης του παραγωγού και καταναλωτή δεδομένων με το SAS

Η ακολουθία των μηνυμάτων ξεκινάει με μία ή περισσότερες διαφημίσεις από τον παραγωγό, εμφανιζόμενο στο σχήμα ως Sensor, για το ποια φαινόμενα μπορεί να παρατηρήσει και στην περιοχή ποιων γεωγραφικών γνωρισμάτων, με το σκοπό την αποστολή μηνυμάτων συναγερμού όταν ισχύσει μια δεδομένη συνθήκη. Η αίτηση Advertise περιέχει επίσης πληροφορίες για τη συχνότητα με την οποία μπορεί ο παραγωγός δεδομένων να ενημερώνει, τη μονάδα μέτρησης για τις μετρούμενες ποσότητες, και το χρονικό όριο μέχρι το οποίο αυτή η ροή πληροφορίας θα είναι διαθέσιμη. Η διαφήμιση αυτή καλείται και δημοσίευση (publication).

Εφόσον η αίτηση είναι έγκυρη, το SAS θα απαντήσει με ένα AdvertiseResponse που περιέχει το αναγνωριστικό της δημοσίευσης (publication ID) και το XMPP URI της ομαδικής διάσκεψης (conference) που δημιούργησε το SAS για τη συγκεκριμένη διαφήμιση. Ο παραγωγός εισέρχεται στο group chat / conference και δημοσιεύει (publish) εκεί τις μετρήσεις σε XML μορφή εντός άμεσων μηνυμάτων (Instant Messages). Το SAS, σα δημιουργός, ιδιοκτήτης και μέλος του group chat, έχει την εποπτεία των μηνυμάτων που εισέρχονται.

Για να αποσταλεί μήνυμα συναγερμού πρέπει ο πελάτης να εκδηλώσει ενδιαφέρον για κάποια από τις παρατηρούμενες ιδιότητες που διαφημίζονται. Σε πρώτο στάδιο, μέσω μιας επερώτησης GetCapabilities και της αντίστοιχης απάντησης GetCapabilitiesResponse, ο πελάτης πληροφορείται ποιες δημοσιεύσεις ισχύουν. Κατόπιν, μέσω αίτησης Subscribe προς το SAS, καταχωρεί μια συνδρομή για μια συγκεκριμένη παρατηρούμενη ιδιότητα και γνώρισμα ενδιαφέροντος, προσδιορίζοντας και το κριτήριο συναγερμού, για παράδειγμα αποτέλεσμα μέτρησης πάνω ή κάτω από κάποιο όριο. Μπορεί επίσης να ορίσει ένα XMPP group chat room (MUC – Multi User Chat) στο οποίο επιθυμεί να λάβει τις ειδοποιήσεις κατάστασης συναγερμού. Αν δεν το ορίσει ρητά, η απάντηση SubscriptionResponse θα περιέχει το conference που δημιούργησε το SAS για την περίπτωση αυτή. Στο XMPP URI αυτό πρέπει να συνδεθεί ο πελάτης στη συνέχεια για να μπορεί να δεχτεί τα σχετικά μηνύματα. Το SAS συνδέεται κι αυτό στην ίδια σύνοδο.

Παρατηρούμε ότι ο μηχανισμός publish-subsribe-alert περιλαμβάνει τη δημιουργία δύο συνόδων XMPP πολλαπλών συμμετεχόντων. Η λογική της χρησιμοποίησης τους είναι η εξής: αν μια μέτρηση που δημοσιευθεί στο πρώτο conference (producer-SAS) βρεθεί από το SAS να πληροί μια συνθήκη συναγερμού όπως αυτή έχει οριστεί στη συνδρομή του πελάτη, τότε αποστέλλει στο δεύτερο conference (SAS-client) μήνυμα XML τύπου SASAlert. Το μήνυμα αυτόματα αναπαράγεται από τον XMPP Server προς τον client που είναι ο έτερος συμμετέχων και με αυτόν τον τρόπο ολοκληρώνεται η διαδικασία ειδοποίησης. Το λογισμικό του πελάτη στη συνέχεια μπορεί να επεξεργαστεί το μήνυμα συναγερμού με όποιον τρόπο απαιτείται από την εφαρμογή.

Ο παραγωγός μπορεί να ακυρώσει τη δημοσίευση με μια αίτηση CancelAdvertisement, ενώ ο πελάτης μπορεί να ακυρώσει μια συνδρομή αποστέλλοντας στο SAS την αίτηση CancelSubscription. Οι αντίστοιχες σύνοδοι σε XMPP conferences καταστρέφονται κατά την ακύρωση, και οι συμμετέχοντες αποσυνδέονται από τα chat rooms.

Στην εκτεταμένη του μορφή το παραπάνω σύστημα μπορεί να περιλαμβάνει και την ασύγχρονη ειδοποίηση του πελάτη μέσω email, SMS, fax ή άλλων IM clients, με χρήση της υπηρεσία ειδοποιήσεων ιστού (WNS - Web Notification Service). Σε αυτήν την περίπτωση στην αίτηση Subscribe πρέπει να ορίσει σαν αποδέκτη των μηνυμάτων συναγερμού το WNS.

2.4.2.1 Από την πλευρά του παραγωγού

Οι ενέργειες που αφορούν στον παραγωγό είναι οι αιτήσεις Advertise και CancelAdvertisement προς το SAS και η δημοσίευση (publication) μετρήσεων στον XMPP Server.

2.4.2.1.1 Advertise

Όπως εξηγήσαμε και παραπάνω, με την αίτηση Advertise ο παραγωγός δεδομένων γνωστοποιεί στο SAS τις παρατηρούμενες ιδιότητες για τις οποίες υποστηρίζεται ο μηχανισμός ειδοποίησης. Τα βασικά πεδία που περιέχει που περιέχει το XML κείμενο μιας τέτοιας αίτησης φαίνονται στο επόμενο παράδειγμα:

```
<?xml version="1.0" encoding="UTF-8"?>
<Advertise xmlns="http://www.opengis.net/sas"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
```

```
xsi:schemaLocation="http://www.opengis.net/sas ../sasAll.xsd"
xmlns:swe="http://www.opengis.net/swe" service="SAS" version="1.0.0">
   <FeatureOfInterest>
      <Name>Muenster</Name>
      <Description>
The humidity is measured by a sensor which is mounted at a weather station
located on the roof of the IFGI in Muenster</Description>
   </FeatureOfInterest>
   <OperationArea>
      <swe:GeoLocation>
         <swe:longitude>
            <swe:Quantity>51.96</swe:Quantity>
         </swe:longitude>
         <swe:latitude>
            <swe:Quantity>7.607</swe:Quantity>
         </swe:latitude>
         <swe:altitude>
            <swe:Quantity>77.8</swe:Quantity>
         </swe:altitude>
      </swe:GeoLocation>
   </OperationArea>
   <AlertMessageStructure>
      <QuantityProperty>
         <Content definition="urn:ogc:humidity" uom="percent" min="0"
max="99"/>
      </QuantityProperty>
   </AlertMessageStructure>
   <AlertFrequency>2</AlertFrequency>
   <DesiredPublicationExpiration>2008-04-
01T00:00:00Z</DesiredPublicationExpiration>
</Advertise>
```

Εν ολίγοις διαφημίζονται μετρήσεις υγρασίας (humidity) σε ποσοστιαίες μονάδες (%) από έναν αισθητήρα βρισκόμενο στη θέση (51.96, 7.607) και σε υψόμετρο 77.8 μέτρων, στην περιοχή του Muenster, που αποτελεί το γνώρισμα ενδιαφέροντος (feature of interest) για την περίπτωση αυτή. Η επιθυμητή, από την πλευρά του παραγωγού, χρονική στιγμή λήξης αυτής της δημοσίευσης είναι 2008-04-01-T00:00:00Z και η συχνότητα αποστολής νέων μετρήσεων από τον αισθητήρα (Alert Frequency) είναι 2 Hz, δηλαδή 2 φορές το δευτερόλεπτο. Να σημειωθεί ότι ο παραγωγός δεν έχει περιορισμό στον αριθμό διαφημίσεων που μπορεί να υποβάλει.

Το SAS αφού καταχωρήσει αυτά τα δεδομένα στη βάση του και δημιουργήσει στον XMPP Server ένα νέο conference, απαντά στην αίτηση Advertise με το AdvertiseResponse:

```
<?xml version="1.0" encoding="UTF-8"?>
<AdvertiseResponse xmlns="http://www.opengis.net/sas"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.opengis.net/sas ../sasAll.xsd" expires="2007-
01-01T00:00:00Z">
   <PublicationID>Pub301</PublicationID>
   <XMPPURI>xmpp://52North.org/sas/publications/c301</XMPPURI>
</AdvertiseResponse>
```

Το SAS έχει αντιστοιχίσει στη δημοσίευση του παραγωγού το αναγνωριστικό (publication ID) Pub301. Η απάντηση επίσης περιέχει το πλήρες URI του conference στο οποίο πρέπει ο παραγωγός να συνδεθεί για να δημοσιεύσει τις μετρήσεις. Το SAS έχει ήδη συνδεθεί στη σύνοδο και παρακολουθεί τη ροή των μηνυμάτων προς αυτήν.

2.4.2.1.2 CancelAdvertisement

Με την αίτηση CancelAdvertisement ο παραγωγός μπορεί να αναιρέσει μια ισχύουσα διαφήμιση παρέχοντας στο SAS το αναγνωριστικό της δημοσίευσης που πρέπει να ακυρωθεί, όπως αυτό είχε αρχικά αντιστοιχιστεί από το SAS στο AdvertisementResponse:

```
<?xml version="1.0" encoding="UTF-8"?>
<CancelAdvertisement xmlns="http://www.opengis.net/sas"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.opengis.net/sas ../sasAll.xsd" service="SAS"
version="1.0.0">
   <PublicationID>Pub301</PublicationID>
</CancelAdvertisement>
```

Η απάντηση CancelAdvertisementResponse περιέχει την ένδειξη επιτυχούς ή μη ολοκλήρωσης της ακύρωσης:

```
<?xml version="1.0" encoding="UTF-8"?>
<CancelAdvertisementResponse xmlns="http://www.opengis.net/sas"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.opengis.net/sas ../sasAll.xsd">
   <PublicationID>Pub301</PublicationID>
   <CancellationStatus>confirmed</CancellationStatus>
</CancelAdvertisementResponse>
```

2.4.2.1.3 Publish

Πρέπει να σημειώσουμε ότι από τη σύσταση του SAS μπορεί να προκύψει σύγχυση για την ορολογία με την οποία πρέπει να αναφερόμαστε στα δεδομένα που δημοσιεύονται στον XMPP. Ο όρος που χρησιμοποείται από τη σύσταση για όλα ανεξαιρέτως τα

μηνύματα, ανεξάρτητα αν αυτά τελικά θα προκαλέσουν ένα μήνυμα συναγερμού προς τον πελάτη ή όχι, είναι “alert”, ο οποίος μεταφράζεται σαν συναγερμός / ειδοποίηση. Ως συναγερμός όμως θεωρείται και το μήνυμα που αποστέλλεται στον πελάτη αν ισχύσει συνθήκη συναγερμού, επίσης ως ειδοποίηση (notification) θεωρείται η γνωστοποίηση κατάστασης συναγερμού προς ένα WNS. Στην παρούσα εργασία θα αποκλίνουμε από τη σύσταση και θα αναφερόμαστε με τους όρους:

- Μήνυμα δημοσίευσης δεδομένων / μέτρησης: Το XML μήνυμα με το οποίο κοινοποιεί ο παραγωγός μια μέτρηση στο conference. Θα αποφύγουμε τον όρο alert μιας και από τη σκοπιά του παραγωγού αυτή η διαδικασία δεν έχει στην ουσία κάποια διαφορά από την αποστολή μετρήσεων όπως την εξετάσαμε στην αίτηση InsertObservation του SOS, με τη διαφορά ότι εδώ οι μετρήσεις δεν αποθηκεύονται παρά μόνο συγκρίνονται με κριτήρια συναγερμού, εφόσον έχουν τεθεί από τον πελάτη.
- Συναγερμός / Ειδοποίηση / Ειδοποίηση συναγερμού: Το XML μήνυμα που αποστέλλει το SAS στο conference μεταξύ του ιδίου και του πελάτη, όταν ληφθεί στο πρώτο conference (μεταξύ SAS και παραγωγού) μέτρηση που να ικανοποιεί μια συνθήκη συναγερμού.
- Μιας και δε θα δώσουμε βάση στο WNS, δε θα μας απασχολήσουν οι ειδοποιήσεις προς αυτό.

Η δημοσίευση των μετρήσεων όπως είπαμε πραγματοποιείται στο conference που έχει δημιουργήσει το SAS στον XMPP Server κατόπιν της αίτησης Advertise. Τα μηνύματα στο XMPP πρωτόκολλο είναι σε μορφή XML εγγράφων. Κατά τη σύσταση του OGC για το SAS ο πελάτης πρέπει να αποστέλλει τα δεδομένα του στο conference στην ακόλουθη μορφή:

```
<Latitude Longitude Altitude TimeOfAlert AlertExpires Value>
```

Δηλαδή:

```
<Γεωγραφικό-Πλάτος Γεωγραφικό-Μήκος Υψόμετρο Χρονική-Στιγμή-Ειδοποίησης Χρονική-Στιγμή-Λήξης-Ειδοποίησης Τιμή>
```

Τα πεδία (fields) βλέπουμε ότι δεν περιέχονται σε ξεχωριστά XML elements με τα δικά τους tags, απλά σε προσυμφωνημένη σειρά των με διαχωριστικό (delimiter) το κενό. Ένα δείγμα τέτοιας δημοσίευσης για τον παράδειγμα που είδαμε παραπάνω είναι:

```
<51.96 7.607 77.8 2008-03-25T10-00-00+2 0 60.8>
```

Αντιστοιχεί σε μέτρηση για την ακριβώς ίδια θέση για την οποία έγινε η διαφήμιση, με χρονική στιγμή αποστολής την 2008-03-25T10-00-00+2 και τιμή υγρασίας 60.8%.

2.4.2.2 Από την πλευρά του πελάτη – καταναλωτή

Οι λειτουργίες που αφορούν στον πελάτη είναι οι αιτήσεις GetCapabilities, Subscribe, CancelSubscription και η λήψη μηνυμάτων SASAlert από το SAS μέσω του XMPP Server.

2.4.2.2.1 GetCapabilities

Όπως είδαμε και στην αντίστοιχη αίτηση για το SOS, η GetCapabilities χρησιμεύει κυρίως για να ενημερωθεί η εφαρμογή του πελάτη για τα μεταδεδομένα της υπηρεσίας SAS. Με αυτόν τον τρόπο μπορεί να κρίνει αν υπάρχει κάποια ισχύουσα διαφήμιση μετρήσεων βάσει των οποίων μπορούν να εκδοθούν ενδιαφέρουσες ειδοποιήσεις κατάστασης συναγερμού, χωρίς ο ίδιος ο πελάτης να απαιτείται να ανακτά συνέχεια (polling) μετρήσεις για να πραγματοποιεί τη σύγκριση ο ίδιος.

Ένα απλό δείγμα αίτησης GetCapabilities που ζητά το τμήμα των κυρίως περιεχομένων (Contents) των μεταδεδομένων του SAS είναι:

```
<?xml version="1.0" encoding="UTF-8"?>
<GetCapabilities xmlns="http://www.opengis.net/sas"
xmlns:ows="http://www.opengeospatial.net/ows"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.opengis.net/sas ../sasAll.xsd" service="SAS">
   <ows:Sections>
      <ows:Section>Contents</ows:Section>
   </ows:Sections>
</GetCapabilities>
```

Λόγω της έκτασης του XML κειμένου της απάντησης, δε θα την παραθέσουμε εδώ. Αναφέρουμε όμως τα βασικά πεδία της απάντησης που έχουν ενδιαφέρον για τον πελάτη:

- Ένδειξη AcceptAdvertisements, με τιμή true ή false, που δηλώνει αν το SAS δέχεται αιτήσεις Advertise ή όχι. Η συγκεκριμένη παράμετρος έχει ενδιαφέρον για τον παραγωγό, σε περίπτωση που αυτός κάνει την αίτηση GetCapabilities για να βεβαιωθεί ότι οι μετέπειτα διαφημίσεις θα γίνουν δεκτές.
- Λίστα με τις προσφερόμενες δημοσιεύσεις διαθέσιμες για συνδρομητικές αιτήσεις (subscription offering list). Κάθε μέλος της λίστας περιέχει:
  - Αναγνωριστικό της προσφερόμενης συνδρομής (subscription offering ID).
  - Περιγραφή της παρατηρούμενης ιδιότητας (alert message structure, quantity / category property).
  - Γεωγραφικό γνώρισμα ενδιαφέροντος (feature of interest).
  - Γεωγραφική περιοχή / θέση (operation area).

Με την εξαίρεση του αναγνωριστικού, τα υπόλοιπα στοιχεία είναι ακριβώς όπως τα έχει υποβάλει ο αντίστοιχος παραγωγός σε Advertise αίτηση.

2.4.2.2.2 Subscribe

Έχοντας προμηθευτεί τη λίστα των διαφημίσεων, ή αλλιώς δημοσιεύσεων για τις οποίες μπορεί να εκδηλώσει ενδιαφέρον, ο πελάτης μπορεί να στείλει αίτηση συνδρομής για μία συγκεκριμένη, καθορίζοντας και το κριτήριο φιλτραρίσματος των μετρήσεων που υποβάλλει ο πάραγωγός, έτσι ώστε να αποστέλλεται στον πελάτη μήνυμα συναγερμού όποτε πληρείται η προσδιορισθείσα συνθήκη.

Μια τυπική αίτηση συνδρομής φαίνεται στη συνέχεια:

```
<?xml version="1.0" encoding="UTF-8"?>
<Subscribe xmlns="http://www.opengis.net/sas"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.opengis.net/sas ../sasAll.xsd" service="SAS"
version="1.0.0">
   <ResultFilter ObservedPropertyDefinition="urn:ogc:humidity" uom="percent">
      <isGreaterThan>80.0</isGreaterThan>
   </ResultFilter>
   <FeatureOfInterestName>Muenster</FeatureOfInterestName>
   <ResultRecipient>
      <XMPPURI>xmpp://foo.bar.org/52NorthSasClient/c3</XMPPURI>
   </ResultRecipient>
</Subscribe>
```

Εν προκειμένω, ο πελάτης ζητά να ειδοποιηθεί όταν το ποσοστό υγρασίας υπερβεί το 80%, και καθορίζει ρητά το URI ενός XMPP Server conference στο οποίο επιθυμεί να ενημερώνεται. Αν παραλείψει αυτήν την πληροφορία, το SAS θα δημιουργήσει το ίδιο ένα conference για το σκοπό αυτό. Να σημειώσουμε ότι ο πελάτης μπορεί να έχει πολλαπλές ενεργές συνδρομές, για διαφορετικά είδη μετρήσεων ή για διαφορετικά κριτήρια επί του ιδίου είδους.

Το SAS απαντά με ένα SubscribeResponse που περιέχει το αναγνωριστικό της συνδρομής (subscription ID), προκειμένου να μπορεί να ζητήσει αργότερα ο πελάτης την τυχόν ακύρωση της. Επίσης περιλαμβάνει τη χρονική στιγμή λήξης της συνδρομής (“expires”) και την επιβεβαίωση επιτυχούς καταχώρησης της συνδρομής (status):

```
<?xml version="1.0" encoding="UTF-8"?>
<SubscribeResponse xmlns="http://www.opengis.net/sas"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.opengis.net/sas ../sasAll.xsd"
SubscriptionID="Sub102" expires="2009-01-01T00:00:00Z">
   <Status>OK</Status>
</SubscribeResponse>
```

2.4.2.2.3 CancelSubscription

Ο πελάτης μπορεί να ακυρώσει μια συνδρομή αποστέλλοντας στο SAS το αναγνωριστικό της συνδρομής (subscription ID), όπως αυτό του δόθηκε μέσα στο SubscibeResponse:

```
<?xml version="1.0" encoding="UTF-8"?>
<CancelSubscription xmlns="http://www.opengis.net/sas"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.opengis.net/sas ../sasAll.xsd" service="SAS"
version="1.0.0">
   <SubscriptionID>Sub102</SubscriptionID>
</CancelSubscription>
```

Το SAS επιβεβαιώνει την επιτυχή ακύρωση της συνδρομής με ένα CancelSubscriptionResponse:

```
<?xml version="1.0" encoding="UTF-8"?>
<CancelSubscriptionResponse xmlns="http://www.opengis.net/sas"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.opengis.net/sas ../sasAll.xsd">
   <SubscriptionID>Sub102</SubscriptionID>
   <Status>OK</Status>
</CancelSubscriptionResponse>
```

Η ακύρωση της συνδρομής συνεπάγεται και την καταστροφή του αντίστοιχου XMPP conference για την επικονωνία μεταξύ SAS και πελάτη.

2.4.2.2.4 SASAlert

Όταν το SAS λάβει μέσω του XMPP Server δημοσιευμένη μέτρηση που ικανοποιεί μια συνθήκη προσδιορισμένη στις ισχύουσες συνδρομές πελατών, τότε σχηματίζει ένα μήνυμα συναγερμού SASAlert με τη μέτρηση που πυροδότησε το συναγερμό και τη στέλνει στο XMPP URI της αντίστοιχης συνδρομής.

Για το παράδειγμα του αισθητήρα υγρασίας που εξετάζουμε στην τρέχουσα ενότητα, ένα SASAlert θα μπορούσε να είναι:

```
<?xml version="1.0" encoding="UTF-8"?>
<SASAlert xmlns="http://www.opengis.net/sas"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.opengis.net/sas ../sasAll.xsd">
   <Header>
      <AlertMessageStructure>
         < QuantityProperty>
            <Content definition="urn:ogc:humidity" uom="percent" min="0"
max="99"/>
         </QuantityProperty>
```

```
      </AlertMessageStructure>
   </Header>
   <Body>51.96 7.607 77.8 2008-03-26T11-15-48+2 0 82.5</Body>
</SASAlert>
```

Ο πελάτης, ο οποίος οφείλει να έχει συνδεθεί στο conference αυτό, λαμβάνει αντίτυπο του μηνύματος και με αυτόν τον τρόπο πληροφορείται για την κατάσταση συναγερμού. Εδώ βλέπουμε ότι ο αισθητήρας υγρασίας έχει καταγράψει 82.5% υγρασία, το οποίο σαφώς υπερβαίνει το ποσοστό 60% που τέθηκε σαν κάτω όριο στην αίτηση Subscribe που είδαμε προηγουμένως. Το λογισμικό του πελάτη στη συνέχεια ενδεχομένως θα ειδοποιήσει με κάποιον εποπτικό τρόπο τον χρήστη, θα καταγράψει το συμβάν σε κάποιο ημερολόγιο (log) ή θα ενεργοποιήσει μηχανισμούς αντιμετώπισης της κατάστασης, ανάλογα με το τι προβλέπεται από τον προγραμματισμό.

2.4.2.3 Λοιπές λειτουργίες και σχόλια επι των λειτουργιών του SAS

Το SAS υποστηρίζει επίσης τα ακόλουθα αιτήματα, στα οποία αναφερόμαστε επιγραμματικά:

- **DoNotification:** Ειδοποίηση προς μια υπηρεσία WNS για ασύγχρονη ενημέρωση
- **RenewAdvertisement:** Ανανέωση μιας διαφήμισης από τον παραγωγό πριν τη λήξη της
- **RenewSubscription:** Ανανέωση μιας συνδρομής από τον πελάτη πριν τη λήξη της

Ένα σημείο που μπορεί να καταλογιστεί στα αρνητικά του SAS, αλλά και των υπηρεσιών του SWE γενικότερα, είναι η έλλειψη επαναχρησιμοποίησης (reusability) υπάρχουσας λειτουργικότητας και ολοκλήρωσης (integration). Συγκεκριμένα, βλέπουμε ότι για το μηχανισμό συνδρομής-συναγερμού (subscribe-alert) ο παραγωγός πρέπει να δημοσιοποιεί τις μετρήσεις του σε ένα XMPP conference. Αν όμως ενημερώνει παράλληλα ένα SOS τότε ουσιαστικά έχουμε αποστολή των ίδιων δεδομένων εις διπλούν, το οποίο είναι αντιπαραγωγικό.

Αν το SAS μπορούσε να παρακολουθήσει τη ροή δεδομένων προς το SOS, ή αν το SAS συνεργαζόταν με το SOS, θα χρειαζόταν μοναδική και όχι διπλότυπη αποστολή μετρήσεων από τον παραγωγό. Κατά αυτόν τον τρόπο η μία υπηρεσία θα είχε αυστηρά το ρόλο της αποθήκευσης δεδομένων και η άλλη μόνο το ρόλο του ελέγχου κριτηρίων συναγερμού, χωρίς να πρέπει να αναπαράγει τμήμα της λειτουργικότητας της πρώτης. Το OGC έχει λάβει υπόψη του αυτήν την αδυναμία και διερευνεί τις διεπαφές μεταξύ των υπηρεσιών του SWE προς την κατεύθυνση αυτή.

Η επικέντρωση στο XMPP ως πρωτόκολλο ειδοποίησης είναι επίσης θέμα προς συζήτηση, παρά την ευρεία διάδοση και αποδοχή του από την κοινότητα. Ακόμα και η αναγκαιότητα ενός εξωτερικού πρωτοκόλλου για το σκοπό αυτό είναι συζητίσιμη, καθώς θεωρητικά και το ίδιο το SAS θα μπορούσε να αναλάβει αυτήν τη λειτουργία.

2.5 Υλοποιήσεις των υπηρεσιών του SWE

Οι υπηρεσίες του SWE έχουν ήδη αρχίσει να υλοποιούνται από διάφορους φορείς [10], όπως τη NASA (National Aeronautics and Space Administration) και τη Northrop Grumman που δραστηριοποιούνται στο χώρο της αεροδιαστημικής και των οπλικών συστημάτων και αεροναυπηγικής, αντίστοιχα. Το πρόγραμμα της Northrop είναι γνωστό με το όνομα PulseNet [11].

Από εταιρίες που δραστηριοποιούνται εμπορικά στο χώρο των γεωγραφικών συστημάτων πληροφόρησης έχουν ασχοληθεί με το SWE η 1Spatial Group Ltd, η Compusult Limited και άλλες.

Ερευνητικοί οργανισμοί και Πανεπιστημιακού φορείς που έχουν δείξει έμπρακτο ενδιαφέρον είναι μεταξύ άλλων το SANY Consortium, η Geomatys με το σύστημα Constellation SDI, το Πανεπιστήμιο της Μελβούρνης, κ.ά.

Ενδιαφέρον για την ακαδημαϊκή κοινότητα έχουν οι πρότυπες υλοποιήσεις των υπηρεσιών του SWE από τη γερμανική εταιρία ερευνών 52°North Initiative for Geospatial Open Source Software GmbH. Η εταιρία αυτή είναι από τους λίγους φορείς που έχουν υλοποιήσει την πλήρη σουίτα των υπηρεσιών του SWE, δηλαδή και τις τέσσερις υπηρεσίες (SOS, SAS, SPS, WNS) [12]. Επίσης αναπτύσσει λογισμικό και για συνοδευτικές εφαρμογές όπως μητρώα υπηρεσιών (registries) για την ανακάλυψη των υπηρεσιών του SWE, προγραμματιστικές διεπαφές εφαρμογών (API) για πελάτες των υπηρεσιών, περιβάλλοντα προσομοίωσης, κ.ά. Ο κώδικας της 52°North διέπεται από την άδεια GPL (General Public License) και είναι ανοιχτός στο κοινό (Open Source).

Την στιγμή εκπόνησης της τρέχουσας εργασίας οι εκδόσεις των υλοποιήσεων της 52°North ήταν οι εξής:

- **SOS**: 3.1.0
- **SAS**: 1.0.0
    - Υπάρχει και η έκδοση 2.0.0 η οποία όμως θα εγκαταλειφθεί προς ώφελος της υπηρεσίας SES που δεν πάσχει από κάποιους από τους έμφυτους περιορισμούς του SAS.
- **SPS**: 1.0.1
- **WNS**: 2.0.0

3. ΜΕΛΕΤΗ ΠΕΡΙΠΤΩΣΗΣ

Πέρα από την εισαγωγή στο αντικείμενο του SWE, η τρέχουσα εργασία στοχεύει και στην παροχή ενός πρακτικού παραδείγματος σαν απόδειξη της εφικτότητας υλοποίησης εφαρμογών που κάνουν χρήση των υπηρεσιών του SWE (proof of concept).

Η υλοποίηση αυτή φυσικά δεν έχει την πληρότητα μιας εμπορικής εφαρμογής, ούτε τη γενικότητα ενός framework για την ανάπτυξη γεωγραφικών πληροφοριακών συστημάτων. Η υλοποίηση επικεντρώθηκε στη μελέτη PULL-based και PUSH-based υπηρεσιών, συνεπώς μελετήθηκε η διαλειτουργικότητα με τις υπηρεσίες ιστού SOS και SAS αντίστοιχα.

Ο κώδικας είναι δυνατόν στο μέλλον να αποτελέσει βάση μιας εκτενέστερης πλατφόρμας λογισμικού, με αυξημένο βαθμό παραμετροποίησης, βελτιστοποιημένους (optimized) εσωτερικούς αλγορίθμους ως προς τη χρήση υπολογιστικών πόρων και επεκτασιμότητα (scalability) για υποστήριξη ευρύτερων αναπτύξεων (deployments).

Το παράδειγμα χρήσης (use case) που υιοθετήθηκε είναι η παρακολούθηση της θερμοκρασίας μιας αποθήκης ευαίσθητων αγαθών, π.χ. τροφίμων. Συνεπώς, το φυσικό μέγεθος το οποίο έχει ενδιαφέρον στην περίπτωσή μας είναι η θερμοκρασία σε βαθμούς Κελσίου (ºC) της αποθήκης που εποπτεύουμε.

Υποθέτουμε ότι ο τελικός χρήστης της εφαρμογής μας ενδιαφέρεται να βλέπει τις μεταβολές της θερμοκρασίας μέσα σε χρονικά διαστήματα που ο ίδιος ορίζει με τρόπο εποπτικό, π.χ. διάγραμμα θερμοκρασίας-χρόνου, καθώς επίσης επιθυμεί να ορίζει συνθήκες θερμοκρασίας που αν ξεπεραστούν προς κάποια κατεύθυνση (πάνω από κάποιο άνω όριο ή κάτω από ένα κατώφλι) μέσα στην αποθήκη, τότε το σύστημα να τον ειδοποιεί μέσα σε σύντομο χρονικό διάστημα ώστε να αναλάβει την πρέπουσα δράση, π.χ. έλεγχο της ψύξης, επιδιόρθωση της θέρμανσης κτλ.

3.1 SunSPOTs

Ως βάση για την υποδομή αισθητήρων στην υλοποίηση μας χρησιμοποιήθηκαν οι ασύρματοι αισθητήρες SunSPOT της Sun Microsystems. Τα αρχικά σημαίνουν Sun Small Programmable Object Technology, και είναι η πρόταση της Sun Microsystems στο χώρο των μικρών προγραμματιζόμενων ασύρματων αισθητήρων.

Εικόνα 1: Ασύρματος προγραμματιζόμενος αισθητήρας SunSPOT από τη Sun Microsystems

Κάθε SunSPOT είναι εξοπλισμένο με έναν 32-bit επεξεργαστή ARM920T χρονισμένο στα 180MHz, με 512K RAM και 4MB Flash μνήμης. Η ασύρματη δικτύωση επιτυγχάνεται μέσω πομποδέκτη κατασκευής Texas Instruments 2.4GHz, ο οποίος διαθέτει ενσωματωμένη κεραία και είναι συμβατός με το πρωτόκολλο IEEE 802.15.4. Η δικτύωση υποστηρίζει πολλαπλά άλματα (multi-hop) και η δρομολόγηση γίνεται μέσω πρωτοκόλλου AODV ή LQRP, ανάλογα με τη επιλογή. Υποστηρίζονται επίσης πρωτόκολλα επιπέδου δικτύου όπως IPv6 και LowPan [13].

Κατά την ανάπτυξη (deployment) ένα SunSPOT διαδραματίζει το ρόλο του σταθμού βάσης (base station) ενώ τα υπόλοιπα «ελεύθερα» (free-range) SunSPOT απαρτίζουν το πεδίο αισθητήρων (sensor field). Ο σταθμός βάσης συνδέεται σε υπολογιστή μέσω USB οπότε δε χρήζει ενσωματωμένου συσσωρευτή. Τα «ελεύθερα» SunSPOT εφοδιάζονται με μπαταρία 750 mAh lithium-ion που επαναφορτίζεται μέσω USB. Η διάρκεια της μπαταρίας εξαρτάται από τη χρήση, συγκεκριμένα από το πόσο συχνά στέλνει ή λαμβάνει ο πομποδέκτης, πόσο συχνά λαμβάνονται μετρήσεις από τους αισθητήρες, το φόρτο του επεξεργαστή, καθώς και από το αν είναι ρυθμισμένη ή όχι η λειτουργία «ύπνου» που αποσκοπεί στη διατήρηση ενέργειας μέσω της απενεργοποίησης μονάδων του SunSPOT. Οι καταστάσεις «ύπνου» κυμαίνονται από «ρηχό» (shallow sleep) έως «βαθύ» (deep sleep) [14].

Σε κατάσταση «βαθέως ύπνου», στην οποία απενεργοποιείται η πλειοψηφία των ηλεκτρονικών, η μπαταρία μπορεί να διαρκέσει περισσότερες από 900 μέρες. Απεναντίας, σε κατάσταση πλήρους λειτουργίας με διαρκώς αναμμένα τα LEDs, τα αποθέματα ενέργειας εξαντλούνται ακόμα και σε 3 ώρες. Με διαρκώς ενεργοποιημένο τον επεξεργαστή και τον πομποδέκτη η αναμενόμενη διάρκεια κυμαίνεται στις 7 ώρες.

Η μονάδα αισθητήρων με την οποία είναι εξοπλισμένο ένα SunSPOT εργοστασιακά περιλαμβάνει ένα επιταχυνσιόμετρο 3 αξόνων, αισθητήρα θερμοκρασίας κι έναν αισθητήρα φωτεινής έντασης. Στην ίδια πλακέτα υπάρχουν επίσης 8 LEDs τριών χρωμάτων, 6 αναλογικές είσοδοι, 2 διακόπτες, 5 ακίδες εισόδου-εξόδου (I/O pins) και 4 ακίδες εξόδου υψηλού ρεύματος. Πρόσθετοι εξωτερικοί αισθητήρες (GPS, υγρασίας κτλ.) υποστηρίζονται μέσω των εισόδων της πλακέτας.

Δε χρησιμοποιείται κάποιο λειτουργικό σύστημα, απεναντίας το SunSPOT εκτελεί απευθείας μια Εικονική Μηχανή Java (Java Virtual Machine – Java VM) ονομαζόμενη Squawk VM, που υλοποιεί τα προφίλ CLDC (Connected Limited Device Configuration) 1.1 και MIDP (Mobile Information Device Profile) 1.0 της Java 2 Micro Edition (J2ME). Το Squawk είναι αποθηκευμένο στη flash μνήμη και υλοποιεί και μερικές απαραίτητες λειτουργίες λειτουργικών συστημάτων όπως είσοδο-έξοδο, χρονοπρογραμματισμό νημάτων κ.ά. Αξίζει να σημειωθεί πως και οι οδηγοί συσκευών είναι γραμμένοι σε Java. Η τρέχουσα έκδοση του λογισμικού των SUSPOTs και των συνοδευτικών προγραμματιστικών διεπαφών (APIs) είναι η V5.0 (Red Release). Να σημειωθεί ότι το λογισμικό είναι ανοιχτού κώδικα και καλύπτεται από την άδεια GPL 2.0.

Τέλος, να αναφέρουμε ότι το λογισμικό των SunSPOT υποστηρίζει κρυπτογράφημένη επικοινωνία, ενώ η SUN έχει επίσης αναπτύξει ένα περιβάλλον προσομοίωσης και διαχείρισης αισθητήρων με το όνομα Solarium.

3.2 Λογισμικά, βιβλιοθήκες και εργαλεία

3.2.1 Πρότυπες υλοποιήσεις SOS και SAS της 52°North

Χρησιμοποιήθηκε η έκδοση 3.1.0 της υλοποίησης του SOS και η έκδοση 1.0.0 του SAS. Η έκδοση 2.0.0 του SAS δοκιμάστηκε αλλά προτιμήθηκε η παλαιότερη έκδοση για λόγους και ευχρηστίας και ύπαρξης πληρέστερης τεκμηρίωσης (documentation). Οι υπηρεσίες είναι υλοποιημένες σε Java.

3.2.2 PostgreSQL και PostGIS

Οι υλοποιήσεις της 52°North χρησιμοποιούν για αποθήκευση δεδομένων το ανοιχτού κώδικα λογισμικό διαχείρισης σχεσιακών βάσεων δεδομένων (RDBMS) PostgreSQL. Χρησιμοποιήθηκε η έκδοση 8.3.9, σε συνδυασμό με τις επεκτάσεις του πρόσθετου λογισμικού PostGIS (spatial extensions), το οποίο επιτρέπει την καταχώρηση γεωγραφικών-χωρικών δεδομένων σε μια βάση PostgreSQL.

3.2.3 Apache Tomcat Servlet Container

Οι υπηρεσίες Ιστού (web services) SOS και SAS εκτελούνται στο περιβάλλον ενός διακομιστή (server) γραμμένου σε Java, του Tomcat από το Apache Software Foundation (ASF). Ο Tomcat υποστηρίζει την εκτέλεση Java servlets (servlet container), σελίδων JSP (Java Server Pages) και υπηρεσιών Ιστού Java (JWS – Java Web Services). Αποτελεί λογισμικό ανοιχτού κώδικα και καλύπτεται από την άδεια Apache Software License.

Χρησιμοποιήθηκε η έκδοση 6.0.20.

3.2.4 Igniterealtime (Jive Software) Openfire XMPP Server

Όπως ήταν εμφανές και στα διαγράμματα ροής μηνυμάτων, ο μηχανισμός του SAS χρησιμοποιεί το πρωτόκολλο XMPP (πρώην Jabber) για την δημοσίευση (publication) μετρήσεων όσο και την αποστολή ειδοποιήσεων κατάστασης συναγερμού (alerts) μέσω άμεσων μηνυμάτων (IM - Instant Messaging). Ο διακομιστής XMPP που χρησιμοποιήθηκε ήταν ο Openfire της κοινότητας Igniterealtime, που στηρίζεται από την εταιρία Jive Software. Ο Openfire είναι ένας εξυπηρέτης συνεργασίας πραγματικού χρόνου (RTC – Real-time Collaboration) που μπορεί να υποστηρίξει διάφορες υπηρεσίες άμεσων μηνυμάτων, τόσο σε διμερείς (2-party) συνόδους, όσο και σε συνόδους με πολλαπλούς συμμετέχοντες (Conferencing, Multi User Chat), καθώς και μετάδοση πληροφοριών διαθεσιμότητας και παρουσίας (presence). Χρησιμοποιήθηκε η έκδοση 3.6.4.

3.2.5 Igniterealtime (Jive Software) Smack XMPP API

Η ίδια ανοιχτή κοινότητα που αναπτύσσει τον Openfire, έχει διαθέσει στο κοινό την προγραμματιστική διεπαφή εφαρμογών (API) Smack, για τη εύκολη χρήση του πρωτοκόλλου XMPP από εφαρμογές Java. Η τελευταία διαθέσιμη έκδοση κατά την περίοδο της υλοποίησης της εργασίας ήταν η 3.1.0.

3.2.6 Joda Time (ISO8601-compliant Java time API)

Το project ανοιχτού κώδικα Joda στοχεύει στη βελτίωση βασικής λειτουργικότητας κλάσεων και διεπαφών (APIs) από τον πυρήνα της Java. Η βιβλιοθήκη Joda Time περιέχει ριζικά βελτιωμένες υλοποιήσεις των κλάσεων Date και Time της Java, υποστηρίζοντας πολλαπλά ημερολόγια. Το προεπιλεγμένο ημερολόγιο ακολουθεί το πρότυπο ISO8601, το οποίο είναι και το πρότυπο για τις ημερομηνίες στην XML.

3.2.7 SUN Netbeans IDE

Ως ολοκληρωμένο περιβάλλον ανάπτυξης (IDE – Integrated Development Environment) χρησιμοποιήθηκε το δημοφιλές Netbeans της Sun, που υποστηρίζει τις περισσότερες τεχνολογίες της Java είτε άμεσα, είτε μέσα από plugins, καθώς και άλλες πλατφόρμες και γλώσσες προγραμματισμού όπως η C/C++, PHP, Ruby on Rails κ.ά. Υπάρχει plugin και για την ανάπτυξη εφαρμογών για SunSPOTs, με πολλές χρήσιμες επιλογές για εκτέλεση λειτουργιών που διαφορετικά πρέπει να εκτελέσει ο χρήστης μέσω ant scripts.

Τα πλεονεκτήματα της χρήσης ενός τέτοιου περιβάλλοντος είναι γνωστά:

- Αυτόματη σήμανση κώδικα (syntax highlighting) για αυξημένη αναγνωσιμότητα
- Αυτόματη συμπλήρωση κώδικα (code completion)
- Μεταγλώττιση σε πραγματικό χρόνο (real time compilation)
- Ενσωματωμένο περιβάλλον αποσφαλμάτωσης (debugger) και εκτέλεσης, κ.ά.

Χρησιμοποιήθηκε η έκδοση 6.8.

3.2.8 SUN Java Platform Standard Edition Development Kit (JDK)

Η Standard έκδοση της πλατφόρμας Java περιλαμβάνει μεταγλωτιστή, περιβάλλον εκτέλεσης (JRE - Java Runtime Environment) ή αλλιώς εικονική μηχανή (JVM – Java Virtual Machine) και βιβλιοθήκες της Java για ανάπτυξη φορητών (portable) εφαρμογών γενικής χρήσης. Είναι γνωστή και ως JDK (Java Development Kit).

Η έννοια της φορητότητας εδώ αφορά στη δυνατότητα εκτέλεσης του ίδιου κώδικα σε διαφορετικά λειτουργικά συστήματα, αρκεί να υπάρχει η κατάλληλη για το δεδομένο σύστημα εικονική μηχανή Java, που αναλαμβάνει να εκτελέσει το μεταγλωττισμένο κώδικα (bytecode) της εφαρμογής. Αποτελεί ένα από τα βασικά πλεονεκτήματα της Java.

Στην εργασία χρησιμοποιήθηκε η έκδοση 1.6.

3.2.9 SUN Java Platform Micro Edition

Η Micro έκδοση της πλατφόρμας Java προσανατολίζεται προς τις φορητές (mobile) συσκευές και ενσωματωμένα (embedded) συστήματα. Εκατομμύρια φορητές συσκευές, κινητά, καθώς και ελεγκτές (controllers) παγκοσμίως εκτελούν λογισμικό γραμμένο σε Java. Χωρίζεται σε επιμέρους προφίλ και υποδιαιρέσεις αυτών, ανάλογα με τις δυνατότητες των συσκευών που υποστηρίζονται. Τα δύο κύρια προφίλ είναι το CLDC και το CDC.

Το CLDC (Connected Limited Device Configuration) αποτελεί ένα μινιμαλιστικό υποσύνολο της Java Standard Edition, περιέχοντας τα αυστηρώς απαραίτητα για τη λειτουργία μιας εικονικής μηχανής Java. Είναι κατάλληλη για συσκευές με χαμηλή

υπολογιστική ισχύ και ενδεχομένως περιορισμένα ενεργειακά αποθέματα. Περιλαμβάνει τα υπό-προφίλ MIDP (Mobile Information Device Profile) το οποίο ενσωματώνει βασικές βιβλιοθήκες για διεπαφές χρήστη, παιχνίδια και δισδιάστατα γραφικά, και το IMP (Information Module Profile), που αφορά σε ενσωματωμένα συστήματα χωρίς δυνατότητες απεικόνισης, συνεπώς δεν έχει τις βιβλιοθήκες αυτές.

Το CDC (Connected Device Profile) περιλαμβάνει ένα πιο πλούσιο υποσύνολο της Java Standard Edition, απευθυνόμενο σε συσκευές με περισσότερες δυνατότητες. Περιλαμβάνει το Foundation Profile, το Personal Basis Profile και το Personal Profile.

3.2.10 Apache Ant και Maven

Το Ant (“Another Neat Tool”) της Apache Software Foundation είναι ένα από τα δημοφιλέστερα εργαλεία για την αυτοματοποίηση της διαδικασίας μεταγλώττισης (build automation) εφαρμογών Java. Χρησιμοποιεί XML για την παραμετροποίηση της διαδικασίας μεταγλώττισης (compilation), σύνδεσης (linking) και ανάπτυξης (deployment). Το SunSPOT API και το Netbeans χρησιμοποιούν το Ant κατά κόρον.

Το Maven είναι ένα εφάμιλλο εργαλείο του ίδιου οργανισμού. Το SOS της 52°North πλέον χρησιμοποιεί αυτό αντί του Ant.

Χρησιμοποιήθηκαν οι εκδόσεις Ant 1.7.0 και Maven 2.2.1.

3.2.11 Apache XMLBeans

Η μεταχείριση εγγράφων XML θα ήταν επίπονη υπόθεση αν δεν είχαν αναπτυχθεί τα κατάλληλα εργαλεία για την αποκωδικοποίηση (parsing) και δημιουργία τους. Ένα από τα πιο διαδεδομένα API του είδους είναι τα XMLBeans από το Apache Software Foundation.

Τα XMLBeans επιτρέπουν τη μεταγλώττιση ενός XML εγγράφου (document) σε αντικείμενο (object) Java, το οποίο στη συνέχεια μπορεί ο χρήστης να μεταχειριστεί σαν μια κλάση (class) με τις δικές της getter και setter μεθόδους. Είναι δυνατόν επίσης να μεταγλωττιστεί το πρότυπο σχήμα ενός εγγράφου XML (XML schema document - .xsd). Κατά αυτόν τον τρόπο μπορούν να δημιουργηθούν νέα στιγμιότυπα εγγράφων XML προγραμματιστικά από μια εφαρμογή Java. Τέλος, η βιβλιοθήκη επιτρέπει την τυχαία προσπέλαση μέσα σε μεταγλωττισμένα έγγραφα XML μέσω δρομέων (cursors) καθώς και τη διενέργεια στοιχειωδών SQL-like επερωτήσεων εντός αυτών (XQuery).

Παρότι η ενσωματωμένη στο JDK βιβλιοθήκη JAXB (Java Architecture for XML Binding) της Sun πλέον παρέχει εφάμιλλη λειτουργικότητα με τη λύση της Apache, προτιμήθηκαν τα XMLBeans καθώς και το λογισμικό της 52°North κάνει χρήση του API αυτού για την κωδικοποίηση και αποκωδικοποίηση εγγράφων XML.

Η έκδοση που ήταν διαθέσιμη την περίοδο πραγματοποίησης της εργασίας ήταν η 2.4.0.

3.2.12 SUN SunSPOT API

Το API της Sun για περιλαμβάνει κλάσεις για τη διαχείριση των περιφερειακών, της δικτύωσης, της βασικής εισόδου-εξόδου και των λοιπών λειτουργιών των SunSPOTs. Υλοποιούνται στο προφίλ MIDP του CLDC της Java Micro Edition.

Οι εφαρμογές των SunSPOT χωρίζονται σε δύο κατηγορίες, τις εφαρμογές που εκτελούνται στο SunSPOT καθαυτό (SunSPOT applications), οι οποίες βασίζονται στην κλάση Midlet του MIDP, και στις εφαρμογές που εκτελούνται σε συνεργαζόμενο υπολογιστή (host applications), οι οποίες έχουν πρόσβαση σε όλη τη λειτουργικότητα της Java Standard Edition, και σε ένα απαραίτητο υποσύνολο του SunSPOT API για την επικοινωνία με τους κόμβους.

Τα SunSPOT διαθέτουν δύο κύριους μηχανισμούς ασύρματης επικοινωνίας μεταξύ τους και με τις συνεργαζόμενες εφαρμογές: την κλάση RadioGramConnection για stream-based μεταδόσεις και την κλάση RadioStreamConnection για datagram-based μεταδόσεις. Για ενσύρματη επικοινωνία υποστηρίζεται διεπαφή USB και USART.

Τέλος, αξίζει να σημειωθεί ότι υποστηρίζεται το πρωτόκολλο HTTP, χρησιμοποιώντας τον υπολογιστή στον οποίο είναι συνδεδεμένος ο σταθμός βάσης σαν proxy.

Χρησιμοποιήθηκε η έκδοση V5.0 (κωδικό όνομα “Red Release”).

3.2.13 SUN Swing και AWT API

Τα γνωστά APIs της SUN για την ανάπτυξη γραφικών διεπαφών χρήστη (GUI – Graphical User Interface) χρησιμοποιήθηκαν για το γραφικό περιβάλλον της εφαρμογής του πελάτη (client). Αξιοποιήθηκε ο ενσωματωμένος GUI Builder του Netbeans. Το Swing και το AWT (Abstract Window Toolkit) αποτελούν βιβλιοθήκη που περιλαμβάνεται στο Java Development Kit (JDK), ως μέρος των βασικών κλάσεων Java (JFC – Java Foundation Classes).

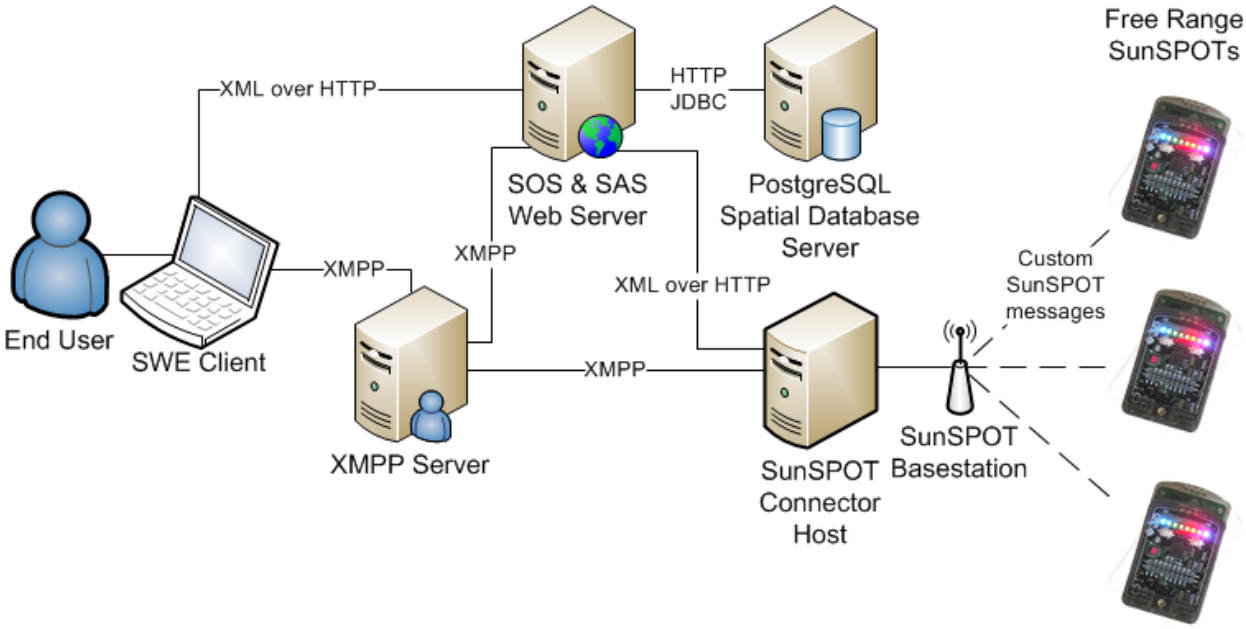
3.2.14 JFreeChart και Jcommon

Η βιβλιοθήκη JFreeChart επιτρέπει την απεικόνιση γραφικών παραστάσεων διάφορων μορφών (ιστογράμματα, ραβδογράμματα, scatter plots, pie chart, time series, κτλ.). Στα πλαίσια της εργασίας χρησιμοποιήθηκε για τη γραφική απεικόνιση των αποτελεσμάτων κατά την ανάκτηση τους από το SOS από την εφαρμογή του πελάτη. Η βιβλιοθήκη JCommon παρέχει κοινή λειτουργικότητα πάνω στην οποία βασίζεται και το JFreeChart API.

Χρησιμοποιήθηκαν οι εκδόσεις 1.0.13 και 1.0.16 αντίστοιχα.

3.3 Σχεδιασμός των επιμέρους μονάδων λογισμικού

Η τοπολογία του δικτύου για μια περίπτωση σαν αυτή που μελετάμε είναι η ακόλουθη:



Σχήμα 5: Τοπολογία δικτύου για το σενάριο μας

Οι μονάδες λογισμικού που απαιτούνται για τη λειτουργία του συστήματος, και οι δικτυακές οντότητες στις οποίες φιλοξενούνται αντίστοιχα είναι:

- **SunSPOT Connector Host Application:** Κεντρική εφαρμογή που αποτελεί τη διεπαφή του παραγωγού δεδομένων με τις υπηρεσίες SOS και SAS. Επικοινωνεί με τα ελεύθερα SunSPOTs (**Free range SunSPOTs**) μέσω ενός ενσύρματου SunSPOT που έχει το ρόλο του σταθμού βάσης (**SunSPOT Basestation**). Εκτελείται σε έναν υπολογιστή που τοποθετείται στο χώρο όπου σκοπεύουμε να λάβουμε μετρήσεις, εν προκειμένω στην αποθήκη που εποπτεύουμε.
- **SunSPOT Application:** Εφαρμογή που εκτελείται στους ασύρματους SunSPOT κόμβους, η οποία χειρίζεται τη σύνδεση με την εφαρμογή Connector και τη συλλογή μετρήσεων.
- **SOS & SAS Web Server:** Διακομιστής Ιστού (Web) που φιλοξενεί τις υπηρεσίες Ιστού (web services) SOS και SAS. Η διεπαφή του με τα άλλα συστατικά είναι μέσω HTTP (προς τη βάση δεδομένων και με τον πελάτη και παραγωγό μέσω XML μηνυμάτων) και XMPP (προς τον XMPP Server).
- **PostgreSQL Spatial Database Server:** Βάση δεδομένων στην οποία αποθηκεύουν δεδομένα οι υπηρεσίες SOS και SAS. Μπορεί να φιλοξενείται στο ίδιο μηχάνημα όπου εκτελούνται και οι υπηρεσίες αυτές ή και σε διαφορετικό υπολογιστή.
- **XMPP Server:** Διακομιστής που υλοποιεί το XMPP πρωτόκολλο, υποστηρίζοντας την άμεση μηνυματοδοσία (Instant Messaging) σε συνόδους πολλαπλών συμμετεχόντων (Conference / Multi User Chat). Δέχεται μηνύματα από τον Connector και το SAS και τα κοινοποιεί στα υπόλοιπα μέλη της

συνόδου, στα πλαίσια του μηχανισμού συνδρομής-συναγερμού (Subscribe-Alert).

- **SWE Client:** Εφαρμογή του πελάτη-καταναλωτή γεωγραφικών δεδομένων και δεδομένων αισθητήρων. Φιλοξενείται σε έναν υπολογιστή οπουδήποτε στο διαδίκτυο. Επικοινωνεί μέσω HTTP με το διακομιστή που εκτελεί τα SOS και SAS, και μέσω XMPP με το διακομιστή XMPP όπου ανακοινώνονται τα μηνύματα συναγερμού.

Για απλότητα, στη δική μας περίπτωση ο διακομιστής που φιλοξενεί όλες αυτές τις μονάδες είναι ο ίδιος, ισοδυναμώντας με τον τοπικό υπολογιστή (localhost). Επίσης στα πλαίσια της εργασίας θα θεωρήσουμε για ευκολία ότι κάθε component γνωρίζει τις διευθύνσεις των άλλων οπότε θα παραλειφθούν διαδικασίες όπως η ανακάλυψη των URL του SOS και SAS με επερωτήσεις σε ένα μητρώο υπηρεσιών Ιστού, π.χ. UDDI (Universal Description Discovery and Integration).

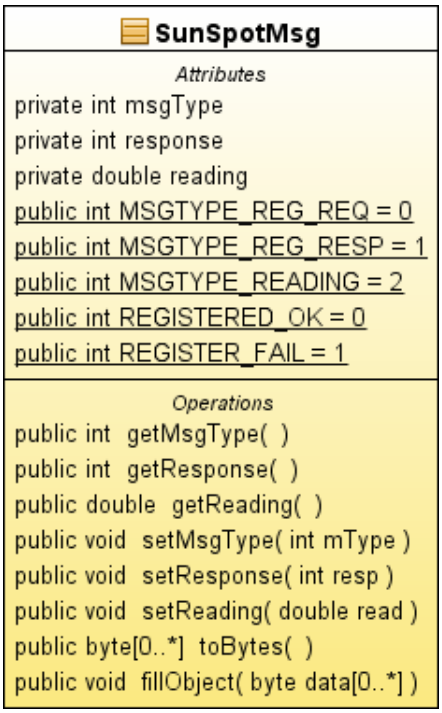
Στη συνέχεια θα εξετάσουμε τη βασική λειτουργικότητα των συστατικών που δημιουργήθηκαν για αυτήν την εφαρμογή. Είναι σκόπιμο να ξεκινήσουμε με τις κοινόχρηστες βιβλιοθήκες.

Σημαντικές κοινόχρηστες λειτουργίες ενσωματώθηκαν σε βοηθητικές κλάσεις. Οι βιβλιοθήκες που δημιουργήθηκαν είναι οι:

- SunSPOT Library, για την επικοινωνία μεταξύ των SunSPOTs και του Connector.
- XMPP Utilities, για τις βασικές αλληλεπιδράσεις του Connector και του Client με τον XMPP Server

3.3.1 SunSPOT Library

Η βιβλιοθήκη αυτή ουσιαστικά αποτελεί ένα SunSPOT Message API, δηλαδή μια προγραμματιστική διεπαφή για την επικοινωνία με τα SunSPOTs, για το δεδομένο σχεδιασμό που έχουμε επιλέξει. Περιέχει την κλάση SunSpotMsg που ορίζει τη μορφή των μηνυμάτων μεταξύ των SunSPOTs και του Connector. Τα μηνύματα αυτά δεν ορίζονται από κάποια σύσταση ή πρότυπο, οπότε υλοποιήθηκε ένα απλό αυτοσχέδιο (custom) πρωτόκολλο ερώτησης-απάντησης (request – response). Το διάγραμμα κλάσης (class diagram) φαίνεται στο ακόλουθο σχήμα:



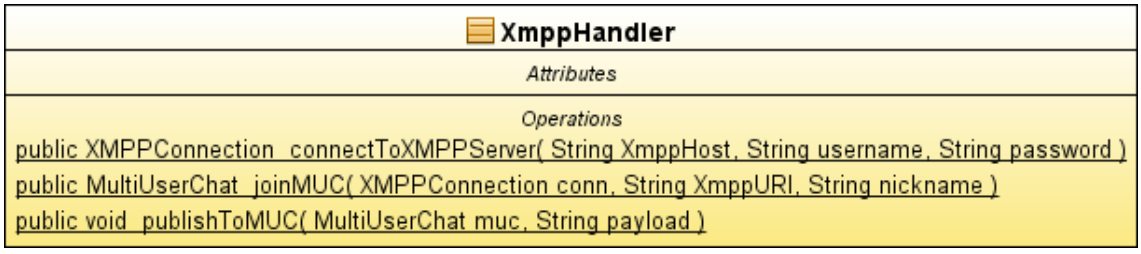
Σχήμα 6: Διάγραμμα κλάσης για το SunSpotMsg

Οι τύποι μηνυμάτων που ορίζονται είναι αίτηση εγγραφής (MSG_TYPE_REG_REQ), απάντηση σε αίτηση εγγραφής (MSG_TYPE_REG_RESP) και μήνυμα μέτρησης (MSG_TYPE_READING). Για την απάντηση σε αίτηση εγγραφής ορίζονται δύο αποτελέσματα, επιτυχία (REGISTERED_OK) ή αποτυχία (REGISTER_FAIL).

Οι μέθοδοι της κλάσης περιλαμβάνουν τυπικές Getter και Setter μεθόδους κατά τη συνήθη πρακτική του βασισμένου σε αντικείμενα προγραμματισμού (object based programming). Η μέθοδος toBytes() μετατρέπει ένα στιγμιότυπο της κλάσης σε δυαδική αναπαράσταση για αποστολή εντός πακέτων Radiogram στην ασύρματη διεπαφή με το σταθμό βάσης και τον υπολογιστή που φιλοξενεί τον Connector. Η μέθοδος fillObject κάνει το ακριβώς αντίστροφο, γεμίζοντας ένα κενό στιγμιότυπο SunSpotMsg με είσοδο ένα σύνολο από bytes που αποτελούν τη δυαδική αναπαράσταση των δεδομένων του μηνύματος, όπως αυτά έχουν ληφθεί σε ένα Radiogram.

3.3.2 XMPP Utilities

Η βιβλιοθήκη XMPP Utilities περιέχει την κλάση XmppHandler, η οποία διεθέτει τρεις μεθόδους για κοινές λειτουργίες που χρειάζονται στον Connector και τον πελάτη.



Σχήμα 7: Διάγραμμα κλάσης για το XmppHandler

Η μέθοδος connectToXMPPServer() συνδέεται σε ένα XMPP Server δοθείσης της διεύθυνσης του διακομιστή, και του συνδυασμού ονόματος χρήστη – κωδικού πρόσβασης (user name - password) για το χρήστη που ζητά τη σύνδεση. Στην καλούσα συνάρτηση επιστρέφεται το αντικείμενο της σύνδεσης (connection handle). Είναι απαραίτητη η δημιουργία λογαριασμών χρήστη στο περιβάλλον διαχείρισης του XMPP Server για τον Connector και τον Client, με τα αντίστοιχα διαπιστευτήρια.

Με τη μέθοδο joinMUC(), ένα συστατικό μπορεί να συνδεθεί σε ένα συγκεκριμένο Multi User Chat room / conference, όπως αυτό ορίζεται από το XMPP URI που παρέχεται. Η μέθοδος επιστρέφει το αντικείμενο του MUC.

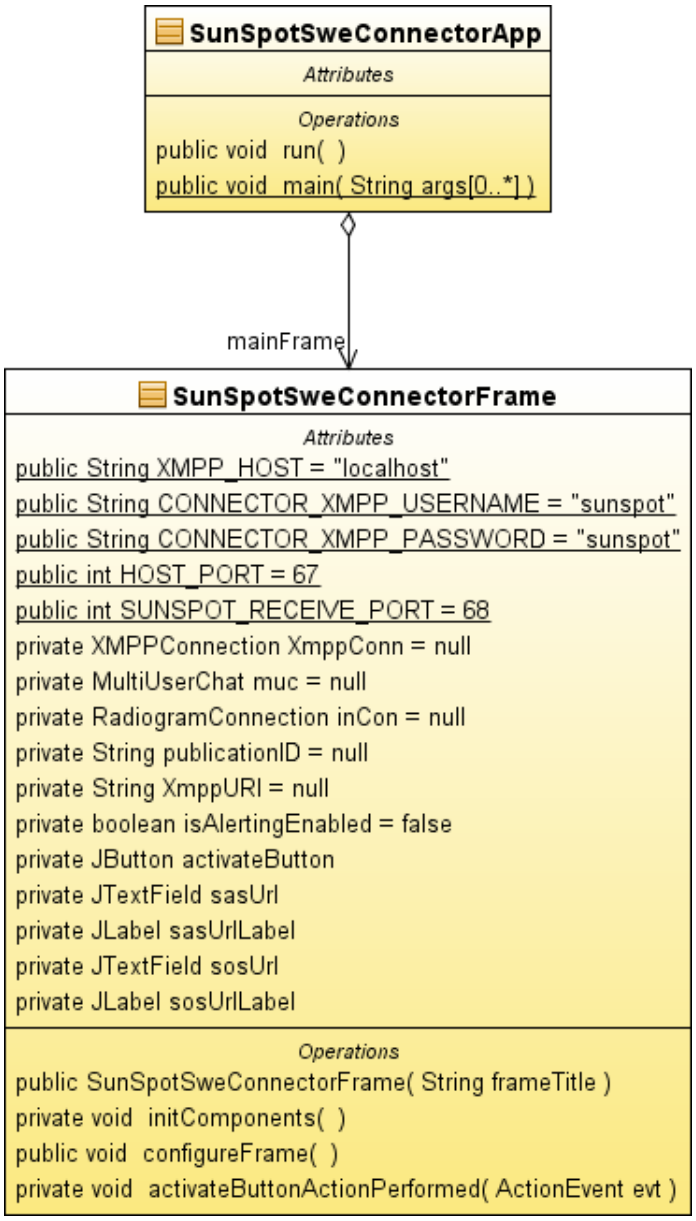
Τέλος, η publishToMUC() δημοσιεύει στο δοθέν conference το μήνυμα που περνιέται ως δεύτερη παράμετρος.

3.3.3 SunSPOT SWE Connector Host Application

Παρά τη δυνατότητα των SunSPOTs να αποστέλλουν πακέτα μέσω HTTP, προτιμήθηκε η υλοποίηση ενός κεντρικού συστατικού που θα παρεμβάλλεται μέσω των κόμβων και των υπηρεσιών SOS και SAS, και θα αναλάβει το ρόλο της διεπαφής, μεταφράζοντας μηνύματα SunSpotMsg σε SWE XML μηνύματα και το ανάποδο.

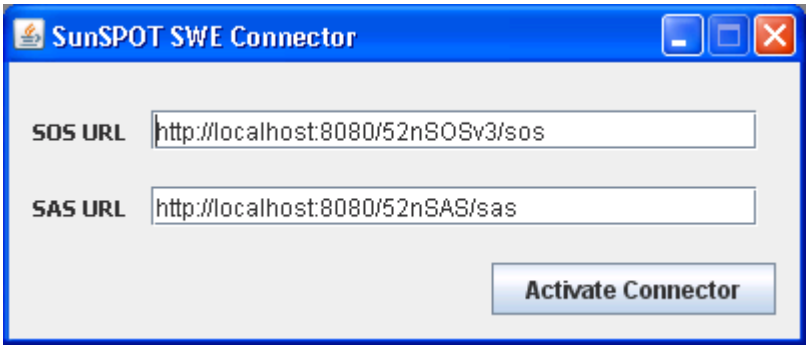
Ένας κύριος λόγος είναι ότι η δημιουργία XML μηνυμάτων με χρήση των XML Beans είναι αρκετά απαιτητική για τις περιορισμένες δυνατότητες επεξεργασίας και αποθήκευσης ενός SunSPOT, δεδομένου και του μεγέθους της βιβλιοθήκης αυτής (~2.6 MB). Εξετάστηκε και το ενδεχόμενο αποστολής των SWE μηνυμάτων από τα SunSPOTs με άμεση συμπλήρωση έτοιμου κειμένου XML, αλλά θεωρήθηκε ότι αυτό ξέφευγε από το ερευνητικό πνεύμα της εργασίας. Εξάλλου, τα SunSPOT θα έπρεπε και πάλι να μπορουν να αποκωδικοποιούν τις XML απαντήσεις, ακόμα και αν οι αιτήσεις ήταν στατικά δημιουργημένες. Συν τοις άλλοις, μια εφαρμογή που εκτελείται σε υπολογιστή έχει πρόσβαση στην πλήρη λειτουργικότητα της Java Standard Edition και φυσικά σε σαφώς μεγαλύτερη υπολογιστική ισχύ, οπότε προτιμήθηκε αυτή η λύση.

Στα παρακάτω σχήματα φαίνονται τα διαγράμματα των κλάσεων που χρησιμοποιεί η εφαρμογή του Connector:



Σχήμα 8: Διάγραμμα κλάσης για τον Connector

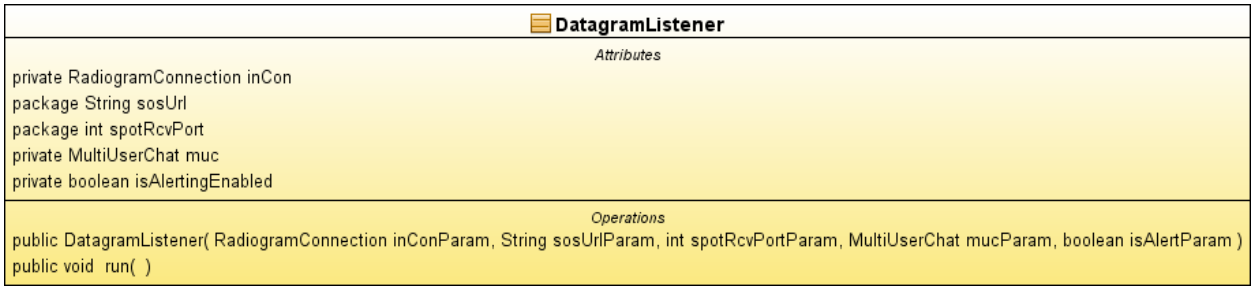
Η κλάση SunSpotConnectorApp περιέχει τη συνάρτηση main() της εφαρμογής. Η main() δημιουργεί το κυρίως πλαίσιο του γραφικού περιβάλλοντος του Connector, που αποτελεί στιγμιότυπο της κλάσης SunSpotSweConnectorFrame (που επεκτείνει την κλάση JFrame του Swing API) και έχει την ακόλουθη εικόνα:



Εικόνα 2: Η διεπαφή χρήστη του Connector

Σαφώς περισσότερη έμφαση στη διεπαφή δόθηκε στην εφαρμογή του πελάτη-καταναλωτή, καθώς αυτή είναι εκτεθειμένη στον τελικό χρήστη. Η διεπαφή του Connector σχεδιάστηκε πολύ απλή, με σκοπό ο διαχειριστής να απαιτείται μόνο να εφοδιάζει τα URL του SOS και SAS και να τον ενεργοποιεί. Η ενεργοποίηση των SOS και SAS σκελών γίνεται μαζί, όπως θα δούμε στη συνέχεια. Ο Connector τερματίζεται με το κλείσιμο του παραθύρου.

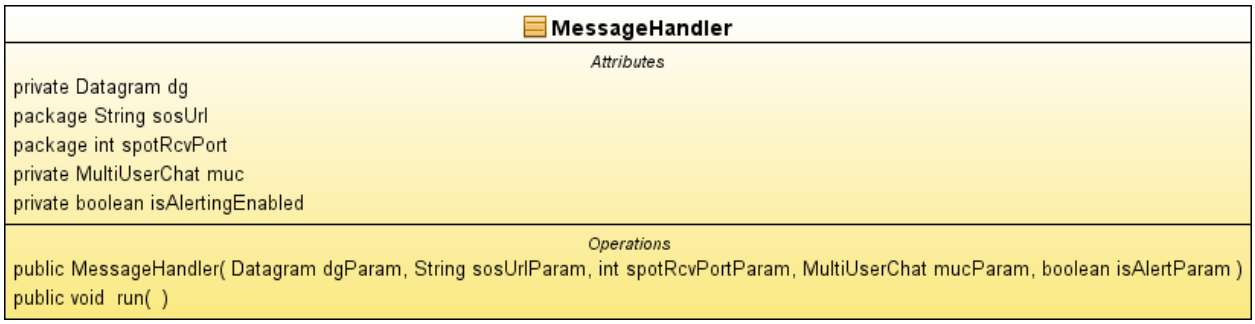
Ο constructor SunSpotSweConnectorFrame() και οι μέθοδοι initComponents() και configureFrame() καλούνται κατά την εκκίνηση της εφαρμογής για την αρχικοποίηση του πλαισίου. Ανάμεσα στα άλλα, ρυθμίζεται η διευθέτηση των συστατικών του παραθύρου στην οθόνη και ορίζεται ένα νήμα για το χειρισμό του τερματισμού του παραθύρου (window closing hook). Η μέθοδος activateButtonActionPerformed() καλείται όταν πιεστεί το κουμπί “Activate Connector”. Πρωτού αναλύσουμε τον αλγόριθμο που εκτελείται στη συγκεκριμένη μέθοδο, είναι χρήσιμο να δούμε δύο κλάσεις νημάτων (threads) που χρησιμοποιεί, τις DatagramListener και MessageHandler, καθώς και μια κλάση που χειρίζεται την αποστολή αιτήσεων SWE για λογαριασμό του παραγωγού δεδομένων:



Σχήμα 9: Διάγραμμα κλάσης για τον DataGramListener

Το νήμα DatagramListener αποτελεί τον υποδοχέα πακέτων τύπου Datagram από τα SunSPOTs προς τον Connector. Δημιουργείται για να μην μπλοκάρεται η εφαρμογή κατά την αναμονή νέων μηνυμάτων, καθώς η κλήση στη μέθοδο receive() του DatagramConnection είναι blocking. Έτσι το κυρίως νήμα της εφαρμογής αποδεσμεύεται και είναι σε θέση να επεξεργάζεται συμβάντα όπως ο τερματισμός του παραθύρου.

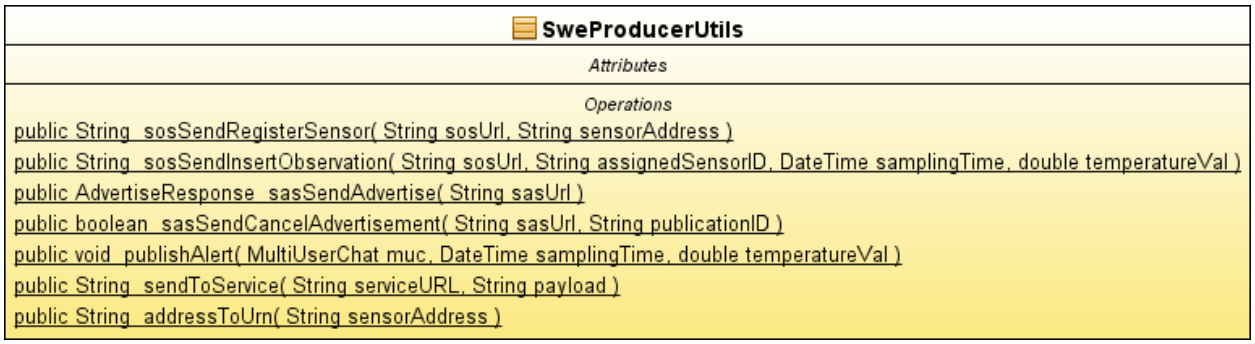
Η μέθοδος run() του νήματος υποδέχεται νέο μήνυμα από τα SunSPOTs και το διαβιβάζει σε ένα νέο νήμα τύπου MessageHandler, το οποίο αναλαμβάνει την κυρίως επεξεργασία. Το διάγραμμα της κλάσης αυτής είναι το ακόλουθο:



Σχήμα 10: Διάγραμμα κλάσης για τον MessageHandler

Ένα νήμα MessageHandler δημιουργείται για κάθε νέο πακέτο. Ανάλογα με το είδος του SunSpotMsg που περιέχεται (MSG_TYPE_REG_REQ ή MSG_TYPE_REG_READING), η μέθοδος run() αποστέλλει στο SOS και SAS τις αντίστοιχες SWE αιτήσεις. Το νήμα μπλοκάρεται εν αναμονή της απάντησης από την αντίστοιχη υπηρεσία, το οποίο δεν αποτελεί βέλτιστο σχεδιασμό, αλλά επηρεάζει μόνο το τρέχον νήμα.

Η βοηθητική κλάση SweProducerUtils περιέχει μεθόδους για την αποστολή αιτήσεων που αφορούν στον παραγωγό δεδομένων, προς το SOS και το SAS, καθώς και την αλληλεπίδραση με τον XMPP Server. Το αντίστοιχο διάγραμμα κλάσης είναι:



Σχήμα 11: Διάγραμμα κλάσης για τη βιβλιοθήκη SweProducerUtils

Ο ρόλος κάθε μεθόδου κατανοείται εύκολα από την ονομασία της. Για παράδειγμα, η μέθοδος sosSendRegisterSensor αποστέλλει στο SOS που προσδιορίζεται από την παράμετρο sosUrl, μια αίτηση RegisterSensor για λογαριασμό του αισθητήρα με διεύθυνση IEEE sensorAddress. Κάθε μέθοδος που αλληλεπιδρά με το SOS ή SAS καλεί τη μέθοδο sendToService() για το χειρισμό της αποστολής του HTTP πακέτου και την λήψη της αντίστοιχης απάντησης 200-OK, που φέρει την απάντηση από την υπηρεσία.

Η μέθοδος publishAlert() χρησιμεύει για τη δημοσίευση μιας μέτρησης στο XMPP conference που έχει δημιουργηθεί μετά τη διαδικασία του Advertise. Η μέθοδος addressToUrn() μετατρέπει μια IEEE διεύθυνση κόμβου SunSPOT σε OGC URN (Universal Resource Name).

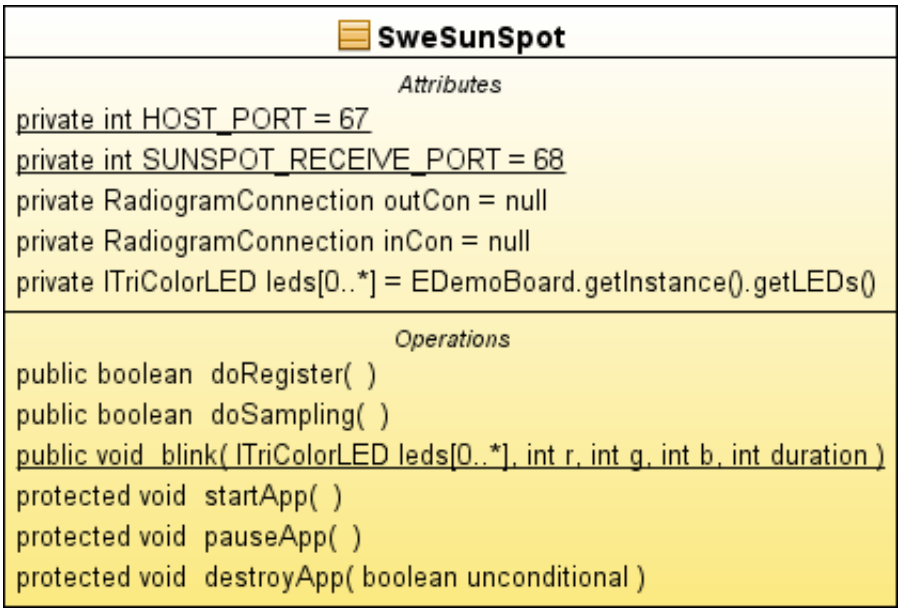
Συνολικά, η λειτουργικότητα του Connector που ενεργοποιείται με το πάτημα του “Activate Connector” συνοψίζεται στα εξής βήματα:

1. Ελέγχεται αν έχουν συμπληρωθεί τα πεδία “SOS URL” και “SAS URL”.

2. Δημιουργείται θύρα (HOST_PORT) υποδοχής Datagram πακέτων από τα SunSPOTs.
3. Αποστέλλεται αίτηση Advertise στο SAS για διαθεσιμότητα δημοσιεύσεων τιμών θερμοκρασίας. Για το σκέλος του alerting προτιμήθηκε να αποστέλλεται μία αίτηση για όλη την εφαρμογή, αντί για ξεχωριστή διαφήμιση για κάθε αισθητήρα. Η σύμβαση έγινε χάριν απλότητας, με θυσία φυσικά στο βαθμό πιστότητας (granularity) ως προς το ποιος κόμβος συγκεκριμένα μέτρησε τιμή θερμοκρασίας που πυροδότησε ένα συναγερμό. Επίσης, δεν τέθηκε χρονικό όριο λήξης της διαφήμισης.
4. Πραγματοποιείται σύνδεση στον XMPP Server με βάση το XMPP URI που επιστρέφεται στο AdvertisementResponse. Υπάρχει περιορισμός σε ένα ενεργό Advertisement για την τρέχουσα εφαρμογή.
5. Εκκινείται το νήμα λήψης πακέτων DatagramListener.
6. Κάθε νέο πακέτο από ένα SunSPOT ανατίθεται σε ένα νέο νήμα MessageHandler για non-blocking επεξεργασία. Έλεγχεται αν είναι:
    - Αίτηση Εγγραφής (MSG_TYPE_REG_REQ):
        - Αποστέλλεται αίτηση RegisterSensor προς το SOS. Να σημειωθεί ότι μόνο οι ικανότητα μέτρησης θερμοκρασίας του SunSPOT υποβάλλεται στο SOS μέσω της αίτησης. Οι δυνατότητες μέτρησης φωτεινής έντασης και επιτάχυνσης δεν αξιοποιήθηκαν, αν και μπορούν να προστεθούν με επέκταση της εφαρμογής του SunSPOT και του Connector.
        - Κατόπιν λήψης της απάντησης από το SOS, δημιουργείται θύρα αποστολής Datagram πακέτων προς το SunSPOT που έστειλε την αίτηση (SUNSPOT_RECEIVE_PORT).
        - Σε περιπτωση επιτυχίας ή εύρεσης υπάρχουσας εγγραφής στο μητρώο του SOS, αποστέλλεται στο SunSPOT απάντηση (MSG_TYPE_REG_RESP) με θετική ένδειξη (REGISTERED_OK).
        - Σε περίτωση αποτυχίας, το SunSPOT λαμβάνει απάντηση με αρνητική ένδειξη (REGISTER_FAIL) ώστε να αναστείλει τη λειτουργία του.
    - Μέτρηση (MSG_TYPE_READING):
        - Αποστέλλεται αίτηση InsertObservation προς το SOS με την τιμή θερμοκρασίας που ανίχνευσε το SunSPOT.
        - Η απάντηση InsertObservationResponse λαμβάνεται και αποκωδικοποιείται, αλλά το αναγνωριστικό της καταχώρησης της μέτρησης (observation ID) δε χρησιμεύει παραπέρα.
        - Η μέτρηση επίσης δημοσιεύεται στο XMPP conference για την ισχύουσα διαφήμιση. Η δημοσίευση αυτή, όπως αναφέραμε, είναι ορατή και στο SAS, το οποίο θα τη συγκρίνει με τυχόν συνδρομές ειδοποίησης συναγερμού από εφαρμογές πελατών.
7. Αν ληφεί σήμα τερματισμού, η εφαρμογή αποδεσμεύει τους πόρους που βρίσκονται σε χρήση:
    - Αποσύνδεση από τον XMPP Server.
    - Αποστολή CancelAdvertisement στο SAS για την ισχύουσα διαφήμιση.
    - Τερματισμός της υποδοχής μηνυμάτων από τα SunSPOT.

3.3.4 SWE SunSPOT Midlet

Η κλάση Java που εκτελείται στο SunSPOT επεκτείνει τη βασική κλάση Midlet του υποπροφίλ MIDP της Java Micro Edition. Το διάγραμμα κλάσης του Midlet που υλοποιήθηκε φαίνεται εδώ:



Σχήμα 12: Διάγραμμα κλάσης για το Midlet SweSunSpot

Η μέθοδος startApp() εκκινεί την εφαρμογή. Η λειτουργικότητα είναι απλή:

- Ανοίγεται μία θύρα για αποστολή μηνυμάτων προς τον Connector (HOST_PORT) και μία για τη λήψη της απάντησης στο αίτημα εγγραφής που θα σταλεί στη συνέχεια (SUNSPOT_RECEIVE_PORT).
- Με κλήση της μεθόδου doRegister() αποστέλλεται μηνύμα SunspotMsg προς τον Connector για εγγραφή (MSG_TYPE_REG_REQ).
- Αν η απάντηση (MSG_TYPE_RESP) είναι θετική (REGISTERED_OK), με χρήση της μεθόδου blink() τα LED ανάβουν στιγμιαία σε πράσινο χρώμα και με κλήση της μεθόδου doSampling() αρχίζει η διαδικασία δειγματοληψίας θερμοκρασίας ανά 30 δευτερόλεπτα. Όπως αναφέρθηκε στην προηγούμενη υποενότητα για τον Connector, αξιοποιείται μόνο ο αισθητήρας θερμοκρασίας του SunSPOT στην τρέχουσα εργασία.
    - Κάθε μέτρηση περικλείεται σε ένα SunSpotMsg τύπου MSG_TYPE_READING και εκμπέμπεται προς τον Connector.
    - Το SunSPOT δεν περιμένει απάντηση από τον Connector για αποστολή μέτρησης.
    - Κάθε επιτυχής λήψη μέτρησης και αποστολή της στον Connector συνοδεύεται από στιγμιαία φωτεινή ένδειξη χρώματος μπλε από τα LED.
    - Το SunSPOT παραμένει στην κατάσταση δειγματοληψίας μέχρι να τερματιστεί με εξωτερική εντολή. Σίγουρα αυτός ο μηχανισμός επιδέχεται

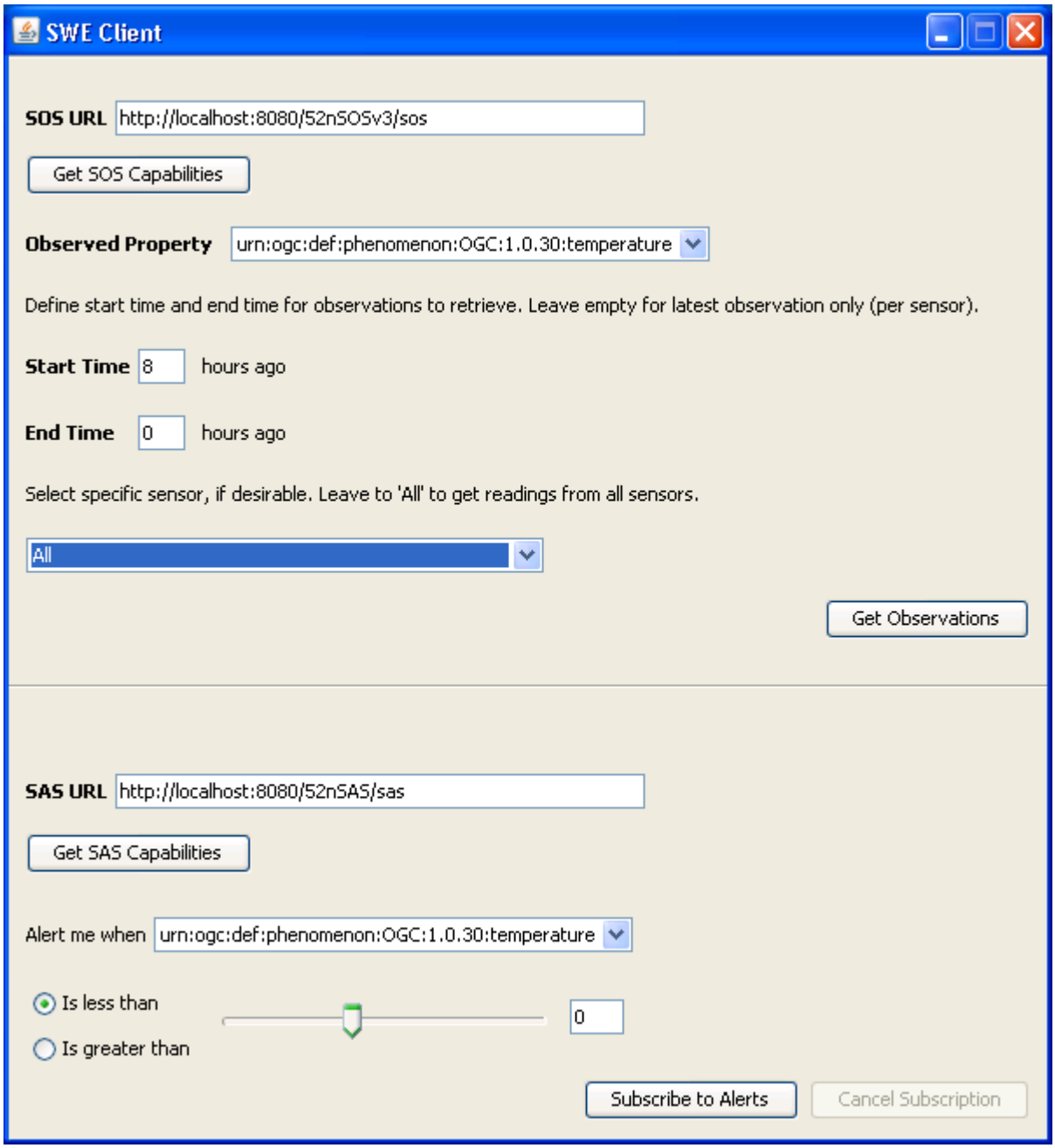
βελτίωσης, π.χ. υποστήριξη παύσης, επανεκκίνησης ή τερματισμού κατόπιν εντολής του Connector.

- Αν ληφθεί αρνητική απάντηση (REGISTER_FAIL), τα LED θα αναβοσβήσουν κόκκινα και το Midlet τερματίζεται.

3.3.5 SWE Client Swing Application

Η εφαρμογή του πελάτη είναι ένα Swing application, όπως και ο Connector, με τη διαφορά ότι η διεπαφή είναι πολύ πιο πλούσια σε επιλογές, ενώ περιέχει πεδία και λίστες επιλογών που παραμετροποιούνται δυναμικά ανάλογα με τα αποτελέσματα στις αιτήσεις που υποβάλλει ο χρήστης.

Οι λειτουργίες που αφορούν στο SOS και το SAS είναι ομαδοποιημένες σε 2 τμήματα που χωρίζονται οπτικά με μια διαχωριστική γραμμή, και μπορούν να χρησιμοποιηθούν ανεξάρτητα. Η εικόνα που παρουσιάζει η διεπαφή κατόπιν υποβολής των αιτημάτων GetCapabilities προς αμφότερες τις υπηρεσίες που υποστηρίζονται είναι:



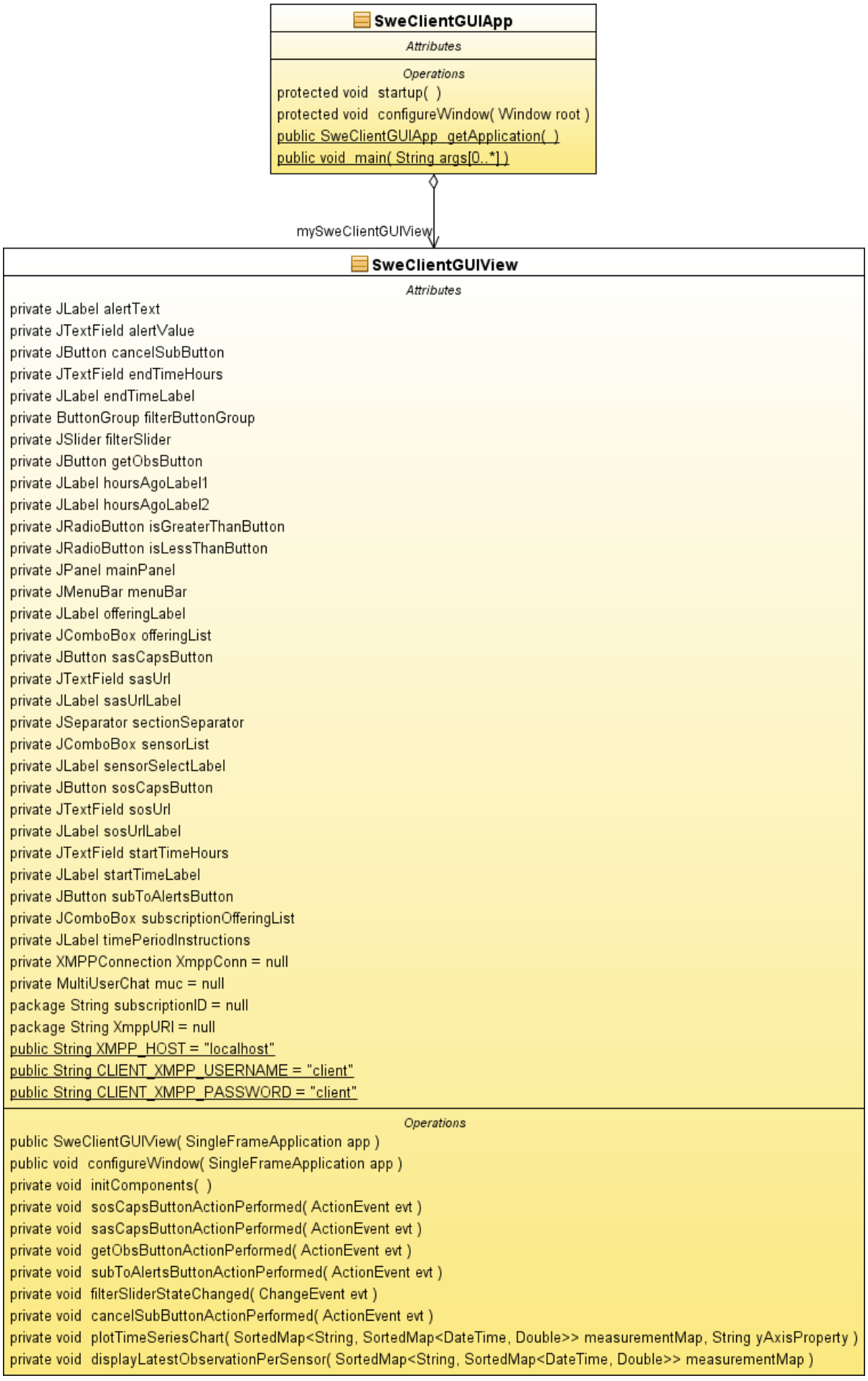
Εικόνα 3: Η διεπαφή χρήστη του SWE Client μετά την υποβολή των ερωτημάτων GetCapabilities

Συγκεκριμένα υπο-τμήματα της διεπαφής που λαμβάνουν τιμή δυναμικά είναι:

- Στο SOS σκέλος:
    - Η λίστα παρατηρούμενων ιδιοτήτων (observed property).
    - Η λίστα διαθέσιμων αισθητήρων.
- Στο SAS σκέλος:
    - Η λίστα παρατηρούμενων ιδιοτήτων (observed property) σε ισχύουσες διαφημίσεις.
    - Το άνω και κάτω όριο της περιοχής τιμών της παρατηρούμενης ιδιότητας,

Να σημειωθεί ότι η λίστα παρατηρούμενων ιδιοτήτων στη συγκεκριμένη περίπτωση περιέχει μόνο την επιλογή της θερμοκρασίας, αλλά εφόσον υπήρχαν και άλλες στο GetCapabilitiesResponse θα εισάγονταν και αυτές.

Το διάγραμμα κλάσεων της εφαρμογής του client φαίνεται στην επόμενη σελίδα. Περιλαμβάνει την κλάση SweClientGUIApp που περιέχει την main() συνάρτηση της εφαρμογής, και την κλάση SweClientGUIView, η οποία ενσωματώνει τη γραφική διεπαφή και την κύρια λειτουργικότητα:

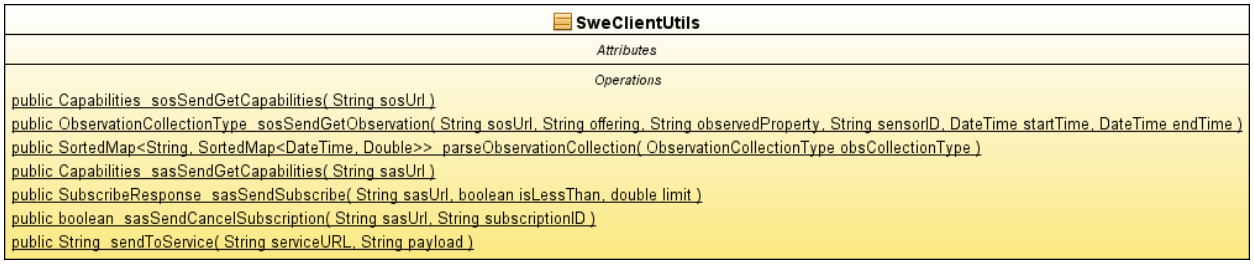


Σχήμα 13: Διάγραμμα κλάσης της γραφικής διεπαφής του SweClient

Ο constructor SweClientGUIView() και οι μέθοδοι initComponents() και configureWindow() της κλάσης SweClientGUIView καλούνται κατά την εκκίνηση της εφαρμογής και αρχικοποιούν όλα τα συστατικά του παραθύρου.

Από τις υπόλοιπες μεθόδους, η πλειοψηφία αφορά συναρτήσεις που χειρίζονται (handlers) τα συμβάντα του παραθύρου, με άλλα λόγια το πάτημα κουμπιών από το χρήστη. Η κύρια λειτουργικότητα της εφαρμογής θα εξηγηθεί παρακάτω. Οι μέθοδοι plotTimeSeriesChart() και displayLatestObservationPerSensor() χρησιμεύουν για την εμφάνιση στον χρήστη των αποτελεσμάτων της επερώτησης για παρατηρήσεις.

Η βοηθητική κλάση SweClientUtils περιέχει μεθόδους για αποστολή αιτήσεων στο SOS και SAS για πελάτες SWE υπηρεσιών. Το διάγραμμα κλάσης της είναι:



Σχήμα 14: Διάγραμμα κλάσης για τη βιβλιοθήκη SweClientUtils

Οι μέθοδοι αυτές, όπως προκύπτει και από τα ονόματά τους, αποστέλλουν στις υπηρεσίες τις αιτήσεις GetCapabilities, GetObservation, Subscribe και CancelSubscription.

Η μέθοδος sendToService() επιτελεί την ίδια λειτουργία όπως και στην κλάση SweProducerUtils που αναφέραμε στην εφαρμογή του Connector, στέλνει δηλαδή το ωφέλιμο μήνυμα (payload) στην υπηρεσία που ζητείται (serviceURL) και επιστρέφει το κείμενο της HTTP απάντησης. Χρησιμοποιείται από όλες τις άλλες μεθόδους της κλάσης.

Η μέθοδος parseObservationCollention() χρησιμεύει για τη μετατροπή μιας συλλογής μετρήσεων σε XML, όπως αυτές επιστρέφονται μέσα στο GetObservationResponse, σε ένα ταξινομημένο Java Map (SortedMap) για χρήση σε άλλες συναρτήσεις, π.χ. στην plotTimeSeriesChart() της κλάσης SweClientGUIView.

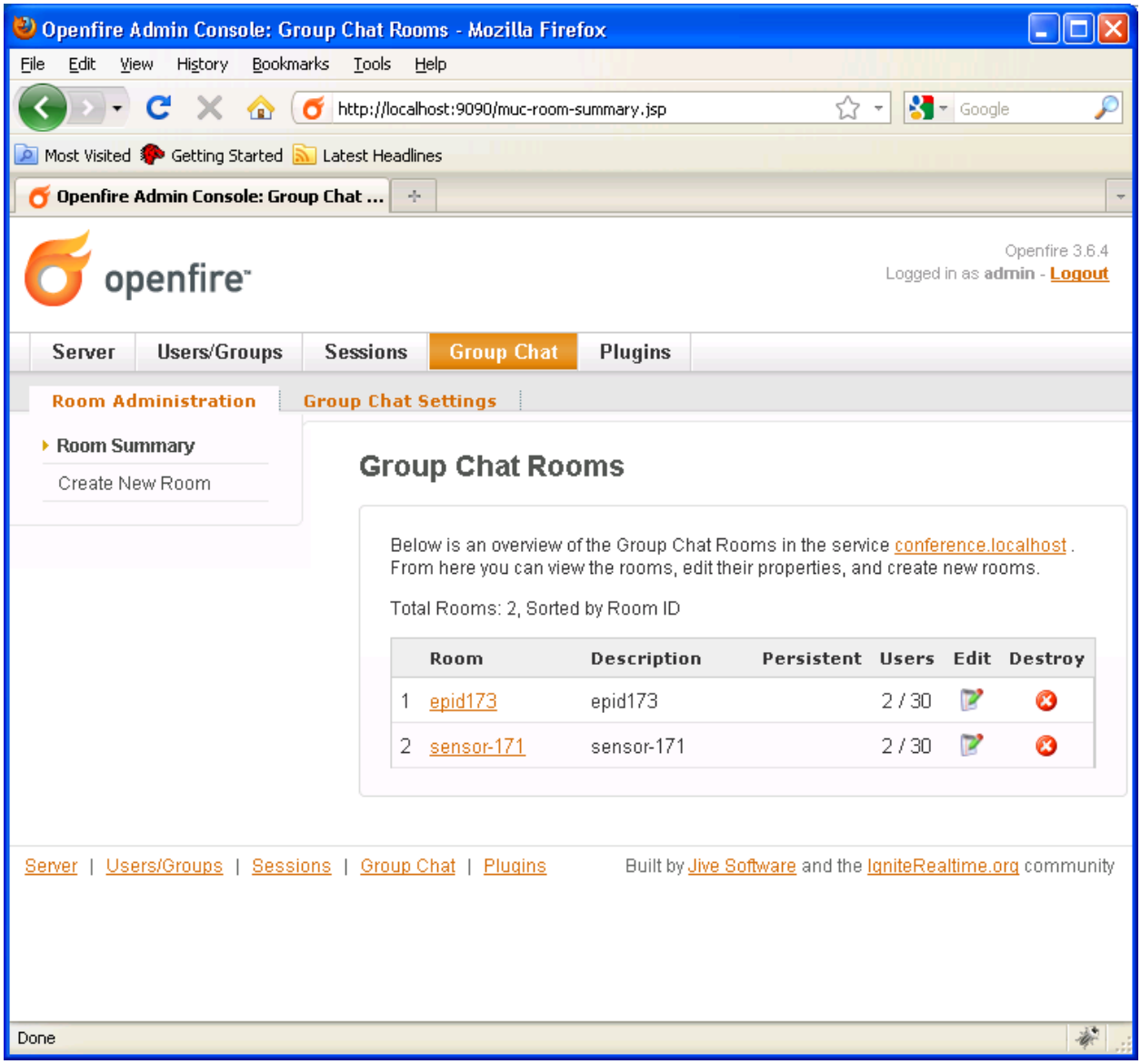
Οι λειτουργικότητα που προσφέρεται από την εφαρμογή στον τελικό χρήστη είναι:

- SOS Τμήμα:
    - Με το πάτημα του κουμπιού “Get SOS Capabilities” η εφαρμογή αποστέλλει μια αίτηση GetCapabilities προς το SOS, υπό την προϋπόθεση ότι το πεδίο “SOS URL” είναι συμπληρωμένο, και αποκωδικοποιεί την απάντηση GetObservationResponse προκειμένου να γεμίσει τη λίστα των διαθέσιμων παρατηρούμενων ιδιοτήτων και αισθητήρων. Μετά τη ολοκλήρωση της λειτουργίας αυτής όλα τα προηγουμένως ανενεργά συστατικά του SOS τμήματος ενεργοποιούνται.
    - Με το κουμπί “Get Observations” υποβάλλεται μια αίτηση GetObservation προς το SOS. Ο χρήστης μπορεί να προσδιορίσει τις ακόλουθες παραμέτρους της αίτησης:
        - Παρατηρούμενη ιδιότητα για την οποία πρέπει να ανακτηθούν μετρήσεις.
        - Χρονικό διάστημα στο οποίο θα κυμαίνεται ο χρόνος λήψης των μετρήσεων. Ο χρήστης πρέπει να ορίσει το σημείο έναρξης και λήξης μετρώμενο σε ώρες που έχουν περάσει από την τρέχουσα στιγμή. Αν δεν οριστεί χρονικό διάστημα, η επερώτηση θα επιστρέψει μόνο την τελευταία μέτρηση, η οποία θα παρουσιαστεί στο χρήστη μέσω της μεθόδου displayLatestObservationPerSensor(). Ειδάλλως, θα χρησιμοποιηθεί η plotTimeSeriesChart() για το σχεδιασμό της γραφικής παράστασης των μετρήσεων ως προς το χρόνο μέσα σε ένα νέο παράθυρο.
        - Αισθητήρας του οποίου πρέπει να ανακτηθούν οι μετρήσεις. Ο χρήστης μπορεί να επιλέξει από τη λίστα ή να αφήσει την επιλογή “All”, που θα του επιστρέψει μετρήσεις για όλους τους εγγεγραμμένους αισθητήρες.
- SAS Τμήμα:
    - Με το πάτημα του κουμπιού “Get SAS Capabilities”, αν το πεδίο “SAS URL” είναι συμπληρωμένο, υποβάλλεται αίτηση GetCapabilities στο SAS για να βρεθούν διαφημίσεις (advertisements) βάσει των οποίων μπορεί να εκδοθούν ειδοποιήσεις συναγερμού (alerts). Η συγκεκριμένη υλοποίηση περιορίζεται στο να παρουσιάζει στο χρήστη τις παρατηρούμενες ιδιότητες που περιλαμβάνονται μόνο στην πιο πρόσφατη εν ενεργεία διαφήμιση. Τα όρια των τιμών του σύρτη (slider) για το όριο του κριτηρίου συναγερμού επίσης τίθενται με βάση το GetCapabilitiesResponse.
    - Η χρήση του κουμπιού “Subscribe to Alerts” επιτρέπει στο χρήστη να γίνει συνδρομητής της υπηρεσίας ειδοποιήσεων συναγερμού. Οι παράμετροι που πρέπει να ορίσει είναι:
        - Παρατηρούμενη ιδιότητα (observed property).
        - Τελεστή σύγκρισης για το αποτέλεσμα, «μικρότερο από» ή «μεγαλύτερο από».
        - Όριο συνθήκης, που αντιστοιχεί σε άνω ή κάτω όριο ανάλογα με τον τελεστή σύγκρισης που έχει επιλεχθεί. Η τιμή μπορεί να τεθεί μέσω του σύρτη (slider) ή απευθείας σαν αριθμητική τιμή στο κουτάκι.

- Το κουμπί “Cancel Subscription”, το οποίο γίνεται διαθέσιμο μετά από μια επιτυχή δημιουργία νέας συνδρομής, ακυρώνει τη συνδρομή αυτή. Ο χρήστης μπορεί στη συνέχεια να ζητήσει άλλη συνδρομή με διαφορετικά κριτήρια συναγερμού.

Το GUI διαθέτει ελέγχους εγκυρότητας των παραμέτρων που εισάγει ο χρήστης για τις επερωτήσεις και τα υπόλοιπα αιτήματα που μπορεί να υποβληθούν.

Όπως φαίνεται στο παράθυρο της διαχείρισης του XMPP Server, για το μηχανισμό του alerting:



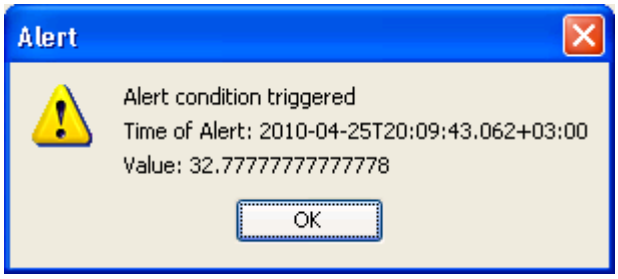
Εικόνα 4: Περιβάλλον διαχείρισης των Group Chat Rooms στον Openfire XMPP server

Έχουν δημιουργηθεί 2 group chat rooms (conferences) από το SAS:

- Μία σύνοδος με το όνομα sensor-171, δημιουργηθείσα από το SAS κατόπιν της αίτησης Advertise του Connector, με 2 συμμετέχοντες: το SAS και τον Connector. Όποια μέτρηση δημοσιεύεται σε αυτό το «δωμάτιο» από τον Connector είναι ορατή και στο SAS.

- Μία σύνοδος με το όνομα epid173, δημιουργηθείσα πάλι από το SAS κατόπιν της αίτησης Subscribe του πελάτη, με 2 συμμετέχοντες, το SAS και τον client. Το SAS κοινοποιεί σε αυτό το «δωμάτιο» ένα συναγερμό SASAlert αν μια μέτρηση που δημοσιευθεί από τον Connector στο πρώτο «δωμάτιο» βρεθεί από το SAS να πληροί τη συνθήκη συναγερμού που έχει οριστεί στο Subscribe. Ο πελάτης λαμβάνει με αυτόν τον τρόπο το μήνυμα συναγερμού.

Στη συγκεκριμένη εφαρμογή πελάτη που αναπτύχθηκε, ένα μήνυμα ειδοποίησης συναγερμού για μέτρηση που ξεπέρασε το όριο των 22 βαθμών Κελσίου φαίνεται στο εξής αναδυόμενο παράθυρο:



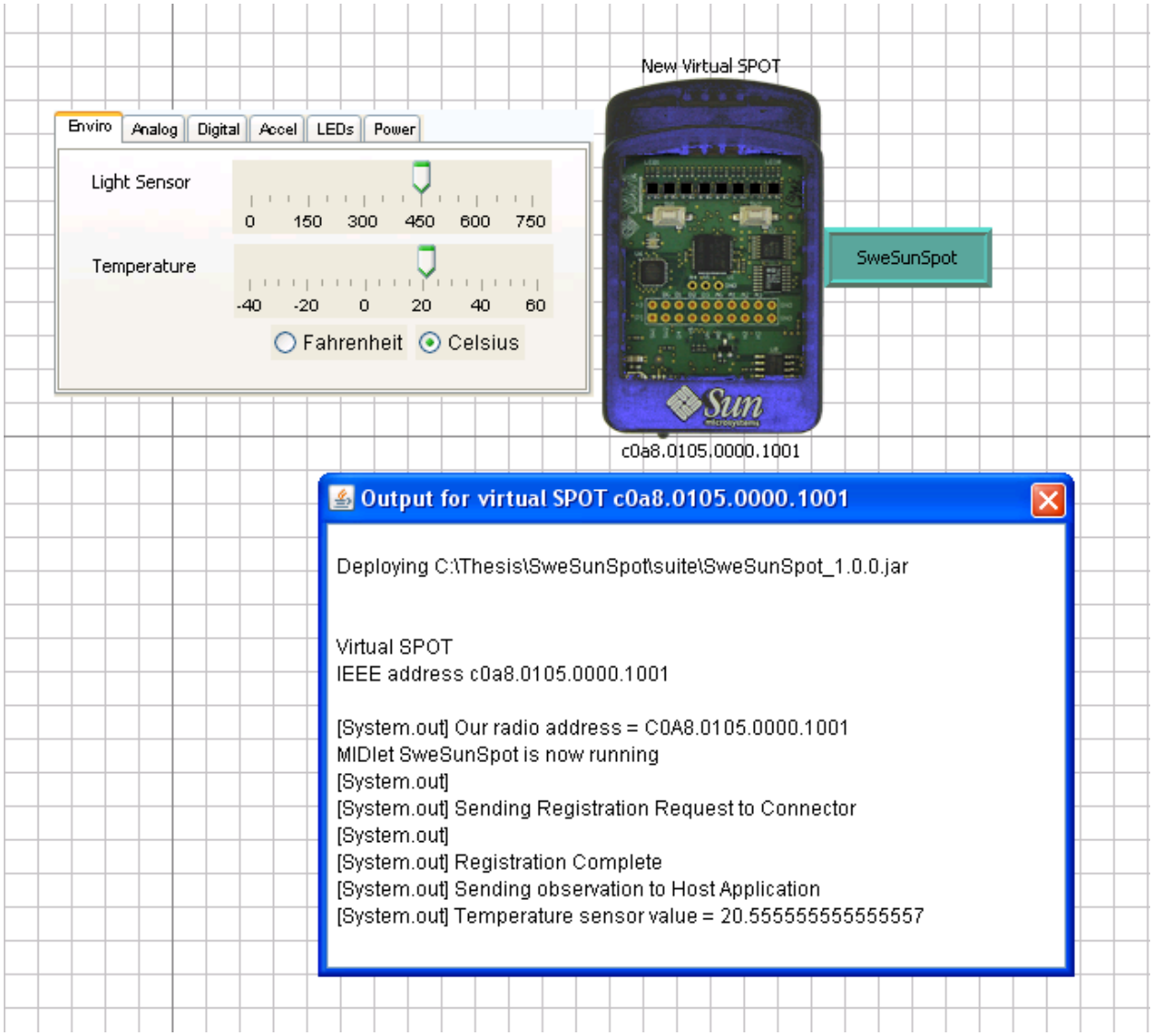
Εικόνα 5: Αναδυόμενο παράθυρο με ειδοποίηση για συνθήκη συναγερμού

Το παράθυρο αυτό δημιουργείται από ένα νήμα τύπου PacketListener, από τη βιβλιοθήκη Smack, το οποίο ενεργοποιείται κάθε φορά που η εφαρμογή του πελάτη δεχτεί XMPP μήνυμα από το conference στο οποίο έχει συνδεθεί.

3.4 Αποτελέσματα και συμπεράσματα

3.4.1 Προσομοίωση

Στην εικόνα που ακολουθεί φαίνεται η φόρτωση και ενεργοποίηση του Midlet σε ένα εικονικό SunSPOT στο περιβάλλον προσομοίωσης Solarium της Sun:

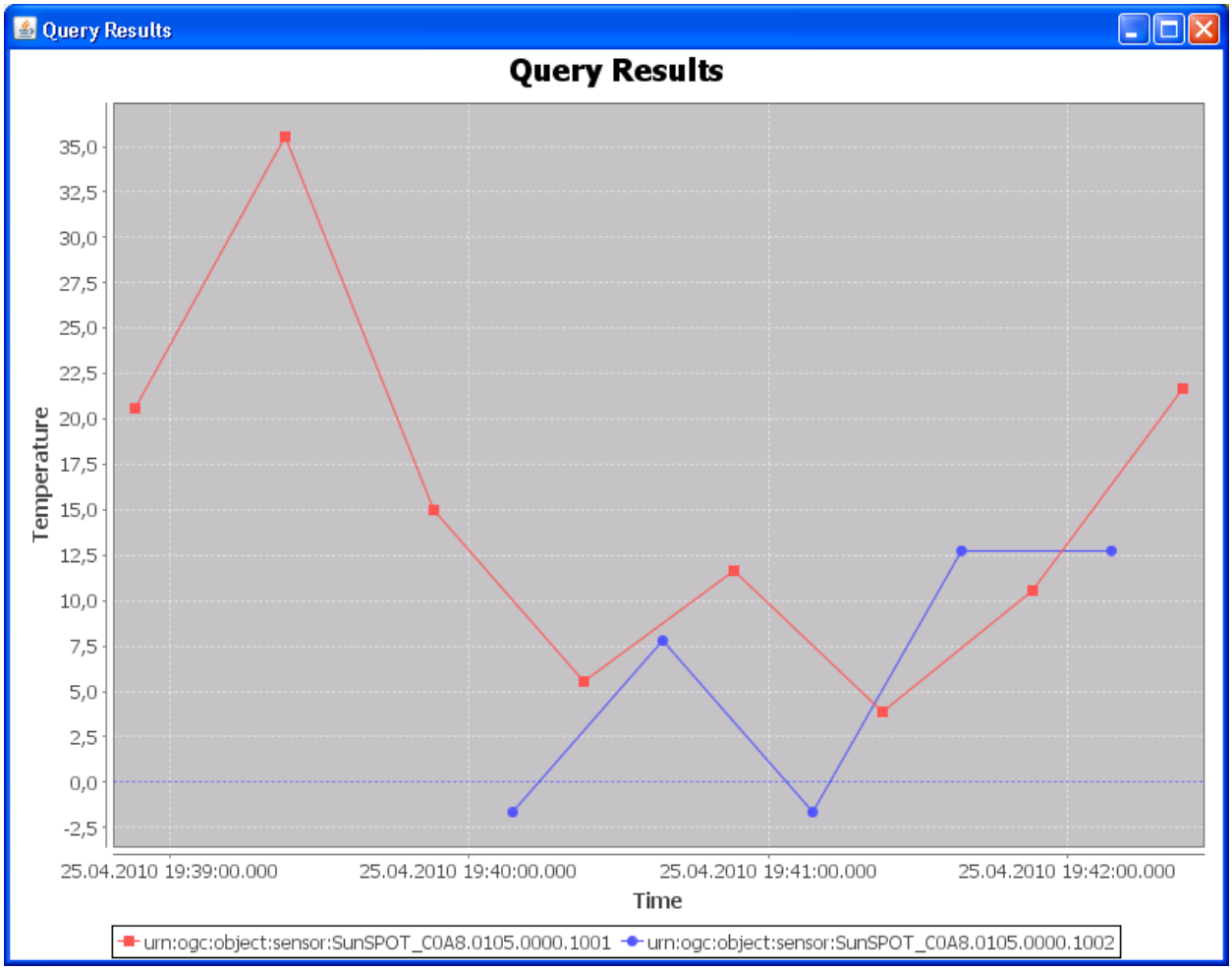


Εικόνα 6: Περιβάλλον προσομοίωσης SunSPOT Solarium

Φαίνεται στο πράσινο πλαίσιο το Midlet που εκτελείται. Στο κάτω λευκό παράθυρο εκτυπώνεται η πρότυπη έξοδος (standard output) της εκτελούμενης εφαρμογής, ενώ αριστερά μέσω του πλαισίου αισθητήρων (sensor panel) ο χρήστης μπορεί να θέσει ο ίδιος τις επιθυμητές τιμές που ανιχνεύουν οι αισθητήρες του εικονικού SunSPOT.

Όπως αναφέρει η έξοδος της εφαρμογής, το εικονικό SPOT με IEEE διεύθυνση C0A8.01.05.0000.1001 εκτελεί το Midlet SweSunSpot. Η εφαρμογή έχει στείλει αίτηση εγγραφής στον Connector (Registration Request) και η απάντητη ήταν θετική (Registration Complete), συνεπώς το SunSPOT έχει εισέλθει στο βρόχο συλλογής μετρήσεων από τον αισθητήρα θερμοκρασίας.

Μια γραφική παράσταση των μετρήσεων θερμοκρασίας που ελήφθησαν στις τελευταίες 8 ώρες (δηλαδή ώρα έναρξης 8 ώρες πριν και ώρα λήξης 0 ώρες πριν, ως παράμετροι στη διεπαφή του πελάτη) από όλους τους αισθητήρες (‘All’), όπως αυτή κατασκευάζεται από την εφαρμογή του πελάτη, φαίνεται στο ακόλουθο σχήμα:



Εικόνα 7: Γραφική απεικόνιση των αποτελεσμάτων μιας αίτησης GetObservation από προσομοίωση

Όπως βλέπουμε, η δοκιμή είναι επιτυχής. Τα δεδομένα από τους αισθητήρες έχουν παρουσιαστεί στην οθόνη, έχοντας μεταδοθεί μέσω του Connector στο SOS και στη συνέχεια μέσω αίτησης GetObservation στην εφαρμογή του πελάτη. Τα εικονικά SunSPOT μας δίνουν μεγάλη ελευθερία στην επιλογή των τιμών θερμοκρασίας, οπότε μπορούμε ελεύθερα να προκαλλούμε εικονικές μεταπτώσεις της θερμοκρασίας από τους 35 στους 4 βαθμούς Κελσίου και το ανάποδο, κάτι που σε συνθήκες πραγματικού κόσμου δε θα ήταν εύκολα επιτεύξιμο.

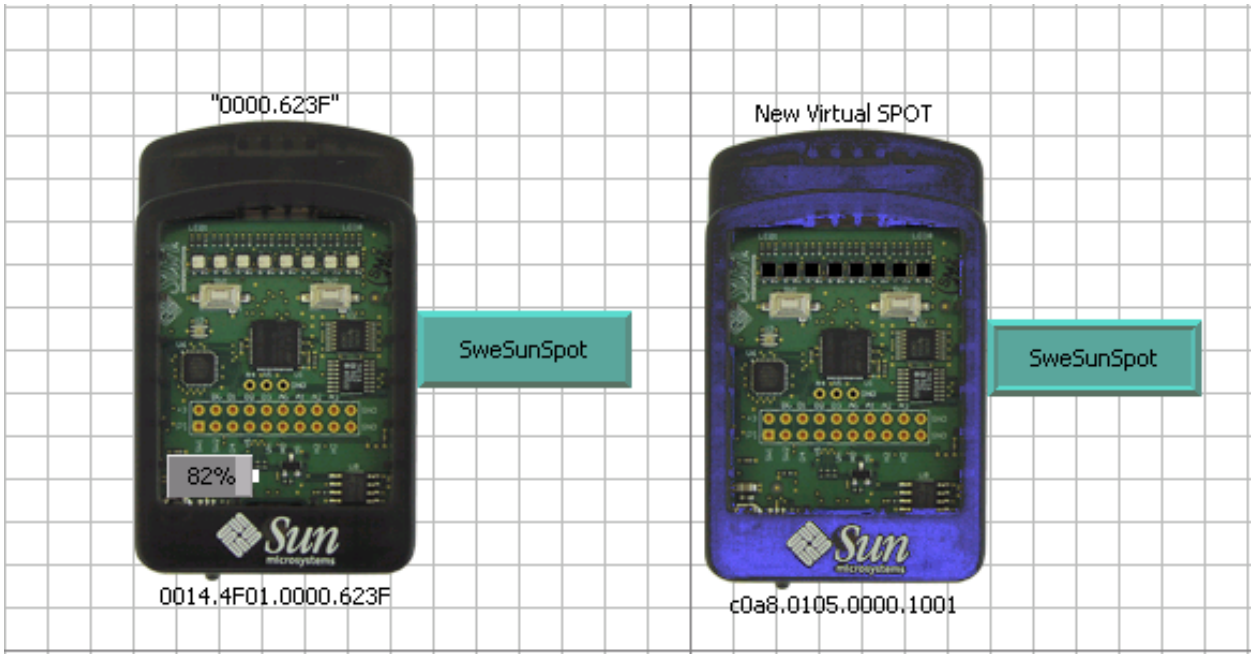
3.4.2 Με πραγματικούς αισθητήρες SunSPOT

Με πραγματικούς αισθητήρες SunsPOT προφανώς δεν είναι πρακτικό να υποβάλουμε τους κόμβους σε παρόμοιες αυξομοιώσεις θερμοκρασίας με αυτές που δοκιμάσαμε κατά την προσομοίωση. Συνθήκες -30 ή 55 βαθμών Κελσίου δεν είναι εύκολο να βρεθούν σε φυσιολογικά δωμάτια, ενώ για την καλή λειτουργία των SunSPOTs, ακραίες

θερμοκρασίες δεν είναι και θεμιτές. Αναμένονται αρκετά στενά εύρη στις διακυμάνσεις των γραφικών παραστάσεων.

Να σημειωθεί ότι ο αισθητήρας θερμοκρασίας του SunSPOT δε μετράει επακριβώς τη θερμοκρασία του εξωτερικού περιβάλλοντος, αλλά τη θερμοκρασία στα κυκλώματα του αναλογικοψηφιακού μετατροπέα (ADC) της πλακέτας (sensor board) με την οποία είναι εφοδιασμένο. Επομένως αποκλίσεις από την πραγματική τιμή είναι βέβαιες. Μπορούμε όμως να δεχτούμε σα σύμβαση ότι οι εξωτερικές μεταβολές στο περιβάλλον επηρεάζουν και τις τιμές που αυτό καταγράφει κατά τρόπο αρκετά συνεπή.

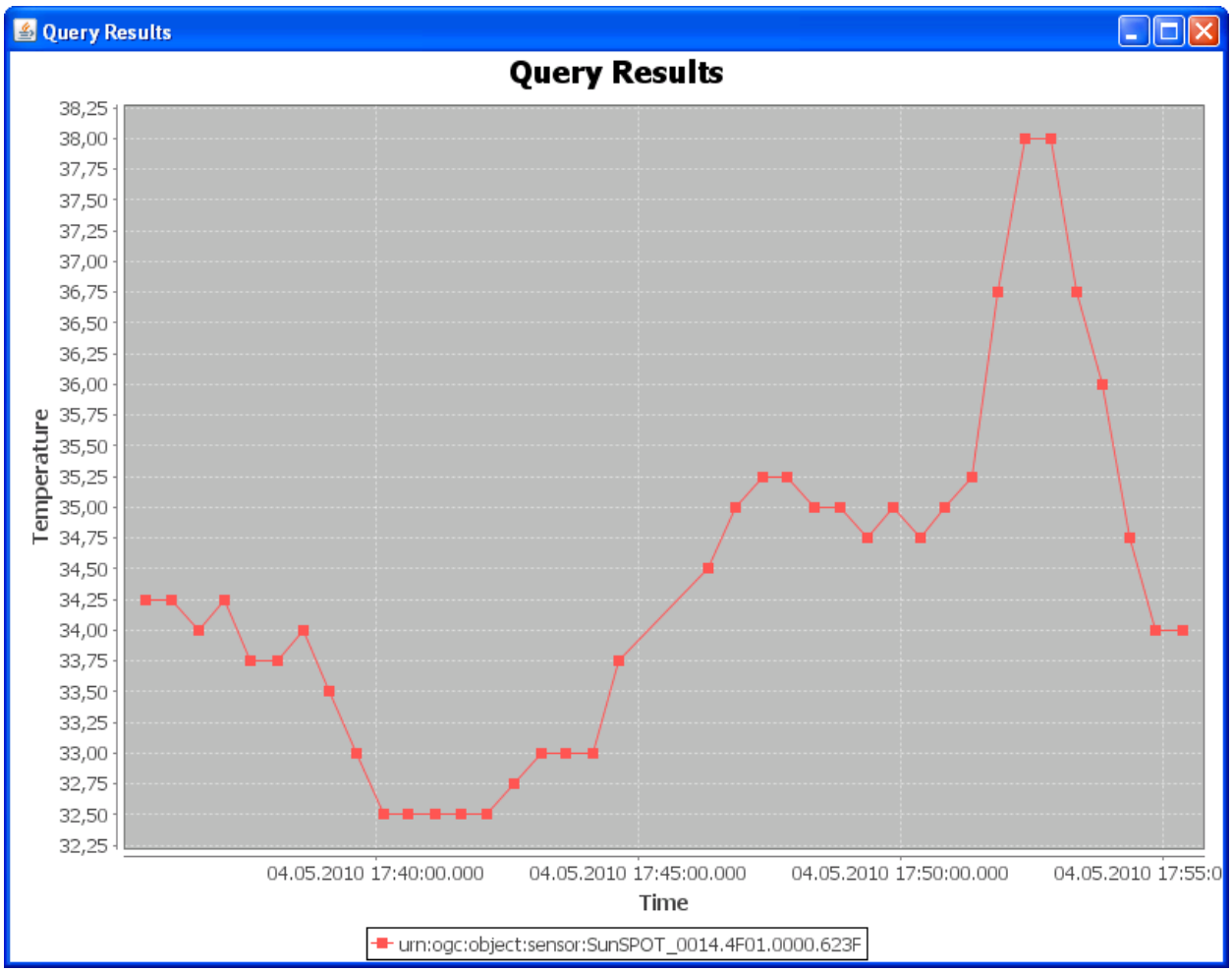
Το Solarium μας επιτρέπει να διαχειριζόμαστε και πραγματικούς κόμβους. Στο παρακάτω σχήμα φαίνεται στο περιβάλλον του Solarium ένα πραγματικό (physical) και ένα εικονικό (virtual) SunSPOT. Το μαύρο αντιστοιχεί στον πραγματικό κόμβο. Και οι δύο κόμβοι παρουσιάζονται να εκτελούν το SweSunSpot Midlet.



Εικόνα 8: Πραγματικό και εικονικό SunSPOT που εκτελούν το SweSunSpot στο περιβάλλον Solarium

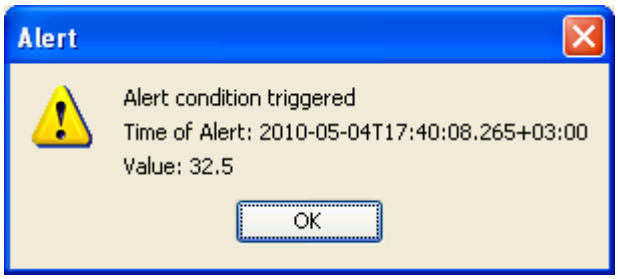
Αξίζει να σημειώσουμε ότι οι εφαρμογές που συνεργάζονται με SunSPOT δεν «ενδιαφέρονται» για το αν ένα SPOT είναι πραγματικό ή όχι. Μπορούμε να εργαστούμε και με μικτή σύνθεση του πεδίου αισθητήρων.

Η δοκιμή με πραγματικούς αισθητήρες SunSPOT στέφθηκε επίσης με επιτυχία, τόσο στο σκέλος του Data-Pull όσο και του Data-Push. Όπως ήταν αναμενόμενο η μετρούμενη θερμοκρασία δεν αντιπροσώπευε απόλυτα τη θερμοκρασία δωματίου, καθώς η θερμοκρασία του ADC ήταν αρκετούς βαθμούς μεγαλύτερη. Στην παρακάτω δοκιμή, οι τιμές κυμάνθηκαν μεταξύ των 32.5°C και 38°C:



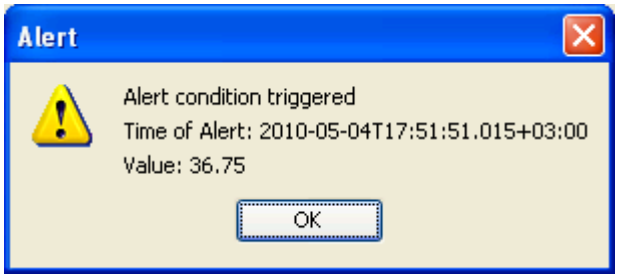
Εικόνα 9: Γραφική απεικόνιση των αποτελεσμάτων μιας αίτησης GetObservation από πραγματικά SunSPOTs

Μεταξύ των λεπτών 17:40 και 17:42 η θερμοκρασία έπεσε στην ελάχιστη τιμή με τη βοήθεια ενός ανεμιστήρα που έψυξε τον αισθητήρα. Για κριτήριο συναγερμού τιμές κάτω από 33°C, η εφαρμογή του πελάτη ειδοποίησε το χρήστη με ένα μήνυμα σαν το ακόλουθο:



Εικόνα 10: Συνθήκη συναγερμού για πτώση θερμοκρασίας κάτω από το όριο

Μεταξύ των λεπτών 17:51 και 17:52, θερμάναμε εξωτερικά τον κόμβο πλησιάζοντας τον σε ένα αναμμένο κερί. Για κριτήριο συναγερμού την υπέρβαση των 36°C, το μήνυμα ειδοποίησης είχε τη μορφή:



Εικόνα 11: Συνθήκη συναγερμού για υπέρβαση του ορίου θερμοκρασίας

Μετά την απομάκρυνση από την εστία θερμότητας η θερμοκρασία ξαναμειώθηκε φυσικά τείνοντας προς την αρχική τιμή των 34°C.

3.5 Περιορισμοί της τρέχουσας υλοποίησης

Όπως επισημάναμε και αρχικά, η τρέχουσα υλοποίηση έχει διερευνητικό χαρακτήρα και σκοπεύει στη δημιουργία ενός στοιχειώδους λειτουργικού παραδείγματος σαν απόδειξη εφικτότητας του σκεπτικού του SWE (proof of concept). Ως εκ τούτου έχουν γίνει πολυάριθμες παραχωρήσεις και απλοποιήσεις, και έχουν τεθεί περιορισμοί ώστε η υλοποίηση να μην έχει υπερβολικό εύρος και να είναι επιτεύξιμη σε εύλογο χρονικό διάστημα.

Ο παραγωγός του σεναρίου μας μπορεί να έχει μία μόνο ενεργή διαφήμιση (advertisement), η οποία αποστέλλεται στο SAS κατά την εκκίνηση του Connector. Παρομοίως, ο πελάτης μπορεί να έχει μόνο μία ισχύουσα συνδρομή (subscription), αν και μπορεί να την ακυρώσει και να αιτηθεί μιας καινούριας, με νέα συνθήκη συναγερμού. Σίγουρα δε διερευνήθηκαν όλες οι υποστηριζόμενες λειτουργίες του SOS και SAS, ούτε όλες οι παράμετροι αυτών, παρά επιλέχθηκε ένα αντιπροσωπευτικό υποσύνολο. Όπως επισημάνθηκε και σε προηγούμενη ενότητα, από τις δυνατότητες του SunSPOT αξιοποιήθηκε μόνο ο αισθητήρας θερμοκρασίας. Επιπλέον, δεν λήφθησαν υπόψη χωρικές παράμετροι όπως η θέση των αισθητήρων.

Οι συναρτήσεις που αποστέλλουν τις αιτήσεις SWE δεν είναι πλήρως παραμετροποιημένες, με άλλα λόγια υπάρχουν αρκετές προκαθορισμένες (hard-coded) τιμές πεδίων στις αιτήσεις, που αφορούν στη συγκεκριμένη εφαρμογή ενός συστήματος παρακολούθησης θερμοκρασίας. Εντούτοις, έχει δοθεί έμφαση σε παραμετροποίηση όπου αυτό έχει αισθητό αποτέλεσμα σε μια εξωτερική διεπαφή, π.χ. τη διεπαφή του χρήστη. Μια πλήρως γενική λύση, για παράδειγμα ένα Java API για αιτήσεις SWE, θα όφειλε να είναι εντελώς παραμετροποιήσιμο, πιθανότατα με σχεδιασμό των συναρτήσεων ώστε να δέχονται ένα αντικείμενο παραμετροποίησης (configuration object), αντί για τεράστιο αριθμό δυνατών μεμονωμένων παραμέτρων.

Προγραμματιστικά, ο βαθμός παραλληλίας έχει πολλά περιθώρια για βελτίωση. Για τη βελτίωση της εμπειρίας του τελικού χρήστη (end-user experience) απαιτείται υψηλή αποκρισιμότητα και έλλειψη εμπλοκής στη γραφική διεπαφή (non-blocking GUI), η οποία επιτυχάνεται με αυξημένη αξιοποίηση νημάτων, εποπτικές ενδείξεις προόδου (progress bar), κ.ά. Βελτιώσεις επιδέχονται επίσης και ο Connector και η εφαρμογή που εκτελείται στο SunSPOT, ώστε να περιορίζονται οι καθυστερήσεις λόγω ενεργειών που δεν είναι απαραίτητο να εκτελούνται σειριακά. Βέβαια, αυτό εισάγει την αυξημένη μέριμνα αποφυγής συγκρούσεων (race conditions), πρόληψης αδιεξόδων (deadlocks) κτλ. Όλες οι εφαρμογές επίσης επιδέχονται βελτιώσεις για αυξημένη ευρωστεία, όπως υλοποίηση μηχανισμών επαναμετάδοσης αιτήσεων αν οι αρχικές αποτύχουν, υποστήριξη παύσης και επανεκκίνησης λειτουργίας, κ.ά.

Τέλος, στο βωμό της απλότητας, οι υλοποιήσεις των συστατικών που εξετάσαμε παραπάνω χαρακτηρίζονται από υποβέλτιστη αξιοποίηση πόρων. Οι εφαρμογές του Connector και του πελάτη χρησιμοποιούν νέα σύνδεση HTTP για κάθε αίτηση αντί για την επαναχρησιμοποίηση μιας κοινής σύνδεσης για όλη τη διάρκεια ζωής της διεργασίας. Η χρήση μοναδικών HTTP συνδέσεων βέβαια θα συνεπάγεται την ανάγκη διαχωρισμού των απαντήσεων στις HTTP αιτήσεις, θέμα το οποίο δεν υφίσταται στην τρέχουσα υλοποίηση καθώς κάθε νήμα χρησιμοποιεί τη δική του HTTP σύνδεση, οπότε δεν μπορεί να υπάρξει σύγχυση ως προς ποια απάντηση αντιστοιχεί σε ποιο αίτημα. Πρόσβαση σε κοινόχρηστες HTTP συνδέσεις προς το SOS και SAS επιπλέον θα πρέπει να συγχρονίζεται μέσω σημαφόρων ή αλλών μεθόδων συγχρονισμού, για να μην υπάρξει αλλοίωση (corruption) της ροής δεδομένων.

4. ΚΑΤΕΥΘΥΝΣΕΙΣ ΓΙΑ ΤΟ ΜΕΛΛΟΝ ΤΟΥ SWE

4.1 Υλοποιήσεις από περισσότερους φορείς

Πέρα από τις ακαδημαϊκής χρήσης υλοποιήσεις της 52°North, όπως ήδη αναφέραμε έχουν αρχίσει να εμφανίζονται υλοποιήσεις από κυβερνητικούς και ιδιωτικούς φορείς. Αυτή η τάση αναμένεται να ενισχυθεί στο μέλλον καθώς τα πρότυπα του SWE θα ωριμάζουν περαιτέρω και η ανάγκη για υλοποιήσεις με αξιοπιστία επιπέδου παραγωγής (production / industrial strength) θα γίνεται πιο έκδηλη.

4.2 Διαλειτουργικότητα με σημαντικά νέα πρωτόκολλα

Οι υπηρεσίες του SWE αναμένεται να χρησιμεύσουν ως στρώμα αφαίρεσης για τη διάδραση με πολλά υπάρχοντα και αναπτυσσόμενα πρωτόκολλα και πρότυπα. Ένα εξ αυτών που παρουσίαζει ιδιαίτερο ενδιαφέρον είναι η σουίτα προτύπων IEEE 1451 [15] για Διεπαφές Έξυπνων Μορφοτροπέων για Αισθητήρες και Ενεργοποιητές (Standard for a Smart Transducer Interface for Sensors and Actuators). Η ομάδα προτύπων αυτή αποσκοπεί στην αυστηρή αυτοπεριγραφή (self-description) συσκευών που υπάγονται στην κατηγορία των μορφοτροπέων, μέσω ενός ηλεκτρονικού «φύλλου δεδομένων» ονόματι TEDS (Transducer Electronics Data Sheet), καθώς και στον ορισμό διεπαφών με εξωτερικές διατάξεις μέσω ενσύρματων ή ασύρματων συνδέσεων.

Σε ένα σύστημα που συνδυάζει SWE και IEEE 1451 το TEDS ενός αισθητήρα θα πρέπει να μεταφράζεται κατά μονοσήμαντο τρόπο σε μεταδεδομένα μιας RegisterSensor αίτησης.

4.3 SES σε αντικατάσταση του SAS

Όπως αναφέρθηκε και νωρίτερα, οι προδιαγραφές του SAS δεν έφτασαν σε πλήρες στάδιο περάτωσης (έκδοση 0.9.0) διότι εντοπίστηκαν εγγενείς αδυναμίες που οδήγησαν στην ανακατεύθυνση του ενδιαφέροντος προς τον αντικαταστάτη του.

Η βελτιωμένη αυτή μορφή υπηρεσίας ιστού για υποστήριξη PUSH-based υπηρεσιών ονομάζεται υπηρεσία συμβάντων αισθητήρων SES (Sensor Event Service). Διαθέτει αυξημένες δυνατότητες φιλτραρίσματος (filtering) των αποτελεσμάτων και στηρίζεται σε ένα νέο μοντέλο επεξεργασίας βασισμένο σε συμβάντα (event – based) για τα οποία έχει αναπτυχθεί και μια ειδική γλώσσα σήμανσης, η γλώσσα σήμανσης προτύπων συμβάντων EML (Event Pattern Markup Language) [16].

Η πρότυπη υλοποίηση του SES από τη 52°North δεν αξιοποιήθηκε στην τρέχουσα εργασία, παρά την αυξημένο ενδιαφέρον που τη χαρακτηρίζει σε σχέση με το απερχόμενο SAS, καθώς η δημοσιευμένη τεκμηρίωση (documentation) κατά τη χρονική περίοδο εκπόνησης δεν ήταν επαρκής για τον πειραματισμό με αυτήν.

Οι κύριες διαφορές μεταξύ του SAS και του SES συνοψίζονται στον παρακάτω πίνακα [17]:

Πίνακας 1: Σύγκριση των δυνατοτήτων του SAS και του SES

Λειτουργικότητα	SAS	SES
χωρικό φιλτράρισμα	υποστηρίζεται μερικώς	υποστηρίζεται
χρονικό φιλτράρισμα	δεν υποστηρίζεται	υποστηρίζεται
φίλτρα σύγκρισης	υποστηρίζεται μερικώς	υποστηρίζεται
σωρευτικά αποτελέσματα	δεν υποστηρίζεται	υποστηρίζεται
θέματα (topics)	δεν υποστηρίζεται	υποστηρίζεται
μετατροπή μονάδων	δεν υποστηρίζεται	υποστηρίζεται
σύνθετη επεξεργασία συμβάντων	δεν υποστηρίζεται	υποστηρίζεται
επεξεργασία ροής συμβάντων	δεν υποστηρίζεται	υποστηρίζεται

ΕΠΙΛΟΓΟΣ

Στα πλαίσια της παρούσας εργασίας εξοικειωθήκαμε με την έννοια του Sensor Web Enablement και δη με δύο σημαντικές υπηρεσίες του, την Υπηρεσία Παρατηρήσεων Αισθητήρων (SOS – Sensor Observation Service) και την Υπηρεσία Συναγερμών Αισθητήρων (SAS – Sensor Alert Service), οι οποίες προσφέρουν Pull και Push-based λειτουργικότητα, αντίστοιχα.

Αφού αναλύσαμε τις βασικές αιτήσεις που προβλέπονται από την OGC για τις δύο αυτές υπηρεσίες με τη βοήθεια των σχετικών διαγραμμάτων και παραδειγμάτων σε XML, είδαμε και μια πρακτική εφαρμογή: την παρακολούθηση της θερμοκρασίας μέσα σε ένα χώρο φύλαξης αγαθών. Η υλοποίηση έκανε χρήση λογισμικού ανοιχτού κώδικα, συμπεριλαμβανομένων των πρότυπων υλοποιήσεων SOS και SAS της 52°North και της πλατφόρμας προγραμματιζόμενων αισθητήριων κόμβων SunSPOT της Sun Microsystems.

Οι επιτυχημένες δοκιμές, τόσο με προσομοίωση όσο και με πραγματικούς αισθητήρες, μας δείχνει ότι τα πρωτόκολλα του SWE αποτελούν μια πρακτική και βιώσιμη λύση για ανάπτυξη εφαρμογών για το Sensor Web. Αναβάθμιση των πρωτοκόλλων για αντιμετώπιση ορισμένων αδυναμιών που επισημάναμε θα δώσει στο Sensor Web Enablement αυξημένη χρησιμότητα. Η καθιέρωση ανοιχτών προτύπων για τη διαλειτουργικότητα των αισθητήρων με τον Παγκόσμιο Ιστό θα αποτελέσει ένα σημαντικό βήμα για την απομάκρυνση από εξατομικευμένες (proprietary) λύσεις και θα δώσει το έναυσμα για αυξημένη παραγωγικότητα και συνεργασία στο χώρο των Γεωγραφικών Συστημάτων Πληροφόρησης (Geographical Information Systems) και της παροχής Βασισμένων στην Τοποθεσία Υπηρεσιών (LBS – Location Based Systems).

ΠΙΝΑΚΑΣ ΟΡΟΛΟΓΙΑΣ

Ξενόγλωσσος Όρος	**Ελληνικός Όρος**
smart sensor	έξυπνος αισθητήρας
micronization	σμίκρυνση
Internet of things	Internet των πραγμάτων
address space	χώρος διευθύνσεων
case study	μελέτη περίπτωσης
query	επερώτηση
markup language	γλώσσα σήμανσης
node	κόμβος
self-description	αυτοπεριγραφή
discovery	ανακάλυψη
tasking	εντολοδότηση
feasibility	επιτευξιμότητα
subscription	συνδρομή
publication	δημοσίευση
alert	συναγερμός
notification	ειδοποίηση
interoperability	διαλειτουργικότητα
network interface	διεπαφή δικτύου
observation	παρατήρηση
observation repository	αποθήκη δεδομένων παρατήρησης
best practice	σύσταση
location	θέση / τοποθεσία
scalar	βαθμωτό
measurement	μέτρηση
feature of interest	γεωγραφικό γνώρισμα ενδιαφέροντος
spatial database	χωρική βάση δεδομένων
observation offering	προσφορά παρατηρήσεων
observed property	παρατηρούμενη ιδιότητα
actuator	ενεργοποιητής
mobile	κινητός
in situ	στατικός
transducer	μορφοτροπέας

aerial interface	εναέρια διεπαφή
path loss	εξασθένηση
sink	συγκεντρωτής
routing	δρομολόγηση
information dissemination	διασπορά της πληροφορίας
power conservation	διατήρηση ενέργειας
metadata	μεταδεδομένα
request – response	αίτηση – απάντηση
exception report	αναφορά λάθους / εξαίρεσης
procedure	διαδικασία
sampling time	χρονική στιγμή δειγματοληψίας
instance	στιγμιότυπο
service identification	ταυτοποίηση υπηρεσίας
service provider	πάροχος υπηρεσίας
filter capabilities	δυνατότητες φιλτραρίσματος
conference	σύσκεψη
instant messages	άμεσα μηνύματα
polling	συνεχής (τακτή) μέτρηση / δειγματοληψία
reusability	επαναχρησιμοποίηση
integration	ολοκλήρωση
registry	μητρώο
scalability	επεκτασιμότητα
deployment	ανάπτυξη
base station	σταθμός βάσης
sensor field	πεδίο αισθητήρων
syntax highlighting	αυτόματη σήμανση κώδικα
code completion	αυτόματη συμπλήρωση κώδικα
debugging	αποσφαλμάτωση
class diagram	διάγραμμα κλάσης
granularity	βαθμός πιστότητας
payload	ωφέλιμο φορτίο μήνυματος
end-user experience	εμπειρία τελικού χρήστη
race condition	σύγκρουση
deadlock	αδιέξοδο
non-blocking	χωρίς εμπλοκή, ασύγχρονο

ΣΥΝΤΜΗΣΕΙΣ – ΑΡΚΤΙΚΟΛΕΞΑ – ΑΚΡΩΝΥΜΙΑ

OGC	Open Geospacial Consortium
GIS	Geographical Information Systems
LBS	Location Based Services
SWE	Sensor Web Enablement
HTTP	HyperText Transfer Protocol
SOAP	Simple Object Access Protocol
XML	Extensible Markup Language
GML	Geography Markup Language
O&M	Observations & Measurements
SensorML	Sensor Model Language
TML (TransducerML)	Transducer Markup Language
SOS	Sensor Observation Service
SOS-T	SOS-Transactional Profile
SAS	Sensor Alert Service
SES	Sensor Event Service
SPS	Sensor Planning Service
WNS	Web Notification Service
UDDI	Universal Description Discovery and Integration
IM	Instant Messaging
XMPP	Extensible Messaging and Presence Protocol
SMS	Short Messaging Service
URI	Uniform Resource Identifier
URN	Uniform Resource Name
SunSPOT	Sun Small Programmable Object Technology
CLDC	Connected Limited Device Configuration
MIDP	Mobile Information Device Profile
IMP	Information Module Profile
AODV	Ad hoc On-Demand Distance Vector
LQRP	Link Quality Routing Protocol
VM	Virtual Machine
USB	Universal Serial Bus

ADC	Analog to Digital Converter
DAC	Digital to Analog Converter
GPL 2.0	GNU Public License 2
RDBMS	Relational Data Base Management System
JDK	Java Development Kit
JRE	Java Runtime Environment
JSP	Java Server Pages
JWS	Java Web Services
GUI	Graphical User Interface
AWT	Abstract Window Toolkit
JFC	Java Foundation Classes
JAXB	Java Architecture for XML Binding
TEDS	Transducer Electronic Data Sheet
EML	Event Pattern Markup Language

ΑΝΑΦΟΡΕΣ

[1] SunSPOTWorld “Our vision” http://www.sunspotworld.com/vision.html

[2] OGC Standard *Observations and Measurement (O&M)*, Document 07-088r1, OGC, 2007

[3] OGC Standard *Sensor Model Language (SensorML)*, Document 07-000, OGC, 2007

[4] OGC Standard *Transducer Markup Language (TML)*, Document 06-010r6, OGC, 2007

[5] OGC Standard *Sensor Observation Service (SOS)*, Document 06-009r6, OGC, 2007

[6] OGC Best Practice *Sensor Alert Service (SAS)*, Document 06-028r3, OGC, 2006

[7] OGC Standard *Sensor Planning Service (SPS)*, Document 07-014r3, OGC, 2007

[8] OGC Best Practice *Web Notification Service (WNS)*, Document 06-095, 2006

[9] OGC Discussion Paper *Sensor Event Service (SES) Interface Specification*, Document 08-133, OGC, 2008

[10] OGC “All Registered Products” http://www.opengeospatial.org/resource/products

[11] Sam Bacharach, OGC Inc., “New Implementations of OGC Sensor Web Enablement Standards”, December 1, 2007 http://www.sensorsmag.com/networking-communications/government-military/new-implementations-ogc-sensor-web-enablement-standards-1437

[12] 52°North Sensor Web Packages http://52north.org/downloads/sensor-web

[13] SunSPOTWorld “Frequently Asked Questions” http://www.sunspotworld.com/docs/general-faq.php

[14] Sun Labs, *SunSPOT Owner’s Manual, Red Release 5.0*, June 2009

[15] IEEE Standard 1451 Smart Transducer Interface Standard, IEEE, 1997-2010

[16] OGC Discussion Paper *Event Pattern Markup Language (EML)*, Document 08-132, 2008

[17] Geostandards “Sensor Alert Service – Outlook on future developments”, 2009 http://geostandards.geonovum.nl/index.php/5.4.3_Sensor_Alert_Service