Ζάκυνθος 29/03/2021
Συγγραφέας Γεώργιος Μυλωνάς

Παραδοτέο 1

Στα πλαίσια Σύμβασης Έργου με Αρ. Πρωτοκόλλου 80383/24247/β1.α

Περιγραφή Δράσης

Ασφάλεια και διαχείριση δεδομένων, οργάνωση και διαδικτυακή διαθεσιμότητα της βάσης δεδομένων που τροφοδοτείται απο τις συνεχώς παραγόμενες μετρήσεις του δικτύου σταθμών, των αριθμητικών αποτελεσμάτων που παράγονται από την Μαθηματική και Στατιστική τους επεξεργασία, και των δεδομένων περιβαλλοντικού κινδύνου που θα τροφοδοτούνται σε πραγματικό χρόνο, στα πλαίσια του ΠΕ 2.1.1 : Επιχειρησιακή διάγνωση Μετεωρολογικών συνθηκών σε πραγματικό χρόνο"

Στα πλαίσια του υποέργου (με κωδικό ΕΛΚΕ Ιονίου Πανεπιστημίου 80383) “Τρέχουσες Μετεωρολογικές Συνθήκες, Κλιματική Μεταβλητότητα, και Κίνδυνος Δασικών Πυρκαγιών στα Επτάνησα”, που αποτελεί διακριτό υποέργο της Πράξης "ΛΑΕΡΤΗΣ - ΚΑΙΝΟΤΟΜΟ ΕΠΙΧΕΙΡΗΣΙΑΚΟ ΣΥΣΤΗΜΑ ΔΙΑΧΕΙΡΙΣΗΣ ΦΥΣΙΚΩΝ ΚΙΝΔΥΝΩΝ ΣΤΗΝ ΠΕΡΙΦΕΡΕΙΑ ΙΟΝΙΩΝ ΝΗΣΩΝ” MIS 5010951 / ΕΣΠΑ 2014-2020

Περιεχόμενα

1 Περιγραφή Έργου

Στα πλαίσια της σχετικής με το υποέργο 2: “Επιχειρησιακή διάγνωση Μετεωρολογικών συνθηκών σε πραγματικό χρόνο" της πράξης ΛΑΕΡΤΗΣ / MIS 5010951 σύμβασης 80383/24247/β1.α, αναπτύχθηκε μια πρώτη σειρά απο κώδικες για την αυτοματοποιημένη λειτουργία του κεντρικού εξυπηρετητή (server) του δικτύου 14 υπαίθριων επιστημονικών (Μετεωρολογικών – Περιβαλλοντικών) σταθμών υπαίθρου που το Εργαστήριο Φυσικής Περιβάλλοντος, Ενέργειας, και Περιβαλλοντικής Βιολογίας του Τμήματος Περιβάλλοντος του Ιονίου Πανεπιστημίου διαθέτει κατά μήκος των Επτανήσων. Οι κώδικες αυτοί επεξέτειναν, βελτίωναν, και συμπλήρωναν ελλείποντα επιχειρησιακά και λειτουργικά μέρη βασικού επιπέδου υποδομής του κεντρικού εξυπηρετητή, υπο μορφή μιας αναπτυσσόμενης πλατφόρμας λογισμικών, προκειμένου να πραγματοποιείται με σύγχρονους όρους ασφάλειας δεδομένων η διάγνωση ορθής λειτουργίας, η διαχείριση σφαλμάτων, και στην συνέχεια η μετάδοση, προ επεξεργασία, και αρχειοθέτηση της συνεχούς ροής των επιτόπιων υπαίθριων μετρήσεων Μετεωρολογικών και Περιβαλλοντικών παραμέτρων απο τους σταθμούς του δικτύου στον server σε πραγματικό χρόνο.

Αναλυτικότερα στόχοι των τμημάτων της πλατφόρμας λογισμικών που αναπτύχθηκε στην παρούσα ενδιάμεση (πρώτη) φάση που καλύπτει η υπάρχουσα σύμβαση είναι οι εξής:

1. Αναβάθμιση των συστημάτων μετάδοσης-ανάκτησης μετρούμενων παραμέτρων και των πρωτοκόλλων ασφάλειας της πλατφόρμας για την διασφάλιση εκροών και αποφυγή κακόβουλων διαδικτυακών επιθέσεων.
2. Βελτιστοποίηση της διαχείρισης δεδομένων πραγματικού χρόνου.
3. Διαχείριση και οργάνωση της δομής των δεδομένων που εισέρχονται

σε συνεχή ροή απο τους υπαίθριους σταθμούς του δικτύου.

4. Βελτιστοποίηση της σύνδεσης της βάσης δεδομένων με τις πηγές παροχής συνεχώς παραγόμενων μετρήσεων του δικτύου σταθμών.
5. Μαθηματική και Στατιστική προ-επεξεργασία των ανακινούμενων τιμών των μετρούμενων παραμέτρων απο τους αισθητήρες υπαίθρου σε πραγματικό χρόνο.

1 Τοπολογία επιχειρησιακού Δικτύου αυτόματων Μετεωρολογικών σταθμών Ιονίου

Οι ατμοσφαιρικές και περιβαλλοντικές παράμετροι που καταγράφονται διαρκώς από τα όργανα και αισθητήρες του επιχειρησιακού δικτύου των Αυτόματων Μετεωρολογικών Σταθμών Ιονίου και στην συνέχεια υπό αρχιτεκτονική Local Area Network μεταδίδονται ασύρματα και σε πραγματικό χρόνο δια μέσω του δικτύου κινητής τηλεφωνίας στον κεντρικό εξυπηρετητή είναι οι εξής:

- Ταχύτητα Ανέμου απο παλμικής εξόδου μηχανικό ανεμόμετρο υψηλής διακριτικής ικανότητας, σε ύψος 10 m,
- Κατεύθυνση Ανέμου απο αναλογικής τάσης εξόδου μηχανικό ανεμοδείκτη υψηλής ευαισθησίας, σε ύψος 10 m,
- Ριπή Ανέμου σε ύψος 10 m,
- Ρυθμός Βροχόπτωσης ανά λεπτό απο παλμικής εξόδου βροχόμετρο ανακρινόμενου συλλέκτη – μαγνητικά διεγειρόμενο διακόπτη, υψηλής διακριτικής ικανότητας,
- Θερμοκρασία σε ύψος 2 m απο αναλογικής τάσης εξόδου θερμόμετρο υψηλής διακριτικής ικανότητας
- Σχετική Υγρασία σε ύψος 10 m απο αναλογικής τάσης εξόδου υγρόμετρο υψηλής διακριτικής ικανότητας
- Βαρομετρική Πίεση σε ύψος 2 m απο αναλογικής τάσης εξόδου βαρόμετρο υψηλής διακριτικής ικανότητας
- Ροή ενέργειας απο Ηλιακή ακτινοβολία οπτικών μηκών κύματος σε οριζόντια επιφάνεια, απο αναλογικής τάσης εξόδου ακτινόμετρο thermistor
- Ροή ενέργειας απο Υπεριώδη Ηλιακή ακτινοβολία σε οριζόντια

επιφάνεια, απο αναλογικής τάσης εξόδου ακτινόμετρο υπεριώδους.

Οι γεωγραφικές θέσεις στις οποίες υπάρχουν εγκατεστημένοι σταθμοί του δικτύου και πραγματοποιούνται μετρήσεις των παραπάνω παραμέτρων είναι οι εξής:

- ΚΕΡΚΥΡΑ :
  - ΒΔ Κέρκυρα, περιοχή Αυλιώτες (ακρώνυμο CRF-1)
  - Κεντρική Κέρκυρα, περιοχή Τεμπλόνι (ακρώνυμο CRF-2)
  - ΝΔ Κέρκυρα, περιοχή Λίμνη Κορισίων (ακρώνυμο CRF-3)
  - Πόλη Κέρκυρας, από Οκτώβριο 2020 και μετά (ακρώνυμο CRF-4)
- ΠΑΞΟΙ :
  - Αγ. Ισαυρος (ακρώνυμο PAX-1)
- ΛΕΥΚΑΔΑ :
  - Λιμνοθάλασσα Λευκάδας (ακρώνυμο LFK-1)
- ΚΕΦΑΛΟΝΙΑ :
  - Β Κεφαλονιά, περιοχή Αντυπάτα Ερισού (ακρώνυμο KEF-1)
  - Δ Κεφαλονιά, περιοχή Κηπούρια Παλικής (ακρώνυμο KEF-2)
  - ΝΑ Κεφαλονιά, περιοχή Σκάλα Πόρου (ακρώνυμο KEF-3)
- ΖΑΚΥΝΘΟΣ :
  - ΝΔ Ζάκυνθος, περιοχή Αγαλάς (ακρώνυμο ZKT-1)
  - Ν Ζάκυνθος, περιοχή Αεροδρόμιο (ακρώνυμο ZKT-2)
  - Β Ζάκυνθος, περιοχή Σκινάρι (ακρώνυμο ZKT-3)
  - Πόλη Ζακύνθου (ακρώνυμο ZKT-4)
  - Νησίδες Στροφάδες (ακρώνυμο STR-1),
- ΗΛΕΙΑ :
  - Ακρωτήριο Κατάκολο (ακρώνυμο KTL-1)

2 Αποτύπωση Διαδικτυακού Συστήματος Διαχείρισης Δεδομένων (Δ.Σ.Δ.Δ.)

Σε αυτό το κεφάλαιο παρουσιάζουμε την αρχιτεκτονική του συστήματος που αναπτύχθηκε στα πλαίσια αυτού του έργου. Παράλληλα παρουσιάζουμε την δομή σύνδεσης του συστήματος με διάφορους σταθμούς υπαίθρου και άλλες υπηρεσίες παροχής δεδομένων.

2.1 Βασικές λειτουργίες Διαδικτυακού Συστήματος Διαχείρισης Δεδομένων

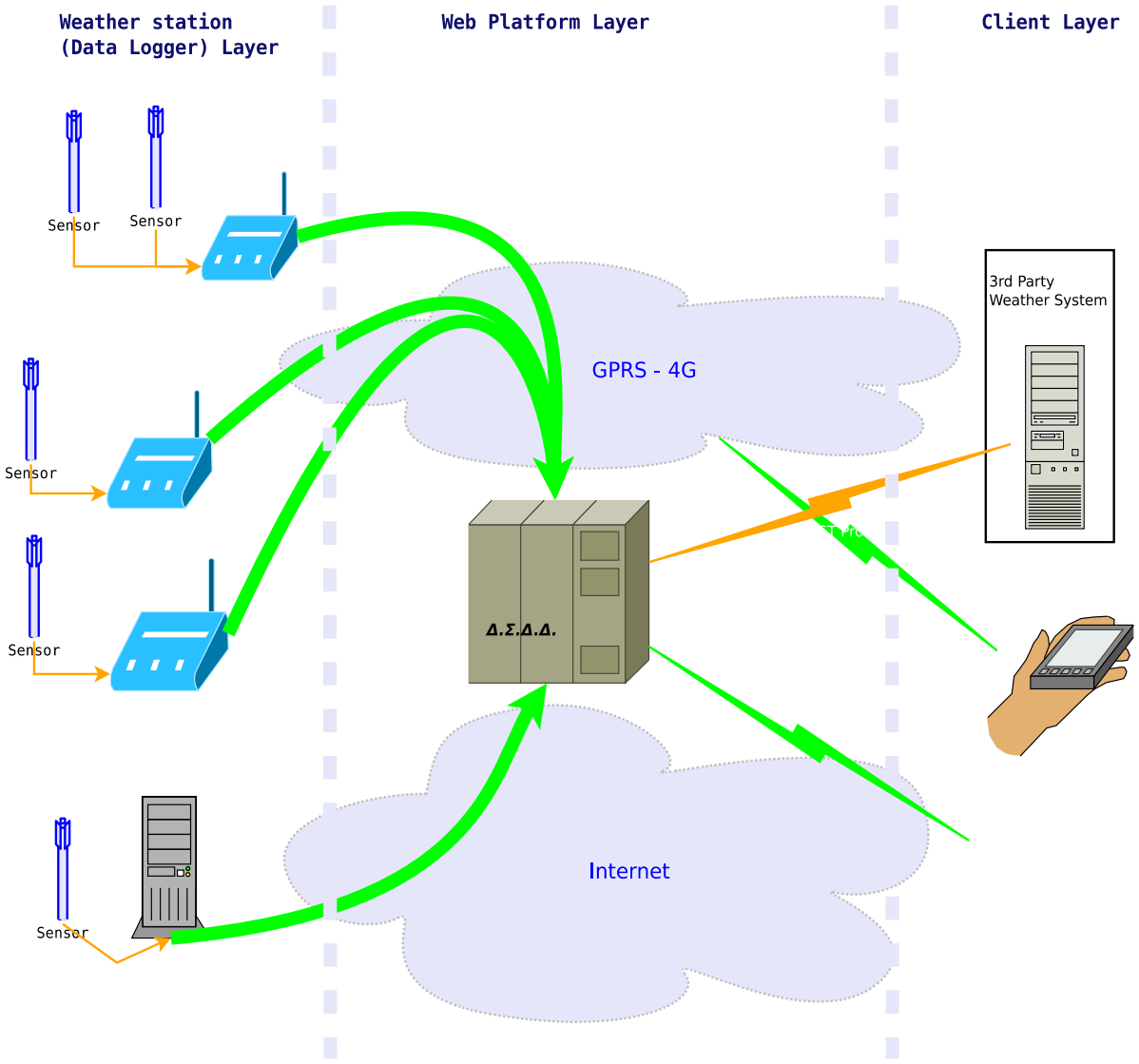
Ο βασικός στόχος του Δ.Σ.Δ.Δ. είναι να αποτελέσει τον κεντρικό κόμβο συλλογής, επεξεργασίας και διαμερισμού των δεδομένων.

Για να επιτευχθεί αυτός ο στόχος υλοποιήθηκε αρχιτεκτονική τύπου “Αστέρα¨ όπου επιτρέπει την σύνδεση στο Δ.Σ.Δ.Δ. όλων των απομακρυσμένων υπαίθριων σταθμών δεδομένων (όπως αυτοί παρουσιάζονται στο κεφάλαιο 1) οι οποίοι παράγουν και μεταδίδουν σε πραγματικό χρόνο μετεωρολογικές μετρήσεις. Το διάγραμμα της εικόνας 1 παρουσιάζει την αρχιτεκτονική αστέρα που υλοποιήθηκε.

Το σύστημα το οποίο έχει υλοποιηθεί μπορούμε να το συνοψίσουμε βάση των τα εξής χαρακτηριστικών και των δυνατοτήτων που προσφέρει:

1 Επικοινωνία με τους ψηφιακούς καταγραφείς (data loggers) των σταθμών υπαίθρου μέσω GPRS modem,

2 Προγραμματιζόμενη από τον χρήστη ανάγνωση και αρχειοθέτηση δεδομένων (real time raw data) που παράγονται σε πραγματικό χρόνο από τους υπαίθριους σταθμούς δεδομένων οι οποίοι βρίσκονται συνδεδεμένοι με το Δ.Σ.Δ.Δ.



Εικόνα 1: Αρχιτεκτονική Δ.Σ.Δ.Δ. παραγωγής και μεταφοράς μετρήσεων απο τους υπαίθριους σταθμούς του δικτύου σε φυσικό επίπεδο

3 Ανάκτηση (downloading) ολοκληρωμένων αρχείων από τους ψηφιακούς καταγραφείς, τροποποιήσεις μορφής αρχείων και διενέργεια βασικής Στατιστικής προ-επεξεργασίας (pre-processing) όπως υπολογισμός και αποθήκευση προγραμματιζόμενων χρονικών μέσων και τυπικών αποκλίσεων καθώς και άλλων περιγραφικών δεικτών.

4 Δυνατότητα άμεσης απεικόνισης τόσο των τρεχουσών τιμών όσο και των αρχειακών χρονοσειρών των καταγραφόμενων ατμοσφαιρικών παραμέτρων, τόσο αυτοδύναμα, όσο και σε αυτοματοποιημένη και

ενσωματωμένη συνεργασία με δημοφιλή λογισμικά γραφικής απεικόνισης χρονοσειρών (όπως το Grapher της Golden Software).

5 Δυνατότητα αυτοματοποιημένης παροχής στατικών τιμών των καταγραφόμενων παραμέτρων σε τρίτους φορείς παροχής πληροφοριών σε τακτά, οριζόμενα από τον χρήστη διαστήματα.

6 Δυνατότητα παροχής τρεχουσών τιμών σε δίκτυα κινητής τηλεφωνίας σε μορφή web-mobile για smart phone ή tablet.

2.2 Διαδικτυακή Πύλη

Το Δ.Σ.Δ.Δ. αναπτύχθηκε με γνώμονα την επεκτασιμότητα και την ευκολία χρήσης.

Στα πλαίσια αυτής της φιλοσοφίας αναπτύχθηκε διαδικτυακή πύλη η οποία επιτρέπει την απομακρυσμένη πρόσβαση στο Δ.Σ.Δ.Δ. και επιτρέπει:

1. Την επιχειρησιακή διαχείριση του Δ.Σ.Δ.Δ. (μέσο περιβάλλοντος διαχείρισης)
2. Πρόσβαση στα συλλεγμένα δεδομένα.

1.1.1 Διαδικτυακό Περιβάλλον Διαχείρισης Δ.Σ.Δ.Δ.

Λαμβάνοντας υποψην την ανάγκες επεκτασιμότητας και παραμετροποίησης του Δ.Σ.Δ.Δ., και χώρις να υπάρχει η ανάγκη επαναπρογραμματισμού του συστήματος σε επίπεδο κώδικα, υλοποιήθηκε διαχειριστικό περιβάλλον για ειδικά διαβαθμισμένους χρήστες τύπου διαχειριστή.

Το περιβάλλον διαχείρισης προσφέρει τις εξής δυνατότητες:

1. Σύνδεση νέων σταθμών υπαίθριων σταθμών δεδομένων και διαχείριση των ιστάμενων
2. Παραμετροποίηση διαφορετικών τύπων καναλιών δεδομένων
3. Παραμετροποίηση εξισώσεων μετα-επεξεργασιας δεδομένων (post-

processing)

4. Παραμετροποίηση προγραμματιζόμενων ενεργειών (cron-jobs)
5. Διαχείριση χρηστών και δικαιωμάτων
6. Δυναμική διαχείριση περιεχομένου σελίδων

1.1.2 Διαδικτυακό Περιβάλλον Διαχείρισης δεδομένων

Το συγκεκριμένο περιβάλλον παρέχει στον χρήστη εργαλεία και λειτουργίες οι οποίες επιτρέπουν την αναζήτηση, ανάκτηση και προβολή των συλλεγμένων δεδομένων σε πραγματικό χρόνο.

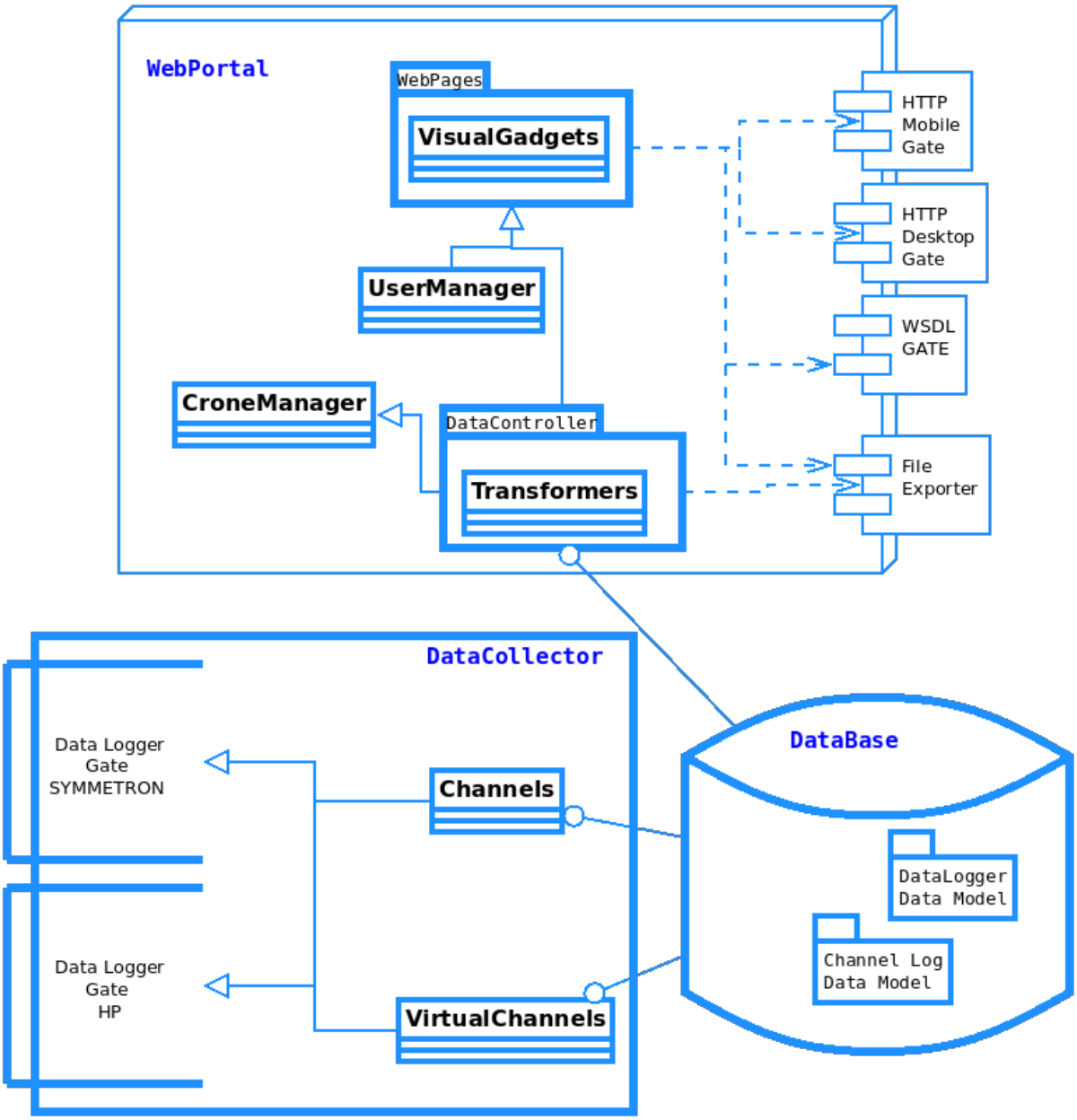
Για να εξασφαλιστεί η ορθή διαχείριση και η ασφάλεια των δεδομένων υλοποιήθηκε σύστημα διαβάθμισης έτσι ώστε να περιορίζεται η πρόσβαση των δεδομένων (μερική ή απόλυτη) ανάλογα με τον ρόλο του χρήστη.

Οι βασικότερες λειτουργίες που προσφέρονται (σε αυτό το στάδιο του έργου) είναι οι εξής:

1. Λειτουργίες για πλήρως διαβαθμισμένους χρήστες ρόλου Ερευνητή:
    1. Πλήρη πρόσβαση σε όλο το κλιματικό αρχείο των συλλεγμένων δεδομένων.
    2. Δυνατότητα εκφόρτωσης (uploading) και χειρισμού μεγάλων αρχείων (ως 100 MB) δεδομένων σε μορφή video ή video-presentations, χωρίς παρεμβολή ή φιλοξενία τρίτων πάροχων ή εξυπηρετητών (servers)
    3. Δυνατότητα δεδομένων σε μορφή xls.
    4. Ταυτόχρονη διαγραμματική προβολή δεδομένων απο 2 διαφορετικούς σταθμούς και απο δυο διαφορετικά κανάλια δεδομένων.
    5. Απεικόνιση δεδομένων σε διαδραστικό χάρτη πραγματικού χρόνου.
2. Λειτουργίες για μη διαβαθμισμένους χρήστες ρόλου επισκέπτη:

1. Περιορισμένη πρόσβαση στο κλιματικό αρχείο σε ζώνη 12 ωρών
2. Διαγραμματική προβολή δεδομένων
3. Απεικόνιση δεδομένων σε διαδραστικό χάρτη πραγματικού χρόνου.

2.3 Αρχιτεκτονική συστήματος



Εικόνα 2: UML Αρχιτεκτονικό διάγραμμα μονάδων συστήματος

Το Δ.Σ.Δ.Δ. υλοποιήθηκε με την ανάπτυξη διάφορων προγραμματιστικών

μονάδων κώδικα έτσι ώστε να υπάρχει ευελιξία σε πιθανόν μελλοντικές επεκτάσεις. Η εικόνα 2 παρουσιάζει το βασικό εννοιολογικό διάγραμμα της αρχιτεκτονικής.

Οι βασικές μονάδες οι οποίες υλοποιούν το διαδικτυακό σύστημα είναι οι εξής:

2.3.1 Data Collector

Η συγκεκριμένη μονάδα αναλαμβάνει την συλλογή δεδομένων απο σταθμούς υπαίθρου και την καταχώρηση των παραγόμενων μετρήσεων στην βάση δεδομένων.

Πρέπει να σημειωθεί οτι οι Data Collector αναπτύχθηκαν ακολουθώντας πρακτικές WSDL έτσι ώστε να υπάρχει η δυνατότητα διασύνδεσης και με άλλες πηγές δεδομένων πέρα των σταθμών υπαίθρου (π.χ. απο άλλα συστήματα μετεωρολογικών δεδομένων όπως το ECMWF Copernicus Atmosphere Monitoring Service)

Τα εργαλεία που ενσωματώνει η συγκεκριμένη μονάδα για να πετύχει το σκοπο της είναι τα παρακάτω:

- Πύλες σύνδεσης.

  Οι πύλες σύνδεσης υλοποιούν υπηρεσίες προσαρμοσμένες στις ανάγκες των διαφορετικών παρόχων (υπαίθριων σταθμών) Μετεωρολογικών μετρήσεων και αναλαμβάνουν την λήψη των δεδομένων. Σε αυτή την φάση του έργου εστιάσαμε στην προσαρμογή των υπαρχόντων σταθμών εδάφους, σε αυτό το πλαίσιο έχουν αναπτυχθεί οι εξης πύλες σύνδεσης:

  - Symmetron. Η συγκεκριμένη πύλη υλοποιεί πρωτόκολλο WSDL το οποίο λαμβάνει τις παραγόμενες Μετεωρολογικές μετρήσεις δεδομένα απο SYMMETRON Data Logger το οποίο χρησιμοποιείτε σε πλυθώρα σταθμών εδάφους. Το συγκεκριμένο Data Loger “διαβάζει” μετεωρολογικέ μετρήσεις απο διαφορετικούς αισθητήρες και τις διοχετεύει στο σύστημα

μέσα απο τα ανάλογα κανάλια δεδομένων.

- Πύλη γενικού σκοπού ενεργής συλλογής δεδομένων. Η συγκεκριμένη πύλη υλοποιεί υπηρεσία ασυγχρονων HTTP requests τα οποία σε τακτά χρονικά διαστήματα καλούν άλλες υπηρεσίες που παρέχουν κλιματικά δεδομένα μέσα απο διαδικτυακές πύλες τύπου WSDL.

- Κανάλια δεδομένων (Channels). Τα κανάλια αποτελούν τον συλλέκτη δεδομένων απο μόνο μιά κλιματική παράμετρο (π.χ. θερμοκρασία. Η συγκριμένη φάση του έργου έχει υλοποίηση τους εξής τύπους καναλιών:
  - Πρωτογενή κανάλια (RAW channels). Τα συγκεκριμένα κανάλια σύνδεσης αναλαμβάνουν την καταγραφή των αριθμητικών τιμών των μετρούμενων παραμέτρων στην βάση δεδομένων όπως αυτές παράγοντε απο τους αισθητήρες των σταθμών υπαίθρου. Η καταγραφή εμπλουτίζεται με διάφορα μεταδεδομένα όπως GIS, Timestamps κ.α.
  - Κανάλια μετα επεξεργασίας. Πρόκειται για κανάλια που αναλαμβάνουν την μεταφορά και αποθήκευση δευτερογενώς υπολογιζόμενων παραμέτρων είτε δια μέσω Στατιστικών χειρισμών (όπως μέγιστων, ελάχιστων, μέσων τιμών) είτε Μαθηματικού τυπολόγιου (όπως υπολογισμών απόλυτης υγρασίας, αναγόμενης στην επιφάνεια της θάλασσας βαρομετρικής πίεσης, υψών βροχόπτωσης σε διάφορες χρονικές κλίμακες, παρεχόμενης Ηλιακής και Αιολικής πυκνότητας ισχύος).

    Τα συγκεκριμένα κανάλια αναλαμβάνουν την προ επεξεργασία των δεδομένων και στην συνέχεια την καταχώρηση των παραγόμενων δευτερογενών παραμέτρων στην βάση δεδομένων.

- Διαχειριστής περιοδικών ενεργειών (Crone Manager). Η

συγκεκριμένη υπομονάδα αναλαμβάνει σε τακτά χρονικά διαστήματα να ενεργοποιεί τα κανάλια που την ενσωματώνουν έτσι ώστε να εκτελέσουν περιοδικά προγραμματιζόμενες εργασίες, όπως υπολογισμό ωριαίας και ημερήσιας βροχόπτωσης.

2.3.2 Web Portal

Η συγκεκριμένη μονάδα έχει δύο βασικούς σκοπούς:

1 Υλοποίηση διαδικτυακού τόπου για την διαθεσιμότητα των παραγόμενων Μετεωρολογικών μετρήσεων στους εμπλεκόμενους με το έργο ερευνητές, ενδιαφερόμενους φορείς ή/και φυσικά πρόσωπα. Η διάθεση των δεδομένων θα παρέχεται μέσο ασφαλής σύνδεσης και θα περιορίζεται ανάλογα με την διαβάθμιση του χρήστη.

2 Υλοποίηση WSDL υπηρεσίας για την διαθεσιμότητα των παραγόμενων μετρήσεων σε άλλους φορείς ή/και διαδικτυακές υπηρεσίες.

Οι βασικότερες υπομονάδες του Web Portal είναι οι εξής:

- User Manager. Η συγκεκριμένη υπομονάδα αναλαμβάνει την εξυπηρέτηση των ακόλουθων διαδικασιών εξής:
  - Διαβάθμιση των χρηστών που συνδέονται στο διαδικτυακό σύστημα διαχείρισης δεδομένων.
  - Παραγωγή πολιτικών πρόσβασης στα δεδομένα και στις υπομονάδες του WebPortal ανάλογα με το ρόλο του χρήστη
- Data Controller. Η συγκεκριμένη υπομονάδα αναλαμβάνει την συλλογή των τιμών των μετρούμενων παραμέτρων από την βάση δεδομένων ανάλογα με την πολιτική πρόσβασης του χρήστη.
- Visual Gadgets. Η συγκεκριμένη υπομονάδα αναλαμβάνει την ενσωμάτωση στον Client εργαλείων γραφικής αναπαράστασης των αριθμητικών τιμών και της χρονικής εξέλιξης των μετρούμενων παραμέτρων απο τους σταθμούς υπαίθρου.
- HTTP πύλες. Οι συγκεκριμένες πύλες υλοποιούν τους διάφορους client πρόσβασης στο διαδικτυακό σύστημα διαχείρισης δεδομένων. Οι πύλες οι

οποίες έχουν υλοποιηθεί είναι οι εξής:

- Web. Αναλαμβάνει την παραγωγή ιστοσελίδων πλοήγησης και προβολής των τιμών των μετρούμενων παραμέτρων πραγματικού χρόνου.
- File Exporter. Αναλαμβάνει την παραγωγή και συγκρότηση αριθμητικών αρχείων διαφόρων μορφών (π.χ. xls, dat, csv) των τιμών των μετρούμενων παραμέτρων πραγματικού χρόνου, σε ημερήσια βάση.
- WSDL. Αναλαμβάνει την διάχυση των παραγόμενων Μετεωρολογικών δεδομένων σε άλλες διαδικτυακές υπηρεσίες, φορείς, και εμπλεκόμενους με το έργο ερευνητές.

3 Τεχνική Ανάπτυξη Συστήματος

Σε αυτό το κεφάλαιο παρουσιάζονται οι τεχνολογίες που εφαρμόστηκαν εφαρμόστηκαν ή τελούν υπό ανάπτυξη (και τελική υλοποίηση στην δεύτερη φάση της σύμβασης) προς κατασκευή του νέου βελτιωμένου συστήματος διαχείρισης δεδομένων που παρέχονται σε συνεχή βάση και σε πραγματικό χρόνο απο τους σταθμούς του επιχειρησιακού δικτύου Μετεωρολογικών Σταθμών Ιονίου.

Το νέο σύστημα θα παρέχει:

- Ασφαλή διαχείριση δεδομένων
- Προηγμένο περιβάλλον λειτουργίας
- Βελτιωμένη μετά-επεξεργασία δεδομένων
- Ευελιξία και δυνατότητα επεκτασιμότητας σε μελλοντικές αναβαθμίσεις του δικτύου υπαίθριων σταθμών.

Στα επόμενα κεφάλαια παρουσιάζονται τα εξής:

- Η τεχνολογία πρωτοκόλλου ασφαλής επικοινωνίας
- Το περιβάλλον λειτουργίας του Δ.Σ.Δ.Δ.
- Η δομή του κώδικα που έχει αναπτυχθεί έως αυτή τη φάση του έργου.

3.1 Πρωτόκολλο ασφαλής επικοινωνίας

Η υπάρχουσα υποδομή του διαδικτυακού συστήματος διαχείρισης δεδομένων βασιζόταν σε πρωτόκολλο επικοινωνίας HTTP το οποίο δεν υποστηρίζει κανενός είδος κρυπτογράφησης δεδομένων και πρακτικά δεν παρέχει καμιά ασφάλεια τόσο για το υλιστικό/λογισμικό τμήμα του δικτύου

όσο και για τις μετρούμενες τιμές και την βάση δεδομένων του. μηδενική ασφάλεια.

Για την αντιμετώπιση του υφιστάμενου προβλήματος μηδενικής ασφάλειας χρησιμοποιήθηκαν το πρωτόκολλο SSL για την ασφάλεια των δεδομένων και η πλατφόρμα λογισμικού Docker για την βέλτιστη λειτουργία του server του δικτύου σταθμών. Παρακάτω αναλύουμε κάθε μια συνιστώσα.

3.1.1 Το πρωτόκολλο SSL (Secure Sockets Layer)

Το πρωτόκολλο SSL (Secure Sockets Layer) αναπτύχθηκε από την εταιρεία Netscape και σχεδιάστηκε για να παρέχει ασφάλεια κατά την μετάδοση ευαίσθητων δεδομένων στο διαδίκτυο. Το πρωτόκολλο χρησιμοποιείται ευρέως για ηλεκτρονικές αγορές και χρηματικές συναλλαγές, καθώς και όχι μόνον. Το SSL χρησιμοποιεί μεθόδους κρυπτογράφησης των δεδομένων που ανταλλάσσονται μεταξύ δύο συσκευών (συνηθέστερα Ηλεκτρονικών Υπολογιστών) εγκαθιδρύοντας μία ασφαλή σύνδεση μεταξύ τους μέσω του διαδικτύου. Το πρωτόκολλο αυτό χρησιμοποιεί το TCP/IP για τη μεταφορά των δεδομένων και είναι ανεξάρτητο από την εφαρμογή που χρησιμοποιεί ο τελικός χρήστης. Για τον λόγο αυτό μπορεί να παρέχει υπηρεσίες ασφαλούς μετάδοσης πληροφοριών σε πρωτόκολλα ανώτερου επιπέδου όπως για παράδειγμα το HTTP, το FTP, το telnet κοκ.

Η μετάδοση πληροφοριών μέσω του διαδικτύου γίνεται ως επί το πλείστον χρησιμοποιώντας τα πρωτόκολλα TCP/IP (Transfer Control Protocol / Internet Protocol). Το SSL λειτουργεί μετά το TCP/IP και πριν τις εφαρμογές υψηλού επιπέδου, όπως είναι για παράδειγμα το HTTP (προβολή ιστοσελίδων), το FTP (μεταφορά αρχείων) και το IMAP (email). Άρα το SSL συλλέγει τις πληροφορίες από τις εφαρμογές υψηλότερων επιπέδων, τις κρυπτογραφεί και στην συνέχεια τις μεταδίδει στο Internet προς τον Η/Υ πελάτη .

Το SSL προσφέρει συνοπτικά τις ακόλουθες υπηρεσίες:

- Πιστοποίηση του server από τον client.
- Πιστοποίηση του client από τον server.
- Εγκαθίδρυση ασφαλούς κρυπτογραφημένου διαύλου επικοινωνίας μεταξύ των δύο μερών.

Οι κρυπτογραφικοί αλγόριθμοι που υποστηρίζονται από το πρωτόκολλο είναι οι εξής: DES - Data Encryption Standard, DSA - Digital Signature Algorithm, KEA - Key Exchange Algorithm, MD5 - Message Digest, RC2/RC4, RSA, SHA-1 - Secure Hash Algorithm, SKIPJACK, Triple-DES.

Τρόπος λειτουργίας

Το πρωτόκολλο SSL χρησιμοποιεί έναν συνδυασμό της κρυπτογράφησης δημοσίου και συμμετρικού κλειδιού. Η κρυπτογράφηση συμμετρικού κλειδιού είναι πολύ πιο γρήγορη και αποδοτική σε σχέση με την κρυπτογράφηση δημοσίου κλειδιού, παρ' όλα αυτά όμως η δεύτερη προσφέρει καλύτερες τεχνικές πιστοποίησης. Κάθε σύνδεση SSL ξεκινά πάντα με την ανταλλαγή μηνυμάτων από τον server και τον client έως ότου επιτευχθεί η ασφαλής σύνδεση, πράγμα που ονομάζεται χειραψία (handshake). Η χειραψία επιτρέπει στον server να αποδείξει την ταυτότητά του στον client χρησιμοποιώντας τεχνικές κρυπτογράφησης δημοσίου κλειδιού και στην συνέχεια επιτρέπει στον client και τον server να συνεργαστούν για την δημιουργία ενός συμμετρικού κλειδιού που θα χρησιμοποιηθεί στην γρήγορη κρυπτογράφηση και αποκρυπτογράφηση των δεδομένων που ανταλλάσσονται μεταξύ τους. Προαιρετικά η χειραψία επιτρέπει επίσης στον client να αποδείξει την ταυτότητά του στον server. Αναλυτικότερα, η διαδικασία χειραψίας έχει ως εξής:

1 Αρχικά ο client στέλνει στον server την έκδοση του SSL που χρησιμοποιεί, τον επιθυμητό αλγόριθμο κρυπτογράφησης, μερικά δεδομένα που έχουν παραχθεί τυχαία και οποιαδήποτε άλλη πληροφορία χρειάζεται ο server για να ξεκινήσει μία σύνδεση SSL.

2 Ο server απαντά στέλνοντας παρόμοιες πληροφορίες με προηγουμένως συμπεριλαμβανομένου όμως και του ψηφιακού πιστοποιητικού του, το οποίο τον πιστοποιεί στον client. Προαιρετικά μπορεί να ζητήσει και το ψηφιακό πιστοποιητικό του client.

3 Ο client λαμβάνει το ψηφιακό πιστοποιητικό του server και το χρησιμοποιεί για να τον πιστοποιήσει. Εάν η πιστοποίηση αυτή δεν καταστεί δυνατή, τότε ο χρήστης ενημερώνεται με ένα μήνυμα σφάλματος και η σύνδεση SSL ακυρώνεται. Εάν η πιστοποίηση του server γίνει χωρίς προβλήματα, τότε η διαδικασία της χειραψίας συνεχίζεται στο επόμενο βήμα.

4 Ο client συνεργάζεται με τον server και αποφασίζουν τον αλγόριθμο κρυπτογράφησης που θα χρησιμοποιηθεί στην ασφαλή σύνδεση SSL. Επίσης ο client δημιουργεί το συμμετρικό κλειδί που θα χρησιμοποιηθεί στον αλγόριθμο κρυπτογράφησης και το στέλνει στον server κρυπτογραφημένο, χρησιμοποιώντας την τεχνική κρυπτογράφησης δημοσίου κλειδιού. Δηλαδή χρησιμοποιεί το δημόσιο κλειδί του server που αναγράφεται πάνω στο ψηφιακό του πιστοποιητικό για να κρυπτογραφήσει το συμμετρικό κλειδί και να του το στείλει. Στην συνέχεια ο server χρησιμοποιώντας το ιδιωτικό του κλειδί μπορεί να αποκρυπτογραφήσει το μήνυμα και να αποκτήσει το συμμετρικό κλειδί που θα χρησιμοποιηθεί για την σύνδεση.

5 Ο client στέλνει ένα μήνυμα στον server ενημερώνοντας τον ότι είναι έτοιμος να ξεκινήσει την κρυπτογραφημένη σύνδεση.

6 Ο server στέλνει ένα μήνυμα στον client ενημερώνοντας τον ότι και αυτός είναι έτοιμος να ξεκινήσει την κρυπτογραφημένη σύνδεση.

7 Από εδώ και πέρα η χειραψία έχει ολοκληρωθεί και τα μηνύματα που ανταλλάσσουν τα δύο μηχανήματα (client - server) είναι κρυπτογραφημένα.

Είδη Πιστοποιητικών Ασφαλείας SSL

Υπάρχουν 3 είδη SSL certificates:

- Domain Validation SSL Certificates (DV SSL): Χρησιμοποιούνται για τη

κωδικοποίηση της πληροφορίας και τη ταυτοποίηση των στοιχείων του καταχωρητή και του ονόματος χώρου της ιστοσελίδας (domain). Με αυτόν το τρόπο ο χρήστης μπορεί να είναι σίγουρος ότι η διεύθυνση της ιστοσελίδας είναι σωστή και παραπέμπει στο σωστό εξυπηρετητή. Αυτό το είδος πιστοποιητικού είναι πολύ εύκολο στην έκδοση / εγκατάσταση του, έχει μικρό κόστος και καλύπτει τη συντριπτική πλειοψηφία των ιστοσελίδων.

- Organization Validation SSL Certificates (OV SSL): Χρησιμοποιούνται για τη κωδικοποίηση της πληροφορίας και τη ταυτοποίηση της ιστοσελίδας και της εταιρίας που βρίσκεται πίσω από αυτήν. Πριν οι Αρχές Πιστοποιητικών εκδώσουν το SSL πιστοποιητικό, ακολουθούν μια διαδικασία ταυτοποίησης της εταιρίας, της διεύθυνσής της καθώς και της ιδιοκτησίας του Domain Name.
- Extended Validation SSL Certificates (EV SSL): Είναι τα πιο πλήρη και ασφαλή SSL πιστοποιητικά που μπορεί να παρέχει μια ιστοσελίδα στους χρήστες της σήμερα. Η αδειοδότηση και έκδοση των πιστοποιητικών αυτών απαιτεί διεξοδικές διαδικασίες, μέσα από τις οποίες οι Αρχές Πιστοποίησης ελέγχουν σχεδόν όλες τις πτυχές μιας εταιρίας και της ιστοσελίδας (εικόνα 3)

Εικόνα 3: Στάδια εκτέλεσης ασφαλούς επικοινωνίας

Certificate Authorities

Το SSL είναι μια ηλεκτρονική βεβαίωση που εκδίδεται (κατόπιν ελέγχων) από ανεξάρτητους και φερέγγυους οργανισμούς οι οποίοι ονομάζονται Αρχές Πιστοποιητικών (Certificate Authorities). Οι εταιρείες αυτές αναλαμβάνουν την ταυτοποίηση των στοιχείων της ιστοσελίδας καθώς και την ασφαλή μεταφορά δεδομένων μεταξύ των ιστοσελίδων αυτών και των χρηστών τους.

Υπάρχουν πολλές τέτοιες ενδιάμεσες εταιρείες για την πραγματοποίηση αυτών των υπηρεσιών (όπως η Let’s Encrypt).

3.1.2 Let's Encrypt (https://letsencrypt.org/)

Η Let's Encrypt αποτελεί μια αυτοματοποιημένη και ανοιχτή αρχή έκδοσης πιστοποιητικών (CA), που λειτουργεί προς όφελος του κοινού. Είναι μια υπηρεσία που παρέχεται από το Internet Security Research Group (ISRG). Δίνει τα ψηφιακά πιστοποιητικά που χρειάζονται για να ενεργοποιηθεί το HTTPS (SSL / TLS) για ιστότοπους, με τον πιο φιλικό προς τον χρήστη τρόπο.

Οι βασικές αρχές του Let's Encrypt είναι:

- Δωρεάν κόστος συντήρησης: Όποιος έχει ένα domain name μπορεί να χρησιμοποιήσει το Let's Encrypt για να αποκτήσει ένα αξιόπιστο πιστοποιητικό χωρίς να απαιτείτε επαναλαμβανόμενη χρέωση για την έκδοση του.
- Αυτοματισμός: Το λογισμικό που εκτελείται σε εναν web server μπορεί να αλληλεπιδράσει με το Let's Encrypt για να αποκτήσει ένα πιστοποιητικό, διαμορφωμένο με ασφάλεια για χρήση και να αναλάβει αυτόματα την ανανέωσή του.
- Ασφάλεια: Η Let's Encrypt λειτουργεί ως πλατφόρμα για την προώθηση των βέλτιστων πρακτικών ασφάλειας TLS, τόσο από την πλευρά της CA όσο και βοηθώντας τους χειριστές ιστότοπων να ασφαλίσουν σωστά τους διακομιστές τους.
- Διαφάνεια: Όλα τα πιστοποιητικά που εκδίδονται ή ανακαλούνται καταγράφονται δημόσια και είναι διαθέσιμα προς ανοικτή επιθεώρηση.
- Ανοιχτό πρότυπο: Το πρωτόκολλο αυτόματης έκδοσης και ανανέωσης δημοσιεύεται ως ανοιχτό πρότυπο.

Ο στόχος της Let's Encrypt και του πρωτοκόλλου ACME είναι να καταστήσει δυνατή τη δημιουργία ενός διακομιστή HTTPS να αποκτήσει αυτόματα ένα πιστοποιητικό αξιόπιστου προγράμματος περιήγησης, χωρίς ανθρώπινη

παρέμβαση. Αυτό επιτυγχάνεται εκτελώντας έναν πράκτορα διαχείρισης πιστοποιητικών (certificate management agent) στον διακομιστή ιστού (web server). Ακολουθώντας την διαδικασία ρύθμισης του https://example.com/ με έναν πράκτορα διαχείρισης πιστοποιητικών που υποστηρίζει το Let's Encrypt πραγματοποιούνται δύο διαδικαστικά βήματα. Πρώτον, ο πράκτορας αποδεικνύει στην CA (αρχή έκδοσης πιστοποιητικών) ότι ο διακομιστής ιστού ελέγχει έναν τομέα. Στη συνέχεια, ο πράκτορας μπορεί να ζητήσει, να ανανεώσει και να ανακαλέσει πιστοποιητικά για αυτόν τον τομέα.

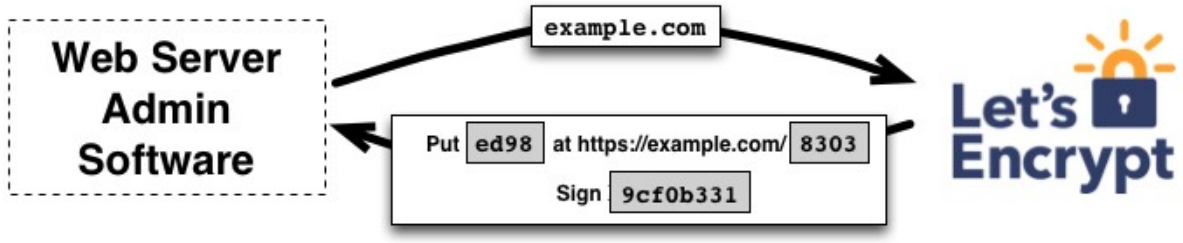
Επικύρωση Τομέα (Domain)

Το Let's Encrypt προσδιορίζει τον διαχειριστή του διακομιστή με δημόσιο κλειδί. Την πρώτη φορά που το λογισμικό του πράκτορα αλληλεπιδρά με το Let's Encrypt, δημιουργεί ένα νέο ζεύγος κλειδιών και αποδεικνύει στο Let's Encrypt CA ότι ο διακομιστής ελέγχει έναν ή περισσότερους τομείς (domain). Αυτό είναι παρόμοιο με την παραδοσιακή διαδικασία CA για τη δημιουργία λογαριασμού και την προσθήκη τομέων σε αυτόν τον λογαριασμό.

Για να ξεκινήσει η διαδικασία, ο πράκτορας ρωτά το Let's Encrypt CA τι πρέπει να κάνει για να αποδείξει ότι ελέγχει το example.com. Το Let's Encrypt CA εξετάζει το ζητούμενο όνομα τομέα και εκδίδει ένα ή περισσότερα σύνολα προκλήσεων. Αυτοί είναι διαφορετικοί τρόποι με τους οποίους ο πράκτορας μπορεί να αποδείξει τον έλεγχο του τομέα. Για παράδειγμα, η CA ενδέχεται να δώσει στον πράκτορα την επιλογή ενός από τα εξής:

- Παροχή εγγραφής DNS στο example.com ή
- Παροχή πόρου HTTP κάτω από ένα γνωστό URI στο http://example.com/

Μαζί με τις προκλήσεις, το Let's Encrypt CA παρέχει επίσης την δυνατότητα στον πράκτορα να υπογράψει με το ζεύγος ιδιωτικών κλειδιών του για να αποδείξει ότι ελέγχει το ζεύγος κλειδιών (εικόνα 4).



Εικόνα 4: Διαδικασία ανάκτησης πιστοποιητικού

Το λογισμικό του πράκτορα ολοκληρώνει ένα από τα παρεχόμενα σύνολα προκλήσεων. Αν είναι σε θέση να ολοκληρώσει τη δεύτερη εργασία τότε δημιουργεί ένα αρχείο σε μια καθορισμένη διαδρομή στον ιστότοπο http://example.com. Ο πράκτορας υπογράφει επίσης την παρεχόμενη nonce (έναν αυθαίρετο αριθμό που μπορεί να χρησιμοποιηθεί μόνο μία φορά σε μια κρυπτογραφική επικοινωνία) με το ιδιωτικό κλειδί του. Μόλις ο πράκτορας ολοκληρώσει αυτά τα βήματα, ειδοποιεί την CA ότι είναι έτοιμη να ολοκληρώσει την επικύρωση.

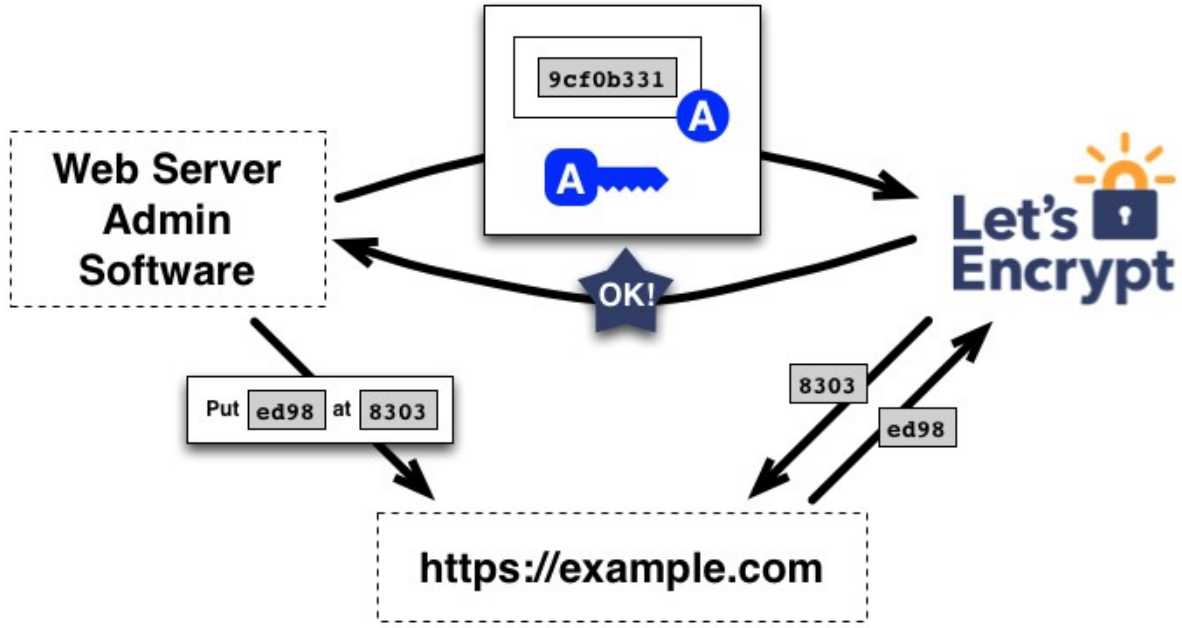
Στη συνέχεια, είναι καθήκον της CA να ελέγχει εάν οι προκλήσεις έχουν ικανοποιηθεί. Η CA επαληθεύει την υπογραφή στο nonce και προσπαθεί να κατεβάσει το αρχείο από τον διακομιστή ιστού και να βεβαιωθεί ότι έχει το αναμενόμενο περιεχόμενο.

Εάν η υπογραφή μέσω του nonce είναι έγκυρη και οι προκλήσεις ελεγχθούν, τότε ο πράκτορας που προσδιορίζεται από το δημόσιο κλειδί είναι εξουσιοδοτημένος να κάνει διαχείριση πιστοποιητικών για το example.com. Καλούμε το ζεύγος κλειδιών που ο πράκτορας χρησιμοποίησε ένα "ζεύγος εξουσιοδοτημένων κλειδιών" (authorized key pair) για το example.com

Έκδοση και ανάκληση πιστοποιητικού

Όταν ο πράκτορας έχει ένα εξουσιοδοτημένο ζεύγος κλειδιών, το αίτημα, η ανανέωση και η ανάκληση πιστοποιητικών είναι απλό - απλώς στέλνονται μηνύματα διαχείρισης πιστοποιητικών και υπογράφονται τα με το

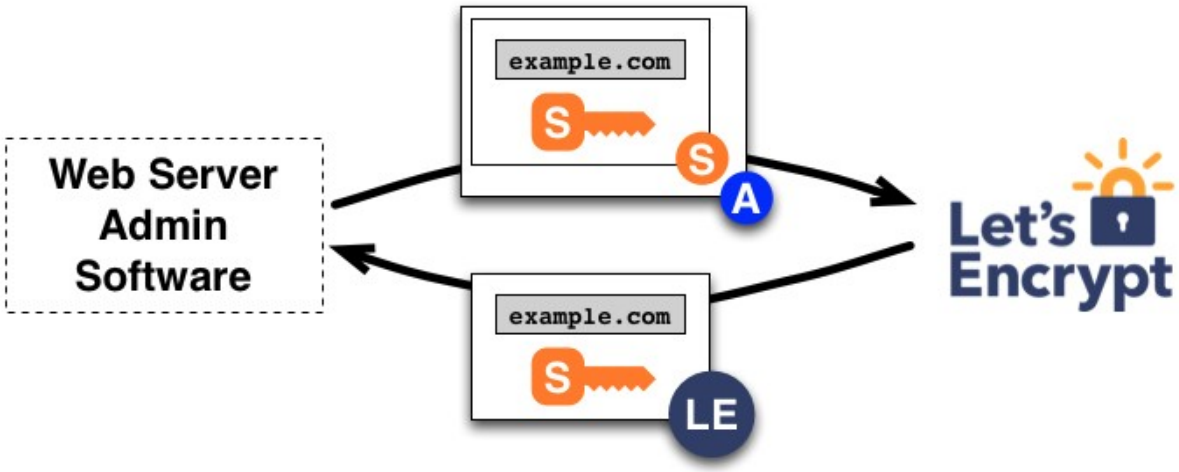
εξουσιοδοτημένο ζεύγος κλειδιών (Εικόνα 5).



Εικόνα 5: Κύκλος παραγωγής πιστοποιητικού

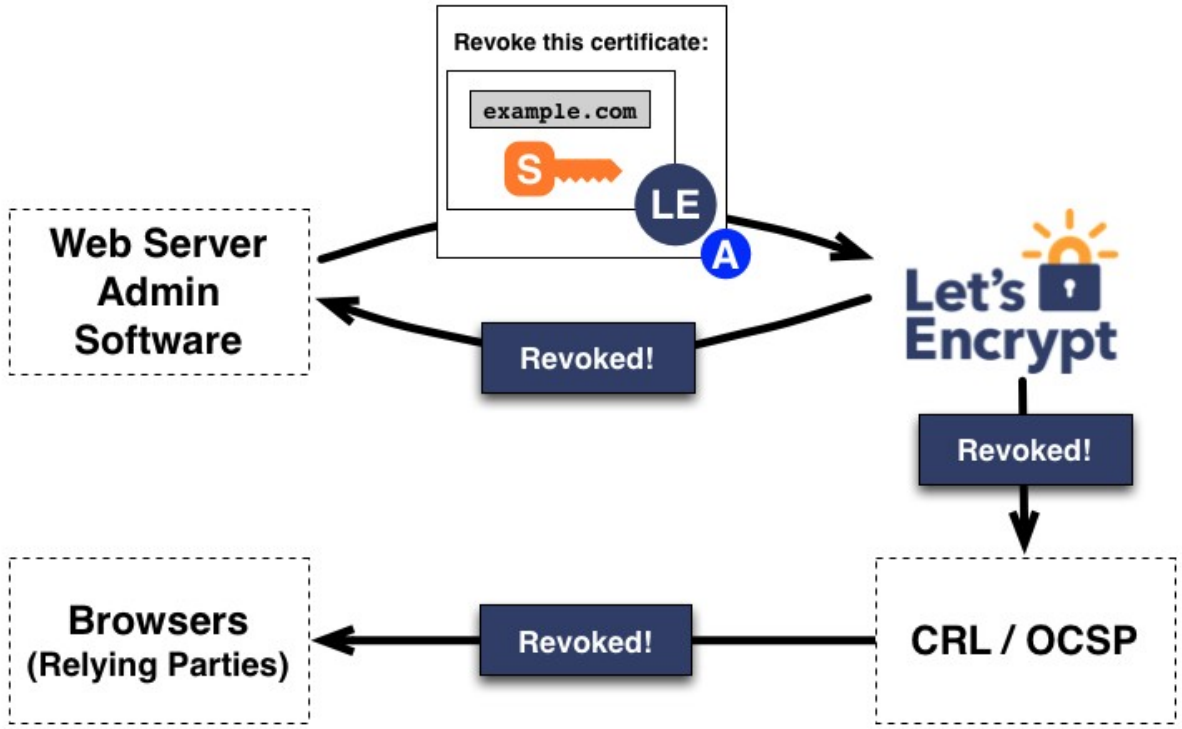
Για να αποκτήσει ένα πιστοποιητικό για τον τομέα, ο πράκτορας δημιουργεί ένα αίτημα υπογραφής πιστοποιητικού PKCS#10 που ζητά από την Let's Encrypt CA να εκδώσει ένα πιστοποιητικό για το example.com με ένα καθορισμένο δημόσιο κλειδί. Η CSR περιλαμβάνει μια υπογραφή από το ιδιωτικό κλειδί που αντιστοιχεί στο δημόσιο κλειδί στην CSR. Ο πράκτορας υπογράφει επίσης ολόκληρη την CSR με το εξουσιοδοτημένο κλειδί για το example.com, έτσι ώστε η Let's Encrypt CA να γνωρίζει ότι είναι εξουσιοδοτημένο.

Όταν η Let's Encrypt CA λαμβάνει το αίτημα, επαληθεύει και τις δύο υπογραφές. Στην συνέχεια, εκδίδει ένα πιστοποιητικό για το example.com με το δημόσιο κλειδί από την CSR και το επιστρέφει στον πράκτορα (Εικόνα 6).



Εικόνα 6: Διαδικασία έκδοσης πιστοποιητικού

Η ανάκληση λειτουργεί με παρόμοιο τρόπο. Ο πράκτορας υπογράφει ένα αίτημα ανάκλησης με το ζεύγος κλειδιών εξουσιοδοτημένο για το example.com και η Let's Encrypt CA επαληθεύει ότι το αίτημα έχει εγκριθεί. Εάν ναι, δημοσιεύει πληροφορίες ανάκλησης στα κανονικά κανάλια ανάκλησης (δηλ. OCSP), έτσι ώστε τα εμπιστευτικά μέρη, όπως τα προγράμματα περιήγησης, να μπορούν να γνωρίζουν ότι δεν πρέπει να αποδεχτούν το ανακληθέν πιστοποιητικό (Εικόνα 7).



Εικόνα 7: Διαδικασία ανάκλησης πιστοποιητικού

3.2 Αναβάθμιση περιβάλλοντος λειτουργίας συστήματος

Η υφιστάμενη βασική υποδομή του διαδικτυακού συστήματος διαχείρισης δεδομένων του δικτύου Μετεωρολογικών Σταθμών Ιονίου λειτουργεί μέσα σε περιβάλλον XAMP το οποίο δεν αποτελεί ένα κλειστό περιβάλλον αλλά αποτελεί συλλογή διαφόρων παρελκόμενων προγραμμάτων (πχ. Mysql, Apache κλπ). Αυτού του τύπου η αρχιτεκτονική καθιστά ολόκληρο το σύστημα (server, ψηφιακούς καταγραφείς στους απομακρυσμένους σταθμούς υπαίθρου, πλατφόρμα λογισμικού, βάση δεδομένων) εκτεθειμένο σε προσβολές απο ιούς και κακόβουλα λογισμικά.

Με σκοπό την επιδιωκόμενη αύξηση ασφάλειας, το νέο σύστημα υλοποιήθηκε και εγκαταστάθηκε σε ασφαλές κλειστό περιβάλλον Docker το οποίο, σε περιβάλλον Windows στο οποίο λειτουργεί ο server του δικτύου σταθμών, βρίσκεται εγκατεστημένο σε Virtual Machine.

3.3 Η πλατφόρμα λογισμικού ανοιχτού κώδικα Docker

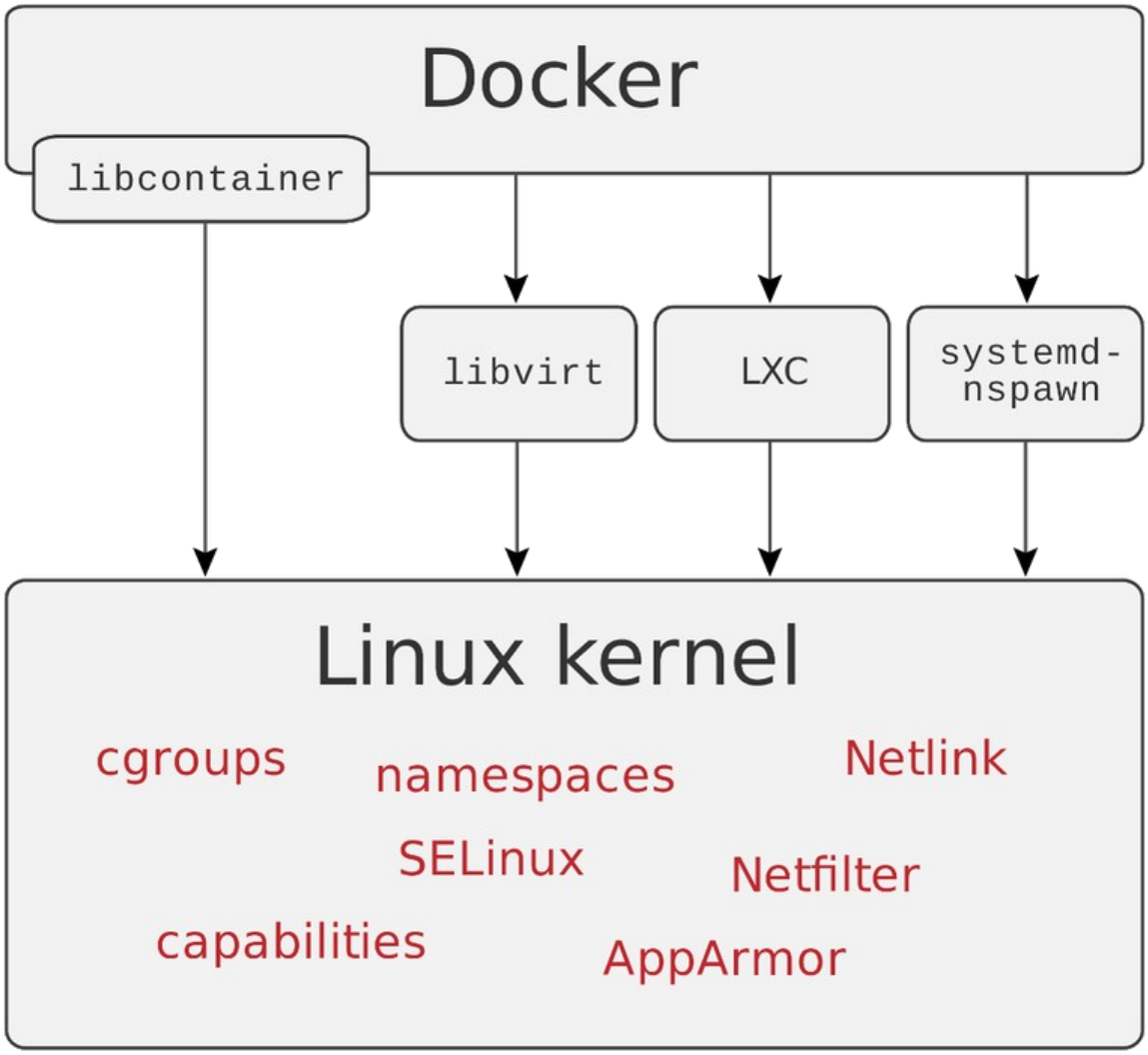
Το Docker είναι μια πλατφόρμα λογισμικού ανοιχτού κώδικα η οποία προσφέρει ενα κλειστό περιβάλλον λειτουργικού συστήματος, μέσα σο οποίο εγκαταστάθηκε και βρίσκετε σε λειτουργία το Δ.Σ.Δ.Δ. πραγματοποιήθηκε στο παρόν έργο.

Στο απομονωμένο περιβάλλον που προσφέρει το Docker εχει εγκατασταθεί μόνο ο απαιτούμενος κώδικας του Δ.Σ.Δ.Δ. και τα απαραίτητα παρελκόμενα αναγκαία συστήματα.

Ενα επιπλέον πλεονέκτημα αυτης της αρχιτεκτονικής είναι η βέλτιστη χρήση της CPU, και της RAM του κεντρικού server του δικτύου σταθμών, το οποίο προσφέρει βελτιωμένη απόκριση του Δ.Σ.Δ.Δ. ιδιαίτερα κατα την στατιστική ανάλυση μεγάλων ομάδων μετρήσεων.

Κύριος στόχος του Docker είναι να αυτοματοποιήσει την διανομή εφαρμογών μέσα σε ασφαλής τομής λογισμικού (containers) καθώς και την αυτοματοποίηση του Virtualization των μικρο-υπηρεσιών του λειτουργικού συστήματος στο Linux (Εικόνα 8).

Το Docker χρησιμοποιείται, σε αυτό το έργο, ως υπηρεσία με την οποία

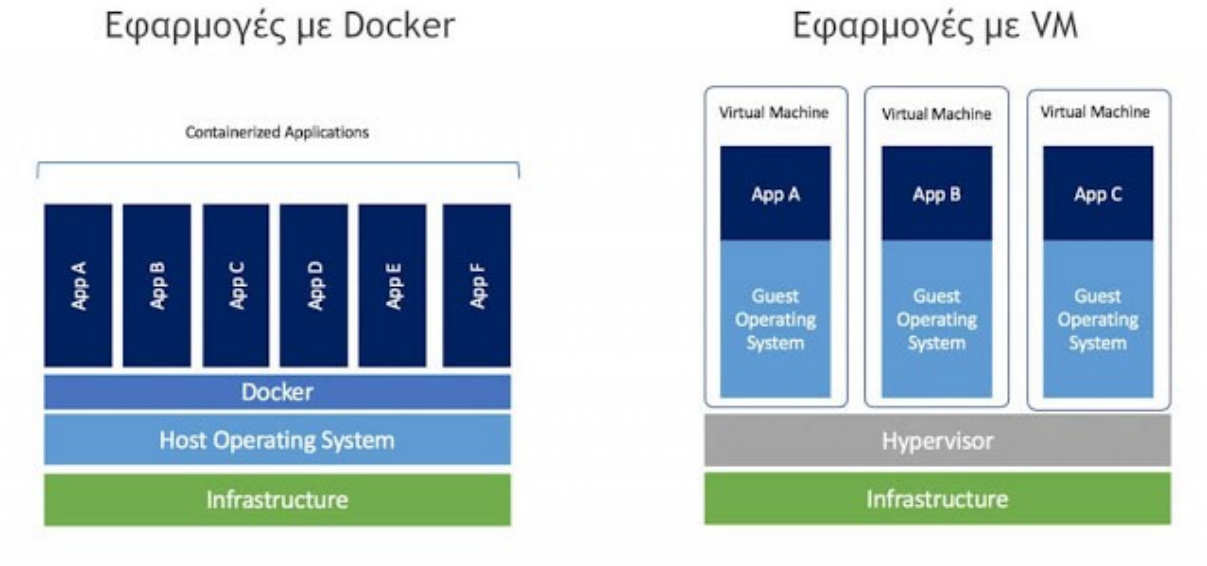


Εικόνα 8: Αρχιτεκτονικη Docker

σε περιβάλλον Windows υλοποιήθηκαν και εκτελούνται διαδικασίες και κώδικες περιβάλλοντος Linux, ή γενικά εκτελούνται ακόμα και άγνωστης προέλευσης διαδικασίες και εφαρμογές χωρίς να κινδυνεύσει η ευστάθεια, τα επι μέρους τμήματα, και το λειτουργικό σύστημα του server του δικτύου σταθμών.

Στο παρόν έργο, το Δ.Σ.Δ.Δ. καθώς και οι παρελκόμενες εφαρμογές οι εφαρμογές αυτές τρέχουν σε έναν ασφαλή τομέα (container), τα περιεχόμενα του οποίου είναι απομονωμένα από το υπόλοιπο περιβάλλον

του κεντρικού server του δικτύου σταθμών.



Εικόνα 9: Συγκριση δομης Docker με δομή V.M.

Τα οφέλη που προέκυψαν απο την εφαρμογή της παραπάνω τεχνικής με χρήση του Docker σε σχέση με ένα "συμβατικό" server είναι ότι:

- το λειτουργικό σύστημα του server του διαδικτυακού συστήματος διαχείρισης των δεδομένων που παραλαμβάνονται απο τους Μετεωρολογικούς Σταθμούς υπαίθρου είναι ασφαλές,
- Οποιαδήποτε λάθος ή μη-επιθυμητή διαδικασία σε κάποιο container δεν επηρεάζονται τα υπόλοιπα μέρη του συστήματος ούτε το λειτουργικό σύστημα του server,
- η δημιουργία και η διαγραφή του κάθε container καθίσταται απλή διαδικασία
- Μελλοντική ανάγκη “μετακόμισης” του Δ.Σ.Δ.Δ. σε άλλο server θα είναι πλέον εφικτή χωρίς να υπάρχει η ανάγκη εγκατάστασης άλλων λογισμικών παρά μόνο το Δ.Σ.Δ.Δ. image του docker το οποίο συμπεριλαμβάνει τον κώδικα του Δ.Σ.Δ.Δ. και όλες τα απαραίτητα συστήματα.

Επιπλέον (Εικόνα 9), ένα container docker -σε αντίθεση με μια εικονική μηχανή και το LXC δεν απαιτεί ούτε περιλαμβάνει ξεχωριστό λειτουργικό σύστημα, αλλά αντίθετα, βασίζεται στη λειτουργικότητα του πυρήνα του Linux και χρησιμοποιεί απομόνωση πόρων. Τα container docker δημιουργούνται από στιγμιότυπα (images) docker που στην πράξη αντιπροσωπεύουν και λειτουργούν ως ζωντανή κατάσταση μιας διαδικτυακής εφαρμογής που τρέχει από ένα αρχείο ISO που όμως είναι το ισοδύναμο του στιγμιότυπου του docker, και περιέχει μόνο την εφαρμογή και τις εξαρτήσεις της.

4 Μονάδες διαχείρισης και επεξεργασίας μετρούμενων παραμέτρων

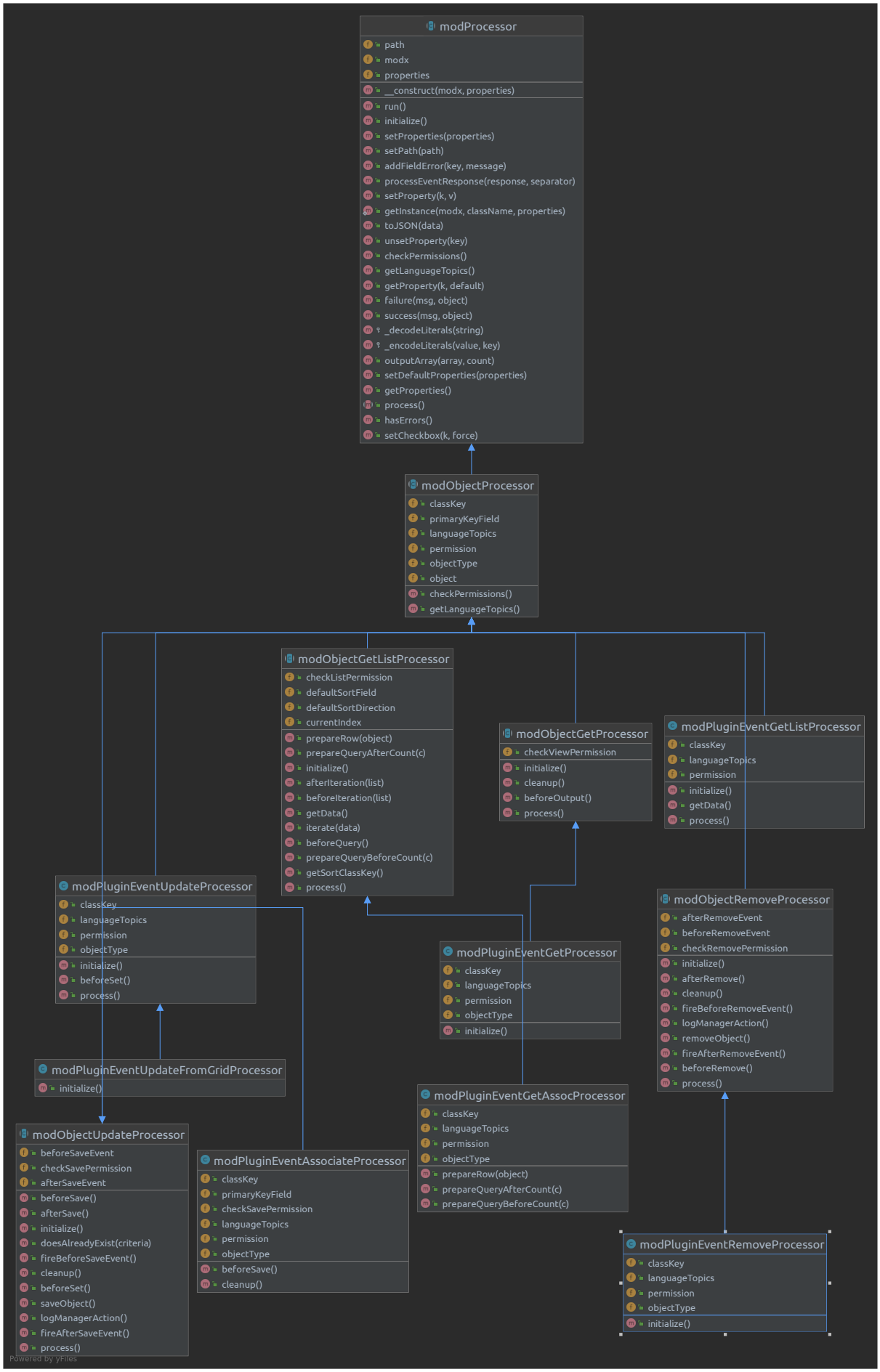
Όπως προαναφέρθηκε η ανάπτυξη του συστήματος βασίστηκε στο προγραμματιστικό μοντέλο (Framework) MODx revolution.

Για να γίνει κατανοητό ο τρόπος που αναπτύχθηκε το σύστημα παρουσιάζουμε στην εικόνα 10 το UML διάγραμμα των κλάσεων που παρέχονται απο το MODx όσον αφορά την διαχείριση των Event.

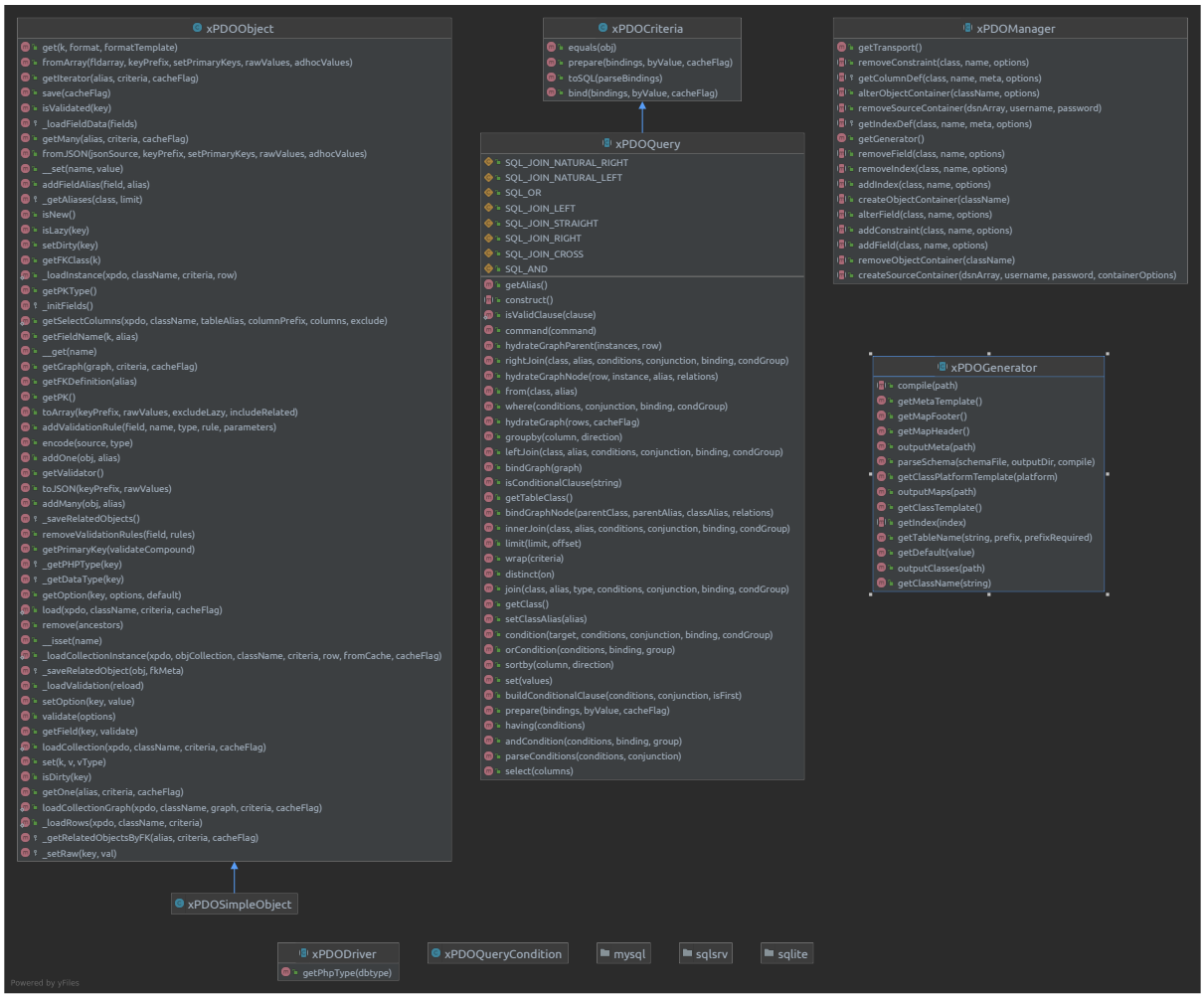
Το MODx παρέχει σημαντική υποστήριξη για event driven προγραμματισμό. Η συγκεκριμένη φιλοσοφία χρησιμοποιήθηκε για την ανάπτυξη των DataCollector ο οποίος ακολουθήθηκε κατα την ανάπτυξη των παραδοτέων αυτού του έργου.

Όσον αφορά την διαχείριση και καταχώρηση των δεδομένων βασιστήκαμε στο προγραμματιστικό πλαίσιο xPDO. Η βασική του αρχιτεκτονική παρουσιάζεται στην εικόνα 11 Το xPDO παρέχει ένα προηγμένο σύστημα διαχείρισης και σύνδεσης σε διαφορετικές τεχνολογίες βάσεων δεδομένων. Παρακάτω παρουσιάζονται τα πιό νευραλγικά σημεία του κώδικα που αναπτύχθηκε στα πλαίσια της ενεργού σύμβασης, μου και αφορά για τη συλλογή και παρουσίαση μεταφορά, ανάκτηση, Στατιστική προ-επεξεργασία και αρχειοθέτηση των τιμών των μετρούμενων παραμέτρων απο τους Μετεωρολογικούς σταθμούς υπαίθρου κατά μήκος του Ιονίου, καθώς και μια περιγραφή του της λειτουργίας του κάθε τμήματός.

Στη συνέχεια αυτού του κεφαλαίου παρουσιάζονται μερικά δείγματα απο τον κώδικα που έχει αναπτυχθεί σε αυτή την φάση του έργου.



Εικόνα 10: UML Event Class Diagram



Εικόνα 11: UML ORM Class Diagram

4.1 Κώδικας σχετικός με μονάδες Data Logger/Channel

Κατά την συλλογή (ανάκτηση) και μεταφορά των δεδομένων απο τους Μετεωρολογικούς σταθμούς του δικτύου, το κάθε κανάλι ανάγνωσης και καταγραφής τιμών κάθε ψηφιακού καταγραφέα παρέχει τις συνεχώς μετρούμενες τιμές για τις ακόλουθες παραμέτρους, σε πραγματικό χρόνο:

- Ταχύτητα και κατεύθυνση του ανέμου σε ύψος 10 m καθώς και τις ανά λεπτό τιμές ριπής ανέμου,

- Ρυθμό βροχόπτωσης ανά λεπτό,
- Θερμοκρασία του αέρα,
- Σχετική υγρασία του αέρα,
- Ατμοσφαιρική πίεση,
- Παροχή Ηλιακής ενέργειας σε οριζόντια επιφάνεια στα οπτικά μήκη κύματος και το κοντινό υπέρυθρο,
- Παροχή ενέργειας σε οριζόντια επιφάνεια απο την υπεριώδη Ηλιακή ακτινοβολία
- Τάση της συστοιχίας μπαταριών μέσω της οποίας τροφοδοτείται κάθε υπαίθριος σταθμός.

Οι DataLogger αναπτύχθηκαν σαν μονάδες προσαρασσόμενες τύπου Plugin ετσι ώστε να μπορούν μελλοντικά να προσαρτηθούν νέες χωρίς να υπάρχει καμία ανάγκη παρέμβασης στον κώδικα άλλων μονάδων του συστήματος.

Το τμήμα του κώδικα παρακάτω παρουσιάζετε ώς παράδειγμα της ανάπτυξης του DataLogger το οποίο αναλαμβάνει την ζεύξη των μονάδων Symmetron.

Αυτό το τμήμα του κώδικα πυροδοτείτε μέσα απο το σύστημα των Event του MODx και αναλαμβάνει των διαχωρισμό των δεδομένων σε διάφορα κανάλια και την διοχέτευση των δεδομένων στο επόμενο στάδιο που αφορά την καταχώρηση τους στην βάση δεδομένων απο το xPDO

```
$rdata = array();
$string_data=explode("@", $_GET["search_str"]);
foreach ($string_data as $sub_data) {
$data = explode("|", $sub_data);

$i=0;
```

```
$dbRecord=array();

$dbRecord["software_channel_sn"]=$data[0];
$dbRecord["software_channel_pwd"]=$data[1];
$dbRecord["software_channel_datetime"]=$data[2];
$dbRecord["software_channel_value"]=$data[5];
$dbRecord["software_channel_datetime"] . " " . date("d/m G:i",time()) );

   switch ($data[3]) {
        case "Ch1":
            $dbRecord["software_channel_type"]="Rain Totalize";
            $dbRecord["software_channel_tag"]="mm";
            $dbRecord["software_channel_name"]="RT";
            $rdata["RT"] = $dbRecord["software_channel_value"];
        break;
        case "Ch2":

          if($dbRecord["software_channel_sn"]=="041A0264"){
                $dbRecord["software_channel_type"]="Relative Humidity of
Air";
              $dbRecord["software_channel_tag"]="%";
              $dbRecord["software_channel_name"]="RH";
              $rdata["RH"] = $dbRecord["software_channel_value"];
          }
          else{
              $dbRecord["software_channel_type"]="UVA Radiation";
              $dbRecord["software_channel_tag"]="W m^-2";
```

```
                $dbRecord["software_channel_name"]="UVA";
                $rdata["UVA"] = $dbRecord["software_channel_value"];
            }
        break;
        case "Ch3":
            $dbRecord["software_channel_type"]="Air Tempreture";
            $dbRecord["software_channel_tag"]="celsius degrees";
            $dbRecord["software_channel_name"]="AT";
            $rdata["AT"] = $dbRecord["software_channel_value"];
        break;
        case "Ch4":
                $dbRecord["software_channel_type"]="Relative Humidity of
Air";
            $dbRecord["software_channel_tag"]="%";
            $dbRecord["software_channel_name"]="RH";
            $rdata["RH"] = $dbRecord["software_channel_value"];
        break;
        case "Ch5":
            $dbRecord["software_channel_type"]="Solar Radiation";
            $dbRecord["software_channel_tag"]="W m^-2";
            $dbRecord["software_channel_name"]="SR";
            $rdata["SR"] = $dbRecord["software_channel_value"];
        break;
        case "Ch6":
            if($dbRecord["software_channel_sn"]=="041A0258"
                                    ||
$dbRecord["software_channel_sn"]=="020A0022"
```

```
                                    ||
$dbRecord["software_channel_sn"]=="041A0270"){
                $dbRecord["software_channel_type"]="UVB Radiation";
                $dbRecord["software_channel_tag"]="W m^-2";
                $dbRecord["software_channel_name"]="UVB";
                $rdata["UVB"] = $dbRecord["software_channel_value"];
            }
            else{
                        $dbRecord["software_channel_type"]="Atmospheric
Pressure";
                $dbRecord["software_channel_tag"]="mBar";
                $dbRecord["software_channel_name"]="AP";
                $rdata["AP"] = $dbRecord["software_channel_value"];
            }
        break;
        case "Ch7":
            $dbRecord["software_channel_type"]="Wind Direction";
            $dbRecord["software_channel_tag"]="degrees";
            $dbRecord["software_channel_name"]="WD";
            $rdata["WD"] = $dbRecord["software_channel_value"];
        break;
        case "Ch8":
            $dbRecord["software_channel_type"]="Wind Speed";
            $dbRecord["software_channel_tag"]="m/s";
            $dbRecord["software_channel_name"]="WS";
            $rdata["WS"] = $dbRecord["software_channel_value"];
        break;
```

```
        case "Ch9":
            $dbRecord["software_channel_type"]="Unit Battery";
            $dbRecord["software_channel_tag"]="volts";
            $dbRecord["software_channel_name"]="UB";
            $rdata["UB"] = $dbRecord["software_channel_value"];
        break;
        case "Ch8-max":
            $dbRecord["software_channel_type"]="Wind Gust";
            $dbRecord["software_channel_tag"]="m/s";
            $dbRecord["software_channel_name"]="MaxWS";
            $rdata["MaxWS"] = $dbRecord["software_channel_value"];
        break;
      default :
        $dbRecord["software_channel_name"]=$data[3];
   }
// }
```

4.2 Κώδικας σχετικός με μονάδες Virtual Channel

4.2.1 Error Correction

Το παρακάτω τμήμα κώδικα αναπτύχθηκε στα πλαίσια καναλιού μετα επεξεργασίας δεδομένων. Σκοπός αυτού του κώδικα είναι η αναγνώριση και διαχείριση σφαλμάτων που μπορεί να προκύψουν απο κακή λειτουργία αισθητήρων. Π.χ. αν παρατηρηθεί οτι υπάρχει ξαφνικό άλμα στην μέτρηση κάποιου αισθητήρα τότε η μέτρηση αυτή φιλτράρεται έτσι ώστε να μήν

“προσβάλει” το κλιματικό αρχείο με ψευδή δεδομένα,

```
// Error Correction
if($dbRecord["software_channel_name"]=="RH" && $dbRecord["software_channel_value"] > 100 && $dbRecord["software_channel_value"] <= 100.5){
	$dbRecord["software_channel_value"] = 100;
}

if(
	// Air Tempreture
	($dbRecord["software_channel_name"]=="AT" && ($dbRecord["software_channel_value"] < -15 || $dbRecord["software_channel_value"] > 50) ) ||
	// Relative Humidity
	($dbRecord["software_channel_name"]=="RH" && ($dbRecord["software_channel_value"] < 0 || $dbRecord["software_channel_value"] > 100) ) ||
	// Atmospheric Pressure
	($dbRecord["software_channel_name"]=="AP" && ($dbRecord["software_channel_value"] < 940 || $dbRecord["software_channel_value"] > 1060) ) ||
	// Solar Radiation
	($dbRecord["software_channel_name"]=="SR" && ($dbRecord["software_channel_value"] < 0 || $dbRecord["software_channel_value"] > 1300) ) ||
```

```
        // UVA Radiation
        ($dbRecord["software_channel_name"]=="UVA" && ($dbRecord["software_channel_value"] < 0 || $dbRecord["software_channel_value"] > 80) ) ||
        // UVB Radiation
        ($dbRecord["software_channel_name"]=="UVB" && ($dbRecord["software_channel_value"] < 0 || $dbRecord["software_channel_value"] > 10) )
){

        $logpath = dirname(MODX_BASE_PATH).'/logs/lost/' . date('Y') . '/' . date('m');
        if (!is_dir($logpath)) mkdir($logpath, 0755, true);
        $log = $logpath . '/' . date('d') . ".log";
        $logdata = print_r($dbRecord, true);
        file_put_contents($log, $logdata, FILE_APPEND | LOCK_EX);

        continue;

}

$newdbrec=$modx->newObject("Weatherdata2");
$newdbrec->fromArray($dbRecord);
$newdbrec->save();

}
$modx->cacheManager->set($data[0],$data[2]);
```

```
Weatherdata2::secondaryParams( $rdata , $dbRecord );

//make daily export
Weatherdata2::makeExport();
```

4.2.2 Δευτερογενές κανάλι

Ο παρακάτω κώδικας παράγει διαφορετικά δευτερογενή κανάλια τα οποία αφορούν δεδομένα ατμοσφαιρικής πίεσης.

```
//save secondary params as channels
public static function secondaryParams( $eData , $record){
    global $modx;

    $channel = $record['software_channel_sn'];
    $date = $record['software_channel_datetime'];

    $T = $eData['AT'];
    $P = $eData['AP'];
    $RH = $eData['RH'];

    if(empty($P)) {
```

```
        if($channel == "020A0022"){
            $pressure = [];
            foreach(['041A0267',            '041A0262',
'041A0259'] as $sn){
                $query            =            $modx-
>newQuery('Weatherdata2');
                $query->where(array(
                    'software_channel_datetime:<='  =>
$date,
                    'software_channel_datetime:>='  =>
date("Y-m-d H:i:s", strtotime("-3 hours")),
                   'software_channel_name:=' => 'AP',
                   'software_channel_sn:=' => $sn
                ));

                $query-
>sortby('software_channel_datetime','DESC');
                $query->limit(1);

                if($ch            =            $modx-
>getObject('Weatherdata2', $query))
                    $ch = $ch->toArray();
                else
                    continue;
```

```
                $pressure[]                                    =
$ch['software_channel_value'];
            }

            if(empty($pressure)) return;

            $P    =    round(array_sum($pressure)    /
count($pressure), 2);

            if(empty($P)) return;
        }
        else{
            $query = $modx->newQuery('Weatherdata2');
            $sn = self::getPressure($channel);
            $query->where(array(
                    'software_channel_datetime:<='  =>
$date,
                    'software_channel_datetime:>='  =>
date("Y-m-d H:i:s", strtotime("-12 hours")),
                   'software_channel_name:=' => 'AP',
                   'software_channel_sn:=' => $sn
               ));

            $query-
>sortby('software_channel_datetime','DESC');
            $query->limit(1);
```

```
            if($ch  =  $modx->getObject('Weatherdata2',
$query))
                $ch = $ch->toArray();
            else
                return;

            $P = $ch['software_channel_value'];

            if(empty($P)) {
                return ;
            }
        }

        $record['software_channel_name'] = 'AP';
        $record['software_channel_value'] = $P;

        $newdbrec=$modx->newObject("Weatherdata2");
        $newdbrec->fromArray($record);
        $newdbrec->save();


    }

    $FP = 1.0016 + 3.15*pow(10,-6)*$P - 0.074/$P ;
```

```
    $a = 7.5;
    $b = 237.3;
    $exp = ($a*$T)/($b+$T);

    $ES = 6.112*$FP*pow(10, $exp);
    $PVAP = $RH*$ES/100;

    $AH = (216.688 * $PVAP) / ($T + 273);

    $record['software_channel_name'] = 'AH';
    $record['software_channel_value'] = $AH;



    $newdbrec=$modx->newObject("Weatherdata2");
    $newdbrec->fromArray($record);
    $newdbrec->save();

    return;
}

private static function getPressure( $channel ){
     switch ($channel) {

            case "041A0258":
                $out = '041A0264';
            break;
```

```
            case "041A0271":
                $out = '041A0265';
            break;

            case "041A0273":
                $out = '041A0270';
            break;

            case "otranto":
                $out = 'leuca';
            break;
              case "041A0270":
                $out = '041A0268';
            break;
            default:
                $out = '041A0260';

        }
    return $out;
}
} // end class Weatherdata2
```

4.2.3 Βασική κλάση αριθμητικής προεπεξεργασία και διαχείρισης δεδομένων

Η παρακάτω κλάση υλοποιήθηκε κάνοντας extend το μοντέλο δεδομένων που παρέχεται απο το xPDO και χρησιμοποιείτε απο τις πύλες DataLogger. Οι βασικές μέθοδοι παρέχονται από την κλάση είναι οι εξής:

1. Υπολογισμό δεδομένων βροχόπτωσης
2. Υπολογισμό δεδομένων ηλιοφάνειας
3. Διαμόρφωση δεδομένων ανέμου

```
///wheather data model



<?php
class Weatherdata extends xPDOSimpleObject {


public function getFieldValueSet($field, $sn=null)
{
global $modx;
$valueSet = array();
$c = $modx->newQuery($this->_class);
$c->select(array($field));
if(!is_null($sn)){
$c->where(array(
  'software_channel_sn' => $sn,
));
}
$c->query['distinct'] = 'DISTINCT';
if ($c->prepare() && $c->stmt->execute()) {
```

```
$valueSet = $c->stmt->fetchAll(PDO::FETCH_ASSOC);
}
$returnSet=array();
foreach($valueSet as $val) $returnSet[]=$val[$field];
return $returnSet;
}



public static function getType($str){
   switch ($str){
      case 'Air Tempreture':
         $type = 'Temperature';
      break;
      case 'Rain Totalize':
         $type = 'Rain Height';
      break;
      case 'Relative Humidity of Air':
         $type = 'Relative Humidity';
      break;
      case 'Solar Radiation':
         $type = 'Radiance';
      break;
      case 'UVA Radiation':
         $type = 'UVA Radiance';
      break;
      case 'UVB Radiation':
         $type = 'UVB Radiance';
```

```
        break;
        default:
            $type = $str;
    }
    return $type;
}

public static function toBeafort($val){
//return $val;
    if($val<0.6)
        $output=0;
    else if($val>=0.6 && $val<1.8)
        $output=1;
    else if($val>=1.8 && $val<3.4)
        $output=2;
    else if($val>=3.4 && $val<5.3)
        $output=3;
    else if($val>=5.3 && $val<7.5)
        $output=4;
    else if($val>=7.5 && $val<9.9)
        $output=5;
    else if($val>=9.9 && $val<12.5)
        $output=6;
    else if($val>=12.5 && $val<15.3)
        $output=7;
    else if($val>=15.3 && $val<18.3)
        $output=8;
```

```
    else if($val>=18.3 && $val<21.6)
        $output=9;
    else if($val>=21.6 && $val<25.5)
        $output=10;
    else if($val>=25.5 && $val<29.0)
        $output=11;
    else if($val>=29.0 && $val<33.0)
        $output=12;



    else
        $output="H/Cat E";

    return $output;

}

public static function getRain(){

    $now = time();
    $hour = date("Y-m-d H:i:s", ($now - (60 * 60)));

    $query = $modx->newQuery($this->_class);
    $query->where(array(
        'software_channel_name:=' => 'RT',
        'software_channel_datetime:>=' => $hour,
        )
```

```
   );
   $data = $modx->getCollection('Weatherdata2',$query);
   foreach ($data as $channel) {
      $sn = $channel->get('software_channel_sn');
      $rain[$sn] += $channel->get('software_channel_value');

   }
         $modx->setDebug(true);
                              $modx->log(modX::LOG_LEVEL_DEBUG,  "RAin:"  .
json_encode($rain));
         $modx->setDebug(false);

   return 0;
}

public static function getDN($date){
    $dt = explode(' ', $date);
    $d =  explode('-', $dt[0]);
    $t = explode(':', $dt[1]);

    $year = intval($d[0]);
    $month = intval($d[1]);
    $day = intval($d[2]);
    $hour = intval($t[0]);
    $min = intval($t[1]);

    if( ($year%4==0 && $year%100!=0) || $year%400==0)
```

```
    $M = [0,0,31,60,91,121,152,182,213,244,274,305,335];
  else
    $M = [0,0,31,59,90,120,151,181,212,243,273,304,334];

  $dn = ((($min/60)+$hour)/24) + $day + $M[$month];

  return $dn;
}
```

4.2.4 Μονάδα εξαγωγής δεδομένων

Η συγκεκριμένη μονάδα αναλαμβάνει την μορφοποίηση των δεδομένων έτσι ωστε να δημιουργήσουν ένα επεξεργάσιμο φύλλο εργασίας τύπου Excel

```
public static function makeExport(){
  global $modx;

  $dateInfo = $modx->cacheManager->get('_date');

  if ($dateInfo){
    if($dateInfo<time()){
      //make daily export
      $startingTime = time();
```

```
	$today = getdate();
		$nextExport = mktime(5, 0, 0, $today['mon'], $today['mday']+1, $today['year']);
	//$nextExport = mktime(7, 0, 0, 0, 0, $today['year']);
	$modx->cacheManager->set('_date',$nextExport);


	$modx->setDebug(true);
			$modx->log(modX::LOG_LEVEL_DEBUG, '---- starting data exporting....----'.' next export at ----> '. date('d/m/Y H:i:s',$nextExport) );
	$modx->setDebug(false);


	$today = getdate();
		$nowDate = mktime(0, 0, 0, $today['mon'], $today['mday'], $today['year']);
		$lastDate = mktime(0, 0, 0, $today['mon'], $today['mday']-1, $today['year']);


	$Date = date("Y-m-d", $lastDate);
	mkdir("assets/data/".$Date ,0755, true);


	$stations = $modx->getCollection('Stations');


	foreach ($stations as $station) {
		$where[]['OR:software_channel_sn:='] = $station->get('sn');
	}
```

```
        $datestart = date("Y-m-d H:i:s", $lastDate);
        $datestop = date("Y-m-d H:i:s", $nowDate);

        $query = $modx->newQuery('Weatherdata2');
        $query->select('Weatherdata2.*,Channels.*');
        $query->where($where);
        $query->where(array(
           'software_channel_datetime:>=' => $datestart,
           'software_channel_datetime:<' => $datestop
        ));
                                                                    $query-
>leftJoin('Channels','Channels',array('Weatherdata2.software_channel_na
me=Channels.id'));
        $query->sortby('software_channel_sn');
        $query->sortby('software_channel_datetime','ASC');

        $weatherData = $modx->getCollection('Weatherdata2',$query);

        unset($objPHPExcel);
        $currentStation = null;
        $stationDate = '';
        $exportData = array();

        //make the export array
        foreach ($weatherData as $stationData) {
           $sn = $stationData->get('software_channel_sn');
```

```
        $datetime = $stationData->get('software_channel_datetime');
        $channel = $stationData->get('software_channel_name');
        $value = $stationData->get('software_channel_value');
        $exportData[$sn][$datetime][$channel] = $value;
    }

    //make the exports (excel)
    foreach ($exportData as $currentStation => $snData) {

        $objPHPExcel = new PHPExcel();
        $objPHPExcel->getProperties()->setCreator("Ionian Weather")
            ->setLastModifiedBy("Ionian Weather")
            ->setTitle("Ionian weather")
            ->setSubject("Ionian weather")
            ->setDescription("Ionian weather")
            ->setKeywords("Ionian weather")
            ->setCategory("Ionian weather")
            ;
         $objPHPExcel->setActiveSheetIndex(0)
            ->setCellValue('A'.$i=1, 'Date')
            ->setCellValue('B'.$i, 'Time')
            ->setCellValue('C'.$i, 'DN')
            ->setCellValue('D'.$i, 'Rain')
            ->setCellValue('E'.$i, 'Vel avg')
            ->setCellValue('F'.$i, 'Vel min')
```

```
->setCellValue('G'.$i, 'Vel max')
->setCellValue('H'.$i, 'Vel sdv')
->setCellValue('I'.$i, 'Dir avg')
->setCellValue('J'.$i, 'Dir min')
->setCellValue('K'.$i, 'Dir max')
->setCellValue('L'.$i, 'Dir sdv')
->setCellValue('M'.$i, 'Pressure')
->setCellValue('N'.$i, 'Temperature')
->setCellValue('O'.$i, 'RH')
->setCellValue('P'.$i, 'Solar Irradiance')
->setCellValue('Q'.$i, 'UV-A')
->setCellValue('R'.$i, 'UV-B')
->setCellValue('S'.$i, 'PM1')
->setCellValue('T'.$i, 'PM2.5')
->setCellValue('U'.$i, 'PM4')
->setCellValue('V'.$i, 'PM10')
->setCellValue('W'.$i, 'Total')
->setCellValue('X'.$i, 'MS')
->setCellValue('Y'.$i, 'FP')
->setCellValue('Z'.$i, 'ES')
->setCellValue('AA'.$i, 'MVAP')
->setCellValue('AB'.$i, 'PVAP')
->setCellValue('AC'.$i, 'SH')
->setCellValue('AD'.$i, 'AH')
->setCellValue('AE'.$i, 'K')
```

```
                ->setCellValue('AF'.$i, 'TDEW')
                ->setCellValue('AG'.$i, 'TVIR')
                ->setCellValue('AH'.$i, 'RHO')
                ->setCellValue('AI'.$i, 'APWR')
                ->setCellValue('AJ'.$i, 'EAN')
                ->setCellValue('AK'.$i, 'SEN')
                ;
            foreach(range('A','Z') as $columnID) {
                                        $objPHPExcel->getActiveSheet()-
>getColumnDimension($columnID)->setAutoSize(true);
                $objPHPExcel->getActiveSheet()->getColumnDimension('A'.
$columnID)->setAutoSize(true);
            }

            foreach ($snData as $stationDate => $dateData) {

                $i=$i+1;
                $objPHPExcel->getActiveSheet()
                  ->setCellValue('A'.$i,  substr($stationDate,0,10) )
                  ->setCellValue('B'.$i, substr($stationDate,11,5) )
                  ->setCellValue('C'.$i,  self::getDN($stationDate)  )
                  ;
                unset($eData);

                foreach ($dateData as $channel => $value) {
                    $stVal = round($value,4);
```

```
switch($channel){
  case 'RT':
    $eData['rain'] = $stVal; break;
  case 'AT':
    $eData['temprature'] = $stVal; break;
  case 'WS':
    $eData['windSpeed'] = $stVal; break;
                                        case 'MaxWS':
    $eData['MaxWS'] = $stVal; break;
                                        case 'MinWS':
    $eData['MinWS'] = $stVal; break;
                                        case 'SDWS':
    $eData['SDWS'] = $stVal; break;
  case 'WD':
    $eData['windDir'] = $stVal; break;
                                        case 'MinWD':
    $eData['MinWD'] = $stVal; break;
                                        case 'MaxWD':
    $eData['MaxWD'] = $stVal; break;
                                        case 'SDWD':
    $eData['MaxWD'] = $stVal; break;
  case 'AP':
    $eData['pressure'] = $stVal; break;
  case 'RH':
    $eData['humidity'] = $stVal; break;
```

```
                case 'SR':
                  $eData['irradiance'] = $stVal; break;
                                                        case 'PM1':
                  $eData['PM1'] = $stVal; break;
                                                        case 'PM2.5':
                  $eData['PM2.5'] = $stVal; break;
                                                        case 'PM4':
                  $eData['PM4'] = $stVal; break;
                                                        case 'PM10':
                  $eData['PM10'] = $stVal; break;
                case 'TP':
                  $eData['total'] = $stVal; break;
                case 'UVA':
                  $eData['uva'] = $stVal; break;
                case 'UVB':
                  $eData['uvb'] = $stVal; break;

                default:
                  $eData['default'] = $stVal;
            }
        }

         if(isset($eData)){
        //make the secondary parameters
        $T = $eData['temprature'];
```

```
                $P = $eData['pressure'];
                $RH = $eData['humidity'];
                $u = $eData['windSpeed'];

                    $eData['MS'] = 3.5346 + 0.2457*$T + 0.0091*pow($T,2) +
0.0002*pow($T,3) ;
                $MS = $eData['MS'];

                $eData['MVAP'] = $RH*$MS/100;

                if($P!=0) {
                  $eData['FP'] = 1.0016 + 3.15*pow(10,-6)*$P - 0.074/$P ;
                  $FP = $eData['FP'];

                  $a = 7.5;
                  $b = 237.3;
                  $exp = ($a*$T)/($b+$T);
                  $eData['ES'] = 6.112*$FP*pow(10, $exp);
                  $ES = $eData['ES'];

                  $eData['PVAP'] = $RH*$ES/100;
                  $PVAP = $eData['PVAP'];

                  $eData['SH'] = 0.622*$PVAP/($P-$PVAP);
                  $SH = $eData['SH'];
```

```
                $eData['AH'] = (216.688 * $PVAP) / ($T + 273);

                $K = log($PVAP/(6.112*$FP));
                $eData['K'] = $K;
                $eData['TDEW'] = 243.12*$K/(17.62-$K);

                $eData['TVIR'] = (1+0.61*$SH)*$T;
                $TVIR = $eData['TVIR'];

                $Rdry = 287;
                $eData['RHO'] = ($P*100)/($Rdry*($TVIR+273));
                $RHO = $eData['RHO'];

                $eData['APWR'] = $RHO*pow($u, 3)/2;


            }
            //
            //save the line
            $objPHPExcel->getActiveSheet()
                ->setCellValue('D'.$i, $eData['rain'])
                ->setCellValue('E'.$i, $eData['windSpeed'])
                                                ->setCellValue('F'.$i,
$eData['MinWS'])
                                                ->setCellValue('G'.$i,
```

```
$eData['MaxWS'])
                                                    ->setCellValue('H'.$i,
$eData['SDWS'])
                    ->setCellValue('I'.$i, $eData['windDir'])
                                                    ->setCellValue('J'.$i,
$eData['MinWD'])
                                                    ->setCellValue('K'.$i,
$eData['MaxWD'])
                                                    ->setCellValue('L'.$i,
$eData['SDWD'])
                    ->setCellValue('M'.$i, $eData['pressure'])
                    ->setCellValue('N'.$i, $eData['temprature'])
                    ->setCellValue('O'.$i, $eData['humidity'])
                    ->setCellValue('P'.$i, $eData['irradiance'])
                    ->setCellValue('Q'.$i, $eData['uva'])
                    ->setCellValue('R'.$i, $eData['uvb'])
                                                    ->setCellValue('S'.$i,
$eData['PM1'])
                                                    ->setCellValue('T'.$i,
$eData['PM2.5'])
                                                    ->setCellValue('U'.$i,
$eData['PM4'])
                                                    ->setCellValue('V'.$i,
$eData['PM10'])
                    ->setCellValue('W'.$i, $eData['total'])
                    ->setCellValue('X'.$i, $eData['MS'])
                    ->setCellValue('Y'.$i, $eData['FP'])
```

```
                    ->setCellValue('Z'.$i, $eData['ES'])
                    ->setCellValue('AA'.$i, $eData['MVAP'])
                    ->setCellValue('AB'.$i, $eData['PVAP'])
                    ->setCellValue('AC'.$i, $eData['SH'])
                    ->setCellValue('AD'.$i, $eData['AH'])
                    ->setCellValue('AE'.$i, $eData['K'])
                    ->setCellValue('AF'.$i, $eData['TDEW'])
                    ->setCellValue('AG'.$i, $eData['TVIR'])
                    ->setCellValue('AH'.$i, $eData['RHO'])
                    ->setCellValue('AI'.$i, $eData['APWR'])

                    ;
                }
            }
               $objWriter = PHPExcel_IOFactory::createWriter($objPHPExcel,
'Excel5');
            $path = "assets/data/".$Date."/".$currentStation.".xls" ;
            $objWriter->save($path);
            unset($objPHPExcel,$objWriter);

            $modx->setDebug(true);
               $modx->log(modX::LOG_LEVEL_DEBUG, '---- data exporting....----
Just saved: '. $currentStation );
            $modx->setDebug(false);

            //break 1;
```

```
        }


        $modx->setDebug(true);
        $endingTime = time();
          $modx->log(modX::LOG_LEVEL_DEBUG, '----end data exporting....----
Completed in : '. ($endingTime-$startingTime) . ' seconds' );
        $modx->setDebug(false);


        //save daily rain value in extra channel
        $rainquery = $modx->newQuery('Weatherdata2');
        $rainquery->where(array(
           'software_channel_name:=' => 'RT',
           'software_channel_datetime:>=' => $datestart,
           'software_channel_datetime:<' => $datestop,
           'software_channel_value:>' => 0,
           )
        );


        $rain = $modx->getCollection('Weatherdata2',$rainquery);
        //$first = 1;
        foreach ($rain as $channel) {
           $sn = $channel->get('software_channel_sn');
           $total[$sn] += $channel->get('software_channel_value');
        }
```

```
        $channel_name = 'RTD';
        foreach ($total as $sn => $val) {
            $dbRecord["software_channel_sn"]=$sn;
            $dbRecord["software_channel_name"]=$channel_name;
            $dbRecord["software_channel_value"]=$val;
            $dbRecord["software_channel_datetime"]=$datestop;

            $ch = $modx->newObject('Weatherdata2');
            $ch->fromArray($dbRecord);
            $ch->save();
        }

        //$modx->cacheManager->refresh();

    }
  }
  else{
    $today = getdate();
          $nextExport = mktime(5, 0, 0, $today['mon'], $today['mday']+1,
$today['year']);
    $modx->cacheManager->set('_date',$nextExport);

    $modx->setDebug(true);
     $modx->log(modX::LOG_LEVEL_DEBUG, 'data exporting app started....
'.'setting next export at ----> '. date('d/m/Y H:i:s',$nextExport) );
    $modx->setDebug(false);
```

```
  }
  return;
}//  end function makeExport
```

5 Σύνοψη

Στην παρούσα φάση του έργου υλοποιήθηκαν και παραδόθηκαν τα ακόλουθα:

1. Διαδικτυακό σύστημα διαχείρισης δεδομένων από σταθμούς υπαίθρου
2. Περιβάλλον λειτουργίας και διαχείρισης διαδικτυακού συστήματος
3. Εφαρμοσμένο πρωτόκολλο ασφαλούς διαχείρισης δεδομένων

Στην επόμενη φάση του έργου θα ολοκληρωθεί ο έλεγχος λειτουργίας των παραπάνω μονάδων σε πραγματικές επιχειρησιακές συνθήκες του δικτύου και θα ακολουθήσει η υλοποίηση των εξής σταδίων:

1. Υλοποίηση μονάδας κώδικα ανίχνευσης και διαχείρισης εσφαλμένων τιμών και λειτουργικών προβλημάτων των αισθητήρων ή/και των μονάδων καταγραφής και επικοινωνιών του δικτύου σε πραγματικό χρόνο, με ταυτόχρονη έγκαιρη ειδοποίηση του διαχειριστή του δικτύου προς ανάληψη δράσεων αποκατάστασης τεχνικών προβλημάτων.
2. Υλοποίηση μονάδας κώδικα δυναμικής διαχείρισης καναλιών μεταγραφόμενων τιμών στον κεντρικό server και επιλογής παραμέτρων διαδικτυακής διάθεσης τιμών, μέσω παραμετρικών επιλογών του διαχειριστή.
3. Κατασκευή μονάδας κώδικα για την εξαγωγή τιμών των μετρούμενων παραμέτρων σε χρονική διακριτική ικανότητα 10-λέπτου προς συμβατότητα με το Εθνικό Δίκτυο της Εθνικής Μετεωρολογικής Υπηρεσίας και άλλα Ευρωπαϊκά Δίκτυα επίγειων σταθμών.
4. Έλεγχος επιχειρησιακής λειτουργίας του συνόλου των παραδοτέων μονάδων με αξιολόγηση λειτουργίας.
5. Απαλοιφή σφαλμάτων και αναβαθμίσεις μετά την πιλοτική λειτουργία του συστήματος.