ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ
ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
ΤΟΜΕΑΣ ΕΠΙΚΟΙΝΩΝΙΩΝ, ΗΛΕΚΤΡΟΝΙΚΗΣ ΚΑΙ
ΣΥΣΤΗΜΑΤΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ

Ολοκληρωμένο περιβάλλον ανάπτυξης οικιακής πύλης

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Σεραφείμ Σ. Παπαστέφανος

Επιβλέπων : Γεώργιος Στασινόπουλος
Καθηγητής Ε.Μ.Π.

Αθήνα, Μάιος 2005

ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ
ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
ΤΟΜΕΑΣ ΕΠΙΚΟΙΝΩΝΙΩΝ, ΗΛΕΚΤΡΟΝΙΚΗΣ ΚΑΙ
ΣΥΣΤΗΜΑΤΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ

Ολοκληρωμένο περιβάλλον ανάπτυξης οικιακής πύλης

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Σεραφείμ Σ. Παπαστέφανος

Επιβλέπων : Γεώργιος Στασινόπουλος
Καθηγητής Ε.Μ.Π.

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την ……

.....	.....	.....
Γ. Στασινόπουλος	Μ. Θεολόγου	Ν. Μήτρου
Καθηγητής Ε.Μ.Π.	Αν. καθηγητής Ε.Μ.Π.	Καθηγητής Ε.Μ.Π.

Αθήνα, Μάιος 2005

.....

Σεραφείμ Σ. Παπαστέφανος

Διπλωματούχος Ηλεκτρολόγος Μηχανικός και Μηχανικός Υπολογιστών Ε.Μ.Π.

Περίληψη

Ο σκοπός αυτής της διπλωματικής εργασίας ήταν η δημιουργία ενός ολοκληρωμένου περιβάλλοντος για την ανάπτυξη μιας οικιακής πύλης η οποία θα μπορεί να ρυθμίζεται απομακρυσμένα με χρήση του πρωτοκόλλου CPE WAN Management Protocol (πρωτόκολλο διαχείρισης συνδρομητικού εξοπλισμού δικτύου ευρείας περιοχής). Το CWMP είναι ένα νέο πρωτόκολλο που βασίζεται στην απομακρυσμένη κλήση μεθόδων μέσω των τεχνολογιών XML/SOAP. Η μεθοδολογία κατασκευής που περιγράφεται στην εργασία είναι αρκετά απλή και απαιτεί τη συγγραφή κώδικα από τον προγραμματιστή της πύλης μόνο εκεί που είναι απολύτως απαραίτητο.

Για τη δημιουργία του περιβάλλοντος γράφτηκαν τρία προγράμματα σε Java: Το DeviceDescriptor που δημιουργεί μια περιγραφή των παραμέτρων που θα έχει η συσκευή, το DeviceCreator που χρησιμοποιεί την περιγραφή του DeviceDescriptor και δημιουργεί τον πηγαίο κώδικα για το πρόγραμμα πελάτη του CWMP που θα τρέχει στη συσκευή και το SimpleACS, το οποίο είναι ένας εξυπηρετητής του CWMP (ACS), μέσω του οποίου μπορεί να γίνει η διαχείριση του προγράμματος που δημιουργεί το DeviceCreator.

Ένα αρκετά μεγάλο μέρος της εργασίας καταλαμβάνεται από την ανάλυση του προγράμματος του πελάτη του CWMP που θα τρέχει στη συσκευή και που δημιουργεί το DeviceCreator. Αυτό το πρόγραμμα είναι αρκετά περίπλοκο αφού, εκτός του ότι υλοποιεί τη στοίβα πρωτοκόλλων του CWMP και υποστηρίζει έλεγχο τύπων, ο κώδικας που κάθε φορά δημιουργείται περιλαμβάνει ένα στατικό μέρος που είναι πάντα το ίδιο και ένα δυναμικό μέρος, το οποίο αλλάζει σύμφωνα με τις επιλογές που γίνονται στα DeviceDescriptor και DeviceCreator.

Λέξεις Κλειδιά

CWMP, αυτόματη ρύθμιση, δημιουργία κώδικα, απομακρυσμένη διαχείριση, XML, SOAP, Java, C, ACS, TR-69, DSL Forum

Abstract

The scope of this thesis was the creation of a complete framework for the development of a home network gate, which will be able to be configured remotely through the CPE WAN Management Protocol (CWMP). CWMP is a new protocol, which is based in remote method invocation through the XML/SOAP technologies. The methodology described here is simple and demands code writing from the developer of the device only where it is absolutely necessary.

For the creation of the framework three Java programs were written: DeviceDescriptor which creates a description of the device's parameters, DeviceCreator which uses the description created through DeviceDescriptor and creates the source code of the CWMP client program and SimpleACS, a CWMP server program (ACS) through which the management of the program that DeviceCreator creates can be done.

A long part of this thesis is covered by the analysis of the CWMP client program which will run on the device and is created by DeviceCreator. This is a sophisticated program as it implements the protocol stack of CWMP, supports type checking and the code which is created includes a static part which is always the same and a dynamic part which changes according to the selections which are made with DeviceDescriptor and DeviceCreator.

Key Words

CWMP, auto configuration, code generation, remote management, XML, SOAP, Java, C, ACS, TR-69, DSL Forum

Περιεχόμενα

1 ΕΙΣΑΓΩΓΗ 10

1.1 Γενικα 10
1.2 Περιεχομενα της εργασιας 12
1.3 Περιεχομενα του κειμενου 18

2 ΤΟ ΠΡΟΓΡΑΜΜΑ ΤΟΥ ΠΕΛΑΤΗ 19

2.1 Σταθερες 20
2.2 Δομες 22
2.3 Επικοινωνια χαμηλου επιπεδου 24
2.4 Αλλες χρησιμες συναρτησεις 27
2.5 Parsing XML/SOAP μηνυματων 29
2.6 Δημιουργια XML/SOAP μηνυματων 32
2.7 Έλεγχος τυπων 36
2.8 Το δυναμικο τμημα του προγραμματος 40
2.9 Το κυριως τμημα του πελατη 45
2.10 Η βιβλιοθηκη IXML 55

3 ΤΟ ΠΡΟΓΡΑΜΜΑ DEVICEDESCRIPTOR 58

3.1 Η χρηση του DeviceDescriptor 58
3.2 Υλοποιηση του DeviceDescriptor 62

4 ΤΟ ΠΡΟΓΡΑΜΜΑ DEVICECREATOR 63

4.1 Η χρηση του DeviceCreator 63
4.2 Η υλοποιηση του DeviceCreator 68

5 ΤΟ ΠΡΟΓΡΑΜΜΑ SIMPLE ACS 69

5.1 Η χρηση του SimpleACS 69
5.2 Η υλοποιηση του SimpleACS 73

6 ΑΝΑΦΟΡΕΣ 74

7 ΣΥΝΤΟΜΟΓΡΑΦΙΕΣ 75

8 ΠΑΡΑΡΤΗΜΑΤΑ 76

8.1 Το makefile για την εκδοση windows 76
8.2 Το XML Schema των περιγραφων του DeviceDescriptor 77
8.3 Μετασχηματισμος XML περιγραφης συσκευης σε HTML 79
8.4 Το XML Schema για τα αρχεια του DeviceCreator 82
8.5 Μετατροπη απο SOAP Fault σε HTML 84
8.6 Ο πηγαιος κωδικας για μια απλη συσκευη 85
8.6.1 calc.dd 85
8.6.2 calc.dc 85
8.6.3 client.h 86
8.6.4 client.cpp 88
8.6.5 helpers.h 97
8.6.6 helpers.cpp 97
8.6.7 low.h 100
8.6.8 low.cpp 101
8.6.9 que.h 109

8.6.10 que.cpp 110
8.6.11 soaparse.h 111
8.6.12 soaparse.cpp 112
8.6.13 soapcrt.h 119
8.6.14 soapcrt.cpp 120
8.6.15 typechk.h 132
8.6.16 typechk.cpp 133
8.6.17 custom.h 136
8.6.18 custom.cpp 136

Εικόνες

Εικόνα 1: Εμβέλεια του CWMP *11*
Εικόνα 2: Στοίβα πρωτοκόλλων του CWMP *11*
Εικόνα 3: Η ιεραρχία αντικειμένων – μεταβλητών κατάστασης *13*
Εικόνα 4: Δημιουργία συσκευής *17*
Εικόνα 5: Ανάλυση της getMessage *26*
Εικόνα 6: Έλεγχος SetParameterValues *39*
Εικόνα 7: Δημιουργία συνάρτησης ανάθεσης *44*
Εικόνα 8: Η συνάρτηση main *46*
Εικόνα 9: Διάγραμμα ροής της handleConnection *52*
Εικόνα 10: Κλήσεις συναρτήσεων για ένα SetParameterValues μήνυμα *53*
Εικόνα 11: Η αρχική οθόνη του DeviceDescriptor *59*
Εικόνα 12: Ιδιότητες μεταβλητής κατάστασης *60*
Εικόνα 13: Εμφάνιση XML *61*
Εικόνα 14: Εμφάνιση HTML *61*
Εικόνα 15 : Η αρχική οθόνη του DeviceCreator *63*
Εικόνα 16: Επιλογή μεταβλητών *64*
Εικόνα 17: Κύρια οθόνη του DeviceCreator *65*
Εικόνα 18: Special properties *66*
Εικόνα 19: Δημιουργία συσκευής *67*
Εικόνα 20: Αρχική οθόνη του SimpleACS *69*
Εικόνα 21: Εμφάνιση παραμέτρων συσκευής *70*
Εικόνα 22: Οθόνη δημιουργίας μηνύματος ανάθεσης *71*
Εικόνα 23: Οθόνη εμφάνισης μηνύματος επιστροφής *71*
Εικόνα 24: Εμφάνιση λάθους *72*

Πίνακες

Πίνακας 1: Είδη μηνυμάτων *20*
Πίνακας 2: Είδη σφαλμάτων *20*
Πίνακας 3: Τύποι C και SOAP *36*

1 Εισαγωγή

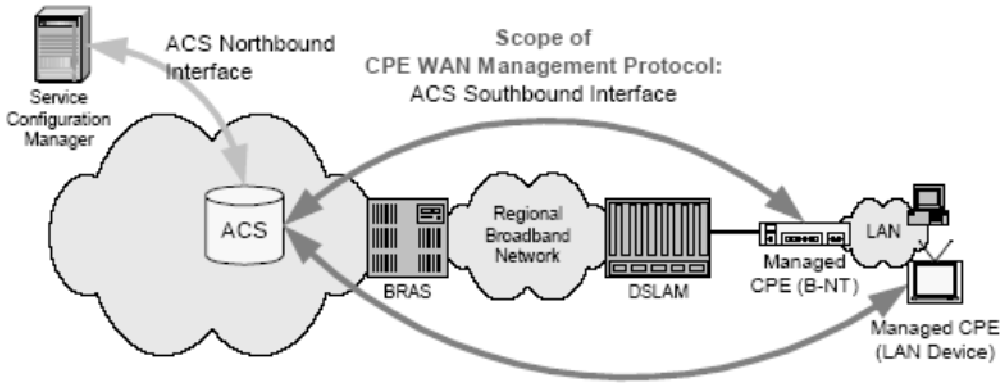
1.1 Γενικά

Η χρήση του διαδικτύου εξαπλώνεται ραγδαία τα τελευταία χρόνια. Όλο και περισσότεροι άνθρωποι θέλουν να συνδεθούν και όλο και περισσότερες εφαρμογές βασίζονται πλέον σε αυτό ενώ, για μερικές εφαρμογές δεν απαιτείται καν η ύπαρξη ηλεκτρονικού υπολογιστή με την αυστηρή έννοια του όρου μέσω του οποίου να γίνεται η σύνδεση στο διαδίκτυο. Μερικές από αυτές τις νέες εφαρμογές είναι η τηλεφωνία μέσω διαδικτύου, η τηλεόραση μέσω διαδικτύου και γενικά οι οικιακές πύλες προς το διαδίκτυο, η χρήση των οποίων θα πρέπει να είναι παρόμοια με τη χρήση των συμβατικών συσκευών και να μην απαιτεί τεχνικές γνώσεις από το χρήστη.

Για να δουλέψουν όλες αυτές οι συσκευές απαιτείται αρχικά, αλλά και καθ' όλη τη διάρκεια της λειτουργίας αυτών, η κατάλληλη ρύθμισή τους ώστε να προσφέρουν στον χρήστη τους τις υπηρεσίες που αυτός έχει πληρώσει. Αυτή η ρύθμιση δε θα πρέπει να γίνει από το χρήστη, μιας και κάτι τέτοιο θα απέτρεπε μια μεγάλη μερίδα χρηστών από τη χρήση τους, μιας και πολλές φορές απαιτεί περίπλοκες τεχνικές γνώσεις τις οποίες ο απλός χρήστης δεν έχει, ενώ δε θα πρέπει να γίνεται ούτε και από εξειδικευμένους τεχνικούς των εταιριών που προσφέρουν τις υπηρεσίες, για λόγους κόστος. Η λύση στο πρόβλημα αυτό είναι η αυτόματη ρύθμιση και διαχείριση όλων αυτών των συσκευών από απόσταση. Η αυτόματη ρύθμιση μιας συσκευής (CPE – συνδρομητικός εξοπλισμός) είναι η διαδικασία μέσω της οποίας η συσκευή παίρνει πληροφορίες ρύθμισης από το δίκτυο, έτσι, η ρύθμισή τους μπορεί να γίνει μέσα από εκεί, με τη βοήθεια πρωτοκόλλων που θα ακολουθούνται και από τη συσκευή που πρόκειται να ρυθμιστεί και από τη συσκευή ή τον άνθρωπο που θα κάνουν τις ρυθμίσεις.

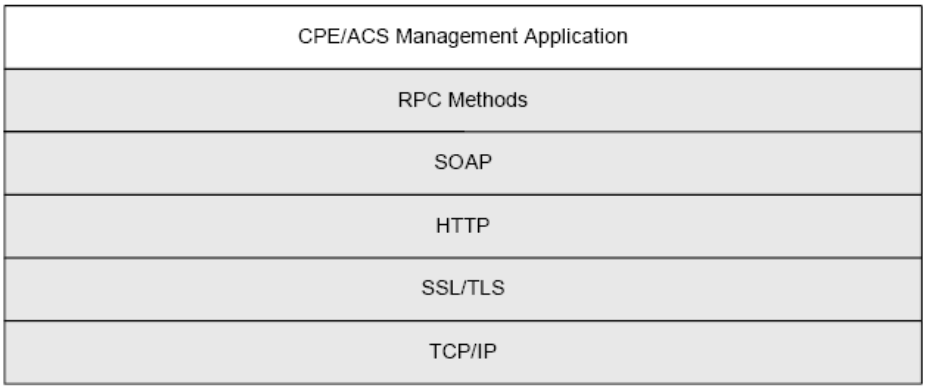
Αυτή η αυτόματη ρύθμιση των συσκευών μπορεί να χωριστεί σε τρία επίπεδα [6]: Στο χαμηλότερο πρέπει να ρυθμιστεί το στρώμα μεταφοράς, δηλαδή να υπάρχει η επικοινωνία δύο σημείων μεταξύ της συσκευής και του παρόχου της σύνδεσης, και αφορά τον NAP. Το μεσαίο στρώμα αφορά τη ρύθμιση της σύνδεσης IP μεταξύ του υπολογιστή μας και του παρόχου του δικτύου NSP. Τέλος, το υψηλότερο επίπεδο αφορά τη ρύθμιση της σύνδεσης με τον πάροχο της εφαρμογής ASP. Σε αρκετές περιπτώσεις δύο ή και οι τρεις προηγούμενοι είναι η ίδια εταιρία. Αυτά τα τρία επίπεδα θα πρέπει να ρυθμιστούν από κάτω προς τα πάνω, αφού τα υψηλότερα προϋποθέτουν την ύπαρξη των χαμηλότερων.

Ένα πρωτόκολλο που αφορά την αυτόματη ρύθμιση τρίτου επιπέδου είναι το CWMP, που περιγράφεται στο [1]. Για τη χρήση αυτού του πρωτοκόλλου πρέπει να υπάρχει σύνδεση IP της συσκευής με έναν ACS. Ο ACS είναι μια συσκευή που χρησιμοποιείται για την αυτόματη ρύθμιση άλλων συσκευών. Όπως φαίνεται και στην Εικόνα 1 ο ACS έχει δύο διεπαφές, την νότια και τη βόρεια. Η νότια συνδέεται με τη συσκευή, και η βόρεια με τον Service Configuration Manager (διαχειριστή ρυθμίσεων υπηρεσιών), ο οποίος κρατάει όλες τις πληροφορίες που χρειάζονται για την αυτόματη ρύθμιση πολλών συσκευών.



Εικόνα 1: Εμβέλεια του CWMP

Το CWMP βασίζεται στη στοίβα πρωτοκόλλων που φαίνεται στην Εικόνα 2. Στο χαμηλότερο επίπεδο υπάρχουν τα πρωτόκολλα TCP/IP και αμέσως πιο πάνω, προαιρετικά, κάποιο από τα πρωτόκολλα SSL ή TLS, τα οποία χρησιμοποιούνται για την κωδικοποίηση των δοσοληψιών δηλαδή για μεγαλύτερη ασφάλεια. Τα μηνύματα που ανταλλάσσονται μεταξύ ACS και πελάτη ακολουθούν το γνωστό πρωτόκολλο HTTP, με τον πελάτη να στέλλει HTTP posts και τον ACS να στέλλει HTTP responses. Μέσα σε κάθε HTTP μήνυμα μπορεί να υπάρχουν μία ή περισσότερες αιτήσεις για κλήση απομακρυσμένων διαδικασιών (Remote Procedure Calls) καθώς και οι απαντήσεις σε αυτές, οι οποίες κωδικοποιούνται κατάλληλα μέσω της XML και του πρωτοκόλλου SOAP (Simple Object Access Protocol).



Εικόνα 2: Στοίβα πρωτοκόλλων του CWMP

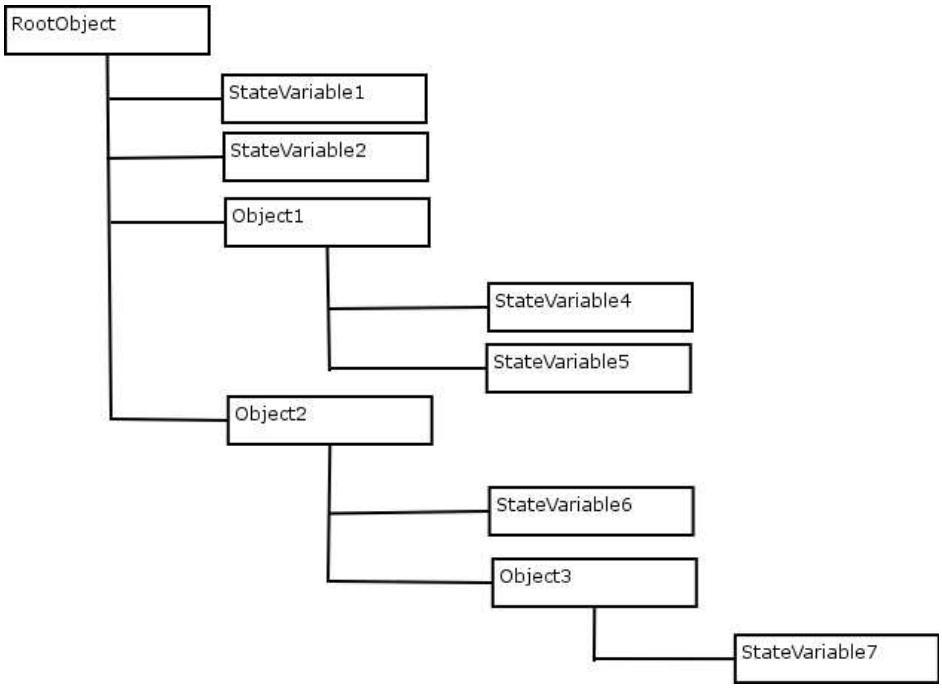
1.2 Περιεχόμενα της εργασίας

Σε αυτή την εργασία, έχουν γραφεί τρία προγράμματα στη γλώσσα προγραμματισμού Java τα οποία συνθέτουν ένα ολοκληρωμένο περιβάλλον μέσω του οποίο μπορεί να δημιουργηθεί εύκολα ένα πρόγραμμα για την αυτόματη ρύθμιση και διαχείριση μιας συσκευής που συνδέεται με το διαδίκτυο, καθώς και η διαχείριση αυτής μέσω του CWMP. Το πρόγραμμα αυτό μπορεί να τρέξει σε windows ή Unix, ανάλογα με το λειτουργικό που χρησιμοποιεί η ίδια η συσκευή.

Μια βασική έννοια στο CWMP είναι η παράμετρος, όπου κάθε παράμετρος έχει ένα όνομα και μια τιμή, των οποίων οι τιμές μπορούν να διαβαστούν ή αλλαχτούν από το πρόγραμμα ή τον άνθρωπο που κάνει τη διαχείριση με τη βοήθεια κλήσεων απομακρυσμένων μεθόδων και μέσω των οποίων μπορεί ουσιαστικά να ρυθμιστεί η συσκευή.

Αυτές οι παράμετροι του CWMP θα πρέπει να συσχετιστούν με τις πραγματικές παραμέτρους της συσκευής μέσω κώδικα που πρέπει να γραφεί στη γλώσσα C και να συμπληρωθεί στον κώδικα που δημιουργείται από τα προγράμματα Java. Μια αρχική ιδέα ήταν η δημιουργία ολόκληρου του τελικού κώδικα μέσω των προγραμμάτων, χωρίς να χρειάζεται να προστεθεί καθόλου κώδικας με το χέρι. Κάτι τέτοιο όμως θα απαιτούσε την προτυποποίηση με κάποιον τρόπο των πραγματικών παραμέτρων κάθε συσκευής από τον κατασκευαστή της, δηλαδή την προτυποποίηση των κλήσεων συστήματος προς το λειτουργικό, ώστε να γίνεται η σύνδεση των πραγματικών παραμέτρων της συσκευής με τις παραμέτρους του CWMP. Η προτυποποίηση αυτή όμως είναι αρκετά περίπλοκη και ως τώρα δεν έχει γίνει.

Μια συσκευή συμβατή με το πρωτόκολλο CWMP, αποτελείται από έναν αριθμό από αντικείμενα (objects) και μεταβλητές κατάστασης (state variables). Η μεταβλητή κατάστασης είναι μια άλλη ονομασία της παραμέτρου της συσκευής. Κάθε αντικείμενο μπορεί να περιέχει έναν αυθαίρετο αριθμό από μεταβλητές κατάστασης και άλλα αντικείμενα, ενώ ένα αντικείμενο βρίσκεται στη ρίζα της ιεραρχίας που δημιουργείται. Τα αντικείμενα ουσιαστικά χρησιμοποιούνται για την κατάλληλη ομαδοποίηση των μεταβλητών κατάστασης και δεν είναι τόσο σημαντικά στο πρωτόκολλο όσο οι μεταβλητές κατάστασης, μιας και μέσω των μεταβλητών κατάστασης γίνεται η πραγματική διαχείριση των παραμέτρων της συσκευής, και όχι μέσω τον αντικειμένων. Ένα διάγραμμα της ιεραρχίας αντικειμένων και μεταβλητών κατάστασης που δημιουργείται σε μια τυπική συσκευή του TR-69 φαίνεται στην Εικόνα 3.



Εικόνα 3: Η ιεραρχία αντικειμένων – μεταβλητών κατάστασης

Το πλήρες όνομα ενός αντικειμένου της ιεραρχίας είναι τα ονόματα των προγόνων του και το δικό του, ξεκινώντας από το αντικείμενο της ρίζας ενωμένα με τελείες «.», με μια τελεία στο τέλος. Για παράδειγμα το πήρες όνομα του αντικειμένου Object3 είναι «RootObject.Object2.Object3.». Το πλήρες όνομα μιας μεταβλητής κατάστασης είναι το ίδιο με αυτό του αντικειμένου, χωρίς όμως την τελεία στο τέλος, για παράδειγμα η μεταβλητή κατάστασης StateVariable7 θα έχει το πλήρες όνομα «RootObject.Object2.Object3. StateVariable7».

Μια ιεραρχία αντικειμένων και μεταβλητών κατάστασης όπως αυτή της Εικόνα 3 μπορεί να δημιουργηθεί με το Java πρόγραμμα **DeviceDescriptor**, το οποίο χρησιμοποιεί ένα απλό γραφικό περιβάλλον μέσω του οποίου ο χρήστης μπορεί να προσθέσει αντικείμενα και μεταβλητές κατάστασης στο ριζικό αντικείμενο, να αλλάξει το όνομα καθώς και να αλλάξει κάποια άλλα χαρακτηριστικά των μεταβλητών κατάστασης, όπως τον τύπο τους, το αν μπορούν να εγγραφούν, το αν μπορούν να διαβαστούν, το αν έχουν καθορισμένες τιμές. Η ιεραρχία από αντικείμενα και μεταβλητές κατάστασης που θα δημιουργηθεί από το DeviceDescriptor θα είναι ουσιαστικά η περιγραφή μιας συσκευής συμβατής με το TR-69. Αυτή η περιγραφή της συσκευής βρίσκεται σε ένα υψηλότερο επίπεδο, πιο αφαιρετικό, και δεν έχει καμία σχέση με τον πηγαίο κώδικα που θα υλοποιεί τη συσκευή. Ουσιαστικά η περιγραφή της συσκευής περιγράφει ένα σύνολο από παρόμοιες συσκευές (ή έναν τύπο συσκευής) οι οποίες κάνουν την ίδια εργασία, όμως είναι φτιαγμένες από διαφορετικούς κατασκευαστές ή είναι άλλα μοντέλα. Η περιγραφή της συσκευής περιγράφει κατά κάποιον τρόπο μια

ελάχιστη πρότυπη συσκευή που θα κάνει μια συγκεκριμένη εργασία, και όλες οι συσκευές που θα κάνουν την ίδια εργασία πρέπει να είναι συμβατές με αυτή την πρότυπη συσκευή, ενώ μπορούν να προσθέσουν και δικιά τους λειτουργικότητα με την προσθήκη αντικειμένων και μεταβλητών κατάστασης οριζόμενων από το χρήστη, τα οποία και θα πρέπει να έχουν ένα ιδιαίτερο όνομα όπως φαίνεται στο [1], σελίδα 52.

Έτσι, μερικά αντικείμενα / μεταβλητές κατάστασης θα είναι απαραίτητα από αυτόν τον τύπο συσκευής, ενώ μερικά άλλα θα είναι προαιρετικά, ώστε να προστεθεί κάποια προ-τυποποιημένη μεν, μη απαραίτητη δε λειτουργικότητα. Τέλος, ίσως υπάρχουν αντικείμενα και μεταβλητές κατάστασης που θα είναι εξαρτώμενα, δηλαδή θα πρέπει να υλοποιηθούν μόνο αν υλοποιηθεί ο πατέρας τους. Έτσι, μια απαραίτητη παράμετρος θα υπάρχει πάντα σε μια συσκευή που ακολουθεί κάποια συγκεκριμένη περιγραφή συσκευής, ενώ μια εξαρτημένη παράμετρος θα υπάρχει αν ο πατέρας της υλοποιείται σε αυτή τη συσκευή. Έτσι, ο πατέρας μιας εξαρτημένης μεταβλητής πρέπει να είναι προαιρετικός, ενώ ένα απαραίτητο αντικείμενο θα έχει μόνο απαραίτητα και προαιρετικά παιδιά. Οι κανόνες λοιπόν είναι:

- Ένας απαραίτητος πατέρας μπορεί να έχει απαραίτητα και προαιρετικά παιδιά
- Ένας προαιρετικός πατέρας μπορεί να έχει προαιρετικά ή εξαρτημένα παιδιά. Αν ο πατέρας υλοποιηθεί θα υλοποιηθούν και όλα τα εξαρτημένα παιδιά του.
- Ένας εξαρτημένος πατέρας μπορεί να έχει προαιρετικά ή εξαρτημένα παιδιά. Αν ο πατέρας υλοποιηθεί θα υλοποιηθούν και όλα τα εξαρτημένα παιδιά του.

Η περιγραφή μιας συσκευής μπορεί να εισαχθεί από το πρόγραμμα **DeviceCreator**. Αυτό το πρόγραμμα βρίσκεται σε χαμηλότερο επίπεδο από το DeviceDescriptor και δημιουργεί τον πηγαίο κώδικα που θα υλοποιεί μια συσκευή. Μια περιγραφή συσκευής που εισάγεται στο DeviceCreator θα καθορίσει το είδος της συσκευής που θα δημιουργηθεί. Ο χρήστης μπορεί να επιλέξει ποιες από τις παραμέτρους της περιγραφής θα υλοποιεί τελικά η συσκευή που θα δημιουργηθεί, σύμφωνα όμως με τους κανόνες που αναφέρθηκαν πριν, π.χ. οι απαραίτητες παράμετροι θα υλοποιούνται πάντοτε. Η επιλογή των παραμέτρων γίνεται μέσω ενός απλού στην όψη, αλλά αρκετά δύσκολου στην υλοποίηση user interface, το οποίο ακολουθεί τους παραπάνω κανόνες και δεν επιτρέπει στο χρήστη να κάνει λανθασμένες επιλογές.

Όταν όλες οι παράμετροι που θα περιλαμβάνει η συσκευή που δημιουργείται έχουν επιλεγεί από το χρήστη, το DeviceCreator θα περάσει στο επόμενο στάδιο, το οποίο είναι η ανάθεση συναρτήσεων. Σε κάθε μεταβλητή κατάστασης πρέπει να ανατεθεί μια συνάρτηση της C που θα επιστρέφει την τιμή της (συνάρτηση επιστροφής) ενώ σε κάθε εγγράψιμη μεταβλητή κατάστασης πρέπει να ανατεθεί μια συνάρτηση της C που θα καθορίζει την τιμή

της (συνάρτηση ανάθεσης). Η υλοποίηση του κώδικα που θα επιστρέφει και θα θέτει την τιμή σε μια μεταβλητή κατάστασης πρέπει να γραφτεί από τον χρήστη. Κάθε συνάρτηση επιστροφής / ανάθεσης μπορεί να ανατεθεί σε ένα αυθαίρετο αριθμό μεταβλητών κατάστασης, έτσι ώστε όταν κάποιες παράμετροι χρειάζονται την ίδια αρχικοποίηση, ή η τιμή τους επιστρέφεται / αλλάζει μέσω των ίδιων κλήσεων συστήματος, να έχουν την ίδια συνάρτηση επιστροφής / ανάθεσης. Ως αποτέλεσμα, η επίδοση του προγράμματος θα είναι καλύτερη και η υλοποίηση της συσκευής ευκολότερη.

Ο κώδικας για μια συνάρτηση επιστροφής / ανάθεσης που έχει ανατεθεί σε N παραμέτρους έχει ένα κοινό μέρος στο οποίο εντολές που χρειάζονται για όλες τις παραμέτρους μπορούν να εισαχθούν, και N μέρη αφιερωμένα στις μεταβλητές κατάστασης οι οποίες έχουν αυτή τη συνάρτηση.

Ο χρήστης μπορεί να αναθέσει μια διαφορετική συνάρτηση ανάθεσης / επιστροφής ή σε κάθε παράμετρο, να αναθέσει την ίδια συνάρτηση ανάθεσης / επιστροφής σε όλες τις παραμέτρους ή να χρησιμοποιήσει μια μέση λύση. Όταν τώρα η συσκευή που υλοποιείται τελικά έχει N μεταβλητές κατάστασης, εκ των οποίων οι M είναι εγγράψιμες, ο προγραμματιστής πάντοτε πρέπει να γράψει N μέρη επιστροφής τιμής και M μέρη ανάθεσης τιμής. Χρησιμοποιώντας την πρώτη επιλογή, θα υπάρχουν N συναρτήσεις επιστροφής και M συναρτήσεις ανάθεσης, κάθε μια εκ των οποίων θα έχει μόνο ένα μέρος. Αν χρησιμοποιηθεί η δεύτερη επιλογή θα υπάρχουν 1 συνάρτηση επιστροφής χωρισμένη σε N μέρη και 1 συνάρτηση ανάθεσης χωρισμένη σε M μέρη. Τέλος με την τρίτη επιλογή θα υπάρχουν n συναρτήσεις επιστροφής κάθε μια εκ των οποίων θα έχει a_i μέρη και m συναρτήσεις ανάθεσης κάθε μια εκ των οποίων θα έχει b_i μέρη, όπου $\sum_{i=1}^{n} a_i = N, \sum_{i=1}^{m} b_i = M$. Φυσικά η τρίτη επιλογή είναι αυτή που προτείνεται, ενώ η επιλογή των n, m, a_i και b_i γίνεται από τον προγραμματιστή για να εξασφαλιστεί η απόδοση και η ευκολία ανάπτυξης.

Εκτός από τις συναρτήσεις ανάθεσης / επιστροφής, ο προγραμματιστής μπορεί επίσης με τη βοήθεια του DeviceCreator να καθορίσει τις stand alone μεταβλητές κατάστασης. Μια τέτοια μεταβλητή δεν συνδέεται απευθείας με τη συσκευή, αλλά χρησιμοποιείται μόνο από το πρόγραμμα του πελάτη, επιτελώντας μια εσωτερική λειτουργία. Οι μεταβλητές αυτές έχουν το πλεονέκτημα ότι ο προγραμματιστής δε χρειάζεται να γράψει κώδικα στις συναρτήσεις ανάθεσης / επιστροφής για να καθορίσει τη συμπεριφορά τους, όπως στις μη stand alone. Αρκεί να καθορίσει το όνομα μιας μεταβλητής της C στην οποία θα αποθηκεύεται η τρέχουσα τιμή αυτής της μεταβλητής κατάστασης καθώς και την αρχική τιμή της, η οποία δε θα αλλάζει αν δεν είναι εγγράψιμη η συγκεκριμένη μεταβλητή κατάστασης.

Αφού τεθούν οι συναρτήσεις ανάθεσης / επιστροφής, οι stand alone variables, καθώς και μερικές ακόμα επιλογές που διευκολύνουν τον προγραμματιστή, ο κώδικας για τον client μπορεί να δημιουργηθεί. Ο κώδικας αυτός μπορεί να γίνει compile χωρίς την οποιαδήποτε παρέμβαση του χρήστη ώστε να προκύψει ένα πλήρες πρόγραμμα πελάτης για το CWMP το οποίο όμως θα έχει περιορισμένη λειτουργικότητα, αφού όπως αναφέρθηκε πιο πάνω, ο χρήστης πρέπει να γράψει κάποια κομμάτια κώδικα που θα συνδέουν τις παραμέτρους του CWMP με τις πραγματικές παραμέτρους της συσκευής.

Εκτός από τα προγράμματα DeviceDescriptor και DeviceCreator που ασχολούνται με τον client, έχει γραφεί και το πρόγραμμα SimpleACS, το οποίο είναι η υλοποίηση ενός αρκετά απλού ACS, ο οποίος έχει μόνο τη νότια διεπαφή και μπορεί να συνδεθεί με έναν πελάτη κάθε φορά, να βρει τα ονόματα των μεταβλητών κατάστασης του πελάτη και να εμφανίσει ένα απλό user interface στο χρήστη του, μέσω του οποίου μπορούν να διαβαστούν ή να αλλαχτούν οι τιμές όλων των παραμέτρων της συσκευής που είναι κάθε φορά συνδεδεμένη. Περισσότερα για τον τρόπο λειτουργίας του SimpleACS υπάρχουν στο αντίστοιχο τμήμα της εργασίας.

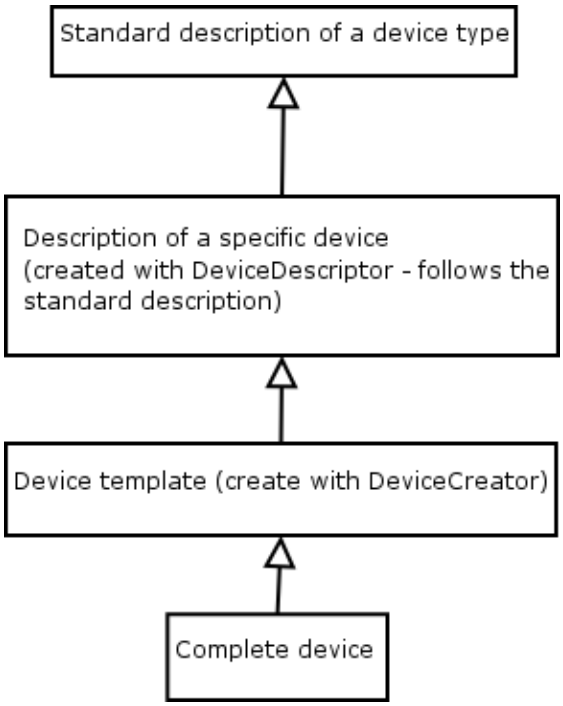
Έτσι, με τα προγράμματα DeviceDescriptor και DeviceCreator μπορεί να δημιουργηθεί ένα πλήρες πρόγραμμα πελάτη συμβατό με το TR-69 για μια συσκευή. Ο σκοπός αυτού του πελάτη είναι η σύνδεση του σε έναν ACS για αυτόματη ρύθμιση και διαχείριση. Στην Εικόνα 4 φαίνεται ένα ιεραρχικό διάγραμμα σχετικά με τα προγράμματα DeviceDescriptor και DeviceCreator. Τα επίπεδα προς την κορυφή της ιεραρχίας είναι γενικότερα και πιο απομακρυσμένα από την πραγματική συσκευή. Η πρότυπη περιγραφή μιας συσκευής πρέπει να δημοσιευθεί από κάποιον οργανισμό και θα περιγράφει τις παραμέτρους μιας μινιμαλιστικής συσκευή που θα υλοποιεί κάποια συγκεκριμένη λειτουργία. Αυτή η περιγραφή θα είναι κάποιο αρχείο XML το οποίο μπορεί είτε να δημιουργηθεί από την αρχή με το DeviceDescriptor ή κάποιο άλλο πρόγραμμα είτε ακόμα και να γραφεί με το χέρι.

Αυτό το αρχείο λοιπόν θα χρησιμοποιηθεί από το πρόγραμμα DeviceDescriptor ώστε η δημιουργία της συσκευής να προχωρήσει στο δεύτερο και λιγότερο γενικό στάδιο. Ο κατασκευαστής, με τη βοήθεια του DeviceDescriptor θα πρέπει να προσθέσει τις δικές του παραμέτρους στη συσκευή. Κάτι που πρέπει να σημειωθεί εδώ είναι ότι ακόμα και αν το αρχείο που θα περιγράφει την πρότυπη συσκευή δεν ακολουθεί το ίδιο XML Schema με τα αρχεία που διαχειρίζεται το DeviceDescriptor, εύκολα θα μπορούσε να γίνει μια μετατροπή με τη βοήθεια ενός μετασχηματισμού XSL.

Ακολούθως, η περιγραφή που ως τώρα έχει δημιουργηθεί θα πρέπει να εισαχθεί στο πρόγραμμα DeviceCreator. Εδώ θα επιλεγούν οι προαιρετικές παράμετροι που θα υλοποιεί

η τελική συσκευή και θα γίνει η ανάθεση συναρτήσεων ανάθεσης και επιστροφής, καθώς και επιλογή των stand alone μεταβλητών. Αφού γίνουν αυτά, μπορεί πλέον να δημιουργηθεί ο πηγαίος κώδικας της συσκευής.

Το μόνο που απομένει είναι να γεμίσει ο προγραμματιστής τα κενά που θα υπάρχουν στις συναρτήσεις ανάθεσης και επιστροφής, ώστε κάθε μια να επιστρέφει και να αλλάζει κατάλληλα την τιμή της παραμέτρους στην οποία έχει ανατεθεί. Όταν τελειώσει αυτή η εργασία, αρκεί να μεταγλωττιστεί ο κώδικας για να είναι έτοιμο το πλήρες πρόγραμμα πελάτης.



Εικόνα 4: Δημιουργία συσκευής

1.3 Περιεχόμενα του κειμένου

Στο ανά χείρας κείμενο, περιέχονται εκτός από την εισαγωγή, στο κεφάλαιο 2 η ανάλυση του δυναμικού προγράμματος του πελάτη στη γλώσσα προγραμματισμού C που τελικά δημιουργείται από τη χρήση των προγραμμάτων.

Στο κεφάλαιο 3 υπάρχει ένα μικρό εγχειρίδιο χρήσης για το πρόγραμμα DeviceDescriptor, στο κεφάλαιο 4 για το DeviceCreator και στο κεφάλαιο 5 για το SimpleACS. Επίσης σε αυτά τα κεφάλαια υπάρχουν και λίγα σχόλια για τον τρόπο υλοποίησης των Java προγραμμάτων, μιας και δεν περιλήφθηκε ο πηγαίος κώδικας αυτών, αφού δε θεωρήθηκε τόσο χρήσιμος μιας και ο κώδικας αυτός από τη μια είναι αρκετά εκτεταμένος και από την άλλη περιέχει κυρίως τεχνικές για την ανάπτυξη του γραφικού περιβάλλοντος των προγραμμάτων στο περιβάλλον Java swing.

Στο παράρτημα, υπάρχει ολόκληρος ο κώδικας σε C μιας απλής συσκευής, η οποία μπορεί να δημιουργηθεί πολύ γρήγορα με τα προγράμματα DeviceDescriptor και DeviceCreator και επιτελεί πράγματι κάποια απλή εργασία.

2 Το πρόγραμμα του πελάτη

Ο πελάτης του CWMP που είναι το πιο πολύπλοκο από τα προγράμματα της εργασίας και δημιουργείται δυναμικά μέσω του προγράμματος DeviceDescriptor μπορεί να χωριστεί στα ακόλουθα μέρη:

a. Επικοινωνία χαμηλού επιπέδου: Εδώ βρίσκονται οι συναρτήσεις που χρησιμοποιούνται για την επικοινωνία μεταξύ του πελάτη και του εξυπηρετητή. Μια μινιμαλιστική έκδοση του πρωτοκόλλου HTTP χρησιμοποιείται. Είναι το μόνο τμήμα του προγράμματος που δεν είναι γραμμένο σε ANSI C.
b. Χρήσιμες συναρτήσεις που δεν μπόρεσαν να περιληφθούν σε άλλο τμήμα. Εδώ υπάρχουν συναρτήσεις για διαχείριση μνήμης, για υλοποίηση μιας δομής ουράς και υποστήριξη της δομής DateTime.
c. XML/SOAP parsing και δημιουργία: Εδώ είναι συναρτήσεις και δομές για δημιουργία και parsing των XML μηνυμάτων που περιέχουν τις αιτήσεις και απαντήσεις σε SOAP.
d. Έλεγχος και parsingτύπων: Οι συναρτήσεις αυτές ελέγχουν τον τύπο των μεταβλητών κατάστασης και κάνουν parse τις τιμές τους.
e. Το δυναμικό τμήμα του πελάτη: Αυτό το τμήμα θα είναι διαφορετικό για κάθε νέο πελάτη και η μορφή του καθορίζεται από τις επιλογές που γίνονται από τα προγράμματα DeviceCreator και DeviceDescriptor.
f. Το κυρίως μέρος του πελάτη: Εδώ βρίσκεται η λογική του προγράμματος.
g. Η βιβλιοθήκη ixml: Μια ελεύθερη βιβλιοθήκη της Intel για τον χειρισμό XML.

Όλα τα μέρη του προγράμματος είναι σε καθαρή C, εκτός των συναρτήσεων επικοινωνίας χαμηλού επιπέδου, που εξαρτώνται από το λειτουργικό αφού η ANSI C δεν υποστηρίζει sockets. Για να χρησιμοποιηθεί το ίδιο αρχείο με πηγαίο κώδικα και στην έκδοση για τα windows και στην έκδοση για το Unix έχει χρησιμοποιηθεί η τεχνική των #ifdef. Αν η σταθερά _CWMP_WIN έχει γίνει #define τότε θα δημιουργηθεί η έκδοση για τα windows, αλλιώς για το Unix, για αυτό και οι δύο εκδόσεις χρησιμοποιούν διαφορετικό makefile. Το makefile για την windows έκδοση φαίνεται στο παράρτημα 8.1 · αντίστοιχο είναι και για την έκδοση για Unix.

2.1 Σταθερές

Μια σειρά από σταθερές χρησιμοποιούνται στο πρόγραμμα αυτό και καθορίζουν τον τύπο κάθε μηνύματος SOAP καθώς και το είδος για κάθε σφάλμα του πελάτη. Κάθε μια από αυτές τις σταθερές δηλώθηκε με ένα #define.

Στον Πίνακας 1 φαίνεται η αντιστοιχία μεταξύ των τύπων μηνυμάτων SOAP και των σταθερών. Οι δηλώσεις αυτές βρίσκονται στο αρχείο "client.h".

Τύπος μηνύματος	**Ορισμός**	**Τιμή**
Inform	INFORM	10
InformResponse	INFORMRESPONSE	15
GetParameterValues	GETPARAMETERVALUES	20
GetParameterValuesResponse	GETPARAMETERVALUESRESPONSE	25
SetParameterValues	SETPARAMETERVALUES	30
SetParameterValuesResponse	SETPARAMETERVALUESRESPONSE	35
GetParameterNames	GETPARAMETERNAMES	40
GetParameternamesResponse	GETPARAMETERNAMESRESPONSE	45
Not implemented method	NOTIMPLEMENTEDMESSAGE	50

Πίνακας 1: Είδη μηνυμάτων

Στον Πίνακας 2 η αντιστοιχία μεταξύ των λαθών του πελάτη και των σταθερών μπορεί να φανεί. Οι δηλώσεις αυτές βρίσκονται στο αρχείο "client.h"

Τύπος λάθους	**2.1.1.1 Ορισμός**	**Τιμή**
Method not supported	METHOD_NOT_SUPPORTED	9000
Request denied	REQUEST_DENIED	9001
Internal error	INTERNAL_ERROR	9002
Invalid arguments	INVALID_ARGUMENTS	9003
Resources exceeded	RESOURCES_EXCEEDED	9004
Invalid parameter name	INVALID_PARAMETER_NAME	9005
Invalid parameter type	INVALID_PARAMETER_TYPE	9006
Invalid parameter value	INVALID_PARAMETER_VALUE	9007
Attempt to set non writable	ATTEMPT_TO_SET_NON_WRITABLE	9008

Πίνακας 2: Είδη σφαλμάτων

Εκτός των παραπάνω, στατικών σταθερών, το πρόγραμμα χρησιμοποιεί και μια δυναμική σταθερά η οποία αναπαριστά τον αριθμό των μεταβλητών κατάστασης που έχει το πρόγραμμα, δηλαδή αλλάζει για κάθε νέο πελάτη, συμβολίζεται με NUMOFVARS και περιέχεται στο αρχείο "custom.h". Περισσότερα στο τμήμα του εγγράφου για το δυναμικό μέρος του πελάτη.

2.2 Δομές

Το πρόγραμμα χρησιμοποιεί δέκα δομές που ορίζονται στο αρχείο "client.h" με χρήση typedef:

DeviceIdStruct: Αυτή η δομή αναπαριστά ένα DeviceIdStruct, [1], Table 34. Περιέχει τέσσερις συμβολοακολουθίες στις οποίες αποθηκεύονται ο κατασκευαστής, το OUI, η κλάση παραγωγής και ο σειριακός αριθμός μιας συσκευής.

EventStruct: Αυτή η δομή αναπαριστά ένα EventStruct, [1], Table 35. Περιέχει μια συμβολοακολουθία στην οποία βρίσκεται ο λόγος σύνδεσης.

EventList: Αυτή η δομή αναπαριστά μια λίστα σταθερού μήκους από EventStructs. Περιέχει έναν απρόσημο ακέραιο για το μέγεθος της λίστας και έναν πίνακα από δομές EventStruct. Οι συναρτήσεις newEventList και freeEventList χρησιμοποιούνται μαζί με αυτή τη δομή έτσι ώστε να δημιουργηθεί μια νέα EventList ή να ελευθερωθεί ο χώρος που καταλαμβάνεται.

DateTime: Αναπαριστά τον τύπο δεδομένων dateTime από το [2]. Περιέχει έξι πεδία, για το έτος, μήνα, μέρα, ώρα, λεπτό, δευτερόλεπτο μιας ημερομηνίας.

ParameterValueStruct: Αναπαριστά ένα ParameterValueStruct, [1], Table 11. Περιέχει δύο συμβολοακολουθίες για το όνομα και την τιμή.

ParameterValueList: Αναπαριστά μια λίστα σταθερού μήκος από ParameterValueStructs. Περιέχει έναν απρόσημο ακέραιο για το μέγεθος της λίστας και έναν πίνακα από δομές ParameterValueStruct. Οι συναρτήσεις newParameterValueList και freeParameterValueList χρησιμοποιούνται με τη δομή αυτή ώστε να δημιουργηθεί μια νέα ParameterValueList ορισμένου μήκους ή να ελευθερωθεί ο χώρος που καταλαμβάνει.

StringList: Αναπαριστά μια λίστα σταθερού μήκος από συμβολοακολουθίες. Περιέχει έναν απρόσημο ακέραιο για το μέγεθος της λίστας και έναν πίνακα από συμβολοακολουθίες (δείκτες σε χαρακτήρα). Οι συναρτήσεις newStringList και freeStringList χρησιμοποιούνται με αυτή τη δομή για τη δημιουργία νέου StringList και την απελευθέρωση του χώρου που καταλαμβάνει.

SoapHeader: Αναπαριστά την επικεφαλίδα ενός μηνύματος SOAP, [1], Table 3. Περιέχει τρία πεδία: το id που είναι μια συμβολοακολουθία, το holdRequests που είναι ένας ακέραιος και το noMoreRequests που είναι επίσης ένας ακέραιος. Τα δύο τελευταία θα έπρεπε να ήταν Booleans, όμως η C δεν έχει τέτοιο τύπο και έτσι χρησιμοποιήθηκαν οι ακέραιοι.

SoapMessage: Αναπαριστά ένα μήνυμα SOAP. Περιέχει τρία πεδία: Το πρώτο είναι ένας δείκτης για μια δομή SoapHeader και καθορίζει την επικεφαλίδα του μηνύματος αυτού. Αν δεν έχει επικεφαλίδα θα είναι ένας δείκτης σε NULL. Το δεύτερο πεδίο είναι ένας ακέραιος και καθορίζει τον τύπο του μηνύματος, σύμφωνα με τον Πίνακας 1. Τέλος, το τρίτο πεδίο της δομής περιέχει μια συμβολοακολουθία, η οποία είναι ολόκληρο το κομμάτι του εγγράφου XML που περιέχει αυτό το μήνυμα SOAP (δηλαδή περιλαμβάνεται το περιεχόμενο του XML element <soap:Body>).

StateVariable: Αυτή η δομή αναπαριστά μια μεταβλητή κατάστασης. Έχει τέσσερα πεδία. Το πρώτο είναι μια συμβολοακολουθία για το πλήρες όνομα της μεταβλητής, το δεύτερο και το τρίτο είναι δείκτες για τη συνάρτηση επιστροφής και ανάθεσης της μεταβλητής. Ο τύπος τους είναι «void (*)(ParameterValueList *)» ενώ το πεδίο για τη συνάρτηση ανάθεσης θα είναι NULL αν δεν είναι εγγράψιμη. Το τελευταίο πεδίο είναι ένας ακέραιος (αντί για Boolean) ο οποίος είναι true (όχι μηδέν) όταν η μεταβλητή είναι εγγράψιμη.

2.3 Επικοινωνία χαμηλού επιπέδου

Η επικοινωνία χαμηλού επιπέδου θα αναλυθεί εδώ. Μια μινιμαλιστική έκδοση του πρωτοκόλλου HTTP/1.1 [4] έχει υλοποιηθεί και χρησιμοποιείται για την αποστολή και λήψη μηνυμάτων. Το μέρος αυτό του προγράμματος χρησιμοποιεί τις γνωστές συναρτήσεις επικοινωνίας μέσω sockets οι οποίες υπάρχουν στο Unix και τα windows.

Ο TR-69 client δουλεύει σαν ένας HTTP client όταν η επικοινωνεί με τον ACS. Όλες οι SOAP requests από τον ACS στέλλονται μέσα σε HTTP response μηνύματα, ενώ όλα τα SOAP responses από τον πελάτη μας στέλλονται μέσα σε HTTP post μηνύματα.

Επίσης, ο πελάτης δουλεύει σαν HTTP εξυπηρετητής όταν γίνεται μια σύνδεση κατ' απαίτηση από τον ACS. Ακούει συνεχώς σε μια προκαθορισμένη θύρα και όταν συνδεθεί κάποιος εκεί ο πελάτης διαβάζει από τη θύρα και εξετάζει αν έλαβε κάποια συγκεκριμένη συμβολοακολουθία. Αν την έλαβε τότε συνδέεται αμέσως με τον ACS, αν όχι απλώς κλείνει τη σύνδεση και περιμένει ξανά στη θύρα.

Η δομή SOCKET που χρησιμοποιείται εδώ είναι ένας απρόσημος ακέραιος και ο ορισμός του γίνεται στο #low.h στη Unix έκδοση ή στο winsock.h στην windows έκδοση. Οι συναρτήσεις που χρησιμοποιούνται εδώ είναι:

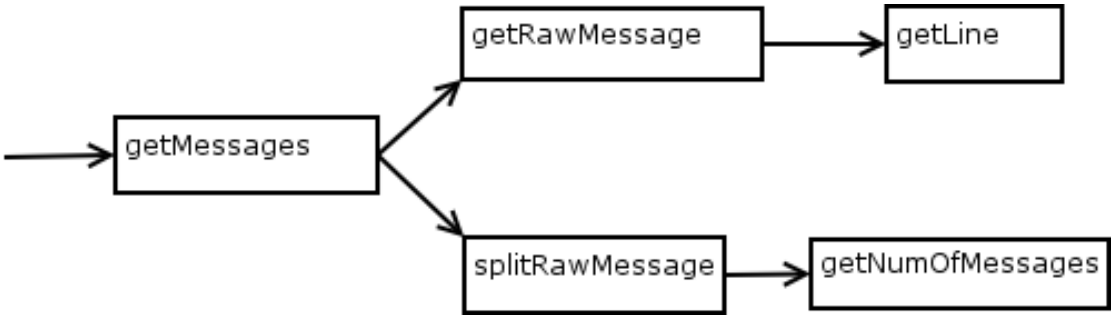
- **void initWinsock(void):** Χρησιμοποιείται μόνο στην windows έκδοση του προγράμματος και αρχικοποιεί το Winsock με χρήση της συνάρτησης WSAStartup.
- **SOCKET makeConnection(char *address, int port):** Ο πελάτης συνδέεται στη διεύθυνση TCP address:port και επιστρέφει ένα SOCKET για επικοινωνία με τη διεύθυνση αυτή. Η συνάρτηση επιστρέφει το 0 αν κάποιο λάθος συμβεί.
- **SOCKET createListeningSocket(int port):** Η συνάρτηση αυτή δημιουργεί ένα SOCKET το οποίο ακούει στη θύρα port. Χρησιμοποιείται όταν ο πελάτης δουλεύει σαν HTTP εξυπηρετητής, για συνδέσεις κατ' απαίτηση από τον ACS. Αν συμβεί λάθος επιστρέφεται το 0.
- **SOCKET acceptConnection(SOCKET lsd):** Και αυτή η συνάρτηση χρησιμοποιείται όταν ο πελάτης λειτουργεί ως εξυπηρετητής HTTP: Αποδέχεται μια σύνδεση στο listening socket lsd και επιστρέφει ένα νέο SOCKET για επικοινωνία μέσω αυτού. Το 0 θα επιστραφεί σε περίπτωση λάθους.
- **char *getLine(SOCKET sd):** Αυτή η συνάρτηση χρησιμοποιείται για την ανάγνωση μιας γραμμής από τις επικεφαλίδες ενός μηνύματος HTTP. Η συνάρτηση αυτή διαβάζει έναν έναν τους χαρακτήρες από το SOCKET και τους τοποθετεί συνεχώς σε ένα buffer, μέχρι να βρει ένα \r (CR) ακολουθούμενο από ένα \n (LF). Η

προηγούμενη ακολουθία σηματοδοτεί το τέλος μιας γραμμής επικεφαλίδων. Αν το CR δεν ακολουθείται από LF η συνάρτηση θα συνεχίσει να διαβάζει χαρακτήρες. Όταν η ακολουθία CRLF βρεθεί η γραμμή θα επιστραφεί, ενώ αν συμβεί κάποιο σφάλμα θα επιστραφεί ένας δείκτης σε NULL.

- **char *getRawMessage(SOCKET sd):** Η συνάρτηση αυτή χρησιμοποιείται για να διαβαστεί ένα πλήρες HTTP μήνυμα από το SOCKET. Αυτό το HTTP μήνυμα μπορεί να περιέχει έναν αυθαίρετο αριθμό από φακέλους SOAP (από 0, μέχρι το όριο που καθορίζει η παράμετρος MaxEnvelopes του πελάτη που στέλλεται με το μήνυμα Inform). Στην αρχή, η συνάρτηση διαβάζει συνεχώς γραμμές από το SOCKET χρησιμοποιώντας τη συνάρτηση getLine, μέχρι να βρει μια γραμμή που περιέχει τη συμβολοακολουθία "Content-Length". Αυτή η γραμμή πρέπει πάντα να υπάρχει σε ένα μήνυμα HTTP και καθορίζει το μήκος των δεδομένων του μηνύματος. Έτσι, όταν βρεθεί η γραμμή αυτή, το μήκος του σώματος του μηνύματος είναι γνωστό και αποθηκεύεται στη μεταβλητή *len*. Μετά από αυτό, η συνάρτηση συνεχίζει την ανάγνωση των γραμμών, μέχρι να βρει μια άδεια γραμμή. Μια άδεια γραμμή θα περιέχει μόνο την ακολουθία CRLF και σηματοδοτεί το τέλος των επικεφαλίδων και την αρχή του σώματος του μηνύματος, το οποίος όμως θα έχει μήκος *len*. Έτσι, πλέον αρκεί να διαβαστούν *len* χαρακτήρες από το SOCKET και να επιστραφούν. Αν συμβεί οποιοδήποτε λάθος επιστρέφεται ένας δείκτης σε NULL.
- **unsigned int getNumOfMessages(char *rawMessage):** Αυτή η συνάρτηση δέχεται ως παράμετρο ένα HTTP μήνυμα και μετράει τον αριθμό των μηνυμάτων SOAP που περιέχονται εντός αυτού. Ψάχνει το rawMessage για τις συμβολοακολουθίες <soap:Envelope και </soap:Envelope> και αυξάνει κατάλληλα τον αριθμό των μηνυμάτων SOAP.
- **StringList *splitRawMessage(char *rawMessage):** Αυτή η συνάρτηση χωρίζει το μήνυμα HTTP που δέχεται ως παράμετρο στα SOAP μηνύματα από τα οποία αποτελείται και τα επιστρέφει σε μια δομή StringList. Αρχικά μετράει τον αριθμό των μηνυμάτων SOAP με τη συνάρτηση getNumOfMessages, δημιουργεί μια δομή StringList κατάλληλου μήκους, η οποία και θα επιστραφεί, και αντιγράφει κάθε μήνυμα SOAP στην κατάλληλη θέση της λίστας.
- **StringList *getMessages(SOCKET sd):** Η συνάρτηση αυτή επιστρέφει τα μηνύματα SOAP που υπάρχουν στο SOCKET χρησιμοποιώντας τις getRawMessage και splitRawMessage. Αυτή η συνάρτηση χρησιμοποιείται και από το κυρίως πρόγραμμα κατά την ανάγνωση από το SOCKET· οι getRawMessage και splitRaw-

Message δεν πρέπει να καλούνται μόνες τους. Ο τρόπος που καλούνται οι συναρτήσεις φαίνεται στην Εικόνα 5: Η getMessages καλείται από το κυρίως πρόγραμμα, και καλεί πρώτα τη getRawMessage και μετά τη splitRawMessage. Η getRawMessage με τη σειρά της καλεί τη getLine, ενώ η splitRawMessage καλεί τη getNumOfMessages.

- **StringList *createHeaders(unsigned int msglen):** Δημιουργεί μια δομή StringList που περιέχει τις επικεφαλίδες του μηνύματος HTTP που πρόκειται να σταλεί. Μία από τις επικεφαλίδες πρέπει να είναι η Content-Length έτσι, το συνολικό μήκος του μηνύματος που πρόκειται να σταλεί πρέπει να περαστεί ως παράμετρος για να συμπληρωθεί η γραμμή αυτή.
- **void sendRawMessage(SOCKET sd, StringList *messages):** Η συνάρτηση αυτή χρησιμοποιείται για να στείλει έναν αριθμό από μηνύματα SOAP μέσω ενός HTTP post. Στην αρχή το συνολικό μήκος των μηνυμάτων SOAP υπολογίζεται και δημιουργούνται οι επικεφαλίδες του HTTP μηνύματος με την createHeaders. Στη συνέχεια όλες οι HTTP επικεφαλίδες και όλα τα μηνύματα αντιγράφονται σε έναν buffer ο οποίος και στέλλεται στο SOCKET.
- **int checkConnectionString(char *myString, SOCKET sd):** Αυτή η συνάρτηση χρησιμοποιείται στις συνδέσεις κατ' απαίτηση του ACS. Όταν ο ACS θέλει να ξεκινήσει μια σύνδεση, πρέπει να στείλει στον πελάτη ένα GET request με μια συγκεκριμένη συμβολοακολουθία. Έτσι, ο πελάτης ελέγχει την συμβολοακολουθία που έλαβε και επιστρέφει 1 αν στάλθηκε η κατάλληλη και 0 αν όχι.



Εικόνα 5: Ανάλυση της getMessage

2.4 Άλλες χρήσιμες συναρτήσεις

Χρήσιμες συναρτήσεις για ημερομηνίες: Αυτές είναι οι **char *datetoasci(DateTime *d)**, **DateTime *ascitodate(char *c)** και **DateTime *getCurrentTime(void)**. Η πρώτη συνάρτηση μετατρέπει μια δομή DateTime σε μια συμβολοακολουθία, χρησιμοποιώντας την αυστηρή μορφή YYYY-MM-DDTHH:MM:SS, π.χ. 2005-04-26T18:14:23. Η δεύτερη συνάρτηση μετατρέπει μια συμβολοακολουθία της προηγούμενης μορφής σε μια δομή DateTime, ενώ η τρίτη επιστρέφει μια δομή DateTime με την ημερομηνία του πελάτη. Η Datetoasci χρησιμοποιεί έναν στατικό buffer για να επιστρέψει τη συμβολοακολουθία που αναπαριστά τη δομή DateTime και έτσι δεν χρειάζεται να απελευθερωθεί η συμβολοακολουθία αυτή, όπως θα χρειαζόταν αν είχε χρησιμοποιηθεί μνήμη στο σωρό. Το μόνο μειονέκτημα αυτής της μεθόδου είναι ότι η συμβολοακολουθία που επιστρέφει η datetoasci πρέπει να αντιγραφεί κάπου αλλού πριν η datetoasci κληθεί για δεύτερη φορά, αλλιώς το αποτέλεσμα της δεύτερης κλήσης θα γραφεί επάνω σε αυτό της πρώτης κλήσης.

Χρήσιμες συναρτήσεις για τη διαχείριση της μνήμης. Αυτές είναι οι **ParameterValueList *newParameterValueList(unsinged int size)**, **void freeParameterValueList(ParameterValueList *)**, **EventList *newEventList(unsigned int size)**, **void freeEventList(EventList *)**, **StringList *newStringList(unsigned int size)**, **void freeStringList(StringList *)**, οι οποίες δημιουργούν μια νέα δομή ParameterValueList/EventList/StringList συγκεκριμένου μεγέθους και ελευθερώνουν τη μνήμη που καταλαμβάνεται από μια αντίστοιχη δομή. Αυτές οι συναρτήσεις χρησιμοποιούνται πάντοτε όταν οι παραπάνω δομές χρησιμοποιούνται στο πρόγραμμα.

Υλοποίηση ουράς: Η ουρά χρησιμοποιείται για να σώζει αιτήσεις από τον ACS προς τον πελάτη, οι οποίες δεν έχον ακόμα εξυπηρετηθεί και χρησιμοποιείται από τη συνάρτηση handleConnection του αρχείου "client.cpp". Μια δυναμική δομή ουράς έχει επιλεγεί επειδή οι αιτήσεις πρέπει να απαντώνται από τον πελάτη με τη σειρά που λαμβάνονται και ο αριθμός των αιτήσεων που εκκρεμούν είναι αυθαίρετος. Ουσιαστικά, αν το MaxEnvelopes του πελάτη είναι πάνω από 1, και το MaxEnvelopes του ACS είναι 1, και ο ACS στέλλει συνεχώς τόσες αιτήσεις όσες και το MaxEnvelopes του πελάτη ο αριθμός των αιτήσεων που θα εκκρεμούν θα αυξάνεται χωρίς όριο, μιας και ο πελάτης κάθε φορά θα μπορεί να απαντήσει μόνο σε μια αίτηση, όμως θα δέχεται πάνω από μία.

Δύο δομές χρησιμοποιούνται στην ουρά, η Node και η Queue. Η Node είναι ένας κόμβος της ουράς, έχει μια συμβολοακολουθία για τα δεδομένα του κόμβου και έναν δείκτη στον επόμενη κόμβο της ουράς. Η δομή Queue περιέχει δύο δείκτες, έναν για τον πρώτο και έναν για τον τελευταίο κόμβο της ουράς, και έναν ακέραιο με το μήκος της ουράς.

Οι συναρτήσεις για τη διαχείριση της ουράς είναι **enque** για την προσθήκη μιας συμβολοακολουθίας στον ουρά, η **viewQueue** για την εμφάνιση των περιεχομένων της ουράς, η **deque** για την αφαίρεση του επομένου κόμβου της ουράς, η **createQueue** για τη δημιουργία μιας νέας ουράς, η **isEmpty** για έλεγχο αν μια ουρά έχει κόμβους και η **freeQueue** για την απελευθέρωση της μνήμης που καταλαμβάνει μια ουρά.

Τέλος, άλλη μια χρήσιμη συνάρτηση είναι η **char *getErrorByNum(int num)** που λαμβάνει σαν παράμετρο έναν αριθμό σφάλματος, δηλαδή 9000, 9001 κλπ και επιστρέφει μια συμβολοακολουθία με την ερμηνεία αυτού του κωδικού. Η getErrorByNum χρησιμοποιεί έναν πίνακα από συμβολοακολουθίες ο οποίος υπάρχει στο αρχείο "helpers.cpp" για να μεταφράσει μεταξύ των μηνυμάτων και των κωδικών: απλώς αφαιρεί το 9000 από τον αριθμό που λαμβάνει και επιστρέφει την αντίστοιχη γραμμή του πίνακα. Έτσι, αν δεχτεί ως παράμετρο το 9001, θα επιστρέψει τη δεύτερη γραμμή του πίνακα (η πρώτη είναι η γραμμή 0). Η αντιστοιχία μεταξύ κωδικών λάθους και εξηγήσεων μπορεί να βρεθεί και στο κεφάλαιο των σταθερών.

2.5 Parsing XML/SOAP μηνυμάτων

Όλες οι συναρτήσεις που χρησιμοποιούνται για το parsing των μηνυμάτων SOAP παίρνουν ως παράμετρο έναν κόμβο XML που περιέχει το μέρος του μηνύματος που τις αφορά (δηλαδή τα περιεχόμενα του <soap:Body>) καθώς και μερικές άλλες παραμέτρους, στις οποίες θα επιστραφούν τα αποτελέσματα του parsing. Μια εξαίρεση στον κανόνα αυτό είναι η συνάρτηση the parseStart, η οποία δέχεται ως παράμετρο μια συμβολοακολουθία που περιέχει ολόκληρο το SOAP μήνυμα (δηλαδή ένα <soap:Envelope>) και επιστρέφει μια δομή SoapMessage που αναπαριστά το μήνυμα. Οι συναρτήσεις που χρησιμοποιούνται είναι:

- **SoapMessage *parseStart(char *buf):** Αυτή η συνάρτηση καλείται κάθε φορά που αρχίζει το parsing ενός μηνύματος SOAP. Αρχικά, δημιουργείται μια νέα δομή SoapMessage, η οποία και θα επιστραφεί αφού τελειώσει η συνάρτηση. Ακολούθως, μετατρέπεται η συμβολοακολουθία που η parseStart παίρνει ως παράμετρο σε μια δομή DOM, με τη συνάρτηση ixmlParseBufferEx της βιβλιοθήκης IXML. Στη συνέχεια, ελέγχεται το μήνυμα για την ύπαρξη SOAP επικεφαλίδας. Αν βρεθεί, γίνεται parse με τη συνάρτηση parseHeader και σώζεται στο κατάλληλο πεδίο του SoapMessage που θα επιστραφεί. Η parseStart στη συνέχεια προχωρά στο <soap:Body> και εξετάζει το παιδί του που πρέπει να είναι και το όνομα της CWMP αίτησης. Αντιγράφει ολόκληρο το υποδένδρο στο κατάλληλο πεδίο του SoapMessage που θα επιστραφεί, και εξετάζει το όνομα της αίτησης CWMP. Αν αυτό υποστηρίζεται τότε θέτει κατάλληλα την τιμή του πεδίου type του SoapMessage που θα επιστραφεί, αν αντίθετα δεν υποστηρίζεται τότε θέτει το πεδίο type στην τιμή "not implemented message". Τελικά το μήνυμα επιστρέφεται, ενώ αν συμβεί κάποιο λάθος κατά την παραπάνω διαδικασία θα επιστραφεί ένας δείκτης σε NULL.
- **int parseInform(IXML_Node *node, DeviceIdStruct *deviceId, EventList *eventList, unsigned int *maxEnvelopes, DateTime *currentTime, unsigned int *retryCount, ParameterValueList *parameterList):** Αυτή η συνάρτηση κάνει parse ένα μήνυμα Inform που περιέχεται στον κόμβο που δέχεται ως παράμετρο και αποθηκεύει τις τιμές αυτού του μηνύματος στις άλλες παραμέτρους. Τα EventList * και ParameterValueList * που περνιούνται ως παράμετροι πρέπει να έχουν μηδενικό μέγεθος. Το όνομα του κόμβου πρέπει να είναι cwmp:Inform. Αυτή η συνάρτηση και όσες ακολουθούν επιστρέφουν το –1 αν συμβεί κάποιο λάθος. Στην πραγματικότητα, κάθε λάθος που συμβαίνει κατά το parsing των παραμέ-

τρων θα συμβαίνει επειδή οι παράμετροι της μεθόδου SOAP δεν ήταν οι αναμενόμενες, έτσι η συνάρτηση που καλεί τις parsing συναρτήσεις θα ξέρει ότι το λάθος που έγινε ήταν το «invalid parameters» και θα δημιουργήσει SOAP fault απάντηση για αυτό.

- **int parseInformResponse(IXML_Node *node, unsigned int *maxEnvelopes):** Αυτή η συνάρτηση κάνει parse το InformResponse μήνυμα και δεν χρειάζεται στον πελάτη, μόνο στον ACS. Ο κόμβος που δέχεται ως παράμετρος πρέπει να έχει το όνομα cwmp:InformResponse.
- **int parseSetParameterValues(IXML_Node *node, ParameterValueList *parameterList, char **parameterKey):** Αυτή η συνάρτηση κάνει parse το μήνυμα SetParameterValues. Τα ζευγάρια παραμέτρων – τιμών που περιέχονται στο μήνυμα θα τοποθετηθούν στην παράμετρο parameterList, ενώ τα περιεχόμενα του στοιχείου ParameterKey θα τοποθετηθούν στην παράμετρο parameterKey. Ο κόμβος που δέχεται η συνάρτηση ως παράμετρο θα πρέπει να είναι ένας κόμβος cwmp:SetParameterValues.
- **int parseSetParameterValuesResponse(IXML_Node *node, unsigned int *status):** Αυτή η συνάρτηση κάνει parse το μήνυμα SetParameterValuesResponse και δε χρησιμοποιείται στον πελάτη. Ο κόμβος που δέχεται ως παράμετρο θα πρέπει να είναι ένας κόμβος cwmp:SetParameterValuesResponse.
- **int parseGetParameterValues(IXML_Node *node, StringList *parameterNames):** Αυτή η συνάρτηση κάνει parse το μήνυμα GetParameterValues. Τα ονόματα των παραμέτρων των οποίων οι τιμές ζητούνται επιστρέφονται μέσω της parameterNames. Ο κόμβος που δέεται ως παράμετρο αυτή η συνάρτηση πρέπει να είναι ένας κόμβος cwmp:GetParameterValues.
- **int parseGetParameterValuesResponse(IXML_Node *node, ParameterValueList *parameterList):** Αυτή η συνάρτηση κάνει parse ένα μήνυμα GetParameterValuesResponse και δε χρησιμοποιείται από τον πελάτη. Ο κόμβος που δέχεται ως παράμετρο πρέπει να είναι ένας κόμβος cwmp:GetParameterValuesResponse.
- **int parseGetParameterNames(IXML_Node *node, char **pathName, unsigned short *nextLevel):** Αυτή η συνάρτηση κάνει parse το μήνυμα GetParameterNames. Τα pathName και nextLevel θα έχουν τις τιμές των αντίστοιχων παραμέτρων του μηνύματος SOAP. Ο κόμβος που δέχεται ως παράμετρο αυτή η συνάρτηση πρέπει να είναι ένας κόμβος cwmp:GetParameterNames.

- **int parseGetParameterNamesResponse(IXML_Node *node, StringList parameterNames):** Αυτή η συνάρτηση κάνει parse το μήνυμα GetParameter-NamesResponse και δε χρησιμοποιείται από τον πελάτη. Ο κόμβος που δέχεται ως παράμετρο είναι ένας cwmp:GetParameterNamesResponse κόμβος.
- **SoapHeader *parseHeader(IXML_Node *node):** Τέλος, αυτή η συνάρτηση κάνει parse έναν κόμβο με την επικεφαλίδα ενός μηνύματος SOAP, δηλαδή έναν κόμβο <soap:Header>. Επιστρέφει μια δομή SoapHeader με τα στοιχεία του κόμβου αυτού.

2.6 Δημιουργία XML/SOAP μηνυμάτων

Τα μηνύματα SOAP δημιουργούνται με βάση τις οδηγίες του [1] που βασίζονται στις κατευθύνσεις του [5].

Όλες οι συναρτήσεις που χρησιμοποιούνται για τη δημιουργία SOAP μηνυμάτων δέχονται ως παράμετρο μια δομή SoapHeader και μερικές ακόμα παραμέτρους που αφορούν το είδος του μηνύματος που πρόκειται να δημιουργηθεί και επιστρέφουν ένα IXML_Document με το πλήρες μήνυμα. Η παράμετρος SoapHeader θα μετατραπεί στην επικεφαλίδα του μηνύματος, αν είναι NULL το μήνυμα δε θα έχει επικεφαλίδα. Οι συναρτήσεις για τη δημιουργία SOAP μηνυμάτων είναι οι εξής:

- **IXML_Document createInform(SoapHeader *sh, DeviceIdStruct deviceId, EventList *events, unsigned int maxEnvelopes, DateTime curentTime, unsignedInt retryCount. ParameterValueList *parameterList):** Δημιουργεί το μήνυμα Inform. Το Inform είναι το μήνυμα που ο πελάτης στέλλει στον ACS στην αρχή κάθε σύνδεσης μεταξύ τους. Οι παράμετροι της συνάρτησης έχουν να κάνουν με τις παραμέτρους του μηνύματος Inform και είναι οι deviceId, για μια δομή DeviceIdStruct που περιέχει τα χαρακτηριστικά της συσκευής, events για μια λίστα από EventStructs που περιέχει τους λόγους που συνδέθηκε ο client στον ACS, maxEnvelopes που περιέχει το μέγιστο αριθμό μηνυμάτων SOAP που μπορεί να λάβει ο πελάτης σε ένα μόνο HTTP μήνυμα από τον ACS(αυτό το νούμερο είναι αυθαίρετο στον πελάτη αυτό), currentTime που είναι μια δομή DateTime και περιέχει την τρέχουσα ώρα του πελάτη, retryCount που αναφέρει τον αριθμό των προσπαθειών που έκανε ο πελάτης μέχρι να καταφέρει να συνδεθεί και parameterList που είναι μια λίστα από δομές ParameterValueList και περιέχει τα ονόματα και τις τιμές των παραμέτρων που πρέπει να στέλλονται στον ACS κάθε φορά που ο πελάτης συνδέεται.
- **IXML_Document *createInformResponse(SoapHeader *sh, unsigned int maxEnvelope):** Δημιουργεί ένα InformResponse μήνυμα. Δε χρησιμοποιείται από τον πελάτη. Το maxEnvelope έχει να κάνει με το μέγιστο αριθμό των SOAP envelopes που μπορεί να χειριστεί ο ACS.
- **IXML_Document *createGetParameterValues(SoapHeader *sh, StringList *parameterNames):** Δημιουργεί το μήνυμα GetParameterValues. Δε χρησιμοποιείται από τον πελάτη. Η παράμετρος είναι μια λίστα από συμβολοακολουθίες που περιέχουν τα ονόματα των παραμέτρων των οποίων οι τιμές ζητούνται.

IXML_Document *createGetParameterValuesResponse (SoapHeader *sh, ParameterValueList *parameterList): Δημιουργεί το μήνυμα ParameterValuesResponse. Η παράμετρος parameterList είναι μια λίστα με ζεύγη ονομάτων και τιμών μεταβλητών κατάστασης που θα περιέχει το μήνυμα που θα δημιουργηθεί.

- **IXML_Document *createSetParameterValues(SoapHeader *sh, ParameterValueList *parameterList, char *parameterKey):** Αυτή η συνάρτηση δημιουργεί το μήνυμα SetParameterValues και χρησιμοποιείται από τον ACS. Οι παράμετροι της αντιστοιχούν στις παραμέτρους του μηνύματος SetParameterValues.
- **IXML_Document *createSetParameterValuesResponse(SoapHeader *sh, int status):** Αυτή η συνάρτηση δημιουργεί το μήνυμα SetParameterValuesResponse. Η παράμετρος status είναι ένας ακέραιος ο οποίος περιέχει την παράμετρο status του SOAP μηνύματος. Αν έχει την τιμή 0 τότε όλες οι παράμετροι πήραν τις νέες του τιμές, ενώ αν έχει την τιμή 1 τότε κάποιες παράμετροι δεν πήραν τις νέες τους τιμές, αλλά θα τις πάρουν μετά από επανεκκίνηση η οποία θα γίνει όταν λήξει η δοσοληψία μεταξύ του πελάτη και το ACS.
- **IXML_Document *createGetParameterNames(SoapHeader *sh, char *parameterPath, unsigned short int nextLevel):** Δημιουργεί το GetParameterNames μήνυμα, δε χρησιμοποιείται από τον πελάτη.
- **IXML_Document *createGetParameterNamesResponse(SoapHeader *sh, char *path, unsigned short nextLevel):** Δημιουργεί το μήνυμα GetParameterNamesResponse. Παίρνει ως παραμέτρους τη συμβολοακολουθία path η οποία πρέπει να περιέχει είτε ένα πλήρες όνομα μεταβλητής κατάστασης είτε ένα πλήρες όνομα αντικειμένου (άρα να τελειώνει με «.»). Αν το path είναι κενό τότε αναφέρεται σε ολόκληρη την ιεραρχία. Το nextLevel τώρα θα είναι είτε 0 είτε 1 (boolean). Αν είναι 0 τότε η απάντηση θα περιέχει όλες τις μεταβλητές κατάστασης το όνομα των οποίων αρχίζει με το path. Αν είναι 1, η απάντηση θα περιέχει μόνο τα αντικείμενα που είναι παιδιά του path.
- **IXML_Document *createFaultResponse(SoapHeader *sh, int code):** Δημιουργεί ένα μήνυμα SOAP fault. Δέχεται ως παράμετρο τον κωδικό του σφάλματος και με τη βοήθεια της συνάρτησης getErrorByNum του «helpers.cpp» δημιουργεί και την επεξήγηση του λάθους που επίσης χρειάζεται να περιληφθεί στο μήνυμα fault.
- **IXML_Document *createSetFaultResponse(SoapHeader *sh, ParameterValueList *params, int * errors):** Αυτή η συνάρτηση δημιουργεί ένα SetParame-

ters fault μήνυμα. Οι παράμετροι που δέχεται είναι η δομή ParameterValueList που έγινε parse από το μήνυμα SetParameterValues και ο πίνακας ακεραίων error (ο οποίος θα έχει δημιουργηθεί με τη βοήθεια της checkSetParameters), που περιέχει τον κωδικό σφάλματος για κάθε μια από τις παραμέτρους της λίστας params. Αν δεν ανιχνεύθηκε λάθος σε κάποια από τις παραμέτρους θα υπάρχει το 0 στην αντίστοιχη θέση του πίνακα error, ενώ αν βρέθηκε λάθος θα υπάρχει ένας αρνητικός αριθμός με το είδος του σφάλματος. Έτσι, για κάθε μια από τις παραμέτρους του SetParameterValues που είχαν λάθος θα δημιουργηθεί ένα SetParameterValuesFault element ([1], page 20) στο SOAP Fault μήνυμα που θα επιστραφεί. Περισσότερες πληροφορίες για τον έλεγχο λαθών μπορούν να βρεθούν στην αντίστοιχη ενότητα.

- **IXML_Node *createHeader(IXML_Document* doc, SoapHeader *sh):** Αυτή η συνάρτηση μετατρέπει τη δομή SoapHeader που δέχεται ως παράμετρο σε έναν κόμβο XML (δομή IXML_Node) που ανήκει στο έγγραφο XML doc (δομή IXML_Document). Χρησιμοποιείται από τις προηγούμενες συναρτήσεις όταν το μήνυμα που θα δημιουργηθεί περιέχει και επικεφαλίδα.

Εκτός από τις παραπάνω κύριες συναρτήσεις, υπάρχουν και αρκετές βοηθητικές συναρτήσεις οι οποίες χρησιμοποιούνται από τις κύριες συναρτήσεις. Αυτές οι βοηθητικές συναρτήσεις είναι οι ακόλουθες: **IXML_Element *createDeviceId(IXML_Document* doc, DeviceIdStruct *deviceId)** που μετατρέπει μια δομή DeviceIdStruct σε κόμβο XML, **IXML_Element *createEvent(IXML_Document* doc, EventList *events)** που μετατρέπει μια δομή EventList σε έναν κόμβο XML, **IXML_Element *createMaxEnvelope(IXML_Document* doc, unsigned int maxEnvelope)** που δημιουργεί ένα element MaxEnvelopes με βάση την ακέραια παράμετρο που δέχεται η συνάρτηση, **IXML_Element *createDateTime(IXML_Document* doc, DateTime *currentTime)** που δημιουργεί ένα CurrentTime element με βάση την παράμετρο, **IXML_Element *createRetryCount (IXML_Document* doc, unsigned int retryCount)** που δημιουργεί το RetryCount element με βάση την παράμετρο (όλες οι προηγούμενες συναρτήσεις χρησιμοποιούνται από την createInform, **IXML_Element *createParameterList(IXML_Document* doc, ParameterValueList *parameterList)** που μετατρέπει μια δομή ParameterList σε XML element και χρησιμοποιείται από την createGetParameterValues και την createInform, **IXML_Element *createStatus(IXML_Document* doc, unsigned int status)**, που δημιουργεί ένα

status element και τέλος **IXML_Element *createParameterNames (IXML_Document *doc, StringList *parameterNames)** που δημιουργεί ένα XML element που περιέχει τις συμβολοακολουθίες της δομής StringList parameterNames.

2.7 Έλεγχος τύπων

Αυτό το πρόγραμμα πρέπει να ελέγχει τον τύπο όλων των τιμών που λαμβάνει σε ένα μήνυμα SetParameterValues πριν αναθέσει τις τιμές αυτές στις αντίστοιχες μεταβλητές κατάστασης. Αυτές οι τιμές πρέπει να έχουν τον ίδιο τύπο με την μεταβλητή κατάστασης της οποίας την τιμή περιέχουν. Μερικές άλλες παράμετροι μπορούν να λάβουν μόνο ένα συγκεκριμένο σύνολο επιτρεπόμενων τιμών ενώ κάποιος άλλες λαμβάνουν ένα εύρος τιμών. Όλες αυτές οι επιλογές καθορίζονται με τη βοήθεια του προγράμματος DeviceDescriptor.

Στο TR-69 χρησιμοποιούνται οι ακόλουθοι τύποι δεδομένων: Int, unsignedInt, dateTime, boolean, string, base64 και any. Αν ο τύπος string και base64 ακολουθείται από έναν αριθμό εντός παρενθέσεως τότε το μέγιστο μήκος του μπορεί να είναι όσο ο αριθμός, ενώ αν δεν ακολουθείται από αριθμό τότε το μέγιστο μήκος του θα είναι 16, δηλαδή το string είναι το ίδιο με το string(16). Οι προηγούμενοι τύποι σχετίζονται με τους τύπος της C όπως φαίνεται στον Πίνακας 3:

SOAP datatype	**C datatype**
int	long int[1]
unsignedInt	unsigned long int[2]
dateTime	struct DateTime
boolean	int[3]
string	char *
base64	char *
any	char *

Πίνακας 3: Τύποι C και SOAP

Κωδικοί σφαλμάτος

Όπως φαίνεται στην ενότητα για τις σταθερές, ένας αριθμός από κωδικούς σφάλματος έχει οριστεί, όπως υπάρχουν και στο Table 58 του [1]. Όλες οι συναρτήσεις ελέγχου θα επιστρέψουν μια αρνητική τιμή όταν βρουν σφάλμα. Έτσι, όταν μια παράμετρος έχει λάθος τύπο τότε θα επιστραφεί το -(πλην) INVALID_PARAMETER_TYPE, οπότε η συνάρτηση που κάλεσε τη συνάρτηση ελέγχου θα μπορέσει να ελέγξει για αρνητική επιστρεφόμενη τιμή και αν βρει τότε θα πρέπει απλώς να πολλαπλασιάσει αυτή την τομή με το –1 και έτσι

[1] Ο τύπος int του SOAP είναι ένας 32μπιτος ακέραιος. Όμως, η ANSI C δε μας εξασφαλίζει ότι ο τύπος long int αυτής είναι πράγματι ένας 32μπιτος ακέραιος. Το μόνο που μας διασφαλίζει η ANSI C είναι ότι ο τύπος long int θα είναι μεγαλύτερος ή ίσος από τον int. Οι περισσότερες υλοποιήσεις πάντως χρησιμοποιούν 4 bytes για την αναπαράσταση του long int. Κανονικά θα έπρεπε να είχαν χρησιμοποιηθεί τύποι που ορίζει ο κάθε compiler της C και εξασφαλίζει ότι θα είναι 4 bytes.

[2] Ισχύουν τα ίδια με τον int.

[3] Θα μπορούσε να είχε χρησιμοποιηθεί και short int.

θα έχει τον κωδικό σφάλματος. Με αυτό τον κωδικό σφάλματος η συμβολοακολουθία που περιγράφει το σφάλμα μπορεί να βρεθεί μέσω της getErrorByCode, και έτσι να δημιουργηθεί εύκολα το κατάλληλο SetParameterFault SOAP μήνυμα.

Μια κεντρική συνάρτηση για τον έλεγχο τύπων είναι η **int checkSetParameterValues(ParameterValueList *parameterList, int *errors)**. Αυτή η συνάρτηση εκτελείται πάντοτε αφού ληφθεί ένα μήνυμα SetParameterValues και πριν ανατεθούν οι νέες τιμές στις παραμέτρους. Οι παράμετροι της συνάρτησης είναι η λίστα των δομών ParameterValueStruct που έγινε parse από το μήνυμα SetParameterValues και ένας πίνακας ακεραίων που έχει το ίδιο μέγεθος με τη λίστα των παραμέτρων. Αυτή η συνάρτηση ελέγχει αν, όλα τα ζεύγη ονομάτων και τιμών των παραμέτρων που δέχεται, είναι εγγράψιμα ή όχι (αν δεν είναι εγγράψιμα δε χρειάζεται να πάμε στο επόμενο βήμα) και στη συνέχεια χρησιμοποιώντας τη δυναμική συνάρτηση checkParameter του «custom.cpp» και θα εξηγηθεί στην κατάλληλη ενότητα. Η συνάρτηση checkParameter χρησιμοποιεί τις συναρτήσεις ελέγχου τύπων που ακολουθούν, έτσι ώστε να ελέγξει αν κάθε μία από τις παραμέτρους που λαμβάνει έχει τον σωστό τύπο και επιστρέφει το 0 αν έχει το σωστό τύπο, η μια αρνητική τιμή με τον κατάλληλο τύπο αν βρει σφάλμα. Αυτή η αρνητική τιμή θα δοθεί στην αντίστοιχη θέση στον πίνακα errors.

Έτσι, η checkSetParameterValues ελέγχει όλα τα ζεύγη ονομάτων / τιμών παραμέτρων της λίστας που παίρνει ως παράμετρος. Αν βρεθούν λάθη θα επιστραφεί μια αρνητική τιμή, αλλιώς θα επιστραφεί το 0. Αν τώρα επιστραφεί αρνητική τιμή, η συνάρτηση που την κάλεσε θα πρέπει να ελέγξει τον πίνακα errors ώστε να βρει ποιες από τις παραμέτρους είχαν σφάλμα και τι ήταν αυτό. Αν επιστραφεί το 0 τότε ο πίνακας errors θα είναι και αυτός γεμάτος με 0, αφού δε θα υπάρχει πουθενά λάθος.

Οι συναρτήσεις ελέγχου τύπων που χρησιμοποιεί η checkParameter λειτουργούν ως εξής: Αρχικά ελέγχουν αν η παράμετρος που δέχονται και είναι μια συμβολοακολουθία είναι δείκτης σε NULL, και επιστρέφουν σφάλμα αν συμβεί αυτό. Στην συνέχεια ελέγχουν τη συμβολοακολουθία και επιστρέφουν το 0 αν δεν βρουν κάποιο σφάλμα, η μια αρνητική τιμή με τον κωδικό του σφάλματος.

- **int checkString(char *buf, unsigned int len):** Η συνάρτηση αυτή ελέγχει αν τα περιεχόμενα του buf είναι μια συμβολοακολουθία με μέγιστο μήκος len.
- **int checkInt(char *buf):** Αυτή η συνάρτηση ελέγχει αν τα περιεχόμενα του buf είναι ένας 32μπιτος ακέραιος. Η συνάρτηση αυτή θα επιστρέψει σφάλμα αν τα περιεχόμενα του buf δε συμφωνούν με την κανονική έκφραση [\t]*(+|-)?[0-9]+[\t]*

η οποία σημαίνει αυθαίρετος αριθμός κενών και tab, ένα προαιρετικό «+» ή «-», ένα η περισσότερα ψηφία και ένας αυθαίρετος αριθμός κενών και tab στο τέλος. Επίσης, η τιμή ελέγχεται έτσι ώστε να μην είναι υπερβολικά μεγάλη ή υπερβολικά μικρή και δε χωράει στα 32 bit του ακεραίου.

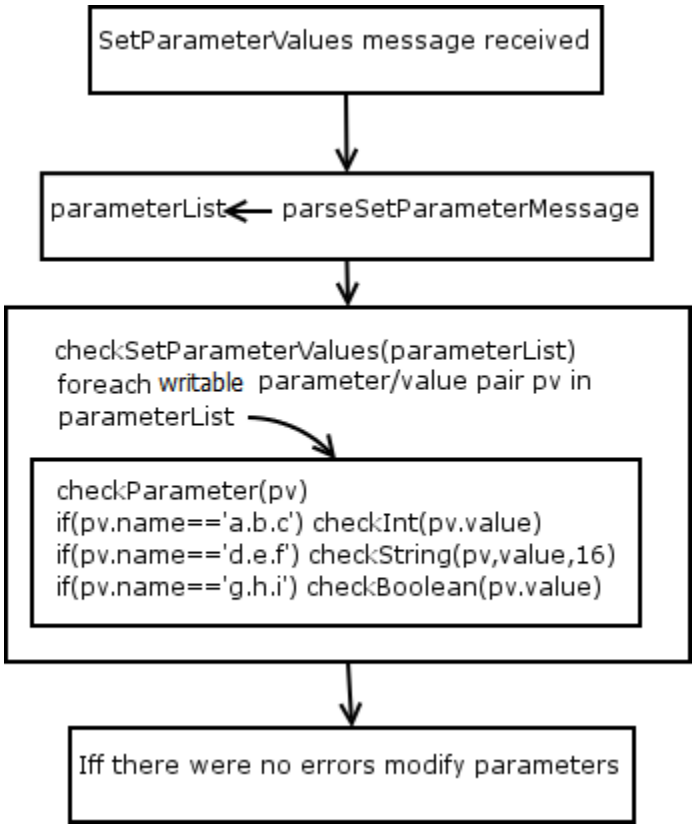
- **int checkUnsignedInt(char *buf):** Δουλεύει το ίδιο με την checkInt, αλλά δεν ελέγχει για πρόσημο. Μια κανονική έκφραση που ακολουθείται είναι η επόμενη: [\t]* [0-9]+[\t]*.
- **int checkBoolean(char *buf):** Ελέγχει τον buf για έναν boolean. Μπορεί να υπάρχει αυθαίρετος αριθμός κενών και tab στην αρχή και το τέλος, και ένα «1» ή ένα «0» ανάμεσα, δηλαδή η κανονική έκφραση είναι [\t]*(0|1)[\t]*.
- **int checkDateTime(char *buf):** Ελέγχει αν στο buf υπάρχει ο τύπος dateTime. Μια κανονική έκφραση είναι [\t]*dddd'-'dd'-'dd'T'dd':'dd':'dd[\t], όπου το d είναι ένα δεκαδικό ψηφίο. Οι λανθασμένες ημερομηνίες, (όπως η 2004-15-55T33:99:84) δεν αναγνωρίζονται από αυτή την κανονική έκφραση (στην πραγματικότητα θα ήταν αρκετά δύσκολο να γραφεί μια κανονική έκφραση που να ταιριάζει μόνο με τις σωστές ημερομηνίες).
- **int checkBase64(char *buf, unsigned int len):** Ελέγχει για τον τύπο base64 με μέγιστο μήκος len. Μια κανονική έκφραση για τους ελέγχους που κάνει η συνάρτηση είναι η [a-zA-Z0-9+/]+ δηλαδή λαμβάνει υπ' όψιν και τα κενά ή tab που τυχόν υπάρχουν πριν τον base64.

Οι ακόλουθες είναι οι συναρτήσεις που κάνουν parse τις τιμές που περιέχονται στον buffer που δέχονται ως παράμετρο σε μια άλλη μεταβλητή του κατάλληλου τύπου. Είναι πολύ απλές αφού δεν κάνουν καθόλου ελέγχους, έτσι πριν κληθεί μια συνάρτηση parse θα πρέπει να έχει κληθεί η αντίστοιχη συνάρτηση ελέγχου τύπου.

- **long int parseInt(char *buf):** Κάνει parse το buf και επιστρέφει έναν ακέραιο.
- **unsigned long int parseUnsignedInt(char *buf):** Επιστρέφει έναν απρόσημο ακέραιο που πρέπει να υπάρχει στον buf.
- **int parseBoolean(char *buf):** Επιστρέφει έναν boolean που υπάρχει στον buf.
- **DateTime parseDateTime(char *buf):** Επιστρέφει μια δομή DateTime που υπάρχει στον buf..

Στο διάγραμμα της Εικόνα 6 φαίνεται η διαδικασία για τον έλεγχο τύπων: Όταν ληφθεί ένα μήνυμα SetParameterValues τα ζεύγη με τα ονόματα των παραμέτρων και τις νέες

τους τιμές θα γίνουν parse από την parseSetParameterValues και θα αποθηκευθούν σε μια δομή ParameterList. Αυτή η δομή θα περαστεί ως παράμετρος στη συνάρτηση checkSetParameterValues (μαζί με τον πίνακα errors στον οποίο θα αποθηκευθούν τυχόν λάθη), και για κάθε ένα ζεύγος ονόματος και τιμής παραμέτρου η συνάρτηση checkParameter θα κληθεί. Η checkParameter που ανήκει στο «custom.cpp» με τη σειρά της, θα καλέσει την κατάλληλη συνάρτηση ελέγχου τύπου, ανάλογα με τον πραγματικό τύπο της κάθε παραμέτρου. Αν δε συμβούν σφάλματα κατά τη διαδικασία των ελέγχων οι νέες τιμές θα ανατεθούν στις παραμέτρους της συσκευής και θα επιστραφεί το μήνυμα SetParameterValuesResponse. Αν όμως γίνει έστω και ένα σφάλμα δε θα αλλαχτεί η τιμή καμιάς παραμέτρου και θα επιστραφεί ένα SOAP SetParameter Faul μήνυμα, στο οποίο θα αναφέρονται τα ονόματα των παραμέτρων των οποίων οι νέες τιμές είχαν σφάλμα καθώς και το είδος του σφάλματος.



Εικόνα 6: Έλεγχος SetParameterValues

2.8 Το δυναμικό τμήμα του προγράμματος

Οι συναρτήσεις και οι δομές που περιέχονται στα αρχεία «custom.h» και «custom.cpp» αλλάζουν κάθε φορά που ένας νέος πελάτης δημιουργείται.

Στο αρχείο «custom.h», δημιουργούνται οι επικεφαλίδες όλων των συναρτήσεων ανάθεσης επιστροφής. Κάθε συνάρτηση λαμβάνει ως παράμετρο μια δομή ParameterList και δεν επιστρέφει τίποτα. Έτσι, για κάθε όνομα συνάρτησης επιστροφής / ανάθεσης που έχει οριστεί, το πρόγραμμα DeviceCreator θα γράψει στο αρχείο «custom.h» μια γραμμή της μορφής **void nameOfSetFunction(ParameterList *)**. Επίσης, στο ίδιο αρχείο θα γραφεί και η δήλωση της σταθεράς NUMOFVARS, ανάλογα με τον αριθμό των μεταβλητών κατάστασης που περιέχει το πρόγραμμα. Τέλος, στο ίδιο αρχείο υπάρχουν και οι ακόλουθες επικεφαλίδες: **DeviceIdStruct *getDeviceId(void)**, **ParameterValueList *getInformParameterList(void)**, **int getConnectionAddress(char *addr, int *port)**, **char* getConnectionString()**, **unsigned int getListeningPort()**, **int checkParameter(ParameterValueStruct pv)**. Το τι κάνει κάθε μια εξ αυτών θα εξηγηθεί στη συνέχεια.

Στην αρχή του αρχείο «custom.cpp» τώρα, θα δημιουργηθεί και αρχικοποιηθεί ένας πίνακας από δομές StateVariable. Αυτός ο πίνακας ουσιαστικά θα περιέχει πληροφορίες για όλες τις μεταβλητές κατάστασης του πελάτη. Όπως είχε γραφεί και στην ενότητα των δομών αυτού του κειμένου, κάθε δομή StateVariable έχει τέσσερα πεδία: το πλήρες της όνομα, έναν δείκτη για τη συνάρτηση επιστροφής τιμής, ένα δείκτη για τη συνάρτηση ανάθεσης τιμής (θα δείχνει σε NULL αν κάποια μεταβλητή κατάστασης δεν είναι εγγράψιμη), και έναν ακέραιο που θα είναι μηδενικός αν η μεταβλητή κατάστασης δεν είναι εγγράψιμη. Έτσι, το πρόγραμμα DeviceCreator γεμίζει αυτόν τον πίνακα ανάλογα με τα στοιχεία της συσκευής που δημιουργείται. Ο πίνακας αυτός θα έχει NUMOFVARS γραμμές.

Για κάθε μεταβλητή κατάστασης λοιπόν μόνο τα παραπάνω τέσσερα χαρακτηριστικά αποθηκεύονται, παρότι οι μεταβλητές κατάστασης έχουν και μερικά ακόμη χαρακτηριστικά, δηλαδή το αν μπορεί η τιμή τους να διαβαστεί, τον τύπο τους και τις επιτρεπόμενες τιμές που θα πάρουν. Το αν η τιμή τους μπορεί να διαβαστεί δεν περιλαμβάνεται λόγω του τρόπου που δουλεύει το χαρακτηριστικό. Αν δεν μπορεί να διαβαστεί η τιμή μιας μεταβλητής κατάστασης και ο ACS ζητήσει την τιμή της τότε θα πρέπει να επιστραφεί μια άδεια συμβολοακολουθία, χωρίς σφάλμα. Έτσι, είναι ευκολότερο να δημιουργηθεί αυτόματα από τον DeviceCreator το τμήμα επιστροφής τιμής για αυτή τη μεταβλητή κατάστασης και να επιστρέφει μια κενή συμβολοακολουθία, αντί να προστεθεί ακόμα μια ιδιότητα στη δομή Stat-

eVariable και να αυξηθεί ο αριθμός των ελέγχων, ενώ πολύ λίγες μεταβλητές θα έχουν αυτή την ιδιότητα.

Επίσης, πληροφορίες για τον τύπο και τις επιτρεπόμενες τιμές μιας μεταβλητής επίσης δεν περιλαμβάνονται στη δομή StateVariable επειδή ήταν αρκετά ευκολότερο να δημιουργηθεί η συνάρτηση checkSetParameters η οποία θα ενσωματώνει τους ελέγχους για κάθε εγγράψιμη μεταβλητή. Ουσιαστικά, και με τις δύο παραπάνω επιλογές έγινε μια προσπάθεια να φύγει πολυπλοκότητα από το πρόγραμμα του πελάτη και να πάει στο DeviceCreator, κυρίως για λόγους ταχύτητας. Αναφορικά με αυτό, η ιδιότητα για το αν μια μεταβλητή κατάστασης είναι εγγράψιμη ή όχι είναι σημαντικό να υπάρχει, αφού κατά τη δημιουργία του μηνύματος GetParameterNamesResponse, ο πελάτης θα πρέπει να στείλει τα πλήρη ονόματα των μεταβλητών κατάστασης αλλά και το αν είναι εγγράψιμες ή όχι.

Στη συνέχεια αρχίζει το τμήμα του αρχείου «custom.cpp» για τις stand alone μεταβλητές κατάστασης. Εδώ, για κάθε stand alone μεταβλητή κατάστασης θα γίνει η δήλωση μιας μεταβλητής με το όνομα που είχε επιλεγεί από το χρήστη του DeviceCreator που θα έχει ως αρχική τιμή την αρχική τιμή που ομοίως είχε επιλεγεί από το DeviceCreator. Κάθε μια από τις μεταβλητές που δηλώνονται εδώ θα έχει τον αντίστοιχο τύπο με τον τύπο με αυτόν της παραμέτρου που αντιπροσωπεύει, όπως φαίνεται στο τμήμα για τον έλεγχο τύπων του κειμένου.

Ακολούθως, γράφονται οι συναρτήσεις ανάθεσης / επιστροφής. Η λογική αυτών των συναρτήσεων είναι παρόμοια. Στην αρχή κάθε συνάρτησης ανάθεσης / επιστροφής υπάρχει ένα κομμάτι όπου μπορούν να γίνουν οι αρχικοποιήσεις που χρειάζονται για όλες τις παραμέτρους που έχουν την ίδια συνάρτηση ανάθεσης / επιστροφής. Όπως έχει ήδη γραφεί ο τύπος όλων είναι **void function(ParameterValueList *)** και έχουν την ίδια λογική. Αρχικά, υπάρχει ένα μέρος όπου γίνονται όλες οι κοινές αρχικοποιήσεις. Στη συνέχεια, με τη βοήθεια ενός βρόχου, το όνομα καθενός ParameterValueStruct που υπάρχει στη λίστα που περνιέται ως παράμετρος ελέγχεται για το αν είναι το ίδιο με τα ονόματα όλων των μεταβλητών κατάστασης στις οποίες έχει ανατεθεί αυτή η συνάρτηση. Έτσι, αν μια συνάρτηση επιστροφής έχει ανατεθεί μόνο σε μια μεταβλητή κατάστασης θα γίνει μόνο ένας έλεγχος ο οποίος και θα είναι πάντοτε αληθής (αλλιώς δε θα γινόταν κλήση αυτής της συνάρτησης επιστροφής). Όταν επιτευχθεί ταίριασμα θα εκτελεστεί το τμήμα ανάθεσης ή επιστροφής για την παράμετρο της οποίας το όνομα ταίριαξε.

Τώρα, για το τμήμα ανάθεσης κάθε παραμέτρου, το πρόγραμμα DeviceCreator γράφει μια γραμμή της μορφής long int value = parseInt(value_of_parameter), όπου, αντί για long int μπορεί να είναι unsigned long int, ή int, ή char *, ή DateTime, και αντί για par-

seInt μπορεί να είναι parseUnsignedInt, ή parseDateTime, ή parseBoolean, ή αν ο τύπος της παραμέτρου είναι base64 ή string ή any, μια απλή ανάθεση τιμής θα γίνει, χωρίς parsing. Μετά από αυτό, ο χρήστης μπορεί να χρησιμοποιήσει τη μεταβλητή value, η οποία θα έχει το σωστό τύπο σύμφωνα με κάθε παράμετρο όπως θέλει. Αυτό εδώ είναι και το τμήμα το οποίο πρέπει να συμπληρωθεί από το χρήστη, έτσι ώστε το περιεχόμενο της μεταβλητής value να ανατεθεί στην κατάλληλη παράμετρο. Στην Εικόνα 7 φαίνεται ο κώδικας που δημιουργήθηκε από το πρόγραμμα DeviceCreator για μια συνάρτηση ανάθεσης που είχε ανατεθεί στις μεταβλητές με ονόματα 'a.b.c', 'd.e.f' και 'g.h.i'. Τα μέρη που πρέπει να συμπληρωθούν από το χρήστη είναι σημειωμένα.

Για το τμήμα επιστροφής κάθε παραμέτρου, το πρόγραμμα DeviceCreator δηλώνει τη μεταβλητή με όνομα value με το σωστό τύπο και έχοντας για αρχική τιμή το 0 ή το «». Ο προγραμματιστής πρέπει να προσθέσει κώδικα έτσι ώστε η τιμή της αντίστοιχης παραμέτρου να ανατεθεί στη μεταβλητή value. Μετά από αυτό η μεταβλητή θα αντιγραφεί σε μια συμβολοακολουθία και θα αντιγραφεί στο πεδίο value του τρέχοντος ParameterValueStruct της λίστας. Η συνάρτησης επιστροφής έχουν την ίδια δομή με αυτή της Εικόνα 7, με την εξαίρεση ότι εδώ η τιμή της παραμέτρου πρώτα ανατίθεται στη μεταβλητή value και ακολούθως η value ανατίθεται στο pv.value.

Μερικές μεταβλητές κατάστασης μπορεί να δηλωθούν ότι η τιμή τους δε μπορεί να διαβαστεί, με τη βοήθεια του προγράμματος DeviceDescriptor. Αυτό σημαίνει ότι η τιμή τους δε μπορεί να σταλεί με ένα μήνυμα GetParameterValues, αφού περιέχουν usernames ή passwords ή άλλες πολύτιμες πληροφορίες τις οποίες ο ACS θα πρέπει να γνωρίζει. Σε μια συνάρτηση επιστροφής λοιπόν μιας μεταβλητής κατάστασης γράφεται αυτόματα κώδικας ώστε να επιστραφεί μια άδεια συμβολοακολουθία, συμπεριφορά την οποία ο προγραμματιστής δε θα πρέπει να αλλάξει.

Πέρα από τις συναρτήσεις ανάθεσης και επιστροφής η οποίες και θα πρέπει να συμπληρωθούν από το χρήστη δημιουργείται αυτόματα και ο κώδικας για τις ακόλουθες συναρτήσεις:

- **DeviceIdStruct *getDeviceId(void):** Επιστρέφει μια δομή DeviceIdStruct για χρήση με τη συνάρτηση createInform που δημιουργεί το μήνυμα Inform, χρησιμοποιώντας τις πληροφορίες που ο χρήστης συμπληρώνει για τη συσκευή με το πρόγραμμα DeviceCreator.
- **ParameterValueList *getInformParameterList(void):** Επιστρέφει μια δομή ParameterValueList η οποία περιέχει μια λίστα με τα ονόματα και τις τιμές των παραμέτρων που θα σταλούν στην αρχή κάθε δοσοληψίας μαζί με το μήνυμα Inform.

Και εδώ, ο χρήστης του προγράμματος DeviceCreator μπορεί να επιλέξει τις μεταβλητές αυτές για την αυτόματη δημιουργία της συνάρτησης.

- **int getConnectionAddress(char *__addr, int *__port):** Επιστρέφει, μέσω των παραμέτρων __addr και __port τη διεύθυνση και τη θύρα του ACS.
- **char *getConnectionString():** Επιστρέφει τη συμβολοακολουθία σύνδεσης του πελάτη, η οποία χρησιμοποιείται όταν μια σύνδεση αιτείται από τον ACS.
- **unsigned int getListeningPort():** Επιστρέφει τη θύρα στην οποία το πρόγραμμα του πελάτη θα πρέπει να ακούει αναμένοντας αιτήσεις σύνδεσης από τον ACS.

Οι παραπάνω τρεις συναρτήσεις δημιουργούνται αυτόματα από το πρόγραμμα DeviceCreator αν η περιγραφή της συσκευής έχεις τις ακόλουθες δύο μεταβλητές κατάστασης: RootObject.ManagementServer.URL, και RootObject.ManagementServer.ConnectionRequestURL. Το όνομα του RootObject είναι αυθαίρετο, όμως το υπόλοιπο μέρος του ονόματος πρέπει να είναι ακριβώς όπως δίδεται. Η παράμετρος RootObject.ManagementServer.URL περιέχει τη διεύθυνση του ACS (μπορεί να περιέχεται επίσης και ο αριθμός της θύρας), ενώ η RootObject.ManagementServer.URL περιέχει τη διεύθυνση και τη συμβολοακολουθία σύνδεσης του πελάτη (για παράδειγμα http://147.102.7.10:24582/CONN). Αν αυτές οι δύο μεταβλητές κατάστασης δεν υπάρχουν στην περιγραφή της συσκευής, το πρόγραμμα DeviceCreator πάλι θα δημιουργήσει τις συναρτήσεις getConnectionAddress, getConnectionString και getListeningPort, αλλά αυτές οι συναρτήσεις θα περιέχουν κάποιες προκαθορισμένες τιμές, και ο προγραμματιστής συνήθως θα πρέπει να τις αλλάξει.

- **int checkParameter(ParameterValueStruct pv):** Αυτή η σημαντική συνάρτηση χρησιμοποιείται από την checkSetParameterValues του typechk.cpp. Όπως έχει γραφτεί, η συνάρτηση checkSetParameterValues καλεί την checkParameter για κάθε μία από τις εγγράψιμες παραμέτρους που δέχεται. Αυτή η συνάρτηση διαιρείται σε μέρη, κάθε ένα εκ των οποίων είναι αφιερωμένο σε μια εγγράψιμη παράμετρο του πελάτη. Σε κάθε ένα από αυτά τα μέρη, το πρόγραμμα DeviceCreator γράφει τις κατάλληλες εντολές ελέγχου, ανάλογα με τον τύπο της παραμέτρου και το αν έχει επιτρεπόμενες τιμές ή κάποιο πεδίο τιμών. Αυτή η συνάρτηση επιστρέφει έναν ακέραιο, ο οποίος θα είναι 0 αν η τιμή της δομής ParameterValueStruct που περάστηκε ως παράμετρος ήτανε κατάλληλη, η μια αρνητική τιμή αν δεν ήταν κατάλληλη. Η δομή αυτής της συνάρτησης μπορεί να φανεί και στην Εικόνα 6. Αν η παράμετρος που ελέγχεται δεν έχει προκαθορισμένες τιμές ή εύρος επιτρεπόμενων τιμών τότε η τιμή που επιστρέφεται θα είναι η τιμή της κατάλληλης συνάρτησης ελέγχου τύπων.

Αν όμως έχει ένα εύρος τιμών (όπως μπορεί να έχουν μόνο οι ακέραιοι), πρώτα θα ελεγχθεί η τιμή της, μετά θα γίνει parse σε μια μεταβλητή και τέλος θα συγκριθεί με τη μέγιστη και την ελάχιστη τιμή· αν είναι εκτός ορίων θα επιστραφεί ο κωδικός σφάλματος για λάθος τιμή παραμέτρου. Αν από την άλλη η παράμετρος που ελέγχεται μπορεί να λάβει μόνο κάποιες συγκεκριμένες τιμές, τότε θα συγκριθεί με κάθε μια από τις προκαθορισμένες τιμές (οι οποίες πρέπει να έχουν οριστεί με το πρόγραμμα DeviceDescriptor)· αν είναι η τιμή της είναι ίση με κάποια εξ αυτών θα επιστραφεί το 0, αλλιώς θα επιστραφεί ο κωδικός σφάλματος για λάθος τιμή παραμέτρου.

```
void function(List of ParameterValueStructs pvlist) {
   Do any initialization for this function here (to be done by developer)
   foreach ParameterValueStruct in pvlist as pv {
      if(pv.name == "a.b.c") {
         declare variable value with the corresponding to a.b.c type
         value = pv.value
         assign value to parameter a.b.c (to be done by developer)
      }
      if(pv.name == "d.e.f") {
         declare variable value with the corresponding to d.e.f type
         value = pv.value
         assign value to parameter d.e.f (to be done by developer)
      }
      if(pv.name == "g.h.i") {
         declare variable value with the corresponding to g.h.i type
         value = pv.value
         assign value to parameter g.h.i (to be done by developer)
      }
   }
}
```

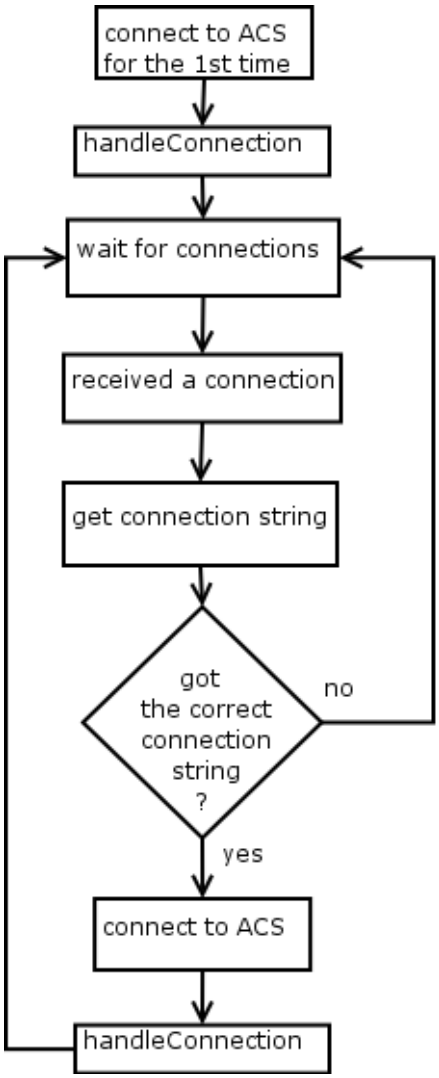
Εικόνα 7: Δημιουργία συνάρτησης ανάθεσης

2.9 Το κυρίως τμήμα του πελάτη

Σε αυτό το μέρος του κειμένου θα εξηγηθεί το κυρίως τμήμα του προγράμματος. Στην Εικόνα 8 φαίνεται ο τρόπος που δουλεύει η συνάρτηση main. Στην αρχή, με τη βοήθεια της συνάρτησης getConnectionAddress του "custom.cpp" λαμβάνεται η διεύθυνση του ACS και ακολούθως ο πελάτης συνδέεται στον ACS για πρώτη φορά, με τη βοήθεια της συνάρτησης makeConnection του "low.cpp". Αν δε συμβούν λάθη, τη σύνδεση θα χειριστεί η συνάρτηση handleConnection. Η HandleConnection, η οποία θα εξηγηθεί αργότερα, είναι μια συνάρτηση από το "client.cpp" που λαμβάνει ως παράμετρο ένα socket και χειρίζεται ολοκληρωτικά τη σύνδεση.

Όταν η πρώτη δοσοληψία μεταξύ του πελάτη και του ACS ολοκληρωθεί, ο έλεγχος θα επιστρέψει από τη συνάρτηση handleConnection στη συνάρτηση main, και θα μπει σε ένα ατέλειωτο βρόχο. Εντός αυτού του βρόχου, ο πελάτης θα αναμένει για αιτήσεις σύνδεσης από τον ACS με τη βοήθεια των συναρτήσεων createListeningSocket και acceptConnection από το "low.cpp". Όταν γίνει μια σύνδεση, ο πελάτης θα πάρει τη συμβολοακολουθία σύνδεσης και θα ελέγξει αν είναι η σωστή με τις συναρτήσεις getConnectionString από το "custom.cpp" και checkConnectionString από το "low.cpp". Η σύνδεση κλείνει, και αν λήφθηκε η σωστή συμβολοακολουθία σύνδεσης ο πελάτης συνδέεται ξανά στον ACS όπως στην αρχή. Αν δε λήφθηκε η σωστή συμβολοακολουθία σύνδεσης ο πελάτης θα αρχίσει ξανά να ακούει για συνδέσεις.

Η μόνη συνάρτηση που δεν έχει αναλυθεί ακόμη είναι η **int handleConnection(SOCKET sd)**, μια περίπλοκη συνάρτηση που χειρίζεται μια σύνδεση μεταξύ του πελάτη και του ACS από την αρχή ως το τέλος. Η παράμετρος της είναι ένα socket για μια σύνδεση που μόλις έχει αρχίσει. Στην αρχή της handleConnection αρχικοποιείται μια δομή ουράς, στην οποία και θα αποθηκεύονται μέχρι να απαντηθούν όλες οι αιτήσεις του ACS προς τον πελάτη.



Εικόνα 8: Η συνάρτηση main

Το πρώτο μήνυμα που πρέπει να σταλεί σε κάθε δοσοληψία είναι το μήνυμα Inform από τον πελάτη στον ACS. Έτσι, η συνάρτηση handleConnection δημιουργεί το μήνυμα Inform με τη συνάρτηση createCPEInform, το αντιγράφει σε μια δομή StringList και το στέλλει με τη sendRawMessage του "low.h". Η συνάρτηση createCPEInform θα αναλυθεί αργότερα, αλλά πρέπει να σημειωθεί σε αυτό το σημείο ότι η παράμετρος MaxEnvelopes του μηνύματος Inform μπορεί να έχει μια αυθαίρετα μεγάλη τιμή arbitrary. Η MaxEnvelopes είναι μια πολύ σημαντική παράμετρος του πελάτη αλλά και του ACS, μέσω της οποίας ο ένας μπορεί να αντιληφθεί το μέγιστο αριθμό SOAP μηνυμάτων σε ένα μοναδικό HTTP μήνυμα τα οποία μπορεί να δεχθεί ο άλλος. Η παράμετρος MaxEnvelopes που στέλλεται μαζί με το μήνυμα Inform αφορά τον πελάτη, ο server θα στείλει τη δικιά του MaxEnvelopes στο μήνυμα InformResponse αμέσως μόλις λάβει το μήνυμα Inform.

Ακολούθως, ο πελάτης δέχεται την απάντηση από τον ACS. Αυτή η απάντηση που είναι ένα HTTP response, μπορεί να περιέχει ένα ή περισσότερα SOAP μηνύματα (αν φυσικά

ο αριθμός της παραμέτρου MaxEnvelopes του πελάτη είναι πάνω από ένα, κάτι που ισχύει στην προκειμένη περίπτωση), αλλά το πρώτο από τα μηνύματα SOAP πρέπει να είναι το μήνυμα InformResponse. Έτσι, η handleConnection λαμβάνει όλα τα SOAP μηνύματα με τη συνάρτηση getMessages του "low.h" και ελέγχει αν το πρώτο είναι πράγματι το μήνυμα InformResponse με τη βοήθεια της parseStart του "soaparse.h". Η parseStart θα επιστρέψει μια δομή SoapMessage της οποίας το πεδίο type θα ελεγχθεί και αν η τιμή του ισούται με τη σταθερά INFORMRESPONSE τότε πράγματι το μήνυμα είναι το InformResponse, οπότε αυτό θα γίνει parse και η τιμή της παραμέτρου MaxEnvelopes του ACS θα αποθηκευθεί, ενώ όλα τα υπόλοιπα μηνύματα που τυχόν λήφθηκαν με αυτό το HTTP μήνυμα θα εισαχθούν στην ουρά ώστε να εξυπηρετηθούν αργότερα. Αν όμως δε λήφθηκε το μήνυμα InformResponse τότε κάποιο πρόβλημα υπήρξε με τον ACS, η σύνδεση κλείνει και η συνάρτηση handleConnection επιστρέφει.

Μετά από αυτό, η ροή του ελέγχου εισέρχεται σε ένα βρόχο, από τον οποίο θα φύγει αν υπάρξουν προβλήματα με τη σύνδεση στον ACS ή αν ο ACS στείλει ένα άδειο μήνυμα και δεν υπάρχουν αναπάντητες αιτήσεις στην ουρά τις οποίες θα πρέπει να απαντήσει ο πελάτης. Όταν ο έλεγχος βγει από το βρόχο η σύνδεση θα τερματιστεί. Όπως φαίνεται στο [1], σελίδα 24 τέσσερις συνθήκες πρέπει να αληθεύουν για το τερματισμό μιας σύνδεσης, όμως οι συνθήκες 2 και 3 (ο πελάτης δεν έχει να στείλει αιτήσεις και ο πελάτης έλαβε όλες τις απαντήσεις από τον ACS) δε μπορεί ποτέ να συμβούν σε αυτό το πρόγραμμα, αφού δε στέλλει αιτήσεις στον πελάτη και αρκεί οι συνθήκες 1 (ο ACS δεν έχει άλλες αιτήσεις να στείλει στον πελάτη – ισχύει όταν σταλεί ένα άδειο μήνυμα προς τον πελάτη) και 4 (ο πελάτης έχει στείλει όλες τις απαντήσεις στον ACS) να αληθεύουν για να τερματιστεί η σύνδεση.

Εντός του βρόχου που ακολουθεί συμβαίνουν τα εξής: Αρχικά ελέγχεται η ουρά. Αν δεν υπάρχουν εκκρεμούσες αιτήσεις στην ουρά θα σταλεί ένα άδειο μήνυμα στον ACS. Αυτό πρέπει να γίνει επειδή ο πελάτης χρησιμοποιεί HTTP posts για να στείλει τα μηνύματα του στον ACS, ενώ ο ACS χρησιμοποιεί HTTP responses για τα μηνύματα του και ένα HTTP response δε μπορεί να σταλεί αν δεν έχει πρώτα ληφθεί ένα HTTP post. Οπότε, σε μια τυπική δοσοληψία, ο πελάτης θα στείλει το μήνυμα Inform με ένα HTTP post Και ο ACS θα στείλει το InformResponse μήνυμα με ένα HTTP response. Τώρα, ο server δε μπορεί να στείλει άλλα μηνύματα εκτός και αν λάβει ακόμα ένα HTTP post (που θα είναι κενό) έτσι ώστε να απαντήσει με ένα HTTP response που θα περιέχει την αίτηση του προς τον πελάτη. Αυτό το σενάριο συμβαίνει αν ο ACS στέλλει ένα μήνυμα SOAP με κάθε HTTP μήνυμα ή αν η παράμετρος MaxEnvelopes του πελάτη είναι ίση με 1 κάτι που φαίνεται στο [1], Figure 3.

Ακολούθως ελέγχεται η ουρά. Αν η ουρά δεν είναι άδεια τότε υπάρχουν αναπάντητες αιτήσεις για τον πελάτη. Ο πελάτης θα απαντήσει όσες περισσότερες μπορεί με το επόμενο μήνυμα HTTP κάτι που σημαίνει ότι θα τις απαντήσει όλες, εκτός και αν η παράμετρος MaxEnvelopes του ACS έχει μικρότερη τιμή από τον αριθμό των αιτήσεων που εκκρεμούν, οπότε και θα απαντηθούν τόσες αιτήσεις όσες και η τιμή του MaxEnvelopes. Όταν καθοριστεί ο αριθμός των αιτήσεων που θα απαντηθούν (συγκρίνονται την τιμή της παραμέτρου MaxEnvelopes του ACS με τον αριθμό των αιτήσεων που εκκρεμούν στην ουρά), κάθε μήνυμα το οποίο περιλαμβάνεται σε αυτά που θα απαντηθούν αφαιρείται από την ουρά και δημιουργείται η απάντηση σε αυτό με τη βοήθεια της συνάρτησης respondTo και αντιγράφεται σε μια δομή StringList. Όταν οι απαντήσεις σε όλα τα μηνύματα έχουν δημιουργηθεί και αντιγραφεί στη StringList θα σταλούν με ένα μόνο HTTP μήνυμα.

Όταν το HTTP post από τον πελάτη σταλεί, το HTTP response από τον ACS θα διαβαστεί. Αν δε στάλθηκαν αιτήσεις από τον ACS και η ουρά είναι άδεια τότε η δοσοληψία θα τερματιστεί. Αν δε στάλθηκαν αιτήσεις αλλά η ουρά δεν ήταν άδεια τότε δε θα συμβεί τίποτα και ο βρόχος θα αρχίσει από την αρχή. Τέλος, αν στάλθηκαν αιτήσεις από τον ACS, αυτές θα εισαχθούν στην ουρά και ο βρόχος θα αρχίσει από την αρχή. Αυτή η συμπεριφορά εξηγεί και το γιατί ελέγχεται το αν η ουρά είναι άδεια στην αρχή του βρόχου: Αν ο server δεν είχε στείλει αιτήσεις και η ουρά ήταν άδεια τότε η δοσοληψία θα είχε τερματιστεί στο τέλος του βρόχου και ο έλεγχος δε θα έφτανε ξανά στην αρχή του βρόχου ώστε να σταλεί το άδειο HTTP μήνυμα.

Η συνάρτηση **char *respondTo(char *req)** χρησιμοποιείται από την handleConnection για να επιστρέψει την απάντηση σε μια αίτηση η οποία λαμβάνεται ως παράμετρος. Αυτή η συνάρτηση θα επιστρέψει NULL σε περίπτωση σφάλματος. Αρχικά, χρησιμοποιεί την parseStart για να αρχίσει το parsing του μηνύματος και να καθορίσει τον τύπο του, αφού η συνάρτηση ParseStart θα επιστρέψει μια δομή SoapMessage της οποίας το πεδίο type θα περιέχει έναν ακέραιο που θα περιέχει τον τύπο του μηνύματος ή ότι το μέθοδος δεν υποστηρίζεται σύμφωνα με τον Πίνακας 1.

Τώρα, ο τύπος του μηνύματος συγκρίνεται με τις σταθερές που καθορίζουν τα είδη των μηνυμάτων. Τα μηνύματα που υποστηρίζονται από τον πελάτη είναι τα GetParameterNames, GetParameterValues and SetParameterValues. Αν ληφθεί οποιοδήποτε άλλο είδος μηνύματος τότε θα επιστραφεί από τη συνάρτηση respondTo ένα μήνυμα fault SOAP.

Αν ο τύπος του μηνύματος ήταν GetParameterNames, τότε θα κληθεί η parseGetParameterNames για να κάνει parse τις παραμέτρους του και η createGetParameter-

NamesResponse θα χρησιμοποιηθεί για τη δημιουργία της απάντησης. Αν συμβεί κάποιο σφάλμα κατά τη διάρκεια της διαδικασίας θα επιστραφεί το κατάλληλο μήνυμα SOAP fault.

Αν ο τύπος του μηνύματος ήταν a GetParameterValues, η respondTo θα καλέσει την parseGetParameterValues. Η συνάρτηση ParseGetParameterValues θα επιστρέψει μια δομή StringList που θα περιέχει τα ονόματα παραμέτρων των οποίων οι τιμές πρέπει να επιστραφούν στον ACS. Αν δε συμβούν λάθη, τα ονόματα των παραμέτρων αυτών θα αντιγραφούν σε μια δομή ParameterValueList με άδειες τιμές στα πεδία value των ParameterValueStruct που περιέχονται στη λίστα, και αυτή η λίστα θα περαστεί στη συνάρτηση handleGetParameterValues που θα εξηγηθεί αργότερα. Όταν αυτή η συνάρτηση επιστρέψει, η ParameterValueList που είχε περαστεί ως παράμετρος θα περιέχει τα ονόματα των παραμέτρων καθώς και τις τιμές τους, οπότε θα περαστεί στη συνάρτηση createGetParameterValuesResponse και τελικά η απάντηση στην αίτηση GetParameterValues θα δημιουργηθεί και επιστραφεί. Σε κάθε βήμα της συνάρτησης respondTo γίνονται έλεγχοι για σφάλματα και επιστρέφεται το κατάλληλο μήνυμα SOAP fault αν βρεθεί κάποιο λάθος.

Τέλος, αν ο τύπος του μηνύματος ήταν SetParameterValues, η συνάρτηση respondTo θα καλέσει αρχικά την parseSetParameterValues η οποία και θα επιστρέψει μια δομή ParameterValueList που θα περιέχει τα ονόματα των παραμέτρων και τις τιμές που θα πρέπει να τους ανατεθούν. Αυτή η λίστα, μαζί με έναν πίνακα ακεραίων με τον ίδιο αριθμό στοιχείων όσος ο αριθμός των δομών ParameterValueStruct που περιέχονται στη λίστα ώστε σε κάθε μια από τις δομές ParameterValueStruct να αντιστοιχεί μια συγκεκριμένη θέση του πίνακα, θα περαστεί στη συνάρτηση checkSetParameterValues, η οποία αναλύθηκε σε προηγούμενη ενότητα και η οποία θα ελέγξει τις τιμές των παραμέτρων για λάθη και για κάθε παράμετρο που θα είναι εσφαλμένη θα τοποθετήσει τον κατάλληλο κωδικό σφάλματος στην αντίστοιχη θέση του πίνακα ακεραίων. Αν δε συμβούν σφάλματα, η συνάρτηση handleSetParameterValues θα κληθεί και η λίστα των ParameterValueStruct θα περαστεί σε αυτήν. Όταν τελικά η handleSetParameterValues επιστρέψει, οι παράμετροι της συσκευής θα έχουν τις νέες τους τιμές και έτσι θα κληθεί η createSetParameterValuesResponse ώστε να δημιουργηθεί η απάντηση στο μήνυμα SetParamaterValues. Αν έγιναν σφάλματα σε οποιοδήποτε σημείο θα επιστραφεί το κατάλληλο μήνυμα SOAP fault.

Εδώ πρέπει να σημειωθεί ότι ανεξαρτήτως του είδους του μηνύματος, ο χειρισμός της SOAP επικεφαλίδας από το πρόγραμμα του πελάτη θα είναι το ίδιο: Αν η αίτηση δεν είχε επικεφαλίδα τότε η απάντηση που θα δημιουργηθεί επίσης δε θα έχει επικεφαλίδα. Αν η αίτηση είχε μια επικεφαλίδα τότε η απάντηση θα έχει επικεφαλίδα ίδια την παράμετρο ID αυτής, όπως ζητείται από το [1], Table 3.

Οι συναρτήσεις **int handleGetParameterValues(ParameterValueList *parameterList)** και **int handleSetParameterValues(ParameterValueList *parameterList)** θα εξηγηθούν στη συνέχεια. Αυτές οι συναρτήσεις επιστρέφουν έναν ακέραιο με αρνητική τιμή όταν συμβεί λάθος. Μάλιστα, η τιμή που επιστρέφει η handleSetParameterValues δε χρειάζεται να ελεγχθεί αφού οποιαδήποτε λάθη θα έπρεπε να είχαν εντοπιστεί από την checkSetParameterValues η οποία καλείται πριν από τη handleSetParameterValues.

Αρχικά θα αναλυθεί η συνάρτηση handleGetParameterValues. Η handleGetParameterValues λαμβάνει ως παράμετρο μια δομή ParameterValueList η οποία είναι μια λίστα από δομές ParameterValueStruct των οποίων το πεδίο name είναι συμπληρωμένο, ενώ το πεδίο value όχι, αφού πρόκειται να συμπληρωθεί από αυτήν. Αυτή η συνάρτηση θα καλέσει μόνο μια φορά κάθε μια από τις συναρτήσεις επιστροφής που πρέπει να κληθούν, κάτι που σημαίνει ότι αν δύο παράμετροι των οποίων οι τιμές πρέπει να επιστραφούν έχουν την ίδια συνάρτηση επιστροφής, η handleGetParameterValues θα καλέσει μόνο μια φορά αυτή τη συνάρτηση επιστροφής τοποθετώντας και τις δύο αυτές παραμέτρους στη λίστα ParameterValueList.

Πρώτα η συνάρτηση δημιουργεί έναν πίνακα ακεραίων με το όνομα group που έχει το ίδιο μέγεθος με τη δομή ParameterList που περνιέται ως παράμετρος και αρχικοποιεί τις τιμές του με το –1. Οι τιμές του πίνακα group θα ορίζουν έναν αριθμό ομάδας, αρχίζοντας από το 0, και κάθε παράμετρος θα ανήκει στην ομάδα που καθορίζεται από την αντίστοιχη θέση του πίνακα group. Η παράμετροι που έχουν την ίδια συνάρτηση επιστροφής θα ανήκουν στην ίδια ομάδα και έτσι θα έχουν τον ίδιο ακέραιο στις θέσεις τους στον πίνακα group. Αυτή η ανάθεση τιμών στον πίνακα group γίνεται με ένα διπλό βρόχο for: Σε κάθε παράμετρος που δεν έχει αριθμό ομάδας δίνεται ένας νέος αριθμός ομάδας και στη συνέχεια όλες οι άλλες παράμετροι ελέγχονται. Αν μια έχει την ίδια συνάρτηση επιστροφής με την αρχική τότε και σε αυτή θα δοθεί ο ίδιος αριθμός ομάδας.

Όταν ανατεθούν οι αριθμοί ομάδων, αρχίζει το τμήμα της handleGetParameterValues στο οποίο καλούνται οι συναρτήσεις. Αυτό το τμήμα θα επαναληφθεί τόσες φορές όσες είναι και οι ομάδες που δημιουργήθηκαν από το προηγούμενο βήμα. Έτσι, για κάθε ομάδα, οι δομές ParameterValueStruct της λίστας που περιέχονται σε αυτή την ομάδα θα αντιγραφούν σε μια νέα δομή ParameterValueList μεγέθους όσο και ο αριθμός των παραμέτρων που ταιριάζουν, και αυτή η νέα ParameterValueList θα περαστεί στη σωστή συνάρτηση επιστροφής. Όταν αυτή η συνάρτηση επιστροφής επιστρέψει, οι τιμές της νέας ParameterValueList θα αντιγραφούν ξανά πίσω στην παλαιά ParameterValueList. Η ίδια διαδικασία θα

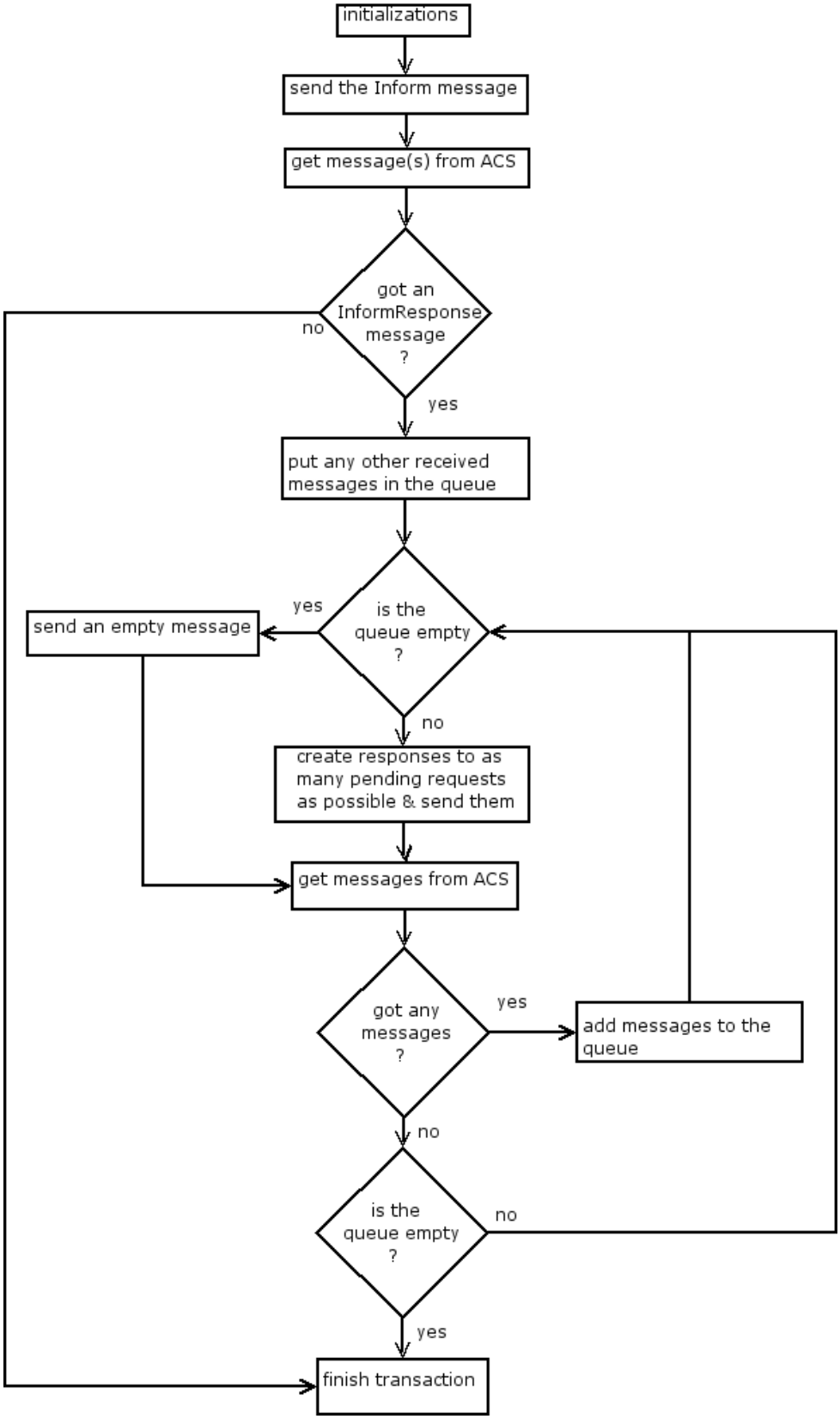
επαναληφθεί για όλες τις ομάδες και έτσι, όταν τελειώσει η δομή ParameterValueList που είχε περαστεί ως παράμετρος στην handleGetParameterValues θα έχει τις τιμές κάθε παραμέτρου. Πολύ περισσότερα σχόλια για τον τρόπο με τον οποίο δουλεύει αυτή η συνάρτηση μπορούν να βρεθούν εντός του πηγαίου κώδικα που βρίσκεται στο τέλος της εργασίας, στο αρχείο "client.cpp".

Η συνάρτηση handleSetParameterValues δουλεύει το ίδιο με την handleGetParameterValues. Λαμβάνει ως παράμετρο μια δομή ParameterValueList, αλλά σε αυτή τη λίστα είναι συμπληρωμένα και το πεδίο name και το πεδίο value των δομών ParameterValueStruct. Ομάδες παραμέτρων επίσης θα σχηματιστούν και οι παράμετροι που ανήκουν σε κάθε ομάδα θα αντιγραφούν σε μια νέα δομή ParameterValueList, η οποία και θα περαστεί στη συνάρτηση ανάθεσης αυτής της ομάδας, ώστε σε αυτή να τεθούν οι νέες τιμές στις παραμέτρους.

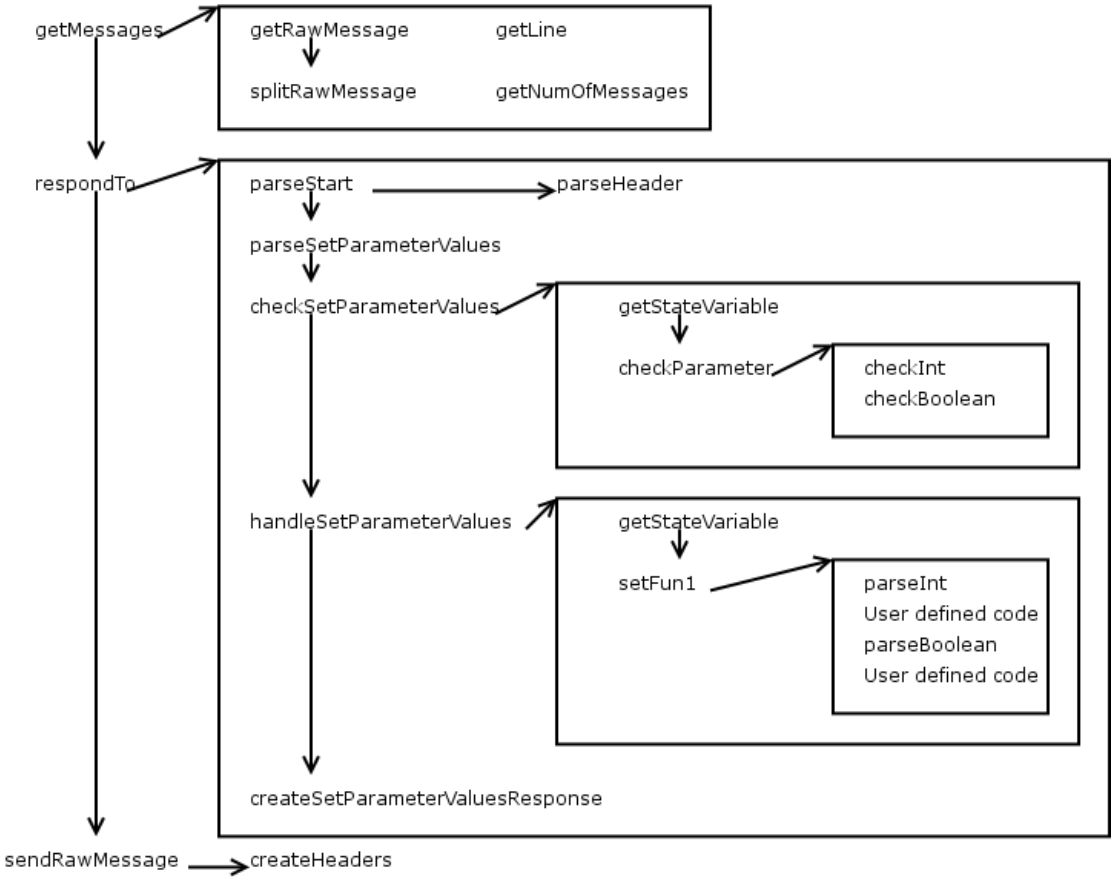
Η συνάρτηση **StateVariable *getStateVariable(char *name)** ψάχνει για μια StateVariable μέσω του ονόματος της. Ελέγχει τον πίνακα των δομών StateVariable με όνομα deviceVariables ο οποίος δημιουργείται αυτόματα στο αρχείο "custom.cpp" και όταν βρει μια μεταβλητή με το σωστό όνομα θα επιστρέψει έναν δείκτη σε αυτήν. Αν δε βρει μεταβλητή με το όνομα name θα επιστρέψει έναν δείκτη σε NULL.

Τέλος, **IXML_Document *createCPEInform(void)** χρησιμοποιείται από την handleConnection όταν δημιουργείται το μήνυμα Inform, κυρίως για να συμμαζευτούν όλες οι άλλες συναρτήσεις που καλούνται όταν δημιουργείται το μήνυμα Inform, οι οποίες και είναι οι getDeviceId από το "custom.cpp" για να επιστραφεί η δομή DeviceIdStruct για τον πελάτη, getCurrentTime από το "helpers.cpp" για να επιστραφεί η ώρα του πελάτη, getInformParameterList από το "custom.cpp" για να επιστραφούν οι παράμετροι οι οποίες θα σταλούν με το μήνυμα Inform καθώς και η συμπλήρωση των παραμέτρων MaxEnvelopes, RetryCount και EventList του Inform με τις σωστές τιμές.

Στην Εικόνα 9 φαίνεται ένα απλοποιημένο διάγραμμα ροής της συνάρτησης handleConnection, ενώ στην Εικόνα 10 φαίνονται οι κλήσεις συναρτήσεων που γίνονται όταν ληφθεί ένα μήνυμα SetParameterValues.



Εικόνα 9: Διάγραμμα ροής της handleConnection



Εικόνα 10: Κλήσεις συναρτήσεων για ένα SetParameterValues μήνυμα

Συγκεκριμένα, υποτίθεται ότι στο SetParameterValues μήνυμα πρόκειται να τεθούν οι τιμές δύο παραμέτρων, μιας Int και μιας Boolean, ότι δε συμβαίνουν λάθη καθώς και ότι και οι δύο παράμετροι έχουν την ίδια συνάρτηση ανάθεσης, με όνομα setFun1. Η χρονική σειρά κλήσης των συναρτήσεων είναι από πάνω προς τα κάτω, ενώ από αριστερά προς τα δεξιά φαίνονται ο τρόπος που η μια συνάρτηση καλή την άλλη, τα κουτιά υπάρχουν για να φαίνεται καλύτερα η ομαδοποίηση των συναρτήσεων που καλούνται από τις άλλες.

Όπως φαίνεται από το σχήμα, αρχικά θα κληθεί η getMessages για να παραλάβει το SetParameterValues μήνυμα η οποία θα καλέσει τις συναρτήσεις που είχαν αναφερθεί στο κεφάλαιο για την επικοινωνία χαμηλού επιπέδου. Ακολούθως, θα κληθεί η respondTo για να απαντήσει. Η respondTo αρχικά θα καλέσει την parseStart για να βρει τον τύπο του μηνύματος, ενώ η parseStart θα καλέσει την parseHeader για να κάνει parse την επικεφαλίδα. Ακολούθως, και αφού ο τύπος του μηνύματος είναι SetParameterValues, θα κληθεί η parseSetParameterValues για να επιστρέψει τη λίστα με τις παραμέτρους και στη συνέχεια η checkSetParameterValues για να ελέγξει τις παραμέτρους. Η checkSetParameterValues με τη σειρά της θα καλέσει για κάθε μια από τις παραμέτρους της λίστας τη getStateVariable και μετά την checkParameter από το «custom.cpp». Η checkParameter τώρα θα καλέσει

μια φορά την checkInt και μια την checkBoolean για να ελέγξει τον τύπο των δύο παραμέτρων.

Αφού τελείωσαν οι έλεγχοι και δεν βρέθηκαν λάθη η respondTo θα καλέσει την handleSetParameterValues. Η handleSetParameterValues αρχικά θα καλέσει την getStateVariable για κάθε παράμετρο της λίστας και, αφού ομαδοποιήσει τις δύο αυτές παραμέτρους (αφού έχουν κοινή συνάρτηση ανάθεσης), θα καλέσει μόνο μια φορά τη setFun1. Η setFun1 τώρα, θα καλέσει μια φορά την parseInt και μια την parseBoolean, για να γίνουν parse οι τιμές των παραμέτρων στις κατάλληλες μεταβλητές. Μετά από αυτά θα υπάρχει ο κώδικας που πρέπει να γραφτεί με το χέρι και που θα καθορίζει το τι θα κάνει η συσκευή. Τέλος, καλείται η createSetParameterValuesResponse για να δημιουργηθεί η απάντηση στο SetParameterValues μήνυμα.

Η respondTo επιστρέφει την απάντηση στο SetParameterValues και αυτό στέλλεται με τη βοήθεια της sendRawMessage, η οποία καλεί την createHeaders για να δημιουργηθούν οι επικεφαλίδες του HTTP μηνύματος.

2.10 Η βιβλιοθήκη IXML

Το πρόγραμμα χρησιμοποιεί πολύ τη βιβλιοθήκη ixml της Intel. Η ixml είναι ένας DOM2 [3] parser για την ANSI C (παρότι το εγχειρίδιο της αναφέρει ότι είναι για Linux, μπορεί να δουλέψει με κάθε ANSI C μεταγλωττιστή), και έτσι μπορεί να χρησιμοποιηθεί και στη windows και στη Unix έκδοση του προγράμματος. Η βιβλιοθήκη ixml περιλαμβάνεται στο αρχείο «libixml.a» το οποίο είναι μια στατική βιβλιοθήκη της C που δημιουργήθηκε από τον πηγαίο κώδικα της βιβλιοθήκης ixml. Η έκδοση για windows χρησιμοποιεί μια διαφορετική έκδοση του αρχείου «libixml.a» από αυτή του Unix επειδή η μορφή των βιβλιοθηκών σε αυτά τα λειτουργικά συστήματα είναι διαφορετική. Αν αντί για τη μεταγλωττισμένη μορφή της βιβλιοθήκης είχε περιληφθεί ο πηγαίος της κώδικας τότε θα αρκούσε μόνο μια έκδοση αυτού, αφού όπως αναφέρθηκε είναι σε ANSI C. Επίσης, το αρχείο «ixml.h» περιέχει τις επικεφαλίδες των συναρτήσεων της βιβλιοθήκης ixml.

Οι επόμενες δομές της ixml χρησιμοποιούνται από το πρόγραμμα:

- IXML_Node: Είναι ένας κόμβος XML.
- IXML_Document: Είναι ένα ολόκληρο έγγραφο XML.
- IXML_Element: Είναι ένα XML Element.
- IXML_NodeList: Είναι μια λίστα από κόμβους XML.
- DOMString: Είναι μια XML συμβολοακολουθία (typedef του char *)
- BOOL: Είναι ένας ακέραιος, το 1 είναι TRUE και το 0 FALSE

Εκτός των παραπάνω δομών, οι επόμενες συναρτήσεις της ixml χρησιμοποιούνται στο πρόγραμμα:

- const DOMString ixmlNode_getNodeName(IXML_Node* nodeptr): Επιστρέφει το όνομα του κόμβου nodeptr.
- DOMString ixmlNode_getNodeValue(IXML_Node* nodeptr): Επιστρέφει την τιμή του (κείμενο) του nodeptr.
- int ixmlNode_setNodeValue(IXML_Node* nodeptr, char *newNodeValue): Θέτει την τιμή (δηλαδή το κείμενο) του nodeptr.
- IXML_NodeList* ixmlNode_getChildNodes(IXML_Node* nodeptr): Επιστρέφει μια λίστα με τα παιδιά του nodeptr.

- IXML_Node* ixmlNode_getFirstChild(IXML_Node* nodeptr): Επιστρέφει το πρώτο από τα παιδιά του nodeptr
- IXML_Node* ixmlNode_getNextSibling(IXML_node* nodeptr): Επιστρέφει τον επόμενο αδερφό του nodeptr.
- int ixmlNode_appendChild(IXML_Node *nodeptr, IXML_Node *newChild): Προσθέτει τον κόμβο newChild στη λίστα των παιδιών του nodeptr.
- IXML_Node* ixmlNode_cloneNode(IXML_Node *nodeptr, BOOL deep): Αντιγράφει τον κόμβο nodeptr. Αν η παράμετρος deep είναι true θα γίνει βαθιά αντιγραφή, δηλαδή θα αντιγραφούν και τα παιδιά του.
- void ixmlNode_free(IXML_Node *nodeptr): Ελευθερώνει τη μνήμη που καταλαμβάνει ο nodeptr και τα παιδιά αυτού.
- IXML_Document* ixmlDocument_createDocument(): Δημιουργεί ένα νέο, κενό έγγραφο.
- IXML_Node* ixmlDocument_createTextNode(IXML_Document* doc, DOMString data): Δημιουργεί έναν νέο κόμβο κειμένου που θα περιέχει τη συμβολοακολουθία data και θα ανήκει στο έγγραφο doc.
- IXML_Element* ixmlDocument_createElementNS(IXML_Document* doc, DOMString namespaceURI, DOMString qualifiedName): Δημιουργεί ένα νέο element του έγγραφου doc, που ανήκει στο namespace namespaecURI και έχοντας το όνομα qualifiedName.
- DOMString ixmlElement_getAttribute(IXML_Element* element, DOMString name): Returns the value of attribution name from element, the actual value and not a copy is returned.
- int ixmlElement_setAttribute(IXML_Element* element, DOMString name, DOMString value): Θέτει την τιμή της ιδιότητας name του element στην τιμή value. Αν δεν υπάρχει η ιδιότητα με το όνομα name στη λίστα των ιδιοτήτων του element τότε πρώτα θα προστεθεί.
- IXML_Node* ixmlNodeList_item(IXML_NodeList *nList, unsigned long index): Επιστρέφει έναν κόμβο από τη λίστα nList. Ο κόμβος που θα επιστραφεί καθορίζεται από τον ακέραιο index.
- unsigned long ixmlNodeList_length(IXML_NodeList *nList): Επιστρέφει το μέγεθος της λίστας nList.

- void ixmlNodeList_free(IXML_NodeList* nList): Ελευθερώνει τη μνήμη που καταλαμβάνει η nList.
- int ixmlParseBufferEx(char *buffer, IXML_Document **doc): Κάνει parse το buffer και το μετατρέπει σε μια δομή DOM.
- DOMString ixmlPrintNode(IXML_Node* doc): Επιστρέφει μια συμβολοακολουθία που περιέχει το XML κείμενο του doc και όλων των παιδιών του.

3 Το πρόγραμμα DeviceDescriptor

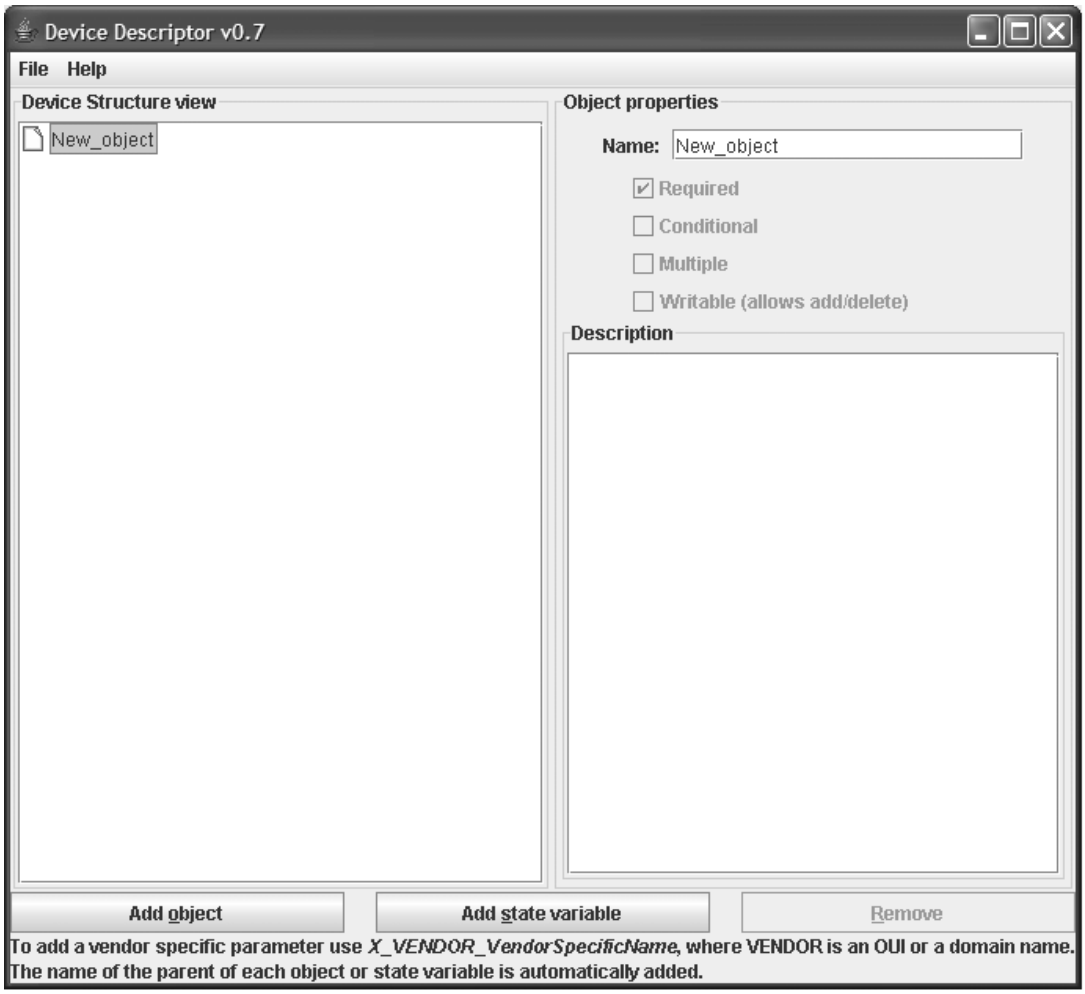
Το DeviceDescriptor είναι ένα πρόγραμμα σε Java για τη δημιουργία «περιγραφών» συσκευών. Όπως είχε γραφτεί και στην εισαγωγή, η περιγραφή μιας συσκευής έχει μια ιεραρχική μορφή με κάποιο αντικείμενο στη ρίζα. Αυτή ακριβώς η μορφή της συσκευής οδηγεί στην κατάλληλη αναπαράσταση της περιγραφής μιας συσκευής και η οποία είναι μέσω ενός αρχείου XML.

3.1 Η χρήση του DeviceDescriptor

Όταν ξεκινήσει το πρόγραμμα, το γραφικό του περιβάλλον είναι όπως φαίνεται στην Εικόνα 11. Η διαπροσωπεία χρήστη χωρίζεται σε δύο μέρη. Στο αριστερό εμφανίζονται οι παράμετροι της συσκευής σε δενδρική μορφή και στο δεξί οι ιδιότητες του αντικειμένου ή της μεταβλητής κατάστασης που έχει επιλεγεί από το δένδρο.

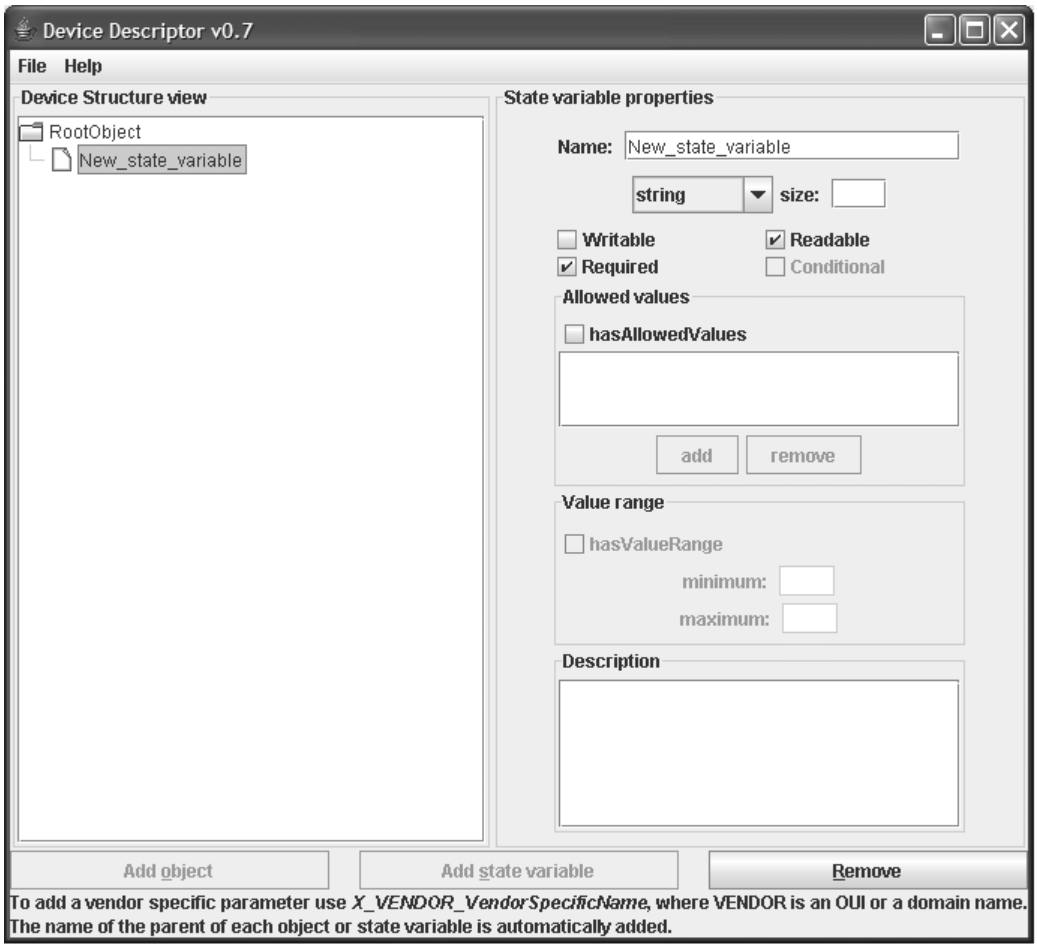
Στο κάτω μέρος του παραθύρου υπάρχουν τρία κουμπιά. Με το «Add object» προστίθεται ένα αντικείμενο ως παιδί στο αντικείμενο που είναι ήδη επιλεγμένο, με το «Add state variable» προστίθεται μια μεταβλητή κατάστασης στο αντικείμενο που είναι ήδη επιλεγμένο ενώ με το «Remove» αφαιρείται το αντικείμενο ή η μεταβλητή κατάστασης από τον πατέρα του. Τα δύο πρώτα κουμπιά είναι ενεργοποιημένα μόνο αν στο δέντρο είναι επιλεγμένο ένα αντικείμενο, ενώ το τρίτο είναι πάντα ενεργό, εκτός και αν είναι επιλεγμένο το ριζικό αντικείμενο της συσκευής.

Στην Εικόνα 11 φαίνονται και οι επιλογές που δίδονται για τις ιδιότητες ενός αντικειμένου. Οι ιδιότητες καθορίζουν το όνομα του αντικειμένου, την περιγραφή του, το αν το αντικείμενο αυτό θα είναι ή όχι απαραίτητο για τη συσκευή, το αν θα είναι εξαρτημένο, το αν θα είναι πολλαπλό και το αν θα είναι εγγράψιμο. Οι ιδιότητες απαραίτητο και εξαρτημένο εξηγούνται στην αρχή της εργασίας. Οι ιδιότητα Multiple πρέπει να επιλεγεί αν το συγκεκριμένο αντικείμενο θα έχει περισσότερα από ένα στιγμιότυπα, δηλαδή αν το αντικείμενο a.b.c είναι Multiple θα αποτελείται από τα στιγμιότυπα a.b.c.1, a.b.c.2, a.b.c.3 κοκ. Η ιδιότητα Writable ενός αντικειμένου έχει να κάνει με το αν μπορούν να κληθούν οι μέθοδοι AddObject και DeleteObject στο αντικείμενο αυτό. Οι μέθοδοι αυτές δεν υποστηρίζονται από το πρόγραμμα πελάτη που θα δημιουργηθεί τελικά, όμως υπάρχουν ως επιλογές στο DeviceDescriptor για πληρότητα.



Εικόνα 11: Η αρχική οθόνη του DeviceDescriptor

Με το που θα προστεθεί μια νέα ή θα επιλεγεί νέα μεταβλητή κατάστασης, θα εμφανιστούν στο δεξί παράθυρο οι ιδιότητες για τη συγκεκριμένη μεταβλητή κατάστασης, όπως φαίνονται και στην Εικόνα 12. Οι ιδιότητες μιας μεταβλητής κατάστασης έχουν να κάνουν με το όνομα της, την περιγραφή της, τον τύπο της, το αν θα είναι εγγράψιμη, το αν θα μπορεί να διαβαστεί η τιμή της, το αν θα είναι απαραίτητη και το αν θα είναι εξαρτημένη. Επίσης, μια μεταβλητή με τον κατάλληλο τύπο μπορεί να δέχεται μόνο συγκεκριμένες επιτρεπόμενες τιμές ή οι τιμές της να κινούνται μόνο σε ένα εύρος τιμών που μπορεί να καθοριστεί.



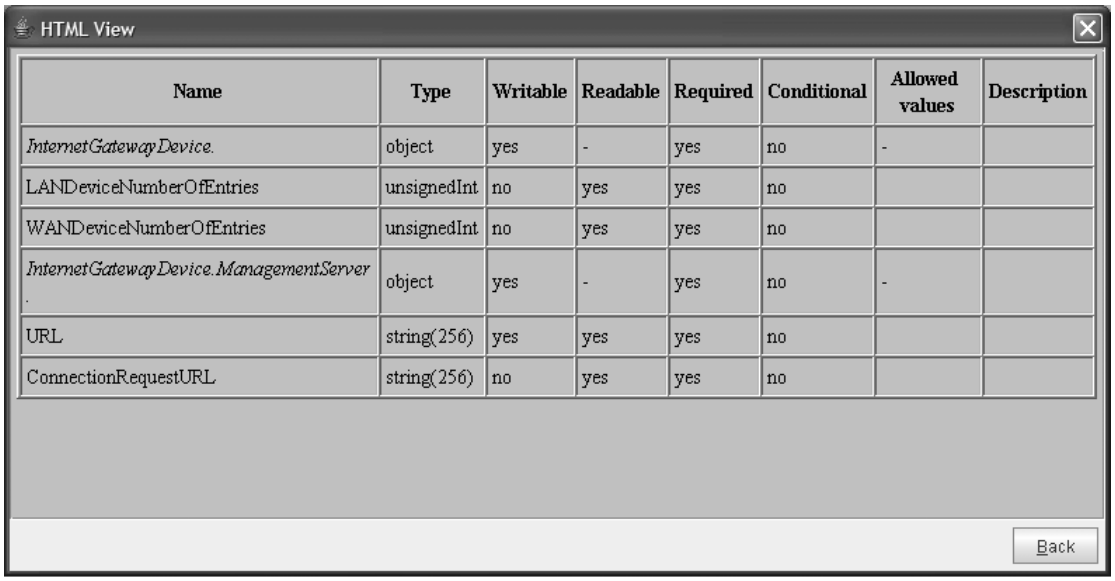
Εικόνα 12: Ιδιότητες μεταβλητής κατάστασης

Το File menu του προγράμματος έχει τις εξής επιλογές: «New device», «Load device», «Save device», «View XML», «View HTML» και «Exit». Η λειτουργία των τριών πρώτων και της τελευταίας επιλογής είναι προφανής. Η τέταρτη εμφανίζει την περιγραφή της συσκευής σε μορφή XML, όπως θα είναι όταν αποθηκευθεί η περιγραφή και στο σκληρό δίσκο με τη «Save Device». Η περιγραφή μιας απλής συσκευής φαίνεται στην Εικόνα 13. Η περιγραφή μιας συσκευής ακολουθεί το XML Schema που φαίνεται στο παράρτημα 8.2.

Τέλος, με την επιλογή «View HTML» εμφανίζεται η περιγραφή της συσκευής σε μορφή που είναι εύκολο να διαβαστεί από το χρήστη του προγράμματος, ανάλογη με το Appendix B του [1]. Η περιγραφή της συσκευής της Εικόνα 13 σε HTML φαίνεται στην Εικόνα 14. Για την HTML εμφάνιση της περιγραφής της συσκευής χρησιμοποιείται ένας μετασχηματισμός XSL ο οποίος μετασχηματίζει το XML σε HTML. Ο μετασχηματισμός αυτός φαίνεται στο παράρτημα 8.3.

Τέλος, στο Help menu υπάρχει μόνο η επιλογή «About» που απλώς εμφανίζει πληροφορίες για το πρόγραμμα.

Εικόνα 13: Εμφάνιση XML



Name	Type	Writable	Readable	Required	Conditional	Allowed values	Description
InternetGatewayDevice.	object	yes	-	yes	no	-	
LANDeviceNumberOfEntries	unsignedInt	no	yes	yes	no		
WANDeviceNumberOfEntries	unsignedInt	no	yes	yes	no		
InternetGatewayDevice.ManagementServer.	object	yes	-	yes	no	-	
URL	string(256)	yes	yes	yes	no		
ConnectionRequestURL	string(256)	no	yes	yes	no		

Εικόνα 14: Εμφάνιση HTML

3.2 Υλοποίηση του DeviceDescriptor

Λίγα λόγια σχετικά με την υλοποίηση του προγράμματος σε Java: Η τάξη που περιέχει το σκελετό του γραφικού περιβάλλοντος είναι η DeviceDescriptor η οποία και κληρονομεί από το JFrame. Αυτή, αριστερά περιέχει την DeviceTreePanel η οποία κληρονομεί από το JPanel, ενώ στα δεξιά έχει ένα JPanel με CardLayout, το οποίο περιέχει δύο άλλα JPanels, το ObjectPanel που έχει να κάνει με τις ιδιότητες ενός αντικειμένου και το StateVariablePanel που έχει να κάνει με τις ιδιότητες μιας μεταβλητής κατάστασης. Ανάλογα με τη ποιο αντικείμενο ή μεταβλητή κατάστασης είναι επιλεγμένο στο DeviceTreePanel στο δεξί JPanel θα φαίνεται είτε ένα ObjectPanel είτε ένα StateVariablePanel, έχοντας επιλεγμένες τις κατάλληλες ιδιότητες.

Δύο άλλες τάξεις που χρησιμοποιούνται είναι οι XmlViewer και HtmlViewer. Αυτές οι τάξεις κληρονομούν και οι δύο από την JDialog και εμφανίζουν στην οθόνη ένα παράθυρο στο οποίο φαίνονται με τη μεν XmlViewer ο κώδικας σε XML που θα δημιουργηθεί και με τη δε HtmlViewer η μετατροπή αυτού του κώδικα σε HTML.

Τέλος, οι τρεις σημαντικότερες τάξεις του προγράμματος, είναι οι DeviceTreeNode, DeviceObject και StateVariable. Οι DeviceObject και StateVariable αναπαριστούν αντίστοιχα ένα αντικείμενο και μια μεταβλητή κατάστασης και κληρονομούν και οι δύο από την DeviceTreeNode η οποία αναπαριστά έναν κόμβο JTree. Η DeviceTreeNode λοιπόν, υλοποιεί τις διεπαφές της Java TreeNode και MutableTreeNode έτσι ώστε να μπορεί αυτή και οι τάξεις που κληρονομούν από αυτή να είναι κόμβοι ενός JTree (μέσω της διεπαφής TreeNode) το οποίο μάλιστα JTree θα μπορεί να αλλάζει δυναμικά (μέσω της διεπαφής MutableTreeNode). Αυτό το JTree περιέχεται στο DeviceTreePanel και αναπαριστά ολόκληρη την ιεραρχία αντικειμένων και μεταβλητών κατάστασης της συσκευής.

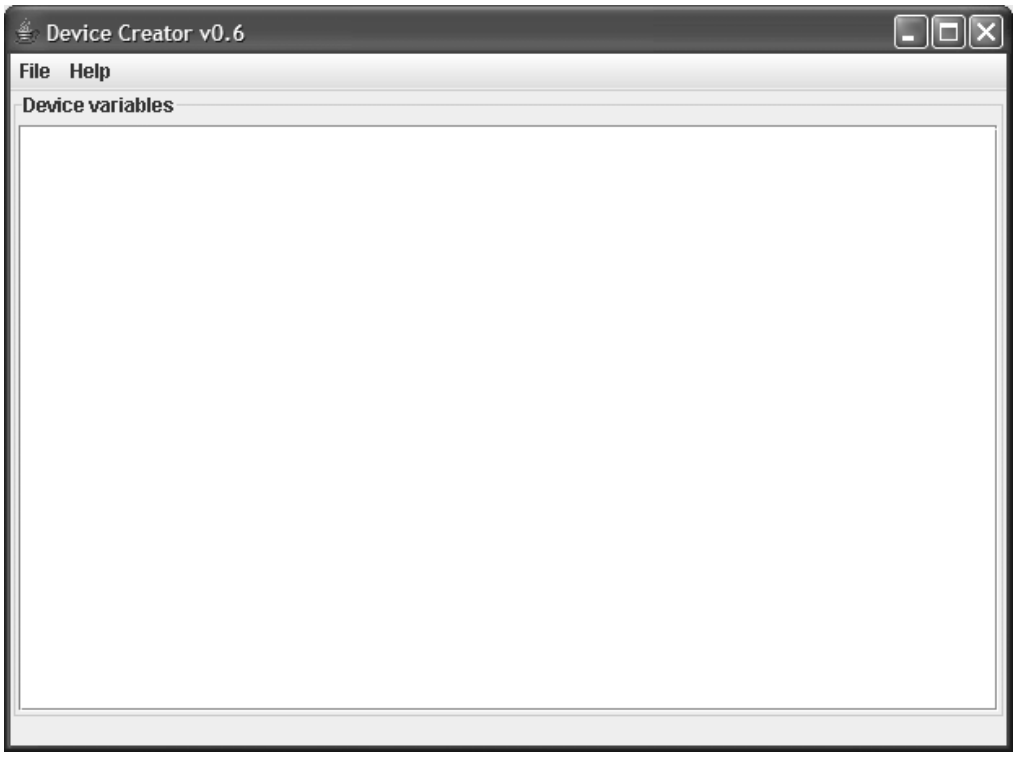
Μέσω των επιλογών του δεξιού μέρους του παραθύρου αλλάζουν οι ιδιότητες του επιλεγμένου κόμβου του δέντρου που υπάρχει στο αριστερό μέρους του παράθυρου. Ουσιαστικά η επικοινωνία μεταξύ των αντικειμένων των τάξεων DeviceTreePanel και StateVariablePanel και ObjectPanel γίνεται μέσω ενός αντικειμένου της τάξης DeviceCreator, το οποίο περιέχει αναφορές για τα αντικείμενα των άλλων τάξεων. Έτσι, όταν αλλάξει κάποια ιδιότητα μιας μεταβλητής κατάστασης από το StateVariablePanel, αυτό καλεί το την κατάλληλη μέθοδο της τάξης DeviceCreator στο οποίο περιέχεται, το οποίο με τη σειρά του καλεί την κατάλληλη μέθοδο του αντικειμένου της τάξης DeviceTreePanel που περιέχει, το οποία επίσης καλεί την κατάλληλη μέθοδο του αντικειμένου της τάξης StateVariable που είναι ο επιλεγμένος κόμβος του δέντρου.

4 Το πρόγραμμα DeviceCreator

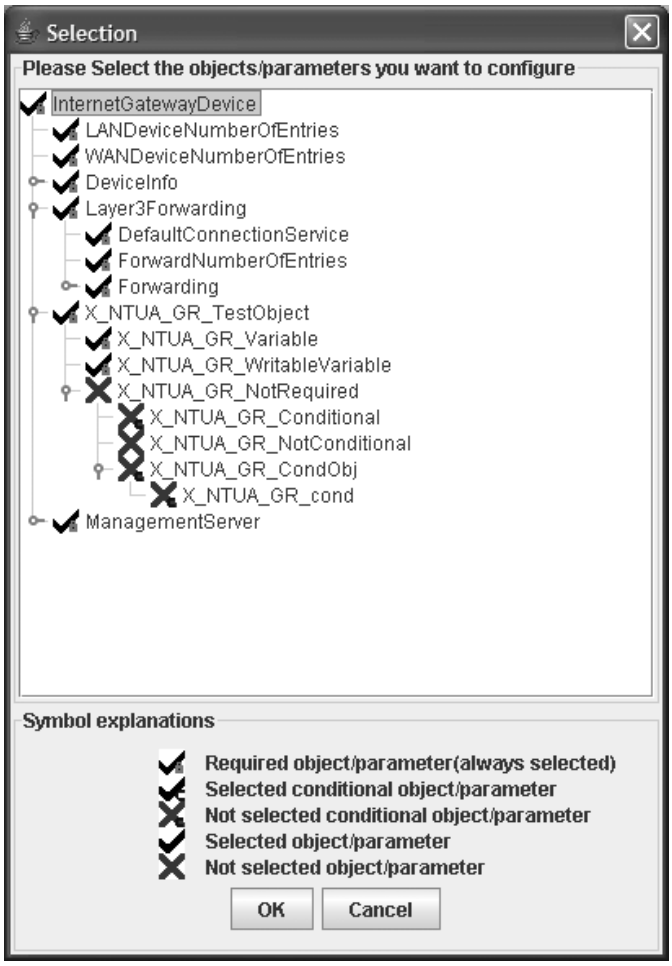
Για τη δημιουργία συσκευών γίνεται χρήση του προγράμματος DeviceCreator. Ουσιαστικά δέχεται την περιγραφή μιας συσκευής, και, αφού ο χρήστης καθορίσει κάποιες κατάλληλες παραμέτρους το πρόγραμμα δημιουργεί τον πηγαίο κώδικα για μια συσκευή.

4.1 Η χρήση του DeviceCreator

Όταν το πρόγραμμα ξεκινήσει, εμφανίζεται η Εικόνα 15. Στην αρχική οθόνη δε μπορεί να γίνει τίποτα το ενδιαφέρον μέχρι να φορτωθεί η περιγραφή μιας συσκευής ή μια ήδη αποθηκευμένη συσκευή. Το menu File έχει τις επιλογές «New device», «Save device», «Load device», «Create device» και «Exit». Με την πρώτη εισάγεται μια περιγραφή μιας συσκευής, με τις δύο επόμενες αποθηκεύεται και φορτώνεται η δουλειά που έχει ήδη γίνει στο DeviceCreator. Το XML Schema που ακολουθούν τα αρχεία που σώζονται και φορτώνονται από το DeviceCreator φαίνεται στο παράρτημα 8.4. Με την Create device δημιουργείται ο κώδικας της συσκευής. Το menu Help έχει μια επιλογή με την οποία εμφανίζεται ένα επεξηγηματικό μήνυμα.



Εικόνα 15 : Η αρχική οθόνη του DeviceCreator

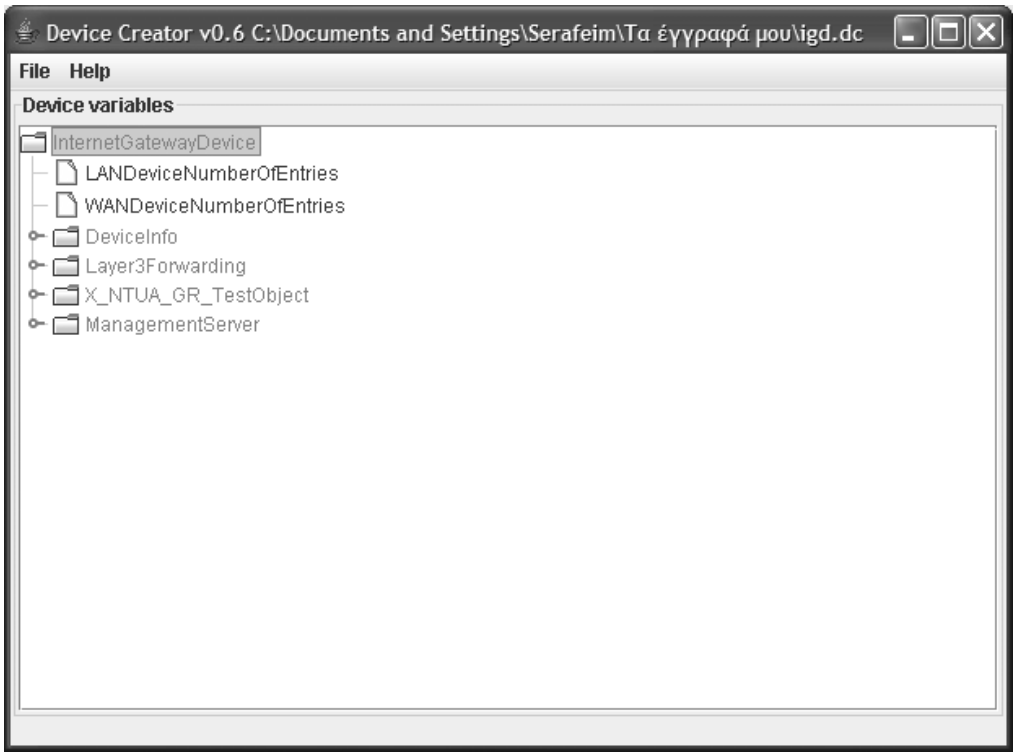


Εικόνα 16: Επιλογή μεταβλητών

Με το που θα επιλεγεί από το menu το «New device», ο χρήστης θα πρέπει να επιλέξει κάποια περιγραφή συσκευής γραμμένη με το DeviceDescriptor ή ακόμα και με το χέρι. Όταν επιλεγεί η περιγραφή της συσκευής, θα εμφανιστεί στον χρήστη το παράθυρο που φαίνεται στην Εικόνα 2. Με τη βοήθεια αυτού του παραθύρου ο χρήστης μπορεί να διαλέξει τις μεταβλητές κατάστασης και τα αντικείμενα που τελικά θα είναι στη συσκευή που θα δημιουργηθεί κάνοντας κλικ στο όνομα κάθε παραμέτρου (το X σημαίνει ότι η συγκεκριμένη παράμετρος δεν είναι επιλεγμένη, ενώ το V σημαίνει ότι είναι επιλεγμένη). Όπως φαίνεται και από το υπόμνημα του παραθύρου, οι μεταβλητές και τα αντικείμενα μπορούν να χωριστούν σε απαραίτητα (required – το εικονίδιο τους έχει ένα λουκέτο στο κάτω δεξί μέρος του) για τη συσκευή, οπότε και είναι πάντοτε επιλεγμένα, εξαρτημένα (conditional – το εικονίδιο τους έχει το γράμμα c στο κάτω δεξί μέρος) και προαιρετικά (optional – το εικονίδιο τους δεν έχει κάποιο ιδιαίτερο χαρακτηριστικό). Η περιγραφή των παραπάνω ιδιοτήτων κάθε παραμέτρου γίνεται σε άλλο σημείο του κειμένου. Η συμπεριφορά όμως του προγράμματος είναι απολύτως συμβατή με τις ιδιότητες, δηλαδή οι απαραίτητες παράμετροι είναι πάντοτε επιλεγμένες, αν επιλεγεί κάποιο μη απαραίτητο αντικείμενο τότε αυτόματα θα

επιλεγούν και όλες οι εξαρτημένες παράμετροι του (καθώς και τα παιδιά αυτών αν έχουμε να κάνουμε με αντικείμενο), αν κάποια εξαρτημένη παράμετρος από-επιλεγεί τότε και το αντικείμενο πατέρας της θα από-επιλεγεί, αφού δε μπορεί να υλοποιηθεί το αντικείμενο αν δεν υλοποιηθούν όλες οι εξαρτημένες παράμετροι παιδιά του κ.ο.κ.

Όταν λοιπόν επιλεγούν οι μεταβλητές που ο χρήστης θέλει να διαχειριστεί και πατηθεί το κουμπί «OK», θα εμφανιστεί η οθόνη που φαίνεται στην Εικόνα 17. Σε αυτήν φαίνονται τα ονόματα των παραμέτρων σε δενδρική μορφή. Θα πρέπει τώρα να καθοριστεί για κάθε μια από τις παραμέτρους της συσκευής μια συνάρτηση που θα καθορίζει την τιμή της και μια συνάρτηση που θα επιστρέφει την τιμή της.



Εικόνα 17: Κύρια οθόνη του DeviceCreator

Για τον καθορισμό των συναρτήσεων αυτών μπορεί να επιλεγεί μια μεταβλητή κατάστασης και, με το πάτημα του δεξιού πλήκτρου του ποντικιού θα εμφανιστεί ένα pop up menu το οποίο έχει τις επιλογές «Set function», «Get function» και «Special properties». Με την επιλογή «Set function» μπορεί να δοθεί το όνομα της συνάρτησης που θα θέτει την τιμή αυτής της μεταβλητής κατάστασης. Ομοίως, με την επιλογή Get function δίνεται το όνομα της συνάρτησης που επιστρέφει τη μεταβλητή αυτή. Οι επιλογές που κάνει ο χρήστης για τα ονόματα των συναρτήσεων φαίνονται στο κάτω μέρος του παραθύρου. Όταν ο χρήστης δώσει για όνομα συνάρτησης ένα μη νόμιμο αναγνωριστικό της C θα εμφανιστεί κατάλληλο μήνυμα λάθους. Επίσης μήνυμα λάθους εμφανίζεται και όταν ο χρήστης προσπα-

θήσει να καθορίσει συνάρτηση που θέτει την τιμή σε μια μεταβλητή που δεν είναι εγγράψιμη.

Θα πρέπει σε όλες τις μεταβλητές κατάστασης να δοθεί όνομα στη συνάρτηση που επιστρέφει την τιμή τους, και σε όλες τις εγγράψιμες να δοθεί όνομα στη συνάρτηση που θέτει την τιμή τους. Αν ένα αντικείμενο είναι επιλεγμένο όταν οριστεί το όνομα μιας συνάρτησης που θέτει ή επιστρέφει τιμή, θα αποδοθεί η συνάρτηση αυτή αναδρομικά σε όλα τα παιδιά του επιλεγμένου αντικειμένου.

Όταν αποδοθούν οι τιμές σε όλες τις μεταβλητές κατάστασης μπορεί πλέον να δημιουργηθεί η συσκευή. Όμως, μπορούν να καθοριστούν και κάποιες επιπλέον δυνατότητες για κάποιες μεταβλητές κατάστασης, με την επιλογή «Special properties».

Εικόνα 18: Special properties

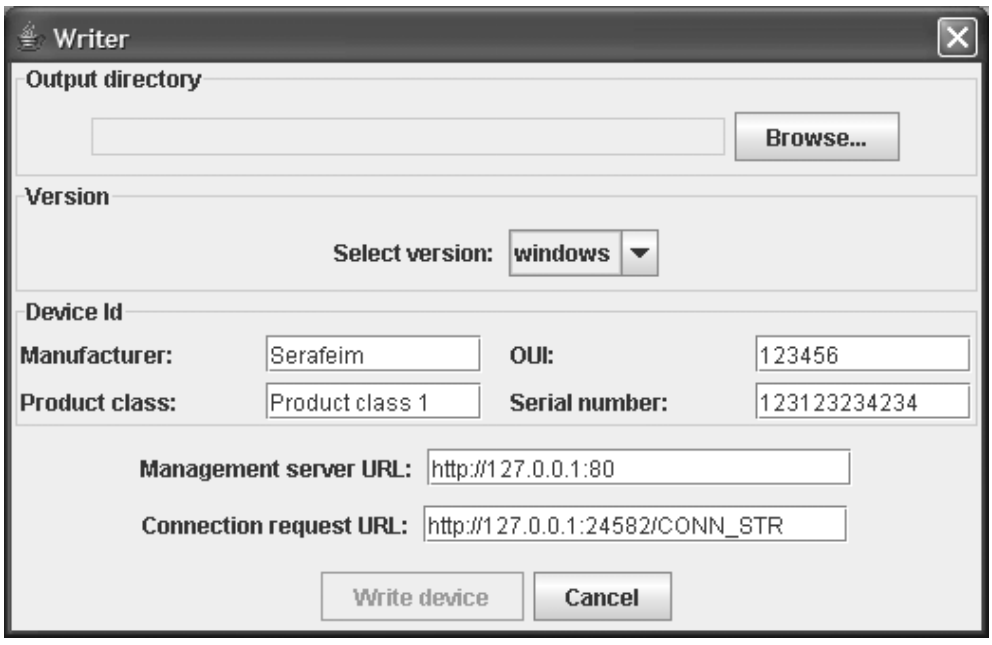
Όταν επιλεγεί το «Special properties» θα εμφανιστεί η οθόνη της Εικόνα 18. Στην αρχή φαίνονται το όνομα, ο τύπος καθώς και το αν η μεταβλητή κατάστασης είναι εγγράψιμη. Ακολούθως, υπάρχει η επιλογή για το αν η μεταβλητή θα είναι stand alone, αν δηλαδή θα υπάρχει μόνο εντός του προγράμματος που θα δημιουργηθεί και δε θα συνδέεται με κάποια άλλη παράμετρο της συσκευής. Σε stand alone μεταβλητή θα πρέπει να καθοριστεί το όνομα της μεταβλητής που θα αποθηκεύει την τιμή της, καθώς και μια αρχική τιμή. Για την παράμετρο αυτή δε θα χρειάζεται να γίνει καμία επέμβαση από το χρήστη στον κώδικα που δημιουργείται, αφού ο χειρισμός της γίνεται αυτόματα με τη βοήθεια της μεταβλητής που ορίζεται εδώ. Το όνομα της μεταβλητής που θα δοθεί πρέπει να είναι κατάλληλο αναγνωριστικό της C, ενώ η αρχική τιμή της πρέπει να είναι σύμφωνη με τον τύπο της παραμέτρου.

Η επιλογή Required inform variable, ορίζει ότι η παράμετρος αυτή είναι μια από τις παραμέτρους που η συσκευή πρέπει να στέλλει στον ACS όταν συνδέεται σε αυτόν, με το inform μήνυμα.

Η συσκευή μπορεί να δημιουργηθεί με την επιλογή «Create device» του file menu. Η οθόνη της Εικόνα 19 θα εμφανιστεί με αυτή την επιλογή. Ο χρήστης μπορεί να καθορίσει το κατάλογο του συστήματος στον οποίο θα αποθηκευθεί ο πηγαίος κώδικας της συσκευής καθώς και το αν θα δημιουργηθεί η έκδοση για τα windows ή το Unix. Επίσης μπορεί να καθοριστεί το περιεχόμενο του DeviceId struct, το οποίο στέλλεται μαζί με το μήνυμα Inform κατά τη σύνδεση του client στον ACS και περιέχει πληροφορίες για τη συσκευή.

Αν η συσκευή αυτή έχει ένα αντικείμενο που να είναι παιδί του ριζικού αντικειμένου με όνομα ManagementServer και αν αυτό το αντικείμενο έχει ως παιδιά τις μεταβλητές URL και ConnectionRequestURL, ο χρήστης θα έχει τη δυνατότητα να ορίσει το URL του ACS στο οποίο θα συνδεθεί αρχικά η συσκευή καθώς και το connection string του client, το οποίο θα πρέπει να λάβει αυτός από τον ACS ώστε να καταλάβει ότι ο ACS θέλει να συνδεθεί μαζί του. Αυτές οι επιλογές θα εμφανιστούν στο χρήστη μόνο αν υπάρχουν οι μεταβλητές με όνομα RootObject.ManagementServer.URL και RootObject. ManagementServer.ConnectionRequestURL, όπως αναφέρθηκε και πιο πάνω.

Αν δεν υπάρχουν αυτές οι μεταβλητές ο χρήστης θα πρέπει να καθορίσει αυτά τα χαρακτηριστικά με την επεξεργασία των κατάλληλων συναρτήσεων του αρχείου custom.h. Επίσης, οι μεταβλητές αυτές πάντοτε θα είναι stand-alone και θα έχουν ένα προκαθορισμένο όνομα μεταβλητής.



Εικόνα 19: Δημιουργία συσκευής

4.2 Η υλοποίηση του DeviceCreator

Αναφορικά με την υλοποίηση σε Java του προγράμματος: Χρησιμοποιείται η τάξη DeviceCreator που κληρονομεί από τη JFrame για το κυρίως πρόγραμμα, και περιέχει ένα αντικείμενο της τάξης MainTreePanel που κληρονομεί από τη JPanel, που παρουσιάζει το δέντρο των αντικειμένων και μεταβλητών κατάστασης. Στην αρχή, όταν γίνεται η επιλογή των παραμέτρων που θα υλοποιούνται, χρησιμοποιείται η τάξη SelectionDialog που κληρονομεί από την JDialog και περιέχει ένα αντικείμενο της τάξης SelectionTreePanel που κληρονομεί από τη JPanel.

Υπάρχουν επίσης και οι τάξεις SpecialPropertiEsDialog και WriteDialog που κληρονομούν από τη JDialog και, η μεν πρώτη αφορά το παράθυρο με τις επιπλέον επιλογές για κάθε μεταβλητή κατάστασης, ενώ η δεύτερη αφορά του παράθυρο που εμφανίζεται όταν ο χρήστης θελήσει να δημιουργήσει τον πηγαίο κώδικα της συσκευής. Για αυτή την εργασία χρησιμοποιείται και η τάξη DeviceWriter, η οποία ουσιαστικά γράφει τον πηγαίο κώδικα της συσκευής σύμφωνα με τις επιλογές του WriteDialog. Η DeviceWriter εξετάζει τις επιλογές που έχουν γίνει καθώς και την ιεραρχία των αντικειμένων / μεταβλητών κατάστασης και δημιουργεί κατάλληλα τα αρχεία «custom.h» και «custom.cpp» που είναι και το δυναμικό μέρος του προγράμματος.

Τέλος, οι τάξεις DeviceObject και StateVariable είναι οι ίδιες που χρησιμοποιήθηκαν στο DeviceDescriptor για την αναπαράσταση των παραμέτρων της συσκευής στο δέντρο και την επιλογή των ιδιοτήτων τους.

5 Το πρόγραμμα Simple ACS

Το Simple ACS είναι ένα πρόγραμμα σε Java το οποίο λειτουργεί ως TR-69 ACS. Μπορεί να συνδεθεί με κάποιο συμβατό client, να λάβει τα ονόματα των παραμέτρων του και να τις διαχειριστεί, αλλάζοντας και επιστρέφοντας τις τιμές τους.

5.1 Η χρήση του SimpleACS

Όταν το πρόγραμμα ξεκινήσει εμφανίζεται το παράθυρο της Εικόνα 20 και ο server ακούει για συνδέσεις σε κατάλληλη θύρα. Στο αριστερό μέρος του παραθύρου θα εμφανίζονται τα ονόματα των παραμέτρων του client, στο δεξί μέρος του παραθύρου τα XML/SOAP μηνύματα που ανταλλάσσονται μεταξύ server και client, στο πάνω μέρος του παραθύρου θα μπορεί να επιλεγεί ένα ήδη υπάρχον Connection string για σύνδεση με κάποιο client ή να εισαχθεί κάποιο νέο, ενώ στο κάτω θα εμφανίζονται οι πληροφορίες για τον client που είναι συνδεδεμένος κάθε φορά.

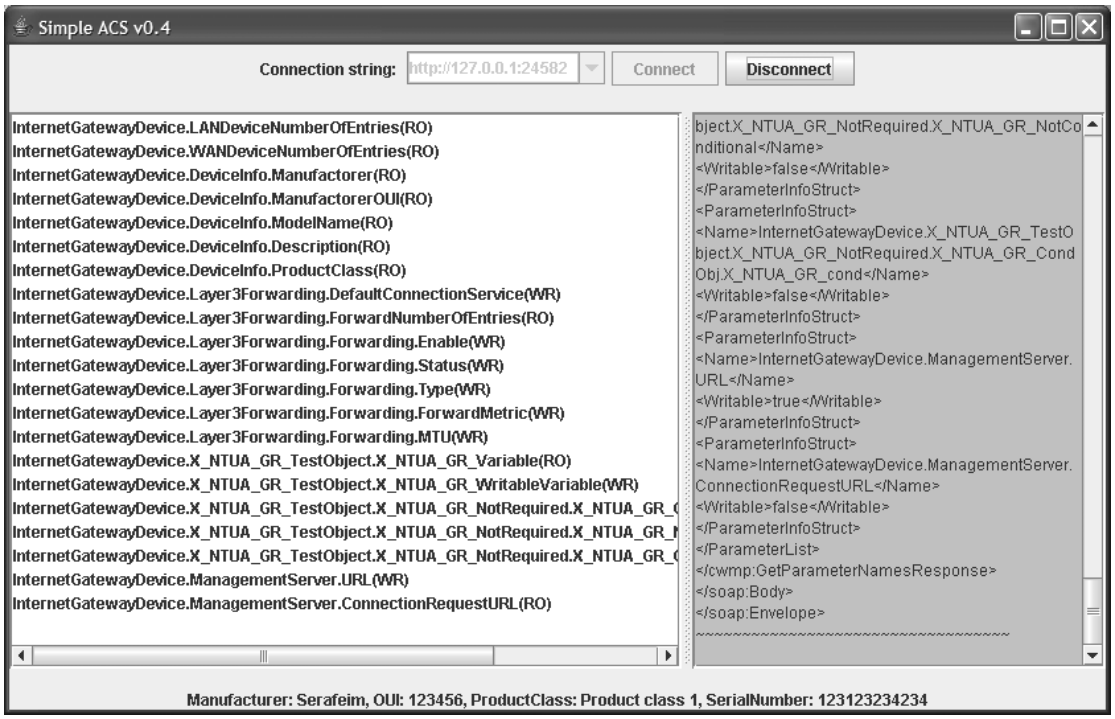
Εικόνα 20: Αρχική οθόνη του SimpleACS

Όταν συνδεθεί κάποιος πελάτης, αφού ο server λάβει το Inform μήνυμα και απαντήσει με το InformResponse, θα στείλει αμέσως το GetParameterNames, έτσι ώστε να λάβει τα ονόματα των παραμέτρων του client, και θα τα εμφανίσει στο αριστερό μέρος του παρα-

θύρου. Η Εικόνα 21 παρουσιάζει τη μορφή του προγράμματος όταν κάποιος client συνδεθεί.

Ο χρήστης μπορεί να εξετάσει τα μηνύματα που ανταλλάχτηκαν από το δεξί παράθυρο ή να δει τις παραμέτρους του client από το αριστερό παράθυρο. Φαίνεται το όνομα τους και το αν είναι ή όχι εγγράψιμες. Σημειώνεται ότι αυτές είναι και οι μοναδικές πληροφορίες οι οποίες μπορεί να σταλούν με το GetParameterNamesResponse μήνυμα. Επίσης, το connection string αυτού του client που φαίνεται γκριζαρισμένο στο πάνω μέρος του παραθύρου αποθηκεύεται στη λίστα με τα connection strings του server και έτσι αργότερα ο server μπορεί να συνδεθεί με τον ίδιο client επιλέγοντας το κατάλληλο connection string και πατώντας connect. Τότε, θα γίνει μια σύνδεση κατ' απαίτηση του server.

Ουσιαστικά, θα πρέπει ο κάθε πελάτης να συνδέεται μια φορά με τον server ώστε να του στείλει το connection string του, και στη συνέχεια να αναμένει από τον server να του ζητήσει να συνδεθούν. Ένας μόνο πελάτης μπορεί κάθε φορά να είναι συνδεδεμένος.



Εικόνα 21: Εμφάνιση παραμέτρων συσκευής

Επιλέγοντας μια ή περισσότερες από τις παραμέτρους του αριστερού παραθύρου, και πατώντας το δεξί πλήκτρο του ποντικιού θα εμφανιστούν οι επιλογές «Set values» και «Get values». Η πρώτη δουλεύει μόνο με εγγράψιμες παραμέτρους, και εμφανίζει ένα παράθυρο όπως αυτό που φαίνεται στην Εικόνα 22. Για κάθε επιλεγμένη παράμετρο ο χρήστης θα πρέπει να δώσει την τιμή στην οποία θέλει η παράμετρος αυτή να τεθεί.

Εικόνα 22: Οθόνη δημιουργίας μηνύματος ανάθεσης

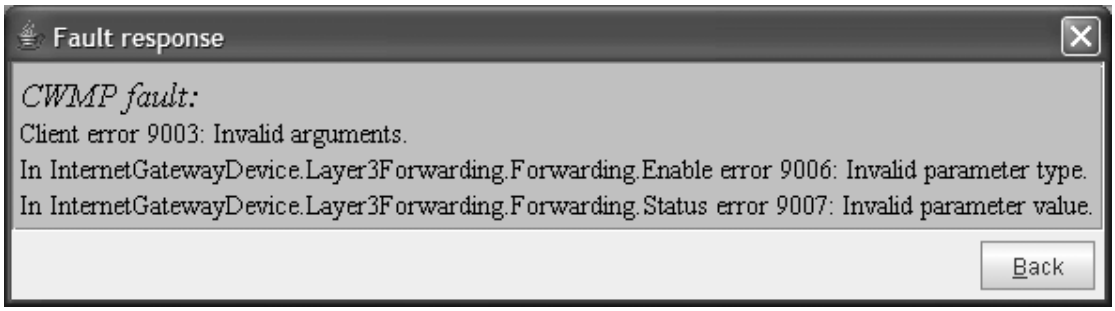
Παρομοίως, όταν επιλεγεί η «Get values» θα εμφανιστεί το παράθυρο όπως αυτό στην Εικόνα 23 που θα εμφανίζει το όνομα και την τιμή των επιλεγμένων μεταβλητών.

Εικόνα 23: Οθόνη εμφάνισης μηνύματος επιστροφής

Σημειώνεται εδώ ότι κάθε φορά μόνο ένα μήνυμα στέλλεται ανεξάρτητα από τον αριθμό των παραμέτρων που έχουν επιλεγεί. Όλες οι επιλεγμένες παράμετροι καθορίζονται από το ίδιο μήνυμα.

Όταν τώρα το SimpleACS λάβει κάποια απάντηση SOAP Fault (όταν π.χ. μια σε μια μεταβλητή δοθεί τιμή που δεν ανήκει στον σωστό τύπο), το μήνυμα λάθους θα εμφανιστεί στην οθόνη. Για να μη χρειαστεί αν γίνει parse το μήνυμα, έχει χρησιμοποιηθεί ένας μετασχηματισμός XSL, ο οποίος εμφανίζει τις πληροφορίες του μηνύματος σε μορφή HTML. Ο κώδικας του μετασχηματισμού XSL φαίνεται στο παράρτημα 8.5. Ο μετασχηματισμός αυτός είναι συμβατός και με απλά SOAP Fault responses και με SetParameter fault responses που επιστρέφονται όταν συμβεί σφάλμα κατά την κλήση της SetParameterValues και δίνουν πιο αναλυτικές πληροφορίες.

Στην Εικόνα 24 φαίνεται το μήνυμα που εμφανίζει το SimpleACS όταν λάβει ένα SetParameter Fault response. Όταν ληφθεί ένα απλό Fault response τότε θα εμφανιστεί το ίδιο παράθυρο, έχοντας μόνο όμως τις δύο πρώτες γραμμές, αφού οι υπόλοιπες αναφέρονται σε κάθε μια από τις μεταβλητές του SetParameterValues μηνύματος που είχαν σφάλμα.



Εικόνα 24: Εμφάνιση λάθους

5.2 Η υλοποίηση του SimpleACS

Στη συνέχεια θα σημειωθούν κάποια σημαντικά στοιχεία για την υλοποίηση του προγράμματος. Η κύρια τάξη είναι η SimpleACS, κληρονομεί από το JFrame και περιέχει αντικείμενα τριών άλλων τάξεων που κληρονομούν και οι τρεις από το JPanel: την ConnectionPanel για τα χαρακτηριστικά του πελάτη στον οποίο θέλει ο ACS να συνδεθεί, τη VariablePanel που περιέχει τις μεταβλητές κατάστασης του πελάτη σε μορφή λίστας (JList) και την MessagePanel που εμφανίζει τα μηνύματα που ανταλλάσσονται.

Υπάρχουν ακόμα οι τάξεις GetDialog και SetDialog που έχουν να κάνουν με τα αποτελέσματα του μηνύματος GetParameterValues και τη δημιουργία του μηνύματος SetParameterValues καθώς και η HtmlViwer που υπάρχει και στον DeviceDescriptor και χρησιμοποιείται για να μετασχηματίσει σε HTML το μήνυμα σφάλματος.

Εκτός από αυτές τις τάξεις για τα γραφικά, υπάρχουν αρκετές τάξεις που αφορούν και την επικοινωνία μέσω του HTTP πρωτοκόλλου. Οι τεχνικές που χρησιμοποιήθηκαν εδώ, παρότι το πρόγραμμα είναι σε Java είναι παρόμοιες με εκείνες που χρησιμοποιήθηκαν για το πρόγραμμα του πελάτη που γράφτηκε σε C. Οι τάξεις που χρησιμοποιούνται για την επικοινωνία είναι η HttpServer, η HttpInputStream, η HttpOutputStream και η SendingThread.

Για τη δημιουργία και parsing των μηνυμάτων SOAP, χρησιμοποιήθηκε η αφαιρετική τάξη SoapMessage, από την οποία κληρονομούσαν οι τάξεις InformMessage, InformResponseMessage, GetParameterNamesMessage, GetParameterValuesMessage, SetParameterValuesMessage, GetParameterNamesResponseMessage, GetParameterValuesResponseMessage, SetParameterValuesResponseMessage που παρίσταναν τα αντίστοιχα μηνύματα SOAP.

Τέλος, υπάρχουν τάξεις που αναπαριστούν τις διάφορες δομές του TR-69, όπως η DeviceStruct, η DateTime, η ParameterValueStruct.

6 Αναφορές

[1] TR-69, CPE WAN Management Protocol, DSL Forum Technical Report

[2] XML Schema Part 2: Datatypes, http://www.w3.org/TR/xmlschema-2/

[3] Document Object Model (DOM) Level 2 Specification, http://www.w3.org/TR/DOM-Level-2-Core

[4] RFC2616, Hypertext Transfer Protocol -- HTTP/1.1

[5] Simple Object Access Protocol (SOAP) 1.1, http://www.w3.org/TR/SOAP

[6] TR-44, Auto-Configuration Architecture & Framework, DSL Forum Technical Report

7 Συντομογραφίες

ACS: Auto Configuration Server

ASP: Application Service Provider

CPE: Customer Premises Equipment

CWMP: CPE WAN Management Protocol

DOM: Document Object Model

HTTP: Hypertext Transfer Protocol

IGD: Internet Gateway Device

IP: Internet Protocol

NAP: Network Access Provider

NSP: Network Service Provider

SOAP: Simple Object Access Protocol

SSL: Secure Sockets Layer

TCP: Transmission Control Protocol

TLS: Transport Layer Security

WAN: Wide Area Network

XML: eXtensible Markup Language

8 Παραρτήματα

8.1 Το makefile για την έκδοση windows

```
OBJS=soapcrt.o soaparse.o helpers.o que.o low.o custom.o typechk.o
HDRS=low.h que.h soapcrt.h soaparse.h helpers.h custom.h typechk.h
CC=g++ -D_CWMP_WIN

all: client

client: $(OBJS) client.o
  g++ -D_CWMP_WIN -Wall $(OBJS) client.o -o client -L. -lixml -
lwsock32

client.o: client.cpp $(HDRS)
  $(CC) -c client.cpp

soaparse.o: soaparse.cpp soaparse.h
  $(CC) -c soaparse.cpp

soapcrt.o: soapcrt.cpp soapcrt.h
  $(CC) -c soapcrt.cpp

low.o: low.cpp low.h
  $(CC) -c low.cpp

que.o: que.cpp que.h
  $(CC) -c que.cpp

helpers.o: helpers.cpp helpers.h
  $(CC) -c helpers.cpp

typechk.o: typechk.cpp typechk.h
  $(CC) -c typechk.cpp

custom.o: custom.cpp custom.h
  $(CC) -c custom.cpp


.PHONY: clean
clean:
  del *.o
  del client.exe
```

8.2 Το XML Schema των περιγραφών του DeviceDescriptor

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
   <xs:element name="allowedValue" type="xs:string"/>
   <xs:element name="minValue" type="xs:string"/>
   <xs:element name="maxValue" type="xs:string"/>
   <xs:element name="description" type="xs:string"/>
   <xs:attribute name="name" type="xs:string"/>
   <xs:attribute name="type" type="xs:string"/>
   <xs:attribute name="writable" type="xs:boolean"/>
   <xs:attribute name="multiple" type="xs:boolean"/>
   <xs:attribute name="required" type="xs:boolean"/>
   <xs:attribute name="conditional" type="xs:boolean"/>
   <xs:attribute name="readable" type="xs:boolean"/>
   <xs:element name="allowedValues">
      <xs:complexType>
         <xs:sequence>
            <xs:element ref="allowedValue" maxOccurs="unbounded"/>
         </xs:sequence>
      </xs:complexType>
   </xs:element>
   <xs:element name="allowedValueRange">
      <xs:complexType>
         <xs:sequence>
            <xs:element ref="minValue" minOccurs="0"/>
            <xs:element ref="maxValue" minOccurs="0"/>
         </xs:sequence>
      </xs:complexType>
   </xs:element>
   <xs:element name="stateVariable">
      <xs:complexType>
         <xs:sequence>
            <xs:element ref="description"/>
            <xs:element ref="allowedValues" minOccurs="0"/>
            <xs:element ref="allowedValueRange" minOccurs="0"/>
         </xs:sequence>
         <xs:attribute ref="name"/>
         <xs:attribute ref="type"/>
         <xs:attribute ref="writable" default="true"/>
         <xs:attribute ref="required" default="true"/>
         <xs:attribute ref="conditional" default="false"/>
         <xs:attribute ref="readable" default="true"/>
      </xs:complexType>
   </xs:element>
   <xs:element name="object">
      <xs:complexType>
         <xs:sequence>
            <xs:element ref="description"/>
            <xs:element ref="stateVariable" minOccurs="0"
maxOccurs="unbounded"/>
            <xs:element ref="object" minOccurs="0"
maxOccurs="unbounded"/>
         </xs:sequence>
         <xs:attribute ref="name"/>
         <xs:attribute ref="multiple" default="false"/>
         <xs:attribute ref="writable" default="true"/>
```

```
        <xs:attribute ref="required" default="true"/>
        <xs:attribute ref="conditional" default="false"/>
      </xs:complexType>
    </xs:element>
    <xs:element name="device">
      <xs:complexType>
        <xs:sequence>
          <xs:element ref="object"/>
        </xs:sequence>
      </xs:complexType>
    </xs:element>
</xs:schema>
```

8.3 Μετασχηματισμός XML περιγραφής συσκευής σε HTML

```
<?xml version="1.0"?>
<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">

<xsl:output method="html" />

<xsl:template match="/">
  <html>
    <head><title>TR-69</title></head>
    <body>
      <xsl:apply-templates select="Device" />
    </body>
  </html>
</xsl:template>

<xsl:template match="Device">
  <table border = '1'>
  <tr>
    <th>Name</th>
    <th>Type</th>
    <th>Writable</th>
    <th>Readable</th>
    <th>Required</th>
    <th>Conditional</th>
    <th>Allowed values</th>
    <th>Description</th>
  </tr>
  <xsl:apply-templates select="object" />
  </table>
</xsl:template>

<xsl:template match="object">
  <xsl:variable name="objname">
    <xsl:call-template name="get-full-name">
    </xsl:call-template>
  </xsl:variable>
  <tr>
    <td><em><xsl:value-of select ="$objname" /></em></td>
    <td><xsl:text>object</xsl:text></td>
    <td>
      <xsl:choose>
        <xsl:when test="@writable='true'">
          <xsl:text>yes</xsl:text>
        </xsl:when>
        <xsl:otherwise>
          <xsl:text>yes</xsl:text>
        </xsl:otherwise>
      </xsl:choose>
    </td>
    <td>
      <xsl:text>-</xsl:text>
    </td>
    <td>
      <xsl:choose>
        <xsl:when test="@required='false'">
          <xsl:text>no</xsl:text>
```

```
          </xsl:when>
          <xsl:otherwise>
            <xsl:text>yes</xsl:text>
          </xsl:otherwise>
        </xsl:choose>
      </td>
      <td>
        <xsl:choose>
          <xsl:when test="@conditional='true'">
            <xsl:text>yes</xsl:text>
          </xsl:when>
          <xsl:otherwise>
            <xsl:text>no</xsl:text>
          </xsl:otherwise>
        </xsl:choose>
      </td>
      <td>
        <xsl:text>-</xsl:text>
      </td>
      <td>
        <xsl:value-of select = "description" />
      </td>

    </tr>
    <xsl:apply-templates select="stateVariable" />
    <xsl:apply-templates select="object" />
</xsl:template>

<xsl:template match="stateVariable">
    <tr>
      <td><xsl:value-of select ="@name" /></td>
      <td>
        <xsl:value-of select ="@type" />
        <xsl:if test="allowedValueRange" >
          <xsl:text>[</xsl:text>
          <xsl:if test="allowedValueRange/minValue" >
            <xsl:value-of select="allowedValueRange/minValue" />
          </xsl:if>
          <xsl:text>:</xsl:text>
          <xsl:if test="allowedValueRange/maxValue" >
            <xsl:value-of select="allowedValueRange/maxValue" />
          </xsl:if>
          <xsl:text>]</xsl:text>
        </xsl:if>

      </td>
      <td>
        <xsl:choose>
          <xsl:when test="@writable='false'">
            <xsl:text>no</xsl:text>
          </xsl:when>
          <xsl:otherwise>
            <xsl:text>yes</xsl:text>
          </xsl:otherwise>
        </xsl:choose>
      </td>
      <td>
        <xsl:choose>
          <xsl:when test="@readable='false'">
            <xsl:text>no</xsl:text>
          </xsl:when>
```

```
          <xsl:otherwise>
            <xsl:text>yes</xsl:text>
          </xsl:otherwise>
        </xsl:choose>
      </td>
      <td>
        <xsl:choose>
          <xsl:when test="@required='false'">
            <xsl:text>no</xsl:text>
          </xsl:when>
          <xsl:otherwise>
            <xsl:text>yes</xsl:text>
          </xsl:otherwise>
        </xsl:choose>
      </td>
      <td>
        <xsl:choose>
          <xsl:when test="@conditional='true'">
            <xsl:text>yes</xsl:text>
          </xsl:when>
          <xsl:otherwise>
            <xsl:text>no</xsl:text>
          </xsl:otherwise>
        </xsl:choose>
      </td>
      <td>
        <xsl:if test="allowedValues">
          <xsl:for-each select="allowedValues/allowedValue">
            <xsl:value-of select="." />
            <br />
          </xsl:for-each>
        </xsl:if>
      </td>
      <td>
        <xsl:value-of select = "description" />
      </td>

    </tr>
  </xsl:template>

  <xsl:template name="get-full-name">
    <xsl:for-each select="ancestor-or-self::*">
      <xsl:if test="name(.)='object'" >
        <xsl:value-of select="concat(@name,'.')" />
        <xsl:if test="@multiple='true'">
          <xsl:text>{i}.</xsl:text>
        </xsl:if>
      </xsl:if>
    </xsl:for-each>
  </xsl:template>

  </xsl:stylesheet>
```

8.4 Το XML Schema για τα αρχεία του DeviceCreator

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">

   <xs:element name="allowedValue" type="xs:string"/>
   <xs:element name="minValue" type="xs:string"/>
   <xs:element name="maxValue" type="xs:string"/>
   <xs:element name="description" type="xs:string"/>

   <xs:attribute name="Manufacturer" type="xs:string"/>
   <xs:attribute name="OUI" type="xs:string"/>
   <xs:attribute name="ProductClass" type="xs:string"/>
   <xs:attribute name="SerialNumber" type="xs:string"/>
   <xs:attribute name="name" type="xs:string"/>
   <xs:attribute name="type" type="xs:string"/>
   <xs:attribute name="writable" type="xs:boolean"/>
   <xs:attribute name="readable" type="xs:boolean"/>
   <xs:attribute name="getFunction" type="xs:string"/>
   <xs:attribute name="setFunction" type="xs:string"/>
   <xs:attribute name="standAlone" type="xs:boolean"/>
   <xs:attribute name="varName" type="xs:string"/>
   <xs:attribute name="varValue" type="xs:string"/>
   <xs:attribute name="theirAddress" type="xs:boolean"/>
   <xs:attribute name="ourAddress" type="xs:boolean"/>

   <xs:element name="allowedValues">
      <xs:complexType>
         <xs:sequence>
            <xs:element ref="allowedValue" maxOccurs="unbounded"/>
         </xs:sequence>
      </xs:complexType>
   </xs:element>
   <xs:element name="allowedValueRange">
      <xs:complexType>
         <xs:sequence>
            <xs:element ref="minValue" minOccurs="0"/>
            <xs:element ref="maxValue" minOccurs="0"/>
         </xs:sequence>
      </xs:complexType>
   </xs:element>
   <xs:element name="stateVariable">
      <xs:complexType>
         <xs:sequence>
            <xs:element ref="allowedValues" minOccurs="0"/>
            <xs:element ref="allowedValueRange" minOccurs="0"/>
         </xs:sequence>
         <xs:attribute ref="name"/>
         <xs:attribute ref="type"/>
         <xs:attribute ref="writable" default="true"/>
         <xs:attribute ref="readable" default="true"/>
         <xs:attribute ref="standAlone" default="false"/>
         <xs:attribute ref="varName" default=""/>
         <xs:attribute ref="varValue" default=""/>
         <xs:attribute ref="theirAddress" default="false"/>
         <xs:attribute ref="ourAddress" default="false"/>
         <xs:attribute ref="setFunction" default=""/>
         <xs:attribute ref="getFunction" default=""/>
```

```
      </xs:complexType>
   </xs:element>
   <xs:element name="object">
      <xs:complexType>
         <xs:sequence>
            <xs:element ref="stateVariable" minOccurs="0"
maxOccurs="unbounded"/>
            <xs:element ref="object" minOccurs="0"
maxOccurs="unbounded"/>
         </xs:sequence>
         <xs:attribute ref="name"/>
      </xs:complexType>
   </xs:element>
   <xs:element name="device">
      <xs:complexType>
         <xs:sequence>
            <xs:element ref="object"/>
         </xs:sequence>
         <xs:attribute ref="Manufacturer" default=""/>
         <xs:attribute ref="OUI" default=""/>
         <xs:attribute ref="ProductClass" default=""/>
         <xs:attribute ref="SerialNumber" default=""/>
      </xs:complexType>
   </xs:element>
</xs:schema>
```

8.5 Μετατροπή από SOAP Fault σε HTML

```
<?xml version="1.0"?>
<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
soap:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:cwmp="urn:dslforum-org:cwmp-1-0">

      <xsl:output method="html"/>
      <xsl:strip-space elements="*"/>

      <xsl:template match="/">
       <html>
         <body>
           <xsl:apply-templates
         select="soap:Envelope/soap:Body/soap:Fault" />
         </body>
       </html>
      </xsl:template>

       <xsl:template match="soap:Fault">
     <font size="+1"> <em>
            <xsl:value-of select = "faultstring" />
            <xsl:text>: </xsl:text><br />
         </em>
       </font>
       <xsl:value-of select = "faultcode" />
       <xsl:text> error </xsl:text>
       <xsl:apply-templates select="detail/cwmp:Fault" />
      </xsl:template>

      <xsl:template match="cwmp:Fault" >
       <xsl:value-of select = "FaultCode" />
       <xsl:text>: </xsl:text>
     <xsl:value-of select = "FaultString" />
       <xsl:text>.</xsl:text><br />
       <xsl:apply-templates select="SetParameterValuesFault" />
      </xsl:template>

      <xsl:template match="SetParameterValuesFault">
       <xsl:text>In </xsl:text>
       <xsl:value-of select="ParameterName" />
     <xsl:text> error </xsl:text>
       <xsl:value-of select="FaultCode" />
       <xsl:text>: </xsl:text>
       <xsl:value-of select="FaultString" />
       <xsl:text>.</xsl:text><br />
      </xsl:template>
</xsl:stylesheet>
```

8.6 Ο πηγαίος κώδικας για μια απλή συσκευή

Η συσκευή αυτή έχει το όνομα Calc και έχει ένα αντικείμενο ως παιδί του ριζικού αντικειμένου με όνομα Add καθώς και το αντικείμενο ManagementServer που περιέχει τις ήδη γνωστές μεταβλητές κατάστασης URL και ConnectionRequestURL που αφορούν τη διεύθυνση του ACS και το connection string του πελάτη αντίστοιχα. Το αντικείμενο Add τώρα έχει τρεις μεταβλητές κατάστασης ως παιδιά, την num1, την num2 και την result. Όλες είναι ακέραιοι, ενώ η result είναι read-only. Μέσω του DeviceCreator, οι num1 και num2 ορίζονται ως stand alone και τους δίνονται αντίστοιχα ονόματα C μεταβλητών με αρχικη τιμή το 0. Η συσκευή αυτή, όπως φαίνεται ως τώρα θα προσθέτει τις τιμές των num1 και num2 και θα επιστρέφει το αποτέλεσμα στο result. Αφού δημιουργηθεί η συσκευή, ο προγραμματιστής αρκεί να προσθέσει τη γραμμή «value = num1 + num2;» στην κατάλληλη θέση του «custom.cpp». Το πρόγραμμα πλέον είναι έτοιμο, οι τιμές των num1 και num2 μπορούν να τεθούν με μηνύματα SetParameterValues ενώ το αποτέλεσμα επιστρέφεται χρήσιμοποιώντας ένα μήνυμα GetParameterValues με παράμετρο τη μεταβλητή result.

8.6.1 calc.dd

```
<?xml version="1.0" encoding="UTF-8"?>
<device>
  <object name="Calc">
    <description />
    <object name="ManagementServer">
      <description />
      <stateVariable name="URL" type="string(256)">
        <description />
      </stateVariable>
      <stateVariable name="ConnectionRequestURL" type="string(256)"
writable="false">
        <description />
      </stateVariable>
    </object>
    <object name="Add">
      <description />
      <stateVariable name="num1" type="int">
        <description />
      </stateVariable>
      <stateVariable name="num2" type="int">
        <description />
      </stateVariable>
      <stateVariable name="result" type="int" writable="false">
        <description />
      </stateVariable>
    </object>
  </object>
</device>
```

8.6.2 calc.dc

```
<?xml version="1.0" encoding="UTF-8"?>
<device Manufacturer="Serafeim" OUI="123456" ProductClass="Calculator"
SerialNumber="1212121">
```

```
  <object name="Calc">
    <object name="ManagementServer">
      <stateVariable name="URL" type="string(256)" standAlone="true"
varName="__ACSconnectionURL" varValue="http://127.0.0.1:80"
theirAddress="true" setFunction="fun1" getFunction="fun2" />
      <stateVariable name="ConnectionRequestURL" type="string(256)"
writable="false" standAlone="true" varName="__CPEconnectionString"
varValue="http://127.0.0.1:24582/CALC_CONN" ourAddress="true"
getFunction="fun2" />
    </object>
    <object name="Add">
      <stateVariable name="num1" type="int" standAlone="true" varName="num1"
varValue="0" setFunction="fun1" getFunction="fun2" />
      <stateVariable name="num2" type="int" standAlone="true" varName="num2"
varValue="0" setFunction="fun1" getFunction="fun2" />
      <stateVariable name="result" type="int" writable="false"
getFunction="fun2" />
    </object>
  </object>
</device>
```

8.6.3 client.h

```
#ifndef _CLIENT_H
#define _CLIENT_H

#include "ixml.h"

/************************ STA9ERES ************************
 * Parakatw dhlwnontai oi sta9eres pou xreiazontai         *
 ***********************************************************/
/** Eidh mhnumatwn **/
#define INFORM 10
#define INFORMRESPONSE 15
#define GETPARAMETERVALUES 20
#define GETPARAMETERVALUESRESPONSE 25
#define SETPARAMETERVALUES 30
#define SETPARAMETERVALUESRESPONSE 35
#define GETPARAMETERNAMES 40
#define GETPARAMETERNAMESRESPONSE 45
#define NOTIMPLEMENTEDMESSAGE 50

/** Eidh la9wn **/
#define METHOD_NOT_SUPPORTED 9000
#define REQUEST_DENIED 9001
#define INTERNAL_ERROR 9002
#define INVALID_ARGUMENTS 9003
#define RESOURCES_EXCEEDED 9004
#define INVALID_PARAMETER_NAME 9005
#define INVALID_PARAMETER_TYPE 9006
#define INVALID_PARAMETER_VALUE 9007
#define ATTEMPT_TO_SET_NON_WRITABLE 9008


/************************ DOMES ************************
 * Edw orizontai oi domes pou prokeitai na xrhsimopoih9oun *
 ***********************************************************/

typedef struct _DeviceIdStruct {
   char *manufactorer;
   char *OUI;
   char *productClass;
   char *serialNumber;
} DeviceIdStruct;
```

```
typedef struct _EventStruct {
   char *eventCode;
} EventStruct;

typedef struct _EventList {
   EventStruct *events;
   unsigned int size;
} EventList;

typedef struct _DateTime {
   unsigned int year;
   unsigned int month;
   unsigned int day;
   unsigned int hour;
   unsigned int min;
   unsigned int sec;
} DateTime;

typedef struct _ParameterValueStruct {
   char *name;
   char *value;

} ParameterValueStruct;

typedef struct _ParameterValueList {
   ParameterValueStruct *parameters;
   unsigned int size;
} ParameterValueList;

typedef struct _StringList {
   char **strings;
   unsigned int size;
} StringList;

typedef struct _SoapHeader {
   char *id;
   short holdRequests;
   short noMoreRequests;
} SoapHeader;

typedef struct _SoapMessage {
   SoapHeader *header;
   unsigned short type;
   IXML_Node *message;
} SoapMessage;


typedef struct _StateVariable {
   char *name;
   void (*getfun)(ParameterValueList *);
   void (*setfun)(ParameterValueList *);
   short int writable;
} StateVariable;

StateVariable *getStateVariable(char *);

char *respondTo(char *req);

int handleSetParameterValues(ParameterValueList *parameterList);
int handleGetParameterValues(ParameterValueList *parameterList);

IXML_Document *createCPEInform(void);

#endif //_CLIENT_H
```

8.6.4 client.cpp

```
/* Papastefanos Serafeim */
/* To programma tou pelath */

#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "ixml.h"
#include "client.h"
#include "soaparse.h"
#include "soapcrt.h"
#include "low.h"
#include "que.h"
#include "helpers.h"
#include "typechk.h"
#include "custom.h"

int handleConnection(SOCKET sock);

int main(void) {
   int listenport ;

   #ifdef _CWMP_WIN
   initWinsock();
   #endif

   listenport = getListeningPort();

   /** Connecting for the first time to the ACS */
   char addr[256];
   int port;
   getConnectionAddress(addr, &port);
   printf("Connecting to %s, port %d", addr, port);
   SOCKET sd = makeConnection(addr, port);
   if(sd > 0) {
      handleConnection(sd);
   } else { // La9os
      fprintf(stderr,"Cannot connect to %s:%d.", addr, port);
   }

   /* Now we start listening for connection-demand from the ACS */
   SOCKET lsd=createListeningSocket(listenport);
   if(lsd == 0) {
      printf("Cannot listen on %d", listenport);
      exit(0);
   }
   while(1) {
      printf("\nWaiting for a connection from the ACS\n");
      SOCKET sd = acceptConnection(lsd);
      if(sd == 0) { // La9os
         continue ;
      }
      if(checkConnectionString(getConnectionString(), sd) == 0 ) {
         printf("Did not receive the correct connection string.");
         shutdown(sd,2);
         #ifdef _CWMP_WIN
         closesocket(sd);
         #else
         close(sd);
         #endif
         continue;
      }  else {
         printf("Received the correct string, connecting to the acs. \n");
         shutdown(sd,2);
         #ifdef _CWMP_WIN
         closesocket(sd);
         #else
```

```
        close(sd);
        #endif

        char addr[256];
        int port;
        getConnectionAddress(addr, &port);
        printf("Connecting to %s, port %d", addr, port);

        SOCKET sd = makeConnection(addr, port);
        if(sd > 0) {
           handleConnection(sd);
        } else { // La9os
           fprintf(stderr,"Cannot connect to %s:%d.", addr, port);
        }
     }

  }

  return 0;
}


/* Epistrefei ena xml document me to inform request pou 9a steilei o
 * client mas ston ACS server. */
IXML_Document *createCPEInform(void) {
  // Arxika dhmiourgeitai to deviceId
  DeviceIdStruct *deviceId = getDeviceId();
  if(deviceId == NULL) {
     return NULL;
     printf("Error while trying to get the device id.");
  }
  // Sth sunexeia o xronos
  DateTime *currentTime = getCurrentTime();
  if(currentTime == NULL) {
     return NULL;
     printf("Error while trying to get current time.");
  }

  // To MaxEnvelopes kai to RetryCount
  int maxEnvelope=10; // Mporei na einai kai perissotero
  int retryCount=0;

  // H lista me tis parametrous pou prepei na stalloun sto inform
  ParameterValueList *parameterList = getInformParameterList();

  // H lista me ta events pou odhghsan sto connection
  EventList *event=newEventList(1);
  event->events[0].eventCode = strdup("6 CONNECTION REQUEST");

  // Telos, dhmiourgeitai ena mhnuma inform xwris header kai me ta stoixeia
  // pou oristhkan prohgoumenws
  IXML_Document *inform = createInform(NULL, *deviceId, event, maxEnvelope,
*currentTime, retryCount, parameterList);
  free(currentTime);
  free(deviceId);
  freeParameterValueList(parameterList);
  freeEventList(event);
  return inform;
}

/* H sunarthsh auth xeirizetai th sundesh me kapoion ACS server. Ws
 * parametro dexetai ena socket sto opoio uparxei mia energh sundesh
 * me ton server. */
int handleConnection(SOCKET sd) {
  unsigned int i=0;
  int r=0;
  unsigned int servermaxenvs=0; // O ari9mos twn fakelwn tou server
  int endTransaction=0; // 9a ginei 1 otan prepei na lhjei h dosolhyia
```

```
   Queue *pendingRequests = createQueue(); // Mia oura me ta requests
   SoapMessage *msg1 = NULL ;
   if(pendingRequests == NULL) {
      perror("malloc cannot create the pending requests queue");
      exit(0);
   }

   /* Kat' arxhn 9a prepei na stalei to inform request. To mono
    * shmantiko stoixeio pou fainetai na exei twra to inform einai to
    * maxenvelopes pou ka9orizei ton ari9mo twn mhnumatwn pou mporei
    * na dextei o client. */
   IXML_Document *doc1 = createCPEInform();
   StringList *m = newStringList(1);
   m->strings[0] = (char *)ixmlPrintNode( (IXML_Node*) doc1);
   sendRawMessage(sd, m);

   // Twra lamvanontai ta mhnumata apo ton server.
   StringList *messages = getMessages(sd);
   if(messages->size == 0) {
      goto informresponseerror;
   }


   // Prepei to prwto mhnuma na einai ena InformResponse
   msg1 = parseStart(messages->strings[0]);
   if( msg1!=NULL && msg1->type==INFORMRESPONSE) {
      r = parseInformResponse(msg1->message, &servermaxenvs);
      if(r!=0) {
         printf("Error %d while parsing the InformResponse message.\n", -r);
         goto informresponseerror;
      }
      printf("O server dexetai to polu %d RPC sto idio mhnuma.\n",
servermaxenvs);
   } else {
      // Edw pame an den er9ei swsta to InformResponse mhnuma apo ton server
      // opote kai h sundesh kleinei . Xrhsimopoieitai ena label kai
      // h entolh goto gia aplothta: Anti tou goto 9a mporousan na
      // epanalhg9oun oi 5 grammes pou akolou9oun to label.
informresponseerror:
      fprintf(stderr, "O Client den elave ena mhnuma InformResponse.\n");
      fprintf(stderr, "Kleinei h sundesh.\n");
      shutdown(sd,2);
      #ifdef _CWMP_WIN
      closesocket(sd);
      #else
      close(sd);
      #endif
      return 0;
   }

   /* Sthn oura pendingRequests 9a uparxoun ta requests tou server pou den
    * exoun akoma ejuphreth9ei. An auth h oura einai adeia se kapoio shmeio
    * tote o client 9a stellei ena adeio http post.  */

   // Arxika prosti9entai sthn oura ta mhnumata pou lhf9hkan arxika, afou
   // o server, ektos apo to InformResponse mporei na eixe steilei kai alla
   // mhnumata
   for(i=1;i<messages->size;i++) enque(strdup(messages->strings[i]),
pendingRequests);
   freeStringList(messages);


   while( endTransaction == 0) {
      if(isEmpty(pendingRequests) ) {
         printf("H oura einai adeia, etsi stelletai ena adeio mhnuma.\n");
         StringList *empty = newStringList(0);
         sendRawMessage(sd, empty);
         freeStringList(empty);
```

```
     } else {
        // To num 9a isoutai eite me ton ari9mo mhnumatwn pou den exoun
        // ejuphreth9ei eite me ton ari9mo twn fakelwn tou server, analoga me
        // to poio ek twn duo einai megalutero.
        unsigned int num=(pendingRequests->size >
servermaxenvs)?servermaxenvs:pendingRequests->size;

        // Sth lista msgs 9a apostaloun ta mhnumata pou prepei na stalloun
        StringList *msgs = newStringList(num);
        if(msgs==NULL) {
           perror("malloc");
           exit(-1);
        }

        for(i=0;i<num;i++) {
           // To req einai to mhnuma pou exei seira na ejuphreth9ei
           char *req = deque(pendingRequests);
           // To res einai h apanthsh sto mhnuma auto
           char *res = respondTo(req);
           if(res == NULL) {
              printf("Problem while trying to respond to %s", req);
              printf("The connection will close");
              free(req);
              freeStringList(msgs);
              endTransaction = 1 ;
              goto out;
           }
           free(req);

           // To res prosti9etai sth lista me ta nhmunata pou 9a stalloun
           msgs->strings[i] = strdup(res);
           free(res);
        }

        // Edw pleon stellontai ta mhnumata
        sendRawMessage(sd,msgs);
        freeStringList(msgs);
     }

     // Twra, lamvanontai ta mhnumata pou esteile o server
     StringList *messages = getMessages(sd);
     if(messages->size == 0) {
        if( isEmpty(pendingRequests) ) {
           printf("Lavame keno mhnuma kai den uphrxan pending requests ");
           printf("apo to server, ara to transaction 9a lhjei.\n");
           endTransaction = 1;
        } else {
           printf("Lavame men keno mhnuma, uparxoun omws pending requests ");
           printf("kai etsi de 9a kanoume tipota. \n");
        }
     } else {
        // An lavame mhnuma tote to pros9etoume sthn oura kai arxizoume
        // pali apo thn arxh to while
        for(i=0;i<messages->size;i++) enque(strdup(messages->strings[i]),
pendingRequests);

     }
     freeStringList(messages);

  }
out: // Edw 9a er9oume otan dhmiourgh9ei kapoio sovaro provlhma to opoio
  // o client den mporei na xeiristei, gia paradeigma an mas er9ei kapoio
  // mhnuma to opoio den einai SOAP. Se mia tetoia periptwsh de 9a
  // apanthsoume me ena fault response, alla 9a kleisoume th sundesh, afou
  // profanws o server den einai katallhlos gia to TR69. Epishs, edw pame
  // otan h sundesh lhjei (xwris jump).
  shutdown(sd,2);
  #ifdef _CWMP_WIN
```

```
   closesocket(sd);
   #else
   close(sd);
   #endif
   free(pendingRequests);
   return 0;
}


int handleSetParameterValues(ParameterValueList *parameterList) {
   unsigned int i,j,k=0;

   int *errors = (int *)malloc(parameterList->size*sizeof(int) );

   // O elegxos autos de xreiazetai, afou exei ginei pio prin
   //if(checkSetParameterValues(parameterList, errors)!=0) {
      // Error
   //}

   int *grouparray = (int *)malloc(parameterList->size*sizeof(int) );
   for(i=0;i<parameterList->size;i++) grouparray[i]=-1;


   for(i=0;i<parameterList->size;i++) {
      if(grouparray[i]!=-1) continue;
      StateVariable *sv=getStateVariable(parameterList->parameters[i].name );
      if(sv==NULL) return -1; // This error has to be found earlier
      grouparray[i]=k;
      for(j=i+1;j<parameterList->size;j++) {
         StateVariable *tmp=getStateVariable(parameterList->parameters[j].name
);
         if(tmp==NULL) return -1; // This error has to be found earlier
         if(tmp->setfun == sv->setfun) grouparray[j]=k;
      }
      k++;
   }

   for(i=0;i<k;i++) {
      int m=0;
      unsigned int groupsize = 0;
      for(j=0;j<parameterList->size;j++) if(grouparray[j]==(int)i)
groupsize++;

      ParameterValueList * pvl = newParameterValueList(groupsize);

      for(j=0;j<parameterList->size;j++) if(grouparray[j]==(int)i) {
         pvl->parameters[m].name=strdup(parameterList->parameters[j].name);
         pvl->parameters[m].value=strdup(parameterList->parameters[j].value);
         m++;
      }
      j=0;
      while(grouparray[j]!=(int)i) j++;
      StateVariable *tmp=getStateVariable(parameterList->parameters[j].name);

      tmp->setfun(pvl);

      freeParameterValueList(pvl);
   }

   free(grouparray);

   return 0;
}

int handleGetParameterValues(ParameterValueList *parameterList) {
   unsigned int i,j,k=0;
   /* To grouparray einai enas pinakas apo int me tosa stoixeia osa
   exei kai to parameterList kai prokeitai na apo9hkeutei se auto
```

```
   o ari9mos tou group sto opoio anhkei h antistoixh parametros.
   Arxika, ta stoixeia tou grouparray arxikopoiountai sto -1.*/
   int *grouparray = (int *)malloc(parameterList->size*sizeof(int) );
   for(i=0;i<parameterList->size;i++) grouparray[i]=-1;

   /* Me to for auto ginetai h ana9esh twn parametrwn se group. H
   ergasia xwrizetai se ena ejwteriko kai ena eswteriko for loop. O
   Ejwterikos vroxos, pairnei mia mia tis parametrous apo th list kai
   elegxei an autes anhkoun hdh se kapoio group (an dld to antistoixo
   stoixeio tou grouparray einai diaforo tou -1). An oi sunarthseis
   anhkoun se group de xreiazetai na ginei tipota opote sunexizoume
   sthn epomenh parametro. An omws h parametros den anhkei se kapoio
   group tote anati9etai h parametros sto group k. To k einai mia
   metavlhth pou krataei ton ari9mo twn group, exei arxikh timh 0 kai
   aujanei ka9e fora pou pername se neo group. Sth sunexeia pername sto
   eswteriko for. Edw, ejetazontai oles oi parametroi apo thn trexousa
   kai pera, kai an vre9ei kapoia me idio get function me thn trexousa
   tote topo9etoume th nea parametro sto group k. Afou teleiwsei kai
   autos o elegxos to k aujanetai mias kai oles oi parametroi pou
   anhkoun sto group auto exoun shmeiw9ei. */
   for(i=0;i<parameterList->size;i++) {
      if(grouparray[i]!=-1) continue;
      StateVariable *sv=getStateVariable(parameterList->parameters[i].name );
      if(sv==NULL) { //Error, cannot find this variable
         free(grouparray);
         return -INVALID_PARAMETER_NAME;
      }
      grouparray[i]=k;
      for(j=i;j<parameterList->size;j++) {
         StateVariable *tmp=getStateVariable(parameterList->parameters[j].name
);
         if(tmp==NULL) { //Error, cannot find this variable
            free(grouparray);
            return -INVALID_PARAMETER_NAME;
         }
         if(tmp->getfun == sv->getfun) grouparray[j]=k;
      }
      k++;
   }

   /* Me to for auto kalountai me ton swsto tropo oi get functions gia tis
   parametrous. Parathroume oti auto to for 9a epanalhf9ei k fores, osa einai
   kai ta group parametrwn pou exoun dhmiourgh9ei sto prohgoumeno vhma. */
   for(i=0;i<k;i++) {
      int m=0;
      unsigned int groupsize = 0;
      /* Arxika, gia to i group, elegxontai oles oi parametroi. An h
      j anhkei sto i group tote aujanei kata 1 h timh ths groupsize,
      h opoia kai ka9orizei ton ari9mo twn parametrwn pou anhkoun i. */
      for(j=0;j<parameterList->size;j++) if(grouparray[j]==(int)i)
groupsize++;

      /* Sth sunexeia, dhmiourgeitai mia nea lista parametrwn, h params. Auth
      exei mege9os iso me to group size kai se authn antigrafetai to onoma
      ka9e mias apo tis parametrous pou anhkoun sto group i. */
      ParameterValueList *params = newParameterValueList(groupsize);
      for(j=0;j<parameterList->size;j++) if(grouparray[j]==(int)i) {
         params->parameters[m].name = strdup(parameterList-
>parameters[j].name);
         m++;
      }

      /* Edw, xrhsimopoiwntas th metavlhth j ws deikth, vriskoume thn prwth
      metavlhth pou anhkei sto group i (opoiadhpote metavlhth pou anhkei sto
      group i fusika 9a mas arkouse, kai kaloume thn getfun gia th metavlhth,
      pernwntas ths ws parametro thn params, thn opoia dhmiourghsame sto
      prohgoumeno vhma. Etsi, sth getfun 9a perastei ena ParameterValueStruct
      sto opoio ta onomata einai ta onomata twn parametrwn pou anhkoun sto
```

```
      group i, enw oi times 9a prepei na sumplhrw9oun apo to xrhsth. */
      j=0;
      while(grouparray[j]!=(int)i) j++;
      StateVariable *tmp=getStateVariable(parameterList->parameters[j].name
);
      tmp->getfun(params);

      /* Se auto to for, ginetai h antistrofh ergasia apo to prohgoumeno.
      Ka9e mia apo tis times pou epestreye h getfun mesa sto params
      antigrafetai pisw sthn arxikh lista parametrwn. */
      m=0;
      for(j=0;j<parameterList->size;j++) if(grouparray[j]==(int)i) {
         parameterList->parameters[j].value=strdup(params-
>parameters[m].value);
         m++;
      }

      freeParameterValueList(params);
   }
   free(grouparray);
   return 0;
}

StateVariable *getStateVariable(char *name) {
   unsigned int i;
   for(i=0;i<NUMOFVARS;i++)  if(strcmp(deviceVariables[i].name,name)==0)
return (deviceVariables+i);
   return NULL;
}


char *respondTo(char *req) {

   SoapMessage *s = parseStart( req );
   if(s == NULL) {
      printf("Did not receive a soap message. ");
      return NULL;
   }
   unsigned int i;

   switch(s->type) {
      case GETPARAMETERNAMES: {
         int r=0;
         SoapHeader *sh = NULL;
         char *pathName;
         unsigned short nextLevel;
         r=parseGetParameterNames(s->message, &pathName, &nextLevel);

         if(s->header != NULL)  {
            sh = (SoapHeader *)malloc(sizeof(SoapHeader) );
            sh->id = s->header->id;
            sh->noMoreRequests = 0;
            sh->holdRequests = 0;
            free(s->header);
         }

         if(r!=0) {
            /* Epistrefetai ena fault response. To la9os pou
            mporei na sumvei se auto to shmeio einai na einai
            la9os ta arguments tou mhnumatos. */
            IXML_Document *doc = createFaultResponse(sh, -r);
            char *tmp = (char *)ixmlPrintNode( (IXML_Node*) doc);
            ixmlNode_free((IXML_Node*) doc);
            return tmp;
         }

         IXML_Document *doc = createGetParameterNamesResponse(sh, pathName,
nextLevel);
```

```
   char *tmp = (char *)ixmlPrintNode( (IXML_Node*) doc);
   ixmlNode_free((IXML_Node*) doc);
   free(sh);
   return tmp;

} break;
case GETPARAMETERVALUES: {
   int r=0;
   StringList *pnames = newStringList(0);
   //unsigned int num=0;
   SoapHeader *sh = NULL;

   r=parseGetParameterValues(s->message, pnames);
   if(s->header != NULL)  {
      sh = (SoapHeader *)malloc(sizeof(SoapHeader) );
      sh->id = s->header->id;
      sh->noMoreRequests = 0;
      sh->holdRequests = 0;
      free(s->header);
   }
   if(r!=0) {
      /* Epistrefetai ena fault response. To la9os pou
      mporei na sumvei se auto to shmeio einai na einai
      la9os ta arguments tou mhnumatos. */
      IXML_Document *doc = createFaultResponse(sh, -r);
      char *tmp = (char *)ixmlPrintNode( (IXML_Node*) doc);
      ixmlNode_free((IXML_Node*) doc);
      freeStringList(pnames);
      return tmp;
   }



   ParameterValueList *parList = newParameterValueList(pnames->size);
   if(parList == NULL) {
      perror("malloc");
      exit(-1);
   }

   for(i=0;i<pnames->size;i++) {
      parList->parameters[i].name=pnames->strings[i];
      parList->parameters[i].value=NULL;
   }

   r=handleGetParameterValues(parList);
   if(r!=0) {
      /* Epistrefetai ena fault response. To la9os pou
      mporei na sumvei se auto to shmeio einai na mhn
      uparxei kapoia apo tis state variables pou zhtountai. */
      IXML_Document *doc = createFaultResponse(sh, -r);
      char *tmp = (char *)ixmlPrintNode( (IXML_Node*) doc);
      ixmlNode_free((IXML_Node*) doc);
      freeParameterValueList(parList);
      free(sh);
      return tmp;
   } else {
      IXML_Document *doc=createGetParameterValuesResponse(sh, parList);
      char *tmp = (char *)ixmlPrintNode( (IXML_Node*) doc);
      ixmlNode_free((IXML_Node*) doc);
      freeParameterValueList(parList);
      free(sh);
      return tmp;
   }

} break;
case SETPARAMETERVALUES: {
   int r = 0;
   ParameterValueList *parList = newParameterValueList(0);
```

```
   char *paramKey;
   SoapHeader *sh = NULL;
   if(s->header != NULL)  {
      sh = (SoapHeader *)malloc(sizeof(SoapHeader) );
      sh->id = s->header->id;
      sh->noMoreRequests = 0;
      sh->holdRequests = 0;
      free(s->header);
   }

   r=parseSetParameterValues(s->message, parList, &paramKey);

   if(r!=0) {
      /* Epistrefetai ena fault response. To la9os pou
      mporei na sumvei se auto to shmeio einai na einai
      la9os ta arguments tou mhnumatos. */
      IXML_Document *doc = createFaultResponse(sh, -r);
      char *tmp = (char *)ixmlPrintNode( (IXML_Node*) doc);
      ixmlNode_free((IXML_Node*) doc);
      free(parList);
      free(sh);
      return tmp;
   }

   int *errors = (int *)malloc(parList->size*sizeof(int) );
   if(checkSetParameterValues(parList, errors)!=0) {
      /* Epistrefetai ena SetParameterNames fault response. To
      la9os pou mporei na sumvei se auto to shmeio einai na einai
      la9os ta arguments tou mhnumatos. */
      IXML_Document *doc = createSetFaultResponse(sh, parList, errors);
      char *tmp = (char *)ixmlPrintNode( (IXML_Node*) doc);
      ixmlNode_free((IXML_Node*) doc);
      free(parList);
      free(errors);
      free(sh);
      return tmp;
   }

   free(errors);

   handleSetParameterValues(parList);

   IXML_Document *doc = createSetParameterValuesResponse(sh, 0);
   char *tmp = (char *)ixmlPrintNode( (IXML_Node*) doc);
   ixmlNode_free((IXML_Node*) doc);
   free(sh);
   return tmp;

} break;

case NOTIMPLEMENTEDMESSAGE: {
   SoapHeader *sh = NULL;

   if(s->header != NULL)  {
      sh = (SoapHeader *)malloc(sizeof(SoapHeader) );
      sh->id = s->header->id;
      sh->noMoreRequests = 0;
      sh->holdRequests = 0;
      free(s->header);
   }

   IXML_Document *doc = createFaultResponse(sh, METHOD_NOT_SUPPORTED );
   char *tmp = (char *)ixmlPrintNode( (IXML_Node*) doc);
   ixmlNode_free((IXML_Node*) doc);
   free(sh);
   return tmp;
} break;
```

```
      default: {
         printf("Unknown message. Control shouldn't reach this point.\n");
      } break;
   }
   printf("Problem, returning null.\n");
   return NULL;
}
```

8.6.5 helpers.h

```
#ifndef _HELPERS_H
#define _HELPERS_H

/* Papastefanos Serafeim */
/* Boh9htikes sunarthseis */

#include "client.h"


/************************
 Diafores voh9htikes sunarthseis
 ************************/

char *datetoasci(DateTime *d);
DateTime *ascitodate(char *);
DateTime *getCurrentTime();
void viewParameterValueStruct(ParameterValueStruct pv);

ParameterValueList *newParameterValueList(unsigned int size);
void freeParameterValueList(ParameterValueList *);

EventList *newEventList(unsigned int size);
void freeEventList(EventList *);

StringList *newStringList(unsigned int size);
void freeStringList(StringList *);


char *getErrorByNum(int num);

#endif //_HELPERS_H
```

8.6.6 helpers.cpp

```
/* Papastefanos Serafeim */
/* Diafores voh9htikes sunarthseis */

#include <stdio.h>
#include <stdlib.h>
#include "ixml.h"
#include "client.h"
#include "soaparse.h"
#include "soapcrt.h"
#include "helpers.h"
#include "time.h"
#include "string.h"


/* Se auto to array uparxei mia antistoixish metaju twn ari9mwn sfalmatos
 * tou TR69 kai tou mhnumatos pou 9a prepei na emfanistei sthn o9onh.
 * Upoti9etai oti o pinakas jekina apo to 9000. */
static char *errorStrings[] = {
   "Method not supported", //9000
   "Request denied",  //9001
   "Internal error", //9002
```

```
   "Invalid arguments", //9003
   "Resources exceeded", //9004
   "Invalid parameter name", //9005
   "Invalid parameter type", //9006
   "Invalid parameter value", //9007
   "Attempt to set a non-writable parameter" //9008
};

/* Emfanizei sthn o9onh to pv */
void viewParameterValueStruct(ParameterValueStruct *pv) {
   printf("Name: %s\n", pv->name);
   printf("Value: %s\n", pv->value);
}

/* Metatrepei mia hmeromhnia se string. Ginetai xrhsh statikhs mnhmhs sthn
 * opoia apo9hkeuetai to string pou 9a epistrafei. Auto ginetai gia na mhn
 * xreiazetai na eleu9erwnetai to memmory buffer pou 9a epestrefe auth h
 * entolh, opws 9a eprepe na ginei an eixame xrhsimopoihsei mnhmh sto heap.
 * H xrhsh statikhs mnhmhs ektos apo to pleonekthma pou molis shmeiw9hke
 * exei kai ena mikro meionekthma: Den 9a prepei na klh9ei gia deuterh
 * fora h datetoasci an prwta den exei antigrafei se enan allo buffer to
 * apotelesma apo thn prwth klhsh ths. */
char *datetoasci(DateTime *d) {
   static char buf[20];
   sprintf(buf,"%04d-%02d-%02dT%02d:%02d:%02d", d->year, d->month, d->day, d-
>hour, d->min, d->sec );
   buf[19]=0;
   return buf;
}

/* Metatrepei ena string se DateTime. Opws fainetai to string prepei na
 * exei polu austhrh domh gia na douleyei swsta auth h sunarthsh. */
DateTime *ascitodate(char *buf) {
   DateTime *d = (DateTime *)malloc(sizeof(DateTime) );
   d->year = atoi(buf);
   d->month= atoi(&buf[5]);
   d->day = atoi(&buf[8]);
   d->hour = atoi(&buf[11]);
   d->min = atoi(&buf[14]);
   d->sec = atoi(&buf[17]);
   return d;
}

/* Dhmiourgei mia nea domh ParameterValueList mege9ous size. */
ParameterValueList *newParameterValueList(unsigned int size) {
   ParameterValueList *pl = (ParameterValueList
*)malloc(sizeof(ParameterValueList) );
   if(pl==NULL) {
      printf("Memmory errors (ParameterValueList creation)");
      return NULL;
   }
   pl->size=size;

   pl->parameters=(ParameterValueStruct
*)malloc(size*sizeof(ParameterValueStruct) );
   if(pl->parameters == NULL) {
      printf("Memmory errors (ParameterValueList creation)");
      return NULL;
   }

   for(unsigned int i=0;i<size;i++) {
      pl->parameters[i].name = NULL;
      pl->parameters[i].value = NULL;

   }

   return pl;
}
```

```
/* Eleu9erwnei thn ParameterValueList pvl */
void freeParameterValueList(ParameterValueList *pvl) {
   unsigned int i;
   for(i=0;i<pvl->size;i++) {
      free(pvl->parameters[i].name);
      free(pvl->parameters[i].value);

   }
   if(pvl->size == 0) {
      free(pvl);
      return;
   }
   free(pvl->parameters);
   free(pvl);
}

/* Dhmiourgei mia nea EventList mege9ous size */
EventList *newEventList(unsigned int size) {
   EventList *el = (EventList *)malloc(sizeof(EventList) );
   if(el==NULL) {
      printf("Memmory errors (EventList creation)");
      return NULL;
   }
   el->size=size;

   el->events=(EventStruct *)malloc(size*sizeof(EventStruct) );
   if(el->events == NULL) {
      printf("Memmory errors (EventList creation)");
      return NULL;
   }
   return el;
}

/* Eleu9erwnei thn EventList evl */
void freeEventList(EventList *evl) {
   unsigned int i;
   for(i=0;i<evl->size;i++) {
      free(evl->events[i].eventCode);
   }
   if(evl->size == 0) {
      free(evl);
      return;
   }
   free(evl->events);
   free(evl);
}

/* Dhmiourgei mia nea StringList mege9ous size */
StringList *newStringList(unsigned int size) {
   StringList *sl = (StringList *)malloc(sizeof(StringList) );
   if(sl==NULL) {
      printf("Memmory errors (StringList creation)");
      return NULL;
   }
   sl->size=size;

   sl->strings=(char **)malloc(size*sizeof(char *) );
   if(sl->strings == NULL) {
      printf("Memmory errors (StringList creation)");
      return NULL;
   }
   return sl;
}

/* Eleu9erwnei th StringList stl */
void freeStringList(StringList *stl ) {
   unsigned int i;
```

```
   for(i=0;i<stl->size;i++) {
      free(stl->strings[i]);
   }
   if(stl->size == 0) {
      free(stl);
      return;
   }
   free(stl->strings);
   free(stl);
}

/* H sunarthsh auth epistrefei mia domh DateTime me thn wra tou upologisth.
 * Xrhsimopoiei thn ansi sunarthsh ltime h opoia epistrefei mia domh tm. */
DateTime *getCurrentTime(void) {
   DateTime *currentTime = (DateTime *)malloc(sizeof(DateTime) );
   if(currentTime == NULL) return NULL;
   time_t t = time(NULL);
   struct tm *ltime = localtime(&t);
   if(ltime == NULL) return NULL;
   currentTime->day=ltime->tm_mday;
   currentTime->hour=ltime->tm_hour;
   currentTime->min=ltime->tm_min;
   currentTime->month=ltime->tm_mon;
   currentTime->sec=ltime->tm_sec;
   currentTime->year=ltime->tm_year+1900;
   return currentTime;
}

/* Epistrefei mia perigrafh tou sfalmatos, analoga me ton ari9mo tou.
 * Ginetai xrhsh tou pinaka errorStrings pou oristhke pio panw. */
char *getErrorByNum(int num) {
   num-=9000;
   if( (num < 0)||(num > 8) ) return NULL;
   else return errorStrings[num];
}
```

8.6.7 low.h

```
/* Papastefanos Serafeim */
/* Sunarthseis gia epikoinwnia */

#ifndef _LOW_H
#define _LOW_H

#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#include <string.h>
#ifdef _CWMP_WIN
//Windows version includes
#include <windows.h>
#include <winsock.h>
//#include <conio.h>
#else
// Unix version include
#include <sys/types.h>
#include <sys/time.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <netdb.h>

#include <unistd.h>
#include <signal.h>

#endif
```

```
#include "helpers.h"
#include "client.h"

#ifndef _CWMP_WIN // To UNIX den exei defined ton tupo SOCKET
#define SOCKET unsigned int
#endif

#define MAXBUF 256 /* H sta9era MAXBUF exei to megisto ari9mo xarakthrwn
pou epitrepetai na periexei mia grammh sthn epikefalida tou HTTP mhnumatos.
Mporoume na ths dwsoume opoia timh 9eloume. */

/* Stellei ta StringList ws HTTP mhnuma - ka9e StringList einai kai ena
oloklhro soap message */
void sendRawMessage(SOCKET sd, StringList *);

/* Diavazei ena mhnuma pou elave - isws uparxoun perissotera apo ena
mhnumata */
char *getRawMessage(SOCKET sd);

/* Xwrizei ena mhnuma pou isws periexei perissotera apo ena soap envelopes
se epimerous mhnumata */
StringList *splitRawMessage(char *rawMessage);

/* Briskei ton ari9mo twn epimerous soap mhnumatwn apo ta opoia apoteleitai
ena lhf9en mhnuma */
unsigned int getNumOfMessages(char *rawMessage);

/* Xrhsimopoiei tis treis parapanw sunarthseis gia na epistrefei ap'eu9eias
ta mhnumata pou esteile o server */
StringList *getMessages(SOCKET sd);

/* Diavazei mia grammh - xrhsimopoieitai gia tis epikefalides tou http */
char *getLine(SOCKET sd);

/* Epistrefei ena string list me thn epikefalida - xreiazetai to mhkos tou
mhnumatos */
StringList *createHeaders(unsigned int msglen);

/* Aparaithto gia to winsock */
void initWinsock(void);

/* Sundesh sth d/sh kai th 9ura pou perniountai ws parametroi. Xrhsimo gia
th leitourgia tou client */
SOCKET makeConnection(char *address, int port);

/* Dhmiourgia listening socket sth 9ura port, gia leitourgia server */
SOCKET createListeningSocket(int port);

/* Anamonh sto listening socket gia incoming connections, gia server */
SOCKET acceptConnection(SOCKET listening);

/* Elegxei an o ACS esteile to katallhlo mhnuma ston client, epistrefei 1 an
ok, 0 an oxi */
int checkConnectionString(char *, SOCKET);

#endif //_LOW_H
```

8.6.8 low.cpp

```
/* Papastefanos Serafeim */
/* Sunarthseis gia epikoinwnia
 * Sto arxeio auto uparxoun non-ansi shmeia kai einai to monadiko
 * sto opoio uparxoun diafores metaju ths ekdoshs gia linux kai gia
 * windows. Oles oi sunarthseis epistrefoun 0 an sumvei kapoio la9os. */

#include "low.h"
#include "helpers.h"
```

```
/* H sunarthsh auth arxikopoiei ta windows sockets. Xreiazetai mono
   sthn windows ekdosh tou client. */

#ifdef _CWMP_WIN // Windows

void initWinsock(void) {
   WSADATA wsaData;
   WORD version;
   int error;

   version=MAKEWORD(2,0);
   error=WSAStartup(version, &wsaData);
   if(error!=0) {
      fprintf(stderr, "Error while initializing winsock2.");
      exit(-1);
   }

   if( (LOBYTE(wsaData.wVersion)!=2)||(HIBYTE(wsaData.wVersion)!=0) ) {
      WSACleanup();
      fprintf(stderr, "Error while initializing winsock2, wrong version.");
      exit(-1);
   }

}


/* Me th sunarthsh auth o client sundeetai sth d/sh address kai th 9ura port
*/
SOCKET makeConnection(char *address, int port) {
   SOCKET sd; // connecting socket
   struct sockaddr_in server; // connecting address

   if ( ( sd=socket(AF_INET, SOCK_STREAM, 0) ) == INVALID_SOCKET ) {
      perror("cannot create socket");
      printf("%d", WSAGetLastError() );
      return 0;
   }

   /* set up sockaddr_in */
   server.sin_family=AF_INET;
   server.sin_port=htons(port);
   server.sin_addr.s_addr=inet_addr(address);
   memset(&(server.sin_zero),'\0',8);
   printf("Sending request...\n");
   /* Connect */
   if (connect(sd, (struct sockaddr *)&server, sizeof(server))==
INVALID_SOCKET) {
      printf("cannot connect to server");
      printf("%d", WSAGetLastError() );
      return 0;
   }
   return sd;
}

/* Epistrefei ton ari9mo twn epimerous SOAP mhnumatwn pou stal9hkan me
 * ena HTTP response apo ton server. */
unsigned int getNumOfMessages(char *rawMessage) {
   int i=0;
   if(rawMessage == NULL) return 0;
   while(strstr(rawMessage, "<soap:Envelope") ) {
      rawMessage = strstr(rawMessage, "</soap:Envelope>" );
      rawMessage = rawMessage+strlen("</soap:Envelope>");
      i++;
   }
   return i;
}
```

```
/* Spazei ena HTTP mhnuma pou periexei 1 h perissotera SOAP mhnumata sta
 * epimerous SOAP mhnumata. */
StringList *splitRawMessage(char *rawMessage) {
   unsigned int i;
   unsigned int num = getNumOfMessages(rawMessage);
   if(num==0) return NULL;
   StringList *messages = newStringList(num);

   for(i=0;i<num;i++) {
      int l = strstr(rawMessage, "</soap:Envelope>") - rawMessage +
strlen("</soap:Envelope>");
      messages->strings[i]=(char *)malloc( (l+1)*sizeof(char) );
      strncpy(messages->strings[i],rawMessage,l);
      messages->strings[i][l]=0;
      rawMessage = strstr(rawMessage, "</soap:Envelope>") +
strlen("</soap:Envelope>");
   }
   return messages;
}

/* Dhmiourgei ena socket to opoio akouei sth 9ura port */
SOCKET createListeningSocket(int port) {
   SOCKET lsd;
   struct sockaddr_in sin;
   int slen=sizeof(struct sockaddr_in);
   /* Create a listening socket */
   if ( ( lsd=socket(AF_INET, SOCK_STREAM, 0) ) == (int)INVALID_SOCKET ) {
      perror("cannot create listening socket");
      return 0;
   }

   /* Bind listening socket to port */
   sin.sin_family=AF_INET;
   sin.sin_port=htons(port);
   sin.sin_addr.s_addr=htonl(INADDR_ANY); /* IP d/sh tou susthmatos mas   */
   if (bind(lsd, (struct sockaddr *)&sin, slen ) == (int)INVALID_SOCKET  ) {
      perror("cannot bind listening socket");
      return 0;
   }

   /* Initiate a listen queue */
   if (listen(lsd, 1) == (int)INVALID_SOCKET  ) {
      perror("cannot listen o socket");
      return 0;
   }
   return lsd;
}

/* Kanei accept mia sundesh */
SOCKET acceptConnection(SOCKET lsd) {
   SOCKET sd;
   int slen=sizeof(struct sockaddr_in);
   struct sockaddr_in in_sa; /* remote address */
// printf("\nServer ready for requests...\n");
   if( (sd=accept(lsd, (struct sockaddr *)&in_sa, &slen)) == INVALID_SOCKET )
{
      perror("Cannot accept the connection");
      return 0;
   }
   printf("Got a connection.\n");
   return sd;
}

/* Epistrefei thn epomenh grammh apo to socket. Xreiazetai mono kata
 * thn anagnwsh ths epikefalidas tou HTTP mhnumatos. Epistrefetai enas
 * NULL pointer se periptwsh sfalmatos. H sunarthsh auth ousiastika
 * diavazei sunexws enan enan tous xarakthres apo to socket mexri na
 * diavasei ton CR (\r). Sth sunexeia, ejetazei an o epomenos xarakthras
```

```
 * einai o LF (\n) opote jeroume oti h grammh exei teleiwsei, alliws
 * sunexizetai h anagnwsh xarakthrwn. */
char *getLine(SOCKET sd) {
   char c;
   char buf[MAXBUF];
   int i=0;
   while(1) {
      if( recv(sd, &c, 1, 0) == SOCKET_ERROR ) {
         printf("%d.", WSAGetLastError() );
         perror("recv, CANNOT READ 1 char");
         return NULL;
      };

      buf[i++]=c;
      if(i==MAXBUF-1) {
         printf("HTTP header line too long");
         return NULL;
      }

      if(c=='\r') { // GOT CR
         recv(sd, &c, 1, 0);
         if(c == '\n') { // GOT LF
            buf[i++]=c;
            break ;
         }
      }
   }
   // Gia na ftasoume edw exei lhf9ei oloklhrh grammh
   char *tmp = (char *)malloc((i+1)*sizeof(char) );
   strncpy(tmp, buf, i);
   tmp[i]='\0';
   return tmp;
}

/* H sunarthsh auth diavazei ena HTTP mhnuma apo to socket. Arxika,
 * diavazei grammh-grammh thn epikefalida mexri na vrei th grammh
 * Content-Length: h opoia kai dinei to mege9os tou HTTP mhnumatos
 * to opoio kai apo9hkeuei. Sth sunexeia, sunexizei na diavazei grammes
 * mexri na vrei mia kenh grammh. Apo ekei kai pera, sumfwna me to HTTP
 * prwtokollo, 9a uparxoun tosoi xarakthres osoi kai to mege9s tou
 * mhnumatos pou diavasthke pio prin, opote kai diavazontai oloi, me
 * th xrhsh epimonou tropou wste na einai sigouro oti de 9a lhf9ei miso
 * mhnuma. */
char *getRawMessage(SOCKET sd) {
   int len;
   char *tmp;
   while(1) {
      tmp=getLine(sd);
      //puts(tmp);
      if(tmp == NULL) return NULL;
      if(strstr(tmp, "Content-Length:") ) {

         len=atoi(&tmp[15]);
         //if(len==0) return NULL;
      }

      if(tmp[0]=='\r') { //Adeia grammh
         free(tmp);
         break;
      }
      free(tmp);
   }

   if(len==0) return NULL;
   char *raw=(char *)malloc((len+1)*sizeof(char ) ) ;
   int totalbytes=0;
   int bytes=0;
   while(totalbytes<len) {
```

```
      bytes=recv(sd, raw+totalbytes, len-totalbytes, 0);
      totalbytes+=bytes;
   }
   raw[len]=0;

   return raw;
}

/* Epistrefei ta SOAP mhnumata pou uparxoun sto socket xrhsimopoiwntas
 * tis getRawMessage() kai splitRawMessage */
StringList *getMessages(SOCKET sd) {
   StringList *slist;
   char *buf = getRawMessage(sd);
   if(buf == NULL) {
      slist = newStringList(0);
      return slist;
   }
   slist = splitRawMessage(buf);
   free(buf);
   return slist;
}

/* Dhmiourgei tous Headers enos HTTP mhnumatos pou 9a stallei san ena
 * StringList. Xreiazetai ws parametro to mege9os tou mhnumatos, gia
 * to Content-Length header. */
StringList *createHeaders(unsigned int msglen) {
   StringList *headers = newStringList(5);

   headers->strings[0]=strdup("POST * HTTP/1.1\r\n");
   headers->strings[1]=strdup("User-Agent: CWPM Client by Papastefanos
Serafeim\r\n");
   headers->strings[2]=(char *)malloc(80*sizeof(char) );
   sprintf(headers->strings[2],"Content-Length: %d\r\n", msglen );

   headers->strings[3]=strdup("SOAPAction:\r\n");
   headers->strings[4]=strdup("\r\n");
   return headers;
}

/* Stellei ta mhnumata messages ws HTTP mhnuma. Afou parei ta headers apo
thn
 * createHeaders, antigrafei headers kai mhnumata se ena string katallhlou
 * mege9ous, to opoio kai stelletai sth sunexeia mesw tou socket. */
void sendRawMessage(SOCKET sd, StringList *messages) {
   unsigned int msglen=0, totalen=0, pos=0, i;
   for(i=0;i<messages->size;i++) msglen+=strlen(messages->strings[i]);
   StringList *headers = createHeaders(msglen);

   for(i=0;i<headers->size;i++) totalen+=strlen(headers->strings[i]);
   totalen+=msglen+1;

   char *raw = (char *)malloc(totalen*sizeof(char) ) ;

   for(i=0;i<headers->size;i++) {
      strncpy(raw+pos, headers->strings[i], strlen(headers->strings[i]) );
      pos+=strlen(headers->strings[i]);
   }

   for(i=0;i<messages->size;i++) {
      strncpy(raw+pos, messages->strings[i], strlen(messages->strings[i]) );
      pos+=strlen(messages->strings[i]);
   }

   raw[totalen-1]=0;

   send(sd, raw, strlen(raw), 0);

   freeStringList(headers);
```

```
   free(raw);
}


/* Ejetazei an lhf9hke to swsto connection string. Ws proth
 * ekdoxh aplws elegxei an sto mhnuma tou server uparxei to
 * mystring. */
int checkConnectionString(char *mystring, SOCKET sd) {
   char *buf = getLine(sd);
   if(strstr(buf, mystring) ) return 1;
   else return 0;
}

/*************************************************************************
****/


#else  // Unix

#include <sys/socket.h>
#include <netinet/in.h>
#include <arpa/inet.h>

#include "low.h"
#include "helpers.h"

SOCKET makeConnection(char *address, int port) {
   SOCKET sd; /* connecting socket */
   struct sockaddr_in server; /* connecting address */

   if ( ( sd=socket(AF_INET, SOCK_STREAM, 0) ) < 0 ) {
      perror("cannot create socket");
      exit(-1);
   }

   /* set up sockaddr_in */
   server.sin_family=AF_INET;
   server.sin_port=htons(port);
   server.sin_addr.s_addr=inet_addr(address);
   memset(&(server.sin_zero),'\0',8);
   printf("Sending request...\n");
   /* Connect */
   if (connect(sd, (struct sockaddr *)&server, sizeof(server))<0) {
      printf("cannot connect to server");
      exit(-1);
   }
   return sd;
}

unsigned int getNumOfMessages(char *rawMessage) {
   int i=0;
   if(rawMessage == NULL) return 0;
   while(strstr(rawMessage, "<soap:Envelope") ) {
      rawMessage = strstr(rawMessage, "</soap:Envelope>" );
      rawMessage = rawMessage+strlen("</soap:Envelope>");
      i++;
   }
   return i;
}

StringList *splitRawMessage(char *rawMessage) {
   unsigned int i;
   unsigned int num = getNumOfMessages(rawMessage);
   if(num==0) return NULL;
   StringList *messages = newStringList(num);

   for(i=0;i<num;i++) {
```

```
      int l = strstr(rawMessage, "</soap:Envelope>") - rawMessage +
strlen("</soap:Envelope>");
      messages->strings[i]=(char *)malloc( (l+1)*sizeof(char) );
      strncpy(messages->strings[i],rawMessage,l);
      messages->strings[i][l]=0;
      rawMessage = strstr(rawMessage, "</soap:Envelope>") +
strlen("</soap:Envelope>");
   }
   return messages;
}

SOCKET createListeningSocket(int port) {
   SOCKET lsd;
   struct sockaddr_in sin;
   int slen=sizeof(struct sockaddr_in);
   /* Create a listening socket */
   if ( ( lsd=socket(AF_INET, SOCK_STREAM, 0) ) < 0 ) {
      perror("cannot create listening socket");
      exit(-1);
   }

   /* Bind listening socket to port */
   sin.sin_family=AF_INET;
   sin.sin_port=htons(port);
   sin.sin_addr.s_addr=htonl(INADDR_ANY); /* IP d/sh tou susthmatos mas   */
   if (bind(lsd, (struct sockaddr *)&sin, slen ) < 0  ) {
      perror("cannot bind listening socket");
      exit(-1);
   }

   /* Initiate a listen queue */
   if (listen(lsd, 1) < 0  ) {
      perror("cannot listen o socket");
      exit(-1);
   }
   return lsd;
}


SOCKET acceptConnection(SOCKET lsd) {
   SOCKET sd;
   int slen=sizeof(struct sockaddr_in);
   struct sockaddr_in in_sa; /* remote address */
// printf("\nServer ready for requests...\n");
   if( (sd=accept(lsd, (struct sockaddr *)&in_sa,(socklen_t*) &slen)) < 0 ) {
      perror("Cannot accept a connection");
      exit(-1);
   }
   return sd;
}

char *getLine(SOCKET sd) {
   char c;
   char buf[256];
   int i=0;
   while(1) {

      if( recv(sd, &c, 1, 0) < 0 ) {
         perror("recv, CANNOT READ 1 char");
         exit(-1);
      };

      buf[i++]=c;
   // printf("%c", c);

      if(c=='\r') { // GOT CR
         recv(sd, &c, 1, 0); // GOT LF
         buf[i++]=c;
```

```
         break ;
      }
   }
   // Gia na ftasoume edw exei lhf9ei oloklhrh grammh
   char *tmp = (char *)malloc((i+1)*sizeof(char) );
   strncpy(tmp, buf, i);
   tmp[i]='\0';
   return tmp;
}

char *getRawMessage(SOCKET sd) {
   int len;
   char *tmp;
   while(1) {
      tmp=getLine(sd);
      //puts(tmp);

      if(strstr(tmp, "Content-Length:") ) {

         len=atoi(&tmp[15]);
//       if(len==0) return NULL; // PROSOXH EDW ISWS UPARXEI PROVLHMA
      }

      if(tmp[0]==13) { //Adeia grammh
         free(tmp);
         break;
      }
      free(tmp);
   }

   if(len==0) return NULL; // ALLAGH
   char *raw=(char *)malloc((len+1)*sizeof(char ) ) ;
   int totalbytes=0;
   int bytes=0;
   while(totalbytes<len) {
      bytes=recv(sd, raw+totalbytes, len-totalbytes, 0);
      totalbytes+=bytes;
   }
   raw[len]=0;

   return raw;
}

StringList *getMessages(SOCKET sd) {
   StringList *slist;
   char *buf = getRawMessage(sd);
   if(buf == NULL) {
      slist = newStringList(0);
      return slist;
   }
   slist = splitRawMessage(buf);
   free(buf); // Teleutaia allagh !!
   return slist;
}

StringList *createHeaders(unsigned int msglen) {
   StringList *headers = newStringList(5);

   headers->strings[0]=strdup("POST * HTTP/1.1\r\n");
   headers->strings[1]=strdup("User-Agent: CWPM Client by Papastefanos
Serafeim\r\n");
   headers->strings[2]=(char *)malloc(80*sizeof(char) );
   sprintf(headers->strings[2],"Content-Length: %d\r\n", msglen );

   headers->strings[3]=strdup("SOAPAction:\r\n");
   headers->strings[4]=strdup("\r\n");
   return headers;
}
```

```
void sendRawMessage(SOCKET sd, StringList *messages) {
   unsigned int msglen=0, totalen=0, pos=0, i;
   for(i=0;i<messages->size;i++) msglen+=strlen(messages->strings[i]);
   StringList *headers = createHeaders(msglen);

   for(i=0;i<headers->size;i++) totalen+=strlen(headers->strings[i]);
   totalen+=msglen+1;

   char *raw = (char *)malloc(totalen*sizeof(char) ) ;

   for(i=0;i<headers->size;i++) {
      strncpy(raw+pos, headers->strings[i], strlen(headers->strings[i]) );
      pos+=strlen(headers->strings[i]);
   }

   for(i=0;i<messages->size;i++) {
      strncpy(raw+pos, messages->strings[i], strlen(messages->strings[i]) );
      pos+=strlen(messages->strings[i]);
   }

   raw[totalen-1]=0;

   send(sd, raw, strlen(raw), 0);

   freeStringList(headers);
   free(raw);
}

int checkConnectionString(char *mystring, SOCKET sd) {
   char *buf = getLine(sd);
   if(strstr(buf, mystring) ) return 1;
   else return 0;
}


#endif
```

8.6.9 que.h

```
/* Papastefanos Serafeim */
/* Sunarthseis ouras */

#ifndef QUE_H
#define QUE_H

#include <stdio.h>
#include <stdlib.h>

typedef struct _Node {
   char *data;
   struct _Node *next;
} Node;

typedef struct _Queue {
   struct _Node *first;
   struct _Node *last;
   unsigned int size;
} Queue;

void enque(char* i, Queue *q);
char* deque(Queue *q);
void viewQueue(Queue *q);
Queue *createQueue(void);
int isEmpty(Queue *q);
void freeQueue(Queue *q);
```

```
#endif //QUE_H
```

8.6.10 que.cpp

```
/* Papastefanos Serafeim */
/* Ulopoihsh sunarthsewn ouras */

#include "que.h"

/* Pros9etei to mhnuma buf sto telos ths oura */
void enque(char *buf, Queue *q) {
   Node *nod;
   nod=(Node *)malloc(sizeof(Node));
   if(nod == NULL) {
      perror("malloc");
      exit(-1);
   }
   nod->data=buf;
   nod->next=NULL;

   q->size++;

   if(q->first==NULL) {
      q->first=nod;
      q->last=nod;
   } else {
      q->last->next=nod;
      q->last=nod;
   }
}

/* Emfanizei ta periexomena ths ouras */
void viewQueue(Queue *q) {
   Node *p;
   p=q->first;
   if(p==NULL) {
      printf("Adeia oura.");
      return;
   } else {
      printf("\nMege9os ouras = %d. ", q->size);
      while(p->next!=NULL) {
         printf(" %s ",p->data);
         p=p->next;
      }
      printf(" %s ",p->data);
   }
}

/* Epistrefei to epomeno mhnuma apo thn arxh ths ouras */
char *deque(Queue *q) {
   if(q->first == NULL) {
      printf("Adeia oura.");
      return NULL;
   }
   Node *p;
   char *c=q->first->data;
   p=q->first;
   q->first=q->first->next;
   if(q->first == NULL) q->last = NULL;
   free(p);
   q->size--;
   return c;
}

/* Dhmiourgei mia nea, adeia oura */
Queue *createQueue(void) {
```

```
   Queue *tmp = (Queue *)malloc(sizeof(Queue) );
   if(tmp == NULL) return NULL;
   tmp->first = NULL;
   tmp->last = NULL;
   tmp->size = 0;
   return tmp;
}

/* Elegxei an h oura einai adeia */
int isEmpty(Queue *q) {
   return (q->first == NULL);
}

/* Eleu9erwnei th mnhmh pou katalamvanei h oura */
void freeQueue(Queue *q) {
   Node *p = q->first;
   while(p->next != NULL) {
      Node *r = p;
      p=p->next;
      free(r);
   }
   free(q);
}
```

8.6.11 soaparse.h

```
/* Papastefanos Serafeim */
/* Sunarthseis gia parsing soap mhnumatwn */

#include <stdio.h>
#include "ixml.h"
#include "client.h"
#include "helpers.h"

/************
 Oi akolou9es sunarthseis kanoun parse ena SOAP message
 ************/

// Prepei na klh9ei kat' arxhn h parseStart gia na arxisei to parsing tou
// mhnumatos, kai akolou9ws arkei na klh9ei kapoia apo tis epomenes
// pairnontas ws parametro to pedio message tou SoapMessage pou 9a
// epistreyei h parseStart.
SoapMessage *parseStart(char *buf);

// Kanei parse ena mhnuma Inform gemizontas katallhlha ta structs. To node
// prepei na einai to pedio message enos SoapMessage pou exei epistrafei apo
// thn parseStart
int parseInform(IXML_Node *node, DeviceIdStruct *deviceId, EventList
*events, unsigned int *maxEnvelopse, DateTime *currentTime, unsigned int
*retryCount, ParameterValueList *parameterList);

// Paromoiws gia to InformResponse
int parseInformResponse(IXML_Node *node, unsigned int *maxEnvelopes);

// Paromoiws gia to SetParameterValues
int parseSetParameterValues(IXML_Node *node, ParameterValueList
*parameterList, char **parameterKey);

// Paromoiws gia to SetParameterValuesResponse
int parseSetParameterValuesResponse(IXML_Node *node, unsigned int *status);

// Paromoiws gia to GetParameterValues
int parseGetParameterValues(IXML_Node *node, StringList *parameterNames);

// Paromoiws gia to GetParameterValuesResponse
int parseGetParameterValuesResponse(IXML_Node *node, ParameterValueList
*parameterList);
```

```
// Paromoiws gia to GetParameterNames
int parseGetParameterNames(IXML_Node *node, char **pathName, unsigned short
*nextLevel);

// Paromoiws gia to GetParameterNamesResponse
int parseGetParameterNamesResponse(IXML_Node *node, StringList
parameterNames);


// h parseHeader kanei parse ena header element kai epistrefei ws apotelesma
mia domh SoapHeader
SoapHeader *parseHeader(IXML_Node *node);
```

8.6.12 soaparse.cpp

```
/* Papastefanos Serafeim */
/* Sunarthseis gia parsing mhnumatwn soap */

#include <stdio.h>
#include <stdlib.h>
#include "ixml.h"
#include "client.h"
#include "soaparse.h"
#include "helpers.h"

/***********************************/
/********** PARSE STARTING **********/
/***********************************/
SoapMessage *parseStart(char *buf) {
   IXML_Document* doc;

   SoapMessage *tmp = (SoapMessage *)malloc(sizeof(SoapMessage) );
   tmp->header=NULL;

   if( ixmlParseBufferEx(buf, &doc)!=IXML_SUCCESS) {
      printf("Error while parsing the XML buffer, not an xml message.\n");
      return NULL;
   }

   IXML_Node* envelope=ixmlNode_getFirstChild( (IXML_Node*) doc);
   IXML_Node* el=ixmlNode_getFirstChild( (IXML_Node*)envelope);
   if( strcmp( (char *)ixmlNode_getNodeName(el), "soap:Header")==0) {
      tmp->header = parseHeader(el);
      /* go to body */
      el=ixmlNode_getNextSibling( (IXML_Node*)el);
   }

   if( strcmp( (char *)ixmlNode_getNodeName(el), "soap:Body")!=0) {
      printf("Error while parsing the XML Buffer, no soap:Body found.\n");
      if(tmp->header!=NULL) free(tmp->header);
      return NULL;
   }

   el=ixmlNode_getFirstChild( (IXML_Node*)el);

   tmp->message = ixmlNode_cloneNode(el, TRUE);

   if( strcmp( (char *)ixmlNode_getNodeName(el), "cwmp:Inform")==0) {
      printf("\nInform\n");
      tmp->type=INFORM;
      ixmlNode_free((IXML_Node *)doc);
      return tmp;
   } else if( strcmp( (char *)ixmlNode_getNodeName(el),
"cwmp:InformResponse")==0) {
      printf("\nInformResponse\n");
```

```
      tmp->type=INFORMRESPONSE;
      ixmlNode_free((IXML_Node *)doc);
      return tmp;
   } else if( strcmp( (char *)ixmlNode_getNodeName(el),
"cwmp:SetParameterValues")==0) {
      printf("\nSetParameterValues\n");
      tmp->type=SETPARAMETERVALUES;
      ixmlNode_free((IXML_Node *)doc);
      return tmp;
   } else if( strcmp( (char *)ixmlNode_getNodeName(el),
"cwmp:SetParameterValuesResponse")==0) {
      printf("\nSetParameterValuesResponse\n");
      tmp->type=SETPARAMETERVALUESRESPONSE;
      ixmlNode_free((IXML_Node *)doc);
      return tmp;
   } else if( strcmp( (char *)ixmlNode_getNodeName(el),
"cwmp:GetParameterValues")==0) {
      printf("\nGetParameterValues\n");
      tmp->type=GETPARAMETERVALUES;
      ixmlNode_free((IXML_Node *)doc);
      return tmp;
   } else if( strcmp( (char *)ixmlNode_getNodeName(el),
"cwmp:GetParameterValuesResponse")==0) {
      printf("\nGetParameterValuesResponse\n");
      tmp->type=GETPARAMETERVALUESRESPONSE;
      ixmlNode_free((IXML_Node *)doc);
      return tmp;
   } else if( strcmp( (char *)ixmlNode_getNodeName(el),
"cwmp:GetParameterNames")==0) {
      printf("\nGetParameterNames\n");
      tmp->type=GETPARAMETERNAMES;
      ixmlNode_free((IXML_Node *)doc);
      return tmp;
   } else if( strcmp( (char *)ixmlNode_getNodeName(el),
"cwmp:GetParameterNamesResponse")==0) {
      printf("\nGetParameterNamesResponse\n");
      tmp->type=GETPARAMETERNAMESRESPONSE;
      ixmlNode_free((IXML_Node *)doc);
      return tmp;
   } else {
      printf("Not implemented RPC method");
      tmp->type=NOTIMPLEMENTEDMESSAGE;
      ixmlNode_free((IXML_Node *)doc);
      return tmp;
   }
}

/**********************************/
/********** INFORM PARSING **********/
/**********************************/

int parseInform(IXML_Node *node, DeviceIdStruct *deviceId, EventList
*eventList, unsigned int *maxEnvelopes, DateTime *currentTime, unsigned int
*retryCount, ParameterValueList *parameterList) {

   IXML_Node* devid=ixmlNode_getFirstChild(node);
   if( strcmp( (char *)ixmlNode_getNodeName(devid), "DeviceId")!=0) {
      printf("Error while parsing the Inform message (DeviceId).");
      return -1 ; // ERRORRR
   }

   IXML_Node* evt=ixmlNode_getNextSibling(devid);
   if( strcmp( (char *)ixmlNode_getNodeName(evt), "Event")!=0) {
      printf("Error while parsing the Inform message (Event).");
      return -1 ; // ERRORRR
   }

   IXML_Node* maxenvs=ixmlNode_getNextSibling(evt);
```

```
   if( strcmp( (char *)ixmlNode_getNodeName(maxenvs), "MaxEnvelopes")!=0) {
      printf("Error while parsing the Inform message (MaxEnvelopes).");
      return -1 ; // ERRORRR
   }

   IXML_Node* ct=ixmlNode_getNextSibling(maxenvs);
   if( strcmp( (char *)ixmlNode_getNodeName(ct), "CurrentTime")!=0) {
      printf("Error while parsing the Inform message (CurrentTime).");
      return -1 ; // ERRORRR
   }


   IXML_Node* retryc=ixmlNode_getNextSibling(ct);
   if( strcmp( (char *)ixmlNode_getNodeName(retryc), "RetryCount")!=0) {
      printf("Error while parsing the Inform message (RetryCount).");
      return -1 ; // ERRORRR
   }

   IXML_Node* paramlist=ixmlNode_getNextSibling(retryc);
   if( strcmp( (char *)ixmlNode_getNodeName(paramlist), "ParameterList")!=0)
{
      printf("Error while parsing the Inform message (ParameterList).");
      return -1 ; // ERRORRR
   }

   /************** Parsing the DeviceId *****************/
   IXML_Node* manuf=ixmlNode_getFirstChild(devid);
   if( strcmp( (char *)ixmlNode_getNodeName(manuf), "Manufacturer")!=0) {
      printf("Error while parsing the Inform message (DeviceId,
Manufacturer)");
      return -1 ; // ERRORRR
   }
   IXML_Node* manuf_text=ixmlNode_getFirstChild(manuf);
   deviceId->manufactorer=(char
*)ixmlCloneDOMString(ixmlNode_getNodeValue(manuf_text));

   IXML_Node* oui=ixmlNode_getNextSibling(manuf);
   if( strcmp( (char *)ixmlNode_getNodeName(oui), "OUI")!=0) {
      printf("Error while parsing the Inform message (DeviceId, OUI)");
      return -1 ; // ERRORRR
   }
   IXML_Node* oui_text=ixmlNode_getFirstChild(oui);
   deviceId->OUI=(char *)ixmlCloneDOMString(ixmlNode_getNodeValue(oui_text));

   IXML_Node* pclass=ixmlNode_getNextSibling(oui);
      if( strcmp( (char *)ixmlNode_getNodeName(pclass), "ProductClass")!=0) {
      printf("Error while parsing the Inform message (DeviceId,
ProductClass)");
      return -1 ; // ERRORRR
   }
   IXML_Node* pclass_text=ixmlNode_getFirstChild(pclass);
   deviceId->productClass=(char
*)ixmlCloneDOMString(ixmlNode_getNodeValue(pclass_text));

   IXML_Node* snum=ixmlNode_getNextSibling(pclass);
      if( strcmp( (char *)ixmlNode_getNodeName(snum), "SerialNumber")!=0) {
      printf("Error while parsing the Inform message (DeviceId,
SerialNumber)");
      return -1 ; // ERRORRR
   }
   IXML_Node* snum_text=ixmlNode_getFirstChild(snum);
   deviceId->serialNumber=(char
*)ixmlCloneDOMString(ixmlNode_getNodeValue(snum_text));

   /************** Parsing the events ****************/
   IXML_NodeList* event_list=ixmlNode_getChildNodes(evt);
   eventList->size = ixmlNodeList_length(event_list);
// printf("Found %d events.\n", *numOfEvents);
```

```
   unsigned long int i;
   eventList->events = (EventStruct *)malloc(eventList-
>size*sizeof(EventStruct) );
   for(i=0;i<eventList->size;i++) {
      IXML_Node *evnt = ixmlNodeList_item(event_list, i);
      if( strcmp( (char *)ixmlNode_getNodeName(evnt), "EventCode")!=0) {
         printf("Error while parsing the Inform message (Event, EventCode)");
         return -1 ; // ERRORRR
      }
      IXML_Node *evtcode = ixmlNode_getFirstChild(evnt);
      eventList->events[i].eventCode=(char
*)ixmlCloneDOMString(ixmlNode_getNodeValue(evtcode));

   }
   ixmlNodeList_free(event_list);

   /************** Parsing max envelops *****************/
   IXML_Node* maxenvs_txt=ixmlNode_getFirstChild(maxenvs);
   char *buf = (char *)ixmlNode_getNodeValue(maxenvs_txt);
   *maxEnvelopes=atoi(buf);

   /************** Parsing current time*****************/
   IXML_Node* ct_txt=ixmlNode_getFirstChild(ct);
   buf = (char *)ixmlNode_getNodeValue(ct_txt);
   *currentTime = *ascitodate(buf);

   /************** Parsing retry count*****************/
   IXML_Node* retryc_txt=ixmlNode_getFirstChild(retryc);
   buf = (char *)ixmlNode_getNodeValue(retryc_txt);
   *retryCount = atoi(buf);

   /************** Parsing parameter list*****************/
   IXML_NodeList* parval_list=ixmlNode_getChildNodes(paramlist);
   parameterList->size = ixmlNodeList_length(parval_list);
// printf("Found %d params.\n", *numOfParams);
   parameterList->parameters = (ParameterValueStruct *)malloc(parameterList-
>size*sizeof(ParameterValueStruct) );
   for(i=0;i<parameterList->size;i++) {
      IXML_Node *parval = ixmlNodeList_item(parval_list, i);

      IXML_Node *name = ixmlNode_getFirstChild(parval);
      if( strcmp( (char *)ixmlNode_getNodeName(name), "Name")!=0) {
         printf("Error while parsing the Inform message (ParameterValues,
Name)");
         return -1 ; // ERRORRR
      }
      IXML_Node *name_text = ixmlNode_getFirstChild(name);
      parameterList->parameters[i].name=(char
*)ixmlCloneDOMString(ixmlNode_getNodeValue(name_text));
      IXML_Node *value = ixmlNode_getNextSibling(name);
      if( strcmp( (char *)ixmlNode_getNodeName(value), "Value")!=0) {
         printf("Error while parsing the Inform message (ParameterValues,
Value)");
         return -1 ; // ERRORRR
      }
      IXML_Node *value_text = ixmlNode_getFirstChild(value);
      parameterList->parameters[i].value=(char
*)ixmlCloneDOMString(ixmlNode_getNodeValue(value_text));
   }
   ixmlNodeList_free(parval_list);

   ixmlNode_free(node);
   return 0;
}

int parseInformResponse(IXML_Node *node, unsigned int *maxEnvelopes) {
   /* Going to <MaxEnvelopes> element */
   IXML_Node* maxenvs=ixmlNode_getFirstChild(node);
```

```
   if( strcmp( (char *)ixmlNode_getNodeName(maxenvs), "MaxEnvelopes")!=0) {
        printf("Error while parsing the InformResponse message.");
        return -INVALID_ARGUMENTS ; // Error, invalid arguments
   }
   /* Going to the text child */
   IXML_Node* maxenvs_txt=ixmlNode_getFirstChild(maxenvs);
   char *buf = (char *)ixmlNode_getNodeValue(maxenvs_txt);
   *maxEnvelopes=atoi(buf);
   if(*maxEnvelopes < 1) {
      printf("Error while parsing the InformResponse message.");
      // Error, maxEnvelopes must be > 0
      return -INVALID_ARGUMENTS;
   }
   ixmlNode_free(node);
   return 0;
}

/*********************************************/
/********** SETPARAMETERVALUES PARSING **********/
/*********************************************/

int parseSetParameterValues(IXML_Node *node, ParameterValueList
*parameterList, char **parameterKey) {
   /* Going to <ParameterList> element */
   IXML_Node* paramlist=ixmlNode_getFirstChild(node);
   if( strcmp( (char *)ixmlNode_getNodeName(paramlist), "ParameterList")!=0)
{
        printf("Error while parsing the SetParameterValues message
(ParameterList)");
        return -INVALID_ARGUMENTS ;
   }

   /* Getting the <ParameterValueStruct> elements */
   IXML_NodeList* parval_list=ixmlNode_getChildNodes(paramlist);

   parameterList->size = ixmlNodeList_length(parval_list);
// printf("Found %d params.\n", *numOfParams);

   unsigned long int i;

   parameterList->parameters = (ParameterValueStruct *)malloc(parameterList-
>size*sizeof(ParameterValueStruct) );
   for(i=0;i<parameterList->size;i++) {
      IXML_Node *parval = ixmlNodeList_item(parval_list, i);

      IXML_Node *name = ixmlNode_getFirstChild(parval);
      if( name == NULL || strcmp( (char *)ixmlNode_getNodeName(name),
"Name")!=0) {
        printf("Error while parsing the SetParameterValues message
(ParameterValueStruct, Name)");
        return -INVALID_ARGUMENTS ;
      }
      IXML_Node *name_text = ixmlNode_getFirstChild(name);
      parameterList->parameters[i].name=(char
*)ixmlCloneDOMString(ixmlNode_getNodeValue(name_text));
      IXML_Node *value = ixmlNode_getNextSibling(name);
      if( strcmp( (char *)ixmlNode_getNodeName(value), "Value")!=0) {
        printf("Error while parsing the SetParameterValues message
(ParameterValueStruct, Value)");
        return -INVALID_ARGUMENTS ;
      }
      IXML_Node *value_text = ixmlNode_getFirstChild(value);
      parameterList->parameters[i].value=(char
*)ixmlCloneDOMString(ixmlNode_getNodeValue(value_text));
   }
   ixmlNodeList_free(parval_list);

   IXML_Node* pkey=ixmlNode_getNextSibling(paramlist);
```

```
   IXML_Node *pkey_text = ixmlNode_getFirstChild(pkey);
   *parameterKey=(char
*)ixmlCloneDOMString(ixmlNode_getNodeValue(pkey_text));
   if(*parameterKey == NULL) *parameterKey = "";

   ixmlNode_free(node);
   return 0;
}

int parseSetParameterValuesResponse(IXML_Node *node, unsigned int *status) {
   /* Going to <Status> element */
   IXML_Node* stat=ixmlNode_getFirstChild(node);
   if( strcmp( (char *)ixmlNode_getNodeName(stat), "Status")!=0) {
      printf("Error while parsing the SetParameterValuesResponse message
(Status)");
      return -INVALID_ARGUMENTS ; // ERRORRR
   }
   /* Going to the text child */
   IXML_Node* stat_txt=ixmlNode_getFirstChild(stat);
   char *buf = (char *)ixmlNode_getNodeValue(stat_txt);
   *status=atoi(buf);
   ixmlNode_free(node);
   return 0;
}

/***********************************************/
/********** GETPARAMETERVALUES PARSING **********/
/***********************************************/

int parseGetParameterValues(IXML_Node *node, StringList *parameterNames) {
   IXML_Node* names=ixmlNode_getFirstChild(node);
   if( strcmp( (char *)ixmlNode_getNodeName(names), "ParameterNames")!=0) {
      printf("Error while parsing the GetParameterValues message
(ParameterNames)");
      return -1 ; // ERRORRR
   }

   IXML_NodeList* name_list=ixmlNode_getChildNodes(names);
   //puts( (char *)ixmlNode_getNodeName(names) );
   parameterNames->size = ixmlNodeList_length(name_list);
   parameterNames->strings = (char **)malloc(parameterNames->size*sizeof(char
*) );
   unsigned int i;
   for(i=0;i<parameterNames->size;i++) {
      IXML_Node *name = ixmlNodeList_item(name_list, i);

      IXML_Node *name_text = ixmlNode_getFirstChild(name);
   // puts( (char *)ixmlNode_getNodeName(name) );
      parameterNames->strings[i] =(char
*)ixmlCloneDOMString(ixmlNode_getNodeValue(name_text));

   }
   ixmlNodeList_free(name_list);
   ixmlNode_free(node);
   return 0;
}

int parseGetParameterValuesResponse(IXML_Node *node, ParameterValueList
*parameterList) {
   /* Going to <ParameterList> element */
   IXML_Node* paramlist=ixmlNode_getFirstChild(node);
   if( strcmp( (char *)ixmlNode_getNodeName(paramlist), "ParameterList")!=0)
{
      printf("Error while parsing the GetParameterValuesResponse message
(ParameterList)");
      return -1 ; // ERRORRR
   }
```

```
   /* Getting the <ParameterValueStruct> elements */
   IXML_NodeList* parval_list=ixmlNode_getChildNodes(paramlist);

   parameterList->size = ixmlNodeList_length(parval_list);
// printf("Found %d params.\n", *numOfParams);

   unsigned long int i;

   parameterList->parameters = (ParameterValueStruct *)malloc(parameterList-
>size*sizeof(ParameterValueStruct) );
   for(i=0;i<parameterList->size;i++) {
      IXML_Node *parval = ixmlNodeList_item(parval_list, i);

      IXML_Node *name = ixmlNode_getFirstChild(parval);
      if( strcmp( (char *)ixmlNode_getNodeName(name), "Name")!=0) {
         printf("Error while parsing the GetParameterValuesResponse message
(ParameterList, ParameterValueStruct, Name)");
         return -1 ; // ERRORRR
      }
      IXML_Node *name_text = ixmlNode_getFirstChild(name);
      parameterList->parameters[i].name=(char
*)ixmlCloneDOMString(ixmlNode_getNodeValue(name_text));
      IXML_Node *value = ixmlNode_getNextSibling(name);
      if( strcmp( (char *)ixmlNode_getNodeName(parval), "Value")!=0) {
         printf("Error while parsing the GetParameterValuesResponse message
(ParameterList, ParameterValueStruct, Vali=ue)");
         return -1 ; // ERRORRR
      }
      IXML_Node *value_text = ixmlNode_getFirstChild(value);
      parameterList->parameters[i].value=(char
*)ixmlCloneDOMString(ixmlNode_getNodeValue(value_text));
   }
   ixmlNodeList_free(parval_list);
   ixmlNode_free(node);
   return 0;

}


/*********************************************/
/********** GETPARAMETERNAMES PARSING **********/
/*********************************************/

int parseGetParameterNames(IXML_Node *node, char **pathName, unsigned short
*nextLevel) {
   IXML_Node* path=ixmlNode_getFirstChild(node);
   if( strcmp( (char *)ixmlNode_getNodeName(path), "ParameterPath")!=0) {
      printf("Error while parsing the GetParameterNames message
(ParameterPath)");
      return -1 ; // ERRORRR
   }

   IXML_Node *path_text = ixmlNode_getFirstChild(path);
   *pathName=(char *)ixmlCloneDOMString(ixmlNode_getNodeValue(path_text));
   if(*pathName == NULL) *pathName = "";

   IXML_Node* nlevel=ixmlNode_getNextSibling(path);
   if( strcmp( (char *)ixmlNode_getNodeName(nlevel), "NextLevel")!=0) {
      printf("Error while parsing the GetParameterNames message
(NextLevel)");
      return -1 ; // ERRORRR
   }

   IXML_Node *nlevel_text = ixmlNode_getFirstChild(nlevel);
   *nextLevel=(strcmp(ixmlNode_getNodeValue(nlevel_text),"false")==0)?0:1;


   ixmlNode_free(node);
```

```
   return 0;
}

int parseGetParameterNamesResponse(IXML_Node *node, StringList
parameterNames) {
// DE XREIAZETAI STON CLIENT
   return 0;

}

SoapHeader *parseHeader(IXML_Node *node) {
   SoapHeader *tmp = (SoapHeader *)malloc(sizeof(SoapHeader) );
   tmp->id = NULL;
   tmp->holdRequests=0;
   tmp->noMoreRequests=0;
   unsigned int i;
   IXML_NodeList* head_list=ixmlNode_getChildNodes(node);

   unsigned int len = ixmlNodeList_length(head_list);

   for(i=0;i<len;i++) {
      IXML_Node *el = ixmlNodeList_item(head_list, i);
      char *name = (char *)ixmlNode_getNodeName(el);
      IXML_Node *el_text = ixmlNode_getFirstChild(el);
      if(strcmp(name,"cwmp:ID")==0) tmp->id = (char
*)ixmlCloneDOMString(ixmlNode_getNodeValue(el_text));
      if(strcmp(name,"cwmp:HoldRequests")==0) tmp->holdRequests = atoi( (char
*)ixmlNode_getNodeValue(el_text) );
      if(strcmp(name,"cwmp:NoMoreRequests")==0) tmp->noMoreRequests = atoi(
(char *)ixmlNode_getNodeValue(el_text) );
   }
   ixmlNodeList_free(head_list);
   return tmp;
}
```

8.6.13 soapcrt.h

```
/* Papastefanos Serafeim */
/* Sunarthseis gia dhmiouria soap mhnumatwn */

#include <stdio.h>
#include "ixml.h"
#include "client.h"
#include "helpers.h"

/****************
 Oi akolou9es sunarthseis dhmiourgoun ta soap envelopes gia ka9e RPC
 ***************/

// Dhmiourgei to soap message gia ena mhnuma Inform. Oi parametroi einai oi
// antistoixes domes se C me tis parametrous tou TR69, ka9ws kai ena header.
IXML_Document *createInform(SoapHeader *sh, DeviceIdStruct deviceId,
EventList *events, unsigned int maxEnvelopse, DateTime currentTime, unsigned
int retryCount, ParameterValueList *parameterList);

// Dhmiourgei to soap message gia ena mhnuma InformResponse. Oi parametroi
// einai to header ka9ws kai ta maximum envelopes tou server.
IXML_Document *createInformResponse(SoapHeader *sh, unsigned int
maxEnvelopes);

// Dhmiourgei to createSetParameterValues mhnuma
IXML_Document *createSetParameterValues(SoapHeader *sh, ParameterValueList
*parameterList, char *parameterKey);
// Dhmiourgei to createSetParameterValuesResponse mhnuma
IXML_Document *createSetParameterValuesResponse(SoapHeader *sh, int status);

// Dhmiourgei to createGetParameterValues mhnuma
```

```
IXML_Document *createGetParameterValues(SoapHeader *sh, StringList
*parameterNames);
// Dhmiourgei to createGetParameterValuesResponse mhnuma
IXML_Document *createGetParameterValuesResponse(SoapHeader *sh,
ParameterValueList *parameterList);

// Dhmiourgei to createGetParameterNames mhnuma
IXML_Document *createGetParameterNames(SoapHeader *sh, char *parameterPath,
unsigned short int nextLevel);
// Dhmiourgei to createGetParameterNamesResponse mhnuma
IXML_Document *createGetParameterNamesResponse(SoapHeader *sh, char *path,
unsigned short int nextLevel);


// Dhmiourgei ena Fault response
IXML_Document *createFaultResponse(SoapHeader *sh, int code);


// Dhmiourgei ena Fault response gia thn SetParameterNames
IXML_Document *createSetFaultResponse(SoapHeader *sh, ParameterValueList
*params, int * errors);

/*******************
 Oi akolou9es sunarthseis einai voh9htikes kai xrhsimopoiountai apo tis
 parapanw sunarthseis. De 9a prepei na klh9oun anejarthta.
 ***************/
// Metafrazei mia domh DeviceStruct se soap
IXML_Element *createDeviceId(IXML_Document* doc, DeviceIdStruct *deviceId);

// Metafrazei mia domh EventStruct se soap
IXML_Element *createEvent(IXML_Document* doc, EventList *events);

// Metafrazei enan akeraio se maxenvelopes element
IXML_Element *createMaxEnvelope(IXML_Document* doc, unsigned int
maxEnvelope);

// Metafrazei thn hmeromhnia se soap
IXML_Element *createDateTime(IXML_Document* doc, DateTime *currentTime);

// Dhmiourgei to retry count element
IXML_Element *createRetryCount(IXML_Document* doc, unsigned int retryCount);

// Metafrazei mia domh ParameterValueStruct se soap
IXML_Element *createParameterList(IXML_Document* doc, ParameterValueList
*parameterListm);

// Dhmiourgei to status element
IXML_Element *createStatus(IXML_Document* doc, unsigned int status);

// Metafrazei mia lista apo sumvoloakolou9es sto ParameterNames element
IXML_Element *createParameterNames(IXML_Document *doc, StringList
*parameterNames);

// Dhmiourgei to header se soap me vash to sh
IXML_Node *createHeader(IXML_Document* doc, SoapHeader *sh);
```

8.6.14 soapcrt.cpp

```
/* Papastefanos Serafeim */
/* Sunarthseis gia dhmiourgia soap mhnumatwn */

#include <stdio.h>
#include "ixml.h"
#include "client.h"
#include "soapcrt.h"
#include "helpers.h"
#include "custom.h"
```

```
/************************************/
/********** INFORM CREATION **********/
/************************************/
IXML_Document *createInform(SoapHeader *sh, DeviceIdStruct deviceId,
EventList *events, unsigned int maxEnvelope, DateTime currentTime, unsigned
int retryCount, ParameterValueList *parameterList) {
   IXML_Document *doc = ixmlDocument_createDocument();

   IXML_Element *envelope= ixmlDocument_createElementNS(doc,
"http://schemas.xmlsoap.org/soap/envelope/", "soap:Envelope");
   ixmlElement_setAttribute(envelope,"xmlns:soap","http://schemas.xmlsoap.org
/soap/envelope/" );
   ixmlElement_setAttribute(envelope,"soap:encodingStyle","http://schemas.xml
soap.org/soap/encoding/" );
   ixmlElement_setAttribute(envelope,"xmlns:xsi",
"http://www.w3.org/2001/XMLSchema/instance/" );
   ixmlElement_setAttribute(envelope,"xmlns:xsd",
"http://www.w3.org/2001/XMLSchema/" );
   ixmlElement_setAttribute(envelope,"xmlns:cwmp", "urn:dslforum-org:cwmp-1-
0" );


   ixmlNode_appendChild( (IXML_Node *)doc,(IXML_Node *) envelope);

   if(sh!=NULL) {
      IXML_Node *header = createHeader(doc, sh);
      ixmlNode_appendChild( (IXML_Node *)envelope,(IXML_Node *)header);
   }

   IXML_Element *body= ixmlDocument_createElementNS(doc,
"http://schemas.xmlsoap.org/soap/envelope/", "soap:Body");
   ixmlNode_appendChild( (IXML_Node *)envelope,(IXML_Node *)body);

   IXML_Element *inform=ixmlDocument_createElementNS(doc,
"xmlns:cwmp=urn:dslforum-org:cwmp-1-0", "cwmp:Inform");
   ixmlNode_appendChild( (IXML_Node *)body,(IXML_Node *)inform);

   IXML_Element *devid = createDeviceId(doc, &deviceId);
   IXML_Element *evt = createEvent(doc, events);
   IXML_Element *maxenv=createMaxEnvelope(doc, maxEnvelope);
   IXML_Element *dt=createDateTime(doc, &currentTime);
   IXML_Element *retrcou=createRetryCount(doc, retryCount);
   IXML_Element *parlist=createParameterList(doc, parameterList);


   ixmlNode_appendChild( (IXML_Node *)inform,(IXML_Node *)devid);
   ixmlNode_appendChild( (IXML_Node *)inform,(IXML_Node *)evt);
   ixmlNode_appendChild( (IXML_Node *)inform,(IXML_Node *)maxenv);
   ixmlNode_appendChild( (IXML_Node *)inform,(IXML_Node *)dt);
   ixmlNode_appendChild( (IXML_Node *)inform,(IXML_Node *)retrcou);
   ixmlNode_appendChild( (IXML_Node *)inform,(IXML_Node *)parlist);

   return doc;

}

IXML_Document *createInformResponse(SoapHeader *sh, unsigned int
maxEnvelope) {
   IXML_Document *doc = ixmlDocument_createDocument();

   IXML_Element *envelope= ixmlDocument_createElementNS(doc,
"http://schemas.xmlsoap.org/soap/envelope/", "soap:Envelope");
   ixmlElement_setAttribute(envelope,"xmlns:soap","http://schemas.xmlsoap.org
/soap/envelope/" );
   ixmlElement_setAttribute(envelope,"soap:encodingStyle","http://schemas.xml
soap.org/soap/encoding/" );
```

```
   ixmlElement_setAttribute(envelope,"xmlns:cwmp", "urn:dslforum-org:cwmp-1-
0" );
   ixmlNode_appendChild( (IXML_Node *)doc,(IXML_Node *) envelope);

   if(sh!=NULL) {
      IXML_Node *header = createHeader(doc, sh);
      ixmlNode_appendChild( (IXML_Node *)envelope,(IXML_Node *)header);
   }

   IXML_Element *body= ixmlDocument_createElementNS(doc,
"http://schemas.xmlsoap.org/soap/envelope/", "soap:Body");
   ixmlNode_appendChild( (IXML_Node *)envelope,(IXML_Node *)body);

   IXML_Element *informresp=ixmlDocument_createElementNS(doc,
"xmlns:cwmp=urn:dslforum-org:cwmp-1-0", "cwmp:InformResponse");
   ixmlNode_appendChild( (IXML_Node *)body,(IXML_Node *)informresp);
   IXML_Element *maxenv=createMaxEnvelope(doc, maxEnvelope);
   ixmlNode_appendChild( (IXML_Node *)informresp,(IXML_Node *)maxenv);
   return doc;
}

IXML_Element *createDeviceId(IXML_Document* doc, DeviceIdStruct *deviceId) {
   IXML_Element *el=ixmlDocument_createElementNS(doc,
"xmlns:cwmp=urn:dslforum-org:cwmp-1-0", "DeviceId");
   ixmlElement_setAttribute(el,"xsi:type","cwmp:DeviceIdStruct" );

   IXML_Element *manuf=ixmlDocument_createElementNS(doc,
"xmlns:cwmp=urn:dslforum-org:cwmp-1-0", "Manufacturer");
   IXML_Node* manuf_text=ixmlDocument_createTextNode(doc, deviceId-
>manufactorer);
   ixmlElement_setAttribute(manuf,"xsi:type","xsd:string[64]" );
   ixmlNode_appendChild( (IXML_Node *) manuf, manuf_text);

   IXML_Element *oui=ixmlDocument_createElementNS(doc,
"xmlns:cwmp=urn:dslforum-org:cwmp-1-0", "OUI");
   IXML_Node* oui_text=ixmlDocument_createTextNode(doc, deviceId->OUI);
   ixmlElement_setAttribute(oui,"xsi:type","xsd:string[6]" );
   ixmlNode_appendChild( (IXML_Node *) oui, oui_text);

   IXML_Element *prclass=ixmlDocument_createElementNS(doc,
"xmlns:cwmp=urn:dslforum-org:cwmp-1-0", "ProductClass");
   IXML_Node* prclass_text=ixmlDocument_createTextNode(doc, deviceId-
>productClass);
   ixmlElement_setAttribute(prclass,"xsi:type","xsd:string[64]" );
   ixmlNode_appendChild( (IXML_Node *) prclass, prclass_text);


   IXML_Element *sn=ixmlDocument_createElementNS(doc,
"xmlns:cwmp=urn:dslforum-org:cwmp-1-0", "SerialNumber");
   IXML_Node* sn_text=ixmlDocument_createTextNode(doc, deviceId-
>serialNumber);
   ixmlElement_setAttribute(sn,"xsi:type","xsd:string[64]" );
   ixmlNode_appendChild( (IXML_Node *) sn, sn_text);

   ixmlNode_appendChild( (IXML_Node *)el, (IXML_Node *)manuf );
   ixmlNode_appendChild( (IXML_Node *)el, (IXML_Node *)oui );
   ixmlNode_appendChild( (IXML_Node *)el, (IXML_Node *)prclass );
   ixmlNode_appendChild( (IXML_Node *)el, (IXML_Node *)sn );

   return el;
}

IXML_Element *createEvent(IXML_Document* doc, EventList *events) {

   IXML_Element *el=ixmlDocument_createElementNS(doc,
"xmlns:cwmp=urn:dslforum-org:cwmp-1-0", "Event");
   char a[40];
   sprintf(a, "cwmp:EventStruct[%d]", events->size );
```

```
   ixmlElement_setAttribute(el,"soap:arrayType",a );

   unsigned int i;

   for(i=0;i<events->size;i++) {
      IXML_Element *el1=ixmlDocument_createElementNS(doc,
"xmlns:cwmp=urn:dslforum-org:cwmp-1-0", "EventCode");
      IXML_Node* el1_text=ixmlDocument_createTextNode(doc, events-
>events[i].eventCode );
      ixmlElement_setAttribute(el1,"xsi:type","xsd:string[64]" );
      ixmlNode_appendChild( (IXML_Node *) el1, (IXML_Node *) el1_text);

      ixmlNode_appendChild( (IXML_Node *) el, (IXML_Node *) el1);
   }

   return el;

}

IXML_Element *createMaxEnvelope(IXML_Document* doc, unsigned int
maxEnvelope) {
   IXML_Element *el=ixmlDocument_createElementNS(doc,
"xmlns:cwmp=urn:dslforum-org:cwmp-1-0", "MaxEnvelopes");
   ixmlElement_setAttribute(el,"xsi:type","xsd:unsignedInt" );
   char *buf = (char *)malloc(4*sizeof(char) ) ;
   sprintf(buf, "%d", maxEnvelope);
   IXML_Node* el_text=ixmlDocument_createTextNode(doc, buf);
   ixmlNode_appendChild( (IXML_Node *) el, el_text);
   return el;
}

IXML_Element *createDateTime(IXML_Document* doc, DateTime *currentTime) {
   IXML_Element *el=ixmlDocument_createElementNS(doc,
"xmlns:cwmp=urn:dslforum-org:cwmp-1-0", "CurrentTime");
   char *dt = datetoasci(currentTime);
   ixmlElement_setAttribute(el,"xsi:type","xsd:dateTime" );
   IXML_Node* el_text=ixmlDocument_createTextNode(doc, dt);
   ixmlNode_appendChild( (IXML_Node *) el, el_text);
// free(dt);
   return el;
}

IXML_Element *createRetryCount(IXML_Document* doc, unsigned int retryCount)
{
   IXML_Element *el=ixmlDocument_createElementNS(doc,
"xmlns:cwmp=urn:dslforum-org:cwmp-1-0", "RetryCount");
   ixmlElement_setAttribute(el,"xsi:type","xsd:unsignedInt" );
   char *buf = (char *)malloc(4*sizeof(char) ) ;
   sprintf(buf, "%d", retryCount);
   IXML_Node* el_text=ixmlDocument_createTextNode(doc, buf );
   ixmlNode_appendChild( (IXML_Node *) el, el_text);
   return el;
}

IXML_Element *createParameterList(IXML_Document* doc, ParameterValueList
*parameterList) {
   IXML_Element *el=ixmlDocument_createElementNS(doc,
"xmlns:cwmp=urn:dslforum-org:cwmp-1-0", "ParameterList");
   char a[50];
   sprintf(a, "cwmp:ParameterValueStruct[%d]", parameterList->size);
   ixmlElement_setAttribute(el,"soap:arrayType",a );

   unsigned int i;

   for(i=0;i<parameterList->size;i++) {
      IXML_Element *el1=ixmlDocument_createElementNS(doc,
"xmlns:cwmp=urn:dslforum-org:cwmp-1-0", "ParameterValueStruct");
```

```
      IXML_Element *name=ixmlDocument_createElementNS(doc,
"xmlns:cwmp=urn:dslforum-org:cwmp-1-0", "Name");
      IXML_Node* name_text=ixmlDocument_createTextNode(doc, parameterList-
>parameters[i].name );
      ixmlElement_setAttribute(name,"xsi:type","xsd:string" );
      ixmlNode_appendChild( (IXML_Node *) name, (IXML_Node *) name_text);

      IXML_Element *value=ixmlDocument_createElementNS(doc,
"xmlns:cwmp=urn:dslforum-org:cwmp-1-0", "Value");
      IXML_Node* value_text=ixmlDocument_createTextNode(doc, parameterList-
>parameters[i].value );
      ixmlElement_setAttribute(value,"xsi:type","xsd:string" );
      ixmlNode_appendChild( (IXML_Node *) value, (IXML_Node *) value_text);

      ixmlNode_appendChild( (IXML_Node *) el1, (IXML_Node *) name);
      ixmlNode_appendChild( (IXML_Node *) el1, (IXML_Node *) value);
      ixmlNode_appendChild( (IXML_Node *) el, (IXML_Node *) el1);
   }

   return el;

}



/************************************************/
/********** GETPARAMETER VALUES CREATION **********/
/************************************************/

IXML_Document *createGetParameterValues(SoapHeader *sh, StringList
*parameterNames) {
   IXML_Document *doc = ixmlDocument_createDocument();

   IXML_Element *envelope= ixmlDocument_createElementNS(doc,
"http://schemas.xmlsoap.org/soap/envelope/", "soap:Envelope");
   ixmlElement_setAttribute(envelope,"xmlns:soap","http://schemas.xmlsoap.org
/soap/envelope/" );
   ixmlElement_setAttribute(envelope,"soap:encodingStyle","http://schemas.xml
soap.org/soap/encoding/" );
   ixmlElement_setAttribute(envelope,"xmlns:cwmp", "urn:dslforum-org:cwmp-1-
0" );
   ixmlNode_appendChild( (IXML_Node *)doc,(IXML_Node *) envelope);

   if(sh!=NULL) {
      IXML_Node *header = createHeader(doc, sh);
      ixmlNode_appendChild( (IXML_Node *)envelope,(IXML_Node *)header);
   }


   IXML_Element *body=ixmlDocument_createElementNS(doc,
"http://schemas.xmlsoap.org/soap/envelope/", "soap:Body");
   ixmlNode_appendChild( (IXML_Node *)envelope,(IXML_Node *)body);


   IXML_Element *inform=ixmlDocument_createElementNS(doc,
"xmlns:cwmp=urn:dslforum-org:cwmp-1-0", "cwmp:GetParameterValues");
   ixmlNode_appendChild( (IXML_Node *)body,(IXML_Node *)inform);

   IXML_Element *parlist=createParameterNames(doc, parameterNames);

   ixmlNode_appendChild( (IXML_Node *)inform,(IXML_Node *)parlist);

   return doc;

}

IXML_Document *createGetParameterValuesResponse(SoapHeader *sh,
ParameterValueList *parameterList) {
```

```
   IXML_Document *doc = ixmlDocument_createDocument();

   IXML_Element *envelope= ixmlDocument_createElementNS(doc,
"http://schemas.xmlsoap.org/soap/envelope/", "soap:Envelope");
   ixmlElement_setAttribute(envelope,"xmlns:soap","http://schemas.xmlsoap.org
/soap/envelope/" );
   ixmlElement_setAttribute(envelope,"soap:encodingStyle","http://schemas.xml
soap.org/soap/encoding/" );
   ixmlElement_setAttribute(envelope,"xmlns:xsi",
"http://www.w3.org/2001/XMLSchema/instance/" );
   ixmlElement_setAttribute(envelope,"xmlns:xsd",
"http://www.w3.org/2001/XMLSchema/" );
   ixmlElement_setAttribute(envelope,"xmlns:cwmp", "urn:dslforum-org:cwmp-1-
0" );
   ixmlNode_appendChild( (IXML_Node *)doc,(IXML_Node *) envelope);

   if(sh!=NULL) {
      IXML_Node *header = createHeader(doc, sh);
      ixmlNode_appendChild( (IXML_Node *)envelope,(IXML_Node *)header);
   }

   IXML_Element *body=ixmlDocument_createElementNS(doc,
"http://schemas.xmlsoap.org/soap/envelope/", "soap:Body");
   ixmlNode_appendChild( (IXML_Node *)envelope,(IXML_Node *)body);


   IXML_Element *inform=ixmlDocument_createElementNS(doc,
"xmlns:cwmp=urn:dslforum-org:cwmp-1-0", "cwmp:GetParameterValuesResponse");
   ixmlNode_appendChild( (IXML_Node *)body,(IXML_Node *)inform);

   IXML_Element *parlist=createParameterList(doc, parameterList);

   ixmlNode_appendChild( (IXML_Node *)inform,(IXML_Node *)parlist);

   return doc;

}

IXML_Element *createParameterNames(IXML_Document *doc, StringList
*parameterNames) {
   IXML_Element *el=ixmlDocument_createElementNS(doc,
"xmlns:cwmp=urn:dslforum-org:cwmp-1-0", "ParameterNames");
   char a[50];
   sprintf(a, "cwmp:ParameterValueStruct[%d]", parameterNames->size);
   ixmlElement_setAttribute(el,"soap:arrayType",a );

   unsigned int i;

   for(i=0;i<parameterNames->size;i++) {
      IXML_Element *el1=ixmlDocument_createElementNS(doc,
"xmlns:cwmp=urn:dslforum-org:cwmp-1-0", "ParameterName");

      IXML_Node* el1_text=ixmlDocument_createTextNode(doc, parameterNames-
>strings[i] );
      ixmlElement_setAttribute(el1,"xsi:type","xsd:string" );
      ixmlNode_appendChild( (IXML_Node *) el1, (IXML_Node *) el1_text);

      ixmlNode_appendChild( (IXML_Node *) el, (IXML_Node *) el1);
   }
   return el;
}



/*********************************************/
/********** SETPARAMETERVALUES CREATION **********/
/*********************************************/
```

```
IXML_Document *createSetParameterValues(SoapHeader *sh, ParameterValueList
*parameterList, char *parameterKey) {
   IXML_Document *doc = ixmlDocument_createDocument();

   IXML_Element *envelope= ixmlDocument_createElementNS(doc,
"http://schemas.xmlsoap.org/soap/envelope/", "soap:Envelope");
   ixmlElement_setAttribute(envelope,"xmlns:soap","http://schemas.xmlsoap.org
/soap/envelope/" );
   ixmlElement_setAttribute(envelope,"soap:encodingStyle","http://schemas.xml
soap.org/soap/encoding/" );
   ixmlElement_setAttribute(envelope,"xmlns:xsi",
"http://www.w3.org/2001/XMLSchema/instance/" );
   ixmlElement_setAttribute(envelope,"xmlns:xsd",
"http://www.w3.org/2001/XMLSchema/" );
   ixmlElement_setAttribute(envelope,"xmlns:cwmp", "urn:dslforum-org:cwmp-1-
0" );
   ixmlNode_appendChild( (IXML_Node *)doc,(IXML_Node *) envelope);

   if(sh!=NULL) {
      IXML_Node *header = createHeader(doc, sh);
      ixmlNode_appendChild( (IXML_Node *)envelope,(IXML_Node *)header);
   }

   IXML_Element *body=ixmlDocument_createElementNS(doc,
"http://schemas.xmlsoap.org/soap/envelope/", "soap:Body");
   ixmlNode_appendChild( (IXML_Node *)envelope,(IXML_Node *)body);


   IXML_Element *setparval=ixmlDocument_createElementNS(doc,
"xmlns:cwmp=urn:dslforum-org:cwmp-1-0", "cwmp:SetParameterValues");
   ixmlNode_appendChild( (IXML_Node *)body,(IXML_Node *)setparval);

   IXML_Element *parlist=createParameterList(doc, parameterList);

   ixmlNode_appendChild( (IXML_Node *)setparval,(IXML_Node *)parlist);


   IXML_Element *parkey = ixmlDocument_createElementNS(doc,
"xmlns:cwmp=urn:dslforum-org:cwmp-1-0", "ParameterKey");
   IXML_Node *parkey_text =ixmlDocument_createTextNode(doc, parameterKey);
   ixmlNode_appendChild( (IXML_Node *)parkey,(IXML_Node *)parkey_text);
   ixmlNode_appendChild( (IXML_Node *)setparval,(IXML_Node *)parkey);

   return doc;

}


IXML_Document *createSetParameterValuesResponse(SoapHeader *sh, int status)
{
   IXML_Document *doc = ixmlDocument_createDocument();

   IXML_Element *envelope= ixmlDocument_createElementNS(doc,
"http://schemas.xmlsoap.org/soap/envelope/", "soap:Envelope");
   ixmlElement_setAttribute(envelope,"xmlns:soap","http://schemas.xmlsoap.org
/soap/envelope/" );
   ixmlElement_setAttribute(envelope,"soap:encodingStyle","http://schemas.xml
soap.org/soap/encoding/" );
   ixmlElement_setAttribute(envelope,"xmlns:cwmp", "urn:dslforum-org:cwmp-1-
0" );
   ixmlNode_appendChild( (IXML_Node *)doc,(IXML_Node *) envelope);

   if(sh!=NULL) {
      IXML_Node *header = createHeader(doc, sh);
      ixmlNode_appendChild( (IXML_Node *)envelope,(IXML_Node *)header);
   }
```

```
   IXML_Element *body=ixmlDocument_createElementNS(doc,
"http://schemas.xmlsoap.org/soap/envelope/", "soap:Body");
   ixmlNode_appendChild( (IXML_Node *)envelope,(IXML_Node *)body);

   IXML_Element *inform=ixmlDocument_createElementNS(doc,
"xmlns:cwmp=urn:dslforum-org:cwmp-1-0", "cwmp:SetParameterValuesResponse");
   ixmlNode_appendChild( (IXML_Node *)body,(IXML_Node *)inform);

   IXML_Element *stat=createStatus(doc, status);

   ixmlNode_appendChild( (IXML_Node *)inform,(IXML_Node *)stat);

   return doc;
}

IXML_Element *createStatus(IXML_Document* doc, unsigned int status) {
   IXML_Element *stat=ixmlDocument_createElementNS(doc,
"xmlns:cwmp=urn:dslforum-org:cwmp-1-0", "Status");
   char *buf = (char *)malloc(4*sizeof(char) ) ;
   sprintf(buf, "%d", status);
   IXML_Node* stat_text=ixmlDocument_createTextNode(doc, buf);
   ixmlNode_appendChild( (IXML_Node *) stat, stat_text);
   return stat;
}


/***********************************************/
/********** GETPARAMETERNAMES CREATION **********/
/***********************************************/

IXML_Document *createGetParameterNames(SoapHeader *sh, char *parameterPath,
unsigned short int nextLevel) {
   IXML_Document *doc = ixmlDocument_createDocument();

   IXML_Element *envelope= ixmlDocument_createElementNS(doc,
"http://schemas.xmlsoap.org/soap/envelope/", "soap:Envelope");
   ixmlElement_setAttribute(envelope,"xmlns:soap","http://schemas.xmlsoap.org
/soap/envelope/" );
   ixmlElement_setAttribute(envelope,"soap:encodingStyle","http://schemas.xml
soap.org/soap/encoding/" );
   ixmlElement_setAttribute(envelope,"xmlns:cwmp", "urn:dslforum-org:cwmp-1-
0" );
   ixmlNode_appendChild( (IXML_Node *)doc,(IXML_Node *) envelope);

   if(sh!=NULL) {
      IXML_Node *header = createHeader(doc, sh);
      ixmlNode_appendChild( (IXML_Node *)envelope,(IXML_Node *)header);
   }

   IXML_Element *body=ixmlDocument_createElementNS(doc,
"http://schemas.xmlsoap.org/soap/envelope/", "soap:Body");
   ixmlNode_appendChild( (IXML_Node *)envelope,(IXML_Node *)body);


   IXML_Element *inform=ixmlDocument_createElementNS(doc,
"xmlns:cwmp=urn:dslforum-org:cwmp-1-0", "cwmp:GetParameterNames");
   ixmlNode_appendChild( (IXML_Node *)body,(IXML_Node *)inform);

   IXML_Element *parpath=ixmlDocument_createElementNS(doc,
"xmlns:cwmp=urn:dslforum-org:cwmp-1-0", "ParameterPath");
   IXML_Node* parpath_text=ixmlDocument_createTextNode(doc, parameterPath);
   ixmlNode_appendChild( (IXML_Node *)parpath,(IXML_Node *)parpath_text);

   IXML_Element *nextlev=ixmlDocument_createElementNS(doc,
"xmlns:cwmp=urn:dslforum-org:cwmp-1-0", "NextLevel");
   char *nl;
   if(nextLevel==0) nl = "false"; else nl="true";
   IXML_Node* nextlev_text=ixmlDocument_createTextNode(doc, nl );
```

```
   ixmlNode_appendChild( (IXML_Node *)nextlev,(IXML_Node *)nextlev_text);

   ixmlNode_appendChild( (IXML_Node *)inform,(IXML_Node *)parpath);
   ixmlNode_appendChild( (IXML_Node *)inform,(IXML_Node *)nextlev);

   return doc;

}


IXML_Document *createGetParameterNamesResponse(SoapHeader *sh, char *path,
unsigned short nextLevel){
   unsigned int i;
   IXML_Document *doc = ixmlDocument_createDocument();

   IXML_Element *envelope= ixmlDocument_createElementNS(doc,
"http://schemas.xmlsoap.org/soap/envelope/", "soap:Envelope");
   ixmlElement_setAttribute(envelope,"xmlns:soap","http://schemas.xmlsoap.org
/soap/envelope/" );
   ixmlElement_setAttribute(envelope,"soap:encodingStyle","http://schemas.xml
soap.org/soap/encoding/" );
   ixmlElement_setAttribute(envelope,"xmlns:xsi",
"http://www.w3.org/2001/XMLSchema/instance/" );
   ixmlElement_setAttribute(envelope,"xmlns:xsd",
"http://www.w3.org/2001/XMLSchema/" );
   ixmlElement_setAttribute(envelope,"xmlns:cwmp", "urn:dslforum-org:cwmp-1-
0" );
   ixmlNode_appendChild( (IXML_Node *)doc,(IXML_Node *) envelope);

   if(sh!=NULL) {
      IXML_Node *header = createHeader(doc, sh);
      ixmlNode_appendChild( (IXML_Node *)envelope,(IXML_Node *)header);
   }

   IXML_Element *body=ixmlDocument_createElementNS(doc,
"http://schemas.xmlsoap.org/soap/envelope/", "soap:Body");
   ixmlNode_appendChild( (IXML_Node *)envelope,(IXML_Node *)body);

   IXML_Element *response=ixmlDocument_createElementNS(doc,
"xmlns:cwmp=urn:dslforum-org:cwmp-1-0", "cwmp:GetParameterNamesResponse");
   ixmlNode_appendChild( (IXML_Node *)body,(IXML_Node *)response);

   // Gia arxh de 9a lhf9ei up' oyin to nextLevel

   //IXML_Element *plist= ixmlDocument_createElementNS(doc,
"xmlns:cwmp=urn:dslforum-org:cwmp-1-0", "ParameterList");


   if(strcmp(path, "")==0) { //Adeio path
      IXML_Element *parameterList = ixmlDocument_createElementNS(doc,
"xmlns:cwmp=urn:dslforum-org:cwmp-1-0", "ParameterList");

   ixmlElement_setAttribute(parameterList,"xmlns:soap","http://schemas.xmlsoa
p.org/soap/envelope/" );
      char a[50];
      sprintf(a, "cwmp:ParameterInfoStruct[%d]", NUMOFVARS);
      ixmlElement_setAttribute(parameterList,"soap:arrayType",a );
      ixmlNode_appendChild( (IXML_Node *)response,(IXML_Node
*)parameterList);

      for(i=0;i<NUMOFVARS;i++) {
         IXML_Element *el = ixmlDocument_createElementNS(doc,
"xmlns:cwmp=urn:dslforum-org:cwmp-1-0", "ParameterInfoStruct");

         IXML_Element *name = ixmlDocument_createElementNS(doc,
"xmlns:cwmp=urn:dslforum-org:cwmp-1-0", "Name");
         IXML_Node *name_text = ixmlDocument_createTextNode(doc,
deviceVariables[i].name);
```

```
        ixmlNode_appendChild( (IXML_Node *)name,(IXML_Node *)name_text);

        IXML_Element *writable = ixmlDocument_createElementNS(doc,
"xmlns:cwmp=urn:dslforum-org:cwmp-1-0", "Writable");
        IXML_Node *writable_text = ixmlDocument_createTextNode(doc, (char
*)(deviceVariables[i].writable?"true":"false"));
        ixmlNode_appendChild( (IXML_Node *)writable,(IXML_Node
*)writable_text);

        ixmlNode_appendChild( (IXML_Node *)el,(IXML_Node *)name);
        ixmlNode_appendChild( (IXML_Node *)el,(IXML_Node *)writable);
        ixmlNode_appendChild( (IXML_Node *)parameterList,(IXML_Node *)el);
     }
   }

   for(i=0;i<NUMOFVARS;i++) {

      if(strncmp(path, deviceVariables[i].name, strlen(path) )==0) {


      }
   }


   return doc;
}


IXML_Document *createFaultResponse(SoapHeader *sh, int code) {
   IXML_Document *doc = ixmlDocument_createDocument();

   IXML_Element *envelope= ixmlDocument_createElementNS(doc,
"http://schemas.xmlsoap.org/soap/envelope/", "soap:Envelope");
   ixmlElement_setAttribute(envelope,"xmlns:soap","http://schemas.xmlsoap.org
/soap/envelope/" );
   ixmlElement_setAttribute(envelope,"soap:encodingStyle","http://schemas.xml
soap.org/soap/encoding/" );
   ixmlElement_setAttribute(envelope,"xmlns:cwmp", "urn:dslforum-org:cwmp-1-
0" );
   ixmlNode_appendChild( (IXML_Node *)doc,(IXML_Node *) envelope);

   if(sh!=NULL) {
      IXML_Node *header = createHeader(doc, sh);
      ixmlNode_appendChild( (IXML_Node *)envelope,(IXML_Node *)header);
   }

   IXML_Element *body=ixmlDocument_createElementNS(doc,
"http://schemas.xmlsoap.org/soap/envelope/", "soap:Body");
   ixmlNode_appendChild( (IXML_Node *)envelope,(IXML_Node *)body);

   IXML_Element *fault=ixmlDocument_createElementNS(doc,
"http://schemas.xmlsoap.org/soap/envelope/", "soap:Fault");
   ixmlNode_appendChild( (IXML_Node *)body,(IXML_Node *)fault);

   IXML_Element *faultcode=ixmlDocument_createElementNS(doc,
"http://schemas.xmlsoap.org/soap/envelope/", "faultcode");
   ixmlNode_appendChild( (IXML_Node *)fault,(IXML_Node *)faultcode);
   IXML_Node *faultcode_text= ixmlDocument_createTextNode(doc,"Client");
   ixmlNode_appendChild( (IXML_Node *)faultcode,(IXML_Node *)faultcode_text);

   IXML_Element *faultstring=ixmlDocument_createElementNS(doc,
"http://schemas.xmlsoap.org/soap/envelope/", "faultstring");
   ixmlNode_appendChild( (IXML_Node *)fault,(IXML_Node *)faultstring);
   IXML_Node *faultstring_text= ixmlDocument_createTextNode(doc,"CWMP
fault");
   ixmlNode_appendChild( (IXML_Node *)faultstring,(IXML_Node
*)faultstring_text);
```

```
   IXML_Element *detail = ixmlDocument_createElementNS(doc,
"http://schemas.xmlsoap.org/soap/envelope/", "detail");
   ixmlNode_appendChild( (IXML_Node *)fault,(IXML_Node *)detail);

   IXML_Element *faultcwmp=ixmlDocument_createElementNS(doc, "urn:dslforum-
org:cwmp-1-0", "cwmp:Fault");
   ixmlNode_appendChild( (IXML_Node *)detail,(IXML_Node *)faultcwmp);


   IXML_Element *faultcodecwmp=ixmlDocument_createElementNS(doc,
"urn:dslforum-org:cwmp-1-0", "FaultCode");
   ixmlNode_appendChild( (IXML_Node *)faultcwmp,(IXML_Node *)faultcodecwmp);
   char a[5];
   sprintf(a, "%d", code);
   IXML_Node *faultcodecwmp_text= ixmlDocument_createTextNode(doc,a);
   ixmlNode_appendChild( (IXML_Node *)faultcodecwmp,(IXML_Node
*)faultcodecwmp_text);

   IXML_Element *faultstringcwmp=ixmlDocument_createElementNS(doc,
"urn:dslforum-org:cwmp-1-0", "FaultString");
   ixmlNode_appendChild( (IXML_Node *)faultcwmp,(IXML_Node
*)faultstringcwmp);
   IXML_Node *faultstringcwmp_text= ixmlDocument_createTextNode(doc,
getErrorByNum(code) );
   ixmlNode_appendChild( (IXML_Node *)faultstringcwmp,(IXML_Node
*)faultstringcwmp_text);

   return doc;

}


IXML_Document *createSetFaultResponse(SoapHeader *sh, ParameterValueList
*params, int * errors) {
   unsigned int i;
   IXML_Document *doc = ixmlDocument_createDocument();

   IXML_Element *envelope= ixmlDocument_createElementNS(doc,
"http://schemas.xmlsoap.org/soap/envelope/", "soap:Envelope");
   ixmlElement_setAttribute(envelope,"xmlns:soap","http://schemas.xmlsoap.org
/soap/envelope/" );
   ixmlElement_setAttribute(envelope,"soap:encodingStyle","http://schemas.xml
soap.org/soap/encoding/" );
   ixmlElement_setAttribute(envelope,"xmlns:cwmp", "urn:dslforum-org:cwmp-1-
0" );
   ixmlNode_appendChild( (IXML_Node *)doc,(IXML_Node *) envelope);

   if(sh!=NULL) {
      IXML_Node *header = createHeader(doc, sh);
      ixmlNode_appendChild( (IXML_Node *)envelope,(IXML_Node *)header);
   }

   IXML_Element *body=ixmlDocument_createElementNS(doc,
"http://schemas.xmlsoap.org/soap/envelope/", "soap:Body");
   ixmlNode_appendChild( (IXML_Node *)envelope,(IXML_Node *)body);

   IXML_Element *fault=ixmlDocument_createElementNS(doc,
"http://schemas.xmlsoap.org/soap/envelope/", "soap:Fault");
   ixmlNode_appendChild( (IXML_Node *)body,(IXML_Node *)fault);

   IXML_Element *faultcode=ixmlDocument_createElementNS(doc,
"http://schemas.xmlsoap.org/soap/envelope/", "faultcode");
   ixmlNode_appendChild( (IXML_Node *)fault,(IXML_Node *)faultcode);
   IXML_Node *faultcode_text= ixmlDocument_createTextNode(doc,"Client");
   ixmlNode_appendChild( (IXML_Node *)faultcode,(IXML_Node *)faultcode_text);

   IXML_Element *faultstring=ixmlDocument_createElementNS(doc,
"http://schemas.xmlsoap.org/soap/envelope/", "faultstring");
```

```
   ixmlNode_appendChild( (IXML_Node *)fault,(IXML_Node *)faultstring);
   IXML_Node *faultstring_text= ixmlDocument_createTextNode(doc,"CWMP
fault");
   ixmlNode_appendChild( (IXML_Node *)faultstring,(IXML_Node
*)faultstring_text);

   IXML_Element *detail = ixmlDocument_createElementNS(doc,
"http://schemas.xmlsoap.org/soap/envelope/", "detail");
   ixmlNode_appendChild( (IXML_Node *)fault,(IXML_Node *)detail);

   IXML_Element *faultcwmp=ixmlDocument_createElementNS(doc, "urn:dslforum-
org:cwmp-1-0", "cwmp:Fault");
   ixmlNode_appendChild( (IXML_Node *)detail,(IXML_Node *)faultcwmp);

   IXML_Element *faultcodecwmp=ixmlDocument_createElementNS(doc,
"urn:dslforum-org:cwmp-1-0", "FaultCode");
   ixmlNode_appendChild( (IXML_Node *)faultcwmp,(IXML_Node *)faultcodecwmp);
   IXML_Node *faultcodecwmp_text= ixmlDocument_createTextNode(doc,"9003");
   ixmlNode_appendChild( (IXML_Node *)faultcodecwmp,(IXML_Node
*)faultcodecwmp_text);

   IXML_Element *faultstringcwmp=ixmlDocument_createElementNS(doc,
"urn:dslforum-org:cwmp-1-0", "FaultString");
   ixmlNode_appendChild( (IXML_Node *)faultcwmp,(IXML_Node
*)faultstringcwmp);
   IXML_Node *faultstringcwmp_text= ixmlDocument_createTextNode(doc, "Invalid
arguments" );
   ixmlNode_appendChild( (IXML_Node *)faultstringcwmp,(IXML_Node
*)faultstringcwmp_text);

   for(i=0;i<params->size;i++) {
      if(errors[i]!=0) {
         IXML_Element *setparamsfault =ixmlDocument_createElementNS(doc,
"urn:dslforum-org:cwmp-1-0", "SetParameterValuesFault");
         ixmlNode_appendChild( (IXML_Node *)faultcwmp,(IXML_Node
*)setparamsfault);

         IXML_Element *pn=ixmlDocument_createElementNS(doc, "urn:dslforum-
org:cwmp-1-0", "ParameterName");
         ixmlNode_appendChild( (IXML_Node *)setparamsfault,(IXML_Node *)pn);
         IXML_Node *pn_text= ixmlDocument_createTextNode(doc,params-
>parameters[i].name);
         ixmlNode_appendChild( (IXML_Node *)pn,(IXML_Node *)pn_text);

         IXML_Element *fc=ixmlDocument_createElementNS(doc, "urn:dslforum-
org:cwmp-1-0", "FaultCode");
         ixmlNode_appendChild( (IXML_Node *)setparamsfault,(IXML_Node *)fc);
         char a[5];
         sprintf(a, "%d", -errors[i]);
         IXML_Node *fc_text= ixmlDocument_createTextNode(doc,a);
         ixmlNode_appendChild( (IXML_Node *)fc,(IXML_Node *)fc_text);

         IXML_Element *fs=ixmlDocument_createElementNS(doc, "urn:dslforum-
org:cwmp-1-0", "FaultString");
         ixmlNode_appendChild( (IXML_Node *)setparamsfault,(IXML_Node *)fs);
         IXML_Node *fs_text= ixmlDocument_createTextNode(doc,getErrorByNum(-
errors[i]) );
         ixmlNode_appendChild( (IXML_Node *)fs,(IXML_Node *)fs_text);


      }

   }

   return doc;


}
```

```
IXML_Node *createHeader(IXML_Document* doc, SoapHeader *sh) {
   IXML_Element *header=ixmlDocument_createElementNS(doc,
"http://schemas.xmlsoap.org/soap/envelope/", "soap:Header");

   IXML_Element *id=ixmlDocument_createElementNS(doc, "urn:dslforum-org:cwmp-
1-0", "cwmp:ID");
   ixmlElement_setAttribute(id,"soap:mustUnderstand", "1" );
   IXML_Node* id_text=ixmlDocument_createTextNode(doc, sh->id);
   ixmlNode_appendChild( (IXML_Node *) id, id_text);
   ixmlNode_appendChild( (IXML_Node *)header,(IXML_Node *)id);

   if(sh->holdRequests!=0) {
      IXML_Element *holdreqs=ixmlDocument_createElementNS(doc, "urn:dslforum-
org:cwmp-1-0", "cwmp:HoldRequests");
      ixmlElement_setAttribute(holdreqs,"soap:mustUnderstand", "1" );
      char *buf = (char *)malloc(4*sizeof(char) ) ;
      sprintf(buf, "%d", sh->holdRequests);
      IXML_Node* holdreqs_text=ixmlDocument_createTextNode(doc, buf );
      free(buf);
      ixmlNode_appendChild( (IXML_Node *) holdreqs, holdreqs_text);
      ixmlNode_appendChild( (IXML_Node *)header,(IXML_Node *)holdreqs);
   }

   if(sh->noMoreRequests!=0) {
      IXML_Element *nmr=ixmlDocument_createElementNS(doc, "urn:dslforum-
org:cwmp-1-0", "cwmp:NoMoreRequests");
      char *buf = (char *)malloc(4*sizeof(char) ) ;
      sprintf(buf, "%d", sh->noMoreRequests);
      IXML_Node* nmr_text=ixmlDocument_createTextNode(doc, buf);
      free(buf);
      ixmlNode_appendChild( (IXML_Node *) nmr, nmr_text);
      ixmlNode_appendChild( (IXML_Node *)header,(IXML_Node *)nmr);
   }

   return (IXML_Node *)header;
}
```

8.6.15 typechk.h

```
/* Papastefanos Serafeim */
/* Sunarthseis gia elegxo tupwn */

#ifndef _TYPECHK_H
#define _TYPECHK_H

#include "client.h"

/* Oi epomenes sunarthseis pernoun ws parametro ena string (kai duo ej
 * autwn pairnoun kai ena integer pou ka9orizei to megisto mege9os) kai
 * ejetazoun an sto string auto uparxei o katallhlos tupos pou fainetai
 * apo to onoma, dhladh string, int, unsignedInt, boolean, dateTime kai
 * base64. Epistrefetai 0 an sto string uparxei o katallhlos tupos kai
 * arnhtikh timh an den uparxei. */
int checkString(char *buf, unsigned int len);
int checkInt(char *buf);
int checkUnsignedInt(char *buf);
int checkBoolean(char *buf);
int checkDateTime(char *buf);
int checkBase64(char *buf, unsigned  int len);

/* Oi epomenes sunarthseis kanoun parse to string kai epistrefoun
 * ton katallhlo tupo. */
long int parseInt(char *buf);
unsigned long int parseUnsignedInt(char *buf);
int parseBoolean(char *buf);
```

```
DateTime parseDateTime(char *buf);

/* Auth h sunarthsh kaleitai prin klh9ei h handleSetParameterValues, wste
 * na elex9ei to an oi parametroi pou perniountai einai oi swstes. */
int checkSetParameterValues(ParameterValueList *parameterList, int *errors);

#endif //_TYPECHK_H
```

8.6.16 typechk.cpp

```
/* Papastefanos Serafeim */
/* Sunarthseis gia elegxo tupwn */

#include "client.h"
#include "typechk.h"
#include "ctype.h"
#include "stdio.h"
#include "string.h"
#include "helpers.h"
#include "custom.h"
#include <stdlib.h>
#include <limits.h>

/* H sunarthsh auth pairnei mia lista parametrwn pou prokeitai na
 * perastei sthn handleSetParameterValues kai enan pinaka apo
 * akeraious, me mege9os iso me th lista paraketrwn. Elegxei ka9e
 * mia apo tis parametrous ths listas xrhsimopoiwntas thn checkParameter
 * h opoia vrisketai sto custom.c kai an uparxei provlhma
 * ti9etai sthn katallhlh 9esh tou pinaka errors h timh tou sfalmatos.
 * An ginei estw kai ena sfalma epistrefetai to -INVALID_ARGUMENTS,
 * opote h sunarthsh pou kalese thn checkSetParameterValues 9a
 * elegjei kai ton pinaka errors gia mh mhdenikes times wste na
 * vrei pou akrivws egine to la9os / la9h. */
int checkSetParameterValues(ParameterValueList *parameterList, int *errors)
{
   unsigned int i;
   int ok = 1;
   int r = 0;
   for(i=0;i<parameterList->size;i++) errors[i]=0;

   for(i=0;i<parameterList->size;i++) {
      StateVariable *sv = getStateVariable(parameterList-
>parameters[i].name);
      if(sv->writable==0) {
         ok = 0;
         errors[i]=-ATTEMPT_TO_SET_NON_WRITABLE;
      } else if( (r=checkParameter(parameterList->parameters[i]))!=0) {
         ok = 0;
         errors[i]=r;
      }
   }
   if(ok==0) return -INVALID_ARGUMENTS; else return 0;
}

/* Elegxei an to string buf einai katallhlou mege9ous */
int checkString(char *buf, unsigned int len) {
   if(buf == NULL) return -INTERNAL_ERROR;
   if(strlen(buf) > len) return -INVALID_PARAMETER_VALUE;
   return 0;
}

/* Elegxei an sto buf uparxei katallhlos akeraios */
int checkInt(char *buf) {
   if(buf == NULL) return -INTERNAL_ERROR;
   long int tmp = strtol(buf, NULL, 10);
   if(tmp == LONG_MAX || tmp == LONG_MIN) return -INVALID_PARAMETER_VALUE;
   int i=0;
```

```
   while(isspace(buf[i]) && buf[i]!=0 ) i++;
   if(buf[i]==0) return  -INVALID_PARAMETER_TYPE;
   if(!isdigit(buf[i])) {
      if(buf[i]!='+' && buf[i]!='-') return -INVALID_PARAMETER_TYPE;
      i++;
   }
   while(isdigit(buf[i])&& buf[i]!=0) i++;
   if(buf[i]==0) return 0;
   else if(isspace(buf[i])) {
      while(isspace(buf[i])&& buf[i]!=0) i++;
      if(buf[i]==0) return 0;
      else return -INVALID_PARAMETER_TYPE;
   } else return -INVALID_PARAMETER_TYPE;

   return 0;
}

/* Elegxei an sto buf uparxei katallhlos aproshmos akeraios */
int checkUnsignedInt(char *buf) {
   if(buf == NULL) return -INTERNAL_ERROR;
   long int tmp = strtol(buf, NULL, 10);
   if(tmp == LONG_MAX || tmp == LONG_MIN) return -INVALID_PARAMETER_VALUE;
   int i=0;
   while(isspace(buf[i]) && buf[i]!=0 ) i++;
   if(buf[i]==0) return  -INVALID_PARAMETER_TYPE;
   if(!isdigit(buf[i])) -INVALID_PARAMETER_TYPE;
   while(isdigit(buf[i])&& buf[i]!=0) i++;
   if(buf[i]==0) return 0;
   else if(isspace(buf[i])) {
      while(isspace(buf[i])&& buf[i]!=0) i++;
      if(buf[i]==0) return 0;
      else return -INVALID_PARAMETER_TYPE;
   } else return -INVALID_PARAMETER_TYPE;

   return 0;
}

/* Elegxei an sto buf uparxei 0 h 1 (boolean) */
int checkBoolean(char *buf) {
   int i=0;
   if(buf == NULL) return -INTERNAL_ERROR;
   while(isspace(buf[i]) && buf[i]!=0 ) i++;
   if(buf[i]==0) return  -INVALID_PARAMETER_TYPE;
   if( (buf[i]!='0')&& (buf[i]!='1') ) return -INVALID_PARAMETER_TYPE;
   i++;
   if(buf[i]==0) return 0;
   else if(isspace(buf[i])) {
      while(isspace(buf[i])&& buf[i]!=0) i++;
      if(buf[i]==0) return 1;
      else return -INVALID_PARAMETER_TYPE;
   } else return -INVALID_PARAMETER_TYPE;
   return 0;
}

/* Elegxei an sto buf uparxei DateTime */
int checkDateTime(char *buf) {
   int i=0;
   if(buf == NULL) return -INTERNAL_ERROR;
   while(isspace(buf[i]) && buf[i]!=0 ) i++;
   if(buf[i]==0) return  -INVALID_PARAMETER_TYPE;
   if(isdigit(buf[i])) i++; else return -INVALID_PARAMETER_TYPE;
   if(isdigit(buf[i])) i++; else return -INVALID_PARAMETER_TYPE;
   if(isdigit(buf[i])) i++; else return -INVALID_PARAMETER_TYPE;
   if(isdigit(buf[i])) i++; else return -INVALID_PARAMETER_TYPE;
   if(buf[i]=='-') i++; else return -INVALID_PARAMETER_TYPE;
   if(isdigit(buf[i])) i++; else return -INVALID_PARAMETER_TYPE;
   if(isdigit(buf[i])) i++; else return -INVALID_PARAMETER_TYPE;
   if(buf[i]=='-') i++; else return -INVALID_PARAMETER_TYPE;
```

```
   if(isdigit(buf[i])) i++; else return -INVALID_PARAMETER_TYPE;
   if(isdigit(buf[i])) i++; else return -INVALID_PARAMETER_TYPE;
   if(buf[i]=='T') i++; else return -INVALID_PARAMETER_TYPE;
   if(isdigit(buf[i])) i++; else return -INVALID_PARAMETER_TYPE;
   if(isdigit(buf[i])) i++; else return -INVALID_PARAMETER_TYPE;
   if(buf[i]==':') i++; else return -INVALID_PARAMETER_TYPE;
   if(isdigit(buf[i])) i++; else return -INVALID_PARAMETER_TYPE;
   if(isdigit(buf[i])) i++; else return -INVALID_PARAMETER_TYPE;
   if(buf[i]==':') i++; else return -INVALID_PARAMETER_TYPE;
   if(isdigit(buf[i])) i++; else return -INVALID_PARAMETER_TYPE;
   if(isdigit(buf[i])) i++; else return -INVALID_PARAMETER_TYPE;
   if(buf[i]==0) return 0;
   else if(isspace(buf[i])) {
      while(isspace(buf[i])&& buf[i]!=0) i++;
      if(buf[i]==0) return 1;
      else return -INVALID_PARAMETER_TYPE;
   } else return -INVALID_PARAMETER_TYPE;
   return 0;
}

/* Elegxei an sto buf uparxei base64 mege9ous len. Ena base64 string
 * epitrepetai na exei tous xarakthres A-Z, a-z, 0-9, +, / kai =. An
 * uparxei allos xarakthras 9a epistrafei -INVALID_PARAMTER_TYPE. */
int checkBase64(char *buf, unsigned int len) {
   unsigned int i;
   if(buf == NULL) return -INTERNAL_ERROR;
   if(strlen(buf) > len) return -INVALID_PARAMETER_VALUE;
   for(i=0;i<strlen(buf);i++) {
      if( isalpha(buf[i]) || isdigit(buf[i]) ) continue ;
      if( (buf[i] == '+')||(buf[i] == '/')||buf[i] == '=' ) continue ;

      return -INVALID_PARAMETER_TYPE;
   }
   return 0;
}

/* Epistrefei ton akeraio pou uparxei sto buf. Kaleitai mono afou
 * exei hdh klh9ei h checkInt */
long int parseInt(char *buf) {
   return( strtol(buf, NULL, 10) );

}

/* Epistrefei ton aproshmo pou uparxei sto buf. Kaleitai mono afou
 * exei hdh klh9ei h checkUnsignedInt */
unsigned long int parseUnsignedInt(char *buf) {
   return( strtoul(buf, NULL, 10) );
}

/* Epistrefei ton boolean pou uparxei sto buf. Kaleitai mono afou
 * exei hdh klh9ei h checkBoolean */
int parseBoolean(char *buf) {
   if(atoi(buf) == 0) return 0;
   else return 1;
}

/* Epistrefei to DateTime pou uparxei sto buf. Kaleitai mono afou
 * exei hdh klh9ei h checkDateTime */
DateTime parseDateTime(char *buf) {
   int i=0;
   while(isspace(buf[i]))i++;
   DateTime *dt=ascitodate(&buf[i]);
   DateTime dateTime = *dt;
   return dateTime;
}
```

8.6.17 custom.h

```
#ifndef _CUSTOM_H
#define _CUSTOM_H

#include "client.h"
#define NUMOFVARS 5

extern StateVariable deviceVariables[];

DeviceIdStruct *getDeviceId(void);
ParameterValueList *getInformParameterList(void);
int getConnectionAddress(char *addr, int *port);
char* getConnectionString();
unsigned int getListeningPort();
int checkParameter(ParameterValueStruct pv);

void fun1(ParameterValueList *);

void fun2(ParameterValueList *);

#endif //_CUSTOM_H
```

8.6.18 custom.cpp

```
#include "custom.h"
#include "helpers.h"
#include "stdlib.h"
#include "typechk.h"

/***** STATE VARIABLES ******/
StateVariable deviceVariables[NUMOFVARS] = {
   "Calc.ManagementServer.URL", fun2, fun1, 1,
   "Calc.ManagementServer.ConnectionRequestURL", fun2, NULL, 0,
   "Calc.Add.num1", fun2, fun1, 1,
   "Calc.Add.num2", fun2, fun1, 1,
   "Calc.Add.result", fun2, NULL, 0,
};

/***** STATIC VARIABLES ******/
/* These are used with stand alone state varriables */
static char *__ACSconnectionURL="http://127.0.0.1:80";
static char *__CPEconnectionString="http://127.0.0.1:24582/CALC_CONN";
static long int num1=0;
static long int num2=0;

/***** SET FUNCTIONS *****/
void fun1(ParameterValueList *params) {
   unsigned int i;
   //TODO: Add initialization code for this function here...

   for(i=0;i<params->size;i++){
      if(strcmp(params->parameters[i].name, "Calc.ManagementServer.URL")==0)
{
         char *value = params->parameters[i].value;
         printf("Klh9hke h set gia thn Calc.ManagementServer.URL me timh %s",
value);
         __ACSconnectionURL = strdup(value);
      }

      if(strcmp(params->parameters[i].name, "Calc.Add.num1")==0) {
         long int value = parseInt(params->parameters[i].value);
         printf("Klh9hke h set gia thn Calc.Add.num1 me timh %d", value);
         num1 = value ;
      }

      if(strcmp(params->parameters[i].name, "Calc.Add.num2")==0) {
```

```
        long int value = parseInt(params->parameters[i].value);
        printf("Klh9hke h set gia thn Calc.Add.num2 me timh %d", value);
        num2 = value ;
     }

   }
   //return 0;
}


/***** GET FUNCTIONS *****/
void fun2(ParameterValueList *params) {
   unsigned int i;
   //TODO: Add initialization code for this function here...

   for(i=0;i<params->size;i++){
      if(strcmp(params->parameters[i].name, "Calc.ManagementServer.URL")==0)
{
        char *value;
        printf("Klh9hke h get gia thn Calc.ManagementServer.URL");
        value = __ACSconnectionURL;
        params->parameters[i].value = strdup(value);
     }

     if(strcmp(params->parameters[i].name,
"Calc.ManagementServer.ConnectionRequestURL")==0) {
        char *value;
        printf("Klh9hke h get gia thn
Calc.ManagementServer.ConnectionRequestURL");
        value = __CPEconnectionString;
        params->parameters[i].value = strdup(value);
     }

     if(strcmp(params->parameters[i].name, "Calc.Add.num1")==0) {
        long int value;
        printf("Klh9hke h get gia thn Calc.Add.num1");
        value = num1;
        char __a[20];
        sprintf(__a,"%ld",value);
        params->parameters[i].value = strdup(__a);
     }

     if(strcmp(params->parameters[i].name, "Calc.Add.num2")==0) {
        long int value;
        printf("Klh9hke h get gia thn Calc.Add.num2");
        value = num2;
        char __a[20];
        sprintf(__a,"%ld",value);
        params->parameters[i].value = strdup(__a);
     }

     if(strcmp(params->parameters[i].name, "Calc.Add.result")==0) {
        long int value;
        printf("Klh9hke h get gia thn Calc.Add.result");
        value=0;
        //TODO: add code here so that value = "Calc.Add.result"
        value = num1+num2; //// Added only 1 line of code ! ////
        char __a[20];
        sprintf(__a,"%ld",value);
        params->parameters[i].value = strdup(__a);
     }

   }
   //return 0;
}


/***** OTHERS *****/
```

```
DeviceIdStruct *getDeviceId(void) {
   DeviceIdStruct *deviceId = (DeviceIdStruct *)malloc(sizeof(DeviceIdStruct)
);
   deviceId->manufactorer = "Serafeim";
   deviceId->OUI = "123456";
   deviceId->productClass = "Calculator";
   deviceId->serialNumber = "1212121";
   return deviceId;
}

ParameterValueList *getInformParameterList(void) {
   ParameterValueList *parameterList = newParameterValueList(2);
   parameterList->parameters[0].name = strdup("Calc.ManagementServer.URL");
   parameterList->parameters[0].value = strdup( "http://127.0.0.1:80");
   parameterList->parameters[1].name =
strdup("Calc.ManagementServer.ConnectionRequestURL");
   parameterList->parameters[1].value = strdup(
"http://127.0.0.1:24582/CALC_CONN");
   return parameterList;
}

int getConnectionAddress(char *__addr, int *__port) {
   char *__a9293 = __ACSconnectionURL;
   char *__tmp245;
   int __p1, __p2;
   __tmp245 = strstr(__a9293,"//");
   if(__tmp245 == NULL) __p1 = 0;
   else __p1 = __tmp245 - __a9293 + 2;
   __tmp245 = strstr(__a9293+__p1, ":");
   if(__tmp245 == NULL) {
      *__port = 80;
      __p2 = strlen(__a9293);
   } else {
      __p2 = __tmp245 - __a9293;
      *__port = atoi(__a9293+__p2+1);
   }
   strncpy(__addr, __a9293+__p1, __p2-__p1);
   *((__addr)+__p2-__p1) = 0;

   return 0;
}

char *getConnectionString() {
   return __CPEconnectionString;
}

unsigned int getListeningPort() {
   return 24582;
}

int checkParameter(ParameterValueStruct pv) {
   if(strcmp(pv.name, "Calc.ManagementServer.URL")==0) {
      return (checkString(pv.value, 256));
   }
   if(strcmp(pv.name, "Calc.Add.num1")==0) {
      return (checkInt(pv.value));
   }
   if(strcmp(pv.name, "Calc.Add.num2")==0) {
      return (checkInt(pv.value));
   }

   return -INVALID_PARAMETER_NAME;
}
```