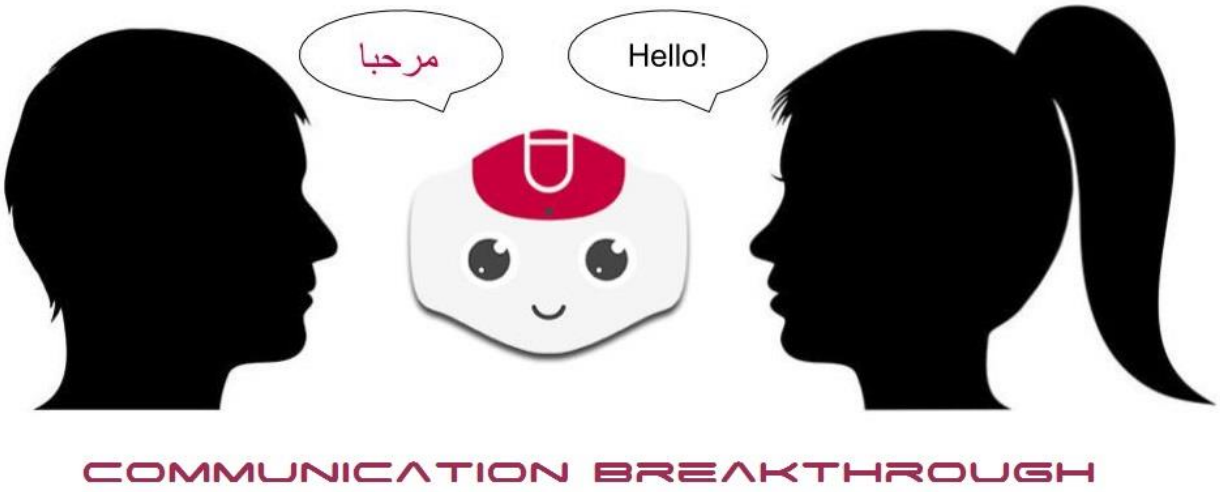
NAO Translate



Απαιτήσεις Λογισμικού
(Προδιαγραφές Λογισμικού)

Del.1.2

Version 0.2

Βαρβαρίγγος Ιωάννης 8782 varvarip@ece.auth.gr

Γιαννακίδου Σοφία 8974 sagiannaki@ece.auth.gr

Σαχίνης Αλέξανδρος 8906 alexsach@ece.auth.gr

Χατζηχαριστού Αλεξάνδρα 8943 chatzich@ece.auth.gr

27/03/2018

Ιστορικό Αλλαγών

Όνομα	Ημερομηνία	Αλλαγή	Έκδοση
Α. Συμεωνίδης	17/05/2007	Δημιουργία εγγράφου. Προσαρμογή των προτύπων του K. E. Wiegers[♣] και του M. Smialek's.	0.1
Α. Συμεωνίδης	29/3/2014	Προσαρμογή του εγγράφου.	0.1.3
NAO Translate	29/4/2018	Προσθήκη πακέτων στατικής μοντελοποίησης	0.1.4
NAO Translate	30/4/2018	Προσθήκη προτύπων σχεδιασμού	0.1.5
NAO Translate	1/5/2018	Τελική μορφοποίηση κειμένου	0.2

Μέλη της Ομάδας Ανάπτυξης

Όνομα	ΟΑ	Email
Α. Συμεωνίδης	*	asymeon@issel.ee.auth.gr
Ι. Βαρβαρίγγος	NAO Translate	varvarip@ece.auth.gr
Σ. Γιαννακίδου	NAO Translate	sagiannaki@ece.auth.gr
Α. Σαχίνης	NAO Translate	alexsach@ece.auth.gr
Α. Χατζηχαριστού	NAO Translate	chatzich@ece.auth.gr

Πίνακας Περιεχομένων

Λίστα Σχημάτων

1. Εισαγωγικά

1.1 Στόχος του Εγγράφου

Το παρόν έγγραφο αποσκοπεί στο να καταστούν κατανοητές οι απαιτήσεις του λογισμικού που διαμορφώνει το σύστημα. Αποτελεί βοήθημα για την πλήρη περιγραφή και αναγνώριση των χαρακτηριστικών αυτού, με κύριο γνώμονα τις απαιτήσεις χρηστών που περιγράφηκαν στο προηγούμενο έγγραφο.

Στόχος του εγγράφου είναι ο αναλυτικός προσδιορισμός των λειτουργιών του συστήματος. Μελετάται το σύστημα και η αλληλεπίδρασή του με τους χρήστες και τα συνεργαζόμενα εξωτερικά συστήματα και αναλύεται ο δυναμικός χαρακτήρας του περιβάλλοντος διεπαφής αυτών. Παράλληλα, το παρόν έγγραφο αποσκοπεί στην περιγραφή των διαχειριζομένων από το σύστημα δεδομένων. Ορίζονται οι τύποι δεδομένων που διέπουν την επικοινωνία του συστήματος με τους χρήστες και τα εξωτερικά συστήματα. Επίσης, περιγράφονται πλήρως τα μηνύματα που επιστρέφει το σύστημα, μέσω των οποίων γίνεται ανταλλαγή δεδομένων.

Για την επίτευξη των στόχων αυτών, απαιτείται στατική μοντελοποίηση του συστήματος. Η στατική μοντελοποίηση του συστήματος σχετίζεται με τη μοντελοποίηση των απαιτήσεων χρηστών. Το σύστημα χωρίζεται σε πακέτα κλάσεων με βάση τις επιμέρους λειτουργικότητες του, οπότε εξασφαλίζεται ευκολότερη κατανόηση και ανάπτυξη του συστήματος. Ανάλογα με τη σπουδαιότητα της λειτουργίας του συστήματος που περιγράφεται, γίνεται και η ιεράρχηση των πακέτων αυτών σε πακέτα υψηλής, μέσης και χαμηλής προτεραιότητας.

Σε κάθε πακέτο περιλαμβάνεται ένα σύνολο κλάσεων, οι οποίες διακρίνονται σε κλάσεις οντοτήτων (entities), σε οριακές κλάσεις (boundaries) και σε κλάσεις ελέγχου (controllers). Οι κλάσεις οντοτήτων καθορίζουν τη μόνιμη πληροφορία του συστήματος. Οι οριακές κλάσεις ή κλάσεις διεπαφής χρηστών αναπαριστούν την αλληλεπίδραση μεταξύ του χρήστη και του συστήματος ή του συστήματος και άλλων εξωτερικών συστημάτων. Οι κλάσεις ελέγχου ή ρυθμιστικές κλάσεις είναι υπεύθυνες για τις διαδικασίες ελέγχου του συστήματος και τον συντονισμό των διεργασιών των επιμέρους κλάσεων. Ακόμη, σχεδιάζονται διαγράμματα για την αναπαράσταση των αλληλεπιδράσεων των κλάσεων και των πακέτων. Τέλος, περιγράφονται τα πρότυπα σχεδιασμού που υιοθετήθηκαν με στόχο την απόδοση επιπλέον λειτουργικότητας στο σύστημα.

Το παρόν έγγραφο θα αποτελέσει τον οδηγό για την ανάπτυξη του λογισμικού από την ομάδα ανάπτυξης.

1.2 Τυπογραφικές παραδοχές

Το παρόν έγγραφο χρησιμοποιεί γραμματοσειρά Calibri μεγέθους 11 για το κυρίως κείμενο. Για τις επικεφαλίδες το μέγεθος μπορεί να είναι 16, 14 ή 12, ανάλογα με το βάθος στο οποίο βρίσκεται η κάθε μία. Υπογράμμιση χρησιμοποιείται κατά την επεξήγηση των χαρακτηριστικών των κλάσεων του συστήματος που αναλύεται. Χρησιμοποιείται bold μωβ γραφή για την αναφορά σε ιδιότητες και μεθόδους μίας κλάσης υπό τη μορφή: **κλάση::Ιδιότητα: Τύπος της Ιδιότητας** και **κλάση::Μέθοδος(όρισμα: Τύπος του ορίσματος): Τύπος Αυτού που επιστρέφεται**. Τα διαγράμματα που παρουσιάζονται ακολουθούν τις παραδοχές της γλώσσας UML.

1.3 Αναγνωστικό κοινό

Το παρόν έγγραφο απευθύνεται στις εξής ομάδες ανθρώπων, με σκοπό τη μελέτη, μοντελοποίηση, σχεδιασμό, προγραμματισμό και τελικά υλοποίηση του συστήματος.

- **Ομάδα ανάπτυξης:** Με την πλήρη κατανόηση των απαιτήσεων λογισμικού του σχεδιαζόμενου συστήματος κάθε μέλος της ομάδας ανάπτυξης θα βρίσκεται σε θέση να συμβάλλει στην ανάλυση, στο σχεδιασμό και την υλοποίηση του συστήματος. Με κριτήριο τη δραστηριότητα ανάπτυξης τα μέλη της ΟΑ μπορούν να έχουν διακριτούς ρόλους. Ένας πιθανός τρόπος καταμερισμού των εργασιών δημιουργεί τους εξής ρόλους: Αναλυτής, Σχεδιαστής Συστήματος, Σχεδιαστής Αντικειμένων, Προγραμματιστής.
- **Ομάδα συντήρησης:** Η πλήρης κατανόηση της λειτουργικότητας και των βασικών οντοτήτων που δημιουργήθηκαν κατά την υλοποίηση και τον προγραμματισμό του συστήματος είναι απαραίτητη και για την βελτιστοποίηση αυτού από την ομάδα συντήρησης και αναβάθμισης.

1.4 Σκοπός του Έργου

Η εφαρμογή έχει σκοπό να χρησιμοποιεί το ΝΑΟ ως διερμηνέα. Η χρήση της εφαρμογής θα παρέχει τη δυνατότητα διεξαγωγής διαλόγου μεταξύ ατόμων που μιλάνε διαφορετικές γλώσσες. Με δυνατότητες συνομιλίας έως και 4 διαφορετικές γλώσσες ταυτόχρονα, γεφυρώνεται το χάσμα που υπήρχε ανάμεσα σε άτομα από διαφορετικές χώρες, ενώ με τη χρήση του ΝΑΟ διασφαλίζεται μια ευχάριστη συζήτηση χάρη στην, κατά το δυνατόν ανθρώπινη αλληλεπίδραση του με τους ομιλητές.

Η ομάδα ανάπτυξης προσδοκεί με τη δημιουργία της εφαρμογής να φέρει τους ανθρώπους πιο κοντά και να εμβαθύνει τις διαπροσωπικές σχέσεις των χρηστών της ανεξάρτητα από την καταγωγή τους. Η χρήση της μπορεί να αποτελέσει κίνητρο γνωριμίας άλλων πολιτισμών, διεύρυνσης των οριζόντων των χρηστών και να θέσει τα θεμέλια για έναν νέο τρόπο επικοινωνίας.

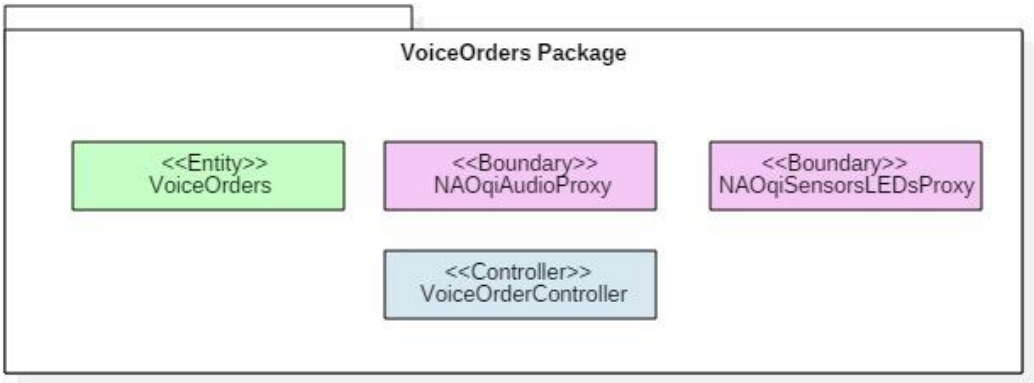
2. Στατική Μοντελοποίηση

Επισημαίνεται ότι επειδή στο παρόν έγγραφο δεν συμπεριλήφθηκε η δυναμική μοντελοποίηση του συστήματος, η ομάδα ανάπτυξης έκρινε ότι θα ήταν εύχρηστο να δοθεί μεγαλύτερη ανάλυση στην επεξήγηση της λειτουργίας της κάθε συνάρτησης, ώστε να αποδειχθεί η ορθότητα του σχεδιασμού του συστήματος.

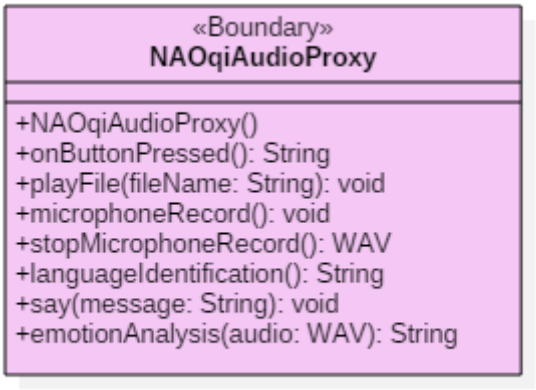
2.1 Πακέτα λεξιλογίου σεναρίων υψηλής προτεραιότητας

2.1.1 Πακέτο VoiceOrders

Το πακέτο αυτό περιλαμβάνει τη διαχείριση των φωνητικών εντολών των χρηστών από το σύστημα. Παρακάτω παρουσιάζονται οι κλάσεις που είναι υπεύθυνες για αυτές τις λειτουργίες.



Boundary NAOqiAudioProxy



Η κλάση αυτή εκφράζει τη διεπαφή του συστήματος με τα APIs του NAOqi OS που σχετίζονται με τον ήχο.

Μέθοδοι της κλάσης

- **NAOqiAudioProxy( )**

  Ο Δομητής (Constructor) της κλάσης NAOqiAudioProxy.

- **onButtonPressed( ): String**

  Η μέθοδος αυτή ενεργοποιείται με το πάτημα του κεντρικού κουμπιού στο κεφάλι του NAO και επιστρέφει ως String τη φωνητική εντολή που εκφωνήθηκε. Αν ο χρήστης δεν εκφωνήσει κάποια φωνητική εντολή, το String θα είναι NULL, ενώ αν εκφωνήσει κάτι που δεν κατάφερε να γίνει αντιληπτό από το NAOqi θα επιστρέψει το String “00”.

- **playFile(fileName: String): void**

  Η μέθοδος αυτή αναπαράγει το αρχείο .wav με τίτλο το δοθέν String.

- **microphoneRecord( ): void**

  Η μέθοδος αυτή ενεργοποιεί το μικρόφωνο του NAO σε περίπτωση που ανιχνευτεί ήχος στο μικρόφωνο και το σύστημα βρίσκεται σε κατάσταση αναμονής. Το σύστημα βρίσκεται πλέον στην διαδικασία της μετάφρασης, ο λόγος του ομιλητή ηχογραφείται και καλείται η συνάρτηση **TranslationController::setSpeaker( ): void**.

- **stopMirophoneRecord( ): WAV**

  Η μέθοδος αυτής απενεργοποιεί το μικρόφωνο του NAO και επιστρέφει ως αρχείο .wav την ηχογράφηση που μόλις τερματίστηκε.

- **languageIdentification( ): String**

  Η μέθοδος αυτή ενεργοποιείται για την αναγνώριση της γλώσσας των συμμετεχόντων και επιστρέφει ένα String με το όνομα αυτής. Αν ο χρήστης δεν εκφωνήσει κάποια γλώσσα, το String θα είναι NULL, ενώ αν εκφωνήσει κάτι που δεν κατάφερε να γίνει αντιληπτό από το NAOqi θα επιστρέψει το String “00”.

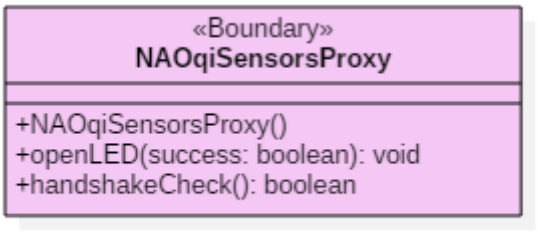
- **say(message: String): void**

  Η μέθοδος αυτή αναπαράγει φωνητικά το String που δίνεται ως είσοδος.

- **emotionAnalysis(audio: WAV): String**

  Η μέθοδος αυτή αναλύει συναισθηματικά το αρχείο ήχου που δέχεται ως όρισμα και επιστρέφει “Unknown”, “Calm”, “Anger”, “Joy” ή “Sorrow” ως χαρακτηρισμό για το ύφος του ομιλητή.

Boundary NAOqiSensorsProxy



Η κλάση αυτή εκφράζει τη διεπαφή του συστήματος με τα APIs του NAOqi OS που σχετίζονται με τους αισθητήρες και τα LEDs.

Μέθοδοι της κλάσης

- **NAOqiSensorsProxy( )**

  Ο Δομητής (Constructor) της κλάσης NAOqiSensorsProxy.

- **openLED(success: boolean): void**

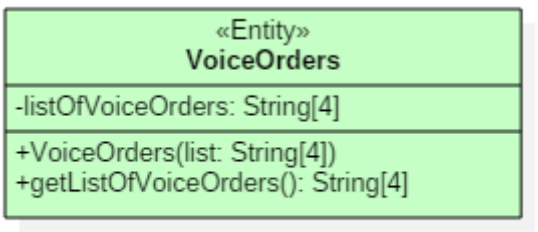
  Η μέθοδος αυτή ενεργοποιεί τα κατάλληλα LEDs στα μάτια του NAO. Πράσινα αν η παράμετρος είναι “true” και κόκκινα αν είναι “false”.

- **handshakeCheck(): boolean**

  Η μέθοδος αυτή ελέγχει αν ο χρήστης έχει κάνει χειραψία ή όχι.

  - Αν ναι, τότε επιστρέφει “true”.
  - Διαφορετικά, επιστρέφει “false”.

Entity VoiceOrders



Η κλάση αυτή περιέχει τη λίστα των φωνητικών εντολών.

Χαρακτηριστικά της κλάσης

- **listOfVoiceOrders: String[4]**

  Η λίστα των φωνητικών εντολών που αναγνωρίζονται από το σύστημα. Αυτές είναι οι "Translate", "Add", "Delete", "Terminate".

Μέθοδοι της κλάσης

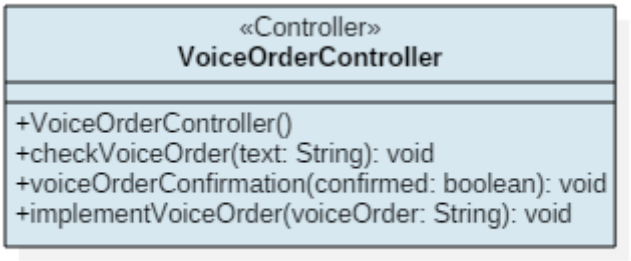
- **VoiceOrders (list: String[4])**

  Ο Δομητής (Constructor) της κλάσης VoiceOrders.

- **getListOfVoiceOrders( ): String[4]**

  Η μέθοδος αυτή επιστρέφει τη λίστα **VoiceOrders::listOfVoiceOrders: String[4]**.

Controller VoiceOrderController



Η κλάση αυτή αποτελεί έναν ελεγκτή ο οποίος διαχειρίζεται τις φωνητικές εντολές.

Μέθοδοι της κλάσης

- **VoiceOrderController( )**

  Ο Δομητής (Constructor) της κλάσης VoiceOrderController.

- **checkVoiceOrder(text: String): void**

  Η μέθοδος αυτή δέχεται ως όρισμα την έξοδο της μεθόδου **NAOqiAudioProxy:: onButtonPressed( ):String** και ελέγχει αν η εντολή που λήφθηκε ανήκει στη λίστα των φωνητικών. Αντίστοιχα, καλεί την **VoiceOrderController:: voiceOrderConfirmation(confirmed: boolean): void** για την επιβεβαίωση ή απόρριψη της εντολής μέσω οπτικοακουστικού μηνύματος. Σε περίπτωση επιβεβαίωσης καλεί την **VoiceOrderController:: implementVoiceOrder(voiceOrder: String): void** για την εκτέλεσή της.

- **voiceOrderConfirmation(confirmed: boolean): void**

  Η μέθοδος αυτή δηλώνει την επιβεβαίωση ή την απόρριψη μίας φωνητικής εντολής καλώντας τις μεθόδους :
  NAOqiAudioProxy::playFile(fileName: String): void

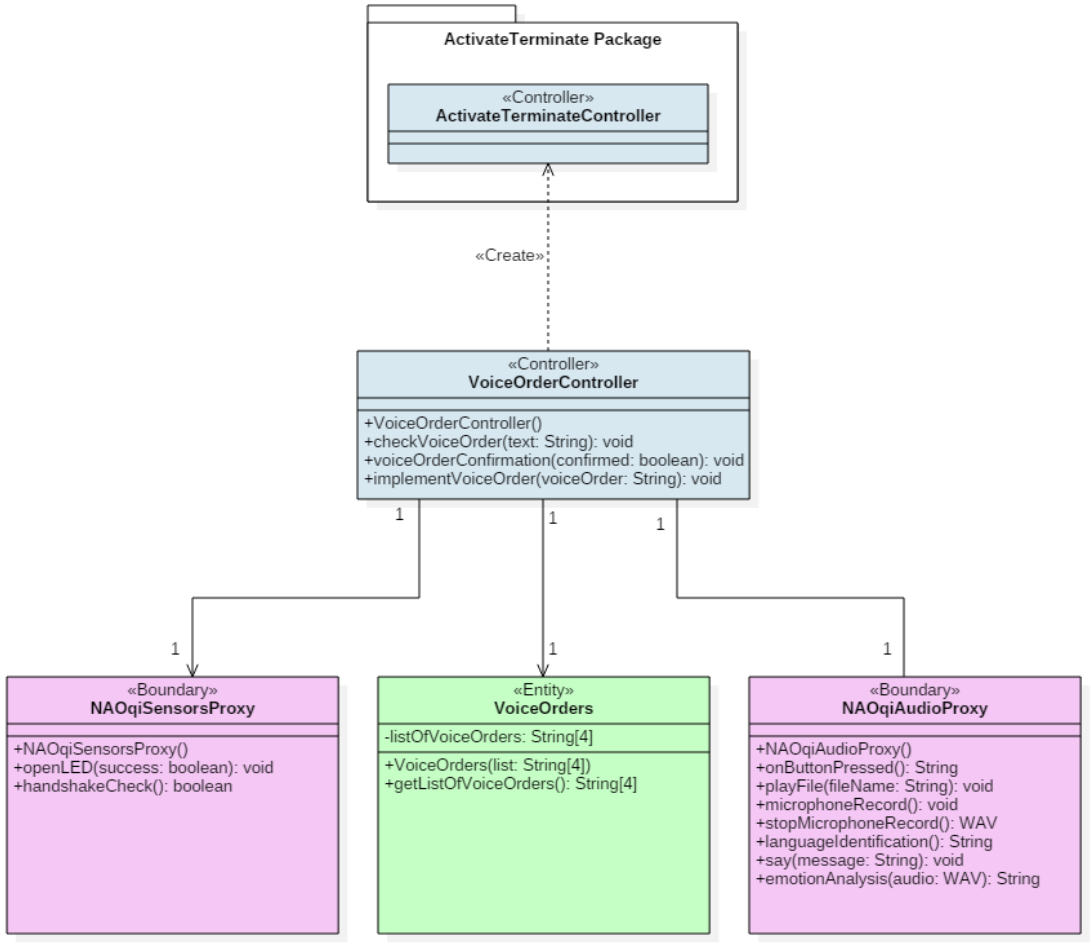
  NAOqiSensorsProxy::openLED(success: boolean): void.

- **implementVoiceOrder(voiceOrder: String): void**

  Η μέθοδος δημιουργεί τα αντικείμενα τύπου **Controller** που είναι υπεύθυνα για την εκτέλεση της εντολής που εισάγεται ως όρισμα τύπου String. Υπάρχουν οι ακόλουθες περιπτώσεις ανάλογα με την τιμή του String:

  - Translate: Δημιουργεί τον ActivateTerminateController και καλεί τη συνάρτηση **ActivateTerminateController::activation( ): void**.
  - Terminate : Καλεί τη συνάρτηση **ActivateTerminateController::termination( ):void**.
  - Add/Delete : Δημιουργεί τον AddDeleteController και καλεί τη συνάρτηση **AddDeleteController:: activity(String): void** με είσοδο το String 'Add'/'Delete'.

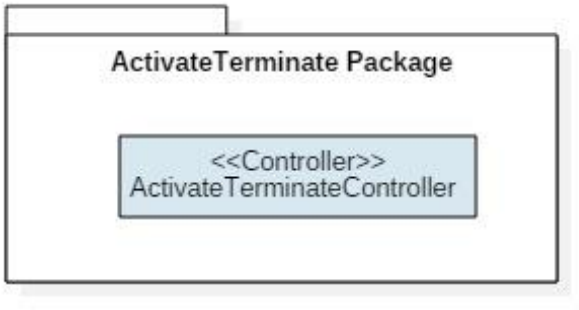
Διάγραμμα Κλάσεων



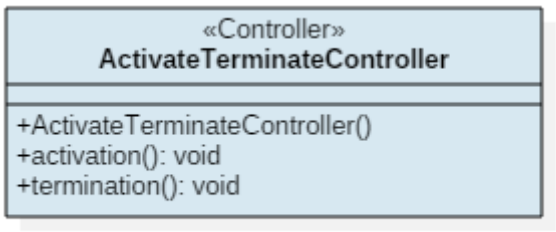
Σχήμα 6: Διάγραμμα Κλάσεων πακέτου VoiceOrders

2.1.2 Πακέτο ActivationTermination

Το πακέτο αυτό είναι αρμόδιο για την ενεργοποίηση και την απενεργοποίηση της εφαρμογής. Παρακάτω παρουσιάζεται η υπεύθυνη κλάση.



Controller ActivateTerminateController



Η κλάση αυτή είναι υπεύθυνη για την ενεργοποίηση και απενεργοποίηση του συστήματος.

Μέθοδοι της κλάσης

- **ActivateTerminateController()**

  Ο Δομητής (Constructor) της κλάσης ActivateTerminateController.

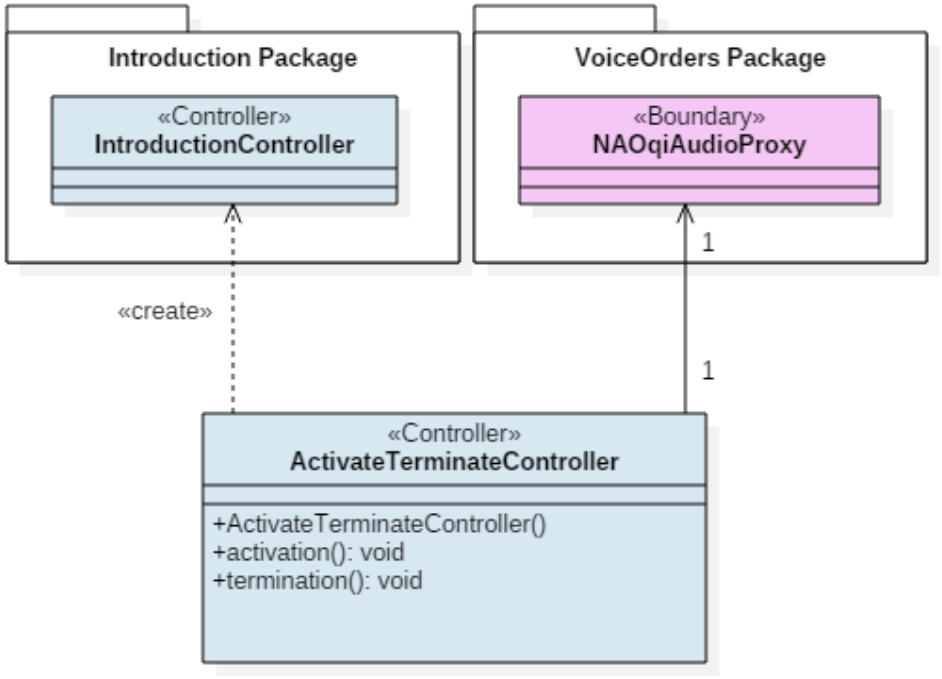
- **activation( ): void**

  Η μέθοδος αυτή ενεργοποιεί το σύστημα, δημιουργεί ένα αντικείμενο της κλάσης IntroductionController και καλεί τη συνάρτηση **IntroductionController:: welcomeAudioPlayer(): void**.

- **termination( ): void**

  Η μέθοδος αυτή καλεί τη συνάρτηση **NAOqiAudioProxy::say(message: String): void** με όρισμα το String ‘Bye-Bye’ και τερματίζει το σύστημα.

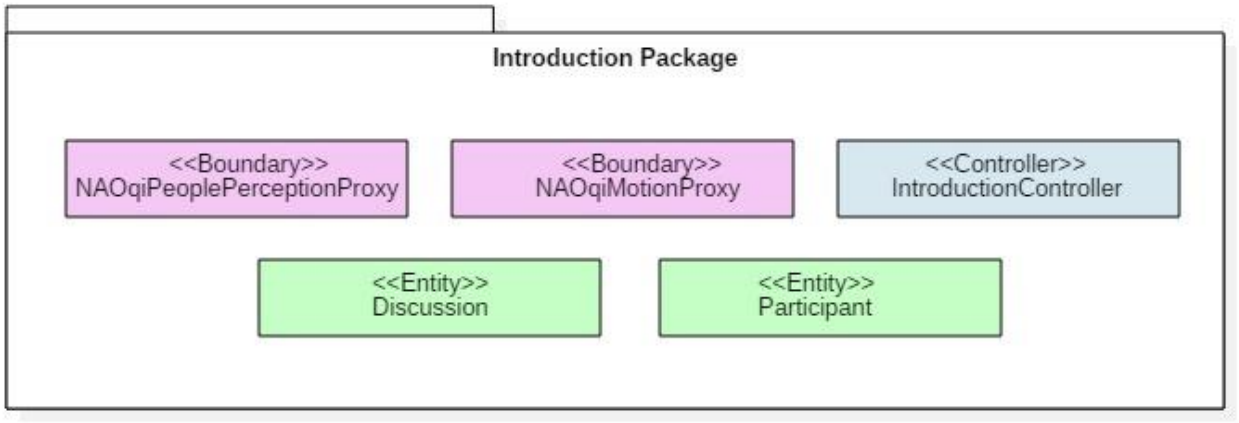
Διάγραμμα Κλάσεων



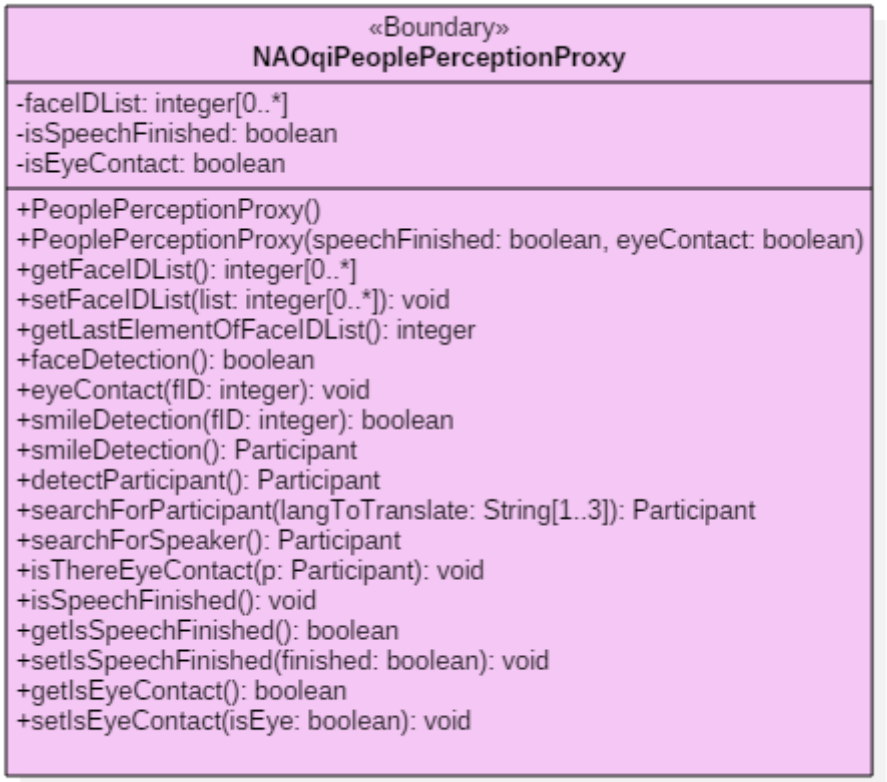
Σχήμα 7: Διάγραμμα Κλάσεων πακέτου ActivateTerminate

2.1.3 Πακέτο Introduction

Το πακέτο αυτό είναι αρμόδιο για την γνωριμία των συμμετεχόντων με το σύστημα. Παρακάτω παρουσιάζονται οι κλάσεις που είναι υπεύθυνες για αυτές τις λειτουργίες.



Boundary NAOqiPeoplePerceptionProxy



Η κλάση αυτή εκφράζει τη διεπαφή του συστήματος με τα APIs του NAOqi OS που σχετίζονται με τη δυνατότητα αντίληψης για την ύπαρξη ατόμων στο χώρο και την αλληλεπίδραση με αυτά.

Χαρακτηριστικά της κλάσης

- **faceIDList: integer[0..*]**

  Είναι μία λίστα που αποτελείται από ακέραιες μεταβλητές, στις οποίες αποθηκεύεται ένας χαρακτηριστικός ακέραιος, που προκύπτει ως το αποτέλεσμα μιας συνάρτησης face recognition και άρα χαρακτηρίζει μοναδικά το πρόσωπο του κάθε χρήστη.

- **isSpeechFinished: boolean**

  Λογική μεταβλητή που δηλώνει αν ο ομιλητής έχει τελειώσει το λόγο του.

- **isEyeContact: boolean**

  Λογική μεταβλητή που δηλώνει αν ο ομιλητής έχει κοιτάξει στα μάτια το NAO για πάνω από ένα δευτερόλεπτο.

Μέθοδοι της κλάσης

- **PeoplePerceptionProxy()**
  Ο κενός Δομητής (Constructor) της κλάσης NAOqiPeoplePerceptionProxy.

- **PeoplePerceptionProxy(speechFinished:boolean, eyeContact:boolean)**
  Ο Δομητής (Constructor) της κλάσης NAOqiPeoplePerceptionProxy με ορίσματα τις δύο από τις τρεις ιδιότητες της κλάσης.

- **getFaceIDList(): integer[0..*]**

  Η μέθοδος αυτή επιστρέφει τη λίστα **NAOqiPeoplePerceptionProxy::faceIDList: integer[0..*]**.

- **setFaceIDList():integer[0..*]**
  Η μέθοδος αυτή αναθέτει τιμές στη λίστα **NAOqiPeoplePerceptionProxy::faceIDList: integer[0..*]**.

- **getLastElementOfFaceIDList(): integer**

  Η μέθοδος αυτή επιστρέφει το τελευταίο στοιχείο της λίστας**NAOqiPeoplePerceptionProxy::faceIDList: integer[0..*]**.

- **faceDetection(): boolean**

  Η μέθοδος αυτή, αναζητά πρόσωπα τα οποία δεν έχουν αποθηκευτεί στη **NAOqiPeoplePerceptionProxy::faceIDList: integer[0..*]**.

  - Αν βρει ένα τέτοιο πρόσωπο, τότε το προσθέτει στην τελευταία θέση της λίστας και επιστρέφει λογική τιμή “true”.
  - Αν όλα τα πρόσωπα τα οποία βρίσκει, ψάχνοντας στο χώρο, έχουν ήδη καταγραφεί στη λίστα, επιστρέφει λογική τιμή “false”.

- **eyeContact(fID: integer): void**

  Η μέθοδος αυτή δέχεται ως είσοδο έναν ακέραιο, που δηλώνει τον χαρακτηριστικό αριθμό του προσώπου ενός χρήστη και ψάχνει αυτό το πρόσωπο στο χώρο. Μόλις το βρει, διατηρεί οπτική επαφή μαζί του.

- **getIsEyeContact(): boolean**
  Η μέθοδος αυτή επιστρέφει την τιμή της μεταβλητής **NAOqiPeoplePerceptionProxy::isEyeContact: boolean**.

- **setIsEyeContact(isEye: boolean): void**
  Η μέθοδος αυτή αναθέτει τιμή στην μεταβλητή **NAOqiPeoplePerceptionProxy::isEyeContact: boolean**.

- **isThereEyeContact(p: Participant): void**

  Η μέθοδος αυτή ενεργοποιείται όταν ο Participant που έχει δοθεί ως όρισμα κοιτάει στα μάτια το NAO για τουλάχιστον ένα δευτερόλεπτο. Όταν αυτό συμβεί, θέτει την τιμή της **NAOqiPeoplePerceptionProxy::isEyeContact: boolean** ίση με “true”.

- **smileDetection(fID: integer): boolean**

  Η μέθοδος αυτή δέχεται ως είσοδο έναν ακέραιο, που δηλώνει τον χαρακτηριστικό αριθμό του προσώπου ενός χρήστη και ψάχνει αυτό το πρόσωπο στο χώρο. Μόλις το βρει, ελέγχει αν χαμογελάει ή όχι.

  - Αν ναι, τότε επιστρέφει “true”.
  - Διαφορετικά, επιστρέφει “false”.

  Συνήθως η συνάρτηση αυτή καλείται μετά από την **NAOqiPeoplePerceptionProxy::eyeContact(fID: integer): void** και γι’ αυτό το λόγο ο χρόνος αναζήτησης του προσώπου ελαχιστοποιείται.

- **smileDetection(): Participant:**

  Η μέθοδος αυτή ψάχνει στο χώρο για να βρει έναν συμμετέχοντα που χαμογελάει.

  - Αν τον βρει, επιστρέφει το συμμετέχοντα.
  - Διαφορετικά, επιστρέφει NULL.

- **searchForParticipant(langToTranslate: String[1..3]): Participant**

  Ψάχνει να βρει έναν συμμετέχοντα που να ομιλεί μία από τις εναπομένουσες γλώσσες προς μετάφραση, οι οποίες δίνονται ως όρισμα και επιστρέφει αυτόν τον Participant.

- **detectParticipant(): Participant**

  Η μέθοδος αυτή επιστρέφει τον πρώτο συμμετέχοντα που εντοπίζεται στο χώρο.

- **searchForSpeaker(): Participant**

  Η μέθοδος αυτή ψάχνει να βρει τον ομιλητή, όσο αυτός μιλάει. Αυτό βασίζεται σε Sound Localization και Face Recognition τεχνικές.

  - Αν δεν καταφέρει να τον βρει, πριν αυτός τελειώσει το λόγο του, επιστρέφει NULL.
  - Διαφορετικά, επιστρέφει το αντικείμενο τύπου Participant που τον αντιπροσωπεύει.

- **isSpeechFinished(): void**

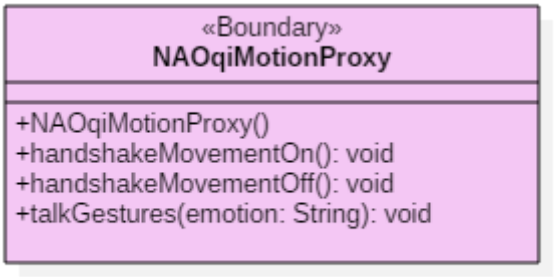
  Η μέθοδος αυτή ενεργοποιείται όταν ο ομιλητής σταματήσει να μιλάει για πάνω από 3 δευτερόλεπτα. Όταν αυτό συμβεί, θέτει την τιμή της **NAOqiPeoplePerceptionProxy::isSpeechFinished: boolean** ίση με “true”.

- **getIsSpeechFinished(): boolean**

  Η μέθοδος αυτή επιστρέφει την τιμή της μεταβλητής **NAOqiPeoplePerceptionProxy::isSpeechFinished: boolean**.

- **setIsSpeechFinished(finished:boolean): void**
  Η μέθοδος αυτή αναθέτει τιμή στην μεταβλητή **NAOqiPeoplePerceptionProxy::isSpeechFinished: boolean**.

Boundary NAOqiMotionProxy



Η κλάση αυτή εκφράζει τη διεπαφή του συστήματος με τα APIs του NAOqi OS που σχετίζονται με κινήσεις του NAO.

Μέθοδοι της κλάσης

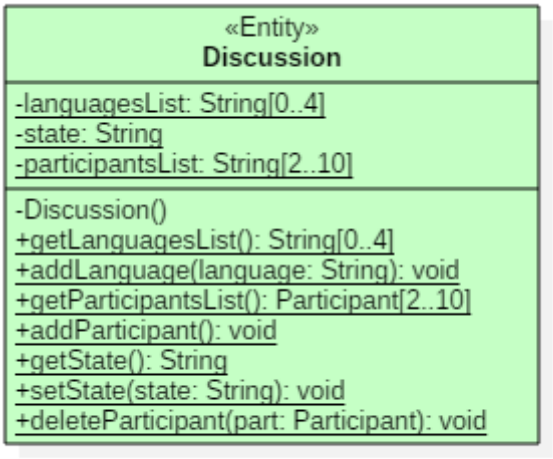
- **NAOqiMotionProxy()**
  Ο Δομητής (Constructor) της κλάσης NAOqiMotionProxy.

- **handshakeMovementOn(): void**
  Η μέθοδος αυτή μετακινεί τα ρομποτικά μέλη του NAO, προκειμένου να εκτελέσει κίνηση χειραψίας.

- **handshakeMovementOff(): void**
  Η μέθοδος αυτή μετακινεί τα ρομποτικά μέλη του NAO, ώστε να επανέλθει στη φυσική του θέση. Χρησιμοποιείται έπειτα από την **NAOqiMotionProxy::handshakeMovementOn(): void**.

- **talkGestures(emotion: String): void**
  Η μέθοδος αυτή δέχεται ως όρισμα ένα String, το οποίο είναι ο χαρακτηρισμός που προκύπτει από τη συναισθηματική ανάλυση των λεγομένων του ομιλητή και με βάση αυτό, εκτελεί τις κατάλληλες χειρονομίες για να συνοδεύσει την απαγγελία της μετάφρασης.

Entity Discussion



Η κλάση αυτή αποθηκεύει τα στοιχεία της συζήτησης. Ο Constructor της έχει οριστεί private και όλες οι συναρτήσεις και μεταβλητές της static, προκειμένου να μη μπορεί να δημιουργηθεί κανένα αντικείμενό της και να γίνεται καθολική αναφορά σε αυτήν, σε όλη τη διάρκεια του προγράμματος.

Χαρακτηριστικά της κλάσης

- **languagesList: String[0..4]**
  Λίστα όλων των διαφορετικών γλωσσών που έχουν δηλώσει οι συμμετέχοντες. Μπορεί να περιέχει μέχρι και 4 στοιχεία.

- **participantsList: Participant[2..10]**
  Λίστα που περιλαμβάνει έως 10 αντικείμενα τύπου Participant. Περιέχει τα στοιχεία όλων των συμμετεχόντων στη συζήτηση[1].

- **state: String**

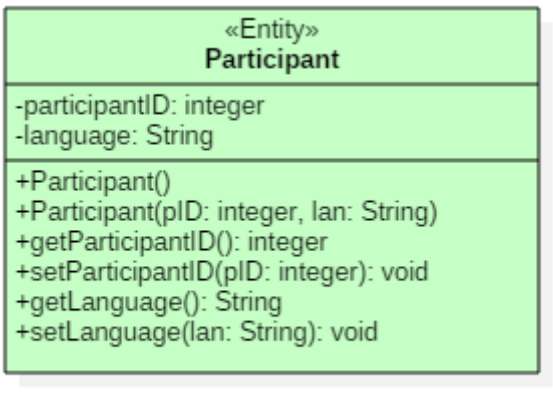
  Μεταβλητή που δηλώνει την κατάσταση στην οποία βρίσκεται το σύστημα. Οι διαθέσιμες καταστάσεις του συστήματος είναι “Stand by”, “Translate”, “Add”, “Delete” και “Introduction”.

[1] Η συγκεκριμένη ιδιότητα εμφανίζεται στο διάγραμμα κλάσεων ως συσχέτιση μεταξύ των κλάσεων Discussion και Participant

Μέθοδοι της κλάσης

- **Discussion()**
  Ο Δομητής (Constructor) της κλάσης Discussion.
- **getLanguagesList(): String[0..4]**
  Η μέθοδος αυτή επιστρέφει τη λίστα **Discussion::languagesList: String[0..4]**.
- **addLanguage(lan: String): void**
  Η μέθοδος αυτή προσθέτει ένα στοιχείο στη λίστα **Discussion::languagesList: String[0..4]**.
- **getParticipantsList():Participant[0..10]**
  Η μέθοδος αυτή επιστρέφει τη λίστα **Discussion::ParticipantsList: String[0..10]**.
- **addParticipant(p: Participant): void**
  Η μέθοδος αυτή προσθέτει ένα στοιχείο στη λίστα **Discussion::ParticipantsList: String[0..10]**.
- **deleteParticipant(p: Participant): void**
  Η μέθοδος αυτή δέχεται ως όρισμα τα στοιχεία ενός συμμετέχοντα. Με βάση αυτά τον αναζητεί στη λίστα, τον διαγράφει από αυτήν και τον διαγράφει και ως αντικείμενο.
- **getState(): String**
  Η μέθοδος αυτή επιστρέφει την τιμή της μεταβλητής **Discussion::state: String**.
- **setState(s: String): void**
  Η μέθοδος αυτή αναθέτει τιμή στη μεταβλητή **Discussion::state: String**.

Entity Participant



Η κλάση αυτή εμπεριέχει όλες τις ιδιότητες και τις μεθόδους που απαιτούνται για να οριστεί πλήρως η οντότητα των συμμετεχόντων.

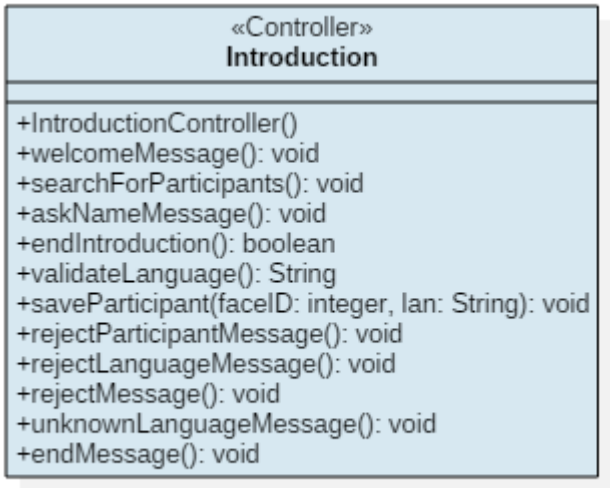
Χαρακτηριστικά της κλάσης

- **participantID: integer**
  Ο μοναδικός αναγνωριστικός αριθμός του συμμετέχοντα.
- **language: String**
  Η γλώσσα που έχει δηλώσει ο συμμετέχων.

Μέθοδοι της κλάσης

- **Participant()**
  Ο κενός Δομητής (Constructor) της κλάσης Participant.
- **Participant(pID: integer, lan: String)**
  Ο Δομητής με ορίσματα (Constructor) της κλάσης Participant.
- **setParticipantID(pID: integer): void**
  Η συνάρτηση αυτή θέτει τον μοναδικό αναγνωριστικό αριθμό του συμμετέχοντα.
- **setLanguage(lan: String): void**
  Η συνάρτηση αυτή αναθέτει ως γλώσσα του συμμετέχοντα τη συμβολοσειρά που λαμβάνει ως όρισμα από τη συνάρτηση **NAOqiAudioProxy::languageIdentification(): String**.
- **getParticipantID(): integer**
  Η μέθοδος αυτή επιστρέφει τον μοναδικό αναγνωριστικό αριθμό του συμμετέχοντα.
- **getLanguage(): String**
  Η συνάρτηση αυτή επιστρέφει τη γλώσσα του συμμετέχοντα.

Controller IntroductionController



Η κλάση αυτή είναι υπεύθυνη για τη γνωριμία του συστήματος με τους συμμετέχοντες στη συζήτηση και την αποθήκευση των στοιχείων τους.

Μέθοδοι της κλάσης

- **introductionController()**

  Ο Δομητής (Constructor) της κλάσης IntroductionController.

- **welcomeMessage(): void**

  Η μέθοδος αυτή καλεί την **NAOqiAudioProxy::say(String): void**, προκειμένου να αναπαράγεται μήνυμα το οποίο θα δίνεται ως όρισμα, για να καλοσωρίζει τους χρήστες και να τους ενημερώνει ότι όσοι επιθυμούν να συμμετάσχουν στη συζήτηση πρέπει να σταθούν σε ημικύκλιο γύρω από το NAO. Στη συνέχεια καλεί την **IntroductionController:: searchForParticipants(): void**.

- **endIntroduction(): boolean**

  Η μέθοδος αυτή ελέγχει αν θα τερματιστεί η διαδικασία της γνωριμίας. Καλεί την **NAOqiPeoplePerceptionProxy::faceDetection(): boolean** και ελέγχει αν επιστρέφει false και την πληρότητα της **Discussion::participantsList: Participant[0..10]** (αν υπάρχουν ήδη 10 συμμετέχοντες). Σε αυτές τις περιπτώσεις ενημερώνει τον χρήστη με αντίστοιχο μήνυμα (καλώντας τις **IntroductionController::rejectParticipant(): void**, **IntroductionController::endMessage(): void** κατάλληλα), επιστρέφει τιμή true και μεταβάλλει το **Discussion::state: String** σε “Stand by”.

- **askNameMessage(): void**

  Η μέθοδος αυτή καλεί τη **NAOqiAudioProxy::say(String): void**, προκειμένου να αναπαράγεται μήνυμα που θα συστήνεται ο NAO και θα ρωτάει το όνομα του χρήστη.

- **validateLanguage(): String**

  Η μέθοδος αυτή θέλει να επιστρέψει τη γλώσσα που έχει δηλώσει ο χρήστης. Καλεί την **NAOqiAudioProxy::say(String): void**, προκειμένου να αναπαράγεται μήνυμα που θα ρωτάει τη γλώσσα του χρήστη. Στη συνέχεια, καλεί τη συνάρτηση **NAOqiAudioProxy ::languageIdentification(): String** και εξετάζει το επιστρεφόμενο String.

  - Αν η γλώσσα δεν έχει γίνει αντιληπτή, καλείται η **IntroductionController::unknownLanguageMessage(): void** και η **NAOqiAudioProxy ::languageIdentification(): String** ξανά.
  - Αν ο χρήστης δεν έχει δηλώσει γλώσσα, καλείται η **IntroductionController::rejectMessage(): void** και η συνάρτηση επιστρέφει τιμή NULL.
  - Διαφορετικά, ελέγχεται αν η λίστα **Discussion::languagesList: String[0..4]** είναι πλήρης (περιέχει ήδη 4 διαφορετικά στοιχεία) και η γλώσσα που δηλώθηκε δεν ανήκει σε αυτά.
    - Σε περίπτωση που ισχύει, καλείται η **IntroductionController:: rejectLanguageMessage(): void** και η συνάρτηση επιστρέφει τιμή NULL.
    - Διαφορετικά, στη συνήθη περίπτωση, η συνάρτηση επιστρέφει τη γλώσσα ως συμβολοσειρά.

- **savеParticipant(faceID: integer, lan: String): void**

  Η μέθοδος αυτή δέχεται ως ορίσματα τον χαρακτηριστικό αριθμό κάθε χρήστη (με κλήση της συνάρτησης **NAOqiPeoplePerceptionProxy::getLastElementOfFaceIDList(): integer**) και τη γλώσσα που έχει δηλώσει και δημιουργεί αντικείμενα τύπου Participant, καλώντας την **Participant::Participant(pID: integer, lan: String): void**. Τα αντικείμενα αυτά αποθηκεύονται στην **Discussion::participantsList: Participant[0..10],** ενώ παράλληλα ενημερώνεται και η **Discussion::languagesList: String[0..4]**.

- **rejectParticipantMessage(): void**

  Η μέθοδος αυτή καλεί την **NAOqiAudioProxy::say(s: String): void** για να αναπαράγεται μήνυμα που θα ενημερώνει τον χρήστη πως ο αριθμός συμμετεχόντων στη συζήτηση είναι ο μέγιστος (10 συμμετέχοντες) και δε μπορεί να εισαχθεί σε αυτή. Επίσης, ενεργοποιεί τα κόκκινα LEDS του NAO, καλώντας την **NAOqiSensorsProxy::openLED(led: boolean): void**.

- **rejectLanguageMessage(): void**

  Η μέθοδος αυτή καλεί την **NAOqiAudioProxy::say(s: String): void**, για να αναπαράγεται μήνυμα που θα ενημερώνει τον χρήστη πως ο αριθμός γλωσσών στη συζήτηση είναι ο μέγιστος (4 γλώσσες) και επομένως δε γίνεται δεκτός σε αυτή. Επίσης, ενεργοποιεί τα κόκκινα LEDS του NAO καλώντας την **NAOqiSensorsLEDsProxy::openLED(led: boolean): void**.

- **rejectMessage(): void**

  Η μέθοδος αυτή καλεί την **NAOqiAudioProxy::say(s: String): void**, για να αναπαράγεται μήνυμα που θα ενημερώνει τον χρήστη ότι δεν εισήχθη στη συζήτηση, σε περίπτωση που ο χρήστης δεν ανταποκριθεί στη χειραψία ή δεν πει τη γλώσσα του εντός 3 δευτερολέπτων. Επίσης, ενεργοποιεί τα κόκκινα LEDS του NAO καλώντας την **NAOqiSensorsProxy::openLED(led: boolean): void**.

- **unknownLanguageMessage(): void**

  Η μέθοδος αυτή καλεί την **NAOqiAudioProxy::say(s: String): void**, για να αναπαράγεται μήνυμα που θα ενημερώνει τον χρήστη ότι η γλώσσα που είπε δεν έγινε κατανοητή και θα πρέπει να την επαναλάβει. Επίσης, ενεργοποιεί τα κόκκινα LEDS του NAO, καλώντας την **NAOqiSensorsProxy::openLED(led: boolean): void**.

- **searchForParticipants(): void**

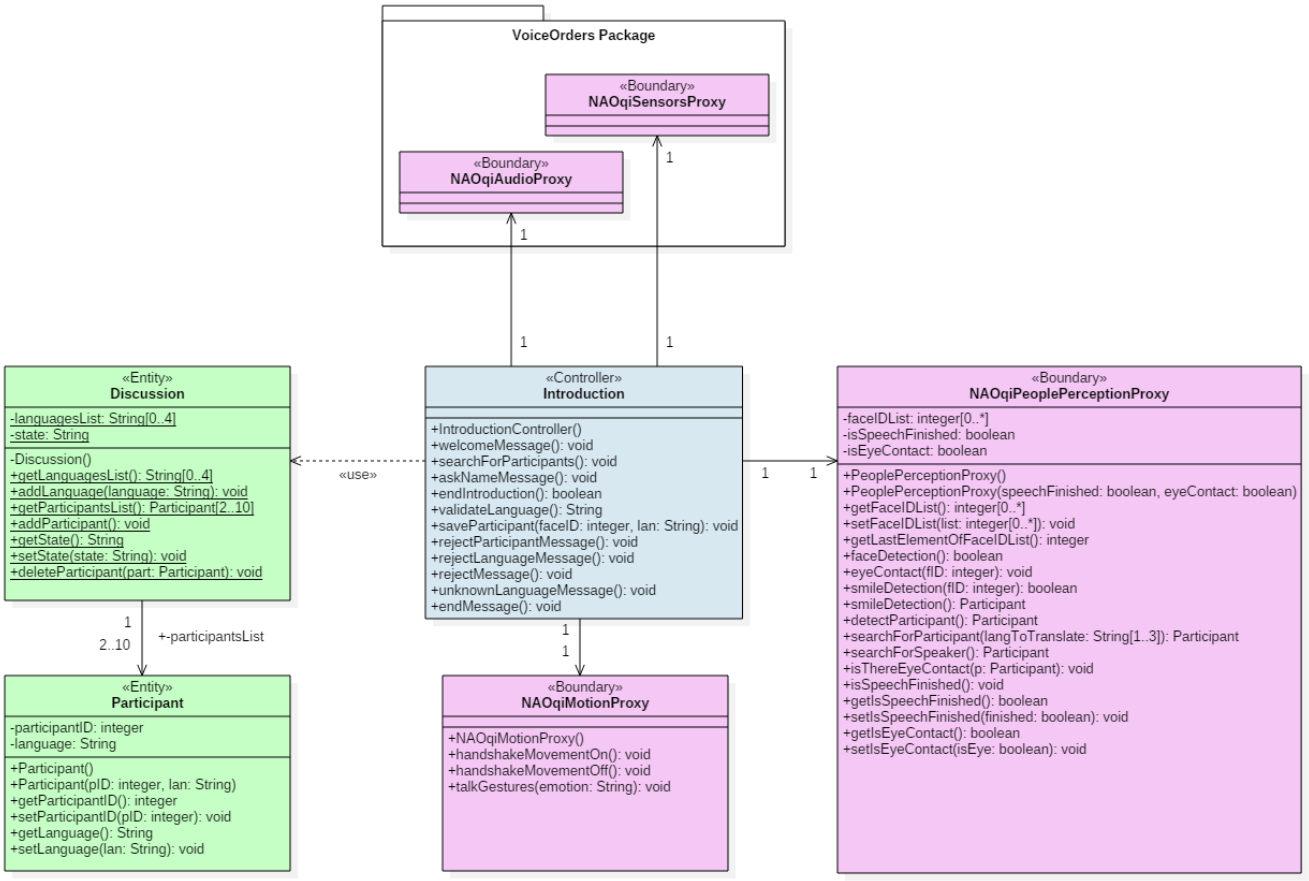
  Η σημαντικότερη μέθοδος της κλάσης, η οποία είναι αρμόδια για τη διαδικασία της γνωριμίας. Αποτελείται από μία επαναληπτική διαδικασία, της οποίας συνθήκη είναι η **IntroductionController::endIntroduction(): boolean**. Όσο αυτή είναι false, καλούνται οι:
  NAOqiPeoplePerceptionProxy::eyeContact(fID: integer): void
  NAOqiPeoplePerceptionProxy::smileDetection(fID: integer): boolean.
  Αν η τελευταία επιστρέψει τιμή true, καλούνται οι:
  IntroductionControler::askNameMessage(): void,
  NAOqiMotionProxy::handshakeMovementOn(): void και
  NAOqiMotionProxy::handshakeMovementOff(): void.
  Μεταξύ αυτών παρεμβάλλεται συνθήκη που εξαρτάται από την τιμή της
  NAOqiSensorProxy::handshakeCheck(): boolean.

Αν είναι αληθής, καλείται η **IntroductionController::validateLanguage():String**. Σε περίπτωση που είναι αποδεκτή τιμή γλώσσας το επιστρεφόμενο String, γίνεται κλήση της **IntroductionController::saveParticipant(faceID:integer, lan:String)**. Η διαδικασία αυτή συνεχίζεται μέχρις ότου η **IntroductionController::endIntroduction(): boolean** γίνει true.

- **endMessage():void**

  Η μέθοδος αυτή καλεί την **NAOqiAudioProxy::say(s: String): void**, για να αναπαράγεται μήνυμα που θα ενημερώνει τον χρήστη για το τέλος της διαδικασίας γνωριμίας και για την έναρξη της συζήτησης.

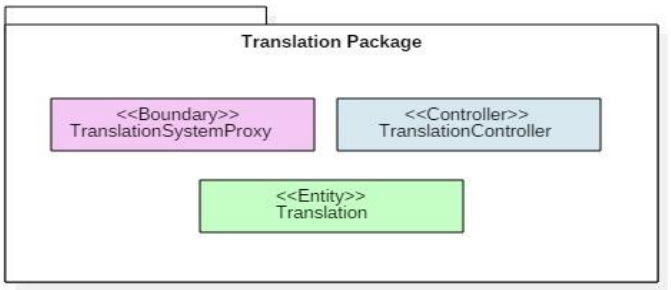
Διάγραμμα Κλάσεων



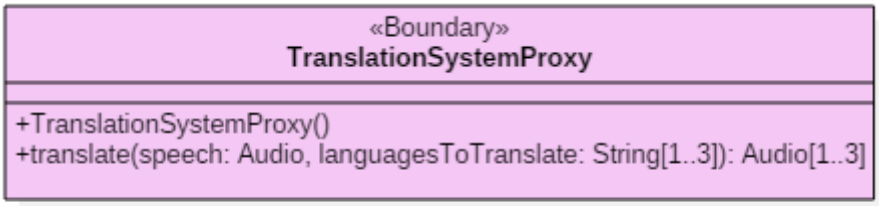
Σχήμα 8: Διάγραμμα Κλάσεων πακέτου Introduction

2.1.4 Πακέτο Translation

Το πακέτο αυτό είναι αρμόδιο για τη μετάφραση των λεγομένων του ομιλητή στους ακροατές. Παρακάτω παρουσιάζονται οι κλάσεις που είναι υπεύθυνες για αυτή τη λειτουργία.



Boundary TranslationSystemProxy



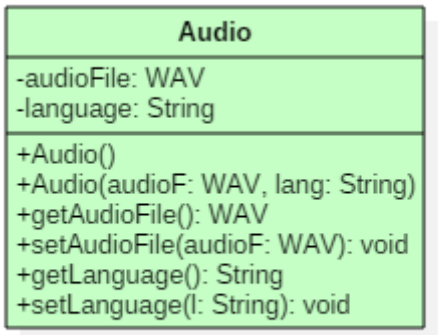
Η κλάση αυτή εκφράζει τη διεπαφή του συστήματος με το εξωτερικό σύστημα μετάφρασης.

Μέθοδοι της κλάσης

- **TranslationSystemProxy()**
  Ο Δομητής (Constructor) της κλάσης TranslationSystemProxy.

- **translate(speech: Audio, languagesToTranslate:String[1..3]): Audio[1..3]**
  Η μέθοδος αυτή δέχεται ως ορίσματα το λόγο του ομιλητή και τις γλώσσες στις οποίες πρέπει να επιστρέψει τη μετάφραση και επιστρέφει έναν πίνακα με αρχεία τύπου Audio που αποτελούν τις μεταφράσεις στις απαραίτητες γλώσσες.

Entity Audio



Η κλάση αυτή χρησιμεύει κυρίως ως μιας μορφής Data Type. Κάθε μεταβλητή αυτού του τύπου έχει αποθηκευμένο ένα αρχείο ήχου, σε συνδυασμό με ένα tag που αφορά στη γλώσσα που το περιεχόμενο του αρχείου έχει διατυπωθεί.

Χαρακτηριστικά της κλάσης

- **audioFIle: WAV**

  Το αρχείο ήχου.
- **language: String**

  Η γλώσσα στην οποία έχει διατυπωθεί το περιεχόμενο του αρχείου ήχου.

Μέθοδοι της κλάσης

- **Audio()**

  Ο κενός Δομητής (Constructor) της κλάσης Audio.
- **Audio (audioF: WAV, lan: String)**

  Ο Δομητής με ορίσματα (Constructor) της κλάσης Audio.
- **getAudioFile (): WAV**

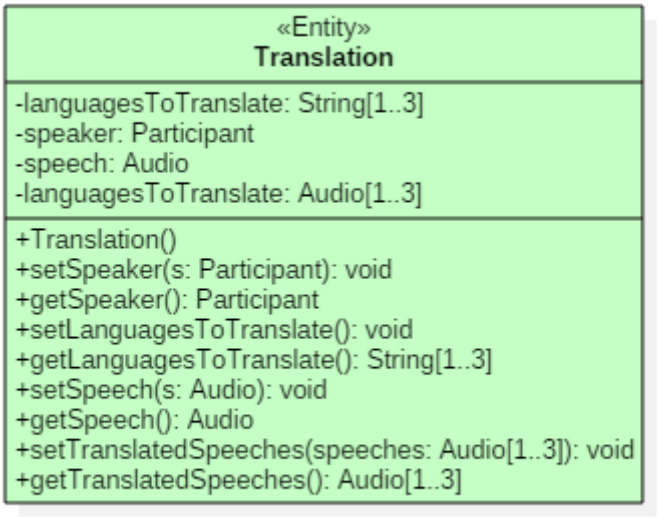
  Η μέθοδος αυτή επιστρέφει το αρχείο ήχου του αντικειμένου Audio.
- **setAudioFile(audioF: WAV): void**

  Η μέθοδος αυτή θέτει το αρχείο ήχου της κλάσης Audio να είναι το αρχείο audioF.
- **getLanguage(): String**

  Η μέθοδος αυτή επιστρέφει τη γλώσσα του αντικειμένου Audio.
- **setLanguage(l: String): void**

  Η μέθοδος αυτή θέτει τη γλώσσα του αντικειμένου Audio να είναι αυτή που δίνεται ως όρισμα.

Entity Translation



Η κλάση αυτή περιέχει τις πληροφορίες που είναι απαραίτητες για τη μετάφραση των λεγομένων του ομιλητή στους ακροατές.

Χαρακτηριστικά της κλάσης

- **speaker: Participant**
  Ο ομιλητής στην παρούσα διαδικασία μετάφρασης[2].
- **languagesToTranslate: String[1..3]**
  Οι γλώσσες στις οποίες το σύστημα θα κληθεί να μεταφράσει.
- **speech: Audio**
  Ο λόγος του ομιλητή αποθηκευμένος μαζί με τη γλώσσα του ως ετικέτα[3].
- **translatedSpeeches: Audio[1..3]**
  Οι μεταφράσεις του λόγου του ομιλητή, σε όλες τις γλώσσες που υπάρχουν στη συζήτηση[4].

Μέθοδοι της κλάσης

- **Translation()**

  Ο Δομητής (Constructor) της κλάσης Translation.
- **setSpeaker(s: Participant): void**
  Η μέθοδος αυτή δέχεται ως όρισμα έναν συμμετέχοντα και τον τοποθετεί στη μεταβλητή **Translation::speaker: Participant**.

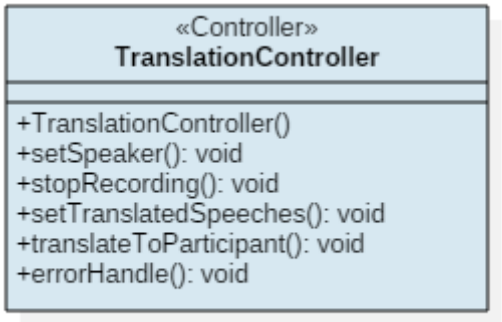
[2] Στο διάγραμμα εμφανίζεται ως συσχέτιση.

[3] Στο διάγραμμα εμφανίζεται ως συσχέτιση.

[4] Στο διάγραμμα εμφανίζεται ως συσχέτιση.

- **getSpeaker(): Participant**
  Η μέθοδος αυτή επιστρέφει την τιμή της μεταβλητής **Translation::speaker: Participant**.
- **setLanguagesToTranslate(): void**
  Η μέθοδος αυτή ορίζει τις γλώσσες στις οποίες πρέπει να γίνει η παρούσα μετάφραση. Ως τέτοιες, ορίζονται οι γλώσσες της συζήτησης που δεν είναι γλώσσες του ομιλητή.
- **getLanguagesToTranslate(): String[1..3]**
  Η μέθοδος αυτή επιστρέφει τις γλώσσες στις οποίες πρέπει να γίνει η μετάφραση.
- **setSpeech(s: Audio): void**
  Η μέθοδος αυτή δέχεται ως όρισμα ένα αντικείμενο τύπου Audio και το αναθέτει στη μεταβλητή speech.
- **getSpeech(): Audio**
  Η μέθοδος αυτή επιστρέφει ένα αντίγραφο του αντικειμένου **Translation::speech: Participant**.
- **setTranslatedSpeeches(speeches: Audio[1..3]): void**
  Η μέθοδος αυτή, αναθέτει στα στοιχεία του πίνακα **Translation::translatedSpeeches:Audio[1..3]** στις τιμές που υπάρχουν στις αντίστοιχες θέσεις του πίνακα που δέχεται ως όρισμα.
- **getTranslatedSpeeches(): Audio[1..3]**
  Η μέθοδος αυτή, επιστρέφει ένα αντίγραφο του πίνακα **Translation::translatedSpeeches:Audio[1..3]**.

Controller TranslationController



Η κλάση αυτή είναι υπεύθυνη για τη μετάφραση των λεγομένων του ομιλητή στους ακροατές.

Μέθοδοι της κλάσης:

- **TranslationController()**
  Ο Δομητής (Constructor) της κλάσης TranslationController.

- **setSpeaker(): void**
  Η μέθοδος αυτή έχει ως σκοπό την εύρεση του ομιλητή και την αποθήκευσή του στο **Translation Entity**. Αρχικά, δημιουργεί ένα αντικείμενο τύπου **NAOqiPeoplePerceptionProxy:: NAOqiPeoplePerceptionProxy(speechFinished: boolean, eyeContact: boolean)** με ορίσματα "false" και καλεί τη συνάρτηση **NAOqiPeoplePerceptionProxy::searchForSpeaker(): Participant**.
  - Αν αυτή επιστρέψει NULL, τότε ο ομιλητής δεν έχει βρεθεί πριν ολοκληρωθεί ο λόγος του και καλείται η **TranslationController::errorHandle(): void**, για να ταυτοποιήσει τον ομιλητή.
  - Διαφορετικά, καλείται η **Translation::setSpeaker(p: Participant): void**, προκειμένου να αποθηκευτεί ο ομιλητής. Μόλις κάποια από τις **NAOqiPeoplePerceptionProxy:: isSpeechFinished: boolean**, **NAOqiPeoplePerceptionProxy::isEyeContact: boolean** γίνει "true", καλείται η **TranslationController::stopRecording(): void**.

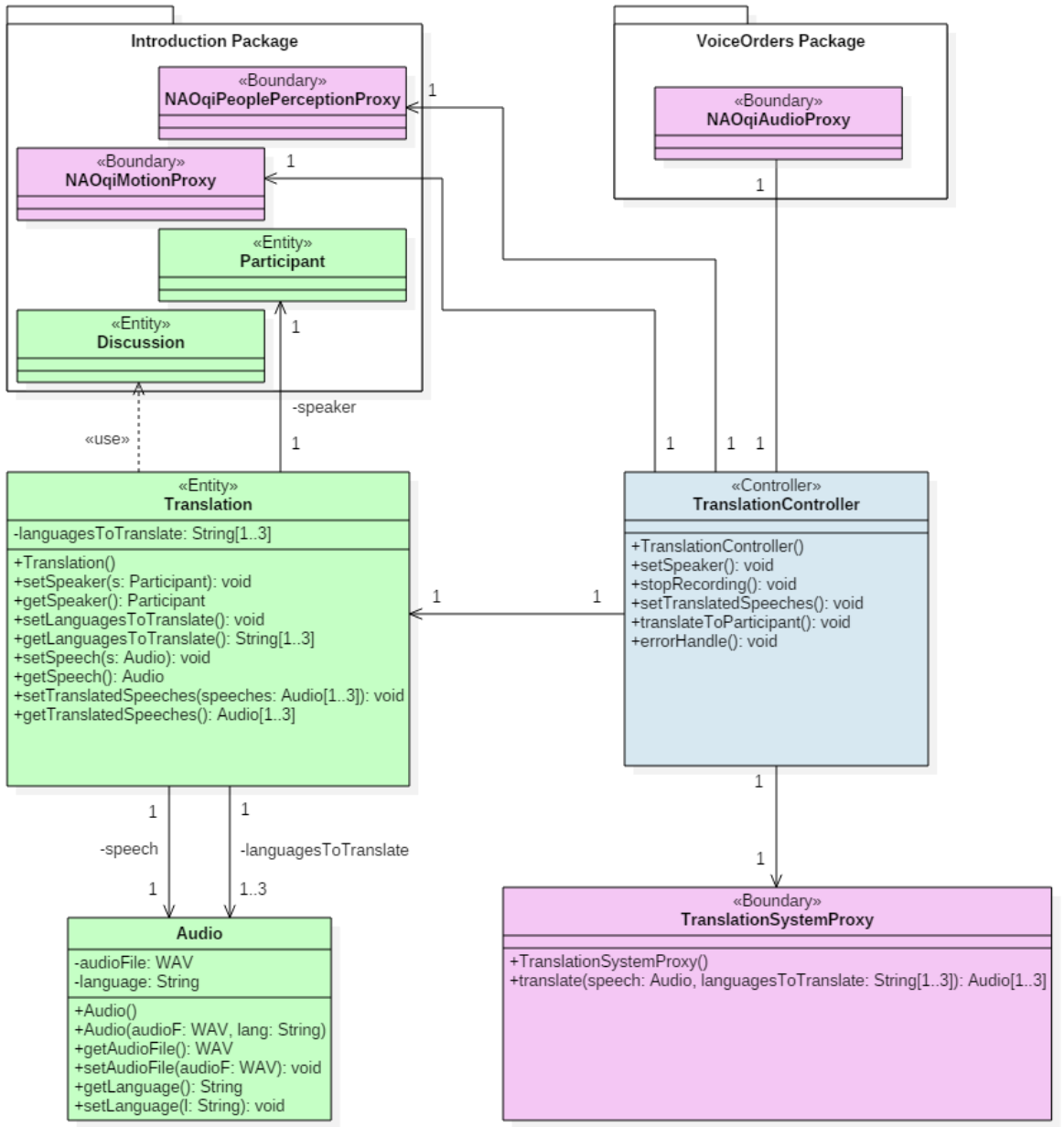
- **stopRecording(): void**
  Η μέθοδος αυτή καλεί την **NAOqiAudioProxy::stopMicrophoneRecord(): WAV**, για να τερματιστεί η ηχογράφηση και στη συνέχεια, καλεί την **Translation::setSpeech(s: Audio): void**, ώστε να αποθηκευτεί ο λόγος του ομιλητή. Τέλος, καλεί την **TranslationController:: setTranslatedSpeeches(speeches: Audio[1..3]): void**.

- **setTranslatedSpeeches(): void**
  Η μέθοδος αυτή καλεί την **TranslationSystemProxy::translate(speech: Audio, languagesToTranslate: String[1..3]): Audio[1..3]**, για να λάβει τα μεταφρασμένα αρχεία από το εξωτερικό σύστημα μετάφρασης. Στη συνέχεια, καλεί την **Translation:: setTranslatedSpeeches(Audio[1..3]): void**, ώστε να αποθηκευτούν τα μεταφρασμένα αρχεία, ενώ τέλος, την **Translation::translateToParticipant(): void** για να ξεκινήσει η μετάφραση.

- **translateToParticipant(): void**
  Η μέθοδος αυτή, καλεί την **NAOqiPeoplePerceptionProxy:: searchForParticipant(languagesToTranslate:String[1..3]): Participant** για να βρει έναν συμμετέχοντα. Στη συνέχεια, την **NAOqiPeoplePerceptionProxy::eyeContact(faceID: integer): void**, ώστε να διατηρήσει οπτική επαφή με το συμμετέχοντα που εντόπισε. Τέλος, χρησιμοποιώντας τις **NAOqiAudioProxy::emotionAnalysis(audio: WAV): String, NAOqiMotionProxy::talkGestures(emotion: String): void και Translation:: getTranslatedSpeeches():Audio[1..3]** μεταφράζει το λόγο του ομιλητή στη γλώσσα του συμμετέχοντα. Η διαδικασία επαναλαμβάνεται, μέχρις ότου γίνει μετάφραση σε όλες τις γλώσσες του **Translation::languagesToTranslate: String[1..3]**.

- **errorHandle(): void**
  Η μέθοδος αυτή, καλεί την **NAOqiAudioProxy::stopMicrophoneRecord():WAV** για να σταματήσει την καταγραφή λόγου. Στη συνέχεια την **NAOqiPeoplePerceptionProxy::smileDetection():Participant** για να εντοπίσει τον ομιλητή, καθώς και τις **Translation::setSpeaker(p: Participant):void** και **Translation::setSpeech(s: Audio):void**, ώστε να αποθηκευτεί ο ομιλητής και η γλώσσα του. Τέλος, καλεί την **Translation::setTranslatedSpeeches():void** ώστε να συνεχιστεί η διαδικασία της μετάφρασης, όπως εξηγήθηκε παραπάνω.

Διάγραμμα Κλάσεων

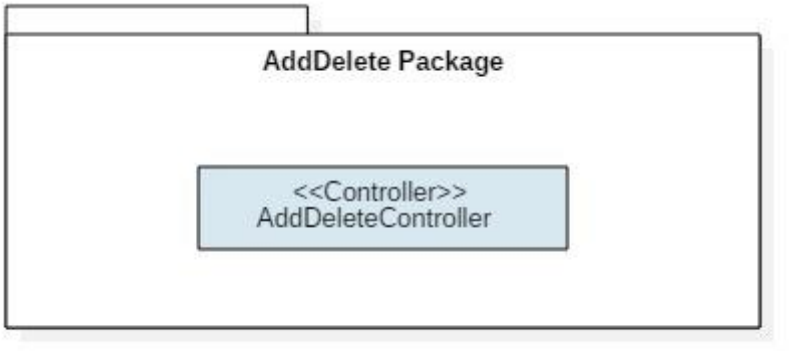


Σχήμα 9: Διάγραμμα κλάσεων πακέτου Translation

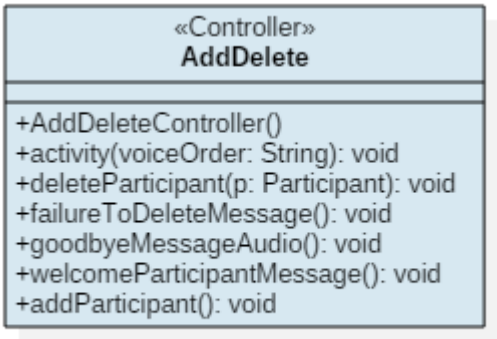
2.2 Πακέτο λεξιλογίου σεναρίων μέσης προτεραιότητας

2.2.1 Πακέτο AddDelete

Το πακέτο αυτό είναι υπεύθυνο για την προσθήκη και την διαγραφή συμμετεχόντων στην εφαρμογή. Παρακάτω παρουσιάζεται η κλάση που είναι υπεύθυνη για αυτές τις λειτουργίες.



Controller AddDelete



Η κλάση αυτή εκτελεί διπλή λειτουργία, εκείνη της διαγραφής ενός συμμετέχοντα κι εκείνη της εισαγωγής ενός νέου μετά το πέρας της διαδικασίας της γνωριμίας.

Μέθοδοι της κλάσης

- **addDeleteController(): void**
  Ο Δομητής (Constructor) της κλάσης AddDeleteController.

- **activity(voiceOrder:String): void**
  Η μέθοδος αυτή δέχεται ως όρισμα τη φωνητική εντολή που έχει δώσει ο χρήστης. Σε περίπτωση που εκείνη είναι η “Delete”, καλείται η συνάρτηση **AddDeleteController::delete(p: Participant): void**, στην οποία δίνεται ως όρισμα η **NAOqiPeoplePercep-**

tionProxy::detectParticipant(): Participant. Αντίστοιχα αν πρόκειται για τη φωνητική εντολή "Add", καλείται η συνάρτηση **AddDeleteController::addParticipant(): void**.

➢ **delete(p: Participant): void**

Η μέθοδος έχει ως όρισμα τον προς διαγραφή συμμετέχοντα, ο οποίος δίνεται από την **NAOqiPeoplePerceptionProxy::detectParticipant(): Participant**. Αρχικά, καλούνται οι **NAOqiPeoplePerceptionProxy::eyeContact(fID: Integer): void** και **NAOqiPeoplePerceptionProxy::smileDetection(fID: Integer): boolean**, με όρισμα το participantID του συμμετέχοντα προς διαγραφή.

- Αν η **NAOqiPeoplePerceptionProxy::smileDetection(): boolean** επιστρέψει ψευδή τιμή, καλείται η **AddDeleteController::failureToDeleteMessage(): void**.
- Διαφορετικά, καλούνται οι **Discussion:deleteParticipant(Participant): void** και **AddDeleteController::goodbyeMessage(): void**, για να πραγματοποιηθεί η διαγραφή.

Η διαδικασία ολοκληρώνεται και μεταβάλλεται το **Discussion::state: String** σε "Stand by" (καλώντας την **Discussion::setState(s: String): void**).

➢ **failureToDeleteMessage(): void**

Η μέθοδος αυτή καλεί την **NAOqiAudioProxy::say(s: String): void**, προκειμένου να αναπαράγεται μήνυμα το οποίο θα δίνεται ως όρισμα, και να ενημερώσει τον προς διαγραφή συμμετέχοντα για την αποτυχία της διαγραφής του. Επίσης, ενεργοποιεί τα κόκκινα LEDS του NAO καλώντας την **NAOqiSensorsProxy::openLED(led: boolean): void**.

➢ **goodbyeMessage(): void**

Η μέθοδος αυτή καλεί την **NAOqiAudioProxy::say(s: String): void**, προκειμένου να αναπαράγεται μήνυμα το οποίο θα δίνεται ως όρισμα, για να αποχαιρετήσει τον προς διαγραφή συμμετέχοντα και τερματίζει την διαδικασία.

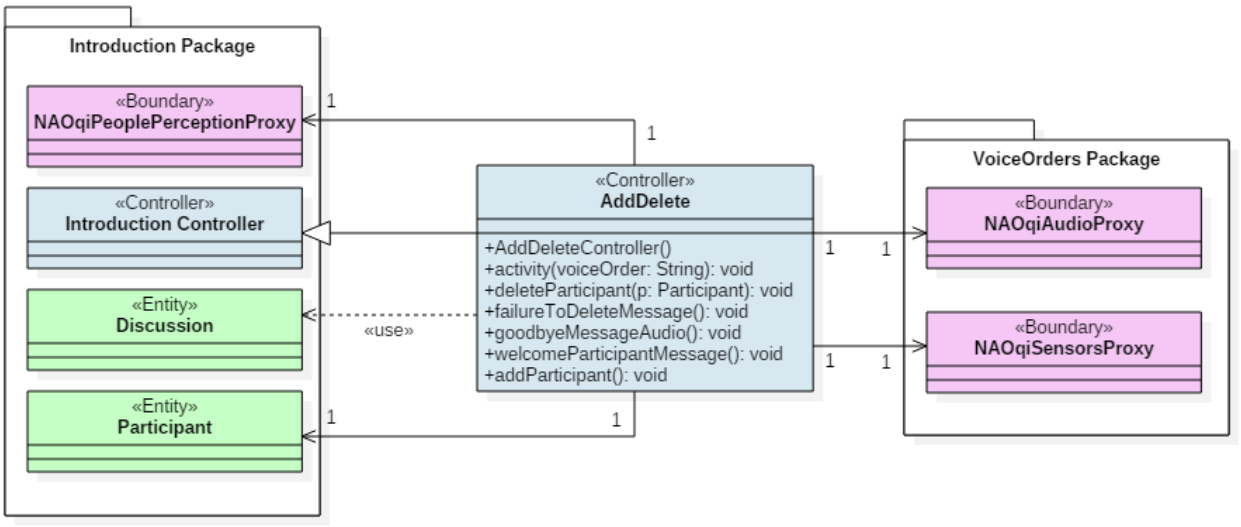
➢ **welcomeParticipantMessage(): void**

Η μέθοδος αυτή καλεί την **NAOqiAudioProxy::say(s: String): void**, προκειμένου να αναπαράγεται μήνυμα το οποίο θα δίνεται ως όρισμα, για να ενημερώνει τον χρήστη ότι έχει εισαχθεί στη συζήτηση.

➢ **addParticipant(): void**

Η μέθοδος πρέπει να ελέγχει η **Discussion::participantsList: Participant[0..10]** είναι πλήρης. Αν η προσθήκη είναι επιτρεπτή, καλείται η **NAOqiPeoplePerceptionProxy::faceDetection(): boolean** και αφού ανιχνεύσει τον προς αποθήκευση συμμετέχοντα καλεί τις **NAOqiPeoplePerceptionProxy::eyeContact(fID: Integer): void** και **NAOqiPeoplePerceptionProxy::smileDetection(fID: Integer): boolean**. Αν αυτή επιστρέψει true, καλούνται οι **IntroductionController::askNameMessage(): void** και **IntroductionController::validateLanguage(): String**. Σε περίπτωση που είναι αποδεκτή τιμή γλώσσας το επιστρεφόμενο String, γίνεται κλήση των **IntroductionController::saveParticipant(faceID:integer, lan:String) AddDeleteController::welcomeParticipantMessage(): void**. Σε κάθε περίπτωση αποτυχίας, πρέπει να ενημερώνεται ο χρήστης καλώντας την ανάλογη συνάρτηση. Η διαδικασία ολοκληρώνεται και μεταβάλλεται το **Discussion::state: String** σε "Stand by".

Διάγραμμα Κλάσεων



Σχήμα 10: Διάγραμμα Κλάσεων πακέτου AddDelete

3. Μη λειτουργικές απαιτήσεις

3.1 Απαιτήσεις επίδοσης

<ΜΛΑ- 2>

Το σύστημα πρέπει να μην καθυστερεί πάνω από 1 δευτερόλεπτο για να εκτελέσει μία φωνητική εντολή και πάνω από 3 δευτερόλεπτα μεταξύ της στιγμής που ο ομιλητής ολοκληρώνει το λόγο του και της στιγμής που ξεκινάει η διατύπωση της μετάφρασης.

Περιγραφή: Το σύστημα πρέπει να είναι αρκετά γρήγορο στην απόκριση του, τόσο κατά τη διαδικασία αναγνώρισης και εκτέλεσης των εντολών, όσο και κατά τη διαδικασία της μετάφρασης.

User Priority (5/5): Η ταχύτητα του συστήματος είναι ένα πολύ σημαντικό χαρακτηριστικό για τον τελικό χρήστη, καθώς επηρεάζει σημαντικά την εμπειρία χρήσης.

Technical Priority (5/5): Η απαίτηση αυτή είναι εξαιρετικά σημαντική για το σύστημα, καθώς επηρεάζει σε μεγάλο βαθμό τη διαδικασία σχεδιασμού του.

Stability (3/5): Ο χρόνος απόκρισης του συστήματος ενδέχεται να αλλάξει ανάλογα με τις πιθανές διαφορετικές απαιτήσεις των χρηστών, καθώς και με την εξέλιξη της υπολογιστικής ισχύος των νέων συστημάτων.

3.2 Απαιτήσεις Χρηστικότητας (Usability)

<ΜΛΑ- 4>

Το σύστημα πρέπει να επιτρέπει μέχρι 4 διαφορετικές γλώσσες και μέχρι 10 συμμετέχοντες σε μία συζήτηση.

Περιγραφή: Το σύστημα δεν πρέπει να επιτρέπει τη μετάφραση σε παραπάνω από 4 γλώσσες και τη συμμετοχή σε παραπάνω από 10 άτομα στη συζήτηση, καθώς έτσι θα μεγάλωνε αρκετά ο χρόνος που θα χρειαζόταν για τις μεταφράσεις και τον εντοπισμό των ομιλητών.

User Priority (2/5): Ο συγκεκριμένος περιορισμός δεν έχει ιδιαίτερη σημασία για τους χρήστες, καθώς είναι κάτι που ακόμα και αν δεν υπήρχε ως λειτουργικότητα, θα μπορούσε να ρυθμιστεί τεχνητά από αυτούς. Παρ' όλα αυτά εισάγει έναν περιορισμό σε αυτούς, ο οποίος όμως βελτιώνει το συνολικό χρόνο που παίρνει η όλη διαδικασία.

Technical Priority (3/5): Η απαίτηση αυτή επιδρά στο σχεδιασμό του συστήματος κυρίως ως προς τους πόρους που έχει να διαχειριστεί, αλλά δεν είναι απαραίτητη για τη λειτουργία του, ούτε δύσκολη στην υλοποίησή της.

Stability (5/5): Η ΜΛΑ-4 έχει λειτουργικό χαρακτήρα και ο αριθμός συμμετεχόντων και γλωσσών έχει οριοθετηθεί έτσι ώστε το σύστημα να προσφέρει το βέλτιστο User Experience.

<ΜΛΑ-6>

Το σύστημα πρέπει να μπορεί να ενημερώνει το χρήστη για την επιτυχία ή αποτυχία ολοκλήρωσης μιας **ενέργειας**, μέσω ανάλογου μηνύματος και **οπτικοακουστικού ερεθίσματος**.

Περιγραφή: Το σύστημα πρέπει να μπορεί να εμφανίζει κατάλληλα μηνύματα στο χρήστη σχετικά με την εξέλιξη κάποιων ενεργειών που εκείνος αναμένει να ολοκληρωθούν.

User Priority (4/5): Για τον χρήστη είναι πολύ σημαντικό να γνωρίζει την κατάσταση ενεργειών που περιμένει να εκτελεστούν, διότι αυτή η γνώση καθορίζει σε σημαντικό βαθμό τις επόμενες κινήσεις του χρήστη.

Technical Priority (1/5): Για το σύστημα, η συγκεκριμένη απαίτηση είναι αμελητέας σημασίας.

Stability (4/5): Καθώς η ενημέρωση του χρήστη για την επιτυχία ή αποτυχία ολοκλήρωσης μιας ενέργειας είναι αρκετά σημαντική, είναι πολύ πιθανό η ΜΛΑ-6 να μην αλλάξει καθόλου ή να μεταβληθεί ελαφρώς ο τρόπος ενημέρωσης.

3.3 Απαιτήσεις Φορητότητας (Portability)

<ΜΛΑ- 3>

Το σύστημα πρέπει να είναι μεταφέρσιμο.

Περιγραφή: Το σύστημα πρέπει να μπορεί να τροποποιηθεί εύκολα, ώστε να λειτουργήσει και πάνω σε άλλο ανθρωποειδές robot.

User Priority (3/5): Ο χρήστης που έχει εξοικειωθεί με τη χρήση του συστήματος ενδιαφέρεται για την δυνατότητα χρήσης του σε άλλο ανθρωποειδές robot.

Technical Priority (4/5): Η απαίτηση αυτή είναι σημαντική για τον τρόπο σχεδιασμού και λειτουργίας του συστήματος.

Stability (5/5): Δεδομένης της συνεχούς δημιουργίας νέων συστημάτων, είναι απαραίτητο η εφαρμογή να είναι μεταφέρσιμη.

3.4 Απαιτήσεις Αξιοπιστίας (Reliability)

<ΜΛΑ- 1>

Το σύστημα πρέπει να εξασφαλίζει αξιόπιστη σύνδεση με το εξωτερικό σύστημα μετάφρασης και το NAOqi OS στο 99% των περιπτώσεων.

Περιγραφή: Το σύστημα πρέπει να μπορεί να συνδέεται με τα εξωτερικά συστήματα που χρησιμοποιεί για να εκμεταλλευτεί τη λειτουργικότητά τους και αυτή η σύνδεση πρέπει, κατά το δυνατό, να μη διακόπτεται ποτέ.

User Priority (5/5): Η αδυναμία σύνδεσης του συστήματος με τα εξωτερικά συστήματα που πρέπει να χρησιμοποιήσει καθιστά αδύνατη τη λειτουργία του και άρα το σύστημα αυτό γίνεται άχρηστο για το χρήστη. Συνεπώς η δυνατότητα αυτή είναι υψίστης σημασίας για το χρήστη.

Technical Priority (5/5): Η δυνατότητα αυτή είναι εξίσου μεγάλης σημασίας και για την ύπαρξη και λειτουργία του ίδιου του συστήματος.

Stability (4/5): Με την εξέλιξη της τεχνολογίας ενδέχεται το ποσοστό αυτό να αυξηθεί.

3.5 Τεχνικές Απαιτήσεις περιβάλλοντος

<ΜΛΑ- 5>

Το σύστημα πρέπει να μπορεί να **αναγνωρίζει φωνητικά** στην αγγλική γλώσσα τις 100 διαφορετικές γλώσσες που υποστηρίζονται από το εξωτερικό σύστημα μετάφρασης.

Περιγραφή: Το σύστημα πρέπει να μπορεί να αποθηκεύει σε μορφή κειμένου τη γλώσσα που εκφωνεί ένας συμμετέχοντας στα αγγλικά, ώστε να μπορεί να εκτελεστεί σωστά η διαδικασία της μετάφρασης.

User Priority (1/5): Το συγκεκριμένο χαρακτηριστικό δεν επηρεάζει αισθητά το χρήστη, καθώς αφορά μια εσωτερική λειτουργία του συστήματος.

Technical Priority (5/5): Η απαίτηση αυτή είναι εξαιρετικά σημαντική για το σύστημα, καθώς είναι απολύτως απαραίτητη για την εκτέλεση της μετάφρασης.

Stability (3/5): Με τη βελτιστοποίηση των συστημάτων μετάφρασης, ο αριθμός των υποστηριζόμενων γλωσσών ενδέχεται να αυξηθεί.

4. Πρότυπα Σχεδιασμού που υιοθετήθηκαν

Τα παρακάτω πρότυπα σχεδιασμού, που προτιμήθηκαν, δεν σχετίζονται απαραίτητα με τις μη λειτουργικές απαιτήσεις του συστήματος, καθώς οι τελευταίες δεν έθεταν περιορισμούς, οι οποίοι θα απαιτούσαν κάποιο πρότυπο σχεδίασης για την υλοποίηση τους. Επιλέχθηκαν προκειμένου να εξασφαλίζουν την καλύτερη οργάνωση και διαχείριση του συστήματός μας.

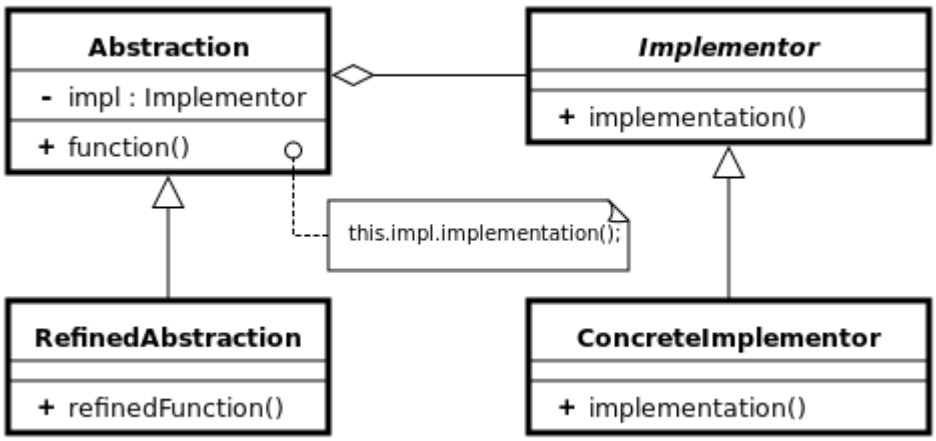
4.1 Δομικά Πρότυπα

Τα δομικά πρότυπα αφορούν τυποποιημένους τρόπους δυναμικής κατασκευής σύνθετων αντικειμένων, τα οποία χρησιμοποιούν υπάρχουσες ιεραρχίες κλάσεων. Εμπεριέχουν σύνθετες δομές, εισάγουν μια abstract κλάση για μελλοντικές επεκτάσεις του συστήματος και επιτυγχάνουν τη μείωση της σύζευξης ανάμεσα σε κλάσεις.

4.1.1 Bridge Design Pattern

Το πρότυπο της Γέφυρας (Bridge Pattern) παρέχει τη δυνατότητα αποσύζευξης μιας αφαίρεσης (abstract) από την υλοποίησή της (implementor), ώστε να μπορούν να υπάρχουν ανεξάρτητα. Συνεπώς, η χρήση της παρέχει πολλές υλοποιήσεις υπό την ίδια διεπαφή (interface) και εφαρμόζεται εκ των προτέρων για να διαχωρίσει τις αφαιρέσεις από τις υλοποιήσεις. Με τον τρόπο αυτό, το σύστημά μας καθίσταται συμβατό με ποικίλα εξωτερικά συστήματα.

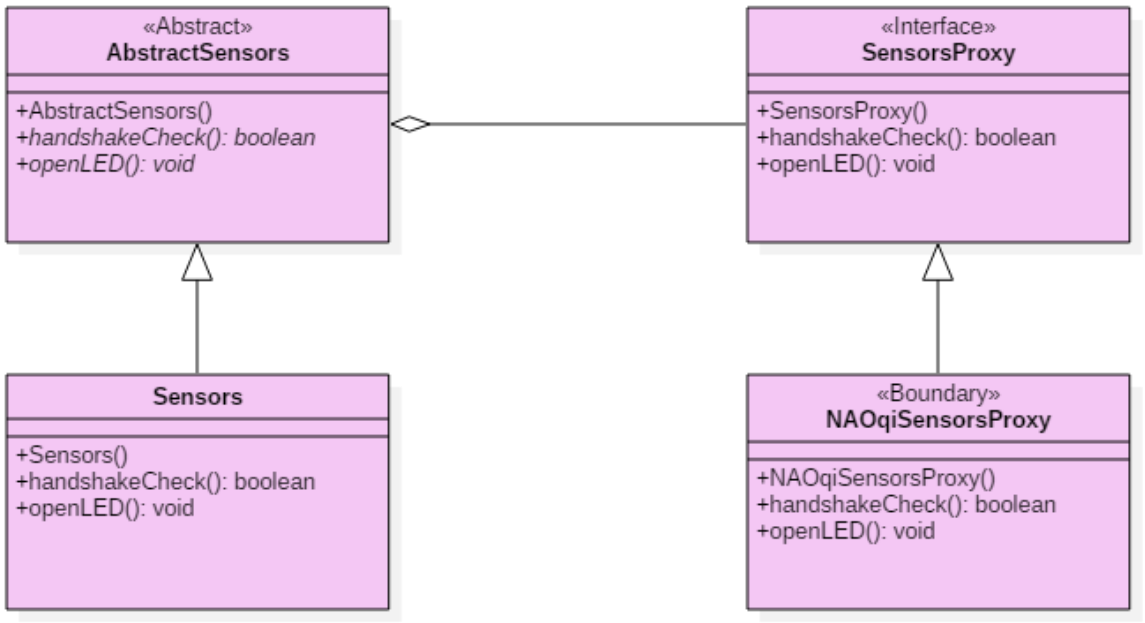
- **Πρόβλημα που αντιμετωπίστηκε:** Το πρότυπο αυτό χρησιμοποιήθηκε για τον σχεδιασμό της διεπαφής του συστήματός μας με συγκεκριμένα υποσυστήματα του NAOqi OS, για την επίτευξη καθοριστικών λειτουργιών του συστήματος. Υιοθετήθηκε το συγκεκριμένο πρότυπο προκειμένου να εξασφαλιστεί συνεργασία της εφαρμογής με ποικίλα εξωτερικά ρομποτικά συστήματα και συστήματα μετάφρασης (με ενδεχομένως ασύμβατες διεπαφές). Με την εφαρμογή του προτύπου της Γέφυρας ικανοποιείται η ΜΛΑ-3.



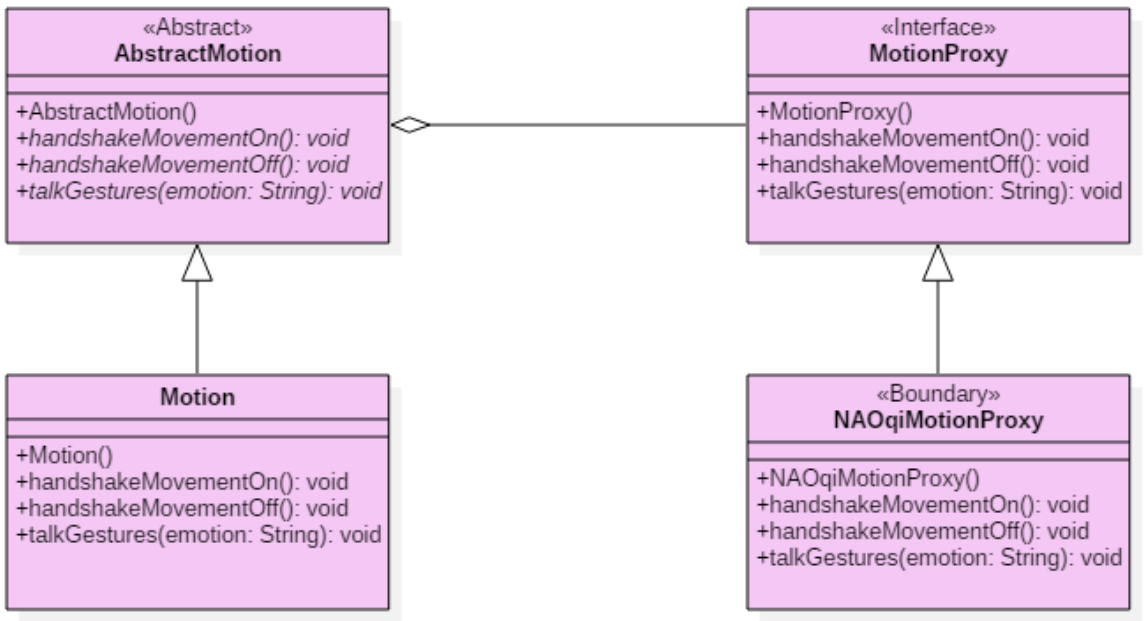
Σχήμα 6: Bridge Design Pattern

Παρακάτω εμφανίζεται η εφαρμογή του προτύπου στις κλάσεις NAOqiSensorsProxy, NAOqiMotionProxy, NAOqiAudioProxy, NAOqiPeolePerceptionProxy:

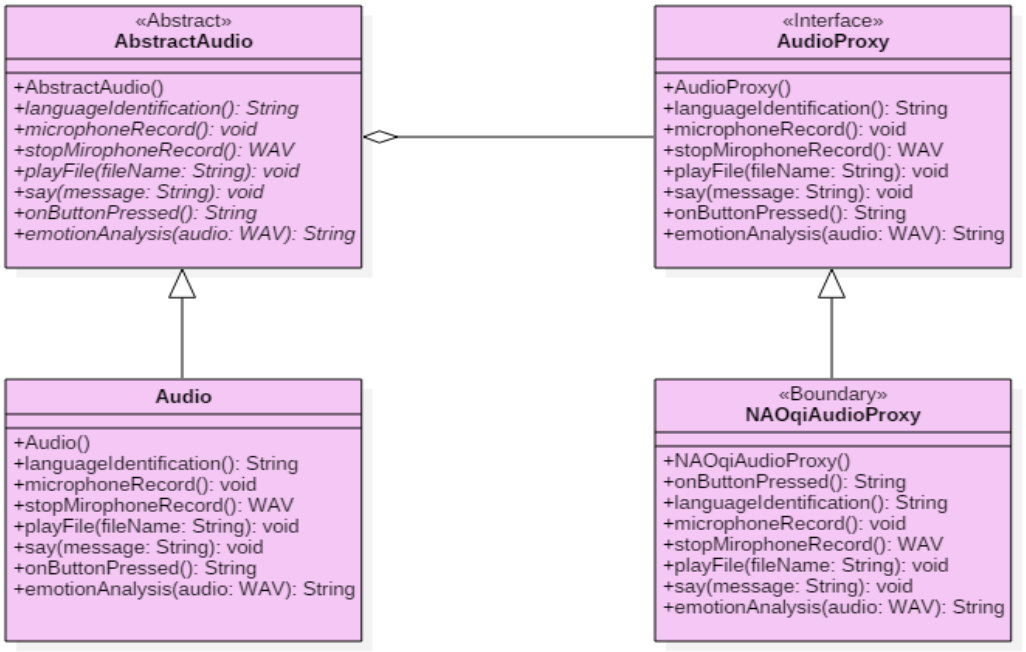
Εφαρμογή προτύπου



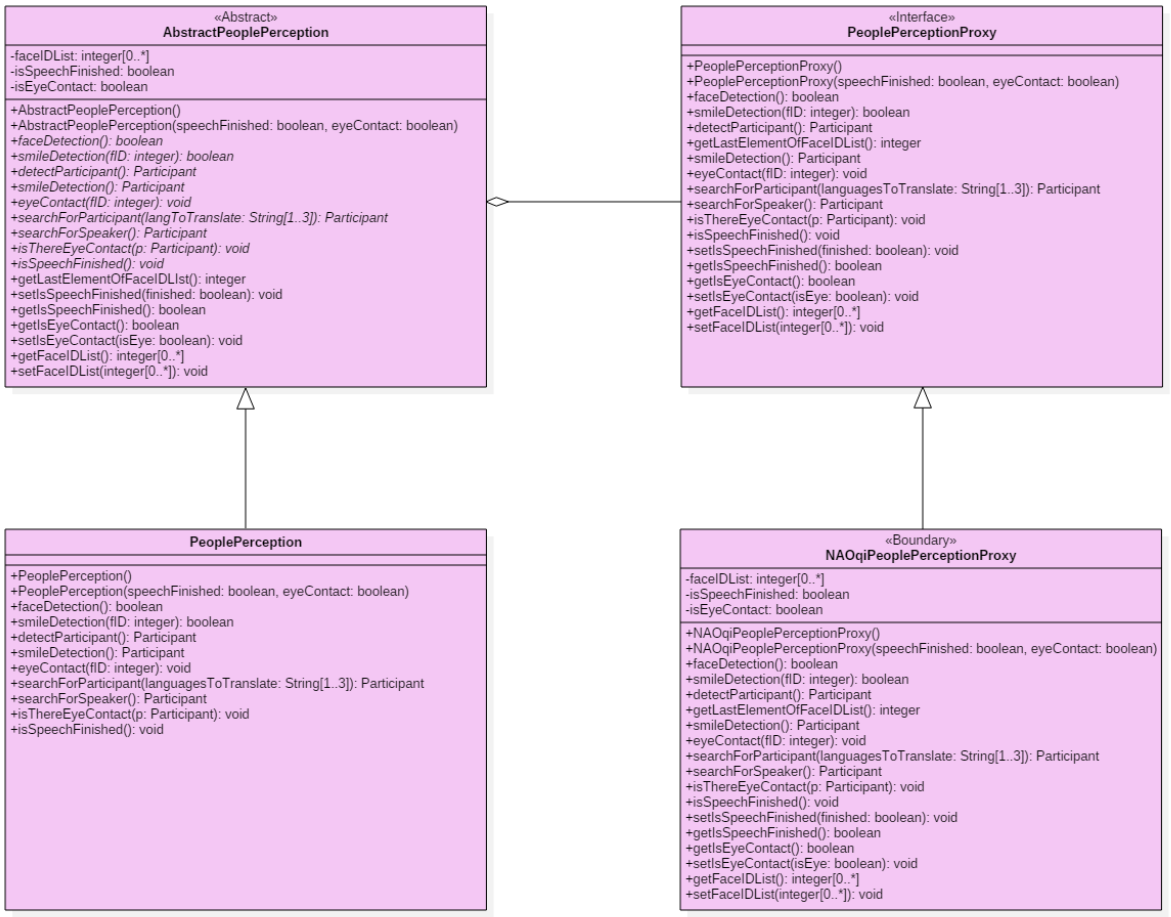
Σχήμα 7: Bridge Pattern για την αρχική κλάση NAOqiSensorsProxy



Σχήμα 8: Bridge Pattern για την αρχική κλάση NAOqiMotionProxy



Σχήμα 9: Bridge Pattern για την αρχική κλάση NAOqiAudioProxy



Σχήμα 10: Bridge Pattern για την αρχική κλάση NAOqiPeoplePerceptionProxy

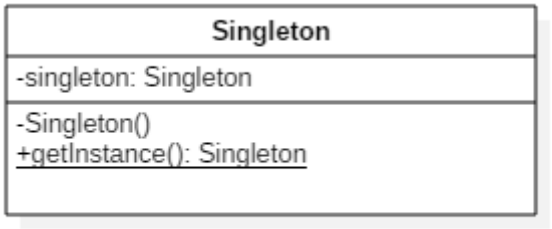
4.2 Δημιουργικά-Κατασκευαστικά Πρότυπα

Τα κατασκευαστικά πρότυπα περιλαμβάνουν τυποποιημένους τρόπους που εξασφαλίζουν δυναμική κατασκευή αντικειμένων κατά τον χρόνο εκτέλεσης. Απώτερος στόχος τους είναι η ανεξαρτητοποίηση του κώδικα που χρησιμοποιεί κάποια αντικείμενα από τις κλάσεις που ορίζουν τα αντικείμενα αυτά και τον τρόπο που καταχωρούνται στη μνήμη.

4.1.2 Singleton Design Pattern

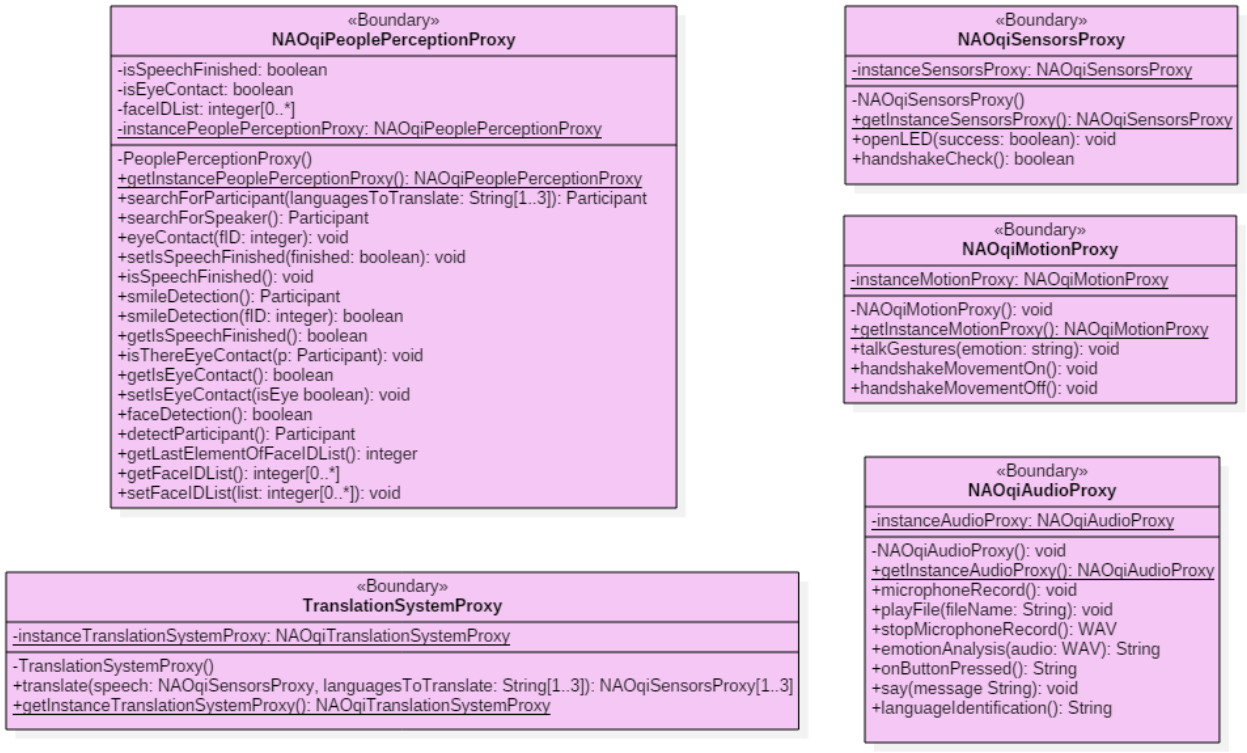
Το Singleton είναι ένα σχεδιαστικό πρότυπο που επιλύει το ζήτημα της εξασφάλισης της ύπαρξης το πολύ ενός στιγμιότυπου κάποιας κλάσης (έστω Α) κατά το χρόνο εκτέλεσης. Με το πρότυπο Singleton η ικανοποίηση αυτού του περιορισμού ανατίθεται στην ίδια την κλάση Α και δε μεταβιβάζεται στο εξωτερικό πρόγραμμα που τη χρησιμοποιεί. Αυτό γίνεται μέσω μίας στατικής δημόσιας μεθόδου της Α, η οποία δημιουργεί το πρώτο στιγμιότυπο της κλάσης και ακολούθως, σε κάθε επόμενη κλήση της, επιστρέφει έναν δείκτη προς το υπάρχον στιγμιότυπο.

- **Πρόβλημα που αντιμετωπίστηκε:** Το πρότυπο αυτό χρησιμοποιήθηκε για τον σχεδιασμό των οριακών κλάσεων NAOqiAudioProxy, NAOqiMotionProxy, NAOqiSensorsProxy, NAOqiPeoplePerception. Έτσι, καθώς στιγμιότυπα αυτής της κλάσης καλούνται αρκετές φορές, σε αρκετές διαφορετικές κλάσεις κατά τη διάρκεια της εκτέλεσης της εφαρμογής, επιτυγχάνεται η δημιουργία το πολύ ενός εξ αυτών. Το πρότυπο αυτό υιοθετήθηκε με σκοπό την δέσμευση μικρότερης ποσότητας μνήμης του συστήματος και την ασφάλεια προσπέλασης των δεδομένων.



Σχήμα 11: Singleton Design Pattern

Εφαρμογή προτύπου

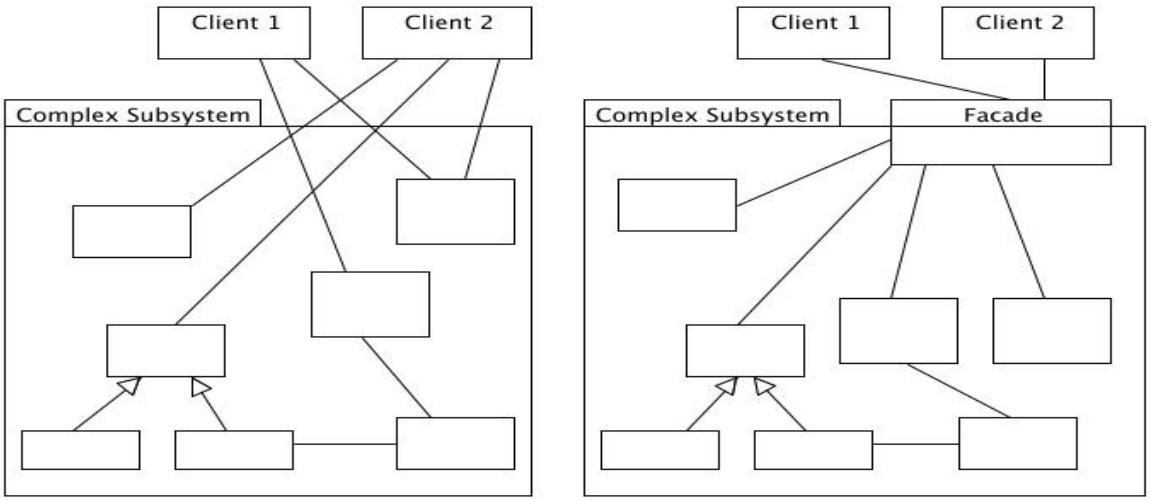


Σχήμα 12: Το πρότυπο Singleton, όπως εφαρμόστηκε στις οριακές κλάσεις

4.2.2 Facade Design Pattern

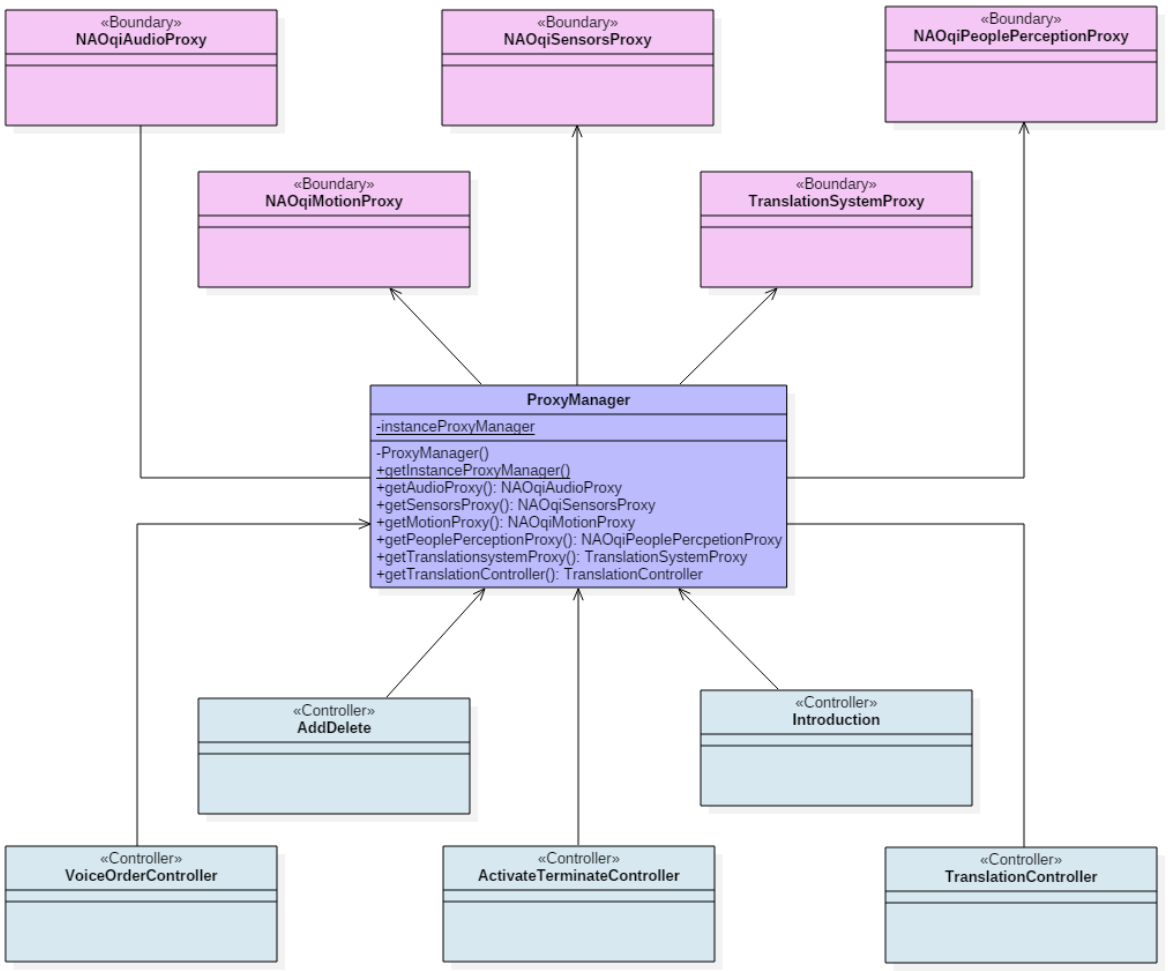
Το πρότυπο Πρόσοψης (Facade) είναι ένα σχεδιαστικό πρότυπο που χρησιμοποιείται σε περίπτωση που για την εκτέλεση μίας εργασίας είναι διαθέσιμο ένα περίπλοκο σύνολο αλληλεπιδρώντων κλάσεων οι οποίες συνδυάζονται με διάφορους τρόπους. Το πρότυπο αυτό παρέχει μια ενοποιημένη, υψηλού επιπέδου διεπαφή σε ένα σετ από αντικείμενα ενός υποσυστήματος, καθιστώντας το πιο εύκολο στη χρήση. Τα πρότυπα Facade επιτρέπουν τη χρήση κλειστών αρχιτεκτονικών.

- **<u>Πρόβλημα που αντιμετωπίστηκε:</u>** Το πρότυπο αυτό χρησιμοποιήθηκε για την επικοινωνία των κλάσεων τύπου «Controller» με τις κλάσεις τύπου «Boundary» που επικοινωνούν με τα υποσυστήματα του NAOqi OS. Οι τελευταίες ομαδοποιήθηκαν ως υποσύστημα και για την ενοποιημένη διεπαφή τους με τις υπόλοιπες κλάσεις του συστήματος δημιουργήθηκε μία επιπλέον Facade κλάση, η ProxyManager. Το πρότυπο αυτό υιοθετήθηκε με σκοπό την ευκολότερη προσπέλαση δεδομένων και την εξασφάλιση της ακέραιας λειτουργίας του συστήματός μας.
  Για την ορθότερη λειτουργία του προτύπου Facade, επιλέχθηκε η υλοποίησή του με βάση το πρότυπο Singleton, το οποίο αναλύθηκε προηγουμένως. Η χρήση του αποσκοπεί στην αποφυγή της δημιουργίας αντικειμένου ProxyManager, πέραν της μίας φοράς.



Σχήμα 13: Facade Design Pattern

Εφαρμογή προτύπου



Σχήμα 14: Το εφαρμοσμένο πρότυπο Facade, υλοποιημένο με τη χρήση του προτύπου Singleton

Παράρτημα Ι – Πίνακας Ιχνηλασιμότητας

Η μεταβολή από το Έγγραφο Απαιτήσεων Χρηστών στο Έγγραφο Απαιτήσεων Λογισμικού δεν εμπεριείχε κάποιες μεταβολές του αρχικού, συνεπώς δεν παρατίθεται Πίνακας Ιχνηλασιμότητας. Οι διαφοροποιήσεις που οφείλονται στα πρότυπα σχεδιασμού, έχουν ήδη αναλυθεί στο Κεφάλαιο 4 και συνεπώς δεν επαναλαμβάνονται.

Παράρτημα ΙΙ – Ανοιχτά Θέματα

1) Δεδομένης της ύπαρξης πολλών συναρτήσεων με λειτουργία παρόμοια με αυτή των Observers, η Ομάδα Ανάπτυξης προτείνει πέραν των αναλυθέντων προτύπων σχεδίασης και τη χρήση του συμπεριφορικού προτύπου Observer pattern για εφαρμογή του στις event driven μεθόδους ορισμένων κλάσεων.

2) Δεδομένων των μετρίων τιμών της σταθερότητας (stability) των μη λειτουργικών απαιτήσεων ΜΛΑ-2, ΜΛΑ-5 και λιγότερο της ΜΛΑ-1, σε περίπτωση που αυτές τελικά μεταβληθούν και το σύστημα υπόκειται πλέον σε νέους περιορισμούς, το έγγραφο αυτό πρέπει να ανανεωθεί σύμφωνα με τα νέα δεδομένα.