ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΑΤΡΩΝ
ΤΜΗΜΑ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ
ΚΑΙ ΤΕΧΝΟΛΟΓΙΑΣ ΥΠΟΛΟΓΙΣΤΩΝ

ΤΟΜΕΑΣ: Τηλεπικοινωνιών και Τεχνολογίας Πληροφορίας

Εργαστήριο: Ενσύρματης Επικοινωνίας & Τεχνολογίας της Πληροφορίας

Διπλωματική Εργασία

του φοιτητή του Τμήματος Ηλεκτρολόγων Μηχανικών και Τεχνολογίας Υπολογιστών της Πολυτεχνικής Σχολής του Πανεπιστημίου Πατρών

Χρήστου Φραντζόλα του Γεωργίου

Αριθμός Μητρώου: 1053706

Θέμα

«Ανάπτυξη Παιχνιδιού Διαδικαστικής Αφήγησης με Δυναμικούς Διαλόγους»

Επιβλέπων
Κυριάκος Σγάρμπας,
Αναπληρωτής Καθηγητής

Αριθμός Διπλωματικής Εργασίας:
(1053706 /2021)

Πάτρα, Ιούλιος 2021
01/07/2021

ΠΙΣΤΟΠΟΙΗΣΗ

Πιστοποιείται ότι η Διπλωματική Εργασία με θέμα

«Ανάπτυξη Παιχνιδιού Διαδικαστικής Αφήγησης με Δυναμικούς Διαλόγους»

του φοιτητή του Τμήματος Ηλεκτρολόγων Μηχανικών και Τεχνολογίας Υπολογιστών της Πολυτεχνικής Σχολής του Πανεπιστημίου Πατρών

Χρήστου Φραντζόλα του Γεωργίου

Αριθμός Μητρώου: 1053706

Παρουσιάστηκε δημόσια και εξετάστηκε στο Τμήμα Ηλεκτρολόγων Μηχανικών και Τεχνολογίας Υπολογιστών στις 01/07/2021

Ο Επιβλέπων	Ο Διευθυντής του Τομέα
Κυριάκος Σγάρμπας	Κυριάκος Σγάρμπας
Αναπληρωτής Καθηγητής	Αναπληρωτής Καθηγητής

Αριθμός Διπλωματικής Εργασίας: 1053706/2021

Θέμα: «Ανάπτυξη Παιχνιδιού Διαδικαστικής Αφήγησης με Δυναμικούς Διαλόγους»

Φοιτητής:	**Επιβλέπων:**
Φραντζόλας Χρήστος του Γεωργίου	Κυριάκος Σγάρμπας

Περίληψη

Διαδικαστική Παραγωγή (Procedural Generation) ονομάζεται η τεχνική δημιουργίας ψηφιακού υλικού αλγοριθμικά, με μικρές απαιτήσεις από πλευράς μνήμης, ούτως ώστε τα αντικείμενα που παράγονται να είναι μοναδικά και επαναλήψιμα. Ένα είδος διαδικαστικής παραγωγής αποτελεί η *Διαδικαστική Αφήγηση* (procedural storytelling ή narrative generation), η οποία αφορά την εφαρμογή των ίδιων αρχών στη δημιουργία σεναρίων. Η διαδικαστική παραγωγή γενικά, αλλά και η διαδικαστική αφήγηση ειδικότερα, χρησιμοποιούνται ευρέως σε βιντεοπαιχνίδια, για τη δημιουργία τοπίων, αντικειμένων, περιγραφών, διαλόγων ή άλλων assets.

Σκοπός της εργασίας αυτής ήταν η υλοποίηση μίας διαδραστικής εφαρμογής η οποία επιδεικνύει διαδικαστική παραγωγή δυναμικών διαλόγων, δηλαδή διαλόγων με αποκρίσεις που δεν είναι προκαθορισμένες αλλά εξαρτώνται από το γενικό πλαίσιο και τα συμφραζόμενα στο περιβάλλον του παιχνιδιού για κάθε χρονική στιγμή. Η εφαρμογή που αναπτύχθηκε είναι στη μορφή ενός παιχνιδιού βασισμένο σε κείμενο (text-based game), το οποίο υλοποιήθηκε εξ ολοκλήρου στη γλώσσα προγραμματισμού Python 3. Για τις ανάγκες της παρούσας εργασίας, πέρα από την ανάπτυξη της συγκεκριμένης εφαρμογής, υλοποιήθηκαν και οι μηχανισμοί μίας πλατφόρμας ανάπτυξης παιχνιδιών *Διαδραστικής Φαντασίας* (Interactive Fiction, IF), όπως για παράδειγμα η γλώσσα προγραμματισμού TADS 3.

Λέξεις κλειδιά: Διαδικαστική Παραγωγή; Διαδικαστική Αφήγηση; Ανάπτυξη Παιχνιδιού; Παιχνίδια Βασισμένα σε Κείμενο; Δυναμικοί Διάλογοι; Διαδραστική Φαντασία; Python 3;

Abstract

Procedural Generation is the technique of creating digital material algorithmically, in a way that guarantees the uniqueness and reproducibility of the generated objects, with minimal memory requirements. Procedural storytelling (also called narrative generation) is a type of procedural generation which implements the same principles in the creation of written stories. Both procedural and narrative generation are widely utilized in game development for constructing sceneries, objects, descriptions, dialogs, and other game assets.

This thesis was aimed at the creation of an interactive application which exhibits procedural generation of dynamic dialogs; that is, dialogs with responses which are not predetermined, but depend on the context and the state of the game at any given time. The application is in the form of a text-based game which was implemented using the programming language Python 3. For the needs of this project, some of the aspects of an Interactive Fiction programming language were also implemented using Python. Inspiration for some of these functions was drawn by the functionalities of the TADS 3 standard libraries for creating IF Games.

Keywords: Procedural Generation; Narrative Generation; Game Development; Text-Based Games; Dynamic Dialogs; Interactive Fiction; Python 3;

Ευχαριστίες

Αρχικά, θα ήθελα να ευχαριστήσω τον επιβλέποντα καθηγητή μου κύριο Κυριάκο Σγάρμπα, για την αμέριστη στήριξη και τη βοήθειά που μου πρόσφερε κατά την υλοποίηση της παρούσας εργασίας.

Επιπλέον, δε θα μπορούσα να είχα πραγματοποιήσει αυτήν την εργασία χωρίς τη στήριξη και την προσπάθεια της οικογένειάς μου, και για αυτό τους ευχαριστώ όλους μέσα από την καρδιά μου. Χωρίς τη μακροχρόνια βοήθειά και ενθάρρυνσή τους είναι σίγουρο πως θα μου ήταν πολύ πιο δύσκολο να βρίσκομαι σήμερα στο τέλος της πορείας μου στο τμήμα ΗΜ&ΤΥ.

Τέλος, είμαι πάντα ευγνώμων για τη στήριξη και την υπομονή των φίλων μου και όλων των άλλων ανθρώπων που με τον δικό τους τρόπο με βοήθησαν να φτάσω ως εδώ.

Περιεχόμενα

Κατάλογος σχημάτων

1. ΕΙΣΑΓΩΓΗ

1.1 Στόχος της διπλωματικής εργασίας

Ο σχεδιασμός και η υλοποίηση ενός παιχνιδιού είναι μία πρόκληση η οποία απαιτεί τόσο καλές γνώσεις προγραμματισμού, όσο και δημιουργικότητα και κατανόηση του τι κάνει ένα παιχνίδι ευχάριστο για τον εκάστοτε παίκτη. Με την αλματώδη ανάπτυξη της τεχνολογίας και με τον σχεδιασμό ενός βιντεοπαιχνιδιού να γίνεται πιο περίτεχνος, η ζήτηση του αγοραστικού κοινού για πιο πρωτότυπα και πιο εντυπωσιακά παιχνίδια ολοένα και αυξάνει. Η ανάγκη των παικτών για περισσότερα assets, σενάρια και χαρακτήρες έχει φτάσει στο σημείο να ξεπεράσει σε μεγάλο βαθμό την ικανότητα των σχεδιαστών να δημιουργήσουν εκ των προτέρων ολόκληρα παιχνίδια.

Μία λύση για την αποφυγή του προβλήματος αυτού του πληθωρισμού στη ζήτηση περισσότερων features αποτελεί η *Διαδικαστική Παραγωγή* (Procedural Generation), η οποία μεταθέτει τη δημιουργική διαδικασία ως έναν βαθμό από τον άνθρωπο σε μηχανές, για την αλγοριθμική παραγωγή *Αντικειμένων* (Artifacts). Ειδικότερα, η διαδικαστική παραγωγή ιστοριών, δηλαδή η *Διαδικαστική Αφήγηση*, παρουσιάζει μία ιδιαίτερη πρόκληση προς τους προγραμματιστές και στους σχεδιαστές παιχνιδιών, εφόσον ο τρόπος με τον οποίο οι ανθρώπινοι παίκτες αντιλαμβάνονται ένα σενάριο καθιστά τα επιμέρους στοιχεία του βαθιά αλληλένδετα και δύσκολο να παραχθούν από έναν υπολογιστή.

Στην παρούσα διπλωματική εργασία στόχος ήταν η προσέγγιση του προβλήματος της διαδικαστικής αφήγησης, με τη δημιουργία μιας διαδραστικής εφαρμογής κειμένου, και ειδικότερα ενός παιχνιδιού κειμένου (text-based game). Παρά την ευρεία διαθεσιμότητα

διαφόρων γλωσσών και εργαλείων ειδικού σκοπού για την δημιουργία παιχνιδιών *Διαδραστικής Φαντασίας* (Interactive Fiction, στο εξής IF), θεωρήθηκε σκόπιμο - για τις ανάγκες της εργασίας - να αναπτυχθούν νέα εργαλεία από μηδενική βάση. Η αποφυγή της χρήσης ήδη έτοιμων εργαλείων αποσκοπούσε τόσο στην περαιτέρω εξοικείωση με μία γλώσσα προγραμματισμού γενικού σκοπού, όπως είναι η Python, όσο και στην εμπέδωση σε βάθος των θεωρητικών γνώσεων και μηχανισμών που απαρτίζουν μία εφαρμογή διαδικαστικής αφήγησης.

Συνολικός στόχος, λοιπόν, της διπλωματικής εργασίας αυτής ήταν η υλοποίηση ενός παιχνιδιού διαδικαστικής αφήγησης, μαζί και με τη δημιουργία των κατάλληλων εργαλείων για την δημιουργία μίας τέτοιας εφαρμογής. Επιπλέον, το παιχνίδι αυτό έπρεπε να επιδεικνύει την παραγωγή δυναμικών διαλόγων, δηλαδή αποκρίσεων σε ερωτήσεις και εντολές του παίκτη, οι οποίες δεν είναι προκαθορισμένες από τον σχεδιαστή του παιχνιδιού, αλλά μεταβάλλονται ανάλογα με την κατάσταση του κόσμου του παιχνιδιού (Game State) και με βάση προηγούμενες πράξεις του παίκτη.

Τέλος, οι παραπάνω στόχοι υλοποιήθηκαν λαμβάνοντας υπόψιν και άλλους παράγοντες, όπως τη δημιουργική πλευρά της συγγραφής ενός text-based παιχνιδιού και τις παρατηρήσεις παικτών σε δοκιμές beta για την βελτίωση της εμπειρίας τους. Πολύ σημαντική ήταν και η υλοποίηση μηχανισμών που μπορούν να επιτρέψουν στη δημιουργία ακόμα πιο περίπλοκων παιχνιδιών από αυτό που υλοποιήθηκε στην παρούσα εργασία.

1.2 Διάρθρωση εργασίας

Κεφάλαιο 1: Αναφορά στους στόχους της διπλωματικής εργασίας.

Κεφάλαιο 2: Στο κεφάλαιο αυτό αναλύονται ορισμένες βασικές έννοιες της διαδικαστικής αφήγησης, ενώ γίνεται αναφορά στις πιο διαδεδομένες πλατφόρμες ειδικού σκοπού για την ανάπτυξη παιχνιδιών διαδραστικής φαντασίας.

Κεφάλαιο 3: Εξετάζονται οι θεωρητικές έννοιες που είναι απαραίτητες για την κατανόηση της λειτουργίας της εφαρμογής. Πιο συγκεκριμένα, εξετάζονται διάφορες τεχνικές συντακτικής ανάλυσης, διαδικαστικής παραγωγής αντικειμένων, κειμένου και διαλόγων,

καθώς και βασικές έννοιες του αντικειμενοστραφούς προγραμματισμού. Τέλος, παρουσιάζονται σημαντικές έννοιες και παρατηρήσεις για τον σχεδιασμό παιχνιδιών διαδικαστικής αφήγησης.

Κεφάλαιο 4: Αρχικά, παρουσιάζονται τα προγραμματιστικά εργαλεία που χρησιμοποιήθηκαν για την ανάπτυξη του παιχνιδιού της παρούσας διπλωματικής εργασίας. Εν συνεχεία, αναλύεται η λειτουργία του παιχνιδιού, με την παρουσίαση των κλάσεων και των σημαντικότερων μεθόδων που αναπτύχθηκαν.

Κεφάλαιο 5: Παρουσιάζονται τα αποτελέσματα της υλοποίησης του παιχνιδιού, πληροφορίες για την υπόθεσή του, καθώς και παρατηρήσεις από δοκιμές beta που έγιναν. Ύστερα παρουσιάζονται κάποια λάθη και ελλείψεις του παιχνιδιού και του συστήματος, καθώς και προτάσεις για μελλοντικές τους βελτιώσεις.

Κεφάλαιο 6: Ο Επίλογος της παρούσας εργασίας.

2. ΒΑΣΙΚΕΣ ΕΝΝΟΙΕΣ ΔΙΑΔΙΚΑΣΤΙΚΗΣ ΑΦΗΓΗΣΗΣ

2.1 Διαδραστικά μυθιστορήματα και παιχνίδια

2.1.1 Ορισμοί

Με την ραγδαία ανάπτυξη της βιομηχανίας των βιντεοπαιχνιδιών τις τελευταίες δεκαετίες και με τη δημιουργία περιεχομένου για τα παιχνίδια αυτά να είναι μία από τις κυριότερες προκλήσεις κατά την ανάπτυξή τους, έχει δημιουργηθεί η ανάγκη για την υλοποίηση μεθόδων παραγωγής νέου υλικού χωρίς την εμπλοκή ανθρώπινων σχεδιαστών.

Σε αντίθεση με την ανθρώπινη παραγωγή περιεχομένου, η *Διαδικαστική Παραγωγή Περιεχομένου* (Procedural Content Generation) για παιχνίδια είναι η χρήση υπολογιστικών μηχανών για την παραγωγή αντικειμένων (artifacts), καθώς και η επιλογή - ανάμεσα στο παραγόμενο υλικό - των πιο χρήσιμων και ικανοποιητικών για τον παίκτη παραδειγμάτων. Η δυσκολία της διαδικαστικής παραγωγής έγκειται όχι μόνο στην υπολογιστική δύναμη που απαιτεί, αλλά στην ανάγκη εύρεσης μετρικών για να καθοριστεί η τεχνική και υποκειμενική αξία του παραγόμενου υλικού [1]. Το υλικό το οποίο παράγεται μπορεί να αφορά είτε τα συστατικά στοιχεία του παιχνιδιού (textures, αντικείμενα, ήχους), τον χώρο του παιχνιδιού (game space), ή και ολόκληρα συστήματα (πολλές φορές υλοποιημένα μέσα από προσομοιώσεις συστημάτων). Η παραγωγή του υλικού μπορεί να συνδυάζει με διαφορετικούς τρόπους τόσο αλγοριθμικές μεθόδους, όσο και συστατικά στοιχεία που έχουν δημιουργηθεί από ανθρώπους.

Συγκεκριμένα, *Διαδικαστική Αφήγηση/Διήγηση ή Παραγωγή Σεναρίου* (Procedural Storytelling ή Narrative Generation) ονομάζεται η παραγωγή κειμένου με τη χρήση αλγοριθμικών μεθόδων. Τα παραγόμενα αυτά σενάρια μπορεί χρησιμοποιηθούν για να βάλουν τον παίκτη στη θέση του αφηγητή και του δημιουργού της υπόθεσης, μετατρέποντας μία στατική εφαρμογή κειμένου σε διαδραστική ιστορία. Η ιστορία μπορεί να παρουσιάζεται στη μορφή απλής αφήγησης, διαλόγων, στοιχείων μέσα στον κόσμο του παιχνιδιού (επιγραφές, αποκόμματα κειμένων) ή σε ενδιάμεσες σκηνές που δημιουργούν ένα αφηγηματικό πλαίσιο για το παιχνίδι [2]. Η διαδικαστική δημιουργία κειμένου, λοιπόν, είναι μία τεχνική η οποία μπορεί να ενσωματωθεί σε ένα παιχνίδι ή άλλη παρόμοια εφαρμογή με πολλούς τρόπους, και σε διαφορετικό βαθμό.

Μία ιδιαίτερη κατηγορία παιχνιδιών που περιλαμβάνουν σε μεγάλο ποσοστό στοιχεία διαδικαστικής αφήγησης είναι τα *Παιχνίδια Διαδραστικής Φαντασίας*. Τα παιχνίδια αυτά έχουν ως κύριο χαρακτηριστικό το ότι η διεπαφή με τον χρήστη (τόσο στην εισαγωγή εντολών, όσο και στην επιστροφή αποτελεσμάτων) γίνεται με τη μορφή κειμένου φυσικής γλώσσας. Οι εντολές του χρήστη ερμηνεύονται από το παιχνίδι ως πράξεις σε έναν μοντελοποιημένο κόσμο, με τον οποίο ο παίκτης μπορεί να αλληλοεπιδράσει. Ως εκ τούτου, ο παίκτης πρέπει να έχει τη δυνατότητα να κατανοήσει τους βασικούς κανόνες του κόσμου, αλλά παράλληλα να έχει την ευκαιρία να ανακαλύψει νέες πτυχές του [3].

2.1.2 Κατηγορίες παιχνιδιών κειμένου

Τα παιχνίδια κειμένου εμφανίζουν ιδιαίτερη ποικιλία μορφών, τόσο στον τρόπο με τον οποίο ο παίκτης αλληλοεπιδρά με τον κόσμο του παιχνιδιού, όσο και με το περιεχόμενο και τους στόχους του παιχνιδιού.

Ένας πρώτος διαχωρισμός μπορεί να γίνει μεταξύ των text-based παιχνιδιών και των graphical adventures. Τα πρώτα αποτελούν πιο χαρακτηριστικό παράδειγμα IF, καθώς οι παίκτες αντιλαμβάνονται τον κόσμο του παιχνιδιού και αλληλοεπιδρούν με αυτόν μόνο μέσω κειμένου. Αντίθετα, τα παιχνίδια της δεύτερης κατηγορίας χρησιμοποιούν εικόνες και βίντεο είτε για την παρουσίαση στοιχείων στον παίκτη, είτε ως μέσο για την λήψη εντολών από αυτόν. Διατηρούν, βέβαια το στοιχείο της αλληλοεπίδρασης και της διαδικαστικής αφήγησης και για αυτό πολλές φορές εντάσσονται στην ίδια κατηγορία με τα παιχνίδια κειμένου.

Ανάλογα με τη μορφή εισόδου των εντολών του παίκτη σε ένα παιχνίδι κειμένου, μπορούμε να τα ταξινομήσουμε ως εξής:

- Parser games: Παιχνίδια που χρησιμοποιούν parsers (συντακτικούς αναλυτές) και επιτρέπουν στον παίκτη να εισάγει εντολές σε μορφή προτάσεων φυσικής γλώσσας (με κάποιους περιορισμούς στην σύνταξη και στο λεξιλόγιο). Οι εντολές του παίκτη αντιστοιχίζονται έπειτα με πράξεις και οντότητες του παιχνιδιού, ώστε να εφαρμοστούν στον προσομοιωμένο κόσμο του [22].
- Hyperlink-based games: Αυτού του είδους τα παιχνίδια (τα οποία αναφέρονται και ως hypertext games) δεν περιλαμβάνουν εντολές που δίνονται από τον παίκτη σε μορφή κειμένου, αλλά αντίθετα οι επιλογές του παίκτη για το επόμενο βήμα που μπορεί να κάνει παρουσιάζονται μέσα στο κείμενο με τη μορφή υπερσυνδέσμων (hyperlinks). Οι υπερσύνδεσμοι αυτοί οδηγούν σε άλλα διασυνδεδεμένα κείμενα που ονομάζονται και lexia [22].
- Menu-based games: Σε παιχνίδια βασισμένα σε μενού επιλογών, οι επιλογές του παίκτη παρουσιάζονται με τη μορφή ενός καταλόγου. Ο κατάλογος αυτός μπορεί να έχει ιεραρχική δομή, εξειδικεύοντας περαιτέρω τις εντολές, ανάλογα με τις επιλογές που κάνει ο χρήστης. Τα συστήματα αυτά έχουν χρησιμοποιηθεί και για την υλοποίηση συστημάτων δυναμικών διαλόγων, όπως για παράδειγμα στην περίπτωση του *Cyborg (1981)* [22].
- Choice-based games: Στα παιχνίδια αυτού του τύπου τα επόμενα πιθανά βήματα δίνονται στον παίκτη στο τέλος ενός αποσπάσματος κειμένου, με τη μορφή πολλαπλών επιλογών [22], οι οποίες όμως, σε αντίθεση με τα menu-based games, δεν εξειδικεύονται περαιτέρω.

Μια ακόμα διάκριση μπορεί να γίνει με βάση το περιεχόμενο των παιχνιδιών, και πιο συγκεκριμένα με βάση το ποια στοιχεία του παιχνιδιού χρησιμοποιούνται περισσότερο για να τραβήξουν την προσοχή του παίκτη. Με αυτόν τον τρόπο, τα παιχνίδια διαδραστικής φαντασίας μπορούν να χωριστούν σε 2 ευρείς κατηγορίες: τα παιχνίδια γρίφων (puzzles) και τα παιχνίδια που δεν έχουν ως κύριο στοιχείο τους την επίλυση ενός γρίφου, αλλά επικεντρώνονται είτε στην εξερεύνηση και ανακάλυψη του κόσμου του παιχνιδιού, είτε στην ίδια την αφήγηση. Οι εφαρμογές της δεύτερης κατηγορίας πολλές φορές δεν

χαρακτηρίζονται ως παιχνίδια IF, αλλά γενικά ως «έργα» διαδραστικής φαντασίας, καθώς μπορεί να μην περιέχουν κάποιον συγκεκριμένο στόχο για τον παίκτη, ούτε κάποιο score το οποίο καλείται να μεγιστοποιήσει [3].

2.1.3 Ιστορική αναδρομή και παραδείγματα παιχνιδιών Interactive Fiction

Όπως αναφέρεται στο κεφάλαιο 46 (A Short History of Interactive Fiction) του [23], ένα από τα πρώτα παιχνίδια τα οποία θα μπορούσαμε να χαρακτηρίσουμε σήμερα ότι ανήκει στην κατηγορία της διαδραστικής φαντασίας είναι το παιχνίδι *Adventure (1975-76),* το οποίο δημιουργήθηκε αρχικά από τον Will Crowther το 1975, ενώ εξελίχθηκε μετά από τον Don Woods στο SAIL (Stanford Artificial Intelligence Laboratory) το 1976. Αρχικά, ο στόχος του Crowther φαίνεται να ήταν η δημιουργία ενός παιχνιδιού εξερεύνησης σε σπήλαια, επηρεασμένο από το Role Playing Game "Dungeons and Dragons". Η διεπαφή μέσω κειμένου φυσικής γλώσσας είχε ως στόχο την φιλικότητα του παιχνιδιού και σε χρήστες που δεν ήταν σχετικοί με το αντικείμενο των υπολογιστών. Αντίθετα, οι επεκτάσεις και οι αλλαγές που έκανε ο Woods στο παιχνίδι στράφηκαν προς μία φιλοσοφία πιο κοντά στην επίλυση γρίφων και όχι προς την εξερεύνηση ενός ρεαλιστικού μοντελοποιημένου κόσμου [23]. Τα επόμενα χρόνια, το Advent γνώρισε μεγάλη απήχηση και αποτέλεσε έμπνευση για άλλα "Adventure-like games", όπως το *Multi-User Dungeon (MUD).*

Τον Ιούνιο του 1979 ιδρύεται η *Infocom,* η οποία δημιούργησε πολλά εμπορικά επιτυχημένα παιχνίδια IF τη δεκαετία του '80. Κάποια από τα σημαντικότερα παιχνίδια της δεκαετίας είναι η τριλογία *'Zork' (1979 και έπειτα), 'Mystery House' (1980), 'Randezvous with Rama' (1984), 'Fahrenheit 451' (1984), 'Amazon' (1984).* Προσπάθειες για την ενσωμάτωση μίας δραματικής ιστορίας για να πλαισιώσει το περιεχόμενο των γρίφων (όπως με το *'Amnesia' (1986)*) δε γνώρισαν ιδιαίτερη εμπορική επιτυχία [23]. Τα πιο συνηθισμένα παιχνίδια της εποχής είχαν τη μορφή παιχνιδιών μυστηρίου (πχ. *'Murder on the Zinderneuf' (1981)*), ή εξερεύνησης και επίλυσης γρίφων [24].

Μπαίνοντας στην δεκαετία του '90, πολλές από τις εταιρείες που ανέπτυσσαν παιχνίδια IF τα προηγούμενα χρόνια έπαψαν να υπάρχουν [23]. Αντ' αυτού, δημιουργήθηκαν πολλές δωρεάν πλατφόρμες όπως το *TADS* και το *Inform*, οι οποίες έδωσαν τη δυνατότητα σε αρχάριους δημιουργούς να αναπτύξουν τα δικά τους παιχνίδια. Με την εξάπλωση των

προσωπικών υπολογιστών και του διαδικτύου, μία κοινότητα άρχισε να αναπτύσσεται γύρω από το είδος, μέσω διαδικτυακών archives και forums. Διαγωνισμοί όπως ο *IFComp (1995 έως σήμερα)* και ο *SpeedIF* αποτέλεσαν πόλο έλξης για επίδοξους δημιουργούς, και οδήγησαν το είδος σε ολοένα και πιο καλλιτεχνικά μονοπάτια που απομακρύνθηκαν από την απαίτηση συμπερίληψης γρίφων στις ιστορίες, όπως το παιχνίδι *'Galatea' (Emily Short, 2000*).

Σήμερα, η πλειοψηφία των νέων παιχνιδιών Interactive Fiction είναι δωρεάν διαθέσιμα. Με την έντονη ανάπτυξη της τεχνητής νοημοσύνης για την παραγωγή κειμένου, έχουν δημιουργηθεί συστήματα τα οποία μπορούν να δημιουργήσουν νέα κείμενα ανοιχτού τύπου δυναμικά. Ένα από τα χαρακτηριστικότερα παραδείγματα του είδους είναι το *AI Dungeon (2019),* το οποίο παράγει νέο περιεχόμενο με τη χρήση του GPT-2, ένα γλωσσικό μοντέλο για την παραγωγή κειμένων φυσικής γλώσσας που αναπτύχθηκε από την OpenAI [25]. Άλλα παραδείγματα σύγχρονων παιχνιδιών IF είναι το *Façade (2005)* [3] και το *Depression Quest (2014).*

2.2 Μηχανές ανάπτυξης εφαρμογών διαδικαστικής αφήγησης

Τις τελευταίες δεκαετίες έχουν αναπτυχθεί αρκετές πλατφόρμες για την ανάπτυξη διαδραστικών εφαρμογών και παιχνιδιών διαδικαστικής αφήγησης. Πολλές από τις πλατφόρμες αυτές είναι ανοιχτού λογισμικού (open-source) και επεκτείνονται συνεχώς από τις κοινότητες των χρηστών τους με βιβλιοθήκες, documentation κλπ. Κάποια παραδείγματα τέτοιων εφαρμογών δίνονται σε αυτήν την ενότητα.

2.2.1 TADS 3

Το *TADS* (Text Adventure Development System) [26] είναι ένα δωρεάν εργαλείο συγγραφής για εφαρμογές Interactive Fiction. Η πλατφόρμα προσφέρει ένα ολοκληρωμένο σύνολο από προγραμματιστικά εργαλεία μέσω μίας γλώσσας προγραμματισμού ειδικού σκοπού, η οποία βασίζεται στη μορφή των γλωσσών προγραμματισμού C++ και JavaScript. Τα εργαλεία του TADS οργανώνονται έτσι ώστε να είναι οικεία σε άτομα που είναι ήδη έμπειρα στον προγραμματισμό. Οι διαδικαστικές πτυχές της γλώσσας (συναρτήσεις, βρόχοι επανάληψης κλπ.) έχουν την ίδια μορφή με τα αντίστοιχα εργαλεία της C.

Λόγω της φύσης των προγραμμάτων IF, η οποία συνδυάζει τόσο κώδικα όσο και δεδομένα με τη μορφή δομημένων αντικειμένων, η αναπαράσταση των δεδομένων αυτών είναι ένα κεντρικό χαρακτηριστικό της γλώσσας [32]. Η ιδιαίτερη προσέγγιση όσον αφορά τον αντικειμενοστραφή προγραμματισμό καθιστά τη δημιουργία νέων αντικειμένων μία πιο απλή διαδικασία για τον προγραμματιστή, από ό,τι με τη χρήση μίας άλλης γλώσσας γενικού σκοπού. Επίσης, το TADS διαθέτει υποστήριξη για μη στατικές συμβολοσειρές, για κείμενο δηλαδή που αλλάζει ανάλογα με τις συνθήκες του παιχνιδιού. Τέλος, η πλατφόρμα περιέχει στην standard βιβλιοθήκη της πολλές μεθόδους για τη διαχείριση αντικειμένων και πράξεων απαραίτητων για ένα παιχνίδι IF, ενώ επιτρέπει την επέκτασή τους, είτε με τη συγγραφή κώδικα, είτε με την εγκατάσταση άλλων βιβλιοθηκών. Το TADS είναι εργαλείο ανοιχτού λογισμικού, και πολλές από τις λειτουργίες του χρησιμοποιήθηκαν ως οδηγός για την ανάπτυξη της εφαρμογής της παρούσας εργασίας.

2.2.2 Inform 7

Το Inform 7 [27] είναι μία δωρεάν εφαρμογή για τη δημιουργία εφαρμογών διαδραστικής φαντασίας (δεν είναι open source). Η γλώσσα προγραμματισμού που χρησιμοποιείται για την ανάπτυξη παιχνιδιών βασίζεται στην αγγλική (φυσική) γλώσσα, και ως εκ τούτου είναι ένα εργαλείο πιο φιλικό σε άτομα που δεν είναι εξοικειωμένα με τον προγραμματισμό. Το Inform συγκαταλέγεται συχνά μεταξύ των 100 πιο σημαντικών γλωσσών προγραμματισμού, σύμφωνα με το TIOBE Index.

Κάθε εφαρμογή Inform 7 αποτελείται από τον κόσμο του παιχνιδιού έτσι όπως δημιουργείται κατά την εκκίνησή του, καθώς και ένα πλήθος κανόνων που καθορίζουν το πώς ο παίκτης αλληλοεπιδρά με τον κόσμο αυτό. Για τη δημιουργία των αντικειμένων του παιχνιδιού ο προγραμματιστής καλείται να γράψει προτάσεις που ονομάζονται assertions. Κάθε αντικείμενο, ανάλογα με την κλάση στην οποία ανήκει κατά τη δημιουργία του, έχει ένα σύνολο συμπεριφορών, οι οποίες μπορούν να εμπλουτιστούν μέσω της συγγραφής κανόνων. Παρότι ο κώδικας δίνει την εντύπωση ότι αποτελεί κείμενο φυσικής γλώσσας, τα είδη προτάσεων τα οποία μπορούν να διατυπωθούν είναι περιορισμένα. Για περισσότερες πληροφορίες βλ. [33].

2.2.3 Twine

Το Twine [28] είναι ένα εργαλείο ανοιχτού λογισμικού για την ανάπτυξη διαδραστικών μη-γραμμικών ιστοριών. Η συγγραφή βασικών εφαρμογών με το Twine δεν απαιτεί οποιαδήποτε γνώση προγραμματισμού, αφού το εργαλείο χρησιμοποιεί ένα γραφικό περιβάλλον για την ανάπτυξη ιστοριών. Οι ιστορίες αυτές, βέβαια, μπορούν να επεκταθούν με τη χρήση προγραμματιστικών εργαλείων (CSS, JavaScript). Σαν αποτέλεσμα, το εργαλείο είναι πολύ φιλικό για δημιουργούς παιχνιδιών IF οι οποίοι δεν έχουν καμία εξοικείωση με τον προγραμματισμό. Σε αντίθεση με το TADS και το Inform, η διεπαφή του παίκτη με τον κόσμο του παιχνιδιού δε γίνεται μέσω κειμένου φυσικής γλώσσας, αλλά με τη χρήση υπερσυνδέσμων, οι οποίοι οδηγούν σε άλλα διασυνδεδεμένα αποσπάσματα κειμένων.

2.2.4 Ren'Py

Η Ren'Py [29] είναι μία μηχανή ανάπτυξης γραφικών μυθιστορημάτων (visual novels) βασισμένη σε γλώσσα Python. Τα εργαλεία που προσφέρει η γλώσσα μπορούν να επεκταθούν με την απευθείας συγγραφή κώδικα Python για τη δημιουργία πιο περίπλοκων εφαρμογών.

2.2.5 Quest

Το Quest [30] είναι ένα δωρεάν και open-source εργαλείο ανάπτυξης παιχνιδιών διαδραστικής φαντασίας, όπως text adventures και gamebooks. Η ανάπτυξη του παιχνιδιού μπορεί να γίνει είτε σε μορφή φυσικής γλώσσας ή με τη χρήση της γλώσσας προγραμματισμού της πλατφόρμας, η οποία διαχειρίζεται τις λειτουργίες στο παρασκήνιο, αλλά είναι προσβάσιμη και από τον προγραμματιστή.

2.2.6 Squiffy

Το Squiffy [31] είναι μία δωρεάν πλατφόρμα ανοιχτού λογισμικού για τη δημιουργία διαδραστικών παιχνιδιών πολλαπλών επιλογών με τη χρήση υπερσυνδέσμων.

3. ΘΕΩΡΗΤΙΚΟ ΥΠΟΒΑΘΡΟ

3.1 Ανάλυση κειμένων φυσικής γλώσσας - Parsing

Ένα από τα σημαντικότερα βήματα κατά την αλληλοεπίδραση ενός παίκτη με ένα παιχνίδι διαδραστικής φαντασίας είναι η συντακτική ανάλυση των εντολών του. Η συντακτική ανάλυση αυτή (που ονομάζεται syntactic parsing ή απλώς parsing) έχει ως στόχο την ταυτοποίηση των αντικειμένων του παιχνιδιού και του ρήματος/πράξης που βρίσκονται μέσα σε μία πρόταση φυσικής γλώσσας. Σε αυτήν την ενότητα αναλύονται κάποιες στρατηγικές parsing από τη βιβλιογραφία για γενικές εφαρμογές (όχι μόνο για παιχνίδια διαδραστικής φαντασίας).

3.1.1 Συντακτική ανάλυση - Syntactic Parsing

Συντακτική ανάλυση μίας πρότασης ονομάζεται η ταυτοποίηση της συντακτικής της δομής. Για την ανάλυση μίας φυσικής γλώσσας απαραίτητος είναι ένας ορισμός της (μέσω μίας γραμματικής και του λεξιλογίου της) και ένας μορφοτροπέας (transducer) που περιλαμβάνει κάποιους φορμαλισμούς για τον συντακτικό αναλυτή (parser) [45]. Μία πολύ γενική θεώρηση του parsing περιλαμβάνει την είσοδο του συστήματος (μία σειρά από λέξεις και σημεία στίξης), κάποια μοτίβα με τα οποία εξετάζεται το αν αντιστοιχεί η είσοδος (αυτά τα μοτίβα συνιστούν τη γραμματική του parser), έναν αλγόριθμο parsing ο οποίος καθορίζει τη διαδικασία αντιστοίχισης εισόδου-μοτίβων, και την έξοδο του συστήματος, η οποία είναι μία γραφική ή γραμμική αναπαράσταση της δομής της εισόδου. Μία τέτοια αναπαράσταση αποτελούν τα parse trees, δηλαδή δέντρα τα οποία αναπαριστούν την ιεραρχική συντακτική δομή μίας πρότασης.

Κάποιοι φορμαλισμοί γραμματικών οι οποίοι παρουσιάζονται στην βιβλιογραφία [45-47] είναι:

- Γραμματικές δομής φράσεων (phrase-structure grammars) ή τυπικές γραμματικές: Ένα τυπικό σύνολο κανόνων για την παραγωγή και ανάλυση συμβολοσειρών μίας συγκεκριμένης γλώσσας. Ειδικό υποσύνολο αυτών των γραμματικών αποτελούν οι γραμματικές ανεξάρτητες συμφραζόμενων (context-free grammars, CFGs), οι οποίες δεν καθορίζουν κάποια συνθήκη ή πλαίσιο με βάση το οποίο τα συστατικά στοιχεία μίας πρότασης μπορούν να συνδυαστούν. Εν ολίγοις, ένα μη-τερματικό σύμβολο σε μία CFG επανεγγράφεται με τους ίδιους κανόνες, ανεξάρτητα από τα σύμβολα που το περιβάλλουν. Κάποιοι μηχανισμοί των φυσικών γλωσσών, όπως η γραμματική συμφωνία λέξεων (π.χ. συμφωνία πτώσεων ή γραμματικού αριθμού) αναπαρίστανται δυσκολότερα με τέτοιες γραμματικές, οπότε και χρησιμοποιούνται επεκτάσεις (augmentations) στους κανόνες των CFGs [45].
- Επαυξημένα δίκτυα μεταβάσεων (Augmented Transition Networks): Αυτού του είδους τα δίκτυα αποτελούνται από κάποια αναδρομικά δίκτυα μεταβάσεων (Recursive Transition Networks) τα οποία συνδυάζονται με μονάδες μνήμης καλούμενες «καταχωρητές επαύξησης» (augmentation registers). Τα RTNs έχουν τους ίδιους περιορισμούς με τις CFGs όσον αφορά την κωδικοποίηση του context μία γλώσσας, και για αυτό χρησιμοποιούνται αυτού του είδους οι επεκτάσεις.
- Λογικές γραμματικές (Logic Grammars): Αναπαριστούν τους κανόνες μίας γλώσσας σαν λογικές προτάσεις, κατάλληλες για τον προγραμματισμό γλωσσών όπως η Prolog.
- Κατηγορικές γραμματικές (Categorical Grammars): Στις κατηγορικές γραμματικές κάθε λέξη του λεξιλογίου μίας γλώσσας αντιστοιχεί σε κάποιες κατηγορίες, ενώ οι κανόνες της γραμματικής υποδεικνύουν το πώς αυτού του είδους οι κατηγορίες μπορούν να συνδυαστούν μεταξύ τους για να δημιουργήσουν νέες κατηγορίες [45].

Όσον αφορά τις στρατηγικές ανάλυσης, μπορεί να γίνει μία διάκριση μεταξύ στρατηγικών Bottom-up και Top-down [45]. Στην πρώτη κατηγορία η ανάλυση ξεκινά από τις λέξεις της πρότασης προς ανάλυση, κατασκευάζοντας το συντακτικό δέντρο από τα φύλλα προς τη ρίζα,

ενώ το ανάποδο ισχύει για τη δεύτερη κατηγορία στρατηγικών. Και οι δύο αντιμετωπίζουν προβλήματα με ορισμένα είδη κανόνων, συγκεκριμένα κανόνες που περιέχουν το κενό σύμβολο 'ε' και κανόνες που περιέχουν αριστερή αναδρομή (left recursion) για τις bottom-up και top-down στρατηγικές αντίστοιχα. Μέθοδοι όπως η ανάλυση με οπισθοδρόμηση (backtracking), στην οποία κάθε πιθανή επιλογή ανάλυσης εξετάζεται «παράλληλα» σε έναν πίνακα, επιλύουν κάποια από τα παραπάνω προβλήματα.

3.1.2 Σημασιολογική ανάλυση - Semantic Parsing

Στη σημασιολογική ανάλυση ένα κείμενο φυσικής γλώσσας αντιστοιχίζεται σε μία σημασιολογική του αναπαράσταση ή λογική μορφή (Meaning Representation - MR ή Logical Form) [47]. Ένας φορμαλισμός ή γραμματική χρησιμοποιούνται για την αντιστοίχιση μίας εισόδου σε λογικές μορφές, για καθεμιά από τις οποίες ένα μοντέλο παράγει μία στατιστική κατανομή. Ο αναλυτής (parser) επιφορτίζεται με την επιλογή της πιο κατάλληλης λογικής μορφής για μία δεδομένη πρόταση, με βάση το μοντέλο που χρησιμοποιείται.

Τρόποι αναπαράστασης μίας γλώσσας σε λογική μορφή αποτελούν [47]:

- Φορμαλισμοί βασισμένοι στη Λογική, όπως η λογική πρώτης τάξης (First Order Logic).
- Φορμαλισμοί βασισμένοι σε γράφους, στους οποίους οι οντότητες ή οι πράξεις μίας πρότασης αποτελούν κόμβους και οι σημασιολογικές σχέσεις μεταξύ τους αποτελούν τις ακμές του γράφου.
- Γλώσσες προγραμματισμού.

Οι μέθοδοι για τη σημασιολογική ανάλυση μίας πρότασης μπορούν να ταξινομηθούν στις παρακάτω κατηγορίες:

- Συστήματα ανάλυσης βασισμένα σε κανόνες, τα οποία χρησιμοποιούνταν παλαιότερα για εφαρμογές ειδικού σκοπού με όχι και τόσο καλά αποτελέσματα [47].

- Στατιστικά συστήματα, με μοντέλα τα οποία προσπαθούν να ανακαλύψουν μία στατιστική κατανομή μέσω δειγμάτων εκπαίδευσης εισόδου (προτάσεις φυσικής γλώσσας) και εξόδου (λογικές αναπαραστάσεις).
- Διάφορες άλλες μέθοδοι μηχανικής μάθησης, εποπτευόμενες ή μη εποπτευόμενες [47].
- Μέθοδοι βασισμένες σε νευρωνικά δίκτυα [46]: Αντιμετωπίζουν το πρόβλημα ως πρόβλημα μηχανικής μετάφρασης (Machine Translation) μεταξύ μίας φυσικής γλώσσας και της σημασιολογικής της αναπαράστασης. Παράδειγμα ενός τέτοιου μοντέλου είναι το Seq2Seq, στο οποίο χρησιμοποιείται ένας κωδικοποιητής που δημιουργεί μία κρυφή αναπαράσταση από μία πρόταση φυσικής γλώσσας, και έναν αποκωδικοποιητή, ο οποίος χρησιμοποιεί την αναπαράσταση αυτή για να δημιουργήσει MRs. Ο κωδικοποιητής και ο αποκωδικοποιητής μπορεί να αποτελούνται από Recursive Neural Networks ή Transformers.
- Μέθοδοι που λαμβάνουν υπόψιν το context [46]. Όπως ορίζεται εδώ το context, αποτελεί ένα σύνολο από προτάσεις που περιβάλλουν το κείμενο προς ανάλυση και παρουσιάζουν κάποια νοηματική και δομική συνοχή μεταξύ τους. Αυτού του είδους οι μέθοδοι μπορεί να βοηθήσουν στην ακριβέστερη ανάλυση ελλειπτικών προτάσεων, στην επίλυση του προβλήματος της αναφορικότητας, στη μείωση του χώρου αναζήτησης και στην εξαγωγή πιο περίπλοκων συμπερασμάτων για το νόημα ενός κειμένου.

3.2 Διαδικαστική Παραγωγή

Τα συστήματα τα οποία είναι υπεύθυνα για τη διαδικαστική παραγωγή περιεχομένου ονομάζονται γεννήτορες (generators). Τέτοιου είδους συστήματα χρησιμοποιούνται σε πολλών τύπων εφαρμογές (παιχνίδια, προσομοιώσεις κλπ.) για την παραγωγή διαφορετικών ειδών αντικειμένων (στο εξής artifacts). Σε αυτήν την ενότητα παρουσιάζονται κάποιες από τις πιο γνωστές μεθόδους διαδικαστικής παραγωγής γενικότερα και διαδικαστικής παραγωγής κειμένου φυσικής γλώσσας ειδικότερα.

3.2.1 Γεννήτορες - Generators

Όπως αναλύεται από την Dr. Kate Compton στο πρώτο κεφάλαιο του [24], η δημιουργία ενός generator ξεκινά πάντα με τον καθορισμό των επιθυμητών και μη επιθυμητών ιδιοτήτων των artifacts τα οποία παράγει. Από τις ιδιότητες αυτές ορίζονται οι περιορισμοί και ο χώρος πιθανοτήτων για τα παραγόμενα αντικείμενα. Μία μέθοδος παραγωγής (generative method) θα πρέπει:

- Να εμπεριέχει κάποιου είδους αναπαράστασης/κανόνων σαν βάση γνώσης για τις πιθανές επιλογές κατά την παραγωγή ενός αντικειμένου.
- Να έχει μία βασική δομή για τα artifacts καθώς και παραλλαγές της δομής αυτής.
- Να μπορεί να εξετάσει το κατά πόσο ικανοποιούνται οι περιορισμοί οι οποίοι έχουν τεθεί για τα artifacts.

Η ταξινόμηση των μεθόδων διαδικαστικής παραγωγής για παιχνίδια παρουσιάζονται στα [1, 24] και έχει ως εξής:

- Γεννήτριες ψευδοτυχαίων αριθμών: Χρησιμοποιούν μηχανισμούς παραγωγής ακολουθιών φαινομενικά τυχαίων αριθμών από τον υπολογιστή, ώστε να επιλέξουν τυχαία μέσα από μία σειρά πιθανών artifacts ή κατανομών ιδιοτήτων.
- Παραμετρικές μέθοδοι: Ξεκινώντας από ένα artifact-πρότυπο, τέτοιου είδους μέθοδοι δημιουργούν διαφορετικά artifacts με το να αλλάζουν ελαφρώς ορισμένες παραμέτρους του.
- Γραμματικές: Η χρήση generative grammars, όπως αποκαλούνται, διευκολύνει στην αποτύπωση κάποιων βασικών - αναδρομικών συνήθως - κανόνων για το πώς δομείται ένα artifact.
- Χωρικοί αλγόριθμοι (spatial methods): Μία σημαντική τέτοια οικογένεια μεθόδων είναι το λεγόμενο tiling («ψηφιδοποίηση»), η διάσπαση δηλαδή ενός artifact σε πολλά ομοιόμορφα σε μέγεθος κομμάτια, τα οποία μπορούν να αναδιαταχθούν ελεύθερα και να γεμίσουν με ήδη έτοιμο περιεχόμενο.
- Φιλτράρισμα εικόνων (δεν αποτελεί αντικείμενο ενδιαφέροντος της παρούσας εργασίας).

- Προσομοίωση περίπλοκων συστημάτων: Σε αυτές τις μεθόδους περιλαμβάνονται τα κυτταρικά αυτόματα, η χρήση μοντελοποιημένων χαρακτήρων (agents) κλπ.
- Μέθοδοι τεχνητής νοημοσύνης: Μέθοδοι όπως οι γενετικοί αλγόριθμοι, τα τεχνητά νευρωνικά δίκτυα, και τα προγράμματα επίλυσης περιορισμών.

Οι γεννήτορες αποτελούν περίπλοκα συστήματα τα οποία ενίοτε δεν παράγουν επιθυμητά αποτελέσματα και αποτυγχάνουν με διάφορους τρόπους, είτε αισθητικά, είτε παράγοντας προκλητικό ή ηθικά προβληματικό περιεχόμενο, είτε αποτυγχάνοντας να τηρήσουν τους περιορισμούς τους οποίους θέτει ο σχεδιαστής τους. Ένα ακόμη πρόβλημα με τους γεννήτορες είναι ότι υπάρχει ο κίνδυνος της εμφάνισης του προβλήματος "Bowl of oatmeal", της παραγωγής, δηλαδή, artifacts τα οποία μπορεί τεχνικά να διαφέρουν μεταξύ τους, αλλά πρακτικά γίνονται αντιληπτά από τον άνθρωπο ως πανομοιότυπα.

3.2.2 Ψευδοτυχαίοι αριθμοί - Pseudorandom number generators

Η χρήση ψευδοτυχαίων αριθμών στη διαδικαστική παραγωγή είναι αρκετά εκτεταμένη. Ψευδοτυχαίοι ονομάζονται οι αριθμοί εκείνοι οι οποίοι παράγονται από τον υπολογιστή με τη χρήση ενός ντετερμινιστικού αλγορίθμου, ο οποίος δίνει την εντύπωση μίας πραγματικά τυχαίας δειγματοληψίας αριθμών από κάποια κατανομή. Τέτοιου είδους μέθοδοι συνήθως βασίζονται σε κάποιον ακέραιο αριθμό-σπόρο (seed) για την παραγωγή ακολουθιών αριθμών.

Επιθυμητές ιδιότητες τέτοιων συστημάτων είναι [50]:

1. Η καλή κατανομή των αριθμών της παραγόμενης ακολουθίας.
2. Η μεγάλη περίοδος της μεθόδου (ο αριθμός των στοιχείων της ακολουθίας που παράγονται μέχρι η ακολουθία να αρχίσει να επαναλαμβάνεται).
3. Η επαναληψιμότητα (reproducibility) των αριθμών που παράγονται. Αυτή η ιδιότητα έχει μεγάλη σημασία στη διαδικαστική παραγωγή, αφού υπάρχει η ανάγκη να επιστρέφεται πάντοτε η ίδια ακολουθία ψευδοτυχαίων αριθμών για έναν δεδομένο seed.
4. Οι μεγάλες disjoint ακολουθίες. Εδώ ως disjoint ορίζονται οι ακολουθίες αριθμών οι οποίες μπορούν να υπολογίζονται ανεξάρτητα σε ένα κατανεμημένο, για παράδειγμα, σύστημα.

5. Η φορητότητα (portability). Το σύστημα θα πρέπει να παράγει τα ίδια αποτελέσματα για όλα τα είδη υπολογιστικών μηχανών στα οποία χρησιμοποιείται.
6. Η αποτελεσματικότητα των υπολογισμών από άποψη μνήμης και χρόνου (ιδιαίτερα σημαντική ιδιότητα σε παλαιότερα συστήματα).

3.2.3 Διαδικαστική παραγωγή κειμένου φυσικής γλώσσας

Η διαδικαστική παραγωγή κειμένου φυσικής γλώσσας μπορεί να επιτευχθεί με διάφορους τρόπους. Κάποιοι από τους πιο διαδεδομένους είναι:

- Με τη χρήση ενός συστήματος Mark-Up, δηλαδή μέσω της σήμανσης ορισμένων κειμένων-προτύπων, τα οποία περιέχουν κενές θέσεις (slots) στις οποίες μπορούν να αντικατασταθούν λέξεις ή φράσεις, τυχαία, ή με βάση κάποιο στοιχείο του context [49]. Τέτοιου είδους συστήματα μπορούν να συνδυαστούν και με μεθόδους ετικετών, σύμφωνα με τις οποίες ομάδες στοιχείων περιεχομένου (λέξεις, φράσεις, templates κλπ.) συνοδεύονται από μεταδεδομένα (tags) τα οποία περιγράφουν κάποιες ιδιότητές τους. Αυτές οι ετικέτες μπορούν να χρησιμοποιηθούν για την ευκολότερη επιλογή μεταξύ μίας σειράς συμπληρωμάτων για κάποιο πρότυπο.
- Η παραγωγή κειμένου μέσω Context-Free Grammars (βλ. 3.1). Οι τυπικές γραμματικές είναι χρήσιμες για την παραγωγή κειμένου με αναδρομικούς κανόνες.
- Μέθοδοι παραγωγής με νευρωνικά δίκτυα [48]. Οι πιο διαδεδομένες μέθοδοι περιλαμβάνουν ένα ζεύγος δικτύων κωδικοποιητή (encoder) και αποκωδικοποιητή (decoder). Τέτοιου είδους συστήματα χρησιμοποιούνται ευρέως στη μηχανική μετάφραση και ονομάζονται και Sequence to Sequence (Seq2Seq) (βλ. 3.1.2). Χρησιμοποιούν μία ενδιάμεση κρυφή αναπαράσταση της εισόδου την οποία δημιουργεί ο κωδικοποιητής και «ερμηνεύει» ο αποκωδικοποιητής για να παράγει την έξοδο. Τα νευρωνικά δίκτυα που αξιοποιούνται πιο συχνά για εφαρμογές παραγωγής κειμένου είναι τα Recursive ή τα Convolutional Neural Networks, καθώς και άλλα δίκτυα τα οποία χρησιμοποιούν κάποιον μηχανισμό προσοχής (attention mechanism) για να διατηρήσουν την πληροφορία από προηγούμενες εισόδους ή εξόδους.

3.3 Μέθοδοι διαδικαστικής αφήγησης

Η παραγωγή μίας ιστορίας διαδικαστικά αποτελεί ένα από τα κύρια ζητήματα που απασχολούν έναν σχεδιαστή παιχνιδιών Interactive Fiction. Η διαδικαστική αφήγηση μπορεί να αφορά είτε τη δημιουργία του βασικού σκελετού μίας ιστορίας (αλληλουχία γεγονότων), ή την διαδικαστική παραγωγή γραπτών αποσπασμάτων (περιγραφών, διαλόγων κλπ.). Επιπλέον, μπορούμε να διακρίνουμε δύο επίπεδα σε κάθε ιστορία τα οποία μπορούν να παραχθούν διαδικαστικά: Το επίπεδο περιεχομένου (content plane) το οποίο περιέχει την λογική αναπαράσταση των αντικειμένων και των γεγονότων μίας ιστορίας, και το επίπεδο έκφρασης (expression plane) το οποίο περιλαμβάνει την περιγραφή της ιστορίας. Αυτά τα δύο πεδία μπορούν να παραχθούν είτε αυτόνομα, είτε συνδυαστικά [41]. Για την διαδικαστική παραγωγή ιστοριών υπάρχει στη βιβλιογραφία και σε παιχνίδια μία πληθώρα από μεθόδους, κάποιες από τις οποίες παρουσιάζονται παρακάτω.

3.3.1 Διαδικαστική αφήγηση με τη χρήση τυπικών γραμματικών

Οι τυπικές γραμματικές αποτελούνται από ένα λεξιλόγιο συμβόλων (φωνήματα, λέξεις, φράσεις κλπ.) και από ένα σύνολο κανόνων (κανόνες επανεγγραφής, rewrite rules) οι οποίοι καθορίζουν τους τρόπους με τους οποίους τα σύμβολα του λεξιλογίου συνδυάζονται μεταξύ τους. Κάθε κανόνας επανεγγραφής έχει μία κεφαλή (οι οποία εκφράζει την αρχική ακολουθία συμβόλων) η οποία ακολουθείται το σώμα του κανόνα (την τελική σειρά συμβόλων) ([2], Κεφάλαιο 18).

Στην προσέγγιση που αναλύεται στο Κεφάλαιο 16 του [24], οι τυπικές γραμματικές χρησιμοποιούνται για την παραγωγή περιγραφών στο παιχνίδι *Voyageur* (2016, Bruno Dias). Για τις ανάγκες του παιχνιδιού δημιουργήθηκε η πλατφόρμα ανάπτυξης Improv, η οποία επιτρέπει τη συγγραφή κανόνων γραμματικής που λαμβάνουν υπόψιν τους την κατάσταση του κόσμου του παιχνιδιού. Κάθε κεφαλή κανόνα αντιστοιχεί σε σώματα κανόνα τα οποία χωρίζονται σε κατηγορίες ανάλογα με το περιεχόμενό τους. Κάθε τέτοια ομάδα σωμάτων κανόνα περιέχει κάποια μεταδεδομένα (metadata) που την περιγράφουν, και τα οποία αντιστοιχούν σε πιθανές καταστάσεις του κόσμου του παιχνιδιού.

Κατά την επανεγγραφή ενός συμβόλου, μία διαδικασία φιλτραρίσματος χρησιμοποιείται για να αναθέσει κάποια μετρική καταλληλότητας σε κάθε σώμα του κανόνα. Κάποια από τα κριτήρια φιλτραρίσματος είναι το κατά πόσο έχει ήδη χρησιμοποιηθεί ένα συγκεκριμένο συμπλήρωμα κανόνα και πόσο ταιριάζουν τα μεταδεδομένα του στην κατάσταση του κόσμου τη δεδομένη χρονική στιγμή. Το μοντέλο του κόσμου δημιουργείται με τον συνδυασμό δύο στρατηγικών, την εκ των προτέρων (a priori) και εκ των υστέρων (a posteriori) πραγματικότητα. Στην πρώτη περίπτωση το μοντέλο του κόσμου δημιουργείται προτού επιλεγεί οποιαδήποτε περιγραφή. Αντίθετα, στη δεύτερη περίπτωση, επιλέγεται πρώτα τυχαία ο πρώτος κανόνας επανεγγραφής, και τα μεταδεδομένα του ανατροφοδοτούνται στο μοντέλο του κόσμου, καθορίζοντας έτσι και τους επόμενους κανόνες που θα επιλεγούν.

Μία άλλη μέθοδος που χρησιμοποιεί τυπικές γραμματικές γράφων (graph grammars) για την κατασκευή της πλοκής μίας ιστορίας παρουσιάζεται από τον Ben Kybartas στο Κεφάλαιο 18 του [2]. Σε αυτή τη μέθοδο, οι ιστορίες δομούνται γύρω από τις σχέσεις μεταξύ χαρακτήρων. Το μοντέλο του κόσμου ξεκινάει με τη δημιουργία ενός κατευθυνόμενου γράφου ο οποίος καλείται κοινωνικό δίκτυο (social network), όπου κάθε κόμβος αποτελεί έναν χαρακτήρα και κάθε ακμή τη σχέση ενός χαρακτήρα με κάποιον άλλον. Η ιστορία δομείται επίσης ως κατευθυνόμενος γράφος, στον οποίο κάθε τμήμα της αφήγησης είναι ένας κόμβος, και οι ακμές υποδηλώνουν διαφορετικά μονοπάτια τα οποία μπορεί να επιλέξει ο παίκτης.

Οι γραμματικές γράφων έχουν την ίδια δομή με τις τυπικές γραμματικές συμβολοσειρών, με τη διαφορά ότι κάθε κανόνας επανεγγραφής έχει τη μορφή:

[κόμβος γράφου] → [τμήμα γράφου]

Για τη δημιουργία μίας ιστορίας διαδικαστικά διακρίνονται δύο τύποι κανόνων, οι αρχικοί και οι δευτερεύοντες κανόνες επανεγγραφής. Οι αρχικοί κανόνες επανεγγραφής δείχνουν το πώς μπορεί να δομηθεί ο βασικός κορμός της ιστορίας, ενώ οι δευτερεύοντες κανόνες συγκεκριμενοποιούν γεγονότα μέσα στον γενικότερο σκελετό της αφήγησης. Για την επιλογή των κατάλληλων κανόνων επανεγγραφής λαμβάνεται υπόψιν το κοινωνικό δίκτυο που ορίστηκε αρχικά, με τρόπο τέτοιον ώστε να σχηματίζονται συγκρούσεις μεταξύ των χαρακτήρων της ιστορίας.

3.3.2 Μέθοδοι διαδικαστικής αφήγησης βασισμένες σε αλληλοεπιδράσεις αυτόνομων χαρακτήρων (agents)

Όπως ειπώθηκε και παραπάνω, μία ιστορία μπορεί να θεωρηθεί ως ένα σύνολο σχέσεων μεταξύ χαρακτήρων και το πώς οι σχέσεις αυτές μεταβάλλονται στον χρόνο. Ένα σύστημα το οποίο βασίστηκε σε αυτήν την αρχή για την παραγωγή ενός αφηγήματος είναι το Thespian [42]. Το Thespian είναι ένα σύστημα εκπαίδευσης αυτόνομων χαρακτήρων (agents) σε παιχνίδια διαδραστικής φαντασίας. Η εκπαίδευση γίνεται έτσι ώστε οι χαρακτήρες αυτοί να δρουν με βάση μίας προγραμματιζόμενης προσωπικότητας.

Κάθε agent διαθέτει μία υποκειμενική θεώρηση του κόσμου του παιχνιδιού, την οποία αξιοποιεί για να πάρει αποφάσεις για το ποια είναι η κατάσταση του περιβάλλοντός του, η κατάσταση των υπολοίπων χαρακτήρων και με ποιο τρόπο μπορεί να τις επηρεάσει. Κάθε χαρακτήρας έχει κάποιους στόχους με τη μορφή μίας συνάρτησης επιβράβευσης (reward function). Οι agents μπορούν έτσι να αναλύσουν το ποια είναι η καλύτερη απόφαση που μπορούν να πάρουν σε μία δεδομένη χρονική στιγμή, λαμβάνοντας υπόψιν και τις πιθανές επόμενες κινήσεις των υπολοίπων χαρακτήρων μέσα στο παιχνίδι.

3.3.3 Εξόρυξη πλοκής (plot mining)

Ένας άλλος μηχανισμός για την παραγωγή αφήγησης παρουσιάζεται στο [43] όπου γίνεται χρήση ελεύθερα προσβάσιμων δεδομένων στο διαδίκτυο για την δημιουργία σεναρίων. Πιο συγκεκριμένα, χρησιμοποιούνται άρθρα της ηλεκτρονικής βιβλιοθήκης Wikipedia, μέσω της DBpedia (στην οποία αποθηκεύονται δεδομένα από την Wikipedia με δομημένο τρόπο), ώστε να βρεθούν σύνδεσμοι μεταξύ 2 τυχαίων υπαρκτών προσώπων. Οι σύνδεσμοι αυτοί είναι άλλα άρθρα της Wikipedia, για πρόσωπα, τοποθεσίες ή άλλες κατηγορίες συσχέτισης. Αυτές οι διασυνδέσεις μετατρέπονται σε οντότητες του παιχνιδιού οι οποίες συνδέονται μεταξύ τους με στοιχεία (clues), τα οποία επίσης δημιουργούνται διαδικαστικά.

Για την παραγωγή μίας ιστορίας συντελούνται τα ακόλουθα βήματα:

- Συλλογή των δεδομένων από την DBpedia για την εύρεση σύνδεσης μεταξύ δύο ατόμων. Η ποιότητα των συνδέσεων οι οποίες εξορύσσονται καθορίζεται με βάση

το μήκος των διασυνδέσεων και τη μοναδικότητα των συσχετίσεων μεταξύ των κόμβων-άρθρων.

- Μετατροπή των άρθρων-κόμβων σε Non-Player Characters, αντικείμενα και τοποθεσίες στον κόσμο του παιχνιδιού.
- Δημιουργία στοιχείων (clues) τα οποία επιτρέπουν στον παίκτη να ανακαλύψει τον επόμενο κόμβο στην αλυσίδα. Πέρα από τα πραγματικά στοιχεία, προστίθενται και άλλα ψευδή στοιχεία, ώστε να δημιουργηθούν σημεία στην ιστορία στα οποία ο παίκτης μπορεί να ακολουθήσει κάποιο λάθος μονοπάτι στην αφήγηση.

3.3.4 Μηχανική μάθηση και παραγωγή αφήγησης - Η περίπτωση του AI Dungeon 2

Τα τελευταία χρόνια, με την άνοδο των τεχνητών νευρωνικών δικτύων και άλλων τεχνικών μηχανικής μάθησης, νέα συστήματα διαδικαστικής αφήγησης έχουν αναπτυχθεί με βάση αυτές τις τεχνολογίες. Ένα από τα πιο δημοφιλή εξ αυτών είναι το AI Dungeon 2, το οποίο αποτελεί ένα εργαλείο για τη δημιουργία «απείρως» παραμετροποιήσιμων παιχνιδιών τύπου Adventure [44]. Το AI Dungeon 2 μπορεί να κατασκευάσει μία ιστορία διαδραστικής φαντασίας χρησιμοποιώντας ένα αρχικό κείμενο, με τον παίκτη να έχει τη δυνατότητα να δώσει εντολές για τη συνέχεια της ιστορίας σε μορφή ελεύθερου κειμένου φυσικής γλώσσας. Δεν υπάρχει κανένας περιορισμός στο τι μπορεί να εισάγει ο παίκτης ως εντολή-πρόταση, και το κείμενο το οποίο παράγεται είναι αρκετά καλής ποιότητας.

Το AI Dungeon 2 βασίζεται στο Γλωσσικό Μοντέλο (Language Model) GPT-2, το οποίο περιέχει 1.5 δισεκατομμύρια παραμέτρους και έχει εκπαιδευτεί σε 40 GB αγγλικού κειμένου. Τα γλωσσικά μοντέλα είναι συστήματα που πραγματοποιούν πιθανοτικούς υπολογισμούς σε ακολουθίες λέξεων βασισμένα στα δεδομένα εκπαίδευσής τους, για την πρόβλεψη της συνέχειας ενός κειμένου. Για την δημιουργία του AI Dungeon 2 από το μοντέλο GPT-2 έγινε fine-tuning (ρύθμιση παραμέτρων ενός ήδη εκπαιδευμένου μοντέλου) με ένα μεγάλο σύνολο κειμένων από παιχνίδια Choose-Your-Own-Adventure [44].

3.4 Δυναμικοί διάλογοι

Οι διάλογοι σε ένα παιχνίδια διαδραστικής φαντασίας αποτελούν μία από τις μεγαλύτερες προκλήσεις. Η τεχνολογική πρόοδος που έχει σημειωθεί στον τομέα δεν επαρκεί για τον προγραμματισμό χαρακτήρων που να μπορούν να συνδιαλέγονται με έναν ανθρώπινο παίκτη με την απαραίτητη φυσικότητα. Για αυτό το λόγο, το σημαντικότερο για ένα παιχνίδι με δυναμικούς διαλόγους είναι να δημιουργήσει την εντύπωση (και όχι την ψευδαίσθηση) ότι ο παίκτης μιλάει σε ένα νοήμον σύστημα [35].

Ως ορισμός των δυναμικών διαλόγων, στα πλαίσια της παρούσας εργασίας, χρησιμοποιείται αυτός του Elan Ruskin στο κεφάλαιο 25 του [24]: Δυναμικοί ορίζονται οι διάλογοι όπου οι απαντήσεις των Non-Player Characters (NPCs) προσαρμόζονται στις πράξεις του παίκτη και σε προηγούμενα γεγονότα. Ως εκ τούτου, ένας μηχανισμός παραγωγής δυναμικών διαλόγων θα πρέπει να λαμβάνει υπόψιν την κατάσταση του κόσμου του παιχνιδιού και το ιστορικό γεγονότων και πράξεων που έχουν λάβει χώρα. Επιπλέον, ένας NPC θα πρέπει να μπορεί να προσαρμόσει τις απαντήσεις του από τις πιο γενικές στις πιο ειδικές, ανάλογα με τις συνθήκες, και να επιλέξει την καταλληλότερη απάντηση σύμφωνα με τα συμφραζόμενα.

Προτού ο σχεδιαστής ενός παιχνιδιού υλοποιήσει ένα σύστημα δυναμικών διαλόγων, θα πρέπει να αποφασίσει τον σκοπό που έχουν οι NPCs στον κόσμο του παιχνιδιού, και τη χρησιμότητα που έχουν για τον παίκτη και τις ανάγκες της πλοκής. Η ταυτότητα του NPC μπορεί να επηρεάσει το είδος των διαλόγων στους οποίους μπορεί να συμμετέχει, αλλά ειδική μέριμνα θα πρέπει να δοθεί επίσης στο τι είδους πληροφορίες πρέπει να παρέχονται μέσω των διαλόγων. Ένα σύστημα που ο μόνος στόχος του είναι η μετάδοση ορισμένων στοιχείων της πλοκής στον παίκτη διαφέρει αρκετά από ένα άλλο στο οποίο η σχέση του παίκτη με τους NPCs είναι το κεντρικό σημείο της ιστορίας.

3.4.1 Είδη συστημάτων διαλόγου

Ανάλογα με τον τρόπο λειτουργίας ενός συστήματος δυναμικών διαλόγων, και τη μέθοδο εισαγωγής των εντολών/απαντήσεων του παίκτη, μπορούμε να διακρίνουμε κάποια από τα παρακάτω είδη μηχανισμών:

- "Cheap" Artificial Intelligence: Με τη χρήση ορισμένων κλασσικών τεχνικών από την επεξεργασία φυσικής γλώσσας, όπως ταίριασμα λέξεων-κλειδιών ή ταίριασμα κανονικών εκφράσεων (regular expressions), μπορεί να επιτραπεί στον παίκτη να εισάγει κάποιο κείμενο διαλόγου σε φυσική γλώσσα. Τέτοιου είδους συστήματα (όπως παραδείγματος χάριν από το παιχνίδι *Starship Titanic*, ή από το σύστημα *Eliza*) επιτρέπουν στον παίκτη μεγάλη ελευθερία έκφρασης, αλλά συνήθως αποτυγχάνουν στο να τον πείσουν για τις ικανότητες της εφαρμογής να κατανοήσει τις προτάσεις του. Συγκεκριμένα, τα συστήματα αυτά αντιμετωπίζουν προβλήματα με ομόγραφες λέξεις, με τα συμφραζόμενα των προτάσεων του παίκτη (πέρα από τις λέξεις κλειδιά) και με την προσομοίωση της δενδροειδούς δομής μίας πραγματικής συζήτησης [35].
- Συζητήσεις κλειστού τύπου (yes/no conversations): Αυτό το είδος συστήματος δυναμικών διαλόγων επιτρέπει στον παίκτη να απαντάει μονολεκτικά (συνήθως με ναι ή όχι) σε ερωτήσεις των NPCs. Ένα παράδειγμα αποτελούν τα παιχνίδια του Andrew Plotkin, *Spider* και *Web* [37].
- Συζητήσεις Talk to: Ένας μηχανισμός talk to επιτρέπει στον παίκτη μόνο να επιλέξει το πότε και με ποιον NPC επιθυμεί να συνομιλήσει. Η συζήτηση που ακολουθεί είναι προκαθορισμένη και από την πλευρά του χαρακτήρα του παίκτη και από την πλευρά των απαντήσεων των άλλων χαρακτήρων. Αυτό επιτρέπει στον σχεδιαστή του παιχνιδιού να έχει πλήρη έλεγχο στα περιεχόμενα των διαλόγων, αλλά αφαιρεί από τον παίκτη την ελευθερία του καθορισμού των θεμάτων προς συζήτηση [35].
- Συζητήσεις βασισμένες σε κατάλογο επιλογών (menu-based conversation) και απαντήσεις υπερσυνδέσμων: Ο παίκτης μπορεί να απαντήσει σε έναν NPC επιλέγοντας μία από τις επιλογές που του δίνονται, είτε με τη μορφή ενός καταλόγου επιλογών, είτε με τη μορφή λέξεων στο κείμενο, οι οποίες οδηγούν σε άλλα

κομμάτια διαλόγου. Ο σχεδιαστής του παιχνιδιού έχει και εδώ τον έλεγχο του περιεχομένου των διαλόγων, αλλά αφαιρείται από τον παίκτη σε μεγάλο βαθμό η ελευθερία του.

- Συστήματα Ask/Tell: Ένας μηχανισμός διαλόγων ο οποίος επιτρέπει στον παίκτη να εισάγει κάποιο θέμα (Topic) προς συζήτηση σε κάποιον NPC, ώστε να πάρει μία σχετική απάντηση. Ο μηχανισμός αυτός αναλύεται εκτενέστερα στην επόμενη υποενότητα.

3.4.2 Διάλογοι τύπου Ask/Tell και στρατηγικές βελτίωσής τους

Ο μηχανισμός Ask/Tell μπορεί να ενσωματωθεί σε ένα παιχνίδι τύπου Parser, με τη δημιουργία ειδικών λέξεων-θεμάτων (topics), δίνοντας έτσι στον παίκτη μία αίσθηση ελευθερίας ως προς το τι μπορεί να εισάγει σαν ερώτηση. Παρότι τέτοιου είδους συστήματα ταιριάζουν στη φιλοσοφία των περισσότερων παιχνιδιών διαδραστικής φαντασίας, στην πράξη δημιουργούν ορισμένα προβλήματα για τον παίκτη και τον σχεδιαστή του παιχνιδιού [37].

Καταρχάς, οι διάλογοι που αναπτύσσονται με αυτό το μοντέλο δεν επηρεάζονται από τα συμφραζόμενα και δεν έχουν τη δομή μίας τυπικής συζήτησης (με χαιρετισμό στην αρχή και στο τέλος). Επίσης, στην πιο απλή εκδοχή ενός διαλογικού συστήματος Ask/Tell, μόνο η απάντηση του NPC εκτυπώνεται σαν πλήρης πρόταση, και όχι η ερώτηση του παίκτη, η οποία υπονοείται μόνο μέσω της εντολής του. Τέλος, δεν δίνεται η ευκαιρία στον παίκτη να απαντήσει με κάποιον τρόπο στις προτάσεις των NPCs, παρά μόνο να ρωτήσει για κάποιο άλλο θέμα. Ορισμένες στρατηγικές [35] για την επίλυση αυτών των προκλήσεων είναι:

- Εκτύπωση του ολοκληρωμένου σχήματος ερώτησης του Player Character και των απαντήσεων των NPCs.
- Η υλοποίηση πρωτοκόλλων χαιρετισμών και αποχαιρετισμών για πιο ολοκληρωμένες συζητήσεις με αρχή και τέλος.
- Δημιουργία «αλυσίδων» διαλόγου, δίνοντας στον παίκτη νύξεις για να ρωτήσει για κάποιο συγκεκριμένο θέμα, μέσα από φράσεις-κλειδιά στις απαντήσεις του NPC.
- Η παροχή της δυνατότητας στον παίκτη να δει έναν κατάλογο με τα διαθέσιμα θέματα που έχει προς συζήτηση σε μία δεδομένη στιγμή με έναν συγκεκριμένο χαρακτήρα.

- Να δοθεί η δυνατότητα στους NPCs να κάνουν ερωτήσεις στις οποίες ο παίκτης μπορεί να απαντήσει με κάποιον τρόπο.
- Να δοθεί βάρος στην ανάπτυξη καλών προεπιλεγμένων απαντήσεων, σε περίπτωση που ένας NPC δεν μπορεί να απαντήσει σε ένα θέμα το οποίο εγείρει ο παίκτης. Είναι σημαντικό να μην χάνεται η εντύπωση της συζήτησης, αλλά ούτε και να δοθεί στον παίκτη η εντύπωση ότι κάτι σημαντικό για την υπόθεση του παιχνιδιού κρύβεται πίσω από μία τέτοια απάντηση.

3.4.3 Προγραμματίζοντας ένα σύστημα δυναμικών διαλόγων

Στην αναφορά [37] παρουσιάζονται ορισμένες στρατηγικές για την υλοποίηση ενός συστήματος δυναμικών διαλόγων. Τα βασικά στοιχεία ενός συστήματος δυναμικών διαλόγων είναι:

- Τα θέματα (topics) στα οποία μπορεί να αναφερθεί ο παίκτης.
- Τα γεγονότα (facts), τα οποία αποτελούν προτάσεις για το περιεχόμενο ενός θέματος.
- Οι ατάκες (quips), οι οποίες αποτελούν τη ρητή έκφραση των γεγονότων, έτσι όπως τυπώνονται προς τον Player Character.
- Οι συνέπειες (effects) που προκαλούνται από την αναφορά κάποιου θέματος (όπως για παράδειγμα αλλαγές στη συναισθηματική κατάσταση ενός NPC).
- Οι στόχοι διαλόγου (conversation goals), οι οποίοι συνοψίζουν το τι θέλουν να πετύχουν οι χαρακτήρες του παιχνιδιού μέσα από συγκεκριμένες απαντήσεις.
- Οι σκηνές (scenes) οι οποίες αποτελούν συγκεκριμένα τμήματα της πλοκής.

Ως εκ τούτου μπορούν να οριστούν τριών ειδών μοντέλα διαλόγου, σύμφωνα με την Emily Short [37]: Το topic quip μοντέλο, στο οποίο κάθε θέμα συσχετίζεται ένα προς ένα με μία απάντηση, το quip tree μοντέλο, πιο ταιριαστό για συστήματα διαλόγου με μενού επιλογών, όπου δημιουργείται ένα δέντρο από πιθανές απαντήσεις, και το scene quip μοντέλο, ιδανικό για talk to συστήματα, όπου οι απαντήσεις εξαρτώνται μόνο από την σκηνή στην οποία μπορούν να ειπωθούν. Αυτά τα τρία συστήματα μπορούν να επεκταθούν με διάφορους τρόπους, όπως με την προσθήκη κάποιας μεθόδου η οποία καταγράφει τις συναισθηματικές

μεταπτώσεις των χαρακτήρων κατά τη διάρκεια της συζήτησης για να προσαρμόσει τις απαντήσεις τους. Ακόμα, μπορεί να υλοποιηθούν θέματα με τα οποία σχετίζονται παραπάνω από μία απαντήσεις.

Στο [36] γίνεται αναφορά και στα δέντρα διαλόγου. Τα δέντρα διαλόγου αποτελούνται από κόμβους, καθένας από τους οποίους περιέχει ένα σύνολο από θέματα και απαντήσεις. Ένας NPC σχετίζεται με κάποιον συγκεκριμένο κόμβο για κάθε χρονική στιγμή. Με τη χρήση των δέντρων διαλόγου μπορούν να μοντελοποιηθούν κάποιες συναισθηματικές μεταπτώσεις των χαρακτήρων, καθώς και πιο περίπλοκες συζητήσεις, οι οποίες κατευθύνουν τον παίκτη προς συγκεκριμένες ερωτήσεις και συμπεράσματα.

Τέλος, στο κεφάλαιο 25 του [24] γίνεται μία εκτενής αναφορά στον τρόπο υλοποίησης ενός συστήματος δυναμικών διαλόγων με τη χρήση κανόνων και απαντήσεων. Σε ένα τέτοιο μοντέλο, το γενικό πλαίσιο (context) στο οποίο γίνεται η συζήτηση μπορεί να αλλάξει το σύνολο των απαντήσεων που δίνονται. Κάθε χαρακτήρας σε ένα τέτοιο μοντέλο διαθέτει έναν πίνακα γνώσης-κατάστασης, και το ίδιο ισχύει για τον κόσμο του παιχνιδιού. Αυτοί οι πίνακες γνώσης περιγράφουν γεγονότα για το παιχνίδι, βάση των οποίων μπορεί να γίνει η επιλογή των απαντήσεων στους διαλόγους. Κάθε ένα τέτοιο γεγονός καλείται context (ουσιαστικά ένα ζεύγος κλειδιού-τιμής), ενώ μία σύγκριση της τιμής ενός context με μία τιμή-στόχο καλείται κριτήριο (criterion). Ένας κανόνας ορίζεται ως ένα σύνολο από κριτήρια, τα οποία πρέπει να ικανοποιηθούν στο σύνολό τους για να θεωρηθεί ότι ισχύει. Μία ερώτηση (query) στον κόσμο του παιχνιδιού, αποτελεί μία συλλογή από contexts από όλο το παιχνίδι, ενώ η απάντηση δεν είναι παρά το αποτέλεσμα ενός ισχύοντος κανόνα. Με αυτό το σύστημα, όταν ο παίκτης αναφέρεται σε κάποιο θέμα, ή όταν συμβαίνει κάποιο άλλο γεγονός, αναζητούνται όλοι οι κανόνες του παιχνιδιού οι οποίοι ισχύουν, και τελικά τυπώνεται η απάντηση του κανόνα με τα περισσότερα κριτήρια (καθώς θεωρείται ο πιο συγκεκριμένος).

3.4.4 Σύγχρονα συστήματα δυναμικών διαλόγων

Τα τελευταία χρόνια, ορισμένα παιχνίδια έχουν χρησιμοποιήσει πιο περίπλοκα συστήματα διαλόγων ώστε να επιτρέψουν στον χρήστη να συνδιαλέγεται με χαρακτήρες του παιχνιδιού με κείμενα φυσικής γλώσσας. Ένα από αυτά είναι το παιχνίδι *Façade (Playabl Studios, 2005)* [38] το οποίο αποτελεί ένα «διαδραστικό δράμα» που χρησιμοποιεί διάφορες τεχνικές

επεξεργασίας φυσικής γλώσσας για να αναλύσει συντακτικά το κείμενο που παρέχεται από τον παίκτη. Ο τελευταίος μπορεί να συζητήσει με δύο χαρακτήρες τεχνητής νοημοσύνης. Παρότι το Façade χρησιμοποιεί πολλές από τις «κλασσικές» τεχνικές parsing που αναφέρθηκαν παραπάνω ως "cheap AI", όπως keyword matching, ο μεγάλος αριθμός μοτίβων προτάσεων που μπορεί να αναγνωρίσει καθιστούν το αποτέλεσμα εντυπωσιακό. Ως εκ τούτου, το παιχνίδι θεωρήθηκε πρωτοποριακό για την εποχή του.

Δύο ακόμα παιχνίδια - *A Family Supper* και *Blood and Laurels,* Emily Short και Richard Evans - τα οποία δημιουργήθηκαν το 2014 με τη χρήση του εργαλείου Versu [39], αποτελούν χαρακτηριστικά παραδείγματα δυναμικών NPCs. Οι χαρακτήρες των παιχνιδιών μοντελοποιούνται με τη βοήθεια του Versu και πραγματοποιούν δράσεις και εκτός οθόνης, δίνοντας την εικόνα ενός ζωντανού κόσμου.

Τέλος, το *Event[0]* (game, Ocelot Society, 2016) [40] περιέχει τον Kaizen-85, έναν υπολογιστή με τον οποίο ο παίκτης μπορεί να συζητάει, δίνοντάς του απαντήσεις σε κείμενο φυσικής γλώσσας. Ένα μεγάλο σύνολο από λέξεις κλειδιά και γραμματικά σχήματα αναγνωρίζονται από το παιχνίδι, και σε συνδυασμό με ένα μοντέλο 9 συναισθηματικών καταστάσεων του υπολογιστή και το γενικό πλαίσιο της σκηνής στην οποία βρίσκεται ο παίκτης, επιλέγεται η ιδανικότερη απάντηση. Επιπλέον ποικιλία στις απαντήσεις δίνεται με την αντικατάσταση συνώνυμων φράσεων σε πρότυπα απαντήσεων.

3.5 Αντικειμενοστραφής προγραμματισμός - Object Oriented Programming

Ο Αντικειμενοστραφής Προγραμματισμός (Object Oriented Programming, ή OOP) αποτελεί μία θεώρηση του κόσμου ως αποτελούμενου από ένα σύνολο οντοτήτων, οι οποίες αλληλοεπιδρούν μεταξύ τους και με το περιβάλλον τους. Οι οντότητες μπορεί να είναι απλές ή σύνθετες. Κάθε οντότητα μπορεί να γενικευθεί σε μία αφηρημένη έννοια που περιγράφει όλες τις οντότητες οι οποίες μοιράζονται κοινά χαρακτηριστικά.

3.5.1 Η έννοια της κλάσης και του αντικειμένου

Κλάση στον OOP ονομάζεται μία αφηρημένη έννοια που περιγράφει τα κοινά χαρακτηριστικά ενός συνόλου ομοειδών αντικειμένων, τα οποία ονομάζονται και στιγμιότυπα (instances) της κλάσης [34]. Υπό αυτή τη θεώρηση, κάθε προγραμματιστικό σύστημα μπορεί να αντιμετωπιστεί ως μία συνάθροιση αντικειμένων τα οποία συνδέονται μεταξύ τους με σχέσεις γενίκευσης - εξειδίκευσης και συνάθροισης, ενώ επικοινωνούν μεταξύ τους μέσω μηνυμάτων.

Κάθε αντικείμενο διαθέτει ένα σύνολο από μεταβλητές (ιδιότητες ή attributes) και ρουτίνες (μέθοδοι ή methods). Κάποια από τα χαρακτηριστικά αυτά κάθε κλάσης δεν είναι ορατά σε όλα τα υπόλοιπα αντικείμενα του συστήματος (και ονομάζονται ιδιωτικά ή private) σε αντίθεση με τις ιδιότητες και τις μεθόδους που είναι προσβάσιμες και έξω από τα αντικείμενα της κλάσης (public). Η έννοια αυτή των ιδιωτικών και δημόσιων χαρακτηριστικών ονομάζεται απόκρυψη πληροφορίας (information hiding). Τα αντικείμενα επομένως μπορεί να θεωρηθεί ότι αποτελούνται από δύο παράλληλα συστήματα, την υλοποίηση του αντικειμένου (object implementation) η οποία διαχειρίζεται τις κρυφές του λειτουργίες, και την διεπαφή του αντικειμένου (object interface) η οποία διαχειρίζεται την επικοινωνία του αντικειμένου με το περιβάλλον του [34]. Αυτή η φιλοσοφία σχεδιασμού των αντικειμένων ονομάζεται και ενθυλάκωση πληροφορίας (data encapsulation).

Κάποια από τα επιθυμητά χαρακτηριστικά των συστημάτων τα οποία βελτιώνονται με τη χρήση της αντικειμενοστραφούς προσέγγισης είναι:

- Η επαναχρησιμοποίηση (reusability) κομματιών κώδικα.
- Η επεκτασιμότητα (extensibility) του συστήματος.
- Η ορθότητα του συστήματος (correctness).
- Η ευρωστία του συστήματος (robustness).
- Η συντηρησιμότητα (maintainability) του κώδικα.

3.5.2 Σύνδεσμοι και μηνύματα μεταξύ αντικειμένων

Τα επιμέρους αντικείμενα συνεργάζονται μεταξύ τους για την παροχή των εξυπηρετήσεων του συστήματος. Όπως ορίζεται στο Κεφάλαιο 7 του [34], τα αντικείμενα ενός συστήματος μπορούν να συνεργάζονται με διαφορετικές σχέσεις -φυσικές ή ιδεατές - οι οποίες ονομάζονται σύνδεσμοι (links). Κατά την επικοινωνία δύο αντικειμένων, ένα αντικείμενο-πελάτης (client) χρησιμοποιεί τις εξυπηρετήσεις ενός άλλου αντικειμένου-προμηθευτή (supplier). Τα αντικείμενα μπορούν να διαχωριστούν στις παρακάτω κατηγορίες, ανάλογα με τις σχέσεις εξυπηρέτησης στις οποίες συμμετέχουν:

- Actors ή Active objects χαρακτηρίζονται όσα αντικείμενα ενεργούν σε άλλα και ποτέ δεν παρέχουν κάποια εξυπηρέτηση.
- Server objects ονομάζονται όσα αντικείμενα παρέχουν μόνο εξυπηρετήσεις σε άλλα αντικείμενα του συστήματος.
- Agent objects ονομάζονται όσα αντικείμενα συμμετέχουν σε σχέσεις εξυπηρέτησης, τόσο ως clients, όσο και ως suppliers.

3.5.3 Σχέσεις μεταξύ κλάσεων

Οι κλάσεις ενός συστήματος μπορούν να συνδέονται με ορισμένες σχέσεις. Η UML (Unified Modeling Language) δίνει τη δυνατότητα περιγραφής των σχέσεων αυτών με τη χρήση των διαγραμμάτων κλάσεων (class diagrams). Κάποιες από τις πιο διαδεδομένες κατηγορίες συσχετίσεων κλάσεων είναι οι ακόλουθες [34]:

- Association: Η πιο γενικευμένη και λιγότερο σημασιολογικά ισχυρή συσχέτιση μεταξύ κλάσεων.
- Using: Η συσχέτιση αυτή υποδηλώνει μία αμφίδρομη σημασιολογική συσχέτιση πελάτη - προμηθευτή. Ουσιαστικά αντιπροσωπεύει τη σχέση link.
- Aggregation: Η συνάθροιση καταγράφει την ιεραρχική σχέση όλου-μέρους μεταξύ των στιγμιότυπων δύο κλάσεων. Η φυσική συμπερίληψη καλείται συμπερίληψη με τιμή (containment by value) όπου το μέρος δεν υπάρχει χωρίς το όλο, ενώ η ιδεατή συμπερίληψη ονομάζεται και συμπερίληψη με αναφορά (containment by reference).

- Instantiation: Η σχέση αυτή χρησιμοποιείται από ορισμένες γλώσσες ως μία μορφή γενίκευσης-εξειδίκευσης. Μία παραμετρική ή γενερική κλάση χρησιμοποιείται ως πρότυπο για την παραγωγή άλλων κλάσεων.
- Metaclass: Μετακλάση είναι μία κλάση της οποίας τα στιγμιότυπα είναι επίσης κλάσεις, και επιτρέπει στο σύστημα να χειρίζεται τις κλάσεις ως αντικείμενα.
- Inheritance: βλ. 3.3.4.

3.5.4 Κληρονομικότητα

Στο κεφάλαιο 8 του βιβλίου [34] ορίζεται η κληρονομικότητα, ως ο μηχανισμός που επιτρέπει την υλοποίηση σχέσεων τόσο γενίκευσης/εξειδίκευσης μεταξύ εννοιών, όσο και την ανάπτυξη νέων κλάσεων οι οποίες κληρονομούν χαρακτηριστικά από ήδη υπάρχουσες κλάσεις. Οι νέες αυτές κλάσεις (υποκλάσεις ή subclasses) υιοθετούν μεθόδους και ιδιότητες από μία οι περισσότερες κλάσεις που ονομάζονται υπερκλάσεις (superclasses). Το μεγάλο πλεονέκτημα της κληρονομικότητας είναι η αύξηση της επαναχρησιμοποίησης των ήδη υπαρχόντων κλάσεων του περιβάλλοντος υλοποίησης.

Υπερκάλυψη (overriding) καλείται ο εκ νέου ορισμός της συμπεριφοράς ή και της δομής της κλάσης προγόνου από κάποια υποκλάση της [34]. Πολλές φορές, μία κλάση χρειάζεται να κληρονομήσει χαρακτηριστικά περισσοτέρων από μίας προγονικών κλάσεων. Η λειτουργία αυτή ονομάζεται Πολλαπλή κληρονομικότητα (Multiple Inheritance). Πολυμορφισμός (polymorphism) καλείται η δυνατότητα ενός συστήματος OOP να εφαρμόσει την ίδια λειτουργία σε πολλαπλές κλάσεις, κάθε μία από τις οποίες μπορεί να την υλοποιήσει με διαφορετικό τρόπο.

3.6 Σχεδιασμός παιχνιδιών διαδραστικής φαντασίας

Η φιλοσοφία του σχεδιασμού των παιχνιδιών Interactive Fiction διαφέρει αρκετά από την περίπτωση των υπόλοιπων computer games, αλλά και την περίπτωση της διάρθρωσης ενός μη-διαδραστικού μυθιστορήματος. Το γεγονός ότι το σύστημα αλληλοεπιδρά άμεσα με τις αποφάσεις του παίκτη, και συνεπώς η δομή της ιστορίας δεν είναι γραμμική, δημιουργεί

προκλήσεις στους δημιουργούς του. Κάποιες από τις θεωρήσεις που πρέπει να ληφθούν υπόψιν από τον σχεδιαστή ενός παιχνιδιού Interactive Fiction αναλύονται παρακάτω.

3.6.1 Συνδυασμός προκατασκευασμένου και διαδικαστικά παραγόμενου περιεχομένου

Η δημιουργία ενός παιχνιδιού που προσαρμόζεται στις επιλογές του παίκτη απαιτεί έναν συνδυασμό προκατασκευασμένου και διαδικαστικώς παραγόμενου περιεχομένου, όπως αναλύεται στο κεφάλαιο 4 του [24]. Ο συνδυασμός των έτοιμων τμημάτων κειμένου με τα παραγόμενα τμήματα πρέπει να γίνεται με μέριμνα στα συμφραζόμενα του κειμένου, και με τρόπο που διασυνδέει διαφορετικά γεγονότα της πλοκής μεταξύ τους, ώστε να δημιουργείται μία πιο αληθοφανής αφηγηματική εμπειρία για τον παίκτη. Παλαιότερα παιχνίδια, όπως το *Murder on the Zinderneuf* έκαναν εκτεταμένη χρήση templates και τυχαίου συνδυασμού στοιχείων όπως χαρακτήρες, γεγονότα και αντικείμενα μεταξύ gameplays για να αυξήσουν την ποικιλία των παραγόμενων ιστοριών.

Κάποιες τεχνικές για τον συνδυασμό διαδικαστικού και στατικού περιεχομένου δίνονται στο κεφάλαιο 8 του [24], όπως η παρουσίαση στατικών κειμένων με τυχαία σειρά. Για να αποφευχθεί η πολυδιάσπαση της αφήγησης, θα πρέπει να χρησιμοποιηθούν κάποια σημεία-άγκυρες για την παρουσίαση σημαντικών πληροφοριών και τον σχηματισμό διασυνδέσεων. Ακόμα, η εμπειρία ενός παίκτη με το παραγόμενο περιεχόμενο μπορεί να βελτιωθεί, εάν ο ίδιος έχει τη δυνατότητα να επηρεάσει με κάποιον τρόπο την τυχαιότητα της διαδικαστικής παραγωγής.

Μία ιδιότητα του διαδικαστικού κειμένου η οποία μπορεί να φαίνεται αρχικά ως μειονέκτημα είναι η τάση να παράγονται κείμενα τα οποία δεν συμφωνούν απόλυτα με τα συμφραζόμενα, ή η σημασία τους δεν είναι ξεκάθαρη. Η συμπεριφορά αυτή όμως μπορεί να χρησιμοποιηθεί ως εργαλείο του σχεδιαστή παιχνιδιών IF. Στο κεφάλαιο 9 του [24] παρουσιάζονται κάποια παιχνίδια interactive fiction στα οποία έγινε χρήση uncanny («αλλόκοτου») κειμένου είτε σε λεπτομέρειες του κόσμου του παιχνιδιού (π.χ. δημιουργία τυχαίων τίτλων βιβλίων), για τη δημιουργία χιουμοριστικών κειμένων, ή για την πρόκληση κάποιου άλλου είδους συναισθηματικής φόρτισης στον παίκτη με την αλλαγή λέξεων-κλειδιών σε στατικό κείμενο.

3.6.2 Η πορεία της πλοκής

Όπως εξηγείται στο κεφάλαιο 8 του [24], υπάρχουν δύο κύριοι τρόποι για έναν σχεδιαστή παιχνιδιών IF να προσεγγίσει την πλοκή της ιστορίας. Ο πρώτος είναι με έναν σχεδιασμό "only forwards" στον οποίο η δομή της πλοκής κινείται προς μία μόνο κατεύθυνση, δημιουργώντας την ψευδαίσθηση μόνο της επιλογής στον παίκτη. Ο δεύτερος είναι το «άνοιγμα» της ροής της πλοκής, όπου επιτρέπεται στον παίκτη να επιλέξει ο ίδιος την κατεύθυνση της ιστορίας.

Παρότι η δεύτερη επιλογή επιτρέπει στον παίκτη πολύ μεγαλύτερη ευελιξία κινήσεων και βελτιώνει τον διαδραστικό χαρακτήρα του παιχνιδιού, η υλοποίηση μίας αφήγησης ανοιχτής ροής αποτελεί πρόκληση. Εάν ο κόσμος του παιχνιδιού παραμένει στατικός χωρίς την παρέμβαση του παίκτη, η εξέλιξη της πλοκής μπορεί να σταματήσει και ο παίκτης να χάσει το ενδιαφέρον του αρκετά εύκολα. Σημαντικό για την πλοκή είναι επίσης το να συμπεριληφθούν οι γνώσεις του Player Character για τον κόσμο στους μηχανισμούς του παιχνιδιού. Η δυναμική απόκτηση γνώσης μπορεί να αναπαρασταθεί στη μορφή ενός ακυκλικού κατευθυνόμενου γράφου (Directed Acyclic Graph). Κάθε κόμβος αναπαριστά ένα στάδιο γνώσης, από το πιο γενικό στο πιο ειδικό. Έτσι δημιουργούνται αλυσίδες γνώσης, οι κόμβοι των οποίων απαιτούν την εκπλήρωση κάποιων αρχικών συνθηκών ώστε να αποκαλυφθούν στον παίκτη και βοηθούν την πλοκή να προχωρήσει μπροστά, με την παροχή νέων πληροφοριών για τον κόσμο του παιχνιδιού.

Η πλοκή μπορεί να διαφοροποιηθεί από παιχνίδι σε παιχνίδι σε διάφορα επίπεδα:

- Στο επίπεδο του κόσμου του παιχνιδιού: Μπορεί να υπάρχει ποικιλία στα σημαντικότερα γεγονότα και στα σημεία ενδιαφέροντος της πλοκής.
- Στο επίπεδο των γεγονότων: Θα πρέπει να δημιουργηθεί μία διακριτή ακολουθία από γεγονότα τα οποία αναφέρονται το ένα στο άλλο, και έτσι δημιουργούν την αίσθηση της χρονικής συνέχειας στην ιστορία. Το σχήμα αίτιου/αποτελέσματος θα πρέπει να είναι ξεκάθαρο, ώστε να δίνει περισσότερη αληθοφάνεια στα γεγονότα του παιχνιδιού.
- Στο επίπεδο της πρότασης, είναι σημαντικό να υπάρχει δυνατότητα για ποικιλία και τυχαιότητα στην επιλογή φράσεων, αλλά με τρόπο που να προκαλεί ευδιάκριτες

αλλαγές στον τόνο και στη σημασία της πρότασης - η ποικιλία συνωνύμων δεν εμπλουτίζει τόσο το διαδικαστικά παραγόμενο κείμενο.

Οι κανόνες για την διαφοροποίηση της ιστορίας σε αυτά τα επίπεδα που αναφέρθηκαν δε χρειάζεται να μοντελοποιηθούν με περίπλοκο τρόπο για να φαίνονται περίτεχνοι στον παίκτη. Απλώς θα πρέπει να υπάρχει αρκετή νοηματική συνέχεια ώστε να μην είναι αρκετά εμφανής η διαδικαστική φύση του κειμένου. Με την διασύνδεση ανεξάρτητων γεγονότων με τη χρήση κοινών φράσεων και αναφορών στο κείμενο, η συνοχή στην πλοκή μπορεί να αναδυθεί μέσα από την τυχαιότητα (κεφάλαιο 11 του [24]).

3.6.3 Η εμπειρία του παίκτη

Η λειτουργία των διαδραστικών εφαρμογών προϋποθέτει την ενεργή συμμετοχή του παίκτη και στην προσήλωση του ενδιαφέροντός του. Σαν αποτέλεσμα, το πώς εξασφαλίζεται η προσοχή του παίκτη αποτελεί έναν σημαντικό παράγοντα κατά τον σχεδιασμό ενός συστήματος διαδραστικής φαντασίας. Η βιωματική εμπειρία του παίκτη κατά την αλληλοεπίδρασή του με το παιχνίδι είναι πολύ σημαντική στη διαδικασία απόδοσης νοήματος στην ιστορία.

Ένας παράγοντας ενδιαφέροντος του παίκτη είναι η δυνατότητα που του παρέχεται να δραματοποιήσει την ιστορία με την οποία αλληλοεπιδρά. Αυτή η ανάγκη για δραματοποίηση είναι παρόμοια με την αντίστοιχη συμπεριφορά που διαμορφώνουν οι παίκτες Role Playing Games (RPGs) με τα οποία η διαδραστική φαντασία μοιράζεται πολλά κοινά χαρακτηριστικά. Ως εκ τούτου, θα πρέπει να δίνεται η δυνατότητα στον παίκτη να κατανοήσει τον Player Character και τα κίνητρά του. Ακόμα το σύστημα του παιχνιδιού θα πρέπει να επιβραβεύσει τον παίκτη με κάποιον τρόπο όταν οι πράξεις του συμβαδίζουν με τον χαρακτήρα του στο παιχνίδι. Ο παίκτης δηλαδή θα πρέπει να έχει αρχικά μία μεγάλη ποικιλία επιλογών ώστε να μπορεί να παραμετροποιήσει την συμπεριφορά του σε πολύ μεγάλο βαθμό, και έπειτα να έχει το κίνητρο να μην παίξει απαραίτητα στρατηγικά, αλλά με γνώμονα τα χαρακτηριστικά του Player Character ([24], κεφάλαιο 10).

Τα συναισθήματα τα οποία προκαλεί η αφήγηση στον παίκτη είναι μία ακόμα σημαντική παράμετρος για τον καθορισμό της εμπειρίας του. Πηγές συναισθηματικής φόρτισης μπορεί να είναι η ταχύτητα και η προβλεψιμότητα με την οποία οι συνθήκες του παιχνιδιού αλλάζουν.

Όσο πιο συχνές είναι οι σημαντικές και συναισθηματικά ενδιαφέρουσες αποφάσεις τις οποίες καλείται να πάρει ο παίκτης, τόσο πιο έντονη είναι η συναισθηματική του εμπειρία. Οι αποφάσεις του παίκτη και οι μηχανισμοί του παιχνιδιού θα πρέπει να έχουν έμπνευση και παραλληλισμό με γεγονότα της πραγματικής ζωής. Όταν κάθε αλληλοεπίδραση έχει ξεκάθαρα κόστη και οφέλη, αυτό οδηγεί τον παίκτη να αναπτύξει στρατηγικές παιχνιδιού εμπνεόμενος από πραγματικές καταστάσεις όπως αναφέρεται στο [24], κεφάλαιο 12.

Τέλος, έμφαση θα πρέπει να δοθεί στις ιστορίες οι οποίες δημιουργούν οι ίδιοι οι παίκτες μέσα από την εμπειρία τους με το παιχνίδι. Οι εσωτερικοί μηχανισμοί και η μοντελοποίηση του κόσμου του παιχνιδιού δεν έχουν τόση αξία για τον παίκτη, όσο έχει η αφήγηση την οποία μπορεί να δημιουργήσει ο ίδιος. Όσον αφορά τους μηχανισμούς του παιχνιδιού, πρέπει να επικεντρώνονται στις διασυνδέσεις μεταξύ τμημάτων της ιστορίας, με τρόπους που να είναι εμφανείς και κατανοητοί από τους παίκτες. Ο σχεδιασμός ενός παιχνιδιού διαδικαστικής αφήγησης θα πρέπει να γίνεται πρωτίστως από την οπτική των παικτών, όσον αφορά την επιλογή των πληροφοριών που αυτοί θα λαμβάνουν, τη μορφή με την οποία τη λαμβάνουν, καθώς και την ελευθερία να διαλέξουν διαφορετικά μονοπάτια στην ιστορία.

4. ΣΧΕΔΙΑΣΗ ΚΑΙ ΥΛΟΠΟΙΗΣΗ ΤΟΥ ΠΑΙΧΝΙΔΙΟΥ

4.1 Εργαλεία υλοποίησης σε Python

Η Python είναι μία γλώσσα προγραμματισμού γενικού σκοπού, και είναι μία από τις πιο διαδεδομένες και πιο φιλικές προς αρχαρίους γλώσσες προγραμματισμού σήμερα. Η γλώσσα αποτελεί ελεύθερο λογισμικό και διατίθεται δωρεάν, ενώ είναι διαθέσιμη για όλα τα λειτουργικά συστήματα. Πέρα από τις βασικές λειτουργίες και συναρτήσεις της γλώσσας, επιπλέον συναρτήσεις ομαδοποιούνται σε βιβλιοθήκες (modules) οι οποίες χρησιμοποιούνται για εφαρμογές ειδικού σκοπού [4]. Για τις ανάγκες της παρούσας εργασίας χρησιμοποιήθηκαν διάφορες βιβλιοθήκες, είτε ενσωματωμένες στη γλώσσα (Built-in), είτε παρεχόμενες εκτός της standard library της Python.

Για την ανάπτυξη του παιχνιδιού ήταν απαραίτητες πολλές από τις λειτουργίες που προσφέρει η Python στην standard βιβλιοθήκη της, και συγκεκριμένα οι λειτουργίες αντικειμενοστραφούς προγραμματισμού, εντολές για τη διεπαφή της εφαρμογής με το λειτουργικό σύστημα, εντολές για τη διαχείριση αρχείων τύπου JSON, καθώς και συναρτήσεις για τη διαχείριση συμβολοσειρών, ψευδοτυχαίων αριθμών και καταγραφής χρόνου. Επιπλέον, πολλές άλλες λειτουργίες ειδικού σκοπού έπρεπε να χρησιμοποιηθούν, όπως κάποια βασικά εργαλεία επεξεργασίας φυσικής γλώσσας και εντολές για την δημιουργία ενός γραφικού περιβάλλοντος διεπαφής χρήστη.

Για την υλοποίηση του κώδικα της εργασίας χρησιμοποιήθηκε το περιβάλλον ανάπτυξης Pycharm (έκδοση 2019.3.4 Professional Edition), ενώ σαν μεταγλωττιστής (interpreter) χρησιμοποιήθηκε η Python 3.7 μέσω του Anaconda 4.10.1 [5][6][7].

4.1.1 Αντικειμενοστραφής προγραμματισμός σε Python

Η Python υποστηρίζει πολλά διαφορετικά μοντέλα προγραμματισμού, ένα από τα οποία είναι το αντικειμενοστραφές μοντέλο που αναλύθηκε στο Κεφάλαιο 3. Οι κλάσεις στην Python ορίζονται με την εντολή class, και τα αντικείμενα που προέρχονται από αυτές δημιουργούνται μέσω της συνάρτησης constructor __init__(), πρώτο όρισμα της οποίας είναι πάντοτε το όνομα με το οποίο το αντικείμενο αναφέρεται μέσα στον ορισμό της κλάσης (συνήθως self) [4]. Για την αναφορά σε ιδιότητες (attributes) και μεθόδους (methods) των αντικειμένων στην Python χρησιμοποιείται το λεγόμενο dot notation, στο οποίο το όνομα του αντικειμένου χωρίζεται από την ιδιότητα με μία τελεία '.'.

Εκτός από την __init__, η Python προσφέρει πολλές άλλες built-in μεθόδους για τις κλάσεις της. Οι μέθοδοι αυτές ονομάζονται magic methods, και χρησιμοποιούνται για να υλοποιήσουν ειδικές λειτουργίες στα αντικείμενα των κλάσεων, όπως το operator overloading (η χρήση των συνηθισμένων τελεστών της γλώσσας πάνω σε ειδικά αντικείμενα), απεικονίσεις των αντικειμένων, η πρόσβαση σε ιδιότητες των αντικειμένων και άλλα [8]. Για τις ανάγκες της παρούσας εργασίας οι «μαγικές» μέθοδοι χρησιμοποιήθηκαν ευρέως.

Η κληρονομικότητα είναι ένα ακόμα στοιχείο των κλάσεων στην Python το οποίο χρησιμοποιήθηκε εκτεταμένα στην ανάπτυξη του παιχνιδιού, τόσο στο κομμάτι της υλοποίησης των Οντοτήτων του παιχνιδιού, όσο και στη διεπαφή με τον παίκτη. Στην Python μία κλάση μπορεί να κληρονομήσει από μία ή περισσότερες κλάσεις, μέσω της συνάρτησης super(). Επιπλέον, οι μέθοδοι της Κλάσης-Γονέα (Parent Class) μπορούν να υπερκαλυφθούν (method overriding), δηλαδή να προσαρμοστεί η λειτουργία τους όταν ορίζονται στην Κλάση-Παιδί (Child Class) [9].

4.1.2 Νήματα στην Python - Threading Built-In Module

Για την υλοποίηση πολλών λειτουργιών που σχετίζονται με το παιχνίδι που αναπτύχθηκε στα πλαίσια αυτής της εργασίας ήταν απαραίτητη η χρήση διεργασιών οι οποίες εκτελούνται

παράλληλα. Για τον σκοπό αυτό, χρησιμοποιήθηκε η ενσωματωμένη βιβλιοθήκη Threading της Python [10], η οποία μπορεί να χρησιμοποιηθεί για την εκτέλεση πολλών I/O-bound εργασιών ταυτοχρόνως.

Ειδικότερα, χρησιμοποιήθηκε η κλάση Thread για την υλοποίηση τόσο της κλάσης του χρονομετρητή (timer.CustomTimer) στο παιχνίδι, όσο και για την εκτέλεση του βρόχου επανάληψης και των λοιπών διεργασιών της κλάσης main.GameThread (βλ. Ενότητα 4.2) παράλληλα με την υλοποίηση της γραφικής διεπαφής. Η Thread κληρονομήθηκε στις αντίστοιχες κλάσεις ως η Parent Class, και η μέθοδος run() υπερκαλύφθηκε για την υλοποίηση των εκάστοτε λειτουργιών τους. Σύμφωνα και με το επίσημο documentation της βιβλιοθήκης [10], μετά από τη δημιουργία ενός thread object, η εκκίνησή του γίνεται μέσω της μεθόδου start, η οποία εκτελεί την μέθοδο run σε ένα ξεχωριστό νήμα ελέγχου.

4.1.3 Διαχείριση αρχείων JSON σε Python - json Built-In Module

Η βιβλιοθήκη json [11] της Python χρησιμοποιείται για τη διαχείριση αρχείων JSON (JavaScript Object Notation). Το JSON είναι ένα format ανταλλαγής δεδομένων που εμπνέεται από το συντακτικό της Javascript. Τα αρχεία αυτά χρησιμοποιήθηκαν κατά την ανάπτυξη του παιχνιδιού για την αποθήκευση πολλών από των αντικειμένων του. Βασικές λειτουργίες που σχετίζονται με το πρότυπο αυτό είναι η σειριοποίηση (serialization) και αποσειριοποίηση (deserialization) των αντικειμένων, δηλαδή η κωδικοποίηση και αποκωδικοποίησή τους σε συμβολοσειρές αντίστοιχα, σύμφωνα με τους κανόνες που διέπουν το πρότυπο.

Το πρότυπο χαρακτηρίζεται από την κωδικοποίηση των αντικειμένων σε συμβολοσειρές, με τη μορφή ζευγών κλειδιού/τιμής (παρόμοια με τον τύπο dictionary της Python). Μία βασική διαφορά του προτύπου από τα λεξικά της Python είναι ότι όλα τα κλειδιά πρέπει να είναι συμβολοσειρές. Η σειριοποίηση των αντικειμένων γίνεται με τη χρήση της κλάσης JSONEncoder, η οποία μπορεί να διαχειριστεί αντικείμενα που ανήκουν στους built-in τύπους της Python dict, list, tuple, str, int, float, bool και None. Για την σειριοποίηση περισσοτέρων ειδών αντικειμένων, η κλάση JSONEncoder πρέπει να κληρονομηθεί από μία custom κλάση στην οποία θα ορίζονται ειδικές αντιστοιχήσεις. Αντίστοιχα, η κλάση JSONDecoder της βιβλιοθήκης json διαχειρίζεται την αποσειριοποίηση συμβολοσειρών

τύπου json σε αντικείμενα Python. Όπως και στην περίπτωση του κωδικοποιητή, ειδική μέριμνα πρέπει να δοθεί στην αποκωδικοποίηση εξειδικευμένων αντικειμένων [11].

4.1.4 Ψευδοτυχαίοι αριθμοί στην Python - Numpy Random Module

Η Numpy αποτελεί το βασικό πακέτο βιβλιοθηκών της Python για επιστημονικούς υπολογισμούς, ενώ διαθέτει μία ποικιλία από μαθηματικές λειτουργίες που πραγματοποιούνται πάνω σε πίνακες (numpy arrays). Μία από τις βιβλιοθήκες που παρέχεται μαζί με τη Numpy είναι και η numpy.random [12]. Η βιβλιοθήκη αυτή περιέχει συναρτήσεις για την παραγωγή ψευδοτυχαίων αριθμών με τον συνδυασμό ενός BitGenerator για την παραγωγή ακολουθιών και ενός Generator για την χρήση αυτών των ακολουθιών για την δειγματοληψία από διάφορες στατιστικές κατανομές.

Καθώς το παιχνίδι που αναπτύχθηκε στα πλαίσια της εργασίας χρησιμοποιεί τεχνικές διαδικαστικής παραγωγής, είναι απαραίτητο να χρησιμοποιηθεί μία γεννήτρια τυχαίων αριθμών η οποία είναι αρκετά αξιόπιστη όσον αφορά την επαναληψιμότητά της και την μοναδικότητα των αριθμών τους οποίους παράγει. Η επαναληψιμότητα των αντικειμένων του παιχνιδιού εξασφαλίζεται με την επιλογή του κατάλληλου σπόρου (seed) σαν είσοδο στη γεννήτρια τυχαίων αριθμών. Η numpy.random υποστηρίζει το seeding του BitGenerator για όλες τις μεθόδους της [12].

4.1.5 Επεξεργασία Φυσικής Γλώσσας σε Python - NLTK Module

Η βιβλιοθήκη NLTK (Natural Language Toolkit) [13] είναι μία από τις πιο δημοφιλείς πλατφόρμες για την υλοποίηση προγραμμάτων Python τα οποία διαχειρίζονται δεδομένα φυσικής γλώσσας. Μπορεί να χρησιμοποιηθεί για την εύκολη διεπαφή ενός προγράμματος Python με σώματα κειμένων, ενώ παρέχει μία μεγάλη λίστα από βιβλιοθήκες για κατηγοριοποίηση, λημματοποίηση, συντακτική και σημασιολογική ανάλυση καθώς και άλλες λειτουργίες απαραίτητες για την επεξεργασία κειμένων φυσικής γλώσσας.

Στο παιχνίδι που υλοποιήθηκε χρησιμοποιήθηκαν λειτουργίες της βιβλιοθήκης που σχετίζονται με τον κατακερματισμό μίας πρότασης σε λέξεις (tokenization), με την ταυτοποίηση της συντακτικής και γραμματικής κατηγορίας μίας λέξης (Part-Of-Speech

Tagging - POS Tagging), με την συντακτική ανάλυση προτάσεων (parsing), όπως και με την παραγωγή λέξεων μέσω μίας τυπικής γραμματικής.

4.1.6 Γραφική Διεπαφή Χρήστη (GUI) σε Python - Kivy Module

Η kivy [14] αποτελεί μία βιβλιοθήκη ανοιχτού λογισμικού της Python για την γρήγορη παραγωγή εφαρμογών που χρησιμοποιούν σύγχρονες μορφές διεπαφής με χρήστες (User Interfaces). Η βιβλιοθήκη είναι λειτουργική σε πολλές πλατφόρμες, τόσο προσωπικών υπολογιστών όσο και κινητών τηλεφώνων, ενώ διατίθεται δωρεάν. Η kivy υποστηρίζει πάνω από 20 τύπους widgets, οι οποίοι είναι εύκολα επεκτάσιμοι.

Στην παρούσα εφαρμογή, η βιβλιοθήκη χρησιμοποιήθηκε για τη δημιουργία μίας απλής γραφικής διεπαφής, η οποία επιτρέπει στον παίκτη να εισάγει τις εντολές του σε ένα παράθυρο, σε αντίθεση με το Command Prompt, καθώς και να διαβάσει τα αποτελέσματα των εντολών του στο ίδιο παράθυρο. Μία ακόμα χρήση της kivy στο παιχνίδι αφορούσε την αναπαραγωγή ήχου, για τη μουσική επένδυση της εφαρμογής.

4.1.7 Άλλες βιβλιοθήκες

Κάποιες ακόμα βιβλιοθήκες της Python που χρησιμοποιήθηκαν κατά την ανάπτυξη του παιχνιδιού είναι οι ακόλουθες:

- string: Η built-in βιβλιοθήκη string της Python χρησιμοποιείται για συνήθεις πράξεις σε συμβολοσειρές, ενώ περιλαμβάνει και αρκετές σταθερές οι οποίες περιέχουν χρήσιμες κατηγορίες χαρακτήρων (όπως για παράδειγμα όλα τα σημεία στίξης) [15].
- random: Η random είναι η ενσωματωμένη βιβλιοθήκη της Python για την παραγωγή ψευδοτυχαίων αριθμών. Στη βιβλιοθήκη περιλαμβάνονται συναρτήσεις για την παραγωγή ακεραίων, την επιλογή τυχαίων στοιχείων μίας ακολουθίας (λίστας, πλειάδας κλπ.) ή για την δειγματοληψία διάφορων τυχαίων κατανομών [16].
- sys: Η ενσωματωμένη αυτή βιβλιοθήκη παρέχει πρόσβαση σε μεταβλητές που χρησιμοποιούνται από τον μεταγλωττιστή της Python, ή σε συναρτήσεις οι οποίες αλληλοεπιδρούν συχνά με τον μεταγλωττιστή [17].

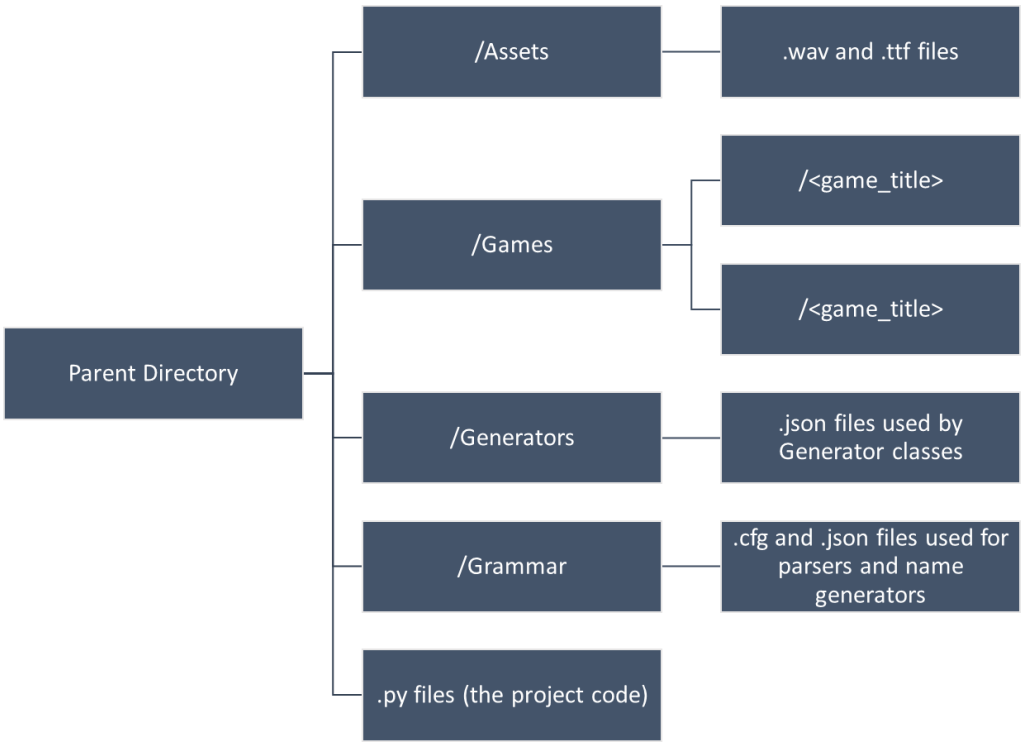
- os: Η βιβλιοθήκη os [18] προσφέρει πρόσβαση σε λειτουργικότητες που σχετίζονται με το λειτουργικό σύστημα, όπως για παράδειγμα τη διαχείριση μονοπατιών (paths) για αρχεία ή φακέλους.
- time: Η built-in βιβλιοθήκη time περιλαμβάνει συναρτήσεις σχετικές με τη μέτρηση του χρόνου [19].
- shutil: Η shutil [20] είναι μία βιβλιοθήκη η οποία περιλαμβάνει συναρτήσεις για πράξεις υψηλού επιπέδου σε αρχεία και συλλογές αρχείων, όπως είναι η αντιγραφή και η διαγραφή αρχείων ή φακέλων.
- pandas: Η βιβλιοθήκη pandas [21] δεν περιλαμβάνεται στην standard library της Python, αλλά είναι μία συχνά χρησιμοποιούμενη βιβλιοθήκη. Η pandas παρέχει γρήγορες και ευέλικτες δομές δεδομένων για την υλοποίηση εργασιών με σχεσιακά ή labeled δεδομένα.

4.2 Η βασική αρχιτεκτονική του παιχνιδιού

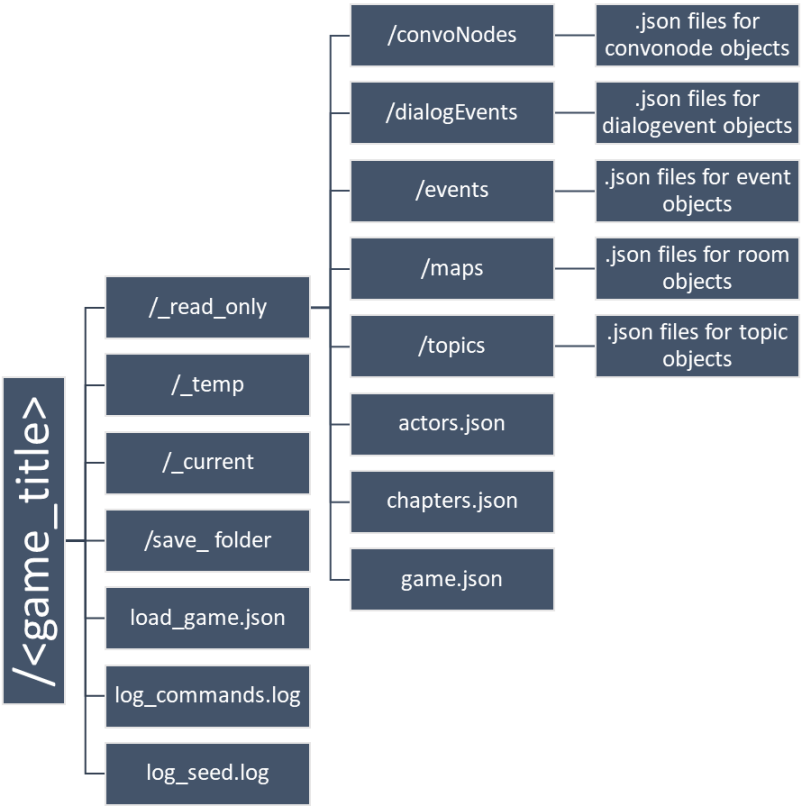
4.2.1 Δομή του καταλόγου (directory) της εφαρμογής

Ο κώδικας του παιχνιδιού που αναπτύχθηκε στα πλαίσια της παρούσας διπλωματικής εργασίας διαχωρίζεται σε πολλά διαφορετικά Python modules, καθένα από τα οποία περιέχει κάποιες από τις κλάσεις και συναρτήσεις που απαρτίζουν το πρόγραμμα. Το module από το οποίο εκκινείται η εφαρμογή βρίσκεται στο αρχείο main.py. Η δομή των φακέλων στους οποίους οργανώνονται τα αρχεία της εφαρμογής φαίνονται στις Εικ. 1 και 2. Τα αρχεία κώδικα της εφαρμογής βρίσκονται μέσα στο parent directory, μαζί με τον φάκελο /Assets (στον οποίο είναι αποθηκευμένα τα wave και ttf αρχεία, για τη μουσική και τις γραμματοσειρές αντίστοιχα), τον φάκελο /Games (που περιέχει τους φακέλους των διάφορων παιχνιδιών που διαχειρίζεται η εφαρμογή), τον φάκελο /Generators (με τα αρχεία JSON που χρησιμοποιούνται για τη διαδικαστική παραγωγή αντικειμένων στο παιχνίδι), και τον φάκελο /Grammar (στον οποίο περιέχονται τα αρχεία απαραίτητα για την συντακτική ανάλυση εντολών του παίκτη και για την παραγωγή ονομάτων αντικειμένων του παιχνιδιού).

Κάθε φάκελος που βρίσκεται στον κατάλογο /Games έχει τη δομή που φαίνεται στην Εικ. 2 και φέρει ως όνομα τον τίτλο του παιχνιδιού του οποίου τα αρχεία περιλαμβάνει. Συγκεκριμένα, απαρτίζεται από 4 -το πολύ- φακέλους της ίδιας μορφής. Ο κατάλογος /_read_only είναι αυτός στον οποίο αποθηκεύονται τα αρχεία από τα οποία φορτώνεται αρχικά το παιχνίδι. Το πρόγραμμα δεν γράφει ποτέ σε αυτόν τον κατάλογο. Αντίθετα, ο φάκελος /_current είναι αυτός στον οποίο αποθηκεύονται τα αρχεία του παιχνιδιού κατά την εκτέλεσή του, και αποτελεί φάκελο και αποθήκευσης και ανάγνωσης. Ο φάκελος /_temp χρησιμοποιείται για την προσωρινή αποθήκευση αρχείων του παιχνιδιού μεταξύ δύο εντολών του παίκτη, για την περίπτωση στην οποία δοθεί η εντολή undo. Τέλος, σε περίπτωση που ο παίκτης έχει επιλέξει να αποθηκεύσει μόνιμα ένα στιγμιότυπο ενός παιχνιδιού, τα αρχεία αυτά θα αποθηκευτούν σε έναν φάκελο /save_*. Καθένας από τους καταλόγους που αναγράφονται παραπάνω περιέχει όλα τα απαραίτητα αρχεία για την λειτουργία του παιχνιδιού, και τα οποία αναλύονται στη συνέχεια.



Εικ. 1 Η δομή φακέλων του parent directory της εφαρμογής.



Εικ. 2 Η δομή των φακέλων για τα αρχεία ενός παιχνιδιού.

Μία γενική εποπτεία των κλάσεων που απαρτίζουν το παιχνίδι υπάρχει στο Παράρτημα Α, όπου μπορούν να βρεθούν τα Class Diagrams της εφαρμογής.

4.2.2 Εκκίνηση του παιχνιδιού - Οι κλάσεις GameThread, MyApp, Initializer

Υπεύθυνες για την εκκίνηση της εφαρμογής είναι οι 3 κλάσεις GameThread, MyApp και Initializer. Η κλάση main.GameThread είναι υπεύθυνη για την εκκίνηση του νήματος στο οποίο 'τρέχει' το παιχνίδι. Σε αυτή, δημιουργείται ένα instance της κλάσης Initializer, ενώ στη μέθοδο run της κλάσης καλείται η μέθοδος Initializer.load_game() για να φορτωθούν όλα τα απαραίτητα αρχεία για τη λειτουργία του παιχνιδιού. Η κλάση app.MyApp είναι υπεύθυνη για τη γραφική διεπαφή του παιχνιδιού με τον παίκτη, και η λειτουργία της αναλύεται στο 4.12. Στην κλάση αυτή εκκινείται επίσης και το νήμα του GameThread. Η κλάση Initializer επιφορτίζεται με την φόρτωση δεδομένων και με τη δημιουργία instances των κλάσεων display.Display, game.Game και load.Loader. Η μέθοδος load_game της Loader διαβάζει από τα κατάλληλα αρχεία και επιστρέφει τα ορίσματα για τον constructor της Game και για τη μέθοδο Game.boot_game, η οποία, τελικά ξεκινάει τη λειτουργία του παιχνιδιού.

4.2.3 Αποθήκευση και ανάγνωση δεδομένων - Οι κλάσεις Loader και Saver

Οι κλάσεις που είναι υπεύθυνες για την αποθήκευση και την ανάγνωση αρχείων κατά τη λειτουργία του παιχνιδιού είναι οι κλάσεις save.Saver και load.Loader αντίστοιχα. Η Loader έχει μεθόδους για την αρχική φόρτωση του παιχνιδιού από τον Initializer, κατά την οποία όλα τα αρχεία μεταφέρονται από τον κατάλογο _read_only στον κατάλογο _current. Επίσης, στην περίπτωση που ο χρήστης θέλει να αναπαράγει τις εντολές που έδωσε σε προηγούμενο παιχνίδι (βλ. 4.11.3) διαβάζει τις εντολές που είναι αποθηκευμένες στο αρχείο log_command.log. Τέλος, η Loader είναι υπεύθυνη και για την ανάγνωση των αρχείων επιμέρους κεφαλαίων (Chapters), και την ανάσυρση της προηγούμενης κατάστασης του παιχνιδιού σε περίπτωση εντολής αναίρεσης.

Αντίστοιχα, η Saver έχει μεθόδους για την μόνιμη αποθήκευση ενός παιχνιδιού σε έναν φάκελο save, σε περίπτωση που ζητηθεί από τον παίκτη, αλλά και για την προσωρινή αποθήκευση του παιχνιδιού στους φακέλους _current και _temp, για την ομαλή λειτουργία της εφαρμογής.

4.2.4 Κωδικοποίηση και αποκωδικοποίηση αντικειμένων από αρχεία JSON - Οι κλάσεις EncodeCustom, DecodeCustom

Στο αρχείο custom_json.py υπάρχουν οι απαραίτητες συναρτήσεις και κλάσεις για την κωδικοποίηση και αποκωδικοποίηση των αντικειμένων του παιχνιδιού σε και από αρχεία JSON. Αρχικά, η κλάση EncodeCustom, η οποία κληρονομεί από την json.JSONEncoder, χρησιμοποιείται για την κωδικοποίηση αντικειμένων σε μορφή σειριοποιήσιμη (serializable) σε αρχεία JSON. Για τον σκοπό αυτό, η μέθοδός της default εξετάζει κατά πόσο ένα αντικείμενο που της εισάγεται για κωδικοποίηση ανήκει σε μία από τις κλάσεις του παιχνιδιού. Αν όχι, καλεί την αντίστοιχη μέθοδο της υπερκλάσης της. Αλλιώς, κωδικοποιεί το ειδικό αντικείμενο με τη χρήση της μεθόδου που διαθέτουν όλα τα σειριοποιήσιμα αντικείμενα του παιχνιδιού, την to_json. Αντίστοιχα, η κλάση DecodeCustom χρησιμοποιείται για την ανάγνωση και αποκωδικοποίηση αντικειμένων της εφαρμογής από αρχεία JSON. Κληρονομεί από την αντίστοιχη κλάση της module json.

Όλα τα αντικείμενα του παιχνιδιού, όταν αποθηκεύονται σε αρχεία JSON, κωδικοποιούνται σε λεξικά που περιέχουν όλα τα απαραίτητα ορίσματα για τον constructor

τους, καθώς και ένα κλειδί "_class_", η τιμή του οποίου είναι το όνομα της κλάσης του αντικειμένου. Η DecodeCustom, λοιπόν, εξετάζει κατά πόσο το αντικείμενο που της δόθηκε έχει την κατάλληλη αυτή δομή, και ανασύρει την κλάση του αντικειμένου μέσω του ονόματός της. Στη συνέχεια επιστρέφει ένα αντικείμενο της κλάσης αυτής, με ορίσματα των οποίων οι τιμές βρίσκονται στο λεξικό που διαβάστηκε από το αρχείο JSON.

4.2.5 Συσχέτιση και ταυτοποίηση αντικειμένων

Όλα τα αντικείμενα στο παιχνίδι τα οποία συσχετίζονται με κάποιο άλλο αντικείμενο ή χρειάζεται να ταυτοποιηθούν για οποιονδήποτε λόγο έχουν ως ιδιότητα την key. Η key έχει ως τιμή μία συμβολοσειρά που χαρακτηρίζει μοναδικά το αντικείμενο (είναι δηλαδή για αυτό ό,τι είναι ένα κλειδί μίας πλειάδας σε μία σχεσιακή βάση δεδομένων).

4.3 Η κλάση Game

Η κλάση Game αποτελεί τον σκελετό του παιχνιδιού. Είναι υπεύθυνη για την εκκίνηση του παιχνιδιού, τη διατήρηση του βρόχου επανάληψης στον οποίο δίνονται οι εντολές από τον παίκτη και συμβαίνουν τα γεγονότα ενός κεφαλαίου, και για την καταγραφή και συλλογή των αντικειμένων του παιχνιδιού σε λεξικά.

Αναλυτικότερα, η κλάση έχει ως ιδιότητες στιγμιότυπα άλλων σημαντικών κλάσεων της εφαρμογής, όπως είναι ο Loader και ο Saver που αναλύθηκαν παραπάνω, και ο Preparser, Parser, NounPhraseParser, CommandHandler, CustomTimer που θα αναλυθούν στη συνέχεια. Ακόμα, attributes της Game αποτελούν διάφορα λεξικά στα οποία συγκεντρώνονται αντικείμενα, δωμάτια, ρήματα, κεφάλαια, θέματα διαλόγου καθώς και άλλες οντότητες του κόσμου του παιχνιδιού. Αυτά τα λεξικά επιτρέπουν στην άμεση πρόσβαση στα αντικείμενα του παιχνιδιού, με τη χρήση του κατάλληλου κλειδιού.

Όσον αφορά τις μεθόδους του παιχνιδιού, οι σημαντικότερες είναι οι εξής:

- boot_game: Φορτώνει τα απαραίτητα δεδομένα για το παιχνίδι (actors, verbs, chapters, τα δεδομένα του τρέχοντος κεφαλαίου κλπ.). Έπειτα, αναμένει μέχρις ότου το νήμα που διαχειρίζεται τη γραφική διεπαφή να δώσει το σήμα να ξεκινήσει

το παιχνίδι. Εάν πρόκειται για replay καλείται η μέθοδος replay_game, αλλιώς καλείται η start_game.

- start_game: Η start_game 'τρέχει' το κεφάλαιο της εισαγωγής ώστε ο χρήστης να δώσει τα απαραίτητα στοιχεία για το seeding του κόσμου του παιχνιδιού. Έπειτα η εφαρμογή εισέρχεται στο βρόχο επανάληψης στον οποίο εναλλάσσονται η σειρά του παίκτη και η εξέλιξη του τρέχοντος κεφαλαίου.
- replay_game: Η replay_game εκτελεί την ίδια λειτουργία με την start_game, με τη μόνη διαφορά ότι φορτώνει τις εντολές από το αμέσως προηγούμενο run του παιχνιδιού και τυπώνει τα αποτελέσματά τους. Όταν οι εντολές εξαντληθούν, τότε η μέθοδος εισέρχεται στο ίδιο είδος βρόχου με την start_game.
- run_pc_turn: Η μέθοδος που διαχειρίζεται τη σειρά του χρήστη και τις εντολές που δίνει. Αρχικά καλεί μία άλλη μέθοδο, την get_command, για την λήψη μίας νέας εντολής από τον παίκτη. Μόλις δοθεί η εντολή, δίνεται ως είσοδος στον προ-συντακτικό αναλυτή, όπου γίνεται μία πρώτη επεξεργασία και διακρίνεται ο τύπος της. Αν πρόκειται για κάποιο είδος ειδικής εντολής (help, quit, undo, save), η διαχείρισή της εντολής 'περνάει' στις αντίστοιχες μεθόδους (help, quit_game, undo_move, Saver.save_game). Αν όμως πρόκειται για εντολή που χρειάζεται συντακτική ανάλυση, δίνεται ως είσοδος στη μέθοδο parsable_command.
- parsable_command: Η μέθοδος αυτή διαχειρίζεται μία εντολή, δίνοντάς την ως είσοδο πρώτα στον συντακτικό αναλυτική, έπειτα στον συντακτικό αναλυτή ονοματικών φράσεων, και μετά στον διαχειριστή εντολών (command handler), για την ανάλυση και την εκτέλεσή της. Σε κάθε στάδιο η συνάρτηση ελέγχει για την περίπτωση κάποιου είδους σφάλματος, το οποίο και πυροδοτεί το αντίστοιχο μήνυμα λάθους.
- run_chapter: Η run_chapter διαχειρίζεται τα γεγονότα και τις εναλλαγές κεφαλαίων (τα οποία παρουσιάζονται πιο αναλυτικά παρακάτω). Σε περίπτωση μηνύματος τέλους τελειώνει το παιχνίδι, ενώ σε περίπτωση εναλλαγής κεφαλαίου καλεί τις κατάλληλες μεθόδους για τη φόρτωση και την εκκίνησή του.
- undo_move: Αναιρεί την προηγούμενη κίνηση του παίκτη.

- help: Τυπώνει ένα μήνυμα βοήθειας για τον παίκτη, που περιέχει τις περιγραφές των ρημάτων του παιχνιδιού, τα αντικείμενα που διαθέτει ο Player Character στην κατοχή του, καθώς και τα θέματα προς συζήτηση.
- end_game: Διαχειρίζεται το τέλος του παιχνιδιού, καθώς και την αποθήκευσή του, σε περίπτωση που ο παίκτης το ζητήσει.
- refresh_things: Η μέθοδος αυτή έχει κεντρικό ρόλο στη διαχείριση των αντικειμένων του παιχνιδιού. Ανανεώνει τα περιεχόμενα των λεξικών currents και things. Το τελευταίο περιέχει όλα τα αντικείμενα του παιχνιδιού με τα οποία ο παίκτης μπορεί να αλληλοεπιδράσει. Σε αυτά συμπεριλαμβάνονται όλα τα αντικείμενα που έχει ο παίκτης στην κατοχή του, όλα τα αντικείμενα του δωματίου στο οποίο βρίσκεται, καθώς και άλλα αντικείμενα (όπως το τωρινό Ηλιακό Σύστημα με τους αντίστοιχους Πλανήτες του, βλ. 5.1). Στο λεξικό currents περιέχονται αντικείμενα όπως το τωρινό Δωμάτιο του παίκτη, το τωρινό Ηλιακό Σύστημα ή και ο τωρινός Πλανήτης, και χρησιμεύει στην εύκολη διαχείριση των αντικειμένων αυτών από το παιχνίδι εσωτερικά.

4.4 Οντότητες

4.4.1 Οντότητες - Η κλάση Entity

Οντότητες στο εξής αναφέρονται όλα τα «απτά» για τον χαρακτήρα του παίκτη αντικείμενα στο παιχνίδι - δηλαδή τα αντικείμενα στα οποία ο παίκτης μπορεί να εφαρμόσει κάποια πράξη. Όλες οι οντότητες του παιχνιδιού προέρχονται από την κλάση entities.Entity. Κάθε αντικείμενο της κλάσης διαθέτει ένα μοναδικό κλειδί (key) το οποίο το χαρακτηρίζει, ένα display_name το οποίο είναι ο τρόπος με τον οποίο εμφανίζεται σε περιγραφές του παιχνιδιού στον παίκτη, καθώς και την ιδιότητα reference_noun που είναι το ουσιαστικό/λέξη με το οποίο ο παίκτης μπορεί να αναφερθεί σε εντολές του στο αντικείμενο. Επίσης, κάθε στιγμιότυπο της κλάσης μπορεί να διαθέτει 3 περιγραφές (description - σύντομη περιγραφή του αντικειμένου, examine_description - αναλυτική περιγραφή του αντικειμένου, audible_description - ηχητική περιγραφή του αντικειμένου).

Άλλες σημαντικές ιδιότητες ενός αντικειμένου της κλάσης είναι η container (το κλειδί μίας άλλης οντότητας στην οποία περιέχεται το αντικείμενο), και το λεξικό contents, στο οποίο βρίσκονται οι οντότητες τις οποίες περιέχει το αντικείμενο. Επιπλέον, η κλάση διαθέτει το attribute entity_state, ένα λεξικό το οποίο χρησιμεύει σαν associative array, παρέχοντας ένα είδος «μνήμης» συσχέτισης κλειδιών-τιμής για το αντικείμενο. Ακόμα, κάθε οντότητα περιέχει 3 λεξικά που χαρακτηρίζουν τις πράξεις οι οποίες μπορούν να γίνουν σε αυτή:

- as_dirobj: Το λεξικό αυτό έχει ως κλειδιά τα ονόματα των ρημάτων του παιχνιδιού και ως τιμές True ή False, ανάλογα με το κατά πόσον η οντότητα μπορεί να χρησιμοποιηθεί ως άμεσο αντικείμενο των ρημάτων αυτών.
- as_indobj: Λεξικό με κλειδιά τους "qualifiers" του παιχνιδιού (κυρίως προθέσεις) και τιμές True ή False, ανάλογα με το εάν η οντότητα μπορεί να συνδυαστεί με αυτούς ως έμμεσο αντικείμενο σε μία πρόταση.
- action_description: Λεξικό με τα ρήματα του παιχνιδιού ως κλειδιά και τιμή False εάν δεν υπάρχει η περιγραφή του αντίστοιχου action για το αντικείμενο, ή τιμή τη συμβολοσειρά που περιγράφει στον παίκτη το αποτέλεσμα της πράξης πάνω στην οντότητα.

Επίσης, η κλάση διαθέτει ορισμένες μεθόδους για την υλοποίηση διαφόρων δράσεων και πράξεων στην οντότητα όπως:

- __str__: Η αναπαράσταση του αντικειμένου σε συμβολοσειρά.
- __iadd__: Overloading του τελεστή '+=' για την προσθήκη ενός αντικειμένου στο λεξικό contents της οντότητας.
- __isub__: Αντίστοιχο με την προηγούμενη μέθοδο, αλλά για αφαίρεση αντικειμένου.
- to_json: Επιστρέφει το private attribute __dict__ της κλάσης, μαζί με το όνομα της κλάσης (για την σειριοποίηση του αντικειμένου).
- get_contents: Επιστρέφει τα αντικείμενα τα οποία βρίσκονται στο λεξικό contents της οντότητας.
- on_dir_object: Διαχειρίζεται μία εισερχόμενη εντολή με το να καλεί την αντίστοιχη _on_<VERB> μέθοδο.

- _on_<VERB> methods: Υπάρχει μία μέθοδος για κάθε ρήμα του οποίου η οντότητα μπορεί να αποτελεί άμεσο αντικείμενο. Βλ. 4.6.

4.4.2 Αντικείμενα - Η κλάση Thing και οι υποκλάσεις της

Η κλάση Thing και οι υποκλάσεις της περιλαμβάνουν οντότητες που αντιπροσωπεύουν άψυχα αντικείμενα που βρίσκονται μέσα στον χώρο που κινείται ο παίκτης. Η κλάση things.Thing, που κληρονομεί από την Entity, έχει επιπλέον τις ιδιότητες reference_adjectives (μία λίστα με τα επίθετα τα οποία μπορεί να χρησιμοποιήσει ο παίκτης για να αναφερθεί σε αυτή), καθώς και την is_known (Boolean, True εάν ο παίκτης μπορεί να αντιληφθεί το αντικείμενο). Μία σημαντική μέθοδος της κλάσης είναι η string, η οποία επιστρέφει μία αναλυτική περιγραφή του αντικειμένου και των περιεχομένων του.

Άλλες κλάσεις που ορίζονται στο αρχείο things.py είναι:

- Fixture: Χρησιμοποιείται για αντικείμενα που δεν μετακινούνται μέσα στο παιχνίδι.
- OpenableContainer: «Δοχεία», αντικείμενα τα οποία ο παίκτης μπορεί να ανοίξει και να τοποθετήσει άλλα αντικείμενα στο εσωτερικό τους.
- Surface: Επιφάνειες πάνω στις οποίες μπορούν να τοποθετηθούν άλλα αντικείμενα.
- Movable: Αντικείμενα τα οποία μπορούν να μετακινηθούν από τον παίκτη.

4.4.3 Χαρακτήρες - Η κλάση Actor

Στο αρχείο actors.py ορίζεται η κλάση των χαρακτήρων του παιχνιδιού, Actor, η οποία κληρονομεί από την Entity, με επιπλέον ιδιότητες τον ενεργό κόμβο συζήτησης του χαρακτήρα (αναλυτικότερα στο 4.8.3), την παράμετρο is_known, καθώς και τα λεξικά abilities και on_command. Το abilities έχει ως κλειδιά τα ονόματα των ρημάτων του παιχνιδιού και ως τιμές True ή False, αναλόγως με το αν ο χαρακτήρας μπορεί να εκτελέσει την αντίστοιχη πράξη. Το on_command έχει επίσης ζεύγη ονόματα ρημάτων/Boolean τιμές, αλλά οι τιμές αυτές αντιστοιχούν στο κατά πόσο ο χαρακτήρας μπορεί να διαταχθεί από τον παίκτη να εκτελέσει την αντίστοιχη εντολή. Τέλος, η κλάση διαθέτει τη μέθοδο _on_go, για την πραγματοποίηση της μετακίνησης του χαρακτήρα μεταξύ δωματίων.

4.4.4 Δωμάτια - Οι κλάσεις Room και Door

Η κλάση rooms.Room περιλαμβάνει τα Δωμάτια του παιχνιδιού, τα οποία αντιπροσωπεύουν χώρους μέσα στους οποίους μπορεί να κινηθεί ο παίκτης. Κληρονομούν από την κλάση Entity, και έχουν ως επιπλέον χαρακτηριστικό το λεξικό directions, το οποίο περιέχει τις κατευθύνσεις από τις οποίες ο παίκτης μπορεί να εξέλθει από τον χώρο, τα κλειδιά του Δωματίου στο οποίο οδηγεί η κάθε κατεύθυνση, και μία περιγραφή. Η κλάση περιλαμβάνει επίσης τη μέθοδο string (παρόμοια με την αντίστοιχη μέθοδο της Thing), καθώς και τη μέθοδο __iand__, για το overloading του τελεστή '&=' - για την προσθήκη μίας επιπλέον κατεύθυνσης στο δωμάτιο.

Η κλάση rooms.Door κληρονομεί από την κλάση Room, αλλά περιέχει κάποιες ιδιότητες και μεθόδους των αντικειμένων Thing. Επιπλέον, διαθέτει την ιδιότητα active_direction, που υποδηλώνει την πλευρά της πόρτας στην οποία βρίσκεται ένας χαρακτήρας.

4.4.5 Ειδικά αντικείμενα

Άλλα ειδικά αντικείμενα που αναπτύχθηκαν για τις ανάγκες του παιχνιδιού είναι αυτά που περιέχονται στο αρχείο space_things.py, δηλαδή οι κλάσεις SolarSystem και Planet. Και οι δύο κλάσεις κληρονομούν από την κλάση Thing, αλλά περιέχουν επιπλέον attributes που εξηγούνται πιο αναλυτικά στο 4.10. Επιπλέον, η κλάση SolarSystem περιλαμβάνει τις κατάλληλες μεθόδους για την είσοδο και την έξοδο του παίκτη από ένα Ηλιακό Σύστημα, ενώ η κλάση Planet έχει τις μεθόδους που διαχειρίζονται την προσγείωση και την απογείωση του παίκτη σε έναν Πλανήτη.

4.5 Συντακτικός αναλυτής

Στο αρχείο my_parser.py βρίσκονται οι κλάσεις που διαχειρίζονται την συντακτική ανάλυση και ταυτοποίηση των συστατικών μερών των εντολών του χρήστη. Οι κλάσεις αυτές είναι ο PreParser, ο Parser και ο NounPhraseParser.

4.5.1 Πριν από τη συντακτική ανάλυση - Η κλάση Preparser

Ο PreParser είναι η κλάση υπεύθυνη για την προ επεξεργασία των εντολών του παίκτη, και την ταυτοποίηση της κατηγορίας εντολών στην οποία ανήκουν. Αρχικά, ο PreParser επεξεργάζεται μία εντολή που λαμβάνει, μετατρέποντας τα γράμματα του κειμένου σε lowercase και αφαιρώντας ειδικούς χαρακτήρες από το τέλος της συμβολοσειράς. Έπειτα, εάν η εντολή που δόθηκε αποτελεί καταχώρηση στον πίνακα προ επεξεργασίας (Grammar\preparse_table.json) τότε αντικαθίσταται από την τιμή της καταχώρησης στον πίνακα. Η χρησιμότητα του pre-parse table είναι ότι δίνει στον χρήστη τη δυνατότητα να δώσει συντομογραφίες εντολών, αντί για ολόκληρες εντολές. Τέλος, οι εντολές διαχωρίζονται σε Commands (εντολές οι οποίες απαιτούν περαιτέρω συντακτική ανάλυση) και σε ειδικές εντολές (save, undo, help, ή quit).

4.5.2 Συντακτικός αναλυτής - Η κλάση Parser

Ο συντακτικός αναλυτής ή Parser διαχειρίζεται τη συντακτική ανάλυση των εντολών του χρήστη, καθώς και την αντιστοίχιση των λέξεων της εντολής με Οντότητες και Ρήματα του παιχνιδιού. Ο Parser χρησιμοποιεί τον ChartParser της βιβλιοθήκης NLTK (βλ. 4.1.5) σε συνδυασμό με μία context-free τυπική γραμματική (τύπου 2). Οι κανόνες της γραμματικής που αναπτύχθηκε φαίνονται στον Πίν. 4.1. Ως τερματικά σύμβολα της γραμματικής χρησιμοποιήθηκαν αντί για λέξεις οι «ετικέτες» μέρους του λόγου (Part of Speech Tags, στο εξής POS Tags) που παρέχει η βιβλιοθήκη NLTK. Πλεονέκτημα της μεθόδου αυτής είναι ότι η γραμματική μπορεί να οριστεί με πολύ λίγους κανόνες και να είναι ευέλικτη για κάθε σύνολο ουσιαστικών, επιθέτων και ρημάτων που περιλαμβάνονται σε ένα παιχνίδι. Μειονέκτημά της είναι ότι περιορίζεται πρακτικά από την ακρίβεια του POS Tagger της NLTK (σε περιπτώσεις που δεν λειτουργεί σωστά χρειάζονται επιπλέον παρεμβάσεις ώστε να μπορεί να γίνει συντακτική ανάλυση των εντολών). Ο συντακτικός αναλυτής της συγκεκριμένης εφαρμογής δε χρειάζεται να αναγνωρίζει όλες τις μορφές προτάσεων της αγγλικής γλώσσας, παρά μόνο ένα μικρό υποσύνολο αυτών. Έτσι, οι δομές των προτάσεων που αναγνωρίζονται μπορεί να είναι είτε SV (Subject-Verb), είτε SV + επιθετικός/επιρρηματικός προσδιορισμός, είτε SVO (Subject-Verb-Object), είτε SVO + Indirect Object, είτε SVO + προθετικό σύνολο.

Πίν. 4.1: Κανόνες επανεγγραφής γραμματικής συντακτικού αναλυτή.

#	Κεφαλή		Σώμα
1	S	→	C V | C V O | C V O I | C V B | C V J
2	C	→	SB
3	V	→	VR | VR B | VR P
4	O	→	N
5	I	→	P N
6	N	→	NN | N Q | A N | J N | NN N
7	Q	→	L N
8	VR	→	vb | vbp | vbz | vbd
9	NN	→	nn | nnp | nns | nnps
10	SB	→	nn | nnp | nns | nnps | prp
11	B	→	rp | rb | in | rbr | rbs
12	P	→	to | in
13	J	→	jj | jjr | jjs
14	L	→	cc
15	A	→	dt | prp$ | wdt

Η λειτουργία του parser μπορεί να συνοψιστεί στα παρακάτω βήματα:

1. Tokenization (χωρισμός της εντολής σε λέξεις): Το κείμενο που δόθηκε από τον παίκτη χωρίζεται αρχικά σε ξεχωριστές προτάσεις - εντολές, κάθε μία από τις οποίες χωρίζεται σε λέξεις με τη βοήθεια του tokenizer της NLTK. Σε περίπτωση που η πρώτη λέξη της κάθε πρότασης δεν είναι είτε "I" είτε το όνομα ενός Non-Player Character, η λέξη "I" προστίθεται στην αρχή της πρότασης.
2. POS Tagging: Για κάθε πρόταση, πριν από την ταυτοποίηση του μέρους του λόγου των λέξεων της υπάρχει ένα βήμα προ επεξεργασίας για τη βελτίωση της τελικής συντακτικής ανάλυσης (κάποιες ειδικές λέξεις αντικαθίστανται προσωρινά με άλλες λέξεις-placeholders που αναγνωρίζονται σωστότερα από τον αναλυτή.). Έπειτα ακολουθεί το Part of Speech Tagging με την nltk.pos_tag().

3. Κατασκευή συντακτικού δέντρου: Τα POS tags που παράχθηκαν από το προηγούμενο στάδιο χρησιμοποιούνται ως είσοδος για τον συντακτικό αναλυτή της NLTK, με τη χρήση της ειδικής γραμματικής που ορίσαμε παραπάνω. Σε περίπτωση που η πρόταση δεν οδηγήσει στην κατασκευή κάποιου συντακτικού δέντρου, η διαδικασία τερματίζεται με ένα Σφάλμα Parser (ParserError). Τέλος, οι ετικέτες στα φύλλα του συντακτικού δέντρου αντικαθίστανται με ζεύγη της μορφής (<αρχική λέξη>, <POS Tag>).
4. Ταυτοποίηση συντακτικών μερών: Μετά τη δημιουργία των συντακτικών δέντρων, κάθε πρόταση χωρίζεται στα εξής τμήματα:
    a. "Actor", που αντιπροσωπεύει τον χαρακτήρα-υποκείμενο της πρότασης.
    b. "Verb", που αντιπροσωπεύει το ρήμα (ή ρηματική φράση) της πρότασης.
    c. "Object": η ονοματική φράση που περιλαμβάνει το/τα άμεσο/α αντικείμενο/α της πρότασης, αν υπάρχει.
    d. "Qualifier": μπορεί να είναι είτε κάποιος επιθετικός ή επιρρηματικός προσδιορισμός που ακολουθεί ένα ρήμα, είτε η πρόθεση πριν από το έμμεσο αντικείμενο της πρότασης, αν υπάρχει.
    e. "Indirect": η ονοματική φράση που συνοδεύει το/α έμμεσο/α αντικείμενο/α μία πρότασης, ή η ονοματική φράση ενός επιρρηματικού προσδιορισμού (π.χ. τοπικός προσδιορισμός), εάν υπάρχει.

    Σε αυτό το στάδιο γίνεται προσπάθεια να ταυτοποιηθούν ο Actor της πρότασης, το Verb και ο Qualifier, με βάση τα αντικείμενα που είναι ορισμένα στο παιχνίδι. Εάν δεν υπάρξει κάποιο ταίριασμα, το συντακτικό δέντρο απορρίπτεται. Τέλος, εξετάζεται αν η συντακτική δομή της πρότασης συμφωνεί με τους πιθανούς τύπους προτάσεων στις οποίες μπορεί να συμμετέχει το ρήμα της, αλλιώς το δέντρο απορρίπτεται. Εάν κανένα από τα συντακτικά δέντρα μίας πρότασης δε δώσει έγκυρο αποτέλεσμα, η πρόταση απορρίπτεται και η διαδικασία τερματίζεται με σφάλμα.

Τελικά, ο Parser επιστρέφει ένα λεξικό με τα συντακτικά μέρη της πρότασης, έτσι όπως αυτά ταυτοποιήθηκαν στο βήμα 4. Το επόμενο βήμα είναι η αποσαφήνιση των αντικειμένων

που περιέχονται στο "Object" και το "Indirect" μέρος της πρότασης. Για την διαδικασία αυτή υπεύθυνος είναι ο NounPhraseParser.

4.5.3 Αποσαφήνιση ονοματικών φράσεων

Η κλάση NounPhraseParser χρησιμοποιείται για την ανάλυση ονοματικών φράσεων σε μία πρόταση. Λειτουργεί διαφορετικά για τις προτάσεις διαλόγων (που περιέχουν ρήματα διαλόγου όπως το "Ask") και διαφορετικά σε απλές προτάσεις εντολών. Στην περίπτωση των προτάσεων εντολών, τόσο για το άμεσο όσο και για το έμμεσο αντικείμενο (εάν υπάρχουν), γίνεται η ίδια ανάλυση της ονοματικής φράσης:

1. Αρχικά οι φράσεις χωρίζονται σε επιμέρους ονοματικές φράσεις, οι οποίες συνδέονται μεταξύ τους με τον σύνδεσμο 'and'. Για κάθε επιμέρους φράση, η τελευταία της λέξη αποτελεί το ουσιαστικό της, και όσες λέξεις βρίσκονται μπροστά από το ουσιαστικό (εκτός από κάποιο άρθρο), χαρακτηρίζονται ως επίθετα του ουσιαστικού.
2. Έπειτα, το ουσιαστικό κάθε ονοματικής φράσης αναζητείται μεταξύ των reference nouns των οντοτήτων που υπάρχουν στο λεξικό Game.things. Σε περίπτωση ταιριάσματος, εξετάζεται το κατά πόσο τα επίθετα της φράσης ταυτίζονται με αυτά που αναμένεται ότι θα χαρακτηρίζουν το αντικείμενο με το οποίο έγινε το ταίριασμα (από την ιδιότητα reference_adjectives).
3. Σε περίπτωση που υπάρξει ολικό ταίριασμα όλων των επιμέρους ονοματικών φράσεων με ένα ή περισσότερα αντικείμενα του παιχνιδιού, επιστρέφεται μία λίστα με τις αντίστοιχες οντότητες. Αλλιώς η διαδικασία τερματίζει με σφάλμα.
4. Σε περίπτωση που μία ονοματική φράση ταιριάξει με περισσότερα από ένα αντικείμενα, ζητείται από τον παίκτη να αποσαφηνίσει το αντικείμενο το οποίο θέλει να επιλέξει, με την εκτύπωση όλων των επιλογών που ταιριάζουν στην εντολή που έδωσε.

Η διαδικασία που ακολουθείται για τις προτάσεις διαλόγου είναι σχεδόν πανομοιότυπη, με τη μόνη διαφορά ότι σε αυτήν την περίπτωση η ονοματική φράση του άμεσου αντικειμένου προσπαθεί να αντιστοιχισθεί με κάποιον από τους χαρακτήρες του παιχνιδιού, και του έμμεσου αντικειμένου με κάποιο θέμα διαλόγου. Τελικά, και στις δύο περιπτώσεις, τα μέρη

της πρότασης Object και Indirect αντικαθίστανται από μία λίστα που περιέχει τα αντικείμενα του παιχνιδιού με τα οποία ταιριάζουν.

4.6 Εντολές και Πράξεις

4.6.1 Ρήματα - Η κλάση Verb

Τα αντικείμενα της κλάσης Verb εκφράζουν όλες τις δυνατές πράξεις τις οποίες μπορεί να κάνει ο παίκτης στο παιχνίδι. Η κλάση έχει τις εξής ιδιότητες:

- forms: Μία λίστα από συμβολοσειρές, κάθε μία από τις οποίες εκφράζει μία πιθανή μορφή με την οποία το ρήμα μπορεί να εμφανιστεί στις εντολές του παίκτη.
- patterns: Οι κατηγορίες προτάσεων στις οποίες μπορεί να εμφανιστεί το ρήμα. Οι κατηγορίες χαρακτηρίζονται από τη μορφή του κατηγορήματος της πρότασης, και είναι:
    - '' ή null → Μία πρόταση στην οποία μόνο το ρήμα αποτελεί μέρος του κατηγορήματος, π.χ. 'Look'.
    - 'O' → Μία πρόταση στην οποία το ρήμα ακολουθείται από ένα ή περισσότερα άμεσα αντικείμενα, π.χ. 'Look at the red button'.
    - 'Q' → Μία πρόταση στην οποία το ρήμα ακολουθείται από κάποιον προσδιορισμό (σύνδεσμο, επίρρημα ή επίθετο), π.χ. 'Go north'.
    - 'OQI' → Μία πρόταση με άμεσο αντικείμενο και είτε έμμεσο αντικείμενο ή έναν προθετικό σύνολο (μία πρόθεση ακολουθούμενη από μία ονοματική φράση), π.χ. 'Put the empty glass on the table'.
- name: Το όνομα του ρήματος, το οποίο χαρακτηρίζει μοναδικά το ρήμα (το κλειδί του).
- description: Μία περιγραφή της πράξης που εκφράζει το ρήμα.

Στον Πίν. 4.2 βλέπουμε την πλήρη λίστα των ρημάτων του παιχνιδιού.

Πίν. 4.2: Κατάλογος ρημάτων του παιχνιδιού.

#	name	forms	patterns	Περιγραφή
1	Go	go	Q	Μετακίνηση του Actor προς την κατεύθυνση που ορίζεται από τον qualifier.
2	Look	look, look at, examine, look around, x	null, O	Όταν εφαρμόζεται σε κάποιο αντικείμενο, τυπώνεται η αναλυτική περιγραφή του αντικειμένου. Αλλιώς, τυπώνεται η περιγραφή του δωματίου.
3	Listen	listen, listen to	null, O	Όταν εφαρμόζεται σε κάποιο αντικείμενο, τυπώνεται η αναλυτική ηχητική περιγραφή του αντικειμένου. Αλλιώς, τυπώνεται η ηχητική περιγραφή του δωματίου.
4	Take	take	O	Ο Actor κρατάει το αντικείμενο.
5	Drop	drop	O	Ο Actor αφήνει το αντικείμενο.
6	Open	open, open up	O	Το αντικείμενο στο οποίο εφαρμόζεται είναι πλέον ανοιχτό.
7	Close	close	O	Το αντικείμενο στο οποίο εφαρμόζεται είναι πλέον κλειστό.
8	Push	push	O	Το αντικείμενο πιέζεται (ειδικά σε ένα κουμπί μπορεί να πυροδοτείται κάποια διαδικασία).
9	Use	use	O, OQI	Χρήση ενός αντικειμένου (το είδος χρήσης εξαρτάται από το αντικείμενο στο οποίο εφαρμόζεται). Κάποιο αντικείμενο μπορεί να χρησιμοποιηθεί σε κάποιο άλλο αντικείμενο.
10	Read	read	O	Ο χρήστης διαβάζει ένα αντικείμενο.
11	Put	put	OQI	Τοποθέτηση ενός αντικειμένου πάνω/μέσα σε/κάτω από κάποιο άλλο.
12	Send	send	OQI	Αποστολή μη επανδρωμένων σκαφών σε κάποιο σώμα για εξερεύνηση. Τα σκάφη επιστρέφουν μια περιγραφή του αντικειμένου.
13	Landon	land on	O	Ο παίκτης προσγειώνεται στην επιφάνεια κάποιου Ουράνιου Σώματος.
14	Takeoff	take off	null	Απογείωση του σκάφους από τον πλανήτη στον οποίο βρίσκεται ο παίκτης.
15	Enter	enter	O	Είσοδος του παίκτη σε κάποιο ηλιακό σύστημα.
16	Leave	leave, exit	O	Έξοδος ή απομάκρυνση από το ηλιακό σύστημα στο οποίο βρίσκεται ή πλησιάζει το σκάφος.
17	Setup	set up, build	O	Κατασκευή αποικίας στην επιφάνεια ενός πλανήτη.
18	Ask	ask, tell	OQI	Ξεκίνα συνομιλία με κάποιον Actor. Το αντικείμενο της πρότασης πρέπει να είναι κάποιος NPC, ο qualifier να είναι η λέξη ‘about’, και το I κάποιο θέμα συζήτησης (βλ. 4.8.1).

4.6.2 Συνθήκες και έλεγχος εντολών - Η κλάση Precondition

Μετά τη συντακτική ανάλυση της εντολής του παίκτη (βλ. 4.5), κάθε πρόταση της εντολής δίνεται ως είσοδος σε ένα αντικείμενο της κλάσης CommandHandler. Η κλάση αυτή επιφορτίζεται με τον έλεγχο της εγκυρότητας της εντολής, καθώς και με την εκτέλεσή της. Τα βήματα για τον έλεγχο της εγκυρότητας μίας εντολής είναι τα εξής:

1. Εάν η εντολή έχει παραπάνω από μία οντότητες στο πεδίο "Object" ή στο πεδίο "Indirect", τότε η εντολή «ξεδιπλώνεται» σε πολλές εντολές, μία για κάθε ξεχωριστή οντότητα-αντικείμενο. Μία εντολή δεν μπορεί να έχει πολλά Objects και Indirects ταυτόχρονα, π.χ. 'Put the glass and the plate on the table and the stove.'
2. Έπειτα, γίνεται διαχωρισμός της πρότασης με βάση του εάν η εντολή απευθύνεται στον Player Character ή σε κάποιον NPC (Non-Player Character). Στην περίπτωση του NPC εξετάζεται κατά πόσο ο χαρακτήρας βρίσκεται στο ίδιο Δωμάτιο με τον παίκτη και κατά πόσο μπορεί να διαταχθεί να κάνει την συγκεκριμένη εντολή. Και στις δύο περιπτώσεις, όμως, εξετάζεται το κατά πόσο ο χαρακτήρας έχει την ικανότητα να εκτελέσει την συγκεκριμένη δράση (με βάση το λεξικό του abilities). Σε περίπτωση που παραβιάζεται οποιαδήποτε συνθήκη, η διαδικασία σταματάει με σφάλμα τύπου CheckCommandError.
3. Στη συνέχεια ελέγχεται ο Qualifier της πρότασης (εάν υπάρχει) για το κατά πόσο μπορεί να χρησιμοποιηθεί με το συγκεκριμένο Ρήμα, ή με το συγκεκριμένο Indirect αντικείμενο (εάν υπάρχει). Επίσης, εάν πρόκειται για εντολή Go, εξετάζεται το κατά πόσο ο Qualifier υπάρχει μέσα στις έγκυρες κατευθύνσεις του δωματίου στο οποίο βρίσκεται ο χαρακτήρας. Σε περίπτωση που κάποια συνθήκη δεν ισχύει έχουμε και πάλι σφάλμα τύπου CheckCommandError.
4. Επιπλέον, εξετάζεται το κατά πόσο το Ρήμα της πρότασης μπορεί να εφαρμοστεί στο άμεσο αντικείμενο της πρότασης (εάν υπάρχει), με βάση το λεξικό as_dirobj.
5. Τέλος, ελέγχεται η Προϋπάρχουσα Συνθήκη (Precondition, βλ. παρακάτω), η οποία αντιστοιχεί στην συγκεκριμένη πρόταση. Εάν η συνθήκη αποτύχει, τότε η διαδικασία τερματίζει με PreconditionsError.

Η κλάση preconditions.Precondition χρησιμοποιείται για τον έλεγχο της εγκυρότητας μίας εντολής, εφόσον η εντολή αυτή είναι έγκυρη συντακτικά και μπορεί να εφαρμοστεί στα αντικείμενα τα οποία περιέχει. Η κλάση περιέχει μεθόδους για κάθε ένα από τα ρήματα του παιχνιδιού. Κάθε μέθοδος της επιστρέφει είτε True, εφόσον η εντολή μπορεί να εκτελεστεί, είτε False στην αντίθετη περίπτωση. Οι συνθήκες που περιγράφονται σε κάθε μέθοδο μπορεί να σχετίζονται με το είδος του αντικειμένου στο οποίο εφαρμόζεται η πράξη, με το είδος του qualifier που χρησιμοποιείται, με κάποια συνθήκη του αντικειμένου, ή γενικά με οποιονδήποτε λόγο μπορεί να απαγορεύει την εκτέλεση της πράξης. Οι συνθήκες μπορούν να επεκταθούν σε περίπτωση που χρειάζεται να γίνουν πιο περίπλοκες στο μέλλον, καθώς και να προστεθούν καινούργια μηνύματα λάθους σε περίπτωση που μία εντολή δεν μπορεί να εκτελεστεί.

4.6.3 Εκτέλεση εντολής

Αφού εξεταστούν όλες οι συνθήκες μίας εντολής, μπορεί να προχωρήσει η εκτέλεσή της. Σε περίπτωση που μία εντολή περιέχει κάποιο ρήμα διαλόγου, η εκτέλεση της εντολής περνάει στην μέθοδο Topic.on_topic του θέματος συζήτησης. Η εντολή μπορεί και εκεί να αποτύχει (όταν π.χ. δεν υπάρχει κάποια απάντηση του NPC για το θέμα της ερώτησης), οπότε και πυροδοτείται ένα DialogError (για περισσότερες λεπτομέρειες βλ. 4.8).

Σε περίπτωση που η εντολή αφορά κάποια πράξη σε Οντότητα, η εκτέλεση της πράξης γίνεται με διαφορετικούς τρόπους, ανάλογα με τον τύπο της πρότασης:

- Για προτάσεις τύπου null: Σε αυτού του είδους τις προτάσεις η εκτέλεση της εντολής γίνεται από την μέθοδο on_dir_object του Δωματίου στο οποίο βρίσκεται ο χαρακτήρας του παίκτη εκείνη τη στιγμή. Στην ειδική περίπτωση του 'Takeoff' η πράξη εκτελείται από την αντίστοιχη μέθοδο του Actor της πρότασης.
- Για προτάσεις τύπου Q: Σε αυτή την κατηγορία προτάσεων ανήκει μόνο η περίπτωση του ρήματος Go. Η εντολή εκτελείται από τη μέθοδο on_dir_object του Actor της πρότασης.
- Για προτάσεις τύπου O: Η εντολή εκτελείται από την μέθοδο on_dir_object του άμεσου αντικειμένου της εντολής. Όπως εξηγείται στο 4.4.1, η κλάση Entity και οι υποκλάσεις της διαθέτουν μεθόδους για την εκτέλεση εντολών - με όνομα της

μορφής _on_<VERB>. Κάθε τέτοια μέθοδος περιέχει τις απαραίτητες εντολές για να αλλάξει την κατάσταση του αντικειμένου ανάλογα με την εντολή που δόθηκε, δηλαδή, για παράδειγμα, να αλλάξει το δοχείο στο οποίο περιέχεται, να ενημερώσει το λεξικό entity_state της Οντότητας κλπ. Τέλος, από την μέθοδο αυτή επιστρέφεται και η περιγραφή της πράξης, με βάση το λεξικό action_description του αντικειμένου.

- Για προτάσεις τύπου OQI: Παρόμοια με τις προτάσεις τύπου O.

4.7 Γεγονότα

Γεγονότα, στα πλαίσια της παρούσας εφαρμογής, καλούνται οι αλλαγές που πραγματοποιούνται στην κατάσταση του κόσμου του παιχνιδιού που δεν είναι άμεση απόρροια των πράξεων του παίκτη, αλλά συντελούνται αυτόματα με βάση κάποιες αρχικές συνθήκες.

4.7.1 Η κλάση Event

Η κλάση που διαχειρίζεται τα γεγονότα στην παρούσα εφαρμογή είναι η κλάση events.Event. Κάθε αντικείμενο της κλάσης έχει ως ιδιότητες:

- ένα μοναδικό κλειδί-αναγνωριστικό (key)
- ένα σύνολο από αρχικές συνθήκες (trigger conditions), όλες από τις οποίες πρέπει να ισχύουν για να πραγματοποιηθεί το γεγονός
- ένα σύνολο από αλλαγές που το γεγονός επιφέρει στον κόσμο του παιχνιδιού (game state changes)
- μία περιγραφή που τυπώνεται κάθε φορά που το γεγονός αυτό συντελείται
- έναν μετρητή ο οποίος καταγράφει πόσες φορές έχει πραγματοποιηθεί το γεγονός
- το χαρακτηριστικό repeatable, που υποδηλώνει πόσες φορές μπορεί να επαναληφθεί το ίδιο γεγονός
- το attribute next_event_key το οποίο περιέχει το αναγνωριστικό κλειδί ενός γεγονότος που συντελείται ως συνέπεια του παρόντος γεγονότος (εάν υπάρχει).

Η κλάση περιέχει επίσης τις μεθόδους:

- trigger: Καλείται όταν το γεγονός συντελείται, ενώ επιστρέφει την περιγραφή του ίδιου του γεγονότος και όσων άλλων γεγονότων πιθανόν πραγματοποιούνται ως συνέπειά του.
- eval_conditions: Επιστρέφει True εάν όλες οι αρχικές συνθήκες που επιφέρουν το γεγονός ισχύουν.
- get_score: Επιστρέφει τον αριθμό των συνθηκών του γεγονότος. Η συνάρτηση αυτή μπορεί να χρησιμεύσει στην ιεράρχηση των γεγονότων, από το πιο συγκεκριμένο (αυτό με τις περισσότερες συνθήκες) μέχρι το λιγότερο συγκεκριμένο (αυτό με τις περισσότερες).
- _change_game_state: Προκαλεί τις αλλαγές του κόσμου του παιχνιδιού.
- _trigger_next: Πυροδοτεί το επόμενο γεγονός (σε περίπτωση που υπάρχει).
- to_json: Επιστρέφει μία μορφή του αντικειμένου που είναι σειριοποιήσιμη σε αρχείο τύπου JSON.

4.7.2 Συνθήκες - Οι κλάσεις Condition και GameQuery

Τα στιγμιότυπα της κλάσης Event, όπως και όλα τα άλλα στοιχεία του παιχνιδιού, πρέπει να μπορούν να αποθηκευτούν εύκολα σε αρχεία JSON, και η δημιουργία νέων γεγονότων πρέπει να είναι μία διαδικασία που δεν απαιτεί απαραίτητα τη συγγραφή επιπλέον κώδικα. Για τον λόγο αυτό, οι αρχικές συνθήκες ενός γεγονότος και οι αλλαγές στην κατάσταση του παιχνιδιού πρέπει να κωδικοποιηθούν με κατάλληλο τρόπο. Όσον αφορά τις αρχικές συνθήκες των γεγονότων τον ρόλο αυτό τον αναλαμβάνει η κλάση conditions.Condition.

Οι αρχικές συνθήκες είναι ουσιαστικά ζεύγη αντικειμένων - τιμών, τα οποία πρέπει να αξιολογηθούν κατά την λειτουργία του παιχνιδιού είτε ως True είτε ως False. Για τον σκοπό αυτό η κλάση Condition έχει τις ιδιότητες attribute_path και values. Η attribute_path είναι μία συμβολοσειρά η οποία καθορίζει ένα αντικείμενο το οποίο βρίσκεται στην κλάση Game ή αποτελεί αντικείμενο κάποιου άλλου αντικειμένου της κλάσης Game. Η συμβολοσειρά «δείχνει», λοιπόν, το «μονοπάτι» του αντικειμένου σε μορφή dot notation. Κάθε στοιχείο της συμβολοσειράς μεταξύ δύο τελειών αποτελεί ένα attribute, ένα κλειδί σε λεξικό Python ή έναν δείκτη σε Python list ή tuple. Η ιδιότητα values, από την άλλη, είναι μία λίστα τιμών,

μία από τις οποίες πρέπει να έχει η μεταβλητή που ορίζεται από το attribute_path, ώστε η συνθήκη να θεωρηθεί αληθής. Οι τιμές που περιέχονται στη λίστα μπορούν να είναι αριθμοί, συμβολοσειρές, Boolean τιμές, ή μία λίστα 2 αριθμών που ορίζουν ένα ανοιχτό διάστημα μέσα στο οποίο η συνθήκη θεωρείται αληθής. Τέλος, η κλάση διαθέτει και την ιδιότητα not_, η οποία είναι αληθής όταν τιμή αλήθειας της συνθήκης αντιστρέφεται μετά από την αξιολόγησή της.

Για την εύρεση της μεταβλητής που καθορίζεται από το attribute_path υπεύθυνη είναι η κλάση game_queries.GameQuery. Ξεκινώντας από το αντικείμενο game, η κλάση επιστρέφει μία λίστα με τις τιμές όλων των «ενδιάμεσων» αντικειμένων στο μονοπάτι, ή None, σε περίπτωση που η μεταβλητή αυτή δεν βρεθεί. Η κλάση μπορεί να αναζητήσει είτε attributes ενός αντικειμένου, είτε κλειδιά ενός Python dictionary, είτε δείκτες σε μία λίστα ή πλειάδα.

4.7.3 Αλλαγές στην κατάσταση του παιχνιδιού - Η κλάση StateUpdate

Οι αλλαγές που επιφέρει ένα γεγονός στην κατάσταση του κόσμου του παιχνιδιού καθορίζονται από την κλάση state_updates.StateUpdate. Κάθε αντικείμενο της κλάσης, έχει, όπως και στην περίπτωση των condition objects, μία ιδιότητα attribute_path, καθώς και την ιδιότητα new_value_expression. Η πρώτη έχει την ίδια μορφή με την αντίστοιχη ιδιότητα της κλάσης Condition, ενώ η δεύτερη αποτελεί μία έκφραση κωδικοποιημένη σε συμβολοσειρά, η οποία αξιολογείται δυναμικά με τη χρήση της συνάρτησης eval() της Python.

Παρά τις γνωστές αδυναμίες της συνάρτησης eval, και τους κινδύνους που μπορεί να δημιουργήσει στην ασφάλεια του συστήματος, ορισμένες βελτιώσεις μπορούν να κάνουν την χρήση της πολύ πιο ασφαλή, με τον περιορισμό των ονομάτων και των μεθόδων στις οποίες έχει πρόσβαση. Το πλεονέκτημα της χρήσης της eval στην συγκεκριμένη περίπτωση είναι ότι επιτρέπει στον προγραμματιστή του παιχνιδιού να γράψει γεγονότα τα οποία όχι απλά προκαλούν μία αλλαγή στην κατάσταση του παιχνιδιού, αλλά π.χ. μπορούν ακόμα και να πολλαπλασιάσουν την τωρινή τιμή μίας μεταβλητής με έναν αριθμό δυναμικά, κατά την εκτέλεση δηλαδή του προγράμματος, χωρίς την συγγραφή επιπλέον κώδικα. Τέλος, η κλάση StateUpdate διαθέτει μεθόδους που διαχειρίζονται ιδιαίτερες αλλαγές στον κόσμο του παιχνιδιού (όπως για παράδειγμα την τοποθέτηση ενός αντικειμένου σε ένα άλλο).

4.8 Διάλογοι

Οι δυναμικοί διάλογοι αποτελούν έναν από τους σημαντικότερους μηχανισμούς του παιχνιδιού που αναπτύχθηκε κατά τη διάρκεια της παρούσας διπλωματικής εργασίας. Αποτελούν ένα στοιχείο πέρα από την αφήγηση το οποίο μπορεί να κρατήσει την προσοχή του παίκτη, να δώσει ποικιλία στην εμπειρία του δραματοποιώντας τα γεγονότα του παιχνιδιού, καθώς και να παρέχει στον παίκτη πολύτιμα στοιχεία για την επίλυση γρίφων. Τους διαλόγους διαχειρίζονται οι κλάσεις topics.Topic, quips.Quip, dialog_events.DialogEvent και convo_nodes.ConvoNode.

4.8.1 Θέματα - Η κλάση Topic

Η κλάση Topic αντιπροσωπεύει τα θέματα τα οποία διατίθενται προς συζήτηση στον παίκτη. Διαθέτει, όπως οι περισσότερες άλλες κλάσεις του παιχνιδιού, μία ιδιότητα-κλειδί, καθώς και ουσιαστικό και επίθετα αναφοράς, όπως και η κλάση Thing. Κάθε αντικείμενο της κλάσης έχει επίσης μία συμβολοσειρά η οποία παρουσιάζει την ατάκα του χαρακτήρα του παίκτη, κάθε φορά που αναφέρεται σε αυτό το θέμα. Τέλος, τα θέματα μπορεί να είναι ενεργά (γνωστά δηλαδή στον παίκτη), εφόσον η ιδιότητά τους is_active είναι αληθής.

Η κλάση διαθέτει την μέθοδο on_topic, η οποία εκτελείται κάθε φορά που ο παίκτης αναφέρεται στο συγκεκριμένο θέμα. Για να δοθεί απάντηση στην ατάκα του παίκτη, η διαχείριση του θέματος περνάει στον κόμβο διαλόγου στον οποίο βρίσκεται ο Non Player χαρακτήρας στον οποίο απευθύνεται ο παίκτης (περισσότερα στο 4.8.2 και στο 4.8.3).

4.8.2 Γεγονότα διαλόγων - Η κλάση DialogEvent

Τα γεγονότα διαλόγου (Dialog Events) είναι γεγονότα που εκτελούνται για την παραγωγή απαντήσεων σε έναν διάλογο με κάποιον NPC. Η κλάση τους κληρονομεί από την κλάση Event (βλ. 4.7.1). Πέρα από τις ιδιότητες και τις μεθόδους τις οποίες διαθέτει μέσω κληρονομικότητας, η κλάση έχει ως επιπλέον παραμέτρους:

- trigger_topics: Μία λίστα με τα κλειδιά των θεμάτων διαλόγου που μπορούν να προκαλέσουν το γεγονός.

- quips: Μία λίστα των αντικειμένων της κλάσης quips.Quip, ατάκες δηλαδή που δίνουν απάντηση στις εντολές ask του παίκτη. Κάθε quip object διαθέτει ένα κείμενο-απάντηση, μία ιδιότητα που ορίζει το κατά πόσο έχει ήδη ειπωθεί, και μία ιδιότητα που καθορίζει αν είναι επαναλήψιμο ή όχι.
- trigger_convonode: Το κλειδί του κόμβου διαλόγου στον οποίο πρέπει να βρίσκεται ο NPC για να πραγματοποιηθεί το γεγονός. Αν έχει την τιμή None, τότε οποιοσδήποτε κόμβος επαρκεί.
- shuffle_quips: Αν έχει την τιμή True, τότε τα quips του γεγονότος μπορούν να ειπωθούν με τυχαία σειρά.

Όπως και τα αντικείμενα της κλάσης Event, έτσι και τα DialogEvent objects μπορούν να προκαλέσουν αλλαγές στην κατάσταση του παιχνιδιού, και να διαθέτουν αρχικές συνθήκες που πρέπει να ισχύουν ώστε να πραγματοποιηθούν. Σε περίπτωση που ικανοποιούνται οι συνθήκες για παραπάνω από ένα γεγονός διαλόγου ταυτόχρονα, επιλέγεται αυτό με τις περισσότερες συνθήκες, δηλαδή, το πιο συγκεκριμένο.

4.8.3 Δέντρα διαλόγων - Η κλάση ConvoNode

Τα δέντρα διαλόγων προσθέτουν ένα επιπλέον στρώμα πολυπλοκότητας στον μηχανισμό διαλόγων του παιχνιδιού. Κάθε NPC στο παιχνίδι διαθέτει το δικό του δέντρο διαλόγων για κάθε σκηνή. Οι κόμβοι των δέντρων αυτών αντιπροσωπεύονται από την κλάση ConvoNode. Κάθε κόμβος σχετίζεται με έναν συγκεκριμένο χαρακτήρα του παιχνιδιού, και διαθέτει έναν πίνακα μεταβάσεων. Στον πίνακα αυτόν (ο οποίος έχει τη μορφή λεξικού), κάθε επόμενος κόμβος δίνεται μαζί με ένα σύνολο θεμάτων που μπορούν να προκαλέσουν τη μετάβαση, εφόσον αναφερθούν από τον παίκτη.

Κάθε NPC διαθέτει τουλάχιστον 3 κόμβους συζήτησης:

- ready: Ο κόμβος στον οποίο βρίσκεται ο χαρακτήρας όταν δεν υπάρχει κάποιος ενεργός διάλογος.
- hey: Ο κόμβος στον οποίο βρίσκεται ο χαρακτήρας όταν ο παίκτης εκκινήσει έναν καινούργιο διάλογο μαζί του. Αυτός ο κόμβος υπάρχει έτσι ώστε να δώσει έναν πιο

ρεαλιστικό χαρακτήρα στους διαλόγους, με μηχανισμούς χαιρετισμών στην αρχή κάθε συζήτησης.

- generic: Ο default κόμβος κατά τη διάρκεια της συζήτησης.

Ένα σύστημα διαλόγων μπορεί να δημιουργηθεί απλά με τη χρήση των 3 παραπάνω κόμβων. Όμως, ένα πιο εκτεταμένο δέντρο διαλόγων μπορεί να μοντελοποιήσει μία πιο περίπλοκη συμπεριφορά των Non Player χαρακτήρων (π.χ. συναισθηματικές μεταβάσεις).

Σε περίπτωση που ένα θέμα διαλόγου δε βρίσκεται μέσα στις συνθήκες εξόδου ενός κόμβου διαλόγων, ο NPC μεταβαίνει στον κόμβο generic, εφόσον η ιδιότητα generic_not_allowed του κόμβου είναι False. Σε αντίθετη περίπτωση, ο NPC μπορεί να δώσει μία απάντηση που υποδηλώνει ότι περιμένει την αναφορά ενός συγκεκριμένου θέματος από τον παίκτη. Αυτός ο μηχανισμός μπορεί να χρησιμοποιηθεί για να δημιουργήσει πιο διαδραστικές συζητήσεις, αφού, για παράδειγμα, μπορεί να ζητηθεί από τον παίκτη να δώσει μία μονολεκτική απάντηση σε μία ερώτηση ενός NPC - και μάλιστα με τρόπο που φαίνεται «φυσικός» στη ροή της συζήτησης.

4.9 Κεφάλαια

Τα κεφάλαια στο παιχνίδι διαδικαστικής αφήγησης το οποίο αναπτύχθηκε δεν αποτελούν μόνο θεματικές ενότητες, αλλά βοηθούν στη διαχείριση των αντικειμένων και των Δωματίων στα οποία μπορεί να περιηγηθεί ο παίκτης, των θεμάτων συζήτησης, αλλά και των γεγονότων του παιχνιδιού. Ως εκ τούτου, τα Κεφάλαια μπορούν να θεωρηθούν ως συλλογές Δωματίων, Θεμάτων και Γεγονότων.

4.9.1 Η κλάση Chapter

Η κλάση Chapter είναι η υπερκλάση όλων των κλάσεων-κεφαλαίων του παιχνιδιού. Κάθε κεφάλαιο διαθέτει μία εισαγωγική περιγραφή, μία εναλλακτική εισαγωγική περιγραφή και μία περιγραφή τέλους, καθώς και λίστες που περιέχουν τα ονόματα των δωματίων, των κόμβων διαλόγου, των γεγονότων, των γεγονότων διαλόγου, καθώς και των θεμάτων που εμφανίζονται στην συγκεκριμένη σκηνή του παιχνιδιού. Επιπλέον ιδιότητες του κεφαλαίου είναι οι συνθήκες τέλους του κεφαλαίου, καθώς και το λεξικό chapter_state, το οποίο μπορεί να

χρησιμοποιηθεί για να την εισαγωγή νέων ιδιοτήτων στο αντικείμενο. Η κλάση διαθέτει μεθόδους για την αρχή του κεφαλαίου, το τέλος του, καθώς και για την συνέχιση της πλοκής (start_chapter, advance_chapter και _end_chapter αντίστοιχα). Η μέθοδος advance_chapter εξετάζει τις συνθήκες των γεγονότων του κεφαλαίου και προκαλεί ένα γεγονός εφόσον οι δικές του ικανοποιούνται. Ακόμα, διαχειρίζεται την μετάβαση μεταξύ κεφαλαίων. Η μετάβαση γίνεται εφόσον όλες οι συνθήκες τέλους του κεφαλαίου είναι αληθείς (όπως ισχύει και για την κλάση Event).

4.9.2 Οι υποκλάσεις της Chapter

Για τις ανάγκες της εφαρμογής, στο ίδιο αρχείο (chapters.py) ορίζονται άλλες κλάσεις που κληρονομούν από την Chapter, για την υλοποίηση ειδικών συμπεριφορών από τα κεφάλαια του παιχνιδιού. Οι υποκλάσεις αυτές είναι:

- IntroChapter: Αποτελεί το αρχικό κεφάλαιο για κάθε πιθανό παιχνίδι που μπορεί να αναπτυχθεί μέσα από τη συγκεκριμένη εφαρμογή.
- IntroChapterUnionColonizer: Διαχειρίζεται την έναρξη του παιχνιδιού της παρούσας εργασίας (βλ. Κεφάλαιο 5) και την αρχικοποίηση της διαδικαστικής παραγωγής της ιστορίας.
- EndChapter: Το τελευταίο κεφάλαιο κάθε παιχνιδιού.
- DeathChapter: Το κεφάλαιο που ενεργοποιείται σε περίπτωση που ο παίκτης χάσει στο παιχνίδι. Γίνεται άμεση μετάβαση στο EndChapter.
- WinChapter: Ενεργοποιείται σε περίπτωση νίκης. Γίνεται άμεση μετάβαση στο EndChapter.
- SpaceChapter: Το κεφάλαιο που ακολουθεί πάντα την εισαγωγή στο συγκεκριμένο παιχνίδι. Σε αυτό γίνεται η διαχείριση της διαδικαστικής παραγωγής Ηλιακών Συστημάτων, ενώ ο παίκτης μπορεί παράλληλα να εξερευνήσει το διαστημόπλοιο στο οποίο βρίσκεται στην αρχή του κεφαλαίου (βλ. 5.1)
- SolarSystemChapter: Το κεφάλαιο που ενεργοποιείται κατά την είσοδο του παίκτη σε ένα Ηλιακό Σύστημα.
- PlanetChapter: Το κεφάλαιο που ενεργοποιείται μετά την προσγείωση του παίκτη σε κάποιον πλανήτη.

- ColonyChapter: Το κεφάλαιο που ενεργοποιείται σε περίπτωση που ο παίκτης επιλέξει να χτίσει μια αποικία στην επιφάνεια ενός πλανήτη. Μετά το κεφάλαιο αυτό ακολουθούν άμεσα, είτε το κεφάλαιο νίκης, είτε το κεφάλαιο θανάτου.

4.10 Διαδικαστική παραγωγή

Οι μηχανισμοί που αναλύθηκαν παραπάνω συνεργάζονται για να καταστήσουν το παιχνίδι διαδραστικό, δηλαδή να δώσουν τη δυνατότητα στον παίκτη να αλληλοεπιδρά με τα αντικείμενα του κόσμου του παιχνιδιού. Επιπλέον, οι μηχανισμοί διαλόγου μπορούν να χρησιμοποιηθούν για να δημιουργήσουν συζητήσεις με Non Player χαρακτήρες και να ενισχύσουν αυτό το στοιχείο διαδραστικότητας. Για να γίνει η αφήγηση πιο δυναμική και να αποδοθεί μεγαλύτερη ποικιλία στον κόσμο του παιχνιδιού για την εξερεύνηση του παίκτη, δημιουργήθηκαν, επιπλέον, κάποιοι μηχανισμοί διαδικαστικής παραγωγής αντικειμένων και περιγραφών στο παιχνίδι.

4.10.1 Διαδικαστική παραγωγή ειδικών αντικειμένων - Ηλιακά Συστήματα και Πλανήτες

Όπως εξηγείται εκτενέστερα στο κεφάλαιο 5, η υπόθεση του παιχνιδιού που αναπτύχθηκε επικεντρώνεται στην εξερεύνηση του διαστήματος από τον χαρακτήρα του παίκτη. Για την εξερεύνηση αυτή είναι απαραίτητη η διαδικαστική παραγωγή στιγμιότυπων των ειδικών κλάσεων SolarSystem και Planet (βλ. 4.4.5). Τα αντικείμενα αυτά διαθέτουν διάφορες εξειδικευμένες ιδιότητες, οι οποίες καθορίζουν τόσο την περιγραφή τους, όσο και τις δυνατότητες που έχει ο παίκτης κατά την εξερεύνησή τους. Για την παραγωγή αυτών των αντικειμένων αναπτύχθηκαν οι κλάσεις SolarSystemGenerator, PlanetGenerator και PlanetDescriptionGenerator. Όλα τα χαρακτηριστικά των αντικειμένων αυτών επιλέγονται μέσω της γεννήτριας παραγωγής ψευδοτυχαίων αριθμών της numpy.random, και όλες οι περιγραφές τους είναι επαναλήψιμες για έναν συγκεκριμένο seed παιχνιδιού.

Ο SolarSystemGenerator χρησιμοποιείται για την διαδικαστική παραγωγή των χαρακτηριστικών και της περιγραφής διαφορετικών ηλιακών συστημάτων με τον εξής τρόπο:

1. Στην αρχή του παιχνιδιού επιλέγεται με βάση τον «σπόρο» (seed) του συγκεκριμένου run ο αστερισμός προς τον οποίο ταξιδεύει ο παίκτης.
2. Στη συνέχεια, ανά τακτά χρονικά διαστήματα, εφόσον ο παίκτης δεν έχει εισέλθει σε κάποιο Ηλιακό Σύστημα ήδη, παράγεται τυχαία ένα ηλιακό σύστημα που βρίσκεται σε αυτόν τον αστερισμό. Κάθε ηλιακό σύστημα που παράγεται είναι υποχρεωτικά μοναδικό.
3. Η παραγωγή του συστήματος γίνεται αρχικά με τη δημιουργία ενός τυχαίου ονόματος, το οποίο μετατρέπεται σε έναν ακέραιο αριθμό για το seeding της υπόλοιπης διαδικασίας παραγωγής.
4. Έπειτα επιλέγεται ο αριθμός των αστεριών του συστήματος, με βάση μία κατανομή που βρίσκεται στα δεδομένα του generator, τα οποία φορτώνονται από ένα αρχείο στον αντίστοιχο κατάλογο /Generators. Επιλέγονται έπειτα οι κατηγορίες των αστεριών του συστήματος.
5. Στη συνέχεια επιλέγονται, ο αριθμός και το είδος των πλανητών που περιέχονται στο σύστημα, και η απόσταση του συστήματος από τη Γη (ένας αριθμός που αυξάνει ανά ηλιακό σύστημα, με βάση μία εκθετική κατανομή).
6. Τέλος, παράγονται διάφορες περιγραφές του συστήματος.

Ο PlanetGenerator είναι υπεύθυνος για την διαδικαστική παραγωγή των πλανητών ενός συστήματος. Για κάθε πλανήτη παράγεται ένα όνομα, καθώς και ο τύπος θερμοκρασίας του, ο αριθμός των δορυφόρων του, η κατηγορία μεγέθους του, η κάλυψή του με νερό, ο τύπος της ατμόσφαιράς του και η ύπαρξη ζωής στην επιφάνειά του (τώρα ή στο παρελθόν).

Σε περίπτωση που ο παίκτης επιλέξει να εισέλθει σε ένα ηλιακό σύστημα οι πλανήτες του καθορίζονται με μεγαλύτερη ακρίβεια. Πιο συγκεκριμένα, η κλάση PlanetDescriptionGenerator επιλέγει επιπλέον χαρακτηριστικά του πλανήτη, όπως είναι η τροχιά του, η περιστροφή γύρω από τον άξονά του, το κατά πόσο έχει δακτυλίους, καθώς και η περιγραφή της επιφάνειάς, της ατμόσφαιρας και των γεωλογικών του χαρακτηριστικών. Επιπρόσθετα, επιλέγονται τα είδη «απειλών» που χαρακτηρίζουν τον πλανήτη. Απειλή καλείται ένας λόγος για τον οποίο ο πλανήτης δεν είναι αποικίσιμος από τον παίκτη, και τα

είδη της απειλής που υπάρχουν σε έναν πλανήτη εξαρτώνται από τα χαρακτηριστικά του. Τέλος, παράγονται διάφορες περιγραφές του πλανήτη (η απλή περιγραφή που τυπώνεται όταν ο παίκτης κάνει Examine τον πλανήτη, όταν τον παρατηρεί μέσω τηλεσκοπίου ή όταν στέλνει drones για την παρατήρησή του).

4.10.2 Διαδικαστική παραγωγή περιγραφών

Οι περιγραφές οι οποίες παράγονται διαδικαστικά δημιουργούνται κυρίως με την χρήση templates. Τα πρότυπα και οι διαφορετικές επιλογές που μπορούν να συμπληρωθούν σε αυτά είναι γραμμένα χειροκίνητα, αλλά συνδυάζονται διαδικαστικά για την παραγωγή μίας πολύ μεγάλης ποικιλίας πιθανών περιγραφών. Για την υλοποίηση των templates, χρησιμοποιήθηκαν εκτενώς εργαλεία της Python για μορφοποίηση string, όπως είναι η μέθοδος format, ή ο τύπος f-strings (string literals που διαμορφώνονται δυναμικά κατά την εκτέλεση του προγράμματος).

4.10.3 Διαδικαστική παραγωγή ονομάτων

Η παραγωγή των ονομάτων του παιχνιδιού γίνεται μέσω της κλάσης NameGenerator, με τη χρήση κανονικών τυπικών γραμματικών ($3^{ου}$ τύπου). Κάθε όνομα πλανήτη είναι μοναδικό για το ηλιακό σύστημα στο οποίο βρίσκεται, και καθορίζεται από έναν seed ο οποίος εξαρτάται από το όνομα του ηλιακού συστήματος που βρίσκεται ο πλανήτης, καθώς και τη σειρά του πλανήτη στη λίστα πλανητών του συστήματος. Ενδεικτικά, παρουσιάζεται η γραμματική για τη δημιουργία ονομάτων πλανητών στον Πίν. 4.3.

Πίν. 4.3: Κανόνες επανεγγραφής γραμματικής ονομάτων πλανητών.

#	Κεφαλή	Σώμα
1	S	→ B B | B B B
2	B	→ C V L | V C | C V | V L | L V
3	C	→ b | c | d | g | h | gh | k | q | p | sh | t | v | w | x | z | zh | kh | qq
4	V	→ y | i | a | o | u | e
5	L	→ r | n | s | m | l | x

4.10.4 Διαδικαστική παραγωγή δωματίων και αντικειμένων

Σε περίπτωση που ο παίκτης επιλέξει να προσγειωθεί σε έναν πλανήτη και να τον εξερευνήσει, πρέπει να παραχθούν διαδικαστικά τα Rooms στα οποία μπορεί να περιηγηθεί. Η κλάση PlanetRoomGenerator διαχειρίζεται την παραγωγή αυτή. Πιο συγκεκριμένα, κάθε πλανήτης έχει συνολικά 100 δωμάτια (τα οποία βρίσκονται σε έναν νοητό τετράγωνο χάρτη 10 επί 10, οι άκρες του οποίου συνδέονται μεταξύ τους, οπότε ένας παίκτης μπορεί να κάνει τον γύρο ενός πλανήτη με 10 κινήσεις). Ανάλογα με το είδος του πλανήτη υπάρχουν διαφορετικές κατηγορίες δωματίων, κάθε μία από τις οποίες έχει το δικό της template για την περιγραφή του χώρου.

Κατά την περιήγηση του παίκτη στον πλανήτη, για την μείωση των απαιτήσεων του συστήματος σε μνήμη, δεν δημιουργούνται ταυτόχρονα και τα 100 δωμάτια του χάρτη, αλλά μόνο το τωρινό δωμάτιο του παίκτη και τα 8 γειτονικά του. Κάθε δωμάτιο, εκτός από την περιγραφή του, μπορεί να περιέχει αντικείμενα-στοιχεία, σχετικά με το είδος «απειλής» του πλανήτη. Όλα τα δωμάτια είναι επαναλήψιμα, και έχει δοθεί ιδιαίτερη προσοχή στην αρχικοποίηση της γεννήτριας ψευδοτυχαίων αριθμών, με τρόπο τέτοιον ώστε να εξασφαλίζεται ότι γειτονικά δωμάτια δε θα έχουν κατά κανόνα ίδιες περιγραφές.

4.11 Άλλοι μηχανισμοί

Πέρα από τους μηχανισμούς που αναφέρθηκαν στις παραπάνω ενότητες, υλοποιήθηκαν για τις ανάγκες της εργασίας και άλλοι συνοδευτικοί μηχανισμοί, μερικοί από τους οποίους αναλύονται παρακάτω.

4.11.1 Διαχείριση σφαλμάτων

Μία από τις σημαντικότερες πτυχές της υλοποίησης ενός παιχνιδιού διαδραστικής φαντασίας με τη χρήση Parser είναι η διαχείριση των λανθασμένων εντολών που δίνονται από τον χρήστη. Λόγω της φύσης της εισόδου (κείμενο φυσικής γλώσσας), αλλά και την άγνοια του παίκτη για τη σωστή χρήση όλων των εντολών του παιχνιδιού, είναι απαραίτητο ο σχεδιαστής ενός IF Game να προβλέψει όλα τα πιθανά σφάλματα που μπορούν να προκύψουν και να τα διαχειριστεί κατάλληλα στην εφαρμογή. Ακόμα, για να βελτιωθεί η εμπειρία του

παίκτη κατά την αλληλοεπίδρασή του με το παιχνίδι, είναι σημαντικό η διαχείριση σφαλμάτων να συνοδεύεται από την εκτύπωση των κατάλληλων μηνυμάτων λάθους, έτσι ώστε ο χρήστης να κατανοήσει τους λόγους για τους οποίους η εντολή του απέτυχε.

Για τους παραπάνω λόγους, κατά την ανάπτυξη του παιχνιδιού δημιουργήθηκαν πολλές κλάσεις λαθών (αρχείο errors.py) για την διαχείριση σφαλμάτων και την εκτύπωση μηνυμάτων λάθους. Οι κλάσεις αυτές είναι:

- Η κλάση Error: Κληρονομεί από την κλάση Exception της Python. Κάθε instance της κλάσης έχει μία συμβολοσειρά που την χαρακτηρίζει (error_type). Ανάλογα με το είδος του σφάλματος, τυπώνεται και το αντίστοιχο μήνυμα λάθους.
- ParserError: Για σφάλματα κατά τη συντακτική ανάλυση (λάθος συντακτική μορφή της πρότασης ή άγνωστο για τον Parser ρήμα)
- NPParserError: Για σφάλματα κατά την ανάλυση ονοματικών φράσεων (όπως άγνωστα ουσιαστικά ή επίθετα).
- CheckCommandError: Για σφάλματα κατά τον νοηματικό έλεγχο μίας εντολής.
- PreconditionsError: Σφάλματα προ υπαρχουσών συνθηκών.
- ActionError: Για σφάλματα κατά την εκτέλεση μίας εντολής.
- DialogError: Για σφάλματα εντολών διαλόγου.
- UndoCommand: Σφάλμα που προκύπτει όταν ο παίκτης δώσει μία εντολή αναίρεσης.

4.11.2 Χρονισμός γεγονότων

Πολλά από τα γεγονότα που συμβαίνουν στο παιχνίδι χρειάζεται να είναι χρονισμένα. Για το σκοπό αυτό δημιουργήθηκε η κλάση timer.CustomTimer, η οποία καταγράφει τον συνολικό χρόνο του παιχνιδιού, καθώς και τον χρόνο του τωρινού κεφαλαίου. Ο timer εκτελείται σε ξεχωριστό νήμα από το υπόλοιπο παιχνίδι.

4.11.3 Εργαλεία συγγραφής και αποσφαλμάτωσης

Ένα από τα μεγαλύτερα μειονεκτήματα της δημιουργίας ενός παιχνιδιού IF χωρίς την χρήση κάποιου περιβάλλοντος ανάπτυξης ή γλώσσας προγραμματισμού ειδικού σκοπού είναι η έλλειψη των κατάλληλων εργαλείων συγγραφής και αποσφαλμάτωσης (authoring and

debugging tools). Μηχανές ανάπτυξης παιχνιδιών διαδικαστικής αφήγησης, όπως το Inform 7, διαθέτουν αρκετά εργαλεία για τη γρηγορότερη ανίχνευση λαθών, και για την ευκολότερη συγγραφή των αντικειμένων και των κανόνων ενός παιχνιδιού (συνήθως μέσω μίας γλώσσας ειδικού σκοπού).

Επομένως, για την ευκολότερη ανάπτυξη του παρόντος παιχνιδιού, αναπτύχθηκαν ορισμένες πρώιμες εκδόσεις τέτοιων εργαλείων. Το σημαντικότερο από αυτά είναι η λειτουργία "Replay", η οποία επιτρέπει στον χρήστη της εφαρμογής να ξαναπαίξει αυτόματα όλες τις εντολές που έδωσε την ακριβώς προηγούμενη φορά που χρησιμοποίησε το παιχνίδι. Η χρησιμότητα του εργαλείου αυτού έγκειται στην ευκολότερη αποσφαλμάτωση της εφαρμογής, καθώς, σε περίπτωση που δημιουργηθεί κάποιο σφάλμα κατά τη διάρκεια ενός παιχνιδιού, ο προγραμματιστής έχει τη δυνατότητα να ακολουθήσει ξανά τα ίδια ακριβώς βήματα που οδήγησαν στην εμφάνιση του λάθους αυτόματα. Για την λειτουργία του εργαλείου αυτού, είναι απαραίτητες οι συναρτήσεις που ορίζονται στο αρχείο log_commands.py, που χρησιμοποιούνται για την καταγραφή των εντολών του παίκτη, της χρονικής στιγμής στην οποία δόθηκαν, καθώς και για την καταγραφή του seed του παιχνιδιού.

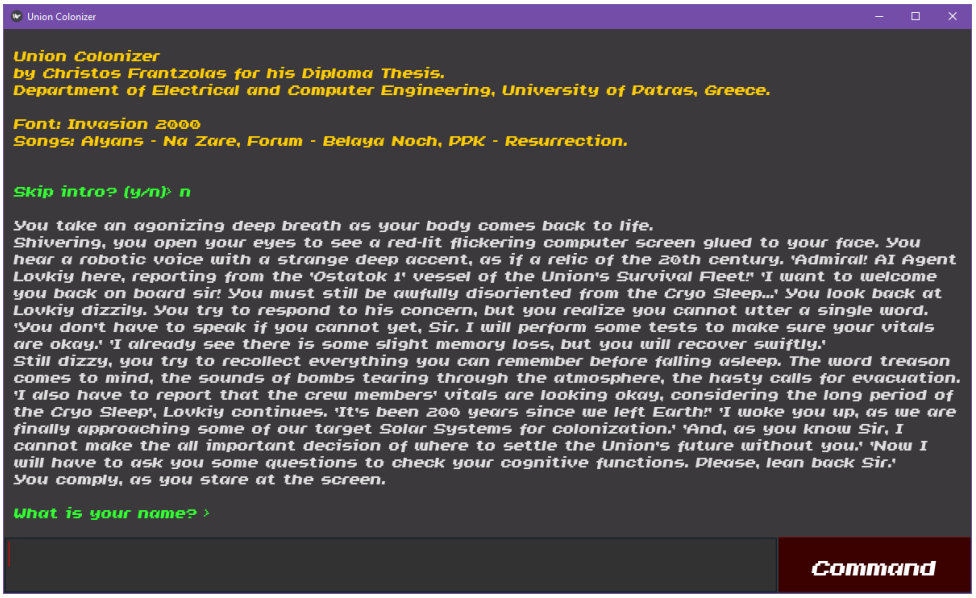
4.12 Γραφική διεπαφή

Η διεπαφή του χρήστη με την εφαρμογή πραγματοποιείται μέσω των κλάσεων app.MyApp, display.Display και game_screen.GamePage.

Η κλάση Display είναι υπεύθυνη για την επικοινωνία μεταξύ του νήματος του παιχνιδιού και του νήματος της γραφικής διεπαφής χρήστη (GUI). Η κλάση διαθέτει μεθόδους, τόσο για τη λήψη μίας εντολής από το παράθυρο της εφαρμογής, όσο και για την προβολή του αποτελέσματος της εντολής στην οθόνη. Ανάλογα με το είδος του αποτελέσματος (μήνυμα σφάλματος, διάλογος, αποτέλεσμα πράξης), η μορφή του κειμένου που εμφανίζεται στην οθόνη είναι διαφορετική.

Η κλάση MyApp κληρονομεί από την κλάση App της βιβλιοθήκης kivy (βλ. 4.1.6) και μαζί με την κλάση GamePage δημιουργούν το User Interface της εφαρμογής. Το UI αποτελείται από δύο οθόνες, μία αρχική οθόνη για την εκκίνηση του παιχνιδιού, και την κύρια οθόνη του παιχνιδιού. Η κύρια οθόνη περιέχει ένα πλαίσιο εισαγωγής κειμένου για την

εισαγωγή εντολών από τον παίκτη, καθώς και ένα ιστορικό με όλες τις εντολές και τα αποτελέσματά τους μέχρι στιγμής στο παιχνίδι. Ακόμα, κατά την διάρκεια του παιχνιδιού γίνεται ταυτόχρονη αναπαραγωγή μουσικής με τη βοήθεια της kivy. Ένα στιγμιότυπο της οθόνης φαίνεται στην Εικ. 3.



Εικ. 3 Ένα στιγμιότυπο της κύριας οθόνης της εφαρμογής.

5. ΑΠΟΤΕΛΕΣΜΑΤΑ ΥΛΟΠΟΙΗΣΗΣ

5.1 Η υπόθεση του παιχνιδιού

5.1.1 Η γενικότερη υπόθεση και ο στόχος του παίκτη

Το παιχνίδι που δημιουργήθηκε στα πλαίσια της παρούσας εργασίας ονομάζεται Union Colonizer, και πρόκειται για ένα παιχνίδι διαδικαστικής αφήγησης με θέμα την εξερεύνηση του διαστήματος. Στόχος του παίκτη είναι να εξερευνήσει τα ηλιακά συστήματα που συναντά, τα οποία δημιουργούνται διαδικαστικά από την εφαρμογή. Κάθε ηλιακό σύστημα περιέχει πλανήτες, τους οποίους ο παίκτης επισκέπτεται και εξερευνά μέχρις ότου να βρει κάποιον πλανήτη ο οποίος διαθέτει τις κατάλληλες συνθήκες για την υποστήριξη μίας ανθρώπινης αποικίας. Παράλληλα με αυτήν την διαδικασία εξερεύνησης του διαστήματος, ο παίκτης μπορεί να περιηγηθεί στα δωμάτια που απαρτίζουν το διαστημόπλοιό του και να συζητήσει με τον Lovkiy (το σύστημα τεχνητής νοημοσύνης του σκάφους). Οι διάλογοι παρέχουν στον παίκτη πληροφορίες τόσο για την εξερεύνηση των πλανητών, όσο και για το παρελθόν του Player Character. Ο παίκτης κερδίζει σε περίπτωση που καταφέρει να χτίσει μία αποικία σε έναν κατάλληλο πλανήτη. Σε περίπτωση που γίνει προσπάθεια προσγείωσης σε έναν ακατάλληλο πλανήτη, κατασκευή αποικίας σε έναν πλανήτη ο οποίος δεν είναι αποικίσιμος, ή ο παίκτης πέσει θύμα κάποιας παγίδας, τότε χάνει το παιχνίδι.

5.1.2 Αλληλουχία κεφαλαίων

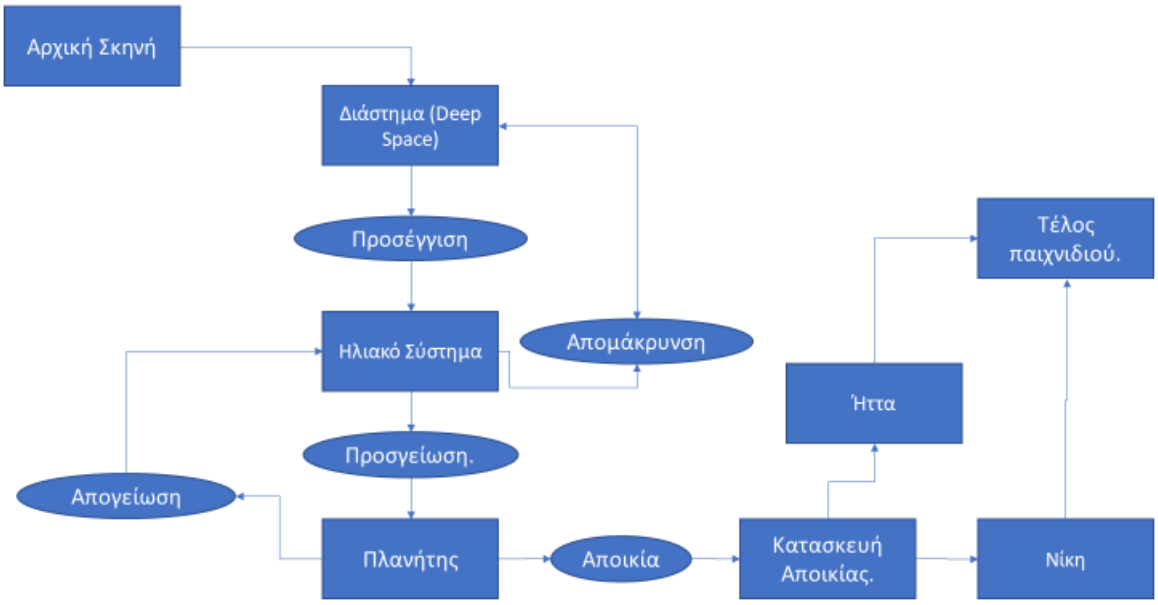
Στην Εικ. 4 φαίνεται η διαδοχή των κεφαλαίων του παιχνιδιού. Το εισαγωγικό κεφάλαιο (Intro Chapter) του παιχνιδιού έχει τους εξής στόχους:

- Να εισάγει τον παίκτη στην υπόθεση του παιχνιδιού.
- Να εισάγει τον παίκτη στους βασικούς μηχανισμούς του Interactive Fiction, και στις βασικές εντολές που μπορεί να δώσει για να πλοηγηθεί στο παιχνίδι.
- Να καθορίσει στον παίκτη τον στόχο του παιχνιδιού.
- Να επιτρέψει στον παίκτη να εισάγει κάποιες απαντήσεις που θα καθορίσουν τον σπόρο (seed) του παιχνιδιού.

Το επόμενο κεφάλαιο (Deep Space Chapter) περιέχει το διαστημόπλοιο (τον χάρτη, τα αντικείμενα και τα θέματά του) τα οποία παραμένουν σταθερά σε κάθε παιχνίδι. Σε τακτικές χρονικές περιόδους, ένα νέο Ηλιακό Σύστημα εμφανίζεται στη σκηνή. Σε περίπτωση που ο παίκτης επιλέξει να εισέλθει στο ηλιακό σύστημα (Enter), το παιχνίδι μεταβαίνει σε κεφάλαιο Solar System. Σε περίπτωση που ο παίκτης δεν επιλέξει να εισέλθει, το Κεφάλαιο παραμένει το Deep Space Chapter, και όλα τα σχετικά με το προηγούμενο ηλιακό σύστημα αντικείμενα διαγράφονται από τη μνήμη, ενώ αρχίζει να μετράει και πάλι ο χρόνος για την επόμενη προσέγγιση Ηλιακού Συστήματος.

Το κεφάλαιο Planet Chapter ενεργοποιείται μετά την εντολή land on του παίκτη σε κάποιον πλανήτη. Το κεφάλαιο μπορεί να τελειώσει με τους εξής τρόπους:

- Προσγείωση σε ακατάλληλο πλανήτη, η οποία οδηγεί στον θάνατο του παίκτη.
- Προσγείωση στον πλανήτη και κατασκευή αποικίας.
- Προσγείωση στον πλανήτη και takeoff → Επιστροφή στη σκηνή του ηλιακού συστήματος.
- Προσγείωση στον πλανήτη και θάνατος, μετά από συνάντηση του παίκτη με κάποια απειλή. Οι απειλές αποτελούνται από ένα σύνολο αντικειμένων και γεγονότων που μπορεί να οδηγήσουν στην ήττα του παίκτη, και καθιστούν αδύνατη τη δημιουργία της αποικίας στον πλανήτη. Κάθε απειλή έχει κάποια προειδοποιητικά σημάδια (στην περιγραφή του πλανήτη, αντικείμενα διάσπαρτα στον πλανήτη κλπ).



Εικ. 4 Η αλληλουχία των κεφαλαίων του παιχνιδιού,

Το κεφάλαιο Colony Chapter ενεργοποιείται όταν ο χρήστης κατασκευάσει μια αποικία ενώ βρίσκεται σε έναν πλανήτη. Εάν ο πλανήτης αυτός είναι αποικίσιμος, ο παίκτης κερδίζει. Εάν όχι, ο παίκτης χάνει. Σε αυτή τη σκηνή δημιουργείται διαδικαστικά η περιγραφή της αποικίας και της κατάληξής της, με βάση τα χαρακτηριστικά του πλανήτη στον οποίο χτίστηκε. Το κεφάλαιο Νίκης ενεργοποιείται σε περίπτωση που ο παίκτης χτίσει μια αποικία σε κάποιον κατάλληλο πλανήτη.

Το κεφάλαιο Θανάτου του παίκτη ενεργοποιείται στις εξής περιπτώσεις:

- Εάν ο παίκτης προσπαθήσει να προσγειωθεί σε κάποια ακατάλληλα είδη πλανητών.
- Εάν ο παίκτης χάσει λόγω κάποιας απειλής σε έναν επικίνδυνο πλανήτη.
- Εάν ο παίκτης χτίσει μια αποικία σε έναν πλανήτη που δεν είναι αποικίσιμος.

5.1.3 Διάλογοι

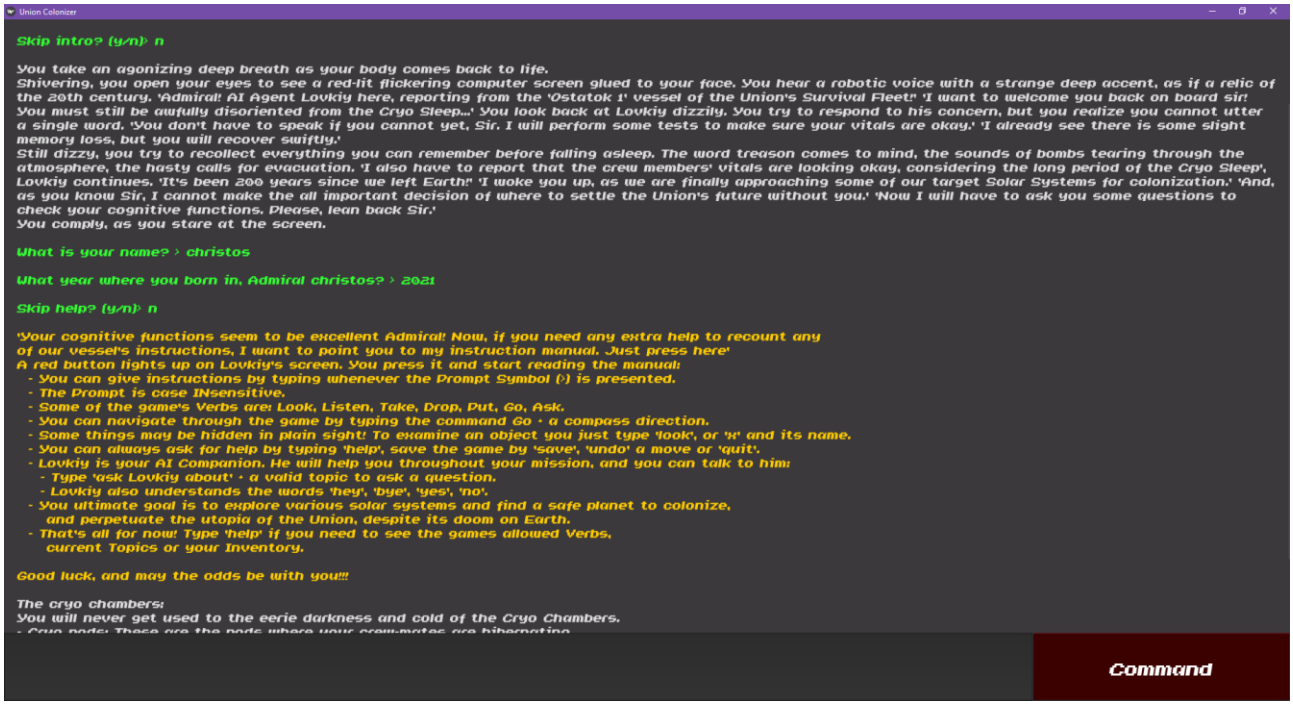
Οι διάλογοι στο παιχνίδι γίνονται εξ ολοκλήρου με έναν Non-Player Character, τον υπολογιστή του σκάφους. Ο χαρακτήρας αυτός έχει ως σκοπό να δώσει περισσότερες πληροφορίες στον παίκτη για το σκάφος στο οποίο βρίσκεται και για τους πλανήτες στους

οποίους προσγειώνεται. Επιπλέον, οι διάλογοι χρησιμοποιούνται ως ένα εργαλείο για να αποκαλυφθούν κάποια στοιχεία της ιστορίας του Player Character. Ο παίκτης μπορεί να κάνει κάποιες ερωτήσεις στον υπολογιστή που τον οδηγούν σε διαφορετικές απαντήσεις κάθε φορά, καθώς και στην ανακάλυψη νέων θεμάτων προς συζήτηση. Οι απαντήσεις στις ερωτήσεις του παίκτη αλλάζουν με βάση αρκετούς παράγοντες, όπως:

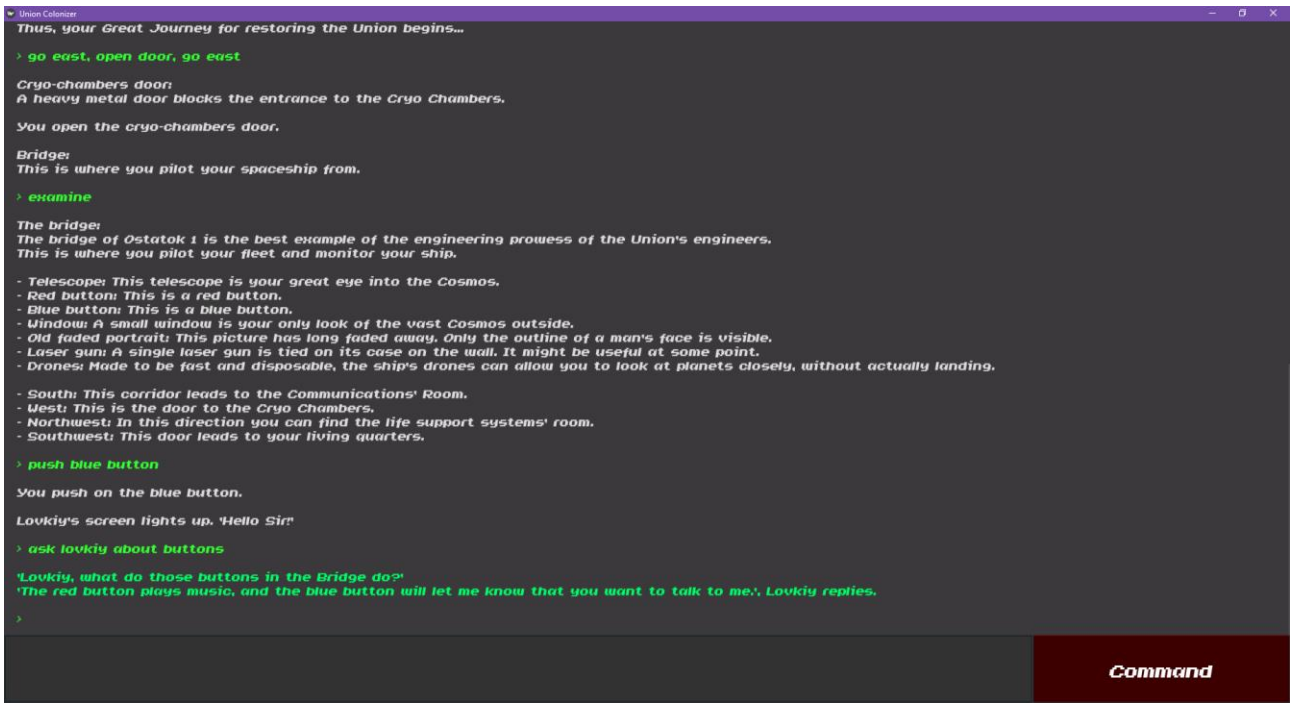
- Το πόσες φορές έχει γίνει ερώτηση από τον παίκτη για το ίδιο θέμα.
- Το ποια άλλα θέματα έχουν αναφερθεί προηγουμένως στον διάλογο.
- Την ανακάλυψη ορισμένων αντικειμένων από τον παίκτη.
- Τον πλανήτη στον οποίο βρίσκεται ο παίκτης, και τα χαρακτηριστικά του πλανήτη αυτού.
- Το κεφάλαιο που είναι ενεργό κάθε δεδομένη χρονική στιγμή.

5.2 Στιγμιότυπα του παιχνιδιού

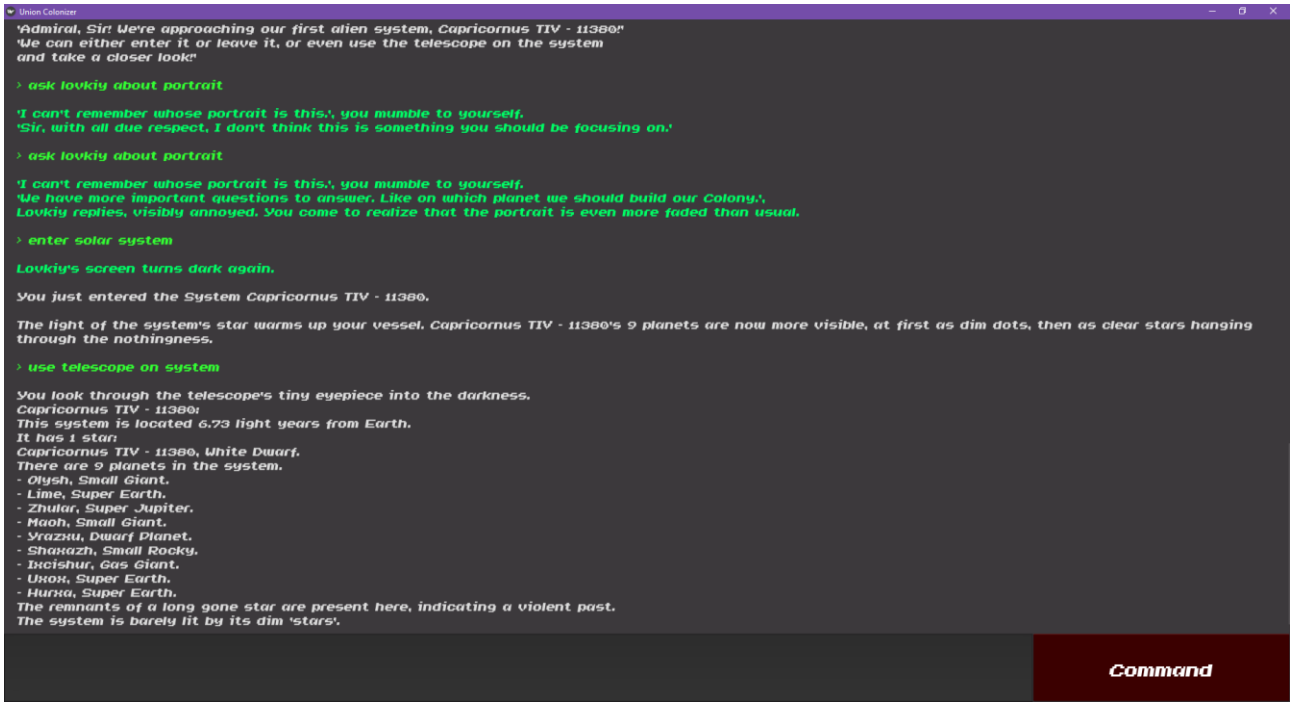
Παρακάτω παρουσιάζονται ορισμένα στιγμιότυπα του παιχνιδιού (για περισσότερα στιγμιότυπα βλ. Παράρτημα Β).



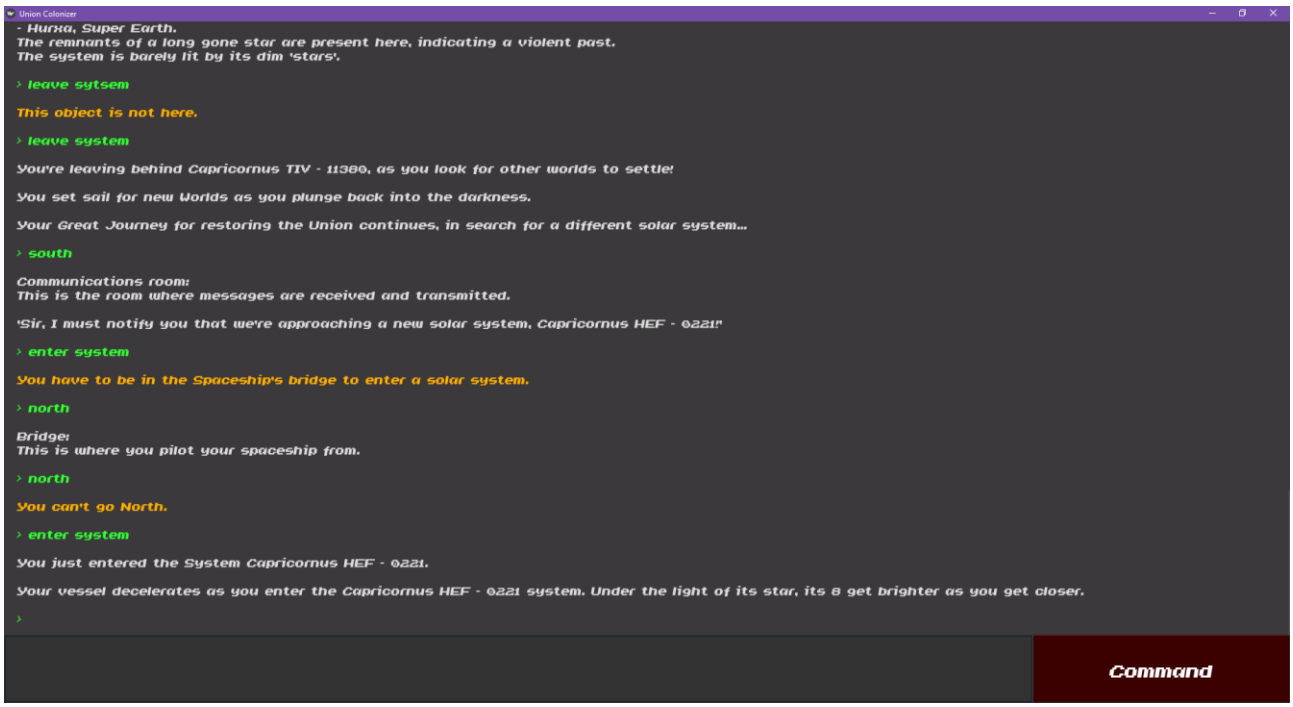
Εικ. 5 Η εισαγωγική περιγραφή του παιχνιδιού και η βοήθεια προς τον παίκτη.



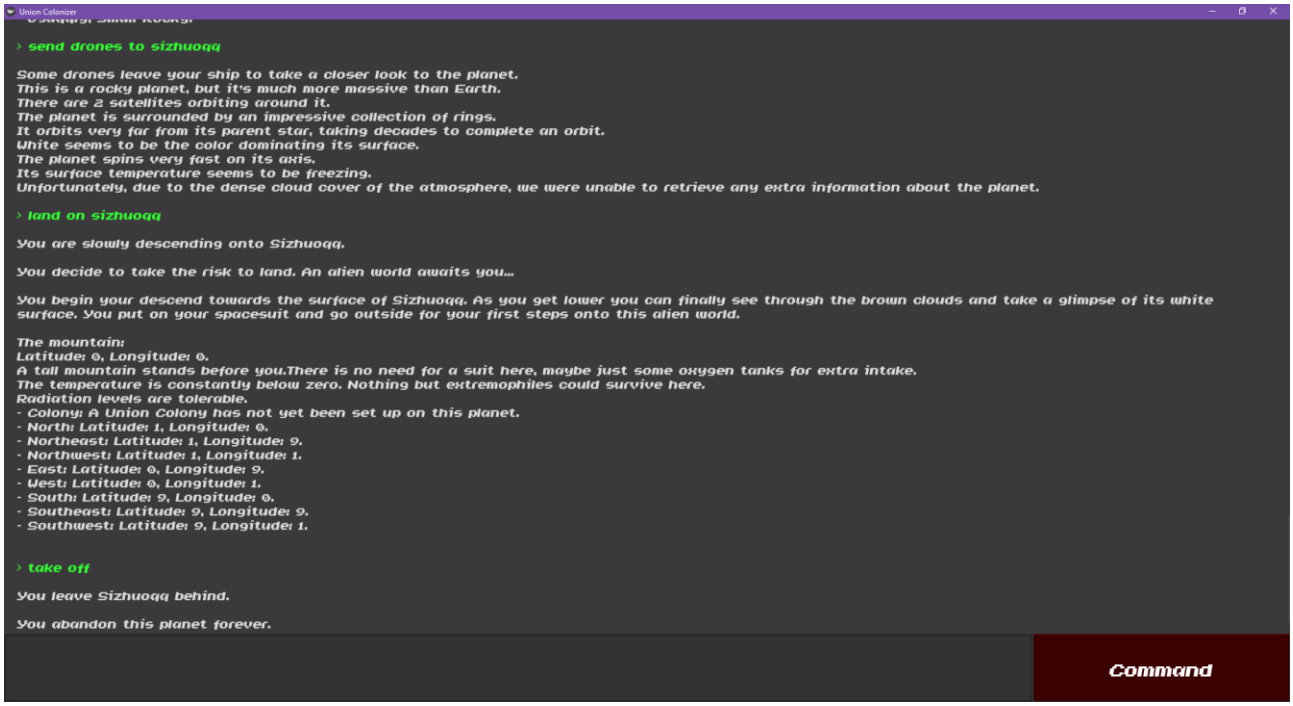
Εικ. 6 Η περιγραφή της "γέφυρας" του διαστημοπλοίου και ένα μικρό κομμάτι διαλόγου.



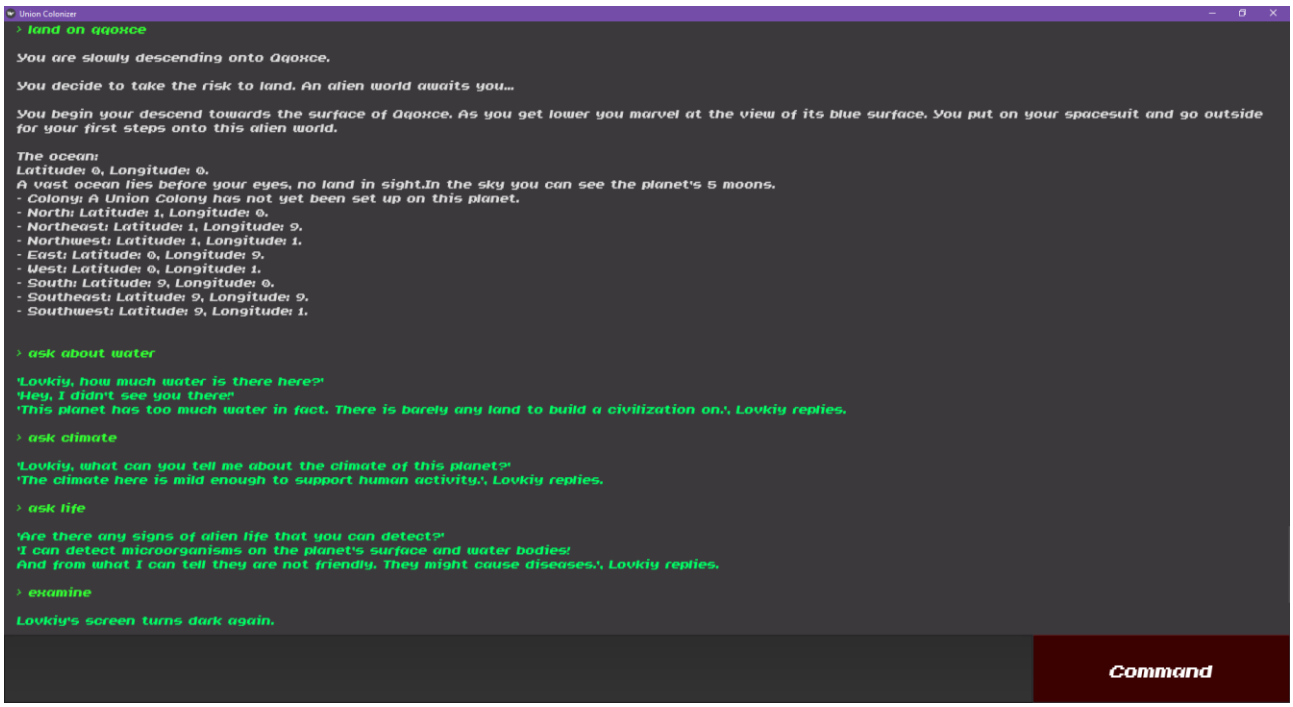
Εικ. 7 Εμφάνιση ενός ηλιακού συστήματος και περιγραφή του μέσω του τηλεσκοπίου (διαδικαστικά παραγόμενη).



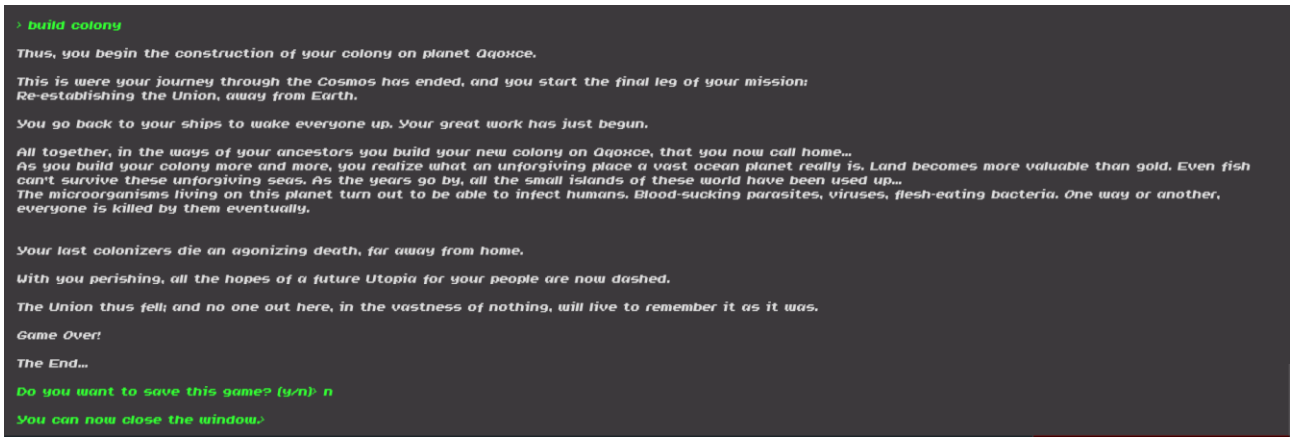
Εικ. 8 Ορισμένα μηνύματα λάθους.



Εικ. 9 Αποστολή drones, προσγείωση και απογείωση σε πλανήτη.



Εικ. 10 Προσγείωση σε πλανήτη και διάλογος για τις συνθήκες στην επιφάνειά του.



Εικ. 11 Κατασκευή αποικίας σε ακατάλληλο πλανήτη και θάνατος του παίκτη.

5.3 Δοκιμές Beta και παρατηρήσεις

Κατά την ανάπτυξη του παιχνιδιού έγιναν δοκιμές Beta με τρία διαφορετικά άτομα, για την λήψη σχολίων και ανατροφοδότησης για το παιχνίδι, και την εύρεση πιθανών σφαλμάτων που δεν είχαν παρατηρηθεί μέχρι εκείνη την στιγμή. Μετά από κάθε δοκιμή έγινε διόρθωση ορισμένων εκ των παρατηρήσεων, ενώ κάποιες άλλες καταγράφηκαν ως ενδιαφέρουσες προτάσεις και βελτιώσεις για κάποια μελλοντική εκδοχή της εφαρμογής. Κάποια από τα σχόλια που αναφέρθηκαν ήταν τα εξής:

- Να μπορεί ο παίκτης να κάνει ερωτήσεις για την τοποθεσία των δωματίων ενός χάρτη, ώστε να μπορεί να προσανατολιστεί ευκολότερα.
- Η τοποθέτηση ορισμένων αντικειμένων (όπως το τηλεσκόπιο) στην γέφυρα του σκάφους ώστε ο παίκτης να μη χρειάζεται να αλλάζει θέση άσκοπα διαρκώς.
- Να προστεθούν οι συντεταγμένες των δωματίων ενός πλανήτη στην περιγραφή του, ώστε ο παίκτης να μπορεί να προσανατολιστεί ευκολότερα σε αυτά.
- Να υπάρχει κάποιου είδους συντομογραφία για τις ερωτήσεις προς τον υπολογιστή (π.χ. αντί ο παίκτης να χρειάζεται να γράφει "ask Lovkiy about <Topic>" να μπορεί απλά να γράψει "ask (about) <Topic>").
- Να υπάρχει κάποιο είδος βοήθειας προς τον παίκτη για το ποια ρήματα μπορούν να χρησιμοποιηθούν με ποια αντικείμενα.
- Η εντολή examine να μπορεί να δεχτεί ως αντικείμενο και το Room object στο οποίο βρίσκεται ο παίκτης.
- Ο παίκτης να μπορεί να αναφερθεί στον χαρακτήρα του μέσω του ονόματος που δίνει στο εισαγωγικό κεφάλαιο.
- Μετά την εντολή για το χτίσιμο μία αποικίας, ο παίκτης να συνδιαλέγεται με τον υπολογιστή για το κατά πόσο ο πλανήτης είναι κατάλληλος, προτού ληφθεί η τελική απόφαση.
- Να μπορούν να χρησιμοποιηθούν κάποια επιπλέον ρήματα διαλόγου, όπως "thank you" και "please".
- Η αναπαραγωγή της μουσικής του παιχνιδιού να σταματάει όταν αυτό ολοκληρωθεί, και να υπάρχει κάποιος ήχος για την περίπτωση του θανάτου ή της νίκης του παίκτη.
- Να δίνεται πληρέστερη αιτιολόγηση από την εφαρμογή για τον λόγο για τον οποίο μία πράξη δεν μπορεί να πραγματοποιηθεί.
- Σε περίπτωση που ο παίκτης προσπαθήσει να μετακινηθεί προς μία κατεύθυνση που δεν υπάρχει στο δωμάτιο, να εμφανίζονται όλες οι επιτρεπτές κατευθύνσεις μαζί με το μήνυμα λάθους.

Επίσης, αποκαλύφθηκαν και διορθώθηκαν ορισμένα σφάλματα, όπως ένα λάθος στην φόρτωση και δημιουργία των δωματίων των πλανητών, ορισμένα ορθογραφικά ή γραμματικά λάθη στις περιγραφές, καθώς και κάποια λάθη του.

5.4 Προκλήσεις και λάθη

Πολλές από τις προκλήσεις που έπρεπε να αντιμετωπιστούν κατά την ανάπτυξη του παιχνιδιού σχετίζονται με τους περιορισμούς που επιβάλλει η διαδραστική φαντασία ως μέσο. Αρχικά, η συγγραφή υλικού που προορίζεται για διαδικαστική αφήγηση είναι μία αρκετά χρονοβόρα διαδικασία. Πολλές φορές η προσπάθεια που απαιτείται είναι μεγαλύτερη από αυτή που καταβάλλεται για την συγγραφή ενός αντίστοιχου μη-διαδικαστικού κειμένου. Ακόμα, οι νέοι παίκτες παιχνιδιών IF χρειάζονται αρκετή εξοικείωση με τις συμβάσεις του είδους, ενώ ο μεγάλος περιορισμός που επιβάλλεται τελικά στα είδη εντολών που μπορεί να δώσει ο παίκτης καθιστούν τέτοιου είδους παιχνίδια λιγότερο ελκυστικά στο ευρύ κοινό. Τέλος, η διαδικαστική παραγωγή μίας μεγάλης αλλά και ουσιαστικής ποικιλίας αντικειμένων αποτελεί μία μεγάλη πρόκληση για όσους αναπτύσσουν τέτοιου είδους εφαρμογές.

Συγκεκριμένα για την παρούσα εφαρμογή, μία από τις παραλείψεις που σημειώθηκαν ήταν ο μικρός αριθμός των αντικειμένων σε κάθε δωμάτιο (ειδικά κατά την εξερεύνηση των πλανητών) και ο σχετικά περιορισμένος αριθμός πράξεων που μπορούν να συντελεστούν σε αυτά τα αντικείμενα. Οι περιγραφές των αντικειμένων θα μπορούσαν να είναι πιο αναλυτικές, όπως και οι περιγραφές των σφαλμάτων, που πρέπει να εξηγούν με περισσότερη λεπτομέρεια τους λόγους για τους οποίους απέτυχε μία πράξη και το ποιες εναλλακτικές εντολές μπορεί να δώσει ο παίκτης. Επίσης, οι δυναμικοί διάλογοι θα μπορούσαν να είναι πιο περίπλοκοι και -πέρα από το να παρέχουν πληροφορίες - να επηρεάζουν την εξέλιξη της πλοκής με κάποιον τρόπο.

5.5 Μελλοντικές βελτιώσεις και επεκτάσεις

Κάποια στοιχεία της εφαρμογής τα οποία θα μπορούσαν να βελτιωθούν μελλοντικά είναι:

- Οι περιγραφές σφαλμάτων να γίνουν πιο εξειδικευμένες, ανάλογα με το είδος του σφάλματος το οποίο προέκυψε, και να οδηγούν τον παίκτη προς τη σωστή κατεύθυνση για την αποφυγή του λάθους.
- Οι περιγραφές των αντικειμένων και των δωματίων του παιχνιδιού να είναι πιο αναλυτικές, παραστατικές και ποικίλες.
- Να προστεθούν περισσότερα αντικείμενα, ειδικά στο κεφάλαιο εξερεύνησης των πλανητών, ώστε να δοθούν στον παίκτη πιο ποικίλοι τρόποι να αλληλοεπιδράσει με τον κόσμο του παιχνιδιού.
- Οι διάλογοι να γίνουν πιο περίπλοκοι, με την προσθήκη συναισθηματικών μεταβάσεων για τους NPCs (κάτι το οποίο μπορεί να επιτευχθεί με τους ήδη υπάρχοντες κόμβους του δέντρου διαλόγων), με πιο ποικίλες απαντήσεις σε ερωτήσεις του χρήστη και πιο περίπλοκα δέντρα διαλόγων. Ακόμα θα μπορούσαν να προστεθούν και άλλα ρήματα διαλόγου, καθώς και περισσότερα θέματα προς συζήτηση.
- Η βοήθεια που δίνεται στον παίκτη να γίνει πιο αναλυτική, καλύπτοντας τομείς όπως είναι τα ρήματα με τα οποία μπορεί να χρησιμοποιηθεί κάθε αντικείμενο, το πώς ακριβώς εκτελούνται οι βασικές πράξεις του παιχνιδιού (μετακίνηση, παρατήρηση, διάλογος) κ.α.
- Να εμπλουτιστεί το σύστημα διαδικαστικής παραγωγής ηλιακών συστημάτων και πλανητών ώστε να μπορεί να παραχθεί μία μεγαλύτερη ποικιλία σωμάτων και δωματίων προς εξερεύνηση.
- Η υπόθεση του παιχνιδιού θα μπορούσε να διακλαδώνεται περισσότερο, με παράλληλες αφηγήσεις.
- Να δημιουργηθεί κάποιο μοντέλο αναπαράστασης της γνώσης του Player Character για τον κόσμο του παιχνιδιού, όπως για παράδειγμα κάποιος γράφος γνώσεων, όπως αναφέρθηκε στο 3.6.2.

- Να υλοποιηθεί κάποια κλάση για συμβολοσειρές που αλλάζουν δυναμικά με βάση τις συνθήκες του κόσμου, ώστε να γίνει πιο εύκολη η δημιουργία και αξιοποίηση templates κατά την συγγραφή του παιχνιδιού.
- Να βελτιωθεί η απόδοση της εφαρμογής ως προς την ταχύτητα απόκρισης, την ποσότητα μνήμης που δεσμεύεται, καθώς και τη διαδικασία αποθήκευσης και ανάγνωσης αρχείων.
- Για τη διευκόλυνση της δημιουργίας νέων αντικειμένων, θεμάτων, διαλόγων, δωματίων κλπ. είναι απαραίτητη η δημιουργία ενός συγγραφικού εργαλείου, ώστε να μην απαιτείται η απευθείας συγγραφή των αντικειμένων σε αρχεία JSON.

6. ΕΠΙΛΟΓΟΣ

Στόχος της παρούσας διπλωματικής εργασίας ήταν η ανάπτυξη ενός παιχνιδιού διαδικαστικής αφήγησης με δυναμικούς διαλόγους. Η υλοποίηση της εφαρμογής έγινε σε γλώσσα ειδικού σκοπού (Python 3), και ως εκ τούτου απαιτήθηκε η εκ του μηδενός δημιουργία ορισμένων εργαλείων τα οποία αποτελούν στο σύνολό τους μία μηχανή ανάπτυξης παιχνιδιών διαδικαστικής αφήγησης. Τα στάδια για την ανάπτυξη του παιχνιδιού είχαν ως εξής:

1. Δημιουργία των κλάσεων των οντοτήτων του παιχνιδιού.
2. Υλοποίηση ενός συστήματος συντακτικής ανάλυσης των εντολών που λαμβάνονται από τον χρήστη.
3. Η ανάπτυξη μεθόδων ελέγχου και εφαρμογής των πράξεων του παίκτη στον κόσμο του παιχνιδιού.
4. Δημιουργία ενός συστήματος δυναμικών διαλόγων.
5. Υλοποίηση των κεφαλαίων και των γεγονότων του παιχνιδιού.
6. Συγγραφή του παιχνιδιού και όλων των αντικειμένων του.
7. Δημιουργία ενός συστήματος γραφικής διεπαφής.

Η διαδικασία της ανάπτυξης του παρόντος παιχνιδιού οδήγησε στην καλύτερη κατανόηση των μηχανισμών που διέπουν τη διαδικαστική αφήγηση. Επιπλέον, υπήρξε η ευκαιρία για την ανάπτυξη ενός διαλογικού συστήματος το οποίο ανταποκρίνεται διαφορετικά στην ίδια ερώτηση αξιοποιώντας μία βάση γνώσης (εδώ ο κόσμος του παιχνιδιού). Για την συγγραφή του κώδικα χρησιμοποιήθηκαν χρήσιμα εργαλεία, όπως για παράδειγμα το Pycharm, αλλά

και βιβλιοθήκες της Python των οποίων η χρησιμότητα βρίσκεται και πέρα από την υλοποίηση της παρούσας εργασίας.

Τα συστήματα που αναπτύχθηκαν σε αυτή τη διπλωματική εργασία είναι επαναχρησιμοποιήσιμα και για την υλοποίηση άλλων παιχνιδιών IF, πέρα από αυτό το οποίο δημιουργήθηκε. Η επεκτασιμότητα και η ευελιξία του κώδικα του συστήματος ήταν μία από τις βασικές προτεραιότητες και, ως αποτέλεσμα, η εφαρμογή παρέχει ακόμα περισσότερες δυνατότητες σε έναν σχεδιαστή παιχνιδιών διαδικαστικής αφήγησης από όσες αξιοποιήθηκαν και παρουσιάστηκαν παραπάνω.

Τέλος, κατά την υλοποίηση της εργασίας αυτής αποδείχθηκαν σημαντικές οι γνώσεις τεχνολογίας λογισμικού. Η διαχείριση ενός τόσο μεγάλου αριθμού κλάσεων χωρίς τα εργαλεία που παρέχει η UML θα ήταν εμφανώς δυσκολότερη, οπότε ήταν απαραίτητη η εξοικείωση με τις βασικές έννοιες της ανάλυσης και του σχεδιασμού λογισμικού, των περιπτώσεων χρήσης (use cases), καθώς και των διαγραμμάτων κλάσεων.

ΒΙΒΛΙΟΓΡΑΦΙΑ

[1] HENDRIKX, Mark, et al. Procedural content generation for games: A survey. *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)*, 2013, 9.1: 1-22.

[2] SHORT, Tanya; ADAMS, Tarn (ed.). *Procedural generation in game design*. CRC Press, 2017.

[3] PLOTKIN, Andrew. Characterizing, if not defining, interactive fiction. *IF Theory Reader*, 2011, 59.

[4] ΑΒΟΥΡΗΣ, Νικόλαος, ΚΟΥΚΙΑΣ, Μιχαήλ, ΠΑΛΙΟΥΡΑΣ, Βασίλειος, ΣΓΑΡΜΠΑΣ, Κυριάκος. *Python: εισαγωγή στους υπολογιστές*. Πανεπιστημιακές Εκδόσεις Κρήτης, 2016.

[5] Jetbrains. *Pycharm: The Python IDE for Professional Developers* [online]. [viewed date 11 June 2021]. Available from: <https://www.jetbrains.com/pycharm/>

[6] Python Software Foundation. *Python* [online]. Python Software Foundation, 2021. [viewed date 11 June 2021]. Available from: <https://www.python.org>

[7] Anaconda, Inc. *Miniconda* [online]. 2017. [viewed date 11 June 2021]. Available from: <https://www.anaconda.com>

[8] KETTLER, Rafe. A Guide to Python's Magic Methods. *URL: https://rszalski. github. io/magicmethods/# intro*, 2015.

[9] ORRU, Matteo, et al. How do Python programs use inheritance? a replication study. In: *2015 Asia-Pacific Software Engineering Conference (APSEC)*. IEEE, 2015. p. 309-315.

[10] Python Software Foundation. *Python Standard Library: threading — Thread-based parallelism* [online]. Python Software Foundation, 2021. [viewed date 11 June 2021]. Available from: <https://docs.python.org/3.7/library/threading.html>

[11] Python Software Foundation. *Python Standard Library: json — JSON encoder and decoder* [online]. Python Software Foundation, 2021. [viewed date 11 June 2021]. Available from: <https://docs.python.org/3.7/library/json.html>

[12] Numpy. *API Reference: Random sampling (numpy.random)* [online]. The SciPy community, 2008-2020. [viewed date 11 June 2021]. Available from: <https://numpy.org/doc/stable/reference/random/index.html>

[13] NLTK Project. *Natural Language Toolkit -- NLTK 3.6.2 documentation* [online]. [viewed date 11 June 2021]. Available from: <https://www.nltk.org>

[14] The Kivy Authors. *Kivy* [online]. [viewed date 11 June 2021]. Available from: <https://kivy.org/#home>

[15] Python Software Foundation. *Python Standard Library: string — Common string operations* [online]. Python Software Foundation, 2021. [viewed date 11 June 2021]. Available from: <https://docs.python.org/3.7/library/string.html>

[16] Python Software Foundation. *Python Standard Library: random — Generate pseudo-random numbers* [online]. Python Software Foundation, 2021. [viewed date 11 June 2021]. Available from: <https://docs.python.org/3.7/library/random.html>

[17] Python Software Foundation. *Python Standard Library: sys — System-specific parameters and functions* [online]. Python Software Foundation, 2021. [viewed date 11 June 2021]. Available from: <https://docs.python.org/3.7/library/sys.html>

[18] Python Software Foundation. *Python Standard Library: os — Miscellaneous operating system interfaces* [online]. Python Software Foundation, 2021. [viewed date 11 June 2021]. Available from: <https://docs.python.org/3.7/library/os.html>

[19] Python Software Foundation. *Python Standard Library: time — Time access and conversions* [online]. Python Software Foundation, 2021. [viewed date 11 June 2021]. Available from: <https://docs.python.org/3.7/library/time.html>

[20] Python Software Foundation. *Python Standard Library: shutil — High-level file operations* [online]. Python Software Foundation, 2021. [viewed date 11 June 2021]. Available from: <https://docs.python.org/3.7/library/shutil.html>

[21] The pandas development team. p*andas-dev/pandas: Pandas* [online]. Zenodo, 2020. [viewed date 11 June 2021]. Available from: <https://doi.org/10.5281/zenodo.3509134>

[22] MONTFORT, Nick. *Twisty Little Passages: an approach to interactive fiction*. Mit Press, 2005.

[23] NELSON, Graham. *The Inform Designer's Manual*. Interactive Fiction Library, 2001.

[24] SHORT, Tanya X.; ADAMS, Tarn (ed.). *Procedural Storytelling in Game Design*. CRC Press, 2019.

[25] HUA, Minh; RALEY, Rita. Playing With Unicorns: AI Dungeon and Citizen NLP. *DHQ: Digital Humanities Quarterly*, 2020, 14.4.

[26] ROBERTS, Michael J. *TADS: Create your own Interactive Fiction* [online]. 2001-2013. [viewed date 11 June 2021]. Available from: <http://www.tads.org>

[27] NELSON, Graham. *Inform 7* [online]. [viewed date 11 June 2021]. Available from: <http://inform7.com>

[28] Interactive Fiction Technology Foundation. *Twine* [online]. [viewed date 11 June 2021]. Available from: <https://twinery.org>

[29] Tom Rothamel. *Ren'Py* [online]. [viewed date 11 June 2021]. Available from: <https://renpy.org>

[30] Textadventures.co.uk. *Quest: Build text adventure games and interactive fiction* [online]. [viewed date 11 June 2021]. Available from: <http://textadventures.co.uk/quest/>

[31] Textadventures.co.uk. *Squiffy: A simple way to write interactive fiction* [online]. [viewed date 11 June 2021]. Available from: <http://textadventures.co.uk/squiffy>

[32] ROBERTS, Michael J. *TADS Overview: The Language* [online]. [viewed date 11 June 2021]. Available from: <http://www.tads.org/ov_lang.htm>

[33] NELSON, Graham. *Inform 7 Documentation: Part I - Writing with Inform* [online]. [viewed date 11 June 2021]. Available from: <http://inform7.com/book/WI_1_1.html>

[34] ΘΡΑΜΠΟΥΛΙΔΗΣ, Κλεάνθης. *Αντικειμενοστρεφής Προγραμματισμός - Java*. Εκδόσεις Τζιόλα, 2019.

[35] ROBERTS, Michael J. *TADS 3 Technical Manual: TADS 3 In Depth*. "Choosing a Conversation System" [online]. 2006. [viewed date 11 June 2021]. Available from: <https://www.tads.org/t3doc/doc/techman/convbkg.htm>

[36] ROBERTS, Michael J. *TADS 3 Technical Manual: TADS 3 In Depth*. "Programming Conversations with NPCs" [online]. 2006. [viewed date 11 June 2021]. Available from: <https://www.tads.org/t3doc/doc/techman/t3conv.htm>

[37] SHORT, Emily. *Emily Short's Interactive Storytelling: Conversation* [online]. [viewed date 11 June 2021]. Available from: <https://emshort.blog/how-to-play/writing-if/my-articles/conversation/>

[38] MATEAS, Michael; STERN, Andrew. Façade: An experiment in building a fully-realized interactive drama. In: *Game developers conference*. 2003. p. 4-8.

[39] EVANS, Richard, SHORT, Emily, GRAHAM, Nelson. *Versu* [online]. [viewed date 11 June 2021]. Available from: <https://versu.com>

[40] Ocelot Society. *event[0]* [online]. [viewed date 11 June 2021]. Available from: <https://ocelotsociety.itch.io/event0>

[41] MONTFORT, Nick. *Generating narrative variation in interactive fiction*. Philadelphia: University of Pennsylvania, 2007.

[42] SI, Mei; MARSELLA, Stacy C.; PYNADATH, David V. Thespian: Using multi-agent fitting to craft interactive drama. In: *Proceedings of the fourth international joint conference on Autonomous agents and multiagent systems*. 2005. p. 21-28.

[43] BARROS, Gabriella AB; LIAPIS, Antonios; TOGELIUS, Julian. Playing with data: Procedural generation of adventures from open data. 2016.

[44] HUA, Minh; RALEY, Rita. Playing With Unicorns: AI Dungeon and Citizen NLP. *DHQ: Digital Humanities Quarterly*, 2020, 14.4.

[45] SANDERS, Alton F.; SANDERS, Ruth H. Syntactic parsing: A survey. *Computers and the Humanities*, 1989, 13-30.

[46] LI, Zhuang; QU, Lizhen; HAFFARI, Gholamreza. Context Dependent Semantic Parsing: A Survey. *arXiv preprint arXiv:2011.00797*, 2020.

[47] KAMATH, Aishwarya; DAS, Rajarshi. A survey on semantic parsing. *arXiv preprint arXiv:1812.00978*, 2018.

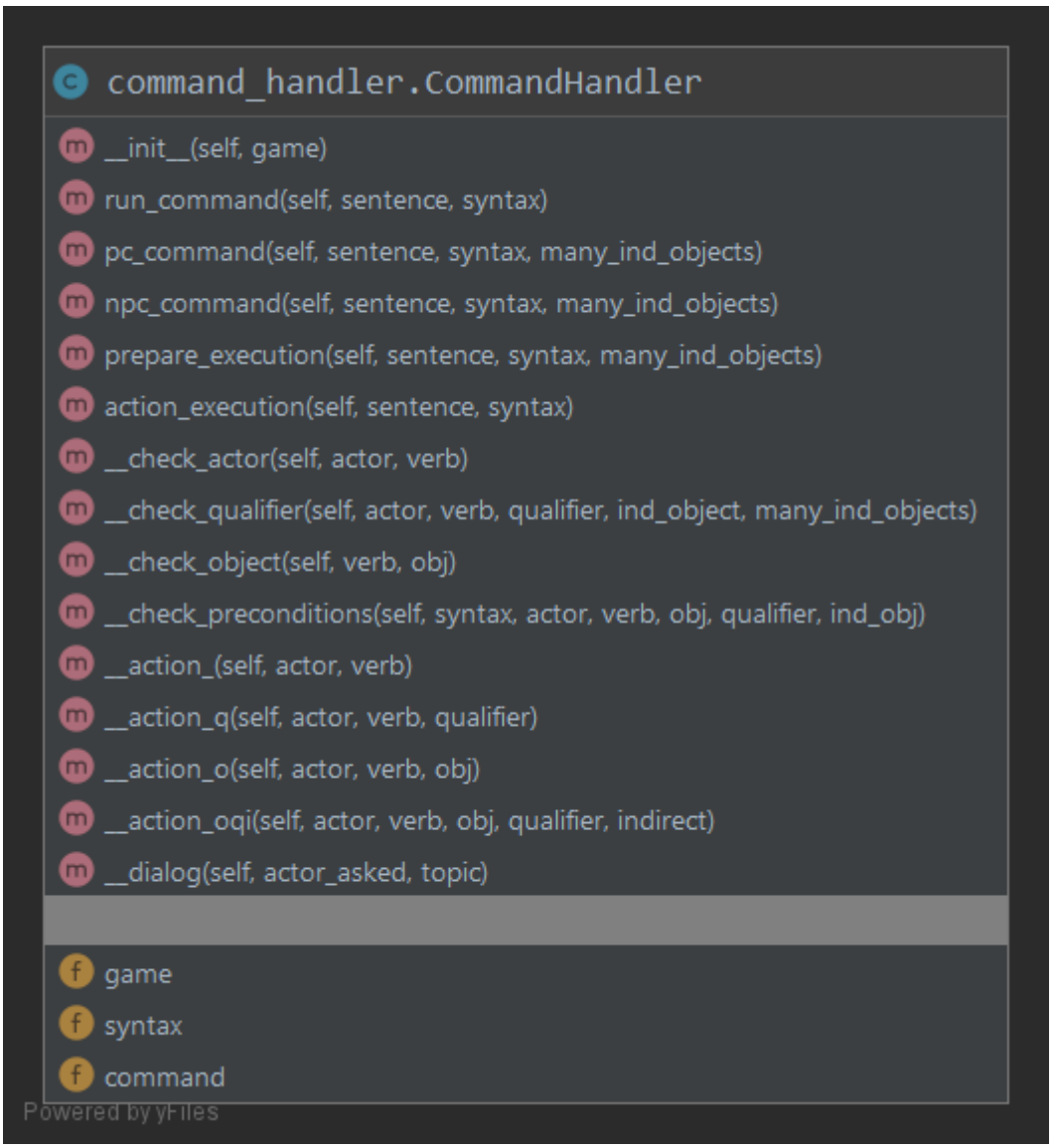
[48] XIE, Ziang. Neural text generation: A practical guide. *arXiv preprint arXiv:1711.09534*, 2017.

[49] UOTILA, Aarne. *Procedural text generation with stateful context-free grammars*. 2018. Master's Thesis.

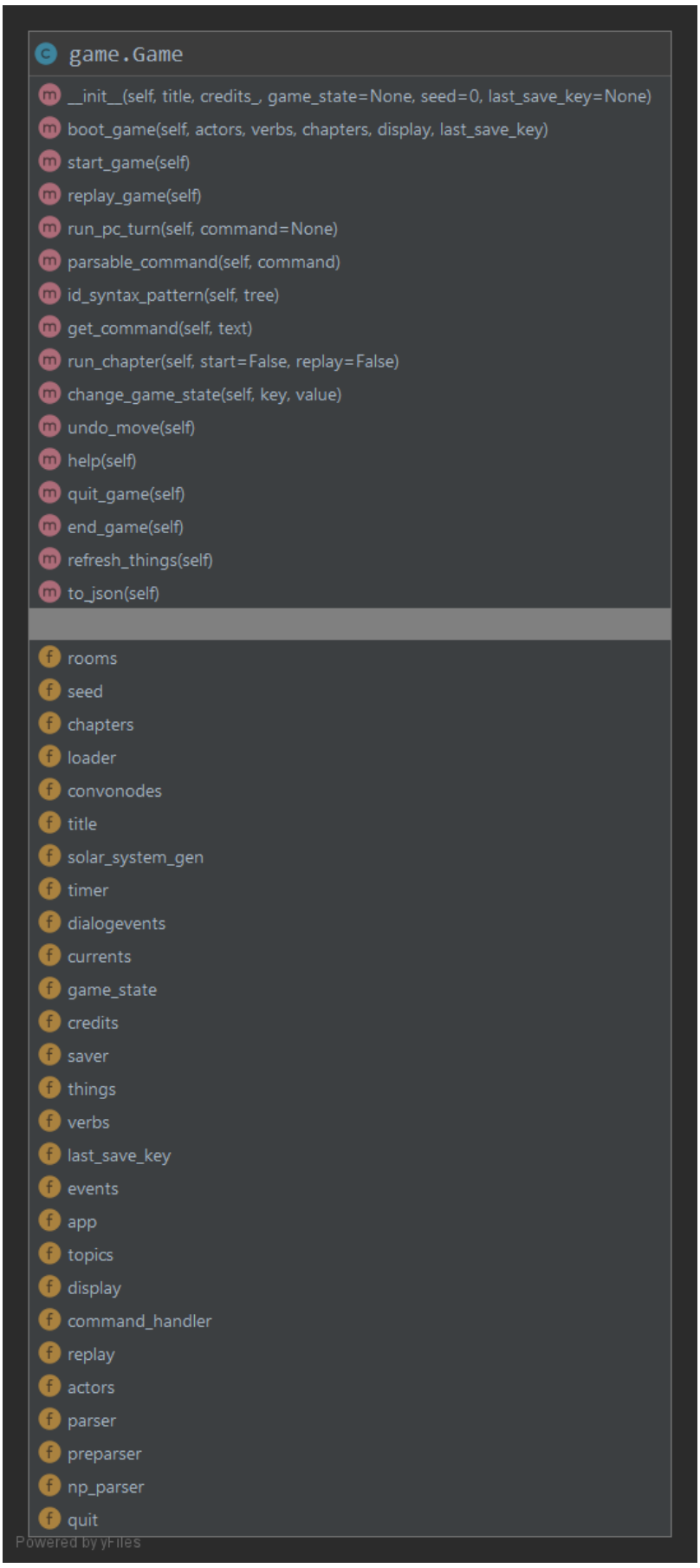
[50] JAMES, Frederick. A review of pseudorandom number generators. *Computer physics communications*, 1990, 60.3: 329-344.

ΠΑΡΑΡΤΗΜΑΤΑ

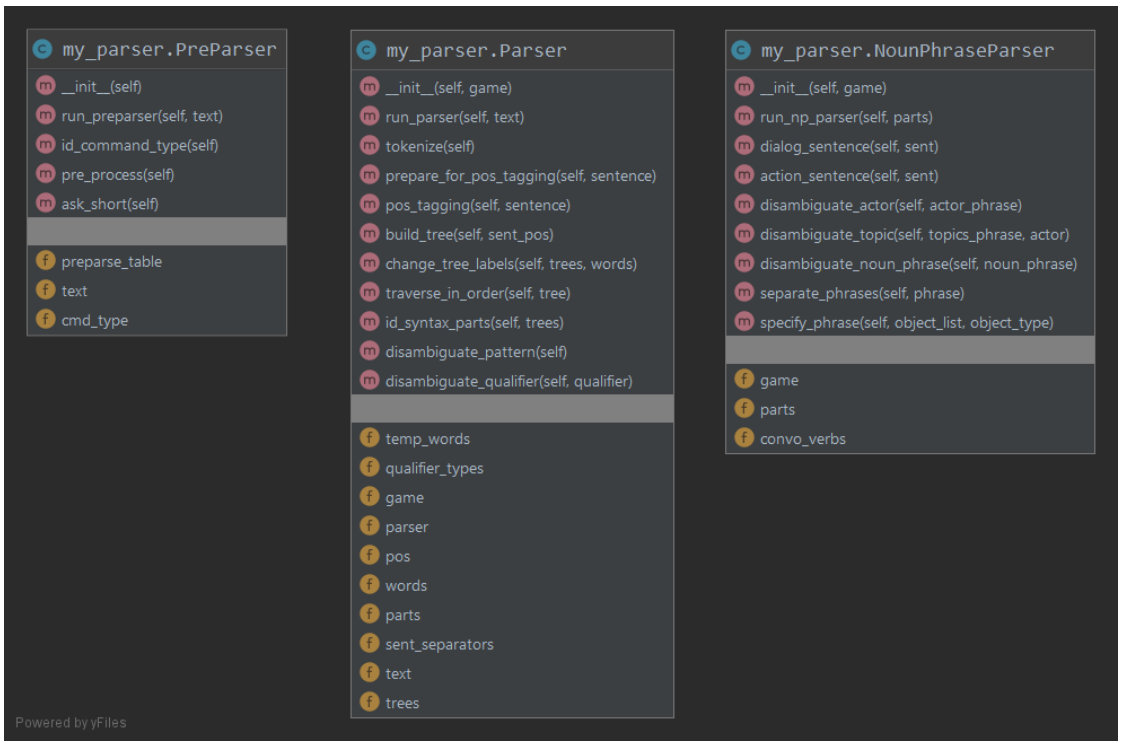
Α. Διαγράμματα κλάσεων



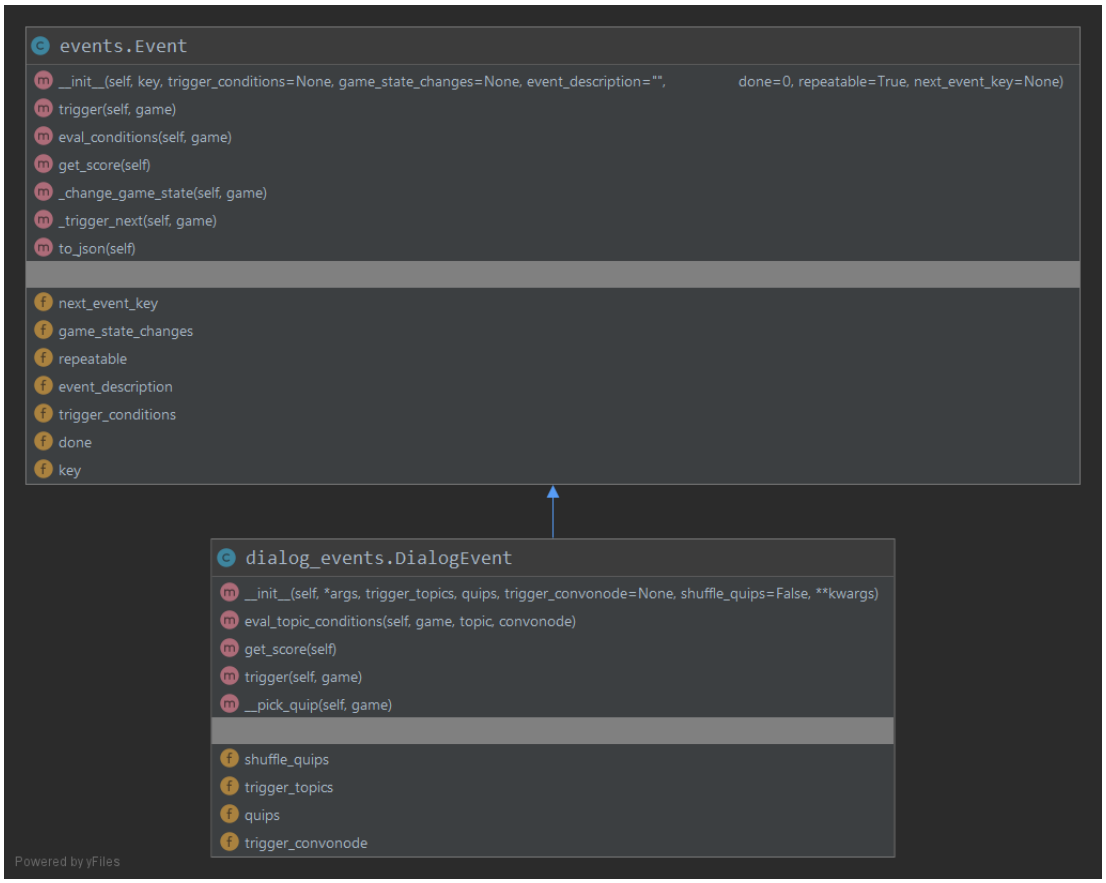
Εικ. 12 Ιδιότητες και μέθοδοι της κλάσης CommandHandler.



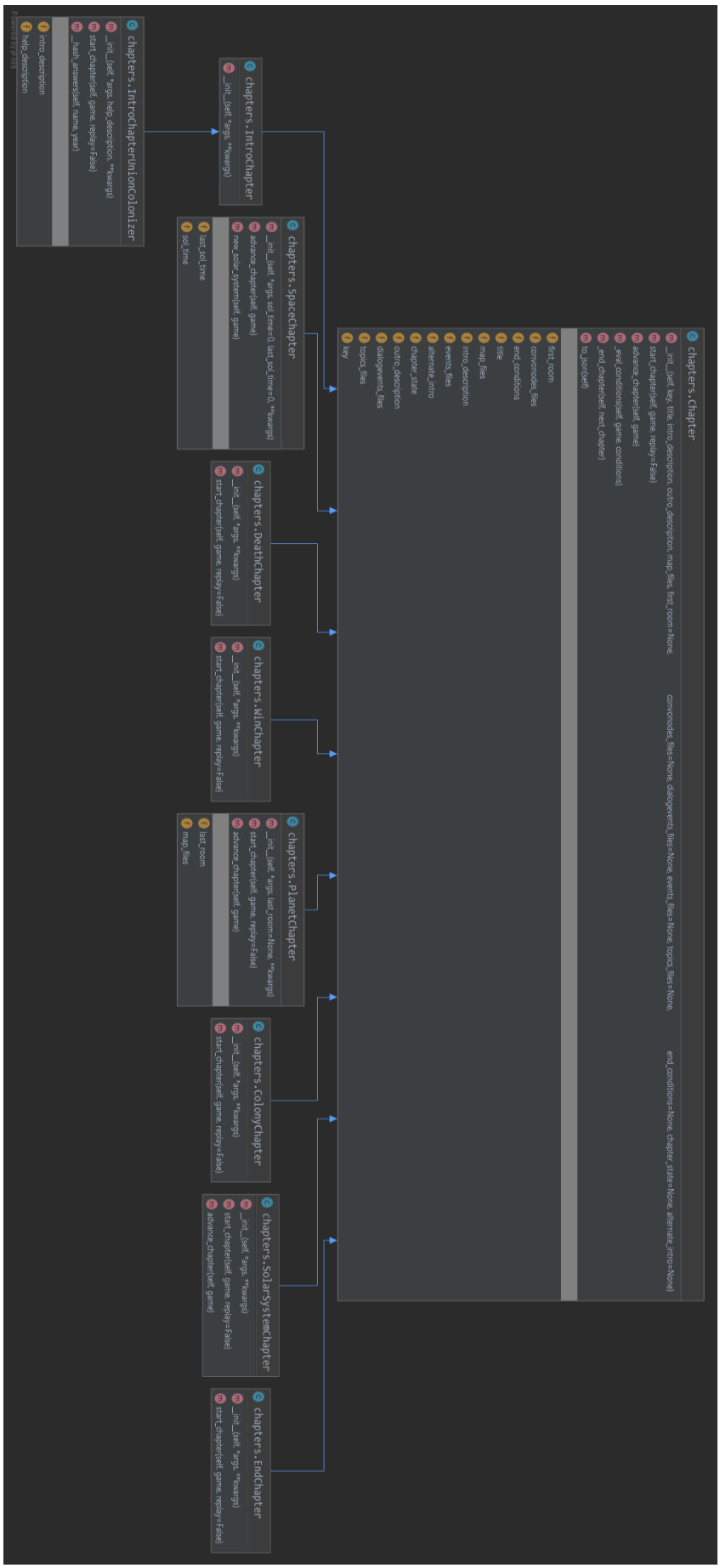
Εικ. 13 Ιδιότητες και μέθοδοι της κλάσης Game.



Εικ. 14 Ιδιότητες και μέθοδοι των κλάσεων PreParser, Parser, NounPhraseParser.



Εικ. 15 Ιδιότητες και μέθοδοι των κλάσεων Event και DialogEvent.



Εικ. 16 Η κλάση Chapter και οι υποκλάσεις της.

Εικ. 17 Η κλάση Entity και οι υποκλάσεις της.

Εικ. 18 Διάγραμμα όλων των κλάσεων της εφαρμογής.

B. GitHub Repository και Demo Video

Το GitHub Repository με τον πλήρη κώδικα που αναπτύχθηκε, καθώς και τον σύνδεσμο προς ένα video που επιδεικνύει τις λειτουργίες της εφαρμογής βρίσκεται εδώ:

https://github.com/Frankkie/Thesis-Project-IF-Game