ΤΕΧΝΟΛΟΓΙΚΟ ΕΚΠΑΙΔΕΥΤΙΚΟ ΙΔΡΥΜΑ ΚΕΝΤΡΙΚΗΣ ΜΑΚΕΔΟΝΙΑΣ

ΣΧΟΛΗ ΤΕΧΝΟΛΟΓΙΚΩΝ ΕΦΑΡΜΟΓΩΝ

ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ ΤΕ

ΑΝΑΠΤΥΞΗ ΕΡΓΑΛΕΙΟΥ ΓΙΑ ΤΗΝ ΣΧΕΔΙΑΣΗ ΠΑΙΧΝΙΔΙΩΝ ΜΕ ΤΗΝ ΒΟΗΘΕΙΑ ΥΠΟ-ΛΟΓΙΣΤΗ

Πτυχιακή εργασία του

Γιώργου Μιχαηλίδη

Επιβλέπων: Δρ. Νικόλαος Πεταλίδης. Επιστημονικός Συνεργάτης

ΣΕΡΡΕΣ, ΦΕΒΡΟΥΑΡΙΟΣ 2016

Υπεύθυνη δήλωση

Υπεύθυνη Δήλωση: Βεβαιώνω ότι είμαι συγγραφέας αυτής της πτυχιακής εργασίας και ότι κάθε βοήθεια την οποία είχα για την προετοιμασία της, είναι πλήρως αναγνωρισμένη και αναφέρεται στην πτυχιακή εργασία. Επίσης έχω αναφέρει τις όποιες πηγές από τις οποίες έκανα χρήση δεδομένων, ιδεών ή λέξεων, είτε αυτές αναφέρονται ακριβώς είτε παραφρασμένες. Επίσης βεβαιώνω ότι αυτή η πτυχιακή εργασία προετοιμάστηκε από εμένα προσωπικά ειδικά για τις απαιτήσεις του προγράμματος σπουδών του Τμήματος Μηχανικών Πληροφορικής ΤΕ του Τ.Ε.Ι. Κεντρικής Μακεδονίας.

Σύνοψη

Τα πρώτα ηλεκτρονικά παιχνίδια είχαν γραφτεί εξ'ολοκλήρου σε υλισμικό. Από τότε, οι κάρτες γραφικών και οι μικροεπεξεργαστές βελτιώθηκαν, δημιουργήθηκαν κονσόλες φτιαγμένες αποκλειστικά για ηλεκτρονικά παιχνίδια, με ειδικά χειριστήρια τα οποία σου προσφέρουν διαφορετικές εμπειρίες. Η διαδικασία ανάπτυξης λογισμικού είναι ακριβή και ο σχεδιασμός γίνεται όλο πιο σύνθετος και περίπλοκος. Τα έργα γίνονται όλο πιο απαιτητικά και δαπανηρά. Δημιουργήθηκε η ανάγκη για ένα εργαλείο το οποίο να παρέχει ένα ομοιογενές περιβάλλον για την ανάπτυξη σύνθετων έργων. Ένα CASE (Computer Aided Software Engineering) tool είναι ένα λογισμικό-εργαλείο το οποίο απλοποιεί τον κύκλο ανάπτυξης ενός λογισμικού. Στο τομέα του σχεδιασμού παιχνιδιών το πιο διαδεδομένο CASE tool είναι η μηχανή γραφικών. Μια μηχανή γραφικών είναι μια σουίτα από επαναχρησιμοποιήσιμα οπτικά εργαλεία τα οποία βρίσκονται σε ένα ενιαίο περιβάλλον. Σκοπός της πτυχιακής είναι να αναγνωριστούν μοτίβα και τεχνικές δημιουργίας παιχνιδιών, ώστε να δημιουργηθεί ένα εργαλείο το οποίο να το προσεγγίζει από υψηλό επίπεδο με αφαιρέσεις για εύκολη μοντελοποίηση και αυτοματοποίηση κατά τη δημιουργία.

Περιεχόμενα

Κατάλογος πινάκων

Κατάλογος διαγραμμάτων

Πρόλογος

Εδώ μπορεί να μπει πρόλογος. (Δεν είναι απαραίτητο).

Ευχαριστίες

Ευχαριστίες (στο μπαμπά, στη μαμά, κτλ)

Ορισμοί

Ορισμοί εννοιών που μπορεί να είναι χρήσιμοι. Για παράδειγμα:

LaTeX Σύστημα στοιχειοθεσίας κειμένων

Κεφάλαιο 1

Εισαγωγή

1.1 Η τυπογραφία σήμερα

Αυτή είναι η αναφορά σε ένα άρθρο περιοδικού:(**Schmidt98**).Αυτή είναι η αναφορά σε ένα βιβλίο:(**goosens93**). Αυτή είναι η αναφορά σε ένα ελληνικό βιβλίο:(**Chatzigeorgiou05**). Βιβλίο στα ελληνικά με ξένο συγγραφέα:(**Sommerville09**). Άρθρο σε συνέδριο (**4343930**).

Τέλος αναφορά σε ιστοσελίδα: (**Wikipedia_BibTeX**).

Εδώ αναφερόμαστε στο σχήμα 1.1:

Διάγραμμα 1.1: Παράδειγμα εικόνας

και εδώ στον πίνακα 1.1:

Πίνακας 1.1: Παράδειγμα πίνακα

Κίνητρα	Παραδείγματα ευρημάτων	Αριθμός μελετών
Ταύτιση με το έργο	Ξεκάθαροι στόχοι	20
Καλό management	Ομαδικότητα	16
Συμμετοχή υπαλλήλων	Συμμετοχή στις αποφάσεις	16
Προοπτικές εξέλιξης	Προοπτικές προαγωγής	15
Ποικιλία στην εργασία	Καλή χρήση ικανοτήτων	14
Αίσθηση του να ανήκεις κάπου	Υποστηρικτικές σχέσεις	14
Αμοιβές και κίνητρα	Αυξημένος μισθός	14

Παράρτημα 1

Μηχανές γραφικών

Παράρτημα 2

Διαδικτύωση

Διαδικτύωση στα ηλεκτρονικά παιχνίδια έχουμε όταν περισσότεροι από ένας παίχτες σε διαφορετικές πλατφόρμες ή υπολογιστές, μοιράζονται και αλληλεπιδρούν στο ίδιο εικονικό περιβάλλον.

2.0.1 Το πρόβλημα

Περιγραφή του προβλήματος Διάφοροι παίχτες σε διάφορα σημεία του πλανήτη θέλουν να μοιραστούν ένα εικονικό περιβάλλον σε πραγματικό χρόνο με σκοπό την συνεργασία ή αντιπαλότητα. Ο κόσμος είναι ένα υπερσύνολο του offline κόσμου με επιπλέων στοιχεία κοινωνικοποίησης όπως η επικοινωνία μέσω μηνυμάτων ή φωνής.

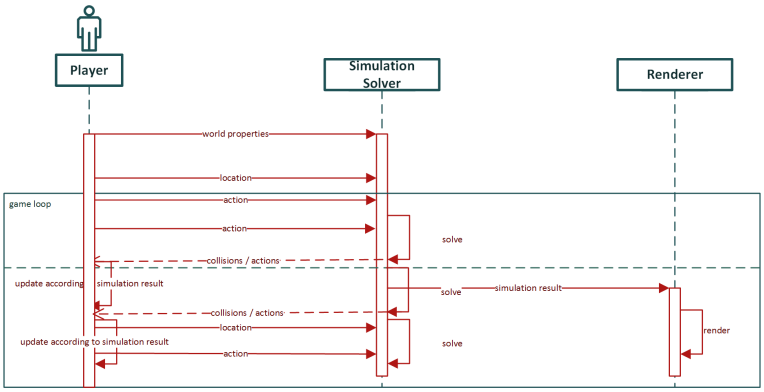
Κατανόηση του προβλήματος Ένας εικονικός κόσμος, περιλαμβάνει πολλές οντότητες οι οποίες αλληλεπιδρούν μεταξύ τους μέσω των μηχανισμών, νόμων και κανόνων που διέπουν τον κόσμο. Παράδειγμα μηχανισμού είναι η προσομοίωση του φυσικού κόσμου, όπου οι οντότητες αναποκρίνονται σε νόμους της φυσικής.

Κατά την ενημέρωση του κόσμου, ο προσομοιωτής χρησιμοποιώντας τους νόμους, τους κανόνες και τους μηχανισμούς που διέπουν τον κόσμο, παίρνει ως είσοδο τις οντότητες και τα ειδικά βάρη των ιδιοτήτων τους, την απόλυτη θέση τους στο σύστημα συντεταγμένων του κόσμου, και το χρονικό διάστημα της προσομοίωσης και αναλύει την προσομοίωση. Η ανάλυση της προσομοίωσης για να είναι επιτρεπτά ακριβής πρέπει να γίνεται περίπου 80-100 φορές το δευτερόλεπτο. [αναφορά σε πηγή]

Στο τέλος της προσομοίωσης, η μηχανή γραφικών αποτυπώνει τον κόσμο στις εξόδους με αναφορές σε στατικά assets, σε αλγόριθμους παραγωγής δυναμικών assets για

την αναπαράσταση του κόσμου.

Εξαγωγή απαιτήσεων Με βάση το πρόβλημα καταλήγουμε στο παρακάτω διάγραμμα ακολουθίας.



Διάγραμμα 2.1: Network Sequence

Βλέποντας το διάγραμμα καταλαβαίνουμε ότι σε η διαδικασία rendering περιλαμβάνει στατικά στοιχεία και αλγόριθμους, τα οποίοι μπορούν να φορτώνονται τοπικά στον κάθε υπολογιστή, και δεν είναι απαραίτητα για την επίλυση της προσομοίωσης. Τα απαραίτητα στοιχεία για την προσομοίωση τα οποία πρέπει να μοιράζονται μεταξύ των παιχτών είναι:

- Οι ιδιότητες της οντότητας με βάση τους νόμους του κόσμου. Οι ιδιότητες αυτές δεν ενημερώνονται συχνά.
- Οι αλλαγές στο σύστημα συντεταγμένων και οι διάφορες ενέργειες της κατευθυνόμενης οντότητας κατά την πάροδο του χρόνου. Οι αλλαγές τοποθεσίας και ενέργειες γίνονται πολλές φορές ανά δευτερόλεπτο. Η προσομοίωση για να είναι ακριβής πρέπει να ενημερώνεται για τις διάφορες ενέργειες σε πραγματικό χρόνο.

2.0.2 Εκπόνηση σχεδίου

Η ανάγκη αποστολής πολλών μηνυμάτων ανά δευτερόλεπτο οδηγεί στη χρήση των network sockets. Τα sockets χρησιμοποιωντας ένα IP και Port και επιτρέπουν την αποστολή και παραλαβή μηνυμάτων με βάση κάποιου πρωτόκολλου.

Επιλογή πρωτοκόλου

TCP Το TCP (transmission control protocol), είναι το πιο συχνά χρησιμοποιημένο πρωτόκολλο. Η διασύνδεση με TCP είναι αξιόπιστη και τα μηνύματα παραλαμβάνονται στη σειρά αποστολής. Η αξιοπιστία όμως έρχεται με ένα μικρό κόστος απόδοσης.

UDP Το UDP (user diagram protocol) δεν περιλαμβάνει την αξιοπιστία και την εγγύηση της αλληλουχίας μηνυμάτων. Η απουσία λειτουργιών όμως, το κάνει το πιο γρήγορο σε αποστολή μηνυμάτων πρωτόκολλο.

Η επιλογή πρωτοκόλλου γίνεται ανάλογα με το γενικότερο πλαίσιο και τη συγκεκριμένη χρήση του πακέτου αποστολής. Στην αποστολή της τοποθεσίας, το οποίο γίνεται 20φορές / δευτερόλεπτο, η αξιοπιστία δεν είναι το βασικότερο, αλλά η απόδοση. Στην αποστολή των στοιχειών του χρήστη κατά την έναρξη, ή ενώς γραπτού μηνύματος πρέπει να είναι αξιόπιστη.

Επιλογή αρχιτεκτονικής δικτύου Οι αρχιτεκτονικές δικτύου χωρίζονται ανάλογα με το που γίνεται η επίλυση και επεξεργασία της προσομοίωσης.

Client-server model στο οποίο ο client απλά κάνει render και το μεγαλύτερο κομμάτι της λογικής και της προσωμοίωσης τρέχει στον server. Ο server στέλνει οδηγίες στον client για το τι να κάνει render και ο client απλά υπακούει.

Client on top of server model ο client είναι και server, δηλαδή οι μηχανές που έχουν τον client έχουν και τον server.

Peer-to-peer οι μηχανές συμπεριφέρονται μερικώς ως clients και μερικώς ως servers, δηλαδή έχουν και στοιχεία λογικής και επεξεργασίας.

Η client-server αρχιτεκτονική εφαρμόζεται σε παιχνίδια τα οποία περιλαμβάνουν πολύ μεγάλους κόσμους και η προσομοίωση περιλαμβάνει αλληλεπιδράσεις μεταξύ πολλών οντοτήτων. Οι προσωμοιώσεις αυτές γίνονται σε εξειδικευμένες μηχανές και όχι στον προσωπικό υπολογιστή του κάθε χρήστη γιατί χρειάζονται μεγάλους υπολογιστικούς πόρους. Επίσης ο server μπορεί να λύσει race conditions και να λειτουργήσει ως ”διαιτητής” μεταξύ δύο οντοτήτων οι οποίες ζητούν πρόσβαση στο ίδιο σύστημα κατά το ίδιο χρονικό διάστημα. Οι αρχιτεκτονικές στις οποίες ο client περιλαμβάνει

στοιχεία επεξεργασίας του κοινόχρηστου κόσμου, είναι πιο δύσκολες στην ανάπτυξη συντήρηση γιατί δημιουργούνται race conditions στο χρονικό διάστημα αποστολής-παραλαβής μηνυμάτων που προορίζονται σε αλληλοεξαρτώμενες λειτουργίες.

2.0.3 Σχεδίαση του framework

Κατά το σχεδιασμό διαδικτυακών παιχνιδιών πολλά μοτίβα και στερεότυπα κώδικα επαναλαμβάνονται χωρίς να συμβάλλουν στην λογική ή μηχανισμούς. Σκοπός του framework είναι να μειώσει και να απλοποιήσει τον επαναλαμβανόμενο κώδικα και να προμηθεύσει τον χρήστη με μοτίβα και μοντέλα ούτως ώστε να εστιάσει μόνο στα απαραίτητα.

2.0.4 Έννοιες

Serialization Σε μια αντικειμενοστραφή γλώσσα χρησιμοποιούνται κλάσεις για να μοντελοποιήσουν το πρόβλημα. Όμως τα αντικείμενα τον κλάσεων δεν μπορούν να σταλούν μέσω network socket στην αρχική τους μορφή, πρέπει να γίνει μια μετατροπή σε binary για να είναι συνεπής με το format των δεδομένων που αποστέλλονται μέσω του socket . Κατά την αποστολή έχουμε το serialization τον αντικείμενων, και κατά την παραλαβή το deserialization του πακέτου στο αντικείμενο που αντιπροσωπεύει.

Message Types Οι διάφοροι τύποι μηνυμάτων χρησιμοποιούνται για να ξεχωρίσουν την κατάσταση στην οποία βρίσκεται ο client ή ο server.

Αλλαγή κατάστασης info

Connecting info

Connected info

Disconnecting info

Disconnected info

Data info

ErrorMessage info

WarningMessage

VerboseMessage

ConnectionApproval

DiscoveryRequest

DiscoveryResponse

Οι τύποι μηνυμάτων μπορούν να μοντελοποιηθούν ως ένα enum και να προστεθεί η πληροφορία τους στο πρώτο byte του serialized μηνύματος. Το πρώτο byte του παραληφθέντα μηνύματος περιλαμβάνει τον τύπο του μηνύματος.

NetworkManager Ο network manager είναι υπεύθυνος για την διαχείριση των sockets, την αποστολή / παραλαβή μηνυμάτων, ενθυλακώνει λειτουργιες για την διαδικτύωση και είναι ο πηρύνας του framework και επεκτίνεται από τον client, server οι οποίοι προσθέτουν λειτουργίες.

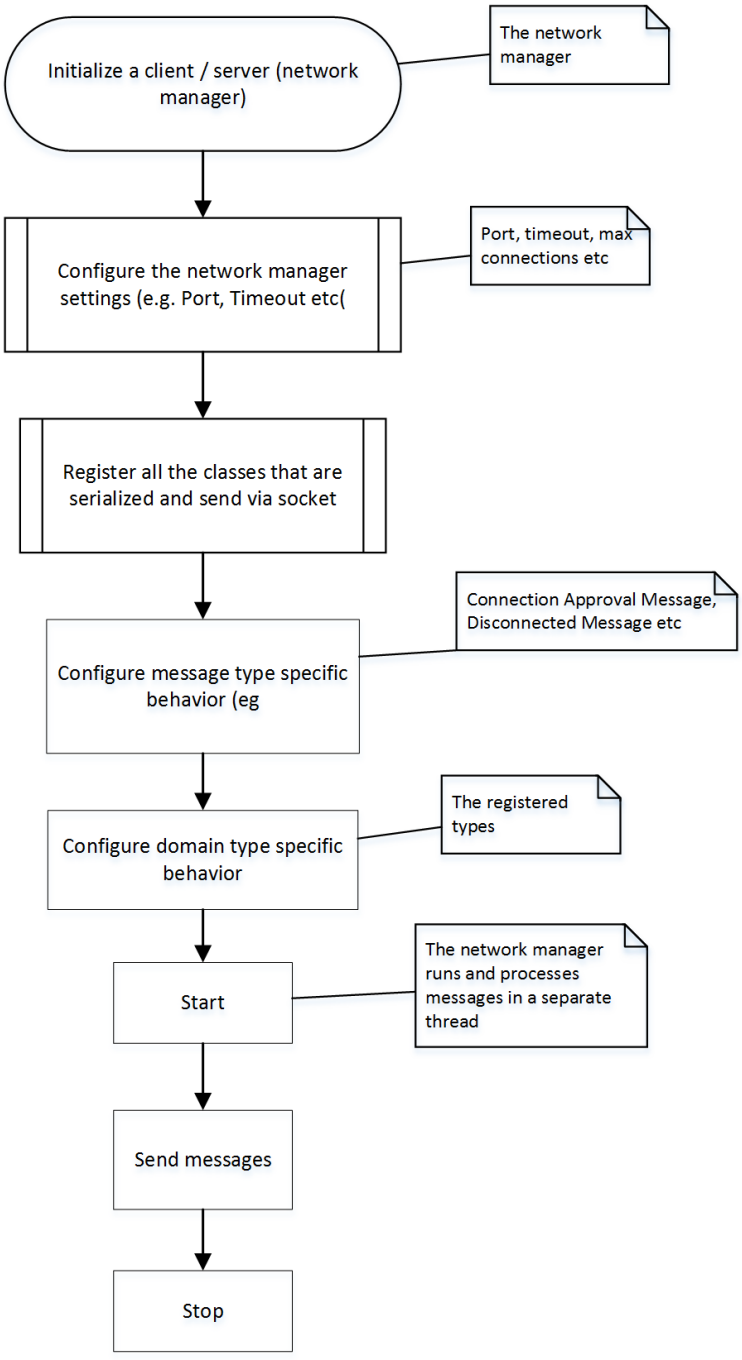
2.0.5 Υλοποίηση

Τρόπος χρήσης Η διαδικασία χρήσης πρέπει να είναι εξορθολογιστική και ξεκάθαρη για εύκολη και γρήγορη ανάπτυξη και περιοριστική για αποφυγή bugs και προβλημάτων.

2.0.6 Αρχιτεκτονική

Η αρχιτεκτονική στηρίζεται στην ανταλλαγή μηνυμάτων-κλάσεων και χωρίζεται στα παρακάτω επίπεδα.

- Packages Χειρίζεται το serialization, αναλύει τα εισερχόμενα μηνύματα και χειρίζεται τις ανακατευθύνσεις του εκτελούμενου κώδικα κατά την παραλαβή μηνυμάτων.
- Networking Διαχειρίζεται τις συνδεσεις των χρηστών, τα sockets και ο,τι έχει να κάνει με διασύνδεση.
- Auditing Αφαιρετικό επίπεδο για logging. Ο χρήστης μπορεί να ενσωματώσει τη δική του λογική παρακολούθησης μηνυμάτων.



Διάγραμμα 2.2: Network Usage Diagram

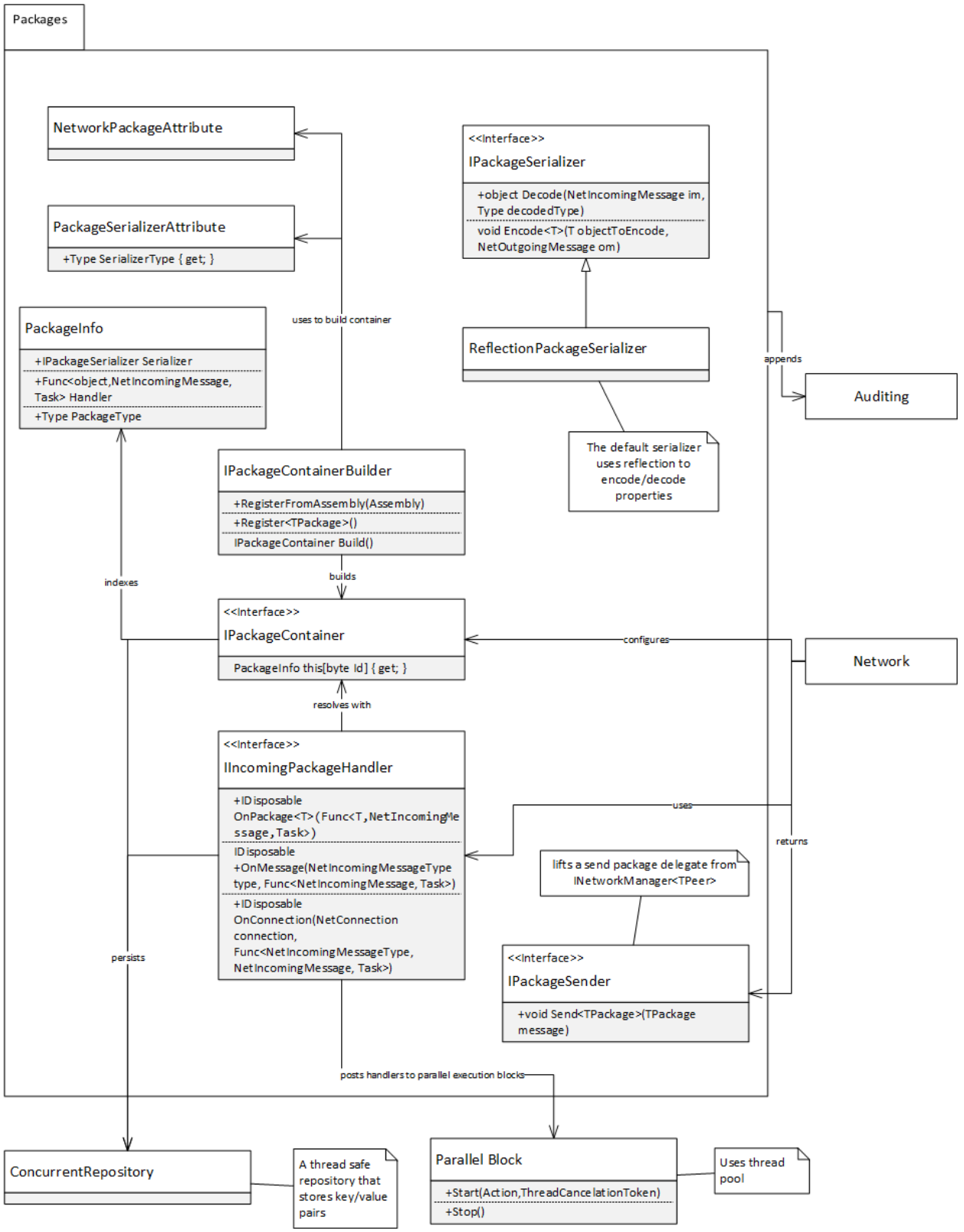
- Infrastructure Το επίπεδο αυτό περιέχει επαναχρησιμοποιήσημο κώδικα των πακέτων όπως το ConcurrentRepository, για thread-safe αποθήκευση κλειδιών και τιμών, και το ParallelBufferBlock το οποίο εκτελεί κώδικα σε buffer άλλου thread για ασύγχρονη επεξεργασία.

Packages Module Το serialization των μηνυμάτων πρέπει να είναι ντερερμενιστικό ώστε να αναγνωρίζεται ο τύπος του μηνύματος και η κλάση την οποία αντιπροσωπεύει και ελαφρύς ώστε να μην επιβαρύνεται το network με επιπλέον δεδομένα. Ο ενσωματομένος serializer χρησιμοποιεί reflection για να αναγνωρίσει τα primitive properties της κλάσης. Για προχωριμένο serialization περίπλοκων τύπων, ο χρήστης έχει τη δυνατότητα να συμπεριλάβει τον δικό του serializer με το implementation του IPackageSerializer interface και την χωρήγηση του σημειώνοντας την κλάση με το PackageSerializerAttribute και τον τύπο του serializer.

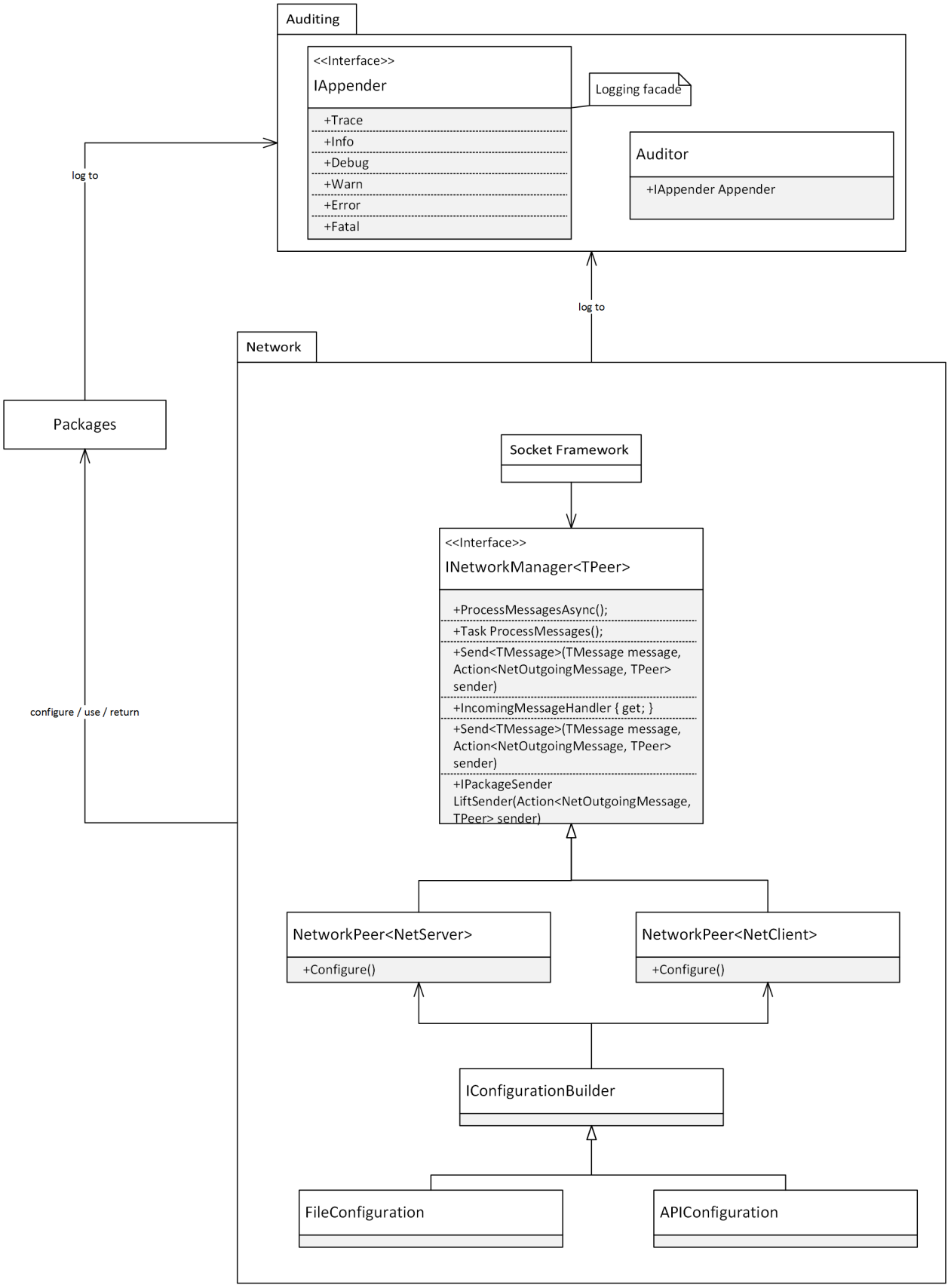
Κατά την προετοιμασία του network manager, ο χρήστης καλείται να καταχωρίσει στο Container ποιες κλασεις θα χρησιμοποιηθούν κατά την ανταλλαγή μηνυμάτων. Οι κλάσεις που προωρίζονται για serialization σημειώνονται με κάποιο Attribute και η καταχώριση μπορεί να γίνει δυναμικά με reflection. Όταν τελειώσει η καταχώριση, το container χρησιμοποιώντας ένα ντετερμενιστικό αλγόριθμο ταξινομώντας με βάση το όνομα της κλάσης στο χώρο ονομάτων κτίζει ένα thread safe repository στο οποίο καταχωρείται ένα μοναδικό byte για αναγνωριστικό του κάθε τύπου και πληροφορίες για την χρήση του τύπου.

Στο τέλος της καταχώρισης και της κατασκευής αλγόριθμου αντιστοίχισης αντικειμένων και λογικής, ο χρήστης μπορεί να καταχωρίσει εκτελέσημο κώδικα υπό μορφή function ο οποίος εκτελείται ασύγχρονα κατά την παραλαβή κάποιου μηνύματος-κλάσης, τύπου μηνύματος, ή συμβάντος συγκεκριμένης σύνδεσης.

Networking Module Οι ρυθμίσεις του network manager κτίζονται από αφαιρέσεις. Η χρήση πρωτοκόλλων και τεχνικών γίνεται με άγνοια της υλοποίησης και λεπτομερειών. Η σχεδίαση επιτρέπει την αποστολή μηνύματος ανεξαρτήτου πρωτοκόλου, τρόπο αποστολής και παραλήπτη. Ο network manager περιλαμβάνει τεχνικές function lifting για αποστολή μηνυμάτων. Ο χρήστης μπορεί να ρυθμίσει διάφορους τρόπους αποστολής ανάλογα με το περιβάλλον χρήση, και να τους κρύψει πίσω από functions.



Διάγραμμα 2.3: Network Packages Module



Διάγραμμα 2.4: Networking Module

API Συντομη επισκόπηση του API.

- Κλάσεις που προορίζονται για ανταλλαγή μηνυμάτων

```
[GinetPackage]
[PackageSerializer(typeof(MyCustomSerializer))] //optional
public class MyPackage
{
    public string Message { get; set; }
}
```

- Ρύθμιση server

```
var server = new NetworkServer("MyServer",
        builder =>
        {
            //via reflection
            builder.RegisterPackages(Assembly.Load("Package
        //or manually
            builder.RegisterPackage<MyPackage>();
        }
        cfg =>
        {
            cfg.Port = 1234;
            cfg.ConnectionTimeout = 5.0f;
            //Additional configuration
        }
        out: new ActionAppender(Console.WriteLine) //optional,
```

- Καταγραφή κινήσεων

```
server.IncomingMessageHandler.LogTraffic();
```

- Απάντηση κατά την παραλαβή μηνύματος

```
server.Broadcast<ChatMessage>((msg, im, om) =>
{
```

```
            server.Out.Info($"Received␣{msg.Message}");
            server.SendToAllExcept(om, im.SenderConnection, NetDeliv
    },
    packageTransformer: msg =>
    msg.Message += "this␣is␣broadcasted");


    server.BroadcastExceptSender<ChatMessage>((sender, msg) =>
    {
    server.Out.Info($"Broadcasting␣{msg.Message}.␣Received␣from:␣{s
    });
```

- Κώδικας ο οποίος εκτελείται κατά την παραλαβή συγκεκριμένου τύπου μηνύματος

```
server.IncomingMessageHandler.OnMessage(NetIncomingMessageType.
{
        //Configure connection approval
        var parsedMsg = server.ReadAs<ConnectionApprovalMessage
        if(parsedMsg.Password == "my␣secret␣and␣encrypted␣passw
        {
                incomingMsg.SenderConnection.Approve();
                incomingMsg.SenderConnection.Tag = parsedMsg.Se
        }
        else
        {
                incomingMsg.SenderConnection.Deny();
        }
});
```

- Εκτελέσιμος κώδικας κατά την παραλαβή συγκεκριμένου μηνύματος-κλάσης

```
server.IncomingMessageHandler
.OnPackage<MyPackage>((msg, sender) =>
```

```
Console.WriteLine($"Received␣{msg.Message}␣from␣{sender
```

- Αποστολή μηνύματος-κλάσης

```
server.Send(new MyPackage { Message = "Hello" },
        (outgoingMessage, peer) =>
                peer.SendMessage(outgoingMessage, NetDeliveryMet
```

- Lift αποστολής μηνύματος

```
var packageSender = client.LiftSender((msg, peer) =>
        peer.SendMessage(msg, NetDeliveryMethod.ReliableOrdered

packageSender.Send(new MyPackage { Message = "Hello" });
```

- Η ρύθμιση και το API του client είναι πανομοιότυπο με του server. Στη θέση του NetworkServer χρησιμοποιείται ο NetworkClient όπου και τα δύο υλοποιούν τον NetworkManager<TPeer> where TPeer:NetPeer

Παράρτημα 3

Game Host

3.1 Αρχιτεκτονική

Παράρτημα 4

Περίληψη

Γλωσσάρι

Το γλωσσάρι μπορεί να είναι χρήσιμο αν χρησιμοποιείτε πολλά ακρώνυμα και συντομογραφίες. Για παράδειγμα

TCP Transmission Control Protocol