National College of Ireland

Scope:

The main objective of this project is the development of a computer system for a Minecraft Game System.

Abstract:

The goal of this project is to outline the requirements of creating a version of Minecraft where Players can purchase and install numerous extension packs to enhance the gaming experience. These optional installs will enable users to create worlds that are of a deeper level of customisation, in part thanks to the greater library of materials available to Players to mine within their Minecraft world.

Bearing in mind that Minecraft is in large part a Multi-player experience, any extension packs that the user chooses to install must be held on a centrally accessible server in order for custom textures and materials deriving from the extension packs to be visible to other Players visiting a world.

In order to achieve this, we will implement a UML concept around a slave Minecraft Server where each paid user of Minecraft is granted a small read/write area of the server to store Minecraft extension packs and worlds, which utilizes Actor model concepts to outline its functionality.

The master Game Server implements the running of the Minecraft game.

Overview / Assumptions:

This Use Case enables the Player to install free and 'paid-for' extension pack(s) which are stored on the dedicated Minecraft server. As part of the Minecraft purchase price, Players are provided with a 'writable' area of the server where they can manage the Worlds they create as well as the extension packs they wish to use to enlarge the scope of play available to them.

Extension packs are used to enhance the gaming experience by providing additional building materials and player related objects into the Game, enabling the player to benefit from greater building opportunities as well as improved defensive abilities, in certain instances.

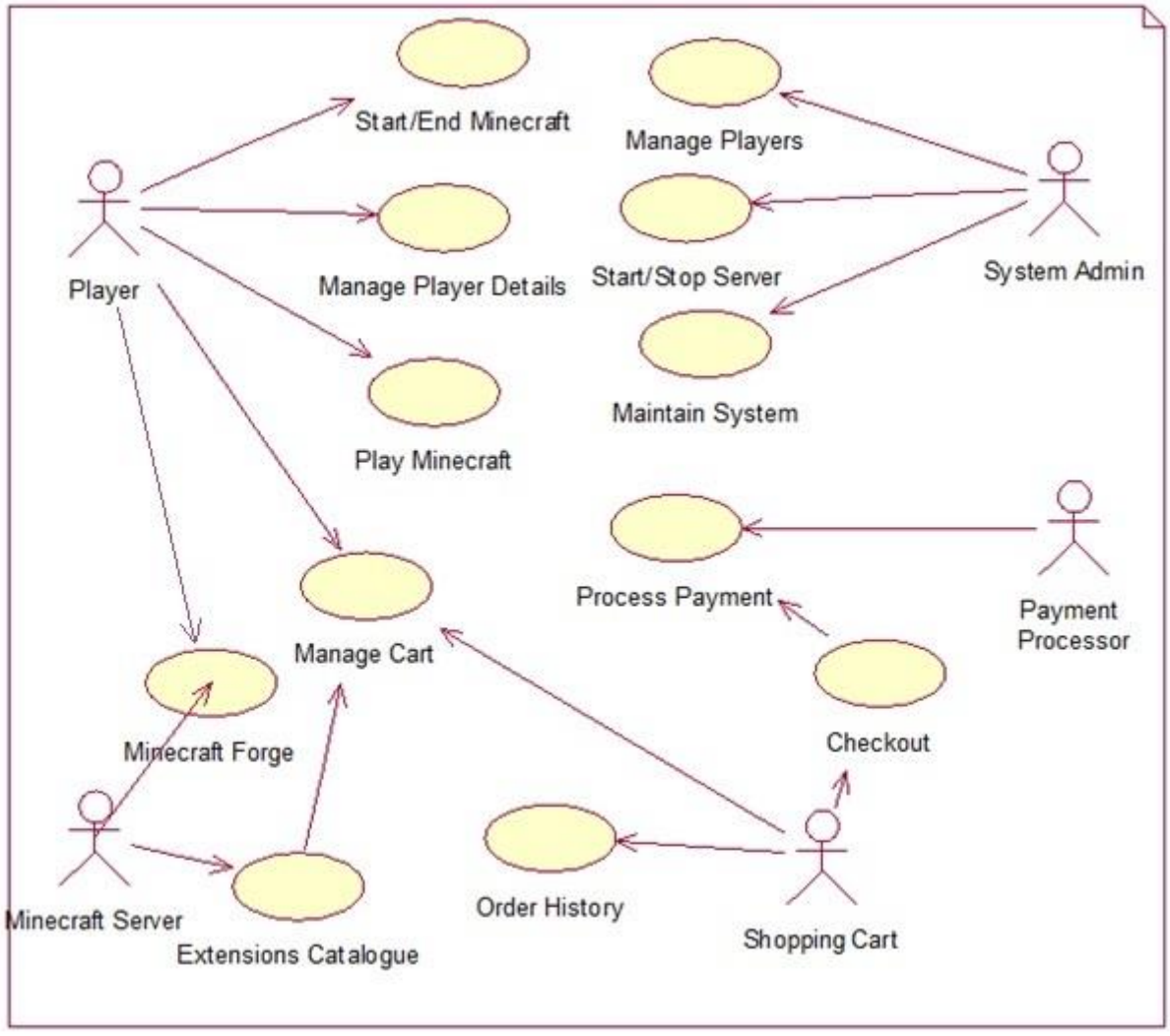
Installation of the extension pack modifies the material inventory and enables the use of a greater range of objects in the game, for both main players as well as visitors to the world.

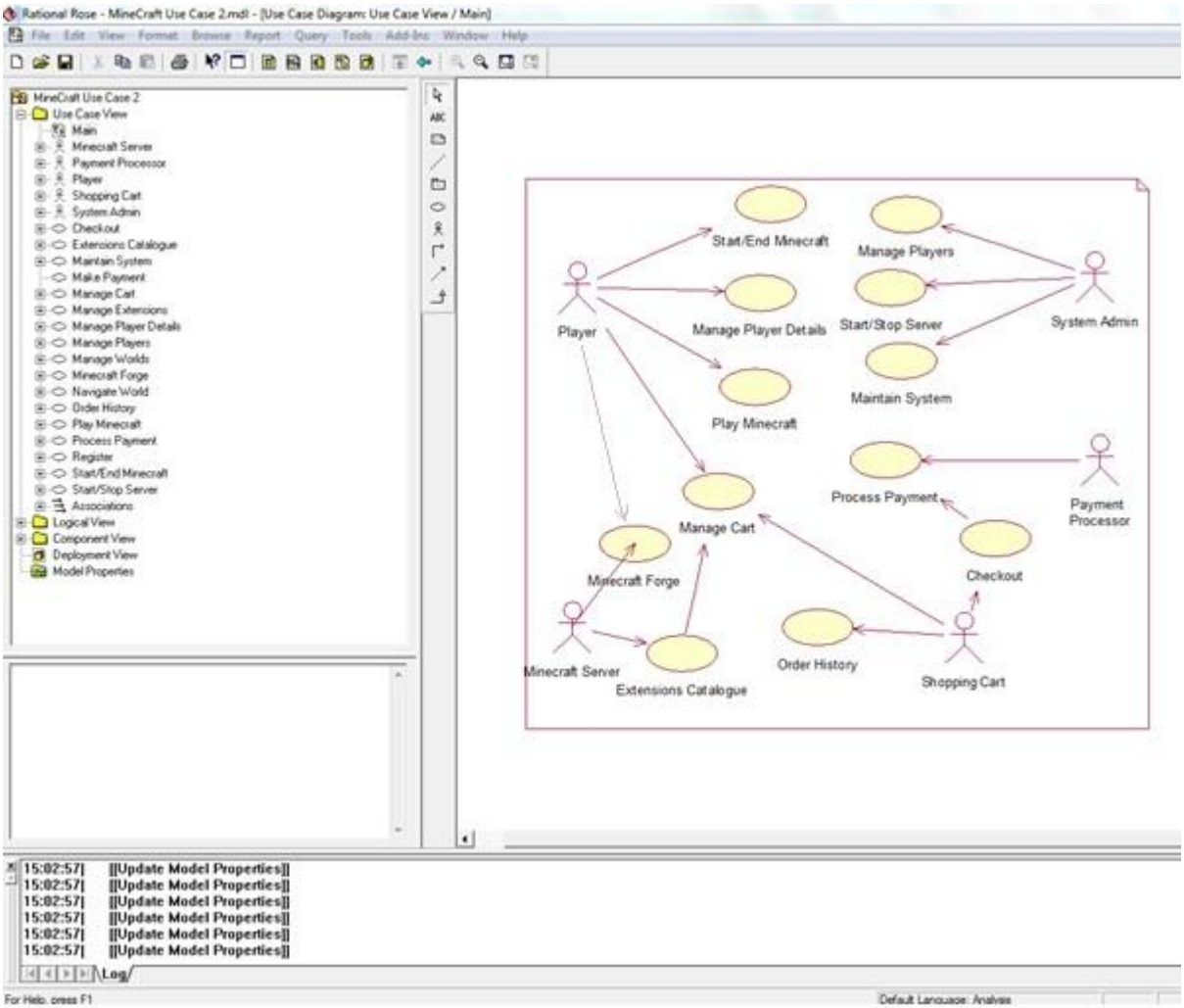
The extension packs are managed on the Minecraft server using the Minecraft Forge API. This enables other Minecraft players invited to interact in the world to view and use extension pack items without having unnecessary files installed on the client-side machine.

The Actors

1. Player – Purchaser of Minecraft.
2. Payment Processor – Credit Card payment processor for extension pack purchases.
3. Minecraft Server – Allows Players to upload Minecraft extension packs to Player Accounts.
4. System Admin – Manages and maintains the Minecraft system.
5. Shopping Cart – Ecommerce system controlling the catalogue of available Minecraft extension packs.

Use Case Diagram




Rational Rose - MineCraft Use Case 2.mdl - [Use Case Diagram: Use Case View / Main]
File Edit View Format Browse Report Query Tools Add-Ins Window Help
MineCraft Use Case 2
Use Case View
Main
Minecraft Server
Payment Processor
Player
Shopping Cart
System Admin
Checkout
Extensions Catalogue
Maintain System
Make Payment
Manage Cart
Manage Extensions
Manage Player Details
Manage Players
Manage Worlds
Minecraft Forge
Navigate World
Order History
Play Minecraft
Process Payment
Register
Start/End Minecraft
Start/Stop Server
Associations
Logical View
Component View
Deployment View
Model Properties
Start/End Minecraft
Manage Players
Player
Manage Player Details
Start/Stop Server
System Admin
Maintain System
Play Minecraft
Process Payment
Payment Processor
Manage Cart
Checkout
Minecraft Forge
Minecraft Server
Order History
Shopping Cart
Extensions Catalogue
15:02:57| [[Update Model Properties]]
15:02:57| [[Update Model Properties]]
15:02:57| [[Update Model Properties]]
15:02:57| [[Update Model Properties]]
15:02:57| [[Update Model Properties]]
15:02:57| [[Update Model Properties]]
Log
For Help, press F1
Default Language: Analysis

Use Case Model Description

Use Case Name: Manage Extension

1. Purpose:

The Manage Extensions use case, enables fully-registered Minecraft players to install extension packs (EP) via Minecraft forge, delete unwanted extensions from the Players world and utilize installed extension packs to enhance the Minecraft experience.

2. Actors:

Player – Installs, deletes, updates and utilizes Minecraft Extensions. Minecraft Server – Files are installed from the extension pack via Minecraft Forge System Admin – Manages server side files installed on the Minecraft server using Minecraft Forge API.

3. Description:

Manage Extension: This Use Case enables the Player Actor to install, use and delete Minecraft EP from their world. Installation of the EP modifies the Players inventory to include the models contained within the EP. The models are added to the Players version of Minecraft using Minecraft Forge. The models in the EP are installed on a designated Read/Write area of the Minecraft Server, in order to facilitate sharing of the models in the world with other Minecraft Players invited to the Players world. Deletion of an EP from the user-controlled area of the Minecraft Server can be initiated wither by the user or by the System Admin (SA). The SA will initiate the deletion of an EP, in the event that a file associated with a particular EP pose a security risk or is suspected of conflicting with the integrity of the system.

4. Flow Description

1) Pre-Conditions

In order to satisfy the pre-condition, the player must have version 1.7.9 (or greater) of Minecraft installed. Player must also have Minecraft forge installed in order to access the user controlled Read/Write area of the Minecraft Server for the Players User Account.

2) Activation

The UC starts when the Player has successfully logged in to their Minecraft account and initiates Minecraft Forge.

3) Main Flow

- Player purchases extension pack. - Player downloads extension pack from Minecraft server - Player enters their correct login details. - Player installs the extension pack on a Minecraft server using the Minecraft forge software to provide a user-friendly portal to the Minecraft server. - Player initiates the Minecraft executable to continue constructing a Minecraft world. - The Players in-world texture library is now updated extension pack items available to be utilised. - Player can now construct and access new textures and interact with the extension pack peripherals. - As the Minecraft extension pack contents are stored centrally on the Minecraft server, items from the extension pack are accessible by any visitor to the Minecraft world. - Use case was successful.

4) Alternative Flow(s)

If the Player has not purchased any extension packs, the Player is shown a message that there are no extension packs to manage. The player is then redirected to the Main screen to prompt them to start their Minecraft game. In step 2, if login details supplied by Player cannot be validated the use case ends with a failure condition. If in step 4, the Player has not installed the extension pack to the appropriate client-side directory the use case ends with a failure condition.

5) Exceptional Flow(s)

If the Minecraft server is offline, the Player is informed that the system is temporarily unavailable and is requested to revisit at a later time.

6) Termination

The UC is terminated upon successful installation of a Minecraft Extension Pack.

7) Post-Conditions

Represents the goals of the UC, if satisfied then UC is valid & successfully completed…

Successful Condition The user can use, interact, mine and build with the extended texture library. All contents of the extension pack are available to users in the Single-Player and Multi-Player environments. **Failure Condition** The logs have been updated accordingly.

5. Special Requirements

Provide a detailed description of the non-functional requirements that are scoped to the feature.

None

6. Interfaces

- Minecraft Login – to verify user details - Minecraft Server interface – to list any extension packs available to Delete / Update / Install - Minecraft Forge – Interface to unpack / install new extension packs.

7. System Characteristics / Performance

Extension pack should take no longer than 60 seconds to unpack install.

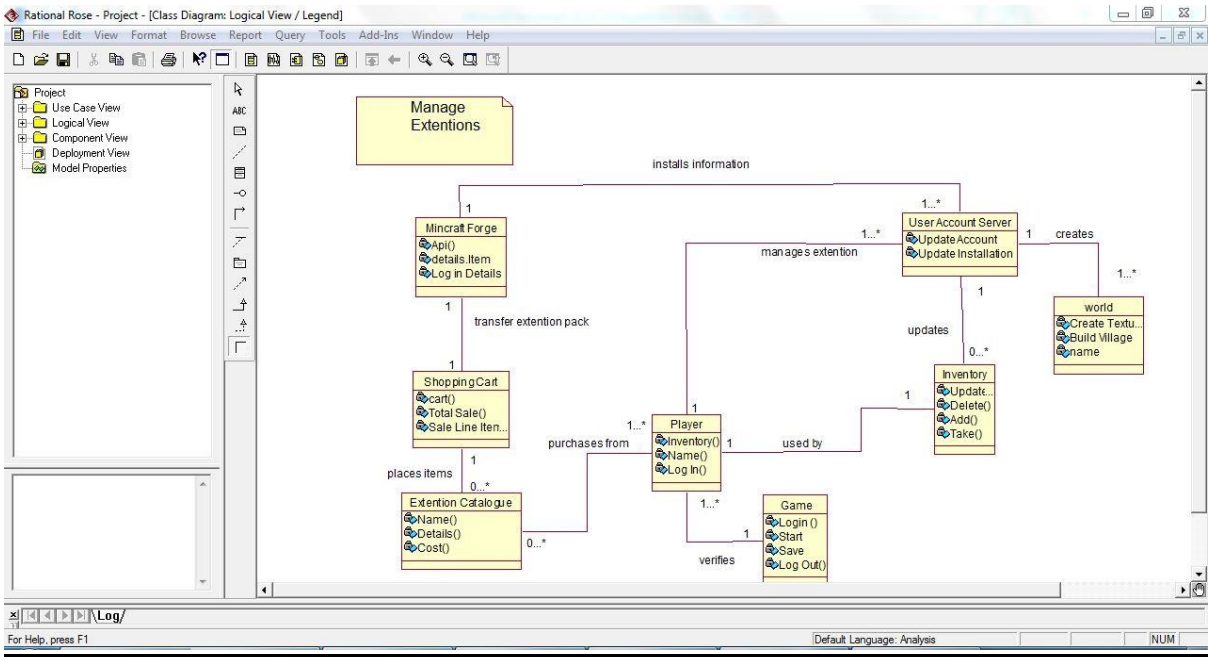
8. Implementation Requirements Software / Hardware / Documentation

None

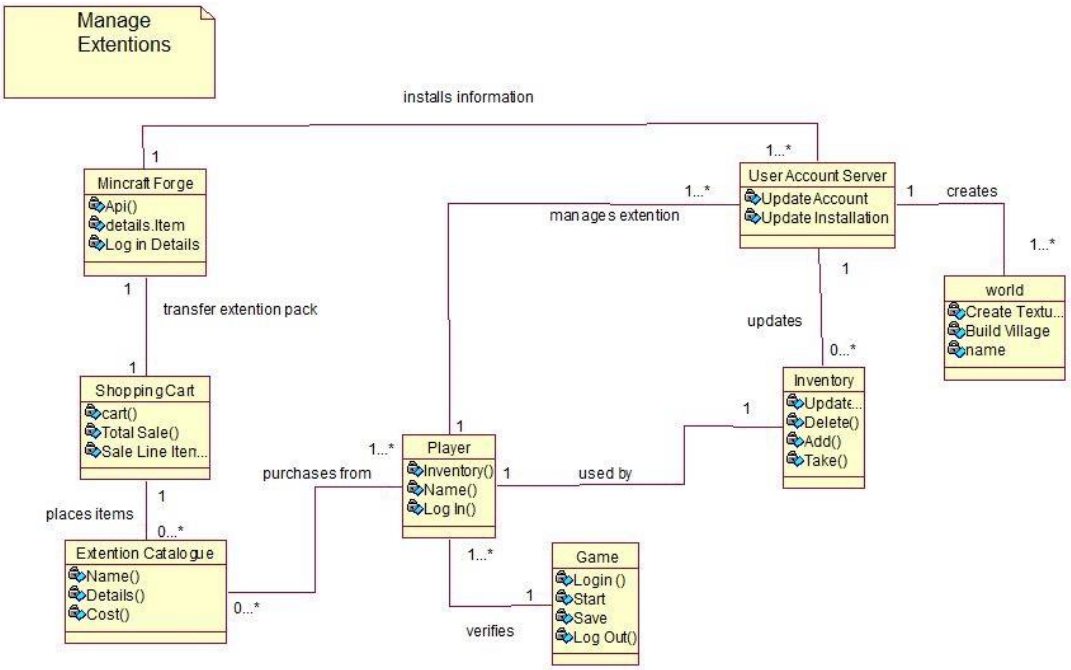
9. Technical Specifications

	Minimum Requirements: CPU: Intel Pentium D or AMD Athlon 64 (K8) 2.6 GHz RAM: 2GB GPU (Integrated): Intel HD Graphics or AMD (formerly ATI) Radeon HD Graphics with OpenGL 2.1 GPU (Discrete): Nvidia GeForce 9600 GT or AMD Radeon HD 2400 with OpenGL 3.1 HDD: At least 200MB for Game Core and Other Files Java 6 Release 45 Recommended Requirements: CPU: Intel Core i3 or AMD Athlon II (K10) 2.8 GHz RAM: 4GB GPU: GeForce 2xx Series or AMD Radeon HD 5xxx Series (Excluding Integrated Chipsets) with OpenGL 3.3 HDD: 1GB Latest release of Java 7 from java.com

Conceptual Class Diagram



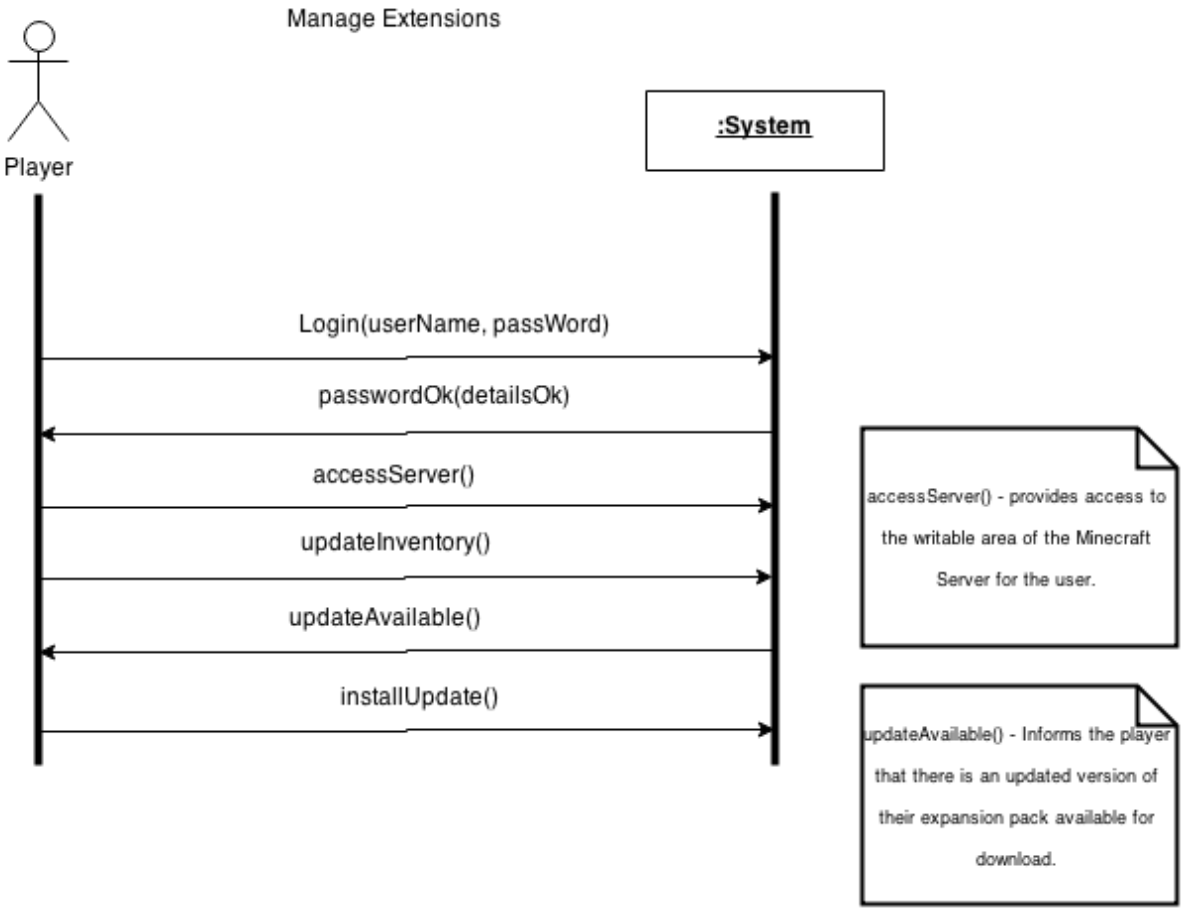
These are the same diagram, showing use of Rational Rose.



Glossary

Term	Category	Comments
Minecraft Forge	class	Enables the Player to upload the content of the extension pack to their account on the Minecraft server using the Minecraft API
Installs Information	Association	Connects Minecraft Forge with the Minecraft server and describes the uploading of Extension pack information and materials to the user account on the Minecraft server.
detailsItem	attribute of Minecraft Forge	Contains relevant information as to the specific contents (materials and blocks) associated with an extension pack.
totalSale	attribute of Shopping Cart	Provides the user with a sales total at checkout in the event of single or multiple purchases.
Inventory	class	Inventory is updated by the action of the player installing new extension packs onto the Minecraft server.
UserAccount Server	class	Represents the area of the Minecraft server that is controlled by the user. The user can install / delete / update extension packs stored here.

System Sequence Diagram

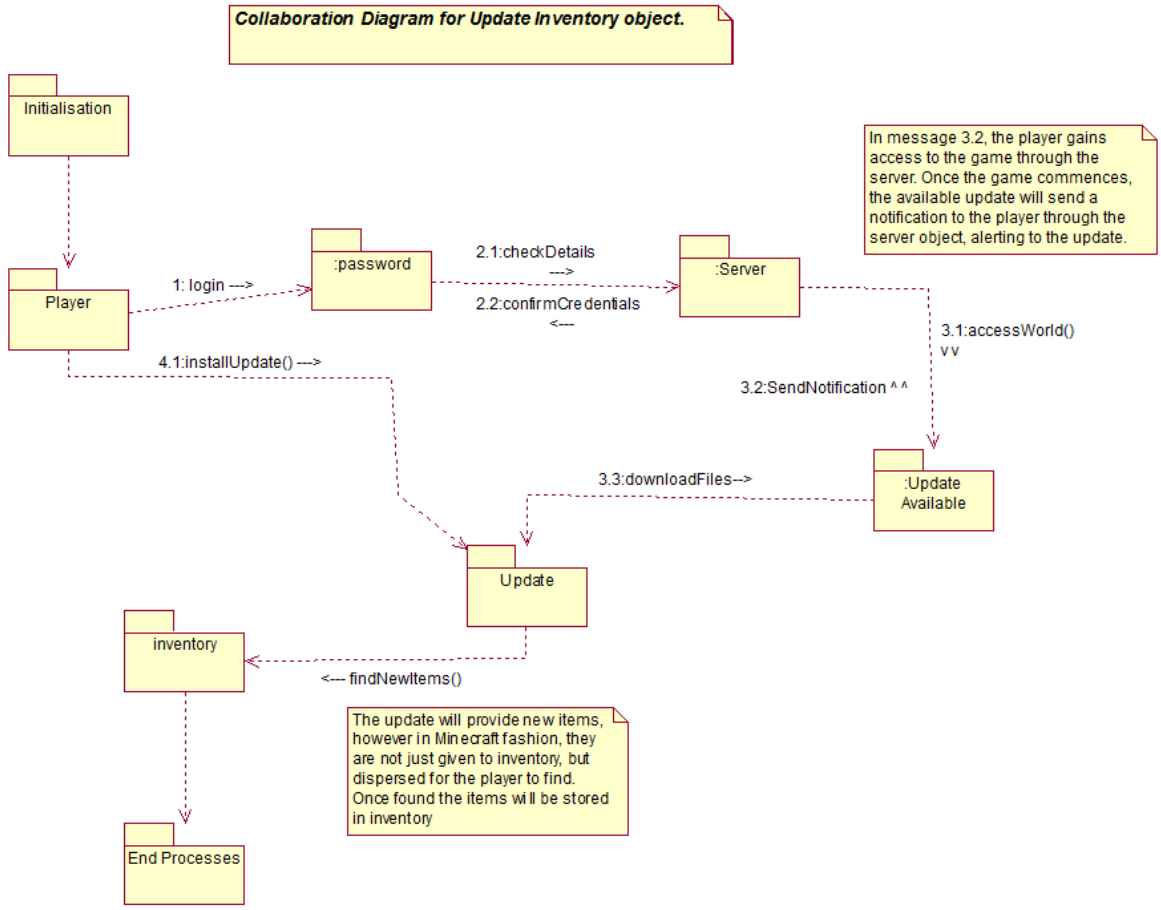


Contract

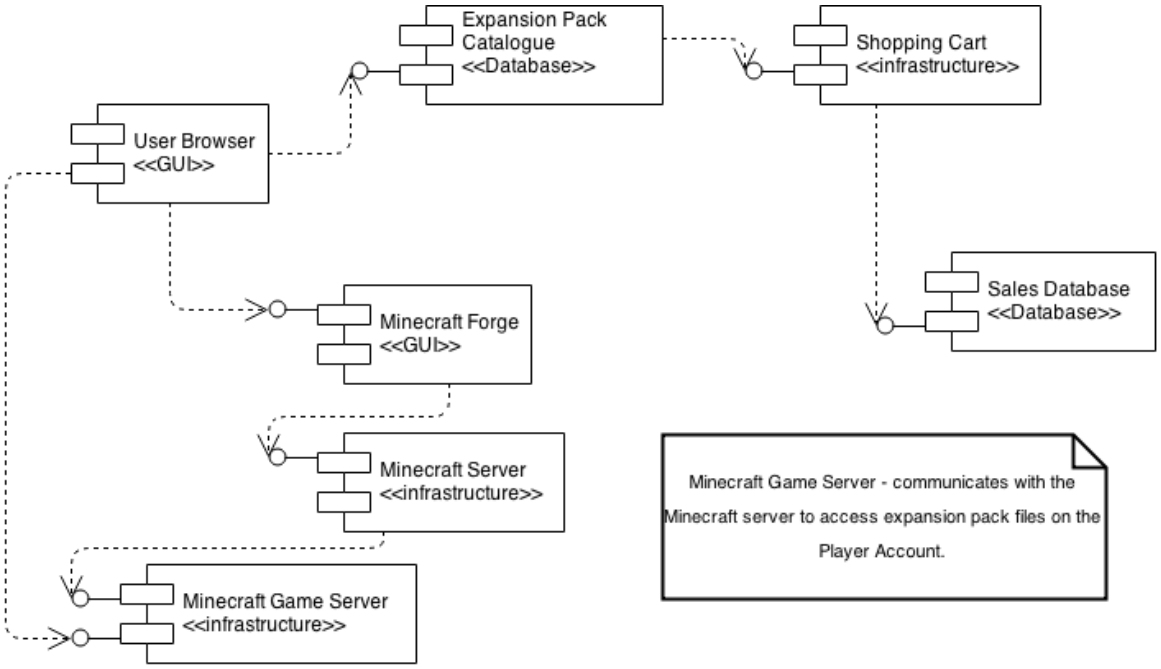
Contract for Update Inventory

Name:	**Update Inventory.**
Responsibilities:	Include any new items to storage and remove any items no longer held.
Type:	System.
Pre-Conditions:	Storage will be available for new items.
Post-Conditions:	If adding new items, available storage will be used, or blocks will be tossed.
	If deleting items, overall available storage space will increase.

Collaboration Diagram



Component Diagram



Deployment Diagram