Client-Server Article Application
Analysis and Design Document

Student: Hunor Barabas
Group: 30432

Table of Contents

1. Requirements Analysis

1.1 Assignment Specification

We are required to make an application for the management of a News Agency. The application has two types of users, namely Writers and Readers. The Latter can only read through existing articles, while the writers can write new articles. The articles are updated in real-time (thanks to the Observer pattern used).

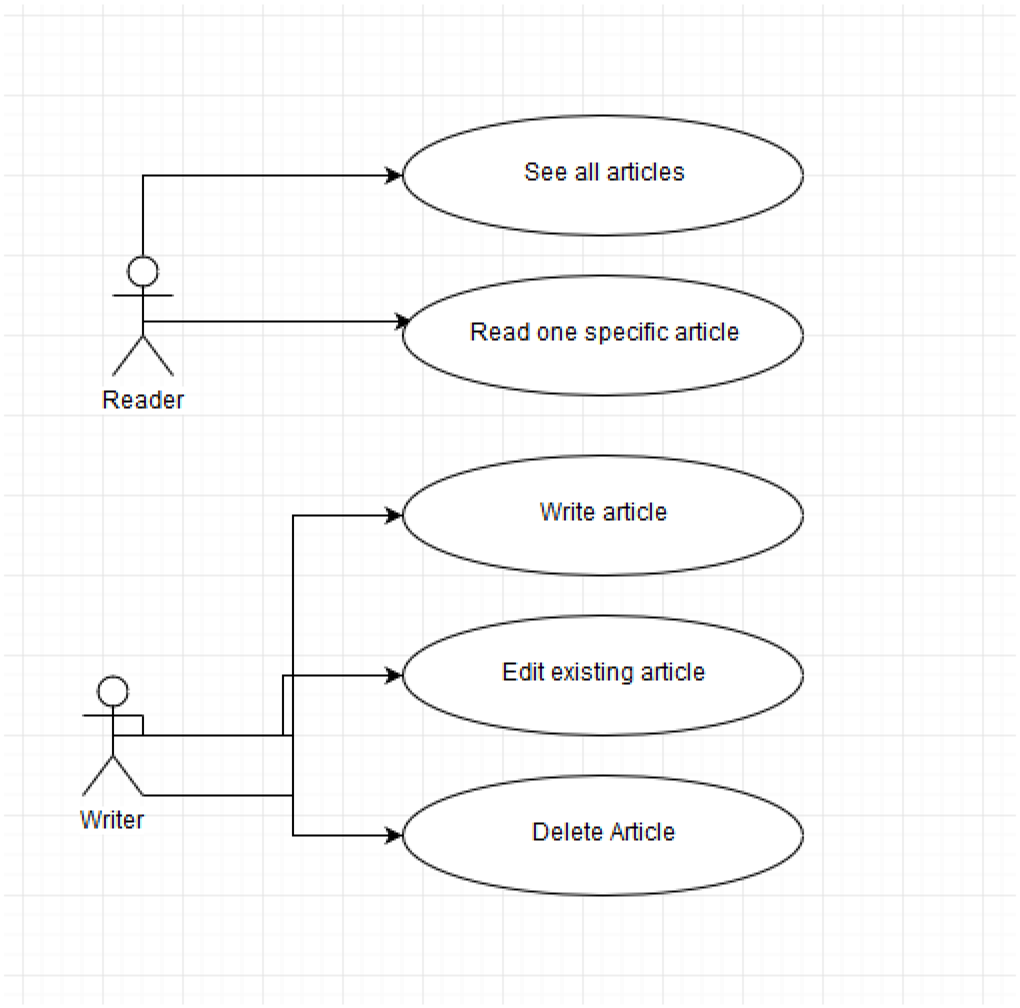
1.2 Functional Requirements

- Readers can see all articles without having to log in
- Readers can select one article to read
- Writers can log in and create/edit articles
- The application has to be client-server, supporting multiple clients at the same time

1.3 Non-functional Requirements

- Security – Personal information should be protected. Clients who are not eligible to write/modify should not have the option to do so.
- Availability – The system should be functional at all time, meaning that the server should be able to listen to requests 24/7
- Capacity – The server should not break under high traffic.
- Performance – The system should not take too long to respond to requests
- Accessibility – All the functions of the system should be clear and easy to access.

2. Use-Case Model



Use case: Read article
Level: user
Primary actor: reader
Description: The user starts the client application. A panel will open in which he needs to select his role (in this case, Reader). After that, another panel will open showing all available articles. He can then proceed to select one and press a button to open it up for reading.
Scenario:

1. User opens up the application
2. A panel will be presented with the 2 different roles
3. User chooses Reader. Another panel opens up
4. User selects an article from the existing ones, clicks on ‘Read Article’
5. Another panel opens up with the article’s content.

Extensions: In case the application can’t show the article, an error message is displayed

Use case: Write article
Level: user
Primary actor: writer
Description: The user starts the client application. A panel will open in which he needs to select his role (in this case, Writer). He then needs to enter his credentials. If they are correct, he will be taken to another panel, on which he can select the option to write an article
Scenario:

1. User opens up the application
2. A panel will be presented with the 2 different roles
3. User chooses Writer. Another panel opens up
4. User selects the option to Write Article
5. Another panel opens up where he can write freely, then save the article.

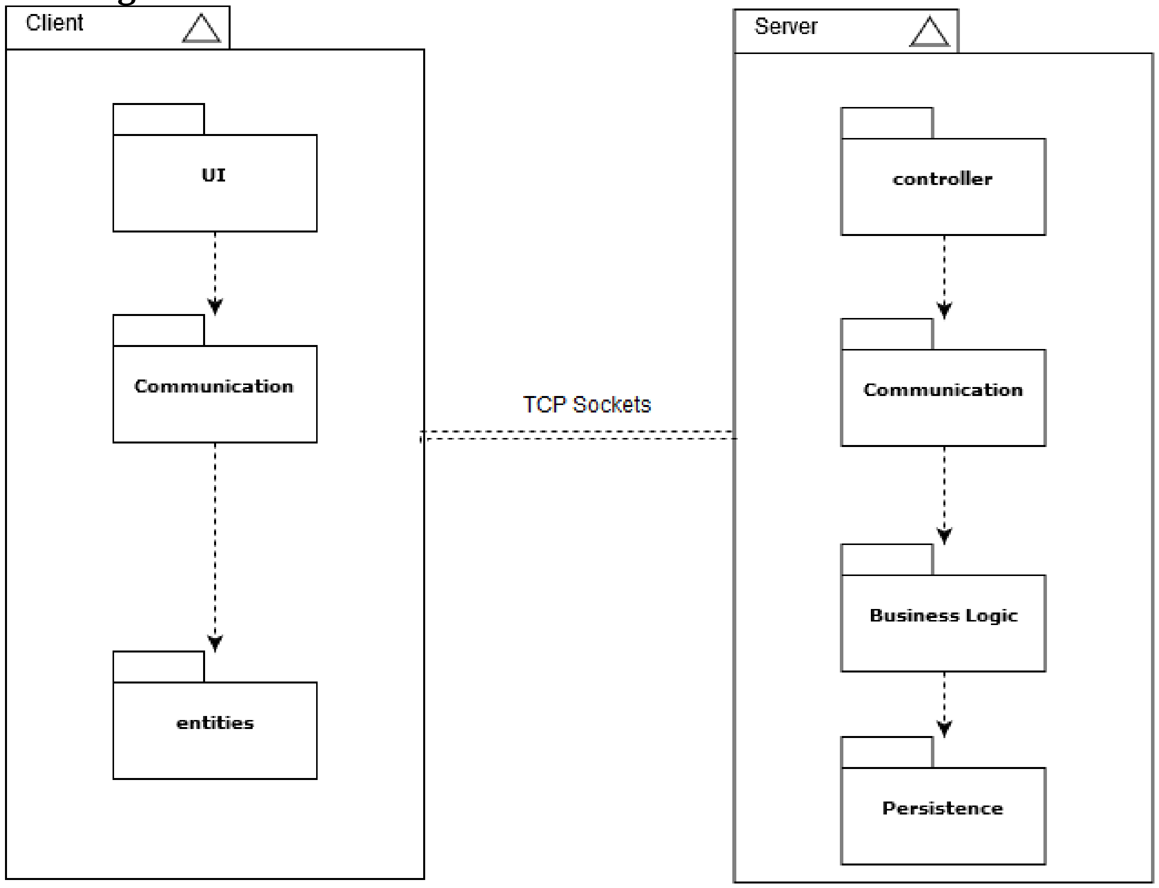
Extensions: In case the credentials are incorrect, an error message is displayed

3. System Architectural Design

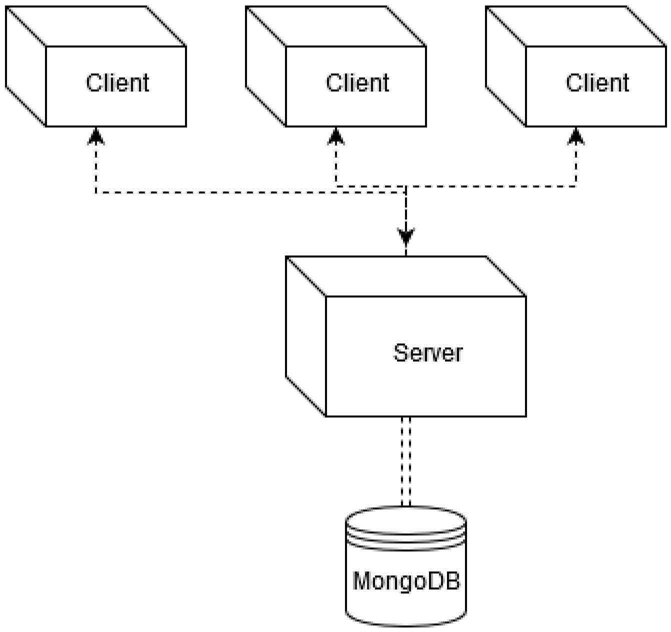
3.1 Architectural Pattern Description

The application is based on a client-server architecture which can serve multiple clients at one time. In addition, a layered architecture is used in both the client and the server, to organize both parts. The client and server communicate through messages which have a command name and a body. They send these messages through sockets, and they both have an interpreter class to translate the contents of each message into actual functionality.

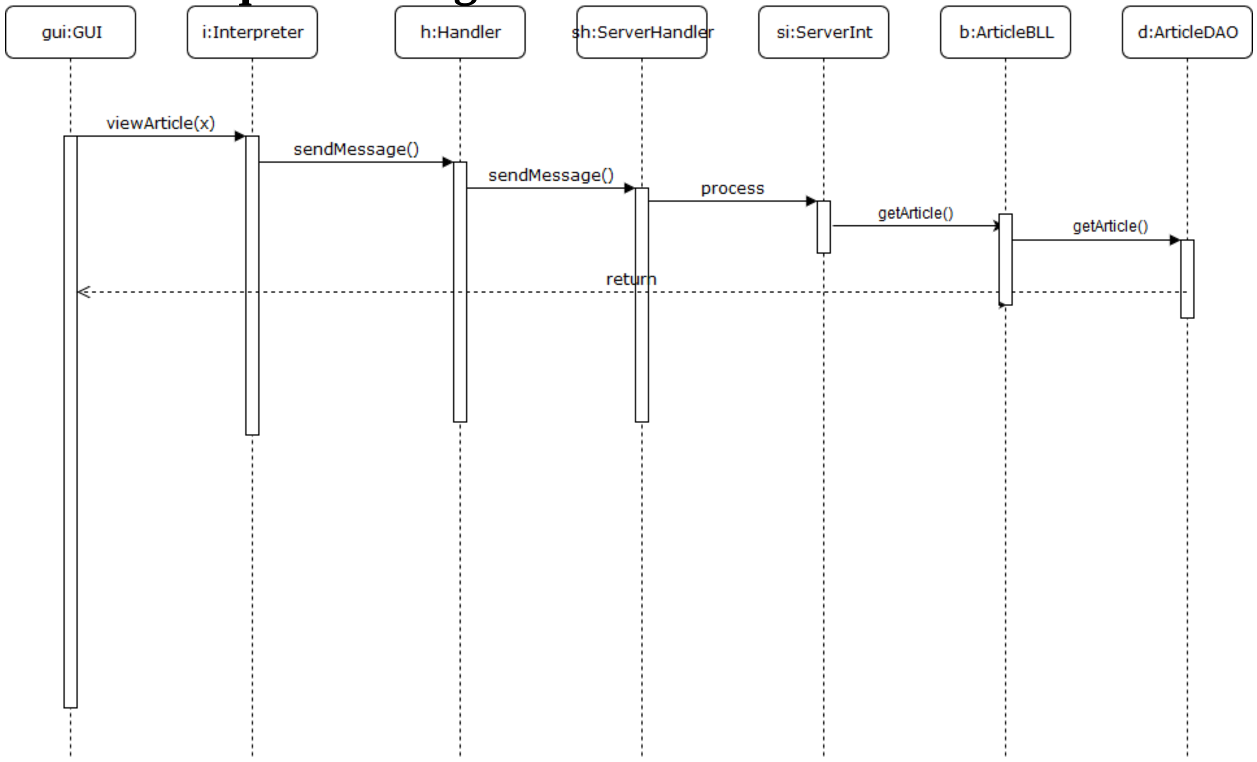
3.2 Diagrams



Package Diagram



4. UML Sequence Diagrams

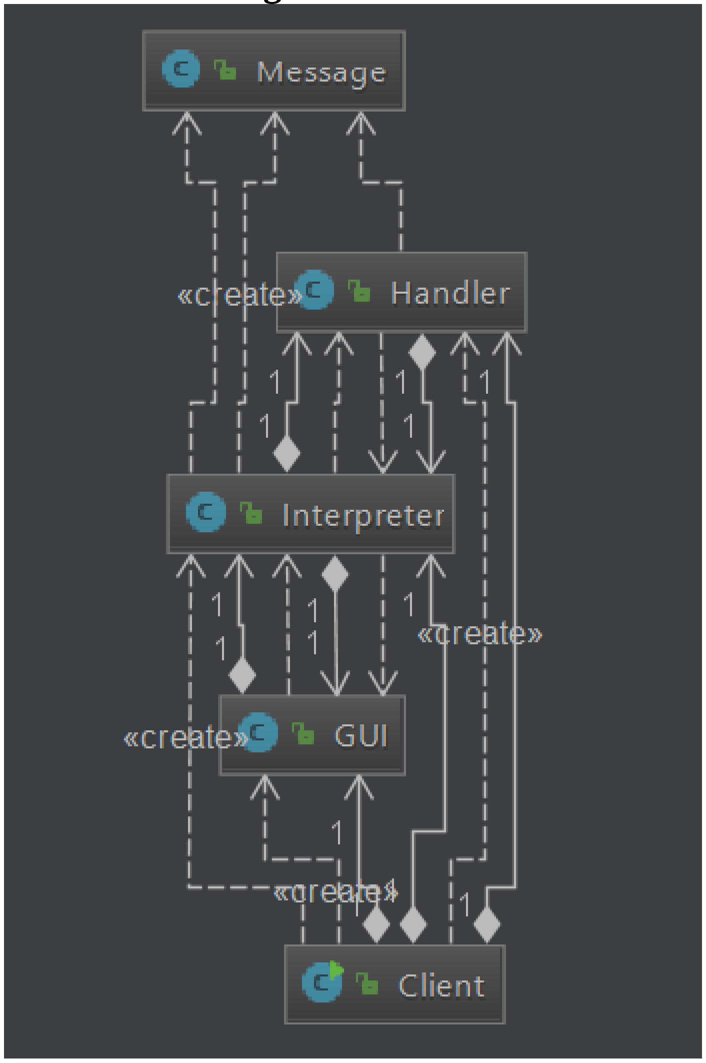


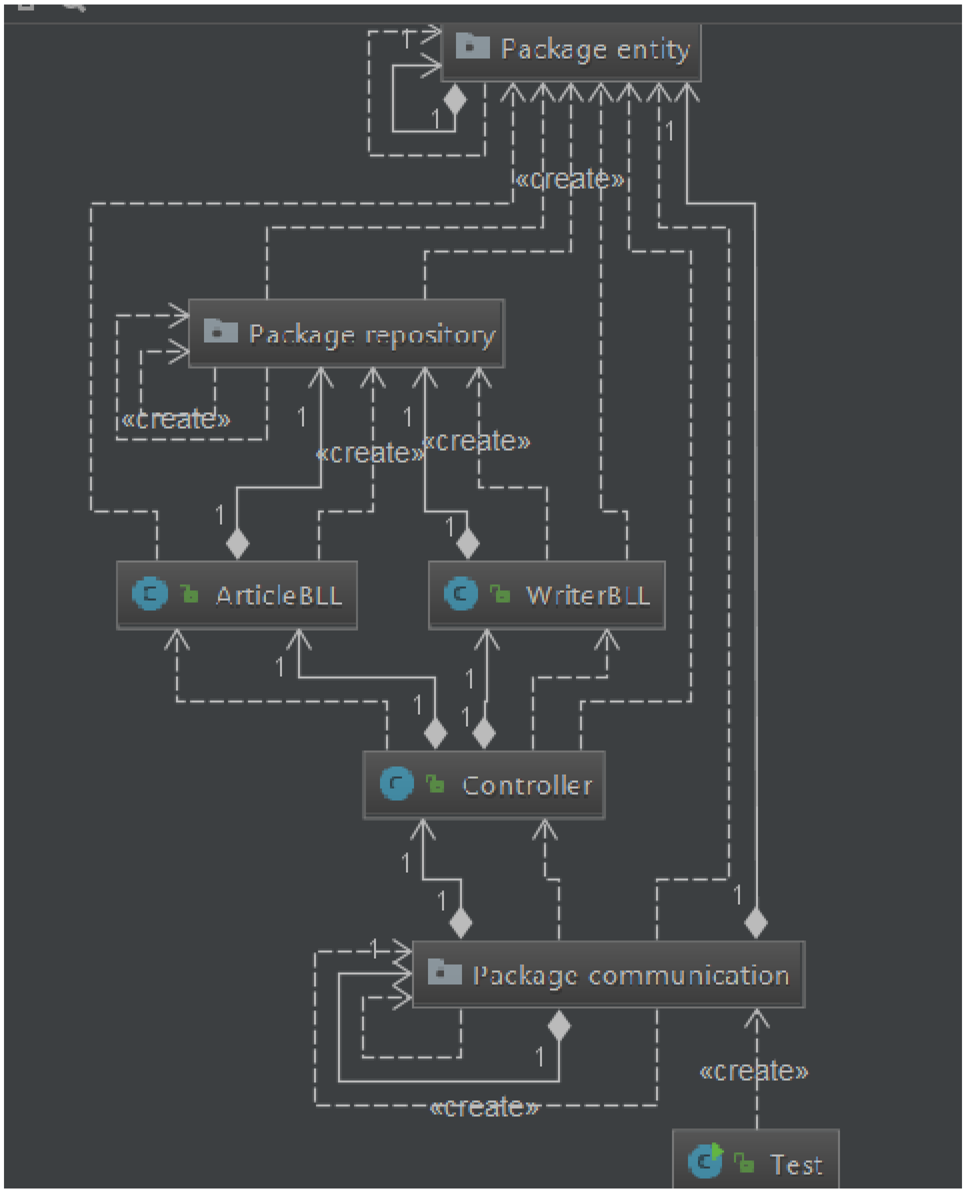
5. Class Design

5.1 Design Patterns Description

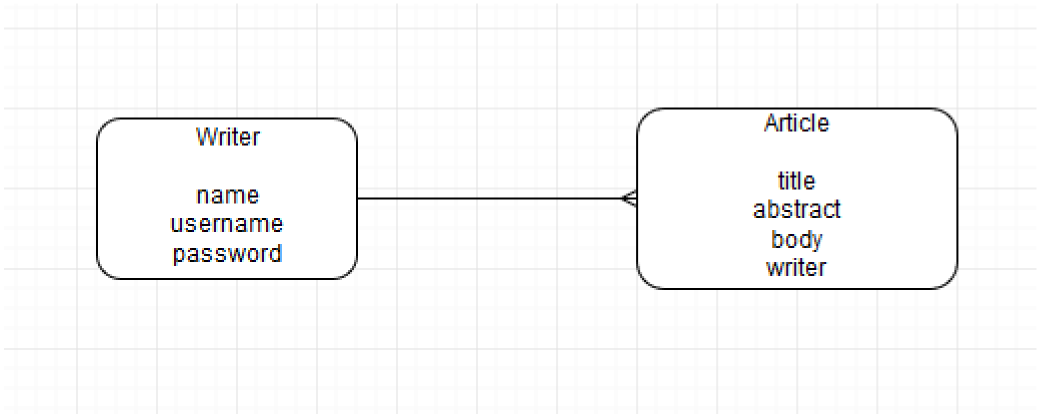
Observer Pattern – The interface which the client sees should be updated in real-time, in case an article is modified in some way while they are reading them. To solve this, I have implemented the Observer pattern, which has two parts: an Observable object, which has a list of objects that “subscribe” to it, and the Observers, who subscribe. Whenever the Observable’s data changes, the subscribers are notified. This is done by going through the list of all observers and calling their update() method.

5.2 UML Class Diagram




Package entity
Package repository
ArticleBLL
WriterBLL
Controller
Package communication
Test
«create»
1

6. Data Model


Writer
name
username
password
Article
title
abstract
body
writer

7. System Testing

8. Bibliography