|  |  |
| --- | --- |
| **Gerb-BMSTU_01** | **Министерство науки и высшего образования Российской Федерации**  Калужский филиал  федерального государственного бюджетного  образовательного учреждения высшего образования  ***«Московский государственный технический университет имени Н.Э. Баумана (национальный исследовательский университет)»***  ***(КФ МГТУ им. Н.Э. Баумана)*** |

**ФАКУЛЬТЕТ** \_ ***ИУК «Информатика и управление»\_\_\_\_\_\_\_\_\_***

**КАФЕДРА** \_\_ ***ИУК5 «Системы обработки информации»***

**РАСЧЕТНО-ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

**к курсовой работе на тему:**

***Разработка многопользовательского игрового приложения “Pong Online”.***

по дисциплине ***Сетевые технологии в автоматизированных системах обработки информации и управления***

Студент гр. ИУК5-62Б \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ (Хромов А.Е. )

(подпись) (Ф.И.О.)

Руководитель \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ ( )

(подпись) (Ф.И.О.)

Оценка руководителя \_\_\_\_\_ баллов \_\_\_\_\_\_\_\_\_\_\_

30-50 (дата)

Оценка защиты \_\_\_\_\_ баллов \_\_\_\_\_\_\_\_\_\_\_

30-50 (дата)

Оценка проекта \_\_\_\_\_ баллов \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(оценка по пятибалльной шкале)

Комиссия: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ (\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_)

(подпись) (Ф.И.О.)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ (\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_)

(подпись) (Ф.И.О.)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ (\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_)

(подпись) (Ф.И.О.)

Калуга, 2022

Калужский филиал   
федерального государственного бюджетного образовательного учреждения высшего образования

***«Московский государственный технический университет имени Н.Э. Баумана (национальный исследовательский университет)»   
(КФ МГТУ им. Н.Э. Баумана)***

УТВЕРЖДАЮ

Заведующий кафедрой **\_\_ИУК5\_\_\_**

\_\_\_\_\_\_\_\_\_\_\_(Е.В. Вершинин)

«\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_20\_\_\_г.

**ЗАДАНИЕ**

**на выполнение курсовой работы**

по дисциплине ***Сетевые технологии в автоматизированных системах обработки информации и управления***

Студент Хромов А.Е. ИУК5-62Б \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(фамилия, инициалы, индекс группы)

Руководитель Вершинин Е.В.

(фамилия, инициалы)

График выполнения проекта: 25% к\_4\_нед., 50% к\_7\_нед., 75% к\_10\_нед., 100% к\_14\_нед.

***1. Тема курсового проекта***

***Разработка многопользовательского игрового приложения “Pong Online”.***

***2. Техническое задание***

*Разработать приложение с использованием методов программирования в вычислительной сети для многопользовательского игрового приложения “Pong Online” с использованием системы комнат для выбора оппонента из списка подключенных к серверу.*

***3. Оформление курсового проекта***

3.1. Расчетно-пояснительная записка на\_\_\_\_\_\_\_ листах формата А4.

3.2. Перечень графического материала КП (плакаты, схемы, чертежи и т.п.)\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Дата выдачи задания «\_\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_2022г.

Руководитель курсового проекта \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/

(подпись) (Ф.И.О.)

Задание получил\_\_\_\_\_\_\_\_\_\_\_\_\_\_/\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/ «\_\_\_\_\_»\_\_\_\_\_\_\_\_\_\_2022г.

(подпись) (Ф.И.О.)

Примечание:

Задание оформляется в двух экземплярах: один выдается студенту, второй хранится на кафедр

СОДЕРЖАНИЕ

[ВВЕДЕНИЕ.........................................................................................................................................4](#_Toc102868040)

[1. ТЕХНИЧЕСКОЕ ЗАДАНИЕ 5](#_Toc102868041)

[1.1. Наименование 5](#_Toc102868042)

[1.2. Основание для разработки 5](#_Toc102868043)

[1.3. Исполнитель 5](#_Toc102868044)

[1.4. Цель разработки 5](#_Toc102868045)

[1.5. Содержание работы: 6](#_Toc102868046)

[1.5.1. Задачи, подлежащие решению. 6](#_Toc102868047)

[1.5.2. Требования к архитектуре АСОИ. 6](#_Toc102868048)

[1.5.3. Требования к прикладным программам. 7](#_Toc102868049)

[1.5.4. Требования к входным/выходным данным 7](#_Toc102868050)

[1.5.5. Требования к временным характеристикам 7](#_Toc102868051)

[1.5.6. Требования к составу технических средств 7](#_Toc102868052)

[1.6. Этапы разработки 8](#_Toc102868053)

[1.7. Техническая документация, предъявляемая по окончании работы 8](#_Toc102868054)

[2. НАУЧНО-ИССЛЕДОВАТЕЛЬСКАЯ ЧАСТЬ 9](#_Toc102868055)

[2.1. Постановка задачи проектирования. 9](#_Toc102868056)

[2.2. Описание предметной области. 9](#_Toc102868057)

[2.3. Анализ аналогов и прототипов. 9](#_Toc102868058)

[2.4. Перечень задач, подлежащих решению в процессе разработки. 11](#_Toc102868059)

[2.5. Обоснование выбора инструментов и платформы для разработки. 12](#_Toc102868060)

[3. ПРОЕКТНАЯ ЧАСТЬ........... 14](#_Toc102868061)

[3.1. Разработка структуры приложения. 14](#_Toc102868062)

[3.2. Разработка алгоритмов обработки информации. 16](#_Toc102868063)

[3.3. Разработка архитектуры приложения. 17](#_Toc102868064)

[3.4. Разработка систем передачи информации. 20](#_Toc102868065)

[3.5. Реализация функционирующего приложения. 22](#_Toc102868066)

[3.6. Разработка интерфейса взаимодействия пользователя с системой. 23](#_Toc102868067)

[4. ПРОИЗВОДСТВЕННО-ТЕХНОЛОГИЧЕСКАЯ ЧАСТЬ 25](#_Toc102868068)

[4.1. Тестирование и отладка макета рабочей программы. 25](#_Toc102868069)

[4.2. Разработка руководства пользователя 26](#_Toc102868070)

[ЗАКЛЮЧЕНИЕ.................................................................................................................................27](#_Toc102868071)

[СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ 28](#_Toc102868072)

[ПРИЛОЖЕНИЕ А.............................................................................................................................29](#_Toc102868073)

# ВВЕДЕНИЕ

Вряд ли можно найти такого человека, который не пробовал играть в компьютерные игры. Этот процесс очень увлекательный и способствует развитию не только физических, но и умственных способностей. Интернет предложил новый вариант досуга в виде онлайн-игр, которые весьма популярными и продолжают неуклонно набирать все большую популярность.

Количество любителей онлайн-игр постоянно растет, поскольку сложно не оценить те перспективы и возможности, которые они дают:

* Основным преимуществом является доступность. Каждый человек, у которого есть доступ в интернет и специальное техническое решение, будь то компьютер или телефон, может воспользоваться предложениями многочисленных сервисов.
* Немаловажно, что онлайн-игры часто носят групповой характер. Таким способом люди знакомятся, находят друзей по интересам и т.д. Игроки могут находиться в разных точках мира и при этом активно взаимодействовать в процессе игры. Кроме того, есть игры, которые предусматривают возможность живого общения.
* Онлайн-игры примечательны тем, что они постоянно усовершенствуются и обновляются без непосредственного участия игрока, то есть ему всегда доступен последний вариант продукта.

Конечно, онлайн-игры могут быть полезными и обеспечивают современному человеку достойный досуг. В данном случае важно четко видеть грань между реальным и виртуальным миром, и контролировать время, отведенное на такое развлечение. [[1]](#лит1)

Следовательно разработка онлайн-игр в данный момент является очень актуальной, так как при хорошем подходе к идее игры и не менее хорошей ее реализации имеются высокие шансы на коммерческий успех продукта. Кроме того, реализация игры как товара сейчас не представляет никаких трудностей за счет удобных онлайн-площадок по продажам видеоигр.

# ТЕХНИЧЕСКОЕ ЗАДАНИЕ

## Наименование

Разработка многопользовательского игрового приложения “Pong Online”.

## Основание для разработки

Pong — одна из ранних аркадных видеоигр. Это теннисная спортивная игра с использованием простой двумерной графики, разработанная и выпущенная фирмой Atari в 1972 году. Pong называют первой в истории коммерчески успешной видеоигрой, а с её именем связывают появление индустрии интерактивных развлечений. Позже игра была выпущена для многих консолей. Разработанная многопользовательская версия Pong разрывает локальные границы игры (вдвоем за игровым автоматом / консолью), позволяя насладиться классической аркадной игрой в любом месте, где возможен выход в сеть. [[2]](#лит2)

## Исполнитель

Студент группы ИУК5-62Б Хромов А.Е.

## Цель разработки

Целью курсовой работы является формирование практических навыков по разработке и реализации программного приложения с использованием изученных средств и методов программирования, функционирующего в вычислительной сети, для автоматизации процессов обработки информации в конкретной области. Как правило, такое приложение имеет клиент-серверную архитектуру.

Задачи проектирования:

1. Овладение первичными навыками ведения научно-исследовательской, проектно-конструкторской и проектно-технологической деятельности, развитие творческих способностей индивидуально для каждого студента.
2. Подготовка к выполнению выпускной квалификационной работы.
3. Усвоение методов грамотного ведения, оформления и редактирования технической документации.

## Содержание работы:

### Задачи, подлежащие решению

1. исследование и анализ существующих многопользовательских игровых приложений, в том числе самого приложения Pong;
2. изучение основных средств и методов программирования приложения для функционирования в вычислительной сети.
3. реализация основной логики игры, методов взаимодействия между клиентами и сервером (создание комнат для подключения, отображение их всем игрокам, поддержание игровой сессии);
4. тестирование приложения;
5. подготовка расчетно-пояснительной записки и графических листов;
6. подготовка презентации и речи для защиты курсовой работы;
7. защита курсовой работы.

### Требования к архитектуре АСОИ

К архитектуре предъявляются следующие требования:

1. тип приложения – клиент-серверное;
2. клиентская часть представляется в виде оконного приложения, серверная часть в виде консольного приложения.
3. модульная архитектура программного обеспечения;

### Требования к прикладным программам

Для работы программного комплекса необходимы:

* Microsoft Windows 8/10 или Linux
* Java Virtual Machine (JVM)
* минимальный набор драйверов, обеспечивающих   
  работоспособность ПК.

### Требования к входным/выходным данным

Входные данные:

1. Имя игрока
2. Выбор или создание комнаты для игры (название комнаты)
3. Присоединение к существующей комнате
4. Нажатия клавиш для управления ракеткой.
5. Дополнительный действия во время игровой сессии (выход из игры, реванш и т.п.)

Выходные данные:

1. Отображение интерфейса выбора или создания комнат.
2. Отображение у оппонентов мяча.
3. Отображение движения ракетки игрока и его оппонента.

### Требования к временным характеристикам

Требования к временным характеристикам программы не предъявляются.

### Требования к составу технических средств

Для функционирования системы необходимы:

* процессор: минимальное требование - Pentium 2 266 МГц
* ОЗУ: 256 МБ и выше
* дисковое пространство: 512 МБ и выше
* видеокарта: поддержка Microsoft DirectX 9;
* монитор;
* клавиатура;
* мышь.

## Этапы разработки

1. исследование существующих многопользовательских игровых приложений и принцип их работы;
2. изучение принципов сетевого программирования и работы с сокетами;
3. реализация методов логики игрового процесса, и методов сетевого взаимодействия пользователей;
4. тестирование приложения;

## Техническая документация, предъявляемая по окончании работы

По окончанию работы предъявлена расчетно-пояснительная записка в состав которой входят:

* техническое задание;
* научно-исследовательская часть;
* проектная часть;
* производственно-технологическая часть.

Также должна быть предоставлена графическая часть работы, выполненная формате А1 на 2 листах, в которую входят:

* демонстрационные чертежи;
* алгоритмические схемы.

# НАУЧНО-ИССЛЕДОВАТЕЛЬСКАЯ ЧАСТЬ

## Постановка задачи проектирования.

Задачей курсовой работы является разработка многопользовательского игрового приложения, осуществляющей подключение игроков в комнаты (и создание комнат), реализующей сетевое взаимодействие между двумя игроками во время сессии и саму логику игры “Pong”. Разработать удобный пользовательский интерфейс для клиентской части, и консольное приложение для серверной.

## Описание предметной области.

Предметной областью является игровое многопользовательское приложение Pong, позволяющее игрокам выбирать оппонента из списка доступных комнат, создавать комнаты и играть один на один.

В процессе анализа можно выделить несколько сущностей:

* Игрок – представляет собой пользователя приложения. Игрок идентифицируется по никнейму, который он обязан ввести при подключении к серверу. У игрока имеется возможность создать комнату и выбрать ей название, количество очков для победы и, при желании, пароль. Игрок может присоединиться к уже существующей комнате.
* Комната – представляет собой отдельный сокет, принимающий максимум двух игроков, после подключения которых запускается игровая сессия.
* Игровая сессия – фаза цикла сокета комнаты, при которой происходит обработка логики игрового процесса и действий игроков.

## Анализ аналогов и прототипов.

Аналогом данного игрового приложения является уже описанное ранее приложение Pong от фирмы Atari. Далее приведены правила игрового процесса классической версии аркадной игры Pong:

Pong является простым спортивным симулятором тенниса. Небольшой квадратик, заменяющий пинг-понговый мячик, двигается по экрану по линейной траектории. Если он ударяется о периметр игрового поля, то его траектория изменяется в зависимости от угла столкновения. Если шарик отбивается ракеткой игрока, то его движение дополнительно зависит от скорости и направления движения ракетки. Периметр игрового поля обозначен краями экрана, а мячик не может покинуть поле через верхний или нижний край. В верхней части поля отображаются очки игроков, у каждого на своей половине экрана.

Игровой процесс состоит в том, что игроки передвигают свои ракетки вертикально для защиты своих ворот. В начале каждого раунда мячик подаётся одному из игроков, и раунд продолжается до тех пор, пока один из игроков не заработает очко. Это происходит тогда, когда его противник не может отбить мячик. Со временем игры скорость движения мячика постепенно увеличивается, и так игра усложняется. Вся игра в Pong длится не более трёх-четырёх минут. [[2]](#лит2)

Еще одним аналогом является многопользовательская версия Pong, реализованная в веб-браузере. В данном приложении возможна как игра против компьютера, так и против реального оппонента (рисунок 1). В данном приложении нет системы комнат, но реализована система приглашений (по ссылке на игровую сессию). Правила игрового процесса идентичны классическому приложению Pong. Управление ракеткой производится с помощью мыши.



Рисунок 1 – Многопользовательское приложение Pong, реализованное в браузере

Данный аналог хоть и не реализует задуманную систему комнат, но показывает, как должен происходить игровой процесс в уже запущенной игровой сессии. В дополнение, это приложение показывает реализацию игры с компьютером (т.е. игровой процесс против определенного алгоритма), которую тоже возможно реализовать в запланированном приложении.

## Перечень задач, подлежащих решению в процессе разработки.

Интерфейс клиентского приложения должен обеспечить комфортное, интерактивное взаимодействие, как и при авторизации и выборе комнат, так и в процессе самой игры. В процессе разработки клиентской части должны быть выделены следующие задачи:

* Выбор никнейма при открытии приложения
* Реализация подключения к основному серверному потоку для дальнейшего отображения существующих комнат
* Выбор и подключение к уже созданной комнате
* Создание собственной комнаты, настройка ее параметров (пароль, количество очков для победы)
* Реализация игровой сессии, логики приложения на клиенте.
* Реализация сетевого обмена данных с сервером, построенном на протоколе UDP (отправление координат ракетки, мяча, синхронизация).

Интерфейс серверной части должен представлять собой подробное отображение информации о возможных ошибках в игровых сессиях, и других событий. В процессе разработки серверной части должны быть выделены следующие задачи:

* Реализация прослушивания основного серверного потока для первичных подключений пользователей.
* При авторизации пользователя должна быть произведена проверка на уникальность введенного никнейма.
* Создание отдельных потоков (со своими сокетами с уникальными портами) для каждой из комнат.
* При присоединении или создание пользователем комнаты – передавать ему порт нового сокета.
* Реализация алгоритмов игровой логики и синхронизации игровой сессии (по средствам сетевого обмена с обоими клиентами сессии).
* Реализация отключения пользователя из сессии, комнаты и основного серверного потока.

## Обоснование выбора инструментов и платформы для разработки.

Для разработки данного приложения был выбран язык программирования Java с платформой JVM.

Java представляет собой язык программирования и платформу вычислений, которая была впервые выпущена Sun Microsystems в 1995 г. Существует множество приложений и веб-сайтов, которые не работают при отсутствии установленной Java, и с каждым днем число таких веб-сайтов и приложений увеличивается. Java отличается быстротой, высоким уровнем защиты и надежностью. От портативных компьютеров до центров данных, от игровых консолей до суперкомпьютеров, используемых для научных разработок, от сотовых телефонов до сети Интернет. [[3]](#лит3)

Java Virtual Machine (сокращенно Java VM, JVM) — виртуальная машина Java — основная часть исполняющей системы Java, так называемой Java Runtime Environment (JRE). Виртуальная машина Java исполняет байт-код Java, предварительно созданный из исходного текста Java-программы компилятором Java (javac). [[4]](#лит4)

JVM имеет две основные функции:

* Позволяет запускать Java-приложения на любых устройствах или операционных системах (принцип кроссплатформенности).
* Управляет и оптимизирует память, используемую приложением.

Технологии Java были выбраны исходя из ее основного принципа – кроссплатформенность. Это позволит использовать готовое десктопное приложение на любой операционной системе. Java имеет собственные библиотеки для реализации сокетов, что позволяет не прописывать данную часть программы для каждой ОС, это остается за JVM. Но в то же время Java не слишком сильно “обертывает” нативные сокеты операционных систем, оставляя разработчику привычные операции (accept, bind, connect и др.), что предоставляет весьма гибкую работу с сетью. Платформа Java позволит в будущем расширить приложение, реализовав, например, мобильную версию приложения (а именно – клиентской части).

Для данного типа приложения (игровое) больше всего подходит использования протокола UDP из-за его быстрой скорости и минимального размера заголовка.

Протокол пользовательских дейтаграмм (UDP) является транспортным протоколом без установки соединения (ненадежный протокол). Он выполняет небольшую проверку на наличие ошибок и ничего не добавляет к службам IP, кроме дополнения связи процесс-процесс вместо связи хост-хост.

UDP – это простой протокол с минимальной дополнительной нагрузкой. Его можно использовать, если процессу требуется отправить небольшое сообщение, и при этом надежности не придается особого значения. Отправка сообщения с помощью UDP требует намного меньше времени по сравнению с использованием TCP. UDP очень подходит для приложений мультимедиа и группового вещания. [[5]](#лит5)

Но в то же время для сокета, принимающего запросы от пользователей, и для сокетов комнат, находящихся вне игрового состояния вполне возможно использование надежного протокола TCP, так как в данной ситуации не требуется высокоскоростной доставки данных.

Протокол управления передачей (TCP) - это надежный, но сложный протокол транспортного уровня. TCP добавляет в IP надежность функции поддержки соединения. [[5]](#лит5)

# ПРОЕКТНАЯ ЧАСТЬ

## Разработка структуры приложения.

На рисунке 2 представлена структурная схема системы.



Рисунок 2 – Структурная схема системы

Клиентская и серверная часть взаимодействует по протоколу, построенному поверх транспортного протокола UDP. Клиент и сервер постоянно поддерживают соединение (см. соединение с “Основной UDP сокет сервера”). По данному соединению также передаются данные о комнатах и игроках. Через это же соединение клиент может запросить присоединение к комнате или ее создание. После присоединении двух игроков сервер создает отдельный поток для обработки игровой сессии данной комнаты (см. потоки игровых сессий). Клиентский сокет начинает работать с двумя сокетами сервера – поддерживать подключение с основным и обмениваться игровыми данными с сокетом игровой сессии.

Для представления принципов взаимодействия пользователя с системой была составлена UML-диаграмма вариантов использования (рисунок 3).



Рисунок 3 – UML-диаграмма вариантов использования

Пользователь может присоединиться к нужному серверу обязательно введя никнейм (обязательно осуществляется проверка на уникальность никнейма для данного сервера).

Далее пользователю доступно два варианта действий – присоединиться к существующей комнате, или создать собственную. После того как пользователь оказался в комнате он может ожидать оппонента или может покинуть комнату до начала игрового процесса.

Если в комнате присутствуют оба игрока, то запускается игровая сессия, в ходе которого пользователю доступно управление одной из ракеток. В любой момент времени пользователь может отключится от сервера, нажав на соответствующую кнопку в окне выбора комнат, либо просто закрыв приложение.

## Разработка алгоритмов обработки информации.

Одним из самых используемых алгоритмов в данном приложении является алгоритм выбора действий, зависящий от команды. Данная схема обычно организована в виде цикла, итерирующего во время выполнения соединения сокета или работы в целом, в котором поток находится либо в состоянии ожидания пакета, либо в состоянии ожидания уведомления от другого потока (подробнее рассмотрено в части архитектуры приложения). Данный алгоритм реализован, как и на клиентской, так и на серверной части. Для примера приведен алгоритм только серверной части (рисунок 4).



Рисунок 4 – Алгоритм обработки сообщения от клиента

## Разработка архитектуры приложения.

Данная система имеет клиент-серверную архитектуру (рисунок 5). Серверная часть представлена в виде консольного приложения, в котором отображаются основные действия сервера и сетевые данные, полученные от клиентов. Клиентская часть представлена в виде десктопного приложения для ПК с графическим интерфейсом пользователя.



Рисунок 5 – Клиент-серверная архитектура

Архитектура клиентской части состоит из нескольких потоков:

1. поток для пользовательского интерфейса
2. поток для UDP сокета
3. поток для изменения пользовательского интерфейса на основе полученных данных

Первые два потока говорят сами за себя. Первый – основной поток, в котором запускается приложение. Этот поток порождает два оставшихся.

Поток для UDP сокета ожидает поступающего сообщения от сервера, либо отправляет данные серверу.

Для перерисовки динамического контента пользователя используется отдельный поток, чтобы избежать неприятных задержек и остановок в работе пользовательского интерфейса. Этот поток постоянно находится в состоянии ожидания (метод wait) на общем, с потоком сокета, объекте. После получения сокетом определенного сообщения с командой, требующей перерисовки контента, поток с сокетом переводит общий монитор в состояние уведомления (метод notify) и переопределяет публичное поле команды для перерисовки интерфейса. После получения уведомления на общем мониторе поток для перерисовки контента переходит в активную фазу, считывает поле команды для перерисовки, и на основе нее производит добавление или удаление компонентов пользователя. После выполнения данных действий весь этот цикл повторяется.

Все вышесказанное представлено в виде диаграммы последовательности (рисунок 6).



Рисунок 6 – UML-диаграмма последовательности работы потоков клиентской части

На UML-диаграмме последовательности (рисунок 7) показана подробная схема подключения и обмена данными между клиентской и серверной частью при присоединении к существующей комнате.

В начале пользователь присоединяется к серверу, и получает данные о комнатах. Выбрав комнату и присоединившись к ней, как уже было сказано при разборе структурной схемы системы, сервер инициализирует отдельный сокет в новом потоке для организации игрового процесса.

После окончания игры (или потери соединения с одним из игроков) сокет игровой сессии, как и поток, в котором он находится, перестает существовать. Пользователи возвращаются к окну выбора комнат.



Рисунок 7 – UML-диаграмма последовательности при присоединении к существующей комнате

## Разработка систем передачи информации.

Протокол реализован с помощью текстового формата обмена данными JSON. Все сообщения протокола “упаковываются” в JSON. Это позволяет весьма легко и быстро преобразовывать сложные структуры классов и массивов в обычную строку, а также наоборот – из строки в объекты. Так же данный формат легко понимается человеком, а в следствии – отладка приложения упрощается.

Протокол передаваемых пакетов в данном приложении должен включать обязательное поле command, без него пакеты будут восприниматься как неопознанные и не рассматриваться. В зависимости от нужной команды остальная часть пакета может меняться, но обязана состоять минимум из одного из доступных вариантов: поле сообщения, поле данных клиента, поле списка комнат, поле данных ракетки, поле данных мяча. (рисунок 8)

* Поле сообщения. Данное поле представляет собой обычную строку. Оно используется если команда предполагает собой отправку примитивных данных. В данном приложении поле сообщения используется всего лишь единожды – когда сервер отправляет информацию клиентам о созданном для них отдельном сокете (поле команды - CONNECT\_TO\_A\_NEW\_SOCKET). В данном поле сообщения передается порт нового сокета.
* Поле данных клиента. Данное поле – самое используемое и является авторизирующим, т.е. позволяет серверу определить является ли полученное сообщение пользовательским. Каждое сообщение пользователя, когда он находится вне игры, должно содержать информацию о нем в этом поле. Сообщения, не содержащие данного поля, рассматриваться не будут (кроме сообщений во время игровой сессии). Данное поле представляет собой экземпляр объекта типа Client, состоящего из полей для никнейма, IP клиента, порта клиента.
* Поле списка комнат. Данное поле используется для отправления сервером данных о комнатах (поле команды - RETURN\_ROOMS). Представляет собой массив объектов типа Room, состоящие из полей ID, списка клиентов в комнате, статуса, флагов isEmpty и isMax.
* Поле данных ракетки. Данное поле состоит из объекта GamePaddle, который включает поля pNum – номер игрока в комнате и yDir – направление движения ракетки вдоль оси Y.
* Поле данных мяча. Данное поле состоит из объекта GameBall, который включает поля X и Y, соответствующие координатам направления движения мяча.



Рисунок 8 – Формат передаваемых сообщений

Полный список команд, использующихся в передаваемых сообщениях приведен в [приложении А](#приложение1). Для улучшения производительности можно реализовать сокращенную кодировку каждой команды (например, обычными числами как в HTTP), но для наглядности и упрощенной отладки был использован вариант с использованием строки.

Передача данных игровой сессии заключается, как уже было сказано выше, в передаче данных направлений ракетки и направления мяча. Особенностью в данном приложении является простота и в тоже время производительность данной системы передачи данных. Изначально планировалось передавать координаты мяча и ракеток постоянно, сильно нагружая сеть. Удалось реализовать вариант, позволяющий разгрузить сеть, передавая сообщения намного реже чем планировалось, но вследствие этого нагрузка возросла на клиентской части, которой необходимо тратить больше ресурсов на вычисления позиций ракеток и мяча. Сама передача данных заключается в двух частях:

* Передача направления движения мяча. Когда мяч касается поверхности (т.е. отражается), передается всего одно сообщение от каждого клиента с его направлением движения после отражения (координаты X и Y точки, к которой должен двигаться мяч). Данные сообщения пересылаются пользователям, в следствии чего достигается синхронизация в момент отражения.
* Передача направления движения ракеток. Данные сообщения передаются в двух вариантах – при нажатии клавиши перемещения ракетки, и при ее отпускании. При нажатии передается номер клиента в игровой сессии и направление движения ракетки вдоль оси Y (-1 или 1, т.е. вверх или вниз соответственно). Когда пользователь перестает нажимать на кнопку, то передается то же сообщение, но с направлением движения ракетки 0 (т.е. ракетка перестает двигаться).

## Реализация функционирующего приложения.

Так как была выбрана платформа Java, то задачу вызова системных функций работы с сетью реализует виртуальная машина Java (JVM). Java имеет удобные для разработчика классы работы с сокетами, являющиеся частью пакета java.net: ServerSocket, Socket, DatagramSocket. Первые два используются для реализации TCP протокола, но так как данное приложение полностью построено на UDP, то был использован класс DatagramSocket как для серверной части, так и для клиентской.

Класс DatagramSocket подразумевает использование класса DatagramPacket. Создание экземпляра данного UDP-пакета нужно для хранения клиентских данных с использованием буфера, в виде массива типа данных byte. Размер массива соответствует размеру передаваемых пользовательских данных.

## Разработка интерфейса взаимодействия пользователя с системой.

Пользовательский интерфейс клиентской части приложения представляет собой два окна:

* окно присоединения и выбора комнат
* игровое окно (рисунок 9)

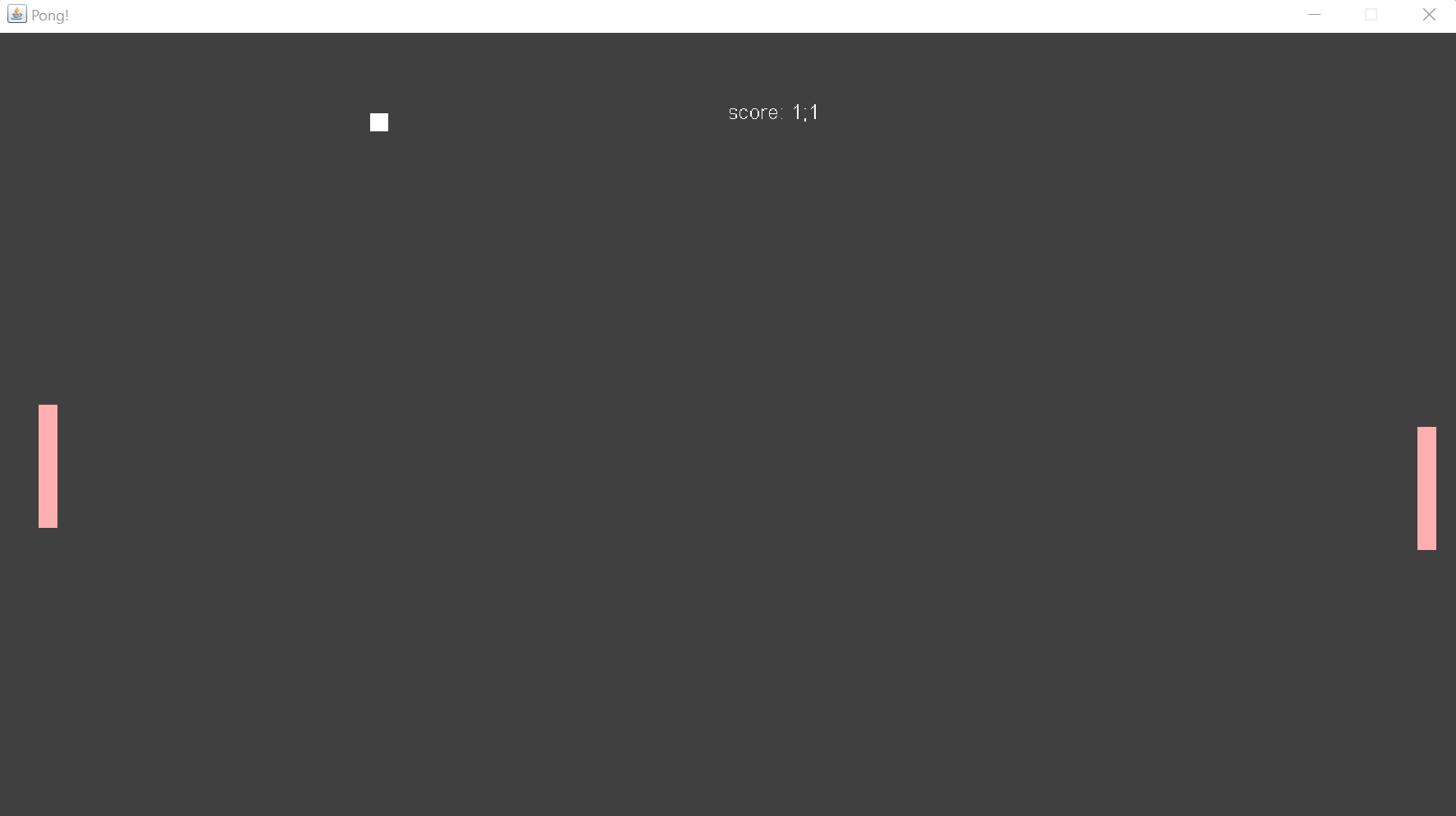


Рисунок 9 – Игровое окно

Окно присоединения и выбора комнат состоит из нескольких компонентов:

* компонент ввода никнейма и данных сервера (рисунок 10)
* компонент выбор или создание комнат (рисунок 11)
* компонент комнаты (рисунок 12)

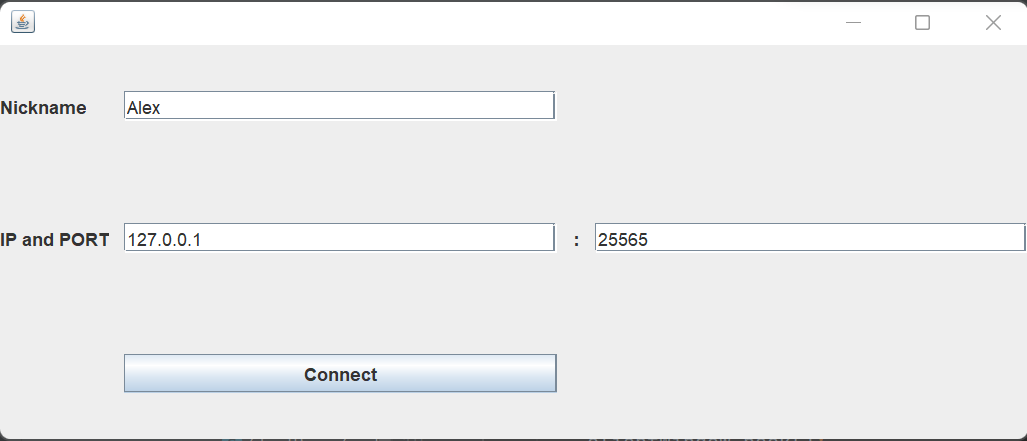


Рисунок 10 – Ввод никнейма и данных сервера

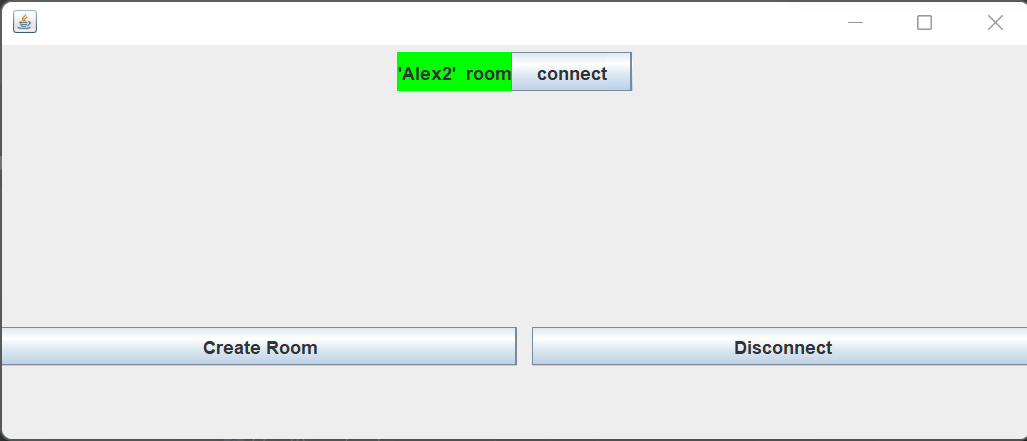


Рисунок 11 – Выбор комнат

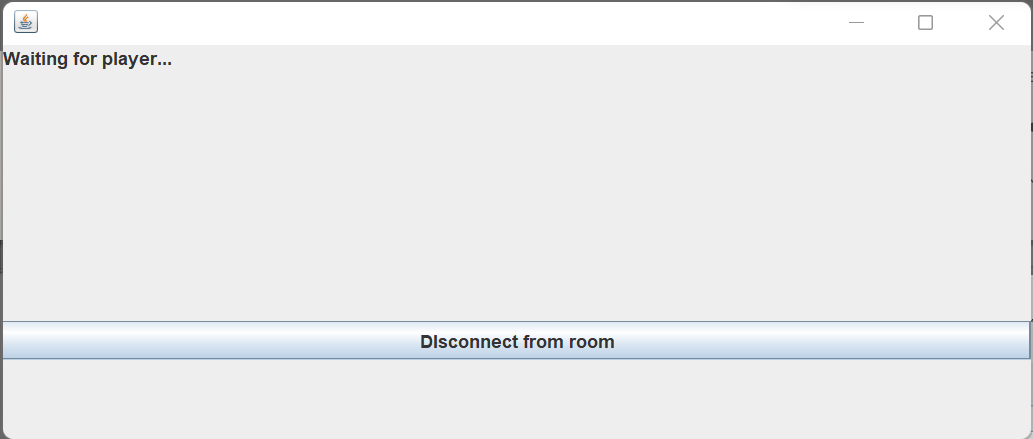


Рисунок 12 – Интерфейс комнаты

Данное окно имеет вид одного из трех перечисленных компонентов. Реализован дополнительный компонент, использующийся при ошибке (рисунок 13). Текст компонента зависит от конкретной ошибки.

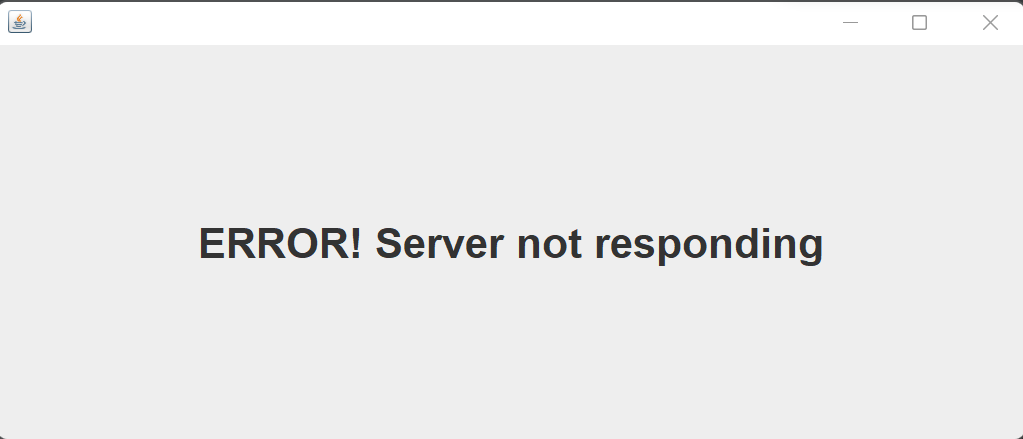


Рисунок 13 – Компонент, отображаемый при возникновении ошибки

# ПРОИЗВОДСТВЕННО-ТЕХНОЛОГИЧЕСКАЯ ЧАСТЬ

## Тестирование и отладка макета рабочей программы.

В ходе отладки приложения были выявлены ошибки, чаще всего связанные с непредвиденным поведением, например: отключение игрока во время открытого окна игры (окно игры не закрывалось моментально у другого игрока), “гонка” сообщений (некоторые сообщения, зависящие друг от друга и отправленные одновременно могли каждый раз приходить в разом порядке), ошибки, связанные с отображением пользовательского интерфейса и т.д. Большинство ошибок и исключительных ситуаций были выявлены и устранены. Но тем не менее программа требует дальнейшего полного тестирования и отладки, так как исключительные или непредвиденные ситуации все еще возможны (например, если использовать программу с некачественным интернетом)

В ходе отладки приложения и изучения системных требований платформы Java были установлены системные требования данного приложения:

* ОС: Windows Vista SP2 или более поздняя; Mac OS X 10.7.3+ или более поздняя; Ubuntu Linux 10.04 или более поздняя
* ОЗУ: 256 МБ и выше
* Дисковое пространство: 512 МБ и выше
* Процессор: минимальное требование - Pentium 2 266 МГц
* Видеокарта: поддержка DirectX9

## Разработка руководства пользователя

1. При запуске приложения откроется окно для ввода данных для сервера. Необходимо ввести уникальный никнейм, IP-адрес сервера и его порт. При подключении может возникнуть ошибка – тогда следует перепроверить введенные данные сервера, а также качество сетевого подключения. Может возникнуть ситуация, при которой никнейм будет уже кем-то занят на этом сервере – тогда следует выбрать другой уникальный никнейм.
2. При успешном подключение откроется окно выбора комнат. Чтобы присоединиться к комнате, следует нажать соответствующую кнопку “Connect” рядом с названием комнаты. При желании возможно создать свою собственную комнату и ожидать подключения другого игрока. Так же имеется кнопка для отключения от сервера.
3. После открытия окна комнаты вы можете отключится от нее или ожидать начала игры, если второй игрок уже присоединился. После окончания 3-х секундного таймера откроется окно непосредственно с игрой.
4. Управление во время игры происходит двумя клавишами – стрелкой вверх и стрелкой вниз. Данные стрелки позволяют вертикально передвигать вашу ракетку. Цель игры – набрать как можно больше очков, отражая мяч ракеткой в сторону противника. Если мяч попадает в область за ракеткой, то оппонент получает одно очко. По окончании игры пользователи вернутся в комнату, где они могут остаться, чтобы началась еще одна игра.

# ЗАКЛЮЧЕНИЕ

В ходе выполнения данной курсовой работы было разработано многопользовательское игровое приложение “Pong Online”.

Работа выполнялась в несколько этапов: разработка структуры и архитектуры приложения, выбор платформы для разработки, реализация серверной и клиентской части.

Были сформированы практические навыки по разработке и реализации программного приложения с использованием изученных средств и методов программирования, функционирующего в вычислительной сети, для автоматизации процессов обработки информации в конкретной области.

В данной работе были выполнены все поставленные задачи.

# СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ

1. Онлайн-игры. Особенности и преимущества [Электронный ресурс] – URL: <https://www.business-gazeta.ru/article/299734>
2. Истории компьютерных игр: Pong и детство Atari [Электронный ресурс] – URL: <https://itc.ua/articles/istorii-kompyuternyih-igr-pong-i-detstvo-atari/>
3. Что такое технология Java и каково ее применение? [Электронный ресурс] – URL: <https://java.com/ru/download/help/whatis_java.html>
4. Что такое JVM? Знакомство с виртуальной машиной Java [Электронный ресурс] – URL: <https://topjava.ru/blog/what-is-the-jvm>
5. Обзор сокетов [Электронный ресурс] – URL: <https://www.ibm.com/docs/ru/rtw/9.0.0?topic=transports-sockets-overview>
6. Э.Таненбаум, Д.Уэзеролл Компьютерные сети. 5-е изд. СПб.: Питер, 2012.
7. Е.В. Смирнова, А.В. Пролетарский, Е.А. Ромашкина и др. Технологии коммутации и маршрутизации в локальных компьютерных сетях. Учебное пособие. М.: МГТУ им. Н.Э. Баумана, 2013.
8. Васильков А.В., Васильков А.А., Васильков И.А. Информационные системы и их безопасность. М.: Форум, 2013.
9. Ачилов Р.Н. Построение защищенных корпоративных сетей. М:ДМК-пресс, 2013. [https://e.lanbook.com/book/66472?category\_pk=1547#book\_name](https://e.lanbook.com/book/66472?category_pk=1547%23book_name)
10. Сидоров В.Н., Сломинская Е.Н., Полникова Т.В., Макарова О.Ю. Оформление графической части выпускной квалификационной работы. Учебное пособие. М.: МГТУ им. Н.Э. Баумана, 2016.

# ПРИЛОЖЕНИЕ А

*// CONNECTION:*

  ALIVE\_CLIENT = "ALIVE\_CLIENT";

  ALIVE\_SERVER = "ALIVE\_SERVER";

  CLIENT\_OFFLINE = "CLIENT\_OFFLINE";

  SERVER\_OFFLINE = "SERVER\_OFFLINE";

  UNKNOWN\_ERROR = "UNKNOWN\_ERROR";

  CLIENT\_CONNECT\_MSG = "FIRST\_CONNECTION";

  NICKNAME\_IS\_TAKEN = "NICKNAME\_IS\_TAKEN";

  NICKNAME\_IS\_EMPTY = "NICKNAME\_IS\_EMPTY";

  MAX\_CLIENTS = "MAX\_CLIENTS";

  NOT\_AUTH = "NOT\_AUTH";

  SUCCESS\_CONNECT = "SUCCESSFUL\_CONNECTION";

*// ROOMS:*

  ROOM\_IN\_GAME = "ROOM\_IN\_GAME";

  ROOM\_WAITING\_PLAYER = "ROOM\_WAITING\_PLAYER";

  ROOM\_WAITING\_START = "ROOM\_WAITING\_START";

  GET\_ROOMS = "GET\_ROOMS";

  NO\_ROOMS = "NO\_ROOMS";

  RETURN\_ROOMS = "RETURN\_ROOMS";

  CREATE\_ROOM = "CREATE\_ROOM";

  DISCONNECT\_FROM\_ROOM = "DISCONNECT\_FROM\_ROOM";

  JOIN\_ROOM = "JOIN\_ROOM";

  REQUEST\_START\_GAME = "REQUEST\_START\_GAME";

  CONNECT\_TO\_A\_NEW\_SOCKET = "CONNECT\_TO\_A\_NEW\_SOCKET";

*// GAME NETWORK:*

  READY\_FOR\_GAME = "READY\_FOR\_GAME";

  CONFIRM\_CONNECTION\_TO\_GAME = "CONFIRM\_CONNECTION\_TO\_GAME";

  SEND\_PADDLE\_Y\_DIR = "SEND\_PADDLE\_Y\_DIR";

  GET\_OPPONENT\_PADDLE\_Y\_DIR = "GET\_OPPONENT\_PADDLE\_Y\_DIR";

  GET\_BALL\_ANGLE = "GET\_BALL\_ANGLE";

  GET\_BALL\_INIT = "GET\_BALL\_INIT";

  SEND\_BALL\_REFLECTION = "SEND\_BALL\_REFLECTION";

  SEND\_BALL\_DESTINATION = "SEND\_BALL\_DESTINATION";

  GET\_BALL\_DESTINATION = "GET\_BALL\_DESTINATION";

  RIGHT\_WON = "RIGHT\_WON";

  LEFT\_WON = "LEFT\_WON";

END\_GAME = "END\_GAME";

*//GAME GUI:*

  PADDLE\_REPAINT = "PADDLE\_REPAINT";

*// GUI:*

  GUI\_REPAINT = "GUI\_REPAINT";

  GUI\_ROOM\_REPAINT = "GUI\_ROOM\_REPAINT";