

СОДЕРЖАНИЕ

ВВЕДЕНИЕ	5
1 Описание и анализ предметной области	6
2 Постановка задачи.....	6
2.1 Характеристика объекта моделирования:.....	6
2.2 Общие требования к проектируемой системе:	7
3 Описание используемых технологий	8
3.2 Java Swing	9
4 Проектирование системы.....	9
4.1 Структурная схема системы	9
4.2 Разработка информационно-логического проекта системы	11
4.2.1 Диаграмма вариантов использования.....	11
4.2.2 Диаграмма деятельности.....	13
4.2.3 Диаграмма компонентов	14
4.2.4 Диаграмма последовательности	16
5 Описание интерфейса системы	17
СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ	20
ПРИЛОЖЕНИЕ А Листинг модулей программы	21

1 Описание и анализ предметной области

Предметной областью часто называют ту часть реального мира, которая имеет непосредственное отношение к разработке той или иной программы. Предметная область состоит из тех частей, которые могут оказать влияние на создание конкретных объектов и их взаимосвязей между объектами.

Распределенная система – набор независимых узлов, не имеющих общей совместно используемой памяти и общего единого времени (таймера) и взаимодействующих через коммуникационную сеть посредством передачи сообщений. В курсовой работе для создания распределённой системы, реализующей игру, применено функциональное разделение на клиентскую и серверную часть. Таким образом, различные узлы системы будут выполнять разные задачи.

Крестики-нолики— логическая игра между двумя противниками на квадратном поле 3х3 клетки или большего размера (вплоть до «бесконечного поля»). В классической игре игроки по очереди ставят на свободные клетки поля $n \times n$ знаки (один всегда крестики, другой всегда нолики). Первый, выстроивший в ряд свои фигуры по вертикали, горизонтали или диагонали, выигрывает. Если игроки заполнили все ячейки и оказалось, что ни в одной вертикали, горизонтали или диагонали нет одинаковых знаков, партия считается закончившейся в ничью. Первый ход делает игрок, ставящий крестики [3].

2 Постановка задачи

2.1 Характеристика объекта моделирования:

- 1) объект моделирования: игра «Крестки-нолики;
- 2) виды моделируемой деятельности:
 - процесс генерирования игрового поля размера $N \times N$;
 - процесс ведения игры согласно правилам;

- процесс выбора хода противником;
 - процесс визуализации игрового процесса.
- 3) правила игры «Крестики-нолики»:
- игровое поле должно иметь размер не менее 3x3 и не более 5x5 клеток;
 - игроки ходят по очереди;
 - игра завершается победой одного из игроков, когда он выстроил в ряд свои фигуры по вертикали, горизонтали или диагонали. Если поле оказалось заполненным и оказалось, что ни в одной вертикали, горизонтали или диагонали нет одинаковых знаков, игра закончена в ничью.
- 4) входные данные системы:
- размеры поля $N \times N$;
 - координаты хода игрока.
- 5) выходные данные системы:
- двумерный массив клеток поля;
 - координаты хода компьютера.

2.2 Общие требования к проектируемой системе:

- 1) клиентское приложение:
- предоставление игроку возможности задать размер игрового поля;
 - предоставление возможности сделать ход игроку;
 - визуализация игрового процесса.
- 2) серверное приложение:
- генерация игрового поля согласно размерам, заданным пользователем;
 - совершение хода противником;
 - определение игрока, который имеет право сделать ход;

- определение итогов игры.
- 3) функции пользователя:
 - задать размеры игрового поля;
 - сделать ход.
- 3 Описание используемых технологий

3.1 Технология RMI

Технология Remote Method Invocation (RMI) впервые была внедрена еще в Java Development Kit(JDK) версии 1.1. Она определяет, как объекты себя ведут, в каком месте и из-за чего могут возникнуть исключения, каким образом происходит управление памятью, а также как параметры передаются в удаленные методы и как они оттуда возвращаются. Технология RMI, поддерживаемая платформой Java, предлагает способы создания объектов, методы которых могут быть вызваны из среды других виртуальных машин, – в том числе и работающих на других хост-компьютерах.

Архитектура технологии RMI основывается на одном важном правиле: определение поведения и реализация этого поведения должны быть физически разделены. Это значит, что RMI позволяет хранить отдельно описание поведения и код его реализации на нескольких отдельно работающих JVM. Изначально в качестве определения удаленного сервиса в RMI используют интерфейсы Java (interfaces). Ниже представлена диаграмма, которая наглядно демонстрирует эту концепцию разделения (см. рисунок 1).

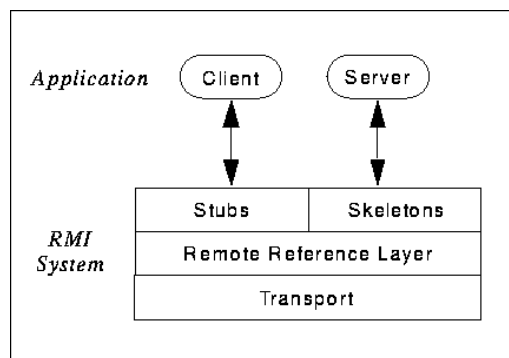


Рисунок 1 – Концепция разделения RMI

- 1) подсистема взаимодействия с сервером, отвечающая за передачу на сервер информации от клиента;
- 2) подсистема визуального отображения, отвечающая за отображение текущего игрового поля;
- 3) подсистема обработки пользовательского ввода, позволяющая игроку ввести размеры игрового поля и сделать ход;

В состав серверной части системы входят следующие подсистемы:

- 1) подсистема взаимодействия с клиентом, отвечающая за получение и передачу игровых данных;
- 2) подсистема контроля состояния игры, контролирующая, какой игрок может сделать ход и хранящая текущее игровое поле;
- 3) подсистема совершения хода компьютером, симулирующая ход компьютера;

4.2 Разработка информационно-логического проекта системы

Для специфицирования (построения точных, недвусмысленных и полных моделей) системы и ее документирования используется унифицированный язык моделирования UML.

Унифицированный язык моделирования (Unified Modeling Language – UML) – это стандартный инструмент для разработки «чертежей» программного обеспечения. Его можно использовать для визуализации, спецификации, конструирования и документирования артефактов программных систем. UML подходит для моделирования любых систем – от информационных систем масштаба предприятия до распределенных Web-приложений и даже встроенных систем реального времени [7].

4.2.1 Диаграмма вариантов использования

Диаграмма вариантов использования представляет собой наиболее общую концептуальную модель сложной системы, которая является исходной для построения всех остальных диаграмм.

- определить общие границы и контекст моделируемой предметной области на начальных этапах проектирования системы;
- сформулировать общие требования к функциональному поведению проектируемой системы;
- разработать исходную концептуальную модель системы для ее последующей детализации в форме логических и физических моделей;
- подготовить исходную документацию для взаимодействия разработчиков системы с ее заказчиками и пользователями.

Диаграмма вариантов использования со стороны пользователя приведена на рисунке 3.

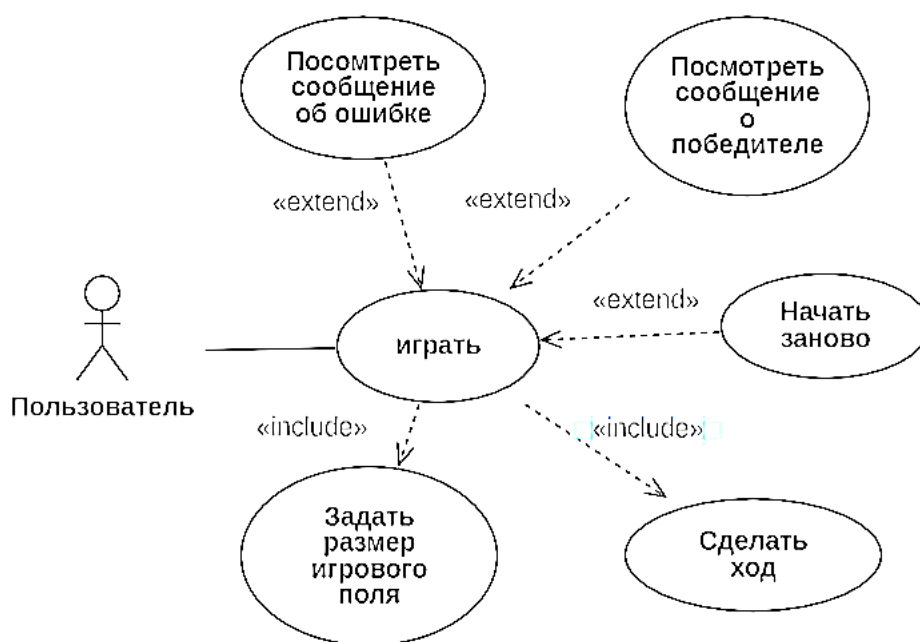


Рисунок 3 – Диаграмма вариантов использования системы

Для начала игры необходимо задать размер игрового поля. В процессе игры основной функцией пользователя является возможность сделать ход. Кроме того, находясь в процессе игры, пользователь может начать заново, нажав на соответствующую кнопку. По окончании игры пользователь может просмотреть сообщение о победителе, в случае ошибки ввода пользователь должен получать сообщение об ошибке.

4.2.2 Диаграмма деятельности

Диаграмма деятельности используется для моделирования динамических аспектов поведения системы. Она является частным случаем диаграмм состояний. Они позволяют реализовать особенности процедурного и синхронного управления, обусловленного завершением внутренних действий и активностей [7].

На рисунке 3 приведена диаграмма деятельности системы. Сначала система отображает главный экран и форму ввода размера игрового поля. В процессе игры система предоставляет пользователю возможность сделать ход. После каждого хода система производит проверку, определяющую, может ли игрок сходить снова, и проверку на окончание игры.

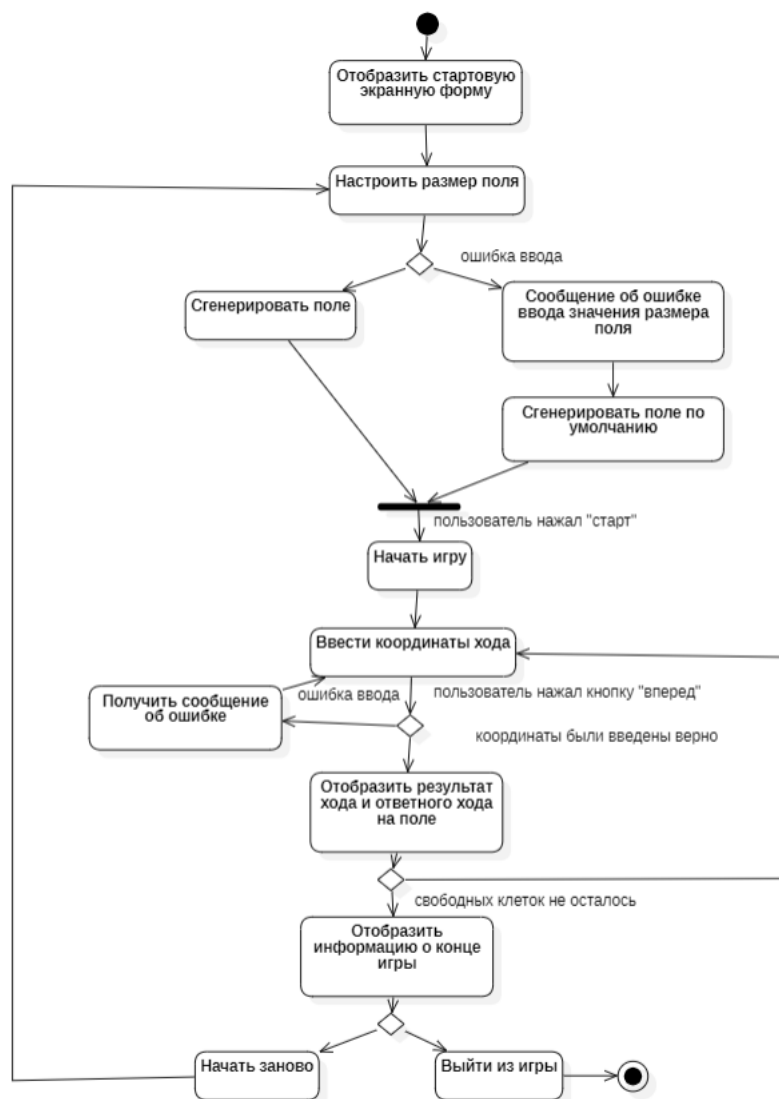


Рисунок 3 – Диаграмма деятельности системы

4.2.3 Диаграмма компонентов

Диаграмма компонентов – диаграмма физического уровня, которая служит для представления программных компонентов и зависимостей между ними [12].

Диаграмма компонентов разрабатывается для следующих целей:

- визуализация общей структуры исходного кода программной системы;
- спецификация исполнимого варианта программной системы;
- обеспечение многократного использования отдельных фрагментов программного кода;
- представление концептуальной и физической схем баз данных.

На рисунке 4 приведена диаграмма компонентов, их описание приведено в таблице 1.

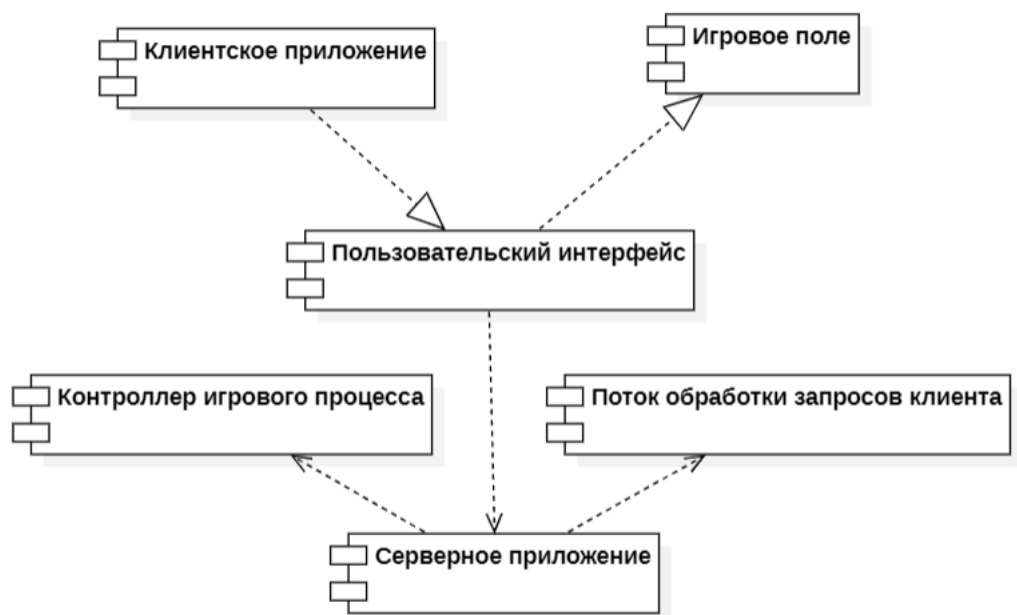


Рисунок 4 – Диаграмма компонентов системы

Таблица 1 – Описание компонентов системы

Название компонента	Назначение компонента	Подсистема
Клиентское приложение	Позволяет пользователю взаимодействовать с системой	Задания размеров поля, совершение хода пользователем
Пользовательский интерфейс		
Игровое поле	Отвечает за отображение текущего состояния игры	Отрисовка игрового поля
Серверное приложение	Позволяет работать с несколькими клиентскими приложениями одновременно	Взаимодействия с клиентом
Поток обработки запросов клиента	Отвечают за обработку запросов клиентов	
Контроллер игрового процесса	Отвечает за последовательность действий в игровом процессе и за хранение позиций на поле	Совершения хода компьютером, хранения клеток поля, определения

		активного игрока, определение итогов игры
--	--	--

4.2.4 Диаграмма последовательности

Диаграмма последовательности UML (sequence diagram) — такая диаграмма, на которой показаны взаимодействия объектов, упорядоченные по времени их проявления. Основные элементы диаграммы последовательности это: обозначения объектов (прямоугольники), вертикальные линии, отображающие течение времени при деятельности объекта, и стрелки, показывающие выполнение действий объектами [20].

На рисунке 5 приведена диаграмма последовательности для варианта использования «Сделать ход».

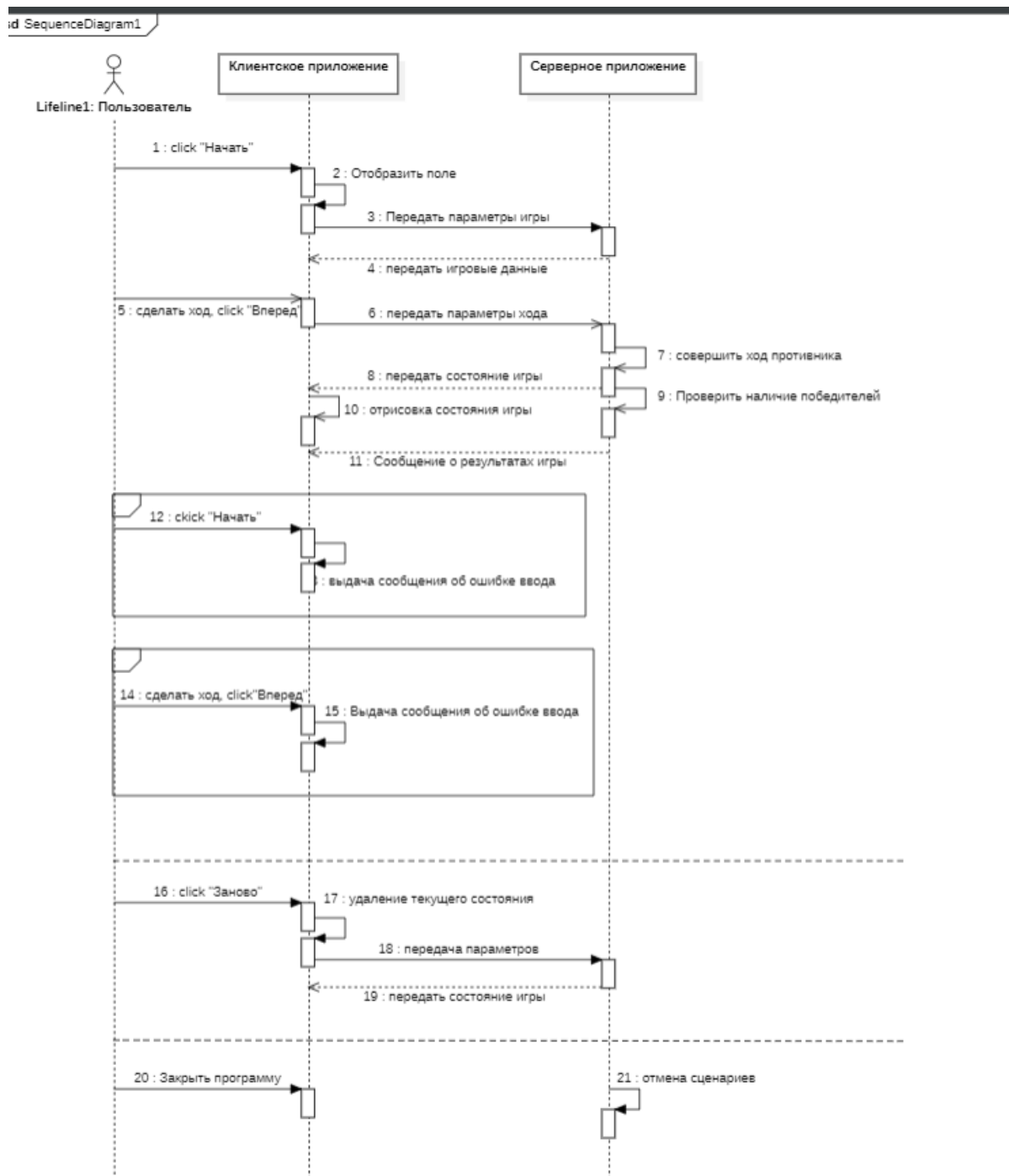


Рисунок 5 – Диаграмма последовательности для варианта использования «Сделать ход»

5 Описание интерфейса системы

При запуске клиентской части приложения пользователь видит начальный экран, представленный на рисунке 6. Здесь пользователь может задать размеры игрового поля, далее по нажатию на кнопку “Начать” генерируется

поле нужного размера. В любой момент игры пользователь может начать игру заново, нажав кнопку «Заново!»

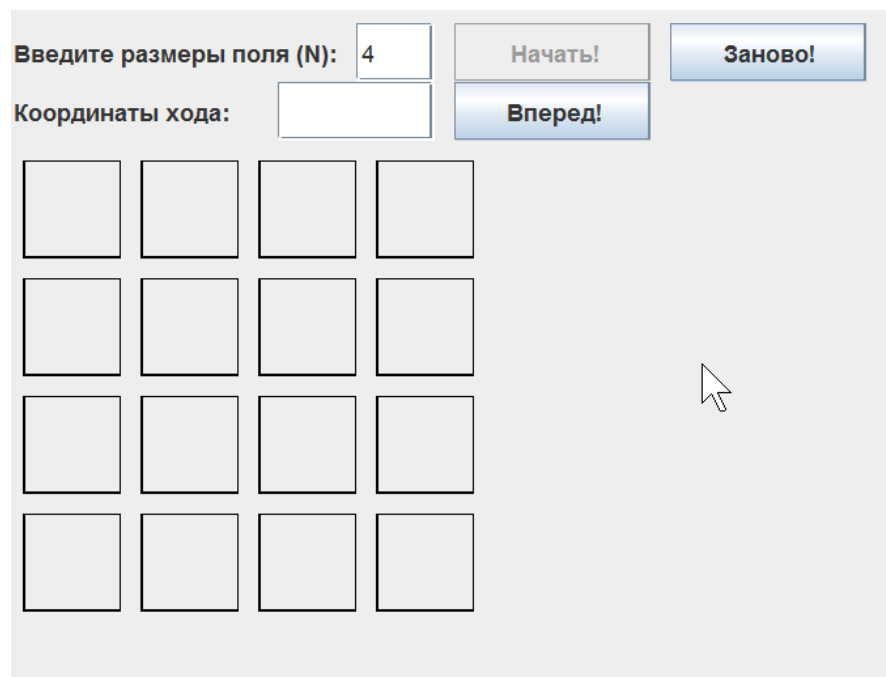


Рисунок 6 – Стартовая экранная форма

После начала игры, пользователь вводит координаты хода в соответствующую область, по нажатию кнопки «вперед» (Рисунок 7), система отображает ход пользователя и компьютер совершает ответный ход.

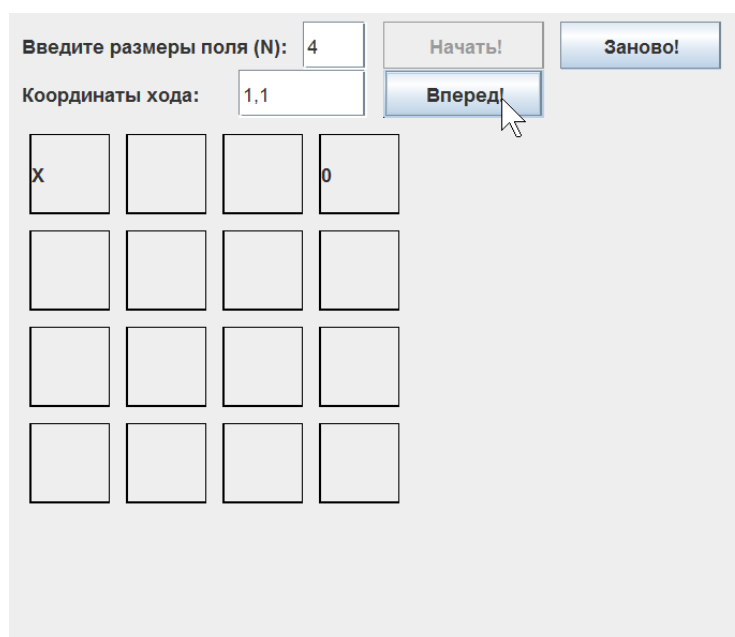


Рисунок 7 – Экранная форма процесса игры

Если какие-либо данные введены некорректно, система выдает сообщение об ошибке, представленное на рисунке 8.

The screenshot shows a game interface with a 3x3 grid. At the top, there is a label "Введите размеры поля (N):" followed by a text input containing "3". To the right of this input are two buttons: "Начать!" and "Заново!". Below the input is a label "Координаты хода:" followed by a text input containing "ску". To the right of this input is a button "Вперед!". The 3x3 grid is empty. At the bottom of the interface, a message reads "Данные некорректны, попробуйте еще!". A mouse cursor is visible near the bottom right of the grid.

Рисунок 8 – Сообщение об ошибке ввода

По окончании игры система выводит сообщение о победителе, представленное на рисунке 9.

The screenshot shows the same game interface as in Figure 8, but with a 3x3 grid containing game pieces. The pieces are 'X' and 'O'. The grid is as follows:

X	X	O
X	O	
O		

At the top, the label "Введите размеры поля (N):" is followed by a text input containing "3". To the right are buttons "Начать!" and "Заново!". Below the input is a label "Координаты хода:" followed by a text input containing "2,1". To the right of this input is a button "Вперед!". At the bottom of the interface, a message reads "Победитель Компьютер". A mouse cursor is visible near the bottom right of the grid.

Рисунок 9 – Информация о победителе