МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

Учреждение образования

«Гомельский государственный технический университет

имени П.О. Сухого»

Факультет автоматизированных и информационных систем

Кафедра «Информационные технологии»

направление специальности 1-40 05 01-12 Информационные системы и  
технологии (в игровой индустрии)

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

к курсовому проекту

по дисциплине «Объектно-ориентированное программирование»

на тему: «»

Исполнитель: студент группы ИТИ-22

Веровкин А.А.

Руководитель: заведующий кафедрой

Курочка К.С.

Дата проверки: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Дата допуска к защите: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Дата защиты: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Оценка работы: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Подписи членов комиссии

по защите курсового проекта: \_\_\_\_\_\_\_\_\_\_\_\_\_\_

Гомель 2019

Содержание

**Введение**

Первая игра появилась в далеком 1952-ом году, её автором считается Дуглас Энгельбард. Сама игра, по нынешним меркам, не представляла ничего особенного. Перестрелка двух игроков в космосе, однако именно она дала свет новому виду развлечений.

Распространение аркадные игры получили в 1970-х года, когда начали распространятся аркадные автоматы и игровые консоли.

В 1980 годах и в начале 90-х игровая индустрия перешла на новый этап. В связи с развитием персональных компьютеров и игровых консолей аркадные автоматы стали не такими популярными. Разработчики начали создавать новые механики отрисовки, а также появились первые трехмерные игры. Окончательно сформировалось разделение игры на жанры: файтинг, рпг, шутер и другие.

Сейчас игровая индустрия приносит неплохую прибыль. За один 2019 год общий доход игровой индустрии составил 150 миллиардов долларов. И эта цифра не максимальна. Игровые студии каждый год представляют нам новые игры, которые по степени своего реализма могут сравняться с реальностью. Или же с невероятной проработкой игрового мира, где каждая мелочь ощущается как реальная. Каждый год появляются сотни тысяч новых видео-летсплеев, гайдов и просто нарезок из игр, которые тоже приносят прибыль своим авторам. Все это говорит о том, что этот вид деятельности является очень прибыльным, а работа, связанная с разработкой игр, будет востребована ещё пару десятков лет.

1. **СРЕДСТВА РАЗРАБОТКИ ИГРОВЫХ ПРИЛОЖЕНИЙ**
   1. **Что такое “аркады”**

Аркада – жанр компьютерной игры, характеризующийся коротким, но весьма насыщенным и интенсивным игровым процессом. Временем расцвета аркадных игр считается период с конца 1970-х до середины 1980-х.

Первой аркадной игрой можно считать *Computer Space.* Разработана она была в 1971-ом году. Это была первая промышленная игра. Она использовала векторную графику (изображение строилось из линий).

В 1980-х аркадные игры начали становится более разнообразны, как относительно интерфейса управления, так и в плане тематики, графики и жанров. Использовались более реалистичные контроллеры, такие как двухместный космический корабль *Tail Gunner*.

С течением времени аркадные автоматы вытеснялись домашними системами, так как их графика начала обходить большинство аркадных игр, и таким образом к концу 1990-х аркадные автоматы покинули индустрию.

* 1. **Характерные особенность жанра “аркада”**

Классические аркады характеризуются несколькими свойствами.

Первое свойство – это игра на одном экране. Все действия игры происходят на одном экране. Это обусловлено исторически из-за технических возможностей машин тех времен. Порой игроделы выкручивались из данной ситуации просто сменяя экраны как будто это разные уровни. Также это влияло на геймдизайн. Игрок всегда мог видеть весь игровой мир и принимать решения, исходя из увиденного.

Второе свойство – бесконечная игра. Это можно посчитать ленью разработчиков, ведь они просто не хотят делать конец игре, однако это вовсе не так. Делая игру бесконечной игрок сам находит себе достижения, которые он должен достигнуть, будь это самое наибольшее количество собранных шариков, либо самое дальнее расстояние, которое он преодолел. Относительно геймдизайна в аркадах игрок никогда не выигрывал, и каждая игра заканчивалась поражением. В то же время, игры проектировались так, чтобы со временем игроку становилось все сложнее и сложнее. Таким образом игра предлагала также и бесконечную сложность. С появление домашних систем игроделы сменили политику с бесконечных игр на конечные. Ведь хорошо прописанный конец игр заставлял игроков желать продолжения.

Третье свойство ­– множество жизней. Имея несколько жизней неопытный игрок мог лучше понять некоторые механики и быстрее влиться в игру. Тем более, когда игрок лучше понимает игру, то вероятность того, что он вернется – очень высока. Так же данная особенность позволяла награждать игроков дополнительной жизнь за их успехи.

Четвертое свойство – игровой счет. За выполнение различный целей или задач игрок получает определенное количество очков в общий счет. После завершения игры игроку показывали конечное число очков. В некоторых автоматах присутствовали таблицы рекордсменов, куда попадали игроки с наибольшим количеством очков. В основном на результатах подсчетов игрок понимал насколько хорошо играл, несмотря на то, что выиграть невозможно.

Пятое свойство – быстрое обучение. Аркады изначально делались для того, чтобы любой мог подойти к автомату и буквально за пару минут научится играть. В аркадах нету такого, как например заучивание приемов путем нажатия определённых клавиш в определённой последовательности. Максимум что вам придется заучить – это на какие кнопки стрелять, а на какие двигаться.

Шестое и последнее свойство – отсутствие сюжета. Если игра бесконечная, то сюжет незачем. Играм жанра всегда требовалось, чтобы игроки быстро поняли что происходит – это научная фантастика, война, спорт или что-то ещё.

* 1. **Разновидность игр “аркада”**

Аркадные игры можно разделить на несколько видов. Такое разделение аркад объясняется тем, что некоторые игровые концепции возможно усложнить так, что они могут стать отдельным жанром или могут быть отнесены к другому жанру, или же эти концепции могут быть сильно упрощены из-за ориентированности на большую аудиторию или по иным причинам. Например, аркадные гонки и гоночные симуляторы являются воплощением одной игровой концепции: дать возможность игроку управлять автомобилем или другим транспортным средством. В основе первого лежит задача сделать максимально удобный и увлекательный игровой процесс, а в основе второго – стремление создать как можно более реалистичную модель управления транспортным средством, бросить вызов игроку.

Так, например, к виду аркадных игр относятся вышеупомянутые гонки. Цель проста ­– добраться до финиша быстрее твоего противника (либо противников). По пути следования игрок может получать какие-либо бонусы, которые могут либо ускорить его автомобиль, либо как-то воздействовать с автомобилем противника, чтобы получить преимущество над ним. Ну либо же в игре не будет никаких бонусов. Тут все будет зависеть от игрока и его навыка обращения с виртуальным автомобилем.

Следующей разновидность будут классические аркады. Концепт таких игр в основном заставляет игрока пробежать по уровню за наименьшее время и при этом собрать наибольшее количество очков. При этом на уровне могут присутствовать потайные места с дополнительными бонусами.

Скроллеры. Довольно простые аркадные игры. Экран в таких играх в основном движется в одном направлении на протяжении всей игры. Прекрасным примером можно считать игру “*Shoot ‘em up”* представленной на рисунке 1.1.



Рисунок 1.1 – Игра “*Shoot ‘em up”*

Скроллеры так же разделяются на два вида: вертикальны скроллеры (экран движется сверху вниз) и горизонтальные скроллеры (экран движется справа налево).

Далее у нас идут файтинги. Главным экраном в файтингах считается арена, где сражаются два персонажа. В файтингах игроки, используя разные удары, захваты и комбинации приемов пытаются победить друг друга. Так же в подобных видах аркад предусмотрен выбор персонажа перед поединком. Каждый персонаж имеет свой уникальный паттерн ударов, особые приемы и особенные комбинации. Но какого-то определённого сильного персонажа в таких играх нет. Разработчики изначально делают их равными друг другу, чтобы не возникало разногласий по поводу того, кто “честно” выиграл. Представителями такого жанра являются серии игр “*Mortal Combat”* и “*Street Fighter”*.

Платформеры. Простые игры с незамысловатой логикой действий. Игроку нужно пройти уровень с препятствиями и врагами. Врагов он может как перепрыгивать, так и уничтожать, например, прыжок на черепаху в “*Mario”*. Основным элементом игры является перемещение игрока по платформам и/или лестницам.

Виртуальный тир. Данный жанр зародился на игровых автоматах, а далее перекочевал как на персональные компьютеры, так и на игровые консоли, но уже с улучшениями. Геймплей заключается в том, что игроку нужно отстреливать постоянно прибывающих врагов, используя специальный контроллер, выполненный в виде пистолета или другого огнестрельного оружия. Игрок как бы “едет по рельсам” и не может управлять камерой и своими движениями в игре. Все движения камерой и перемещения по уровню остаются за разработчиками, поэтому порой всю игру снимают на видеокамеру, в определенных местах подставляя разные варианты видео отрывков.

Так же есть и другие виды аркад, но они либо просто не распространены, либо же имеют какие-либо особенности, которые не дают приписать их к аркадам, хоть они на них очень похожи.

* 1. **Прибыльность и актуальности разработки аркадных игр**

Аркады на данный момент разрабатывают либо разработчики мобильных игр, либо просто любители ну или люди, изучающие разработку игр в учебных целях.

На нынешний момент рынок мобильных игр ничем не уступает рынку на том же персональном компьютере и игровых консолях, а порой даже выигрывает. Залог такой популярности заключается в том, что мобильные игры – это удобно. Телефон всегда под рукой, и можно зайти в игру в любой свободный момент, когда для персонального компьютера или игровой консоли требуется ещё множество побочных элементов по типу экрана, множества проводов и питания от сети (в данный моменты мы не рассматриваем ноутбуки). Так же мобильные игры в большинстве своем бесплатны и не требуют первоначальных вложений. Разработчики в таких бесплатных играх получают прибыль по средству внутри-игровых покупок, а также рекламы. Так как разработчикам мобильных игр нужно охватить наибольшую публику им нужен простой незамысловатый игровой процесс, который быстро втягивает и долго не отпускает. Поэтому они и выбирают аркады.

Разработчики-любители занимаются разработкой аркадных игр из-за простоты их написания, распространенности игровых платформ, на которые строятся игры, а также низких требований к знаниям программирования. Таким разработчикам проще реализовывать свои идеи, из-за отсутствия дедлайнов разработки. В основном свои проекты такие разработчики заливают на разные сайты на бесплатной основе и без какой-либо монетизации (внутриигровые покупки и реклама). Такие проекты в основном рассчитаны на получение хоть какой-то огласки автора и редко становятся полноценными играми, в основном они являются дешевой копией популярной игры.

Что говорить про разработку игр с целью научиться созданию простых игр, то тут прибыли никакой не будет. Разработка аркадной игры будет неплохим первым опытом, но не более. Если выложить свой проект в сеть, то успеха ждать не стоит, т.к. ежедневно выходят сотни подобных проектов.

* 1. **Обзор графических библиотек *DirectX* и *OpenGL***

Для разработки нынешних игр используются две популярные графические библиотеки. *OpenGL* и *DirectX* являются ведущими графическими библиотеками на рынке.

*OpenGl* – это программный интерфейс для работы с двумерной и трехмерной графикой. Библиотека *OpenGl* разработана в виде независимого интерфейса и может быть использована в любом аппаратном обеспечении. Интерфейс содержит около 250 команд для отображения объектов и произведения графических операций. Примерами таких операций являются: создание точек, линий, изображений, полигонов, побитовых карт, позиционирование объектов в трехмерном пространстве, рисовка текстур, цвета, света. С помощью библиотеки можно создать эффект наложения объектов друг на друга, где часть одной фигуры будет закрывать другую. Это происходит посредством удаления части объекта.

Также в *OpenGl* существует возможность разделения работы программы на два компьютера. Например, код программы может запускаться на одном устройстве, а отображение графики происходит на другом. Таким образом, первый компьютер является клиентом, а второй сервером. Пересылка команд между клиентом и сервером имеет один и тот же формат, поэтому программа может работать по сети при абсолютно разных компьютерах.

При работе с *OpenGl* все данные сохраняются в списки (*display* *lists*). Визуализация – создание компьютером изображений – происходит посредством построения геометрических примитивов, которые задаются своими вершинами. Базисные функции позволяют построить параметрические поверхности и кривые. Это нужно для того, чтобы узнать координаты текстуры, цвета, нормаль поверхности и значения координат в пространстве. Информация о цвете содержится в памяти в виде *bitplain’*ов – битовых поверхностей, где на один экранный пиксель приходится один бит информации. Битовые поверхности хранятся в буфере кадра, который обладает информацией для контроля цвета и насыщенности пикселей на мониторе. Рисовка примитивов происходит после прохождения геометрическими данными повершинных операций.

К основным возможностям *OpenGl* можно также отнести:

– видовые и модельные преобразования – с их помощью созданные фигуры можно перемещать, вращать в пространстве, изменять их форму и управлять положением камеры;

– двойная буферизация – позволяет избавиться от дефектов отображения объектов во время их движения. Примитивы рисуются сначала в невидимом буфере и только после полной готовности кадра отображаются на экране;

– сглаживание – изменяет интенсивность и цвет пикселей на экране, что приводит к более линейному отображению объектов и устраняет видимость ступенчатости растровых дисплеев;

– растровые примитивы – кроме геометрических примитивов можно использовать *bitmap* – битовый массив и *image* – образ, картинку;

– освещение – возможность установки источника света, управление его интенсивностью, расположением, цветом и прочих характеристик.

К преимуществам *OpenGl* можноотнести его независимость от платформ и производителей. Программа с использованием этой библиотеки сможет работать на операционных системах *Windows*, *MacOS*, *Linux*, *Solaris* и других. *OpenGl* использовался даже в написании первых игр для игровых консолей *Xbox* 360 и *Playstation* 2. Широкий выбор дополнений и спецификаций, постоянные обновления библиотеки предоставляют возможность написания программ на таких языках программирования, как: *С*, *С++*, *Fortran*, *Java*, а также *C#*, после подключения специальной надстройки *SharpGL* или *OpenTK*.

*DirectX* – это программный интерфейс для создания продвинутой двумерной и трехмерной графики, разработанный компанией *Microsoft* для операционной системы *Windows*. Интерфейс *DirectX* позволяет разработчикам компьютерных игр просто и быстро формировать на экране сложные элементы графики и мультипликации. Он предоставляет возможности *Z*-буферизации, пространственного сжатия, альфа-смешения, смешивания цветов и текстур, отсечения фигур и плоскостей, создания атмосферных эффектов и программирования шейдеров и использования звука. Интеграция *DirectX* с другими разработками компании позволяет отображать видео, производить 3*D*-рендеринг и 2*D*-рендеринг в плоскостях наложения.

Основными компонентами, входящими в *DirectX,* являются: *DirectX Graphics*, содержащий библиотеки *Direct3D* и *DirectDraw* для работы с графикой; *DirectShow*, отвечающий за управление захвата и воспроизведение мультимедиапотоков; *DirectInput* для работы с устройствами ввода; *DirectSound* и *DirectMusic*, позволяющий работать с звуковыми потоками и форматами файлов *MIDI*; *DirectPlay* для разработки мультипользовательских приложений и *DirectX Media Objects* для поддержки потоковых объектов.

* 1. **Сравнение графических библиотек *DirectX* и *OpenGL***

*DirectX* предназначен для виртуализации трехмерных аппаратных интерфейсов и освобождает разработчика от использования графического оборудования. *OpenGl* разработан как аппаратно-ускоренная система 3*D*-рендеринга, которую можно эмулировать программно. Это означает, что интерфейсы разработаны для двух разных способов мышления.

Одним из функциональных различий работы этих технологий является то, как они управляют аппаратными ресурсами. *DirectX* ожидает это от приложения, а *OpenGl* делает это сам во время выполнения программы. Данное отличие упрощает разработку на *OpenGl*, но при этом увеличивает сложность создания реализации или драйвера, которые будут показывать хорошие результаты выполнения. При работе с *DirectX* разработчик должен сам управлять аппаратными ресурсами, но реализация происходит намного проще и появляется возможность распределения ресурсов наиболее эффективным способом для своего приложения.

К отличиям двух технологий можно также отнести механизм расширений *OpenGL*. Он заключается в возможности любого драйвера иметь свои собственные расширения к интерфейсу, таким образом, вводятся новые функции, такие как режимы наложения, новые способы для передачи данных в графические процессоры или другие параметры обтекания текстур. Это позволяет быстро получать новые функции, но также может привести к путанице, если разные производители реализуют одинаковые расширения с разными интерфейсами. Многие из этих расширений периодически стандартизируются.

С другой стороны, *DirectX* определяется только одним производителем (*Microsoft*), что приводит к более согласованной реализации интерфейса, но запрещает доступ к функциям, специфичным для конкретного производителя. Например, технология *NVIDIA UltraShadow* недоступна в стандартном интерфейсе *DirectX*.Когда в видеокарты добавили поддержку пиксельных шейдеров (в *OpenGL* это фрагментные шейдеры), *DirectX* предоставил один стандарт *Pixel Shader 1.1*, с которым ряд графических процессоров имеют совместимость. В *OpenGL* те же функции стали доступны через различные пользовательские расширения, но немного позднее.

За исключением нескольких незначительных функциональных различий, эти две технологии обеспечивают практически одинаковый уровень функциональности. Производители аппаратного и программного обеспечения обычно быстро реагируют на изменения в *DirectX*, в то время как новые функции *OpenGl* обычно реализуются сначала поставщиками, а затем применяются к стандарту.

На практике, главными отличиями *OpenGl* и *DirectX* являются кроссплатформенность у *OpenGl* ибиблиотеки для звука и видео у *DirectX*.

В качестве одного из основных критериев игры была выделена работа с двумерными объектами. Для корректной работы интерфейса технология *DirectX* требует большего количества подключенных библиотек, чем *OpenGL*, что сказывается на скорости работы программы и отображении графических объектов.

Так как при подготовке к разработке игрового приложения управление звуковыми и видео эффектами не было включено в его функционал, «плавность» работы игры является важным показателем, а кроссплатформенность позволяет охватить большую аудиторию пользователей, после изучения возможностей двух технологий и их сравнения, в качестве графического средства разработки был выбран интерфейс *OpenGL*.

1. **ПРОГРАММНАЯ РЕАЛИЗАЦИЯ ПРИЛОЖЕНИЯ “КЛАД”**
   1. **Условия для реализуемого приложения**

Цель курсового проекта – разработка игрового приложения “Клад” с возможностью игры двух пользователей друг против друга. В процессе разработки использовать язык программирования *C#* и использовать интерфейс программирования приложений Windows Form. Использовать паттерн программирования “Фабрика” для создания различных призов в лабиринте и паттерн программирования “Декоратор” для изменения характеристик лабиринта. Объекты в приложении должны быть отрисованы с помощью спрайтовой графики и средств *OpenGL*.

* 1. **Подготовка исходных данных**

Создание программы производится в среде разработки *Visual Studio 2019*. Для работы с функциями OpenGL в Windows Form приложении были загружены необходимые графические библиотеки. Так как программа пишется на языке C#, а библиотеки OpenGL написаны на языке C++, была загружена библиотеки OpenTK, позволяющая работать с OpenGL на языке C#. OpenTK содержится основную оболочка OpenGL, но только с одним исключением ­– все названия методов OpenGL и иерархия классов были адаптированы под язык C#.

Игровое приложение имеет несколько карт, но загружается только одна из файла bmp. Карта выбирается случайно во время загрузки приложения.

* 1. **Проектирование архитектуры ПО**

В процессе проектирования архитектуры игрового приложения был использован процесс декомпозиции. Это универсальный научный метод решения сложных задач, основанный на использовании структуры задачи и ее разбиении на более простые подзадачи [6, 30]. Подзадачи проектировались и разрабатывались, а после были объединены в общее решение. Каждый элемент декомпозиции является уникальным, что позволило сформировать и отличить составляющие объекта одну от другой. Элементы, полученные в результате декомпозиции, связаны с функциями отдельных подзадач единой задачи.

Так, перед началом написания программы при помощи процесса декомпозиции была создана обобщенная функциональная схема приложения, описывающая последовательность работы ее функций. Данная схема представлена на рисунке 2.1:

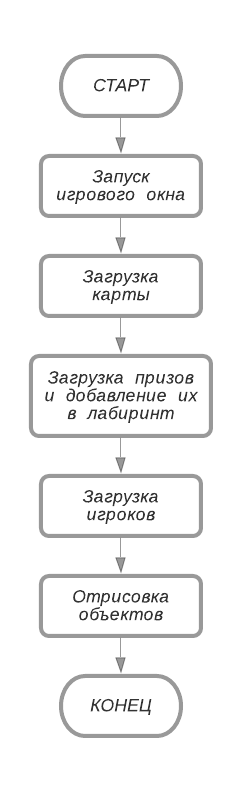


Рисунок 2.1 – Схема приложения

Процесс декомпозиции позволил добиться решения задач с минимальными затратами и максимальной гибкостью. Каждая подсистема проходила тестирование, что позволило устранить возникновение ряда ошибок на начальных этапах разработки.

* 1. **Функциональная схема приложения и иерархия классов**