

Федеральное государственное автономное  
образовательное учреждение  
высшего образования  
«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»  
Институт космических и информационных технологий  
институт  
Вычислительная техника  
кафедра

УТВЕРЖДАЮ  
Заведующий кафедрой

\_\_\_\_\_ А.И. Легалов

подпись      инициалы, фамилия

« \_\_\_\_ » \_\_\_\_\_ 2017 г.

**БАКАЛАВРСКАЯ РАБОТА**

09.03.01 Информатика и вычислительная техника  
код – наименование направления

Интеграция технологий виртуальной реальности в систему интерактивного  
обучения  
тема

Пояснительная записка

Руководитель	_____	<u>доцент, к.т.н.</u>	<u>А.В. Редькина</u>
	подпись, дата	должность, ученая степень	инициалы, фамилия
Выпускник	_____		<u>П.С. Мамаев</u>
	подпись, дата		инициалы, фамилия
Нормоконтролер	_____		<u>В.И. Иванов</u>
	подпись, дата		инициалы, фамилия
Консультант	_____		<u>В.С. Будников</u>
	подпись, дата		инициалы, фамилия

Красноярск 2017

## СОДЕРЖАНИЕ

ВВЕДЕНИЕ.....	4
1 Анализ предметной области .....	6
1.1 Анализ существующих систем обучения в виртуальной реальности .....	6
1.1.1 Система InMind VR.....	6
1.1.2 Система Unimersiv .....	7
1.1.3 Система VRtisan .....	8
1.1.4 Сравнение существующих систем .....	9
1.2 Классификация систем распознавания речи .....	10
1.2.1 Признаки классификации систем распознавания речи.....	11
1.3 Способы интеграции программных систем .....	12
1.3.1 Топология взаимодействия интегрируемых систем.....	12
1.3.2 Методы интеграции .....	15
1.4 Выводы.....	17
2 Архитектура разрабатываемой системы интерактивного обучения .....	18
2.1 Выбор устройства виртуальной реальности.....	22
2.1.1 Устройство виртуальной реальности Oculus Rift .....	22
2.1.2 Устройство виртуальной реальности PlayStation VR.....	23
2.1.3 Устройство виртуальной реальности Samsung Gear VR.....	24
2.1.4 Обоснование выбора Samsung Gear VR.....	25
2.2 Выбор системы распознавания речи .....	26
2.2.1 Система Google Voice Recognition .....	26
2.2.2 Система Julius .....	27
2.2.3 Система PocketSphinx .....	27
2.2.4 Сравнение систем распознавания речи.....	27
2.3 Средства программной разработки .....	28
3 Реализация интеграции технологий виртуальной реальности в систему интерактивного обучения.....	31
3.1 Интеграция системы распознавания PocketSphinx.....	32
3.2 Интеграция подсистемы загрузки нового контента .....	35
3.3 Интеграция внешних устройств .....	36

3.4 Модель программы .....	38
4 Руководство для реализации интеграции технологий виртуальной реальности в проект Unity .....	39
ЗАКЛЮЧЕНИЕ .....	43
СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ .....	44
ПРИЛОЖЕНИЕ А Сценарий .....	47
ПРИЛОЖЕНИЕ Б Листинг разработанного программного обеспечения .....	48

## ВВЕДЕНИЕ

Современные системы интерактивного обучения позволяют повысить качество обучения, делая учебный материал понятным и доступным, а образовательный процесс качественным и интересным.

Они появились не так давно, но уже успели стать популярными. Область применения интерактивных технологий настолько широка, что уже трудно представить сферу жизни, где эти новинки прогресса не успели бы себя проявить. Не стала исключением и сфера образования, интерактивные системы обучения широко применимы в учебных заведениях.

Интерактивность (в контексте информационной системы) – это возможность информационно-коммуникационной системы по-разному реагировать на любые действия пользователя в активном режиме. Интерактивные технологии являются непременным условием для функционирования высокоэффективной модели обучения, основной целью которой является активное вовлечение каждого из учеников в образовательный и исследовательский процессы [1].

В настоящее время все большее количество учебных заведений оснащает свои аудитории и классы интерактивными досками. Их использование во время занятия дает учащимся возможность увидеть реалистичные двух и трех мерные модели объектов изучения, наблюдать за их изменениями и управлять ими, просто касаясь доски руками. Подобная технология позволяет реализовывать принципы развивающего обучения на практике. А с увеличением спроса на виртуальную реальность по всему миру и в России в частности, интерактивные системы обучения могут получить новый виток развития.

Виртуальная реальность – это созданный техническими средствами мир (объекты и субъекты), передаваемый человеку через его ощущения: зрение, слух, обоняние, осязание и другие. Виртуальная реальность имитирует как воздействие, так и реакции на воздействие [2].

Системы виртуальной реальности включают множество программ и устройств: датчики положения головы, беспроводные контроллеры управления, видеозэкран и т.д. Для организации их взаимодействия разработаны программы, объединяющие их в единую систему.

Интеграция – это процесс объединения частей в целое [3].

Интеграция программных систем и подсистем – это обмен данными между системами с возможной последующей их обработкой.

Интеграция устройств и программ в систему виртуальной реальности осуществляется благодаря подсистемам управления, которые организуют взаимодействие между ними и системой на программном уровне.

Целью данной работы является повышение эффективности процесса обучения и мотивации обучающихся, посредством интеграции технологий виртуальной реальности: системы распознавания речи, беспроводного контроллера, устройства виртуальной реальности, системы загрузки нового контента – в систему интерактивного обучения.

Для достижения поставленной цели необходимо решить следующие задачи:

- проанализировать предметную область;
- проанализировать существующие аналоги систем;
- разработать программный продукт для интеграции технологии виртуальной реальности в систему интерактивного обучения и продемонстрировать его работу;
- написать документацию для реализации интеграции технологий виртуальной реальности в проект Unity.

## 1 Анализ предметной области

### 1.1 Анализ существующих систем обучения в виртуальной реальности

Существуют системы обучения в виртуальной реальности как в России, так и за рубежом, но их основными недостатками являются высокая стоимость либо не качественное исполнение, а также высокие требования к вычислительным мощностям устройств. Даже при скудном функционале системы обучения в виртуальной реальности могут обладать высокой стоимостью. Существует так же ряд бесплатных и открытых систем, но большинство из них находятся на ранней стадии разработки и нуждаются в серьезной доработке. Далее мы рассмотрим ряд систем обучения в виртуальной реальности.

#### 1.1.1 Система InMind VR

InMind VR – интерактивное приложение для изучения нейронов человеческого мозга, разработанное для устройств виртуальной реальности с помощью интегрированной среды разработки Unreal Engine [4]. Процесс работы в InMind VR представлен на рисунке 1.1.

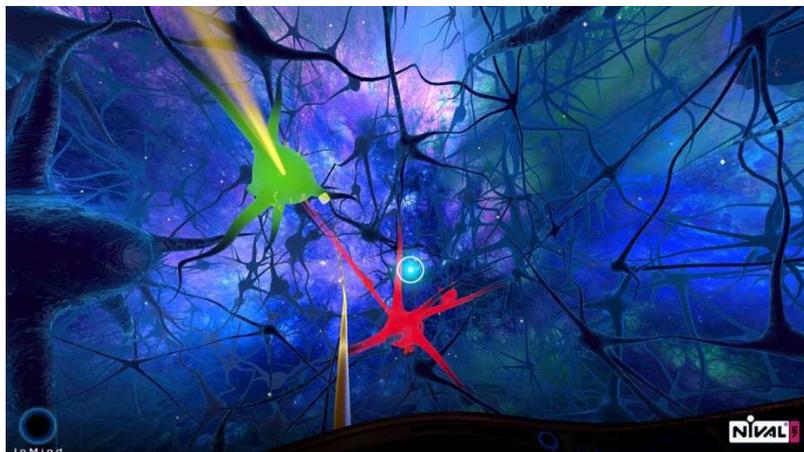


Рисунок 1.1 – Процесс работы в InMind VR

Приложение выпущено российской компанией Nival в начале 2015 года для устройства виртуальной реальности Oculus Rift. Система InMind VR является образовательным приложением для самообучения.

На данный момент приложение полностью бесплатно и доступно в магазинах приложений для устройств виртуальной реальности Google Cardboard и Oculus Rift.

Система InMind VR первая система обучения в виртуальной реальности, разработанная в России. Имеет ограниченный функционал и минимум интерактивности.

### 1.1.2 Система Unimersiv

Unimersiv – это приложение для обучения в виртуальной реальности, разработанное с использованием интегрированной среды разработки Unity. Оно включает в себя уровни различной тематики, такие как: исследование Стоунхенджа, строения Международной космической станции или внутреннего строения человека и т.д. [5]. Процесс работы в Unimersiv представлен на рисунке 1.2.



Рисунок 1.2 – Процесс работы в Unimersiv

Приложение выпущено британской компанией Unimersiv в 2016 году для устройства виртуальной реальности Oculus Rift. Unimersiv является приложением для обучения в образовательных учреждениях, таких как школы и вузы.

Разработчики Unimersiv стремятся поддерживать школы, колледжи и другие учебные центры. Педагоги могут зарегистрироваться в качестве представителя образовательного учреждения, чтобы получить особые льготы.

Само приложение доступно бесплатно, но отдельные уровни нужно приобретать в формате дополнительных покупок внутри приложения. Стоимость колеблется: начиная от £ 1,99 до 3,09 фунтов стерлингов.

Unimersiv поддерживает большинство современных устройств виртуальной реальности, такие как:

- Samsung Gear VR;
- Oculus Rift;
- Google Cardboard.

Система Unimersiv первая система виртуальной реальности, предназначенная для обучения в образовательных учреждениях, но в ней отсутствует поддержка русского языка и возможность интерактивного взаимодействия с системой.

### **1.1.3 Система VRtisan**

VRtisan – это интерактивное приложение для проектирования дизайна помещений, созданное для устройств виртуальной реальности с использованием интегрированной среды разработки Unreal Engine [6]. Процесс работы в VRtisan представлен на рисунке 1.3.

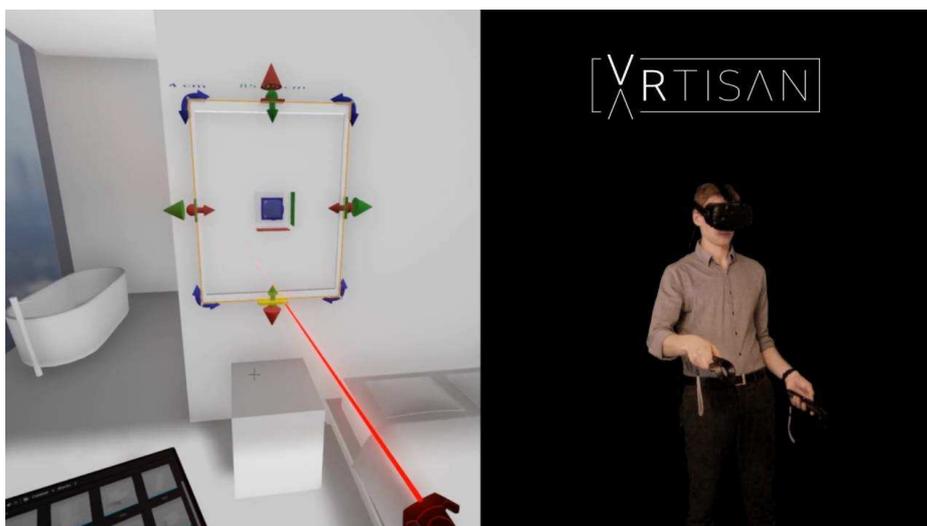


Рисунок 1.3 – Процесс работы в VRtisan

Приложение выпущено британской компанией Artisan в 2016 году для устройства виртуальной реальности HTC Vive. VRtisan является приложением для проектирования дизайна помещений в виртуальной реальности.

Во время работы приложения пользователь находится в виртуальной комнате и с помощью двух манипуляторов создает перегородки и встроенную мебель, изменяя их пропорции. Приложение позволяет выбирать материалы и подгружать библиотеки предметов, моментально проверять несколько решений и тестировать эргономику помещений.

Распространяется приложение по персональной лицензии, которую необходимо приобрести на сайте проекта.

На данный момент VRtisan поддерживает только устройство виртуальной реальности HTC Vive.

Система VRtisan обладает высокой стоимостью, а с учетом цены устройства виртуальной реальности, делает данное приобретение очень дорогим.

#### **1.1.4 Сравнение существующих систем**

При разработке систем поддержки обучения важными являются следующие критерии: низкая цена, интерактивность, поддержка русского языка,

дружественный интерфейс, возможность расширения системы, контроль со стороны преподавателя за процессом обучения. Для сравнения существующих систем была составлена сравнительная таблица 1.

Таблица 1 – Сравнение существующих систем

Критерии сравнения	InMind VR	Unimersiv	VRTisan
Низкая цена	Да	Да	Нет
Интерактивность	Да	Нет	Да
Поддержка русского языка	Да	Нет	Нет
Дружественный интерфейс	Нет	Да	Да
Возможность расширения	Нет	Да	Да
Контроль со стороны преподавателя	Нет	Нет	Нет

В ходе сравнения существующих систем было выявлено, что среди них нет той системы, которая бы полностью удовлетворяла всем критериям для организации процесса обучения в виртуальной реальности. Поэтому будет разработана система интерактивного обучения с использованием технологии виртуальной реальности, удовлетворяющая всем критериям.

## 1.2 Классификация систем распознавания речи

Распознавание речи – процесс преобразования речевого сигнала в цифровую информацию (например, текстовые данные) [7].

На сегодняшний день, под понятием “распознавание речи” скрывается целая сфера научной и инженерной деятельности. В общем, каждая задача распознавания речи сводится к тому, чтобы выделить, классифицировать и

соответствующим способом отреагировать на человеческую речь из входного звукового потока. Это может быть и выполнение определенного действия на команду человека, и выделение определенного слова – маркера из системы для голосового ввода текста.

### **1.2.1 Признаки классификации систем распознавания речи**

Каждая система распознавания речи имеет некоторые задачи, которые она призвана решать и комплекс подходов, которые применяются для решения поставленных задач. Рассмотрим основные признаки, согласно которым можно классифицировать системы распознавания человеческой речи и то, как этот признак может влиять на работу системы:

- Размер словаря [8].

Очевидно, что чем больше размер словаря, который заложен в систему распознавания, тем больше частота ошибок при распознавании слов системой. Например, словарь из десяти цифр может быть распознан практически безошибочно, тогда как частота ошибок при распознавании словаря в сто тысяч слов может достигать сорока пяти процентов. С другой стороны, даже распознавание небольшого словаря может давать большое количество ошибок распознавания, если слова в этом словаре очень похожи друг на друга.

- Дикторозависимость или дикторонезависимость системы [8].

По определению, дикторозависимая система предназначена для использования одним пользователем, в то время как дикторонезависимая система предназначена для работы с любым диктором. Дикторонезависимость – труднодостижимая цель, так как при обучении системы, она настраивается на параметры того диктора, на примере которого обучается. Частота ошибок распознавания таких систем обычно в 3-5 раз больше, чем частота ошибок дикторозависимых систем.

- Раздельная или слитная речь [8].

Если в речи каждое слово разделяется от другого участком тишины, то говорят, что эта речь – раздельная. Слитная речь – это естественно произнесенные предложения. Распознавание слитной речи намного труднее в связи с тем, что границы отдельных слов не четко определены и их произношение сильно искажено смазыванием произносимых звуков.

– Назначение [8].

Назначение системы определяет требуемый уровень абстракции, на котором будет происходить распознавание произнесенной речи. В командной системе (например, голосовой набор в сотовом телефоне) скорее всего, распознавание слова или фразы будет происходить как распознавание единого речевого элемента. А система диктовки текста потребует большей точности распознавания и, скорее всего, при интерпретации произнесенной фразы будет полагаться не только на то, что было произнесено в текущий момент, но и на то, как оно соотносится с тем, что было произнесено до этого. Также, в системе должен быть встроен набор грамматических правил, которым должен удовлетворять произносимый и распознаваемый текст. Чем строже эти правила, тем проще реализовать систему распознавания и тем ограниченной будет набор предложений, которые она сможет распознать.

### **1.3 Способы интеграции программных систем**

#### **1.3.1 Топология взаимодействия интегрируемых систем**

Существует два способа организации связей между интегрируемыми системами. Первый – это прямое взаимодействие интегрированных систем по принципу «каждая с каждой», или «точка-точка», которая представлена на рисунке 1.4. Второй – взаимодействие через центральный узел, подобную звездообразную архитектуру, обычно называют «хаб + спицы», которая представлена на рисунке 1.5. Топология не зависит от физической архитектуры

информационной системы, а определяет логические маршруты взаимодействия и передачи данных между интегрированными системами [9].

При способе «точка-точка» интегрированные системы взаимодействуют напрямую друг с другом. Преимущества данного способа – это простота и отсутствие необходимости в дополнительном программном обеспечении. Но, имеются и недостатки. Во-первых, интегрированные приложения должны взаимодействовать друг с другом, используя одинаковые методы взаимодействия и форматы вызова данных. При модификации одного из приложений (если оно привело к изменению интерфейса взаимодействия данного приложения) приходится модифицировать или перенастраивать все интегрированные с ним системы. Во-вторых, в информационной системе возникает слишком много связей, каждую из которых нужно контролировать и поддерживать в работоспособном состоянии.

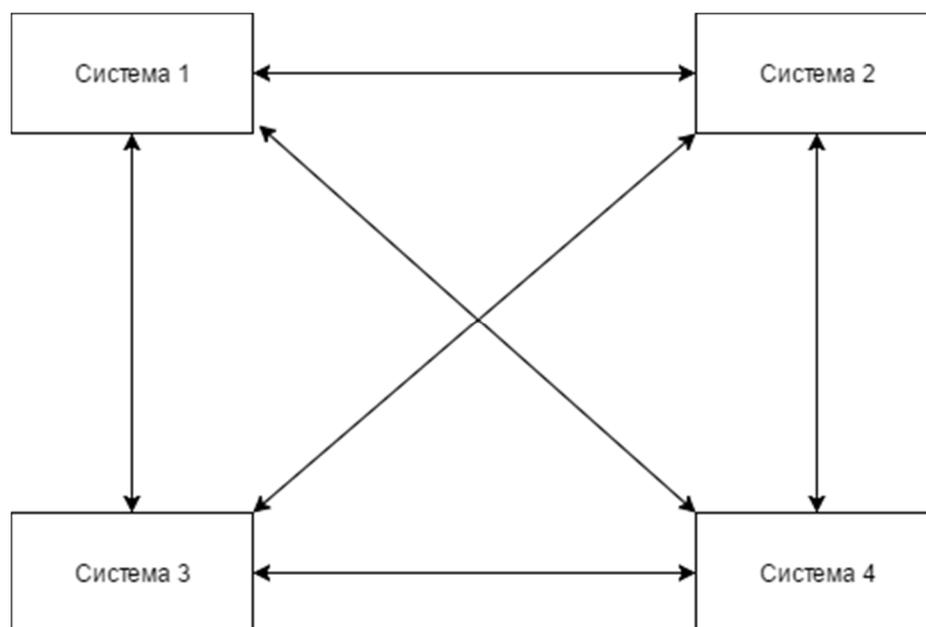


Рисунок 1.4 – Интеграция по типу «точка-точка»

В случае если взаимодействующих приложений много, стоимость поддержки, интегрированной таким образом информационной системы становится слишком высокой. Тем не менее подход «точка-точка» обширно

применяется. Это происходит, как правило, в тех случаях, когда при взаимодействии конкретных приложений необходимо передавать большие объемы данных или обеспечивать нормированное время взаимодействия, а кроме того в случае если используемые приложения имеют встроенные средства взаимодействия (это зачастую происходит при внедрении нескольких систем от одного поставщика, а кроме того в случае если при разработке заказных программных систем или внедрении новых к ним изначально предъявляется требование по взаимодействию с уже имеющимися системами).

Связь согласно виду «точка-точка» формирует очень большое количество взаимодействий и потребует согласования интерфейсов и форматов данных между взаимодействующими приложениями. Данные недостатки призвана найти решение архитектура взаимодействия, в которой все приложения напрямую объединены только лишь с основным узлом, решающим следующие задачи:

- организация маршрутизации взаимодействия между интегрированными приложениями;
- преобразование форматов файлов и данных;
- обеспечение взаимодействия приложений с применением различных методов и протоколов взаимодействия.

С помощью введения промежуточного звена, снижается количество взаимосвязей между приложениями, устраняются непосредственные связи, а система интеграции становится наиболее эластичной и недорогой в эксплуатации. В случае если изменяется одно из интегрированных приложений, то – при условии правильно спроектированной системы интеграции – необходимо изменять только одну связь, между данным приложением и хабом.

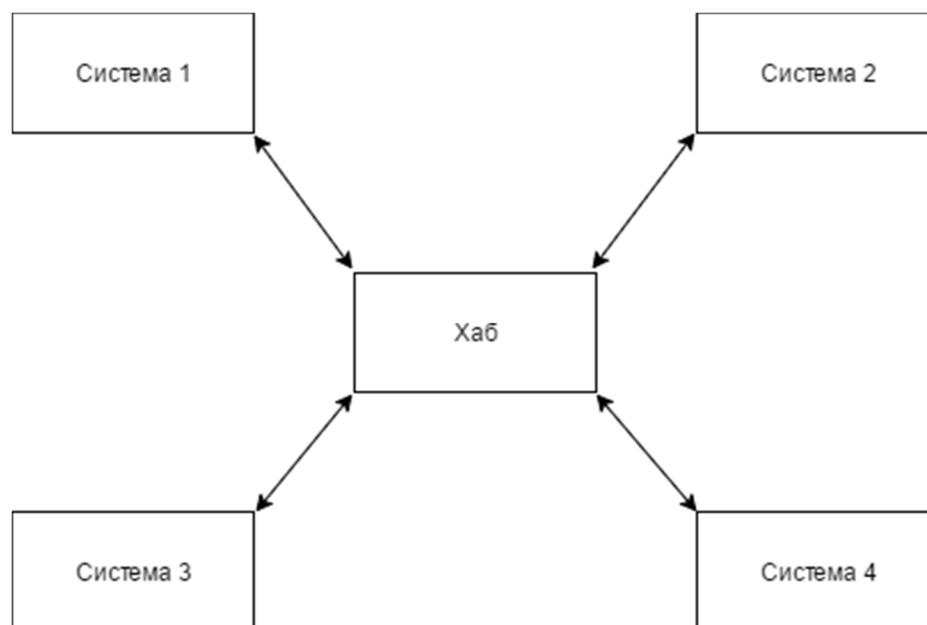


Рисунок 1.5 – Интеграция по типу «хаб + спицы»

Недостатком топологии «хаб + спицы» является высокая цена приобретения и трудность освоения программного инструментария, играющего роль хаба, а кроме того недостаток экспертов, обладающих опытом использования аналогичных программных средств.

### 1.3.2 Методы интеграции

Для взаимодействия приложений как правило используют такие методы, как обмен файлами, общая база данных, удаленный вызов и асинхронный обмен сообщениями [10].

Обмен файлами, самый известный метод организации взаимодействия. Это связано с сравнительной простотой реализации, а кроме того существованием стандартных форматов обмена. К примеру, значительная доля корпоративных информационных систем позволяет загружать и выгружать файлы, например, в формате CSV (Comma-Separated Values – «поля, разделенные запятыми»). Однако у данного способа есть и недостатки, в случае если необходимо оперировать сложными структурами, то простые форматы обмена

уже не пригодны. Образующиеся в таких случаях специализированные форматы файлов обязаны «понимать» взаимодействующие системы, что ведет к строгой зависимости систем друг от друга. Этот недостаток как правило решают утилитами конвертации данных. Кроме того, обмен файлами предполагает участие человека – кто-то должен выгрузить файл, скопировать его на другой компьютер, загрузить. Но в случае если интегрируемые методом обмена файлы системы имеют возможность автоматической загрузки/выгрузки (к примеру, по расписанию), в таком случае этот способ дает возможность создать полностью автоматизированное решение, которое вследствие своей простоты обладает высокой надежностью и пропускной способностью.

Метод общая база данных очень прост – ряд информационных систем или приложений пользуются одной базой данных. Основной недостаток данного метода – это то, что связь между интегрированными приложениями настолько тесная, что возникают случаи, когда невозможно определить границу между ними (как правило так интегрируются продукты одного производителя). Но слишком тесная связь превращает набор интегрированных приложений в монолит, в «суперсистему», отдельные части которой с трудом поддаются самостоятельной модернизации и замене. С этим борются, применяя механизмы серверов баз данных (представления данных, промежуточные таблицы и т.п.), однако не всегда продуктивно.

Метод удаленного вызова процедур. Этот метод позволяет программному коду, который выполняется на одном компьютере, вызывать код на другом. Основной недостаток удаленного вызова – требование работоспособности абсолютно всех задействованных приложений в момент взаимодействия.

Асинхронный обмен сообщениями, единственный из перечисленных выше методов, который создавался специально для интеграции информационных систем. Идея очень проста и похожа на работу электронной почты. Если приложению А необходимо вызвать, то или иное действие в приложении Б, то оно создает соответствующее сообщение с данными и инструкциями и отправляет его с помощью системы доставки сообщений. Термин

«асинхронный» обозначает, что приложение А не должно ждать, пока сообщение дойдет до Б, будет обработано, сформирован ответ и т.д. Сообщение гарантированно доставляется с помощью механизма очередей сообщений, который убирает с интегрированных систем заботу о надежности сети передачи данных, работоспособности взаимодействующих систем в конкретные моменты времени и т.д.

#### **1.4 Выводы**

При интеграции систем возникает множество проблем, так как порождаемая система становится более сложной, требующей согласования разнородных данных, единства взглядов разработчиков, реализующих предметно-ориентированные компоненты системы, на принципы их разработки, отладки, развертывания и функционирования.

Интеграция существующих и разрабатываемых систем, выявление общих принципов взаимодействия и построения предметно-ориентированных компонентов позволят упростить разработку ПО для поддержки процесса обучения.

Интеграция подсистем и устройств в систему интерактивного обучения, будет происходить по гибридной топологии, основанной на топологиях «хаб + спицы» и «точка-точка». Что позволяет использовать различные методы взаимодействия между подсистемами и устройствами для оптимизации работы определенных систем. Взаимодействие между интегрированными подсистемами и устройствами будет осуществляться методами: асинхронный обмен сообщениями, удаленный вызов процедур и общая база данных.

Интегрируемая подсистема распознавания речи должна обладать параметрами: дикторонезаисимость, большой словарь и возможность распознавания слитной речи.

## 2 Архитектура разрабатываемой системы интерактивного обучения

Разрабатываемая система работает по клиент-серверной архитектуре и состоит из 5 основных элементов: сервер, персональный компьютер, мобильное устройство, устройство виртуальной реальности и беспроводной контроллер. Архитектура разрабатываемой системы представлена на рисунке 2.1.

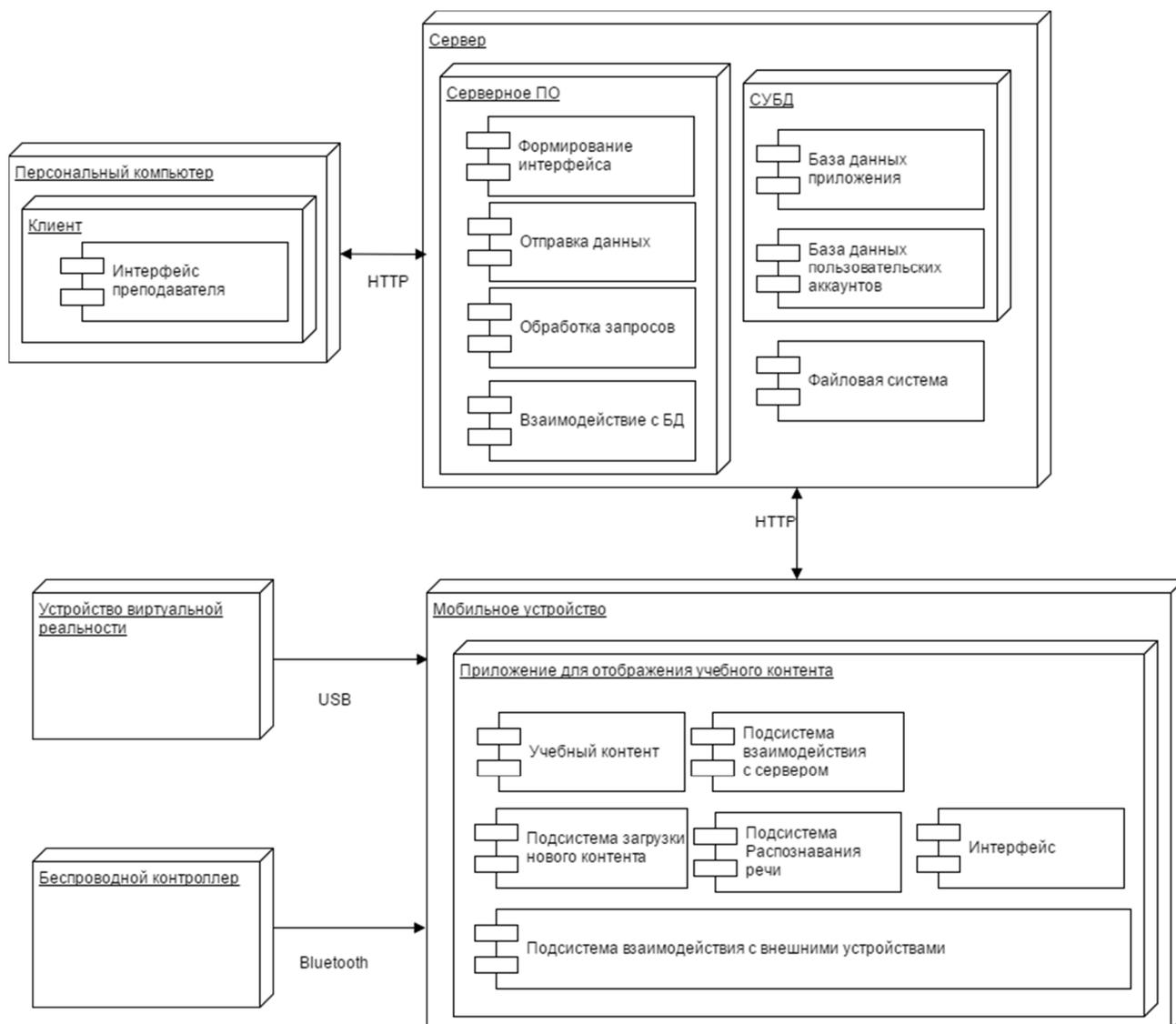


Рисунок 2.1 – Архитектура системы интерактивного обучения

На сервере располагаются:

- подсистема обработки входящих запросов реализует взаимодействие с приложением отображения визуального контента и интерфейса программного обеспечения;

- подсистема, служащая для отправки данных по сетевому протоколу;
- подсистема взаимодействия с базой данных, производящий чтение, удаление и изменение данных;
- подсистема формирования пользовательского интерфейса реализует динамическое отображение данных содержащихся в системе интерактивного обучения.

На сервере также располагаются две базы данных:

- база данных приложения;
- база данных пользовательских аккаунтов.

На персональном компьютере находится интерфейс программного обеспечения для преподавательского состава и администраторов системы.

На мобильном устройстве находится приложение для отображения учебного контента, которое включает в себя учебный контент и интегрированные подсистемы:

- подсистема взаимодействия с внешними устройствами – реализует инициализацию и обработку сигналов с беспроводного контроллера и с устройства виртуальной реальности;
- подсистема распознавания речи – необходимая для распознавания голосового ввода и взаимодействия с элементами учебного контента;
- подсистема загрузки нового контента – это подпрограмма реализующая доступ к учебным материалам, хранящимся на сервере;
- подсистема взаимодействия с сервером – реализует функции авторизации, отправки данных о прохождении учебного контента учащимся.

Передача сообщений между сервером и мобильным устройством, а также с персональным компьютером осуществляется по беспроводному протоколу HTTP.

HTTP – это протокол прикладного уровня. Обмен сообщениями идёт по схеме «запрос-ответ». Для идентификации ресурсов HTTP использует глобальные URL [11].

Взаимодействие между беспроводным устройством осуществляется по беспроводному протоколу Bluetooth.

Bluetooth – это спецификация беспроводных персональных сетей. Bluetooth обеспечивает обмен информацией между устройствами на надёжной, бесплатной, повсеместно доступной радиочастоте для ближней связи. Bluetooth позволяет этим устройствам общаться, когда они находятся в радиусе до 10 м друг от друга [12].

Взаимодействие между устройством виртуальной реальности и мобильным устройством, осуществляется по проводному интерфейсу USB.

USB – это последовательный интерфейс для подключения периферийных устройств к вычислительной технике [13].

Взаимодействие между интегрированными подсистемами и устройствами виртуальной реальности будет осуществляться по гибридной топологии методами: асинхронный обмен сообщениями, удаленный вызов процедур и общая база данных.

Подробнее схема взаимодействия между интегрированными подсистемами и устройствами представлена на рисунке 2.2.

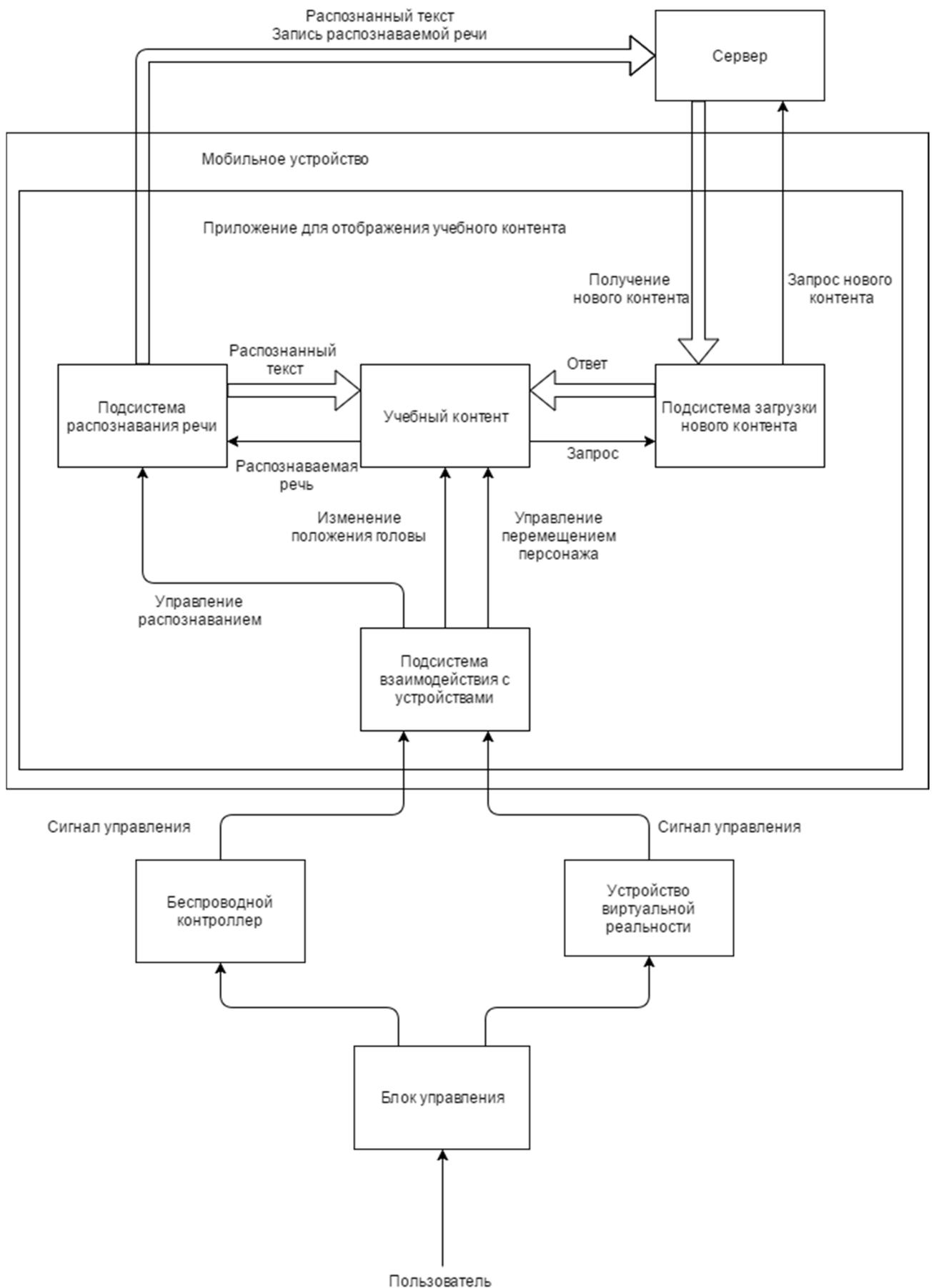


Рисунок 2.2 – Схема взаимодействия интегрируемых подсистем

Подсистема взаимодействия с внешними устройствами принимает с беспроводного контроллера и устройства виртуальной реальности сигналы управления и в зависимости от полученного сигнала, передает параметры обзора и перемещения персонажа либо на запуск подсистемы распознавания речи.

При вызове подсистемы распознавания речи записывается аудиофайл с распознаваемой речью. Распознанная речь преобразуется в текст, который передается ученику, а также вместе с записанным аудиофайлом отправляется на сервер.

При вызове подсистемы загрузки нового контента отправляется запрос на сервер. В ответ сервер отправляет список доступных уровней, если на мобильном устройстве отсутствуют некоторые уровни из списка, сервер предлагает загрузить их.

## **2.1 Выбор устройства виртуальной реальности**

В данной работе необходимо интегрировать устройство виртуальной реальности в систему интерактивного обучения. Для этого рассмотрим ряд популярных устройств, сравним их и выберем подходящее.

### **2.1.1 Устройство виртуальной реальности Oculus Rift**

Oculus Rift – устройство виртуальной реальности, разработанное компанией Oculus VR. Представляет из себя очки с экраном, которые надеваются на голову и подключаются к компьютеру. В комплекте так же идут два специально разработанных геймпада, которые позволяют манипулировать объектами в виртуальной реальности и камеры для отслеживания положения очков в пространстве [14]. Устройство представлено на рисунке 2.3.



Рисунок 2.3 – Внешний вид Oculus Rift

### **2.1.2 Устройство виртуальной реальности PlayStation VR**

PlayStation VR – устройство виртуальной реальности, разработанное Sony Interactive Entertainment. PlayStation VR рассчитан на совместную работу с игровой приставкой PlayStation 4. Управление в играх и приложениях виртуальной реальности осуществляется с помощью контроллера DualShock 4 или PlayStation Move. Движение головы в шлеме отслеживается с помощью PlayStation Camera [15]. Устройство представлено на рисунке 2.4.



Рисунок 2.4 – Внешний вид PlayStation VR

### 2.1.3 Устройство виртуальной реальности Samsung Gear VR

Samsung Gear VR – устройство виртуальной реальности, разработанное компанией Samsung в сотрудничестве с Oculus VR. В отличие от устройств, перечисленных выше Gear VR полностью автономен и не требует подключения к ПК [16]. Устройство представлено на рисунке 2.5

Samsung Gear VR не имеет собственного дисплея и работает в связке со смартфонами Samsung Galaxy. Устройство представляет собой приспособление для закрепления смартфона на голове перед глазами. Снабжено линзами, регулятором фокусировки, сенсорной панелью управления, регулятором громкости, а также оснащено дополнительными датчиками наклона головы, что увеличивает точность позиционирования и уменьшает время задержки изображения. Источником звука так же является смартфон. Соединение очков со смартфоном осуществляется посредством microUSB.



Рисунок 2.5 – Внешний вид Samsung Gear VR

## 2.1.4 Обоснование выбора Samsung Gear VR

Проанализировав рассматриваемые устройства виртуальной реальности, была составлена сравнительная таблица 2.

Таблица 2 – Сравнение устройств виртуальной реальности

Критерии сравнения	Oculus Rift	PlayStation VR	Samsung Gear VR
Низкая цена	Нет	Да	Да
Поддержка Android	Нет	Нет	Да
Дополнительные датчики положения головы	Да	Да	Да
Возможность интеграции с другими устройствами	Да	Нет	Да
Беспроводной контроллер	Да	Да	Нет
Высокая вычислительная мощность	Да	Да	Нет

На данный момент Samsung Gear VR является лидирующим на рынке устройств виртуальной реальности для мобильных платформ.

Достоинством Samsung Gear VR являются:

- дешевизна устройства;
- поддержка мобильной платформы Android;
- наличие дополнительных датчиков наклона головы;
- возможность интеграции с другими устройствами.

Недостатками данного устройства являются отсутствие в комплекте беспроводного контроллера и зависимость вычислительной мощности от мобильного устройства.

Устройство виртуальной реальности Samsung Gear VR является предпочтительным для интеграции в систему интерактивного обучения. К тому же важным фактором является, то что в системе интерактивного обучения используется мобильная платформа Android, то есть из всего перечисленного оно является единственным подходящим устройством для интеграции в систему интерактивного обучения.

## **2.2 Выбор системы распознавания речи**

В данной работе необходимо реализовать интеграцию в систему интерактивного обучения возможность распознавания речи. Для этого воспользуемся одной из готовых систем распознавания речи. Существуют такие системы с открытым и закрытым исходным кодом.

Далее рассмотрим ряд систем с открытым исходным кодом, сравним их и выберем подходящую.

### **2.2.1 Система Google Voice Recognition**

Google Voice Recognition – это продукт компании Google, разработанный в 2011 году для мобильной платформы Android, как часть голосового поиска Google Voice Search. С 2014 года распространяется бесплатно на большинстве современных платформ. Работает по принципу облачного сервиса. В данный момент размер словаря неизвестен [17].

Достоинствами являются: наличие поддержки русского языка, возможность интеграции в различные платформы.

Недостатками являются: время отклика, которое может составлять несколько секунд в зависимости от скорости интернет соединения и ограничение бесплатного количества запросов на использование в сутки.

### **2.2.2 Система Julius**

Julius – это система распознавания речи, созданная в рамках консорциума систем распознавания речи, предназначена для распознавания непрерывной речи в режиме реального времени, имеет словарь размером около 60000 слов. Изначально Julius разработан для работы на платформах Linux и UNIX, позднее была включена поддержка Windows [18].

Julius использует собственную спроектированную форму грамматики конечного автомата как языковую модель. Она может использоваться для строительства систем голосовой навигации или других разговорных систем для распознавания разного рода диалогов.

Достоинствами являются: отсутствие необходимости подключения интернет соединения и полностью бесплатное использование.

Недостатками являются: поддержка ограниченного количества платформ и языков.

### **2.2.3 Система PocketSphinx**

PocketSphinx – это версия системы распознавания речи CMU Sphinx, написанная на языке программирования Java для мобильных платформ.

CMU Sphinx – система распознавания речи, созданная разработчиками университета Карнеги – Меллон. Является дикторонезависимой системой, поддерживает распознавание длительной непрерывной речи, использует скрытую Марковскую модель и n-граммную статистическую языковую модель, имеет огромный словарь распознавания [19].

### **2.2.4 Сравнение систем распознавания речи**

Для интеграции системы распознавание речи в систему интерактивного обучения на основе анализа предметной области были выделены следующие

критерии: возможность работать локально, поддержка Android, дикторнезависимость, бесплатное использование в коммерческих целях, возможность распознавания слитной речи и наличие большого словаря. Для сравнения систем распознавания речи, была составлена сравнительная таблица 3.

Таблица 3 – Сравнение систем распознавания речи

Критерии сравнения	Google Voice Recognition	Julius	PocketSphinx
Возможность работать локально	Нет	Да	Да
Поддержка Android	Да	Нет	Да
Дикторнезависимость	Да	Нет	Да
Бесплатное использование	Нет	Да	Да
Большой словарь	Да	Да	Да
Возможность распознавания слитной речи	Да	Да	Да

В ходе сравнения было выявлено что система распознавания речи PocketSphinx лучше всего подходит для интеграции ее в проект, так как она единственная из списка, удовлетворяет всем критериям.

### 2.3 Средства программной разработки

Разработка приложения отображения учебного контента была реализована с помощью интегрированной среды разработки Unity.

Unity – это мультиплатформенный инструмент для разработки двух и трехмерных приложений и игр. Созданные при помощи Unity приложения работают на обширном количестве операционных систем и аппаратных платформ [20]. Интерфейс Unity представлен на рисунке 2.9.

Среда имеет интуитивно понятный интерфейс с возможностью настройки, благодаря чему возможно производить отладку приложения непосредственно в окне редактора.

Проект в Unity делится на сцены – отдельные файлы, содержащие наборы объектов, сценариев и настроек. Объекты в свою очередь содержат ряд компонентов, с которыми взаимодействуют сценарии. Среда поддерживает физику твердых тел, присутствует поддержка наследования объектов: дочерние объекты повторяют параметры позиции, вращения и масштаба родительского объекта.

Разработка сценариев взаимодействия объектов возможна при помощи языков программирования C# и JavaScript. Среда разработки Unity поддерживает более двадцати программных платформ среди которых значатся: Linux, Microsoft Windows, OSX, iOS, Anrroid и др.



Рисунок 2.9 – Интерфейс Unity

Для создания программных интерфейсов на языке программирования C#, было решено использовать интегрированную среду разработки Microsoft Visual Studio.

Так как исходный код системы распознавания PocketSphinx написан на языке программирования Java для мобильной платформы Android, то для реализации интеграции ее в приложение была выбрана интегрированная среда разработки Android Studio.

Android Studio – это интегрированная среда разработки мобильных приложений для мобильной платформы Android, которая является набором интегрированных инструментов для эффективной разработки, отладки и тестирования программ [21].

Данная среда разработки доступна для Windows, OS X и Linux. Официально поддерживает языки программирования Java и C++.

### 3 Реализация интеграции технологий виртуальной реальности в систему интерактивного обучения

Так как задачей работы является интеграция, были разработаны подсистемы управления, позволяющие интегрированным устройствам и программам взаимодействовать с системой на программном уровне. Схема уровней интеграции подсистем представлена на рисунке 3.1.

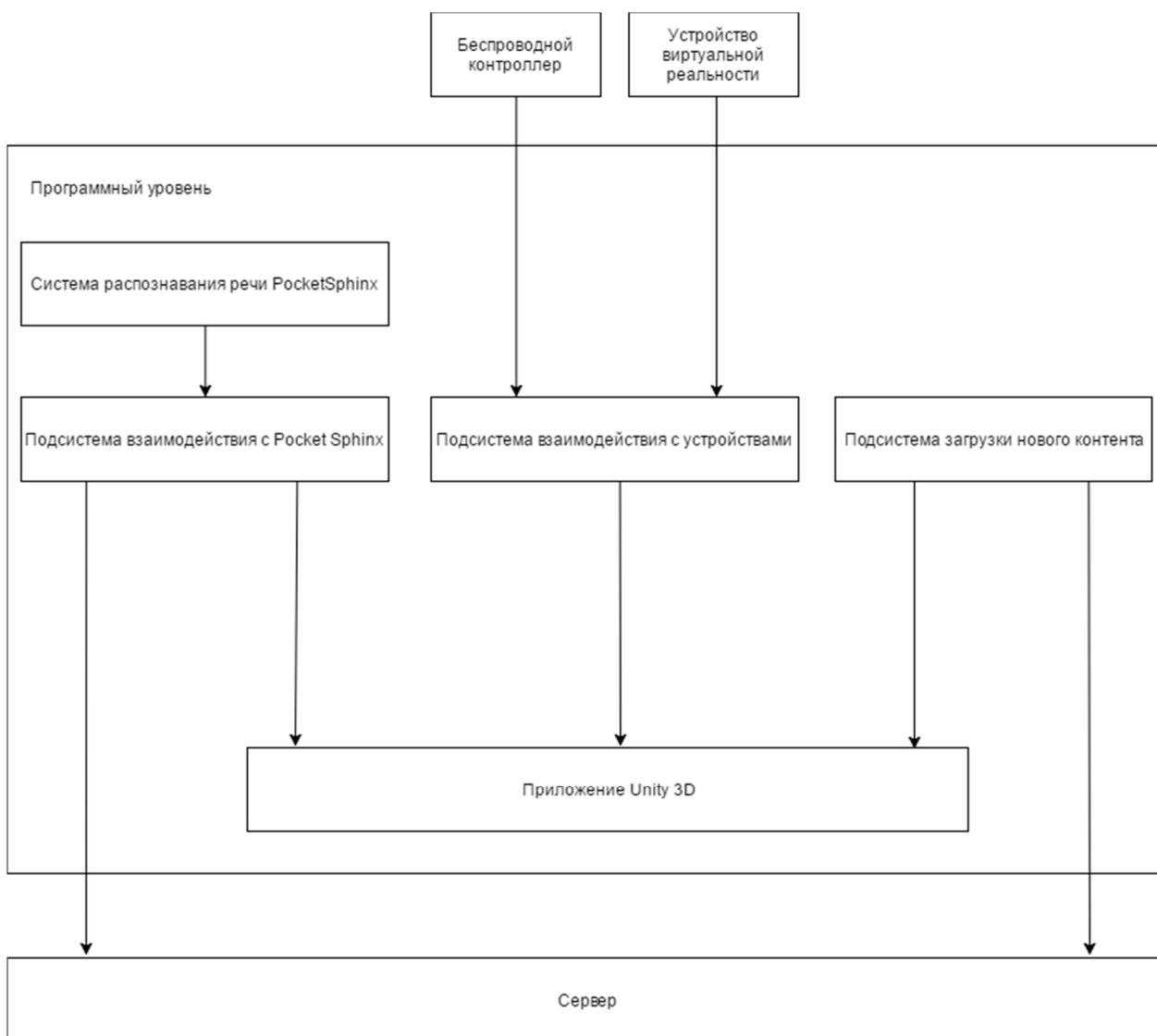


Рисунок 3.1 – Схема уровней интеграции подсистем

### 3.1 Интеграция системы распознавания PocketSphinx

Так как приложение для отображения учебного контента разработано с помощью среды разработки Unity для мобильной платформы Android, за основу был взят открытый исходный код системы распознавания PocketSphinx написанный в Android Studio.

Чтобы PocketSphinx мог работать в среде разработки Unity его исходный код из Android Studio был экспортирован в виде файлов библиотек классов и помещен в проект Unity в папку Plugins. Для того, чтобы использовать функции PocketSphinx была создана подпрограмма на языке программирования C#, которая инициализирует и позволяет вызывать функции из библиотек классов PocketSphinx. Инициализация функций PocketSphinx представлена на рисунке 3.2.

```
public delegate void initCallbackDelegate(int n);
public delegate void resultCallbackDelegate(int score, int speed, string words);
public delegate void completionCallbackDelegate(int completion);
public delegate void newWordsCallbackDelegate(string words);
public delegate void newAudioCallbackDelegate(int volume, string pitch, string waveform);
public delegate void playbackDoneCallbackDelegate();

[DllImport("upal")]
private static extern int startInitialization(initCallbackDelegate icb, string path);
[DllImport("upal")]
private static extern int setPlaybackDoneCallback(playbackDoneCallbackDelegate pcb);
[DllImport("upal")]
private static extern int setResultCallback(resultCallbackDelegate rcb);
[DllImport("upal")]
private static extern int setCompletionCallback(completionCallbackDelegate ccb);
[DllImport("upal")]
private static extern int setNewWordsCallback(newWordsCallbackDelegate nwcb);
[DllImport("upal")]
private static extern int setNewAudioCallback(newAudioCallbackDelegate nacb);
[DllImport("upal")]
private static extern int startRecording(string sentences);
[DllImport("upal")]
private static extern int stopRecording();
[DllImport("upal")]
private static extern int setStrictness(int strictness);
[DllImport("upal")]
private static extern int startPlayback();
[DllImport("upal")]
private static extern int stopPlayback();
[DllImport("upal")]
private static extern int addWord(string word, string pronunciation);
```

Рисунок 3.2 – Инициализация функций PocketSphinx

Запись распознаваемой речи происходит после вызова функции `startrecord` и корутина `startrecord2`.

Корутины (Coroutines, сопрограммы) в Unity – простой и удобный способ запускать функции, которые должны работать параллельно в течение некоторого времени [22].

Корутины позволяют делить основной поток программы для одновременного выполнения нескольких функций в одном потоке. Такой способ позволяет избежать зависаний программы.

Функция `startrecord` вызывает функцию распознавания речи `startRecording` из библиотеки классов `PocketSphinx` и передает ей текст, который необходимо распознать. Далее запускается таймер, по истечению которого распознавание завершится. Функция `startrecord` представлена на рисунке 3.3.

```
private void startrecord()
{
    Debug.Log("Starting record");
    start = true;
    if (stopped)
        return;
    startRecording(text_record);
    timer = new System.Timers.Timer(3000);
    timer.Elapsed += onRecordingDone;
    timer.AutoReset = false;
    timer.Enabled = true;
}
```

Рисунок 3.3 – Функция `startrecord`

Функция `startrecord2` инициализирует запуск микрофона для записи аудиоклипа и сохраняет полученную запись на мобильном устройстве с помощью функции `Save`. Далее она инициализирует и запускает новый поток, который отправляет сохраненный аудио файл и распознанный текст на сервер. Вызывается корутин благодаря функции запуска корутина `StartCoroutine("startrecord2")`. Корутин `startrecord2` представлен на рисунке 3.4.

```

IEnumerator startrecord2()
{
    myAudioClip = Microphone.Start(null, false, 3, 44100);
    yield return new WaitForSeconds(3);
    Savewav.Save("/mnt/sdcard/myfile", myAudioClip);
    Thread thread1 = new Thread(mythread1);
    thread1.Start();
}

```

Рисунок 3.4 – Корути startrecord2

Функция Save конвертирует полученный аудио клип в аудиофайл на мобильном устройстве, создавая новый поток в памяти устройства, благодаря классу MemoryStream. Функция Save представлена на рисунке 3.5.

```

public static bool Save(string filename, AudioClip clip)
{
    if (!filename.ToLower().EndsWith(".wav"))
    {
        filename += ".wav";
    }

    var filepath = filename;

    Debug.Log(filepath);

    Directory.CreateDirectory(Path.GetDirectoryName(filepath));
    ClipData clipdata = new ClipData();
    clipdata.samples = clip.samples;
    clipdata.channels = clip.channels;
    float[] dataFloat = new float[clip.samples * clip.channels];
    clip.GetData(dataFloat, 0);
    clipdata.samplesData = dataFloat;
    using (var fileStream = CreateEmpty(filepath))
    {
        MemoryStream memstrm = new MemoryStream();
        ConvertAndWrite(memstrm, clipdata);
        memstrm.WriteTo(fileStream);
        WriteHeader(fileStream, clip);
    }

    return true;
}

```

Рисунок 3.5 – Функция Save

## 3.2 Интеграция подсистемы загрузки нового контента

Для реализации загрузки нового контента была написана программа на языке программирования C#, позволяющая скачивать новые уровни в виде AssetBundles и распаковывать их для дальнейшего использования.

AssetBundles - это файлы, которые можно экспортировать из Unity, для упаковки ресурсов (assets) в один файл, по выбору. Эти файлы используют собственный формат сжатия, и могут быть загружены приложением по необходимости. Это позволяет подгружать различный контент, такой как модели, текстуры, аудио клипы, или даже целые сцены, отдельно от тех, в которых они будут использоваться. AssetBundles созданы для упрощения скачивания контента в приложение [23]. Пример работы с AssetBundles представлен на рисунке 3.6.

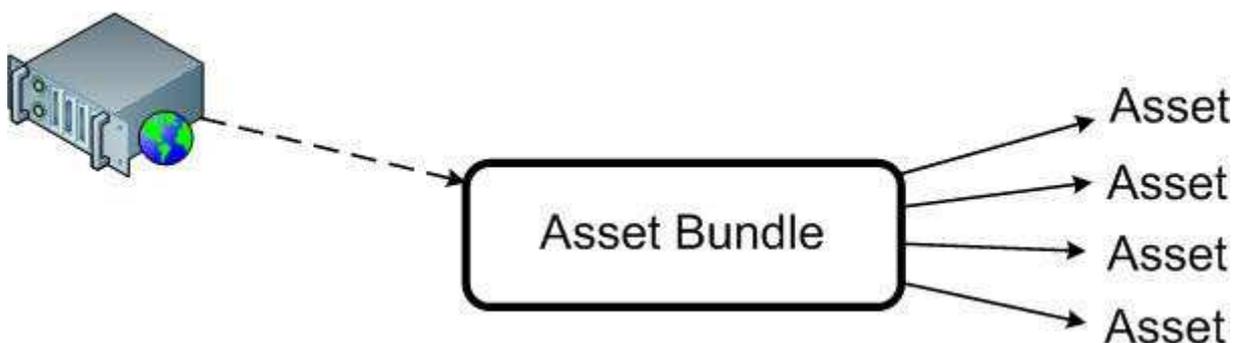


Рисунок 3.6 – Пример работы с AssetBundles

Во время работы программы приложение загрузит AssetBundles по требованию и позволит управлять отдельными ассетами в рамках каждого AssetBundle по мере необходимости. Процесс загрузки AssetBundles представлен на рисунке 3.7.

```

IEnumerator Start ()
{
    using (WWW www = new WWW(url))
    {
        yield return www;
        if (!string.IsNullOrEmpty(www.error))
        {
            Debug.LogError(www.error);
            yield break;
        }

        assetBundle = www.assetBundle;
        string[] scenes = assetBundle.GetAllScenePaths();

        foreach (string sceneName in scenes)
        {
            labelText.text = Path.GetFileNameWithoutExtension(sceneName);

            var clone = Instantiate(prefab.gameObject) as GameObject;

            clone.GetComponent<Button>().AddEventListener(labelText.text, LoadAssetBundleScene);

            clone.SetActive(true);
            clone.transform.SetParent(rootContainer);
        }
    }
}

```

Рисунок 3.7 – Процесс скачивания AssetBundles

### 3.3 Интеграция внешних устройств

Для организации взаимодействия устройств с приложением для отображения учебного контента была написана подпрограмма на языке программирования C#, реализующая взаимодействие устройств с персонажем. Обработка сигналов с устройств представлено на рисунке 3.8.

```

control = Input.GetAxis("Horizontal");
button = Input.GetButton("Y");

if (control < 0)
    rb.AddForce(VR_camera.transform.forward * 10);
if (control > 0)
    rb.AddForce(-1*(VR_camera.transform.forward * 10));

if (button > 0)
    PocketSphinkx.startrecord();

```

Рисунок 3.8 – Обработка сигналов с устройств

Переменная `control` принимает данные о положении аналогового стика с беспроводного контроллера и в зависимости от сигнала персонаж перемещается вперед или назад.

Аналоговый стик (англ. `analog stick`) – это устройство управления (в частности, управления играми), чья подвижность ограничена двумя степенями свободы [24].

Направление движения и угол обзора персонажа изменяются от полученных параметров  $X$ ,  $Y$ ,  $Z$  с устройства виртуальной реальности, которые передаются в объект `Vr_camera`. Принцип изменения параметров  $X$ ,  $Y$ ,  $Z$  представлен на рисунке 3.9.

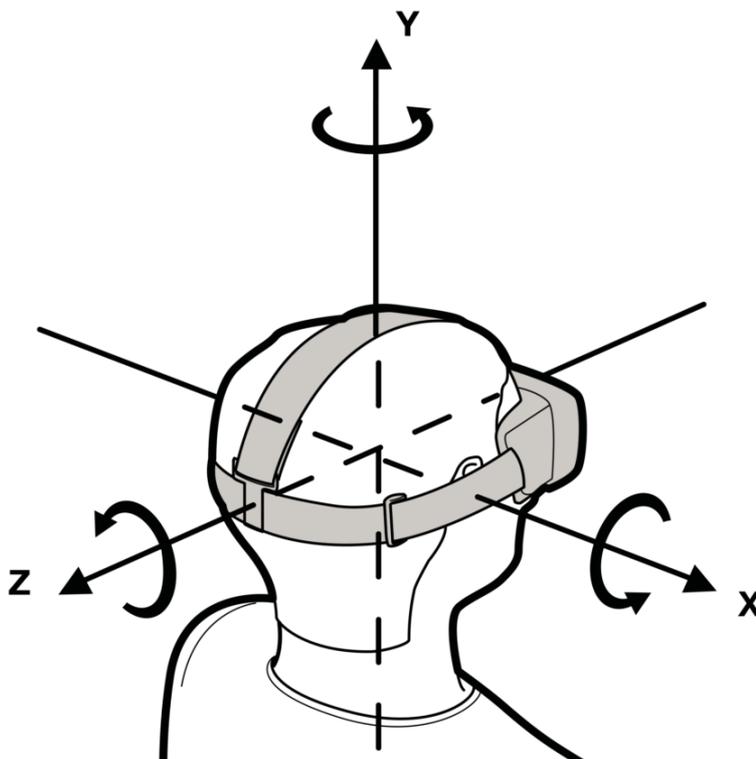


Рисунок 3.9 – Изменение параметров  $X$ ,  $Y$ ,  $Z$

Переменная `button` принимает значения кнопки `Y` на беспроводном контроллере. Если кнопка нажата, то вызывается подсистема распознавания речи.

### 3.4 Модель программы

Программа служит для обеспечения, обучающегося возможностью повысить разговорные навыки английского языка путем погружения его в виртуальную реальность. В созданной симуляционной среде ученик должен пройти несколько уровней, выполняя в каждом из них ряд заданий по заранее подготовленному сценарию, который был подготовлен преподавателем в соответствии с программой обучения. Пример сценария представлен в приложении «А».

Обучающемуся предоставляется возможность управлять персонажем, который идентифицируется самим обучающимся. Персонаж может вести беседу с другими персонажами, управляя которым обучающийся выполняет задания.

В зависимости от уровня обучающийся может наблюдать сцену: парк, дорога или улица города, на которой находятся различные игровые персонажи, здания и указатели.

Для осмотра местности обучающийся должен использовать устройство виртуальной реальности, которое надевается на голову и переносит её движения в симуляционную среду, позволяет менять угол обзора и направление движения персонажа. Для перемещения персонажа по уровню обучающийся должен использовать геймпад, взаимодействуя с которым он может перемещать персонажа по сцене. Во время прохождения уровня для общения с другими персонажами обучающемуся необходимо подойти своим персонажем к собеседнику и с помощью геймпада активировать микрофон для записи и распознавания своей речи.

Для увеличения количества уровней можно использовать подсистему загрузки нового контента, которая позволяет загружать новые уровни.

Текст разработанных программных интерфейсов представлен в приложении «Б».

## 4 Руководство для реализации интеграции технологий виртуальной реальности в проект Unity

Чтобы интегрировать устройства и программы в другие проекты, необходимо поместить написанные интерфейсы взаимодействия и подпрограммы в папку ресурсов (Assets) проекта Unity и произвести настройку. Пример размещения файлов в проекте представлено на рисунке 4.1.



Рисунок 4.1 – Размещение файлов в проекте

Далее эти файлы необходимо поместить на объект, который находится на сцене.

Для того чтобы система распознавания работала необходимо создать папку Plugins и поместить туда папку с библиотеками классов PocketSphinx. Содержимое папки Plugins представлено на рисунке 4.2.

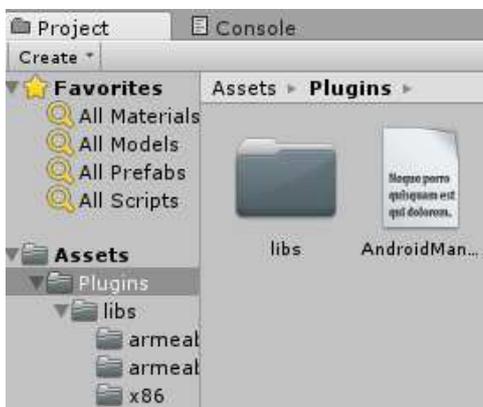


Рисунок 4.2 – содержимое папки Plugins

Для активации поддержки устройства виртуальной реальности необходимо перейти во вкладку Edit-> Project Settings-> Player-> Other Settings-> Rendering. Переход в настройки представлен на рисунке 4.3.

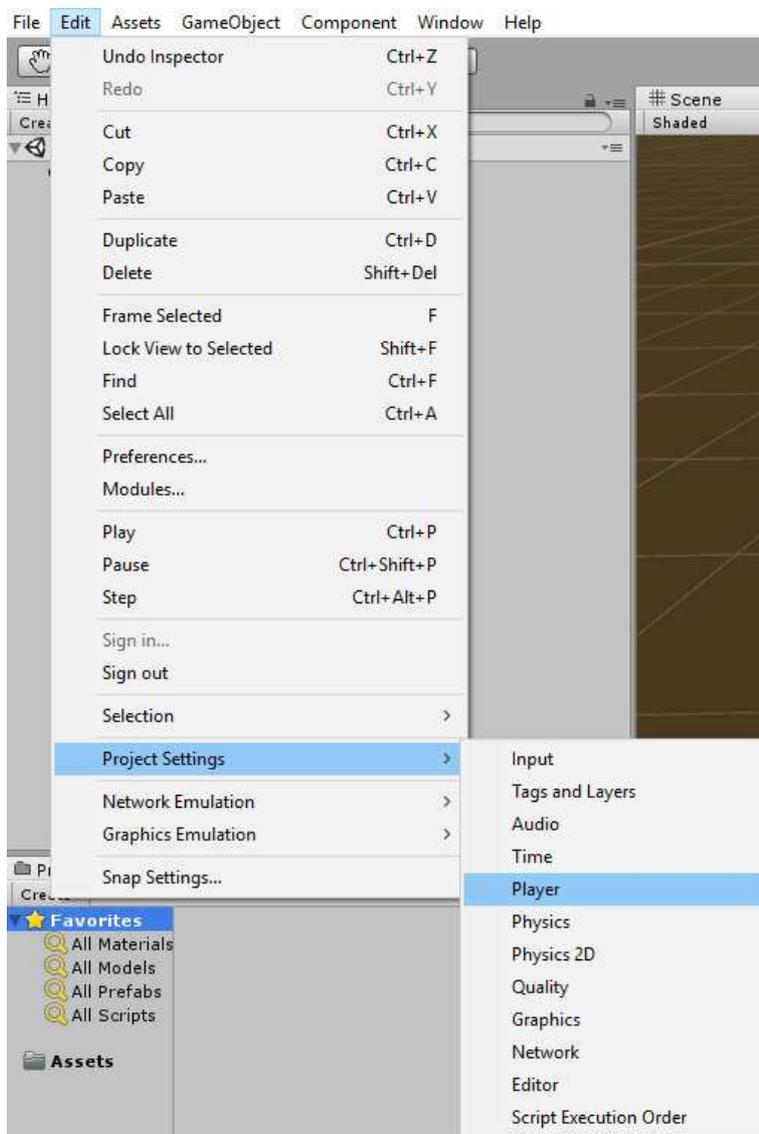


Рисунок 4.3 – Переход в настройки

И активировать пункт Virtual Reality Supported, представленный на рисунке 4.4.

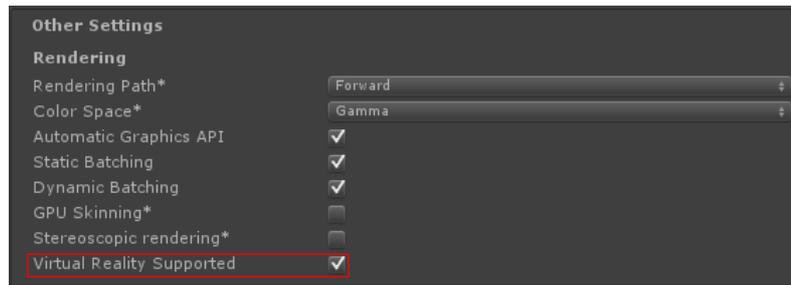


Рисунок 4.4 – Активация поддержки виртуальной реальности

Для настройки беспроводного контроллера необходимо перейти во вкладку Edit-> Project Settings-> Input. Переход в настройки управления представлен на рисунке 4.5.

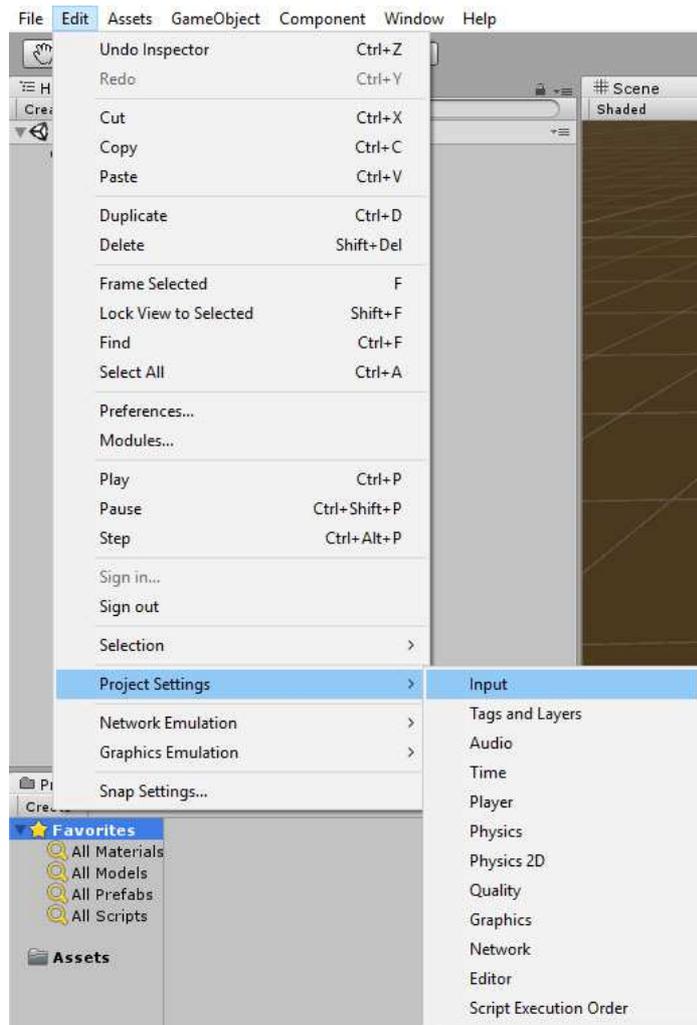


Рисунок 4.5 Переход в настройки управления

И настроить клавиши управления беспроводного контроллера согласно рисунку 4.6.

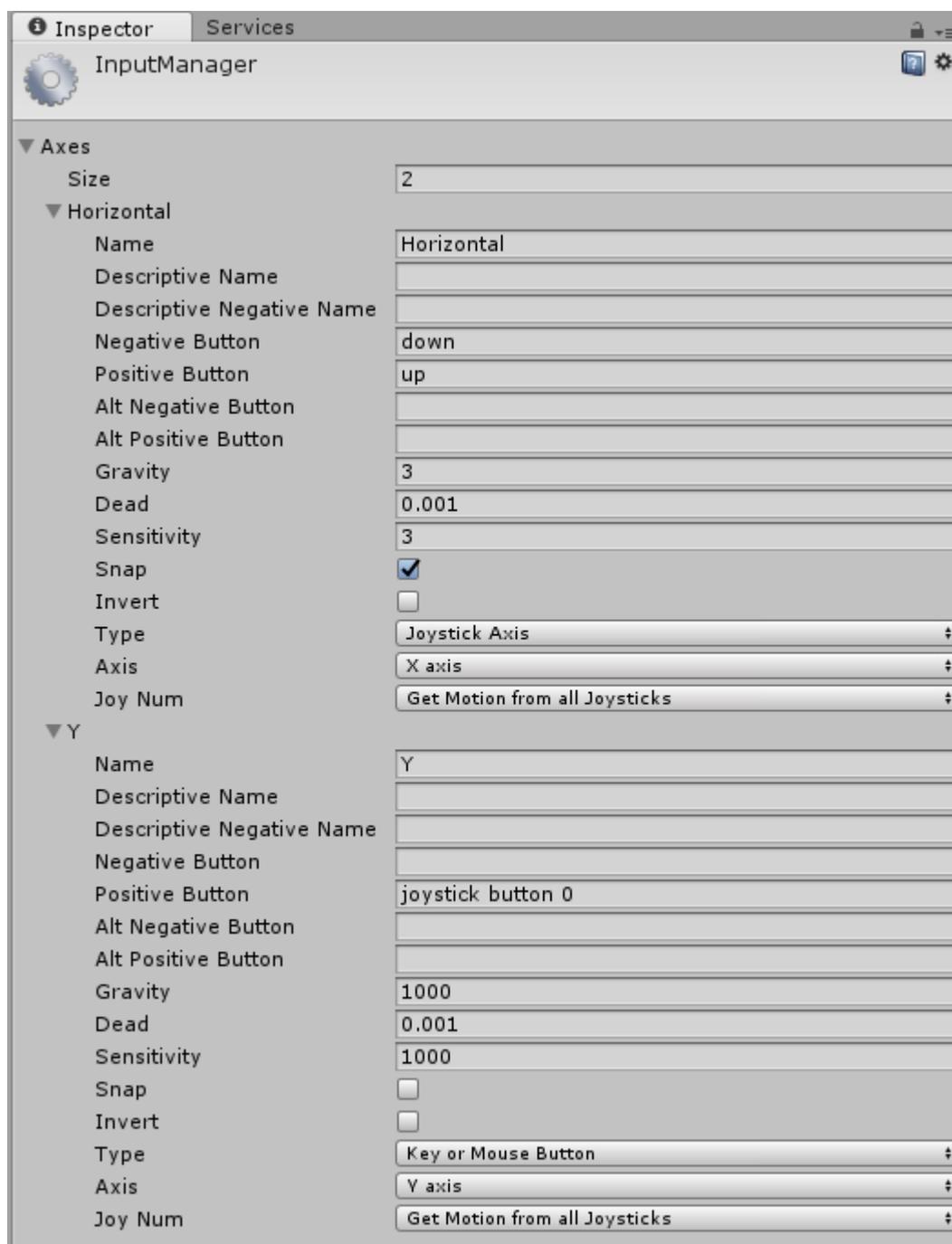


Рисунок 4.6 – Настройка управления беспроводного контроллера

## ЗАКЛЮЧЕНИЕ

В ходе выполнения данной выпускной квалификационной работы были разработаны программные интерфейсы, реализующие интеграцию технологий виртуальной реальности в систему интерактивного обучения.

Для достижения поставленной цели были решены следующие задачи:

- изучена информация о данной предметной области;
- проанализированы существующие аналоги систем виртуальной реальности;
- выбраны необходимые устройства и программы для интеграции в систему интерактивного обучения;
- выбраны программные средства для реализации интеграции в систему интерактивного обучения;
- разработана архитектура системы;
- реализована интеграция систем;
- написана документация для интеграции технологий виртуальной реальности в проект Unity.

Основным преимуществом написанных программных интерфейсов является возможность параллельной работы подсистем, позволяющая избежать зависаний программы.

Основным недостатком является низкая скорость работы подсистемы распознавания речи.

## СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Polymedia [Электронный ресурс]. – Интерактивные технологии в образовании. – Режим доступа: <http://www.polymedia.ru/o-kompanii/stati/interaktivnyie-tekhnologii-v-obrazovanii/> (дата обращения 29.04.2017)
2. Страница Виртуальная реальность на Википедия [Электронный ресурс]. – Режим доступа: [https://ru.wikipedia.org/wiki/Виртуальная\\_реальность](https://ru.wikipedia.org/wiki/Виртуальная_реальность) (дата обращения 29.04.2017)
3. Страница Интеграция программного обеспечения на Хабрахабр [Электронный ресурс]. – Режим доступа: <https://habrahabr.ru/post/245615/> (дата обращения 29.04.2017)
4. Nival [Электронный ресурс]. – Inmind VR. – Режим доступа: <http://ru.nival.com/> (дата обращения 29.04.2017)
5. Unimersiv [Электронный ресурс]. – About us. – Режим доступа: <https://unimersiv.com/about-us/> (дата обращения 29.04.2017)
6. VRtisan [Электронный ресурс]. – Режим доступа: <http://www.vrtisan.co.uk/> (дата обращения 29.04.2017)
7. Открытые системы СУБД [Электронный ресурс]. – Интеграция приложений: методы взаимодействия, топология, инструменты. – Режим доступа: <https://www.osp.ru/os/2006/09/3776464/> (дата обращения 30.04.2017)
8. Страница HTTP на Википедия [Электронный ресурс]. – Режим доступа: <https://ru.wikipedia.org/wiki/HTTP> (дата обращения 30.04.2017)
9. Страница Bluetooth на Википедия [Электронный ресурс]. – Режим доступа: <https://ru.wikipedia.org/wiki/Bluetooth> (дата обращения 30.04.2017)
10. Страница USB на Википедия [Электронный ресурс]. – Режим доступа: <https://ru.wikipedia.org/wiki/USB> (дата обращения 30.04.2017)

11. Oculus [Электронный ресурс]. – Rift. – Режим доступа: <https://www.oculus.com/rift/> (дата обращения 1.05.2017)
12. Playstation [Электронный ресурс]. – PS VR. – Режим доступа: <https://www.playstation.com/ru-ru/explore/playstation-vr/> (дата обращения 1.05.2017)
13. Samsung [Электронный ресурс]. – Gear VR. – Режим доступа: <http://www.samsung.com/ru/wearables/gear-vr-r323/> (дата обращения 1.05.2017)
14. Страница Распознавание речи на Википедия [Электронный ресурс]. – Режим доступа: [https://ru.wikipedia.org/wiki/Распознавание\\_речи](https://ru.wikipedia.org/wiki/Распознавание_речи) (дата обращения 2.05.2017)
15. Geektimes [Электронный ресурс]. – Распознавание речи. Часть 1. Классификация систем распознавания речи. – Режим доступа: <https://geektimes.ru/post/64572/> (дата обращения 2.05.2017)
16. Speech-text [Электронный ресурс]. – Компьютерное распознавание и порождение речи. – Режим доступа: <http://speech-text.narod.ru/index3.html> (дата обращения 2.05.2017)
17. Cloud.google [Электронный ресурс]. – Cloud speech api. – Режим доступа: <https://cloud.google.com/speech/> (дата обращения 2.05.2017)
18. Страница Поиск оптимальной аудиосистемы распознавания речи на базе открытого исходного кода на Хабрахабр [Электронный ресурс]. – Режим доступа: <https://habrahabr.ru/post/230133/> (дата обращения 5.05.2017)
19. CMU Sphinx [Электронный ресурс]. – Режим доступа: <https://cmusphinx.github.io/> (дата обращения 5.05.2017)
20. Страница Работа с Корутинами в Unity на Хабрахабр [Электронный ресурс]. – Режим доступа: <https://habrahabr.ru/post/216185/> (дата обращения 5.05.2017)

21. Unity [Электронный ресурс]. – AssetBundles. – Режим доступа: <https://docs.unity3d.com/ru/530/Manual/AssetBundlesIntro.html> (дата обращения 7.05.2017)
22. Android Studio [Электронный ресурс]. – Режим доступа: <https://developer.android.com/studio/index.html> (дата обращения 7.05.2017)
23. Unity [Электронный ресурс]. – Режим доступа: <https://unity3d.com/ru/unity> (дата обращения 7.05.2017)
24. Наука 21 век [Электронный ресурс]. – Что такое аналоговый стик. – Режим доступа: <http://наука21vek.ru/archives/53432> (дата обращения 7.05.2017)

## ПРИЛОЖЕНИЕ А

### Сценарий

Тема: покупка билета.

Цель: развитие навыка аудирования, совершенствование лексического навыка по теме покупка билета.

Задание: пройди в офис турфирмы и приобрети два билета (детский и взрослый) на автобус до Лос-Анжелеса и обратно.

Содержание.

Персонаж идет по улице, согласно указателям-знакам, у дороги:

1. Go along Hill street. (Иди прямо по улице Hill street);
2. At the traffic lights turn right and cross the street. (У светофора поверни на право и пересеки улицу);
3. Go past the butcher's. (Пройди мимо мясной лавки);
4. You are here! (Ты на месте, табличка с надписью Travel Agent's на следующем здании).

Подходит к стойке и ведет диалог с работником агентства:

1. Ticket seller: Hello. Can I help you?
2. Ответ ученика: I'd like two bus tickets to Los Angeles, please, one adult and one child.
3. Ticket seller: Single or return?
4. Ответ ученика: Return, please.
5. Ticket seller: What time would you like to leave?
6. Ответ ученика: I'd like to take the 3 p.m. bus on Saturday.
7. Ticket seller: That is 45 pounds, please.
8. Ответ ученика: Here you are.
9. Ticket seller: Have a nice journey.
10. Ответ ученика: Thank you.

## ПРИЛОЖЕНИЕ Б

### Листинг разработанного программного обеспечения

Файл PocketSphinx.cs:

```
using UnityEngine;
using System.Collections;
using System.Runtime.InteropServices;
using System;
using System.Timers;
using UnityEngine.UI;
using System.Linq;
using System.IO;
using System.Threading;
using System.Collections.Generic;

[RequireComponent(typeof(AudioSource))]

public class PocketSphinx : MonoBehaviour
{
    // инициализация функций
    public delegate void initCallbackDelegate(int n);
    public delegate void resultCallbackDelegate(int score, int speed, string words);
    public delegate void completionCallbackDelegate(int completion);
    public delegate void newWordsCallbackDelegate(string words);
    public delegate void newAudioCallbackDelegate(int volume, string pitch, string waveform);
    public delegate void playbackDoneCallbackDelegate();

    [DllImport("upal")]
    private static extern int startInitialization(initCallbackDelegate icb, string path);
    [DllImport("upal")]
    private static extern int setPlaybackDoneCallback(playbackDoneCallbackDelegate pcb);
    [DllImport("upal")]
```

```

private static extern int setResultCallback(resultCallbackDelegate rcb);
[DllImport("upal")]
private static extern int setCompletionCallback(completionCallbackDelegate ccb);
[DllImport("upal")]
private static extern int setNewWordsCallback(newWordsCallbackDelegate nwcb);
[DllImport("upal")]
private static extern int setNewAudioCallback(newAudioCallbackDelegate nachb);
[DllImport("upal")]
private static extern int startRecording(string sentences);
[DllImport("upal")]
private static extern int stopRecording();
[DllImport("upal")]
private static extern int setStrictness(int strictness);
[DllImport("upal")]
private static extern int startPlayback();
[DllImport("upal")]
private static extern int stopPlayback();
[DllImport("upal")]
private static extern int addWord(string word, string pronunciation);

private static System.Timers.Timer timer;
private static bool stopped;

void Awake()
{
    Debug.Log("About to initialize plugin");
    stopped = false;
    Debug.Log(startInitialization(new initCallbackDelegate(this.initCallback),
Application.persistentDataPath));
}

```

```

void StopAll()
{
    Debug.Log("Stopping all");
    stopped = true;
    stopRecording();
    stopPlayback();
}

void OnApplicationQuit()
{
    StopAll();
}

public static string text123;
public static string[] text_source = File.ReadAllLines("/mnt/sdcard/text.txt");
public static string text_result = "";
public static string text_record = "";
public static bool start = false;

GUIStyle largeFont;

void Start()
{
    largeFont = new GUIStyle();
    largeFont.fontSize = 72;

    for (int i = 0; i < text_source.Length; i++)
    {
        text_record += text_source[i] + ",";
    }
}

```

```

static void mythread1()
{
    // отправка на сервер
    ftp ftpClient = new ftp(@"ftp://185.91.177.214/ftp/", "server", "12345678");
    ftpClient.upload("myfile.wav", @"/mnt/sdcard/myfile.wav");
}

void Update()
{
    string[] StringsArray = text123.Split('-', ' ');
    int[] IntArray = StringsArray.Select(x => int.Parse(x)).ToArray();
    text_result = "";
    for (int j = 0; j < text_sourse.Length; j++)
    {
        for (int i = 0; i < IntArray.Length; i = i + 4)
        {
            if (IntArray[i] == j)
            {
                string[] StringsSourse = text_sourse[j].Split(' ');
                text_result += " " + StringsSourse[IntArray[i + 1]];
            }
        }
    }
}

void OnGUI()
{
    if (text123 != "")
    {
        GUILayout.Label(new Rect(50, 50, 200, 100), text_result, largeFont);
        GUILayout.Label(new Rect(50, 200, 200, 100), text123, largeFont);
    }
    else
}

```

```

    GUI.Label(new Rect(50, 50, 200, 100), "lol", largeFont);

if (start == true)
    GUI.Label(new Rect(50, 400, 200, 100), "говоруме", largeFont);
else
    GUI.Label(new Rect(50, 400, 200, 100), "молчуме", largeFont);

if (GUI.Button(new Rect(Screen.width / 2 - 50, Screen.height / 2 - 50, 200, 100), "Занучь"))
{
    startrecord();
    StartCoroutine("record2");
}
}

public void initCallback(int status)
{
    Debug.Log("Plugin initialization done");
    Debug.Log(status);
    Debug.Log("About to wire callbacks");
    setResultCallback(new resultCallbackDelegate(resultCallback));
    setCompletionCallback(new completionCallbackDelegate(completionCallback));
    setNewWordsCallback(new newWordsCallbackDelegate(newWordsCallback));
    setNewAudioCallback(new newAudioCallbackDelegate(newAudioCallback));
}

private void onRecordingDone(object source, ElapsedEventArgs e)
{
    Debug.Log("Stopping recording");
    stopRecording();
    start = false;
}

```

```

public static void resultCallback(int score, int speed, string words)
{
    Debug.Log("Result");
    Debug.Log(score);
    Debug.Log(speed);
    Debug.Log(words);
}

```

```

private void startrecord()
{
    Debug.Log("Starting record");
    start = true;
    if (stopped)
        return;
    startRecording(text_record);
    timer = new System.Timers.Timer(3000);
    timer.Elapsed += onRecordingDone;
    timer.AutoReset = false;
    timer.Enabled = true;
}

```

```

static AudioClip myAudioClip;

```

```

IEnumerator record2()
{
    //запись аудио
    myAudioClip = Microphone.Start(null, false, 3, 44100);
    yield return new WaitForSeconds(3);
    Savewaw.Save("/mnt/sdcard/myfile", myAudioClip);
    Thread thread1 = new Thread(mythread1);
    thread1.Start();
}

```

```
public static void completionCallback(int completion)  
{  
    Debug.Log("Completion");  
    Debug.Log(completion);  
}
```

```
public static void newWordsCallback(string words)  
{  
    Debug.Log("New words");  
    Debug.Log(words);  
    text123 = words;  
}
```

```
public static void newAudioCallback(int volume, string pitch, string waveform)  
{  
    Debug.Log("New audio data");  
  
    Debug.Log(volume);  
    Debug.Log(pitch);  
    Debug.Log(waveform);  
}  
}
```

Файл Player\_control.cs:

```
using UnityEngine;  
using System.Collections;  
  
public class Player_control : MonoBehaviour {  
    public GameObject VR_camera;  
    float control;  
  
    public float maxSpeed;
```

```

public Vector3 direction;
public Rigidbody rb;
void Start () {
    rb = GetComponent<Rigidbody>();
}

void FixedUpdate() {

    control = Input.GetAxis("Horizontal");
    button = Input.GetButton("Y");

    if (control < 0)
        rb.AddForce(VR_camera.transform.forward * 10);
    if (control > 0)
        rb.AddForce(-1*(VR_camera.transform.forward * 10));

    if (button > 0)
        PocketSphinx.startrecord();

    if (rb.velocity.magnitude > maxSpeed)
    {
        rb.velocity = rb.velocity.normalized * maxSpeed;
    }
}
}

```

Файл Savewaw.cs:

```

using UnityEngine;
using System.Collections;
using System.Runtime.InteropServices;
using System;
using System.Timers;
using UnityEngine.UI;
using System.Linq;
using System.IO;

```

```
using System.Threading;  
using System.Collections.Generic;
```

```
public class Savewaw : MonoBehaviour {
```

```
    const int HEADER_SIZE = 44;
```

```
    struct ClipData
```

```
    {
```

```
        public int samples;
```

```
        public int channels;
```

```
        public float[] samplesData;
```

```
    }
```

```
    public static bool Save(string filename, AudioClip clip)
```

```
    {
```

```
        if (!filename.ToLower().EndsWith(".wav"))
```

```
        {
```

```
            filename += ".wav";
```

```
        }
```

```
        var filepath = filename;
```

```
        Debug.Log(filepath);
```

```
        Directory.CreateDirectory(Path.GetDirectoryName(filepath));
```

```
        ClipData clipdata = new ClipData();
```

```
        clipdata.samples = clip.samples;
```

```
        clipdata.channels = clip.channels;
```

```
        float[] dataFloat = new float[clip.samples * clip.channels];
```

```
        clip.GetData(dataFloat, 0);
```

```
        clipdata.samplesData = dataFloat;
```

```
        using (var fileStream = CreateEmpty(filepath))
```

```
        {
```

```
            MemoryStream memstrm = new MemoryStream();
```

```
            ConvertAndWrite(memstrm, clipdata);
```

```

        memstrm.WriteTo(fileStream);
        WriteHeader(fileStream, clip);
    }

    return true;
}

public AudioClip TrimSilence(AudioClip clip, float min)
{
    var samples = new float[clip.samples];

    clip.GetData(samples, 0);

    return TrimSilence(new List<float>(samples), min, clip.channels, clip.frequency);
}

public AudioClip TrimSilence(List<float> samples, float min, int channels, int hz)
{
    return TrimSilence(samples, min, channels, hz, false, false);
}

public AudioClip TrimSilence(List<float> samples, float min, int channels, int hz, bool _3D,
bool stream)
{
    int i;

    for (i = 0; i < samples.Count; i++)
    {
        if (Mathf.Abs(samples[i]) > min)
        {
            break;
        }
    }

    samples.RemoveRange(0, i);
}

```

```

for (i = samples.Count - 1; i > 0; i--)
{
    if (Mathf.Abs(samples[i]) > min)
    {
        break;
    }
}
samples.RemoveRange(i, samples.Count - i);
var clip = AudioClip.Create("TempClip", samples.Count, channels, hz, _3D, stream);
clip.SetData(samples.ToArray(), 0);
return clip;
}

```

```

static FileStream CreateEmpty(string filepath)
{
    var fileStream = new FileStream(filepath, FileMode.Create);
    byte emptyByte = new byte();
    for (int i = 0; i < HEADER_SIZE; i++)
    {
        fileStream.WriteByte(emptyByte);
    }
    return fileStream;
}

```

```

static void ConvertAndWrite(MemoryStream memStream, ClipData clipData)
{
    float[] samples = new float[clipData.samples * clipData.channels];

    samples = clipData.samplesData;
    Int16[] intData = new Int16[samples.Length];
    Byte[] bytesData = new Byte[samples.Length * 2];
    const float rescaleFactor = 32767;

    for (int i = 0; i < samples.Length; i++)

```

```

    {
        intData[i] = (short)(samples[i] * rescaleFactor);
    }
    Buffer.BlockCopy(intData, 0, bytesData, 0, bytesData.Length);
    memStream.Write(bytesData, 0, bytesData.Length);
}

static void WriteHeader(FileStream fileStream, AudioClip clip)
{
    var hz = clip.frequency;
    var channels = clip.channels;
    var samples = clip.samples;

    fileStream.Seek(0, SeekOrigin.Begin);
    Byte[] riff = System.Text.Encoding.UTF8.GetBytes("RIFF");
    fileStream.Write(riff, 0, 4);

    Byte[] chunkSize = BitConverter.GetBytes(fileStream.Length - 8);
    fileStream.Write(chunkSize, 0, 4);

    Byte[] wave = System.Text.Encoding.UTF8.GetBytes("WAVE");
    fileStream.Write(wave, 0, 4);

    Byte[] fmt = System.Text.Encoding.UTF8.GetBytes("fmt ");
    fileStream.Write(fmt, 0, 4);

    Byte[] subChunk1 = BitConverter.GetBytes(16);
    fileStream.Write(subChunk1, 0, 4);

    UInt16 two = 2;
    UInt16 one = 1;

    Byte[] audioFormat = BitConverter.GetBytes(one);
    fileStream.Write(audioFormat, 0, 2);

```

```

    Byte[] numChannels = BitConverter.GetBytes(channels);
    fileStream.Write(numChannels, 0, 2);

    Byte[] sampleRate = BitConverter.GetBytes(hz);
    fileStream.Write(sampleRate, 0, 4);

    Byte[] byteRate = BitConverter.GetBytes(hz * channels * 2);
    fileStream.Write(byteRate, 0, 4);

    UInt16 blockAlign = (ushort)(channels * 2);
    fileStream.Write(BitConverter.GetBytes(blockAlign), 0, 2);

    UInt16 bps = 16;
    Byte[] bitsPerSample = BitConverter.GetBytes(bps);
    fileStream.Write(bitsPerSample, 0, 2);

    Byte[] datastring = System.Text.Encoding.UTF8.GetBytes("data");
    fileStream.Write(datastring, 0, 4);

    Byte[] subChunk2 = BitConverter.GetBytes(samples * channels * 2);
    fileStream.Write(subChunk2, 0, 4);
}
}

```

Файл Loading\_new\_content.cs:

```

using UnityEngine;
using UnityEngine.UI;
using System.Collections;
using System.IO;
using System;
using UnityEngine.SceneManagement;

public static class ButtonExtension
{
    public static void AddEventListener <T>(this Button button, T param, Action<T> OnClick)

```

```

    {
        button.onClick.AddListener(delegate
        {
            OnClick(param);
        });
    }
}

```

```

public class Loading_new_content : MonoBehaviour {

```

```

    public string url;
    [Header("UI Stuff")]
    public Transform rootContainer;
    public Button prefab;
    public Text labelText;

```

```

    AssetBundle assetBundle;

```

```

    IEnumerator Start ()

```

```

    {
        using (WWW www = new WWW(url))
        {
            yield return www;
            if (!string.IsNullOrEmpty(www.error))
            {
                Debug.LogError(www.error);
                yield break;
            }

            assetBundle = www.assetBundle;
            string[] scenes = assetBundle.GetAllScenePaths();

            foreach (string sceneName in scenes)
            {
                labelText.text = Path.GetFileNameWithoutExtension(sceneName);
            }
        }
    }

```

```

    var clone = Instantiate(prefab.gameObject) as GameObject;

    clone.GetComponent<Button>().AddEventListener(labelText.text,
LoadAssetBundleScene);
    clone.SetActive(true);
    clone.transform.SetParent(rootContainer);
    }
    }
}

public void LoadAssetBundleScene (string scenName)
{
    SceneManager.LoadScene(scenName);
}

```

Листинг разработанного программного обеспечения в полном объеме приведен на CD-диске.

Федеральное государственное автономное  
образовательное учреждение  
высшего образования  
«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»  
Институт космических и информационных технологий  
институт  
Вычислительная техника  
кафедра

УТВЕРЖДАЮ  
Заведующий кафедрой

 А.И. Легалов

подпись      инициалы, фамилия

« 14 » 06 2017 г.

**БАКАЛАВРСКАЯ РАБОТА**

09.03.01 Информатика и вычислительная техника  
код – наименование направления

Интеграция технологий виртуальной реальности в систему интерактивного  
обучения  
тема

Пояснительная записка

Руководитель

 — доцент, к.т.н.

А.В. Редкина

подпись, дата      должность, ученая степень      инициалы, фамилия

Выпускник

 12.06.2017

П.С. Мамаев

подпись, дата      инициалы, фамилия

Нормоконтролер

 14.06.17

В.И. Иванов

подпись, дата      инициалы, фамилия

Консультант

 12.06.2017

В.С. Будников

инициалы, фамилия

Красноярск 2017