TOMSK POLYTECHNIC UNIVERSITY ТОМСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

Министерство науки и высшего образования Российской Федерации
федеральное государственное автономное
образовательное учреждение высшего образования
«Национальный исследовательский Томский политехнический университет» (ТПУ)

Инженерная школа информационных технологий и робототехники
Направление подготовки 09.04.02 «Информационные системы и технологии»
Отделение информационных технологий

МАГИСТЕРСКАЯ ДИССЕРТАЦИЯ

Тема работы
Библиотека динамических ссылок для создания, управления и доступа к именованной общей памяти для взаимодействия приложений и языков программирования

УДК 004.451.2:004.451.7:031.43

Студент

Группа	ФИО	Подпись	Дата
8ИМ91	Гальярдо Паредес Кристиан Маурисио		

Руководитель ВКР

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Доцент ОИТ ИШИТР	Демин Антон Юрьевич	к.т.н		

КОНСУЛЬТАНТЫ ПО РАЗДЕЛАМ:

По разделу «Финансовый менеджмент, ресурсоэффективность и ресурсосбережение»

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Доцент ОСГН ШБИП	Верховская Марина Витальевна	к.э.н		

По разделу «Социальная ответственность»

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Профессор ООД ШБИП	Федоренко Ольга Юрьевна	д.м.н.		

ДОПУСТИТЬ К ЗАЩИТЕ:

Руководитель ООП	ФИО	Ученая степень, звание	Подпись	Дата
Доцент ОИТ ИШИТР	Савельев А.О.	к.т.н		

Томск – 2021 г.

ПЛАНИРУЕМЫЕ РЕЗУЛЬТАТЫ ОСВОЕНИЯ ООП

Код результатов	Результат обучения (выпускник должен быть готов)
Общепрофессиональные компетенции	
Р1	Воспринимать и самостоятельно приобретать, развивать и применять математические,естественнонаучные, социально-экономические и профессиональные знания длярешения нестандартных задач, в том числе в новой или незнакомой среде и вмеждисциплинарном контексте.
Р2	Владеть ипримен‌ять методы и средства получения, хранения, переработки и трансляцииинформации посредством современных компьютерных технологий, в том числе вглобальных компьютерных сетях.
Р3	Демонстрировать культуру мышления, способность выстраивать логику рассуждений ивысказываний, основанных на интерпретации данных, интегрированных из разныхобластей науки и техники, выносить суждения на основании неполных данных,анализировать профессиональную информацию, выделять в ней главное,структурировать, оформлять и представлять в виде аналитических обзоров собоснованными выводами и рекомендациями
Р4	Анализировать и оценивать уровни своих компетенций в сочетании со способностью иготовностью к саморегулированию дальнейшего образования и профессиональноймобильности. Владеть, по крайней мере, одним из иностранных языков на уровнесоциального и профессионального общения, применять специальную лексику ипрофессиональную терминологию языка.
Профессиональные компетенции	
Р5	Разрабатывать стратегии и цели проектирования, критерии эффективности и ограничения применимости, новые методы, средства и технологии проектирования геоинформационных систем (ГИС) или промышленного программного обеспечения.
Р6	Планировать и проводить теоретические и экспериментальные исследования в области создания интеллектуальных ГИС и ГИС технологии или промышленного программного обеспечения с использованием методов системной инженерии.
Р7	Осуществлять авторское сопровождение процессов проектирования, внедрения и сопровождения ГИС и ГИС технологий или промышленного программного обеспечения с использованием методов и средств системной инженерии, осуществлять подготовку и обучение персонала.
Р8	Формировать новые конкурентоспособные идеи в области теории и практики ГИС и ГИС технологий или системной инженерии программного обеспечения. Разрабатывать методы решения нестандартных задач и новые методы решения традиционных задач. Организовывать взаимодействие коллективов, принимать управленческие решения, находить компромисс между различными требованиями как при долгосрочном, так и при краткосрочным планировании.
Общекультурные компетенции	
Р9	Использовать на практике умения и навыки в организации исследовательских, проектных работ и профессиональной эксплуатации современного оборудования и приборов, в управлении коллективом.
Р10	Свободно пользоваться русским и иностранным языками как средством делового общения.
Р11	Совершенствовать и развивать свой интеллектуальный и общекультурный уровень. Проявлять инициативу, в том числе в ситуациях риска, брать на себя всю полноту ответственности.
Р12	Демонстрировать способность к самостоятельному обучению новым методам исследования, к изменению научного и научно-производственного профиля своей профессиональной деятельности, способность самостоятельно приобретать с помощью информационных технологий и использовать в практической деятельности новые знания и умения, в том числе в новых областях знаний, непосредственно не связанных со сферой деятельности, способность к педагогической деятельности.

TOMSK POLYTECHNIC UNIVERSITY ТОМСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

Министерство науки и высшего образования Российской Федерации
федеральное государственное автономное
образовательное учреждение высшего образования
«Национальный исследовательский Томский политехнический университет» (ТПУ)

Инженерная школа информационных технологий и робототехники
Направление подготовки 09.04.02 «Информационные системы и технологии»
Отделение информационных технологий

УТВЕРЖДАЮ:
Руководитель ООП
_____ _____ Савельев А.О.

ЗАДАНИЕ
на выполнение выпускной квалификационной работы

В форме:

Магистерской диссертации

(бакалаврской работы, дипломного проекта/работы, магистерской диссертации)

Студенту:

Группа	**ФИО**
8ИМ91	Гальярдо Паредес Кристиан Маурисио

Тема работы:

Библиотека динамических ссылок для создания, управления и доступа к именованной общей памяти для взаимодействия приложений и языков программирования	
Утверждена приказом директора (дата, номер)	09.02.2021, 40-4/с

Срок сдачи студентом выполненной работы:	

ТЕХНИЧЕСКОЕ ЗАДАНИЕ:

Исходные данные к работе *(наименование объекта исследования или проектирования; производительность или нагрузка; режим работы (непрерывный, периодический, циклический и т. д.); вид сырья или материал изделия; требования к продукту, изделию или процессу; особые требования к особенностям функционирования (эксплуатации) объекта или изделия в плане безопасности эксплуатации, влияния на окружающую среду, энергозатратам; экономический анализ и т. д.).*	Официальные литературные источники используемых инструментов и официальная информация Microsoft служили фактическим материалом для выполнения работы. Была также использована научная документация, полученная из статей, а также руководства по использованию и технические отчеты.
Перечень подлежащих исследованию, проектированию и разработке вопросов *(аналитический обзор по литературным источникам с целью выяснения достижений мировой науки техники в рассматриваемой области; постановка задачи исследования, проектирования, конструирования; содержание процедуры исследования, проектирования, конструирования; обсуждение результатов выполненной работы; наименование дополнительных разделов, подлежащих разработке; заключение по работе).*	- Анализ научно-технической документации, описывающей предметную область, а также ее различные типы реализации и области применения. (Межпроцессное взаимодействие - именованная общая память; библиотеки в операционной системе Windows; структура данных). - Разработка функций: создание и открытие общих воспоминаний с использованием принципов структуры данных и динамического

	программирования. - Разработка функций чтения и записи данных в общих памятях Разработка библиотек, содержащих созданные функции.
Перечень графического материала *(с точным указанием обязательных чертежей)*	- Схема архитектуры высокого уровня на модели телеоперации.

Консультанты по разделам выпускной квалификационной работы
(с указанием разделов)

Раздел	Консультант
Финансовый менеджмент, ресурсоэффективность и ресурсосбережение	Верховская Марина Витальевна
Социальная ответственность	Федоренко Ольга Юрьевна
Иностранный язык	Ануфриева Татьяна Николаевна

Названия разделов, которые должны быть написаны на русском и иностранном языках:

Literature review: Interprocess Communication - named shared memories.

Дата выдачи задания на выполнение выпускной квалификационной работы по линейному графику	

Задание выдал руководитель:

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Доцент ОИТ ИШИТР	Демин Антон Юрьевич	к.т.н		

Задание принял к исполнению студент:

Группа	ФИО	Подпись	Дата
8ИМ91	Гальярдо Паредес Кристиан Маурисио		

TOMSK POLYTECHNIC UNIVERSITY ТОМСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

Министерство науки и высшего образования Российской Федерации
федеральное государственное автономное
образовательное учреждение высшего образования
«Национальный исследовательский Томский политехнический университет» (ТПУ)

Инженерная школа информационных технологий и робототехники
Направление подготовки 09.04.02 «Информационные системы и технологии»
Уровень образования магистратура
Отделение информационных технологий
Период выполнения _____(осенний / весенний семестр 2020 /2021 учебного года)_____

Форма представления работы:

магистерская диссертация

(бакалаврская работа, дипломный проект/работа, магистерская диссертация)

КАЛЕНДАРНЫЙ РЕЙТИНГ-ПЛАН
выполнения выпускной квалификационной работы

Срок сдачи студентом выполненной работы:	

Дата контроля	Название раздела (модуля) / вид работы (исследования)	Максимальный балл раздела (модуля)
20.02.2021	1) Анализ изученности проблемы	
28.02.2021	2) Исследование по структурированному программированию	
15.03.2021	3) Исследование работы IPC - именованные общие памяти	
15.04.2021	4) Разработка функций управления общей памятью	
10.05.2021	5) Разработка динамических библиотек ссылок	
12.05.2021	6) Финансовый менеджмент, ресурсоэффективность и ресурсосбережение	
11.05.2021	7) Социальная ответственность	
21.05.2021	8) Раздел на английском языке «Literature review»	

СОСТАВИЛ:
Руководитель ВКР

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Доцент ОИТ ИШИТР	Демин Антон Юрьевич	к.т.н		

СОГЛАСОВАНО:

Руководитель ООП

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Доцент ОИТ ИШИТР	Савельев А.О.	к.т.н		

ЗАДАНИЕ ДЛЯ РАЗДЕЛА
«СОЦИАЛЬНАЯ ОТВЕТСТВЕННОСТЬ»

Студенту:

Группа	ФИО
8ИМ91	Гальярдо Паредес Кристиан Маурисио

Школа	ИШИТР	**Отделение (НОЦ)**	ОИТ
Уровень образования	Магистратура	**Направление/специальность**	Информационные системы и технологии

Тема ВКР:

Библиотека динамических ссылок для создания, управления и доступа к именованной общей памяти для взаимодействия приложений и языков программирования	
Исходные данные к разделу «Социальная ответственность»:	
1. Характеристика объекта исследования (вещество, материал, прибор, алгоритм, методика, рабочая зона) и области его применения	Объект исследования: Методы и технологии межпроцессного обмена информацией. Область применения: Доступность информации в режиме реального времени, когда функция список очерёдности данных не требуется. Рабочее место. Оборудование, используемое на рабочем месте: персональный компьютер, дополнительный монитор, мышь, зарядное устройство компьютера. Помещение офисного типа имеет следующие характеристики: площадь 32 квадратных метра, естественное и искусственное освещение, естественная вентиляция, отопление батареями.
Перечень вопросов, подлежащих исследованию, проектированию и разработке:	
1. Правовые и организационные вопросы обеспечения безопасности: - специальные (характерные при эксплуатации объекта исследования, проектируемой рабочей зоны) правовые нормы трудового законодательства; - организационные мероприятия при компоновке рабочей зоны.	- Трудовой кодекс Российской Федерации от 30.12.2001 N 197-ФЗ. (ред. От 24.04.2020). - СанПиН 1.2.3685-21 Гигиенические нормативы и требования к обеспечению безопасности и (или) безвредности для человека факторов среды обитания. - ГОСТ 12.0.003-2015 Опасные и вредные производственные факторы. Классификация. Перечень опасных и вредных факторов. - СанПиН 2.2.4.548-96 Гигиенические требования к микроклимату производственных помещений. - СП 52.13330.2016 Естественное и искусственное освещение.

	Актуализированная редакция СНиП 23-05-95. - СН 2.2.4/2.1.8.562-96. Шум на рабочих местах, в помещениях жилых, общественных зданий и на территории жилой застройки . - ГОСТ 12.1.003-2014 ССБТ. Шум. Общие требования безопасности - ГОСТ 12.1.003-83 Система стандартов безопасности труда (ССБТ). Шум. Общие требования безопасности. - ГОСТ 12.2.032-78 ССБТ Рабочее место при выполнении работ сидя. Общие эргономические требования. - ГОСТ Р 50923-96. Дисплеи. Рабочее место оператора. Общие эргономические требования и требования к производственной среде. Методы измерения - ТОИ Р-45-084-01 Типовая инструкция по охране труда при работе на персональном компьютере. - ГОСТ Р 12.1.019-2017 ССБТ Электробезопасность. Общие требования и номенклатура видов защиты. - ГОСТ 12.1.038-82 Система стандартов безопасности труда (ССБТ). Электробезопасность. Предельно допустимые значения напряжений прикосновения и токов. - ГОСТ 12.1.004-91 Система стандартов безопасности труда (ССБТ). Пожарная безопасность. Общие требования.
2. Производственная безопасность: 2.1. Анализ выявленных вредных и опасных факторов 2.2. Обоснование мероприятий по снижению воздействия	Вредные факторы: - Повышенная или пониженная температура и относительная влажность воздуха. - Превышение уровня шума. - Отсутствие или недостаток освещения. - Психофизиологические факторы (монотонность труда, умственное и эмоциональное напряжение, перенапряжение зрительных анализаторов). Опасные факторы: - Поражение электрическим током; - Короткое замыкание;

	- Статическое электричество.
3. Экологическая безопасность:	Негативное воздействие на литосферу происходит при утилизации компьютера и периферийных устройств (принтеры, МФУ, веб-камеры, наушники, колонки, телефоны), люминесцентных ламп, макулатуры.
4. Безопасность в чрезвычайных ситуациях:	- Возможные ЧС: пожары, грозы, ураганы, оползни. - Наиболее типичная ЧС: пожар.

Дата выдачи задания для раздела по линейному графику	**01.03.2021**

Задание выдал консультант:

Должность	**ФИО**	**Ученая степень, звание**	**Подпись**	**Дата**
Профессор ООД ШБИП	Федоренко Ольга Юрьевна	д.м.н.		

Задание принял к исполнению студент:

Группа	**ФИО**	**Подпись**	**Дата**
8ИМ91	Гальярдо Паредес Кристиан Маурисио		

ЗАДАНИЕ ДЛЯ РАЗДЕЛА
«ФИНАНСОВЫЙ МЕНЕДЖМЕНТ, РЕСУРСОЭФФЕКТИВНОСТЬ И РЕСУРСОСБЕРЕЖЕНИЕ»

Студенту:

Группа	ФИО
8ИМ91	Гальярдо Паредес Кристиан Маурисио

Школа	ИШИТР	**Отделение школы (НОЦ)**	ОИТ
Уровень образования	Магистратура	**Направление/специальность**	Информационные системы и технологии

Исходные данные к разделу «Финансовый менеджмент, ресурсоэффективность и ресурсосбережение»:	
1. *Стоимость ресурсов научного исследования (НИ): материально-технических, энергетических, финансовых, информационных и человеческих*	Компьютер, дополнительный монитор, периферийные устройства, интернет, электричество, рабочая сила, канцелярские принадлежности.
2. *Нормы и нормативы расходования ресурсов*	
3. *Используемая система налогообложения, ставки налогов, отчислений, дисконтирования и кредитования*	– коэффициент отчислений во внебюджетные фонды – 30 %.
Перечень вопросов, подлежащих исследованию, проектированию и разработке:	
1. *Расчёт инновационного потенциала НТИ*	– SWOT-анализ; – оценка научного уровня исследования.
2. *Расчёт сметы затрат на выполнение проекта*	– расчёт материальных затрат; – расчёт основной и дополнительной заработной платы: – расчёт отчислений во внебюджетные фонды; – расчёт бюджета проекта.
Перечень графического материала *(с точным указанием обязательных чертежей)***:**	
1. *Оценка конкурентоспособности технических решений* 2. *Матрица SWOT* 3. *График проведения НТИ* 4. *Оценка ресурсной, финансовой и экономической эффективности разработки*	

Дата выдачи задания для раздела по линейному графику	

Задание выдал консультант:

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Доцент ОСГН ШБИП	Верховская Марина Витальевна	к.э.н		

Задание принял к исполнению студент:

Группа	ФИО	Подпись	Дата
8ИМ91	Гальярдо Паредес Кристиан Маурисио		

РЕФЕРАТ

Выпускная квалификационная работа 106 с., 25 рис., 23 табл., 52 источников, 1 текстовое приложение.

Ключевые слова: Именованные общие памяти, библиотеки динамических ссылок, взаимодействие между процессами, совместимость, библиотеки Windows.

Объектом исследования является процесс разработки библиотек динамических ссылок и использование именованной общей памяти для высокоскоростной связи между процессами.

Целью работы является разработка библиотеки динамических ссылок, содержащих функции создания, доступа, чтения, записи и выпуска именованных общих памятей. Именованные общие памяти будут использоваться в качестве метода высокоскоростного межпроцессного соединения.

Исследовательский процесс фокусируется на анализе сценариев, требующих высокоскоростного обмена данными, в частности, объектом исследования являются коммуникационные архитектуры, используемые для телеоперации. Эти сценарии были изучены как пример, в котором скорость связи играет фундаментальную роль. Из всей архитектуры связи в модели телеоперации наиболее важным моментом, который был проанализирован, является доступность данных на локальном сайте.

В результате выполнения работы было разработано несколько библиотек динамической привязки как для 32-разрядных, так и для 64-битных операционных систем Windows, а также разработаны библиотеки под различными конвенционными вызовами (Cdecl, Fastcall, Stdcall).

Область применения: данное приложение предназначено для сценариев, в которых основным требованием является доступность информации в режиме реального времени. Данные не помещаются в список ожидания при хранении, последнее записанное значение заменяет предыдущее.

Значимость работы заключается в улучшении скорости обмена информацией и повышении совместимости между различными типами клиентов в ситуациях, когда время отклика является решающим моментом. Сокращение времени отклика снижает вероятность несчастных случаев из-за задержки доступа к данным о состоянии устройств или систем.

Содержание

Перечень условных обозначений

IPC	Inter-process Communication
IDE	Integrated Development Environment
DLL	Dynamic Link Library
JDK	Java Development Kit
IP	Internet Protocol
TCP	Transmission Control Protocol
UDP	User Datagram Protocol
GNU	GNU's Not Unix
BSD	Berkeley Software Distribution
RPC	Remote Procedure Call
COM	Component Object Model
OLE	Object Linking & Embedding
DCOM	Distributed Component Object Model
IPX	Internetwork Packet Exchange
SPX	Sequenced Packet Exchange
FIFO	First in, first out
API	Application Programming Interface
IBM	International Business Machines
NetBIOS	Network Basic Input/Output System
POSIX	Portable Operating System Interface
OSI	Open Systems Interconnection
GCC	GNU Compiler Collection
API	Application Binary Interface
JNI	Java Native Interface
DOF	Degree Of Freedom
UAV	Unmanned aerial vehicle
DTW	Dynamic time warping

ВВЕДЕНИЕ

В конкретных сценариях, в которых время отклика играет ведущую роль, использование высокоскоростных механизмов обмена данными является решающим моментом, и разработчики вынуждены улучшать качество связи своих процессов.

В анализируемых исследовательских группах отмечается использование традиционных методов обмена данными. Наиболее широко используемыми межпроцессном взаимодействием (IPC) являются сетевые протоколы TCP или UDP, также ценится использование протокола Websocket, хотя он является протоколом, ориентированным на подключение через интернет, его скорей удобство использования приводит к тому, что несколько продуктов разрабатывают клиенты для локального обмена данными на одном компьютере. Такие методы IPC обладают большим количеством механизмов для обеспечения целостности данных также, как они обладают списками ожидания для передачи данных, они также обеспечивают безопасность в шифровании связи и других механизмах безопасности. Все механизмы, описанные выше, требуют времени на их выполнение, и это время увеличивает время отклика, необходимое для обмена информацией.

Целью работы является разработка библиотек динамических ссылок, содержащих именованные функции обработки общей памяти для связи между процессами, потоками или приложениями внутри компьютера. Это приложение также может быть использовано для обмена данными между различными языками программирования.

В первом разделе составляется список наиболее часто используемых IPCs в операционной системе Windows, добавляется краткое описание каждого из них.

Во втором разделе проводится исследование по использованию библиотек в операционной системе Windows, исследование проводится как для

динамических, так и для статических библиотек ссылок.

В третьем разделе представлены алгоритмы управления оперативной памятью, а также представлена разработка функций управления именованными общей памятью.

В четвертом разделе представлен процесс разработки библиотек динамических ссылок, содержащих функции управления именованными общей памятью.

В пятом разделе представлены различные экспериментальные тесты, проведенные с использованием библиотек, предложенных в работе.

1 Типы межпроцессных взаимодействий

Межпроцессное взаимодействие (IPC) присутствует в вычислениях с момента его создания, поскольку оно является основным ресурсом, используемым для обмена сообщениями между процессами либо внутри одного компьютера, либо между несколькими узлами сети. Использование IPC может наблюдаться в основном при синхронизации действий или событий посредством отправки сообщений или прерываний, но, имея характеристику обмена данными, они также могут использоваться для обмена информацией между вашими клиентами. Кроме того, в области вычислительной техники существуют специализированные приложения, которые, работая в координации, могут решать сложные проблемы, связанные с междисциплинарными проектами или требующие распределения обработки, и поэтому необходимо использовать некоторый метод взаимосвязи процессов.

Среди наиболее часто используемых IPC есть сокеты, будь то TCP или UDP, они широко используются для обмена информацией в сценариях, которые требуют сложных алгоритмов обеспечения данных между отправителем и получателем. В средах отправки сигналов и отправки прерываний наиболее часто используемыми IPC являются общие памяти и отправка сообщений [1–4].

Общие памяти имеют большие преимущества по сравнению с другими процессами межпроцессного взаимодействия на уровне скорости обмена, поэтому с самого начала они широко использовались, особенно в операционных системах GNU/Linux и BSD [5–7]. Среди альтернатив для связи между процессами существуют: буфер обмена, компонентная объектная модель, копирование данных, динамический обмен данными, почтовый ящик, каналы, RPC и другие. Механизмы IPC существуют во всех операционных системах, но для данной разработки основное внимание уделяется анализу механизмов, используемых на платформах Windows.

1.1 Обзор технической документации по функциям IPC в операционной системе Windows

1.1.1 Использование буфера обмена в качестве IPC

Буфер обмена - это пространство, используемое для передачи текста, данных, файлов или объектов из исходного места в место назначения. Его наиболее распространенное использование встречается среди текстовых редакторов для обмена текстовыми хотя он также может использоваться для обмена информацией между процессами, которые могут получить доступ к пространству буфера обмена.

Когда пользователь выполняет операцию вырезания или копирования в приложении, приложение помещает выбранные данные в буфер обмена в одном или нескольких стандартных или определенных приложением форматах, и любое другое приложение может извлекать данные из буфера обмена, выбирая из доступных форматов, приложения могут находиться на одном компьютере или удаленных компьютерах [8].

1.1.2 Использование компонентной объектной модели (COM) в качестве IPC

COM – это платформа, созданная Microsoft и используемая для связи между процессами. Термин COM охватывает многие технологии, включая OLE, OLE Automation, ActiveX, COM+ и DCOM.

COM предназначен для того, чтобы клиенты могли прозрачно взаимодействовать с другими объектами независимо от того, где они работают, будь то на том же процессе, на том же компьютере или на другом компьютере.

DCOM – это расширение Компонентной объектной модели на распределенные среды, которое определяет механизмы подключения и сетевой протокол, необходимые для выполнения вызовов удаленных процедур. Эта функция делает ее полезной для распределенных систем всех видов, основанных

на компонентах.

Разработчики приложений должны быть осторожны, чтобы их приложения были как можно более независимыми от сетевой инфраструктуры, но многие распределенные приложения должны быть интегрированы в инфраструктуру существующей сети, что требует определенного сетевого протокола, и это потребует адаптации от всех клиентов, что во многих ситуациях неприемлемо. Благодаря языковой независимости DCOM разработчики приложений могут выбирать инструменты и языки программирования, с которыми они наиболее знакомы, поскольку DCOM обеспечивает такую прозрачность: DCOM может использовать любой транспортный протокол, такой как TCP/IP, UDP, IPX/SPX и NetBIOS [9].

1.1.3 Использование сопоставления файлов (File Mapping) в качестве IPC

File mapping – это часть виртуальной памяти, в которой устанавливается прямое байтовое сопоставление с частью файла или аналогичного ресурса. Этот ресурс обычно представляет собой файл на жестком диске, объект общей памяти или другие типы ресурсов, на которые операционная система может ссылаться через файловый дескриптор. Как только это сопоставление между файлом и пространством памяти доступно, приложения могут управлять доступом к этому ресурсу точно так же, как если бы это была основная память.

Сопоставление файлов позволяет процессу обрабатывать содержимое файла, как если бы это был блок памяти в адресном пространстве процесса. Процесс может использовать простые операции с указателями для просмотра и изменения содержимого файла. Когда два или более процессов обращаются к одному и тому же распределению файлов, каждый процесс получает указатель памяти в своем собственном адресном пространстве, который он может использовать для чтения или изменения содержимого файла. Процес-

сы должны использовать объект синхронизации, например семафор, чтобы избежать повреждения данных в многозадачной среде [10].

1.1.4 Использование именованной общей памяти в качестве IPC

Именованная общая память – это тип сопоставления файлов, в котором блок памяти может быть указан в качестве файла хранения данных. Другие процессы могут получить доступ к тому же блоку памяти, открыв тот же объект выделения файлов, что представляет значительное преимущество по сравнению с работой с физическими файлами, поскольку файл, проецируемый в память, имеет гораздо более быстрое время отклика, поскольку вся его информация хранится в оперативной памяти.

Файлы, проецируемые в память, загружают полную страницу в память за один раз, и размер страницы определяется операционной системой для оптимальной производительности [10].

1.1.5 Использование почтового слота (Mailslot) в качестве IPC

Mailslot – это псевдофайл, который находится в памяти и предлагает механизм односторонней связи между процессами (IPC), причем процессы могут быть как локальными, так и удаленными. Концептуально mailslots похожи на именованные каналы Windows и очереди сообщений POSIX, но их использование намного проще, особенно при обработке относительно небольшого количества коротких сообщений. Данные из сообщения в почтовый ящик могут быть любого формата, но максимальный размер сообщений, которые могут быть отправлены, составляет 424 байта.

Mailslots работают в модели клиент-сервер, где сервер mailslot – это процесс, который может создавать mailslot и, следовательно, тот, кто может читать сообщения. Только сервер, создавший почтовый ящик, или процессы, которые его наследуют, могут читать сообщения. Одна из особенностей, которую следует учитывать, заключается в том, что сервер может создавать

почтовые слоты только локально, он не может создавать удаленный почтовый слот.

Клиент mailslot – это процесс, который может записывать сообщения в почтовый ящик по имени. Этот механизм позволяет отправлять короткие широковещательные сообщения, которые будут прослушиваться всеми компьютерами, имеющими общий сетевой домен [11].

1.1.6 Использование труб для IPC в качестве IPC

В вычислительной технике трубопровод (трубопровод или канал) состоит из цепочки процессов, соединенных таким образом, что выход каждого элемента цепочки является входом следующего. Они позволяют осуществлять связь и синхронизацию между процессами, и использование буферов данных между последовательными элементами является обычным явлением.

Связь по каналам основана на взаимодействии производителя и потребителя, процессы-производители (те, которые отправляют данные) взаимодействуют с процессами-потребителями (которые получают данные) в соответствии с порядком FIFO. Как только процесс-потребитель получает данные, он удаляется из конвейера.

Безымянные каналы имеют файл, связанный с ними в основной памяти, поэтому они являются временными и удаляются, когда они не используются производителями или потребителями. Они позволяют осуществлять связь между процессом, создающим канал, и дочерними процессами после создания конвейера.

Именованные каналы создают причину в файловой системе и, следовательно, не являются временными. Они обрабатываются системными вызовами (открытие, закрытие, чтение и запись), как и остальные системные файлы. Они позволяют осуществлять связь между процессами, использующими указанный конвейер, хотя между ними нет иерархической связи [12].

1.1.7 Использование сокетов Windows в качестве IPC

Сокет обозначает абстрактную концепцию, с помощью которой два процесса (возможно, расположенные на разных компьютерах) могут обмениваться любым потоком данных, как правило, надежным и упорядоченным образом. Термин сокет также используется в качестве названия интерфейса прикладного программирования (API) для семейства интернет-протоколов TCP/IP, обычно предоставляемых операционной системой. Интернет-сокеты-это механизм доставки пакетов данных с сетевой карты в соответствующие процессы или потоки. Сокет определяется парой локальных и удаленных IP-адресов, транспортным протоколом и парой локальных и удаленных номеров портов.

Сокеты Windows основаны на сокетах, впервые популяризированных Berkeley Software Distribution (BSD). Приложение, использующее сокеты Windows, может взаимодействовать с другой реализацией сокетов в других типах систем. Однако не все поставщики транспортных услуг поддерживают все доступные варианты [13].

2 Использование библиотек в windows

В вычислительной технике библиотека (от английского языка “library”), представляет собой набор функциональных реализаций, закодированных на языке программирования, который предлагает четко определенный интерфейс для вызываемой функциональности. В отличие от исполняемой программы, поведение, реализуемое библиотекой, не предполагает автономного использования, поскольку, в то время как исполняемая программа может выполняться индивидуально, поскольку она имеет основную точку входа, цель библиотеки состоит в том, чтобы использоваться другими программами. Библиотеки могут быть связаны с программой (или с другой библиотекой) на разных этапах разработки или выполнения, в зависимости от типа связи,

которую требуется установить.

Создание и использование общих библиотек родились вместе с первыми базовыми операционными системами в истории, есть записи о повторном использовании функций из системы ввода-вывода GM-NAA (input/output) General Motors и North American Aviation, которая была операционной системой, созданной для IBM 704. Появление общих библиотек было связано с необходимостью занимать как можно меньше места и памяти, создавая функции, которые использовались повторно [14, 15].

На сегодняшний день можно найти использование библиотек, особенно в управлении интерфейсами и использовании драйверов оборудования, позволяющих всем приложениям операционной системы совместно использовать их, экономя диск, память и ресурсы обработки. Эти функции возможны, поскольку исходный код уникален и физически отделен от приложений, позволяя каждому приложению использовать эти функции без необходимости копировать код в каждом из них. На уровне времени отклика следует отметить, что использование библиотек значительно увеличивает скорость работы операционных систем, это связано с тем, что библиотеки создаются один раз в оперативной памяти, и их функции могут использоваться из любого приложения без необходимости повторного создания экземпляров, что повышает производительность не только программ, использующих библиотеки, но и всей операционной системы [16–18].

2.1 Статические библиотеки

Статическая библиотека – это файл-контейнер с несколькими упакованными файлами объектного кода, во время выполнения эти библиотеки создают статическую связь с приложением, которое их использует, эта связь известна как “статическая конструкция”, “раннее связывание” или “статическое связывание”. Когда приложение связано с библиотекой, оно может использовать функции, содержащиеся в библиотеке.

Статическая библиотека в процессе разработки просто действует как контейнер для файлов объектного кода, которые не отличаются (кроме семантических) от промежуточных объектных файлов, созданных на предыдущем этапе компиляции программы. В области программирования статическая библиотека “копируется” в нашу программу при ее компиляции, после того, как исполняемый файл программы генерируется, библиотека больше не требуется. При удалении статической библиотеки скомпилированная программа будет продолжать работать, поскольку она имеет копию необходимых функций статической библиотеки. Важным моментом является то, что при компиляции нашей программы копируется только та часть библиотеки, которая необходима. Например, если библиотека имеет две функции, а наша программа вызывает только одну, копируется только требуемая функция.

2.2 Динамические библиотеки

Динамические библиотеки, динамически связанные, с динамическими ссылками или обычно называемые DLL, представляют собой файлы, содержащие сконструированный объектный код, который подготовлен для использования и загружается во время выполнения различными программами одновременно. Поэтому они должны быть доступны в виде файлов, независимых от исполняемой программы (как правило, в системных каталогах).

Одним из больших преимуществ динамической ссылки по отношению к статической ссылке является то, что она позволяет повторно использовать не только код, но и физическое пространство: один и тот же файл общей библиотеки может использоваться несколькими программами без копирования его содержимого в них. С другой стороны, самым большим недостатком динамической связи является увеличение времени загрузки (из-за необходимости искать файл библиотеки, загружать его и перемещать вызовы в программе), а также увеличение времени из-за косвенного обращения ко времени вызова библиотечных функций.

В области программирования код библиотеки динамических ссылок не копируется в нашу программу при компиляции. Каждый раз, когда приложение запускается, и ему нужна функция библиотеки динамических ссылок, оно будет использовать файл библиотеки, физически отделенный от приложения. При удалении библиотеки динамических ссылок программа покажет ошибку, так как она не может найти библиотеку.

3 Разработка функций управления именованных общей памятью

При анализе скорости работы IPCs, следует отметить, что большинство IPCs используют сложные функции шифрования, очереди данных или продвинутые архитектуры, такие как сетевые протоколы, использующие стек OSI. Эти сложные архитектуры требуют времени обработки, к этому добавляется, что многие IPCs работают на жестком диске, который является очень медленным периферийным устройством по сравнению со скоростью памяти компьютера. При сравнении скорости передачи жестких дисков получены следующие результаты:

- Традиционный механический жесткий диск обрабатывает скорость передачи данных от 40Мбит/с до 90Мбит/с.
- Традиционный твердотельный диск может достигать около 500МБ/с, самые быстрые твердотельные диски M2 или технология NVMe могут достигать 3000МБ/с или 3ГБ/с.

Скорость передачи оперативной памяти, например, DDR4-3200 обрабатывает скорость 25,6ГБ/с и может выполнять 3200Мт/с.

МТ/с (мегатрансфер в секунду) – это показатель переводов в секунду, 1 МТ/с-это 10^6 или один миллион переводов в секунду.

В результате использование памяти для передачи информации является самым быстрым возможным вариантом, если не требуются функции очереди данных или сложные средства контроля целостности данных [19–22].

3.1 Используемые инструменты

Для проведения анализа и разработки различных решений были использованы следующие инструменты.

Codelite IDE – это бесплатная кроссплатформенная среда разработки с открытым исходным кодом для языков C / C++, которая использует wxWidgets для своего графического интерфейса. Чтобы соответствовать духу с открытым исходным кодом, CodeLite компилируется и отлаживается с использованием только бесплатных инструментов (MinGW и GDB).

CodeLite предлагает управление проектами (рабочими пространствами/проектами), завершение кода, навигацию по исходным файлам, подсветку синтаксиса, интеграцию с Subversion, Cscope и UnitTest++, интерактивный отладчик на основе gdb и мощный редактор кода на основе Scintilla.

CodeLite имеет лицензию GNU General Public License v2 или более поздней версии [23].

Java Development Kit (JDK) – это пакет разработки для Java, то есть, это набор инструментов (библиотек и программ), которые позволяют разрабатывать на языке Java. Когда речь идет о разработке, это включает в себя компиляцию, выполнение, создание документации и многое другое [24].

Компилятор TDM-GCC: MinGW (минималистский GNU для Windows), ранее известный как MinGW32, является реализацией компиляторов GCC для платформы Win32, которая позволяет переносить емкость этого компилятора в средах Windows. Это вилка Cygwin в версии 1.3.3. MinGW включает в себя набор Win32 API, позволяющий разрабатывать собственные приложения для этой платформы, а также создавать исполняемые файлы и библиотеки с помощью Windows API [25].

Средство просмотра экспорта (DLL Export Viewer) – это утилита которая отображает список всех экспортированных функций и их адреса виртуальной памяти для указанных файлов DLL [26].

3.2 Разработка функций управления именованными общими памятями

Все функции для создания, управления, доступа, чтения, записи и удаления общих памяти объявлены в файле .h, а их соответствующие определения реализованы в исходном файле с расширением .c.

Функции изолированы таким образом, что их можно вызывать в других структурах кода. Например, один и тот же код функцией может быть импортирован в общий проект разработки, проект разработки общей библиотеки и проект разработки библиотеки Java без необходимости изменения выполняемых функций, как показано на рисунке 3.1.

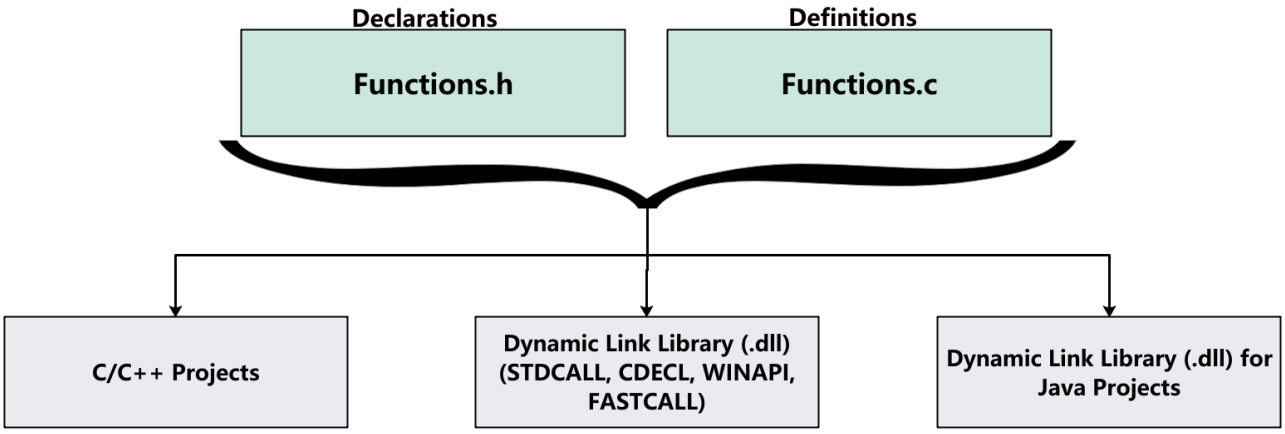

Рисунок 3.1 – Схема возможного использования функций, созданных из их исходных файлов

Существуют два типа распределения памяти для использования переменных в приложении: статическое и динамическое распределение. При статическом выделении память резервируется во время выполнения, этот объем памяти, доступный для использования, не может быть увеличен, поэтому необходимо использовать определенное количество переменных.

С другой стороны, динамическое выделение памяти может управлять памятью, необходимой для использования переменными внутри программы во время выполнения. Говоря об управлении памятью, можно создавать и освобождать память во время выполнения; например, во встраиваемых си-

стемах память считается наиболее ценным ресурсом, поскольку устройства не могут быть оснащены большими объемами вычислительной памяти, по этой причине динамическое управление памятью является фундаментальным моментом для управления количеством требуемых переменных и объемом памяти, который им требуется при запуске приложения [27, 28].

Для предлагаемой модели внутренней связи используется динамическое распределение памяти, что обеспечивает функциональность создания и удаления общих воспоминаний во время выполнения. Вся разработка будет выполняться на языке программирования C, и поскольку этот язык не имеет сборщика мусора, функции должны быть запрограммированы на освобождение памяти, когда они не требуются [29].

В жизни именованной общей памяти есть два важных момента: ее создание и доступ к ней. Хотя для создания каждой именованной общей памяти требуется только использование функции, для доступа к каждой именованной общей памяти необходимо выполнить два шага. Сначала необходимо создать дескриптор, указывающий на желаемую общую память, а затем создать представление общей памяти с помощью дескриптора, созданного на первом шаге.

Для выполнения процессов, описанных выше, для каждого процесса используется тип структуры (узел) :

- Тип узла, который создает общие памяти и добавляет их в связанный список. Структура узла показана в листинге 1.

- Узел обработчика типа, который указывает на каждую желаемую общую память. Необходимо управлять новым узлом для каждой памяти, которая требуется доступ. Структура узла показана в листинге 2.

- Узел представления типа, который указывает на узел дескриптора, содержащий указатель на доступ к общей памяти. Структура узла показана в листинге 3.

Листинг 1 – Код создания узла для создания общих памятей

```
35 struct memoryNode{
36         HANDLE memoryHandle;     //A handle to the file from which to create
37                                  // a file mapping object.
38         struct memoryNode *pNextMemoryNode;
39 };
```

Листинг 2 – Код создания узла для доступа к существующей общей памяти

```
35 struct mappingNode{
36     HANDLE mapHandle;           // handle to the specified file mapping
37                                 // object.
38     struct mappingNode *pNextMapNode;
39 };
```

Листинг 3 – Код создания узла для создания представления в общую память

```
35 struct viewNode{
36     void *pViewHandle;  //is the starting address of the mapped view
37     char viewName[MEM_NAME_LIMIT];  //The name of the Named Shared Memory
38     struct viewNode *pNextView;
39 };
```

3.2.1 Создание Именованной Общих памятей

Для создания каждой общей памяти требуется три параметра:

– Имя общей памяти

– Количество управляемых пространств

– Тип создаваемой общей памяти. Тип общей памяти имеет следующий код:

- 1: Общая память типа Integer
- 2: Общая память типа Float
- 3: Общая память типа Double
- 4: Общая память типа String (По умолчанию для каждого общего пространства памяти типа String устанавливается предельный размер в 64 символа хранения)

Объявление функции, создающей общие памяти, показано в листинге 4.

Листинг 4 – Объявление функции, которая создает общие памяти

```
127 int create_Memory(char *memoryName, int quantity, int typeCode);
```

Чтобы предложить функцию создания общих памятей во время выполнения, предлагается использовать концепцию связанных списков, и, таким образом, можно создать столько общих памятей, сколько требуется пользователю. Алгоритм, код и краткое объяснение исходного кода показаны на рис. 3.2.

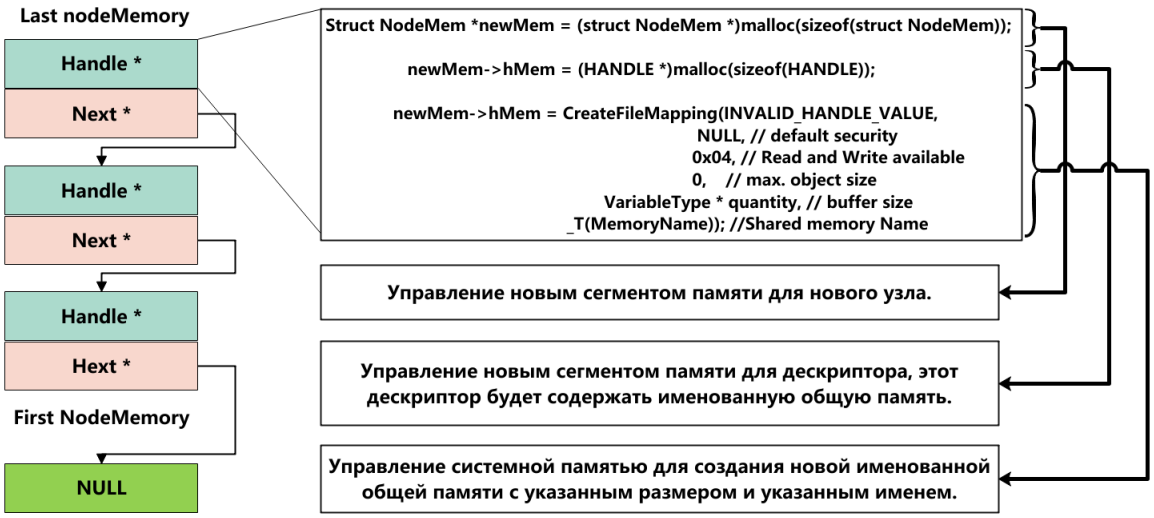

Рисунок 3.2 – Управление и создание именованных общих памятей

Общий размер управляемой памяти – это сумма всех созданных памятей зная, что каждая общая память принадлежит типу (Integer, Float, Double, String) и имеет одно или несколько мест хранения.

Приведен следующий пример, в котором создано несколько общих памятей различных типов и различной емкости хранения, всего создано 7:

- Создается память с именем “Memory1”, которая будет хранить значения типа Integer и будет иметь место для хранения 2 значений.
- Создается память с именем “Memory2”, которая будет хранить значения типа String и будет иметь место для хранения 2 значений.

– Создается память с именем “Memory3”, которая будет хранить значения типа Double и будет иметь место для хранения 2 значений.

– Создается память с именем “Memory4”, которая будет хранить значения типа Float и будет иметь место для хранения 2 значений.

– Создается память с именем “Memory5”, которая будет хранить значения типа Integer и будет иметь место для хранения 2 значений.

– Создается память с именем “Memory6”, которая будет хранить значения типа String и будет иметь место для хранения 2 значений.

– Создается память с именем “Memory7”, которая будет хранить значения типа Double и будет иметь место для хранения 2 значений.

Код построения для каждой общей памяти приведен в листинге 5.

Листинг 5 – Код создания общих памятей

```
12 create_Memory(Memory1, 2, 1);
13 create_Memory(Memory2, 4, 4);
14 create_Memory(Memory3, 2, 3);
15 create_Memory(Memory4, 2, 2);
16 create_Memory(Memory5, 2, 1);
17 create_Memory(Memory6, 4, 4);
18 create_Memory(Memory7, 2, 3);
```

Примечание: Каждая память может быть создана с именем и количеством сохраняемых значений, необходимых пользователю.

Расчет общего объема физической памяти, используемой для хранения данных предложенного примера.

Для расчета общего объема управляемой памяти используется следующая таблица, в которой указан размер переменных в зависимости от их типа, на таблице 3.1.

Таблица 3.1 – размер различных типов переменных (в соответствии с компилятором языка Си)

Тип	Описание
Integer	4 byte (-2,147,483,648 to 2,147,483,647)
Float	4 byte (1.2E-38 to 3.4E+38)
Double	8 byte (2.3E-308 to 1.7E+308)
String	64 byte (Настроено хранение по умолчанию 64 символов на пространство памяти типа String)

Для расчета общего объема управляемой памяти используется следующая формула 3.1.

$$TotalBytes = \sum_{i=0}^{a} (MemoryType * S_i) \tag{3.1}$$

Где переменная a - это количество управляемых именованных общих памятей, а переменная S - количество пространств, управляемых для хранения в текущей памяти.

Расчет будет решаться следующим образом:

$$Memory1 = 4_{(bytes)} * 2 => \qquad Memory1 = 8_{(bytes)}$$
$$Memory2 = 64_{(bytes)} * 4 => \qquad Memory2 = 128_{(bytes)}$$
$$Memory3 = 8_{(bytes)} * 2 => \qquad Memory3 = 16_{(bytes)}$$
$$Memory4 = 4_{(bytes)} * 2 => \qquad Memory4 = 8_{(bytes)}$$
$$Memory5 = 64_{(bytes)} * 4 => \qquad Memory5 = 256_{(bytes)}$$
$$Memory6 = 8_{(bytes)} * 2 => \qquad Memory6 = 8_{(bytes)}$$

Total Bytes = 8 Bytes + 128 Bytes + 16 Bytes + 8 Bytes + 256 Bytes + 8 Bytes = 424 Bytes

Для создания вышеупомянутых 7 общих памятей потребуется 424 байта. Ограничение на объем памяти, который может быть создан, будет ограничено объемом памяти, доступной в системе.

Если требуется создание общих памятей только для хранения число-

вых значений, был бы создан массив, но когда требуется создание общих памятей значения типа string, получен объект, который хранит значения в трех измерениях. Рис. 3.3.

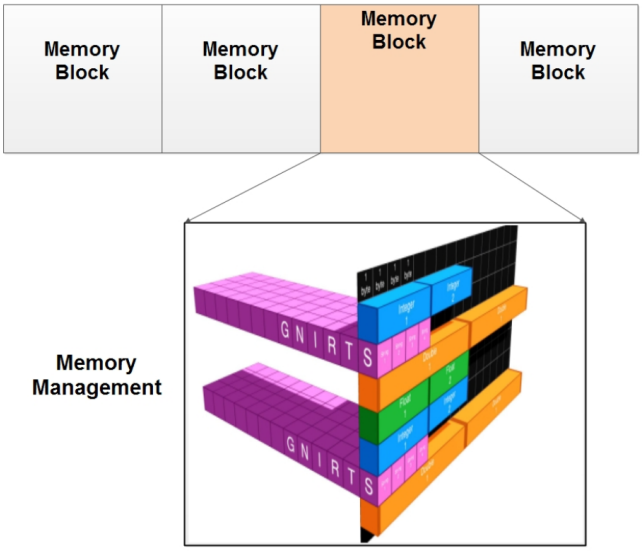

Рисунок 3.3 – Структура общих памятей, созданных в примере

3.2.2 Доступ к именованным общим памятям

После создания именованных общих памятей клиенты должны иметь возможность доступа к содержимому созданных общих памятей и управления им. Для доступа к именованной общей памяти необходимо выполнить два процесса:

Во-первых, требуется открыть объект типа ”сопоставление файлов“ с помощью функции OpenFieMapping(), чтобы использовать эту функцию, нам нужно знать имя именованной общей памяти, к которой требуется получить доступ.

Во-вторых, необходимо использовать функцию, которая назначает представление файла адресному пространству вызывающего процесса, для этого используется функцию MapViewOfFie(), среди ее параметров требуется отправить обработчик, был создан на первом шаге. Для создания узлов, хранящих представления, предлагается использовать аморфную переменную (тип

Void), которая может быть преобразована в любой тип указателя типа переменной, который нам нужен. Например, один и тот же тип узла может быть преобразован в примитивные типы данных, такие как целые числа, поплавки, двойники, но в то же время можно создавать различные типы переменных в виде структур.

Эти два процесса выполняются в соответствии с концепцией динамического управления памятью и создания связанных списков как для каждого узла типа “сопоставление файлов” , так и для каждого узла типа представления.

Процесс открытия общих памятей показан на рис. 3.4. В верхней части изображения описан узел типа дескрипторов “сопоставление файлов”, которые указывают на общую память, к которой требуется получить доступ. В нижней части изображения описан узел типа представления, которые используют узлы типа дескрипторов. Именно через узлы типа представления можно получить доступ к данным общих памятей.

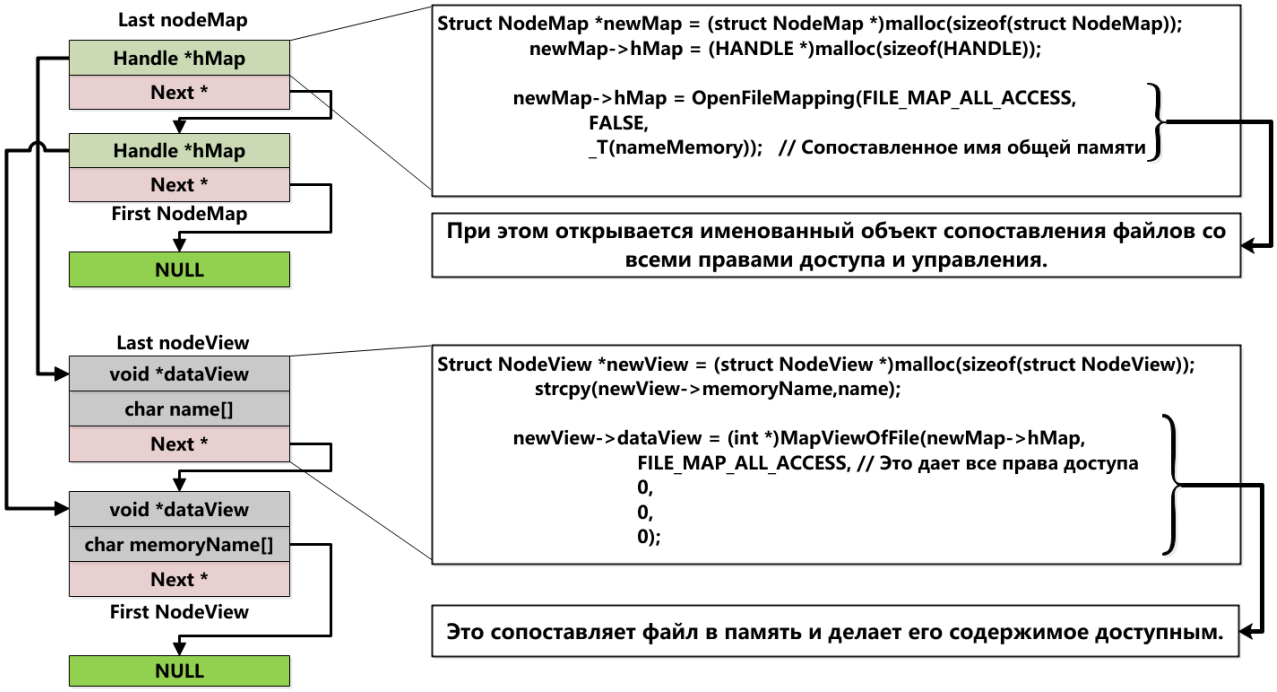

Рисунок 3.4 – Процесс отображения и открытия именованной общей памяти

3.2.3 Чтение и запись в именованные общие памяти

Как только получение доступа к общим памятям, с которыми требуется работать, нам понадобятся функции для чтения и записи значений. Логика, предложенная для навигации между общими памятями, одинакова для всех типов данных. Чтобы найти именованную общую память, с которой требуется взаимодействовать, нужно перейти по списку именованных общих памятей, используем временный узел типа представления, который начинается с указания на конец списка, а затем просматриваем весь список до тех пор, пока не найдем именованную общую память, к которой требуется получить доступ, после получения доступа к памяти, с которой требуется взаимодействовать, можно начать чтение или запись данных.

На рис. 3.5 показан процесс создания временного узла типа представления, который начинает перемещаться по списку существующих именованных общих памятей, пока не найдет общую память, с которой требуется взаимодействовать.

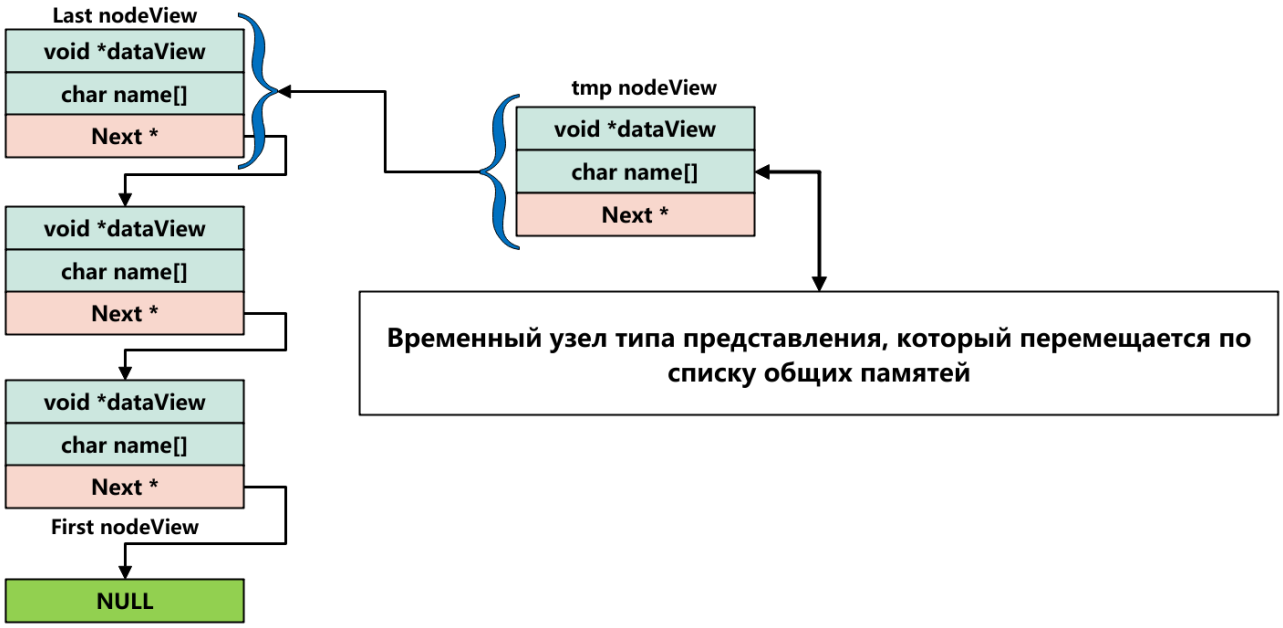

Рисунок 3.5 – Процесс навигации по списку общих воспоминаний

Функции чтения и записи создаются для всех типов данных, необходимых в этом исследовании. Процесс прокрутки списка общих памятей

(показан выше) используется всеми функциями чтения и записи, поскольку, как только желаемая общая память найдена, желаемое значение считывается или записывается. Функции, доступные как для чтения, так и для записи, отображаются в листинге 6.

Листинг 6 – Список функций чтения и записи

```
18 int get_Int(char memoryName, int valuePosition);
19 void set_Int(char memoryName, int valuePosition,int dValue);
20 float get_Float(char memoryName, int valuePosition);
21 void set_Float(char memoryName, int valuePosition, float fValue);
22 double get_Double(char memoryName, int valuePosition);
23 void set_Double(char memoryName, int valuePosition, double lfValue);
24 *void set_String(char memoryName, int valuePosition, char strValue);
25 *void get_String(char memoryName, int valuePosition, char strRetValue);
26 *void get_StringMatlab(char *memoryName, int valuePosition, char strRetValue);
```

Примечание: Специальная функция создается для чтения значений строкового типа из Matlab, так как обработка внутренней памяти в этом приложении отличается.

4 Разработка и использование библиотек в Windows

Для выполнения работы требуется обработка типов данных Integer, Float, Double и String. Для достижения поставленной цели проводится всестороннее исследование по разработке и использованию динамических библиотек ссылок в операционных системах Windows.

Есть важные моменты, которые следует учитывать при разработке библиотеки, например, необходимо выбрать между статической библиотекой и библиотекой динамических ссылок, это важно, поскольку существуют приложения, которые поддерживают только библиотеки динамических ссылок, такие как Matlab.

Для достижения большей совместимости это предложение создает библиотеки динамических ссылок. Если требуются библиотеки статических ти-

пов, необходимо только изменить настройки компилятора для создания библиотеки статических ссылок, код не требует каких-либо изменений.

4.1 Вызывающий абонент (Caller), Вызываемый абонент (Callee) и Соглашения о вызовах

Когда речь идет о взаимодействии между приложениями и библиотеками, появляются два автора, которые идентифицируют взаимодействие, являются вызывающим (caller) и вызываемым (callee), это может происходить при взаимодействии нескольких файлов, например, между исполняемым приложением и библиотекой; но это также может быть дано в одном и том же приложении или библиотеке, в частности, при взаимодействии между функциями. Пример взаимодействия между этими двумя субъектами показан на рис. 6.

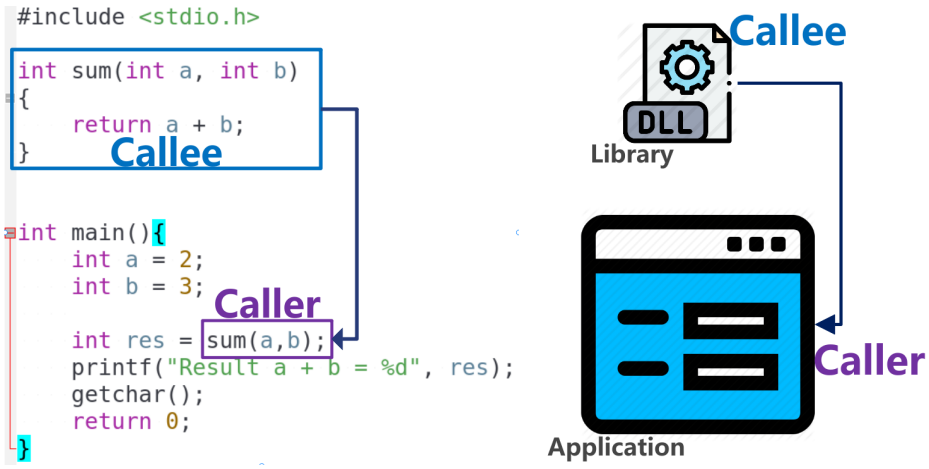

Рисунок 4.1 – Пример взаимодействия вызывающего абонента и вызываемого абонента. a) взаимодействие между функциями в рамках одного кода; b) взаимодействие между приложением и библиотекой

Соглашение о вызове – это низкоуровневая реализация, которая определяет взаимодействие между вызывающим (caller) и вызываемым (Callee) при работе с функциями, в частности определяет порядок и в каких регистрах процессора будут храниться параметры и значения ответа функции, а также соглашение о вызове указывает, кто отвечает за очистку стека, проблема

возникает только в 32-разрядных системах Windows, так как в 64-разрядной архитектуре Windows это было исправлено с помощью единого соглашения о вызовах, называемого двоичным интерфейсом приложения (ABI), что обеспечивает максимальную совместимость [30].

В 32-разрядных операционных системах Windows существует большая проблема совместимости на уровне соглашений о вызовах. В начале каждый компилятор предлагал свой способ обработки параметров и возвращаемых функций с различными процедурами обработки реестров и очистки стека. Было предложено несколько предложений по соглашениям о вызовах для 32-разрядных систем Windows [31].

Основными отличительными характеристиками различных соглашений о вызовах являются способ получения параметров и их возвращаемых значений. В таблице 4.1 приведены сведения об основном 32-разрядном соглашении о вызовах Windows.

Таблица 4.1 – Соглашения о передаче аргументов и именовании.
Корпорация Майкрософт

Ключевое слово	Очистка стека	Передача параметров
__cdecl	Caller	Помещает параметры в стек в обратном порядке (справа налево).
__clrcall	n/a	Загрузите параметры в стек выражений CLR по порядку (слева направо.
__stdcall	Callee	Помещает параметры в стек в обратном порядке (справа налево).
__fastcall	Callee	Хранится в регистрах, а затем помещается в стек.
__thiscall	Callee	Нажата на стек; этот указатель хранится в ECX.
__vectorcal	Callee	Хранится в регистрах, затем помещается в стек в обратном порядке (справа налево).

Примечание: Проект выполняется с использованием языка программирования C, и используется соглашение о вызовах __cdecl, __stdcall и __fastcall.

4.2 Создание динамических библиотек ссылок (обычные)

Чтобы указать использование соглашения о вызове, каждая функция должна начинаться с модификатора __declspec и должна принимать dllexport в качестве аргумента.

В исходном файле будут реализованы различные функции. Файл заголовка будет содержать объявления функций, включая различные соглашения о вызовах, которые будут раскомментированы один за другим для создания различных библиотек в соответствии с соглашением о вызовах.

В этом предложении было создано только файл общей библиотеки ссылок, не библиотеку импорта с расширением .lib, которая предлагает такие функции, как указание имен экспорта функций.

На рис. 4.2 показаны некоторые функции управления общей памятью. Показана реализация файла заголовка и файла исходного кода.

На рисунке 3.1 было объяснено, что функции реализованы в разных файлах, чтобы упростить их использование в нескольких проектах, только импортируя файлы. В этом случае импортируется файл заголовка под названием “функция.h”, и требуется только добавление вызовов каждой функции в структуру, необходимую для создания библиотеки динамических ссылок.

4.3 Создание динамических библиотек ссылок (Java)

Библиотеки, созданные на предыдущем шаге, могут использоваться из многих языков программирования и приложений, но не будут доступны из проектов, созданных на языке программирования java. Для проектов на языке программирования Java необходимо создать другую библиотеку, механизм, который позволяет нам взаимодействовать с библиотеками, написанными на языке C/C++ из Java, называется Java Native Interface (JNI).

Следует помнить, что Java имеет зависимость имени между исходным файлом и его внутренним классом, поэтому необходимо было бы создать биб-

```h
#ifndef SMCLIENT_H
#define SMCLIENT_H
#define EXPORT __declspec(dllexport)

// #define CONVENTION __fastcall
// #define CONVENTION __cdecl
#define CONVENTION __stdcall

EXPORT int CONVENTION  openMemory
  (char *memoryName, int typeCode);

EXPORT int CONVENTION  getInt
  (char *memoryName, int valuePosition);

EXPORT void CONVENTION  setInt
  (char *memoryName, int valuePosition,int dValue);

EXPORT void CONVENTION  setFloat
  (char *memoryName, int valuePosition, float fValue);

EXPORT float CONVENTION  getFloat
  (char *memoryName, int valuePosition);

EXPORT void CONVENTION  setDouble
  (char *memoryName, int valuePosition, double lfValue);

EXPORT double CONVENTION  getDouble
  (char *memoryName, int valuePosition);

EXPORT void CONVENTION  setString
  (char *memoryName, int valuePosition, char *strValue);

EXPORT void CONVENTION  getString
  (char *memoryName, int valuePosition, char *strRetValue);

EXPORT void CONVENTION  getStringMatlab
  (char *memoryName, int valuePosition, char **strRetValue);


EXPORT int CONVENTION  createMemory
  (char *memoryName, int quantity, int typeCode);

EXPORT void CONVENTION  freeMemories();

EXPORT void CONVENTION  freeViews();

#endif
```

(a)

```c
#include "functions.h"
#include "smClient.h"
#include <stdio.h>
#include <string.h>
#include <windows.h>
#include <tchar.h>

EXPORT int CONVENTION  openMemory
  (char *memoryName, int typeCode){
    int errorCode = open_Memory(memoryName, typeCode);
    return errorCode;
}

EXPORT int CONVENTION  getInt
  (char *memoryName, int valuePosition){
    int dRetValue = get_Int(memoryName, valuePosition);
    return dRetValue;
}

EXPORT void CONVENTION  setInt
  (char *memoryName, int valuePosition,int dValue){
    set_Int(memoryName,valuePosition,dValue);
}

EXPORT void CONVENTION  setFloat
  (char *memoryName, int valuePosition, float fValue){
    set_Float(memoryName,valuePosition,fValue);

}

EXPORT float CONVENTION  getFloat
  (char *memoryName, int valuePosition){
    float fRetValue = get_Float(memoryName,valuePosition);
    return fRetValue;
}

EXPORT void CONVENTION  setDouble
  (char *memoryName, int valuePosition, double lfValue){
    set_Double(memoryName,valuePosition,lfValue);

}

EXPORT double CONVENTION  getDouble
  (char *memoryName, int valuePosition){
    double lfRetValue = get_Double(memoryName,valuePosition);
    return lfRetValue;
}
```

(b)

Рисунок 4.2 – Код для создания библиотеки динамических ссылок: a) файл заголовка .h; b) исходный код .c

лиотеку для каждого необходимого проекта Java, в качестве решения этой проблемы предлагается создать файл как часть пакета, который имеет реализацию промежуточных функций, взаимодействующих с объявленными функциями JNI, так как таким образом библиотека может использоваться в любом проекте java только путем создания пакета и файла с тем же именем, что и исходный файл, используемый для создания файла заголовка.

Различные фрагменты кода показаны на рис. 4.3. На рисунке (a) показан исходный код файла с именем SmClient.java, этот файл содержит имена и структуру функций, которые будут созданы в библиотеке. На рисунке (b) показан файл eros_SmClient.h, автоматически сгенерированный из SmClient.java файл с помощью команды javac. На рисунке (c) показан код

исходного файла, реализующий функции файла заголовка.

```java
package eros;

public class SmClient {
        public native int openMemory(String nombre, int type);
        public native int getInt(String memName, int position);
        public native void setInt(String memName, int position, int value);
        public native void setFloat(String memName, int position, float value);
        public native float getFloat(String memName, int position);
        public native void setDouble(String memName, int position, double value);
        public native double getDouble(String memName, int position);
        public native void setString(String memName, int position, String word);
        public native String getString(String memName, int position);
        public native int createMemory(String memName, int quantity, int type);
        public native void freeMemories();
        public native void freeViews();
}
```

(a)

```c
/* DO NOT EDIT THIS FILE - it is machine generated */
#include <jni.h>
/* Header for class eros_SmClient */

#ifndef _Included_eros_SmClient
#define _Included_eros_SmClient
#ifdef __cplusplus
extern "C" {
#endif

JNIEXPORT jint JNICALL Java_eros_SmClient_openMemory
  (JNIEnv *, jobject, jstring, jint);

JNIEXPORT jint JNICALL Java_eros_SmClient_getInt
  (JNIEnv *, jobject, jstring, jint);

JNIEXPORT void JNICALL Java_eros_SmClient_setInt
  (JNIEnv *, jobject, jstring, jint, jint);

JNIEXPORT void JNICALL Java_eros_SmClient_setFloat
  (JNIEnv *, jobject, jstring, jint, jfloat);

JNIEXPORT jfloat JNICALL Java_eros_SmClient_getFloat
  (JNIEnv *, jobject, jstring, jint);

#ifdef __cplusplus
}
#endif
#endif
```

(b)

```c
#include "eros_SmClient.hpp"
#include <ctype.h>
#include <string.h>
#include "../smClient/functions.c"

JNIEXPORT jint JNICALL Java_eros_SmClient_openMemory
  (JNIEnv* env, jobject obj, jstring memoryName, jint typeCode){
        const char* str = env->GetStringUTFChars(memoryName, 0);
      char memory_name[32];
      strcpy(memory_name, str);
      env->ReleaseStringUTFChars(memoryName, str);

          return open_Memory(memory_name, typeCode);

  }

JNIEXPORT jint JNICALL Java_eros_SmClient_getInt
  (JNIEnv* env, jobject obj, jstring memoryName, jint valuePosition){
        const char* str = env->GetStringUTFChars(memoryName, 0);
      char memory_name[32];
      strcpy(memory_name, str);
      env->ReleaseStringUTFChars(memoryName, str);

          return get_Int(memory_name, valuePosition);
  }

JNIEXPORT void JNICALL Java_eros_SmClient_setInt
  (JNIEnv* env, jobject obj, jstring memoryName, jint valuePosition, jint value){
        const char* str = env->GetStringUTFChars(memoryName, 0);
      char memory_name[32];
      strcpy(memory_name, str);
      env->ReleaseStringUTFChars(memoryName, str);

          set_Int(memory_name, valuePosition, value);
  }

JNIEXPORT void JNICALL Java_eros_SmClient_setFloat
  (JNIEnv* env, jobject obj, jstring memoryName, jint valuePosition, jfloat value){
        const char* str = env->GetStringUTFChars(memoryName, 0);
      char memory_name[32];
      strcpy(memory_name, str);
      env->ReleaseStringUTFChars(memoryName, str);

          set_Float(memory_name, valuePosition, value);
  }
```

(c)

Рисунок 4.3 – Программный код для использования в проектах Java: a) Базовая структура в Java - SmClient.java; b) Сгенерированный файл - eros_-SmClient.h; c) Реализация функций - smClient.cpp

После создания библиотеки файл может быть переименован, требуется только указать имя библиотеки в параметре функции System.LoadLibrary(“ имя библиотеки ”);.

5 Примеры использования и экспериментальные результаты

Библиотеки – это контейнеры функций, которые используются другими приложениями, поэтому единственный способ показать полезность предложения это сценарии, в которых оно использовалось.

Ниже описаны различные исследовательские работы, в которых предлагаемая библиотека использовалась в качестве средства взаимосвязи в режиме реального времени.

5.1 Unity3D-MatLab Симулятор в режиме реального времени для робототехнических приложений

В данной исследовании была проведена реализация 3D-симулятора, применяемого в области робототехники. Тренажер позволяет анализировать работу различных математических моделей автономного и/или телеуправляемого управления в структурированных, частично структурированных и неструктурированных средах. Виртуальный сценарий создается в Unity3D. Unity3D – это программное обеспечение обменивается информацией с Matlab, где выполняются алгоритмы управления, обмен данными между этими двумя приложениями осуществляется за счет использования общей памяти. Обмен информацией в режиме реального времени между двумя программами имеет важное значение, поскольку при расширенном управлении алгоритмы требуют обратной связи от взаимодействия робота с окружающей средой для замыкания схемы управления, эти значения состояния робота анализируются в Matlab, а затем передаются кинематические и динамические параметры в зависимости от переменных управляемости, рассчитанных Matlab. Наконец, 3D-симулятор оценивается путем реализации автономной схемы управления для решения задачи отслеживания траектории руки робота 6DOF [32].

Использование библиотеки, предложенной в работе, было сосредоточено на обмене информацией между Matlab и Unity3D рис. 5.1.

Общие памяти создаются в двух группах, одна группа общих памятей используется для Matlab для записи значений, считываемых Unity3D, вторая группа общих памятей используется для Unity3D для хранения значений, которые будут считываться Matlab.

Обмен информацией также позволяет просматривать в Unity3D состо-

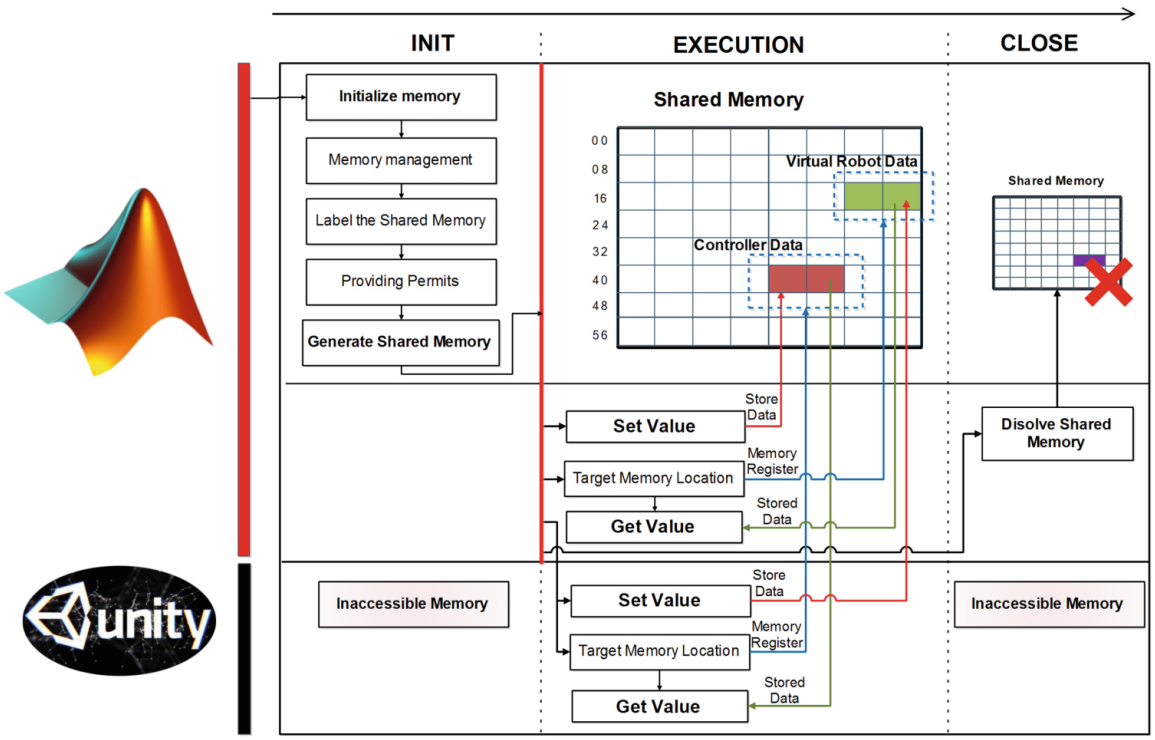

Рисунок 5.1 – Межпроцессная связь через общую память

яние мобильного манипулятора в режиме реального времени, рис. 5.2.

Этот тип связи в реальном времени позволял пользователю-оператору наблюдать реальное состояние мобильного манипулятора в дополненной реальности, рис. 5.3.

Удалось разработать оптимальный сценарий проведения испытаний предложенных математических моделей, используя высокоскоростную связь, это помогло в исследовании, так как математические модели являются обратной связью в режиме реального времени для принятия решений.

Результат этого проекта показывает ошибки, близкие к 0 метрам в его траектории, рис. 5.4.

5.2 Координированное управление роботами для выполнения служебных задач в средах виртуальной реальности

После проверки правильности работы предложения создаются более сложные сценарии для продолжения испытаний его работы.

Исследовательский проект включал создание виртуальной среды, в

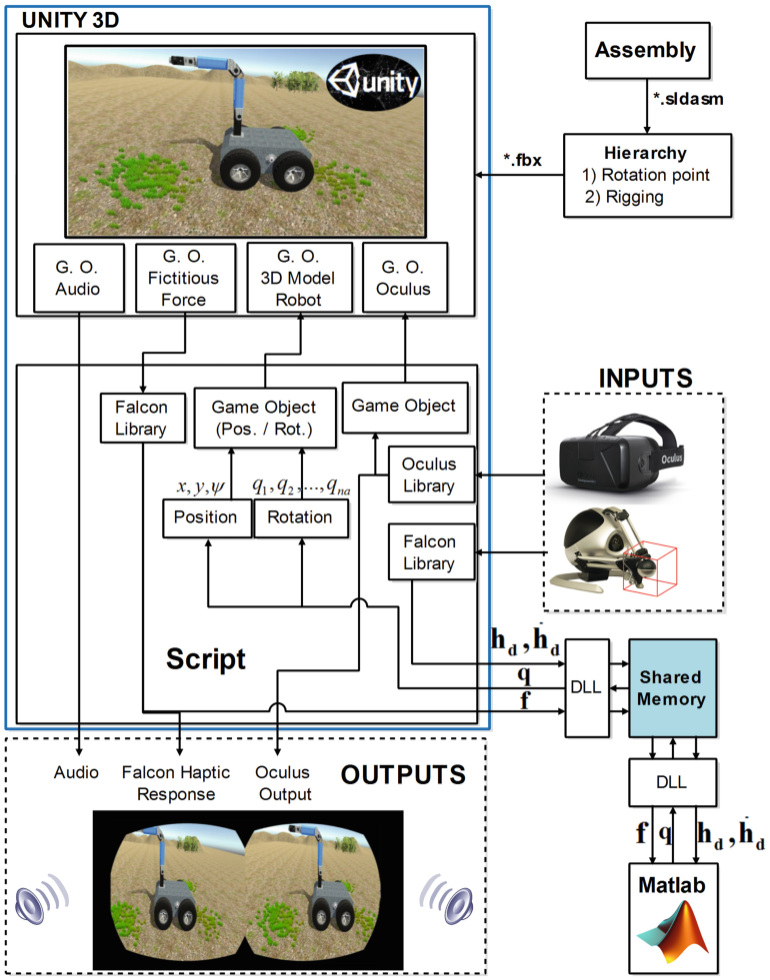

Рисунок 5.2 – Взаимодействие Unity3D – Matlab

Рисунок 5.3 – Локальный сайт схемы телеоперации

которой можно было бы протестировать основные контроллеры робота. Для проверки каждого из элементов управления предлагается переключатель [33]. Рассматриваются три контроллера:

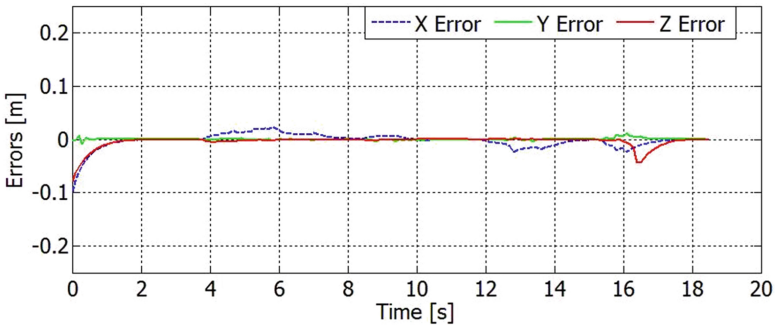

Рисунок 5.4 – Результат анализа ошибок на траектории, выполняемой мобильным манипулятором

– Точка-точка управление, которое переносит робота из начальной точки в другую точку с определенной ориентацией.

– Управление траекторией слежения, которая имеет положение и ориентацию в определенное время.

– Отслеживание траектории, не параметризованной во времени.

Общие памяти создаются в двух группах, одна группа общих памятей используется для Matlab для записи значений, считываемых Unity3D, вторая группа общих памятей используется для Unity3D для хранения значений, которые будут считываться Matlab, рис. 5.5.

Это исследование также измерило уровень точности отслеживания рабочего конца робота по ранее определенному пути.

Для этого было загружено изображение, из которого извлекаются его края, затем на краю изображения запускается математическая модель для следования по маршруту. Этот процесс показан на рис. 5.6.

Все результаты должны быть проанализированы в зависимости от желаемого пути в отличие от фактического пути, который выполнил манипулятор. Анализ диапазона ошибок показан на рис. 5.7.

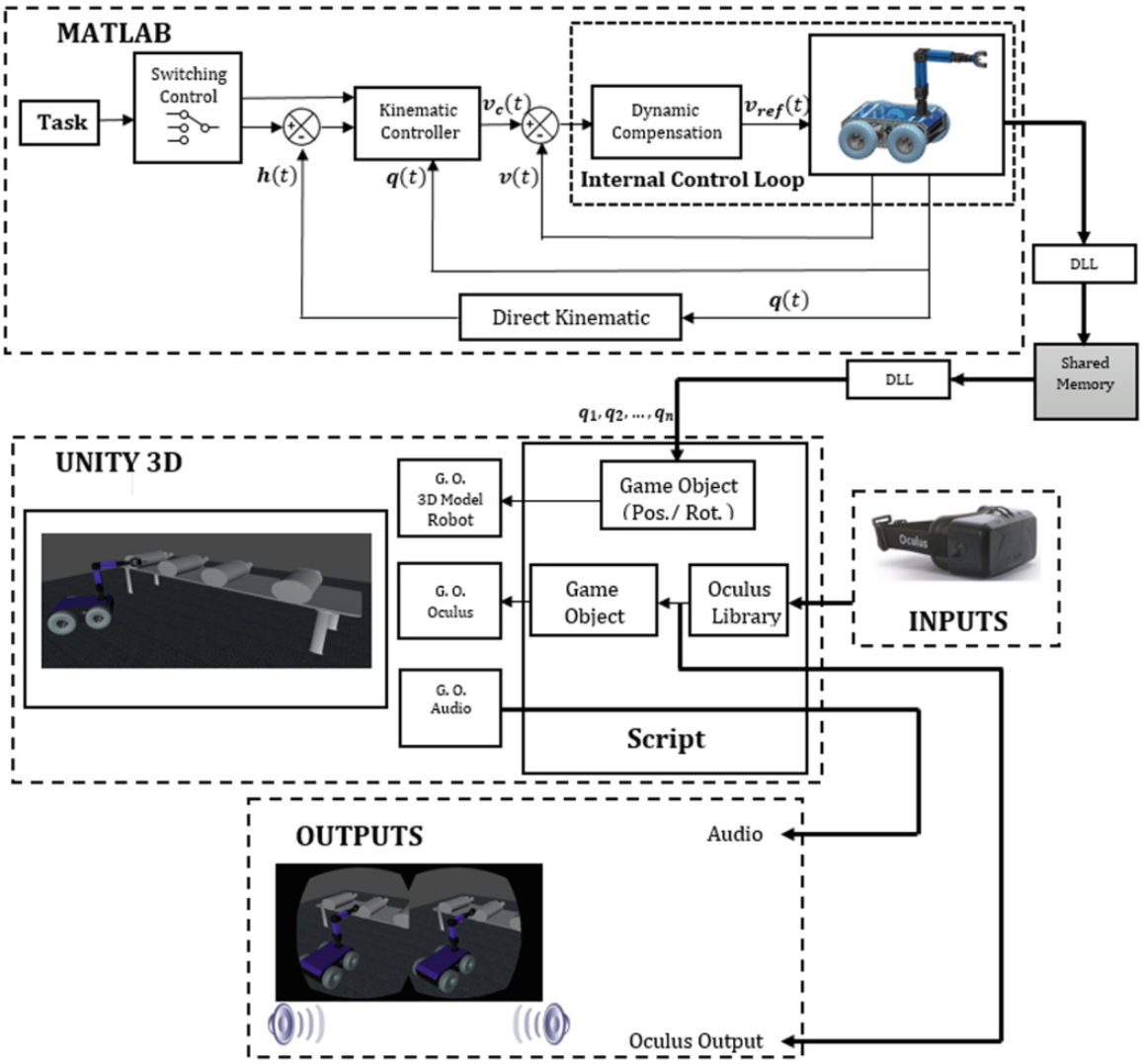

Рисунок 5.5 – Функциональная схема робота

5.3 Визуализатор виртуальной реальности в реальном времени для беспилотных летательных аппаратов

Проводились прикладные научно - исследовательские работы в области управления UAV. Было предложено приложение, основанное на виртуальной реальности, для воссоздания в режиме реального времени задач выполнения UAV. Беспилотник управлялся дистанционно или автономно в реальных условиях. Реконструкция реальной среды, в которой проводятся полевые испытания, была рассмотрена для достижения визуализации всех выполненных задач в режиме реального времени. Информация о состоянии воздушного судна была доступна через общие памяти [34].

Представленные результаты подтверждают:

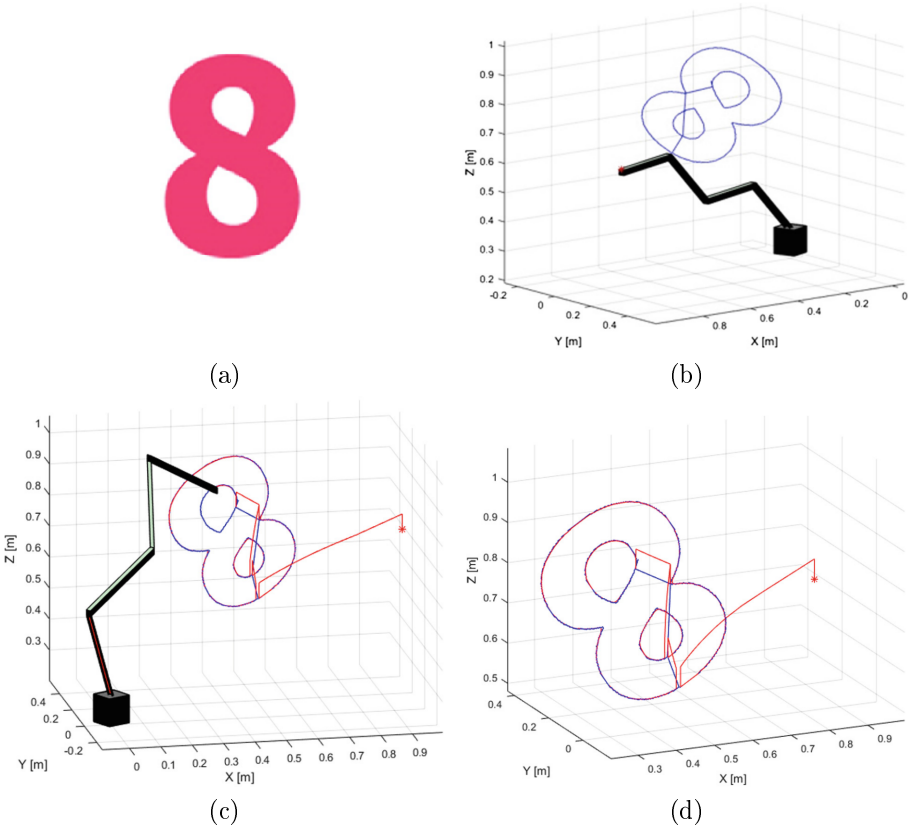

Рисунок 5.6 – Цифровая обработка изображения, выбранного для задачи: a) исходное изображение; b) Векторизованное изображение в пространстве руки робота; c) Задача выполнена на 80%; d) Задача выполнена

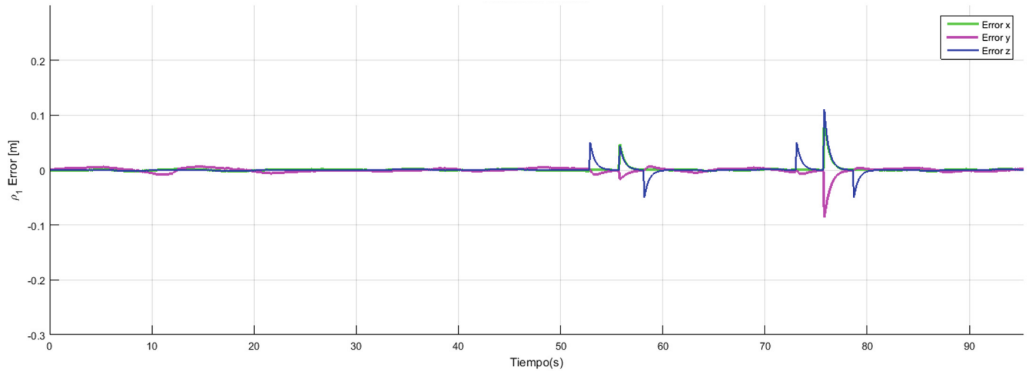

Рисунок 5.7 – Результат диапазона ошибок при выполнении задачи

– Унификация 3D - моделей и реконструкция окружающей среды.

– Потребление данных о состоянии с беспилотного летательного аппарата.

– Правильная работа математических моделей, которые применяются к движению UAV.

В рамках модели взаимодействия между UAVs, математической моделью телеоперации и визуализацией в виртуальной реальности. Общая память использовалась для доступа в режиме реального времени ко всей информации о состоянии дрона, рис. 5.8.

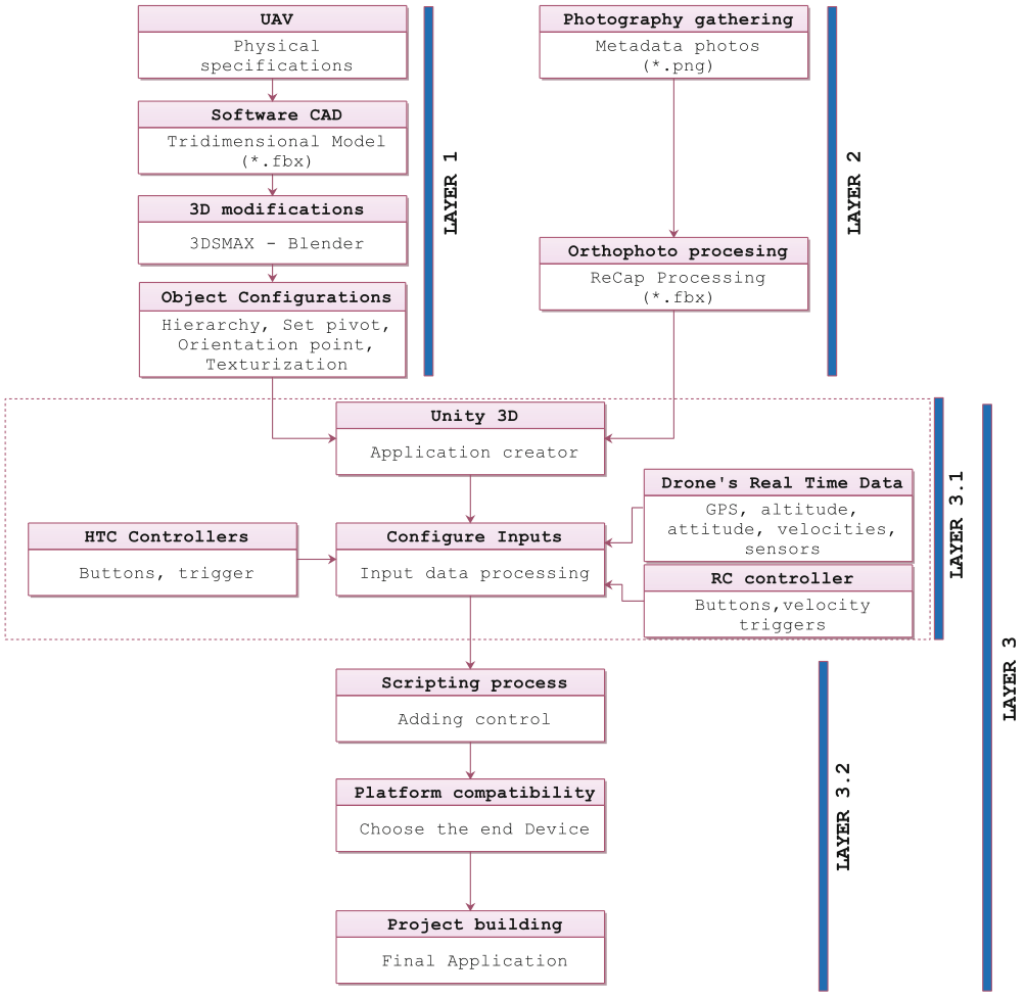

Рисунок 5.8 – Разработка интерфейса виртуальной реальности

Для достижения визуализации реальной среды в виртуальной среде была создана 3D-модель местности, по которой должен был двигаться летательный аппарат, рис. 5.9.

В результате смещения UAV удалось получить точность с погрешностью менее 1,5 метров от желаемого маршрута. Следует учитывать, что ветровые условия при проведении испытаний составляли около 12,2 километра в час. Данные о состоянии UAV показаны на рисунке 5.10.

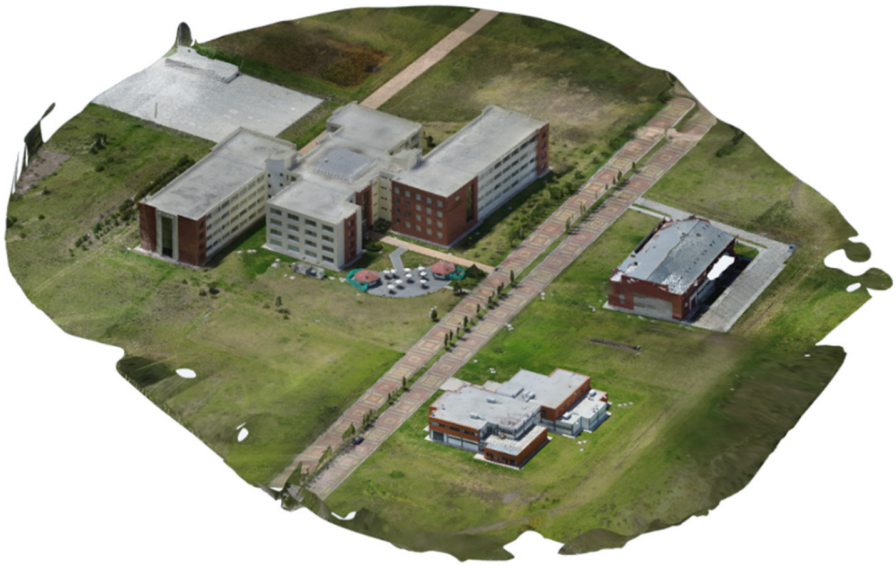

Рисунок 5.9 – Реконструкция реальных сред

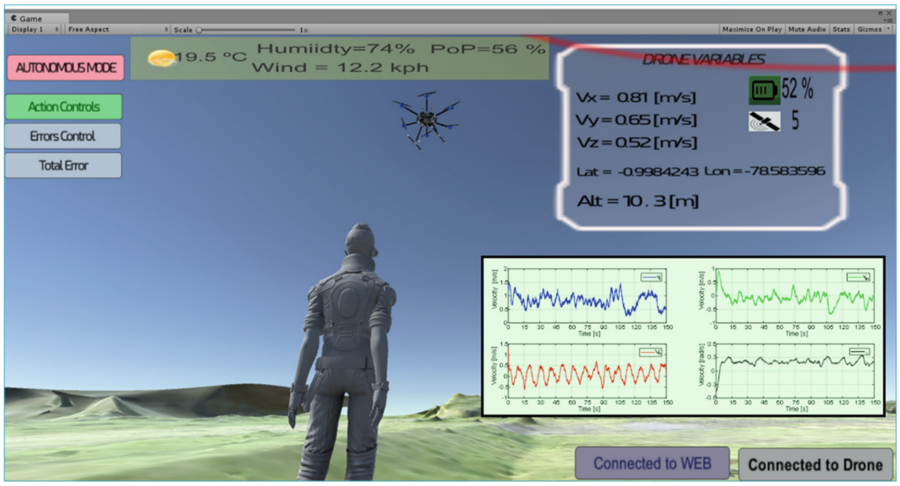

Рисунок 5.10 – Автономное управляемое исполнение

5.4 Изучение Базового Языка знаков с помощью Дидактических игр

Предложение об использовании общих памятей для обмена информацией было также использовано при разработке дидактической игры для обучения-изучения основного языка эквадорских знаков. Эта работа ориентирована на людей с нарушениями слуха и/или людей, заинтересованных в изучении базового языка жестов Эквадора. Устройство ввода представляет собой “скачкообразное движение - Leap Motion”, которое обнаруживает сигналы жестов в программном обеспечении Unity3D. В MatLab реализована

система классификации для обработки сигналов для двух типов конфигурации сигналов: статической и динамической; корреляция используется, если конфигурация статична (сигнал движения отсутствует), а алгоритм DTW используется для динамических конфигураций (сигнал движения). Связь между Unity3D и MatLab осуществляется через общую память. Эта разработка приобретает дидактическое применение с двумя играми, реализованными в Unity3D. В игре есть как однопользовательская, так и многопользовательская опции [35]. Схема связи между Unity3D и Matlab показана на рисунке 5.11.

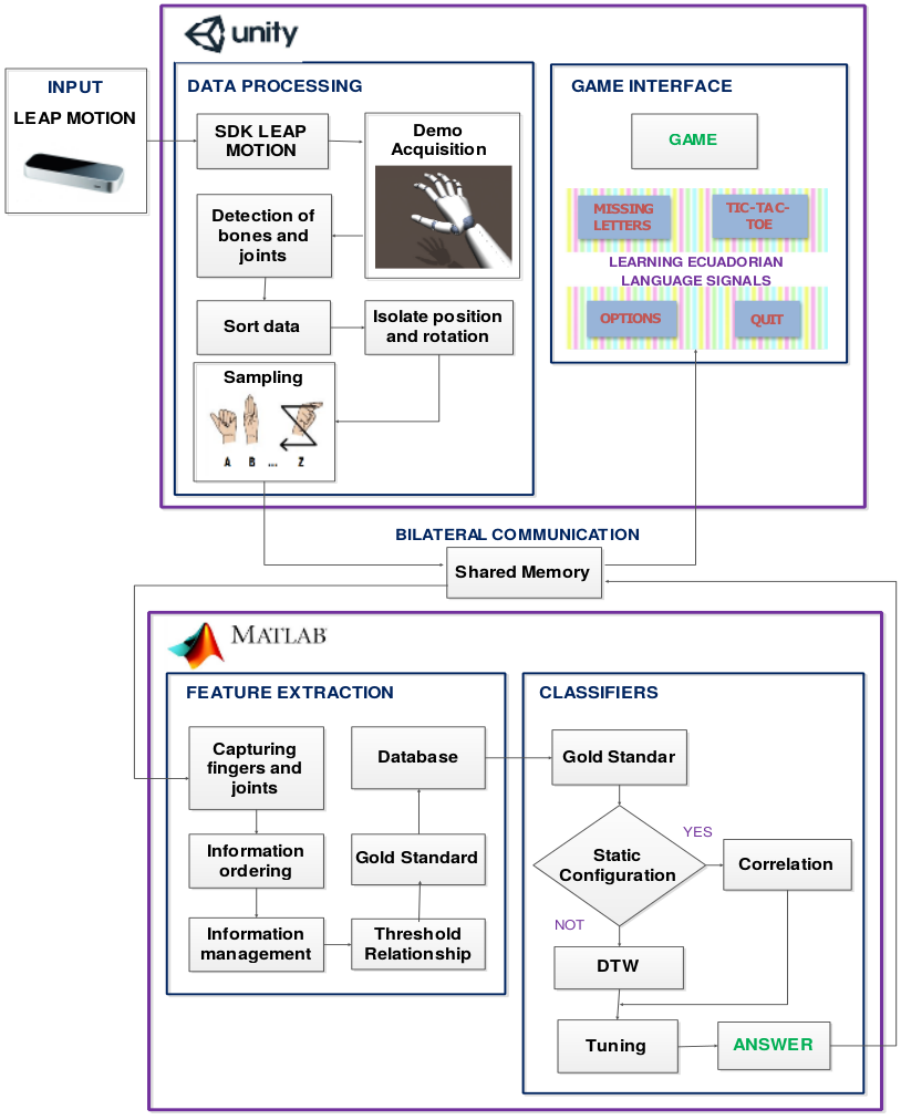

Рисунок 5.11 – Схема работы системы

Получение данных от прыжкового движения соответствует монито-

рингу движений рук пользователя, как показано на рисунках 5.12 и 5.13.

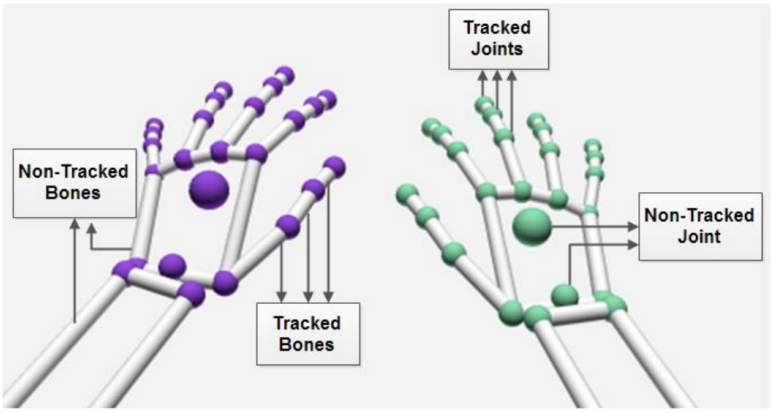

Рисунок 5.12 – Отслеживание рук с помощью LeapMotion

Рисунок 5.13 – Отслеживание рук

6 Социальная ответственность

ВВЕДЕНИЕ

Данная исследовательская работа направлена на создание библиотеки динамических ссылок, содержащей функции создания, управления, открытия, закрытия и выпуска общих памяти, которые позволят обмениваться информацией между процессами и/или приложениями. Динамическая библиотека ссылок использует общую память в качестве межпроцессной связи (IPC Interprocess Comunication) на высокой скорости.

Сфера применения разработки - это прикладные работы или исследовательские работы, требующие представления данных в режиме реального времени для использования другими процессами и / или приложениями. Следует подчеркнуть, что при хранении только одной данной для каждого управляемого пространства в каждой общей памяти не предлагаются функцию список очерёдности данных , поэтому доступны только самые текущие данные.

Разработка этого проекта велась с использованием персонального компьютера на нескольких рабочих площадках, расположенных Томским политехническим университетом, которые обладают характеристиками офисного типа.

Поскольку данная работа по разработке программного обеспечения напрямую связана с использованием компьютерного оборудования, необходимо обеспечить соблюдение трудового законодательства, а также законов, защищающих окружающую среду от возможных вредных последствий, доказанных использованием компьютера.

6.1 Правовые и организационные вопросы обеспечения безопасности

К теме трудового права, основным источником информации является трудовой кодекс Российской Федерации. Данный официальный документ даёт регулирования в сфере, связанной с заработной платы, рутинного труда, особенности регулирования труда женщин, регулирования труда детей, регулирования труда лиц с ограниченными возможностями и другие. Трудовой кодекс, помимо основных положений отношений между работником и работодателем, содержит руководящие принципы по вопросу безопасности труда, которые также должны применяться в процессе реализации магистерских диссертаций. Следует подчеркнуть, что российское законодательство запрещает дискриминацию по любым признакам и принудительный труд.

На вопрос о правилах, которые контролируют режим работы, в пределах Российской Федерации, режима широко распространённый в считанные делопроизводство-это: Рабочая неделя состоит из 5 дней работы и двух дней, что соответствует конце недели, в течение рабочего времени в Рабочий диапазон широко распространённый берет начало в 9:00 и имеет своё завершение в 17:00, следует уточнить, что в это время работы есть отдыха, что соответствует с 13:00 до 14:00. Кроме того, в соответствии с правилами нормальное рабочее время не может превышать 40 часов в неделю, хотя дети в возрасте до 16 лет могут работать, рабочее время в течение рабочей недели для детей в возрасте до 16 лет не должно превышать 24 часа, для молодых людей в возрасте от 16 до 18 лет число рабочих часов в течение рабочей недели не может превышать 35 часов, эти правила применяются также к инвалидам I и II группы [36].

При анализе должности, которую занимает работодатель, правила Российской Федерации обязывают его обеспечить надлежащие условия, чтобы его сотрудники могли безопасно выполнять свою работу. Информация, изло-

женная в нормативных актах Российской Федерации, содержит подробную информацию о характеристиках рабочего места, исправности используемых машин, исправности технологического оборудования, своевременном предоставлении необходимой технической документации, надлежащем качестве используемых материалов, требуемом качестве используемых инструментов и т. д. Ко всему вышеописанному следует также добавить условия, необходимые в области здоровья и безопасности на производстве.

Рассматривая тему выполнения диссертационной работы как трудовой деятельности, следует считать, что это вид работы, предполагающий использование компьютера или также называемый персональные электронно-вычислительные машины (ПЭВМ), помимо персонального компьютера основными элементами рабочего места программиста являются: письменный стол, рабочее кресло, дополнительный дисплей, дополнительная клавиатура, дополнительная мышь. Место, в котором развивается эту работу, необходимо обеспечить правильные условия для выполнения поставленных задач, рабочие операции должны выполняться в зоне действия поля двигателя, выполнение рабочих операций необходимо убедиться в зоне легкого доступа, как это указывают нормы [37, 38] также должна быть обеспечена оптимальная осанка человека, правильное расположение и порядок рабочего места, свобода трудового передвижения, использование оборудования, отвечающего требованиям эргономики и инженерной психологии, все это обеспечивает более эффективный трудовой процесс, снижает усталость и предотвращает риск профессиональных заболеваний.

В Российской Федерации основным документом, регулирующим любую работу с компьютерами (или ПЭВМ), является ТОИ Р-45-084-01 Типовая инструкция по охране труда при работе на персональном компьютере, этот документ содержит информацию, регулирующую условия и организацию, а также руководящие принципы санитарно-эпидемиологического регулирования, в дополнение к этому официальному документу использу-

ются также правила, регулирующие рабочее место пользователя, такие как: ПЭВМ: ГОСТ 12.2.032-78 ССБТ «Рабочее место при выполнении работ сидя. Общие эргономические требования» и ГОСТ Р 50923-96. «Дисплеи. Рабочее место оператора. Общие эргономические требования и требования к производственной среде. Методы измерения» [38–40].

Помещения, используемые в качестве рабочего пространства в Томском политехническом университете, обеспечивают комфортную и безопасную среду для работы с компьютером, конструкции рабочих мест учитывают требования расстояния до настольных компьютеров, но при использовании персонального компьютера пользователь обязан уважать расстояние от глаз до монитора портативного компьютера, который должен быть 650 мм. При работе с персональным компьютером это позволяет поворачивать экран в горизонтальной и вертикальной плоскости с фиксацией в указанном положении для обеспечения переднего наблюдения монитора. В заключение, при выполнении этой работы были приняты во внимание рекомендуемые меры по обеспечению правильного размещения персонального компьютера на поверхности рабочего стола, помимо расположения компьютера, была также учтена правильная калибровка яркости и контрастности монитора для повышения комфорта при работе с персональным компьютером. К вышеизложенному следует добавить, что, используя описанную информацию о том, что рекомендуемое время отдыха должно составлять 10-15 минут после каждых 45-60 минут работы, было решено делать 10-минутный отдых каждые 50 минут работы, в течение этого времени отдыха выполняются упражнения для глаз и минуты физической подготовки. Сидячее положение - это положение, выбранное в работе с компьютерами, в сидячем положении основная нагрузка ложится на мышцы, поддерживающие позвоночник и голову. В связи с этим, когда человек сидит в течение длительного времени, необходимо время от времени менять фиксированные рабочие позы.

6.2 Производственная безопасность

Чтобы обеспечить безопасность студента во время разработки магистерской диссертации, выполняя свою работу перед персональным компьютером, необходимо проанализировать возможные вредные и опасные воздействия, которые могут возникнуть в зависимости от среды, в которой студент работает. Когда воздействие человека может привести к заболеванию, оно считается вредным фактором, вместо этого, когда воздействие человека может привести к травме, оно считается опасным фактором.

Для проведения анализа вредных и опасных факторов, которые могут повлиять на развитие этой работы в качестве источника берётся ГОСТ 12.0.003-2015 «Опасные и вредные производственные факторы. Классификация» [41]. Анализируя тип выполняемой работы, в данном случае ориентированной на разработку программного обеспечения, целесообразно учитывать вредные и опасные физические и психофизические факторы производства, этот тип анализа идеально подходит для работ, связанных с использованием компьютеров, ниже рассматриваются физические факторы . Список факторов показан на таблице 6.1.

Температура и относительная влажность воздуха

Проводя анализ микроклимата, особенно в связи с его температурой и относительной влажностью, известно, что длительное воздействие на человека плохих условий на уровне микроклимата ухудшает самочувствие работника, снижает его производительность и может привести к заболеваниям, поэтому на рабочем месте следует уделять большое внимание контролю микроклимата.

Оптимальные микроклиматические условия устанавливаются в соответствии с критериями оптимального теплового и функционального состояния человека. Допустимые значения показателей микроклимата устанавливаются в тех случаях, когда по технологическим, техническим и экономическим

Таблица 6.1 – Возможные и вредные факторы

Факторы (ГОСТ 12.0.003-2015)	Этапы работ			Нормативные документы
	Разработка	Изготовление	Эксплуатация	
1.Повышенная или пониженная температура и относительная влажность воздуха.	+	+	+	СанПиН 2.2.4.548-96 Гигиенические требования к микроклимату производственных помещений
2.Превышение уровня шума	+	+	+	ГОСТ 12.1.003-2014 ССБТ. Шум. Общие требования безопасности.
3. Отсутствие или недостаток освещения.	+			СП 52.13330.2016 Естественное и искусственное освещение. Актуализированная редакция СНиП 23-05-95* (с Изменением N 1)
4. Психофизиологические факторы (монотонность труда, умственное и эмоциональное напряжение, перенапряжение зрительных анализаторов)	+	+	+	Об утверждении санитарных правил и норм СанПиН 1.2.3685-21 “Гигиенические нормативы и требования к обеспечению безопасности и (или) безвредности для человека факторов среды обитания”
5. Поражение электрическим током	+		+	ГОСТ 12.1.038-82 Система стандартов безопасности труда (ССБТ). Электробезопасность. Предельно допустимые значения напряжений прикосновения и токов (с Изменением N 1)
6. 6. Короткое замыкание	+	+	+	ГОСТ Р 50571.4.44-2019. Электроустановки Низковольтные. Часть 4.44. Защита для обеспечения безопасности. Защита от резких отклонений напряжения и электромагнитных возмущений
7. Статическое электричество	+	+	+	ГОСТ 12.1.044-89. Система стандартов безопасности труда. Пожаровзрывоопасность веществ и материалов

причинам оптимальные значения не могут быть предоставлены. Допустимые значения не вызывают вреда или расстройств здоровья, но могут привести к общим и локальным ощущениям теплового дискомфорта, напряжения механизмов терморегуляции, ухудшения самочувствия и снижения работоспособности.

Параметры микроклимата включают: температуру воздуха, температуру поверхностей, относительную влажность воздуха, скорость воздуха. Оптимальные значения зависят от сезона, а также от физических усилий, прилагаемых работником. При разработке этой работы, поскольку это разработка программного обеспечения, вся работа выполняется сидя без систематических физических усилий.

Мы берём в качестве ссылки значения, указанные в СанПиН 2.2.4.548–96

[42], которые показаны в таблице 6.2.

Таблица 6.2 – Оптимальные и допустимые величины показателей микроклимата на рабочих местах производственных помещений

Оптимальные значения характеристик микроклимата			
Период года	Температура воздуха, С	Относительная влажность воздуха, %	Скорость движения воздуха, м/с
Холодный	22 - 24	40 - 60	0,1
Теплый	23 - 25	40 - 60	0,1
Допустимые значения характеристик микроклимата			
Холодный	20 - 25	15 - 75	0,1
Теплый	21 - 28	15 - 75	0,1 - 0,2

Превышение уровня шума

При анализе шумов, которые могут повлиять на работников, возможными источниками шума являются: рабочее оборудование, компьютерные вентиляторы, копиры и кондиционеры.

Шум оказывает негативное влияние на организм человека: снижает работоспособность, повышает усталость, влияет на органы слуха и центральную нервную систему, снижает внимание.

Уровень шума на рабочем месте программистов не должен превышать 50дБА, а в залах обработки информации на вычислительных машинах 65дБА [43, 44]. Необходимые ограничения пространства показаны в таблице 6.3.

Таблица 6.3 – Допустимые значения уровней звукового давления в октавных полосах частот и уровня звука, создаваемого ПЭВМ

Уровни звукового давления в октавных полосах со среднегеометрическими частотами									Уровн и звука в дБА
31,5 Гц	63,0 Гц	125,0 Гц	250,0 Гц	500,0 Гц	1000,0 Гц	2000,0 Гц	4000,0 Гц	8000,0 Гц	
86,0 дБ	71,0 дБ	61,0 дБ	54,0 дБ	49,0 дБ	45,0 дБ	42,0 дБ	40,0 дБ	38,0 дБ	50,0

В рабочих пространствах, используемых в Томском политехническом университете являются пространства, в которых очень мало шума, особенно учебные комнаты, которые являются местами, которые чаще всего использовались.

Отсутствие или недостаток освещения

Ещё один важный момент, который следует учитывать, - это освещение на рабочем месте, недостаточное освещение негативно влияет на зрение работника, кроме того, это приводит к быстрой усталости, снижает его работоспособность, поскольку вызывает физические дискомфорты, такие как головные боли и бессонница. С учётом правил СП 52.13330.2016 Естественное и искусственное освещение. Актуализированная редакция СНиП 23-05-95, минимальное освещение на рабочих местах не должно отличаться от нормализованного среднего освещения в помещении более чем на 10% [45].

Таблица 6.4 – Требования к освещению помещений промышленных предприятий

Помещения	Рабочая поверхность и плоскость нормирования КЕО и освещенности, и высота плоскости над полом, м	Искусственное освещение					Естественное освещение	
		Освещенность рабочих поверхностей, лк		Объединенный показатель дискомфорта UGR, не более	Коэффициент пульсации освещенности, %, не более	Индекс цветопередачи источников в света Ra	КОЕ * $e_н$, %	
		При комбинированном освещении	При общем освещении				При вернем или комбинированном освещении	При боковом освещении
Кабинеты и рабочие комнаты, офисы, представительства	Г-0,8	400/200	300	21	15	80	3,0	1,0

*КОЕ - коэффициента естественной освещенности.

Также согласно СП 52.13330.2016 Естественное и искусственное освещение. Актуализированная редакция СНиП 23-05-95 уровень освещения на

поверхности рабочего стола при работе с ПЭВМ должен быть в диапазоне от 300 до 500 лк [45].

Выполненная работа имеет тип разработки программного обеспечения, поэтому она включает в себя большую работу визуального типа, упущение этого фактора может привести к заболеваниям зрения.

Расчёт искусственного освещения

Дано помещение с размерами: длина A = 8 м, ширина B = 4 м, высота H = 3,0 м. Высота рабочей поверхности hрп = 0,65 м. Требуется создать освещенность E = 300 лк. Коэффициент отражения стен ρc = 30 %, потолка ρn = 50 %. Коэффициент запаса k =1,5 ; коэффициент неравномерности Z = 1,1. Светильники типа ОД, λ = 1,5. hc = 0,05 м.

Определение расчетную высоту:

$$h = H - h_c - h_{\text{рп}} \tag{6.1}$$

$$2,3\text{м} = 3,0\text{м} - 0,05\text{м} - 0,65\text{м}$$

Расстояние между светильниками:

$$L = \lambda \cdot h \tag{6.2}$$

$$3,45\text{м} = 1,5 \cdot 2,3\text{м}$$

Расстояние от крайнего ряда светильников до стены:

$$\frac{L}{3} \tag{6.3}$$

$$\frac{3,45\text{м}}{3} = 1,15\text{м}$$

Определение количество рядов светильников и количество светильников в ряду:

$$n_{\text{ряд}} = \frac{\left(B - \frac{2}{3}L\right)}{L} + 1 \tag{6.4}$$

$$1 \approx= \frac{\left(4 - \frac{2}{3}3,45\right)}{3,45} + 1$$

$$n_{\text{св}} = \frac{\left(A - \frac{2}{3}L\right)}{l_{\text{св}} + 0,5} \tag{6.5}$$

$$3 \approx= \frac{\left(8 - \frac{2}{3}3,45\right)}{1,53 + 0,5}$$

Светильники необходимо размещать в 1 ряд. В каждом ряду можно установить 3 светильников типа ПВЛ мощностью 80 Вт (с длиной 1,53 м), при этом разрывы между светильниками в ряду составят 50 см. Изображение в масштабе план помещения и размещения на нем светильников отображается на рисунке 6.1. Учитывая, что в каждом светильнике установлено две лампы, общее число ламп в помещении N = 6.

Находим индекс помещения:

$$i = \frac{S}{h(A+B)} \tag{6.6}$$

$$1,16 = \frac{32}{2,3(8+4)}$$

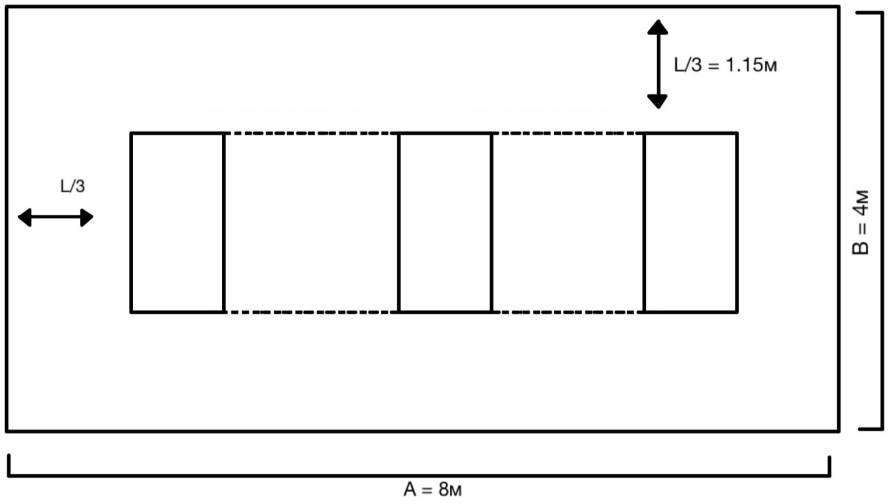

Рисунок 6.1 – План помещения и размещения светильников с люминесцентными

Коэффициент использования светового потока:

$$\eta = 0,48 \tag{6.7}$$

Определяем потребный световой поток ламп в каждом из рядов:

$$\theta = \frac{E_{\text{н}} \cdot S \cdot K_{\text{З}} \cdot Z}{N_{\text{л}} \cdot \eta} \tag{6.8}$$

$$5500 = \frac{300 \cdot 32 \cdot 1,5 \cdot 1,1}{6 \cdot 0,48}$$

Ближайшая стандартная лампа – ЛТБ 80 Вт с потоком 5200 лм. Проверка выполнения условия:

$$-10\% \leq \frac{\theta_{\text{л.станд}} - \theta_{\text{л.расч}}}{\theta_{\text{л.станд}}} \cdot 100\% \leq +20\% \tag{6.9}$$

Результат: -10% $\leq$ -5,77% $\leq$ +20%

Определение электрическую мощность осветительной установки:

$$P = 6 \cdot 80 = 4800\text{Вт}$$

Электрический ток

Поражение электрическим током является опасным фактором производства, и, поскольку при разработке программного обеспечения программист находится в контакте с электронным оборудованием, следует обратить внимание на потенциальные проблемы безопасности, связанные с электричеством. Нормы электробезопасности на рабочем месте регламентируются ГОСТ Р 12.1.019-2017 ССБТ Электробезопасность. Общие требования и номенклатура видов защиты [46].

ГОСТ 12.1.038-82 Система стандартов безопасности труда (ССБТ). Электробезопасность. Предельно допустимые значения напряжений прикосновения и токов, вопросы требований к защите от поражения электрическим током освещены в ГОСТ Р 12.1.019-2009 ССБТ [47, 48].

Поражение человеческого организма электрическим током может быть разнообразным. Разряд, проходящий через ткани, оказывает на него тепловое, электролитическое, биологическое и динамическое действие. Напряжения прикосновения и токи, протекающие через тело работника при неаварийном режиме электроустановки, не должны превышать следующие значений:

– При 50 Гц переменного тока – 2,0 В и 0,3 мА, соответственно.

– При 400 Гц переменного тока – 3,0 В и 0,4 мА.

– При постоянном токе – 8,0 В и 1,0 мА.

Существуют также защитные меры, которые указывают нормальные условия (защита от прямого контакта), которые будут приняты во внимание, это достигается за счёт основной защиты, такой как: безопасное расположение токоведущих частей, размещение их вне зоны досягаемости частями тела,

конечностями; предупредительная световая, звуковая сигнализации, блокировки безопасности, знаки безопасности; основная изоляция; и другие технические мероприятия.

Помещение, где расположено рабочее место оператора ПЭВМ, относится к помещениям без повышенной опасности ввиду отсутствия следующих факторов: сырость, токопроводящая пыль, токопроводящие полы, высокая температура, возможность одновременного прикосновения человека к имеющим соединение с землёй металлоконструкциям зданий, технологическим аппаратам, механизмам и металлическим корпусам электрооборудования.

6.3 Экологическая безопасность

В настоящее время давление во всех странах на заботу об окружающей среде привело к разработке законов, которые помогают заботиться о экосистеме. Одним из способов помочь в уходе за окружающей средой является переработка компьютерного оборудования, хотя это очень сложный процесс из-за сложности его классификации, последующей гомогенизации и отправки для повторного использования. Переработка макулатуры представляет собой многоэтапный процесс, цель которого заключается в восстановлении бумажного волокна и, зачастую, других компонентов бумаги (таких как минеральные наполнители) и использование их в качестве сырья для производства новой бумаги [49].

В нормативном документе ТОИ Р-45-084-01 Типовая инструкция по охране труда при работе на персональном компьютере даются следующие общие рекомендации по снижению опасности для окружающей среды, исходящей от компьютерной техники:

- Применять оборудование, соответствующее санитарным нормам и стандартам экологической безопасности;
- Применять расходные материалы с высоким коэффициентом использо-

вания и возможностью их полной или частичной регенерации;

– Отходы в виде компьютерного лома утилизировать;

– Использовать экономичные режимы работы оборудования.

Люминесцентные лампы, используемые для искусственного освещения на рабочих местах, также требуют специального удаления, поскольку они содержат от 10 до 70 мг ртути, которая является чрезвычайно опасным химическим веществом и может вызывать отравление живых существ, а также загрязнение атмосферы, гидросферы и литосферы. Срок службы этих ламп составляет около 5 лет, после чего они должны быть доставлены для обработки в специальные пункты приёма. Во время разработки и написания ВКР образовывался мусор, такой как: канцелярские принадлежности, бумажные отходы [50].

6.4 Безопасность в чрезвычайных ситуациях

Наиболее вероятным случаем чрезвычайной ситуации в офисе является пожар, поскольку внутри офисов много бумаги, пыли и других легковоспламеняющихся материалов. Возникновение пожара может происходить из-за нескольких факторов, среди которых есть:

– Возникновение короткого замыкания в проводке из-за неисправностей проводки или электрических соединений и электрических распределительных панелей;

– Пожар в устройствах компьютерной техники из-за нарушения изоляции или неисправности самого оборудования;

– Пожар мебели или полов из-за нарушения правил пожарной безопасности, а также ненадлежащего использования дополнительных электроприборов и электроустановок;

– Возгорание устройств искусственного освещения.

Правила, нормализующие события, связанные с пожарами, содержатся в ГОСТ 12.1.004-91 Система стандартов безопасности труда (ССБТ). Пожарная безопасность. Общие требования [51].

Во всех местах, используемых для разработки этой работы, есть нормативная вывеска, ведущая к аварийным выходам, также схемы эвакуации расположены в видимых местах, эти схемы содержат информацию о аварийных выходах и инструкции, которые должны соблюдаться в случае аварийной эвакуации.

На рисунке 6.2 показана схема эвакуации четвёртого этажа 20-го здания Томского политехнического университета.

Рисунок 6.2 – Схема эвакуации четвёртого этажа 20-го здания Томского политехнического университета

6.5 Выводы по разделу

После анализа политики Российской Федерации по вопросам, о безопасности работников и работодателей, можно сделать вывод, что рабочие

места, используемые для выполнения данной работы, соответствуют необходимым правилам, поскольку соблюдаются нормы безопасности, при использовании компьютеров, рабочие места не приводят к ухудшению здоровья ученика, кроме того, установки, в которых выполняется работа, отвечают нормам и правилам, регулирующим соответствующие параметры микроклимата, освещения и электрической безопасности, наконец, экологическая безопасность на предприятии соответствует действующим нормативным документам.

Со стороны студента, чтобы избежать негативного воздействия на здоровье во время выполнения работы, устанавливается план запланированных перерывов, включающих небольшие физические и визуальные упражнения, которые помогают расслабиться.

7 Финансовый менеджмент, русурсоэффективность и ресурсосбережение

По завершении этой исследовательской работы была получена библиотека, содержащая функции создания, управления, доступа, удаления и выпуска именованных общих памятью. Общие памяти будут использоваться для обмена информацией на высокой скорости, это предложение использовать общие памяти в качестве метода межпроцессной связи (Inter-Process Communication — IPC).

Использование высокоскоростных механизмов межсоединений является важным моментом в коммуникационных архитектурах, ориентированных на телеоперацию. В сценарии телеоперации телеоперационное оборудование может иметь очень высокие производственные затраты, поэтому очень важно обеспечить получение данных о состоянии телеоперационного оборудования с максимально возможной скоростью, чтобы избежать запоздалого принятия решений. Запоздало принятие решения может привести к сбою в выполнении желаемой операции или в худшем случае телеоперационное оборудование может привести к серьёзным авариям и разрушению.

В настоящее время перспективность научного исследования определяется ни сколько масштабом открытия, оценить которое на первых этапах жизненного цикла высокотехнологического и ресурсоэффективного продукта бывает достаточно трудно, сколько коммерческой ценностью разработки. Оценка коммерческой ценности (потенциала) разработки является необходимым условием при поиске источников финансирования для проведения научного исследования и коммерциализации его результатов. Это важно для разработчиков, которые должны представлять состояние и перспективы проводимых научных исследований. Через такую оценку учёный может найти партнёра для дальнейшего проведения научного исследования, коммерциализации результатов такого исследования и открытия бизнеса.

Необходимо понимать, что коммерческая привлекательность научного исследования определяется не только превышением технических параметров над предыдущими разработками, но и насколько быстро разработчик сумеет найти ответы на такие вопросы – будет ли продукт востребован рынком, какова будет его цена, чтобы удовлетворить потребителя, каков бюджет научного проекта, сколько времени потребуется для выхода на рынок и т.д.

Целью раздела «Финансовый менеджмент, ресурсоэффективность и ресурсосбережение» является определение перспективности и успешности научного и технического исследования, оценка его эффективности, уровня возможных рисков, разработка механизма управления и сопровождения конкретных проектных решений на этапе реализации.

Для достижения обозначенной цели необходимо решить следующие задачи:

- разработка общей экономической идеи проекта, формирование концепции проекта;
- организация работ по научно-исследовательскому проекту;
- определение возможных альтернатив проведения научных исследований;
- планирование научно-исследовательских работ;
- оценки коммерческого потенциала и перспективности проведения научных исследований с позиции ресурсоэффективности и ресурсосбережения;
- определение ресурсной (ресурсосберегающей), финансовой, бюджетной, социальной и экономической эффективности исследования.

С учётом решения данных задач была сформирована структура и содержание раздела «Финансовый менеджмент, ресурсоэффективность и ре-

сурсосбережение».

Прототипная работа была протестирована с использованием общей памяти для обмена данными о состоянии телеоперированного оборудования, такого как беспилотные летательные аппараты и мобильные манипуляторы. Информация была доступна из нескольких приложений одновременно.

7.1 Предпроектный анализ

7.1.1 Потенциальные потребители результатов исследования

Целевым потребителем этого исследования являются приложения или исследовательские проекты, требующие доступности источника данных с высокой скоростью. Клиенты, которые подключаются и потребляют данные, - это приложения, процессы или потоки, которые могут использовать библиотеки динамических ссылок. Следует подчеркнуть, что предлагаемая модель взаимосвязи эффективна, когда функция списка ожидания данных не требуется.

В качестве успешных примеров использования этого исследовательского предложения может быть предложен портфель проектов, которые использовали предлагаемый книжный магазин, чтобы потенциальные клиенты могли проверить, соответствует ли этот книжный магазин их потребностям, просматривая различные ранее выполненные реализации.

7.1.2 Анализ конкурентных технических решений с позиции ресурсоэффективности и ресурсосбережения

Анализ конкурентных технических решений с позиции ресурсоэффективности и ресурсосбережения позволяет провести оценку сравнительной эффективности научной разработки и определить направления для ее будущего повышения.

В настоящее время на рынке существует множество вариантов межпроцессного обмена информацией, однако все они ориентированы на охват

как можно большего объёма рынка, оставляя в стороне проекты, в которых не требуется столько функций безопасности данных, а их главная потребность заключается только в доступе к самым современным данным и в том, чтобы процесс осуществлялся как можно быстрее, например в исследовательских проектах телеоперации в режиме реального времени.

В таблице 7.1 приведена оценка конкурентов, где Ф – разрабатываемый проект, к1 – исследовательская группа "REWA-RD RD (Renewable Energy and Web Architecture к2 – Исследовательская группа по моделям производства программного обеспечения — GrIMPSoft .

Таблица 7.1 – Оценочная карта для сравнения конкурентных технических решений (разработок)

Критерии оценки	**Вес критерия**	**Баллы**			**Конкурентоспособность**		
		$Б_ф$	$Б_{к1}$	$Б_{к2}$	$К_ф$	$К_{к1}$	$К_{к2}$
1	2	3	4	5	6	7	8
Технические критерии оценки ресурсоэффективности							
1. Повышение производительности труда пользователя	0,15	5	4	5	0,75	0,60	0,75
2. Точность	0,15	5	3	4	0,75	0,45	0,60
3. Скорость	0,19	5	3	4	0,95	0,57	0,76
4. Удобство в эксплуатации	0,15	4	5	5	0,60	075	0,75
Экономические критерии оценки эффективности							
1. Конкурентоспособность продукта	0,05	5	4	4	0,25	0,20	0,20
2. Цена	0,15	5	4	4	0,75	0,60	0,60
3. Срок выполнения работ	0,16	4	5	3	0,64	0,80	0,48
Итого	**1**	**33**	**28**	**29**	**4,69**	**3,97**	**4,14**

Критерии оценки подбираются, исходя из выбранных объектов сравнения с учётом их технических и экономических особенностей разработки, создания и эксплуатации.

Вес показателей в сумме должны составлять 1. Позиция разработки и конкурентов оценивается по каждому показателю по пятибалльной шкале, где 1 – наиболее слабая позиция, а 5 – наиболее сильная.

Анализ конкурентных технических решений определяется по формуле:

$$K = \sum B_i \cdot \text{Б}_i \tag{7.1}$$

где: К – конкурентоспособность научной разработки или конкурента;

B_i - вес показателя (в долях единицы);

Б_i - балл i-го показателя.

Основываясь на анализе, проведённом конкурентами, можно сказать, что есть конкурентное преимущество в определённых характеристиках, которое связано с ценой, производительностью, точностью и скоростью разрабатываемого проекта. Однако уязвимость разрабатываемого проекта заключается в том, что его способ использования/эксплуатации представляет собой большую сложность.

7.1.3 SWOT-анализ

SWOT – представляет собой комплексный анализ научного и исследовательского проекта, применяется для исследования внешней и внутренней среды проекта (таблица 7.2). Анализ проводится в 3 этапа.

Первый этап заключается в описании сильных и слабых сторон проекта, в выявлении возможностей и угроз для реализации проекта, которые проявились или могут появиться в его внешней среде.

Второй этап состоит в выявлении соответствия сильных и слабых сторон научно-исследовательского проекта внешним условиям окружающей среды.

Интерактивная матрица проекта представлена в таблицах 7.3,7.4, 7.5, 7.6. Каждый фактор помечается либо знаком «+» (означает сильное соответствие сильных сторон возможностям), либо знаком «-» (что означает слабое соответствие);«0» – если есть сомнения в том, что поставить «+» или «-».

В рамках третьего этапа должна быть составлена окончательная мат-

Таблица 7.2 – Матрица SWOT-анализа

Сильные стороны	**Слабые стороны**
С1. Нет никаких ограничений на использование или выпуск исходного кода и документации. С2. Существует несколько успешных исследовательских проектов, в которых использовался прототип этого предложения. С3. Поддержка исследовательских групп, работающих в области телеоперации в реальном времени.	Сл1. Он ориентирован на конкретный сектор рынка, который очень мал. Сл2. Он не обладает функциями защиты целостности хранимых данных. Сл3. Ограничивается только основными типами переменных (Integer, Float, Double, String)
Возможности	**Угрозы**
В1. Существует не так много IPC, ориентированных на соединения в реальном времени. В2. Рост числа программистов, заинтересованных в улучшении. В3. Растущий заинтересованность к новым исследовательским группам в рамках одной и той же исследовательской сети.	У1. Отсутствие спроса у пользователей У2. Неактуальность тематики проекта У3. Появление сильных конкурентов.

Таблица 7.3 – Сильные стороны

	Сильные стороны проекта			
Возможности проекта		С1.	С2.	С3.
	В1.	-	-	0
	В1.	+	+	-
	В1.	+	+	+

Таблица 7.4 – Слабые стороны

	Слабые стороны проекта			
Возможности проекта		Сл1.	Сл2.	Сл3.
	В1.	-	0	0
	В1.	-	+	+
	В1.	-	+	+

Таблица 7.5 – Сильные стороны

	Сильные стороны проекта			
Угрозы проекта		С1.	С2.	С3.
	У1.	-	+	+
	У2.	-	+	+
	У3.	-	+	+

рица SWOT-анализа со стратегиями для каждого анализируемого случая (Таблица 7.7).

После проведения SWOT-анализа были разработаны чёткие страте-

Таблица 7.6 – Слабые стороны

	Слабые стороны проекта			
Угрозы проекта		Сл1.	Сл2.	Сл3.
	У1.	+	-	-
	У2.	+	-	-
	У3.	-	-	-

гии, направленные на то, чтобы быть готовыми к потенциальным угрозам и пытаться укрепить слабые стороны.

7.1.4 Оценка готовности проекта к коммерциализации

На какой бы стадии выполнения проекта не находилась научная разработка полезно оценить степень ее готовности и уровня собственных знаний для осуществления проекта. к коммерциализации и выяснить уровень собственных знаний для ее проведения (или завершения). Показатели о степени проработанности проекта с позиции коммерциализации и компетенциям разработчика научного проекта представлены в таблице 5.

При проведении анализа по таблице, по каждому показателю ставится оценка по пятибалльной шкале. При оценке степени проработанности научного проекта 1 балл означает не проработанность проекта, 2 балла – слабую проработанность, 3 балла – выполнено, но в качестве не уверен, 4 балла – выполнено качественно, 5 баллов – имеется положительное заключение независимого эксперта. Для оценки уровня имеющихся знаний у разработчика система баллов принимает следующий вид: 1 означает не знаком или мало знаю, 2 – в объёме теоретических знаний, 3 – знаю теорию и практические примеры применения, 4 – знаю теорию и самостоятельно выполняю, 5 – знаю теорию, выполняю и могу консультировать.

Оценка готовности научного проекта к коммерциализации (или уровень имеющихся знаний у разработчика) определяется по формуле:

$$Б_{сум} = \sum Б_i \tag{7.2}$$

Таблица 7.7 – SWOT-анализ

	Сильные стороны	**Слабые стороны**
	С1. Нет никаких ограничений на использование или выпуск исходного кода и документации. С2. Существует несколько успешных исследовательских проектов, в которых использовался прототип этого предложения. С3. Поддержка исследовательских групп, работающих в области телеоперации в реальном времени.	Сл1. Он ориентирован на конкретный сектор рынка, который очень мал. Сл2. Он не обладает функциями защиты целостности хранимых данных. Сл3. Ограничивается только основными типами переменных (Integer, Float, Double, String)
Возможности В1. Существует не так много IPC, ориентированных на соединения в реальном времени. В2. Рост числа программистов, заинтересованных в улучшении. В3. Растущий заинтересованность к новым исследовательским группам в рамках одной и той же исследовательской сети.	Стратегия ВС В2С1С2: не имея ограничений на выпуск документации и кода, может быть использован для выпуска лицензионного проекта свободного программного обеспечения и позволить другим программистам вносить улучшения. В3С1С2С3: с помощью местной исследовательской группы возможно создавать рабочие программы вместе с другими заинтересованными исследовательскими группами, чтобы поделиться всей информацией и способом использования предложения.	Стратегия ВСл В2Сл2Сл3: Можно использовать интерес новых программистов для реализации некоторого метода контроля целостности данных, а также увеличить поддержку новых типов данных. В3Сл2Сл3: Можно использовать интерес других исследовательских групп, чтобы выяснить, какие методы и какие типы данных лучше всего подходят для реализации.
Угрозы У1. Отсутствие спроса у пользователей У2. Неактуальность тематики проекта У3. Появление сильных конкурентов.	Стратегия УС У1У2У3Сл2Сл3: Использование успешных случаев и названия исследовательской группы, поддерживающей работу, чтобы повысить интерес потенциальных пользователей, а также показать актуальность работы и противостоять потенциальным конкурентам.	Стратегия УСл У1УСл1: Увеличение сектора рынка, на который направлено предложение, чтобы увеличить спрос пользователей. У2Сл2: Интеграция функций защиты данных для повышения актуальности темы проекта. У3Сл3: Увеличение поддерживаемых типов переменных для достижения в различных секторах рынка.

где: $Б_{сум}$ – суммарное количество баллов по каждому направлению;

$Б_i$ – балл по i-му показателю.

Таблица 7.8 – SWOT-анализ

№ п/п	Наименование	Степень проработанности научного проекта	Уровень имеющихся знаний у разработчика
1.	Определён имеющийся научно-технический задел	5	5
2.	Определены перспективные направления коммерциализации научно-технического задела	3	4
3.	Определены отрасли и технологии для предложения на рынке	4	4
4.	Определена товарная форма научно-технического задела для представления на рынок	3	3
5.	Определены авторы и осуществлена охрана их прав	4	4
6.	Проведена оценка стоимости интеллектуальной собственности	3	3
7.	Проведены маркетинговые исследования рынков сбыта	2	2
8.	Разработан бизнес-план коммерциализации научной разработки	2	2
9.	Определены пути продвижения научной разработки на рынок	4	4
10.	Разработана стратегия (форма) реализации научной разработки	5	5
11.	Проработаны вопросы международного сотрудничества и выхода на зарубежный рынок	3	3
12.	Проработаны вопросы использования услуг инфраструктуры поддержки, получения льгот	4	4
13.	Проработаны вопросы финансирования коммерциализации научной разработки	3	3
14.	Имеется команда для коммерциализации научной разработки	2	2
15.	Проработан механизм реализации научного проекта	5	5
	ИТОГО БАЛЛОВ	52	53

Итоговые значение $Б_{сум}$, позволяет говорить о мере готовности научной разработки и её разработчика к коммерциализации. Таким образом, в ходе суммирования баллов выяснено, что значение $Б_{сум}$ попадает диапазоне от 45 до 59, это говорить о том, что перспективность разработка выше сред-

него.

По результатам оценки выделяются слабые стороны исследования, дальнейшего улучшения необходимо провести маркетинговые исследования рынков сбыта, разработать бизнес-план коммерциализации научной разработки проработать вопросы международного сотрудничества и выхода на зарубежный рынок. Для повышения этого фактора следует привлечь недостающих специалистов и быть готовым к новым возможностям, которые на данный момент ограничены.

7.2 Планирование управления научно-техническим проектом

Группа процессов планирования состоит из процессов, осуществляемых для определения общего содержания работ, уточнения целей и разработки последовательности действий, требуемых для достижения данных целей.

План управления научным проектом должен включать в себя следующие элементы:

- иерархическая структура работ проекта;
- контрольные события проекта;
- план проекта;
- бюджет научного исследования.

7.2.1 Иерархическая структура работ проекта

Иерархическая структура работ (ИСР) – детализация укрупненной структуры работ. В процессе создания ИСР структурируется и определяется содержание всего проекта (рисунок 7.1).

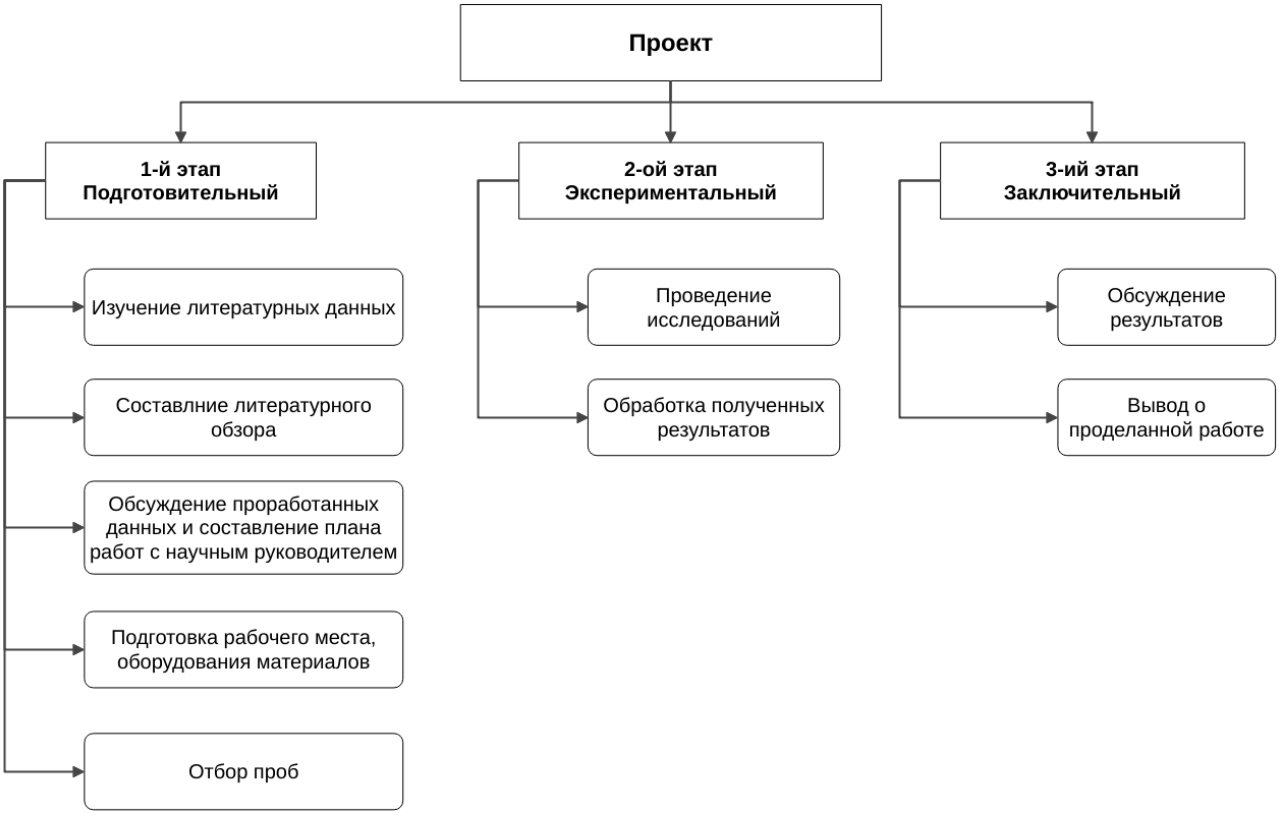

Рисунок 7.1 – Иерархическая структура работ

7.2.2 План проект

В рамках планирования научного проекта построены календарный график проекта (таблица 7.9 и рисунок 7.2).

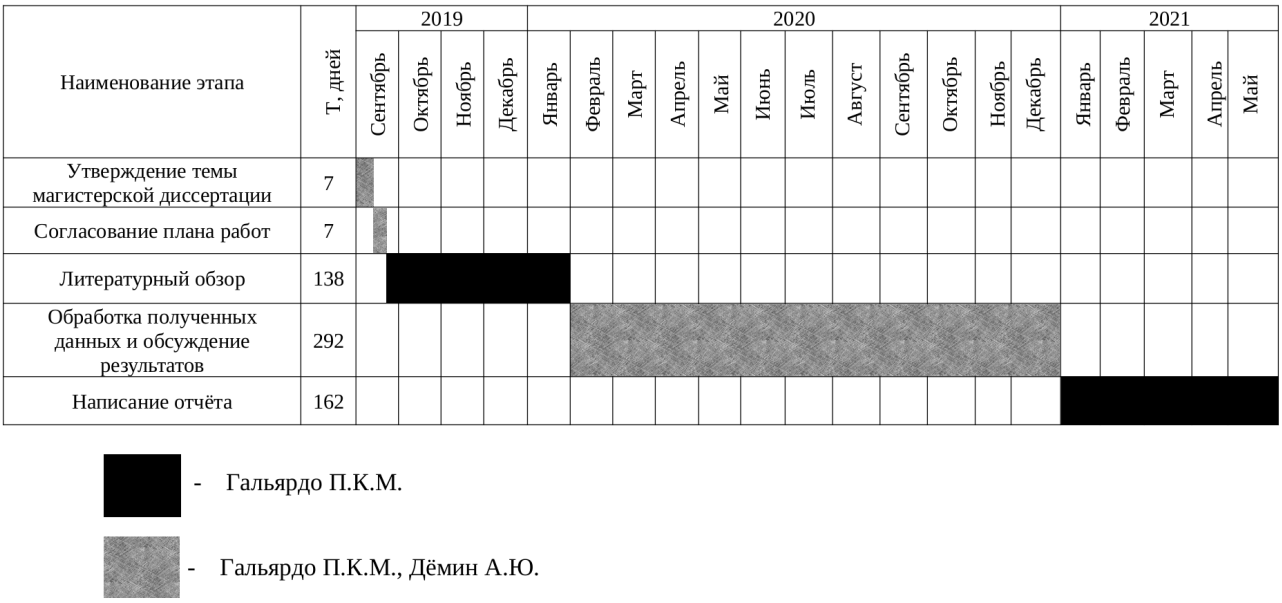

Рисунок 7.2 – Календарный план график проведения НИОКР по теме

Таблица 7.9 – Календарный план проекта

Название	Длительность, дни	Дата начала работ	Дата окончания работ	Состав участников
Утверждение темы магистерской диссертации	7	01.09.19	07.09.19	Гальярдо П.К.М, Дёмин А.Ю.
Согласование плана работ	7	08.09.19	15.09.19	Гальярдо П.К.М, Дёмин А.Ю.
Литературный обзор	138	16.09.19	31.01.20	Гальярдо П.К.М, Дёмин А.Ю.
Обработка полученных данных и обсуждение результатов	292	01.02.20	20.12.20	Гальярдо П.К.М, Дёмин А.Ю.
Написание отчета	162	21.12.20	31.05.21	Гальярдо П.К.М
Итого:	606			

7.3 Бюджет научного исследования

При планировании бюджета научного исследования должно быть обеспечено полное и достоверное отражение всех видов планируемых расходов, необходимых для его выполнения. В процессе формирования бюджета, планируемые затраты сгруппированы по статьям. К материальным затратам относятся: приобретаемые сырье и материалы, канцелярские принадлежности, картриджи и т.п. В данном исследовании выделены следующие статьи:

1. Сырье, материалы, покупные изделия и полуфабрикаты;
2. Специальное программы для научных работ;
3. Заработная плата;
4. Оплата работ, выполняемых сторонними организациями и предприятиями;
5. Накладные расходы.

Сырье, материалы, покупные изделия и полуфабрикаты (за вычетом отходов). В эту статью включаются затраты на приобретение всех видов материалов, комплектующих изделий и полуфабрикатов, необходимых для выполнения работ по данной теме (таблица 7.10).

Таблица 7.10 – Расчёт затрат по статье «Сырье и материалы»

Наименование	Количество, шт	Цена за единицу, руб.	Сумма, руб.
Тетрадь	1	40	40
Ручка шариковая	5	31	150
Краска для принтера	5	600	3000
Бумага для принтера, формат А4, пачка	3	350	1050
Электроэнергия, кВт*ч	130	3,5	455
Всего за материалы			4700
Транспортно-заготовительные расходы (5%)			2350
Итого по статье			7050

Специальное оборудование для научных (экспериментальных) работ. В данную статью включены все затраты, связанные с приобретением специального оборудования, необходимого для проведения работ по теме НИР (таблица 7.11).

Таблица 7.11 – Расчёт затрат по статье «Спецоборудование для научных работ»

№ п/п	Наименование оборудования	Кол-во единиц оборудования	Цена единицы оборудования, руб.	Общая стоимость оборудования, руб.
1	Компьютер (Acer)	1	60000	60000
1	Windows 10 Pro	1	12500	12500
Итого, руб.				72500

Расчёт основной заработной платы. В настоящую статью включается основная заработная плата научных и инженерно-технических работников, рабочих макетных мастерских и опытных производств, непосредственно участвующих в выполнении работ по данной теме. Величина расходов по заработной плате определяется исходя из трудоёмкости выполняемых работ и действующей системы оплаты труда. Расчёт основной заработной платы сво-

дится в таблице 14.

$$C_{\text{зп}} = З_{\text{осн}} + З_{\text{доп}} \tag{7.3}$$

где $З_{\text{осн}}$ – основная заработная плата;

$З_{\text{доп}}$ – дополнительная заработная плата

Основная заработная плата ($З_{\text{осн}}$) руководителя (лаборанта, инженера) от предприятия (при наличии руководителя от предприятия) рассчитывается по следующей формуле:

$$З_{\text{осн}} = З_{\text{дн}} \cdot T_{\text{раб}} \tag{7.4}$$

где $З_{\text{осн}}$ – основная заработная плата одного работника;

$T_{\text{раб}}$ – продолжительность работ, выполняемых научно-техническим работником, раб. дн.;

$З_{\text{дн}}$ – среднедневная заработная плата работника, руб.

Среднедневная заработная плата рассчитывается по формуле:

$$З_{\text{дн}} = \frac{З_{\text{м}} \cdot M}{F_{\text{д}}} \tag{7.5}$$

где: $З_{\text{м}}$ – месячный должностной оклад работника, руб.;

М – количество месяцев работы без отпуска в течение года:

при отпуске в 24 раб. дня М =11,2 месяца, 5-дневная неделя;

при отпуске в 56 раб. дней М=10,4 месяца, 6-дневная неделя;

$F_{\text{д}}$ – действительный годовой фонд рабочего времени научно-технического персонала, раб. дн.

Расчёт заработной платы научно – производственного и прочего персонала проекта проводили с учётом работы 2-х человек – научного руководителя и исполнителя. Баланс рабочего времени исполнителей представлен в таблице 7.12.

При расчёте заработной платы научно-производственного и прочего

Таблица 7.12 – Баланс рабочего времени

Показатели рабочего времени	Руководитель	Эксперт	Магистрант
Календарное число дней	365	365	365
Количество нерабочих дней - выходные дни - праздничные дни	 53 12	 106 12	 106 12
Потери рабочего времени - отпуск - невыходы по болезни	 56 0	 24 0	 24 0
Действительный годовой фонд рабочего времени	244	223	223

персонала проекта учитывались месячные должностные оклады работников, которые рассчитывались по формуле:

$$З_м = З_б \cdot k_p \tag{7.6}$$

где $З_б$ – базовый оклад, руб.;

k_p – районный коэффициент, равный 1,3 (для Томска).

Расчет основной заработной платы приведён в таблице 7.13.

Таблица 7.13 – Расчёт основной заработной платы

Исполнители	$З_б$, руб.	k_p	$З_м$, руб	$З_{дн}$, руб.	T_p, раб. дн.	$З_{осн}$, руб.
Руководитель Доцент ОИТ - к.т.н.	35111,50	1,3	45644,95	1496,56	3	4489,68
Эксперт Ассистент ОИТ	22695	1,3	29503,50	1481,79	18	26672,22
Инженер	22695,68	1,3	29504,38	1481,83	125	185228,75

Дополнительная заработная плата научно-производственного персонала. В данную статью включается сумма выплат, предусмотренных законодательством о труде, например, оплата очередных и дополнительных отпусков; оплата времени, связанного с выполнением государственных и общественных обязанностей; выплата вознаграждения за выслугу лет и т.п. (в среднем – 12% от суммы основной заработной платы). Дополнительная заработная плата рассчитывается исходя из 10-15% от основной заработной платы, ра-

ботников, непосредственно участвующих в выполнение темы:

$$З_{доп} = З_{осн} \cdot k_{доп} \tag{7.7}$$

где $З_{доп}$ – дополнительная заработная плата, руб.;

$k_{доп}$ – коэффициент дополнительной зарплаты;

$З_{осн}$ – основная заработная плата, руб.

В таблице 7.14 приведена форма расчёта основной и дополнительной заработной платы.

Таблица 7.14 – Заработная плата исполнителей НТИ

Заработная плата	Руководитель	Эксперт	Магистрант
Основная зарплата	4489,68	26672,22	185228,75
Дополнительная зарплата	448,96	2667,22	18522,87
Итого по статье $C_{зп}$	4938,64	29339,44	203751,62

Отчисления на социальные нужды. Статья включает в себя отчисления во внебюджетные фонды.

$$C_{внеб} = K_{внеб} \cdot (З_{осн} + З_{доп}) \tag{7.8}$$

где $K_{внеб}$ – коэффициент отчисления на уплату во внебюджетные фонды.

На 2014 г. в соответствии с Федеральным законом от 24.07.2009 №212-ФЗ установлен размер страховых взносов равный 30%. На основании пункта 1 ст.58 закона №212-ФЗ для учреждений, осуществляющих образовательную и научную деятельность в 2014 году, водится пониженная ставка – 27,1%. Стипендиальный выплаты студентам, магистрам и аспирантам не облагаются налогом. Отчисления на социальные нужды составляют:

$$C_{внеб} = K_{внеб} \cdot (З_{осн} + З_{доп})$$

$$1481,59 = 0,3 \cdot (4489,68 + 448,96)$$

Накладные расходы. Расчёт накладных расходов провели по следующей формуле:

$$C_{\text{накл}} = K_{\text{накл}} \cdot (З_{\text{осн}} + З_{\text{доп}}) \tag{7.9}$$

$$38084,75 = 0,16 \cdot (4938,64 + 29339,44 + 203751,62)$$

Таким образом, затраты проекта составляет 357146,04, которые приведены в таблице 7.15.

7.4 Оценка сравнительной эффективности исследования

Определение эффективности происходит на основе расчёта интегрального показателя эффективности научного исследования. Его нахождение связано с определением двух средневзвешенных величин: финансовой эффективности и ресурсоэффективности.

Интегральный показатель финансовой эффективности научного исследования получают в ходе оценки бюджета затрат трёх (или более) вариантов исполнения научного исследования. Для этого наибольший интегральный показатель реализации технической задачи принимается за базу расчёта (как знаменатель), с которым соотносится финансовые значения по всем вариантам исполнения.

Интегральный финансовый показатель разработки определяется по следующей формуле:

$$I_{\text{финр}}^{\text{исп.}i} = \frac{Ф_{pi}}{Ф_{max}} \tag{7.10}$$

где: $I_{\text{финр}}^{\text{исп.}i}$ – интегральный финансовый показатель разработки;

$Ф_{pi}$ – стоимость i-го варианта исполнения;

$Ф_{max}$ – максимальная стоимость исполнения научно-исследовательского проекта (в т.ч. аналоги).

Таблица 7.15 – Затраты научно-исследовательской работы

Вид исследования	Затраты по статьям								
	Сырье, материалы (за вычетом возвратных отходов), покупные изделия и полуфабрикаты	Специальное оборудование для научных (экспериментальных) работ	Основная заработная плата	Дополнительная заработная плата	Отчисления на социальные нужды	Прочие прямые расходы	Накладные расходы	Итого плановая себестоимость
Данное исследование	7050,00	72500,00	216390,65	21639,05	1481,59	-	38084,75	357146,04
Аналог	4000	300000	999919,2	99991,6	429653,2	-	789928,6	2623492,60

Полученная величина интегрального финансового показателя разработки отражает соответствующее численное увеличение бюджета затрат разработки в разах (значение больше единицы), либо соответствующее численное удешевление стоимости разработки в разах (значение меньше единицы, но больше нуля).

Интегральный показатель ресурсоэффективности вариантов исполнения объекта исследования можно определить по следующей формуле:

$$I_{pi} = \sum a_i \cdot b_i \tag{7.11}$$

где: I_{pi} – интегральный показатель ресурсоэффективности для i-го варианта исполнения разработки;

a_i – весовой коэффициент i-го варианта исполнения разработки;

$\mathrm{b}_i^a, \mathrm{b}_i^p$ – бальная оценка i-го варианта исполнения разработки, устанавливается экспертным путём по выбранной шкале оценивания;

n – число параметров сравнения.

Расчёт интегрального показателя ресурсоэффективности приведён в форме таблицы (таблице 7.16).

Таблица 7.16 – Сравнительная оценка характеристик вариантов исполнения проекта

ПО Критерии	Весовой коэффициент параметра	Текущий проект	Аналог 1	Аналог 2
1. Выход продукта	0,1	5	4	3
2. Удобство в эксплуатации	0,25	5	5	3
3. Надежность	0,2	4	4	3
4. Безопасность	0,15	4	4	4
5. Простота эксплуатации	0,15	5	4	4
6. Возможность автоматизации данных	0,15	4	3	5
Итого	1	27	24	22

$$I_m^p = 5 \cdot 0,1 + 5 \cdot 0,25 + 4 \cdot 0,2 + 4 \cdot 0,15 + 5 \cdot 0,15 + 4 \cdot 0,15$$

$$I_m^p = 4,50$$

$$I_1^A = 4 \cdot 0,1 + 5 \cdot 0,25 + 4 \cdot 0,2 + 4 \cdot 0,15 + 4 \cdot 0,15 + 3 \cdot 0,15$$

$$I_1^A = 4,1$$

$$I_2^A = 3 \cdot 0,1 + 3 \cdot 0,25 + 3 \cdot 0,2 + 4 \cdot 0,15 + 4 \cdot 0,15 + 5 \cdot 0,15$$

$$I_2^A = 3,6$$

Интегральный показатель эффективности разработки $I^p_{\text{финр}}$ и аналога $I^a_{\phi uhp}$ определяется на основании интегрального показателя ресурсоэффективности и интегрального финансового показателя по формуле:

$$I^{\text{р}}_{\text{финр}} = \frac{I^p_m}{I^p_\phi}; I^a_{\text{финр}} = \frac{I^a_m}{I^a_\phi} \tag{7.12}$$

Сравнение интегрального показателя эффективности текущего проекта и аналогов позволит определить сравнительную эффективность проекта. Сравнительная эффективность проекта определяется по формуле:

$$Э_{ср} = \frac{I^p_{\text{финр}}}{I^a_{\text{финр}}} \tag{7.13}$$

где: $Э_{ср}$ – сравнительная эффективность проекта;

$I^p_{\text{финр}}$ – интегральный показатель разработки;

$I^a_{\text{финр}}$ – интегральный технико-экономический показатель аналога.

Сравнительная эффективность разработки по сравнению с аналогами представлена в таблице 7.17.

Выводы: Сравнение значений интегральных показателей эффективности позволяет понять, что разработанный вариант проведения проекта является наиболее эффективным при решении поставленной в магистерской

Таблица 7.17 – Сравнительная эффективность разработки

№ п/п	Показатели	Разработка	Аналог 1	Аналог 2
1	Интегральный финансовый показатель разработки	0,38	0,25	0,2
2	Интегральный показатель ресурсоэффективности разработки	4,5	4,1	3,6
3	Интегральный показатель эффективности	11,84	16,4	18
4	Сравнительная эффективность вариантов исполнения	1	0,72	0,66

диссертации технической задачи с позиции финансовой и ресурсной эффективности.

8 Заключение

В ходе работы были разработаны библиотеки динамических ссылок, которые содержат функции для создания, управления, доступа и освобождения общих памятей. Общие памяти используются в качестве механизма межпроцессного обмена данными (IPC).

Чтобы подтвердить полезность работы, были представлены некоторые из реализаций, в которых используется выполненное предложение IPC для обмена данными. Все результаты, полученные и представленные в разделе "Результаты экспериментов", соответствуют научным публикациям, индексируемым в Scopus.

Исследование, проведенное по разработке и использованию библиотек в операционных системах Windows, также внесло большой вклад, как и экспериментальные результаты, результаты были опубликованы и проиндексированы в Scopus.

В настоящее время активно осуществляются 2 проекта: один в исследовательской группе ARSI при Университете Вооруженных сил Эквадора, штаб-квартира Латакунга, и другой проект в исследовательской группе SISAu при Технологическом университете Индоамерики, штаб-квартира Амбато, Эквадор. Эти проекты используют это предложение в качестве метода обмена данными.

Библиотеки, разработанные в данной работы, продолжают постоянно совершенствоваться. На уровне разработки планируется перейти от циклов к хэш-таблицам в качестве метода перемещения между общими памятями. На уровне новых функций анализируется целесообразность добавления функции обмена изображениями между несколькими приложениями через общую память.

Список литературы

1. Shamsudeen.E Dr. A Study on Interprocess Communications in Distributed Computer systems // IOSR Journal of Engineering (IOSRJEN). — 2018. — Апр. — Т. 08, № 4. — С. 09–15. — Режим доступа: http://www.iosrjen.org/Papers/vol8_issue4/Version-3/B0804030915.pdf.

2. Distributed Shared Memory on IP Networks : techreport / University of Wisconsin-Madison Department of Computer Science ; исполн.: Chuck Tsen Dan Gibson. — http://pages.cs.wisc.edu/ gibson/pdf/dsmoip.pdf : 2005.

3. TreadMarks: Shared Memory Computing on Networks of Workstations / Cristiana Amza, Alan L. Cox, Sandhya Dwarkadas и др. // Computer. — 1996. — Фев. — Т. 29, № 2. — С. 18–28. — Режим доступа: https://doi.org/10.1109/2.485843.

4. Cabillic Gilbert, Puaut Isabelle. Stardust: An Environment for Parallel Programming on Networks of Heterogeneous Workstations // Journal of Parallel and Distributed Computing. — 1997. — Т. 40, № 1. — С. 65–80. — Режим доступа: https://www.sciencedirect.com/science/article/pii/S074373159691271X.

5. Khaneghah Ehsan Mousavi, Mirtaheri Seyedeh Leili, Sharifi Mohsen. Evaluating the Effect of Inter Process Communication Efficiency on High Performance Distributed Scientific Computing // 2008 IEEE/IFIP International Conference on Embedded and Ubiquitous Computing. — Т. 1. — 2008. — С. 366–372.

6. hui Cheng Xiao, Zhang Liang. A research of inter-process communication based on shared memory and address-mapping // Proceedings of 2011 International Conference on Computer Science and Network Technology. — Т. 1. — 2011. — С. 111–114.

7. Adve S.V., Gharachorloo K. Shared memory consistency models: a tutorial // Computer. — 1996. — Т. 29, № 12. — С. 66–76.

8. Mike Jacobs Michael Satran. Clipboard [Электронный ресурс]. — 2018. — Май. — Режим доступа: https://docs.microsoft.com/en-us/windows/win32/dataxchg/clipboard (дата обращения: 05.06.2021).

9. McLean Schofield David Coulter Drew Batchelor Michael Satran. Component Object Model (COM) [Электронный ресурс]. — 2018. — Май. — Режим доступа: https://docs.microsoft.com/en-us/windows/win32/com/component-object-model–com–portal (дата обращения: 05.06.2021).

10. McLean Schofield Kent Sharkey David Coulter Drew Batchelor Michael Satran. File Mapping [Электронный ресурс]. — 2018. — Май. — Режим доступа: https://docs.microsoft.com/en-us/windows/win32/memory/file-mapping (дата обращения: 05.06.2021).

11. McLean Schofield Kent Sharkey David Coulter Mike Jacobs Michael Satran. Mailslots [Электронный ресурс]. — 2018. — Май. — Режим доступа: https://docs.microsoft.com/en-us/windows/win32/ipc/mailslots (дата обращения: 05.06.2021).

12. McLean Schofield Kent Sharkey Drew Batchelor Michael Satran. Pipes (Interprocess Communications) [Электронный ресурс]. — 2018. — Май. — Режим доступа: https://docs.microsoft.com/en-us/windows/win32/ipc/pipes (дата обращения: 05.06.2021).

13. Steven White Kent Sharkey David Coulter Drew Batchelor Michael Satran. Windows Sockets 2 [Электронный ресурс]. — 2018. — Режим доступа: https://docs.microsoft.com/en-us/windows/win32/winsock/windows-sockets-start-page-2 (дата обращения: 05.06.2021).

14. Hurd Cuthbert C. Early Computers at IBM // Annals of the History of Computing. — 1981. — Т. 3, № 2. — С. 163–182.

15. Ryckman George F. 17. The IBM 701 Computer at the General Motors Research Laboratories // Annals of the History of Computing. — 1983. — Т. 5, № 2. — С. 210–212.

16. Method and system for dynamic-link library : patentus US5375241A Wash James E. Walsh, Redmond ; — Режим доступа: https://patentimages.storage.googleapis.com/5c/6c/86/c7f62b3fb9b181/US 5375241.pdf.

17. Han Liang Daniel Rugerio Lu Chen Simonx Xu. What is a DLL [Электронный ресурс]. — 2020. — Сент. — Режим доступа: https://docs.microsoft.com/en-US/troubleshoot/windows-client/deployment/dynamic-link-library (дата обращения: 05.06.2021).

18. Young Michael J. Software Tools for Os/2: Creating Dynamic Link Libraries. — Addison-Wesley, 1989. — ISBN: 0-201-51787-6.

19. Pastor Javier. La memoria RAM es superrápida y (relativamente) barata, nos preguntamos por qué no usarla como unidad de disco [Электронный ресурс]. — 2018. — Дек. — Режим доступа: https://www.xataka.com/componentes/memoria-ram-superrapida-relativamente-barata-nos-preguntamos-que-no-usarla-como-unidad-disco (дата обращения: 05.06.2021).

20. Wikipedia. Random-access memory [Электронный ресурс]. — 2021. — Режим доступа: https://en.wikipedia.org/wiki/Random-access_memory (дата обращения: 05.06.2021).

21. Wikipedia. Hard disk drive [Электронный ресурс]. — 2021. — Режим доступа: https://en.wikipedia.org/wiki/Hard_disk_drive (дата обращения: 05.06.2021).

22. Wikipedia. Transfer (computing) [Электронный ресурс]. — 2021. — Режим доступа: https://en.m.wikipedia.org/wiki/Transfer_(computing) (дата обращения: 05.06.2021).

23. CodeLite. CodeLite - A free, Open Source, Cross Platform C, C++, PHP and Node.js IDE [Электронный ресурс]. — 2021. — Режим доступа: https://codelite.org/ (дата обращения: 05.06.2021).

24. Oracle. JDK 16 Documentation [Электронный ресурс]. — 2021. — Режим доступа: https://docs.oracle.com/en/java/javase/16/index.html (дата обращения: 05.06.2021).

25. TDM-GCC. TDM-GCC Official webpage [Электронный ресурс]. — 2020. — Режим доступа: https://jmeubank.github.io/tdm-gcc/download/ (дата обращения: 05.06.2021).

26. NirSoft. DLL Export Viewer v1.66 [Электронный ресурс]. — 2016. — Режим доступа: https://www.nirsoft.net/utils/dll_export_viewer.html (дата обращения: 05.06.2021).

27. Zhou Yi-min, Mu Tong-xin. Research on Dynamic Memory Management Mechanism for Embedded System // 2009 First International Conference on Information Science and Engineering. — 2009. — С. 360–362.

28. Awais Muhammad Abdullah. Memory Management: Challenges and Techniques for traditional Memory Allocation Algorithms in Relation with Today's Real Time Needs // Advances in Computer Science : an International Journal. — 2016. — Т. 5, № 2. — С. 22–27. — Режим доступа: http://www.acsij.org/acsij/article/view/459.

29. Anthony Richard. System Programming - Designing and developing distributed application. — Morgan Kaufmann, 2015. — С. 203 – 276. — ISBN: 012800729X.

30. Fog Agner. — Calling conventionsfor different C++ compilers and operating systems. — Technical University of Denmark, 2021. — Режим доступа: https://www.agner.org/optimize/calling_conventions.pdf (дата обращения: 05.06.2021).

31. Colin Robertson Kent Sharkey Souna Nick Schonning Matthew Sebolt Mike Jones Gordon Hogenson Saisang Cai. Argument Passing and Naming Conventions [Электронный ресурс]. — 2018. — Режим доступа: https://docs.microsoft.com/en-us/cpp/cpp/argument-passing-and-naming-conventions?view=msvc-160&viewFallbackFrom=vs-2019 (дата обращения: 05.06.2021).

32. Unity3D-MatLab Simulator in Real Time for Robotics Applications / Víctor Hugo Andaluz, Fernando A. Chicaiza, Cristian Gallardo и др. // Augmented Reality, Virtual Reality, and Computer Graphics / Под ред. Lucio Tommaso De Paolis, Antonio Mongelli. — Cham : Springer International Publishing, 2016. — С. 246–263.

33. Robots Coordinated Control for Service Tasks in Virtual Reality Environments / Esteban X. Castellanos, Carlos García-Sánchez, Wilson Bl. Llanganate и др. // Augmented Reality, Virtual Reality, and Computer Graphics / Под ред. Lucio Tommaso De Paolis, Patrick Bourdot, Antonio Mongelli. — Cham : Springer International Publishing, 2017. — С. 164–175.

34. Real–Time Virtual Reality Visualizer for Unmanned Aerial Vehicles / Fernando A. Chicaiza, Cristian Gallardo, Christian P. Carvajal и др. // Augmented Reality, Virtual Reality, and Computer Graphics / Под ред. Lucio Tommaso De Paolis, Patrick Bourdot. — Cham : Springer International Publishing, 2018. — С. 479–495.

35. Teaching-Learning of Basic Language of Signs through Didactic Games / Mateo A. Parreño, Carmen J. Celi, Washington X. Quevedo и др. // Proceedings of the 2017 9th International Conference on Education Technology and Computers. — ICETC 2017. — New York, NY, USA : Association for Computing Machinery, 2017. — С. 46–51. — Режим доступа: https://doi.org/10.1145/3175536.3175584.

36. Дума Государственная. Трудовой кодекс Российской Федерации: Кодекс РФ от 30.12.2001 N 197-ФЗ. — 2001. — Дек.

37. СССР Госстандарт. ГОСТ 12.2.033-78 Система стандартов безопасности труда (ССБТ). Рабочее место при выполнении работ стоя. Общие эргономические требования: постановлением Госстандарта СССР от 26.04.1978 N 1100. — 1978. — Апр.

38. СССР Госстандарт. ГОСТ 12.2.032-78 Система стандартов безопасности труда (ССБТ). Рабочее место при выполнении работ сидя. Общие эргономические требования: постановлением Госстандарта СССР от 26.04.1978 N 1102. — 1978. — Апр.

39. России Минсвязи. ТОИ Р-45-084-01 Типовая инструкция по охране труда при работе на персональном компьютере: приказом Минсвязи России от 02.07.2001 N 162. — 2001. — Июль.

40. России Госстандарт. ГОСТ Р 50923-96 Дисплеи. Рабочее место оператора. Общие эргономические требования и требования к производственной среде. Методы измерения: постановлением Госстандарта России от 10.07.1996 N 451. — 1996. — Июль.

41. Росстандарт. ГОСТ 12.0.003-2015 Система стандартов безопасности труда (ССБТ). Опасные и вредные производственные факторы. Классифи-

кация (с Поправкой): приказом Росстандарта от 09.06.2016 N 602-ст. — 2016. — Июнь.

42. России Госкомсанэпиднадзор. СанПиН 2.2.4.548-96 Гигиенические требования к микроклимату производственных помещений: постановлением Госкомсанэпиднадзора России от 01.10.1996 N 21. — 1996. — Окт.

43. Росстандарт. ГОСТ 12.1.003-2014 Система стандартов безопасности труда (ССБТ). Шум. Общие требования безопасности (Переиздание): приказом Росстандарта от 29.12.2014 N 2146-ст. — 2014. — Дек.

44. России Госкомсанэпиднадзор. СН 2.2.4/2.1.8.562-96 Шум на рабочих местах, в помещениях жилых, общественных зданий и на территории жилой застройки. Санитарные нормы: постановлением Госкомсанэпиднадзора России от 31.10.1996 N 36. — 1996. — Окт.

45. Федерации Министерство строительства и жилищно-коммунального хозяйства Российской. СП 52.13330.2016 Естественное и искусственное освещение. Актуализированная редакция СНиП 23-05-95* (с Изменением N 1): приказом Министерства строительства и жилищно-коммунального хозяйства Российской Федерации от 07.11.2016 N 777/пр. — 2016. — Нояб.

46. Росстандарт. ГОСТ 12.1.019-2017 Система стандартов безопасности труда (ССБТ). Электробезопасность. Общие требования и номенклатура видов защиты (с Поправкой): приказом Росстандарта от 07.11.2018 N 941-ст. — 2016. — Нояб.

47. СССР Госстандарт. ГОСТ 12.1.038-82 Система стандартов безопасности труда (ССБТ). Электробезопасность. Предельно допустимые значения напряжений прикосновения и токов (с Изменением N 1): постановлением Госстандарта СССР от 30.07.1982 N 2987. — 1982. — Июль.

48. Росстандарт. ГОСТ Р 12.1.019-2009 Система стандартов безопасности труда (ССБТ). Электробезопасность. Общие требования и номенклатура видов защиты: приказом Росстандарта от 10.12.2009 N 681-ст. — 2009. — Дек.

49. Росстандарт. ГОСТ Р 53692-2009 Ресурсосбережение. Обращение с отходами. Этапы технологического цикла отходов: приказом Росстандарта от 15.12.2009 N 1092-ст. — 2009. — Дек.

50. СССР Госстандарт. ГОСТ 12.3.031-83 Система стандартов безопасности труда (ССБТ). Работы со ртутью. Требования безопасности: постановлением Госстандарта СССР от 10.10.1983 N 4833. — 1983. — Окт.

51. СССР Госстандарт. ГОСТ 12.1.004-91 Система стандартов безопасности труда (ССБТ). Пожарная безопасность. Общие требования (с Изменением N 1): постановлением Госстандарта СССР от 14.06.1991 N 875. — 1991. — Июнь.

52. McLean Schofield Kent Sharkey David Coulter Drew Batchelor Michael Satran. Creating Named Shared Memory [Электронный ресурс]. — 2018. — Май. — Режим доступа: https://docs.microsoft.com/en-us/windows/win32/memory/creating-named-shared-memory (дата обращения: 05.06.2021).

Приложения

A Literature review

(справочное)

Literature review

Студент:

Группа	ФИО	Подпись	Дата
8ИМ91	Гальярдо Паредес Кристиан Маурисио		

Руководитель ВКР

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Доцент	Демин Антон Юрьевич	к.т.н		

Консультант-лингвист отделения иностранных языков ШБИП

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Старший преподаватель	Ануфриева Татьяна Николаевна			

Introduction

PCs (Interprocess Communication) are mechanisms to interconnect processes, threads, applications, or network nodes, and thus be able to share information.

From the beginning, interconnection processes were the main resource used for interconnection between processes, either on the same computer or between several nodes in a network. Most interconnection processes were used to synchronize actions or events, but having the data sharing feature can also be used to share information among clients. In the world of computing, there are specialized applications that, when working in coordination, can solve complex problems belonging to multidisciplinary projects or that require fragmentation of processing, making it necessary to use some interconnection method of processes [2–4, 7]. Among the alternatives for communication between processes we have: clipboard, Component Object Model, Data Copy, Dynamic Data Exchange, Mailslot, Pipes, RPC, and others.

IPC mechanisms exist in all operating systems, but this development focuses on analyzing those used on Windows platforms.

Review of Technical Documentation for IPC Features on Windows Operative System

Using the Clipboard for IPC

It is a space used for the transfer of text, data, files, or objects from a source place to a destination place. Its most accepted and widespread use is found among text editors for the exchange of writings, although it can also be used for the exchange of information between processes that can access the clipboard space.

When a user performs a cut or copy operation in an application, the application places the selected data on the clipboard in one or more standard

or application-defined formats and any other application can retrieve the data from the clipboard, choosing from the available formats, the applications can be on the same computer or remote computers [8].

Using Component Object Model (COM) for IPC

It is a platform created by Microsoft and used for communication between processes. The term COM encompasses many technologies including LE, OLE Automation, ActiveX, COM+, and DCOM.

COM is designed to allow clients to communicate with other objects transparently regardless of where they are running, whether on the same process, the same computer, or a different computer.

DCOM is the extension of the Component Object Model to distributed environments, which defines the connection mechanisms and the network protocol necessary to make calls to remote object-oriented procedures, this feature makes it useful for distributed systems of all kinds based on components.

Application developers must be careful to keep their applications as independent as possible from the network infrastructure but, many distributed applications have to be integrated into the infrastructure of an existing network, which requires a specific network protocol, and this will force adaptation from all customers, which is unacceptable in many situations. With the language independence of DCOM, application developers can choose the tools and programming languages they are most familiar with as DCOM provides this transparency: DCOM can use any transport protocol, such as TCP / IP, UDP, IPX / SPX, and NetBIOS [9].

Using a File Mapping for IPC

It is a portion of virtual memory in which a direct-byte-by-byte mapping is established with a part of a file or similar resource. This resource is usually a file on the hard disk, or a shared memory object, or other types of resource that the operating system can refer to through the file descriptor. Once this mapping between file and memory space is available, applications can manage

access to that resource exactly as if it were primary memory.

File mapping allows a process to treat the contents of a file as if it were a block of memory in the process's address space. The process can use simple pointer operations to browse and modify the contents of the file. When two or more processes access the same file allocation, each process receives a memory pointer in its own address space that it can use to read or modify the contents of the file. Processes must use a synchronization object, such as a semaphore, to avoid data corruption in a multitasking environment [10].

Using a named shared memory for IPC

It is a type of file mapping in which a block of memory can be specified as a data storage file. Other processes can access the same memory block by opening the same file allocation object, this presents a considerable advantage compared to when working with physical files since the file projected in memory has much faster response times because all its information is stored in operative memory.

Files projected into memory load a full page into memory at a time, and the size of the page is determined by the operating system for optimal performance [10].

Using a Mailslot for IPC

A mailslot is a pseudo-file that resides in memory and offers a one-way communication mechanism between processes (IPC) and the processes can be both local and remote. Conceptually, they are similar to Windows named pipes and POSIX message queues, but their use is much simpler especially when handling relatively small numbers of short messages. The data from a message to a mailslot can be of any format but the maximum size of the messages that can be sent is 424bytes.

The mailslots work on a client-server model where the mailslot server is the process that can create a mailslot and therefore is the one who can read the messages. Only the server that created the mailslot, or the processes that inherit it, can read the messages. One feature to take into account is that a

server can only create mailslots locally, it cannot create a remote mailslot.

The mailslot client is a process that can write messages to a mailslot by name. The mechanism allows sending short broadcast messages to be listened to by all the computers that share the same network domain [11].

Using Pipes for IPC

In computing, a pipeline (pipeline or channel) consists of a chain of processes connected in such a way that the output of each element of the chain is the input of the next. They allow communication and synchronization between processes and the use of data buffers between consecutive elements are common.

Communication through pipes is based on the producer/consumer interaction, the producer processes (those that send data) communicate with the consumer processes (that receive data) following a FIFO order. Once the consuming process receives data, it is removed from the pipeline.

Unnamed pipes have a file associated with them in main memory, therefore, they are temporary and are removed when they are not being used by producers or consumers. They allow communication between the process that creates a channel and child processes after the creation of the pipeline.

Named pipes create the cause in the file system, and therefore are not temporary. They are handled by system calls (open, close, read and write) like the rest of the system files. They allow communication between the processes that use said pipeline, although there is no hierarchical connection between them [12].

Using Windows Sockets for IPC

Socket designates an abstract concept by which two processes (possibly located on different computers) can exchange any data stream, generally in a reliable and orderly manner. The term socket is also used as the name of an application programming interface (API) for the TCP / IP family of Internet protocols, usually provided by the operating system. Internet sockets are the mechanism for delivering data packets from the network card to the appropriate processes

or threads. A socket is defined by a pair of local and remote IP addresses, a transport protocol, and a pair of local and remote port numbers.

Windows Sockets are based on the sockets first popularized by Berkeley Software Distribution (BSD). An application that uses Windows Sockets can communicate with other socket implementation on other types of systems. However, not all transport service providers support all available options [13].

Use of named shared memory for IPCs

Named shared memory vs Others IPCs

When analyzing the speed of operation of the IPCs we must bear in mind that most of the IPCs use complex encryption functions, data queuing, or advanced architectures such as network protocols that make use of the OSI stack. These complex architectures require processing time, to this is added that many IPCs work on the hard disk which is a very slow peripheral compared to the speed of the computer's memory. If the transfer speeds of hard drives are compared, we have the following results:

- Traditional mechanical hard disk handles transfer speeds between 40MB/s to 90MB/s.
- A traditional solid state disk can reach about 500MB/s, the fastest solid state disks M2 or NVMe technology can reach 3000MB/s or 3GB/s.

The transmission speed of the ram memories, for example a DDR4-3200 handles speeds of 25.6GB/s and can perform 3200MT/s.

MT/s (Megatransfer per second) is a measure of transfers per second, 1 MT/s is 10^6 or one million transfers per second.

As a result, using memory for information transfer is the fastest possible option as long as data queuing features or complex data integrity controls are not required [19–22].

Features and process of creating named shared memories

File mapping can be used to share a file or memory between two or more processes. To share a file or memory, all of the processes must use the name or the handle of the same file mapping object.

A file mapping object must be created by calling the CreateFileMapping function to which several parameters must be assigned for example:

- The invalid identifier value.
- A name that will correspond to the object.
- The permissions, depending on the need of the programmer, for example, ReadWrite offers to any process read and write permissions in memory through any file view that is created.

There are two possible scenarios to access the shared memories created.

The first possibility is when it is required to access the memory from another thread or function within the same process that created the shared memory. In this case, it is accessed through the use of views of the file in the address space of the process, for this the MapViewOfFile function is used. The MapViewOfFile function returns a pointer to the file's view, after which, we can interact with the shared memory.

The second possibility is when it is required to access the shared memory from a different process or application than the one that created the shared memory. To access the shared memory it is necessary to first open the memory using the OpenFileMapping function and specifying the name of the mapping object (the name of the existing shared memory), then a view to the shared memory is created using the MapViewOfFile function to obtain a pointer to the file view.

Finally, when the process no longer needs to access the file allocation object, it must call the CloseHandle function. When all handles are closed, the

system can release the section of the paging file that the object uses [52].

It should be noted that shared memory can contain multiple storage spaces and each storage space can contain a different type of data. This work implements support for the Integer, Float, Double, or character string data types.

Depending on the type of data to be stored within a shared memory its structure can be understood as two-dimensional if it uses only numerical data or three-dimensional if strings of characters are also stored (figure 3.3).

Review conclusions

Once the characteristics of the different IPC options for Windows operating systems have been reviewed, it allows concluding that the fastest option for exchanging information is the use of named shared memories. This option is not widely used since it does not offer data queuing characteristics or data integrity characteristics but, for the scenario in which it is implemented, it does not require these characteristics, therefore, it is a viable option that offers access to the data in real-time.