Министерство науки и образования
Российской Фежерации
Институт математики, экономики и информатики
ФБУН «Иркутский государственный университет»

Черкашин Е. А.

Как спроектировать текст выпускной квалификационной работы

учебное пособие

Иркутск—2017

Оглавление

Введение **3**

1. Логическая структура ВКР **4**
1.1. Первый этап – заключение 4
1.2. Второй этап — Введение 5
1.3. Собственные эксперименты
Реализация программной системы 8

2. Оформление текста **19**
2.1. Текстовые объекты 19

3. Формальные части **22**
3.1. Список литературы 22

Литература **23**

Введение

Что-то о логической структуре ВКР, а также о логических связях в структуре информации.

Структура ВКР, в целом, соответствуе структуре жизненного цикла программного обеспечения (ЖЦ ПО) [1].

1. Логическая структура ВКР

Текст ВКР представляет собой отчет о проделанной работе с дополнительными комментариями. Для того, чтобы заполнить ВКР текстом необходимо знать, прежде всего, что во время дипломной практики было сделано. И тогда самое простое – это написать раздел «Заключение».

1.1. Первый этап – заключение

Заключение представляет собой несколько абзацев текста, подобного следующему

> В данной выпускной квалификационной работе/дипломной работе/диссертации/статье рассмотрена задача…Для этого решены следующие задачи:
>
> 1. Исследована предметная область …;
> 2. Выделен набор функций, которые необходимо было реализовать;
> 3. Спроектирована архитектура [программной системы];
> 4. При помощи … метода разработана информационная модель …;
> 5. Произведена реализация [программной системы] на языке программирования …;
> 6. Проведено тестирование ….

Далее делаем вывод о том, достигнута ли цель ВКР или нет и помещаем его после списка задач.

В заключении так же имеет смысл изложить в одном абзаце критические замечания по полученному результату: какие требования удалось реализовать, а какие нет; оценить качество полученного результата; показать каким образом можно удовлетворить требования и улучшить качество.

Кроме того, один абзац, последний, можно посвятить направлениям дальнейшего продвижения в разработке: что еще можно улучшить, какие функции добавить к программной системе, а какие убрать; оценить возможности программного продукта как инструмента решения более широкого класса задач.

Что можно улучшить в тексте заключения Откомментировать каждую задачу, только очень кратко. Например, в п. 3 на стр 4, который посвящен архитектуре, можно указать ИЛИ название архитектуры (например, клиент–сервер), ИЛИ добавить фразу о том, как было получено решение (например, «В результате анализа организационной структуры предприятия спроектирована архитектура …»).

Теперь имеем текст заключения, который уже практически готов, к нему мы вернемся уже в самом конце для проверки его непротиворечивости основному тексту и, возможно, дополнению еще парой–тройкой фраз.

1.2. Второй этап — Введение

Введение представляет собой сокращенный вариант основного текста с добавлением общей информации. Поэтому писать введение удобно [для мозга] с середины его текста: копируем первый, третий и второй абзацы заключения (задача ВКР, список задач и вывод о достижимости цели).

Сначала изменим список задач. Для этого выкинем из предложений все комментарии.

> …Для этого решены следующие задачи:
>
> 1. Исследована предметная область …;
> 2. Выделен набор функций, которые необходимо было реализовать;
> 3. Спроектирована архитектура [программной системы];
> 4. ~~При помощи … метода~~ Разработана информационная модель …;
> 5. Произведена реализация [программной системы] ~~на языке программирования …~~;
> 6. Проведено тестирование ….

Обращаю внимание, что текст списка задач представлен в том же времени и форме, что и в заключении, так как ВКР — это отчет о проделанной работе, а не план работ на будущее.

На следующем этапе формулируется *цель*[1] ВКР – это либо проведение исследования (научная цель), разработка программы (инженерная).

[1] предсказуемый результат некоторой деятельности

Научные исследования ведутся с целью обогащения наших знаний новыми знаниями об исследуемом *объекте* и его *свойствах*. Научные исследования делятся на две категории – *финдаментальные* и *прикладные*. Фундаметальные исследования направлены на решения принципиальных вопросов, например, есть ли жизнь на Марсе, или $P = NP$. Признаком фундаментальной проблемы является ее постановка в виде вопроса, на который должен быть получен ответ – “Да” или “Нет”. Известна метафора о прикладных исследованиях: “Из лука стреляем в стену и вокруг места попадания рисуем мишень”.

Прикладные исследования направлены на выработку подходдов, методов и методик решения конкретных технических задач. Примерами выступают все задачи Теории изобретательства и рационализации (ТРИЗ) [**?**].

/**метод, методика подход…/**

Таким образом исследовательская цель записывается в следующем виде: “Целью [ВКР] является исследование [такого-то] объекта и поиск в нем [таких-то] свойств/зависимостей”.

Разработка программ направлена на автоматизацию обработки информации, что на самом деле значит, что происходит процесс частичной реализации функций (задач, последовательности действий), выполняемых вручную, при помощи методов обработки информации. Таким образом цель программистской ВКР записывается как фраза о проектировании и разработке программы или программной системы, автоматизмрующей какие-либо рутинные процессы.

Записываем цель ВКР перед списком задач. Следующий этап – перечисление требований к используемым методам исследования и технологиям. Требования возникают как ограничения, накладываемые на любую деятельность. Например, зарабатывать состояние необходимо в условиях соблюдения законодательства страны, или, если программа разрабатывается под заказ, то всегда эту программу необходимо интегрировать с существующими у заказчика программными системами и форматами данных. Вот эти ограничения с краткими комментариями необходимо перечислить сразу после списка задач.

Пример списка требований для программной системы

К программной системе предъявлены следущие требования:

- База данных должна быть реализована при помощи Microsoft SQL Server версии 2012;
- Обеспечить обмен данными с существующими программными системами при помощи технологии RESTful;

- Интерфейс пользователя реализовать в виде Интернет-приложения;
- Обеспечить доступ к приложению из любой точки мира.

Перечисленные требования отражают сетевую инфраструктуру предприятия–заказчика.

Для исследования, например, список выглядит следующим образом:

В процессе исследования рекомендован к использованию следующий перечень подходов и методов:

- Использовать численные методы для решения линейных дифференциальных уравнений, реализуемых на вычислительных кластерах;
- …

[комментарий, объясняющий]

Если объем ВКР предполагается большим, то принято помещать пару абзацев, излагающих общую структуру ВКР, в коце введения.

Квалификационная работа состоит из Главый 1, где в п. 1.1 приводятся определения теории …, в п. 1.2 представлен литературный обзор, и приводятся основные свойства [объекта исследования]. В п. 1.3 представлен предмет исследования в виде гипотезы [о наличии в системе интересующего нас свойства]. … В главе 2 представлены собственные эксперименты, а именно в п. 2.1. …Третья глава посвзена вопросам реализации разработанного аодхода на ЭВМ (в виде программной системы). …

Про введение в структуре ничего писать не надо, так как оно уже прочитано. А мы ценим время читателя, которым также является и ваш рецензент.

Если к этому моменту или в процессе написания разделов появиляются новые мысли, то желательно их записывать на бумажки. Как только вы заканчиваете мысль, абзац или раздел, надо быстро перемещаться в тот раздел, куда записанные мысли должны быть помещены, ну и, собственно, записать их туда в содержательном смысле. Форме изложения не надо много внимания уделять, так как основная задача – схватить мысль (grasp) и зафиксировать ее[1]. В результате, например, во введени появится пара абзацев, касающихся какому-либо аспекту разработки, но носящие достаточно общий вид.

[1] Этот абзац как раз появился благодаря такой мысли и реализации этого принципа

1.2.1. Объект и предмет исследования

К этому моменту у вас в мозгу уже долно появиться некоторое понимание того, с каким объектом исследования или автоматизации вы имеете дело. *Объект исследования* – часть предметной области, на который или на часть которого направлена ваше деятельность. Например, в задаче определения наличие жизни на Марсе объектом исследования является Марс; в задаче автоматизации бухгалтерской деятельности – автоматизируемое предприятие. Как правило, в качестве объекта исследования/автоматизации выступает физический или идеальный обхект, физический или идеальный процесс. Пример объекта – проесса – это автоматизация химического производства без привязки к конкретному предприятию.

Предмет исследования – часть объекта исследования, модель которой строится[1]. В задача с Марсом предмет исследования – наличие различных признаков жизни, например, аминокислот в почве, наличие воды и т.д., т.е. необходимых условий существования жизни. В задаче бухгалтерского учета предмет автоматизации – учет материальных и денежных средств предприятия.

Объект и предмет исследования/автоматизации представляются в тексте одним абзацем каждый и помещаются последовательно где-по перед целью.

На этом с разделом введения можно на время закончить и перейти к разделу, текст которого вам написать проще всего. Обычно – это то, как программа пелизована, какие именно исследования были проведеный и какие результаты былиполучены, т.е. это то, что есть уже на самом деле.

1.3. Собственные эксперименты Реализация программной системы

Проще всего данную **главу**[2] начать с описания результата: какие свойства объекта и предмета были получены, какая программная систма реализована, какие общие выводы можно сделать из результатов.

[Описание РЕЗУЛЬТАТА исследований]

Что можно сказать о программе? Да много чего. Общая мысль – прорекламировать пользователям свою программу. Пишите текст, отвечая на следующие вопросы:

1. Для кого предназначена программа? Каким (конкретный класс) физическим или юридическим лицам использование вашей программ поз-

[1] Программа – это реализациямодели вычислительного процесса, процесса обработки информации, что тоже модель.

[2]

воляет избавляться от рутинных операций, экономить материальные средства, зарабатывать больше денег, совершенствовать организационную структуру и т.п.

2. В чем состоит основная автоматизируемая функция? Сначала дается определение. Затем кратко перечисляются этапы автоматизируемого процесса, снабженные краткими комментариями о том, как эти этапы реализованы в программе.

3. Какие особенности у вашей программы по сравнению с программами–аналогами? Здесь имеет смысл выделить одну-три выдающейся характеристик, которые значительно лучше, чем у остальных[1].

4. …

Полученный текст с перечислением достоинств программы (результатов исследований) помещаем в **конец главы ???**.

Общая идея представления результата – программной системы – состоит в том, чтобы представлять материал от общего к частному. Т.е. по тексту главы сначала описываются модели программы, например, Функциональная модель, архитектура, ER-диаграмма (модель Чена), UML-модель классов (Диаграмма классов), …, затем переходим к программному коду компонент, и потом уже к результатам тестирования, и т.д. В принципе, такой порядок отражает жизненный цикл программного обеспечения [**?**]: дизайн, проектирование, реализация, тестирование, ввод в эксплуатацию, сопровождение и снятие с эксплуатации. Все эти разделы могут быть представлены в ВКР.

Общая стратегия изложения текста ВКР в главе, посвященной реализации, – от общего к частному. Сначала излагаются общие вопросы, как вышеописанный перечень характристик, затем переходим к описанию моделей программы, затем к программному коду, тестированию, развертыванию и т.д. Такой стиль изложения удобен рецензенту: рецензент является специалистом в предметной области, основных методах решения проблем, он также знаком с информаионными технологиями. Чтобы оценить вас как специалиста, он должен понять задачу и то, как вы справились с ее решением. Построение текста от общего к частному позволяет при чтении постепенно детализировать внимание рецензента. Рецензент встраивает ваш ВКР в свою систему знаний. А это удобно делать именно в этом порядке.

В начале главы надо представить читателю общую концепцию проекта вашей разработки – в двух словах что и как вы собираетесь делать. Например, «Программная система представляет собой АРМ на основе реляционной базы данных, при этом, взаимодействие строится на основе клиент-

[1] У гиков этот раздел называется features

серверных технологий.» Из примера сразу становится видно, что собой представляет программный комплекс и каким классом программных технологий будет дальнейшее изложение ограничено.

В общих вопросах рассматривается также концепция взаимодействия пользователя с программной системой, и то, какую стратегию вы выбираете для удовлетворения требований.

> Интерфейс пользователя АРМа представляет собой динамическую WEB-страницу. Это позволяет решить проблему зависимости программного обеспечения от операционной системы. Взамодействие киентской (веб-браузер) и серверной части реализовано на основе двустороннего обмена информацией, что позволяет обеспечить обмен сообщениями между пользователями в решиме, близком к реальному времени. Для обеспечения доступа к серверной части по проьоколу RESTful предусмотрен специальный модуль к WEB-серверу. …

Стиль фраз В примере появилась важная особенность стиля изложения текста – *доказательный*. Важно не просто описать вашу программу – ее возможности и то, как она устроена – но и обосновать **каждое** вами принятое **решение**. Чтобы проверить, удалось ли выдржать такой стиль достаточно к каждой фразе, где излагается решение задать вопросы «Почему это так?» и «Зачем это нужно?». Если в тексте есть фраза отвечающая на один из этих вопросов, щначит все хорошо[1].

1.3.1. Место программы в организационной структуре

Каждая программа где-то и кем-то используется для решения какой-то задачи. В рамках ВКР чаще всего в качестве задачи рассматривается автоматизция обработки информации в учреждении или предприятии. Согласно системному подходу [**?**] получается, что программа является функциональным блоком некторый организационной структуры или системы бизнес-процессов [**?**]. Вот эту структуру и бизнес-процессы необходимо описать сразу после раздела общих вопросов.

1.3.1.1. Организационная структура

Оргмнизационная структура современных учреждений представляет собой, как правило, древовидный граф иерархического подчинения, либо

[1] Вообще чаще всего задается первый вопрос. Вопрос «Зачем?» надо задавать, чтобы оценивать уровень рационализма принятых решений.

коммитетную матрицу. Чтобы представить предприятие с иерархическим подчинением в ВКР надо нарисовать граф, отражающий а) верхний уровень подчинения, б) места отделов, где находится в эксплуатации ваша программа. На диаграмме можно учесть также элементы организаионной структуры связанные с вашей программой коссвенно, например, в этих отделах готовят исходные данные для обработки или анализуруют отчеты. то есть диаграмма должна отражать ту часть организационной структуры, которая связана с функционированием вашей программы.

Комитетную организационную структуру, которая встречается «в природе» несколько реже, удобно отображать в виде матрицы. По горизонтали располагаются, например, виды деятельности, а по вертикали – специалисты или отделы. Комитетная тсруктура удобна, если предприятие занимается несколькими проектами одновременно. Каждый проект в матрице отражается при помощи выделенных цветами клеточек. На самом деле чисто комитетной сруктуры не бывает, всегда есть лица, ответственные за функционирование всего предприятия/учреждения. Внутри проктов сотрудники таже выстраиваются в иерархии подчинения и ответственности. …В матрице выделяются проекты и клетки, в которых используется разработанная программа.

Полученная диаграмма сопровождаетя текстом пояснения, … В тексте в общем виде указывается, какие функции выполняет (решает задачи) программа в каждом выделенном блоке диаграммы, а также информация какого вида там преобразуется. При чтении текста к текущему моменту становится понятным, зачем и кому нужна программа. А вы в результате вашей творческой деятельности получили перечень функций, это перечень – первая абстракная модель вашей программы – *функциональная модель*.

1.3.1.2. Функциональная модель

Функциональная модель представляет программую систему как механизм выполняющий (исполняющий) некоторый набор функций. Модель показывает **что** делает программа в более формальном виде, чем просто перечень свойств (фич). Модель может учитывать связи между входными и выходными данными функций или нет. Можно объединять функции в классы и функциональные блоки. Функциональное моделирование выполняется при помощи различных *нотаций*[1].

Перечень функций – самцый простой метод моделирования, где функции

[1] Нотациями в программировании называют методы моделирования, включающие язык представления элементов моделей, правила построениая корректных моделей, семантику и интерпретацию лементов моделей и композиций.

просто перечисляются одна за одной, например в виде обощенной таблицы спецификаций.

IDEF0 на основе SADT. IDEF0 – это нотация иерархического представления организационной структуры сложного объекта (пердприятия, прибора и т.п.), преобразующего некоторый «вход» в «выход» при помощи «механизма» в условиях некоторых «ограничений» и под воздействием «управления» (см. рис. **??**). Популярной методикой анализа сложного объекта и построения иерархии IDEF0-диаграмм является *SADT* (Structural Analysis and Design Technique), «методика структурного анализа и дизайна».

Диаграмма UML UseCase используется для моделирования программного обеспечения как граф, отражающий взаимоотношения между функциями системы и агентами, которые пользуются этими функциями. В диаграмме можно представлять наследование свойств объектов и декомпозицию функций (рис. **??**).

ARIS Европейский проект по расширению нотации IDEF0 в направлении поддержки состояний (а не только структур). Методики проектирования, основанные на ARIS, применяются, например, в SAP/R3-системах авыматизации производства.

BPMN2.0 Нотация называется Bussiness Process Modelling Notation, нотация для моедлирования бизнес-процессов. В настоящее время едва ли не самый выразительный подход к моделированию. Позволяет представлять как декомпозицию структуры автоматизируемого объекта, агентов, документы, потоки информации, события и состояния, но также и альтернативные пути ралиизации функций, задание и обработку исключений, порождение подпроцессов и т.д. Существуют даже системы интерпретации таких моделей в виде приложений.

В данном пособии спользуем классический подход к функциональному моделированию – методику SADT. Согласно этой методике автоматизация – это процесс преобразования организационной структуры предпроятия. Это процесс называется *реинжениринг бизнес-процессов*. Модель IDEF0 представляет собой иерархию блоков, каэдый из которых представляет процесс (функцию преобразования входа нв выход). Реинжениринг состоит из двух этапов: анализ структуры предприятия и выявление тех его процессов, автоматизация которых позволит улучшить качественные (упрощение организационной структуры) или количественные характеристики (производительность труда). В результате первого этапа создается модель

(иерархический набор диаграмм), называемая «As Is», т.е. «Как есть сейчас». Второй этап – преобразование результата предыдущего этапа в модель «To Be». Декомпозиция проводится при помощи программ BpWin [**?**] или /**как ее там, которая на яве ;-)[?]…**/.

При построении модели «As is» сначала выбирается сотрудник организации, с точки зрения которого будет производится декомпозиция бизнес-процессов. Таким сотрудником может быть директор, бухгалер, прораб и др. Далее рассматривется овновная функция предприятия, что она входе этой функции (деньги, люди, документы, заказы), что на выходе (деньги, люди, документы, отчеты), какиме механизмы реализуют эту функцию (сотрудники, приборы, программное обеспечение), какие ограничения накладываются на реализацию функции (законодательство Российской Федерации, стандарты отрасли). Далее производится рекурсивная декомпозиция основной функции предприятия до уровня, где автоматизируемая вами функция представляет собой блок со входом, выходом, механизмами и ограничениями. В результате выполнения декомпозиции вы получаете

- модель предприятия с различными уровнями абстракции представления ее деятельности;
- понимание того, какие объекты на входе и какие на выходе, т.е. информацию об интерфейсах вашей программы;
- перечень агентов (сотрудников), которые будут работать с вашей программной системой;
- перечень программ, вовлеченных в реализацию функции;
- информацию об источниках стандартизованных процедур, используемых в реализации функций программы.

Согласитесь, стоит потратить один день – провести декомпозицию – для того, чтобы всю эту информацию получить.

Модель «To be» – это декомпозиция автоматизируемого блока на один или два дополнительных уровня с учетом реализации автоматизации. Для построения этой модели может быть придется декомпозировать блок в модели «As is», чтобы выявить автоматизируемые функции и информационные связи между ними.

Теперь остается все это оформить и описать в виде текста. Файл с декомпозицией поместить на сервер (Dropbox или Github), чтобы специалисты могли получить доступ к вашим результатам, а в текст ВКР обучно помещвется в виде рисунка верхний уровень (функция организации) «As

is», модель A0, уровень, на корором выявлена автоматизируемая функция, а также декомпозиции «As is» и «To be» этой функции. К каждому рисунку пишете по одному абзацу комментариев, где кратко описывается бизнес процесс, например, как процесс преобразования входа (в основной блок) разными функциями в разные выходы, и, в конечном счете, в выход основного блока. Затем комментируете использование механизмов и ограничений. Интересные конструкции, например, если выход одного блока превращается в управление/ограницение другого, или вообще в механизм, то этот процесс хорошо рассмотреть отдельно и подробно.

1.3.1.3. Диаграмма прецедентов

На основе полученной IDEF0- фуннкциональной модели уже можно очень быстро построить UML диаграмму прецедентов. Что она модет содержательно[1] дополнить – так это указать как агенты, процессы и ресурсы наследуют свойства между сосбой. Например, в нотариальной конторе сам нотариус может выполнять функции секретаря, т.е. готовить тексты документов. Поэтому нотариус – это частный случай секретаря, наследует все его свойства. Этот факт фиксируется стрелкой с пуствм треуголником на конце (рис. **??**). Аналогично можно классифицировать ресурсы.

С диаграммой прецендентов долго возиться не надо. Просто рисуем агентов, ресурсы, фукции, ставми стрелки. Надо только внимательно отнестись к функциям – их следует организовать в цепочки или деревья, выделяя общие части в отдельный эллипс. Задача Use Case – диаграммы – классифицировать повторяющиеся функции разных бизнес-процессов, выявить общие части, назначить роли агентам.

После рисунка диаграммы пишем комментирующий абзац, где в вербальной форме, например, описываются роли агентов (пользователей).

1.3.2. Архитектура программной системы

Итак, вы описали функциональные модели вашей программной системы, теперь понято **что делает ваша система**, и это организационный аспект моделирования. Теперь надо переходить к тому, как она устроена, т.е. к инженерно-организационному аспекту моделирования.

На Википедии дано следующее определение архитектуре [**?**] *Архитектура программного обеспечения* (англ. *software architecture*) – совокупность важнейших решений об организации программной системы. Архитектура включает:

[1] Добавить конкретной новой (неповторяющеся) осмысленной информации.

- выбор структурных элементов и их интерфейсов, при помощи которых сконструирована система, а также задание их поведения в рамках сотрудничества друг с другом;
- соединение выбранных элементов структуры и поведения во более крупные системы;
- архитектурный стиль, который задает принципиально всю организацию программной системы – все элементы, их интерфейсы и их соединение и взаимодействие.

Документирование архитектуры программного обеспечения упрощает процесс коммуникации между разработчиками, позволяет зафиксировать принятые проектные решения и предоставить информацию о них эксплуатационному персоналу системы, повторно использовать компоненты и шаблоны проекта в других задачах.

Общепринятого определения «архитектуры программного обеспечения» не существует. Так, сайт Software Engineering Institute приводит более 150 определений этого понятия.

На каком-то этапе разработки вашей программной системы вы приняли концептуальное решение, например, реализовать базу данных на сервере PostgreSQL. Что это означает? Это означает, что взаимодествие ядра системы с БД будет вестись по правилам «Клиент-сервер», где в качестве клиента выступает ядро, а сервера – сервер БД PostgreSQL. Ядро с целью получения кортежей данных делает запросы к серверу, а он, в свою очередь, реализует загрузку кортежей с дисковой памяти, их фильтрацию, а ткуже другие преобразования кортежей, заканчивающиеся пересылкой данных ядру. В качестве сервера и клиента, например, выступают веб-вервер Apache и Google Chrome, соответственно.

В тексте ВКР под описание архитектуры выделяется раздел в главе «Реализация …», либо в конце главы «Теоретические основы …». Сначала идет абзац с общей идеей, вложенной в архитектуру, например, что она «клиент–серверная». Затем следует рисунок архитектуры. На рисунке при помощи прямоугольников и стрелок рисуется система взаимодействия между функциональными блоками, агентами и ресурсами.

Клиент и сервер взаимодействуют друг с другом на основе некоторых заданных разработчиком правил. Эти правила называются *протоколом*. Нарисовав архитектуру, описываем ее функциональные блоки и протоколы взаимодействия.

Предположим, что мы разрабатываем транслятор языка управления каким-нибудь АРМом, использующем базу данных, хранящююся на сервере.

> …Архитектура системы состоит из … функциональных блоков. Пользователь при помощи «Интерфейса пользователя» (функциональный блок) вводит запросы в систему. Запросы в виде строки (протокол) передаются на вход [функционального] блока «Лексический анализатор», где входная строка преобразуется в поток лексем с соответствующими семантическими атрибутами. Лексемы подаются на вход Синтаксического анализатора (можно писать и не с большой буквы), который создает дерево синтаксического разбора, интерпретируемое в …

Затем пишем, что там дальше происходит, как осуществляется запрос к БД: какие блоки в этом процессе участвую, какие данные передают и в какой форме. Отслеживаем цепочку обратно. Читателю ведь интересно, как результат запроса к БД преобразуется и выдается пользователю.

Далее добавляем пару предложений, обосновывающих эту модель, что она то, что нужно и придумать что-то лучше можно, но не особо нужно (ничего особенного новыя модель не добавит).

1.3.3. Обоснование выбора технологий реализации

Следующим решением, от которого зависит то, как реализована программная система, является выбор программных технологий, при помощи которых все функциональные блоки создаются и обеспечивается их взаимодействие. В качестве информационных технологий выступают:

- аппаратное обеспечение рабочих станций, серверов, сетей, маршрутизаторов и т.д.;
- операционные системы;
- прошивки ПЗУ устройств;
- структуры данных – форматы данных, теоретико-множественные представления структур данных программы (графы, категории);
- библиотеки и готовые системы (frameworks);
- Сервера баз данных, сторонние программы и их интерфейсы, а также отдельные функциональные блоки и динамические библиотеки;
- программные платформы (Microsoft, Oracle, Linux/Apache/Mysql/Perl);
- языки программирования;

- трансляторы – компиляторы, интерпретаторы, виртуальные машины;
- инструментальные средства разработки (IDE).

Все перечиленное является кирпичами и инструмантами, при помощи которых создается программная система, реализуется ее архитекттура и функции в виде конкретного приложения.

В начале раздела испольуемые технологии просто можно перечислить с небольшими комментариями.

> Программная система реализована при помощи следующих информационных технологий:
>
> 1. база данных создана на основе сервера PostgreSQL версии 9.1;
> 2. …
> 3. интерфейс пользователя реализован в HTML с использованием фреймворка /**AngularJS…/**;
> 4. сервер приложения разработан на основе платформы Node.js и библиотеки ExpressJS для реализации интерфейса и передачи данных по протоколу RESTFul на сервер;
> 5. …

Добавляем абзац с общим обоснованием и самокритикой этого совокупного набоора технологий. Сопоставляем с задачами и требованиями из раздела ВКР «Введение». Только подробно по пунктам это делать не надо, иначе читатель подумает, что издеваетесь над ним. Достаточно в общем охарактеризовать, обратив внимание на что-то действительно важное, например на основную задачу и самое «занятное и замысловатое» требование.

Далее в разделе под подзаголовками третьего уровня просто описываем основные функции, архитектурные особенности, протоколы, инструментарий каждой (или основных) из перечисленных технологий, обращая внимание в тексте на то как замечательно данная технология будет решать ваши задачи и удовлетворять требования.

1.3.4. Информационная модель

Н. Вирт дал такое опредеоение программе:

> Алгоритм + Структуры данных = Программа.

При чтении чужой программы сначала удобно изучить ее структуры данных, а потом уже алгоритмы и их реализации. Думаю не ошибусь, если скажу, что именно структуры данных определяют общий вид программы. Поэтому качество моделирования структур данных значительно влияют на качество проекта.

2. Оформление текста

Структура ВКР состоит из основных и дополнительных разделов. К основным разделам относятся:

1. Титульный лист, на котором представлена информация формального характера:
    - вуз, в котором выполнена работа,
    - /**кафедра…**/,
    - название ВКР,
    - автор и научный руководитель;
2. Введение, где изл....;
3. Глава, посвященная изложению теоретических аспектов работы;
4. Глава, где рассмотрены аспеты реализации программного продукта;
5. Заключение, в котором кратко перечисляются полученные результаты, /**самокритика…**/ и /**направления дальнейшего совершенствования…**/.
6. Список используемых источников (список литературы).

2.1. Текстовые объекты

2.1.1. Формулы

Пример правильно оформленной формулы:

$$\vec{F} = -G\frac{Mm}{|r^3|}\vec{r}, \tag{2.1}$$

где $\vec{F}$ – вектор силы тяжести, действующий на материальную точку массой m со стороны тела (материальной точки) массой M, находящегося в начале координат. Радиус-вектор $\vec{r}$ направлен из начала координат в цетр другого тела. Уравнение Ньютона (2.1) в скалярной форме выглядит в следующем виде:

$$F = G\frac{Mm}{r^2}, \quad r = |\vec{r}|, F = |\vec{F}|.$$

2.1.2. Рисунки

Рисунки в тексте отображаются в разрыве текста, по центру с нумерацией и подписью.

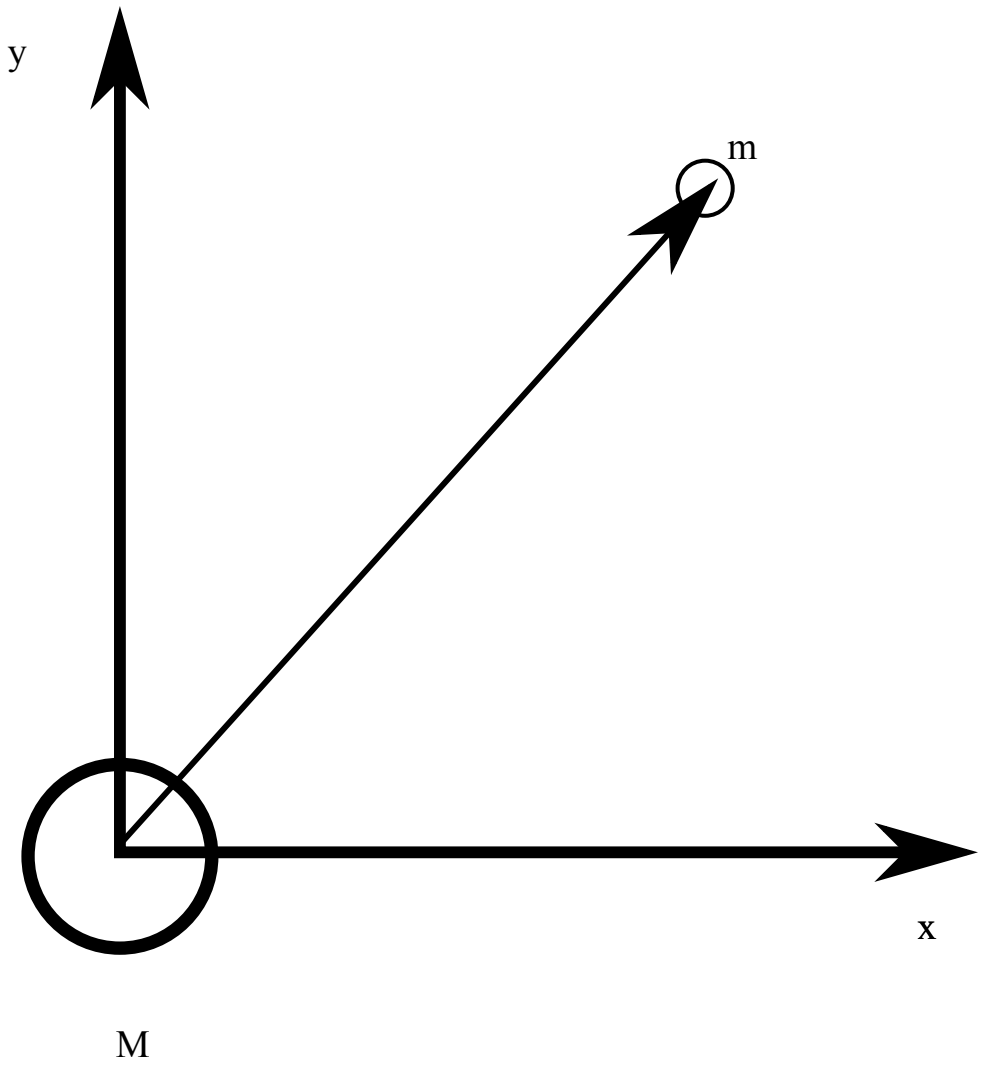

Рис. 2.1. *Ньютоновская система двух тел*

2.1.3. Алгоритмы

2.1.4. Тексты программ

```
/* HelloWorld.java
 */

public class HelloWorld
{
        public static void main(String[] args) {
                System.out.println("Hello World!");
```

```
        }
}

// Hello3.cs
// arguments: A B C D
using System;

public class Hello3
{
   public static void Main(string[] args)
   {
      Console.WriteLine("Hello, World!");
      Console.WriteLine("You entered the following {0} command line
  ↪  arguments:",
         args.Length );
      for (int i=0; i < args.Length; i++)
      {
         Console.WriteLine("{0}", args[i]);
      }
   }
}
```

3. Формальные части

3.1. Список литературы

Литература

1. Жизненный цикл программного обеспечения — Википедия. [электронный ресурс] URL: https://ru.wikipedia.org/wiki/%D0%96%D0%B8%D0%B7%D0%BD%D0%B5%D0%BD%D0%BD%D1%8B%D0%B9_%D1%86%D0%B8%D0%BA%D0%BB_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%BD%D0%BE%D0%B3%D0%BE_%D0%BE%D0%B1%D0%B5%D1%81%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%B8%D1%8F (дата обращения: 28.04.2016)