

АННОТАЦИЯ

Отчет 100 стр., 35 рис., 9 ист., 2 прил. ИМИТАЦИОННОЕ МОДЕЛИРОВАНИЕ, ПРОИГРЫВАНИЕ ИМИТАЦИОННОЙ МОДЕЛИ, СОСТОЯНИЕ ИМИТАЦИОННОЙ МОДЕЛИ.

Объект разработки — программный продукт, подсистема проигрывания имитационных моделей в системе имитационного моделирования Rao X.

Цель работы — создание инструмента позволяющего проигрывать состояния имитационной модели. Интеграция в систему Rao X.

При разработке системы проведены исследования, целью которых было изучить механизмы манипуляции временем.

Результат разработки — интерактивная подсистема проигрывания имитационных моделей, представляющая из себя пользовательский интерфейс, позволяющий проигрывать состояния модели, по средствам использования графических элементов интерфейса.

Эффективность системы заключается в интеграции в систему Rao X с возможностью проигрывания состояний модели в произвольном порядке, с изменением скорости проигрывания. Данная система может быть использована как прототип и служить базой для дальнейших разработок.

Оглавление

ПЕРЕЧЕНЬ СОКРАЩЕНИЙ	10
ВВЕДЕНИЕ	11
1. Техническое задание	12
1.1. Основания для разработки	12
1.2. Назначение разработки.....	12
1.3. Требования к программе или программному изделию	12
1.3.1. Требования к функциональным характеристикам	12
1.3.6. Требования к маркировке и упаковке	14
1.3.7. Требования к транспортированию и хранению.....	14
1.3.8. Требования к программной документации	14
1.4. Техничко-экономические показатели.....	14
1.4.1. Стадии и этапы разработки	14
1.4.2. Порядок контроля и приемки	14
1.4.3. Приложение.....	15
2. Предпроектное исследование.....	16
2.1. Введение	16
2.2. Эволюция сложных систем.....	17
2.3. Подходы к построению имитационной модели.....	18
2.4. Необходимость создания проигрывателя	23
2.5. Сравнение видео и имитационной модели.....	25
2.6. Скорость. Масштаб. Направление.	26
2.7. Взаимосвязь модельного и компьютерного времени.	28

2.8.	Задержка для одновременных событий	29
3.	Концептуальный этап проектирования	31
3.1.	Элементы управления.....	31
3.2.	Выбор архитектуры системы.....	34
3.3.	Состояние модели	36
4.	Технический этап.....	38
4.1.	Обзор IDE Eclipse.....	38
4.2.	Сериализация. Описание процесса.	39
4.3.	Типы сериализации.....	40
5.	Рабочий этап.....	44
5.1.	Выбор библиотеки для JSON сериализации	44
5.2.	Сериализация массива объектов разных типов с использованием GSON	44
5.3.	Запись состояний модели	46
5.4.	Чтение состояний модели	46
5.5.	Проигрывание состояний модели	41
6.	Исследовательская часть.....	48
6.1.	Работа со временем для разных типов моделей	48
6.2.	Стратегии работы со временем	51
6.3.	Стратегии выбора зависимостей между масштабом, скоростью и задержкой.....	53
6.4.	Выбор универсальной стратегии управления временем	57
7.	Организационно-экономическая часть.....	59
7.1.	Введение	59

7.2.1.	Расчет трудоемкости разработки технического задания	62
7.2.2.	Расчет трудоемкости выполнения эскизного проекта	63
7.2.3.	Расчет трудоемкости выполнения технического проекта	64
7.2.4.	Расчет трудоемкости выполнения рабочего проекта	65
7.2.5.	Расчет трудоемкости выполнения внедрения	66
7.2.6.	Расчет суммарной трудоемкости.....	67
7.3.	Определение стоимости разработки ПП	69
7.3.1.	Расчет основной заработной платы	69
7.3.2.	Расчет дополнительной заработной платы	70
7.3.3.	Отчисления на социальное страхование	70
7.3.4.	Накладные расходы	70
7.3.5.	Расходы на оборудование (амортизация)	71
7.3.6.	Результаты расчетов затрат на разработку программного продукта.....	72
7.4.	Вывод	72
8.	Мероприятия по охране труда и технике безопасности	73
3.1.	Введение	73
3.1.1	Анализ опасных и вредных факторов.....	73
3.2.	Опасные и вредны факторы.....	75
3.2.1.	Физические.....	75
3.2.1.1.	Опасные факторы	75
3.2.1.2.	Вредные факторы.....	75
3.3.9.	Типовой расчет виброизоляции системы кондиционирования	82

3.4.2.1. Разборка персональных компьютеров (ПЭВМ), рабочих станций и серверов	85
3.4.2.2. Обеспечение комплексности технологии разборки	87
3.4.2.3. Извлечение вторичных чёрных металлов	88
3.4.2.4. Извлечение вторичных цветных металлов.....	89
3.4.3. Реализация партии	90
3.4.3.1. Классификация отходов	91
3.4.3.2. Сертификация партий	92
ЗАКЛЮЧЕНИЕ.....	95
СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ.....	97
СПИСОК ИСПОЛЬЗОВАННОГО ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ ..	98

ПЕРЕЧЕНЬ СОКРАЩЕНИЙ

UML	Universal Modeling Language (Универсальный язык моделирования)
IDE	Integrated Development Environment – Интегрированная среда разработки
ИМ	Имитационная модель
ОС	Операционная система
ПИ	Пользовательский интерфейс
ПП	Программный продукт
ПО	Программное обеспечение
ПЭВМ	Персональная электронно-вычислительная машина
РДО	Ресурс Действие Операция – система имитационного моделирования
СДС	Сложная дискретная система
ТЗ	Техническое задание
ЭП	Эскизный проект
JDK	Java Development Kit

ВВЕДЕНИЕ

Математическое моделирование является важным элементом современных информационных технологий. Год от года в мире проектируются все более сложные технические системы. Иногда очень сложно представить систему без дополнительных средств визуализации и исследования. Для исследования применяется моделирование практически на каждом этапе жизненного цикла.

Исследование любой системы можно разделить на два основных типа: эксперимент с реальной системой и эксперимент с ее моделью. В свою очередь модель может быть физической или математической. А математические модели можно разделить на те, у которых есть аналитическое решение и на те, для которых используют имитационное моделирование. В рамках данного проекта рассматривается применение имитационного моделирования.

Широкое распространение ИМ в задачах анализа и синтеза объясняется сложностью (а иногда и невозможностью) строгих методов оптимизации, обусловленных высокой размерностью решаемых задач и невозможностью формализации сложных систем.

Одной из систем имитационного моделирования является Rao X. В системе реализованы процессно-ориентированный и событийный подходы. Для визуализации в данной системе применяется графическая оболочка. В ней реализованы модули построения графиков, анимации и трассировки.

Задача данного проекта состоит в том, чтобы сохранить и в дальнейшем проиграть состояния модели. Такая возможность необходима для упрощения анализа модели и возможности демонстрации модели непосредственному заказчику.

1. Техническое задание

Техническое задание разработано в соответствии с ГОСТ 19.201-78.

1.1. Основания для разработки

- 1) Разработка ведется на основании следующих документов:
 - Задание на выполнение дипломного проекта.
 - Календарный план на выполнение дипломного проекта.
- 2) Документы утверждены «12» марта 2016 года.
- 3) Тема дипломного проекта:

Интерактивный проигрыватель имитационных моделей в
системе имитационного моделирования Rao X

1.2. Назначение разработки

Основная цель данного дипломного проекта – создание программного продукта позволяющего сохранять и проигрывать состояния имитационной модели в системе Rao X.

1.3. Требования к программе или программному изделию

1.3.1. Требования к функциональным характеристикам

В системе должны быть представлены следующие функции:

- Сохранение файлов, содержащих информацию о прогоне модели;
- Чтение файлов, содержащих информацию о прогоне модели;
- Управление направлением проигрывания;
- Управление скоростью проигрывания;
- Управление задержкой для одновременных событий;
- Управление направлением проигрывания
- Запуск, пауза и остановка проигрывания

1.3.2. Требования к надёжности

Основное требование к надёжности направлено на поддержание в исправном и работоспособном состоянии ЭВМ, на которой происходит использование программного комплекса.

1.3.3. Условия эксплуатации

Аппаратные средства должны эксплуатироваться в помещениях с выделенной розеточной электросетью 220В $\pm 10\%$, 50 Гц с защитным заземлением при следующих климатических условиях:

- температура окружающей среды – от 15 до 30 градусов С;
- относительная влажность воздуха - от 30% до 80%;
- атмосферное давление - от 630 мм. р.с. до 800 мм. р.с.

1.3.4. Требования к составу и параметрам технических средств

Программный продукт должен работать на компьютерах со следующими характеристиками:

- объем ОЗУ не менее 4 Гб;
- объем жёсткого диска не менее 20 Гб;
- микропроцессор с тактовой частотой не менее 3ГГц;
- монитор с разрешением от 1366*768 и выше.

1.3.5. Требования к информационной и программной совместимости

Данная система должна работать под управлением операционной системы Linux.

1.3.6. Требования к маркировке и упаковке

Не предъявляются.

1.3.7. Требования к транспортированию и хранению

Не предъявляются.

1.3.8. Требования к программной документации

Не предъявляются.

1.4. Технико-экономические показатели

Расчет экономической эффективности разработанного приложения не является целью дипломного проектирования, однако возможный экономический эффект может быть достигнут за счет следующих преимуществ системы:

- 1) Работа под операционной системой Linux.
- 2) Открытие возможностей для дальнейшего развития системы.

1.4.1. Стадии и этапы разработки

Состав, содержание и сроки выполнения работ по созданию системы в выполнены соответствии с календарным планом на выполнение дипломного проекта.

1.4.2. Порядок контроля и приемки

Контроль и приемка приложения должны состоять из следующих этапов:

1. Создание файлов содержащих информацию о прогоне модели;
2. Чтения файлов содержащих информацию о прогоне модели;
3. Управление направлением проигрывания;
4. Выбора скорости проигрывания;

5. Выбора задержки для одновременных событий;

1.4.3. Приложение

Документы, используемые при разработке, приведены в списке использованных источников.

Используемое при разработке программное обеспечение:

- Операционная система Ubuntu Linux;
- JDK 8
- Среда разработки Eclipse Mars.2 Release (4.5.2);
- Библиотека Xtext, Xtend;
- Система управления версиями Git;
- GitHub веб-сервис для хостинга IT-проектов и их совместной разработки

2. Предпроектное исследование

2.1. Введение

Сложные системы, бизнес-процессы, системность, управление сложными системами – все эти термины широко используются практически во всех сферах деятельности современного человека. Это напрямую связано с желанием найти общесистемные принципы и методы для различных областей человеческой деятельности. Тем самым пытаюсь обобщить накопленный опыт и выработать универсальные решения для разных проблем.

Системный подход в решении любых задач в теории и практики является базисом, помогающим исследователю изучать систему, не обращая внимание, на ее физические особенности или какие-либо ограничения предметной области. В свою очередь в роли средства обеспечивающего эту возможность выступает модель.

За отсутствием модели человек может опираться лишь на практический опыт, тем самым, используя некоторые эвристики, на практике оказывается, что задачи имеющие аналитическое решение куда более быстро решаются средствами моделирования. Существуют модели физические, математические, цифровые, компьютерные и многие другие. Тип моделирования выбирается исходя из особенностей самого объекта исследования.

Имитационное моделирование (ИМ) заключается в построение компьютерной модели и проведение на ней экспериментов (иногда с участием человека – деловой игры)[1]. Представляя модели с любой глубиной и точности ИМ позволяет решать задачи недоступные для решения с использованием математических и статистических методов. Логично, что чем точнее и универсальнее модель, тем больше ресурсов

необходимо затратить на ее создание. При создании имитационной модели следует четко определять баланс между ними.

Инструменты ИМ появились достаточно давно примерно одновременно с появлением таких языков как Алгол и Фортран. [1]

2.2. Эволюция сложных систем

Отсутствие определения сложной системы привело к тому, что к ним относят практически все системы. Но вполне бесспорно, что не все являются таковыми. Более того, одна и та же система может быть одновременно и простой и сложной зависит от того с какой стороны на нее посмотреть и какие характеристики исследуются аналитиком.

Так, многие сложные системы обладают общими качествами:

- Отсутствие формального и полного математического описания.
- Стохастичность поведения, вызванная как внутренними особенностями системы, так и внешними воздействиями.
- “Нетерпимость” к управлению. Часто внешние воздействия к объекту управления может противоречить нормальному функционированию системы.
- Непостоянность характеристик системы. Элементы системы могут быть подвержены старению или просто меняются в зависимости от времени.
- Невоспроизводимость экспериментов.

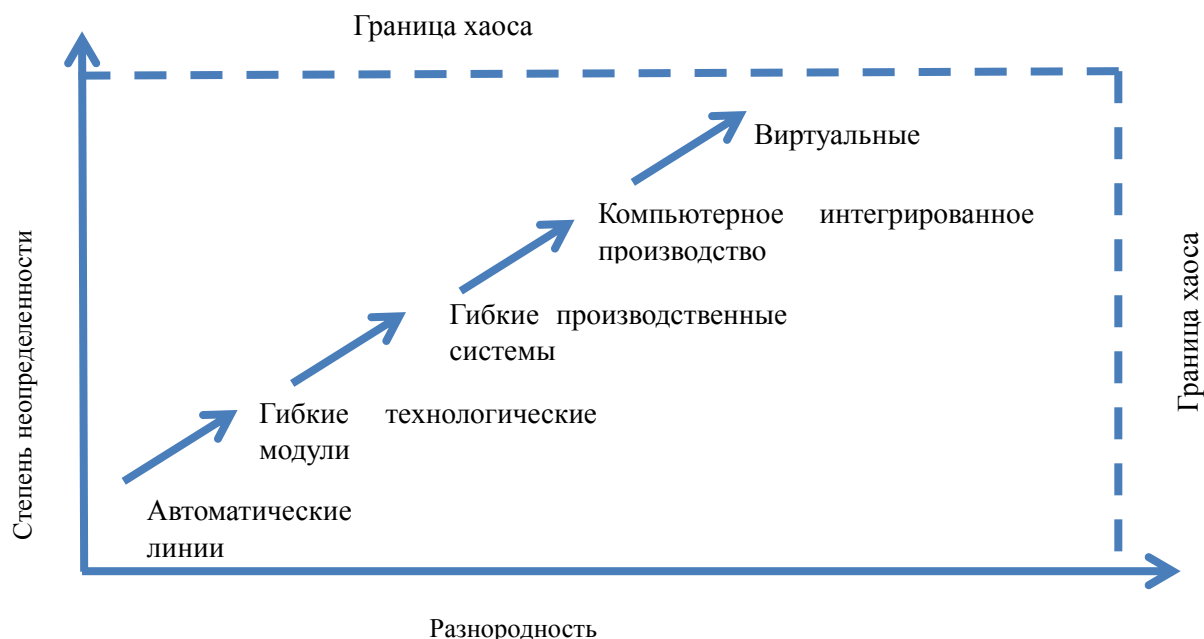


Рис 2.2. Эволюция производственных систем.

В качестве примера можно рассмотреть производственную систему. Чтобы получить некоторое схематичное отображение такой системы можно построить в координатах “степень неопределенности” и “разнородность” и получить картину эволюции такой систем.

Законы эволюции общие для живого и неживого мира. Сходства имеют место не только в структуре и функционировании, но так же и в принципах управления и принятия решений. Любая сложная система система сформированная в процессе эволюции не может управляться по средствам некоторого единого центра. Такие системы управляются распределенной системой управления, с большим количеством локальных систем управления. Все это способствует возникновению и развитию многоагентных систем и интеллектуальных организаций.

2.3. Подходы к построению имитационной модели

Модель это абстрактное представление системы глубина детализации, которой зависит от множества факторов. Существенность тех или иных факторов исследователь оценивает исходя из собственного

представления о важности и существенности данного фактора в системе с точки зрения исследователя.

Как и любой другой осмысленный процесс жизнедеятельности человека, моделирование начинается с формирования цели данного исследования. Целеполагание отличает человека разумного от других существ на нашей планете. А в рамках ИМ цель это основное ограничение для исследователя. Затем выбираются границы исследования и необходимый уровень детализации. От того насколько правильно было выбраны эти параметры будет зависеть результативность моделирования, а с другой стороны и в некотором роде наглядность результатов. Большое количество шума (несущественной информации) может осложнить анализ модели. В связи с этим моделирование это творческий процесс, требующий от исследователя глубинного понимания возложенных на него задач.

В описании системы необходимо включить критерии эффективности системы. В свою очередь оцениваемые альтернативные решения могут рассматриваться как часть модели или как ее входы. В случае оценки по критериям эффективности – выходы модели. На практике получение альтернативных решений делать процесс моделирования итеративным, что требует перестройки модели.

ИМ (В широком смысле) – процесс конструирования модели реальной системы и постановка экспериментов на этой модели с целью либо понять поведение системы, либо оценить (в рамках накладываемых ограничений) различные стратегии, обеспечивающие функционирование этой системы. [1]

ИМ (В узком смысле) – представление динамического поведения системы посредством продвижения ее от одного состояния к другому в

соответствии с хорошо известными операционными правилами (алгоритмами) [1]

Все системы ИМ в зависимости от способа представления основных процессов происходящих в моделируемом объекте разделяют на: детерминированные и стохастические, стационарные и нестационарные, дискретные и непрерывные, пошаговые и событийные.



Рис 2.3. 1. Этапы имитационного моделирования

На рис. 2.3.1. представлены основные этапы ИМ, наибольший интерес для нас представляет этап создания имитационной модели. Остальные этапы достаточно подробно описаны в литературе по ИМ.

Рассмотрим структурную схему вариантов построения модели.

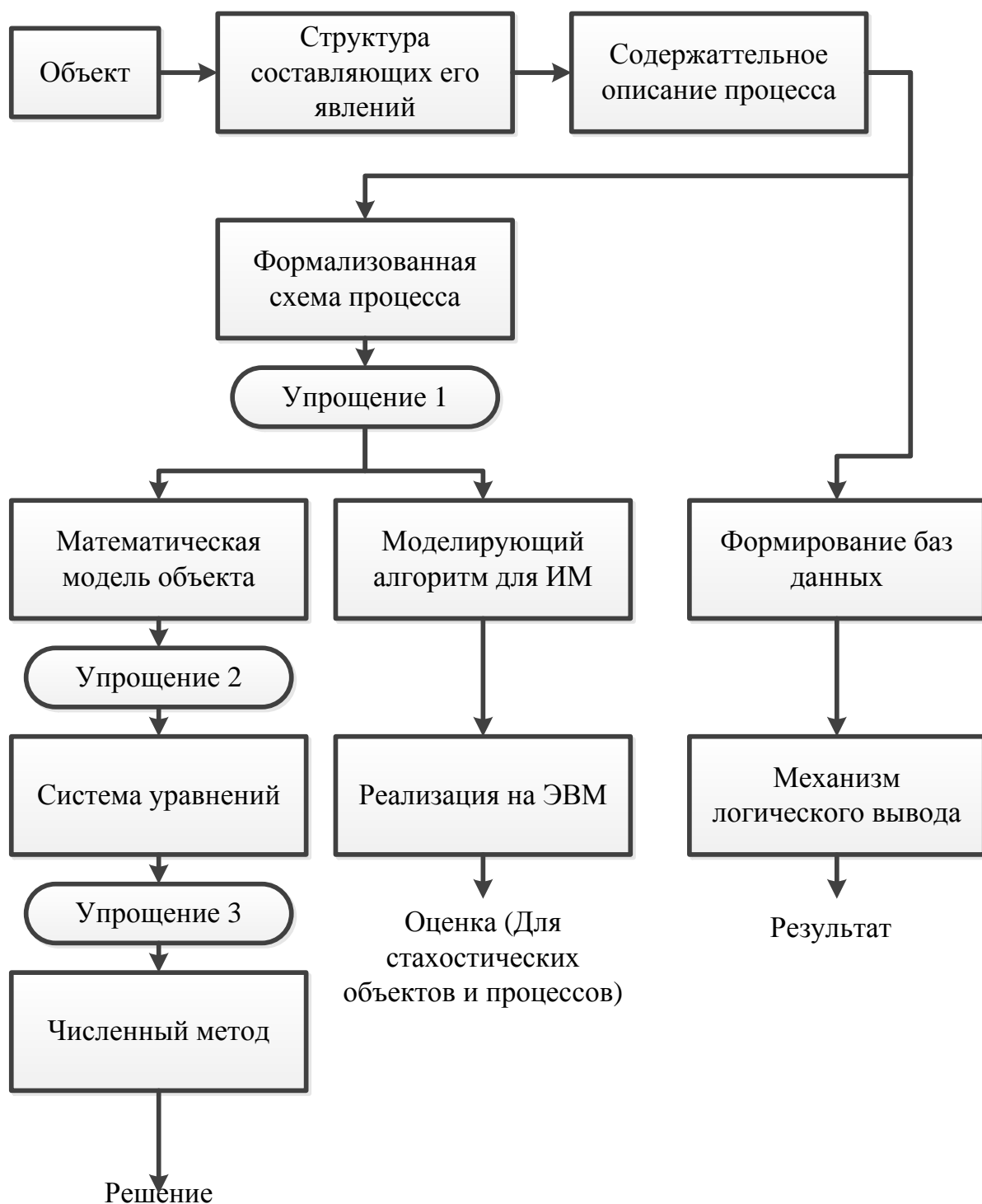


Рис 2.3.2. Структурная схема вариантов модели

Данный пример рассматривается для математической модели, имитационной модели и имитационной модели на основе искусственного интеллекта.

В общем случае для описания функционирования сложной системы можно могут быть использованы: события, действия и процессы (Рис 2.3.3)

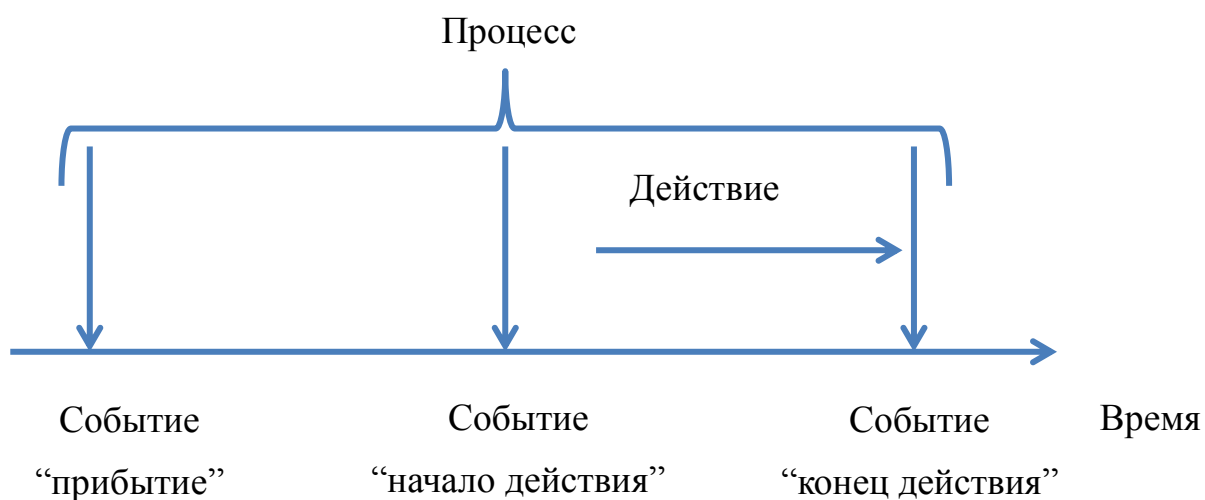


Рис 2.3.3 Взаимосвязь между событиями, действиями и процессом

В соответствии с этим выделяют три основных методологических подхода в построении имитационной модели: дискретно-событийный, подход сканирования активностей и процессно-ориентированный.

В случае событийного подхода исследователь фокусируется на описании событий происходящих в реальной системе. При этом процесс моделирования заключается в выборе события из некоторого списка будущих событий. Как критерий выбора выступает время наступления события. Выполнение события приводит к созданию новых событий в будущем. Все порожденные события упорядочиваются по времени.

В случае подхода сканирования активностей разработчик описывает все действия в имитационной модели. Внутри системы есть свои

собственные часы время, на которых продвигает сама система. На каждом шаге модельного времени происходит проверка условия наступления того или иного состояния в случае если все состояния отработали модельное время продвигается вперед

В случае процессно-ориентированного подхода исследователь описывает определенную схему для компонентов модели. Логика наступления задается одним оператором языка. Имитатор моделирует продвижение активных компонентов через систему, который сопровождается соответствующей последовательностью событий. [1]

2.4. Необходимость создания проигрывателя

Создание имитационной модели требует от исследователя глубокого понимания объекта исследования. При этом немало важно обладать еще и навыками программирования, и необходимо использовать современные компьютерные средства имитационного моделирования, обладающие удобным пользовательским интерфейсом. Иногда достаточно просто использовать заранее приготовленные блоки и лишь соединить элементы, описав их зависимости.

Rao X не является исключением. В системе реализованы: сканирование активностей, дискретно-событийный, и процессно-ориентированный подход. Пользовательский интерфейс достаточно удобен.

Основной алгоритм работы системы реализован в симуляторе. Симулятор позволяет моделировать, как бы странно это не звучало. Он просчитывает состояния модели при этом скоростью работы симулятора можно управлять. Так же есть возможность поставить модель на паузу или полностью остановить. Основная цель данной работы заключается в том, чтобы дать пользователю посмотреть на модель под другим углом. Часто

разработчик системы хочет переместиться в какой-то конкретный момент модели, сделать такой прыжок симулятор не позволяет. Предоставляется возможность только последовательного просмотра всех состояний модели.

Имитационная модель часто является подтверждением догадок исследователя. Часто это происходит в формате некоторого исследования. И рано или поздно возникает необходимость демонстрировать результаты работы. Текущий принцип работы системы обязывает показывать модель с начала до конца. А что если расчет модели происходил в течении нескольких дней? Заставить заказчика смотреть все с начала до конца?

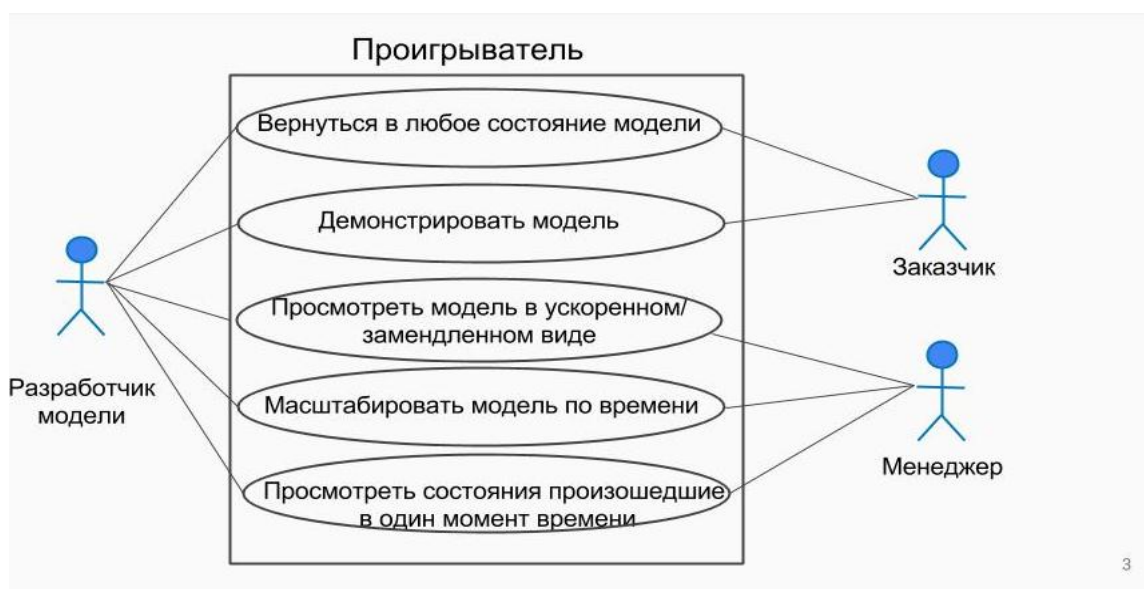


Рис. 2.4. Диаграмма вариантов использования

С другой стороны как правило модель создается для того чтобы подтвердить или опровергнуть ту или иную догадку. Т.е. получить некоторый конечный результат. Тут опять нужно показать только короткий промежуток времени модели.

Скорость моделирования ограничивается вычислительными характеристиками компьютера. И есть некоторый предел быстрее которого симулировать нельзя. А иногда хочется проигрывать существенно быстрее.

В симуляторе нигде не задаются единицы модельного времени и пользователь сам должен определять их. А точнее держать их в голове. Но хотелось бы иметь возможность выбора единиц измерения.

В имитационном моделировании существует целый класс моделей, в которых все события происходят в один момент времени. В рамках работы симулятора сложно понять, что же происходит в модели. Пользователь видит только конечный результат моделирования. Хотелось бы получить возможность просмотра событий произошедших в один момент времени.

Реализацией всех этих и многих других функциональностей и является предназначением проигрывателя.

2.5. Сравнение видео и имитационной модели

Видеофильм это хороший, а самое главное наглядный способ представления информации. Хотелось бы дать пользователю возможность просматривать модель как киноленту. Основываясь на подходах и технических решениях использованных в киноиндустрии попробуем найти различия и сходства модели и видеофильма.

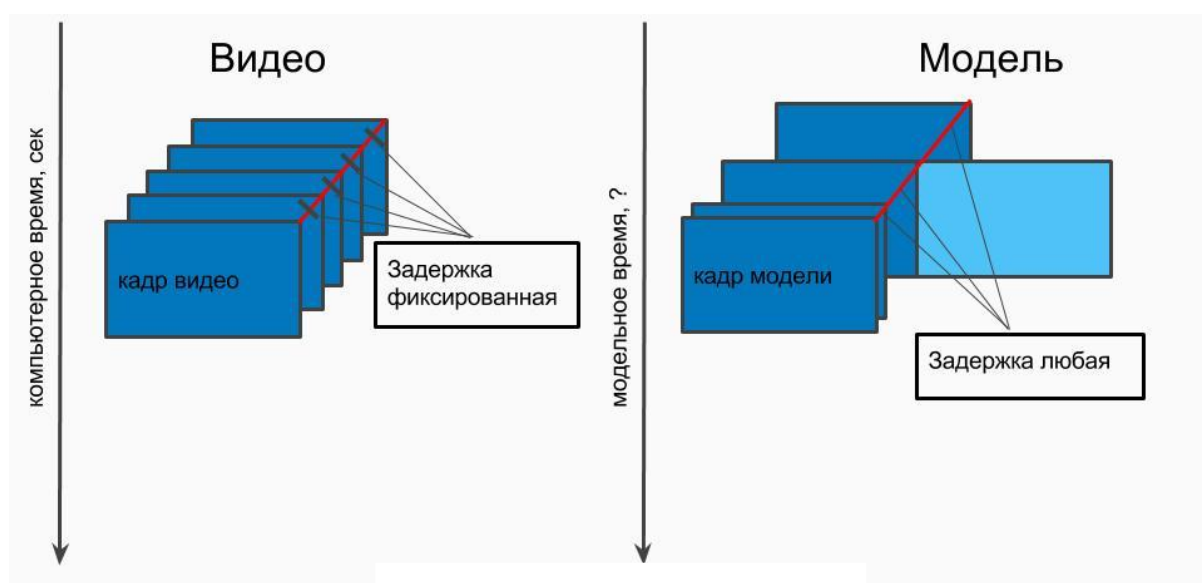


Рис. 2.5. Видео и модель

Состояние модели можно воспринимать как некоторый кадр. По крайней мере, пользователь системы так и видит все то, что происходит. Как раз динамические элементы пользовательского интерфейса и создают ощущение просмотра фильма.

Как и в видео состояния модели следуют друг за другом. В видео задержка между кадрами определяется исключительно частотой кадров в секунду, и она постоянна. В модели эта задержка не должна быть, а иногда ее вообще нет.

Так же в видео все проникает в единицах времени компьютера - секундах, а в модели это свои модельные единицы которые не имеют размерности.

2.6. Скорость. Масштаб. Направление.

Рассмотрим проектирование основных механизмов работы проигрывателя. В мире проигрывателя существуют две шкалы времени: время симуляторов, которое фиксируется на этапе выполнения модели и

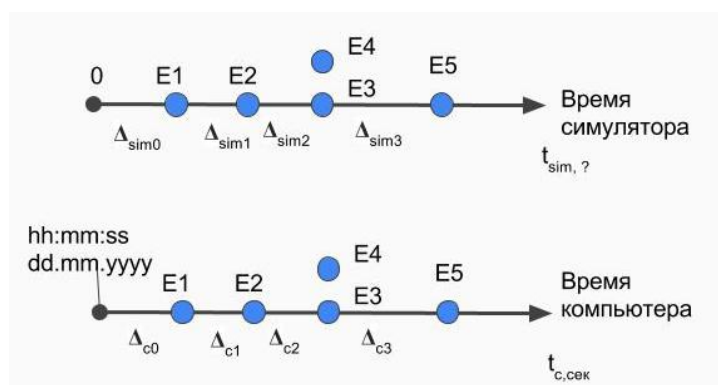


Рис. 2.6. 1. Шкалы времени симулятора и компьютера

отсчитывается от нуля. С другой стороны время компьютера, на которое мы ни как не можем повлиять, мы можем лишь взять некоторое текущее значение времени компьютера и зафиксировать его как время начала. Как я уже раньше упоминал, единицы времени симулятора нам не известны, а

время компьютера имеет чрезвычайно маленькую дискрету, но мы будем округлять время до миллисекунд. Я предлагаю, связывать время симулятора со временем компьютера и тем самым попробовать управлять скоростью проигрывания.

$$V = \frac{\Delta sim [?]}{\Delta c [сек]} \cdot V^*$$

Рис.2.6.3 Формула скорости

Для того что бы нам соотносить процессы протекающие в компьютере со временем симулятора, будем считать время в дельтах. Так отношение дельты времени симулятора к дельте времени компьютера, умноженное на коэффициент скорости будут являться реальной скоростью проигрывателя. Как я уже сказал, нам неизвестна единицы времени симулятора, для этого в отношении дельт нам нужно будет учесть масштабный коэффициент.

Так же следует отметить, что скорость является векторной величиной. Это

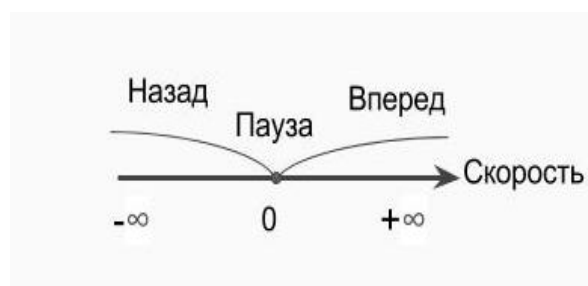


Рис.2.6.3 Скорость

можно использовать для управления направлением проигрывания. В

случае если, значение скорости больше нуля, то мы будем проигрывать вперед, когда же значение скорости будет нулевым, для нас это будет эквивалентно паузе. Если же значение скорости меньше нуля, то проигрывание модели будет осуществляться в обратном направлении.

2.7. Взаимосвязь модельного и компьютерного времени.

Как же связано модельное и компьютерное время? Время продвигается внутри самого симулятора. В свою очередь никак не зависит от времени компьютера. Существует много моделей, в которых в один момент времени, может происходить несколько событий. В случае симулятора, пользователь увидит скорее всего только последнее событие, так как задержка между событиями будет обусловлена лишь вычислительными мощностями компьютера. В современных компьютерах они достаточно высоки и задержка в доли миллисекунды, не позволит

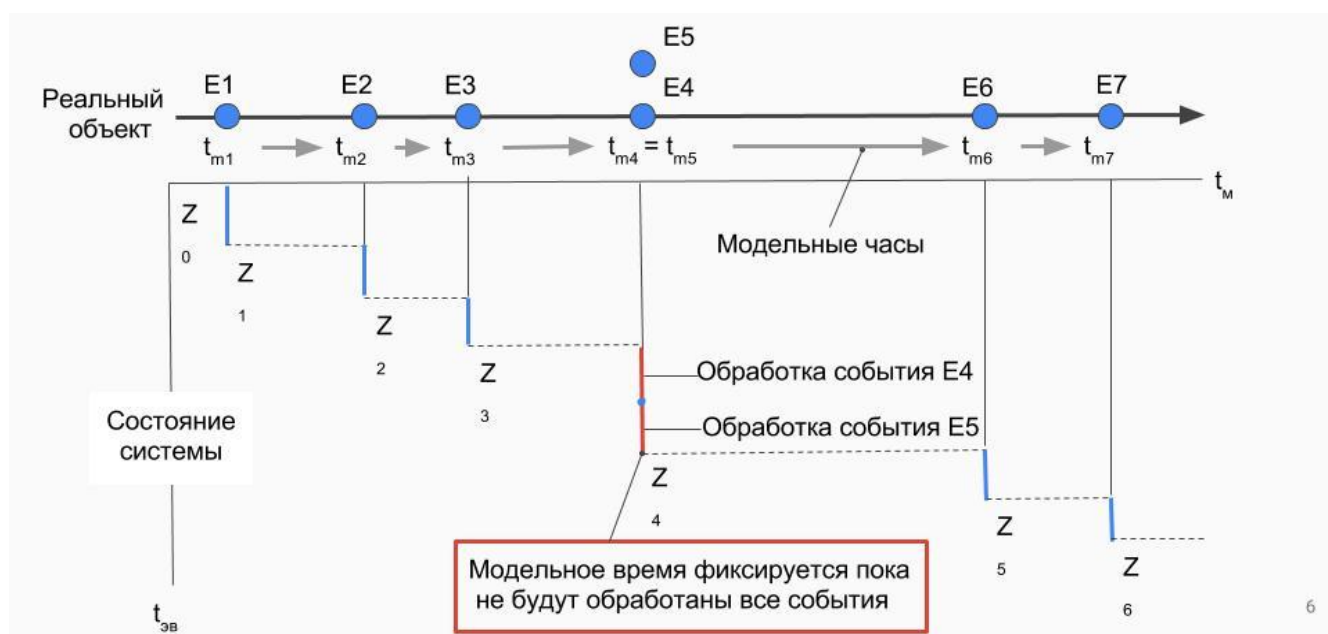


Рис.2.7 Время

пользователю корректно воспринимать эти события.

Как же дать пользователю возможность просматривать событие происходящее в один момент времени. Вполне логично, просто дать задержку между ними.

2.8. Задержка для одновременных событий

На первый взгляд никаких сложностей с задержкой нет, можно просто дать некоторую фиксированную задержку между событиями. А вдруг ее будет недостаточно, да и что считать достаточным? Если мы сделаем задержку наоборот слишком большой, то скорость при этом будет маленькой, в таком случае у пользователя возникнет ложное впечатление о реальной модели. Будет казаться, что те события, которые происходили одновременно, на самом деле имеют большую длительность, чем она есть в реальной модели, что приведет к неверной трактовке результатов моделирования. В своем проекте, я предлагаю предоставить пользователю возможность самостоятельно выбирать задержку.

Существует проблема взаимосвязи скорости и времени задержки. Должны ли быть связаны между собой они? Я провел ряд экспериментов и

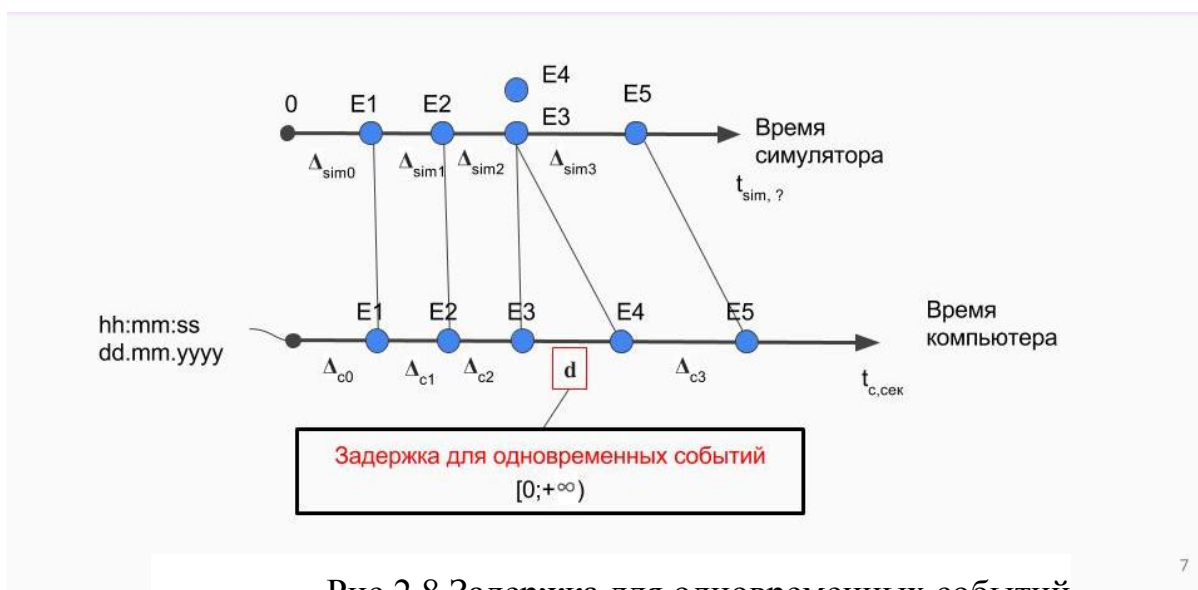


Рис.2.8 Задержка для одновременных событий

выяснил, что ожидаемым, интуитивным поведением для пользователя,

будет как раз изменение той самой задержки. До этого я упоминал о необходимости применения масштабного коэффициента. На мой взгляд, следует учитывать его вместе со скоростью и задержкой.

3. Концептуальный этап проектирования

3.1. Элементы управления

Рассмотрим проектирование пользовательского интерфейса. Основными элементами управления являются: кнопки старта, паузы и стоп. В этом случае больших отличий от стандартного интерфейса видео проигрывателя нет, за исключением того что обычно функции проигрывания в обратном направлении не добавляют. Для нашей системы, данная функциональность является очень важной.

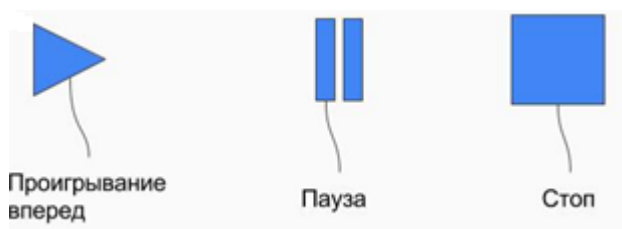


Рис. 3.1. 1. Базовые элементы ПИ

Давайте рассмотрим проектирование двух вариантов пользовательского интерфейса.

В первом случае, к выше указанным кнопкам необходимо будет добавить ползунок управления скоростью, который примерно в середине будет иметь риску.

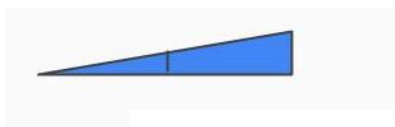


Рис. 3.1.2. Ползунок скорости

Если ползунок находится на этой риске, то проигрывание должно осуществляться со скоростью равной скорости использованной на этапе

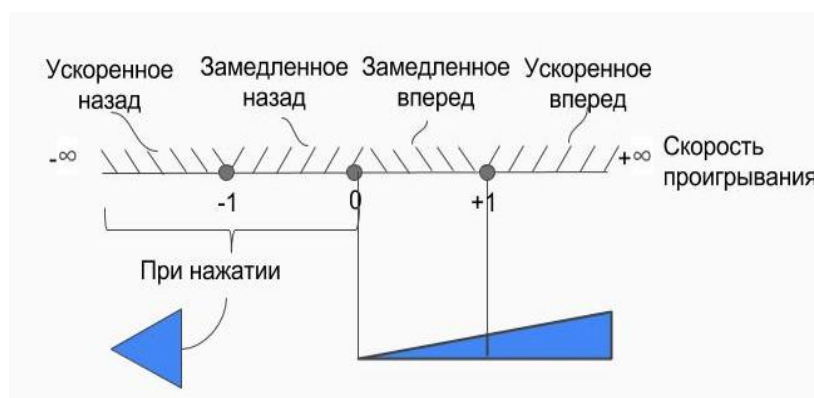


Рис. 3.1.3. Ползунок скорости. Описание.

симулирования. Если значение ползунка находится правее риски, то скорость будет превышать скорость на этапе симулирования. Если же левее, то будет меньше. В таком подходе следует учитывать, что в случае если пользователь захочет изменить направление проигрывания, функциональность ползунка остается прежней. На предыдущем этапе было принято, что значение скорости может быть как положительной, так и отрицательной величиной. В этом случае при выборе направления проигрывания назад, мы просто будем брать выбранное пользователем значением со знаком минус.

Во втором случае, предлагается использовать единый элемент управления в виде

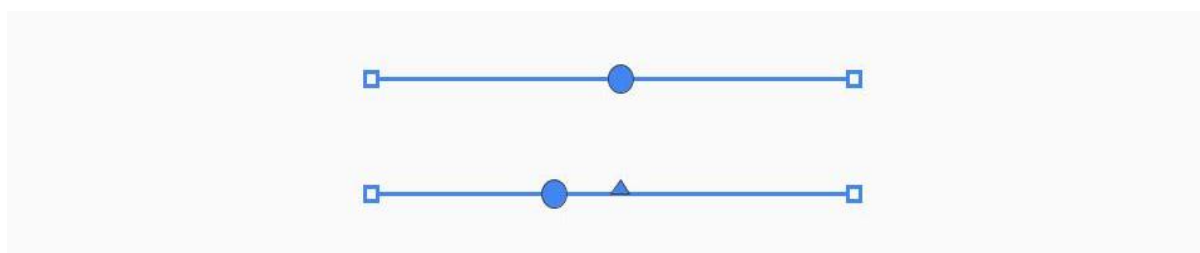


Рис. 3.1.4. Элемент управления.

ползунок, в среднем положении которого модель не проигрывается, то есть остановлена. Когда же мы начинаем сдвигать ползунок вправо, тем самым

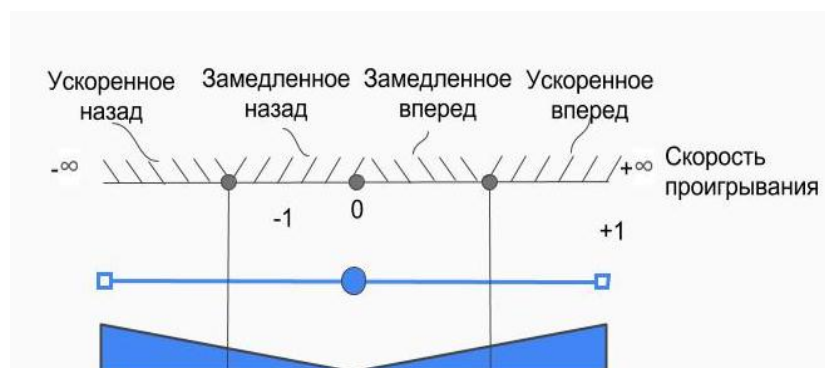


Рис. 3.1.4. Схема положений элемента управления.

мы не только увеличиваем значение скорости проигрывания, но и определяем направление проигрывания в этом случае вперед. Когда же из нулевого положения пользователь будет перемещать ползунок влево, направление будет выбрано в противоположную сторону. В этом случае мы можем заменить три элемента управления, использованных в первом случае, одним, обладающим теми же свойствами.

На мой взгляд, второй вариант хоть и выглядит более компактно, нежели первый, но его использование не кажется интуитивным. Хотя с точки зрения функционирования самого проигрывателя, данный подход отражает внутреннее устройство более точно.

Как для первого, так и для второго случая, следует учитывать, что в крайнем положении пользователь на самом деле ожидает получить не линейное увеличение скорости, а скорее логарифмическое. Так как в крайнем положении интуитивно кажется, что это максимально допустимая скорость.

Для выбора масштаба необходимо добавить выпадающее меню.



Рис. 3.1.5. Масштаб

Как было указано в исследовании, скорость, масштаб и задержка для одновременных событий должны коррелировать.

Для управления временем задержки для одновременных событий предлагается использовать ползунок, похожий на тот, что мы использовали в первом варианте. Величина максимальной задержки должна меняться в зависимости от того какой масштаб выбрал пользователь.

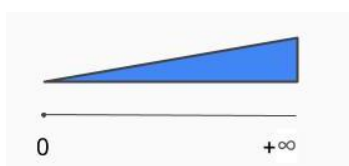


Рис. 3.1.6. Задержка

3.2. Выбор архитектуры системы

Важным аспектом проектирования интерактивного проигрывателя в системе имитационного моделирования Rao X является формат представления данной системы.

Мною были рассмотрены несколько основных вариантов реализации.

Первый вариант - это создать независимую от основного проекта систему, которая на входе будет получать исключительно результаты моделирования. Изучив подробнее структуру системы, я пришел к выводу,

что сделать это конечно можно, но при этом возникнут некоторые трудности. Оба проекта должны иметь техническую поддержку. В случае, если графики в первом проекте претерпят архитектурные изменения, это повлечет за собой необходимость внесения изменений в другую систему, что значительно усложнит поддержку и разработку. Так же хотелось бы иметь схожие интерфейсы в двух системах и, соответственно, использовать одни и те же виджеты. В силу того, что это разные проекты, необходимо будет копировать код из одного проекта в другой, что с моей точки зрения нерациональным.

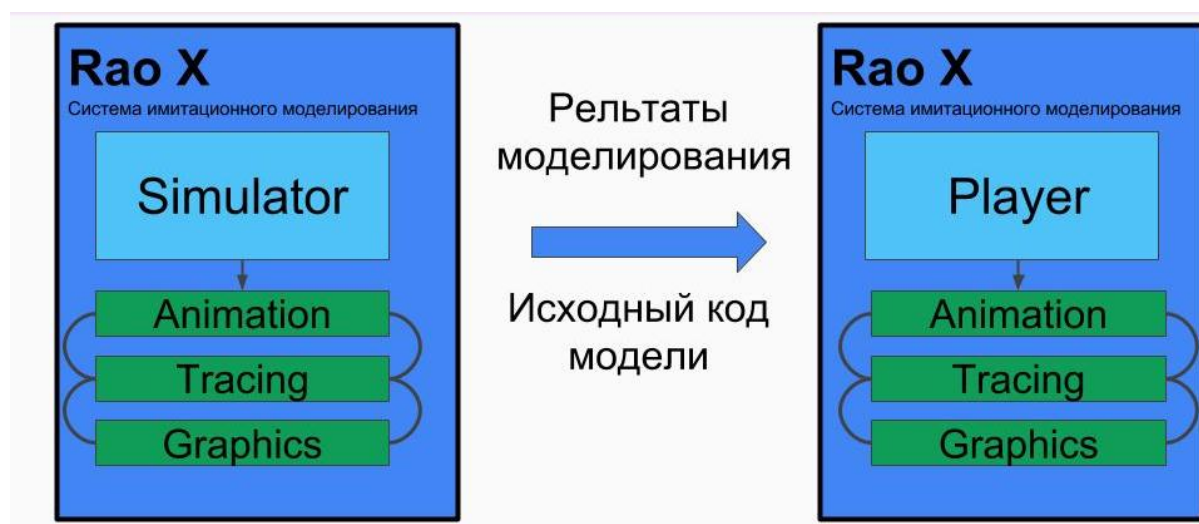


Рис. 3.2.1. Первый вариант интерфейсов

Второй вариант — это создать подсистему для системы имитационного моделирования. В этом случае мы избавляемся от необходимости переносить код из одного проекта в другой и получаем возможность использовать уже имеющиеся функциональности. Я предлагаю создать новую перспективу для проигрывателя, которая будет содержать все виджеты из основной системы, но при этом иметь свой уникальный механизм управления.

С другой стороны стоит правильно понимать основное предназначение проигрывателя. Это все же больше некоторое дополнение к системе, нежели отдельное полноценно функционирующий модуль.

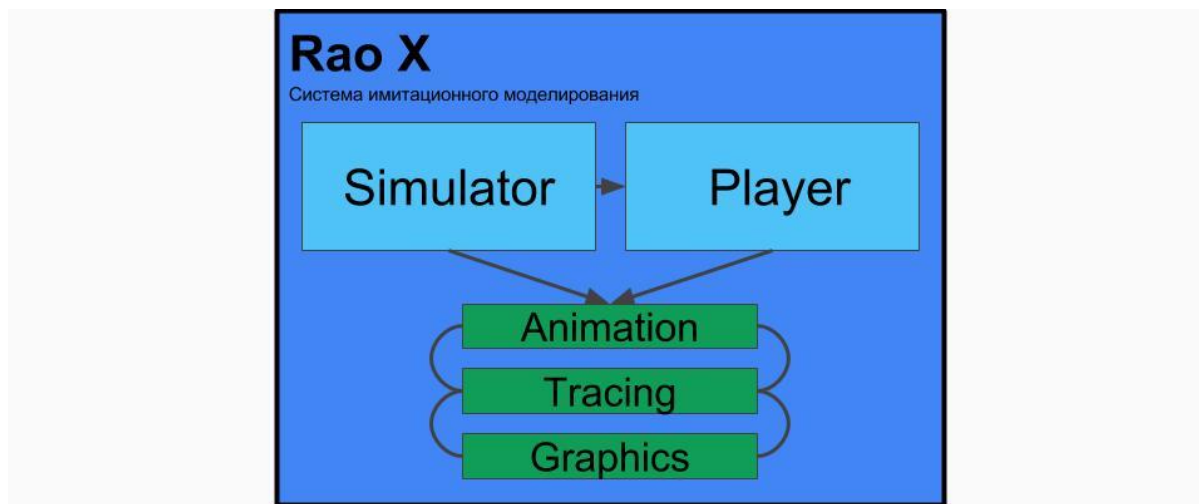


Рис. 3.2.2. Второй вариант интерфейсов

3.3. Состояние модели

Основной задачей проекта является проигрывание состояний модели. Для этого необходимо понять, что же является состоянием модели? Подробно изучив структуру системы, я обнаружил, что состояние модели - это на самом деле множество всех ресурсов. Для этого необходимо создать некоторый элемент в системе, который будет хранить в себе все ресурсы.

Для нас модель представляет собой множество всех состояний для каждого момента времени. Задача проигрывателя уметь воспринимать эти состояния и корректно их обрабатывать.

Прежде чем перейти к обработке, необходимо сохранить состояние модели. Более подробно можно прочитать о сохранении состояния модели в технологической части проекта.

Помимо самих состояний, как атрибут, нам необходимо сохранять и модельное время, соответствующее каждому состоянию. Это необходимо для сохранения задержек между событиями.

Сам процесс записи данного массива состояний называется сериализацией. Процесс обратный сериализации называется десериализацией, он необходим для считывания состояний из файла.

Стоит отметить, что потребность сохранения состояний модели в файл, а не просто хранение в памяти компьютера, необходимо при повторном использовании результатов моделирования а так же для возможности просмотра модели в текстовом представлении.

Следующим важным этапом работы является чтение состояний модели из файла, десериализация. Как критерий правильности десериализации, можно выдвинуть повторную сериализацию и сверку на идентичность их текстовых представлений.

4. Технический этап

4.1. Обзор IDE Eclipse

Eclipse — свободная интегрированная среда разработки модульных кроссплатформенных приложений. Развивается и поддерживается Eclipse Foundation.

Наиболее известные приложения на основе Eclipse Platform — различные «Eclipse IDE» для разработки ПО на множестве языков (например, наиболее популярный «Java IDE», поддерживавшийся изначально, не полагается на какие-либо закрытые расширения, использует стандартный открытый API для доступа к Eclipse Platform).

Основой Eclipse является платформа расширенного клиента (RCP — от англ. rich client platform). Её составляют следующие компоненты:

Ядро платформы (загрузка Eclipse, запуск модулей);

OSGi (стандартная среда поставки комплектов (англ. bundles));

SWT (портируемый инструментарий виджетов);

JFace (файловые буферы, работа с текстом, текстовые редакторы);

Рабочая среда Eclipse (панели, редакторы, проекции, мастера).

GUI в Eclipse написан с использованием инструментария SWT. Последний, в отличие от Swing (который самостоятельно эмулирует графические элементы управления), использует графические компоненты данной операционной системы. Пользовательский интерфейс Eclipse также зависит от промежуточного слоя GUI, называемого JFace, который упрощает построение пользовательского интерфейса, базирующегося на SWT.

Разработка Rao X, а так же интерактивного проигрывателя, ведется в системе Eclipse. Rao X по сути является дочерней версией Eclipse. Многие элементы использованные в системы заимствованы.

4.2. Сериализация. Описание процесса.

Сериализация (в программировании) — процесс перевода какой-либо структуры данных в последовательность битов. Обратной к операции сериализации является операция десериализации (структуризации) — восстановление начального состояния структуры данных из битовой последовательности [2].

Любой из схем сериализации присуще то, что кодирование данных последовательно по определению, и извлечение любой части сериализованной структуры данных требует, чтобы весь объект был считан от начала до конца и воссоздан. Во многих приложениях такая линейность полезна, потому что позволяет использовать простые интерфейсы ввода-вывода общего назначения для сохранения и передачи состояния объекта. В приложениях, где важна высокая производительность, может иметь смысл использовать более сложную, нелинейную, организацию хранения данных.

Сериализация предоставляет несколько полезных возможностей:

- метод реализации сохраняемости объектов, который более удобен, чем запись их свойств в текстовый файл на диск и повторная сборка объектов чтением файлов;
- метод осуществления удалённых вызовов процедур, как, например, в SOAP;
- метод распространения объектов, особенно в технологиях компонентно-ориентированного программирования, таких как COM и CORBA;

- метод обнаружения изменений в данных, изменяющихся со временем.

Для наиболее эффективного использования этих возможностей необходимо поддерживать независимость от архитектуры. Например, необходимо иметь возможность надёжно воссоздавать сериализованный поток данных независимо от порядка байтов, используемого в данной архитектуре. Это значит, что наиболее простая и быстрая процедура прямого копирования участка памяти, в котором размещается структура данных, не может работать надёжно для всех архитектур. Сериализация структур данных в архитектурно-независимый формат означает, что не должно возникать проблем из-за различного порядка следования байт, механизмов распределения памяти или различий представления структур данных в языках программирования.

4.3. Типы сериализации.

Для сериализации существует множество форматов. Наиболее популярные приведены ниже.



Рис. 4.3.1. Типы сериализации

Для выбора подходящего типа сериализации было выдвинуто 4 основных критерия: может ли человек прочитать формат, имеется ли стандартизованный API, используется ли в основном проекте и поддерживаются ли ссылки.

	Человек прочитает?	Стандартное API?	Используется в осн. проекте?	Поддерживает ссылки?
XML	да	да	нет	да
JSON	да	да	да	да
Бинарная	нет	да	нет	да
ini	да	нет	нет	да
Текстовый	да	нет	нет	нет

Рис. 4.3.1. Сравнительная таблица типов сериализации

Больше других форматов по предложенным критериям подошел JSON. Его мы и будем использовать в проекте.

4.4. Алгоритм проигрывания состояний модели

Проигрыватель состояний моделей находится в 5 основных состояниях: играет, пауза, остановлен, проинициализирован, окончено.

На первом этапе проигрывателя происходит инициализация. В этот момент считываются состояния модели из файла, и происходит их десериализация, а так же инициализируются виджеты.

После того как все необходимые элементы проинициализированы происходит прогон основного цикла проигрывателя. В процессе проверяется, не выполнены ли условия остановки проигрывания. [См. 4.4.]

На блок-схеме ниже не описывается подробно механизм выставления задержки. Он проверяет, есть ли выставленная пользователем задержка для одновременных событий. И в зависимости от текущих значений скорости, масштаба и задержки выставляет задержку.

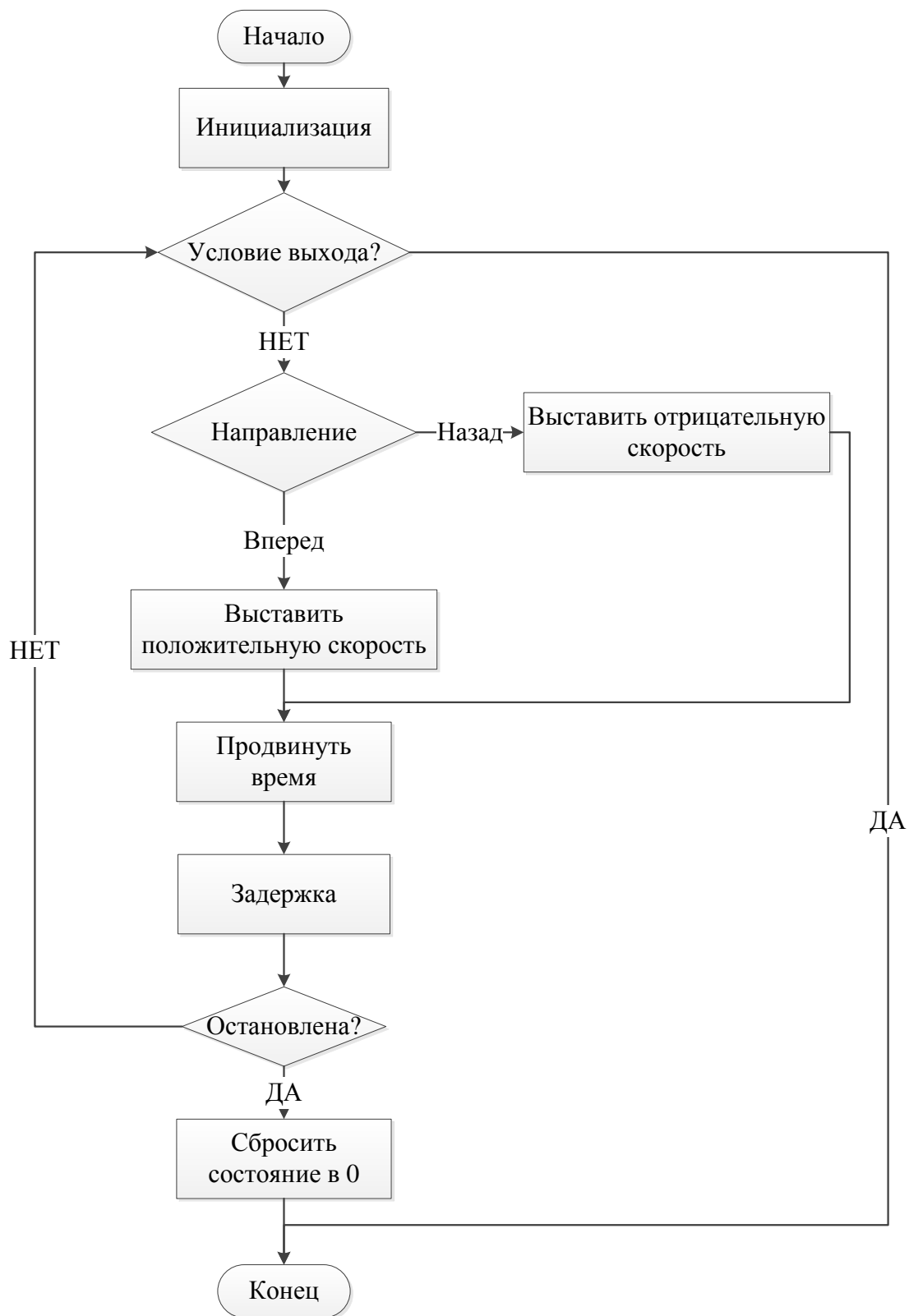


Рис. 4.4. Блок-схема алгоритма проигрывателя

5. Рабочий этап

5.1. Выбор библиотеки для JSON сериализации

Выбор из библиотек, позволяющих сериализовывать JSON, был достаточно проблематичен, все библиотеки обладают примерно одним функционалом. По сути, доверившись авторитету качества продуктов google, я решил использовать библиотеку gson.

Как выяснилось на рабочем этапе, работа gson не так уж проста. Немного помучавшись с сериализацией сложных объектов, задача все же была решена. Возможно, было бы проще использовать другую библиотеку. Но в силу того, что получилось решить задачу и таким способом другие библиотеки не рассматривались.

5.2. Сериализация массива объектов разных типов с использованием GSON

Одной из задач данного проекта состоит в том чтобы сериализовать Java объект в json. Для сериализации был использовал gson. Для начала я попробовал сериализовать простой объект:

```
public class Ex1Write {  
    static class Event {  
        private int IntNumber;  
        private double DoubleNumber;  
        private String name;  
  
        public Event() {  
  
        }  
  
        public Event(int IntNumber, double DoubleNumber, String name) {  
            this.IntNumber = IntNumber;  
            this.DoubleNumber = DoubleNumber;  
            this.name = name;  
        }  
    }  
}
```

Рис. 5.5.1. Пример объекта

Все прошло хорошо. Без особых трудностей.

Потом я попробовал выполнить это с нужным мне объектом. Опять-таки, я успешно с первого раза смог сериализовать большой и сложный

объект. Получая полноценный json файл. Однако, на этапе десиаризации, возникли проблемы. Библиотека Gson ничего не знала об объектах, записанных в файл. На что он выводил неприятное сообщение об ошибке: «Attempted to serialize java.lang.Class: <MyClass>. Forgot to register a type adapter?».

После продолжительных поисков, в документации была обнаружена информация по этой проблеме:

При десириализации с использованием функции fromJson подобъектов разных типов (Json, Collection.class) Gson не будет работать, так как он не знает, как отобразить входные данные в типы. Gson требует, чтобы вы предоставляли абстрактную версию коллекции в fromJson (). Есть три варианта:

- Использование Gson в анализаторе (API низкого уровня потоковый парсер или JsonParser) для синтаксического анализа элементов массива, а затем использование Gson.fromJson() на каждом элементе массива.
- Регистрация адаптера типа для Collection.class, который смотрит на каждый из элементов массива и отображает его на соответствующие объекты. Недостаток такого подхода в усложнении десериализацию других типов коллекций в Gson.
- Регистрация адаптера типа для MyCollectionMemberType и использовать fromJson () с Collection <MyCollectionMemberType>.

Первый вариант, предложенный разработчиками достаточно трудоемок. В этом подходе необходимо точно знать порядок записи объектов, а так же при чтении нужно указывать их порядок.

По данной тематике было проведено несколько испытаний. Программный код можно посмотреть в Приложении 1.

5.3. Запись состояний модели

Для записи состояний модели использовался паттерн наблюдатель.

Наблюдатель (англ. Observer) — поведенческий шаблон проектирования. Также известен как «подчинённые» (Dependents). Создает механизм у класса, который позволяет получать экземпляру объекта этого класса оповещения от других объектов об изменении их состояния, тем самым наблюдая за ними [2].

Так получая оповещения при каждом событии, происходящем в симуляторе, мы вызывается метод, позволяющий нам сохранить состояние модели. К концу прогона модели, мы получаем массив состояний системы. С использованием Gson записываем данный массив в файл в формате json.

Параллельно с записью состояний модели, используя паттерн наблюдатель, мы получаем массив времен наступлений событий и записываем его в файл в формате json.

Запись начинается после оповещения от симулятора о том, что прогон закончен.

Более подробно о записи информации о прогоне в json можно посмотреть в Приложении 2.

5.4. Чтение состояний модели

Чтение информации является более трудоемким процессом с точки зрения разработки. Сама проблема десериализации с общей точки зрения описана в главе 5.2.

Часть информации, которой мы обладали на этапе записи, не доступна на этапе чтения. Это информация о структуре модели. Поэтому

для того чтобы десериализовать модель, необходимо получить эту информацию. Так как, архитектурно, проигрыватель является дополнением к системе, мы имеем доступ к скомпилированным исходникам модели, где мы берем информацию о недостающих классах.

Имея всю необходимую информацию, мы можем получить состояния модели в виде java объектов и использовать их для проигрывания

Подробнее о чтении информации из файла можно посмотреть в Приложении 2.

6. Исследовательская часть

6.1. Работа со временем для разных типов моделей

Как было упомянуто в предпроектном исследовании, работа со временем в проигрывателе имитационных моделей несколько отличается от видео проигрывателей. Для того что бы лучше исследовать данный вопрос, я разделил все модели на 3 основных типа моделей: все события в которых происходит в разные моменты времени, все события в которых происходит в один момент времени и события в которых происходят в смешенные моменты времени (в один момент времени и в разные моменты времени)

1. Модели, все события в которых происходит в разные моменты времени

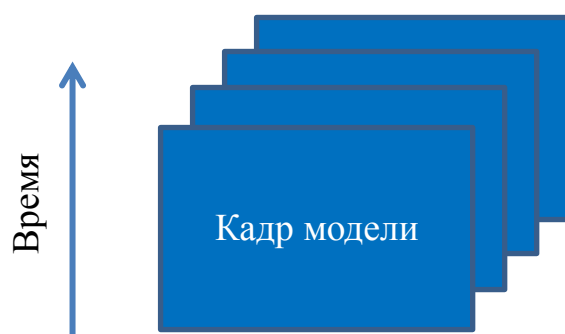


Рис. 6.1.1. Модель с событиями в разные моменты времени

Применимые стратегии работы со временем для данного типа моделей (см. 6.2.):

Как записывали модель, так и проигрываем;

Данная стратегия ничего не изменяет в исходном порядке состояний модели. Что собственно и требуется.

Применимые стратегии выбора зависимостей между масштабом, скоростью и задержкой (см. 6.3.):

Масштаба и задержки нет, есть только скорость;

Скорость + масштаб, задержки нет;

Наличие задержки для данного типа моделей не является необходимым. Поэтому среди стратегий были выбраны те, которые не содержат ее.

2. Модели, все события в которых происходит в один момент времени



Рис. 6.1.2. Модель с событиями в один моменты времени

Применимые стратегии работы со временем для данного типа моделей (см. 6.2.):

Фиксированная задержка для одновременных событий;

Настраиваемая задержка для одновременных событий;

Использование стратегии, в которой не меняется исходный порядок состояний модели, не применим в данном случае. Так мы просто потеряем все события кроме последнего в случае 1, из за того что компьютер проиграет их слишком быстро в случае 2 только будет последнее событие.

Применимые стратегии выбора зависимостей между масштабом, скоростью и задержкой (см. 6.3.):

Скорости и масштаба нет, только задержка;

Скорости нет, масштаб + задержка;

Скорость не является важным параметром для данного типа моделей. Так как все события происходят в один момент времени, то и ускорять/замедлять нечего.

3. Модели, события в которых происходят в смешанные моменты времени (в один момент времени и в разные моменты времени)



Рис. 6.1.3. Модель с событиями в смешанные моменты времени

Применимые стратегии выбора зависимостей между масштабом, скоростью и задержкой (см. 6.3.):

Как записывали модель, так и проигрываем:

Для одновременных событий показывается только последнее событие:

Фиксированная задержка для одновременных событий;

Настраиваемая задержка для одновременных событий;

Применимые стратегии выбора зависимостей между масштабом, скоростью и задержкой (см. 6.3.):

Масштаб отсутствует. Скорость и задержка независимы;

Масштаб отсутствует. Скорость и задержка зависимы;

Масштаб и скорость зависимы, задержка независима;

Масштаб, задержка и скорость связаны;

6.2. Стратегии работы со временем

Для того чтобы проработать все возможные варианты работы со временем, необходимо разработать несколько стратегий. Подробно изучив предметную область, я разработал ряд стратегий для управления временем. Каждая из стратегий применима для определенных моделей, то есть на данном этапе, мы не планируем создать некоторую универсальную стратегию, а лишь предлагаем стратегии, которые возможно применить лишь в ограниченном количестве моделей.

А. Как записывали модель, так и проигрываем

В таком подходе мы не стараемся изменить записанную информацию. Стараемся повторить ровно то, что происходило в



Рис. 6.2.1. Стратегия «Как записали, так и проиграем»

симуляторе.

Б. Для одновременных событий показывается только последнее событие

В данном подходе происходит оптимизация показа модели. В случае, если в смешанной модели мы имеем несколько одновременных событий, пользователю будет отображаться только последнее.

Данный подход позволяет оптимизировать вычисления. Так как информацией о части состояний модели мы пренебрегаем. Использование такого подхода оправдано, когда для разработчика модели не важны события, происходящие в один момент времени.

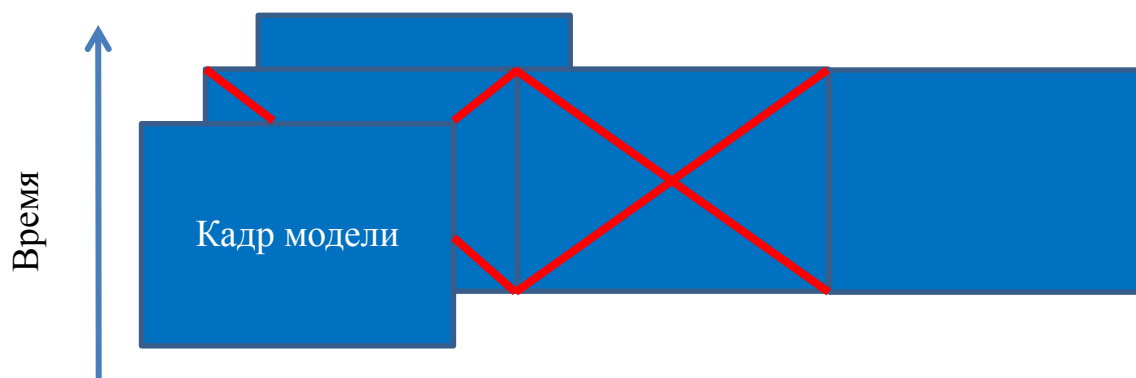


Рис. 6.2.2. Стратегия «Для одновременных, только последнее событие»

В. Фиксированная задержка для одновременных событий

Использование фиксированной задержки для событий происходящих

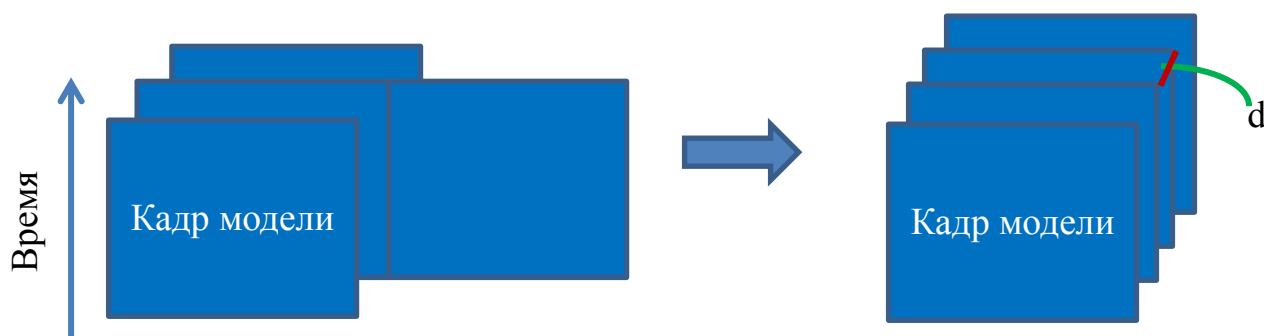


Рис. 6.2.3. Стратегия «Фиксированная задержка»

одновременно позволяет пользователю системы не задумываться о выборе задержки. Но при этом пользователь получает возможность просматривать события, происходящие в один момент времени. К недостаткам следует отнести неконсистентность с моделью, которую пользователь видел на этапе симулирования, а так же неосведомленность пользователя о

величине задержки. Формально такую стратегию можно назвать слайд шоу.

Г. Настраиваемая задержка для одновременных событий

Настраиваемая задержка (d^*) дает возможность пользователю

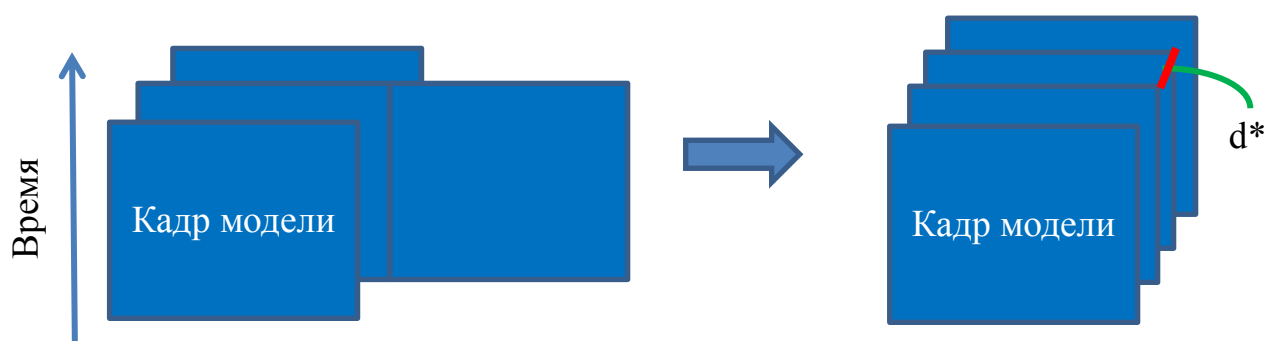


Рис. 6.2.3. Стратегия «Фиксированная задержка»

управлять временем задержки в зависимости от его желаний. Такой подход куда более гибок, чем в первом случае, но при этом требует от пользователя дополнительных действий, а так же усложняет пользовательский интерфейс, что может негативно отразиться на простоте пользования системой.

На мой взгляд, данный подход является предпочтительным для системы, нацеленной на использование для всех типов моделей.

6.3. Стратегии выбора зависимостей между масштабом, скоростью и задержкой

Наиболее сложным аспектом работы со временем является описание зависимости между масштабом, скоростью и задержкой. Ниже список разработанных стратегий:

1. Масштаб отсутствует. Скорость и задержка независимы

Рассмотрим общий случай формулы скорости:

$$V = \frac{[M]}{[sec]} \cdot V^*$$

,где V^* - коэффициент скорости, выбранный пользователем

, M - величина масштаба единиц симулятора

, V – скорость проигрывания

В общем случае в связи с тем, что размерность единиц времени симулятора не задана на этапе симуляции, мы не знаем величину масштаба.

$$V = V^*$$

Для того чтобы описать зависимость между скоростью, масштабом и задержкой можно использовать формулу времени проигрывания.

$$T = \sum_{i=0}^n (V_i \cdot \Delta t_i + d_i)$$

$$T = \sum_{i=0}^n (V_i^* \cdot \Delta t_i + d_i)$$

,где V_i скорость в i -ый момент времени

, Δt_i разница между следующим и текущим моментом времени

, d_i задержка в i -ый момент времени

, T время проигрывания

Формула времени не учитывает задержку на обработку состояний компьютером.

2. Масштаб отсутствует. Скорость и задержка зависимы

$$V = V^*$$

$$T = \sum_{i=0}^n V_i \cdot (\Delta t_i + d_i)$$

$$T = \sum_{i=0}^n V_i^* \cdot (\Delta t_i + d_i)$$

3. Масштаб и скорость зависимы, задержка независима

$$V = \frac{[M]}{[1]} \cdot V^*$$

$$T = \sum_{i=0}^n (V_i \cdot \Delta t_i + d_i)$$

$$T = \sum_{i=0}^n (M_i \cdot V_i^* \cdot \Delta t_i + d_i)$$

4. Масштаб и задержка зависимы, а скорость независима

$$V = V^*$$

$$T = \sum_{i=0}^n (V_i \cdot \Delta t_i + M_i \cdot d_i)$$

$$T = \sum_{i=0}^n (V_i^* \cdot \Delta t_i + M_i \cdot d_i)$$

5. Масштаб, задержка и скорость связаны

$$V = \frac{[M]}{[1]} \cdot V^*$$

$$T = \sum_{i=0}^n M_i \cdot V_i \cdot (\Delta t_i + d_i)$$

$$T = \sum_{i=0}^n M_i \cdot V_i^* \cdot (\Delta t_i + d_i)$$

6. Масштаба и задержки нет, есть только скорость

$$V = V^*$$

$$T = \sum_{i=0}^n (V_i \cdot \Delta t_i)$$

$$T = \sum_{i=0}^n V_i^* \cdot \Delta t_i$$

7. Скорость + масштаб, задержки нет

$$V = \frac{[M]}{[1]} \cdot V^*$$

$$T = \sum_{i=0}^n M_i \cdot V_i \cdot \Delta t_i$$

$$T = \sum_{i=0}^n M_i \cdot V_i^* \cdot \Delta t_i$$

8. Скорости и масштаб нет, только задержка

$$T = \sum_{i=0}^n d_i$$

9. Скорости нет, масштаб + задержка

$$T = \sum_{i=0}^n M_i \cdot d_i$$

6.4. Выбор универсальной стратегии управления временем

Результатом данного исследования является формирование стратегии управления временем, которая позволит работать с любым типом моделей. Хотя автор понимает, что использование специальной стратегии для каждого типа моделей будет куда более точно отражать потребности конечного пользователя, но в виде ограничения в данном вопросе выступает пользовательский интерфейс. Так как, если пользовательский интерфейс будет меняться, в зависимости от типа модели, это приведет к усложнению взаимодействия, что негативно скажется на удобстве пользования системой. Из предложенных вариантов работы со временем для разного типа моделей, наиболее подходящим является последний вариант со смешанным типом времен. В данной работе использовалась такая комбинация:

Применимые стратегии выбора зависимостей между масштабом, скоростью и задержкой (см. 6.3.):

1. Для одновременных событий показывается только последнее событие;
2. Настраиваемая задержка для одновременных событий;

Предлагается скомбинировать вариант один вариантом два. Система будет работать как в варианте 1 в случае если задержка равна 0. Во всех остальных случаях предлагается использовать вариант 2.

Применимые стратегии выбора зависимостей между масштабом, скоростью и задержкой (см. 6.3.):

Масштаб, задержка и скорость связаны;

Такой подход позволяет контролировать процесс больше других. И ,проведя несколько испытаний, оказалось, что достаточно удобно использовать.

7. Организационно-экономическая часть

7.1. Введение

В этой главе выполнена организационно-экономическая часть дипломного проекта, выполнен расчет затрат на разработку интегрированной среды разработки языка РДО. Для этого были произведены следующие расчёты:

- Оценка трудоёмкости разработки и внедрения программного обеспечения;
- Оценка состава и численности разработчиков программного обеспечения;
- Оценка трудоёмкости работ каждого участника проектной группы;
- Оценка стоимости разработки интегрированной среды разработки языка РДО.

Расчет действителен на 2-ой квартал 2016 года (цены на программное обеспечение, оборудование, расходные материалы, уровень заработной платы исполнителей и т.д.).

7.2. Организация и планирование процесса разработки ПП

В данном разделе дипломного проекта, посвященном экономическим вопросам, будет произведен расчет трудоемкости и стоимости разработки ПП. Целью данного расчета является определение затрат на разработку ПП.

Организация и планирование процесса разработки ПП предусматривает выполнение следующих работ:

- формирование состава выполняемых работ и группировка их по стадиям разработки;

- расчет трудоемкости выполнения работ;
- установление профессионального состава и расчет количества исполнителей;
- определение продолжительности выполнения отдельных этапов разработки;
- построение календарного графика выполнения разработки;
- контроль выполнения календарного графика.

Трудоемкость разработки ПП зависит от ряда факторов, рассмотрим эти факторы применительно к данной системе.

По степени новизны разрабатываемый программный продукт может быть отнесен к группе новизны «Б» (разработка ПП, не имеющей аналогов, в том числе разработка пакетов ПП).

По степени сложности алгоритма функционирования - к 1 группе сложности (реализующие оптимизационные и моделирующие алгоритмы)

По виду представления исходной информации ПП относится к группе 11 – исходная информация представлена в форме документов, имеющих различный формат и структуру.

По структуре выходной информации относим ПП к группе 22 - требуется вывод на печать одинаковых документов, вывод информационных носителей на машинные носители.

Ниже приведен перечень стадий и состава работ (Таблица 2.1) при разработке программной продукции.

Таблица 2.1. Перечень стадий и состав работ

Стадия разработки ПП	Состав выполняемых работ
Техническое задание	Постановка задач, выбор критериев эффективности. Разработка технико-экономического обоснования разработки. Определение состава пакета прикладных программ, состава и структуры информационной базы. Предварительный выбор методов выполнения работы. Разработка календарного плана выполнения работ.
Эскизный проект	Предварительная разработка структуры входных и выходных данных. Разработка общего описания алгоритмов реализации решения задач. Разработка пояснительной записки. Консультации разработчиков постановки задач. Согласование и утверждение эскизного проекта.
Технический проект	Разработка алгоритмов решения задач. Разработка пояснительной записки. Согласование и утверждение технического проекта. Разработка структуры программы. Разработка программной документации и передача ее для включения в технический проект. Уточнение структуры, анализ и определение формы представления входных и выходных данных. Выбор конфигурации технических средств.

Рабочий проект	Комплексная отладка задач и сдача в опытную эксплуатацию. Разработка проектной документации. Программирование и отладка программ. Описание контрольного примера. Разработка программной документации. Разработка, согласование программы и методики испытаний. Предварительное проведение всех видов испытаний.
Внедрение	Подготовка и передача программной документации для сопровождения с оформлением соответствующего Акта приема-сдачи работ. Проверка алгоритмов и программ решения задач, корректировка документации после опытной эксплуатации программного продукта.

7.2.1. Расчет трудоемкости разработки технического задания

Трудоемкость разработки технического задания рассчитывается по формуле: $t_{ТЗ} = t_{РЗ} + t_{РП}$,

где $t_{РЗ}$ - затраты времени разработчика постановки задачи на разработку ТЗ, чел.-дн.;

$t_{РП}$ - затраты времени разработчика ПП на разработку ТЗ, чел.-дн.

$$t_{РЗ} = t_3 \cdot K_{РЗ},$$

$$t_{РП} = t_3 \cdot K_{РП},$$

где t_3 - норма времени на разработку ТЗ на ПП, чел.-дн.

Исходя из рекомендаций [15] в данном случае $t_3 = 57$ чел.-дн. (Группа новизны ПП - Б; функциональное назначение ПП – реализует оптимизационные и моделирующие алгоритмы);

K_{P3} - коэффициент, учитывающий удельный вес трудоёмкости работ, выполняемых разработчиком постановки задачи на стадии ТЗ. В случае совместной с разработчиком ПО разработки: $K_{P3} = 0,65$;

K_{PP} - коэффициент, учитывающий удельный вес трудоемкости работ, выполняемых разработчиком ПП на стадии ТЗ. В случае совместной с разработчиком ПО разработки: $K_{PP} = 0,35$.

$$t_{T3} = 57 \cdot 0,65 + 57 \cdot 0,35 = 57 \text{ чел.-дн.}$$

7.2.2. Расчет трудоемкости выполнения эскизного проекта

Трудоемкость разработки эскизного проекта $t_{ЭП}$ рассчитывают по формуле:

$$t_{ЭП} = t_{P3} + t_{PP}, \text{ где:}$$

t_{P3} - затраты времени разработчика постановки задач на разработку ЭП, чел.-дни;

t_{PP} - затраты времени разработчика ПП на разработку ЭП, чел.-дни;

$$t_{P3} = t_{Э} \cdot K_{P3},$$

$$t_{PP} = t_{Э} \cdot K_{PP}$$

где:

$t_{Э}$ - норма времени на разработку ЭП программного продукта, чел.-дни.

$$t_{Э} = 117 \text{ чел.-дней};$$

K_{P3} - коэффициент, учитывающий удельный вес трудоемкости работ, выполняемых разработчиком постановки задачи на стадии ЭП. В случае совместной с разработчиком ПО разработки ЭП: $K_{P3} = 0,7$;

K_{PI} - коэффициент, учитывающий удельный вес трудоемкости работ, выполняемых разработчиком ПП на стадии ЭП. В случае совместной с разработчиком постановки задач разработки ЭП: $K_{PI} = 0,3$;

$$t_{ЭП} = 117 \cdot 0,7 + 117 \cdot 0,3 = 117 \text{ чел.-дней.}$$

7.2.3. Расчет трудоемкости выполнения технического проекта

Трудоемкость разработки технического проекта зависит от функционального назначения ПП, количества разновидностей форм входной и выходной информации и определяется как сумма времени, затраченного разработчиком постановки задач и разработчиком ПП, и определяется по формуле: $t_{ТП} = (t_{PЗ} + t_{PI}) \cdot K_B \cdot K_P$, где:

$t_{PЗ}$, t_{PI} - норма времени, затрачиваемого на разработку ТП разработчиком постановки задач и разработчиком ПО:

$$t_{PЗ} = 86 \text{ ч. ,}$$

$$t_{PI} = 23 \text{ ч.}$$

K_P - коэффициент учёта режима обработки информации:

$$K_P = 1.45$$

K_B - коэффициент учёта вида используемой информации, определяется из выражения:

$$K_B = \frac{K_{PI} \cdot N_{PI} + K_{НС} \cdot N_{НС} + K_B \cdot N_B}{N_{PI} + N_{НС} + N_B}, \text{ где:}$$

K_{PI} , $K_{НС}$, K_B - значения коэффициентов учёта вида используемой информации для переменной, нормативно-справочной информации и баз данных соответственно;

N_{PI} , $N_{НС}$, N_B - количество наборов данных для переменной, нормативно-справочной информации и баз данных соответственно.

Для группы новизны Б:

$$K_{II} = 1,2 \quad N_{II} = 2 \text{ j}$$

$$K_{HC} = 1,08 \quad N_{HC} = 1$$

$$K_B = 3,12 \quad N_B = 0$$

Таким образом
$$K_B = \frac{1,2 \cdot 2 + 1,08 \cdot 1 + 3,12 \cdot 0}{2 + 1 + 0} = 1,16$$

$$t_{TII} = (86 + 23) \cdot 1,16 \cdot 1,45 = 183,34 \text{ чел.-дней}$$

7.2.4. Расчет трудоемкости выполнения рабочего проекта

Трудоёмкость разработки рабочего проекта зависит от функционального назначения ПП, количества разновидностей форм входной и выходной информации, сложности алгоритма функционирования, сложности контроля информации, степени использования готовых программных модулей, уровня алгоритмического языка программирования и определяется по формуле:

$$t_{PII} = K_K \cdot K_P \cdot K_J \cdot K_3 \cdot K_{IIA} \cdot (t_{P3} + t_{PII}),$$

где:

K_P - коэффициент учёта режима обработки информации = 1.45;

K_K - коэффициент учёта сложности контроля информации = 1,07;

K_J - коэффициент учёта уровня алгоритмического языка программирования = 1,0;

K_3 - коэффициент учёта степени использования готовых программных модулей – 0.8;

(на 20-25% использования готовых программных модулей);

K_{IA} - коэффициент учёта вида используемой информации и сложности алгоритма ПП.

Значение коэффициента K_{IA} определяют из выражения:

$$K_{IA} = \frac{K'_П \cdot N_П + K'_{HC} \cdot N_{HC} + K'_Б \cdot N_Б}{N_П + N_{HC} + N_Б},$$

где:

$K'_П, K'_{HC}, K'_Б$ - значения коэффициентов учёта сложности алгоритма ПП и вида используемой информации для переменной, нормативно-справочной информации и баз данных соответственно.

В нашем случае присутствует только один вид используемой информации – переменной; соответствующее группе новизны Б значение коэффициента $K'_П = 1.62$ $K'_{HC} = 0,97$ $K'_Б = 0.81$

(группа новизны - Б)

$$K_{IA} = \frac{1,62 \cdot 2 + 0,97 \cdot 1 + 0,81 \cdot 0}{2 + 1 + 0} = 1,403$$

$t_{PЗ}$, $t_{PП}$ - норма времени, затраченного на разработку РП на алгоритмическом языке высокого уровня разработчиком постановки задач и разработчиком ПП.

$$t_{PЗ} = 28$$

$$t_{PП} = 156$$

$$t_{PП} = 1,16 \cdot 1,45 \cdot 1,0 \cdot 0,8 \cdot 1,403 \cdot (28 + 156) = 347,37 \text{ чел.-дней.}$$

7.2.5. Расчет трудоемкости выполнения внедрения

Трудоёмкость выполнения стадии внедрения может быть рассчитана по формуле:

$$t_B = (t_{PЗ} + t_{PП}) \cdot K_K \cdot K_P \cdot K_3,$$

где:

$t_{PЗ}$, $t_{PП}$ - норма времени, затраченного разработчиком постановки задач и разработчиком ПО на выполнение процедур внедрения ПП.

$$t_{PЗ} = 29$$

$$t_{PП} = 33$$

$$t_B = (29 + 33) \cdot 1,16 \cdot 1,45 \cdot 0,8 = 83,42 \text{ чел.-дней.}$$

7.2.6. Расчет суммарной трудоемкости

Таким образом, суммарная трудоемкость разработки программной продукции:

$$t_{III} = 57 + 117 + 183,34 + 347,37 + 83,42 = 788,13 \text{ чел.-дней.}$$

Трудоемкость этапов разработки программного продукта (Таблица 7.2.):

Таблица 7.2.

№пп	Стадия разработки	Трудоемкость чел.-дни
1	Техническое задание	57
2	Эскизный проект	117
3	Технический проект	183,34
4	Рабочий проект	347,37
5	Внедрение	83,42
Итого:		788,13

Для целей контроля и планирования выполнения работ в данном случае используем ленточный график, потому что разработку осуществляет

небольшой, стабильный по составу коллектив исполнителей . Для построения ленточного графика необходимо знать срок начала работ, срок окончания работ и количество работников, участвующих на каждом этапе разработки.

Продолжительность выполнения всех работ по этапам разработки программного продукта определяется из выражения

$$T_i = \frac{\tau_i + Q}{n_i} \quad , \text{ где}$$

τ_i - трудоемкость i -й работы, чел.-дни;

Q - трудоемкость дополнительных работ, выполняемых исполнителем, чел.-дни;

n_i - количество исполнителей, выполняющих i -ю работу, чел.

Пусть разработка системы на стадии разработки технического задания ведется одним специалистом, не привлекаемым к дополнительным работам, на стадии разработки эскизного проекта – тремя специалистами, на стадии разработки технического и рабочего проекта - тремя специалистами, а на стадии внедрения – двумя специалистами, то продолжительность разработки программного продукта:

$$T = \frac{57}{1} + \frac{117}{3} + \frac{183,34}{3} + \frac{347,37}{3} + \frac{83,42}{2} = 314,61 \text{ раб.дня.}$$

Для построения календарного плана необходимо перевести рабочие дни в календарные. Для этого длительность каждого этапа нужно разделить на поправочный коэффициент $K=0.7$. В результате получим:

$$T = \frac{314,61}{0.7} = 449,44 \text{ кал.дн}$$

$$T_{\text{ТЗ}} = \frac{57}{0.7} = 81,42 \text{ кал.дн}$$

$$T_{\text{Эп}} = \frac{117}{0,7 \cdot 3} = 55,71 \text{ кал.дн}$$

$$T_{\text{ТП}} = \frac{183,34}{0,7 \cdot 3} = 87,3 \text{ кал.дн}$$

$$T_{PP} = \frac{347,37}{0,7 \cdot 3} = 165,41 \text{ кал.дн}$$

$$T_{\text{в}} = \frac{83,42}{0,7 \cdot 2} = 59,58 \text{ кал.дн}$$

7.3. Определение стоимости разработки ПП

Для определения стоимости работ, необходимо на основании плановых сроков выполнения работ и численности исполнителей, рассчитать общую сумму затрат на разработку программного продукта.

Себестоимость ПП представляет собой стоимостную оценку используемых в процессе производства продукции работ, услуг, природных ресурсов, сырья материалов, топлива, энергии, основных фондов, трудовых ресурсов, а также других затрат на ее производство и реализацию.

Затраты, образующие себестоимость ПП, группируются в соответствии с их экономическим содержанием по следующим элементам:

- Расчёт основной заработной платы;
- Расчёт дополнительной заработной платы;
- Амортизация оборудования
- Отчисления в социальные фонды;
- Накладные расходы.

7.3.1. Расчет основной заработной платы

В статью включается основная заработная плата всех исполнителей, непосредственно занятых разработкой данного ПП, с учётом их должностного оклада и времени участия в разработке. Расчёт ведётся по

формуле:
$$C_{30} = \sum_i \frac{3_i}{d} \cdot \tau_i,$$

где 3_i - среднемесячный оклад i -го исполнителя, руб.; d – среднее количество рабочих дней в месяце; τ_i - трудоемкость работ, выполняемых i -м исполнителем, чел.-дни (определяется из календарного плана-графика).

$$Z_i = 32000 \text{ руб.};$$

$$d = 21;$$

$$\tau_i = 788,13$$

$$C_{zo} = \frac{32000}{21} \cdot 788,13 = 1200960 \text{ руб.}$$

7.3.2. Расчет дополнительной заработной платы

В статье учитываются все выплаты непосредственным исполнителям за время (установленное законодательством), непроработанное на производстве, в том числе: оплата очередных отпусков, компенсация за неиспользованный отпуск, оплата льготных часов подросткам и др. Расчет ведётся по формуле: $C_{zd} = C_{zo} \cdot \alpha_d$,

где α_d - коэффициент на дополнительную заработную плату.

$$\alpha_d = 0,2$$

$$C_{zd} = 1200960 \cdot 0,2 = 240192 \text{ руб.}$$

7.3.3. Отчисления на социальное страхование

В статье учитываются отчисления в бюджет социального страхования по установленному законодательному тарифу от суммы основной и дополнительной заработной платы, т.е.

$$C_{cc} = \alpha_{cc} \cdot (C_{zo} + C_{zd})$$

$$\alpha_d = 0,30 + 0,002 = 0,302$$

$$C_{zd} = (1200960 + 240192) \cdot 0,302 = 435228 \text{ руб.}$$

7.3.4. Накладные расходы

В статье учитываются затраты на общехозяйственные расходы, непроизводительные расходы и расходы на управление. Накладные

расходы определяют в процентном отношении к основной заработной плате, т.е.

$$C_n = C_{zo} \cdot \alpha_n,$$

$$\alpha_n = 1,8$$

$$C_{zo} = 1200960 \cdot 1,8 = 2161728 \text{ руб.}$$

7.3.5. Расходы на оборудование (амортизация)

В статье учитываются суммарные затраты на приобретение или проектирование специального оборудования:

$$C_{co} = \sum_i \frac{C_{bi} \cdot \alpha_i}{100 \cdot F_d} t_i, \text{ где}$$

C_{bi} - балансовая цена i -го вида оборудования, руб.;

α_i - норма годовых амортизационных отчислений для оборудования i -го вида, %;

t_i - время использования i -го вида оборудования при выполнении данной разработки, ч.

F_d - действительный годовой фонд рабочего времени сотрудника, ч. (5-ти дневная неделя, 8-и часовой рабочий день – 2080 ч.);

Табл. 7.3. Специальное оборудование и ПО, используемое при разработке

№ п/п	Наименование	Единица измерения	Количество	Цена за единицу	Сумма, руб
1	ПЭВМ	шт	1	35000	35000

Затраты на амортизацию оборудования (ПЭВМ):

$$C_{oo} = 35\,000 \cdot 20 \cdot 449,44 \cdot 8 / (100 \cdot 2080) = 12101 \text{ руб.}$$

7.3.6. Результаты расчетов затрат на разработку программного продукта

№ пп	Наименование статьи	Сметная стоимость, руб.
1	основная заработная плата	1200960
2	Дополнительная заработная плата	240192
3	Отчисления на социальное страхование	435228
4	Накладные расходы	2161728
5	Расходы на использование оборудования	12101
	Итого	3810017

7.4. Вывод

Произведенный расчет показал:

- Суммарная трудоемкость продукта составляет 788,13 чел.дней. Поэтому на каждом этапе необходимо привлечение нескольких специалистов.
- Длительность разработки программного продукта составляет 449,44 календарный день.
- Себестоимость программного продукта составляет 3810017 руб.

8. Мероприятия по охране труда и технике безопасности

8.1. Введение

В данном разделе дипломного проекта рассматриваются и анализируются опасные и вредные факторы при разработке и проектировании интерактивного проигрывателя моделей в среде имитационного моделирования Rao X.

Описываются мероприятия по обеспечению безопасности и безвредных условия труда, приводятся рекомендации по способам и методам ограничения действия вредных и исключению воздействия опасных факторов на студентов и преподавательский состав, проходящих и проводящих занятия в компьютерной сфере. При этом на человека, работающего в зоне действия ПЭВМ, влияет большое количество вредных факторов, состояние которых необходимо контролировать.

8.1.1. Анализ опасных и вредных факторов

Согласно ГОСТ 12.0.003-74 заполним таблицу 8.2.1.

Табл. 8.2.1. Опасные и вредные факторы

Фактор	опасный	вредный
1. физические		
1.1. повышенный уровень шума на рабочем месте	-	+

1.2. повышенный уровень статического электричества	-	+
1.3. повышенный уровень электромагнитных излучений	+	-
2. химические		
2.1 по пути проникания в организм человека через: органы дыхания; желудочно-кишечный тракт; кожные покровы и слизистые оболочки	-	+
3. биологические		
3.1 патогенные микроорганизмы (бактерии, вирусы, риккетсии, спирохеты, грибы, простейшие) и продукты их жизнедеятельности	-	+
4. психофизиологические		
4.1 нервно-психические перегрузки	-	+

8.2. Опасные и вредны факторы

8.2.1. Физические

8.2.1.1. Опасные факторы

Пожарная опасность, обусловленная наличием на рабочем месте мощного источника энергии. Поражение электротоком, обусловленное повышенным значением напряжения в электрической цепи, замыкание которой может произойти через тело человека.

8.2.1.2. Вредные факторы

Современные компьютеры становятся все более тихими, но в тоже время не все рабочие места оборудованы таковыми. С другой стороны в некоторых предприятиях устанавливают более мощные компьютеры-серверы. Даже при самой хорошей шумоизоляции полностью избавиться от шума на практике от шума избавиться не удастся. А так же иногда отделы разработки ПО находятся непосредственно на производстве. В силу чего все существующие на производстве источники шума так же пагубно влияют на программистов.

8.2.2. Химические

Повышенное содержание в воздухе рабочей зоны двуокиси углерода, озона, аммиака, фенола и формальдегидов негативно влияют на здоровье трудящегося.

8.2.2. Биологические

Могут привести к заболеванию или ухудшению состояния здоровья пользователя, относится повышенное содержание в воздухе патогенных микроорганизмов, особенно в помещении с большим количеством

работающих при недостаточной вентиляции, в период эпидемий.

8.2.3. Психофизиологические

Характерная при работе с ПЭВМ является такая физическая перегрузка, как длительное статическое напряжение мышц. Оно обусловлено вынужденным продолжительным сидением в одной и той же позе, часто неудобной, необходимостью постоянного наблюдения за экраном (напрягаются мышцы шеи, ухудшается мозговое кровообращение), набором большого количества знаков за рабочую смену (статическое перенапряжение мышц плечевого пояса и рук). При этом возникает также локальная динамическая перегрузка пальцев и кистей рук.

Статическим перенапряжениям мышц способствуют неудовлетворительные эргономические параметры рабочего места и его компонентов (отсутствие подлокотников, пюпитра, подставки для ног), отсутствие возможности регулировки параметров рабочего стула, высоты рабочей поверхности стола, неудобное расположение клавиатуры и дисплея, отсутствие регламентированных перерывов, невыполнение специальных упражнений для снятия напряжения и расслабления мышечных групп плечевого пояса, рук, шеи, спины, улучшения кровообращения.

8.2.4. Нервно-психические перегрузки

Являются следствием информационного взаимодействия в системе «пользователь – ПЭВМ». Они обусловлены неудовлетворительными условиями зрительного восприятия изображения, несогласованностью параметров информационных технологий с психофизиологическими возможностями человека, необходимостью постоянного наблюдения за информационными символами, быстрого анализа динамично меняющейся информации, принятия на его основе адекватных решений и реализации

соответствующих корректирующих воздействий. К основным нервно-психическим перегрузкам относятся повышенные зрительные напряжения; умственные и нервно-эмоциональные перегрузки; длительная концентрация внимания; монотонность труда (однообразие трудового процесса, повторяемость операций, отсутствие возможности переключения внимания или изменения вида работы).

8.3. Требования к помещениям для работы с ПЭВМ.

Помещения для эксплуатации ПЭВМ должны иметь естественное и искусственное освещение. Эксплуатация ПЭВМ в помещениях без естественного освещения допускается только при соответствующем обосновании и наличии положительного санитарно-эпидемиологического заключения, выданного в установленном порядке. Естественное и искусственное освещение должно соответствовать требованиям действующей нормативной документации. Окна в помещениях, где эксплуатируется вычислительная техника, преимущественно должны быть ориентированы на север и северо-восток. Оконные проемы должны быть оборудованы регулируемыми устройствами типа: жалюзи, занавесей, внешних козырьков и др. Не допускается размещение мест пользователей ПЭВМ во всех образовательных и культурно-развлекательных учреждениях для детей и подростков в цокольных и подвальных помещениях. Площадь на одно рабочее место пользователей ПЭВМ с ВДТ на базе электроннолучевой трубки (ЭЛТ) должна составлять не менее 6 м², в помещениях культурно-развлекательных учреждений и с ВДТ на базе плоских дискретных экранов (жидкокристаллические, плазменные) - 4,5 м².

При использовании ПЭВМ с ВДТ на базе ЭЛТ (без вспомогательных устройств - принтер, сканер и др.), отвечающих требованиям международных стандартов безопасности компьютеров, с продолжительностью работы менее 4-х часов в день допускается минимальная площадь 4,5 м² на одно рабочее место пользователя (взрослого и учащегося высшего профессионального образования).

Для внутренней отделки интерьера помещений, где расположены ПЭВМ, должны использоваться диффузно-отражающие материалы с коэффициентом отражения для потолка - 0,7 - 0,8; для стен - 0,5 - 0,6; для пола - 0,3 - 0,5. Полимерные материалы используются для внутренней отделки интерьера помещений с ПЭВМ при наличии санитарно-эпидемиологического заключения. Помещения, где размещаются рабочие места с ПЭВМ, должны быть оборудованы защитным заземлением (занулением) в соответствии с техническими требованиями по эксплуатации. Не следует размещать рабочие места с ПЭВМ вблизи силовых кабелей и вводов, высоковольтных трансформаторов, технологического оборудования, создающего помехи в работе ПЭВМ

8.4 Возможные негативные последствия и меры предосторожности

8.3.1. Короткое замыкание

Причины возникновения:

- В помещении установлены устройства, требующие электропитания от сети переменного тока с номинальным напряжением 220В, к ним проложена необходимая проводка. Максимальное опасное напряжение в цепи, замыкание которой может пройти через тело человека, составляет 220В.
- Устройства имеют металлические и пластиковые корпуса, которые являются потенциально опасными местами, в результате

соприкосновения с которыми человек может попасть в электрическую цепь.

Влияние на человека:

- Электрический ток, протекающий через организм человека, воздействует на него термически, электролитически и биологически
- Термическое действие характеризуется нагревом тканей, вплоть до ожог
- Электролитическое - разложением органических жидкостей, в том числе и крови;
- Биологическое действие электрического тока проявляется в нарушении биоэлектрических процессов и сопровождается раздражением и возбуждением живых тканей и сокращением мышц, в том числе - сердечной и мышц, ответственных за дыхание

Предлагаемые меры по обеспечению безопасности:

- Инструктаж сотрудников по организации безопасной работы с электроприборами;
- Создание регламента по обслуживанию электроприборов с целью своевременного определения ненадежных соединений и
- возможных мест замыкания на корпус;
- Оборудование защитного заземления/зануления;
- Размещение предупреждающих наклеек;
- Установка устройств защитного отключения.

8.3.6. Ухудшение зрения

Остановимся подробнее на недостаточной освещенности рабочей зоны помещения, где установлены ПЭВМ, а также на влиянии

повышенной яркости света, пониженной контрастности, прямой и обратной блёсткости и повышенной пульсации светового потока. При работе на ПЭВМ органы зрения пользователя выдерживают большую нагрузку с одновременным постоянным напряженным характером труда, что приводит к нарушению функционального состояния зрительного анализатора и центральной нервной системы.

Нарушение функционального состояния зрительного анализатора проявляется в снижении остроты зрения, устойчивости ясного видения, аккомодации, электрической чувствительности и лабильности.

Причинами нарушения функционального состояния зрительного анализатора являются:

- постоянная переадаптация органов зрения в условиях наличия в поле зрения объекта различения и фона различной яркости;
- недостаточная четкость и контрастность изображения на экране;
- срочность воспринимаемой информации;
- постоянные яркостные мелькания;
- наличие ярких пятен на клавиатуре и экране за счет отражения светового потока;
- большая разница между яркостью рабочей поверхности и яркостью окружающих предметов, наличие равноудаленных предметов;
- невысокое качество исходной информации на бумаге;
- неравномерная и недостаточная освещенность на рабочем месте. Наряду с перечисленными общепринятыми особенностями работы пользователя на рабочем месте ПЭВМ существуют особенности восприятия информации с экрана монитора.

Особенностями восприятия информации с экрана монитора органами зрения пользователя ПЭВМ являются следующие:

- экран монитора является источником света, на который в процессе работы непосредственно обращены органы зрения пользователя, что вводит оператора в другое психофизиологическое состояние;
- привязанность внимания пользователя к экрану монитора является причиной длительности неподвижности глазных и внутриглазных мышц, что приводит к их ослаблению;
- длительная и повышенная сосредоточенность органов зрения приводит к большим нагрузкам, а следовательно, к утомлению органов зрения, способствует возникновению близорукости, головной боли и раздраженности, нервного напряжения и стресса;
- длительная привязанность внимания пользователя к экрану монитора создает дискомфортное восприятие информации, в отличие от чтения обычной печатной информации;
- экран монитора является источником падающего светового потока на органы зрения пользователя, в отличие от обычной печатной информации, которая считывается за счет отраженного светового потока;
- информация на экране монитора периодически обновляется в процессе сканирования электронного луча по поверхности экрана и при низкой частоте происходит мерцание изображения, в отличие от неизменной информации на бумаге.

Для снижения нагрузки на органы зрения пользователя при работе на ПЭВМ необходимо соблюдать следующие условия зрительной работы.

При работе на ПЭВМ пользователь выполняет работу высокой точности, при минимальном размере объекта различения 0,3-0,5мм (толщина

символа на экране), разряда работы III, подразряда работы Г (экран - фон светлый, символ - объект различения - темный или наоборот).

Естественное боковое освещение должно составлять 2%, комбинированное искусственное освещение - 400 лк, при общем освещении - 200 лк.

8.3.9. Типовой расчет виброизоляции системы кондиционирования

В ходе обследования помещения, где происходит эксплуатация рассматриваемой системы, было обнаружено оборудование, производящее вибрации – кондиционер с частотой вращения вентилятора 700 оборотов в минуту. Проведем расчет жесткости виброизоляторов и их количества.

Оптимальное соотношение частоты вращения вентилятора в кондиционере и его собственной частоты определяется формулой (согласно ГОСТ 12.4.093-80) $\frac{f}{f_0} = 3$

Частота вращения вентилятора, рассчитывается по формуле:

$$f = \frac{n}{60} = \frac{700}{60} = 11.66667 \text{Гц}$$

Таким образом, собственная частота:

$$f_0 = \frac{f}{3} = \frac{11.66667}{3} = 3.8888 \text{Гц}$$

С другой стороны, собственная частота рассчитывается по формуле:

$$f_0 = \frac{1}{2 \cdot \pi} \cdot \sqrt{\frac{q_1 \cdot N}{m}}, \text{ где}$$

q_1 - вертикальная жесткость виброизолятора, Н/см

N - число виброизоляторов, штук

m - масса виброизолятора, кг

Выразим и рассчитаем соотношение $\frac{q_1 \cdot N}{m}$:

$$\frac{q_1 \cdot N}{m} = (2 \cdot \pi \cdot f_0)^2 = (2 \cdot 3.1415 \cdot 3.8888)^2 = 596.99$$

Параметры виброизолятора ДО-38:

Обозначение	Нагрузка Р, Н	Вертикальная жесткость Н/см		Высота h в свободном состоянии, мм	Осадка пружины, мм под нагрузкой, Н		Число рабочих витков пружины	Масса, кг
		Рабочая, Р _{раб}	Предельная, Р _{пр}		Р _{раб}	Р _{пр}		
ДО38	122	152	45	72	27	33,7	5,6	0,3

Таблица 8.3.9.

Для 4 опор-виброизоляторов, согласно таблице выше, выражение

$$\frac{q_1 \cdot N}{m} = \frac{45 \cdot 4}{0.3} = 600, \text{ собственная частота } f_0 = \frac{1}{2 \cdot \pi} \cdot \sqrt{\frac{q_1 \cdot N}{m}} = \frac{1}{2 \cdot 3.1415} \cdot \sqrt{600} = 3.8986 \text{ Гц}$$

Таким образом, соотношение частоты вращения вентилятора в кондиционере и его собственной частоты будет равно:

$$\frac{f}{f_0} = \frac{11.6667}{3.8986} = 2,9925$$

Это соотношение оптимально.

Таким образом, для устранения в помещении вибрации, создаваемых кондиционером, необходимо установить на 4 опоры-виброизоляторы марки ДО-38.

8.4. Утилизация ПЭВМ

Извлечение драгоценных металлов из вторичного сырья является частью проблемы использования возвратных ресурсов, которая включает в себя следующие аспекты: нормативно-правовой, организационный, сертификационный, технологический, экологический, экономико-финансовый. Проблема использования вторичного сырья, содержащего драгоценные материалы из компьютеров, периферийного оборудования и иных средств вычислительной техники (СВТ) актуальна в связи с техническим перевооружением отраслей промышленности.

К драгоценным металлам относятся: золото, серебро, платина, палладий, родий, иридий, рутений, осмий, а также любые химические соединения и сплавы каждого из этих металлов. Статья 2 п. 4

"Федерального закона о драгоценных металлах и драгоценных камнях" от 74 26 марта 1998 года №1463 гласит: "Лом и отходы драгоценных металлов подлежат сбору во всех организациях, в которых образуются указанные лом и отходы. Собранные лом и отходы подлежат обязательному учёту и могут перерабатываться собирающими их организациями для вторичного использования или реализовываться организациям, имеющим лицензии на данный вид деятельности, для дальнейшего производства и аффинажа драгоценных металлов".

Порядок учёта, хранения, транспортировки, инвентаризации, сбор и сдача отходов драгоценных металлов из СВТ, деталей и узлов, содержащих в своём составе драгоценные металлы для предприятия, учреждения и организации (далее - предприятие), независимо от форм собственности, установлен инструкцией Министерства финансов Российской Федерации от 4 августа 1992 года №67. Все виды работ с драгоценными металлами строго регламентированы нормативно-правовыми документами, перечень которых представлен в Приложении 1.

8.4.2. Разборка изделия

Последовательность разборки определяется типом изделия СВТ, его конструктивными особенностями и комплектацией. Как правило, процесс разборки должен выполняться в последовательности, обратной процессу сборки изделия. Основные направления деятельности на этапе «Разборка».

8.4.2.1. Разборка персональных компьютеров (ПЭВМ), рабочих станций и серверов

Технологии разборки ПЭВМ, рабочих станций, серверов и информационно-вычислительных систем едины поскольку состав их модулей стандартный. Он содержит системный блок и комплект периферийных устройств.

Разборку ПЭВМ и составных модулей целесообразно осуществлять по технологической схеме представленной на рисунке 8.4.2.1.

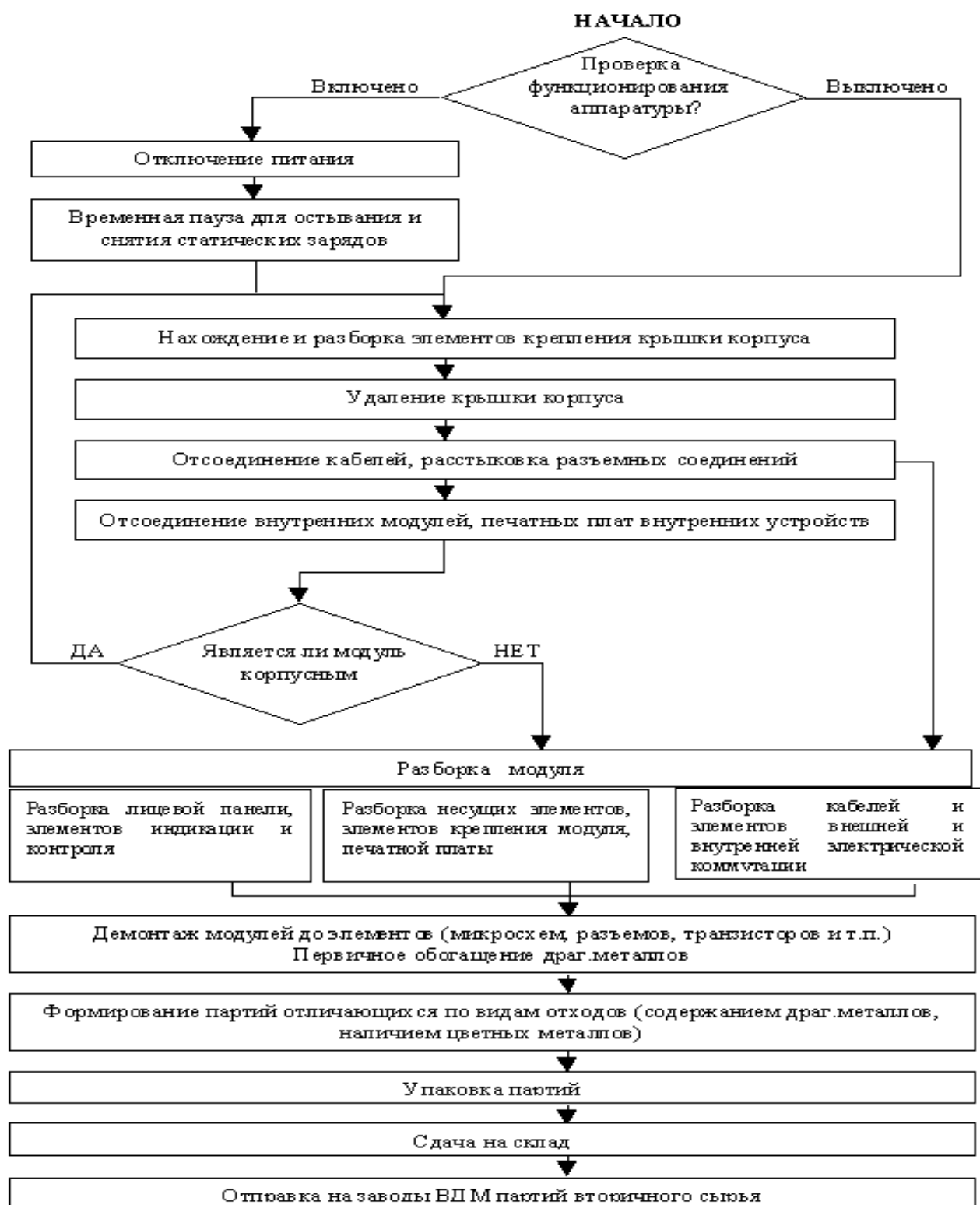


Рис. 8.4.2.1. Технологическая схема разборки

8.4.2.2. Обеспечение комплексности технологии разборки

Порядок разборки системного блока:

1. Выключить компьютер и отсоединить шнур питания от розетки и системного блока.
2. Отсоединить переходной шнур питания от системного блока к монитору.
3. Отсоединить от компьютера клавиатуру, монитор, манипулятор "мышь", принтер, сканер и иные внешние устройства.
4. Найти элементы крепления крышки корпуса (винты, шурупы, пружинные защелки и т.д.). Освободить крышку от элемента крепления.
5. Снять крышку.
6. Отсоединить внутренние кабели и плоские шлейфы.
7. Найти элементы крепления дисководов (НМД, НГМД) в отсеке для дисководов (винты, шурупы, саморезные винты, пружинные защелки и др.).
8. Освободить дисководы и извлечь их из дискового отсека.
9. Освободить от крепёжных элементов периферийные платы.
10. Извлечь из разъёмов непосредственного контактирования все периферийные платы.
11. Найти элементы крепления системной платы к корпусу (винты, шурупы). Освободить элементы крепления и извлечь системную плату из корпуса.
12. Извлечь модули памяти из разъёмов системной платы.
13. Найти элементы крепления блока питания к корпусу (винты, шурупы, саморезные винты, пружинные защелки и пр.).
14. Освободить элементы крепления и извлечь блок питания.

- 15.Разобрать блок питания и извлечь высоковольтные конденсаторы содержащие тантал.
- 16.Разобрать ПП и модули памяти до компонентов (микросхем, транзисторов, разъёмов и т.п.).
- 17.Произвести сортировку компонентов и сформировать партии электронного лома.
- 18.Упаковать партии, составить опись, произвести расчёт (анализ) драгметаллов и передать их на склад.
- 19.Провести сортировку цветных и чёрных металлов, пластмасс, сформировать партии и передать их на склад или на переработку.

При оценке содержания драгоценных металлов в партии электронного лома отечественных ПЭВМ необходимо руководствоваться паспортными данными.

При оценке ПЭВМ импортного производства необходимо провести ориентировочные расчёты по отечественным аналогам.

8.4.2.3. Извлечение вторичных чёрных металлов

Отечественная практика показывает, что на 1 г извлекаемого золота приходится около 1 кг лома чёрных металлов. В связи с высокой стоимостью транспортно-погрузочных работ рекомендуется производить отгрузку предприятиям-покупателям партий лома чёрных металлов весом не менее 10 тонн. Блоки, панели, съёмные кожухи, рамы, каркасы шкафов и стоек стационарных ЭВМ, изготовленные из стального нормализованного профиля или листа, подвергаются сортировке, набираются в партии и реализуются.

Предпочтительно заключение договоров при условии, когда предприятие-покупатель своим транспортом вывозит вторичные металлы с

территории предприятия-продавца. Крепёжные изделия, заготовки стального профиля, листов, вентиляторы, электропускатели, кнопки, электрический кабель направляются на реализацию непосредственно в торговую сеть. Опыт показывает, что денежные средства от реализации этих изделий не превышают 0,6 % от общей суммы.

8.4.2.4. Извлечение вторичных цветных металлов

В процессе разборки изделий СВТ образуется лом (содержащий медь) классификация которого должна проводиться по ГОСТ 1639.

В соответствии с ГОСТ 1639 медные шины целесообразно относить к классу А, группам I и II; латунь - к группам IV-VIII; бронзу - к группам XI-XII; отходы кабеля и проводов ПП следует относить к классу Г, группа XIII. Все виды ломов необходимо сортировать по классам и группам, формировать в партии и реализовывать. В процессе разборки изделий СВТ алюминий и его сплавы обычно содержатся в типовых конструкциях изделий. По ГОСТ 1639 их следует относить к классам АЗ и Б5. 79. Все виды отходов необходимо сортировать, формировать в партии и реализовывать. Свинцово-оловянные припои содержатся в печатных платах и их количество превышает количество золота в десятки раз. Припои регенерируются при переработке печатных плат.

При разборке СВТ танталовые конденсаторы необходимо складировать отдельно для последующей реализации. Переработка изделий из пластмасс

Пластмассы следует сортировать по видам. Переработке подлежат термопласты: поливинилхлорид, полиэтилен, полистирол и т.п.

Стёкла люминесцентных экранов электронно-лучевых трубок следует использовать в производстве керамики и в качестве сырья при производстве новых люминесцентных трубок.

8.4.3. Реализация партии

На рисунке ниже (см. Рисунок 8.4.3) представлены основные направления деятельности на этапе "Реализация партий".

Основные действия на этапе "Реализация партий" представляют собой последовательность действий, создающих основу для успешного

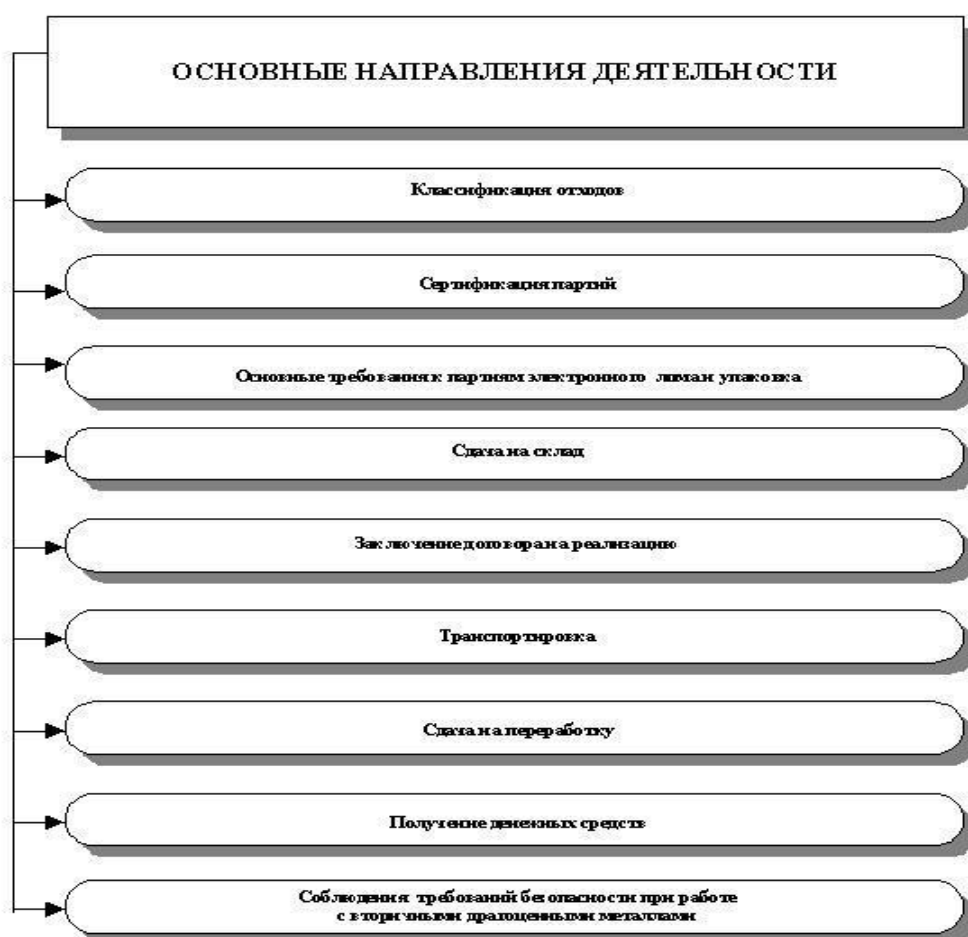


Рис. 8.4.3.. Направления деятельности на этапе «Реализация партии» ПЭВМ

выполнения процедур завершающего этапа утилизации СВТ.

8.4.3.1. Классификация отходов

В настоящее время в России и за рубежом не существует единой

классификации вторичного сырья, содержащего драгоценные металлы.

Поэтому, возможно разделение вторичного сырья по следующим признакам.

По содержанию драгоценных металлов:

- бедное (менее 1 % золота, 5 % серебра и 1 % металлов платиновой группы);
- богатое (более 1 % золота, 5 % серебра и 1 % металлов платиновой группы).

По составу материала основания:

- на металлической основе;
- на органической (пластиковой) основе;
- на керамической основе;
- на комбинированной основе.

По физическим признакам:

- твёрдые компактные отходы;
- сыпучие (порошки);
- жидкие.

Возможна классификация вторичного сырья в зависимости от сферы производства в:

- ювелирной промышленности;
- химической промышленности;

- электронной,
- электрохимической,
- оборонной,
- радиопромышленности (радиолампы, разъёмы, контакты, контактные устройства, платы на органической основе, микросхемы, радиодетали, кабели и провода, лента, высечка, вырубка, аккумуляторы, элементы питания, прочие отходы); бытовых отходах (лом бытовой радиоэлектронной аппаратуры, бытовой стеклянный и фарфоровый бой, лом ювелирных украшений и т.д.).

Отходы классифицируются по элементному составу.

При этом электронный лом отличается особым многообразием состава. Например, современный компьютерный лом содержит несколько десятков видов деталей, содержащих благородные, цветные и чёрные металлы.

8.4.3.2. Сертификация партий

В целях обеспечения строгого учёта, сохранности, сокращения потерь и эффективности использования драгоценных металлов, содержащихся в электронном ломе и отходах, а также для обеспечения единства и требуемой точности измерений при опробовании и проведении анализов руководствоваться химического состава, документами необходимо утверждёнными

Комитетом Российской Федерации по драгоценным металлам и драгоценным камням, Комитетом Российской Федерации по стандартизации, метрологии и сертификации: "Порядком выдачи сертификатов химического состава на партии электронного лома и

отходов, содержащих драгоценные металлы", "Временной методикой опробования электронного лома и отходов, содержащих драгоценные металлы".

Указанные документы определяют порядок проведения работ и требования по опробованию и сертификации химического состава партий электронного лома и отходов.

Сертификация химического состава электронного лома и отходов, содержащих драгоценные металлы, включает следующие работы:

- оформление и представление заявки в соответствующий орган
- по сертификации;
- создание комиссии по апробированию;
- апробирование, оформление документов по результатам
- апробирования, передача пробы на анализ;
- собственно анализ пробы и оформление количественного
- химического анализа;
- оформление сертификата.

Выполнение измерений химического состава проб (анализ) электронного лома и отходов, содержащих драгоценные металлы, следует производить методами количественного химического анализа по аттестованным методикам, Приложение 2.

Анализ проб осуществляется аналитическими лабораториями, которые аккредитованы Комитетом Российской Федерации по стандартизации, метрологии и сертификации, а также рекомендованными организациями и органами по сертификации.

По результатам анализа аккредитованная лаборатория оформляет

протокол измерений химического состава пробы электронного лома или отходов по установленной форме. По результатам процедур опробования и анализа химического состава электронного лома или отходов, орган по сертификации оформляет и выдает заявителю Сертификат химического состава на содержание драгоценных металлов установленной формы.

Сертификат химического состава и комплекс документов по опробованию сертифицируемой партии электронного лома включается в состав сопроводительной документации при передаче партии вторичного сырья от сдатчика заготовителю или переработчику, а также при вывозе за границу для переработки. При возникновении разногласий, в процессе передачи сертифицированных партий электронного лома и отходов от сдатчика заготовителю или переработчику, производится арбитраж. В этом случае партия не может быть передана переработчику до получения заключения арбитражной лаборатории, аккредитованной Комитетом Российской Федерации по стандартизации, метрологии и сертификации.

ЗАКЛЮЧЕНИЕ

В рамках данного дипломного проекта были получены следующие результаты.

Проведено предпроектное исследование подходов имитационного моделирования использованных в РДО, описаны основные механизмы работы со временем, проведено сравнение имитационной модели и видео.

Сформировано техническое задание на разработку подсистемы проигрывателя имитационных моделей в системе имитационного моделирования Rao X.

На этапе концептуального проектирования выявлены требования к элементам управления проигрывателя, проведен анализ и выбор архитектуры системы, описано понятие состояния модели.

На этапе технического проектирования разработан алгоритм работы проигрывателя имитационных моделей, выбран подходящий тип сериализации.

На этапе рабочего проектирования разработаны диаграммы классов, вариантов использования, написан программный код для реализации ранее разработанных алгоритмов, средства управления проигрыванием. Проведена исследовательская работа, в результате которой была выбрана универсальная стратегия управления временем.

В организационно-экономической части дипломного проекта определены трудоемкость и затраты на разработку подсистемы интерактивного проигрывателя имитационных моделей в системе Rao X.

В части, посвященной мероприятиям по охране труда и технике безопасности, проведен анализ опасных и вредных факторов, воздействующих на разработчика проигрывателя, выполнен типовой

расчет виброизоляции системы кондиционирования и выбран рациональный способ утилизации ПЭВМ. Таким образом, задание на выполнение дипломного проекта выполнено полностью, поставленная цель дипломного проекта достигнута в полном объеме.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Емельянов В.В., Ясиновский С.И. Введение в интеллектуальное имитационное моделирование сложных дискретных систем и процессов. Язык РДО. — М.: Анвик, 1998. - 427 с., ил. 136.
2. Документация по языку РДО [Электронный ресурс] — http://raox.ru/docs/reference/base_types_and_functions.html
3. Прицкер А. Введение в имитационное моделирование и язык СЛАМ II: Пер. с англ. — М.: Мир, 1987. — 646 с., ил.
4. Java Platform, Standart Edition 7. API Specification [Электронный ресурс] — <http://docs.oracle.com/javase/7/docs/api/>
5. Арсеньев В.В., Сажин Ю.Б. Методические указания к выполнению организационно-экономической части дипломных проектов по созданию программной продукции. — М.: Изд-во МГТУ, 1994. — 52 с. 6. Безопасность жизнедеятельности: Учебник для вузов/С.В. Белов, А.В. Ильницкая, А.Ф. Козьяков и др.; Под общ. ред. С.В. Белова. 7-е изд., стер. — М.: Высш.шк., 2007. — 616 с.: ил.
7. Охрана труда в машиностроении: Ученик для машиностроительных вузов/Е.Я. Юдин, С.В. Белов, С.К. Баланцев и др.; Под ред. Е.Я. Юдина, С.В. Белова — 2-е изд., перераб. и доп. — М.: Машиностроение. — 1983, 432 с., ил.
8. Утилизация пластмассовых отходов ПЭВМ [Электронный ресурс] — http://studopedia.ru/14_131224_utilizatsiya-plastmassovih-othodovpevm.html
9. Библиотека Gson [Электронный ресурс] — <https://github.com/google/gson>
10. Единая система программной документации. Техническое задание.

СПИСОК ИСПОЛЬЗОВАННОГО ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

1. Eclipse IDE for Java Developers Mars.2 Release (4.5.2)
2. openjdk version "1.8.0_40-internal"
3. UMLet v14.2
5. КОМПАС-3D V16
6. yEd Graph Editor v3.14.4
7. Microsoft® Office Word 2010
8. Microsoft® Office Excel 2010
9. Microsoft® Office Visio 2010
10. Foxit Reader

ПРИЛОЖЕНИЕ А

Пример сериализация массива объектов разных типов с использованием GSON

ПРИЛОЖЕНИЕ Б

Классы проигрыватель, читатель, писатель