TOMSK POLYTECHNIC UNIVERSITY ТОМСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

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

Инженерная школа информационных технологий и робототехники
Направление подготовки 15.04.04 «Автоматизация технологических процессов и производств»
Отделение автоматизации и робототехники

МАГИСТЕРСКАЯ ДИССЕРТАЦИЯ

Тема работы
Разработка системы планирования производственного процесса на основе математической модели

УДК 004.896:658.012.2:519.876

Студент

Группа	ФИО	Подпись	Дата
8ТМ91	Никитин Андрей Сергеевич		

Руководитель ВКР

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Доцент ОАР ИШИТР	Филипас Александр Александрович	к.т.н., доцент		

КОНСУЛЬТАНТЫ ПО РАЗДЕЛАМ:

По разделу «Финансовый менеджмент, ресурсоэффективность и ресурсосбережение»

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Доцент ОСГН ШБИП	Гончарова Наталья Александровна	к.э.н		

По разделу «Социальная ответственность»

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Доцент ООД ШБИП	Сечин Андрей Александрович	к.т.н		

ДОПУСТИТЬ К ЗАЩИТЕ:

Руководитель ООП	ФИО	Ученая степень, звание	Подпись	Дата
Доцент ОАР ИШИТР	Гайворонский Сергей Анатольевич	к.т.н., доцент		

Томск – 2021 г.

Планируемые результаты освоения ООП

15.04.04 – Автоматизация технологических процессов и производств
(Киберфизическая автоматизация технологических процессов и производств)

Код компетенции	Наименование компетенции
Универсальные компетенции	
УК(У)-1	Способен осуществлять критический анализ проблемных ситуаций на основе системного подхода, вырабатывать стратегию действий
УК(У)-2	Способен управлять проектом на всех этапах его жизненного цикла
УК(У)-3	Способен организовывать и руководить работой команды, вырабатывая командную стратегию для достижения поставленной цели
УК(У)-4	Способен применять современные коммуникативные технологии, в том числе на иностранном (-ых) языке (-ах), для академического и профессионального взаимодействия
УК(У)-5	Способен анализировать и учитывать разнообразие культур в процессе межкультурного взаимодействия
УК(У)-6	Способен определять и реализовывать приоритеты собственной
Общепрофессиональные компетенции	
ОПК(У)-1	Способен формулировать цели и задачи исследования, самостоятельно изучать научно-техническую документацию своей профессиональной деятельности
ОПК(У)-2	Способен определить математическую и техническую сущность задач и провести их качественно-количественный анализ
ОПК(У)-3	Способен на основании статистических методов участвовать в проведении корректирующих и превентивных мероприятий, направленных на улучшение качества, интерпретировать и представлять результаты
ОПК(У)-4	Способен анализировать полученные результаты измерений на основе их физической природы и принимать обоснованные решения в области профессиональной деятельности
Профессиональные компетенции выпускников	
ПК(У)-1	Обладает способностью разрабатывать технические задания на модернизацию и автоматизацию действующих производственных и технологических процессов и производств, технических средств и систем автоматизации, управления, контроля, диагностики и испытаний, новые виды продукции, автоматизированные и автоматические технологии ее производства, средства и системы автоматизации, управления процессами, жизненным циклом продукции и ее качеством;
ПК(У)-2	Обладает способностью проводить патентные исследования с целью обеспечения патентной чистоты и патентоспособности новых проектных решений и определения показателей технического уровня проектируемой продукции, автоматизированных и автоматических технологических процессов и производств, средств их технического и аппаратно-программного обеспечения;

ПК(У)-3	Обладает способностью: составлять описание принципов действия и конструкции устройств, проектируемых технических средств и систем автоматизации, управления, контроля, диагностики и испытаний технологических процессов и производств общепромышленного и специального назначения для различных отраслей национального хозяйства, проектировать их архитектурно-программные комплексы;
ПК(У)-4	Обладает способностью разрабатывать эскизные, технические и рабочие проекты автоматизированных и автоматических производств различного технологического и отраслевого назначения, технических средств и систем автоматизации управления, контроля, диагностики и испытаний, систем управления жизненным циклом продукции и ее качеством с использованием современных средств автоматизации проектирования, отечественного и зарубежного опыта разработки
ПК(У)-5	Обладает способностью разрабатывать функциональную, логическую и техническую организацию автоматизированных и автоматических производств, их элементов, технического, алгоритмического и программного обеспечения на базе современных методов, средств и технологий проектирования;

TOMSK POLYTECHNIC UNIVERSITY ТОМСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

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

Инженерная школа информационных технологий и робототехники
Направление подготовки 15.04.04 «Автоматизация технологических процессов и производств»
Отделение автоматизации и робототехники

УТВЕРЖДАЮ:
Руководитель ООП
_____ _____ Гайворонский С.А.
(Подпись) (Дата) (Ф.И.О.)

ЗАДАНИЕ
на выполнение выпускной квалификационной работы

В форме:

Магистерская диссертация

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

Студенту:

Группа	ФИО
8ТМ91	Никитин Андрей Сергеевич

Тема работы:

Разработка системы планирования производственного процесса на основе математической модели	
Утверждена приказом директора (дата, номер)	19.02.2021 № 50-14/с

Срок сдачи студентом выполненной работы:	01.06.21

ТЕХНИЧЕСКОЕ ЗАДАНИЕ:

Исходные данные к работе *(наименование объекта исследования или проектирования; производительность или нагрузка; режим работы (непрерывный, периодический, циклический и т. д.); вид сырья или материал изделия; требования к продукту, изделию или процессу; особые требования к особенностям функционирования (эксплуатации) объекта или изделия в плане безопасности эксплуатации, влияния на окружающую среду, энергозатратам; экономический анализ и т. д.).*	**Объектом исследования является многоэтапный процесс производства полупроводниковых пластин, представляющий из себя совокупность параллельных потоков.**

Перечень подлежащих исследованию, проектированию и разработке вопросов *(аналитический обзор по литературным источникам с целью выяснения достижений мировой науки техники в рассматриваемой области; постановка задачи исследования, проектирования, конструирования; содержание процедуры исследования, проектирования, конструирования; обсуждение результатов выполненной работы; наименование дополнительных разделов, подлежащих разработке; заключение по работе).*	**-Аналитический обзор методов моделирования производственных процессов;** **-Разработка и реализация программной имитационной модели производственных процессов;** **-Разработка и реализация подпрограммы управления данными;** **-Разработка и реализация человеко-машинного интерфейса приложения;** **-Социальная ответственность;** **-Экономическое обоснование.**
Перечень графического материала *(с точным указанием обязательных чертежей)*	

Консультанты по разделам выпускной квалификационной работы
(с указанием разделов)

Раздел	Консультант
Финансовый менеджмент, ресурсоэффективность и ресурсосбережение	**Гончарова Наталья Александровна**
Социальная ответственность	**Сечин Андрей Александрович**

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

Введение, Описание разработки, Модель производственных процессов, Типы обработки изделий, Алгоритмизация моделирования, Алгоритм моделирования.

Дата выдачи задания на выполнение выпускной квалификационной работы по линейному графику	01.03.21

Задание выдал руководитель / консультант (при наличии):

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Доцент ОАР ИШИТР	Филипас Александр Александрович	к.т.н., доцент.		

Задание принял к исполнению студент:

Группа	ФИО	Подпись	Дата
8ТМ91	Никитин Андрей Сергеевич		

TOMSK POLYTECHNIC UNIVERSITY ТОМСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

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

Школа– инженерная школа информационных технологий и робототехники
Направление подготовки - 15.04.04 Автоматизация технологических процессов и производств
Отделение школы (НОЦ) - Отделение автоматизации и робототехники
Период выполнения _____(осенний / весенний семестр 2020 /2021 учебного года)

Форма представления работы:

Магистерская диссертация

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

КАЛЕНДАРНЫЙ РЕЙТИНГ-ПЛАН
выполнения выпускной квалификационной работы

Срок сдачи студентом выполненной работы:	01.06.21

Дата контроля	Название раздела (модуля) / вид работы (исследования)	Максимальный балл раздела (модуля)
01.06.21	*Основная часть*	*65*
01.06.21	*Финансовый менеджмент*	*15*
01.06.21	*Социальная ответственность*	*10*

СОСТАВИЛ:
Руководитель ВКР

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Доцент ОАР ИШИТР	Филипас Александр Александрович	к.т.н., доцент.		

СОГЛАСОВАНО:
Руководитель ООП

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Доцент ОАР ИШИТР	Гайворонский Сергей Анатольевич	к.т.н., доцент.		

ЗАДАНИЕ ДЛЯ РАЗДЕЛА
«ФИНАНСОВЫЙ МЕНЕДЖМЕНТ, РЕСУРСОЭФФЕКТИВНОСТЬ И РЕСУРСОСБЕРЕЖЕНИЕ»

Студенту:

Группа	ФИО
8ТМ91	Никитин Андрей Сергеевич

Школа	Инженерная школа информационных технологий и робототехники	**Отделение школы (НОЦ)**	Отделение автоматизации и робототехники
Уровень образования	Магистр	**Направление/специальность**	15.04.04 Автоматизация технологических процессов и производств

Исходные данные к разделу «Финансовый менеджмент, ресурсоэффективность и ресурсосбережение»:	
Работа с информацией, представленной в российских и иностранных научных публикациях, аналитических материалах, статистических бюллетенях и изданиях, нормативно-правовых документах.	
Перечень вопросов, подлежащих исследованию, проектированию и разработке:	
1. *Оценка коммерческого и инновационного потенциала НТИ*	*Анализ потенциальных потребителей результатов исследования; Анализ конкурентных технических решений; FAST анализ; SWOT анализ; Оценка готовности проекта к коммерциализации;*
2. *Разработка устава научно-технического проекта*	*Цели и результатов проекта; Организационная структура проекта;*
3. *Планирование процесса управления НТИ: структура и график проведения, бюджет, риски и организация закупок*	*Структура работ; План проекта; Бюджет научного исследования; Риски проекта;*
4. *Определение ресурсной, финансовой, экономической эффективности*	*Абсолютная эффективность исследования; Сравнительная эффективность исследования;*
Перечень графического материала *(с точным указанием обязательных чертежей):*	
1. *Сегментирование рынка* 2. *Оценка конкурентоспособности технических решений* 3. *Матрица SWOT* 4. *График проведения и бюджет НТИ* 5. *Оценка ресурсной, финансовой и экономической эффективности НТИ* 6. *Потенциальные риски*	

Дата выдачи задания для раздела по линейному графику	01.03.21

Задание выдал консультант:

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Доцент ОСГН ШБИП	Гончарова Наталья Александровна	к.э.н		

Задание принял к исполнению студент:

Группа	ФИО	Подпись	Дата
8ТМ91	Никитин Андрей Сергеевич		

ЗАДАНИЕ ДЛЯ РАЗДЕЛА

«СОЦИАЛЬНАЯ ОТВЕТСТВЕННОСТЬ»

Студенту:

Группа	**ФИО**
8ТМ91	Никитин Андрей Сергеевич

Школа	Инженерная школа информационных технологий и робототехники	**Отделение школы (НОЦ)**	Отделение автоматизации и робототехники
Уровень образования	Магистр	**Направление/специальность**	15.04.04 Автоматизация технологических процессов и производств

Тема ВКР:

Разработка системы планирования производственного процесса на основе математической модели.	
Исходные данные к разделу «Социальная ответственность»:	
1. Характеристика объекта исследования (вещество, материал, прибор, алгоритм, методика, рабочая зона) и области его применения	Создание математической модели для комплексной многокритериальной оптимизации производственного процесса, с учётом особенностей функционирования и ограничений.
Перечень вопросов, подлежащих исследованию, проектированию и разработке:	
1. Правовые и организационные вопросы обеспечения безопасности: – специальные (характерные при эксплуатации объекта исследования, проектируемой рабочей зоны) правовые нормы трудового законодательства; – организационные мероприятия при компоновке рабочей зоны.	Рабочее место при выполнении работ в положении сидя должно соответствовать требованиям ГОСТ 12.2.032-78. Требования к организации оборудования рабочих мест с ПК регулируется в СанПиН 2.2.2/2.4.1340-03. Трудовой кодекс Российской Федерации от 30.12.2001 N 197-ФЗ (ред. от 05.02.2018)
2. Производственная безопасность 2.1. Анализ выявленных вредных и опасных факторов 2.2. Обоснование мероприятий по снижению воздействия	Анализ выявленных вредных факторов: – недостаточная освещённость рабочей зоны; отсутствие или недостаток естественного света; – повышенный уровень шума; – повышенный уровень электромагнитных излучений; – повышенная напряжённость электрического поля; – повышенная или пониженная влажность воздуха; – статические перегрузки; – умственные перегрузки, перегрузки анализаторов; Анализ выявленных опасных факторов:

	– электрический ток (источником является ПК); – короткое замыкание; – статическое электричество;
3. Экологическая безопасность:	Воздействие объекта на атмосферу, гидросферу не происходит. В работе проведён анализ воздействия на литосферу (образование отходов при выходе из строя ПК, возникновения отходов при печати и утилизации ламп).
4. Безопасность в чрезвычайных ситуациях:	В рабочем помещении, возможно, ЧС техногенного характера – пожар (возгорание).

Дата выдачи задания для раздела по линейному графику	01.03.21

Задание выдал консультант:

Должность	**ФИО**	**Ученая степень, звание**	**Подпись**	**Дата**
Доцент ООД ШБИП	Сечин Андрей Александрович	к.т.н		

Задание принял к исполнению студент:

Группа	**ФИО**	**Подпись**	**Дата**
8ТМ91	Никитин Андрей Сергеевич		

Реферат

Пояснительная записка выполнена на 99 листах машинописного текста, содержит 13 иллюстраций, 27 таблиц, 25 источников, 7 приложений.

Ключевые слова: МОДЕЛИРОВАНИЕ ПРОИЗВОДСТВЕННОГО ПРОЦЕССА, СИСТЕМА ПОДДЕРЖКИ ПРИНЯТИЯ РЕШЕНИЙ, ИМИТАЦИОННОЕ МОДЕЛИРОВАНИЕ, МАТЕМАТИЧЕСКОЕ МОДЕЛИРОВАНИЕ, ПЛАНИРОВАНИЕ ПРОИЗВОДСТВА.

Цель работы: разработка системы СППР планирования производственного процесса, на основе математической модели.

В ходе работы была создана система моделирования дискретно событийных производственных процессов. Реализованы библиотека моделирования система хранения и обеспечения доступа к данным, веб приложение для работы с пользователем.

Данная работа является продолжением работы «Разработка и реализация алгоритма планирования производственного процесса» [10] и открывает перспективу объединения результатов этих работ для создания комплексной системы планирования производственных процессов.

Область применения: планирование загрузки производственных мощностей малых и средних промышленных предприятий.

Оглавление

Введение

В управлении предприятием необходимо принимать управленческие решения различного характера. Зачастую их долгосрочный эффект и методы его предсказания не очевидны. Для обоснованного принятия решений используется инструменты и методы исследования процессов и прогнозирования

- оценка рисков;
- методы статистического анализа;
- моделирование процессов.

Моделирование является одним из наиболее перспективных методов, обладающим рядом преимуществ. Позволяет исследовать системы и процессы без влияния на них. Позволяет использовать модульный подход в исследовании, т.е. вычленять части систем и исследовать их по отдельности или наоборот исследовать взаимодействие нескольких систем. Возможность получить дополнительные данные об объекте в ходе исследования его модели [1].

Наиболее применимым методом моделирования в исследовании промышленных производств является математическое моделирование. Оно хорошо поддаётся программированию и вычислению в ЭВМ, что позволяет автоматизировать этот процесс и создавать системы поддержки принятия решений (СППР).

СППР помогают ответственным сотрудникам провести анализ заданной ситуации, рассчитать и проанализировать последствия принятия решения. Важно понимать, что СППР не управляет объектом и никак на него не влияет.

В СППР применяется множество технологий и методов, как по отдельности, так и совокупно:

– базы знаний и алгоритмы поиска;

– интеллектуальный анализ данных;

– генетические алгоритмы;

– нейронные сети;

– имитационное моделирование.

Существует два вида СППР. Пассивная предлагает эксперту набор информации, который помогает принять решение, но не может выдвинуть собственного решения. Активная может на основе данных выдвинуть собственное решение, некоторые системы поддерживают возможность последующей корректировки решения экспертом и проверки скорректированного решения, пока эксперт не одобрит результаты.

Одно из повседневно принимаемых решений на промышленном предприятии является планирование цепей поставок и загрузки производственных мощностей (производственное расписание). Существуют методики расчёта оптимального производственного расписания, причём как управленческие концепции: бережливое производство, шесть сигм и т.п. Так и прикладные математические дисциплины: исследование операций и методы оптимизации, теория массового обслуживания и т.д.

Проведение таких расчётов «вручную» (если они вообще проводятся) приводит к ухудшению эффективности производства, замедленному наращиванию объёмов производства, при увеличении мощностей, и большему времени устранения нештатных ситуаций.

Решением этих проблем может быть внедрение СППР планирования цепей поставок. Большинство современных предприятий уже используют АСУП [2,3], но они не имеют функционала планирования. А специализированные решения имеют стоимость не позволительную для многих малых и средних предприятий, к тому же требуют постоянной поддержки или наем сотрудников соответствующей квалификации.

Цель работы: разработка системы СППР планирования производственного процесса, на основе математической модели.

В ходе данной работы реализована среда имитационного моделирования производственных процессов. Разработка предназначена для

станочных предприятий различной степени автоматизации, имеет следующие возможности:

– построение моделей с различным количеством обрабатывающих машин;

– три различных режима обработки партий и возможность их расширения;

– отработка заданных расписаний загрузки производства и генерация выходных сценариев;

– отработка планов на различных моделях, для моделирования различных условий и нештатных ситуаций.

Разработка выполнена в виде модульного, клиент-серверного приложения. Авторизованный пользователь осуществляет работу браузере, установка на клиентском компьютере не требуется. Приложение закодировано на ЯП C# и C++, в среде разработки Visual Studio Community 2019.

1 Обзор литературы

Принятие любого решения при управлении предприятием имеет три составляющие:

- задача;
- альтернативы (способы решения задачи);
- критерии отбора альтернатив.

Обоснованное принятие решений в управлении предприятием опирается на методологию исследования операций. Методология исследования операций включает в себя следующие этапы [4]:

1. формализация задачи – описание задачи математическим языком;
2. построение математической модели на основе формализованной задачи;
3. решение модели;
4. валидация модели – проверка на соответствия поведения модели поведению реального объекта в разных ситуациях;
5. реализация решения – применение результатов решения модели на реальных объектах.

Существуют три метода формального описания систем [5]:

Аналитические – отображение свойств многомерной и многосвязной системы в *n*-мерном пространстве в виде точки, совершающей какое-либо движение в нём. Отображение осуществляется посредством оператора (функции, функционала) $\Phi[S_x]$.

Статистические – отображение системы с помощью случайных величин, которые описываются вероятностными характеристиками и статистическими закономерностями.

Дискретные – включает в себя описание систем с помощью множеств и их отношений, логических функций / выражений и алгебры логики, с помощью сетей и графов.

Моделирование – изучение характеристик и свойств объекта-оригинала, посредством изучения объекта-заменителя, воспроизводящего оригинал с некоторой точностью. В исследовании операций применяется математическое моделирование – метод, в котором моделью объекта является его описание в математических выражениях, исследование модели производится с использованием математических методов.

1.1 Методы решения математических моделей

Аналитическое моделирование – описание системы совокупность дифференциальных уравнений, решение которых даёт формулу, устанавливающую какие параметры и как влияют на систему.

Постановка формализованной задачи аналитического моделирования и решение модели содержит в себе следующие этапы [6]:

1) установить границы объекта моделирования – в рамках данного этапа необходимо определить интересующие нас свойства объекта и отделить их от не относящихся к данной задаче, для упрощения модели объекта.

2) построение математической модели системы. Структура модели включает в себя уравнения, описывающие процессы в объекте исследования, их дополняют неравенства, определяющие область допустимых значений независимых параметров, что показывает границы возможных изменений характеристик объекта.

3) выбор критерия оптимизации. В зависимости от решаемой задачи имеет различный характер (экономический, надёжностный, точностный и др.). При этом лучшему решению всегда соответствует минимум или максимум критерия, в зависимости от задачи.

4) формирование целевой функции – в этом этапе на основании выбранного критерия составляют целевую функцию, отражающую зависимость критерия от параметров объекта. Таким образом задача оптимизации сводится к нахождению экстремума целевой функции [6,7].

5) построение алгоритма оптимизации и решение экстремальной задачи – в данном этапе производятся выбор и разработка алгоритмов решения для реализации на вычислительной технике.

Таким образом, сама по себе задача оптимизации выглядит следующим образом: заданы множество X и функция $f(x)$, определённая на X, требуется найти точки максимума или минимума f на X [7]. Задачу можно записать в следующем виде:

$$\begin{gathered} f(x) \to \min,\ x \in X; \\ h_k(x) = 0,\ k = \overline{1,K}; \\ g_j(x) \geq 0,\ j = \overline{1,J}; \\ X \subset R^n. \end{gathered} \tag{1.1}$$

где f – целевая функция;

X – множество допустимых значений параметров;

h_k – ограничения первого рода, заданные в виде равенств;

K – размерность вектора ограничений первого рода;

g_j – ограничения второго рода, заданные в виде неравенств;

J – размерность вектора ограничений второго рода.

Для решения данных моделей используются методы математического программирования – численные методы решения многомерных экстремальных задач в ограничениях. По характеру задачи методы их решения принято классифицировать [8]:

– линейное программирование – целевая функция $f(x)$ и все ограничения $h_k(x)$ и $g_j(x)$ линейны. Такие модели применяются для построения оптимальных производственных расписаний и планировании грузопотоков;

– нелинейное программирование – если целевая функция $f(x)$ или хотя бы одна из функций $h_k(x)$ и $g_j(x)$ нелинейны. Обычно такие модели применяют в распределении ограниченных ресурсов;

– целочисленное программирование – на некоторые *x* наложено условие целочисленности, применяется в планировании комплектных и контейнерных поставок и планировании маршрута;

– Динамическое программирование – если ограничения зависят от времени, т.е. приобретают вид:

$$\begin{aligned} h_k(x,t) = 0,\ k = \overline{1,K}; \\ g_j(x,t) \geq 0,\ j = \overline{1,J}; \end{aligned} \tag{1.2}$$

Или целевая функция приобретает аддитивный:

$$f(x) = \sum_{i=1}^{n} f_i(x_i); \tag{1.3}$$

либо мультипликативный характер:

$$f(x) = \prod_{i=1}^{m} f_i(x_i); \tag{1.4}$$

Методами динамического программирования решаются задачи планировании поставок, запасов, производства в условиях изменяющегося спроса.

Стохастическое моделирование – закономерности в моделях описываются набором дифференциальных уравнений, но некоторые величины в модели имеют случайных характер. Существуют методы решения таких задач:

Стохастическое программирование – некоторые *x* имеют вероятностный характер, применяется в решении задач распределения транспортных потоков и планирования производства в условиях случайного спроса [8].

Теория массового обслуживания – модель состоит из модели обслуживания (обычно аналитическая) и стохастической мокли очереди запросов на обслуживание [4].

1.2 Имитационное моделирование

Зачастую описать всю систему аналитически требует слишком много трудозатрат или вообще не представляется возможность. В этих случаях

прибегают к имитационному моделированию – моделирование системы, путём логико-арифметического описания поведения отдельных частей системы и правил их взаимодействия, отображающих последовательность событий в моделируемой системе. Такое моделирование обычно проводится на ЭВМ Результаты моделирования выражаются во временных диаграммах событий. Существует несколько методов построения временных диаграмм [9].

В первом методе каждая часть системы моделируется отдельно и по окончанию моделирования события из всех частей системы сводятся в диаграммы по типам событий. Такой подход сопряжен с большими затратами оперативной памяти.

Вторым методом является пошаговое моделирование, состоит в определении глобального времени модели и шаге Δt его изменяя. В каждый момент времени Δt_1, необходимо рассчитать какие события произошли в предыдущем интервале Δt_{1-1}. Таким образом сокращается расходование памяти, но возникает другая проблема. Каким образом выбрать интервал Δt, с одной стороны, чем он меньше, тем точнее моделируется система, с другой, чем он больше, тем меньше затрат памяти и процессорной мощности.

Для решения этой проблемы применяется моделирование с переменным шагом и принцип «продвижения модельного времени до ближайшего события. По этому принципу, для всех процессов, параллельно протекающих в модели, в каждый останов модели рассчитывается время наступления ближайшего события в будущем. И модельное время продвигается до наступления этого события [9].

2 Описание разработки

В ходе данной работы реализована среда имитационного моделирования производственных процессов. В целях повышения мобильности, облегчения тестирования и обнаружения ошибок и контроля поведения программы, в разработка был принят модульный подход. Разработка состоит из трёх модулей (рис. 1):

1. Библиотека моделирования производственных процессов;
2. Модуль хранения и доступа к данным;
3. Клиент-серверный модуль

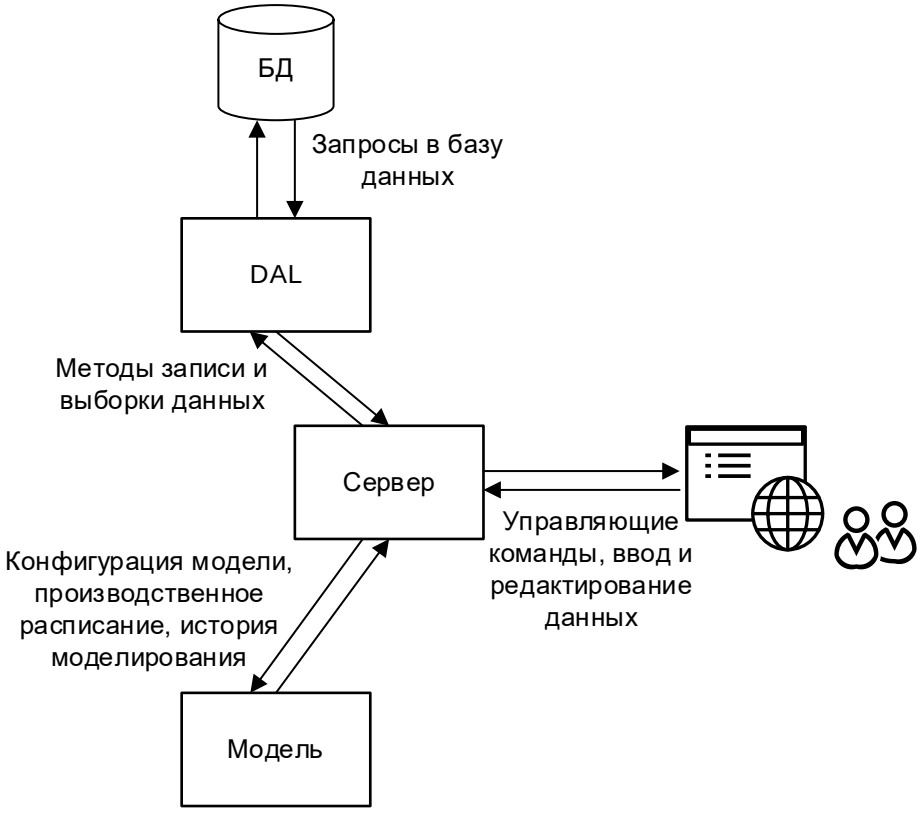


Рисунок 1 – Модульная структура программы

Первый модуль выполняет функции моделирования процесса обработки изделий, в качестве входных данных он получает конфигурацию модели и производственный план, выраженный в машиночитаемом коде (XML файл). В качестве выходных данных, программа отдаёт «историю» моделирования, также выраженную в машиночитаемом коде. Запрограммирован данный модуль в виде динамически подключаемой

библиотеки, в работе его принято называть «модель производственных процессов» или «модель».

Второй модуль обеспечивает связь приложения с БД, реализует методы сохранения данных в базе и их выборки из базы. Этот модуль принято называть – слой доступа к данными или DAL (от англ. Data Access Layer).

Третий модуль отвечает за взаимодействие с пользователем и управление потоками данных. Модуль выполнен в виде клиент-серверного приложения, функционал двух предыдущих модулей инкапсулируется в данном модуле, в работе его принято называть «сервер». Пользовать работает с приложением из своего браузера. После авторизации пользователь может запросить данные по предыдущим моделям и сервер вызовет необходимые методы из DAL или вызовет методы сохранения данных, при создании пользователем новой модели или изменении существующей. После конфигурирования модели, пользователь может загрузить разработанный вручную, в предыдущих разработках [10] или в аналогичных продуктах, производственный план и вызвать моделирование этого расписания. Сервер запрашивает данные по модели в DAL, предаёт их в модель и по окончании моделирования данные передаются на экран управления пользователю и по желанию пользователя сохраняются в базе, вызовом метода в DAL.

2.1 Модель производственных процессов

Модель реализует пошаговое производственных процессов и представляет из себя чёрный ящик. В качестве входных данных модель получает производственное расписание и данные о партиях изделий. На выход модель генерирует историю для каждой машины – какие партии, в какое время через неё прошли, и каждой партии – через какие машины пройдена обработка и сколько времени она занимала.

Всю модель производства, можно разделить на отдельные модели каждой производственной установки. Модель установки также представляет чёрный ящик. На вход модель получает данные о текущей партии на обработку

и рецепт обработки. На выходе модель установки отдаёт время обработки партии. Функции и правила преобразования входных данных в выходные определяются на этапе проектирования. На данный момент в модели реализовано три типа обработки изделий.

2.1.1 Типы обработки изделий

Потоковая обработка, изделия обрабатываются непрерывно конвейерным методом одно за другим. В объекте исследования характерным примером такого типа обработки является обработка изделий по очереди на ЧПУ станке.

Групповая обработка, обрабатываются несколько изделий одновременно, но не вся партия целиком. Характерный пример, химическая обработка нескольких изделий в ваннах с реагентами.

Обработка группой партий, отличается от групповой обработки тем, что количество обрабатываемых изделий ограничено несколькими партиями. Пример на производстве, обработка нескольких партий изделий в печах.

2.1.2 Математическое моделирование типов обработки

Время обработки партии в установке определяется формулой:

$$T = f(n,r). \tag{2.5}$$

где n – количество изделий в партии;

r – рецепт обработки;

$f(n,r)$ – функция времени.

Формула может меняться в зависимости от необходимости перенастройки рецепта. Большинство производственных установок могут выполнять обработку различных типов изделий (несколько «рецептов» обработки), и перенастройка с одного рецепта на другой требует некоторого времени, соответственно формула (2.5) приобретает вид:

$$T = f(n,r) + t_0 . \quad (2.6)$$

где t_0 – константа времени на перенастройку машины.

При потоковой обработке, время обработки всей партии складывается из времени времён обработки каждого изделия. Функция $f(n,r)$ приобретает следующий вид:

$$f(n,r) = n \cdot r . \quad (2.7)$$

Здесь r имеет значение константы времени обработки рецепта, т.е. время, которое установка тратит на обработку одного изделия.

При групповой обработке время обработки партии кратно количеству обрабатываемых пластин за раз. Функция $f(n,r)$ приобретает следующий вид:

$$f(n,r) = \left\lceil \frac{n}{m} \right\rceil \cdot r . \quad (2.8)$$

где m – константа, количества обрабатываемых пластин за раз;

r – константа времени обработки рецепта, т.е. время, которое установка тратит на обработку одной группы изделий.

Частное количества пластин в партии и константы количества округляется в большую сторону.

При обработке группой партий, не имеет значения, какое именно количество пластин и партий (если оно не превышает максимально возможное). Время обработки в машине представляет собой константу и определяется только текущим рецептом обработки. Функция $f(n,r)$ редуцируется до следующего вида.

$$f(n,r) = r . \quad (2.9)$$

где r – константа времени обработки рецепта, т.е. время, которое установка тратит на обработку загруженных в неё партий.

2.1.3 Алгоритмизация моделирования

Модель производственных процессов разработана в парадигме объектно-ориентированного программирования (ООП). Основополагающей составляющей ООП является классы и объекты.

Классы представляет собой описание структуры объекта и его поведения, в ходе работы приложения создаются объекты, которые определяются классом. Каждый класс содержит атрибуты (поля) и методы (или только атрибуты или методы).

Атрибут – данные, которые инкапсулируются в классе. Могут быть переменной базового типа или некоторым кортежем базовых типов, поддерживаемый в ЯП.

Метод – функция, которая проводит некоторые операции с данными. Данные могут как принадлежать классу (его поля) или переданы в метод извне. Также, в качестве результата вызова, метод может возвращать данные.

Поля и методы класса имеют модификаторы доступа. Модификатор доступа определяет какие объекты могут обращаться (использовать) поля и методы этого класса:

Private (приватный) – к таким полям и методам может обратиться только сам объект, которому они принадлежат;

Public (публичный) – к таким полям и методам может обратиться объект любого класса;

Protected (защищённый) – к таким полям и методам может обратиться только объект, класс которого находится в отношении наследника к классу данного объекта;

2.1.3.1 Алгоритм моделирования

Для реализации моделирования разработан принцип потока событий. В программе реализованы (в виде классов) сущности, составляющие потока событий: Машина (Machine), Партия (Batch), Рецепт (Recipe), Событие

(Event). Событие отражает факт окончания обработки партии в машине, событие происходит с задержкой – временем обработки партии в машине. Соответственно поток событий – это последовательность событий (выходов партий и установок), происходящих друг за другом.

В основу механики потока событий заложено три принципа:

1. Все машины в производстве (и в модели соответственно) работают параллельно.

2. Единовременно в установке происходит только одно событие. Или можно сказать, что происходит обработка только по одному рецепту.

3. Не предусмотрено запаздывание между окончанием обработки в текущей установке и началом следующей.

После ввода данных и запуска моделирования происходит инициализация модели. Каждая установка, во входной очереди которой есть партии, генерирует одно событие (принцип 2) и передаёт его в общий поток. Поток автоматически распределяется по возрастанию времени длительности события, место события в потоке определяется при вставке.

Процесс моделирования происходит по шагам. Когда модель получает команду (вызов функции) «выполнить шаг моделирования», из потока изымается первое событие, т.к. поток распределён по возрастанию времени длительности, длительность этого события наименьшая. У всех остальных событий длительность уменьшается, на величину длительности изъятого события (принцип 1), а глобальное модельное время увеличивается на эту же величину. Машина, в которой произошло событие, подставляет поток новое событие, для следующей в очереди партии (принцип 3). Разберём на примере.

На рисунке 2 схематично изображён поток событий для трёх случаев: начальные условия (инициализация), вызов одного шага моделирования и вызов двух шагов моделирования.

После вызова «Выполнить один шаг», из потока изымается событие Event1 (отмечено крестом). Глобальное модельное время увеличивается на

длительность Event1, т.е. на 100, а длительность остальных событий наоборот уменьшается. Установка mch1 добавляет новое событие в поток (Event 5).

При вызове «Выполнить два шага», выполняется один шаг моделирования дважды. На первом шаге выполняется Event2, время всех событий уменьшается на 100, а глобальное модельное время увеличивается. Машина mch2 добавляет событие Event6 в поток.

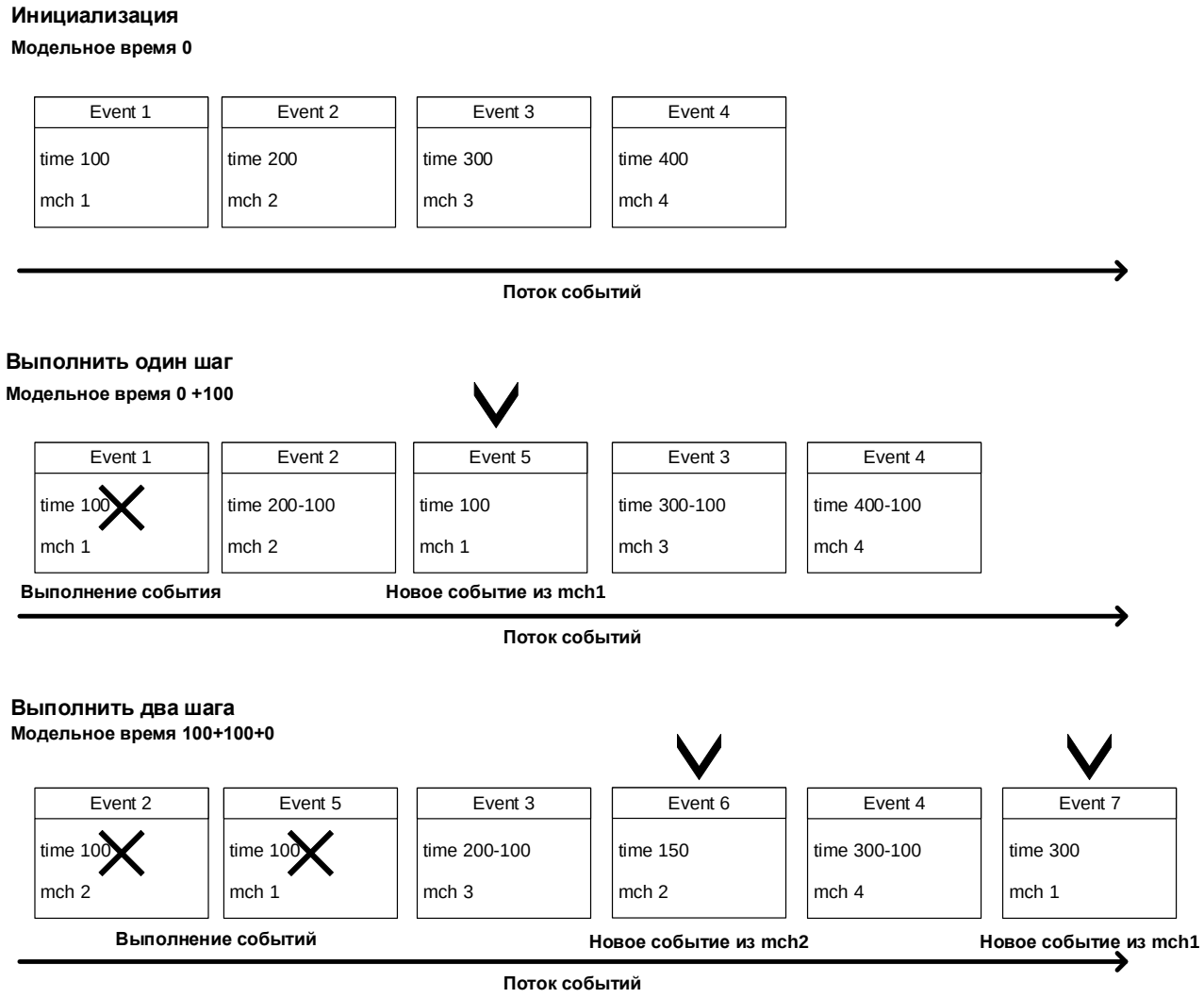


Рисунок 2 – Схема работы потока событий

Время длительности Event5 до первого шага составляло 100 единиц, соотнесённо после него, оно стало равно нулю. Что полностью согласуется с принципами моделирования и не нарушает работу системы, такой случай означает, что текущее и предыдущее события завершились одновременно. Время остальных событий и глобальное не изменится. Установка mch1 добавит событие в поток.

2.1.3.2 Программная структура

Иерархия классов в разработанном приложении содержит 9 классов, которые описывают поведение моделируемых объектов или служебных сущностей. Графически иерархия классов отражена в UML диаграмме на рисунке 3.

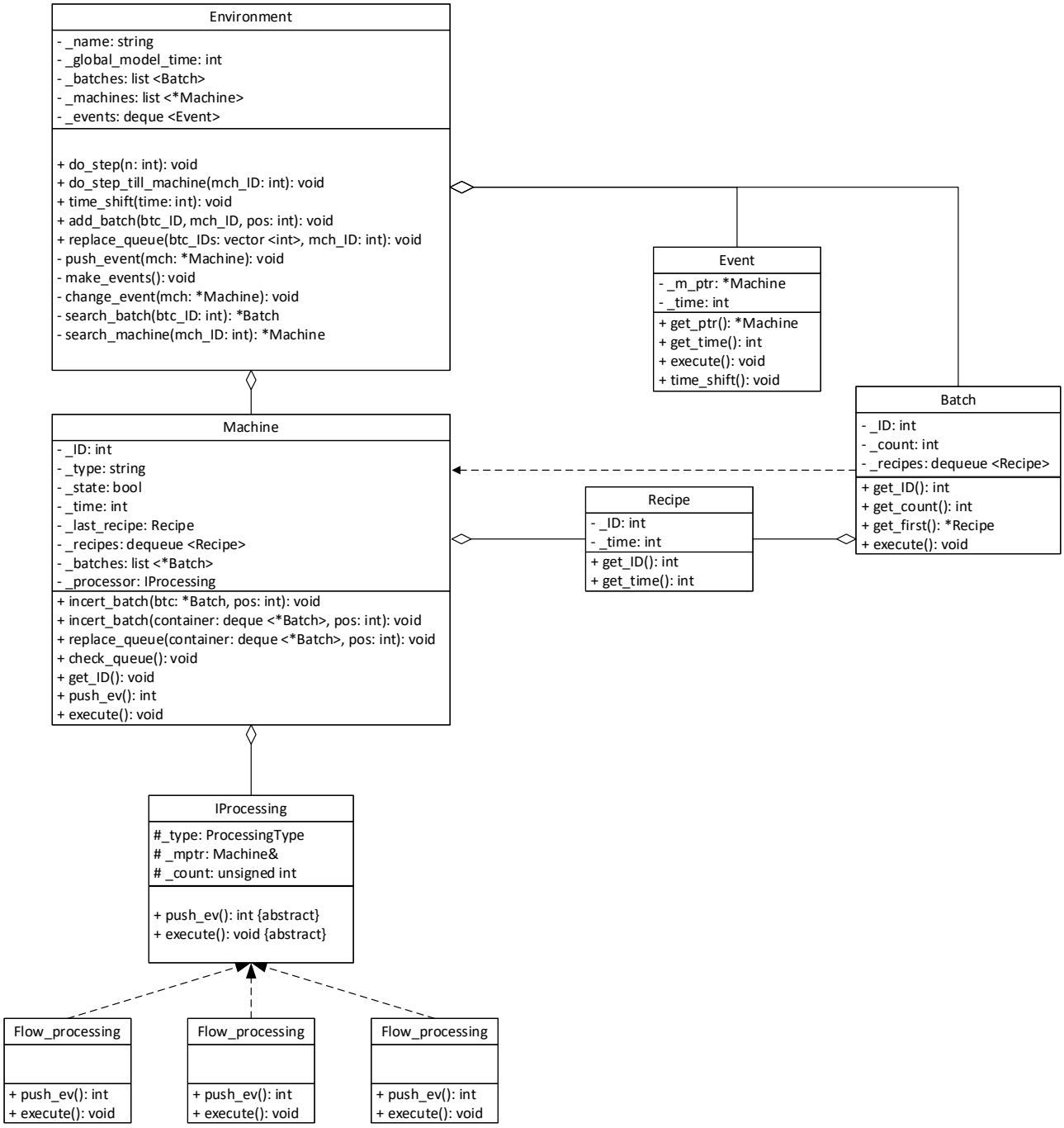


Рисунок 3 – UML диаграмма классов модели

Рецепт (класс Recipe) – класс, описывающий рецепты обработки в машинах. Поля, содержащиеся в классе защищены приватным модификатором доступа:

– _time – константу времени *r* из выражений в 2.1.2;

– _ID – уникальный идентификатор рецепта;

Доступ к значениям полей (изменять поля запрещено) осуществляется с помощью публичных методов:

– get_ID – метод возвращает идентификатор рецепта (поле _ID);

– get_time – метод, возвращающий константу времени (поле _time).

Партия (класс Batch) – класс, описывающий партии изделий, проходящие через установки. Поля этого класса также защищены private модификатором:

– _count – количество изделий в партии, *n* из выражений в 2.1.2;

– _ID – уникальный идентификатор партии;

– _recipes – контейнер типа двухсторонняя очередь, хранящий в себе последовательность рецептов, по очереди обработки партии в машинах.

Public методы, реализуют доступ к значениям полей:

– get_ID – метод возвращает идентификатор партии (поле _ID);

– get_count – возвращающий количество изделий в партии (поле _count);

– get_first – метод, возвращающий ссылку на текущий первый рецепт в маршрутном листе (ссылка на объект класса Recipe).

Событие (класс Event) – описывает событие, прохождение партии через машину. Поля защищены private модификатором:

– __m_ptr – указатель на машину, в которой происходит событие;

– _ID – уникальный идентификатор события;

– _time – длительность события.

Доступ к значениям и модификация полей осуществляется публичными методами класса:

– get_time – метод возвращающий время события (атрибут _time);

– get_ptr – метод возвращающий ссылку на машину, в которой происходит событие (ссылка на объект класса Machine);

– execute – метод «выполняющий» событие, вызывает метод execute в машине;

– time_shift – метод, уменьшающий длительность события.

Машина (класс Machine) – класс, описывающий машину и обработку партии в машине. Поля класса приватны:

– _ID – уникальный идентификатор машины;

– _time – константа времени смены рецепта, t_0 из формулы (2.6);

– _state – поле отражает состояние машины (включено / выключено);

– _batches – контейнер типа двусвязный список, содержит указатели на партии, описывает входную очередь установку;

– _recipes – контейнер типа динамический массив (deque), содержащий рецепты, выполняемые в данной установке.

– _last_resipe – обьект типа Recipe, хранит последний применённый в установке рецепт;

– _processor – особое поле, в нём класс Machine инкапсулирует класс IProcessing, который описывает и реализует *f(x,n)* из 2.1.2. Для доступа к полям класса Machine, наследники класса IProcessing определены как «дружественные» классу Machine.

Класс реализует публичные методы доступа и модификации значений полей:

– get_ID – возвращает значение поля _ID;

– insert_batch – в качестве входных данных принимает ссылку на объект класса Batch (или массив ссылок) и добавляет эти данные в поле _batches;

– replace_queue – принимает на вход массив ссылок на объект типа Batch и заменяет данные в поле _batches на него;

– check_queue, – проверяет, существуют ли партии в очереди (ссылки на Batch в _batches);

– addRecipe – принимает объект типа Recipe, добавляет его к полю _recipies;

– push_ev – данный метод отдаёт время длительности текущего события в машине, вызывает соответствующий метод у _processing;

– execute – «выполняет» событие в машине, вызывает соответствующий метод у _processing;

Среда (Environement) – этот класс инкапсулирует в себе вышеописанные классы и описывает всю модель в целом. Именно с ним (а точнее с его интерфейсом) взаимодействует программа управления моделью, а его поля и часть медов защищены приватным уровнем доступа. Содержит следующие приватные поля:

– _name – строковое имя модели;

– _batches – контейнер типа «словарь» (массив пар ключ - значение), ключами в котором являются идентификаторы партий, а значениями объекты типа Batch, представляет собой совокупность всех партий в модели;

– _machines – контейнер типа словарь, ключами в котором являются идентификаторы установок, а значениями объекты класса Machine, представляет собой совокупность всех установок в модели

– _events – контейнер типа deque, хранящий объекты типа Event, представляет собой поток событий;

– _ _global_model_time – целочисленная переменная, представляет собой модельное время.

Класс реализует следующие методы:

– time_shift – метод увеличивает глобальное модельное время на заданную величину, время всех событий в потоке уменьшается на эту же величину. Если у события в потоке оставшееся время меньше, чем заданная величина, событие выполняется;

– do_step_till_machine – на вход метод принимает целочисленный идентификатор, модель будет производить шаг моделирования до тех пор, пока первое событие в потоке не будет указывать на машину с переданным идентификатором;

– do_step – метод получает на вход целочисленное количество шагов, производит заданное количество шагов моделирования;

– q_add_batch – принимает целочисленные идентификаторы машины и партии (или массив партий). Добавляет партию во входную очередь машины, посредством вызова в объекте класса Machine метода insert_batch;

– replace_queue – принимает массив целочисленных идентификаторов партий и идентификатор установки. Заменяет очередь в заданной установке, посредством вызова в объекте класса Machine метода replace_queue;

Помимо публичных методов, реализовано несколько служебных приватных методов:

– push_event – получает на вход ссылку на объект типа Machine (установку) и вызывает у этого объекта метод push_event;

– make_events – метод инициализирующий поток событий, вызывает у всех объектов в контейнере _machines метод push_event;

– change_event – принимает на вход указатель на объект типа Machine, удаляет в контейнере _events Event, который указывает на этот объект и вызывает у него метод push_event;

– search_batch – получает на вход целочисленный идентификатор партии и возвращает указатель на соответствующий объект типа Batch;

– search_machine – получает на вход целочисленный идентификатор установки и возвращает указатель на соответствующий объект типа Machine;

– search_event – получает на вход целочисленный идентификатор события и возвращает указатель на соответствующий объект типа Event;

2.1.3.3 Реализация типов обработки изделий

Как было написано выше производственные агрегаты в приложении представлены классом Machine. Этот класс даёт только базовое описание установки, а различные типы обработки описывают наследники класса IProcessing.

IProcessing является классом интерфейсом для трёх наследников согласно типам обработки в 2.1.2, который даёт возможность полиморфно взаимодействовать с наследниками, и описывает их функциональность. Содержит поля:

– _type – тип обработки;

– _mptr – ссылка на объект типа Machine, которому принадлежит этот экземпляр;

В классе объявлены виртуальные методы:

– push_ev – рассчитывает и отдаёт время длительности текущего события в машине;

– execute – «выполняет» событие в машине, записывает данные в лог-файл;

Класс Flow_Processing, реализует потоковый тип обработки. Его метод push_ev рассчивает время длительности события по (2.7), ниже приведён листинг кода этой функции.

```
unsigned int Flow_Processing::push_ev()
{
    if (!_mptr._batches.empty())
    {
        Batch* it = _mptr._batches.front();
        const Recipe& rcp = it->get_first();
        if  (std::any_of(_mptr._recipes.begin(),  _mptr._recipes.end(),
[rcp](Recipe const r) {return rcp == r; }))
        {
            return  ((_mptr._last_resipe == it->get_first()) ?
                rcp.get_time() * it->get_count() : rcp.get_time() * it-
>get_count() + _mptr._time);
        }
        else throw (it);
    }
    else
    {
        Batch* ptr = nullptr;
        throw (ptr);
    }
}
```

Сначала производится проверка на пустую очередь, если очередь пуста, функция возвращает объект типа null (тип обозначающий ничто). По ссылке на первую партию в _batches, вызывается метод get_first, который возвращает

первый рецепт из _recipes партии (Batch). По массиву рецептов машины (_recipes объекта Machine), ищется совпадение с рецептом из партии. Если совпадение не найдено, функция пробрасывает исключение (сообщение об ошибке), т.к. это свидетельствует о том, что данная партия не может быть обработана в этой машине. Если рецепт найден, из рецепта берётся константа *r* (get_time) из партии константа *n* (get_count) и согласно формуле 2.3 рассчитывается длительность обработки. Если текущий рецепт не равен последнему, использованному в машине (_last_resipe), к длительности обработки прибавляется время переключения. Получившийся результат функция возвращает, ка длительность события.

Классы Group_Processing и Stack_Processing реализуют обработку группами пластин и партий. Их реализация отличается только тем, что в функции push_ev используются соответственно формулы (2.8) и (2.9), листинги кода этих функций приведены в приложениях Б и В

Рассмотрим алгоритм моделирования с учётом описанных классов и методов (рис. 4). При обращении к полям и методам, принадлежащим объектам используется нотация с точкой вида: **имя_обьекта . имя_поля/метода**.

Объект типа Environement назовём сокращённо Env, объекты типа Machine – mch *n*, где *n*, порядковый номер экземпляра в контейнере Env._machines, объекты типа Event – Event *n*, где *n*, порядковый номер экземпляра в контейнере Env._events.

После конфигурирования модели вызывается метод make_events и контейнер Env._events заполняется объектами типа Event, от каждого объекта типа Machine, в котором контейнер _batches не пустой.

При вызове метода Env.do_step(1), к полю _global_model_time прибавляется Event 1.time (длительность события), у всех экземпляров Event Env._events, кроме первого, вызывается метод time_shift(Event 1.get_time). Далее у Event 1 вызывается метод execute, Event 1 удаляется из контейнера, а у mch 1, на которую он указывал, вызывается метод push_event.

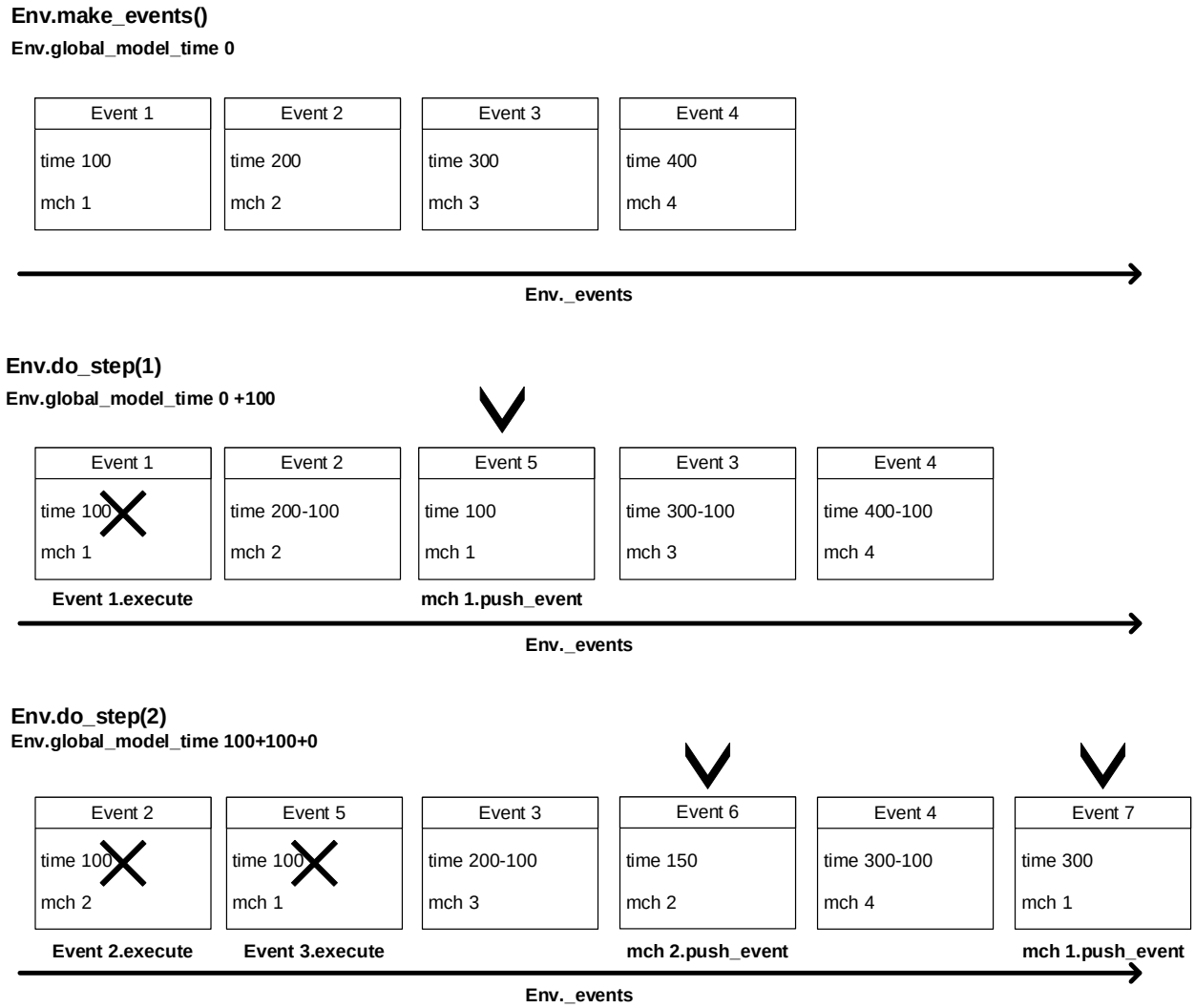


Рисунок 4 – Схема работы потока событий

При вызове Env.do_step(2), тот же алгоритм циклически повторяется дважды. В приложениях Г, Д. Вы можете увидеть листинг кода и блок-схему функции do_step() класса Environement.

2.2 Слой доступа к данным

Слой доступа к данным – подпрограмма, обеспечивающая обмен данным между приложением и БД. Эта программа включает в себя:

– Модели данных;

– Контекст базы данных;

– Методы взаимодействия с данными;

Для развёртывания и конфигурирования БД выбрана объектно-реляционная СУБД PostgreSQL – это сдобно распространяемое ПО с открытым исходным кодом и длительной историей разработки, имеющее ряд преимуществ:

– Высокопроизводительные и надёжные механизмы транзакций и репликации;

– Расширяемая система встроенных языков программирования;

– Наследование;

– Расширяемость;

История версий PostgreSQL насчитывает 13 редакций (и 14 экспериментальную), СУБД работает на языке SQL и поддерживает стандарт SQL 2016.

БД конфигурируется с помощью графического клиента (один из вариантов) pgAdmin (рис. 5).

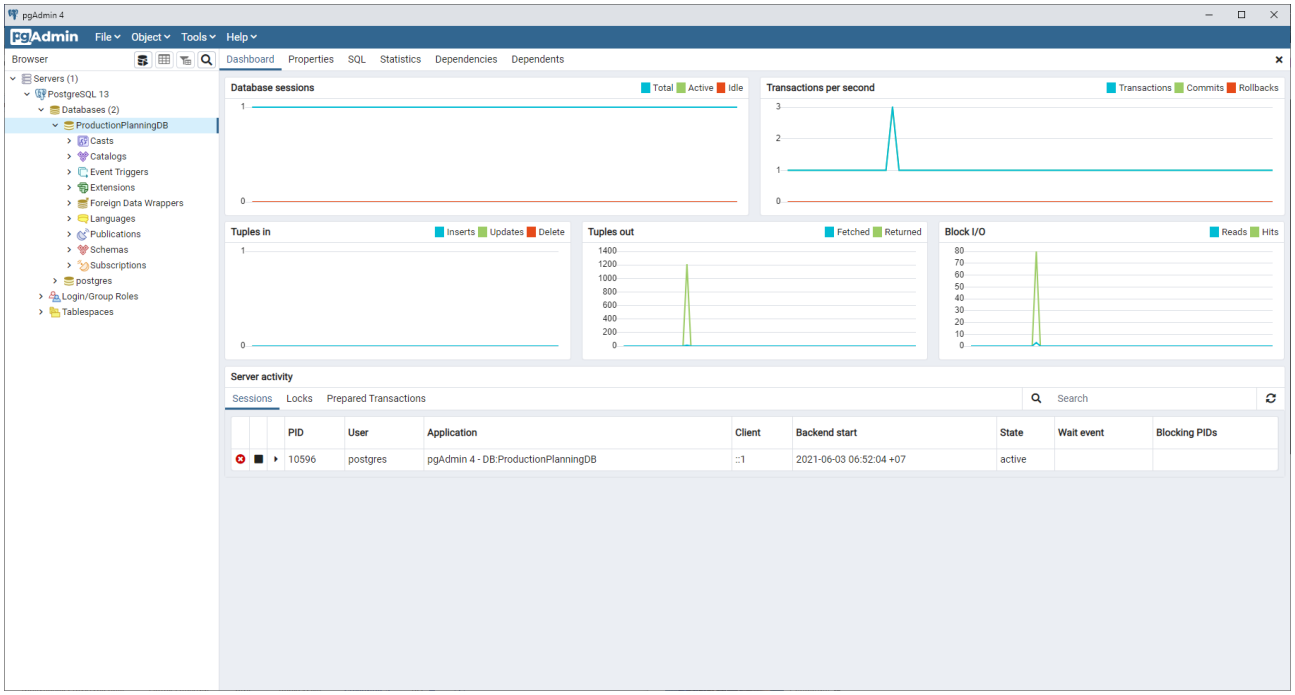


Рисунок 5 – Окно программы pgAdmin

В этой программе создаётся новая база данных и настраивается пользователь для доступа программы к базе, для него отключены специализированные администраторские разрешения.

2.2.1 Структура БД

Хотя структуру базы и её наполнение данными можно осуществить вручную с помощью клиентского приложения (pgAdmin или консоли). Существует другой способ конфигурирования структуры, заключаетсяв использовании специального модуля среды .Net – Entity Framework.

Entity Framework (EF) – технология среды .Net для работы с данными. Позволяет взаимодействовать с данными из различных хранилищ, абстрагируясь от работы с ними напрямую. При обработке данных также необходимо работать с таблицами, сущностями и ключами, но EF переносит эти концепции на объектно-ориентированный подход.

Основным компонентом EF является Entity Data Model или контекст базы данных. Контекст сопоставляет классы и объекты из программы с таблицами и сущностями в БД. Состоит из трёх уровней:

Концептуальный – определяются классы, объекты которых будут использоваться в приложении, а сущности в БД отражать их в базе;

Хранилища – данные о таблицах, столбцах, ключах и связях в базе данных;

Маппинга – сопоставляет классы из концептуального уровня с сущностями из уровня хранилища;

EF предлагает три возможных пути взаимодействия с базой данных:

- Database first – EF создаёт классы концептуального уровня исходя из текущей конфигурации БД;
- Model first – с помощью специальных средств разработчик создаёт модель БД, по которой EF создаёт концептуальные классы и БД на сервере;
- Code first – разработчик создаёт концептуальные классы, которые будут храниться в базе, по ним EF создаёт базу данных на сервере.

Для конфигурирования базы в проекте был выбран подход Code first, т.к. он подразумевает наименее «ручную» работу и опирается на языки программирования.

Для конфигурирования структуры БД разработаны модели данных – классы содержащие только поля, в большинстве своём они воспроизводят классы определённые в модуле моделирования производственных, но также добавляют некоторые служебные классы. Каждый класс в последствии становится сущностью (таблицей в базе данных). Рассмотрим эти классы (рис. 6).

Структура базы данных насчитывает семь классов. Пять из них дублируют классы в модуле моделирования: Model, Machine, Batch, Recipe, Event. Их поля базовых типов имеют тоже смысл, что и у классов в модели. Поля типов коллекции или классов предназначены для формирования навигационных свойств. Код определения моделей данных можно увидеть в приложении Е.

Добавлены два служебных класса:

Company – потребителем разработки выступают предприятия и эта сущность отражает их в системе. Класс инкапсулирует уникальный идентификатор (поле Id) и наименование организации (поле Name).

User – сущность отражает в системе работника организации, пользователя системы. Класс инкапсулирует уникальный идентификатор (поле Id) и фамилию, имя сотрудника (поле Name).

Указанные на схеме связи конфигурируются в контексте базы данных. Для определения контекста базы данных необходимо создать класс и унаследоваться в нём от встроенного в EF класса DbContext, модифицируя этот базовый класс определяется структура пользовательской базы данных. Реализацию контекста проектной базы данных можно увидеть в приложении Ж.

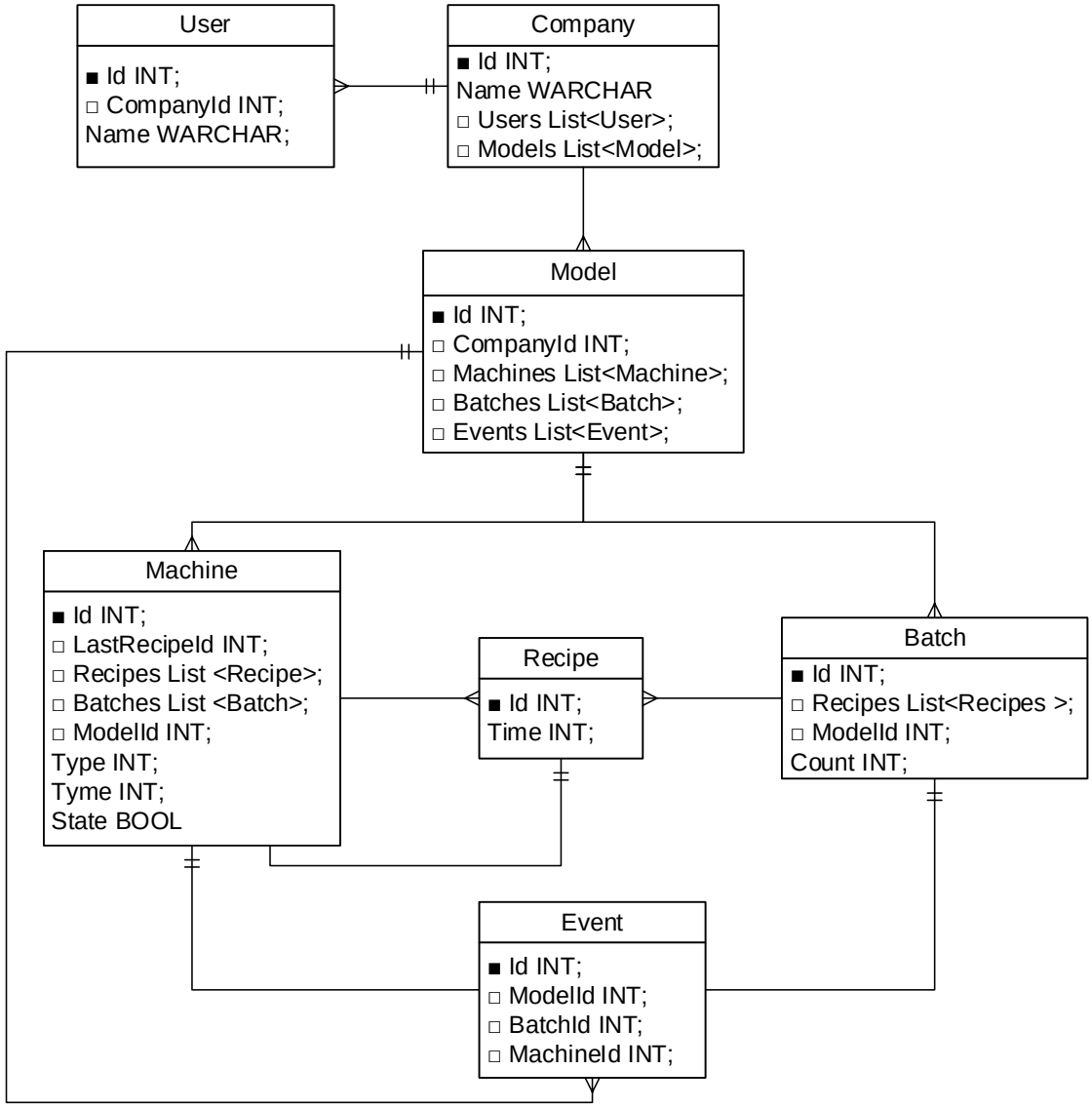


Рисунок 6 – Структурная схема БД

Сущности добавляются в базу путём инкапсуляции в классе объектов обобщённого класса DbSet<T> где T это класс модели данных, сущность которой необходимо добавить в базу.

Соединение с базой и прочие технические параметры можно конфигурировать двумя путями: переопределением метода OnConfiguring или, как в этом проекте, определением конструктора с параметрами и передачей параметров в конструктор базового класса.

Связи в базах данных бывают двухсторонние – в обоих сущностях в связи необходимы ссылки друг на друга и односторонние – ссылка существует только в одной сущности, а вторая никак не зависит от первой. Так же существуют три вида связей:

- Многие ко многим – каждой сущности с обоих сторон связи соответствует одна или несколько сущностей с другой стороны связи;
- Один ко многим – каждой сущности с одной стороны связи соответствует одна или несколько сущностей с другой, при этом каждой сущности с другой стороны связи соответствует только одна сущность с другой, если связь двухсторонняя.
- Один к одному – каждой сущности с одной стороны связи соответствует только одна сущность с другой, также и для сущности с другой стороны связи, если связь двухсторонняя.

В структуре БД (рис. 6) определены следующие связи:

– Двухсторонняя связь один ко многим между сущностями Company и User, т.к. у одной организации может быть несколько сотрудников, пользователей системы;

– Односторонняя связь один ко многим между Company и Model, для реализации возможности работы с несколькими моделями.

– Двусторонняя один ко многим между Model и Machine, т.к. в одной модели существует несколько установок, и каждая установка принадлежит только одной модели;

– Двусторонняя один ко многим между Model и Batch, т.к. в одной модели существует несколько партий, и каждая партия принадлежит только одной модели;

– Двусторонняя один ко многим между Model и Event, т.к. в одной модели существует несколько одно или несколько, и каждая событие принадлежит только одной модели;

– Односторонняя один к одному между Event и Machine, таким образом класс события ссылается на машину, в которой обрабатывается партия.

– Односторонняя один к одному между Event и Batch, таким образом класс события ссылается на партию, которая проходит установку в машине.

– Одностороння связь один ко многим между Machine и Recipe, т. к. рецепты инкапсулируются не только в установках, но и в партиях, к тому же может существовать несколько установок с одинаковыми рецептами;

– Дополнительно предусмотрена односторонняя связь один к одному между Machine и Recipe, для хранения последнего использованного в машине рецепта;

– Одностороння связь один ко многим между Batch и Recipe, т. к. рецепты инкапсулируются не только в партиях, но и в установках, к тому же может рецепты в партиях могут повторяться;

Связи в контексте базы данных конфигурируются переопределением метода базового класса OnModelCreating. На вход этого метода подаётся объект класса DbModelBuilder. И каждая связь конфигурируется посредством вызова у него цепочки методов в следующем формате:

DbModelBuilder.Entity<T>().Has[Multiplicity](Property).With[Multiplicity](Property)

Метод Entity<T>() (где T класс модели данных), возвращает конфигурационный объект для данной модели данных. У этого объекта определены две группы методов, конфигурирующие связь:

Методы группы Has задают навигационное свойство – свойство класса, которое отсылает к другому объекту. Multiplicity означает характер связи к одному или ко многим.

Методы группы With определяют отношение второй сущности в связи к первой. В метод передаётся навигационное свойство для связи к одному или ко многим, или ничего, если связь одностороння.

Листинг кода переопределения метода OnModelCreating приведён ниже.

```
protected override void OnModelCreating(DbModelBuilder modelBuilder)
{
    modelBuilder.Entity<Company>().HasMany(c => c.Users)
.WithRequired(u => u.Company);
    modelBuilder.Entity<Company>().HasMany(c => c.Models)
        .WithOptional();
    modelBuilder.Entity<Model>().HasMany(m => m.Machines)
        .WithRequired(mch => mch.Model);
    modelBuilder.Entity<Model>().HasMany(m => m.Batches)
        .WithRequired(b => b.Model);
    modelBuilder.Entity<Model>().HasMany(m => m.Events)
        .WithRequired(e => e.Model);
    modelBuilder.Entity<Machine>().HasMany(m => m.Recipes)
        .WithOptional();
    modelBuilder.Entity<Machine>().HasRequired(m => m.LastRecipe)
        .WithOptional();
    modelBuilder.Entity<Batch>().HasMany(b => b.Recipes)
        .WithOptional();
    modelBuilder.Entity<Event>().HasRequired(e => e.Machine)
        .WithOptional();
    modelBuilder.Entity<Event>().HasRequired(e => e.Batch)
        .WithOptional();
}
```

2.3 Клиент-серверный модуль

Данный модуль является одновременно HMI интерфейсом и центральным узлом программы, обеспечивая запросы данных и DAL и передачу данных, и вызов необходимых методов в модели производственных процессов.

Для реализации данного модуля решено разработать Web приложение. Такие приложения работают по протоколу HTTP (HTTPS если с шифрованием) по принципу запрос – ответ. Пользователь работает с HTML страницей в браузере, после ввода данных пользователь тем или иным образом инициирует отправку формы на сервер (в формате HTTP запроса). Сервер обрабатывает полученные данные и возвращает браузеру новую форму.

По сравнению с desktop, такие приложения имеют ряд преимуществ. Работа приложения не зависит от архитектуры ОС потребителя, все взаимодействие пользователя с приложением происходит в браузере. Исключено нежелательное влияние на программу, т.к. она выполняется на

защищённых серверах нет необходимости для каждого нового потребителя разворачивать инфраструктуру заново и возможность быстро расширять количество пользователей системы. Консистентность данных, т.к. к ним исключен доступ «в обход» разработанного приложения.

Существуют различные паттерны проектирования Web приложений, их применение сводится к расширению функциональности готовых механизмов поведения под нужды разработки. Для разработки принято решения воспользоваться паттерном Модель-представление-контроллер, хотя этот паттерн разработан сравнительно давно, он не считается устаревшим, и среда разработки приложения имеет готовый шаблон реализации данной архитектуры.

Проекты других модулей внедрены в приложение с помощью механизма внедрения зависимостей в виде SingleTone объектов и статических классов.

2.3.1 Архитектура приложения

Модель-представление-контроллер (Model-View-Controller, MVC) – шаблон проектирования приложений, в основе которого находится принцип разделения приложения на три модуля: модель, представление и контроллер, модификация которых может осуществляться независимо.

Модель (model)

Описывает используемые в приложении данные, а также логику, которая связана непосредственно с данными, например, логику валидации данных. Как правило, объекты моделей хранятся в базе данных.

В MVC модели представлены двумя основными типами: модели представлений, которые используются представлениями для отображения и передачи данных, и модели домена, которые описывают логику управления данными.

Модель может содержать данные, хранить логику управления этими данными. В то же время модель не должна содержать логику взаимодействия

с пользователем и не должна определять механизм обработки запроса. Кроме того, модель не должна содержать логику отображения данных в представлении.

Представления (view)

Отвечают за визуальную часть или пользовательский интерфейс, нередко HTML-страница, через который пользователь взаимодействует с приложением. Также представление может содержать логику, связанную с отображением данных. В то же время представление не должно содержать логику обработки запроса пользователя или управления данными.

Контроллер (controller)

Представляет центральный компонент MVC, который обеспечивает связь между пользователем и приложением, представлением и хранилищем данных. Он содержит логику обработки запроса пользователя. Контроллер получает вводимые пользователем данные и обрабатывает их. И в зависимости от результатов обработки отправляет пользователю определенный вывод, например, в виде представления, наполненного данными моделей.

Отношения между компонентами схематически обозначены на рис. 7.

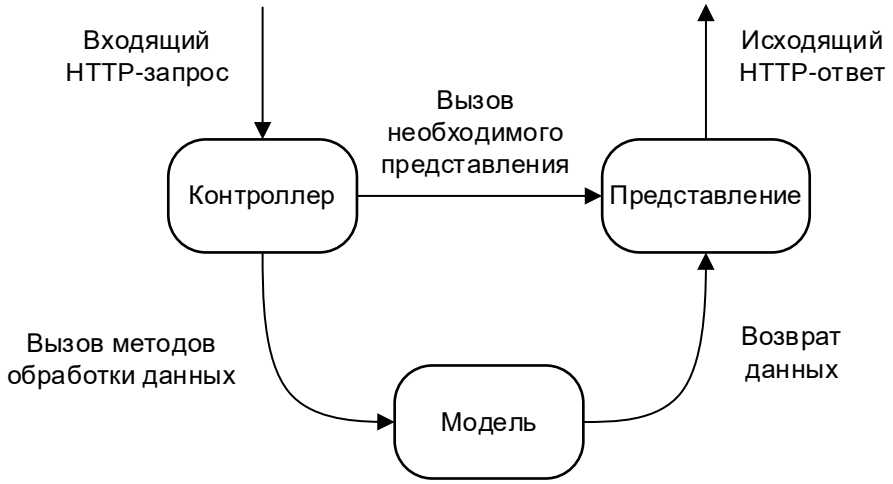


Рисунок 7 – Схема паттерна MVC

В этой схеме модель является независимым компонентом - любые изменения контроллера или представления никак не влияют на модель. Контроллер и представление являются относительно независимыми компонентами. Так, из представления можно обращаться к определенному

контроллеру, а из контроллера генерировать представления, но при этом нередко их можно изменять независимо друг от друга.

Такое разграничение компонентов приложения позволяет реализовать концепцию разделение ответственности, при которой каждый компонент отвечает за свою строго очерченную сферу. В связи с чем легче построить работу над отдельными компонентами. И благодаря этому приложение легче разрабатывать, поддерживать и тестировать отдельные компоненты.

2.3.2 Структура веб-форм

Контроллер в MVC конфигурируется классом наследником от класса Controller. В этом классе конфигурируется результаты запроса и действия, производимые по запросу: возвращение формы, вызов метода в модели и т.п. Результат запроса конфигурируется функцией, которая возвращает объект класса ActionResult, пример функций предоставлен ниже.

```
public ActionResult Index(){
    return View();
}
public ActionResult Model(){
    return View();
}
public ActionResult Machines(){
    return View();
}
public ActionResult About(){
    return View();
}
```

Приложение MVC конфигурируется в специальном классе Startup – этот класс является входной точкой в ASP.NET приложение, производит конфигурацию приложения, настраивает сервисы, которые приложение будет использовать, устанавливает компоненты для обработки запроса.

Адрес HTTP запроса и функция, возвращающая ActionResult из контроллера, конфигурируются в таблице маршрутов. Таблица маршрутов определяется с помощью метода UseEndpoints, листинг кода метода представлен ниже.

```
app.UseEndpoints(endpoints =>
{
    endpoints.MapControllerRoute(
        name: "default",
        pattern: "{controller=Api}/{action=Index}");

    endpoints.MapControllerRoute(
        name: "model-config",
        pattern: "{controller=Api}/{action=ModelConfigurator}");

    endpoints.MapControllerRoute(
        name: "production-planning",
        pattern: "{controller=Api}/{action=ProductionPlanning}");

    endpoints.MapControllerRoute(
        name: "about",
        pattern: "{controller=Api}/{action=About}");
});
```

В разработанном приложении принята следующая иерархия (HTML страниц) HMI экранов (рис. 8).

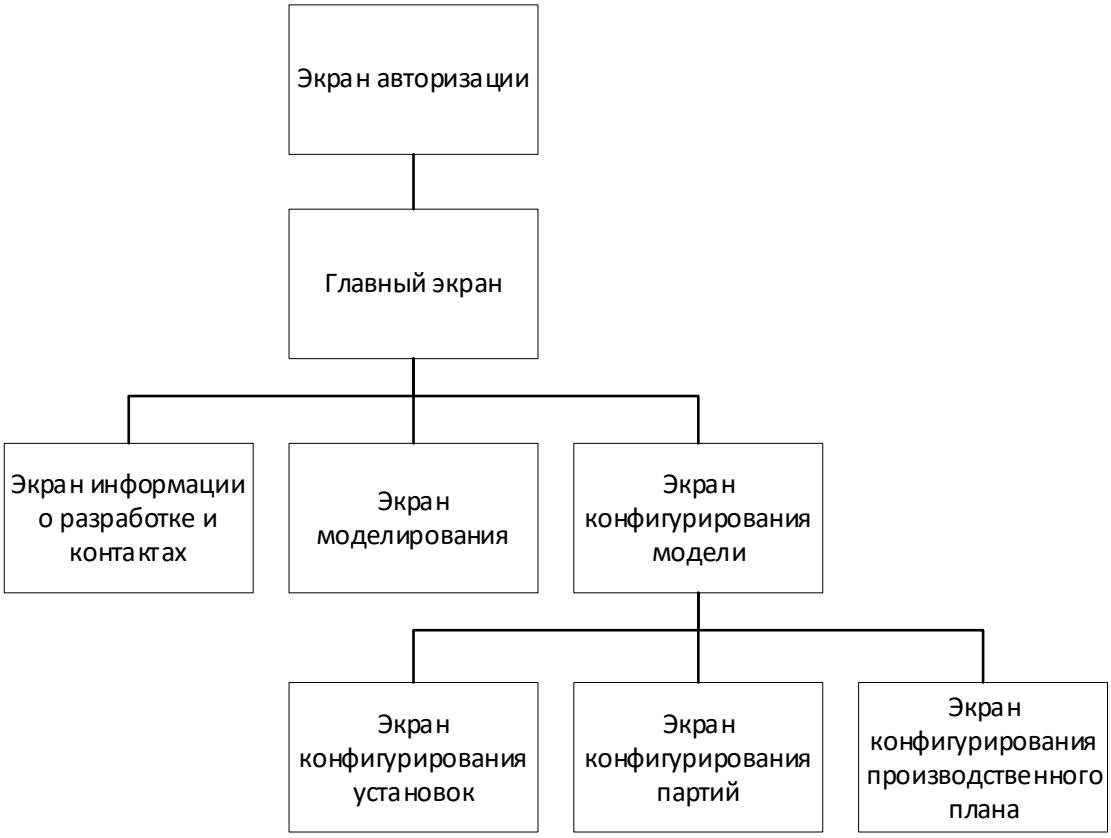


Рисунок 8 – Иерархия HMI экранов в приложении

На рисунке 9 приведён пример HMI экрана (экран авторизации).

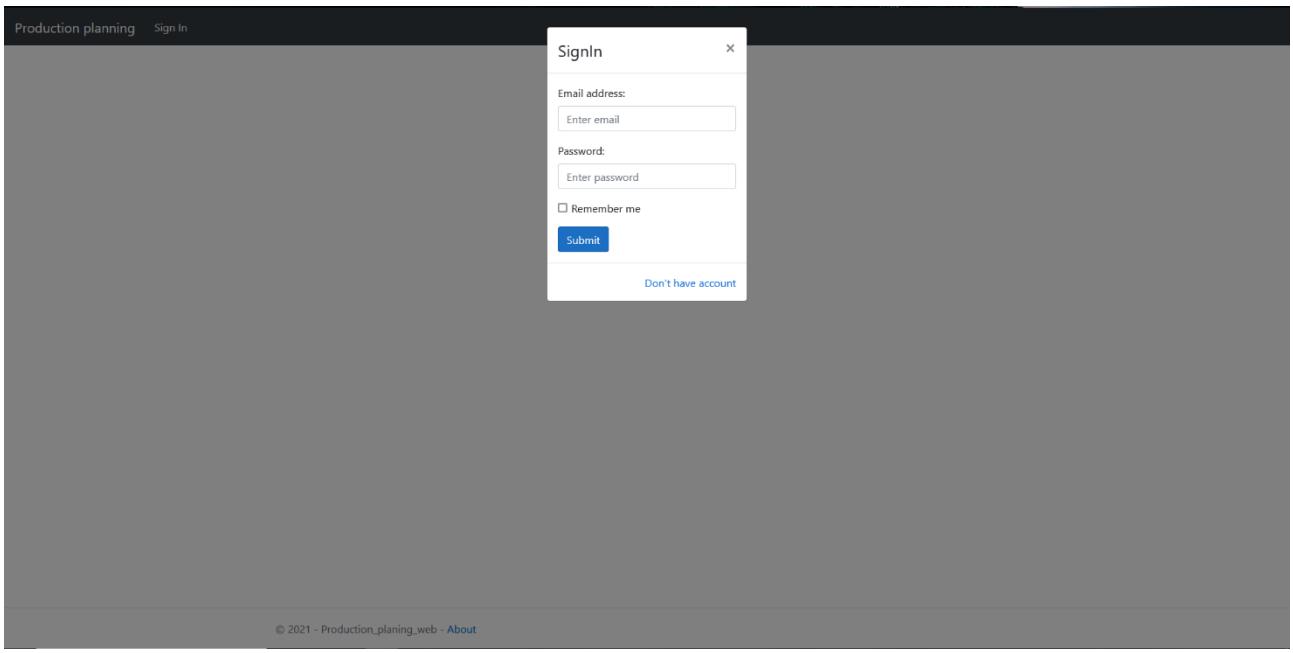


Рисунок 9 – Экран авторизации пользователя

3 Финансовый менеджмент, ресурсоэффективность и ресурсосбережение

Компьютерное моделирование является одним из наиболее востребованных направлений развития математической науки и информационных технологий.

Цель раздела «Финансовый менеджмент, ресурсоэффективность и ресурсосбережение» ¬ показать успешное экономическое будущее проекта, оценить его эффективность, возможные риски, предоставить информацию об управлении в процессе разработки.

Чтобы достичь поставленной цели необходимо решить задачи по организации работы над проектным решением, по планированию этапов разработки, оценить перспективность и коммерческий потенциал проекта, рассчитать бюджет, необходимый для реализации проекта, оценить социальную и экономическую эффективность проекта.

Цель данной НИР разработать среду моделирования дискретного потокового производства. В дальнейшем данная система будет использована в разработке и тестировании алгоритмов производственного планирования.

3.1 Оценка коммерческого и инновационного потенциала НТИ

Перед планированием работы, определением ресурсного и экономического потенциала разработки программно-алгоритмического комплекса, следует уделить особое внимание оценки коммерческого потенциала и перспективности новой разработки в целом, дать характеристику и определить сегмент рынка, на который будет ориентироваться компания, при продаже данной продукции.

3.1.1 Потенциальные потребители результатов исследования

Целевой рынок можно разделить на три сегмента, по характеру применения разработки:

• Научные и образовательные учреждения – коммерческие и некоммерческие научнопрактические разработки;

• Компании разработчики ПО – коммерческие научнопрактические разработки;

• Крупные промышленные предприятия – практическое применение;

Карта сегментирования рынка следующая (т 1).

Таблица 1 – Карта сегментирования рынка продаж

Использование продукции / Группа потребителей	Некоммерческая разработка	Коммерческая разработка	Практическое использование
ВУЗы и НИИ			
Разработчики ПО			
Промышленные предприятия			

ТПУ		AnyLogic Company		Транспортная компания «ПЭК»	

Согласно карте сегментирования рынка, можно сделать вывод об относительной свободе сегмента продаж программно-алгоритмических комплексов автоматизированного планирования процессов.

3.1.2 Анализ конкурентных технических решений

Анализ конкурентных технических решений с позиции ресурсоэффективности и ресурсосбережения будет основываться на сравнении разрабатываемой системы (ф) и двух конкурентных решений производителей систем автоматизированного управления и планирования производства, а именно Simatic IT Preactor компании Siemens (к1) и AnyLogic, компании AnyLogic (к2).

Таблица 2 – Анализ конкурентных технических решений

Критерий оценки	Вес критерия	Баллы			Конкурентно-способность		
		$Б_ф$	$Б_{к1}$	$Б_{к2}$	$К_ф$	$К_{к1}$	$К_{к2}$
Технические критерии оценки ресурсоэффективности							
1. Повышение загрузки оборудования	0,1	3	5	5	0,3	0,5	0,5
2. Надёжность алгоритмов	0,1	3	4	5	0,3	0,4	0,5
3. Потребность в вычислительных мощностях	0,1	5	2	3	0,5	0,2	0,3
4. Скорость обработки поступающих массивов данных	0,1	4	3	4	0,4	0,3	0,4
5. Ресурсоэффективность	0,05	3	4	5	0,15	0,2	0,25
6. Простота эксплуатации	0,1	5	4	4	0,5	0,4	0,4
7. Масштабируемость	0,05	4	3	3	0,2	0,15	0,15
8. Многопрофильность	0,1	3	5	5	0,3	0,5	0,5
Экономические критерии оценки эффективности							
1. Конкурентоспособность продукта	0,1	3	5	5	0,3	0,5	0,5
2. Уровень проникновения на рынок	0,05	0	4	5	0	0,2	0,25
3. Цена	0,1	5	3	4	0,5	0,3	0,4
4. Предполагаемый срок эксплуатации	0,1	5	5	5	0,5	0,5	0,5
5. Послепродажное обслуживание	0,05	3	5	5	0,15	0,25	0,25
Итого	1	43	47	53	3,8	3,9	4,4

В целом разработка уступает индустриальным конкурентам, но разрыв позволяет предположить о возможности продукта конкурировать с ними. Кроме того, по отдельным пунктам: масштабируемость, потребление вычислительных мощностей и цена, разработка опережает конкурентов.

3.1.3 SWOT анализ

Проведём исследование внутренних и внешних свойств проекта с помощью методики SWOT-анализа. Матрица SWOT-анализа () описывает сильные и слабые стороны проекта (внутренние факторы), а также показывает возможности и угрозы (внешние факторы) и возможные направления реализации.

В рамках первого этапа анализа построим матрицу SWOT (т 3) с описанием сильных и слабых сторон проекта, а также возможностей и угроз.

Таблица 3 – Матрица SWOT

	Сильные стороны: С1. Многопрофильность; С2. Возможность внедрения в другие программные комплексы; С3. Расширяемость системы; С4. Спонсирование по грантовой программе.	**Слабые стороны:** Сл1. Опыт в разработке. Сл2. «Новый игрок» на рынке систем моделирования и планирования. Сл3. Использование зарубежного ПО в разработке.
Возможности: В1. Заинтересованность промышленных предприятий. В2. Получение дополнительных грантов и финансирование из внебюджетных средств; В3. Выход на международный рынок систем планирования производства и моделирования; В4. Сотрудничество с российскими разработчиками ПО планирования и моделирования.		
Угрозы: У1. Наличие сильных конкурентов; У2. Ограничения на экспорт разработки; У3. Введение дополнительных государственных требований к сертификации ПО; У4. Исчерпание финансирования.		

В рамках второго этапа проведём анализ соответствия сильных и слабых сторон проекта внешним условиям окружающей среды. Анализ представлен ниже в виде интерактивных матриц, сильное соответствие отмечено знаком «+», слабое знаком «–».

Таблица 4 – Интерактивная матрица сильных сторон и возможностей проекта

Сильные стороны					
Возможности		С1	С2	С3	С4
	В1	+	–	+	+
	В2	+	+	+	+
	В3	+	–	+	+
	В4	–	+	+	–

Направления реализации сильных сторон и возможностей: В1В2В3С1С3С4, В2В4С2С3С4.

Таблица 5 – Интерактивная матрица слабых сторон и возможностей проекта

Слабые стороны				
Возможности		Сл1	Сл2	Сл3
	В1	–	+	–
	В2	+	+	+
	В3	–	–	–
	В4	+	–	+

Направления реализации слабых сторон и возможностей: В2В4Сл1, В1В2Сл2, В2В4Сл3.

Таблица 6 – Интерактивная матрица сильных сторон и угроз проекта

Сильные стороны					
Угрозы		С1	С2	С3	С4
	У1	+	+	+	–
	У2	–	–	+	+
	У3	–	–	+	+
	У4	+	+	+	–

Направления реализации сильных сторон и угроз: У1С1С2С3, У2У3С3С4, У4С1С2С3.

Таблица 7 – Интерактивная матрица слабых сторон и угроз проекта

Слабые стороны				
Угрозы		Сл1	Сл2	Сл3
	У1	+	+	–
	У2	–	–	+
	У3	–	–	–
	У4	+	+	+

Направления реализации слабых сторон и угроз: У1У4Сл1Сл2, У2У4Сл3.

В рамках третьего этапа построим итоговую матрицу SWOT анализа, результат представлен в таблице (т 8)

Таблица 8 – Итоговая матрица SWOT

	Сильные стороны: С1. Многопрофильность; С2. Возможность внедрения в другие программные комплексы; С3. Расширяемость системы; С4. Спонсирование по грантовой программе.	**Слабые стороны:** Сл1. Опыт в разработке. Сл2. «Новый игрок» на рынке систем моделирования и планирования. Сл3. Использование зарубежного ПО в разработке.
Возможности: В1. Заинтересованность промышленных предприятий. В2. Получение дополнительных грантов и финансирование из внебюджетных средств; В3. Выход на международный рынок систем планирования производства и моделирования; В4. Сотрудничество с российскими разработчиками ПО планирования и моделирования.	В1В2В3С1С3С4 – Внедрение ПО как в России так и за рубежом. В2В4С2С3С4 – Дальнейшие разработки, расширение функционала, интеграция с другими продуктами.	В2В4Сл1 – Привлечение специалистов, совместные разработки с другими компаниями; В1В2Сл2 – Рекламная кампания успешных внедрений; В2В4Сл3 – Переход на отечественное ПО по мере развития проекта.
Угрозы: У1. Наличие сильных конкурентов; У2. Ограничения на экспорт разработки; У3. Введение дополнительных государственных требований к сертификации ПО; У4. Исчерпание финансирования.	У1С1С2С3 – наращивание функционала, интеграция; У2У3С3С4 – Ориентирование разработки на внутренний рынок; У4С1С2С3 – лицензирование, продажа прав на использование интеллектуальной собственности.	У1У4Сл1Сл2 – Сотрудничество с разработчиками ПО планирования и моделирования, обмен опытом; У2У4Сл3 – Привлечение финансирования из гос. сектора на импортозамещение.

3.1.4 Оценка готовности проекта к коммерциализации

Для оценки готовности проекта определим показатели по вопросам в таблице 9. Оценка проводится по пятибалльной шкале. При оценке научного проекта: 1 балл – не проработано, 2 балла – проработка слабая, 3 балла – выполнено, качество посредственное, 4 балла – удовлетворительное качество,

5 баллов – качество подтверждено сторонним специалистом. При оценке знаний разработчика: 1 балл – не знаю, 2 балла – только теоретические знания, 3 балла – теоретические знания с практическими примерами, 4 балла – умею, практикую, 5 баллов – могу консультировать по вопросу.

Таблица 9 – Таблица оценки готовности научного проекта к коммерциализации

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

Продолжение таблицы 9

№ п/п	Наименование	Степень проработанности научного проекта	Уровень имеющихся знаний у разработчика
14	Имеется команда для коммерциализации научной разработки	3	3
15	Проработан механизм реализации научной разработки	4	4
ИТОГО		49	48

Итоговые результаты показывают проработанность проекта выше среднего, для успешного дальнейшего продвижения необходимо развить компетенции в области экономики и маркетинга.

3.2 Инициация проекта

В данном этапе фиксируются начальные цели, содержание и финансовые ресурсы. Определяются заинтересованы стороны, которые могут повлиять на конечный результат проекта. Эта информация закрепляется в уставе проекта.

3.2.1 Цели и результат проекта

Сначала определим заинтересованные стороны (т 10). Заинтересованные стороны – это лица или организации, которые активно заинтересованы и/или могут быть как положительно, так и отрицательно затронуты в результате проекта.

Таблица 10 – Заинтересованные в проекте стороны

Заинтересованные стороны	Ожидания
Томский политехнический университет	Лицензирование научных разработок, публикация научных материалов.
Транспортная компания «ПЭК»	Программный пакет для практического применения

В таблице 11 представим цель и результаты проекта, а также критерии их достижения и требования к результатам.

Таблица 11 – Цели и результаты проекта

Характеристика	Значение
Цель проекта:	Создание программного комплекса управления производством, повышение эффективности производства.
Ожидаемые результаты:	Программный продукт, реализующий автоматизированное планирование производственного процесса и управление движением продукции в производстве микроэлектроники.
Критерии приёмки результатов:	• Уменьшение времени простоя продукции на (10-50)%; • Снижение производственного брака на 10%; • Понижение затрат производственных средств на (5-10)%; • Увеличение производительности отдела планирования за счет уменьшения временных затрат на (40-70)% и уменьшения количества работников, участвующих в процессе составления производственного расписания до одного человека на цех.
Требования к результатам:	1. Разработать и внедрить улучшенные алгоритмы планирования в существующий прототип. 2. Разработать подсистему программного динамического моделирования производства. 3. Разработать подсистему мониторинга текущей производственной ситуации. 4. Разработать экспериментальный образец интеллектуальной системы планирования производственного процесса предприятия по производству микроэлектроники.

3.2.2 Организационная структура проекта

В таблице 12 отразим организационную структуру, роль и функции каждого члена команды.

Таблица 12 – Рабочая группа

№ п/п	ФИО, основное место работы, должность	Роль	Функции	Трудо-затраты, час.
1	Филипас Александр Александрович, Томский политехнический университет, доцент	Руководитель	Заверение документов, определение направления развития проекта.	
2	Никитин Андрей Сергеевич, Томский политехнический университет, магистр	Исполнитель	Разработка математических алгоритмов и ПО, документирование результатов.	

3.2.3 Ограничения и допущения

Определим факторы, которые могут послужить ограничением степени свободы участников команды проекта (т 12).

Таблица 13 – Ограничения проекта

Фактор	Ограничения / допущения
Бюджет	500 т.р.
Источники	Фонд развития инноваций
Сроки	27.12.2019 – 04.06.21
Дата утверждения плана управления проектом	01.02.20
Дата завершения	04.06.21
Прочие ограничения	Операционная система Windows; Использование технологии ASP.NET в реализации серверной части; Реализация программной модели в виде подключаемой библиотеки

3.3 Планирование управления НТИ

Для планирования хода работ НТИ определим структуру работ, распланируем контрольные события и определим календарный план проекта, который выразим в диаграмме Ганта.

3.3.1 Иерархическая структура работ

Построим иерархическую структуру (рис. 10), которая отражает и структурирует объём работ по проекту.

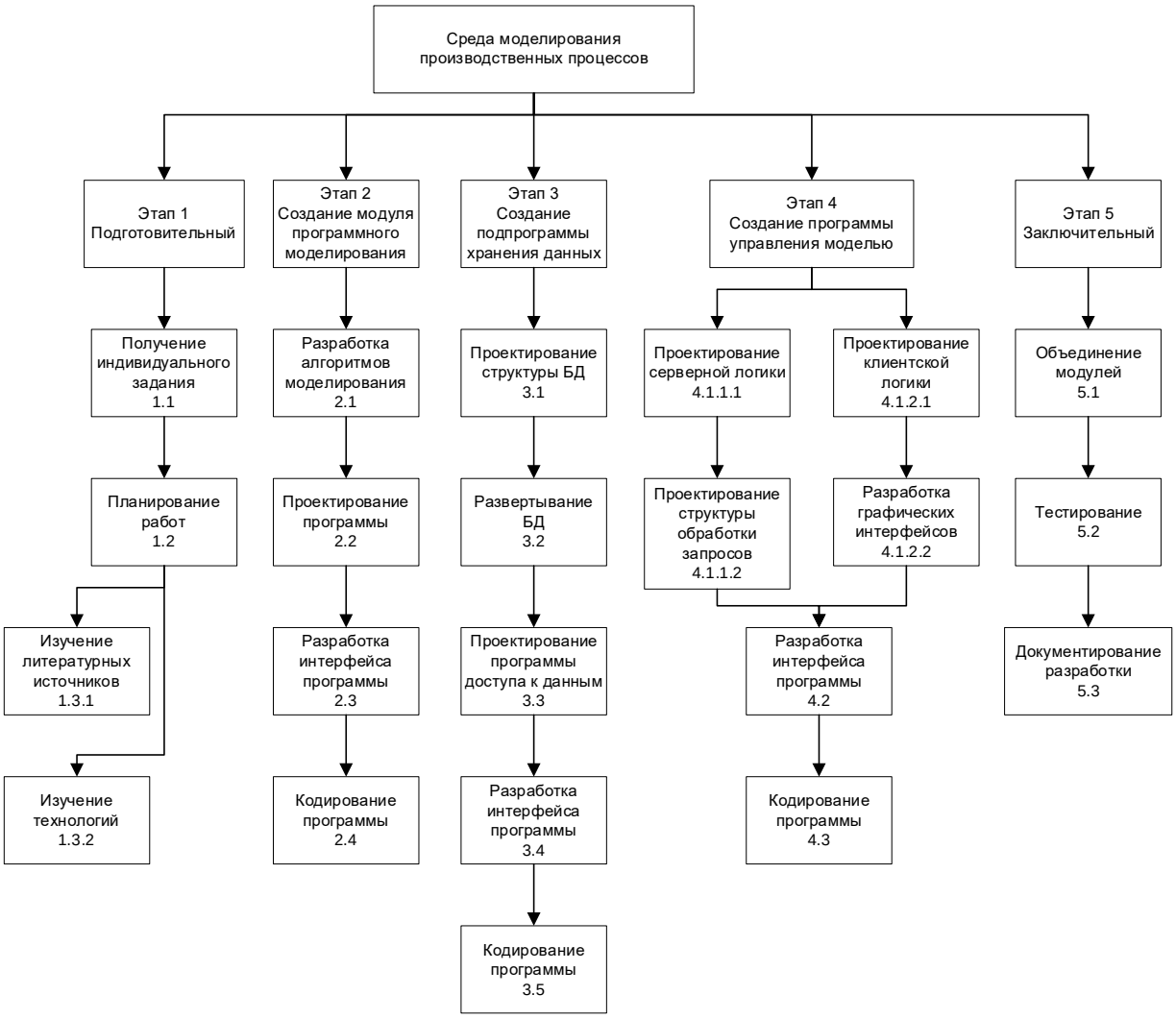


Рисунок 10 – ИСР проекта

3.3.2 Контрольные события проекта

Определим ключевые события проекта, их даты и результаты в таблице 14.

Таблица 14 – Контрольные события проекта

№ п/п	Контрольное событие	Дата	Результат
1	Сбор предвратных данных	01.03.20	Раздел Вкр
2	Создание модуля программного моделирования	01.09.20	Раздел ВКР
3	Создание подпрограммы хранения данных	15.12.20	Раздел ВКР
4	Создание программы управления моделью	01.03.21	Раздел ВКР
5	Объединение модулей в единый программный пакет	01.05.21	Раздел ВКР
6	Формирование отчёта по разработке	04.06.21	ВКР

3.3.3 План проекта

Составим линейный график работ по проекту (таблица 15). В котором отразим даты начала и окончания, длительность и ответственных лиц по каждому этапу работ.

Таблица 15 – Календарный план проекта

Код работы	Название	Длитель-ность, дни	Дата начала	Дата окончания	ФИО
1.1	Получение индивидуального задания	7	10.01.2020	17.01.2020	Филипас А.А. Никитин А.С.
1.2	Планирование работ	7	17.01.2020	24.01.2020	Филипас А.А. Никитин А.С.
1.3.1	Изучение литературных источников	19	24.01.2020	12.02.2020	Никитин А.С.
1.3.2	Изучение технологий	18	12.02.2020	01.03.2020	Никитин А.С.
2.1	Разработка алгоритмов моделирования	60	01.03.2020	30.04.2020	Никитин А.С.
2.2	Проектирование программы	60	30.04.2020	29.06.2020	Никитин А.С.
2.3	Проектирование интерфейса программы	21	29.06.2020	20.07.2020	Никитин А.С.

Продолжение таблицы 15

Код работы	Название	Длительность, дни	Дата начала	Дата окончания	ФИО
2.4	Кодирование программы	43	20.07.2020	01.09.2020	Никитин А.С.
3.1	Проектирование структуры БД	21	01.09.2020	22.09.2020	Никитин А.С.
3.2	Развертывание БД	7	22.09.2020	29.09.2020	Никитин А.С.
3.3	Проектирование программы доступа к данным	28	29.09.2020	27.10.2020	Никитин А.С.
3.4	Проектирование интерфейса программы	21	27.10.2020	17.11.2020	Никитин А.С.
3.5	Кодирование программы	28	17.11.2020	15.12.2020	Никитин А.С.
4.1.2.1	Проектирование серверной логики	14	15.12.2020	29.12.2020	Никитин А.С.
4.1.1.2	Проектирование структуры обработки запросов	7	29.12.2020	05.01.2021	Никитин А.С.
4.1.2.1	Проектирование клиентской логики	14	05.01.2021	19.01.2021	Никитин А.С.
4.1.2.2	Разработка графических интерфейсов	7	19.01.2021	26.01.2021	Никитин А.С.
4.2	Проектирование интерфейса программы	7	26.01.2021	02.02.2021	Никитин А.С.
4.3	Кодирование программы	27	02.02.2021	01.03.2021	Никитин А.С.
5.1	Объединение модулей	45	01.03.2021	15.04.2021	Никитин А.С.
5.2	Тестирование	16	15.04.2021	01.05.2021	Никитин А.С.
5.3	Документирование разработки	34	01.05.2021	04.06.2021	Никитин А.С.

Отразим план график в виде диаграммы Ганта (рис. 11).

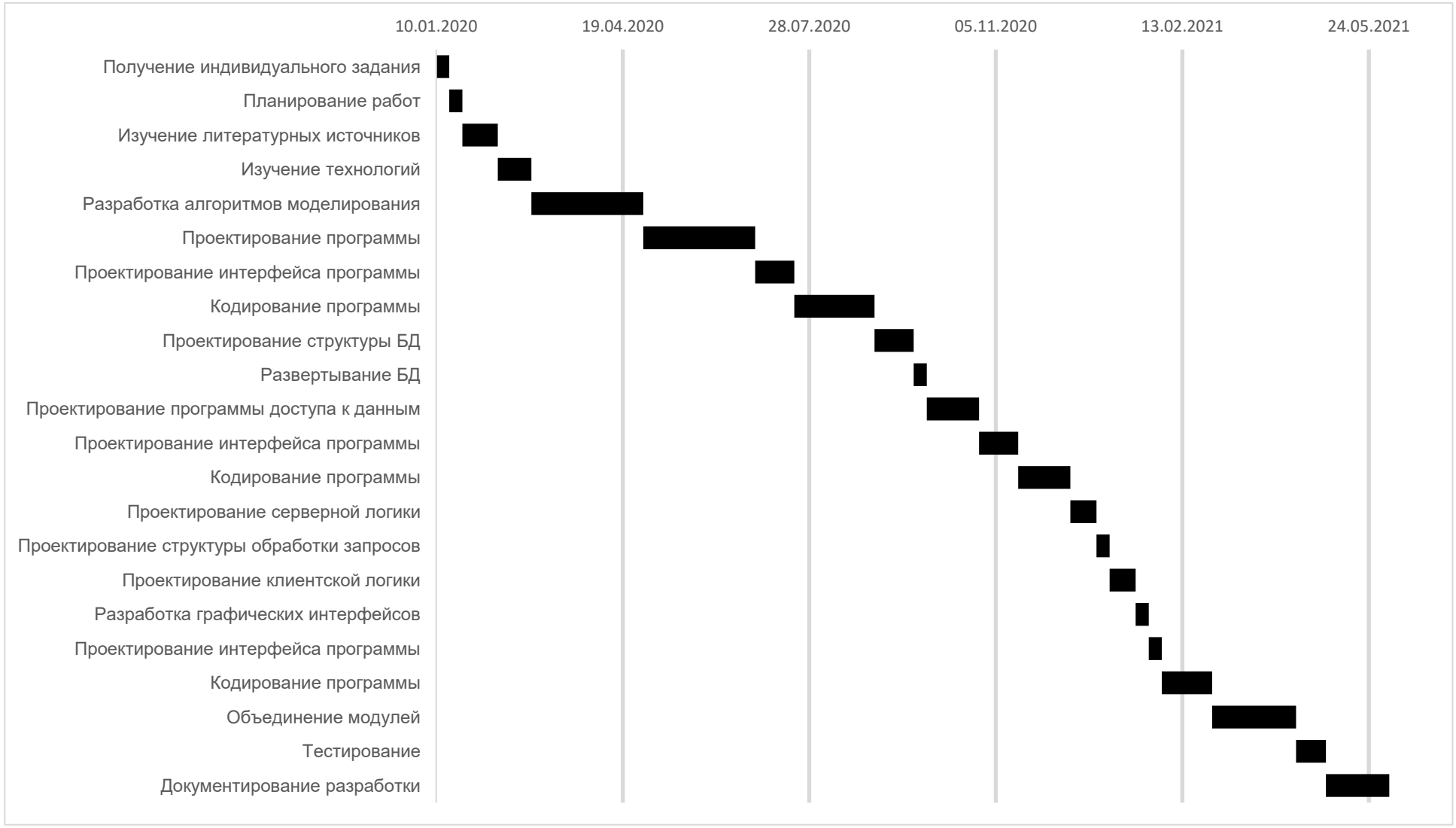


Рисунок 11 – Диаграмма Ганта по проекту

3.3.4 Бюджет НТИ

Бюджет научного исследования должен в полной мере отражать все планируемые расходы на его выполнение. Бюджет формируется по следующим статьям: сырьё и материалы, специальное оборудование, основная заработная плата, дополнительная заработная плата, отчисления в социальные фонды, командировки, оплата работ сторонних организаций.

Сырьё и материалы

Разработка программного обеспечения проводится на персональном компьютере без использования материального сырья. Расчёт данной статьи расходов не требуется.

Специальное оборудование

В качестве специального оборудования подразумеваются лицензии на ПО и затраты на приобретение ПК для разработки.

№ п/п	Наименование	Кол-во, шт.	Цена единицы, руб.	Общая стоимость, руб.
1	Лицензия Microsoft Visual Studio на месяц	17	2 812,50	47 812,50
2	Лицензия Asure DevOps на месяц	17	375,00	6 375,00
3	ПК	1	120 000,00	120 000,00
Итог:				174 187,50

Основная заработная плата

Заработная плата работника включает основную заработную плату, и дополнительную заработную плату:

$$З_{зп} = З_{осн} + З_{доп}. \quad (3.10)$$

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

$З_{доп}$ – дополнительная заработная плата (12-20 % от $З_{осн}$).

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

$$З_{осн} = З_{дн} \cdot T_p . \quad (3.11)$$

где $З_{осн}$ – основная заработная плата одного работника, руб.;

$Т_р$ – продолжительность работ, выполняемых научно-техническим работником, раб. дн. (таблица);

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

Среднедневная заработная плата рассчитывается по формуле:

$$З_{дн} = \frac{З_м \cdot М}{F_д} . \quad (3.12)$$

где $З_м$ – месячный должностной оклад работника, руб.;

М – количество месяцев работы без отпуска в течение года: при отпуске в 48 раб. дня М =10,4 месяца, 6-дневная неделя;

$F_д$ – действительный годовой фонд рабочего времени научно-технического персонала, рабочие дни (таблица 16).

Таблица 16 – Баланс рабочего времени

Показатели рабочего времени	Руководитель	Студент
Календарное число дней	365	365
Количество нерабочих дней - выходные дни - праздничные дни	118	118
Потери рабочего времени - отпуск - невыходы по болезни	48	48
Действительный годовой фонд рабочего времени	199	199

Месячный должностной оклад работника:

$$З_м = З_{тс} \cdot (1 + k_{пр} + k_д) \cdot k_р . \quad (3.13)$$

где $З_{тс}$ – заработная плата по тарифной ставке, руб.;

$k_{пр}$ – премиальный коэффициент, равный 0,3 (т.е. 30% от $З_{тс}$);

$k_д$ – коэффициент доплат и надбавок составляет примерно 0,2 – 0,5 (в НИИ и на промышленных предприятиях – за расширение сфер обслуживания, за профессиональное мастерство, за вредные условия: 15-20 % от $З_{тс}$);

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

Для предприятий, не относящихся к бюджетной сфере, тарифная заработная плата (оклад) рассчитывается по тарифной сетке, принятой на данном предприятии. Расчёт основной заработной платы приведён в таблице .

Таблица 17 – Расчёт основной заработной платы

Исполнители	$З_{тс}$, руб.	$k_{пр}$	$k_{д}$	$k_{р}$	$З_{м}$, Руб	$З_{дн}$, руб.	$Т_{р}$, раб.	$З_{осн}$, руб.
Руководитель	33 664,00	0,3	0,15	1,3	63 456,64	3 316,33	14	46 428,58
Студент	2 500,00	0,3	0,15	1,3	4 712,50	246,28	511	125 849,80
Итого:								172 278,38

Дополнительная заработная плата

Расчет дополнительной заработной платы ведется по следующей формуле:

$$З_{доп} = k_{доп} \cdot З_{осн}. \quad (3.14)$$

где $k_{доп}$ – коэффициент дополнительной заработной платы (на стадии проектирования принимается равным 0,12 – 0,15).

Примем коэффициент равный 0,12. Результаты расчета приведены в таблице 18

Таблица 18 – Расчёт дополнительной заработной платы

Исполнители	Дополнительная заработная плата, руб
Руководитель	5 571,43
Студент	15 101,98
Итого	20 673,41

Отчисления во внебюджетные фонды

Величина отчислений во внебюджетные фонды определяется исходя из следующей формулы:

$$З_{внеб} = k_{внеб} \cdot (З_{осн} + З_{доп}). \quad (3.15)$$

где $k_{внеб}$ – коэффициент отчислений на уплату во внебюджетные фонды (пенсионный фонд, фонд обязательного медицинского страхования и пр.). Результаты расчета приведены отчислений и общий итог по фонду заработной платы приведены в таблице 19.

Таблица 19 – Общий итог по зарплатному фонду

ч	Коэффициент отчислений во внебюджетные фонды	Основная заработная плата, руб.	Дополнительная заработная плата, руб.	Отчисления во внебюджетные фонды
Руководитель проекта	0,3	46 428,58	5 571,43	15 600,00
Студент		125 849,80	15 101,98	42 285,53
Итог:		172 278,38	20 673,41	57 885,53

Научные и производственные командировки

Данный вид работ не запланирован, расчёт статьи расходов не требуется.

Контрагентные расходы

Привлечение сторонних организаций не запланировано, расчёт статьи расходов не требуется.

Накладные расходы

Накладные расходы учитывают прочие затраты организации, не попавшие в предыдущие статьи расходов: печать и ксерокопирование материалов исследования, оплата услуг связи, электроэнергии, почтовые и телеграфные расходы, размножение материалов и т.д. Их величина определяется по следующей формуле:

$$З_{накл} = k_{накл} \cdot (З_{осн} + З_{доп}) . \quad (3.16)$$

где $k_{накл}$ – коэффициент, учитывающий накладные расходы.

Примем коэффициент равным 0,8. Таким образом сумма накладных расходов составит (таблица):

Таблица 20 – Расчёт накладных расходов

Исполнитель	Коэффициент учёта накладных расходов	Основная заработная плата, руб.	Дополнительная заработная плата, руб.	Накладные расходы
Руководитель проекта	0,8	46 428,58	5 571,43	41 600,00
Студент		125 849,80	15 101,98	112 761,42
Итог:		172 278,38	20 673,41	154 361,42

Бюджет НТИ

Сведём статьи, рассчитанные в предыдущих пунктах, в таблицу 21.

Таблица 21 – Итоговый бюджет НТИ

Статьи							
Сырье и материалы, руб.	Специальное оборудование, руб.	Основная заработная плата, руб.	Дополнительная заработная плата, руб.	Отчисления на социальные нужды, руб.	Научные и производственные, руб.	Контрагентные расходы, руб.	Накладные расходы, руб
0,00	174 187,50	172 278,38	20 673,41	57 885,53	0,00	0,00	154 361,42
Итог:							579 386,24

3.3.5 Риски проекта

Составим реестр возможных рисков, сведём его в таблицу 22.

Таблица 22 – Реестр рисков проекта

№	Риск	Потенциальное воздействие	Вероятность	Влияние риска	Уровень риска*	Способы смягчения риска	Условия наступления
1	Несо-ответствие модели реальным процессам	Увеличение сроков разработки	2	5	Сред-ний	Привлечение дополнительных специалистов	Ошибки в разработке мат. алгоритмов
2	Крити-ческие ошибки прило-жения	Отказ приложения	1	5	Сред-ний	Привлечение дополнительных специалистов	Ошибки в разработке приложения

Продолжение таблицы 22

№	Риск	Потенциальное воздействие	Вероятность	Влияние риска	Уровень риска*	Способы смягчения риска	Условия наступления
3	Невозможность доступа к приложению	Невозможность пользования приложением	1	3	Низкий	Нет	Отказ сетей связи
	Срыв сроков разработки	Увеличение сроков разработки	2	2	Средний	Привлечение дополнительных специалистов	Нарушение сроков разработки

3.4 Определение ресурсной, финансовой, экономической эффективности

Сравнительная эффективность разработки выражается в интегральном показателе эффективности. Этот показатель состоит из двух средневзвешенных величин:

Определение интегральных показателей эффективности проведём в сравнении со следующими аналогами:

1. ПО имитационного моделирования AnyLogic компании AnyLogic;
2. ПО имитационного моделирования MATLAB компании The MathWorks;

Интегральный финансовый показатель разработки определяется как:

$$I_{\text{финр}}^{исп.i} = \frac{\Phi_{pi}}{\Phi_{max}}. \quad (3.17)$$

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

Φ_{pi} – стоимость *i*-го варианта исполнения;

Φ_{max} – максимальная стоимость исполнения научно-исследовательского проекта.

Результаты вычислений приведены в таблице 23.

Таблица 23 – Расчёт интегрального финансового показателя

Продукт	$Ф_{pi}$	$Ф_{max}$	$I_{финр}^{исп.i}$
НТИ	150 000,00	1 560 000,00	0,10
AnyLogic	1 560 000,00		1,00
MATLAB	173 000,00		0,11

Интегральный показатель ресурсоэффективности вариантов исполнения объекта исследования можно определить следующим образом:

$$I_{pi} = \sum a_i \cdot b_i . \quad (3.18)$$

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

a_i – весовой коэффициент *i*-го варианта исполнения разработки;

b_i – бальная оценка *i*-го варианта исполнения разработки, устанавливается экспертным путем по выбранной шкале оценивания;

n – число параметров сравнения.

Расчёт показателя приведён в таблице 24.

Таблица 24 – Сравнительная оценка характеристик продуктов

Критерии	Весовой коэффициент параметра	НТИ	AnyLogic	MATLAB
1. Надежность алгоритмов	0,3	3	5	5
2. Быстродействие	0,3	4	5	3
3. Удобство	0,1	4	5	5
4. Функциональность	0,2	2	4	5
5. Интерфейс	0,1	5	5	3
ИТОГО	1	3,4	4,8	4,2

Интегральный показатель эффективности вариантов исполнения разработки ($I_{испi.}$) определяется на основании интегрального показателя ресурсоэффективности и интегрального финансового показателя по формуле:

$$I_{исп.1} = \frac{I_{р-исп1}}{I_{финр}^{исп.1}}, \quad I_{исп.1} = \frac{I_{р-исп1}}{I_{финр}^{исп.1}} \text{ и т.д.} \tag{3.19}$$

Сравнение интегрального показателя эффективности вариантов исполнения разработки позволит определить сравнительную эффективность проекта (таблица 25) и выбрать наиболее целесообразный вариант из предложенных.

Таблица 25 – Расчёт интегрального показателя эффективности

	$I_{\text{финр}}^{исп.i}$	$I_{р-исп}$	$I_{исп}$
НТИ	0,10	3,4	35,36
AnyLogic	1,00	4,8	4,80
MATLAB	0,11	4,2	37,87

Сравнительную эффективность проекта определим по следующей формуле:

$$Э_{ср} = \frac{I_{финр}^{р}}{I_{финр}^{а}}; \tag{3.20}$$

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

$I_{тэ}^{р}$ – интегральный показатель разработки;

$I_{тэ}^{а}$ – интегральный технико-экономический показатель аналога.

Таким образом сравнительная эффективность проекта с аналогами:

1. В сравнении с AnyLogic $Э_{ср} = 7,37$;
2. В сравнении с MATLAB $Э_{ср} = 0,93$;

Вывод

В ходе проделанной работы над данным разделом можно сделать вывод о целесообразности разработки. В текущем состоянии разработка не является лучшим вариантом решением поставленной задачи, но имеет свои преимущества перед конкурентными решениями. Выявлены направления возможного развития проекта для повышения его конкурентоспособности.

4 Социальная ответственность

С каждым годом возрастает интенсивность применения компьютерной техники в сферах жизнедеятельности человека. При работе с компьютером человек подвергается воздействию ряда опасных и вредных производственных факторов: электромагнитных полей, радиочастотному (высоких, ультравысоких и средних частот) и инфракрасному излучению, шуму и вибрации, статическому электричеству. Работа с компьютером характеризуется значительным умственным напряжением, высокой напряженностью зрительной работы и большой нагрузкой на кисти рук при работе с периферийными устройствами ЭВМ.

4.1 Правовые и организационные вопросы обеспечения безопасности

4.1.1 Особенности законодательного регулирования проектных решений

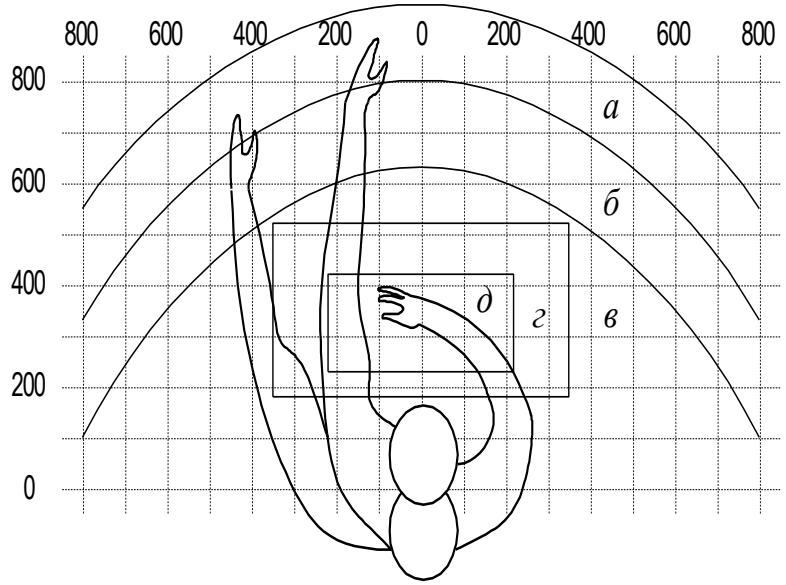
Государственный надзор и контроль в организациях независимо от организационно-правовых форм и форм собственности осуществляют специально уполномоченные на то государственные органы и инспекции в соответствии с федеральными законами.

К таким органам относятся Федеральная инспекция труда, Государственная экспертиза условий труда Федеральная служба по труду и занятости населения (Минтруда России Федеральная служба по экологическому, технологическому и атомном надзору (Госгортехнадзор, Госэнергонадзор, Госатомнадзор России) Федеральная служба по надзору в сфере защиты прав потребителей и благополучия человека (Госсанэпиднадзор России) и др.

Так же в стране функционирует Единая государственная система предупреждения и ликвидации чрезвычайных ситуаций (РСЧС), положение о которой утверждено Постановлением Правительства Российской федерации [21], в соответствии с которым, система объединяет органы управления, силы и средства.

4.1.2 Организационные мероприятия при компоновке рабочей зоны

4.1.2.1 Эргономические требования к рабочему месту



а - зона максимальной досягаемости;

б - зона досягаемости пальцев при вытянутой руке;

в - зона легкой досягаемости ладони;

г - оптимальное пространство для грубой ручной работы;

д - оптимальное пространство для тонкой ручной работы.

Рисунок 12 – зоны досягаемости рук в горизонтальной плоскости

На рисунке 13 показан пример размещения основных и периферийных составляющих ПК на рабочем столе разработчика.

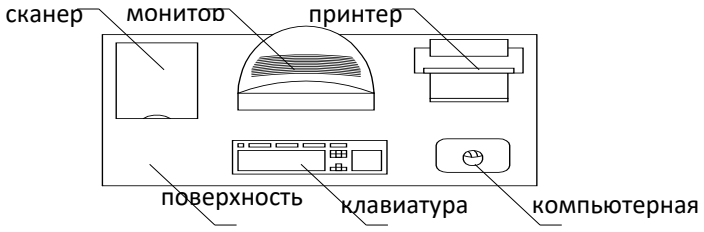


Рисунок 13 – Размещение основных и периферийных составляющих ПК

Для комфортной работы эргономика рабочего пространства должна удовлетворять следующим требованиям [23].

На рабочем месте оборудованы рабочие столы, которые соответствуют рисунку 13.

4.2 Производственная безопасность

Ниже приведем перечень опасных и вредных факторов, характерных для проектируемой производственной среды в виде таблицы (таблица 26).

Таблица 26 – Опасные и вредные факторы при выполнении работ по разработке программно-алгоритмического комплекса планирования производственных процессов.

Источник фактора, наименование видов работ	Факторы *(по ГОСТ 12.0.003-2015)*		Нормативные документы
	Вредные	Опасные	
Разработка комплекса управления: 1) Разработка системы управления в учебной аудитории; 2) Проверка работы симулятора в учебных целях в учебной аудитории; 3) Эксплуатация симулятора в учебной аудитории.	1) Недостаточная освещённость рабочей зоны; отсутствие или недостаток естественного света; 2) Повышенный уровень шума; 3) Повышенный уровень электро-магнитных излучений; 4) Повышенная или пониженная влажность воздуха; 5) Статические перегрузки; 6) Умственные перегрузки, перегрузки анализаторов.	1) Электрический ток; 2) Короткое замыкание. 3) Статическое электричество 4) Повышенная напряжённость электрического поля;	ГОСТ 12.0.003-2015 Система стандартов безопасности труда (ССБТ). Опасные и вредные производственные факторы. Классификация. СанПиН 2.2.2/2.4.1340-03 Гигиенические требования к персональным электронно-вычислительным машинам и организации работы. СНиП 23-05-95 Естественное и искусственное освещение. ГОСТ Р 50377-92 (МЭК 950-86) Безопасность оборудования информационной технологии, включая электрическое конторское оборудование.

4.2.1 Недостаточная освещённость рабочей зоны; отсутствие или недостаток естественного света;

Рабочее место находится на цокольном этаже здания. Естественное освещение в аудитории отсутствует. Освещение в аудитории производится посредством общего искусственного освещения.

В процессе разработки комплекса управления, разработчик все время работы пользуется ПК. При этом согласно [11] освещение не создает бликов на поверхности экрана ПЭВМ, а освещенность экрана не более 300 лк. Кроме того, работа за ПЭВМ относится к категории зрительных работ к разряду Б зрительных работ (восприятие информации) [12].

В помещении, которое предназначено для эксплуатации ПЭВМ, искусственное освещение осуществляется путем системы общего освещения. Освещение не создает бликов и ограничивает прямую блёскость от источников освещения на поверхности экрана ПВЭМ [11].

4.2.2 Повышенный уровень шума

Звуковые колебания, издаваемые движущимися частями механизмов и приборов, могут воздействовать на здоровье человека. Громкие звуки, могут стать причиной проблем со слухом, а длительное воздействие шума более 80 дБ может стать причиной его потери ли ухудшения. Чувствительность к монотонным звукам является индивидуальным показателем. Но постоянно повторяющиеся шумы на рабочем месте провоцируют проблемы, связанные с нервной системой и органами слуха.

В данной работе основным источником шума является системный блок ПЭВМ, внутри которого работает система охлаждения, состоящая из вентиляторов, воспроизводящих непрерывный шелест или гудение.

Уровень звука, при этом не превышает 50дБ [14]. Постоянный уровень шума влияет на работоспособность и сосредоточенность человека. рабочее место соответствует нормам [14] и является помещением с минимальным уровнем шума при программировании и разработке программного обеспечения планирования производственных процессов. Кроме того, каждый академический час в работе делается перерыв, который позволяет отключить компьютер и/или выйти из помещения для разгрузки нервной системы и органов слуха.

Характеристикой постоянного шума на рабочем месте является уровень звукового давления, определяемый по формуле:

$$L_{\text{а}} = 20\lg\frac{P_a}{P_0}. \qquad (4.21)$$

где P_a среднеквадратичная величина звукового давления;

P_0 – исходное звуковое давление в воздухе, равное $2\cdot10^{-5}$ Па

Замеры звукового давления на рабочем месте показали $P_a = 385 \cdot 10^{-5}$. Согласно формуле 4.21:

$$L_a = 20 \lg \frac{385 \cdot 10^{-5}}{2 \cdot 10^{-5}} = 37.73;$$

Предельно допустимые уровни звука на рабочих местах с учётом тяжести труда представлены в таблице 27 [14]:

Таблица 27 – Допустимые уровни звука на рабочем месте

Категория напряженности трудового процесса	Категория тяжести трудового процесса				
	легкая физическая нагрузка	средняя физическая нагрузка	тяжелый труд 1 степени	тяжелый труд 2 степени	тяжелый труд 3 степени
Напряженность легкой степени	80	80	75	75	75
Напряженность средней степени	70	70	65	65	65
Напряженный труд 1 степени	60	60	-	-	-
Напряженный труд 2 степени	50	50	-	-	-

Можно сделать вывод, что уровень шума допустимый для лёгкой физической нагрузки и напряжённого труда 2.

4.2.3 Повышенный уровень электромагнитных излучений

В повседневной жизни для людей не заметно воздействия электромагнитных излучений. Мощность источника излучения должна быть достаточно большой, чтобы это отражалось на здоровье и самочувствие организма.

Основные излучающие электромагнитное поле части ПЭВМ – это системный блок, в котором находится процессор, и экран монитора [15].

На рабочем месте установлены ПЭВМ, оснащённые жидкокристаллическим монитором. Они излучают электромагнитные волны, которые не причиняют человеку вреда, даже при длительной работе [15].

4.2.4 Повышенная напряжённость электрического поля;

Работа ПЭВМ, кроме электромагнитных излучений сопровождается электростатическим полем, которое может деионезировать окружающую воздушную оболочку. Для обеспечения безопасности рабочего персонала и оборудования необходимо проводить ионизацию воздуха, путём установки ионизаторов или обеспечения проветривания помещения.

4.2.5 Повышенная или пониженная влажность воздуха;

Давление, температура и влажность воздуха влияют на здоровье работников, следовательно, он влияют на общее самочувствие, работоспособность и выполнение поставленных задач.

Рабочее место из-за своего расположения может быть слишком влажным. Поэтому в стенах аудитории установлена механическая вентиляция, оснащенная вентилятором и отводящая воздух из помещения. Это обеспечивает проветриваемость помещения и как следствие убирает лишнюю влагу. Кроме того, аудитория подключена к системе центрального городского отопление. Это помогает регулировать температуру в период зимних месяцев.

4.2.6 Статические перегрузки;

4.2.6.1 Умственные перегрузки, перегрузки анализаторов

В современном мире, почти каждая работа, так или иначе, связанна с работой за компьютером. Разработчикам программ, инженерам и всем, кто учится, приходится проводить за ПВЭМ многие часы. При этом пользователь вынужден принимать одну и ту же позу в течение длительного времени, тем самым создавая в работе мышечного корсета статические перегрузки. Неудобная поза, нахождение центра тяжести в одном месте, постоянный наклон вперед вызывают боли в шее и спине.

Работа с ПЭВМ подразумевает обработку большого количества информации. Анализ данных, инженерные исследования, расчеты и разработка

программных продуктов требуют высокой концентрации внимания. При работе с визуальной информацией напрягаются глаза, которые являются зрительными анализаторами человека. Расстояние расположения предмета постоянного визуального контроля не меняется в процессе работы, из-за этого устают глазные мышцы, из-за этого снижается острота зрения. При длительных контактах с дисплеем, постоянного наблюдения схожей по структуре зрительной информации, человек начинает испытывать стресс.

Согласно [17], разработка программно-алгоритмического комплекса относится к группе В, I категории (до 2х часов) – творческая работа в режиме диалога с ПЭВМ. При выполнении разных групп работ в течение смены за основную принимают такую, которая занимает не менее 50% времени рабочего дня. Для обеспечения оптимальной работоспособности и сохранения здоровья пользователей на протяжении рабочей смены должны устанавливаться регламентированный перерыв, при 8-ми часовом рабочем дне 30 минут. Продолжительность непрерывной работы с ПК не должна превышать 2 часов. Для I категории работ - через 2 часа от начала работы и через 1,5 - 2 часа после обеденного перерыва продолжительностью 15 минут каждый [17].

Во время регламентированных перерывов с целью сохранения высокой работоспособности выполняется комплекс упражнений [17]. С целью уменьшения отрицательного влияния монотонности целесообразно чередование операций осмысленного текста и числовых данных, чередование редактирования текстов и ввода данных (изменение содержания работы).

Работа над разработкой программно-алгоритмического комплекса требует сосредоточенности и частого переключения между использованием нескольких программ одновременно. Поэтому для того чтобы зрительные анализаторы работали на нужном уровне каждый академический час проводится перерыв в 5-10 минут, а в каждый второй академический час перерыв в 20 минут. Во время перерыва есть возможность выйти из аудитории и выключить на время ПЭВМ [17].

4.2.7 Электробезопасность

Электробезопасность – система организационных и технических мероприятий и средств, обеспечивающих защиту людей от вредного и опасного воздействия электрического тока, электрической дуги, электромагнитного поля и статического электричества.

Согласно [18] помещение, в котором находится рабочее место, относится к категории помещений без повышенной опасности. Его можно охарактеризовать, как сухое, непыльное, с токонепроводящими полами и нормальной температурой воздуха. Температурный режим, влажность воздуха, химическая среда не способствуют разрушению изоляции электрооборудования.

Защита от электрического тока на рабочем месте производится с помощью изоляции токопроводящих частей (все провода изолированы). Электрические устройства, в частности процессор от ПЭВМ расположен в защитном коробе.

Короткое замыкание – это соединение двух точек с разным потенциалом с последующим увеличением тока и выделением большого количества тепла. Вследствие чего короткое замыкание может стать причиной пожара в помещении, при коротком замыкании от электрического тока могут пострадать люди, находящиеся в непосредственной близости от источника возникновения.

На рабочем месте короткое замыкание может быть вызвано либо неисправностью в проводке, либо при работе с компьютером, когда внутри корпуса создается разность фаз и ток может так же повредить всю электросеть [25].

Для защиты электрической сети от короткого замыкания предусмотрены устройства защитного отключения (УЗО), оснащенные устройствами автоматического отключения – автоматами и предохранителями. Кроме того, в помещении установлены датчики дыма, которые при возникновении возгорания, вызванного коротким замыканием, оповещают все здание о начавшемся пожаре.

Таким образом, рабочее место полностью защищено от возможного короткого замыкания.

4.2.8 Статическое электричество

В рабочем пространстве находится много устройств, которые работают от электрического тока и сделаны из материалов, которые, так или иначе, накапливают на себе статически заряд. Может возникнуть разность потенциалов от статического электричества, и прикосновение человека к заряду может вызвать травмы, ожоги или пожар [19].

Для защиты оборудования и персонала применяется общее заземление электропроводки и увлажняющие устройства.

4.3 Экологическая безопасность

4.3.1 Воздействие на литосферу

При выходе из строя, а также устаревании компонентов, ПЭВМ начинает представлять собой источник второсортного сырья. Каждый ПЭВМ содержит цветные металлы и целый набор опасных для окружающей среды веществ. Это производные газов, тяжелые металлы, среди которых кадмий, ртуть и свинец. Попадая на свалку, все эти вещества под воздействием внешней среды постепенно проникают в почву.

Люминесцентные лампы при перегорании становятся источником загрязнения. Лампы содержат внутри себя ртуть, которая загрязняет окружающую среду. Кроме того, их корпус состоит преимущественно из стеклянной трубки, которая при неосторожном обращении может разбиться на мелкие осколки.

Документы, перенесенные на бумагу, становятся источником бумажных отходов. Такие отходы медленнее разлагаются из-за предварительной обработки бумаги, а также содержат на себе следы краски, которая, попадая в почву ее загрязняет.

Юридические лица имеют право утилизировать оргтехнику только при прохождении процедуры полного списания, подтвержденного актом [24].

Томский политехнический университет является юридическим лицом, поэтому перегоревшие люминесцентные лампы собираются техническим персоналом, а затем передаются в центр по переработке таких ламп, у которого имеется лицензия на право сбора и переработки люминесцентных ламп [24]. Для макулатуры существуют специально установленные контейнеры, в которые помещаются отработавшие печатные издания, офисная бумага и другие изделия из переработанной целлюлозы. Они отвозятся в пункты по сбору макулатуры, где утилизируются.

4.4 Безопасность в чрезвычайных ситуациях

4.4.1 Пожарная безопасность

Пожарная безопасность может быть обеспечена мерами пожарной профилактики и активной пожарной защиты. Пожарная профилактика включает комплекс мероприятий, направленных на предупреждение пожара или уменьшение его последствий. Возникновение пожара в помещении аудитории может привести к большим материальным потерям и возникновению чрезвычайной ситуации. Чрезвычайные ситуации приводят к полной потере информации и большим трудностям восстановления всей информации в полном объёме.

Согласно нормам технологического проектирования [20], данное помещение относится к категории В [21], производства, связанные с обработкой или применением твердых сгораемых веществ и материалов.

В случае возникновения пожара необходимо отключить электропитание, вызвать по телефону пожарную команду, произвести эвакуацию и приступить к ликвидации пожара огнетушителями. При наличии небольшого очага пламени можно воспользоваться подручными средствами с целью прекращения доступа

воздуха к объекту возгорания. Покидать помещение необходимо согласно плану эвакуации.

Пожар будет являться чрезвычайной ситуацией для людей, находящихся в помещении. При возникновении пожара сработают датчики дыма, которые подадут сигнал общего оповещения всего здания.

Вывод:

В ходе исследования по социальной ответственности было выявлено три источника вредных и опасных факторов: разработка системы планирования в учебной аудитории; проверка работы системы на модели производственного цикла.

Исходя из этого, были выделены и проанализированы вредные и опасные факторы. Для них были установлены средства, которые помогают защитить человека, который находится в данном помещении, от выявленных вредных и опасных факторов.

Анализу были подвергнуто влияние работы на рабочем месте на окружающую среду. Установлено, что после работы остаются бумажные отходы и перегоревшие люминесцентные лампы, которые при неправильной утилизации будут влиять на литосферу. Выявлено, что отходы утилизируются согласно ГОСТ.

Кроме того, исследовались правовые и организационные вопросы обеспечения безопасности и организационные мероприятия при компоновке рабочей зоны.

Заключение

В ходе работы была создана система моделирования дискретно событийных производственных процессов. Данная работа является продолжением работы «Разработка и реализация алгоритма планирования производственного процесса» [10] и открывает перспективу объединения результатов этих работ для создания комплексной системы планирования производственных процессов.

В ходе работы исследованы меды моделирования производственных процессов. Для реализации работы был выбран метод имитационного моделирования.

На основе теоретической информации были разработаны математические модели различных типов обработки изделий и метод моделирования «поток событий». Для реализации модели спроектирована иерархическая структура классов и их функции. Модель закодирована на языке C++ и скомпилирована в виде подключаемой библиотеки.

Для работы с моделью разработаны модули окружения:

Модуль доступа к данным, обеспечивающий обмен данными между базой данных и приложением, использована БД PostgreSQL и модуль библиотеки .Net Entity Framework.

Web приложение, инкапсулирующее другие модули и являющееся интерфейсом для работы пользователя. При разработке модуля использован паттерн MVC.

Список источников

1. Имакаева Дария Амангельдиевна Имитационное моделирование при экономической оптимизации // Проблемы экономики и юридической практики. 2017. №4. URL: https://cyberleninka.ru/article/n/imitatsionnoe-modelirovanie-pri-ekonomicheskoy-optimizatsii (дата обращения: 29.05.2021).

2. Анализ рынка ERP систем в России от делового портала TAdviser [Электронный ресурс]. – Режим доступа: www.tadviser.ru/index.php/ERP – Заглавие с экрана – (Дата обращения 03.04.2020).

3. Анализ рынка MES систем в России от делового портала TAdviser [Электронный ресурс]. – Режим доступа: www.tadviser.ru/index.php/MES – Заглавие с экрана – (Дата обращения 03.04.2020).

4. Таха, Х. А. Введение в исследование операций / Х. А. Таха. — Москва : Издательский дом "Вильямс", 2005.

5. Волкова, В. Н. Теория систем и системный анализ: учебник для академического бакалавриата / В. Н. Волкова, А. А. Денисов. — Москва : Юрайт, 2014.

6. Кочегурова Е.А. Теория и методы оптимизации: учебное пособие / Е.А. Кочегурова. – Томск: Изд-во Томского политехнического университета, 2012. – 157с.

7. Сухарев А.Г. Курс методов оптимизации: учебное пособие / А.Г. Сухарев, А.В. Тимохов, В.В. Фёдоров. – 2-е изд. – М.: ФИЗМАТЛИТ, 2005. – 368 с.

8. Кузнецов, А. В. Высшая математика. Математическое программирование : учебник / А. В. Кузнецов, В. А. Сакович, Н. И. Холод. — 4-е изд., стер. — Санкт-Петербург : Лань, 2013. — 352 с.

9. Алиев, Т. И. Основвы моделирования дискретных систем / Т. И. Алиев. — СПб : СПбГУ ИТМО, 2009. – 363 с.

10. Никитин А. С. Разработка и реализация алгоритма планирования производственного процесса : бакалаврская работа / А. С. Никитин ; Национальный исследовательский Томский политехнический университет (ТПУ), Инженерная школа информационных технологий и робототехники (ИШИТР), Отделение автоматизации и робототехники (ОАР) ; науч. рук. А. А. Филипас. — Томск, 2019.

11. СанПиН 2.2.2/2.4.1340-03 Гигиенические требования к персональным электронно-вычислительным машинам и организации работы.

12. СНиП 23-05-95 Естественное и искусственное освещение.

13. Гигиенические требования к естественному, искусственному и совмещенному освещению жилых и общественных зданий.

14. СН 2.2.4/2.1.8.562 – 96. Шум на рабочих местах, в помещениях жилых, общественных зданий и на территории застройки.

15. СанПиН 2.2.2/2.4.1340-03 Гигиенические требования к персональным электронно-вычислительным машинам и организации работы.

16. ГОСТ 30494—2011 Здания жилые и общественные. Параметры микроклимата в помещениях.

17. Инструкция по организации работ, охране труда и экологической безопасности при работе на ПЭВМ /ПК/ в издательствах и на полиграфических предприятиях Госкомпечати России.

18. ГОСТ Р 50377-92 (МЭК 950-86) Безопасность оборудования информационной технологии, включая электрическое конторское оборудование.

19. ГОСТ 12.2.032-78 Система стандартов безопасности труда (ССБТ). Рабочее место при выполнении работ сидя. Общие эргономические требования.

20. НПБ 105-03. Нормы пожарной безопасности. Определение категорий помещений, зданий и наружных установок по взрывопожарной и пожарной опасности.

21. Технический регламент «о требованиях пожарной безопасности» [Электронный ресурс] / Единая справочная служба Консорциума «Кодекс». – URL: http://ezproxy.ha.tpu.ru:2065/docs/, свободный – Загл. с экрана. Язык русс. Дата обращения: 3.04.2018 г.

22. Трудовой кодекс Российской Федерации от 30.12.2001 N 197-ФЗ.

23. СанПиН 2.2.2/2.4.1340 – 03. Санитарно-эпидемиологические правила и нормативы «Гигиенические требования к персональным электронно-вычислительным машинам и организации работы». – М.: Госкомсанэпиднадзор, 2003.

24. Федеральный закон об отходах производства и потребления.

25. ГОСТ 12.1.030–81 ССБТ. Защитное заземление, зануление.

Список публикаций

1. Никитин А. С. Система автоматизированного управления производственным комплексом, разработанная на базе пакета Matlab / А. С. Никитин // Молодежь и современные информационные технологии : сборник трудов XVII Международной научно-практической конференции студентов, аспирантов и молодых учёных, 17-20 февраля 2020 г., г. Томск. — Томск : Изд-во ТПУ, 2020. — [С. 317-318].

2. Никитин А. С. Разработка библиотеки моделирования дискретнособытийных потоковых производственных процессов / А. С. Никитин // Современные проблемы машиностроения : сборник трудов XIII Международной научно-технической конференции, г. Томск, 26-30 октября 2020 г. — Томск : Томский политехнический университет, 2020. — [С. 212-213].

Приложение А

(справочное)

Раздел ВКР на английском языке

Introduction, Project description, Manufacturing process model, Types of product processing, Simulation algorithmization, Simulation algorithm

Студент:

Группа	ФИО	Подпись	Дата
8ТМ91	Никитин Андрей Сергеевич		

Руководитель ВКР

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Доцент ОАР	Филипас Александр Александрович	к.т.н.		

Консультант – лингвист ШБИП ОИЯ

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Доцент ШБИП	Сидоренко Татьяна Валерьевна	к.п.н.		

Introduction

In an enterprise management there are a lot of different decisions. Often their long-term effects and methods to predict that are not obvious. Methods for process research and forecasting are used for informed decision-making:

- risk assessment;
- methods of statistical analysis;
- process modeling.

Modeling is one of the most promising methods with a number of advantages. Allows you to explore systems and processes without affecting them. Allows you to use a modular approach in research, i.e. isolate parts of systems and study them separately, or explore the interaction of several systems. The ability to obtain additional data about the object during the study of its model.

Mathematical modeling is one of the types of modeling, which consists in the representation of reality in mathematical expressions and their further research. Mathematical modeling lends itself well to programming and computing in a computer, which makes it possible to automate this process and create decision support systems (DSS).

DSS help responsible employees to analyze a given situation, calculate and analyze the consequences of a decision. It is important to understand that the DSS does not control the object and does not affect it in any way.

DSS uses a variety of technologies and methods, both individually and collectively:

– knowledge bases and search algorithms;
– intelligent data analysis;
– genetic algorithms;
– neural networks;
– simulation modeling.

There are two types of DSS. Passive offers the expert a set of information that helps to make a decision, but cannot put forward its own decision. An active one can,

on the basis of the data, put forward its own solution, some systems support the possibility of subsequent correction of the solution by an expert and checking the corrected solution until the expert approves the results.

One of the day-to-day decisions in an industrial enterprise is supply chain planning and capacity utilization (production schedule). There are methods for calculating the optimal production schedule, as management concepts: lean manufacturing, Six Sigma, etc. Also, as mathematical disciplines: operations research and optimization methods, queuing theory, etc.

Manual calculation of an effective production schedule leads to a deterioration in production efficiency, a slower increase in production volumes, with an increase in capacity, and a longer time for eliminating emergency situations. The solution to these problems can be the introduction of a DSS for supply chain planning. Most modern enterprises already use CAM, but they do not have planning functionality. And specialized solutions have a cost that is not affordable for many small and medium-sized enterprises, moreover, they require constant support or the hiring of employees with appropriate qualifications.

Purpose of work: development of a DSS system for supply chain planning. In the course of this work, the environment for simulation of production processes has been implemented. The development is intended for machine tool enterprises of various degrees of automation, has the following capabilities:

– building models with a different number of processing machines;

– three different modes of batch processing and the possibility of their expansion;

– development of the specified production load schedules and generation of output scenarios;

– development of plans on various models to simulate various conditions and emergency situations.

The development is done in the form of a modular, client-server application. An authorized user operates the browser, no installation is required on the client

computer. The application is coded in C# and C ++ in the Visual Studio Community 2019 development environment.

Project description

In this project, an environment for simulation of production processes is made. In order to increase portability, facilitate testing and detect errors and control program behavior, a modular approach has been adopted in development. The development consists of three modules:

1. library for modeling production processes;
2. module for storing and accessing data;
3. client-server module

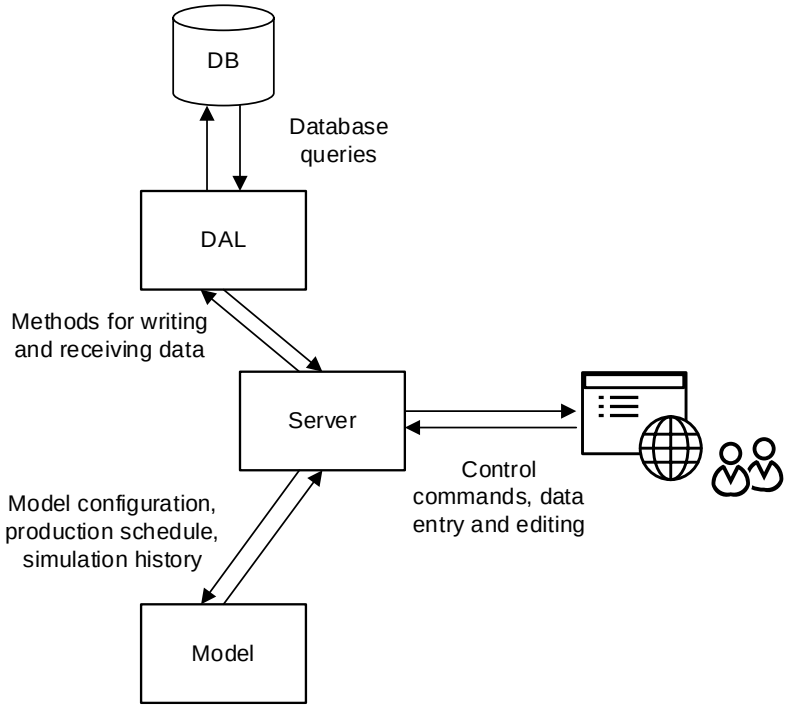


Figure 1 – Modular program structure

The first module performs the functions of modeling the processing of products, as input it receives the configuration of the model and the production plan, expressed in machine-readable code (XML file). As an output, the program gives the "history" of the simulation, also expressed in machine-readable code. This module is programmed in the form of a dynamically connected library, in its work it is customary to call it "model of production processes" or "model".

The second module provides connection between the application and the database, implements methods for storing data in the database and retrieving them from the database. This module is usually called the data access layer (DAL).

The third module is responsible for user interaction and data stream control. The module is made as a client-server application, the functionality of the two previous modules is encapsulated in this module, in this paper it is usually called a "server". The user works with the application from their web browser. After authorization, the user can request data on previous models and the server will call the necessary methods from the DAL or call methods for saving data when the user creates a new model or changes an existing one. After configuring the model, the user can load the manually developed, in previous developments or in similar products, the production plan and call the simulation of this schedule. The server requests data on the model in the DAL, transfers them to the model, and at the end of the simulation, the data is transferred to the user control screen and, at the user's request, is saved in the database by calling the method in the DAL.

Manufacturing process model

The model implements step-by-step production processes and is a black box. The model receives production schedule and batch data as input. At the exit, the model generates a history for each machine - which batches, at what time they went through it, and for each batch - which machines went through the processing and how long it took.

The entire production model can be divided into separate models for each production unit. The installation model also features a black box. As an input, the model receives data about the current batch for processing and a processing recipe. At the output, the installation model gives the batch processing time. Functions and rules for transforming input data into outputs are determined at the design stage. At the moment, the model implements three types of product processing.

Types of product processing

In-line processing – products are continuously conveyed one after the other. In the object of research. A typical example of this type of processing is the processing of products in turn on a CNC machine.

Group processing – several products are processed at the same time, but not the entire batch as a whole. A typical example is the chemical treatment of several products in baths with reagents.

Group of batches processing differs from batch processing in that the number of items processed is limited to a few batches. An example in production, processing of several batches of products in ovens.

Simulation algorithmization

The production process model is developed in the object-oriented programming (OOP) paradigm. The fundamental component of OOP is classes and objects.

object and its behavior; in the course of the application, objects are created that are defined by the class. Each class contains attributes (fields) and methods (or just attributes or methods).

Attribute is the data that is encapsulated in the class. Can be a variable of base type or some tuple of base types supported in PL.

Method is a function that performs some operations with data. The data can either belong to the class (its fields) or passed to the method from the outside. Also, as a result of the call, the method can return data.

Class fields and methods have access modifiers. The access modifier determines which objects can access (use) the fields and methods of this class:

– private – such fields and methods can only be accessed by the object to which they belong;

– public – such fields and methods can be accessed by an object of any class;

– protected – such fields and methods can only be accessed by an object whose class is in the inheritor's relation to the class of this object;

Simulation algorithm

The principle of the stream of events has been developed to implement the simulation. The program implements (in the form of classes) entities that make up the stream of events: Machine, Batch, Recipe, Event. The event reflects the fact of the end of batch processing in the machine, the event occurs with a delay - the time of batch processing in the machine. Accordingly, the stream of events is a sequence of events (outputs of parties and installations) that occur one after another.

The mechanics of the stream of events are based on three principles:

1. all machines in production (and in the model, respectively) work in parallel.
2. only one event occurs at a time in the installation. Or we can say that processing takes place according to only one recipe.
3. there is no delay between the end of processing in the current setting and the beginning of the next.

After entering the data and starting the simulation, the model is initialized. Each installation, in the input queue of which there are parties, generates one event (principle 2) and transfers it to the general stream. The stream is automatically distributed in ascending order of the duration of the event, the place of the event in the stream is determined upon insertion.

The modeling process takes place in steps. When the model receives the command (function call) "execute the modeling step", the first event is removed from the stream, since the stream is distributed in ascending order of duration, the duration of this event is the smallest. For all other events, the duration decreases by the duration of the withdrawn event (principle 1), and the global model time increases by the same amount. The machine in which the event occurred submits the new event to the thread for the next batch in the queue (principle 3). Let's take an example.

Figure 2 schematically shows the stream of events for three cases: initial conditions (initialization), invoking one simulation step, and invoking two simulation steps.

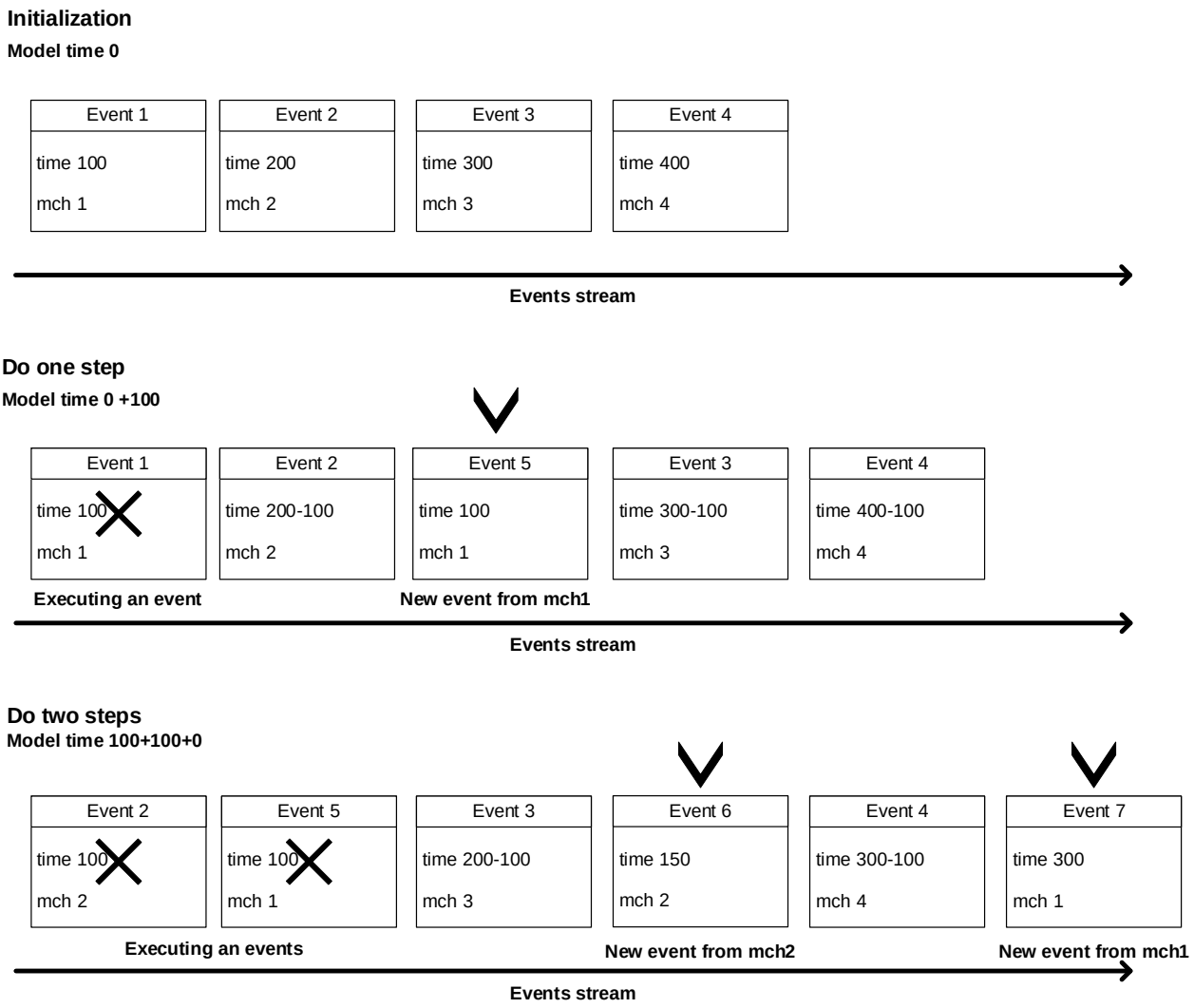


Figure 2 – How the event stream works

After calling "Execute one step", the Event1 event (marked with a cross) is removed from the stream. The global model time increases by the duration of Event1, i.e. by 100, and the duration of other events, on the contrary, decreases. Installing mch1 adds a new event to the stream (Event 5).

When you call do two steps, one simulation step is executed twice. At the first step, Event2 is executed, the time of all events is decreased by 100, and the global model time is increased. The mch2 machine adds Event6 to the stream.

The duration of Event5 before the first step was 100 units, correlated after it, it became equal to zero. Which is fully consistent with the principles of modeling and does not disrupt the operation of the system, such a case means that the current and previous events ended simultaneously. The time of other events and the global will not change. Installing mch1 will add the event to the stream.

Приложение Б

(дополнительное)

Листинг кода реализации функции push_ev класса Group_processing

```
unsigned int Group_Processing::push_ev()
{
    if (!_mptr._batches.empty())
    {
        Batch* firstBatchPtr = _mptr._batches.front();
        const Recipe& firstBatchRecipeRef = firstBatchPtr->get_first();
        if  (std::any_of(_mptr._recipes.begin(),  _mptr._recipes.end(),
[firstBatchRecipeRef](Recipe const r) {return firstBatchRecipeRef == r;
}))
        {
            return  ((_mptr._last_resipe == firstBatchPtr->get_first())
?
                firstBatchRecipeRef.get_time() :
                firstBatchRecipeRef.get_time() + _mptr._time);
        }
        else throw (firstBatchPtr);
    }
    else
    {
        Batch* ptr = nullptr;
        throw (ptr);
    }
}
```

Приложение В

(дополнительное)

Листинг кода реализации функции push_ev класса Stack_processing

```
unsigned int Stack_Processing::push_ev()
{
    if (!_mptr._batches.empty())
    {
        Batch* it = _mptr._batches.front();
        const Recipe& rcp = it->get_first();
        if  (std::any_of(_mptr._recipes.begin(),  _mptr._recipes.end(),
[rcp](Recipe const r) {return rcp == r; }))
        {
            return  ((_mptr._last_resipe == it->get_first()) ?
                rcp.get_time() * (int(it->get_count() / this->_count) +
1) :
                rcp.get_time() * (int(it->get_count() / this->_count) +
1) + _mptr._time);
        }
        else throw (it);
    }
    else
    {
        Batch* ptr = nullptr;
        throw (ptr);
    }
}
```

Приложение Г

(дополнительное)

Листинг кода реализации функции do_step класса Environement

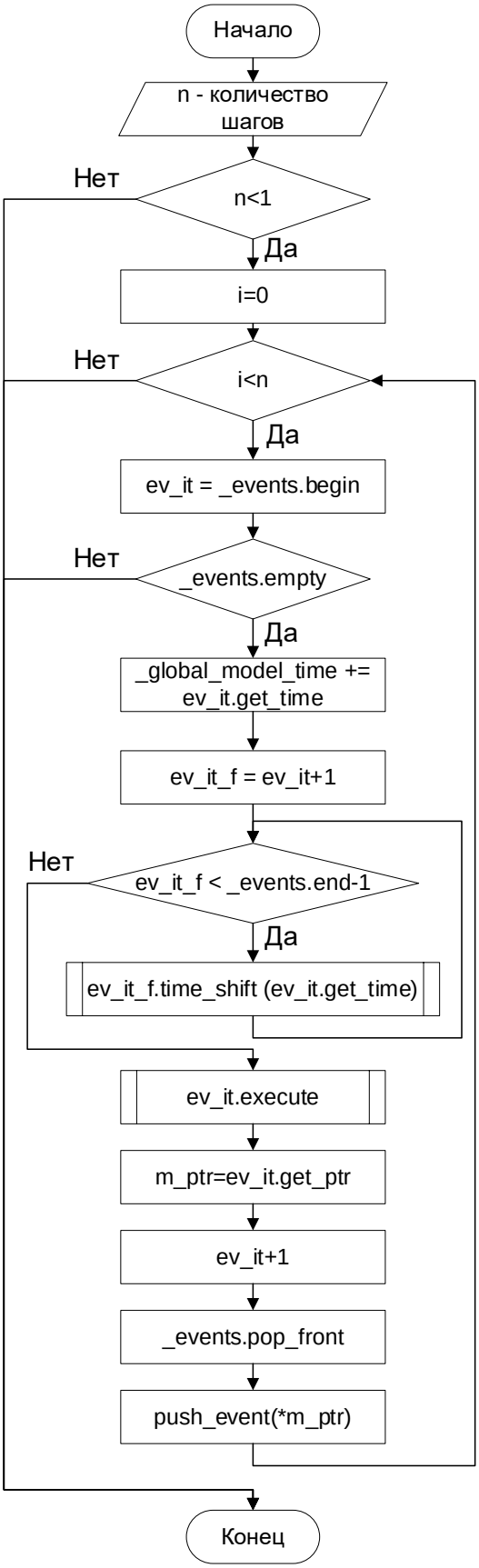
```
void Environment::do_step(unsigned int n)
{
    if (n < 1) return;
    try
    {
        if (this->_events.empty()) throw (-1);
        for (int i = 0; i < n; ++i)
        {
            auto ev_it = this->_events.begin();
            *this->_messages    <<    "MODEL_TIME:    "    <<    this-
>_global_model_time << '\n';
            if (this->_events.empty()) throw (i);
            this->_global_model_time += ev_it->get_time();
            auto ev_it_f = ev_it;
            ++ev_it_f;
            for (ev_it_f; ev_it_f != this->_events.end()--; ev_it_f++)
            {
                ev_it_f->time_shift(ev_it->get_time());
            }
            ev_it->execute(this->_messages);
            Machine * m_ptr=ev_it->get_ptr();
            ev_it++;
            this->_events.pop_front();
            this->push_event(*m_ptr);
        }
    }
    catch (int err)
    {
        switch (err)
        {
            case -1:
                *this->_messages << "Event vector is empty!\n";
                break;
            default:
                *this->_messages << "End of event vector reached by
step:\t"<<err<<'\n';
        }
        return;
    }

}
```

Приложение Д

(дополнительное)

Блок-схема функции do_step класса Environement



Приложение Е

(дополнительное)

Листинг кода моделей данных в DataModels

```
using System.Collections.Generic;
namespace DAL.DataModels
{
    public abstract class BaseData{
        public int Id { get; set; }}
    public abstract class NamedBaseData : BaseData{
        public string Name { get; set; }
    }
    public class User : NamedBaseData {
        public Company Company { get; set; }
    }
    public class Company : NamedBaseData {
        public List<User> Users { get; set; }
        public List<Model> Models { get; set; }
    }
    public class Model : NamedBaseData{
        public Company Company { get; set; }
        public List<Machine> Machines { get; set;}
        public List<Batch> Batches { get; set; }
        public List<Event> Events { get; set; }
    }
    public class Event : BaseData {
        public Model Model { get; set; }
        public Machine Machine { get; set; }
        public Batch Batch { get; set; }
    }
    public class Machine : BaseData{
        public int Type { get; set; }
        public int Time { get; set; }
        public bool State { get; set; }
        public Model Model { get; set; }
        public List<Recipe> Recipes { get; set; }
        public Recipe LastRecipe { get; set; }
    }
    public class Batch : BaseData{
        public int Count { get; set; }
        public Model Model { get; set; }
        public List<Recipe> Recipes { get; set; }
    }
    public class Recipe : BaseData{
        public int Time { get; set; }
    }
}
```

Приложение Ж

(дополнительное)

Листинг кода контекста БД в Context

```
using DAL.DataModels;
using System.Data.Entity;

namespace DAL
{
    class Context : DbContext
    {
        public DbSet<User> Users { get; set; }
        public DbSet<Company> Companies { get; set; }
        public DbSet<Model> Models { get; set; }
        public DbSet<Event> Events { get; set; }
        public DbSet<Machine> Machines { get; set; }
        public DbSet<Batch> Batches { get; set; }
        public DbSet<Recipe> Recipes { get; set; }

        public Context(string connection) : base(connection) { }

protected override void OnModelCreating(DbModelBuilder modelBuilder)
{
    modelBuilder.Entity<Company>().HasMany(c => c.Users)
            .WithRequired(u => u.Company);
    modelBuilder.Entity<Company>().HasMany(c => c.Models)
        .WithOptional();
    modelBuilder.Entity<Model>().HasMany(m => m.Machines)
        .WithRequired(mch => mch.Model);
    modelBuilder.Entity<Model>().HasMany(m => m.Batches)
        .WithRequired(b => b.Model);
    modelBuilder.Entity<Model>().HasMany(m => m.Events)
        .WithRequired(e => e.Model);
    modelBuilder.Entity<Machine>().HasMany(m => m.Recipes)
        .WithOptional();
    modelBuilder.Entity<Machine>().HasRequired(m => m.LastRecipe)
        .WithOptional();
    modelBuilder.Entity<Batch>().HasMany(b => b.Recipes)
        .WithOptional();
    modelBuilder.Entity<Event>().HasRequired(e => e.Machine)
        .WithOptional();
    modelBuilder.Entity<Event>().HasRequired(e => e.Batch)
        .WithOptional();
}
    }
}
```