**АННОТАЦИЯ**

**СОДЕРЖАНИЕ**

[ВВЕДЕНИЕ 4](#_326xx943hs56)

[1 АНАЛИЗ СИСТЕМ ОПТИМИЗАЦИЙ РАСПИСАНИЙ ДЛЯ ВОССТАНОВЛЕНИЯ ТРАНСПОРТНЫХ УЗЛОВ И АГРЕГАТОВ НА СПЕЦИАЛИЗИРОВАННОМ ПРЕДПРИЯТИИ 6](#_z6ei9530smt9)

[1.1 Анализ предметной области и основных ее характеристик 6](#_2lgbc8h2vmn8)

[1.2. Анализ математических моделей 10](#_20poqx5skwfc)

[Выводы по разделу 1 10](#_7s2hyxdvhp6h)

[2 СИСТЕМОТЕХНИЧЕСКИЙ АНАЛИЗ И ПРОЕКТИРОВАНИЕ ИНФОРМАЦИОННОЙ СИСТЕМЫ ХРАНЕНИЯ И ВВОДА ДАННЫХ ОПТИМИЗИРОВАННОГО РАСПИСАНИЯ 11](#_gp6w4ejzd6n7)

[2.1 Построение диаграммы потоков данных для описания процессов взаимодействия пользователя с системой 11](#_fo75q065kk5a)

[3 РАЗРАБОТКА СИСТЕМЫ ВВОДА И ХРАНЕНИЯ ДАННЫХ 12](#_pycx5oxbnf0n)

[3.1 Обоснование выбора средств программной реализации системы ввода и хранения данных 12](#_bxh6ssecxr01)

[3.2 Обоснование структурно-функциональной схемы системы ввода и хранения данных для оптимизации расписания 13](#_3ejw7tzfe372)

[3.3. Обоснование алгоритма программы системы хранения и ввода для оптимизации расписания 13](#_bzk1rjjf91zf)

[3.4 Описание разработанной программы и результаты работы 13](#_l28iz23qp3fk)

[ЗАКЛЮЧЕНИЕ 14](#_433rb5jcnf0z)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 15](#_a02zj8k1bxk3)

# **ВВЕДЕНИЕ**

**Актуальность темы.** В современном мире компании по производству транспортного оборудования используют различные узлы и агрегаты, которые имеют свойство изнашиваться и терять работоспособность. Отсюда возникает необходимость проведения ремонтных работ или замены. При ремонте часто используется специализированное дорогостоящее оборудование, и для оптимизации ресурсов у предприятия есть необходимость в ведении учета о поврежденных деталях.

На начальном этапе ремонта выполняется разборка узлов и агрегатов на различные отдельные детали, для подготовке к процессу дефектации. При дефектации определяется тип и возможность восстановления детали, а также определяются необходимые работы по восстановлению формы или размера детали. В зависимости от типов деталей определяются способы устранения дефектов.

Соответственно, алгоритм системы хранения и ввода данных может быть актуален для записи и хранении данных о деталях, которые подлежат восстановлению, и может быть эффективен в последующих расчетах времени восстановления деталей для составления расписания времени выполнения ремонтных работ. При учете времени в начале необходимо упорядочить технологические процессы восстановления деталей, а после выполнять необходимые расчеты. Таким образом, при наличии данных о времени восстановлении система оптимизации расписаний восстановления узлов и агрегатов технологического и транспортного оборудования на специализированных ремонтных предприятиях будет полезна в обеспечении эффективной работы предприятий. Она предназначена для планирования, оптимизации и эффективном управлении процесса восстановления различных узлов и агрегатов техники и оборудования.

**Цель работы.** Целью текущей выпускной квалификационной работы является разработка алгоритма системы хранения и ввода данных о деталях, которые в последствии будут формироваться в партии в составлении и оптимизации расписания восстановления.

**Объект и предмет исследования.** Объектом исследования в данной квалификационной работе является процесс восстановления детали на предприятии.

**Предметом исследования** является алгоритм хранения и ввода данных для оптимизации расписаний восстановления узлов и агрегатов.

Структура работы.

# **1 АНАЛИЗ СИСТЕМ ОПТИМИЗАЦИЙ РАСПИСАНИЙ ДЛЯ ВОССТАНОВЛЕНИЯ ТРАНСПОРТНЫХ УЗЛОВ И АГРЕГАТОВ НА СПЕЦИАЛИЗИРОВАННОМ ПРЕДПРИЯТИИ**

## 1.1 Анализ предметной области и основных ее характеристик

На специализированных предприятиях регулярно возникает необходимость минимизировать время получения результатов и максимизировать эффективность использования оборудования для получения хорошего качества. Создание математической модели для оптимизации расписания позволяет специализированным ремонтным предприятиям повысить производительность, сократить временные и ресурсные затраты, а также обеспечить высокое качество ремонтных работ, что в конечном итоге приводит к более эффективному и рентабельному производству.

На данный момент, транспортное оборудование регулярно требует обслуживания и ремонта, так как детали имеют срок эксплуатации и подвергаются износу. Современные ремонтные специализированные предприятия производят ремонтно-восстановительные работы узлов и агрегатов транспортного оборудования. Для проведения ремонта, в начале производится разбор деталей транспортного оборудования на составляющие компоненты.

Ввести понятие технологического процесса.

Порядок восстановления детали начинается с определения вида ремонтных работ. После разбора узлов, требующих ремонта, проводится начальный этап восстановительных работ - дефектация. Данный этап помогает определить возможность восстановления детали путем оценки остаточного ресурса самой детали, который можно использовать при восстановлении. Затраты на восстановление составляют 10 — 50 % от стоимости детали [Воловик]. Таким образом, при дефектации определяется, какие детали подлежат замене, какие детали можно восстановить, и какие детали могут быть использованы дальше, без необходимости проведения восстановительных работ. Для последующих работ детали, которые можно восстановить, поступают на склады, а после на цеха восстановительного ремонта.

Износ цилиндрической поверхности составляет 52% [Воловик]. По этой причине, на специализированных ремонтных предприятиях выделяется особое внимание к восстановлению деталей цилиндрической формы.

К деталям цилиндрической формы больших размеров относятся различные валы. К деталям цилиндрической формы малых размеров относятся пальцы, валики, оси. Детали этих типов не восстанавливаются и подлежат замене [Диссертация].

Методы устранения дефектов входят в состав технологических процессов предприятия и зависят от видов дефектов деталей. Процесс восстановления детали цилиндрической формы включает в себя:

— удаление следов износа.. Поверхность детали для начала подвергается механической обработке, так как износ на поверхности чаще всего неравномерен, и важно обеспечить равномерное покрытие при восстановлении формы и размеров деталей;

– восстановительные операции (наплавка, сварка, металлизация и др.);

– после восстановления детали проходят токарную механическую обработку;

– заключительным этапом является шлифование.

Проведение ремонтно-восстановительных работ на любого рода предприятиях требует наличия специальной техники, которая могла бы сделать процесс восстановления некоторых параметров детали (например, размер, форма) автоматическим. Наиболее часто применимы такие технологические операции, как:

– автоматическая электродуговая наплавка под слоем флюса;

– вибродуговая наплавка;

– металлизация (напыление расплавленного металла на подготовленную поверхность деталей).

Операция наплавки применяется в 50% случаев восстановления деталей [Воловик]. Наиболее распространенными способами восстановления деталей цилиндрической формы являются автоматическая электродуговая наплавка под слоем флюса и вибродуговая наплавка.

Ремонт деталей цилиндрической формы осуществляется путем объединения их в партии [Тельнов]. Партии - это группы или наборы деталей, которые объединяются для выполнения определённых технологических операций, таких как ремонт, обработка или сборка. Объединение деталей в партии позволяет оптимизировать производственный процесс, улучшить управление качеством и повысить эффективность выполнения задач.

При включении деталей в партии, учитываются такие аспекты:

– Единый технологический способ устранения дефектов (различные дефекты устраняются одинаковым методом);

– Единые дефекты восстанавливаемых деталей (одинаковые дефекты цилиндрических деталей устраняются с применением одного технологического процесса - маршрутной технологии);

Для группирования деталей в партии используется маршрутная технология восстановления деталей с однотипными дефектами. Восстанавливаемые детали относятся к одному типу, если они имеют одинаковую форму и размеры, а также одинаковые виды дефектов, которые могут быть устранены одним технологическим процессом (время устранения дефектов этих деталей одинаковое). Соответственно, по маршрутной технологии осуществляется восстановление деталей разных типов, но с одинаковыми видами дефектов [диссертация].

Количество типов восстанавливаемых деталей и число деталей каждого типа является достаточно большим. Это затрудняет использование методов частично целочисленного линейного программирования для планирования восстановления партий деталей разных типов, а также для оптимизации состава этих партий и расписания их восстановления. При высокой размерности задачи, её решение невозможно получить с помощью известных методов. Поэтому возникает необходимость применения математических моделей выполнения пакетов заданий в многостадийных системах, а также численных методов оптимизации состава партий и расписания операций для планирования восстановления деталей разных типов. [диссертация]

Таким образом, при большом количестве типов деталей и деталей каждого типа задача планирования их восстановления в поточных системах специализированных ремонтных предприятий остается нерешенной. Применение математических моделей и численных методов оптимизации позволяет сократить общее время восстановления деталей разных типов и повысить эффективность работы оборудования за счет уменьшения его простоев.

## 1.2. Анализ существующих систем оптимизации расписания

**ERP-СИСТЕМЫ**

## Выводы по разделу 1

# **2 СИСТЕМОТЕХНИЧЕСКИЙ АНАЛИЗ И ПРОЕКТИРОВАНИЕ ИНФОРМАЦИОННОЙ СИСТЕМЫ ХРАНЕНИЯ И ВВОДА ДАННЫХ ОПТИМИЗИРОВАННОГО РАСПИСАНИЯ**

# **2.1 Построение диаграммы потоков данных для описания процессов взаимодействия пользователя с системой**

Подсистема ввода и хранения данных является частью общей системы оптимизации расписания процессов восстановления узлов и агрегатов технологического и транспортного оборудования на специализированных ремонтных предприятиях.

Работа с подсистемой осуществляется непосредственно пользователем, путем записи данных о деталях, а также составлением наборов деталей для передачи данных о технологических процессах подсистеме рещения MILP, оптимизирующей расписание.

Подсистема решения MILP в свою очередь также взаимодействует с данной подсистемой, предоставляя оптимизированное расписание восстановления транспортных узлов и агрегатов, которое в дальнейшем будет отформатировано подсистемой ввода и хранения данных и отображено пользователю.

Таким образом, можно выделить две внешние сущности, взаимодействующие с подсистемой — «Пользователь» и «Подсистема MILP».

Рассмотрим потоки данных. Важно выделить, что входными данными от пользователя будут информация о деталях: тип детали, изначальный диаметр, износ и размеры повреждений. Далее для дальнейшей работы с подсистемой от пользователя поступает запрос на составление расписания восстановления набора деталей. В данном запросе пользователем формируется необходимый набор деталей, и далее система рассчитывает необходимые данные для его составления.

Выходными потоками данных из подсистемы являются результаты: данная подсистема передает подсистеме решения MILP результаты проведенных ею вычислений времени выполнения восстановления детали для набора, определенного пользователем. После того, как подсистема смешанного целочисленного линейного программирования предоставит результаты оптимизированного расписания, подсистема формирует и отображает результаты пользователю.

На рисунке 2.1 продемонстрирована описанная выше диаграмма потоков данных.



Рисунок 2.1 - Основная DFD-диаграмма

Процесс «система ввода и хранения данных» можно декомпозировать. Данная система непосредственно работает с чтением и записью в базе данных (БД). Опишем процессы обмена информации с базой данных в подсистеме.

Анализ входных данных подразумевает ввод данных пользователем и запрос на выполнение записи в БД. При поступлении от пользователя запроса на составление оптимизированного расписания, выполняется процесс формирования набора деталей для расписания. Соответственно, для составления набора выполняется выборка из существующих данных в БД и запрос в базу данных на чтение. Далее, из базы данных поступает ответ о свойствах деталей в составляемом наборе, и выполняется расчет общего и вспомогательного времени выполнения восстановления узлов и агрегатов транспортного оборудования.

От подсистемы решения MILP посредством файла поступают данные об оптимизированном расписании восстановления, которое подвергается интерпретации для дальнейшей записи в БД и последующем форматировании и отображении данных пользователю. Рисунок 2.2 демонстрирует декомпозированный процесс «система ввода и хранения данных».

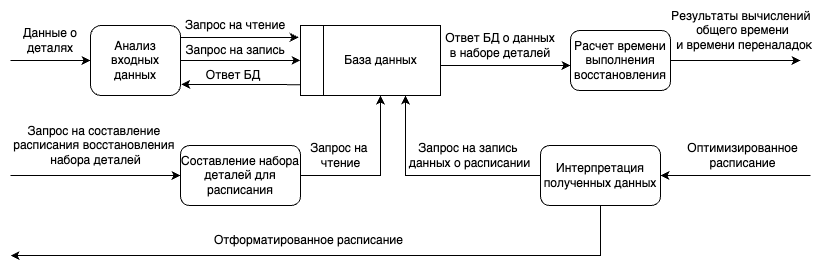


Рисунок 2.2 - Декомпозиция процесса «Система ввода и хранения данных»

## 2.2 Разработка функциональных и информационных моделей IDEF0-IDEF1 системы ввода и хранения данных

Для построения функциональной модели используется IDEF0. С помощью нее можно продемонстрировать бизнес-процессы разрабатываемой системы.

Рисунок 2.3 демонстрирует основную функцию - «Система оптимизации расписания восстановления узлов и агрегатов транспортного оборудования на специализированных предприятиях».

На вход данной системы поступают запросы: на создание оптимизированного расписания, ввод и хранения данных о деталях, запрос на интерпретацию и отображение оптимизированного расписания. Инструментами выполнения запроса являются различные формы, формулы и шаблоны: форма ввода данных и создания заявки, формулы расчетов данных времени выполнения, шаблон интерпретации оптимизированного расписания. Взаимодействовать с данным процессом могут как система так и пользователь. В итоге, на выходе результатом будут данные о времени восстановления узлов и агрегатов и содержании ПЗ, а также интерпретированное оптимизированное расписания выполнения ремонтных работ транспортных узлов и агрегатов.



Рисунок 2.3 - Диаграмма основного процесса системы в нотации IDEF0

Основной процесс можно декомпозировать. На рисунке 2.4 продемонстрирована декомпозиция основного процесса А0 разрабатываемой системы

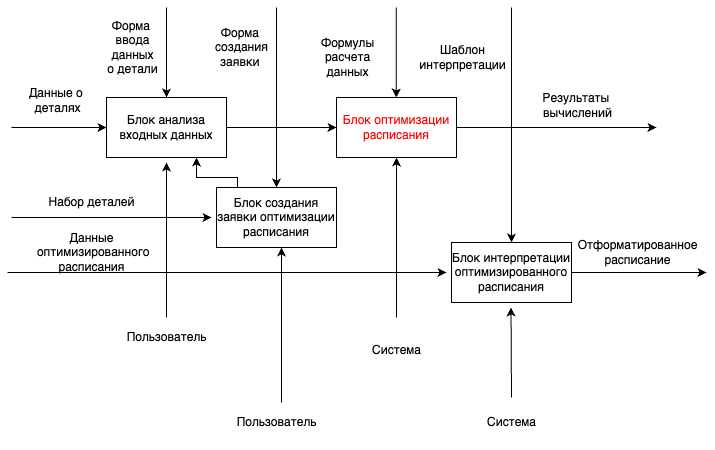


Рисунок 2.4 - Декомпозиции основного бизнес-процесса А0 в нотации IDEF0

В данной выпускной работе будут рассматриваться только три блока - блок анализа ввода данных, блок создания заявки оптимизации расписания и блок интерпретации оптимизированного расписания. Блок оптимизации расписания относится к подсистеме решения задач целочисленного программирования MILP.

Декомпозиция включает в себя:

1. Функциональный блок «А1» - Блок анализа входных данных. Данный блок подразумевает ввод пользователям входных данных о детали: тип детали, износ, изначальный диаметр и размер участков повреждения детали. Декомпозиция данного процесса представлена на рисунке 2.5.

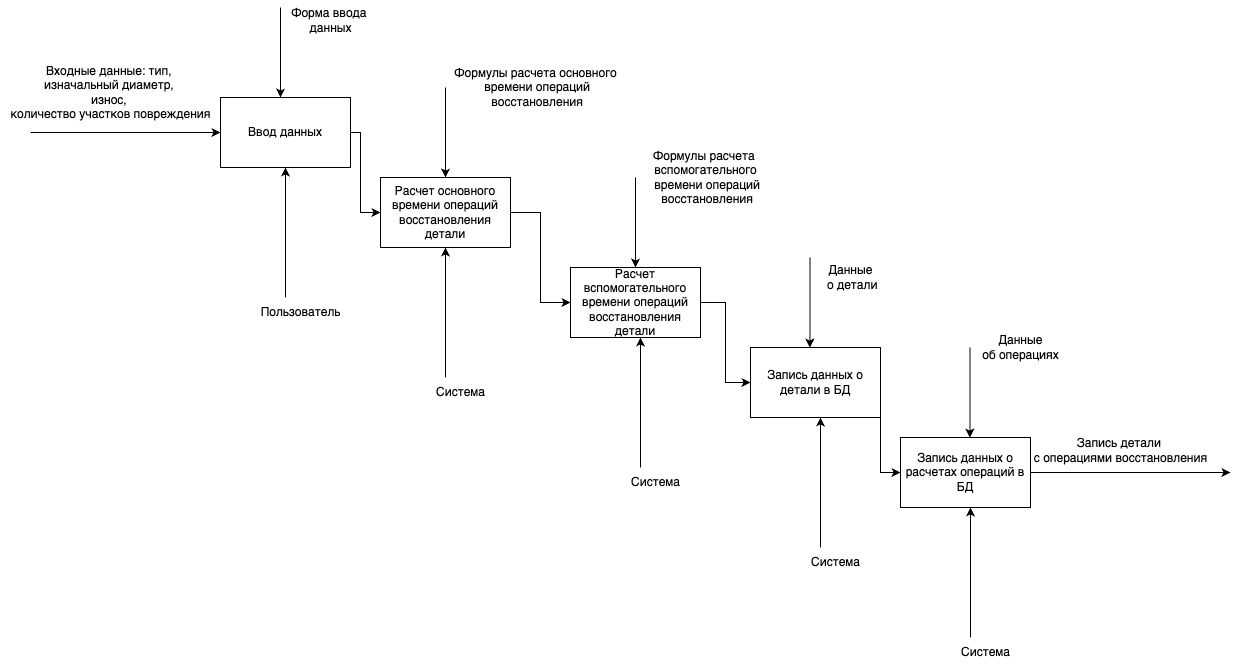


Рисунок 2.5 - Декомпозиция блока анализа входных данных

1. Функциональный блок «А2» - Блок создания заявки системе оптимизации расписания набора деталей. В этом блоке подозревается, что пользователь формирует набор деталей, подлежащих восстановлению. Декомпозиция данного процесса представлена на рисунке 2.6.

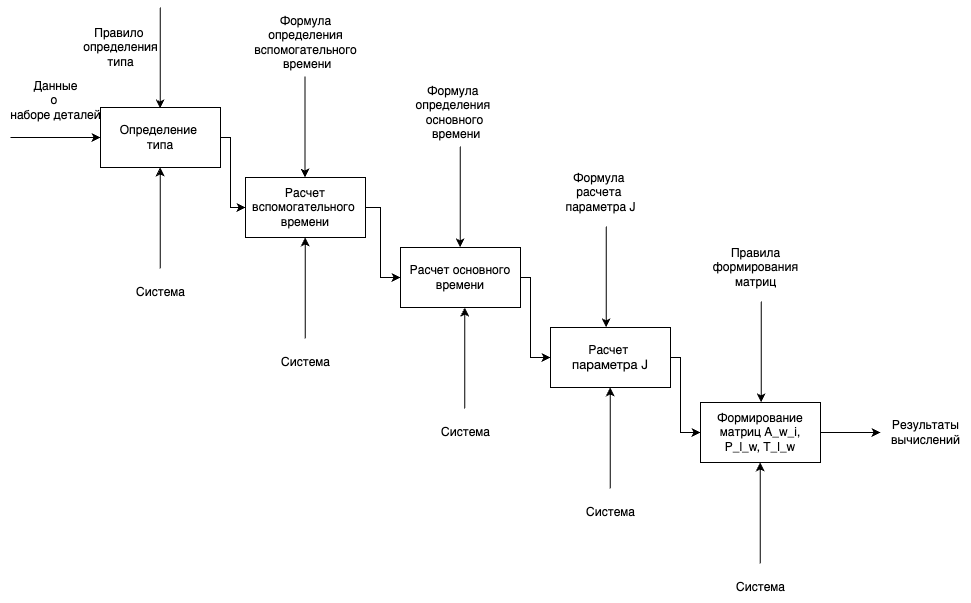


Рисунок 2.6 - Декомпозиция блока создания заявки

1. Функциональный блок «А3» - «Блок интерпретации оптимизированного расписания». Входными данными для данного блока является оптимизированное системой расписание в виде файла. Декомпозиция данного процесса представлена на рисунке 2.7.

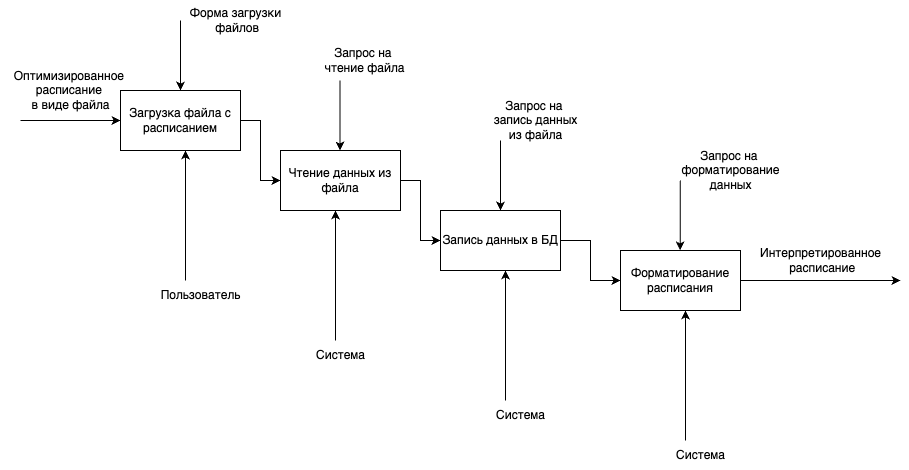


Рисунок 2.7 - Декомпозиция блока интерпретирования оптимизированного расписания.

В итоге была построена и декомпозирована диаграмма IDEF0, которая в дальнейшем будет использоваться при проектировании подсистемы.

Для определения ключевых сущностей системы, для дальнейшего проектирования. Таблица 2.1 содержит перечень потенциальных сущностей системы.

Таблица 2.1 - Потенциальные сущности системы

| № | Название сущности | Описание |
| --- | --- | --- |
| 1 | Деталь | Содержит основные данные о детали в системе |
| 2 | Заявка | Содержит основную информацию о наборе деталей, требуемых к восстановлению |
| 3 | Операция | Содержит основную информацию о расчетах времени восстановления на различных этапах |
| 4 | Технологический процесс | Содержит информацию о времени процесса восстановления для одной детали на различных этапах |
| 5 | Расписание | Содержит данные оптимизированного расписания, получаемого от подсистемы решения MILP |

Список, представленный в таблице 2.1 помог выявить основные сущности и их атрибуты. Таблица 2.2 демонстрирует разделение сущностей и их атрибутов.

Таблица 2.2 - Атрибуты сущностей

| № | Название сущности | Описание |
| --- | --- | --- |
| 1 | Деталь | id\_детали, тип, название, износ, участки\_износа, начальный\_диаметр |
| 2 | Заявка | id\_заявки, id\_деталей, количество\_деталей, количество\_приборов, матрица\_основного\_времени\_восстановления, матрица\_переналадок, матрица\_соответствия\_задания\_типу\_задания, J\_parameter |
| 3 | Операция | id\_детали, id\_операции, тип\_операции, основное время, вспомогательное\_время |
| 4 | Технологический процесс | id\_процесса, id\_деталей, массив\_операций |
| 5 | Расписание | id\_расписания, данные\_расписания |

Также было составлено описание предметной области на естественном языке

1. Каждая заявка (сущность 2) <может> <иметь> <много> деталей (сущность 1).
2. Каждая деталь (сущность 1) <может> <содержать> <много> операций (сущность 3).
3. Каждый технологический процесс (сущность 4) может <может> <содержать> <много> операций (сущность 3).
4. Каждое расписание (сущность 5) может состоять из множества технологических процессов (сущность 6).
5. Каждый технологический процесс (сущность 4) может <может> <содержать> <одну> деталь (сущность 1).

Далее были определены имена и связи сущностей. Также заданы мощности связей между сущностями. Результаты представлены в таблице 2.3.

Таблица 2.3 - Матрица отношений между сущностями

|  | Деталь | Заказ | Операция | Тех. процесс | Расписание |
| --- | --- | --- | --- | --- | --- |
| Деталь |  | M:N | 1:N | 1:1 | M:1 |
| Заказ | N:М |  | N:M | 1:M | 1:1 |
| Операция | N:1 | M:N |  | M:1 | M:1 |
| Тех. процесс | 1:1 | M:1 | 1:N |  | M:1 |
| Расписание | M:1 | 1:1 | M:1 | M:1 |  |

Таблица 2.3 демонстрирует основные сущности и их ключевые атрибуты.

Таблица 2.3 - Ключевые атрибуты сущностей

| № | Название сущности | Описание |
| --- | --- | --- |
| 1 | Деталь | id\_детали, тип, название, износ, участки\_износа, начальный\_диаметр |
| 2 | Заявка | id\_заявки, id\_деталей, количество\_деталей, количество\_приборов, матрица\_основного\_времени\_восстановления, матрица\_переналадок, матрица\_соответствия\_задания\_типу\_задания, J\_parameter |
| 3 | Операция | id\_детали, id\_операции, тип\_операции, основное время, вспомогательное\_время |
| 4 | Технологический процесс | id\_процесса, id\_деталей, массив\_операций |
| 5 | Расписание | id\_расписания, данные\_расписания |

Исходя из всех данных выше требуется создать информационную модель уровня «сущность-связь» – ER-диаграмму в нотации П.Чена. Рисунок 2.8 демонстрирует диаграмму.

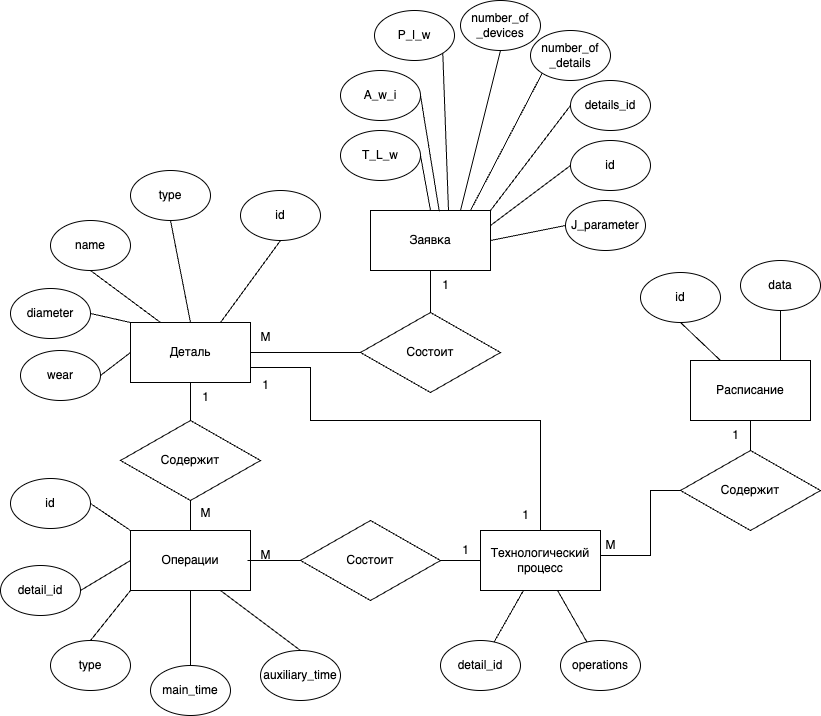


Рисунок 2.8 - ER-диаграмма в нотации П.Чена

Исходя из диаграммы опишем, какие связи имеют описанные сущности.

Сущность “Заявка” состоит из деталей и имеет связь многие-ко-многим с сущностью “Детали”. Деталь же содержит операции восстановления, соответственно сущность “Деталь” имеет связь один-ко-многим с сущностью “Операции”. Технологический процесс содержит множество операций соответствующей детали, соответственно сущность “Технологический процесс” имеет связь один-ко-многим с сущностью “Операции” и связь один-к-одному. Далее рассмотрим сущность “Расписание”. Данная сущность может хранить несколько технологических процессов, следовательно, она имеет связь один-ко-многим с сущностью “Технологический процесс”.

Далее, необходимо построить логическую модель данных, основанную на ключах. Исходя из составленных таблиц выше, можем проанализировать, что первичным ключом для каждой таблицы был выбран id - уникальный идентификатор записи в таблице.

Также из таблиц видно, что в системе есть связь многие-ко-многим между сущностями “Деталь” и “Заявка”. Для разбиения данной связи была введена таблица деталь-заявка, в которой в качестве атрибутов будут id заявки и id детали.

В итоге, каждая таблица имеет связь.

**2.3 Разработка обобщенной архитектуры и алгоритма функционирования системы**

## Выводы по разделу 2

# **3 РАЗРАБОТКА СИСТЕМЫ ВВОДА И ХРАНЕНИЯ ДАННЫХ**

## 3.1 Обоснование выбора средств программной реализации системы ввода и хранения данных

Алгоритм ввода и хранения данных будет реализован с помощью фреймворка для создания веб-приложений Laravel, в котором в качестве языка программирования используется PHP, а также для реализации интерфейса будет использован встроенный шаблонизатор Blade.

PHP (Hypertext Preprocessor) — современный, кроссплатформенный, интерпретируемый скриптовый язык программирования с открытым исходным кодом. Данный язык программирования имеет большое количество библиотек, активно поддерживается сообществом и используется для разработки статических и динамических сайтов и веб-приложений. [ссылка]

Laravel — фреймворк для веб-разработки, созданный на языке программирования PHP. Laravel предназначен для упрощения задач, связанных с созданием веб-приложений, и предлагает множество встроенных инструментов и функций. Фреймворк также имеет активную поддержку, и имеет удобную работу с объектно-реляционным отображением (ORM) Eloquent, что позволяет работать с базами данных с помощью моделей PHP, и упрощает выполнение CRUD операций и взаимодействие с реляционными данными. Также фреймворк содержит встроенный шаблонизатор Laravel, который предлагает легковесный, но мощный способ создания динамических HTML-шаблонов с использованием логики на PHP. [ссылка]

Для выполнения операций с базой данных (БД) в работе был выбран PostgreSQL. Это мощная, объектно-реляционная система управления базами данных (СУБД) с открытым исходным кодом, которая соответствует стандартам SQL и поддерживает большое количество функций — расширяемость, сложные запросы и транзакции. Также имеет активное и большое сообщество поддержки с большим количеством документации. [ссылка]

В качестве инструмента для реализации интерфейса был выбран встроенный в Laravel шаблонизатор Blade. Это простой, но мощный движок шаблонов, входящий в состав Laravel. Все шаблоны Blade компилируются в обычный PHP-код и кэшируются до тех пор, пока не будут изменены, что добавляет фактически нулевую нагрузку приложения. Данный шаблонизатор поддерживает язык разметки HTML, написание сценариев на Javascript а также поддержка стилей с помощью CSS [ссылка]. Для улучшения UI использовалась библиотека Bootstrap.

HTML — гипертекстовый язык разметки. Данный язык программирования используется для создания различных гипертекстовых документов, которые можно просматривать с помощью любого типа веб-браузеров. [ссылка]

CSS — каскадная таблица стилей, которая позволяет управлять размером и стилем шрифтов, таблиц, позиционированием элементов и многими другими параметрами отображения HTML – документов [ссылка].

JavaScript — это высокоуровневый, мультипарадигменный, функциональный часто компилируемый в режиме реального времени язык, соответствующий стандарту ECMAScript. Используется для создания различных скриптов и сценариев, которые встраиваются в HTML-документы или включаются в них и взаимодействуют с DOM.

Bootstrap — открытый HTML и CSS фреймворк, который используется для ускорения верстки адаптивного дизайна. Включает в себя CSS и HTML-шаблоны оформления для веб-форм, меток, типографики, кнопок, блоков навигации и других компонентов веб-интерфейса. [ссылка]

## 3.2 Обоснование структурно-функциональной схемы системы ввода и хранения данных для оптимизации расписания

## 3.3. Обоснование алгоритма программы системы хранения и ввода для оптимизации расписания

## 3.4 Описание разработанной программы и результаты работы

# **ЗАКЛЮЧЕНИЕ**

# **СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ**