Введение

В процессе автоматизации производства, при расчете расписания работы сотрудников либо прогнозировании сроков выполнения заказов, существует необходимость привязки абстрактных расчетов, базирующихся на времени выполнения операций, к конкретно заданному календарю производственной площадки, который учитывает выходные дни, праздничные и шаблонные смены занятых сотрудников. Данный шаг позволяет оценить применимость составленного расписания в данных условиях и оценить реальные сроки выполнения заказа.

почему Go

Содержание

rename

1 Обзорная часть

1.1 Индустрия 4.0

Индустрия 4.0 - прогнозируемое массовое внедрение киберфизических систем в промышленность и повседневный быт человека. Под киберфизической системой (CPS - cyber-physical system) подразумевается взаимодействие цифровых, аналоговых, физических и человеческих компонентов, разработанных для функционирования посредством интегрированной физики и логики или другими словами: киберфизические системы - это интеллектуальные системы, которые состоят из тесно взаимосвязанных сетей физических и вычислительных компонентов. [3]

Появилось понятие индустрии 4.0 во время ганноверской выставки 2011 года как обозначение стратегического плана развития и поддержания конкурентоспособности немецкой экономики, предусматривающий совершение прорыва в области информационных технологий для промышленности. Также считается, что данное направление знаменует собой четвертую промышленную революцию. [2]

Если рассматривать производство, на что и нацелена индустрия 4.0, то, используя определение данное выше, процесс внедрения киберфизической системы разделяется на внедрение физической части (например датчики, собирающие и передающие данные посредством различных сетей) и вычислительную - систему планирования и прогнозирования.

Система планирования и прогнозирования (СПП) обеспечивает расчет объемно-календарного и оперативного планов, автоматизированный подбор поставщиков, автоматизированный перерасчет планов по фактическим результатам деятельности, направленный на минимизацию временных и финансовых потерь.

Объемно-календарный план - задание для каждой производственной площадки на заданный интервал времени, представляющее собой план-график, на котором каждому интервалу соответствует номенклатура и объем подлежащих к производству изделий. [1]

Оперативный план - план, согласно которому выполняется привязка каждой операции для каждой единицы продукции к временным интервалам, конкретному работнику и конкретным производственным средствам. [1]

Если говорить простым языком, то оперативный план является более подробной версией объемно-календарного плана, которая, в силу ограничений со стороны вычислительной сложности расчета, составляется на срок в порядок меньший, чем для объемно-календарного плана.

схема и сравнение планов

что расчитывается первее

1.2 Обзор существующих решений

Пара слов

1.2.1 Свободно распространяемые решения

FrePPLe, Odoo, qcadoo?По презентации

1.2.1.1 FrePPLe

FrePPLe - программное обеспечение для планирования цепочек поставок. "FrePPLe - простой в использовании и расширении (функционала) инструмент с открытым исходным кодом для планирования в производственных компаниях". Данный инструмент нацелен в первую очередь на планирование поставок, но также позволяет производить оперативное планирование и прогнозирование спроса.

1.2.2 Коммерческие решения

Нужно найти

1.3 Постановка задачи

Целью данной работы является разработка и тестирование компонент для системы планирования и прогнозирования таких как:

— модуль ресурса сборочной линии;

— модуль отображения логического времени на физическое;

— организация консистентного хранилища данных.

2 Система планирования и прогнозирования

2.1 Архитектура системы планирования и прогнозирования

Система планирования и прогнозирования имеет модульную структуру. Между собой модули взаимодействуют посредством внутренних структур данных

схема и описание

2.2 Подсистема имитационного моделирования

Подсистема имитационного моделирования - модуль, производящий моделирование производственных процессов для для получения приблизительной оценки времени выполнения набора операций (например карты технологического процесса).

Карта технологического процесса - документ, предназначенный для операционного описания технологического процесса изготовления или ремонта изделия (составных частей изделия) в технологической последовательности по всем операциям одного вида формообразования, обработки, сборки или ремонта с указанием переходов, технологических режимов и данных о средствах технологического оснащения, материальных и трудовых затратах.

Для определения оценки времени модуль рассматривает карту технологического процесса как систему уравнений, в которой неизвестными являются времена начала выполнения операций. Каждое

система уравнений

ние предстает в виде $x_n = x_{n-1} + dur$, а система в целом:

$$x_n = x_{n-1} + dur \tag{2.1}$$

В начале работы система выберет одно из уравнений (в зависимости от накладываемых моделью ресурса, о которой пойдет речь в следующем разделе, ограничений), которое может быть рассчитано, то есть которое может начинаться с логического нуля и произведет

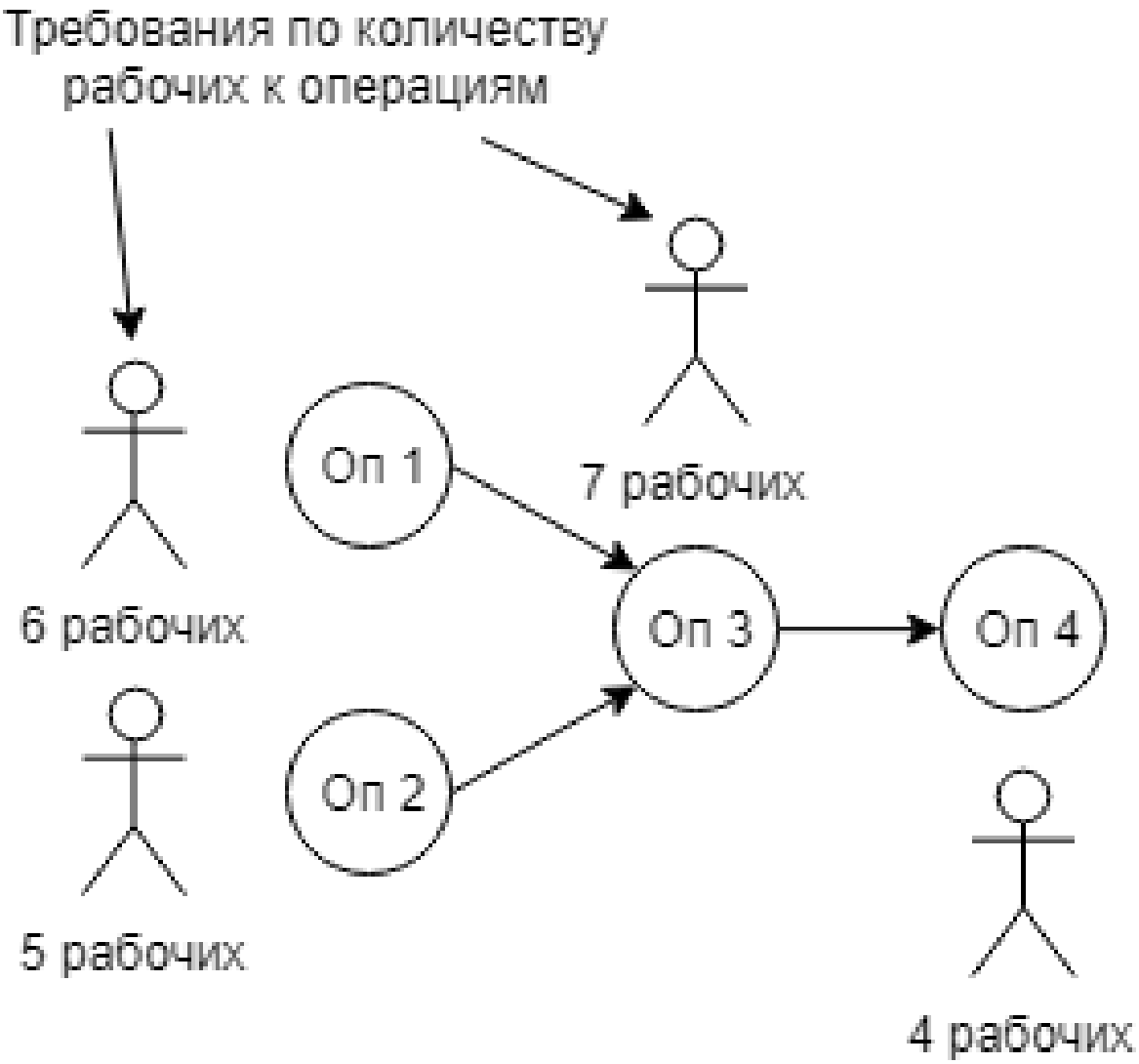

Рисунок 2.1 — Визуальное представление технологической карты с ограничением по количеству рабочих на операцию

подсчет времени, когда закончится выполнение данной операции. Затем по тому же принципу будет выбрано следующее уравнение и так далее пока есть неразрешенные переменные. Когда они закончатся система завершит свое выполнение, передав результирующее значение и необходимые данные другому модулю, который произведет отображение полученного подсистемой имитационного моделирования числа в физическое, о чем речь пойдет ниже.

система уравнений

схема и описание

2.3 Модель ресурса сборочной линии

В процессе создания оперативного плана, для получения корректной оценки времени выполнения операции или набора операций СПП необходимо ввести систему ограничений, которая будет отражать как ресурс, участвующий в операции может влиять на её время выполнения. Это привело к созданию модели ресурсов накладывающей ограничения на выбор операции для расчета ядром имитационного моделирования. Под ресурсом подразумевается любое устройство,деталь,инструмент или средство,за исключением сырьевого материала и промежуточного продукта, находящиеся в распоряжении предприятия для производства товаров и услуг. В соответствии с данным определением к ресурсам относятся в том числе и человеческие ресурсы, которые в данной системе не рассматриваются с точки зрения поведения или других аспектов человеческой жизни, а лишь с точки зрения возможности выполнить конкретную задачу. Также необходимо обозначить, что в данном разделе под моделью ресурса будет пониматься упрощенная модель реального ресурса, отражающая его основные (в рамках выполняемых операций) характеристики.

Каждая модель ресурса представляет из себя структуру данных, которая должна реализовывать три метода:

— метод привязки операции к модели ресурса;

— метод, осуществляющий проверку возможности выполнения данной операции моделью ресурса;

— метод, осуществляющий логику работы и в котором происходит изменение состояния данной модели.

Под привязкой операции к модели подразумевается добавление операции в очередь на выполнение и, если это первая привязанная для данного продукта операция, то добавление продукта в очередь на распределение. Привязка осуществляется в начале работы системы, что позволяет ресурсам манипулировать ядром имитационного моделирования разрешая или запрещая выбирать привязанные к ним

операции для расчета, что может повлечь за собой изменение последовательности выполнения операций и, соответственно, расчетного времени выполнения карты технологического процесса.

Проверка производится во время работы системы и именно здесь происходит отбор операций в соответствии с внутренним состоянием модели.

Логика осуществляется при выборке операции ядром и для каждой вызывается два раза: чтобы отметить состояние модели в начале и в конце расчета операции.

блок-схема ресурсов

СПП имеет несколько видов ресурсов, одним из которых является модель ресурса сборочной линии, который описывает несколько однотипных, то есть с одинаковым числом рабочих постов, физических сборочных линий. Сборочная линия - это способ перемещения заготовки от одного рабочего поста к другому; на каждом посту выполняются закрепленные за ним операции. Под постами понимаются заготовкоместа, оснащенные соответствующим технологическим оборудованием и предназначенными для технического воздействия на заготовку для осуществления фиксированного перечня операций. Объединение нескольких сборочных линий в одну обуславливается упрощением как взаимодействия с ядром имитационного моделирования так и упрощением управления моделью потому как в любом случае (даже когда все линии будут различны по количеству постов) количество линий в модели будет всегда меньше либо равно количеству физических сборочных линии, а также возможностью инкапсуляции реализации распределения заготовок и связанных с ними операций по сборочным [illegible]1 внутри модели.

Добавить цифры к линиям, чтобы ссылаться на конкретные

Главным предназначением данной модели является ограничение перемещения продукции внутри ресурса (см. 2.2), и моделирование работы физического сборочного конвейера. С одной стороны ограничивается перемещение между сборочными линиями: к какой продукт был

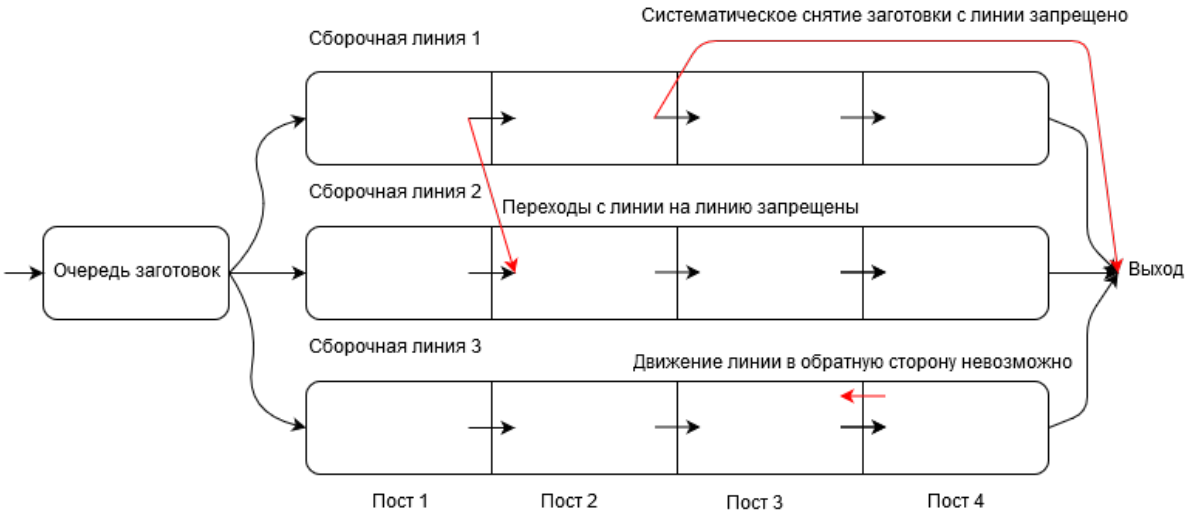

Рисунок 2.2 — Схема модели ресурса с вариантами перемещения заготовки внутри

привязан, на той он и останется до окончания выполнения всех операций, относящихся к данному продукту и привязаны к постам данной сборочной линии. С другой - ограничивается перемещение продукции между рабочими постами: продукт должен двигаться последовательно с поста на пост (см. 2.2).

Одной из ключевых особенностей практически любой сборочной линии является синхронизация передвижения продукции между постами. Это означает что такт производства (время,в течение,которого заготовка пребывает на посту) будет равен максимальной временной отметке среди всех постов или другими словами: каждая заготовка сможет сменить пост только после того, как все остальные заготовки будут готовы к смене своих постов.

Для реализации необходимого функционала, были введены структуры, описывающие линии, посты, очередь продукции, и проекцию привязки операций к постам линий. Каждая из линий модели сборочной линии характеризуется временем начала текущего рабочего такта сборочной линии, максимальным временем рабочего такта и набором рабочих постов, каждый из которых описывается состоянием (выполняются работы, простаивает, отсутствует продукция на посту), временной меткой данного поста и продукцией, которая на данный момент находится на нем. Максимальное время отражает какое время

работал пост с начала работы системы. Так как карта технического процесса позволяет, при возможности, производить параллельные операции над заготовкой, то необходимо знать, с какого времени был начат такт, для чего и вводится время начала работы поста. Время начала текущего рабочего такта показывает время с которого началась работа на текущем посту над текущей заготовкой и для всех постов она равна (из-за синхронизации постов).

схема постов с флагами

Проекция привязки операций к постам линий - структура данных необходимая для динамической привязки операций. Так как во время привязки операции система не может оценить время выполнения продукции и, следовательно, распределив продукты до начала работы может сложиться ситуация, когда одна из линий будет работать намного дольше или наоборот по сравнению с другими линиями, что может привести к простою производству. Следовательно необходимо, не привязывая операцию к определенной линии, обозначить к какому посту относится операция (см. 2.3). Для этого и была создана данная структура, содержащая то же количество постов что и остальные линии, но не описывающая какую либо физическую сборочную линию, а являющуюся по сути проекцией всех остальных. Внутри каждого поста данной проекции находится хэш-таблица (структура данных, позволяющая хранить пары (ключ, значение) и выполнять три операции: операцию добавления новой пары, операцию поиска и операцию удаления пары по ключу), в которой ключом является конкретная единица продукции, характеризуемая типом продукции (названием) и серийным номером, а значением - список операций, которые необходимо выполнить над данным продуктом на данном посту.

Во время выполнения система, для определения возможности выполнения операции опрашивает все линии с целью определения возможности выполнения данной операции и местонахождения продукта:

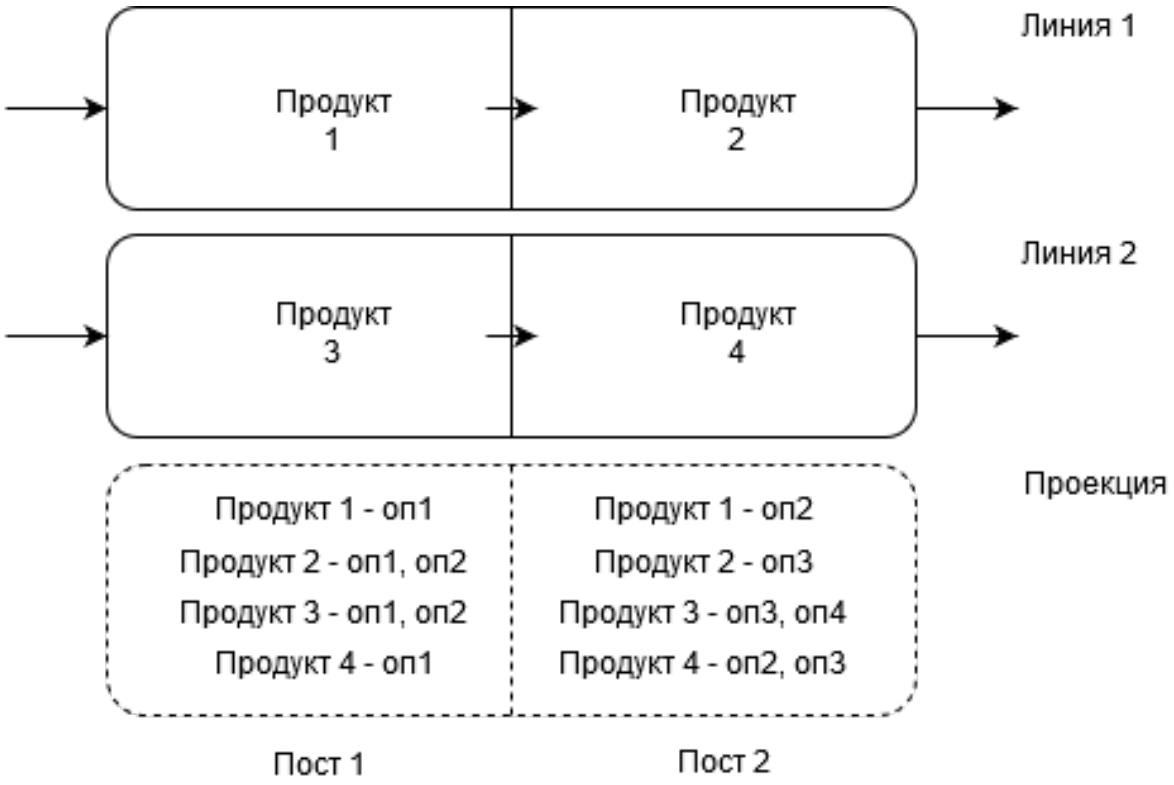

Рисунок 2.3 — Схема сборочной линии с проекцией привязки операций к постам

— при отсутствии продукта на всех линиях:

1) инициируется проверка всех линий на возможность загрузки первого поста (то есть пуст ли он);

2) из получившегося списка линий (если количество больше нуля, иначе переход к 4) выбирается линия, временная отметка которой является наименьшей среди всех доступных линий;

3) возвращается разрешение на выполнение данной операции и временная метка, выбранная на предыдущем шаге;

4) иначе (количество линий равно нулю) возвращается запрет на выполнение данной операции на текущей итерации.

— если продукт находится на какой-либо линии, то производится опрос проекции:

а) при нахождении нужной операции на посту, на котором находится в данный момент продукт, возвращается разрешение на выполнение и временная метка, с которой может производиться данная операция;

б) иначе возвращается запрет выполнения.

Ядро имитационного моделирования последовательно переберет все доступные на данной итерации и выберет одну из тех, что получили разрешение от всех моделей на выполнение. Если таковых не будет, то система известит о невозможности дальнейшей работы вследствие логической ошибки при задании параметров. В ином случае произойдет вызов метода для того, чтобы отметить состояние модели в начале выполнения операции. При этом если продукт не был загружен на линию, то произойдет загрузка продукта на линию с минимальной временной отметкой, иначе произойдет проверка на возможность сдвига линии.

Во второй раз метод будет вызван для того, чтобы отметить окончание операции, что означает что данная операция будет удалена из проекции, временная метка поста будет изменена с учетом длительности операции, и, если операций на данном посту для данного продукта не осталось, то состояние поста изменится на 'готов к сдвигу линии' и будет совершена проверка возможности сдвига линии.

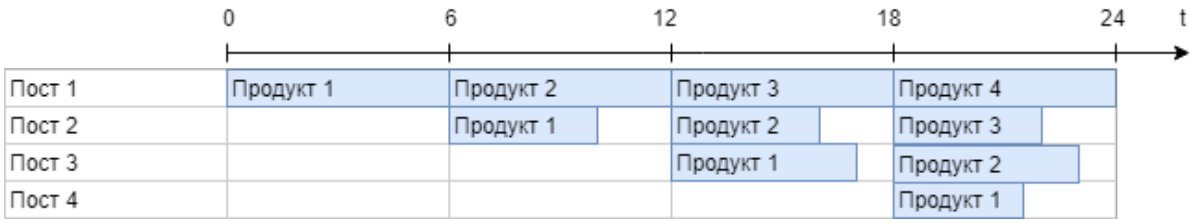

Рисунок 2.4 — Диаграмма, изображающая сдвиг линии

Сдвиг линии - процесс, при котором все посты на сборочной линии передают свою заготовку на следующий пост и принимают заготовку с предыдущего. Данный процесс единовременен для всех постов, не может быть разделен или проигнорирован каким-либо постом и выполняется после получения от всех постов сигнала о готовности к сдвигу. При этом происходит синхронизация постов, то есть все

посты, и, соответственно вся линия, получают одну временную отметку - сумма предыдущей отметки начала рабочего такта и длительности текущего рабочего такта линии. С этой отметки начинается отсчет следующего такта (см. 2.4).

```
2019/05/12 19:25:08 Line (0) product (M, 0) loaded on station 0
2019/05/12 19:25:08 Line (0) product (M, 0) changing timestamp (4)
2019/05/12 19:25:08 Line (0) product (M, 0) changing station (0 → 1)
2019/05/12 19:25:08 Line (0) product (M, 0) changing timestamp (7)
2019/05/12 19:25:08 Line (1) product (M, 1) loaded on station 0
2019/05/12 19:25:08 Line (1) product (M, 1) changing timestamp (4)
2019/05/12 19:25:08 Line (1) product (M, 1) changing station (0 → 1)
2019/05/12 19:25:08 Line (1) product (M, 1) changing timestamp (7)
2019/05/12 19:25:08 Line (0) product (M, 2) loaded on station 0
2019/05/12 19:25:08 Line (0) product (M, 2) changing timestamp (8)
2019/05/12 19:25:08 Line (0) product (M, 0) changing station (1 → 2)
2019/05/12 19:25:08 Line (0) product (M, 2) changing station (0 → 1)
2019/05/12 19:25:08 Line (0) product (M, 0) changing timestamp (14)
2019/05/12 19:25:08 Line (1) product (M, 1) changing station (1 → 2)
2019/05/12 19:25:08 Line (1) product (M, 1) changing timestamp (13)
2019/05/12 19:25:08 Line (1) product (M, 1) unloaded from line (timestamp: 13)
2019/05/12 19:25:08 Line (0) product (M, 2) changing timestamp (11)
2019/05/12 19:25:08 Line (0) product (M, 0) unloaded from line (timestamp: 14)
2019/05/12 19:25:08 Line (0) product (M, 2) changing station (1 → 2)
2019/05/12 19:25:08 Line (0) product (M, 2) changing timestamp (20)
2019/05/12 19:25:08 Line (0) product (M, 2) unloaded from line (timestamp: 20)
```

Рисунок 2.5 — Лог с результатами работы модели ресурса сборочной линии

обосновать переход к заключению

На каждой итерации работы модели, создается несколько записей в лог для отслеживания состояния модели в данный момент и по которым можно отследить некорректное поведение либо просто узнать причины, по которым ядро имитационного моделирования отработало именно в такой последовательности (естественно при условии работы с моделью ресурса сборочной линии). На рисунке 2.5 видно, что модель ресурса охватывает две линии (0 и 1) физических конвейеров и работает с тремя продуктами типа ’M’ (0, 1 и 2). Также в данном логе видны все передвижения продуктов между постами (station) и все изменения временных меток и результирующие метки при выгрузке с последних постов.

По завершении основной разработке было произведено тестирование как ручное (просмотром логов в поисках ошибок работы), так и

автоматизированное с написанием автоматизированных как модульных (для проверки самой модели) так и нескольких интеграционных тестов для проверки работы с ядром имитационного моделирования и другими моделями ресурсов.

2.4 Модуль отображения логического времени на физическое

Так как расчет выполнения операции (как и всей карты технологического процесса) ядром имитационного моделирования производится в логическом времени,то есть во времени отсчитываемом от нуля, существует необходимость в отображении (соответствие между элементами двух множеств) логического времени на физическое, которое используется в повседневной жизни. Одной из главных сложностей, возникающих при этом, является неоднородность рабочего времени, которая проявляется в рабочем графике (чередование интервалов рабочего и нерабочего времени), наличии выходных, перенесенных дней. Другой сложностью является наличие в системе 'обратного расчета', при котором планирование ведется от даты 'дедлайна' (дата или время, к которому должна быть выполнена задача), что накладывает некоторые ограничения на реализацию данной компоненты.

Для отображения логического времени на физическое был предложен итеративный процесс, который осуществляет 'переход' к необходимому времени путем последовательного перебора дат.

Как было сказано ранее, из-за того что рабочее время является дискретным, мы не имеем возможности просуммировать начальную дату и значение поданного логического времени. Это ведет к тому, что необходимо синхронизировать логическое и физическое время,и это достигается путем последовательного периодического отображения конкретного логического времени на физическое (см. 2.6). Это подразумевает под собой наличие двух массивов чисел или 'осей':

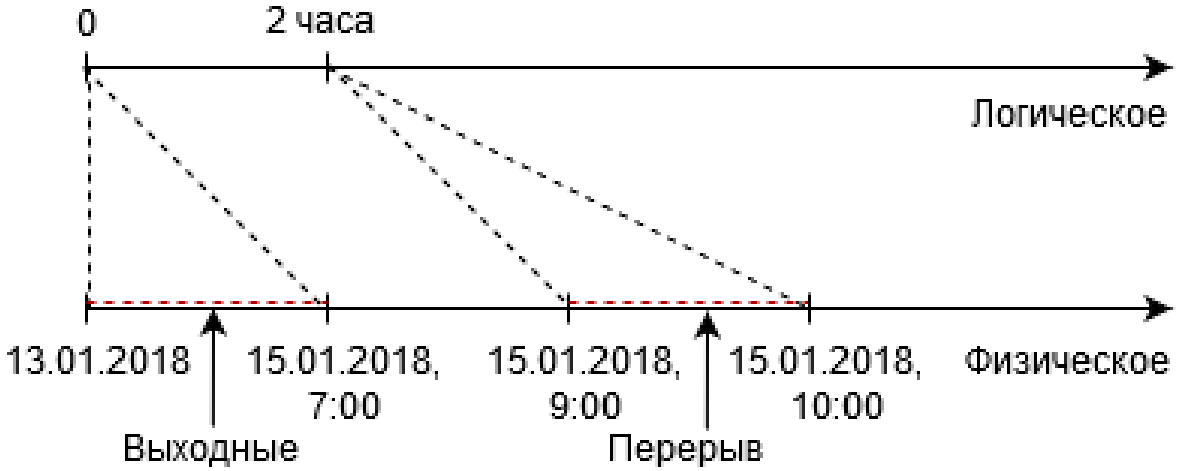

Рисунок 2.6 — Пример отображения оси логического на ось физического времени

— оси логического времени, которая начинается с нуля и единица которой соответствует одной секунде (необходимости в более точном отображении нет);

— оси физического времени, на которой может быть отложено любая дата физического времени, отсчет которой начинается 1 января 1970 года 00:00:00 (эпоха Unix).

Особенностью оси физического времени является наличие на ней ’выколотых’ промежутков времени, в которые работа не ведется и операции не выполняются, следовательно, об этих промежутках системе необходимо знать, что подразумевает данную информацию как входные данные.

Входными данными для модуля являются:

— дата с которой необходимо начинать отсчет;

— логическое время, которого необходимо достигнуть;

— конфигурация модуля.

Дата является точкой на физической оси, на которую будет отображаться нуль логической. Представляет собой количество секунд, прошедшее с начала эпохи Unix.

Логическое время - количество секунд, которое должно быть отложено на логической оси. В силу непрерывности физической оси, каждой логической точке сопоставляется отрезок на физической оси, сопоставляется пара чисел - границ данного отрезка.

Конфигурация модуля - вспомогательные данные используемые для определения модулем какие промежутки необходимо пропускать в процессе работы. Состоит из данных о рабочем графике занятого персонала (интервалы рабочего времени), шаблонном расписании на неделю (например, суббота, воскресенье - выходные, пятница - 'короткий' день, остальные - стандартные рабочие дни) и набор информации о датах, которые являются днями-исключениями и соответствующей информацией о графике работы в данные дни.

После запуска, модуль получает параметры и совершает проверку последних на корректность и непротиворечивость (например, если два дня имеют пересечения временных промежутков то они противоречивы, ведь ресурс не может работать одновременно в двух сменах) как в рамках смен одного так соседних дней. Далее производится определение режима работы: прямой, обратный расчет или проверка времени:

— прямой расчет - задается дата начала отсчета, логическое время и расчет ведется до нахождения даты окончания работ (см 2.7);

— обратный расчет - задается дата дедлайна, логическое время и расчет ведется до нахождения времени начала работ(см 2.8);

— проверка времени - задается дата и логическое время равное нулю, что запускает оба предыдущих расчета пока не будет найдено первое ненулевое время в обоих направлениях от даты начала расчета(см 2.9).

Выбрав режим работы сбрасывается счетчик текущего логического времени до нуля и счетчик текущего физического времени до стартовой даты. Затем итеративно, пока текущее логическое время не превысит необходимое производится поиск следующей даты. Алгоритмически, поиск даты работает следующим образом:

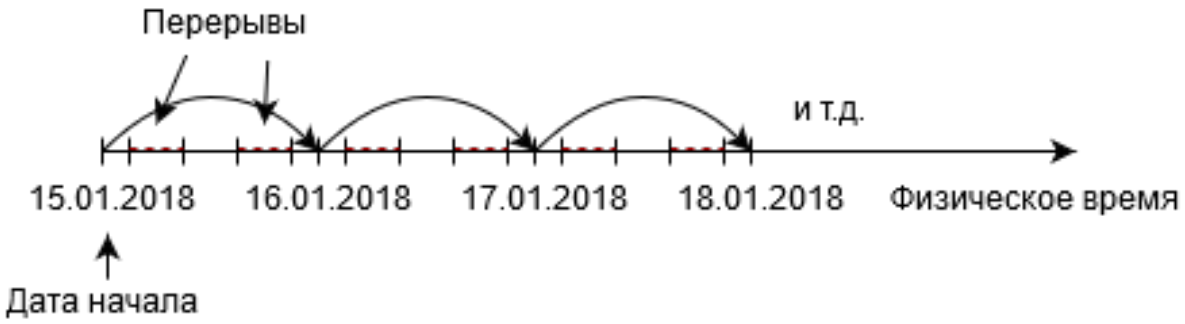

Рисунок 2.7 — Схема прямого расчета

Рисунок 2.8 — Схема обратного расчета

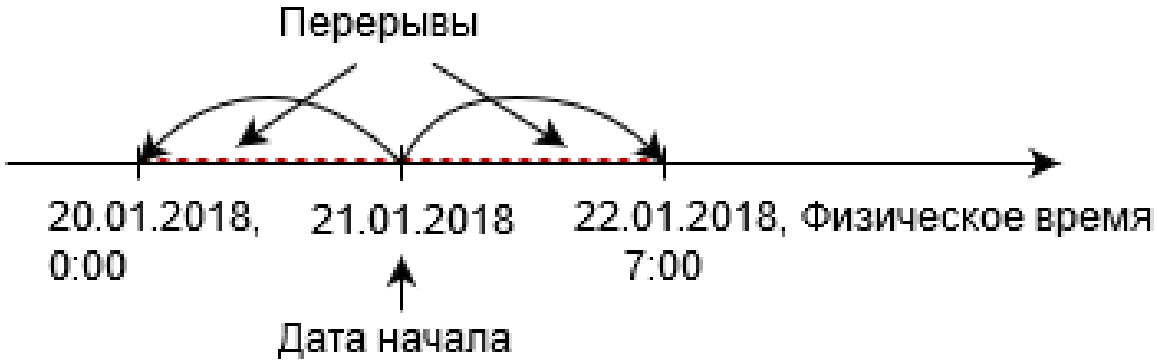

Рисунок 2.9 — Схема проверки времени

1) определяются интервалы рабочих смен относящихся к текущему дню:

а) при отсутствии таковых, к текущей дате прибавляется один день и затем возврат к п.1.

2) отсортированные в порядке возрастания интервалы последовательно перебираются и их длительности прибавляются к текущему логическому и физическому времёнам:

а) при превышении текущим логическим временем необходимого, переход к п.3;

б) если все интервалы были просуммированы, но необходимое логическое время не превышено - переход к п.1;

3) разность текущего и необходимого логического времён вычитается из физического времени, при этом сохраняя данное значение как левую (правую при обратном расчете) границу и продолжается расчет для выявления правой (левой) границы промежутка.

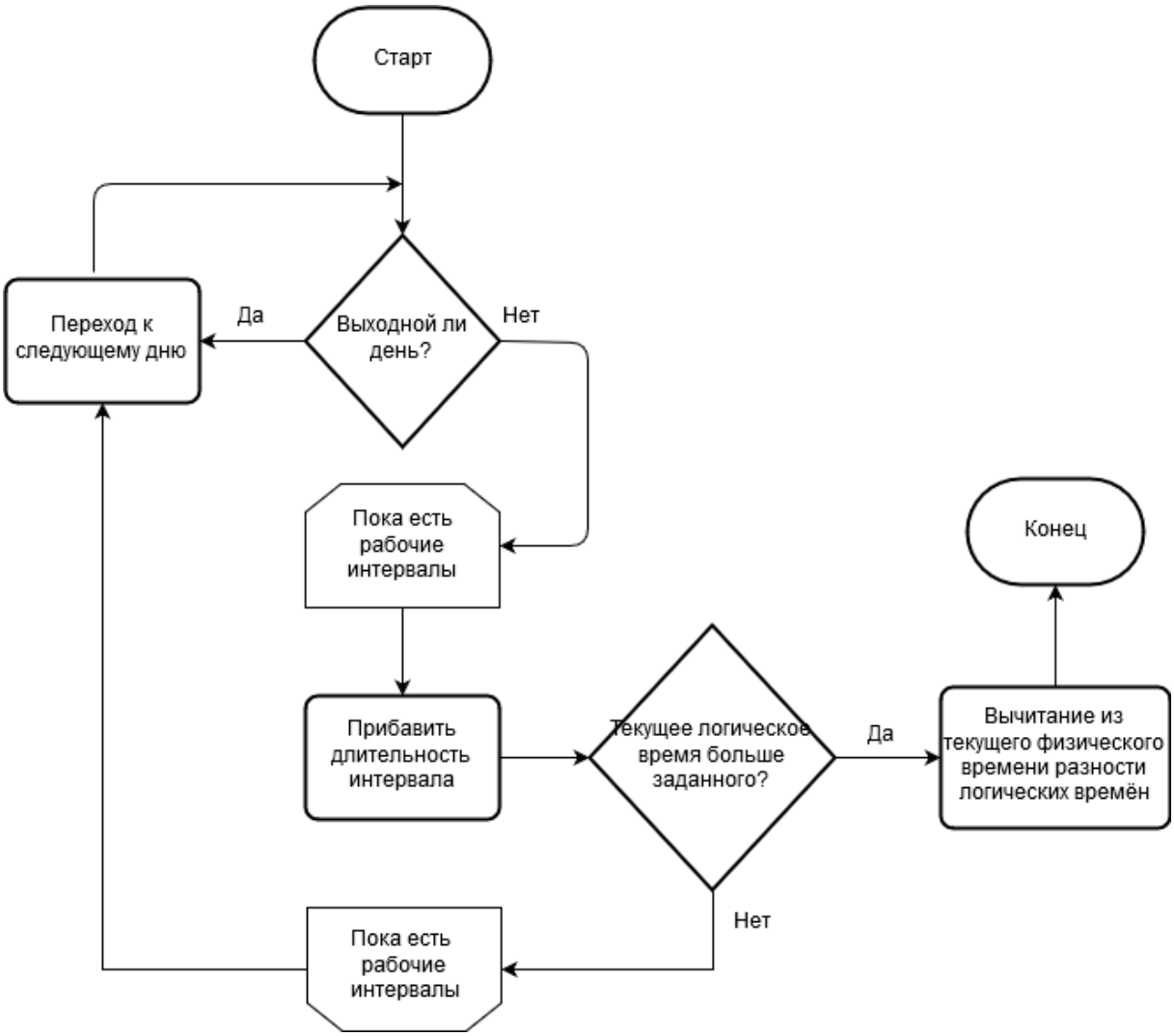

Рисунок 2.10 — Блок-схема модуля

Определение интервалов рабочего времени происходит взятием даты из текущего физического времени, после чего начинается определение принадлежности данной даты к перенесенным датам после чего есть два варианта развития ситуации:

— дата является перенесенным днем и модуль получает информацию о расписании которое нужно применить;

— дата не является перенесенным днем и получение информации происходит исходя из того, каким днем недели является данная дата.

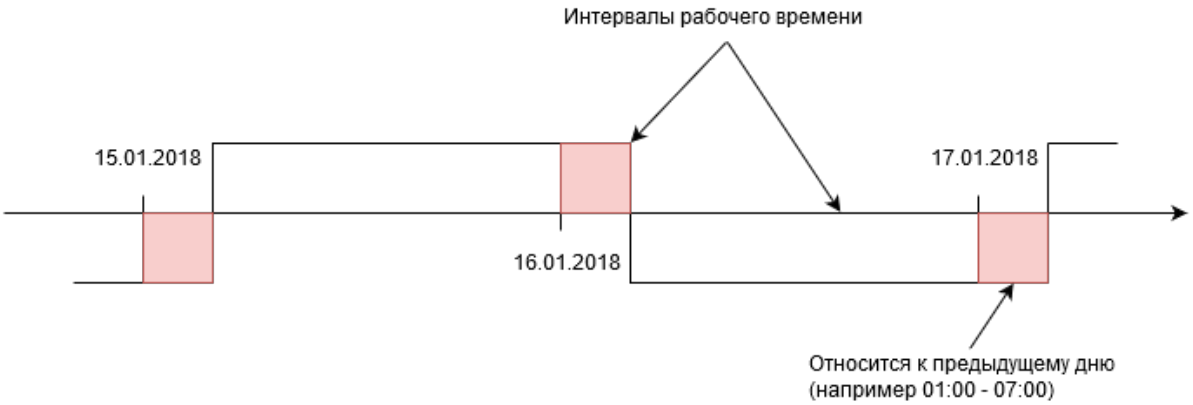

Рисунок 2.11 — Смещение интервалов рабочего времени относительно рабочего дня

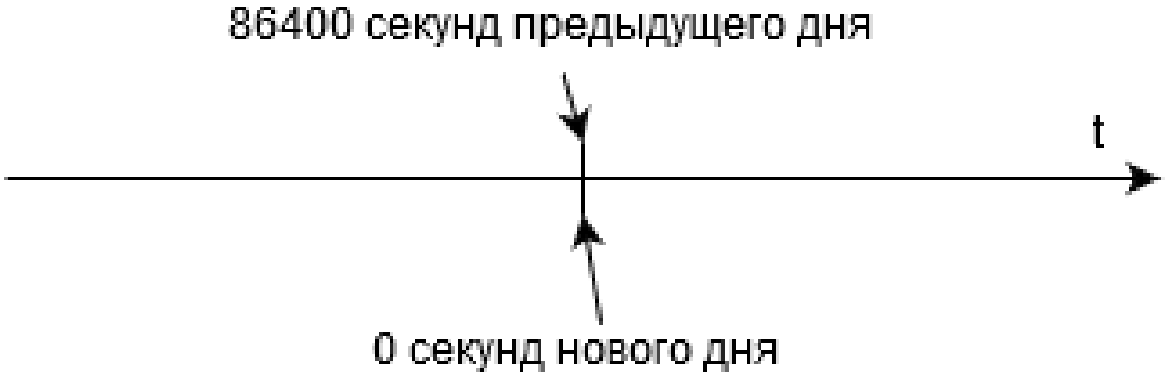

Рисунок 2.12 — Граница двух дней

Так как на реальных производственных площадках нередко случается так, что рабочие интервалы, принадлежащие к рабочему дню и сам рабочий день смещены относительно друг друга (см. 2.11). При этом возникают трудности с определением интервалов рабочего времени в связи с тем, что для определения используется количество секунд с начала эпохи Unix и до нуля часов нуля минут и нуля секунд нужной даты, по которому, используя инструментарий языка, каким днем недели является нужная дата и, соответственно, задать для нее шаблон рабочего времени. И если эти трудности практически не влияют на прямой расчет, то с обратным все немного сложнее. Так как одном дне 86400 секунд и, если рассматривать границу двух дней (см. 2.12), то 86400 секунда предыдущего дня будет равна или тем же что и

нулевая секунда нового дня. Можно сказать, что 86400 - левый предел, а 0 - правый предел в данной точке (особенно если изображать в виде окружности). Но, в связи с тем, что данная недетерминированность присутствует лишь при рассмотрении человеком данной ситуации, а система может распознавать диапазон от 0 до 86399 секунд, то для определения интервалов рабочего графика (при обратном расчете) используется именно величина в 86399 секунд, при условии, что данное допущение не влияет на расчет, как последняя секунда текущего дня.

```
2019/05/10 16:19:10 New config №1: {
    ScheduleTemplate: {
        Sunday    :→ Day Off,
        Monday    :→ Workday,
        Tuesday   :→ Workday,
        Wednesday:→ Workday,
        Thursday :→ Workday,
        Friday    :→ Shortday,
        Saturday :→ Day Off,
    },
    ExceptionDates:{
        1514764800 (01/01/2018 00:00:00): Day Off,
        1514851200 (02/01/2018 00:00:00): Day Off,
        1514937600 (03/01/2018 00:00:00): Day Off,
        1515024000 (04/01/2018 00:00:00): Day Off,
        1515110400 (05/01/2018 00:00:00): Day Off,
        1515196800 (06/01/2018 00:00:00): Day Off,
        1515283200 (07/01/2018 00:00:00): Day Off,
        1515369600 (08/01/2018 00:00:00): Day Off,
        1515456000 (09/01/2018 00:00:00): Day Off,
        1515542400 (10/01/2018 00:00:00): Day Off,
        1515628800 (11/01/2018 00:00:00): Day Off,
        1515715200 (12/01/2018 00:00:00): Day Off,
        1515801600 (13/01/2018 00:00:00): Day Off,
        1515888000 (14/01/2018 00:00:00): Day Off,
    },
    ScheduleForDay:{
        Day Off : {
        }
        Shortday : {
            { ShiftID: 2, Starts: 28800  (01 day 08:00:00), Ends: 46800  (01 day 13:00:00) }
            { ShiftID: 2, Starts: 50400  (01 day 14:00:00), Ends: 64800  (01 day 18:00:00) }
        }
        Workday : {
            { ShiftID: 1, Starts: 28800  (01 day 08:00:00), Ends: 46800  (01 day 13:00:00) }
            { ShiftID: 1, Starts: 50400  (01 day 14:00:00), Ends: 64800  (01 day 18:00:00) }
            { ShiftID: 1, Starts: 68400  (01 day 19:00:00), Ends: 86400  (02 day 00:00:00) }
            { ShiftID: 1, Starts: 90000  (02 day 01:00:00), Ends: 111600 (02 day 07:00:00) }
        }
    }
}
```

Рисунок 2.13 — Пример конфигурации модуля

Возвращаясь к пункту 3 алгоритма, при нахождении нужного времени, работа модуля не прекращается, а ведется до момента нахождения второй границы промежутка, на который отображается необходимое логической время.

Как было сказано ранее - физическое время непрерывно, а следовательно когда производится отображение на него логического времени,

в результате должен получиться промежуток (см. 2.6), который и характеризуют две границы. Эта пара чисел, характеризующие начало и конец отрезка которые отображаются на логическую ось в точке, значение которой равно входному логическому времени и являются выходными данными данного модуля.

```
2019/05/10 16:54:09 Start time is: 1514678400 (31/12/2017 00:00:00)
2019/05/10 16:54:09 Given logical time is: 518400 (144 in hours)
2019/05/10 16:54:09  Logical time | Physical time
2019/05/10 16:54:09 --------------|---------------------------------
2019/05/10 16:54:09  0             | 1515974400 (15/01/2018 00:00:00)
2019/05/10 16:54:09  18000         | 1516021200 (15/01/2018 13:00:00)
2019/05/10 16:54:09  32400         | 1516039200 (15/01/2018 18:00:00)
2019/05/10 16:54:09  50400         | 1516060800 (16/01/2018 00:00:00)
2019/05/10 16:54:09  72000         | 1516086000 (16/01/2018 07:00:00)
2019/05/10 16:54:09  90000         | 1516107600 (16/01/2018 13:00:00)
2019/05/10 16:54:09  104400        | 1516125600 (16/01/2018 18:00:00)
2019/05/10 16:54:09  122400        | 1516147200 (17/01/2018 00:00:00)
2019/05/10 16:54:09  144000        | 1516172400 (17/01/2018 07:00:00)
2019/05/10 16:54:09  162000        | 1516194000 (17/01/2018 13:00:00)
2019/05/10 16:54:09  176400        | 1516212000 (17/01/2018 18:00:00)
2019/05/10 16:54:09  194400        | 1516233600 (18/01/2018 00:00:00)
2019/05/10 16:54:09  216000        | 1516258800 (18/01/2018 07:00:00)
2019/05/10 16:54:09  234000        | 1516280400 (18/01/2018 13:00:00)
2019/05/10 16:54:09  248400        | 1516298400 (18/01/2018 18:00:00)
2019/05/10 16:54:09  266400        | 1516320000 (19/01/2018 00:00:00)
2019/05/10 16:54:09  288000        | 1516345200 (19/01/2018 07:00:00)
2019/05/10 16:54:09  306000        | 1516366800 (19/01/2018 13:00:00)
2019/05/10 16:54:09  320400        | 1516579200 (22/01/2018 00:00:00)
2019/05/10 16:54:09  338400        | 1516626000 (22/01/2018 13:00:00)
2019/05/10 16:54:09  352800        | 1516644000 (22/01/2018 18:00:00)
2019/05/10 16:54:09  370800        | 1516665600 (23/01/2018 00:00:00)
2019/05/10 16:54:09  392400        | 1516690800 (23/01/2018 07:00:00)
2019/05/10 16:54:09  410400        | 1516712400 (23/01/2018 13:00:00)
2019/05/10 16:54:09  424800        | 1516730400 (23/01/2018 18:00:00)
2019/05/10 16:54:09  442800        | 1516752000 (24/01/2018 00:00:00)
2019/05/10 16:54:09  464400        | 1516777200 (24/01/2018 07:00:00)
2019/05/10 16:54:09  482400        | 1516798800 (24/01/2018 13:00:00)
2019/05/10 16:54:09  496800        | 1516816800 (24/01/2018 18:00:00)
2019/05/10 16:54:09  514800        | 1516838400 (25/01/2018 00:00:00)
2019/05/10 16:54:09  518400        | 1516845600 (25/01/2018 02:00:00)
2019/05/10 16:54:09 Result:
2019/05/10 16:54:09 ------------------------------------------------------------
2019/05/10 16:54:09  518400        | [1516845600          - 1516845600          ]
2019/05/10 16:54:09  518400        | [25/01/2018 02:00:00 - 25/01/2018 02:00:00]
2019/05/10 16:54:09 Found config №1
2019/05/10 16:54:09 Start time is: 1514678400 (31/12/2017 00:00:00)
2019/05/10 16:54:09 Given logical time is: 504000 (140 in hours)
2019/05/10 16:54:09  Logical time | Physical time
2019/05/10 16:54:09 --------------|---------------------------------
```

Рисунок 2.14 — Пример прямого расчета модулем

В результате работы над модулем были написаны тесты и проведено тестирование, результаты одного из которых можно увидеть на изображениях 2.13 и 2.14. На 2.13 изображена конфигурация модуля, которая показывает как производится совмещение логического и физического дня, обработка перенесенных дней. На 2.14, перед расчетом нового отображения, можно отметить, что при использовании той же конфигурации не производится ее повторный вывод в лог, что позволяет сократить его длину, а следовательно улучшить читаемость. Рисунок

2.14 показывает итеративный процесс как результат прибавления каждого нового интервала к текущему времени.

```
2019/05/10 19:33:39 Start time is: 1516845600 (25/01/2018 02:00:00)
2019/05/10 19:33:39 Given logical time is: -518400 (-144 in hours)
2019/05/10 19:33:39   Logical time | Physical time
2019/05/10 19:33:39 --------------|---------------------------------
2019/05/10 19:33:39   0           | 1516845600 (25/01/2018 02:00:00)
2019/05/10 19:33:39   3600        | 1516838400 (25/01/2018 00:00:00)
2019/05/10 19:33:39   21600       | 1516820400 (24/01/2018 19:00:00)
2019/05/10 19:33:39   36000       | 1516802400 (24/01/2018 14:00:00)
2019/05/10 19:33:39   54000       | 1516780800 (24/01/2018 08:00:00)
2019/05/10 19:33:39   75600       | 1516752000 (24/01/2018 00:00:00)
2019/05/10 19:33:39   93600       | 1516734000 (23/01/2018 19:00:00)
2019/05/10 19:33:39   108000      | 1516716000 (23/01/2018 14:00:00)
2019/05/10 19:33:39   126000      | 1516694400 (23/01/2018 08:00:00)
2019/05/10 19:33:39   147600      | 1516665600 (23/01/2018 00:00:00)
2019/05/10 19:33:39   165600      | 1516647600 (22/01/2018 19:00:00)
2019/05/10 19:33:39   180000      | 1516629600 (22/01/2018 14:00:00)
2019/05/10 19:33:39   198000      | 1516406400 (20/01/2018 00:00:00)
2019/05/10 19:33:39   212400      | 1516370400 (19/01/2018 14:00:00)
2019/05/10 19:33:39   230400      | 1516348800 (19/01/2018 08:00:00)
2019/05/10 19:33:39   252000      | 1516320000 (19/01/2018 00:00:00)
2019/05/10 19:33:39   270000      | 1516302000 (18/01/2018 19:00:00)
2019/05/10 19:33:39   284400      | 1516284000 (18/01/2018 14:00:00)
2019/05/10 19:33:39   302400      | 1516262400 (18/01/2018 08:00:00)
2019/05/10 19:33:39   324000      | 1516233600 (18/01/2018 00:00:00)
2019/05/10 19:33:39   342000      | 1516215600 (17/01/2018 19:00:00)
2019/05/10 19:33:39   356400      | 1516197600 (17/01/2018 14:00:00)
2019/05/10 19:33:39   374400      | 1516176000 (17/01/2018 08:00:00)
2019/05/10 19:33:39   396000      | 1516147200 (17/01/2018 00:00:00)
2019/05/10 19:33:39   414000      | 1516129200 (16/01/2018 19:00:00)
2019/05/10 19:33:39   428400      | 1516111200 (16/01/2018 14:00:00)
2019/05/10 19:33:39   446400      | 1516089600 (16/01/2018 08:00:00)
2019/05/10 19:33:39   468000      | 1516060800 (16/01/2018 00:00:00)
2019/05/10 19:33:39   486000      | 1516042800 (15/01/2018 19:00:00)
2019/05/10 19:33:39   500400      | 1516024800 (15/01/2018 14:00:00)
2019/05/10 19:33:39   518400      | 1514592000 (30/12/2017 00:00:00)
2019/05/10 19:33:39 Result:
2019/05/10 19:33:39 ------------------------------------------------------------
2019/05/10 19:33:39   518400      | [1516003200          - 1514570400         ]
2019/05/10 19:33:39   518400      | [15/01/2018 08:00:00 - 29/12/2017 18:00:00]
```

Рисунок 2.15 — Пример обратного расчета модулем

Также были проведены тесты обратного расчета и проверки времени, результаты которых можно видеть на изображениях 2.15 и 2.16. Пояснение к проверке даты: тут используется конфигурация, когда 2 января является выходным, проверка которой и производится, что позволяет продемонстрировать работу данного решения.

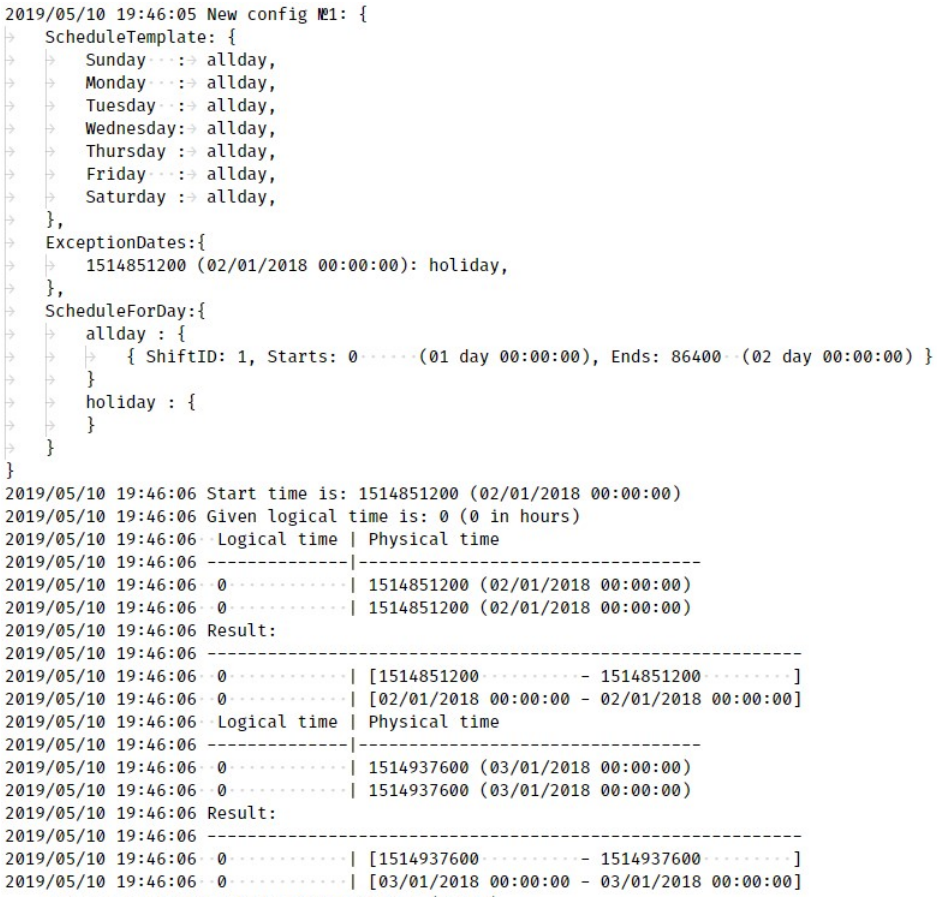

Рисунок 2.16 — Пример проверки модулем времени

3 Организация консистентного хранилища данных

Выбор хранилища данных

Система планирования и прогнозирования, как и практически любая система получающая и обрабатывающая данные, нуждается в хранилище данных - базе данных. Хранилище данных может быть разделено на две составляющие:

— хранилище временных данных;

— хранилище постоянных данных.

Под временными данными подразумеваются данные, которые получаются в процессе работы системы (например оперативные и объемно-календарные планы) и, при необходимости, могут быть рассчитаны заново, хоть и с некоторыми затратами (время, вычислительные мощности), при условии отсутствия хранилища данных. В данной работе этот вид данных и хранилище для них не рассматривается.

С другой стороны постоянные данные - данные, которые задаются пользователем (в данном случае организацией) и потеря которых в лучшем случае приведет к необходимости заново добавлять их в систему, а в худшем - приведет к утрате этих данных. В любом случае потеря постоянных данных ведет к нарушениям в работе системы, что ведет к необходимости организации консистентного хранилища данных.

Консистентность - требование к данным, получаемым из базы данных, которое заключается в том, что последние должны быть целостны и непротиворечивы. Под целостностью данных подразумевается соответствие имеющейся в базе данных информации её внутренней логике, структуре и явно заданным правилам. Любое правило, направленное на ограничение возможного состояния базы данных называют ограничением целостности. Помимо целостных, данные должны также быть непротиворечивыми, что означает, что в базе данных нет логического противоречия, то есть некоторого утверждения и его отрицания. Со стороны систем управления базами данных (совокупность программ-

ных и лингвистических средств общего или специального назначения, обеспечивающих управление созданием и использованием баз данных) свойство консистентности выполняется на уровне транзакций (одним из требований к которой и является консистентность): если одна из команд транзакции не прошла проверку ограничений целостности, то вся транзакция откатывается, то есть база данных возвращается в состояние, в котором была начата транзакция.

Транзакция - группа последовательных операций с базой данных, которая представляет собой логическую единицу работы с данными. Транзакция может либо быть целиком и успешно независимо от идущих параллельно транзакций и соблюдая все ограничения целостности, либо не быть выполненной вообще, что в таком случае не должно оказать на систему никакого влияния.

В соответствии с требованиями описанными выше, можно сделать вывод о необходимости использования реляционной базы данных, одним из преимуществ которой является соответствие требованиям ACID (Atomicity, Consistency, Isolation, Durability - атомарность, консистентность, которая и интересует в первую очередь, изолированность, долговечность).

Структура базы данных

Из предыдущей главы видно, что в постоянной базе данных необходимо хранить информацию по которой будет производиться расчет объемно-календарного и оперативного планов. Для этого необходимо хранить:

— данные о заказах;

— данные о типах продукции;

— данные о продукции, которая относится к каждому заказу;

— данные о самой последовательности операций и самих операциях для каждого типа продукции;

— данные о каждой модели ресурсов;

— данные о связях между моделью ресурса и операциями.

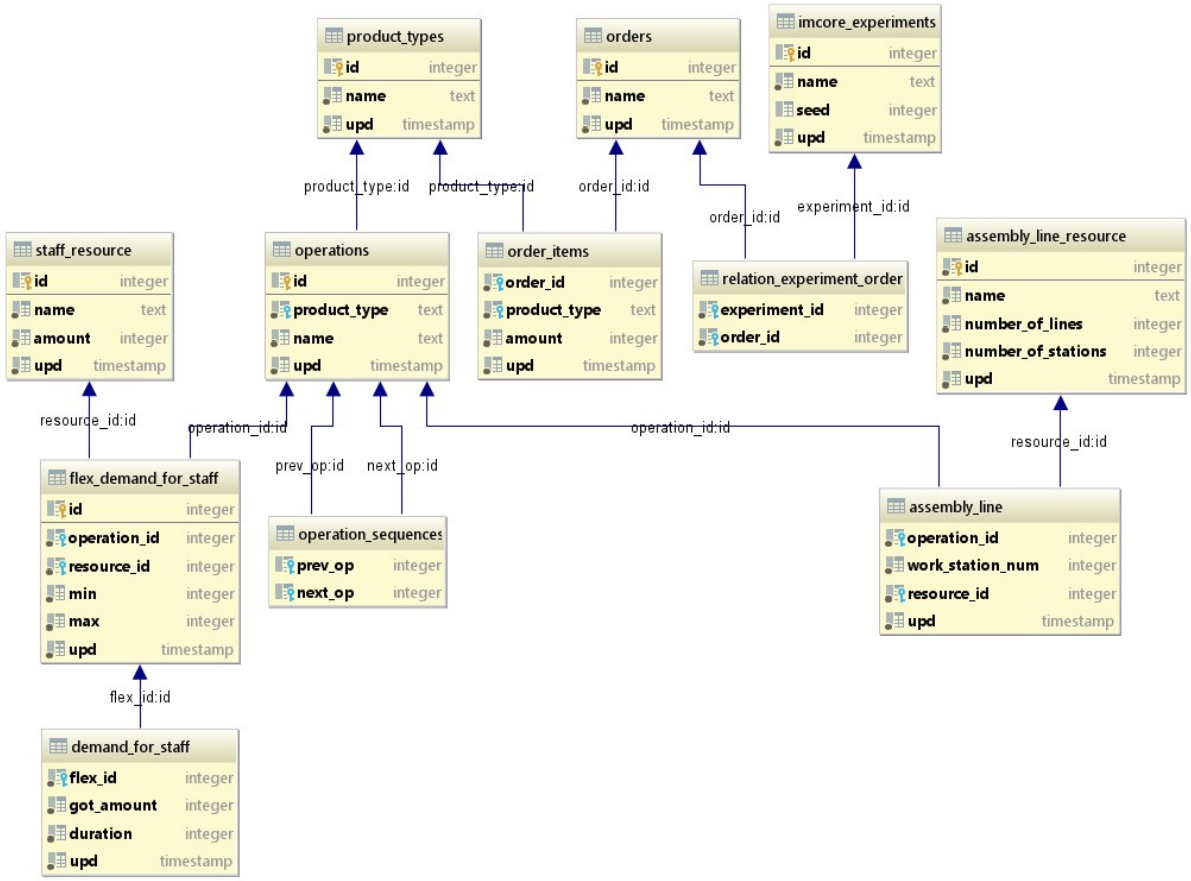

Рисунок 3.1 — Схема базы данных

Помимо этого, нужно учитывать, что кроме простого хранения данных, необходимо соблюдать ’версионность’, ввиду того, что карта технологического процесса может меняться во времени, а значит и в базе данных, требуется следить за тем, чтобы в любой момент времени было возможно использовать любую из версий данной карты, иначе новый расчет оперативного плана (при условии, что другие данные остались неизменными) приведет к созданию новой версии плана, а, следовательно, предыдущую версию восстановить будет либо очень сложно, либо, в худшем случае - невозможно.

Для достижения данной цели, в определенных таблицах было введено дополнительное поле - временная отметка обновления - ’upd’. С помощью этой метки можно как получить последние данные из базы, просто максимизируя метку ’upd’, так и получить данные на

определенный момент времени ограничивая эту метку необходимым временем.

Обоснование/Доказательство

Теория реляционных баз данных оперирует понятиями отношения (relation), от которого и пошло само название теории и баз данных основанных на ней, атрибута, домена, кортежа.

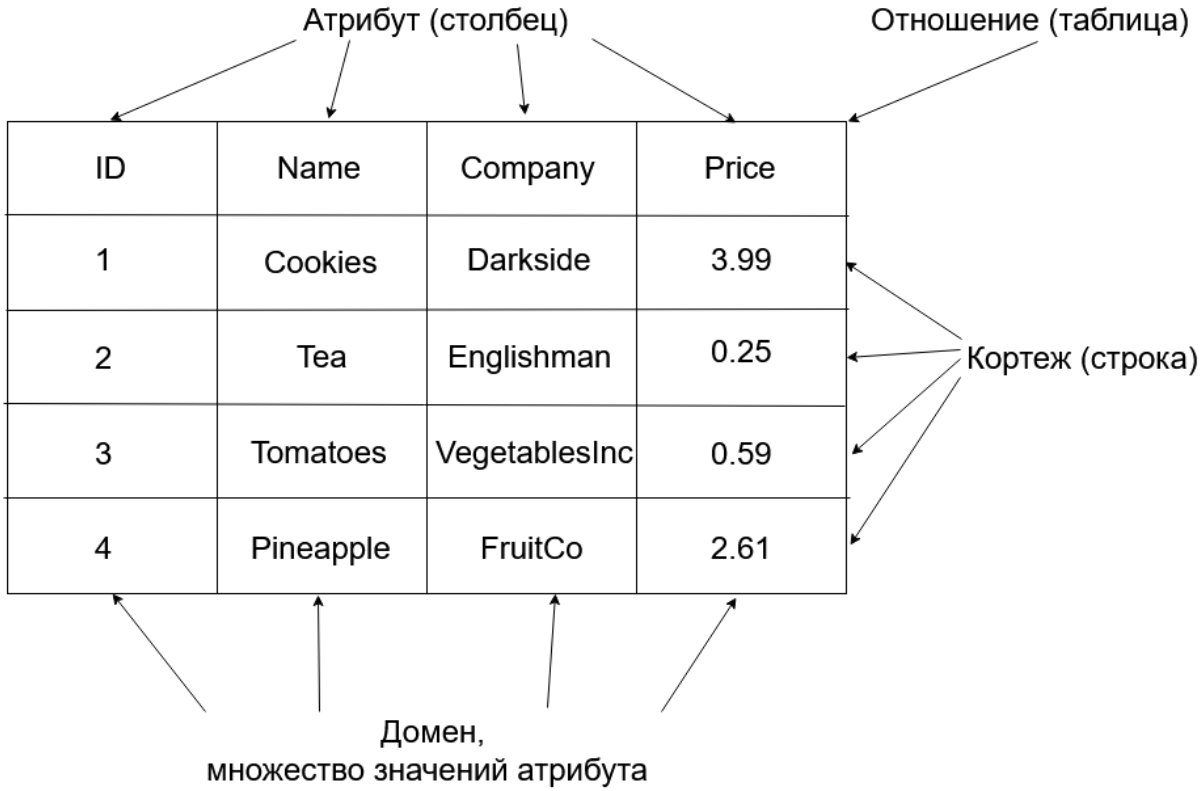

Рисунок 3.2 — Понятия реляционной базы данных

Атрибут - именованный столбец отношения. В пределах одного атрибута все значения должны быть одного типа данных, то есть принадлежать одному домену.

Домен - тип данных, множество всех допустимых значений атрибута.

Кортеж - упорядоченный набор из N элементов, где N - это число атрибутов отношения. По-другому можно сказать, что кортеж это строка или запись таблицы.

Отношение - множество упорядоченных N-кортежей. Другими

словами отношение - это двумерная (плоская) таблица, состоящая из столбцов и строк - атрибутов и кортежей.(см. 3.2)

математическое обоснование

Заключение

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Кремлев А.С., Маргун А.А., Юрьева Р.А., Власов С.М., Васильков С.Д., Пенской А.В., Николаев Н.А., Слита О.В., Жиленков А.А., Вахвиянова П.Д., Волков А.В., Коновалов Д.Е., Добриборщ Д.Э., Зименко К.А., Кирсанова А.С., Базылев Д.Н., Гребнев И.С., Болдырев И.П., Евдокимов П.С., Зимин А.Ю., Иванов В.О., Черных А.В., Халанчук Р.А., Анисимов И.В., Тулькова И.А., Лопацкий А.В., Июдина О.С., Гурин Д.А., Алексанин С.А., Малкина Е.А., Иващенко А.А., Кукин Н.А., Куприенко А.М., Леонтьев В.М., Литвяков А.С., Малютина Е.В., Попова В.О., Сурский П.В., Халанчук Р.А., Уваров М.М., Якобсон А.К., Андрианов Д.И., Анспук Д.А., Альтшулер Е.В., Телешман А.И., Николаев П.С., Пронин К.В., Мухина И.В., Вражевский С.А., Голубей А.А., Дунаев В.И., Матвеев А.А., Медведовский Е.В., Круглова А.И., Корепанов П.Ю., Колеватова М.В., Кердоль З.О., Иголкин В.А., Седунов Д.Ю., Веровенко В.Ю., Вороненкова Ю.В., Бормотов А.В., and Белявская И.Н. Интеллектуальные технологии цифрового производства. Technical report, 2018.

2. Новиков Олег. ЧТО ТАКОЕ ИНДУСТРИЯ 4.0? ЦИФРЫ И ФАКТЫ, 2015.

3. Thomas P. Roth, Yuyin Song, Martin J. Burns, Himanshu Neema, William Emfinger, and Janos Sztipanovits. Cyber-physical system development environment for energy applications, 2017.