Оглавление

Введение

««Сложные системы», «системность», «бизнес-процессы», «управление сложными системами», «модели» – все эти термины в настоящее время широко используются практически во всех сферах деятельности человека». Причиной этого является обобщение накопленного опыта и результатов в различных сферах человеческой деятельности и естественное желание найти и использовать некоторые общесистемные принципы и методы. Именно системность решаемых задач в перспективе должна стать той базой, которая позволит исследователю работать с любой сложной системой, независимо от ее физической сущности. Именно модели и моделирование систем является тем инструментом, которое обеспечивает эту возможность.

Имитационное моделирование (ИМ) на ЭВМ находит широкое применение при исследовании и управлении сложными дискретными системами (СДС) и процессами в них. К таким системам можно отнести экономические и производственные объекты, морские порты, аэропорты, комплексы перекачки нефти и газа, программное обеспечение сложных систем управления, вычислительные сети и многие другие. Широкое использование ИМ объясняется сложностью (а иногда и невозможностью) применения строгих методов оптимизации, которая обусловлена размерностью решаемых задач и неформализуемостью сложных систем. Так выделяют, например, следующие проблемы в исследовании операций, которые не могут быть решены сейчас и в обозримом будущем без ИМ:

1. Формирование инвестиционной политики при перспективном планировании.
2. Выбор средств обслуживания (или оборудования) при текущем планировании.
3. Разработка планов с обратной информационной связью и операционных предписаний.

Эти классы задач определяются тем, что при их решении необходимо одновременно учитывать факторы неопределенности, динамическую взаимную обусловленность текущих решений и последующих событий, комплексную

взаимозависимость между управляемыми переменными исследуемой системы, а часто и строго дискретную и четко определенную последовательность интервалов времени. Указанные особенности свойственны всем сложным системам.

Проведение имитационного эксперимента позволяет:

1. Сделать выводы о поведении СДС и ее особенностях:
   - без ее построения, если это проектируемая система;
   - без вмешательства в ее функционирование, если это действующая система, проведение экспериментов над которой или слишком дорого, или небезопасно;
   - без ее разрушения, если цель эксперимента состоит в определении пределов воздействия на систему.
2. Синтезировать и исследовать стратегии управления.
3. Прогнозировать и планировать функционирование системы в будущем.
4. Обучать и тренировать управленческий персонал и т.д.

ИМ является эффективным, но и не лишенным недостатков, методом. Трудности использования ИМ, связаны с обеспечением адекватности описания системы, интерпретацией результатов, обеспечением стохастической сходимости процесса моделирования, решением проблемы размерности и т.п. К проблемам применения ИМ следует отнести также и большую трудоемкость данного метода.

Интеллектуальное ИМ, характеризующееся возможностью использования методов искусственного интеллекта и, прежде всего, знаний, при принятии решений в процессе имитации, при управлении имитационным экспериментом, при реализации интерфейса пользователя, создании информационных банков ИМ, снимает часть проблем использования ИМ.

1.Предпроектное исследование.

1.1. Основные подходы к построению ИМ.

Системы имитационного моделирования СДС в зависимости от способов представления процессов, происходящих в моделируемом объекте, могут быть дискретными и непрерывными, пошаговыми и событийными, детерминированными и статистическими, стационарными и нестационарными.

Рассмотрим основные моменты этапа создания ИМ. Чтобы описать функционирование СДС надо описать интересующие нас события и действия, после чего создать алфавит, то есть дать каждому из них уникальное имя. Этот алфавит определяется как природой рассматриваемой СДС, так и целями ее анализа. Следовательно, выбор алфавита событий СДС приводит к ее упрощению – не рассматриваются многие ее свойства и действия не представляющие интерес для исследователя.

Событие СДС происходит мгновенно, то есть это некоторое действие с нулевой длительностью. Действие, требующее для своей реализации определенного времени, имеет собственное имя и связано с двумя событиями – начала и окончания. Длительность действия зависит от многих причин, среди которых время его начала, используемые ресурсы СДС, характеристики управления, влияние случайных факторов и т.д. В течение времени протекания действия в СДС могут возникнуть события, приводящие к преждевременному завершению действия. Последовательность действий образует процесс в СДС (Рис. 2.).

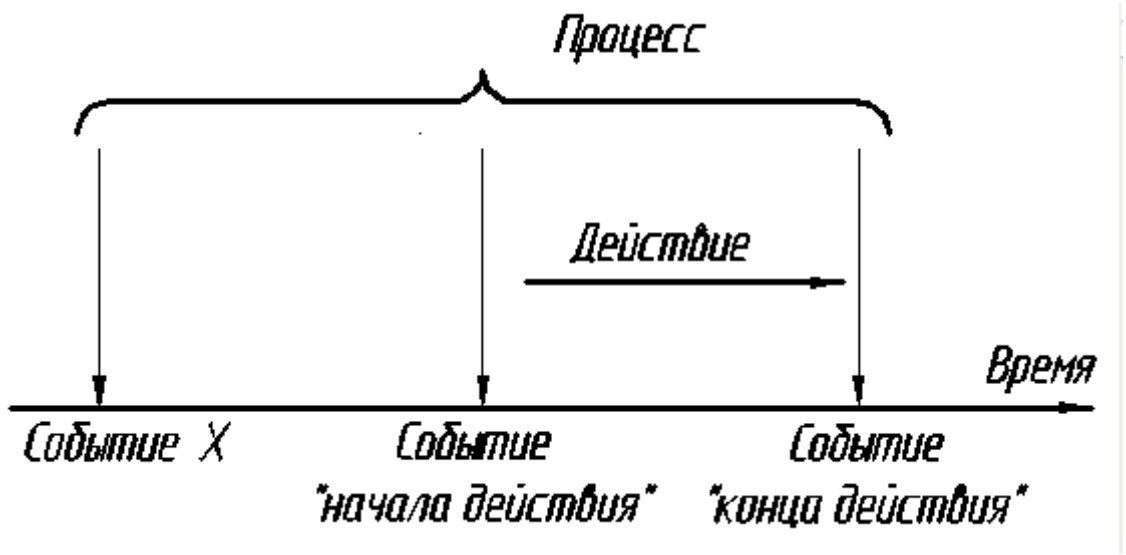


Рис. 1. Взаимосвязь между событиями, действием и процессом.

В соответствии с этим выделяют три альтернативных методологических подхода к построению ИМ: событийный, подход сканирования активностей и процессно-ориентированный.

1.2. Процесс имитации в РДО.

Для имитации работы модели в РДО реализованы два подхода: событийный и сканирования активностей.

Событийный подход.

При событийном подходе исследователь описывает события, которые могут изменять состояние системы, и определяет логические взаимосвязи между ними. Начальное состояние устанавливается путем задания значений переменным модели и параметров генераторам случайных чисел. Имитация происходит путем выбора из списка будущих событий ближайшего по времени и его выполнения. Выполнение события приводит к изменению состояния системы и генерации будущих событий, логически связанных с выполняемым. Эти события заносятся в список будущих событий и упорядочиваются в нем по времени наступления. Например, событие начала обработки детали на станке приводит к появлению в списке будущих событий события окончания обработки детали, которое должно наступить в момент времени равный текущему

времени плюс время, требуемое на обработку детали на станке. В событийных системах модельное время фиксируется только в моменты изменения состояний.

Рис. 2. Выполнение событий в ИМ.

Подход сканирования активностей.

При использовании подхода сканирования активностей разработчик описывает все действия, в которых принимают участие элементы системы, и задает условия, определяющие начало и завершение действий. После каждого продвижения имитационного времени условия всех возможных действий проверяются и если условие выполняется, то происходит имитация соответствующего действия. Выполнение действия приводит к изменению состояния системы и возможности выполнения новых действий. Например, для начала действия обработка детали на станке необходимо наличие свободной детали и наличие свободного станка. Если хотя бы одно из этих условий не выполнено, действие не начинается.

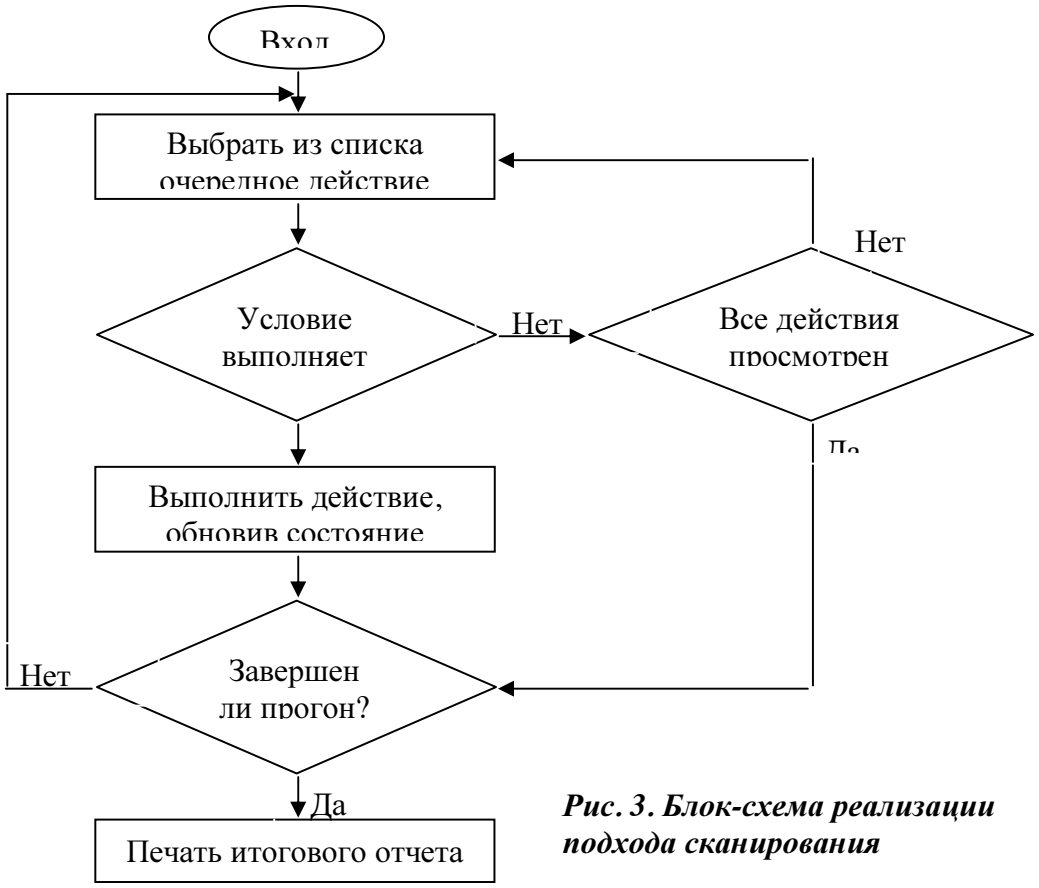


Рис. 3. Блок-схема реализации подхода сканирования

1.3. Основные положения языка РДО.

В основе системы РДО – «Ресурсы, Действия, Операции» – лежат следующие положения:

- Все элементы сложной дискретной системы (СДС) представлены как ресурсы, описываемые некоторыми параметрами.
- Состояние ресурса определяется вектором значений всех его параметров; состояние СДС – значением всех параметров всех ресурсов.
- Процесс, протекающий в СДС, описывается как последовательность целенаправленных действий и нерегулярных событий, изменяющих определенным

образом состояния ресурсов; действия ограничены во времени двумя событиями: событиями начала и конца.

- Нерегулярные события описывают изменение состояния СДС, непредсказуемые в рамках продукционной модели системы (влияние внешних по отношению к СДС факторов либо факторов, внутренних по отношению к ресурсам СДС). Моменты наступления нерегулярных событий случайны.
- Действия описываются операциями, которые представляют собой модифицированные продукционные правила, учитывающие временные связи. Операция описывает предусловия, которым должно удовлетворять состояние участвующих в операции ресурсов, и правила изменения ресурсов в начале и конце соответствующего действия.

При выполнении работ, связанных с созданием и использованием ИМ в среде РДО, пользователь оперирует следующими основными понятиями:

Модель - совокупность объектов РДО-языка, описывающих какой-то реальный объект, собираемые в процессе имитации показатели, кадры анимации и графические элементы, используемые при анимации, результаты трассировки.

Прогон - это единая неделимая точка имитационного эксперимента. Он характеризуется совокупностью объектов, представляющих собой исходные данные и результаты, полученные при запуске имитатора с этими исходными данными.

Проект - один или более прогонов, объединенных какой-либо общей целью. Например, это может быть совокупность прогонов, которые направлены на исследование одного конкретного объекта или выполнение одного контракта на имитационные исследования по одному или нескольким объектам.

Объект - совокупность информации, предназначенной для определенных целей и имеющая смысл для имитационной программы. Состав объектов обусловлен РДО-методом, определяющим парадигму представления СДС на языке РДО.

Объектами исходных данных являются:

- типы ресурсов (с расширением .rtp);

- ресурсы (с расширением .rss);
- образцы операций (с расширением .pat);
- операции (с расширением .opr);
- точки принятия решений (с расширением .dpt);
- константы, функции и последовательности (с расширением .fun);
- кадры анимации (с расширением .frm);
- требуемая статистика (с расширением .pmd);
- прогон (с расширением .smr).

Объекты, создаваемые РДО-имитатором при выполнении прогона:

- результаты (с расширением .pmv);
- трассировка (с расширением .trc).

1.4. Постановка задачи.

Основная идея бакалаврской работы – добавление в механизм логического вывода РДО иерархии логик. Под логикой, в данном контексте, понимается такая сущность системы, которая может объединять в себе простые атомарные активности и имеет свой алгоритм работы. Т.е. логиками являются точки принятия решений всех видов (some, prior, free и search).

Другими словами, в РДО должна появиться возможность встраивать точки принятия решений внутрь дригих точек принятия решений.

Сейчас точки принятия решений могут содержать лишь активности.

Это накладывает определенные ограничения на процесс разработки моделей на языке РДО. Для демонстрации этого недостатка рассмотрим модель работы почтового отделения связи. Прием клиентов может вестись в 5-ти окнах. Каждое из них работает по своему расписанию. Каждое из окон предоставляет клиентам весь спектр услуг. Емкость очереди в каждое окно ограничена максимальным значением. Все клиенты, которые не помещаются в очередь, уходят без обслуживания. Все клиенты, которые стоят в очереди

на момент окончания рабочего дня также покидают почту не обслужившись. Почта работает с 9 часов утра до 7 вечера.

Модель данной СМО на языке РДО представлена в Приложении 2.

Обратим внимание на описание БЗ модели на закладке DPT:

```
$Decision_point приход_клиентов : some
$Condition Отделение.Состояние = Рабочий_День
$Activities
      Еще_Один_Пришел     : Приход_Клиента_На_Почту
$End

$Decision_point размещение_клиентов : some
$Condition Отделение.Состояние = Рабочий_День
$Activities
      Какое_Окно          : Определить_Окно
      Потерянного_Вон     : Ликвидирование
      В_Очередь_Его       : Приход_Клиента_В_Очередь
$End

$Decision_point обслуживание_клиентов : some
$Condition Отделение.Состояние = Рабочий_День
$Activities
      Наконец_То_В_Окно1 : Обслуживание_Первого 1
      Наконец_То_В_Окно2 : Обслуживание_Первого 2
      Наконец_То_В_Окно3 : Обслуживание_Первого 3
      Наконец_То_В_Окно4 : Обслуживание_Первого 4
      Наконец_То_В_Окно5 : Обслуживание_Первого 5
$End

$Decision_point продвижение_очередей : some
$Condition Отделение.Состояние = Рабочий_День
$Activities
      Вперед1             : Сдвинуть_Очередь 1
      Вперед2             : Сдвинуть_Очередь 2
      Вперед3             : Сдвинуть_Очередь 3
      Вперед4             : Сдвинуть_Очередь 4
      Вперед5             : Сдвинуть_Очередь 5
$End

$Decision_point уход_клиентов : some
$Condition Отделение.Состояние = Рабочий_День
$Activities
      От_Окна1            : Уход_Клиента 1
      От_Окна2            : Уход_Клиента 2
      От_Окна3            : Уход_Клиента 3
      От_Окна4            : Уход_Клиента 4
      От_Окна5            : Уход_Клиента 5
$End

$Decision_point открытие_закрытие_окна : some
$Condition Отделение.Состояние = Рабочий_День
$Activities
      Я_Работаю           : Открыть_Окно
```

```
      Я_Закрылось          : Закрыть_Окно
$End

$Decision_point конец_дня : some
$Condition Отделение.Состояние = Конец_Дня
$Activities
      Поехали_Снова        : Переход_К_Новому_Дню
$End

$Decision_point уход_клиентов_в_конце_дня : some
$Condition Отделение.Состояние = Конец_Дня
$Activities
      Лишних_Вон1          : Уничтожение 1
      Лишних_Вон2          : Уничтожение 2
      Лишних_Вон3          : Уничтожение 3
      Лишних_Вон4          : Уничтожение 4
      Лишних_Вон5          : Уничтожение 5
$End

$Decision_point служебные : some
$Activities
      Часы_На_Экран        : Соответствие_Времени
      Поехали              : Функционирование
$End
```

При разработке данной имитационной модели работа почты была разделена на 2 этапа: рабочий день и окончание рабочего дня. Технически это реализовано с помощью условия запуска точек принятия решений. На первом этапе происходит поступление клиентов в систему, их размещение в очереди либо отказ в обслуживании, ожидание, обслуживание, уход. Также во время рабочего дня некоторые окна закрываются на перерыв и снова открываются. В свою очередь, после окончания рабочего дня на почту перестают приходить новые клиенты, все клиенты из очереди удаляются не обслуженными, клиенты, чьи заявки уже выполняются заканчивают свое обслуживание и уходят. После того, как на почте не остается клиентов, производятся некоторые технические процедуры по завершению текущего рабочего дня и подготовке к следующему. После этого может начинаться следующий день. Кроме этого также можно выделить некоторые служебные знания, присущие имитационным моделям, отображение времени, его синхронизация, переход между описанными выше этапами функционирования почты.

Все это говорит о иерархических знаниях, присущих описанию этой системы массового обслуживания. Но текущая версия РДО не позволяет это записывать в явном виде.

Таким образом, целью бакалаврской работы является разработка логического вывода системы имитационного моделирования РДО, основанного на иерархических логиках.

2.Концептуальный этап проектирования.

2.1.Анализ альтернативных концепций решения задачи.

Существует несколько альтернативных способов решения поставленной задачи, отличающихся простотой использования, наглядностью, гибкостью, предоставляемыми возможностями и, конечно, сложностью реализации. Поэтому необходимо провести их сравнительный анализ с целью выбора одной из альтернативных концепций решения задачи, которая позволит создать новый инструмент удобным, выразительным и мощным в сроки, отведенные для бакалаврской работы.

Первый вариант решения задачи - помещение полного описания одной (дочерней) точки принятия решений внутрь другой (родительской). Это не вносит никаких изменений в набор ключевых слов системы, и кажется весьма простым, интуитивно понятным решением с точки зрения разработчика моделей. Его основной недостаток заключается в большом объеме плохо структурированного текста, что может сказаться негативно на удобстве написания и сопровождения моделей при недостаточно строгом форматировании их кода.

Второй вариант - использование «ссылок» из дочерних точек на родительские. Т.е. в заголовке точки принятия решений должна появиться новая строка с именем той точки, активностью которой должна стать текущая. Недостатком этого метода является наличие ограничений на порядок следования точек: родительская точка должна быть описана раньше, чем дочерняя. Однако такое решение задачи не вносит каких-либо серьезных изменений в стилистику языка РДО и является весьма лаконичным.

Третьй способ - заведение в БЗ точки в виде активности. Т.е. необходимо добавить в систему новый тип образца активности - образца точки принятия решений. Этот метод является функциональнее других багодаря возможности многократного использования одного образца и передачи ему параметров. Частным случаем этих параметров может быть условие запуска или приоритет активности-точки, которые могут быть переданы от родительской точки дочерней, либо оставлены без изменений (эти значения у дочерней точки фактически являются значениями по умолчанию).

Последний из рассотренных вариантов, без сомнения, является наиболее мощным и гибким инструментом, но его разработка связана с большим числом изменений, которые необходимо внести в РДО, и, соответственно, с наличием временного ресурса большего, чем предоставляется на выполнение бакалаврской работы. Поэтому был сделан выбор в пользу второй концепции решения поставленной задачи.

2.2.Диаграмма компонентов.

Система имитационного моделирования РДО безусловно является сложной и статически, и динамически. На это указывает сложная иерархическая структура системы со множеством различных связей между компонентами и ее сложное поведение во времени.

Ярко выраженная иерархическая структура и модульность системы определяют направление изучения системы сверху вниз. Т.е. мне необходимо применять принцип декомпозиции нужных модулей до тех пор, пока не будет достигнут уровень абстракции, представление на котором нужных объектов не нуждается в дальнейшей детализации для решения данной задачи.

Для отображения зависимости между компонентами системы РДО и выделения среди них модернизируемых служит соответствующая диаграмма в нотации UML.

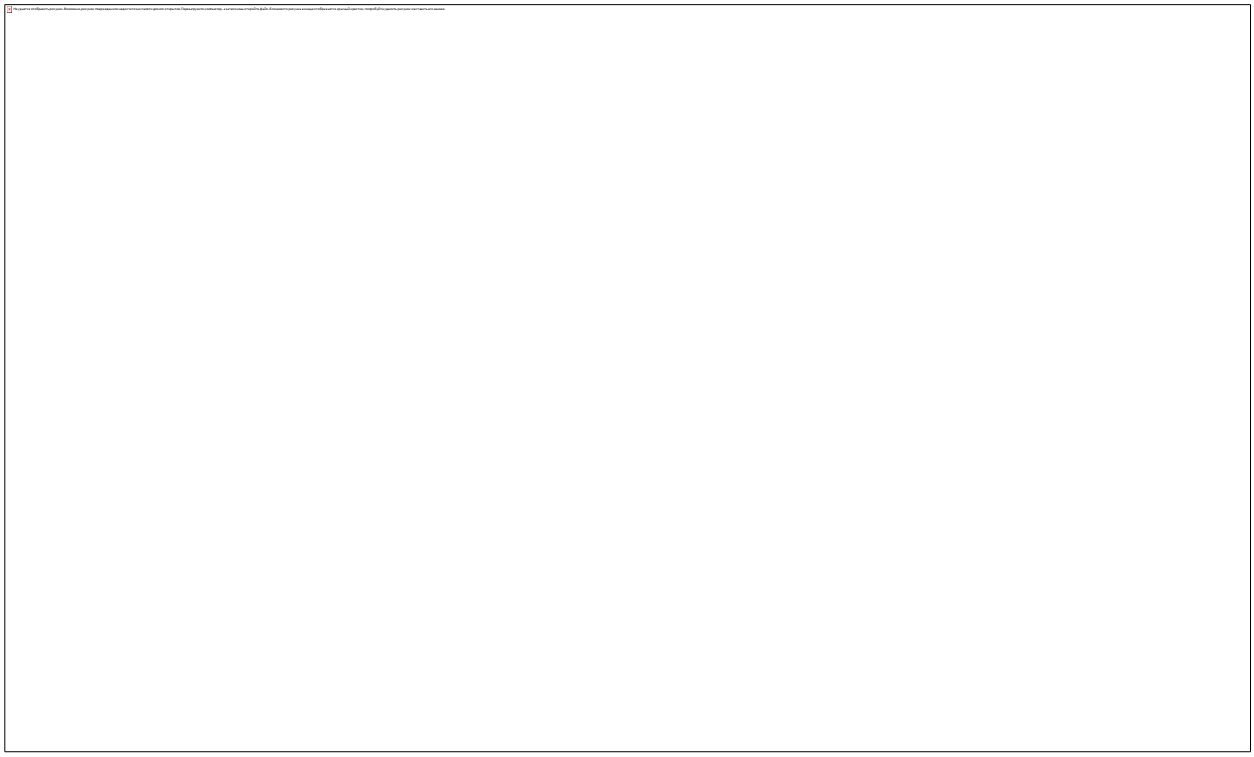


Рис. 4. Упращенная диаграмма компонентов.

Базовый функционал представленных на диаграмме компонентов:

rdo_kernel реализует ядровые функции системы. Не изменяется при разработке системы.

RAO-studio.exe реализует графический интерфейс пользователя. Не изменяется при разработки системы.

rdo_repository реализует управление потоками данных внутри системы и отвечает за хранение и получение информации о модели. Не изменяется при разработке системы.

rdo_mbuilder реализует функционал, используемый для программного управления типами ресурсов и ресурсами модели. Не изменяется при разработке системы.

rdo_simulator управляет процессом моделирования на всех его этапах. Он осуществляет координацию и управление компонентами rdo_runtime и rdo_parser. Не изменяется при разработке системы.

rdo_parser производит лексический и синтаксический разбор исходных текстов модели, написанной на языке РДО. Модернизируется при разработке системы.

rdo_runtime отвечает за непосредственное выполнение модели, управление базой данных и базой знаний. Модернизируется при разработке системы.

Объекты компонента rdo_runtime инициализуруются при разборе исходного текста модели компонентом rdo_parser. Например, конструктор rdoParse::RDODPTSome::RDODPTSome содержит следующее выражение:

```
m_rt_logic = new rdoRuntime::RDODPTSome( parser()->runtime() );
```

которое выделяет место в свободной памяти и инициализирует объект rdoRuntime::RDODPTSome, учавствующий в дальнейшем процессе имитации.

В дальнейшем компоненты rdo_parser и rdo_runtime описываются более детально.

2.2.Структура логического вывода РДО.

Логический вывод системы РДО представляет собой алгоритм, который определяет какое событие в моделируемой системе должно произойти следующим в процессе имитации работы системы.

Во время имитации работы модели в системе существует одна МЕТА-логика. Она является контейнером для хранения разных логик. Сами логики являются одновременно и контейнерами, в которых хранятся различные атомарные операции (например, нерегулярные события и правила) и атомарной (базовай) операцией. Т.е. данная архитектура БЗ РДО уже позволяет включать логики внутрь логик и в рамках данного проекта нужно создать инструмент, который будет встраивать логики модели в нужные места существующего дерева БЗ. Это позволит создавать иерархические БЗ.

Поиск активности, которая должна быть запущена следующей, начинается с обращения класса RDOSimulator к своему атрибуту m_logics, в котором хранится описанная выше МЕТА-логика. Далее от корня дерева к листьям распространяется волна вызовов метода onCheckCondition(). Т.е. onCheckCondition() вызывается у МЕТА-логики, затем циклически у ее логик, и наконец, циклически проверяются все атомарные операции каждой логики. Как только найдена активность, которая может быть выполнена, происходит ее кэширование (запоминание) внутри логики и кэширование самой логики внутри МЕТА-логики. После этого управление снова передается в RDOSimulator и найденная активность выполняется. Т.е. существующий механизм будет правильно обрабатывать многоуровневые иерархические деревья логик.

2.3.Техническое задание.

2.3.1.Общие сведения.

В системе РДО разрабатывается новый инструмент построения иерархической базы знаний в виде многоуровневых деревьев логик. Основной разработчик РДО – кафедра РК-9, МГТУ им. Н.Э. Баумана.

2.3.2.Назначение и цели развития системы.

Основная цель данного курсового проекта – разработать механизм логического вывода в системе имитационного моделирования РДО на основе иерархических логик.

2.3.3.Характеристики объекта автоматизации.

РДО – язык имитационного моделирования, включающий все три основные подхода описания дискретных систем: процессный, событийный и сканирования активностей.

2.3.4.Требования к системе.

При описании точки принятия решения пользователь может после имени точки указать родительскую точку - слово “$Parent” и имя описанной ранее точки типа Some или Prior.

Т.е. с учетом возможности описания приоритетов точек принятия решений, модель производственного участка должна иметь вид:

```
$Decision_point main : some
$Activities
$End

$Decision_point рабочий_день : some trace
$Parent main
$Condition Отделение.Состояние = Рабочий_День
$Activities
$End

$Decision_point приход_клиентов : some
$Parent рабочий_день
$Activities
      Еще_Один_Пришел     : Приход_Клиента_На_Почту
$End

$Decision_point размещение_клиентов : some
$Parent рабочий_день
$Activities
      Какое_Окно          : Определить_Окно
      Потерянного_Вон     : Ликвидирование
      В_Очередь_Его       : Приход_Клиента_В_Очередь
$End

$Decision_point обслуживание_клиентов : some
$Parent рабочий_день
$Activities
      Наконец_То_В_Окно1 : Обслуживание_Первого 1
      Наконец_То_В_Окно2 : Обслуживание_Первого 2
      Наконец_То_В_Окно3 : Обслуживание_Первого 3
      Наконец_То_В_Окно4 : Обслуживание_Первого 4
      Наконец_То_В_Окно5 : Обслуживание_Первого 5
$End

$Decision_point продвижение_очередей : some
$Parent рабочий_день
$Activities
      Вперед1             : Сдвинуть_Очередь 1
      Вперед2             : Сдвинуть_Очередь 2
      Вперед3             : Сдвинуть_Очередь 3
      Вперед4             : Сдвинуть_Очередь 4
      Вперед5             : Сдвинуть_Очередь 5
$End

$Decision_point уход_клиентов : some
$Parent рабочий_день
$Activities
      От_Окна1            : Уход_Клиента 1
      От_Окна2            : Уход_Клиента 2
      От_Окна3            : Уход_Клиента 3
      От_Окна4            : Уход_Клиента 4
      От_Окна5            : Уход_Клиента 5
$End
```

```
$Decision_point открытие_закрытие_окна : some
$Parent рабочий_день
$Activities
      Я_Работаю           : Открыть_Окно
      Я_Закрылось         : Закрыть_Окно
$End

$Decision_point конец_дня : some
$Parent main
$Condition Отделение.Состояние = Конец_Дня
$Activities
      Поехали_Снова       : Переход_К_Новому_Дню
$End

$Decision_point уход_клиентов_в_конце_дня : some
$Parent конец_дня
$Activities
      Лишних_Вон1         : Уничтожение 1
      Лишних_Вон2         : Уничтожение 2
      Лишних_Вон3         : Уничтожение 3
      Лишних_Вон4         : Уничтожение 4
      Лишних_Вон5         : Уничтожение 5
$End

$Decision_point служебные : some
$Parent main
$Activities
      Часы_На_Экран       : Соответствие_Времени
      Поехали             : Функционирование
$End
```

3.Технический этап проектирования.

3.1.Разработка синтаксиса точки принятия решения.

Объект точек принятия решений имеет следующий формат:

<описание_точки_принятия_решений> { <описание_точки_принятия_решений> }

Описание точки принятия решений имеет следующий формат:

[<заголовок_точки_принятия_решений>] <блок_активностей>

У точки принятия решений заголовок может состоять лишь из описания приоритета точки (или в частном случае отсутствовать вовсе). В таком случае она называется свободной точкой принятия решений или блоком свободных активностей.

Заголовок точки принятия решений имеет следующий формат:

$Decision_point <имя_точки> : <тип_точки> [<признак_трассировки>]

$Parent <имя_родительской_точки>

$Condition <условие_активизации_точки>

$Priority <приоритета_точки>

[$Term_condition <терминальное_условие>

$Evaluate_by <оценка_стоимости_оставшегося_пути_на_графе>

$Compare_tops = <признак_сравнения_вершин>]

Имена точек принятия решений должны быть различными для всех точек принятия решений и не должны совпадать с ранее определенными именами.

Тип точки может быть одним из следующих: some, prior или search (подробнее см. в справке РДО).

Имя родительской точки - имя одной из описанных выше точек принятия решений типа Some или Prior.

Признак трассировки может быть одним из следующих: no_trace, trace_stat, trace_tops, trace_all (подробнее см. в справке РДО).

Условие активизации точки – это логическое выражение. Алгоритм обработки точки принятия решений активизируется только в том случае, если состояние системы удовлетворяет этому выражению.

Приоритет точки - арифметическое выражение целого или вещественного типа данных, ограниченное диапазоном [0; 1].

Терминальное условие поиска, эвристическая оценочная функция стоимости оставшегося пути до целевой вершины и признак трассировки вершин записывается только для точек типа search (подробнее см. в справке РДО).

Блок активностей имеет следующий формат:

$Activities

<описание_активности> {<описание_активности>}

$End

Описание каждой активности подробно описано в справке РДО.

3.2.Разработка архитектуры компонента rdo_parser.

Для возможности обработки новой конструкции в коде модели требуют изменений лексический и синтаксический анализаторы РДО.

В пространстве имен rdoParse у конструкторов точек принятия решений должен появиться дополнительный параметр - указатель на родительскую точку. При конструировании точки этот параметр должен передаваться из rdoParse в rdoRuntime.

3.3.Разработка архитектуры компонента rdo_runtime.

В пространстве имен rdoRuntime указатель на родительскую точку должен появиться у базового класса для всех логик - RDOLogic.

Конструктора всех логик должны сохранять передаваемый им из rdoParse указатель на родительскую точку.

После создания точки происходит ее инициализация, в ходе которой точка должна быть «привязана» к нужной ветви существующего дерева БЗ модели.

Если rdoParse не передал в rdoRuntime указатель на родительскую точку, то в качестве нее должна быть использована META-логика.

4.Рабочий этап проектирования.

4.1.Синтаксический анализ приоритета логик.

Для реализации в среде имитационного моделирования нового инструмента разработанного на концептуальном и техническом этапах проектирования, в первую очередь необходимо добавить новые термальные символы в лексический анализатор РДО и нетермальные символы в грамматический анализатор.

В лексическом анализаторе (flex) я добавил новый токен RDO_Parent, который может быть записан двумя разными способами (с заглавной и строчной буквы):

```
$Parent           return(RDO_Parent);

$parent           return(RDO_Parent);
```

Этот токен необходимо также добавить в генератор синтаксического анализатора (bison):

```
%token RDO_ Parent                                                  377
```

Далее нужно добавить описание приоритета точки:

```
dpt_some_parent:
        /* empty */
        {
                $$ = 0;
        }
        | RDO_Parent RDO_IDENTIF
        {
                $$ = $2;
        }
        | RDO_Parent error
        {
                PARSER->error( @1, "Ошибка в имени родительской точки" );
        };

dpt_some_begin:
        RDO_Decision_point RDO_IDENTIF_COLON RDO_some dpt_some_trace dpt_some_parent
        {
                RDOValue* name        = reinterpret_cast<RDOValue*>($2);
                RDOValue* parent_name = reinterpret_cast<RDOValue*>($5);
                if ( parent_name != 0 )
                {
                        const RDODPTPrior*  parentDPTPrior  = PARSER->findDPTPrior(  parent_name-
>value().getIdentificator() );
                        const RDODPTSearch* parentDPTSearch = PARSER->findDPTSearch( parent_name-
>value().getIdentificator() );
                        const RDODPTSome*   parentDPTSome   = PARSER->findDPTSome(   parent_name-
>value().getIdentificator() );
                        if ( parentDPTPrior == NULL && parentDPTSearch == NULL && parentDPTSome ==
NULL )
                        {
```

```
                        PARSER->error( @1, rdo::format("Не найдена родительская точка %s",
parent_name->value().getIdentificator().c_str()) );
                    }
                    if ( parentDPTSearch  != NULL )
                    {
                        PARSER->error( @5, @1, "Точка принятия решений типа search может
содержать лишь активности типа rule и не может быть указана в качестве родительской точки" );
                    }
                    if ( parentDPTPrior  != NULL )
                    {
                        LPILogic parent = parentDPTPrior->getLogic();
                        $$ = (int)new RDODPTSome( PARSER, name->src_info(), parent );
                    }
                    if ( parentDPTSome   != NULL )
                    {
                        LPILogic parent = parentDPTSome->getLogic();
                        $$ = (int)new RDODPTSome( PARSER, name->src_info(), parent );
                    }
                }
                if ( parent_name == 0 )
                {
                    $$ = (int)new RDODPTSome( PARSER, name->src_info() );
                }
        };
```

Из этого кода можно сделать вывод, что точка не обязательно должна иметь родительскую точку. Это сделано для поддержки уже существующих моделей, написанных на РДО. Если пользователь опустит родительскую точку, то для создания новой точки будет вызван специальный конструктор, который сам укажет в качестве родительской логики - META-логику.

Если родительская точка до сих пор была не объявлена, то пользователь получит сообщение «Не найдена родительская точка». Т.е. это также произойдет, если родительская точка была описана ниже.

Если в качестве родительской указана точка типа Search, то пользователь также получит сообщение об ошибке «Точка типа Search не может быть указана в качестве родительской точки».

Подробное описание всех точек принятия решений приведено в Приложении 1.

4.2.Изменения в пространстве имен rdoParse.

Объявление класса RDODPTPrior

```
class RDODPTPrior: public RDOLogicActivity<rdoRuntime::RDODPTPrior, RDODPTPriorActivity>
{
public:
        RDODPTPrior( RDOParser* _parser, const RDOParserSrcInfo& _src_info, LPILogic _parent =
NULL  );

        LPILogic getLogic() const                          { return m_rt_logic;     }
```

```
	RDOFUNLogic* getConditon() const                  { return m_conditon;     }
	void setCondition( RDOFUNLogic* conditon = NULL ) { m_conditon = conditon; }

	void end();

private:
	RDOFUNLogic* m_conditon;
};
```

LPILogic getLogic( ) - метод класса RDODPTPrior, возвращает указатель на логику. Этот метод используется для получения указателя на родительскую точку.

Объявление класса RDODPTSome

```
class RDODPTSome: public RDOLogicActivity<rdoRuntime::RDODPTSome, RDODPTSomeActivity>
{
public:
	RDODPTSome( RDOParser* _parser, const RDOParserSrcInfo& _src_info, LPILogic _parent = NULL
);

	LPILogic getLogic() const                         { return m_rt_logic;     }
	RDOFUNLogic* getConditon() const                  { return m_conditon;     }
	void setCondition( RDOFUNLogic* conditon = NULL ) { m_conditon = conditon; }

	void end();

private:
	RDOFUNLogic* m_conditon;
};
```

LPILogic getLogic( ) - метод класса RDODPTSome, возвращает указатель на логику. Этот метод используется для получения указателя на родительскую точку.

Определение конструктора класса RDODPTActivityHotKey

```
RDODPTActivityHotKey::RDODPTActivityHotKey( LPIBaseOperationContainer dpt, const RDOParserObject*
parent, const RDOParserSrcInfo& _src_info, const RDOParserSrcInfo& _pattern_src_info ):
	RDODPTActivity( parent, _src_info, _pattern_src_info )
{
	switch ( pattern()->getType() )
	{
		case RDOPATPattern::PT_IE:
		{
			m_activity = static_cast<rdoRuntime::RDOPatternIrregEvent*>(pattern()-
>getPatRuntime())->createActivity( dpt, parser()->runtime(), name() );
			break;
		}
		case RDOPATPattern::PT_Rule:
		{
			m_activity = static_cast<rdoRuntime::RDOPatternRule*>(pattern()-
>getPatRuntime())->createActivity( dpt, parser()->runtime(), name() );
			break;
		}
		case RDOPATPattern::PT_Operation:
		{
			m_activity = static_cast<rdoRuntime::RDOPatternOperation*>(pattern()-
>getPatRuntime())->createActivity( dpt, parser()->runtime(), name() );
			break;
		}
		case RDOPATPattern::PT_Keyboard:
		{
```

```
                        m_activity = static_cast<rdoRuntime::RDOPatternKeyboard*>(pattern()-
>getPatRuntime())->createActivity( dpt, parser()->runtime(), name() );
                        break;
                }
                default:
                {
                        parser()->error_push_only( src_info(), "Íåèçâåñòíûé òèï îáðàçöà" );
                        parser()->error_push_only( pattern()->src_info(), "Ñì. îáðàçåö" );
                        parser()->error_push_done();
                }
        }
}
```

Этот конструктор создает активность с помощью метода createActivity(), которому передает в качестве параметра указатель на логику, в которой находится данная активность. С помощью этого указателя активность привязывается к нужной ветви БЗ модели.

4.3.Изменения в пространстве имен rdoRuntime.

Объявление класса RDOLogic

```
template <class Order>
class RDOLogic: public IBaseOperation, public IBaseOperationContainer, public ILogic,
CAST_TO_UNKNOWN
{
QUERY_INTERFACE_BEGIN
        QUERY_INTERFACE(IBaseOperation)
        QUERY_INTERFACE(IBaseOperationContainer)
        QUERY_INTERFACE(ILogic)
QUERY_INTERFACE_END

public:
        typedef  BaseOperationList                 ChildList;
        typedef  BaseOperationList::iterator       Iterator;
        typedef  BaseOperationList::const_iterator CIterator;

protected:
        RDOLogic(PTR(RDOSimulator) sim = NULL, LPIBaseOperationContainer parent = NULL);
        virtual ~RDOLogic();

        DECLARE_IBaseOperationContainer;

        PTR(RDOCalc)               m_condition;
        rbool                      m_lastCondition;
        ChildList                  m_childList;
        LPIBaseOperation           m_first;
        LPIBaseOperationContainer  m_parent;

private:
        rbool checkSelfCondition(PTR(RDOSimulator) sim);
        void  start             (PTR(RDOSimulator) sim);
        void  stop              (PTR(RDOSimulator) sim);

        DECLARE_IBaseOperation;
        DECLARE_ILogic;
};
```

В атрибуте m_parent класса RDOLogic хранится указатель на родительскую логику (точку).

Определение конструктора класса RDOLogic

```
template <class Order>
inline RDOLogic<Order>::RDOLogic(PTR(RDOSimulator) sim, LPIBaseOperationContainer parent)
	: m_condition    (NULL       )
	, m_lastCondition(false      )
	, m_first        (NULL       )
	, m_parent       (parent ? parent : (sim ? sim->m_metaLogic : NULL))
{}
```

Если конструктор RDOLogic был вызван с указателем на родительскую точку, то она сохранится атрибуте m_parent. Если конструктор RDOLogic был вызван без указателя на родительскую точку, но с указателем на симулятор, то в качестве родительской логике сохранится META-логика. Иначе, если конструктор RDOLogic был вызван без обоих параметров (это нужно для заведения META-логики), то в атрибут m_parent запишется нулевой указатель.

Определение функции-члена init() класса RDOLogic

```
template <class Order>
inline void RDOLogic<Order>::init(PTR(RDOSimulator) sim)
{
	if (sim)
		sim->appendLogic(rdo::UnknownPointer(this).query_cast<IBaseOperation>(), this->m_parent);
}
```

В этом методе происходит «привязывание» логики к дереву БЗ модели.

Определение функции-члена init() классов RDOPatternRule

```
LPIRule RDOPatternRule::createActivity(LPIBaseOperationContainer logic, PTR(RDORuntime) runtime, PTR(RDOCalc) condition, CREF(tstring) _oprName)
{
	LPIRule rule = F(RDORule)::create(runtime, this, traceable(), condition, _oprName);
	runtime->addRuntimeRule(logic, rule);
	return rule;
}

LPIOperation RDOPatternOperation::createActivity(LPIBaseOperationContainer parent, PTR(RDORuntime) runtime, CREF(tstring) _oprName)
{
	LPIOperation operation = F(RDOOperation)::create(runtime, this, traceable(), _oprName);
	runtime->addRuntimeOperation(parent, operation);
	return operation;
}

LPIIrregEvent RDOPatternIrregEvent::createActivity(LPIBaseOperationContainer parent, PTR(RDORuntime) runtime, CREF(tstring) oprName)
{
	LPIIrregEvent ie = F(RDOIrregEvent)::create(runtime, this, traceable(), oprName);
	runtime->addRuntimeIE(parent, ie);
	return ie;
}

LPIKeyboard RDOPatternKeyboard::createActivity(LPIBaseOperationContainer parent, PTR(RDORuntime) runtime, CREF(tstring) _oprName)
```

```
{
        LPIKeyboard keyboard = F(RDOKeyboard)::create(runtime, this, traceable(), _oprName);
        runtime->addRuntimeOperation(parent, keyboard);
        return keyboard;
}
```

В этих методах происходит «привязывание» активностей к дереву БЗ модели.

Заключение

В рамках данной квалификационной работы бакалавра были получены следующие результаты:

1) Проведено предпроектное исследование системы имитационного моделирования РДО и сформулированы предпосылки создания в системе инструмента для построения иерархической базы знаний модели.

2) На этапе концептуального проектирования системы проанализированы три варианта решения поставленной задачи и сделан выбор в пользу второго. С помощью диаграммы компонентов нотации UML укрупненно показано внутреннее устройство РДО и выделены те компоненты, которые потребуют внесения изменений в ходе этой работы. Разработана функциональная диаграмма (в нотации IDEF0) компиляции модели, на которой обозначены те функции системы, в которые будут внесены изменения.

3) На этапе технического проектирования разработан новый синтаксис точек принятия решений, который представлен на синтаксической диаграмме. С помощью диаграммы классов разработана архитектура новой системы. С помощью блок-схемы разработаны алгоритмы, реализующие в системе РДО механизм логического вывода на основе иерархических логик.

4) На этапе рабочего проектирования написан программный код для реализации спроектированных раннее алгоритмов работы и архитектуры компонентов rdo_parser и rdo_runtime системы РДО. Проведены отладка и тестирование новой системы, в ходе которых исправлялись найденные ошибки.

5) Для демонстрации новых возможностей системы модель, представленная на этапе постановки задачи, была переписана так, чтобы задействовать новый механизм логического вывода РДО. Результаты проведения имитационного исследования позволяют сделать вывод об адекватности нового логического вывода.

Поставленная цель работы достигнута в полном объеме.

Список использованных источников

1. RAO-Studio – Руководство пользователя, 2007 [http://rdo.rk9.bmstu.ru/forum/viewtopic.php?t=900].

2. Справка по языку РДО (в составе программы) [http://rdo.rk9.bmstu.ru/forum/viewforum.php?f=15].

3. Емельянов В.В., Ясиновский С.И. Имитационное моделирование систем: Учеб. пособие. – М.: Изд-во МГТУ им.Н.Э. Баумана, 2009. – 584 с.: ил. (Информатика в техническом университете).

4. Единая система программной документации. Техническое задание. Требования к содержанию и оформлению. ГОСТ 19.201-78.

5. Единая система программной документации. Схемы алгоритмов, программ, данных и систем. ГОСТ 19.701-90. Условные обозначения и правила выполнения.

6. Леоненков. Самоучитель по UML [http://khpi-iip.mipk.kharkiv.edu/library/case/leon/index.html].

7. Бьерн Страуструп. Язык моделирования C++. Специальное издание. Пер. с англ. – М.: ООО «Бином-пресс», 2007 г. – 1104 с.: ил.

Приложение 1. Полный синтаксический анализ точек принятия решений (rdodpt.y).

```
// --------------------------------------------------------------------------------
// ---------- DPT
// --------------------------------------------------------------------------------
dpt_main:
               | dpt_main dpt_search_end
               | dpt_main dpt_some_end
               | dpt_main dpt_prior_end
               | dpt_main dpt_free_end
               | dpt_main dpt_process_end
               | error
              {
                     PARSER->error( @1, "Ожидается описание точки или свободного блока
активностей" );
               };

// --------------------------------------------------------------------------------
// ---------- DPTSearch
// --------------------------------------------------------------------------------
dpt_search_trace:            /* empty */
                                          {
                                                 $$ =
rdoRuntime::RDODPTSearchTrace::DPT_no_trace;
                                          }
                                          | RDO_no_trace
                                          {
                                                 $$ =
rdoRuntime::RDODPTSearchTrace::DPT_no_trace;
                                          }
                                          | RDO_trace
                                          {
                                                 PARSER->error( @1, "Данный признак
трассировки не используется в точке типа search" );
                                          }
                                          | RDO_trace_stat
                                          {
                                                 $$ =
rdoRuntime::RDODPTSearchTrace::DPT_trace_stat;
                                          }
                                          | RDO_trace_tops
                                          {
                                                 $$ =
rdoRuntime::RDODPTSearchTrace::DPT_trace_tops;
                                          }
                                          | RDO_trace_all
                                          {
                                                 $$ =
rdoRuntime::RDODPTSearchTrace::DPT_trace_all;
                                          };

dpt_search_parent:           /* empty */
                                          {
                                                 $$ = 0;
                                          }
                                          | RDO_Parent RDO_IDENTIF
                                          {
                                                 $$ = $2;
                                          }
                                          | RDO_Parent error
                                          {
                                                 PARSER->error( @1, "Ошибка в имени
родительской точки" );
                                          };

dpt_search_begin:            RDO_Decision_point RDO_IDENTIF_COLON RDO_search dpt_search_trace
dpt_search_parent
                                          {
                                                 RDOValue* name        =
reinterpret_cast<RDOValue*>($2);
```

```
                                                RDOValue* parent_name =
reinterpret_cast<RDOValue*>($5);
                                                if ( parent_name != 0 )
                                                {
                                                        const RDODPTPrior*  parentDPTPrior  =
PARSER->findDPTPrior(  parent_name->value().getIdentificator() );
                                                        const RDODPTSearch* parentDPTSearch =
PARSER->findDPTSearch( parent_name->value().getIdentificator() );
                                                        const RDODPTSome*   parentDPTSome   =
PARSER->findDPTSome(   parent_name->value().getIdentificator() );
                                                        if ( parentDPTPrior == NULL &&
parentDPTSearch == NULL && parentDPTSome == NULL )
                                                        {
                                                                PARSER->error( @1,
rdo::format("Не найдена родитеская точка %s", parent_name->value().getIdentificator().c_str()) );
                                                        }
                                                        if ( parentDPTSearch  != NULL )
                                                        {
                                                                PARSER->error( @1, "Точка
принятия решений типа search может содержать лишь активности типа rule и не может быть указана в
качестве родительской точки" );
                                                        }
                                                        if ( parentDPTPrior  != NULL )
                                                        {
                                                                LPILogic parent =
parentDPTPrior->getLogic();
                                                                $$ = (int)new RDODPTSearch(
PARSER, name->src_info(), *reinterpret_cast<rdoRuntime::RDODPTSearchTrace::DPT_TraceFlag*>(&$4),
parent );
                                                        }
                                                        if ( parentDPTSome   != NULL )
                                                        {
                                                                LPILogic parent =
parentDPTSome->getLogic();
                                                                $$ = (int)new RDODPTSearch(
PARSER, name->src_info(), *reinterpret_cast<rdoRuntime::RDODPTSearchTrace::DPT_TraceFlag*>(&$4),
parent );
                                                        }
                                                }
                                                if ( parent_name == 0 )
                                                {
                                                        $$ = (int)new RDODPTSearch( PARSER,
name->src_info(), *reinterpret_cast<rdoRuntime::RDODPTSearchTrace::DPT_TraceFlag*>(&$4) );
                                                }
                                        }
                                        | RDO_Decision_point RDO_IDENTIF_COLON error
                                        {
                                                PARSER->error( @2, @3, "Ожидается тип точки"
);
                                        }
                                        | RDO_Decision_point RDO_IDENTIF error
                                        {
                                                PARSER->error( @2, "Ожидается двоеточие" );
                                        }
                                        | RDO_Decision_point error
                                        {
                                                PARSER->error( @1, @2, "После ключевого слова
$Decision_point ожидается имя точки" );
                                        };

dpt_search_condition: dpt_search_begin RDO_Condition fun_logic
                                        {
                                                RDODPTSearch* dpt =
reinterpret_cast<RDODPTSearch*>($1);
                                                dpt->setCondition((RDOFUNLogic *)$3);
                                        }
                                        | dpt_search_begin RDO_Condition RDO_NoCheck
                                        {
                                                RDODPTSearch* dpt =
reinterpret_cast<RDODPTSearch*>($1);
                                                dpt->setCondition();
                                        }
                                        | dpt_search_begin RDO_Condition error
                                        {
```

```
                                                PARSER->error( @2, @3, "После ключевого слова
$Condition ожидается условие начала поиска (начальная вершина)" );
                                        }
                                        | dpt_search_begin error
                                        {
                                                PARSER->error( @2, "Ожидается ключевое слово
$Condition" );
                                        };

dpt_search_prior:         dpt_search_condition
                                        | dpt_search_condition RDO_Priority fun_arithm
                                        {
                                                if (!PARSER->getLastDPTSearch()->setPrior(
reinterpret_cast<RDOFUNArithm*>($3) ))
                                                {
                                                        PARSER->error(@3, _T("Точка принятия
решений пока не может иметь приоритет"));
                                                }
                                        }
                                        | dpt_search_condition RDO_Priority error
                                        {
                                                PARSER->error( @1, @2, "Ошибка описания
приоритета точки принятия решений" )
                                        }
                                        | dpt_search_condition error
                                        {
                                                PARSER->error( @1, @2, "Ожидается ключевое
слово $Priority" )
                                        };

dpt_search_term:              dpt_search_prior RDO_Term_condition fun_logic
                                        {
                                                RDODPTSearch* dpt =
reinterpret_cast<RDODPTSearch*>($1);
                                                dpt->setTermCondition((RDOFUNLogic *)$3);
                                        }
                                        | dpt_search_prior RDO_Term_condition RDO_NoCheck
                                        {
                                                RDODPTSearch* dpt =
reinterpret_cast<RDODPTSearch*>($1);
                                                dpt->setTermCondition();
                                        }
                                        | dpt_search_prior RDO_Term_condition error
                                        {
                                                PARSER->error( @2, @3, "После ключевого слова
$Term_condition ожидается условие остановки поиска (конечная вершина)" );
                                        }
                                        | dpt_search_prior error
                                        {
                                                PARSER->error( @2, "Ожидается ключевое слово
$Term_condition" );
                                        };

dpt_search_evaluate:  dpt_search_term RDO_Evaluate_by fun_arithm
                                        {
                                                RDODPTSearch* dpt =
reinterpret_cast<RDODPTSearch*>($1);
                                                dpt->setEvaluateBy((RDOFUNArithm *)$3);
                                        }
                                        | dpt_search_term RDO_Evaluate_by error
                                        {
                                                PARSER->error( @2, @3, "После ключевого слова
$Evaluate_by ожидается оценочная функция, например, 0 для поиска в ширину" );
                                        }
                                        | dpt_search_term error
                                        {
                                                PARSER->error( @2, "Ожидается ключевое слово
$Evaluate_by" );
                                        };

dp_searcht_compare:           dpt_search_evaluate RDO_Compare_tops '=' RDO_NO
                                        {
                                                RDODPTSearch* dpt =
reinterpret_cast<RDODPTSearch*>($1);
```

```
                                        dpt->setCompareTops(false);
                                    }
                                    | dpt_search_evaluate RDO_Compare_tops '=' RDO_YES
                                    {
                                        RDODPTSearch* dpt =
reinterpret_cast<RDODPTSearch*>($1);
                                        dpt->setCompareTops(true);
                                    }
                                    | dpt_search_evaluate RDO_Compare_tops '=' error
                                    {
                                        PARSER->error( @3, @4, "Ожидается режим
запоминания пройденных вершин (YES или NO)" );
                                    }
                                    | dpt_search_evaluate RDO_Compare_tops error
                                    {
                                        PARSER->error( @2, @3, "Ожидается знак
равенства" );
                                    }
                                    | dpt_search_evaluate error
                                    {
                                        PARSER->error( @2, "Ожидается ключевое слово
$Compare_tops" );
                                    };

dpt_search_descr_param:      /* empty */
                                    | dpt_search_descr_param  '*'
                                    {
                                        PARSER->getLastDPTSearch()-
>getLastActivity()->addParam( RDOValue(RDOParserSrcInfo(@2, "*")) )
                                    }
                                    | dpt_search_descr_param  RDO_INT_CONST
                                    {
                                        PARSER->getLastDPTSearch()-
>getLastActivity()->addParam( *reinterpret_cast<RDOValue*>($2) )
                                    }
                                    | dpt_search_descr_param  RDO_REAL_CONST
                                    {
                                        PARSER->getLastDPTSearch()-
>getLastActivity()->addParam( *reinterpret_cast<RDOValue*>($2) )
                                    }
                                    | dpt_search_descr_param  RDO_BOOL_CONST
                                    {
                                        PARSER->getLastDPTSearch()-
>getLastActivity()->addParam( *reinterpret_cast<RDOValue*>($2) )
                                    }
                                    | dpt_search_descr_param  RDO_STRING_CONST
                                    {
                                        PARSER->getLastDPTSearch()-
>getLastActivity()->addParam( *reinterpret_cast<RDOValue*>($2) )
                                    }
                                    | dpt_search_descr_param  RDO_IDENTIF
                                    {
                                        PARSER->getLastDPTSearch()-
>getLastActivity()->addParam( *reinterpret_cast<RDOValue*>($2) )
                                    };

dpt_search_descr_value:      RDO_value_before fun_arithm
                                    {
                                        RDODPTSearch* dpt = PARSER-
>getLastDPTSearch();
                                        dpt->getLastActivity()->setValue(
IDPTSearchActivity::vt_before, reinterpret_cast<RDOFUNArithm*>($2), @1 );
                                    }
                                    | RDO_value_after fun_arithm
                                    {
                                        RDODPTSearch* dpt = PARSER-
>getLastDPTSearch();
                                        dpt->getLastActivity()->setValue(
IDPTSearchActivity::vt_after, reinterpret_cast<RDOFUNArithm*>($2), @1 );
                                    }
                                    | RDO_value_before error
                                    {
                                        PARSER->error( @1, @2, "Ошибка в
арифметическом выражении" );
```

```
                                        }
                                        | RDO_value_after error
                                        {
                                                PARSER->error( @1, @2, "Ошибка в
арифметическом выражении" );
                                        };

dpt_search_name:              RDO_IDENTIF_COLON RDO_IDENTIF
                                        {
                                                RDODPTSearch* dpt   = PARSER-
>getLastDPTSearch();
                                                RDOValue* name    =
reinterpret_cast<RDOValue*>($1);
                                                RDOValue* pattern =
reinterpret_cast<RDOValue*>($2);
                                                $$ = (int)dpt->addNewActivity( name-
>src_info(), pattern->src_info() );
                                        }
                                        | RDO_IDENTIF_COLON error
                                        {
                                                PARSER->error( @1, @2, "Ожидается имя
образца" );
                                        }
                                        | RDO_IDENTIF
                                        {
                                                PARSER->error( @1, "Ожидается ':'" );
                                        }
                                        | error
                                        {
                                                PARSER->error( @1, "Ожидается имя активности"
);
                                        };

dpt_searcht_activity: /* empty */
                                        | dpt_searcht_activity dpt_search_name
dpt_search_descr_param dpt_search_descr_value
                                        {
                                                RDODPTSearchActivity* activity =
reinterpret_cast<RDODPTSearchActivity*>($2);
                                                activity->endParam( @3 );
                                        }
                                        | dpt_searcht_activity dpt_search_name
dpt_search_descr_param error
                                        {
                                                PARSER->error( @3, @4, "Ожидаются ключевые
слова value before или value after и стоимость применения правила" );
                                        };

dpt_search_header:            dp_searcht_compare RDO_Activities dpt_searcht_activity
                                        {
                                        }
                                        | dp_searcht_compare error
                                        {
                                                PARSER->error( @1, @2, "После режима
запоминания пройденных вершин ожидается ключевое слово $Activities" );
                                        };

dpt_search_end:                       dpt_search_header RDO_End
                                        {
                                                RDODPTSearch* dpt =
reinterpret_cast<RDODPTSearch*>($1);
                                                dpt->end();
                                        }
                                        | dpt_search_header
                                        {
                                                PARSER->error( @1, "Ожидается ключевое слово
$End" );
                                        };

// ----------------------------------------------------------------------
// ---------- DPTSome
// ----------------------------------------------------------------------
dpt_some_trace:                       /* empty */
                                        {
```

```
                                        $$ = 1;
                                    }
                                    | RDO_no_trace
                                    {
                                        $$ = 1;
                                    }
                                    | RDO_trace
                                    {
                                        $$ = 2;
                                    }
                                    | RDO_trace_stat
                                    {
                                        PARSER->error( @1, "Данный признак
трассировки не используется в точке типа some" );
                                    }
                                    | RDO_trace_tops
                                    {
                                        PARSER->error( @1, "Данный признак
трассировки не используется в точке типа some" );
                                    }
                                    | RDO_trace_all
                                    {
                                        PARSER->error( @1, "Данный признак
трассировки не используется в точке типа some" );
                                    };

dpt_some_parent:            /* empty */
                                    {
                                        $$ = 0;
                                    }
                                    | RDO_Parent RDO_IDENTIF
                                    {
                                        $$ = $2;
                                    }
                                    | RDO_Parent error
                                    {
                                        PARSER->error( @1, "Ошибка в имени
родительской точки" );
                                    };

dpt_some_begin:                     RDO_Decision_point RDO_IDENTIF_COLON RDO_some dpt_some_trace
dpt_some_parent
                                    {
                                        RDOValue* name        =
reinterpret_cast<RDOValue*>($2);
                                        RDOValue* parent_name =
reinterpret_cast<RDOValue*>($5);
                                        if ( parent_name != 0 )
                                        {
                                            const RDODPTPrior*  parentDPTPrior  =
PARSER->findDPTPrior(  parent_name->value().getIdentificator() );
                                            const RDODPTSearch* parentDPTSearch =
PARSER->findDPTSearch( parent_name->value().getIdentificator() );
                                            const RDODPTSome*   parentDPTSome   =
PARSER->findDPTSome(   parent_name->value().getIdentificator() );
                                            if ( parentDPTPrior == NULL &&
parentDPTSearch == NULL && parentDPTSome == NULL )
                                            {
                                                PARSER->error( @1,
rdo::format("Не найдена родитеская точка %s", parent_name->value().getIdentificator().c_str()) );
                                            }
                                            if ( parentDPTSearch  != NULL )
                                            {
                                                PARSER->error( @5, @1, "Точка
принятия решений типа search может содержать лишь активности типа rule и не может быть указана в
качестве родительской точки" );
                                            }
                                            if ( parentDPTPrior  != NULL )
                                            {
                                                LPILogic parent =
parentDPTPrior->getLogic();
                                                $$ = (int)new RDODPTSome(
PARSER, name->src_info(), parent );
                                            }
```

```
                                                        if ( parentDPTSome   != NULL )
                                                        {
                                                                LPILogic parent =
parentDPTSome->getLogic();
                                                                $$ = (int)new RDODPTSome(
PARSER, name->src_info(), parent );
                                                        }
                                                }
                                                if ( parent_name == 0 )
                                                {
                                                        $$ = (int)new RDODPTSome( PARSER,
name->src_info() );
                                                }
                                        };

dpt_some_condition:          dpt_some_begin RDO_Condition fun_logic
                                        {
                                                RDODPTSome* dpt =
reinterpret_cast<RDODPTSome*>($1);
                                                dpt->setCondition(
reinterpret_cast<RDOFUNLogic*>($3) );
                                        }
                                        | dpt_some_begin RDO_Condition RDO_NoCheck
                                        {
                                                RDODPTSome* dpt =
reinterpret_cast<RDODPTSome*>($1);
                                                dpt->setCondition();
                                        }
                                        | dpt_some_begin RDO_Condition error
                                        {
                                                PARSER->error( @2, @3, "После ключевого слова
$Condition ожидается условие запуска точки" );
                                        }
                                        | dpt_some_begin
                                        {
                                                RDODPTSome* dpt =
reinterpret_cast<RDODPTSome*>($1);
                                                dpt->setCondition();
                                        };

dpt_some_prior:               dpt_some_condition
                                        | dpt_some_condition RDO_Priority fun_arithm
                                        {
                                                if (!PARSER->getLastDPTSome()->setPrior(
reinterpret_cast<RDOFUNArithm*>($3) ))
                                                {
                                                        PARSER->error(@3, _T("Точка принятия
решений пока не может иметь приоритет"));
                                                }
                                        }
                                        | dpt_some_condition RDO_Priority error
                                        {
                                                PARSER->error( @1, @2, "Ошибка описания
приоритета точки принятия решений" )
                                        }
                                        | dpt_some_condition error
                                        {
                                                PARSER->error( @1, @2, "Ожидается ключевое
слово $Priority" )
                                        };

dpt_some_name:               RDO_IDENTIF_COLON RDO_IDENTIF
                                        {
                                                RDODPTSome* dpt   = PARSER->getLastDPTSome();
                                                RDOValue* name    =
reinterpret_cast<RDOValue*>($1);
                                                RDOValue* pattern =
reinterpret_cast<RDOValue*>($2);
                                                $$ = (int)dpt->addNewActivity( name-
>src_info(), pattern->src_info() );
                                        }
                                        | RDO_IDENTIF_COLON error
                                        {
```

```
                                                PARSER->error( @1, @2, "Ожидается имя
образца" );
                                        };

dpt_some_descr_param: /* empty */
                                        | dpt_some_descr_param  '*'
                                        {
                                                PARSER->getLastDPTSome()->getLastActivity()-
>addParam( RDOValue(RDOParserSrcInfo(@2, "*")) )
                                        }
                                        | dpt_some_descr_param  RDO_INT_CONST
                                        {
                                                PARSER->getLastDPTSome()->getLastActivity()-
>addParam( *reinterpret_cast<RDOValue*>($2) )
                                        }
                                        | dpt_some_descr_param  RDO_REAL_CONST
                                        {
                                                PARSER->getLastDPTSome()->getLastActivity()-
>addParam( *reinterpret_cast<RDOValue*>($2) )
                                        }
                                        | dpt_some_descr_param  RDO_BOOL_CONST
                                        {
                                                PARSER->getLastDPTSome()->getLastActivity()-
>addParam( *reinterpret_cast<RDOValue*>($2) )
                                        }
                                        | dpt_some_descr_param  RDO_STRING_CONST
                                        {
                                                PARSER->getLastDPTSome()->getLastActivity()-
>addParam( *reinterpret_cast<RDOValue*>($2) )
                                        }
                                        | dpt_some_descr_param  RDO_IDENTIF
                                        {
                                                PARSER->getLastDPTSome()->getLastActivity()-
>addParam( *reinterpret_cast<RDOValue*>($2) )
                                        }
                                        | dpt_some_descr_param error
                                        {
                                                PARSER->error( @1, @2, "Ошибка описания
параметра образца" )
                                        };

dpt_some_activity:          /* empty */
                                        | dpt_some_activity dpt_some_name
dpt_some_descr_param
                                        {
                                                RDODPTSomeActivity* activity =
reinterpret_cast<RDODPTSomeActivity*>($2);
                                                activity->endParam( @3 );
                                        };

dpt_some_header:            dpt_some_prior RDO_Activities dpt_some_activity
                                        {
                                        }
                                        | dpt_some_prior error
                                        {
                                                PARSER->error( @1, @2, "Ожидается ключевое
слово $Activities" );
                                        };

dpt_some_end:               dpt_some_header RDO_End
                                        {
                                                RDODPTSome* dpt =
reinterpret_cast<RDODPTSome*>($1);
                                                dpt->end();
                                        }
                                        | dpt_some_header
                                        {
                                                PARSER->error( @1, "Ожидается ключевое слово
$End" );
                                        };

// --------------------------------------------------------------------------
// ---------- DPT Prior
// --------------------------------------------------------------------------
```

```
dpt_prior_trace:              /* empty */ {
                                        $$ = 1;
                                    }
                                    | RDO_no_trace {
                                        $$ = 1;
                                    }
                                    | RDO_trace {
                                        $$ = 2;
                                    }
                                    | RDO_trace_stat {
                                        PARSER->error( @1, "Данный признак
трассировки не используется в точке типа prior" );
                                    }
                                    | RDO_trace_tops {
                                        PARSER->error( @1, "Данный признак
трассировки не используется в точке типа prior" );
                                    }
                                    | RDO_trace_all {
                                        PARSER->error( @1, "Данный признак
трассировки не используется в точке типа prior" );
                                    };

dpt_prior_parent:             /* empty */
                                    {
                                        $$ = 0;
                                    }
                                    | RDO_Parent RDO_IDENTIF
                                    {
                                        $$ = $2;
                                    }
                                    | RDO_Parent error
                                    {
                                        PARSER->error( @1, "Ошибка в имени
родительской точки" );
                                    };

dpt_prior_begin:              RDO_Decision_point RDO_IDENTIF_COLON RDO_prior dpt_prior_trace
dpt_prior_parent
                                    {
                                        RDOValue* name        =
reinterpret_cast<RDOValue*>($2);
                                        RDOValue* parent_name =
reinterpret_cast<RDOValue*>($5);
                                        if ( parent_name != 0 )
                                        {
                                            const RDODPTPrior*  parentDPTPrior  =
PARSER->findDPTPrior(  parent_name->value().getIdentificator() );
                                            const RDODPTSearch* parentDPTSearch =
PARSER->findDPTSearch( parent_name->value().getIdentificator() );
                                            const RDODPTSome*   parentDPTSome   =
PARSER->findDPTSome(   parent_name->value().getIdentificator() );
                                            if ( parentDPTPrior == NULL &&
parentDPTSome == NULL && parentDPTSearch == NULL )
                                            {
                                                PARSER->error( @1,
rdo::format("Не найдена родитеская точка %s", parent_name->value().getIdentificator().c_str()) );
                                            }
                                            if ( parentDPTSearch  != NULL )
                                            {
                                                PARSER->error( @5, @1, "Точка
принятия решений типа search может содержать лишь активности типа rule и не может быть указана в
качестве родительской точки" );
                                            }
                                            if ( parentDPTPrior  != NULL )
                                            {
                                                LPILogic parent =
parentDPTPrior->getLogic();
                                                $$ = (int)new RDODPTPrior(
PARSER, name->src_info(), parent );
                                            }
                                            if ( parentDPTSome   != NULL )
                                            {
                                                LPILogic parent =
parentDPTSome->getLogic();
```

```
                                                        $$ = (int)new RDODPTPrior(
PARSER, name->src_info(), parent );
                                                }
                                        }
                                        if ( parent_name == 0 )
                                        {
                                                $$ = (int)new RDODPTPrior( PARSER,
name->src_info() );
                                        }
                                };

dpt_prior_condition:  dpt_prior_begin RDO_Condition fun_logic {
                                        RDODPTPrior* dpt =
reinterpret_cast<RDODPTPrior*>($1);
                                        dpt->setCondition(
reinterpret_cast<RDOFUNLogic*>($3) );
                                }
                                | dpt_prior_begin RDO_Condition RDO_NoCheck {
                                        RDODPTPrior* dpt =
reinterpret_cast<RDODPTPrior*>($1);
                                        dpt->setCondition();
                                }
                                | dpt_prior_begin RDO_Condition error {
                                        PARSER->error( @2, @3, "После ключевого слова
$Condition ожидается условие запуска точки" );
                                }
                                | dpt_prior_begin {
                                        RDODPTPrior* dpt =
reinterpret_cast<RDODPTPrior*>($1);
                                        dpt->setCondition();
                                };

dpt_prior_prior:              dpt_prior_condition
                                | dpt_prior_condition RDO_Priority fun_arithm
                                {
                                        if (!PARSER->getLastDPTPrior()->setPrior(
reinterpret_cast<RDOFUNArithm*>($3) ))
                                        {
                                                PARSER->error(@3, _T("Точка принятия
решений пока не может иметь приоритет"));
                                        }
                                }
                                | dpt_prior_condition RDO_Priority error
                                {
                                        PARSER->error( @1, @2, "Ошибка описания
приоритета точки принятия решений" )
                                }
                                | dpt_some_condition error
                                {
                                        PARSER->error( @1, @2, "Ожидается ключевое
слово $Priority" )
                                };

dpt_prior_name:                       RDO_IDENTIF_COLON RDO_IDENTIF {
                                        RDODPTPrior* dpt  = PARSER-
>getLastDPTPrior();
                                        RDOValue* name    =
reinterpret_cast<RDOValue*>($1);
                                        RDOValue* pattern =
reinterpret_cast<RDOValue*>($2);
                                        $$ = (int)dpt->addNewActivity( name-
>src_info(), pattern->src_info() );
                                }
                                | RDO_IDENTIF_COLON error {
                                        PARSER->error( @1, @2, "Ожидается имя
образца" );
                                };

dpt_prior_descr_param:/* empty */
                                | dpt_prior_descr_param  '*'               { PARSER-
>getLastDPTPrior()->getLastActivity()->addParam( RDOValue(RDOParserSrcInfo(@2, "*")) ) }
                                | dpt_prior_descr_param  RDO_INT_CONST     { PARSER-
>getLastDPTPrior()->getLastActivity()->addParam( *reinterpret_cast<RDOValue*>($2) ) }
```

```
                                        | dpt_prior_descr_param  RDO_REAL_CONST    { PARSER-
>getLastDPTPrior()->getLastActivity()->addParam( *reinterpret_cast<RDOValue*>($2) ) }
                                        | dpt_prior_descr_param  RDO_BOOL_CONST    { PARSER-
>getLastDPTPrior()->getLastActivity()->addParam( *reinterpret_cast<RDOValue*>($2) ) }
                                        | dpt_prior_descr_param  RDO_STRING_CONST  { PARSER-
>getLastDPTPrior()->getLastActivity()->addParam( *reinterpret_cast<RDOValue*>($2) ) }
                                        | dpt_prior_descr_param  RDO_IDENTIF       { PARSER-
>getLastDPTPrior()->getLastActivity()->addParam( *reinterpret_cast<RDOValue*>($2) ) }

                                        | dpt_prior_descr_param error
                                        {
                                                PARSER->error( @1, @2, "Ошибка описания
параметра образца" )
                                        };

dpt_prior_activ_prior:/* empty */
                                        | RDO_CF '=' fun_arithm
                                        {
                                                if (!PARSER->getLastDPTPrior()-
>getLastActivity()->setPrior( reinterpret_cast<RDOFUNArithm*>($3) ))
                                                {
                                                        PARSER->error(@3, _T("Активность не
может иметь приоритет"));
                                                }
                                        }
                                        | RDO_CF '=' error
                                        {
                                                PARSER->error( @1, @2, "Ошибка описания
приоритета активности" )
                                        }
                                        | RDO_CF error
                                        {
                                                PARSER->error( @1, @2, "Ошибка: ожидается
знак равенства" )
                                        };

dpt_prior_activity:          /* empty */
                                        | dpt_prior_activity dpt_prior_name
dpt_prior_descr_param dpt_prior_activ_prior{
                                                RDODPTPriorActivity* activity =
reinterpret_cast<RDODPTPriorActivity*>($2);
                                                activity->endParam( @3 );
                                        };

dpt_prior_header:            dpt_prior_prior RDO_Activities dpt_prior_activity {
                                        }
                                        | dpt_prior_prior error {
                                                PARSER->error( @1, @2, "Ожидается ключевое
слово $Activities" );
                                        };

dpt_prior_end:               dpt_prior_header RDO_End {
                                                RDODPTPrior* dpt =
reinterpret_cast<RDODPTPrior*>($1);
                                                dpt->end();
                                        }
                                        | dpt_prior_header {
                                                PARSER->error( @1, "Ожидается ключевое слово
$End" );
                                        };

// ----------------------------------------------------------------------------
// ---------- DPT Free
// ----------------------------------------------------------------------------
dpt_free_prior:                         dpt_free_header
                                                | RDO_Priority fun_arithm dpt_free_header
                                                {
                                                        if (!PARSER->getLastDPTFree()-
>setPrior( reinterpret_cast<RDOFUNArithm*>($2) ))
                                                        {
                                                                PARSER->error(@3, _T("Точка
принятия решений пока не может иметь приоритет"));
                                                        }
                                                }
```

```
                                                | RDO_Priority error dpt_free_header
                                                {
                                                        PARSER->error( @1, @2, "Ошибка
описания приоритета точки принятия решений" )
                                                }
                                                | error dpt_free_header
                                                {
                                                        PARSER->error( @1, @2, "Ожидается
ключевое слово $Priority" )
                                                };

dpt_free_header:                        RDO_Activities {
                                                        $$ = (int)new RDODPTFree( PARSER, @1
);
                                                };

dpt_free_activity:                      /* empty */
                                                | dpt_free_activity dpt_free_activity_name
dpt_free_activity_param dpt_free_activity_keys {
                                                };

dpt_free_activity_name:                 RDO_IDENTIF_COLON RDO_IDENTIF {
                                                        RDODPTFree* dpt     = PARSER-
>getLastDPTFree();
                                                        RDOValue*   name    =
reinterpret_cast<RDOValue*>($1);
                                                        RDOValue*   pattern =
reinterpret_cast<RDOValue*>($2);
                                                        $$ = (int)dpt->addNewActivity( name-
>src_info(), pattern->src_info() );
                                                }
                                                | RDO_IDENTIF_COLON error {
                                                        PARSER->error( @1, @2, "Ожидается имя
образца" );
                                                };

dpt_free_activity_param:        /* empty */
                                                | dpt_free_activity_param  '*'
{ PARSER->getLastDPTFree()->getLastActivity()->addParam( RDOValue(RDOParserSrcInfo(@2, "*")) ) }
                                                | dpt_free_activity_param  RDO_INT_CONST
{ PARSER->getLastDPTFree()->getLastActivity()->addParam( *reinterpret_cast<RDOValue*>($2) ) }
                                                | dpt_free_activity_param  RDO_REAL_CONST
{ PARSER->getLastDPTFree()->getLastActivity()->addParam( *reinterpret_cast<RDOValue*>($2) ) }
                                                | dpt_free_activity_param  RDO_BOOL_CONST
{ PARSER->getLastDPTFree()->getLastActivity()->addParam( *reinterpret_cast<RDOValue*>($2) ) }
                                                | dpt_free_activity_param  RDO_STRING_CONST
{ PARSER->getLastDPTFree()->getLastActivity()->addParam( *reinterpret_cast<RDOValue*>($2) ) }
                                                | dpt_free_activity_param  RDO_IDENTIF
{ PARSER->getLastDPTFree()->getLastActivity()->addParam( *reinterpret_cast<RDOValue*>($2) ) }

                                                | dpt_free_activity_param error
                                                {
                                                        PARSER->error( @1, @2, "Ошибка
описания параметра образца" )
                                                };

dpt_free_activity_keys:                 /* empty */
                                                | dpt_free_activity_keys RDO_STRING_CONST {
                                                        RDODPTFreeActivity* activity = PARSER-
>getLastDPTFree()->getLastActivity();
                                                        std::string         key      =
reinterpret_cast<RDOValue*>($2)->value().getString();
                                                        activity->addHotKey( key, @2 );
                                                }
                                                | dpt_free_activity_keys '+' RDO_STRING_CONST
{
                                                        RDODPTFreeActivity* activity = PARSER-
>getLastDPTFree()->getLastActivity();
                                                        std::string         key      =
reinterpret_cast<RDOValue*>($3)->value().getString();
                                                        activity->addHotKey( key, @3 );
                                                };

dpt_free_end:                           dpt_free_prior dpt_free_activity RDO_End {
```

```
                                        }
                                        | dpt_free_header error {
                                              PARSER->error( @1, "Ожидается ключевое
слово $End" );
                                        };
```

Приложение 2. Код имитационной модели работы почтового отделения связи на языке РДО.

Postoffice.rtp (Типы ресурсов):

```
$Resource_type Отделения : permanent
$Parameters
       Состояние : (Начало_Дня, Рабочий_День, Конец_Дня)
$End

$Resource_type Окна : permanent
$Parameters
       Номер              : integer[1..5]
       Работоспособность  : (Открыто, Закрыто)
       Занятость          : (Свободен, Занят)
       Обслужено          : integer[0..32530] = 0
       В_Очереди          : integer[0..32530] = 0
       Интервал1_Начало   : real
       Интервал1_Конец    : real
       Интервал2_Начало   : real
       Интервал2_Конец    : real
       Интервал3_Начало   : real
       Интервал3_Конец    : real
$End

$Resource_type Клиенты : temporary
$Parameters
       Заявка            : (Rq1,
                                        Rq2,
                                        Rq3,
                                        Rq4,
                                        Rq5,
                                        Rq6,
                                        Rq7,
                                        Rq8,
                                        Rq9,
                                        Rq10,
                                        Rq11,
                                        Rq12,
                                        Rq13,
                                        Rq14,
                                        Rq15,
                                        Rq16,
                                        Rq17,
                                        Rq18,
                                        Rq19)
       Состояние         : (Возник, Пришел, В_Очереди, Обслуживается, Обслужился, Ушел)
       Номер_В_Очереди   : integer[1..32530]
       Номер_Окна        : such_as Окна.Номер
       Приход_В_Очередь  : real
       Время_В_Очереди   : real
$End

$Resource_type Изображаемые_Клиенты : permanent
$Parameters
       Номер  : such_as Окна.Номер
       Заявка : such_as Клиенты.Заявка
$End

$Resource_type Счетчики : permanent
$Parameters
       Номер          : such_as Окна.Номер
       Текущий_Клиент : integer[2..32530]
$End

$Resource_type Счетчики_Заявок : permanent
$Parameters
       Заявка     : such_as Клиенты.Заявка
```

```
        Количество : integer[0..32530]
$End

$Resource_type Счетчики_Времени : permanent
$Parameters
        Номер              : such_as Окна.Номер
        Время_На_Обработку : real
$End

$Resource_type Счетчики_Дней : permanent
$Parameters
        Количество_Дней : integer
$End

$Resource_type Смотрители : permanent
$Parameters
        Номер     : such_as Окна.Номер
        Разрешить : (Да, Нет)
$End

$Resource_type Виды_Нагрузки : permanent
$Parameters
        Вид                   : (Постоянный
                                                    ,Линейный_Увеличение
                                                    ,Линейный_Уменьшение
                                                    ,Равномерный
                                                    ,Экспоненциальный
                                                    ,Нормальный) = Постоянный
        Минимум_Среднее_Число : real
        Максимум_Дисперсия    : real
        Грузить               : (Да, Нет)
$End

$Resource_type Счетчики_Часов : permanent
$Parameters
        Часы                : real
        Соответствие        : (Да,Нет)
        Последнее_Изменение : real
$End

$Resource_type Потерянные_Клиенты : permanent
$Parameters
        Количество : integer
$End
```

Postoffice.rss (Ресурсы):

```
$Resources
        Окно1          : Окна 1 Закрыто Свободен * * 10.0 11.0 12.0 13.0 15.0 18.0
        Окно2          : Окна 2 Закрыто Свободен * *  9.0 18.0  0.0  0.0  0.0  0.0
        Окно3          : Окна 3 Закрыто Свободен * *  9.0 10.0 17.0 18.0  0.0  0.0
        Окно4          : Окна 4 Закрыто Свободен * *  9.0 12.0  0.0  0.0  0.0  0.0
        Окно5          : Окна 5 Закрыто Свободен * * 11.0 16.0  0.0  0.0  0.0  0.0
        Нагрузка       : Виды_Нагрузки Постоянный 0.016666666667 0.0 Да
        Из_Клиент1     : Изображаемые_Клиенты 1 Rq1
        Из_Клиент2     : Изображаемые_Клиенты 2 Rq1
        Из_Клиент3     : Изображаемые_Клиенты 3 Rq1
        Из_Клиент4     : Изображаемые_Клиенты 4 Rq1
        Из_Клиент5     : Изображаемые_Клиенты 5 Rq1
        Count1         : Счетчики_Заявок Rq1  0
        Count2         : Счетчики_Заявок Rq2  0
        Count3         : Счетчики_Заявок Rq3  0
        Count4         : Счетчики_Заявок Rq4  0
        Count5         : Счетчики_Заявок Rq5  0
        Count6         : Счетчики_Заявок Rq6  0
        Count7         : Счетчики_Заявок Rq7  0
        Count8         : Счетчики_Заявок Rq8  0
        Count9         : Счетчики_Заявок Rq9  0
        Count10        : Счетчики_Заявок Rq10 0
        Count11        : Счетчики_Заявок Rq11 0
        Count12        : Счетчики_Заявок Rq12 0
```

```
        Count13         : Счетчики_Заявок Rq13 0
        Count14         : Счетчики_Заявок Rq14 0
        Count15         : Счетчики_Заявок Rq15 0
        Count16         : Счетчики_Заявок Rq16 0
        Count17         : Счетчики_Заявок Rq17 0
        Count18         : Счетчики_Заявок Rq18 0
        Count19         : Счетчики_Заявок Rq19 0
        Счетчик1        : Счетчики 1 2
        Счетчик2        : Счетчики 2 2
        Счетчик3        : Счетчики 3 2
        Счетчик4        : Счетчики 4 2
        Счетчик5        : Счетчики 5 2
        Смотритель1     : Смотрители 1  Нет
        Смотритель2     : Смотрители 2  Нет
        Смотритель3     : Смотрители 3  Нет
        Смотритель4     : Смотрители 4  Нет
        Смотритель5     : Смотрители 5  Нет
        Отделение       : Отделения Начало_Дня
        С_Времени1      : Счетчики_Времени 1 0.0
        С_Времени2      : Счетчики_Времени 2 0.0
        С_Времени3      : Счетчики_Времени 3 0.0
        С_Времени4      : Счетчики_Времени 4 0.0
        С_Времени5      : Счетчики_Времени 5 0.0
        Время           : Счетчики_Часов  9.0 Нет 0.0
        Потерянные      : Потерянные_Клиенты 0
        С_Рабочих_Дней : Счетчики_Дней 1
$End
```

Postoffice.fun (Константы, последовательности и функции):

```
$Constant
  Месяц                : integer = 25
  Квартал              : integer = 73
  Год                  : integer = 289
  Начало1_X            : integer = 112
  Начало_Y_R           : integer = 55
  Начало_Y_T           : integer = 57
  Начало2_X            : integer = 440
  Ширина1              : integer = 329
  Ширина2              : integer = 185
  Высота               : integer = 19
  Начало_Текст1        : integer = 119
  Начало_Текст2        : integer = 448
  См_X                 : integer = 250
  См_Y                 : integer = 47
  См_Ширина            : integer = 329
  См_Высота            : integer = 12
  Начало_Рабочего_Дня : real    = 9.0
  Время_Рабочего_Дня  : real    = 10.0
  Коэф_Времени         : real    = 1.0
  Переполнение_Очереди: integer = 10
$End

$Function Заявки_Окна1 : integer = 0
$Type = list
$Parameters
        Заявка : such_as Клиенты.Заявка
$Body
        Rq1  = 1
        Rq2  = 1
        Rq3  = 1
        Rq4  = 1
        Rq5  = 1
        Rq6  = 1
        Rq7  = 1
        Rq8  = 1
        Rq9  = 1
        Rq10 = 1
        Rq11 = 1
        Rq12 = 1
        Rq13 = 1
```

```
        Rq14 = 1
        Rq15 = 1
        Rq16 = 1
        Rq17 = 1
        Rq18 = 1
        Rq19 = 1
$End

$Function Заявки_Окна2 : integer = 0
$Type = list
$Parameters
        Заявка : such_as Клиенты.Заявка
$Body
        Rq1  = 2
        Rq2  = 2
        Rq3  = 2
        Rq4  = 2
        Rq5  = 2
        Rq6  = 2
        Rq7  = 2
        Rq8  = 2
        Rq9  = 2
        Rq10 = 2
        Rq11 = 2
        Rq12 = 2
        Rq13 = 2
        Rq14 = 2
        Rq15 = 2
        Rq16 = 2
        Rq17 = 2
        Rq18 = 2
        Rq19 = 2
$End

$Function Заявки_Окна3 : integer = 0
$Type = list
$Parameters
  Заявка : such_as Клиенты.Заявка
$Body
  Rq1 = 3
  Rq2 = 3
  Rq3 = 3
  Rq4 = 3
  Rq5 = 3
  Rq6 = 3
  Rq7 = 3
  Rq8 = 3
  Rq9 = 3
  Rq10 = 3
  Rq11 = 3
  Rq12 = 3
  Rq13 = 3
  Rq14 = 3
  Rq15 = 3
  Rq16 = 3
  Rq17 = 3
  Rq18 = 3
  Rq19 = 3
$End

$Function Заявки_Окна4 : integer = 0
$Type = list
$Parameters
  Заявка : such_as Клиенты.Заявка
$Body
  Rq1 = 4
  Rq2 = 4
  Rq3 = 4
  Rq4 = 4
  Rq5 = 4
  Rq6 = 4
  Rq7 = 4
  Rq8 = 4
  Rq9 = 4
```

```
  Rq10 = 4
  Rq11 = 4
  Rq12 = 4
  Rq13 = 4
  Rq14 = 4
  Rq15 = 4
  Rq16 = 4
  Rq17 = 4
  Rq18 = 4
  Rq19 = 4
$End

$Function Заявки_Окна5 : integer = 0
$Type = list
$Parameters
  Заявка : such_as Клиенты.Заявка
$Body
  Rq1 = 5
  Rq2 = 5
  Rq3 = 5
  Rq4 = 5
  Rq5 = 5
  Rq6 = 5
  Rq7 = 5
  Rq8 = 5
  Rq9 = 5
  Rq10 = 5
  Rq11 = 5
  Rq12 = 5
  Rq13 = 5
  Rq14 = 5
  Rq15 = 5
  Rq16 = 5
  Rq17 = 5
  Rq18 = 5
  Rq19 = 5
$End

$Sequence Заявка_Клиента : such_as Клиенты.Заявка
$Type = by_hist
$Body
  Rq1 4.0
  Rq2 44.0
  Rq3 2.0
  Rq4 7.0
  Rq5 9.0
  Rq6 5.0
  Rq7 2.0
  Rq8 30.0
  Rq9 2.0
  Rq10 2.0
  Rq11 2.0
  Rq12 2.0
  Rq13 3.0
  Rq14 74.0
  Rq15 28.0
  Rq16 6.0
  Rq17 30.0
  Rq18 25.0
  Rq19 11.0
$End

$Sequence Нормальный_Закон : real
$Type = normal
$End

$Sequence Равномерный_Закон : real
$Type = uniform
$End

$Sequence Экспоненциальный_Закон : real
$Type = exponential
$End
```

```
$Function Показать_Спрятать : such_as Смотрители.Разрешить
$Type = algorithmic
$Parameters
   Сейчас : such_as Смотрители.Разрешить
   Номер  : such_as Окна.Номер
$Body
  Calculate_if Сейчас = Да  Показать_Спрятать = Нет
  Calculate_if Сейчас = Нет and
                Номер = 1 and
               [Смотритель2.Разрешить = Да or
                Смотритель3.Разрешить = Да or
                Смотритель4.Разрешить = Да or
                Смотритель5.Разрешить = Да ]
                           Показать_Спрятать = Нет
  Calculate_if Сейчас = Нет and
                Номер = 1 and
               [Смотритель2.Разрешить = Нет and
                Смотритель3.Разрешить = Нет and
                Смотритель4.Разрешить = Нет and
                Смотритель5.Разрешить = Нет ]
                           Показать_Спрятать = Да
  Calculate_if Сейчас = Нет and
                Номер = 2 and
               [Смотритель1.Разрешить = Да or
                Смотритель3.Разрешить = Да or
                Смотритель4.Разрешить = Да or
                Смотритель5.Разрешить = Да ]
                           Показать_Спрятать = Нет
  Calculate_if Сейчас = Нет and
                Номер = 2 and
               [Смотритель1.Разрешить = Нет and
                Смотритель3.Разрешить = Нет and
                Смотритель4.Разрешить = Нет and
                Смотритель5.Разрешить = Нет ]
                           Показать_Спрятать = Да
  Calculate_if Сейчас = Нет and
                Номер = 3 and
               [Смотритель1.Разрешить = Да or
                Смотритель2.Разрешить = Да or
                Смотритель4.Разрешить = Да or
                Смотритель5.Разрешить = Да ]
                           Показать_Спрятать = Нет
  Calculate_if Сейчас = Нет and
                Номер = 3 and
               [Смотритель1.Разрешить = Нет and
                Смотритель2.Разрешить = Нет and
                Смотритель4.Разрешить = Нет and
                Смотритель5.Разрешить = Нет ]
                           Показать_Спрятать = Да
  Calculate_if Сейчас = Нет and
                Номер = 4 and
               [Смотритель1.Разрешить = Да or
                Смотритель2.Разрешить = Да or
                Смотритель3.Разрешить = Да or
                Смотритель5.Разрешить = Да ]
                           Показать_Спрятать = Нет
  Calculate_if Сейчас = Нет and
                Номер = 4 and
               [Смотритель1.Разрешить = Нет and
                Смотритель2.Разрешить = Нет and
                Смотритель3.Разрешить = Нет and
                Смотритель5.Разрешить = Нет ]
                           Показать_Спрятать = Да
  Calculate_if Сейчас = Нет and
                Номер = 5 and
               [Смотритель1.Разрешить = Да or
                Смотритель2.Разрешить = Да or
                Смотритель3.Разрешить = Да or
                Смотритель4.Разрешить = Да ]
                           Показать_Спрятать = Нет
  Calculate_if Сейчас = Нет and
                Номер = 5 and
               [Смотритель1.Разрешить = Нет and
                Смотритель2.Разрешить = Нет and
```

```
                Смотритель3.Разрешить = Нет and
                Смотритель4.Разрешить = Нет ]
                          Показать_Спрятать = Да
$End

$Function Нормальный_Закон_Abs : real
$Type = algorithmic
$Parameters
  Результат : real
$Body
  Calculate_if Результат > 0 Нормальный_Закон_Abs = Результат
  Calculate_if Результат < 0 Нормальный_Закон_Abs = -1 * Результат
  Calculate_if Результат = 0 Нормальный_Закон_Abs = 1/60
$End

$Function Время_Прихода_Клиента  : real
$Type = algorithmic
$Parameters
  Вид    : such_as Виды_Нагрузки.Вид
  Число1 : real
  Число2 : real
$Body
  Calculate_if Вид = Постоянный
                       Время_Прихода_Клиента = Число1

  Calculate_if Вид = Линейный_Увеличение and
               Время.Часы = 9.0
                       Время_Прихода_Клиента = Число1
  Calculate_if Вид = Линейный_Увеличение and
               Время.Часы = 10.0
                       Время_Прихода_Клиента = Число1 + 1 * (Число2 - Число1) /
(Время_Рабочего_Дня - 1)
  Calculate_if Вид = Линейный_Увеличение and
               Время.Часы = 11.0
                       Время_Прихода_Клиента = Число1 + 2 * (Число2 - Число1) /
(Время_Рабочего_Дня - 1)
  Calculate_if Вид = Линейный_Увеличение and
               Время.Часы = 12.0
                       Время_Прихода_Клиента = Число1 + 3 * (Число2 - Число1) /
(Время_Рабочего_Дня - 1)
  Calculate_if Вид = Линейный_Увеличение and
               Время.Часы = 13.0
                       Время_Прихода_Клиента = Число1 + 4 * (Число2 - Число1) /
(Время_Рабочего_Дня - 1)
  Calculate_if Вид = Линейный_Увеличение and
               Время.Часы = 14.0
                       Время_Прихода_Клиента = Число1 + 5 * (Число2 - Число1) /
(Время_Рабочего_Дня - 1)
  Calculate_if Вид = Линейный_Увеличение and
               Время.Часы = 15.0
                       Время_Прихода_Клиента = Число1 + 6 * (Число2 - Число1) /
(Время_Рабочего_Дня - 1)
  Calculate_if Вид = Линейный_Увеличение and
               Время.Часы = 16.0
                       Время_Прихода_Клиента = Число1 + 7 * (Число2 - Число1) /
(Время_Рабочего_Дня - 1)
  Calculate_if Вид = Линейный_Увеличение and
               Время.Часы = 17.0
                       Время_Прихода_Клиента = Число1 + 8 * (Число2 - Число1) /
(Время_Рабочего_Дня - 1)
  Calculate_if Вид = Линейный_Увеличение and
               Время.Часы = 18.0
                       Время_Прихода_Клиента = Число1 + 9 * (Число2 - Число1) /
(Время_Рабочего_Дня - 1)
  Calculate_if Вид = Линейный_Увеличение and
               Время.Часы = 19.0
                       Время_Прихода_Клиента = Число1 + 10 * (Число2 - Число1) /
(Время_Рабочего_Дня - 1)
  Calculate_if Вид = Линейный_Увеличение and
               Время.Часы = 20.0
                       Время_Прихода_Клиента = Число1 + 11 * (Число2 - Число1) /
(Время_Рабочего_Дня - 1)

  Calculate_if Вид = Линейный_Уменьшение and
```

```
              Время.Часы = Время_Рабочего_Дня + 9.0
                      Время_Прихода_Клиента = Число1
  Calculate_if Вид = Линейный_Уменьшение and
              Время.Часы = 9.0
                      Время_Прихода_Клиента = Число2
  Calculate_if Вид = Линейный_Уменьшение and
              Время.Часы = 10.0
                      Время_Прихода_Клиента = Число2 -  1 * (Число2 - Число1) /
(Время_Рабочего_Дня - 1)
  Calculate_if Вид = Линейный_Уменьшение and
              Время.Часы = 11.0
                      Время_Прихода_Клиента = Число2 - 2 * (Число2 - Число1) /
(Время_Рабочего_Дня - 1)
  Calculate_if Вид = Линейный_Уменьшение and
              Время.Часы = 12.0
                      Время_Прихода_Клиента = Число2 - 3 * (Число2 - Число1) /
(Время_Рабочего_Дня - 1)
  Calculate_if Вид = Линейный_Уменьшение and
              Время.Часы = 13.0
                      Время_Прихода_Клиента = Число2 - 4 * (Число2 - Число1) /
(Время_Рабочего_Дня - 1)
  Calculate_if Вид = Линейный_Уменьшение and
              Время.Часы = 14.0
                      Время_Прихода_Клиента = Число2 - 5 * (Число2 - Число1) /
(Время_Рабочего_Дня - 1)
  Calculate_if Вид = Линейный_Уменьшение and
              Время.Часы = 15.0
                      Время_Прихода_Клиента = Число2 - 6 * (Число2 - Число1) /
(Время_Рабочего_Дня - 1)
  Calculate_if Вид = Линейный_Уменьшение and
              Время.Часы = 16.0
                      Время_Прихода_Клиента = Число2 - 7 * (Число2 - Число1) /
(Время_Рабочего_Дня - 1)
  Calculate_if Вид = Линейный_Уменьшение and
              Время.Часы = 17.0
                      Время_Прихода_Клиента = Число2 - 8 * (Число2 - Число1) /
(Время_Рабочего_Дня - 1)
  Calculate_if Вид = Линейный_Уменьшение and
              Время.Часы = 18.0
                      Время_Прихода_Клиента = Число2 - 9 * (Число2 - Число1) /
(Время_Рабочего_Дня - 1)
  Calculate_if Вид = Линейный_Уменьшение and
              Время.Часы = 19.0
                      Время_Прихода_Клиента = Число2 - 10 * (Число2 - Число1) /
(Время_Рабочего_Дня - 1)
  Calculate_if Вид = Линейный_Уменьшение and
              Время.Часы = 20.0
                      Время_Прихода_Клиента = Число2 - 11 * (Число2 - Число1) /
(Время_Рабочего_Дня - 1)
  Calculate_if Вид = Равномерный
                      Время_Прихода_Клиента = Равномерный_Закон(Число1,Число2)
  Calculate_if Вид = Нормальный
                      Время_Прихода_Клиента =
Нормальный_Закон_Abs(Нормальный_Закон(Число1,Число2))
  Calculate_if Вид = Экспоненциальный
                      Время_Прихода_Клиента = Экспоненциальный_Закон(Число1)
$End

$Function Время_Обслуживания : real
$Type = algorithmic
$Parameters
  Заявка : such_as Клиенты.Заявка
$Body
  Calculate_if Заявка = Rq1 Время_Обслуживания = Равномерный_Закон(0.0375, 0.045833333333)
  Calculate_if Заявка = Rq2 Время_Обслуживания = Равномерный_Закон(0.023333333333,
0.028666666667)
  Calculate_if Заявка = Rq3 Время_Обслуживания = Равномерный_Закон(0.06, 0.073333333333)
  Calculate_if Заявка = Rq4 Время_Обслуживания = Равномерный_Закон(0.024, 0.029333333333)
  Calculate_if Заявка = Rq5 Время_Обслуживания = Равномерный_Закон(0.102, 0.12466666667)
  Calculate_if Заявка = Rq6 Время_Обслуживания = Равномерный_Закон(0.039, 0.047666666667)
  Calculate_if Заявка = Rq7 Время_Обслуживания = Равномерный_Закон(0.1725, 0.21083333333)
  Calculate_if Заявка = Rq8 Время_Обслуживания = Равномерный_Закон(0.0515, 0.062833333333)
  Calculate_if Заявка = Rq9 Время_Обслуживания = Равномерный_Закон(0.0225, 0.0275)
  Calculate_if Заявка = Rq10 Время_Обслуживания = Равномерный_Закон(0.0225, 0.0275)
```

```
  Calculate_if Заявка = Rq11 Время_Обслуживания = Равномерный_Закон(0.0975, 0.11916666667)
  Calculate_if Заявка = Rq12 Время_Обслуживания = Равномерный_Закон(0.0375, 0.045833333333)
  Calculate_if Заявка = Rq13 Время_Обслуживания = Равномерный_Закон(0.015, 0.018333333333)
  Calculate_if Заявка = Rq14 Время_Обслуживания = Равномерный_Закон(0.019666666667, 0.024)
  Calculate_if Заявка = Rq15 Время_Обслуживания = Равномерный_Закон(0.0165, 0.020166666667)
  Calculate_if Заявка = Rq16 Время_Обслуживания = Равномерный_Закон(0.035, 0.042666666667)
  Calculate_if Заявка = Rq17 Время_Обслуживания = Равномерный_Закон(0.020333333333,
0.024833333333)
  Calculate_if Заявка = Rq18 Время_Обслуживания = Равномерный_Закон(0.0275, 0.0335)
  Calculate_if Заявка = Rq19 Время_Обслуживания = Равномерный_Закон(0.036, 0.044)
$End

$Function ВремечкО : real = 0.0
$Type = list
$Parameters
  Заявка : such_as Клиенты.Заявка
$Body
  Rq1 = 0.033333333333
  Rq2 = 0.033333333333
  Rq3 = 0.016666666667
  Rq4 = 0.033333333333
  Rq5 = 0.021666666667
  Rq6 = 0.145
  Rq7 = 0.03
  Rq8 = 0.0016666666667
  Rq9 = 0.145
  Rq10 = 0.033333333333
  Rq11 = 0.033333333333
  Rq12 = 0.03
  Rq13 = 0.033333333333
  Rq14 = 0.033333333333
  Rq15 = 0.033333333333
  Rq16 = 0.033333333333
  Rq17 = 0.033333333333
  Rq18 = 0.033333333333
  Rq19 = 0.033333333333
$End
```

Postoffice.pat (Образцы):

```
$Pattern Приход_Клиента_На_Почту : operation
$Relevant_resources
       Рел_Отделение : Отделения      NoChange NoChange
       Рел_Нагрузка  : Виды_Нагрузки  Keep     Keep
       Рел_Клиент    : Клиенты        Create   NoChange
$Time = Время_Прихода_Клиента(Рел_Нагрузка.Вид,
                              Рел_Нагрузка.Минимум_Среднее_Число,
                              Рел_Нагрузка.Максимум_Дисперсия)
$Body
  Рел_Отделение
    Choice from Рел_Отделение.Состояние = Рабочий_День
    first
  Рел_Нагрузка
    Choice from Рел_Нагрузка.Грузить = Да
    first
    Convert_begin
      Грузить set Нет
    Convert_end
      Грузить set Да
  Рел_Клиент
    Convert_begin
      Заявка           set Заявка_Клиента
      Состояние        set Возник
      Номер_В_Очереди  set 1
      Номер_Окна       set 1
      Приход_В_Очередь set Time_now
      Время_В_Очереди  set 0.0
$End

$Pattern Определить_Окно : rule
$Relevant_resources
```

```
  Рел_Клиент : Клиенты  Keep
  Рел_Окно   : Окна     NoChange
$Body
  Рел_Клиент
    Choice from Рел_Клиент.Состояние = Возник
    first
    Convert_rule
      Состояние  set Пришел
      Номер_Окна set Рел_Окно.Номер
  Рел_Окно
    Choice from [Рел_Окно.Номер = Заявки_Окна1(Рел_Клиент.Заявка) and Окно1.Работоспособность =
Открыто] or
                [Рел_Окно.Номер = Заявки_Окна2(Рел_Клиент.Заявка) and Окно2.Работоспособность =
Открыто] or
                [Рел_Окно.Номер = Заявки_Окна3(Рел_Клиент.Заявка) and Окно3.Работоспособность =
Открыто] or
                [Рел_Окно.Номер = Заявки_Окна4(Рел_Клиент.Заявка) and Окно4.Работоспособность =
Открыто] or
                [Рел_Окно.Номер = Заявки_Окна5(Рел_Клиент.Заявка) and Окно5.Работоспособность =
Открыто]
    with_min(Рел_Окно.В_Очереди)
$End

$Pattern Ликвидирование : rule
$Relevant_resources
  Рел_Клиент            : Клиенты             Erase
  Рел_Потерянный_Клиент : Потерянные_Клиенты  Keep
$Body
  Рел_Клиент
    Choice from Рел_Клиент.Состояние = Возник
    first
  Рел_Потерянный_Клиент
    Choice NoCheck
    first
    Convert_rule
      Количество set Рел_Потерянный_Клиент.Количество + 1
$End

$Pattern Открыть_Окно : rule
$Relevant_resources
       Рел_Время : Время  NoChange
       Рел_Окно  : Окна   Keep
$Body
  Рел_Время
    Choice NoCheck
    first
  Рел_Окно
    Choice from Рел_Окно.Работоспособность = Закрыто and
               (Рел_Окно.Интервал1_Начало = Рел_Время.Часы  or
                Рел_Окно.Интервал2_Начало = Рел_Время.Часы  or
                Рел_Окно.Интервал3_Начало = Рел_Время.Часы)
    first
    Convert_rule
      Работоспособность set Открыто
$End

$Pattern Закрыть_Окно : rule
$Relevant_resources
  Рел_Время : Время  NoChange
  Рел_Окно  : Окна   Keep
$Body
  Рел_Время
    Choice NoCheck
    first
  Рел_Окно
    Choice from Рел_Окно.Работоспособность = Открыто and
               (Рел_Окно.Интервал1_Конец = Рел_Время.Часы  or
                Рел_Окно.Интервал2_Конец = Рел_Время.Часы  or
                Рел_Окно.Интервал3_Конец = Рел_Время.Часы)
    first
    Convert_rule
      Работоспособность set Закрыто
$End
```

```
$Pattern Приход_Клиента_В_Очередь : rule
$Relevant_resources
  Рел_Клиент : Клиенты  Keep
  Рел_Окно   : Окна     Keep
$Body
  Рел_Клиент
    Choice from Рел_Клиент.Состояние = Пришел
    first
    Convert_rule
      Состояние        set В_Очереди
      Номер_В_Очереди  set Рел_Окно.В_Очереди + 1
      Приход_В_Очередь set Time_now
  Рел_Окно
    Choice from Рел_Окно.Номер = Рел_Клиент.Номер_Окна
    first
    Convert_rule
      В_Очереди set Рел_Окно.В_Очереди + 1
$End

$Pattern Сдвинуть_Очередь : rule
$Parameters
  Номер : such_as Окна.Номер
$Relevant_resources
  Рел_Счетчик : Счетчики  Keep
  Рел_Клиент  : Клиенты   Keep
$Body
  Рел_Счетчик
    Choice from Рел_Счетчик.Номер = Номер
    first
    Convert_rule
      Текущий_Клиент set Рел_Счетчик.Текущий_Клиент + 1
  Рел_Клиент
    Choice from Рел_Клиент.Номер_В_Очереди = Рел_Счетчик.Текущий_Клиент  and
                Рел_Клиент.Номер_Окна = Номер
    first
    Convert_rule
      Номер_В_Очереди set Рел_Клиент.Номер_В_Очереди - 1
$End

$Pattern Функционирование : operation
$Relevant_resources
  Рел_Отделение : Отделения  Keep Keep
$Time = Время_Рабочего_Дня
$Body
  Рел_Отделение
    Choice from Рел_Отделение.Состояние = Начало_Дня
    first
    Convert_begin
      Состояние set Рабочий_День
    Convert_end
      Состояние set Конец_Дня
$End

$Pattern Обслуживание_Первого : operation
$Parameters
  Номер : such_as Окна.Номер
$Relevant_resources
  Рел_Отделение       : Отделения             NoChange NoChange
  Рел_Окно            : Окна                  Keep     Keep
  Рел_Клиент          : Клиенты               Keep     Keep
  Рел_Счетчик         : Счетчики              Keep     NoChange
  Рел_Счетчик_Заявок  : Счетчики_Заявок       NoChange Keep
  Рел_Счетчик_Времени : Счетчики_Времени      NoChange Keep
  Рел_Из_Клиент       : Изображаемые_Клиенты  Keep     NoChange
$Time = Коэф_Времени * Время_Обслуживания(Рел_Клиент.Заявка)
$Body
  Рел_Отделение
    Choice from Рел_Отделение.Состояние = Рабочий_День
    first
  Рел_Окно
    Choice from Рел_Окно.Номер = Номер and
                Рел_Окно.Занятость = Свободен
    first
    Convert_begin
```

```
      Занятость set Занят
    Convert_end
      Занятость set Свободен
      Обслужено set Рел_Окно.Обслужено + 1
      В_Очереди set Рел_Окно.В_Очереди - 1
  Рел_Клиент
    Choice from  Рел_Клиент.Состояние = В_Очереди and
                 Рел_Клиент.Номер_Окна = Номер and
                 Рел_Окно.Занятость = Свободен and
                 Рел_Клиент.Номер_В_Очереди = 1
    first
    Convert_begin
      Состояние       set Обслуживается
      Время_В_Очереди set Time_now - Рел_Клиент.Приход_В_Очередь
    Convert_end
      Состояние      set  Ушел
  Рел_Счетчик
    Choice from Рел_Счетчик.Номер = Номер and
                Рел_Отделение.Состояние = Рабочий_День
    first
    Convert_begin
      Текущий_Клиент set 2
  Рел_Счетчик_Заявок
    Choice from Рел_Счетчик_Заявок.Заявка = Рел_Клиент.Заявка
    first
    Convert_end
      Количество set Рел_Счетчик_Заявок.Количество + 1
  Рел_Счетчик_Времени
    Choice from Рел_Счетчик_Времени.Номер = Номер
    first
    Convert_end
      Время_На_Обработку set Рел_Счетчик_Времени.Время_На_Обработку +
                               ВремечкО(Рел_Клиент.Заявка)
 Рел_Из_Клиент
   Choice from Рел_Из_Клиент.Номер = Номер
   first
   Convert_begin
     Заявка set Рел_Клиент.Заявка
$End

$Pattern Уничтожение : rule
$Parameters
  Номер : such_as Окна.Номер
$Relevant_resources
  Рел_Отделение : Отделения  NoChange
  Рел_Окно      : Окна       Keep
  Рел_Клиент    : Клиенты    Erase
$Body
  Рел_Отделение
    Choice from Рел_Отделение.Состояние = Конец_Дня
    first
  Рел_Окно
    Choice from Рел_Окно.Номер = Номер and
                Рел_Окно.Занятость = Свободен and
                Рел_Окно.В_Очереди > 0
    first
    Convert_rule
      В_Очереди set Рел_Окно.В_Очереди - 1
  Рел_Клиент
    Choice from Рел_Клиент.Состояние = В_Очереди and
                Рел_Клиент.Номер_Окна = Номер
    first
$End

$Pattern Переход_К_Новому_Дню : rule
$Relevant_resources
  Рел_Отделение         : Отделения          Keep
  Рел_Счетчик_Дней      : Счетчики_Дней      Keep
  Рел_Счетчик_Времени1 : Счетчики_Времени  Keep
  Рел_Счетчик_Времени2 : Счетчики_Времени  Keep
  Рел_Счетчик_Времени3 : Счетчики_Времени  Keep
  Рел_Счетчик_Времени4 : Счетчики_Времени  Keep
  Рел_Счетчик_Времени5 : Счетчики_Времени  Keep
  Рел_Время             : Время              Keep
```

```
$Body
  Рел_Отделение
    Choice from Окно1.В_Очереди + Окно2.В_Очереди + Окно3.В_Очереди +
                Окно4.В_Очереди + Окно5.В_Очереди = 0 and
                Рел_Отделение.Состояние = Конец_Дня
    first
    Convert_rule
      Состояние set Начало_Дня
  Рел_Счетчик_Дней
    Choice  NoCheck
    first
    Convert_rule
      Количество_Дней set Рел_Счетчик_Дней.Количество_Дней + 1
  Рел_Счетчик_Времени1
    Choice from Рел_Счетчик_Времени1.Номер = 1
    first
    Convert_rule
      Время_На_Обработку set 0.0
  Рел_Счетчик_Времени2
    Choice from Рел_Счетчик_Времени2.Номер = 2
    first
    Convert_rule
      Время_На_Обработку set 0.0
  Рел_Счетчик_Времени3
    Choice from Рел_Счетчик_Времени3.Номер = 3
    first
    Convert_rule
      Время_На_Обработку set 0.0
  Рел_Счетчик_Времени4
    Choice from Рел_Счетчик_Времени4.Номер = 4
    first
    Convert_rule
     Время_На_Обработку set 0.0
  Рел_Счетчик_Времени5
    Choice from Рел_Счетчик_Времени5.Номер = 5
    first
    Convert_rule
      Время_На_Обработку set 0.0
  Рел_Время
    Choice NoCheck
    first
    Convert_rule
      Часы                set 9.0
      Последнее_Изменение set Time_now
$End

$Pattern Уход_Клиента : rule
$Parameters
  Номер : such_as Окна.Номер
$Relevant_resources
  Рел_Окно    : Окна     NoChange
  Рел_Клиент  : Клиенты  Erase
$Body
  Рел_Окно
    Choice from Рел_Окно.Номер = Номер
    first
  Рел_Клиент
    Choice from Рел_Клиент.Состояние = Ушел
    first
$End

$Pattern Обслуживание : keyboard
$Parameters
  Номер : such_as  Окна.Номер
$Relevant_resources
  Рел_Смотритель : Смотрители  Keep NoChange
$Time = 0.0
$Body
  Рел_Смотритель
    Choice from Рел_Смотритель.Номер  =  Номер
    first
    Convert_begin
      Разрешить set Показать_Спрятать(Рел_Смотритель.Разрешить,Номер)
$End
```

```
$Pattern Соответствие_Времени : operation
$Relevant_resources
  Рел_Время : Время  Keep Keep
$Time = 1.0
$Body
  Рел_Время
    Choice from Рел_Время.Соответствие = Нет and
                (Time_now - Рел_Время.Последнее_Изменение) >= 1.0
    first
    Convert_begin
      Часы                set Рел_Время.Часы + 1
      Соответствие        set Да
      Последнее_Изменение set Time_now
    Convert_end
      Соответствие        set Нет
$End
```

Postoffice.dpt (Точки принятия решений):

```
$Decision_point main : some
$Activities
$End

$Decision_point рабочий_день : some trace
$Parent main
$Condition Отделение.Состояние = Рабочий_День
$Activities
$End

$Decision_point приход_клиентов : some
$Parent рабочий_день
$Activities
       Еще_Один_Пришел    : Приход_Клиента_На_Почту
$End

$Decision_point размещение_клиентов : some
$Parent рабочий_день
$Activities
       Какое_Окно         : Определить_Окно
       Потерянного_Вон    : Ликвидирование
       В_Очередь_Его      : Приход_Клиента_В_Очередь
$End

$Decision_point обслуживание_клиентов : some
$Parent рабочий_день
$Activities
       Наконец_То_В_Окно1 : Обслуживание_Первого 1
       Наконец_То_В_Окно2 : Обслуживание_Первого 2
       Наконец_То_В_Окно3 : Обслуживание_Первого 3
       Наконец_То_В_Окно4 : Обслуживание_Первого 4
       Наконец_То_В_Окно5 : Обслуживание_Первого 5
$End

$Decision_point продвижение_очередей : some
$Parent рабочий_день
$Activities
       Вперед1            : Сдвинуть_Очередь 1
       Вперед2            : Сдвинуть_Очередь 2
       Вперед3            : Сдвинуть_Очередь 3
       Вперед4            : Сдвинуть_Очередь 4
       Вперед5            : Сдвинуть_Очередь 5
$End

$Decision_point уход_клиентов : some
$Parent рабочий_день
$Activities
       От_Окна1           : Уход_Клиента 1
       От_Окна2           : Уход_Клиента 2
       От_Окна3           : Уход_Клиента 3
```

```
       От_Окна4            : Уход_Клиента 4
       От_Окна5            : Уход_Клиента 5
$End

$Decision_point открытие_закрытие_окна : some
$Parent рабочий_день
$Activities
       Я_Работаю           : Открыть_Окно
       Я_Закрылось         : Закрыть_Окно
$End

$Decision_point конец_дня : some
$Parent main
$Condition Отделение.Состояние = Конец_Дня
$Activities
       Поехали_Снова       : Переход_К_Новому_Дню
$End

$Decision_point уход_клиентов_в_конце_дня : some
$Parent конец_дня
$Activities
       Лишних_Вон1         : Уничтожение 1
       Лишних_Вон2         : Уничтожение 2
       Лишних_Вон3         : Уничтожение 3
       Лишних_Вон4         : Уничтожение 4
       Лишних_Вон5         : Уничтожение 5
$End

$Decision_point служебные : some
$Parent main
$Activities
       Часы_На_Экран       : Соответствие_Времени
       Поехали             : Функционирование
$End
```

Postoffice.frm (Анимация):

```
$Frame Зал
$Back_picture = <255 255 255> post
Show
  bitmap[318,398,w_day,w_day_m]
  text[555,405,150,12,transparent,<0 255 0>,C_Рабочих_Дней.Количество_Дней]
Show_if Окно1.Работоспособность = Открыто
  bitmap[138,120,k1,k_m]
  bitmap[140,160,oper,oper_m]
Show_if Окно1.Работоспособность = Закрыто and
        Окно1.В_Очереди >= 1
  bitmap[138,120,nk1,k_m]
  bitmap[140,160,oper,oper_m]
Show_if Окно1.Работоспособность = Закрыто and
        Окно1.В_Очереди = 0
  bitmap[138,120,nk1,k_m]
Show_if Окно2.Работоспособность = Открыто
  bitmap[243,120,k2,k_m]
  bitmap[245,160,oper,oper_m]
Show_if Окно2.Работоспособность = Закрыто and
        Окно2.В_Очереди >= 1
  bitmap[243,120,nk2,k_m]
  bitmap[245,160,oper,oper_m]
Show_if Окно2.Работоспособность = Закрыто and
        Окно2.В_Очереди  = 0
  bitmap[243,120,nk2,k_m]
Show_if Окно3.Работоспособность = Открыто
  bitmap[348,120,k3,k_m]
  bitmap[350,160,oper,oper_m]
Show_if Окно3.Работоспособность = Закрыто and
        Окно3.В_Очереди >= 1
  bitmap[348,120,nk3,k_m]
  bitmap[350,160,oper,oper_m]
Show_if Окно3.Работоспособность = Закрыто and
```

```
        Окно3.В_Очереди  = 0
  bitmap[348,120,nk3,k_m]
Show_if Окно4.Работоспособность = Открыто
  bitmap[453,120,k4,k_m]
  bitmap[455,160,oper,oper_m]
Show_if Окно4.Работоспособность = Закрыто and
        Окно4.В_Очереди  >= 1
  bitmap[453,120,nk4,k_m]
  bitmap[455,160,oper,oper_m]
Show_if Окно4.Работоспособность = Закрыто and
        Окно4.В_Очереди  = 0
  bitmap[453,120,nk4,k_m]
Show_if Окно5.Работоспособность = Открыто
  bitmap[559,120,k5,k_m]
  bitmap[561,160,oper,oper_m]
Show_if Окно5.Работоспособность = Закрыто and
        Окно5.В_Очереди  >= 1
  bitmap[559,120,nk5,k_m]
  bitmap[561,160,oper,oper_m]
Show_if Окно5.Работоспособность = Закрыто and
        Окно5.В_Очереди  = 0
  bitmap[559,120,nk5,k_m]
Show_if Окно1.В_Очереди >= 1
  bitmap[148,157,others,others_m]
Show_if Окно1.В_Очереди >= 2
  bitmap[171,157,second,second_m]
Show_if Окно1.В_Очереди >= 3
  bitmap[195,160,others,others_m]
Show_if Окно1.В_Очереди >= 4
  bitmap[195,168,others,others_m]
Show_if Окно1.В_Очереди >= 5
  bitmap[195,176,others,others_m]
Show_if Окно1.В_Очереди >= 6
  bitmap[195,184,others,others_m]
Show_if Окно1.В_Очереди >= 7
  bitmap[195,192,others,others_m]
Show_if Окно1.В_Очереди >= 8
  bitmap[195,200,others,others_m]
Show_if Окно1.В_Очереди >= 9
  bitmap[195,208,others,others_m]
Show_if Окно1.В_Очереди >= 10
  bitmap[195,216,others,others_m]
Show_if Окно2.В_Очереди >= 1
  bitmap[255,157,others,others_m]
Show_if Окно2.В_Очереди >= 2
  bitmap[278,157,second,second_m]
Show_if Окно2.В_Очереди >= 3
  bitmap[302,160,others,others_m]
Show_if Окно2.В_Очереди >= 4
  bitmap[302,168,others,others_m]
Show_if Окно2.В_Очереди >= 5
  bitmap[302,176,others,others_m]
Show_if Окно2.В_Очереди >= 6
  bitmap[302,184,others,others_m]
Show_if Окно2.В_Очереди >= 7
  bitmap[302,192,others,others_m]
Show_if Окно2.В_Очереди >= 8
  bitmap[302,200,others,others_m]
Show_if Окно2.В_Очереди >= 9
  bitmap[302,208,others,others_m]
Show_if Окно2.В_Очереди >= 10
  bitmap[302,216,others,others_m]
Show_if Окно3.В_Очереди >= 1
  bitmap[365,157,others,others_m]
Show_if Окно3.В_Очереди >= 2
  bitmap[388,157,second,second_m]
Show_if Окно3.В_Очереди >= 3
  bitmap[412,160,others,others_m]
Show_if Окно3.В_Очереди >= 4
  bitmap[412,168,others,others_m]
Show_if Окно3.В_Очереди >= 5
  bitmap[412,176,others,others_m]
Show_if Окно3.В_Очереди >= 6
  bitmap[412,184,others,others_m]
```

```
Show_if Окно3.В_Очереди >= 7
  bitmap[412,192,others,others_m]
Show_if Окно3.В_Очереди >= 8
  bitmap[412,200,others,others_m]
Show_if Окно3.В_Очереди >= 9
  bitmap[412,208,others,others_m]
Show_if Окно3.В_Очереди >= 10
  bitmap[412,216,others,others_m]
Show_if Окно4.В_Очереди >= 1
  bitmap[465,157,others,others_m]
Show_if Окно4.В_Очереди >= 2
  bitmap[488,157,second,second_m]
Show_if Окно4.В_Очереди >= 3
  bitmap[512,160,others,others_m]
Show_if Окно4.В_Очереди >= 4
  bitmap[512,168,others,others_m]
Show_if Окно4.В_Очереди >= 5
  bitmap[512,176,others,others_m]
Show_if Окно4.В_Очереди >= 6
  bitmap[512,184,others,others_m]
Show_if Окно4.В_Очереди >= 7
  bitmap[512,192,others,others_m]
Show_if Окно4.В_Очереди >= 8
  bitmap[512,200,others,others_m]
Show_if Окно4.В_Очереди >= 9
  bitmap[512,208,others,others_m]
Show_if Окно4.В_Очереди >= 10
  bitmap[512,216,others,others_m]
Show_if Окно5.В_Очереди >= 1
  bitmap[565,157,others,others_m]
Show_if Окно5.В_Очереди >= 2
  bitmap[588,157,second,second_m]
Show_if Окно5.В_Очереди >= 3
  bitmap[612,160,others,others_m]
Show_if Окно5.В_Очереди >= 4
  bitmap[612,168,others,others_m]
Show_if Окно5.В_Очереди >= 5
  bitmap[612,176,others,others_m]
Show_if Окно5.В_Очереди >= 6
  bitmap[612,184,others,others_m]
Show_if Окно5.В_Очереди >= 7
  bitmap[612,192,others,others_m]
Show_if Окно5.В_Очереди >= 8
  bitmap[612,200,others,others_m]
Show_if Окно5.В_Очереди >= 9
  bitmap[612,208,others,others_m]
Show_if Окно5.В_Очереди >= 10
  bitmap[612,216,others,others_m]
Show_if [Окно1.В_Очереди >= Переполнение_Очереди or
         Окно2.В_Очереди >= Переполнение_Очереди or
         Окно3.В_Очереди >= Переполнение_Очереди or
         Окно4.В_Очереди >= Переполнение_Очереди or
         Окно5.В_Очереди >= Переполнение_Очереди] and
         Frac(Floor(Seconds) / 2) = 0
  bitmap[318,426,ocher,ocher_m]
Show_if [Окно1.В_Очереди >= Переполнение_Очереди or
         Окно2.В_Очереди >= Переполнение_Очереди or
         Окно3.В_Очереди >= Переполнение_Очереди or
         Окно4.В_Очереди >= Переполнение_Очереди or
         Окно5.В_Очереди >= Переполнение_Очереди] and
         Frac(Floor(Seconds) / 2) <> 0
  bitmap[318,426,ocher_1,ocher_m]
Show_if Время.Часы = 9.0
  bitmap[601,398,c9,c0_m]
Show_if Время.Часы = 10.0
  bitmap[601,398,c10,c0_m]
Show_if Время.Часы = 11.0
  bitmap[601,398,c11,c0_m]
Show_if Время.Часы = 12.0
  bitmap[601,398,c12,c0_m]
Show_if Время.Часы = 13.0
  bitmap[601,398,c13,c0_m]
Show_if Время.Часы = 14.0
  bitmap[601,398,c14,c0_m]
```

```
Show_if Время.Часы = 15.0
  bitmap[601,398,c15,c0_m]
Show_if Время.Часы = 16.0
  bitmap[601,398,c16,c0_m]
Show_if Время.Часы = 17.0
  bitmap[601,398,c17,c0_m]
Show_if Время.Часы = 18.0
  bitmap[601,398,c18,c0_m]
Show_if Время.Часы = 19.0
  bitmap[601,398,c19,c0_m]
Show_if Время.Часы = 20.0
  bitmap[601,398,c20,c0_m]
$End

$Frame Окошки
$Back_picture = <255 255 255> post_1
Show_if Окно1.Работоспособность = Открыто
  bitmap[7,79,f1,f1_m]
  bitmap[109,75,oper,oper_m]
Show_if Окно1.Работоспособность = Закрыто and
        Окно1.В_Очереди >= 1
  bitmap[7,79,nf1,nf1_m]
  bitmap[109,75,oper,oper_m]
Show_if Окно1.Работоспособность = Закрыто and
        Окно1.В_Очереди = 0
  bitmap[7,79,nf1,nf1_m]
Show_if Окно2.Работоспособность = Открыто
  bitmap[7,164,f2,f2_m]
  bitmap[109,160,oper,oper_m]
Show_if Окно2.Работоспособность = Закрыто and
        Окно2.В_Очереди >= 1
  bitmap[7,164,nf2,nf2_m]
  bitmap[109,160,oper,oper_m]
Show_if Окно2.Работоспособность = Закрыто and
        Окно2.В_Очереди = 0
  bitmap[7,164,nf2,nf2_m]
Show_if Окно3.Работоспособность = Открыто
  bitmap[7,249,f3,f3_m]
  bitmap[109,246,oper,oper_m]
Show_if Окно3.Работоспособность = Закрыто and
        Окно3.В_Очереди >= 1
  bitmap[7,249,nf3,nf3_m]
  bitmap[109,246,oper,oper_m]
Show_if Окно3.Работоспособность = Закрыто and
        Окно3.В_Очереди = 0
  bitmap[7,249,nf3,nf3_m]
Show_if Окно4.Работоспособность = Открыто
  bitmap[7,334,f4,f4_m]
  bitmap[109,332,oper,oper_m]
Show_if Окно4.Работоспособность = Закрыто and
        Окно4.В_Очереди >= 1
  bitmap[7,334,nf4,nf4_m]
  bitmap[109,332,oper,oper_m]
Show_if Окно4.Работоспособность = Закрыто and
        Окно4.В_Очереди = 0
  bitmap[7,334,nf4,nf4_m]
Show_if Окно5.Работоспособность = Открыто
  bitmap[7,419,f5,f5_m]
  bitmap[109,418,oper,oper_m]
Show_if Окно5.Работоспособность = Закрыто and
        Окно5.В_Очереди >= 1
  bitmap[7,419,nf5,nf5_m]
  bitmap[109,418,oper,oper_m]
Show_if Окно5.Работоспособность = Закрыто and
        Окно5.В_Очереди = 0
  bitmap[7,419,nf5,nf5_m]
Show
  text[155,  32, 150, 15, transparent, <0 128 0>, < С_Рабочих_Дней.Количество_Дней]
  text[542,  99,  57, 12, transparent, <0 0 0>, = Окно1.В_Очереди]
  text[542, 184,  57, 12, transparent, <0 0 0>, = Окно2.В_Очереди]
  text[542, 270,  57, 12, transparent, <0 0 0>, = Окно3.В_Очереди]
  text[542, 356,  57, 12, transparent, <0 0 0>, = Окно4.В_Очереди]
  text[542, 442,  57, 12, transparent, <0 0 0>, = Окно5.В_Очереди]
  text[632,  99, 150, 12, transparent, <0 0 0>, < Окно1.Обслужено]
```

```
  text[632, 184, 150, 12, transparent, <0 0 0>, < Окно2.Обслужено]
  text[632, 270, 150, 12, transparent, <0 0 0>, < Окно3.Обслужено]
  text[632, 356, 150, 12, transparent, <0 0 0>, < Окно4.Обслужено]
  text[632, 442, 150, 12, transparent, <0 0 0>, < Окно5.Обслужено]
  text[729,  99,  65, 12, transparent, <0 0 0>, < С_Времени1.Время_На_Обработку]
  text[729, 184,  65, 12, transparent, <0 0 0>, < С_Времени2.Время_На_Обработку]
  text[729, 270,  65, 12, transparent, <0 0 0>, < С_Времени3.Время_На_Обработку]
  text[729, 356,  65, 12, transparent, <0 0 0>, < С_Времени4.Время_На_Обработку]
  text[729, 442,  65, 12, transparent, <0 0 0>, < С_Времени5.Время_На_Обработку]
Show_if [Окно1.В_Очереди >= 1] and [Из_Клиент1.Заявка = Rq1]
  text[200, 99, 320, 17, transparent, <0 0 0>, < 'Выдача до востребования']
Show_if [Окно2.В_Очереди >= 1] and [Из_Клиент2.Заявка = Rq1]
  text[200, 184, 320, 17, transparent, <0 0 0>, < 'Выдача до востребования']
Show_if [Окно3.В_Очереди >= 1] and [Из_Клиент3.Заявка = Rq1]
  text[200, 270, 320, 17, transparent, <0 0 0>, < 'Выдача до востребования']
Show_if [Окно4.В_Очереди >= 1] and [Из_Клиент4.Заявка = Rq1]
  text[200, 356, 320, 17, transparent, <0 0 0>, < 'Выдача до востребования']
Show_if [Окно5.В_Очереди >= 1] and [Из_Клиент5.Заявка = Rq1]
  text[200, 442, 320, 17, transparent, <0 0 0>, < 'Выдача до востребования']
Show_if [Окно1.В_Очереди >= 1] and [Из_Клиент1.Заявка = Rq2]
  text[200, 99, 320, 17, transparent, <0 0 0>, < 'Выплата пенсий']
Show_if [Окно2.В_Очереди >= 1] and [Из_Клиент2.Заявка = Rq2]
  text[200, 184, 320, 17, transparent, <0 0 0>, < 'Выплата пенсий']
Show_if [Окно3.В_Очереди >= 1] and [Из_Клиент3.Заявка = Rq2]
  text[200, 270, 320, 17, transparent, <0 0 0>, < 'Выплата пенсий']
Show_if [Окно4.В_Очереди >= 1] and [Из_Клиент4.Заявка = Rq2]
  text[200, 356, 320, 17, transparent, <0 0 0>, < 'Выплата пенсий']
Show_if [Окно5.В_Очереди >= 1] and [Из_Клиент5.Заявка = Rq2]
  text[200, 442, 320, 17, transparent, <0 0 0>, < 'Выплата пенсий']
Show_if [Окно1.В_Очереди >= 1] and [Из_Клиент1.Заявка = Rq3]
  text[200, 99, 320, 17, transparent, <0 0 0>, < 'Ксерокопия']
Show_if [Окно2.В_Очереди >= 1] and [Из_Клиент2.Заявка = Rq3]
  text[200, 184, 320, 17, transparent, <0 0 0>, < 'Ксерокопия']
Show_if [Окно3.В_Очереди >= 1] and [Из_Клиент3.Заявка = Rq3]
  text[200, 270, 320, 17, transparent, <0 0 0>, < 'Ксерокопия']
Show_if [Окно4.В_Очереди >= 1] and [Из_Клиент4.Заявка = Rq3]
  text[200, 356, 320, 17, transparent, <0 0 0>, < 'Ксерокопия']
Show_if [Окно5.В_Очереди >= 1] and [Из_Клиент5.Заявка = Rq3]
  text[200, 442, 320, 17, transparent, <0 0 0>, < 'Ксерокопия']
Show_if [Окно1.В_Очереди >= 1] and [Из_Клиент1.Заявка = Rq4]
  text[200, 99, 320, 17, transparent, <0 0 0>, < 'Оплата п/перевода']
Show_if [Окно2.В_Очереди >= 1] and [Из_Клиент2.Заявка = Rq4]
  text[200, 184, 320, 17, transparent, <0 0 0>, < 'Оплата п/перевода']
Show_if [Окно3.В_Очереди >= 1] and [Из_Клиент3.Заявка = Rq4]
  text[200, 270, 320, 17, transparent, <0 0 0>, < 'Оплата п/перевода']
Show_if [Окно4.В_Очереди >= 1] and [Из_Клиент4.Заявка = Rq4]
  text[200, 356, 320, 17, transparent, <0 0 0>, < 'Оплата п/перевода']
Show_if [Окно5.В_Очереди >= 1] and [Из_Клиент5.Заявка = Rq4]
  text[200, 442, 320, 17, transparent, <0 0 0>, < 'Оплата п/перевода']
Show_if [Окно1.В_Очереди >= 1] and [Из_Клиент1.Заявка = Rq5]
  text[200, 99, 320, 17, transparent, <0 0 0>, < 'Оплата т/перевода']
Show_if [Окно2.В_Очереди >= 1] and [Из_Клиент2.Заявка = Rq5]
  text[200, 184, 320, 17, transparent, <0 0 0>, < 'Оплата т/перевода']
Show_if [Окно3.В_Очереди >= 1] and [Из_Клиент3.Заявка = Rq5]
  text[200, 270, 320, 17, transparent, <0 0 0>, < 'Оплата т/перевода']
Show_if [Окно4.В_Очереди >= 1] and [Из_Клиент4.Заявка = Rq5]
  text[200, 356, 320, 17, transparent, <0 0 0>, < 'Оплата т/перевода']
Show_if [Окно5.В_Очереди >= 1] and [Из_Клиент5.Заявка = Rq5]
  text[200, 442, 320, 17, transparent, <0 0 0>, < 'Оплата т/перевода']
Show_if [Окно1.В_Очереди >= 1] and [Из_Клиент1.Заявка = Rq6]
  text[200, 99, 320, 17, transparent, <0 0 0>, < 'Отправление з/письма']
Show_if [Окно2.В_Очереди >= 1] and [Из_Клиент2.Заявка = Rq6]
  text[200, 184, 320, 17, transparent, <0 0 0>, < 'Отправление з/письма']
Show_if [Окно3.В_Очереди >= 1] and [Из_Клиент3.Заявка = Rq6]
  text[200, 270, 320, 17, transparent, <0 0 0>, < 'Отправление з/письма']
Show_if [Окно4.В_Очереди >= 1] and [Из_Клиент4.Заявка = Rq6]
  text[200, 356, 320, 17, transparent, <0 0 0>, < 'Отправление з/письма']
Show_if [Окно5.В_Очереди >= 1] and [Из_Клиент5.Заявка = Rq6]
  text[200, 442, 320, 17, transparent, <0 0 0>, < 'Отправление з/письма']
Show_if [Окно1.В_Очереди >= 1] and [Из_Клиент1.Заявка = Rq7]
  text[200, 99, 320, 17, transparent, <0 0 0>, < 'Отправление ц/б']
Show_if [Окно2.В_Очереди >= 1] and [Из_Клиент2.Заявка = Rq7]
  text[200, 184, 320, 17, transparent, <0 0 0>, < 'Отправление ц/б']
Show_if [Окно3.В_Очереди >= 1] and [Из_Клиент3.Заявка = Rq7]
```

```
  text[200, 270, 320, 17, transparent, <0 0 0>, < 'Отправление ц/б']
Show_if [Окно4.В_Очереди >= 1] and [Из_Клиент4.Заявка = Rq7]
  text[200, 356, 320, 17, transparent, <0 0 0>, < 'Отправление ц/б']
Show_if [Окно5.В_Очереди >= 1] and [Из_Клиент5.Заявка = Rq7]
  text[200, 442, 320, 17, transparent, <0 0 0>, < 'Отправление ц/б']
Show_if [Окно1.В_Очереди >= 1] and [Из_Клиент1.Заявка = Rq8]
  text[200, 99, 320, 17, transparent, <0 0 0>, < 'Перерыв']
Show_if [Окно2.В_Очереди >= 1] and [Из_Клиент2.Заявка = Rq8]
  text[200, 184, 320, 17, transparent, <0 0 0>, < 'Перерыв']
Show_if [Окно3.В_Очереди >= 1] and [Из_Клиент3.Заявка = Rq8]
  text[200, 270, 320, 17, transparent, <0 0 0>, < 'Перерыв']
Show_if [Окно4.В_Очереди >= 1] and [Из_Клиент4.Заявка = Rq8]
  text[200, 356, 320, 17, transparent, <0 0 0>, < 'Перерыв']
Show_if [Окно5.В_Очереди >= 1] and [Из_Клиент5.Заявка = Rq8]
  text[200, 442, 320, 17, transparent, <0 0 0>, < 'Перерыв']
Show_if [Окно1.В_Очереди >= 1] and [Из_Клиент1.Заявка = Rq9]
  text[200, 99, 320, 17, transparent, <0 0 0>, < 'Получение з/письма']
Show_if [Окно2.В_Очереди >= 1] and [Из_Клиент2.Заявка = Rq9]
  text[200, 184, 320, 17, transparent, <0 0 0>, < 'Получение з/письма']
Show_if [Окно3.В_Очереди >= 1] and [Из_Клиент3.Заявка = Rq9]
  text[200, 270, 320, 17, transparent, <0 0 0>, < 'Получение з/письма']
Show_if [Окно4.В_Очереди >= 1] and [Из_Клиент4.Заявка = Rq9]
  text[200, 356, 320, 17, transparent, <0 0 0>, < 'Получение з/письма']
Show_if [Окно5.В_Очереди >= 1] and [Из_Клиент5.Заявка = Rq9]
  text[200, 442, 320, 17, transparent, <0 0 0>, < 'Получение з/письма']
Show_if [Окно1.В_Очереди >= 1] and [Из_Клиент1.Заявка = Rq10]
  text[200, 99, 320, 17, transparent, <0 0 0>, < 'Получение п/б']
Show_if [Окно2.В_Очереди >= 1] and [Из_Клиент2.Заявка = Rq10]
  text[200, 184, 320, 17, transparent, <0 0 0>, < 'Получение п/б']
Show_if [Окно3.В_Очереди >= 1] and [Из_Клиент3.Заявка = Rq10]
  text[200, 270, 320, 17, transparent, <0 0 0>, < 'Получение п/б']
Show_if [Окно4.В_Очереди >= 1] and [Из_Клиент4.Заявка = Rq10]
  text[200, 356, 320, 17, transparent, <0 0 0>, < 'Получение п/б']
Show_if [Окно5.В_Очереди >= 1] and [Из_Клиент5.Заявка = Rq10]
  text[200, 442, 320, 17, transparent, <0 0 0>, < 'Получение п/б']
Show_if [Окно1.В_Очереди >= 1] and [Из_Клиент1.Заявка = Rq11]
  text[200, 99, 320, 17, transparent, <0 0 0>, < 'Получение посылки']
Show_if [Окно2.В_Очереди >= 1] and [Из_Клиент2.Заявка = Rq11]
  text[200, 184, 320, 17, transparent, <0 0 0>, < 'Получение посылки']
Show_if [Окно3.В_Очереди >= 1] and [Из_Клиент3.Заявка = Rq11]
  text[200, 270, 320, 17, transparent, <0 0 0>, < 'Получение посылки']
Show_if [Окно4.В_Очереди >= 1] and [Из_Клиент4.Заявка = Rq11]
  text[200, 356, 320, 17, transparent, <0 0 0>, < 'Получение посылки']
Show_if [Окно5.В_Очереди >= 1] and [Из_Клиент5.Заявка = Rq11]
  text[200, 442, 320, 17, transparent, <0 0 0>, < 'Получение посылки']
Show_if [Окно1.В_Очереди >= 1] and [Из_Клиент1.Заявка = Rq12]
  text[200, 99, 320, 17, transparent, <0 0 0>, < 'Получение ц/б']
Show_if [Окно2.В_Очереди >= 1] and [Из_Клиент2.Заявка = Rq12]
  text[200, 184, 320, 17, transparent, <0 0 0>, < 'Получение ц/б']
Show_if [Окно3.В_Очереди >= 1] and [Из_Клиент3.Заявка = Rq12]
  text[200, 270, 320, 17, transparent, <0 0 0>, < 'Получение ц/б']
Show_if [Окно4.В_Очереди >= 1] and [Из_Клиент4.Заявка = Rq12]
  text[200, 356, 320, 17, transparent, <0 0 0>, < 'Получение ц/б']
Show_if [Окно5.В_Очереди >= 1] and [Из_Клиент5.Заявка = Rq12]
  text[200, 442, 320, 17, transparent, <0 0 0>, < 'Получение ц/б']
Show_if [Окно1.В_Очереди >= 1] and [Из_Клиент1.Заявка = Rq13]
  text[200, 99, 320, 17, transparent, <0 0 0>, < 'Получение ц/письма']
Show_if [Окно2.В_Очереди >= 1] and [Из_Клиент2.Заявка = Rq13]
  text[200, 184, 320, 17, transparent, <0 0 0>, < 'Получение ц/письма']
Show_if [Окно3.В_Очереди >= 1] and [Из_Клиент3.Заявка = Rq13]
  text[200, 270, 320, 17, transparent, <0 0 0>, < 'Получение ц/письма']
Show_if [Окно4.В_Очереди >= 1] and [Из_Клиент4.Заявка = Rq13]
  text[200, 356, 320, 17, transparent, <0 0 0>, < 'Получение ц/письма']
Show_if [Окно5.В_Очереди >= 1] and [Из_Клиент5.Заявка = Rq13]
  text[200, 442, 320, 17, transparent, <0 0 0>, < 'Получение ц/письма']
Show_if [Окно1.В_Очереди >= 1] and [Из_Клиент1.Заявка = Rq14]
  text[200, 99, 320, 17, transparent, <0 0 0>, < 'Прием коммун. платежей']
Show_if [Окно2.В_Очереди >= 1] and [Из_Клиент2.Заявка = Rq14]
  text[200, 184, 320, 17, transparent, <0 0 0>, < 'Прием коммун. платежей']
Show_if [Окно3.В_Очереди >= 1] and [Из_Клиент3.Заявка = Rq14]
  text[200, 270, 320, 17, transparent, <0 0 0>, < 'Прием коммун. платежей']
Show_if [Окно4.В_Очереди >= 1] and [Из_Клиент4.Заявка = Rq14]
  text[200, 356, 320, 17, transparent, <0 0 0>, < 'Прием коммун. платежей']
Show_if [Окно5.В_Очереди >= 1] and [Из_Клиент5.Заявка = Rq14]
```

```
  text[200, 442, 320, 17, transparent, <0 0 0>, < 'Прием коммун. платежей']
Show_if [Окно1.В_Очереди >= 1] and [Из_Клиент1.Заявка = Rq15]
  text[200, 99, 320, 17, transparent, <0 0 0>, < 'Прием платы за ГТС']
Show_if [Окно2.В_Очереди >= 1] and [Из_Клиент2.Заявка = Rq15]
  text[200, 184, 320, 17, transparent, <0 0 0>, < 'Прием платы за ГТС']
Show_if [Окно3.В_Очереди >= 1] and [Из_Клиент3.Заявка = Rq15]
  text[200, 270, 320, 17, transparent, <0 0 0>, < 'Прием платы за ГТС']
Show_if [Окно4.В_Очереди >= 1] and [Из_Клиент4.Заявка = Rq15]
  text[200, 356, 320, 17, transparent, <0 0 0>, < 'Прием платы за ГТС']
Show_if [Окно5.В_Очереди >= 1] and [Из_Клиент5.Заявка = Rq15]
  text[200, 442, 320, 17, transparent, <0 0 0>, < 'Прием платы за ГТС']
Show_if [Окно1.В_Очереди >= 1] and [Из_Клиент1.Заявка = Rq16]
  text[200, 99, 320, 17, transparent, <0 0 0>, < 'Прием подписки']
Show_if [Окно2.В_Очереди >= 1] and [Из_Клиент2.Заявка = Rq16]
  text[200, 184, 320, 17, transparent, <0 0 0>, < 'Прием подписки']
Show_if [Окно3.В_Очереди >= 1] and [Из_Клиент3.Заявка = Rq16]
  text[200, 270, 320, 17, transparent, <0 0 0>, < 'Прием подписки']
Show_if [Окно4.В_Очереди >= 1] and [Из_Клиент4.Заявка = Rq16]
  text[200, 356, 320, 17, transparent, <0 0 0>, < 'Прием подписки']
Show_if [Окно5.В_Очереди >= 1] and [Из_Клиент5.Заявка = Rq16]
  text[200, 442, 320, 17, transparent, <0 0 0>, < 'Прием подписки']
Show_if [Окно1.В_Очереди >= 1] and [Из_Клиент1.Заявка = Rq17]
  text[200, 99, 320, 17, transparent, <0 0 0>, < 'Продажа ЗПО']
Show_if [Окно2.В_Очереди >= 1] and [Из_Клиент2.Заявка = Rq17]
  text[200, 184, 320, 17, transparent, <0 0 0>, < 'Продажа ЗПО']
Show_if [Окно3.В_Очереди >= 1] and [Из_Клиент3.Заявка = Rq17]
  text[200, 270, 320, 17, transparent, <0 0 0>, < 'Продажа ЗПО']
Show_if [Окно4.В_Очереди >= 1] and [Из_Клиент4.Заявка = Rq17]
  text[200, 356, 320, 17, transparent, <0 0 0>, < 'Продажа ЗПО']
Show_if [Окно5.В_Очереди >= 1] and [Из_Клиент5.Заявка = Rq17]
  text[200, 442, 320, 17, transparent, <0 0 0>, < 'Продажа ЗПО']
Show_if [Окно1.В_Очереди >= 1] and [Из_Клиент1.Заявка = Rq18]
  text[200, 99, 320, 17, transparent, <0 0 0>, < 'Справка о пенсии']
Show_if [Окно2.В_Очереди >= 1] and [Из_Клиент2.Заявка = Rq18]
  text[200, 184, 320, 17, transparent, <0 0 0>, < 'Справка о пенсии']
Show_if [Окно3.В_Очереди >= 1] and [Из_Клиент3.Заявка = Rq18]
  text[200, 270, 320, 17, transparent, <0 0 0>, < 'Справка о пенсии']
Show_if [Окно4.В_Очереди >= 1] and [Из_Клиент4.Заявка = Rq18]
  text[200, 356, 320, 17, transparent, <0 0 0>, < 'Справка о пенсии']
Show_if [Окно5.В_Очереди >= 1] and [Из_Клиент5.Заявка = Rq18]
  text[200, 442, 320, 17, transparent, <0 0 0>, < 'Справка о пенсии']
Show_if [Окно1.В_Очереди >= 1] and [Из_Клиент1.Заявка = Rq19]
  text[200, 99, 320, 17, transparent, <0 0 0>, < 'Справка о подписке']
Show_if [Окно2.В_Очереди >= 1] and [Из_Клиент2.Заявка = Rq19]
  text[200, 184, 320, 17, transparent, <0 0 0>, < 'Справка о подписке']
Show_if [Окно3.В_Очереди >= 1] and [Из_Клиент3.Заявка = Rq19]
  text[200, 270, 320, 17, transparent, <0 0 0>, < 'Справка о подписке']
Show_if [Окно4.В_Очереди >= 1] and [Из_Клиент4.Заявка = Rq19]
  text[200, 356, 320, 17, transparent, <0 0 0>, < 'Справка о подписке']
Show_if [Окно5.В_Очереди >= 1] and [Из_Клиент5.Заявка = Rq19]
  text[200, 442, 320, 17, transparent, <0 0 0>, < 'Справка о подписке']
Show_if Окно1.Занятость = Занят
  bitmap[125,73,others,others_m]
Show_if Окно2.Занятость = Занят
  bitmap[125,158,others,others_m]
Show_if Окно3.Занятость = Занят
  bitmap[125,244,others,others_m]
Show_if Окно4.Занятость = Занят
  bitmap[125,330,others,others_m]
Show_if Окно5.Занятость = Занят
  bitmap[125,416,others,others_m]
Show
//  active Кликнул1  [65,66,114,80]
//  active Кликнул2  [65,151,114,80]
//  active Кликнул3  [65,237,114,80]
//  active Кликнул4  [65,323,114,80]
//  active Кликнул5  [65,409,114,80]
Show_if Смотритель1.Разрешить = Да
  bitmap[См_X - 10, См_Y - 40, order_1, order_m]
  rect  [См_X - 10, См_Y - 5, См_Ширина, 20 * См_Высота - 2, <0 255 255>, <0 0 0>]
  text  [См_X, См_Y + 0 * См_Высота, См_Ширина, См_Высота, transparent, <0 0 0>, < 'Выдача до
востребования']
  text  [См_X, См_Y + 1 * См_Высота, См_Ширина, См_Высота, transparent, <0 0 0>, < 'Выплата
пенсий']
```

```
  text  [См_X, См_Y + 2 * См_Высота, См_Ширина, См_Высота, transparent, <0 0 0>, < 'Ксерокопия']
  text  [См_X, См_Y + 3 * См_Высота, См_Ширина, См_Высота, transparent, <0 0 0>, < 'Оплата
п/перевода']
  text  [См_X, См_Y + 4 * См_Высота, См_Ширина, См_Высота, transparent, <0 0 0>, < 'Оплата
т/перевода']
  text  [См_X, См_Y + 5 * См_Высота, См_Ширина, См_Высота, transparent, <0 0 0>, < 'Отправление
з/письма']
  text  [См_X, См_Y + 6 * См_Высота, См_Ширина, См_Высота, transparent, <0 0 0>, < 'Отправление
ц/б']
  text  [См_X, См_Y + 7 * См_Высота, См_Ширина, См_Высота, transparent, <0 0 0>, < 'Перерыв']
  text  [См_X, См_Y + 8 * См_Высота, См_Ширина, См_Высота, transparent, <0 0 0>, < 'Получение
з/письма']
  text  [См_X, См_Y + 9 * См_Высота, См_Ширина, См_Высота, transparent, <0 0 0>, < 'Получение
п/б']
  text  [См_X, См_Y + 10 * См_Высота, См_Ширина, См_Высота, transparent, <0 0 0>, < 'Получение
посылки']
  text  [См_X, См_Y + 11 * См_Высота, См_Ширина, См_Высота, transparent, <0 0 0>, < 'Получение
ц/б']
  text  [См_X, См_Y + 12 * См_Высота, См_Ширина, См_Высота, transparent, <0 0 0>, < 'Получение
ц/письма']
  text  [См_X, См_Y + 13 * См_Высота, См_Ширина, См_Высота, transparent, <0 0 0>, < 'Прием
коммун. платежей']
  text  [См_X, См_Y + 14 * См_Высота, См_Ширина, См_Высота, transparent, <0 0 0>, < 'Прием платы
за ГТС']
  text  [См_X, См_Y + 15 * См_Высота, См_Ширина, См_Высота, transparent, <0 0 0>, < 'Прием
подписки']
  text  [См_X, См_Y + 16 * См_Высота, См_Ширина, См_Высота, transparent, <0 0 0>, < 'Продажа
ЗПО']
  text  [См_X, См_Y + 17 * См_Высота, См_Ширина, См_Высота, transparent, <0 0 0>, < 'Справка о
пенсии']
  text  [См_X, См_Y + 18 * См_Высота, См_Ширина, См_Высота, transparent, <0 0 0>, < 'Справка о
подписке']
Show_if Смотритель2.Разрешить = Да
  bitmap[См_X - 10, См_Y - 40, order_2, order_m]
  rect  [См_X - 10, См_Y - 5, См_Ширина, 20 * См_Высота - 2, <0 255 255>, <0 0 0>]
  text  [См_X, См_Y + 0 * См_Высота, См_Ширина, См_Высота, transparent, <0 0 0>, 'Выдача до
востребования']
  text  [См_X, См_Y + 1 * См_Высота, См_Ширина, См_Высота, transparent, <0 0 0>, 'Выплата
пенсий']
  text  [См_X, См_Y + 2 * См_Высота, См_Ширина, См_Высота, transparent, <0 0 0>, 'Ксерокопия']
  text  [См_X, См_Y + 3 * См_Высота, См_Ширина, См_Высота, transparent, <0 0 0>, 'Оплата
п/перевода']
  text  [См_X, См_Y + 4 * См_Высота, См_Ширина, См_Высота, transparent, <0 0 0>, 'Оплата
т/перевода']
  text  [См_X, См_Y + 5 * См_Высота, См_Ширина, См_Высота, transparent, <0 0 0>, 'Отправление
з/письма']
  text  [См_X, См_Y + 6 * См_Высота, См_Ширина, См_Высота, transparent, <0 0 0>, 'Отправление
ц/б']
  text  [См_X, См_Y + 7 * См_Высота, См_Ширина, См_Высота, transparent, <0 0 0>, 'Перерыв']
  text  [См_X, См_Y + 8 * См_Высота, См_Ширина, См_Высота, transparent, <0 0 0>, 'Получение
з/письма']
  text  [См_X, См_Y + 9 * См_Высота, См_Ширина, См_Высота, transparent, <0 0 0>, 'Получение п/б']
  text  [См_X, См_Y + 10 * См_Высота, См_Ширина, См_Высота, transparent, <0 0 0>, 'Получение
посылки']
  text  [См_X, См_Y + 11 * См_Высота, См_Ширина, См_Высота, transparent, <0 0 0>, 'Получение
ц/б']
  text  [См_X, См_Y + 12 * См_Высота, См_Ширина, См_Высота, transparent, <0 0 0>, 'Получение
ц/письма']
  text  [См_X, См_Y + 13 * См_Высота, См_Ширина, См_Высота, transparent, <0 0 0>, 'Прием коммун.
платежей']
  text  [См_X, См_Y + 14 * См_Высота, См_Ширина, См_Высота, transparent, <0 0 0>, 'Прием платы за
ГТС']
  text  [См_X, См_Y + 15 * См_Высота, См_Ширина, См_Высота, transparent, <0 0 0>, 'Прием
подписки']
  text  [См_X, См_Y + 16 * См_Высота, См_Ширина, См_Высота, transparent, <0 0 0>, 'Продажа ЗПО']
  text  [См_X, См_Y + 17 * См_Высота, См_Ширина, См_Высота, transparent, <0 0 0>, 'Справка о
пенсии']
  text  [См_X, См_Y + 18 * См_Высота, См_Ширина, См_Высота, transparent, <0 0 0>, 'Справка о
подписке']
Show_if Смотритель3.Разрешить = Да
  bitmap[См_X - 10, См_Y - 40, order_3, order_m]
  rect  [См_X - 10, См_Y - 5, См_Ширина, 20 * См_Высота - 2, <0 255 255>, <0 0 0>]
  text  [См_X, См_Y + 0 * См_Высота, См_Ширина, См_Высота, transparent, <0 0 0>, 'Выдача до
востребования']
```

```
  text  [См_X, См_Y + 1 * См_Высота, См_Ширина, См_Высота, transparent, <0 0 0>, 'Выплата
пенсий']
  text  [См_X, См_Y + 2 * См_Высота, См_Ширина, См_Высота, transparent, <0 0 0>, 'Ксерокопия']
  text  [См_X, См_Y + 3 * См_Высота, См_Ширина, См_Высота, transparent, <0 0 0>, 'Оплата
п/перевода']
  text  [См_X, См_Y + 4 * См_Высота, См_Ширина, См_Высота, transparent, <0 0 0>, 'Оплата
т/перевода']
  text  [См_X, См_Y + 5 * См_Высота, См_Ширина, См_Высота, transparent, <0 0 0>, 'Отправление
з/письма']
  text  [См_X, См_Y + 6 * См_Высота, См_Ширина, См_Высота, transparent, <0 0 0>, 'Отправление
ц/б']
  text  [См_X, См_Y + 7 * См_Высота, См_Ширина, См_Высота, transparent, <0 0 0>, 'Перерыв']
  text  [См_X, См_Y + 8 * См_Высота, См_Ширина, См_Высота, transparent, <0 0 0>, 'Получение
з/письма']
  text  [См_X, См_Y + 9 * См_Высота, См_Ширина, См_Высота, transparent, <0 0 0>, 'Получение п/б']
  text  [См_X, См_Y + 10 * См_Высота, См_Ширина, См_Высота, transparent, <0 0 0>, 'Получение
посылки']
  text  [См_X, См_Y + 11 * См_Высота, См_Ширина, См_Высота, transparent, <0 0 0>, 'Получение
ц/б']
  text  [См_X, См_Y + 12 * См_Высота, См_Ширина, См_Высота, transparent, <0 0 0>, 'Получение
ц/письма']
  text  [См_X, См_Y + 13 * См_Высота, См_Ширина, См_Высота, transparent, <0 0 0>, 'Прием коммун.
платежей']
  text  [См_X, См_Y + 14 * См_Высота, См_Ширина, См_Высота, transparent, <0 0 0>, 'Прием платы за
ГТС']
  text  [См_X, См_Y + 15 * См_Высота, См_Ширина, См_Высота, transparent, <0 0 0>, 'Прием
подписки']
  text  [См_X, См_Y + 16 * См_Высота, См_Ширина, См_Высота, transparent, <0 0 0>, 'Продажа ЗПО']
  text  [См_X, См_Y + 17 * См_Высота, См_Ширина, См_Высота, transparent, <0 0 0>, 'Справка о
пенсии']
  text  [См_X, См_Y + 18 * См_Высота, См_Ширина, См_Высота, transparent, <0 0 0>, 'Справка о
подписке']
Show_if Смотритель4.Разрешить = Да
  bitmap[См_X - 10, См_Y - 40, order_4, order_m]
  rect  [См_X - 10, См_Y - 5, См_Ширина, 20 * См_Высота - 2, <0 255 255>, <0 0 0>]
  text  [См_X, См_Y + 0 * См_Высота, См_Ширина, См_Высота, transparent, <0 0 0>, 'Выдача до
востребования']
  text  [См_X, См_Y + 1 * См_Высота, См_Ширина, См_Высота, transparent, <0 0 0>, 'Выплата
пенсий']
  text  [См_X, См_Y + 2 * См_Высота, См_Ширина, См_Высота, transparent, <0 0 0>, 'Ксерокопия']
  text  [См_X, См_Y + 3 * См_Высота, См_Ширина, См_Высота, transparent, <0 0 0>, 'Оплата
п/перевода']
  text  [См_X, См_Y + 4 * См_Высота, См_Ширина, См_Высота, transparent, <0 0 0>, 'Оплата
т/перевода']
  text  [См_X, См_Y + 5 * См_Высота, См_Ширина, См_Высота, transparent, <0 0 0>, 'Отправление
з/письма']
  text  [См_X, См_Y + 6 * См_Высота, См_Ширина, См_Высота, transparent, <0 0 0>, 'Отправление
ц/б']
  text  [См_X, См_Y + 7 * См_Высота, См_Ширина, См_Высота, transparent, <0 0 0>, 'Перерыв']
  text  [См_X, См_Y + 8 * См_Высота, См_Ширина, См_Высота, transparent, <0 0 0>, 'Получение
з/письма']
  text  [См_X, См_Y + 9 * См_Высота, См_Ширина, См_Высота, transparent, <0 0 0>, 'Получение п/б']
  text  [См_X, См_Y + 10 * См_Высота, См_Ширина, См_Высота, transparent, <0 0 0>, 'Получение
посылки']
  text  [См_X, См_Y + 11 * См_Высота, См_Ширина, См_Высота, transparent, <0 0 0>, 'Получение
ц/б']
  text  [См_X, См_Y + 12 * См_Высота, См_Ширина, См_Высота, transparent, <0 0 0>, 'Получение
ц/письма']
  text  [См_X, См_Y + 13 * См_Высота, См_Ширина, См_Высота, transparent, <0 0 0>, 'Прием коммун.
платежей']
  text  [См_X, См_Y + 14 * См_Высота, См_Ширина, См_Высота, transparent, <0 0 0>, 'Прием платы за
ГТС']
  text  [См_X, См_Y + 15 * См_Высота, См_Ширина, См_Высота, transparent, <0 0 0>, 'Прием
подписки']
  text  [См_X, См_Y + 16 * См_Высота, См_Ширина, См_Высота, transparent, <0 0 0>, 'Продажа ЗПО']
  text  [См_X, См_Y + 17 * См_Высота, См_Ширина, См_Высота, transparent, <0 0 0>, 'Справка о
пенсии']
  text  [См_X, См_Y + 18 * См_Высота, См_Ширина, См_Высота, transparent, <0 0 0>, 'Справка о
подписке']
Show_if Смотритель5.Разрешить = Да
  bitmap[См_X - 10, См_Y - 40, order_5, order_m]
  rect  [См_X - 10, См_Y - 5, См_Ширина, 20 * См_Высота - 2, <0 255 255>, <0 0 0>]
  text  [См_X, См_Y + 0 * См_Высота, См_Ширина, См_Высота, transparent, <0 0 0>, 'Выдача до
востребования']
```

```
  text  [См_X, См_Y + 1 * См_Высота, См_Ширина, См_Высота, transparent, <0 0 0>, 'Выплата
пенсий']
  text  [См_X, См_Y + 2 * См_Высота, См_Ширина, См_Высота, transparent, <0 0 0>, 'Ксерокопия']
  text  [См_X, См_Y + 3 * См_Высота, См_Ширина, См_Высота, transparent, <0 0 0>, 'Оплата
п/перевода']
  text  [См_X, См_Y + 4 * См_Высота, См_Ширина, См_Высота, transparent, <0 0 0>, 'Оплата
т/перевода']
  text  [См_X, См_Y + 5 * См_Высота, См_Ширина, См_Высота, transparent, <0 0 0>, 'Отправление
з/письма']
  text  [См_X, См_Y + 6 * См_Высота, См_Ширина, См_Высота, transparent, <0 0 0>, 'Отправление
ц/б']
  text  [См_X, См_Y + 7 * См_Высота, См_Ширина, См_Высота, transparent, <0 0 0>, 'Перерыв']
  text  [См_X, См_Y + 8 * См_Высота, См_Ширина, См_Высота, transparent, <0 0 0>, 'Получение
з/письма']
  text  [См_X, См_Y + 9 * См_Высота, См_Ширина, См_Высота, transparent, <0 0 0>, 'Получение п/б']
  text  [См_X, См_Y + 10 * См_Высота, См_Ширина, См_Высота, transparent, <0 0 0>, 'Получение
посылки']
  text  [См_X, См_Y + 11 * См_Высота, См_Ширина, См_Высота, transparent, <0 0 0>, 'Получение
ц/б']
  text  [См_X, См_Y + 12 * См_Высота, См_Ширина, См_Высота, transparent, <0 0 0>, 'Получение
ц/письма']
  text  [См_X, См_Y + 13 * См_Высота, См_Ширина, См_Высота, transparent, <0 0 0>, 'Прием коммун.
платежей']
  text  [См_X, См_Y + 14 * См_Высота, См_Ширина, См_Высота, transparent, <0 0 0>, 'Прием платы за
ГТС']
  text  [См_X, См_Y + 15 * См_Высота, См_Ширина, См_Высота, transparent, <0 0 0>, 'Прием
подписки']
  text  [См_X, См_Y + 16 * См_Высота, См_Ширина, См_Высота, transparent, <0 0 0>, 'Продажа ЗПО']
  text  [См_X, См_Y + 17 * См_Высота, См_Ширина, См_Высота, transparent, <0 0 0>, 'Справка о
пенсии']
  text  [См_X, См_Y + 18 * См_Высота, См_Ширина, См_Высота, transparent, <0 0 0>, 'Справка о
подписке']
$End

$Frame Статистика
$Back_picture = <0 255 0> post_2
Show
  rect [Начало1_X , Начало_Y_R + Высота * 0, Ширина1, Высота + 1, transparent, <0 0 0>]
  text [Начало_Текст1, Начало_Y_T + Высота * 0, 320, 17, transparent, <0 0 0>, < 'Выдача до
востребования']
  rect [Начало2_X , Начало_Y_R + Высота * 0, Ширина2, Высота + 1, transparent, <0 0 0>]
  text [Начало_Текст2, Начало_Y_T + Высота * 0, 167, 17, transparent, <0 0 0>, =
Count1.Количество]
  rect [Начало1_X , Начало_Y_R + Высота * 1, Ширина1, Высота + 1, transparent, <0 0 0>]
  text [Начало_Текст1, Начало_Y_T + Высота * 1, 320, 17, transparent, <0 0 0>, < 'Выплата
пенсий']
  rect [Начало2_X , Начало_Y_R + Высота * 1, Ширина2, Высота + 1, transparent, <0 0 0>]
  text [Начало_Текст2, Начало_Y_T + Высота * 1, 167, 17, transparent, <0 0 0>, =
Count2.Количество]
  rect [Начало1_X , Начало_Y_R + Высота * 2, Ширина1, Высота + 1, transparent, <0 0 0>]
  text [Начало_Текст1, Начало_Y_T + Высота * 2, 320, 17, transparent, <0 0 0>, < 'Ксерокопия']
  rect [Начало2_X , Начало_Y_R + Высота * 2, Ширина2, Высота + 1, transparent, <0 0 0>]
  text [Начало_Текст2, Начало_Y_T + Высота * 2, 167, 17, transparent, <0 0 0>, =
Count3.Количество]
  rect [Начало1_X , Начало_Y_R + Высота * 3, Ширина1, Высота + 1, transparent, <0 0 0>]
  text [Начало_Текст1, Начало_Y_T + Высота * 3, 320, 17, transparent, <0 0 0>, < 'Оплата
п/перевода']
  rect [Начало2_X , Начало_Y_R + Высота * 3, Ширина2, Высота + 1, transparent, <0 0 0>]
  text [Начало_Текст2, Начало_Y_T + Высота * 3, 167, 17, transparent, <0 0 0>, =
Count4.Количество]
  rect [Начало1_X , Начало_Y_R + Высота * 4, Ширина1, Высота + 1, transparent, <0 0 0>]
  text [Начало_Текст1, Начало_Y_T + Высота * 4, 320, 17, transparent, <0 0 0>, < 'Оплата
т/перевода']
  rect [Начало2_X , Начало_Y_R + Высота * 4, Ширина2, Высота + 1, transparent, <0 0 0>]
  text [Начало_Текст2, Начало_Y_T + Высота * 4, 167, 17, transparent, <0 0 0>, =
Count5.Количество]
  rect [Начало1_X , Начало_Y_R + Высота * 5, Ширина1, Высота + 1, transparent, <0 0 0>]
  text [Начало_Текст1, Начало_Y_T + Высота * 5, 320, 17, transparent, <0 0 0>, < 'Отправление
з/письма']
  rect [Начало2_X , Начало_Y_R + Высота * 5, Ширина2, Высота + 1, transparent, <0 0 0>]
  text [Начало_Текст2, Начало_Y_T + Высота * 5, 167, 17, transparent, <0 0 0>, =
Count6.Количество]
  rect [Начало1_X , Начало_Y_R + Высота * 6, Ширина1, Высота + 1, transparent, <0 0 0>]
```

```
  text [Начало_Текст1, Начало_Y_T + Высота * 6, 320, 17, transparent, <0 0 0>, < 'Отправление
ц/б']
  rect [Начало2_X , Начало_Y_R + Высота * 6, Ширина2, Высота + 1, transparent, <0 0 0>]
  text [Начало_Текст2, Начало_Y_T + Высота * 6, 167, 17, transparent, <0 0 0>, =
Count7.Количество]
  rect [Начало1_X , Начало_Y_R + Высота * 7, Ширина1, Высота + 1, transparent, <0 0 0>]
  text [Начало_Текст1, Начало_Y_T + Высота * 7, 320, 17, transparent, <0 0 0>, < 'Перерыв']
  rect [Начало2_X , Начало_Y_R + Высота * 7, Ширина2, Высота + 1, transparent, <0 0 0>]
  text [Начало_Текст2, Начало_Y_T + Высота * 7, 167, 17, transparent, <0 0 0>, =
Count8.Количество]
  rect [Начало1_X , Начало_Y_R + Высота * 8, Ширина1, Высота + 1, transparent, <0 0 0>]
  text [Начало_Текст1, Начало_Y_T + Высота * 8, 320, 17, transparent, <0 0 0>, < 'Получение
з/письма']
  rect [Начало2_X , Начало_Y_R + Высота * 8, Ширина2, Высота + 1, transparent, <0 0 0>]
  text [Начало_Текст2, Начало_Y_T + Высота * 8, 167, 17, transparent, <0 0 0>, =
Count9.Количество]
  rect [Начало1_X , Начало_Y_R + Высота * 9, Ширина1, Высота + 1, transparent, <0 0 0>]
  text [Начало_Текст1, Начало_Y_T + Высота * 9, 320, 17, transparent, <0 0 0>, < 'Получение п/б']
  rect [Начало2_X , Начало_Y_R + Высота * 9, Ширина2, Высота + 1, transparent, <0 0 0>]
  text [Начало_Текст2, Начало_Y_T + Высота * 9, 167, 17, transparent, <0 0 0>, =
Count10.Количество]
  rect [Начало1_X , Начало_Y_R + Высота * 10, Ширина1, Высота + 1, transparent, <0 0 0>]
  text [Начало_Текст1, Начало_Y_T + Высота * 10, 320, 17, transparent, <0 0 0>, < 'Получение
посылки']
  rect [Начало2_X , Начало_Y_R + Высота * 10, Ширина2, Высота + 1, transparent, <0 0 0>]
  text [Начало_Текст2, Начало_Y_T + Высота * 10, 167, 17, transparent, <0 0 0>, =
Count11.Количество]
  rect [Начало1_X , Начало_Y_R + Высота * 11, Ширина1, Высота + 1, transparent, <0 0 0>]
  text [Начало_Текст1, Начало_Y_T + Высота * 11, 320, 17, transparent, <0 0 0>, < 'Получение
ц/б']
  rect [Начало2_X , Начало_Y_R + Высота * 11, Ширина2, Высота + 1, transparent, <0 0 0>]
  text [Начало_Текст2, Начало_Y_T + Высота * 11, 167, 17, transparent, <0 0 0>, =
Count12.Количество]
  rect [Начало1_X , Начало_Y_R + Высота * 12, Ширина1, Высота + 1, transparent, <0 0 0>]
  text [Начало_Текст1, Начало_Y_T + Высота * 12, 320, 17, transparent, <0 0 0>, < 'Получение
ц/письма']
  rect [Начало2_X , Начало_Y_R + Высота * 12, Ширина2, Высота + 1, transparent, <0 0 0>]
  text [Начало_Текст2, Начало_Y_T + Высота * 12, 167, 17, transparent, <0 0 0>, =
Count13.Количество]
  rect [Начало1_X , Начало_Y_R + Высота * 13, Ширина1, Высота + 1, transparent, <0 0 0>]
  text [Начало_Текст1, Начало_Y_T + Высота * 13, 320, 17, transparent, <0 0 0>, < 'Прием коммун.
платежей']
  rect [Начало2_X , Начало_Y_R + Высота * 13, Ширина2, Высота + 1, transparent, <0 0 0>]
  text [Начало_Текст2, Начало_Y_T + Высота * 13, 167, 17, transparent, <0 0 0>, =
Count14.Количество]
  rect [Начало1_X , Начало_Y_R + Высота * 14, Ширина1, Высота + 1, transparent, <0 0 0>]
  text [Начало_Текст1, Начало_Y_T + Высота * 14, 320, 17, transparent, <0 0 0>, < 'Прием платы за
ГТС']
  rect [Начало2_X , Начало_Y_R + Высота * 14, Ширина2, Высота + 1, transparent, <0 0 0>]
  text [Начало_Текст2, Начало_Y_T + Высота * 14, 167, 17, transparent, <0 0 0>, =
Count15.Количество]
  rect [Начало1_X , Начало_Y_R + Высота * 15, Ширина1, Высота + 1, transparent, <0 0 0>]
  text [Начало_Текст1, Начало_Y_T + Высота * 15, 320, 17, transparent, <0 0 0>, < 'Прием
подписки']
  rect [Начало2_X , Начало_Y_R + Высота * 15, Ширина2, Высота + 1, transparent, <0 0 0>]
  text [Начало_Текст2, Начало_Y_T + Высота * 15, 167, 17, transparent, <0 0 0>, =
Count16.Количество]
  rect [Начало1_X , Начало_Y_R + Высота * 16, Ширина1, Высота + 1, transparent, <0 0 0>]
  text [Начало_Текст1, Начало_Y_T + Высота * 16, 320, 17, transparent, <0 0 0>, < 'Продажа ЗПО']
  rect [Начало2_X , Начало_Y_R + Высота * 16, Ширина2, Высота + 1, transparent, <0 0 0>]
  text [Начало_Текст2, Начало_Y_T + Высота * 16, 167, 17, transparent, <0 0 0>, =
Count17.Количество]
  rect [Начало1_X , Начало_Y_R + Высота * 17, Ширина1, Высота + 1, transparent, <0 0 0>]
  text [Начало_Текст1, Начало_Y_T + Высота * 17, 320, 17, transparent, <0 0 0>, < 'Справка о
пенсии']
  rect [Начало2_X , Начало_Y_R + Высота * 17, Ширина2, Высота + 1, transparent, <0 0 0>]
  text [Начало_Текст2, Начало_Y_T + Высота * 17, 167, 17, transparent, <0 0 0>, =
Count18.Количество]
  rect [Начало1_X , Начало_Y_R + Высота * 18, Ширина1, Высота + 1, transparent, <0 0 0>]
  text [Начало_Текст1, Начало_Y_T + Высота * 18, 320, 17, transparent, <0 0 0>, < 'Справка о
подписке']
  rect [Начало2_X , Начало_Y_R + Высота * 18, Ширина2, Высота + 1, transparent, <0 0 0>]
  text [Начало_Текст2, Начало_Y_T + Высота * 18, 167, 17, transparent, <0 0 0>, =
Count19.Количество]
```

```
$End
```

Postoffice.smr (Прогон):

```
Model_name = Postoffice

Resource_file  = Postoffice
Frame_file     = Postoffice
Statistic_file = Postoffice
Results_file   = Postoffice
Trace_file     = Postoffice
Show_mode      = NoShow
Frame_number   = 1
Show_rate      = 50.0

Terminate_if С_Рабочих_Дней.Количество_Дней = 2
Break_point  Завершение_моделирования С_Рабочих_Дней.Количество_Дней = Год
```

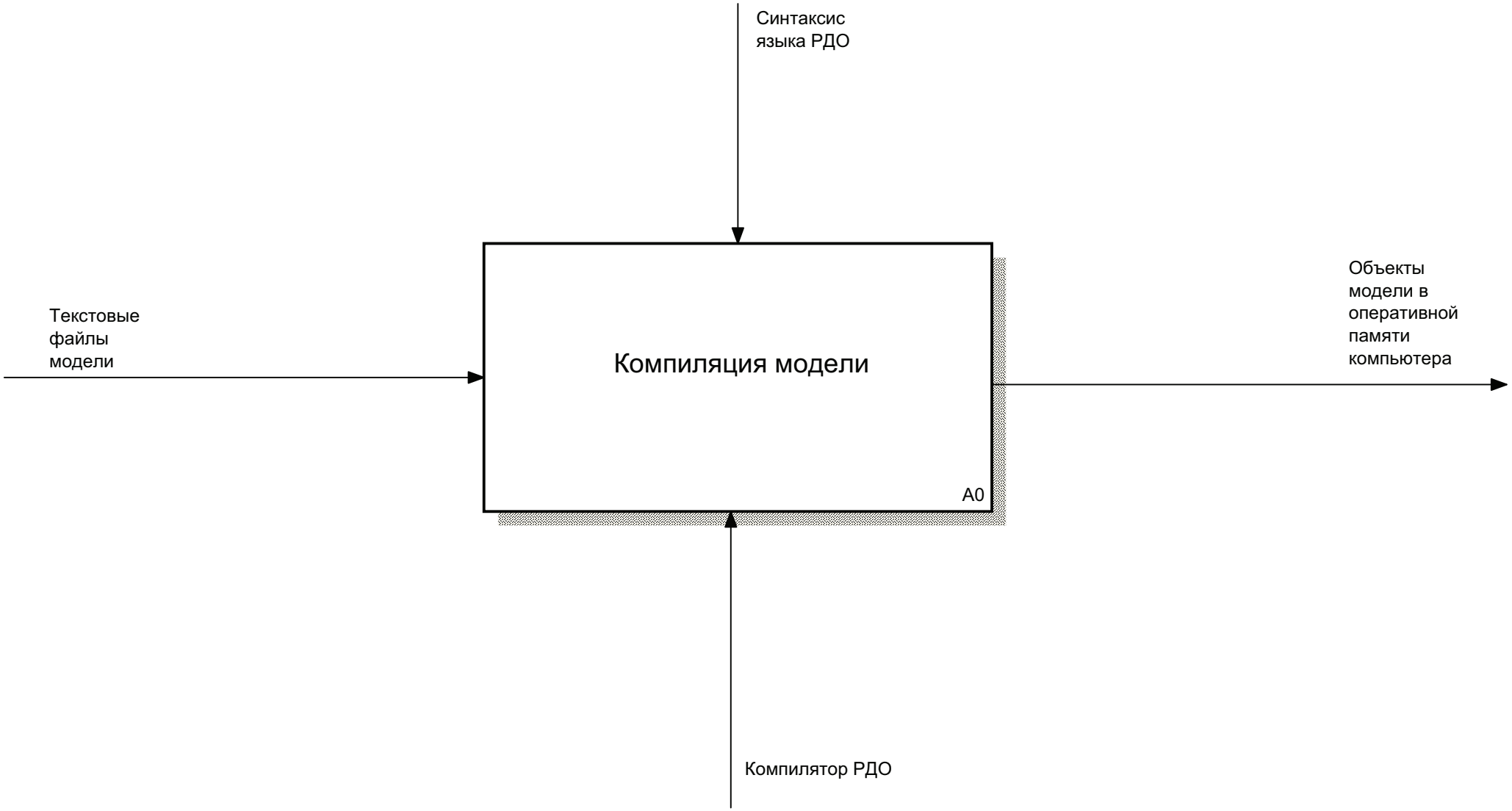
Postoffice.pmd (Показатели):

```
$Results
        Время_Пребывания_В_Очереди : watch_value Клиенты NoCheck Клиенты.Время_В_Очереди
        Количество_В_Очереди_1     : watch_par Окно1.В_Очереди
        Количество_В_Очереди_2     : watch_par Окно2.В_Очереди
        Количество_В_Очереди_3     : watch_par Окно3.В_Очереди
        Количество_В_Очереди_4     : watch_par Окно4.В_Очереди
        Количество_В_Очереди_5     : watch_par Окно5.В_Очереди
        Количество_Обслуженных1    : get_value Окно1.Обслужено
        Количество_Обслуженных2    : get_value Окно2.Обслужено
        Количество_Обслуженных3    : get_value Окно3.Обслужено
        Количество_Обслуженных4    : get_value Окно4.Обслужено
        Количество_Обслуженных5    : get_value Окно5.Обслужено
        Количество_Обслуженных     : get_value Окно5.Обслужено + Окно4.Обслужено + Окно3.Обслужено
+ Окно2.Обслужено + Окно1.Обслужено
        Число_Клиентов_Без_Очереди : watch_quant Клиенты Клиенты.Время_В_Очереди = 0 and
Клиенты.Состояние = Ушел
        Время_моделирования        : get_value Time_now
        Занятость_Оператора_з_1    : watch_state Окно1.Занятость = Занят
        Занятость_Оператора_з_2    : watch_state Окно2.Занятость = Занят
        Занятость_Оператора_з_3    : watch_state Окно3.Занятость = Занят
        Занятость_Оператора_з_4    : watch_state Окно4.Занятость = Занят
        Занятость_Оператора_з_5    : watch_state Окно5.Занятость = Занят
        Занятость_Оператора_с_1    : watch_state Окно1.Занятость = Свободен and
Окно1.Работоспособность = Открыто
        Занятость_Оператора_с_2    : watch_state Окно2.Занятость = Свободен and
Окно2.Работоспособность = Открыто
        Занятость_Оператора_с_3    : watch_state Окно3.Занятость = Свободен and
Окно3.Работоспособность = Открыто
        Занятость_Оператора_с_4    : watch_state Окно4.Занятость = Свободен and
Окно4.Работоспособность = Открыто
        Занятость_Оператора_с_5    : watch_state Окно5.Занятость = Свободен and
Окно5.Работоспособность = Открыто
        Время_Обработки_В_Окне1    : watch_par С_Времени1.Время_На_Обработку
        Время_Обработки_В_Окне2    : watch_par С_Времени2.Время_На_Обработку
        Время_Обработки_В_Окне3    : watch_par С_Времени3.Время_На_Обработку
        Время_Обработки_В_Окне4    : watch_par С_Времени4.Время_На_Обработку
        Время_Обработки_В_Окне5    : watch_par С_Времени5.Время_На_Обработку
        Потерянных                 : get_value Потерянные.Количество
        Rq1                        : get_value Count1.Количество
        Rq2                        : get_value Count2.Количество
        Rq3                        : get_value Count3.Количество
        Rq4                        : get_value Count4.Количество
        Rq5                        : get_value Count5.Количество
        Rq6                        : get_value Count6.Количество
        Rq7                        : get_value Count7.Количество
        Rq8                        : get_value Count8.Количество
        Rq9                        : get_value Count9.Количество
```

```
        Rq10                    : get_value Count10.Количество
        Rq11                    : get_value Count11.Количество
        Rq12                    : get_value Count12.Количество
        Rq13                    : get_value Count13.Количество
        Rq14                    : get_value Count14.Количество
        Rq15                    : get_value Count15.Количество
        Rq16                    : get_value Count16.Количество
        Rq17                    : get_value Count17.Количество
        Rq18                    : get_value Count18.Количество
        Rq19                    : get_value Count19.Количество
$End
```

Приложение 3. Функциональная диаграмма компиляции модели на языке РДО. Уровень А-0.

USED AT:	AUTHOR: Лущан Дмитрий	DATE: 15.06.2009		WORKING	READER	DATE	CONTEXT:
	PROJECT: Механизм логического вывода в РДО на основе иерархических логик	REV: 20.12.2009		DRAFT			TOP
				RECOMMENDED			
	NOTES: 1 2 3 4 5 6 7 8 9 10		■	PUBLICATION			



NODE:	TITLE:	NUMBER:
A-0	Компиляция модели	