

## Лабораторная работа № 4

### Часть 1. Основы моделирования в Arena

#### Цель работы

Ознакомление с интерфейсом Arena 16.0. Изучение основных блоков. Построение простейшей модели. Изучение структуры отчетов Arena 16.0 и возможности их настройки. Анимация ресурсов и построение графиков.

#### Интерфейс Arena

Рабочая область моделирования в Arena состоит из трех областей (на рис. 1):

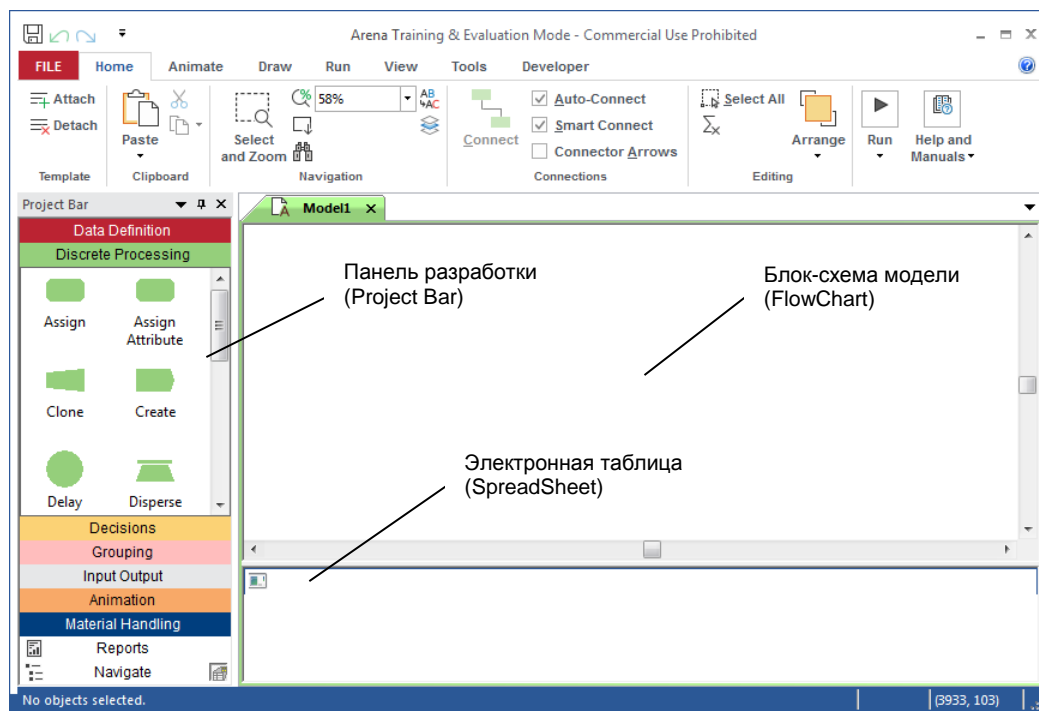
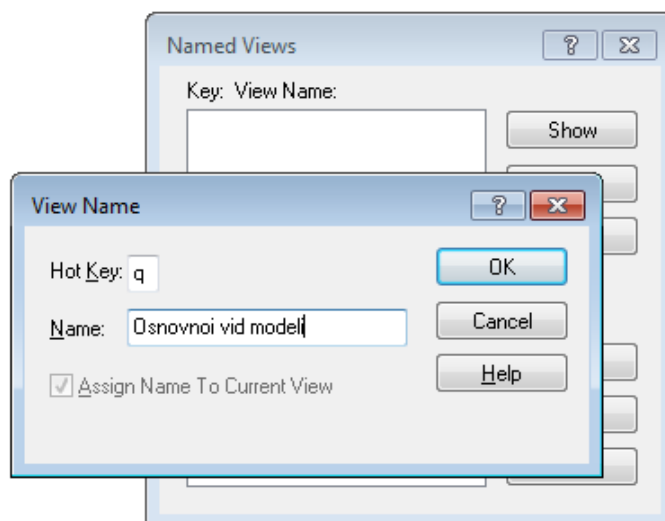


Рис. 1. Рабочая область Arena 9.0

- **Project Bar** содержит панели с объектами:
  - *Discrete Processing* – библиотека базовых функциональных модулей, для добавления в Project Bar других библиотек модулей необходимо из контекстного меню Project Bar выбрать команду **Attach...**;
  - *Reports Panel* – панель для просмотра отчетов моделирования;
  - *Navigate Panel* – обеспечивает навигацию по различным видам модели, включая субмодели. Для сохранения нового вида модели необходимо выбрать **View ⇒ Named Views...**?, в открывшемся диалоговом окне можно дать имя виду, а также назначить «быструю» клавишу для перехода (Рис. 2).
- **FlowChart** – содержит графические элементы модели, модули, анимацию;
- **SpreadSheet** – отображает данные модели (время, издержки) и др. параметры.

Описание панелей инструментов Arena приведено в Приложении 1. Описание главного меню Arena приведено в Приложении 2. Назначение основных функциональных модулей приведено в приложении 3.

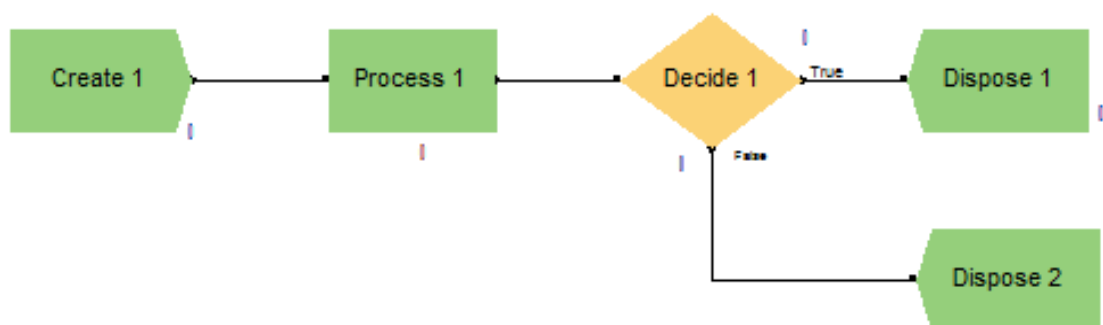


**Рис. 2. Добавление Вида к модели**

### Порядок создания модели в Arena

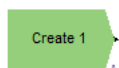
Рассмотрим работу налогового инспектора, который рассматривает налоговые декларации в течение некоторого времени и принимает решение об их приеме или возврате. Налоговые декларации в таком примере - транзакты, налоговый инспектор – ресурс.

Работу налогового инспектора можно смоделировать в виде блок-схемы рис. 3. Рассмотрим подробнее этапы ее создания.



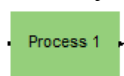
**Рис. 3. Создание модели**

### Создание транзактов



Модуль **Create** (панель *Discrete Processing*) используется для генерации транзактов. Служит стартовой точкой для транзактов в имитационной модели. Транзакты в данном модуле генерируются на основе заданного закона временного распределения или расписания. Затем транзакты покидают модуль, чтобы начать свою обработку в модели. Также в этом блоке определяется тип генерируемого транзакта.

1. Перетащите модуль **Create** в область модели **FlowChart**. По умолчанию модулю будет присвоено имя **Create 1**.



Модуль **Process** (панель *Discrete Processing*) осуществляет имитацию процесса обработки транзакта. Время, затрачиваемое на процесс обработки транзакта, может быть задано фиксированной величиной или подчиняться распределению. Также в этом модуле задаются ресурсы, с помощью которых и выполняется обработка транзакта.

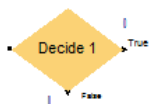
2. Перетащите модуль **Process** в область модели (модуль **Create** должен быть выделен для автоматического соединения этих модулей). По умолчанию модулю будет присвоено имя **Process 1**.

Если соединения между блоками **Create 1** и **Process 1** не появилось, выберите **Home**



⇒ **Connections** ⇒ **Auto-Connect** или нажмите кнопку **Connect** на вкладке **Home**. Соедините модули, начав на точке выхода (►) модуля **Create 1** и заканчивая на точке входа (■) модуля **Process 1**.

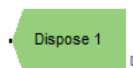
### Принятие решения



Модуль **Decide** (панель *Decisions*) имитирует процесс выбора из двух или более альтернатив. Он включает опции для принятия решения, базирующегося на одном или более условиях, а также на основе вероятностного распределения. Условия выбора альтернативы могут основываться на значениях атрибутов, значениях переменных, типах транзактов или на выражениях.

3. Если вы используете **Auto-connect**, убедитесь, что модуль **Process 1**, является активным.
4. Перетащите модуль **Decide** (панель *Decisions*) в окно модели, как показано на рис. 3.

### Удаление транзактов из модели



Модуль **Dispose** (панель *Discrete Processing*) используется как конечная точка для транзактов в модели. Попадая в этот модуль, транзакты уничтожаются. Перед уничтожением транзактов статистические данные об их обработке в модели записываются в базу данных модели.

5. Выберите модуль **Dispose**.
6. Перетащите модуль **Dispose** в окно модели и поместите его справа от модуля **Decide 1**. При этом модуль **Dispose 1** автоматически присоединиться к первому выходу (TRUE) модуля **Decide 1**.
7. Для добавления второго модуля **Dispose**, снова выделите модуль **Decide**, перетащите модуль **Dispose** из панели *Discrete Processing* в окно модели и поместите его снизу модуля **Decide 1**. При этом произойдет автоматическое соединение модуля **Dispose 2** с нижним выходом модуля **Decide 1**.

После составления блок-схемы процесса (как на рис. 3) необходимо описать данные, связанные с модулями, включая имя модуля и информацию, которая будет использоваться при моделировании.

### Описание параметров модели

#### Создание транзактов (модуль **Create**)

Назовем модуль **Create 1** - *Generaciya potoka nalogovih deklaracij*, данные этого модуля описывают, как часто будут создаваться транзакты (в нашей модели каждые 2 часа), тип транзактов *Zayavleniya*. (рис. 4).

Рис. 4. Параметры модуля Create

1. Дважды щелкните мышью на модуле **Create 1**, для открытия диалогового окна параметров этого модуля. В поле *Name* введите ***Generaciya potoka nalogovih deklaracii***.
2. В поле *Entity Type* введите ***Zayavleniya***, для названия генерируемых транзактов.
3. В поле *Value* введите **2**.

#### Рассмотрение заявлений (модуль Process)

Для рассмотрения налоговой декларации налоговому инспектору требуется время, поэтому транзакты будут задерживаться именно в этом модуле блок-схемы. Также этот процесс предполагает наличие ресурса, который исполняет эту деятельность (этим ресурсом будет налоговый инспектор).

Для описания задержки времени, необходимо осуществить отобразить случайность, которая присуща большинству процессов. Очень часто, для работы, выполняемой людьми, хорошее приближение обеспечивает треугольное распределение (рис. 5). Сначала вы определяете минимальное время ( $a$ ), в течение которого может быть сделана работа, затем оптимальную (желаемую) величину этого времени ( $m$ ), и наконец, максимальную длительность этого процесса ( $b$ ). Во время имитации, когда транзакт входит в этот блок, Arena вычисляет значение из указанного распределения, в нашем случае треугольного.

Для нашего процесса: минимальное время – 1 час, оптимальное – 1,75 часа (1 час 45 мин), максимальное время - 3 часа.

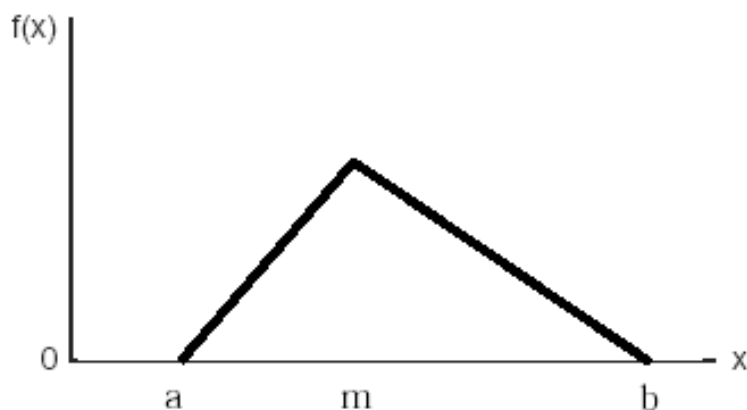


Рис. 5 Треугольное распределение

1. Откройте окно параметров модуля **Process 1**.
2. В поле *Name* введите **Rassmotrenie nalogovih deklaracii**.
3. Для описания ресурса (налогового инспектора) выберите пункт **Seize Delay Release** из списка *Action*.

Прибывшие в этот блок транзакты ждут своей очереди. Когда их очередь подходит они занимают ресурс (**Seize**), задерживают его в течении некоторого времени (**Delay**), затем освобождают (**Release**).

4. Список ресурсов появится в центре диалогового окна. Нажмите кнопку **Add** для добавлении ресурса в процесс.
5. В появившемся окне *Resources*, введите **Nalogovii inspector** в поле *Resource Name* (Рис. 6) и нажмите кнопку ОК.

**Рис. 6. Окно ресурсов процесса**

6. Опишите параметры задержки времени процесса **Minimum = 1, Value (Most Likely) = 1.75, Maximum = 3** (Рис. 7). Нажмите ОК.

## Рис. 7. Окно параметров модуля Process

Остальные поля оставим без изменений.

### Принятие решений (модуль Decide)

Когда налоговая декларация рассмотрена налоговым инспектором, он решает, принять ее или вернуть на дооформление. Там где существует только одна альтернатива, используется модуль **Decide**. Здесь будем использовать вероятность того, что 88% налоговых деклараций приняты.

1. Откройте окно параметров модуля **Decide 1**.
2. В поле *Name* введите **Prinyat nalogovuyu deklaraciyu?** (рис. 8).

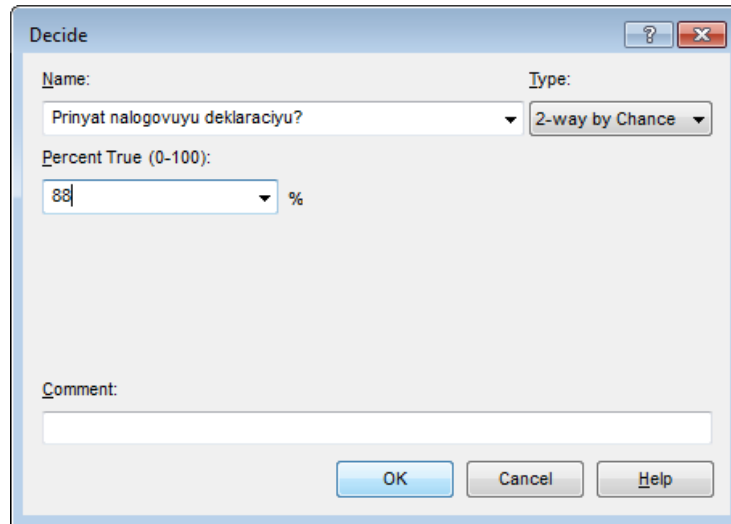


Рис. 8. Параметры модуля Decide

3. В поле *Percent True* введите **88** для определения процента заявок, которые будут связаны с решением TRUE (т.е. будут покидать модуль в правой точке выхода). Нажмите OK.

### Окончание процесса (модуль Dispose)

В нашем простом процессе рассмотрения налоговых деклараций вся интересующая нас работа проделана. Сейчас необходимо удалить транзакты из модели. Для этого используется модуль **Dispose**.

1. Дважды щелкните мышью на первом модуле **Dispose 1**, который соединен с условием TRUE модуля **Decide**. В окне параметров модуля **Dispose** введите **Prinyatye** в поле *Name* (Рис. 9). Нажмите OK.
2. Дважды щелкните мышью на втором модуле **Dispose 2**. В окне параметров модуля **Dispose** введите **Vozvrashennie** в поле *Name*. Нажмите OK.

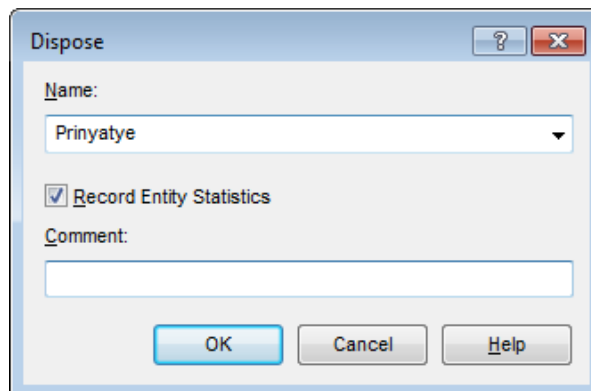


Рис. 9. Параметры модуля Dispose

Редактировать данные каждого модуля можно, используя электронные таблицы, включенные в модуль. Щелкните на значке модуля и электронная таблица, соответствующая модулю появится в нижней части экрана.

### Модуль Resource



**Resource** Вместе с блок-схемой можно описать параметры, связанные с другими элементами модели, такими как ресурсы, очереди и т.д. Для данного процесса опишем стоимость работы налогового инспектора (\$ 12 в час).

1. Щелкните мышью на значке **Resource** из панели **Data Definition**. В нижней части экрана появится электронная таблица **Resources**.
2. Так как ресурс – налоговый инспектор – был описан в блоке **Process**, Arena автоматически добавила его в список ресурсов модели. Щелкните на ячейке **Busy/Hour** и введите **12** для описания оклада налогового инспектора в час (**Hour**), когда он занят (**Busy**). Щелкните на ячейке **Idle/Hour** и введите **12** для описания оклада клерка в час (**Hour**), когда у него простой (**Idle**).

### Подготовка к запуску процесса

Для того чтобы подготовить модель к запуску, определим длительность процесса и другие параметры проекта.

1. В меню **Run** ⇒ **Setting** ⇒ **Setup** откройте вкладку **Project Parameters**. В поле **Project Title** введите *Priem deklaracii*, т.е. название проекта. В секции **Statistic Collection** определяются параметры проекта, которые следует выводить в отчет. Выберите *Entities, Queues, Resources, Processes* и *Costing*.
2. Выберите вкладку **Replication Parameters**. В поле **Replication Length** введите **20**, в соответствующем поле **Time Units** выберите *Days* из списка. Нажмите **OK**.

### Запуск процесса моделирования

Созданная модель содержит все данные, необходимые для процесса моделирования. Запустите модель, щелкнув по кнопке **Go** ►►, выбрав **Run** ⇒ **Interaction** ⇒ **Go**. Arena, после проверки правильности введенных вами данных, запустит модель.

В процессе моделирования вы увидите маленькие символические изображения транзактов,двигающихся от модуля к модулю. Также вы увидите как разные переменные изменяют свои значения по мере того, как создаются, обрабатываются, удаляются транзакты (Рис. 10).

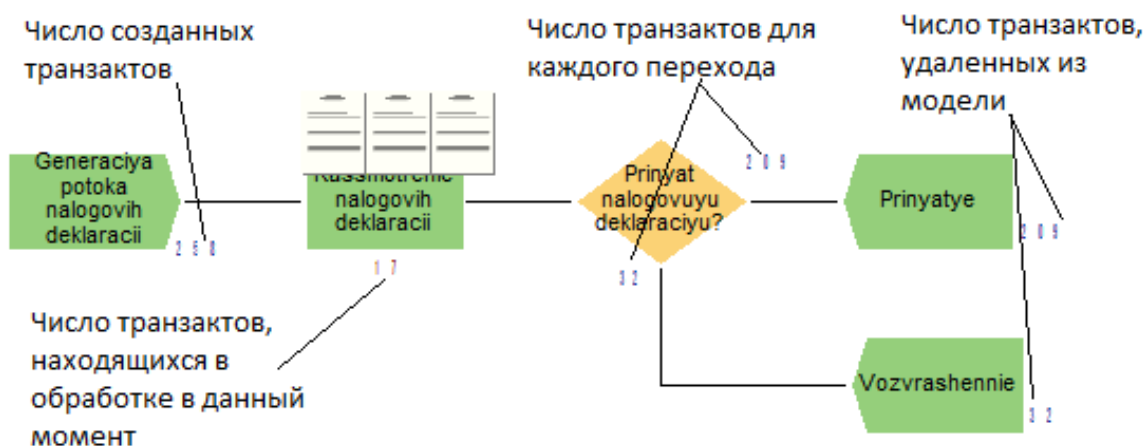





Рис. 10. Работа модели

Если изображения двигаются слишком быстро, возможно уменьшить скорость их движения через вкладку **Run ⇒ Setting ⇒ Setup ⇒ Run Speed** и в поле **Animation Speed Factor** введите меньшую величину.

Во время прогона модели, предусмотрена возможность перехода к концу процесса моделирования, для того, чтобы просмотреть отчет, таким образом не обязательно каждый раз дожидаться конца прогона модели. Приостановите прогон модели, нажав кнопку **Pause**  на панели инструментов или клавишу **Esc**, затем нажмите кнопку **Fast-Forward** , чтобы перейти к концу процесса моделирования. Возможно также просмотреть работу модели в режиме пошагового просмотра, наблюдая прохождение транзактов по модели, с помощью кнопки **Step**  на панели инструментов.

### Просмотр результатов моделирования

После завершения всех прогонов модели Arena можно посмотреть отчеты **Reports** на панели **Project Bar**.

Выберите раздел **Reports ⇒ Category Overview**. На левой стороне окна отчета размещено дерево, которое предназначено для навигации по отчету, находя нужную информацию. Имя проекта (в нашем случае это **Priem deklaracii**) отображается на вершине дерева. Щелкая по пунктам дерева можно видеть результаты моделирования. Каждый из формируемых в конце прогона статистических отчетов содержит поля, которые, в зависимости от объектной направленности отчета, характеризуют поведение того или иного объекта моделирования или процесса (Приложение 4).

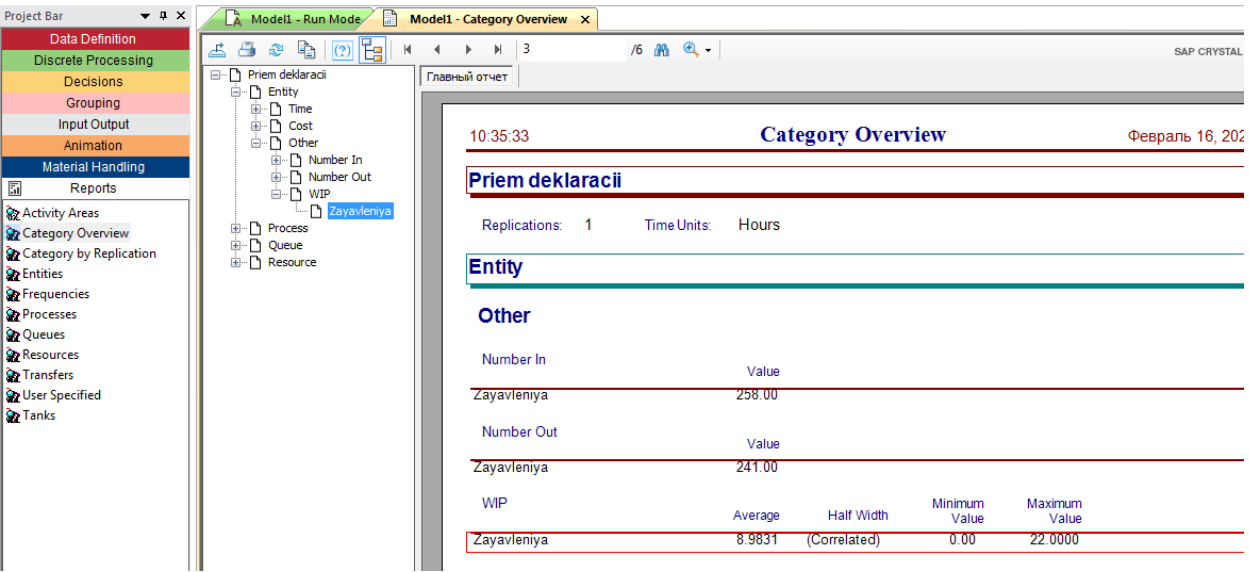


Рис. 12. Просмотр отчета

В таблице 1 приведены ряд вопросов и ответов по результатам моделирования описываемого процесса (выбранная строка выделяется красной рамкой).

Таблица 1.

### Результаты моделирования

Вопрос	Где искать	Ответ
Каково среднее время проведенное транзактом (налоговая декларация) в модели	Entity/ Time /Total Time / Колонка Average	16.51 часа
Какова средняя стоимость рассмотрения налоговой декларации налоговым инспектором	Entity/ Cost /Total Cost Колонка Average	\$22.99



Вопрос	Где искать	Ответ
Каково наибольшее время нахождения налоговой декларации в процессе рассмотрения	Process/Time per Entity /Total Time per Entity Колонка Maximum Value	33.45 часа
Какова была максимальная очередь	Queue/Other/Number Waiting Колонка Maximum Value	21 декларация
Средняя загруженность ресурса (налогового инспектора)	Resources/Usage/Scheduled Utilization Колонка Average	0.9654 или 96.54 %






## Расширение визуализации процесса

Итак, закончив основные шаги для анализа процесса можно вернуться к модели и добавить графическую анимацию, чтобы улучшить визуализацию динамики процесса.

Добавим 2 анимационных компонента в модель. Во-первых, можно показать налогового инспектора, сидящего за столом, занятого или свободного. А для того, чтобы увидеть сколько заявок находятся в очереди во время процесса, также добавим динамический график (WIP-Work-In-process).

### Анимация ресурса *Nalogovii inspector*

Во время процесса налоговый инспектор может быть в двух положениях. Если нет налоговых деклараций, то инспектор не занят (**Idle**). Будем использовать картинку с человеком, сидящим за столом. Когда же налоговая декларация появляется, состояние меняется на занят (**Busy**). В этом случае будем использовать картинку, на которой человек просматривает документ.

1. Перейдите на вкладку **Animate**.
2. Щелкните на кнопке **Resource**  группа **Pictures**.
3. Появится диалоговое окно **Resource Picture Placement** (Рис. 13). Выберите *Nalogovii inspector* из списка поля **Identifier**.
4. Откройте библиотеку *people*, щелкнув мышью на кнопке **Open** и указав имя файла **People.plb**.
5. Для изменения значка **Idle**:
  - ✓ Щелкните по кнопке **Idle**.
  - ✓ Выберите из набора значков справа, значок с изображением человека, сидящего за столом, например такую .
  - ✓ Нажмите кнопку  для использования этого выбранного значка для состояния **Idle** (свободен).
6. Для изменения значка **Busy**:
  - ✓ Щелкните по кнопке **Busy**.
  - ✓ Выберите из набора значков справа, значок с изображением человека читающего документ, например такую .
  - ✓ Нажмите кнопку  для использования этого выбранного значка для состояния **Busy** (занят).
7. Нажмите ОК.
8. Курсор примет вид крестика со значком ресурса. Переместите его в окно модели и щелкните на пустом месте, в результате в модель добавилась картинка визуализирующая работу налогового инспектора.
9. Если вы хотите изменить размер изображения, используйте маркеры размера.

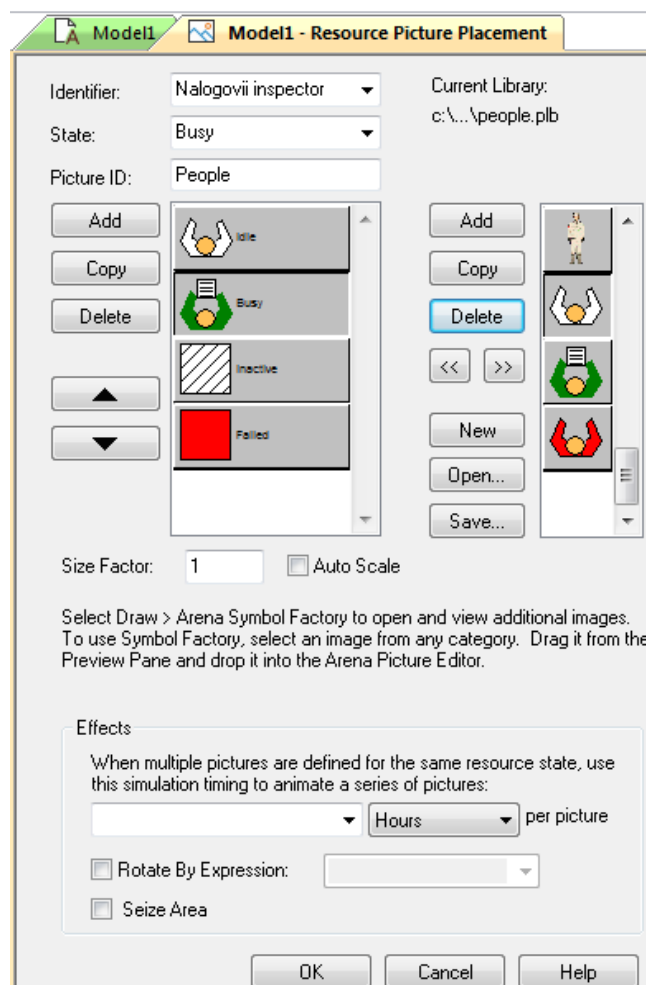
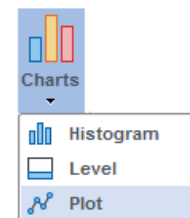


Рис. 13. Окно анимации ресурса с введенными параметрами

## График



1. На вкладке **Animate** ⇒ **Status** ⇒ **Charts** щелкните по кнопке **Plot**.
2. Будем строить график из одной переменной **Work-In-Process (WIP)**. Щелкните по кнопке **Add** для добавления этой переменной.
3. В появившемся окне в поле **Expression** введите **Rassmotrenie nalogovih deklaracii.WIP** (Рис. 14).

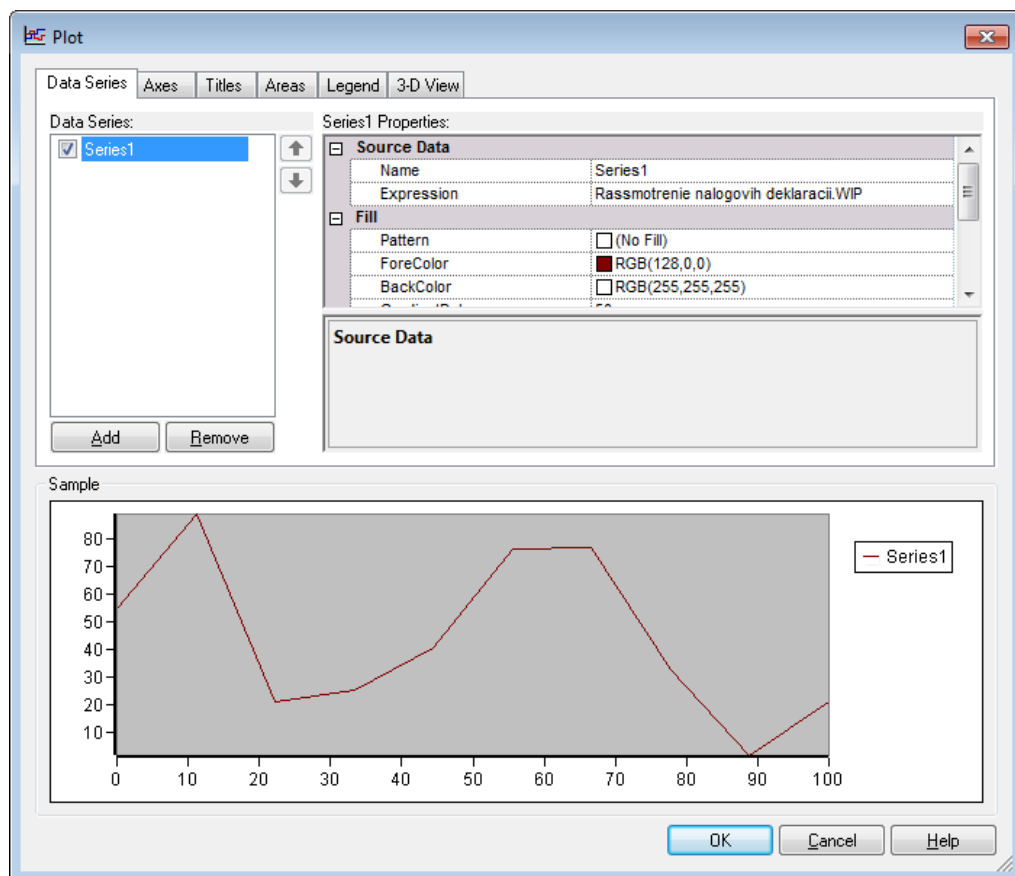


Рис. 14. Диалоговое окно Plot

4. В разделе **Series1 Properties 1** выберите строку **Expression** В появившемся окне в поле **Expression** введите **Rassmotrenie nalogovih deklaracii.NumberIn** и исправьте **NumberIn** на **WIP** (рис. 14).
5. Из отчета было видно, что максимальная очередь составляла **21** (таблица 1). Установим максимальное значение для графика **25**. В диалоговом окне **Plot** перейдем на вкладку **Axes** и в разделе **Axes** выберем **Left Value (Y) Axis**, а в разделе **Left Value (Y) Axis Properties** выберем группу **Scale Maximum** и установим значение **25** (рис. 15).

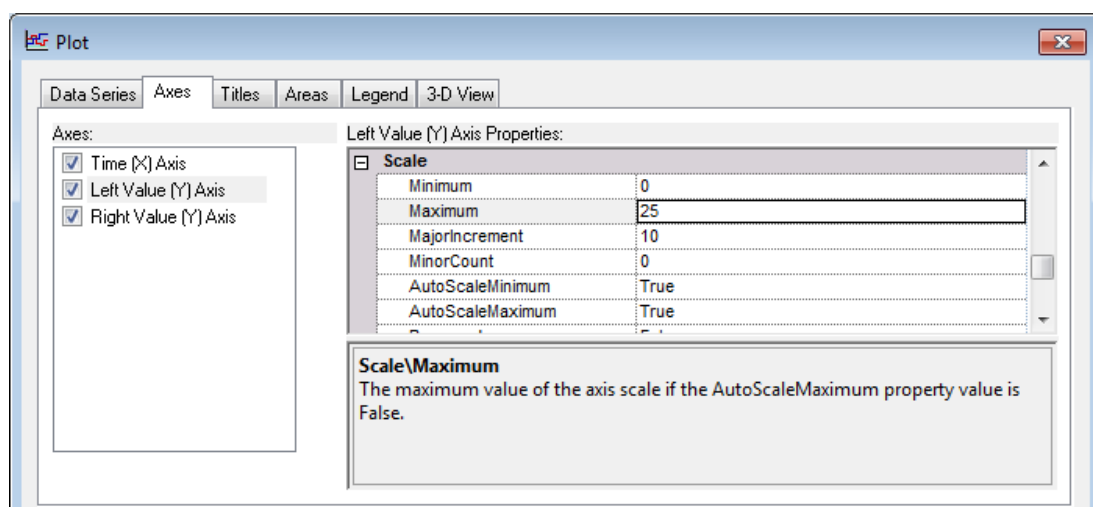


Рис. 15. Выбор вида зависимости для построения графика

6. Для завершения описания графика измените в диалоговом окне **Plot** на вкладке **Axes**, в разделе **Axes** выберите **Time (X) Axis**, а в разделе **Time (X) Axis Properties**, выберем группу **Scale Maximum** и установим значение **480**. Нажмите **OK**.

Горизонтальная ось нашего графика будет отображать 480 часов (20 дней), в соответствии с продолжительностью нашей модели. Курсор примет вид крестика. Поместите график снизу блок-схемы. Для изменения размеров используйте маркеры размера.

Запустите модель, нажав клавишу F5. В течение прогона модели видно, как налоговый инспектор меняет свое состояние со свободного на занятое, и наоборот (Рис. 15).

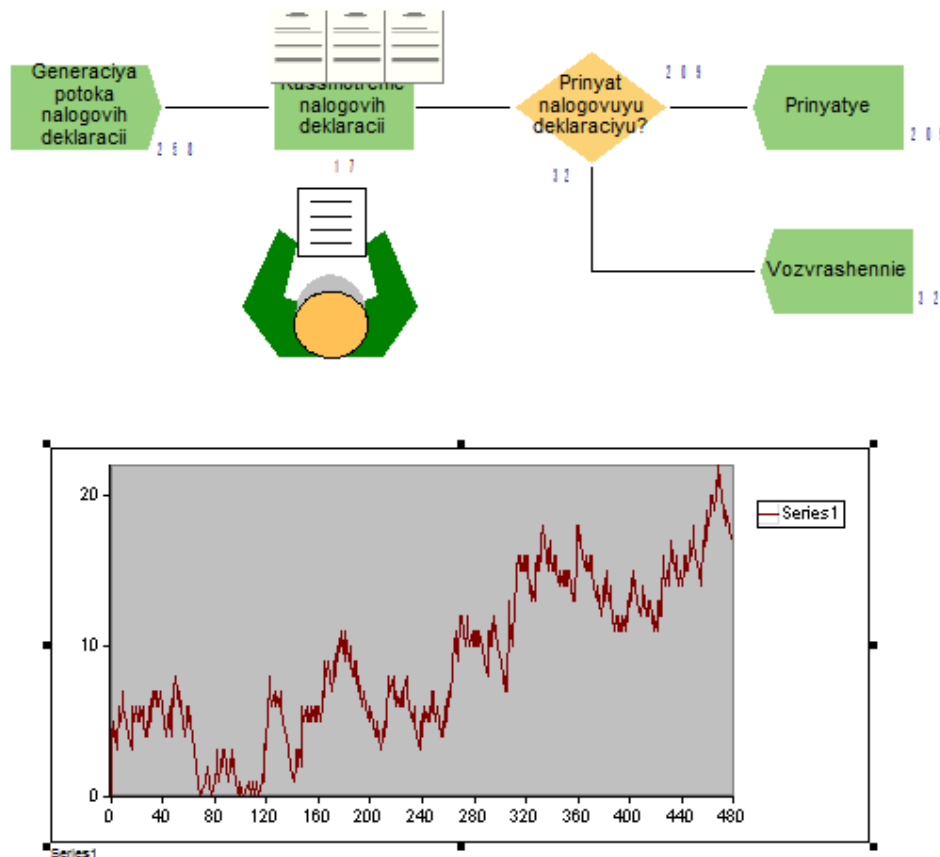


Рис. 16. Модель с анимацией

### Задание к части 1

1. Познакомиться с интерфейсом Arena.
2. Построить модель, описанную выше, сделать все настройки, анимацию и просмотреть результаты моделирования, сравнить их с данными, приведенными в Таблице 1.
3. Добавьте процесс регистрации налоговой декларации до того, как она попала к налоговому инспектору. Заявка может быть зарегистрирована минимум за 15 минут. Чаще всего это занимает 25 минут, иногда 45 минут. Опишите регистратора (ставка 6.75\$ в час). Анимируйте ресурс регистратора. Какую функцию распределения времени нужно использовать в этой задаче?
4. По завершении процесса регистрации 10 % налоговых деклараций должны быть возвращены. Процент налоговых деклараций, рассмотренных налоговым инспектором и принятых им должен возрасти с 88% до 95%, а время процесса уменьшится на 10%. Как изменится стоимость рассмотрения 1 налоговой декларации?

### Контрольные вопросы

1. Назовите основные рабочие области Arena.
2. Как добавить в **Project Bar** панели модулей **Blocks**, **InputOutput** и др.?
3. Как сохранить вид модели?

4. Опишите назначение и основные свойства модулей **Create, Decide, Process, Dispose**. Каким образом можно их редактировать?
5. Опишите способы создания соединений между модулями модели.
6. Опишите способы запуска модули, настройки прогона, изменение скорости просмотра модели.
7. Опишите структуру отчетов Arena.
8. Где можно настраивать отображаемые в отчетах параметры?
9. Как создать анимацию для ресурса?
10. Как построить график в Arena?

## Часть 2. Визуализация модели в Arena

### Цель работы

Изучение возможностей пакета Arena для визуализации моделей.

### Разработка анимации

Создадим простую модель, которая отображает различные действия и сделки в банке. Будем моделировать три действия:

1. Работа банкомата
2. Сделки, которые можно выполнять, не покидая автомобиля
3. Сделки кассира

Рассмотрим поэтапно создание анимации для каждого действия. Все анимационные действия выполняются в одном файле. Сначала откройте файл `pril_lab.doe` с планом банка. Рекомендуется просматривать анимацию при добавлении каждого нового элемента. Все три процесса необходимо анимировать на одной схеме.

Установите для модели скорость 0.2 и следующие параметры (Рис. 17).

**Replication Parameters**

Number of Replications: 1

Start Date and Time: 16 февраля 2021 г. 14:53:22

Warm-up Period: 0.0 Minutes

Replication Length: 1 Days

Hours Per Day: 24

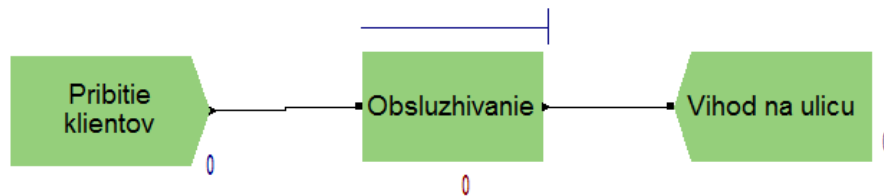
Terminating Condition:

Base Time Units: Minutes

Рис. 17. Run Setup

### Работа банкомата

Процесс обслуживания клиентов банкоматом описывается моделью представленной на рис. 18.




**Рис. 18. Блок-схема работы банкомата**

Прибытие клиентов (klient) подчиняется экспоненциальному распределению, клиенты приходят обслуживаться у банкомата каждые 10 vbyen (Рис. 19).

**Рис. 19. Модуль Create**



Используя модуль **Entity** назначьте для этого типа транзактов (klient) какое-нибудь графическое изображение (Initial Picture) в виде человечка, например **Picture.Man**.

Обслуживание клиентов подчиняется равномерному распределению (UNIF) максимальное время обслуживания – 12 мин, минимальное 7 мин. Будем считать, что у нас один банкомат обслуживает клиентов, назовем этот ресурс ATM Machine. Используя кнопку  **Resource** анимируйте ресурс ATM Machine и разместите на схеме как показано на рис. 22. Рисунки для анимации ресурса ATM Machine показаны на рис. 20.

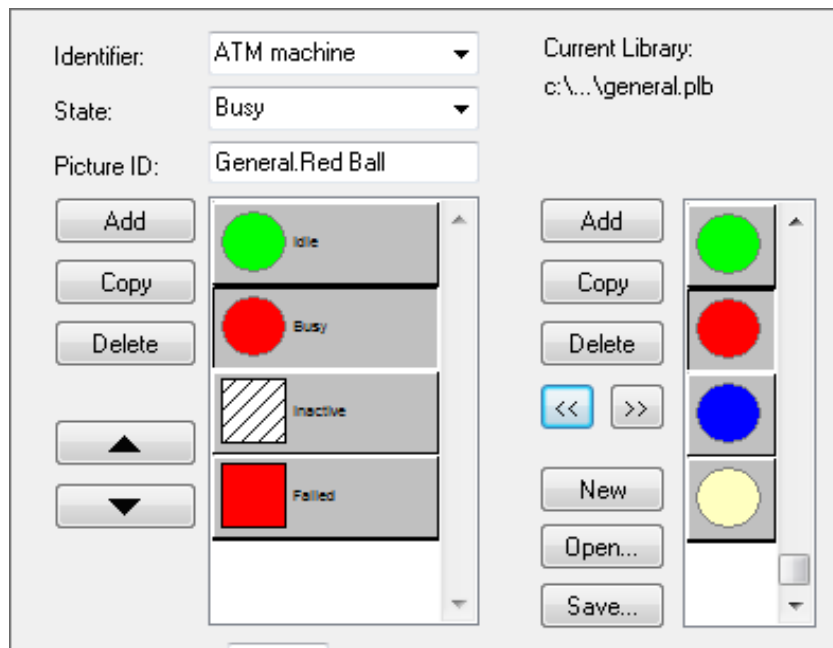


Рис. 20. Анимация ресурса ATM Machine

Преобразуем модель на рис. 18 для создания анимации обслуживания клиентов у банкомата (Рис.21). Здесь добавились два новых типа блока – **Station** (ATM Entrance, ATM Area, Exit Bank) и **Route** (Leave ATM Entrance, Leave ATM Area), которые и используются для анимации модели.

**Station** – логическая точка, в которую прибывают транзакты, которые отправлены на станцию. Предназначен для графического отображения перемещения транзактов. Указывается идентификатор графической станции, с которой принимается транзакт.

В диалоговом окне **Station** задайте имена для станций (**Station name**):

ATM Entrance - ATM\_Entry;

ATM Area - ATM Mach\_ST;

Exit Bank - Exit Bank\_St.

**Route** – определяет путь транзакта между станциями.

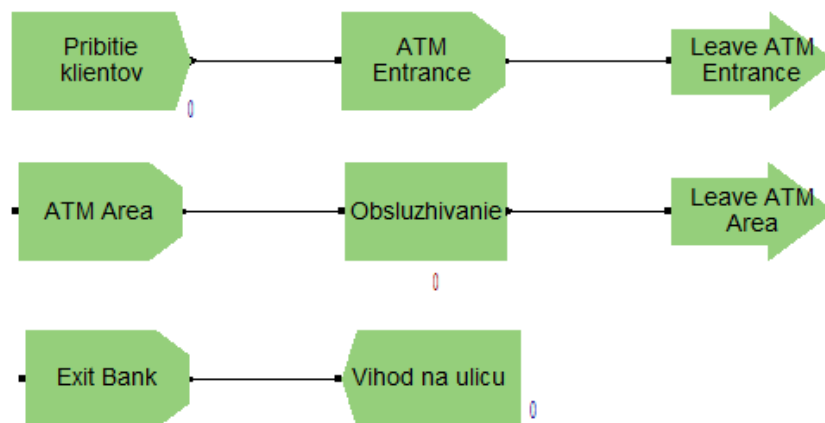




Рис. 21. Анимационная блок-схема работы банкомата

Для блока Leave ATM Entrance введите в поле Route Time - UNIF(1, 2) (в минутах); в поле Station Name - ATM Mach\_ST. Для блока Leave ATM Area введите в поле Route Time - 1 (в минутах); в поле Station Name - Exit Bank\_St.

Используя инструмент  **Station** (вкладка **Animate**) расставьте станции как показано на рис. 22. Используя инструмент  **Route** соедините станции как показано на рис. 22. У

маршрута между станциями ATM\_Entry и ATM Mach\_ST задайте 13 точек, у другого маршрута - 18.

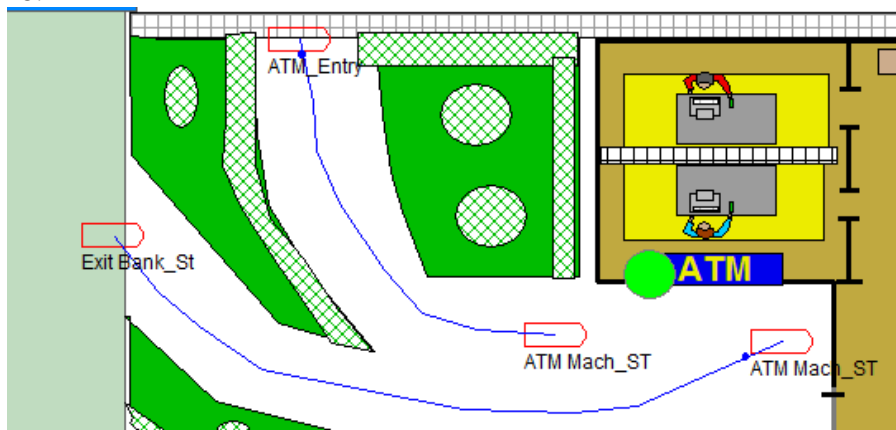



Рис. 22. Расстановка станций.

Перенесите очередь, расположенную над блоком Obsluzhivanie, в ту часть схемы модели как показано на рис. 23.



Рис. 23. Размещение очереди на анимационной схеме

Для более наглядной демонстрации обслуживания клиента у банкомата, воспользуемся инструментом Seize, разместите его так как показано на рис. 24. Для этого выберите кнопку  Seize (вкладка **Animate**) и перетащите ее на изображение ресурса ATM Machine.

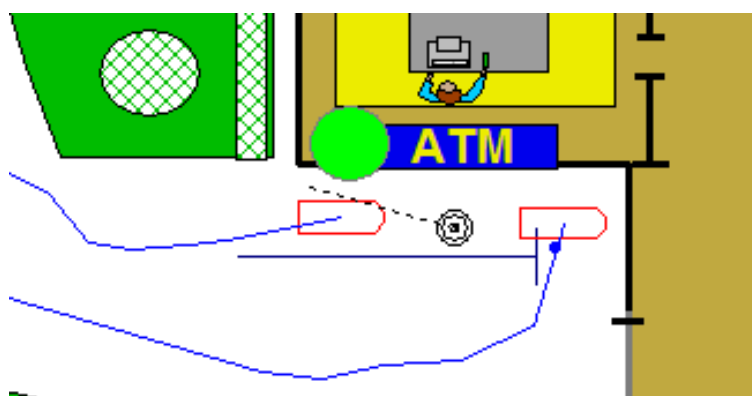


Рис. 24. Размещение элемента Seize

### **Сделки, которые можно выполнять, не покидая автомобиля**

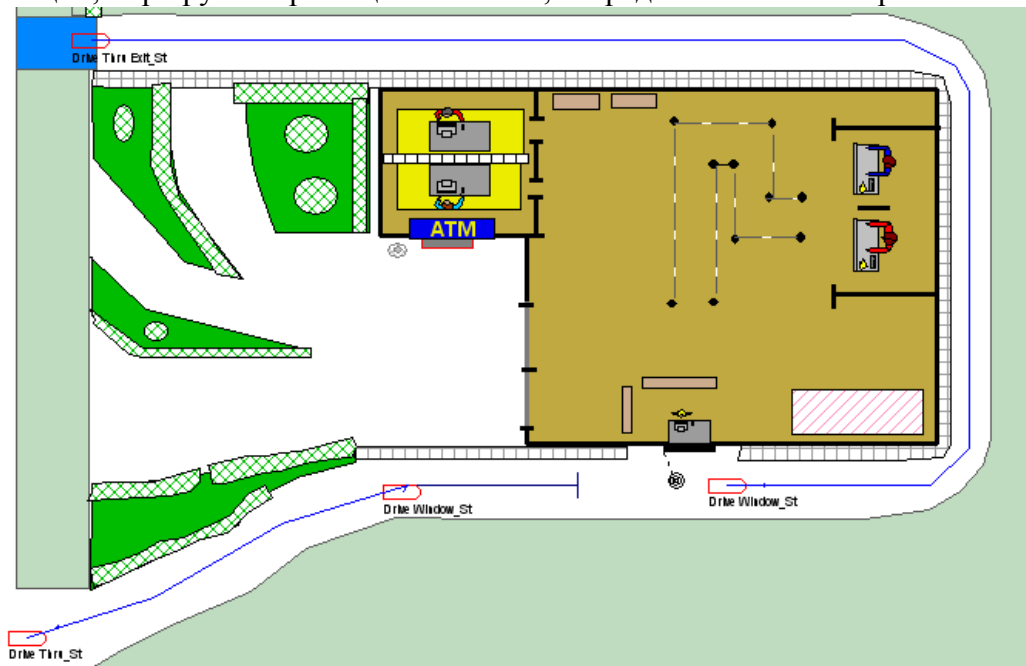
#### Выполнить самостоятельно.

Прибытие клиентов на машинах подчиняется экспоненциальному распределению, машины подъезжают обслуживаться с интервалом в 21 минуту.

Обслуживание клиентов на машинах подчиняется равномерному распределению (UNIF) максимальное время обслуживания – 20 мин, минимальное 7 мин.



Станции, маршруты перемещения машин, очереди и показаны на рис. 25



**Рис. 25. Анимационная процесса «сделки, которые можно выполнять не покидая автомобиля»**

### **Сделки кассира**

Это задание показывает использование набора ресурса в пределах банковской области услуг. Прибытие клиентов к 2 кассам подчиняется экспоненциальному распределению, клиенты прибывают с интервалом в 11 минут. Когда клиент прибывает на обслуживание в этот процесс, для обработки требуется один из кассиров. Выбор кассира устанавливается циклически, на основании их готовности к обслуживанию клиента.

Обслуживание клиентов у кассиров подчиняется экспоненциальному распределению (expo) со средним значением 17 мин.

В окне Resources установите следующие настройки:

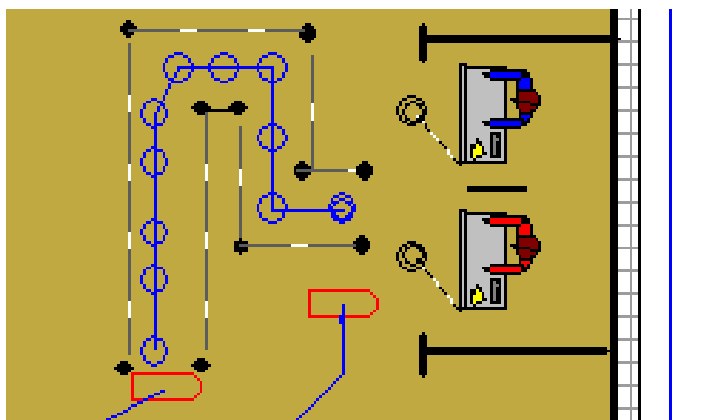
**Рис. 26. Использование набора ресурсов**



Используя модуль Set создайте набор ресурсов: два кассира Nancy и Judy, для этого щелкните мышью по кнопке **0 rows** и создайте двух членов набора. Потом выберите модуль

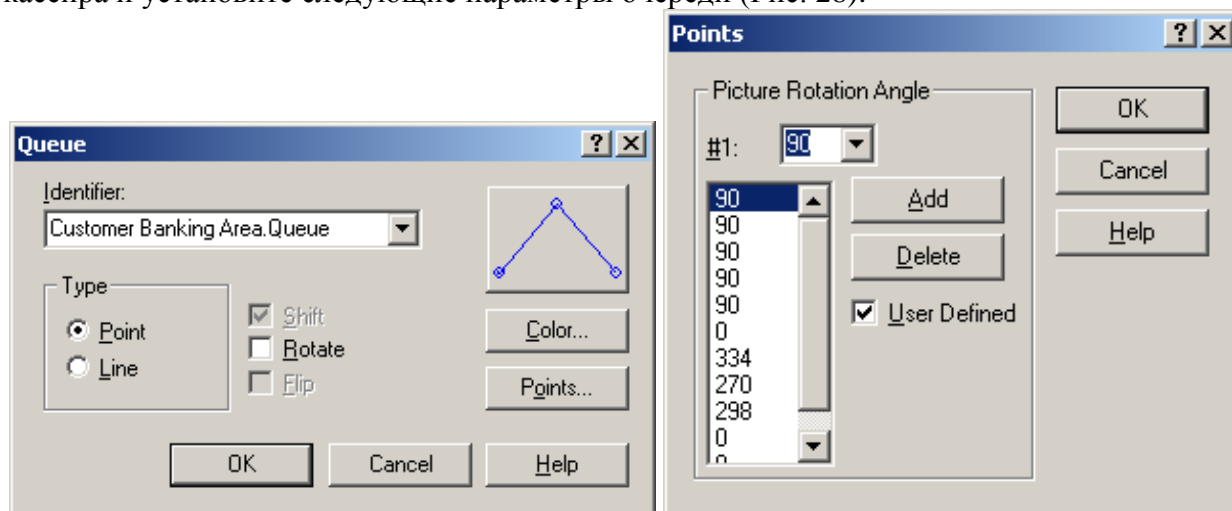


Resource и анимируйте ресурсы Nancy и Judy, как показано на рис. 27. Для более наглядной демонстрации обслуживания клиентов у кассиров, воспользуемся инструментами Seize и разместим их так как показано на рис. 27.



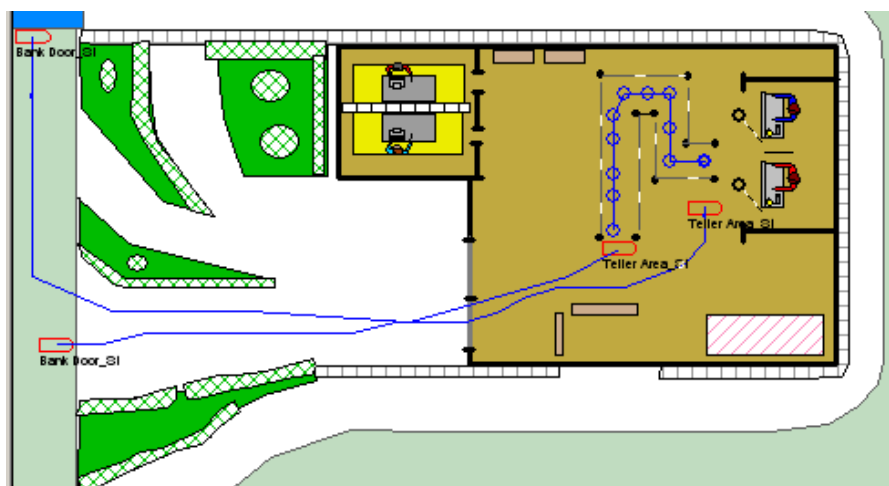
**Рис. 27. Анимация набора ресурсов**

Выберите окно редактирования свойств очереди процесса обслуживания клиентов у кассира и установите следующие параметры очереди (Рис. 28):



**Рис. 28. Редактирование свойств очереди**

Разместите очередь так как показано на рис. 27. Станции, маршруты перемещения клиентов кассиров, очереди и показаны на рис. 29.



**Рис. 29. Размещение станций процесса сделки кассира**

Используя функцию NQ (длина очереди), постройте графики очередей к процессам работы банкомата, сделкам кассира, обслуживания автомобилей (рис. 30).

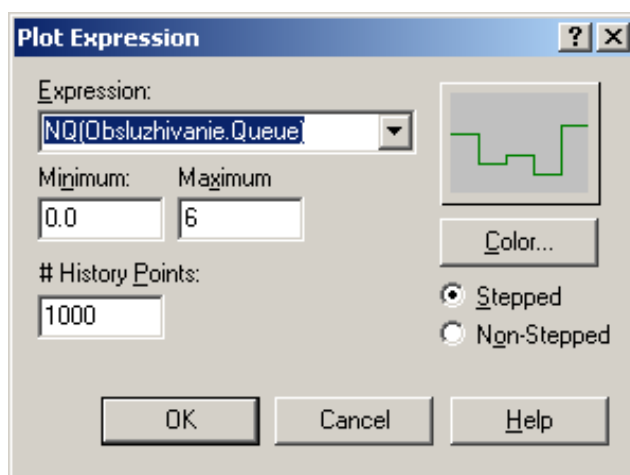


Рис. 30. Использование функции NQ

#### Контрольные вопросы:

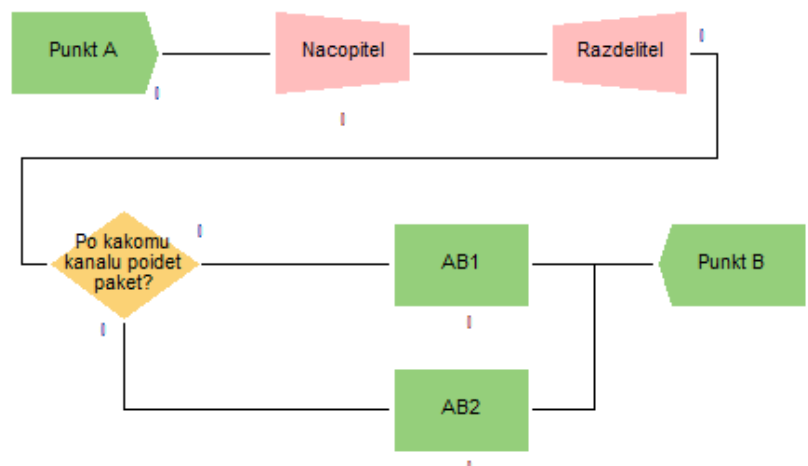
1. Как настроить анимацию ресурсов модели?
2. Какие блоки используются для создания анимации в Arena?
3. Опишите назначение и основные параметры блока Route.
4. Опишите назначение и основные параметры блока Station.
5. Для чего используется инструмент Seize при создании анимационной схемы?
6. Как задать «набор» ресурсов?

### Часть 3. Моделирование производственных процессов в Arena

#### Модуль Batch

Блок **Batch** предназначен для склеивания (группировки) определенного количества транзактов. При этом склеивании из нескольких транзактов получается один (так называемый Batch – пакет). Организовывается очередь, в которой скапливаются транзакты пока их не накопится заданное количество. Полученный склеенный транзакт может быть временным (**Temporary**), т.е. предназначенный для последующей расклейки, или постоянным (**Permanent**), тогда в последствии его не удастся расклеить. В данном блоке можно задать тип и идентификатор графической очереди, в которой будет происходить ожидание.

**Пример 1.** Система передачи данных обеспечивает передачу пакетов данных из пункта А в пункт В. В пункт А пакеты поступают по нормальному закону со средним временем  $20 \pm 2$  с. Здесь они буферизуются в накопителе емкостью 20 пакетов и передаются по любой из двух линий АВ1 — за время  $15 \pm 2$  с или АВ2 — по показательному закону со средним временем, равным 10 с. Блок схема такой ситуации представлена на рис. 12.



**Рис. 12. Блок-схема системы передачи данных. Пример 1**

Модуль **Create (Punkt A)** организует поступление пакетов данных по нормальному закону со средним временем 10 с и квадратическим отклонением 1с (Expression: NORM(10, 1); Units: Seconds).

Модуль **Batch (Nacopitel)** накапливает 20 пакетов, а затем выпускает их как единое целое, но с возможностью его разделения на первоначальные единицы (Batch Size: 20; Type: Temporary)

Модуль **Separate (Razdelitel)** разделяет группу поступившую из **Nacopitel** на первоначальные 20 пакетов (Type: Split Existing Batch; Member Attributes: Retain Original Entity Values)

Модуль **Decide (Po kakomu kanalu poidet paket?)** случайно распределяет пакеты по 2-ум каналам передачи с вероятностью каждого варианта 50%.

Модули **Process (AB1 и AB)** имитируют прохождение пакетов по линиям AB1 (Action: Delay; Expression: NORM(15,2); Units: Seconds) и AB2 (Action: Delay; Expression: EXPO(10); Units: Seconds).

#### **Модуль Assign**

**Модуль Assign** используется для присвоения значений. Когда транзакт попадает в этот блок, происходит присвоение значений, которые указаны в этом блоке. Новые значения могут присваиваться атрибутам транзакта, переменным, состояния ресурсов, также можно поменять картинку у транзакта. Присваиваемое значение вычисляется в момент присвоения, поэтому могут быть заданы формулы или выражения (Expressions).

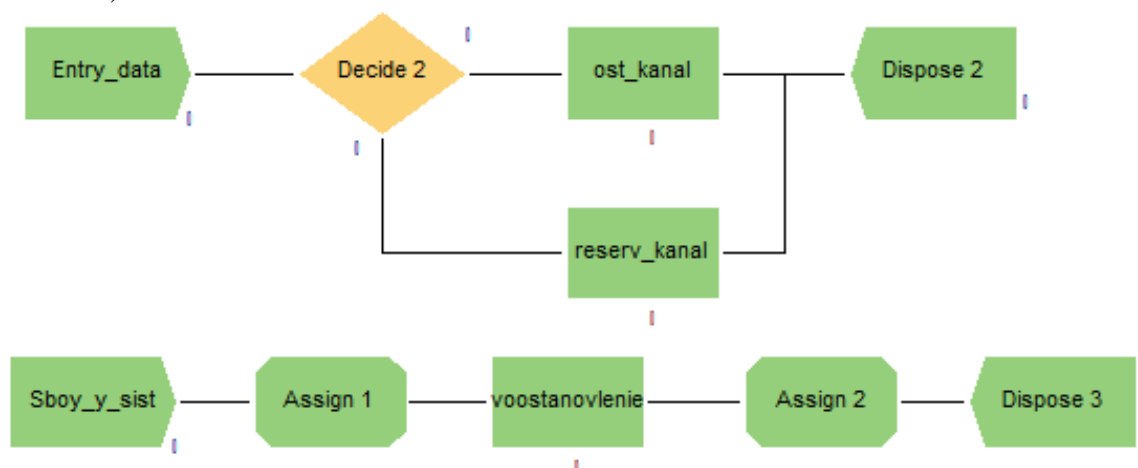
**Пример 2.** Магистраль передачи данных состоит из двух каналов (основного и резервного) и общего накопителя. При нормальной работе сообщения передаются по основному каналу за среднее время 8 с. в соответствии с пуассоновским законом распределения. В основном канале происходят сбои каждые  $200 \pm 45$  с. Если сбой происходит во время передачи, то запускается резервный канал. Восстановление основного канала занимает в среднем 60 с. по экспоненциальному закону обслуживания. После восстановления резервный канал выключаются, и основной канал продолжает работу с очередного сообщения. Поступающие распределены по нормальному закону со средним 10 с. и квадратическим отклонением 2 с. (Блок-схема модели представлена на рис. 13).

Модуль **Create (Entry\_data)** организует поступление сообщений нормальному закону со средним временем 10 с и квадратическим отклонением 2 с (Expression: NORM(10,2); Units: Seconds).

Модуль **Create (Sboy\_v\_sist)** имитирует наступление сбоя в основном канале, который происходит каждые  $200 \pm 45$ с (Expression: UNIF(155,245); Units: Seconds; First Creation:200).

Модули **Process (osn\_kanal и reserv\_kanal)** имитируют прохождение сообщений по каналам **osn\_kanal** (Action: Delay; Expression: POIS(8); Units: Seconds) и **reserv\_kanal** (Action: Delay; Expression: POIS(8); Units: Seconds).

Модуль **Process (voostanovlenie)** имитируют восстановление основного канала по экспоненциальному закону со средним значением 60с (Action: Delay; Expression: EXPO(60); Units: Seconds).



**Рис. 13. Блок-схема магистрали передачи данных. Пример 2**

В модуле **Decide 1** проверяется осуществление сбоя в основном канале (Type: 2-way by Condition; If: Expression; Value: Tsboy.NE.1). Условие **Tsboy.NE.1** означает: если значение переменной Tsboy не равно 1, то сбой не произошел (или основной канал восстановлен), следовательно, передача сообщений будет происходить по основному каналу. Иначе, передача сообщения произойдет по резервному каналу.

В модулях **Assign 1** и **Assign 2** происходит присваивание переменной **Tsboy** значений **1** и **0** соответственно. Это необходимо для проверки условия осуществления сбоя.

### Модули Hold и Signal

Модуль **Hold** накапливает транзакты в очереди в ожидании специального сигнала или бесконечно (чуть позже транзакты удалятся модулем **Remote**). Для выбора модулей **Hold** и **Signal** подключите панель **AdvancedProcess.tpo**.

Если транзакты в модуле **Hold** ожидают сигнала, то модуль **Signal** используется, чтобы позволить транзактам переходить к следующему модулю модели.

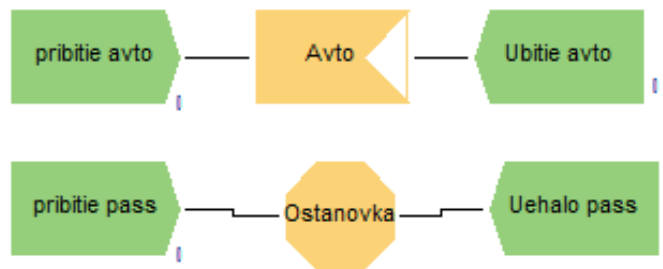
**Пример 3.** На автобусную остановку приходят пассажиры, чтобы уехать в различные районы города. Они дожидаются на остановке автобуса и садятся в него при условии наличия свободных мест. Число свободных мест в прибывающих автобусах распределяется по равномерному закону  $5 \pm 5$ . Блок-схема модели показана на рис. 14. Исходные данные для моделирования представлены в таблицах 3 и 4.

**Таблица 3.**

Время между моментами прибытия пассажиров, мин	8	10	12	14	16
Вероятность	0,1	0,38	0,28	0,15	0,09

**Таблица 4.**

Интервал последовательными прибытиями автобусов, мин	0	1	2	3	4	5	6
Вероятность	0,04	0,16	0,25	0,28	0,16	0,10	0,02



**Рис. 14. Блок-схема модели прибытия на остановку автобусов и пассажиров**

Модуль **Create (Pribitie avto)** имитирует прибытие автобусов на остановку, с определенными интервалами времени. Время прибытия автобусов описывается по дискретному закону (Expression: DISC(0.1,8,0.48,10,0.76,12,0.91,14,1,16); Units: Minutes).

Модуль **Create (Pribitie pass)** имитирует прибытие пассажиров на остановку с определенными интервалами времени. Время прибытия пассажиров описывается по дискретному закону (Expression: DISC(0.04,0,0.2,1,0.44,2,0.72,3,0.88,4,0.98,5,1,6); Units: Minutes).

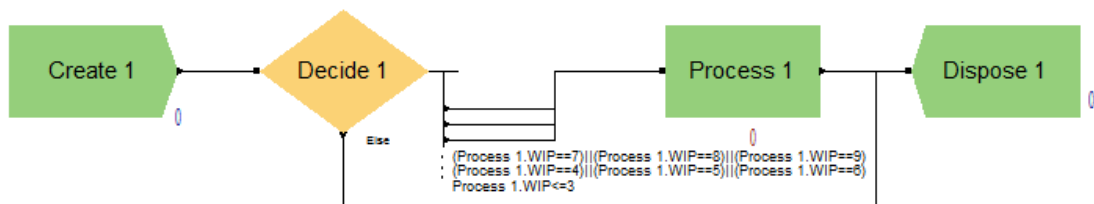
Модуль **Signal (Avto)** посылает сигнал о прибытии автобуса в модуль **Hold** для его освобождения (Signal value: 1).

В модуле **Hold (Ostanovka)** осуществляется имитация посадки пассажиров в автобус. Пассажиры приходят на остановку, становятся в очередь и ждут прибытия автобуса. Когда автобус прибывает на остановку модуль **Signal** посылает сигнал в модуль **Hold** для его освобождения, т.е. посадки пассажиров в автобус, при этом количество свободных мест в автобусе изменяется по равномерному закону  $5 \pm 5$  (Type: Wait for Signal; Wait for Value: 1; Limit: UNIF(0,10); Queue Type: Queue; Queue Name: Ostanovka.Queue).

Модули **Dispose (Ubitie avto и Uehalo pass)** имитирует убытие автобусов и отъезд пассажиров.

### Модуль N-Decide

**Пример 4.** Соберите блок-схему как на рис. 15.а.

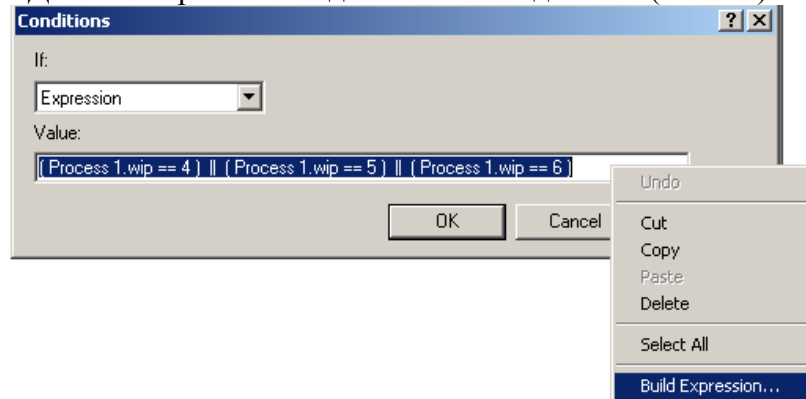


**Рис. 15. Блок-схема**

Транзакты прибывают в модель, и далее с помощью блока **Decide** разделяются на 3 потока в зависимости от количества транзактов, обрабатываемых в блоке **Process 1**. (рис.16)

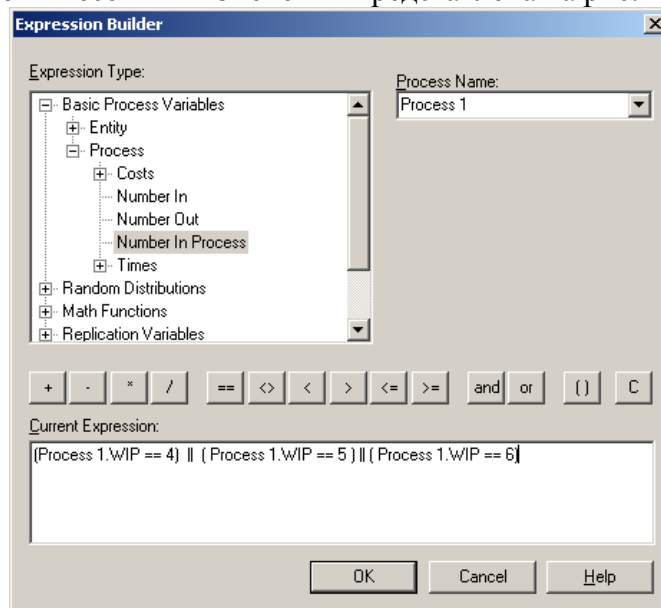
**Рис. 16. Окно блока Decide**

Данные выражения задаются во вкладке Edit (Рис. 17)



**Рис. 17. Окно блока Decide (вкладка Edit)**

Для редактирования данного выражения используйте Build Expression. (рис. 18). Блок-схема Примера 4 с учетом внесенных изменений представлена на рис. 15.б.



**Рис. 18. Окно Build Expression**

### Задание 3

#### 1. Выполните Примеры 1-4.

#### 2. Решите задачи:

№ 2.1. Трасса, ведущая к селу Три Протоки, состоит из двух дорог (через переезд со шлагбаумом - главная и без него - объездная). Когда поезда нет, машины проходят промежуток рельс за 7 секунд в соответствии с пуассоновским законом распределения. Трасса перекрывается каждые  $80 \pm 15$  минут. Если трасса перекрыта во время движения, то машины едут по объездной дороге. Восстановление движения на основной дороге занимает в среднем 4 минуты по экспоненциальному закону. После восстановления машины едут по основной дороге с очередной машины. Проезжающие распределены по нормальному закону со средним 8 с. и квадратическим отклонением 2 с.

№ 2.2. Чтобы попасть в деревню на машине в летнее время, необходимо переправиться через реку. Это возможно сделать двумя путями: в объезд по мосту (основной путь) или при помощи паромы (дополнительный путь). При нормальной работе моста машины проезжают по нему за среднее время 56 секунд в соответствии с пуассоновским законом распределения. Мост разводят каждые  $480 \pm 30$  минут. Если мост разводят во время движения машин, то им приходится ехать к пароме и переправляться через реку с его помощью. Мост стоит разведенным в течение примерно 120 минут по экспоненциальному

закону. После сведения моста машины снова начинают движение по мосту с очередной машины. Проезжающие машины распределены по нормальному закону со средним 1 минуту и квадратическим отклонением 10 секунд.

№ 2.3. В продуктовый минимаркет в течение дня приезжают поставщики различного товара, оставляя определенную его партию в магазине. Покупатели заходят в магазин для того, чтобы купить продукты. Товар покупается при условии, что он удовлетворяет вкусу потребителя. Число единиц продукции, покупаемых потребителями распределяется по равномерному закону  $10 \pm 10$ . Исходные данные для моделирования представлены в таблицах ниже (используйте функции распределения вероятностей DISC):

Время между моментами привоза товара, мин	0	1	2	3	4	5	6
Вероятность	0,02	0,18	0,21	0,32	0,12	0,13	0,02

Интервал между последовательными прибытиями потребителей, мин	0	1	2	3	4	5	6
Вероятность	0,02	0,18	0,21	0,32	0,12	0,13	0,02

№ 2.4. Перевозку пассажиров с причала на городской пляж осуществляют 4 трамвайчика (2 в сторону пляжа и 2 в сторону причала). На причал постоянно приходят люди по нормальному закону распределения со средним временем  $20 \pm 10$  секунд. Здесь они собираются в очереди по 40 человек и едут на любом из двух отправляющихся на данный момент трамвайчиков: на первом – за время  $18 \pm 5$  секунд или на втором – по показательному закону со средним временем, равным 11 секунд.

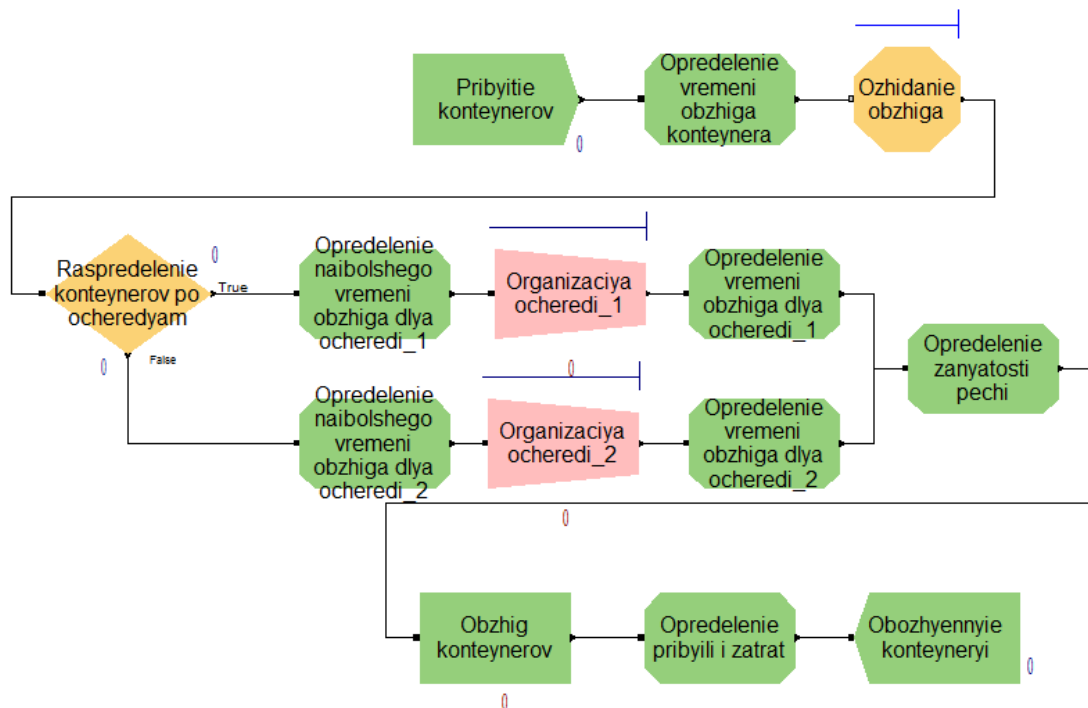
№ 2.5. Перед участием в соревнованиях спортсмены проходят мед. комиссию в физ. диспансере. Организован прием сразу двумя независимыми комиссиями. Спортсмены приходят в диспансер по нормальному закону со средним временем  $60 \pm 15$  секунд. В холле они собираются в группы по 6 человек и идут на прием к одной из двух комиссий: к первой – за время  $55 \pm 10$  секунд или ко второй – по показательному закону со средним временем, равным 50 секунд.

### 3. Решите задачу моделирования производственного процесса

Контейнеры с керамическими изделиями поступают в цех обжига (входной поток пуассоновский с параметром 1/20). Каждый контейнер содержит партию из 100 изделий, которые требуют одинакового времени обжига. Контейнеры разделяются на две очереди: очередь с большим временем обжига (более 20 минут) и очередь с меньшим временем обжига (менее 20 минут) изделий в печи. В печь загружаются по три контейнера из каждой очереди, выбор осуществляется по принципу FIFO. Для поддержки этих двух очередей необходимо 15 единиц стоимости. Время обжига - равномерно распределенная величина в интервале  $20 \pm 8$  минут. Время обжига соответствует наибольшему из времен, необходимых для обжига изделий из этих трех контейнеров. Прибыль от обжига каждого изделия составляет 5 единиц стоимости. Одна минута работы печи требует 20 единиц стоимости (учитывается только "чистое" время работы печи).

На основе входных данных построить имитационную модель цеха обжига керамических изделий и определить экономическую эффективность. Организовать останов модели после того как показатель «Чистая прибыль» составит определенное количество единиц стоимости (5000, 10000, 15000).





**Рис. 19. Блок-схема производственного процесса**

Имитационная модель цеха обжига для дисциплины обслуживания состоит из следующих блоков:

- Create:
  - "*Pribytie konteynerov*" – генерирует прибытие транзактов по пуассоновскому закону с интенсивностью 20 (в час).
- Process:
  - "*Obzhig konteynerov*" – производит обжиг контейнеров (Seize Delay Release; Resources: "Resource, pech, 1"; Expression: "max\_t\_obzhiga"; Minutes; Value Added).
- Decide:
  - "*Raspredelenie konteynerov po ocheredyam*" - определяет, в какую очередь направить очередной контейнер (N-way by Condition; Conditions: "Attribute, t\_obzhiga, <=, 20").
- Assign:
  - "*Opredelenie vremeni obzhiga konteynera*" - генерирует случайное значение атрибута t\_obzhiga – время обжига контейнера – по нормальному закону с параметрами (12, 28).
  - "*Opredelenie naibolshego vremeni obzhiga dlya ocheredi\_1 (ocheredi\_2)*" – определяет значение переменной max\_t\_obzhiga\_1 (max\_t\_obzhiga\_2) – наибольшее время обжига для очередных 3-х контейнеров – по формуле  $MX(max\_t\_obzhiga\_1, t\_obzhiga)$  (для очереди 2 меняется только переменная).
  - "*Opredelenie vremeni obzhiga dlya ocheredi\_1 (ocheredi\_2)*" – присваивает переменной max\_t\_obzhiga значение max\_t\_obzhiga\_1 (max\_t\_obzhiga\_2), которое определяет время, необходимое для обжига блока из 3-х контейнеров, поступающего в цех, затем приравнивает к нулю переменную max\_t\_obzhiga\_1, чтобы правильно определялось наибольшее время обжига для следующего блока из 3-х контейнеров, и устанавливает значение pech\_zanyata в 1, указывающее на то, что печь занята.
  - «*Opredelenie zanyatosti pechi*» - присваивает переменной pech\_zanyata значение 1.
  - "*Opredelenie pribyli i zatrat*" – подсчитывает текущее значение выручки – pribyil ( $Obzhig\ konteynerov.NumberOut * 3 * 100 * 5$ ), затрат на функционирование печи – zatratyi ( $zatratyi + max\_t\_obzhiga * 20$ ), затрат на поддержку очереди – zatratyi\_na\_ochered ( $TNOW * 15$ ), общих затрат – obschie\_zatratyi ( $zatratyi + zatratyi\_na\_ochered$ ), чистой прибыли – chistaya\_pribyil ( $pribyil - obschie\_zatratyi$ ), а также устанавливает значение pech\_zanyata в 0.
- Hold:
  - "*Ozhidanie obzhiga*" – накапливает контейнеры, пока не освободится печь (Scan for

Condition; Condition: " pech\_zanyata==0"), это нужно чтобы во время обжига текущее значение переменной max\_t\_obzhiga не изменялось.

- Batch:

- "*Organizaciya ocheredi\_1 (ocheredi\_2, ocheredi\_3)*" – объединяет в один блок 3-и контейнера ocheredi\_1 (соответственно, ocheredi\_2, ocheredi\_3) (Permanent; 3; Last; Any Entity).

- Dispose:

- "*Obozhyennyye konteynery*" – используется для удаления транзактов – контейнеров из модели.

Для вывода значений переменных прибыли/затрат предусмотрите объекты "*Variable*":

- прибыль от обжига: pribyil;
- затраты на функционирование печи: zratyati;
- затраты на поддержку функционирования очереди: zratyati\_na\_ochered;
- общие затраты: obschie\_zratyati (zratyati+zratyati\_na\_ochered);
- чистая прибыль: chistaya\_pribyil (pribyil-obschie\_zratyati).

Прибыль от обжига:

3 4 5 0 0

Затраты на функционирование печи:

1 0 5 3 4

Затраты на поддержку функционирования очереди:

8 6 3 8

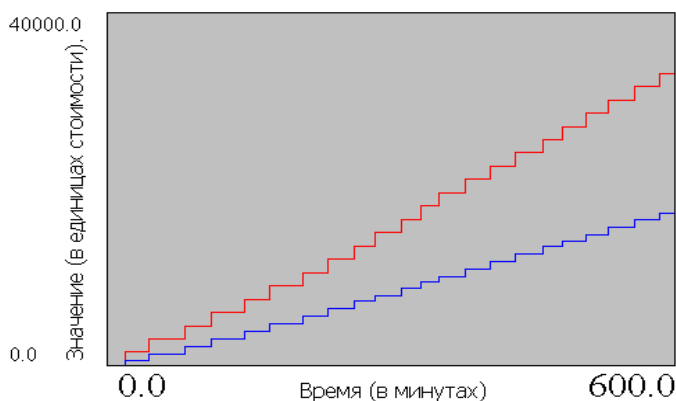
Общие затраты:

1 9 1 7 2

Чистая прибыль:

1 5 3 2 8

Для наглядности во время прогона текущие значения прибыли (выручки) и общих затрат отобразите на графике.



Используя диалоговое окно **Run** ⇒ **Setup** ⇒ **Replication Parameters** ⇒ **Terminating Conditions** создайте условие для останова модели при достижении показателя «Чистая прибыль» определенного количества единиц стоимости (5000, 10000, 15000).

## Вопросы к практической работе № 2

1. Как сохранить вид модели?
2. Опишите назначение и основные свойства модулей Create, Decide, Process, Dispose. Каким образом можно их редактировать?
3. Опишите способы создания соединений между модулями модели.

4. Опишите способы запуска модули, настройки прогона, изменение скорости просмотра модели.
5. Опишите структуру отчетов Arena.
6. Где можно настраивать отображаемые в отчетах параметры?
7. Как создать анимацию для ресурса?
8. Как построить график в Arena?
9. Как настроить анимацию ресурсов модели?
10. Какие блоки используются для создания анимации в Arena?
11. Опишите назначение и основные параметры блока Route.
12. Опишите назначение и основные параметры блока Station.
13. Для чего используется инструмент Seize при создании анимационной схемы?
14. Как задать «набор» ресурсов?