ФГБОУ ВПО «Воронежский государственный
технический университет»

Э. И. Воробьев

МОДЕЛИРОВАНИЕ СИСТЕМ МАССОВОГО ОБСЛУЖИВАНИЯ В ПАКЕТЕ ARENA 9.0

Утверждено Редакционно-издательским советом
университета в качестве учебного пособия

Воронеж 2013

УДК 681.3

Воробьев Э.И. Моделирование систем массового обслуживания в пакете Arena 9.0: учеб. пособие / Э.И. Воробьев. Воронеж: ФГБОУ ВПО «Воронежский государственный технический университет», 2013. 97 с.

В учебном пособии рассматриваются теоретические и практические сведения для изучения основных возможностей моделирования систем в пакете Arena, представляющем собой программный пакет для моделирования информационных процессов и систем на основе аппарата СМО и сетей Петри.

Издание соответствует требованиям Федерального государственного стандарта высшего профессионального образования по направлениям 230100.62 «Информатика и вычислительная техника», профиль «Системы автоматизированного проектирования» и 230400.62 «Информационные системы и технологии», профиль «Информационные системы и технологии», дисциплинам «Методы исследования и моделирования информационных процессов и технологий», «Моделирование процессов и систем».

Учебное пособие подготовлено в электронном виде в текстовом редакторе MS Word 2003 и содержится в файле Пособие, Арена.doc

Табл. 53. Ил. 27. Библиогр.: 8 назв.

Научный редактор д-р техн. наук, проф. Я.Е. Львович

Рецензенты: кафедра вычислительной техники Воронежской государственной лесотехнической академии (зав. кафедрой д-р техн. наук, проф. В.К. Зольников);
д-р техн. наук, проф. О.Ю. Макаров

ВВЕДЕНИЕ

Данное учебное пособие ориентировано на студентов технических специальностей, в специализацию которых входят следующие курсы: «Компьютерное моделирование», «Моделирование систем», «Моделирование экономических процессов», «Математическое моделирование систем», «Моделирование и анализ бизнес-процессов», «Реинжиниринг бизнес-процессов» и другие.

Учебное пособие «Моделирование систем массового обслуживания в пакете Arena 9.0» ориентировано на формирование у студентов навыков и знаний в теории моделирования систем и процессов различной природы с целью последующего их анализа и оптимизации с использованием современных компьютерных технологий.

Теоретическая часть пособия дает сведения об основных понятиях моделирования, о возможных методах классификации моделей. Также рассматриваются методологии структурного анализа: IDEF0, IDEF3 и DFD, и методы и средства имитационного моделирования: сети Петри, системы массового обслуживания.

Особое внимание в этом учебном пособие уделено концепции ARIS, которая за последние 5-10 лет приобретает все большую популярность. Рассмотрена теоретическая часть моделирования в ARIS, также приведен пример, который позволит более глубоко ознакомиться с этим аппаратом моделирования. В качестве математической модели ИС часто используются системы массового обслуживания (СМО). Это системы, которые обслуживают входящий поток заявок. На выходе имеем поток обслуженных заявок. В процессе обслуживания могут создаваться очереди конечной и бесконечной длины. Часть входящих заявок может получить отказ.

Кроме того, различают одноканальные и многоканальные СМО.

Исходные данные для анализа: параметры распределения входящих и исходящих потоков, а также характеристики самой СМО, например среднее время обслуживания. В результате расчетов определяют такие характеристики СМО, как среднее число заявок в системе, средняя продолжительность пребывания заявок в системе, среднее число заявок в очереди, средняя продолжительность пребывания заявок в очереди, средняя длина очереди и т.д.

Такие модели исследуют двумя методами, дающими близкие результаты. Аналитические методы теории СМО позволяют выполнять вероятностные расчеты и вычислять теоретические значения характеристик СМО. Имитационное моделирование позволяет получить приблизительные оценки тех же параметров, причем с увеличением длительности моделирования они приближаются к теоретическим значениям. Имитационное моделирование можно использовать для исследования сложных систем, для которых непосредственное применение теории СМО затруднительно.

Данное методическое пособие описывает работу пакета имитационного моделирования Arena (фирма Rockwell Software).

Система позволяет строить визуализированные имитационные модели, проигрывать их и анализировать результаты.

Одним из наиболее эффективных свойств данного инструмента является его интеграция со средством функционального моделирования BPwin, в котором имеется возможность экспорта диаграммы IDEF3 в имитационную модель Arena.

1.ИМИТАЦИОННОЕ МОДЕЛИРОВАНИЕ СИСТЕМ

1.1. Достоинства и недостатки имитационного моделирования систем

Математические модели могут быть аналитическими, численными, алгоритмическими и имитационными.

Когда явления в системе настолько сложны и многообразны, что аналитическая модель становится слишком грубым приближением к действительности, то исследователь вынужден использовать имитационное моделирование [1,2].

Имитационное моделирование – это метод исследования, заключающийся в имитации на ЭВМ (с помощью комплекса программ) процесса функционирования системы или отдельных ее частей и элементов.

Сущность метода имитационного моделирования заключается в разработке таких алгоритмов и программ, которые имитируют поведение системы, ее свойства и характеристики в необходимом для исследования системы составе, объеме и области изменения ее параметров [3].

При имитационном моделировании реализующий модель алгоритм воспроизводит процесс функционирования системы во времени, причем имитируются явления, составляющие процесс, с сохранением их логической структуры и последовательности протекания во времени, что позволяет по исходным данным получить сведения о состояниях процесса в определенные моменты времени, дающие возможность оценить характеристики системы [4].

Имитационное моделирование позволяет осуществлять многократные испытания модели с нужными входными данными, чтобы определить их влияние на выходные критерии оценки работы системы.

При таком моделировании компьютер используется для численной оценки модели, а с помощью полученных данных рассчитываются ее реальные характеристики.

Имитационное моделирование может применяться в самых различных сферах деятельности. Ниже приведен список задач, при решении которых моделирование особенно эффективно [1,3,5]:

– проектирование и анализ производственных систем;

– оценка различных систем вооружений и требований к их материально-техническому обеспечению;

– определение требований к оборудованию и протоколам сетей связи;

– определение требований к оборудованию и программному обеспечению различных компьютерных систем;

– проектирование и анализ работы транспортных систем, например: аэропортов, автомагистралей, портов и метрополитена;

– оценка проектов создания различных организаций массового обслуживания, например: центров обработки заказов, заведений быстрого питания, больниц, отделений связи;

– модернизация различных процессов в деловой сфере;

– определение политики в системах управления запасами;

– анализ финансовых и экономических систем;

– при подготовке специалистов и освоении новой техники на имитаторах (тренажёрах).

Например, имитационное моделирование может использоваться при рассмотрении производственной компанией возможности постройки больших дополнительных помещений для одного из ее подразделений, если руководство компании не уверено, что потенциальный рост

производительности сможет оправдать затраты на строительство. Невозможно соорудить помещения, а затем убрать их в случае нерентабельности, в то время как моделирование работы производственной компании в ее текущем состоянии и с якобы созданными дополнительными помещениями помогает в решении этой проблемы.

В качестве второго примера можно рассмотреть случай, когда необходимо определить загруженность ресурсов (оборудование или люди) предприятия и принять управленческое решение по закупке нового оборудования или найме/увольнении сотрудников. Реальные действия могут привести к ненужным затратам: купили новое оборудование, а оно простаивает; уволили людей, а в реальности оказалось, что оставшийся персонал не справляется с объемом работы.

Имитационные модели позволяют достаточно просто учитывать такие факторы, как наличие дискретных и непрерывных элементов, нелинейные характеристики элементов системы, многочисленные случайные воздействия и другие, которые часто создают трудности при аналитических исследованиях. В настоящее время имитационное моделирование – наиболее эффективный метод исследования больших систем, а часто и единственный практически доступный метод получения информации о поведении системы, особенно на этапе ее проектирования [3,6].

Несмотря на это, существуют обстоятельства, из-за которых могут получиться неадекватные результаты моделирования:

– нечеткая постановка задачи и цели моделирования;

– недостаточность или неполнота исходных данных для моделирования;

– неверное определение источников и распределений случайных величин в реальных системах;

– недостаточный уровень проработки модели;

– недостаточные знания методологии моделирования;

– неподходящее программное обеспечение для моделирования;

– неправильное использование анимации;

– анализ выходных данных, полученных только в результате одного прогона модели [4,7].

В настоящее время имитационное моделирование широко применяется в мире для исследования сложных систем. Этому способствуют преимущества, присущие этому методу, а именно:

1) Большинство сложных реальных систем с вероятностными параметрами нельзя точно описать с использованием математических моделей.

2) Путем моделирования можно разработать ряд альтернативных вариантов моделей системы и затем определить, какой из них наиболее соответствует исходным требованиям.

3) Имитационное моделирование в ряде случаев гораздо менее затратное, чем проведение экспериментов с реальными системами, тем более что иногда эксперименты на реальных системах в принципе невозможны.

4) Моделирование позволяет изучить длительный интервал функционирования системы в сжатые сроки или, наоборот, изучить более подробно работу системы в развернутый интервал времени [5].

5) При динамическом имитационном моделировании можно получать любое количество оценок вероятностной модели, проводя ее прогоны. Подробное изучение полученных оценок приемлемо использовать при оптимизации модели.

6) Моделирование позволяет оценить некоторые эксплуатационные показатели системы при различных условиях эксплуатации.

Одно из наиболее важных решений, которые приходится принимать разработчикам моделей или системным аналитикам, касается выбора программного обеспечения. Если программное обеспечение недостаточно гибко или с ним сложно работать, то имитационное моделирование может дать неправильные результаты или оказаться вообще невыполнимым.

Использование пакета имитационного моделирования в сравнении с применением универсального языка программирования дает несколько преимуществ:

1) Пакеты имитационного моделирования автоматически предоставляют большинство функциональных возможностей, требующихся для создания имитационной модели, что позволяет существенно сократить время, необходимое для программирования, и общую стоимость проекта.

2) Пакеты имитационного моделирования обеспечивают естественную среду для создания имитационных моделей. Их основные моделирующие конструкции больше подходят для имитационного моделирования, чем соответствующие конструкции в универсальных языках программирования, таких как C.

3) Имитационные модели, которые созданы с помощью пакетов моделирования, как правило, проще модифицировать и использовать.

4) Пакеты имитационного моделирования обеспечивают более совершенные механизмы обнаружения ошибок, поскольку они выполняют автоматический поиск ошибок многих типов. И так как модель не требует большого числа структурных компонентов, уменьшаются шансы совершить какую-либо ошибку [3].

Современные программные пакеты поддерживают следующие функциональные возможности:

– создание анимационных картинок на базе основной модели с целью визуализации и увеличения наглядности;

– генерирование случайных величин с заданными распределениями вероятностей;

– создание независимых прогонов моделей (по набору случайных величин);

– сбор выходных статистических данных и создание отчета по каждому прогону модели, а также общий отчет по всем прогонам;

– определение и изменение атрибутов объектов и глобальных переменных;

– использование заложенных и созданных пользователем математических выражений и функций;

– создание собственных логических конструкций и использование стандартных схем;

– встроенное средство отладки модели с автоматической возможностью поиска ошибок в модели;

– экспорт и импорт данных (входные данные и результаты моделирования);

– имеющаяся хорошо структурированная документация по пакету.

К сожалению, несмотря на неоспоримые достоинства имитационного моделирования, в настоящее время в России этот метод исследования сложных систем используется мало, это связано с тем, что разработка таких моделей требует больших временных и стоимостных затрат. Но тенденции последнего времени вселяют надежду на то, что ситуация изменится и имитационное моделирование в России будут также широко и активно использовать, как в США, Канаде и Европе. Именно для того чтобы компенсировать этот пробел российской действительности, в курсе «Компьютерное моделирование» рассматриваются средства имитационного

моделирования на примере мощного программного пакета (ПП) Arena 9.0.

1.2. Математические основы ПП Arena 9.0

В основе ПП Arena 9.0 заложен математический аппарат систем массового обслуживания (СМО) и сетей Петри. В связи с этим – для лучшего понимания процесса имитации, модулей и их параметров в ПП Arena 9.0 – рассмотрим основы этих математических аппаратов [2].

1.2.1. Системы массового обслуживания

Теория систем массового обслуживания (СМО) появилась в конце 50-х годов наряду с линейным и динамическим программированием, теорией игр и др. из-за активного внедрения в различные сферы деятельности человека вычислительной техники и новых численных методов решения задач. Теория СМО являлась одним из новых разделов теории вероятностей на тот момент.

Исторически считается, что особое влияние на развитие теории СМО оказал датский ученый А. К. Эрланг, труды которого в области проектирования и эксплуатации телефонных станций, послужили толчком к появлению ряда работ в области массового обслуживания [1].

В настоящее время теория СМО является самостоятельным, классическим аппаратом, использующимся во многих средствах имитационного моделирования.

Процесс обслуживания в теории СМО несет широкое значение, под обслуживанием понимается обслуживание клиента, обработка документа, изготовление детали, прием пациента и т.д. В процессе деятельности человека происходят ситуации, когда возникает необходимость в обслуживании

различных требований. При этом под требованием мы будем понимать запрос на удовлетворение какой-либо потребности. Под обслуживанием будем понимать удовлетворение потребности[38]. Основными показателями процесса являются: качество и организация процесса обслуживания. Составление СМО определенной предметной области, ее анализ и выдача рекомендаций по повышению эффективности – вот основные этапы работы с моделью СМО.

Предметом теории массового обслуживания является количественная сторона процессов, связанных с массовым обслуживанием. Целью теории является разработка математических методов для отыскания основных характеристик процессов массового обслуживания для оценки качества функционирования обслуживающей системы.

Система массового обслуживания состоит из одного или более обрабатывающих устройств (сервисов), обслуживающих прибытие сущностей, ещё называемых требованиями или фишками, в систему [5].

Сущности (требования) – это индивидуальные элементы, обрабатываемые в системе. Сущность, находящая сервис занятым, встает в очередь перед сервисом (обрабатывающим устройством). Сущности представляют собой описание динамических процессов в реальных системах.

Они могут описывать как реальные физические объекты, так и нефизические объекты. Сущностями могут быть: клиенты, обслуживаемые в ресторане, больнице, аэропорту; документы, части, которые должны быть обслужены или обработаны. В бизнес-процессах – это документы или электронные отчеты (чеки, заказы, контракты). В производственных моделях, сущностями являются сырье, компоненты или готовая продукция. Кроме этого, под сущностями понимают различные типы объектов, типы

пакетов данных в сети, данные в программных пакетах. В табл. 1.1 приведены элементы СМО.

Таблица 1.1

Основные элементы СМО

Название элемента СМО	Назначение элемента СМО
Генераторы	Генерируют поступление сущностей в систему и временные интервалы их прибытия
Обрабатывающие устройства (сервисы)	Количество обрабатывающих устройств в системе, количество очередей, время обработки одной сущности
Очередь	Правило, по которому обрабатывающее устройство выбирает сущность для обслуживания

В зависимости от поведения сущности, поступившей в систему обслуживания в момент, когда все обрабатывающие устройства заняты, СМО делятся на три группы [8]:

– системы с отказами, или системы с потерями;

– системы с ожиданием, или системы без потерь;

– системы смешанного типа.

В системах с отказами (системах с потерями) любая вновь поступившая сущность на обслуживание, застав все сервисы занятыми, покидает систему. Примером системы с отказами может служить работа автоматической телефонной станции: абонент получает отказ, если необходимая линия связи занята.

В системах с ожиданием (системах без потерь) сущность, поступившая в систему, может её покинуть только после того, как будет обслужена. В таких системах сущности, поступившие в момент, когда все сервисы заняты, образуют очередь. Примером системы обслуживания без потерь является система ремонта техники связи: неисправная техника не может быть использована без ремонта.

В системах смешанного типа сущность, поступившая, когда все сервисы заняты, некоторое время ожидает в очереди, и если за это время не принимается к обслуживанию, то покидает систему. Примером такой системы является обслуживание абонента в переговорном зале междугородной телефонной станции (МТС): абоненту разговор должен быть предоставлен в течение 1 часа. Если за это время разговор не состоялся, то, как правило, абонент покидает МТС.

По числу обрабатывающих устройств (сервисов) различают: одноканальные СМО и многоканальные СМО.

В свою очередь, многоканальные системы могут состоять из однотипных и разнотипных (по пропускной способности) каналов.

По числу сущностей, которые могут одновременно находиться в обслуживающей системе, различают системы с ограниченным и неограниченным потоком требований.

Существуют системы обслуживания, в которых обрабатывающие устройства расположены последовательно (пронумерованы). Очередное требование поступает сначала на первое из них и лишь в том случае, если оно занято, передается второму и т. д. Такие системы называются упорядоченными. Все остальные системы обслуживания, в которых требования распределяются между обрабатывающими устройствами по любому другому принципу, относятся к числу неупорядоченных систем.

По характеру источника сущностей (генератора) различают СМО с конечным и бесконечным количеством требований на входе, соответственно различают замкнутые и разомкнутые СМО. В первом случае в системе циркулирует конечное, обычно постоянное количество требований, которые после завершения обслуживания возвращаются в генератор.

Кроме того, все СМО можно разделить по дисциплине обслуживания [8]. Дисциплина обслуживания определяется правилом, которое устройство обслуживания использует для выбора из очереди следующего требования (если таковые есть)

по завершении обслуживания текущего требования. Обычно используются такие дисциплины очереди:

– FIFO (First-In, First-Out): требования обслуживаются по принципу «первым прибыл – первым обслужен»;

– LIFO (Last-In, First-Out): требования обслуживаются по принципу «последним прибыл – первым обслужен»;

– приоритет: требования обслуживаются в порядке их значимости.

В качестве примеров СМО могут быть следующие системы:

1) Банк (устройства обслуживания – Кассы, требования – Клиенты).

2) Производство (устройства обслуживания – Станки, рабочие, транспортные средства, требования – Детали, комплектующие).

3) Аэропорт (устройства обслуживания – Кассы, взлетно-посадочные полосы, выходы на посадку, пункты регистрации пассажиров, требования – Пассажиры, самолеты).

4) Больница (устройства обслуживания – Доктора, медперсонал, больничные места, медицинское оборудования, требования – Пациенты).

5) Компьютер (устройства обслуживания – Процессор, память, терминалы, требования – Задачи, сообщения).

6) Порт (устройства обслуживания – Причалы, портовые краны, докеры, требования – Прибывающие танкеры, лайнеры).

7) Супермаркет (устройства обслуживания – Тележки, кассы, менеджеры, требования – Покупатели).

Задачи анализа и оптимизации процессов массового обслуживания, которые сейчас стоят перед исследователями, более сложны и требуют реализации дополнительных атрибутов запросов, условий срабатывания, различного времени обслуживания и т. д., в связи с этим, в современных программных средствах и пакетах, в алгоритмах их реализации, заложен комбинированный математический аппарат, например СМО и сети Петри.

1.2.2. Сети Петри

Часто аналитики в задачах моделирования и анализа сложных параллельных и асинхронных систем, обращаются к формальным системам, основанным на использовании математического аппарата сетей Петри. Формальная часть теории сетей Петри, основанная в начале 60-х годов немецким математиком Карлом А. Петри, в настоящее время содержит большое количество моделей, методов и средств анализа, имеющих обширное количество приложений практически во всех отраслях вычислительной техники [2].

Прикладная теория сетей Петри связана главным образом с применением сетей Петри к моделированию систем, их анализу и получающимся в результате этого глубоким проникновением в моделируемые системы [5].

Моделирование в сетях Петри осуществляется на событийном уровне. Определяются, какие действия происходят в системе, какие состояние предшествовали этим действиям и какие состояния примет система после выполнения действия. Выполнения событийной модели в сетях Петри описывает поведение системы. Анализ результатов выполнения может сказать о том, в каких состояниях пребывала или не пребывала система, какие состояния в принципе не достижимы. Однако, такой анализ не дает числовых характеристик, определяющих состояние системы. Развитие теории сетей Петри привело к появлению, так называемых, «цветных» или «раскрашенных» сетей Петри. Понятие цветности в них тесно связано с понятиями переменных, типов данных, условий и других конструкций, более приближенных к языкам программирования.

Таким образом, структура сети Петри задается ориентированным двудольным мультиграфом, в котором одно множество вершин состоит из позиций, а другое множество – из переходов [2, 4, 5], причем множество вершин этого графа разбивается на два подмножества и не существует дуги, соединяющей две вершины из одного подмножества.

Итак, сеть Петри – это набор

$N = (T, P, A)$, $T \cap P = \varnothing$,

где $T = \{t1, t2, ..., tn\}$ – подмножество вершин, называющихся переходами;

$P = \{p1, p2, ..., pm\}$ – подмножество вершин, называющихся позициями (местами);

$A \subseteq (T \times P) \cap (P \times T)$ – множество ориентированных дуг.

В сетях Петри вводятся объекты двух типов: динамические – изображаются метками (маркерами) внутри позиций и статические – им соответствуют вершины сети Петри.

Распределение маркеров по позициям называют маркировкой.

Маркеры могут перемещаться в сети. Каждое изменение маркировки называют событием, причем каждое событие связано с определенным переходом. Считается, что события происходят мгновенно и разновременно при выполнении некоторых условий.

Каждому условию в сети Петри соответствует определенная позиция. Совершению события соответствует срабатывание (возбуждение или запуск) перехода, при котором маркеры из входных позиций этого перехода перемещаются в выходные позиции. Последовательность событий образует моделируемый процесс. Перемещаемые по сети маркеры часто называют фишками.

Основные элементы сети Петри представлены в табл.1.2.

Таблица 1.2

Элементы сетей Петри

Название элемента	Изображение элемента
Позиция (P)	
Переход (T)	
Дуга	

Переходы в сети Петри являются событиями, которые изменяют состояния в реальной системе. На рис. 1.1 приведен пример интерпретации сети Петри.

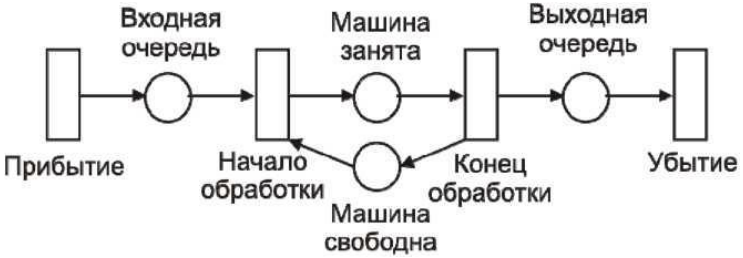


Рис. 1.1. Интерпретация сети Петри

Формальный аппарат сетей Петри предназначен для моделирования систем различного рода и отражает состояния исследуемой системы состоянием сети. Состояние сети Петри определяется ее маркировкой. Количество и распределение фишек сети определяют динамику исследуемой системы. Сеть Петри выполняется посредством запусков переходов в результате удаления фишек из его входных позиций и добавления их в выходные позиции перехода. Последовательность срабатываний переходов полностью определяет поведение сети. Таким образом, сеть Петри описывает структуру системы, ее состояние и поведение [2].

При введении ряда дополнительных правил и условий в алгоритмы моделирования получают различные разновидности сетей Петри. Это необходимо для определения модельного времени, которое позволит моделировать не только последовательность событий, но и их привязку ко времени. В настоящее время выделяют следующие разновидности сетей Петри:

1) Временная сеть Петри – переходы обладают весом, определяющим продолжительность срабатывания (задержку).

2) Стохастическая сеть Петри — задержки являются случайными величинами.

3) Функциональная сеть Петри — задержки определяются как функции некоторых аргументов, например, количества меток в каких-либо позициях, состояния некоторых переходов.

4) Цветная (раскрашенная) сеть Петри — метки могут быть различных типов, обозначаемых цветами, тип метки может быть использован как аргумент в функциональных сетях.

Основными свойствами сети Петри являются:

1) Ограниченность – число меток в любой позиции сети не может превысить некоторого значения K.

2) Безопасность – частный случай ограниченности.

3) Сохраняемость – постоянство загрузки ресурсов.

4) Достижимость – возможность перехода сети из одного заданного состояния (характеризуемого распределением меток) в другое.

5) Живость – возможностью срабатывания любого перехода при функционировании моделируемого объекта.

Среди достоинств аппарата сетей Петри можно указать следующие:

– позволяет моделировать асинхронность и недетерминизм параллельных независимых событий (в сети Петри могут одновременно и независимо друг от друга сработать несколько переходов), конфликтные взаимодействия между процессами;

– позволяет использовать единые методологические позиции для описания программного обеспечения, аппаратных средств и информационного обмена между системами;

– предоставляет возможность введения любой степени иерархической детализации описываемых программных и аппаратных подсистем модели;

– имеет большую анализирующую мощность, которая позволяет формальными средствами доказывать существование или отсутствие определенных состояний сети Петри.

Однако формальная модель сетей Петри, в силу своей универсальности, имеет ряд недостатков, затрудняющих практическое применение для моделирования сложных систем. К основным таким недостаткам можно отнести следующие:

– высокая трудоемкость анализа сетей большой размерности, а реальные бизнес-процессы предприятия моделируются именно сетями большой размерности;

– описательная мощность сетей Петри недостаточна для содержательного моделирования систем;

– обычные сети Петри не отражают требуемые временные характеристики моделируемой системы;

– фишка сети Петри не представляет собой никакой информации, кроме самого факта ее наличия, поэтому чрезвычайно сложно отразить преобразование информации при срабатывании переходов сети Петри;

– невозможность проведения логических преобразований и, как следствие, – невозможность управления продвижением фишек по сети. Недостатки сетей Петри не позволяют описывать сложные системы и в настоящее время используются для описания простейших операций. Также эти факторы явились причиной разработки подклассов и расширений сетей Петри, в которых вводятся определенные ограничения на структуру сети, что позволяет использовать более простые алгоритмы для ее анализа либо дополнительные элементы формальной

системы, призванные увеличить ее описательную мощность. Большого внимания заслуживают сети высокого уровня, такие, как раскрашенные сети Петри (Color Petri Net), являющиеся модификацией сетей Петри и отличающиеся хорошо разработанным математическим аппаратом, широко применяемые для самых разнообразных практических целей. Основной причиной высокой эффективности этих формальных моделей является то, что они без потери возможностей формального анализа позволяют исследователю получить значительно более краткие и удобные описания, чем те, которые могут быть сделаны с помощью сетей низкого уровня. В сетях высокого уровня сложность моделей может быть разделена между

структурой сети, надписями и описаниями. Это позволяет осуществлять описание значительно более сложных систем и анализировать процессы преобразования данных с помощью общепринятых математических выражений вместо сложного набора позиций, переходов и дуг. Раскрашенные сети Петри, в отличие от обычных сетей Петри, позволяют описывать структуру системы в виде иерархии диаграмм. Но у данного аппарата моделирования также не устранен ряд недостатков, которые присущи сетям Петри. К таким недостаткам можно отнести:

– необходимость знания разработчиком специфического языка описания моделей [6];

– отсутствие использования принципов объектно-ориентированного подхода;

– низкую гибкость и трудоемкость описания систем в случае их декомпозиции до уровня некоторых элементарных бизнес-операций. Раскрашенные сети Петри до сих пор применяются для моделирования сложных систем.

Все недостатки СМО и сетей Петри учтены и устранены разработчиками ПП Arena 9.0. Кроме того, этот программный пакет имеет множество необходимых операторов, законов распределения и других элементов, которые привели к его широкому распространению.

Хотелось бы добавить несколько слов о том, почему Arena 7.0 является программным пакетом. Это связано с тем, что Arena 7.0, кроме основного модуля моделирования и анализа систем, имеет следующие встроенные программные средства:

1) nput Analyzer. Это средство позволяет анализировать входные данные, определять закономерности входных данных для дальнейшего их использования при моделировании систем.

2) Output Analyzer. Это средство позволяет анализировать выходные данные, полученные в результате проведенных экспериментов с моделью.

3) Process Analyzer. Меняет значения параметров модели, структуру модели, занятость ресурсов, их полезность и т. д.,

сравнивает альтернативные сценарии и выбирает тот сценарий, который имеет наилучший результат. Сравнивая эти сценарии работы модели, можно определить лучшее решение (но не оптимальное, т. к. нельзя просмотреть все возможные решения, т. е. исследовать полностью область допустимых решений), но все-таки определить лучшее решение таким способом возможно.

4) Генератор отчетов. Выводит данные по результатам моделирования в виде текстовых данных, графиков, диаграмм.

5) Visio Process Analyzer.

6) OptQuest. Является инструментом оптимизации задач, предназначен и специально настроен для анализа результатов моделирования, выполненного с помощью пакета Arena.

Система имитационного моделирования Arena – основной программный продукт Systems Modeling. Корпорация Systems Modeling была основана в 1982 г. Деннисом Педгеном, автором SIMAN – первого промышленно-ориентированного общецелевого языка имитационного моделирования. В настоящее время область деятельности Systems Modeling включает в себя имитационное моделирование и разработку технологического программного обеспечения [8].

Система Arena позволяет моделировать виды деятельности, представленные на рис. 1.2.

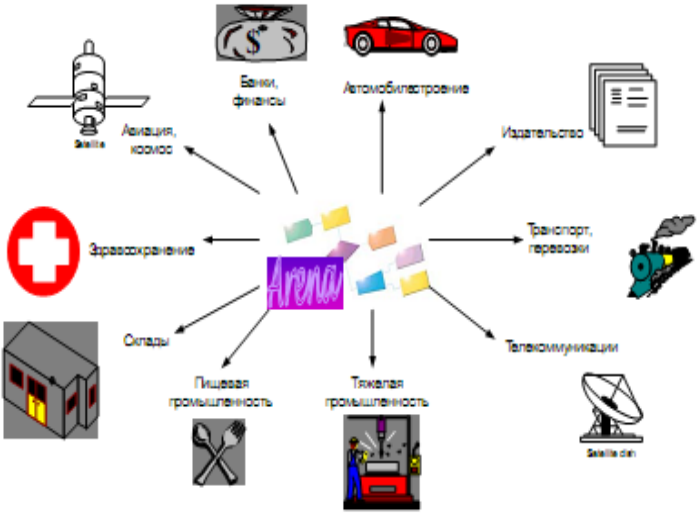


Рис. 1.2. Области применения Arena

С помощью Arena можно достичь основных целей моделирования сложных систем:

– понять, как устроен исследуемый объект: какова его структура, основные свойства, законы развития и взаимодействие с окружающей средой;

– выявить «узкие места» в материальных, информационных и других потоках;

– выделить переменные, наиболее важные для успешного функционирования моделируемой системы, и проанализировать имеющиеся между ними связи;

– научиться управлять системой, определять наилучшие способы управления при заданных целях и критериях;

– прогнозировать прямые и косвенные последствия реализации заданных форм и способов воздействия на систему.

Данные по моделированию в ПП Arena, модулям, их свойствам, можно найти в [6,7].

1.3. Начало работы с программным пакетом Arena 9.0

Для того чтобы создать новую модель, необходимо открыть ПП Arena 9.0 через Пуск → Rockwell Software → Arena7.0 → Arena7.0.1. После запуска Arena автоматически открывается новый файл. Модули помещаются на панель методом «drug & drop», соединяются с помощью коннектора. Если модуль остается «горячим» (т. е. выделенным), то при помещении нового модуля на рабочую область (окно блок-схемы) эти модули автоматически соединяются друг с другом. Среда моделирования Arena представлена на рис. 1.3.

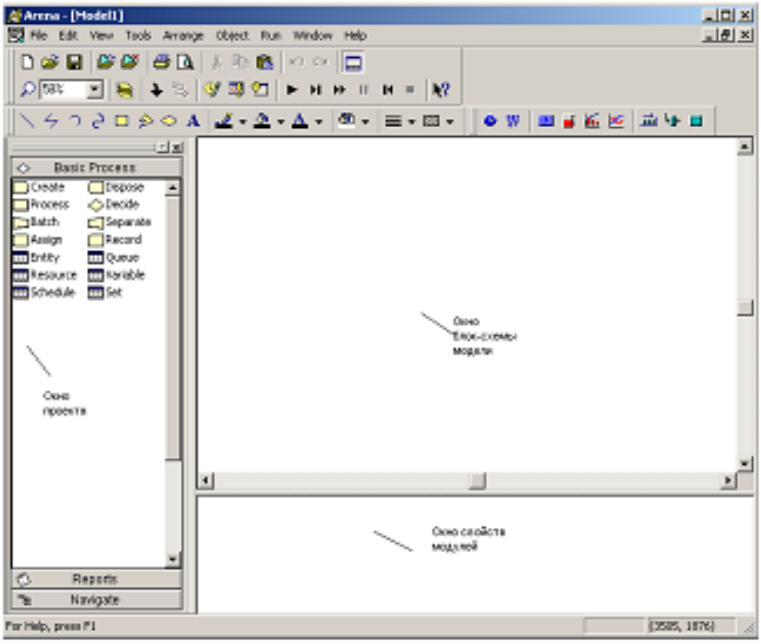


Рис. 1.3. Среда моделирования Arena

Окно приложения разделено на три области:

1) Окно рабочего поля модели, в котором описывается логика модели с использованием схемных (графических) мо-

дулей. Окно рабочего поля представляет графику модели, включая блок-схему процесса, анимацию и другие элементы.

2) Окно свойств модулей, в котором отображаются свойства всех модулей (как модулей данных, так и схемных), имеющихся и используемых в модели.

3) Окно проекта – это навигатор системы, в котором отображается рабочая панель со всеми модулями и другие доступные и открытые панели.

Окно проекта включает в себя несколько панелей:

1) Basic Process Panel (панель основных процессов) – содержит модули, которые используются для моделирования основной логики системы.

2) Advanced Process Panel (панель усовершенствованных процессов) – содержит дополнительные модули для создания моделей со сложной логикой процесса.

3) Advanced Transfer Panel (панель перемещения) – содержит специально разработанные блоки для моделирования процесса перемещения объектов с помощью транспортера или конвейера.

4) Reports (панель отчетов) – панель сообщений: содержит сообщения, которые отображают результаты имитационного моделирования.

5) Navigate (панель навигации) – панель управления позволяет отображать все виды модели, включая управление через иерархические подмодели.

Таким образом, для того чтобы разрабатывать имитационные модеи с использованием ПП Arena, необходимо изучить 3 основные панели:

Basic Process Panel, Advanced Process Panel и Advanced Transfer Panel. Каждая из этих панелей состоит из двух типов модулей: схемных модулей (Flowchart Modules) и модулей данных (Data Modules).

Рассмотрим более подробно состав каждой панели, свойства и назначение каждого модуля.

1.4. Basic Process Panel (панель основных процессов)

1.4.1. Схемные модули

Модуль «Create» является отправной точкой для сущностей в имитационной модели. Сущности – это индивидуальные элементы, обрабатываемые. Создание сущностей модулем происходит по расписанию или же, основываясь на значении времени между прибытиями сущности в модель. Покидая модуль, сущности начинают обрабатываться в системе. Тип создаваемых сущностей определяется в этом модуле.

Применение: прибытие различных документов в сфере бизнеса (например: заказы, чеки, документация); прибытие клиентов в сфере обслуживания (например: в ресторан, в магазин); начало изготовления продукции на производственной линии. Параметры модуля Create приведены в табл.1.3.

Таблица 1.3

Параметры модуля Create

Параметры	Описание
Name	Уникальное имя модуля, которое будет отражено в блок-схеме
Entity Type	Название типа сущности, который будет создаваться модулем
Type	Способ формирования потока прибытия. Type может иметь значения: Random (используется экспоненциальное распределение со средним значением, определенным пользователем), Schedule (определяется модулем Schedule), Constant (будет использоваться постоянное значение, определенное пользователем) или Expression (поток прибытия будет формироваться по определенному выражению)

Продолжение табл.1.3

Value	Определяет среднее значение времени между прибытиями сущностей
Schedule Name	Имя расписания, которое определяет характер прибытия сущности в систему
Expression	Этот параметр задает тип распределения или любое выражение, определяющее время между прибытиями сущностей в модель
Units	Единицы измерения времени между прибытиями (день, час, минута, секунда)
Entities per arrival	Количество сущностей, входящих в систему за одно прибытие
Max arrivals	Максимальное число сущностей, которое может создать этот модуль (ресурс генератора)
First Creation	Время, через которое прибудет первая сущность в модель, от начала моделирования

Модуль «Process» является основным модулем процесса обработки сущностей в имитационной модели. В модуле имеются опции использования ресурсов, т. е., как и при любой обработке, захватываются какие-то ресурсы. Кроме стандартного модуля Process, можно использовать подмодель, придавая ей особую, определенную пользователем, иерархическую логическую схему. В модуле можно также задавать добавочные стоимостные и временные характеристики процесса обработки сущности.

Наиболее частое применение модуля Process: проверка документов; выполнение заказов; обслуживание клиентов; обработка деталей. В табл.1.4 приведены параметры модуля Process.

Таблица 1.4

Параметры модуля Process

Параметры	Описание
Name	Уникальное имя модуля, которое будет отражено в блок-схеме
Type	Определяет логическую схему модуля. Standard означает, что логическая схема находится внутри модуля и зависит от параметра Action. Submodel показывает, что логическая схема будет находитья ниже в иерархической модели. Подмодель может содержать любое количество логических модулей

Продолжение табл. 1.4

Action	Тип обработки, происходящей внутри модуля, может быть четырех типов: Delay просто показывает, что процесс занимает какое-то время и не отражает использование ресурсов; Seize Delay указывает на то, что в этом модуле были размещены ресурсы и будет происходить их захват и задержка, ресурсы будут захватываться (т. е. будут заняты обработкой сущности), а их освобождение будет происходить позднее с помощью какого-то другого модуля; Seize Delay Release указывает на то, что ресурсы были захвачены, а затем (через время) освободились, и Delay Release означает, что ресурсы до этого были захвачены сущностью, а в таком модуле сущность задержится и освободит ресурс. Все эти параметры доступны только тогда, когда Type = Standard
Priority	Значение приоритета модулей, использующих один и тот же ресурс где угодно в модели. Это свойство недоступно, если Action = Delay (или Delay Release) или когда Type = Submodel
Resources	Определяет ресурсы или группы ресурсов, которые будут обрабатывать сущности в этом модуле
Delay Type	Тип распределения или процедура, определяющая параметры задержки
Units	Единицы измерения времени задержки (день, час, минута, секунда)
Allocation	Определяет стоимостные характеристики обработки. Value Added означает учитывать стоимостные характеристики, а Non-Value Added - не учитывать

Продолжение табл.1.4

Minimum	Поле, определяющее минимальное значение для равномерного и треугольного распределения
Maximum	Поле, определяющее максимальное значение для равномерного и треугольного распределения
Value	Поле, определяющее среднее значение для нормального и треугольного распределения или значения для постоянной временной задержки
Std Dev	Параметр, определяющий стандартное отклонение для распределения
Expression	Поле, в котором задается выражение, определяющее значение временной задержки, если Delay Type = Expression

Более подробно остановимся на параметре Priority (приоритет) модуля Process. Говоря об этом параметре, мы должны ввести понятие «приоритет ресурса» и «приоритет очереди». Рассмотрим пример и объясним, что такое «приоритет ресурса».

На прием к доктору приходят пациенты двух типов: взрослые и дети. Доктор (наш ресурс) – один. Он ведет прием и детей, и взрослых, но детей доктор принимает около 30 минут, а взрослых около 20 минут, причем у детей приоритет выше, чем у взрослых. Каким образом мы можем реализовать это с помощью модуля Process? Во-первых, параметр Action этого модуля должен быть установлен Seize Delay Release для назначения ресурса, т. е. когда сущность «пациент» зайдет в модуль, то она захватит ресурс «доктор» на определенное время. Во-вторых, у нас по условию время обслуживания пациентов различное; таким образом, мы процесс обслуживания пациентов доктором смоделируем в виде двух блоков Process с разными временными задержками (в 30 и 20 минут), но одним

и тем же ресурсом «док-тор». В-третьих, чтобы установить приоритет у детей выше, мы в параметре Priority в том процессе, где время обслуживания 30 минут, т. е. обслуживание детей, установим приоритет – High, а во втором процессе – Low или Medium. Таким образом, когда у нас будут приходить сущности «дети», они будут иметь наивысший приоритет в обслуживании. Рассмотрение понятия «приоритет очереди» будет приведено ниже (см. модуль данных очередь Queue).

Модуль «Decide» позволяет описать и задать логику модели, учитывая принятие решений. Он включает опции принятия решений, основанных на условии By Condition (например, если тип сущности Car) или основанных на вероятности By Chance (например, 75 % – true, а 25 % – false). Условия могут быть основаны на значении атрибута Attribute, значении переменной Variable, типе сущности Entity Type или основанные на выражении Expression.

Если поставленное условие выполняется, то сущности будут покидать модуль через ветку True, иначе – по ветке False. Данный модуль позволяет выполнять проверку не только одного условия, но и нескольких. Это достигается с помощью свойства Type → N-way by Chance/by Condition. В зависимости от условия сущность идет по нужной ветке. Таким образом, по ветке True у модуля может быть любое количество выходов (по ветке False – всегда один выход).

Применение: разделение дел на срочные дела и несрочные; перенаправление недоделанных или сделанных неправильно работ на доработку. Параметры модуля Decide приведены в табл.1.5.

Таблица 1.5

Параметры модуля Decide

Параметры	Описание
Name	Уникальное имя модуля, которое будет отражено в блок-схеме

Продолжение табл.1.5

Type	Тип принятия решения: By Chance - выбор направления основывается на вероятности и By Condition - проверка на выполнение конкретно заданного условия
Percent True	Значение, определяющее процент сущностей, который пойдет по направлению True
If	Тип условия, которое будет проверяться на вы-
Named	Имя переменной, атрибута или типа сущности, который будет проверяться при входе сущности в
Is	Математический знак условия, например: больше, меньше, равно и т. д.
Value	Значение, с которым будет сравниваться атрибут или переменная пришедшей сущности. Если тип условия - Expression, то в выражении должен стоять знак условия, например Color<> Red

Модуль «Batch» отвечает за механизм группировки сущностей в имитационной модели. Группировка может быть постоянной или временной. Временно сгруппированные комплекты сущностей позднее могут быть разъединены с помощью модуля Separate. Комплекты могут состоять из любого числа входящих сущностей, определенного пользователем, или же сущности могут объединяться в комплект в зависимости от атрибута сущности. Временные и стоимостные характеристики выходящей сущности, представляющей комплект, будут равны сумме характеристик вошедших в группу сущностей.

Сущности прибывают в модуль, становятся в очередь и остаются там до тех пор, пока в модуле не будет набрано заданное количество сущностей. Когда соберется нужное число сущностей, создается сущность, представляющая комплект.

Применение: собрать необходимое количество данных, прежде чем начинать их обработку; собрать ранее разделенные копии одной формы; соединить пациента и его больничную карту приема к врачу.

В табл.1.6 приведены параметры модуля Batch

Таблица 1.6

Параметры модуля Batch

Параметры	Описание
Name	Уникальное имя модуля, которое будет отражено в блок-схеме
Type	Способ группировки сущностей может быть: Temporary (временная) и Permanent (постоянная)
Batch Size	Число сущностей, образующих один комплект
Rule	Определяет, по какому признаку будут группироваться. Если Rule = Any Entity, - это значит, что первые 3 (если Batch Size = 3) сущности будут сгруппированы. Если Rule = By Attribute, то будет объединяться заданное количество сущностей с определенным атрибутом. Например, если Attribute Name = Color, то все сущности, имеющие одинаковое значение атрибута Color, будут сгруппи-
Attribute Name	Имя атрибута, по значению которого будут группироваться сущности

Модуль Separate может использоваться в двух возможных вариантах:

1) Для создания копий входящих сущностей. Если модуль создает копии сущностей, то пользователь может задать количество дубликатов сущности. У дублированной сущности значения атрибута, а также анимационная картинка такие же,

как и оригинала. Оригинальная сущность также покидает модуль.

2) Для разделения ранее сгруппированных сущностей. Правило для разделения стоимостных и временных характеристик копий сущностей и разделенных сущностей определяется пользователем. Когда временно сгруппированные сущности прибывают в модуль, они раскладываются на составные сущности. Сущности покидают модуль в той же последовательности, в которой они добавлялись в комплект.

Применение: разъединение ранее сгруппированных комплектов документов; для параллельной обработки счетов и документов; для параллельной обработки счетов и документов по одному заказу. В табл.1.7 приведены параметры модуля Separate.

Таблица 1.7

Параметры модуля Separate

Параметры	Описание
Name	Уникальное имя модуля
# of Duplic	Количество создаваемых копий входящей сущности
Type	Способ разделения входящей в модуль сущности. Duplicate Original - просто делает дубликаты входящей сущности. Split Existing Batch проводит разгруппировку
Allocation Rule	Метод разделения стоимости и времени, если выбран Type=Split Existing Batch. Retain Original Entity Values сохраняет оригинальные значения сущностей. Take All Representative Values - все сущности принимают одинаковое значение. Take Specific Representative Values - сущности принимают специфическое значение

Модуль Assign предназначен для задания нового значения переменной, атрибуту сущности, типу сущности, анимационной картинке сущности или другой переменной в системе.

В одном модуле можно сделать только любое количество назначений: сменить тип сущности, ее картинку, задать любое количество переменных и т. д.

Пример применения модуля Assign: установление приоритета для клиентов; присвоение номера вышедшему приказу. Параметры модуля Assign приведены в табл.1.8.

Таблица 1.8

Параметры модуля Assign

Параметры	Описание
Name	Уникальное имя модуля, которое будет отражено в блок-схеме
Type	Тип назначения, которое будет осуществляться. Other может включать в себя встроенные переменные, такие, как вместимость ресурса или конечное время моделирования
Variable Name	Имя переменной, которая будет изменяться в этом модуле
Attribute Name	Имя атрибута, который будет изменяться в этом модуле
Entity Type	Новый тип сущности, присваиваемый сущности в этом модуле
Entity Picture	Новая анимационная картинка для сущности, прошедшей этот модуль
New Value	Присваиваемое новое значение для атрибута, переменной

Модуль Record предназначен для сбора статистики в имитационной модели. Модуль может собирать различные типы статистики, включая время между выходами сущностей из модуля, статистику сущности (время цикла, стоимость), статистику за период времени (период времени от заданной точки до текущего момента). Также доступен количественный тип статистики. Частое применение модуля: подсчитать, какое количество заказов было выполнено с опозданием; подсчитать количество работы, совершаемое за один час. Параметры модуля Record приведены в табл.1.9.

Таблица 1.9

Параметры модуля Record

Параметры	Описание
Name	Уникальное имя модуля, которое будет отражено в блок-схеме
Type	Определяет тип статистики, которая будет собираться. Count будет увеличивать или уменьшать статистику на заданное значение**.** Entity Statistics будет собирать общую статистику о сущности, например: время цикла, стоимостные характеристики и т. д. Time Interval будет считать разницу между значением атрибута и текущим временем моделирования. Time Between будет отслеживать время между вхождением сущностей в модуль. Expression будет просто фиксировать значение, определяемое выражением
Attribute Name	Имя атрибута, значение которого будет использоваться для интервальной статистики
Value	Значение, которое будет добавляться к статистике, когда в модуль будет прибывать сущность

Этот модуль является выходной точкой из имитационной модели. Статистика о сущности может собираться до того момента, пока она не выйдет из системы.

Модуль Dispose является выходной точкой из имитационной модели. Применение: окончание бизнес-процесса; клиенты покидают отдел. В табл.1.10 приведены параметры модуля Dispose.

Таблица 1.10

Параметры модуля Dispose

Параметры	Описание
Name	Уникальное имя модуля, которое будет отражено в блок-схеме
Record Entity Statistics	Определяет, будет ли вестись статистика о выходе сущности из системы

1.4.2. Модули данных

Все модули данных в навигаторе панелей имеют одинаковый вид, т. к. они не отображаются физически в блок-схеме модели, в связи с этим их изображение не приводится. Также мы не будем рассматривать стоимостные параметры модулей, т. к. они не влияют на логику модели.

Модуль Entity определяет тип сущности и ее анимационную картинку в имитационном процессе, также определяет стоимостную информацию. Для каждого источника должен быть определен тип сущности, который он генерирует.

Применение модуля Entity: документы (факсы, письма, отчеты и т. д.); люди в моделях больницы или магазина.

В табл.1.11 приведены параметры модуля Entity.

Таблица 1.11

Параметры модуля Entity

Параметры	Описание
Entity Type	Название типа сущности
Initial Picture	Графическое представление сущности в начале имитационного процесса. Это значение может быть впоследствии изменено с помощью модуля Assign. Просмотреть анимационные картинки можно так: Edit/ Entity picture

Модуль данных Queue предназначен для изменения правила расстановки сущностей в очереди, т. е. задается правило обслуживания сущности в процессе. По умолчанию тип очереди First in First out.

Применение: стопка документов, ожидающих освобождения ресурса; место для собирания частей, ожидающих упаковки (группировки). В табл.1.12 приведены параметры модуля Queue.

Таблица 1.12

Параметры модуля Queue

Параметры	Описание
Name	Уникальное имя модуля, которое будет отражено в блок-схеме
Attribute Name	Имя атрибута, значение которого будет учитываться, если тип = Lowest Attribute Value или Highest Attribute Value

Продолжение табл.1.12

Type	Правило расстановки сущностей в очереди: First in First out - первый вошел, первый вышел; Last in first out - последний пришел, первый вышел; Lowest Attribute Value - первый выйдет из очереди тот, значение атрибута у которого низшее; ***Highest Attribute Value*** - первый выйдет из очереди тот, значение атрибута у которого наивысшее

Более подробно хотелось бы остановиться на параметре Type, т. к. именно с помощью него можно определить, что такое «приоритет очереди» и как его необходимо задавать. Рассмотрим несколько изменный наш пример.

На прием к доктору приходят пациенты двух типов: взрослые и дети. Доктор (наш ресурс) – один. Он ведет прием и детей, и взрослых, причем время приема одинаково (около 30 минут), но у детей приоритет при обслуживании выше, чем у взрослых.

Каким образом мы это можем реализовать? Во-первых, в модуле Process задается ресурс «доктор»; с помощью параметра Action, который устанавливаем Seize Delay Release для назначения ресурса. Таким образом, когда сущность «пациент» зайдет в модуль процесс, то она захватит ресурс «доктор» на определенное время (около 30 минут). Во-вторых, у нас по условию время обслуживания пациентов одинаковое, таким образом, мы процесс обслуживания пациентов доктором смоде-лируем в виде одного блока Process, с временной задержкой в 30 минут.

Но здесь возникает вопрос: каким образом задать приоритет? В данном случаем, мы рассматриваем ситуацию, когда ресурс задан в одном блоке, т. е. нет смысла менять параметр Priority модуля Process. В этом случае, возникает ситуация, когда приоритет не ресурса, а приоритет очереди. И задается

он в модуле Queue. Необходимо выбрать, у какого типа сущности он выше. Это производится с помощью параметра Type:

Lowest Attribute Value – первый выйдет из очереди тот, значение атрибута у которого низшее, или Highest Attribute Value – первый выйдет из очереди тот, значение атрибута у которого наивысшее. Таким образом, когда у нас будут приходить сущности «дети», они будут иметь наивысший приоритет в обслуживании.

Модуль Resource предназначен для определения ресурсов и их свойств в имитационном процессе; кроме того, модуль включает в себя стоимостную информацию о ресурсах и вместимость ресурсов. Ресурсы могут иметь фиксированную вместимость или же основанную на расписании.

У ресурсов с фиксированной вместимостью в течение имитационного процесса вместимость изменяться не может. Ресурс должен быть связан с каким-либо процессом.

Применение: люди (клерки, продавцы, бухгалтеры, рабочие и т. д.); оборудование (телефонная линия, станок, компьютер). Параметры модуля Resource приведены в табл.1.13.

Таблица 1.13

Параметры модуля Resource

Параметры	Описание
Name	Имя ресурса
Type	Метод, определяющий вместимость ресурса. Fixed Capacity - фиксированная вместимость ресурса. Based on Schedule - вместимость ресурса определяется модулем Schedule
Capacity	Число ресурсов, находящихся в системе
Schedule Name	Имя Schedule модуля, который определяет вместимость ресурса, если Type = Based on Schedule

Продолжение табл. 1.13

Busy / Hour	Почасовая стоимость обработки сущности ресурсом. Время учитывается только тогда, когда ресурс занят обработкой, и прекращает учитываться, когда ресурс освобождается
Idle / Hour	Стоимость ресурса, когда он не занят
Per Use	Стоимость обработки ресурсом одной сущности (не зависит от времени)

Модуль Schedule может использоваться вместе с модулем Resource для определения вместимости ресурса и с модулем Create – для задания расписания прибытия сущностей.

Применение: расписание работы персонала с перерывами на обед; значение покупателей, прибывающих в супермаркет.

Параметры модуля Schedule приведены в табл.1.14.

Таблица 1.14

Параметры модуля Schedule

Параметры	Описание
Name	Название расписания
Type	Тип расписания, который может быть Capacity (расписание для ресурсов), Arrival (для модуля Create) или Other (разнообразные временные задержки или факторы)
Time Units	Масштаб оси времени в графике расписания

Модуль данных Set, который описывает группу ресурсов, использующихся в модуле Process. В группе могут находиться несколько ресурсов. Модуль Set автоматически создает ресурсы, вместимость которых по умолчанию равна 1, и без всякой стоимостной информации. Следовательно, если для ресурсов, входящих в группу, не нужно стоимостной информации и вместимость более 1, то можно обойтись созданием только модуля Set.

Возможно применение модуля для организации работы группы работников, например по очереди. Параметры модуля Set приведены в табл.1.15.

Таблица 1.15

Параметры модуля Set

Параметры	Описание
Name	Название группы
Members	Перечисляет ресурсы, входящие в группу. Порядок перечисления ресурсов важен, когда в модуле Process используется правило выбора Cyclical или Preferred Order
Resource Name	Названия ресурсов, входящих в группу

Модуль Variable определяет значение переменных. Переменные, относящиеся к модулю Decide или Assign, могут использоваться в выражениях. Если переменная не описана в этом модуле, то ее первоначальное значение равно 0. Применение: число документов обрабатываемых в час; присвоение серийного номера для идентификации продукции.

В табл.1.16 приведены параметры модуля Variable.

Таблица 1.16

Параметры модуля Variable

Параметры	Описание
Name	Имя переменной
Initial Value	Первоначальное значение переменной. Это значение в последствии может меняться модулем Assign
Rows	Число строк в размерной переменной
Columns	Число столбцов в размерной переменной
Clear Option	Определяет время, когда значение переменной сбрасывается в начальное значение. Statistics - сбрасывает переменную в начальное значение в любой момент, когда статистика была расчищена. System - сбрасывает переменную в начальное значение в любой момент, когда система была расчищена. None - никогда не сбрасывает переменную в начальное значение, исключая предшествующую первой репликации
Statistics	Определяет, будет ли вестись статистика по этой переменной

1.5. Advanced Process Panel (панель усовершенствованных процессов)

1.5.1. Схемные модули

Модуль Hold удерживает (захватывает) сущности. Процесс удержания может продолжаться до бесконечности или до выполнения условия.

Применение модуля: складируются детали; пассажиры ожидают транспорт на остановке.

Таблица 1.17

Параметры модуля Hold

Параметры	Описание
Name	Уникальное имя модуля, которое будет отражено в блок-схеме
Тур	Тип удержания сущности: Infinite Hold (удерживает до бесконечности, в этом случае у блока нет выхода**),** Scan of Condition (ожидает срабатывания определенного условия), Wait of Signal (ожидает сигнала, который вырабатывается только модулем Signal)

В табл.1.17 приведены параметры модуля Hold.

Если у модуля тип Infinite Hold, то забрать сущность из блока можно другими специальными модулями: Remove, Signal или Pickup. Соответственно, сущность выйдет по ветке именно из этих модулей, а не из Hold.

Поля Queue Type и Queue name присутствуют среди параметров модуля Hold всегда, задаются чаще всего автоматически (менять не рекомендуется).

Если тип имеет значение Wait for signal, то появляются поля Wait for value и Limit (ожидание конкретного значения сигнала и предел количества сущностей для освобождения из модуля Hold).

Если тип принимает значение Scan of Condition, то в этом случае становится доступным поле Condition, т. е. задержка напрямую зависит от выражения, заданного в этом поле.

Модуль Signal

Этот модуль посылает значение сигнала каждому модулю Hold в модели, в котором установлен тип Wait for signal, и освобождает заданное число сущностей.

Когда сущность прибывает в модуль Signal, то вырабатывается сигнал и посылается код сигнала в систему. В это время сущности в модуле Hold, который ожидает этого же сигнала, удаляются из очереди Hold и выходят из модуля.

Применение: прием преподавателем экзамена у определенного количества студентов; ожидание людьми определенного автобуса. Параметры модуля Signal приведены в табл. 1.18.

Таблица 1.18

Параметры модуля Signal

Параметры	Описание
Name	Уникальное имя модуля, которое будет отражено в блок-схеме
Signal value	Значение посылаемого сигнала для модуля Hold
Limit	Число сущностей, которые будут освобождены из модуля Hold, когда сигнал будет получен

Модуль Pickup предназначен для удаления определенного количества последовательно стоящих сущностей из определенной очереди. Сущности, которые удаляются из очереди, добавляются в конец сущности, вошедшей в блок Pickup. Чаще всего используется для удаления сущностей из модуля Hold при условии, что тип Infinity Hold (без выхода). В модуле Pickup задается имя очереди, из которой будут забираться сущности, и определяется количество забираемых сущностей.

Все сущности (вместе с исходной) выйдут из модуля Pickup в виде временной группировки.

Применение: развоз товаров по магазинам со склада; посадка пассажиров в автобус на автобусной остановке.

Параметры модуля Pickup приведены в табл.1.19

Таблица 1.19

Параметры модуля Pickup

Параметры	Описание
Name	Уникальное имя модуля, которое будет отражено в блок-схеме
Quantity	Количество сущностей, которые должны быть удалены из очереди
Queue Name	Имя очереди, из которой будут удаляться сущности
Starting Rank	Позиция сущностей в очереди, с которой начинается удаление

Модуль Remove предназначен для удаления сущностей из любой очереди при условии, что эти сущности задерживаются бесконечно (Infinity).

Отличие этого модуля от других заключается в том, что он может забрать только одну сущность из очереди. И у этого модуля 2 выхода: original и removed entity. По ветке original выходит та сущность, которая зашла (активировала) в этот модуль, а по ветке removed entity выходит та сущность, которая была забрана из очереди другого модуля (чаще всего модуля Hold).

Параметры модуля Remove приведены в табл.1.20.

Таблица 1.20

Параметры модуля Remove

Параметры	Описание
Name	Уникальное имя модуля, которое будет отражено в блок-схеме
Queue name	Название очереди, из которой будет произведено удаление
Rank of entity	Глубина удаления (количество сущностей для удаления)

Модуль Match предназначен для синхронизации движения двух или более сущностей, расположенных в различных, несвязанных очередях.

Количество очередей может варьироваться от двух до пяти. Сущность ждет в очереди до тех пор, пока в остальных очередях не появятся другие сущности либо с таким же значением атрибута, как и у исходной сущности.

Применение: сборка частей детали для дальнейшей обработки; собирание различных, но строго определенных продуктов по заказу клиента; синхронизация выхода покупателя с выходом заполненного заказа. В табл.1.21 приведены параметры модуля Match

Таблица 1.21

Параметры модуля Match

Параметры	Описание
Name	Уникальное имя модуля, которое будет отражено в блок-схеме
Number to Match	Количество очередей для синхронизации сущностей

Продолжение табл.1.21

Type	Метод сравнивания входящих сущностей для синхронизации. Значения: Any Entities - в каждой очереди должно быть по одной любой сущности, для того чтобы они вышли. Based on Attribute - в каждой очереди должна быть хотя бы одна сущность с таким же атрибутом для выхода
Attribute Name	Название атрибута, по которому сущности должны сравниваться. Используется, только если установлен тип Based on Attribute

Модуль Dropoff перемещает определенный набор сущностей из группы сущностей и посылает их в другой модуль, связанный с ним графическим соединением.

В этот модуль приходит временная группировка, из которой мы можем выделить требуемое количество сущностей, они пойдут по ветке Members, оставшаяся группа (в виде одной сущности) пойдет по ветке Original.

В этот модуль приходит временная группировка, из которой мы можем выделить требуемое количество сущностей, они пойдут по ветке Members, оставшаяся группа (в виде одной сущности) пойдет по ветке Original. Параметры модуля Dropoff приведены в табл.1.22.

Таблица 1.22

Параметры модуля Dropoff

Параметры	Описание
Name	Уникальное имя модуля, которое будет отражено в блок-схеме
Quantity	Число сущностей, которое будет выбрано из всех пришедших в группу сущностей
Starting Rank	Начальное значение выбрасываемой сущности

Продолжение табл.1. 22

Member Attributes	Метод определения того, как назначить значение атрибута представленной сущности (такие как стоимость, время) для выброса оригинальных сущностей
Attribute Name	Название атрибутов сущности, которые обозначены для выброса оригинальной сущности из группы

Модуль Search необходим для поиска определенного элемента в очереди, в пакете либо в каком-то выражении. Он имеет два выхода: True, если элемент найден, и False, если элемент не найден.

Применение: поиск среди коробок самой легкой. Параметры модуля Search приведены в табл.1.23.

Таблица 1.23

Параметры модуля Search

Параметры	Описание
Name	Уникальное имя модуля, которое будет отражено в блок-схеме
Type	Тип поиска: среди сущностей, объединенных в очередь, сущности, объединенные в пакет, или поиск выражения
Queue Name	Имя очереди, в которой будет осуществляться поиск
Starting Value	Начальный класс в очереди или в пакете или начальное значение для переменной в выражении
Ending Value	Конечный класс в очереди или в пакете или конечное значение для переменной в выражении
Search condition	Условия, включающие индекс поиска выражений, или содержащие атрибут при поиске пакетов или сущностей в очереди

Модуль Seize предоставляет сущности один или несколько ресурсов. Он может быть использован для того, чтобы захватывать отдельный ресурс, ресурс из набора ресурсов или ресурс, определённый альтернативным методом, таким как атрибут или выражение.

Когда сущность поступает в этот модуль, она ждёт в очереди, пока определённые в этом модуле ресурсы не будут доступны. Также здесь определяется тип распределения ресурсов для поступивших сущностей.

Замечания:

1) Сущности, которые захватываются с более высокой величиной приоритета, имеют более высокий приоритет, чем сущности, которые захватываются с более низкой величиной. Приоритетные выражения, оцененные как отрицательные величины, рассматриваются как нулевой приоритет. Если несколько сущностей с равными приоритетами пытаются захватить один и тот же ресурс, то его получает сущность с наибольшим временем ожидания.

2) Возможно определить набор состояний (State set) для ресурса и назначить состояние ресурса в определённых ситуациях, используя область состояния ресурса (Resource State Field). Затем можно собрать статистику: сколько времени приходится на каждое состояние ресурса. В табл.1.24 приведены параметры модуля Seize.

Таблица 1.24

Параметры модуля Seize

Параметры	Описание
Name	Уникальное имя модуля, которое будет отражено в блок-схеме
Allocation	Определяет категорию, по которой будет распределена стоимость использования ресурса для сущности, проходящей через модуль Seize

Продолжение табл.1.24

Priority	Приоритет сущности, ожидающей в этом модуле ресурс. Определяется в случае, когда 1 или несколько сущностей из других модулей ожидают тот же ресурс (1 - высокий, 2 - средний, 3 - низкий, др.)
Type	Тип ресурса, который должен быть захвачен. Определяет конкретный ресурс или выбирает набор ресурсов. Имя ресурса также может быть определено атрибутом или выражением (Resource, Set, Attribute, Expression)
Resource name	Имя ресурса, который должен быть захвачен
Selection rule	Метод выбора среди доступных ресурсов в наборе

Модуль Delay задерживает сущность на определённое количество времени. По прибытии сущности в модуль выражение времени задержки оценивается и сущность остаётся в модуле на результирующее время. Затем время выделяется и, в зависимости от Allocation, либо добавляется к значению сущности, либо не добавляется, либо передаётся, либо ждет другое время. Также стоимости складываются, вычисляются и выделяются.

Параметры модуля Delay приведены в табл.1. 25.

Таблица 1.25

Параметры модуля Delay

Параметры	Описание
Name	Уникальное имя модуля, которое будет отражено в блок-схеме

Продолжение табл.1.25

Allocation	Тип категории, в которой сущности могут быть подвергнуты задержке времени и добавлению стоимости
Delay Time	Определяет значение задержки времени для сущности
Units	Указывает единицу измерения задержки времени

Модуль Release используется для того, чтобы освобождать ресурсы, которые прежде были захвачены сущностью.

Этот модуль может быть использован для того, чтобы освобождать индивидуальные ресурсы или ресурсы в пределах набора. Для каждого ресурса, который нужно освободить, определяется имя и количество. Когда сущность поступает в модуль, она теряет управление определённым ресурсом. Любые сущности, ожидающие в очередях этот ресурс, получают его немедленно.

Замечания:

1) Если есть сущность, ожидающая в очередях для захвата определённого ресурса, то, когда ресурс освобождается, он автоматически распределяется в ждущую сущность. Эта ждущая сущность будет обработана, как только сущность, которая освободила ресурс, переместится.

2) Системная переменная NR (имя ресурса) возвращает номер последнего занятого ресурса. Когда сущность поступает в модуль Release, NR уменьшается на количество освобождённых ресурсов, если ресурс не будет немедленно захвачен другой сущностью.

3) Если освобождается большее количество ресурсов, чем было ранее захвачено, то происходит ошибка.

4) Освобождение множества ресурсов выполняется в порядке их появления в модуле Release.

Параметры модуля Release приведены в табл.1.26.

Таблица 1.26

Параметры модуля Release

Параметры	Описание
Name	Уникальное имя модуля, которое будет отражено в блок-схеме
Resources	Имя освобождаемых модулем ресурсов

Модуль ReadWrite используется для чтения данных из входного файла или с клавиатуры и задания данных в список переменных или атрибутов (или других выражений). Этот модуль также используется, чтобы записать данные на выходное устройство, например на экран или в файл.

Когда объект приходит в модуль ReadWrite, то файл изучается для того, чтобы увидеть, открыт он или нет. Если нет, файл автоматически открывается. Величины атрибутов, переменные или выражения будут прочитаны или записаны в зависимости от того, какой формат определен. В табл.1.27 приведены параметры модуля ReadWrite.

Таблица 1.27

Параметры модуля ReadWrite

Параметры	Описание
Name	Уникальный модульный идентификатор. Это имя отображается в модульной форме
Type	Метод использования (чтение или запись). Данные могут быть записаны в файл или на экран. Данные могут быть считаны из файла или с клавиатуры
Arena File Format	Имя файла, чтобы идентифицировать файл в пределах модуля File

Продолжение табл.1.27

Overriding File Format	Формат для записи или чтения данных. Этот формат аннулирует любой формат, определенный в структурной области модуля File. FORTRAN или C может использоваться, чтобы описать тип и позицию каждой области
Variable Type	Тип информации, что будет прочитана или записана
Attribute Name	Определяет символьное имя атрибута для записи или чтения
Variable Name	Определяет символьное имя переменной для записи или чтения
Other	Определяет выражение для чтения или записи других типов информации

1.5.2. Модули данных

Модуль Advanced Set определяет наборы (очередей, хранилищ или другие наборы) с соответствующими его составляющими. Набор определяет группу схожих элементов, к которым можно обращаться через имя и индекс. К элементам, входящим в набор, можно обращаться как к членам этого набора.

Наборы очередей могут быть определены при помощи модуля Seize.

Параметры модуля Advanced Set приведены в табл.1.28.

Таблица 1.28

Параметры модуля Advanced Set

Параметры	Описание
Name	Уникальный идентификатор
Set Type	Тип набора. Может быть Queue, Store, Other (другой)
Members	Задаются конкретные составляющие (очереди, хранилища), входящие в набор

Модуль Expression позволяет определять выражения и задавать им значения. К выражению обращаются при помощи имени. Выражения могут быть заданы как одномерный или двумерный массив. В табл.1.29 приведены параметры модуля Expression.

Таблица 1.29

Параметры модуля Expression

Параметры	Описание
Name	Уникальное имя выражения
Row	Максимальное количество строк в определяемом выражении
Column	Максимальное количество столбцов в определяемом выражении. Данное свойство задается, только когда задано свойство Row
Expression Value	Значение, которое соответствует выражению

Этот модуль необходим для того, чтобы задавать какие-то часто использующиеся выражения, чтобы разгрузить модель, например в модулях Decide, Hold, Pickup.

Модуль Statistic используется для того, чтобы определить дополнительную статистику, которая должна собираться в течение времени моделирования, а также чтобы определить файлы выходных данных. Параметры модуля Statistic приведены в табл.1.30.

Таблица 1.30

Параметры модуля Statistic

Параметры	Описание
Name	Уникальное имя модуля
Type	Тип статистики. Тип может быть time-persistent, tallies (observational data), count-based, outputs, and frequency-based

В зависимости от выбранного типа статистики появляются дополнительные поля.

1) Если выбран тип Tally: Tally Name – определяется символьное имя для типа статистики Tally, Tally Output File – имя выходного файла.

2) Если выбран тип Counter: Counter Name – определяется символьное имя для типа статистики Counter; Limit определяет лимит счетчика; Counter Output File – имя выходного файла.

Модуль Storage определяет имя хранилища. Хранилище автоматически создается любым модулем, который на него ссылается.

Модуль File должен быть включен всякий раз, когда обращаются к внешнему файлу, используя ReadWrite модуль. Этот модуль выделяет системный файл, называет и определяет метод доступа, форматирование и эксплуатационные характеристики файла. Параметры модуля File приведены в табл.1.31.

Таблица 1.31

Параметры модуля File

Параметры	Описание
Operating System File Name	Операционное системное имя, путь к файлу, откуда читаем или записываем. Символьная строка
Structure	Тип файловой структуры. Неформатированный, свободный формат, WorksSheet, специфические C- или FORTRAN-форматы
End of File Action	Тип действия, которое произойдет, когда будет достигнут конец файла. Ошибка, выход, на начало, игнорировать
Comment Character	Символ, указывающий отображение комментирующей записи. Одиночный символ

Модуль StateSet используется для того, чтобы определить состояние ресурса или набора ресурсов. Состояния могут быть связаны с автосостоянием или могут быть заданы новые состояния для ресурса. Модуль Resourse в базовой панели Process ссылается на StateSet, который данный ресурс будет использовать. В табл.1.32 приведены параметры модуля StateSet.

Таблица 1.32

Параметры модуля StateSet

Параметры	Описание
StateSet Name	Название набора состояний, которые могут быть назначены ресурсу в течение модельного времени
State Name	Имя пользователя определившего состояние

Продолжение табл.1. 32

Auto State or Failure	Используется, чтобы связать State Name с автосостоянием или с заданным пользователем, именем отказа

Модуль Failure разработан для использования с ресурсами, а именно для имитации отказов ресурса. Может использоваться для ресурсов с однократной способностью или для ресурсов многократной способности, когда индивидуальные единицы ресурса заняты в одно и то же время.

Таблица 1.33

Параметры модуля Failure

Параметры	Описание
Name	Имя отказа
Count	Определяет число ресурсов, реализуемых для отказов
Time	Определяет время для отказов
Up Time	Определяет время между отказами (число)
Up Time Units	Задаем формат времени (секунда, минута, час, день)
Down Time	Определяем продолжительность отказа (число)
Down Time Units	Задаем формат времени (секунда, минута, час, день)

Параметры модуля Failure приведены в табл.1.33.

1.6. Advanced Transfer Panel (панель перемещения)

1.6.1. Схемные модули

Модуль Station определяет станцию или набор станций для физической или логической обработки, некая логическая («отправная») точка в модели. В табл. 1.34 приведены параметры модуля Station.

Таблица 1.34

Параметры модуля Station

Параметры	Описание
Name	Уникальное имя модуля, которое будет отражено в блок-схеме
Station Type	Тип станции
Station Name	Имя станции
Set Name	Уникальное имя набора станций
Save Attribute	Название атрибута, куда будут сохраняться значения атрибутов сущностей
Station Set Members	Перечисляется набор станций

Модуль Route позволяет принять указанную сущность на заданную станцию, при этом позволяет имитировать время, которое будет затрачено сущностью на дистанцию к заданной станции.

Параметры модуля Route приведены в табл.1.35.

Таблица 1.35

Параметры модуля Route

Параметры	Описание
Name	Уникальное имя модуля, которое будет отражено в блок-схеме
Route Time	Время прохода через этот модуль
Units	Единицы измерения времени задержки (день, час, минута, секунда)
Destination Type	Тип станции назначения, на которую должна прибыть сущность (Station, Sequential, Attribute, Expression)

Модуль PickStation позволяет сущностям выбирать определенную станцию из множества существующих (маршрутизатор). Параметры модуля PickStation приведены в табл.1.36.

Таблица 1.36

Параметры модуля PickStation

Параметры	Описание
Name	Уникальное имя блока
Test Condition	Определяется тип выбора станции (минимум или максимум по полям): Number In Queue (количество в очереди); Number En Route to Station (количество маршрутизированных станций); Number of Resources Busy (количество занятых ресурсов) и Expression (выражение)
Route Time	Время в пути (до станции)
Units	Единицы измерения времени пути (день, час, минута, секунда)

Продолжение табл.1.36

Save Attribute	Имя атрибута, который хранит имя станции
Transfer Type	Определяет, каким образом сущности будут транспортироваться до следующей станции (Route, Transport, Convey or Connect)

Модуль Enter определяет станцию (или станции), соответствующую физическим или логическим позициям, где происходит обработка. Если модуль Enter определяет конкретную станцию, он эффективно определяет многочисленные обработки позиций.

Станция (или каждая станция в пределах решаемого комплекта) соотносится к области деятельности, которая используется, чтобы сообщить о времени и издержках, повышенных сущностями, на этих станциях. Эта сущность имени AreaTs также называется станцией.

Сущность может переместиться из предыдущего модуля в модуль Enter, причем двумя способами: отправление на станцию, связанную с модулем дистанционно или через реальное графическое соединение.

Когда сущность прибывает в модуль Enter, «разгружая», может произойти задержка и любое действие с передачей. В табл.1.37 приведены параметры модуля Enter.

Таблица 1.37

Параметры модуля Enter

Параметры	Описание
Name	Уникальное имя модуля, которое будет отражено в блок-схеме

Продолжение табл.1.37

Station Type	Определяет индивидуальную станцию или комплект станций, чтобы определить точку входа в этот модуль. Если выбран комплект (set), - это указывает, что этот модуль входит в подмодель станции
Station Name	Имя станции активно в том случае, когда выбран тип Type Station
Parent Activity Area	Имя места отправления
Delay	Время задержки сущности по прибытии на данную станцию
Allocation	Тип категории, к которому будет добавляться время сущности и цена
Transfer In	Если выбран ресурс (транспортер или конвейер), чтобы доставить сущность к станции, используется для «отпускания», «освобождения» или «выхода»

Модуль Leave используется для передачи сущности к станции или другому модулю.

Когда сущность прибывает в модуль Leave, она ожидает прибытия транспорта, когда прибывает транспорт, тратится время на загрузку, и в конечномитоге сущность отправляется в пункт модуля назначения.

В табл.1.38 приведены параметры модуля Leave.

Таблица 1.38

Параметры модуля Leave

Параметры	Описание
Name	Уникальное имя модуля, которое будет отражено в блок- схеме
Allocation	Тип категории, к которому будет добавляться время сущности
Delay	Время задержки сущности по прибытии на данную станцию
Unit	Величина задержки: день, час, минута, секунда
Transfer Out	Тип, содержащий запрос на транспорт

Далее будут подробно рассмотрены модули транспортера.

Модуль Request вызывает (запрашивает) транспортер по прибытии в него сущности. Когда сущность достигает модуля Request, она размещается на транспортере, когда он доступен. Сущ-ность остается в модуле Request, пока транспортер не достиг станции. Только тогда сущность перемещается из модуля Request для дальнейшего движения по модели. Параметры модуля Request приведены в табл.1.39.

Таблица 1.39

Параметры модуля Request

Параметры	Описание
Name	Уникальное имя модуля
Transporter Name	Название (имя) транспортера

Продолжение табл.1.39

Velocity	Скорость, с которой транспортер перемещает (единица длины в единицу времени). Единица времени определена в поле Units
Units	Определяет единицы времени для Velocity (т. е. в минуту, в час и т. д.)
Queue Type	Определяет тип очереди при загруженности транспортера
Queue Name	Эта область видима, только если тип очереди - очередь, и это определяет имя символа очереди

Модуль Activate активирует или увеличивает вместимость предварительно приостановленного транспортера или транспортера, который был первоначально бездействующим (как определено в модуле Transporter).

Таблица 1.40

Параметры модуля Activate

Параметры	Описание
Name	Уникальное имя модуля, которое будет отражено в блок-схеме
Transporter Name	Имя транспортера, с которым работает модуль
Unit Number	Определяет, насколько увеличится вместимость

В табл.1.40 приведены параметры модуля Activate.

Модуль Allocate аналогичен модулю Request. Различие только в том, что модуль Allocate не позволяет задавать скорость и единицы измерения скорости транспортера.

Модуль Move продвигает транспортер от одной станции к другой, которая является пунктом назначения. Контролируемая сущность ожидает в текущем модуле, пока транспортер прибудет в назначенный пункт. После этого сущность может перемещаться в другой модуль модели.

Время задержки перемещения транспортера из одного пункта (модуля Station) в другой основано на скорости транспортера, которая определяется в модуле Trans-porter, и расстоянии между пунктами, определенном в модуле Distance.

Сущность не может быть перемещена транспортером, если он не вызван с помощью модулей Request или Allocate. Сущность будет оставаться в модуле Move, пока транспортер не достигнет своего пункта назначения. Если определена скорость движения, это изменение временно и утилизируется только для определенного транспортера, который перемещается. Параметры модуля Move приведены в табл.1.41

Таблица 1.41

Параметры модуля Move

Параметры	Описание
Name	Уникальное имя модуля, которое будет отражено в блок-схеме
Transporter Name	Имя транспортера для перемещения
Unit Number	Определяет транспортер из множества транспортеров
Destination Type	Тип места назначения транспортера
Station Name	Имя места назначения (станции), в которое транспортер переместится

Продолжение табл.1.41

Velocity	Скорость, с которой транспортер переместится в пункт назначения, в единицах времени. Единицы времени определяются в поле Units
Units	Определяет единицы времени (секунды, минуты, часы, дни)

Модуль Transport по прибытии в него сущности запускает транспортер и перемещает его от одной станции к другой. Время задержки на перемещение и передачу сущности от одной станции к другой основывается на скорости транспортера и расстоянии между станциями.

Когда сущность входит в модуль Transport, то атрибут станции (Entity.Station) подставляется в станцию назначения, затем сущность передается в станцию назначения. Если станция назначения входит как Sequential, то следующая станция определяется посредством «Запроса сущности» и Jobstep с множеством (специально определенных атрибутов Entity.Sequence and Entity.Jobstep, respectively).

Модуль Transport является эквивалентом модуля Move, с той разницей, что Transport передает сущности дистанционно.

Таблица 1.42

Параметры модуля Transport

Параметры	Описание
Name	Уникальное имя модуля, которое будет отражено в блок-схеме
Transporter Name	Определяет имя транспортера для передачи
Unit Number	Определяет, какой из транспортеров из множества транспортеров подлежит перемещению

Продолжение табл.1.42

Destination Type	Определяет тип места назначения сущности
Station Name	Определяет имя места назначения (станции), в которое сущность будет перемещаться
Velocity	Скорость, с которой транспортер перемещается к станции назначения
Units	Это поле определяет единицы измерения времени для скорости

В табл.1.42 приведены параметры модуля Transport.

Модуль Free освобождает транспортер для дальнейшего его использования. Параметры модуля Free приведены в табл. 1.43.

Таблица 1.43

Параметры модуля Free

Параметры	Описание
Name	Уникальное имя модуля, которое будет отражено в блок-схеме
Transporter Name	Имя транспортера, который освободится

Модуль Halt изменяет состояние (статус) транспортера на неактивное. Если транспортер занят, в то время как сущность вошла в мо-дуль Halt, то его статус определяется как занят и неактивен до тех пор, пока сущность, которая управляет транспортером, не освободится. Если во время вхождения сущности в модуль Halt транспортер является

свободным, то статус транспортера изменяется на неактивный немедленно.

Никакая сущность не может получить управление над остановленным транспортером, пока он снова не будет активизирован. Параметры модуля Halt приведены в табл.1.44.

Таблица 1.44

Параметры модуля Halt

Параметры	Описание
Name	Уникальное имя модуля, которое будет отражено в блок-схеме
Transporter Name	Имя транспортера, который требуется остановить
Unit Number	Определяет, какие из модулей транспортера из набора транспортера следует останавливать

Далее будут подробно рассмотрены модули конвейера.

Модуль Access вызывает конвейер, распределяет ячейки конвейера для перемещения сущности от станции к станции. Получив контроль над ячейками конвейера, сущность может переместиться к другойстанции конвейера. Этот модуль является эквивалентом модуля Request.

Таблица 1.45

Параметры модуля Access

Параметры	Описание
Name	Уникальное имя модуля, которое будет отражено в блок-схеме
СеП	Количество ячеек, необходимых для перемещения конвейера

Продолжение табл.1. 45

Conveyor Name	Имя конвейера-исполнителя
Queue Name	Имя очереди, в которую поступают сущности конвейера, если конвейер занят

Параметры модуля Access приведены в таблице 45.

Модуль Convey перемещает сущности по конвейеру от одной к другой. Время задержки сущности в пути определяется полем Velosity модуля Conveyor и расстоянием между станциями, определенным в модуле Segment. Этот модуль является эквивалентом модуля Transport. Параметры модуля Convey приведены в табл.1.46.

Таблица 1.46

Параметры модуля Convey

Параметры	Описание
Name	Уникальное имя модуля, которое будет отражено в блок-схеме
СеП	Количество ячеек, необходимых для перемещения конвейера
Conveyor Name	Имя конвейера, который будет использоваться
Destination Type	Определяет метод для определения пункта назначения сущности: Station Name - имя станции; Attribute Name - имя атрибута, который хранит имя станции; Sequential - следующая станция, которая определяется атрибутами сущности Entity.Sequence и Entity.JobStep, и Expression - выражение, которое определяет станцию

Модуль Start изменяет статус конвейера от бездействующего до активного, т. е. активизирует (вызывает) конвейер. Конвейер может быть остановлен модулем Stop или окончанием создания сущности в начале моделирования. Скорость конвейера может изменяться постоянно после начала работы конвейера. Является эквивалентом модуля Move. В табл.1.47 приведены параметры модуля Start.

Таблица 1.47

Параметры модуля Start

Параметры	Описание
Name	Уникальное имя модуля, которое будет отражено в блок-схеме
Conveyor Name	Имя конвейера, который требуется активировать
Velocity	Скорость, с которой конвейер переместится в пункт назначения, в единицах времени. Единицы времени определяются в поле Units
Units	Определяет единицы времени (секунды, минуты, часы, дни)

Модуль Stop устанавливает действующий статус конвейера в неактивный. Конвейер может быть активирован для любого модуля Start или по причине активации в начале моделирования. Когда сущность входит в модуль Stop, конвейер мгновенно останавливается, принимая во внимание тип конвейера или номер сущности, вошедшей в конвейер. Является эквивалентом модуля Halt для транспортера.

Параметры модуля Stop приведены в табл.1.48.

Таблица 1.48

Параметры модуля Stop

Параметры	Описание
Name	Уникальное имя модуля, которое будет отражено в блок-схеме
Conveyor Name	Имя конвейера для остановки

Модуль Exit выпускает сущности из определенного конвейера и освобождает его для дальнейшей перевозки сущностей. Является эквивалентом модуля Free транспортера.

Параметры модуля Exit приведены в табл.1.49.

Таблица 1.49

Параметры модуля Exit

Параметры	Описание
Name	Уникальное имя модуля, которое будет отражено в блок-схеме
Conveyor Name	Имя конвейера, который освободится
# of Cells	Число последовательных сущностей для выпуска

1.6.2. Модули данных

Модуль Transporter предназначен для определения транспортера в модели. Чаще всего модуль связан со схемным модулем Request, который вызывает транспортер, и модулем Move, который передвигает транспортер по схеме.

В табл.1.50 приведены параметры модуля Transporter.

Таблица 1.50

Параметры модуля Transporter

Параметры	Описание
Name	Уникальное имя транспортера
Capacity	Количество транспортеров в наборе
Distance set	Определяет имя дистанции (пути), по которому будет двигаться транспортер
Velocity	Определяет начальную скорость транспортера
Units	Единицы измерения скорости
Initial Position	Определяет начальную станцию, с которой транспортер начнет свое движение

Модуль Distance предназначен для определения пути, по которому будет двигаться транспортер. Параметры модуля Distance приведены в табл.1.51.

Таблица 1.51

Параметры модуля Distance

Параметры	Описание
Name	Уникальное имя дистанции
Beginning Station	Начальная станция дистанции
Ending Station	Конечная станция дистанции
Distance	Длина дистанции

Модуль Conveyor позволяет перемещать сущности между станциями, является аналогом модуля Transporter.

В табл.1.52 приведены параметры модуля Conveyor.

Таблица 1.52

Параметры модуля Conveyor

Параметры	Описание
Name	Название конвейера
Segment Name	Имя сегмента, по которому будет двигаться конвейер
Type	Существует 2 типа конвейера: накапливающий и не накапливающий
Velocity	Определяет начальную скорость транспортера
Units	Единицы измерения скорости

Модуль Segment определяет путь, по которому будет двигаться конвейер. Параметры модуля Segment приведены в табл.1.53.

Таблица 1.53

Параметры модуля Segment

Параметры	Описание
Name	Имя сегмента
Beginning Station	Начальная станция
Next Station	Следующая станция в сегменте (может задаваться набором)
Length	Расстояние до предыдущей станции

1.7. Панель отчётов

С помощью панели отчетов можно просмотреть результаты имитации. На панели отчетов представлены несколько видов отчетов: Отчет «Краткий обзор категорий» и отчеты по четырем категориям, такие, как Сущности, Процессы, Очереди и Ресурсы.

1. Отчет Category Overview категорий (Краткий обзор категорий) отражает итоговую информацию о сущностях, процессах, очередях и ресурсах. Также показывает информацию о заданных пользователем переменных и информацию, собранную модулем Record.

2. Отчет о сущностях разделен на несколько частей:

2.1. Cycle Time: в этой части отчета показано среднее, максимальное и минимальное время существования сущности. Время существования сущности считается с момента её прибытия в систему и до того момента, когда сущность попадает в модуль Dispose. Ниже представляется гистограмма среднего времени цикла для каждого типа сущности.

2.2. NVA Cost: в этой части показано среднее, максимальное и минимальное значение недобавочной стоимости сущностей по каждому типу. Недобавочная стоимость рассчитывается на основании значения NVA Time.

2.3. Total Cost: в этой части показано среднее, максимальное и минимальное значение общей стоимости сущностей по каждому типу. Общая стоимость вычисляется путем сложения стоимости ожидания, добавочной стоимости и недобавочной стоимости для каждой сущности.

2.4. VA Cost: в этой части показано среднее, максимальное и минимальное значение добавочной стоимости сущностей по каждому типу. Добавочная стоимость рассчитывается на основании VA Time.

2.5. Wait Cost: в этой части показано среднее, максимальное и минимальное значение стоимости ожидания сущностей по каждому типу. Стоимость ожидания подсчитывается, исходя из времени ожидания, стоимости ресурса и стоимости нахождения сущности в системе.

2.6. Wait Time: в этой части показано среднее, максимальное и минимальное значение времени ожидания сущностей по каждому типу. Время ожидания - это период времени с момента поступления сущности в очередь (либо в модуле Process ожидает ресурс, либо в модуле Batch ожидает группиров-

ки) и до момента выхода из нее (начнет обрабатываться либо будет сгруппирована).

2.7. WIP (Work In Process): в этой части показано среднее, максимальное и минимальное значение времени ожидания сущностей в процессах.

3. Отчет о процессах разделен на такие же части, как и отчет по сущностям, только с уклоном на процессы.

4. Отчет о ресурсах содержит информацию о загруженности и простое ресурсов.

Отчет по очередям содержит информацию о среднем, минимальном и максимальном времени нахождения сущности в очереди и максимальных, средних и минимальных очередях.

1.8. Панель навигации

С помощью панели навигации можно быстро передвигаться по различным уровням модели, быстро менять виды. Можно задать быстрые клавиши для изменения вида. Виды подмоделей создаются автоматически, но также возможно добавить новые виды с помощью команды Add View. Можно передвигаться не только по различным уровням модели, но также быстро получать нужный масштаб какой-либо части модели.

1.9. Построитель выражений

ПП Arena позволяет строить сложные выражения. Это достигается с помощью Expression Builder. Построитель выражений имеет внешний вид, показанный на рис. 1.4.

Построитель выражений имеет 3 секции:

1. Окно типов выражений. Рассмотрим более подробно *окно типов выражений,* которое состоит из четырех разделов:

1.1. Random Distributions (Вероятностные распределения). В ПП Arena 7.0 заложены 13 типов стандартных распределений:

- normal (нормальное): Mean, StdDev;
- exponential (экспоненциальное): Mean;
- uniform (равномерное): Min, Max;
- poisson (пуассоновское): Mean;
- gamma (гамма): Beta, Alpha;
- beta (бета): Beta, Alpha;
- triangular (тереугольное): Min, Mode, Max;
- continious (непрерывное): $CumP_1$, Val_1, ... $CumP_n$, Val_n;
- discrete (дискретное): $CumP_1$, Val_1, …. $CumP_n$, Val_n;
- erlang (распределение Эрланга): ExpoMean, k;
- johnson (распределение Джонсона): Gamma, Delta, Lambda, Xi;
- lognormal (логнормальное): LogMean, LogStd;
- weibull (распределение Вейбулла): Beta, Alpha.

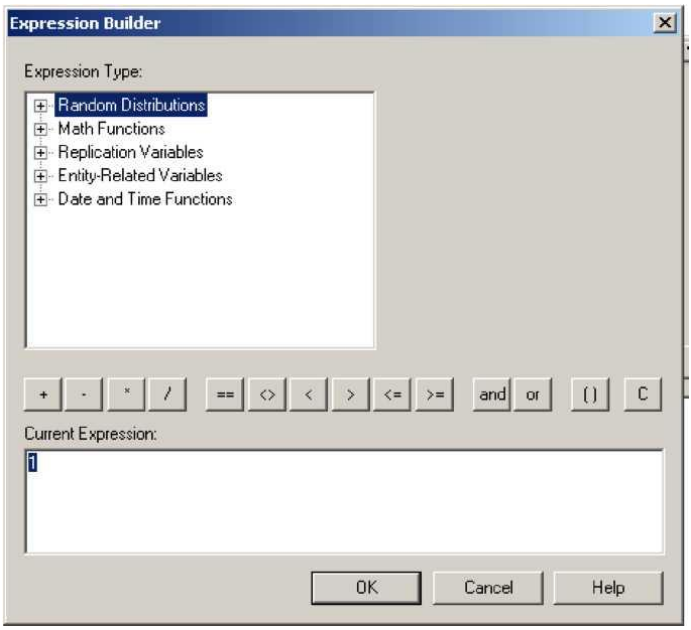


Рис. 1.4. Внешний вид построителя выражений

1.2. Math Functions (Математические функции), к которым относятся 11 алгебраических операторов:

- абсолютное значение;
- округление до ближайшего целого;
- целая часть от нецелочисленного значения;
- минимальное значение;
- максимальное значение;
- натуральный логарифм;
- корень квадратный и т. д. и 9 геометрических функций:
- синус;
- косинус;
- тангенс;
- арксинус и т. д.;
- Replication Variables (переменные, связанные с репликациями модели);
- Maximum Replications (максимальное количество повторений);

Current Replication Number (текущее количество повторений).

1.3. Math Functions (Математические функции), к которым относятся 11 алгебраических операторов:

- абсолютное значение;
- округление до ближайшего целого;
- целая часть от нецелочисленного значения;
- минимальное значение;
- максимальное значение;
- натуральный логарифм;
- корень квадратный и т. д. и 9 геометрических функций:
- синус;
- косинус;
- тангенс;

- арксинус и т. д.;
- Replication Variables (переменные, связанные с репликациями модели);
- Maximum Replications (максимальное количество повторений);
- Current Replication Number (текущее количество повторений).

1.4. Entity-Related Variables (переменные, связанные с сущностью):

- Attributes (Атрибуты). К наиболее интересным атрибутам следует отнести: Entity.Type (тип сущности), Entity.SerialNumber (серийный номер сущности), Entity .Picture (анимационная картинка сущности), Entity.CreateTime (Время создания сущности), User-Defined Attribute Value (атрибуты, заданные пользователем);
- Group Member Variables (групповые переменные).

1.5. Date and Time Functions (временные функции). Наиболее интересный и часто используемый оператор из этой группы - это TNOW (Current Simulation Time - текущее время моделирования).

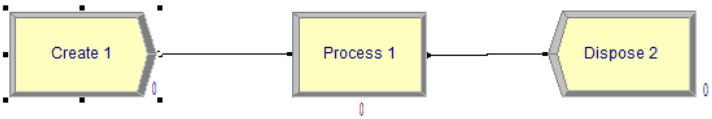
2. Панель операторов, используемых в выражениях (сложение, вычитание, и т. д.; элементы сравнения, логические операторы и т. д.).

3. Окно записи выражения.

2.ПРАКТИЧЕСКОЕ ПРИМЕНЕНИЕ ПАКЕТА ARENA

2.1. Пример простейшего моделирования

Ниже представлен пример простейшего моделирования в пакете Arena. Была построена простейшая конструкция на которой рассматривались основные свойства модели в пакете Arena.



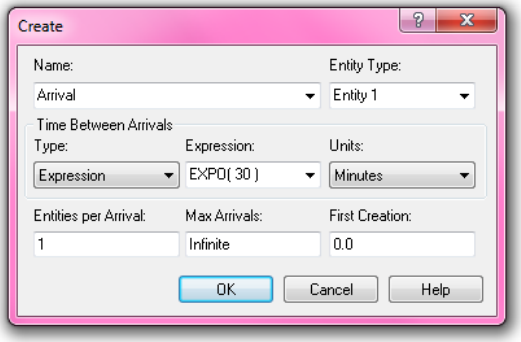


Рис. 2.1. Настройка блока Create

После создания простейшей конструкции были изменены параметры блоков в соответствии с указанным в лабораторной работе (рис. 2.1 – 2.3).

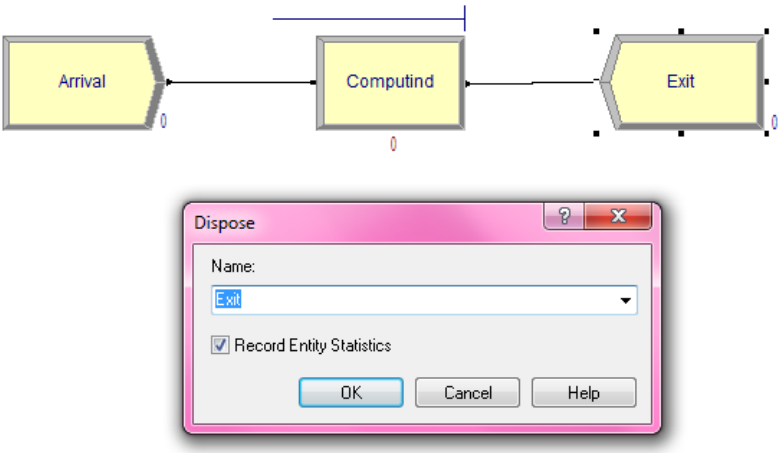


Рис. 2.2. Настройка блока Dispose

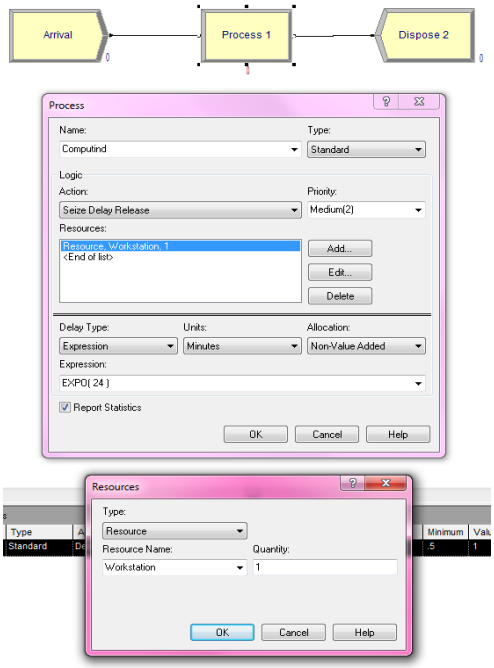


Рис. 2.3. Настройка блока Process

После этого были изменены параметры блока создания сущностей (рис. 2.4).

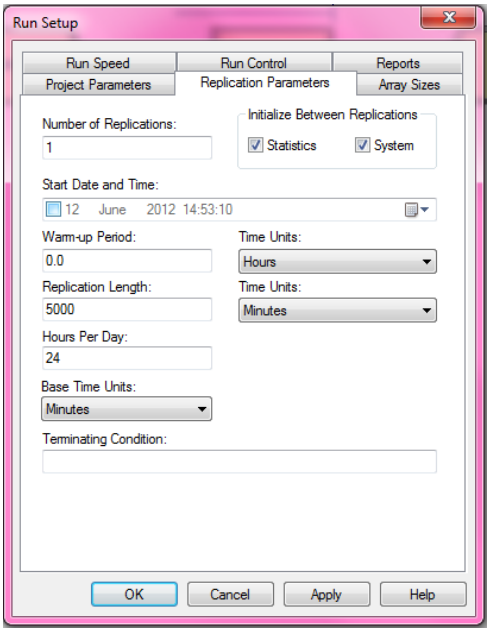


Рис. 2.4. Настройка блока Process.

2.2 Input Analyzer

Для работы с инструментом пакета Arena, называемым Input Analyzer необходимо создать таблицу случайных значений используя MS Excel, либо любой другой редактор таблиц. После этого сохранить результаты в таблице в формате “.txt (MS DOS)”.

После этого файл можно загрузить в Input Analyzer.

Можно провести подбор возможного распределения значений основываясь на графическом показателе. На рис. 2.5 представлено гамма распределение.

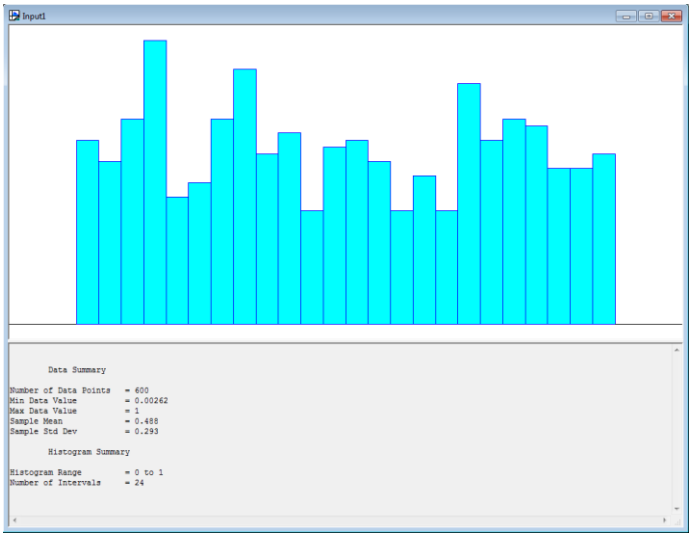


Рис. 2.5. Гистограмма используемых данных.

Автоматический подбор выбрал Бета распределение как наиболее подходящее.

Основываясь на значениях критериев хи-квадрат и Колмогорова-Смирнова, полученных при сравнении, Бета распределение гораздо более подходит для полученного набора значений.

С помощью Input Analizer можно провести анализ случайных чисел, сгенерированных с помощью программы MS Excel. Полученные значения были приведены к Бета и Гамма распределениям (рис. 2.6 – 2.7). Основываясь на результатах критериев, используемых в ходе работы Бета распределение оказалось наиболее близким к полученным значениям.

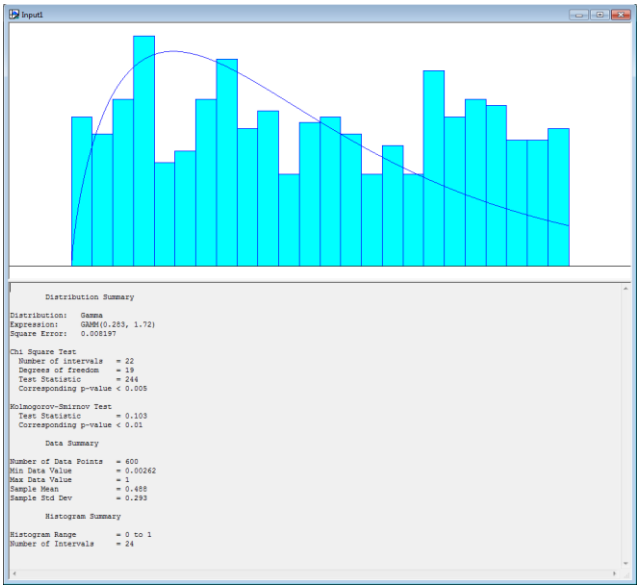


Рис. 2.6. Гамма распределение

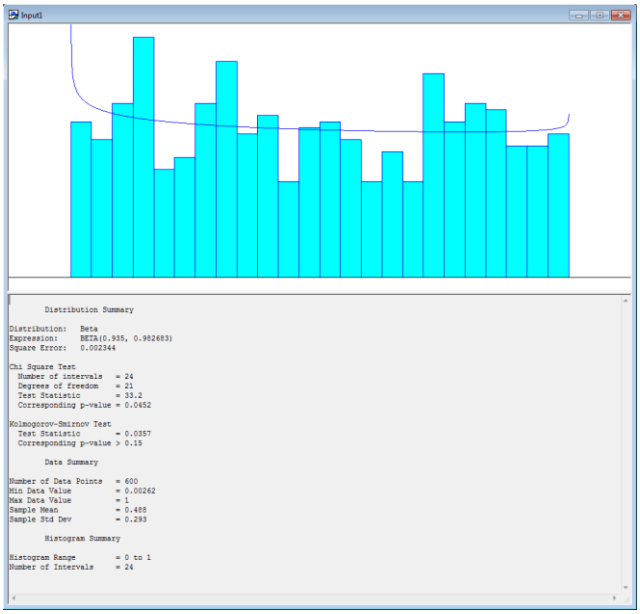


Рис. 2.7. Бета распределение

2.3. Пример более сложного моделирования

Ниже представлен пример более сложного моделирования в пакете Arena (рис. 2.8 – 2.11).

Блоки поступают в цех по расписанию согласно работы предприятия. За один подход завозится два блока питания. После чего они проходят осмотр и проверку работоспособности. Если проверка прошла успешно, то они перенаправляются в один из трех возможных цехов, для последующей эксплуатации. Если проверка не пройдена, то блок отправляется на вторичный осмотр, определяющий возможность переработки данного блока, после чего блок питания либо отправляется в переработку, либо на свалку.

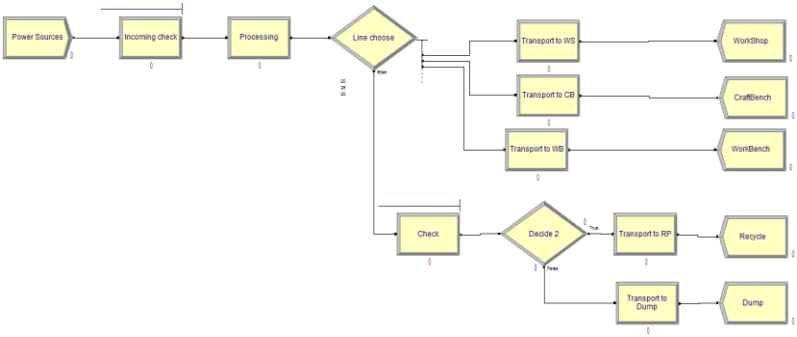


Рис. 2.8. Модель цеха проверки блоков питания

Resource - Basic Process

	Name	Type	Capacity	Busy / Hour	Idle / Hour	Per Use	StateSet Name	Failures	Report Statistics
1	Workman	Fixed Capacity	5	15	5	10		0 rows	☑
2	Merchandiser	Fixed Capacity	4	13	5	10		0 rows	☑
3	Checker	Fixed Capacity	30	15	5	10		0 rows	☑

Double-click here to add a new row.

Рис. 2.9. Сводная таблица Resource

Queue - Basic Process

	Name	Type	Shared	Report Statistics
1	Incoming check.Queue	First In First Out	☐	☑
2	Check.Queue	First In First Out	☐	☑

Double-click here to add a new row.

Рис. 2.10. Сводная таблица Queue

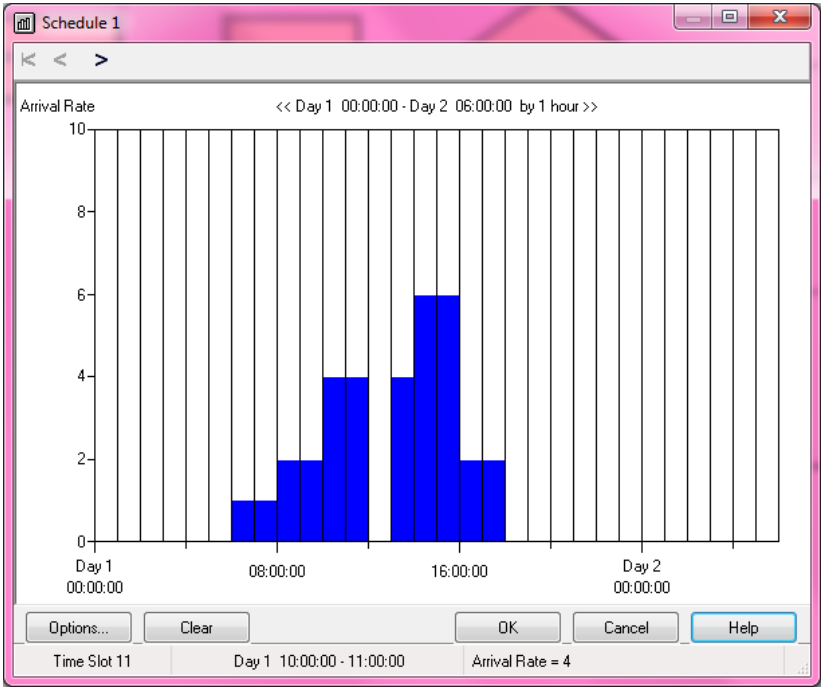


Рис.2.11. Сложное расписание, используемое для блока Create

В данном примере выполнения работы была составлена и смоделирована сложная схема описывающая цех проверки качества блоков питания. В результатах имитации была обнаружена аномалия в работе блока, вторичного осмотра. В блоке с очередью, не было ни одного простоя. Это объясняется тем, что осмотр предыдущего блока питания заканчивался до поступления нового, как следствие отсутствие простоев.

Ниже представлен пример добавления в модель более сложных конструкций и элементов (рис. 2.12).

Необходимо добавить к уже использованной ранее схеме блоки Record и Assign. Назначение блоков Assign было выбрано в качестве подсчета влияния цеха на капитал предприятия. Так как сам цех ничего не создает и ничего не тратит, то изменения капитала могут быть лишь от отправления неисправных блоков на переработку. Так же существует потенциальная прибыль от обмена с одним из трех цехов, в котором незапланированный выход из строя одного из блоков питания может привести к большим потерям прибыли в будущем.

Блоки Record были установлены для подведения временной статики обмена с одним из трех цехов, а так же на дополнительный численный контроль утилизируемых блоков.

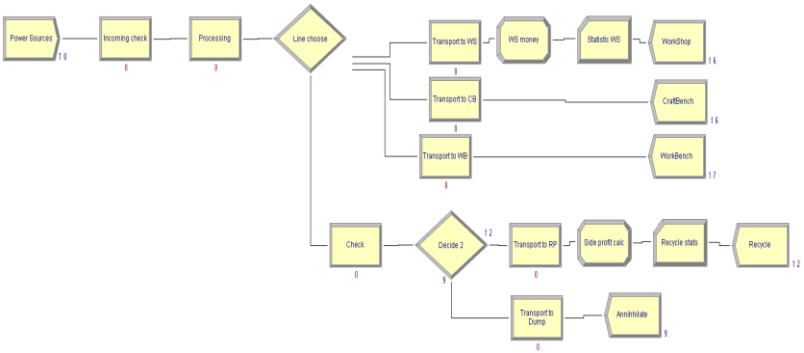


Рис.2.12. Общий вид усложненной модели

По сравнению с результатами прошлого примера схема стала более точна, так же появились блоки для подсчета промежуточных и потенциальных приболей. Используемые в данной модели блоки Record используемые для контроля не играют значительной роли на данном этапе, но при усложнении схемы, и приведению ее к наиболее соответствующей реальности модели, эти блоки помогут отслеживать ошибки персонала и возможную потерю блоков питания.

Усложнение существующей схемы заметно затруднило поддержание ее в работоспособном состоянии, но при верной настройке и использовании блоков была получена работоспособная схема, более точная, чем в предыдущей работе. Переменные, используемые в блоках Assign для подсчета прибылей – оптимальный вариант для данной задачи.

2.4. Process Analyzer

Ниже представлен пример работы с инструментом Process Analizer.

Для представления модели графически в пакете Arena 9.0 используется инструмент Process Analyzer.

В качестве примера использования анализатора процессов, будет рассмотрена модель Movie Theater Analysis из готовых примеров пакета Arena. Эта модель используется для изучения максимально эффективного размещения служащих в комплексе кинотеатров. Основная цель компании состоит в максимизации чистой прибыли, при ограничении штатного расписания.

Запуск анализатора процессов

Этот инструмент можно запустить из меню Пуск → RockWell Software → Arena → Process Analyzer.

Второй способ запуска – из окна запущенной программы, в главном выпадающем меню надо выбрать Tools → Process Analyzer.

Создание нового проекта

Создайте новый проект выбрав File → New пункт меню.

Добавление нового сценария

Что бы добавить новый сценарий необходимо два раза кликнуть левой кнопкой мыши на пустой области (там где написано “Double-click here to add a new scenario”). После этого откроется окно свойств нового сценария. Используя кнопку “Browse…” откройте файл Movie Theater Analysis.p находящийся в папке \Arena\Examples (рис. 2.13).

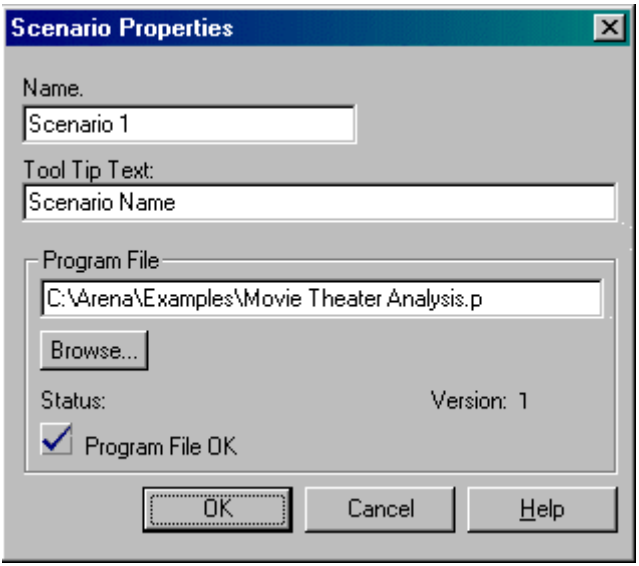


Рис. 2.13. Свойства сценария

После нажатия кнопки OK, окно закроется, а в таблице сценариев появится новая строка (рис. 2.14).

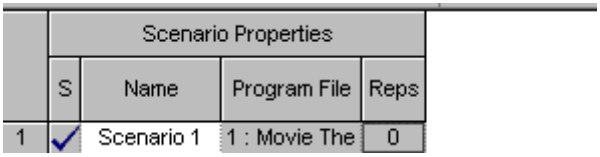


Рис. 2.14. Таблица сценариев

Добавление контролера

Клик правой кнопкой мыши на новой строке сценария, откроет выпадающее меню. Выберите пункт Insert Control для установления нового контролера.

В появившемся окне выберете пункт "Resources" в колонке Controls, для того что бы увидеть выпадающее дерево со всеми доступными контролерами относящимися к ресурсам. Затем выберите ресурс называющийся "Ticket Takers". Обра-

тите внимание на то, что когда выбирается какая либо строка, нижняя часть окна заполняется автоматически, данными описывающими выбранный пункт. Используя установки по умолчанию нажмите OK и строка "Ticket Takers" появится в таблице предыдущего окна. Повторите последний пункт, добавив "Satellite Refreshment Staff" и "Main Refreshment Staff."(рис. 2.15).

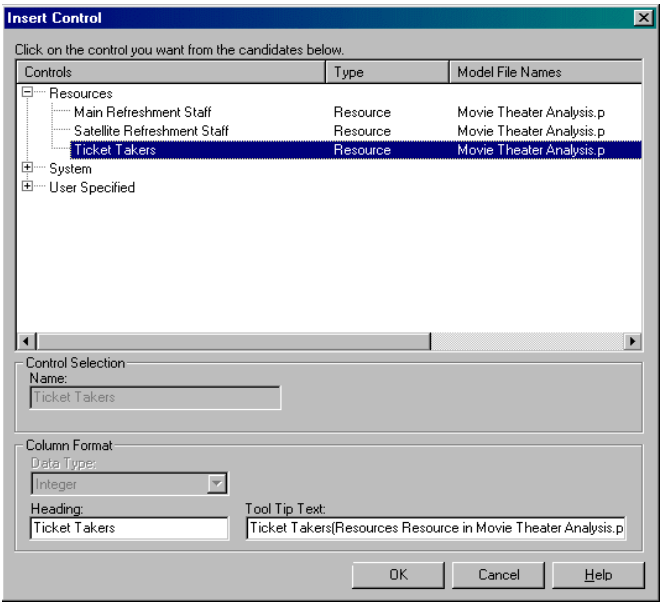


Рис. 2.15. Добавление нового контроллера

После того как были добавлены все три контролера для управления сценарием, вы можете ввести свои значения для них (рис. 2.16).

	Scenario Properties				Controls		
	S	Name	Program File	Reps	Ticket Takers	Satellite Refreshment Staff	Main Refreshment Staff
1	✓	Scenario 1	1 : Movie Theater Analysis.p	0	2	1	2

Рис. 2.16. Таблица сценариев с контроллерами

Добавление результативных переменных

Используя описанную выше процедуру, добавьте к сценарию результативные переменные, единственное отличие в том, что необходимо выбрать пункт выпадающего меню Insert Responses, вместо Insert Control. Они находятся во вкладке "User Specified", в колонке Responses. Необходимые для добавления пункты "Net Profit", “Gross Profit”, “Max Waiting Line Length”, “Number leaving food line” и “Number leaving ticket line” (рис. 2.17).

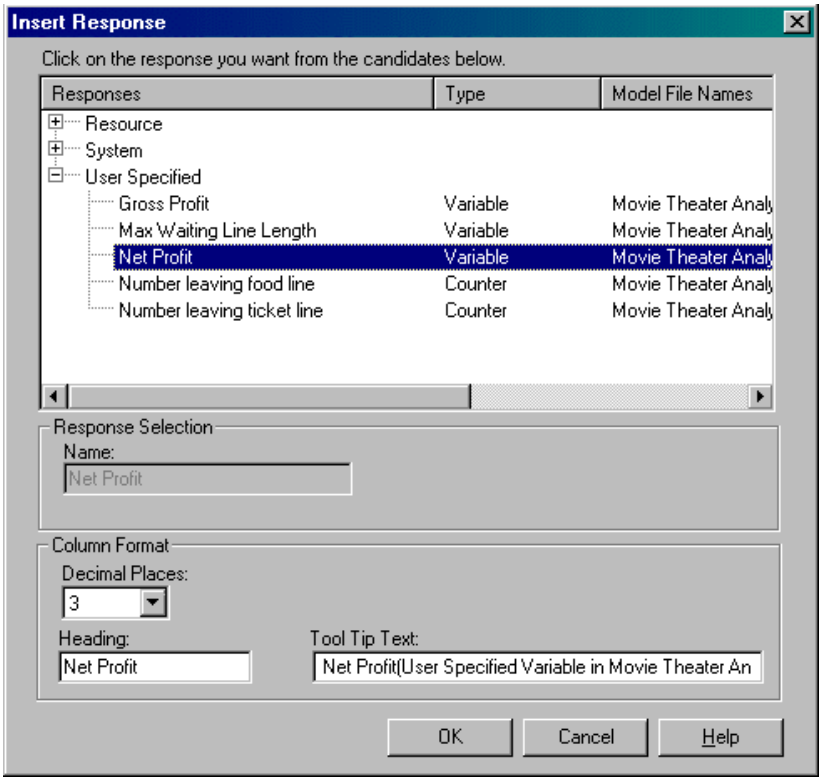


Рис. 2.17. Добавление переменных

Запуск сценария

Создайте несколько копий полученного сценария, изменяя в последующих значения столбцов Resourse.

Для запуска сценария необходимо выбрать сценарий кликом на заголовке строки в таблице сценариев, затем нажать кнопку Go в меню Run. Это запустит диалоговое окно с отображением выбранных для запуска эмуляции сценариев. Нажатие кнопки OK начнет выполнение эмуляции. Если до этого проект не был сохранен эмуляция не будет запущена, но после сохранения запустится автоматически.

Эта модель была установлена на 10 повторов. Когда выполнение сценария закончится синяя буква «A», в строке статуса закончит вращаться, и в столбце статуса рядом с процессами появятся флажки.

Просмотр результатов

После выполнения сценария результаты будут отображаться сразу же в таблице (рис. 2.18). Если данные не могут быть получены, то будет отображаться строка ’r;---’.

Responses		
Net Profit	Number leaving ticket line	Number leaving food line
1062.657	0	136

Рис. 2.18. Результаты выполнения сценария

Построение графиков, основываясь на результатах

Выделив столбец результативных переменных в главном меню станет доступна кнопка Insert Chart которая запустит помощника построения графиков (рис. 2.19).

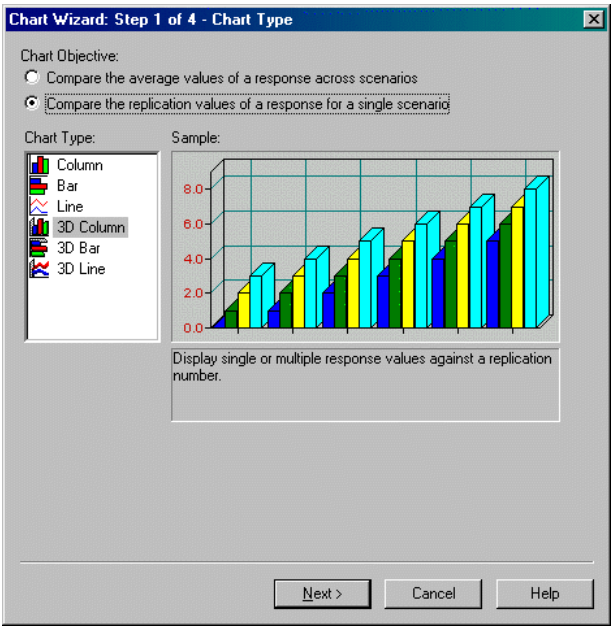


Рис. 2.19. Построение графиков

Используя настройки по умолчанию в помощнике, в нижней половине окна появится график, основанный на результатах сценариев (рис. 2.20).

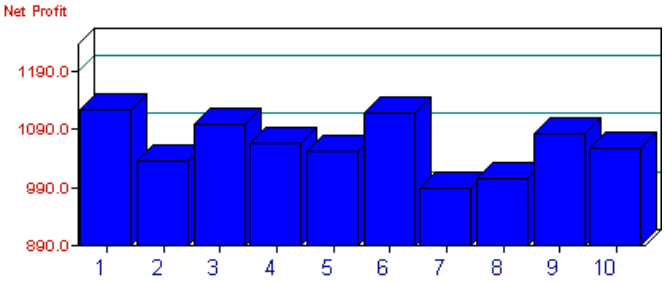


Рис. 2.20. График результатов

2.5. Output Analyzer

Ниже представлен пример использования инструмента Output Analizer

Использование Output Analyzer помогает принимать решения по количеству используемых ресурсов. В нашем случае было изменено количество работников задействованных в первичном осмотре.

Сравнение этих двух потенциальных возможностей отображено на рис. 2.21

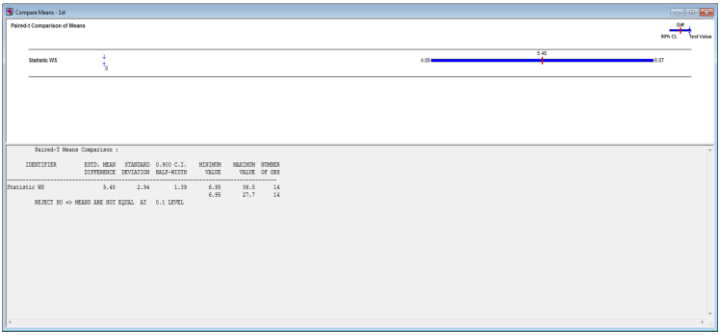


Рис. 2.21. Сравнение двух вероятностей

При значении доверительного интервала в 90%, было получено такое графическое описание. Если бы стрелочки около значения 0 попадали в голубой интервал около значения 5.48, то тогда наем нового работника в цех был бы оправдан.

Гистограммы для каждого из случаев приведенные ниже описывают занятость группы работников на пункте первичного осмотра. В первом случае распределение более равномерно, чем во втором, хотя на работу и уходит больше времени. Во втором же случае количество простоев уменьшилось незначительно, а повременная нагрузка возросла, следовательно, данное изменение количества персонала при прежнем расписании работы цеха не актуально.

Основываясь на полученных результатах можно сказать, что увеличение числа работников на этапе первичного осмотра блоков питания не даст значительного прироста производительности для всего цеха (рис. 2.22 – 2.23).

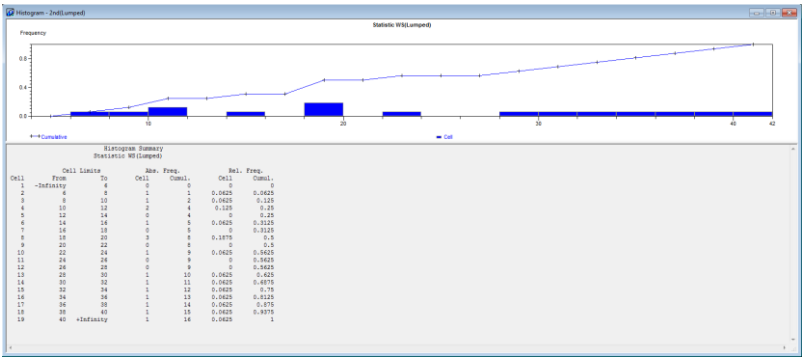


Рис. 2.22. Гистограмма для изначального случая

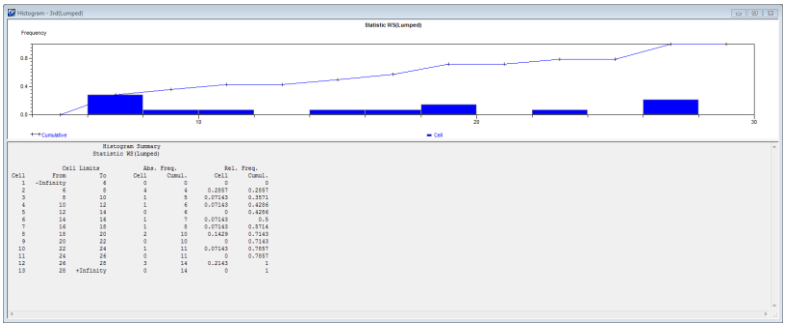


Рис. 2.23. Гистограмма для изучаемого случая

ЗАКЛЮЧЕНИЕ

В данном учебном пособии рассмотрен программный пакет имитационного моделирования Arena. Рассмотрены математические основы данного пакета. Приведено описание основных модулей и дан обзор основных инструментов. Рассмотрены достоинства и недостатки имитационного моделирования систем.

Также даны примеры простого и сложного моделирования систем в ПП Arena, дано описание таких инструментов, как Input Analyzer, Process Analyzer и Output Analyzer, а также пример работы с ними.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. Шакин В.Н. Моделирование систем и сетей связи: учеб. пособие / В.Н. Шакин, Л.А. Воробейчиков, С.Е. Шибанов, Т.И. Семенова. – М.: МИС, 1988.

2. Игельник Б.М. Аналитическое моделирование систем связи: учеб. пособие / Б.М. Игельник, В.М. Лившиц, С.Е. Шибанов. – М.: МИС, 1989.

3. Шеннон Р. Имитационное моделирование систем - искусство и наука: пер. с англ. / Р. Шеннон. - М.: Мир, 1978.

4. Советов Б.Я. Моделирование систем. Практикум: учеб. пособие для вузов / Б.Я. Советов, С.А. Яковлев.- 2-е изд., перераб. и доп. - М.: Высш. шк., 2003. - 295 с.

5. Советов Б.Я. Моделирование систем: учеб. для вузов / Б.Я. Советов, С.А. Яковлев. – 3-е изд., перераб. и доп.- М.: Высш. шк., 2001. – 343 с.: ил.

6. Шрайбер Т.Дж. Моделирование на GPSS: Пер. с англ./ Т.Дж. Шрайбер. - М.: Машиностроение, 1980.

7. Королёв А.Г. Моделирование систем средствами Object GPSS. Практический подход в примерах и задачах: учеб. пособие / А.Г. Королёв. - Луганск: изд-во Восточно-украинского нац. ун-та, 2005. - 307 с.

8. Каштанов Д.И. Документация на комплекс “ВиРОМ 2.0” / Д.И. Каштанов. – Владивосток, 2002.

ОГЛАВЛЕНИЕ

Учебное издание

Воробьёв Эдуард Игоревич

МОДЕЛИРОВАНИЕ СИСТЕМ МАССОВОГО ОБСЛУЖИВАНИЯ В ПАКЕТЕ ARENA 9.0

В авторской редакции

Компьютерный набор Т.И. Куксина

Подписано к изданию 14.11.2013.
Уч.-изд. л. 5,0.

ФГБОУ ВПО «Воронежский государственный технический университет»
394026 Воронеж, Московский просп., 14