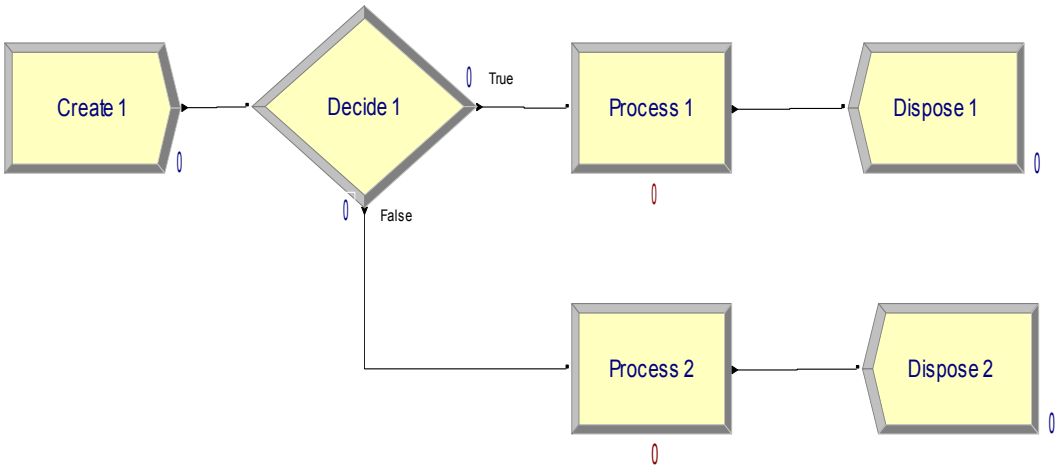
ФЕДЕРАЛЬНОЕ АГЕНТСТВО ПО ОБРАЗОВАНИЮ
Государственное образовательное учреждение
Высшего профессионального образования
«Томский политехнический университет»

О. М. Замятина

МОДЕЛИРОВАНИЕ СИСТЕМ

Учебное пособие



Издательство ТПУ
Томск 2009

УДК 681.3.06
ББК 32.973.2

Замятина О. М.
М 34 Моделирование систем: Учебное пособие. – Томск: Изд-во ТПУ, 2009. – 204 с.

В учебном пособии кратко изложены основы теории моделирования систем, приведены различные виды классификации моделирования и моделей, рассмотрена математическая основа моделирования процессов и систем, рассмотрены методологии структурного анализа и методы и средства имитационного моделирования систем.
Пособие подготовлено на кафедре оптимизации систем управления Томского политехнического университета, соответствует программе дисциплин: «Компьютерное моделирование», «Моделирование систем», «Реинжиниринг бизнес-процессов» и предназначено для студентов технических и экономических специальностей, а также для IT-специалистов.

УДК 681.3.06

Рекомендовано к печати Редакционно-издательским советом
Томского политехнического университета

Рецензенты

М. П. Силич – профессор кафедры автоматизации обработки информации Томского университета систем управления и радиоэлектроники, доктор технических наук;
В. Г. Спицын – профессор кафедры вычислительной техники Томского политехнического университета, доктор технических наук;
А. В. Лиепиньш – начальник отдела технологического проектирования и СМК ОАО «ТомскНИПИнефть», кандидат технических наук.

Введение

Данное учебное пособие ориентировано на студентов технических и экономических 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» 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.

Практическая часть курса позволяет студентам освоить и практически применять одно из самых современных пакетов имитационного моделирования Arena 7.0, в основу которого заложен математический аппарат раскрашенных сетей Петри и систем массового обслуживания. Описание модулей, их свойств, правил моделирования до настоящего времени не опубликованы в русскоязычной литературе, хотя этот инструмент все чаще используется для динамического моделирования. Отдельным параграфом рассмотрен пример разработки имитационной модели.

Учебный материал, ставший основой этого учебного пособия, уже в течение нескольких лет читается студентам Томского политехнического университета и апробированы его теоретическая и практическая части. В конце глав приведены практические задания, которые позволят на практике освоить теоретический материал, а также вопросы, ответы на которые можно найти, проанализировав и осмыслив предшествующую главу.

Автор выражает благодарность всем тем, кто принял участие в подготовке этого пособия, особенно Саночкиной Н. Г., совместная работа с которой принесла позитивные результаты, а также хочется выделить студентов, которые участвовали в этом: Карпову Евгению, Магдик Татьяну и Нгуен Минь Ки.

От автора

Ваши замечания, предложения и вопросы отправляйте по адресу электронной почты zamyatina@tpu.ru. Я буду рада узнать Ваше мнение!

1. Основные понятия теории моделирования

1.1. Модель и моделирование

Слово «модель» (от лат. modelium) означает «мера», «способ», «сходство с какой-то вещью».

Термин «модель» широко используется в различных сферах человеческой деятельности и имеет множество смысловых значений. Мы под «моделью» будем понимать некий материальный или мысленно представляемый объект, который в процессе исследования замещает объект-оригинал так, что его непосредственное изучение дает новые знания об объекте-оригинале.

Модель – это объект или описание объекта, системы для замещения (при определенных условиях, предложениях, гипотезах) одной системы (т. е. оригинала) другой системой для лучшего изучения оригинала или воспроизведения каких-либо его свойств [4, 24].

Модель – это результат отображения одной структуры (изученной) на другую (малоизученную). Любая модель строится и исследуется при определенных допущениях, гипотезах. Модель должна строиться так, чтобы она наиболее полно воспроизводила те качества объекта, которые необходимо изучить в соответствии с поставленной целью [17, 28]. Во всех отношениях модель должна быть проще объекта и удобнее его для изучения. Таким образом, для одного и того же объекта могут существовать различные модели, классы моделей, соответствующие различным целям его изучения. Необходимым условием моделирования является подобие объекта и его модели. В этом случае мы должны говорить об ***адекватности*** модели объекту-оригиналу.

Если результаты моделирования подтверждаются и могут служить основой для прогнозирования процессов, протекающих в исследуемых объектах, то говорят, что модель адекватна объекту. При этом адекватность модели зависит от цели моделирования и принятых критериев.

Под адекватной моделью понимается модель, которая с определенной степенью приближения на уровне понимания моделируемой системы разработчиком модели отражает процесс ее функционирования во внешней среде. Под ***адекватностью*** (от лат. adaequatus – приравненный) будем понимать степень соответствия результатов, полученных по разработанной модели, данным эксперимента или тестовой задачи. Если система, для которой разрабатывается модель, существует, то сравнивают выходные данные модели и этой системы. В том случае, когда два набора данных оказываются подобными, модель существующей системы считается адекватной. Чем больше общего между сущест-

вующей системой и ее моделью, тем больше уверенность в правильности модели системы.

Проверка адекватности модели необходима для того, чтобы убедиться в справедливости совокупности гипотез, сформулированных на первом этапе разработки модели, и точности полученных результатов; соответствует точности, требуемой техническим заданием.

Для моделей, предназначенных для приблизительных расчетов, удовлетворительной считается точность 10-15 %, а для моделей, предназначенных для использования в управляющих и контролирующих системах, – 1-2 % [28].

Любая модель обладает следующими свойствами:

– конечностью: модель отображает оригинал лишь в конечном числе его отношений;

– упрощенностью: модель отображает только существенные стороны объекта;

– приблизительностью: действительность отображается моделью грубо или приблизительно;

– адекватностью: модель успешно описывает моделируемую систему;

– информативностью: модель должна содержать достаточную информацию о системе в рамках гипотез, принятых при построении модели.

Процесс построения, изучения и применения моделей будем называть моделированием, т. е. можно сказать, что ***моделирование*** – это метод исследования объекта путем построения и исследования его модели, осуществляемое с определенной целью, и состоит в замене эксперимента с оригиналом экспериментом на модели.

Моделирование базируется на математической теории подобия, согласно которой абсолютное подобие может иметь место лишь при замене одного объекта другим, точно таким же. При моделировании большинства систем (за исключением, возможно, моделирования одних математических структур другими) абсолютное подобие невозможно, и основная цель моделирования – модель достаточно хорошо должна отображать функционирование моделируемой системы [2].

1.2. Классификация моделей

В общем случае все модели, независимо от областей и сфер их применения, бывают трех типов: познавательные, прагматические и инструментальные.

Познавательная модель – форма организации и представления знаний, средство соединения новых и старых знаний. Познавательная

модель обычно подгоняется под реальность и является теоретической моделью.

Прагматическая модель – средство организации практических действий, рабочего представления целей системы для ее управления. Реальность в них подгоняется под некоторую прагматическую модель. Это, как правило, прикладные модели.

Инструментальная модель – средство построения, исследования и/или использования прагматических и/или познавательных моделей.

Познавательные отражают существующие, а прагматические – хоть и не существующие, но желаемые и, возможно, исполнимые отношения и связи.

Вся остальная классификация моделей выстраивается по отношению к объекту-оригиналу, методам изучения и т. п.

1.2.1. Классификация моделей по степени абстрагирования модели от оригинала

По степени абстрагирования от оригинала (см. рис. 1.1) модели могут быть разделены на материальные (физические) и идеальные. К ***материальным*** относятся такие способы, при которых исследование ведется на основе модели, воспроизводящей основные геометрические, физические, динамические и функциональные характеристики изучаемого объекта. Основными разновидностями физических моделей являются [17]:

- натурные;
- квазинатурные;
- масштабные;
- аналоговые.

Натурные – это реальные исследуемые системы, которые являются макетами и опытными образцами. Натурные модели имеют полную адекватность с системой-оригиналом, что обеспечивает высокую точность и достоверность результатов моделирования; другими словами, модель натурная, если она есть материальная копия объекта моделирования. Например, глобус – натурная географическая модель земного шара.

Квазинатурные (от лат. «квази» – почти) – это совокупность натурных и математических моделей. Этот вид моделей используется в случаях, когда математическая модель части системы не является удовлетворительной или когда часть системы должна быть исследована во взаимодействии с остальными частями, но их еще не существует либо их включение в модель затруднено или дорого.

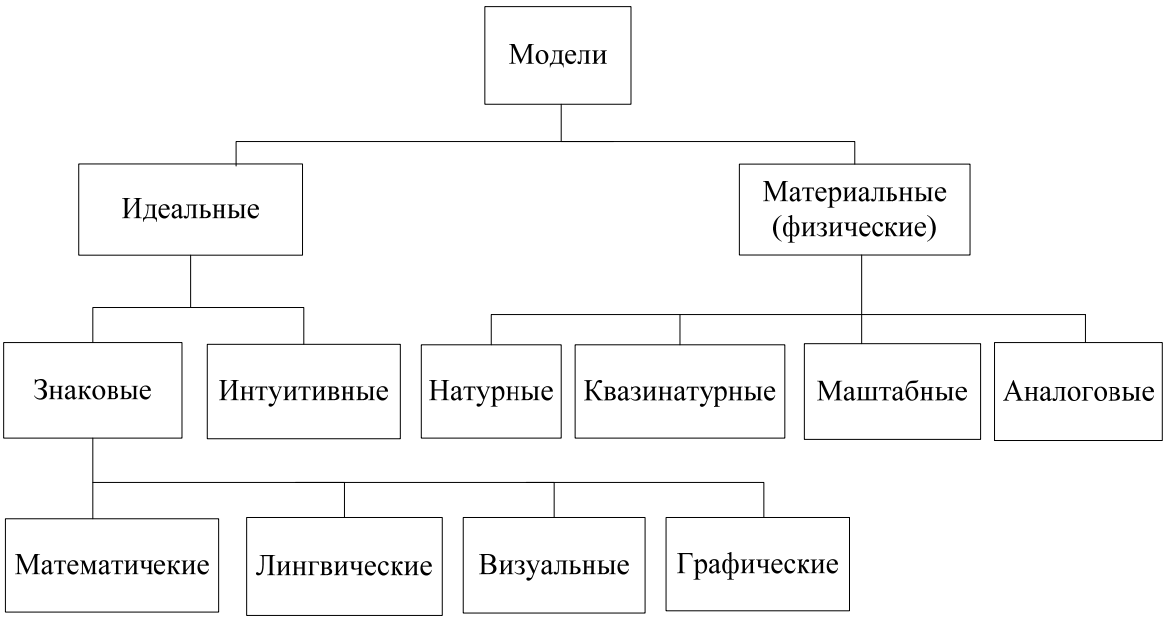


Рис. 1.1. Схема классификации моделей
по степени абстрагирования от объекта-оригинала

Масштабные модели – это системы той же физической природы, что и оригинал, но отличающиеся от него размерами. В основе масштабных моделей лежит математический аппарат теории подобия, который предусматривает соблюдение геометрического подобия оригинала и модели и соответствующих масштабов для их параметров. Примером масштабного моделирования являются любые разработки макетов домов, а порой и целых районов при проведении проектных работ при строительстве. Также масштабное моделирование используется при проектировании крупных объектов в самолетостроении и кораблестроении.

Аналоговое моделирование основано на аналогии процессов и явлений, имеющих различную физическую природу, но одинаково описываемых формально (одними и теми же математическими уравнениями, логическими схемами и т. п.). В качестве аналоговых моделей используются механические, гидравлические, пневматические системы, но наиболее широкое применение получили электрические и электронные аналоговые модели, в которых сила тока или напряжение является аналогами физических величин другой природы. Например, является общеизвестным, что математическое уравнение колебания маятника имеет эквивалент при записи уравнения колебаний тока.

Идеальное моделирование носит теоретический характер. Различают два типа идеального моделирования: интуитивное и знаковое.

Под ***интуитивным*** будем понимать моделирование, основанное на интуитивном представлении об объекте исследования, не поддающемся формализации либо не нуждающемся в ней. В этом смысле, на-

пример, жизненный опыт каждого человека может считаться его интуитивной моделью окружающего мира.

Знаковым называется моделирование, использующее в качестве моделей знаковые преобразования различного вида: схемы, графики, чертежи, формулы, наборы символов и т. д., включающие совокупность законов, по которым можно оперировать с выбранными знаковыми элементами. Знаковая модель может делиться на лингвистическую, визуальную, графическую и математическую модели.

Модель ***лингвистическая***, – если она представлена некоторым лингвистическим объектом, формализованной языковой системой или структурой. Иногда такие модели называют вербальными, например, правила дорожного движения – языковая, структурная модель движения транспорта и пешеходов на дорогах.

Модель ***визуальная***, – если она позволяет визуализировать отношения и связи моделируемой системы, особенно в динамике. Например, на экране компьютера часто пользуются визуальной моделью объектов, клавиатуры в программе-тренажере по обучению работе на клавиатуре.

Модель ***графическая***, – если она представима геометрическими образами и объектами, например, макет дома является натурной геометрической моделью строящегося дома.

Важнейшим видом знакового моделирования является ***математическое*** моделирование, классическим примером математического моделирования является описание и исследование основных законов механики И.Ньютона средствами математики.

Классификация математических моделей

Математические модели классифицируются:
– по принадлежности к иерархическому уровню;
– характеру отображаемых свойств объекта;
– способу представления свойств объекта;
– способу получения модели;
– форме представления свойств объекта.

• ***По принадлежности к иерархическому уровню*** математические модели делятся на модели микроуровня, макроуровня, метауровня (см. рис. 1.2).

Математические модели на ***микроуровне*** процесса отражают физические процессы, протекающие, например, при резании металлов. Они описывают процессы на уровне перехода (прохода).

Математические модели на ***макроуровне*** процесса описывают технологические процессы.

Математические модели на ***метауровне*** процесса описывают технологические системы (участки, цехи, предприятие в целом).

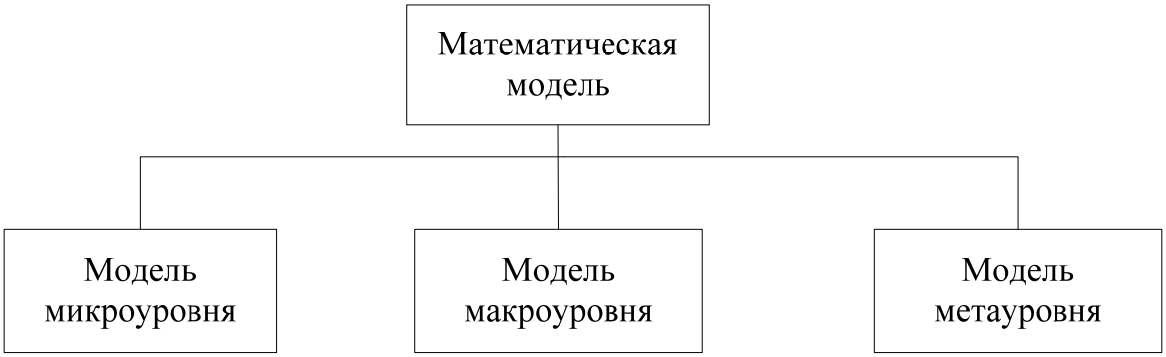


Рис. 1.2. Схема классификации математических моделей по принадлежности к иерархическому уровню

- ***По характеру отображаемых свойств объекта*** модели можно классифицировать на структурные и функциональные (рис. 1.3).

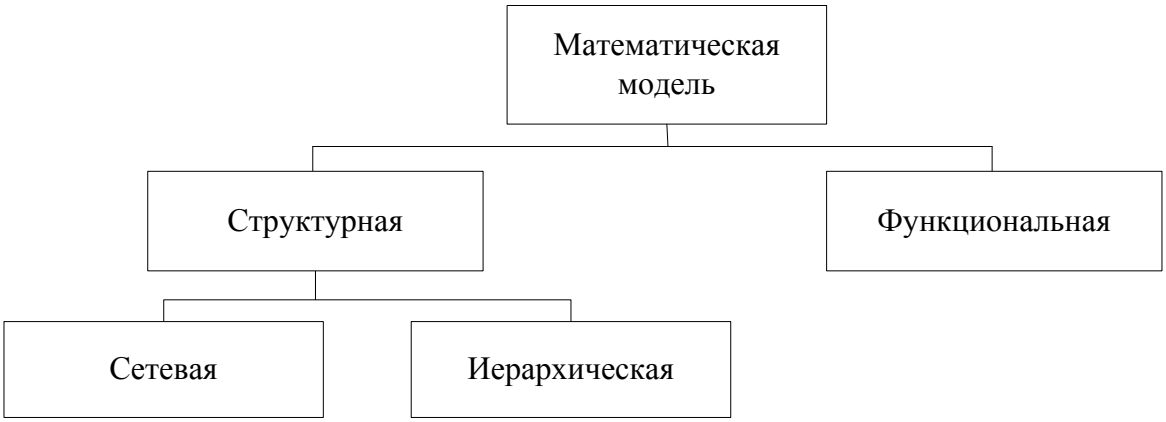


Рис. 1.3. Схема классификации математических моделей по характеру отображаемых свойств объекта

Модель ***структурная***, – если она представима структурой данных или структурами данных и отношениями между ними; например, структурной моделью может служить описание (табличное, графовое, функциональное или другое) трофической структуры экосистемы. В свою очередь, структурная модель может быть иерархической или сетевой.

Модель ***иерархическая*** (древовидная), – если представима некоторой иерархической структурой (деревом); например, для решения задачи нахождения маршрута в дереве поиска можно построить древовидную модель, приведенную на рис. 1.4.

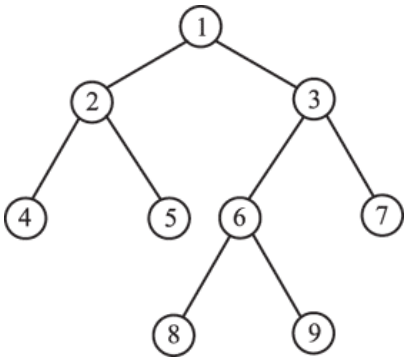


Рис. 1.4. Модель иерархической структуры

Модель ***сетевая***, – если она представима некоторой сетевой структурой. Например, строительство нового дома включает операции, приведенные в нижеследующей таблице. Эти операции можно представить в виде сетевой модели, приведенной на рис. 1.5 и в табл. 1.1.

Таблица 1.1

Таблица работ при строительстве дома

№	Операция	Время выполнения (дни)	Предшествующие операции	Дуги графа
1	Расчистка участка	1	Нет	–
2	Закладка фундамента	4	Расчистка участка (1)	1–2
3	Возведение стен	4	Закладка фундамента (2)	2–3
4	Монтаж электропроводки	3	Возведение стен (3)	3–4
5	Штукатурные работы	4	Монтаж электропроводки (4)	4-5
6	Благоустройство территории	6	Возведение стен (3)	3–6
7	Отделочные работы	4	Штукатурные работы (5)	5–7
8	Настил крыши	5	Возведение стен (3)	3–8

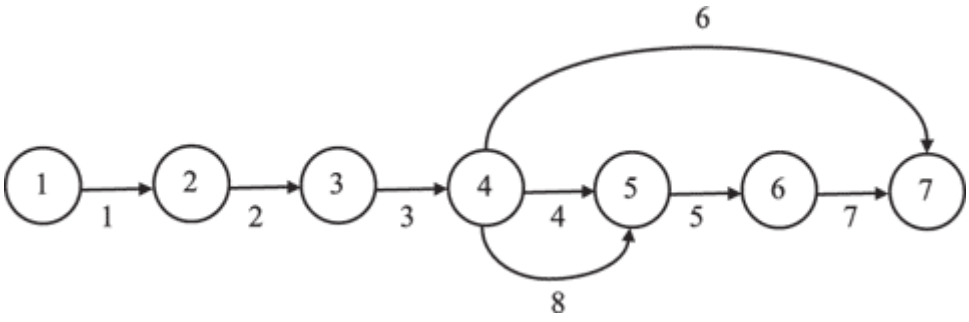


Рис. 1.5. Сетевой график строительства работ

Модель ***функциональная***, – если она представима в виде системы функциональных соотношений. Например, закон Ньютона и модель производства товаров – функциональные.

- ***По способу представления свойств объекта*** (рис. 1.6) модели делятся на аналитические, численные, алгоритмические и имитационные [18].

Рис. 1.6. Схема классификации математических моделей по способу представления свойств объекта

Аналитические математические модели представляют собой явные математические выражения выходных параметров как функций от параметров входных и внутренних и имеют единственные решения при любых начальных условиях. Например, процесс резания (точения) с точки зрения действующих сил представляет собой аналитическую модель. Также квадратное уравнение, имеющее одно или несколько решений, будет аналитической моделью.

Модель будет ***численной***, если она имеет решения при конкретных начальных условиях (дифференциальные, интегральные уравнения).

Модель ***алгоритмическая***, – если она описана некоторым алгоритмом или комплексом алгоритмов, определяющим ее функционирование и развитие. Введение данного типа моделей (действительно, кажется, что любая модель может быть представлена алгоритмом её исследования) вполне обосновано, т. к. не все модели могут быть исследованы или реализованы алгоритмически. Например, моделью вычисления суммы бесконечного убывающего ряда чисел может служить алгоритм вычисления конечной суммы ряда до некоторой заданной степени точности. Алгоритмической моделью корня квадратного из числа X мо-

жет служить алгоритм вычисления его приближенного, сколь угодно точного значения по известной рекуррентной формуле.

Модель ***имитационная***, – если она предназначена для испытания или изучения возможных путей развития и поведения объекта путем варьирования некоторых или всех параметров модели, например модель экономической системы производства товаров двух видов. Такую модель можно использовать в качестве имитационной с целью определения и варьирования общей стоимости в зависимости от тех или иных значений объемов производимых товаров.

- ***По способу получения модели*** делятся на теоретические и эмпирические (рис. 1.7).

Теоретические математические модели создаются в результате исследования объектов (процессов) на теоретическом уровне. Например, существуют выражения для сил резания, полученные на основе обобщения физических законов. Но они неприемлемы для практического использования, т. к. очень громоздки и не совсем адаптированы к реальным процессам обработки материалов.

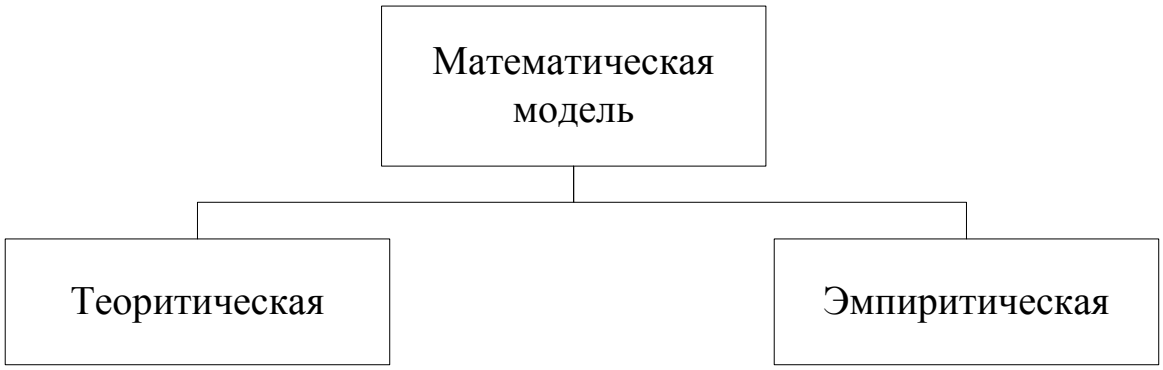


Рис. 1.7. Схема классификации математических моделей по способу получения модели

Эмпирические математические модели создаются в результате проведения экспериментов (изучения внешних проявлений свойств объекта с помощью измерения его параметров на входе и выходе) и обработки их результатов методами математической статистики.

- ***По форме представления свойств объекта*** модели делятся на логические, теоретико-множественные и графовые (рис. 1.8).

Модель ***логическая***, если она представима предикатами, логическими функциями, например, совокупность двух логических функций может служить математической моделью одноразрядного сумматора.

Модель ***теоретико-множественная***, – если она представима с помощью некоторых множеств и отношений принадлежности к ним и между ними.

Модель ***графовая***, – если она представима графом или графами и отношениями между ними.

Рис. 1.8. Схема классификации математических моделей по форме представления свойств объекта

1.2.2. Классификация моделей по степени устойчивости

Все модели могут быть разделены на устойчивые и неустойчивые (см. рис. 1.9).

Устойчивой является такая система, которая, будучи выведена из своего исходного состояния, стремится к нему. Она может колебаться некоторое время около исходной точки, подобно обычному маятнику, приведенному в движение, но возмущения в ней со временем затухают и исчезают.

Рис. 1.9. Схема классификации математических моделей по устойчивости

В ***неустойчивой*** системе, находящейся первоначально в состоянии покоя, возникшее возмущение усиливается, вызывая увеличение

значений соответствующих переменных или их колебания с возрастающей амплитудой.

1.2.3. Классификация моделей по отношению к внешним факторам

По отношению к внешним факторам модели могут быть разделены на открытые и замкнутые.

Замкнутой моделью является модель, которая функционирует вне связи с внешними (экзогенными) переменными. В замкнутой модели изменения значений переменных во времени определяются внутренним взаимодействием самих переменных. Замкнутая модель может выявить поведение системы без ввода внешней переменной. Пример: информационные системы с обратной связью являются замкнутыми системами. Это самонастраивающиеся системы, и их характеристики вытекают из внутренней структуры и взаимодействий, которые отражают ввод внешней информации.

Модель, связанная с внешними (экзогенными) переменными, называется ***открытой.***

1.2.4. Классификация моделей по отношению ко времени

По отношению к временному фактору модели делятся на динамические и статические (рис. 1.10).

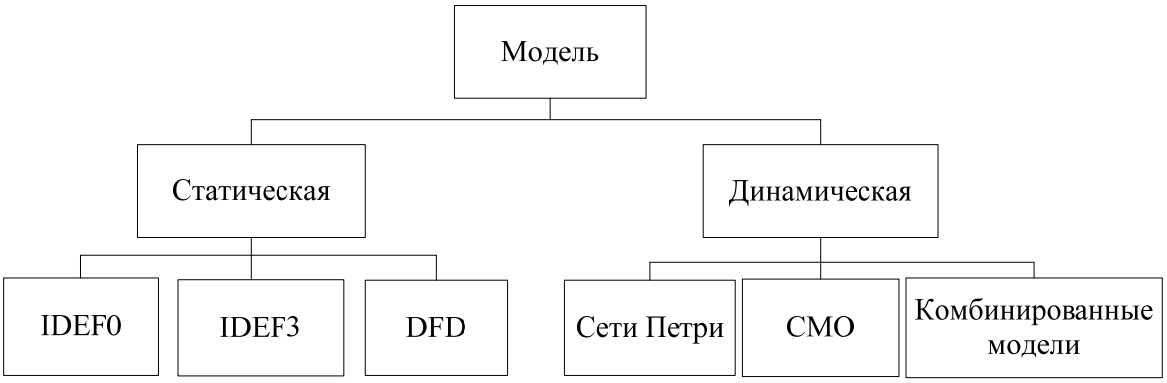


Рис. 1.10. Схема классификации математических моделей по отношению ко времени

Модель называется ***статической***, если среди параметров, участвующих в ее описании, нет временного параметра. Статическая модель в каждый момент времени дает лишь «фотографию» системы, ее срез. Одним из видов статических моделей являются структурные модели.

Динамической моделью называется модель, если среди ее параметров есть временной параметр, т. е. она отображает систему (процессы в системе) во времени.

Во второй и третьей главе этого учебного пособия будут рассматриваться методологии и средства компьютерного моделирования, позволяющие разрабатывать статические и динамические модели. Вторая глава будет посвящена методологиям структурного анализа: IDEF0, IDEF3 и DFD.

Третья глава пособия позволит изучить имитационное моделирование систем на примере современного программного пакета Arena 7.0. Эти имитационные модели, в свою очередь, являются дискретными, динамическими и стохастическими одновременно. Такой вид моделей чаще всего называют дискретно-событийным, он используется для построения моделей, отражающих развитие системы во времени, когда состояние переменных системы меняется в конкретные моменты времени. В такие моменты времени происходят события, которые изменяют состояния системы.

1.3. Этапы разработки моделей

В этой работе мы будем рассматривать процесс создания компьютерной модели. Процесс моделирования имеет итерационный характер и проводится в рамках ранее сформулированных целей и с соблюдением границ моделирования. Построение начинается с изучения (обследования) реальной системы, ее внутренней структуры и содержания взаимосвязей между ее элементами, а также внешних воздействий и завершается разработкой модели.

Моделирование – от постановки задачи до получения результатов – проходит следующие этапы:

I. Анализ требований и проектирование.

1. Постановка и анализ задачи и цели моделирования.
2. Сбор и анализ исходной информации об объекте моделирования.
3. Построение концептуальной модели.
4. Проверка достоверности концептуальной модели.

II. Разработка модели.

1. Выбор среды моделирования.
2. Составление логической модели.
3. Назначение свойств модулям модели.
4. Задание модельного времени.
5. Верификация модели.

III. Проведение эксперимента.

1. Запуск модели, прогон модели.
2. Варьирование параметров модели и сбор статистики.
3. Анализ результатов моделирования.

IV. Подведение итогов моделирования согласно поставленной цели и задачи моделирования.

Схема этапов моделирования представлена на рис. 1.11.

Рис. 1.11. Схема создания модели

Необходимо отметить, что при разработке конкретных моделей с определенными целями и границами моделирования необязательно все подэтапы должны выполняться. Например, при разработке статических моделей IDEF0, DFD 3 и 4 подэтапы «Разработки модели» не выполняются, т. к. эти методологии не предусматривают задание временных параметров модели.

На первом этапе моделирования – «***Анализ требований и проектирование***» – формулируется концептуальная модель, строится ее формальная схема и решается вопрос об эффективности и целесообразности моделирования системы.

Концептуальная модель (КМ) – это абстрактная модель, определяющая состав и структуру системы, свойства элементов и причинно-следственные связи, присущие анализируемой системе и существенные для достижения целей моделирования. В таких моделях обычно в словесной форме приводятся сведения о природе и параметрах (характеристиках) элементарных явлений исследуемой системы, о виде и степени взаимодействия между ними, о месте и значении каждого элементарного явления в общем процессе функционирования системы. При создании КМ практически параллельно формируется область исходных дан-

ных (информационное пространство системы) – этап подготовки исходных данных. На данном этапе выявляются количественные характеристики (параметры) функционирования системы и ее элементов, численные значения которых составят исходные данные для моделирования. Очевидно, что значительная часть параметров системы – это случайные величины. Поэтому особое значение при формировании исходных данных имеют выбор законов распределения случайных величин, аппроксимация функций и т. д. В результате выявления свойств модели и построения концептуальной модели необходимо проверить адекватность модели.

На втором этапе моделирования – «***Разработка модели***» – происходит уточнение или выбор программного пакета моделирования. Выбор средств моделирования: программные и технические средства выбираются с учетом ряда критериев. Непременное условие при этом – достаточность и полнота средств для реализации концептуальной модели. Среди других критериев можно назвать доступность, простоту и легкость освоения, скорость и корректность создания программной модели.

После выбора среды проектирования концептуальная модель, сформулированная на предыдущем этапе, воплощается в компьютерную модель, т. е. решается проблема алгоритмизации и детализации модели.

Модель системы представляется в виде совокупности частей (элементов, подсистем). В эту совокупность включаются все части, которые обеспечивают сохранение целостности системы, с одной стороны, а с другой – достижение поставленных целей моделирования (получения необходимой точности и достоверности результатов при проведении компьютерных экспериментов над моделью). В дальнейшем производится окончательная детализация, локализация (выделение системы из окружающей среды), структуризация (указание и общее описание связей между выделенными элементами системы), укрупненное описание динамики функционирования системы и ее возможных состояний.

Для того чтобы выполнить подэтап «Задание модельного времени» введем понятие модельного времени. В компьютерной модели переменная, обеспечивающая текущее значение модельного времени, называется ***часами модельного времени.***

Существует два основных подхода к продвижению модельного времени: ***продвижение времени от события к событию*** и ***продвижение времени с постоянным шагом*** [14].

Подход, использующий продвижение времени в модели от события к событию, применяется всеми основными компьютерными программами и большинством разработчиков, создающих свои модели на универсальных языках (рис. 1.12) [14].

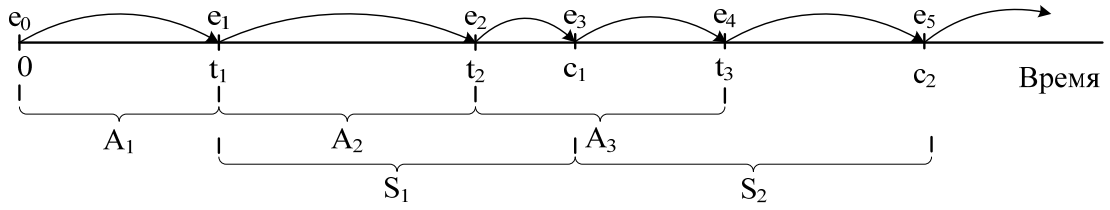


Рис. 1.12. Механизм продвижения модельного времени
от события к событию

При использовании продвижения времени от события к событию часы модельного времени в исходном состоянии устанавливаются в 0, и определяется время возникновения будущих событий. После этого часы модельного времени переходят на время возникновения ближайшего события, и в этот момент обновляются состояние системы, с учетом произошедшего события, а также сведения о времени возникновения будущих событий. Затем часы модельного времени продвигаются ко времени возникновения следующего нового ближайшего события, обновляется состояние системы и определяется время будущих событий и т. д. Процесс продвижения модельного времени от времени возникновения одного события ко времени возникновения другого продолжается до тех пор, пока не будет выполнено какое-либо условие останова, указанное заранее. Поскольку в дискретно-событийной имитационной модели все изменения происходят только во время возникновения событий, периоды бездействия системы просто пропускаются, и часы переводятся со времени возникновения одного события на время возникновения другого. При продвижении времени с постоянным шагом такие периоды бездействия не пропускаются, что приводит к большим затратам компьютерного времени. Следует отметить, что длительность интервала продвижения модельного времени от одного события к другому может быть различной [14].

При продвижении времени с постоянным шагом Δt часы модельного времени продвигаются точно на Δt единиц времени для какого-либо соответствующего выбора значения Δt. После каждого обновления часов выполняется проверка, чтобы определить, произошли какие-либо события в течение предыдущего интервала времени Δt или нет. Если на этот интервал запланированы одно или несколько событий, считается, что данные события происходят в конце интервала, после чего состояние системы и статистические счетчики соответствующим образом обновляются. Продвижение времени посредством постоянного шага показано на рис. 1.13, где изогнутые стрелки показывают продви-

жение часов модельного времени, а e_i (i = 1, 2, ...) – это действительное время возникновения события i любого типа, а не значение часов модельного времени. На интервале [0, Δt) событие происходит в момент времени e_1, но оно рассматривается как произошедшее в момент времени Δt. На интервале [Δt, $2\Delta t$) события не происходят, но все же модель выполняет проверку, чтобы убедиться в этом. На интервале [$2\Delta t$, $3\Delta t$) события происходят в моменты времени e_2 и e_3, однако считается, что они произошли в момент времени $3\Delta t$ и т. д. В ситуациях, когда принято считать, что два или несколько событий происходят в одно и то же время, необходимо применение ряда правил, позволяющих определять, в каком порядке обрабатывать события. Таким образом, продвижение времени посредством постоянного шага имеет два недостатка: возникновение ошибок, связанных с обработкой событий в конце интервала, в течение которого они происходят, а также необходимость решать, какое событие обрабатывать первым, если события, в действительности происходящие в разное время, рассматриваются как одновременные. Подобного рода проблемы можно частично решить, сделав интервалы Δt менее продолжительными, но тогда возрастает число проверок возникновения событий, что приводит к увеличению времени выполнения задачи. Принимая во внимание это обстоятельство, продвижение времени с помощью постоянного шага не используют в дискретно-событийных имитационных моделях, когда интервалы времени между последовательными событиями могут значительно отличаться по своей продолжительности [14].

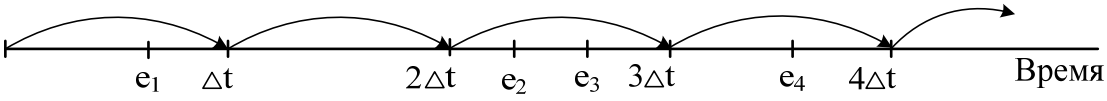


Рис. 1.13. Пример продвижения модельного времени посредством постоянного шага

В основном этот подход предназначен для систем, в которых можно допустить, что все события в действительности происходят в один из моментов n времени Δt (n = 0, 1, 2, ...) для соответственно выбранного Δt. Так, в экономических системах данные часто предоставляются за годичные промежутки времени, поэтому естественно в имитационной модели установить продвижение времени с шагом, равным одному году. Следует заметить, что продвижение времени посредством постоянного шага может быть выполнено с помощью механизма продвижения времени

от события к событию, если планировать время возникновения событий через Δt единиц времени, т. е. данный подход является разновидностью механизма продвижения времени от события к событию.

Третий этап – «***Проведение эксперимента***» – является решающим, на котором, благодаря процессу имитации моделируемой системы, происходит сбор необходимой информации, ее статической обработки в интерпретации результатов моделирования, в результате чего принимается решение: либо исследование будет продолжено, либо закончено. Если известен результат, то можно сравнить его с полученным результатом моделирования. Полученные выводы часто способствуют проведению дополнительной серии экспериментов, а иногда и изменению модели. Основой для выработки решения служат результаты тестирования и экспериментов. Если результаты не соответствуют целям моделирования (реальному объекту или процессу), значит, допущены ошибки на предыдущих этапах или входные данные не являются лучшими параметрами в изучаемой области, поэтому разработчик возвращается к одному из предыдущих этапов.

Подэтап «Анализ результатов моделирования» представляет собой всесторонний анализ полученных результатов с целью получения рекомендаций по проектированию системы или ее модификации.

На этапе «Подведение итогов моделирования согласно поставленной цели и задачи моделирования» проводят оценку проделанной работы, сопоставляют поставленные цели с полученными результатами и создают окончательный отчет по выполненной работе.

1.4. Современные средства моделирования, представленные на ИТ рынке

1.4.1. ARIS Toolset

ARIS – это методология, или как называют авторы «концепция» [35, 36], и базирующееся на ней семейство программных продуктов, разработанных компанией IDS Scheer AG (Германия) для структурированного описания, анализа, последующего совершенствования бизнес-процессов предприятия и управления ими, а также подготовки к внедрению сложных информационных систем.

ARIS Toolset имеет сильный графический интерфейс, а также наличие большого числа стандартных объектов для описания бизнес-процессов. В общем, методология ARIS позволяют решить широкий

комплекс задач по организационному проектированию, разработке и сопровождению технического проекта.

ARIS поддерживает четыре типа моделей, отражающих различные аспекты исследуемой системы:

– организационные модели, представляющие структуру системы – иерархию организационных подразделений, должностей и конкретных лиц, многообразие связей между ними, а также территориальную привязку структурных подразделений;

– функциональные модели, содержащие иерархию целей, стоящих перед аппаратом управления, с совокупностью деревьев функций, необходимых для достижения поставленных целей;

– информационные модели, отражающие структуру информации, необходимой для реализации всей совокупности функций системы;

– модели управления, представляющие комплексный взгляд на реализацию деловых процессов в рамках системы.

Семейство программных продуктов включает:

1. ARIS Toolset предназначен для выполнения проектов по реинжинирингу и совершенствованию бизнес-процессов. Он позволяет документировать бизнес-процессы, проводить их анализ и дальнейшую оптимизацию. Этот программный продукт можно успешно использовать при разработке решений в области повышения прозрачности бизнеса для анализа, документирования, реинжиниринга и оптимизации бизнес-процессов, а также для управления ими. Он обеспечивает профессиональную поддержку при решении вопросов, связанных с организационным развитием предприятий, внедрением различных управленческих и информационных технологий.

2. ARIS Easy Design – инструментальное средство для начинающих, предназначенное для разработки и реинжиниринга бизнес-процессов.

3. Программный модуль ARIS Web Designer предназначен для разработки бизнес-процессов с использованием Интернет.

4. Программный продукт ARIS Server предназначен для сетевой инсталляции программных продуктов ARIS.

5. Программный модуль ARIS BSC предназначен для выполнения работ по стратегическому управлению компанией.

6. Программный модуль ARIS ABC (Activity Based Cost) предназначен для функционально-стоимостного анализа бизнес-процессов.

7. Программный модуль предназначен для динамического моделирования, анализа и оптимизации бизнес-процессов.

8. Программный модуль ARIS Web Publisher (работает с ARIS Toolset и ARIS Easy Design) предназначен для публикации информации о бизнес-процессах в Интернет.

9. Программный продукт ARIS Process Performance Manager предназначен для измерения, анализа и оптимизации бизнес-процессов.

10. Программный продукт ARIS Quality Management Scout предназначен для выполнения проектов по созданию системы управлению качеством.

11. Программный продукт ARIS Process Risk Scout используется для оценки рисков выполнения бизнес-процессов.

12. Для выполнения работ по внедрению процессно-ориентированных решений на базе mySAP.com разработаны специальные версии основных программных продуктов семейства ARIS: ARIS Toolset for mySAP.com и ARIS Easy Design for mySAP.com.

Также ARIS имеет дополнительные модули-интерфейсы, обеспечивающие интеграцию с системами Microsoft Project, ERwin, Designer/2000, IBM Flowmark (класс workflow), Staffware и т. д.

Основными недостатками ARIS являются сложность его изучения, которое требует значительных временных затрат (при обучении персонала до 5 месяцев) и высокая стоимость.

Наименование	Стоимость	
	Продукт	Ежег. обновление
ARIS ToolSet (проектирование и оптимизация – один пользователь)	$ 9 600	$ 4 800
ARIS Easy Design (проектирование – 3 пользователя)	$ 8 820	$ 1 470
ARIS ABC (ABC- анализ)	$ 3 360	$ 1 680
ARIS Server	$ 6 600	$ 3 330
ARIS Simulation	$ 3 360	$ 3 330

Достоинством семейства программных продуктов ARIS является не высокие требования к аппаратному и программному обеспечению. Аппаратное обеспечение: процессор Intel Pentium 166 МГц, 64 МБ ОП (рекомендуется 128 МБ), монитор с разрешением 640x480 (SVGA), 256 цветов (рекомендуется 1024x768, не менее 256 цветов).

Программное обеспечение: операционная система: Windows 95/98/2000/NT V40+Service Pack 4+MS Y2K Patch, Service Pack 5 или 6a).

Документация по ARIS представляет собой инструкции по установке и началу эксплуатации, также документация доступна как помощь online и на компакт-диске в форме текста подготовленного для печати.

1.4.2. ITHINK

ITHINK представляет собой инструмент управления и планирования, является средством экспертного анализа ситуации, разработанный компанией High Performance Systems (США). Аналогом ITHINK является программный продукт, предназначенный для составления графических диаграмм, использующихся при анализе бизнес-ситуации, iDecide 2000.

В ITHINK реализованы идеи структурного проектирования Е. Йордана, структурного анализа Т. Де Марко, системного анализа С. Гейна и Т. Сарсона.

ITHINK представляет собой компактный, объектно-ориентированный пакет прикладных программ с Desktop-интерфейсом, обеспечивающий графическую, вычислительную и информационную поддержку процедурам высокоуровневого системного анализа сложных процессов организации управления, бизнеса, финансов, политики и др.

Модель в ITHINK создается путем отображения на экране моделируемых объектов и взаимосвязей. Она выглядит как совокупность стандартных блоков, соединенных стрелками. Стандартные блоки имеются на выбор и могут быть перенесены в «окно» модели при помощи мыши. Стрелки указывают направление финансовых платежей и потоков данных. Перестроение графической части модели приводит к изменениям в ее программе и алгоритме. Таким образом, обеспечивается удобный эффект визуального моделирования данного программного пакета.

С помощью пакета ITHINK может быть смоделирован весь производственно-сбытовой цикл предприятия, начиная от закупки сырья и заканчивая его производством и реализацией. В сферу потенциального применения входит и транспорт (в том числе трубопроводный), сети газо- и водоснабжения и другие распределительные системы.

Система ITHINK позволяет решать следующие задачи системного анализа:

– возможность разработки моделей разнообразных систем;

– построение моделей систем средней сложности как совокупности простых;

– определение механизма взаимодействия компонентов системы с целью поведенческого описания в терминах ее основных характеристик;

– использование и разработка формальных эффективных процедур имитации соответствующих поведенческих описаний для получе-

ния статистических оценок и определения качественных характеристик;

– формирование процедур генерации представительных тестовых наборов, необходимых для исследования нетривиального поведения;

– целенаправленное изучение поведения моделей для составления предварительных расписаний работы с компонентами системы.

В ITHINK существует 4 основных стандартных блока: поток, конвейер, накопитель, распределитель.

Модели инвестиционных операций ITHINK совместимы с любой экспертной системой или базой данной, поддерживающей режим DDE. Единственное требование к экспертным системам заключается в том, что любые данные на выходе выступали в виде дат начала операций, интервалов, срочности, размеров и т. п.

Система ITHINK отличается функциональной простотой и гармоничностью. Пакет ITHINK не требует специальных навыков и владения сложными математическими методами. Также пакет ITHINK не требует больших затрат аппаратного обеспечения (процессор 486 или более мощный, 8 Мб RAM, Windows 95, наличие 13 Мб свободной памяти на жестком диске).

Стоимость Ithink 8.0 (HPS) равна $ 2750 по каталогу стоимости.

1.4.3. Powersim Studio

Пакет PowerSim Studio создан и распространяется фирмой Powersim Software AS (Норвегия). Используемая методология построена на базе классических методов системной динамики, созданных Дж. Форрестером. Пакет является приложением таких известных систем, как SAP, SEM, BPS, но также может использоваться как автономное приложение. Пакет имеет развитые средства визуального программирования и различные расширенные возможности, в том числе встроенные блоки анализа рисков и оптимизации бизнес-процессов.

При разработке моделей используется визуальное программирование. Модель, включающая в себя различные элементы, строится на понятийном уровне. Разработчик располагает на экране элементы, создает связи между ними, вводит различные зависимости, создает временную динамику развития системы. В модель вводятся различные управляющие элементы. Система имеет развитые способы представления результатов моделирования: временные графики, таблицы, гистограммы. Вид результатов может быть легко приведен к требуемому стандарту.

Модели, создаваемые в PowerSim, могут быть как классическими динамическими моделями (т. е. учитывающими изменения моделируе-

мой системы во времени), так и просто расчетными, позволяющими рассчитывать сложные производственные и финансовые показатели. PowerSim позволяет моделировать как дискретные, так и непрерывные процессы.

К расширенным возможностям системы можно отнести: иерархические способы разработки программ (аналог использования подпрограмм на языках высокого уровня), возможность написания алгоритмов на встроенном языке Visual Basic, комбинация анализа рисков и оптимизации, что позволяет найти лучшее (оптимальное) решение, если параметры изменяются по случайному закону.

Для решения задач не соответствующих встроенным возможностям интеграции PowerSim, служит входящий в пакет поставки программного обеспечения (ПО) PowerSim Studio SDK – комплект для разработки программного обеспечения. С его помощью в системы можно интегрировать программы на языках высокого уровня, вызывать модули PowerSim из различных приложений, организовывать связь модели с такими информационными системами, как Oracle, SQL и другими.

Пакет PowerSim Studio имеет развитые средства использования внешних данных из информационной среды предприятия. Он имеет встроенные механизмы работы с обычными текстовыми файлами, файлами Excel и хранилищами данных SAP BW. Использование расширенных возможностей возможна интеграция пакета в любую информационную среду. Также пакет PowerSim Studio обладает мощным встроенным оптимизатором PowerSim Solver 2.5.

Системы, построенные на базе моделей в среде PowerSim Studio, могут использоваться:

– для создания единых производственно-экономических моделей предприятия;

– для расчета различных производственных и экономических показателей;

– для сценарных расчетов и определения, таким образом, влияния управляющих воздействий и анализа их эффективности;

– для анализа рисков;

– для оптимизации деятельности предприятия;

– для формирования оптимальной стратегии на различных горизонтах планирования.

Эффективная работа с программой оптимизации требует определенной подготовленности пользователя. Пользователь должен уметь представить требование к портфелю в виде целевых функций и критериев оптимизации.

Пакет PowerSim Studio требует следующих параметров аппаратного обеспечения: Microsoft Windows 2000, XP или более поздняя версия, минимум 64 Мб RAM, минимум 50 Мб свободного места на жестком диске, Microsoft Internet Explorer 5.0, или более поздней версии.

Стоимость PowerSim Studio равна 4800 р. по каталогу стоимости.

1.4.4. Extend

Extend – универсальный пакет имитационного моделирования процессов модернизации и обслуживания, разработанный компанией Imagine That, Inc. (США).

Пакет Extend имеет мощный графический интерфейс, позволяющий создавать схемы процессов и производить имитационные эксперименты. Имеется возможность просмотра моделей в виде графиков, а также с использованием 2D и 3D анимации.

Некоторые возможности 3D-аниматора:

– создание виртуальной среды, представляющей собой нужный процесс перехода 3D-изображения;

– возможность изучения просмотром с расширенными возможностями (вращение объектов и моделей под любым углом);

– возможность использования встроенной библиотеки 3D-объектов;

– интеграция с созданной моделью.

Extend может использоваться для моделирования как дискретных, так и непрерывных систем.

Моделирование процессов в Extend требует частичного написания кода при задании свойств блоков (пакет Extend содержит внутренний язык для задания новых блоков). Данный пакет поддерживает импорт и экспорт данных с Microsoft Visio (импорт существующих деловых и технических чертежей, а также поддерживает возможность внесения технических чертежей в процесс имитационного моделирования).

Extend – пакет моделирования дискретных процессов, использующийся для моделирования, анализа и улучшения любого процесса, который может быть представлен в виде «блок-схем». Extend позволяет создавать динамические модели разнородных процессов и систем в терминах предметной области, оптимизировать построенную модель. Он позволяет создавать стохастические динамические модели любого предприятия. Динамические модели позволяют оптимизировать, прогнозировать, планировать деятельность предприятий, а также проводить анализ деятельности предприятия на основании полученных моделей и

выдавать рекомендации по улучшению работы конкретного предприятия.

Пакет Extend включает 5 типов элементов:

– блоки (действия);
– стрелки (пути перемещения динамических объектов (тэгов));
– ромбы (разветвления путей);
– очереди тэгов;
– точки ввода тэгов в модель.

Необходимыми техническими затратами являются: процессор класса Intel Pentium – 90МГц, 64 Мб RAM (128 Мб рекомендуется), 150 Мб свободного пространства на жестко диске, CD-ROM, Microsoft Windows Server 2003, XP, 2000, 98, ME (операционные системы должны поддерживать Microsoft.NET 2.0).

1.4.5. GPSS/H

Фирма-разработчик:

GPSS/H – язык для моделирования дискретных систем, разработанный Wolverine Software (США). В основе этого программного продукта лежит язык имитационного моделирования GPSS (General Purpose Simulating System – общецелевая система моделирования).

Основное назначение GPSS – это моделирование систем массового обслуживания, хотя наличие дополнительных встроенных средств позволяет моделировать и некоторые другие системы (например, распределение ресурсов между потребителями). GPSS имеет блочную структуру и может быть легко приспособлен для структурно-функционального моделирования не очень сложных систем.

Основными понятиями языка GPSS являются транзакт, блок и оператор. Транзакт GPSS – это динамический объект, под которым может подразумеваться клиент, требование, вызов или заявка на обслуживание прибором обслуживания. Транзакты в GPSS могут создаваться (вводиться), уничтожаться (выводиться), задерживаться, размножаться, сливаться, накапливаться и т. д. Блок GPSS представляет собой некоторый самостоятельный элемент моделируемой системы. Каждый блок реализует одну или несколько операций над транзактом, группой транзактов или параметрами транзактов, а совокупность блоков составляет моделирующую программу.

Существующая сейчас версия содержит свыше 70 типов блоков, содержит управляющих операторов, позволяющих организацию циклов; поддерживает представление времени как числа с плавающей запятой;

имеет графические средства манипулирования с блок-схемами и гибкий интерфейс связи с C++.

В 1980 году было представлено два программных пакета для использования в GPSS/H, оба с возможностями анимации.

TESS – расширенная система моделирования для использования в компании Pritsker & Associates и авто симулятор AutoGram. Эти анимационные пакеты стали неинтересны пользователям GPSS/H, с момента, когда Wolverine Software в 1990 году разработал собственный анимационный пакет Proof Animation. В авто симуляторе также была программа AutoMod – препроцессор для автоматической генерации текстов GPSS/H программ. В 80-е годы исследователи также использовали RESQ для быстрого ввода моделей GPSS/H (Mathewson, 1989). В 1993 году Элниски представил свою программу, ускоряющую ввод текстов моделей и оформленную, как оболочка GPSS/H, для прогона моделей на компьютерах типа IBM PC (Wolverine, 1993). В начале 90-х MOGUL от High Performance Software был использован для генерации GPSS/H кодов при моделировании систем связи (Rodrigues, 1993).Также в начале 90-х фирма GfL из Аахена (Германия) реализовала GPSS/H EDITOR – ускоритель ввода GPSS/H программ в основном, простым нажатием на кнопки с текстом данных блоков, но без настоящего графического интерфейса, т. е. без меню с символами блок диаграмм блоков (Knepper and Krönchen, 1993). Начиная с 1994 года, вместе с каждой версией GPSS/H Professional поставляется программа UniFit II, позволяющая пользователю подбирать наиболее подходящие вероятностные распределения для своих данных. В середине 90-х была также разработана система SIMSTAT, которая читала и анализировала выходные данные GPSS/H (Crain, 1996).

GPSS/H достаточно прост в освоении, а наличие в нем функций, переменных, стандартных атрибутов, графики и статистических блоков существенно расширяет его возможности.

1.4.6. GPSS World

GPSS World – самая современная версия GPSS для персональных ЭВМ и ОС Windows.

Система GPSS World – это мощная среда компьютерного моделирования общего назначения, разработанная для профессионалов в области моделирования. Это комплексный моделирующий инструмент, охватывающий области как дискретного, так и непрерывного компьютерного моделирования, обладающий высочайшим уровнем интерактивности и визуального представления информации.

На уровне интерфейса GPSS World представляет собой реализацию архитектуры «документ-вид», общей для всех приложений операционной системы Windows. Объекты могут быть открыты в нескольких окнах, изменены и сохранены на постоянных носителях информации. Привычное меню главного окна и блокировка недоступных команд меню, не отвлекая внимания, направляет пользователя к конечной цели. GPSS World был разработан с целью достижения тесной интерактивности даже в многозадачной среде с использованием виртуальной памяти

В GPSS World существует ряд анимационных возможностей. Уровень их реализма изменяется от абстрактной визуализации, не требующей никаких усилий, до высоко реалистических динамических изображений, включающих в себя сложные элементы, созданные пользователем.

GPSS World является объектно-ориентированным языком. Его возможности визуального представления информации позволяют наблюдать и фиксировать внутренние механизмы функционирования моделей. Его интерактивность позволяет одновременно исследовать и управлять процессами моделирования. С помощью встроенных средств анализа данных можно легко вычислить доверительные интервалы и провести дисперсионный анализ. Кроме того, теперь есть возможность автоматически создавать и выполнять сложные отсеивающие и оптимизирующие эксперименты.

Возможности GPSS World:

– объектно-ориентированный интерфейс пользователя, включающий объекты: модель, процесс моделирования, отчет и текст;

– высокопроизводительный транслятор моделей;

– программные эксперименты с автоматическим анализом данных;

– многозадачность позволяет совместно запускать несколько процессов моделирования и экспериментов;

– сохранение и продолжение выполнения запущенных процессов моделирования;

– использование механизма виртуальной памяти позволяет моделям реально достигать размера миллиарда байт;

– ввод/вывод во время выполнения процесса моделирования;

– свыше 20 встроенных вероятностных распределений;

– 17 различных графических окон для наблюдения за выполняющимся процессом моделирования;

– автоматическое интегрирование обыкновенных дифференциальных уравнений любого порядка;

– быстрая и удобная отладка с использованием графического интерфейса;

– автоматические генераторы отсеивающих и оптимизирующих экспериментов;

– пакетный режим с контролируемой процедурой выхода из приложения;

– диалоговые окна ввода блоков;

– настраиваемые интервалы табуляции;

– возможность динамического вызова функций из внешних файлов.

Основное назначение GPSS – это моделирование систем массового обслуживания, хотя наличие дополнительных встроенных средств позволяет моделировать и некоторые другие системы.

Последняя версия GPSS включает в себя 53 типа блоков и 25 команд, а также более чем 35 системных числовых атрибутов, которые обеспечивают текущие переменные состояния, доступные в любом месте модели.

Основными объектами GPSS являются: модель, процесс моделирования, отчет и текст.

GPSS World совместим с GPSS/PC и выдаёт результаты, которые статистически неотличимы от результатов, выдаваемых GPSS/PC. Этот уровень совместимости может быть достигнут исправлением некоторых отличий и запуском процесса моделирования.

Кроме того, доступен ещё более высокий уровень совместимости, называемый режимом совместимости с GPSS/PC. В большинстве случаев можно достигнуть точного повторения результатов. Тем не менее, GPSS World использует новую исполняемую библиотеку. Применяемый в нём метод округления чисел с плавающей запятой немного отличается от используемого в GPSS/PC. Но даже в этом случае большинство моделей GPSS/PC с небольшими изменениями могут давать идентичные результаты при выполнении под управлением коммерческой версии GPSS World в режиме совместимости с GPSS/PC.

1.4.7. SIMPROCESS

SIMPROCESS – это иерархический пакет имитационного моделирования бизнес-процессов, который позволяет строить схему (карту) моделируемого процесса, поддерживает дискретно-событийное моделирование и функционально-стоимостный анализ с использованием ABC-метода. Компания-разработчик SIMPROCESS –CACI Products Company (США).

Данный пакет ориентирован на организации, которым необходимо анализировать различные сценарии развития предприятия и уменьшать риски в соответствии с меняющимися условиями деятельности.

SIMPROCESS поддерживает дискретно-событийное моделирование. Анимированный интерфейс позволяет визуализировать динамику моделирования «узких мест» в модели.

SIMPROCESS содержит следующие компоненты: процессы (Processes), подпроцессы (Alternative Sub-Processes), действия (Activities), сущности (Entities), ресурсы (Resources), соединители (Connectors), площадки (Pads).

SIMPROCESS требует частичного написания программного кода и поддерживает использование UML/XML технологии, XML/XRDL процессы, поддерживает диаграммы Dot, SOAP (Simple Object Acces Protocol), а также экспорт и импорт данных с ODBC и Java, Java RMI, C4ISR/DoDAF, TOGAF, 6 Sigma.

SIMPROCESS имеет модуль «дистанционного управления», с помощью данной настройки, использующей Java в PMO технологии, позволяет по желанию пользователя задать параметры метода, которые будут ссылаться на объект.

Необходимые технические затраты: Windows NT, 2000/XP, 256 MGB минимум, 512 MGB рекомендовано, 50 MB размер файла 32.9 MB.

1.4.8. AllFusion Process Modeler (BPWin)

AllFusion Process Modeler (ранее BPWin) – ведущий инструмент визуального моделирования бизнес-процессов не требующий написания программного кода, который дает возможность наглядно представить любую деятельность или структуру в виде модели, что позволяет оптимизировать работу организации, проверить ее на соответствие стандартам ISO 9000, спроектировать оргструктуру, снизить издержки, исключить ненужные операции, повысить гибкость и эффективность.

AllFusion Process Modeler, разработанный компанией Computer Associates (США), предлагается для использования компаниям, стремящимся к оптимальности и эффективности собственного бизнеса или бизнеса заказчиков; руководителям проектов, бизнес-аналитикам, системным аналитикам, тон-менеджменту предприятий, маркетологам, консультантам, менеджерам по качеству.

AllFusion Process Modeler обладает интуитивно-понятным графическим интерфейсом, быстро и легко осваивается, что позволяет сосредоточиться на анализе самой предметной области, не отвлекаясь на изучение инструментальных средств. AllFusion Process Modeler помогает

быстро создавать и анализировать модели с целью оптимизации деловых и производственных процессов.

Простой в использовании интерфейс предоставляет превосходные возможности заполнения моделей, но репрезентативные свойства BPWin низки и отсутствуют стандартные объекты для описания бизнес-процессов.

AllFusion Process Modeler – поддерживает сразу три стандартные нотации – IDEF0 (функциональное моделирование), DFD (моделирование потоков данных) и IDEF3 (моделирование потоков работ). Эти три основных ракурса позволяют комплексно описать предметную область.

К основным недостаткам данного программного продукта можно отнести: возможность разработки только статических моделей, никак не привязанных к временным параметрам реальных процессов, и неоднозначность, возникающую после разработки иерархической совокупности диаграмм, которая может включать в себя до ста и более диаграмм, и что же делать далее с полученной моделью, как ее анализировать и оптимизировать.

Интегрирован с ERwin (для проектирования БД), Paradigm Plus (для моделирования компонентов ПО), со средством имитационного моделирования Arena; с многочисленным ПО компании CA-Platinum. При необходимости информация может быть экспортирована в другие приложения (например, в Microsoft Word или Excel).

AllFusion Process Modeler имеет широкий набор средств документирования моделей, проектов и содержит собственный генератор отчётов. Также AllFusion Process Modeler позволяет ведение библиотеки типовых бизнес-моделей.

Необходимое аппаратное обеспечение: операционная система: Windows 2000, XP или Server 2003 512 MB RAM 1024 MB RAM; Linux (Red Hat 6.x through 8.x, или compatible) Intel x86. Увеличение объема RAM улучшает работу продукта.

Цены, на данный момент, различны на промежутке: $ 4 245 – $23 685. Относительно низкая стоимость, вероятно, связана с тем, что основные затраты на разработку требований к системе несет департамент правительства США.

1.4.9. ProcessModel

ProcessModel – это инструмент для визуализации, анализа и совершенствования бизнес-процессов различных типов, включая обработку транзакций, обслуживание покупателей, производственные и сборочные операции, транспортные услуги и т. д., разработанный Process-Model, Inc. (США).

ProcessModeler объединяет простую технологию бизнес-диаграмм с мощными возможностями имитационного моделирования со встроенными средствами графической анимации.

Анимация ProcessModeler – это визуальное динамически обновляемое окно результатов с ключевыми индикаторами (показателями) производительности, отображающее изменение значений переменных на экране, изменение графических элементов модели в процессе моделирования и возможность импорта графики из выбранной пользователем программы.

ProcessModel позволяет реализовать следующие опции:

- расписание работы персонала и график рабочего времени;
- приоритетность и возможность прерывания задач;
- возможность выбора различных методов управления;
- планирование емкости операции или процесса;
- возможность задания размера партии обработки;
- расписание назначений;
- определение последовательности работ;
- производственное расписание;
- повышение продуктивности;
- сокращение цикла обработки;
- снижение стоимости;
- управление качеством;
- анализ «узких мест»;
- оценка стоимости по операциям и ресурсам;
- расписание доступности ресурсов для перерывов в работе и простоев.

ProcessModel позволяет создавать иерархические модели для лучшей организации и управления большими проектами. Группы разработчиков могут создавать различные части сложной модели, а потом, объединив их, анализировать процесс целиком.

ProcessModel позволяет проводить анализ «что будет – если» и дает возможность при минимальных временных и ресурсных затратах найти пути совершенствования процессов, даже в тех случаях, когда другие методы невозможно использовать.

Основные блоки ProcessModeller: Activity shape (функция), Decision shape (условие), Links (соединительные линии (связи)) и Or-join (логическое ИЛИ), Split Worksteps (точка слияния), And-joins (логическое И).

Достоинствами ProcessModel являются: возможность его быстрого освоения и легкого использования, особенно для специалистов, владеющих техниками моделирования и интерактивная, контекстно-

зависимая система помощи, которая обеспечивает всю необходимую информацию для быстрого и легкого создания модели предметной области.

ProcessModel обеспечивает поддержку связей между моделями и между объектами модели и Excel, Access, SQL и т. д., полная интеграция с VBA (Visual BASIC for Application).

Вместе с мощной базовой программой, поставляются еще три дополнительных компонента: LIVE Animation (живая анимация), OneStep Modeling (моделирование за один шаг) и Visual Staffing (визуальная работа с ресурсами).

Для установки ProcessModelr на компьютер необходимо наличие платформы Microsoft Windows 2000 или Microsoft Windows XP Professional.

1.4.10. AnyLogic

Компания-разработчик программного продукта AnyLogic является Экс Джей Текнолоджис, динамично развивающаяся российская компания и один из немногих разработчиков коммерческого программного обеспечения для имитационного моделирования в России, имеющий дистрибьюторскую сеть по всему миру.

AnyLogic – инструмент имитационного моделирования, позволяющий эффективно использовать и сочетать все существующие подходы к моделированию.

AnyLogic имеет дружественный пользовательский графический интерфейс, позволяющий не ограничивать себя в средствах описания модели, используя графическое задание моделей и создание интерактивной 2D и 3D анимации, визуально отображающей результаты работы модели в реальном времени.

Анимация в AnyLogic дает возможность наглядно представить динамику всей системы в процессе моделирования. Средства анимации позволяют пользователю легко создать виртуальный мир (совокупность графических образов, мнемосхему и т. п.), управляемый динамическими параметрами модели по законам, определенным пользователем с помощью уравнений и логики моделируемых объектов.

Области применения программного продукта AnyLogic: рынок и конкурентоспособность, управление проектами, социальные и экологические системы, развитие городов, перемещение людей и транспортных средств в непрерывном пространстве, перекрестки, парковки, здания, музеи, очереди, транспорт, перевозки, эвакуация, производственные процессы, здравоохранение и другие.

AnyLogic имеет существенное преимущество перед традиционными инструментами моделирования именно в тех проектах, разработка которых требует выхода за границы одной единственной парадигмы моделирования.

AnyLogic применяется в диапазоне от микромоделей «физического» уровня, где важны конкретные размеры, расстояния, скорости, времена, до макромоделей «стратегического» уровня, на котором рассматривается глобальная динамика обратных связей, тенденции на длительных временных отрезках и оцениваются стратегические решения.

AnyLogic поддерживает как моделирование систем с дискретными, так и моделей с непрерывными событиями, а также комбинировать их.

Построение модели в AnyLogic не требует написания программного кода, но если стандартных средств не хватает (или их использование неудобно), есть возможность использования языка Java. В простейшем случае, это сводится к описанию действий, совершаемых при переходе в другое состояние, срабатывании таймера или приходе сообщения. Кроме того, можно добавлять собственный код на Java к активному объекту, а также использовать сторонние библиотеки. Это делает систему AnyLogic легко расширяемой.

Любой объект модели, разрабатываемой в AnyLogic, представляется как класс Java, пользователь может добавить в модель свои классы, переопределять методы базовых классов, использовать базовые и разработать свои библиотеки классов и т. п. По модели, представленной в графическом редакторе, AnyLogic генерирует Java программу, с которой работает написанный на Java «движок». При построении модели в AnyLogic разработчик, фактически, создает Java-классы активных объектов и определяет отношения между ними. Во время выполнения модель представляет собой иерархию экземпляров активных объектов. Собранная модель может работать локально, на одном компьютере, или же пользователь может одним кликом мыши построить Java-апплет, который можно запустить под управлением браузера.

Простота освоения AnyLogic определяется знанием пользователя языка Java , который используется в комбинации с графической средой разработки моделей и дает AnyLogic огромную гибкость и выразительность, что одновременно является преградой для разработчиков, не владеющих этим языком.

В AnyLogic существует возможность создания моделей архитектуры, что позволяет интегрировать их с офисным и корпоративным ПО, включая электронные таблицы, БД, ERP и CRM системы и модулями, написанными на других языках.

В системе AnyLogic введено понятие активного объекта – расширение классических объектов, которые используются в объектно-ориентированных языках программирования. Как и обычные объекты, активные объекты могут иметь свойства и методы. В AnyLogic активный объект может включать также другие элементы:

– параметры – особые свойства, определяющие параметры модели;

– внешние переменные – используются для связи двух активных объектов, когда изменение переменной в одном объекте автоматически меняет ее значение в другом;

– порты – используются для обмена сообщениями между активными объектами;

– стейтчарты (диаграммы состояний) – задают переходы между состояниями активных объектов;

– таймеры – позволяют совершать периодические действия, задавать временные задержки;

– математические функции – задают функцию с использованием математических формул.

Необходимое аппаратное обеспечение: Windows 2000, Windows XP, SP4, Pentium 4, 512MB RAM.

Ценовая политика AnyLogic представлена в следующей таблице:

Наименование	1 лицензия	2 лицензии	3 лицензии
AnyLogic 5.x.x Advanced	139 900 р.	214 700 р.	289 000 р.
Сервис поддержки (один год)	49 500 р.	69 900 р.	92 800 р.

Широкое распространение и активное использование AnyLogic в России, прежде всего, ограничено ценовой политикой компании- разработчика и уровнем использования моделирования в общем.

1.4.11. Witness

Witness – программный продукт для моделирования производственных систем, разработанный компанией Lanner (США).

Witness – одна из ведущих систем моделирования бизнес-процессов, которая дает возможность гибкого моделирования рабочей среды, а также моделирования последствий различных хозяйственных решений и понимание любых процессов, сколько бы сложными они не были.

Witness применяется для:

– анализа входных данных и результатов экспериментальных данных;

– для выявления правил и структуры данных;

– для повышения точности моделей.

Witness способен изучить данные используемые в модели, с целью выявления тенденции и сопоставления данных, а также обеспечивает возможность определения фундаментальных связей, которые могут повысить уровень, принимаемых управленческих решений.

Данный пакет поддерживает блочное графическое моделирование. Witness включает более 50 стандартных блоков. Основные блоки Witness: детали, станки, буфера, работы.

В последних версиях Witness добавлен новый компонент модуля, который позволяет разрабатывать собственный код в Witness- модели, что позволяет не только обеспечить связь с запланированными COM-библиотеками, но и автоматически создать функции в Witness для доступа к необходимым библиотекам.

Witness позволяет поддерживать связь баз данных (Oracle, SQL Server, Access и т. п.), имеет прямой доступ со всеми электронными таблицами, за исключением форматов сообщений – XML, HTML. Полностью интегрирован с 3D/VR views или Post Processed VR. Поддерживает связь с Microsoft Visio, обеспечивает спектр прямых графических решений CAD и многое другое.

Также Witness позволяет создавать отчеты (документацию) созданных процессов.

На российском рынке средств моделирования, и среди бизнес-аналитиков, этот программный продукт не распространен, возможно, это связано с ограниченностью и недостатком информации об этом средстве и тем, что нет дистрибьютеров Witness в России, нет образовательных программ, позволяющих внедрить этот программный продукт в учебный процесс и т. п.

1.4.12. Arena

Arena, программный продукт, разработанный компанией Systems Modeling Corporation (США), предназначен для имитационного моделирования, позволяет создавать подвижные компьютерные модели, используя которые можно адекватно представить очень многие реальные системы. Первая версия системы была разработана в 1993 г.

Arena снабжена удобным объектно-ориентированным интерфейсом и обладает удивительными возможностями по адаптации к всевозможным предметным областям. Система не требует написания программного кода и исключительно проста в использовании, но требует

значительного времени для освоения и достаточно глубоких знаний теории вероятностей, мат. статистики, теории систем массового обслуживания и сетей Петри.

Для отображения результатов моделирования используется анимационная система Cinema animation. Интерфейс Arena включает в себя всевозможные средства для работы с данными, в том числе электронные таблицы, базы данных, ODBC, OLE, поддержку формата DXF.

Система имитационного моделирования Arena включает:

– двумерный графический редактор;

– трехмерный графический редактор (пакет 3D player);

– редакторы временных шаблонов и расписаний;

– редактор символов и библиотека графических заготовок;

– связь с библиотекой графических заготовок и буфером обмена Microsoft.

Область применения системы имитационного моделирования Arena:

– службы работы с клиентами, управление внутренними бизнес-процессами, такими как выполнение заказов, обслуживание или управление простыми производственными потоками;

– сложные крупномасштабные проекты, отличающиеся высокой чувствительностью к изменениям в системах управления логистическими цепочками, производственными процессами, логистикой, сбытом, складированием и системами обслуживания. Возможность создания специализированных шаблонов для сложной, повторяющейся логики, что позволяет упростить процессы и снизить время разработки моделей;

– оперативные и стратегические вопросы разработки упаковочных линий, такие, как инвестиции в новое оборудование, разработка чувствительной логики и конвейерной обработки, а также промышленные процессы крупносерийного производства в смешанных дискретно-непрерывных системах;

– разработка стратегий работы с клиентами, таких как переход на новые сценарии обработки вызовов, виртуальные справочные службы, переадресация на основе практического опыта и моделирование кадрового обеспечения;

– оперативные и стратегические вопросы разработки упаковочных линий, например, инвестиции в новое оборудование, разработка чувствительной логики и конвейерной обработки;

– разработка стратегий работы с клиентами, таких как переход на контакты в электронной форме, виртуальные центры обработки вызовов, переадресация на основе практического опыта и моделирование кадрового обеспечения;

– эффективное средство пост-обработки, обеспечивающее возможность создания и просмотра трехмерных анимаций существующих моделей Arena;

– инструмент оптимизации задач, предназначенный и специально настроенный на анализ результатов моделирования, выполненного с помощью пакета Arena;

– распространение моделей Arena для просмотра и проведения экспериментов.

Arena позволяет использовать дискретное, непрерывное, а также совмещенное дискретно-непрерывное моделирование.

Данный программный продукт поддерживает возможность взаимодействия с пакетом VBA Visual Basic for Applications корпорации Microsoft; объектной моделью ActiveX для внешнего управления, доступом через ADO/ODBC к базам данных (Oracle, Access, Excel, SQL); поддерживает импортирование файлов из пакета AutoCad (в формате dxf); импортирование данных из пакета Visio; импортирование данных из пакета Blue Pumpkin Workforce, коммуникация между отдельными процессами.

Arena включает 200 учебных пособий и библиотек SMART (в зависимости от версии), электронные руководства и универсальные встроенные справочные системы, базы знаний с web-интерфейсом.

При моделировании процессов и систем в Arena, используются три строительные панели:

1. *Basic Process Panel* – модули панели являются фундаментом для создаваемых моделей. Объекты данной панели инструментов состоят из графических модулей: Create, Dispose, Process, Deeside, Separate, Batch, Assign, Record и модулей данных: Entity, Resource, Queue, Variable, Schedule, и Set.

2. Advanced Process Panel – панель усовершенствованных процессов, позволяет моделировать более сложные процессы. Если в Basic Process Panel один модуль может быть наделен несколькими внутренними свойствами, то в Advanced Process эти свойства вынесены как самостоятельные графические модули. Панель состоит из 13 графических модулей (Flowchart Modules): Delay, Dropoff, Hold, Match, Pickup, ReadWrite, Release, Remove, Seize, Search, Signal, Store, Unstore и семи модулей данных (Data Modules): Advanced Set Module, Expression, Failure, File, StateSet, Statistic и Storage.

3. Advanced Transfer Panel – панель процессов передачи, перемещения, содержит 17 графических модулей: Enter Module, Leave Module, Pick Station, Route, Station; Access, Convey, Exit, Start, Stop; Activate, Allocate, Free, Halt, Move, Request, Transport и пять модулей данных: Sequence, Conveyor, Segment, Transporter, Distance.

На данный момент ценовая стоимость различных версий системы имитационного моделирования Arena изменяется в пределах: $ 955 – $ 22 200.

В курсах «Компьютерное моделирование», «Моделирование и анализ сложных систем» и «Консалтинг при автоматизации предприятий» для обеспечения учебного процесса IT-специальностей факультета автоматики и вычислительной техники Томского политехнического университета для обучения студентов моделированию и анализу процессов и систем используются:

1. Пакет имитационного моделирования Arena 7.0, а именно, его версия, которая может быть использована в учебном процессе для образовательных, а не коммерческих целей. Этот программный пакет является современным средством моделирования высокого уровня, позволяющий создавать имитационные модели со сложной логикой.

Это программное средство в настоящее время только начинает использоваться в России, но его успешная апробация прошла за рубежом на ряде крупных предприятий, в таких областях как машиностроительная отрасль, фармация, авиа- и кораблестроение, промышленные производства, оборонная промышленность и т.п.

Arena имеет дружественный пользователю интерфейс, широкую панель моделирования и отчетов по результатам моделирования, специальные встроенные средства оптимизации, анализа входных и выходных данных.

Более подробно методология, заложенная в основу этого программного пакета, будет рассмотрена в третьей главе.

2. Программное средство статического моделирования процессов и систем AllFusion Process Modeler, как универсальный инструмент моделирования, который наиболее часто используется бизнес-аналитиками при выполнении широкого спектра проектов с использованием трех наиболее распространенных методологий: IDEF0, DFD и IDEF3. Более подробно эти три методологии, представляющие основу этого программного пакета, будут рассмотрены во второй главе.

3. Программный пакет ARIS Toolset, его демо-версия используется в учебном процессе, как одно из молодых, но очень перспективных средств моделирования и анализа бизнес-процессов в контексте проектов внедрения ERP-систем и других управленческих информационных систем. Более подробно концепция ARIS будет рассмотрена во второй главе.

1.5. Вопросы к главе 1

1. Что такое модель, и как Вы понимаете процесс моделирования?
2. Для чего и почему проводят моделирование реальных систем?
3. Приведите примеры различных классификаций моделей и назовите параметры этой классификации.
4. Расскажите о классификации математических моделей.
5. Перечислите и опишите основные этапы процесса моделирования.
6. Что такое «модельное время»? Какие механизмы изменения модельного времени существуют?

2. Методологии и средства структурного моделирования процессов и систем

2.1. SADT-методология

В настоящее время много написано и сказано о методологии SADT (Structured Analysis and Design Technique – методология структурного анализа и проектирования) [3, 7, 8, 18, 22, 26, 27], но, несмотря на это, до сих пор существуют различные ее трактовки. Мы будем придерживаться следующей.

Методология SADT – это совокупность методов, правил и процедур, предназначенных для построения моделей объекта предметной области.

SADT-методология является основой семейства методологий моделирования IDEF. Семейство IDEF (ICAM Definition – определение основных терминов программы ICAM) появилось в США в рамках правительственной программы ICAM (Integrated Computer Aid of Manufactory – интегрированная компьютерная помощь производству).

Методология SADT была разработана и предложена Дугласом Россом в конце 60-х годов. В эти годы большинство специалистов работало над созданием программного обеспечения, но немногие старались разрешить более сложную задачу создания крупномасштабных систем, включающих как людей и машины, так и программное обеспечение, аналогичных системам, применяемым в телефонной связи, промышленности, управлении и контроле за вооружением [18]. Методы, такие как SADT, на начальных этапах создания системы позволяют гораздо лучше понять рассматриваемую проблему, и это сокращает затраты как на создание, так и на эксплуатацию системы, а кроме того, повышает ее надежность. Таким образом, SADT – это способ уменьшить количество дорогостоящих ошибок за счет структуризации на ранних этапах создания системы, улучшения контактов между пользователями и разработчиками и сглаживания перехода от анализа к проектированию [18].

В настоящее время семейство IDEF представляет собой IDEF0, IDEF1, IDEF2, ..., IDEF16. В рамках этого учебного пособия и лекционных курсов, проводимых автором, будут рассмотрены две наиболее распространенные методологии моделирования:

1. Методология функционального моделирования IDEF0.
2. Методология событийного моделирования IDEF3.

2.1.1. Методология функционального моделирования IDEF0

За счет своей универсальности, строгости и простоты в настоящее время IDEF0-модели получили широкое распространение и используются:

1) при создании систем менеджмента качества (СМК) на предприятии. Процесс разработки СМК включает в себя разработку документированных процедур, которые представляют собой статическое описание процессов в виде IDEF0-моделей;

2) при проведении обследования деятельности предприятия. Обследование является важнейшим и определяющим этапом консалтинговых проектов, при которых осуществляется построение и анализ моделей деятельности предприятия двух типов: «как есть» и «как должно быть», отображающих текущее и целевое состояние предприятия;

3) при реинжиниринге, включающем изменение технологий целевой и текущей деятельности предприятия, операций учета, планирования, управления и контроля; построение рациональных технологий работы предприятия с учетом существующих автоматизированных систем; создание перспективной оргштатной структуры предприятия, осуществляющей реализацию рациональных технологий работы; изменение информационных потоков и документооборота, обеспечивающих реализацию рациональных технологий работы; разработку проектов схем внутреннего и внешнего документооборота, проекта положения о документообороте, проекта альбома форм входных и выходных документов;

4) при выборе критериев для внедрения корпоративных информационных систем (КИС);

5) при разработке и внедрении новых информационных систем (ИС);

6) при выборе программного обеспечения, автоматизирующего полностью или частично деятельность предприятия (например, системы электронного документооборота);

7) при стратегическом и оперативном планировании деятельности предприятия.

В основе IDEF0-методологии заложена следующая концепция [18]:

1. *Блочное моделирование и его графическое представление*. Графика блоков и дуг SADT-диаграммы отображает функцию в виде блока, а интерфейсы входа/выхода представляются дугами, соответственно входящими в блок и выходящими из него. Взаимодействие блоков друг с другом описывается посредством интерфейсных дуг, выражающих ограничения, которые, в свою очередь, определяют, когда и каким образом функции выполняются и управляются.

2. *Лаконичность и точность.* Выполнение правил SADT требует лаконичности и точности разрабатываемой документации и именования структурных элементов (блоков и стрелок), не накладывая в то же время чрезмерных ограничений на действия аналитика.

3. *Передача информации.* SADT-модель обычно является одной из первых стадий разработки проекта, затем модель передается для дальнейшей работы. Таким образом, модель должна быть разработана так, чтобы в дальнейшем с ней могли работать и понимать, что в нее заложено.

4. *Строгость и формализм.* Разработка моделей требует соблюдения строгих формальных правил, обеспечивающих преимущества методологии в отношении однозначности и целостности сложных многоуровневых моделей.

5. *Итеративное моделирование.* Разработка модели представляет собой пошаговую, итеративную процедуру. На каждом шаге итерации аналитик предлагает эксперту вариант модели, который подвергают обсуждению, рецензированию и редактированию.

6. *Отделение «организации» от «функций».* Исключение влияния организационной структуры на функциональную модель.

2.1.1.1. Основные понятия и состав IDEF0-модели

Состав и изображение IDEF0-модели приведены на рис. 2.1.

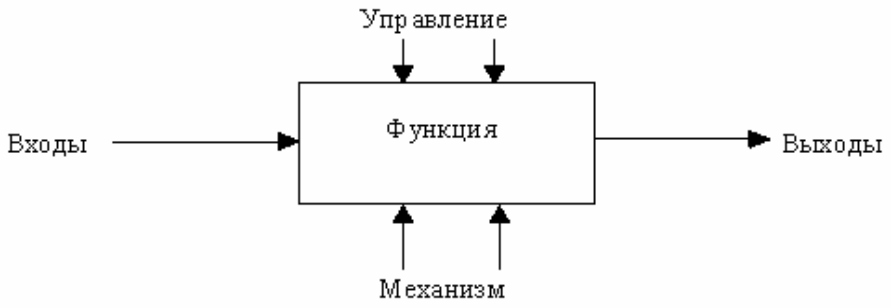


Рис. 2.1. Состав IDEF0-модели

Исходя из названия и информационного наполнения, основным структурным элементом IDEF0-методологии является ***функция***, которая определяет процессы, действия, операции. Имя функции задается глаголом (например, определить стоимость, выполнить операцию). Второй структурный элемент IDEF0-методологии – это стрелка. Стрелки бывают пяти видов:

– входная стрелка, которая показывает то, что необходимо для выполнения функции (детали, заказы);

– выходная стрелка, которая является результатом выполнения функции (прибыль, готовая продукция);

– стрелка-механизм – это то, с помощью кого или чего выполняется функция (сотрудники, оборудование);

– стрелка-управление, которая регламентирует выполнение функции (устав, ГОСТы);

– стрелка-вызов представляет собой техническую стрелку, которая необходима для слияния/расщепления моделей, не несет информативной нагрузки.

Все стрелки (кроме стрелки-вызова) могут быть классифицированы на два вида: внутренние и граничные стрелки (рис. 2.2).

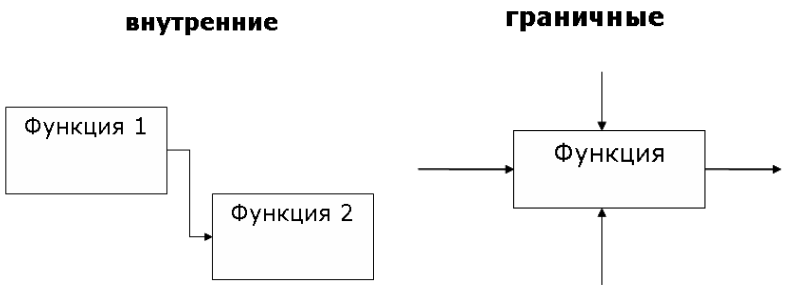


Рис. 2.2. Внутренние и граничные стрелки

В общем виде IDEF0-модель представляет собой набор согласованных диаграмм, фрагмента текста и глоссария (словаря данных). Диаграмма – часть модели, состоящая из взаимосвязанных блоков. Существует специальный вид диаграммы, который называется контекстной диаграммой. Контекстная диаграмма – это диаграмма самого верхнего уровня (уровень А-0), представляющая систему в общем, в виде «черного ящика», и связывающая ее с внешним миром с помощью интерфейсных дуг. Контекстная диаграмма состоит из одного функционального блока, любого количества стрелок, цели моделирования и точки зрения. Пример контекстной диаграммы приведен на рис. 2.3. Цель моделирования указывает, для чего разрабатывается конкретная модель. Точка зрения определяет должностное лицо или подразделение организации, с точки зрения кого разрабатывается модель.

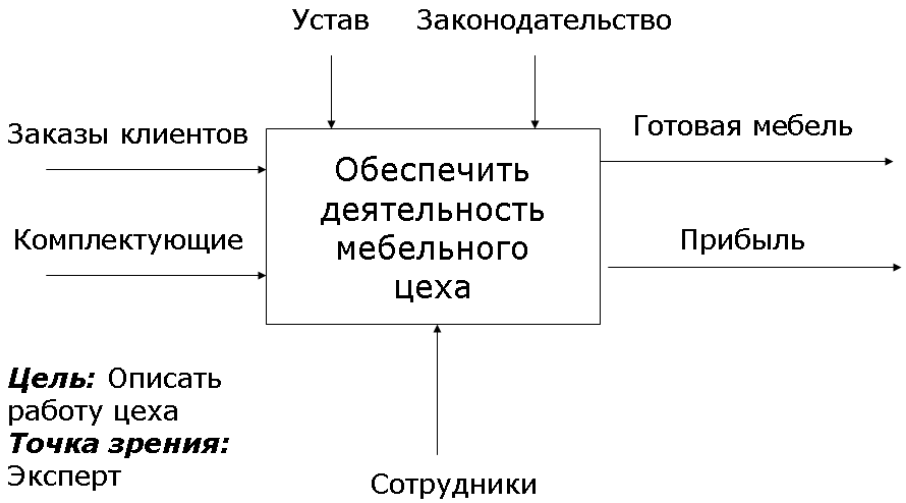


Рис. 2.3. Пример контекстной диаграммы

После разработки контекстной диаграммы проводят процесс декомпозиции. Декомпозиция – это разбиение функции на подфункции, т. е. более детальное ее представление. Говоря о декомпозиции, следует упомянуть об ICOM-кодогенерации (Input, Control, Output, Mechanism), которая позволяет сохранить целостность модели. На практике ICOM-кодогенерация – это процесс, который автоматически переносит стрелки, присоединенные к функциональному блоку, на диаграммы декомпозиции (диаграммы-потомки). Таким образом поддерживается связь между диаграммами-родителями и диаграммами-потомками, сохраняется целостность модели.

Для схожих целей в IDEF0-модели существует понятие туннелирования, или туннельной стрелки. Туннельная стрелка – это специальный вид стрелки (это может быть вход, выход, механизм или управление), которая на модели отображается в виде круглых или квадратных скобок. Квадратные скобки предупреждают разработчика, что в модели появилась ошибка. Квадратные скобки необходимо либо совсем убрать, либо заменить на круглые. Круглые скобки у блока или границы означают, что стрелка является туннельной. Туннель у границы показывает, что этой стрелки нет на диаграмме-родителе, т. е. на верхнем уровне декомпозиции эта стрелка неважна. Туннель у блока говорит о том, что эта стрелка не важна на диаграмме-потомке, и там она не отобразится. Пример туннельных стрелок приведен на рис. 2.4.

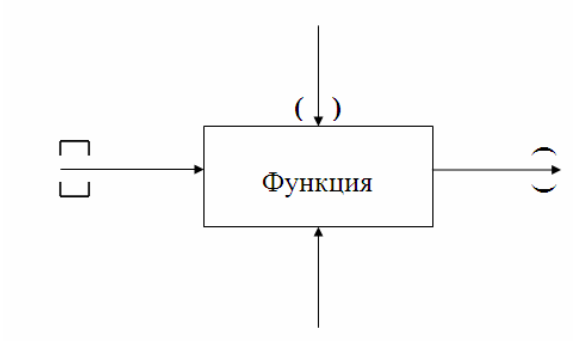


Рис. 2.4. Пример туннельных стрелок

В IDEF0-модели также могут быть стрелки ветвления и слияния, существуют правила отображения этих стрелок в модели. Пример стрелок приведен на рис. 2.5 и рис. 2.6 (*а* – неверный способ отображения, *б* – верный способ отображения).

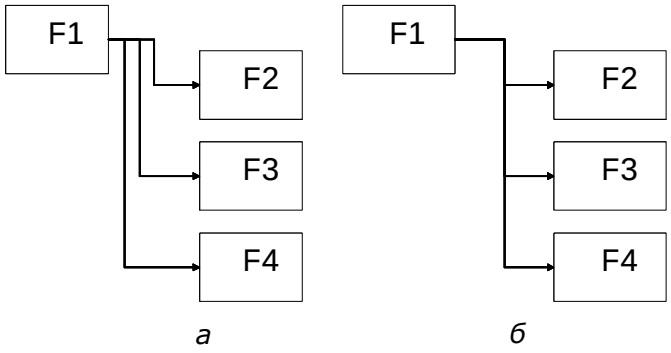


Рис. 2.5. Способ отображения стрелок ветвления:
а – неверный способ отображения; *б* – верный способ отображения

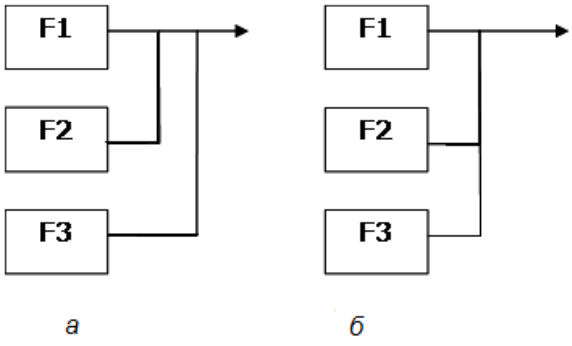


Рис. 2.6. Способ отображения стрелок слияния:
а – неверный способ отображения; *б* – верный способ отображения

Нумерация блоков в IDEF0-модели представляет собой отображение префикса (чаще всего используют А) и описание всей иерархии (рис. 2.7).

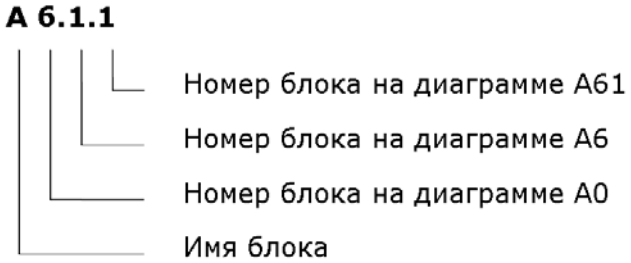


Рис. 2.7. Пример нумерации блока в модели

Между функциями в модели существуют определенные типы отношений:

- ***доминирование***. Это один из распространенных типов отношений между функциями. Отношение доминирования имеет два возможных значения: во-первых, блоки, расположенные выше, более важны и доминантны в рамках рассматриваемой предметной области, во-вторых, блоки, расположенные выше, выполняются раньше по времени, например, если рассмотреть начальную стадию работы промышленного предприятия, то первой функцией будет проведение маркетинговых исследований, затем проектные работы, после чего закупка материалов, производство и продажа готовой продукции (рис. 2.8);

Рис. 2.8. Отношение *доминирования*

- ***управление***. Этот тип отношения используется довольно редко, т. к. результат первой функции – это управляющее воздействие для других функций. В качестве примера можно привести следующее: первая функция – «разработать учебно-методические указания», выход функции – «учебно-методические указания», тогда вторая функция – «выполнить лабораторную работу». Пример приведен на рис. 2.9;

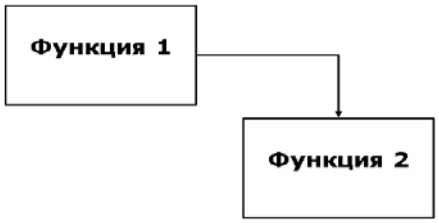


Рис. 2.9. Отношение *управления*

- ***выход–вход***. Самый применяемый тип отношения, когда выход одной функции является входом для другой (рис. 2.10);

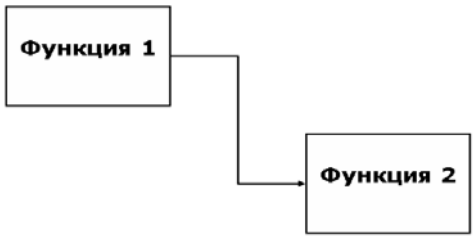


Рис. 2.10. Отношение *выход–вход*

- ***обратная связь (ОС) по управлению***. Выход одной функции является управлением для другой, схож с отношением управления (см. рис. 2.11);

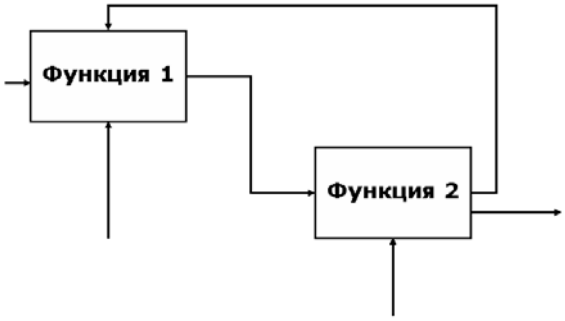


Рис. 2.11. Отношение *обратная связь по управлению*

- ***ОС по входу***. Является аналогом отношения выход–вход (рис. 2.12);

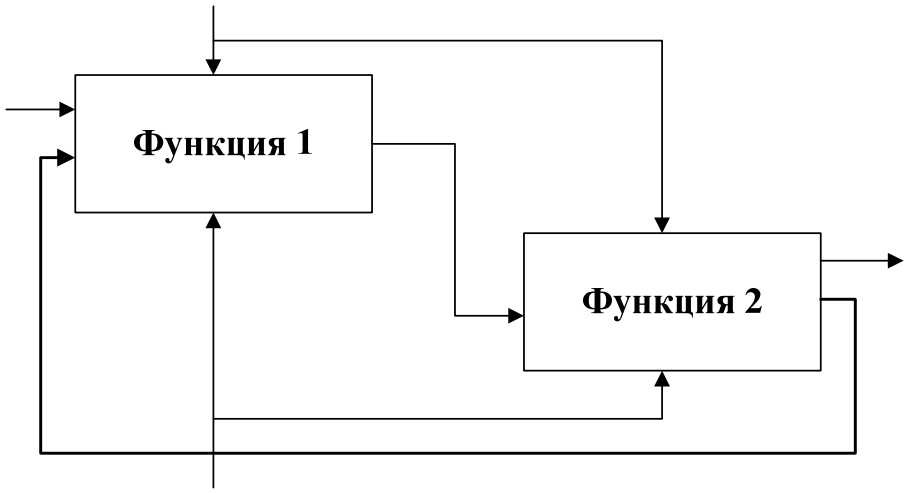


Рис. 2.12. Отношение *обратная связь по входу*

- ***выход–механизм***. Редкий тип отношения, в качестве примера можно привести следующее: предприятие занимается выпуском продукции, а потом в своей дальнейшей деятельности использует это оборудование на других этапах (рис. 2.13).

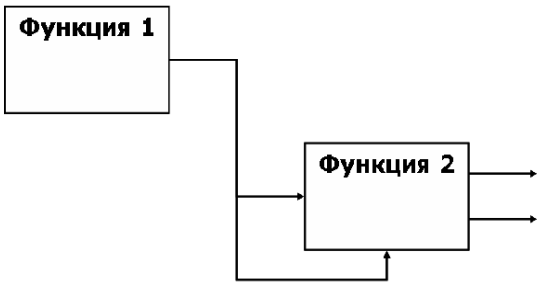


Рис. 2.13. Отношение *выход–механизм*

2.1.1.2. Правила построения диаграмм

1. В состав модели обязательно должна входить контекстная диаграмма уровня А-0.

2. Блоки на диаграмме должны располагаться (предпочтительно) по диагонали (отношение доминирования).

3. Неконтекстные диаграммы должны содержать количество функциональных блоков от 3 до 6. Три функциональных блока определяется тем, что на меньшее количество (два или один) декомпозировать не целесообразно, лучше добавить один или два блока на диаграмме-родителе. Шесть функциональных блоков определено тем, что большее

количество блоков, и соответственно, стрелок, не адекватно воспринимается человеком.

4. Имена функций и стрелок должны быть уникальными. Имена функций должны быть заданы глаголом. Имена стрелок – именем существительным.

5. У любого функционального блока обязательно должна быть хотя бы одна стрелка-управления и одна стрелка-выход. Стрелки-входа может и не быть, но в этом случае, стрелка-управления будет одновременно представлять управляющую и исходную информации. Насчет стрелки-механизма в стандарте функционального моделирования, как в англоязычном, так и в русскоязычном варианте, ничего не сказано, но трудно представить функцию, которая может выполняться автономно без человека или оборудования, исключением являются, например, ядерные реакции.

6. При разработке модели необходимо стремиться к уменьшению количества необязательных пересечений стрелок, минимизировать число петель и поворотов каждой стрелки (рис. 2.14, *а* – неверный способ отображения, рис. 2.14, *б* – верный способ отображения).

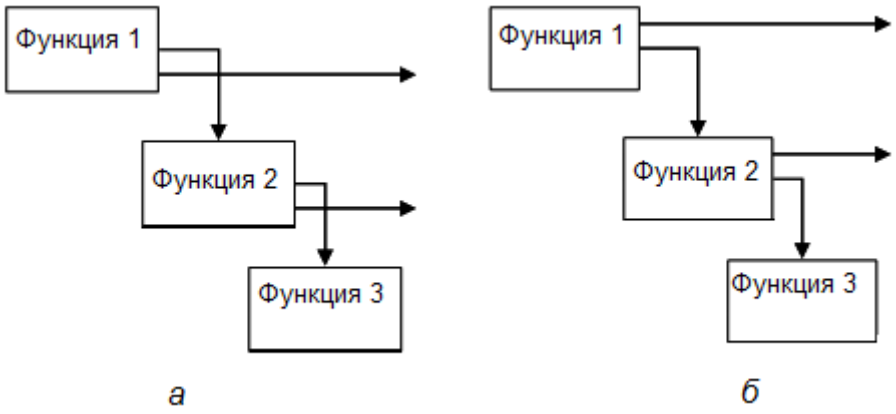


Рис. 2.14. Пример изображения стрелок в модели:
а – неверный способ отображения; *б* – верный способ отображения

7. Стрелки должны объединяться, если имеют общий источник (рис. 2.15, *а* – неверный способ отображения, рис. 2.15, *б* – верный способ отображения).

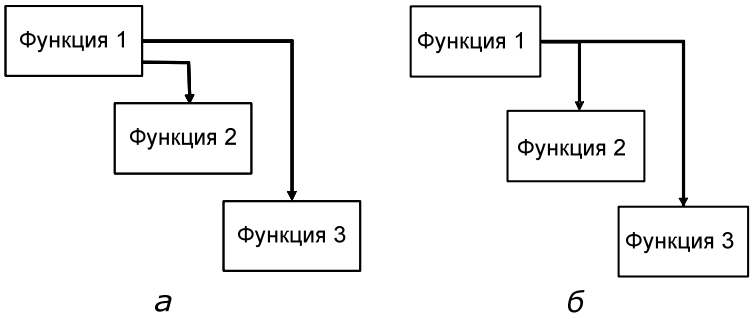
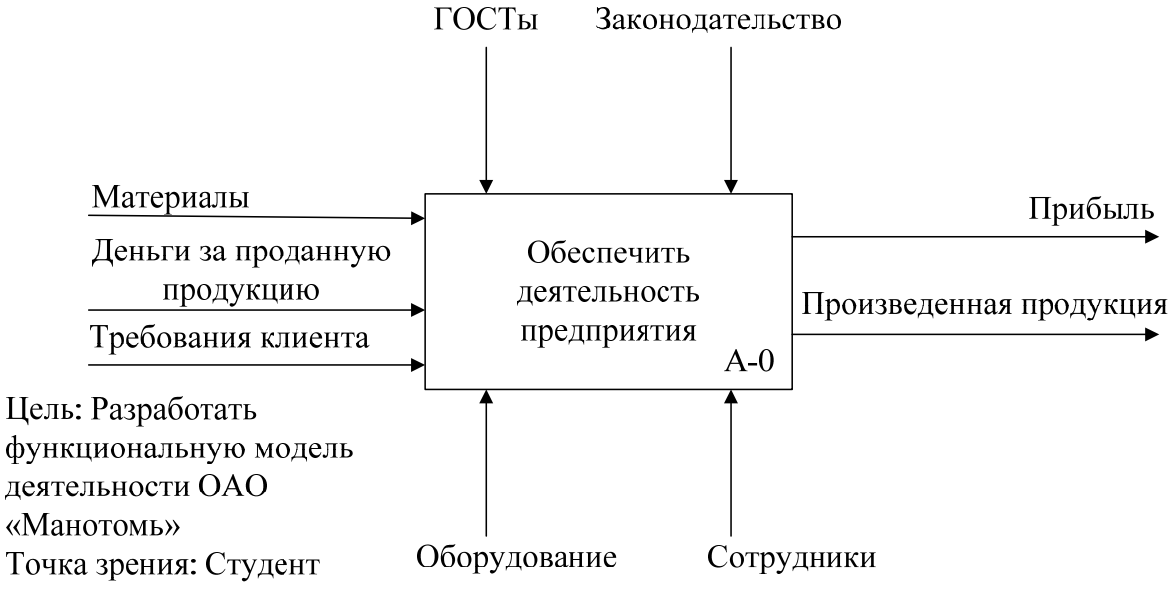
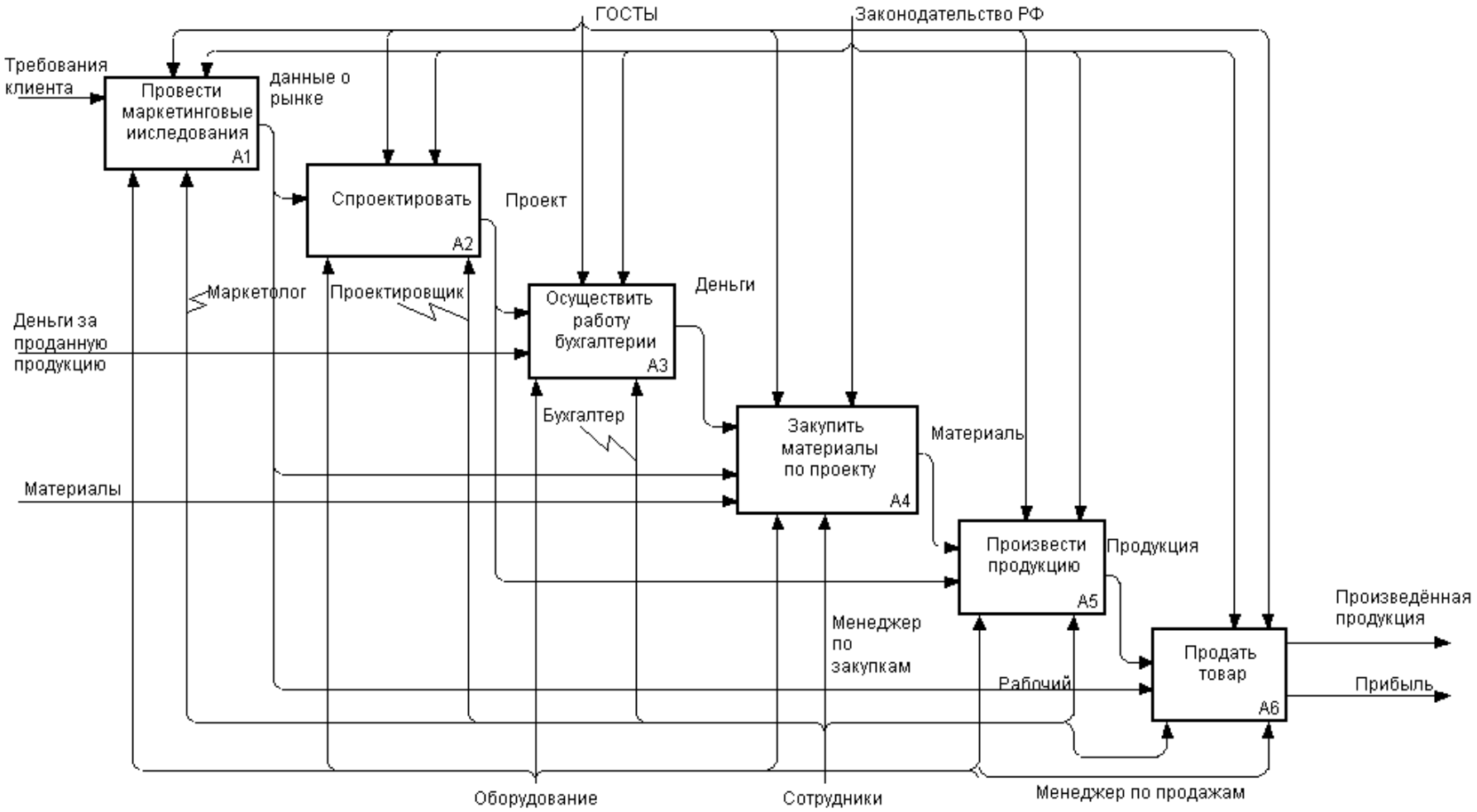


Рис. 2.15. Пример изображения ветвления стрелок в модели:
а – неверный способ отображения; *б* – верный способ отображения

На рис. 2.16 приведен пример IDEF0-модели деятельности промышленного предприятия (*а* – контекстная диаграмма, *б* – диаграмма декомпозиции).



а



б

Рис. 2.16. Пример IDEF0-модели деятельности промышленного предприятия:
а – контекстная диаграмма (уровень А-0);
б – диаграмма основных бизнес-процессов (уровень А0)

2.1.1.3. Глоссарий модели (словарь данных)

Словарь данных – это определенным образом организованный список всех элементов данных системы с их точными определениями, что дает возможность различным категориям пользователей однозначно понимать терминологию предметной области. Словарь данных дает возможность разработчикам, работающим с моделью, иметь общее представление об элементах системы. Также словарь данных содержит информацию, которую не удалось отобразить в модели. В словаре осуществляется определение элементов данных, к которым относятся функции, потоки слияния и ветвления, все виды стрелок и т. д.

Потоки, хранящиеся в словаре, могут быть [7]:

- простыми или групповыми;
- внутренними или внешними;
- потоками данных или потоками управления;
- непрерывными или дискретными.

Атрибуты потока данных [7]:

- имена-синонимы потока данных в соответствии с узлами изменения имени;
- БНФ-определение;
- единицы измерения потока;
- диапазон значений;
- список значений;
- список номеров диаграмм различных типов;
- список потоков;
- комментарий.

2.1.1.4. БНФ-нотация (Бэкуса-Наура форма)

БНФ-нотация позволяет формально описать слияние или ветвление потоков в виде БНФ-спецификации. В БНФ-спецификации существуют стандартные операторы, с помощью которых и происходит детализация и определение потока данных. БНФ-спецификация начинается с символа коммерческой А «@», после которой идет оператор (ИМЯ, ТИП, БНФ, ЕДИНИЦА ИЗМЕРЕНИЯ, НОРМА, КОММЕНТАРИЙ).

Синтаксис БНФ-спецификации:

@БНФ = <простой оператор> ! <БНФ-выражение>;
<простой оператор> – это текстовое описание;
<БНФ-выражение> – это выражение в форме Бэкуса-Наура.

Между выражениями могут использоваться следующие отношения:

= означает «композиция из»;
+ означает логическое «И»;
! означает логическое «ИЛИ»;
«» означает литерал.

Примеры БНФ-спецификаций

Пример 1
@ИМЯ = ВВЕДЕННАЯ КРЕДИТНАЯ КАРТА
@ТИП = управляющий поток
@БНФ = /указывает, что кредитная карта введена/

Пример 2
@ИМЯ = ДАННЫЕ КРЕДИТНОЙ КАРТЫ
@ТИП = поток данных
@БНФ = ПАРОЛЬ + ДЕТАЛИ КЛИЕНТА + ЛИМИТ ДЕНЕГ

Пример 3
@ИМЯ = ДАННЫЕ КЛИЕНТА
@ТИП = поток данных
@БНФ = ФИО + адрес + телефон + ИНН

Пример 4
@ИМЯ = ДЕНЬГИ
@ТИП = дискретный поток
@БНФ = /деньги, выдаваемые клиенту/
@ЕДИНИЦА ИЗМЕРЕНИЯ = доллар
@НОРМА = 5...1000
@КОММЕНТАРИЙ Сумма выдаваемых денег должна делиться на 5

Пример 5
@ИМЯ = СООБЩЕНИЕ
@ТИП = поток данных
@БНФ = e-mail ! факс ! письмо

2.1.1.5. Количественный анализ диаграмм

Для проведения количественного анализа моделей будем использовать следующие показатели [26]:

– количество блоков на диаграмме – N;
– уровень декомпозиции диаграммы – L;
– сбалансированность диаграммы – B;
– число стрелок, соединяющихся с блоком – A.

Данный набор показателей относится к каждой диаграмме в модели, далее используя коэффициенты (формула 1, 2), по которым можно определить количественные характеристики модели в целом.

Для увеличения понятийности модели необходимо стремиться к тому, чтобы количество блоков (N) на диаграммах нижних уровней было меньше, чем количество блоков на родительских диаграммах, то есть с увеличением уровня декомпозиции (L) коэффициент декомпозиции d убывал:

$$d = \frac{N}{L} \qquad (1)$$

Таким образом, убывание этого коэффициента говорит о том, что по мере декомпозиции модели функции должны упрощаться, следовательно, количество блоков должно убывать. Пример графика приведен на рис. 2.17.

Диаграммы должны быть сбалансированы. Это обозначает, что количество стрелок, входящих в блок и выходящих, должно быть равнораспределено, то есть количество стрелок не должно сильно варьироваться. Следует отметить, что данная рекомендация может не соблюдаться для производственных процессов, которые подразумевают получение готового продукта из большого количества составляющих (выпуск узла машины, выпуск продовольственного изделия и другие).

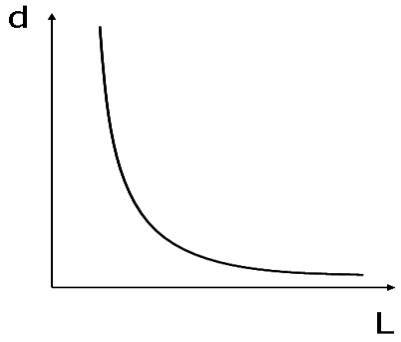


Рис. 2.17. График коэффициента декомпозиции

Коэффициент сбалансированности диаграммы рассчитывается по следующей формуле:

$$K_b = \left| \frac{\sum_{i=1}^{N} A_i}{N} - \max A_i \right| \qquad (2)$$

Желательно, чтобы коэффициент сбалансированности был минимален для диаграммы, а в модели был постоянен (рис. 2.18).

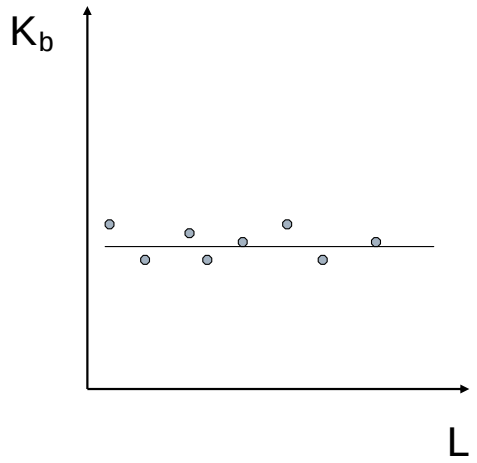


Рис. 2.18. График коэффициента сбалансированности

Кроме оценки качества диаграмм в модели и в целом самой модели по коэффициентам сбалансированности и декомпозиции можно провести анализ и оптимизацию описанных бизнес-процессов. Физический смысл коэффициента сбалансированности определяется количеством стрелок, соединенных с блоком, и соответственно его можно интерпретировать как оценочный коэффициент по количеству обрабатываемых и получаемых конкретным подразделением или сотрудником документов и должностных функций. Таким образом, на графиках зависимости коэффициента сбалансированности от уровня декомпозиции существующие пики относительно среднего значения показывают перегруженность и недогруженность сотрудников на предприятии, так как различные уровни декомпозиции описывают деятельность различных подразделений или сотрудников предприятия. Соответственно, если на графиках реальных бизнес-процессов имеются пики, то аналитик может выдать ряд рекомендаций по оптимизации описанных бизнес-процессов: распределению выполняемых функций, обработке документов и информации, введению дополнительных коэффициентов при оплате труда сотрудников.

2.1.2. Методология событийного моделирования IDEF3

IDEF3-методология менее популярна, чем IDEF0, но в последнее время все чаще встречаются программные продукты, ее реализующие, и сами IDEF3-модели более интересны, т. к. позволяют описать логику процесса за счет введения ряда новых структурных элементов. Практически IDEF3-модели используются:

1) для документирования технологических процессов, где важна последовательность выполнения процесса;

2) описания различных ситуаций (событий) дальнейшего развития процесса с целью прогнозирования (по принципу «что будет, если...»);

3) принятия эффективных управленческих решений при реорганизации процессов [7].

Различают два типа IDEF3-моделей: диаграммы выполнения последовательности этапов (Process Flow Description Diagram) и диаграммы изменения состояний объекта (Object State Transition Network). Отличаются эти диаграммы точкой зрения, которая рассматривается при создании модели. Диаграммы выполнения последовательности этапов разрабатываются с точки зрения стороннего наблюдателя, а диаграммы изменения состояний объекта – с точки зрения самого рассматриваемого объекта. Наиболее часто при моделировании процессов используют диаграммы выполнения последовательности этапов, именно их в дальнейшем мы и будем подразумевать, говоря о IDEF3-моделях.

Выделяют четыре элемента IDEF3-модели.

1. ***Единицы работ*** (Unit of work), которые отображают действия, процессы, события, этапы выполнения работ (рис. 2.19). Имя задается в форме глагола, указывается номер и кто исполняет данную единицу работ.

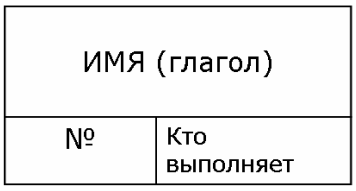


Рис. 2.19. Графическое изображение единицы работ

Говоря об единицах работ, необходимо отметить, что IDEF3-модели являются ***моделями «один вход – один выход»*** («single input – single output»), т. е. у любой единицы работ может быть только один вход и один выход, иначе необходимо вводить дополнительные элементы – перекрестки.

2. ***Ссылки*** (Referents) (см. рис. 2.20) могут выполнять две роли:

– необходимые элементы для выполнения технологического процесса либо результат технологического процесса (металл, компоненты, готовое изделие и т. п.);

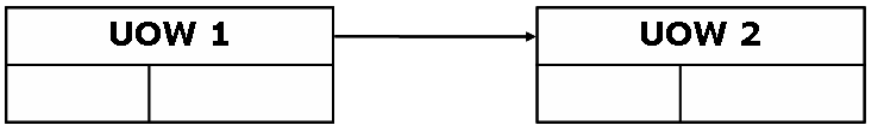
– активаторы процесса (клиент, поставщик и т. п.).

Имя ссылки задается именем существительным.

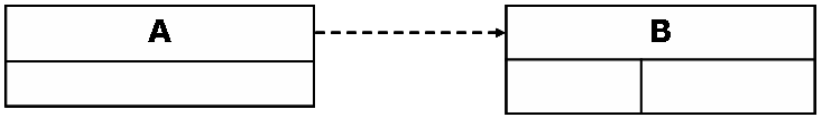
Рис. 2.20. Графическое изображение ссылки

3. ***Связи*** (Links) отображают передачу действия от одной единицы работ к другой либо соединяют ссылку с единицей работ, т. е. активируют единицу работ.

Сплошная стрелка соединяет между собой единицы работ.



Пунктирная стрелка соединяет ссылки с единицами работ.



4. ***Перекрестки*** (Junctions) являются элементами модели, за счет которых описывается логика и последовательность выполнения этапов в модели. Перекресток кардинально отличает IDEF3-модель от других видов моделей, т. к. за счет него описывается событийность модели.

Перекрестки бывают двух видов: перекрестки слияния – Fan In и перекрестки ветвления – Fan Out (рис. 2.21).

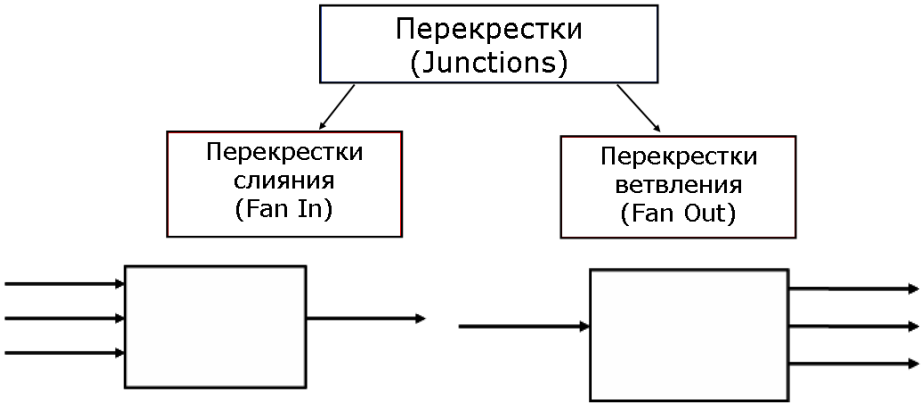


Рис. 2.21. Перекрестки слияния и ветвления

Перекресток не может быть одновременно перекрестком слияния и ветвления (рис. 2.22, *а*), т. к. в этом случае будет неясно правило его срабатывания. Эта ситуация разрешается путем введения в модель каскада перекрестков (рис. 2.22, *б*).

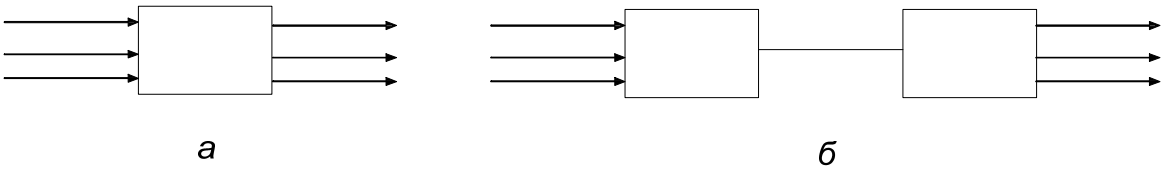
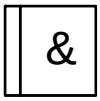
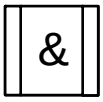


Рис. 2.22. Графическое изображение перекрестка:
а – неверный способ отображения; *б* – верный способ отображения

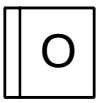
В свою очередь, все перекрестки могут быть пяти типов:

1. 

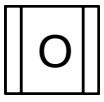
Asynchronous AND (Асинхронное И)

2. 

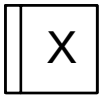
Synchronous AND (Синхронное И)

3. 

Asynchronous OR (Асинхронное ИЛИ)

4. 

Synchronous OR (Синхронное ИЛИ)

5. 

XOR (Exclusive OR) (Исключающее ИЛИ)

Рассмотрим подробно правила срабатывания перекрестков.

Asynchronous AND (Асинхронное И)

Правило срабатывания перекрестка слияния (рис. 2.23, *а*): выходной процесс запустится, если завершились все входные процессы.

Вариантов срабатывания этого перекрестка – **один.**

Правило срабатывания перекрестка ветвления (рис. 2.23, *б*): после завершения входного процесса запустятся все выходные процессы.

Вариантов срабатывания этого перекрестка – **один.**

Пример: после завершения входного процесса «рассчитать клиента» запустятся все выходные процессы «пробить кассовый чек», «принять деньги» и «упаковать покупки», причем эти процессы начнутся не одновременно.

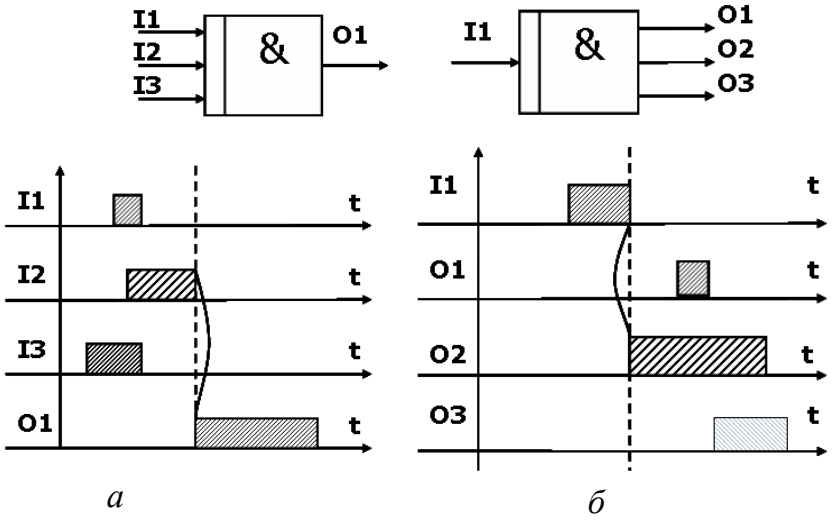


Рис. 2.23. Asynchronous AND (Асинхронное И):
а – перекресток слияния; *б* – перекресток ветвления

Synchronous AND (Синхронное И)

Правило срабатывания перекрестка слияния (рис. 2.24, *а*): выходной процесс запустится, если завершились одновременно все входные процессы. Одновременность не означает, что события произойдут в одну и ту же секунду, это может быть различный по длительности промежуток времени: минута, час, день (зависит от предметной области).

Вариантов срабатывания этого перекрестка – **один.**

Пример: выходной процесс «начать кирпичную кладку» начнется, если выполнены, причем к какому-то определенному моменту времени,все входные процессы «привезти кирпич», «приготовить раствор» и «нанять рабочих».

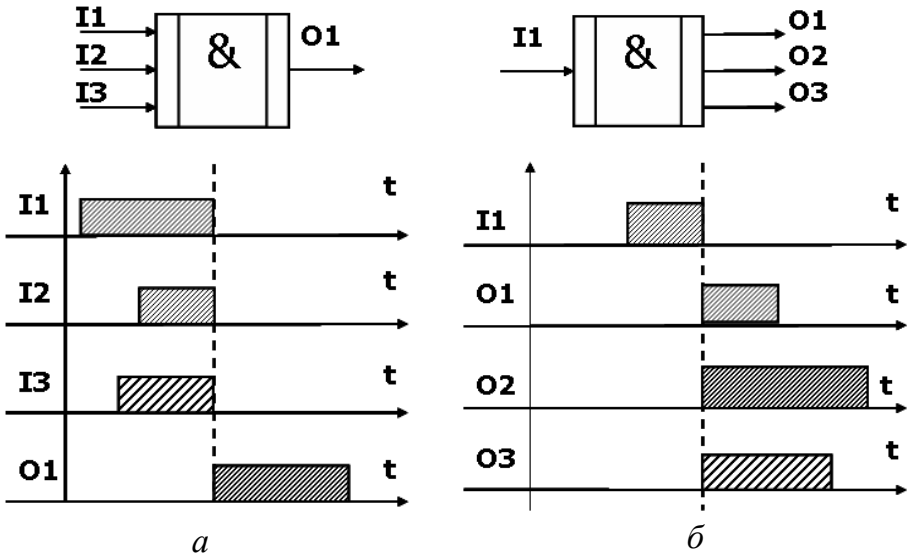


Рис. 2.24. Synchronous AND (Синхронное И):
а – перекресток слияния; *б* – перекресток ветвления

Правило срабатывания перекрестка ветвления (см. рис. 2.24, *б*): после завершения входного процесса запустятся все выходные процессы, причем запустятся одновременно.

Вариантов срабатывания этого перекрестка – **один.**

Asynchronous OR (Асинхронное ИЛИ)

Правило срабатывания перекрестка слияния (см. рис. 2.25, *а*): выходной процесс запустится, если завершится один или любая возможная комбинация входных процессов.

Вариантов срабатывания этого перекрестка будет $\mathbf{2}^{\boldsymbol{N}} \mathbf{- 1}$, где $\boldsymbol{N}$ – количество входов перекрестка.

Правило срабатывания перекрестка ветвления (см. рис. 2.25, *б*): после завершения входного процесса запустятся один или любая возможная комбинация выходных процессов.

Вариантов срабатывания этого перекрестка будет $\mathbf{2}^{\boldsymbol{N}} \mathbf{- 1}$, где $\boldsymbol{N}$ – количество выходов перекрестка.

Пример: после завершения входного процесса «подготовить официальное приглашение зарубежной делегации» может запуститься первый выходной процесс «отправить приглашение обычной почтой» и третий выходной процесс «отправить приглашение факсом», но может не сработать второй выходной процесс «отправить приглашение по e-mail». Либо при срабатывании этого перекрестка может сработать любая другая комбинация выходных процессов.

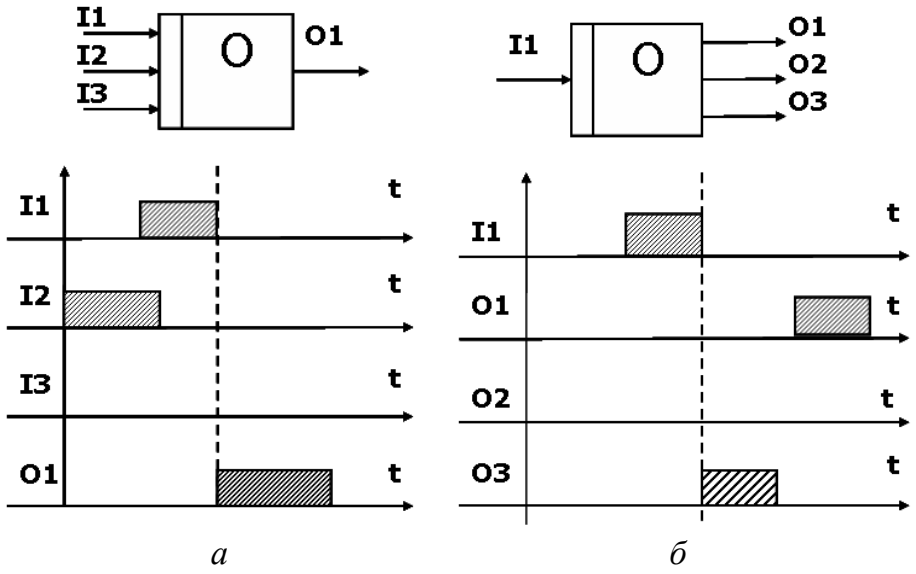


Рис. 2.25. Asynchronous OR (Асинхронное ИЛИ):
а – перекресток слияния; *б* – перекресток ветвления

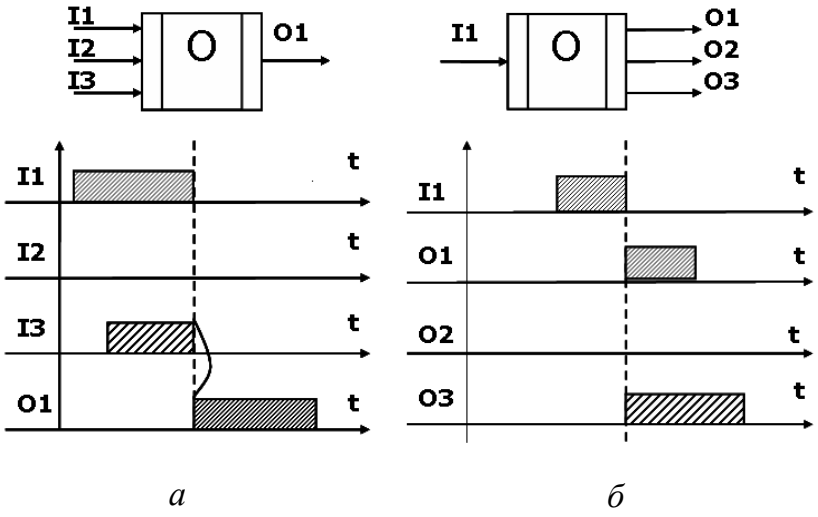
Synchronous OR (Синхронное ИЛИ)

Правило срабатывания перекрестка слияния (см. рис. 2.26, *а*): выходной процесс запустится, если завершится один или любая возможная комбинация входных процессов, но если сработала комбинация процессов, то тогда завершится она должна одновременно.

Вариантов срабатывания этого перекрестка будет $\mathbf{2^N - 1}$, где $\boldsymbol{N}$ – количество входов перекрестка.

Пример: после одновременного завершения входных процессов «заштукатурить стены» и «сделать стяжку» запустится выходной процесс «вставить окна».

Правило срабатывания перекрестка ветвления (рис. 2.26, *б*): после завершения входного процесса запустится один или любая возможная комбинация выходных процессов, но если сработала комбинация процессов, то тогда запуститься она должна одновременно.



а *б*

Рис. 2.26. Synchronous OR (Синхронное ИЛИ):
а – перекресток слияния; *б* – перекресток ветвления

Вариантов срабатывания этого перекрестка будет $\mathbf{2^N - 1}$, где $\boldsymbol{N}$ – количество выходов перекрестка.

Exclusive OR (исключающее ИЛИ)

Правило срабатывания перекрестка слияния (см. рис. 2.27, *а*): выходной процесс запустится, если завершился только один входной процесс.

Вариантов срабатывания этого перекрестка – $\boldsymbol{N}$, где $\boldsymbol{N}$ – количество входов перекрестка.

Правило срабатывания перекрестка ветвления (см. рис. 2.27, *б*): после завершения входного процесса запустится только один выходной процесс.

Вариантов срабатывания этого перекрестка – $\boldsymbol{N}$, где $\boldsymbol{N}$ – количество выходов перекрестка.

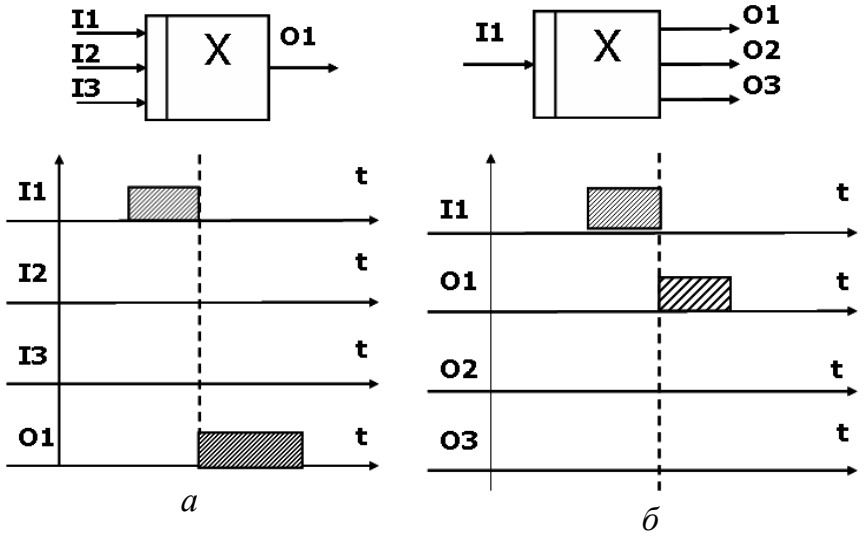


Рис. 2.27. Exclusive OR (исключающее ИЛИ):
а – перекресток слияния; *б* – перекресток ветвления

Пример: после завершения входного процесса «завершить пошив платья» запустится только один, например первый, выходной процесс – «отдать платье заказчику». Хотя возможна и альтернатива, но тогда она исключает первый выход, например «выставить платье на продажу». Невозможно одновременно одно платье «отдать заказчику» и «выставить на продажу».

Пример IDEF3-модели разработки базы данных (БД) информационной системы приведен на рис. 2.28.

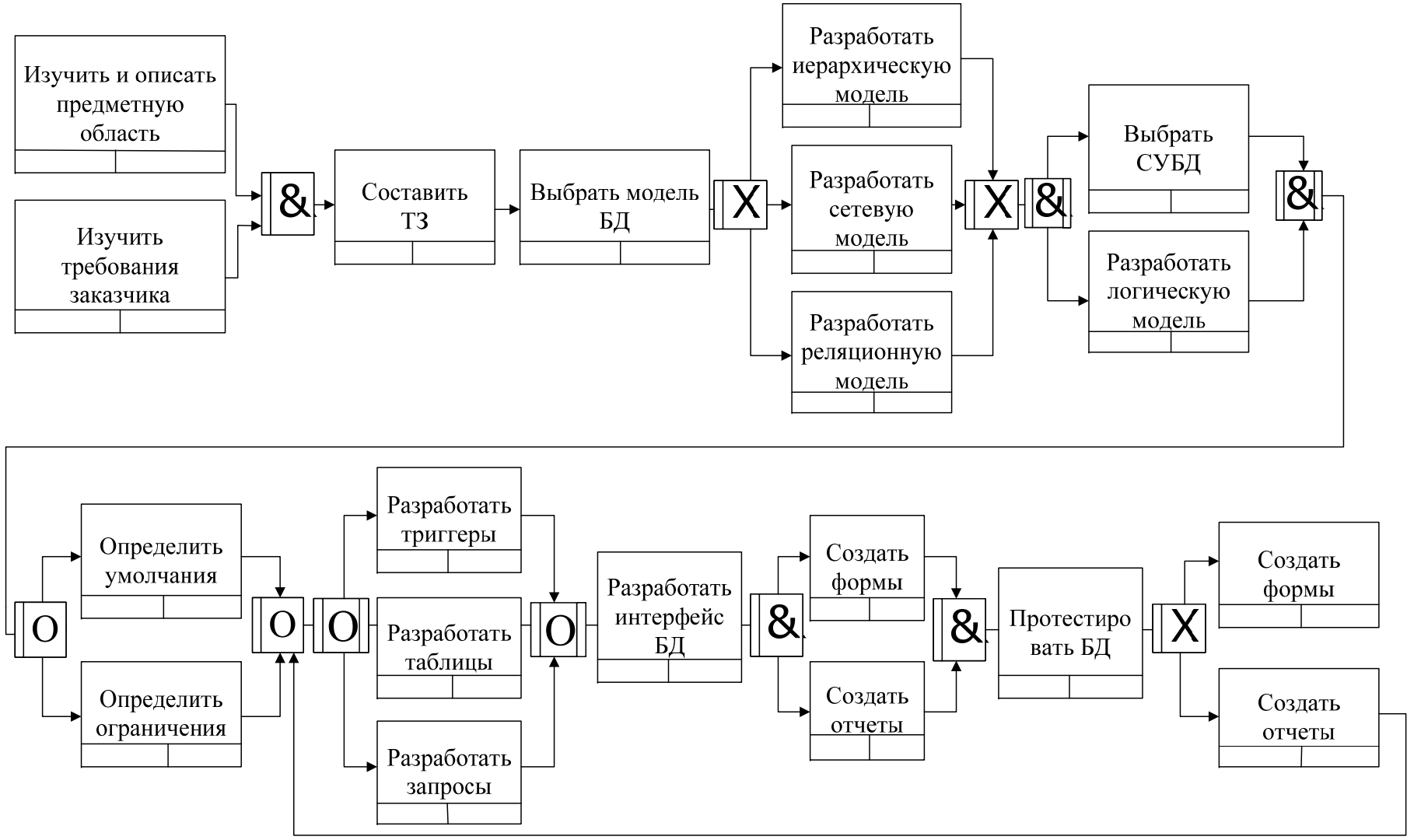


Рис. 2.28. Пример IDEF3-модели разработки базы данных

2.2. Методология моделирования потоков данных (Data Flow Diagram)

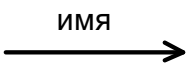
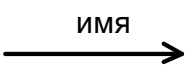
Первоначально диаграммы потоков данных разрабатывались и использовались при проектировании информационных систем. В настоящее время область применения DFD значительно расширилась и их используют [22]:

– при проведении обследования деятельности предприятия;
– при проведении работ по реинжинирингу;
– при анализе и оптимизации бизнес-процессов предприятия;
– при внедрении систем электронного документооборота.

При построении диаграмм потоков данных наиболее часто используют две нотации: Йордана и Гейна-Сарсона [7]. Обе нотации имеют одинаковый по названиям и значению элементный состав, но имеют различное его графическое изображение (табл. 2.1).

Таблица 2.1

Графические элементы DFD

Компонента	Нотация Йодана	Нотация Гейна-Сарсона
поток данных		
процесс		
хранилище		
внешняя сущность		

Всего в DFD используется четыре структурных элемента:

1. ***Процессы.*** Процессы в DFD обозначают функции, операции, действия, которые обрабатывают и изменяют информацию. Процессы показывают, каким образом входные потоки данных преобразуются в выходные.

2. ***Потоки данных***. Потоки данных идут от объекта-источника к объекту-приемнику, обозначая информационные потоки в системе. Взаимодействие работ с внешним миром и между собой описывается в виде стрелок (потоков данных). Поток данных соединяет выход объекта (или процесса) с входом другого объекта (или процесса).

3. ***Внешние сущности***. Внешние сущности определяют элементы вне контекста системы, которые участвуют в процессе обмена информацией с системой, являясь источниками или приемниками информации. Внешние сущности изображают входы в систему и/или выходы из системы. Внешние сущности обычно изображаются на контекстной диаграмме. Внешние сущности представляют собой материальный предмет или физическое лицо, например: *ЗАКАЗЧИК*, *ПЕРСОНАЛ*, *ПОСТАВЩИК*, *КЛИЕНТ*, *СКЛАД*, *БАНК*.

4. ***Хранилища данных***. Хранилища данных представляют собой собственно данные, к которым осуществляется доступ, эти данные также могут быть созданы или изменены процессами. Хранилище данных изображают объекты в покое и данные, которые сохраняются в памяти между последующими процессами. Информация, которую содержит хранилище данных, может использоваться в любое время после её определения. При этом данные могут выбираться в любом порядке.

В диаграммах потоков данных все используемые символы складываются в общую картину, которая дает четкое представление о том, какие данные используются и какие функции выполняются системой документооборота.

Пример DFD-модели, разработанной в нотации Гейна-Сарсона, приведен на рис. 2.29 (контекстная диаграмма) и рис. 2.30 (диаграмма основных бизнес-процессов).

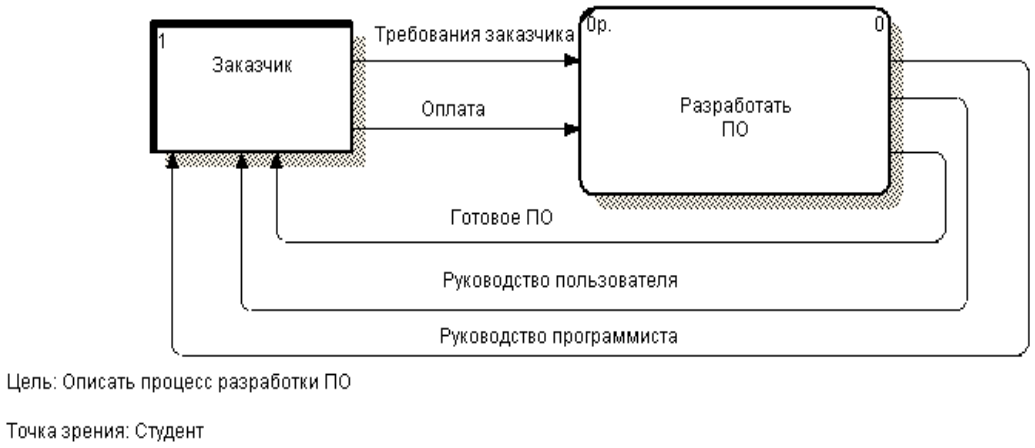


Рис. 2.29. Контекстная диаграмма DFD

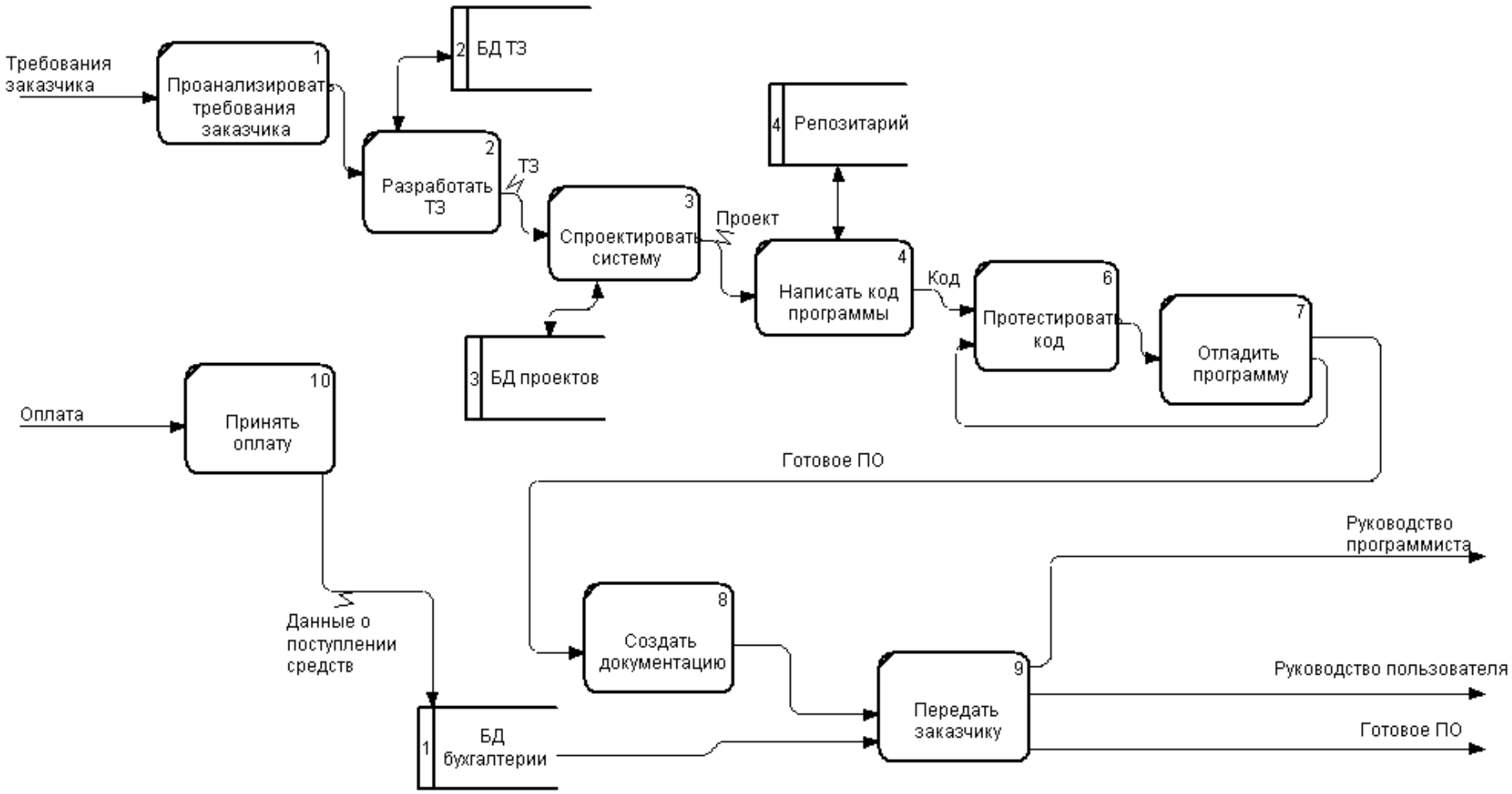


Рис. 2.30. Диаграмма основных бизнес-процессов

2.3. Концепция ARIS

Концепция ARIS (Architecture of Integrated Informational System) была предложена и разработана немецким исследователем, профессором Августом Вильгемом Шеером. Концепция ARIS представляет собой подход к формализации информации о деятельности организации и представление ее в виде графических моделей, удобных для понимания и анализа. Модели, создаваемые по концепции ARIS, отражают существующую ситуацию «как есть» и предполагают построение моделей «как должно быть». Степень детализации описания зависит от целей проекта, в рамках которого проводится моделирование. Модели ARIS могут быть использованы:

1) для анализа и оптимизации существующих бизнес-процессов предприятия. Для решения задач:
   – изменения структуры процесса путем введения одновременно выполняемых задач, что позволяет устранить лишние циклы и сделать структуру более рациональной,
   – изменения структуры организационной отчетности и повышения квалификации сотрудников путем комплексного совершенствования процесса,
   – сокращения объема документации, рационализации и ускорения документооборота и потока данных,
   – рассмотрения возможных мер по привлечению внешних ресурсов (т. е. по передаче функции создания выхода внешнему исполнителю),
   – внедрения новых производственных и ИТ-ресурсов для улучшения функций обработки [35, 36, 37];
2) для выработки различного рода рекомендаций и решений по реорганизации деятельности предприятия;
3) при внедрении информационной системы управления, чаще всего класса MRPII/ERP,
4) для хранения корпоративных знаний, в том числе в виде моделей-прототипов;
5) при создании и постоянном контроле технологической документации для получения сертификата ISO-9000;
6) при исчисление стоимости бизнес-процессов;
7) при проектировании информационных систем.

Концепция ARIS объединяет принципы системного структурного анализа [18, 20] и процессного подхода, позволяет всесторонне отразить в ARIS-модели основные компоненты организации, протекающие процессы, производимую и потребляемую продукцию, используемую информацию, а также выявить взаимосвязи между ними.

Создаваемые модели представляют собой формализованное описание предприятия, включая организационную структуру, протекающие процессы, взаимодействия между организацией и субъектами рынка, состав и структуру документов, последовательность бизнес-процессов, должностные инструк-

ции отделов и их сотрудников. В отличие от других подходов, концепция ARIS предполагает хранение всей информации в едином репозитарии, что обеспечивает целостность и непротиворечивость процесса моделирования и анализа, а также позволяет проводить верификацию моделей.

Достоинствами концепции ARIS являются:

– возможность описания объекта с разных точек зрения: организация, функции, данные и управление;

– широкий набор методов моделирования, отражающих различные аспекты исследуемой предметной области, позволяет моделировать разносторонний спектр систем (организационно-хозяйственных, технологических и прочих);

– все модели и объекты создаются и хранятся в единой базе проекта, что обеспечивает построение интегрированной и целостной модели предметной области;

– возможность многократного применения результатов моделирования путем накопления знаний о различных предметных областях;

– возможность интеграции ARIS-моделей с другими программными продуктами.

Как было сказано выше, в концепции ARIS выделено четыре основных вида моделей (точек зрения):

– организационные модели, описывающие иерархическую структуру системы – иерархию организационных подразделений, должностей, полномочий конкретных лиц, многообразие связей между ними, а также территориальную привязку структурных подразделений;

– функциональные модели, описывающие функции (процессы, операции), выполняемые в организации;

– информационные модели (модели данных), отражающие структуру информации, необходимой для реализации всей совокупности функций системы;

– модели процессов/управления, представляющие комплексный взгляд на реализацию бизнес-процессов в рамках системы и объединяющие вместе другие модели.

Говоря об этих четырех видах моделей, вводят понятие «здания» ARIS, которое интегрирует различные точки зрения в единое целое. Графически «здание» ARIS имеет вид, приведенный на рис. 2.31 [35, 36, 37].

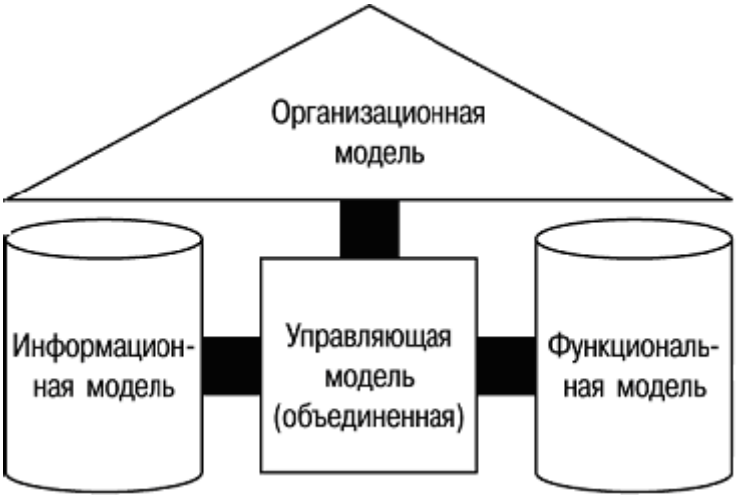


Рис. 2.31. «Здание» ARIS

В рамках каждого типа представления создаются модели, отражающие ту или иную сторону исследуемой системы. Концепция ARIS включает большое количество методов моделирования, в том числе известных как диаграммы Чена ERM, язык UML (Unified Modeling Language), методики OMT (Object Modeling Technique), BSC (Balanced Scorecard) и т.п.

Также в концепции ARIS вводят определение фазовой модели, которая характеризует этапы создания информационных систем и подходы, применяемые к описанию моделей бизнес-процессов. Согласно концепции ARIS, модель организации структурируется в соответствии с концепцией жизненного цикла информационных систем, который представляется в виде последовательности этапов жизненного цикла. Жизненный цикл в концепции ARIS представлен в виде 3 уровней фазовой модели:

1. Определение требований
2. Спецификация проекта
3. Описание реализации

На первом этапе проводят определение требований и анализ проблем предприятия. Модели на этом уровне представляют собой семантические описания бизнес-процессов, однако они достаточно точно отражают цели, которые стоят перед разработчиками. На этом этапе в описание включаются характеристики будущей модели организации, связанные с бизнес-процессами. На уровне определения требований необходимо описать требования к прикладной информационной системе, создаваемой для решения рассматриваемой проблемы предприятия.

Второй этап спецификации проекта описывает пользовательские или модульные транзакции, которые выполняют функции. Это может рассматриваться как отображение сформулированных требований в категории и методы описания, связанные непосредственно с информационными системами и выраженные в терминах соответствующих технологий.

На третьем этапе описания реализации спецификация проекта трансформируется в конкретные аппаратные и программные компоненты. Таким образом, осуществляется физическая связь с информационной системой.

Таким образом, фазовая модель имеет вид, приведенный на рис. 2.32 [35, 36, 37].

Создание различных видов моделей и проработка каждой из них по уровням описания в сочетании с формулировкой проблем предприятия и составляет процесс работы в архитектуре ARIS. Каждый тип представления подвергается разложению на три уровня описания: определение требований, спецификацию проекта и описание реализации.

Концепция ARIS, прежде всего, позволяет зафиксировать широкий спектр описательных аспектов бизнес-процессов, подобрать способы для их рассмотрения, определить возможные наложения методов и выявить пробелы в описании. Преимущества ARIS очевидны как при решении административных и организационных вопросов бизнеса, так и при проектировании компьютеризованных информационных систем [35, 36].

Основные типы моделей

1. Организационная модель (Organizational chart)
2. Модель дерева функций (Function tree)
3. Модель цепочки добавленной стоимости (Value-added chain diagram, VACD)
4. Расширенная событийно-ориентированная модель (extended Event-driven Process Chain, eEPC)
5. Модель описания функций (Function allocation diagram, FAD)
6. Офисная модель (Office Process)
7. Модель промышленного процесса (Industrial process)
8. С3-модель

Рис. 2.32. Фазовая модель

С учетом фазовой модели «здание» ARIS модифицируется (рис. 2.33).

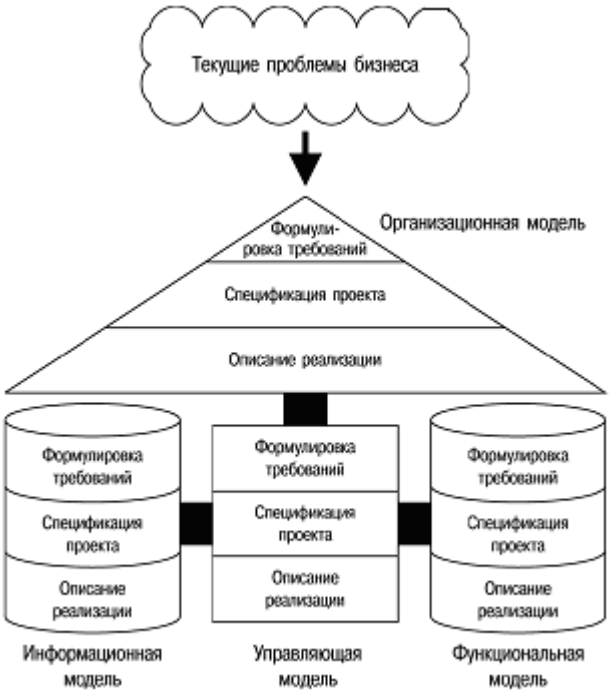


Рис. 2.33. «Здание» ARIS с учетом фазовой модели

2.3.1. Организационная модель (Organizational chart)

Организационная модель обеспечивает описание статических отношений между различными структурными элементами – участниками бизнес-процесса, ответственными за выполнение функций на предприятии.

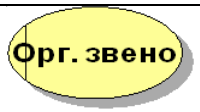
Важной особенностью организационной модели является возможность задать территориальную привязку для каждой организационной единицы и/или его составляющей. Это достигается путем описания иерархии территорий (комплекс, здания, этажи, кабинеты и т.д.) и последующей привязки подразделений к указанным компонентам территории.

Основные элементы организационной модели приведены в табл.2.2.

Организационная модель строится иерархически, от верхнего уровня структуры к нижнему. В модель верхнего уровня включаются самостоятельные подразделения, входящие в структуру организации. Каждое из них детализируется на более низкие уровни – уровни структурных подразделений. Каждое структурное подразделение, в свою очередь, детализируется на структурные подразделения в его составе, которые изображаются на одной диаграмме [37].

Таблица 2.2

Основные элементы организационной модели

Изображение и название элемента	**Описание элемента**
 Организационная единица (Organizational unit)	Обозначает отдельное штатное подразделение (департамент, отдел, сектор)
 Группа (Group)	Может отображать группу сотрудников, работающих вместе для решения конкретной задачи
 Позиция (Position)	Представляет должность в организации
 Личность внутренняя (Internal Person)	Представляет конкретных сотрудников, состоящих в штате предприятия
 Личность внешняя (External Person)	Представляет конкретных сотрудников, не состоящих в штате предприятия
 Место (Location)	Означает физическое место расположения подразделения, сотрудника

Низшим уровнем является описание подразделений на уровне должностей – штатных единиц, занимаемых конкретными сотрудниками. При детализации моделей подразделений до уровня сотрудников целесообразно полностью указывать должность в составе подразделения. В случае, если в одном подразделении имеется несколько одинаковых должностей, то они нумеруются [37].

Допустимые типы отношений

- is composed of – состоит из
- is superior – вышестоящий
- is belong to – является частью
- is located at – расположен
- is assigned to – приписывается к
- is organization manager for – организационно управляет
- occupies – занимает пост
- has member – является членом
- cooperates with – взаимодействует с

В табл. 2.3 приведены возможные и допустимые типы отношений в организационной модели между элементами.

Таблица 2.3

Типы отношений в организационной модели

						
	Is composed of Is superior	Is belong to	Is belong to	Is composed of	Is located at	Is assigned to
	Is organization manager for	Is organization manager for	Is organization manager for	Is organization manager for Occupies	Is located at	Is organization manager for
	Is belong to	Is organization manager for	Is organization manager for	Is organization manager for	Is located at	X
	Is organization manager for	Is organization manager for	Is organization manager for	Is organization manager for	Is organization manager for	Is organization manager for
	Is located at	Is located at	Is located at	Is organization manager for	Subsumes	Is located at
	Is assigned to	Has member	Has member	Is composed of	Is located at	Cooperates with

2.3.2. Модель дерева функций (Function tree)

Модель дерева функций – это граф, показывающий взаимоотношения между функциями. Модель показывает иерархию функций и их полный состав. Основные элементы дерева функций приведены в табл. 2.4, а типы отношений в модели дерева функций в табл. 2.5.

В этом типе модели функции могут быть описаны с различными уровнями детализации. При этом функции представляются не обязательно в хронологическом порядке. На самом верхнем уровне описываются наиболее сложные функции, представляющие собой отдельный бизнес-процесс или процедуру. Детализация функций образует иерархическую структуру их описаний. Разделение функций на элементы может происходить на нескольких иерархических уровнях. Базовые функции представляют самый нижний уровень в семантическом дереве функций. Базовая функция – это функция, которая уже не может быть разделена на составные элементы с целью анализа бизнес-процесса [37].

Таблица 2.4

Основные элементы в модели дерева функций

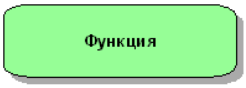
Изображение и название элемента	**Описание элемента**
 Функция (Function)	Показывает функции, выполнение которых направлено на достижение цели

Таблица 2.5

Типы отношений в модели дерева функций

	
	Is process-oriented superior процессно-ориентированное подчинение

2.3.3. Модель цепочки добавленной стоимости (VACD)

Модель цепочки добавленной стоимости показывает, из чего складывается конечная стоимость готового продукта. То есть определяются процессы, добавляющие стоимость продукта. Модель цепочки добавленной стоимости описывает функции организации, которые непосредственно влияют на реальный выход ее продукции. Эти функции создают последовательность действий, формируя добавленные значения: стоимость, количество, качество и т.д. Построение модели цепочки добавленной стоимости целесообразно начинать с обзорного представления взаимосвязанных частей процесса путем расположения элементов процесса согласно временной последовательности их выполнения.

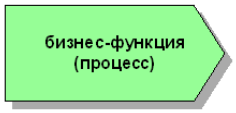
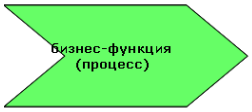
Затем рекомендуется отразить взаимозависимость различных элементов процесса путем нанесения соответствующих связей. После отображения в модели структуры процесса каждый из элементов процесса рассматривается с точки зрения необходимости его детализации. Если это требуется, то элемент детализируется на соответствующие блоки.

Аналогично дереву функций описываемые функции могут размещаться в диаграмме согласно иерархическому принципу, т.е. наиболее важные функции располагаются левее и выше. Эта иерархия всегда иллюстрирует подчинение функций [37].

Чаще всего, VACD – это модель верхнего уровня (эквивалент контекстной диаграммы A-0). Основные элементы модели VACD приведены в табл. 2.6.

Таблица 2.6

Основные элементы VACD

Изображение и название элемента	**Описание элемента**
 Бизнес-функция (процесс)	Начальный элемент модели цепочки добавленной стоимости
 Бизнес-функция (процесс)	Все остальные элементы модели цепочки добавленной стоимости

2.3.4. Расширенная событийно-ориентированная модель (eEPC)

eEPC (extended Event-driven Process Chain)-модель применяется для подробного описания бизнес-логики процесса с использованием четырех групп элементов (рис. 2.34): функциональные элементы, логические элементы, элементы данных и организационные элементы. Графическое изображение элементов приведено в табл. 2.7. При использовании всех групп элементов получается всесторонняя модель бизнес-процесса, показывающая основные действия, выполняемые конкретными сотрудниками с помощью прикладных и технических устройств.

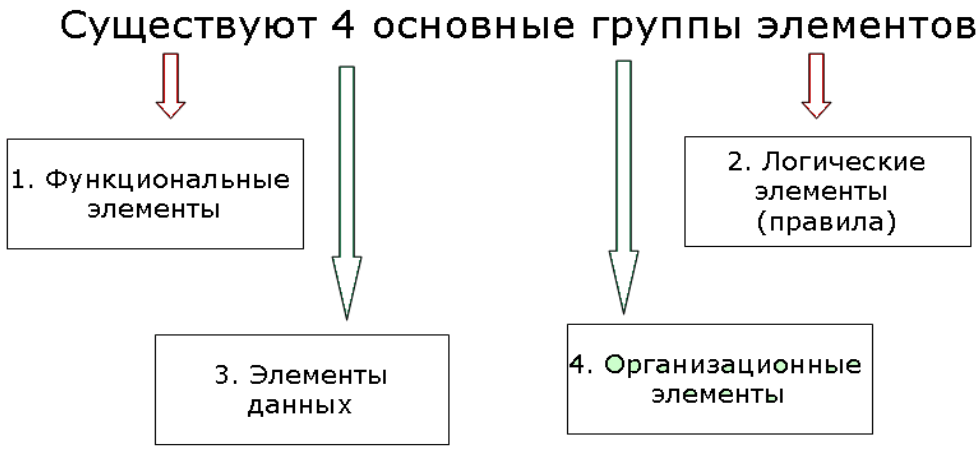
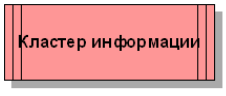
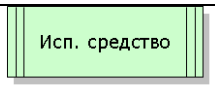
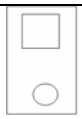
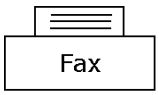
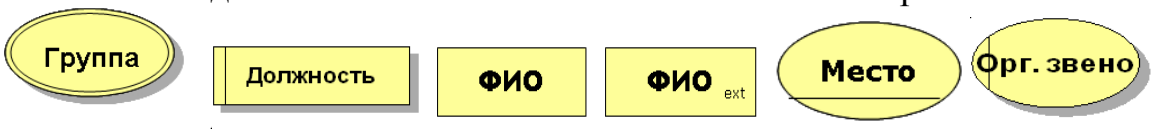


Рис. 2.34. Группы элементов в eEPC-модели

Таблица 2.7

Основные элементы eEPC-модели

Изображение и название элемента	**Целевое использование**	**Правила именования**
Функциональные элементы		
 Событие (Event)	Отображение событий, происходящих при выполнении бизнес-процесса	Имя начинается с имени объекта, состояние или событие по отношению к которому произошло
 Функция (Function)	Описание бизнес-функции в цепочке выполнения бизнес-процесса	Имя начинается с действия или обозначения процесса
Логические элементы		
 Правило (Rules) Исключающее ИЛИ	Правила ветвления или слияния функций или событий	Объекты данного типа не именуются
 Правило (Rules) Логическое И	Правила ветвления или слияния функций или событий	Объекты данного типа не именуются
 Правило (Rules) Логическое ИЛИ	Правила ветвления или слияния функций или событий	Объекты данного типа не именуются
Элементы данных		
 Набор данных (Cluster)	Описание абстрактного (на концептуальном уровне) набора формализованных данных	В имени необходимо упомянуть название документа или источника информации
 Используемое средство (Application System)	Реальное средство или система, автоматизирующая рабочие процессы	Реальное имя средства или системы (Например, 1С: Предприятие, Axapta)

 Документ (Document)	Представление информационного носителя данных в материализованном виде (напр. на бумаге)	Имя должно содержать наименование документа
 База данных (файл) (Database/File)	Представление информационного носителя данных в нематериальной форме (напр. на магнитном диске или флеш-памяти)	Именуется названием файла или именем информационной базы данных
 Папка (Folder)	Указывает вид хранения документов	Имя должно содержать наименование папки с документами
 Телефон (Telephone)	Представление результата человеческих действий, может являться как реальным устройством, так и информацией	Полное наименование
 Факс (Fax)	Представление результата человеческих действий, может являться как реальным устройством, так и информацией с него поступающей	Полное наименование
 Экспертиза (Expertise)	Человек или государственный орган, осуществляющий контролирующие или экспертные функции	Тип эксперта или государственного органа
Организационные элементы		
Применятся все элементы из организационной модели с такими же названиями и областью применения 		

Допустимые типы отношений

– is predecessor of – является предшественником
– leads to – приводит к результату, является причиной
– activates – активизирует
– creates – порождает
– has state – имеет режим
– provides input to – обеспечивает ввод информацию
– creates output to – создает результат
– supports – поддерживает

Правила построения eEPC-моделей

1. Каждая модель должна начинаться, как минимум, одним стартовым инициирующим событием и завершаться, как минимум, одним результирующим событием.

2. События и функции по ходу выполнения процесса должны чередоваться (сменять друг друга).

3. События и функции должны иметь только по одному входящему и одному исходящему отношению, показывающему ход выполнения процесса (модель «один вход – один выход» (single input – single output))

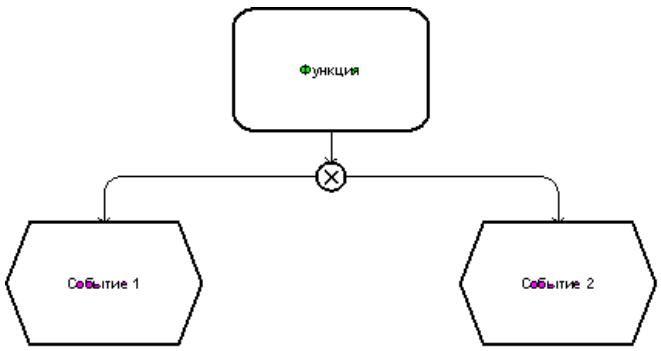
4. Путь процесса всегда разделяется и объединяется с помощью правил.

5. События, следующие после правил выбора, должны описывать все возможные результаты выбора

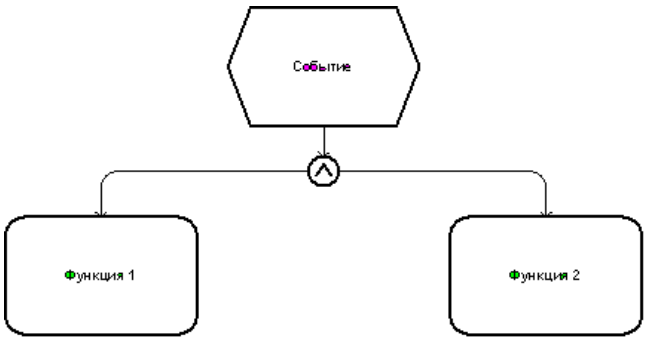
6. Правила ветвления/слияния не могут располагать одновременно несколькими входящими и исходящими соединениями.

7. На диаграмме eEPC-модели допустимы следующие варианты использования правил ветвления/слияния:

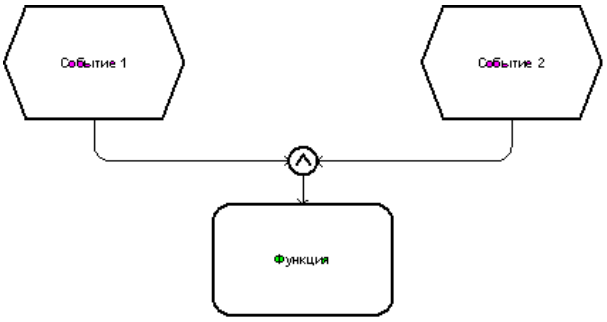
– условное ветвление процесса с помощью правила «исключающее ИЛИ»:



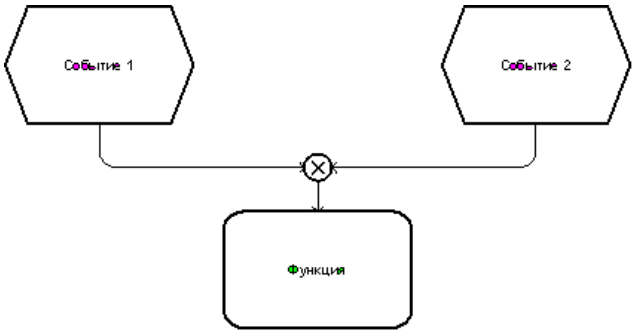
– распараллеливание процесса с помощью правила «И»:



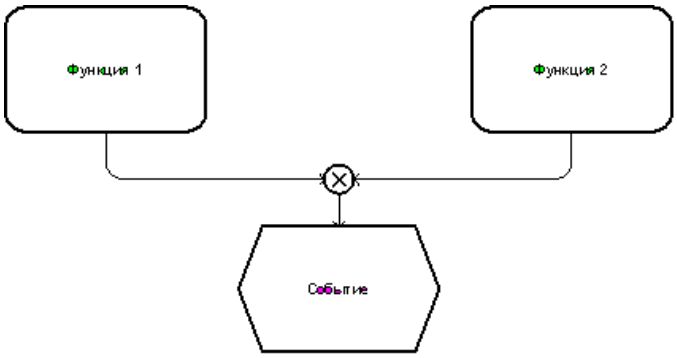
– ожидание наступления нескольких состояний с помощью правила «И»:



– наступление строго одного из состояний с помощью правила «исключающее ИЛИ»:



– выполнение одной из функций необходимого состояния с помощью правила «исключающее ИЛИ»:



2.3.5. Модель описания функций (Function allocation diagram, FAD)

Модель описания функций необходима для того, чтобы разгрузить eEPC-модель. Модель описания функций чаще всего является декомпозиций eEPC-модели. Основные элементы точно такие же, как в eEPC-модели, за исключением событий и правил (логических операторов). FAD-модель показывает:

– потоки входной информации, необходимой для реализации данной функции;

– потоки выходной информации, образующейся в результате переработки входной;

– физические лица или группы людей, принимающие участие в выполнении описываемого процесса;

– средства связи, посредством которых информация поступает в систему.

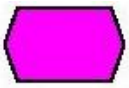
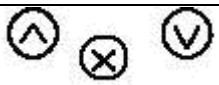
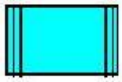
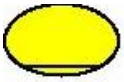
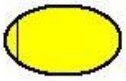
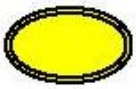
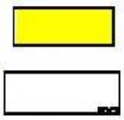
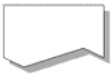
2.3.6. Офисная модель

2.3.7. Модель промышленного процесса

Эти модели представляют собой графический вид eEPC-модели (более простой и наглядный) и автоматически получаются из eEPC-модели. Основные элементы имеют такие же названия и назначения, как и в eEPC-модели, но отличаются графически. Ниже приведена таблица соответствия элементов (табл. 2.8).

Таблица 2.8

Таблица соответствия элементов

Название	**Изображение элемента в eEPC**	**Изображение элемента в производственной модели**	**Изображение элемента в офисной модели**
Событие (Event)			
Функция (Function)			
Правила (Rules)			
Используемое средство (Application System)			
Место (Location)			
Организационная единица			
Группа (Group)			
Должность (Position)			
Сотрудник (Internal person External person)			
Элементы данных	 Телефон  Документ	 Документ  Телефон	 Документ  Телефон

2.3.8. С3-модель

С3-модель используется для описания процессов реорганизации деятельности предприятия. Для создания С3-модели выделяется отдельный бизнес-процесс, который детализируется с целью дальнейшего реинжиниринга

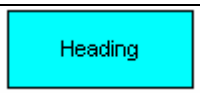
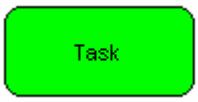
Для каждого процесса (на диаграмме) отображаются следующие аспекты:

– организационные аспекты: ответственные за процесс и реорганизацию;
– ключевые показатели, представляющие меру улучшения процесса;
– инструменты, используемые для улучшения процесса;
– потенциальные возможности улучшения процесса;
– запланированные действия для обеспечения изменений процесса;
– необходимые навыки;
– цели, преследуемые проектом.

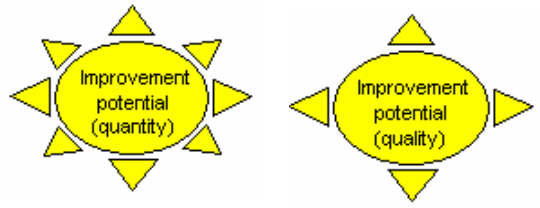
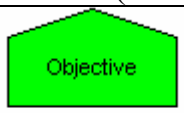
Основные элементы С3-модели приведены в табл. 2.9.

Таблица 2.9

Основные элементы С3-модели

Изображение и название элемента	**Описание элемента**
 Заголовок (Heading)	Предназначен для ввода комментариев
 Функция (Function)	Предназначена для определения бизнес-процесса
 Задание (Task)	Предназначено для определения бизнес-процесса
 Позиция (Position)	Предназначена для определения ответственного лица за бизнес-процесс
 Ключевые показатели (качественные и количественные)	Предназначены оценки качества процесса реорганизации

Продолжение табл. 2.9

 Инструменты (целевые и реальные)	Предназначены для определения средств и систем, требуемых для улучшения бизнес-процесса
 Потенциальные возможности улучшения (качественные и количественные)	Предназначены для определения параметров улучшения
 Требуемые навыки (Core competence)	Определяют навыки, которые необходимы для процесса реорганизации
 Цели (Objective)	Определяют цели, которые желают достичь в процессе реорганизации

В С3-модели существуют строго формализованные правила построения. Каждый элемент имеет определенную ячейку размещения, каждый столбец имеет собственный смысл. В первом столбце размещается информация о текущем состоянии бизнес-процесса, во втором столбце – информация о необходимых критериях и инструментах реорганизации процесса, третий столбец показывает, что получиться в процессе реорганизации.

2.3.9. Пример ARIS-модели

Для более детального изучения концепции ARIS и возможностей применения рассмотрим пример моделирования деятельности банка.

Первоначально при создании ARIS-модели разрабатывается Value Added Chain Diagram (VACD), которая представляет собой согласованный набор основных видов деятельности предприятия, создающих ценность (приносящих прибыль), начиная от исходных источников сырья и заканчивая готовой продукцией/ услугами, доставленных конечному пользователю. VACD-модель представляет собой верхний уровень иерархии модель (эквивалент контекстной диаграммы).

Работа банка рассматривается как выполнение 8 основных бизнес-функций, которые также могут декомпозироваться (рис. 2.35).

После создания VACD-модели была проведена декомпозиция функций и разработаны 5 диаграмм декомпозиции в формате eEPC-моделей. Полученные eEPC-модели позволяют подробно описать логику бизнес-процесса, информационные объекты, необходимые для выполнения процесса и являются его результатами, а также ресурсы, посредством которых будет реализован этот процесс.

Ниже представлены наиболее интересные модели (по компонентному составу и содержанию) процессов оформления и погашения кредита для физических лиц, описанные с помощью eEPC-модели (Рис. 2.36 и Рис. 2.37).

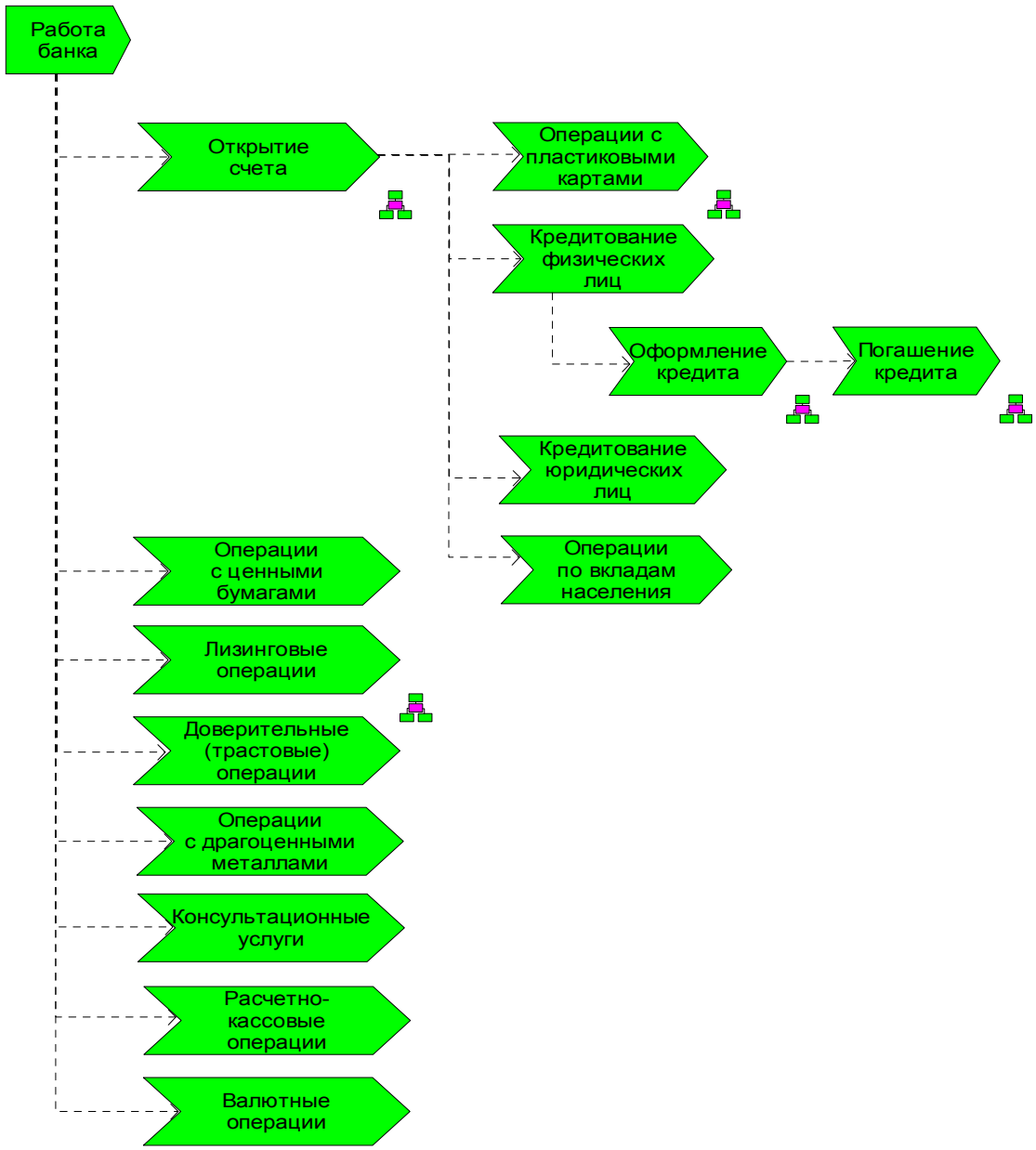


Рис. 2.35. Модель VACD – основные виды деятельности (бизнес-процессы) коммерческого банка

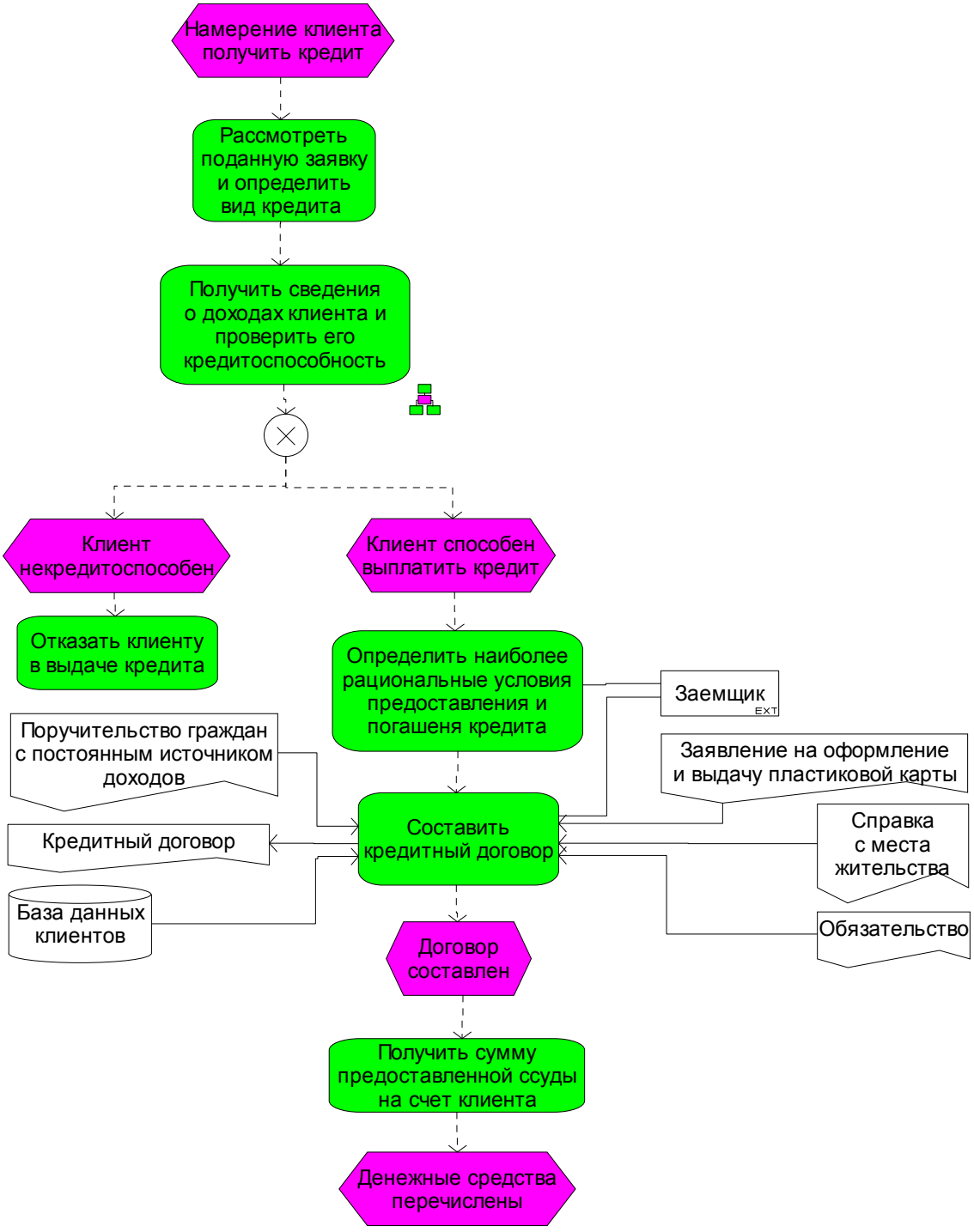


Рис .2.36. eEPC-модель процесса оформления кредита физическому лицу

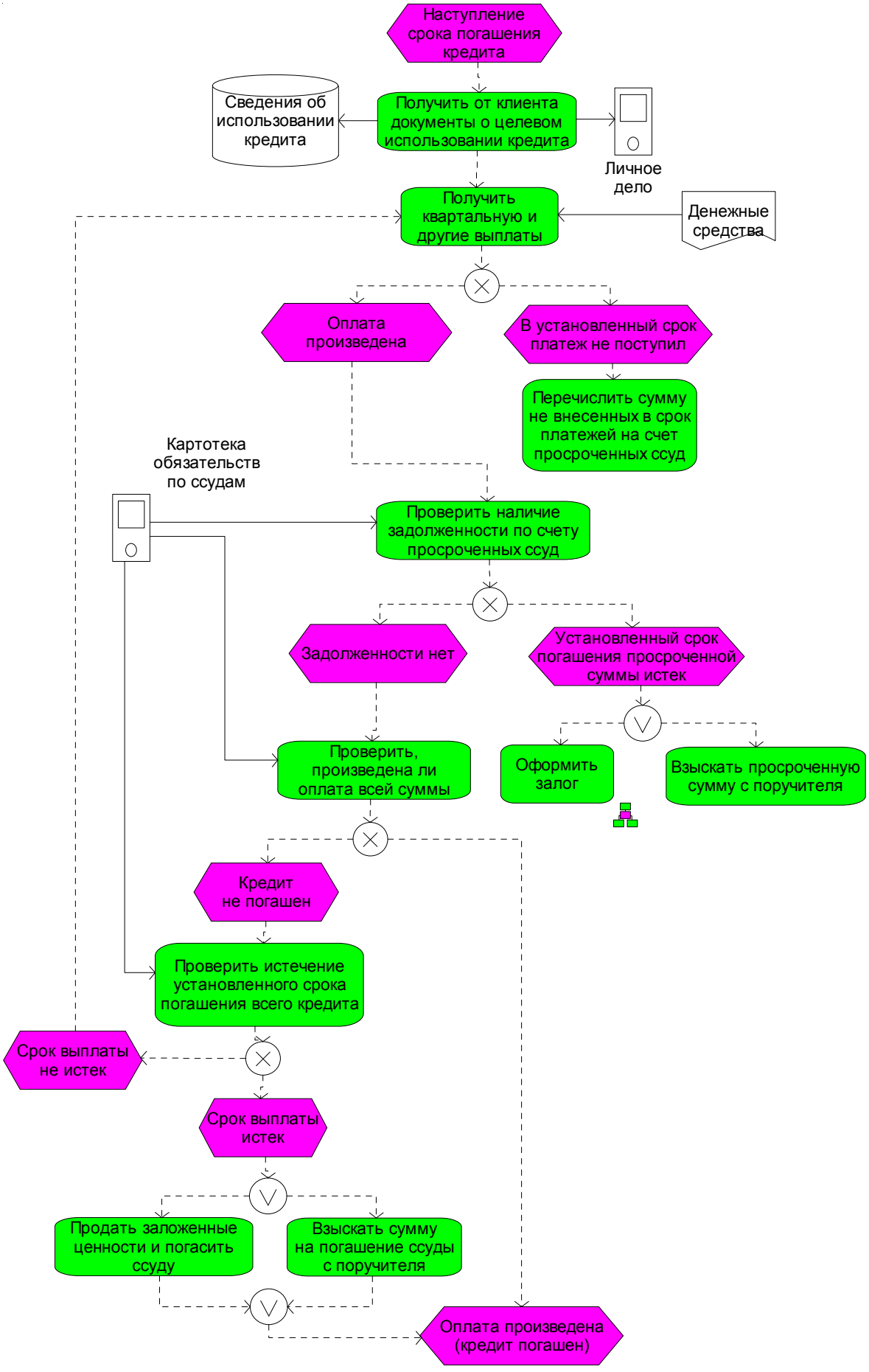


Рис. 2.37. eEPC-модель процесса погашения кредита физическим лицом

После создания eEPC-модели, если существует необходимость в дальнейшей декомпозиции или для того, чтобы разгрузить eEPC-модель, разрабатывают Function Allocation Diagram (FAD). FAD-модель служит для описания простейших процессов (отдельных функций). В нашем случае, была разработана FAD-модель процесса «Получить сведения о доходах клиента и проверить его кредитоспособность» (рис. 2.38.), так как именно этот простейший процесс имеет 11 присоединенных элементов, которые несомненно загромоздили бы eEPC-модель.

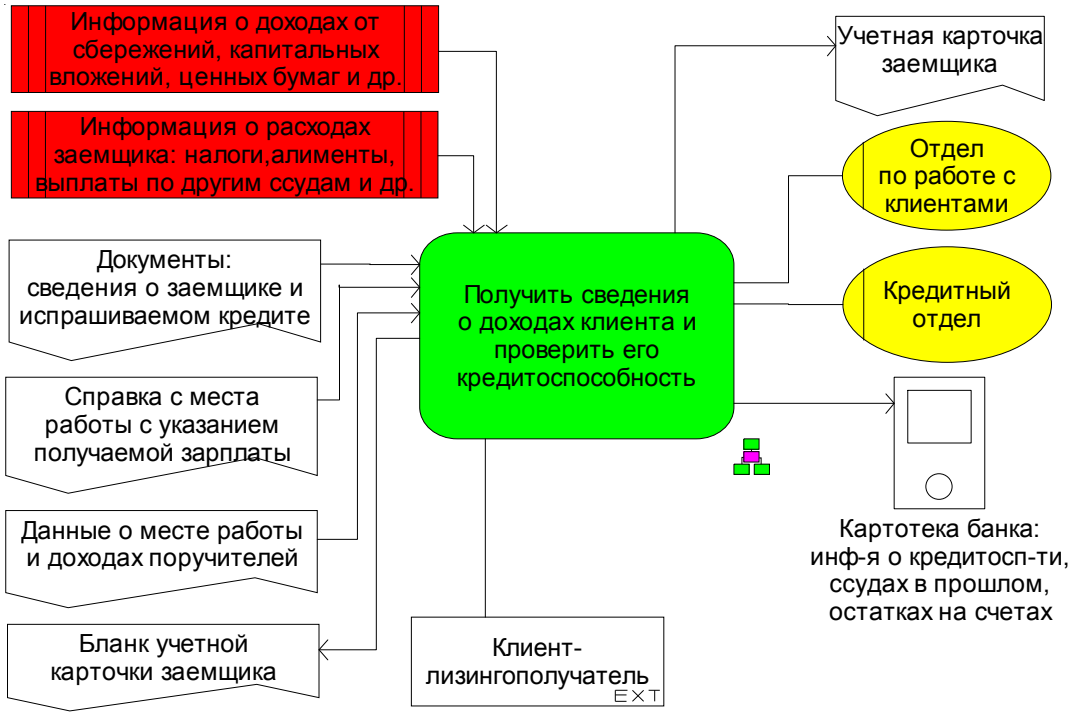


Рис. 2.38. FAD-модель процесса «Получить сведения о доходах клиента и проверить его кредитоспособность»

После чего, была создана организационная модель, представляющая структуру предприятия в виде иерархии организационных подразделений, должностей, групп сотрудников и конкретных лиц. Организационная модель данного коммерческого банка представлена на рис. 2.39 и описывает статические отношения между различными структурными элементами.

Модель дерева функций предназначена для представления иерархической структуры функций, выполняемых банком для реализации тех или иных бизнес-процессов, и отображает статические связи между ними (рис. 2.40).

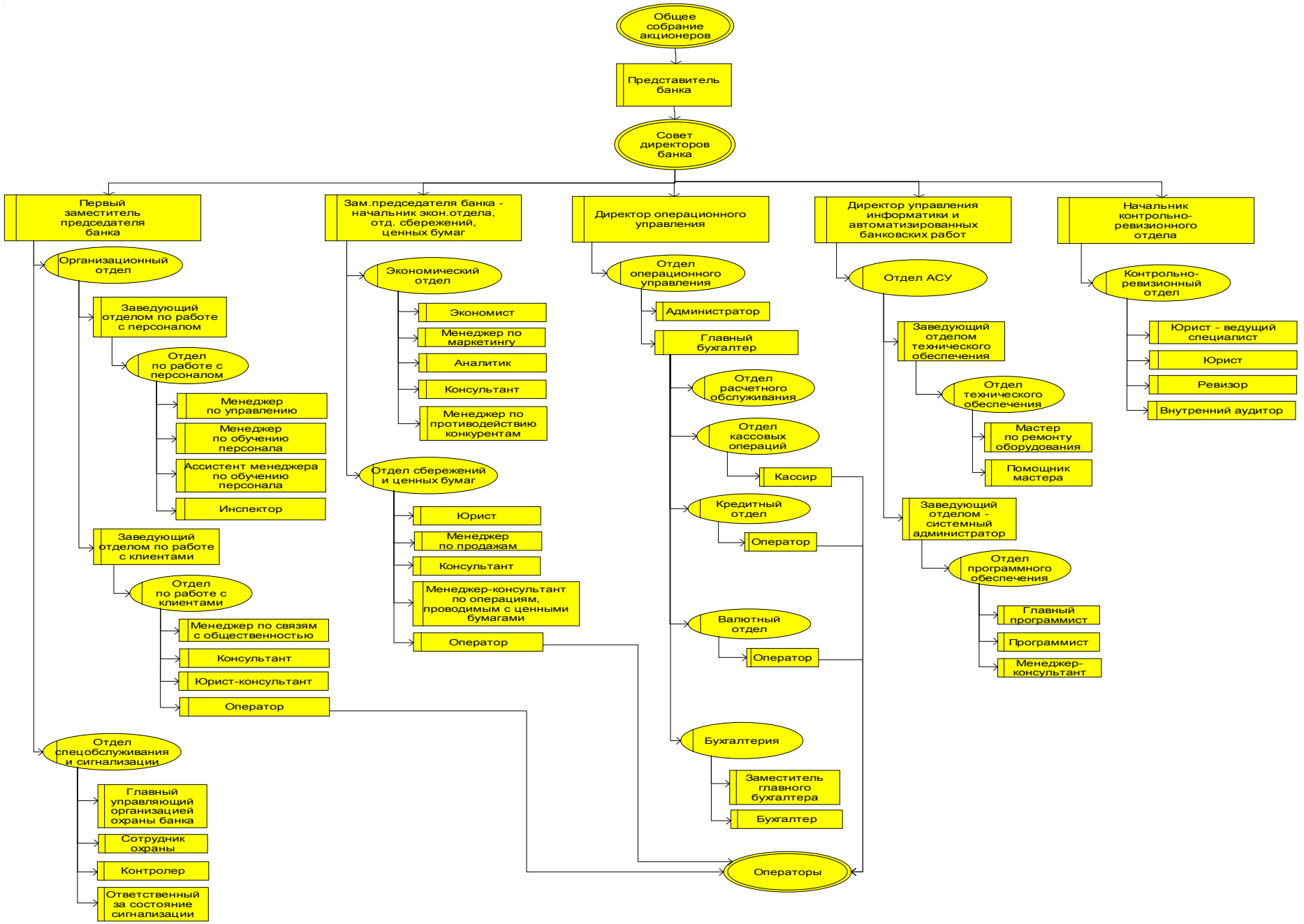


Рис. 2.39. Организационная модель коммерческого банка

Рис. 2.40. Модель дерева функций коммерческого банка

Рис. 2.44. Модель дерева функций банка

Последним этапом разработки комплексной ARIS-модели было создание С3-модели, которая предназначена для описания бизнес-процессов, которые необходимо реорганизовать. В качестве бизнес-процесса, требующего реорганизации был выбран процесс «Осуществление лизинговых операций», определены потенциальные возможности для улучшения: увеличение знаний работника о процессе, сокращение времени выполнения процесса и повышение конкурентоспособности. Было выявлено, каким образом можно достичь желаемых результатов: проведением курсов повышения квалификации сотрудников, внедрением CRM-системы, развитием электронных ресурсов банка и проведением широкомасштабных рекламных мероприятий. В результате проводимых в банке изменений необходимо создать новый операторский центр, модифицировать информационную систему, обучить персонал и разработать новые маркетинговые стратегии. При достижении всех изменений сотрудники банка за счет снижения времени обслуживания клиентов, должны увеличить их количество. С3-модель реорганизации бизнес-процесса «Осуществление лизинговых операций» приведена на рис. 2.41.

Рис. 2.41. С3-модель процесса «Осуществление лизинговых операций»

Итак, в результате описания деятельности банка была получена комплексная модель, показывающая организационную структуру, структуру и иерархию основных бизнес-процессов и модель реорганизации. Модель можно тиражировать и модифицировать для описания работы других банков.

2.4. Задачи к главе 2

Задание № 1

1. Создайте иерархическую IDEF0-модель, согласно варианту задания. Окончательная модель должна содержать четыре уровня иерархии (A-0 (контекстная диаграмма), A0 (основные бизнес-процессы), A1…A6 и 3 диаграммы декомпозиции 4 уровня по выбору студента).

2. Для полученной модели создайте дерево функций и организационную модель.

3. Проделайте процесс слияния и расщепления моделей.

4. Проведите количественный анализ полученной модели (рассчитать коэффициент декомпозиции и сбалансированности).

Вариант 1

Создать функциональную модель деятельности библиотеки, учитывая работу библиотеки с клиентами и поставщиками книг. Следует отметить, что кроме выдачи книг современные библиотеки оказывают своим клиентам дополнительные услуги: выдают клиентам CD, видео и аудио кассеты, проводят конференции, делают копирование, ламинирование, позволяют работать с электронными каталогами и выходить в Интернет.

Вариант 2

Создать функциональную модель деятельности компьютерной фирмы, учитывая, что фирма торгует компьютерами в собранном виде и комплектующими. Фирма работает как с производителями компьютерной техники, так и с клиентами. Фирма оказывает ряд дополнительных услуг: установка программного обеспечения, подключает к интернету клиентов, гарантийное обслуживание и т.д.

Вариант 3

Создать функциональную модель деятельности торговой фирмы по реализации продовольственной продукции, учитывая работу фирмы с клиентами, поставщиками, доставку продукции от поставщиков и по торговым точкам клиентов.

Вариант 4

Создать функциональную модель деятельности крупного автосалона, учитывая то, что автосалон оказывает услуги по гарантийному обслуживанию клиентов, имеет собственную автомастерскую, работает непосредственно с производителями машин, с клиентами, оказывает услуги по оформлению документов.

Вариант 5

Создать функциональную модель работы аэропорта, учитывая работу аэропорта с авиакомпаниями, клиентами, поставщиками и т.д. Учесть, всевозможные работы аэропорта по техническому обслуживанию самолетов, обслуживанию клиентов через кассы, работу диспетчерской службы аэропорта.

Вариант 6

Создать функциональную модель работы строительной фирмы. Описать работу фирмы, как с поставщиками, так и с клиентами. Следует отметь, что в настоящее время строительные организации обеспечивают полный технологический процесс, начиная проведения исследований рынка, создания проекта, закупки материалов, непосредственного строительства и заканчивая продажей квартир.

Задание № 2

Согласно варианту задания разработайте одноуровневую IDEF3-модель технологического или бизнес–процесса. В модели используйте ссылки, единицы работ, связи и максимально возможное количество различных типов перекрестков.

Варианты заданий

1. Технологический процесс создания микросхемы.
2. Технологический процесс сборки компьютера.
3. Технологический процесс изготовления электроламп.
4. Технологический процесс ремонта телевизора.

5. Технологический процесс производства мебели на заказ.
6. Технологический процесс пошива изделия.
7. Технологический процесс разработки программного продукта.
8. Технологический процесс выпуска сотовых телефонов.

Задание № 3

Согласно варианту задания разработать иерархическую DFD-модель (А-0, А0 и 3 диаграммы третьего уровня). Особое внимание уделить потокам данных и хранилищам данным. На каждом уровне декомпозиции выделить хранилища данных.

Вариант 1

Создать диаграмму потоков данных процесса «ПРОВЕСТИ ОБСЛЕДОВАНИЕ ДЕЯТЕЛЬНОСТИ ПРЕДПРИЯТИЯ» при работе консалтинговой группы.

Создать словарь данных, описав все хранилища данных и внешние сущности.

Вариант 2

Создать диаграмму потоков данных процесса «ПРОВЕСТИ МАРКЕТИНГОВЫЕ ИССЛЕДОВАНИЯ», подробно рассмотрев все процессы, происходящие при этом. В качестве внешних сущностей можно выбрать «КЛИЕНТ» и «РЫНОК».

Создать словарь данных, описав все хранилища данных и внешние сущности.

Вариант 3

Создать диаграмму потоков данных процесса «ПЛАНИРОВАТЬ ДЕЯТЕЛЬНОСТЬ ПРЕДПРИЯТИЯ», учитывая финансовую, хозяйственную и прочие деятельности предприятия.

Создать словарь данных, описав все хранилища данных и внешние сущности.

Вариант 4

Создать диаграмму потоков данных процесса «СОЗДАТЬ ПРОГРАММУ» при работе программиста над разработкой и созданием ПО.

Создать словарь данных, описав все хранилища данных и внешние сущности.

Вариант 5

Создать диаграмму потоков данных процесса «РАЗРАБОТАТЬ КОНСАЛТИНГОВЫЙ ПРОЕКТ», учитывая основные этапы при проведении консалтинга:

- анализ первичных требований;
- проведение обследования деятельности предприятия;
- построение моделей «как есть» и «как должно быть»;
- оценка эффективности деятельности предприятия;
- реорганизация деятельности;
- разработка системного проекта;
- разработка предложений по автоматизации;
- выбор, разработка и внедрение новой информационной системы.

Создать словарь данных, описав все хранилища данных и внешние сущности.

Задание № 4

Разработайте комплексную BPWin-модель, состоящую из трех видов диаграмм: IDEF0, DFD и IDEF3. Контекстная диаграмма уровня A-0 и диаграмма уровня A0, с использованием IDEF0-методологии, затем 3 блока декомпозируются на DFD-диаграммы и по 1 блоку каждого уровня DFD декомпозируются на IDEF3 (3 IDEF3-диаграммы). Таким образом, должна получиться модель, состоящая из 8 диаграмм.

2.5. Вопросы к главе 2

1. Что такое SADT, и как SADT связана с IDEF?
2. Перечислите основные структурные элементы IDEF0-методологии.
3. Какова роль стрелки вызова, и чем она отличается от других стрелок?
4. Для чего необходимы IDEF3-модели, и назовите их основное отличие от IDEF0-моделей?
5. Скажите, к какому типу стрелки будут относиться *ПРИКАЗЫ РУКОВОДСТВА*? А *АВТОТРАНСПОРТ*?
6. В чем разница синхронных и асинхронных перекрестков?
7. Что такое ссылка?
8. Почему перекресток «Исключающее ИЛИ» не может быть синхронным?
9. Нарисуйте временную диаграмму срабатывания перекрестка «Асинхронное И».
10. В виде какого элемента будет изображен ЗАКАЗЧИК в IDEF3-модели?
11. Назовите при выполнении каких проектов лучше всего использовать DFD?
12. Перечислите нотации, с использованием которых можно строить DFD-модель? В чем отличие этих нотаций?
13. Перечислите, в порядке значимости, элементы DFD-методологии, начиная с самого важного.
14. В виде, какого элемента будет изображено *КНИГОХРАНИЛИЩЕ* на диаграмме, описывающей работу библиотеки?
15. Как расшифровывается сокращение ARIS? Для каких целей наиболее эффективно использование концепции ARIS?
16. Почему, несмотря на свою «молодость», концепция ARIS находит широкое распространения при моделировании бизнес-процессов предприятия?
17. Перечислите основные типы моделей, предложенные разработчиками концепции ARIS.
18. Каким образом связан процесс реинжиниринга бизнес-процессов предприятия и концепция ARIS?
19. В чем заключается разница в понятиях «реорганизации» и «реинжиниринга»?
20. Что такое «бизнес-процесс»? Дайте определение этому термину.

3. Имитационное моделирование систем

3.1. Достоинства и недостатки имитационного моделирования систем

Как было рассмотрено в главе 1, 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 [4, 17, 23, 27].

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 [29]. 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 [14].

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 [14]:

– 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 [14]. 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 [14].

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 [14].

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 С.

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 [14].

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 за-

трат. 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 7.0.

3.2. Математические основы ПП Arena 7.0

В основе ПП Arena 7.0 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 7.0 – рассмотрим основы этих математических аппаратов [30].

3.2.1. 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 [38].

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. 3.1 приведены элементы СМО [6, 10, 11].

Таблица 3.1

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 [1, 5, 12]:

– 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 [27]. 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.

3.2.2. 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 [39].

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 [9, 39].

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 [9, 11, 15, 19, 21], 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 $T = \{t_1, t_2, ..., t_n\}$ – подмножество вершин, называющихся переходами;

$P = \{p_1, p_2, ..., p_m\}$ – 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. 3.2.

Таблица 3.2

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. 3.1 приведен пример интерпретации сети Петри.

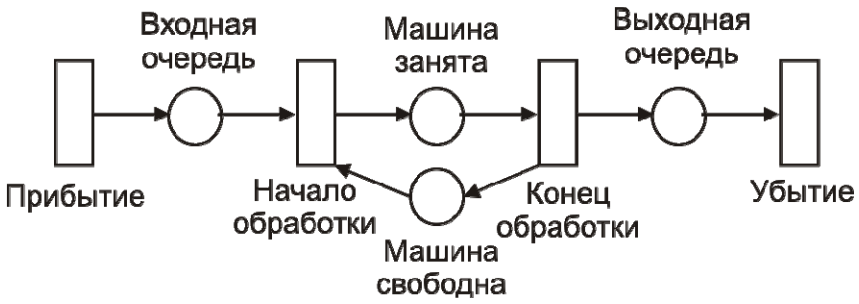


Рис. 3.1. Интерпретация сети Петри

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 [21].

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 [12];

– 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 7.0. 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. ***Input 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 [30, 32, 34].

Система Arena позволяет моделировать виды деятельности, представленные на рис. 3.2.

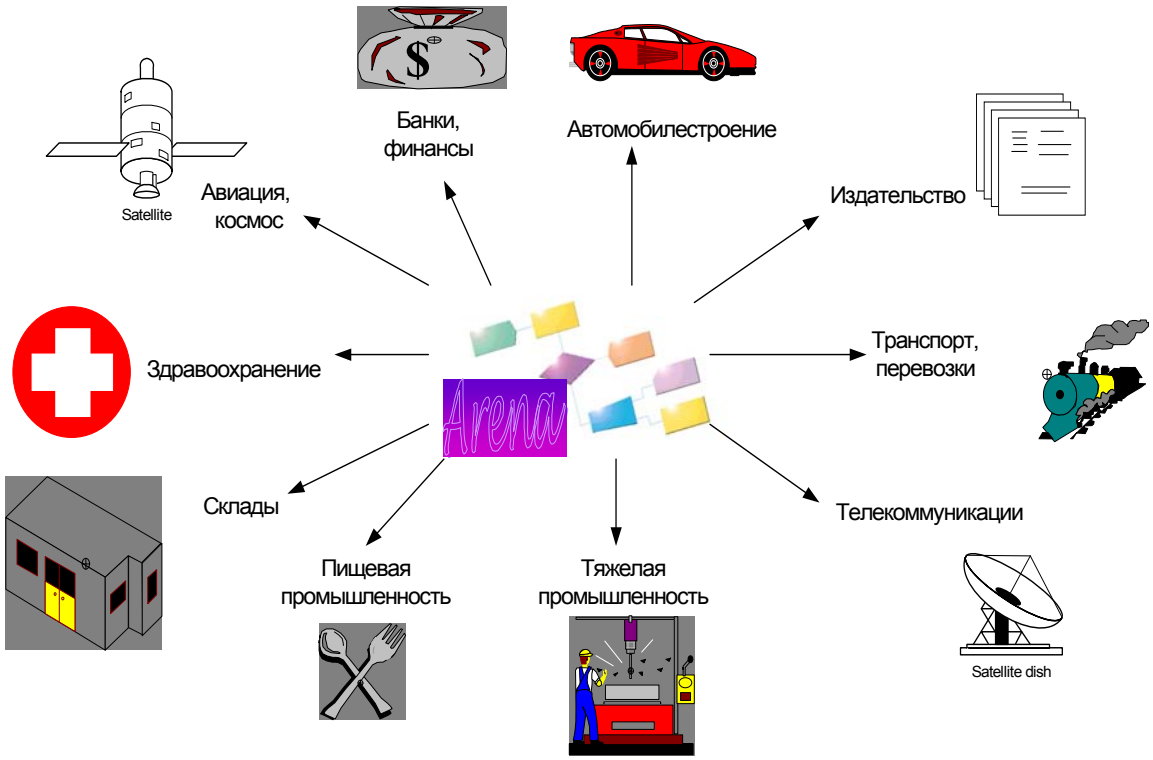


Рис. 3.2. Области применения Arena

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 [30, 32, 33, 34].

3.3. Начало работы с программным пакетом Arena 7.0

Для того чтобы создать новую модель, необходимо открыть ПП Arena 7.0 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. 3.3.

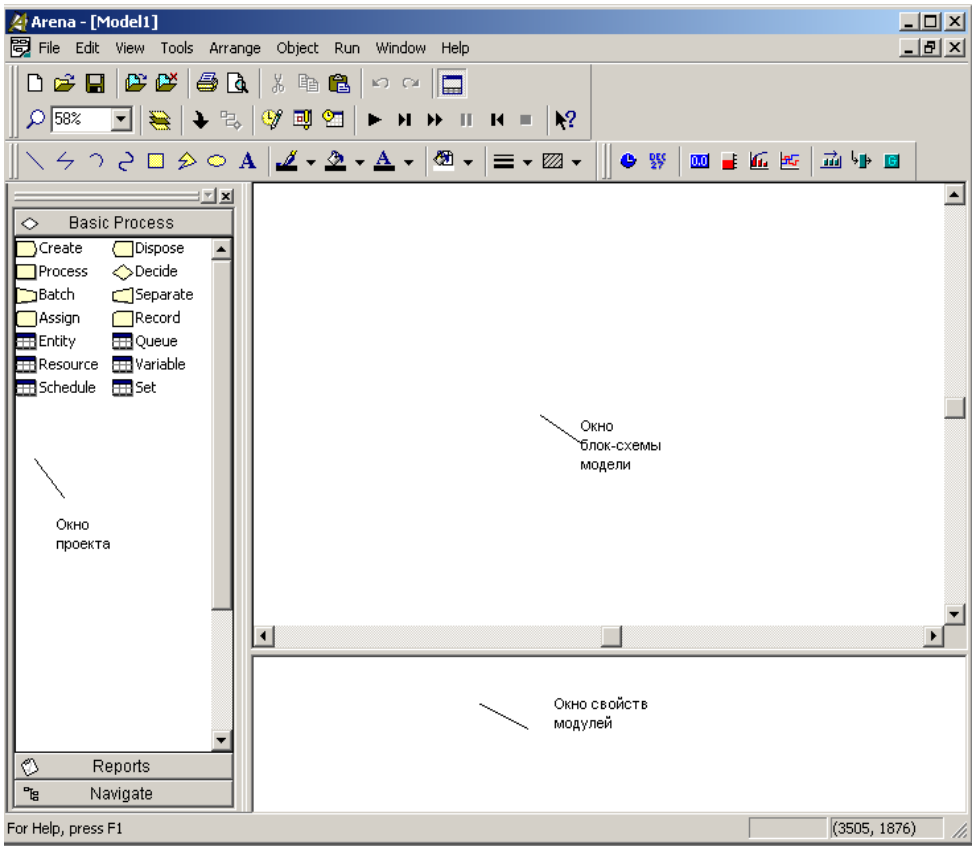


Рис. 3.3. Среда моделирования Arena

Окно приложения разделено на три области:

1. *Окно рабочего поля модели*, в котором описывается логика модели с использованием схемных (графических) модулей. 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.

3.4. Basic Process Panel (панель основных процессов)

3.4.1. Схемные модули

Модуль Create

Этот модуль является отправной точкой для сущностей в имитационной модели. Сущности – это индивидуальные элементы, обрабатываемые в системе. 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.

Таблица 3.3

Параметры модуля Create

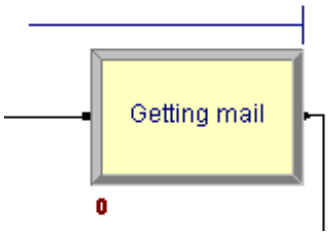
Параметры	Описание
Name	Уникальное имя модуля, которое будет отражено в блок-схеме
Entity Type	Название типа сущности, который будет создаваться модулем
Type	Способ формирования потока прибытия. Type может иметь значения: *Random* (используется экспоненциальное распределение со средним значением, определенным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

Окончание табл. 3.3

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

Этот модуль 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.

Таблица 3.4

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

Окончание табл. 3.4

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
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».

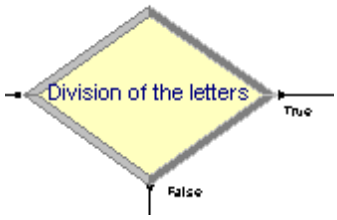
На прием к доктору приходят пациенты двух типов: взрослые и дети. Доктор (наш ресурс) – один. Он ведет прием и детей, и взрослых, но детей доктор принимает около 30 минут, а взрослых около 20 минут, причем у детей приоритет выше, чем у взрослых.

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 «доктор». 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

Этот модуль 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.

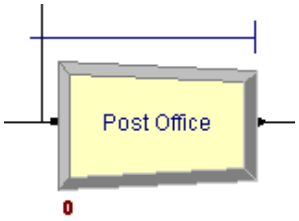
Таблица 3.5

Параметры модуля Decide

Параметры	Описание
Name	Уникальное имя модуля, которое будет отражено в блок-схеме
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 выполнение
Named	Имя переменной, атрибута или типа сущности, который будет проверяться при входе сущности в модуль
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

Этот модуль 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.

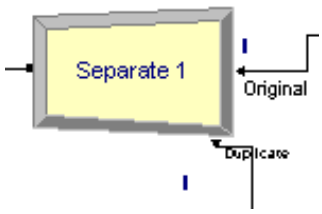
Применение: собрать необходимое количество данных, прежде чем начинать их обработку; собрать ранее разделенные копии одной формы; соединить пациента и его больничную карту приема к врачу.

Таблица 3.6

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 сгруппированы
Attribute Name	Имя атрибута, по значению которого будут группироваться сущности

Модуль Separate



Этот модуль 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 по одному заказу.

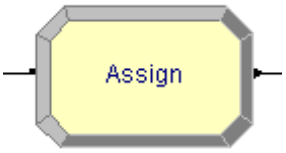
Таблица 3.7

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

Этот модуль 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.

Таблица 3.8

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

Этот модуль 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.

Таблица 3.9

Параметры Модуль 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

Модуль Dispose

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

Применение: окончание бизнес-процесса; клиенты покидают отдел.

Таблица 3.10

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

3.4.2. 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

Этот модуль 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.

Таблица 3.11

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 Queue

Этот модуль данных 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).

Таблица 3.12

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
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 смоделируем 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

Этот модуль 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).

Таблица 3.13

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

Окончание табл. 3.13

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

Этот модуль 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.

Таблица 3.14

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 Set

Это модуль данных, 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.

Таблица 3.15

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

Этот модуль данных 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.

Таблица 3.16

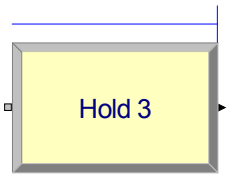
Параметры модуля Variable

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

3.5. Advanced Process Panel (панель усовершенствованных процессов)

3.5.1. Схемные модули

Модуль Hold



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 3.17

Параметры модуля Hold

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

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.

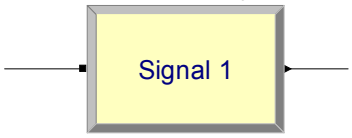
Поля Queue Type и Queue name присутствуют среди параметров модуля Hold всегда, задаются чаще всего автоматически (менять не рекомендуется).

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.

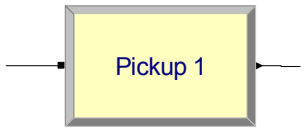
Таблица 3.18

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

Этот модуль 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.

Таблица 3.19

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

Модуль 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).

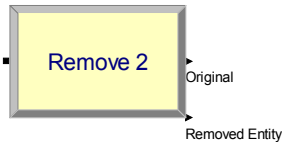


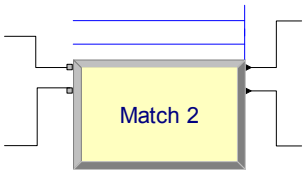
Таблица 3.20

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

Этот модуль 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.

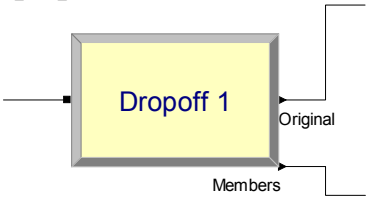
Таблица 3.21

Параметры модуля Match

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

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.

Таблица 3.22

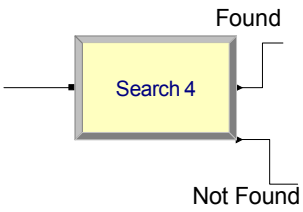
Параметры модуля Dropoff

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

Окончание табл. 3.22

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



Этот модуль 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.

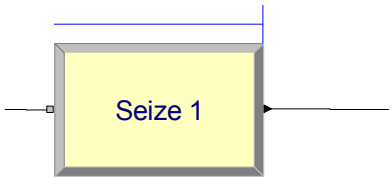
Применение: поиск среди коробок самой легкой.

Таблица 3.23

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



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.

Таблица 3.24

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
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.)

Окончание табл. 3.24

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

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.

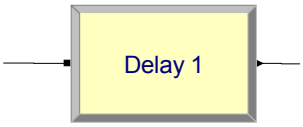


Таблица 3.25

Параметры модуля Delay

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

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 сущ-

ность 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.

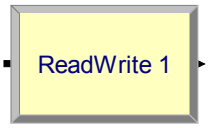
Таблица 3.26

Параметры модуля Release

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

Модуль ReadWrite

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.

Таблица 3.27

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
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

3.5.2. Модули данных

Модуль Advanced Set

Этот модуль 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.

Таблица 3.28

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

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

Таблица 3.29

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

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

Таблица 3.30

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

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

Модуль File

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.

Таблица 3.31

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

Модуль 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.

Таблица 3.32

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

Модуль Failure

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 3.33

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)

3.6. Advanced Transfer Panel (панель перемещения)

3.6.1. Схемные модули

Модуль Station

Модуль Station определяет станцию или набор станций для физической или логической обработки, некая логическая («отправная») точка в модели.

Таблица 3.34

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

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

Таблица 3.35

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

Модуль PickStation позволяет сущностям выбирать определенную станцию из множества существующих (маршрутизатор).

Таблица 3.36

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)
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

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.

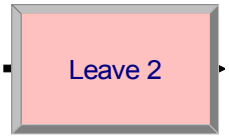
Таблица 3.37

Параметры модуля Enter

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

Этот модуль 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 конечном итоге сущность отправляется в пункт модуля назначения.

Таблица 3.38

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

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

Таблица 3.39

Параметры модуля Request

Параметры	Описание
Name	Уникальное имя модуля
Transporter Name	Название (имя) транспортера
Velocity	Скорость, с которой транспортер перемещает (единица длины в единицу времени). Единица времени определена в поле Units

Окончание табл. 3.39

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

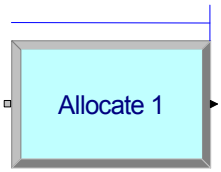
Модуль Activate активирует или увеличивает вместимость предварительно приостановленного транспортера или транспортера, который был первоначально бездействующим (как определено в модуле Transporter).

Таблица 3.40

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

Модуль Allocate

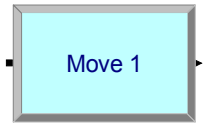


Модуль Allocate аналогичен модулю Request. Различие только в том, что модуль Allocate не позволяет задавать скорость и единицы измерения скорости транспортера.

Модуль Move

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 Transporter, 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.

Таблица 3.41

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
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

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 3.42

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
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

Модуль Free

Модуль Free освобождает транспортер для дальнейшего его использования.

Таблица 3.43

Параметры модуля Free

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

Модуль Halt

Модуль Halt изменяет состояние (статус) транспортера на неактивное. Если транспортер занят, в то время как сущность вошла в модуль 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.

Таблица 3.44

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

Этот модуль вызывает конвейер, распределяет ячейки конвейера для перемещения сущности от станции к станции. Получив контроль над ячейками конвейера, сущность может переместиться к другой

станции конвейера. Этот модуль является эквивалентом модуля Request.

Таблица 3.45

Параметры модуля Access

Параметры	Описание
Name	Уникальное имя модуля, которое будет отражено в блок-схеме
Cell	Количество ячеек, необходимых для перемещения конвейера
Conveyor Name	Имя конвейера-исполнителя
Queue Name	Имя очереди, в которую поступают сущности конвейера, если конвейер занят

Модуль Convey

Модуль Convey перемещает сущности по конвейеру от одной станции 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.

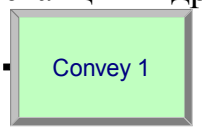


Таблица 3.46

Параметры модуля Convey

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

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.

Таблица 3.47

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

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.

Таблица 3.48

Параметры модуля Stop

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

Модуль Exit

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

Таблица 3.49

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

3.6.2. Модули данных

Модуль Transporter

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.

Таблица 3.50

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

Модуль Distance предназначен для определения пути, по которому будет двигаться транспортер.

Таблица 3.51

Параметры модуля Distance

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

Модуль Sequence

Этот модуль используется для определения последовательности для сущностей в модели при их движении. Последовательность состоит из списка станций, которые сущность должна посетить.

Таблица 3.52

Параметры модуля Sequence

Параметры	Описание
Name	Название последовательности
Station Name	Название станции
Step Name	Название станции, которая может быть в последовательности
Next Step	Шаг последовательности

Модуль Conveyor

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

Таблица 3.53

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

Модуль Segment определяет путь, по которому будет двигаться конвейер.

Таблица 3.54

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

3.7. Панель отчетов

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 группировки) 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.

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

3.8. 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.

3.9. Построитель выражений

ПП Arena позволяет строить сложные выражения. Это достигается с помощью Expression Builder. Построитель выражений имеет внешний вид, показанный на рис. 3.4.

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*.

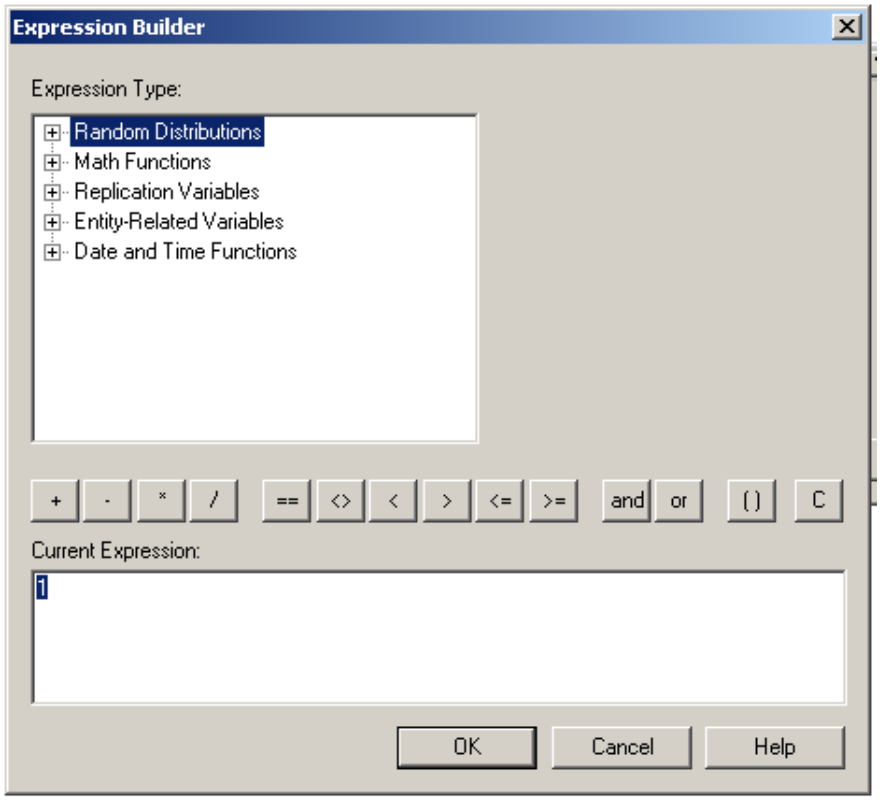


Рис. 3.4. Внешний вид построителя выражений

Остановимся более подробно на нескольких видах распределений, которые наиболее часто используются при моделировании сложных систем. Это равномерное (UNIF или Uniform), нормальное (NORM), экспоненциальное (EXPO) и треугольное (TRIA или Triangular) распределения. Эти распределения относятся к непрерывным [14].

Равномерное распределение $U(a, b)$

Варианты применения	Используется как модель величины, которая случайно изменяется между a и b
Плотность (рис. 3.5)	$f(x)=1/b-a$, если $a \le x \le b$ $f(x)=0$ в противном случае
Распределение	$f(x)=x-a/b-a$, если $a \le x \le b$ $f(x)=0$ $x<a$ $f(x)=1$ $b<x$
Параметры	a и b – вещественные числа, для которых $a<b$
Область	$[a, b]$
Среднее	$(a+b)/2$
Дисперсия	$(a+b)^2/12$

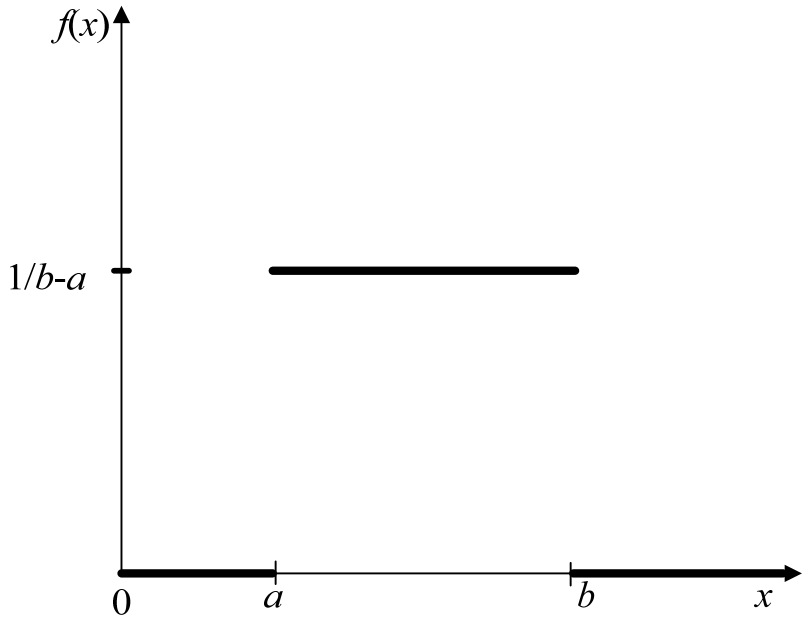


Рис. 3.5. Функция плотности распределения вероятностей равномерного распределения

Нормальное распределение $N(\mu, \sigma^2)$

Варианты применения	Ошибки различного типа, например точка попадания бомбы; величины, представляющие собой сумму большого количества других величин
Плотность (рис. 3.6)	$f(x)=(1/\sqrt{2*\pi*\sigma^2})*e^{-(x-\mu 2)/(2*\sigma 2)}$ для всех вещественных x
Параметры	μ □ (-∞; ∞), $\sigma > 0$
Область	(-∞; ∞)
Среднее	μ
Дисперсия	σ^2
Мода	μ

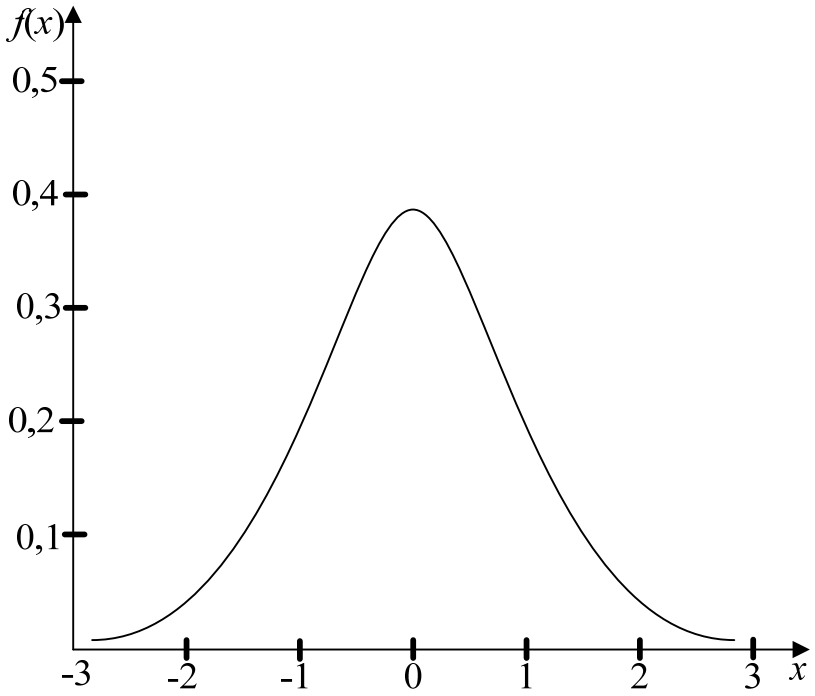


Рис. 3.6. Функция плотности распределения вероятностей нормального распределения

Экспоненциальное распределение expo(β)

Варианты применения	Интервал времени между поступлениями требований в систему, происходящими с постоянной интенсивностью; время безотказной работы устройства
Плотность (рис. 3.7)	$f(x)=(1/\beta)*e^{-x/\beta}$, если $x \geq 0$ $f(x)=0$ в противном случае
Параметры	$\beta > 0$
Область	$[0; \infty)$
Среднее	β
Дисперсия	β^2
Мода	0

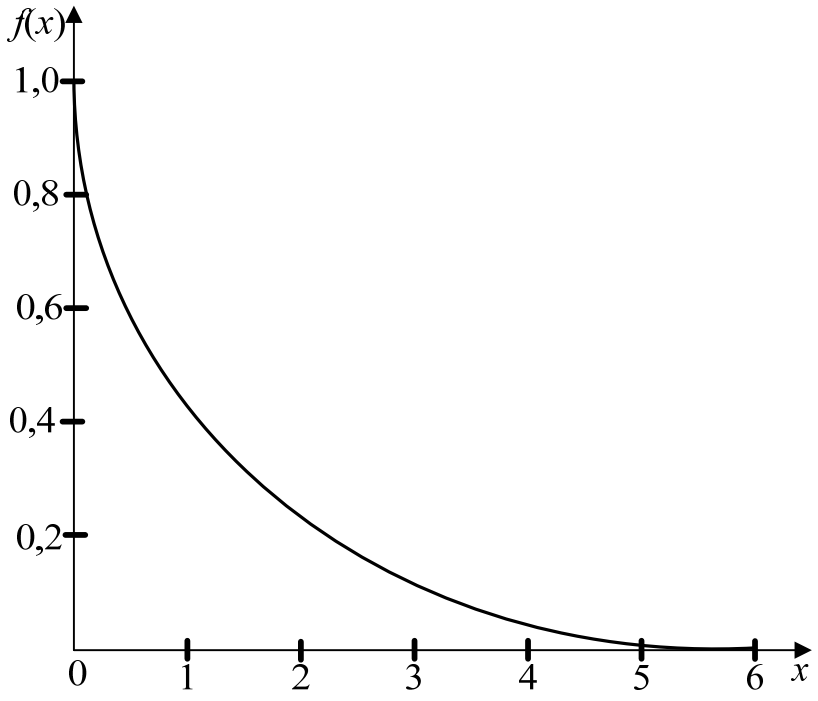


Рис. 3.7. Функция плотности распределения вероятностей экспоненциального распределения

Треугольное распределение triang(*a*, *b*, *c*)

Варианты применения	Используется как приблизительная модель в отсутствие данных
Плотность (рис. 3.8)	$f(x)=2*(x-a)/(b-a)*(c-a)$, если $a \le x \le c$ $f(x)= 2*(b-x)/(b-a)*(b-c)$, если $c \le x \le b$ $f(x)=0$ в противном случае
Параметры	*a, b* и *c* – вещественные числа, для которых $a< c <b$
Область	[*a, b*]
Среднее	$(a+b+c)/3$
Дисперсия	$(a^2+b^2+ c^2-ab-ac-bc)/18$
Мода	*c*

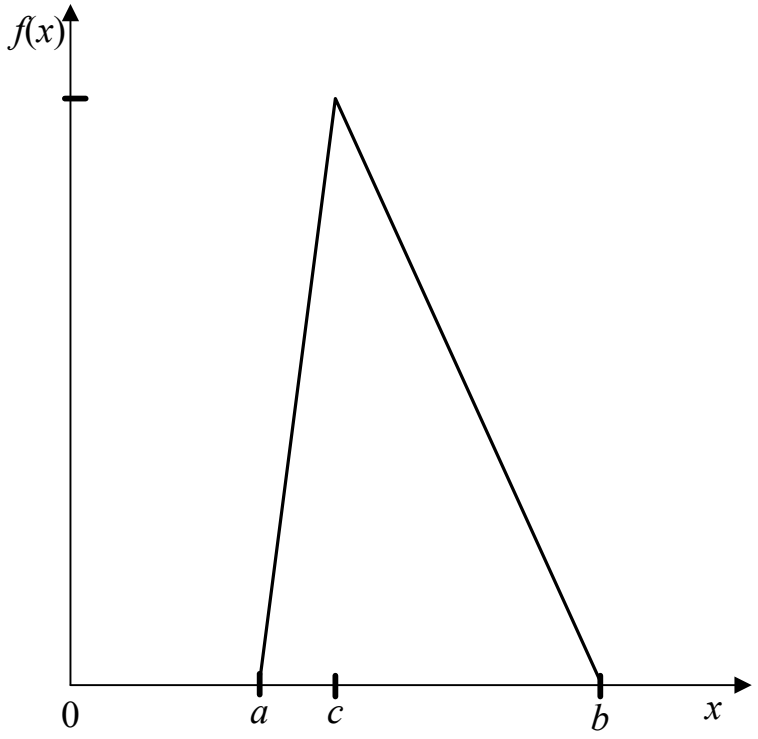


Рис. 3.8. Функция плотности распределения вероятностей треугольного распределения

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. 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).

1.4. 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.

3.10. Примеры выполнения заданий

Задание 1: Самолеты прибывают для посадки в район крупного аэропорта каждые 10±5 мин. Если взлетно-посадочная полоса свободна, прибывший самолет получает разрешение на посадку. Если полоса занята, самолет выполняет полет по кругу и возвращается к аэропорту через каждые 4 мин. Если после пятого круга самолет не получает разрешения на посадку, он отправляется на запасной аэродром.

В аэропорту через каждые 10±2 мин к взлетно-посадочной полосе выруливают готовые к взлету машины и получают разрешение на взлет, если полоса свободна. Для взлета и посадки самолеты занимают полосу ровно на 2 мин. Если при свободной полосе одновременно один самолет прибывает для посадки, а другой – для взлета, полоса предоставляется взлетающей машине.

Смоделировать работу аэропорта в течение суток. Подсчитать количество самолетов, которые взлетели, сели и были направлены на запасной аэродром. Определить коэффициент загрузки взлетно- посадочной полосы.

Рассмотрим подробно логику реализованной на рис. 3.9 модели.

1. Прибытие самолетов для взлета имитируется модулем Create «Take off». Этот модуль генерирует сущности Entity 1 в виде самолетов.

2. Главным условием взлета этих самолетов является то, что взлетно-посадочная полоса должна быть свободна. В нашей модели взлетно-посадочная полоса моделируется модулем Process 1, которому соответствует Recourse 1. После того как появляется самолет, желающий взлететь, он попадает в модуль Hold 2, который выпустит этот самолет при условии, что полоса освободилась. Взлетевший самолет, т. е. обработанный модулем Process 1, уходит из системы через модуль Dispose 2.

3. Прибытие самолетов для посадки имитируется модулем Create «Landing». Этот модуль генерирует сущности Entity 2 в виде самолетов. Модуль Assign 2 задает значение Attribute 1, равное 1; это необходимо далее для подсчета кругов.

4. При посадке по заданию должны выполняться следующие условия: полоса должна быть свободна и не должно быть самолетов, идущих на взлет, т. к. у них приоритет выше.

Это мы будем реализовывать через модуль Decide 1, в котором мы будем проверять занятость Recourse 1 в Process 1, и проверять очередь в Hold 2. Приземлившийся самолет, т. е. обработанный модулем Process 1, уходит из системы через модуль Dispose 2.

5. В Decide 2 будет проверяться следующее: если по прибытии самолета для посадки полоса (Recourse 1) будет занята и /или будут присутствовать самолеты на взлет в Hold 2, то этот самолет пойдет не по ветке True на полосу, а по ветке False.

6. В ветке False первым стоит модуль Assign 1, который увеличивает Attribute 1 на единицу каждый раз, когда он проходит по этой ветке. Затем модуль Process 2 имитирует круг над аэропортом, после чего в модуле Decide 2 проверяется, сколько уже кругов сделал этот самолет: если меньше 5, то он опять возвращается к аэропорту для поверки условий, а если уже 5, то летит на запасной аэропорт.

7. Модули Assign 3, Assign 4 и Assign 5 необходимы для сбора статистики по взлетевшим, севшим самолетам и самолетам, ушедшим на запасной аэропорт.

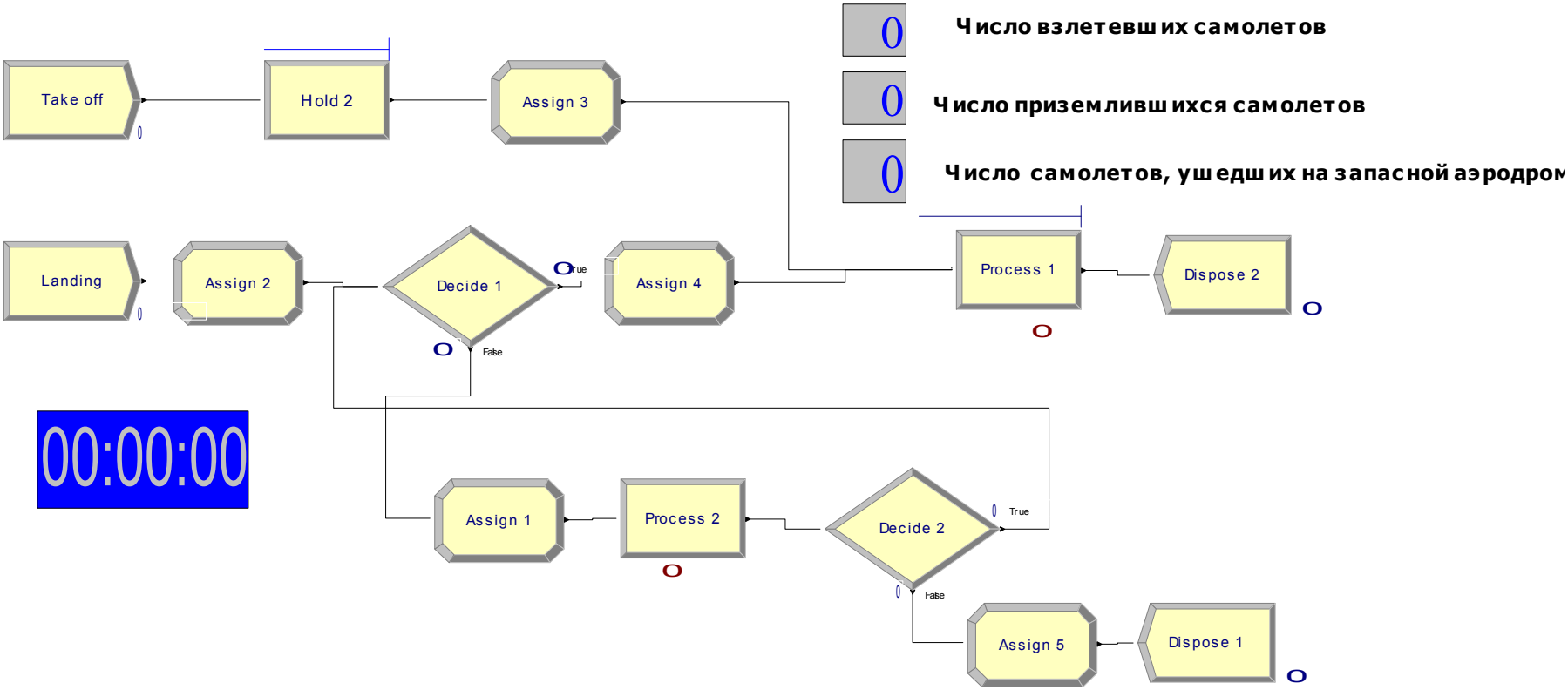
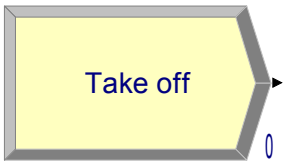
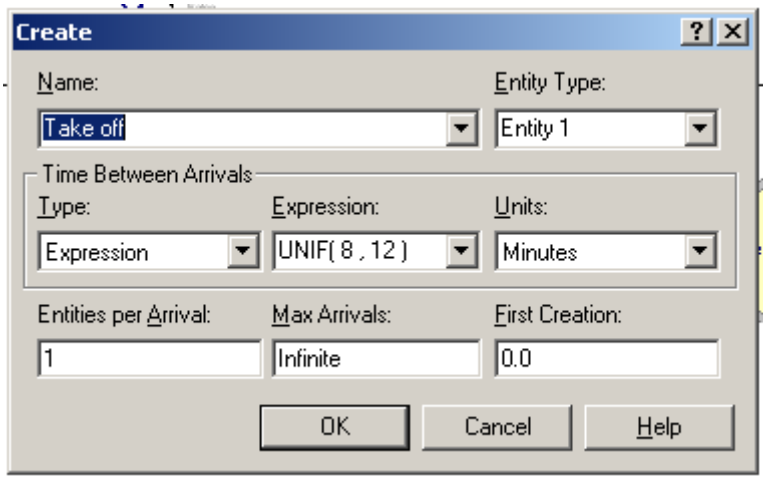


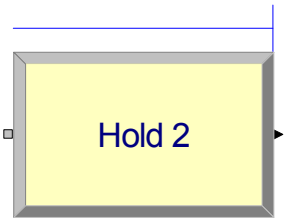
Рис. 3.9. Модель функционирования взлетно-посадочной полосы аэропорта в ПП Arena 7.0

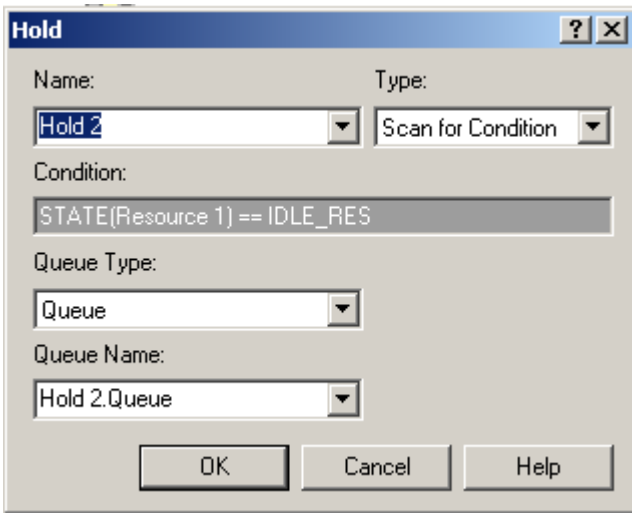
Рассмотрим более подробно наиболее интересные модули.



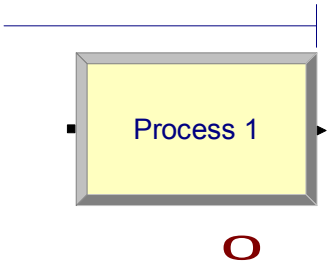


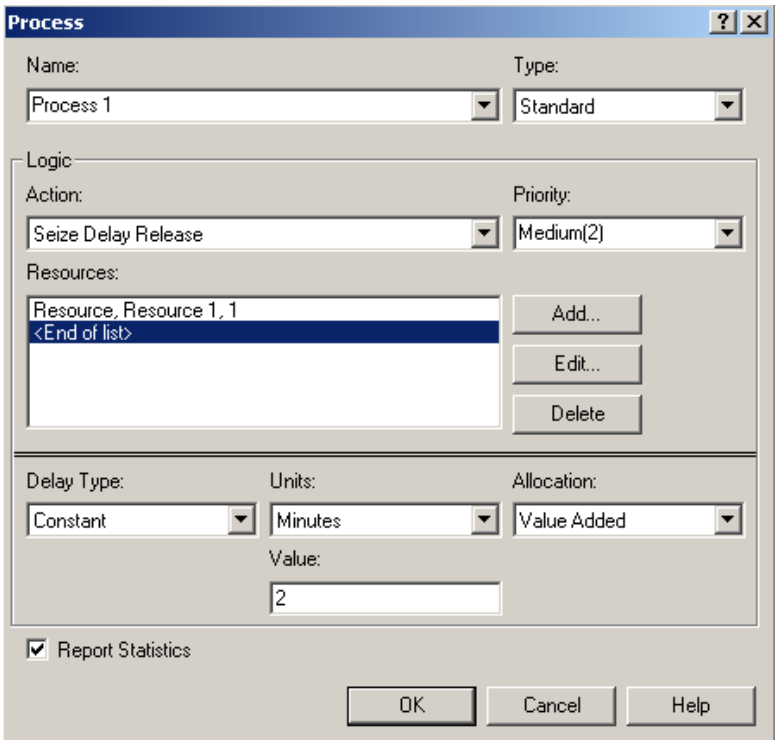
В аэропорту через каждые 10±2 мин к взлетно-посадочной полосе выруливают готовые к взлету машины.



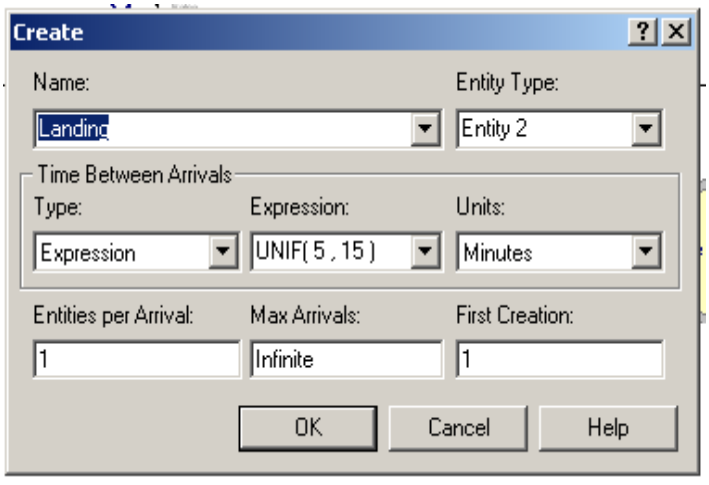


Готовые к взлету машины получают разрешение на взлет, если полоса свободна: STATE(Resource 1) == IDLE_RES.

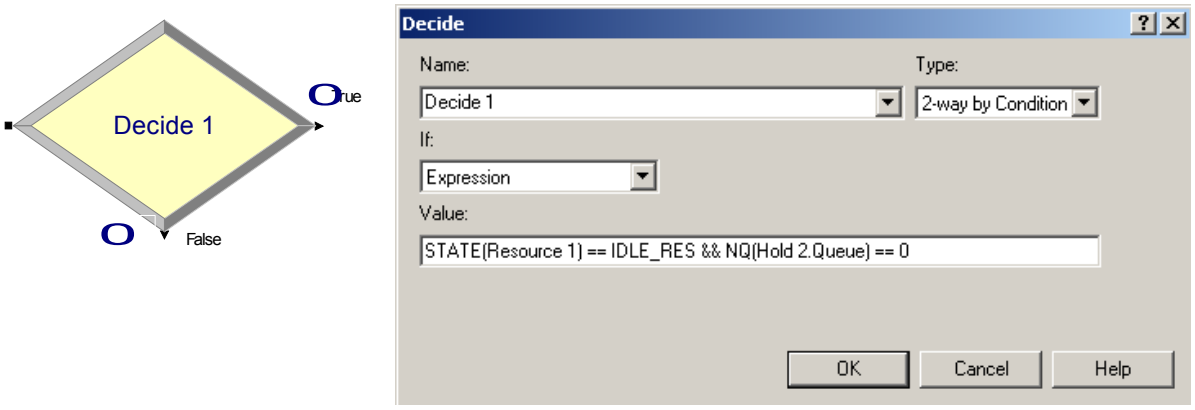




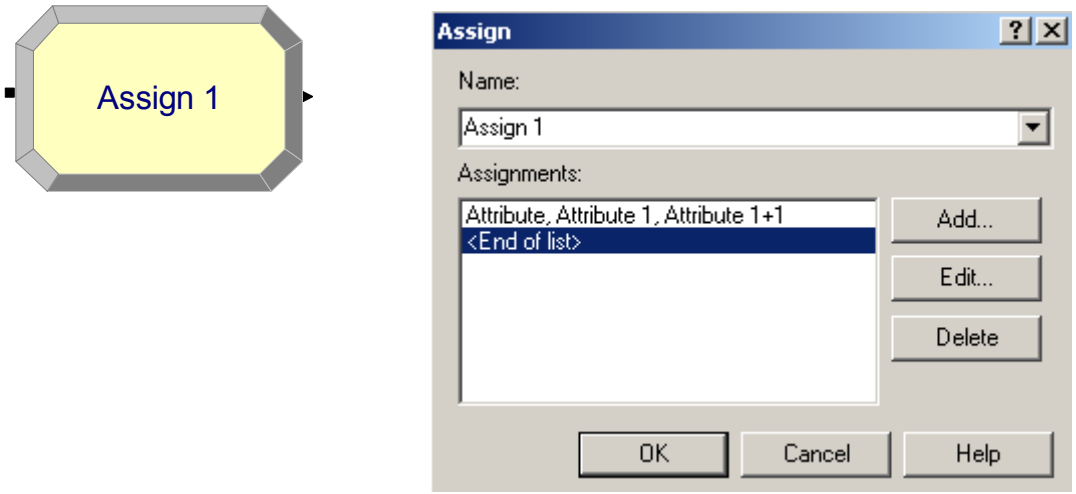
Для взлета и посадки самолеты занимают полосу ровно на 2 мин, Process 1 имитирует взлетно-посадочную полосу.



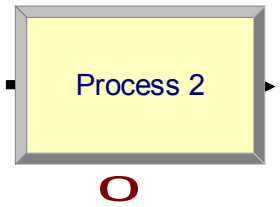
Самолеты прибывают для посадки в район крупного аэропорта каждые 10±5 мин.

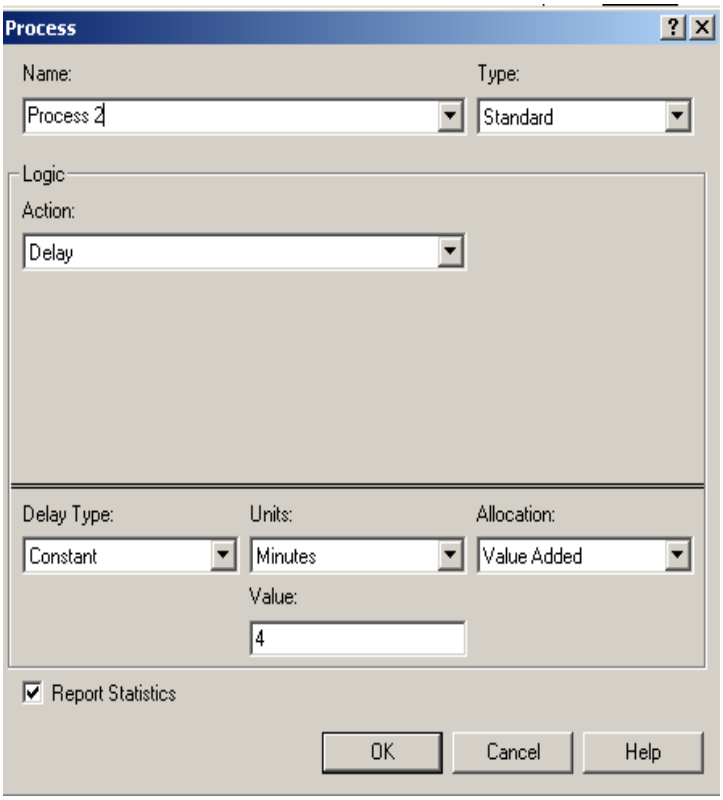


Если взлетно-посадочная полоса свободна, прибывший самолет получает разрешение на посадку и у них приоритет ниже, т. е. очередь в Hold 2 равна 0:
STATE(Resource 1) == IDLE_RES && NQ(Hold 2.Queue) == 0.

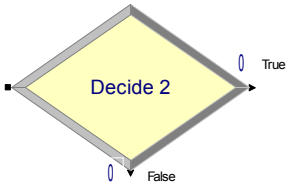


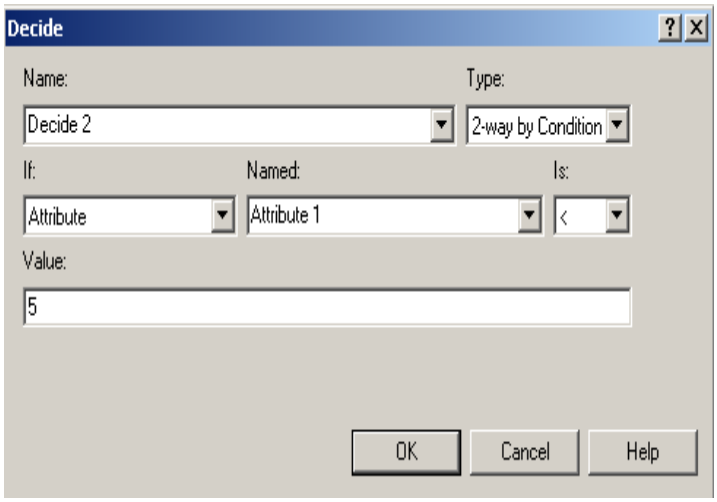
Этот модуль увеличивает Attribute 1+1, который моделирует количество кругов.



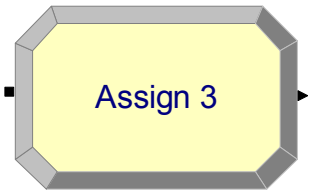


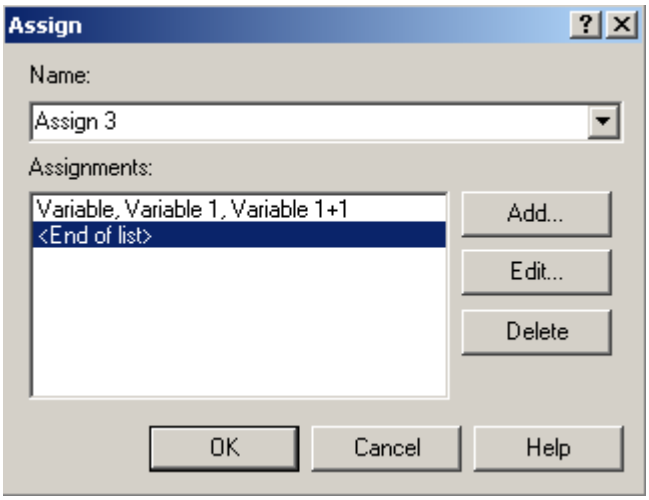
Если полоса занята, самолет выполняет полет по кругу и возвращается к аэропорту через каждые четыре минуты. Process 2 моделирует процесс совершения по кругу.





Этот модуль проверяет, сколько кругов сделал самолет: если 5, то он уходит на запасной аэропорт в Dispose 1.





Модули Assign 3, Assign 4 и Assign 5 аналогичны и необходимы для сбора статистики по взлетевшим, севшим самолетам и самолетам, ушедшим на запасной аэропорт:

– Variable 1 подсчитывает взлетевшие самолеты;
– Variable 2 подсчитывает севшие самолеты;
– Variable 3 подсчитывает самолеты, ушедшие на запасной аэродром.

Просмотреть значения переменных, полученных в результате моделирования, можно в стандартных отчетах, которые формируются в результате каждого прогона модели.

Time Persistent

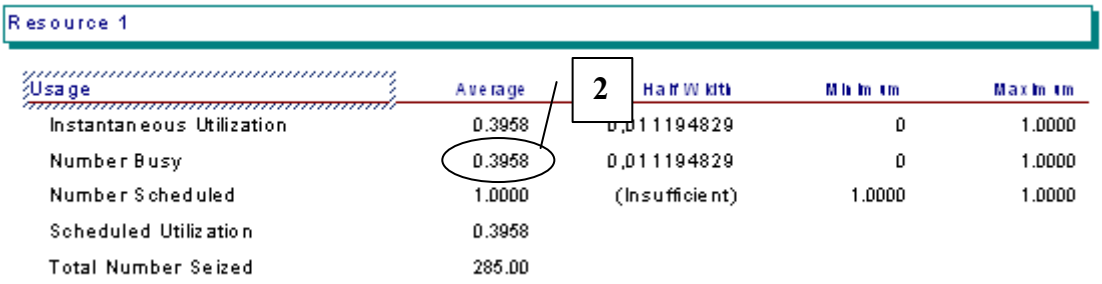
Variable	Average	Half Width	Minimum	Maximum
Variable 1	72.3690	(Insufficient)	0	144.00
Variable 2	69.3066	(Insufficient)	0	141.00
Variable 3	0.9783	(Insufficient)	0	1.0000

1

Таким образом, из отчета видно, что значение переменных следующее (см. сноска **1**):

– Variable 1 = 144;
– Variable 2 = 141;
– Variable 3 = 1.

Также в отчетах мы можем просмотреть загруженность полосы, которая у нас задана Resource 1.



Resource 1

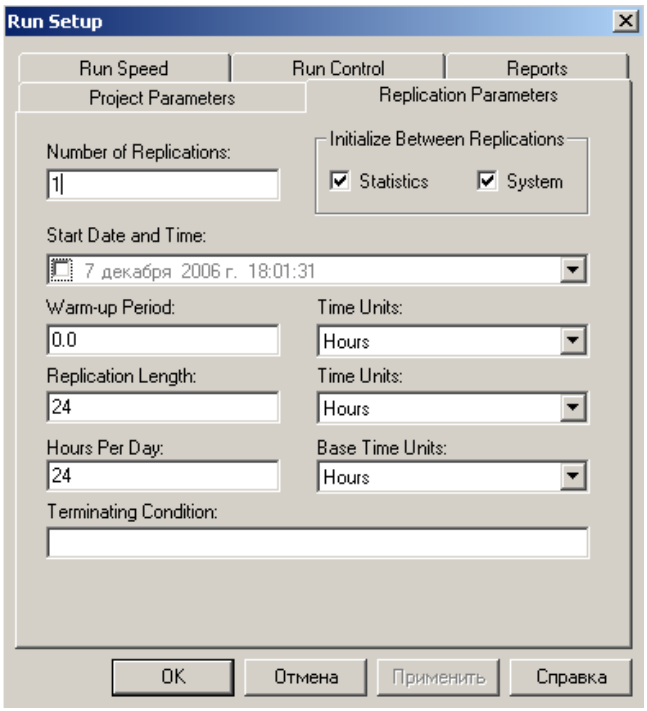
Usage	Average	Half Width	Minimum	Maximum
Instantaneous Utilization	0.3958	0,011194829	0	1.0000
Number Busy	0.3958	0,011194829	0	1.0000
Number Scheduled	1.0000	(Insufficient)	1.0000	1.0000
Scheduled Utilization	0.3958			
Total Number Seized	285.00			

2

Загруженность определяется параметром Number Busy и в нашем случае коэффициент загруженность взлетно-посадочной полосы равен 0,3958 (см. сноска **2**).

В этом примере, согласно заданию, необходимо было смоделировать работу аэропорта в течение 24 часов. Эта настройка задается в модели в окне Run/Setup/Replication Parameters.

В этих настройках мы длину репликации с бесконечности заменили на 24 часа.



Задание 2: Технологическая линия включает источник деталей, два взаимосвязанных станка, накопитель, технологический модуль для окончательной обработки деталей, рабочее место комплектации паллет и транспортировочный робот для их транспортировки на склад.

Время поступления деталей из источника распределено равномерно на интервале (10±2) сек., причем деталь поступает в минимальную из очередей к станкам.

Если деталь поступает на 1 станок, то затем она поступает на 2 станок. Если поступает сначала на 2, то потом на 1.

Время обработки деталей на станках распределено равномерно на интервале (10±4) сек., (9±3) сек. соответственно.

После цикла механообработки деталь поступает в накопитель по 10 деталей. Из накопителя все детали одновременно выдаются в технологический модуль для окончательной обработки (6±2) сек. Затем осуществляется укладка деталей в паллеты по 10 штук. Транспортировочный робот отбирает 2 паллеты и транспортирует их на склад. Время транспортировки распределено равномерно на интервале (16±4) сек.

Промоделировать 1 час работы. Определить

– количество обработанных деталей каждым станком,

– загрузку всех элементов технологической линии,

– максимальное, минимальное и среднее время наполнения накопителя после цикла механообработки.

Рассмотрим подробно логику реализованной на рис. 3.10 модели.

Модель состоит из двух параллельных цепочек:

1. Модуль Create 1 имитирует поступление деталей в систему, затем модуль Decide 1 решает, на каком из станов минимальная очередь на обработку, после чего деталь поступает на обработку либо на станок 1 (Process 1), либо на станок 2 (Process 2). После чего в модулях Assign 1 и Assign 2 маркируется, какое количество раз, деталь подвергалась механообработке (маркер – Attribute 1). Затем модули Decide 1 и Decide 2 определяют, требуется ли детали еще обработка. Если нет, то детали поступают в накопитель Batch 1, где формируются в комплект по 10 штук. Затем комплект разгруппировывается модулем Separate 5 и поступает на дальнейшую обработку в Process 3. После чего детали укладываются в паллеты по 10 штук, это имитировано модулем Batch 1, а готовые паллеты поступают на склад Hold 1 и ждут дальнейшей транспортировки роботом.

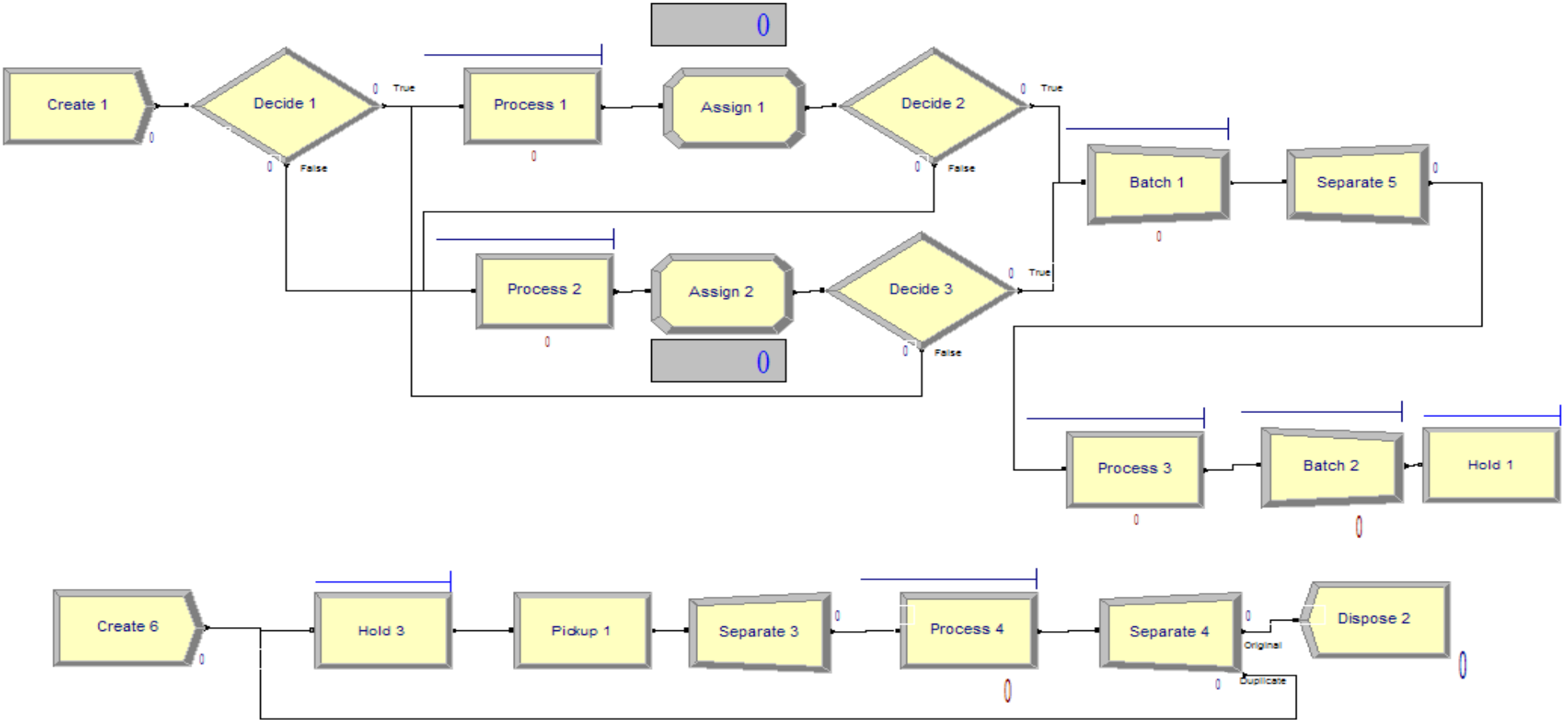
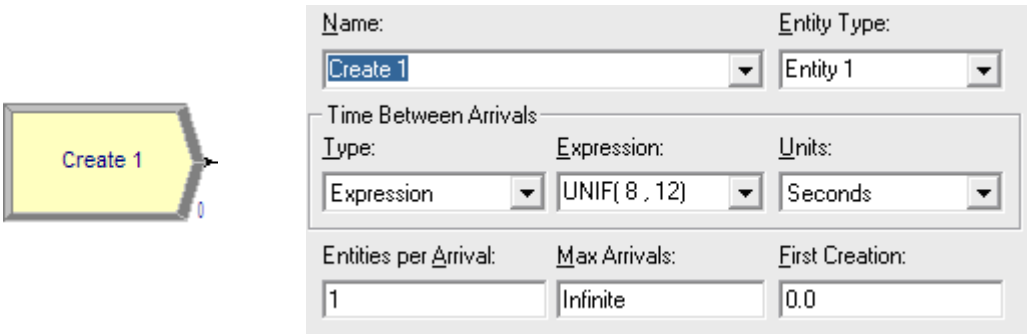


Рис. 3.10. Модель функционирования цеха механообработки в ПП Arena 7.0

2. Модуль Create 6 имитирует транспортировочного робота, который ждет в Hold 3, когда на складе Hold 1 накопится достаточное количество паллет (2 штуки). Когда имеется требуемое количество паллет, робот выезжает и забирает 2 паллету модулем Pickup 1. Process 4 моделирует переезд транспортировочного робота из одного пункта в другой. Затем Separate 4 разделяет делали и самого робота. Робот возвращается на место, где продолжает ожидать следующую партию паллет, а детали перемещаются в Dispose 2.

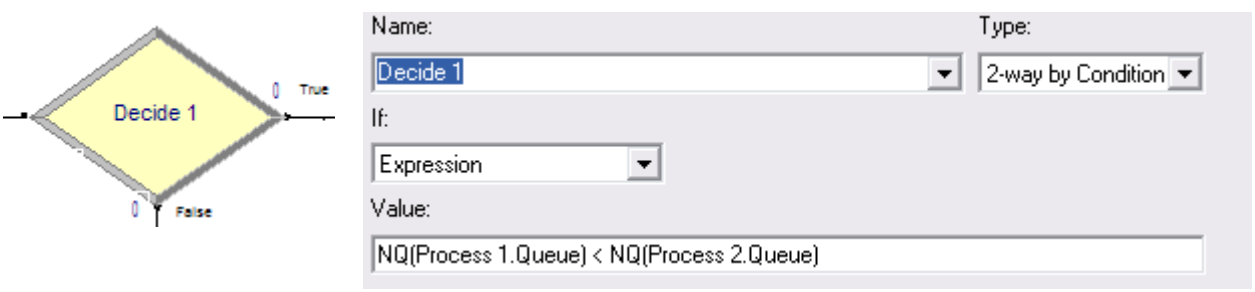
Рассмотрим более подробно наиболее интересные модули.

Поступление деталей из источника имитируется модулем Create 1, временные параметры которого равномерно распределены на интервале (10±2) сек. Модуль Create 1 вырабатывает сущности – детали.

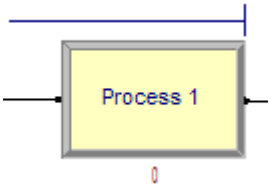


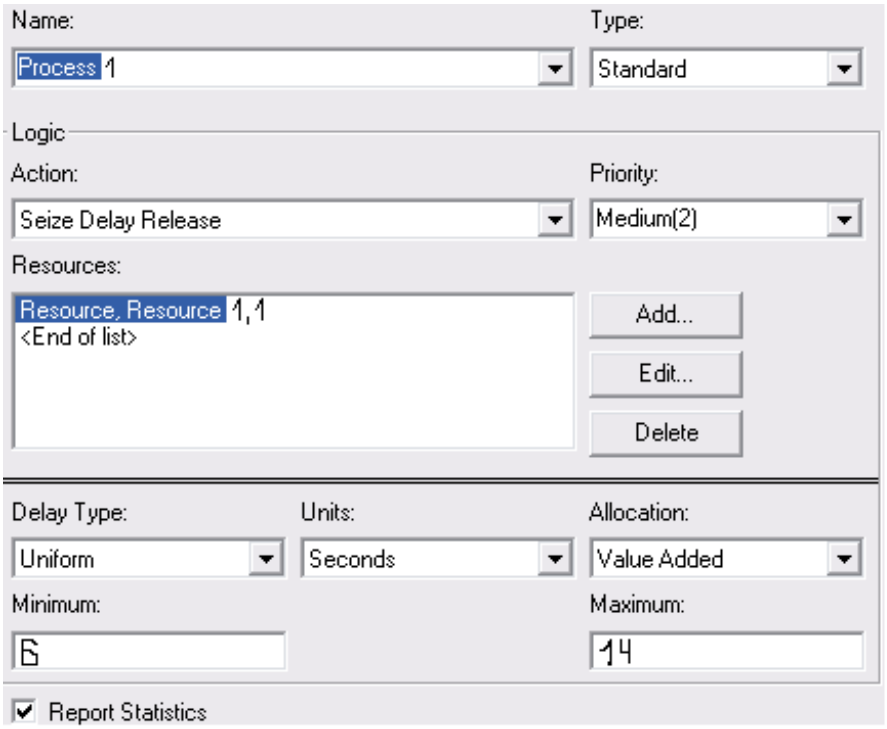
Определение условия, какова минимальная очередь, осуществляется в модуле Decide 1. При определении минимальной очереди используется оператор NQ(Process 1.Queue), подсчитывающий текущее количество сущностей в очереди к первому станку, который реализован в виде Process 1.

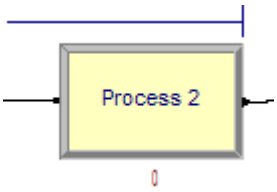
Модуль Decide1 проверяет, к какому станку очередь меньше, и направляет деталь к станку с минимальной очередью.

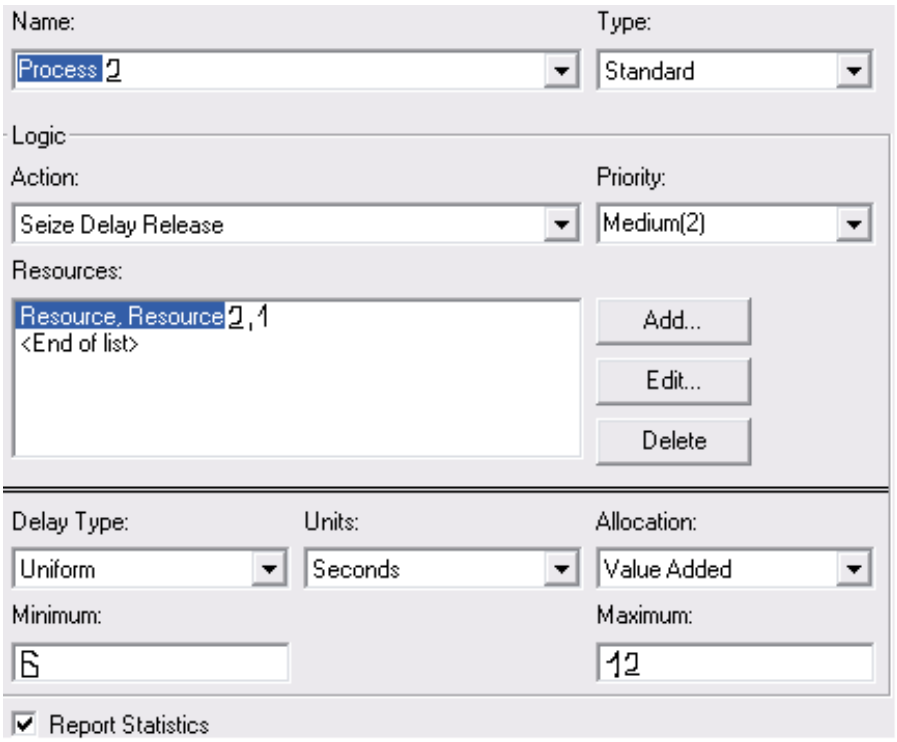


Обработка деталей 1 и 2 станком реализована в модулях Process 1 и Process 2. Время обработки деталей на станках распределено равномерно на интервале (10±4) сек., (9±3) сек. соответственно.



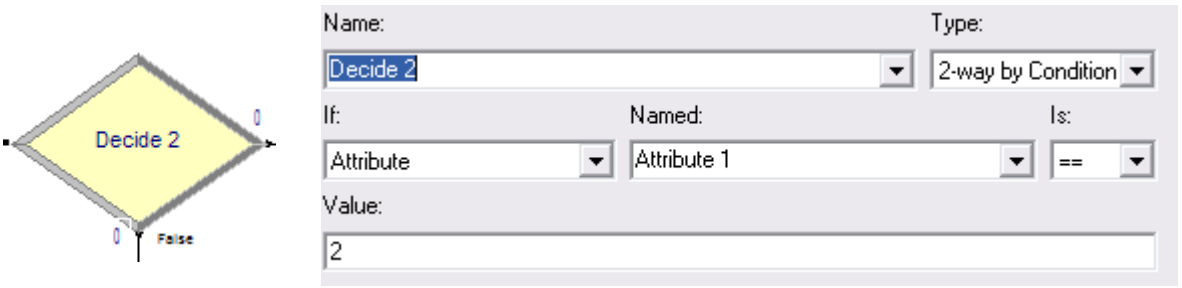


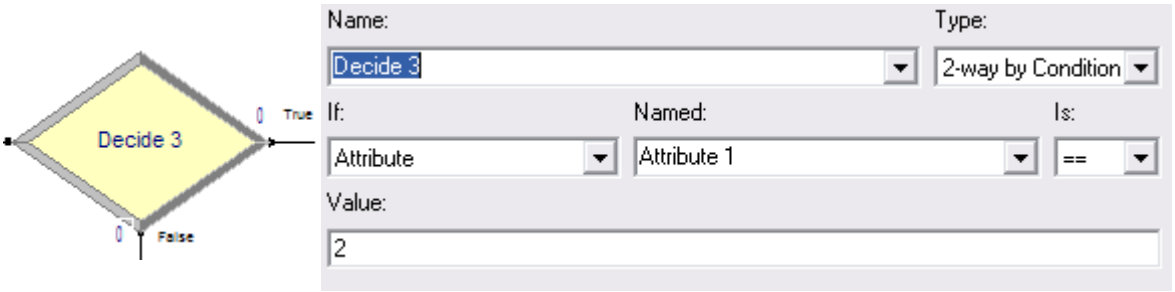




Проверка условия, что «если деталь поступает на 1 станок (модуль Process 1), то затем она поступает на 2 станок (модуль Process 2). Если поступает сначала на 2, то потом на 1», осуществляется при помощи модулей Decide 2 и Decide 3 в связке с Assign 1 и Assign 2.

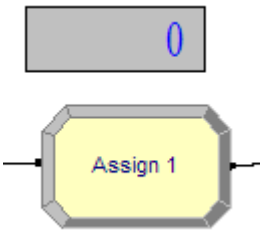
Модули Decide 2 и Decide 3 проверяют атрибут (Attribute 1), который отвечает за циклы механообработки деталей, вышедших после первого и второго станка соответственно. Если Attribute 1, задаваемый в Assign 1 и Assign 2, у детали равен двум, то деталь прошла два цикла механообработки и поступает в накопитель для дальнейшего соблюдения технологии процесса. Если Attribute 1 у детали не равен двум, то это означает, что деталь прошла всего один цикл механообработки и ее нужно перенаправить к другому станку.

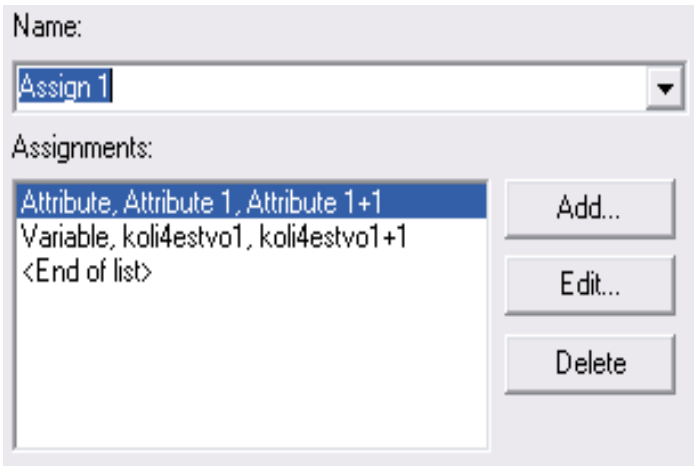


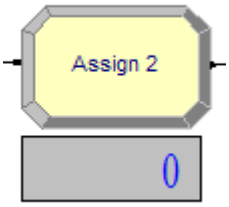


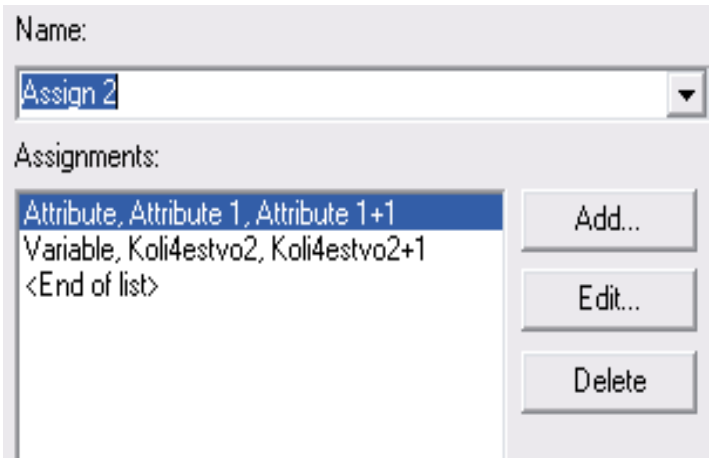
Модули Assign 1 и Assign 2 используются для задания атрибута у детали. В случае, когда деталь прошла очередной цикл механообработки Attribute 1 увеличивается на единицу.

Также в этих же модулях ведется подсчет количества обработанных деталей каждым станком, это необходимо для определения статистических параметров процесса. Подсчет обработанных деталей задается в виде переменных koli4estvo1 и koli4estvo2, путем увеличения этих переменных при прохождении сущности через модули Assign 1 и Assign 2.

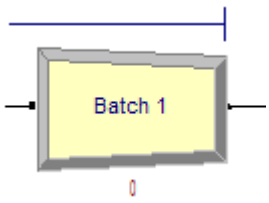


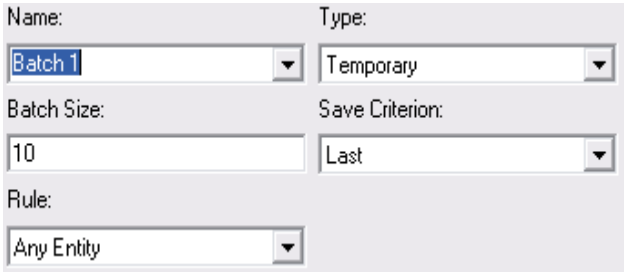


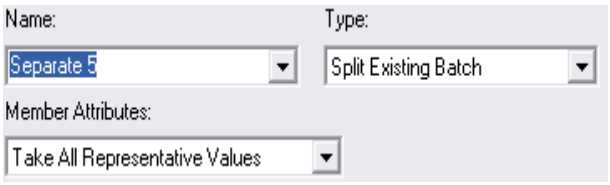




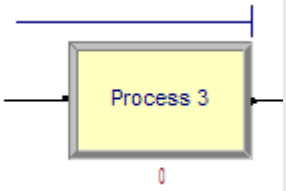
После цикла механообработки деталь поступает в накопитель (связка модулей Batch 1 – Separate 1), который вмещает 10 деталей. Batch 1 имитирует накопитель, где создается группировка из 10 деталей, а модуль Separate 5, в дальнейшем, разъединяет пришедшую группировку, для дальнейшей механообработки деталей по отдельности.

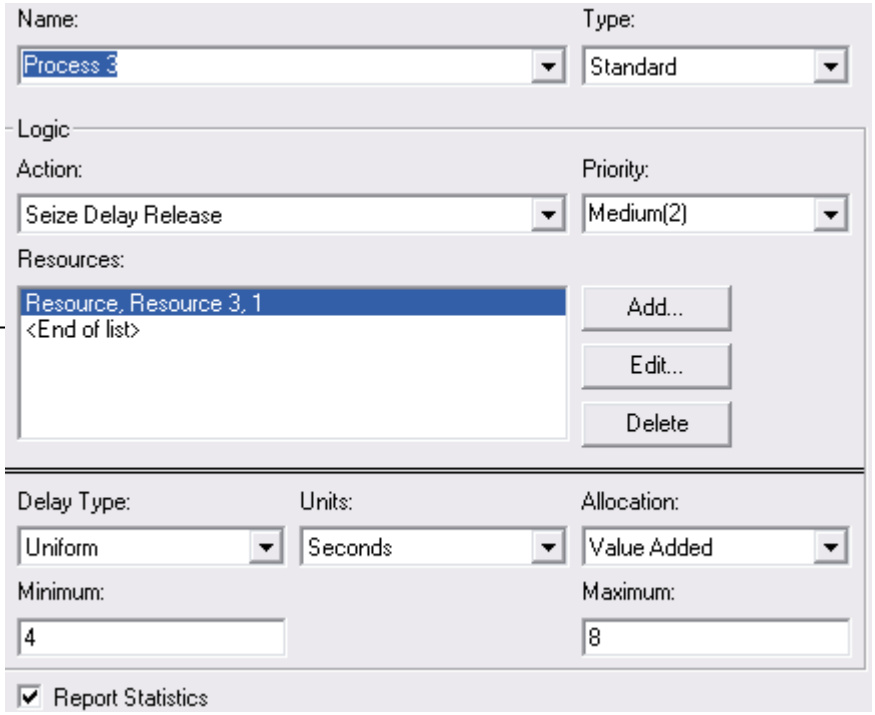




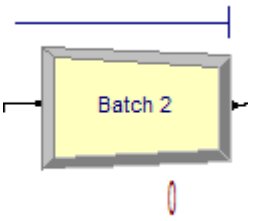


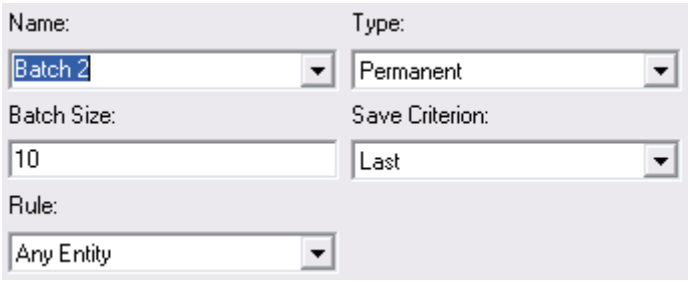
Из накопителя все детали подаются в технологический модуль Process 3 для окончательной обработки (6±2) сек. Process3 имитирует процесса окончательной обработки деталей (6±2) сек.



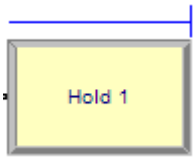


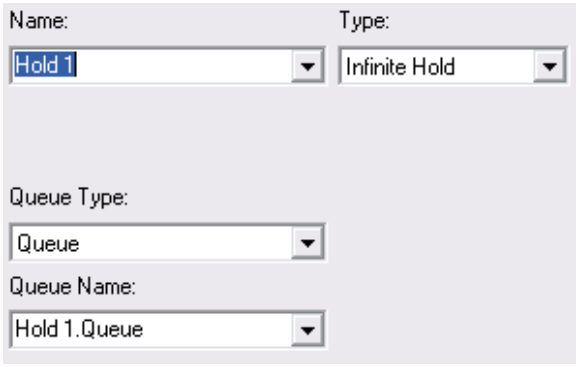
Затем осуществляется укладка деталей в паллеты по 10 штук. Это моделируется в виде модуля Batch 2, который создает комплект из 10 деталей.





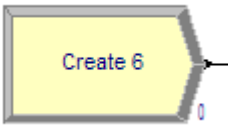
Затем комплект поступает в модуле Hold 1 для временного хранения. Тип модуля Hold 1 задаем Infinite Hold, т. е. бесконечное хранение, при этом параметре у данного модуля нет выхода, и сущности из него мы можем забрать, только используя другой дополнительный модуль, которым в нашем случае будет имитировать транспортировочный робот.

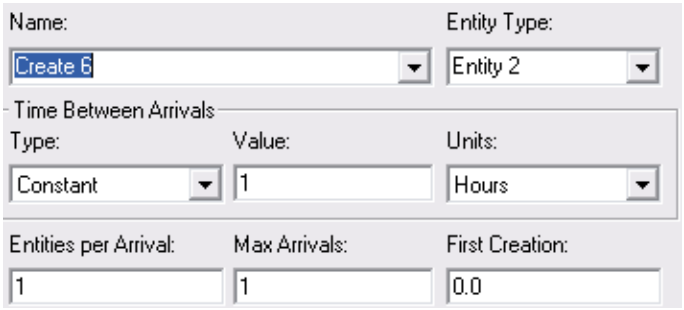




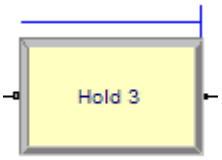
Имитация работы транспортировочного робота, представляет собой цепочку модулей, начиная от Create 6 и заканчивая Dispose 2.

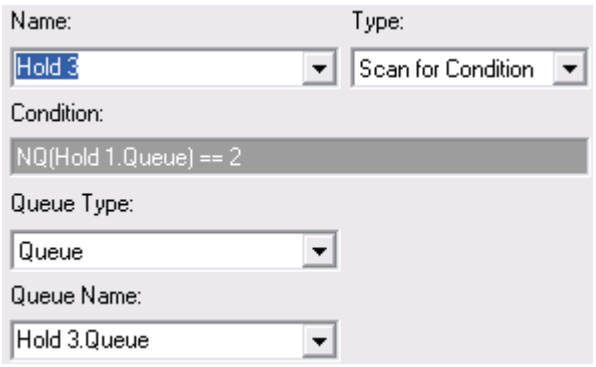
Модуль Create 6 создает всего одну сущность – транспортировочный робот.



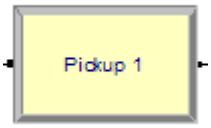


Эта сущность поступает в модуль Hold 3, где задерживается до тех пор, пока не выполнится условие NQ(Hold 1.Queue)==2, т. е. две паллеты готовы для транспортировки. Таким образом, модуль Hold 3 осуществляет управление движением транспортировочного робота.

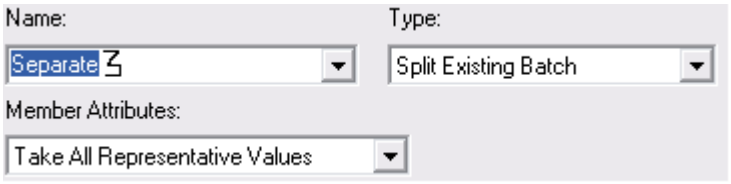




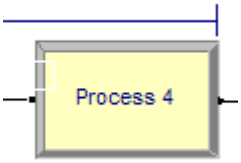
После того как в модуле Hold 1 есть две паллеты, готовые к транспортировки, транспортировочный робот забирает их модулем Pickup 1.

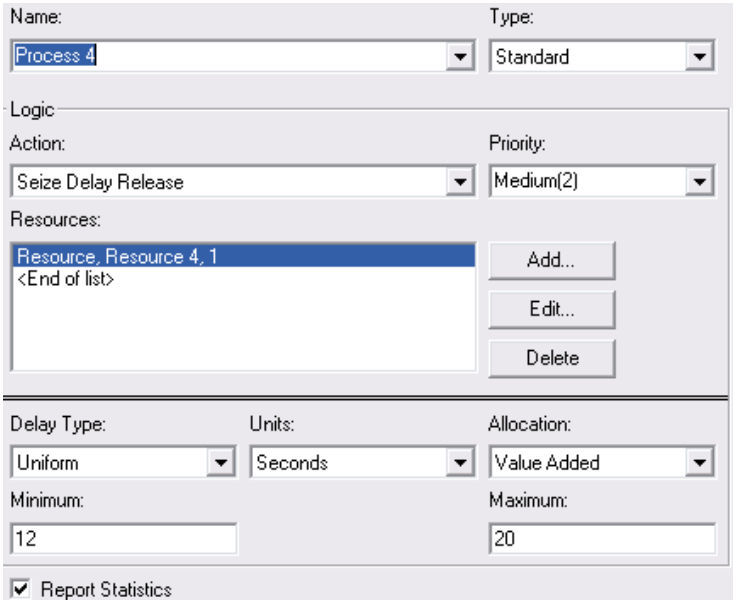


Модуль Separate 3 необходим по так называемым «техническим» причинам, никакой важной информационной нагрузки он не несет. Назначение этого модуля – разъединить существующую временную группировку, как известно, модуль Pickup 1, забирая сущности из очереди создает временную группировку, если эта группировка попадет в модуль Dispose 2, то появится сообщение об ошибке, именно для избежания этой ошибки мы добавляем в модель модуль Separate 3.

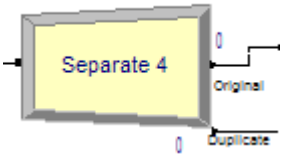


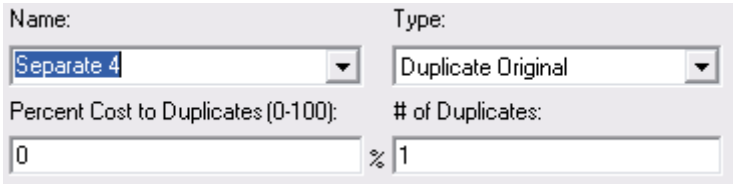
Время транспортировки деталей роботом распределено равномерно на интервале (16±4) сек. Модуль Process 4 используем для имитации времени процесса транспортировки.





Модуль Separate 4 используется для создания копии пришедшей в него сущности, что имитирует возвращение транспортировочного робота в модуль Hold 3, а перевезенных деталей в Dispose 2.





В результате моделирования 1 часа работы автоматической технологической линии (см. сноска **4**), как было задано в исходных данных, были получены следующие результаты:

1) количество обработанных деталей первым станком = 360 деталей, вторым станком = 362 деталь, эти данные можно взять из стандартных отчетов (см. сноска **1**);

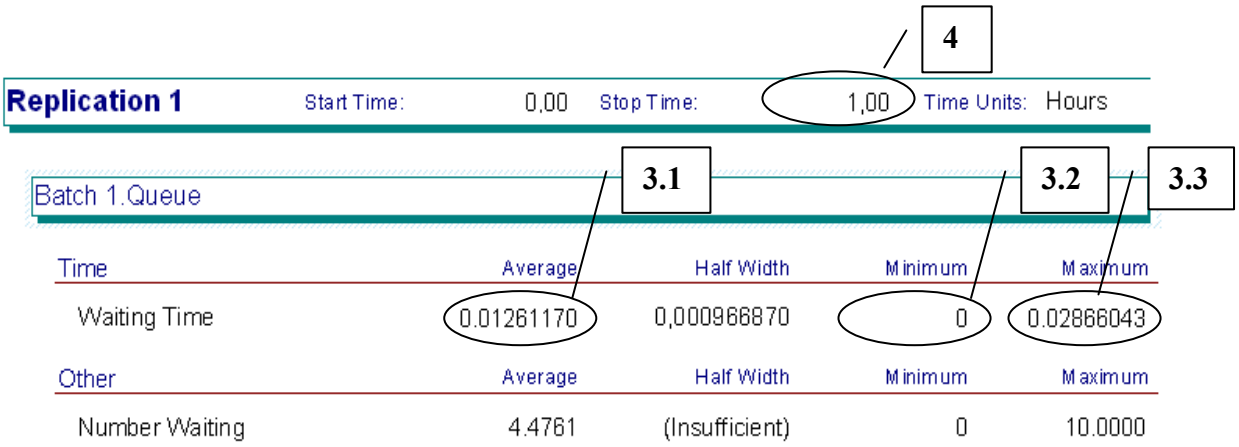
2) загруженность станков можно взять из стандартных отчетов, коэффициент загруженности первого станка (Resource 1) = 0,99; второго станка (Resource 2) = 0,90 (см. сноска **2**);

Usage

2 **1**

	Inst Util	Num Busy	Num Sched	Num Seized	Sched Util
Resource 1	0,99	0,99	1,00	360,00	0,99
Resource 2	0,90	0,90	1,00	362,00	0,90
Resource 3	0,58	0,58	1,00	350,00	0,58
Resource 4	0,15	0,15	1,00	34,00	0,15

3) максимальное, минимальное и среднее время наполнения накопителя (в нашем случае, это было смоделировано модулем Batch 1) после цикла механообработки также можно взять из стандартных отчетов, предлагаемых программным пакетом Arena 7.0.: среднее время ожидания деталей в накопителе = 0,013 часа (см. сноска **3.1**); минимальное время ожидания = 0 (см. сноска **3.2**); максимальное время ожидания = 0,029 часа (см. сноска **3.3**).



3.11. Задачи к главе 3

<u>Задание № 1</u>. Работа парикмахерской

В парикмахерскую могут приходить клиенты двух типов. Клиенты первого типа желают только стричься. Распределение интервалов их прихода 35±10 мин. Клиенты второго типа желают постричься и побриться. Распределение интервалов их прихода 60±20 мин. Парикмахер обслуживает клиентов в порядке «первым пришел – первым обслужен». На стрижку уходит 18±6 мин., а на бритье 10±2 мин.

Доходы от работы парикмахерской определяются количеством клиентов, обслуженных в течение рабочего дня (9 часов с часовым перерывом на обед стоимость стрижки 100 рублей, бритья 20 рублей), убытки определяются временем простоев парикмахера (в отсутствие клиентов) и количеством необслуженных клиентов в очереди.

Моделирование проведите для рабочей недели (6 дней по 8 часов).

После разработки модели, согласно заданию, внесите в нее следующие дополнения и/или изменения:

1. Клиенты первого типа имеют анимационную картинку «Woman» (в виде женщины), а клиенты второго типа – «Man».
2. Задайте анимацию ресурсу «Парикмахер», когда он свободен (Idle) Рис. 3.11 *а*, и когда он занят (Busy) Рис. 3.11 *б*.
3. Измените правило обслуживания: приоритет в обслуживании имеют женщины (клиенты первого типа).
4. Рассмотрите возможность ввода в модель второго парикмахера. Как измениться доход парикмахерской?

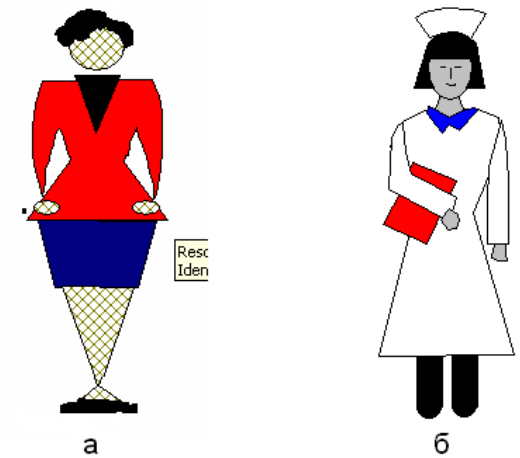


Рис. 3.11. Анимационная картинка ресурса «Парикмахер»:
а – ресурс свободен; *б* – ресурс занят

<u>Задание № 2</u>. Работа сборочного цеха

В сборочный цех поступают детали трех видов. Детали первого типа (Д1) поступают 20±3 мин (наиболее часто 20 мин). Детали второго типа (Д2) – 16±5 мин. Детали третьего типа (Д3) – 20 мин. Как только сборщику поступают три детали (любые), он производит монтаж готового изделия за 5 мин. Из собранных изделий 15 % бракованные. Если изделие бракуется в первый раз, то оно поступает на повторный монтаж к сборщику. Если изделия бракуются 2 раза, то они идут в отходы (10 мин). Не бракованные изделия упаковываются по 5 штук за 3 мин. упаковщиком.

Смоделировать 8 часовой рабочий день.

Построить модель согласно заданию и выполнить следующие задания:

1. Определить каждому типу деталей свою анимационную картинку.
2. Определить анимационную картинку готовому изделию и упакованному изделию.
3. Задать анимационную картинку ресурсам «Сборщик» и «Упаковщик», когда они свободны и заняты.
4. Собрать статистику по бракованным изделиям (отходы и один раз бракованные), количеству упаковок, по загруженности ресурсов «Сборщик» и «Упаковщик».
5. Изменить модель следующим образом: сборщик собирает изделие из деталей разного типа, и готовые не бракованные изделия складируются. Один раз в 10 часов из гаража выезжает грузовик и забирает со склада все упаковки.

Задание № 3. Работа системы сбора информации

Распределенный банк данных системы сбора информации организован на базе ЭВМ, соединенных дуплексным каналом связи. Поступающий запрос обрабатывается на первой ЭВМ и с вероятностью 50 % необходимая информация обнаруживается на месте. В противном случае необходима посылка запроса во вторую ЭВМ.

Запросы поступают через 10 ± 3 с., первичная обработка запроса занимает 2 с., выдача ответа требует 18 ± 2 с., передача по каналу связи занимает 3 с. Временные характеристики второй ЭВМ аналогичны первой.

Смоделировать прохождение 400 запросов.

Определить необходимую емкость накопителей перед ЭВМ, обеспечивающую безотказную работу системы, и функцию распределения времени обслуживания заявки.

Построить модель согласно заданию и выполнить следующие задания:

1. В систему первоначально поступают сущности в виде дискет, а затем преобразовываются в самолеты и лодки.
2. Первые 200 запросов идут по ветке True, остальные по False.
3. Первые 30 мин. все запросы шли по True, остальные по False.
4. Первые 200 запросов проходили первичную обработку 2 с., остальные 4 с.
5. Запросы моделируются с разными приоритетами, в модуле условия с большим приоритетом по True, с меньшим по False.

Задание № 4

В компанию поступают запросы 20 ± 4 мин. Поступающий запрос обрабатывается двумя сотрудниками, причем первый сотрудник обрабатывает 75 % запросов, второй обрабатывает остальные запросы. Первичная обработка запроса занимает 23 мин., выдача ответа требует 18 ± 5 мин., как у первого, так и у второго сотрудника.

Смоделировать прохождение 350 запросов.

Построить модель согласно заданию и выполнить следующие пункты:

1. Определить количество запросов, обработанных каждым сотрудником за 24 часа.
2. В систему первоначально поступают сущности в виде телефонных звонков, а затем к первому сотруднику приходят в виде отчетов, а ко второму сотруднику в виде дискет.
3. Измените модель: Первые 50 запросов идут к первому сотруднику на обработку, остальные ко второму.
4. Измените модель: Первые 4 часа все запросы идут ко второму сотруднику на обработку, остальные к первому.
5. Измените модель: Первые 150 запросов проходят первичную обработку 23 мин., остальные 30 мин.
6. Измените модель: На обработку поступают 2 вида запросов (телефонные звонки и письма). Причем при первичной обработке у телефонных звонков приоритет выше, чем у писем.
7. Создайте анимационные картинки ресурсам, когда они свободны и заняты. Первый сотрудник должен быть изображен также

как на Рис. 3.12 (*а* – свободен, *б* – занят). Второй сотрудник должен быть изображен также как на Рис. 3.13 (*а* – свободен, *б* – занят).

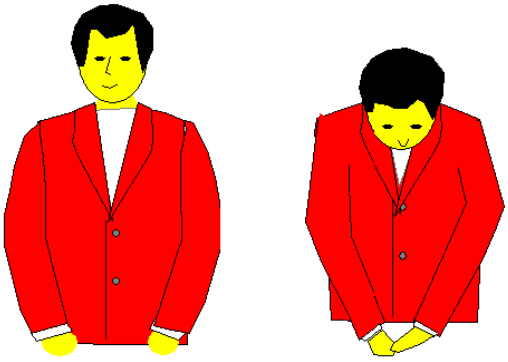


Рис. 3.12. Анимационная картинка ресурса «Сотрудник 1»:
а – ресурс свободен; *б* – ресурс занят

Рис. 3.13. Анимационная картинка ресурса «Сотрудник 2»:
а – ресурс свободен; *б* – ресурс занят

Задание № 5

В слот бар приходят клиенты. В игральный автомат, типа «однорукий бандит», каждые 5 – 10 минут опускается монета номиналом 5 рублей. Автомат, случайным образом в течение 10 секунд, выдает три цифры от 0 до 9.

В случае совпадения всех трех цифр, игрок выигрывает 50 монет (по 5 рублей), в случае выпадения любой другой комбинации — монета игрока уходит в доход казино. Принятые монеты автомат упаковывает в пачки по 10 штук в каждой.

В случае выигрыша игрок опускает в автомат дополнительно от 10 до 15 монет, на каждую монету он тратит 20 секунд.

Смоделировать работу автомата в течение 24 часов. Определить сумму денег выигранных игроками и чистую прибыль казино, в рублях.

Задание № 6

Создать модель полета рейсовых самолетов.

Клиенты, желающие приобрести билет на самолет, приходят в кассу аэропорта в среднем через 20±5, чаще 10 минут, причем 25 % из них приобретают билеты в первый класс, 70 % - во второй класс, а остальные вообще отказываются приобретать билеты и уходят.

Время вылета самолета определяется его полной загрузкой, т.е. самолет вылетит только при наличии 10 пассажиров первого класса и 20 пассажиров второго класса.

Самолеты прибывают в аэропорт в среднем раз в 6-12 часов, максимальное количество самолетов = 20.

Время полета занимает в среднем (5±3) часов, чаще 6 часов. По прилету пассажиров отвозят в здание аэропорта, а самолет на техническое обслуживание.

Задание № 7

Участок ремонта кузовов автомобилей состоит из двух рабочих мест: первое рабочее место – это кузовной ремонт автомобиля, второе рабочее место – окраска кузова. После восстановления кузова автомобили поступают в окрасочную камеру.

Время поступления на ремонт поврежденных автомобилей первой модели – случайная величина, равномерно распределенная на интервале от 1 до 6 часов, второй модели – от 1 до 2 часов.

На кузовной ремонт автомобилей первой модели тратится от 1 до 3 часов, второй модели – от 2 до 5 часов.

Время окраски любого автомобиля равномерно распределено на интервале (15 – 20) минут.

Модели первого типа при обслуживании имеют более высокий приоритет.

В случае, если ремонтная мастерская и покрасочная камера заняты, автомобили дожидаются обслуживания в очередях, длины которых не ограничены.

За 12 часов оценить отдельно для 1 и 2 модели:

- среднее время, которое тратится на ремонт автомобилей,
- среднее время ожидания в очередях,
- количество отремонтированных автомобилей,
- максимальный размер очереди «ожидания» начала обслуживания и очереди перед операцией окраски.

Проанализировать зависимость приведенных выше характеристик при изменении их числовых значений.

Сделать анимацию в модели:
1. Модели 1 типа – грузовики (Entity Picture = Track); модели 2 типа – минивэны (Entity Picture = Van).
2. Первое рабочее место имеет анимационную картинку, приведенную на Рис. 3.14, второе рабочее место на Рис. 3.15.

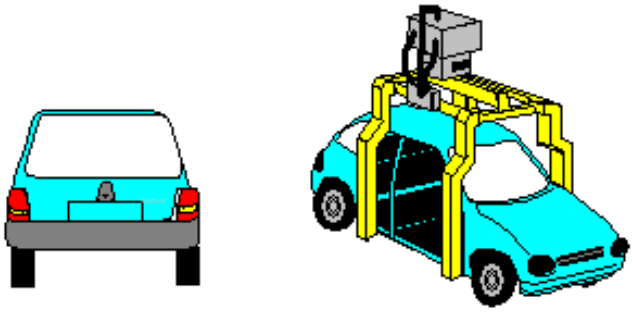


Рис. 3.14. Анимационная картинка ресурса «Первое рабочее место»:
а – ресурс свободен; *б* – ресурс занят

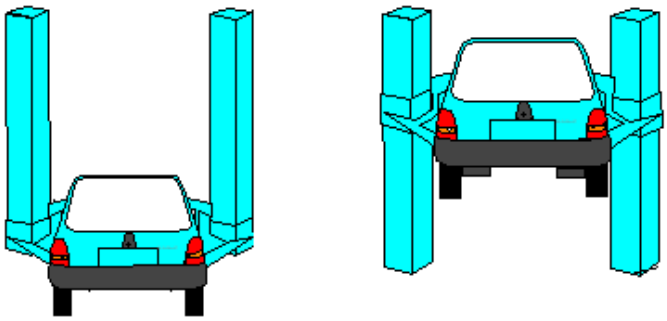


Рис. 3.15. Анимационная картинка ресурса «Второе рабочее место»:
а – ресурс свободен; *б* – ресурс занят

Задание № 8

В магазин за покупками приходят клиенты. Для работников магазина клиенты классифицируются на постоянных и обычных. Продавцы (менеджеры) затрачивают в среднем 2 минуты на человека для разъяснения информации по товарам и ответа на вопросы. Приоритетное право на обслуживание без очереди имеют постоянные клиенты. 25 % посетителей уходят без покупок, а остальные встают в очередь в кассу. Кассир один, он обслуживает из очереди постоянных клиентов, а потом обычных посетителей, причем как только кассир рассчитал одного клиента, он сразу же обслуживает следующего, время обслуживания клиента занимает 5 минут.

Постоянные клиенты в основном приходят с утра с 9 до 11 часов и в конце рабочего дня с 16 до 18 часов, а обычные посетители в основном в середине дня.

Построить график, отображающий уровень посещаемости магазина покупателями, и график загруженности кассира.

Задание № 9

Люди приносят на почту письма, которые могут быть двух видов: заказные и обычные. Затем почтовые работники их обрабатывают.

Заказные письма поступают круглосуточно раз в 5-20 минут, а простые письма принимаются только с 8.00 до 20.00 (8.00-12.00 их количество увеличивается от 50 до 120, наибольшее их количество (260) поступает между 12.00 и 14.00, а затем их количество плавно убывает от 180 до 70).

Оба вида писем обрабатываются одним работником почтовой службы, причем заказные письма обрабатываются вне очереди, т.к. они важнее. Время обработки заказных писем 1-3 минуты, а время обработки простых писем – 3-5, чаще 4 минуты.

Затем все эти письма поступают в отдел подготовки к отправлению, где заказные письма обслуживаются также вне очереди. Время подготовки писем к отправке 10-15 минут.

Создать анимацию работы сотрудников почты и отразить процесс обработки простых писем на гистограмме.

Задание № 10

На участке термической обработки выполняются цементация и закаливание шестерен, поступающих через 10 ± 5 мин.

Цементация занимает 10 ± 7 мин., а закаливание – 10 ± 6 мин. Качество определяется суммарным временем обработки.

Шестерни со временем обработки

- больше 25 мин покидают участок,
- от 20 до 25 мин передаются на повторную закалку,
- меньше 20 мин должны пройти повторную полную обработку.

Детали с суммарным временем обработки меньше 20 мин считаются вторым сортом.

Смоделировать процесс обработки на участке 400 шестерен.

Определить

- количество обработанных деталей,
- число повторений полной и частичной обработки.

Задание № 11 (повышенной сложности)

Рассматривается работа столовой самообслуживания. Обеды выдают 3 повара. Количество мест за столами всегда достаточно для размещения лиц, уже получивших обед.

Длины временных промежутков между прибытиями посетителей в столовую распределены по равномерному закону на интервале (0 – 20) мин. Время обслуживания на одного посетителя описывается равномерно распределенной величиной на интервале (1 –2) мин.

Условия работы столовой таковы, что в очереди могут одновременно стоять не более 40 человек. Посетитель стоит в очереди 30 мин., после чего он покидает столовую.

На обед посетитель затрачивает время, которое распределено равномерно на интервале (15 – 25) мин.

В течение 4 часов оценить:

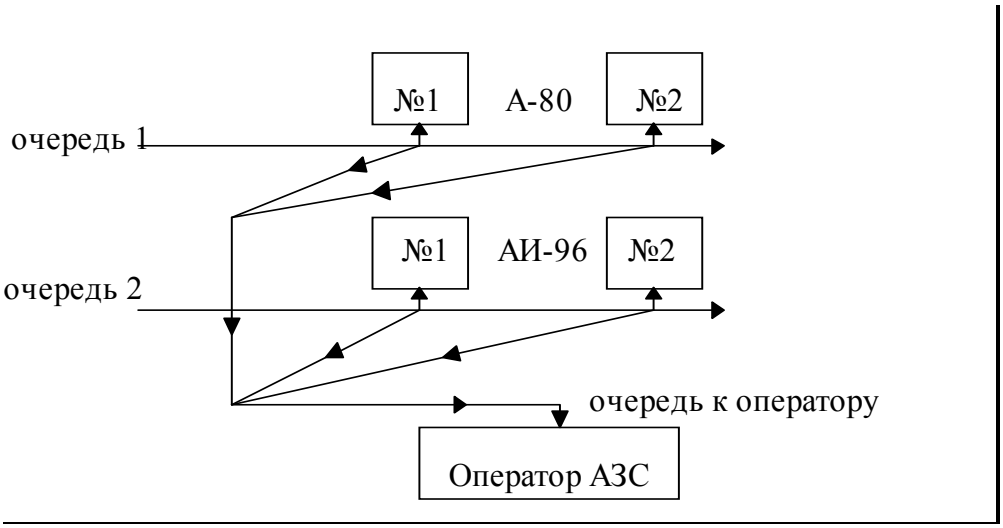
- сколько в среднем требуется посетителю времени на посещение столовой,
- среднее время, которое посетитель проводит в очереди,
- среднее число одновременно обедающих, их максимальное и минимальное число,
- количество посетителей, покинувших столовую.

Самостоятельно провести исследования каких-либо других характеристик функционирования данной модели и их зависимостей между собой.

Задание № 12 (повышенной сложности)

На автозаправочной станции (АЗС) осуществляется заправка автомобилей бензином марок А-80 и АИ-96. На каждый вид топлива работает по две заправочных колонки, стоящих друг за другом, которые подают бензин со скоростью 1 литр/сек. Автомобили заезжают на АЗС каждые (45±15) сек., причем 2 из 3-х автомобилей заправляется топливом марки АИ-96 и только 1/3 автомобилей топливом А-80. Автолюбители покупают от 5 до 75 литров бензина, причем заказываемый объем кратен 5.

Заказы от клиентов принимает один оператор АЗС, который осуществляет обслуживание одного клиента за (30±15) сек.



Исследование разработанной модели проводить в следующих направлениях:

Смоделировать 10 часов работы АЗС, определить:

- максимальную длину очередей 1, 2 и очереди к оператору АЗС;
- суммарное количество проданного топлива каждого вида и количество топлива каждого вида, проданное колонками № 1 и № 2;
- суммарное время простоя каждой из колонок;

Выяснить, сколько раз за смену (10 часов) в каждой из очередей возникает задержка в обслуживании в случае, когда автомобиль заправился из колонки № 1, а перед ним еще обслуживается из колонки № 2 другой автомобиль. Определить суммарное время таких задержек.

Выявить параметры, наиболее сильно влияющие на эффективность обслуживания автолюбителей. Сделать выводы о целесообразности использования пары колонок в каждой из линий.

Задание № 13 (повышенной сложности)

Вычислительная система включает три ЭВМ. В систему в среднем через 30с поступают задания, которые попадают в очередь на обработку к первой ЭВМ, где они обрабатываются около 30с. После этого задание поступает одновременно во вторую и третью ЭВМ.

Вторая ЭВМ может обработать задание за (14 ± 5) с., а третья – за (16 ±1) с. Окончание обработки задания на любой ЭВМ означает снятие ее с решения с той и другой машины. В свободное время вторая и третья ЭВМ заняты обработкой фоновых задач.

Задание № 14 (повышенной сложности)

Магистраль передачи данных состоит из двух каналов (основного и резервного) и общего накопителя. При нормальной работе сообщения передаются по основному каналу за 7 ± 3 с. В основном канале происходят сбои через интервалы времени 200 ± 35 с. Если сбой происходит во время передачи, то за 2 с запускается запасной канал, который передает прерванное сообщение с самого начала. Восстановление основного канала занимает 23 ± 7 с. После восстановления резервный канал выключается и основной канал продолжает работу с очередного сообщения. Сообщения поступают через 9 ± 4 с. и остаются в накопителе до окончания передачи. В случае сбоя передаваемое сообщение передается повторно по запасному каналу.

Смоделировать работу магистрали передачи данных в течение 1 ч. Определить загрузку запасного канала, частоту отказов канала и число прерванных сообщений. Определить функцию распределения времени передачи сообщений по магистрали.

Задание № 15 (повышенной сложности)

Портовый грузооборот связан с заливкой танкеров сырой нефтью для ее дальнейшей транспортировки. Имеется возможность заливать одновременно до трех танкеров. Танкеры, прибывающие в порт каждые 11±7 часов, могут быть одного из трех различных типов. Относительная частота прихода танкеров различных типов и требуемое время для их заливки приведены в таблице.

Тип танкера	Относительная частота	Время заливки (часы)
1	0,25	18±2
2	0,55	24±3
3	0,20	36±4

Прибывшему танкеру любого типа для подхода к стоянке и отхода от нее требуются услуги буксира. В порту имеется один буксир, который транспортирует танкер в один конец примерно за 1 час.

В этой части океана часто штормит, а в период шторма для танкера невозможен ни подход к стоянке, ни отход от нее. Продолжительность шторма 4 ± 2 часа, время между окончанием шторма и началом следующего подчиняется экспоненциальному распределению со средним значением в 48 часов.

3.12. Вопросы к главе 3

1. Перечислите основные достоинства ПП имитационного моделирования Arena 7.0.
2. Какие основные панели используются в ПП Arena 7.0 для моделирования процессов и систем?
3. На какие типы подразделяются модули в строительных панелях?
4. Перечислите, с помощью каких модулей можно забрать (освободить) сущности из модуля Hold, если тип задержания в модуле Infinity Hold?
5. Для чего необходим модуль Match, и в связке с каким модулем он обычно работает?
6. Приведите пример использования модуля Dropoff?
7. Какие параметры необходимо задать модулю Process, чтобы появилась очередь?
8. Поясните, каким образом можно смоделировать, чтобы модуль Process мог обрабатывать по 5 сущностей, а только 6 и последующие становились в очередь?
9. Объясните, в чем разница типов Split existing batch и Duplicate Original модуля Separate?
10. Что такое Resource, и что значит его параметр Capacity?

Заключение

В настоящее время компьютерное моделирование и анализ данных являются широко используемыми инструментами, применяющимися в науке, но хотелось, чтобы компьютерное моделирование более активно внедрялось и на реальных предприятиях и производствах. Общеизвестно, что компьютерные эксперименты гораздо дешевле, чем реальные действия с людьми и оборудованием. В связи с этим, дисциплина «Компьютерное моделирование» должна входить не только в рабочие программы специальностей, связанных с информационными технологиями, а также для студентов других технических специальностей.

В этом учебном пособии были изложены основы теории моделирования систем, рассмотрены основные понятия, приведены различные классификации систем. Вторая глава пособия посвящена наиболее часто использующимся при моделировании бизнес-процессов предприятия структурным моделям и методологиям, позволяющим разрабатывать структурные модели: IDEF0, IDEF3 и DFD. Отдельное внимание уделено концепции ARIS, которая широко применяется в последние годы при описании и проектирование бизнес-процессов и информационных систем.

Во второй главе раскрыты понятия имитационного моделирования процессов и систем. Рассмотрено современное программное средство имитационного моделирования, в основы которого заложены два наиболее распространенных математических аппарата сети Петри и системы массового обслуживания.

Автор понимает, что в настоящее время в высших учебных заведениях при обучении студентов используют различные программные средства моделирования. Но использование пакета Arena является перспективным и апробировано на ряде зарубежных предприятий различных отраслей экономики. Тем более, что основа всех средств моделирования одна (сети Петри и СМО).

Список использованных источников

1. Бахвалов Л. А. Компьютерное моделирование: долгий путь к сияющим вершинам [Электронный ресурс]. – Режим доступа: http://www.gpss-forum.narod.ru/GPSSmodeling.html, свободный.
2. Бешенков С. А. Моделирование и формализация: методическое пособие. – М.: Лаборатория базовых знаний, 2002.
3. Большаков А. С. Моделирование в менеджменте: учеб. пособие. – М.: Филинъ, 2000.
4. Бусленко Н. П. Моделирование сложных систем. – М.: Наука, 1978.
5. Бычков С. П., Храмов А. А. Разработка моделей в системе моделирования GPSS: учеб. пособие. – М.: МИФИ, 1997.
6. Введение в математическое моделирование: учеб. пособие / под ред. П. В. Трусова. – М.: Интермет инжиниринг, 2000.
7. Вендров А. М. CASE-технологии. Современные методы и средства проектирования информационных систем. – М.: Финансы и статистика, 1998.
8. Волчков С., Балахонова И. Бизнес-моделирование для совершенствования деятельности промышленного предприятия // ЦИТ «Платон» "КомпьютерПресс". 2001. №11.
9. Докукин В. П. Основы математического моделирования: конспект лекций. Санкт-Петербургский ГГИ. – М.: Дело, 2000.
10. Имитационное моделирование производственных систем / под ред. А. А. Вавилова. – М.: Машиностроение, 1983.
11. Калашников В. В., Рачев С. Т. Математические методы построения стохастических моделей обслуживания. – М.: Наука, 1988.
12. Киндлер Е. Языки моделирования. – М.: Энергия, 1985.
13. Клейнен Дж. Статистические методы в имитационном моделировании. – М.: Статистика, 1978.
14. Лоу А. М., Кельтон В. Д. Имитационное моделирование. Классика CS. – 3-е изд. – СПб.: Питер; Киев: Издательская группа BHV, 2004. –847 с.: ил.
15. Лукасевич И. Я. Имитационное моделирование инвестиционных рисков [Электронный ресурс]. – Режим доступа: http://www.cfin.ru/finanalysis/imitation_model-2-1.shtml, свободный.
16. Марков А. А. Моделирование информационно- вычислительных процессов: учеб. пособие. – М.: Изд-во МГТУ им. Н. Э.Баумана, 1999.
17. Максимей И. В. Имитационное моделирование на ЭВМ. – М.: Радио и связь, 1988.

18. Марка Д. А., МакГоуэн К. Методология структурного анализа и проектирования. – М.: МетаТехнология, 1993.

19. Математическое моделирование: методы, описания и исследования сложных систем / под ред. А. А. Самарского. – М.: Наука, 1989.

20. Перегудов Ф. И. Основы системного анализа: учебник. – 2-е изд., доп. – Томск: Изд-во НТЛ, 1997.

21. Питерсон Дж. Теория сетей Петри и моделирование систем: пер. с англ. – М.: Мир, 1984.

22. Рапопорт Б. М. Инжиниринг и моделирование бизнеса. – М: Тандем, 2001.

23. Романовский И. В. Исследование операций и статистическое моделирование. – СПб.: СПб. гос. ун-т, 1994.

24. Советов Б. Я. Моделирование систем: учебник для вузов. – 3-е изд., перераб. и доп. – М.: Высш. шк., 2001.

25. Суворова Н. Информационное моделирование: величины, объекты, алгоритмы. – М.: Лаборатория базовых знаний, 2002.

26. Федотова Д. Э., Семенов Ю. Д., Чижик К. Н. CASE-технологии. – М.: Горячая линия – Телеком, 2003.

27. Шебеко Ю. А. Имитационное моделирование и ситуационный анализ бизнес-процессов принятия управленческих решений (учебное и практическое пособие). – М.: Диаграмма, 1999.

28. Шеннон Р. Ю. Имитационное моделирование систем – искусство и наука: пер. с англ. – М.: Мир, 1978.

29. Щепетова С. Е. Динамическое моделирование функционирования предприятия и формирование стратегии его поведения в конкурентной среде: автореф. дис. на соискание ученой степени к.э.н. – М.: Финансовая академия при Правительстве РФ, 2001.

30. ARENA Users Guide, Sewickley: Systems Modeling Co., 1996.

31. Barjis J., Shishkov B. UML based business systems modeling and simulation. – Proceedings of EuroSim 2001, 2001.

32. Giaglis G. M., Paul R.G., Okeefe R. M. Discrete simulation for business simulation. – Berlin: Springer – Verlag, 2003.

33. Goldsman D. A Whirlwind Tour of Computer Simulation Techniques. – www.hyperionics.com , 2003.

34. Hlupic V., Robinson S. Business Process Modeling and Analysis using discrete-event simulation. – Proceedings of the 1998 Winter Simulation Conference, pp.1363–1369.

35. Шеер А.-В. Моделирование бизнес-процессов. – М.: Весть-МетаТехнология, 2000.

36. Шеер А.-В. Бизнес-процессы. Основные понятия. Теория. Методы. – М.: Весть-МетаТехнология, 1999.

37. Каменнова М. С., Громов А. И., Ферапонтов М. М., Шматалюк А. Е. Моделирование бизнеса. Методология ARIS. – М.: Весть-МетаТехнология, 2001.

38. Розенберг В. Я. , Прохоров А. И. Что такое теория массового обслуживания. – М.: Изд-во «Советское радио», 1962.

39. Малышкин В.Э. Основы параллельных вычислений: Учебное пособие / Часть 1. – Новосибирск: Изд-во НГТУ, 1998.

Содержание

Замятина Оксана Михайловна

кандидат технических наук, доцент кафедры оптимизации
систем управления Томского политехнического университета

МОДЕЛИРОВАНИЕ СИСТЕМ

Учебное пособие

Научный редактор
доктор технических наук, профессор В.А. Силич

Редактор

Подписано к печати_____. Формат 60x84/16. Бумага «Классика».
Печать RISO. Усл.печ.л._____. Уч.-изд.л._____.
Заказ_____. Тираж 100 экз.

Томский политехнический университет
Система менеджмента качества
Томского политехнического университета сертифицирована
NATIONAL QUALITY ASSURANCE по стандарту ISO 9001:2000