Министерство образования и науки Республики Казахстан
Костанайский государственный университет
имени А. Байтурсынова

Кафедра машиностроения

Шаяхметов А. Б.

ИСПОЛЬЗОВАНИЕ ЧИСЛЕННЫХ МЕТОДОВ ДЛЯ МОДЕЛИРОВАНИЯ СИСТЕМ

Учебное пособие

Костанай, 2017

УДК 621:52-17(075.8)
ББК 30в6я73
Ш32

Рецензенты:
Кушнир В.Г., доктор технических наук, доцент, Костанайский государственный университет имени А.Байтурсынова;
Моисеенко О.В., кандидат технических наук, доцент, Костанайский инженерно-экономический университет;
Бедыч Т. В., кандидат технических наук, Костанайский инженерно-экономический университет.

Автор:
Шаяхметов Амангельды Булатович, и.о.доцента, к.т.н.

Шаяхметов А.Б.
Ш 32 Использование численных методов для моделирования систем. Учебное пособие. По специальности 6М072400-Технологические машины и оборудование (по отраслям) / А.Б. Шаяхметов Часть 1. – Костанай: КГУ имени А.Байтурсынова, 2017. – 109 с.
ISBN 978-601-7933-06-7

В учебном пособии рассматриваются основные понятия теории моделирования, основы моделирования систем, методологии и средства структурного моделирования процессов и систем, имитационное моделирование систем. Предназначено для магистрантов, обучающихся по специальности 6М072400-Технологические машины и оборудование

УДК 621:52-17(075.8)
ББК 30в6я73

Утверждено и рекомендовано Учебно-методическим советом Костанайского государственного университета имени А. Байтурсынова, от 20.01.2017г. протокол № 1.

ISBN 978-601-7933-06-7

Содержание

Введение

Моделирование технических систем – упрощенное отображение реального изделия и его описания сцелью оценки соответствия его какому-либо требованию или осуществления выбора наилучшего изделия изнескольких альтернативных вариантов. Обычно используют три способа моделирования технических систем: 1) мысленное моделирование технических систем, в ходе которого человек, изучая изделие, егопроект или др. описание, интуитивнооценивает соответствие определенным требованиям или выбор наилучшего варианта; 2) математическое моделирование технических систем связано с разработкой способов расчета и компьютерных программ для получения необходимых оценок; 3) физическоемоделирование технических систем связано с изготовлением и испытанием упрощенных физическихмоделей реального изделия. Мысленное моделирование технических систем основывается на знаниях и, главное, на собственном опыте проектирования и эксплуатации данного класса тех. систем и представляет собой одно из средств моделирования технических систем. Точность мысленного моделирования зависит от личного опыта и природных способностей эксперта и для мало изученных технических систем может превосходить точность математической модели. Основные преимущества мысленного моделирования: малое время и низкая стоимость оценки. Умение осуществлять быстрое и точное мысленное моделирование является одним из необходимых качеств изобретателей и творческих личностей, которые должны его развивать и совершенствовать. При физическом моделировании решение принимается поизмеряемым параметрам, данным измерительных приборов и оборудования, способу обработки полученных результатов. С целью снижения трудоѐмкости и стоимости часто изготавливают уменьшѐнные (в несколько раз, на порядок и более) образцы технических систем, исключая из них малозначимые детали. При изменении масштаба технические системы выбирают и обосновывают систему критериев подобия, спомощью которых выполняют перерасчет значений параметров, полученных путем измерений науменьшенных моделях, для натуральных размеров. В различных прикладных областях (гидравлика, аэродинамика, строительная механика, электродинамика и т.д.) разработаны свои системы критериев подобия и накоплен специфический опыт их использования. Физическое моделирование часто используют для обоснования достоинств новых технических решений.

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 мысленнопредставляемый 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.

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. 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 существующей 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 %.

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.

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. рисунок 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:

- 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.

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.).

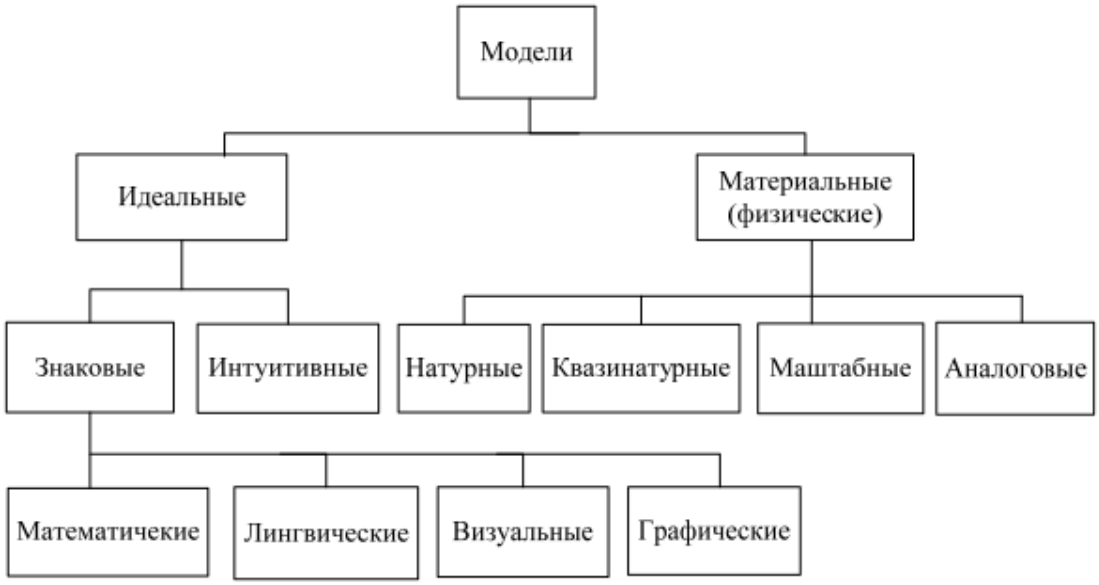


Рисунок 1 - Схема классификации моделей по степени абстрагирования от объекта-оригинала

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, например, 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. рисунок 2).

Рисунок 2 - Схема классификации математических моделей по принадлежности к иерархическому уровню

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).

По характеру отображаемых свойств объекта модели можно классифицировать на структурные и функциональные (рисунок 3).

Рисунок 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 рисунок 4.

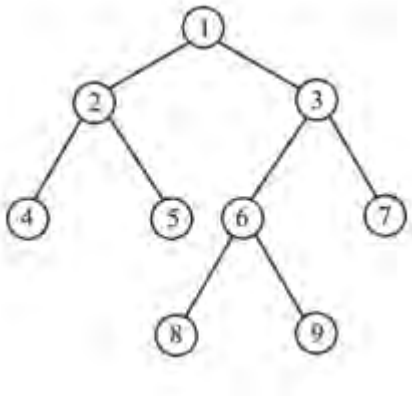


Рисунок 4 - Модель иерархической структуры

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 рисунке 5 и в таблице 1.

Таблица 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

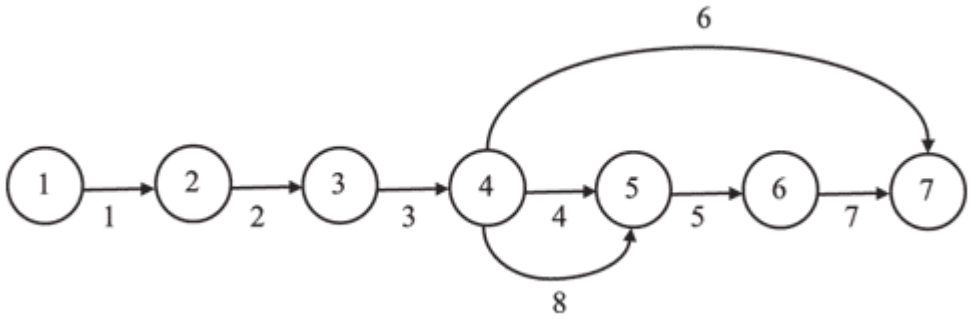


Рисунок 5 - Сетевой график строительства работ

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

По способу представления свойств объекта (рисунок 6) модели делятся на аналитические, численные, алгоритмические и имитационные.

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.

Рисунок 6 - Схема классификации математических моделей по способу представления свойств объекта

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 еѐ 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 12 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 (рисунок 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.

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 (рисунок 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.

Рисунок 7 - Схема классификации математических моделей по способу получения модели

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

Рисунок 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. рисунок 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, приведенном у в движение, но возмущения в ней со временем затухают и исчезают.

Рисунок 9 - Схема классификации математических моделей

В неустойчивой системе, находящейся первоначально в состоянии покоя, возникающее 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 системам 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 (рисунок 10).

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.

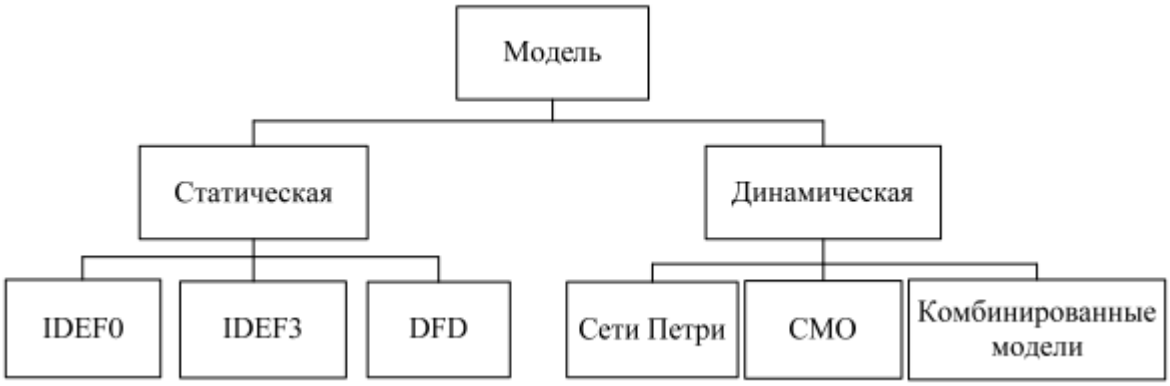


Рисунок 10. Схема классификации математических моделей по отношению ко времени

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 рисунке 11.

Рисунок 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 в моделирования системы.

Концептуальная модель (КМ) – это абстрактная модель, определяющая состав и структуру системы, свойства элементов и причинноследственные 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 данных (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 и.

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 и корректность создания программной модели.

После выбора среды проектирования концептуальная модель, сформулированная на предыдущем этапе, воплощается в компьютерную 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.

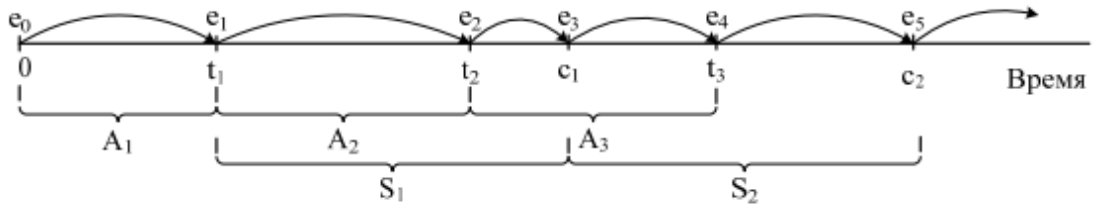


Рисунок 12 - Механизм продвижения модельного времени от события к событию

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.

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 рисунке 13, где изогнутые стрелки показывают продвижение 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, 2Δt) события не происходят, но все же модель выполняет проверку, чтобы убедиться в этом. На интервале [2Δt, 3Δt) события происходят в моменты времени e 2 и e 3 , однако считается, что они произошли в момент времени 3Δt 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 события

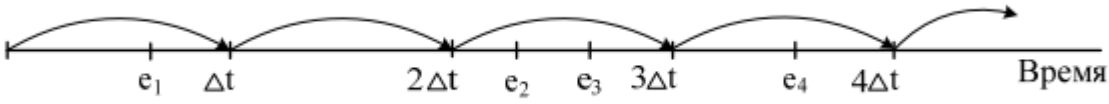


Рисунок 13 - Пример продвижения модельного времени посредством постоянного шага

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 ому из предыдущих этапов.

Подэтап «Анализ результатов моделирования» представляет собой всесторонний анализ полученных результатов.

На этапе «Подведение итогов моделирования согласно поставленной цели и задачи моделирования» проводят оценку проделанной работы, сопоставляют поставленные цели с полученными

1.4 Современные средства моделирования, представленные на ИТ рынке

1.4.1 ARIS Toolset

ARIS – это методология, или как называют авторы «концепция», 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, ю 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, состоящих перед аппаратом управления, с совокупностью деревьев функций, необходимых для достижения поставленных целей;

− информационные модели, отражающие структуру информации, необходимой для реализации всей совокупности функций 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.

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 интернета.

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

Достоинством семейства программных продуктов ARIS является не высокие требования к аппаратному и программному обеспечению.

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.

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. С помощью пакета ITHINK может быть смоделирован весь производственно-сбытовой цикл предприятия, начиная от закупки сырья и заканчивая его производством и реализацией.

Система ITHINK позволяет решать следующие задачи системного анализа:

- возможность разработки моделей разнообразных систем;
- определение механизма взаимодействия компонентов системы;
- использование и разработка формальных эффективных процедур имитации;
- 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).

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 моделируемой 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.

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++.

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 выдаѐт 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 ещѐ 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 нѐм 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 АВС-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.

КОНТРОЛЬНЫЕ ВОПРОСЫ

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 Основы моделирования систем

Модель и моделирование - универсальные понятия, атрибуты одного из наиболее мощных методов познания в любой профессиональной области, познания системы, процесса, явления.

Модели и моделирование объединяют специалистов различных областей, работающих над решением межпредметных проблем, независимо от того, где эта модель и результаты моделирования будут применены. Вид модели и методы ее исследования больше зависят от информационно-логических связей элементов и подсистем моделируемой системы, ресурсов, связей с окружением, используемых при моделировании, а не от конкретной природы, конкретного наполнения системы.

У моделей, особенно математических, есть и дидактические аспекты - развитие модельного стиля мышления, позволяющего вникать в структуру и внутреннюю логику моделируемой системы.

Построение модели - системная задача, требующая анализа и синтеза исходных данных, гипотез, теорий, знаний специалистов. Системный подход позволяет не только построить модель реальной системы, но и использовать эту модель для оценки (например, эффективности управления, функционирования) системы.

Модель - 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. Модель - результат отображения одной структуры (изученной) на другую (малоизученную). Отображая физическую систему (объект) на математическую систему (например, математический аппарат уравнений), получим физико-математическую модель системы или математическую модель физической системы. Любая модель строится и исследуется при определенных допущениях, гипотезах.

Пример. Рассмотрим физическую систему: тело массой m скатывающееся по наклонной плоскости с ускорением a, на которое воздействует сила F. Исследуя такие системы, Ньютон получил математическое соотношение: $F=ma$. Это физико-математическая модель системы или математическая модель физической системы. При описании этой системы (построении этой модели) приняты следующие гипотезы: 1) поверхность идеальна (т.е. коэффициент трения равен нулю); 2) тело находится в вакууме (т.е. сопротивление воздуха равно нулю); 3) масса тела неизменна; 4) тело движется с одинаковым постоянным ускорением в любой точке.

Пример. Физиологическая система - система кровообращения человека - подчиняется некоторым законам термодинамики. Описывая эту систему на физическом (термодинамическом) языке балансовых законов, получим физическую, термодинамическую модель физиологической

системы. Если записать эти законы на математическом языке, например, выписать соответствующие термодинамические уравнения, то уже получим математическую модель системы кровообращения. Назовем ее физиолого-физико-математической моделью или физико-математической моделью.Пример. Совокупность предприятий функционирует на рынке, обмениваясь товарами, сырьем, услугами, информацией. Если описать экономические законы, правила их взаимодействия на рынке с помощью математических соотношений, например, системы алгебраических уравнений, где неизвестными будут величины прибыли, получаемые от взаимодействия предприятий, а коэффициентами уравнения будут значения интенсивностей таких взаимодействий, то получим математическую модель экономической системы, т.е. экономико-математическую модель системы предприятий на рынке.

Пример. Если банк выработал стратегию кредитования, смог описать ее с помощью экономико-математических моделей и прогнозирует свою тактику кредитования, то он имеет большую устойчивость и жизнеспособность.

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

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.

Модели, если отвлечься от областей, 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, как правило, 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.

По уровню, "глубине" моделирования модели бывают:

1. эмпирические - на основе эмпирических фактов, зависимостей;

2. теоретические - на основе математических описаний;

3. смешанные, полуэмпирические - на основе эмпирических зависимостей и математических описаний.

Проблема моделирования состоит из трех задач:

1. построение модели (эта задача менее формализуема и конструктивна, в том смысле, что нет алгоритма для построения моделей);

2. исследование модели (эта задача более формализуема, имеются методы исследования различных классов моделей);

3. использование модели (конструктивная и конкретизируемая задача).Модель M, описывающая систему S(x1, x2, ..., xn; R), имеет вид: M=(z1, z2, ..., zm; Q), где zi Z, i=1, 2, ..., n, Q, R - множества отношений над X - множеством входных, выходных сигналов и состояний системы, Z - множество описаний, представлений элементов и подмножеств X.

Схема построения модели M системы S с входными сигналами X и выходными сигналами Y изображена на рисунке 14.

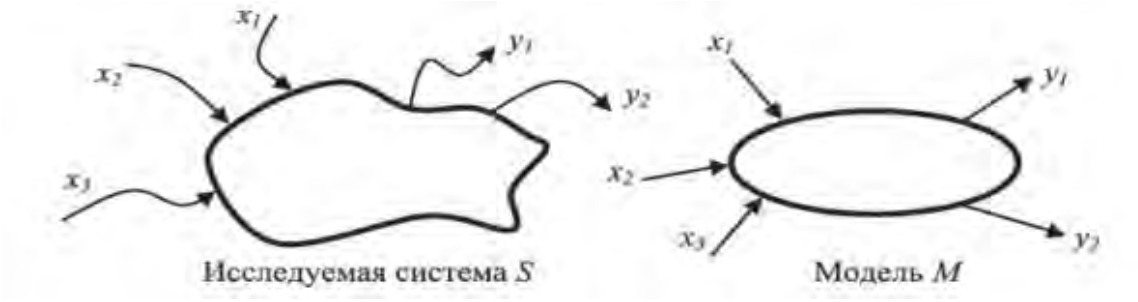


Рисунок 14 - Схема построения модели

Если на вход M поступают сигналы из X и на входе появляются сигналы Y, то задан закон, правило f функционирования модели, системы.

Моделирование - это универсальный метод получения, описания и использования знаний. Он используется в любой профессиональной деятельности. В современной науке и технологии роль и значение моделирования усиливается, актуализируется проблемами, успехами других наук.

Моделирование реальных и нелинейных систем живой и неживой природы позволяет перекидывать мостики между нашими знаниями и реальными системами, процессами, в том числе и мыслительными.

Классификацию моделей проводят по различным критериям. Мы будем использовать наиболее простую и практически значимую.

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.

Пример. Закон Ньютона F=am - это статическая модель движущейся с ускорением a материальной точки массой m. Эта модель не учитывает изменение ускорения от одной точки к другой.

Модель динамическая, если среди ее параметров есть временной параметр, т.е. она отображает систему (процессы в системе) во времени.

Модель имитационная, если она предназначена для испытания или изучения возможных путей развития и поведения объекта путем варьирования некоторых или всех параметров модели.

Модель детерминированная, если каждому входному набору параметров соответствует вполне определенный и однозначно определяемый набор выходных параметров; в противном случае - модель недетерминированная, стохастическая (вероятностная).

Модель функциональная, если она представима в виде системы каких- либо функциональных соотношений.

Модель теоретико-множественная, если она представима с помощью некоторых множеств и отношений принадлежности им и между ними.

Пример. Пусть заданы множество X={Николай, Петр, Николаев, Петров, Елена, Екатерина, Михаил, Татьяна} и отношения: Николай - супруг Елены, Екатерина - супруга Петра, Татьяна - дочь Николая и Елены, Михаил - сын Петра и Екатерины, семьи Михаила и Петра дружат друг с другом.

Тогда множество X и множество перечисленных отношений Y могут служить теоретико-множественной моделью двух дружественных семей.

Модель логическая, если она представима предикатами, логическими функциями.

Модель игровая, если она описывает, реализует некоторую игровую ситуацию между участниками игры (лицами, коалициями).

Модель алгоритмическая, если она описана некоторым алгоритмом или комплексом алгоритмов, определяющим ее функционирование, развитие. Введение такого, на первый взгляд, непривычного типа моделей (действительно, кажется, что любая модель может быть представлена алгоритмом еѐ исследования), на наш взгляд, вполне обосновано, так как не все модели могут быть исследованы или реализованы алгоритмически.

Пример. Моделью вычисления суммы бесконечного убывающего ряд чисел может служить алгоритм вычисления конечной суммы ряда до некоторой заданной степени точности. Алгоритмической моделью корня квадратного из числа x может служить алгоритм вычисления его приближенного сколь угодно точного значения по известной рекуррентной формуле.

Модель структурная, если она представима структурой данных или структурами данных и отношениями между ними.

Пример. Структурной моделью может служить описание (табличное, графовое, функциональное или другое) трофической структуры экосистемы. Модель графовая, если она представима графом или графами и отношениями между ними.Модель иерархическая (древовидная), если представима некоторой иерархической структурой (деревом).

Пример. Для решения задачи нахождения маршрута в дереве поиска можно построить, например, древовидную модель (рисунок 15):

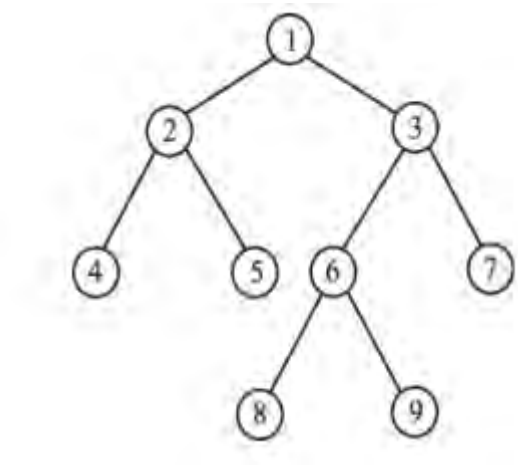


Рисунок 15 - Модель иерархической структуры

Модель сетевая, если она представима некоторой сетевой структурой.

Пример. Строительство нового дома включает операции, приведенные в нижеследующей таблице.

Таблица 2 – Работы 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

Сетевая модель (сетевой график) строительства дома дана на рисунке16.

Две работы, соответствующие дуге 4-5, параллельны, их можно либо заменить одной, представляющей совместную операцию (монтаж электропроводки и настил крыши) с новой длительностью 3+5=8, либо ввести на одной дуге фиктивное событие, тогда дуга 4-5 примет вид.

Модель языковая, лингвистическая, если она представлена некоторым лингвистическим объектом, формализованной языковой системой или структурой.

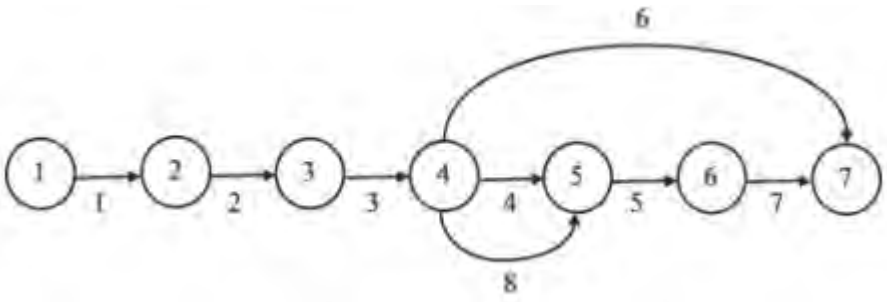


Рисунок 16 - Сетевой график строительства работ

Иногда такие модели называют вербальными, синтаксическими и т.п.

Модель визуальная, если она позволяет визуализировать отношения и связи моделируемой системы, особенно в динамике.

Модель натурная, если она есть материальная копия объекта моделирования.

Пример. Глобус - натурная географическая модель земного шара.

Модель геометрическая, графическая, если она представима геометрическими образами и объектами.

Пример. Макет дома является натурной геометрической моделью строящегося дома. Вписанный в окружность многоугольник дает модель окружности. Именно она используется при изображении окружности на экране компьютера. Прямая линия является моделью числовой оси, а плоскость часто изображается как параллелограмм. Модель клеточно-автоматная, если она представляет систему с помощью клеточного автомата или системы клеточных автоматов. Клеточный автомат - дискретная динамическая система, аналог физического (непрерывного) поля. Клеточно-автоматная геометрия - аналог евклидовой геометрии.

Неделимый элемент евклидовой геометрии - точка, на основе ее строятся отрезки, прямые, плоскости и т.д. Неделимый элемент клеточно-автоматного поля - клетка, на основе еѐ строятся кластеры клеток и различные конфигурации клеточных структур. Это "мир" некоторого автомата, исполнителя, структуры.

Представляется клеточный автомат равномерной сетью клеток ("ячеек") этого поля. Эволюция клеточного автомата разворачивается в дискретном пространстве - клеточном поле. Такие клеточные поля могут быть вещественно-энерго-информационными. Законы эволюции локальны, т.е. динамика системы определяется задаваемым неизменным набором законов или правил, по которым осуществляется вычисление новой клетки эволюции и его материально-энерго-информационной характеристики в

зависимости от состояния окружающих ее соседей (правила соседства, как уже сказано, задаются). Смена состояний в клеточно-автоматном поле происходит одновременно и параллельно, а время идет дискретно. Несмотря на кажущуюся простоту их построения, клеточные автоматы могут демонстрировать разнообразное и сложное поведение. В последнее время они широко используются при моделировании не только физических, но и социально-экономических процессов.

Клеточные автоматы (поля) могут быть одномерными, двумерными (с ячейками на плоскости), трехмерными (с ячейками в пространстве) или же многомерными (с ячейками в многомерных пространствах).

Пример. Классическая клеточно-автоматная модель - игра "Жизнь" Джона Конвея. Она описана во многих книгах. Мы рассмотрим другую клеточно-автоматную модель загрязнения среды, диффузии загрязненителя в некоторой среде. 2D-клеточный автомат (на плоскости) для моделирования загрязнения среды может быть сгенерирован следующими правилами:

1. плоскость разбивается на одинаковые клетки: каждая клетка может находиться в одном из двух состояний: состояние 1 - в ней есть диффундирующая частица загрязнителя, и состояние 0 - если ее нет;

2. клеточное поле разбивается на блоки 2×2 двумя способами, которые будем называть етным и нечетным разбиениями (у чѐтного разбиения в кластере или блоке находится четное число точек или клеток поля, у нечетного блока - их нечетное число);

3. на очередном шаге эволюции каждый блок четного разбиения поворачивается (по задаваемому правилу распространения загрязнения или генерируемому распределению случайных чисел) на заданный угол (направление поворота выбирается генератором случайных чисел);

4. аналогичное правило определяется и для блоков нечетного разбиения;

5. процесс продолжается до некоторого момента или до очищения среды.

Пусть единица времени - шаг клеточного автомата, единица длины - размер его клетки. Если перебрать всевозможные сочетания поворотов блоков четного и нечетного разбиения, то видим, что за один шаг частица может переместиться вдоль каждой из координатных осей на расстояние 0, 1 или 2 (без учета направления смещения) с вероятностями, соответственно, p 0 =1/4, p 1 =1/2, p 2 =1/4. Вероятность попадания частицы в данную точку зависит лишь от ее положения в предыдущий момент времени, поэтому рассматриваем движение частицы вдоль оси x (y) как случайное.

На рисунке 17 - фрагменты работы программы клеточно-автоматной модели загрязнения клеточной экосреды (размеры клеток увеличены).

Модель фрактальная, если она описывает эволюцию моделируемой системы эволюцией фрактальных объектов. Если физический объект

однородный (сплошной), т.е. в нем нет полостей, можно считать, что плотность не зависит от размера.

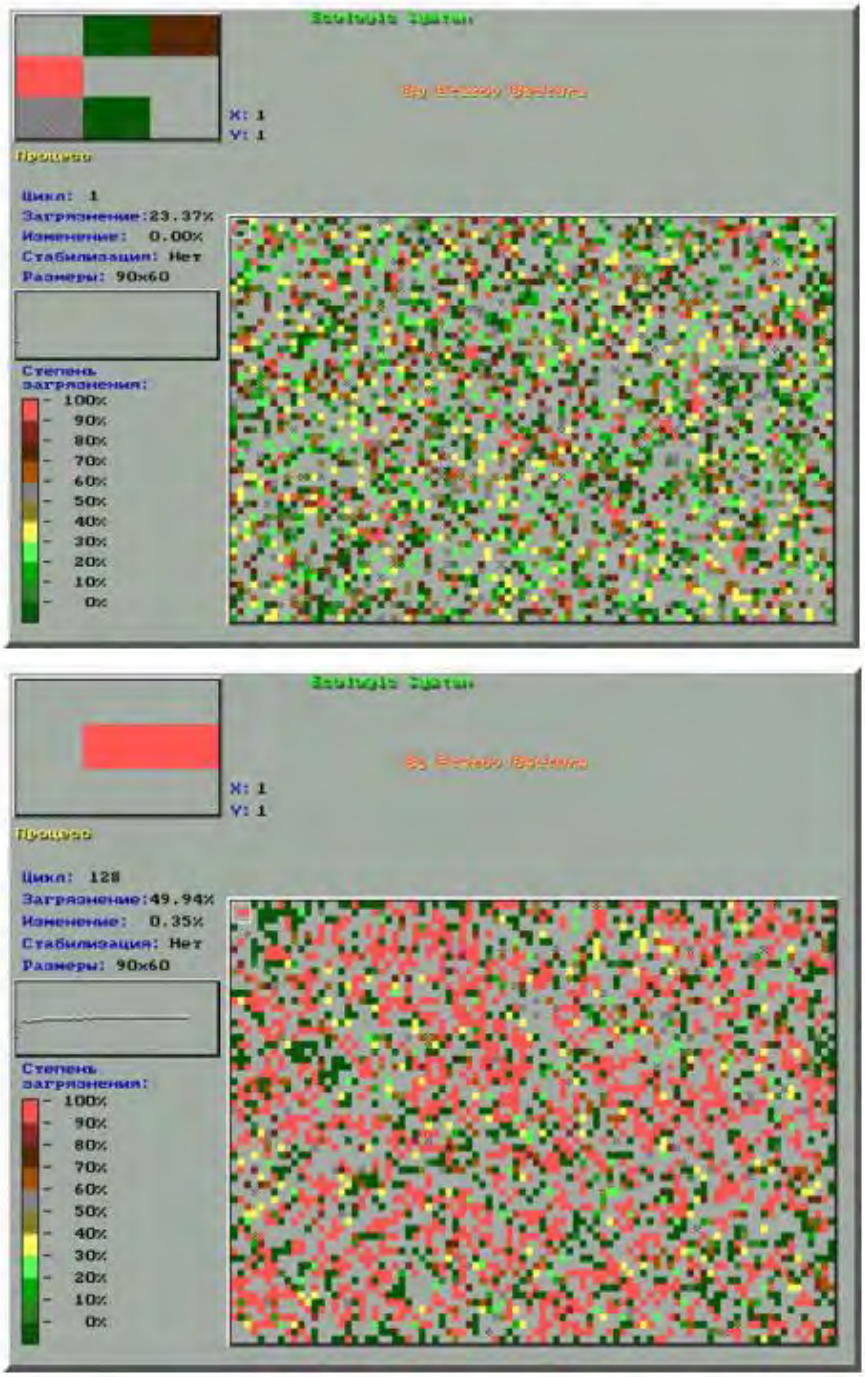


Рисунок 17 - Окно справа - состояние клеточного поля (в верхнем - исходное, слабо загрязненное, в нижнем - после 120 циклов загрязнения), в левом верхнем углу - "Микроскоп", увеличивающий кластер поля, в середине слева - график динамики загрязнения, внизу слева - индикаторы загрязнения

Фрактальные объекты самоподобны, если они выглядят одинаково в любом пространственном масштабе, масштабно инвариантны, фрагменты

структуры повторяются через определенные пространственные промежутки. Поэтому они очень хорошо подходят для моделирования нерегулярностей, так как позволяют описывать (например, дискретными моделями) эволюцию таких систем для любого момента времени и в любом пространственном масштабе. Самоподобие встречается в самых разных предметах и явлениях.

Пример. Самоподобны ветки деревьев, снежинки, экономические системы (волны Кондратьева), горные системы.

Фрактальная модель применяется обычно тогда, когда реальный объект нельзя представить в виде классической модели, когда имеем дело с нелинейностью (многовариантностью путей развития и необходимостью выбора) и недетерминированностью, хаотичностью и необратимостью эволюционных процессов.

Тип модели зависит от информационной сущности моделируемой системы, от связей и отношений его подсистем и элементов, а не от его физической природы.Пример. Математические описания (модели) динамики эпидемии инфекционной болезни, радиоактивного распада, усвоения второго иностранного языка, выпуска изделий производственного предприятия и т.д. являются одинаковыми с точки зрения их описания, хотя процессы различны.

Границы между моделями различного типа или же отнесение модели к тому или иному типу часто весьма условны. Можно говорить о различных режимах использования моделей - имитационном, стохастическом и т.д.

Основные свойства любой модели:

1. целенаправленность - модель всегда отображает некоторую систему, т.е. имеет цель;

2. конечность - модель отображает оригинал лишь в конечном числе его отношений и, кроме того, ресурсы моделирования конечны;

3. упрощенность - модель отображает только существенные стороны объекта и, кроме того, должна быть проста для исследования или воспроизведения;

4. приблизительность - действительность отображается моделью грубо или приблизительно;

5. адекватность - модель должна успешно описывать моделируемую систему;

6. наглядность, обозримость основных ее свойств и отношений;

7. доступность и технологичность для исследования или воспроизведения;

8. информативность - модель должна содержать достаточную информацию о системе (в рамках гипотез, принятых при построении модели) и должна давать возможность получить новую информацию;

9. сохранение информации, содержавшейся в оригинале (с точностью рассматриваемых при построении модели гипотез);

10. полнота - в модели должны быть учтены все основные связи и отношения, необходимые для обеспечения цели моделирования;

11. устойчивость - модель должна описывать и обеспечивать устойчивое поведение системы, если даже она вначале является неустойчивой;

12. целостность - модель реализует некоторую систему (т.е. целое);

13. замкнутость - модель учитывает и отображает замкнутую систему необходимых основных гипотез, связей и отношений;

14. адаптивность - модель может быть приспособлена к различным входным параметрам, воздействиям окружения;

15. управляемость (имитационность) - модель должна иметь хотя бы один параметр, изменениями которого можно имитировать поведение моделируемой системы в различных условиях;

16. эволюционируемость - возможность развития моделей (предыдущего уровня).

Жизненный цикл моделируемой системы:

1. сбор информации об объекте, выдвижение гипотез, предмодельный анализ;

2. проектирование структуры и состава моделей (подмоделей); 3. построение спецификаций модели, разработка и отладка отдельных подмоделей, сборка модели в целом, идентификация (если это нужно) параметров моделей;

4. исследование модели - выбор метода исследования и разработка алгоритма (программы) моделирования;

5. исследование адекватности, устойчивости, чувствительности модели;

6. оценка средств моделирования (затраченных ресурсов);

7. интерпретация, анализ результатов моделирования и установление некоторых причинно-следственных связей в исследуемой системе;

8. генерация отчетов и проектных (народно-хозяйственных) решений;

9. уточнение, модификация модели, если это необходимо, и возврат к исследуемой системе с новыми знаниями, полученными с помощью модели и моделирования.

Моделирование - метод системного анализа. Но часто в системном анализе при модельном подходе исследования может совершаться одна методическая ошибка, а именно, - построение корректных и адекватных моделей (подмоделей) подсистем системы и их логически корректная увязка не дает гарантий корректности построенной таким способом модели всей системы. Модель, построенная без учета связей системы со средой и ее поведения по отношению к этой среде, может часто лишь служить еще одним подтверждением теоремы Геделя, а точнее, ее следствия, утверждающего, что в сложной изолированной системе могут

существовать истины и выводы, корректные в этой системе и некорректные вне ее.

Наука моделирования состоит в разделении процесса моделирования (системы, модели) на этапы (подсистемы, подмодели), детальном изучении каждого этапа, взаимоотношений, связей, отношений между ними и затем эффективного описания их с максимально возможной степенью формализации и адекватности. В случае нарушения этих правил получаем не модель системы, а модель "собственных и неполных знаний".

Моделирование (в значении "метод", "модельный эксперимент") рассматривается как особая форма эксперимента, эксперимента не над самим оригиналом (это называется простым или обычным экспериментом), а над копией (заместителем) оригинала. Здесь важен изоморфизм систем (оригинальной и модельной) - изоморфизм, как самой копии, так и знаний, с помощью которых она была предложена.

Модели и моделирование применяются по основным направлениям:

1. обучение (как моделям, моделированию, так и самих моделей);
2. познание и разработка теории исследуемых систем (с помощью какихлибо моделей, моделирования, результатов моделирования);
3. прогнозирование (выходных данных, ситуаций, состояний системы);
4. управление (системой в целом, отдельными подсистемами системы), выработка управленческих решений и стратегий;
5. автоматизация (системы или отдельных подсистем системы).

КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Что такое модель, для чего она нужна и как используется?
2. Какая модель называется статической (динамической, дискретной и т.д.)?
3. Каковы основные свойства моделей и насколько они важны?
4. Что такое жизненный цикл моделирования (моделируемой системы)?

3 Методологии и средства структурного моделирования процессов и систем

3.1 SADT-методология

В настоящее время много написано и сказано о методологии SADT (Structured Analysis and Design Technique – методология структурного анализа и проектирования), 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. 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.

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. 45

3.1.1 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: 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.

3.1.1.1 Основные понятия и состав IDEF0-модели

Состав и изображение IDEF0-модели приведены на рисунке 18.

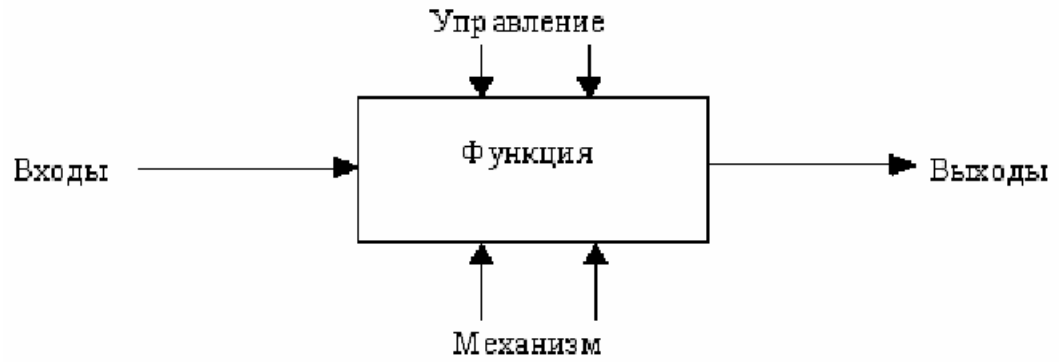


Рисунок 18 - Состав IDEF0-модели

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);

46 − 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 (рисунок 19).

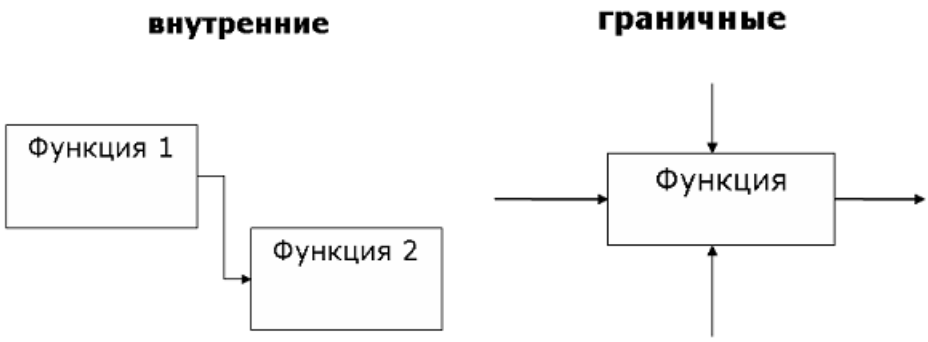


Рисунок 19 - Внутренние и граничные стрелки

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 рисунке 20. Цель моделирования указывает, для чего разрабатывается конкретная модель. Точка зрения определяет должностное лицо или подразделение организации, с точки зрения кого разрабатывается модель.

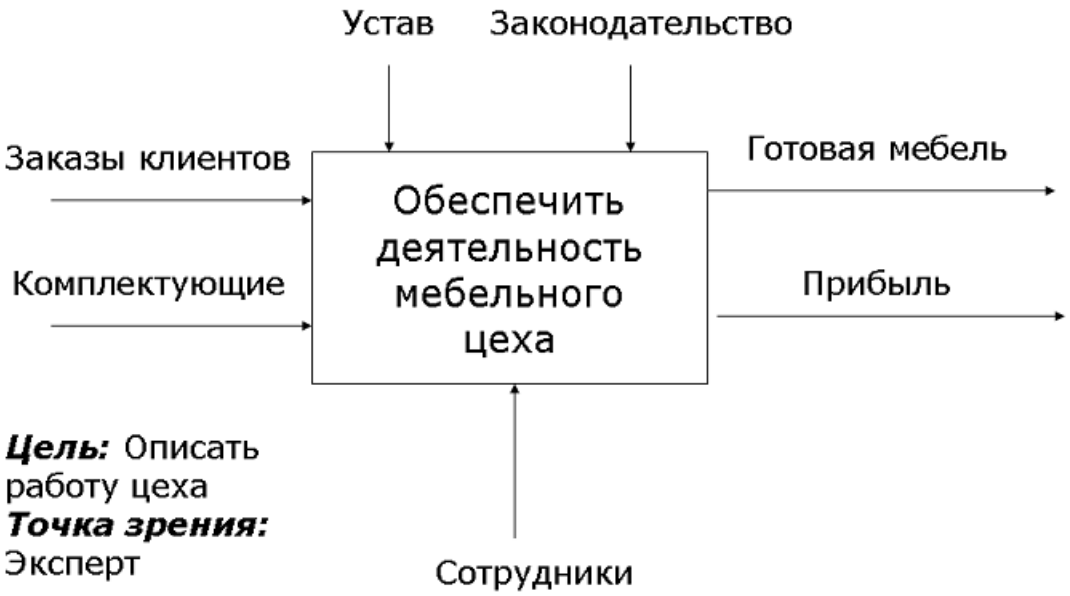


Рисунок 20 - Пример контекстной диаграммы

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 рисунке 21.

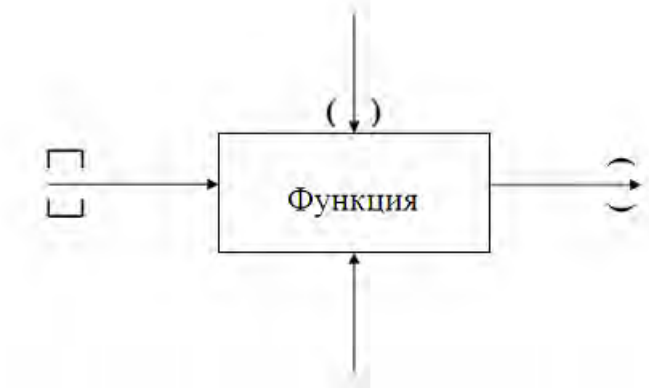


Рисунок 21 - Пример туннельных стрелок

В IDEF0-модели также могут быть стрелки ветвления и слияния, существуют правила отображения этих стрелок в модели. Пример стрелок приведен на рисунке 22 и рисунке 23 (а – неверный способ отображения, б – верный способ отображения).

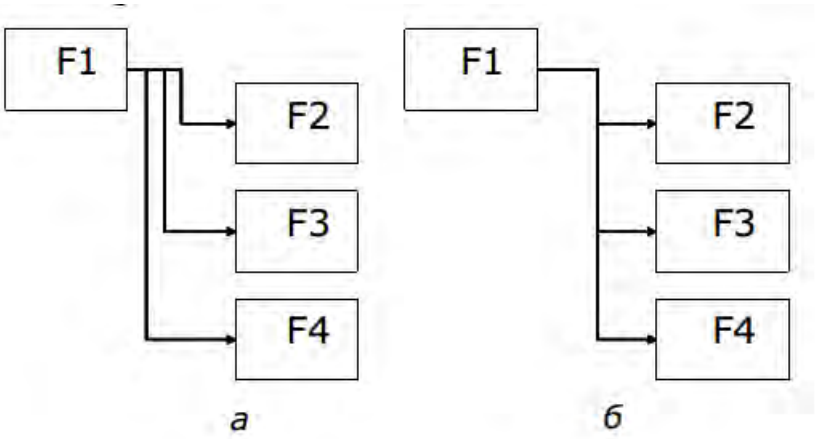


Рисунок 22 - Способ отображения стрелок ветвления:
а – неверный способ отображения; б – верный способ отображения

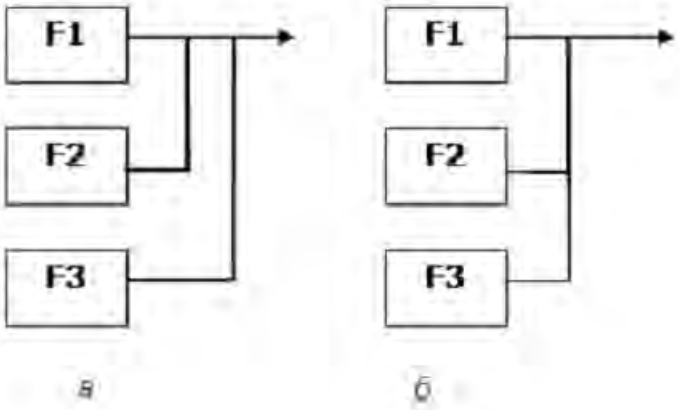


Рисунок 23 - Способ отображения стрелок слияния:
а – неверный способ отображения; б – верный способ отображения

Нумерация блоков в IDEF0-модели представляет собой отображение префикса (чаще всего используют А) и описание всей иерархии (рисунок 24).

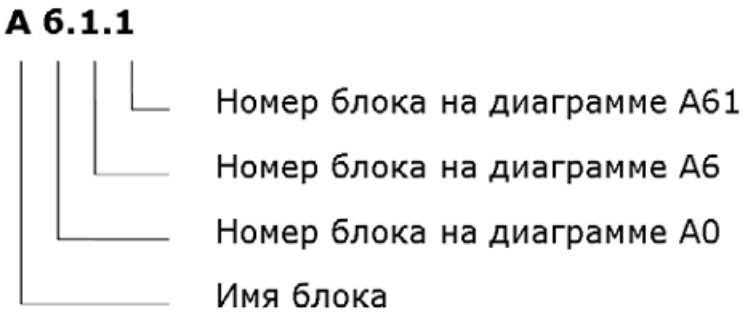


Рисунок 24 - Пример нумерации блока в модели

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 (рисунок 25);

Рисунок 25 - 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 рисунке 26.

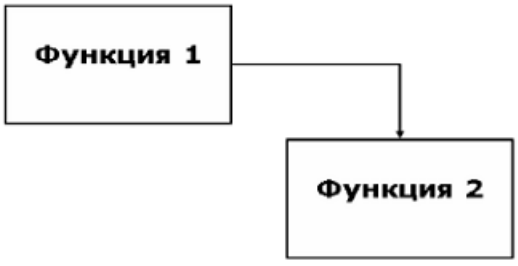


Рисунок 26 - Отношение управления

• выход–вход. Самый применяемый тип отношения, когда выход одной функции является входом для другой (рисунок 27);

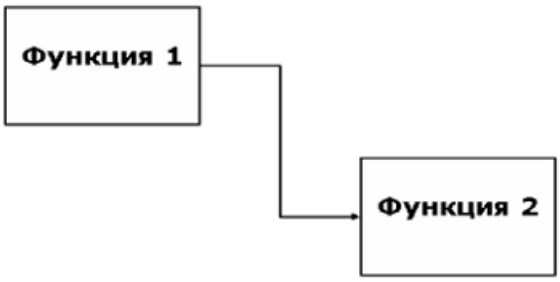


Рисунок 27 - Отношение выход–вход

• обратная связь (ОС) по управлению. Выход одной функции является управлением для другой, схож с отношением управления (см. рисунок 28);

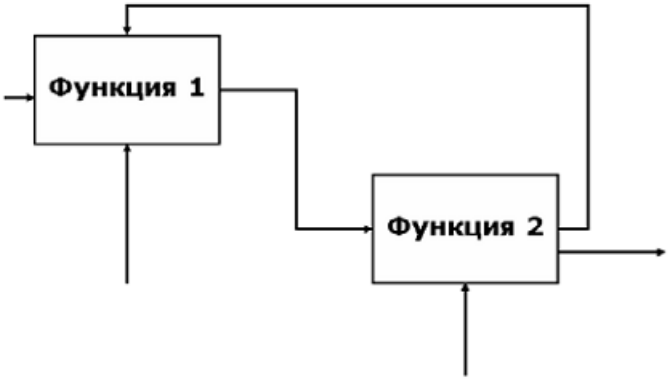


Рисунок 28 - Отношение обратная связь по управлению

• ОС по входу. Является аналогом отношения выход–вход (рисунок 29);

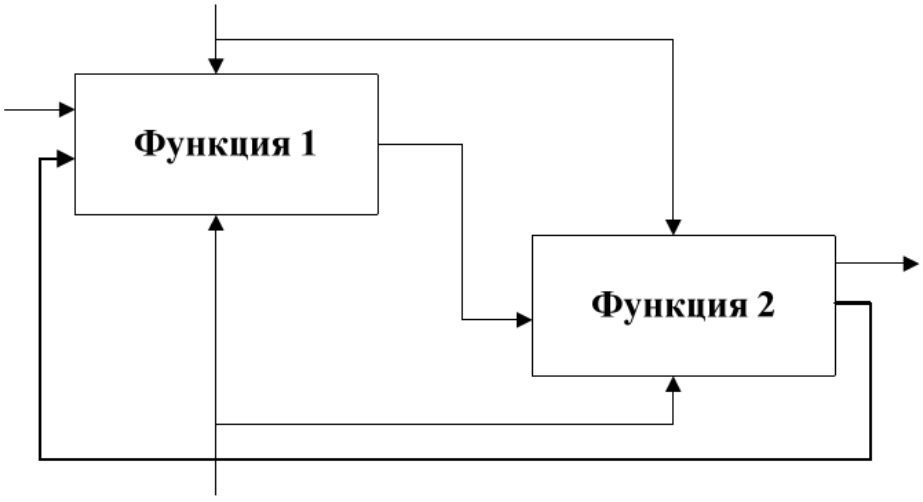


Рисунок 29 - Отношение обратная связь по входу

• 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 (рисунок 30).

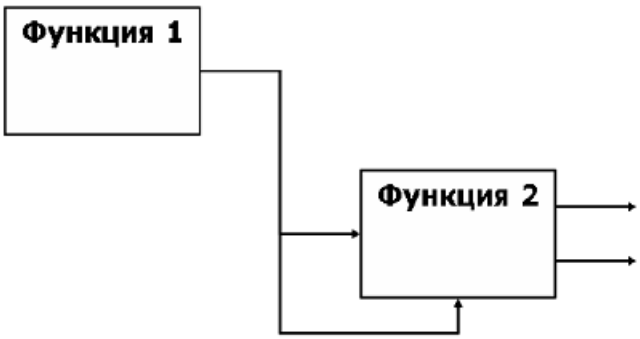


Рисунок 30 - Отношение выход–механизм

3.1.1.2 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 диаграммеродителе. 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 (рисунок 31, а – неверный способ отображения, рисунок 31, б – верный способ отображения).

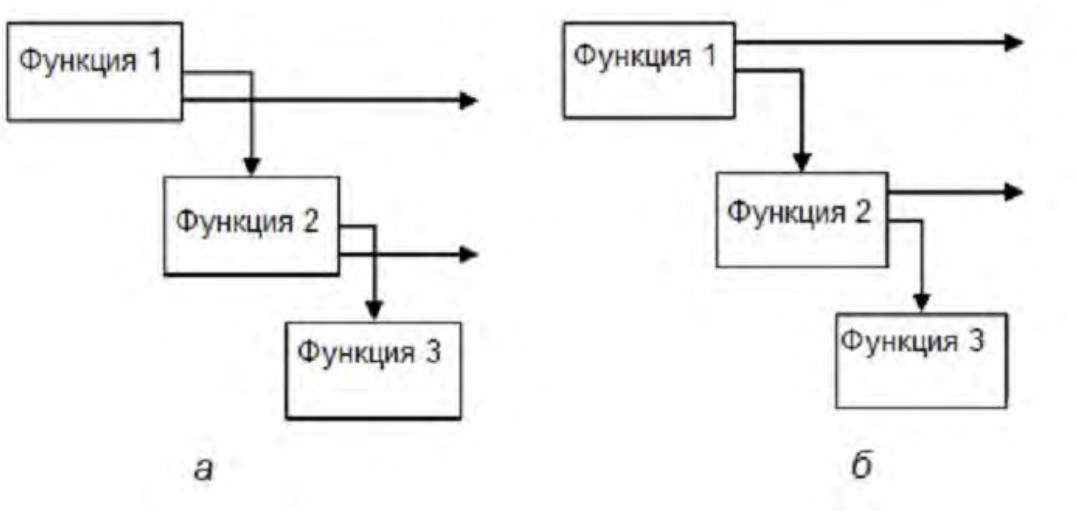


Рисунок 31 - Пример изображения стрелок в модели:
а – неверный способ отображения; б – верный способ отображения

7. Стрелки должны объединяться, если имеют общий источник (рисунок 32, а – неверный способ отображения, рисунок 32, б – верный способ отображения).

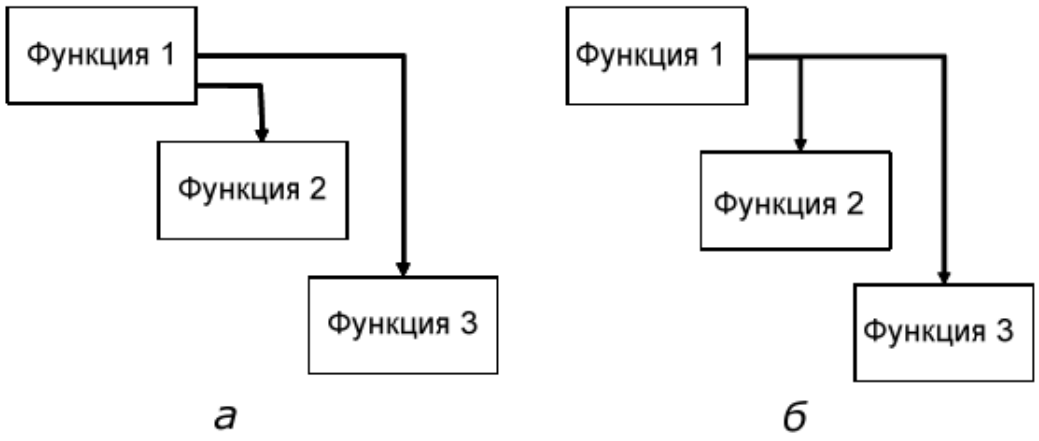


Рисунок 32 - Пример изображения ветвления стрелок в модели:
а – неверный способ отображения; б – верный способ отображения

На рисунке 33 приведен пример IDEF0-модели деятельности промышленного предприятия (а – контекстная диаграмма, б – диаграмма декомпозиции).

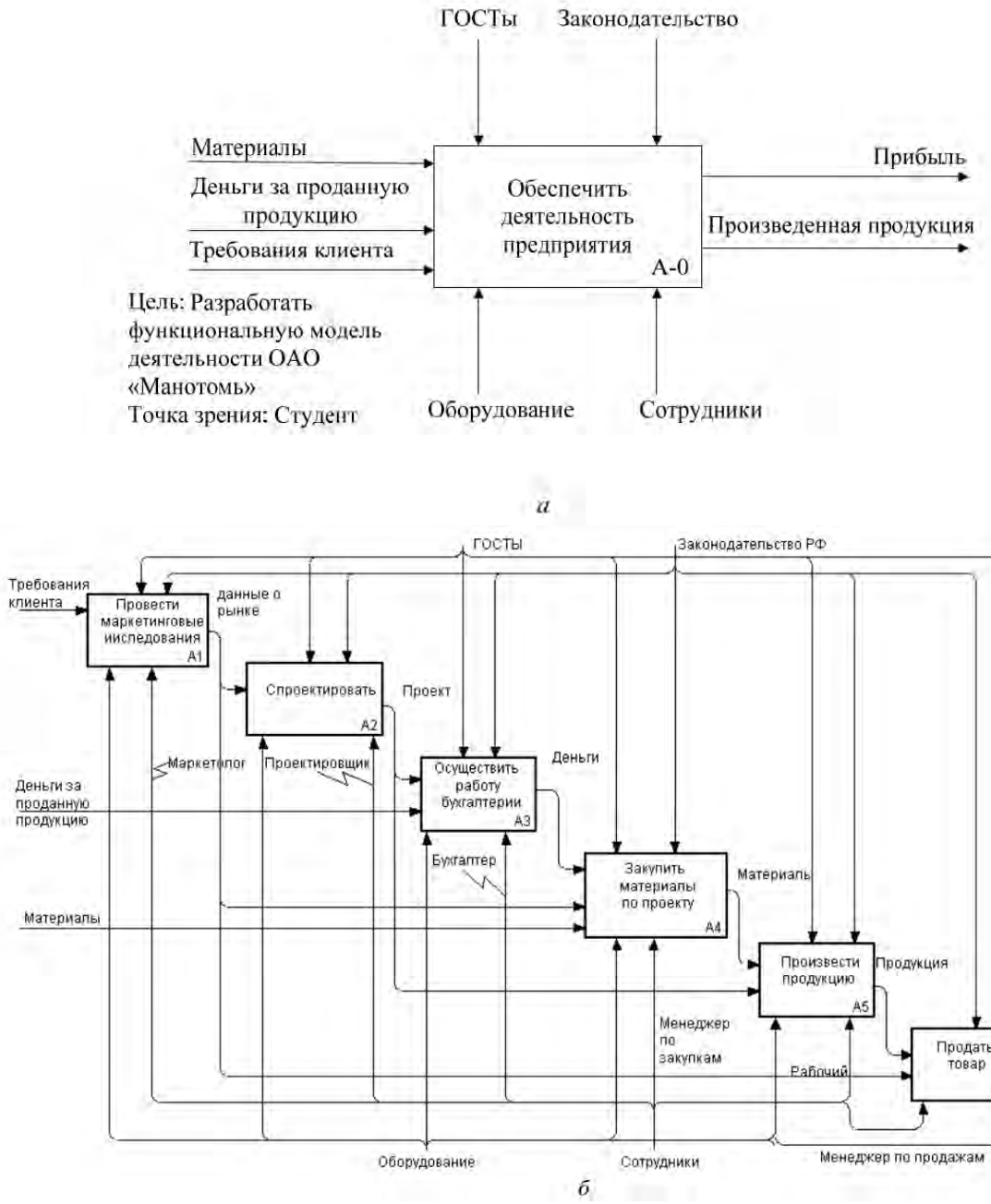


Рисунок 33 - Пример IDEF0-модели деятельности промышленного предприятия: а – контекстная диаграмма (уровень А-0); б – диаграмма основных бизнес-процессов (уровень А0)

3.1.1.3 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:

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

Атрибуты потока данных:

- 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.

3.1.1.4 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

3.1.1.5 Количественный анализ диаграмм

Для проведения количественного анализа моделей будем использовать следующие показатели:

- 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 рисунке 34.

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).

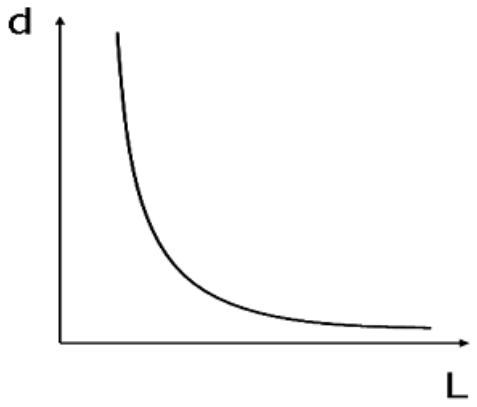


Рисунок 34 - График коэффициента декомпозиции

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 (рисунок 35).

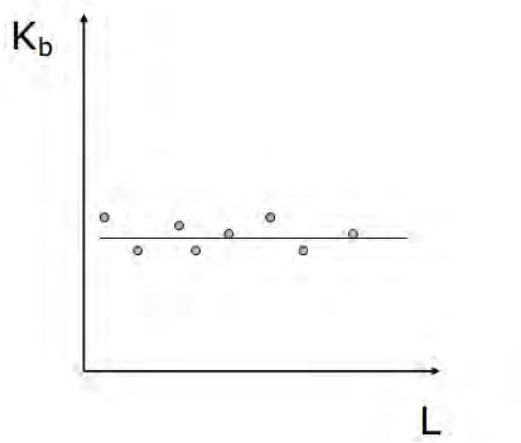


Рисунок 35 - График коэффициента сбалансированности

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.

3.1.2 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.

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 (рисунок 36). Имя задается в форме глагола, указывается номер и кто исполняет данную единицу работ.

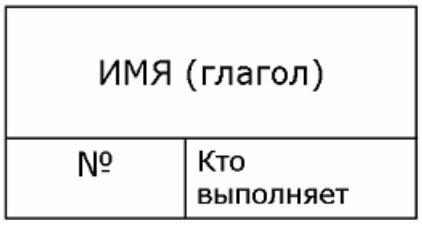


Рисунок 36 - Графическое изображение единицы работ

Говоря об единицах работ, необходимо отметить, что IDEF3модели 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. рисунок 37) могут выполнять две роли: − необходимые элементы для выполнения технологического процесса либо результат технологического процесса (металл, компоненты, готовое изделие и т. п.);

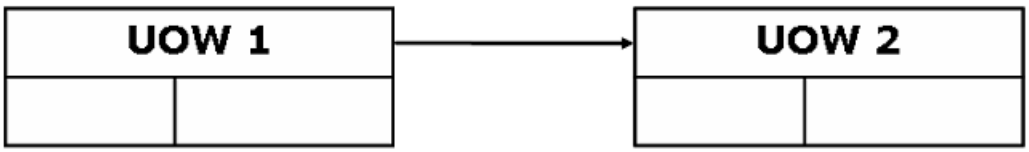
− активаторы процесса (клиент, поставщик и т. п.).

Имя ссылки задается именем существительным.

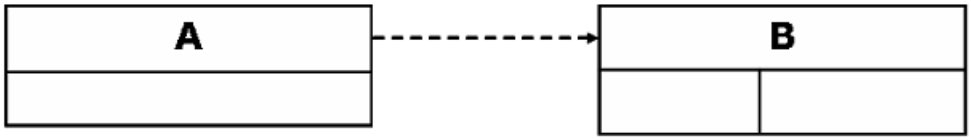
Рисунок 37 - Графическое изображение ссылки

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 (рисунок 38).

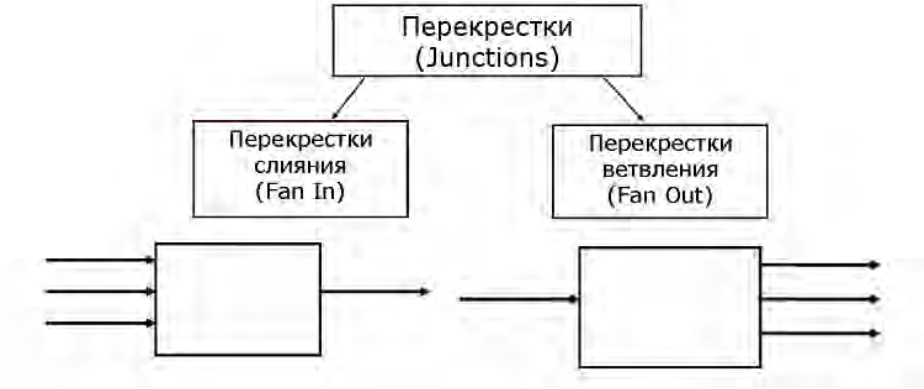


Рисунок 38 - Перекрестки слияния и ветвления

Перекресток не может быть одновременно перекрестком слияния и ветвления (рисунок 39, а), т. к. в этом случае будет неясно правило его срабатывания. Эта ситуация разрешается путем введения в модель каскада перекрестков (рисунок 39, б).

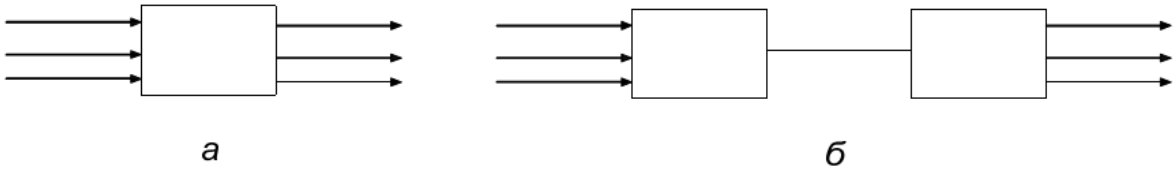


Рисунок 39 - Графическое изображение перекрестка:
а – неверный способ отображения; б – верный способ отображения

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 (рисунок 40, а): выходной процесс запустится, если завершились все входные процессы.

Вариантов срабатывания этого перекрестка – один.

Правило срабатывания перекрестка ветвления (рисунок 40, 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.

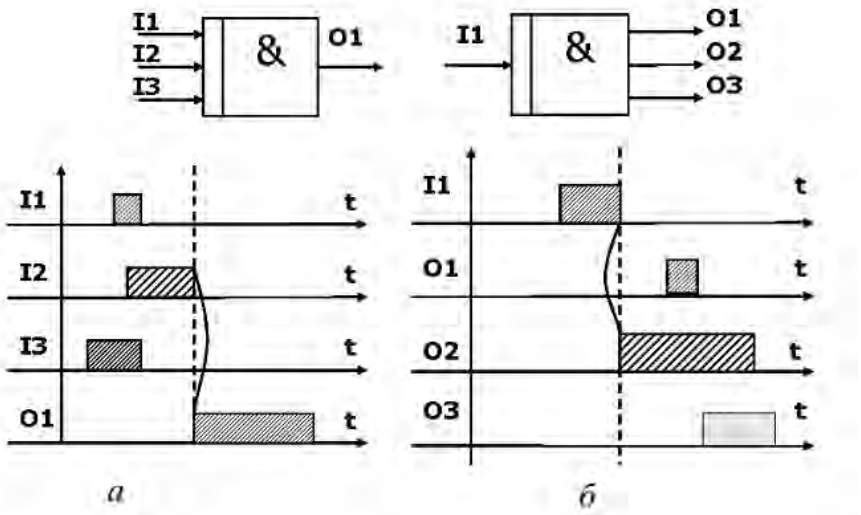


Рисунок 40 - Asynchronous AND (Асинхронное И):
а – перекресток слияния; б – перекресток ветвления

Synchronous AND (Синхронное И)

Правило срабатывания перекрестка слияния (рисунок 41, 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».

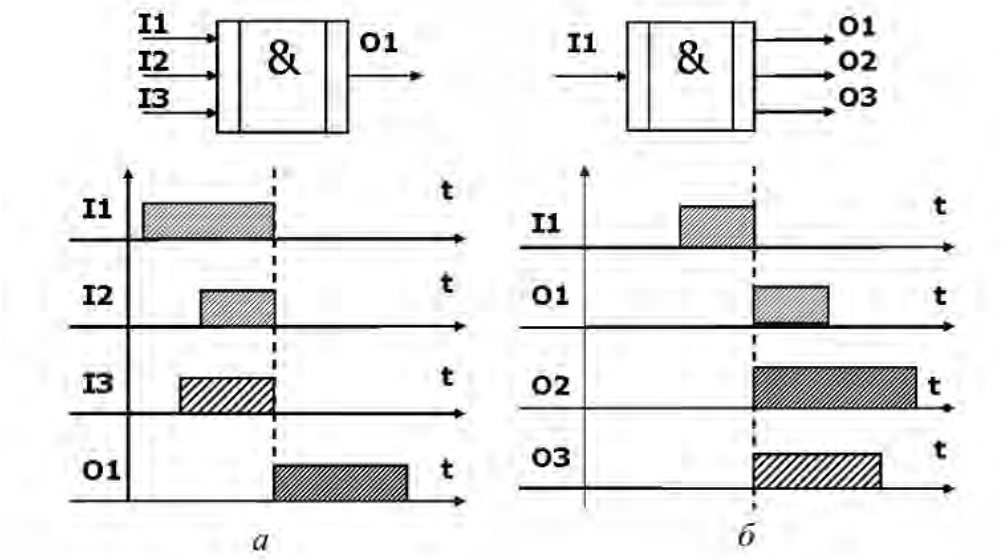


Рисунок 41 - Synchronous AND (Синхронное И):
а – перекресток слияния; б – перекресток ветвления

Правило срабатывания перекрестка ветвления (см. рисунок 41, 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. рисунок 42, а): выходной процесс запустится, если завершится один или любая возможная комбинация входных процессов.

Вариантов срабатывания этого перекрестка будет $2^N - 1$, где N – количество входов перекрестка.

Правило срабатывания перекрестка ветвления (см. рисунок 42, б): после завершения входного процесса запустятся один или любая возможная комбинация выходных процессов.

Вариантов срабатывания этого перекрестка будет $2^N - 1$, где 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 е-

mail». Либо при срабатывании этого перекрестка может сработать любая другая комбинация выходных процессов.

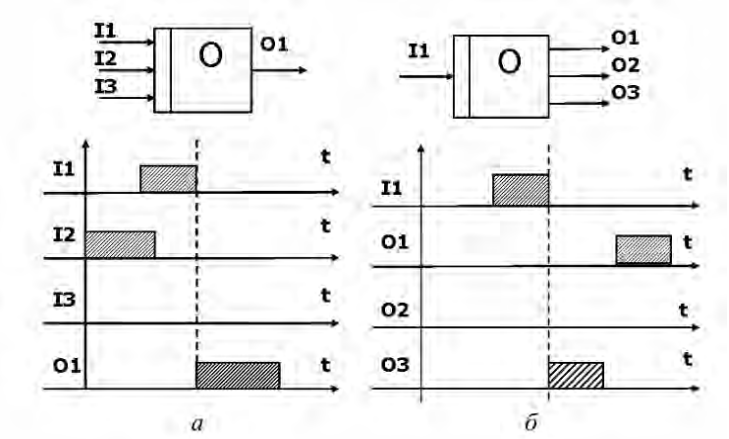


Рисунок 42 - Asynchronous OR (Асинхронное ИЛИ):
а – перекресток слияния; б – перекресток ветвления

Synchronous OR (Синхронное ИЛИ)

Правило срабатывания перекрестка слияния (см. рисунок 43, 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 $2^N - 1$, где N – количество входов перекрестка.

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

Правило срабатывания перекрестка ветвления (рисунок 43, б): после завершения входного процесса запустится один или любая возможная комбинация выходных процессов, но если сработала комбинация процессов, то тогда запуститься она должна одновременно.

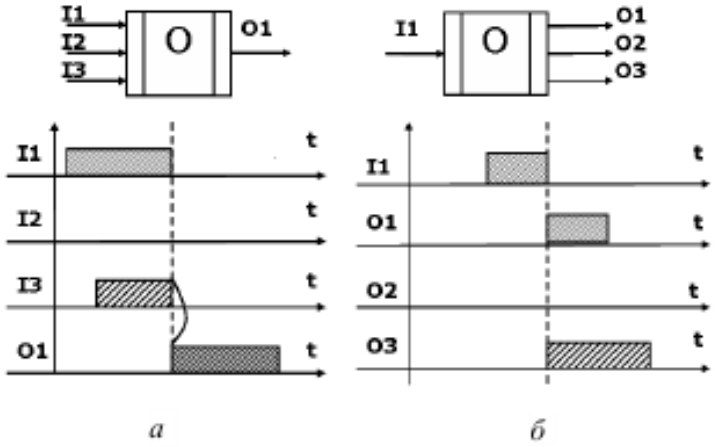


Рисунок 43 - Synchronous OR (Синхронное ИЛИ):
а – перекресток слияния; б – перекресток ветвления

Вариантов срабатывания этого перекрестка будет $2^N - 1$, где N – количество выходов перекрестка.

Exclusive OR (исключающее ИЛИ)

Правило срабатывания перекрестка слияния (см. рисунок 44, а): выходной процесс запустится, если завершился только один входной процесс.

Вариантов срабатывания этого перекрестка – N, где N – количество входов перекрестка.

Правило срабатывания перекрестка ветвления (см. рисунок 44, б): после завершения входного процесса запустится только один выходной процесс.

Вариантов срабатывания этого перекрестка – N, где N – количество выходов перекрестка.

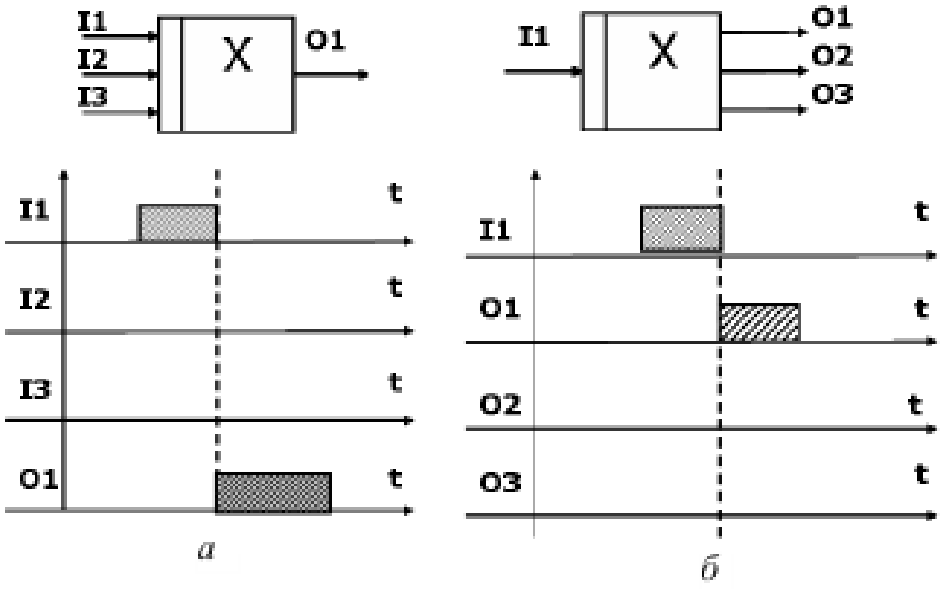


Рисунок 44 - Exclusive OR (исключающее ИЛИ):
а – перекресток слияния; б – перекресток ветвления

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 рисунок 45.

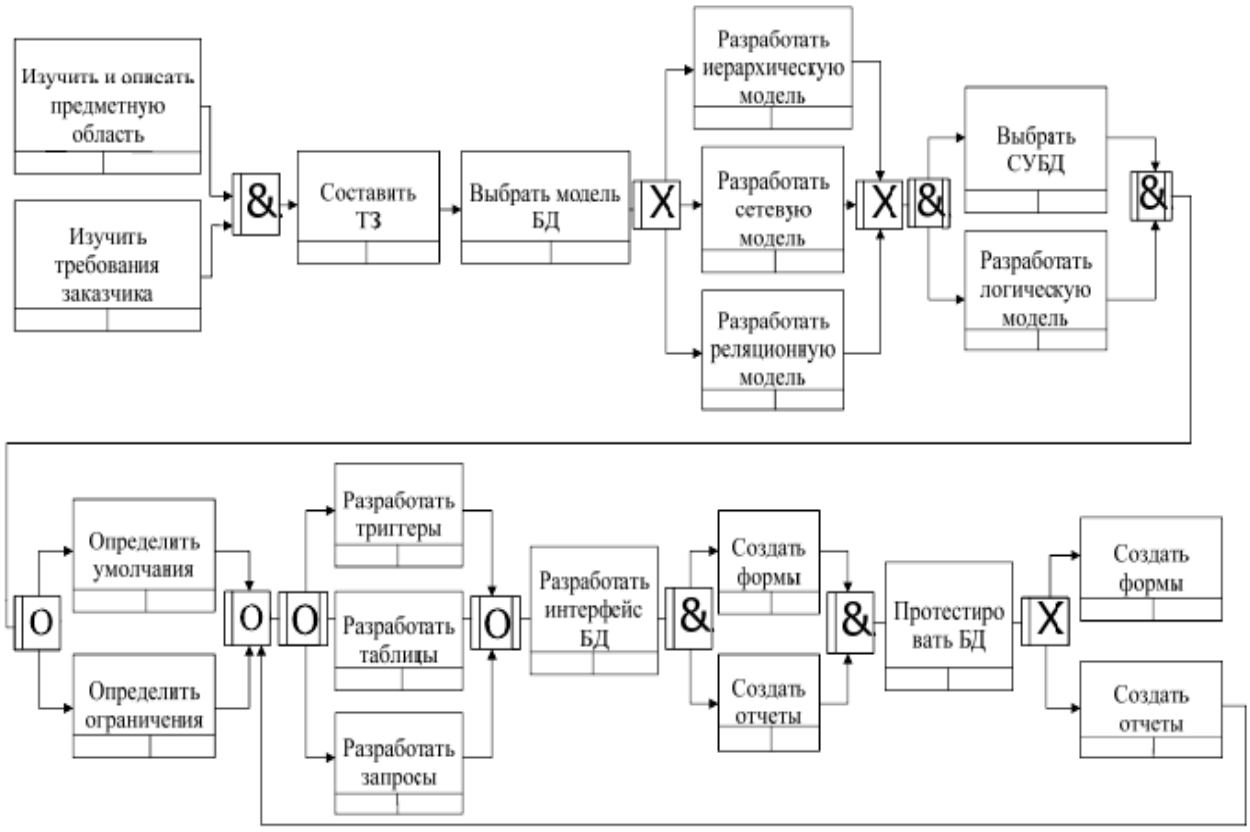


Рисунок 45 - Пример IDEF3-модели разработки базы данных

3.2 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:

- 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. Обе нотации имеют одинаковый по названиям и значению элементный состав, но имеют различное его графическое изображение (таблица 3).

Таблица 3 - 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 еѐ 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 рисунке 46 (контекстная диаграмма) и рисунке 47 (диаграмма основных бизнес-процессов).

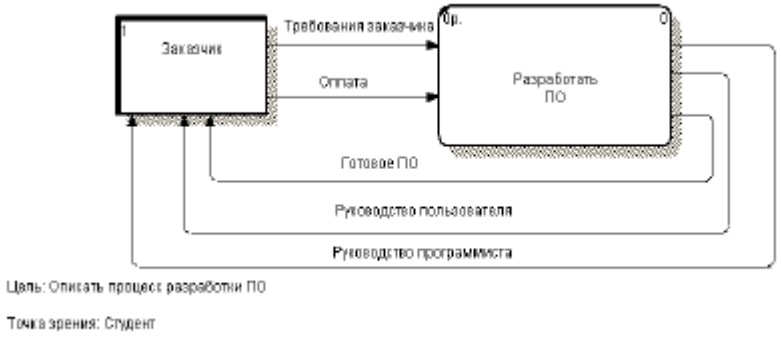


Рисунок 46 - Контекстная диаграмма DFD

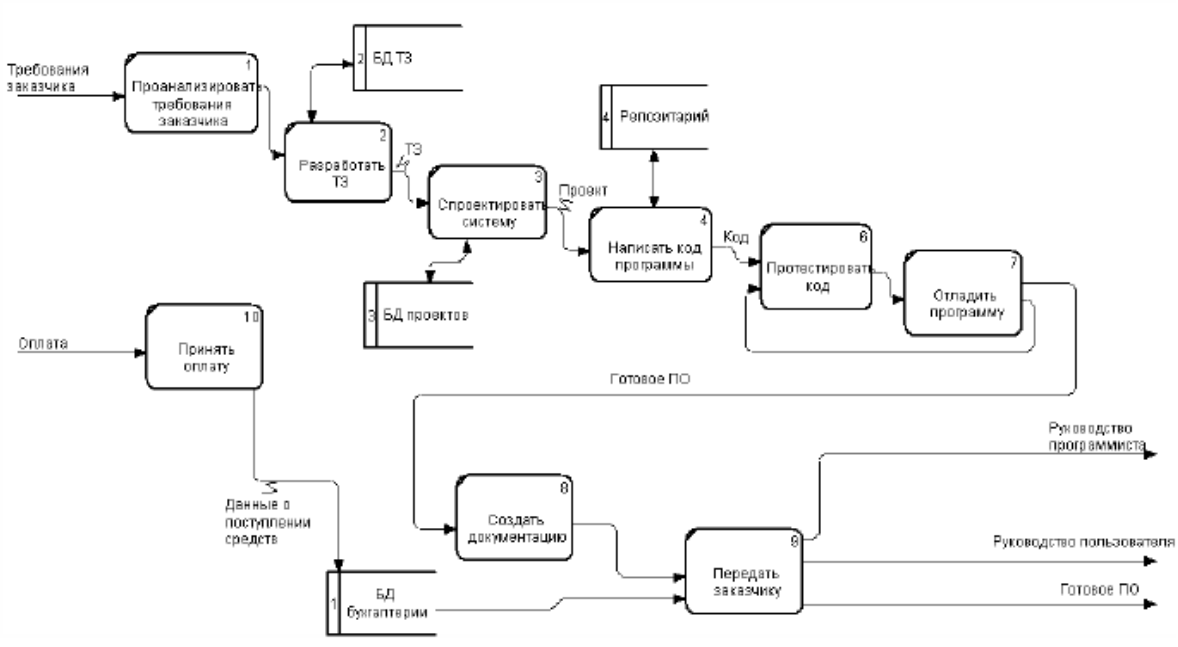


Рисунок 47 - Диаграмма основных бизнес-процессов

3.3 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;

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 моделейпрототипов;

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 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 инструкции 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 рисунке 48.

Рисунок 48 - «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 рисунке 49.

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.

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

Рисунок 49 - Фазовая модель

С учетом фазовой модели «здание» ARIS модифицируется (рисунок 50).

Рисунок 50 - «Здание» ARIS с учетом фазовой модели

3.3.1 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 таблице 4.

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.

Таблица 4 - 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.

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 таблице 5 приведены возможные и допустимые типы отношений в организационной модели между элементами.

Таблица 5 - Типы отношений в организационной модели

	Орг. звено	ФИО	ФИО	Должность	Место	Группа
Орг. звено	Is composed of Is superior	Is belong to	Is belong to	Is composed of	Is located at	Is assigned to
ФИО	Is organization manager for	Is organization manager for	Is organization manager for	Is organization manager for Occupies	Is located at	Is organization manager for
ФИО	Is belong to	Is organization manager for	Is organization manager for	Is organization manager for	Is located at	X
Должность	Is organization manager for	Is organization manager for	Is organization manager for	Is organization manager for	Is organization manager for	Is organization manager for
Место	Is located at	Is located at	Is located at	Is organization manager for	Subsumes	Is located at
Группа	Is assigned to	Has member	Has member	Is composed of	Is located at	Cooperates with

3.3.2 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 таблице 6, а типы отношений в модели дерева функций в таблице 7.

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.

Таблица 6 - Основные элементы в модели дерева функций

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

Таблица 7 - Типы отношений в модели дерева функций

	
	Is process-oriented superior процессно-ориентированное подчинение

3.3.3 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.

Чаще всего, VACD – это модель верхнего уровня (эквивалент контекстной диаграммы А-0). Основные элементы модели VACD приведены в таблице 8.

Таблица 8 - 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

3.3.4 Расширенная событийно-ориентированная модель (eEPC)

eEPC (extended Event-driven Process Chain)-модель применяется для подробного описания бизнес-логики процесса с использованием четырех групп элементов (рисунок 51): функциональные элементы, логические элементы, элементы данных и организационные элементы.

Графическое изображение элементов приведено в таблице 9.

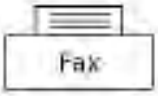
При использовании всех групп элементов получается всесторонняя модель бизнес-процесса, показывающая основные действия, выполняемые конкретными сотрудниками с помощью прикладных и технических устройств.

Рисунок 51 - Группы элементов в eEPC-модели

Таблица 9 - 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)

Продолжение таблицы 9

 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)	Имеется 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»:

3.3.5 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;

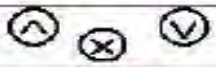
− физические лица или группы людей, принимающие участие в выполнении описываемого процесса;

− средства связи, посредством которых информация поступает в систему.

3.3.6 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 (таблица 10).

Таблица 10 - 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			

3.3.7 С3-модель

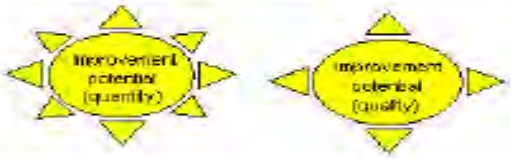
С3-модель используется для описания процессов реорганизации деятельности предприятия. Для создания С3-модели выделяется отдельный бизнес-процесс, который детализируется с целью дальнейшего реинжиниринга.

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 таблице 11.

Таблица 11 - 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
 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

Продолжение таблицы 11

 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.

3.3.8 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 бизнесфункций, которые также могут декомпозироваться (рисунок 52).
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 (рисунок 53 и рисунок 54).

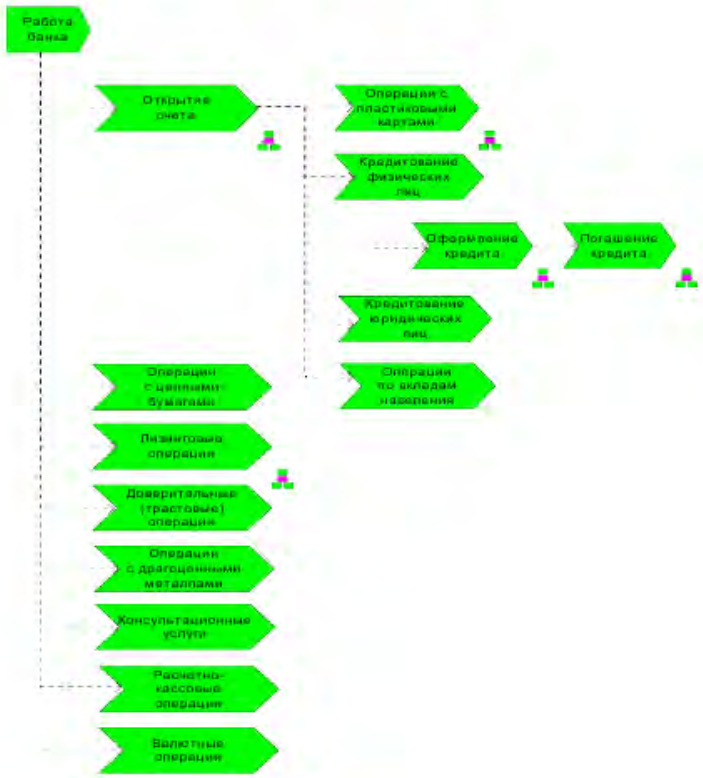


Рисунок 52 - Модель VACD – основные виды деятельности (бизнес-процессы) коммерческого банка

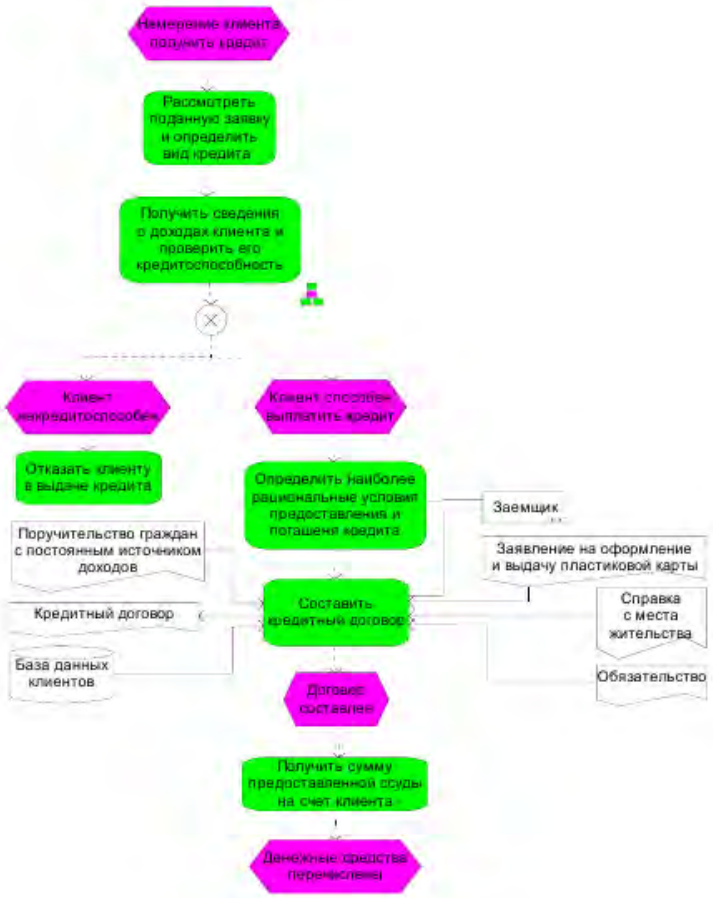


Рисунок 53 - eEPC-модель процесса оформления

Рисунок 54 - 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» (рисунок 55), так как

именно этот простейший процесс имеет 11 присоединенных элементов, которые несомненно загромоздили бы eEPC-модель.

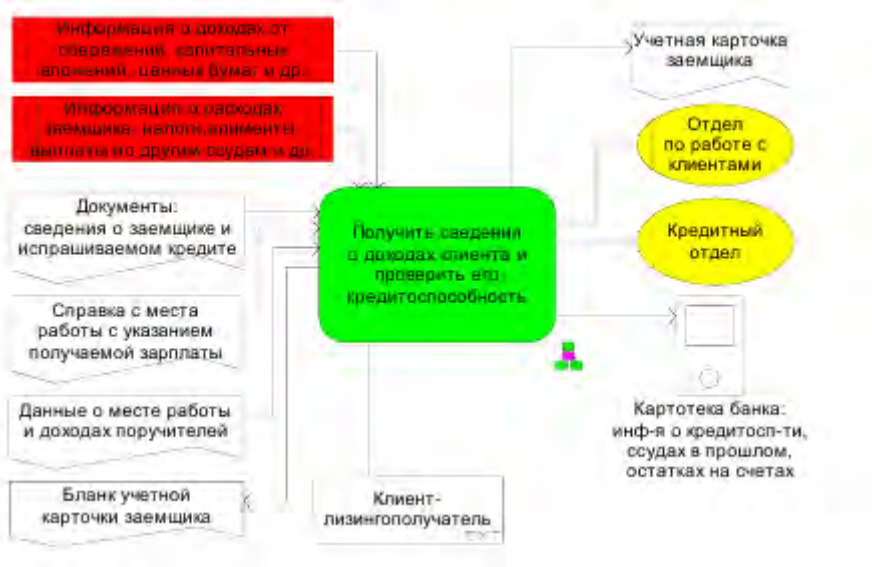


Рисунок 55 - FAD-модель процесса «Получить сведения о доходах клиентаи проверить его кредитоспособность»

КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Что такое SADT, и как SADT связана с IDEF?
2. Перечислите основные структурные элементы IDEF0методологии.
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? 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 бизнеспроцессов предприятия?
17. Перечислите основные типы моделей, предложенные разработчиками концепции ARIS.
18. Каким образом связан процесс реинжиниринга бизнеспроцессов предприятия и концепция ARIS?
19. В чем заключается разница в понятиях «реорганизации» и «реинжиниринга»?
20. Что такое «бизнес-процесс»? Дайте определение этому термину.

4 Имитационное моделирование систем

4.1 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.

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.

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.

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:

– проектирование и анализ производственных систем;

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

– определение требований к оборудованию и протоколам сетей;

– 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 (тренажѐрах).

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.

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.

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

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

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

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

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

5. 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.

Современные программные пакеты поддерживают следующие функциональные возможности:

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

– генерирование случайных величин с заданными распределениями вероятностей;

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

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

– определение и изменение атрибутов объектов и глобальных переменных;

– использование заложенных и созданных пользователем математических выражений и функций;

– создание собственных логических конструкций и использование стандартных схем;

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

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

– имеющаяся хорошо структурированная документация по пакету.

К сожалению, несмотря на неоспоримые достоинства имитационного моделирования, в настоящее время в России этот метод исследования сложных систем используется мало, это связано с тем, что разработка таких моделей требует больших временных и стоимостных затрат. Но тенденции последнего времени вселяют надежду на то, что ситуация изменится и имитационное моделирование в России будут также широко и активно использовать, как в США, Канаде и Европе. Именно для того чтобы компенсировать этот пробел российской действительности, в курсе «Компьютерное моделирование» рассматриваются средства имитационного моделирования на примере мощного программного пакета (ПП) Arena 7.0.

4.2 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.

4.2.1 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.

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. 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, ещѐ называемых требованиями или фишками, в систему.

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 таблице 12 приведены элементы СМО.

Таблица 12 - 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:

– 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 еѐ 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. 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, взлетнопосадочные 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.

4.2.2. 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.

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.

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, причем множество вершин этого графа разбивается на два подмножества и не существует дуги, соединяющей две вершины из одного подмножества.

Итак, сеть Петри – это набор

$$N = (T, P, A),\ T \cap P = \varnothing \qquad (3)$$

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 ⊆ (T×P) ∩ (P×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 таблице 13.

Таблица 13 - 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 рисунке 56 приведен пример интерпретации сети Петри.

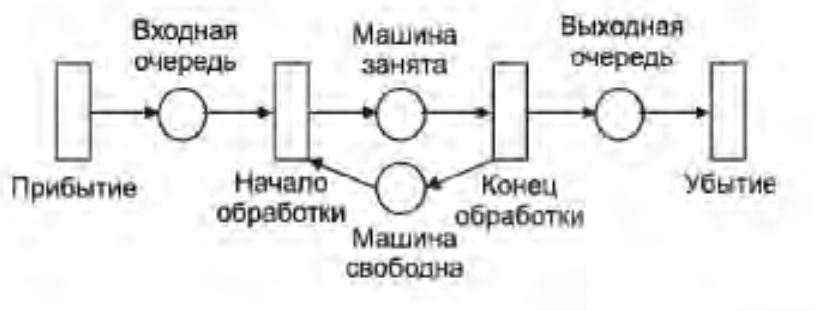


Рисунок 56 - Интерпретация сети Петри

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.

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;

– 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 допусти-

мых 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.

Система Arena позволяет моделировать виды деятельности, представленные на рисунке 57.

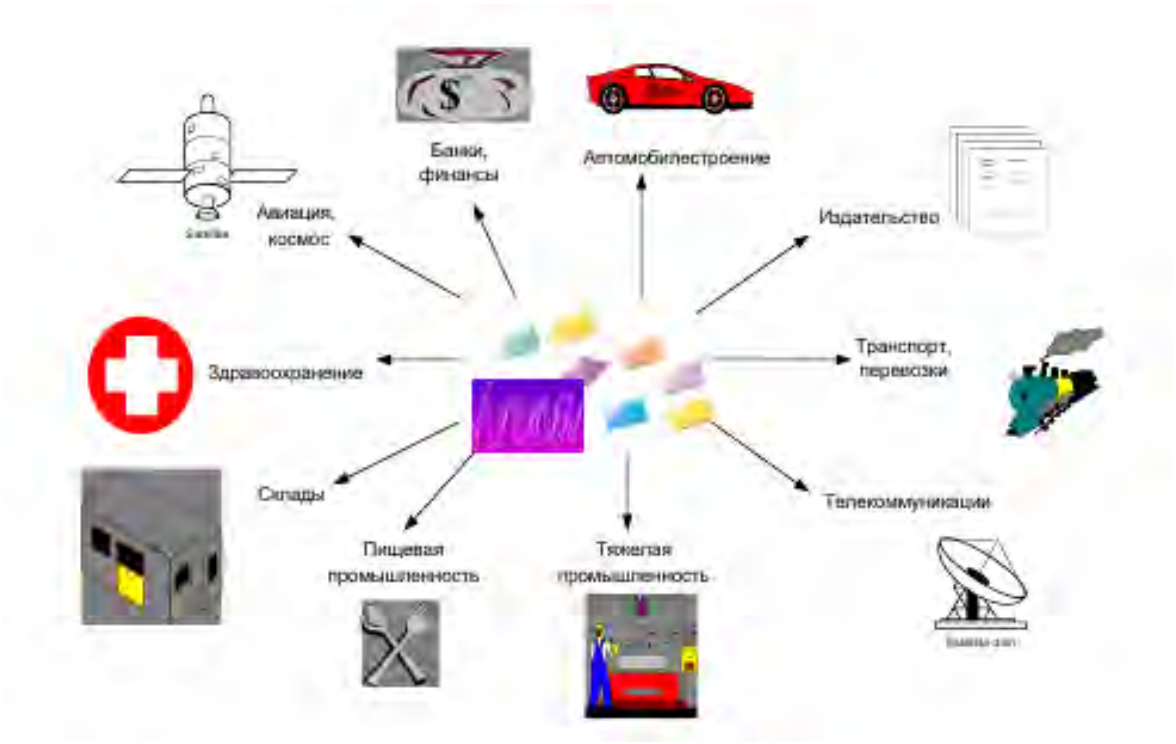


Рисунок 57 - Области применения Arena

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.

4.3 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 рисунке 58.

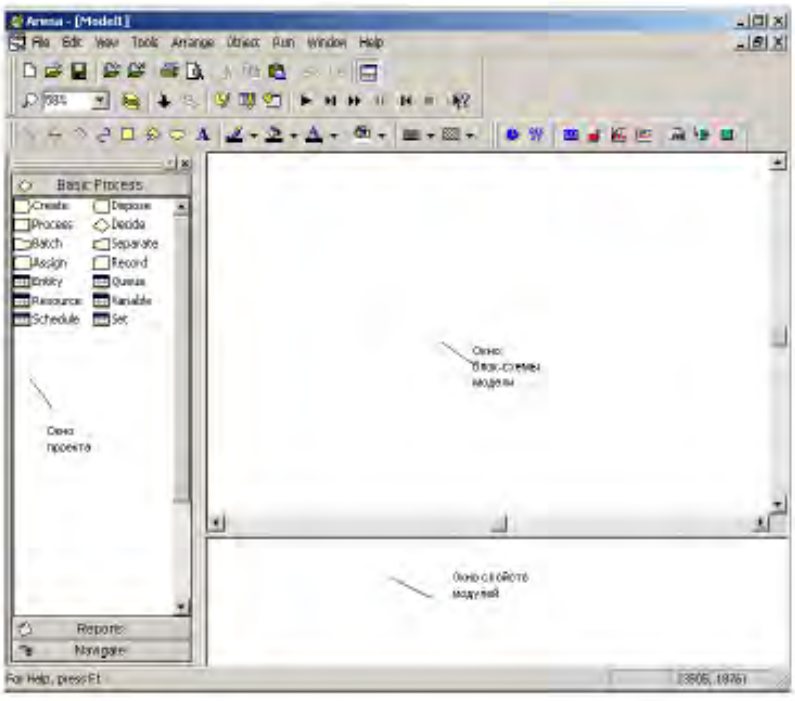


Рисунок 58 - Среда моделирования Arena

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.

4.4 Basic Process Panel (панель основных процессов)

4.4.1 Схемные модули

Модуль Create

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

Таблица 14 - 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

Продолжение таблицы 14

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 какието 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 15 - 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
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

Продолжение таблицы 15

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 122 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 16 - 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 17 - 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 дублика-

тов сущности. У дублированной сущности знае анимационная картинка такие же, как и оригинала. Оригинальная сущность также покидает модуль.

2. Для разделения ранее сгруппированных сущностей. Правило для 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.

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 18 - 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

Таблица 19 - 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, какое количество работы, совершаемое за один час.

Таблица 20 - 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 Dis

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 21 - 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

4.4.2 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 22 - 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 23 - 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). Вовторых, 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 24 - 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
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 25 - 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 26 - Параметры модулям 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

КОНТРОЛЬНЫЕ ВОПРОСЫ

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?

Список литературы

1. Замятина О. М. Моделирование систем: Учебное пособие. – Томск: Изд-во ТПУ, 2009.
2. Каменнова М. С., Громов А. И., Ферапонтов М. М., Шматалюк А. Е. Моделирование бизнеса. Методология ARIS. – М.: Весть-Мета Технология, 2001.
3. Советов Б. Я. Моделирование систем: учебник для вузов. – 3-е изд., перераб. и доп. – М.: Высш. шк., 2001.
4. Бешенков С. А. Моделирование и формализация: методическое пособие. – М.: Лаборатория базовых знаний, 2002.