

«Теория информационных процессов и систем»

© Прокимов Н.Н., 2011

© Московский финансово-промышленный университет «Синергия», 2011

Содержание

Аннотация

Тема 1. Основные понятия

Вопрос 1. Понятие информационной системы.

Вопрос 2. История появления и развития информационных систем.

Вопрос 3. Классификация информационных систем.

Вопрос 4. Жизненный цикл информационной системы.

Вопросы для самопроверки:

Литература по теме:

Тема 2. Информационная система как сложная система

Вопрос 1. Основные понятия теории систем.

Вопрос 2. Сложная система.

Вопрос 3. Модели систем.

Вопрос 4. Методология системного анализа.

Вопрос 5. Агрегатное описание систем.

Вопрос 6. Иерархичность информационных систем.

Вопросы для самопроверки:

Литература по теме:

Тема 3. Качественные методы системного анализа в теории информационных процессов и систем

Вопрос 1. Структуризация целей.

Вопрос 2. Мозговой штурм.

Вопрос 3. Экспертное оценивание.

Вопрос 4. Метод Дельфи.

Вопросы для самопроверки:

Литература по теме:

Практические задания.

Тема 4. Количественные методы системного анализа в теории информационных процессов и систем

Вопрос 1. Количественное оценивание систем.

Вопрос 2. Оценивание в условиях определенности.

Вопрос 3. Оценивание в условиях риска.

Вопрос 4. Оценивание в условиях неопределенности.

Вопросы для самопроверки:

Литература по теме:

Практические задания.

Тема 5. Моделирование деловых процессов

Вопрос 1. Деловые процессы и их моделирование.

Вопрос 2. Методология моделирования IDEF.

Вопрос 3. Язык моделирования UML.

Вопрос 4. Автоматизация моделирования деловых процессов.

[Вопросы для самопроверки:](#)

[Литература по теме:](#)

[Практические задания.](#)

[Тема 6. Моделирование потоков работ](#)

[Вопрос 1. Концепция потоков работ.](#)

[Вопрос 2. Моделирование на основе сетей Петри.](#)

[Вопрос 3. Высокоуровневые сети Петри.](#)

[Вопрос 4. Модель сетевого планирования.](#)

[Вопросы для самопроверки:](#)

[Литература по теме:](#)

[Практические задания.](#)

[Тема 7. Имитационное моделирование информационных процессов и систем](#)

[Вопрос 1. Метод статистических испытаний.](#)

[Вопрос 2. Концепции имитационного моделирования.](#)

[Вопрос 3. Создание имитационных моделей с помощью систем моделирования.](#)

[Вопрос 4. Примеры имитационных моделей.](#)

[Вопросы для самопроверки:](#)

[Литература по теме:](#)

[Практические задания.](#)

[Список сокращений](#)

[Глоссарий](#)

[Приложение](#)

Аннотация

Дисциплина «Теория информационных процессов и систем» разработана на основе учебной программы дисциплины «Теория информационных процессов и систем» МФПУ «Синергия» с учетом государственного образовательного стандарта по специальности «Информационные системы», утвержденного Министерством образования и науки Российской Федерации, и предназначена для студентов факультета информационных систем и технологий МФПУ «Синергия».

Дисциплина входит в состав цикла специальных дисциплин. Она посвящена изучению основных понятий, методов и инструментальных средств анализа и проектирования информационных систем, приемов и математических моделей, применяемых для решения связанных с этим процессом задач. Дисциплина формирует общую систему теоретических и концептуальных представлений о методологической основе теории, а также развивает ряд практических навыков и умений, позволяющих студентам впоследствии применять полученные знания и навыки для решения прикладных задач в своей области деятельности.

Целью изучения дисциплины является углубленное изучение постановок основных задач теории, моделей информационных процессов и систем и методик их решения, подготовка студентов к профессиональной деятельности в сфере создания и применения информационных систем и технологий, используемых в организациях различного профиля и форм собственности.

Задачи спецкурса:

- раскрытие сущности и содержания основных понятий и определений теории информационных процессов и систем;
- ознакомление с основными практическими приложениями теории;
- изучение основных математических моделей, применяемых в теории информационных процессов и систем;
- формирование навыков самостоятельной и коллективной работы студентов по решению типичных задач в области анализа и проектирования информационных процессов и систем на основе моделей и подходов теории.

В результате изучения дисциплины студенты должны:

- **знать:** классификацию информационных систем, постановку основных задач анализа и проектирования информационных систем, содержательные функции, математические модели теории и области применения моделей и методов;
- **уметь:** правильно анализировать ситуацию и формулировать задачи анализа и синтеза информационных систем, проводить рациональный выбор моделей и методов решения поставленной задачи, интерпретировать и анализировать результаты моделирования информационных процессов и систем;
- **иметь представление:** об особенностях и предмете изучения теории информационных процессов и систем как науки, универсальных законах теории, методологии системного анализа применительно к задачам исследования и проектирования информационных систем и наиболее важных моделях, применяемых для решения задач.

Тема 1. Основные понятия

Цели изучения темы:

- познакомиться с понятиями «информационная система» и «информационный процесс»;
- выяснить основные задачи теории.

Задачи изучения темы:

- понять предмет изучения и основную цель теории информационных процессов и систем;
- овладеть основными терминами и понятиями.

Успешно изучив тему, Вы:

получите представление о:

- основных исторических этапах развития информационных систем;
- классификации информационных систем;
- жизненном цикле информационных систем;

будете знать:

- какие требования предъявляются к информационным системам;
- как используются информационные системы;
- типы систем с управлением.

Вопросы темы:

1. Понятие информационной системы.
2. История появления и развития информационных систем.
3. Классификация информационных систем.
4. Жизненный цикл информационной системы.

Вопрос 1. Понятие информационной системы.

Информация представляет собой важнейший **стратегический ресурс** любой организации. Едва ли сегодня можно найти предприятие даже очень скромное по своим размерам, которое не имело бы персонального компьютера, доступа в интернет, копировально-множительной и другой оргтехники. Все эти средства позволяют значительно повысить эффективность работы предприятий, на которых они используются.

Используемый во многих производных от него терминах термин *информация* имеет смысл, отличный от смысла, который имеет термин *данные*. Данные представляют собой «сырье», необходимое для получения «продукта» в виде информации. Они не наделены содержательным смыслом, это просто числа или наборы символов, необходимые для указания определенных свойств некоторого объекта, например, номера заказа, табельного номера сотрудника, вида производимого на предприятии продукта, размера выплачиваемого пособия и т.п. Данные могут возникать как результат некоторых вычислений, например:

$$\text{Выплата} = \text{Оклад} + \text{Надбавка} + \text{Премия} - \text{Налог}$$

Если данные могут быть организованы и представлены в виде, пригодном для их восприятия субъектом, то они приобретают полезные для последнего свойства, позволяющие рассматривать их уже как **информацию**. Эти свойства позволяют извлекать определенный **смысл**, как из самих данных, так и из результатов, производимых с ними манипуляций.

Энциклопедия *Britannica* определяет **информационную систему** как *единое множество компонентов для сбора, хранения, обработки и передачи информации*. Информация необходима для управления предприятием, обеспечения проектирования, организации учебного процесса, удовлетворения потребностей отдельных субъектов или социальных групп.

Процессы сбора хранения, обработки и передачи информации, протекающие в информационных системах, называются **информационными процессами**.

Основными **компонентами** информационных систем являются **компьютерные** и **телекоммуникационные** средства и технологии - аппаратное и программное обеспечение компьютеров, базы данных, телекоммуникационные системы, человеческие ресурсы и процедуры, а также **способы взаимодействия людей** с этими технологиями для поддержки деловых процессов (более подробно этот термин будет обсуждаться далее).

Постоянно возрастающее значение информации и возникновение новых форм деловой деятельности, таких как электронная коммерция, привели к пересмотру роли и значения информационных систем. Теперь информационные системы уже не просто обеспечивают выполнение отдельных работ и операций, из которых складывается деятельность организации, а фактически становятся одной из ее наиболее важных **органических составляющих**, без которой продолжение нормальной работы оказывается невозможным. Степень влияния, которое оказывают информационные системы, трудно переоценить. Их создание и внедрение существенно изменяет применяемую технологию, организационную структуру предприятий, состава и численности персонала, характере взаимодействия предприятий между собой и со своими контрагентами.

Информационная система, как и всякая система, должна отвечать определенным требованиям и обладать определенным набором показателей эффективности. Из основных **требований**, которые в большинстве случаев предъявляются к информационным системам, следует выделить такие как:

- максимально точное соответствие функциональных возможностей и параметров системы целям функционирования владеющей ею организации и требованиям со стороны ее пользователей;
- простота адаптации системы к изменениям, протекающим в динамичной среде;
- актуальность, точность, надежность и доступность хранимой, обрабатываемой и представляемой информации;
- приемлемые стоимостные характеристики, под которыми понимаются, в первую очередь, затраты на создание системы и стоимость владения ею.

На рис. 1 отражает взгляд на информационную систему с точки зрения ее применения в деловой деятельности предприятия.

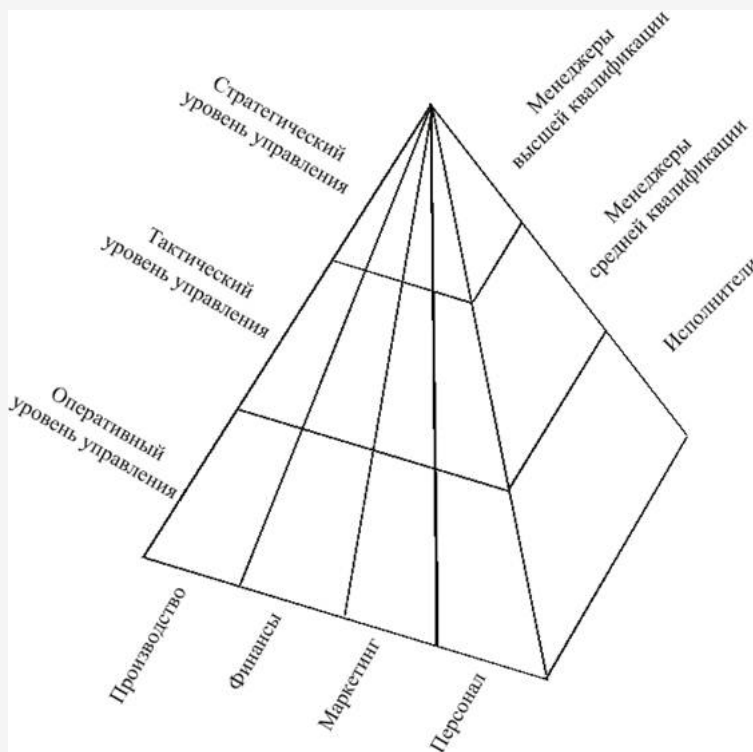


Рис. 1. Использование информационной системы

В основании пирамиды находится информационная система. На нижнем уровне выполняется оперативная работа с данными и осуществляется (руководителями нижнего звена) оперативное управление этой работой. Ближе к вершине пирамиды

осуществляется решение задач, необходимых для стратегического управления, характерной особенностью которых является их плохая структурированность. По этой причине с ростом уровня управления объем работ, выполняемых специалистом или руководителем с помощью информационной системы, уменьшается. Вместе с тем, существенно увеличивается значимость принимаемых ими решений.

Вопрос 2. История появления и развития информационных систем.

Развитие информационных систем шло по нескольким относительно независимым направлениям.

Системы обработки данных в основном использовались при создании систем сбора и обработки статистической информации и аналогичных по характеру систем. Они являются наиболее простым типом информационных систем и предназначены для решения **хорошо структурированных задач**, для которых имеются входные данные, известны алгоритмы, ведущие к решению задач. В настоящее время термином *обработка данных* пользуются на уровне оперативного управления предприятием (решение задач бухгалтерского учета, статистической отчетности, учета валютных операций в банке и т.п.). Основными функциями в этих системах являются сбор данных и перенос их на машинные носители, передача в места хранения и обработки, хранение, обработка информации по стандартным алгоритмам, вывод и представление информации пользователю в виде регламентных форм.

По мере увеличения памяти ЭВМ начало стремительно развиваться направление, связанное с организацией **систем баз данных**. По мере его развития появились термины *базы знаний*, *базы целей* с расширенной трактовкой задач, которые решались с помощью систем баз данных.

Начиная с шестидесятых годов прошлого века разработка информационных систем в нашей стране шла по двум основным направлениям:

- 1) автоматизированных информационных систем как составной части *автоматизированных систем управления*;
- 2) *автоматизированных систем научно-технической информации*. Задача была поставлена и решалась системно: были разработаны классификация информационных систем, система стандартов и нормативной документации.

Автоматизированные информационные системы создавались как **фактографические** информационно-поисковые системы, имеющих дело с конкретными сведениями об объектах, событиях или явлениях (например, дата рождения, масса, вес человека). Автоматизированные информационные системы создавались как первый уровень **автоматизированных систем управления**. Информация предоставлялась пользователям в виде регламентированных форм, сгруппировано в соответствии с решаемыми прикладными задачами. В большинстве случаев ввод и вывод осуществлялись с помощью специальных форм документов. Работы за рубежом развивались преимущественно в области создания программных средств автоматизации бухгалтерского учета, складских операций и других функциональных составляющих деятельности предприятий, совершенствования вычислительной техники, рациональной организации информационных массивов, пользовательского интерфейса.

Автоматизированные системы научно-технической информации относятся к категории **документальных**, в которых информация хранится в виде отдельных документов. Работы по их созданию были обусловлены возрастающим значением информации во всех сферах деятельности и велись в нашей стране в рамках общегосударственной автоматизированной системы научно-технической информации, имеющих многоуровневую структуру. Развитие этого направления за рубежом сначала шло по линии создания отдельных центров сбора и анализа информации, затем появилась тенденция к их объединению.

С начала девяностых годов возрастает роль новых видов информации - нормативно-правовой и нормативно-методической, регламентирующей деятельность предприятий при предоставлении им юридической и экономической самостоятельности, для чего разрабатываются **системы нормативно-методического обеспечения** управления. Учитывая важную роль нормативно-правовой информации при внедрении рыночных принципов управления экономикой, создаются автоматизированные системы нормативно-правовой (типа систем «Консультант», «Кодекс», «Гарант» и т.п.) и экономической информации, бухгалтерские и банковские информационные системы, информационные системы рынка ценных бумаг и другие.

Вопрос 3. Классификация информационных систем.

Классифицировать информационные системы можно различными способами и с разной степенью глубины детализации. Приведем некоторые из основных классификаций (рис. 2), необходимые для понимания материала настоящего курса, ограничившись одним уровнем иерархии.



По степени автоматизации:

- **Ручные** системы средств автоматизации выполнения операций не имеют.
- **Автоматизированные** системы автоматизируют выполнение операций частично.
- **Автоматические** системы автоматизируют выполнение операций полностью.

По архитектуре:

- **Файл-серверные** системы хранят данные в виде отдельных файлов операционной системы на специально выделенном компьютере (файл-сервере), на котором могут также храниться и исполнительные модули. Доступ к файлам осуществляется обычным для файловой системы способом в виде задания пути к файлу и указания необходимой файловой операции.
- **Клиент-серверные** системы используют концепцию, согласно которой выделяется сторона, предоставляющая по запросу определенную услугу (**сервер**), и сторона, запрашивающая и получающая услугу (**клиент**). Сервер реализуется на одном из компьютеров вычислительной сети, преимущественно (хотя и необязательно) на отдельном компьютере (*выделенный сервер*). Запрос услуги является для клиента «прозрачным» (никакие физические параметры как в случае архитектуры файл-сервер указывать не нужно), сервер обрабатывает запрос и отправляет результат обработки клиента точно в том виде, как он был затребован (что значительно снижает нагрузку на сеть в отличие от архитектуры файл-сервер, где пересылаются большие объемы ненужной информации).
- **Трехслойная архитектура** основана на разделении всех компонентов системы по трем слоям (рис. 3): слоя представления, или вида (View layer), слоя источника данных (Data Source Layer) и слоя логики домена, или деловой логики (Domain Layer). Слой **представления** отвечает за взаимодействие пользователя с системой (ввод и интерпретация, преобразование вида представления и отображение данных). Слой **источника данных** отвечает за организацию интерфейса со слоем деловой логики и интерфейса внешних систем с источником данных. Слой **деловой логики** содержит описания совокупности правил, необходимых для выполнения прикладных функций, и может быть реализован как на стороне выделенного сервера, так и на стороне клиента. Архитектура обеспечивает сочетание низких требований к вычислительным ресурсам клиента с высокой гибкостью.



Рис. 3. Трехслойная архитектура

По охвату применением:

- **Персональные (однопользовательские)** системы автоматизируют работу одного пользователя. Реализуются, как правило, либо на основе специализированных «коробочных» пакетов прикладных программ (пакет для индивидуального предпринимателя, составление расписания занятий, планирование ремонтных работ), либо офисных программ типа табличного процессора (например, Microsoft Excel). В последнем варианте пользователь сам осуществляет как создание нужных ему приложений, так и их сопровождение.
- **Групповые** системы автоматизируют работу коллектива людей (структурное подразделение, проектная бригада). Реализуются, как правило, в виде автоматизированных рабочих мест членов группы с определенными функциональными обязанностями (например, рабочие места «Бухгалтер», «Кладовщик», «Кассир») с возможностью работы с общими информационными массивами.
- **Корпоративные** системы обеспечивают комплексную автоматизацию деятельности предприятия (корпорации), иначе говоря, автоматизацию всех деловых процессов. Корпоративные системы реализуются как интегрированный набор приложений единой информационной среды с поддержкой всех функций, показанных на рис. 1, и являются продуктом проектирования специализированных компаний ИТ-индустрии. Характерными для этих систем являются высокая степень адаптации к корпоративной среде, уровня информационной безопасности и доступности, открытость, масштабируемость, независимость от территориального расположения предприятия, его подразделений и филиалов.

По степени структуризации решаемых задач:

- **Хорошо структурированные (формализуемые)** задачи характеризуются строгой математической постановкой, точной спецификацией исходных данных и алгоритмом решения. Задачи этого типа обычно довольно просты и решаются часто (например, расчет заработной платы сотрудников).
- **Слабо (частично) структурированные** задачи характерны для большинства информационных систем. Такие задачи допускают структуризацию только некоторой своей части, решение которой может быть автоматизировано. Решение задач другой части осуществляется человеком.
- **Неструктурированные** задачи не допускают построения математической модели своего решения.

По источникам происхождения (способу создания):

- **Наследованные (legacy)** системы. Это системы, которые находятся в эксплуатации и которые следует, по возможности, адаптировать к сформулированным требованиям в дальнейшем с целью сохранить сделанные вложения.
- **Типовые (стандартные)** системы представляют собой готовые прикладные программные продукты (например, система бухучета), предлагаемые их поставщиками на рынке. Обычно эти системы обходятся дешевле, но нуждаются в

настройке (оптимизации) к специфическим требованиям внедряющей организации.

- **Уникальные (новые)** системы проектируются и разрабатываются «с нуля» в соответствии со специальными требованиями (техническим заданием) и обеспечивают максимально возможную адаптацию к среде внедряющей организации.

В настоящее время наблюдается тенденция к стиранию явных различий между классами создаваемых и применяемых систем. Системы все чаще сочетают в себе признаки и возможности систем разных типов, поскольку для управления современным предприятием требуется использовать информацию всех разновидностей. На первый план при анализе систем выдвигается **степень интегрированности** информационной системы в среду организации, иначе говоря, то, насколько полно в системе используются знания о процессах и операциях автоматизируемой деятельности. В соответствии с этим подходом можно выделить следующую иерархию (в порядке возрастания степени интегрированности) систем.

Офисные информационные системы. Системы этой категории предназначены для выполнения базовых операций по обработке информации типа создания и редактирования текстовых и графических файлов, хранения и передачи файлов, пересылки сообщений. Для этого применяются текстовые (типа Microsoft Word) и табличные (типа Microsoft Excel) процессоры, программы работы с несложной графикой (типа Microsoft Visio) и базами данных (типа Microsoft Access) и работы с почтой (типа Microsoft Outlook). Внутри этих систем отсутствует какая-либо привязка к среде.

Системы обработки транзакций. Системы этого типа обеспечивают фиксацию каких-либо изменений в основных процессах, передачу и сохранение сведений об изменениях для чего они используют некоторые данные о процессах. Центральным компонентом этих систем является система управления базами данных, к которым в усовершенствованных версиях может добавляться также и такой компонент как система управления потоком работ (концепция потока работ будет рассматриваться позднее в настоящем курсе).

Системы управления знаниями. Такие системы предоставляют возможность накапливать знания и применять их специалистами и лицами, принимающими решения (ЛПР). Примером могут служить системы управления документами (например, законодательными), позволяющие отыскивать наиболее точно отвечающие запросу (релевантные) документы. Часто в таких системах применяются *хранилища данных* – базы агрегированных данных в виде *многомерных кубов* (например, число покупателей определенных видов продукта в определенных регионах за определенные периоды). Источниками данных для хранилищ являются статистические данные (в приведенном примере, источником может служить корпоративная база данных по продажам).

Системы поддержки принятия решений. В этих системах решение получается как результат человеко-машинного взаимодействия, причем, в качестве ЛПР могут выступать лица различного уровня управления. Различают два типа систем данного класса: *системы, основанные на математических моделях*, и *экспертные системы*. Примером систем первого типа являются системы планирования бюджета, системы производственного планирования. Примером экспертной системы может быть система диагностики неисправности автомобиля на станции технического обслуживания.

Управляющие системы. Процедуры принятия и исполнения решений в таких системах (для них могут применяться и другие названия, например, системы автоматического управления, системы управления технологическими процессами, системы программного управления) выполняются полностью автоматически в соответствии с набором строгих правил (алгоритмов). Примерами могут служить биллинговая система, система электронного бронирования авиабилетов, система контроля температуры и влажности помещения.

Тип создаваемой системы должен определяться на основе результатов анализа, проводимого на начальном этапе *жизненного цикла*.

Вопрос 4. Жизненный цикл информационной системы.

Информационная система, как и любой объект или субъект, существует в течение некоторого отрезка времени, который проходит с момента возникновения потребности в системе до снижения эффективности функционирования и ее ликвидации. Этот отрезок времени принято называть **жизненным циклом**. В течение этого времени система проходит ряд более или менее четко различаемых этапов. По этой причине под жизненным циклом часто понимают просто **последовательность этапов**.

Существует целый ряд подходов к трактовке жизненного цикла системы. Будем (без ограничения общности дальнейшего рассмотрения) считать жизненный цикл состоящим из следующих этапов.

Системный анализ.

На данном этапе проводится подробное изучение деловых процессов предприятия и определяются варианты их усовершенствования. Для анализа создаются модели существующего и модернизированного вариантов, на основе чего определяются требования к информационной системе. Требования образуют две группы – функциональные и нефункциональные.

Под **функциональными** требованиями понимается совокупность *функций*, выполнение которых должно обеспечиваться информационной системой для протекающих деловых процессов.

Нефункциональные требования распространяются на все выполняемые системой функции и включают требования к таким ее характеристикам как *устойчивость функционирования, безопасность, доступность и производительность*.

Сформулированные требования подвергаются анализу на предмет их реализуемости на последующих этапах.

Необходимо заметить, что название *системный анализ* в данном контексте означает стадию жизненного цикла информационной системы и не равнозначен понятию системного анализа как универсального метода решения сложных задач,

Системное проектирование.

На этом этапе, который является одним из наиболее продолжительных, создается проект будущей системы.

Разработка системы.

На этом этапе создаются все входящие в систему компоненты на основе созданного на предыдущем этапе проекта.

Внедрение системы.

На этом этапе система запускается в эксплуатацию. В зависимости от конкретных условий выбирается та или иная стратегия внедрения.

Сопровождение системы.

Сопровождение преследует цель устранения не выявленных на предыдущих этапах ошибок и недочетов.

Обратим внимание на тот факт, что выше рассматривались этапы жизненного цикла *информационной системы*. Существуют также и стандарты жизненного цикла программного обеспечения, которые могут существенно отличаться от описанной последовательности. Различные методологии и стандарты, применяемые в области информационных технологий, могут по-разному определять этапы и состав задач каждого этапа, а также предусматривать свою логику следования этих этапов и определять их результаты.

Выводы:

1. Информационные системы являются одним из важнейших стратегических ресурсов организаций и предприятий. Нормальная работа в отсутствие современных информационных технологий и средств их поддержки в современных условиях практически невозможна.
2. Внедряемые в практическую деятельность современного предприятия или иной структуры информационные системы должны отвечать ряду строгих требований, включая такие как функциональность, адаптируемость, надежность, безопасность и экономичность.
3. Наряду со значительным прогрессом специализированных информационных технологий существует тенденция к стиранию четких границ между отдельными типами информационных систем.
4. Возрастающая роль информации и серьезность требований к организации и автоматизации информационных процессов в различных сферах деятельности современного общества обуславливает необходимость построения и применения методических и инструментальных средств решения задач, связанных с созданием информационных систем. Подходы к построению моделей и методик во многом обусловлены стадиями жизненного цикла информационной системы.

Вопросы для самопроверки:

1. Что понимается под термином *информация*, чем она отличается от *данных*?
2. Как можно определить понятия *информационная система*, *информационный процесс*?
3. Какие основные компоненты образуют информационную систему?
4. Что входит в число основных требований, предъявляемых к информационным системам?
5. Охарактеризуйте основные этапы развития информационных систем.
6. По каким признакам могут классифицироваться информационные системы?
7. Как можно классифицировать информационные системы по степени автоматизации?
8. Как можно классифицировать информационные системы по архитектуре?
9. Как можно классифицировать информационные системы по охвату автоматизацией?
10. Как можно классифицировать информационные системы по степени структуризации решаемых задач?
11. Как можно классифицировать информационные системы по способу создания их создания и внедрения?
12. Что такое жизненный цикл информационной системы?
13. Какие можно выделить этапы жизненного цикла информационной системы, существенные с точки зрения характера и специфики решаемых задач?
14. Что является основным результатом выполнения этапа системного анализа?
15. В чем состоит цель этапа системного проектирования?

Литература по теме:

1. Теория систем и системный анализ в управлении организациями: справочник/ ред. В. Н. Волкова и А. А. Емельянов. – М.: Финансы и статистика, 2009. – 848с.

Тема 2. Информационная система как сложная система

Цели изучения темы:

- понять методологические основы теории информационных процессов и систем.

Задачи изучения темы:

- овладеть основными понятиями и фундаментальными принципами теории систем;
- познакомиться с отличительными особенностями информационных систем с точки зрения общей теории систем и системного анализа.

Успешно изучив тему, Вы: получите представление о:

- сущности методологии системного анализа, применительно к анализу информационных процессов и систем;
- подходе к решению задач синтеза информационных процессов и систем на основе агрегирования;
- применении концепций иерархических многоуровневых систем к исследованию и созданию информационных систем;

будете знать:

- чем характеризуются сложные системы, к которым относятся информационные системы;
- как можно математически описывать систему;
- что понимается под открытой информационной системой и на каких принципах может быть основано их создание.

Вопросы темы:

1. Основные понятия теории систем.
2. Сложная система.
3. Модели систем.

4. Методология системного анализа.
5. Агрегатное описание систем.
6. Иерархичность информационных систем.

Вопрос 1. Основные понятия теории систем.

Теория информационных процессов и систем занимается задачами, связанными с исследованиями и созданием информационных систем. Основу этой теории составляет **общая теория систем**, использующая концепцию «системы» как универсального представления некоторой сущности вне зависимости от конкретного воплощения и специфических свойств. Такое представление позволяет применять совокупность принципов и методов теории систем для изучения объектов и явлений любой природы.

Термин **система** не имеет общепринятого определения. Основоположник теории систем Л. Бераланфи определял систему как *совокупность элементов, находящихся в определенных отношениях друг с другом и со средой*.

Применение системного подхода к задачам построения и применения информационных систем должно учитывать тот факт, что эти системы создаются и используются для достижения четко определенных целей и задач. Под **целью** понимается ситуация или область ситуаций, которая должна быть достигнута в течение некоторого периода времени функционирования системы. Цель может задаваться требованиями к показателям результативности, ресурсоемкости, оперативности функционирования системы либо к траектории достижения заданного результата. Как будет видно из дальнейшего, цель для системы во многих случаях определяется *старшей* системой, для которой рассматриваемая система является *подсистемой* или *элементом*.

Наличие или отсутствие у системы цели функционирования лежит в основе их разделения на искусственные и естественные системы. **Естественные** (природные) **системы** не имеют четко выраженной цели своего функционирования (во всяком случае, достоверно известной их исследователям). **Искусственные системы** представляют собой продукт целенаправленной деятельности человека и используются для решения определенных задач, представляющих некоторую полезность для своих создателей. Как следует из этой классификации, все информационные системы относятся к искусственным системам.

Рассмотрение информационных систем как систем **целенаправленных** дает возможность применять специальные методы, в частности, методику структуризации целей и функций, которая рассматривается далее. С учетом этого обстоятельства будем опираться в дальнейшем на определение системы, предложенное В.Н. Волковой, где система S определяется как совокупность нескольких укрупненных компонентов, необходимых для ее существования и функционирования:

$$S \underset{\text{def}}{=} \langle \{Gls\}, \{Str\}, \{Tec\}, \{Cnd\} \rangle,$$

где

$\{Gls\}$ – совокупность, или структура целей;

$\{Str\}$ – совокупность структур, реализующих цели;

$\{Tec\}$ – совокупность технологий (методов, алгоритмов и т.п.), реализующих систему;

$\{Cnd\}$ – совокупность условий существования системы, т.е., факторов, влияющих на создание и функционирование системы.

Под **структурой** в этом определении понимается набор существенных связей между элементами системы, которые определяют, с одной стороны, как взаимодействуют ее подсистемы и элементы, так и, с другой стороны, свойства системы в целом.

Приведенное определение системы позволяет проводить анализ информационных систем, отталкиваясь от целей, и рассматривать множество применяемых моделей в качестве их неотъемлемой составляющей.

Вопрос 2. Сложная система.

Одной из основных классификаций систем является их разделение на *простые* и *сложные*. Общепризнанной границы между этими классами нет. Однако в большинстве случаев отнесение системы к **сложной** делается тогда, когда система характеризуется тремя основными признаками: *эмерджентностью*, *робастностью* и *неоднородностью связей* между ее элементами.

1) Наиболее существенным свойством сложных систем является свойство **интегративности**, называемое также **целостностью**, или **эмерджентностью**. Оно означает, что система обладает некоторым качеством, которого нет ни у одной из ее составляющих.

- Например, отдельный компьютер или локальная сеть корпоративной информационной системы сами по себе не обеспечивают поддержку решения всех задач предприятия, но объединение их в единое целое такую возможность уже предоставляет. С другой стороны, утрата локальной сетью своей работоспособности как одной из подсистем корпоративной информационной системы приведет к утрате последней своей функциональной полезности.

2) Под свойством **робастности** сложных систем понимается способность системы сохранять свою полную или частичную работоспособность при отказе отдельных ее элементов или подсистем. Во многих искусственных системах эта возможность обеспечивается включением в систему функциональной или информационной избыточности, позволяющей уменьшить степень деградации выполняемых функций в зависимости от влияния возмущений.

Примером использования такого подхода для построения информационных систем может служить технология RAID (*Redundant Array of Inexpensive Disks*, или *Redundant Array of Independent Disks*). В этой технологии надежность (робастность) системы достигается за счет размещения массивов основных данных и их дубликатов на нескольких относительно недорогих (*Inexpensive*), физически независимых (*Independent*), но логически связанных между собой дисках.

Простая система может находиться не более, чем в двух состояниях: полной работоспособности (исправном) и полного отказа (неисправном).

3) Характерным свойством сложных систем помимо значительного количества элементов является присутствие в них **многочисленных и неоднородных связей** между своими элементами. Основными типами считаются следующие виды связей:

- структурные (в том числе иерархические);
- функциональные;
- каузальные (причинно-следственные, отношения истинности);
- информационные;
- пространственно-временные.

Простым примером разнородных связей могут служить компоненты программного обеспечения, с помощью которого отдельные задачи (функциональные связи) могут реализоваться несколькими модулями (структурные, конструктивные связи), использующими определенные базы данных (информационные связи).

Вопрос 3. Модели систем.

Одной из наиболее важных задач начального этапа системного анализа является задача **определения границ**, или **изоляции** системы от окружающей среды. Под **окружающей средой**, или **окружением** понимается множество систем за пределами рассматриваемой системы, которые оказывают влияние на эту систему и сами подвержены воздействию со стороны системы. Это множество может быть образовано как системами естественного, так и искусственного происхождения.

В кибернетике и теории систем решение этой задачи представляется в виде модели *черного ящика* (рис. 4):



Рис. 4. «Черный ящик»

Причиной появления названия *черный ящик* является то, что в силу сложности изучения внутренних структуры и механизма поведения системы исследователь ограничивается изучением входных воздействий на систему и выходных результатов ее функционирования. Как будет видно из дальнейшего, модель *черного ящика* используется в целом ряде задач создания и применения информационных систем.

Присутствующие в модели черного ящика связи системы со средой имеют направленный характер: связи, по которым среда влияет на систему, называют **входами** системы, связи, по которым система оказывает влияние на среду, называют **выходами** системы. Например, в системе автоматизированного расчета заработной платы входы могут представлять собой сведения из табеля учета и штатного расписания предприятия, выходом – платежная ведомость по конкретному временному периоду.

Показанные на рисунке рис. 4 входы представляют собой множество **управляемых** входов, помимо них на систему оказывают влияние также **неуправляемые** входы, представляющие собой возмущающие воздействия со стороны окружающей среды. Эти воздействия могут вызываться как явлениями **природного**, так и **искусственного** происхождения. Например, причиной помех в радиоканале передачи данных могут служить как грозовые разряды, так и работа расположенных поблизости радиостанций. Воздействия искусственного происхождения могут носить как просто нежелательный (спам электронной почты, всплывающие окна), так и целенаправленный, злонамеренный (вирусные атаки, фишинг) характер.

Каждая система характеризуется совокупностью своих **свойств**, выделяющих данную систему из множества других. Свойства задаются набором значений, называемых **показателями**. Показатели подразделяются на количественные и качественные. К **количественным** относятся показатели, значение которых может быть измерено с помощью количественной шкалы (примером могут служить масса и размеры корпуса сотового телефона); остальные показатели относятся к **качественным** (например, цвет корпуса сотового телефона).

Некоторые из показателей системы достаточно полно и однозначно характеризуют систему. Значения этих показателей в какой-то момент времени позволяют получить как бы мгновенную фотографию, «срез» системы. Совокупность этих значений называется **состоянием** системы. Если считать, что число показателей (переменных) состояния равно k , то состояние можно представить как точку в k -мерном фазовом пространстве. Совокупность всех возможных значений состояний $\{Z\}$ называется **пространством состояний**. Текущее состояние системы Z определяет значение ее выходных параметров.

Если система обладает способностью переходить из одного состояния в другое, то об этих переходах говорят как о **поведении** системы. Если обозначить через $z(t_i)$ состояние системы в момент времени t_i , то некоторую совокупность состояний системы $z(t_1), z(t_2), \dots, z(t_r)$, упорядоченных по возрастанию t_i , называется **процессом функционирования** системы, или просто **процессом**. Процесс функционирования можно представить как упорядоченное множество точек k -мерного фазового пространства. Каждой реализации процесса будет соответствовать своя фазовая траектория.

Под **эффективностью процесса** понимается степень его приспособленности к достижению цели функционирования.

Системы, в которых со временем происходят некоторые изменения, называются **динамическими** системами. Системы, где изменения со временем не происходят, носят название **статических**.

Если обозначить множество управляемых входов через $\{x\}$ множество неуправляемых входов (отражающих влияние среды на систему) через $\{n\}$, множество возможных состояний системы через $\{z\}$ множество выходов системы через $\{y\}$ (рис. 5).

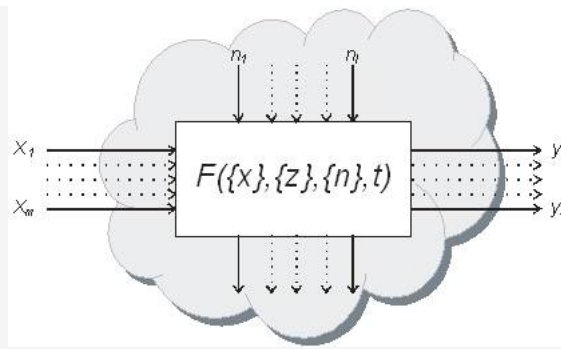


Рис. 5. Математическая запись модели черного ящика

то модель черного ящика можно записать математически в виде

$$\{y\}=F(\{x\}, \{z\}, \{n\}, t),$$

где

t - означает момент времени функционирования.

Оператор (функция) F описывает **закон функционирования** системы. Он осуществляет преобразование независимых переменных в зависимые и отражает поведение системы во времени. Такое представление основано на **кибернетическом подходе** к анализу систем.

Под **качеством** системы понимают совокупность существенных ее свойств, обуславливающих ее пригодность для использования по назначению. Оценивать качество можно и по одному обобщенному свойству, которое определяется на основе обобщенного показателя качества системы (об этом речь пойдет далее).

В зависимости от степени взаимовлияния системы и среды системы разделяют на открытые и замкнутые. В **открытой** системе степень взаимовлияния велика, в силу чего невозможно однозначно предсказать, каким будет выход системы при известном входе. Понятие открытой системы конкретизируется в каждой предметной области. В частности, **открытыми информационными системами** называются такие, которые обладают свойствами:

- **переносимости** или **мобильности**: программное обеспечение может переноситься на другие аппаратные платформы и в операционные среды;
- **стандартности**: программное обеспечение соответствует общепризнанному стандарту независимо от конкретного разработчика;
- **масштабируемости**: в систему могут добавляться программные и технические средства, отсутствующие в первоначальном варианте;
- **интероперабельности**: системой обеспечивается доступ пользователя, как в локальном, так и в удаленном режимах;
- **совместимости**: система способна взаимодействовать с другими системами с помощью информационных и программных интерфейсов.

Построение открытых информационных систем может проводиться на основе **технологии открытых систем**, которая позволяет формировать необходимую среду. Для поддержки технологии открытых систем рабочей группой POSIX Института инженеров по электротехнике и электронике (IEEE) разработана эталонная **модель среды открытых систем** (*Open System Environment Reference Model - OSE Reference Model, OSE/RM*). В этой модели (рис. 6) присутствуют объекты трех типов (прикладного программного обеспечения, платформы программного обеспечения и объектов внешней среды) и интерфейсы (с объектами прикладных программ и с объектами внешней среды).



Рис. 6. Модель OSE/RM

Нужное семейство стандартов в модели OSE определяется с помощью **профиля**, специфически задаваемого для определенного класса системы, - совокупности согласованных стандартов для спецификации служб, интерфейсов, протоколов, форматов данных. Следование стандартам принятого профиля в процессе создания системы обеспечит соблюдение перечисленных выше требований, предъявляемых к открытым системам.

В **замкнутой, или закрытой** системе взаимовлияние системы и среды пренебрежимо мало, так что их можно считать изолированными друг от друга и есть возможность точно предсказать реакцию системы на управляемые входные воздействия. Примером замкнутой системы может служить локальная сеть для обработки конфиденциальной информации.

Всякий объект может быть представлен как система, которую в целях более детального рассмотрения ее свойств можно представить состоящей из некоторого числа взаимодействующих между собой **подсистем**. Подсистемы, в свою очередь, можно также представить в виде множества подсистем. Этот процесс **декомпозиции** можно продолжать до момента, пока не будет достигнут уровень представления в виде отдельных **элементов**.

Уровень разбиения на элементы и сами элементы не являются заранее установленным для всех систем и неизменным. Он определяется требованиями конкретной задачи: так, для некоторых задач анализа информационных систем и процессов, происходящих на некотором предприятии, элементами будут отдельные подразделения или службы этого предприятия, для других – отдельные исполнители или функциональные подсистемы, для третьих – отдельные устройства или программные модули.

Вместе с тем, любой объект можно трактовать как одну из подсистем другой **старшей системы** (или **суперсистемы**, или **сверхсистемы**, или **надсистемы**). Эта подсистема взаимодействует с другими подсистемами. Продолжая процесс дальше, приходим к тому, что старшая система также представляет собой подсистему более старшей системы и так далее.

Вопрос 4. Методология системного анализа.

На начальных этапах жизненного цикла сведений, необходимых для решения задач анализа и синтеза информационной системы бывает, как правило, недостаточно. Таким образом, одну общую формализованную модель, которая позволяла бы получить все нужные для проектирования результаты (как качественные, так и количественные) построить практически невозможно. Одним из наиболее эффективных подходов к решению в этом случае является системный анализ. Это название впервые появилось в 1948 году применительно к работам, проводимым американской корпорацией RAND.

Под системным анализом можно понимать логически связанную совокупность теоретических и эмпирических положений из области математики, естественных наук и опыта разработки сложных систем, обеспечивающую повышение обоснованности решения конкретной проблемы. Значительное развитие этот подход получил с началом бурного развития средств вычислительной техники, по причине чего академик Н.Моисеев определял системный анализ как *совокупность методов, основанных на использовании ЭВМ и ориентированных на исследование сложных систем – технических, экономических, экологических и т.д.*

Хотя однозначной трактовки термина не существует, можно выделить ряд характерных особенностей, присущих этому подходу. К этим особенностям относятся следующие.

- Постановка задачи отличается высокой степенью неопределенности и не допускает однозначной формализации.
- В системном анализе сочетаются количественные (формальные) методы с методами качественного анализа.
- В системном анализе используются математический аппарат и аппарат теории систем.
- Системный анализ концентрирует свое внимание на процессах целеобразования, используя для этого методику структуризации целей.
- Системный анализ использует в качестве одного из основных методов уменьшения неопределенности (расчленение системы на подсистемы).

Таким образом, системный анализ имеет, **универсальный и междисциплинарный** характер.

Системный анализ, проводимый в процессе создания информационной системы, включает задачи *декомпозиции, анализа и синтеза*.

Декомпозиция (структуризация) рассматривается как составная часть анализа и означает представление системы в виде подсистем, состоящих из более мелких элементов.

Анализ состоит в нахождении различного рода свойств системы или окружающей. Целью анализа может быть определение закона преобразования информации, задающего поведение системы. В этом случае необходимо провести *агрегирование (композицию)* системы в представление ее одним элементом.

Синтез преследует цель построения системы в соответствии с выявленными на этапе анализа целями, законами функционирования и требованиями к качеству системы.

Ранее рассмотренные модели (в частности, модель черного ящика) являлись некоторым **абстрактным** представлением реального объекта. Такие модели носят название **математических**. Математическая модель в отличие от модели **физической** не имеет материального воплощения, и представляет собой описание объекта на некотором неформализованном, полуформализованном или формализованном языке (таким языком может быть, в частности, язык математический). Точность математических моделей обычно уступает точности физических, но затраты на их создание значительно меньше. В основу системного анализа положены математические модели.

Вопрос 5. Агрегатное описание систем.

Для достижения конечной цели этапа синтеза необходимо осуществить решение задачи обратной задаче анализа, основным инструментом решения которой является декомпозиция. Способом решения задачи синтеза является **агрегирование**, сущность которого заключается в построении единой модели системы на основе моделей, разработанных в результате анализа. Построенная в результате агрегирования модель носит название **агрегата**.

Аналогично тому, как в процессе анализа для изучения различных свойств системы могут применяться различные модели, так и в процессе агрегирования каждая постановка задачи может потребовать построения своей модели. Например, при построении модели надежности функционирования корпоративной информационной системы могут оставаться без рассмотрения стоимостные показатели системы. Как следствие, для разработки моделей в типичном случае требуется применять различные средства (языки) моделирования.

Принято различать несколько видов агрегирования.

В тех случаях, когда целью агрегирования является синтез *структуры* системы, то говорят об **агрегатах-структурах**. При этом проектирование системы может потребовать составления описаний различных аспектов структуры системы. Например,

структура программных средств информационной системы может характеризоваться как своим функциональным составом, так и составом модулей, которые представляют собой образующие систему конструктивные единицы.

Если требуется построить описание *количественных* признаков синтезируемой системы, то для этих целей используются **агрегаты-операторы**, представляющие собой зависимости выходных показателей от входных в виде функционала. Под агрегатом-оператором понимается объект, определяемый множествами T, X, U, Y, Z , оператором переходов H и оператором выходов G .

Здесь множество T есть множество моментов времени, X – множество входных сигналов, U – множество сигналов управления, Y – множество выходных сигналов, Z – множество состояний системы. Все сигналы рассматриваются как функции времени, так что операторы H и G реализуют функции соответственно $z(t)$ и $y(t)$.

Оператор переходов H однозначно определяет состояние системы $z(t+\Delta t)$ в момент времени $t+\Delta t$ по известным состоянию системы $z(t)$, входному сигналу $x(t)$ и сигналу управления $u(t)$ в момент времени t :

$$z(t+\Delta t)=H\{t, x(t), u(t), z(t)\}$$

Оператор выходов G однозначно определяет значения выходных показателей системы $z(t+\Delta t)$ в момент $t+\Delta t$ по известным состоянию системы $z(t)$, входному сигналу $x(t)$ и сигналу управления $u(t)$ в момент времени t :

$$y(t)=G\{t, x(t), u(t), z(t)\}$$

В общем случае оператор выходов G является случайным оператором и задается функцией распределения, а состояние системы в каждый момент времени t может быть задано значениями совокупности случайных величин $z(t)=(z_1(t), \dots, z_k(t))$. Процесс, значения переменных состояния которого в любой момент времени являются случайными величинами, носит название **случайного процесса**. Существуют важные частные случаи случайных процессов (кусочно-непрерывные, кусочно-линейные, винеровские и др.), которые являются объектами изучения специальных теорий.

Так как выбор фазовых переменных может производиться различными способами, то для одной и той же системы может быть построено множество агрегатов-операторов.

Формализованное представление системы в виде агрегата-оператора позволяет получить значения различных числовых характеристик. Однако практическому построению и применению такой модели часто препятствуют трудности, вызываемые необходимостью включать в рассмотрение большое число параметров для обеспечения приемлемой точности модели.

Вопрос 6. Иерархичность информационных систем.

Одной из основных закономерностей, присущих сложным системам, является **иерархия** (ερός – священный, σῆς – власть), или иерархическая упорядоченность. Под иерархией в теории систем понимается любой согласованный по подчиненности порядок объектов. В частности, управление большинством социальных организаций, предприятий и государственных структур строится по иерархическому принципу.

Впервые иерархия была исследована Л. Берталанфи, который показал ее связь с закономерностями самоорганизации и развития открытых систем. В теории систем принято выделять три основных вида иерархии – *стратифицированные* системы, *многослойные* системы, *многосежелонные* системы.

Страты.

В случае создания и сопровождения информационных систем сложность задачи заключается в необходимости соблюсти баланс между **целостностью представления** системы, которое должно быть получено в начальный период жизненного цикла, и ее **детальным описанием** на последующих этапах (проектирование, разработка, внедрение). Разработать единую методику, которая обеспечила бы решение всех задач на всех этапах жизненного цикла практически невозможно.

Вместе с тем, использование системного подхода к созданию и сопровождению информационных систем позволяет на разных этапах жизненного цикла наполнять термин *система* разным смыслом. Это дает возможность аналитикам и проектировщикам в зависимости от характера стоящих перед ними задач рассматривать систему в разных формах ее представления и описания, используя различные модели и методы.

Поэтому на практике применяется так называемое **стратифицированное** представление процесса проектирования, а сами уровни носят названия **страт**. Практика применения системного анализа показывает, что построение методик исследования и проектирования систем применительно к отдельным стратам является значительно более эффективным.

Примером стратификации может служить модель электронной вычислительной машины. Ее функционирование обычно описывается не менее чем на двух стратах (рис. 7). На первой страте система описывается на языке физических законов, управляющих работой и взаимодействием ее составных частей, на второй страте используются абстрактные, не физические понятия – программа, оператор, информационные потоки. На страте физических законов объектом интереса является функционирование различных электронных компонентов, на страте обработки информации изучаются проблемы организации вычислений, разработки программ и т.д., изолированно от лежащих в их основе физических законов.

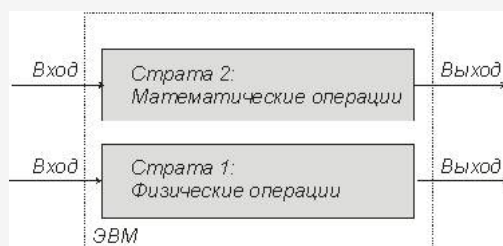


Рис. 7. Стратифицированное представление ЭВМ

Используя этот подход можно рассматривать такую проблему как анализ текста, выделяя страты букв, слов, предложений, абзацев, текста в целом. Еще одним примером стратификации может служить модель взаимодействия открытых систем OSI, в котором выделяются семь уровней протокола (рис. 8).

Уровни протокола	Страты
Прикладной	7 Доступ приложений
Представлений	6 Преобразования данных
Сеансовый	5 Межмашинный обмен
Транспортный	4 Доставка данных
Сетевой	3 Маршрутизация
Канальный	2 Кадры
Физический	1 Сигналы

Рис. 8. Сетевая модель OSI

Модели позволяют преобразовывать **вербальное** (словесное) описание проблемной ситуации в **формализованное** описание, позволяющее применять математические методы, как для получения количественных характеристик, так и для их интерпретации аналитиком. Степень формализации определяется, в первую очередь, уровнем рассмотрения. Описания и проблемы на верхних стратах менее структурированы. Они содержат больше неопределенностей и более трудны для количественной формализации. Используемые здесь методы и модели часто носят чисто описательный характер или используют лишь частично формализованный язык описания. Проблема принятия решений на верхних уровнях может рассматриваться как более сложная. Для решения задачи на верхнем уровне могут использоваться приближенные модели и методы, что необходимо учитывать при интерпретации результатов.

Переход к задачам нижележащего уровня осуществляется таким образом, чтобы учитывались результаты решения задач вышележащих уровней. Этот учет реализуется в виде определенной совокупности ограничений, задающих степень упрощения для моделей, создаваемых и применяемых на данном уровне.

Таким образом, спектр моделей, применяемых в системном анализе и проектировании, достаточно широк. Существует много классификаций, которые могут оказаться полезными. Вместе с тем, одна и та же ситуация может описываться моделями разных классов в зависимости от требований задачи и объема имеющихся у исследователя знаний.

Тем не менее, для целей изучения основных моделей, используемых в теории информационных систем, можно достаточно условно рассматривать различные подходы и методы решения в зависимости от уровня рассмотрения. Хотя установить однозначное соответствие между моделями и задачами нельзя, можно условно выделить некоторые основные виды моделей, применяемых для решения задач анализа информационных процессов и синтеза структуры и параметров процессов и систем на различных этапах их жизненного цикла.

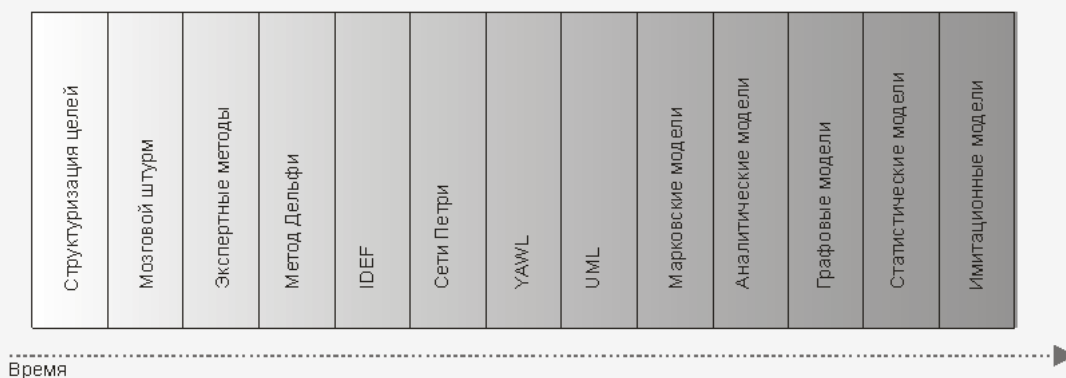


Рис. 9. Модели и этапы жизненного цикла информационных систем

Модели информационных процессов и систем, создаваемые на начальных этапах, рассматривают в первую очередь внешние характеристики системы без учета реализационных особенностей. Такие модели называют **каноническими** в отличие от **рабочих** моделей, которые создаются на последующих этапах. Концепции, положенные в основу моделей, будут рассматриваться в дальнейших разделах курса.

Слои.

Решения, принимаемые в сложных системах, характеризуются различной степенью неопределенности ситуации. В целях достижения большего качества результата и эффективности применяемой для его получения процедуры последняя реализуется в виде совокупности последовательных шагов, на каждом из которых решается своя задача. Выделение отдельных задач производится таким образом, что каждой из них соответствует свой уровень иерархии, называемый **слоем**. Решение задачи вышележащего слоя задает ограничения для модели, применяемой для решения на нижележащем слое. Это позволяет снижать неопределенность задачи нижележащего слоя.

Многослойную иерархию иллюстрирует рис. 10.

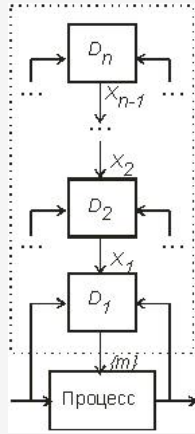


Рис. 10. Многослойная иерархия принятия решения

Каждый слой на рис. 10 представлен блоком принятия решения D_j , который вырабатывает ограничения X_{j-1} для блока D_{j-1} . Пример многослойного управления процессом показан на рис. 11.

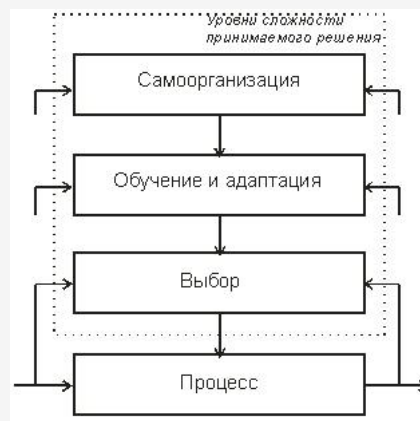


Рис. 11. Пример многослойного управления процессом

На *верхнем слое*, слое самоорганизации, выбираются структура, функции и стратегия, используемые на нижележащих слоях с тем, чтобы обеспечить наилучшее приближение к цели, которая формулируется, как правило, на вербальном уровне.

Задачей слоя *обучения*, или *адаптации* является сужение неопределенностей для нижележащего слоя, что позволяет упростить используемую на этом слое модель. Решение этой задачи достигается путем проведения наблюдений за процессами и использования дополнительных источников информации.

Нижний слой, слой *выбора*, определяет способ (алгоритм) воздействия на управляемый процесс.

Эшелоны.

Для представления сложных систем с организационной точки зрения в теории иерархических многоуровневых систем вводится понятие **многоэшелонной** иерархической структуры. Такая структура представляется в виде относительно независимых подсистем, каждая из которых имеет право принятия решения, а их иерархическое расположение определяется влиянием, которое оказывает на них вышестоящие подсистемы (рис. 12).

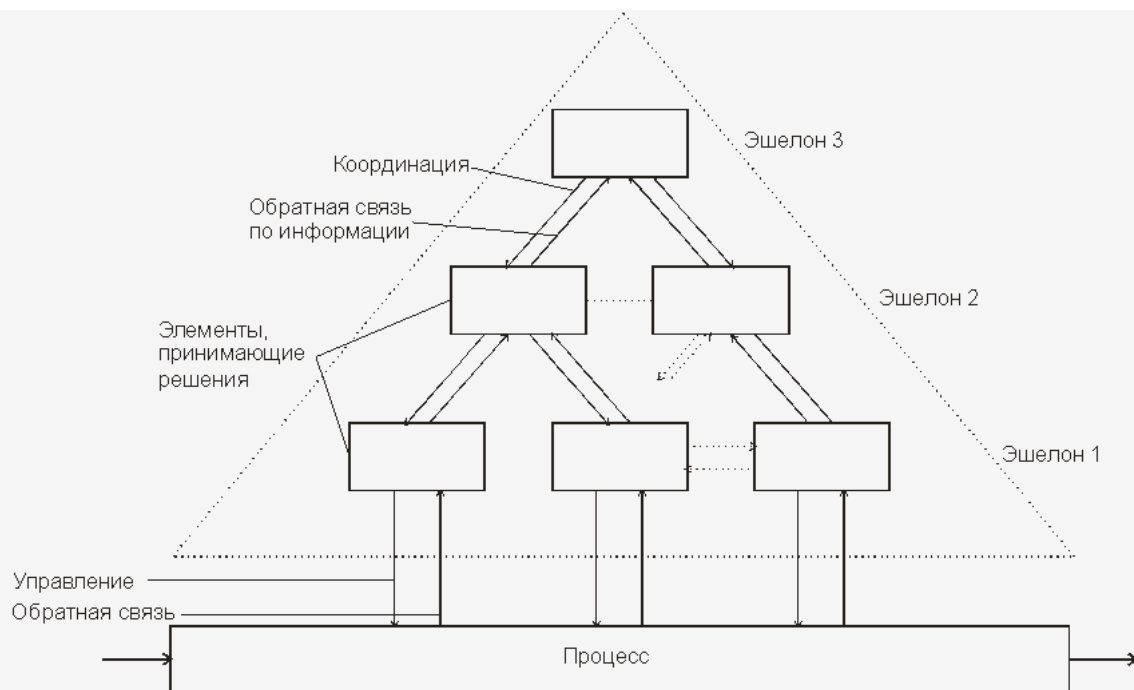


Рис. 12. Многоэшелонная структура

Подсистемы всех уровней в многоэшелонных структурах наделяются определенной свободой, как выбора решения, так и выбора целей. Доказано, что этот принцип повышает эффективность функционирования всей системы.

Разрешение возможных конфликтов, обусловленных предоставлением свободы выбора, разрешается вмешательством вышестоящего эшелона. Для отражения степени вмешательства вышестоящего эшелона используются термины *управление* и *координация*. Система управления принятием решений может использовать различные формы и способы вмешательства, поэтому многоэшелонные системы называются также **организационной иерархией**.

Выводы:

1. В силу своих существенных свойств и особенностей информационная система относится к сложным системам. Это обуславливает необходимость проводить рассмотрение проблем анализа и синтеза информационных процессов и систем, опираясь на основные понятия и принципы общей теории систем с учетом специфических особенностей информационных систем.
2. Главным инструментом описания информационных систем с целью решения возникающих задач является математическое моделирование. Ввиду многообразия постановок задач и сложности объекта исследования создать единую модель практически невозможно, поэтому необходимо разрабатывать набор моделей. На начальном этапе исследований целесообразно применять модель «черного ящика», не требующую описывать внутреннюю структуру системы и взаимодействие ее элементов, о которых на этом этапе жизненного цикла информации еще недостаточно.
3. Основным подходом к исследованию информационных процессов и систем является методология системного анализа, которая носит универсальный характер и позволяет применять ее для исследования различных процессов в различных предметных областях. Одним из ключевых методов системного анализа является метод декомпозиции, реализующий систематическим образом принцип «разделяй и властвуй» по отношению к объекту исследования.
4. Основным подходом к решению задач синтеза является агрегирование, преследующее цель построения единой модели на основе построенной на этапе анализа совокупности моделей. В зависимости от конкретного случая могут применяться различные типы агрегатов.
5. В силу присущему информационным процессам и системам свойству иерархической упорядоченности построение и анализ моделей целесообразно проводить, представляя исследуемую систему в различных видах иерархии. Создание моделей системы с использованием концепции страт, слоев и эшелонов позволяет добиться достаточной точности описания, ограничить сложность создаваемых моделей и рационально организовать процесс исследования.

Вопросы для самопроверки:

1. Что называется системой?
2. Что понимается под целью функционирования системы?
3. Какие системы относят к естественным?
4. Какие системы относят к искусственным?
5. Какие системы называют целенаправленными?
6. Что понимается под структурой системы?
7. Какая система называется сложной?
8. Что означает свойство интегративности?
9. Что означает свойство робастности?
10. Что называется окружающей средой?
11. Что собой представляет модель *черного ящика*?
12. В каких случаях применяется модель *черного ящика*?
13. Что означает и для чего используется понятие *свойства* системы?
14. Что понимается под термином *показатель*?

15. Какие типы показателей существуют?
16. Что означает термин *поведение системы*?
17. Что называется процессом функционирования системы?
18. Что понимают под эффективностью процесса функционирования?
19. Что понимается под качеством системы?
20. В чем состоит отличие между открытой и закрытой системами?
21. Какими свойствами обладают открытые информационные системы?
22. Что такое подсистема?
23. Что понимается под термином *декомпозиция*?
24. Как можно определить сущность методологии системного анализа?
25. Что означают термины *агрегат* и *агрегирование*?
26. Какие типы агрегатов вы знаете?
27. Как записывается математическая модель агрегата-оператора?
28. Дайте определение понятия *иерархия*.
29. Что представляет стратифицированное представление процесса и зачем оно необходимо?
30. Что называют слоями, что отображается с помощью слоев?
31. Что можно представить с помощью многоэшелонной иерархической структуры?

Литература по теме:

Основная литература:

1. Анфилов В. С. Системный анализ в управлении: учебное пособие/ Анфилов В. С. , Емельянов В. С. , Кукушкин А. А. – М.: Финансы и статистика, 2008. - 368с.
2. Теория систем и системный анализ в управлении организациями: справочник/ ред. В. Н. Волкова и А. А. Емельянов. – М.: Финансы и статистика, 2009. - 848с.

Дополнительная литература:

1. Месарович М. Теория иерархических многоуровневых систем/ Месарович М., Мако Д., Такахара И. М. - М.: Мир, 1973 – 344 с.
2. Холл А. Опыт методологии для системотехники. - М.: Советское радио, 1975. – 448 с.

Тема 3. Качественные методы системного анализа в теории информационных процессов и систем

Цели изучения темы:

- познакомиться с задачами начального этапа жизненного цикла информационных систем и методами их решения на основе качественного оценивания.

Задачи изучения темы:

- изучить методику построения дерева целей;
- изучить подходы к качественному оцениванию альтернатив.

Успешно изучив тему. Вы:

получите представление о:

- специфике начального этапа проектирования систем;
- методах, применяемых для качественного оценивания альтернатив;
- достоинствах и недостатках каждого метода;

будете знать:

- как выявлять и представлять цели создаваемой информационной системы на основе метода структуризации;
- как получить необходимые для исследования идеи с помощью метода мозгового штурма;
- как организуется работа экспертной группы, как собираются и обрабатываются данные экспертизы.

Вопросы темы:

1. Структуризация целей.
2. Мозговой штурм.
3. Экспертное оценивание.
4. Метод Дельфи.

Вопрос 1. Структуризация целей.

Любая деятельность подчинена определенной цели. От того, насколько точно и полно сформулирована цель функционирования моделируемой организации, настолько же точным и полным будет результат создания и внедрения информационной системы. Цель может быть выражена как в **количественном** (численное значение какого-либо показателя с указанием его размерности), так и в **качественном** (описание на естественном языке) виде.

Например, формулировки «доля производственного брака в конечной продукции не должна превышать 1%», «ответ на обращение гражданина должен быть отправлен по электронной почте в течение 48 часов с момента получения» относятся к целям первой категории, а формулировка «предлагаемый предприятием пакет услуг должен полностью соответствовать потребностям его клиентов» может служить примером второй категории целей.

Как уже отмечалось, описания или модели информационных систем часто могут быть получены лишь на основе стратификации. Кроме того, глобальная цель, для осуществления которой создается система, может быть конкретизирована

путем установления иерархии необходимых составляющих ее целей (подцелей). Эти подцели, в свою очередь, могут быть представлены своими подцелями и т.д. Результатом такого процесса, который носит название **структуризации целей**, является **дерево целей**, представляющее собой иерархический граф (рис. 13).

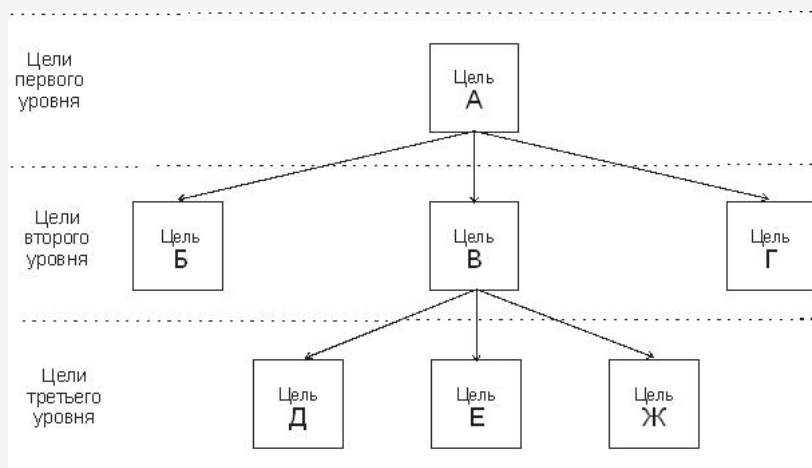


Рис. 13. Дерево целей

Цель А является главной (глобальной) целью. Она декомпозируется на цели (подцели) Б, В, Г, которые теперь образуют нижеследующий (второй) уровень иерархии целей.

Цель второго уровня В декомпозирована на подцели и включает цели (подцели) Д, Е, Ж, образующие третий уровень иерархии.

Как видно из рисунка, все цели показанного на нем дерева связаны между собой отношением *строгого порядка*, при котором каждая цель (вершина дерева) подчинена только одной цели верхнего уровня. Однако на практике встречаются и случаи *слабой иерархии*, когда некоторые цели нижних уровней подчинены одновременно более чем одной цели верхнего уровня.

Каждая цель дерева целей может быть как количественной, так и качественной. Формулировки целей должны удовлетворять требованиям

- *однозначности*: формулировка не должна допускать неточное или искаженное понимание цели;
- *реалистичности*: цели должны выполняться в приемлемые сроки и при имеющихся ограничениях на ресурсы;
- *непротиворечивости*: цели не должны противоречить друг другу;
- *контролируемости*: должна иметься возможность применить формальную процедуру для установления факта достижения цели;
- *измеримости*: должна иметься возможность оценить степень прогресса.

Впервые методика дерева целей была реализована в американской корпорации РЭНД (RAND) и получила название ПАТТЕРН (PATTERN – Planning Assistance Through Technical Evaluation from Relevance Number), конечной целью которой были составление и реализация планов обеспечения глобального военного превосходства США. В своем первоначальном виде для формирования и оценки дерева целей в методике использовались сценарии и прогнозы с глубиной прогнозирования 10-15 лет, вводились классы критериев с коэффициентами их относительной важности, взаимной полезности, состояния и сроков разработки. Применение методики ПАТТЕРН в чистом виде сопряжено с решением довольно сложных задач и имеет смысл только для очень крупных структур.

В системном анализе предложены и другие методики структуризации целей (методики Ю. Черняка, Е. Голубкова, Обобщенная методика анализа целей и функций систем управления), имеющие *большую* практическую направленность.

Рис. 14 иллюстрирует возможный результат структуризации стратегической цели некоторой компании.

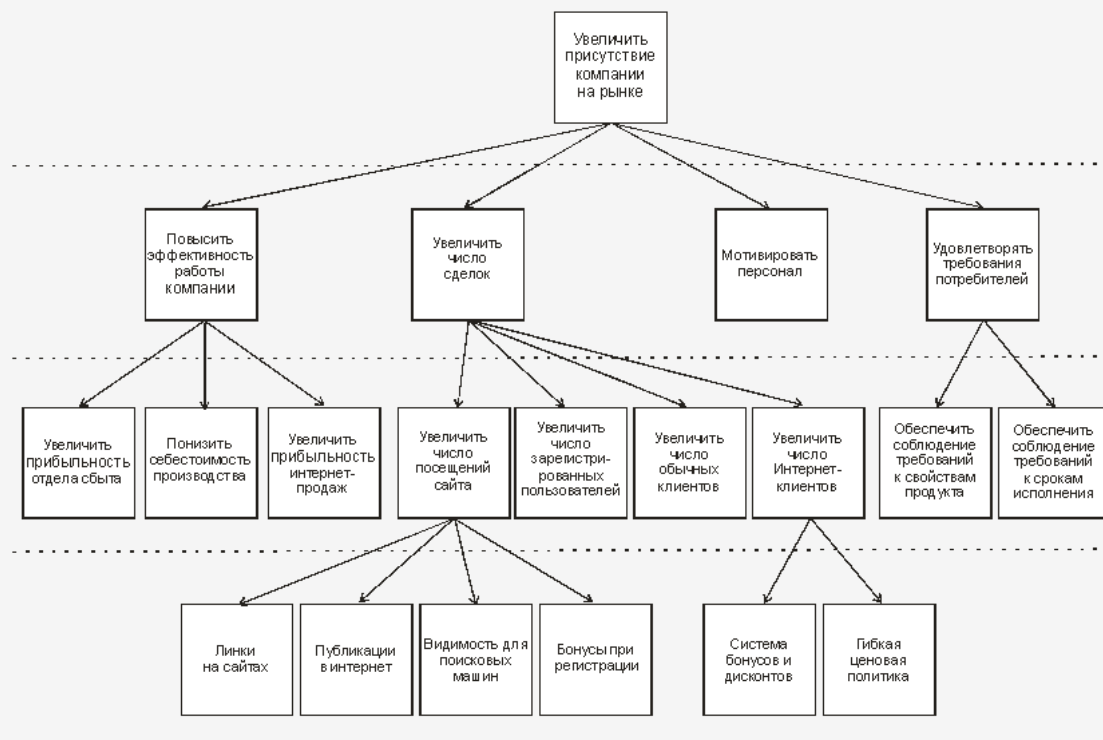


Рис. 14. Пример структуризации цели

Вопрос 2. Мозговой штурм.

Метод **мозгового штурма** (метод мозговой атаки, коллективная генерация идей) представляет собой один из эффективных методов создания новых идей. Он применяется, начиная с 50-х годов прошлого века, и основан на предположении о наличии в некоторой совокупности идей нескольких достаточно хороших. Сущность метода состоит в коллективном поиске различных путей решения поставленной задачи.

Типичными задачами для мозгового штурма являются задачи, формулировки которых могут быть представлены конструкцией «Как сделать нечто?», «Как достичь чего-то?». Такие задачи называют **синтетическими**. Применительно к задачам создания информационных систем к таким задачам могут относиться задачи создания деловых моделей, например:

- Как обеспечить пятипроцентный рост производства продукции при имеющихся ограничениях на ресурсы?
- Какие виды сервиса из ныне отсутствующих в компании следует включить в перечень обязательных?

Задачами, относящимися к созданию информационной системы, которые могут решаться с применением метода мозгового штурма, могут быть, например, такие:

- Для достижения каких целей нужна информационная система?
- Какие деловые процессы должны поддерживаться информационной системой?
- Каким основным требованиям должна отвечать информационная система?

Проведению мозгового штурма предшествует подготовительный этап (рис. 15, а).

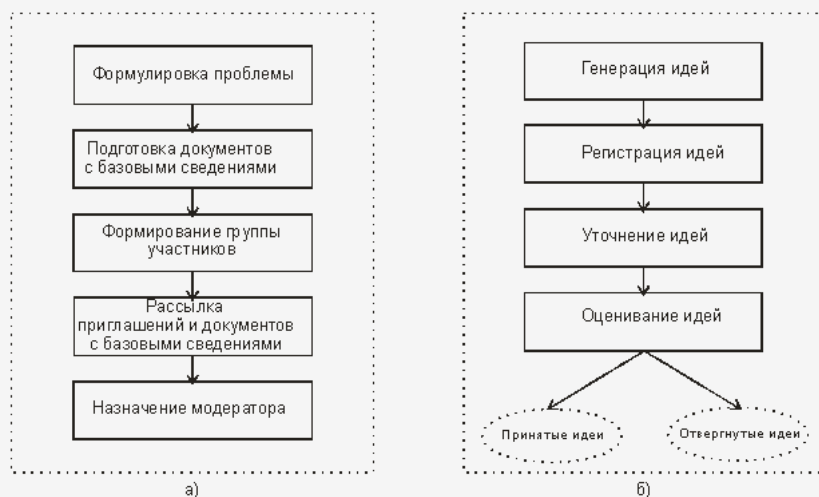


Рис. 15. Мозговой штурм: а) подготовка, б) проведение

На подготовительном этапе организаторы ставят задачу, подбирают необходимые материалы справочно-организационного характера и формируют состав участников, которых знакомят с предметом, местом и временем предстоящего обсуждения. Задача должна излагаться максимально точно и лаконично (типа вышеприведенных примеров). При подборе участников стараются включить в их состав как участников проекта, по проблеме которого ведется обсуждение, так и сторонних экспертов. Один из участников выполняет функции модератора (председателя) и секретаря, фиксирующего высказанные идеи.

Совещание обычно разбивается на два этапа: целью первого является выработка идей, второго – оценивание предложенных идей (рис. 15, б). При проведении первого этапа совещания стремятся обеспечить получение как можно большего количества идей, уровень качества идей во главу угла не ставится, обосновывать свои идеи их авторы не должны. Для достижения этой цели участники должны придерживаться некоторых правил:

- фиксируется любая идея независимо от производимого ей первого впечатления;
- приветствуется любая высказанная идея;
- не допускается никакая критика высказываемых идей;

На втором этапе все высказанные и записанные идеи подвергаются оцениванию, в результате чего одна часть идей отвергается, другая анализируется более тщательно. Обычно итогом обсуждения является одна, которая и предлагается от имени группы участников. На этом этапе также применяются определенные правила в виде критериев оценки предложенных идей.

В ряде случаев на первый и второй этапы приглашаются разные участники. Это, с одной стороны, обеспечивает непредвзятость и беспристрастность специалистов и, с другой стороны, с учетом разной склонности людей к решению этих разных по характеру задач рационально использует потенциал специалистов.

Практикой выработан определенный порядок (процедура), в соответствии с которой реализуются как первый, так и второй этапы совещания.

Существуют различные формы и разновидности проведения мозгового штурма. Так, в силу того обстоятельства, что эксперты и специалисты обычно заняты на своей основной работе и организовать одновременное присутствие на совещании всех участников бывает трудно, часто организуются **электронные** совещания. На них осуществляется дистанционное взаимодействие участников, для чего применяются как специальные программные системы (*системы поддержки электронных совещаний*), так и обычные средства типа интернет-браузеров и электронной почты.

Некоторым аналогом мозгового штурма являются разного рода совещательные органы типа директоратов, ученых советов, создаваемых на временной основе комиссий.

Вопрос 3. Экспертное оценивание.

Название метода происходит от латинского expertus (опытный). Метод основан на предположениях о том, что 1) суммарный объем знаний группы специалистов превосходит объем знаний одного специалиста, и 2) число вариантов решения задачи, которые способна порождать и оценивать группа специалистов превосходит число вариантов, которые способен порождать и оценивать один специалист. В процессе экспертизы выделяются несколько основных этапов (рис. 16).

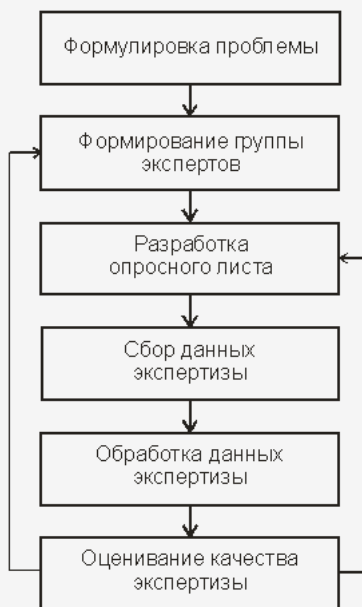


Рис. 16. Основные этапы процедуры экспертного оценивания

Совокупность опрашиваемых участников процесса называется **референтной**, или **экспертной** группой, а для оцениваемых объектов принято использовать термин **факторы**. Результатом проведения экспертизы является совокупность предпочтений, для представления которых используются различные методы. Одним из методов представления мнения экспертов является метод **ранжирования**: наиболее предпочтительному фактору присваивается ранг, равный единице, второму по предпочтительности – ранг, равный двум, и т.д. Для одинаковых по значимости факторов используются **связанные ранги**: например, если неразличимыми являются факторы Φ_3, Φ_4, Φ_5 , то каждому из них присваивается ранг $\frac{(3+4+5)}{3} = 4$. При таком способе сумма всех рангов для каждого эксперта должна быть одинаковой и равной $1+2+ \dots + n = \frac{n(n+1)}{2}$.

Поясним на примере сущность решаемых в процессе проведения экспертизы задач. Будем полагать, что необходимо оценить значимость для компании включения в состав выпускаемых ею продуктов пяти новых единиц, для чего привлекаются четыре эксперта.

Результатом опроса экспертов по данной проблеме является матрица $R = R_{ij}$, $i=1 \dots n$, $j=1 \dots m$, n -число факторов (продуктов), m -число экспертов:

$$R = \{R_{ij}\} = \begin{pmatrix} 1 & 1 & 3 & 1,5 \\ 3 & 2 & 1 & 1,5 \\ 2 & 4 & 3 & 3 \\ 5 & 3 & 3 & 5 \\ 4 & 5 & 5 & 4 \end{pmatrix}$$

Видим, что сумма каждого столбца равна $\sum_{i=1}^5 R_{ij} = \frac{5 \cdot 6}{2} = 15$. В третьем и четвертом столбцах присутствуют связанные ранги: третий эксперт считает равноценными первый, третий и четвертый продукты, поэтому значения связанных рангов равны $R = \frac{(2+3+4)}{3} = 3$; четвертый эксперт не различает первый и второй продукты, поэтому здесь $R = \frac{(1+2)}{2} = 1,5$.

Для обработки мнений экспертов строится матрица **преобразованных рангов** $R'_{ij} = n - R_{ij}$ (в данном примере $R'_{ij} = 5 - R_{ij}$):

$$R' = \{R'_{ij}\} = \begin{pmatrix} 4 & 4 & 2 & 3,5 \\ 2 & 3 & 4 & 3,5 \\ 3 & 1 & 2 & 2 \\ 0 & 2 & 2 & 0 \\ 1 & 0 & 0 & 1 \end{pmatrix}$$

и вслед за ней - матрица **нормированных весов** $X'_{ij} = \frac{R'_{ij}}{\sum_{i=1}^n R'_{ij}} = \frac{R'_{ij}}{\frac{n(n-1)}{2}}$ (в данном примере $\frac{n(n-1)}{2} = 10$):

$$X = \{X_{ij}\} = \begin{pmatrix} 0,4 & 0,4 & 0,2 & 0,35 \\ 0,2 & 0,3 & 0,4 & 0,35 \\ 0,3 & 0,1 & 0,2 & 0,2 \\ 0 & 0,2 & 0,2 & 0 \\ 0,1 & 0 & 0 & 0,1 \end{pmatrix}$$

Наиболее распространенный метод построения **группового мнения** состоит в нахождении **вектора-столбца весов** оцениваемых факторов w :

$$w = \begin{pmatrix} w_1 \\ \vdots \\ w_i \\ \vdots \\ w_n \end{pmatrix}$$

такого, что:

$$\Delta(w) = \sum_{i=1}^n \sum_{j=1}^m (w_i - x_{ij})^2 = \min_w \Delta(w)$$

Иначе говоря, веса факторов выбираются так, чтобы минимизировать сумму расстояний до весов, назначенных экспертами. Известно, что это выполняется тогда и только тогда, когда $w_i = \frac{1}{m} \sum_{j=1}^m x_{ij}$ ($i=1, \dots, n$), т. е., каждое w_i является средним арифметическим оценок экспертов. В данном случае:

$$w_1 = \frac{0,4 + 0,4 + 0,2 + 0,35}{4} = 0,344$$

$$w_2 = \frac{0,2 + 0,3 + 0,4 + 0,35}{4} = 0,31$$

$$w_3 = \frac{0,3 + 0,1 + 0,2 + 0,2}{4} = 0,20$$

$$w_4 = \frac{0 + 0,2 + 0,2 + 0}{4} = 0,10$$

$$w_5 = \frac{0,1 + 0 + 0 + 0,1}{4} = 0,05$$

и вектор-столбец нормированных весов:

$$w = \begin{pmatrix} 0,34 \\ 0,31 \\ 0,20 \\ 0,10 \\ 0,05 \end{pmatrix}$$

Если дополнительно представить групповое мнение в виде **вектора-столбца рангов** оцениваемых факторов, то с учетом того, что более предпочтительному фактору присвоен больший вес, получаем:

$$R = \begin{pmatrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{pmatrix}$$

После получения группового мнения необходимо оценить, насколько можно доверять полученному результату. Очевидно, что степень доверия к нему будет тем выше, чем выше плотность оценок факторов, данных экспертами. Например, для одного и того же результата (группового мнения) варианты рассеяния мнений экспертов могут быть такими, как показано на рис. 17.

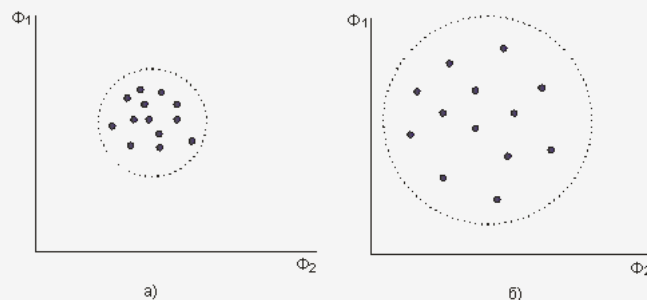


Рис. 17. Возможные варианты экспертного оценивания

Одной из широко распространенных мер оценки плотности мнений для метода ранжирования является **коэффициент конкордации Кендалла**. Коэффициент W изменяется от 1 до приблизительно нуля, при этом он равен 1, если ранги всех факторов, присвоенные экспертами, совпадают, и наоборот, равен нулю, если они образуют все возможные перестановки (в случае $n=m$).

Для нахождения W для каждой строки матрицы рангов $R = \{R_j\}$ вычисляется сумма ее элементов $R_i = \sum_{j=1}^m R_{ij}$.

$$\{R_i\} = \begin{pmatrix} R_1 \\ R_2 \\ R_3 \\ R_4 \\ R_5 \end{pmatrix} = \begin{pmatrix} 6,5 \\ 7,5 \\ 12,0 \\ 16,0 \\ 18,0 \end{pmatrix}$$

после чего вычисляется среднее значение \bar{R} по матрице R в целом:

$$\bar{R} = \frac{1}{n} \sum_{i=1}^n R_i = \frac{1}{n} \sum_{i=1}^n \sum_{j=1}^m R_{ij} = \frac{1}{n} \left(\frac{mn(n+1)}{2} \right) = \frac{m(n+1)}{2} = \frac{4(5+1)}{2} = 12$$

Далее определяется сумма s квадратов отклонений значений в строке R_i матрицы R от \bar{R} :

$$s = \sum_{i=1}^n (R_i - \bar{R})^2 = (6,5 - 12)^2 + (7,5 - 12)^2 + (12,5 - 12)^2 + (16,5 - 12)^2 + (18 - 12)^2 = \\ = 5,5^2 + 4,5^2 + 0^2 + 4^2 + 6^2 = 30,25 + 20,25 + 16 + 36 = 102,5$$

Коэффициент W вычисляется на основе выражения:

$$W = \frac{12s}{m^2(n^3 - n)} = \frac{12 \cdot 102,5}{16 \cdot (125 - 5)} = \frac{1230}{1920} \approx 0,64$$

Значения W большие 0,6-0,7 свидетельствуют о достаточно высокой плотности мнений экспертов, т.е., для данного случая качество проведенной экспертизы можно считать приемлемым.

Вопрос 4. Метод Дельфи.

Метод Дельфи впервые был описан в докладе американской корпорации РЭНД (RAND) в 1964 г. Впоследствии он стал основным средством повышения объективности экспертных опросов с использованием количественных оценок при оценке деревьев цели и при разработке сценариев за счет использования обратной связи, ознакомления экспертов с результатами предшествующего тура опроса и учета этих результатов при оценке значимости мнений экспертов.

Процедура метода Дельфи заключается в следующем:

- 1) организуется последовательность циклов «мозговой атаки»;
- 2) разрабатывается программа последовательных индивидуальных опросов с помощью вопросников, исключающая контакты между экспертами, но предусматривающая ознакомление их с мнениями друг друга между турами; вопросники от тура к туру могут уточняться;
- 3) в наиболее развитых методиках экспертам присваиваются весовые коэффициенты значимости их мнений, вычисляемые на основе предшествующих опросов, уточняемые от тура к туру и учитываемые при получении обобщенных результатов оценок.

Как правило, на практике оказывается достаточным проведение четырех туров опросов, после чего мнения экспертов либо сближаются, либо распадаются на несколько групп существенно различающихся мнений. В первом случае достигнутый результат со значительной степенью обоснованности может быть рассмотрен в качестве группового решения, во втором — необходимо продолжить исследование проблемы с учетом выдвигаемых различными группами аргументов.

Метод Дельфи имеет несомненные преимущества по сравнению с методами, основанными на обычной статистической обработке результатов индивидуальных опросов или мнений отдельных экспертов. Он позволяет уменьшить колебания по всей совокупности отдельных мнений и колебания внутри групп. При этом, как показывает практика, влияние на групповую оценку экспертов невысокой квалификации оказывается менее заметным, поскольку они получают возможность скорректировать ответы за счет получения информации от группы.

Вместе с тем, необходимо отметить и недостатки метода Дельфи, а именно:

- значительный расход времени на проведение экспертизы, связанный с большим количеством последовательных повторений оценок;
- необходимость неоднократного пересмотра экспертами своих ответов, что вызывает негативные реакции и отражается на результатах экспертизы.

Выводы:

1. Информационные системы принадлежат к категории целенаправленных систем, поэтому ключевую роль в успешном их создании играет правильный выбор целей их функционирования и их точная формулировка. Эффективным способом решения задачи целеполагания является метод структуризации целей.
2. На начальных этапах жизненного цикла необходимо принимать ряд решений, касающихся назначения системы, требований к ее характеристикам, стратегии внедрения. В силу недостаточности сведений относительно изучаемых процессов и малой степени определенности в отношении требований к планируемой для внедрения системе применяются подходы, основанные на качественном оценивании возможных альтернатив.
3. В число наиболее часто применяемых методик создания и качественного оценивания альтернатив входят методы и процедуры основанные на групповом подходе к выработке и оцениванию альтернатив. К таким методам относятся рассмотренные методы мозгового штурма, экспертного оценивания и метод Дельфи.
4. В силу отсутствия количественных оценок показателей качества и формального аппарата результаты применения рассмотренных методик в той или иной мере отражают субъективные предпочтения участников.

Вопросы для самопроверки:

1. Что понимается под целью?
2. Как классифицируются цели?
3. Что собой представляет метода структуризации целей?
4. Какие требования предъявляются к целям?
5. Что такое мозговой штурм?
6. Как организуется мозговой штурм?
7. Что такое экспертиза?

8. Из каких основных шагов состоит процесс проведения экспертного оценивания?
9. Каким образом определяется групповое мнение на основании собранных данных экспертизы?
10. Как определяется и как рассчитывается коэффициент конкордации?
11. Каким должно быть значение коэффициента конкордации качественно проведенной экспертизы?
12. В чем состоит суть метода Дельфи?
13. Из каких основных шагов состоит процедура метода Дельфи?
14. Какие преимущества дает применение метода Дельфи?
15. В чем заключаются недостатки метода Дельфи?

Литература по теме:

Основная литература:

1. Анфилатов В. С. Системный анализ в управлении: учебное пособие/ Анфилатов В. С. , Емельянов В. С. , Кукушкин А. А. – М.: Финансы и статистика, 2008. - 368с.
2. Теория систем и системный анализ в управлении организациями: справочник/ ред. В. Н. Волкова и А. А. Емельянов. – М.: Финансы и статистика, 2009. - 848с

Практические задания.

Задание 1.

Рассчитайте коэффициент согласованности для результатов экспертизы, представленных матрицей рангов:

	Э1	Э2	Э3	Э4	Э5
Ф1	3,0	4,0	3,5	2,5	2,0
Ф2	2,0	3,0	3,5	4,0	4,0
Ф3	1,0	1,0	1,0	2,5	1,0
Ф4	5,0	5,0	5,0	5,0	6,0
Ф5	6,0	2,0	2,0	1,0	3,0
Ф6	4,0	6,0	6,0	6,0	5,0

Тема 4. Количественные методы системного анализа в теории информационных процессов и систем

Цели изучения темы:

- познакомиться с основными подходами, возникающими на начальных этапах жизненного цикла информационных систем, и существующими подходами к их решению.

Задачи изучения темы:

- изучить подходы к формальному описанию качества систем;
- познакомиться с критериями количественного оценивания систем;
- понять сущность методов количественного оценивания.

Успешно изучив тему, Вы:

получите представление о:

- классификации условий функционирования систем;
- методах, применяемых для количественного оценивания альтернатив;
- как выбрать метод оценивания в зависимости от условий;
- как применяются методы оценивания;

будете знать:

- что такое частные и обобщенный показатели качества системы;
- что такое нормирование показателей;
- как уменьшить сложность задачи выбора;
- как осуществить свертку частных показателей.

Вопросы темы:

1. Количественное оценивание систем.
2. Оценивание в условиях определенности.
3. Оценивание в условиях риска.
4. Оценивание в условиях неопределенности.

Вопрос 1. Количественное оценивание систем.

Как уже обсуждалось ранее, для описания свойств системы вводится понятие показателя. Ряд этих показателей используется как **показатели качества** системы. Однако оценивание информационной системы как системы сложной не может ограничиваться рассмотрением одного единственного показателя. В частности, принятие решения относительно возможных схем организации потока работ должно учитывать целую совокупность технологических (число и сложность этапов, время выполнения и пр.) и экономических характеристик (трудовые и финансовые затраты), с которыми сопряжена реализация каждой из анализируемых схем.

Поэтому дополнительно к **частным показателям** качества некоторой системы j вводится понятие **обобщенного показателя качества**, под которым понимают вектор $Y^j = \langle y_1^j, y_2^j, \dots, y_n^j \rangle$. Компонентами этого вектора являются частные

показатели качества системы, каждый из которых может принимать значения из множества допустимых значений. Размерность обобщенного вектора показателя качества определяется числом существенных свойств системы (для информационной системы в число этих свойств могут входить функциональные, надежностные, временные и др.). Следует отметить, что обобщенным показателем качества является именно вектор, а не простое множество частных показателей, поскольку между отдельными свойствами могут существовать связи, которые в рамках теории множеств описать весьма сложно.

Частные показатели имеют различную физическую природу и в соответствии с этим различную размерность. Поэтому при переходе к обобщенному показателю качества следует оперировать не с самими показателями, а с их **нормированными** значениями, обеспечивающими приведение показателей к одному масштабу, что необходимо для их сопоставления.

Задача нормировки решается, как правило, введением относительных безразмерных показателей, представляющих собой отношение измеренного частного показателя к некоторой нормирующей величине, измеряемой в тех же единицах, что и сам показатель:

$$y_i^{норм} = \frac{y_i}{y_i^o},$$

где y_i^o - некоторое «идеальное» значение i -го показателя. Выбор этого значения в значительной мере носит субъективный характер и должен обосновываться в каждом конкретном случае. Укажем несколько возможных подходов к выбору нормирующего делителя.

- 1) y_i^o задается ЛПР (т.е., значение y_i^o следует воспринимать как образцовое);
- 2) $y_i^o = \max y_i$;
- 3) $y_i^o = y_i^{\max} - y_i^{\min}$.

Требуемое качество системы задается правилами (условиями), которым должны удовлетворять показатели существенных свойств, а проверка их выполнения называется **оцениванием качества системы**. Таким образом, критерий качества есть показатель существенных свойств системы и правило его оценивания.

Все критерии в общем случае могут принадлежать одному из трех классов.

Согласно **критерию пригодности** j -ая система считается пригодной, если значения всех частных показателей y_i^j этой системы принадлежат области допустимых значений частных показателей.

Согласно **критерию оптимальности** j -ая система считается оптимальной по i -му показателю качества, если все значения частных показателей качества y_i^j принадлежат допустимой области и существует хотя бы один частный показатель качества y_i^j , по которому достигается оптимум.

Согласно **критерию превосходства** j -ая система считается превосходной, если все значения частных показателей качества y_i^j принадлежат допустимой области и система оптимальна по всем показателям.

Пример применения критериев иллюстрируется

Рис. 18, где по свойствам y_1 и y_2 сравниваются характеристики пяти систем S^1, S^2, S^3, S^4, S^5 , имеющих допустимые области значений y_1 и y_2 , для которых оптимальные значения определены как $y_1^{opt} = y_1^{\max}$ и $y_2^{opt} = y_2^{\max}$ соответственно.

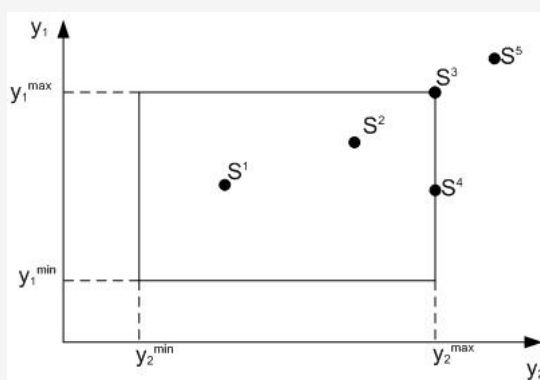


Рис. 18. Примеры применения критериев качества

Из рисунка видно, что системы S^1, S^2, S^3, S^4 пригодны по показателям y_1 и y_2 , система S^5 не является пригодной, системы S^3, S^4 оптимальны по показателю y_2 , система S^3 является превосходной.

Принято различать три вида условий, в которых может производиться оценивание сложных систем и принятие решения, а именно:

- условия определенности;
- условия риска;
- условия неопределенности.

Вопрос 2. Оценивание в условиях определенности.

Под условиями определенности понимают такие условия, когда поведение системы (и, в частности, значения каждого ее показателя качества) полностью предсказуемы.

Оценка сложных систем в условиях определенности на основе методов векторной оптимизации проводится в три этапа:
 На *первом этапе* с использованием системного анализа определяются частные показатели и критерии эффективности.
 На *втором этапе* находится множество Парето и формулируется задача многокритериальной оптимизации.
 На *третьем этапе* задача решается путем скаляризации критериев с устранением многокритериальности.

Множество Парето определяется как подмножество A^* множества альтернатив A . Множество A^* (**переговорное множество, множество компромиссов**) включает альтернативы, которые всегда более предпочтительны по сравнению с любой альтернативой из множества $A \setminus A^*$. При этом любые две альтернативы из множества Парето по предпочтению несравнимы. Альтернативы a_i и a_j называются **несравнимыми**, если альтернатива a_i превосходит альтернативу a_j по одним критериям, а альтернатива a_j превосходит альтернативу a_i по другим. Выражение $K(a^*) > K(a)$ означает, что:

$$k_1(a^*) \geq k_1(a); k_2(a^*) \geq k_2(a); \dots k_n(a^*) \geq k_n(a)$$

и хотя бы одно из этих неравенств является строгим.

Понятие множества Парето можно пояснить на примере. Пусть имеем задачу оптимизации по двум критериям $k_1: \rightarrow \min y_1$, $k_2: \rightarrow \min y_2$ где y_1 и y_2 - показатели свойств системы (параметры), значения которых можно выбирать. Целью является выбор оптимальных (в данном случае минимальных) значений параметров. Если изобразить множество критериев в виде многоугольника (рис. 19):

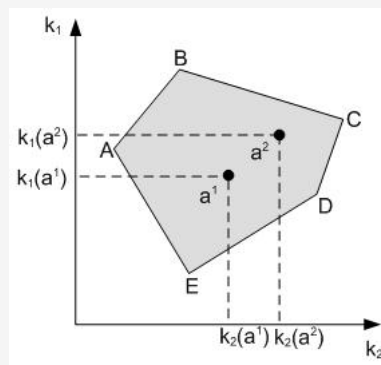


Рис. 19. Множество Парето

и рассмотреть пару точек a^1 и a^2 как показано на рисунке, то легко заметить, что для этих точек справедливо:

$$k_1(a^1) < k_1(a^2)$$

$$k_2(a^1) < k_2(a^2)$$

т.е., точка a^1 является более предпочтительной по отношению к точке a^2 . Поэтому оставлять для рассмотрения точку a^2 не имеет смысла.

Продолжая рассмотрение других точек области многоугольника, приходим к тому, что для выбора остаются только точки стороны AE . Для каждой пары этих точек предпочтение по одному показателю сопровождается ухудшением по другому.

Таким образом, нахождение множества Парето может значительно уменьшить затраты на решение задачи выбора за счет сокращения числа подлежащих рассмотрению вариантов.

Методы, основанные на свертке векторного критерия в скалярный, предполагают переход к решению задачи:

$$k(a) \rightarrow \underset{a \in A}{extr},$$

где $k(a)$ - скалярный критерий, представляющий собой некоторую функцию от значений компонентов векторного критерия:

$$k(a) = f(k_1(a), k_2(a), \dots k_n(a))$$

Основной проблемой этого подхода является построение функции f , называемой **сверткой**. Свертка производится в несколько этапов.

Обоснование допустимости свертки. Требуется подтверждения, что рассматриваемые показатели эффективности являются однородными. Известно, что показатели эффективности разделяются на три группы: показатели результативности, ресурсоемкости и оперативности. В общем случае разрешается свертка показателей, входящих в обобщенный показатель для каждой группы отдельно. Свертка показателей разных групп может привести к потере физического смысла такого критерия.

Нормализация критериев. Проводится аналогично нормировке показателей.

Учет приоритетов критериев. Осуществляется в большинстве методов свертывания путем задания вектора коэффициентов важности критериев λ :

$$\lambda = \lambda_1, \lambda_2, \dots, \lambda_n, \sum_{i=1}^n \lambda_i = 1,$$

где λ_i - коэффициент важности критерия k_i , обычно совпадающий с коэффициентом значимости частного показателя качества.

Определение коэффициентов важности критериев, как и в случае с показателями, сталкивается с серьезными трудностями и сводится либо к использованию формальных процедур, либо к применению экспертных оценок.

В результате нормализации и учета приоритетов критериев вместо исходной векторной оценки $K(a)$ альтернативы a образуется новая векторная оценка:

$$k(a) = (\lambda_1 k_1(a), \lambda_2 k_2(a), \dots, \lambda_n k_n(a)),$$

где $k_i(a)$ - нормированный критерий, он находится аналогично нормированному показателю. Именно эта полученная векторная оценка подлежит преобразованию с использованием функции свертки.

Свертка показателей. Способ свертки зависит от характера показателей и целей оценивания систем. Известны несколько видов свертки. Наиболее часто используют аддитивную и мультипликативную свертки компонентов векторного критерия.

Аддитивная свертка компонентов векторного критерия состоит в представлении обобщенного скалярного критерия в виде суммы взвешенных нормированных частных критериев:

$$k(a) = \sum_{i=1}^n \lambda_i \cdot \frac{k_i(a)}{k_i^0(a)} \quad (1)$$

Такие критерии образуют группу аддитивных критериев, их свертка основана на использовании принципа справедливой компенсации абсолютных значений нормированных частных критериев. Сутью этого принципа является утверждение: справедливым следует считать такой компромисс, при котором суммарный уровень абсолютного снижения значений одного или нескольких показателей не превышает суммарного уровня абсолютного увеличения значений других показателей. Главный недостаток аддитивных критериев состоит в том, что они не вытекают из объективной роли частных критериев в определении качества системы и поэтому выступают как формальный математический прием, придающий задаче удобный вид. Кроме того, низкие оценки по одним критериям могут компенсироваться высокими оценками по другим критериям. Это значит, что уменьшение одного из критериев вплоть до нулевого значения может возмещаться возрастанием другого.

Мультипликативная свертка критерия состоит в представлении обобщенного скалярного критерия в виде произведения:

$$k(a) = \prod_{i=1}^n k_i(a)^{\lambda_i} \quad (2)$$

Мультипликативный критерий образуется путем перемножения частных критериев $k_i(a)$ возведенных в степени λ_i . Если все частные критерии имеют одинаковую важность, то $\lambda_i = 1$. При разной важности критериев $\lambda_i \neq 1$.

В мультипликативных критериях компромисс достигается по отношению не к абсолютным, а к относительным изменениям частных критериев, а именно, суммарный уровень относительного снижения значений одного или нескольких критериев не превышает суммарного уровня относительного увеличения значений других критериев.

Достоинством мультипликативного критерия является то, что при его использовании не требуется нормировки частных критериев. К его недостаткам относится то, что он компенсирует недостаточную величину одного частного критерия избыточной величиной другого и имеет тенденцию сглаживать уровни частных критериев за счет неравнозначных первоначальных значений частных критериев.

Выбор между аддитивной и мультипликативной свертками частных критериев определяется степенью важности абсолютных или относительных изменений значений частных критериев соответственно.

Вопрос 3. Оценивание в условиях риска.

Условиями риска называют такие условия функционирования систем, когда результат выполняемых системой операций (действий, работ) характеризуется как вероятностный. Однозначность соответствия между системами и исходами в вероятностных операциях нарушается. Например, в силу ненадежности сетевого оборудования время передачи сообщений может меняться случайным образом по известному закону. Очевидно, оценивать системы, функционирующие в условиях риска, таким же образом, как в детерминированных системах, нельзя.

Эффективность функционирования систем в условиях риска чаще всего находится через математическое ожидание функции полезности на множестве исходов отдельных операций. Поясним применение процедуры оценивания на следующем примере.

Пусть необходимо оценить варианты оборудования локальной вычислительной сети с целью выбора наилучшего. Полезность оборудования определяется обеспечиваемой им интенсивность обмена сообщениями между пользователями, функция полезности строится экспертным путем (в зависимости от числа передаваемых в сети сообщений). Исходные данные для оценивания представлены в табл. 1:

Таблица 1.

Исходные данные для оценивания

i	S_k	$P(S_k / i)$	$F(S_k)$

1	60	0,30	0,8
	40	0,50	0,5
	20	0,20	0,1
2	60	0,25	0,8
	40	0,60	0,5
	20	0,15	0,1

Здесь i - номер варианта сетевого оборудования, $S_k, k=1,2,3$, - возможные варианты производительности (число передаваемых в течение некоторого времени сообщений), $P(S_k / i), k=1,2,3, i=1,2$ - вероятность того, что в сети i будет передано S_k сообщений, $F(S_k)$ - значение функции полезности передачи по сети S_k сообщений.

Расчет математического ожидания функции полезности \bar{F}_i для каждого варианта i сетевого оборудования производим согласно определению математического ожидания:

$$\bar{F}_i = \sum_{k=1}^3 F(S_k)P(S_k / i), i=1,2$$

Подставляя значения из таблицы, получаем:

$$\bar{F}_1 = 0,3 \cdot 0,8 + 0,50 \cdot 0,5 + 0,20 \cdot 0,1 = 0,510$$

$$\bar{F}_2 = 0,25 \cdot 0,8 + 0,60 \cdot 0,15 + 0,20 \cdot 0,1 = 0,515$$

Из полученных оценок заключаем, что предпочтение следует отдать варианту сетевого оборудования № 2.

Кроме оптимизации в среднем в вероятностных операциях в теории принятия решений рассматриваются и используются и другие критерии оценки систем:

- максимум вероятности случайного события;
- максимум степени вероятностной гарантии достижения результата не ниже требуемого уровня;
- минимум среднего квадрата отклонения результата от требуемого;
- минимум дисперсии результата;
- максимум вероятностно-гарантированного результата;
- минимум среднего (байесовского) риска (минимум средних потерь).

Вопрос 4. Оценивание в условиях неопределенности.

Специфические черты информационных систем не позволяют при решении ряде задач свести их ни к детерминированным, ни к вероятностным и, следовательно, применить методы, применяемые в этих случаях. Описание условий неопределенности, удобное с точки зрения применения методов оценивания, можно представить в виде таблицы (Табл. 2):

Таблица 2.

Исходные данные для оценивания в условиях неопределенности

i	n_1	n_2	...	n_l
1	k_{11}	k_{12}	...	k_{1l}
2	k_{21}	k_{22}	...	k_{2l}
...
n	k_{n1}	k_{n2}	...	k_{nl}

Здесь i – идентификатор системы (операции), n_j - состояние внешней среды, k_{ij} - эффективность системы i для состояния внешней среды n_j .

Для оценивания альтернатив в теории принятия решения используются различные критерии. К числу наиболее часто используемых относятся критерии:

- среднего выигрыша;
- Лапласа;
- осторожного наблюдателя (критерий Вальда);
- максимакса;
- пессимизма-оптимизма (критерий Гурвица);
- минимального риска (критерий Сэвиджа).

Критерии основаны на разных презумпциях, поэтому выбор того или иного во многом определяется субъективными предпочтениями аналитика или ЛПР. Поясним сказанное на примере применения двух из вышеназванных критериев.

Пусть необходимо оценить пять программных продуктов для защиты от вирусных атак. Известны значения эффективности применения каждой программы по отражению каждого вида атак, общее число видов атак равно 4 (Табл. 3).

Таблица 3.

Данные для примера оценивания

	1	2	3	4
--	---	---	---	---

1	0,4	0,3	0,4	0,2
2	0,2	0,4	0,5	0,2
3	0,1	0,4	0,5	0,3
4	0,4	0,4	0,3	0,2
5	0,5	0,4	0,3	0,3

Каждая i -ая строка таблицы содержит значения эффективности k_{ij} применения i -ой программы, $i=1,..5$, для всех видов вирусов, а каждый j -ый столбец - значения эффективности k_{ij} применения для борьбы с j -ым вирусом, $j=1,..4$, всех программ.

Для определения наилучшего варианта (в данном случае, антивирусной программы) в первых пяти вышеперечисленных критериях используется правило, в соответствии с которым для этого наилучшего варианта должно выполняться:

$$K_{opt} = \max_i K_i,$$

где K_i - значение обобщенного критерия эффективности применения i -ой программы. Вместе с тем, каждый из этих критериев использует свое правило для нахождения этого значения.

В частности, согласно **критерию Вальда** значения критерия для каждого i -го варианта K_i находятся как минимально возможные значения критерия для всех вариантов j -ой обстановки (что объясняет причину второго названия критерия – осторожного наблюдателя):

$$K_i = \min_j k_{ij}$$

В нашем примере минимальная эффективность применения каждой из программ для всех вирусов будет соответственно:

$$K_1 = 0,2;$$

$$K_2 = 0,2;$$

$$K_3 = 0,1;$$

$$K_4 = 0,2;$$

$$K_5 = 0,3.$$

Таким образом, по критерию Вальда следует считать наилучшей пятую программу как имеющую наибольшее значение критерия из всей совокупности.

Смысл **критерия Сэвиджа** состоит в выборе варианта, минимизирующего потери эффективности при наихудших условиях. Для оценки систем на основе данного критерия матрица эффективности должна быть преобразована в матрицу потерь (риска). Каждый элемент матрицы потерь определяется как разность между максимальным и текущим значениями оценок эффективности в столбце

$$\Delta k_{ij} = \max_i k_{ij} - k_{ij}$$

В нашем примере для матрицы потерь имеем (значения максимумов по столбцам $\max_i k_{ij}$ равны соответственно 0,5; 0,4; 0,5; 0,3):

	1	2	3	4
Δk_{1j}	0,1	0,1	0,1	0,1
Δk_{2j}	0,3	0,0	0,0	0,1
Δk_{3j}	0,4	0,0	0,0	0,0
Δk_{4j}	0,1	0,0	0,2	0,1
Δk_{5j}	0,0	0,0	0,2	0,0

По матрице потерь определяются значения критерия как:

$$K_i = \max_j \Delta k_{ij}$$

Для нашего примера:

$$K_1 = 0,1;$$

$$K_2 = 0,3;$$

$$K_3 = 0,4$$

$$\Delta_3 = 0,4;$$

$$K_4 = 0,2;$$

$$K_5 = 0,2.$$

Очевидно, что целесообразно выбирать тот вариант системы, который обеспечит минимум потерь, т.е., для которого:

$$K_{opt} = \min_i \{ \max_j \Delta k_{ij} \}$$

В нашем случае минимальные потери должны быть в случае выбора первой программы, для которой $K_1 = 0,1$. Критерий Сэвиджа, как и критерий Вальда, относится к числу осторожных критериев.

Выводы:

1. Если для оценивания вариантов принимаемых решений используются показатели качества, значение которых можно выразить количественно, применяются специальные методы оценивания. Методы могут использовать как частные, так и обобщенные показатели качества оцениваемой системы. Обобщенные показатели качества получаются из частных на основе применения специальных правил и процедур.
2. Для решения задачи выбора наилучшего варианта решения из имеющегося множества показатели обычно нормализуются, что дает возможность их сопоставимости за счет устранения размерности и установления одинаковых областей изменения.
3. Выбору наилучшего решения из множества возможных обычно предшествует нахождение множества Парето, что позволяет значительно уменьшить трудоемкость процедуры.
4. Наиболее употребительным подходом к выбору наилучшего решения является подход на основе свертки частных показателей. Он позволяет применить один из математических методов оптимизации или формальных критериев оценивания.
5. Выбор критерия и метода количественного оценивания определяется условиями функционирования оцениваемой системы и наличием информации относительно окружающей среды и возможного поведения системы. Условия подразделяются на условия определенности, риска и неопределенности, для каждого вида условий разработаны свои методы решения задачи.

Вопросы для самопроверки:

1. Что такое «показатель», какие типы показателей существуют?
2. Что называется частным показателем качества системы?
3. Что такое обобщенный показатель, как можно осуществить переход к обобщенному показателю?
4. Зачем требуется нормирование показателей, какие правила для этого могут применяться?
5. Какие существуют критерии оценивания качества системы?
6. Что собой представляет и зачем необходима свертка показателей?
7. Из каких основных этапов состоит процедура свертки?
8. Как определяется аддитивный критерий свертки?
9. Как определяется мультипликативный критерий свертки?
10. В чем заключаются преимущества и недостатки аддитивного и мультипликативного критериев свертки?
11. Что такое множество Парето и как оно используется в оценивании альтернатив?
12. Какими условиями функционирования систем определяется выбор подхода к оцениванию?
13. Что может приниматься в качестве критерия выбора наилучшей альтернативы в условиях риска?
14. Какие существуют методы выбора наилучшей альтернативы в условиях неопределенности?
15. В чем состоит сущность критерия Вальда?
16. В чем состоит сущность критерия Сэвиджа?

Литература по теме:

Основная литература:

1. Анфилов В. С. Системный анализ в управлении: учебное пособие/ Анфилов В. С. , Емельянов В. С. , Кукушкин А. А. – М.: Финансы и статистика, 2008. - 368с.
2. Теория систем и системный анализ в управлении организациями: справочник/ ред. В. Н. Волкова и А. А. Емельянов. – М.: Финансы и статистика, 2009. - 848с.

Практические задания.

Задание 1.

Для условий неопределенности определите наилучшую по критерию Сэвиджа систему из пяти имеющихся. Показатели эффективности применения каждой системы для шести вариантов внешней обстановки даны в таблице:

	1	2	3	4	5	6
1	12,3	9,7	17,7	18,8	14,3	13,1
2	14,8	10,2	13,3	14,6	16,9	15,7
3	13,4	9,7	17,0	19,4	11,9	10,4
4	15,6	7,8	18,7	19,4	17,3	17,6
5	16,4	7,5	15,8	18,6	19,0	19,2

Цели изучения темы:

- познакомиться с постановкой задачи моделирования деловых процессов в контексте создания информационных систем;
- познакомиться с основными подходами к моделированию деловых процессов.

Задачи изучения темы:

- изучить основные термины и понятия, относящиеся к моделированию деловых процессов;
- познакомиться с наиболее известными методологиями моделирования деловых процессов.

Успешно изучив тему, Вы:

получите представление о:

- сущности, назначении и применении деловых моделей;
- методиках, применяемых для создания деловых моделей;
- проблемах и тенденциях развития методик создания деловых моделей;

будете знать:

- требования к деловым моделям;
- особенности и средства конкретных методологий, таких IDEF;
- основные принципы процесса разработки деловых моделей.

Вопросы темы:

1. Деловые процессы и их моделирование.
2. Методология моделирования IDEF.
3. Язык моделирования UML.
4. Автоматизация моделирования деловых процессов.

Вопрос 1. Деловые процессы и их моделирование.

Как уже отмечалось, информационная система в настоящее время должна рассматриваться не как самостоятельная единица, а в ее тесной взаимосвязи с основной деятельностью предприятия или структуры, в которую она включается как органическая составная часть. Поэтому требования к системе формулируются на основе детального рассмотрения всех протекающих в реальной системе (организации, структуре) **деловых процессов**, к которым будем относить *любой вид регулярно выполняющихся действий или операций, направленных на достижение одной или нескольких определенных целей и имеющих (или использующих) необходимый для этого набор ресурсов*.

Здесь следует сделать небольшое отступление. Часто английский термин *business process* в отечественной литературе дается в транслитерированном виде – *бизнес-процесс*. Оставляя без обсуждения возможные в подобных случаях издержки для русского языка, заметим, что перевод термина на русский язык (*деловой процесс*) более точно передает его смысл. Английское *business* означает (согласно словарю Merriam-Webster) деятельность преимущественно коммерческого характера. Однако в данном контексте под термином *business process* подразумеваются процессы, протекающие не обязательно в коммерческих организациях, к ним могут относиться процессы, протекающие, например, в органах государственного управления или благотворительных организациях. На этот расширительный смысл термина авторы книг на английском языке часто особо обращают внимание читателя. В русском языке слово *деловой* в отличие от английского *business* не несет никакого коммерческого оттенка, поэтому в дополнительных разъяснениях необходимости нет.

Цели выражают желаемое состояние ресурсов, которое достигается в результате протекания процессов. Цели формулируются в виде одного или нескольких *правил*.

Понятие **ресурс** включает объекты, которые использует или потребляет процесс. В качестве таких объектов могут быть выступать люди, сырье, информация или произведенный продукт. Ресурсы образуют связанные между собой структуры. Ресурсы могут представлять собой объекты, которые потребляются или подвергаются преобразованию, такие как сырье в материальном производстве. Этот вид ресурсов относится к **входным** ресурсам. **Выходными** ресурсами являются конечные результаты, имеющие ценность для их получателя и могут представлять собой совершенно новый объект или объект, получившийся как преобразование входных ресурсов. В задачах анализа бывает полезно классифицировать ресурсы на **материальные** (физические), **абстрактные** и **информационные**.

Правило (деловое правило) представляет собой утверждение относительно каких-либо характеристик деловых процессов или наложенных на них ограничений. Правила могут отражать особенности внешней среды и иметь вид законов, постановлений, государственных или местных стандартов, либо диктоваться внутренними условиями в виде установленных корпоративных стандартов и нормативов, распорядка или регламента. Они могут указывать способ, которым должны выполняться операции, требования к составу и параметрам промежуточных и итоговых результатов, вводить ограничения на использование ресурсов, задавать последовательность выполнения отдельных операций. Правила можно классифицировать на **функциональные**, **поведенческие** и **структурные**.

Формулировку целей и распределение ресурсов осуществляет **владелец деловых процессов**.

Проектирование среды протекания деловых процессов, их логической структуры и параметров, определение потребностей характера необходимых ресурсов, их объема и распределения входят в задачу **аналитика**, которую он решает в типичном случае на основе разработанной им **модели**. Моделирование деловых процессов преследует цель получить абстракцию реально протекающих процессов, предоставив тем самым возможность всем лицам, имеющим отношение к работе предприятия, во-первых, **понять** текущее состояние, и, во-вторых, определить пути для его *улучшения* и *инноваций*. Последние термины нуждаются в уточнении.

Под **улучшением** деловых процессов понимаются их небольшие поэтапные изменения, носящие непрерывный эволюционный характер. **Инновации** означают существенные изменения, носящие дискретный характер. Дающие потенциальную возможность значительно повысить эффективность деловых процессов инновации вместе с тем характеризуются высоким уровнем риска неудачи. Наиболее радикальным видом инноваций является **реинжиниринг** деловых процессов, применение которого может, с одной стороны, повысить эффективность в несколько раз, с другой стороны, увеличивает риск.

Одним из основных вариантов инновации является внедрение информационной системы, и здесь деловая модель помогает решить одну из главных задач, решаемых на начальных этапах ее жизненного цикла. Если под **архитектурой** информационной системы понимать совокупность элементов и отношений между ними, которые формируют систему с ее основными функциями, то этой главной задачей является задача **определения архитектуры системы и требований** к ее параметрам. Реализация (физическое воплощение) системы может происходить по-разному (Рис. 20), главное, чтобы информационная система (или несколько систем) представляла собой органичную составляющую всей системы.

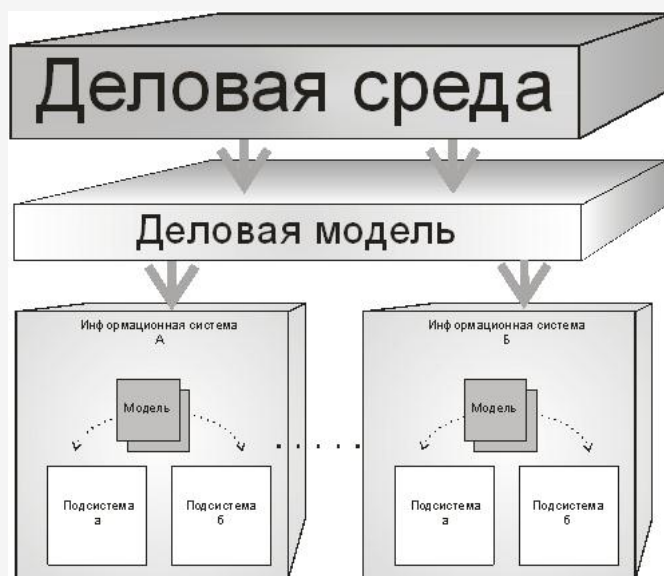


Рис. 20. Взаимосвязи моделей и систем

Построение **деловой модели** (более точно, модели деловой среды, предприятия, объекта) позволяет не только значительно снизить уровень риска за счет выявления возможных узких мест и упущений. Применение современных методов разработки моделей дает возможность использовать одну и ту же модель для анализа и проектирования на ее основе различных реализаций информационной системы, что положительно сказывается на стоимости внедрения.

Другими важными задачами, решение которых обеспечивает деловая модель, являются задачи **обучения** персонала предприятия и выявление возможных вариантов для **аутсорсинга** – передачи каких-либо из выполняемых функций (действий процесса) сторонним исполнителям.

Задачей **разработчика системы** является создание и внедрение поддерживающей деловые процессы информационной системы путем разработки новой или адаптации какой-либо существующей ее реализации.

Правильно составленная деловая модель позволяет, как отмечено выше, применять ее для определения архитектуры информационной системы. В частности, с помощью деловой модели можно достаточно точно определить такие требования к **программному обеспечению** информационной системы как:

- наиболее подходящий для данной деловой модели тип информационной системы (новая, типовая, наследованная);
- функции, которые должна выполнять система;
- тактико-технические характеристики системы.

Однако перенос компонентов деловой модели в модель программной архитектуры не может осуществляться чисто формально и требует приложения определенных усилий.

С другой стороны, обратившись к истории можно констатировать, что задача разработки программной архитектуры возникла значительно раньше, чем задача моделирования деловых процессов, и в течение этого периода для моделирования архитектуры программного обеспечения появилось довольно много методов и инструментальных средств. Несмотря на то, что прямое применение этих методов и инструментальных средств к моделированию деловых процессов невозможно, ряд подходов, предложенных для моделирования деловых процессов, основан на адаптации методологий моделирования программного обеспечения, в типичном случае - путем включения в языки моделирования специальных дополнений.

В число наиболее широко применяемых методологий входят методология IDEF и язык моделирования UML.

Вопрос 2. Методология моделирования IDEF.

Методология IDEF (Integrated DEfinition) была разработана в 1970-х годах в рамках программы BBC США под названием ICAM (Integrated Computer Aided Manufacturing). Первоначально методология предназначалась для моделирования систем производственного назначения и включала метод функционального моделирования IDEF0, метод концептуального моделирования IDEF1 и метод спецификаций моделей динамических систем IDEF2. В дальнейшем методология была дополнена другими методами и стала широко применяться для моделирования деловых процессов. В настоящее время в нее входят методы от IDEF0 до IDEF14 (IDEF7 к настоящему времени не реализован). Наиболее часто используемыми методами являются IDEF0, IDEF1x, IDEF3 и IDEF4. В силу своих интуитивно понятных не подготовленному читателю изобразительных средств, упрощающих и повышающих надежность взаимодействия аналитиков и конечных пользователей, методы моделирования IDEF получили довольно широкое распространение и продолжают совершенствоваться.

Для целей моделирования деловых процессов применяются в основном методы IDEF0 и IDEF3, в основу которых положены графические языки описания моделируемых объектов и процессов.

Начальный этап построения модели требует принятия решения по нескольким ключевым вопросам, которые уже обсуждались в разделе, посвященном основам системного анализа настоящего курса. В частности, разработчиками должны быть

определены:

Назначение модели.

В первую очередь, определяется тип создаваемой модели выбором из двух возможностей:

- Как есть (AS IS) - описание существующей деловой среды.
- Как должно быть (TO BE) - описание желаемой деловой среды, построенной с применением средств автоматизации деловых процессов.

Границы моделирования.

Специфицируются широта охвата в смысле включаемых в рассмотрение компонентов деловой среды, деловых процессов и глубина детализации их описаний. Решения во многом предопределяются решением по назначению модели.

Целевая аудитория.

Определяется круг заинтересованных лиц, для которых создаются модель.

Точка зрения.

Выбирается перспектива, под которой наблюдается деловая среда в процессе создания модели. При необходимости могут создаваться несколько моделей, отображающих различные точки зрения (например, пользователь услуг предприятия, системный администратор, финансовый директор).

IDEF0.

Метод IDEF0 предназначен для разработки **функциональных** моделей, т.е., спецификаций того, *что* должна делать система. В основу метода положена методика SADT (Structured Analysis and Design Technique), разработанная в 1970-х годах.

С помощью IDEF0 аналитик может описать деловые процессы, отображая также существенные для понимания объекты типа *вход*, *выход*, *управление* и *механизм* (*исполнительный механизм*). Эти объекты, которые часто именуются также как *ICOM* (Input-Control-Output-Mechanism), определяются следующим образом (рис. 21):

- **входы** представляют собой ресурсы, потребляемые или подвергаемые преобразованию описываемым процессом;
- **выходы** представляют собой некоторые объекты, появляющиеся как результат потребления или преобразования входов;
- **управления** представляют собой некоторые объекты, организующие и контролирующие протекание процесса - стратегии, законы, правила, стандарты;
- **механизмы** представляют собой объекты и субъекты, участвующие в выполнении отдельных операций процесса, такие как люди или устройства и оборудование.



Рис. 21. Общий вид IDEF0-диаграммы

Название функции записывается в *функциональном (процессном) блоке* в виде глагола повелительном наклонении или отглагольного существительного.

На рис. 22 показан пример IDEF0-диаграммы.



Рис. 22. Пример IDEF0-диаграммы

Основным методологическим приемом IDEF0 является *декомпозиция* – каждый функциональный блок может быть представлен своим более подробным описанием в виде отдельной (*дочерней*) диаграммы, на которой могут присутствовать несколько соединенных между собой функциональных блоков (рис. 23).

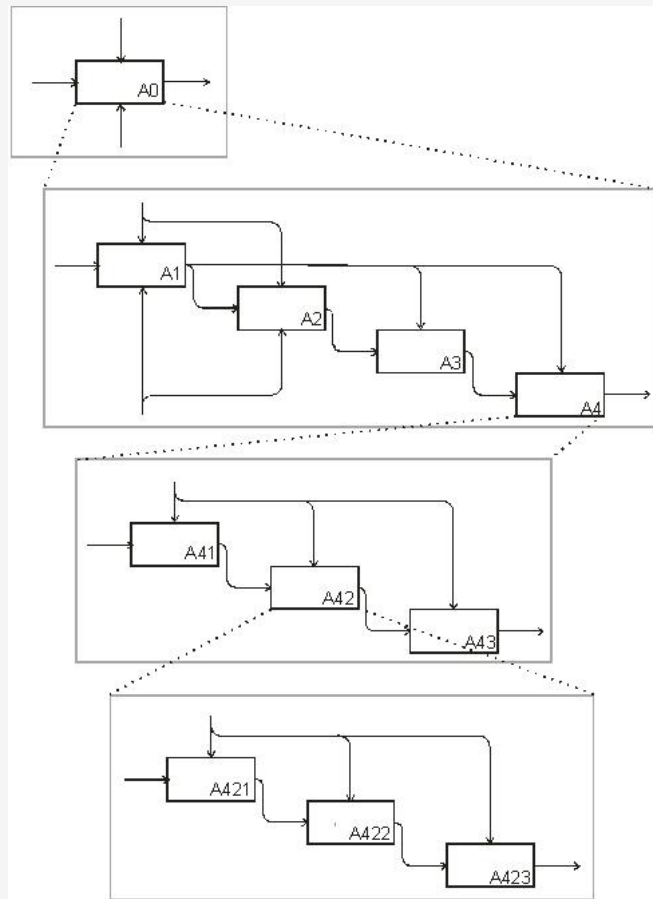


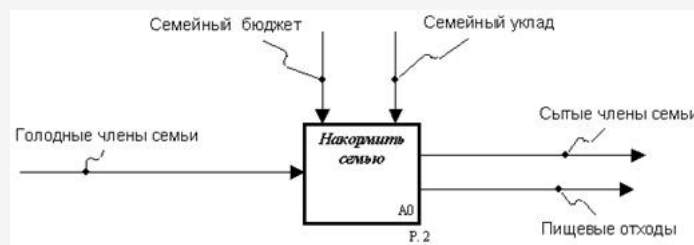
Рис. 23. Декомпозиция блоков IDEF0-диаграмм

Иерархия связей между *родительской* и *дочерними* диаграммами отображается системой обозначений блоков диаграмм. Так на рис. 23. блок диаграммы высшего уровня (*контекстной диаграммы*) обозначается A0, блоки дочерней диаграммы – A1, A2, A3, A4. Следующая диаграмма, показанная на рисунке, является декомпозицией блока A4, три блока которой обозначаются A41, A42, A43. Наконец, последняя диаграмма рисунка, дочерняя для блока A42, содержит блоки с обозначениями A421, A422, A423.

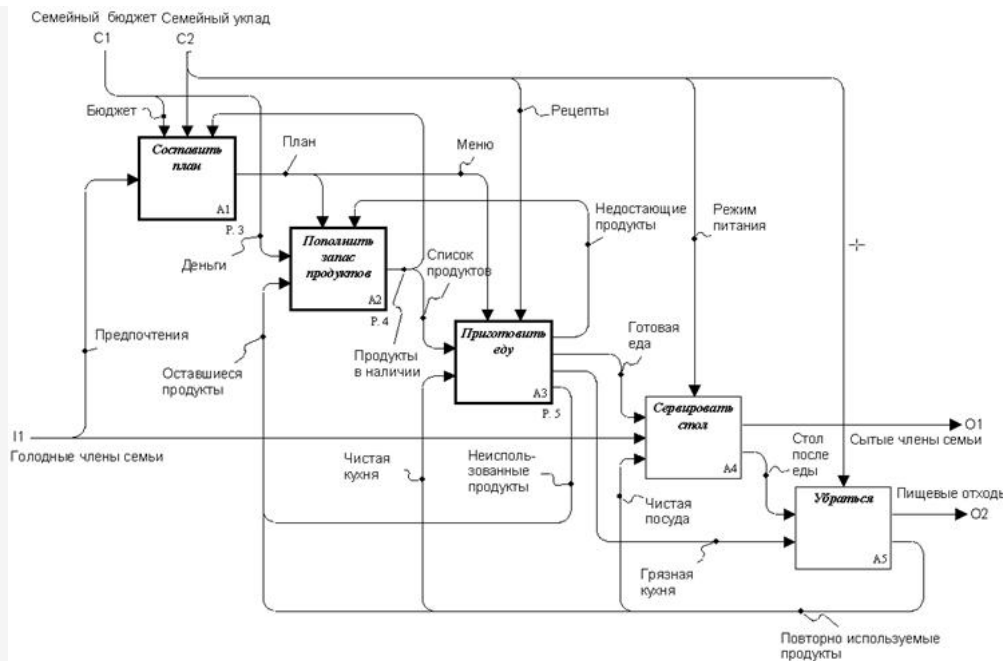
Стрелки IDEF0-модели могут, как *разветвляться*, так и *сливаться*, причем слияние сохраняет за каждой стрелкой ее содержательную идентичность.

Часть стрелок может *туннелироваться*, т.е., стрелки могут впервые появляться только на дочерней диаграмме некоторого уровня, оставаясь на всех родительских диаграммах скрытыми. Допускается также и обратное – можно отменить присутствие ненужных стрелок на всех дочерних диаграммах. Такая возможность позволяет избежать загромождения диаграмм верхних уровней излишними деталями (в первом случае) или, наоборот, сведениями слишком неконкретного характера (во втором). На IDEF0-диаграммах туннели обозначаются парой круглой скобок, стоящих в начале стрелки (скрытой на родительских диаграммах) или в конце стрелки (скрытой на дочерних диаграммах).

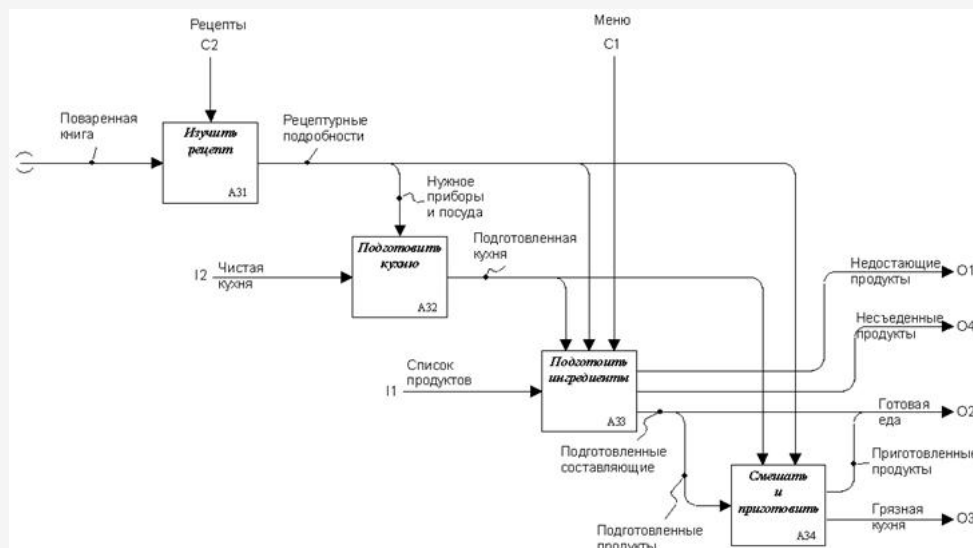
На рис. 24 показан пример, иллюстрирующий описанные механизмы.



а) контекстная диаграмма



б) диаграмма декомпозиции блока A0



в) диаграмма декомпозиции блока A3

Рис. 24. Пример декомпозиции

Помимо примеров сливающихся и ветвящихся стрелок на рисунке присутствует и пример туннеля в виде стрелки «Поваренная книга» диаграммы декомпозиции блока A3.

Однако метод IDEF0 не позволяет дать представление о хронологии протекания процессов, иначе говоря, диаграмма отражает только *состав* функций, но не логическую *последовательность* их выполнения. Элементы диаграммы могут содержать сведения относительно условий выполнения функций, но не сведения о том, каковы должны быть условия для того, чтобы процесс мог начаться, или в какой момент процесс будет завершен. Описание этих характеристик может быть получено с помощью средств моделирования IDEF3.

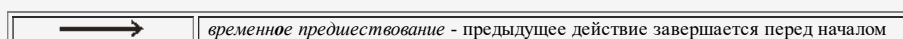
IDEF3.

Аналогично методу IDEF0 объектом рассмотрения метода IDEF3 являются деловые процессы предприятия. Модель IDEF3 дополняет описание системы IDEF0-модели информацией о *динамических (поведенческих)* аспектах протекающих в системе процессов. При этом в отличие от IDEF0-модели IDEF3-модель может сочетать в себе несколько описаний (*представлений*) о временных предшествованиях процессов. По этой причине итог применения метода правильнее, строго говоря, характеризовать как частично формализованное описание системы, но не как модель.

Изобразительные средства включают два подхода.

В первом из них, в *диаграммах протекания процесса*, сведения о процессах представляются в виде *сценария*, отдельными единицами описания которого являются *действия*, или *работы* (activity), называемые UOB (Unit Of Behavior - единица поведения), или UOW (Unit Of Work - единица работы). Применительно к конкретной задаче UOB может представлять отдельную функцию, действие, операцию или процесс. Связи с блоками IDEF0 устанавливаются посредством механизма ссылок.

Связи между действиями (arrow, link) могут быть трех типов:



	последующего;
→	объектный поток - выход предыдущего действия передается на вход последующего;
----->	отношение – конкретизируется разработчиком модели для каждого случая отдельно.

Аналогично имеющейся в методе IDEF0 возможности декомпозиции функциональных блоков, действия (UOB) в модели IDEF3 также могут декомпозироваться, поэтому для номера действия обычно используется десятичная нотация (рис. 25).



Рис. 25. Нумерация действий в модели IDEF3

На рис. 26 показан пример IDEF3-диаграммы процессов.

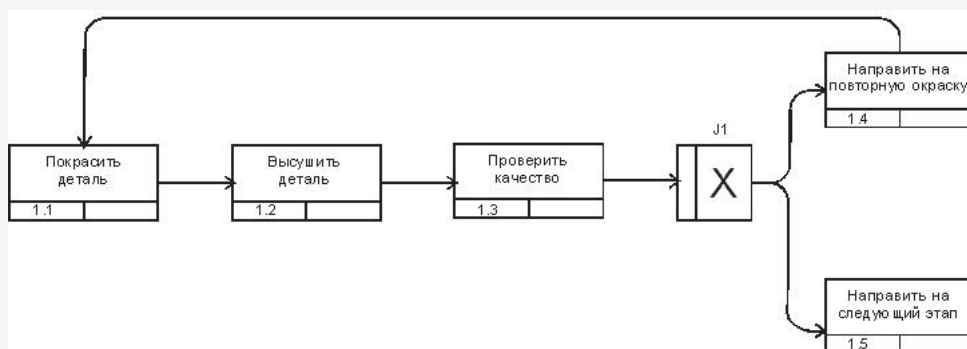


Рис. 26. Диаграмма протекания процесса в модели IDEF3

Действие J1 на рис. 26 означает, что в данной точке осуществляется обязательный выбор ровно одного из двух возможных направлений дальнейшего протекания процесса: будет выполняться либо действие 4, либо действие 5.

J1 представляет собой один из трех определяемых в методе IDEF3 видов соединений, или перекрестков (junction). Помимо показанного на рис. 26 разворачивающего соединения *Исключающее (Эксклюзивное) ИЛИ* в IDEF3 определены и другие типы, перечень которых приведен в табл. 4.

Таблица 4.

Соединения в IDEF3

Символ	Название	Вид	Смысл
&	И	Р	Запуск всех последующих действий.
		С	Завершение всех предшествующих действий.
X	Исключающее ИЛИ	Р	Запуск ровно одного последующего действия.
		С	Завершение ровно одного предшествующего действия.
O	ИЛИ	Р	Запуск не менее одного последующего действия.
		С	Завершение не менее одного предшествующего действия.

Буква Р в Табл. 4 означает *разворачивающее* соединение (Fan-out Junction), буква С- *сворачивающее* соединение (Fan-in Junction).

На рис. 27 и рис. 28 показаны примеры использования разворачивающих и сворачивающих соединений *ИЛИ* и *И*.

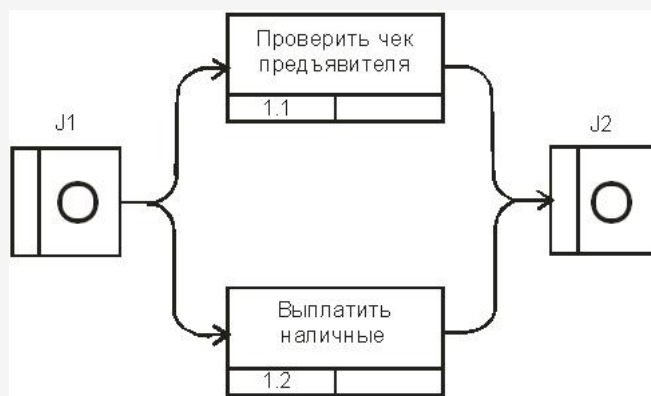


Рис. 27. Пример использования соединений типа ИЛИ



Рис. 28. Пример использования соединений типа И

В модели IDEF3 может также отражаться зависимость между моментами начала действий, выходящих из соединения, или окончания действий, входящих в соединения. В отличие от рассмотренных в примерах *асинхронных* соединений, где моменты начала или окончания могут быть отличаться друг от друга, в *синхронных* соединениях, обозначаемых на диаграммах прямоугольниками с двумя вертикальными линиями вместо одной, эти моменты должны совпадать.

Диаграммы *переходов состояний объектов* в IDEF3 отражают видение происходящего в изучаемой системе с точки зрения возможных изменений состояний объекта, который подвергается трансформации или обработке в результате процесса. Эти диаграммы состоят символов, означающих состояние объекта (окружности), символов перехода из одного состояния в другое (стрелки) и объектов ссылок (прямоугольники). *Состояния* определяются как значения существенного свойства (признака) объекта и условий. *Переходы* осуществляются в зависимости от выполнения предусловий или постусловий.

На рис. 29 показан пример диаграммы переходов (диаграмма процесса этого же примера показана на рис. 26).

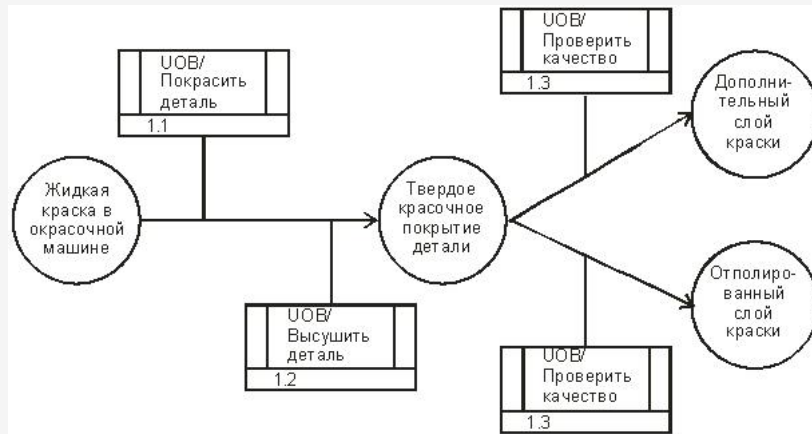


Рис. 29. Диаграмма состояний в модели IDEF3

Описание исследуемой системы, представленное средствами методики IDEF3, является достаточно подробным для построения на его основе моделей для количественного оценивания показателей системы, в частности, аналитических и имитационных моделей, которые будут рассмотрены далее.

Вопрос 3. Язык моделирования UML.

Язык UML (Unified Modelling Language - Единый Язык Моделирования) появился в результате развития методов объектно-ориентированного анализа и проектирования конца 1980-х – начала 1990-х годов. На этом языке основана, в частности, известная методология проектирования и разработки программного обеспечения фирмы IBM называемая RUP (Rational Unified Process).

Язык UML образуют множество символов и множество правил. В языке определены девять типов **диаграмм**, из которых для моделирования деловых процессов применяются семь.

Диаграммы классов (class diagram) служат для описания структуры моделируемой системы и содержат спецификации классов и отношений между классами. *Классами* могут быть информация, продукция, документы или организации. Пример диаграммы классов показан на рис. 30 (на концах связей поставлены обозначения кратности и указаны роли, которые каждый класс играет в данном отношении).

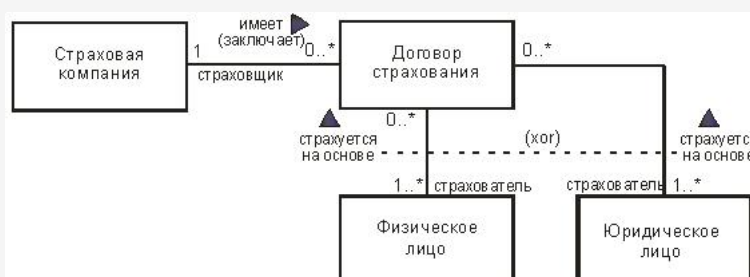


Рис. 30. Пример диаграммы классов

Классы связываются друг с другом с помощью *ассоциаций* (связей, отношений - association), которые могут представлять собой *агрегат* (aggregation - целое, составленное из частей), *композицию* (composition - специальный тип агрегата, в котором действуют ограничения на владение и время жизни), *обобщение* (generalisation) и *зависимость* (dependency). На диаграммах виды ассоциаций отображаются в виде различных окончаний у стрелок, связывающих классы (на Рис. 30 ассоциация обозначена треугольным маркером и дополнена именем, показывающим природу отношений между объектами).

Диаграммы объектов (object diagram) представляют множество объектов - экземпляров классов, присутствующих на диаграмме классов для специфических ситуаций (в некоторый момент времени).

Диаграммы состояний (statechart diagram) служат для описания основных этапы жизненного цикла объектов или систем путем отображения возможных их состояний и событий, которые могут перевести их в другое состояние (рис. 31).

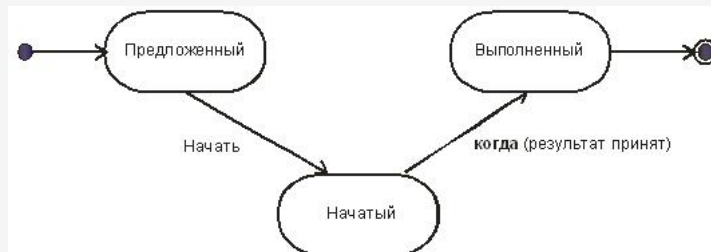


Рис. 31. Пример диаграммы состояний

Диаграммы деятельности, или активности (activity diagram) описывают действия и мероприятия, происходящие в системе. Их можно использовать для отображения последовательности выполнения отдельных этапов деловых процессов (рис. 32).

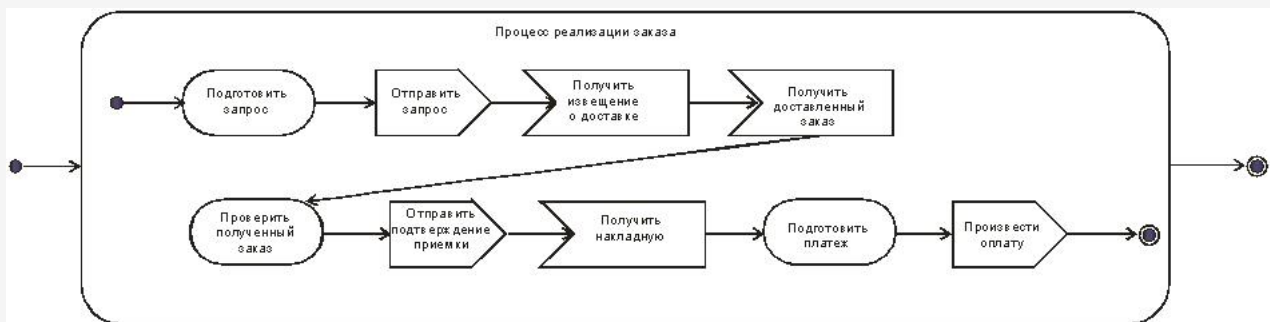


Рис. 32. Пример диаграммы деятельности

Диаграммы последовательностей действий (sequence diagram) описывают передачу сообщений между множеством объектов. На ней точно воспроизводятся логическая последовательность и хронология действий, сообщения могут быть как синхронными (выполняться полностью до начала каких-либо действий), так и асинхронными (отправитель не ждет ответа перед продолжением своих действий), снабжаться параметрами и указателями.

Диаграммы кооперации (collaboration diagram) схожи с диаграммами последовательности действий, но могут отображать более сложные правила взаимодействия между объектами. Информативность достигается за счет усложнения восприятия этих диаграмм.

Диаграммы вариантов использования, или прецедентов (use case diagram) применяются для описания функций системы. Прецедент характеризует конкретное использование возможностей системы *актером, или актантом* (actor), обозначающим пользователя каких-либо возможностей системы. Диаграмма отражает взаимосвязи между прецедентами, каждый из которых может включать какой-либо прецедент, расширять какой-либо прецедент или быть обобщением других прецедентов.

Диаграммы компонентов используются для описания компонентной структуры программных систем и для моделирования деловых процессов не применяются.

Диаграммы развертывания (deployment diagram) представляют собой диаграммы классов, описывающих отображение оборудование - программное обеспечение, и для моделирования деловых процессов не используются.

С тем, чтобы предоставить пользователям возможность адаптировать язык к их специфическим потребностям, в языке предусмотрены средства его расширения. В состав этих средств входят следующие.

Стереотипы (stereotypes), позволяющие разработчику добавлять новые конструктивные блоки на основе существующего набора.

Именованные значения (tagged values) содержат дополнительные сведения в виде пары тег-значение (tag-value), которыми можно снабжать любой элемент UML-модели.

Ограничения (constraints) отображают правила, применяемые в UML-модели, которые можно записать на языке описания ограничений OCL (Object Constraint Language), являющимся составной частью языка UML. При моделировании деловых процессов OCL применяется для описания деловых правил.

Спецификации языка содержат также «Расширения UML для моделирования деловых процессов», описывающие возможные расширения и правила их применения для моделирования деловых процессов.

Однако в силу того, что возможностей языка для моделирования деловых процессов недостаточно, некоторыми авторами предложены свои расширения UML. В качестве примера таких расширений можно указать метод *Эриксона-Пенкера* (Eriksson-

Penker). Язык OCL используется в этом методе для описания деловых правил. Деловой процесс представляется образцом на диаграмме деятельности (как деятельность со стереотипом `<<process>>`), процесс использует входные ресурсы (левая часть диаграммы) и формирует выходные ресурсы (правая часть диаграммы). Основными объектами в процессной модели являются объекты *цели* (goal), *входа* (input), *выхода* (output), *поставки* (supplying), *управления* (control). Обобщенная диаграмма процесса показана на рис. 32.

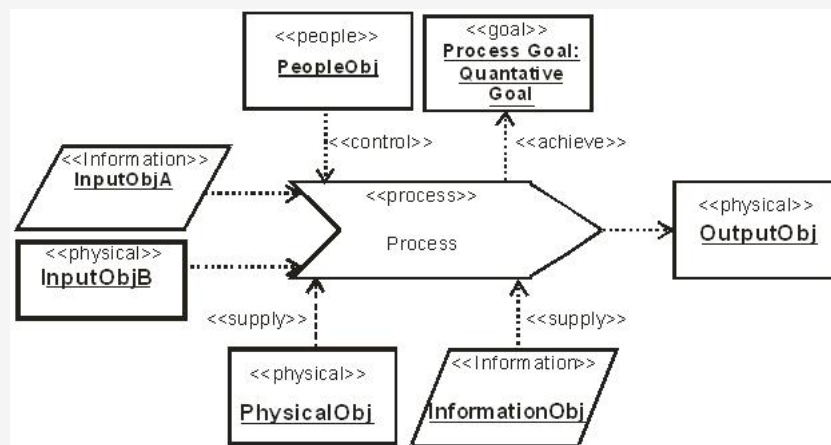


Рис. 33. Обобщенный вид диаграммы процесса

Нетрудно видеть, что диаграмма имеет много общего IDEF0-диаграммой, что свидетельствует о том, что, несмотря на различие изобразительных средств, обе методологии строятся на одинаковых концепциях.

Диаграмма деятельности используется в модифицированном виде под названием диаграмма *сборочной линии* (assembly line). Обычно эти диаграммы служат для представления информационных объектов информационной системы и показывают, как осуществляется доступ к информации и как она используется в процессах, которые показываются в их верхней части диаграммы. Обобщенный вид диаграммы показан на рис. 34, R-read (чтение), W-write (запись).

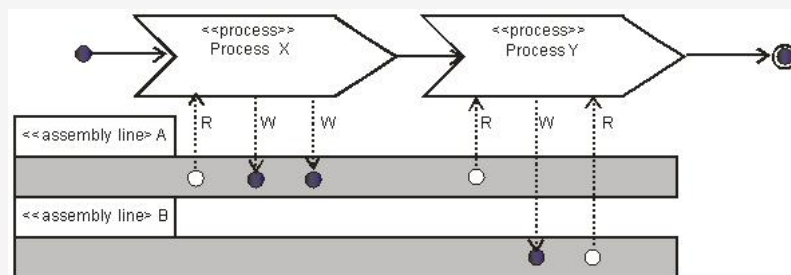


Рис. 34. Обобщенный вид диаграммы сборочной линии

Диаграммы сборочной линии связаны с диаграммами вариантов использования, так как отображают интерфейс между деловыми процессами и информационной системой.

Для обеспечения возможности повторного использования разработанной модели для описания схожих процессов можно применять механизм *шаблонов*, или *паттернов* (pattern), которые классифицируются по их применению. Для целей моделирования деловых процессов также предложен ряд шаблонов.

Вопрос 4. Автоматизация моделирования деловых процессов.

К настоящему времени разработано довольно много программных продуктов, поддерживающих процессы разработки моделей по различным методикам, в частности, IDEF и UML. Степень автоматизации варьируется в широком диапазоне.

Средства нижнего уровня, например, MS Office Visio предоставляют средства графического интерфейса.

Программы промежуточного уровня, например, CA AllFusion Business Process Modeler - BPwin помимо изобразительных средств позволяют проводить проверку корректности построенных моделей.

Средства верхнего уровня (*CASE-системы*) автоматизируют основные этапы системного анализа с возможностью генерации схем баз данных, программного кода на различных языках и выпуска проектной документации. В качестве примеров таких систем можно назвать системы ARIS и IBM Rational Rose.

Выбор конкретного средства зависит, в первую очередь, от масштаба проектируемой системы (предприятия), а также от состава группы аналитиков (консультантов) – увеличение численных характеристик будет смещать предпочтения при выборе от наиболее простых систем автоматизации в сторону более крупных.

Пакеты распространяются на коммерческой основе (как указанные выше), свободно распространяются в виде дистрибутивных файлов (Design/IDEF) или доступны для бесплатного применения через Веб-интерфейс (IDEF0 on WEB for free using <http://e-a-m.ru/rsvIDEF0net.aspx>, <http://e-a-m.ru/Docs/Help.htm>). Есть продукты (например, EAM IDEF0/Doctor) снабженные исходными текстами, что позволяет проводить их функциональную адаптацию к потребностям конкретных пользователей.

В Приложении к курсу содержатся краткие сведения относительно функциональных возможностей, правил и порядка работы с пакетом Design/IDEF, которыми можно воспользоваться для выполнения практического задания по настоящей теме.

Выводы:

1. Эффект внедрения информационной системы определяется в первую очередь степенью ее интеграции с деловой средой. Средством повысить вероятность успеха внедрения, выявить архитектуру будущей системы и сформулировать перечень требований к ней является модель деловых процессов, содержащая их максимально точное описание.

2. Создание и применение деловых моделей может вестись с использованием различных методологий и стандартов. Одними из наиболее широко распространенных подходов к моделированию являются методики семейства IDEF и язык UML. Преимуществами первой методологии являются простота применения и понимания конечными пользователями, вторая методология, базируясь на объектно-ориентированном подходе, позволяет автоматизировать преобразование созданной деловой модели в программный код.

3. Большинство известных подходов к моделированию деловых процессов строятся на базовых принципах и методиках общей теории систем и системного анализа, в частности, таких как иерархичность и декомпозиция. Несмотря на различия в средствах описания, методологии (в частности, IDEF и UML) не имеют принципиальных отличий и которые позволяют представить в достаточном объеме существенные сведения о моделируемых деловых процессах.

Вопросы для самопроверки:

1. Что называется деловым процессом?
2. Что понимается под целью процесса?
3. Что такое ресурс процесса?
4. Какие выделяются виды ресурсов?
5. Что такое деловое правило?
6. Как классифицируются деловые правила?
7. В чем состоят отличия между улучшением, инновацией и реинжинирингом?
8. Что понимается под архитектурой информационной системы?
9. Какие выгоды можно получить от создания деловой модели?
10. Какие требования к программному обеспечению информационной системы дает возможность сформулировать деловая модель?
11. Что представляет собой методология IDEF?
12. Какие методы методологии IDEF в основном используются для моделирования деловых процессов?
13. Какие требования и ограничения необходимо определить до начала разработки IDEF0-модели?
14. Как можно классифицировать IDEF0-модели по их назначению??
15. Из каких конструктивных единиц состоит IDEF0-модели?
16. Какие объекты определены в IDEF0-модели? Дайте краткую характеристику каждому типу.
17. В чем состоит сущность приема декомпозиции IDEF0-диаграммы?
18. Какие термины используются для отображения иерархии IDEF0-диаграмм?
19. Что такое туннель и для чего он используется?
20. Что не позволяют и что не позволяют отобразить IDEF0-модели?
21. Какую информацию можно представить с помощью IDEF3-модели?
22. Какие виды изобразительных средств могут применяться в IDEF3-модели?
23. Что понимается под терминами *сценарий* и *действие* в IDEF3-модели?
24. Какие типы связей между действиями могут задаваться в IDEF3-модели?
25. Что понимается под *соединением* в IDEF3-модели, какие существуют виды соединений?
26. Что отражает диаграмма переходов состояний объектов в IDEF3-модели?
27. Что представляет собой язык UML? Какие множества элементов образуют выразительные средства языка?
28. Какие диаграммы определены в языке UML?
29. Для чего нужны расширения языка UML?
30. Какие средства расширения существуют в языке UML?
31. Для чего применяется метод Эриксона-Пенкера?
32. Как представляется деловой процесс в методе Эриксона-Пенкера?
33. Какие сведения даются с помощью диаграммы сборочной линии в методе Эриксона-Пенкера?
34. Дайте сравнительную характеристику подходам к построению деловых моделей на основе методологий IDEF и UML.

Литература по теме:

Основная литература:

1. Д. Марка, К. Макгоуэн. SADT™. Методология структурного анализа и проектирования. – М.:МетаТехнология, 1993 - 240 с.
2. С.В.Черемных, И.О.Семенов, В.С.Ручкин Моделирование и анализ систем. IDEF-технологии: практикум, М., Финансы и статистика, 2005.-192с.
3. Г. Буч, Д. Рамбо, А. Джекобсон. Язык UML. Руководство пользователя - М.: ДМК Пресс, 2003. - 432 с.

Дополнительная литература:

1. Hans-Erik Eriksson. Business Modeling with UML: Business Patterns at Work/ Hans-Erik Eriksson, - Magnus Penker John Wiley & Sons, 2000 -459 p.
2. Noran O. Business modeling: UML vs. IDEF Griffith University, 2002.

Практические задания.

Задание 1.

Установите на компьютере (скачав из интернета) один из свободно распространяемых пакетов моделирования деловых процессов по методологии IDEF (Design/IDEF, IDEF/Doctor). Постройте в среде установленного пакета IDEF0-модель, диаграммы которой приведены на рис. 24. Для практической работы с пакетом Design/IDEF используйте приведенные в Приложении описание его команд и рекомендации по применению.

Задание 2.

Разработайте IDEF0-модель какого-либо известного вам процесса. В качестве таковых могут быть, к примеру:

- Строительство дачного дома.
- Подготовка студенческой конференции.
- Прохождение технического обслуживания автомобиля.

Если вам хорошо знакома среда, где вы работаете или работали, постройте модель известного вам делового процесса своей компании (или подразделения).

Тема 6. Моделирование потоков работ

Цели изучения темы:

- познакомиться с концепцией потока работ и подходами к анализу и проектированию систем, основанных на концепции потока работ.

Задачи изучения темы:

- познакомиться с концепцией потока работ;
- познакомиться с наиболее известными методологиями моделирования потока работ;
- познакомиться с моделями, позволяющими проводить анализ и оптимизацию потоков работ.

Успешно изучив тему, Вы:

получите представление о:

- области применения подхода, основанного на концепции потока работ;
- преимуществах подхода к построению систем, основанного на концепции потока работ;
- современных достижениях в области методического обеспечения создания систем, основанных на концепции потока работ;

будете знать:

- основные понятия теории сетей Петри;
- возможности сетей Петри для моделирования информационных процессов;
- назначение и сущность метода критического пути и метода;
- назначение и сущность метода PERT.

Вопросы темы:

1. Концепция потоков работ.
2. Моделирование на основе сетей Петри.
3. Высокоуровневые сети Петри.
4. Модель сетевого планирования.

Вопрос 1. Концепция потоков работ.

В процессе стремительного развития и применения достижений информационных технологий, в первую очередь, в таких областях как электронная коммерция, возникла необходимость построения новой концепции описания деловых процессов. Одной из таких концепций, получивших широкое распространение, стала концепция **потока работ**.

Например, чтобы обработать пользовательский запрос к системе бронирования авиабилетов через Интернет, системой реализуется довольно сложная процедура, в процессе которой (согласно указанным в запросе параметрам - пункты и даты вылета и прилета, класс билета и их число) производятся обращения на сайты разных авиакомпаний. На основании полученных сведений готовятся варианты с маршрутами, датами и временем вылета и прилета и стоимости билета, которые предлагаются пользователю. Процесс может еще более усложниться, если в запросе предусмотрено бронирование мест в гостинице или аренда автомобиля. Все выполняемые действия рассматриваются как отдельные работы, состав и логика выполнения которых описываются моделью потока работ.

Основным органом, координирующим усилия по развитию работ в области моделирования потока работ, выступает основанная в 1993 году международная организация разработчиков, пользователей, консультантов и аналитиков Workflow Management Coalition (WfMC, <http://www.wfmc.org/>). В соответствии с определением WfMC **поток работ** (workflow, WF) есть *полная или частичная автоматизация делового процесса, когда документы, информация или задачи передаются от одного участника другому для проведения с ними действий согласно установленному перечню правил*.

Система управления потоками работ (Workflow Management System - WFMS) представляет собой *систему, которая определяет, создаёт и управляет выполнением потоков работ посредством программного обеспечения, работающего на одном или более механизмах исполнения потока работ (workflow engines) и которая в состоянии интерпретировать определения процессов, взаимодействовать с участниками потока работ и в нужных случаях прибегать к средствам ИТ и приложений*. (WfMC).

Система управления потоками работ назначает каждой работе исполнителей, контролирует прохождение работы и отслеживает степень её выполнения.

Автоматизированные системы управления потоками работ **позволяют:**

- избежать потерь или «зависаний» работ, устраняя тем самым необходимость в участии диспетчеров или контролёров;
- предоставить управленческому персоналу возможность уделить большее внимание задачам повышения производительности труда отдельных исполнителей и оптимизации деловых процедур, освободив от рутинных процедур распределения задач по исполнителям и проверки исполнения;

- обеспечить выполнение процедур в строгом соответствии с установленной последовательностью и обязательным документированием результатов, что гарантирует точное следование деловому плану и соблюдение требований;
- подобрать наиболее подходящего исполнителя (сотрудника или устройство) для выполнения каждой работы с учётом относительной важности отдельных работ и прецедентов;
- достичь по сравнению с традиционным подходом максимального распараллеливания (одновременного или попеременного) выполнения работ.

В настоящее время среда WF встроена, в частности, в ряд продуктов Microsoft, таких Visual Studio, SharePoint PortalServer, Microsoft Outlook и Microsoft Exchange Server.

Модель потока работ (workflow model) представляет собой *модель повторяющихся последовательностей действий или операций реальной системы с целью их оценивания, анализа, реорганизации или преобразования*. Применение таких моделей может помочь определить наиболее рациональный способ организации производственных процессов, процессов обслуживания или обработки информации. При проектировании программного обеспечения информационных систем, модели потока работ используются для построения человеко-машинного интерфейса.

В типичном случае поток работ состоит из совокупности выполняющихся в логической последовательности единиц работ, называемых **действиями**, или **активностями** (activity, или task). Действие может представлять собой ручную операцию, взаимодействие с пользователем или участником потока работ, а также операцию, выполняемую автоматически. Действия растянуты во времени, они могут перемежаться и взаимодействовать между собой. Отдельные экземпляры действий образуют **список работ** (worklist).

То, что вызывает действия или является их объектом, называется **прецедент**, или **ситуация** (case, или workflow instance). Например, в потоке работ системы управлением обработки заказов прецедент создается каждый раз, когда поступает очередной заказ. Прецедент может представлять собой как имеющий конкретное воплощение объект, так и некоторую абстракцию (например, строительный проект или заявление на выплату страхового возмещения).

В системах управления потоками работ используется множество различных языков и концепций. В большинстве систем применяется свой собственный язык. Наряду с ними в последнее время все большее распространение получает подход на основе аппарата сетей Петри, который будет рассмотрен далее.

В информационных системах, ориентированных на процессы, для спецификации потоков выделяются несколько **перспектив** ([perspective](#)). Согласно классификации образованной в 1999 году Workflow Patterns initiative (<http://www.workflowpatterns.com/patterns/>), ставящей своей задачей создание концептуальной основы для процессной технологии, в качестве перспектив выделяются:

- **Перспектива потока управления** ([control-flow perspective](#)). Фокусируется на зависимостях по управлению между различными задачами, например, параллельное выполнение работ, выбор из нескольких работ, синхронизация работ.
- **Перспектива данных** ([data perspective](#)). Имеет дело с передачей информации, областью видимости переменных и т.п.
- **Перспектива ресурсов** ([resource perspective](#)). Рассматриваются назначения ресурсов работам, их передача другим работам и т.п.
- **Перспектива обработки исключительных ситуаций** ([exception handling perspective](#)). Отображаются причины отклонений от нормального хода выполнения и действия по их устранению.

Для каждой из перспектив создаются **шаблоны** (pattern). Всего на сегодняшний день создано около 100 шаблонов (для перспективы потока управления создано более 40 шаблонов) разной степени сложности. На рис. 35 дано визуальное представление двух шаблонов (имеющих на сайте номера 1 и 2): Последовательность (Sequence) и Распараллеливание (Parallel Split).

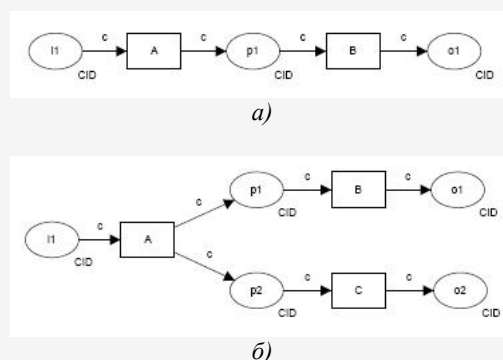


Рис. 35. Шаблоны Последовательность (а) и Распараллеливание (б)

Последовательность (а) является базовым блоком для организации процессов. В этом случае работы выполняются последовательно одна за одной.

Распараллеливание (б) означает что после завершения работы А инициируются два процессных потока так, что работы В и С могут выполняться одновременно.

Шаблоны потоков работ и теория сетей Петри положены в основу языка YAWL (Yet Another Workflow Language – еще один язык потока работ), который был создан сотрудником Технологического университета г.Эйндховена (Голландия) Вилем ван дер Альстом (Wil van der Aalst) и сотрудником Квинслендского университета г.Брисбена (Австралия) Артуром Хофстеде (Arthur ter Hofstede) в 2002 году. Для языка разработана и поддерживается программная среда, которая свободно распространяется через Интернет (<http://sourceforge.net/projects/yawl/>) и состоит из нескольких компонентов для различных платформ. Для ознакомительных целей и изучения предназначен компонент YAWL4Study, который просто устанавливается и легко осваивается. На Рис. 36 показан вид интерфейса с редактором YAWLEditor, где показано описание потока работ для примера, который приводился в начале данного вопроса.

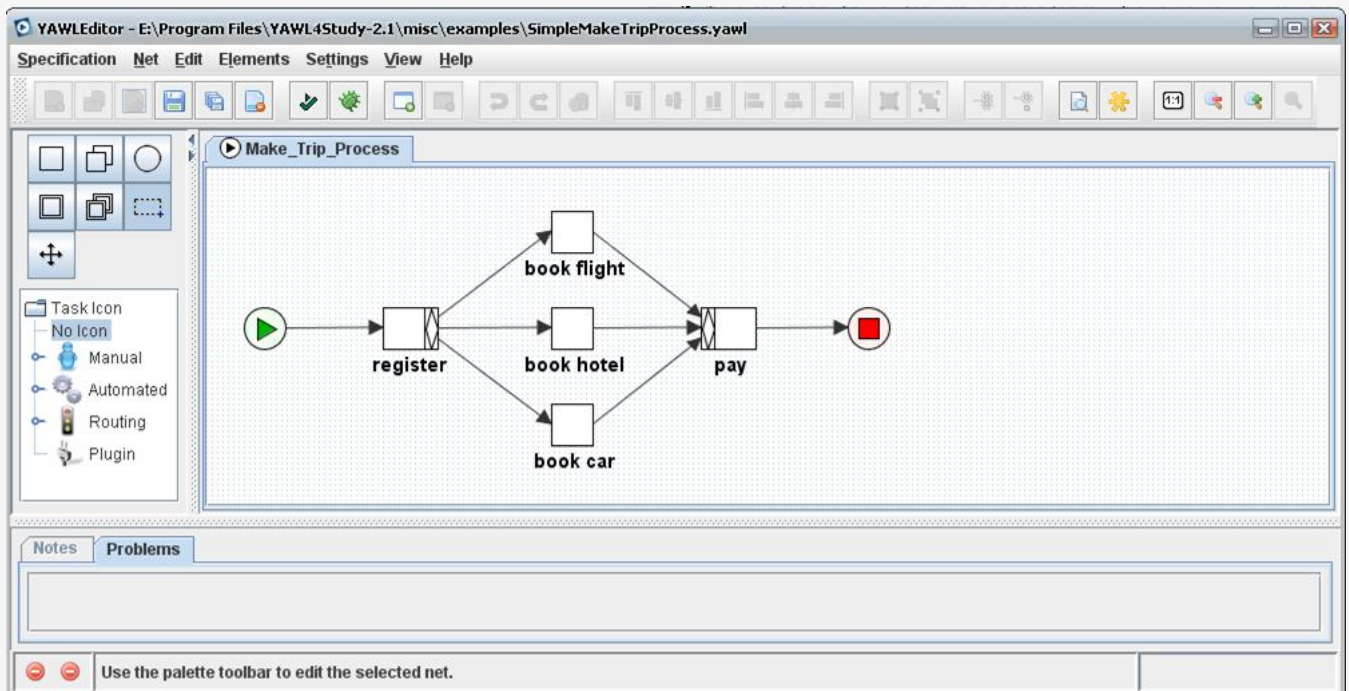


Рис. 36. Пример YAWLEditor

Символы на рисунке имеют следующий смысл:

	начало работ
	завершение работ
	действие разделяется на несколько (но не обязательно на все сразу) потоков (OR-Split)
	действие ожидает, пока не завершатся (или никогда не завершатся) все входящие потоки (OR-Join).
	действие

Вопрос 2. Моделирование на основе сетей Петри.

Большими возможностями с точки зрения своего использования для моделирования деловых процессов и, в особенности, для моделирования потока работ обладают сеть Петри. Впервые теория сети Петри была предложена в 1962 году немецким исследователем Карлом Адамом Петри для моделирования и анализа систем с параллельно протекающими взаимодействующими процессами. Оригинальное изложение теории основано на применении строгого формального аппарата, что является важной отличительной стороной этой теории и выделяют её из других подходов к моделированию задач указанного класса.

Несмотря на то, что математическая строгость является сильной стороной теории, более наглядной для объяснения её сути и более удобной для практического применения является её графическая интерпретация. В ней используется два основных понятия, с которыми связано представление процессов в системе: событие и условие. **Событие** есть некоторое наблюдаемое в системе действие, возникновение которого определяется множеством **условий**, представляющих собой логические высказывания (предикаты). Для наступления каждого из событий необходимо, чтобы было выполнено одно или несколько условий, называемых **предусловиями**. Факт наступления события может привести к тому, что будут выполнены другие условия, называемые **послеусловиями**. Эти понятия реализованы элементами **двудольного ориентированного графа**, которым может быть представлена классическая сеть Петри.

Напомним, что двудольным называется граф, в котором все множество вершин можно представить двумя непересекающимися подмножествами так, что вершины, соединенные дугой графа, всегда принадлежат разным подмножествам (см. рис. 37):

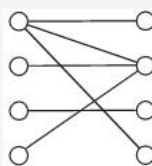


Рис. 37. Двудольный граф

Вершины первого множества (будем их обозначать p) называются **местами, или позициями (places)** и изображаются графически в виде окружностей.

Вершины второго множества (будем их обозначать t) называются **переходами (transitions)** и изображаются графически либо в виде толстых линий, либо в виде прямоугольников (будем далее использовать это обозначение).

Вершины соединяются посредством **направленных дуг**. Соединяться могут вершины только разных типов.

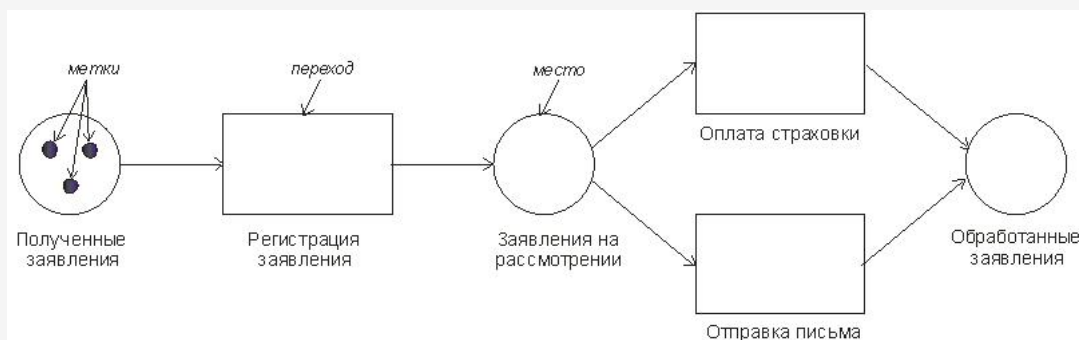


Рис. 38. Пример сети Петри

Например, на рис. 38 показана модель процесса обработки в страховой компании заявлений на выплату страховой суммы после наступления страхового случая. В этом графе присутствуют три места - *Полученные заявления*, *Заявления на рассмотрении*, *Обработанные заявления* и три перехода - *Регистрация заявления*, *Оплата страховки*, *Отправка письма*.

Места в графе подразделяются на *входные* и *выходные*.

Место p называется **входным** для перехода t тогда и только тогда, когда существует дуга, направленная из t в p .

Место p называется **выходным** для перехода t тогда и только тогда, когда существует дуга, направленная из p в t .

Места могут содержать **метки**, или **фишки**, или **маркеры** (tokens), которые изображаются черными кружками и управляют переходами сети.

Запускаться переход может только при условии, что каждое из входных мест содержит число меток, называемых **разрешающими метками**, не меньшее, чем число исходящих из места дуг. Переход в этом случае называется **разрешенным**. Так, на рис. 38 разрешенным является переход *Регистрация заявления*, переходы *Оплата страховки*, *Отправка письма* запрещены.

Сеть Петри представляет собой в общем случае **мультиграф**. На рис. 39 показан пример сети, содержащей более одной дуги, которые соединяют входное место с переходом:

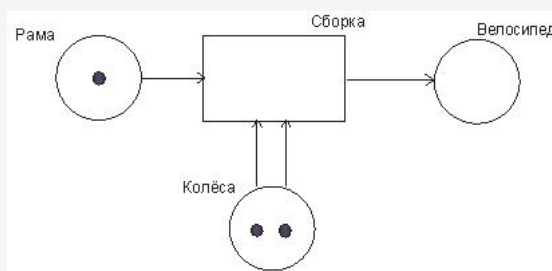


Рис. 39. Сеть Петри с множественными дугами

Граф на рис. 39 отражает тот факт, что для сборки велосипеда необходимо иметь раму и два колеса, на что указывает пара стрелок, идущих из входного места *Колёса* в переход *Сборка*. Для того, чтобы переход *Сборка* был разрешен, необходимо, чтобы входное место *Колёса* содержало две метки. Когда переход запускается, число используемых переходом меток определяется числом стрелок, входящих в этот переход.

В каждый отдельный момент времени сеть однозначно характеризуется своим **состоянием**, называемым также **маркировкой** (marking), которое может быть задано перечислением числа меток, находящихся в каждом из мест сети. Например, состояние сети на рис. 38 можно записать в виде вектора $(3,0,0)$, или выражения $[3p_1 + 0p_2 + 0p_3]$, означающих, что место *Полученные заявления* (p_1) содержит три метки, место *Рассматриваемые заявления* (p_2) содержит ноль меток и место *Обработанные заявления* (p_3) содержит ноль меток.

Переход **запускается**, или **срабатывает** (fire) в результате удаления меток из входных мест и внесения меток в выходные места. Например, результатом срабатывания перехода *Регистрация заявления* на рис 38 будет граф, показанный на рис. 40:

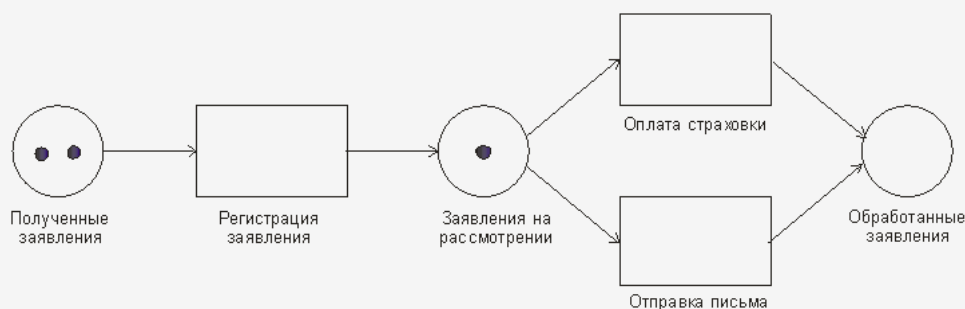


Рис. 40. Результат срабатывания перехода сети Петри рис. 38

При моделировании деловые процессы отображаются на отдельные элементы сети Петри.

К **активным** элементам сети Петри относятся *переходы*, срабатывание которых переводит сеть из одного состояния в другое. Переходы могут представлять отдельные операции делового процесса, такие как преобразования или транспортировка

каких-либо объектов.

К **пассивным** элементам сети Петри относятся *места*, которые не меняют состояние сети. Места могут представлять буферные накопители, среду хранения, физическое местонахождение, состояние обработки какого-либо объекта или условие, в котором он находится.

Метки сети Петри представляют собой некоторые объекты, которые в зависимости от сущности моделируемой системы могут быть как физической, так и информационной природы.

В **расширенной** сети Петри можно отразить наличие определённых **ограничений** на протекание процессов (**деловых правил**). Например, если вместо модели рис. 39, которая допускает возможность одновременной сборки нескольких велосипедов, необходима модель, в которой будет отражено ресурсное ограничение, допускающее в каждый момент сборки не более одного велосипеда, то граф следует преобразовать в граф рис. 41.

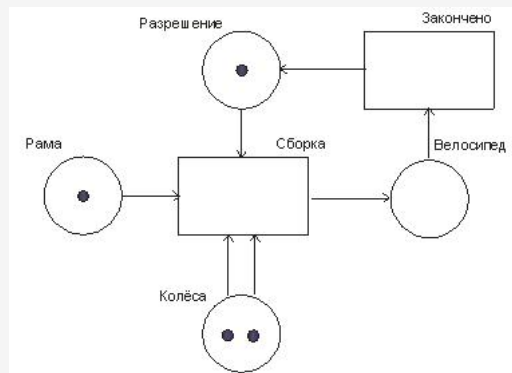


Рис. 41. Сеть Петри с ограничениями

В преобразованном варианте в момент, когда очередной комплект компонентов поступит на сборочный участок, место *Разрешение*, запретит сборку очередного велосипеда. Эта возможность появится по окончании процесса сборки текущего экземпляра и появления метки в месте *Разрешено* после срабатывания перехода *Закончено*.

Вопрос 3. Высокоуровневые сети Петри.

Практическое использование теории сетей Петри довольно скоро привело к выводу об ограниченности её возможностей применительно к моделированию реальных систем. Этот вывод привел к созданию ряда расширений теории, ориентированных на решение этой задачи. Наибольший интерес для моделирования деловых процессов представляют раскрашенные сети Петри, временные сети Петри и высокоуровневые сети Петри, которые называются **высокоуровневыми**.

Раскрашенные сети Петри позволяют преодолеть ограничения обычной сети Петри, вызываемые неразличимостью меток. Так, в рассмотренной выше модели процесса обработки заявлений на выплату страховой суммы (Рис. 38) невозможно отобразить такую, например, характеристику, как величина возмещаемой суммы, что может быть существенным для применения процедурных правил работы с заявлением. Возможность наделить метки набором значений определённых признаков обеспечивается приданием метке «цвета», в котором могут храниться все нужные для применения деловых правил значения. Запуск перехода осуществляется на основе проверки не только наличия меток во входных местах, но и на основе учета **значений** входных меток. Иначе говоря, предусловие может быть сформулировано более детально.

Кроме того, в раскрашенных сетях Петри от значений входных меток зависят как **число** получаемых выходных меток, так и **значения**, которые им должны быть присвоены.

Временные сети Петри реализует возможность осуществить привязку присутствующих в них событий к временному параметру с помощью **временной отметки**, или **веса** (timestamp), которые получает метка в дополнение к своим значениям. Например, значение временной отметки равное 12 будет означать, что метка будет доступна для поглощения в момент времени 12.

Переход считается разрешенным в случае, если все метки во входных местах имеют в качестве значений своих временных отметок значения, не превышающие текущего времени. Метки изымаются в соответствии с дисциплиной «первый пришел - первый обслужен», т.е., первой изымается метка с минимальным значением временной отметки. При срабатывании перехода выходные метки получают значения временных отметок не меньшие, чем момент времени срабатывания перехода, увеличенный на **время задержки**. Величина задержки в общем случае зависит от значений изымаемых входных меток, но может быть и величиной постоянной или случайной.

На рис. 42 представлено описание с помощью сети Петри работы отдела технического обслуживания компьютерного класса, состоящего из n компьютеров (вместо меток в месте *Компьютерный класс* стоит обозначение n), каждый из которых может выходить из строя.

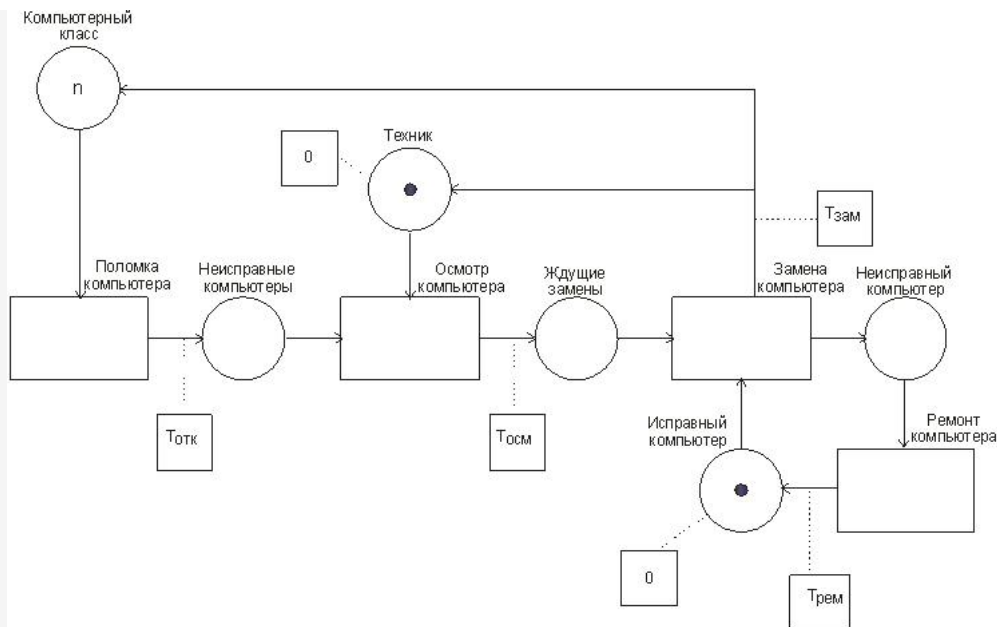


Рис. 42. Временная сеть Петри

Сопровождение осуществляет один техник, который после поступления заявки пользователей проводит осмотр и проверку отказавшего компьютера, заменяет его исправным из резерва и отправляет отказавший в ремонт. Отремонтированный компьютер возвращается в резерв, на котором в начале процесса есть один исправный компьютер.

Как видно из графа, переход *Поломка компьютера* будет разрешен, во входном месте *Компьютерный класс* есть метки. Чтобы отразить тот факт, что компьютеры выходят из строя по прошествии некоторого времени, возле перехода на графе поставлена временная отметка *Тотк*, представляющая собой значение среднего времени между двумя последовательными поломками (наработка на отказ). Переход *Диагностика компьютера*, возможный при наличии меток в местах *Неисправные компьютеры* и *Техник свободен*, будет запущен с временной задержкой *Тосм*, равной среднему времени осмотра и проверки. Аналогичным образом, переход *Ремонт компьютера*, возможный при наличии метки в мест *Исправный компьютер* будет запускаться с задержкой на время ремонта *Трем*, переход *Замена компьютера* будет запускаться с задержкой *Тзам*. Метки *Техник* и *Исправный компьютер* доступны для использования в момент начала моделируемого процесса.

Раскрашенные и временные сети Петри позволяют создавать модели довольно сложных процессов, но они не позволяют выявить их **структуру**. Такую возможность предоставляют **иерархические сети Петри**. Для представления процессов в укрупненном виде в сеть Петри добавляется элемент, называемый процесс, который обозначается прямоугольником с двойной граничной линией. Например, если мы ходим детализировать переход *Ремонт компьютера* на Рис. 42, то модель может быть преобразована в вид, показанный на рис. 43.

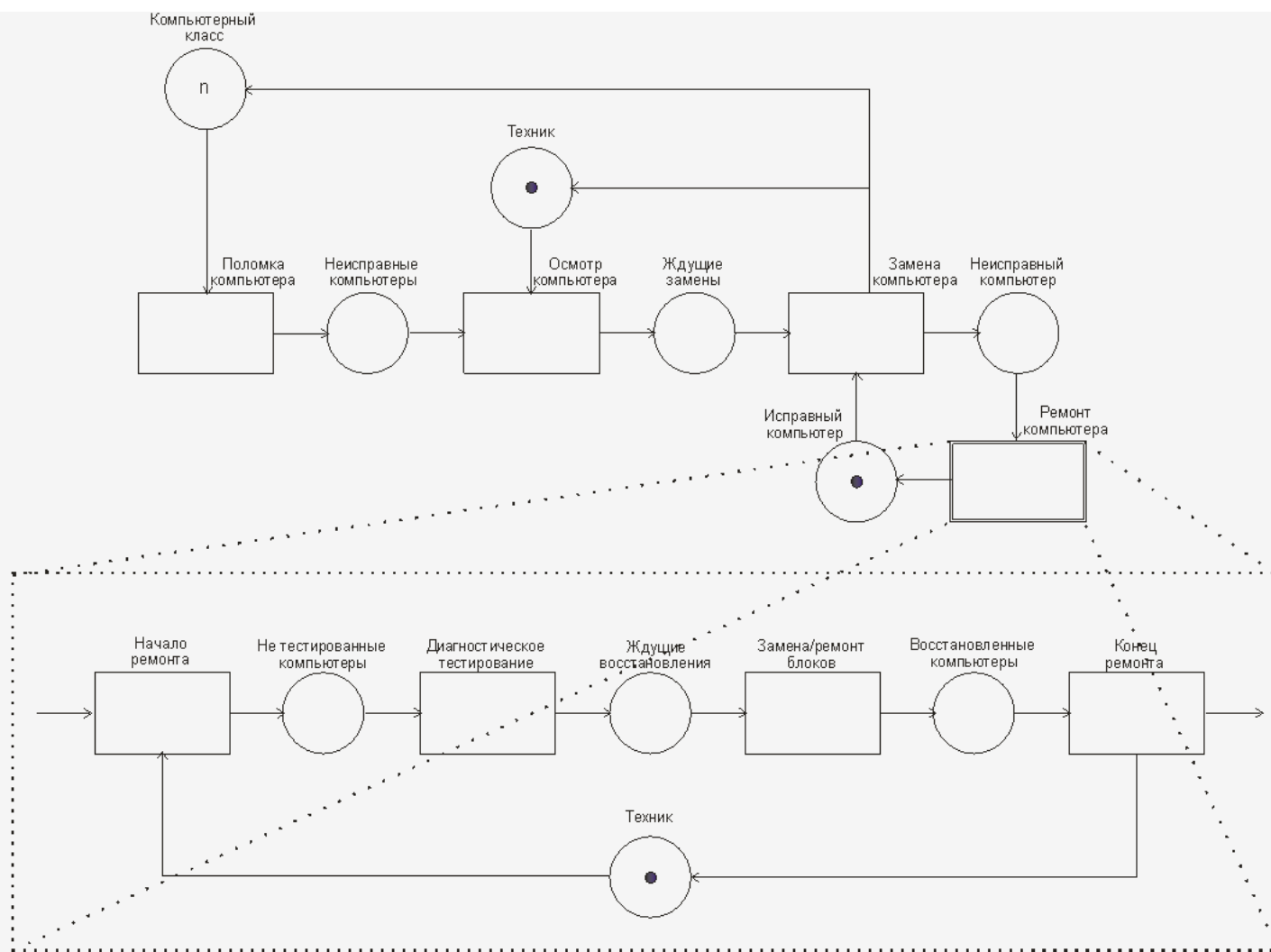


Рис. 43. Иерархическая сеть Петри

Представленный как отдельный **подпроцесс** переход *Ремонт компьютера* включает теперь этапы *Начало ремонта*, *Диагностическое тестирование*, *Замена/ремонт блоков*, *Конец ремонта*. При этом число компьютеров, которые могут ремонтироваться одновременно, ограничено одним.

Концепция иерархических сетей Петри реализует один из основополагающих принципов системного подхода: систематически применяя принцип декомпозиции процесса на подпроцессы можно значительно **упростить анализ** в сложных случаях. Кроме того, появляется возможность использовать модели отдельных процессов **многократно** - в случаях, когда он как подпроцесс встречается в нескольких местах создаваемой модели.

Однако непосредственное применение теории сетей Петри для моделирования потока работ сталкивается с рядом **проблем**, к которым относятся следующие.

- Раскрашенные сети Петри дают возможность представить несколько реализаций подпроцессов. Однако, средств моделирования их взаимодействия таких, как трассировка, ветвление и слияние реализаций не предусмотрено.
- В процессе анализа часто возникает необходимость слияния потоков, относительно которых неизвестно, какой вид синхронизации необходим. Если активными являются оба потока, требуется AND-соединение, в противном случае XOR-соединение. Моделировать предварительную синхронизацию для таких ситуаций в модели сети Петри довольно трудно, поскольку правило запуска требует задавать определённый вид соединения (AND-соединение для переходов и XOR-соединение для мест).
- Запуски переходов разрешаются своими входными местами и изменяют только свои выходные места. Однако эффект наступления некоторых событий, связанных с потоком работ, может выходить за рамки локального. Так, может возникнуть необходимость удалить ошибочные метки из всех мест, где они в данный момент находятся, или реализовать действия, которые требуется принять по истечении контрольного срока.

Поэтому для представления моделей в системах управления потоками работ используют сети Петри в адаптированном виде.

Вопрос 4. Модель сетевого планирования.

Сеть Петри может быть легко преобразована в сетевую модель, которая вместе со связанным с нею методом критического пути появилась в конце пятидесятых годов прошлого века, и находит широкое применение для задач планирования. С ее помощью могут решаться как задачи проектирования информационных систем, так и задачи оптимальной организации потока работ в процессе их моделирования.

Сетевая модель (см. пример рис. 44) базируется на теории ориентированных графов.

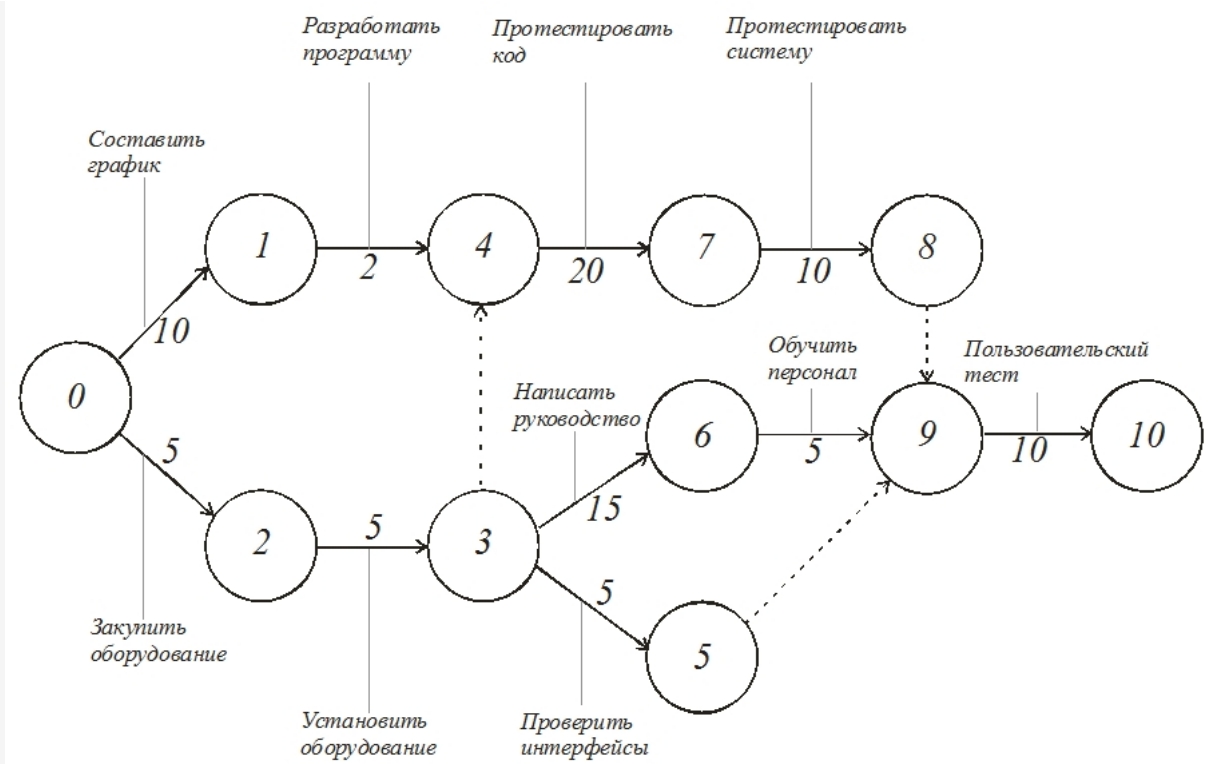


Рис. 44. Пример сетевой модели

Основными элементами модели являются *событие, работа и путь*.

Событие (nod) - факт завершения всех предшествующих работ и готовности к выполнению всех последующих. Исходное событие (не имеющее входящих дуг) сети называется **начальным**, или **исток**ом и обычно обозначается 0, завершающее событие (не имеющее исходящих дуг) называется **конечным** или **сток**ом.

Работа (task) есть активность, связанная с расходом ресурсов. Работа определяется своими начальным (i) и конечным (j) событиями. Основным ресурсом, связанным с работами является время ее выполнения, по причине чего на графе проставляются пометки со значениями длительности выполнения работы, выраженной в условных единицах (см. рис. 44).

Работы, показанные на графе рис. 44 пунктиром, называются *фигтивными* – они имеют нулевую длительность (пометка 0 обычно не ставится), но необходимы для отражения логической последовательности работ. Например, присутствие фиктивной работы (3, 4) означает, что работа (4, 7) может начаться после завершения не только работы (1, 4), как это было бы в случае отсутствия работы (3, 4), но и работы (2, 3).

Все события (вершины графа) нумеруются так, чтобы соблюдалось следующее правило: номер начального события каждой работы должен быть меньше номера ее конечного события.

Путь (path) – последовательность работ в сети, в которой конечное событие любой работы совпадает с начальным событием следующей за ней работы. Путь, имеющий наибольшую продолжительность, называется **критическим** (critical path) $L_{кр}$, его длительность обозначается $T_{кр}$. Время выполнения процесса (проекта) в целом не может быть меньше $T_{кр}$, поэтому первая задача при анализе сетевой модели состоит в нахождении $L_{кр}$ и критических работ и поиск возможностей по сокращению их длительности.

На этом базируется и в этом состоит сущность **метода критического пути** (Critical Path Method - CPM). Дадим описание этого метода на примере модели, показанной на рис. 45, ограничившись решением упрощенной задачи нахождения только величины критического пути.

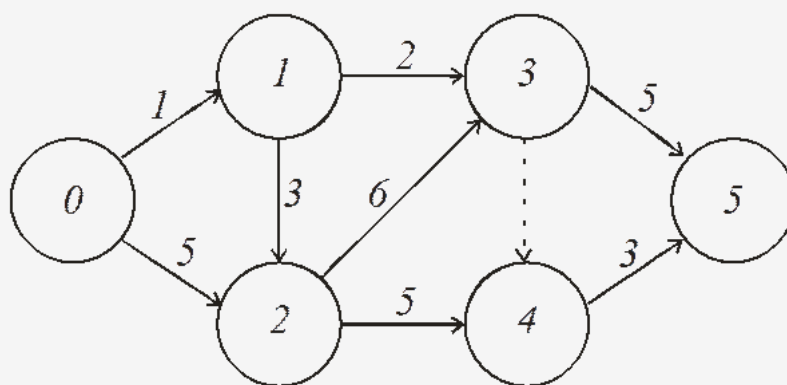


Рис. 45. Пример нахождения критического пути

Для нахождения критического пути будем последовательно определять наиболее ранние моменты наступления событий $t(i)$, начиная с начального ($i=0$).

Событие 0. Естественно считать, что событие 0 наступает в момент времени равный 0:

$$t(0) = 0$$

Событие 1. Поскольку в событие 1 входит только одна работа (0, 1), то событие 1 наступит тогда, когда завершится работа (0, 1). Так как она началась в нулевой момент, то время ее завершения равно ее продолжительности, т.е.,

$$t(1) = 1$$

Событие 2. Событие 2 наступит после завершения работ (0, 2) и (1, 2). Первая из них начинается в момент $t(0) = 0$ и длится 5 единиц времени, вторая – в момент $t(1) = 1$ и длится 3 единицы времени. Таким образом, работа (0, 2) закончится в момент $(0+5) = 5$, работа (1, 2) – в момент $(1+3) = 4$. Событие 2 наступит тогда, когда завершатся обе работы, т.е., в момент

$$t(2) = \max(5, 4) = 5$$

Событие 3. Событие 3 наступит после завершения работ (1, 3) и (2, 3). Первая из них начинается в момент $t(1) = 1$ и длится 2 единицы времени, вторая – в момент $t(2) = 5$ и длится 6 единиц времени. Таким образом,

$$t(3) = \max(1+2, 5+6) = \max(3, 11) = 11$$

Событие 4. Событие 4 наступит после завершения работ (3, 4) и (2, 5). По аналогии с предыдущим получаем $t(4) = \max(11+0, 5+2) = \max(11, 7) = 11$.

Событие 5. Событие 5 – конечное, для него

$$t(5) = \max(11+5, 11+3) = \max(16, 14) = 16$$

Таким образом, максимально быстро весь процесс может закончиться за время $T_{кр} = 16$.

Полное описание метода, который помимо длины критического позволяет определить сам критический путь, а также параметры событий и работ, можно найти в литературе, рекомендованной для изучения темы.

Однако в реальности имеет место неопределенность как в структуре графа (те или иные события или работы могут присутствовать или же нет), так и во временных параметрах - времена выполнения работ, моменты наступления событий, резервы и пр. Поэтому для практических случаев применяются другие методы, одним из которых является метод PERT (Program Evaluation and Review Technique), относящийся к аналитическим методам и позволяющим получить значения вероятностных параметров модели при условии соблюдения некоторых ограничений.

Метод, в частности, предполагает, что продолжительности работ $t(i, j)$ подчиняются так называемому β -распределению с одинаковыми значениями параметров и такими, что средние $\overline{t(i, j)}$ и дисперсии σ_{ij}^2 длительностей работ могут быть приближенно рассчитаны по формулам:

$$\overline{t(i, j)} \approx \frac{3t_{\min}(i, j) + 2t_{\max}(i, j)}{5} \quad (1)$$

и

$$\sigma_{ij}^2 \approx \frac{1}{25} (t_{\max}(i, j) - t_{\min}(i, j))^2 \quad (2)$$

где t_{\min} и t_{\max} означают минимальную и максимальную продолжительности выполнения работ.

Кроме того, предполагается, что длительности работ независимы, а средняя величина критического пути существенно превосходит длительности прочих полных путей.

Порядок расчета вероятностной модели методом PERT таков.

- 1) Вершины графа нумеруются.
- 2) Для каждой работы находят оценки t_{\min} и t_{\max} .
- 3) Определяются математическое ожидание $\overline{t(i, j)}$ и дисперсия σ_{ij}^2 длительностей работ.
- 4) На основании совокупности значений $\overline{t(i, j)}$ проводится обычный расчет характеристик, как для детерминированной сетевой модели.
- 5) Определяется критический путь $L_{кр}$ и его среднее значение:

$$\overline{T}_{кр} = \sum_{(i, j) \in L_{кр}} \overline{t(i, j)}$$

- 6) Определяется дисперсия длительностей $L_{кр}$ как сумма дисперсий длительностей критических работ (предположение о независимости работ):

$$\sigma_{кр}^2 = \sum_{(i, j) \in L_{кр}} \sigma_{ij}^2$$

7) Поскольку длительности $t(i, j)$ – независимые случайные величины, их сумму $I_{кр}$ можно считать случайной величиной, распределенной по нормальному закону $N(\bar{T}_{кр}, \sigma_{кр})$ со средним $\bar{T}_{кр}$ и дисперсией $\sigma_{кр}^2$, для которого функция плотности вероятности имеет вид:

$$f(T_{кр}) = \frac{1}{\sigma_{кр} \sqrt{2\pi}} e^{-\frac{(T_{кр} - \bar{T}_{кр})^2}{2\sigma_{кр}^2}}$$

8) Из свойств нормального распределения следует (правило «трех сигма»), что с вероятностью 0,9974 значение $\bar{T}_{кр}$ будет находиться в интервале:

$$(\bar{T}_{кр} - 3\sigma_{кр}, \bar{T}_{кр} + 3\sigma_{кр})$$

Поэтому можно утверждать, что:

$$\bar{T}_{кр.min} = \bar{T}_{кр} - 3\sigma_{кр},$$

$$\bar{T}_{кр.max} = \bar{T}_{кр} + 3\sigma_{кр}$$

9) Если определен некоторый плановый срок выполнения всей совокупности работ (проекта) $T_{пл}$, то вероятность выполнения в этот срок всей совокупности работы определяется следующим образом:

$$P(T_{кр} \leq T_{пл}) = \int_{-\infty}^{T_{пл}} \frac{1}{\sigma_{кр} \sqrt{2\pi}} e^{-\frac{(T_{кр} - \bar{T}_{кр})^2}{2\sigma_{кр}^2}} dT_{кр} \quad (3)$$

Чтобы найти значение функции нужно перейти от $N(\bar{T}_{кр}, \sigma_{кр})$ к стандартному распределению $N(0,1)$. Для этого осуществляется замена переменной:

$$y = \frac{T_{кр} - \bar{T}_{кр}}{\sigma_{кр}} \quad (4)$$

что приводит к изменению подынтегральной функции и пределов интегрирования:

$$T_{кр} = -\infty;$$

$$y = -\infty;$$

$$T_{кр} = T_{пл},$$

$$y = \frac{T_{пл} - \bar{T}_{кр}}{\sigma_{кр}};$$

$$T_{кр} = \bar{T}_{кр},$$

$$y=0;$$

$$dT_{кр} = \sigma_{кр} \cdot dy.$$

Тогда

$$x = \frac{T_{пл} - \bar{T}_{кр}}{\sigma_{кр}} \quad (5)$$

и

$$P(T_{кр} \leq T_{пл}) = \Phi(x),$$

где

$\Phi(x)$ называется **функцией Лапласа**.

Значение функции можно получить с использованием библиотечных функций, которые присутствуют во многих компиляторах и программных пакетах.

Проиллюстрируем применение метода на примере сетевой модели рис. 45, в которой значения t_{\min} и t_{\max} для каждой работы указаны через точку с запятой, а вычисленные значения показаны жирным шрифтом (рис. 46).

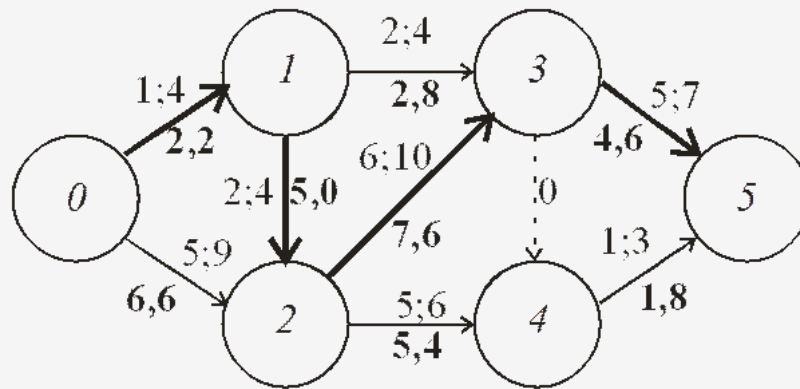


Рис. 46. Пример расчета методом PERT

В соответствии с методом по формуле (1) находим средние значения работ. Например,

$$\overline{t(0,1)} = \frac{3 \cdot 1 + 2 \cdot 2}{5} = 2,2$$

$$\overline{t(1,2)} = \frac{3 \cdot 3 + 2 \cdot 8}{5} = 6,0$$

критический путь

$$L_{кр} = 0, 1, 2, 3, 5 \text{ (отмечен на Рис. 46 жирной линией)}$$

и его величину

$$\overline{T}_{кр} = \overline{t(0,1)} + \overline{t(1,2)} + \overline{t(2,3)} + \overline{t(3,5)} = 2,2 + 5,0 + 7,6 + 4,6 = 19,4$$

По формуле (2) вычисляются дисперсии σ_{ij}^2 работ критического пути:

$$\sigma^2(0,1) = \left(\frac{4-1}{5} \right)^2 = 0,36$$

$$\sigma^2(1,2) = \left(\frac{8-3}{5} \right)^2 = 1,0$$

$$\sigma^2(2,3) = \left(\frac{10-6}{5} \right)^2 = 0,64$$

$$\sigma^2(3,5) = \left(\frac{7-3}{5} \right)^2 = 0,64$$

дисперсия

$$\sigma_{кр}^2 = 0,36 + 1,0 + 0,64 + 0,64 = 2,64$$

и среднее квадратичное отклонение

$$\sigma_{кр} = \sqrt{2,64} = 1,63$$

критического пути.

Определяются граничные значения критического пути:

$$T_{кр\min} = \overline{T}_{кр} - 3\sigma_{кр} = 19,4 - 1,63 \cdot 3 = 19,4 - 4,9 = 14,5$$

$$T_{кр\max} = \overline{T}_{кр} + 3\sigma_{кр} = 19,4 + 1,63 \cdot 3 = 19,4 + 4,9 = 24,3$$

(иначе говоря, с вероятностью 0,9974 все работы будут выполнены в срок от 14,5 до 24,3 временных единиц).

Оценим вероятность выполнения работы в срок $P(T_{кр} \leq T_{пл})$ для двух значений $T_{пл} = 20$ и $T_{пл} = 17$.

Для этого по формуле (4) находим соответственно значения x :

$$x = \frac{T_{пл} - \bar{T}_{кр}}{\sigma_{кр}} = \frac{20 - 19,4}{1,63} = 0,38$$

$$x = \frac{T_{пл} - \bar{T}_{кр}}{\sigma_{кр}} = \frac{17 - 19,4}{1,63} = -1,47$$

по которым определяем (например, с помощью функции НОРМСТРАСП программы Excel) значения вероятности:

$$P(T_{кр} \leq 20) = 0,648$$

$$P(T_{кр} \leq 17) = 0,070781$$

Из практики известно, что если вероятность $P(T_{кр} \leq T_{пл}) > 0,65$, то проект спланирован с запасом, и даже избыточным. Величина $P(T_{кр} \leq T_{пл}) < 0,35$ свидетельствует об угрозе срыва. Видим, что в нашем примере вероятность выполнить все работы в срок 17 временных единиц крайне низка.

Другим возможным способом решить задачу является *метод статистических испытаний*, или *метод Монте-Карло*, который представляет собой универсальный метод решения многих задач и будет рассматриваться далее.

В заключение обсуждения сетевой модели отметим, что наряду с возможностью преобразования модели, созданной на основе аппарата сетей Петри в сетевую модель, возможно и **обратное преобразование**: сетевая модель может быть легко преобразована в модель на основе сети Петри.

Выводы:

1. Прогресс информационных технологий в ряде областей обусловил появление новой концепции организации и моделирования деловых процессов под названием *поток работ*. Построенные на базе данной концепции автоматизированные системы управления потоками работ позволяют получить ряд существенных выгод от их внедрения.
2. В настоящий момент не существует общепринятого стандарта формального описания потоков работ, поэтому проблема создания унифицированного и независимого от платформы языка моделирования потока работ остается актуальной и исследования в данном направлении продолжаются.
3. Для моделирования потока работ применяются различные методы. Одним из наиболее широко распространенных является подход на основе теории сетей Петри. Вместе с тем, как этот, так и другие подходы имеют ряд ограничений и нуждаются в расширении.
4. Для определения количественных характеристик моделируемых процессов можно использовать разработанные методы и модели, такие как модель сетевого планирования. Степень проработки ее формального аппарата дает возможность применять ее во многих практических ситуациях с достаточной точностью.

Вопросы для самопроверки:

1. Что понимается под потоком работ?
2. Какие задачи решаются системой управления потоками работ?
3. Какие преимущества дает использование автоматизированных систем управления потоками работ?
4. Какие средства используются для моделирования потока работ?
5. Что представляет собой сеть Петри?
6. Какие основные объекты определяются в сети Петри?
7. Что такое места (позиции) в сети Петри?
8. Что бывают места в сети Петри?
9. Что такое переходы в сети Петри?
10. Что такое метки (фишки, маркеры) в сети Петри?
11. Что называется состоянием сети Петри и как оно задается?
12. Как реализуются ограничения в сети Петри?
13. Что позволяют описать и каким образом раскрашенные сети Петри?
14. Что позволяют описать и каким образом временные сети Петри?
15. Что позволяют описать и каким образом иерархические сети Петри?
16. Что ограничивает применение теории сети Петри для моделирования потока работ?
17. Что называется сетевой моделью?
18. Что понимается под событием?
19. Что называется работой?
20. Что такое фиктивная работа, и в каких случаях она нужна?
21. Что такое путь?
22. Какой путь называется критическим, каков его физический смысл?
23. Опишите логическую последовательность шагов нахождения величины критического пути на сетевом графе.
24. В чем могут состоять отличия между вероятностной и детерминированной сетевыми моделями?

Литература по теме:

Основная литература:

1. Wil van der Aalst. Workflow Management: Models, Methods, and Systems/ Wil van der Aalst and Kees Max van Hee-The MIT Press Cambridge, Massachusetts London, 2002 -368p.
2. Питерсон Дж. Сети Петри – М.: Мир, 1984 – 264с.
3. Кофман А., Дебазей Г. Сетевые методы планирования и их применение - М.: Прогресс, 1968. - 182 с.

Дополнительная литература:

1. Rob Allen. Workflow: An Introduction // <http://www.wfmc.org/View-document-details/Workflow-Interoperability-Enabling-E-Commerce-Rob-Allen.html?tmpl=component>
2. Workflow Management Coalition (WfMC) // <http://www.wfmc.org/>
3. Шаблоны потоков работ // <http://www.workflowpatterns.com/>

Практические задания.

Задание 1.

Преобразуйте сеть Петри рис. 42 для следующих случаев:

- а) ремонтом компьютеров занят только один техник;
- б) один техник занимается как осмотром компьютеров, так и их ремонтом.

Тема 7. Имитационное моделирование информационных процессов и систем

Цели изучения темы:

- изучить сущность подхода на основе методологии имитационного моделирования к анализу и синтезу информационных процессов и систем.

Задачи изучения темы:

- изучить сущность и границы применимости метода статистических испытаний;
- изучить основные принципы компьютерной имитации стохастических процессов;
- понять возможности моделирующих комплексов для создания и применения имитационных моделей.

Успешно изучив тему, Вы:

получите представление о:

- когда применяется и как практически реализуется модель на основе метода статистических испытаний;
- принципах построения и механизме программной имитации стохастических процессов;

будете знать:

- как можно создавать модели с помощью систем имитационного моделирования;
- из чего состоят и как следует анализировать результаты прогонов программной модели.

Вопросы темы:

1. Метод статистических испытаний.
2. Концепции имитационного моделирования.
3. Создание имитационных моделей с помощью систем моделирования.
4. Примеры имитационных моделей.

Вопрос 1. Метод статистических испытаний.

Аналитические модели, рассмотренные ранее, могут с успехом применяться на этапе системного анализа и для решения ряда задач этапа проектирования, когда требования к точности результата не являются слишком строгими. Такими задачами могут быть задачи выбора конфигурации или модели сервера, где требуется определить только класс выбираемого устройства или подсистемы.

Однако в ряде практических ситуаций применение аналитических моделей приводит к большой потере точности получаемого результата из-за того, что модели строятся на слишком приблизительном описании реальных процессов. В этих случаях можно использовать подход на основе имитационного моделирования процессов, который позволяет практически с любой степенью точности описать исследуемые процессы и отличается универсальностью применения.

Рассмотрение имитационных моделей начнем с метода статистических испытаний.

Статистическое моделирование является разновидностью имитационного моделирования и состоит в обработке данных о системе (модели) с целью получения статистических характеристик системы. Чаще всего оно применяется как способ исследования процессов поведения систем в условиях, когда внутренние взаимодействия в системах неизвестны, и построение аналитической модели явления затруднено или вовсе неосуществимо. Метод может применяться и на этапе системного анализа информационных систем, но основными для его применения являются задачи исследования операций, массового обслуживания и многие другие, связанные со случайными процессами. Эти задачи возникают в основном на этапах системного проектирования и разработки.

Смысл метода Монте-Карло состоит в том, что исследуемый процесс моделируется путем многократных повторений его случайных реализаций (рис. 47).

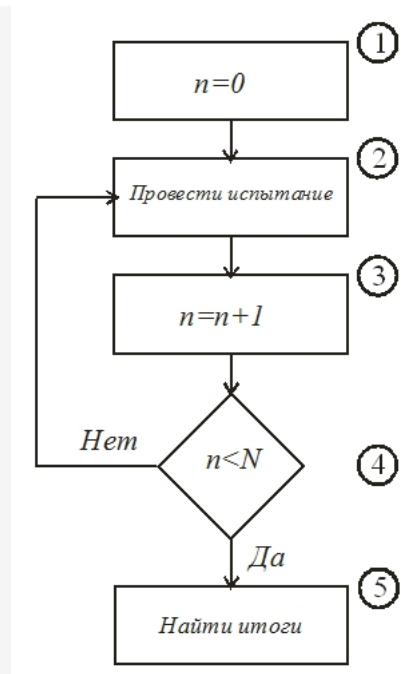


Рис. 47. Схема метода Монте-Карло

Единичные реализации называются **статистическими испытаниями**, в силу чего метод называется также методом статистических испытаний.

Поясним применение метода Монте-Карло к решению задачи оценивания величины критического пути сетевой модели, показанной на рис. 45. На примере этой модели рассматривался метод нахождения величины критического пути при детерминированных условиях. Будем теперь предполагать, что длительности работ не являются неизменными, а варьируются в зависимости от случайных факторов, что, как уже отмечалось, встречается на практике значительно чаще.

Каждое испытание (блок 2 на рис. 47) будет включать два шага:

- 1) генерацию значений длительностей всех работ $t(i,j)$ на графе сетевой модели;
- 2) расчет на основе применения описанного ранее метода критического пути величины критического пути $T_{кр}$ для данной совокупности значений длительностей работ.

Шаг 1 выполняется на основе наблюдений за случайными величинами, представляющими собой длительности работ графа. Результат этих наблюдений выражается некоторым законом распределения вероятностей. Генерация случайных величин для каждой их реализации проводится с помощью программных средств. Эти средства строятся на основе специальных алгоритмов получения *псевдослучайных* чисел, т.е., не являющихся, строго говоря, результатом измерения характеристик каких-то физических явлений, а достаточно точно их имитирующих. К настоящему времени предложено и реализовано большое количество этих алгоритмов, которые позволяют, с одной стороны, получить достаточно хорошие выборки значений и, с другой стороны, весьма сводят к минимуму потребности в вычислительных ресурсах, которые необходимы для выполнения программ.

Шаг 2 в данном случае состоит в обработке данных, полученных в результате выполнения шага 1, а именно, массива N значений $T_{кр}$. Если задача ставится, как задача нахождения только среднего значения $T_{кр}$, то обработка сведется к выполнению лишь одной операции:

$$\overline{T_{кр}} = \frac{\sum_{i=1}^N T_{кр}^i}{N},$$

где $T_{кр}^i$ - значение $T_{кр}$, полученное в i -ом испытании.

Однако данные, собранные в результате моделирования, содержат в себе более полную информацию относительно исследуемого показателя. Поэтому целесообразно использовать результат для построения гистограммы частот значений $T_{кр}$. Например, моделирование методом Монте-Карло графа рис. 45, в котором длительности работ распределены по равномерному закону со средними значениями, показанными на рисунке, и величиной полуинтервалов изменений равными 1/3 средних значений, даст следующие результаты:

Гистограмма распределения длины критического пути

MIN	MAX	N	%
<<<	14	1032	0.1032
14	15	1551	0.1551
15	16	2133	0.2133
16	17	2166	0.2166
17	18	1712	0.1712

18	19	956	0.0956
19	20	365	0.0365
20	>>>	85	0.0085

$T_{кр.ср}=16.1486$

Представление гистограммы частот в графическом виде дано на рис. 48.



Рис. 48. Пример гистограммы частот $T_{кр}$

Пр этой гистограмме можно оценить величину вероятности выполнения совокупности работ в течение заданного времени $P(T_{кр} \leq T_{пл})$. Например, если нужно оценить, с какой вероятностью может быть выдержан срок $T_{пл}=18$, то находим:

$$P(T_{кр} \leq T_{пл}) = 0,1032 + 0,1551 + 0,2133 + 0,2166 + 0,1712 = 0,8594$$

что можно считать очень большой величиной.

Ниже приводится текст программы на языке C++ для выполнения на ее основе практического задания.

```
//
// Нахождение распределения величины критического пути сетевого графа методом Монте-Карло
//
// Заголовочные файлы -----
#include "stdafx.h" //стандартные заголовки
#include <time.h> //функции работы с системным временем (time)
#include <math.h> //математические функции (log())
#include <io.h> //для функции creat()
#include <fstream.h> //поточковый класс
#include <sys/stat.h> //значения параметров amode
#include <iomanip.h> //манипуляторы ввода-вывода
#include <float.h> //константы предельных значений
// Функции -----
// Равномерное распределение [0,1] (конгруэнтный алгоритм)
float rundum(void)
{
long static bx = (long)time(NULL);
unsigned long m = 2<<31 - 1;
if (bx % 2 == 0) bx++;
bx = (16807 * bx) % m ;
return((long double) bx / (long double)m);
}

//Нормальное распределение
float normal(float m, float s )
{
float a;
int i;
for (i=0,a=0.0; i<12;i++) a+=rundum();
return(m+(a-6.0)*s);
}
//Экспоненциальное распределение (метод обратной функции)
float expont(float m )
{
return(m*(-log(rundum())) );
}
//Равномерное распределение на произвольном интервале [m-s,m+s]
float unifrm (float m , float s)
{
}
```



```

return ( m-s + 2*s*rundum() );
}
//Треугольное распределение
float triplex(float a, float m, float b )
{
float x,r;
r=rundum();
if (r<=(m-a)/(b-a))
x=a+sqrt(r*(m-a)*(b-a));
else
x=b-sqrt((1.0-r)*(b-m)*(b-a));
return(x);
}
//Длительность работы для текущего испытания
float duration (int flag, float parm_1, float parm_2, float parm_3)
{
if (flag == 1) return(parm_1);
if (flag == 2) return(normal(parm_1,parm_2));
if (flag == 3) return(expont(parm_1));
if (flag == 4) return(unifrm(parm_1,parm_2));
if (flag == 5) return(triplex(parm_1,parm_2, parm_3));
}
//Вывод гистограммы распределения
void printhist(char hdr[],int fileNumb,int dim,float bounds[],int counts[])
{ofstream fileOut;
int i;
float sum;
fileOut.attach (fileNumb);
fileOut << hdr << setprecision(4);
fileOut << "\n-----"
<<"\n"<<setw(5)<<"MIN"<<setw(10)<<"MAX"<<setw(10)<<"N"<<setw(10)<<" % "
<< "\n-----";
for ( i=0,sum=0; i<dim; sum+=counts[i++]); //сумма значений счетчиков
for ( i=0; i<dim;i++)
{
if (i==0) fileOut << "\n" << setw(5) << "<<<" << setw(10) << bounds [i+1];
else
if (i< dim-1) fileOut << "\n" << setw(5) << bounds[i] << setw(10) << bounds[i+1];
else fileOut << "\n" << setw(5) << bounds[i] << setw(10) << ">>>";
fileOut << setw(10) << counts[i]
<< " " << static_cast<float> (counts[i])/sum; //*100;
}
fileOut << "\n-----";
}
int APIENTRY WinMain(HINSTANCE hInstance,HINSTANCE hPrevInstance, LPSTR lpCmdLine,int
nCmdShow)
{
//Параметры моделирования

```

int	Nexp=10000;	//число испытаний метода Монте-Карло
int	stock=5;	//номер конечной вершины (стока) графа работ
int	distr=4;	//вид распределения длительностей работ:
		// 1-постоянное,
		// 2-нормальное,
		// 3-экспоненциальное
		// 4-равномерное
		// 5-треугольное
//Параметры распределения длительности работ :::::::::::::::::::::::::::::::		
float	t_1[100][100]= {FLT_MAX};	//средние(distr=1-3),минимальные(distr=4)
	t_1[0][1]=1;	
	t_1[0][2]=5;	
	t_1[1][2]=3;	
	t_1[1][3]=2;	
	t_1[2][3]=6;	
	t_1[2][4]=5;	
	t_1[3][4]=0;	
	t_1[3][5]=5;	
	t_1[4][5]=3;	
float	t_2[100][100]= {FLT_MAX};	//ст.откл.(distr=2),полуинтервалы(distr=4),наиб.вероятн. (distr=5)
	t_2[0][1]=t_1[0][1]/3;	
	t_2[0][2]=t_1[0][2]/3;	
	t_2[1][2]=t_1[1][2]/3;	
	t_2[1][3]=t_1[1][3]/3;	
	t_2[2][3]=t_1[2][3]/3;	
	t_2[2][4]=t_1[2][4]/3;	
	t_2[3][4]=t_1[3][4]/3;	
	t_2[3][5]=t_1[3][5]/3;	
	t_2[4][5]=t_1[4][5]/3;	
float	t_3[100][100]= {FLT_MAX};	//максимальные(distr=5)
	bounds[]={FLT_MIN,14.0f,15.0f,16.0f,17.0f,18.0f,19.0f,20.0f,FLT_MAX};	//границы

float	интервалов	
//Организация вывода результатов -----		

int	reportfile=creat ("D:\\MyResults.txt", S IWRITE);	//создание файла
ofstream	fileOut;	//выводной файл
fileOut.attach (reportfile);		//включение в поток
//Переменные =====		
long	i,j,k;	
int const	dim = sizeof bounds / sizeof (float)-1;	//число интервалов
int	counts [dim]={ 0 };	//счетчики
float	task[100][100];	//длительности работ в конкретном испытании
float	times[50];	//моменты наступления событий
float	Tsum=0;	//накопитель Tкр
//Испытания =====		
for (k = 1; k <= Nexр; k++) //повторение испытаний		
{		
//Генерация длительностей работ для текущего испытания		
for(i = 0; i <99 ; i++)		
for (j = 0; j <99; j++)		
if (t_1[i][j]<FLT_MAX)		
task[i][j]=duration(distr,t_1[i][j],t_2[i][j],t_3[i][j]);		
else		
task[i][j]=FLT_MAX;		
//Определение величины критического пути для текущего испытания		
for(i = 1,times[0]=0; i <= stock; i++)		
for(j = 0,times[i]=0; j < i; j++)		
if (task[j][i]<FLT_MAX)		
if(times[j]+task[j][i]>times[i])		
times[i]=times[j]+task[j][i];		
//Сбор статистики		
for(i=0;i<dim;i++)		
if(times[stock]>bounds[i] & times[stock] <=bounds[i+1])		
counts[i]++;		
Tsum+=times[stock];		
}		
//Итоги :-----		
printhist("Гистограмма распределения длины критического пути", reportfile, dim, bounds, counts);		
fileOut << "\nTкр.cp=" << Tsum / Nexр;		
//=====		
return(0);		
}		

Значения параметров моделирования задаются в разделе с заголовком (строкой комментария) «Параметры моделирования» (выделен в вышеприведенном тексте полужирным шрифтом). В разделе задаются:

Nexр	Число повторений экспериментов метода Монте-Карло
stock	Номер конечной вершины графа сетевой модели
distr	Целочисленный код вида распределения длительностей работ в сетевом графе (перечень возможных видов распределения и соответствующие коды см. в тексте программы)
t_X	Двумерные массивы, задающие параметры распределения для каждой работы (i,j) сетевого графа. Смысл параметров t_1, t_2, t_3 зависит от вида распределения, задаваемого значением distr (смысл параметра, задаваемого каждым массивом см. в тексте программы).
bounds	Массив с границами диапазонов (интервалов) значений критического пути, используемый для сбора статистики попаданий значения в каждый интервал.

Результаты выводятся в файл, обозначаемый переменной *reportfile*.

Недостатком метода Монте-Карло являются большие затраты машинного времени, требуемые для достижения большей статистической точности (известно, что для уменьшения дисперсии результата в k раз, число испытаний необходимо увеличить в k^2 раз).

Вопрос 2. Концепции имитационного моделирования.

Аналитические модели, как и модели вообще строятся для некоторых допущений. В случае аналитических моделей эти допущения могут ухудшать точность получаемых на их основе результатов в силу ограничений, накладываемых используемым математическим аппаратом. Кроме того, получение аналитических выражений для расчета показателей часто представляет большую трудность. Альтернативным подходом к решению задач оценки показателей является исследование системы на основе **имитационной** модели, с помощью которой можно получить результаты для случаев произвольных распределений временных интервалов и других случайных величин. Такую модель можно построить с помощью специальной моделирующей системы, что ускоряет процесс создания, улучшает характеристики конечного результата (программной модели) и имеет ряд других достоинств.

Как уже обсуждалось ранее поведение (процесс функционирования) динамической дискретной системы можно рассматривать как последовательность состояний $z(t_1), z(t_2), \dots, z(t_n)$ в моменты времени t_1, t_2, \dots, t_n . При этом факты смены состояния системой рассматриваются как **события**. В основу идеи имитационного моделирования положен принцип, согласно которому на ЭВМ осуществляется воспроизведение (имитация) процесса функционирования исследуемой системы в контексте смены ее состояний в результате происходящих в ней событий. В процессе проведения имитации собираются данные о состоянии системы

или отдельных ее элементов в определенные моменты времени, что позволяет в конце процесса моделирования провести агрегирование и обработку собранных данных и получить значения нужных исследователю показателей.

Пусть, например, нам необходимо исследовать работу системы массового обслуживания (СМО) с очередью и одним обслуживающим прибором (Рис. 49).

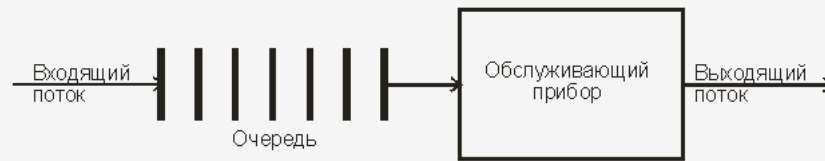


Рис. 49. Одноканальная система массового обслуживания с очередью

Заявки, образующие входящий поток, поступают в систему в случайные моменты времени. Если обслуживающий прибор (канал) свободен, заявка принимается на обслуживание, которое продолжается некоторое время (его величина является, как правило, также случайной), после чего она уходит из системы (выходящий поток). Если в момент прихода заявки канал занят, заявка попадает в очередь ждущих заявок.

Решая задачу нахождения показателей системы, для описания состояния процесса достаточно использовать один параметр $z(t_i)$, который означает число заявок в системе в момент времени t_i . Имитацию процесса обслуживания заявки можно провести, определяя моменты ожидаемого изменения состояния (наступления события), связанных с изменением состояния $z(t_i)$. В данном случае, это моменты поступления очередной заявки в систему и моменты окончания обслуживания заявки, находящейся в обслуживающем приборе. Моменты наступления будущих событий могут быть найдены на основе рекуррентных выражений с помощью функций распределения интервалов времени путём проведения случайных испытаний аналогично тому, как это производилось в методе статистических испытаний. Это позволяет построить алгоритм имитации, состоящий из повторения следующей совокупности шагов:

- находится момент наступления ближайшего события, событие с минимальным временем — наиболее раннее событие;
- стрелки модельных часов передвигаются на этот момент времени;
- определяется тип произошедшего события;
- в зависимости от типа события модифицируются значения переменных, описывающих состояние системы (в данном случае, число находящихся в системе заявок), и определяются следующие моменты наступления событий.

Логическая последовательность шагов моделирования показана на рис. 50.

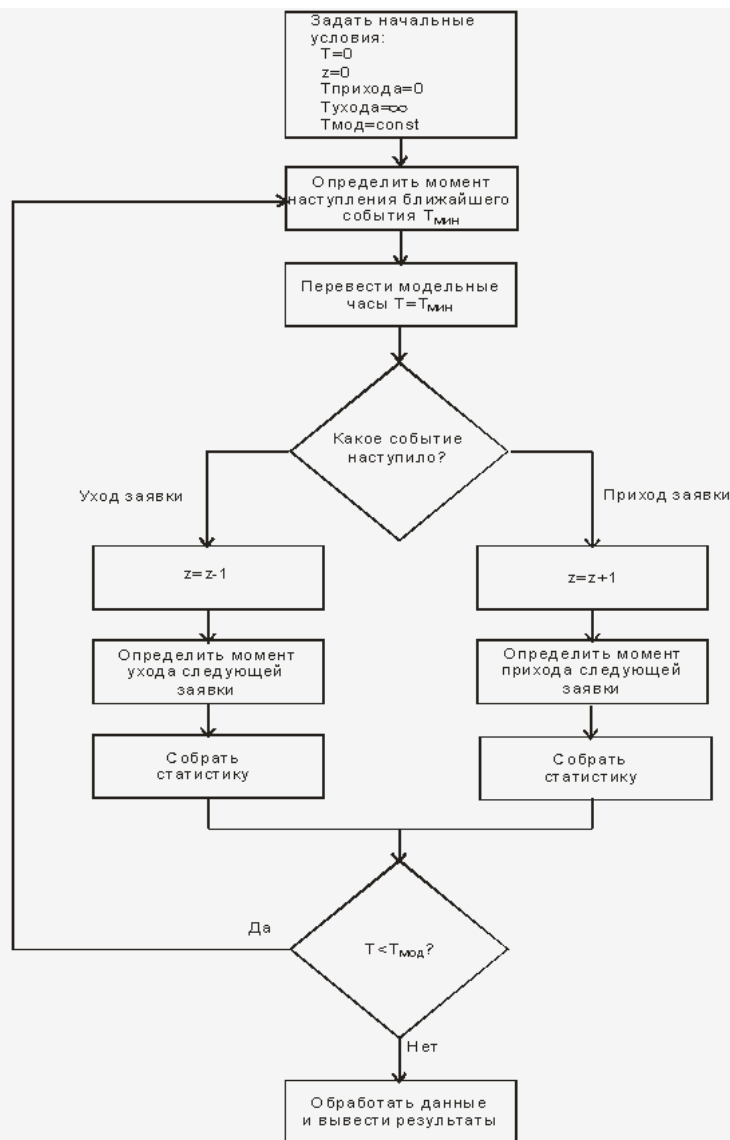


Рис. 50. Блок-схема имитации процессов в одноканальной СМО с очередью

В процессе выполнения алгоритма производится накопление значений существенных параметров моделируемой системы. По окончании моделирования осуществляется статистическая обработка полученных величин и выдача результатов. Описанный алгоритм носит название **событийного алгоритма**.

Альтернативным способом построения алгоритма имитационного моделирования является перемещение по временной оси с фиксированным интервалом (шагом). После каждого перехода на новую временную отметку проводится сканирование всех протекающих в моделируемой системе процессов и выявление всех свершившихся на данный момент существенных событий. Такой алгоритм носит название **пошагового алгоритма**.

Преимуществом пошагового алгоритма является простота его программной реализации. Вместе с тем, в условиях, когда наступление событий, обусловленных разными процессами, происходит с сильно отличающимися интервалами, то значительное время моделирования оказывается затраченным впустую (например, при моделировании производственного процесса на предприятии, работающем в односменном режиме, все вечернее и ночное время программная модель не будет выполнять никакой содержательной работы). Поэтому большинство программных реализаций имитационных моделей использует событийный алгоритм имитации.

Имитационное моделирование представляет собой процесс, реализуемый с помощью средств вычислительной техники, что позволяет автоматизировать решение таких задач как:

- 1) Создание или модификация имитационной модели.
- 2) Проведение модельных экспериментов и интерпретация получаемых результатов.

В случаях разработки сложных моделей, предназначенных для проведения большого объема экспериментов, для решения этих задач используются **моделирующие комплексы (системы)**.

Имитационное моделирование как информационная технология включает несколько основных этапов.

На этапе **структурного анализа процессов** проводится формализация реального процесса, его декомпозиция на подпроцессы, которые могут также подвергаться декомпозиции. Задачи, решаемые на этом этапе, соответствуют задачам создания модели деловых процессов, вместе с тем, в зависимости от применяемых средств моделирования результат может потребовать преобразования формы представления в вид, наиболее полно соответствующий задачам, решаемым на дальнейших шагах.

Описание модели в вербальном, графическом или полуформализованном виде должно быть представлено на **специальном (формальном) языке** для ввода в моделирующую систему. При этом могут применяться:

- *Ручной способ.* Описание составляется на языке какой-либо системы моделирования, например, GPSS, Pilgrim или алгоритмическом языке, например, Visual Basic (размер результирующего текста в последнем случае будет многократно превосходить размер текста на языке специальных систем).
- *Автоматизированный способ.* Текст на формальном языке получается в результате автоматической генерации описания, которое строится в интерактивном режиме с помощью высокоуровневого графического конструктора. Такой конструктор, которым оснащена, в частности, система Pilgrim, позволяет фактически совместить этап разработки программной модели с этапом системного анализа.

Созданное формальное описание далее **преобразуется в компьютерную программу**, способную воспроизводить реальные процессы. Преобразование, которое обычно представляет собой трансляцию и редактирование связей, осуществляется в некоторой программной среде, может проводиться в двух режимах.

- *Интерпретация.* В этом режиме специальная программа-интерпретатор осуществляет всю процедуру имитации, непосредственно выполняя инструкции формального описания. Примером системы, где предусмотрен режим интерпретации, является система GPSS.
- *Компиляция.* В этом режиме на основе формального описания создается отдельная программа (исполнительный модуль), который может запускаться независимо от программной среды, в которой он был создан. Примером системы, где предусмотрен режим интерпретации, является система Pilgrim.

Целью следующего этапа является **обоснование модели**, заключающегося в демонстрации пригодности модели для получения нужных показателей, т.е. обеспечение необходимых точности и надежности получаемых на ее основе результатов.

Основной, последний этап моделирования сводится к проведению **машинных экспериментов** с программной моделью, сбору, накоплению, обработке и анализу собранных данных.

Вопрос 3. Создание имитационных моделей с помощью систем моделирования.

С началом развития средств вычислительной техники и связанного с ним прогресса в области теории и практики имитационного моделирования создано более 300 языков моделирования дискретных систем и процессов. Одной из наиболее старых систем моделирования является система GPSS, которая была разработана сотрудником компании IBM Джеффри Гордоном в 1961 году (он же был и создателем описанного выше событийного алгоритма). Система на сегодняшний день остается в числе широко распространенных, к тому же в своей несколько усеченной версии доступна для свободного применения. Поэтому основные концепции моделирующих комплексов рассмотрим на примере системы GPSS.

Система GPSS (General Purpose Simulating System) предназначена для построения имитационных моделей систем с дискретным множеством состояний различной степени сложности и включает в себя язык моделирования и программу-интерпретатор текста описания модели.

Элементами **языка моделирования** системы GPSS являются абстрактные объекты и операции. Объекты в системе подразделяются на 7 категорий и 15 типов. К первой категории и первому типу относятся *динамические* объекты, называемые **транзактами**, представляющие собой единицы исследуемых потоков. Работа GPSS-модели сводится к перемещению транзактов между блоками модели согласно определенным в описании модели правилам. Содержательный смысл транзактов вытекает из смысла моделируемой системы или процесса. В процессе моделирования транзакты могут создаваться и уничтожаться, в каждый момент в модели может находиться много транзактов, но перемещается только один.

Основу GPSS-интерпретатора составляет диспетчерская программа (**симулятор**), выполняющая следующие функции:

- перемещает транзакты по заданным маршрутам;
- планирует происходящие в модели события посредством выявления моментов наступления событий в будущем и их реализации в порядке возрастания значений моментов наступления;
- производит сбор статистики о функционировании модели;
- осуществляет продвижение модельного времени в процессе моделирования.

Время в модели **дискретно**, масштаб (соотношение между безразмерной единицей модельного времени и единицей реального времени) определяется разработчиком.

Для наглядного представления модели может применяться язык **блок-схем**. Блок-схемы представляют собой набор фигур, соединенных линиями, показывающими порядок прохождения транзактов по блокам модели. Блок может иметь несколько входящих в него линий, что указывает на то, что он является общим для нескольких последовательностей событий, а также несколько исходящих линий, что указывает на то, что выходящий транзакт может направляться в разные блоки в зависимости от некоторых правил или условий.

В процессе выполнения программной модели транзакты хранятся в **списках**. Этих списков всего пять, но в каждый момент времени транзакт может находиться только в одном.

Список текущих событий хранит транзакты, планируемое время наступления событий для которых равно или меньше текущего времени (транзакты должны были уже перемещаться ранее, но были заблокированы).

Список будущих событий хранит транзакты, планируемое время наступления событий для которых больше текущего времени (транзакты должны переместиться в будущем).

Список отложенных прерываний хранит отдельные транзакты, обслуживание которых занятыми или захваченными устройствами было прервано.

Список пользователя хранит транзакты, которые пользователь удалил из списка текущих событий и поместил в список временно пассивных.

Список синхронизируемых сообщений хранит транзакты, находящиеся в данный момент в состоянии сравнения.

В зависимости от различных условий транзакты помещаются в соответствующие списки. **Симулятор** продвигает каждый транзакт до того момента, пока его продвижение не будет заблокировано занятым блоком или когда транзакт войдет в блок, задающий временную задержку. Задержанные транзакты попадают в список будущих событий, заблокированные транзакты – в список текущих событий. Просматривая список текущих событий, симулятор пытается передвинуть содержащиеся в нем транзакты к следующим блокам модели. Если это сделать нельзя, то системные часы, хранящие *абсолютное время модели*,

переводятся на момент наиболее раннего из будущих событий, соответствующий транзакт переносится в список текущих событий и передвигается, если возможно, к следующему блоку модели. Если рассматриваемый транзакт входит в блок задержки, то он снова попадает в список будущих событий. В случае блокировки транзакт остается в списке текущих событий.

Каждому из GPSS-объектов выделяется участок оперативной памяти, в котором в процессе моделирования хранятся атрибуты объекта. Те из них, которые доступны пользователю, носят название **стандартных числовых атрибутов (СЧА)**. Так, время нахождения транзакта в модели (*резидентное время*) хранится в СЧА с именем M1. Для различных объектов модели (прибор, многоканальное устройство, очередь) определены свои СЧА, что отражается в их **именах**, первая часть которых идентифицирует тип объекта, а вторая часть его номер или имя.

Генерация случайных чисел осуществляется с помощью восьми встроенных **генераторов**, что позволяет назначить каждому из имитируемых в модели случайных процессов свой собственный генератор.

Модель в системе GPSS описывается с помощью последовательности **операторов**, каждый из которых может относиться к одному из трех типов.

- *Блоки*, образующие описание основных действий и их последовательности (блоки образуют основу блок-схемы модели).
- *Декларативные операторы*, необходимые для описания переменных, функций и памяти (многоканальных устройств).
- *Команды*, применяемые как для описания логических правил выполняемых действий, так и для организации взаимодействия пользователя с моделью.

Описание блока состоит из нескольких **полей**, отделяемых друг от друга пробелами или ограничителями (необязательные части заключены в квадратные скобки):

[<Метка>]<Операция><Операнды>[<Комментарии>]

Метка начинается с латинской буквы.

Операция означает команду, выполняемую при входе транзакта в блок.

Операнды необходимы для задания значений характеризующих операцию параметров.

Комментарии служат целям документирования и отделяются от операндов символом «;» или символом «*».

Объект, переменная и местоположение в программе обозначается **именем**, образуемым как последовательность символов длиной не более 250, начинающихся с символа и не совпадающего с ключевым словом языка.

Вопрос 4. Примеры имитационных моделей.

Рассмотрим несколько примеров моделирования с помощью системы GPSS. Для реализации модели использована свободно распространяемая версия системы GPSS World Student Version 5.2.2.

Начнем с ранее рассмотренной модели одноканальной СМО (рис. 49). Блок-схема, реализующая имитацию протекающих в этой СМО процессов, показана на рис. 50.

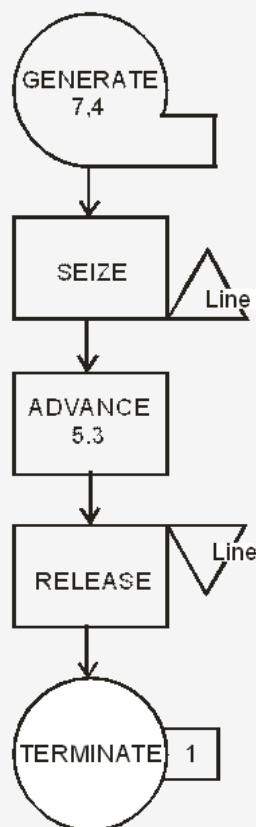


Рис. 51. Блок-схема GPSS-модели одноканальной СМО с очередью

Соответствующее описание с помощью операторов языка GPSS имеет вид:

	GENERATE	7,4
--	----------	-----

	SEIZE	Line
	ADVANCE	5,3
	RELEASE	Line
	TERMINATE	1

В приведенном описании присутствуют пять блоков.

Первый блок, GENERATE, обеспечивает поступление в модель транзактов (заявок в СМО). Транзакты поступают через интервалы времени, определяемые как равномерно распределенные случайные величины со средним значением 7 и полуинтервалом (размахом) 4, что задается операндами блока.

Второй блок, SEIZE, осуществляет имитацию попытки захвата транзактом устройства с именем Line.

Если попытка оказывается успешной (устройство Line свободно), транзакт попадает в это устройство и остается в нем на протяжении времени, которое определяется как равномерно распределенная случайная величина со средним значением 5 и полуинтервалом 3.

Эти значения задаются в *третьем блоке, ADVANCE*, имитирующем процесс обслуживания.

Если попытка захвата устройства оказывается неуспешной, то транзакт остается ждать освобождения устройства (возможно, вместе с другими ожидающим транзактами).

Четвертый блок, RELEASE, имитирует факт (событие) завершение обслуживания, состоящего в освобождении транзактом устройства с именем Line.

Пятый блок, TERMINATE, имитирует уход транзакта из системы.

После выполнения команды *Create Simulation* и запуска модели командой *Start* с операндом 1000 (число транзактов, которые должны пройти через модель в процессе моделирования) получаем отчет, показанный на рис. 52.

GPSS World Simulation Report - Untitled Model 1.1.1									
Tuesday, January 18, 2011 12:16:24									
START TIME		END TIME		BLOCKS	FACILITIES		STORAGES		
0.000		7049.623		5	1		0		
NAME				VALUE					
LINE				10000.000					
LABEL	LOC	BLOCK TYPE	ENTRY COUNT	CURRENT	COUNT	RETRY			
1	GENERATE	1001	0	0					
2	SEIZE	1001	1	0					
3	ADVANCE	1000	0	0					
4	RELEASE	1000	0	0					
5	TERMINATE	1000	0	0					
FACILITY	ENTRIES	UTIL.	AVE. TIME	AVAIL.	OWNER	PEND	INTER	RETRY	DELAY
LINE	1001	0.703	4.952	1	1001	0	0	0	0
CEC XN	PRI	M1	ASSEM	CURRENT	NEXT	PARAMETER	VALUE		
1001	0	7048.997	1001	2	3				
FEC XN	PRI	BDT	ASSEM	CURRENT	NEXT	PARAMETER	VALUE		
1002	0	7057.474	1002	0	1				

Рис. 52. Отчет с результатами моделирования одноканальной СМО

В верхней части отчета приводятся значения времени начала (0.000) и окончания (7049.623) моделирования, число блоков модели (5), число устройств модели (1) и число многоканальных устройств (0). Вслед за этим приводятся сведения по каждому имени, каждому блоку, каждому устройству модели вместе с относящейся к ним статистикой, после чего дается информация о списках текущих (CEC) и будущих (FEC) событиях.

Построенная модель позволяет достаточно точно (в предположении того, что имеется достаточно точная информация относительно распределений интервалов между приходами заявок и продолжительностью их обслуживания) имитировать реальные процессы, но содержит довольно скудную информацию относительно показателей качества моделируемой системы. В частности, значительный интерес для системного аналитика могут представлять сведения, касающиеся образующейся очереди заявок. Например, если исследуется процесс обработки потока файлов-заявок в информационной системе, характеристики очереди будут во многом определять требования к производительности сервера и объемам памяти, необходимой для организации хранения ждущих заявок.

Для сбора и вывода таких данных в системе GPSS предусмотрены специальные средства. Проиллюстрируем их применение на примере той же одноканальной СМО, модифицированная блок-схема модели которой показана на рис. 53.

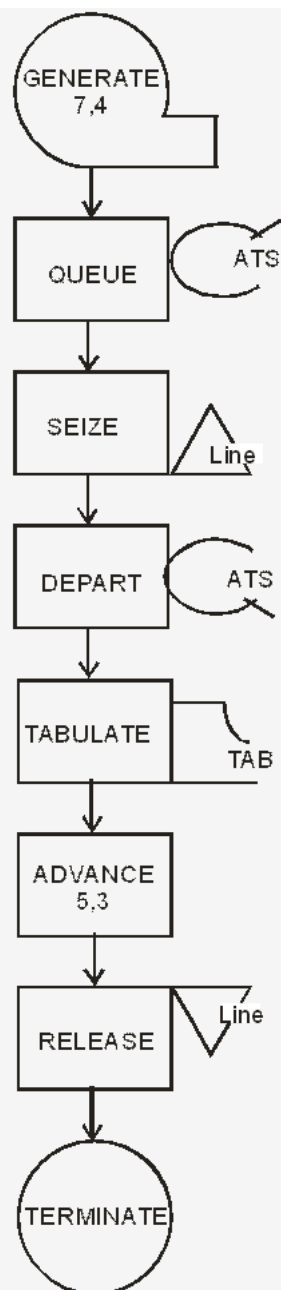


Рис. 53. Модифицированная блок-схема GPSS-модели одноканальной СМО с очередью

Соответствующее описание на языке GPSS имеет вид:

	GENERATE	7,4
TAB	TABLE	QT\$ATS,0,0.25,10
	QUEUE	ATS
	SEIZE	Line
	DEPART	ATS
	TABULATE	TAB
	ADVANCE	5,3
	RELEASE	Line
	TERMINATE	1

В новом варианте модели для сбора статистики появились блоки, с именами QUEUE и DEPART. Блок QUEUE используется для размещения в нем транзактов, которые не могут быть продвинуты к следующему блоку в силу его занятости. Формат QUEUE оператора включает в качестве обязательного операнд с именем очереди (в данном случае ATS). Когда появляются условия для дальнейшего продвижения транзакта, он должен покинуть очередь, что обеспечивается блоком DEPART (обязательный операнд оператор указывает на имя этой очереди).

Система GPSS обеспечивает в случае использования этих блоков автоматическую регистрацию и накопление статистики относительно времени нахождения транзактов в очереди и ее размере. Средние значения этих величин выводятся в стандартном отчете.

Однако система предоставляет средства для сбора более подробной информации в виде гистограмм, аналогичных рассмотренным в разделе, посвященном моделированию по методу Монте-Карло. Для этого необходимо в GPSS-модели описать структуру гистограммы, что реализуется с помощью декларативного оператора (карты) TABLE, и внести в нужных точках модели оператор TABULATE, который обеспечивает автоматическое обновление данных таблицы.

Для данной модели в операторе TABLE указаны:

QTSATS	имя регистрируемого СЧА, в данном случае, среднего времени пребывания в очереди (QT) с именем ATS
0	начало отсчета времени
0.25	интервал (шаг) времени
10	число интервалов

Отчет с результатами моделирования показан на рис. 54.

GPSS World Simulation Report - Q_1.8.1									
Tuesday, January 18, 2011 12:12:09									
START TIME		END TIME		BLOCKS	FACILITIES		STORAGES		
0.000		7049.623		8	1		0		
NAME				VALUE					
ATS				10002.000					
EXPON				10000.000					
LINE				10003.000					
TAB				10001.000					
LABEL	LOC	BLOCK TYPE	ENTRY COUNT	CURRENT COUNT	RETRY				
	1	GENERATE	1001	0	0				
	2	QUEUE	1001	0	0				
	3	SEIZE	1001	1	0				
	4	DEPART	1000	0	0				
	5	ADVANCE	1000	0	0				
	6	RELEASE	1000	0	0				
	7	TABULATE	1000	0	0				
	8	TERMINATE	1000	0	0				
FACILITY LINE	ENTRIES	UTIL.	AVE. TIME	AVAIL.	OWNER	PEND	INTER	RETRY	DELAY
	1001	0.703	4.952	1	1001	0	0	0	0
QUEUE	MAX	CONT.	ENTRY	ENTRY(0)	AVE.CONT.	AVE.TIME	AVE. (-0)	RETRY	
ATS	2	1	1001	672	0.098	0.691	2.101	0	
TABLE	MEAN	STD.DEV.	RANGE		RETRY FREQUENCY		CUM. %		
TAB	0.826	0.161			0				
			-	-	0.000		8	0.80	
			0.000	-	0.250		1	0.90	
			0.250	-	0.500		9	1.80	
			0.500	-	0.750		340	35.80	
			0.750	-	1.000		536	89.40	
			1.000	-	1.250		91	98.50	
			1.250	-	1.500		12	99.70	
			1.500	-	1.750		3	100.00	
CEC XN	PRI	M1	ASSEM	CURRENT	NEXT	PARAMETER	VALUE		
1001	0	7048.997	1001	3	4				
FEC XN	PRI	BDT	ASSEM	CURRENT	NEXT	PARAMETER	VALUE		
1002	0	7057.474	1002	0	1				

Рис. 54. Расширенные результаты моделирования одноканальной СМО с очередью

Теперь помимо средних значений размера очереди (AVE.CONT) и времени нахождения в ней транзактов (AVE.TIME) в строке QUEUE выводится и статистика по частотам (FREQUENCY) и интегральной доле в процентах (CUM%) попаданий длительности нахождения в очереди в соответствующие интервалы (RANGE). Эти данные можно представить более наглядно в виде диаграммы, которую легко построить с помощью имеющихся графических программ (пример такой диаграммы приводился в разделе с описанием метода Монте-Карло).

Выводы:

1. Аналитические модели, применяемые для анализа и проектирования информационных систем, отличает экономичность. Однако для ряда задач на этапе проектирования они не могут обеспечить требуемой точности результатов. Универсальным средством преодолеть эти трудности является методология имитационного моделирования, практически не имеющая ограничений в смысле точности оценок количественных показателей системы, получаемых с ее помощью.
2. В случае, когда задача нахождения значений показателей изучаемого процесса не требует его воспроизведения в динамике и в модель не включаются описания внутренних механизмов протекания процессов, полезным является применение метода статистических испытаний. Его схема отличается простотой, реализация на компьютере не требует специальных средств, однако для сложных задач сопряжена с большим расходом времени на проведение машинных экспериментов.
3. Для исследования стохастических систем применяется имитационное моделирование. Механизм машинной реализации этой методологии основан на агрегатном описании моделируемых процессов, при котором программным образом воспроизводится последовательность смен моделируемой системой своих состояний.

4. Эффективность применения подхода на основе имитационного моделирования к решению сложных задач можно значительно повысить путем использования моделирующих комплексов. Их применение позволяет значительно сократить затраты на создание модели проведение на ней машинных экспериментов.

Вопросы для самопроверки:

1. Что такое компьютерное моделирование, и в каких случаях целесообразно его применять?
2. В чем заключается идея метода статистических испытаний?
3. Когда следует применять метод статистических испытаний?
4. Из каких шагов состоит вычислительная процедура метода статистических испытаний?
5. Поясните применение метода статистических испытаний на примере определения распределения длины критического пути сетевой модели.
6. Чем достигается статистическая точность метода статистических испытаний?
7. Как зависит точность результата, полученного на основе метода статистических испытаний, от числа проведенных испытаний?
8. Каковы основные достоинства и недостатки метода статистических испытаний?
9. Что такое имитационная модель?
10. В чем состоит сущность вычислительной процедуры имитации реальных процессов?
11. Что собой представляет событийный алгоритм имитационного моделирования?
12. Что собой представляет пошаговый алгоритм имитационного моделирования?
13. В чем преимущества и недостатки событийный событийного и пошагового алгоритмов?
14. Поясните на блок-схеме логическую последовательность действий по имитации работы одноканальной СМО с очередью.
15. Из какие основных этапов состоит процесс моделирования?
16. В чем заключается задача этапа структурного анализа процессов?
17. Какие способы используются для формализации содержательного описания модели?
18. Какие способы используются для получения программного кода по формальному описанию модели?
19. Решение каких задач можно автоматизировать при помощи моделирующих комплексов?
20. Для чего предназначена система GPSS?
21. Что входит в элементы языка моделирования системы GPSS?
22. Что такое транзакт?
23. Какую работу выполняет симулятор в системе GPSS?
24. Как определяется модельное время в системе GPSS?
25. Что такое блок-схемы в системе GPSS?
26. Какие списки ведутся в процессе выполнения программной модели в системе GPSS и что в них хранится?
27. Что такое стандартные числовые атрибуты?
28. Как осуществляется генерация случайных чисел в системе GPSS?
29. Что такое оператор в системе GPSS, какие существуют типы операторов?
30. Как выглядит описание (формат) блока в системе GPSS?
31. Какие результаты входят в отчет моделирования в системе GPSS?
32. Какие блоки используются для моделирования процессов в одноканальной СМО с очередью?
33. Какие блоки используются для получения средних величин времени нахождения транзактов в очереди и ее размера?
34. Каким образом можно получить статистику (гистограмму) распределения случайных величин (показателей функционирования)?
35. Какой блок можно использовать для маршрутизации транзактов в модели?
36. Как можно задать вид и параметры распределения временных параметров в операторе GENERATE?

Литература по теме:

Основная литература:

1. Компьютерное моделирование/ ред. А. А. Емельянов. – М.: Маркет ДС, 2010. - 464с.

Дополнительная литература:

1. Т. Дж. Шрайбер, Моделирование на GPSS - М.: Машиностроение, 1980. — 592 с.

Практические задания.

Задание 1.

Используя имеющийся текст программы расчета временных показателей сетевой модели методом Монте-Карло, проведите следующие эксперименты:

- а) Запустите программу с разным числом экспериментов Nexp (1000, 10000, 100000). Проанализируйте результаты.
- б) Запустите программу для случая нормального закона распределения длительностей работ (distr=2). Проанализируйте результаты.
- в) Получите распределение величины критического пути для сетевой модели, представленной на рис. 44. Представьте результат в графическом виде (диаграмма).

Задание 2.

Используя свободно распространяемую программу GPSS World Student (можно установить на свой компьютер, скачав из Интернета) проведите следующие эксперименты:

- а) Используя имеющийся текст GPSS-модели одноканальной СМО с очередью, осуществите запуски с разными значениями параметра команды START (1000, 10000, 100000).
- б) Модифицируйте имеющуюся (или создайте новую) модель СМО, в которой заявки, застающие прибор занятым, покидают систему не обслуженными (подсказка: попробуйте воспользоваться оператором TRANSFER, о котором можно

прочитать в предлагаемой литературе).

Список сокращений

ГСП	Граф состояний и переходов
ДМ	Деловая модель
ДП	Деловой процесс
ИС	Информационная система
ЛПР	Лицо, принимающее решение
СМО	Система массового обслуживания
СЧА	Стандартный числовой атрибут

Глоссарий

- CASE-система** - *Computer-aided software engineering* - система автоматизированной поддержки системного анализа или разработки программного обеспечения информационных систем.
- Архитектура информационной системы** - множество элементов информационной системы и связей между ними.
- Аутсорсинг** - выполнение части функциональных задач предприятия силами подрядных организаций.
- Бизнес-процесс** - см. *Деловой процесс*.
- Декомпозиция** - см. *Структуризация*.
- Деловой процесс** - совокупность регулярно выполняющихся, структурированных и взаимосвязанных действий (работ, задач, операций), направленных на достижение одной или нескольких определенных целей и обеспеченных необходимым набором ресурсов.
- Жизненный цикл** - последовательность стадий, через которые проходит система в промежутке от начала до окончания своего существования.
- Замкнутая система** - система, лишенная способности обмениваться массой, энергией или информацией с окружающей средой.
- Иерархия** - любой согласованный по подчиненности порядок объектов.
- Инновации** - повышение эффективности деловых процессов в результате одного или нескольких радикальных изменений.
- Информационная безопасность** - процесс защиты доступности, конфиденциальности и целостности информации и информационных систем.
- Информационная доступность** - возможность использования информации. Количественной мерой информационной доступности может быть отношение длины интервала времени, когда информация могла использоваться, к общему времени работы системы.
- Масштабируемость** - показатель того, насколько система позволяет осуществлять расширение.
- Окружающая среда** - совокупность всех объектов, изменение свойств которых влияет на систему, а также тех объектов, чьи свойства меняются в результате поведения системы.
- Открытая система** - система, обладающая способностью обмениваться массой, энергией или информацией с окружающей средой.
- Состояние** - совокупность значений наиболее существенных показателей системы.
- Среда** - см. *Окружающая среда*.
- Стохастическая система** - система, поведение которой в некоторый момент в будущем с полной определенностью предсказать невозможно.
- Структура** - отражение определенных взаимосвязей, взаиморасположения составных частей системы.
- Структуризация** - расчленение системы на подсистемы в целях проведения анализа или разработки.
- Улучшение** [деловых процессов] - повышение эффективности деловых процессов путем ряда растянутых во времени небольших усовершенствований.
- Цель** - состояние, которое должно быть достигнуто в результате функционирования системы за определенный промежуток времени.

Приложение

Сведения для работы с пакетом Design/IDEF.

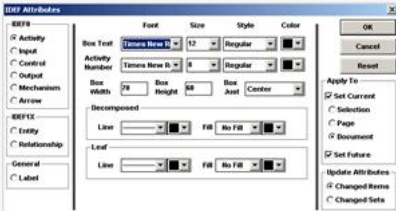
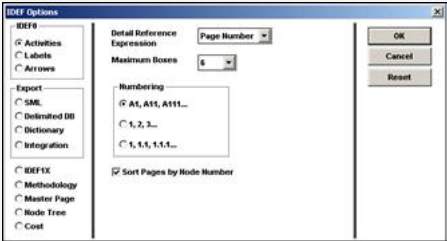
Основные команды.

Меню FILE.

New	Создание новой страницы в проекте или нового проекта. Для создания нового проекта IDEF 0 в появившемся окне Select New Page Type нужно указать IDEF0 .
Open...	Открытие ранее сохраненного проекта.
Close	Закрытие проекта.
Save	Сохранение проекта на диске.
Save as...	Сохранение проекта под новым именем.
Print	Печать листов.
Print Setup	Настройка принтера.
Validate...	Проверка модели на соответствие синтаксису IDEF0. Отчет выводится на экран и (или) в текстовый файл с указанным именем.
Report...	Вывод описания проекта на экран или в файл. В появившемся окне Select Report, в качестве значений параметров Report Type и Destination следует задать

	значения <i>IDEF0</i> (для вывода краткого описания IDEF0- модели) и <i>Screen</i> (для вывода отчета на экран).
Import...	Импорт страницы из файла. При вызове в окне <i>File Import</i> указывается информация о типе и названии файла.
Export...	Экспорт страницы в файл.
Exit	Выход из программы.

Меню EDIT.

Undo	Отмена последнего действия.														
Redo	Возврат отмененного действия.														
Cut	Вырезать активный объект и поместить его в буфер.														
Copy	Копировать активный объект в буфер.														
Paste	Вставить из буфера.														
Cut	Вырезать активный объект и поместить его в буфер														
Delete	Удаление элемента.														
Duplicate	Создание дубликата.														
Copy Page	Копирование страницы.														
Delete Page...	Удаление страницу или ее дочерних.														
Check Spelling	Проверка орфографии.														
Update Number	Перенумерация всех блоков.														
Renumber Box...	Перенумерация указанного блока.														
Set Attributes...	<div>Настройка атрибутов отображаемых элементов</div> <div></div>														
Set Options...	<div>Настройка системных параметров.</div> <div></div> <div>Настройки для IDEF0:</div> <table><tr><td>Activities</td><td>Вид ссылок, максимальное число блоков, которое можно создать на странице, вид нумерации.</td></tr><tr><td>Labels</td><td>Метки.</td></tr><tr><td>Arrows</td><td>Вид стрелок (закругленные и т.д.).</td></tr></table>	Activities	Вид ссылок, максимальное число блоков, которое можно создать на странице, вид нумерации.	Labels	Метки.	Arrows	Вид стрелок (закругленные и т.д.).								
Activities	Вид ссылок, максимальное число блоков, которое можно создать на странице, вид нумерации.														
Labels	Метки.														
Arrows	Вид стрелок (закругленные и т.д.).														
Set Page Attributes...	Настройка атрибутов страницы.														
Set Colors...	Настройка палитры цветов.														
Set User Options...	<div>Настройка внешнего вида среды проектирования.</div> <table><tr><td>View</td><td>Отображение страницы.</td></tr><tr><td>Scroll</td><td>Линейки прокрутки.</td></tr><tr><td>Grid</td><td>Параметры координатной сетки.</td></tr><tr><td>Undo</td><td>Разрешение отмены/повтора выполненных операций.</td></tr><tr><td>File</td><td>Дополнительные настройки отображения.</td></tr><tr><td>Import/Export</td><td>импорта/экспорта файлов.</td></tr><tr><td>Save Settings</td><td>Сохранение настроек.</td></tr></table>	View	Отображение страницы.	Scroll	Линейки прокрутки.	Grid	Параметры координатной сетки.	Undo	Разрешение отмены/повтора выполненных операций.	File	Дополнительные настройки отображения.	Import/Export	импорта/экспорта файлов.	Save Settings	Сохранение настроек.
View	Отображение страницы.														
Scroll	Линейки прокрутки.														
Grid	Параметры координатной сетки.														
Undo	Разрешение отмены/повтора выполненных операций.														
File	Дополнительные настройки отображения.														
Import/Export	импорта/экспорта файлов.														
Save Settings	Сохранение настроек.														
Save Settings...	Сохранение текущих настроек на диске (сохраненные настройки при создании новых проектов будут устанавливаться автоматически).														

Меню CREATE.

Place Boxes...	Создание нескольких блоков на листе. При вызове запрашивается количество блоков. Блоки размещаются равномерно по диагонали от левого верхнего угла до правого нижнего.
----------------	--

IDEF Box	Создание одного блока. После вызова необходимо указать мышью местоположение нового блока.
Arrow	Создание стрелки. После вызова следует указать место на исходящем блоке, откуда должна идти стрелка, и место на входящем блоке.
Label	Создание метки.
Branch	Ветвление имеющихся стрелок. После вызова мышью нужно щелкнуть на стрелке, которую следует разветвить, а затем на блоке, в который должна войти ветвь.
Join	Объединение ветвей. Сначала указывается блок, от которого должна пойти ветвь, затем стрелка, с которой она должна слиться.
Link Arrows	Связывание двух имеющихся стрелок. Перед вызовом следует выбрать одну из стрелок, затем командой Link Arrows указать вторую стрелку.
Tunnel	Создание туннеля.
Attach Label	Привязка метки к стрелке. Перед вызовом процедуры метка должна быть выбрана. После вызова указывается место на стрелке, к которому привяжется метка.
Raise Label	Поднятие метки на один уровень вверх. При вызове выделенная метка перемещается на лист родительской диаграммы.
Create Parent	Создание родительского листа (если его нет).
Decompose	Декомпозиция блока. Перед вызовом декомпозируемый блок должен быть выбран. После декомпозиции двойным щелчком по блоку можно открыть лист дочерней диаграммы.
Push Down	Укрупнение модели, функция обратная декомпозиции. Группа из нескольких выделенных блоков заменяется одним. Новый блок содержит в себе дочернюю диаграмму, в которую переходит указанная группа. При этом все стрелки, связывающие данные блоки друг с другом, перейдут на новую диаграмму, а стрелки, связывающие их с прочими блоками, становятся входными (выходными) для нового блока.
New Page...	Создание новой страницы. При создании запрашивается тип новой страницы.

Меню GLOSSARY.

Содержит функции редактирования глоссария открытого проекта.

Меню MODIFY.

Turn On Text	Включить/выключить режим изменения текста (имеется кнопка на панели инструментов). Если режим включен, то появляется возможность редактировать текст внутри выделенного блока, метки и т.д.
Fit Box to Text	Изменение размеров выделенного блока по размерам текста
Modify Text Box	Изменение местоположения текста, соответствующего выделенному блоку. Эта функция позволяет, например, расположить название блока не внутри по центру, а за его пределами.
Align	Выравнивание выделенной группы объектов. Перед вызовом функции должна быть выделена группа объектов (чаще всего это группа меток), к которым применяется выравнивание. Имеется возможность расположить метки по вертикали, по горизонтали, по диагонали, равномерно между двумя точками на диаграмме и др.
Spread	Распределение объектов внутри группы
Drag	Перетаскивание группы объектов. Перед вызовом перетаскиваемая группа должна быть выделена. Данная функция используется редко, так как режим перетаскивания включен по умолчанию.
Displace	Перенос объекта или группы объектов на некоторое расстояние.
Adjust Size	Настройка размеров. Размеры любого объекта диаграммы можно изменять путем перетаскивания маркеров (черных квадратиков), появляющихся при выделении объекта. Однако, если имеется необходимость изменить размеры объекта, не изменяя положения его центра, то для этого удобнее использовать данную функцию. Она позволяет подстроить размеры как по горизонтали/вертикали, так и по всем направлениям.
Same Size	Задание всем выделенным объектам одних и тех же размеров.
Merge	Слияние текстов.
Move To Page...	Перемещение выделенного объекта на другую страницу.
Refine	Функция, обратная декомпозиции. После вызова декомпозированный блок становится не декомпозированным, а объекты дочерней страницы, связанной с блоком, переносятся на уровень вверх.
Attach	Создание гиперссылок на другие страницы. Гиперссылкой может служить как метка, так и какой-либо графический объект. После связывания появляется возможность двойным щелчком по метке (или графическому объекту) переходить на другую страницу.
Bring To Front	Перенос объекта на передний план (если несколько объектов расположены друг над другом).
Send To Back	Перенос объекта на задний план.

Меню SELECT.

Select One	Выбор одного объекта. После вызова этой функции при перемещении курсора мыши все объекты, над которыми курсор перемещается, начинают мигать.
Form Multiple Selection	Выбор нескольких объектов. После вызова функции требуется нарисовать мышью границу вокруг выделяемых объектов.

Restore Previous Selection	Обновление выбранной группы.
Set Selectability...	Позволяет для выделенного объекта (группы объектов) указать, можно ли этот объект (объекты) выбирать с помощью мыши или нет.
Set Selection Domain...	Позволяет указать, какие типы объектов будут выделяться при групповом выделении: узлы сети, объекты узлов, стрелки или объекты стрелок.
Select All	Выбор всех объектов на странице.
Select All Text	Выбор только текстовых объектов.
Child	Переход на дочернюю страницу.
Parent	Переход на родительскую страницу.
Page...	Выбор страницы по иерархии страниц.
Next	Переход на следующий объект диаграммы (порядок переходов от объекта к объекту совпадает с порядком их создания).
Previous	Переход на предыдущий объект
Next Reference	Переход к следующему объекту, ставшему причиной ошибки во время проверки синтаксиса модели
Previous Reference	Переход к предыдущему ошибочному объекту
Find	Поиск объекта.
Find Next	Поиск следующего вхождения объекта.

Меню VIEW.

Zoom...	Задание масштаба в %.
Zoom To Area	Увеличение области диаграммы.
Fit Page	Вывести на экран страницу целиком.
Fit Objects	Изменение масштаба таким образом, что на экране отображается только область диаграммы.
100%	Установить масштаб один к одному.
Reduce	Уменьшить масштаб в два раза.
Enlarge	Увеличить масштаб в два раза.
Node Tree	Вывод на экран иерархии узлов модели.
Show Page	Просмотр всей страницы.
Redraw Page	Перерисовка страницы.

Меню DICTIONARY.

Команды работы со словарем.

Меню WINDOWS.

Cascade	Каскадное расположение окон.
Tile	Мозаичное расположение окон.

Работа с пакетом.

После запуска программы с помощью команды **New** меню **File** создается **новый проект**. В выпадающем списке поля *Methodology* появившегося окна Select New Page Type следует выбрать IDEF0:

После нажатия кнопки **OK** создается страница верхнего уровня модели. На данной странице находится только один блок, который должен представлять систему в виде единого модуля. Все остальные блоки будут располагаться на страницах более низкого уровня. Название блока должно отражать цель всего проекта.

Чтобы в дальнейшем не устанавливать настройки для каждого созданного блока в отдельности целесообразно **настроить среду проектирования**, для чего в меню **Edit** нужно:

- командой **Set Attributes...** настроить все шрифты, которые будут действовать по умолчанию (из предлагаемого списка шрифтов следует выбирать кириллические);
- командой **Set Options...** установить максимальное количество блоков на странице ([Activities] - Maximum Boxes), тип нумерации ([Activities] - Numbering), кривизну стрелок ([Arrows] - Automating Routing).

В **настройках принтера** (File → Print Setup...) следует установить альбомное расположение листа, а также соответствующий формат.

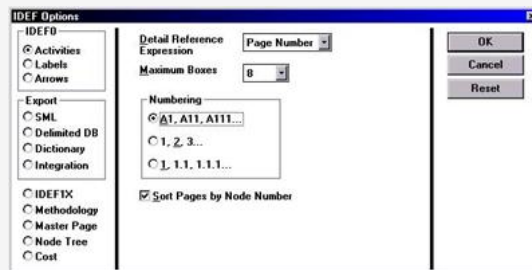


Рис. 55. Задание опций

Соответствующие значения устанавливаются в **настройках атрибутов (Edit → Set Page Attributes...)**. Целесообразно задать и другие настройки командами **Set Colors...**, **Set User Options...**, **Set Settings...** меню **Edit**.

Создание нового блока выполняется командой **IDEF Box** меню **Create** или кнопкой панели инструментов.



После выбора команды или нажатия кнопки следует указать место, где будет расположен новый блок. Если требуется разместить несколько блоков на диаграмме, то можно использовать команду **Place Boxes...** меню **Create** или соответствующую кнопку. После выбора команду или нажатия на кнопку в появившемся окне нужно ввести количество новых блоков. Новые блоки размещаются равномерно по диагонали из левого верхнего в правый нижний угол.

Размеры новых блоков устанавливаются по умолчанию. Размеры каждого блока изменяются путем перетаскивания маркеров, которые появляются после щелчка мыши на блоке.

Добавление текста в блок выполняется командой **Turn On Text** меню **Modify**, нажатием клавиши F2 или кнопки панели инструментов. Режим ввода текста выключается клавишей Esc.

Создание меток выполняется командой **Label** меню **Create**, нажатием клавиши F3 или кнопки панели инструментов. После нажатия следует указать место, где будет введена надпись. Отмена режима ввода надписей происходит при нажатии на клавишу Esc или на любую кнопку на панели инструментов.

Редактировать метки можно включив режим редактирования текста (**Modify → Turn On Text** или нажав клавишу F2 или кнопку на панели инструментов). Выключение режима происходит при нажатии клавиши Esc.

Создание новой стрелки выполняется командой **Arrow** меню **Create** или кнопки панели инструментов. Необходимо затем курсором мыши нажать на точку диаграммы, откуда будет выходить стрелка, и, не отпуская кнопки мыши, протащить стрелку до того места объекта, куда стрелка должна войти.

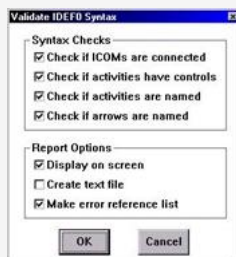
Изменение стрелки на диаграмме производится с помощью маркеров, появляющихся при ее выделении.

Ветвление /объединение стрелок выполняется с помощью команд **Branch/Join** меню **Create** или кнопки панели инструментов. Стрелка, подвергающаяся ветвлению (объединению), должна быть предварительно выделена. После вызова процедуры требуется указать место на любом из объектов диаграммы (объект начинает мигать при приближении курсора мыши), куда (откуда) будет направлена ветвь.

Декомпозиция производится с помощью команды **Decompose** меню **Create** или кнопки панели инструментов. В результате декомпозиции создается новый лист диаграммы более низкого уровня, перейти на который можно двойным щелчком по декомпозированному блоку.

Переход на родительскую или дочернюю страницу выполняется с помощью команды **Parent** и **Child** меню **Select** или кнопки панели инструментов. Для перехода на произвольную страницу может быть использовано иерархическое дерево страниц проекта, которое вызывается командой **Page...** меню **Select**.

Проверка синтаксиса модели выполняется с помощью команды **Validate** меню **File**. После ее вызова появляется окно, где перечислены все параметры, по которым будет вестись проверка.



Результат проверки будет выведен на экран и (или) в текстовый файл.

Печать диаграмм запускается командой **Print...** меню **File**.

Сохранить модель в том же файле можно командой **Save** меню **File**. Сохранение под новым именем или в другом месте выполняется командой **Save As....**

Сохранения избранных листов модели производится командой **Export....** В появившемся окне нужно указать, в каком месте сохранять текущую страницу, имя файла и тип файла (в стандартном виде Design/IDEF виде, в виде графического или текстового файла).