МИНИСТЕРСТВО ВЫСШЕГО И СРЕДНОГО СПЕЦИАЛЬНОГО ОБРАЗОВАНИЯ РЕСПУБЛИКИ УЗБЕКИСТАН
ТАШКЕНТСКИЙ ИНСТИТУТ ТЕКСТИЛЬНОЙ И ЛЁГКОЙ ПРОМЫШЛЕННОСТИ

Кафедра «Автоматизация и управление технологических процессов и производств»

Доц.Халматов Д.А.
Ст.пр.Хушназарова Д.Р.

Курс лекции по курсу

СИСТЕМА АВТОМАТИЗОРОВАННОГО ПРОЕКТИРОВАНИЕ

(для 4-го курса на VIII семестр)

Для студентов бакалавриата по направлению
5311000- Автоматизация и управление технологических процессов и производств

Ташкент–2018

Материалы курс лекции по предмету «Система автоматизированного проектирование» предназначены для изучения основ САПР и изучение пакетов прикладных программ по автоматизации систем проектирование. В курс лекции подробно освещены основные задачи и документации проектирование, состав, классификация и стадии проектирования, системные среды, схемы проектирование технологических процессов, виды компонентов обеспечения, инструментальные средства концептуального проектирование и STEP – технологии.

Курс лекции по предмету «Система автоматизированного проектирование» состоит из 21 лекции и рассчитана на 48 часов и предназначено для бакалавров направление 5311000 – Автоматизация и управление технологических процессов и производств (для хлопковой, текстильной и лёгкой промышленности). Материалы курс лекции подготовлены в соответствии и типовой программы предмета.

Составители: Доц.Халматов Д.А.
Ст.пр. Хушназарова Д.Р.

Рецензенты: PhD по техническим направлением Рузиев У. (ТГТУ)
к.т.н., доц. Каххаров А.А. (ТИТЛП)

Рассмотрено и утверждено на заседании учебно-методического совета ТИТЛП

Протокол № от года

ВВЕДЕНИЕ

Человечество вступило в XXI в., в котором придется решать ряд сложных проблем, связанных с экологией, поиском новых источников энергии, материалов, технологий, соответствующих постиндустриальному обществу. Определяющая роль в решении названных проблем отводится информационным технологиям.

Среди информационных технологий автоматизация проектирования занимает особое место. Во-первых, автоматизация проектирования – синтетическая дисциплина, ее составными частями являются многие другие современные информационные технологии. Так, техническое обеспечение систем автоматизированного проектирования (САПР) основано на использовании вычислительных сетей и телекоммуникационных технологий, в САПР используются персональные компьютеры и рабочие станции, есть примеры применения мейнфреймов. Математическое обеспечение САПР отличается богатством и разнообразием используемых методов вычислительной математики, статистики, математического программирования, дискретной математики, искусственного интеллекта. Программные комплексы САПР относятся к числу наиболее сложных современных программных систем, основанных на операционных системах Unix, Windows-NT, языках программирования C, C++, Java и других, современных CASE-технологиях, реляционных и объектно-ориентированных системах управления базами данных (СУБД), стандартах открытых систем и обмена данными в компьютерных средах.

Во-вторых, знание основ автоматизации проектирования и умение работать со средствами САПР требуются практически любому инженеру-разработчику.

Компьютерами насыщены проектные подразделения, конструкторские бюро и офисы. Работа конструктора за обычным кульманом, расчеты с помощью логарифмической линейки или оформление отчета на пишущей машинке стали анахронизмом. Предприятия, ведущие разработки без САПР или лишь с малой степенью их использования, оказываются неконкурентоспособными вследствие как больших материальных и временных затрат на проектирование, так и невысокого качества проектов.

Появление первых программ для автоматизации проектирования за рубежом ив нашей стране относится к началу 1960-х годов. Тогда были созданы программы для решения задач строительной механики, анализа электронных схем, проектирования печатных плат. Дальнейшее развитие САПР шло по пути создания аппаратных и программных средств машинной графики, повышения вычислительной эффективности программ

моделирования и анализа, расширения областей применения САПР, упрощения пользовательского интерфейса, внедрения в САПР элементов искусственного интеллекта.

К настоящему времени создано большое число программно-методических комплексов для САПР с различными степенью специализации и прикладной ориентацией. В результате автоматизация проектирования стала необходимой составной частью подготовки инженеров разных специальностей; инженер, не владеющий знаниями и не умеющий работать в САПР, не может считаться полноценным специалистом.

Подготовка инженеров разных специальностей в области САПР включает базовую и специальную компоненты. Наиболее общие положения, модели и методики автоматизированного проектирования входят в программу курса, посвященного основам САПР, детальное изучение тех методов и программ, которые специфичны для конкретных специальностей, предусматривается в профильных дисциплинах.

Лекция №1.

Тема: Цель и задачи курса «Система автоматизированного проектирование». Основные понятие о автоматизированных и управляемых систем.

План

1.1. Цель и задачи курса.

1.2. Понятие инженерного проектирование.

1.3. Основные понятие системотехники.

Ключевые слова: проектирование, техническое задание, проект, система автоматизированного проектирование, автоматическое проектирование, автоматизированное проектирование, элемент, сложная система, подсистема, надсистема, структура, параметр, фазовая переменная, состояние, поведение системы, пространство состояний, фазовая траектория, целенаправленность, целостность, иерархичность, моделирование.

1.1. Цель и задачи курса.

В данном предмете приведены сущность, методы, особенности, цель и задачи предмета «Система автоматизированного проектирование». Предмет «Система автоматизированного проектирование» имеет большой значительность в развитии уровень знании квалифицированных кадров в республике. При этом совершенное составление программы предмета имеет значительное место.

Цель изучения предмета «Система автоматизированного проектирование» - заключается в проектировании современных технических систем с помощью множество прикладных программ. Основным значимостью рассматривается CAD проектирование систем, изучение способов решение задач которое осуществляется во время процессов проектирование при построении плоских и пространственных фигур.

Задачей предмета является: работа с прикладными программами на каждом этапе проектирование, применение CAD системы при анализа специальных задач и синтез систем, изучение специальные возможности построение стандартных принципов системы автоматического проектирование, развитие направлений и современное состояние системы автоматического проектирование, иметь понятие о достижении науки и техники в САПР, специальные возможности системы CAD, CAM, CAE, иметь представление о развитие тенденции и состоянии системы

проектирование, знать и применять основные возможности программы Компас, иметь представление развитие системы проектирование.

1.2. Понятие инженерного проектирование.

Проектирование технического объекта - создание, преобразование и представление в принятой форме образа этого еще не существующего объекта. Образ объекта или его составных частей может создаваться в воображении человека в результате творческого процесса или генерироваться в соответствии с некоторыми алгоритмами в процессе взаимодействия человека и ЭВМ. В любом случае инженерное проектирование начинается при наличии выраженной потребности общества в некоторых технических объектах, которыми могут быть объекты строительства, промышленные изделия или процессы. Проектирование включает в себя разработку технического предложения и (или) технического задания (ТЗ), отражающих эти потребности, и реализацию ТЗ в виде проектной документации.

Обычно ТЗ представляют в виде некоторых документов, и оно является исходным (первичным) описанием объекта. Результатом проектирования, как правило, служит полный комплект документации, содержащий достаточные сведения для изготовления объекта в заданных условиях. Эта документация и есть *проект,* точнее, окончательное описание объекта. Более коротко, проектирование - процесс, заключающийся в получении и преобразовании исходного описания объекта в окончательное описание на основе выполнения комплекса работ исследовательского, расчетного и конструкторского характера [1,2].

Преобразование исходного описания в окончательное порождает ряд промежуточных описаний, подводящих итоги решения некоторых задач и используемых при обсуждении и принятии проектных решений для окончания или продолжения проектирования.

Проектирование, при котором все проектные решения или их часть получают путем взаимодействия человека и ЭВМ, называют *автоматизированным,* в отличие от ручного (без использования ЭВМ) или автоматического (без участия человека на промежуточных этапах). Система, реализующая автоматизированное проектирование, представляет собой *систему автоматизированного проектирования* САПР (в англоязычном написании *CAD System - Computer Aided Design System).*

Автоматическое проектирование возможно лишь в отдельных частных случаях для сравнительно несложных объектов. Превалирующим в настоящее время является автоматизированное проектирование.

Проектирование сложных объектов основано на применении идей и принципов, изложенных в ряде теорий и подходов. Наиболее общим подходом является системный подход, идеями которого пронизаны различные методики проектирования сложных систем.

1.3. Основные понятия системотехники

В теории систем и системотехнике введен ряд терминов, среди них к базовым нужно отнести следующие понятия.

Система - множество элементов, находящихся в отношениях и связях между собой.

Элемент - такая часть системы, представление о которой нецелесообразно подвергать при проектировании дальнейшему членению.

Сложная система - система, характеризуемая большим числом элементов и, что наиболее важно, большим числом взаимосвязей элементов. Сложность системы определяется также видом взаимосвязей элементов, свойствами целенаправленности, целостности, членимости, иерархичности, много аспектности. Очевидно, что современные автоматизированные информационные системы и, в частности, САПР являются сложными в силу наличия у них перечисленных свойств и признаков.

Подсистема - часть системы (подмножество элементов и их взаимосвязей), которая имеет свойства системы.

Надсистема - система, по отношению к которой рассматриваемая система является подсистемой.

Структура - отображение совокупности элементов системы и их взаимосвязей; понятие структуры отличается от понятия самой системы также тем, что при описании структуры принимают во внимание лишь типы элементов и связей без конкретизации значений их параметров.

Параметр - величина, выражающая свойство или системы, или ее части, или влияющей на систему среды. Обычно в моделях систем в качестве параметров рассматривают величины, не изменяющиеся в процессе исследования системы. Параметры подразделяют на *внешние, внутренние* и *выходные,* выражающие свойства элементов системы, самой системы, внешней среды соответственно. Векторы внутренних, выходных и внешних параметров далее обозначены $X = (x_1, x_2, ..., x_n)$, $Y = (y_1, y_2, ..., y_j)$, $Q = (q_1, q_2, ..., q_n)$соответственно.

Фазовая переменная - величина, характеризующая энергетическое или информационное наполнение элемента или подсистемы.

Состояние - совокупность значений фазовых переменных, зафиксированных в одной временной точке процесса функционирования.

Поведение (динамика) системы - изменение состояния системы в процессе функционирования.

Система без последействия - ее поведение при $t > tQ$ определяется заданием состояния в момент tQ и вектором внешних воздействий $Q(t)$. В системах с последействием, кроме того, нужно знать предысторию поведения, т.е. состояния системы в моменты, предшествующие /0.

Вектор переменных V, характеризующих состояние (вектор переменных состояния), - не избыточное множество фазовых переменных, задание значений которых в некоторый момент времени полностью определяет поведение системы в дальнейшем (в автономных системах без последействия).

Пространство состояний - множество возможных значений вектора переменных состояния.

Фазовая траектория - представление процесса (зависимости $V(t)$) в виде последовательности точек в пространстве состояний. К характеристикам сложных систем, как сказано выше, часто относят следующие понятия.

Целенаправленность - свойство искусственной системы, выражающее назначение системы. Это свойство необходимо для оценки эффективности вариантов системы.

Целостность - свойство системы, характеризующее взаимосвязанность элементов и наличие зависимости выходных параметров от параметров элементов, при этом большинство выходных параметров не является простым повторением или суммой параметров элементов.

Иерархичность - свойство сложной системы, выражающее возможность и целесообразность ее иерархического описания, т. е. представления в виде нескольких уровней, между компонентами которых имеются отношения целое - часть.

Составными частями системотехники являются следующие основные разделы:

- иерархическая структура систем, организация их проектирования;
- анализ и моделирование систем;
- синтез и оптимизация систем.

Моделирование имеет две четко различимые задачи: 1 - создание моделей сложных систем (в англоязычном написании - modeling); 2 - анализ свойств систем на основе исследования их моделей (simulation).

Синтез также подразделяют на две задачи: 1 - синтез структуры проектируемых систем (структурный синтез); 2 - выбор численных значений параметров элементов систем (параметрический синтез). Эти задачи

относятся к области принятия проектных решений. Моделирование и оптимизацию желательно выполнять с учетом статистической природы систем. Детерминированность - лишь частный случай. При проектировании характерны нехватка достоверных исходных данных, неопределенность условий принятия решений. Учет статистического характера данных при моделировании в значительной мере основан на методе статистических испытаний (методе Монте-Карло), а принятие решений - на использовании нечетких множеств, экспертных систем, эволюционных вычислений [2,3].

Примеры 1. Компьютер является сложной системой в силу наличия у него большого числа элементов, разнообразных связей между элементами и подсистемами, свойств целенаправленности, целостности, иерархичности. К подсистемам компьютера относятся процессор (процессоры), оперативная память, кэш-память, шины, устройства ввода - вывода. В качестве надсистемы могут выступать вычислительная сеть, автоматизированная и (или) организационная система, к которым принадлежит компьютер. Внутренние параметры - времена выполнения арифметических операций, чтения (записи) в накопителях, пропускная способность шин и др. Выходные параметры - производительность компьютера, емкость оперативной и внешней памяти, себестоимость, время наработки на отказ и др. Внешние параметры - напряжение питания сети и его стабильность, температура окружающей среды и др.

Примеры 2. Для двигателя внутреннего сгорания подсистемами являются коленчатый вал, механизм газораспределения, поршневая группа, системы смазывания и охлаждения. Внутренние параметры - число цилиндров, объем камеры сгорания и др. Выходные параметры - мощность двигателя, КПД, расход топлива и др. Внешние параметры характеристики топлива, температура воздуха, нагрузка на выходном валу.

Примеры 3. Подсистемы электронного усилителя - усилительные каскады; внутренние параметры - сопротивления резисторов, емкости конденсаторов, параметры транзисторов; выходные параметры - коэффициент усиления на средних частотах, полоса пропускания, входное сопротивление; внешние параметры - температура окружающей среды, напряжения источников питания, сопротивление нагрузки.

1.4. Жизненный цикл изделий

Жизненный цикл промышленных изделий (ЖЦИ) включает ряд этапов, начиная от зарождения идеи нового продукта до его утилизации по окончании срока использования. Основные этапы жизненного цикла промышленной продукции представлены на рис.1.1. К ним относятся этапы *проектирования*, технологической подготовки производства (ТПП),

собственно производства, реализации продукции, эксплуатации и, наконец, утилизации (в число этапов жизненного цикла могут также входить маркетинг, закупки материалов и комплектующих, предоставление услуг, упаковка и хранение, монтаж и ввод в эксплуатацию).

Рассмотрим содержание основных этапов ЖЦИ для изделий машиностроения.

На этапе проектирования выполняются *проектные процедуры* - формирование принципиального решения, разработка *геометрических моделей* и *чертежей, расчеты, моделирование процессов, оптимизация* и т.п.

На этапе подготовки производства разрабатываются маршрутная и операционная технологии изготовления деталей, реализуемые в программах для станков ЧПУ; технология сборки и монтажа изделий; технология контроля и испытаний.

На этапе производства осуществляются: календарное и оперативное планирование; приобретение материалов и комплектующих с их входным контролем; механообработки и другие требуемые виды обработки; контроль результатов обработки; сборка; испытания и итоговый контроль.

На пост производственных этапах выполняются консервация, упаковка, транспортировка; монтаж у потребителя; эксплуатация, обслуживание, ремонт; утилизация.

На всех этапах жизненного цикла имеются свои целевые установки. При этом участники жизненного цикла стремятся достичь поставленных целей с максимальной эффективностью. На этапах проектирования, ТПП и производства нужно обеспечить выполнение требований, предъявляемых к производимому продукту, при заданной степени надежности изделия и минимизации материальных и временных затрат, что необходимо для достижения успеха в конкурентной борьбе в условиях рыночной экономики. Понятие эффективности охватывает не только снижение себестоимости продукции и сокращение сроков проектирования и производства, но и обеспечение удобства освоения и снижения затрат на будущую эксплуатацию изделий. Особую важность требования удобства эксплуатации имеют для сложной техники, например, в таких отраслях, как авиа- или автомобилестроение [9].

Достижение поставленных целей на современных предприятиях, выпускающих сложные технические изделия, оказывается невозможным без широкого использования *автоматизированных систем* (АС), основанных на применении компьютеров и предназначенных для создания, переработки и использования всей необходимой информации о свойствах изделий и сопровождающих процессов. Специфика задач, решаемых на различных

этапах жизненного цикла изделий, обусловливает разнообразие применяемых АС.

На рис.1.1. указаны основные типы АС с их привязкой к тем или иным этапам жизненного цикла изделий.

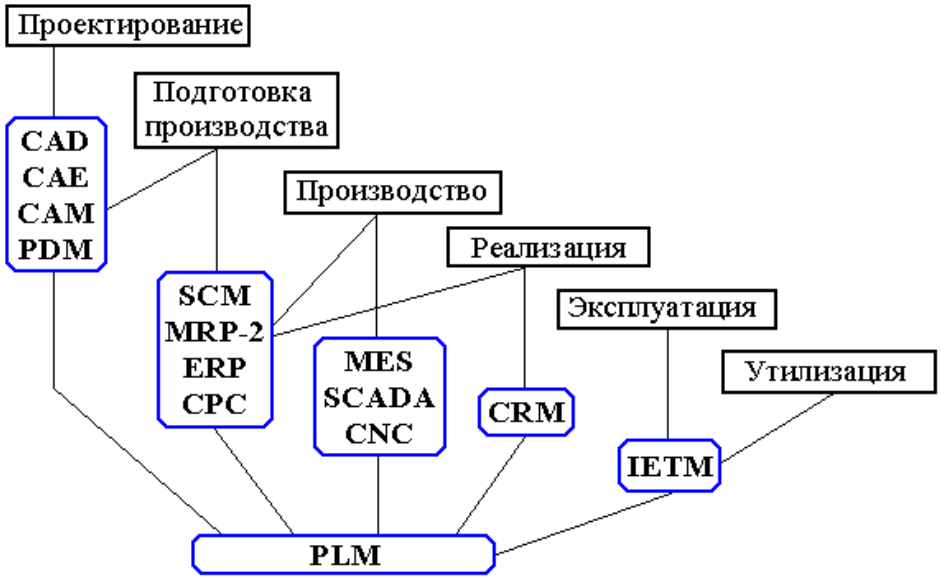


Рис.1.1. Основные типы автоматизированных систем

Автоматизация проектирования осуществляется САПР. В САПР машиностроительных отраслей промышленности принято выделять системы функционального, конструкторского и технологического проектирования. Первые из них называют системами расчетов и инженерного анализа или *системами CAE* (Computer Aided Engineering). *Системы конструкторского проектирования* называют системами CAD (Computer Aided Design). Проектирование технологических процессов выполняется в автоматизированных системах технологической подготовки производства (*АСТПП*), входящих как составная часть в *системы CAM* (Computer Aided Manufacturing).

Для решения проблем совместного функционирования компонентов САПР различного назначения, координации работы систем CAE/CAD/CAM, *управления проектными данными* и проектированием разрабатываются системы, получившие название систем управления проектными данными PDM (Product Data Management). Системы PDM либо входят в состав модулей конкретной САПР, либо имеют самостоятельное значение и могут работать совместно с разными САПР.

На большинстве этапов жизненного цикла, начиная с определения предприятий - поставщиков исходных материалов и компонентов и кончая

реализацией продукции, требуются услуги системы управления цепочками поставок - Supply Chain Management (SCM). Цепь поставок обычно определяют как совокупность стадий увеличения добавленной стоимости продукции при ее движении от компаний-поставщиков к компаниям-потребителям. Управление цепью поставок подразумевает продвижение материального потока с минимальными издержками. При планировании производства система SCM управляет стратегией позиционирования продукции. Если время производственного цикла меньше времени ожидания заказчика на получение готовой продукции, то можно применять стратегию "изготовление на заказ". Иначе приходится использовать стратегию "изготовление на склад". При этом во время производственного цикла должно входить время на размещение и исполнение заказов на необходимые материалы и комплектующие на предприятиях-поставщиках.

В последнее время усилия многих компаний, производящих программно-аппаратные средства автоматизированных систем, направлены на создание систем электронного бизнеса (E-commerce). Задачи, решаемые системами E-commerce, сводятся не только к организации на сайтах Internet витрин товаров и услуг. Они объединяют в едином информационном пространстве запросы заказчиков и данные о возможностях множества организаций, специализирующихся на предоставлении различных услуг и выполнении тех или иных процедур и операций по проектированию, изготовлению, поставкам заказанных изделий. Проектирование непосредственно под заказ позволяет добиться наилучших параметров создаваемой продукции, а оптимальный выбор исполнителей и цепочек поставок ведет к минимизации времени и стоимости выполнения заказа. Координация работы многих предприятий-партнеров с использованием технологий Internet возлагается на системы E-commerce, называемые *системами управления данными в интегрированном информационном пространстве* CPC (Collaborative Product Commerce).

Управление в промышленности, как и в любых *сложных системах*, имеет иерархическую структуру. В общей структуре управления выделяют несколько иерархических уровней, показанных на рис.1.2. Автоматизация управления на различных уровнях реализуется с помощью *автоматизированных систем управления* (АСУ).

Информационная поддержка этапа производства продукции осуществляется *автоматизированными системами управления предприятием* (АСУП) и *автоматизированными системами управления технологическими процессами* (АСУТП).

Рис.1.2. Общая структура управления

К АСУП относятся системы планирования и управления предприятием ERP (Enterprise Resource Planning), планирования производства и требований к материалам *MRP-2* (Manufacturing Requirement Planning) и упомянутые выше системы SCM. Наиболее развитые системы ERP выполняют различные бизнес-функции, связанные с планированием производства, закупками, сбытом продукции, анализом перспектив маркетинга, управлением финансами, персоналом, складским хозяйством, учетом основных фондов и т.п. Системы MRP-2 ориентированы, главным образом, на бизнес-функции, непосредственно связанные с производством. В некоторых случаях системы SCM и MRP-2 входят как подсистемы в ERP, в последнее время их чаще рассматривают как самостоятельные системы.

Промежуточное положение между АСУП и АСУТП занимает производственная исполнительная система *MES* (Manufacturing Execution Systems), предназначенная для решения оперативных задач управления проектированием, производством и маркетингом [5,6].

В состав АСУТП входит *система SCADA* (Supervisory Control and Data Acquisition), выполняющая диспетчерские функции (сбор и обработка данных о состоянии оборудования и технологических процессов) и помогающая разрабатывать ПО для встроенного оборудования. Для непосредственного программного управления технологическим оборудованием используют системы *CNC* (Computer Numerical Control) на

базе *контроллеров* (специализированных компьютеров, называемых промышленными), которые встроены в технологическое оборудование с числовым программным управлением (ЧПУ). Системы CNC называют также *встроенными компьютерными системами.*

На этапе реализации продукции выполняются функции управления отношениями с заказчиками и покупателями, проводится анализ рыночной ситуации, определяются перспективы спроса на планируемые изделия. Эти функции возложены на систему *CRM.*

Функции обучения обслуживающего персонала выполняют *интерактивные электронные технические руководства* IETM (Interactive Electronic Technical Manuals). С их помощью выполняются диагностические операции, поиск отказавших компонентов, заказ дополнительных запасных деталей и некоторые другие операции на этапе эксплуатации систем.

Управление данными в едином информационном пространстве на протяжении всех этапов жизненного цикла изделий возлагается на систему управления жизненным циклов продукции *PLM* (Product Lifecycle Management). Характерная особенность PLM - обеспечение взаимодействия различных автоматизированных систем многих предприятий, т.е. технологии PLM (включая технологии CPC) являются основой, интегрирующей информационное пространство, в котором функционируют САПР, ERP, PDM, SCM, CRM и другие автоматизированные системы многих предприятий.

Контрольные вопросы

1. Дайте определение понятия «проектирование».
2. Назовите признаки, присущие сложной системе.
3. Приведите примеры иерархической структуры технических объектов, их внутренних, внешних и выходных параметров.
4. Приведите примеры условий работоспособности.
5. Почему проектирование обычно имеет итерационный характер? С чем отличается автоматизированное проектирование от автоматического проектирования?
6. На что основано проектирование сложных объектов?
7. Как подразделяется синтез?

Лекция №2.

Тема: Основные понятие о САПР. Принципы создание САПР. Системный подход процесс проектирования. Состав процесс проектирования.

План

2.1. Принципы создания САПР.

2.2. Системный подход процесс проектирования.

2.3. Состав процесс проектирования.

Ключевые слова: системное единства, совместимост, типизация, развития, системный подход, структурный подход, блочно – иерархичный подход, структуризация, итерационный, конструктурский, иерархический уровен.

2.1. Принципы создания САПР.

В технической литературе по САПР времен “перестройки” обычно выделяют 4 принципа создания САПР: принцип *системного единства,* принцип *совместимости*, принцип *типизации* и принцип *развития*.

Принцип системного единства обеспечивает целостность системы и иерархичность проектирования отдельных элементов и всего объекта проектирования.

Принцип совместимости обеспечивает совместное функционирование составных частей САПР и сохраняет открытую систему в целом.

Принцип типизации ориентирует на преимущественное создание и использование типовых и унифицированных элементов САПР. Типизации подлежат элементы, имеющие перспективу многократного применения.

Принцип развития обеспечивает пополнение, совершенствование и обновление составных частей САПР, а также взаимодействие и расширение взаимосвязи с другими автоматизированными системами различного функционального назначения.

Эти принципы определяют и основные особенности САПР, о которых мы сейчас немножко поподробней поговорим.

Во-первых, САПР - это иерархическая система. Она реализует комплексный подход к автоматизации всех уровней проектирования. Блочно-иерархический подход, который обычно используется при проектировании новых изделий, должен быть сохранен и при создании САПР. Иерархия уровней проектирования отражается в структуре программного обеспечения САПР в виде иерархии подсистем. Следует

особо подчеркнуть необходимость обеспечения комплексного характера САПР, так как автоматизация на одном из уровней проектирования при сохранении старых форм проектирования на соседних уровнях оказывается значительно менее эффективной, чем автоматизация всех уровней.

Во-вторых, САПР должна быть совокупностью информационно согласованных модулей. Передача данных от одной программы к другой должна осуществляться без участия человека. Можно заметить, что современные программные системы различного назначения имеют довольно развитую систему экспорта-импорта данных, позволяющую передавать и воспринимать файлы различных форматов. Опыт внедрения САПР в различных отраслях промышленности показал, что если человеку приходится “вручную” перерабатывать информацию, полученную от одной подсистемы для ввода ее в другую, то такие САПР являются малоэффективными. Это отнюдь не означает, что функция человека в САПР должна быть минимизирована. Наоборот, САПР обычно занимается решением сложных, плохо формализуемых задач, которые предполагают активное использование интерактивных методов проектирования[5,6].

Этот факт характеризует третью особенность САПР как человеко-машинной системы. Несмотря на удивительные способности компьютера решать многие сложные задачи, в том числе, и задачи САПР, человек часто по своим эвристическим способностям превосходит самые изощренные системы искусственного интеллекта, поэтому чисто автоматическое проектирование, о котором мы говорили раньше, на практике встречается очень редко. В качестве иллюстрации этого тезиса рассмотрим всем известную систему AUTOCAD американской фирмы AUTODESK Ltd. для автоматизированного проектирования чертежной документации на персональных компьютерах. Эта чисто интерактивная среда проектирования чертежей пользуется большой популярностью у конструкторов, хотя по своей сути представляет собой просто хорошо “автоматизированный кульман”. В “Автокаде” можно реализовать и автоматический метод проектирования, если написать, например, программу формирования какого-либо чертежа на встроенном языке AUTOLISP, однако, эта программа, естественно, не будет универсальной и позволит проектировать только чертежи для одного типа деталей.

Еще одна особенность САПР связана с необходимостью обеспечения в системе свойства открытости, т.е. свойства удобства включения новых методов и средств. Это свойство, естественно, желательно для любого программного продукта, но для САПР просто необходимо, поскольку, как правило, САПР-овские системы делаются долго и достаточно сложны, что

исключает возможность быстрой замены на другую систему (“жалко выбрасывать” целиком, проще что-нибудь добавить или модифицировать).

При разработке САПР следует также помнить, что, несмотря на специализированность системы, в ней надо максимально использовать унифицированные модули. Ясно, что требования универсальности и эффективности взаимно противоречивы: высокоэффективной может быть только специализированная система. Вместе с тем, использование унифицированных модулей расширяет возможности САПР и снижает время на ее разработку, что в условиях “рынка” может явиться определяющим фактором. Известны множество случаев, когда фирмы, сделав свои системы на несколько месяцев раньше своих конкурентов, завоевывали рынок, несмотря на значительно худшее качество своих программных продуктов.

Вообще, умение разрешать различного рода противоречия и находить “золотою середину” - это главное достоинство для разработчика САПР. Естественно, что высокая квалификация как инженера и программиста тоже не помешает, но без глубокого понимания диалектики хороший САПР сделать нельзя. В этом также заключается, на наш взгляд, один из главных принципов создания САПР.

2.2. Системный подход процесс проектирования.

Основные идеи и принципы проектирования сложных систем выражены в системном подходе. Для специалиста в области системотехники они являются очевидными и естественными, однако их соблюдение и реализация зачастую сопряжены с определенными трудностями, обусловливаемыми особенностями проектирования. Как и большинство взрослых образованных людей, правильно использующих родной язык без привлечения правил грамматики, инженеры используют системный подход без обращения к пособиям по системному анализу. Однако интуитивный подход без применения правил системного анализа может оказаться недостаточным для решения все более усложняющихся задач инженерной деятельности.

Основной общий принцип системного подхода заключается в рассмотрении частей явления или сложной системы с учетом их взаимодействия. *Системный подход включает в себя выявление структуры системы, типизацию связей, определение атрибутов, анализ влияния внешней среды.*

Системный подход рассматривают как направление научного познания и социальной политики. Он является базой для обобщающей дисциплины “*Теория систем”* (другое используемое название - “*Системный анализ”).*

Теория систем - дисциплина, в которой конкретизируются положения системного подхода; она посвящена исследованию и проектированию сложных экономических, социальных, технических систем, чаще всего слабоструктурированных. Характерными примерами таких систем являются производственные системы. При проектировании систем цели достигаются в многошаговых процессах принятия решений. Методы принятия решений часто выделяют в самостоятельную дисциплину, называемую "*Теория принятия решений"*.

В технике дисциплину, в которой исследуются сложные технические системы, их проектирование, и аналогичную теории систем, чаще называют *системотехникой. Предметом системотехники являются, во-первых, организация процесса создания, использования и развития технических систем, во-вторых, методы и принципы их проектирования и исследования.* В системотехнике важно уметь сформулировать цели системы и организовать ее рассмотрение с позиций поставленных целей. Тогда можно отбросить лишние и малозначимые части при проектировании и моделировании, перейти к постановке оптимизационных задач.

Системы автоматизированного проектирования и управления относятся к числу наиболее сложных современных искусственных систем. Их проектирование и сопровождение невозможны без системного подхода. Поэтому идеи и положения системотехники входят составной частью в дисциплины, посвященные изучению современных автоматизированных систем и технологий их применения. Интерпретация и конкретизация системного подхода имеют место в ряде известных подходов с другими названиями, которые также можно рассматривать как компоненты системотехники. Таковы структурный, блочно-иерархический, объектно-ориентированный подходы.

При *структурном подходе,* как разновидности системного, требуется синтезировать варианты системы из компонентов (блоков) и оценивать варианты при их частичном переборе с предварительным прогнозированием характеристик компонентов[3,4].

Блочно-иерархический подход к проектированию использует идеи декомпозиции сложных описаний объектов и соответственно средств их создания на иерархические уровни и аспекты, вводит понятие стиля проектирования (восходящее и нисходящее), устанавливает связь между параметрами соседних иерархических уровней.

Ряд важных структурных принципов, используемых при разработке информационных систем и прежде всего их программного обеспечения (ПО), выражен в *объектно-ориентированном подходе* к проектированию (ООП).

Такой подход имеет следующие преимущества в решении проблем управления сложностью и интеграции ПО: 1) вносит в модели приложений большую структурную определенность, распределяя представленные в приложении данные и процедуры между классами объектов; 2) сокращает объем спецификаций, благодаря введению в описания иерархии объектов и отношений наследования между свойствами объектов разных уровней иерархии; 3) уменьшает вероятность искажения данных вследствие ошибочных действий за счет ограничения доступа к определенным категориям данных в объектах. Описание в каждом классе объектов допустимых обращений к ним и принятых форматов сообщений облегчает согласование и интеграцию ПО.

Для всех подходов к проектированию сложных систем характерны также следующие особенности.

1. *Структуризация* процесса проектирования, выражаемая декомпозицией проектных задач и документации, выделением стадий, этапов, проектных процедур. Эта структуризация является сущностью блочно-иерархического подхода к проектированию.

2. *Итерационный* характер проектирования.

Типизация и *унификация* проектных решений и средств проектирования.

2.3. Структура процесса проектирования

Составными структурными частями САПР являются подсистемы, в которых при помощи различных комплексов средств выполняется решение функционально законченных задач в определенной последовательности. Как мы уже определили выше, подсистемы САПР сами обладают всеми свойствами системы, т.е. обычно реализуют вполне законченные этапы или стадии проектирования или группу непосредственно связанных между собой проектных задач.

Проектирующие подсистемы непосредственно выполняют проектные процедуры. Примерами проектирующих подсистем могут служить подсистемы геометрического трехмерного моделирования механических объектов, изготовления конструкторской документации, схемотехнического анализа, трассировки соединений в печатных платах.

Обслуживающие подсистемы обеспечивают функционирование проектирующих подсистем, их совокупность часто называют системной средой (или оболочкой) САПР. Типичными обслуживающими подсистемами являются подсистемы управления проектными данными (PDM - Product Data Management), управления процессом проектирования (DesPM - Design Process Management), пользовательского интерфейса для связи разработчиков

с ЭВМ, CASE (Computer Aided Software Engineering) для разработки и сопровождения программного обеспечения САПР, обучающие подсистемы для освоения пользователями технологий, реализованных в САПР.

Структурирование САПР по различным аспектам обусловливает появление *видов обеспечения* САПР. Принято выделять семь видов обеспечения:

- *техническое* (ТО), включающее различные аппаратные средства (ЭВМ, периферийные устройства, сетевое коммутационное оборудование, линии связи, измерительные средства);
- *математическое* (МО), объединяющее математические методы, модели и алгоритмы для выполнения проектирования;
- *программное* (ПО), представляемое компьютерными программами САПР;
- *информационное* (ИО), состоящее из баз данных (БД), систем управления базами данных (СУБД), а также других данных, используемых при проектировании; отметим, что вся совокупность используемых при проектировании данных называется информационным фондом САПР, а БД вместе с СУБД носит название банка данных (БнД);
- *лингвистическое* (ЛО), выражаемое языками общения между проектировщиками и ЭВМ, языками программирования и языками обмена данными между техническими средствами САПР;
- *методическое* (МетО), включающее различные методики проектирования, иногда к МетО относят также математическое обеспечение;
- *организационное* (ОО), представляемое штатными расписаниями, должностными инструкциями и другими документами, регламентирующими работу проектного предприятия.

Разновидности САПР. Классификацию САПР осуществляют по ряду признаков, например, по приложению, целевому назначению, масштабам (комплексности решаемых задач), характеру базовой подсистемы - ядра САПР.

По *приложениям* наиболее представительными и широко используемыми являются следующие группы САПР.

1. САПР для применения в отраслях общего машиностроения. Их часто называют машиностроительными САПР или MCAD (Mechanical CAD) системами.

2. САПР для радиоэлектроники. Их названия - ECAD (Electronic CAD) или EDA (Electronic Design Automation) системы.

3. САПР в области архитектуры и строительства.

Кроме того, известно большое число более специализированных САПР, или выделяемых в указанных группах, или представляющих самостоятельную ветвь в классификации. Примерами таких систем являются САПР больших интегральных схем (БИС); САПР летательных аппаратов; САПР электрических машин и т.п [9].

По *целевому назначению* различают САПР или подсистемы САПР, обеспечивающие разные аспекты (страты) проектирования. Так, в составе MCAD появляются CAE/CAD/CAM системы:

1. САПР функционального проектирования, иначе САПР-Ф или CAE (Computer Aided Engineering) системы.

2. *конструкторские* САПР общего машиностроения - САПР-К, часто называемые просто CAD системами;

3. *технологические* САПР общего машиностроения - САПР-Т, иначе называемые автоматизированными системами технологической подготовки производства АСТПП или системами CAM (Computer Aided Manufacturing).

По *масштабам* различают отдельные программно-методические комплексы (ПМК) САПР, например, комплекс анализа прочности механических изделий в соответствии с методом конечных элементов (МКЭ) или комплекс анализа электронных схем; системы ПМК; системы с уникальными архитектурами не только программного (software), но и технического (hardware) обеспечений.

По *характеру базовой подсистемы* различают следующие разновидности САПР.

1. САПР *на базе подсистемы машинной графики и геометрического моделирования.* Эти САПР ориентированы на приложения, где основной процедурой проектирования является конструирование, т.е. определение пространственных форм и взаимного расположения объектов. Поэтому к этой группе систем относится большинство графических ядер САПР в области машиностроения.

2. В настоящее время появились унифицированные графические ядра, применяемые более чем в одной САПР, это ядра Parasolid фирмы EDS Unigraphics и ACIS фирмы Intergraph.

Комплексные (интегрированные) САПР, состоящие из совокупности подсистем предыдущих видов. Характерными примерами комплексных САПР являются CAE/CAD/CAM-системы в машиностроении или САПР БИС. Так, САПР БИС включает в себя СУБД и подсистемы проектирования компонентов, принципиальных, логических и функциональных схем, топологии кристаллов, тестов для проверки годности изделий. Для

управления столь сложными системами применяют специализированные *системные среды.*

Иерархическая структура проектных спецификаций и иерархические уровни проектирования. При использовании блочно-иерархического подхода к проектированию представления о проектируемой системе расчленяют на *иерархические уровни.* На верхнем уровне используют наименее детализированное представление, отражающее только самые общие черты и особенности проектируемой системы. На следующих уровнях степень подробности описания возрастает, при этом рассматривают уже отдельные блоки системы, но с учетом воздействий на каждый из них его соседей. Такой подход позволяет на каждом иерархическом уровне формулировать задачи приемлемой сложности, поддающиеся решению с помощью имеющихся средств проектирования. Разбиение на уровни должно быть таким, чтобы документация на блок любого уровня была обозрима и воспринимаема одним человеком.

Другими словами, блочно-иерархический подход есть декомпозиционный подход (его можно назвать также диакоптическим), который основан на разбиении сложной задачи большой размерности на последовательно и (или) параллельно решаемые группы задач малой размерности, что существенно сокращает требования к используемым вычислительным ресурсам или время решения задач.

Можно говорить не только об иерархических уровнях спецификаций, но и об *иерархических уровнях проектирования,* понимая под каждым из них совокупность спецификаций некоторого иерархического уровня совместно с постановками задач, методами получения описаний и решения возникающих проектных задач.

Список иерархических уровней в каждом приложении может быть специфичным, но для большинства приложений характерно следующее наиболее крупное выделение уровней:

- *системный* уровень, на котором решают наиболее общие задачи проектирования систем, машин и процессов; результаты проектирования представляют в виде структурных схем, генеральных планов, схем размещения оборудования, диаграмм потоков данных и т.п.;

- *макроуровень,* на котором проектируют отдельные устройства, узлы машин и приборов; результаты представляют в виде функциональных, принципиальных и кинематических схем, сборочных чертежей и т.п.;

- *микроуровень,* на котором проектируют отдельные детали и элементы машин и приборов.

В каждом приложении число выделяемых уровней и их наименования могут быть различными. Так, в радиоэлектронике микроуровень часто называют компонентным, макроуровень - схемотехническим. Между схемотехническим и системным уровнями вводят уровень, называемый функционально-логическим. В вычислительной технике системный уровень подразделяют на уровни проектирования ЭВМ (вычислительных систем) и вычислительных сетей. В машиностроении имеются уровни деталей, узлов, машин, комплексов.

В зависимости от последовательности решения задач иерархических уровней различают нисходящее, восходящее и смешанное проектирование (стили проектирования). Последовательность решения задач от нижних уровней к верхним характеризует *восходящее* проектирование, обратная последовательность приводит к *нисходящему* проектированию, в *смешанном* стиле имеются элементы как восходящего, так и нисходящего проектирования. В большинстве случаев для сложных систем предпочитают нисходящее проектирование. Отметим однако, что при наличии заранее спроектированных составных блоков (устройств) можно говорить о смешанном проектировании.

Неопределенность и нечеткость исходных данных при нисходящем проектировании (так как еще не спроектированы компоненты) или исходных требований при восходящем проектировании (поскольку ТЗ имеется на всю систему, а не на ее части) обусловливают необходимость прогнозирования недостающих данных с последующим их уточнением, т.е. последовательного приближения к окончательному решению *(итерационность* проектирования).

Наряду с декомпозицией описаний на иерархические уровни применяют разделение представлений о проектируемых объектах на аспекты.

Аспект описания (страта) - описание системы или ее части с некоторой оговоренной точки зрения, определяемой функциональными, физическими или иного типа отношениями между свойствами и элементами.

Различают аспекты функциональный, информационный, структурный и поведенческий (процессный). *Функционалъное* - описание относят к функциям системы и чаще всего представляют его функциональными схемами. *Информационное* описание включает в себя основные понятия предметной области (сущности), словесное пояснение или числовые значения характеристик (атрибутов) используемых объектов, а также описание связей между этими понятиями и характеристиками. Информационные модели можно представлять графически (графы, диаграммы сущность-отношение), в виде таблиц или списков. *Структурное*

описание относится к морфологии системы, характеризует составные части системы и их межсоединения и может быть представлено структурными схемами, а также различного рода конструкторской документацией. *Поведенческое* - описание характеризует процессы функционирования (алгоритмы) системы и (или) технологические процессы создания системы. Иногда аспекты описаний связывают с подсистемами, функционирование которых основано на различных физических процессах.

Отметим, что в общем случае выделение страт может быть неоднозначным. Так, помимо указанного подхода. очевидна целесообразность выделения таких аспектов, как *функционалъное* (разработка принципов действия, структурных, функциональных, принципиальных схем), *конструкторское* (определение форм и пространственного расположения компонентов изделий), *алгоритмическое* (разработка алгоритмов и программного обеспечения) и *технологическое* (разработка технологических процессов) проектирование систем. Примерами страт в случае САПР могут служить также рассматриваемые далее виды обеспечения автоматизированного проектирования.

Контрольные вопросы

1. Какие принципы создания существует в САПР?
2. Что обеспечивает принцип системного единства?
3. Какие элементы подлежать к принципу типизацию?
4. Объясните системный подход проектирования.
5. Как описывается объектно-ориентированный подход?
6. Какие особенности характерны к проектированию сложных систем?
7. Какие подсистемы выполняют проектные процедуры?
8. Как структурируется САПР по различным аспектам?
9. Какие разновидности имеет САПР по характеру базавой подсистемы?
10. Как характеризуется последовательность уровней?

Лекция № 3.
Тема: Основные задачи и основные документации проектирования. Классификация и стадия САПР.

План

3.1. Классификация САПР.
3.2. Стадия САПР.
3.3. Типовые проектные процедуры.
3.4. Комплексные автоматизированные системы.

Ключевые слова: уровень, сложность, сервер, стадии проектирования, процедура, техническое задание, работоспособность, параметр, модель, макромодел, структурный синтез, параметрический синтез, комплексные системы, маршрутизация.

3.1. Классификация САПР.

Основные критерии выбора САПР:

1) функциональные возможности;
2) ·наличие уникальных функций;
3) стоимость;
4) простота интерфейса и легкость обучения

По степени сложности объекта проектирования САПР делятся: *простые* - до100 составных частей, *средней сложности* - от 100 до 1000, *сложные* - от 1000 до 10000, *очень сложные* - свыше 10000 составных частей.

По уровню автоматизации проектирования САПР делятся: *низко автоматизированные* (до 25% проектных работ автоматизировано); *средне автоматизированные* (25%-50%); *высоко автоматизированные* (свыше 50%).

По комплексности автоматизации, под которой понимается число выполняемых этапов проектирования, САПР делятся на: *легкие, средние**,*** *тяжелые.*

Рис.3.1. Классификация САПР приборостроения

Легкие (одноэтапные) - выполняется один этап проектирования, например разводка печатных плат, создание сборочного чертежа.

Примером легких САПР являются: **AutoCAD, Компас, P-CAD).** Они применяются для выполнения почти всех работ с двумерными чертежами и имеют ограниченный набор функций по трехмерному моделированию. С помощью этих систем выполняются порядка 90% всех работ по проектированию. Имеющиеся ограничения делают их не всегда довольно

удобными. Типичная легкая САПР должна решать следующие задачи: производить геометрические построения; выполнять стандартное нанесение размеров; выполнять3-хмерное моделирование; иметь возможность работы с библиотекой графических и текстовых объектов; работа с технической документацией [5,6].

Средние (многоэтапные) САПР. Занимают промежуточное положение между тяжелыми и легкими САПР, но при этом позволяющие выполнять 90% всех функций тяжелых, а по стоимости близки к легким.

Тяжелые САПР (комплексные) выполняется несколько этапов проектирования, например, оформляется полная конструкторская документация на изготовление.

Эти системы, которые, во-первых, обеспечивают весь цикл создания изделия от концептуальной идеи до реализации, а во-вторых(и это самое главное), создают проектно-технологическую среду для одновременной работы всех участников создания изделия с единой виртуальной электронной моделью этого изделия. Эти САПР объединяют систем CAD/CAM/CAE, наиболее громоздки и сложны в работе, имеют значительную стоимость. Системы применяются для решения наиболее трудоемких задач - моделирования поведения сложных систем в реальном масштабе времени, оптимизации расчетов с визуализацией результатов. В состав системы входят как графические, так и модули для проведения расчетов и моделирования.

Количество уровней технического обеспечения. Одноуровневые САПР устанавливаются на одной ЭВМ и образуют автономное ***автоматизированное рабочее место*** **(АРМ)** - программно-технический комплекс, предназначенный для автоматизации деятельности определенного вида. АРМ объединяет программно-аппаратные средства, обеспечивающие взаимодействие человека с компьютером.

Двухуровневые системы организуются для сложных, многоэтапных САПР. На АРМ выполняются отдельные этапы проектирования. АРМы объединены в сеть и работают под управлением сервера.

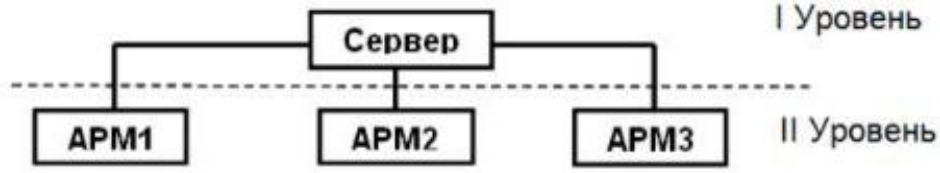


Рис.3.2. Двухуровневая САПР

Роль сервера выполняет мощная ЭВМ с большим быстродействием. Сервер выполняет следующие функции:

- ·управляет процессом проектирования;
- ·решает наиболее сложные задачи с большим объемом вычислений;
- ·содержит базы данных и выполняет поиск по запросам, поступающим со второго уровня, т. е. работает в архитектуре "Клиент - сервер".

Трехуровневая система используется для очень сложных САПР и образуется при объединении двухуровневых САПР, причем на верхнем уровне используется мощная мейнфреймовая ЭВМ.

Сейчас на базе двух- и трехуровневых систем начинают функционировать, так называемые, ***виртуальные подразделения.*** Конструкторы, расчетчики и технологи могут обмениваться информацией на основе удаленного доступа через Интернет, т. е. пользователи географически разделены.

3.2. Стадии САПР

Стадии проектирования. *Стадии* проектирования - наиболее крупные части проектирования, как процесса, развивающегося во времени. В общем случае выделяют стадии научно-исследовательских работ (НИР), эскизного проекта или опытно-конструкторских работ (ОКР), технического, рабочего проектов, испытаний опытных образцов или опытных партий. Стадию НИР иногда называют пред проектными исследованиями или стадией технического предложения. Очевидно, что по мере перехода от стадии к стадии степень подробности и тщательность проработки проекта возрастают, и рабочий проект уже должен быть вполне достаточным для изготовления опытных или серийных образцов. Близким к определению стадии, но менее четко оговоренным понятием, является понятие этапа проектирования.

Стадии (этапы) проектирования подразделяют на составные части, называемые *проектными процедурами*. Примерами проектных процедур могут служить подготовка деталировочных чертежей, анализ кинематики, моделирование переходного процесса, оптимизация параметров и другие проектные задачи. В свою очередь, проектные процедуры можно расчленить на более мелкие компоненты, называемые *проектными операциями*, например, при анализе прочности детали сеточными методами операциями могут быть построение сетки, выбор или расчет внешних воздействий, собственно моделирование полей напряжений и деформаций, представление результатов моделирования в графической и текстовой формах. Проектирование сводится к выполнению некоторых последовательностей проектных процедур - *маршрутов проектирования.*

Иногда разработку ТЗ на проектирование называют *внешним* проектированием, а реализацию ТЗ – *внутренним* проектированием.

Содержание технических заданий на проектирование. В ТЗ на проектирование объекта указывают, по крайней мере, следующие данные.

1. Назначение объекта.

2. Условия эксплуатации. Наряду с качественными характеристиками (представленными в вербальной форме) имеются числовые параметры, называемые *внешними* параметрами, для которых указаны области допустимых значений. Примеры внешних параметров: температура окружающей среды, внешние силы, электрические напряжения, нагрузки и т.п.

3. Требования к *выходным* параметрам, т.е. к величинам, характеризующим свойства объекта, интересующие потребителя. Эти требования выражены в виде *условий работоспособности*

$$y_i R T_i$$

где $y_i - i$ - й выходной параметр, $R \in \{=, <, >, \geq, \leq\}$, (равно, меньше, больше, больше или равно, меньше или равно) - вид отношения; T_i - норма i - го выходного параметра. В случае R = "равно" нужно задать требуемую точность выполнения равенства.

Классификация моделей и параметров, используемых при автоматизированном проектировании. В автоматизированных проектных процедурах вместо еще не существующего проектируемого объекта оперируют некоторым квазиобъектом - *моделью,* которая отражает некоторые интересующие исследователя свойства объекта. Модель может быть физическим объектом (макет, стенд) или спецификацией. Среди моделей-спецификаций различают упомянутые выше функциональные, поведенческие, информационные, структурные модели (описания). Эти модели называют *математическими* - если они формализованы средствами аппарата и языка математики [9].

В свою очередь, математические модели могут быть геометрическими, топологическими, динамическими, логическими и т.п., если они отражают соответствующие свойства объектов. Наряду с математическими моделями при проектировании используют рассматриваемые ниже функциональные ГОББ0-модели, информационные модели в виде диаграмм сущность-отношение, геометрические модели-чертежи. В дальнейшем, если нет специальной оговорки, под словом "модель" будем подразумевать математическую модель (МО).

Математическая функциональная модель в общем случае представляет собой алгоритм вычисления вектора выходных параметров **Y** при заданных векторах параметров элементов **X** и внешних параметров **Q**.

Математические модели могут быть символическими и численными. При использовании *символических* моделей оперируют не значениями величин, а их символическими обозначениями (идентификаторами). *Численные* модели могут быть *аналитическими,* т.е. их можно представить в виде явно выраженных зависимостей выходных параметров **Y** от параметров

внутренних **X** и внешних **Q**, или *алгоритмическими,* в которых связь **Y, X** и **Q** задана неявно в виде алгоритма моделирования. Важнейший частный случай алгоритмических моделей - *имитационные*, они отображают процессы в системе при наличии внешних воздействий на систему. Другими словами, имитационная модель - это алгоритмическая поведенческая модель.

Классификацию математических моделей выполняют также по ряду других признаков.

Так, в зависимости от принадлежности к тому или иному иерархическому уровню выделяют модели уровней системного, функционально-логического, макроуровня (сосредоточенного) и микроуровня (распределенного).

По характеру используемого для описания математического аппарата различают модели лингвистические, теоретико-множественные, абстрактно-алгебраические, нечеткие, автоматные и т.п.

Например, на системном уровне преимущественно применяют модели систем массового обслуживания и сети Петри, на функционально-логическом уровне - автоматные модели на основе аппарата передаточных функций или конечных автоматов, на макроуровне - системы алгебро-дифференциальных уравнений, на микроуровне - дифференциальные уравнения в частных производных. Особое место занимают геометрические модели, используемые в системах конструирования.

Кроме того, введены понятия полных моделей и макромоделей, моделей статических и динамических, детерминированных и стохастических, аналоговых и дискретных, символических и численных.

Полная модель объекта в отличие от *макромодели* описывает не только процессы на внешних выводах моделируемого объекта, но и внутренние для объекта процессы.

Статические модели описывают статические состояния, в них не присутствует время в качестве независимой переменной. *Динамические* модели отражают поведение системы, т.е. в них обязательно используется время.

Стохастические и *детерминированные* модели различаются в зависимости от учета или не учета случайных факторов.

В *аналоговых* моделях фазовые переменные - непрерывные величины, в *дискретных* - дискретные, в частном случае дискретные модели являются *логическими (булевыми),* в них состояние системы и ее элементов описывается булевыми величинами. В ряде случаев полезно применение *смешанных* - моделей, в которых одна часть подсистем характеризуется аналоговыми моделями, другая - логическими.

Информационные - модели относятся к информационной страте автоматизированных систем, их используют прежде всего при инфологическом проектировании баз данных (БД) для описания связей между единицами информации.

Наибольшие трудности возникают при создании моделей слабоструктурированных систем, что характерно прежде всего для системного уровня проектирования. Здесь значительное внимание уделяется экспертным методам. В теории систем сформулированы общие рекомендации по подбору экспертов при разработке модели, организации экспертизы, по обработке полученных результатов. Достаточно общий подход к построению моделей сложных слабоструктурированных систем выражен в методиках IDEF.

Обычно в имитационных моделях фигурируют фазовые переменные. Так, на макроуровне имитационные модели представляют собой системы алгебро-дифференциальных уравнений

$$\mathbf{\Phi}(d\mathbf{V}/dt, \mathbf{V}, t) = 0 \text{ при } t = 0, \mathbf{V} = \mathbf{V}_0, \qquad (3.1)$$

где V - вектор фазовых переменных; t - время; V_0 - вектор начальных условий. К примерам фазовых переменных можно отнести токи и напряжения в электрических системах, силы и скорости - в механических, давления и расходы - в гидравлических.

Выходные параметры систем могут быть двух типов. Во-первых, это *параметры-функционалы,* т.е. функционалы зависимостей *V(t)* в случае использования (3.1). Примеры таких параметров: амплитуды сигналов, временные задержки, мощности рассеивания и т.п. Во-вторых, это параметры, характеризующие способность проектируемого объекта работать при определенных внешних условиях. Эти выходные параметры являются граничными значениями диапазонов внешних переменных, в которых сохраняется работоспособность объекта.

3.3.Типовые проектные процедуры.

Создать проект объекта (изделия или процесса) означает выбрать структуру объекта, определить значения всех его параметров и представить результаты в установленной форме. Результаты (проектная документация) могут быть выражены в виде чертежей, схем, пояснительных записок, программ для программно-управляемого технологического оборудования и других документов на бумаге или на машинных носителях информации.

Разработка (или выбор) структуры объекта есть проектная процедура, называемая *структурным синтезом,* а расчет (или выбор) значений параметров элементов **X** - процедура *параметрического синтеза.*

Задача структурного синтеза формулируется в системотехнике как *задача принятия решений* (ЗПР). Ее суть заключается в определении цели, множества возможных решений и ограничивающих условий.

Классификацию ЗПР осуществляют по ряду признаков. По числу критериев различают задачи одно- и многокритериальные. По степени неопределенности различают ЗПР детерминированные, ЗПР в условиях риска - при наличии в формулировке задачи случайных параметров, ЗПР в

условиях неопределенности, т.е. при неполноте или недостоверности исходной информации.

Реальные задачи проектирования, как правило, являются многокритериальными. Одна из основных проблем постановки многокритериальных задач - установление правил предпочтения вариантов. Способы сведения многокритериальных задач к однокритериальным и последующие пути решения изучаются в дисциплинах, посвященных методам оптимизации и математическому программированию.

Наличие случайных факторов усложняет решение ЗПР. Основные подходы к решению ЗПР в условиях риска заключаются или в решении “для наихудшего случая”, или в учете в целевой функции математического ожидания и дисперсии выходных параметров. В первом случае задачу решают как детерминированную при завышенных требованиях к качеству решения, что является главным недостатком подхода. Во втором случае достоверность результатов решения намного выше, но возникают трудности с оценкой целевой функции. Применение метода Монте-Карло в случае алгоритмических моделей становится единственной альтернативой и, следовательно, для решения требуются значительные вычислительные ресурсы [4,5].

Существуют две группы ЗПР в условиях неопределенности. Одна из них решается при наличии противодействия разумного противника. Такие задачи изучаются в *теории игр,* для задач проектирования в технике они не характерны. Во второй группе достижению цели противодействие оказывают силы природы. Для их решения полезно использовать теорию и методы *нечетких множеств.*

Например, при синтезе структуры автоматизированной системы постановка задачи должна включать в качестве исходных данных следующие сведения:

- множество выполняемых системой функций (другими словами, множество работ, каждая из которых может состоять из одной или более операций); возможно, что в этом множестве имеется частичная упорядоченность работ, что может быть представлено в виде ориентированного графа, в котором вершины соответствуют работам, а дуги - отношениям порядка;

- типы допустимых для использования серверов (машин), выполняющих функции системы;

- множество внешних источников и потребителей информации;

- во многих случаях задается также некоторая исходная структура системы в виде взаимосвязанной совокупности серверов определенных типов; эта структура может рассматриваться как обобщенная избыточная или как вариант первого приближения;

-различного рода ограничения, в частности, ограничения на затраты

материальных ресурсов и (или) на времена выполнения функций системы.

Задача заключается в синтезе (или коррекции) структуры, определении типов серверов (программно-аппаратных средств), распределении функций по серверам таким образом, чтобы достигался экстремум целевой функции при выполнении заданных ограничений.

Конструирование, разработка технологических процессов, оформление проектной документации - частные случаи структурного синтеза.

Задачу параметрического синтеза называют параметрической *оптимизацией* (или оптимизацией), если ее решают как задачу математического программирования и т.е.

$$\mathbf{extr}\ F(\mathbf{X}), \mathbf{X} \in \mathbf{D}_x,$$

где *F(X)* - целевая функция; *X* - вектор управляемых (называемых также проектными или варьируемыми) параметров; $\mathbf{D}_x = \{\mathbf{X} \mid \varphi(\mathbf{X}) < 0, \psi(\mathbf{X}) = 0\}$ - допустимая область; $\varphi(\mathbf{X})$ и $\psi(\mathbf{X})$ - функции-ограничения.

3.4. Комплексные автоматизированные системы.

Известно, что частичная автоматизация зачастую не дает ожидаемого повышения эффективности функционирования предприятий. Поэтому предпочтительным является внедрение интегрированных САПР, автоматизирующих все основные этапы проектирования изделий. Дальнейшее повышение эффективности производства и повышение конкурентоспособности выпускаемой продукции возможно за счет интеграции систем проектирования, управления и документооборота.

Такая интеграция лежит в основе создания *комплексных систем автоматизации,* в которых помимо функций собственно САПР реализуются средства для автоматизации функций управления проектированием, документооборота, планирования производства, учета и т. п.

Проблемы интеграции лежат в основе технологии Юпитер, пропагандируемой фирмой Intergraph. Пример сращивания некоторых подсистем из САПР и АСУ - программный продукт TechnoDOCS (российская фирма Весть). Его функции:

- интеграция программ документооборота с проектирующими пакетами (конкретно с AutoCAD, Microstation и другими программами, исполняемыми в Windows-средах и поддерживающими взаимодействие по технологиям DDE или OLE, разработанным фирмой Microsoft);
- ведение архива технической документации;
- маршрутизация работ и прохождение документации, контроль исполнения;
- управление параллельным проектированием, т.е. координацией проектных работ, выполняемых коллективно.

Очевидно, что подобная интеграция является неотъемлемой чертой CALS-систем. В основу

CALS-технологии положен ряд стандартов и прежде всего это стандарты STEP, а также Parts Library, Mandate, SGML (Standard Generalized Markup Language), EDIFACT (Electronic Data Interchange For Administration, Commerse, Transport) и др. Стандарт SGML устанавливает способы унифицированного оформления документов определенного назначения - отчетов, каталогов, бюллетеней и т.п., а стандарт EDIFACT - способы обмена подобными документами.

Одна из наиболее известных реализаций CALS-технологии разработана фирмой Computervision. Это технология названа EPD (Electronic Product Definition) и ориентирована на поддержку процессов проектирования и эксплуатации изделий машиностроения.

В CALS-системах на всех этапах жизненного цикла изделий используется документация, полученная на этапе проектирования. Поэтому естественно, что составы подсистем в CALS и комплексных САПР в значительной мере совпадают.

Технологию EPD реализуют:

- CAD - система автоматизированного проектирования;
- CAM - автоматизированная система технологической подготовки производства (АСТПП);
- CAE - система моделирования и расчетов;
- CAPE (Concurrent Art-to-Product Environoment) - система поддержки параллельного проектирования (concurrent engineering);
- PDM - система управления проектными данными, представляющая собой специализированную СУБД (DBMS - Data Base Management System);
- *3D* Viewer -система трехмерной визуализации;
- CADD - система документирования;
- CASE - система разработки и сопровождения программного обеспечения;
- методики обследования и анализа функционирования предприятий.

Основу EPD составляют системы CAD и PDM, в качестве которых используются CADDS5 и Optegra соответственно.

В значительной мере специфику EPD определяет система Optegra. В ней отображается иерархическая структура изделий, включающая все сборочные узлы и детали. В Optegra можно получить информацию об атрибутах любого элемента структуры, а также ответы на типичные для баз данных вопросы типа “Укажите детали из материала *P"* или “В каких блоках используются детали изготовителя Y?” и т.п.

Важной для пользователей особенностью Optegra является работа вместе с многооконной системой визуализации 3D Viewer. Пользователь

может одновременно следить за информацией в нескольких типовых окнах:

- информационный браузер, в котором высвечиваются данные, запрашиваемые пользователем, например, из почтового ящика, Internet, корпоративных ресурсов, его персональной БД;
- окно структуры изделия, представляемой в виде дерева. Можно получать ответы на запросы подсветкой деталей *Dj* (листьев дерева), удовлетворяющих условиям запроса;
- 3D визуализатор, в этом окне высвечивается трехмерное изображение изделия, ответы на запросы даются и в этом окне цветовым выделением деталей D_j;
- окно пользовательского процесса, в котором в нужной последовательности в виде иконок отображается перечень задач, заданный пользователю для решения.

В системе Optegra связи между объектами задаются по протоколам стандартов STEP, внешний интерфейс осуществляется через базу данных SDAI.

Системы управления в составе комплексных автоматизированных систем. Системы управления в промышленности, как и любые сложные системы, имеют иерархическую структуру. Если рассматривать предприятие как систему верхнего уровня, то следующими уровнями по нисходящей линии будут уровни завода, цеха, производственного участка, производственного оборудования. Автоматизация управления реализуется с помощью автоматизированных систем управления (АСУ).

Среди АСУ различают *автоматизированные системы управления предприятием* (АСУП) и *автоматизированные системы управления технологическими процессами* (АСУТП). АСУП охватывает уровни от предприятия до цеха, АСУТП - от цеха и ниже (на уровне цеха могут быть средства и АСУП, и АСУТП).

В АСУП выделяют подсистемы, выполняющие определенные функции, типичными среди них являются:

- календарное планирование производства, потребностей в мощностях и материалах;
- оперативное управление производством;
- сетевое планирование проектов;
- управление проектированием изделий;
- учет и нормирование трудозатрат;
- учет основных фондов;
- управление финансами;
- управление запасами (складским хозяйством);
- управление снабжением (статистика закупок, контракты на закупку);

• маркетинг (статистика и анализ реализации, контракты на реализацию, прогноз, реклама).

Процедуры, выполняющие эти функции, часто называют *бизнес-функциями,* а маршруты решения задач управления, состоящие из бизнес-функций, называют *бизнес-процессами.*

3.5. Этапы проектирования

К проектированию АС непосредственное отношение имеют два направления деятельности: 1) собственно проектирование АС конкретных предприятий (отраслей) на базе готовых программных и аппаратных компонентов с помощью специальных инструментальных средств разработки; 2) проектирование упомянутых компонентов АС и инструментальных средств, ориентированных на многократное применение при разработке многих конкретных автоматизированных систем.

Сущность первого направления можно охарактеризовать словами *«системная интеграция*» (другое близкое понятие имеет название *консалтинг*). Разработчик АС должен быть специалистом в области системотехники, хорошо знать соответствующие международные стандарты, состояние и тенденции развития информационных технологий и программных продуктов, владеть инструментальными средствами разработки приложений (CASE-средствами) и быть готовым к восприятию и анализу автоматизируемых процессов в сотрудничестве со специалистами-прикладниками [1,2].

Существует ряд фирм, специализирующихся на разработке проектов АС (например, Price Waterhouse, Jet Info, Consistent Software, Interface и др.)

Второе направление в большей мере относится к области разработки МО и ПО для реализации функций АС - моделей, методов, алгоритмов, программ на базе знания системотехники, методов анализа и синтеза проектных решений, технологий программирования, операционных систем и т. п. Существует ряд общеизвестных технологий (методик) проектирования ПО АС, среди которых прежде всего следует назвать компонентно-ориентированную разработку - технологию индустриальной разработки программных систем.

Для каждого класса АС (САПР, ERP, геоинформационные системы и т. д.) можно указать фирмы, специализирующиеся на разработке программных (а иногда и программно-аппаратных) систем. Многие из них на основе одной из базовых технологий реализуют свой подход к созданию АС и придерживаются стратегии либо тотального поставщика, либо открытости и расширения системы приложениями и дополнениями третьих фирм.

Существует и международный стандарт на стадии жизненного цикла программной продукции (ISO 12207:1995). Как собственно АС, так и компоненты АС являются сложными системами, и при их проектировании нужно использовать один из стилей проектирования:

- *нисходящее* (*Top-of-Design)*\ четкая реализация нисходящего проектирования приводит к *спиральной модели* разработки ПО, на каждом витке спирали блоки предыдущего уровня детализируются, используются обратные связи (альтернативой является так называемая *каскадная модель,* относящаяся к поочередной реализации частей системы);
- *восходящее* (*Bottom-of-Design*);
- *эволюционное* (*Middle-of-Design).*

Чаще всего применяют нисходящий стиль блочно-иерархического проектирования.

Рассмотрим этапы нисходящего проектирования АС.

Верхний уровень проектирования АС часто называют *концептуальным* проектированием. Концептуальное проектирование выполняют в процессе пред проектных исследований, формулировки ТЗ, разработки эскизного проекта и прототипирования (согласно ГОСТ 34.601-90, эти стадии называют формированием требований к АС, разработкой концепции АС и эскизным проектом).

Предпроектные исследования проводят путем анализа (обследования) деятельности предприятия (компании, учреждения, офиса), на котором создается или модернизируется АС. При этом нужно получить ответы на вопросы: что не устраивает в существующей технологии? Что можно улучшить? Кому это нужно и, следовательно, каков будет эффект? Перед обследованием формируются и в процессе его проведения уточняются цели обследования - определение возможностей и ресурсов для повышения эффективности функционирования предприятия на основе автоматизации процессов управления, проектирования, документооборота и т. п. Содержание обследования - выявление структуры предприятия, выполняемых функций, информационных потоков, имеющихся опыта и средств автоматизации. Обследование проводят системные аналитики (системные интеграторы) совместно с представителями организации-заказчика.

На основе анализа результатов обследования строят модель, отражающую деятельность предприятия на данный момент (до реорганизации). Такую модель называют *«As Is» (как есть).* Далее разрабатывают исходную концепцию АС. Эта концепция включает в себя предложения по изменению структуры предприятия, взаимодействию подразделений, информационным потокам, что выражается в модели «To Be» (как должно быть).

Результаты анализа конкретизируются в ТЗ на создание АС. В ТЗ указывают потоки входной информации, типы выходных документов и предоставляемых услуг, уровень защиты информации, требования к производительности (пропускной способности) и т. п. ТЗ направляют заказчику для обсуждения и окончательного согласования.

Эскизный проект (техническое предложение) представляют в виде проектной документации, описывающей архитектуру системы, структуру ее подсистем, состав модулей. Здесь же содержатся предложения по выбору базовых программно-аппаратных средств, которые должны учитывать прогноз развития предприятия.

В отношении аппаратных средств и особенно ПО такой выбор чаще всего есть выбор фирмы-поставщика необходимых средств (или, по крайней мере, базового ПО), так как правильная совместная работа программ разных фирм достигается с большим трудом. В проекте может быть предложено несколько вариантов выбора. При анализе выясняются возможности покрытия автоматизируемых функций имеющимися программными продуктами и, следовательно, объемы работ по разработке оригинального ПО. Подобный анализ необходим для предварительной оценки временных и материальных затрат на автоматизацию. Учет ресурсных ограничений позволяет уточнить достижимые масштабы автоматизации, подразделить проектирование АС на работы первой, второй очереди и т. д.

После принятия эскизного проекта разрабатывают *прототип* АС, представляющий собой набор программ, эмулирующих работу готовой системы. Благодаря прототипированию можно не только разработчикам, но и будущим пользователям АС увидеть контуры и особенности системы и, следовательно, заблаговременно внести коррективы в проект.

Как на этапе предпроектных исследований, так и на последующих этапах целесообразно придерживаться определенной дисциплины фиксации и представления получаемых результатов, основанной на той или иной методике формализации спецификаций. Формализация нужна для однозначного понимания исполнителями и заказчиком требований, ограничений и принимаемых решений.

При концептуальном проектировании применяют ряд спецификаций, среди которых центральное место занимают модели преобразования, хранения и передачи информации. Модели, полученные в процессе обследования предприятия, являются моделями его функционирования. В процессе разработки АС модели, как правило, претерпевают существенные изменения (переход от «As Is» к «To Be») и в окончательном виде модель *«To Be»* рассматривают в качестве модели проектируемой АС.

Различают функциональные, информационные, поведенческие и структурные модели. *Функциональная* модель системы описывает совокупность выполняемых системой функций. *Информационная* модель отражает структуры данных - их состав и взаимосвязи. *Поведенческая* модель описывает информационные процессы (динамику функционирования), в ней фигурируют такие категории, как состояние системы, событие, переход из одного состояния в другое, условия перехода, последовательность событий, осуществляется привязка ко времени. *Структурная* модель характеризует морфологию системы (ее построение) - состав подсистем, их взаимосвязи.

Содержанием последующих этапов нисходящего проектирования (согласно ГОСТ 34.601-90, это стадии разработки технического проекта, рабочей документации, ввода в действие) являются уточнение перечней приобретаемого оборудования и готовых программных продуктов, построение системной среды, детальное инфологическое проектирование баз данных и их первоначальное наполнение, разработка собственного оригинального ПО, которая, в свою очередь, делится на ряд этапов нисходящего проектирования. Эти работы составляют содержание *рабочего проектирования.* После этого следуют закупка и инсталляция программно-аппаратных средств, внедрение и опытная эксплуатация системы.

Особое место в ряду проектных задач занимает разработка проекта корпоративной вычислительной сети, поскольку ТО АС имеет сетевую структуру. Если территориально АС располагается в одном здании или в нескольких близко расположенных зданиях, то корпоративная сеть может быть выполнена в виде совокупности нескольких локальных подсетей, связанных опорной локальной сетью. Кроме выбора типов подсетей, связных протоколов и коммутационного оборудования приходится решать задачи распределения узлов по подсетям, выделения серверов, выбора сетевого ПО, определения способа управления данными в выбранной схеме распределенных вычислений и т. п.

В случае если АС располагается в удаленных друг от друга пунктах, в частности расположенных в разных городах, то решается вопрос об аренде каналов связи для корпоративной сети, поскольку альтернативный вариант использования выделенного канала в большинстве случаев оказывается неприемлемым вследствие высокой цены. Естественно, что при этом прежде всего рассматривается возможность использования услуг Internet. Возникающие при этом проблемы связаны с обеспечением информационной безопасности и надежности доставки сообщений.

Контрольные вопросы

1. Дайте определение понятия «проектирование».
2. Что является предметом изучения в теории систем?
3. Назовите признаки, присущие сложной системе.
4. Приведите примеры иерархической структуры технических объектов, их внутренних, внешних и выходных параметров.
5. Приведите примеры условий работоспособности.
6. Почему проектирование обычно имеет итерационный характер?
7. Назовите основные стадии проектирования технических систем. Чем обусловлено прототипирование?

11. Что понимают под комплексной АС?

Лекция № 4.
Тема: Системные среды САПР. Назначение и состав системных средств.

План

4.1. Назначение системных сред автоматизированных систем.
4.2. Системы управления базами данных.
4.3. Распределенные базы данных.
4.4. Интеллектуальные средства поддержки принятия решений.
4.5. Интеграция ПО в САПР.

Ключевые слова: база данных, специфика, транзакция, иерархическая структура, файловый сервер, трехфазная схема, минимизация, интероперабельность, тиражирование, интеграция, методология.

4.1.Назначение системных сред автоматизированных систем

Системы автоматизированного проектирования относятся к числу наиболее сложных и наукоемких АС. Наряду с выполнением собственно проектных процедур необходимо автоматизировать также управление проектированием, поскольку сам процесс проектирования становится все более сложным и зачастую приобретает распределенный характер. На крупных и средних предприятиях заметна тенденция к интеграции САПР с АСУП и СДО. Для управления столь сложными интегрированными системами в их составе имеется специальное ПО - системная среда САПР или АС, называемая в настоящее время системой управления проектными данными или системой управления жизненным циклом изделий.

История систем управления проектными данными - систем PDM - непосредственно связана с развитием САПР. Появление системных сред в САПР ознаменовало переход от использования отдельных не связанных друг с другом программ, решающих частные проектные задачи, к применению интегрированной совокупности таких программ.

Интегрирующим компонентом в 1970-е годы стала единая *база данных* САПР. Однако попытки использовать имевшиеся в то время СУБД не приводили к удовлетворительным результатам в силу разнообразия типов проектных данных, распределенного и параллельного характера процессов проектирования, с одной стороны, и недостаточной развитости баз данных - с другой.

Специализированные СУБД, ориентированные на САПР, были созданы в 80-е годы. Однако они не учитывали или в недостаточной степени удовлетворяли требованиям обеспечения целостности данных, управления

потоками проектных работ, много аспектного доступа пользователей к данным.

И лишь на рубеже 80 - 90-х годов появились системы управления проектными данными, названные в то время Framework или системными средами, сначала в САПР электронной промышленности, а позднее и в САПР машиностроения, где они и получили наименование PDM.

На протяжении 90-х годов роль системных сред неуклонно повышалась. Во-первых, из-за роста сложности проектируемых объектов и необходимости сокращать сроки проектирования. Во-вторых, вследствие необходимости интеграции систем проектирования с системами управления предприятием и технологическими процессами. Благодаря развитию Internet, Web- и CALS-технологий такая интеграция стала возможной в глобальном масштабе. Современные системы управления проектными данными предназначены для информационного обеспечения проектирования и выполняют следующие основные функции:

- хранение проектных данных и доступ к ним, в том числе ведение распределенных архивов документов, их поиск, редактирование, маршрутизация и визуализация;
- управление конфигурацией изделия, т. е. ведение версий проекта, управление внесением изменений;
- создание спецификаций;
- защита информации;
- интеграция данных (поддержка типовых форматов, конвертирование данных).

Основной компонент систем PDM - *банк данных* (БнД). Он состоит из системы управления базами данных и баз данных. Межпрограммный интерфейс в значительной мере реализуется через информационный обмен с помощью БнД. PDM отличает легкость доступа к иерархически организованным данным, обслуживание запросов, выдача ответов не только в текстовой, но и в графической форме, привязанной к конструкции изделия. Поскольку взаимодействие внутри группы проектировщиков в основном осуществляется путем обмена данными, то в системе PDM часто совмещают функции управления данными и управления параллельным проектированием.

4.2. Системы управления базами данных

В большинстве автоматизированных информационных систем применяют СУБД, поддерживающие реляционные модели данных. Среди общих требований к СУБД можно отметить: 1) обеспечение целостности данных (их полноты и достоверности); 2) защита данных от

несанкционированного доступа и от искажений вследствие возникающих сбоев аппаратуры; 3) удобство пользовательского интерфейса; 4) в большинстве случаев важна возможность распределенной обработки в сетях ЭВМ.

Первые два требования обеспечиваются ограничением прав доступа, запрещением одновременного использования одних и тех же обрабатываемых данных (при возможности их модификации), введением контрольных точек (checkpoints) для защиты от сбоев и т. п.

Банк данных в САПР является важной обслуживающей подсистемой, он выполняет функции информационного обеспечения и имеет ряд особенностей. В нем хранятся как редко изменяемые данные (архивы, справочные данные, типовые проектные решения), так и сведения о текущем состоянии различных версий выполняемых проектов. Как правило, БнД работает в многопользовательском режиме, с его помощью осуществляется информационный интерфейс (взаимодействие) различных подсистем САПР. Построение БнД САПР - сложная задача, что обусловлено следующими особенностями САПР [4,5].

Разнообразие проектных данных, фигурирующих в процессах обмена как по своей семантике (многоаспектность), так и по формам представления. В частности, значительна доля графических данных.

1. Нередко обмены должны производиться с высокой частотой, что предъявляет жесткие требования к быстродействию средств обмена (полагают, что СУБД должна работать со скоростью обработки нескольких тысяч сущностей в секунду).

2. В САПР проблема целостности данных оказывается более трудной для решения, чем в большинстве других систем, поскольку проектирование является процессом взаимодействия многих проектировщиков, которые не только считывают данные, но и изменяют их, причем в значительной мере работают параллельно. Из этого факта вытекают следствия: во-первых, итерационный характер проектирования обычно приводит к наличию по каждой части проекта нескольких версий, любая из них может быть принята в дальнейшем в качестве основной, поэтому нужно хранить все версии с возможностью возврата к любой из них; во-вторых, нельзя допускать использования неутвержденных данных, поэтому проектировщики должны иметь свое рабочее пространство в памяти и работать в нем автономно, а моменты внесения изменений в общую базу данных должны быть согласованными и не должны порождать для других пользователей неопределенности данных.

3. Транзакции могут быть длительными и трудоемкими. *Транзакцией*

называют последовательность операций по удовлетворению запроса. В САПР внесение изменений в некоторую часть проекта может вызвать довольно длинную и разветвленную сеть изменений в других его частях из-за существенной взаимозависимости компонентов проекта (многошаговость реализации запросов). В частности, транзакции могут включать в себя такие трудоемкие операции, как верификация проектного решения с помощью математического моделирования. В результате транзакции могут длиться даже несколько часов и более. Одна из трудностей заключается в отображении взаимозависимости (ассоциативности) данных. При хранении компонентов проекта во внешней памяти затраты времени на обработку запросов оказываются значительно выше, чем в большинстве других автоматизированных систем с менее выраженными взаимозависимостями данных.

4. Иерархическая структура проектных данных и, следовательно, отражение наследования в целях сокращения объема базы данных.

В определенной мере названные особенности учитываются в СУБД третьего поколения, в которых стали применяться черты объектно-ориентированных (объектных) СУБД. В них наборы данных, характеризующих состояние предметной области (состояние проекта в случае САПР), помещаются в отдельные файлы. Интерпретация семантики данных осуществляется с помощью специальных процедур (методов), сопровождающих наборы. Наследование свойств объектов предметной области выражается с помощью введения категорий класса, надкласса, подкласса. Информационные модели приложений для таких СУБД разрабатываются на основе методик типа IDEF1X.

Объектные базы данных выгодны тем, что, во-первых, данные по конкретным объектам проектирования не разбросаны по множеству таблиц, как это имеет место в реляционных базах данных, а сосредоточены в определенных местах. Во-вторых, для каждого объекта могут быть назначены свои типы данных. В результате проще решаются задачи управления и удовлетворения запросов.

Наряду с чисто объектными СУБД, применяют СУБД объектно-реляционные. В последних происходит объединение свойств реляционных и объектно - ориентированных СУБД: объектно-ориентированная СУБД снабжается непроцедурным языком запросов или в реляционную СУБД вводятся наследование свойств и классы. Непроцедурность входного языка обеспечивается использованием языка SQL. Его операторы непосредственно включаются в программы на языке С. Возможно написание дополнительных программ, интерпретирующих SQL-запросы.

Отличительные особенности СУБД третьего поколения: расширенный набор возможных типов данных (это абстрактные типы, массивы, множества, записи, композиции разных типов, отображение величин со значениями разных типов), открытость (доступность из разных языков программирования, возможность обращения к прикладным системам из СУБД), непроцедурность языка (общепринятым становится язык запросов SQL), управление асинхронными параллельными процессами, состояние которых отражает база данных.

Рассмотренные особенности БнД в САПР позволяют квалифицировать их как системы *Data Warehouse (DW),* т. е. *хранилища данных.* Для хранилищ данных характерен ряд особенностей, совпадающих с названными выше особенностями БнД САПР: 1) длительное хранение информации, отражающей историю разработок; 2) частота операций чтения данных выше частоты операций обновления данных; 3) использование единых форматов для однотипных данных, полученных из различных источников (например, от разных программно-методических комплексов).

Эти особенности позволяют управлять конфигурацией проектов, что, в частности, означает хранение в САПР всех версий проекта и, возможно, данных по проектам предыдущих разработок, удовлетворение сложных запросов, для ответа на которые требуются извлечение и обработка данных из различных частей хранилища (так называемая многомерная обработка). Модели данных в DW отличаются от реляционных моделей (RM): в RM использованием нормальных форм стремятся максимально уменьшить избыточность данных, что приводит к увеличению числа таблиц, но уменьшенных размеров, при этом многомерный поиск в множестве таблиц затруднен. Поэтому в DW чаще используется модель данных «звезда», в которой имеется общая таблица фактов (Fact Table) и каждому факту ставится в соответствие несколько таблиц с необходимыми атрибутами. Целостность данных в DW обеспечивается проверкой и трансформацией данных, вводимых из внешних источников, наличием дисциплины обновления данных, централизованным хранением основной базы, при этом достаточное быстродействие поддерживается передачей копий определенных частей базы в локальные базы, называемые киосками данных (Data Mart) и ориентированные на отдельные группы пользователей.

Варианты управления данными в сетях АС. При сетевой организации АС информационное обеспечение может быть реализовано по одному из следующих вариантов: 1) FS - файловый сервер; 2) RDA - доступ к удаленным данным; 3) DBS - сервер баз данных; 4) AS - сервер приложений. Варианты различаются распределением между разными узлами сети

функций хранения данных, управления данными, обработки данных в приложениях и интерфейса с пользователем. На рис. 4.1 место среды передачи данных показано вертикальной чертой для первых трех вариантов.

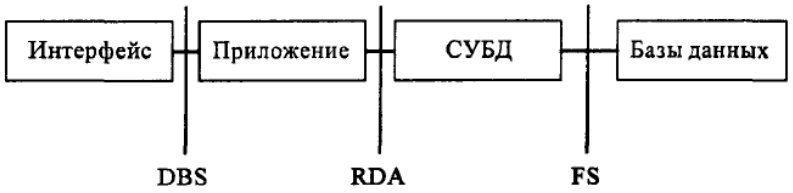


Рис. 4.1. Варианты двухзвенных схем распределенных вычислений

Каждый вариант имеет свою область применения.

Вариант файл-сервера характерен для локальных сетей на персональных ЭВМ с небольшим числом пользователей. Вследствие интенсивного трафика и трудностей с защитой информации эта структура для большинства АС малоэффективна. Поэтому предпочтительнее иметь СУБД в узле сервера. Вариант RDA - это модель удаленного узла, она наиболее распространена в настоящее время среди АС. В ней уменьшен трафик по сравнению с FS, унифицирован интерфейс с СУБД на основе языка SQL.

Примечание. Клиентов в FS и RDA иногда именуют «толстыми» клиентами, так как в них сосредоточены средства выполнения приложений.

Дальнейший переход к системе распределенных вычислений приводит к перемещению прикладного ПО или его части на специальный сервер или сервер базы данных, т. е. реализуются двух- и трехзвенные схемы. DBS - двухзвенная структура дистанционного управления, основанная на разделении прикладных процедур на две части: индивидуальные для каждого пользователя и общие для многих задач. В этой структуре под приложением понимают совокупность именно общих процедур. Эта совокупность обычно представляется на процедурных расширениях SQL и сохраняется в специальном словаре базы данных. В альтернативных вариантах (например, в RDA) все прикладные процедуры включаются в прикладные программы и, следовательно, при необходимости их изменения приходится модифицировать практически все прикладное ПО. Выделение таких процедур в отдельное приложение облегчает их модификацию. Кроме того, в DBS снижается трафик, так как обмены по сети происходят не для каждой операции с базой данных, а для каждой транзакции, состоящей из нескольких операций.

Вариант AS реализуется по *трехзвенной схеме,* в которой для приложений используются узлы, отделенные от терминального (локального) узла и от сервера базы данных, т. е. одновременно используются модели DBS и RDA.

Помимо проблемы распределения серверных функций между узлами сети имеется проблема разделения этих функций между многими пользователями АС. Эта проблема решается либо по *схеме «один к одному»*, либо по *многопотоковой схеме.* В первой из них для каждого активного пользователя создается своя копия СУБД. Во второй СУБД должна быть реентерабельной программой, обслуживающей одновременно многих пользователей.

4.3. Распределенные базы данных

В крупных АС, построенных на основе корпоративных сетей, не всегда удается организовать централизованное размещение всех баз данных и СУБД на одном узле сети. Поэтому появляются *распределенные базы данных* (*РБД*).

При построении РБД приходится решать ряд сложных проблем, связанных с минимизацией трафика, обеспечением интероперабельности обработки данных и целостности данных [7].

Минимизация трафика нужна в связи с тем, что при обслуживании запроса могут потребоваться данные из многих узлов, пересыпаемые по сети. Возможности минимизации видны из примера обработки данных нескольких таблиц из разных узлов. Очевидно, что целесообразна однократная пересылка таблиц (причем таблиц именно меньшего размера) на один узел, на котором и будет обрабатываться запрос.

Интероперабельность выражает способность взаимодействия программ, работающих в гетерогенных сетях (в разных операционных средах или с разными СУБД). Интероперабельность обеспечивается или с помощью программ- шлюзов (конверторов) для каждой пары взаимодействующих сред, или с помощью единого унифицированного языка взаимодействия. Таким языком для доступа к базам данных является язык SQL, интероперабельность на его основе имеет место в системе ODBC (Open Data Base Connectivity), пример реализации которой показан на рис. 4.2. В примере СУБД FoxPro находится в локальном узле, а СУБД Ingres и Informix - в удаленных узлах. Прикладная программа имеет ODBC-интерфейс, не зависимый от особенностей различных СУБД. Менеджер драйверов реализует на базе унифицированного языка SQL все нюансы доступа к базам данных, общие для разных СУБД. Драйвер конкретной СУБД преобразует инвариантные к СУБД запросы в форму, принятую в данной СУБД. В

трехзвенной структуре менеджер драйверов может быть размещен на промежуточном сервере.

Обеспечение целостности в РБД намного сложнее, чем в одноузловых базах данных. Различают два подхода к построению РБД: 1) тиражирование (репликация), при котором на нескольких серверах (в узлах) сети расположены копии базы данных; 2) полномасштабная распределенность, при которой разные части базы данных находятся на разных серверах сети (классическая распределенность).

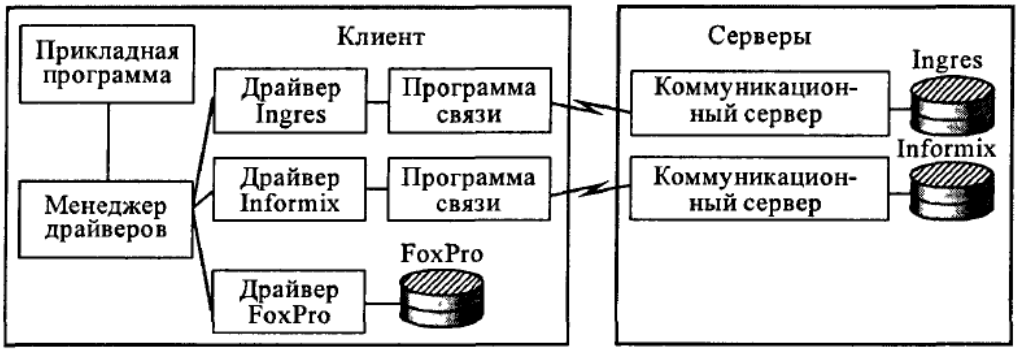


Рис. 4.2. Структура системы ODBC

Применяют два способа тиражирования.

Способ, называемый репликацией первой копии, основан на выделении среди серверов с копиями базы данных одного первичного сервера (репликатора). Внесение изменений пользователями возможно только в базы данных первичного сервера, который в дальнейшем осуществляет тиражирование. *Тиражирование* - это перенос изменений баз данных из первичного сервера во все вторичные (локальные) серверы, которые используются клиентами только для чтения данных. Репликатор реагирует на события, фиксируемые триггерами, периодически пересылает обновленные данные в копии базы данных. Недостаток способа - невысокая надежность, присущая любым централизованным структурам.

Надежность повышается при использовании способа голосования: изменения посылаются не в один первичный, а в некоторые N серверов. При этом любой запрос на чтение направляется к некоторым M серверам, причем $N + M > K$, где K-общее число серверов. Принимается последняя по времени обновления версия ответа.

Тиражирование вносит избыточность в хранимые данные, появляются трудности с разрешением конфликтов ввиду возможных несогласованных изменений в локальных базах данных. Однако по сравнению с классическими

РБД, в которых данные не дублируются, заметно уменьшается трафик, надежнее и проще работа с локальными базами данных.

В классических распределенных СУБД (РСУБД) необходимо управлять одновременным доступом, что должно гарантировать целостность (сериализуемость) баз данных. Наиболее широко используются алгоритмы управления, основанные на механизме блокировки. При этом *блокировкой* называют ситуацию, при которой некоторая транзакция объявила о желании получить полномочия на доступ к странице памяти и, следовательно, другие транзакции не имеют права занимать этот ресурс.

Одним из способов управления является централизованное блокирование, при котором на одном из узлов поддерживается единая таблица блокировок. Такой узел устанавливает очередность выполнения транзакций, что исключает конфликты. Однако при централизованном управлении невысока надежность и требуется мощный сервер.

В РСУБД с репликацией нет проблемы согласования при записи действий многих узлов. Собственно тиражирование чаще всего выполняется по правилу полной эквивалентности - обновленные данные сразу же после изменившей их транзакции рассылаются по всем локальным базам данных. Чтение же выполняется из базы данных одного конкретного узла, наиболее близкого к пользователю в функциональном или географическом смысле.

Сложнее решать проблемы распределенного управления, что требуется в РСУБД без тиражирования. Одним из распространенных протоколов распределенного управления является протокол двухфазной фиксации транзакций. На первой фазе инициатор транзакции (координатор) рассылает участникам выполнения транзакции оповещения о блокировке. В ответ узлы сообщают о своей готовности или неготовности. На второй фазе координатор сообщает либо о «глобальной фиксации», т.е. о выполнении транзакции, либо об откате транзакции. Неприятности возможны при сбоях, которые могут оставить некоторый узел в заблокированном состоянии: он не может ни выполнять транзакцию, ни отменять ее в одностороннем порядке.

4.4. Интеллектуальные средства поддержки принятия решений

В общем случае полная формализация управления проектированием не может быть достигнута, поэтому полезную роль играют *системы DSS (Decision Support Systems)* поддержки решений, принимаемых людьми. В качестве таких систем часто используют хранилища данных и *OLAP-средства {On-Line Analytical Processing*).

OLAP-средства должны обеспечивать оперативный доступ к данным, на основе которого выявляются зависимости между параметрами (измерениями в многомерной модели приложения). В OLAP-системах на

реляционных СУБД аналитическая обработка, или, другими словами, многомерный динамический анализ данных, требует просмотра большого числа записей из разных таблиц. Поэтому производительность оказывается невысокой. В специализированных OLAP-системах, обеспечивающих более быстрый многомерный анализ, но с более существенными ограничениями на объем базы данных, данные хранятся в виде гиперкубов или поликубов - многомерных таблиц с постоянным или переменным числом ячеек соответственно. Пример OLAP-системы - Oracle Express, которая помогает менеджерам и аналитикам получать данные в виде разрезов таких многомерных таблиц, готовить отчеты, обосновывать решения.

В составе подсистем управления методологией проектирования полезно иметь средства консультирования по принятию проектных решений. Они могут быть представлены в виде множества модулей, объединяемых гипертекстовой оболочкой. Каждый модуль содержит некоторый совет по выбору решения, преодолению противоречий, возникающих в процессе проектирования. Здесь уместно использование методов и приемов решения изобретательских задач.

4.5. Интеграция ПО в САПР

Интеграция ПО базируется на идеях объектно-ориентированного программирования. Следует различать синтаксический и семантический аспекты интеграции. Синтаксическая интеграция реализуется с помощью унифицированных языков и форматов данных, технологий типа ODBC для доступа к общему банку данных или компонентно-ориентированных (CBD - Component-Based Development) технологий. Семантическая интеграция подразумевает автоматическое распознавание разными системами смысла передаваемых между ними данных и достигается значительно труднее. Для создания ПО САПР, так же как и других сложных автоматизированных информационных систем, определяющее значение имеют вопросы интеграции ПО. Теоретической базой для создания технологий интеграции ПО в САПР являются:

1) методология автоматизированного проектирования, в соответствии с которой осуществляются типизация проектных процедур и маршрутов проектирования в различных предметных областях, выявление типичных входных и выходных данных процедур, построение информационных моделей приложений и их обобщение, сравнительный анализ альтернативных методов и алгоритмов выполнения типовых процедур;

2) объектно-ориентированная методология, в соответствии с которой множества сущностей, фигурирующих в процессах проектирования, подразделяются на классы, в классах появляются свои процедуры и типы

данных с отношениями наследования. Эти классы могут быть инвариантными и прикладными. Их обобщение и унификация приводят к появлению таких понятий и средств, как интегрированные ресурсы и прикладные протоколы, фигурирующие в стандартах STEP, или унифицированные программные компоненты типа графических ядер конструкторских САПР. Именно наличие типовых процедур и единообразное толкование атрибутов объектов в рамках конкретных протоколов позволяют разным программным системам «понимать» друг друга при взаимодействии.

Наряду с типовыми графическими ядрами известны типовые ПМК имитационного моделирования, конструирования деталей и механизмов, технологической подготовки производства и др. Возможность использования типовых программ в составе программных комплексов обусловлена именно унификацией интерфейсов при обменах данными [4,5].

В некоторых маршрутах проектирования обмены данными должны происходить с высокой частотой, что обусловливает специфические требования к интерфейсам. Примером могут служить задачи имитационного моделирования, в которых требуется имитировать взаимодействие процессов, описываемых с помощью различного МО (например, на сосредоточенном и распределенном иерархических уровнях или с помощью аналоговых и дискретных моделей). Для таких задач при моделировании характерно воспроизведение временной последовательности событий, происходящих в анализируемых взаимодействующих системах. Соответственно взаимодействие программ моделирования может происходить через фиксированное число временных шагов или по мере совершения тех или иных событий в моделируемых системах.

Так, в программах смешанного аналого-дискретного моделирования электронных устройств аналоговая часть моделируется с помощью программы анализа электронных схем, а дискретная часть - с помощью программы логического моделирования. Влияние аналоговой части на дискретную отображается в математических моделях путем преобразования непрерывных фазовых переменных в логические переменные в местах сопряжения частей модели, об ратное влияние выражается в преобразовании идеализированных логических сигналов в заданные функции времени, соответствующие электрическим сигналам заданной формы. Очевидно, что в содержательной части сообщений, передаваемых из одной части в другую, должны быть сведения либо о состояниях, выражаемых значениями фазовых переменных в интерфейсных узлах, либо о событиях - изменениях фазовых переменных. Обмен сообщениями может происходить многократно в течение акта одновариантного анализа.

В программно-методических комплексах конструирования происходит обработка графической информации. Содержательная часть сообщений относится к геометрическим элементам, их размерам и положению в пространстве. В программах технологической подготовки механической обработки деталей наряду с геометрической информацией о конструкциях заготовок в передаваемые сообщения могут входить сведения об инструменте, технологической оснастке, оборудовании, режимах обработки, нормах времени, траекториях движения инструмента и рабочих органов оборудования и т. п.

Другими словами, в каждом приложении совокупность используемых при обменах понятий, предметных переменных и числовых параметров существенно ограничена и достаточно определена для того, чтобы можно было ставить вопрос о типизации моделей и языка взаимодействия. Такие вопросы решаются в рамках технологий STEP/CALS. Число приложений, нашедших свое описание в прикладных протоколах STEP, ограничено, но совокупность таких протоколов может расширяться.

Прикладные протоколы STEP представляют семантическую сторону интеграционных технологий. Для интеграции нужна не только унификация моделей приложений, но и унификация механизмов взаимодействия, примерами которых являются технологии OLE, DDE, а также компонентно-ориентированные технологии.

Современные ОС позволяют работать одновременно с несколькими задачами с выделением каждой задаче своего окна на экране дисплея. Межпрограммные взаимодействия осуществляются путем посылки сообщений, как это принято в объектно-ориентированном программировании. Используются специальные средства организации взаимодействий.

Так, ОС Unix поддерживает взаимодействие асинхронных параллельных процессов, в том числе в разных узлах сети. Каждый клиент должен предварительно зафиксировать свои потребности в виде имен используемых сообщений. Сообщения имеют структуру фрейма. Получатель сообщения определяет, что сообщение относится к нему, вызывает обработчик сообщения и использует полученные данные в соответствии со своими функциями.

В операционных системах Microsoft для организации межпрограммных взаимодействий были предложены средства Clipboard, DDE, OLE и в дальнейшем технология ActiveX.

Работа Clipboard основана на традиционном способе обменных зон - выделении кармана (некоторой области оперативной памяти, разделяемой

взаимодействующими программами). При обменах одна программа посылает сообщение в карман, а другая извлекает, интерпретирует и использует это сообщение. Аналогичный режим работы осуществляется с помощью технологии формирования составных документов OLE, но расширены возможности комбинирования данных различных типов в передаваемых документах.

Различают два способа взаимодействия: связь (linking) и внедрение (embedding). При связи в создаваемый документ включается не сам текст из источника, а лишь ссылка на него. Очевидно, что здесь меньше затраты памяти, изменения в источнике автоматически переходят в документ. При внедрении текст из источника физически переносится в документ. После этого документ можно редактировать независимо от источника. Оба этих способа реализованы в технологии *OLE,* что и зафиксировано в ее названии (*Object Linking and Embedding).*

При обмене с помощью *DDE (Dynamic Data Exchange)* программа-клиент запрашивает режим диалога с программой-сервером. В сообщении указывается имя сервера, имя раздела (обычно раздел - это файл), имя элемента (обмениваемая порция информации). Предварительно такой элемент (атом) должен быть создан, а его адрес зафиксирован в таблице атомов. В ответ на запрос создается канал, по которому сервер передает данные или, что реализуется чаще, пересылает адрес нужного атома. По этому адресу клиент дополнительной командой может получить данные.

Функции систем PDM. Системы PDM предназначены для управления проектированием и его информационного обеспечения. Это осуществляется путем упорядочения информации о проекте и управления соответствующими документами, включая спецификации и другие виды представления данных. С помощью систем PDM поддерживаются информационные связи не только внутри САПР, но и с производственной и маркетинговой документацией, а также доступ к данным по различным атрибутам, навигация по иерархической структуре проекта. К системным вопросам, решаемым в PDM, относятся также управление проектами, интеграция программного обеспечения, пользовательский интерфейс и интерфейс с другими АС.

В системах PDM разнообразие типов проектных данных поддерживается их классификацией и соответствующим выделением групп с характерными множествами атрибутов. Такими группами данных являются аспекты описания, т. е. описания изделий с различных точек зрения. Для большинства САПР машиностроения характерными аспектами являются свойства компонентов и сборок (эти сведения называют *Bill of materials - BOM),* модели и их документальное выражение (основными примерами

могут служить чертежи, 3*D*- модели визуализации, сеточные представления для конечно-элементого анализа, текстовые описания), структура изделий, отражающая взаимосвязи между компонентами и сборками и их описаниями в разных группах.

Контрольные вопросы

1. Какие функции выполняет сетевое ПО?
2. Что понимают под менеджером и агентом в ПО управления сетью?
3. Что означает термин «эмуляция терминала»?
4. Охарактеризуйте различия между телеконференцией и видеоконференцией.
5. Назовите основные функции браузера.
6. Что такое «электронная подпись»?
7. Приведите примеры проектных процедур, выполняемых в системах ECAD.
8. Назовите основные функции АС: САПР, АСУП, АСУТП, АСД.
9. Почему в маршрут проектирования СБИС обычно не входят операции схемотехнического проектирования?
10. Каким образом обеспечивается тестируемость СБИС?
11. В чем заключается назначение систем SCADA?
12. Что понимают под диаграммой потока данных?
13. Приведите пример неспецифического отношения.
14. Перечислите основные особенности БнД в САПР.
15. Что такое «транзакция» в системах обработки данных?

Лекция № 5.

Тема: Виды САПР. Основные понятие о CAD, CAM и CAE системах. Схемы проектирование технологических процессов.

План

5.1. Промышленные САПР. CAD/CAM/CAE системы.

5.2. Классификация САПР. Сравнительный анализ отечественных и зарубежных систем.

5.3. Взаимодействие сапр с другими автоматизированными системами. CALS-технологии.

5.4. Средства реализации технологии параллельного проектирования.

5.5. Системы визуализации, разработки документации и средств обмена данными.

Ключевые слова: CAD, CAM, CAE, подсистемы, система CATIA, низкоавтоматизированное, среднеавтоматизированное, высокоавтоматизироанное, реинжиниринг, CALS технология, параллельное проектирование.

5.1. Промышленные САПР. CAD/CAM/CAE системы.

В данном лекции мы познакомимся с наиболее известными зарубежными и отечественными системами автоматизированного проектирования, широко применяющимися в промышленном производстве.

Прежде всего, приведем еще несколько “англоязычных” терминов.

CAD (Computer-Aided Design) - дословно переводится как “проектирование с помощью компьютера”, фактически этот термин означает системы геометрического моделирования и САПР чертежно-конструкторских работ.

CAM (Computer-Aided Manufacturing) - можно перевести как “производство с помощью компьютера”. Общепринятое значение этого термина в настоящее время - САПР технологической подготовки производства.

CAE (Computer-Aided Engineering) - означает компьютерные системы инженерного анализа.

Ведущие мировые производители программного обеспечения для автоматизации проектирования, как правило, поставляют на рынок интегрированные универсальные системы, удовлетворяющие требованиям проектирования и изготовления изделий в различных отраслях промышленности. Эти системы содержат в себе возможности автоматизации всех трех аспектов и называются поэтому CAD/CAM/CAE системами. Прежде чем перейти к рассмотрению лидеров мирового САПР-овского бизнеса, приведем примеры некоторых специализированных систем автоматизации проектирования конструкторско-технологического назначения.

Типичным примером “чисто” CAD - системы является всем известный AUTOCAD (Autodesk, Ltd.., США). Последние версии этой системы позволяют причислить ее к системам трехмерного моделирования, но для большинства российских пользователей, Autocad остается 2D - системой для машиностроительного черчения.

Из отечественных CAD-систем , не претендующих на автоматизацию технологической подготовкой производства, мы бы отметили разработку АО “Топ-системы” (г.Москва), которая раньше называлась Top-CAD, а сейчас получила название T-Flex CAD. Отличительной особенностью этой системы является, на наш взгляд, очень удачная реализация идеи параметризации

объектов. В Top-CADe все параметризировано. Это факт позволяет очень эффективно производить модификацию чертежей и сократить время проектирование новых. Известно, что в системе Autocad тоже есть возможности параметрического программирования, но они не являются органичными для системы и представляют собой некоторые “надстройки” над системой. Кстати, о надстройках.

Среди недорогих CAM-систем, имеющих некоторое распространение и в России, в качестве примера можно привести систему PEPS английской фирмы CAMTEK. Как и практически все САПР такого рода, PEPS обеспечивает автоматизацию подготовки управляющих программ для широкого класса станков с ЧПУ, включая токарные, фрезерные, сверлильные, электроэрозионные, прессы с ЧПУ, машины термической резки листовых материалов и т.д. Геометрию обрабатываемых деталей она может принимать из других широко распространенных CAD-систем через стандартные форматы файлов (например, через тот же “автокадовский” DXF-формат).

Следует отметить, что большинство уважающих себя CAM - систем имеют собственную несложную CAD-подсистему для описания геометрии деталей.

Отечественным аналогом PEPSa является система TURBO TIGRAS (дистрибьютер - АО “КАМЕЯ”, г.Москва).

Системы инженерного анализа (CAE-системы) включают в себя решение следующих основных задач:

- прочностной анализ (все виды расчетов на прочность конструкций методом конечных элементов);

- теплофизические расчеты;

- пластический анализ (анализ пластической деформации и оценка технологичности изготовления деталей методом литья под давлением);

- механический анализ (моделирование и прогнозирование поведения и движения механических систем).

Примером полнофункциональной CAE-системы является система ANSYS американской фирмы Swanson Analysis Systems. Отечественные аналоги нам не известны. Ряд организаций ведут работы по отдельным проблемам инженерного анализа, однако, еще раз повторимся, полномасштабные полнофункциональные коммерческие версии российских CAE-систем нам не известны.

Перейдем теперь к обзору интегрированных CAD/CAM/CAE систем.

Отметим, на наш взгляд, основные особенности лучших современных CAD/CAM/CAE- систем, являющихся признанными мировыми лидерами.

1.На этапе проектирования (геометрического моделирования) обеспечивают:

- твердотельное, поверхностное и каркасное 3-х мерное моделирование;
- черчение;
- моделирование «больших» сборок;
- фотореалистическое отображение;
- конструирование и раскрой из листовых материалов;
- поддержку промышленных графических стандартов типа IGES, DXF,VDA, STEP и т.д.;
- поддержку вывода документации во всех основных чертежных стандартах- ANSY, ISO, DIN, ЕСКД и т.д.
- прямые интерфейсы с другими CAD/CAM/CAE системами.

2. CAM- подсистемы обеспечивают:

- эффективную подготовку управляющих программ для фрезерной обработки от 2,5 до 5 координат, всех видов токарной, электроэрозионной, а также для газорежущих, плазморежущих, лазерных машин и прессов с ЧПУ;
- быстрое генерирование постпроцессоров для различных систем ЧПУ и контроллеров;

3. CAE-подсистемы являются полномасштабными (т.е. полностью решающими весь спектр задач без каких-либо ограничений) и обеспечивают все виды инженерного анализа.

4. Полные версии систем функционируют на основных типах рабочих станций (т.е. на UNIX- платформах). Для платформ MS WINDOWS 95 и WINDOWS NT существуют ограниченные версии систем.

Законодателями мод в разработке CAD/CAM/CAE систем являются, разумеется, американцы и, как ни странно, французы.

США являются автором 3 крупных интегрированных CAD/CAM/CAE систем, а именно:

- системы CADDS5 (разработчик фирма ComputerVision(CV));
- системы UNIGRAPHICS (разработчик фирма Electronic Data System(EDS));
- системы Pro/Engineer (разработчик фирма Parametric Technology Company(PTC),

Франция - 2-х:

- системы CATIA (разработчик фирма Dassault Systemes);
- системы EUCLID (разработчик фирма Matra Datavision);

Собственно говоря поставщиком (или, как сейчас говорят, “эксклюзивным дистрибьютером”) системы CATIA является компьютерный гигант - фирма IBM, так что эту систему тоже можно считать американской.

Говоря о CAD/CAM/CAE системах, покупаемых в России и в Уральском регионе нельзя не сказать еще о двух западных разработках: о системе CIMATRON одноименной израильской фирмы и системе DUCT английской фирмы DELCAM;

В Екатеринбурге имеется дочернее предприятие фирмы DELCAM (оно называется DELCAM-URAL), а представительство фирмы BEE PITRRON, которая является дистрибьютером системы CIMATRON, расположено непосредственно в УГТУ-УПИ.

5.2. Классификация САПР. Сравнительный анализ отечественных и зарубежных систем.

Предусматриваем деление САПР на девять групп:

1) САПР изделий машиностроения;
2) САПР изделий приборостроения;
3) САПР технологических процессов в машино- и приборостроении;
4) САПР объектов строительства;
5) САПР технологических процессов в строительстве;
6) САПР программных модулей;
7) САПР организационных систем;
8) резерв;
9) резерв.

Фактически такая классификация означала разделение систем «по назначению». Однако, универсальные САПР успешно применяют в различных предметных областях, кроме того, приведенный список не содержит, например, «геодезические» системы, которые, по некоторым данным, сегодня составляют около 13% рынка всех продаваемых в мире САПР.

САПР также разделяют по сложности объекта проектирования:

- до 100 составных частей - простых объектов;
- от 100 до 1000 - объектов средней сложности;
- от 1000 до 10000 - сложных;
- от 10000 до 1000000- очень сложных;
- свыше 1000000 - объектов очень высокой сложности.

Для любителей русского языка можно предложить небольшое логическое упражнение: попробуйте “почувствовать разницу” между САПР *очень сложных объектов* и САПР *объектов очень высокой сложности.*

Если объектом проектирования является некоторое изделие, то составной частью объекта является деталь. Если проектируется некий технологический процесс, то что является составной частью САПР до сих пор никто так и не определил.

Системы САПР также различаются по уровню автоматизации:

- низкоавтоматизированные САПР (до 25% проектных процедур);
- среднеавтоматизированные(от 25% до 50% проектных процедур);
- высокоавтоматизированные(свыше 50%)

Лет 6-8 назад все машиностроительные предприятия должны были периодически посылать в свои министерства в Москву отчеты об уровне автоматизации.

Есть еще несколько признаков классификации САПР, которые определяются ГОСТом, типа “кол-во выпускаемых документов”, но мы их рассматривать не будем.

“Буржуазные” CAD/CAM/CAE системы классифицируются гораздо проще: полномасштабные полнофункциональные CAD/CAM/CAE системы на рабочих станциях называются “тяжелыми” САПР-ами, а все остальные - “легкими”.

Все выше перечисленные CAD/CAM/CAE системы являются “тяжелыми”, а AUTOCAD и PEPS - “легким”. В России “тяжелых“ САПР в полном смысле этого слова не разработано до сих пор. Следует отметить, что на Западе в смысле классификации САПР тоже нет устойчивой терминологии. Некоторые специалисты относят, например, CIMATRON к “средним” системам по показателю цены за одно рабочее место. Цена CIMATRONa, действительно, значительно меньше цены, скажем, CADDS5 да и требования израильской системы к вычислительным ресурсам компьютера более скромные. В отдельных публикациях “тяжелой” называется САПР, 1 копия программного обеспечения которой стоит больше 15000$.

Теперь о технических характеристиках.

Из источников, заслуживающих доверия, можно сделать вывод, что CAD -подсистема лучше всего реализована в системе CATIA. Речь идет именно о моделировании объектов, а не о сборке изделий и проектировании чертежей.

Подсистема CAM (имеются в виду ЧПУ-шные задачи) предпочтительней у UNIGRAPHICS и Pro/Engineer. Pro/Engineer также обладает наиболее органичными функциями параметризации изделия в CAD подсистеме.

Наиболее полная подсистема CAE- во французких системах CATIA и EUCLID.

Наилучшая подсистема управления информацией и наибольшая “интегрированность”- у мирового лидера N 1, системы CADDS5.

Наилучший показатель “производительность/цена” - у CIMATRON-а. У него же, видимо, ввиду известной демографической “близости”, наилучшая документация на русском языке.

Этот анализ можно продолжить, но, думается, -”не стоит”. Главное, о чем следует помнить - это то, что при выборе системы необходимо аналитически мыслить и, опять же, понимать диалектику. Кажется очевидным, что если нам нужна система для проектирования управляющих программ для 5-ти координатного фрезерного станка, то следует искать CAM - систему, наилучшую по этому показателю. Однако, это делать надо не всегда. Например, плохой постпроцессор может все испортить.

Перейдем теперь к рассмотрению отечественных разработок. Как мы уже отмечали, национальные системы работают, в основном, “на платформе” MS DOS. MS WINDOWS версии систем представляют собой, чаще всего, адаптацию под новую ОС только «головной» части системы, а ядро остается без изменений.

Из наиболее распространенных разработок, имеющих, по крайней мере, несколько сот инсталляций в различных отраслях промышленности, следует назвать системы “КОМПАС” АО “АСКОН”(г.Санкт-Петербург) и “СПРУТ” АО “СПРУТ-Технологии”(г.Набережные Челны). В последнее время в Уральском регионе стала продаваться “московско-ижевская” система ADEM. Ее распространителем является екатеринбургское представительство фирмы BEE PITRON, то самое, которое продает и CAD/CAM/CAE систему CIMATRON.

5.3. Взаимодействие САПР с другими автоматизированными системами. CALS-технологии.

Снижение стоимости продукции, уменьшение времени ее выхода на рынок и повышение ее качества одновременно с качеством обслуживания являются конкурентно-определяющими факторами промышленности сейчас и в обозримом будущем. С учетом этих факторов, особенно если продукцией является технически сложное изделие, успешная деятельность предприятия на рынке зависит от реинжиниринга проектно-технологических и производственных процессов. Наиболее радикальным средством решения задач реинжиниринга является внедрение интегрированных информационных технологий с использованием современных средств вычислительной техники и сетевых решений как на отдельных предприятиях,

так и в рамках «расширенного предприятия», объединяющего всех поставщиков, соисполнителей и участников процессов проектирования, производства и эксплуатации (обслуживания) изделия.

Направление развития информационных технологий, полностью покрывающих запросы современной промышленности, получило название CALS* (Continious Acquisition Lifecycle Support), под которым понимается обеспечение непрерывной информационной поддержки изделия в течение всего его жизненного цикла.

**Термином CALS в середине 80-ых годов была названа программа Министерства обороны США по созданию автоматизированной системы закупок и материально-технического обеспечения военной техники и вооружений на всех этапах их жизненного цикла. Первоначальная расшифровка аббревиатуры CALS - Computer-aided Logistic Support (Компьютерная поддержка логистических систем) претерпела изменения и применяется теперь в нижеуказанном значении.*

Ключевым моментом CALS-технологий является безбумажное, т.е. формализованное на основе международных стандартов электронное представление информационной модели изделия, включающей все данные о нем. Возникнув в сфере военной промышленности западных стран, в первую очередь США, CALS-технологии получили свое дальнейшее развитие и послужили основой создания новейших наукоемких технологий и в других отраслях. Наиболее значительные успехи при внедрении указанных технологий достигнуты в автомобилестроении, авиастроении и судостроении, т.е. в отраслях, где сроки эксплуатации и затраты на обслуживание изделий особенно велики.

Эффективность CALS-технологий базируется на их комплексном применении. Грамотный реинжиниринг, повышающий эффективность реализации CALS-технологий, приводит к успеху изделия у конечного пользователя, для которого стоимость эксплуатации важна не менее стоимости самого изделия.

Таким образом, CALS-технологии, обеспечивая снижение стоимости изделия и затрат на его эксплуатацию при одновременном повышении качества его обслуживания, дают преимущество расширенному предприятию в конкурентной борьбе.

К сожалению, на большинстве предприятий даже сам САПР не является интегрированным. CAD-системы, как мы уже отмечали раньше, используются, в основном, для автоматизации чертежных работ в конструкторских подразделениях предприятия. В этих же подразделениях применяют и CAE-системы, но самое интересное то, что геометрическая

модель объекта при этом строится заново, а не берется, как это казалось бы естественным, из CAD-системы. CAM-системы преимущественно используются в двух направлениях:

- автоматизация подготовки управляющих программ для технологического оборудования с ЧПУ;

- автоматизация подготовки бумажной технологической документации (маршрутные, операционные карты и т.п.).

Исходные данные для этих подсистем (в том числе и геометрические) также вводятся вновь средствами CAM.

Таким образом, “сквозные” САПР все еще остаются большой редкостью и реально работают, как правило, на очень небольшом количестве заводов и проектных организаций.

Вместе с тем, следует отметить, что процесс интеграции САПР с другими автоматизированными системами, работающими внутри предприятия, в настоящее время активно развивается на многих предприятиях. В основном, следует говорить об активизации связи САПР с системами АСУП, откуда САПР получает информацию планового характера, касающуюся состава изделий(заказов) и графика технической подготовки для их производства. Полученные в результате работы проектные решения используются подсистемами АСУП для формирования экономических показателей производства и дальнейших управленческих решений. Например, полученная в CAM-подсистеме управляющая программа для изготовления детали, скажем, на фрезерном станке, обычно содержит всю необходимую информацию о трудоемкости ее изготовления, что является исходным данным для формирования решений и документов типа сменного задания, нарядов и др. АСУП-овской информации.

Связь САПР с подсистемами АСНИ (автоматизированными системами научных исследований) носит сегодня больше теоретический характер. Даже в научно-исследовательских институтах АСНИ и САПР обычно функционируют независимо.

Естественно, что в идеале все автоматизированные системы на предприятиях должны функционировать согласованно и обеспечивать передачу информации друг другу также автоматизированным способом. Рассмотрим некоторые особенности современного понимания интегрированных информационных технологий и, в частности, CALS-технологии [4,5].

Основные компоненты CALS-технологий.

К числу основных компонентов, относящихся к CALS-технологиям относятся следующие системы и средства:

- CAD / CAM/ CAE - системы;
- средства реализации технологии параллельного проектирования в режиме группового использования данных (Concurrent Engineering);
- средства управления проектными и инженерными данными (EDM - Enterprise Data Management) *(В настоящее время часто используется и другой термин для обозначения такого рода средств – PDM (Product Data Management);*
- системы визуализации и разработки документации;
- средства обмена данными и стандартные интерфейсы к специализированным системам;
- средства разработки прикладного программного обеспечения;
- методики анализа процессов предприятия в проектно-технологической, производственной и управленческой сферах для реинжиниринга этих процессов.

5.4. Средства реализации технологии параллельного проектирования.

О CAD/CAM/CAE-системах мы, будем считать, уже достаточно поговорили, о технологии параллельного проектирования также упоминали. Напомню суть этой технологии. Она заключается в ведении работ по созданию изделия в параллельной технологической среде (CAPE - Concurrent Art-to-Product Environment). Основным методом, который связывается с внедрением CAPE, является осознание и продвижение новой проектно-конструкторской философии, которая заключается в замене последовательного (ведомственного) процесса сквозной разработки изделия в интегрированный, параллельный процесс его создания на основе концепции расширенного предприятия. Рабочая среда расширенного предприятия предполагает включение в совместную работу всех участников - специалистов по конструированию, технологической подготовке производства, вопросам качества, покупки, продажи, маркетингу, поставщиков, заказчиков. Участники такого коллектива, работая в качестве тесно взаимодействующей многопрофильной группы, могут внести значительные улучшения в общий процесс разработки.

Необходимым условием работы в среде CAPE является наличие программных средств, позволяющих осуществлять параллельный санкционированный доступ к информационной модели, ориентирование и навигацию по структуре изделия для выбора необходимой геометрической информации или атрибутов, относящихся к тем или иным элементам структуры.

В качестве примера можно привести программный продукт CAMU (Concurrent Assembly Mock-Up). В нем предусмотрены удобные средства навигации по сборке с визуализацией структуры всей сборки и доступа к каждому компоненту. Любой элемент структуры, деталь либо узел может быть снабжен любым количеством атрибутов. Количество компонентов в сборке принципиально не ограничено. CAMU реализует технологию параллельного проектирования и работу с крупными сборками в групповом режиме использования данных, предоставляя всем участникам проектно-конструкторской бригады возможность одновременного и согласованного создания, анализа и модификации компонентов модели сборки составных частей и изделия в целом.

Система управления проектными и инженерными данными.

Система управления проектными и инженерными данными является одним из краеугольных камней GALS-технологий. Использование такой системы дает предприятию ряд преимуществ, основными среди которых являются следующие:

- экономия времени и средств за счет параллельного выполнения работ, уменьшения количества ошибок и переделок при совместном использовании данных в рамках расширенного предприятия;
- надежное централизованное хранение всего множества проектных данных, которые представляют собой «полное электронное определение изделия»;
- защита данных, исключающая несанкционированный доступ;
- архивирование и резервное копирование, позволяющее производить быстро восстановление в случае потери данных;
- управление атрибутными данными, обеспечивающее немедленный доступ к неграфической информации об изделии;
- электронное сопровождение процессов, управление визуализацией и процедурами прохождения документов, обеспечивающие рациональный и эффективный мониторинг.

Назначение и функции этой системы можно проиллюстрировать на примере системы Optegra, одного из лидеров среди систем класса EDM.

Optegra обеспечивает глобальный доступ к данным и использование их на любом рабочем месте в любое время, поддерживает режим параллельной коллективной работы различных групп пользователей и обеспечивает управление информационной моделью изделия в системе всех этапов его жизненного цикла. В качестве данных в системе Optegra могут использоваться любые геометрические и технологические данные, включая трехмерные модели, полученные в любых CAD/CAM-системах, чертежи в

электронном виде, готовые модели и данные инженерных расчетов CAE-систем, NC-программы, растровые изображения, бумажные чертежи, спецификации, извещения об изменениях, инструкции, приказы, технические публикации и т.п. Если документ не присутствует в электронном виде, то он имеет электронный дескриптор, который и является объектом хранения и управления в системе Optegra.

Optegra позволяет сотрудникам всего предприятия через свои терминалы на рабочих станциях или персональных компьютерах получать доступ к любым данным об изделии, хранящимся в электронном виде на локальном или удаленном сервере, и управлять этими данными.

Система Optegra является открытой, модульной и наращиваемой. За счет открытости построения всех модулей Optegra может быть интегрирована в общую вычислительную сеть предприятия. Система обеспечивает легкость настройки с учетом специфики предприятия и практически неограниченную расширяемость с целью увеличения ее мощности и поддержки пользователей в рамках расширенного предприятия.

5.5. Системы визуализации, разработки документации и средств обмена данными.

Важным компонентом CALS-технологий являются средства интеллектуальной визуализации. Центральную роль среди них играет система визуализации сложных геометрических моделей и сборок. Она должна позволять выводить реалистические изображения модели в любых проекциях, осуществлять так называемые «прогулки» внутри модели в интерактивном режиме, определять наложение объектов друг на друга и расстояния между ними, создавать видеоизображения, фотореалистичные цветные изображения, а также осуществлять доступ к базе данных, чтобы получить справку об объектах.

Как мы уже отмечали выше, для вывода чертежно-конструкторской документации необходимо иметь программные средства, обеспечивающие поддержку вывода документации во всех основных чертежных стандартах - ЕСКД, ANSY, JIS, ISO, DIN, GD&T и др.

Существенным требованием, вытекающим из необходимости обмена данными между различными системами, реализующими GALS-технологии, является поддержка стандартов на представление и обмен данными в интегрированных системах, к числу которых относятся, в первую очередь, IGES, STEP, VDA-FS, SET, DXF и ряд специализированных стандартов, например, CGM, STL и др. Кроме указанных стандартов должны предусматриваться так называемые прямые трансляторы между различными системами, совместно применяемыми в рамках CALS-технологий, например,

трансляторы для обмена между системами автоматизированного проектирования: CADDS-CATIA-CADDS, CADDS-Unigraphics-CADDS, CADDS-Pro/Engineer-CADDS.

Системы разработки прикладного ПО. Средства разработки прикладного программного обеспечения имеют важное значение для реинжиниринга процессов предприятия, поскольку обеспечивают локализацию программных компонентов CALS-технологий и интеграцию в них информационного задела предприятия. При этом все программные системы, включая прикладные программы, должны функционировать в единой среде.

Практически все известные производители САПР-овских систем предлагают различные средства разработки прикладного ПО (вспомните, например, СПРУТ).

Computervision**,** предлагает, в частности, инструментальную программную платформу PELORUS, которая, как "они" утверждают, знаменует собой принципиально новый подход к модификации существующего программного обеспечения и создания любых программных модулей, интегрируемых в CAD/CAM/CAE - системы. Эта система позволяет разработать прикладные программы, возможности которых могут превосходить все имеющиеся в настоящее время системы автоматизированного проектирования, причем трудоемкость разработки снижается как минимум в два раза при значительном повышении эффективности программ, требующих минимум системных ресурсов.

Методика анализа предприятия. Опыт передовых предприятий показывает, что для повышения конкурентоспособности недостаточно вкладывать средства только в современное производственное оборудование, компьютеры, коммуникации и программное обеспечение. Необходимо на самом раннем этапе вложить средства и провести анализ существующих процессов во всех сферах деятельности предприятия. В результате такого анализа определяются цели и средства реинжиниринга этих процессов.

В этом направлении фирма Computervision давно и успешно использует в своей работе методику PDD (Product Development Diagnostic) для анализа проектно-технологической и производственной сферы предприятия с целью разработки комплексных планов реинжиниринга и внедрения CALS-технологий.

Аппаратное обеспечение CALS-технологий. Практически все программные продукты, которые могут использоваться в рамках CALS-технологий, ориентированы на наиболее распространенные платформы:

- для персональных компьютеров: Microsoft Windows;

• для рабочих станций UNIX-платформы: Sun SPARC Solaris, Sun SPARS SunOS, Digital UNIX (Alpha), SGI IRIX, Hewlett Packard/Hp-Ux и IBM AIX/600.

Ориентация на указанные платформы позволяет обеспечить преемственность и переносимость программного обеспечения наряду с высокими характеристиками надежности компьютерной техники вышеназванных фирм и высокими значениями отношения «функциональные возможности/цена».

Контрольные вопросы

1. Что означает термин CAD?
2. Объясните термины CAD/CAM/CAE системы.
3. Какие основные задачи включают себе системы инженерного анализа?
4. Какие основные особенности имеет современные CAD/CAM/CAE системы?
5. Как классифицируется САПР по сложности объекта проектирования?
6. Как различаются системы САПР по уровню автоматизацию?
7. Какие технические характеристики имеет CAD/CAM/CAE системы?
8. Что является ключевым моментом CALS-технологий?
9. Какие основные компоненты имеет CALS-технологий?
10. Какие возможности имеет программа Optegra?

Лекция № 6.

Тема: Виды компонентов обеспечения САПР. Лингивистическое обеспечение САПР (ЛО) и программное обеспечение САПР (ПО).

План

6.1. Свойства и структура ПО САПР.

6.2. Подходы к созданию общесистемного ПО.

6.3. Специализированное ПО САПР. Частота использования программ.

6.4. Показатели качества программ проектирования, программные системы и комплексы.

6.5. Основные принципы проектирования ПО САПР.

6.6. Лингвистическое обеспечение САПР.

6.7. Языки программирования и проектирования.

6.8. Языковые процессоры.

Ключевые слова: сопровождаемость, адаптируемость, обшесистемное ПО, управляющие программы, транслятор, драйвера, модульность, иерархичность, совокупность, комбинируемость, независимость.

6.1. Свойства и структура ПО САПР.

Свойства ПО САПР. К ПО САПР предъявляются требования экономичности, удобства использования, надежности, правильности, универсальности, открытости, сопровождаемости и мобильности.

Экономичность ПО оценивается затратами вычислительных ресурсов - машинного (процессорного) времени Тм и оперативной памяти Пм. Характер зависимости Тм и Пм от размерности задачи определяется в первую очередь свойствами МО. Однако неудачная программная реализация может существенно увеличить Тм и Пм. Недостаточная экономичность ПО обычно оказывается основным фактором, ограничивающим возможности исчерпывающего анализа, оптимизации и структурного синтеза проектируемых объектов.

Удобство использования ПО определяется его надежностью, наличием проблемно-ориентированных входных языков и средств диагностики ошибок пользователя.

Надежность ПО - свойство выполнять заданные функции в заданных условиях. Функции и условия формулируются в терминах той области, к которой относятся проектируемые объекты. Основной показатель надежности - вероятность получения правильного результата при использовании программы в сформулированных условиях.

Правильность ПО - свойство, характеризующее соответствие ПО спецификациям математического характера, т.е. правильность реализации в ПО выбранного МО. Несоответствие требований пользователей и выбранного МО снижает надежность, но не влияет на правильность. Например, если в программе анализа статических состояний некоторого класса объектов безошибочно реализован метод Ньютона, то программа будет правильной, но, по-видимому, ненадежной, так как условия сходимости метода Ньютона будут выполняться не для любого объекта из заданного класса при произвольном задании начального приближения.

Универсальность ПО характеризуется ограничениями на его применение. Эти ограничения могут относиться к типам и элементному составу анализируемых или синтезируемых структур, диапазонам числовых значений внутренних и внешних параметров, перечню выполняемых проектных процедур. Универсальность связана с надежностью ПО - чем тщательнее и полнее выявлены и оговорены ограничения, тем ниже степень

универсальности программы, но выше ее надежность. В САПР необходимо стремиться к достижению высокой надежности ПО. Поэтому эксплуатация нескольких узкоспециализированных, но надежных программ предпочтительнее применения одной универсальной программы, если за повышение степени универсальности приходится платить снижением надежности.

Открытость ПО характеризуется возможностями внесения в него изменений в процессе эксплуатации (модернизации). Понятие открытости близко к понятию адаптируемости, под которым подразумевается возможность модификации ПО для поддержания его работоспособности и эффективности в изменяющихся условиях применения (выборочная установка).

Сопровождаемость ПО - свойство, близкое свойству открытости, характеризует удобство поддержания ПО в работоспособном состоянии и обеспечивается структурированностью ПО и наличием необходимой эксплуатационной документации.

Мобильность ПО, называемая также переносимотью, определяется легкостью перестройки ПО, эксплуатировавшегося на ЭВМ с одной системой команд, на ЭВМ с другой системой команд. Программы, записанные на машинно-ориентированных языках, непереносимы. Использование языков высокого уровня создает предпосылки для создания мобильных программ. Однако для повышения мобильности необходимы дополнительные меры по фиксации и выделению в сменяемые блоки элементов ПО, отражающих специфику архитектуры ЭВМ и связанных с ними операционных систем (ОС).

Структура ПО САПР. ПО САПР делится на базовое, общесистемное и специализированное (прикладное).

Базовое ПО разрабатывается и поставляется совместно с аппаратурой АРМ и предназначено для использования многими проектными организациями. Типичными примерами базового ПО является ПО обслуживающих подсистем САПР - графических редакторов, диалоговых мониторов и т.п.

Общесистемное ПО является инвариантным к объектам проектирования и должно быть защищено от пользователей-проектировщиков. Основные функции общесистемного ПО САПР:

- управление процессом вычислений; ввод, вывод и обработка инструкций пользователей; диалоговая
- взаимосвязь с пользователем в процессе проектирования; хранение, поиск, анализ, модификация данных, защита их целостности; решение

общесистемных задач; контроль и диагностика в процессе решения задач проектирования. В состав общесистемного ПО входят:

- мониторная диалоговая система; системы управления базами данных(СУБД);информационно-поисковые системы;
- геометрические и графические процессоры; средства формирования графической и текстовой информации; средства для выполнения общетехнических расчетов.

Общесистемное ПО САПР состоит из двух видов программ: обрабатывающие программы и управляющие программы.

Управляющие программы осуществляют первоначальную загрузку оперативной памяти и управление всей работой системы, включая обработку прерываний, распределение работы каналов, загрузку программ из библиотеки в оперативную память. Управляющие программы обеспечивают мультипрограммную работу, осуществляют связь с оператором, представляют пользователю широкие возможности в управлении массивами данных. Управляющие программы включают в себя три группы программ: управление задачами, управление заданиями и управление данными (рис. 6.1).

Рис. 6.1. Структура общесистемного ПО САПР

Управление задачами осуществляется программой, которая носит название супервизора (иначе диспетчер, монитор). Именно супервизор реализует режим мультипрограммирования или режим разделения времени, который состоит в реализации параллельного выполнения нескольких рабочих программ. Супервизор вначале включает в работу первую из задач, находящихся в главной памяти. Если эта задача потребовала обмен данными с ВУ, то после включения в работу этого ВУ управление передается на начало программы второй задачи, а перед этим запоминается адрес возврата на программу первой задачи и организуется защита памяти. Когда работа ВУ будет закончена, работа второй задачи прервется, таким же образом и происходит возврат к запомненному адресу. Режим разделения времени

обеспечивает возможность многим пользователям работать в системе, подавая заявки на решение требуемых задач.

Программы управления заданиями выполняют интерпретацию директив командного языка, который в любой ОС служит для расписания последовательности действий вычислительной системы при решении задач. Анализ и исполнение команд пользователя, включая загрузку готовых программ из файлов в оперативную память ЭВМ и их запуск, осуществляется командным процессором ОС, который выполняет важную функцию поддержки взаимодействия с пользователями.

Кроме ввода отдельных команд, которые немедленно выполняются, имеется возможность составления целых программ на командном языке, с помощью которых можно задать довольно сложную последовательность действий, не прибегая к обычному языку программирования. Командный процессор позволяет создать удобную операционную обстановку для конкретного пользователя, избавив его от утомительных служебных операций.

Программы управления данными обеспечивают поиск, хранение, передачу и обработку файлов (данных).

В состав ***обрабатывающих программ*** входят:

- файловая система; - драйверы ВУ;
- трансляторы с алгоритмических языков.

Рассмотрим указанные компоненты подробнее.

Файловая система - хранилище программ и данных. Одна из важнейших функций ОС - организация файловой системы. Файл - это место постоянного хранения информации - программ, данных для их работы, текстов, закодированных изображений и др. Реализуются файлы как участки памяти на жестких магнитных дисках. Каждый файл имеет имя, зарегистрированное в каталоге - оглавление файлов. Каталог (иногда называемый директорией) доступен пользователю через командный язык ОС - его можно просматривать, переименовывать зарегистрированные в нем файлы, переносить их содержимое на новое место и удалять. Каталог может иметь собственное имя и храниться в другом каталоге наряду с обычными файлами; так образуются иерархические файловые структуры.

Структура файловой системы и структура хранения данных определяют удобство работы пользователя, скорость доступа к файлам, возможность создания хороших баз данных и т.д. От файловой системы во многом зависит организация многопользовательской работы.

ЭВМ может иметь довольно широкий набор ВУ. Помимо стандартных ВУ - дисплея, клавиатуры, жестких дисков, принтера, к машине могут

подключаться по последовательным и параллельным коммуникационным каналам дополнительные устройства ввода/вывода - графопостроители, модемы, контроллеры локальных сетей, АЦП, ЦАП и др. Более того, даже стандартные устройства, например, принтеры, могут иметь несколько режимов работы и считаться вследствие этого разными устройствами. Каждое ВУ характеризуется своей пропускной способностью и структурой передаваемых данных. Поддержка широкого набора ВУ - одна из важнейших функций ОС. Для ее осуществления введено понятие ***драйвера*** - программы специального типа, ориентированной на управление ВУ. Каждому типу ВУ сопоставляется свой драйвер. Драйверы стандартных устройств образуют в совокупности базовую систему ввода/вывода BIOS, которая заносится в ПЗУ системного блока ПЭВМ. Драйверы дополнительных устройств могут подключаться к ОС динамически при запуске машины. Некоторые типы ОС предоставляют средства для составления новых драйверов, ориентированных на особые устройства.

В ДОС и ОС ЕС имеются ***трансляторы*** с языков ПЛ-1,ФОРТРАН, АЛГОЛ, КОБОЛ, АССЕМБЛЕР.

Типовая последовательность исполнения программ ОС при прохождении задачи пользователя на ЭВМ:

-в соответствии с директивами языка управления заданиями осуществляется ввод программы, записанной на алгоритмическом языке;

-вызов соответствующего транслятора;

-транслятор переводит исходную программу в объектный модуль, который включается в каталог;

-программа-редактор связей собирает необходимые модули (например, объектный модуль и некоторые модули библиотек) в загрузочный модуль;

-программа выборки настраивает и размещает в оперативной памяти загрузочный модуль;

-программа исполняется под управлением супервизора.

6.2. Подходы к созданию общесистемного ПО

ОС оценивается по эффективности управления вычислительной системой, основными показателями которой являются мера накладных расходов, возможности ОС и легкость освоения и использования.

Мера накладных расходов измеряется затратами оперативной памяти, затратами машинного времени на функционирование самой ОС и часто выражается таким параметром как, КПД.

$$КПД=1-Тос/Тм, \qquad (6.1)$$

где Тос - время на работу ОС; Тм - общее машинное время.

ОС по возможности не должна ограничивать пользователя в применении естественных и логических возможных операций. Легкость освоения и использования ОС представляется важным показателем системы.

Возможны два подхода к созданию общесистемного ПО САПР:

а) использование одной из существующих ОС (например, типа ДОС) в качестве основы и дополнение ее необходимыми программными средствами, решающими специфические задачи САПР;

б) создание проблемно-ориентированной ОС САПР вместо универсальной.

6.3. Специализированное ПО САПР. Частота использования программ.

В специализированном ПО САПР различают несколько подсистем, связанных с определенным уровнем иерархического проектирования или аспектом проектирования. Действительно, имеется заметная разница между программами конструкторского, технологического, схемотехнического, функционально-логического проектирования, которая может проявляться в особенностях входных языков, математических методах прикладных программ.

Программы одного уровня проектирования объединены в одну библиотеку, в которой содержатся по одной или несколько программ на каждую проектно конструкторскую задачу (ПКЗ). Для большинства задач нельзя назвать также метод и алгоритм, которые были ли бы наилучшими со всех точек зрения. Несколько программ одинакового целевого назначения, но с разными методами представляют проектировщику возможность выбирать оптимальный способ решения в каждой конкретной ситуации.

Реальные задачи проектирования, как правило, решаются с помощью последовательностей взаимодействующих программ, называемых маршрутами или цепочками программ.

Возможны два вида взаимодействия программ в маршрутах. Первый вид взаимодействия при работе с равноправными программами одного уровня характеризуется тем, что переход из *i*-ой программы в (i+1) происходит после завершения работы *i*-ой программы. Из (i+1)-ой программы возврата в *i*-ю программу уже не будет, переход осуществляется только в (i+2)-ю программу (рис. 6.2.а.).

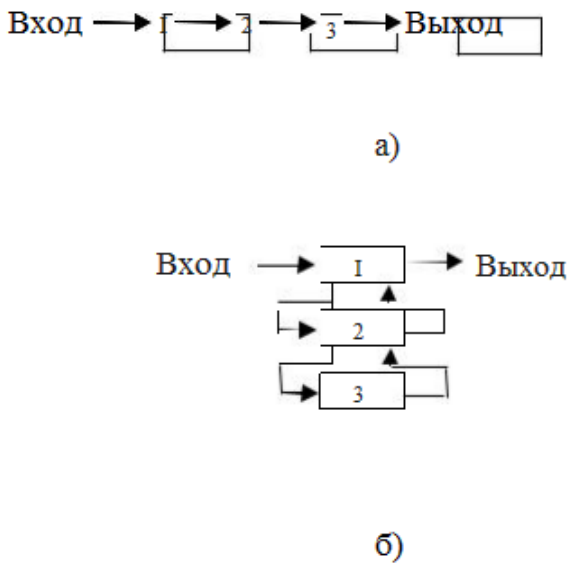


Рис. 6.2. Взаимодействие программ в маршрутах проектирования

При взаимодействии модулей разных уровней переход из *i*-ой программы в *i+1* происходит до завершения работы *i*-ой программы.

После завершения работ по (i+1)-й программе выполняется вновь возврат в *i*-ую, чаще всего *i*-ая программа заставляет вложенную в нее (i+1)-ю программу выполняться не один, а *f* раз, где *f* - частота использования (i+1)-й программы.

Разнесение прикладных программ по иерархическому уровню и установление частот их использования для типовых последовательностей программ способствует правильному решению ряда задач по созданию ПО САПР. К таким задачам относятся выбор языка программирования, выбор численных методов и выбор методов генерации рабочих программ.

Использование универсальных алгоритмических языков упрощает программирование, но снижает эффективность объектных программ. Программы, имеющие небольшие частоты использования *f*, целесообразно разрабатывать на алгоритмических языках. Программы низших иерархических уровней, имеющие высокие *f*, должны быть максимально эффективными, для чего целесообразно использовать язык Ассемблера.

Выбор численных методов и тщательность отработки алгоритмов зависят от уровня программы в иерархии ПО. Наибольшее внимание следует уделять отработке экономичности методов и алгоритмов для программ низших уровней.

Важное значение имеет выбор метода генерации рабочих программ: метод компиляции и метод интерпретации. При разработке программ используются элементы обоих методов. Чем выше частота использования *f*,

тем более обоснованным будет применение метода компиляции, т.к. это заметно сокращает затраты машинного времени.

6.4. Показатели качества программ проектирования, программные системы и комплексы

Одним из важнейших показателей эффективности программ проектирования являются затраты машинного времени Тм. Однако Тм зависит не только от особенностей ПКЗ но и от быстродействия применяемой ЭВМ. Поэтому более объективным будет показатель удельной трудоемкости вычислений:

$$N \quad \frac{1}{C}\frac{T_M}{T_{cp}}, \tag{6.2}$$

где $T_{cp} \quad \underline{1}_{n_i 1}^{n} \quad P_i t_i, P_i$ - вероятность появление команды i – го типа при решения задач рассматриваемого класса; t_i - время исполнения команды i –го типа; С-показатель сложности объекта проектирования.

Показатель удельной трудоемкости N определяется на специально подобранных тестовых примерах. Другим показателем эффективности является предельная сложность решаемых задач *Cmax*. Оценка сложности во многом определяется природой ПКЗ, но обычно это некоторый показатель объекта, обуславливающий размерность задачи: порядок системы уравнений, количество управляемых параметров и т.п. Показатель *Cmax* косвенно характеризует затраты машинной памяти Пм.

Имеются и другие показатели эффективности программ, такие как точность решения, степень универсальности программы как с позиций применимости к различным объектам проектирования, так и с позиций возможностей решения различных задач проектирования, удобство входного языка и т.п.

Пакеты прикладных программ (ППП), программные системы и комплексы. При структурировании ПО используют понятия ППП, программных систем, комплексов и компонентов. ППП - совокупность программ, объединенных общностью применения, т.е. возможностью совместного исполнения или ориентацией на определенный класс задач. Комплекс по определению в Единой системе программной документации (ЕСПД) - сложная программа, которую можно разделить на составные части. Компоненты - составные части программ, имеющие свое функциональное назначение. Понятия «комплекс-компонент»аналогичны понятиям «система-элемент»в блочно – иерархическом проектировании сложных систем, следовательно, на каждом иерархическом уровне проектирования ПО эти

понятия наполняются своим конкретным содержанием. Так, операционная система ОС ЕС - комплекс, а компилятор с ФОРТРАНа - его компонент. На уровне проектирования компилятора он рассматривается как комплекс, а синтаксический анализатор и генератор кода – как его компоненты.

Различают несколько типов ППП в зависимости от состава пакета. ППП *простой структуры* характеризуются наличием только обрабатывающей части набора функциональных программ, каждая из которых предназначена для выполнения некоторой проектной процедуры. Объединение нужных программ для реализации маршрутов проектирования происходит средствами ОС САПР на основе соответствующего языка управления заданиями. Совместное исполнение этих программ определяется возможностями организации их информационного интерфейса. В ППП простой структуры организация информационного интерфейса в значительной мере возлагается на пользователя. Такие пакеты просты в разработке, но неразвитость средств управления, информационного интерфейса и отсутствие удобного лингвистического обеспечения ограничивают их возможности. Они применяются лишь на маршрутах проектирования, на которых последовательно выполняемые операции достаточно автономны, информационные связи программ между собой и с пользователем достаточно слабые.

ППП *сложной структуры* и программные системы появились в результате развития прикладного ПО. В первых из них имеется собственная управляющая часть, называемая монитором, а во вторых, кроме этого, - языковой процессор с проблемно-ориентированным входным языком. Программные системы вместе с соответствующим лингвистическим и информационным обеспечением, предназначенные для реализации функций подсистемы САПР, называются программно-методическими комплексами (ПМК) САПР.

6.5. Основные принципы проектирования ПО САПР

Принципы модульности и иерархичности - основные принципы блочно иерархического проектирования сложных систем - позволяют организовывать коллективную параллельную разработку различных частей ПО, создавать открытые программные системы, облегчают их комплексную отладку и информационное согласование.

Модуль - структурная составляющая ПО, рассматриваемая как единое целое на определенных стадиях разработки или в процессе эксплуатации. Модули могут задаваться в текстовом виде, на некотором промежуточном языке загрузки, в машинных командах. Различают модули исходные, объектные и абсолютные.

Исходные - это модули на исходном языке текста.

Объектные (модули загрузки) - это модули, полученные в результате трансляции, однако не обработанные еще редактором внешних связей и загрузчиком.

Абсолютные - модули в истинных адресах оперативной памяти, записанные в машинных командах.

Ценность каждого модуля определяется тем, насколько часто он встречается в задачах данного класса.

Совокупность всех модулей пакета входит в тело пакета, который может также включать статистическую информацию об использовании различных компонентов пакета, каталоги, справочные данные и др. Различные компоненты пакета могут храниться на исходном языке, языке загрузки, в машинных кодах и на языке управления заданиями.

Сборка модулей в рабочую программу осуществляется с помощью системных средств пакета. Сборка может производиться в автоматическом режиме или по указанию пользователя. При сборке осуществляется стыковка модулей по управлению и информации.

При стыковке по управлению надо задать управляющей программе порядок включения в работу модулей собираемой программы. Кроме того, каждый модуль должен иметь возможность получать данные от другого модуля, считывая их с определенных областей оперативной памяти и помещая в соответствующие области свои результаты для другого блока. В этом и состоит установление информационных связей.

Требования, предъявляемые к модулям и средствам связи:

1. Независимость разработки, т.е. возможность вести разработку модулей без взаимного общения разработчиков.

2. Комбинируемость - возможность получения нового модуля при комбинировании имеющихся модулей средствами сборки. Это значительно сокращает сборку и упрощает получение готовых модулей.

3.Контекстная независимость - возможность проводить сборку программы, не зная, как работает каждый модуль. В большой системе всегда происходят изменения, и если при каждой сборке надо выяснять их, то ясно, что такая система неэффективна.

4.Информационная независимость. Если происходят изменения в некоторой группе данных, которые в модуле не используются, то, естественно, это не должно приводить к изменению такого модуля.

Существуют следующие способы сборки модулей текста в рабочую программу:

-библиотечный способ;

-сборка с изначальным заданием связей;

-сборка с изначальным заданием только информационных связей;

-сборка путем установления связей с помощью языка.

Простейший способ сборки основан на использовании библиотек и систем программирования. В библиотеке хранятся модули загрузки. Они транслируются независимо друг от друга и никак не связаны (ни по управлению, ни по информации). Пользователь сам в тексте своей программы организует эти связи указанием обращений к подпрограммам (связи по управлению) и за счет указаний фактических параметров в обращениях и переменных в общих (common-) блоках (связи по информации). В этом случае пользователь целиком берет на себя рутинную работу по сборке: он должен везде указать фактические параметры, внимательно распределить память в общих (common-) блоках.

Сборка с изначальным заданием связей основана на жестком задании связей как по памяти, так и по управлению, причем должны быть предприняты специальные меры для облегчения замены модулей. Такие системы эффективны в случае, когда схема задач остается практически неизменной. Заменяемые модули могут быть одинаковыми по входным и выходным данным, но могут отличаться по физической постановке реализуемой ими задачи (взята другая модель). Все информационные связи модулей при сборке концентрируются в одном месте - модуле памяти. Неудобство таких методов сборки состоит в том, что заранее должны быть запрограммированы все связи, предусмотрены все возможные варианты программ.

6.6. Лингвистическое обеспечение САПР

Лингвистическое обеспечение САПР включает в себя языки для представления информации о проектируемых объектах, процессе и средствах проектирования. Языки САПР делятся на языки программирования и проектирования.

Языки программирования используются для написания программ и применяются главным образом разработчиками САПР. Языки проектирования служат для описания информации об объектах и задачах проектирования и являются средством общения пользователя САПР с ЭВМ. Особую группу составляют языки описания управляющей информации для программно управляемого технологического оборудования (для фотонаборных установок, графопостроителей, металлообрабатывающих станков с ЧПУ и т.п.), называемые языками управления.

Языки могут быть процедурными и непроцедурными. Процедурные языки применяются для описания процессов в виде последовательностей

действий и процедур. В частности, большинство языков программирования служит для описания вычислительных процессов и потому относится к процедурным языкам. Непроцедурные языки применяются для описания семантических сетей, структур проектируемых объектов и других статических систем.

6.7. Языки программирования и проектирования.

Языки программирования. Среди языков программирования различают машинно-ориентированные, называемые языками ассемблера и автокодами, и алгоритмические языки высокого уровня. Автокод язык, предложения которого по структуре подобны машинным командам. Язык ассемблера - автокод, расширенный макрокомандами, выражениями, средствами, обеспечивающими модульность программ.

Использование машинно-ориентированных языков позволяет достигать наивысшей эффективности объектных программ с точки зрения затрат вычислительных ресурсов - машинного времени и памяти. Эти языки универсальны в смысле применимости к решению задач различных классов - научно-технических и экономических, системных и прикладных. Однако программирование на этих языках требует высокой квалификации программиста и приводит к увеличению сроков разработки прикладного ПО. Главный недостаток этих языков - непереносимость программ на ЭВМ с системой команд, отличной от той, на которую ориентирован язык.

Алгоритмические языки высокого уровня не зависят от типа ЭВМ и являются основным средством разработки прикладного ПО. В САПР наибольшее распространение получили языки ФОРТРАН, ПЛ/1, ПАСКАЛЬ, СИ, СИМУЛА-67и относительно новые -МОДУЛА-2,АДА, ЛИСП, ПРОЛОГ.

Язык ФОРТРАН относится к наиболее ранним (первая версия языка создана в 1954 г.) и простым алгоритмическим языкам, в нем нет средств для удобного описания разнообразных структур данных, запрещены рекурсивные обращения к процедурам, нет строгого описания языка. Несмотря на недостатки, ФОРТРАН широко использовался в САПР, особенно разработанных в 1960-1970-е годы, благодаря простоте разработки эффективных трансляторов. В настоящее время применяется усовершенствованная версия языка - ФОРТРАН-77.

Язык ПЛ/1 (создан в 1965 г.) обладает широкими возможностями описания различных процессов обработки данных, однако труден для освоения и для разработки эффективных трансляторов. Поэтому, хотя он и применяется при разработке ПО многих САПР, заметна заинтересованность в переходе к более совершенным языкам.

Язык ПАСКАЛЬ и его развитие, МОДУЛА-2,являются претендентами на роль основных языков для написания прикладного ПО. Положительные свойства этих языков - развитые средства для написания хорошо структурированных программ, для представления различных типов и структур данных, удачное сочетание простоты и строгости в описании языков.

Язык СИ является другим претендентом на роль основного языка программирования в САПР. Он сочетает черты языков высокого уровня и языков ассемблера, что делает удобным его применение при разработке общесистемного ПО. Язык СИ остается машинно-независимыми, следовательно, обеспечивает создание мобильных (переносимых) программ.

Язык АДА можно назвать наиболее универсальным среди созданных языков. В этот язык включены средства для описания параллельных процессов. Однако трансляторы с этого языка пока не получили достаточного распространения.

Язык СИМУЛА-67 сочетает в себе возможности универсального алгоритмического языка (в него как составная часть входит алгоритмический язык АЛГОЛ-60) и языка имитационного моделирования систем массового обслуживания. Его применение перспективно в ПМК на верхних иерархических уровнях проектирования.

Языки ЛИСП и ПРОЛОГ ориентированы на обработку символьной информации. С их помощью удобно описывать отношения между различными понятиями той или иной предметной области, программировать действия по выявлению заданных отношений, производить логические действия. Чаще всего эти языки используются в САПР для реализации ПМК, относящихся к экспертным системам.

Языки проектирования. Существует большое количество языков проектирования, которые делятся на входные, выходные, сопровождения, промежуточные, внутренние.

Входные языки служат для задания информации об объектах и задачах проектирования, передаваемой от человека к ЭВМ. В большинстве входных языков САПР можно выделить две части: непроцедурную, служащую для описания структур объектов, и процедурную, предназначенную для описания заданий на выполнение определенных проектных процедур. Языковые средства в этих двух частях составляют соответственно язык описания объекта (ЯОО) и язык описания заданий (ЯОЗ). Среди ЯОО различают языки описания схем, чертежей, процессов функционирования. Названия этих разновидностей ЯОО соответственно схемные, графические, моделирования.

Выходные языки используются для представления информации, идущей от ЭВМ к человеку.

Языки сопровождения применяются для корректировки и редактирования данных при выполнении проектных процедур. В диалоговых режимах работы с ЭВМ средства входного, выходного и языка сопровождения тесно связаны и объединяются под названием диалогового языка.

Промежуточные языки используются для описания информации о задачах проектирования на определенной стадии трансляции. Введение единого для ПМК промежуточного языка облегчает адаптацию комплекса к новым входным языкам, т.е. делает ПМК открытым по отношению к новым составляющим лингвистического обеспечения.

Внутренние языки являются языками внутреннего представления данных (ВПД). Введение единого ВПД означает принятие определенных соглашений об интерфейсах отдельных программ в ПМК и делает ПМК открытым по отношению к новым элементам ПО.

6.8. Языковые процессоры

Исполнение на ЭВМ заданий, представленных на каком-либо языке, отличном от машинного, требует преобразования информации, которое осуществляется программами или техническими устройствами интерпретаторами или трансляторами. Объединяющее название для интерпретаторов и трансляторов - ***языковые процессоры.***

Интерпретатор поочередно анализирует и исполняет задания, выраженные предложениями ***входного*** языка. В оперативной памяти ЭВМ при решении задачи присутствуют прикладная программа ***на входном*** языке и интерпретатор.

Транслятор преобразует заданную информацию с одного языка на другой. Программа на входе транслятора и ее язык называются исходными, на выходе транслятора - объектными. Если объектный язык - машинный или близкий к машинному, то трансляция и транслятор называются компиляцией и компилятором соответственно. Если исходный и объектный языки относятся к одному и тому же уровню языков, то транслятор называется конвертором. Решение задач по методу компиляции происходит в два этапа. Сначала в оперативной памяти размещаются исходная программа и компилятор, результатом работы компилятора будет рабочая программа. Затем скомпилированная рабочая программа исполняется.

При интерпретации циклических участков, неизбежно присутствующих в реальных программах, приходится многократно обрабатывать одни и те же предложения исходного текста, что обуславливает повышенные затраты машинного времени по сравнению с затратами при трансляции. Но в большинстве случаев интерпретация экономичнее по

затратам памяти, так как не расходует память под скомпилированную программу.

Трансляторы интерпретирующего типа. Вводится некоторый промежуточный язык, обладающий следующими свойствами: объектная программа на таком языке занимает меньший объем памяти, чем на машинном языке; время интерпретации промежуточного описания меньше, чем исходного. Тогда решение задачи осуществляется трансляцией на промежуточный язык с последующей интерпретацией получающегося промежуточного описания.

Другой способ сочетания трансляции и интерпретации - применение шаговых компиляторов, в которых исходная программа транслируется и исполняется достаточно крупными частями.

Структура трансляторов. Типичные функции трансляторов - контроль правильности исходной информации, генерация текста объектной программы. Процесс трансляции состоит из нескольких этапов, называемых фазами трансляции. Основные этапы - лексический и синтаксический анализ, генерация кода.

Лексический анализ, называемый также сканированием, служит для разделения исходного текста на отдельные элементарные языковые единицы - лексемы, в качестве которых фигурируют идентификаторы, числа, метки, знаки операций и т.п. Составляются таблицы лексем, используемые при дальнейшей трансляции. Выявляются недопустимые сочетания символов языка, которые нельзя выделить как лексемы (например, идентификатор, начинающийся с цифры, неразрешенный символ и т.п.).

Синтаксический анализ (грамматический разбор) - фаза, на которой проверяется соблюдение синтаксиса языка, т.е. правильности построения предложений. В процессе анализа должны выявляться все ошибки в исходном описании, которые можно обнаружить по формальным признакам, и выдаваться пользователю соответствующие диагностические сообщения. Результат синтаксического анализа - представление информации на промежуточном языке.

Генерация кода осуществляется генератором кода, который использует данные синтаксического анализа для построения объектной программы.

Контрольные вопросы

1. Что представляет собой ПО САПР?
2. Перечислите документы, которые входят в состав ПО САПР.
3. Какова структура общесистемного ПО?
4. Поясните классы системного ПО.

5. Приведите примеры операционных систем для ПЭВМ.
6. Какие функции выполняет программа управления задачами?
7. Какие функции выполняет программа управления заданиями?
8. Что представляет собой *ППП*?
9. Что характерно для *ППП* простой структуры?
10. Чем характеризуется *ППП* сложной структуры и программные системы?
11. Что называется программно-методическим комплексом САПР?
12. Какие функции выполняет операционная система?
13. Что включает в себе лингвистическое обеспечение?
14. Как различаются языки программирования?
15. Объясните режим работы языковых процессоров?

Лекция № 7.

Тема: Информационное обеспечения САПР (ИО), методическое обеспечение САПР (МО) и организационная обеспечения САПР (ОО).

План

7.1. Информационное обеспечение САПР.

7.2. Способы ведения ИФ.

7.3. Базы знаний (БЗ).

7.4. Методическое обеспечение САПР.

7.5. Организационное обеспечение САПР.

Ключевые слова: программные модули, банк данных, связь, уровень обращения, целостность, совокупность, искусственный интеллект, взаимодействие, экстенсиональная, интенсиональная.

7.1. Информационное обеспечение САПР

Информационное обеспечение САПР состоит из информационного фонда (ИФ) и средств управления этим фондом. ИФ включает в себя информацию, необходимую для выполнения АПР, и представляется в виде печатных документов, чертежей, файлов, микрофиш и т.п [4,5].

Система управления ИФ организует хранение и доступ к информации. Значительная часть ИФ предназначена для многоразового использования различными проектировщиками и различными прикладными программами в маршрутах проектирования.

Состав ИФ. В состав ИФ САПР входят:

а) программные модули, которые хранятся в виде объектных файлов. Как правило, эти данные мало изменяются в течение жизненного цикла САПР, имеют фиксированные размеры и появляются на этапе создания ИФ. Потребителями этих данных являются мониторы различных подсистем САПР;

б) исходные и результирующие данные, которые необходимы при выполнении программных модулей в процессе преобразования. Эти данные часто меняются в процессе проектирования, однако их тип постоянен и полностью определяется соответствующим программным модулем. При организации промежуточных данных возможны конфликтные ситуации в процессе согласования между собой данных различных типов;

в) нормативно-справочная проектная документация (НСПД), включающая в себя справочные данные о материалах, элементах схем, унифицированных узлах и конструкциях. Эти данные, как правило, хорошо структурированы. К НСПД относятся также ГОСТы и ОСТы, руководящие материалы и указания, типовые проектные решения, регламентирующие документы (слабо структурированные документальные данные);

г) содержание экранов дисплеев, представляющее собой связанную совокупность данных, задающих форму кадра и позволяющих отобразить на экран дисплея информацию с целью организации диалога в ходе проектирования. Обычно эти данные не изменяются в течение жизненного цикла САПР; имеют фиксированный размер и по своим характеристикам занимают промежуточное место между программными модулями (а) и исходными данными (б). Используются диалоговыми системами САПР в процессе реализации заданного графа диалога;

д) текущая проектная информация, отражающая состояние и ход выполнения проекта. Как правило, эта информация слабо структурирована, часто изменяется в процессе проектирования и представляется в форме исходных текстовых документов.

При выборе способов ведения ИФ САПР важно сформулировать принципы и определить средства ведения ИФ, структурирования данных, выбрать способы управления массивами данных.

7.2. Способы ведения ИФ

Различают следующие способы ведения ИФ САПР:

- использование файловой системы;
- построение библиотек;
- использование банков данных;
- создание информационных программ адаптеров.

Использование файловой системы и построение библиотек широко распространено в организации ИО вычислительных систем, так как поддерживается средствами ОС. Однако файловые и библиотечные системы неудобны и неэффективны при коллективном использовании большей части ИФ, когда требуется быстрая выборка отдельных записей, добавление и замена данных с сохранением их целостности. Целостность и правильное коллективное использование этой части ИФ достигаются при его организации в виде банка данных.

Банк данных есть система программных, языковых, организационных и технических средств для централизованного хранения, накопления и обновления данных, обеспечивающая прямой доступ и использование информации. Банки данных получили широкое распространение в виде самостоятельных информационно - поисковых или информационно-советующих систем. Наличие банка данных характерно для вычислительного центра коллективного пользования, где банк данных выступает в качестве самостоятельной единицы с собственным ТО.

Строение банка данных можно представить в виде базы данных (БД) и системы управления базой данных (СУБД).

БД можно определить как совокупность взаимосвязанных, хранящихся вместе данных при наличии такой минимальной избыточности, которая допускает их использование оптимальным образом для одного или нескольких приложений: данные запоминаются так, чтобы они были независимы от программ, их использующих; для добавления новых или модификации существующих данных, а также для поиска данных в БД применяется общий управляемый способ. Назначение БД заключается в том, чтобы одну и ту же совокупность данных можно было использовать для максимально возможного числа приложений. БД для САПР можно трактовать как хранилище такой информации, необходимость в которой возникает в процессе решения определенных задач проектирования.

БД рассматривается в виде совокупности физически и логически организованных данных, определенным образом связанных между собой и являющихся аналогом информационной модели семейства проектируемых объектов САПР.

Описание данных и отношений между ними бывает двух видов: физическое и логическое.

Физическое описание отражает совокупность элементов данных, а также связей между ними, осуществляемых в среде их хранения на жестком диске.

Логическое описание отражает совокупность элементов данных ***на уровне обращения*** к ним пользователя посредством программных средств, а также ***связей*** информационной модели объекта проектирования. Логическая организация или логическое описание данных подразделяются на файловое и глобальное. Глобальная организация (описание) содержит различные файлы данных и связи между ними, предназначенные для реализации функций банка данных. (Здесь: запись (логическая запись) - родственные данные, рассматриваемые как единица данных; файл - совокупность данных, состоящая из записей, относящихся к одной теме).

Файловая организация (описание) данных представляет собой набор записей и связей между ними, используемых прикладной программой при решении конкретной задачи пользователя.

Для обслуживания глобальной организации данных используется понятие администратора данных, отвечающего за их сохранность, функции которого обычно осуществляет определенное подразделение, имеющее в своем составе соответствующих специалистов.

Комплекс программ, служащих интерфейсом между пользователем и хранимыми данными, образует СУБД.

7.3. Базы знаний (БЗ)

БЗ - совокупность фактов, метаданных, т.е. описаний закономерностей, которым подчиняются данные, а также правил вывода, с помощью которых возможен вывод одних данных и метаданных на основе других. Его обеспечивают механизмы специальной программной системы - системы управления базами знаний (СУБЗ), в чьей среде создается и функционирует БЗ.

БЗ отражает систему понятий, их связей и зависимостей некоторой предметной области. БЗ предназначена для машинного вывода новой информации на основе известных фактов, содержащихся в БД, что возможно благодаря способности автоматизированной системы обрабатывать метаинформацию в БЗ. База данных представляет собой организованную семантическую сеть, составленную на языке, близком к естественному. Направление, в рамках которого решаются научные и практические задачи организации БЗ, называют искусственным интеллектом.

Основные исследования, которые ведутся в области искусственного интеллекта, можно свести к следующим четырем направлениям:

1. Представление знаний и работа с ними (создание моделей и языков для представления знаний в ЭВМ, а также программных и аппаратных средств для их преобразования: пополнения, логической обработки и т.д.).

2. Планирование целесообразного поведения (исследования по созданию методов формирования целей и решения задач планирования

действий автоматического устройства, функционирующего в сложной внешней среде).

3. Общение человека с ЭВМ (задачи создания языковых средств, так называемого «дружественного интерфейса», позволяющего эффективно взаимодействовать с ЭВМ непрограммирующему пользователю).

4. Распознавание образов и обучение (исследования по восприятию зрительной, слуховой и других видов информации, методам ее обработки, способам адаптации искусственных технических систем к среде путем обучения).

Любая интеллектуальная деятельность опирается на знания о предметной области, в которой ставятся и решаются задачи. Роль знаний в интеллектуальной деятельности определяет характерную особенность интеллектуальных систем - наличие в них блока представления знаний и интеллектуального банка данных (ИБД), который, в свою очередь, подразделяют на БД и БЗ. БД включает фотографические, количественные данные, характеризующие предметную область [6,7].

Переход от данных к знаниям - логическое следствие развития и усложнения информационных структур, обрабатываемых на ЭВМ. Рассмотрим особенности знаний, которые отличают их от данных.

Интерпретируемость. Данные, помещенные в ЭВМ, могут интерпретироваться лишь соответствующими программами. В отрыве от программ они не несут никакой содержательной информации. Знания отличаются от данных тем, что возможность содержательной интерпретации присутствует в них всегда.

Наличие классифицирующих отношений. Несмотря на разнообразие форм хранения данных, возможности компактного описания всех связей между различными типами данных ограничены. При переходе к знаниям в БЗ устанавливаются многочисленные разнообразные отношения между единицами знаний (например, элемент - множество, тип - подтип, ситуация – под ситуация и т.п.), отражающие характер их взаимосвязи. Это позволяет записать и хранить отдельно информацию, одинаковую для всех элементов множества, но при необходимости ее можно непосредственно передать описанию любого элемента множества. Такой процесс передачи называют «наследованием» информации.

Наличие ситуативных связей. Эти связи определяют ситуативную совместимость отдельных событий или фактов, хранимых или вводимых в память, и позволяют строить процедуры анализа знаний.

Появление знаний как информационных объектов для обработки на ЭВМ определило переход от БД к БЗ. Системы управления базами знаний

(СУБЗ) являются развитием СУБД и имеют более мощные обслуживающие процедуры. В частности, с помощью СУБЗ пользователь может работать не только с теми структурами информации, которые реализованы в БЗ, но и создавать свои. СУБЗ автоматически обеспечивает связь между структурами пользователя и структурами, хранимыми в БЗ.

Следует отметить, что БЗ не отвергает и не заменяет БД. Они рассматриваются как разные уровни представления информации, хранящейся в ИБД. Основным принципом организации ИБД, построенным, например, на основе семантических сетей, является разделение экстенсиональных и интенсиональных знаний; при этом экстенсиональная семантическая сеть является основой БД, а интенсиональная - основой БЗ. Такое «расслоение» системы представления знаний существенно повышает ее возможности по сравнению, например, с логическими моделями.

7.4. Методическое обеспечение САПР.

Под методическим обеспечением (МетО) САПР понимают входящие в ее состав документы, регламентирующие ***порядок ее эксплуатации***. Причем документы, относящиеся к процессу создания САПР, не входят в состав МетО. Так как документы МетО носят в основном инструктивный характер и их разработка является процессом творческим, то о специальных способах и средствах реализации данного компонента САПР говорить не приходится. В последнее время совершенствование организации работ в области автоматизации проектирования направлено на централизованное создание типовых ПМК с целью широкого тиражирования. Такие ПМК должны включать, наряду с программами для ЭВМ и базами данных, комплекты документации. Таким образом, указанная документация станет частью методического обеспечения САПР.

7.5. Организационное обеспечение САПР

Стандарты по САПР требуют выделения в качестве самостоятельного компонента ОО, которое включает в себя положения, инструкции, приказы, штатные расписания, квалификационные требования и другие документы, регламентирующие организационную ***структуру подразделений*** проектной организации и ***взаимодействие*** подразделений с комплексом средств АПР.

Функционирование САПР возможно только при наличии и взаимодействии перечисленных средств АПР.

Контрольные вопросы

1. Что включает в себе информационное обеспечение?
2. Объясните состав информационного обеспечения.
3. Как различаются способы ведения ИФ?
4. Что отражает логическое описание?

5. В каких направлениях ведутся исследования в области искусственного интеллекта?

6. Что означает интерпретируемость?

7. Что понимается под методическом обеспечении?

8. Что включает в себе организационное обеспечение?

9. Выходит ли в состав МетО документы, которые относящиеся к процессу создание САПР?

10. Что автоматически обеспечивает СУБЗ?

Лекция № 8.
Тема: Техническое обеспечение САПР (ТО). Выбор технических средств при проектирование САПР.

План

8.1. Техническое обеспечение (ТО) и состав ТС САПР.

8.2. Уровни ТО САПР.

8.3. Структура технического обеспечения. Требования, предъявляемые к техническому обеспечению.

8.4. Эталонная модель взаимосвязи открытых систем.

8.5. Вычислительные системы в САПР.

8.6. Периферийные устройства.

Ключевые слова: дигитайзер, графопостроитель, координатографы, сервер, принтер, плоттер алфавитно-цифровые печатные устройства, устройства микрофильмирования, периферийные устройства, мультиплексирование, коммутация, эталонный модель, инкапсуляция, декапсуляции.

8.1. Техническое обеспечение (ТО) и состав ТС САПР.

Техническое обеспечение (ТО) САПР представляет собой совокупность ТС, среди которых выделяют несколько групп устройств.

Состав ТС САПР. Устройства программной обработки данных включают ЭВМ (процессоры, оперативные и внешние запоминающие устройства), осуществляющие прием данных с устройств ввода или КАНАЛОВ связи, их обработку, накопление и выдачу на устройства отображения.

Устройства подготовки и ввода данных с промежуточных носителей включают устройства подготовки данных на магнитных носителях, кодировщики графической информации (дигитайзеры).

Устройства вывода, документирования данных и архива проектных решений выполняют функции вывода результатов решения задач из ЭВМ и представления их в форме необходимых документов. К ним относятся различные алфавитно-цифровые печатные устройства (АЦПУ),

графопостроители, координатографы, устройства микрофильмирования, фотонаборные установки для изготовления фотошаблонов.

Устройства оперативного взаимодействия человека с ЭВМ, служащие для ввода и вывода данных в интерактивном режиме и их редактирования. Представлены алфавитно-цифровыми(символьными) и графическими дисплеями, клавиатурой, мышью.

Устройства передачи данных предназначены для обеспечения совместной работы технических средств в составе вычислительных систем и сетей. К ним относятся аппаратура передачи данных, различные устройства сопряжения, адаптеры, мультиплексоры передачи данных.

Выбор типов и числа различных устройств для САПР производят, исходя из требований: достаточной производительности и емкости памяти для выполнения всех процедур АПР; обеспечения одновременной работы необходимого числа проектировщиков путем организации соответствующего числа автоматизированных рабочих мест (АРМ), а также необходимого информационного взаимодействия подразделений проектного предприятия между собой и доступа к соответствующим информационным и программным ресурсам САПР; укомплектования рабочих мест аппаратурой, обеспечивающей взаимодействие с ЭВМ в удобной для человека форме.

8.2. Уровни ТО САПР

Для выполнения перечисленных выше требований структура ТО САПР должна быть многоуровневой. Выделяют следующие уровни ТО САПР.

Уровень центрального вычислительного комплекса (ЦВК) предназначен для решения наиболее сложных в вычислительном отношении задач. Это задачи математической физики, решаемые на компонентном иерархическом уровне проектирования; моделирования сложных функциональных и принципиальных схем; переборные; конструкторского проектирования и др., требующие больших объемов вычислений. ЦВК комплектуется преимущественно ЭВМ высокой производительности, часто несколько таких ЭВМ объединяют в многомашинный вычислительный комплекс.

Уровень интерактивно-графического комплекса (ИГК) предназначен для обеспечения оперативного взаимодействия проектировщика с ЭВМ (ввода и редактирования данных, просмотра им результатов выполнения проектных процедур) и решения простых и умеренных по сложности задач. ИГК состоит из нескольких программно-технических комплексов, представляющих собой АРМ, состоящее из ЭВМ средней производительности, набора периферийных устройств, обеспечивающих ввод и вывод информации в символьной и графической форме, и необходимого ПО.

Уровень технологического комплекса (ТК) выделяется в структуре ТО САПР с целью сосредоточения устройств подготовки носителей для программно-управляемого технологического оборудования и получения

комплектов документации. Например, в САПР БИС в состав ТК включают ЭВМ с внешней памятью большого объема, графопостроители, координатографы, устройства оперативной связи, фотонаборные установки для изготовления фото оригиналов. ТК САПР является связующим звеном между САПР и производством изделий.

Распределение функций между уровнями ТО САПР в значительной мере зависит от технических характеристик используемых ЭВМ и периферийного оборудования.

8.3. Структура технического обеспечения. Требования, предъявляемые к техническому обеспечению

Техническое обеспечение САПР включает в себя различные технические средства (hardware), используемые для выполнения автоматизированного проектирования, а именно ЭВМ, периферийные устройства, сетевое оборудование, а также оборудование некоторых вспомогательных систем (например, измерительных), поддерживающих проектирование.

Используемые в САПР технические средства должны обеспечивать:

1) выполнение всех необходимых проектных процедур, для которых имеется соответствующее ПО;

2) взаимодействие между проектировщиками и ЭВМ, поддержку интерактивного режима работы;

3) взаимодействие между членами коллектива, работающими над общим проектом.

Первое из этих требований выполняется при наличии в САПР вычислительных машин и систем с достаточными производительностью и емкостью памяти.

Второе требование относится к пользовательскому интерфейсу и выполняется за счет включения в САПР удобных средств ввода-вывода данных и прежде всего устройств обмена графической информацией.

Третье требование обусловливает объединение аппаратных средств САПР в вычислительную сеть.

В результате общая структура ТО САПР представляет собой сеть узлов, связанных между собой средой передачи данных (рис. 8.1). Узлами (станциями данных) являются рабочие места проектировщиков, часто называемые *автоматизированными рабочими местами {АРМ)* или *рабочими станциями (WS - Workstation),* ими могут быть также большие ЭВМ (мейнфреймы),

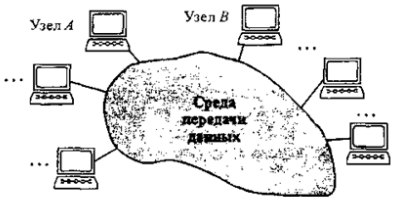


Рис.8.1. Структура технического обеспечения САПР
отдельные периферийные и измерительные устройства.

Именно в АРМ должны быть средства для интерфейса проектировщика с ЭВМ. Что касается вычислительной мощности, то она может быть распределена между различными узлами вычислительной сети.

Среда передачи данных представлена каналами передачи данных, состоящими из линий связи и коммутационного оборудования.

В каждом узле можно выделить *оконечное оборудование данных* (*ООД),* выполняющее определенную работу по проектированию, и *аппаратуру окончание канала данных (АКД*), предназначенную для связи ООД со средой передачи данных. Например, в качестве ООД можно рассматривать персональный компьютер, а в качестве АКД - вставляемую в компьютер сетевую плату.

Канал передачи данных - средство двустороннего обмена данными, включающее в себя АКД и линию связи. *Линией связи* называют часть физической среды, используемую для распространения сигналов в определенном направлении; примерами линий связи могут служить коаксиальный кабель, витая пара проводов, волоконно-оптическая линия связи (ВОЛС). Близким является понятие *канала (канала связи),* под которым понимают средство односторонней передачи данных. Примером канала связи может быть полоса частот, выделенная одному передатчику при радиосвязи. В некоторой линии можно образовать несколько каналов связи, по каждому из которых передается своя информация. При этом говорят, что линия разделяется между несколькими каналами.

Типы сетей. Существуют два метода разделения линии передачи данных: *временное мультиплексирование* (иначе разделение по времени, или *TDM - Time Division Method*), при котором каждому каналу выделяется некоторый квант времени, и *частотное разделение (FDM — Frequency Division Method),* при котором каналу выделяется некоторая полоса частот.

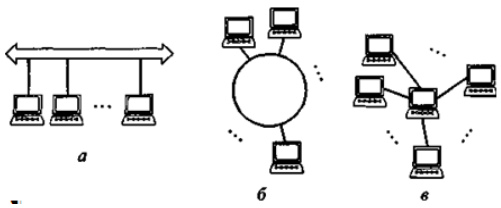


Рис. 8.2. Варианты топологии локальных вычислительных сетей: *а* - шинная; б - кольцевая; в - звездная

В САПР небольших проектных организаций, насчитывающих не более единиц - десятков компьютеров, которые размещены на малых расстояниях один от другого (например, в одной или нескольких соседних комнатах), объединяющая компьютеры сеть является локальной. *Локальная вычислительная сеть {ЛВС*), или *LAN {Local Area Network),* имеет линию связи, к которой подключаются все узлы сети. При этом топология соединений узлов (рис. 8.2) может быть шинная (bus), кольцевая (ring), звездная (star). Протяженность линии и число подключаемых узлов в ЛВС ограничены.

В более крупных по масштабам проектных организациях в сеть включены десятки-сотни и более компьютеров, относящихся к разным проектным и управленческим подразделениям и размещенных в помещениях одного или нескольких зданий. Такую сеть называют *корпоративной.* В ее структуре можно выделить ряд ЛВС, называемых *подсетями,* и средства связи ЛВС между собой. В эти средства входят коммутационные серверы (блоки взаимодействия подсетей). Если коммутационные серверы объединены отделенными от ЛВС подразделений каналами передачи данных, то они образуют новую подсеть, называемую *опорной* (или транспортной), а вся сеть оказывается иерархической структуры.

Если здания проектной организации удалены друг от друга на значительные расстояния (вплоть до их расположения в разных городах), то корпоративная сеть по своим масштабам становится *территориальной сетью {WAN - Wide Area Network).* В территориальной сети различают *магистральные* каналы передачи данных (магистральную сеть), имеющие значительную протяженность, и каналы передачи данных, связывающие ЛВС (или совокупность ЛВС отдельного здания или кампуса) с магистральной сетью и называемые *абонентской линией* или соединением *«последней мили».*

Обычно создание выделенной магистральной сети, т.е. сети, обслуживающей единственную организацию, обходится для нее слишком дорого. Поэтому чаще прибегают к услугам провайдера, т.е. организации, предоставляющей телекоммуникационные услуги многим пользователям. В этом случае внутри корпоративной сети связь на значительных расстояниях осуществляется через *магистральную сеть общего пользования.* В качестве такой сети можно использовать, например, городскую или междугородную телефонную сеть или территориальные сети передачи данных. Наиболее распространенной формой доступа к этим сетям в настоящее время является обращение к глобальной вычислительной сети Internet [4,5].

Для многих корпоративных сетей возможность выхода в Internet является желательной не только для обеспечения взаимосвязи удаленных сотрудников собственной организации, но и для получения других информационных услуг. Развитие виртуальных предприятий, работающих на основе CALS-технологий, с необходимостью подразумевает информационные обмены через территориальные сети, как правило, через Internet. Нужно, однако, отметить, что использование сетей общего

пользования существенно усложняет задачу обеспечения информационной безопасности.

Структура ТО САПР для крупной организации представлена на рис. 8.3. Здесь показана типичная структура крупных корпоративных сетей САПР, называемая архитектурой *клиент - сервер.* В сетях клиент - сервер выделяется один или несколько узлов, называемых *серверами,* которые выполняют в сети управляющие или общие для многих пользователей проектные функции, а остальные узлы (рабочие места) являются терминальными, их называют *клиентами,* в них работают пользователи. В общем случае сервером называют совокупность программных средств, ориентированных на выполнение определенных функций, но если эти средства сосредоточены на конкретном узле вычислительной сети, то тогда понятие «сервер» относится именно к узлу сети.

Сети клиент - сервер различают по характеру распределения функций между серверами, другими словами, их классифицируют по типам серверов. Различают *файл-серверы* для хранения файлов, разделяемых многими пользователями, *серверы баз данных* АС, *серверы приложений* для решения конкретных прикладных задач, *коммутационные серверы* (называемые также блоками взаимодействия сетей или серверами доступа) для взаимосвязи сетей и подсетей, *специализированные серверы* для выполнения определенных телекоммуникационных услуг, например серверы электронной почты.

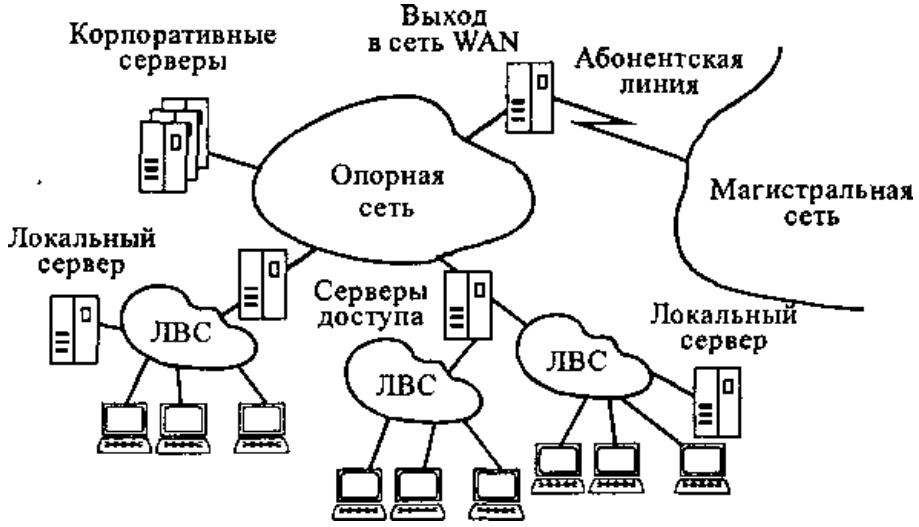


Рис. 8.3. Структура корпоративной сети САПР

В случае специализации серверов по определенным приложениям сеть называют *сетью распределенных вычислений.* Если сервер приложений обслуживает пользователей одной ЛВС, то естественно назвать такой сервер локальным. Но поскольку в САПР имеются приложения и базы данных, разделяемые пользователями разных подразделений и, следовательно, клиентами разных ЛВС, то соответствующие серверы относят к группе корпоративных, подключаемых обычно к опорной сети (см. рис. 8.3).

Наряду с архитектурой клиент - сервер применяют одноранговые сети, в которых любой узел в зависимости от решаемой задачи может выполнять функции как сервера, так и клиента. Организация взаимодействия в таких сетях при числе узлов более нескольких десятков становится довольно сложной, поэтому одноранговые сети нашли преимущественное распространение в небольших по масштабам САПР.

В соответствии со способами коммутации различают сети с *коммутацией каналов* и *коммутацией пакетов.* В первом случае при обмене данными между узлами AuB в сети создается физическое соединение между $Am\ B,$ которое во время сеанса связи используется только этими абонентами. Примером сети с коммутацией каналов может служить телефонная сеть. Здесь передача информации происходит быстро, но каналы связи используются неэффективно, так как при обмене данными возможны длительные паузы и канал «простаивает». При коммутации пакетов физического соединения, которое в каждый момент сеанса связи соединяло бы абонентов *А* и *В,* не создается. Сообщения разделяются на порции, называемые *пакетами,* которые передаются в разветвленной сети от Л к *В* или обратно через промежуточные узлы с возможной буферизацией (временным запоминанием) в них. Таким образом, любая линия может разделяться многими сообщениями, попеременно пропуская при этом пакеты разных сообщений с максимальным заполнением упомянутых пауз.

8.4. Эталонная модель взаимосвязи открытых систем

Для удобства модернизации сложных информационных систем их делают максимально открытыми, т.е. приспособленными для внесения изменений в некоторую часть системы при сохранении неизменными остальных частей. В отношении вычислительных сетей реализация концепции открытости привела к появлению *эталонной модели взаимосвязи открытых систем (ЭМВОС*), предложенной Международной организацией стандартизации *(ISO - International Standard Organization).* В этой модели дано описание общих принципов, правил, соглашений, обеспечивающих взаимодействие информационных систем и называемых *протоколами.*

Информационную сеть в ЭМВОС рассматривают как совокупность функций (протоколов), которые подразделяют на группы, называемые *уровнями.* Именно разделение на уровни позволяет вносить изменения в средства реализации одного уровня без перестройки средств других уровней, что значительно упрощает и удешевляет модернизацию средств по мере развития техники.

Различают семь уровней ЭМВОС.

На *физическом* (*physical*) уровне осуществляется представление информации в виде электрических или оптических сигналов, преобразование формы сигналов, выбор параметров физических сред передачи данных, организуется передача информации через физические среды.

На *канальном (link)* уровне выполняется обмен данными между соседними узлами сети, т.е. узлами, непосредственно связанными

физическими соединениями без других промежуточных узлов. Отметим, что пакеты канального уровня обычно называют *кадрами.*

На *сетевом (network)* уровне происходит формирование пакетов по правилам тех промежуточных сетей, через которые проходит исходный пакет, и *маршрутизация* пакетов, т.е. определение и реализация маршрутов, по которым передаются пакеты. Другими словами, маршрутизация сводится к образованию логических каналов. *Логическим каналом* называют виртуальное соединение двух или более объектов сетевого уровня, при котором возможен обмен данными между этими объектами. Понятию логического канала необязательно соответствует физическое соединение линий передачи данных между связываемыми пунктами. Это понятие введено для абстрагирования от физической реализации соединения. Еще одной важной функцией сетевого уровня после маршрутизации является контроль нагрузки на сеть с целью предотвращения перегрузок, отрицательно влияющих на работу сети.

На *транспортном (transport)* уровне обеспечивается связь между оконечными пунктами (в отличие от предыдущего сетевого уровня, на котором обеспечивается передача данных через промежуточные компоненты сети). К функциям транспортного уровня относятся мультиплексирование и демультиплексирование (сборка-разборка сообщений на пакеты в конечных пунктах), обнаружение и устранение ошибок в переданных данных, задание требуемого уровня услуг (например, заказанных скорости и надежности передачи).

На *сеансовом* (*session*) уровне определяются тип связи (дуплекс или полудуплекс), начало и окончание заданий, последовательность и режим обмена запросами и ответами взаимодействующих партнеров.

На *представительном* (*presentation*) уровне реализуются функции представления данных (кодирование, форматирование, структурирование). Например, на этом уровне выделенные для передачи данные преобразуются из одного кода в другой, в частности, с целью шифрования.

На *прикладном* (*application*) уровне определяются и оформляются в сообщения те данные, которые подлежат передаче по сети.

В конкретных случаях может возникать потребность в реализации лишь части названных функций, тогда соответственно сеть будет содержать лишь часть уровней. Так, в простых (неразветвленных) ЛВС отпадает необходимость в средствах сетевого и транспортного уровней. Одновременно сложность функций канального уровня делает целесообразным его разделение в ЛВС на два подуровня: управление доступом к каналу (*MAC - Medium Access Control*) и управление логическим каналом (*LLC - Logical Link Control*). К подуровню LLC в отличие от подуровня MAC относится часть функций канального уровня, не зависящих от особенностей передающей среды.

Передача данных через разветвленные сети происходит при использовании *инкапсупяции-декапсуляции* порций данных. Так, сообщение,

пришедшее на транспортный уровень, делится на сегменты, которые получают заголовки и передаются на сетевой уровень. Сегментом обычно называют пакет транспортного уровня. Сетевой уровень организует передачу данных через промежуточные сети. Для этого сегмент может быть разделен на части (пакеты), если сеть не поддерживает передачу сегментов целиком. Пакет снабжается своим сетевым заголовком (т. е. происходит инкапсуляция сегмента в пакет сетевого уровня). При передаче между узлами промежуточной ЛВС требуется инкапсуляция пакетов в кадры с возможной разбивкой пакета. Приемник декапсулирует сегменты и восстанавливает исходное сообщение.

8.5. Вычислительные системы в САПР

В качестве средств обработки данных в современных САПР широко используют рабочие станции, серверы, персональные компьютеры. Применение больших ЭВМ и в том числе суперЭВМ нехарактерно, так как они дороги и их отношение производительность - цена существенно ниже подобного показателя серверов и многих рабочих станций.

На базе рабочих станций или персональных компьютеров создают АРМ.

Типичный состав устройств АРМ: ЭВМ с одним или несколькими микропроцессорами, дисковой, оперативной и кэш-памятью и шинами, служащими для взаимной связи устройств; устройства ввода-вывода, включающие в себя, как минимум, клавиатуру, мышь, дисплей; дополнительно в состав АРМ могут входить принтер, сканер, плоттер (графопостроитель) и некоторые другие периферийные устройства.

Память ЭВМ обычно имеет иерархическую структуру. Поскольку в памяти большого объема трудно добиться одновременно высокой скорости записи и считывания данных, память делят на сверхбыстродействующую кэш-память малой емкости, основную оперативную память умеренного объема и сравнительно медленную внешнюю память большой емкости, причем, в свою очередь, кэш-память часто разделяют на кэш первого и второго уровней.

Например, в персональных компьютерах на процессорах Pentium III кэш первого уровня имеет по 16 К байт для данных и для адресов, он и кэш второго уровня емкостью 256 К байт встроены в процессорный кристалл, емкость оперативной памяти составляет десятки-сотни мегабайтов.

Для связи наиболее быстродействующих устройств (процессора, оперативной и кэш-памяти, видеокарты) используется системная шина с пропускной способностью до 1 ...2 Гбайт/с. Кроме системной шины на материнской плате компьютера имеются шина расширения для подключения сетевого контроллера и быстрых внешних устройств (например, шина PCI с пропускной способностью 133 Мбайт/с) и шина медленных внешних устройств, таких, как клавиатура, мышь, принтер и т. п.

Рабочие станции по сравнению с персональными компьютерами представляют собой вычислительную систему, специализированную на

выполнение определенных функций. Специализация обеспечивается как набором программ, так и аппаратно за счет использования дополнительных специализированных процессоров. Так, в САПР для машиностроения преимущественно применяют графические рабочие станции для выполнения процедур геометрического моделирования и машинной графики. Эта направленность требует мощного процессора, высокоскоростной шины, памяти достаточно большой емкости.

Высокая производительность процессора необходима по той причине, что графические операции (например, перемещения изображений, повороты, удаление скрытых линий и др.) часто выполняются по отношению ко всем элементам изображения. Такими элементами в трехмерной (3*D)* графике при аппроксимации поверхностей полигональными сетками являются многоугольники, их число может превышать 10^4. В то же время для удобства работы проектировщика в интерактивном режиме задержка при выполнении команд указанных выше операций не должна превышать нескольких секунд. Но поскольку каждая такая операция по отношению к каждому многоугольнику реализуется большим числом машинных команд, требуемое быстродействие составляет десятки миллионов машинных операций в секунду. Такое быстродействие при приемлемой цене достигается применением наряду с основным универсальным процессором также дополнительных специализированных (*графических*) *процессоров,* в которых определенные графические операции реализуются аппаратно.

В наиболее мощных рабочих станциях в качестве основных обычно используют высокопроизводительные микропроцессоры с сокращенной системой команд (с RISC-архитектурой), работающие под управлением одной из разновидностей операционной системы Unix. В менее мощных все чаще используют технологию Wintel (т.е. микропроцессоры Intel и операционные системы Windows). Графические процессоры выполняют такие операции, как, например, растеризация - представление изображения в растровой форме для его визуализации, перемещение, вращение, масштабирование, удаление скрытых линий ит. п.

Типичные характеристики рабочих станций: несколько процессоров, сотни мегабайтов оперативной и десятки гигабайтов внешней памяти, наличие кэш-памяти, системная шина со скоростями от сотен мегабайтов в секунду до 1... 2 Гбайт/с.

В зависимости от назначения существуют АРМ конструктора, АРМ технолога, АРМ руководителя проекта и т. п. Они могут различаться составом периферийных устройств, характеристиками ЭВМ.

В АРМ конструктора (графических рабочих станциях) используются растровые мониторы с цветными трубками. Типичные значения характеристик мониторов находятся в следующих пределах: размер экрана по диагонали 17... 24 дюйма (фактически изображение занимает площадь на 5 ... 8 % меньше, чем указывается в паспортных данных). Разрешающая способность монитора, т. е. число различимых пикселей (отдельных точек, из

которых состоит изображение), определяется шагом между отверстиями в маске, через которые проходит к экрану электронный луч в электронно-лучевой трубке. Этот шаг находится в пределах 0,21 ... 0,28 мм, что соответствует количеству пикселей изображения от 800 х 600 до 1920 х 1200 и более. Чем выше разрешающая способность, тем шире должна быть полоса пропускания электронных блоков видеосистемы при одинаковой частоте кадровой развертки. Полоса пропускания видео усилителя находится в пределах 110 ... 150 МГц, и потому частота кадровой развертки *обычно снижается с 135 Гц для разрешения 640* х 480 до 60 Гц для разрешения 1600 х 1200. Отметим, что чем ниже частота кадровой развертки, а это есть частота регенерации изображения, тем заметнее мерцание экрана. Желательно, чтобы эта частота была не ниже 75 Гц.

Специально выпускаемые ЭВМ как серверы высокой производительности обычно имеют структуру симметричной многопроцессорной вычислительной системы. В них системная память разделяется всеми процессорами, каждый процессор может иметь свою сверхоперативную память сравнительно небольшой емкости, число процессоров невелико (единицы, редко более десяти). Например, сервер Enterprise 250 (Sun Microsystems) имеет один-два процессора, его цена в зависимости от комплектации колеблется в диапазоне 24 ... 56 тыс. долл., а сервер Enterprise 450 с четырьмя процессорами стоит от 82 до 95 тыс. долл.

8.6. Периферийные устройства

Для ввода графической информации с имеющихся документов в САПР используют дигитайзеры и сканеры.

Дигитайзер применяют для ручного ввода. Он имеет вид кульмана, по его электронной доске перемещается курсор, на котором расположены визир и кнопочная панель. Курсор имеет электромагнитную связь с сеткой проводников в электронной доске. При нажатии кнопки в некоторой позиции курсора происходит занесение в память информации о координатах этой позиции. Таким образом может осуществляться ручная «сколка» чертежей.

Для автоматического ввода информации с имеющихся текстовых или графических документов используют *сканеры* планшетного или протяжного типа. Способ считывания оптический. В сканирующей головке размещаются оптоволоконные самофокусирующиеся линзы и фотоэлементы. Разрешающая способность в разных моделях составляет от 300 до 800 точек на дюйм (этот параметр часто обозначают dpi). Считанная информация имеет растровую форму, программное обеспечение сканера представляет ее в одном из стандартных форматов, например TIFF, GIF, PCX, JPEG, и для дальнейшей обработки может выполнить векторизацию - перевод графической информации в векторную форму, например в формат DXF.

Для вывода информации применяют *принтеры* и *плоттеры*. Первые из них ориентированы на получение документов малого формата (А3, А4), вторые - на вывод графической информации на широкоформатные носители.

В этих устройствах преимущественно используется растровый (т.е. построчный) способ вывода со струйной технологией печати. Печатающая система в струйных устройствах включает в себя картридж и головку. Картридж - баллон, заполненный чернилами (в цветных устройствах имеется несколько картриджей, каждый с чернилами своего цвета). Головка - матрица из сопел, из которых мельчайшие чернильные капли поступают на носитель. Физический принцип действия головки термический или пьезоэлектрический. При термопечати выбрасывание капель из сопла происходит под действием его нагревания, что вызывает образование пара и выбрасывание капель под давлением. При пьезоэлектрическом способе пропускание тока через пьезоэлемент приводит к изменению размера сопла и выбрасыванию капли чернил. Второй способ дороже, но позволяет получить более высококачественное изображение.

Типичная разрешающая способность принтеров и плоттеров 300 dpi, в настоящее время она повышена до 720dpi. В современных устройствах управление осуществляется встроенными микропроцессорами. Типичное время вывода монохромного изображения формата А1 находится в пределах 2 ... 7 мин, цветного - в 2 раза больше.

Дигитайзеры, сканеры, принтеры, плоттеры могут входить в состав АРМ или разделяться пользователями нескольких рабочих станций в составе локальной вычислительной сети.

Контрольные вопросы

1. Поясните состав и назначение устройств графической рабочей станции.
2. Что такое «растеризация» и «векторизация»?
3. Что такое «промышленный компьютер»? Каковы его особенности?
4. Дайте сравнительную характеристику методов коммутации каналов и коммутации пакетов.
5. В чем заключается сущность методов временного (TDM) и частотного (FDM) разделения каналов?
6. Почему в МДКН/ОК повторные попытки захвата линии разрешаются через случайные интервалы времени?
7. Рассчитайте размер окна столкновений в сети 10Base-5, если линия передачи данных представлена одним сегментом кабеля длиной 500 м.
8. Что такое «стаффинг»?
9. В чем сущность метода предотвращения конфликтов в RadioEthemet?
10. Почему способ кодирования 4Ь/5Ь или 8Ь/10Ь позволяет увеличить информационную скорость передачи данных?

Лекция № 9.

Тема: Математическое обеспечение анализа проектных решений. Компоненты математического обеспечения. Математические модели в процедурах анализа на макроуровне.

План

9.1. Компоненты математического обеспечения. Математический аппарат в моделях разных иерархических уровней.

9.2. Математические модели в процедурах анализа на макроуровне.

9.3. Представление топологических уравнений.

9.4. Характеристика методов формирования ММС.

Ключевые слова: микроуровень, макроуровень, дискретизация, функционвльно - логический уровень, адекватность, экономичность, фазовые переменные, математический модель, базисные переменные, ветв дерева, хорд, алгебризация.

9.1. Компоненты математического обеспечения.

Математический аппарат в моделях разных иерархических уровней. К МО анализа относятся математические модели, численные методы, алгоритмы выполнения проектных процедур.

Компоненты МО определяются базовым математическим аппаратом, специфичным для каждого из иерархических уровней проектирования.

На *микроуровне* типичные математические модели представлены дифференциальными уравнениями в частных производных вместе с краевыми условиями. К этим моделям, называемым *распределенными,* относятся многие уравнения математической физики. Объектами исследования здесь являются поля физических величин, что требуется при анализе прочности строительных сооружений или машиностроительных деталей, исследовании процессов в жидких средах, моделировании концентраций и потоков частиц в электронных приборах и т. п.

Число совместно исследуемых различных сред (число деталей, слоев материала, фаз агрегатного состояния) в практически используемых моделях микроуровня не может быть большим ввиду сложностей вычислительного характера. Резко снизить вычислительные затраты в многокомпонентных средах можно, только применив иной подход к моделированию, основанный на принятии определенных допущений.

Допущение, выражаемое дискретизацией пространства, позволяет перейти к моделям *макроуровня.* Моделями макроуровня, называемыми также *сосредоточенными,* являются системы алгебраических и обыкновенных дифференциальных уравнений, поскольку независимой переменной здесь остается только время *t.* Упрощение описания отдельных компонентов (деталей) позволяет исследовать модели процессов в

устройствах, приборах, механических узлах, число компонентов в которых может доходить до нескольких тысяч.

В тех случаях, когда число компонентов в исследуемой системе превышает некоторый порог, сложность модели системы на макроуровне вновь становится чрезмерной. Поэтому, принимая соответствующие допущения, переходят на *функционально-логический* уровень. На этом уровне используют аппарат передаточных функций для исследования аналоговых (непрерывных) процессов или аппарат математической логики и конечных автоматов, если объектом исследования является дискретный процесс, т.е. процесс с дискретным множеством состояний.

Наконец, для исследования еще более сложных объектов, примерами которых могут служить производственные предприятия и их объединения, вычислительные системы и сети, социальные системы и другие подобные объекты, применяют аппарат теории массового обслуживания, возможно использование и некоторых других подходов, например сетей Петри. Эти модели относятся к *системному* уровню моделирования.

Требования к математическим моделями численным методам в САПР. Основными требованиями к МО являются требования адекватности, точности, экономичности.

Модель всегда лишь приближенно отражает некоторые свойства объекта. *Адекватность* имеет место, если модель отражает заданные свойства объекта с приемлемой точностью. Под точностью понимают степень соответствия оценок одноименных свойств объекта и модели.

Экономичность (вычислительная эффективность) определяется затратами ресурсов, требуемых для реализации модели. Поскольку в САПР используются математические модели, далее речь пойдет о характеристиках именно математических моделей, и экономичность будет характеризоваться затратами машинных времени и памяти.

Адекватность оценивается перечнем отражаемых свойств и областями адекватности. *Область адекватности* - область в пространстве параметров, в пределах которой погрешности модели остаются в допустимых пределах. Например, область адекватности линеаризованной модели поверхности детали определяется системой неравенств

$$\max |\varepsilon_{ij}| \le \varepsilon_{доп},$$

где ε_{ij} и $\varepsilon_{доп}$ - допущенная и предельно допустимая относительные погрешности моделирования поверхности, максимум берется по всем координатам и контролируемым точкам;

$$\varepsilon_{ij} = (x_{ij\,ист} - x_{ij\,мод})/x_{ij\,ист},$$ $x_{ij\,ист}$ и $x_{ij\,мод}$

- *i-я* координата7-й точки поверхности в объекте и модели соответственно.

Отметим, что в большинстве случаев области адекватности строятся в пространстве внешних переменных. Так, область адекватности модели

электронного радиоэлемента обычно выражает допустимые для применения модели диапазоны изменения моделируемых температур, внешних напряжений, частот.

Аналогичные требования по точности и экономичности фигурируют при выборе численных методов решения уравнений модели.

Место процедур формирования моделей в маршрутах проектирования. Вычислительный процесс при анализе состоит из этапов формирования модели и ее исследования (решения). В свою очередь, формирование модели включает две процедуры: во-первых, разработку моделей отдельных компонентов, во-вторых, формирование модели системы из моделей компонентов.

Первая из этих процедур выполняется предварительно по отношению к типовым компонентам вне маршрута проектирования конкретных объектов. Как правило, модели компонентов разрабатываются специалистами в прикладных областях, причем знающими требования к моделям и формам их представления в САПР. Обычно в помощь разработчику моделей в САПР предлагаются методики и вспомогательные средства, например, в виде программ анализа для экспериментальной отработки моделей. Созданные модели включаются в библиотеки моделей прикладных программ анализа.

На маршруте проектирования каждого нового объекта выполняется вторая процедура (рис.9.1) - формирование модели системы с использованием библиотечных моделей компонентов. Как правило, эта процедура выполняется автоматически по алгоритмам, включенным в заранее разработанные программы анализа. Примеры таких программ имеются в различных приложениях и прежде всего в отраслях общего машиностроения и радиоэлектроники.

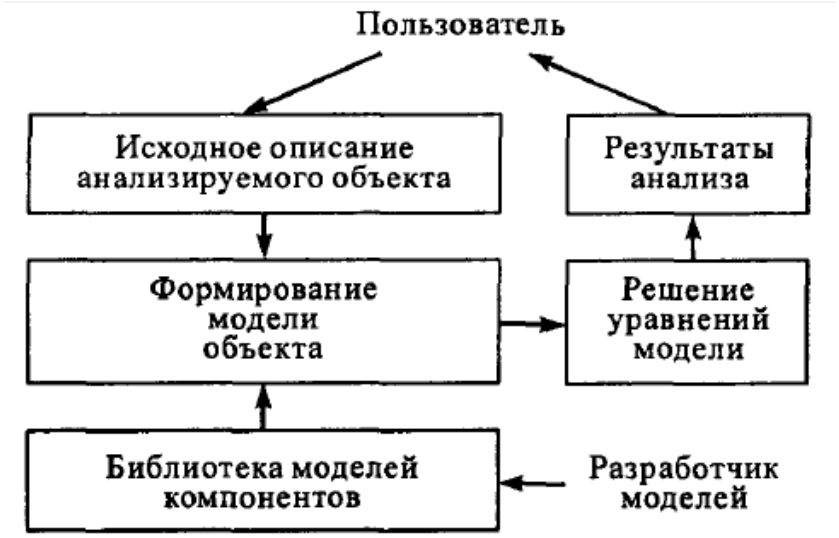


Рис. 9.1. Место процедур формирования моделей на маршрутах проектирования

При применении этих программ пользователь описывает исследуемый

объект на входном языке программы анализа не в виде системы уравнений, которая будет получена автоматически, а в виде списка элементов структуры, эквивалентной схемы, эскиза или чертежа конструкции.

9.2. Математические модели в процедурах анализа на макроуровне

Исходные уравнения моделей. Исходное математическое описание процессов в объектах на макроуровне представлено системами обыкновенных дифференциальных и алгебраических уравнений. Аналитические решения таких систем при типичных значениях их порядков в практических задачах получить не удается, поэтому в САПР преимущественно используются алгоритмические модели. В этом параграфе изложен обобщенный подход к формированию алгоритмических моделей на макроуровне, справедливый для большинства приложений.

Исходными для формирования математических моделей объектов на макроуровне являются компонентные и топологические уравнения.

Компонентными уравнениями называют уравнения, описывающие свойства элементов (компонентов), другими словами, это уравнения математических моделей элементов (ММЭ).

Топологические уравнения описывают взаимосвязи в составе моделируемой системы.

В совокупности компонентные и топологические уравнения конкретной физической системы представляют собой исходную *математическую модель системы* (*ММС).*

Очевидно, что компонентные и топологические уравнения в системах различной физической природы отражают разные физические свойства, но могут иметь одинаковый формальный вид. Одинаковая форма записи математических соотношений позволяет говорить о формальных аналогиях компонентных и топологических уравнений. Такие аналогии существуют для механических поступательных, механических вращательных, электрических, гидравлических (пневматических), тепловых объектов. Наличие аналогий приводит к практически важному выводу: значительная часть алгоритмов формирования и исследования моделей в САПР оказывается инвариантной и может быть применена к анализу проектируемых объектов в разных предметных областях. Единство математического аппарата формирования ММС особенно удобно при анализе систем, состоящих из физически разнородных подсистем.

В перечисленных выше приложениях компонентные уравнения имеют вид

$$F_K(dV/dt, V, t) = 0, \qquad (9.1)$$

топологические уравнения -

$$F_T(V) = 0, \qquad (9.2)$$

где $V = (v_1, v_2, \ldots, v_n)$ - вектор фазовых переменных; t - время.

Различают фазовые переменные двух типов, их обобщенные наименования - *фазовые переменные* типа потенциала (например, электрическое напряжение) и типа потока (например, электрический ток). Каждое компонентное уравнение характеризует связи между разнотипными фазовыми переменными, относящимися к одному компоненту (например, закон Ома описывает связь между напряжением и током в резисторе), а топологическое уравнение - связи между однотипными фазовыми переменными в разных компонентах.

Модели можно представлять в виде систем уравнений или в графической форме, если между этими формами установлено взаимно однозначное соответствие. В качестве графической формы часто используют эквивалентные схемы.

Примеры компонентных и топологических уравнений

Рассмотрим несколько типов систем.

Электрические системы. В электрических системах фазовыми переменными являются электрические напряжения и токи. Компонентами систем могут быть простые двухполюсные элементы и более сложные двух- и многополюсные компоненты. К простым двухполюсникам относятся следующие элементы: сопротивление, емкость и индуктивность, характеризуемые одноименными параметрами *R, C, L*. В эквивалентных схемах эти элементы обозначают в соответствии с рис. 9.2, *а*.

Компонентные уравнения простых двухполюсников:

для сопротивления $u=iR$ (закон Ома); (9.3)

для емкости $i=Cdu/dt;$ (9.4)

для индуктивности $u=Ldi/dt,$ (9.5)

где u - напряжение (точнее, падение напряжения на двухполюснике); i - ток.

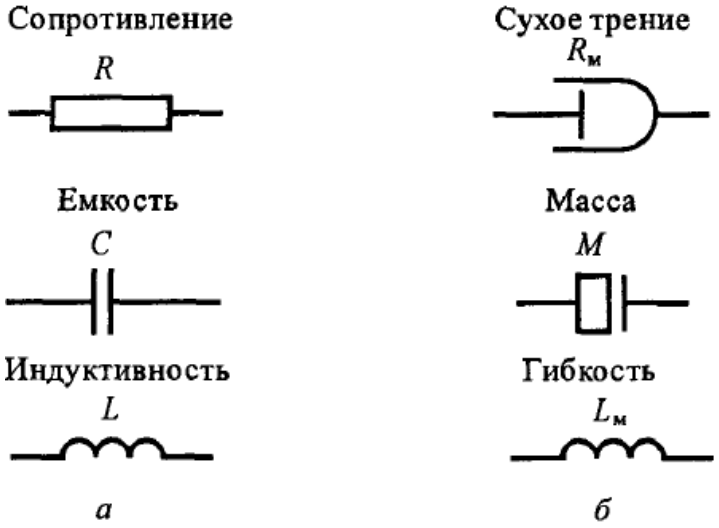


Рис. 9.2. Условные обозначения простых элементов в эквивалентных схемах: *а* - электрических, гидравлических, тепловых; *б* - механических

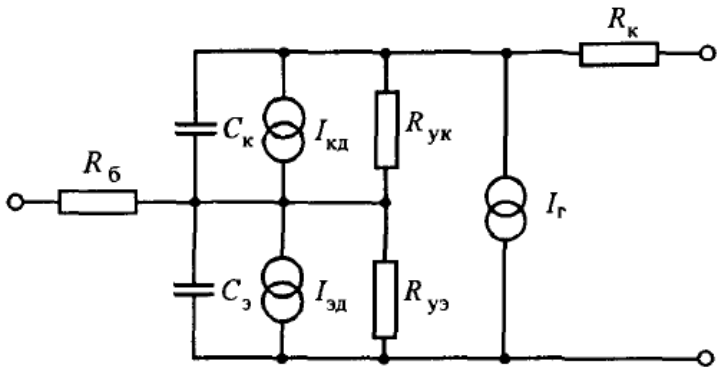


Рис. 9.3. Эквивалентная схема биполярного транзистора

Эти модели лежат в основе моделей других возможных более сложных компонентов. Большая сложность может определяться нелинейностью уравнений (9.3) - (9.5) (т. е. зависимостью *R, C, L* от фазовых переменных), или учетом зависимостей параметров *R, C, L* от температуры, или наличием более двух полюсов. Однако многополюсные компоненты могут быть сведены к совокупности взаимосвязанных простых элементов.

Топологические уравнения выражают законы Кирхгофа для напряжений (ЗНК) и токов (ЗТК). Согласно ЗНК, сумма напряжений на компонентах вдоль любого замкнутого контура в эквивалентной схеме равна нулю, а в соответствии с ЗТК сумма токов в любом замкнутом сечении эквивалентной схемы равна нулю:

$$\sum_{k \in K_p} u_к = 0, \tag{9.6}$$

$$\sum_{j \in J_q} i_j = 0, \tag{9.7}$$

где *Кр* - множество номеров элементов *р-го* контура; J_q - множество номеров элементов, входящих в *q-е* сечение.

Примером математической модели сложного компонента может служить модель транзистора. На рис. 9.3 представлена эквивалентная схема биполярного транзистора, на которой зависимые от напряжений источники тока $i_{зд} = i_{тэ}\exp(u, m\varphi,))$ и $i_{кд} = i_{тк}\exp(u_к/m\varphi,))$ отображают статические вольтамперные характеристики $p-n$ - переходов; $i_{тэ}$ и $i_{тк}$ - тепловые токи переходов; $m\varphi_T$ - температурный потенциал;$u_э$ $и$ $u_К$ - напряжения на эмиттерном и коллекторном переходах; $C_э$ и $C_К$ - емкости переходов; $R_{уэ}$ и $R_{ук}$ - сопротивления утечки переходов; $R_б$ и $R_к$ - объемные сопротивления тел базы и коллектора; $i = Bi_{ЭД} - B_и i_{КД}$ - источник тока, моделирующий усилительные свойства транзистора; B и $B_И$ - прямой*и инверсный

коэффициенты усиления тока базы. Здесь $u_э, u_к, i_{эд}, i_{кд}, i_г$ фазовые переменные, а остальные величины - параметры модели транзистора.

Механические системы. Фазовыми переменными в механических поступательных системах являются силы и скорости. Используют одну из двух возможных электромеханических аналогий. В дальнейшем будем использовать ту из них, в которой скорость относят к фазовым переменным типа потенциала, а силу считают фазовой переменной типа потока. Учитывая формальный характер подобных аналогий, в равной мере можно применять и противоположную терминологию.

Компонентное уравнение, характеризующее инерционные свойства тел, в силу второго закона Ньютона имеет вид

F=Mdu/dt, (9.8)

где *F* - сила; *M* - масса; *u* - поступательная скорость.

Упругие свойства тел описываются компонентным уравнением, которое можно получить из уравнения закона Гука. В одномерном случае (если рассматриваются продольные деформации упругого стержня)

$G = E\varepsilon$ (9.9)

где G - механическое напряжение; E - модуль упругости;$\varepsilon = \Delta l/l$ - относительная деформация; Δl - изменение длины l упругого тела под воздействием *G*. Учитывая, что $G = F/S$ где *F* - сила, *S* - площадь поперечного сечения тела, и дифференцируя (9.9), имеем

$$dF/dt = (SE/l)\, d(\Delta l)/dt$$

или

$$dF/dt = gu, \quad (9.10)$$

где $g = SE/l$ - жесткость (величину, обратную жесткости, называют гибкостью L_M); $u = d(\Delta l)/dt$ - скорость.

Диссипативные свойства в механических системах твердых тел выражаются соотношениями, характеризующими связь между силой трения и скоростью взаимного перемещения трущихся тел, причем в этих соотношениях производные сил или скоростей не фигурируют, как и в случае описания с помощью закона Ома диссипативных свойств в электрических системах.

Топологические уравнения характеризуют, во-первых, закон равновесия сил: сумма сил, приложенных к телу, включая силу инерции, равна нулю (принцип Даламбера); во-вторых, закон скоростей, согласно которому сумма относительной, переносной и абсолютной скоростей равна нулю.

В механических вращательных системах справедливы компонентные и топологические уравнения поступательных систем с заменой

поступательных скоростей на угловые, сил - на вращательные моменты, масс - на моменты инерции, жесткостей - на вращательные жесткости.

Условные обозначения простых элементов механической системы показаны на рис. 9.2, *б*.

Нетрудно заметить наличие аналогий между электрической и механической системами. Так, токам и напряжениям в первой из них соответствуют силы (либо моменты) и скорости механической системы, компонентным уравнениям (9.4) и (9.5) и фигурирующим в них параметрам С и L - уравнения (9.8) и (9.10) и параметры M и $L_м$, очевидна аналогия и между топологическими уравнениями. Далее параметры C и M будем называть емкостными (емкостного типа), параметры L и $L_м$ - индуктивными (индуктивного типа), а параметры R и $R_{Тр}=\partial u/\partial P$ - резистивными (резистивного типа).

Имеется и существенное отличие в моделировании электрических и механических систем: первые из них одномерны (на макроуровне), а процессы во вторых часто приходится рассматривать в двумерном *(2D)* или трехмерном *(3D)* пространстве. Следовательно, при моделировании механических систем в общем случае в пространстве *3D* нужно использовать векторное представление фазовых переменных, каждая из которых имеет шесть составляющих, соответствующих шести степеням свободы.

Однако отмеченные выше аналогии остаются справедливыми, если их относить к проекциям сил и скоростей на каждую пространственную ось, а при графическом представлении моделей использовать шесть эквивалентных схем - три для поступательных составляющих и три для вращательных.

9.3. Представление топологических уравнений

Известен ряд методов формирования ММС на макроуровне. Получаемые с их помощью модели различаются ориентацией на те или иные численные методы решения и набором *базисных переменных,* т. е. фазовых переменных, остающихся в уравнениях итоговой ММС. Общей для всех методов является исходная совокупность топологических и компонентных уравнений (9.1) и (9.2).

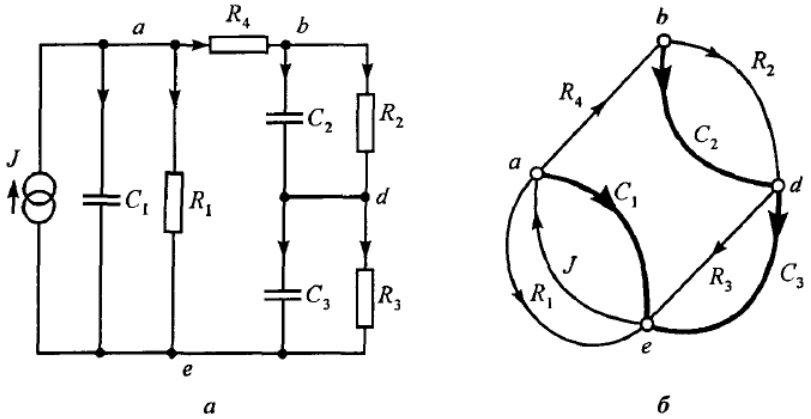


Рис. 9.4. Эквивалентная схема (а) и ее граф (б)

При записи топологических уравнений удобно использовать промежуточную графическую форму - представление модели в виде эквивалентной схемы, состоящей из двухполюсных элементов. Общность подхода при этом сохраняется, так как любой многополюсный компонент можно заменить подсхемой из двухполюсников. В свою очередь, эквивалентную схему можно рассматривать как направленный граф, дуги которого соответствуют ветвям схемы. Направления потоков в ветвях выбираются произвольно (если реальное направление при моделировании окажется противоположным, то это приведет лишь к отрицательным численным значениям потока).

Пример некоторой простой эквивалентной схемы и соответствующего ей графа приведен на рис.9.4. Для конкретности и простоты изложения на рисунке использованы условные обозначения, характерные для электрических эквивалентных схем, по той же причине далее в этом параграфе часто применяется электрическая терминология. Очевидно, что поясненные выше аналогии позволяют при необходимости легко перейти к обозначениям и терминам, привычным для механиков.

Для получения топологических уравнений все ветви эквивалентной схемы разделяют на подмножества *хорд* и *ветвей дерева.* Имеется в виду покрывающее (фундаментальное) дерево, т. е. подмножество из $\beta - 1$ дуг, не образующее ни одного замкнутого контура, где β - число вершин графа (узлов эквивалентной схемы). На рис. *9.6, а,* утолщенными линиями выделено одно из возможных покрывающих деревьев.

Выбор дерева однозначно определяет векторы напряжений $U_х$ и токов $I_х$ хорд, напряжений $U_{вд}$ и токов $I_{вд}$ ветвей дерева и приводит к записи топологических уравнений в виде

$$\mathbf{U_х} + \mathbf{M}\mathbf{U_{вд}} = 0; \tag{9.11}$$

$$\mathbf{I_{вд}} - \mathbf{M}^{\mathrm{T}}\mathbf{I_х} = 0, \tag{9.12}$$

где M - матрица контуров и сечений; M^T - транспонированная М-матрица.

В M-матрице число строк соответствует числу хорд, число столбцов равно числу ветвей дерева. M-матрица формируется следующим образом. Поочередно к дереву подключаются хорды. Если при подключении к дереву p- й хорды q-я ветвь входит в образовавшийся контур, то элемент матрицы M_{pq} = +1 при совпадении направлений ветви и подключенной хорды, $M_м$=-1 при несовпадении направлений. В противном случае M_m= 0.

Для схемы на рис. 9.4 М-матрица представлена в виде табл. 9.1.

Особенности эквивалентных схем механических объектов. Для каждой степени свободы строят свою эквивалентную схему. Каждому телу с учитываемой массой соответствует узел схемы (вершина графа). Один узел,

называемый базовым, отводится телу, отождествляемому с инерциальной системой отсчета.

Таблица 9.1.

Хорды	Ветви дерева		
	*C*1	*C*2	*C*3
*R*1	–1	0	0
*R*2	0	–1	0
*R*3	0	0	–1
*R*4	–1	+1	+1
J	+1	0	0

Каждый элемент массы изображают ветвью, соединяющей узел, соответствующий массе тела с базовым узлом; каждый элемент упругости - ветвью, соединяющей узлы тел, связанных упругой связью; каждый элемент трения - ветвью, соединяющей узлы трущихся тел. Внешние воздействия моделируются источниками сил и скоростей.

В качестве примера на рис. 9.5, *а* изображена некоторая механическая система - тележка, движущаяся по дороге и состоящая из платформы *A,* колес B1, B2 и рессор Cl, C2. На рис. 9.5, *б* приведена эквивалентная схема для вертикальных составляющих сил и скоростей, на которой телам системы соответствуют одноименные узлы, учитываются массы платформы и колес, упругость рессор, трение между колесами и дорогой; неровности дороги вызывают воздействие на систему, изображенное на рис. 9.5, *б,* источниками силы.

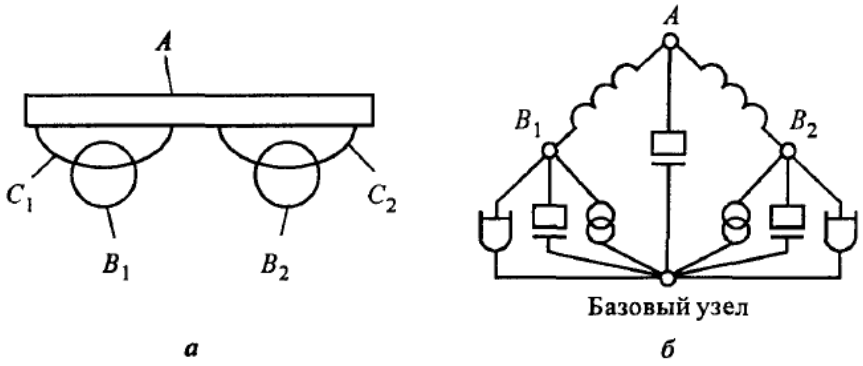


Рис. 9.5. Простая механическая система: *а* - эскизное изображение; *б*- эквивалентная схема

9.4. Характеристика методов формирования ММС

Исходную систему компонентных и топологических уравнений (9.1) и (9.2) можно рассматривать как окончательную ММС, которая и подлежит численному решению. Численное решение этой системы уравнений предполагает *алгебраизацию* дифференциальных уравнений, например, с помощью преобразования Лапласа или формул численного интегрирования. В программах анализа нелинейных объектов на макроуровне, как правило, применяются формулы численного интегрирования, примером которых может служить неявная формула Эйлера

$$d\mathbf{V}/dt\,|_n = (\mathbf{V}_n - \mathbf{V}_{n-1})/h_n,$$

где V - значение переменной V на i-м шаге интегрирования; $h_n=t_n-t_{n-1}$ - шаг интегрирования. Алгебраизация подразумевает предварительную дискретизацию независимой переменной t (вместо непрерывной переменной t получаем конечное множество значений t_n) она заключается в представлении ММС в виде системы уравнений

$$\begin{aligned} \mathbf{F}_к(\mathbf{Z}_n, \mathbf{V}_n, t_n) &= 0; \\ \mathbf{F}_т(\mathbf{V}_n) &= 0; \\ \mathbf{Z}_n &= (\mathbf{V}_n - \mathbf{V}_{n-1})/h_n \end{aligned} \qquad (9.13)$$

с неизвестными V_n и Z_n, где использовано обозначение $Z = dV/dt$. Эту систему алгебраических уравнений, в общем случае нелинейных, необходимо решать на каждом шаге численного интегрирования исходных дифференциальных уравнений.

Однако порядок этой системы довольно высок и примерно равен $2\alpha + \gamma$, где α - число ветвей эквивалентной схемы (каждая ветвь дает две неизвестные величины - фазовые переменные типа потока и типа потенциала, за исключением ветвей внешних источников, у каждой из которых не известна лишь одна фазовая переменная), γ - число элементов в векторе производных. Чтобы снизить порядок системы уравнений и тем самым повысить вычислительную эффективность ММС, желательно выполнить предварительное преобразование модели (в символическом виде) перед ее многошаговым численным решением. Предварительное преобразование сводится к исключению из системы части неизвестных и соответствующего числа уравнений. Оставшиеся неизвестные называют *базисными.* В зависимости от набора базисных неизвестных различают несколько методов формирования ММС.

Согласно *методу переменных состояния (более* полное название метода - метод переменных, характеризующих состояние), вектор базисных переменных W состоит из *переменных состояния.* Этот вектор включает

неизбыточное множество переменных, характеризующих накопленную в системе энергию. Например, такими переменными могут быть скорости тел (кинетическая энергия определяется скоростью, так как равна $Mu^2/2$), емкостные напряжения, индуктивные токи и т. п. Очевидно, что число уравнений не превышает у. Кроме того, итоговая форма ММС оказывается приближенной к явной форме представления системы дифференциальных уравнений, т. е. к форме, в которой вектор *dW/dt* явно выражен через вектор **W,** что упрощает дальнейшее применение явных методов численного интегрирования. Метод реализуется путем особого выбора системы хорд и ветвей дерева при формировании топологических уравнений. Поскольку явные методы численного интегрирования дифференциальных уравнений не нашли широкого применения в программах анализа, то метод переменных состояния также теряет актуальность и его применение оказывается довольно редким.

В классическом варианте *узлового метода* в качестве базисных переменных используются узловые *потенциалы* (т.е. скорости тел относительно инерциальной системы отсчета, абсолютные температуры, перепады давления между моделируемой и внешней средой, электрические потенциалы относительно базового узла). Число узловых потенциалов и соответственно уравнений в ММС оказывается равным $\beta - 1$, где β - число узлов в эквивалентной схеме. Обычно β заметно меньше а, и, следовательно, порядок системы уравнений в ММС снижен более чем в 2 раза по сравнению с порядком исходной системы.

Однако классический вариант узлового метода имеет ограничения на применение, и потому в современных программах анализа наибольшее распространение получил *модифицированный узловой метод.*

Контрольные вопросы

1. Дайте определение области адекватности математической модели.

2. Представьте схему на рис. 9.6 в виде графа, постройте покрывающее дерево, запишите матрицу контуров и сечений **M.**

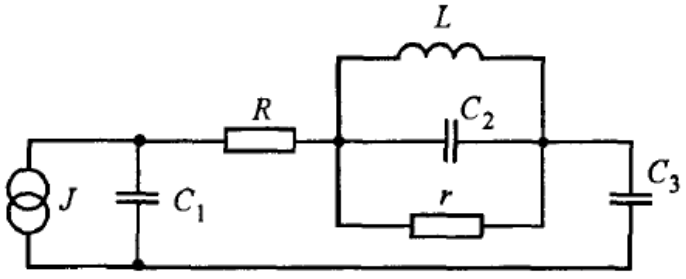


Рис.9.6. Эквивалентная схема.

3. Запишите компонентные и топологические уравнения для эквивалентной схемы на рис. 9.6.

4. Составьте эквивалентную схему для гидромеханической системы (цилиндра с поршнем), представленную на рис.9.7, где *F* - сила, действующая на поршень.

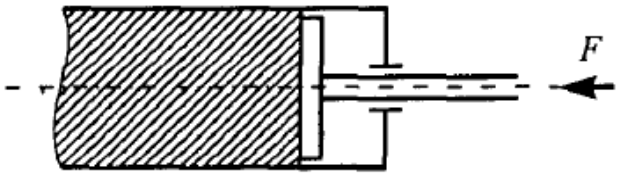


Рис. 9.10. Гидромеханическая система

5. Напишите выражения для проводимостей ветвей схемы (см.рис. 9.6) в случае использования неявного метода Эйлера для интегрирования системы дифференциальных уравнений.

6. Сформулируйте математическую модель по модифицированному методу узловых потенциалов для схемы на рис. 9.6.

7. Что понимают под постоянной времени физической системы?

8. Выполните несколько шагов интегрирования для дифференциального уравнения $dx/dt=10-2x$ явным и неявным методами Эйлера с начальным условием $x_0=0$ и с шагом $h=2$, нарушающим условие. Сделайте заключение об устойчивости или неустойчивости вычислений.

9. Каким образом обеспечивается сходимость итераций при решении СНАУ?

10. На чем основаны алгоритмы автоматического выбора шага интегрирования при решении систем дифференциальных уравнений?

Лекция № 10.
Тема: Методы и алгоритмы анализа на макроуровне.

План

10.1. Выбор методов анализа во временной области.

10.2.Алгоритм численного интегрирования СОДУ.

10.3. Методы решения систем нелинейных алгебраических уравнений.

10.4. Методы решения систем линейных алгебраических уравнений.

10.5. Организация вычислительного процесса в универсальных программах анализа на макроуровне.

Ключевые слова: анализ, модель, обусловленность, устойчивость, метод Эйлера, метод трапеции, метод интегрирования, метод Гаусса, разреженная матрица, интерпретация, компиляция.

10.1. Выбор методов анализа во временной области

Анализ процессов в проектируемых объектах можно проводить во временной и частотной областях. *Анализ во временной области* (динамический анализ) позволяет получить картину переходных процессов, оценить динамические свойства объекта, он является важной процедурой при исследовании как линейных, так и нелинейных систем. *Анализ в частотной*

области более специфичен, его применяют, как правило, к объектам с линеаризуемыми математическими моделями при исследовании колебательных стационарных процессов, анализе устойчивости, расчете искажений информации, представляемой спектральными составляющими сигналов, и т.п.

Методы анализа во временной области, используемые в универсальных программах анализа в САПР, - это численные методы интегрирования систем обыкновенных дифференциальных уравнений (СОДУ):

$$\mathbf{F}(d\mathbf{V}/dt, \mathbf{V}, t) = 0.$$

Другими словами, это методы алгебраизации дифференциальных уравнений. Формулы интегрирования СОДУ могут входить в математическую модель независимо от компонентных уравнений, как это имеет место или быть интегрированными в математические модели компонентов, как это выполнено в узловом методе.

От выбора метода решения СОДУ существенно зависят такие характеристики анализа, как точность и вычислительная эффективность. Эти характеристики определяются прежде всего типом и порядком выбранного метода интегрирования СОДУ.

Применяют два типа методов интегрирования - явные (иначе экстраполяционные, или методы, основанные на формулах интегрирования вперед) и неявные (интерполяционные, основанные на формулах интегрирования назад). Различия между ними удобно показать на примере простейших методов первого порядка - методов Эйлера.

Формула *явного метода Эйлера* представляет собой следующую формулу замены производных в точке t_n:

$$d\mathbf{V}/dt\,|_n = (\mathbf{V}_{n+1} - \mathbf{V}_n)/h_n.$$

Здесь индекс равен номеру шага интегрирования; $h_n=t_{n+1}-t_n$ - размер шага интегрирования (обычно h_n называют просто шагом интегрирования).

В формуле *неявного метода Эйлера* использовано дифференцирование назад:

$$d\mathbf{V}/dt\,|_n = (\mathbf{V}_n - \mathbf{V}_{n-1})/h_n,$$

где $h_n=t_{n+1}-t_n$.

Выполним сравнительный анализ явных и неявных методов на примере модельной задачи:

$$d\mathbf{V}/dt = \mathbf{A}\mathbf{V} \qquad (10.1)$$

при ненулевых начальных условиях $\mathbf{V}_0 \neq 0$ и при использовании методов Эйлера с постоянным шагом h. Здесь A - постоянная матрица; V - вектор фазовых переменных.

При алгебраизации явным методом имеем

$$(\mathbf{V}_{n+1} - \mathbf{V}_n)/h = \mathbf{A}\,\mathbf{V}_n$$

или

$$\mathbf{V}_{n+1} = (\mathbf{E} + h\,\mathbf{A})\,\mathbf{V}_n,$$

где E - единичная матрица. Вектор V_{n+1} можно выразить через вектор начальных условий V_0:

$$\mathbf{V}_{n+1} = (\mathbf{E} + h\mathbf{A})^n\,\mathbf{V}_0 \qquad (10.2)$$

Обозначим

$$\mathbf{B} = \mathbf{E} + h\mathbf{A} \qquad (10.3)$$

и применим преобразование подобия для матрицы B:

$$\mathbf{B} = \mathbf{T}^{-1}\mathbf{diag}\{\lambda_{\mathbf{B}_J}\}\mathbf{T}.$$

Здесь T - преобразующая матрица; **diag{A,B** } - диагональная матрица с собственными значениямиλ_{BJ} матрицы B на диагонали. Нетрудно видеть, что

$$\mathbf{B}^n = \mathbf{T}^{-1}\mathbf{diag}\{\lambda^n_{\mathbf{B}_J}\}\mathbf{T}.$$

Из линейной алгебры известно, что собственные значения матриц, связанных арифметическими операциями, оказываются связанными такими же преобразованиями. Поэтому из (10.3) следует:

$$\lambda_{\mathbf{B}_J} = 1 + h\lambda_{\mathbf{A}_J}.$$

Точное решение модельной задачи (10.1) $V(t) \to 0$ при $t \to \infty$, следовательно, условием устойчивости процесса численного решения можно считать

$$\mathbf{V}_{n+1} \to 0 \text{ при } n \to \infty,$$

откуда последовательно получаем

$$(\mathbf{E} + h\mathbf{A})^n\,\mathbf{V}_0 \to 0,$$

так как $\mathbf{V}_0 \neq 0$, то $(\mathbf{E} + h\mathbf{A})^n \to 0$, поскольку $\mathbf{T} \neq 0$, то $\lambda^n_{\mathbf{B}_J} \to 0$ и условие устойчивости

$$-1 < |1 + h\lambda_{\mathbf{A}_J}| < 1. \qquad (10.4)$$

Известно, что для физически устойчивых систем собственные значения матрицы коэффициентов в ММС оказываются отрицательными. Если к тому же все λ_{AJ} действительные величины (характер процессов в ММС с моделью (10.1) апериодический), то естественно определить постоянные времени физической системы как

$$\tau_j = -1/\lambda_{\mathbf{A}_J},$$

и условие (10.4) конкретизируется следующим образом

$$-1 < |1 - h/\tau_j| < 1 \text{ или } 0 < h < 2\tau_{\min}, \qquad (10.5)$$

где т - минимальная *постоянная времени.* Если использовать явные методы более высокого порядка, то может увеличиться коэффициент перед τ_{min} в (10.5) но это принципиально не меняет оценки явных методов.

Если нарушено условие (10.5), то происходит потеря устойчивости вычислений, а это означает, что в решении задачи возникают ложные колебания с увеличивающейся от шага к шагу амплитудой и быстрым аварийным остановом ЭВМ вследствие переполнения разрядной сетки. Конечно, ни о какой адекватности решения говорить не приходится.

Для соблюдения (10.5) применяют те или иные алгоритмы автоматического выбора шага. Отметим, что в сложной модели расчет τ_{min} для непосредственного выбора шага по (10.5) слишком трудоемок, кроме того, однократный расчет τ_{min} мало эффективен, так как в нелинейных моделях τ_{min} может изменяться от шага к шагу.

Условие (10.5) накладывает жесткие ограничения на шаг интегрирования. В результате вычислительная эффективность явных методов резко падает с ухудшением *обусловленности ММС.* Действительно, длительность $T_{\text{ИНТ}}$ моделируемого процесса должна быть соизмеримой с временем успокоения системы после возбуждающего воздействия, т. е. соизмерима с максимальной постоянной времени τ_{max}. Требуемое число шагов интегрирования равно

$$\text{Ш} = T_{\text{инт}}/h \sim \tau_{\max}/\tau_{\min}.$$

Отношение $\text{Ч} = \tau_{\max}/\tau_{\min}$ называют *разбросом постоянных времени* или *числом обусловленности.* Чем больше это число, тем хуже обусловленность.

Попытки применения явных методов к любым ММС чаще всего приводят к недопустимо низкой вычислительной эффективности, поскольку в реальных моделях $\text{Ч} > 10^5$ - обычная ситуация. Поэтому в настоящее время в универсальных программах анализа явные методы решения СОДУ не применяют.

Аналогичный анализ числовой устойчивости неявных методов дает следующие результаты. Вместо (10.2) имеем

$$\mathbf{V}_n = (\mathbf{E} - h\mathbf{A})^{-n}\,\mathbf{V}_0,$$

и условие числовой устойчивости принимает вид принимает вид при любых $h > 0$. Следовательно, неявный метод Эйлера обладает так называемой *А-устойчивостъю.*

Примечание. Метод интегрирования СОДУ называют *А* - устойчивым, если погрешность интегрирования остается ограниченной при любом шаге $h > 0$.

Применение *А* - устойчивых методов позволяет существенно уменьшить требуемые числа шагов Ш. В этих методах шаг выбирается автоматически не из условий устойчивости, а только из соображений точности решения.

Выбор порядка метода решения СОДУ довольно прост; во-первых, более высокий порядок обеспечивает более высокую точность, во-вторых,

среди неявных разностных методов кроме метода Эйлера А - устойчивы также методы второго порядка и среди них - метод трапеций. Поэтому преобладающее распространение в программах анализа получили методы второго порядка - модификации метода трапеций.

10.2.Алгоритм численного интегрирования СОДУ

Одна из удачных реализаций неявного метода второго порядка, которую можно считать модификацией *метода трапеций,* основана на комбинированном использовании явной и неявной формул Эйлера. Рассмотрим вопрос, почему такое комбинирование снижает погрешность и приводит к повышению порядка метода.

Предварительно отметим, что в методах *р-го* порядка локальная погрешность, т.е. погрешность, допущенная на одном *n-м* шаге интегрирования, оценивается старшим из отбрасываемых членов в разложении решения *V*(*t)* в ряд Тейлора, где *c* - постоянный коэффициент, зависящий от метода; $|\mathbf{V}^{(p+1)}(\tau)|$ - норма вектора *(p*+1) - *x* производных *V*(*t),* которая оценивается с помощью конечно-разностной аппроксимации;τ - значение времени *t* внутри шага.

Если *n*-й шаг интегрирования в комбинированном методе был неявным, т. е. выполненным по неявной формуле, то следующий шаг с тем же значением *h* должен быть явным. Используя разложение решения *V(t)* в ряд Тейлора в окрестностях точки t_{n+1}, получаем для *(n*+1)-го неявного шага

$$\mathbf{V}(t_n) = \mathbf{V}(t_{n+1}) - (d\mathbf{V}/dt)h_н + (d^2\mathbf{V}/dt^2)h_н^2/2! - (d^3\mathbf{V}/dt^3)h_н^3/3! + \ldots \quad (10.6)$$

и для (*n*+2)-го явного шага

$$\mathbf{V}(t_{n+2}) = \mathbf{V}(t_{n+1}) + (d\mathbf{V}/dt)h_я + (d^2\mathbf{V}/dt^2)h_я^2/2! + (d^3\mathbf{V}/dt^3)h_я^3/3! + \ldots, \quad (10.7)$$

где $h_н$ и $h_я$ - величины неявного и явного шагов, а значения производных относятся к моменту t_{n+1r}. Подставляя (10.6) в (10.7), при $h=h_я=h_н$ получаем

$$\mathbf{V}(t_{n+2}) = \mathbf{V}(t_n) + 2(d\mathbf{V}/dt)h + 2(d^3\mathbf{V}/dt^3)h_я^3/3! + \ldots,$$

т. е. погрешности, обусловливаемые квадратичными членами в (10.6) и (10.7), взаимно компенсируются и старшим из отбрасываемых членов становится член с *h3.* Следовательно, изложенное комбинирование неявной и явной формул Эйлера дает метод интегрирования второго порядка.

Неявные методы и, в частности, рассмотренный комбинированный метод целесообразно использовать только при переменной величине шага. Действительно, при заметных скоростях изменения фазовых переменных погрешность остается в допустимых пределах только при малых шагах, в квазистатических режимах шаг может быть во много раз больше.

Алгоритмы автоматического выбора шага основаны на сравнении допущенной и допустимой локальных погрешностей. Например, вводится некоторый диапазон (коридор) погрешностей *d,* в пределах которого шаг сохраняется неизменным. Если же допущенная погрешность превышает

верхнюю границу диапазона, то шаг уменьшается, если же выходит за нижнюю границу, то шаг увеличивается.

10.3. Методы решения систем нелинейных алгебраических уравнений

Вычисления при решении СОДУ состоят из нескольких вложенных один в другой циклических процессов. Внешний цикл - это цикл пошагового численного интегрирования, параметром цикла является номер шага. Если модель анализируемого объекта нелинейна, то на каждом шаге выполняется промежуточный цикл - итерационный цикл решения системы нелинейных алгебраических уравнений (СИЛУ). Параметр цикла - номер итерации. Во внутреннем цикле решается СЛАУ, например, при применении узлового метода формирования ММС такой системой является. Поэтому в математическое обеспечение анализа на макроуровне входят методы решения СНАУ и СЛАУ.

Для решения СНАУ можно применять прямые итерационные методы, такие, как метод простой итерации или метод Зейделя, но в современных программах анализа наибольшее распространение получил метод Ньютона, основанный на линеаризации СНАУ. Собственно, модель получена именно в соответствии с методом Ньютона. Основное преимущество метода Ньютона - высокая скорость сходимости.

Представим СНАУ в виде

$$\mathbf{F}(\mathbf{X}) = 0. \tag{10.8}$$

Разлагая F(X) в ряд Тейлора в окрестностях некоторой точки X_k, получаем

$$\mathbf{F}(\mathbf{X}) = \mathbf{F}(\mathbf{X}_k) + (\partial\mathbf{F}/\partial\mathbf{X})(\mathbf{X} - \mathbf{X}_k) + (\mathbf{X} - \mathbf{X}_k)^{\mathrm{T}}(\partial^2\mathbf{F}/\partial\mathbf{X}^2)(\mathbf{X} - \mathbf{X}_k)/2 + \ldots = 0.$$

Сохраняя только линейные члены, имеем СЛАУ с неизвестным вектором X:

$$\mathbf{Я}_k(\mathbf{X} - \mathbf{X}_k) = -\mathbf{F}(\mathbf{X}_k) \tag{10.9}$$

где $\mathbf{Я}_k = (\partial\mathbf{F}/\partial\mathbf{X})|_k$. Решение системы (10.9) дает очередное приближение к корню системы (10.8), которое удобно обозначить X_{k+1}.

Вычислительный процесс стартует с начального приближения X_0 и в случае сходимости итераций заканчивается, когда погрешность, оцениваемая как станет меньше допустимой погрешности e.

Однако метод Ньютона не всегда приводит к сходящимся итерациям. Условия сходимости метода Ньютона выражаются довольно сложно, но существует легко используемый подход к улучшению сходимости. Это близость начального приближения к искомому корню СНАУ. Использование этого фактора привело к появлению метода решения СНАУ, называемого *продолжением решения по параметру.*

В методе продолжения решения по параметру в ММС выделяется некоторый параметр a, такой, что при $\alpha = 0$ корень $X_{\alpha=0}$ системы (10.8)

известен, а при увеличении α от 0 до его истинного значения составляющие вектора X плавно изменяются от $X_{\alpha=0}$ до истинного значения корня. Тогда задача разбивается на ряд подзадач, последовательно решаемых при меняющихся значениях α, и при достаточно малом шаге $\Delta\alpha$ изменения α условия сходимости выполняются.

В качестве параметра α можно выбрать некоторый внешний параметр, например, при анализе электронных схем им может быть напряжение источника питания. Но на практике при интегрировании СОДУ в качестве а выбирают шаг интегрирования *h.* Очевидно, что при *h=0* корень СНАУ равен значению вектора неизвестных на предыдущем шаге. Регулирование значений *h* возлагается на алгоритм автоматического выбора шага.

В этих условиях очевидна целесообразность представления математических моделей для анализа статических состояний в виде СОДУ, как и для анализа динамических режимов.

10.4. Методы решения систем линейных алгебраических уравнений

В программах анализа в САПР для решения СЛАУ чаще всего применяют метод Гаусса или его разновидности. *Метод Гаусса* - метод последовательного исключения неизвестных из системы уравнений. При исключении *k*-й. неизвестной x_k из системы уравнений

$$\mathbf{AX = B} \tag{10.10}$$

все коэффициенты a_{ij} при $i > k$ и $j > k$ пересчитывают по формуле

$$a_{ij} := a_{ij} - a_{ik}\, a_{kj} / a_{kk}. \tag{10.11}$$

Исключение *n-1* неизвестных, где *n* - порядок системы (10.10), называют прямым ходом, в процессе которого матрица коэффициентов приобретает треугольный вид. При обратном ходе последовательно вычисляют неизвестные, начиная с x_n.

В общем случае число арифметических операций для решения (10.10) по Гауссу пропорционально n^3. Это приводит к значительным затратам машинного времени, поскольку СЛАУ решается многократно в процессе одновариантного анализа, и существенно ограничивает сложность анализируемых объектов. Можно заметно повысить вычислительную эффективность анализа, если использовать характерное практически для всех приложений свойство высокой разреженности матрицы А в модели (10.10).

Матрицу называют *разреженной,* если большинство ее элементов равно нулю. Эффективность обработки разреженных матриц велика потому, что не требуются, во-первых, пересчет по формуле (10.11), если хотя бы один из элементов a_{ik} или a_{kj} оказывается нулевым, во-вторых, затраты памяти для хранения нулевых элементов. Хотя алгоритмы обработки разреженных матриц более сложны, но в результате удается получить затраты машинного времени, близкие к линейным, например, затраты оказываются пропорциональными $n^{1,2}$.

При использовании методов разреженных матриц нужно учитывать зависимость вычислительной эффективности от того, как представлена матрица коэффициентов A, точнее, от того, в каком порядке записаны ее строки и столбцы.

Для пояснения этой зависимости рассмотрим два варианта представления одной и той же СЛАУ. В первом случае система уравнений имеет вид

$$a_{11}x_1 + a_{12}x_2 + a_{13}x_3 + a_{14}x_4 + a_{15}x_5 = b_1;$$
$$a_{21}x_1 + a_{22}x_2 = b_2;$$
$$a_{31}x_1 + a_{33}x_3 = b_3;$$
$$a_{41}x_1 + a_{44}x_4 = b_4;$$
$$a_{51}x_1 + a_{55}x_5 = b_5.$$

При прямом ходе в соответствии с формулой (10.11) все элементы матрицы, которые первоначально были нулевыми, становятся ненулевыми, а матрица оказывается полностью насыщенной. Элементы, становящиеся ненулевыми в процессе гауссовых исключений, называют вторичными ненулями. Вторичные ненули в табл. 10.1 отмечены знаком « ».

Во втором случае меняются местами первое и пятое уравнения. Матрицы коэффициентов имеют вид табл. 10.1 и 10.2, где ненулевые элементы представлены знаком «+». Теперь вторичные ненули не появляются, матрица остается разреженной, высокая вычислительная эффективность сохраняется.

Таким образом, методы разреженных матриц должны включать в себя способы оптимального упорядочения строк и столбцов матриц. Используют несколько критериев оптимальности упорядочения. Простейшим из них является критерий расположения строк в порядке увеличения числа первичных ненулей, более сложные критерии учитывают не только первичные ненули, но и появляющиеся вторичные ненули.

Методом разреженных матриц называют метод решения СЛАУ на основе метода Гаусса с учетом разреженности (первичной и вторичной) матрицы коэффициентов.

Таблица 10.3

+	+	+	+	+
+	+	.	.	.
+	.	+	.	.
+	.	.	+	.
+	.	.	.	+

Таблица 10.4

+				+
	+			+
		+		+
			+	+
+	+	+	+	+

Метод разреженных матриц можно реализовать путем интерпретации и компиляции. В обоих случаях создаются массивы ненулевых коэффициентов

матрицы (с учетом вторичных ненулей) и массивы координат этих ненулевых элементов.

При этом выигрыш в затратах памяти довольно значителен. Так, при матрице умеренного размера (200x200) без учета разреженности потребуется 320 Кбайт. Если же взять характерное значение 9 для среднего числа ненулей в одной строке, то для коэффициентов и указателей координат потребуется не более 28 Кбайт.

В случае *интерпретации* моделирующая программа для каждой операции в соответствии с (10.11) при $a_{ik} \neq 0$ и $a_{kj} \neq 0$ находит, используя указатели, нужные коэффициенты и выполняет арифметические операции по (3.33). Поскольку СЛАУ в процессе анализа решается многократно, то и операции поиска нужных коэффициентов также повторяются многократно, на что, естественно, тратится машинное время.

Способ *компиляции* более экономичен по затратам времени, но уступает способу интерпретации по затратам памяти. При компиляции поиск нужных для (10.11) коэффициентов выполняется однократно перед численным решением задачи. Вместо непосредственного выполнения арифметических операций для каждой из них компилируется команда с найденными адресами ненулевых коэффициентов. Такие команды образуют рабочую программу решения СЛАУ, которая и будет решаться многократно. Очевидно, что теперь в рабочей программе будет выполняться минимально необходимое число арифметических операций [4,5].

10.5. Организация вычислительного процесса в универсальных программах анализа на макроуровне

Граф-схема вычислительного процесса при анализе во временной области на макроуровне представлена на рис.10.1. Алгоритм отражает решение системы алгебро-дифференциальных уравнений.

На каждом шаге численного интегрирования решается система нелинейных алгебраических уравнений

$$\varphi(d\,\mathbf{V}/dt, \mathbf{V}, t) = 0.$$

методом Ньютона. На каждой итерации выполняется решение системы линейных алгебраических уравнений

$$\mathbf{F(X)} = 0\,.$$

Другие используемые на рис.10.9. обозначения: $V_0(t_0)$ - начальные условия; h и $h_{\text{нач}}$ - шаг интегрирования и его начальное значение; $U_{ВН}(t)$ - вектор внешних воздействий; N и $N_Д$ - число ньютоновских итераций и его максимально допустимое значение; ε - предельно допустимая погрешность решения СНАУ; δ - погрешность, допущенная на одном шаге интегрирования; $m1$ - максимально допустимое значение погрешности интегрирования на одном шаге; $m2$ - нижняя граница коридора рациональных погрешностей интегрирования.

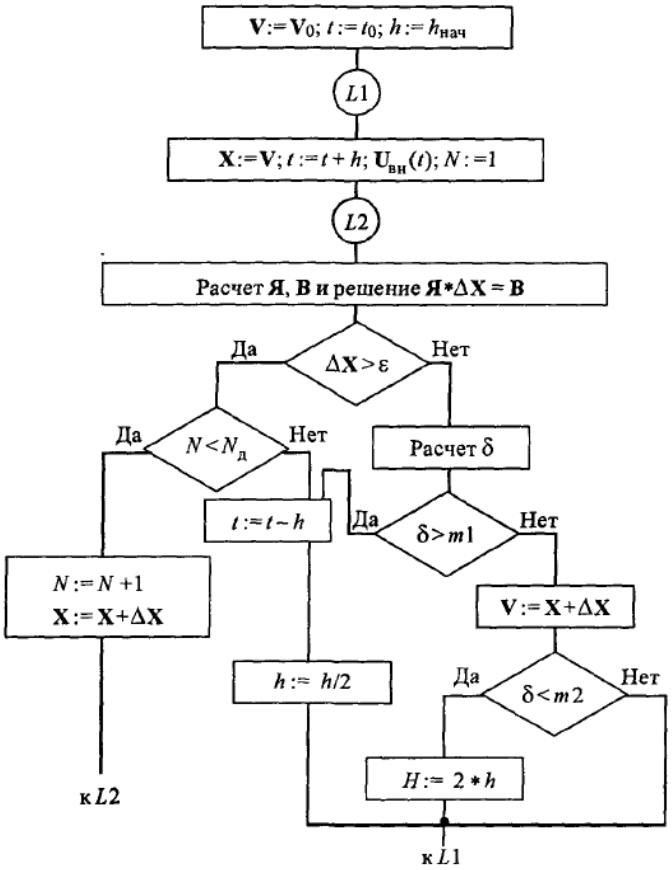


Рис.10.1. Граф-схема вычислительного процесса анализа на макроуровне

Из рисунка ясно, что при $N \geq N_Д$ фиксируется несходимость ньютоновских итераций и после дробления шага происходит возврат к интегрированию при тех же начальных для данного шага условиях. При сходимости рассчитывается δ и в зависимости от того, выходит погрешность за пределы диапазона *[m2, m1]* или нет, шаг изменяется либо сохраняет свое прежнее значение.

Параметры $N_Д$,*m1*,*m2*,ε,$h_{знач}$ задаются по умолчанию и могут настраиваться пользователем.

Матрицу Якоби ***Я*** и вектор правых частей ***B*** необходимо рассчитывать по программе, составляемой для каждого нового исследуемого объекта. Составление программы выполняет компилятор, входящий в состав программного комплекса анализа. Общая структура такого комплекса представлена на рис. 10.2.

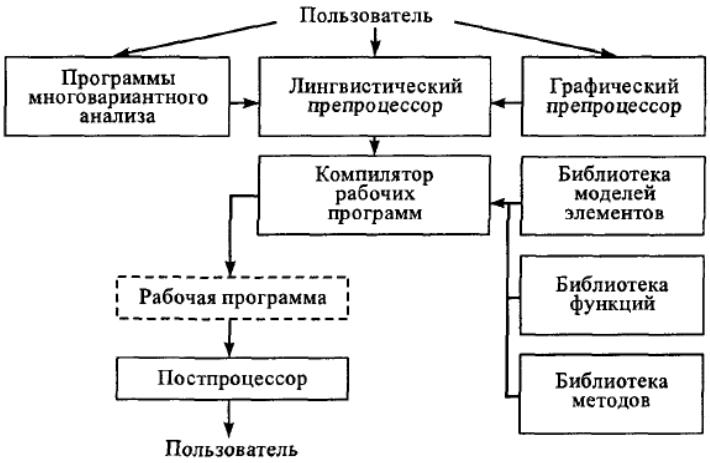


Рис.10.2. Структура программного комплекса анализа на макроуровне.

Исходные данные об объекте можно задавать в графическом виде (в виде эквивалентной схемы) или на входном языке программы анализа. Запись на таком языке обычно представляет собой список компонентов анализируемого объекта с указанием их взаимосвязей. Вводимые данные преобразуются во внутреннее представление с помощью графического и лингвистического препроцессоров, в которых предусмотрена также диагностика нарушений формальных языковых правил. Графическое представление более удобно, особенно для малоопытных пользователей. Задав описание объекта, пользователь может приступить к многовариантному анализу либо по одной из программ такого анализа, либо в интерактивном режиме, изменяя условия моделирования между вариантами с помощью лингвистического препроцессора.

Наиболее сложная часть комплекса - компилятор рабочих программ, именно в нем создаются программы расчета матрицы Якоби ***Я*** и вектора правых частей В, фигурирующих в вычислительном процессе (см.рис.10.1). Собственно рабочая программа (см.рис.10.2)- это и есть программа процесса, показанного на рис.10.2. Для каждого нового моделируемого объекта составляется своя рабочая программа. При компиляции используются заранее разработанные математические модели типовых компонентов, известные функции для отображения входных воздействий, алгоритмы расчета выходных параметров из соответствующих библиотек.

Постпроцессор представляет результаты анализа в табличной и графической формах, это могут быть зависимости фазовых переменных от времени, значения выходных параметров-функционалов и т. п.

Контрольные вопросы

1. Что понимают под постоянной времени физической системы?

2. Выполните несколько шагов интегрирования для дифференциального уравнения dx/dt=10-$2x$ явным и неявным методами Эйлера с начальным условием x_0=0 и с шагом h-2, нарушающим условие

(10.5). Сделайте заключение об устойчивости или неустойчивости вычислений.

3. Каким образом обеспечивается сходимость итераций при решении СНАУ?

4. На чем основаны алгоритмы автоматического выбора шага интегрирования при решении систем дифференциальных уравнений?

5. Что такое «вторичные ненулевые элементы» в методах разреженных матриц?

Лекция № 11.
Тема: Математическое обеспечение анализа на функционально – логическом уровне.

План

11.1. Моделирование и анализ аналоговых устройств.

11.2. Математические модели дискретных устройств

11.3. Методы логического моделирования

Ключевые слова: функционально-логический уровень, алгебраизация, инерционность, макроуровень, дизъюнктор, конъюнктор, инвертор, триггер, синхронный модель, асинхронный модель, ранжирование.

11.1. Моделирование и анализ аналоговых устройств

На функционально-логическом уровне исследуют устройства, в качестве элементов которых принимают достаточно сложные узлы и блоки, считавшиеся системами на макроуровне. Поэтому необходимо упростить представление моделей этих узлов и блоков по сравнению с их представлением на макроуровне. Другими словами, вместо полных моделей узлов и блоков нужно использовать их макромодели.

Вместо двух типов фазовых переменных в моделях функционально-логического уровня фигурируют переменные одного типа, называемые сигналами. Физический смысл сигнала, т. е. его отнесение к фазовым переменным типа потока или типа потенциала, конкретизируют в каждом случае исходя из особенностей задачи [4,5].

Основой моделирования аналоговых устройств на функционально-логическом уровне является использование аппарата передаточных функций. При этом модель каждого элемента представляют в виде уравнения вход-выход, т. е. в виде

$$V_{вых} = f(V_{вх}) \qquad (11.1)$$

где $V_{вых}$ и $V_{вх}$ - сигналы на выходе и входе узла соответственно. Если узел имеет более чем один вход и один выход, то в (11.1) скаляры $V_{вых}$ и $V_{вх}$ становятся векторами.

Однако известно, что представление модели в виде (11.1) возможно, только если узел является безынерционным, т.е. в полной модели узла не фигурируют производные. Следовательно, для получения (11.1) в общем случае требуется предварительная алгебраизация полной модели. Такую алгебраизацию выполняют с помощью интегральных преобразований, например с помощью преобразования Лапласа, переходя из временной области в пространство комплексной переменной p. Тогда в моделях типа (11.1) имеют место не оригиналы, а изображения сигналов $V_{вых}(p)$ и $V_{вх}(p)$, сами же модели реальных блоков стараются по возможности максимально упростить и представить моделями типовых блоков (звеньев) из числа заранее разработанных библиотечных моделей. Обычно модели звеньев имеют вид

$$V_{вых}(p) = h(p)V_{вх}(p),$$

где $h\{p)$ - передаточная функция звена.

В случае применения преобразования Лапласа появляются ограничения на использование нелинейных моделей, а именно в моделях не должно быть нелинейных инерционных элементов.

Другое упрощающее допущение при моделировании на функционально-логическом уровне - неучет влияния нагрузки на характеристики блоков. Действительно, подключение к выходу блока некоторого другого узла не влияет на модель блока (11.1).

Собственно получение ММС из ММЭ оказывается вследствие принятых допущений значительно проще, чем на макроуровне: ММС есть совокупность ММЭ, в которых отождествлены сигналы на соединенных входах и выходах элементов. Эта ММС представляет собой систему алгебраических уравнений.

Получение ММС проиллюстрируем простым примером (рис. 11.1), где показана система из трех блоков с передаточными функциями $h_1(p)$, $h_2(p)$ и $h_3(p)$. ММС имеет вид

$$V_2 = h_1(p)V_1;$$
$$V_{вых}(p) = h_2(p)V_2;$$
$$V_1 = V_{вх}(p) + h_3(p)V_{вых}(p)$$

или

$$V_{вых}(p) = H(p)V_{вх}(p),$$

где $H(p) = h_1(p)h_2(p)/(1 - h_1(p)h_2(p)h_3(p))$.

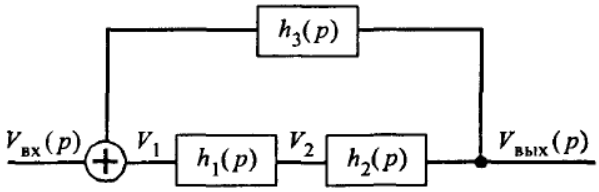


Рис. 11.1. Пример схемы из трех блоков

Таким образом, анализ сводится к следующим операциям:

1) заданную схему устройства представляют совокупностью звеньев, и если схема не полностью покрывается типовыми звеньями, то разрабатывают оригинальные модели;

2) формируют ММС из моделей звеньев;

3) применяют прямое преобразование Лапласа к входным сигналам;

4) решают систему уравнений ММС и находят изображения выходных сигналов;

5) с помощью обратного преобразования Лапласа возвращаются во временную область из области комплексной переменной p.

11.2. Математические модели дискретных устройств

Анализ дискретных устройств на функционально-логическом уровне требуется прежде всего при проектировании устройств вычислительной техники и цифровой автоматики. Здесь дополнительно к допущениям, принимаемым при анализе аналоговых устройств, используют дискретизацию сигналов, причем базовым является двузначное представление сигналов. Удобно этими двумя возможными значениями сигналов считать «истину» (иначе 1) и «ложь» (иначе 0), а сами сигналы рассматривать как булевы величины. Тогда для моделирования можно использовать аппарат математической логики. Находят применение также трех- и более значные модели. Смысл значений сигналов в многозначном моделировании и причины его применения будут пояснены далее на некоторых примерах.

Элементами цифровых устройств на функционально-логическом уровне служат элементы, выполняющие логические функции и возможно функции хранения информации. Простейшими элементами являются дизъюнктор, конъюнктор, инвертор, реализующие соответственно операции дизъюнкции (ИЛИ) $y = a$ ***or*** b, конъюнкции (И) $y = a$ ***and*** b, отрицания (НЕ) $y=$***not*** a, где y - выходной сигнал, a и b - входные сигналы. Число входов

может быть и более двух. Условные схемные обозначения простых логических элементов показаны на рис.11.2.

Математические модели устройств представляют собой систему математических моделей элементов, входящих в устройство, при отождествлении сигналов, относящихся к одному и тому же соединению элементов.

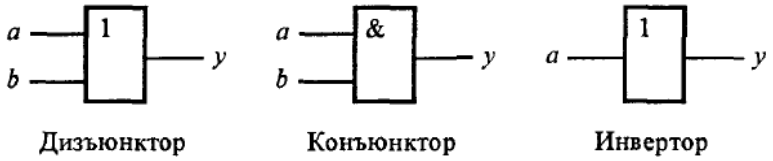


Рис.11.2. Условные обозначения логических элементов на схемах

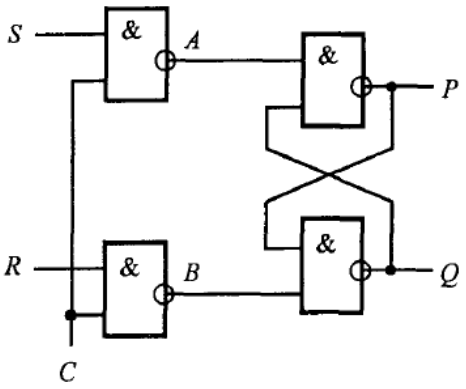


Рис.11.3. Схема *RS*-триггера

Различают синхронные и асинхронные модели.

Синхронная модель представляет собой систему логических уравнений, в ней отсутствует такая переменная, как время. Синхронные модели используют для анализа установившихся состояний.

Примером синхронной модели может служить следующая система уравнений, полученная для логической схемы триггера (рис. 11.3):

$$B = \textbf{not}\ (R\ \textbf{and}\ C);\ Q = \textbf{not}\ (B\ \textbf{and}\ P);\ P = \textbf{not}\ (A\ \textbf{and}\ Q);\ A = \textbf{not}\ (S\ \textbf{and}\ C).$$

Асинхронные модели отражают не только логические функции, но и временные задержки в распространении сигналов. Асинхронная модель логического элемента имеет вид

$$y(t + t_{зд}) = f(\mathbf{X}(t)), \qquad (11.2)$$

где $t_{зд}$ - задержка сигнала в элементе; f - логическая функция. Запись (11.2) означает, что выходной сигналу принимает значение логической функции, соответствующее значениям аргументов $X(t)$, в момент времени $t+t_{зд}$.

Следовательно, асинхронные модели можно использовать для анализа динамических процессов в логических схемах.

Термины «синхронная» и «асинхронная модель» можно объяснить ориентированностью этих моделей на синхронные и асинхронные схемы соответственно. В синхронных схемах передача сигналов между цифровыми блоками происходит только при подаче на специальные синхро входы тактовых (синхронизирующих) импульсов. Частота тактовых импульсов выбирается такой, чтобы к моменту прихода синхроимпульса переходные процессы от предыдущих передач сигналов фактически закончились. Следовательно, в синхронных схемах расчет задержек не актуален, быстродействие устройства определяется заданием тактовой частоты.

Синхронные модели можно использовать не только для выявления принципиальных ошибок в схемной реализации заданных функций. С их помощью можно обнаруживать места в схемах, опасные с точки зрения возникновения в них искажающих помех. Ситуации, связанные с потенциальной опасностью возникновения помех и сбоев, называют *рисками сбоя.*

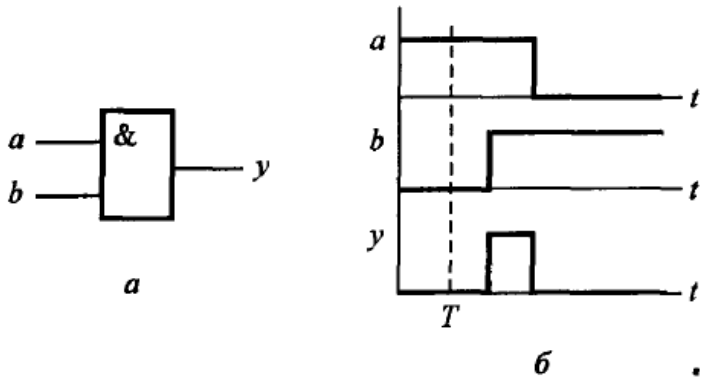


Рис. 11.4. Статический риск сбоя: *а* - схема; *б* - диаграмма сигналов

Различают статический и динамический риски сбоя. Статический риск сбоя иллюстрирует ситуация на рис. 11.4, если на два входа элемента И могут приходить перепады сигналов в противоположных направлениях, как это показано на рис. 11.4,б. Если вместо идеального случая, когда оба перепада приходят в момент времени *T,* перепады вследствие разброса задержек придут неодновременно, причем так, как показано на рис.11.4,б, то на выходе элемента появляется импульс помехи, который может исказить работу всего устройства. Для устранения таких рисков сбоя нужно уметь их выявлять. С этой целью применяют трехзначное синхронное моделирование.

При этом тремя возможными значениями сигналов являются 0,1 и 0, причем значение 0 интерпретируется как неопределенность. Правила

выполнения логических операций И, ИЛИ, НЕ в трехзначном алфавите очевидны из рассмотрения табл.11.1. В ней вторая строка отведена для значений одного аргумента, а первый столбец - для значений второго аргумента, значения функций представлены ниже второй строки и правее первого столбца.

При анализе рисков сбоя на каждом такте вместо однократного решения уравнений модели выполняют двукратное решение, поэтому можно говорить об исходных, промежуточных (после первого решения) и итоговых (после второго решения) значениях переменных. Для входных сигналов допустимы только такие последовательности исходных, промежуточных и итоговых значений: 0–0–0, 1–1–1, $0-\otimes-1$, $1-\otimes-0$. Для других переменных появление последовательности $0-\otimes-0$ или $1-\otimes-1$ означает неопределенность во время переходного процесса, т.е. возможность статического риска сбоя.

Таблица 11.1.

Значение сигнала	Операция								
	И			ИЛИ			НЕ		
	0	⊗	1	0	⊗	1	0	⊗	1
0	0	0	0	0	⊗	1	1	⊗	0
⊗	0	⊗	⊗	⊗	⊗	1	–		
1	0	⊗	1	1	1	1	–		

Таблица 11.2.

Значение	a	b	y
Исходное	1	0	0
Промежуточное	⊗	⊗	⊗
Итоговое	0	1	0

Для простейшей схемы (рис. 11.4, *а*) результаты трехзначного моделирования представлены в табл.11.2.

Динамический риск сбоя иллюстрируют схема и временные диаграммы (рис.11.5). Сбой выражается в появлении вместо одного перепада на выходе, что имеет место при правильном функционировании, нескольких перепадов. Обнаружение динамических рисков сбоя также выполняют с помощью двукратного решения уравнений модели, но при использовании пятизначного алфавита с множеством значений $\{0,1,\otimes,\alpha,\beta\}$, где α интерпретируется как положительный перепад, β - как отрицательный перепад, остальные символы имеют прежний смысл.

В отсутствие сбоев последовательности значений переменных в исходном, промежуточном и итоговом состояниях могут быть такими: 0–0–0, 1–1–1, $0-\alpha-1$, $1-\beta-0$. Последовательности $0-\otimes-1$ или $1-\otimes-0$ указывают на динамический риск сбоя.

Трехзначный алфавит можно использовать и в асинхронных моделях. Пусть в модели $y(t+t_{зд})=f(\mathbf{X}(t))$ в момент времени t_1 входы $X(t_1)$ таковы, что в момент времени $t_1+t_{зд}$ происходит переключение выходного сигнала y. Но если учитывать разброс задержек, то $t_{зд}$ принимает некоторое случайное значение в диапазоне $[t_{зд\ min}, t_{зд\ max}]$ и, следовательно, в модели в интервале времени от $t_1 + t_{зд\ min}$ до $t_1 + t_{зд\ max}$ сигналу должен иметь неопределенное значение $\otimes$. Именно это и достигается с помощью трехзначного асинхронного моделирования.

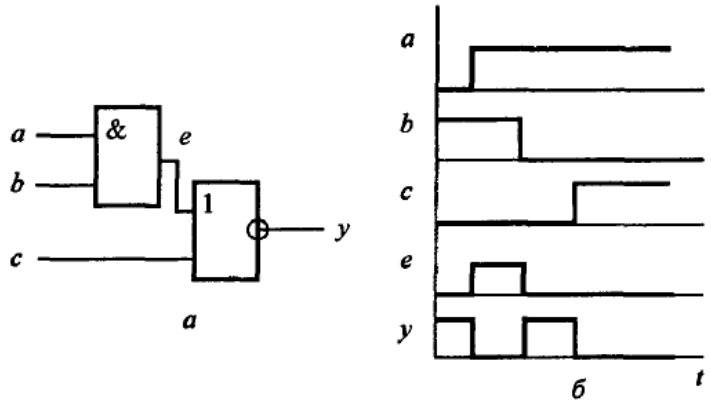


Рис. 11.5. Динамический риск сбоя: *а* - схема; *б* – временные диаграммы

11.3. Методы логического моделирования

В отношении асинхронных моделей возможны два метода моделирования - пошаговый (инкрементный) и событийный.

В *пошаговом методе* время дискретизируется и вычисления по выражениям модели выполняются в дискретные моменты времени t_0, t_1, t_2... и т.д. Шаг дискретизации ограничен сверху значением допустимой погрешности определения задержек и потому оказывается довольно малым, а время анализа - значительным.

Для сокращения времени анализа используют *событийный метод.* В этом методе событием называют изменение любой переменной модели. Событийное моделирование основано на следующем правиле: обращение к модели логического элемента происходит только в том случае, если на входах этого элемента произошло событие. В сложных логических схемах на каждом такте синхронизации обычно происходит переключение всего лишь

2... 3 % логических элементов, и соответственно в событийном методе в несколько раз уменьшаются вычислительные затраты по сравнению с пошаговым моделированием.

Методы анализа синхронных моделей представляют собой методы решения систем логических уравнений. К этим методам относятся *метод простых итераций* и *метод Зейделя,* которые аналогичны одноименным методам решения систем алгебраических уравнений в непрерывной математике.

Применение этих методов к моделированию логических схем удобно проиллюстрировать на примере схемы триггера (см.рис. 11.3). В табл. 3.8 представлены значения переменных модели в исходном состоянии и после каждой итерации в соответствии с методом простых итераций. В исходном состоянии задают начальные (можно произвольные) значения промежуточных и выходных переменных, в данном примере это значения переменных *B, Q,P, A,* соответствующие предыдущему состоянию триггера. Новое состояние триггера должно соответствовать указанным в таблице изменившимся значениям входных сигналов *R,S* и C. Вычисления заканчиваются, если на очередной итерации изменений переменных нет, что и наблюдается в данном примере на четвертой итерации.

Согласно методу простых итераций, в правые части уравнений модели на каждой итерации подставляют значения переменных, полученные на предыдущей итерации. В отличие от этого в методе Зейделя, если у некоторой переменной обновлено значение на текущей итерации, именно его и используют в дальнейших вычислениях уже на текущей итерации. Метод Зейделя позволяет сократить число итераций, но для этого нужно предварительно упорядочить уравнения модели так, чтобы последовательность вычислений соответствовала последовательности прохождения сигналов по схеме. Такое упорядочение выполняют с помощью ранжирования.

Таблица 11.3.

Итерация	*R*	*S*	*C*	*B*	*Q*	*P*	*A*
Предыдущее состояние	0	0	0	1	1	0	1
Исходные значения (итерация 0)	0	1	1	1	1	0	1
Итерация 1	0	1	1	1*	1	0	0*
Итерация 2	0	1	1	1	1	1*	0
Итерация 3	0	1	1	1	0*	1	0
Итерация 4	0	1	1	1	0	1*	0

Ранжирование заключается в присвоении элементам и переменным модели значений рангов в соответствии со следующими правилами: 1) в схеме разрываются все контуры обратной связи, что приводит к появлению дополнительных входов схемы (псевдовходов); 2) все внешние переменные (в том числе на псевдовходах) получают ранг 0; 3) элемент и его выходные переменные получают ранг *к,* если у элемента все входы проранжированы и старший среди рангов входов равен *к* - 1.

Так, если в схеме (см.рис.11.5) разорвать имеющийся контур обратной связи в цепи переменной Q и обозначить переменную на псевдовходе Q_1, то ранги переменных оказываются следующими: R, S, C, Q_1 имеют ранг 0, A и B - ранг 1 ,P- ранг 2 и Q- ранг 3. В соответствии с этим переупорядочивают уравнения в модели триггера:

$$A = \textbf{not}\,(S\ \textbf{and}\ C);\ B = \textbf{not}\,(R\ \textbf{and}\ C);\ P = \textbf{not}\,(A\ \textbf{and}\ Q);\ Q = \textbf{not}\,(B\ \textbf{and}\ P).$$

Теперь уже на первой итерации (по Зейделю) получаем требуемый результат. Если разорвать контур обратной связи в цепи переменной P, то решение в данном примере будет получено после второй итерации, но это все равно заметно быстрее, чем при использовании метода простой итерации.

Для сокращения объема вычислений в синхронном моделировании возможно использование событийного подхода. По-прежнему обращение к модели элемента происходит, только если на его входах произошло событие.

Для триггера (см. рис. 11.5) применение событийности в рамках метода простых итераций приводит к сокращению объема вычислений: вместо 16-кратных обращений к моделям элементов, как это следует из табл. 11.3, происходит лишь пятикратное обращение. В табл. 3.8 звездочками помечены значения переменных, вычисляемые в событийном методе. Так, например, на итерации 0 имеют место изменения переменных S и C, поэтому на следующей итерации обращения происходят только к моделям элементов с выходами A и B.

Контрольные вопросы

1. В чем заключается различие способов интерпретации и компиляции при реализации метода разреженных матриц?

2. Что понимают под областью работоспособности?

3. Как организуется вычислительный процесс в универсальных программах анализа?

4. Найдите координатные функции для одномерной задачи при линейной аппроксимации функции *f(x)* (рис.11.6 на котором показаны конечные элементы длиной L).

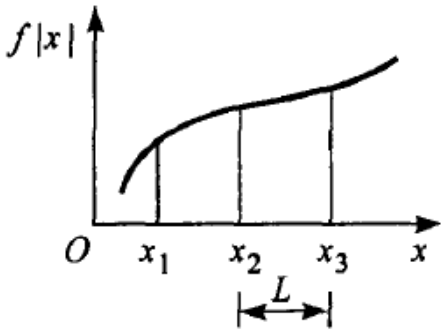


Рис.11.6. Функция для конечно - элементной аппроксимации

5. В чем заключается ранжирования элементов?

Лекция № 12.
Тема: Математическое обеспечение анализа на системном уровне.

План

12.1. Основные сведения из теории массового обслуживания

12.2. Аналитические модели СМО

12.3. Пример аналитической модели

12.4. Имитационное моделирование СМО.

12.5. Событийный метод моделирования

Ключевые слова: транзакт, статический объект, ресурс, динамический объект, приоритет, имитационное моделирование, марковский цепь, вероятности, имитационное моделирование, событийный метод.

12.1. Основные сведения из теории массового обслуживания

Объектами проектирования на системном уровне являются такие сложные системы, как производственные предприятия, транспортные системы, вычислительные системы и сети, автоматизированные системы проектирования и управления и т.п. В этих приложениях анализ процессов функционирования систем связан с исследованием прохождения через систему потока заявок (иначе называемых *требованиями* или *транзактами).* Разработчиков подобных сложных систем интересуют прежде всего такие параметры, как производительность (пропускная способность) проектируемой системы, продолжительность обслуживания (задержки) заявок в системе, эффективность используемого в системе оборудования.

Заявками могут быть заказы на производство изделий, задачи, решаемые в вычислительной системе, клиенты в банках, грузы, поступающие на транспортировку, и др. Очевидно, что параметры заявок, поступающих в систему, являются случайными величинами и при проектировании могут быть известны лишь их законы распределения и числовые характеристики этих распределений. Поэтому анализ функционирования на системном

уровне, как правило, носит статистический характер. В качестве математического аппарата моделирования удобно принять теорию массового обслуживания, а в качестве моделей систем на этом уровне использовать *системы массового обслуживания* О*СМО*).

Типичными выходными параметрами в СМО являются числовые характеристики таких величин, как время обслуживания заявок в системе, длины очередей заявок на входах, время ожидания обслуживания в очередях, загрузка устройств системы, а также вероятность обслуживания в заданные сроки и т. п.

В простейшем случае СМО представляет собой некоторое средство (устройство), называемое *обслуживающим аппаратом* (*ОА*), вместе с очередями заявок на входах. Более сложные СМО состоят из многих взаимосвязанных ОА. Обслуживающие аппараты СМО в совокупности образуют *статические объекты* СМО, иначе называемые *ресурсами.* Например, в вычислительных сетях ресурсы представлены аппаратными и программными средствами.

В СМО кроме статических объектов фигурируют *динамические объекты* - транзакты. Например, в вычислительных сетях динамическими объектами являются решаемые задачи и запросы на информационные услуги.

Состояние СМО характеризуется состояниями составляющих ее объектов. Например, состояния ОА выражаются булевыми величинами, значения которых интерпретируются как *true* (занято) и *false* (свободно), и длинами очередей на входах ОА, принимающими неотрицательные целочисленные значения. Переменные, характеризующие состояние СМО, будем называть переменными состояния или фазовыми переменными.

Правило, согласно которому заявки выбирают из очередей на обслуживание, называют *дисциплиной обслуживания,* а величину, выражающую преимущественное право на обслуживание, - *приоритетом.* В *бесприоритетных дисциплинах* все транзакты имеют одинаковые приоритеты. Среди бесприоритетных дисциплин наиболее популярны дисциплины FIFO (первым пришел - первым обслужен), LIFO (последним пришел - первым обслужен) и со случайным выбором заявок из очередей.

В приоритетных дисциплинах для заявок каждого приоритета на входе ОА выделяется своя очередь. Заявка из очереди с низким приоритетом поступает на обслуживание, если пусты очереди с более высокими приоритетами. Различают приоритеты абсолютные, относительные и динамические. Заявка из очереди с более высоким абсолютным приоритетом, поступая на вход занятого ОА, прерывает уже начатое обслуживание заявки более низкого приоритета. В случае относительного приоритета прерывания не происходит, более высокоприоритетная заявка ждет окончания уже начатого обслуживания. Динамические приоритеты могут изменяться во время нахождения заявки в СМО.

Исследование поведения СМО, т. е. определение временных зависимостей переменных, характеризующих состояние СМО, при подаче на

входы любых требуемых в соответствии с заданием на эксперимент потоков заявок, называют *имитационным моделированием* СМО. Имитационное моделирование проводят путем воспроизведения событий, происходящих одновременно или последовательно в модельном времени. При этом под *событием* понимают факт изменения значения любой фазовой переменной.

Подход, альтернативный имитационному моделированию, называют аналитическим исследованием СМО. Аналитическое исследование заключается в получении формул для расчета выходных параметров СМО с последующей подстановкой значений аргументов в эти формулы в каждом отдельном эксперименте.

Модели СМО, используемые при имитационном и аналитическом моделировании, называются имитационными и аналитическими соответственно.

Аналитические модели удобны в использовании, поскольку для аналитического моделирования не требуются сколько-нибудь значительные затраты вычислительных ресурсов, часто без постановки специальных вычислительных экспериментов разработчик может оценить характер влияния аргументов на выходные параметры, выявить те или иные общие закономерности в поведении системы. Но, к сожалению, аналитическое исследование удается реализовать только для частных случаев сравнительно несложных СМО. Для сложных СМО аналитические модели если и удается получить, то только при принятии упрощающих допущений, ставящих под сомнение адекватность модели.

Поэтому основным подходом к анализу САПР на системном уровне проектирования считают имитационное моделирование, а аналитическое исследование используют при предварительной оценке различных предлагаемых вариантов систем.

Некоторые компоненты СМО характеризуются более чем одним входным и (или) выходным потоками заявок. Правила выбора одного из возможных направлений движения заявок входят в соответствующие модели компонентов. В одних случаях такие правила относятся к исходным данным (например, выбор направления по вероятности), но в некоторых случаях желательно найти оптимальное управление потоками в узлах разветвления. Тогда задача моделирования становится более сложной задачей синтеза, характерными примерами являются маршрутизация заявок или синтез расписаний и планов.

12.2. Аналитические модели СМО

Как отмечено выше, аналитические модели СМО удается получить при довольно серьезных допущениях. К числу типичных допущений относятся следующие.

Во-первых, как правило, считают, что в СМО используются бесприоритетные дисциплины обслуживания типа FIFO.

Во-вторых, времена обслуживания заявок в устройствах выбираются в соответствии с экспоненциальным законом распределения.

В-третьих, в аналитических моделях СМО входные потоки заявок аппроксимируются простейшими потоками, т. е. потоками, обладающими свойствами стационарности, ординарности (невозможности одновременного поступления двух заявок на вход СМО), отсутствия последействия.

В большинстве случаев модели СМО отображают процессы с конечным множеством состояний и с отсутствием последействия. Такие процессы называют *конечными марковскими цепями.*

Марковские цепи характеризуются множеством состояний S, матрицей вероятностей переходов из одного состояния в другое и начальными условиями (начальным состоянием). Удобно представлять марковскую цепь в виде графа, в котором вершины соответствуют состояниям цепи, дуги - переходам, веса дуг - вероятностям переходов (если время дискретно) или интенсивностям переходов (если время непрерывно).

Отметим, что интенсивностью перехода называют величину $V_{ij} = \lim P_{ij}(t_1)/t_1$ при $t_1 \to 0$, где $P_{ij}(t_1)$ - вероятность перехода из состояния S_i в состояние S_j за время t_1 Обычно используют условие

$$V_{ii} = -\sum_{j \neq i} V_{ij},$$

что эквивалентно

$$\sum_{j=1}^{N} V_{ij} = 0,$$

где N - число состояний. На рис. 12.1 - приведен пример марковской цепи в виде графа с состояниями $S_1,\ldots,S_4$ а в табл.12.1 представлена матрица интенсивностей переходов для этого примера.

Таблица 12.1.

Состояние	S_1	S_2	S_3	S_4
S_1	$-V_{12}-V_{13}-V_{14}$	V_{12}	V_{13}	V_{14}
S_2	V_{21}	$-V_{21}$	0	0
S_3	0	0	$-V_{34}$	V_{34}
S_4	0	V_{42}	0	$-V_{42}$

Большинство выходных параметров СМО можно определить, используя информацию о поведении СМО, т. е. информацию о состояниях СМО в установившихся (стационарных) режимах и об их изменениях в переходных процессах. Эта информация имеет вероятностную природу, что обусловливает описание поведения СМО в терминах вероятностей нахождения системы в различных состояниях. Основой такого описания, а следовательно, и многих аналитических моделей СМО являются *уравнения Колмогорова.*

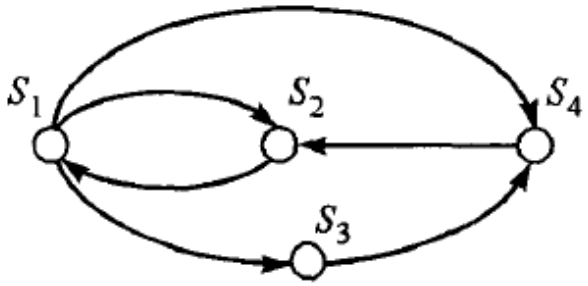


Рис.12.1. Пример марковской цепи

Уравнения Колмогорова можно получить следующим образом.

Изменение вероятности P нахождения системы в состоянии S_t за время есть вероятность перехода системы в состояние S_1 из любых других состояний за вычетом вероятности перехода из состояния S_t в другие состояния за время t_1 т. е.

$$P_i(t) = P_i(t+t_1) - P_i(t) = \sum_{j\in \mathbf{J}} P_{ji}(t_1)P_j(t) - \sum_{k\in \mathbf{K}} P_{ik}(t_1)P_i(t), \tag{12.1}$$

где $P_i(t)$ и $P_j(t)$ - вероятности нахождения системы в состояниях S_i и S_j соответственно в момент времени t, а $P_{ik}(t_1)$ и $P_{jk}(t_1)$ - вероятности изменения состояний в течение времени t_1, произведение вида $P_{i1}(t_1)P_j(t)$ есть безусловная вероятность перехода из S_j в S_i, равная условной вероятности перехода, умноженной на вероятность условия; $\boldsymbol{J}$ и $\boldsymbol{K}$ - множества индексов инцидентных вершин по отношению к вершине S_t по входящим и исходящим дугам на графе состояний соответственно.

Разделив выражение (12.1) на t_1 и перейдя к пределу при $t\to 0$, получим

$$\lim_{t\to 0} P_i(t)/t_1 = \sum_j \left(\lim_{t\to 0} P_{ji}/t_1\right)P_j - \sum_k \left(\lim_{t\to 0} P_{ik}/t_1\right)P_i,$$

откуда следуют уравнения Колмогорова

$$dP_i/dt = \sum_j (V_{ji}P_j) - P_i \sum_k V_{ik}. \tag{12.2}$$

В стационарном состоянии $dP_1/dt=0$ и уравнения Колмогорова составляют систему алгебраических уравнений, в которой i-й узел представлен уравнением

$$\sum_j (V_{ji}P_j) = P_i \sum_k V_{ik}. \tag{12.3}$$

Прибавляя $V_{ij}P_i$, к левой и правой частям уравнения (12.3) и учитывая (12.1), получаем

$$\sum_{j=1}^{N}(V_{ji}P_j) = P_i\sum_{k=1}^{N}V_{ik} = 0,$$

где P_j - финальные вероятности.

12.3. Пример аналитической модели

Примером СМО, к которой можно применить аналитические методы исследования, является одноканальная СМО с простейшим входным потоком интенсивностью λ и длительностью обслуживания, подчиняющейся экспоненциальному закону обслуживания интенсивностью μ. Для этой СМО нужно получить аналитические зависимости среднего числа N_A заявок, находящихся в системе, среднюю длину Q_{AV} очереди к OA, время T_{AV} пребывания заявки в системе, время T_m ожидания в очереди [4,5].

На рис. 12.1 представлен граф состояний рассматриваемой СМО, где S_k - состояние с k заявками в системе. Матрица интенсивностей представлена в табл.12.2. Уравнения Колмогорова для установившегося режима имеют вид

$$\begin{aligned}
&\lambda P_0 + \mu P_1 = 0;\\
&\lambda P_0 - (\lambda+\mu)P_1 + \mu P_2 = 0;\\
&\mu P_1 - (\lambda+\mu)P_2 + \lambda P_3 = 0;\\
&\mu P_2 - (\lambda+\mu)P_3 + \lambda P_4 = 0\\
&\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots
\end{aligned}$$

Используя уравнения Колмогорова, можно выразить все $P_i, i=1,2,3,\ldots$, через P_0. Получим

$$P_1 = \lambda P_0/\mu = aP_0;$$
$$P_2 = ((\lambda+\mu)P_1 - \lambda P_0)/\mu = (1+a)P_1 - aP_0 = a^2P_0;$$
$$P_3 = (1+a)P_2 - aP_1 = a^2P_1 = a^3P_0 \text{ и т. д.}$$

Здесь введено обозначение $a=\lambda/\mu$. Отметим также, что установившийся режим возможен только при $a<1$.

$$\text{Так как } \sum_{i=0}^{\infty}P_i = 1, \text{ то } P_0 = 1/(1+a+a^2+a^3+\ldots) = 1-a.$$

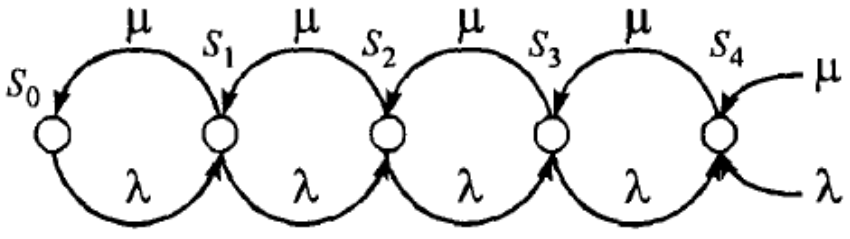


Рис.12.2. Граф состояний.

Таблица 12.2.

Состояние	S_0	S_1	S_2	S_3	S_4	...
S_0	$-\lambda$	λ	0	0	0	...
S_1	μ	$-\lambda-\mu$	λ	0	0	...
S_2	0	μ	$-\lambda-\mu$	λ	0	...
S_3	0	0	μ	$-\lambda-\mu$	λ	...
S_4	0	0	0	μ	$-\lambda-\mu$	...
...	...	...	...	...	...	...

Теперь не трудно получить и остальные требуемые результаты:

$$N_{av} = \sum_{k=1}^{\infty} P_k k = P_1 + 2P_2 + 3P_3 + ... = a(1 - a)(1 + 2a + 3a^2 + ...) =$$
$$= a(1 - a)/(1 - a)^2 = a/(1 - a);$$
$$Q_{av} = P_2 + 2P_3 + 3P_4 + ... = \sum_{k=2}^{\infty} (k - 1)P_k = P_0 a^2(1 + 2a + 3a^2 + ...) = a^2/(1 - a).$$

Времена пребывания в системе и очереди определяются соотношениями:

$$N_{av} = \lambda T_{av}$$
$$Q_{av} = \lambda T_{or},$$

которые называют *формулами Литтла*:

$$T_{av} = a/(1 - a)/\lambda = 1/(\mu - \lambda);$$
$$T_{or} = a^2/(1 - a)/\lambda = a/(\mu - \lambda).$$

12.4. Имитационное моделирование СМО.

Для представления имитационных моделей можно использовать языки программирования общего применения, однако такие представления оказываются довольно громоздкими. Поэтому обычно используют специальные языки имитационного моделирования на системном уровне. Среди языков имитационного моделирования различают языки, ориентированные на описание событий, средств обслуживания или маршрутов движения заявок (процессов). Выбор языка моделирования определяет структуру модели и методику ее построения.

Ориентация на устройства характерна для функционально-логического и более детальных иерархических уровней описания объектов.

Для описания имитационных моделей на системном уровне (иногда их называют *сетевыми имитационными моделями - СИМ)* чаще используют языки, ориентированные на события или процессы. Примерами первых могут служить языки Симскрипт, SMPL и ряд других. К числу вторых относятся языки Симула, SOL, а также популярный язык *GPSS.*

Языки имитационного моделирования реализуются в программно-методических комплексах моделирования СМО, имеющих ту или иную степень специализации. Так, комплексы на базе языка GPSS можно использовать во многих приложениях, но есть специализированные

комплексы для моделирования вычислительных сетей, систем управления предприятиями и т. п.

При использовании языков, ориентированных на процессы, в составе СИМ выделяются элементарные части и ими могут быть источники входных потоков заявок, устройства, накопители и узлы.

Источник входного потока заявок представляет собой алгоритм, в соответствии с которым вычисляются моменты t_k появления заявок на выходе источника. Источники могут быть зависимыми и независимыми. В зависимых источниках моменты появления заявок связаны с наступлением определенных событий, например с приходом другой заявки на вход некоторого устройства. Типичным независимым источником является алгоритм выработки значений t_k случайной величины с заданным законом распределения.

Устройства в имитационной модели представлены алгоритмами выработки значений интервалов (длительностей) обслуживания. Чаще всего это алгоритмы генерации значений случайных величин с заданным законом распределения. Но могут быть устройства с детерминированным временем обслуживания или временем, определяемым событиями в других частях СИМ. Модель устройства отображает также заданную дисциплину обслуживания, поскольку в модель входит алгоритм, управляющий очередями на входах устройства.

Накопители моделируются алгоритмами определения объемов памяти, занимаемых заявками, приходящими на вход накопителя. Обычно объем памяти, занимаемый заявкой, вычисляется как значение случайной величины, закон и (или) числовые характеристики распределения могут зависеть от типа заявки.

Узлы выполняют связующие, управляющие и вспомогательные функции в имитационной модели, например, для выбора направлений движения заявок в СИМ, изменения их параметров и приоритета, разделения заявок на части, их объединения и т. п.

Обычно каждому типу элементарной модели, за исключением лишь некоторых узлов, в программной системе соответствует определенная процедура (подпрограмма). Тогда СИМ можно представить как алгоритм, состоящий из упорядоченных обращений к этим процедурам, отражающим поведение моделируемой системы.

В процессе моделирования происходят изменения модельного времени, которое чаще всего принимается дискретным, измеряемым в тактах. Время изменяется после того, как закончена имитация очередной группы событий, относящихся к текущему моменту времени t_k. Имитация сопровождается накоплением в отдельном файле статистики таких данных, как количества заявок, вышедших из системы обслуженными и необслуженными, суммарное время занятого состояния для каждого из устройств, средние длины очередей и т. п. Имитация заканчивается, когда текущее время превысит заданный отрезок времени или когда входные источники выработают заданное число

заявок. После этого производят обработку накопленных в файле статистики данных, что позволяет получить значения требуемых выходных параметров.

12.5. Событийный метод моделирования

В программах имитационного моделирования СМО преимущественно реализуется *событийный метод* организации вычислений. Сущность событийного метода заключается в отслеживании на модели последовательности событий в том же порядке, в каком они происходили бы в реальной системе. Вычисления выполняют только для тех моментов времени и тех частей (процедур) модели, к которым относятся совершаемые события. Другими словами, обращения на очередном такте моделируемого времени осуществляются только к моделям тех элементов (устройств, накопителей), на входах которых в этом такте произошли изменения. Поскольку изменения состояний в каждом такте обычно наблюдаются лишь у малой доли ОА, событийный метод может существенно ускорить моделирование по сравнению с инкрементным методом, в котором на каждом такте анализируются состояния всех элементов модели.

Рассмотрим возможную схему реализации событийного метода имитационного моделирования.

Моделирование начинается с просмотра операторов генерирования заявок, т.е. с обращения к моделям источников входных потоков. Для каждого независимого источника такое обращение позволяет рассчитать момент генерации первой заявки. Этот момент вместе с именем - ссылкой на заявку - заносится в список будущих событий (СБС), а сведения о генерируемой заявке - в список заявок (СЗ). Запись в СЗ включает в себя имя заявки, значения ее параметров (атрибутов), место, занимаемое в данный момент в СИМ. События в СБС упорядочиваются по мере увеличения моментов наступления.

Затем из СБС выбирают совокупность сведений о событиях, относящихся к наиболее раннему моменту времени. Эта совокупность переносится в список текущих событий (СТС), из которого извлекаются ссылки на события. Обращение по ссылке к СЗ позволяет установить место в СИМ заявки *A,* с которой связано моделируемое событие. Пусть этим местом является устройство *X*. Далее программа моделирования выполняет следующие действия (рис. 12.3):

1) изменяет параметры состояния устройства *X;* например, если заявка *A* освобождает *X*, а очередь к *X* не была пуста, то в соответствии с заданной дисциплиной обслуживания из очереди к *X* выбирается заявка *B* и поступает на обслуживание в *X;*

2) прогнозируется время наступления следующего события, связанного с заявкой *B,* путем обращения к модели устройства *X,* в которой рассчитывается продолжительность обслуживания заявки *B;* сведения об этом будущем событии заносятся в СБС и СЗ;

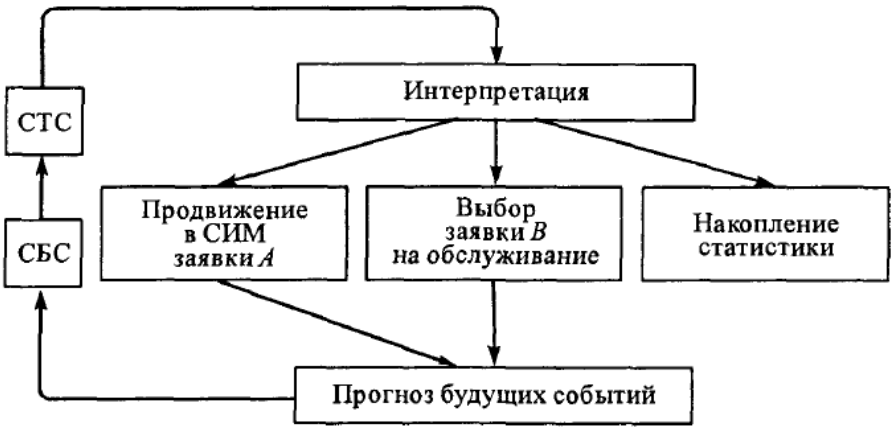


Рис. 12.3. Иллюстрация событийного моделирования

3) происходит имитация движения заявки A в СИМ по маршруту, определяемому заданной программой моделирования, до тех пор, пока заявка не придет на вход некоторого ОА; здесь либо заявка задерживается в очереди, либо путем обращения к модели этого ОА прогнозируется наступление некоторого будущего события, связанного с дальнейшей судьбой заявки A; сведения об этом будущем событии также заносятся в СБС и СЗ;

4) в файл статистики добавляются необходимые данные.

После отработки всех событий, относящихся к моменту времени t_k, происходит увеличение модельного времени до значения, соответствующего ближайшему будущему событию, и рассмотренный процесс имитации повторяется.

Контрольные вопросы

1. Найдите передаточную функцию для схемы на рис. 12.4.

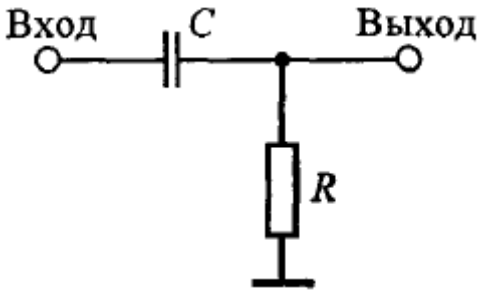


Рис.12.4. Дифференцирующая цепь

2. Постройте таблицы логических функций И и ИЛИ для пятизначного алфавита.

3. Поясните сущность событийного метода моделирования.

4. Приведите вывод уравнений Колмогорова для систем массового обслуживания.

5. Постройте граф состояний для системы массового обслуживания, состоящей из двух идентичных ОА с интенсивностью обслуживания μ каждый и включенных параллельно при общем входном потоке с интенсивностью поступления заявок λ. Если свободны оба ОА, пришедшая заявка занимает первый ОА. Если очередь равна 2, то приходящие заявки покидают систему без обслуживания.

Лекция № 13.
Тема: Математическое обеспечение подсистем машинной графики и геометрического моделирования. Задачи анализа и синтеза.

План

13.1. Компоненты математического обеспечения.

13.2. Геометрические модели

13.3. Методы и алгоритмы машинной графики (подготовки к визуализации).

Ключевые слова: машинная графика, геометрическое моделирование, каркасный модель, поверхностная модель, объёмный модель, конструктивная геометрия, форма Безье.

13.1. Компоненты математического обеспечения.

Подсистемы машинной графики и геометрического моделирования (МГиГМ) занимают центральное место в машиностроительных САПР-К. Конструирование изделий в них, как правило, проводится в интерактивном режиме при оперировании геометрическими моделями, т.е. математическими объектами, отображающими форму деталей, состав сборочных узлов и возможно некоторые дополнительные параметры (масса, момент инерции, цвета поверхности и т.п.).

В подсистемах МГиГМ типичный маршрут обработки данных включает в себя получение проектного решения в прикладной программе, его представление в виде геометрической модели (геометрическое моделирование), подготовку проектного решения к визуализации, собственно визуализацию в аппаратуре рабочей станции и при необходимости корректировку решения в интерактивном режиме. Две последние операции реализуются на базе аппаратных средств машинной графики. Когда говорят о математическом обеспечении МГиГМ, имеют в виду прежде всего модели, методы и алгоритмы для геометрического моделирования и подготовки к визуализации. При этом часто именно МО подготовки к визуализации называют МО машинной графики.

Различают МО двумерного (*2D*) и трехмерного (*3D*) моделирования. Основные применения 20-графики - подготовка чертежной документации в машиностроительных САПР, топологическое проектирование печатных плат

и кристаллов БИС в САПР электронной промышленности. В развитых машиностроительных САПР используют как *2D*-, так и *3D* - моделирование для синтеза конструкций, представления траекторий рабочих органов станков при обработке заготовок, генерации сетки конечных элементов при анализе прочности и т.п.

В *3D*-моделировании различают каркасные (проволочные), поверхностные, объемные (твердотельные) модели.

Каркасная модель представляет собой форму детали в виде конечного множества линий, лежащих на поверхностях детали. Для каждой линии известны координаты концевых точек и указана их инцидентность ребрам или поверхностям. Оперировать каркасной моделью на дальнейших операциях маршрутов проектирования неудобно, и поэтому каркасные модели в настоящее время используют редко.

Поверхностная модель отображает форму детали с помощью задания ограничивающих ее поверхностей, например, в виде совокупности данных о гранях, ребрах и вершинах.

Особое место занимают модели деталей с поверхностями сложной формы, так называемыми скульптурными поверхностями. К таким деталям относятся корпуса многих транспортных средств (например, судов, автомобилей), детали, обтекаемые потоками жидкостей и газов (лопатки турбин, крылья самолетов), и др.

Объемные модели отличаются тем, что в них в явной форме содержатся сведения о принадлежности элементов внутреннему или внешнему по отношению к детали пространству.

В настоящее время применяют следующие подходы к построению геометрических моделей.

1. Задание граничных элементов - граней, ребер, вершин.

2. Кинематический метод, согласно которому задают двумерный контур и траекторию его перемещения; след от перемещения контура принимают в качестве поверхности детали.

3. Позиционный подход, в соответствии с которым рассматриваемое пространство разбивают на ячейки (позиции) и деталь задают указанием ячеек, принадлежащих детали; очевидна громоздкость этого подхода.

4. Представление сложной детали в виде совокупностей *базовых элементов формы* (*БЭФ*) и выполняемых над ними теоретико-множественных операций. К БЭФ относятся заранее разработанные модели простых тел, это в первую очередь модели параллелепипеда, цилиндра, сферы, призмы. Типичными теоретико-множественными операциями являются объединение, пересечение, разность. Например, модель плиты с отверстием в ней может быть получена вычитанием цилиндра из параллелепипеда.

Метод на основе БЭФ часто называют *методом конструктивной геометрии.* Это основной способ конструирования сборочных узлов в современных САПР-К.

В памяти ЭВМ рассмотренные модели обычно хранятся в векторной форме, т.е. в виде координат совокупности точек, задающих элементы модели. Операции конструирования также выполняются над моделями в векторной форме. Наиболее компактна модель в виде совокупности связанных БЭФ, которая преимущественно и используется для хранения и обработки информации об изделиях в системах конструктивной геометрии.

Однако для визуализации в современных рабочих станциях в связи с использованием в них растровых дисплеев необходима растризация - преобразование модели в растровую форму. Обратную операцию перехода к векторной форме, которая характеризуется меньшими затратами памяти, называют векторизацией. В частности, векторизация должна выполняться по отношению к данным, получаемым сканированием изображений в устройствах автоматического ввода.

13.2. Геометрические модели

Важной составной частью геометрических моделей является описание поверхностей. Если поверхности детали - плоские грани, то модель может быть выражена достаточно просто определенной информацией о гранях, ребрах, вершинах детали. При этом обычно используется метод конструктивной геометрии. Представление с помощью плоских граней имеет место и в случае более сложных поверхностей, если эти поверхности аппроксимировать множествами плоских участков - полигональными сетками. Тогда можно поверхностную модель задать одной из следующих форм:

1) модель есть список граней, каждая грань представлена упорядоченным списком вершин (циклом вершин); эта форма характеризуется значительной избыточностью, так как каждая вершина повторяется в нескольких списках;

2) модель есть список ребер, для каждого ребра заданы инцидентные вершины и грани.

Однако аппроксимация полигональными сетками при больших размерах ячеек сетки дает заметные искажения формы, а при малых размерах ячеек оказывается неэффективной по вычислительным затратам. Поэтому более популярны описания неплоских поверхностей кубическими уравнениями в *форме Безье* или 5-сплайнов.

Знакомство с этими формами удобно выполнить, показав их применение для описания геометрических объектов первого уровня - пространственных кривых.

Примечание. Геометрическими объектами нулевого, первого и второго уровней называют соответственно точки, кривые, поверхности.

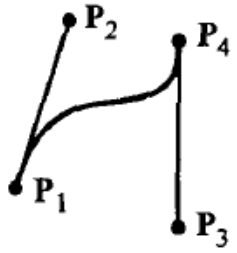


Рис.13.1. Кривая Безье.

В подсистемах МГиГМ используются параметрически задаваемые кубические кривые

$$x(t) = a_x t^3 + b_x t^2 + c_x t + d_x;$$
$$y(t) = a_y t^3 + b_y t^2 + c_y t + d_y;$$
$$z(t) = a_z t^3 + b_z t^2 + c_z t + d_z, \qquad (13.1)$$

где $1 \geq t \geq 0$. Такими кривыми описывают сегменты аппроксимируемой кривой, т.е. аппроксимируемую кривую разбивают на сегменты и каждый сегмент аппроксимируют уравнениями (13.1).

Применение кубических кривых обеспечивает (соответствующим выбором четырех коэффициентов в каждом из трех уравнений) выполнение четырех условий сопряжения сегментов. В случае кривых Безье этими условиями являются прохождение кривой сегмента через две заданные концевые точки и равенство в этих точках касательных векторов соседних сегментов. В случае 5-сплайнов выполняются условия непрерывности касательного вектора и кривизны (т.е. первой и второй производных) в двух концевых точках, что обеспечивает высокую степень «гладкости» кривой, хотя прохождение аппроксимирующей кривой через заданные точки здесь не обеспечивается. Применение полиномов выше третьей степени не рекомендуется, так как велика вероятность появления «волнистости».

В случае формы Безье коэффициенты в (13.1) определяются, во-первых, подстановкой в (13.1) значений $t = 0$ и $t = 1$ и координат заданных концевых точек P_1 и P_4 соответственно, во-вторых, подстановкой в выражения производных

$$dx/dt = 3a_x t^2 + 2b_x + c_x,$$
$$dy/dt = 3a_y t^2 + 2b_y t + c_y,$$
$$dz/dt = 3a_z t^2 + 2b_z t + c_z$$

тех же значений $t = 0$ и $t = 1$ и координат точек P_2 и P_3, задающих направления касательных векторов (рис.13.1). В результате для формы Безье получаем

$$x(t) = \mathbf{T}^{\mathrm{T}}\,\mathbf{M}\mathbf{G}_x;$$
$$y(t) = \mathbf{T}^{\mathrm{T}}\,\mathbf{M}\mathbf{G}_y;$$
$$z(t) = \mathbf{T}^{\mathrm{T}}\,\mathbf{M}\mathbf{G}_z,$$

(13.2)

Таблица 13.1.

– 1	3	–3	1
3	– 6	3	0
– 3	3	0	0
1	0	0	0

Таблица 13.2.

–1/6	1/2	–1/2	1/6
1/2	–1	1/2	0
–1/2	0	1/2	0
1/6	2/3	1/6	0

где $T^T = (t^3, t^2, t, 1)$ − вектор - строка, матрица M представлена в табл. 13.1, - вектор координат $\boldsymbol{P_{xt}}$ точек **P_1, P_2, P_3 и P_4,** аналогично G_y, G_z - векторы координат P_{y1}, P_{z1}, тех же точек.

В случае 5-сплайнов аппроксимируемая кривая делится на *n* участков, выделяемых последовательными точками $\mathbf{P_0}, \mathbf{P_1}, \mathbf{P_2}, \ldots, \mathbf{P_n}$. Участок между парой соседних точек $\mathbf{P_t}$ и $\mathbf{P_{t+1}}$ аппроксимируется 5-сплайном, построенным с использованием четырех точек P_{t-1}, P_t, P_{t+1}, P_{t+2}. 5-сплайн на участке $[P_t, P_{t+1}]$ может быть представлен выражениями, аналогичными (13.2),

$$x(t) = \mathbf{T}^{\mathrm{T}}\,\mathbf{M}\mathbf{G}_x;$$
$$y(t) = \mathbf{T}^{\mathrm{T}}\,\mathbf{M}\mathbf{G}_y;$$
$$z(t) = \mathbf{T}^{\mathrm{T}}\,\mathbf{M}\mathbf{G}_z,$$

для которых матрица M имеет иной вид и представлена в табл. 13.2, а векторы G_x, G_y, G_z содержат соответствующие координаты точек P_{t-1}, P_t, P_{t+1}, P_{t+2}.

Покажем, что в точках сопряжения для первой и второй производных аппроксимирующего выражения выполняются условия непрерывности, что требуется по определению 5-сплайна. Обозначим участок аппроксимирующего B-сплайна, соответствующий участку $[P_t, P_{t+1}]$ исходной кривой, через $[Q_t, \mathbf{Q}_{t+1}]$. Тогда для этого участка и координаты *x* в точке сопряжения Q_{t+1}, имеем $t=1$ и

$$dx(t)/dt|_{t=1} = [3t^2, 2t, 1, 0]\,\mathbf{M}\,[x_{t-1}, x_t, x_{t+1}, x_{t+2}]^{\mathrm{T}} = [3, 2, 1, 0]\,\mathbf{M}\,[x_{t-1}, x_t, x_{t+1}, x_{t+2}]^{\mathrm{T}} = (x_{t+2} - x_t)/2;$$
$$d^2x(t)/dt^2|_{t=1} = [6t, 2, 0, 0]\,\mathbf{M}\,[x_{t-1}, x_t, x_{t+1}, x_{t+2}]^{\mathrm{T}} = [6, 2, 0, 0]\,\mathbf{M}\,[x_{t-1}, x_t, x_{t+1}, x_{t+2}]^{\mathrm{T}} = x_t - 2x_{t+1} + x_{t+2}.$$

Для участка $[\mathbf{Q}_{t+1}, \mathbf{Q}_{t+2}]$ в той же точке $\mathbf{Q}_{t+1}$ имеем $t = 0$ и

$$dx(t)/dt|_{t=0} = [0, 0, 1, 0]\,\mathbf{M}\,[x_t, x_{t+1}, x_{t+2}, x_{t+3}]^{\mathrm{T}} = (x_{t+2} - x_t)/2;$$
$$d^2x(t)/dt^2|_{t=0} = [0, 2, 0, 0]\,\mathbf{M}\,[x_t, x_{t+1}, x_{t+2}, x_{t+3}]^{\mathrm{T}} = x_t - 2x_{t+1} + x_{t+2},$$

т.е. равенство производных в точке сопряжения на соседних участках подтверждает непрерывность касательного вектора и кривизны. Естественно,

что значение x_q координаты x точки Q_{t+1} аппроксимирующей кривой на участке $[Q_t, Q_{t+1}]$.

$$x_q = [1, 1, 1, 1]\,M\,[x_{t-1}, x_t, x_{t+1}, x_{t+2}]^T = (x_t + 4x_{t+1} + x_{t+2})/6$$

равно значению x_q, подсчитанному для той же точки на участке $[Q_{t+1}\ Q_{t+2}]$, но значения координат узловых точек x_q и x_{t+1} аппроксимирующей и аппроксимируемой кривых не совпадают.

Аналогично можно получить выражения для форм Безье и 5-сплайнов применительно к поверхностям с учетом того, что вместо (13.1) используются кубические зависимости от двух переменных.

13.3. Методы и алгоритмы машинной графики (подготовки к визуализации).

К методам машинной графики относят методы преобразования графических объектов, представления (развертки) линий в растровой форме, выделения окна, удаления скрытых линий, проецирования, закраски изображений.

Преобразование графических объектов выполняется с помощью операций переноса, масштабирования, поворота.

Перенос точки из положения **P** в новое положение **C** можно выполнять по формулам типа

$$C_{xt} = P_{xt} + \Delta x_t,$$

где Δx - приращение по координате x_t,. Однако удобнее операции преобразования представлять в единой матричной форме

$$\mathbf{C} = \mathbf{PT}, \tag{13.3}$$

где **T** - преобразующая матрица. При этом точки **C** и **P** в двумерном случае изображают векторами - строками 1×3, в которых кроме значений двух координат, называемых при таком представлении однородными, дополнительно указывают масштабный множитель ***W***. Тогда перенос для случая ***2D*** можно выразить в виде (13.4), где ***T*** есть табл. 13.3, а W=1.

Для операций масштабирования и поворота матрицы ***T*** представлены в табл. 3.4 и 13.5 соответственно, где m_x, m_y - масштабные множители, ср - угол поворота.

Таблица 13.3

1	0	0
0	1	0
Δx_1	Δx_2	1

Таблица 13.4.

m_x	0	0
0	m_y	0
0	0	1

Таблица 13.5.

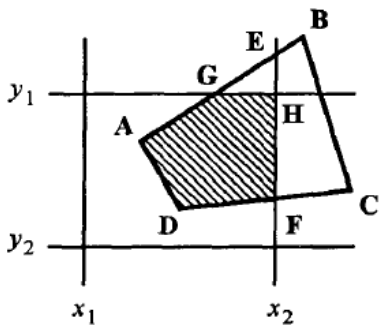
cos φ	sin φ	0
−sin φ	cos φ	0
0	0	1

Удобство (13.3) объясняется тем, что любую комбинацию элементарных преобразований можно описать формулой (13.3). Например, выражение для сдвига с одновременным поворотом имеет вид

$$\mathbf{C = PT_{сд}T_{пов} = PT,}$$

где $\mathbf{T=T_{сд}T_{пов}}$; $\mathbf{T_{сд}}$ - матрица сдвига; $\mathbf{T_{пов}}$ - матрица поворота.

Представление графических элементов в растровой форме требуется для отображения этих элементов на битовую карту растровой видеосистемы. Пусть требуется развернуть отрезок **AB** прямой $y=ax+b$, причем $1 \geq a \geq 0$ (при других значениях a рассматриваемый ниже алгоритм остается справедливым после определенных модификаций). Введем обозначения: **A**=*(xa, ya)*, **B** = *(xb, yb)*; за величину дискрета (пиксела) примем единицу. В алгоритме развертки номера строк и столбцов карты, на пересечении которых должны находиться точки отрезка, определяются следующим образом:



1) $\Delta x := xb - xa$;
$\Delta y := yb - ya$;
$x := xa$;
$y := ya$;
2) $d = 2\,\Delta y - \Delta x$;
3) если $d \geq 0$, то $\{y := y + 1; d := d + 2(\Delta y - \Delta x)\}$;
иначе $d := d + 2\,\Delta y$;
4) $x := x + 1$;
5) переход к пункту 3, пока не достигнута точка **B**.

Рис.13.2. Вкделение окна.

Экономичность этого алгоритма обусловливается отсутствием длинных арифметических операций типа умножения.

Выделение окна требуется при определении той части сцены, которая должна быть выведена на экран дисплея.

Пусть окно ограничено линиями $x = x_1$, $x = x_2$, $y = y_1$, $y = y_2$ (рис. 13.2). Поочередно для каждого многоугольника проверяется расположение его вершин и ребер относительно границ окна. Так, для многоугольника **ABCD** при отсечении по границе $x = x_2$ просматривается множество вершин в порядке обхода по часовой стрелке. Возможны четыре ситуации для двух последовательных вершин **P** и **R**:

1) если $x_P > x_2$ и $x_R > x_2$, то обе вершины и инцидентное им ребро находятся вне окна и исключаются из дальнейшего анализа;

2) если $x_p \leq x_2$ и $x_R \leq x_2$, то обе вершины и инцидентное им ребро остаются для дальнейшего анализа;

3) если $x_P < x_2$ и $x_R > x_2$, то вершина **P** остается в списке вершин, а вершина **R** заменяется новой вершиной с координатами $x = x_2$, $y = y_p + (y_R - y_p)(x_2 - x_p)/(x_R - x_p)$; в нашем примере такой новой вершиной будет **E**;

4) если $x_p > x_2$ и $x_R < x_2$, то вершина **P** заменяется новой вершиной с координатами $x = x_2$, $y = y_R + (y_p - y_R)(x_2 - x_R)/(x_p - x_R)$, а вершина **R** остается в

списке вершин; в нашем примере новой вершиной будет **F**.

После окончания просмотра применительно ко всем границам в окне оказываются оставшиеся в списке вершины.

Применяя эти правила в нашем примере, получаем сначала многоугольник AEFD, а после отсечения по верхней границе $y = y_2$ - многоугольник AGFD (см.рис.3.13). Однако правильный результат несколько иной, а именно многоугольник AGHFD. Этот правильный результат получается при двойном обходе вершин сначала по часовой стрелке, затем против с включением в список новых вершин, появляющихся при каждом обходе.

Применяют ряд алгоритмов *удаления скрытых линий.* Один из наиболее просто реализуемых алгоритмов - алгоритм *z* - буфера, где *z* - буфер - область памяти, число ячеек в которой равно числу пикселов в окне вывода. Предполагается, что ось *z* направлена по нормали к видовой поверхности и наблюдатель расположен в точке $z = 0$.

В начале исполнения алгоритма все пикселы соответствуют максимальному значению *z*, т.е. максимальному удалению от наблюдателя, что приводит к помещению во все ячейки *z* - буфера значений пикселов фонакартины (чертежа). Далее поочередно для всех точек граней рассчитываются значения координаты *z*. Среди точек, относящихся к одному и тому же пикселу (одной и той же ячейке *z* - буфера *S)*, выбирается точка с наименьшим значением *z* и ее код (т.е. цвет и яркость) помещается в *S*. В итоге *z* - буфер будет содержать пикселы наиболее близких к наблюдателю граней.

Алгоритмы построения проекций преобразуют трехмерные изображения в двумерные. В случае построения центральной проекции каждая точка трехмерного изображения отображается на картинную поверхность путем пересчета координат *x* и *y* (рис.13.3). Так, координату x'_a точки A′ вычисляют по очевидной формуле

$$x'_a = x_a d/z,$$

аналогично рассчитывается координата y'_a точки A'.

В параллельных проекциях $d \to \infty$ и координаты *x* и *y* точек A' и A совпадают. Поэтому построение параллельных проекций сводится к выделению окна, при необходимости к повороту изображения и возможно к удалению скрытых линий.

Закраска матовых поверхностей основана на законе Ламберта, согласно которому яркость отраженного от поверхности света пропорциональна $\cos \alpha$, где α - угол между нормалью к поверхности и направлением луча падающего света. В алгоритме Гуро яркость внутренних точек определяется линейной интерполяцией яркости в вершинах многоугольника. При этом сначала проводится интерполяция в точках ребер, а затем по строкам горизонтальной развертки. Более реалистичными получаются изображения в алгоритме

Фонга, основанном на линейной интерполяции векторов нормалей к поверхности.

Контрольные вопросы

1. Опишите на языке GPSS модель системы, состоящей из трех станков и обрабатывающей детали типов *A* и *B*. Заданы интенсивности поступления деталей этих типов и интенсивности обработки их на каждом станке. Маршруты деталей типа *A* включают станки 1 и 2, деталей типа *B* - станки 1 и 3.
2. Как и в предыдущем примере на входе системы имеются потоки деталей типов *A* и *B,* но система представляет собой сборочную линию, на выходе которой каждое изделие состоит из *n* деталей типа *A* и *m* деталей типа *B*. Требуется разработать модель системы и представить ее на языке GPSS.
3. Что такое «параметрическая модель» и «ассоциативное моделирование»?
4. Представьте матрицу преобразования, включающего сжатие плоского изображения в *к* раз и его перемещение вдоль оси *x* на величину *D*.
5. В чем заключается различие геометрических моделей Безье и B-сплайнов?

Лекция № 14.
Тема: Математическое обеспечение синтеза проектных решений. Постановка задач параметрического синтеза.

План

14.1. Место процедур синтеза в проектировании.

14.2. Критерии оптимальности.

14.3. Задачи оптимизации с учетом допусков

Ключевые слова: синтез, проектное решение, анализ, структурный синтез, параметрический синтез, оптимизация, работаспособность, аддитивный критерия.

14.1. Место процедур синтеза в проектировании

Сущность проектирования заключается в принятии проектных решений, обеспечивающих выполнение будущим объектом предъявляемых к нему требований. Синтез проектных решений - основа проектирования; от успешного выполнения процедуры синтеза в определяющей мере зависят потребительские свойства будущей продукции. Конечно, анализ -

необходимая составная часть проектирования, служащая для верификации принимаемых проектных решений. Именно анализ позволяет получить необходимую информацию для целенаправленного выполнения процедур синтеза в итерационном процессе проектирования. Поэтому синтез и анализ неразрывно связаны.

Как отмечено выше указанных лекциях, синтез подразделяют на параметрический и структурный. Проектирование начинается со *структурного синтеза*, при котором генерируется принципиальное решение. Таким решением может быть облик будущего летательного аппарата, или физический принцип действия датчика, или одна из типовых конструкций двигателя, или функциональная схема микропроцессора. Но эти конструкции и схемы выбирают в параметрическом виде, т.е. без указания числовых значений параметров элементов. Поэтому, прежде чем приступить к верификации проектного решения, нужно задать или рассчитать значения этих параметров, т.е. выполнить *параметрический синтез.* Примерами результатов параметрического синтеза могут служить геометрические размеры деталей в механическом узле или в оптическом приборе, параметры электрорадио элементов в электронной схеме, параметры режимов резания в технологической операции и т.п.

В случае если по результатам анализа проектное решение признается неокончательным, то начинается процесс последовательных приближений к приемлемому варианту проекта. Во многих приложениях для улучшения проекта удобнее варьировать значения параметров элементов, т.е. использовать параметрический синтез на базе многовариантного анализа. При этом задача параметрического синтеза может быть сформулирована как задача определения значений параметров элементов, наилучших с позиций удовлетворения требований технического задания при неизменной структуре проектируемого объекта. Тогда параметрический синтез называют параметрической оптимизацией, или просто *оптимизацией.* Если параметрический синтез не приводит к успеху, то повторяют процедуры структурного синтеза, т.е. на очередных итерациях корректируют или перевыбирают структуру объекта [4,5].

14.2. Критерии оптимальности.

Процедуры параметрического синтеза в САПР либо выполняются человеком в процессе многовариантного анализа (в интерактивном режиме), либо реализуются на базе формальных методов оптимизации (в автоматическом режиме). В последнем случае находят применение несколько постановок задач оптимизации.

Наиболее распространенной является детерминированная постановка: заданы условия работоспособности на выходные параметры **Y** и нужно найти номинальные значения проектных параметров X, к которым относятся параметры всех или части элементов проектируемого объекта. Назовем эту задачу оптимизации *базовой.* В частном случае, когда требования к выходным параметрам заданы нечетко, к числу рассчитываемых величин могут быть отнесены также нормы выходных параметров, фигурирующие в их условиях работоспособности.

Если проектируются изделия для дальнейшего серийного производства, то важное значение приобретает такой показатель, как процент выпуска годных изделий в процессе производства. Очевидно, что успешное выполнение условий работоспособности в номинальном режиме не гарантирует их выполнения при учете производственных погрешностей, задаваемых допусками параметров элементов. Поэтому целью оптимизации становится максимизация процента выхода годных, а к результатам решения задачи оптимизации относятся не только номинальные значения проектных параметров, но и их допуски.

Базовая задача оптимизации ставится как задача математического программирования

$$\underset{\mathbf{X}\in\mathbf{D}_x}{\mathbf{extr}}\, F(\mathbf{X}),$$
$$\mathbf{D}_x = \{\mathbf{X} |\, \varphi(\mathbf{X}) > 0,\ \psi(\mathbf{X}) = 0\}, \qquad (14.1)$$

где $\boldsymbol{F(X)}$ - целевая функция; $\boldsymbol{X}$ - вектор управляемых (проектных) параметров; $\boldsymbol{\varphi(X)}$ и $\boldsymbol{\psi(X)}$ - функции-ограничения; **D** - допустимая область в пространстве управляемых параметров. Запись (14.1) интерпретируется как задача поиска экстремума целевой функции путем варьирования управляемых параметров в пределах допустимой области.

Таким образом, для выполнения расчета номинальных значений параметров необходимо, вопервых, сформулировать задачу в виде (14.1), вовторых, решить задачу поиска экстремума **F(X).**

Сложность постановки оптимизационных проектных задач обусловлена наличием у проектируемых объектов нескольких выходных параметров, которые могут быть критериями оптимальности, но в задаче (14.1) целевая функция должна быть одна. Другими словами, проектные задачи являются многокритериальными, и возникает проблема сведения многокритериальной задачи к однокритериальной.

Применяют несколько способов выбора критерия оптимальности.

В *частном критерии* среди выходных параметров один выбирают в качестве целевой функции, а условия работоспособности остальных выходных параметров относят к ограничениям задачи (14.1). Эта постановка вполне приемлема, если действительно можно выделить один наиболее критичный выходной параметр. Но в большинстве случаев сказывается недостаток частного критерия (рис.14.1).

На этом рисунке представлено двумерное пространство выходных параметров y_1 и y_2, для которых заданы условия работоспособности $y_1 < T_1$ и $y_2 < T_2$. Кривая АВ является границей достижимых значений выходных параметров. Это ограничение объективное и связано с существующими физическими и технологическими условиями производства, называемыми условиями реализуемости. Область, в пределах которой выполняются все условия реализуемости и работоспособности, называют *областью работоспособности.* Множество точек пространства выходных параметров, из которых невозможно перемещение, приводящее к улучшению всех выходных параметров, называют областью компромиссов или *областью Парето.* Участок кривой АВ (см. рис.14.1) относится к области Парето.

Если в качестве целевой функции в ситуации рис.14.1. выбрать параметр y_1 то результатом оптимизации будут параметры **X,** соответствующие точке **В**. Но это граница области работоспособности, и, следовательно, при нестабильности внутренних и внешних параметров велика вероятность выхода за пределы области работоспособности. Конечно, результаты можно улучшить, если применять так называемый метод уступок, при котором в качестве ограничения принимают условие работоспособности со скорректированной нормой в виде

$$y_2 < T_2 + \Delta,$$

где Δ - уступка. Но возникает проблема выбора значений уступок, т.е. результаты оптимизации будут иметь субъективный характер. Очевидно, что ситуация не изменится, если целевой функцией будет выбран параметр y_2, так как оптимизация приведет в точку **А**.

Аддитивный критерий объединяет (свертывает) все выходные параметры (частные критерии) в одну целевую функцию, представляющую собой взвешенную сумму частных критериев

$$F(\mathbf{X}) = \sum_{j=1}^{m} \omega_j y_j(\mathbf{X}), \qquad (14.2)$$

где ω_j - весовой коэффициент; m - число выходных параметров. Функция (14.2) подлежит минимизации, при этом если условие работоспособности имеет вид $y_j > T$, то $\omega_j < 0$.

Недостатки аддитивного критерия - субъективный подход к выбору весовых коэффициентов и неучет требований ТЗ. Действительно, в (14.2) не входят нормы выходных параметров.

Аналогичные недостатки присущи и мультипликативному *критерию,* целевая функция которого имеет вид

$$F(\mathbf{X}) = \prod_{j=1}^{m} y_j^{\omega_j}(\mathbf{X}). \qquad (14.3)$$

Нетрудно видеть, что если прологарифмировать (14.3), то мультипликативный критерий превращается в аддитивный.

Более предпочтительным является *максиминный критерий,* в качестве целевой функции которого принимают выходной параметр, наиболее неблагополучный с позиций выполнения условий работоспособности. Для оценки степени выполнения условия работоспособности *j*-го выходного параметра вводят *запас работоспособности* этого параметра *S* и этот запас можно рассматривать как нормированный *j* - й выходной параметр. Например (здесь и далее для лаконичности изложения предполагается, что все выходные параметры приведены к виду, при котором условия работоспособности становятся неравенствами в форме $y_j < T_j$*):*

$$S_j = (T_j - y_j)/T_j,$$

или

$$S_j = (T_j - y_{\text{ном } j})/\delta_j,$$

где $y_{\text{НОМJ}}$ - номинальное значение, а δ_j - некоторая характеристика рассеяния *j*-го выходного параметра, например, трехсигмовый допуск. Тогда целевая функция в максиминном критерии есть

$$F(\mathbf{X}) = \min_{j \in [1\ m]} S_j(\mathbf{X}).$$

Здесь запись [1: m] означает множество целых чисел в диапазоне от 1 до *m*. Задача (14.1) при максиминном критерии конкретизируется следующим образом:

$$F(\mathbf{X}) = \max_{\mathbf{X}\in \mathbf{D}_x} \min_{j\in[1\ m]} S_j(\mathbf{X}), \tag{14.4}$$

где допустимая область D_x определяется только прямыми ограничениями на управляемые параметры x_t:

$$x_{t\,\min} < x_t < x_{t\,\max}.$$

14.3. Задачи оптимизации с учетом допусков

Содержательную сторону оптимизации с учетом допусков поясняет рис. 14.2, на котором представлены области работоспособности и допусковая в двумерном пространстве управляемых параметров. Если собственно допуски заданы и не относятся к управляемым параметрам, то цель оптимизации - максимальным образом совместить эти области так, чтобы вероятность выхода за пределы области работоспособности была минимальной.

Решение этой задачи исключительно трудоемко, так как на каждом шаге оптимизации нужно выполнять оценку упомянутой вероятности методами статистического анализа, а для сложных моделей объектов таким методом является метод статистических испытаний. Поэтому на практике подобные задачи решают, принимая те или иные допущения.

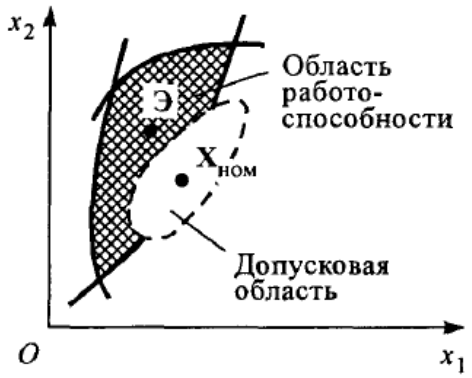


Рис.14.2. Области допусковая и работоспособности

Например, если допустить, что цель оптимизации достигается при совмещении центров областей работоспособности **Э** и допусковой $\mathbf{X}_{\text{ном}}$, то оптимизация сводится к *задаче центрирования,* т. е. к определению центра **Э**. Задачу центрирования обычно решают путем предварительного нормирования управляемых параметров x с последующим вписыванием

гиперкуба с максимально возможными размерами в нормированную область работоспособности.

Примечание. Нормирование проводят таким образом, что допусковая область приобретает форму гиперкуба, получающегося после нормирования.

Очевидно, что решение задачи центрирования позволяет не только оптимизировать номинальные значения проектных параметров, но и их допуски, если последние относятся к управляемым параметрам.

Контрольные вопросы

1. Дайте формулировку задачи математического программирования.
2. В чем заключаются трудности решения многокритериальных задач оптимизации?
3. Что такое «множество Парето»?
4. Для функции, заданной своими линиями равного уровня (рис. 14.3) постройте траектории поиска методами конфигураций, деформируемого многогранника, наискорейшего спуска из исходной точки X_0?

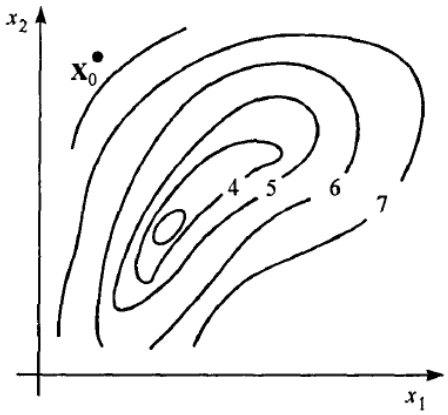


Рис.14.3. Пример для построения траекторий поиска

5. Как вы считаете, можно ли применять метод проекции градиента для решения задач оптимизации с ограничениями типа неравенств?

Лекция № 15.
Тема: Обзор методов оптимизации.

План

15.1. Классификация методов математического программирования.

15.2. Методы одномерной оптимизации.

15.3. Методы безусловной оптимизации.

15.4. Необходимые условия экстремума

Ключевые слова: одномерное оптимизация, многомерное оптимизация, экстремум, локальный метод, аппроксимация, овраг, метод конфигураций.

15.1. Классификация методов математического программирования

Основными методами оптимизации в САПР являются поисковые методы, которые основаны на пошаговом изменении управляемых параметров

$$\mathbf{X}_{k+1} = \mathbf{X}_k + \Delta\mathbf{X}_k, \qquad (15.1)$$

где в большинстве методов приращение $\Delta X_к$ вектора управляемых параметров вычисляется по формуле

$$\Delta\mathbf{X}_k = h\mathbf{g}(\mathbf{X}_k). \qquad (15.2)$$

Здесь $\boldsymbol{X}_к$ - значение вектора управляемых параметров на *к*-м шаге; *h* - шаг; $\mathbf{g(X_k)}$ - направление поиска. Следовательно, если выполняются условия сходимости, то реализуется пошаговое (итерационное) приближение к экстремуму.

Методы оптимизации классифицируют по ряду признаков.

В зависимости от числа управляемых параметров различают методы *одномерной* и *многомерной оптимизации,* в первых из них управляемый параметр единственный, во вторых размер вектора **X** не менее двух. Реальные задачи в САПР многомерны, методы одномерной оптимизации играют вспомогательную роль на отдельных этапах многомерного поиска.

Различают методы условной и безусловной оптимизации по наличию или отсутствию ограничений. Для реальных задач характерно наличие ограничений, однако методы безусловной оптимизации также представляют интерес, поскольку задачи условной оптимизации с помощью специальных методов могут быть сведены к задачам без ограничений.

В зависимости от числа экстремумов различают задачи одно - и многоэкстремальные. Если метод ориентирован на определение какого-либо локального экстремума, то такой метод относится к *локальным методам.* Если же результатом является глобальный экстремум, то метод называют *методом глобального поиска.* Удовлетворительные по вычислительной эффективности методы глобального поиска для общего случая отсутствуют, и потому на практике в САПР используют методы поиска локальных экстремумов.

Наконец, в зависимости от того, используются при поиске производные целевой функции по управляемым параметрам или нет, различают методы нескольких порядков. Если производные не используются, то имеет место *метод нулевого порядка,* если используются первые или вторые производные, то соответственно метод первого или второго порядка. Методы первого порядка называют также градиентными, поскольку вектор первых производных ***F(X)*** по **X** есть градиент целевой функции

$$\mathbf{grad}\ (F(\mathbf{X})) = (\partial F/\partial x_1,\ \partial F/\partial x_2,\ ...,\ \partial F/\partial x_n).$$

Конкретные методы определяются следующими факторами:

1) способом вычисления направления поиска $\mathbf{g(X_k)}$ в формуле (15.2);
2) способом выбора шага *h;*
3) способом определения окончания поиска.

Определяющим фактором является первый из перечисленных в этом списке, он подробно описан далее.

Шаг может или быть постоянным, или выбираться исходя из одномерной оптимизации - поиска минимума целевой функции в выбранном направлении $\mathbf{g(X_k)}$. В последнем случае шаг будем называть оптимальным.

Окончание поиска обычно осуществляют по правилу: если на протяжении *r* подряд идущих шагов траектория поиска остается в малой ε-окрестности текущей точки поиска $\boldsymbol{X_к}$, то поиск следует прекратить, следовательно, условие окончания поиска имеет вид $|\mathbf{X}_k - \mathbf{X}_{k-r}| < \varepsilon$.

15.2. Методы одномерной оптимизации.

К методам одномерной оптимизации относятся методы дихотомического деления, золотого сечения, чисел Фибоначчи, полиномиальной аппроксимации и ряд их модификаций.

Пусть задан отрезок [A, B], на котором имеется один минимум (в общем случае нечетное число минимумов). Согласно *методу дихотомического деления* (рис. 15.1, *а)* отрезок делят пополам и в точках, отстоящих от центра *C* отрезка на величину допустимой погрешности *q,* рассчитывают значения целевой функции *F(C + q)* и *F(C - q).* Если окажется, что *F(C + q) > F(C - q),* то минимум находится на отрезке *[A, C],* если *F(C + q) < F(C - q),* то минимум - на [C,B], если *F(C + q) = F(C - q)* - на [C - *q*,*C + q*]. Таким образом, на следующем шаге вместо отрезка *[A, B]* нужно исследовать суженный отрезок *[A,C], [C, B]* или [*C - q*,*C + q*]. Шаги повторяются, пока длина отрезка не уменьшится до значения погрешности *q.* Таким образом, требуется не более *N* шагов, где *N* - ближайшее к log *((B-A)/q)* целое значение, но на каждом шаге целевую функцию следует вычислять дважды.

В соответствии с *методом золотого сечения* (рис.15.3, *б*) внутри отрезка *[A,B]* выделяют две промежуточные точки C, и Z), на расстоянии $s = aL$ от его конечных точек, где $L = B - A$ - длина отрезка. Затем вычисляют значения целевой функции *F(x)* в точках C_1 и D_1. Если $F(C_1) < F(D_1)$, то минимум находится на отрезке $[A,D_1]$, если $F(C_1) > F(D_1)$), то - на отрезке $[C_1, B]$, если $F(C_1) = F(D_1)$ - на отрезке $[C_1, B]$. Следовательно, вместо отрезка *[A,B]* теперь можно рассматривать отрезок $[A,D_1]$, $[C_1,B]$ или $[C_1,D_1]$, т. е. длина отрезка уменьшилась не менее чем в $L/(L - aL) = 1/(1 - a)$ раз. Если подобрать значение *a* так, что в полученном отрезке меньшей длины одна из промежуточных точек совпадет с промежуточной точкой от предыдущего шага, т.е. в случае выбора отрезка $[A,D_1]$ точка D_2 совпадет с точкой C_1, а в случае выбора отрезка $[C_1, B]$ точка C_2 - с точкой D_1, то это позволит сократить число вычислений целевой функции на всех шагах (кроме первого) в 2 раза.

Условие получения такого значения *a* формулируется следующим образом: $(1 - 2a)L_k = aL_{k\,1}$, откуда с учетом того, что $L_k/L_{k-l}=1(1 - a)$, имеем $a = 0{,}382$. Это значение и называют золотым сечением.

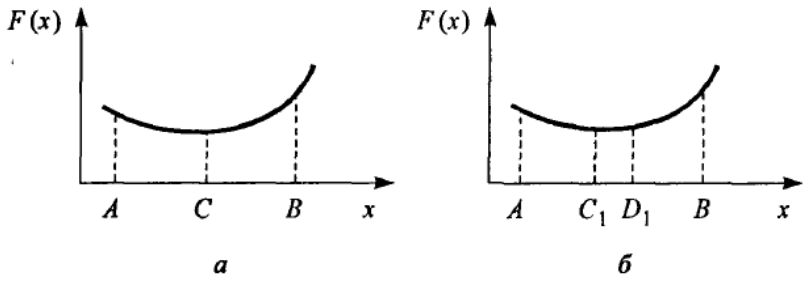


Рис.15.1. Методы дихотомического деления (а) и золотого сечения *(б)*

Таким образом, требуется не более *N* шагов и $N + 1$ вычисление целевой функции, где *N* можно рассчитать, используя соотношение $(B - A)/E = (1 - a)N$ при заданной погрешности *E* определения экстремума.

Согласно *методу чисел Фибоначчи,* используют числа Фибоначчи *R*, последовательность которых образуется по правилу $R_{l+1}=R_{l+1}+R_l$ при $R_0=R_1=1$, т.е. ряд чисел Фибоначчи имеет вид 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144 ... Метод аналогичен методу золотого сечения с тем отличием, что коэффициент *a* равен отношению R_{i-2}/R_i, начальное значение *i* определяется из условия, что R_i должно быть наименьшим числом Фибоначчи, превышающим величину $(B-A)/E$, где *E* - заданная допустимая погрешность определения экстремума. Так, если $(B - A)/E = 100$, то начальное значение $i = 12$, поскольку $R_1=144$, и $a = 55/144 = 0{,}3819$, на следующем шаге будет $a = 34/89 = 0{,}3820$ и т.д.

В соответствии с *методом полиномиальной аппроксимации* при аппрокимации *F(x)* квадратичным полиномом

$$P(x) = a_0 + a_1 x + a_2 x^2 \qquad (15.3)$$

выбирают промежуточную точку С и в точках *A, B, C* вычисляют значения целевой функции. Далее решают систему из трех алгебраических уравнений, полученных подстановкой в (15.3) значений *A*, *B, C* вместо *x* и вычисленных значений функции вместо *P(x).* В результате становятся известными значения коэффициентов $a_к$ в (15.3), и, исходя из условия *dP(x)/dx*=0, определяют экстремальную точку Э полинома. Например, если точка *C* выбрана в середине отрезка *[A, B],* то Э = С + (С - *A)(F(A) - F(B))* / (2*(F(A) - 2F(C) + F(B)))*.

15.3. Методы безусловной оптимизации.

Среди методов нулевого порядка в САПР находят применение методы Ро - зенброка, конфигураций, деформируемого многогранника, случайного поиска. К методам с использованием производных относятся методы наискорейшего спуска, сопряженных градиентов, переменной метрики.

Метод Розенброка является улучшенным вариантом метода покоординатного спуска.

Метод покоординатного спуска характеризуется выбором направлений поиска поочередно вдоль всех *n* координатных осей, шаг рассчитывается на основе одномерной оптимизации, критерий окончания поиска $|X_k - X_{k-n}| < \varepsilon$, где ε - заданная точность определения локального экстремума, *n* - размерность пространства управляемых параметров. Траектория покоординатного спуска для примера двумерного пространства управляемых параметров показана на рис.15.2, где X_k - точки на траектории поиска, x_t - управляемые параметры. Целевая функция представлена своими линиями равного уровня, около каждой линии записано соответствующее ей значение ***F(X)***. Очевидно, что Э есть точка минимума.

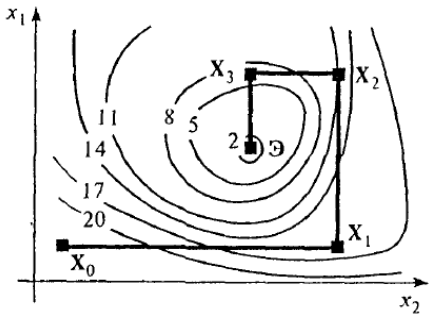


Рис. 15.2. Траектория покоординатного спуска

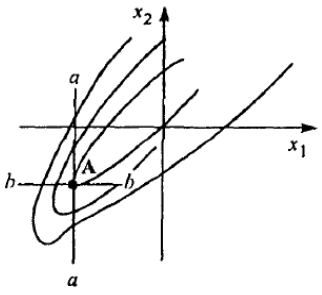


Рис.15.3. «Застревание» спуска покоординатного спуска на дне оврага

При использовании метода покоординатного спуска велика вероятность «застревания» поиска на дне оврага вдали от точки экстремума. На рис. 15.3 видно, что после попадания в точку **А**, расположенную на дне оврага, дальнейшие шаги возможны лишь в направлениях *аа* или *bb,* но они приводят к ухудшению целевой функции. Следовательно, поиск прекращается в точке **А.**

Примечание. *Оврагом* называют часть пространства управляемых параметров, в которой наблюдаются слабые изменения производных целевой функции по одним направлениям и значительные изменения с переменой знака по некоторым другим направлениям. Знак производной меняется в точках, принадлежащих дну оврага.

В то же время при благоприятной ориентации дна оврага, а именно при положении одной из координатных осей, близком к параллельности с дном оврага, поиск оказывается весьма быстрым. Эта ситуация показана на рис. 15.4.

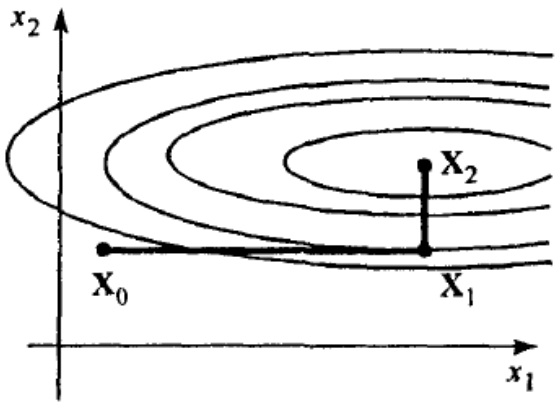
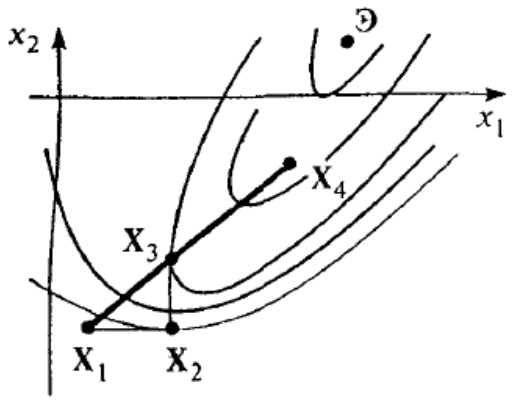


Рис.15.4. Траектория по координатного спуска при благоприятной ориентации координатных осей

Метод Розенброка заключается в таком повороте координатных осей, чтобы одна из них оказалась квазипараллельной дну оврага. Такой поворот осуществляют на основе данных, полученных после серии из n шагов покоординатного спуска. Положение новых осей s_t может быть получено линейным преобразованием прежних осей x_t: ось s_1 совпадает по направлению с вектором X_{k+n}-$X_к$; остальные оси выбирают из условия ортогональности к X_i, и друг к другу.

Другой удачной модификацией покоординатного спуска является *метод конфигураций {Хука - Дживса).* В соответствии с этим методом вначале выполняют обычную серию из n шагов покоординатного спуска, затем делают дополнительный шаг в направлении вектора $X_к$-X_{k+n}, как

показано на рис. 15.5, где дополнительный шаг выполняют в направлении вектора X_3- X_i, что и приводит в точку X_4.



15.5. Иллюстрация метода конфигураций

Поиск экстремума *методом деформируемого многогранника (Нелдера - Мида)* основан на построении многогранника с (n+1) вершинами на каждом шаге поиска, где n - размерность пространства управляемых параметров. В начале поиска эти вершины выбирают произвольно, на последующих шагах выбор подчинен правилам метода.

Эти правила поясняются рис.15.6 на примере двумерной задачи оптимизации. Выбраны вершины исходного треугольника: X_1, X_2, X_3. Новая вершина X_4 находится на луче, проведенном из худшей вершины X_1 (из вершины с наибольшим значением целевой функции) через центр тяжести ЦТ многогранника, причем рекомендуется X_4 выбирать на расстоянии d от ЦТ, равном |ЦТ-X_t,|. Новая вершина X_4 заменяет худшую вершину X_t. Если оказывается, что X_4 имеет лучшее значение целевой функции среди вершин многогранника, то расстояние d увеличивают. На рисунке именно эта ситуация имеет место и увеличение d дает точку X_5. В новом многограннике с вершинами X_2, X_3, X_5 худшей является вершина X_2, аналогично получают вершину X, затем вершину X_7 и т.д. Если новая вершина окажется худшей, то в многограннике нужно сохранить лучшую вершину, а длины всех ребер уменьшить, например вдвое (стягивание многогранника к лучшей вершине). Поиск прекращается при выполнении условия уменьшения размеров многогранника до некоторого предела.

Случайные методы поиска характеризуются тем, что направления поиска g выбирают случайным образом.

Особенностью *метода наискорейшего спуска* является выполнение шагов поиска в градиентном направлении

$$\mathbf{X}_{k+1} = \mathbf{X}_k + h_k \, \mathbf{grad}\, F(\mathbf{X}_k) \,/\, |\mathbf{grad}\, F(\mathbf{X}_k)|,$$

шаг h_k выбирается оптимальным с помощью одномерной оптимизации.

При использовании метода наискорейшего спуска, как и большинства других методов, эффективность поиска существенно снижается в овражных ситуациях. Траектория поиска приобретает зигзагообразный вид с медленным продвижением вдоль дна оврага в сторону экстремума. Чтобы повысить эффективность градиентных методов, используют несколько приемов.

Один из приемов, использованный в *методе сопряженных градиентов* (называемом также методом Флетчера-Ривса), основан на понятии сопряженности векторов. Векторы **A** и **B** называют **Q** - сопряженными, если $\mathbf{A^TQB = O}$, где **Q** - положительно определенная квадратная матрица того же порядка, что и размер *N* векторов **A** и **B** (частный случай сопряженности - ортогональность векторов, когда **Q** является единичной матрицей порядка *N);* $\mathbf{A^T}$ - вектор строка; **B** вектор-столбец.

Особенность сопряженных направлений для **Q = Г**, где **Г** - матрица Гессе, в задачах с квадратичной целевой функцией **F(X)** заключается в следующем: одномерная минимизация **F(X)** последовательно по *N* сопряженным направлениям позволяет найти экстремальную точку не более чем за *N* шагов.

Примечание. *Матрицей Гессе* называют матрицу вторых частных производных целевой функции по управляемым параметрам.

Основанием для использования поиска по Г-сопряженным направлениям является то, что для функций *F(X)* общего вида может быть применена квадратичная аппроксимация, что на практике выливается в выполнение поиска более чем за *N* шагов.

15.4. Необходимые условия экстремума

В задачах безусловной оптимизации необходимые условия представляют собой равенство нулю градиента целевой функции

$$\mathbf{grad}\ F(X) = \mathbf{0}.$$

В общей задаче математического программирования необходимые условия экстремума, называемые *условиями Куна - Танкера,* формулируются следующим образом.

Для того чтобы точка **Э** была экстремальной точкой выпуклой задачи математического программирования (ЗМП), необходимо наличие неотрицательных коэффициентов u_i, таких, что

$$u_i \varphi_i(\mathbf{Э}) = 0,\ i = 1, 2, \ldots, m, \qquad (15.4)$$

и соблюдение при этом ограничений задачи, а также выполнение условия

$$\mathbf{grad}\, F(\text{Э}) + \sum_{i=1}^{m} u_i \mathbf{grad}\, \varphi_i(\text{Э}) + \sum_{j=1}^{L} a_j \psi_j(\text{Э}) = 0, \qquad (15.5)$$

где m - число ограничений типа неравенств; L - то же типа равенств; $a_j > 0$ - коэффициенты.

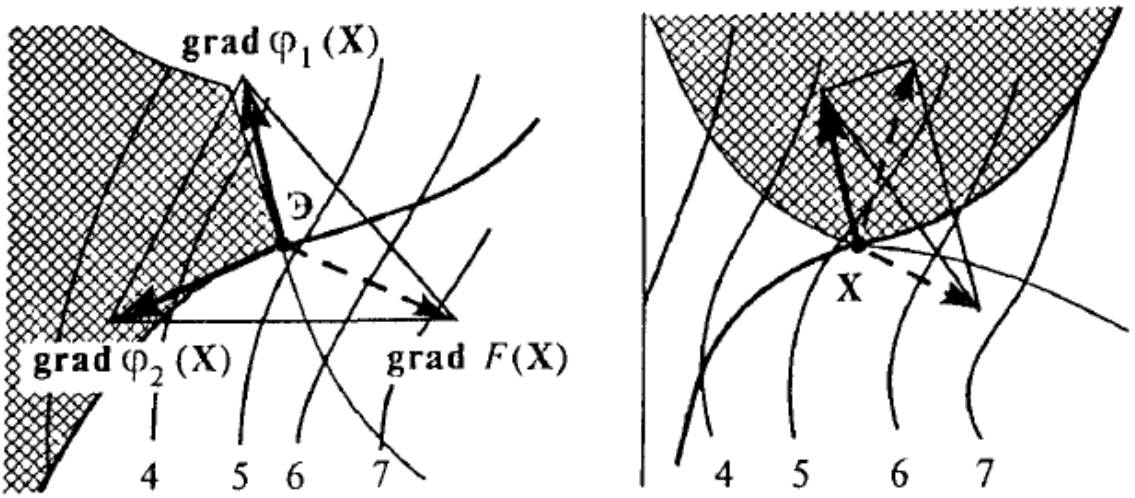


Рис. 15.6. К пояснению условий Куна - Таккера

За приведенной абстрактной формулировкой условий скрывается достаточно просто понимаемый геометрический смысл. Действительно, рассмотрим сначала случай с ограничениями только типа неравенств. Если максимум находится внутри допустимой области **R**, то, выбирая все $u_i = 0$, добиваемся выполнения (15.4); если же точка максимума **Э** лежит на границе области **R**, то, как видно из левой части рис.15.6, эту точку всегда соответствующим подбором неотрицательных u_i можно поместить внутрь оболочки, натянутой на градиенты целевой функции ***F(X)*** и функций-ограничений **φ(X)**. Наоборот, если точка не является экстремальной, то (15.4) нельзя выполнить при любом выборе положительных коэффициентов и (см. правую часть рис.15.6, где рассматриваемая точка **X** лежит вне выпуклой оболочки, натянутой на градиенты). Учет ограничений типа равенств очевиден, если добавляется последняя из указанных в (15.5) сумма.

Контрольные вопросы

1. Как вы считаете, можно ли применять метод проекции градиента для решения задач оптимизации с ограничениями типа неравенств?
2. Что такое «овражная целевая функция»?
3. Приведите пример такой функции для двумерного случая в виде совокупности линий равного уровня.
4. Какие свойства характеризуют класс NP-полных задач?
5. Какие методы применяется в общих задаче математического программирования для необходимые условия экстремума?

Лекция № 16.
Тема: Постановка задач структурного синтеза.

План

16.1. Процедуры синтеза проектных решений

16.2. Задача принятия решений.

16.3. Представление множества альтернатив.

16.4. Морфологические таблицы.

16.5. Альтернативные графы.

16.6. Планирование процессов и распределение ресурсов.

Ключевые слова: синтез, лингивистические переменные, фрейм, искусственный интеллект, идентификация, аппроксимация, морфологическая таблица, алтернативные графы, вершина, инцидентние ветвы, алгоритмизация.

16.1. Процедуры синтеза проектных решений

Принятие проектных решений охватывает широкий круг задач и процедур - от выбора вариантов в конечных и обозримых множествах до задач творческого характера, не имеющих формальных способов решения.

Соответственно в САПР применяют как средства формального синтеза проектных решений, выполняемого в автоматическом режиме, так и вспомогательные средства, способствующие выполнению синтеза проектных решений в интерактивном режиме. К вспомогательным средствам относятся базы типовых проектных решений, системы обучения проектированию, программно методические комплексы верификации проектных решений, унифицированные языки описания ТЗ и результатов проектирования.

Задачи синтеза структур проектируемых объектов относятся к наиболее трудно формализуемым. Существует ряд общих подходов к постановке этих задач, однако практическая реализация большинства из них не очевидна. Поэтому имеются лишь «островки» автоматического выполнения процедур синтеза среди «моря» проблем, ждущих автоматизации.

Именно по этой причине структурный синтез, как правило, выполняют в интерактивном режиме при решающей роли инженера-разработчика, а ЭВМ играет вспомогательную роль: предоставление необходимых справочных данных, фиксация и оценка промежуточных и окончательных результатов.

Однако имеются и примеры успешной автоматизации структурного синтеза в ряде приложений. Среди них заслуживают упоминания в первую очередь задачи конструкторского проектирования печатных плат и кристаллов БИС, логического синтеза комбинационных схем цифровой автоматики и вычислительной техники, синтеза технологических процессов и управляющих программ для механообработки в машиностроении и некоторые другие [4,5].

Структурный синтез заключается в преобразовании описаний проектируемого объекта: исходное описание содержит информацию о требованиях к свойствам объекта, об условиях его функционирования, ограничениях на элементный состав и т.п., а результирующее описание должно содержать сведения о структуре, т.е. о составе элементов и способах их соединения и взаимодействия.

Постановки и методы решения задач структурного синтеза в связи с трудностями формализации не достигли степени обобщения и детализации, свойственной математическому обеспечению процедур анализа. Достигнутая степень обобщения выражается в установлении типичной последовательности действий и используемых видов описаний при их преобразованиях в САПР. Исходное описание, как правило, представляет собой ТЗ на проектирование, по нему составляют описание на некотором формальном языке, являющемся входным языком используемых подсистем САПР. Затем выполняют преобразования описаний и получаемое итоговое для данного этапа описание документируют - представляют в виде твердой копии или файла в соответствующем формате для передачи на следующий этап.

Важное значение для развития подсистем синтеза в САПР имеют разработка и унификация языков представления описаний (спецификаций). Каждый язык, поддерживая выбранную методику принятия решений, формирует у пользователей САПР - разработчиков технических объектов - определенный стиль мышления; особенности языков непосредственно влияют на особенности правил преобразования спецификаций. Примерами унифицированных языков описания проектных решений являются язык VHDL для радиоэлектроники, сочетающий в себе средства для функциональных, поведенческих и структурных описаний, и язык Express - универсальный язык спецификаций для представления и обмена информацией в компьютерных средах.

16.2. Задача принятия решений

Имеется ряд подходов для обобщенного описания задач принятия проектных решений в процессе структурного синтеза.

Задачу принятия решений формулируют следующим образом:

$$ЗПР = < \mathbf{A}, \mathbf{K}, Мод, П >,$$

где **А** - множество альтернатив проектного решения; **К** = $(K_1\ K_2,\ ,.,\ K_m)$ - множество критериев (выходных параметров), по которым оценивается соответствие альтернативы поставленным целям; Мод: **А→К** - модель, позволяющая для каждой альтернативы рассчитать вектор критериев; П - решающее правило для выбора наиболее подходящей альтернативы в многокритериальной ситуации.

В свою очередь, каждой альтернативе конкретного приложения можно поставить в соответствие значения упорядоченного множества (набора) атрибутов **X**=< $x_1, x_2, \ldots, x_n$ >, характеризующих свойства альтернативы. При этом x_i может быть величиной типа *real, integer, Boolean, string* (в последнем случае величину называют предметной или лингвистической). Множество X называют записью (в теории баз данных), *фреймом* (в искусственном интеллекте) или *хромосомой* (в генетических алгоритмах). Модель Мод называют структурно-критериальной, если среди x имеются параметры, характеризующие структуру моделируемого объекта.

Основными проблемами в ЗПР являются:

- компактное представление множества вариантов (альтернатив);
- построение модели синтезируемого устройства, в том числе выбор степени абстрагирования для оценки значений критериев;
- формулировка предпочтений в многокритериальных ситуациях (т. е. преобразование векторного критерия **К** в скалярную целевую функцию);
- установление порядка (предпочтений) между альтернативами в отсутствие количественной оценки целевой функции (что обычно является следствием неколичественного характера всех или части критериев);
- выбор метода поиска оптимального варианта (сокращение перебора вариантов).

Присущая проектным задачам неопределенность и нечеткость исходных данных, а иногда и моделей, диктуют использование специальных методов количественной формулировки исходных неколичественных данных и отношений. Эти специальные методы либо относятся к области построения измерительных шкал, либо являются предметом теории нечетких множеств.

Измерительные шкалы могут быть:

1) абсолютными;

2) номинальными (классификационными), значения шкалы представляют классы эквивалентности, примером может служить шкала цветов; такие шкалы соответствуют величинам неколичественного

характера;

3) порядковыми, если между объектами ***A*** и ***B*** установлено одно из следующих отношений: простого порядка, гласящее, что если ***A*** лучше ***B,*** то ***B*** хуже ***A*** и соблюдается транзитивность; или слабого порядка, т.е. либо ***A*** не хуже ***B,*** либо ***A*** не лучше ***B;*** или частичного порядка. Для формирования целевой функции ***F(X)*** производится оцифровка порядковой шкалы, т.е. если при минимизации ***A*** предпочтительнее ***B,*** то $F(X_a<F(X_b)$, где $\mathbf{X_a}$ и X_b - множества атрибутов объектов ***A*** и ***B*** соответственно;

4) интервальными, отражающими количественные отношения интервалов: шкала единственна с точностью до линейных преобразований, т.е. $y - ax + b$, $a > 0$, $-\infty < b < \infty$, или $y = ax$ при $a \neq 0$, или $y = x + b$.

В большинстве случаев структурного синтеза математическая модель в виде алгоритма, позволяющего по заданному множеству **X** и заданной структуре объекта рассчитать вектор критериев *K,* оказывается известной. Например, такие модели получаются автоматически в программах анализа типа *Spice, Adams* или ПА-9 для объектов, исследуемых на макроуровне. Однако в ряде других случаев такие модели не известны в силу недостаточной изученности процессов и их взаимосвязей в исследуемой среде, но известна совокупность результатов наблюдений или экспериментальных исследований. Тогда для получения моделей используют специальные методы идентификации и аппроксимации (модели, полученные подобным путем, иногда называют феноменологическими).

Среди методов формирования моделей по экспериментальным данным наиболее известны методы планирования экспериментов. Не менее популярным становится подход, основанный на использовании искусственных нейронных сетей.

Если же математическая модель **X** → **K** остается неизвестной, то стараются использовать подход на базе *систем искусственного интеллекта (экспертных систем).*

Возможности практического решения задач дискретного математического программирования (ДМП) изучаются в теории сложности задач выбора, где показано, что задачи даже умеренного размера, относящиеся к классу NP-полных задач, в общем случае удается решать только приближенно.

Поэтому большинство практических задач структурного синтеза решают с помощью приближенных (эвристических) методов. Это методы, использующие специфические особенности того или иного класса задач и не гарантирующие получения оптимального решения. Часто они приводят к

результатам, близким к оптимальным, при приемлемых затратах вычислительных ресурсов.

Если все управляемые параметры альтернатив, обозначаемые в виде множества **X**, являются количественными оценками, то используют приближенные методы оптимизации. Если в **X** входят также параметры неколичественного характера и пространство **X** неметризуемо, то перспективными являются эволюционные методы вычислений, среди которых наиболее развиты генетические методы. Наконец, в отсутствие, обоснованных моделей Мод их создают, основываясь на экспертных знаниях в виде некоторой системы искусственного интеллекта.

16.3. Представление множества альтернатив

Решению проблем упорядочения и описания множества альтернатив и связей между ними в конкретных приложениях посвящена специальная область знания, которую по аналогии с наукой описания множеств животных и растений в биологии можно назвать систематикой.

Простейший способ задания множества **A** - явное перечисление всех альтернатив. Семантика и форма описания альтернатив существенно зависят от приложения. Для представления таких описаний в памяти ЭВМ и доступа к ним используют информационно-поисковые системы (ИПС). Каждой альтернативе в ИПС соответствует поисковый образ, состоящий из значений атрибутов x_t и ключевых слов вербальных характеристик.

Явное перечисление альтернатив при представлении множества альтернатив возможно лишь при малой мощности **A**. В большинстве случаев используют неявное описание **A** в виде способа (алгоритма или набора правил Р) синтеза проектных решений из ограниченного набора элементов Э. Поэтому здесь **A**= < **Р**, **Э** >, а типичный процесс синтеза проектных решений состоит из следующих этапов:

1) формирование альтернативы (это может быть выбор из базы данных ИПС по сформированному поисковому предписанию или генерация из **Э** в соответствии с правилами **Р**);

2) оценка альтернативы по результатам моделирования с помощью модели Мод;

3) принятие решения относительно перехода к следующей альтернативе или прекращения поиска (выполняется лицом, принимающим решение (ЛПР), или автоматически).

Для описания множеств **Р** и **Э** используют следующие подходы:

- морфологические таблицы и альтернативные И- ИЛИ - деревья;
- представление знаний в интеллектуальных системах - фреймы, семантические сети, продукции; генетические методы;

- базы физических эффектов и эвристических приемов, применяемые при решении задач изобретательского характера.

16.4. Морфологические таблицы

Морфологическая таблица (М) представляет собой обобщенную структуру в виде множества функций, выполняемых компонентами синтезируемых объектов рассматриваемого класса, и подмножеств способов их реализации. Каждой функции можно поставить в соответствие одну строку таблицы, каждому способу ее реализации - одну клетку в этой строке. Следовательно, в морфологических таблицах элемент M_{ij} означает j-й вариант реализации i-й функции в классе технических объектов, описываемом матрицей M.

Другими словами, множество альтернатив можно представить в виде отношения M, называемого морфологической таблицей:

$$\mathbf{M = < X, R >,}$$

где **X** - множество свойств (характеристик или функций), присущих объектам рассматриваемого типа; n - число этих свойств; $\mathbf{R = < R_1, R_2, ..., R_n >}$; **R** - множество значений (способов реализации) i -го свойства, мощность этого множества далее обозначена N_t. При этом собственно множество альтернатив **A** представлено композицией множеств **R**, т. е. каждая альтернатива включает по одному элементу (значению) из каждой строки морфологической таблицы. Очевидно, что общее число альтернатив $\boldsymbol{\kappa}$, представляемых морфологической таблицей, равно

$$k = \prod_{i=1}^{n} N_i.$$

Морфологические таблицы обычно считают средством неавтоматизированного синтеза, помогающим человеку просматривать компактно представленные альтернативы, преодолевать психологическую инерцию. Последнее связано с тем, что внимание ЛПР обращается на варианты, которые без морфологической таблицы оставались бы вне его поля зрения.

Собственно таблица **M** не содержит сведений о способе синтеза. Однако на базе **M** возможно построение методов синтеза с элементами алгоритмизации. В таких методах вводится метризация морфологического пространства. Морфологическое пространство составляют возможные законченные структуры, принимается, что расстояние между структурами С, и $\mathbf{C_2}$ есть число несовпадающих элементов (каждая клетка таблицы **M** есть один элемент). Поэтому можно говорить об окрестностях решений. Далее исходят из предположения о компактности «хороших» решений, которое

позволяет вместо полного перебора ограничиваться перебором в малой окрестности текущей точки поиска. Таким образом, гипотеза о компактности и метризация пространства решений фактически приводят к построению математической модели, к которой можно применить методы дискретной оптимизации, например локальные методы.

К недостаткам таблицы **M** относятся неучет запрещенных сочетаний элементов в законченных структурах и отражение состава элементов в структурах без конкретизации их связей. Кроме того, морфологические таблицы строят в предположении, что множества R взаимно независимы, т. е. состав способов реализации *i*-й функции не меняется при изменении значений других функций. Очевидно, что предположение о взаимной независимости множеств **R** оправдано лишь в сравнительно простых структурах. Последний недостаток устраняется путем обобщения метода морфологических таблиц - при использовании метода альтернативных (И-ИЛИ) графов.

16.5. Альтернативные графы

Любую морфологическую таблицу можно представить в виде дерева (рис.16.1). На рисунке функции показаны ребрами, идущими вниз из вершины **M** (вершина И), значения функций - множество ребер, идущих вниз из вершин ИЛИ (светлые кружки). Очевидно, что таблица представляет собой множество однотипных объектов, поскольку все они характеризуются одним и тем же множеством функций.

Для разнотипных объектов применяют многоярусные *альтернативные графы*. Например, на рис.16.2 показан двухъярусный граф, в котором для разных типов объектов предусмотрены разные подмножества функций.

Если допустить некоторую избыточность при изображении И-ИЛИ-графа, то его можно превратить в И-ИЛИ-дерево, что ведет к определенным удобствам.

Очевидно, что И-ИЛИ-дерево можно представить как совокупность морфологических таблиц. Каждая И-вершина дерева соответствует частной морфологической таблице, т.е. множеству функций так, что *i*-я выходящая ветвь отображает *i-ю* функцию. Каждая ИЛИ - вершина, инцидентная *i*-й ветви, соответствует множеству вариантов реализации *i*-й функции, при этом *j-я* исходящая из ИЛИ-вершины ветвь отображает *j-й* вариант реализации.

Алгоритмизация синтеза на базе И-ИЛИ-деревьев требует введения правил выбора альтернатив в каждой вершине ИЛИ. Эти правила чаще всего имеют эвристический характер, связаны с требованиями ТЗ, могут отражать запреты на сочетания определенных компонентов структур.

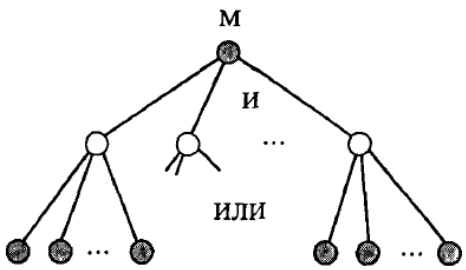


Рис. 16.1. Дерево, соответствующее морфологической таблице

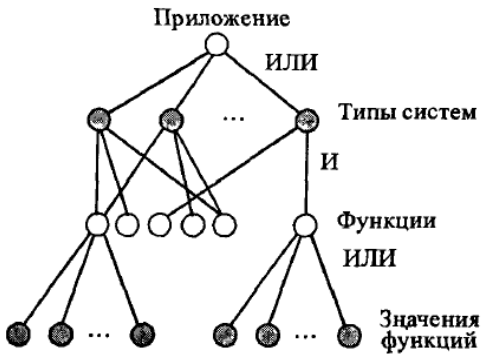


Рис. 16.2. И-ИЛИ-граф

Трудности эффективного решения задачи существенно возрастают при наличии ограничений, типичными среди которых являются ограничения на совместимость способов реализации разных функций, т.е. ограничения вида

$$C_{ij} \textbf{ and } C_{pq} = \text{false}, \tag{16.1}$$

где $C_{ij} = true$, если в оцениваемый вариант вошел элемент $Э_{ij}$, иначе $C_{ij} = false$. Условие (16.1) означает, что в допустимую структуру не могут входить одновременно элементы $Э_{ij}$ и $Э_{pq}$. Совокупность ограничений типа (16.1) можно представить как систему логических уравнений с неизвестными С. Тогда задачу синтеза можно решать эволюционными методами, если предварительно или одновременно с ней решать систему логических уравнений (задачу о выполнимости).

Исчисления. Очевидно, что в большинстве случаев структурного синтеза вместо нереализуемого явного представления всего множества проектных решений задают множество элементов и совокупность правил объединения этих элементов в допустимые структуры (проектные решения).

Эти множества элементов и правил часто представляют в виде формальной системы (исчисления), т. е. задача синтеза имеет вид

$$\textbf{ЗС} = \textbf{< Э; НТ; АК; П >},$$

где **Э** - алфавит исчисления (алфавит представлен базовыми элементами, из которых синтезируется структура); **НТ** - множество букв, не совпадающих с буквами алфавита **Э** и служащих для обозначения переменных; **АК** - множество аксиом исчисления, под которыми понимаются задаваемые исходные формулы (слова) в алфавите Э (например, соответствия функций и элементов); **П** - множество правил вывода новых формул в алфавите **Э** из аксиом и ранее выведенных корректных формул. Каждую формулу можно

интерпретировать как некоторую структуру, поэтому синтез - это процесс вывода формулы, удовлетворяющей исходным требованиям и ограничениям.

Другие примеры компактного задания множества альтернатив **А** через множества **Э** и **П** связаны с использованием систем искусственного интеллекта, в которых **Э** есть база данных, **П** - база знаний, или эволюционных методов, в которых **Э** - также база данных, **П** - множество эвристик, последовательность их применения определяется эволюционными и генетическими принципами.

16.6. Планирование процессов и распределение ресурсов

В отдельную группу задач структурного синтеза выделяют планирование процессов и распределение ресурсов. В нее входят синтез технологических процессов в различных отраслях промышленности, проектирование вычислительных процессов для многопроцессорных систем и сетей, синтез логистических процессов (например, планирование перевозок грузов при наличии множества заказов и ограниченном числе транспортных средств), а также планирование работ при управлении проектами. Эти задачи объединяет общность ряда свойств и подходов к решению, как задач синтеза расписаний.

Базовым понятием в синтезе расписаний является понятие работы - элементарной планируемой части процесса. Нужно составить план выполнения работ, в котором фиксируются объемы работ, распределение ресурсов всех видов, моменты (даты) начала и окончания каждой работы, называемые событиями (или вехами), стоимости работ. Ресурсы - обеспечивающие компоненты деятельности, включающие исполнителей, энергию, материалы, оборудование и т.д.

С каждой работой можно связать функцию потребности в ресурсах. Различают ресурсы унарные и объемные. Единица унарного ресурса, называемая далее сервером, может одновременно выполнять не более одной работы, и по каждому виду ресурса на работу может быть назначен не более чем один сервер. Примерами унарных ресурсов могут быть токарный станок, процессор ЭВМ, водитель автомашины. Значение объемного ресурса (энергии, финансов, пропускной способности канала), назначаемое для конкретной работы, может быть выбрано в некотором диапазоне и от выбранного значения зависят длительность и (или) стоимость выполнения работы.

Результаты синтеза обычно представляют в виде таблиц и диаграмм. PERT- диаграмма (сеть типа «вершина - событие») - ориентированный граф без контуров, имеющий одну исходную и одну завершающую вершины, в котором вершины поставлены в соответствие событиям, а дуги - работам.

Диаграмма Ганта (Gantt diagram) - горизонтальная линейная диаграмма, на которой задачи проекта представляются протяженными во времени отрезками, распределенными между серверами и характеризующимися датами начала и окончания, задержками и, возможно, другими временными параметрами.

Для задач планирования процессов и управления проектами характерны следующие особенности:

1)широкий диапазон размеров задач, причем верхняя граница диапазона может достигать значений в десятки тысяч и более работ;

2)многокритериальность, основные критерии - время и стоимость выполнения плана, в качестве целевой функции часто выбирают стоимость, в число ограничений включают времена окончания работ и, возможно, ряд условий использования ресурсов;

3)разнообразие типов управляемых переменных, среди которых могут быть величины действительные, целые, нечисловые.

Указанные особенности обусловливают сложность решения задач синтеза расписаний, являющихся задачами дискретной оптимизации.

Контрольные вопросы

1. Какие свойства характеризуют класс NP-полных задач?
2. Морфологическая таблица содержит 8 строк и 24 столбца. Сколько различных вариантов структуры представляет данная таблица?
3. Приведите пример И-ИЛИ-графа для некоторого знакомого вам приложения.
4. Приведите примеры продукций из знакомого вам приложения.
5. Дайте предложения по постановке задачи компоновки модулей в блоки для ее решения генетическими методами. Какова структура хромосомы?

Лекция № 17.
Тема: Методы структурного синтеза в САПР. Внутренние и внешние связи сооружение автоматизации.

План

17.1. Метод ветвей и границ.

17.2. Элементы теории сложности.

17.3. Методы локальной оптимизации и поиска с запретами.

17.4. Методы распространения ограничений.

Ключевые слова: ветвы, границы, целовая функция, локальная опитимизация, фрейм, семантическая сеть, экспертная система.

17.1. Метод ветвей и границ

Выбор метода поиска решения - вторая проблема после формализации задачи. Если при формализации все управляемые параметры удалось представить в числовом виде, то можно попытаться применить известные методы ДМП.

Задача ДМП определяется следующим образом:

$$\underset{X \in D}{\mathbf{extr}}\ F(\mathbf{X}),$$

$$\mathbf{D} = \{\ \mathbf{X} \mid \mathbf{W(X)} > 0, \mathbf{Z(X)} = 0\ \}, \qquad (17.1)$$

где ***F(X)*** - целевая функция; ***W(X), Z (X)*** - вектор-функции, связанные с представленными в ТЗ требованиями и ограничениями; **D** - дискретное множество. В полученной модели, во-первых, каждый элемент множества рассматриваемых законченных структур должен иметь уникальное сочетание значений некоторого множества числовых параметров, вектор которых обозначим **X.** Во-вторых, необходимо существование одной или нескольких функций **Ф(X)**, значения которых могут служить исчерпывающей оценкой соответствия структуры предъявляемым требованиям. В-третьих, функции **Ф(X)** должны отражать внутренне присущие данному классу объектов свойства, что обеспечит возможность использования **Ф(X)** в качестве не только средств оценки достигнутого при поиске успеха, но и средств указания перспективных направлений продолжения поиска. Эти условия выполнимы далеко не всегда, что и обусловливает трудности формализации задач структурного синтеза.

Однако наличие формулировки (17.1) еще не означает, что удастся подобрать метод (алгоритм) решения задачи (17.1) с приемлемыми затратами вычислительных ресурсов. Другими словами, применение точных методов математического программирования вызывает непреодолимые трудности в большинстве случаев практических задач типичного размера из-за их принадлежности к классу NP-трудных задач. Поэтому лидирующее положение среди методов решения задачи (17.1) занимают приближенные методы, в частности декомпозиционные методы, отражающие принципы блочно-иерархического проектирования сложных объектов. Декомпозиционные методы основаны на выделении ряда иерархических уровней, на каждом из которых решаются задачи приемлемого размера.

Основу большой группы математических методов, выражающих стремление к сокращению перебора, составляют операции разделения множества вариантов на подмножества и отсечения неперспективных подмножеств. Эти методы объединяются под названием *метода ветвей и границ.* Основная разновидность метода ветвей и границ относится к точным методам решения комбинаторных задач. Рассмотрим эту разновидность.

Пусть имеется множество решений ***M***, в котором нужно выбрать оптимальный по критерию $\mathbf{F(X_j)}$ вариант, где - вектор параметров варианта

$m_j \in M$; пусть также имеется алгоритм для вычисления нижней границы $L(M_k)$ критерия $F(X)$ в любом подмножестве M_k множества M, т.е. такого $L(M_k)$ значения, что $F(X_j) \geq L(M_k)$ при любом j (подразумевается минимизация $F(X)$). Тогда основная схема решения задач в соответствии с методом ветвей и границ содержит следующие процедуры: 1) в качестве M_k принимаем все множество M; 2) ветвление: разбиение M_k на несколько подмножеств M_q; 3) вычисление нижних границ $L(M_q)$ в подмножествах M_q; 4) выбор в качестве M_k подмножества M_p с минимальным значением нижней границы критерия (среди всех подмножеств, имеющихся на данном этапе вычислений), сведения об остальных подмножествах M_q и их нижних границах сохраняются в отдельном списке; 5) если $|m_k| > 1$, то переход к процедуре 2, иначе одноэлементное множество M_k есть решение.

Метод ветвей и границ в случае точного вычисления нижних границ относится к точным методам решения задач выбора и потому в неблагоприятных ситуациях может приводить к экспоненциальной временной сложности. Однако метод часто используют как приближенный, поскольку можно применять приближенные алгоритмы вычисления нижних границ.

Среди других приближенных методов решения задачи ДМП отметим *метод локальной оптимизации.* Так как пространство D метризовано, то можно использовать понятие α-окрестности $S_\alpha X_k$ текущей точки поиска $X_к$. Вместо перебора точек во всем пространстве D осуществляется перебор точек только в $S_\alpha(X_k)$. Если $F(X_j \geq F(X_k)$ для всех $X_j \in S_\alpha(X_k)$, то считается, что найден локальный минимум целевой функции в точке $X_к$. В противном случае точку X_q, в которой достигается минимум $F(X)$ в $S_\alpha(X_k)$, принимают в качестве новой текущей точки поиска.

17.2. Элементы теории сложности.

В теории сложности выделяют массовые и индивидуальные задачи. Первые из них сформулированы в общем виде, вторые представлены с конкретными числовыми значениями исходных данных. Исследования сложности проводятся в отношении массовых задач, и получаемые выводы, как правило, относятся к наихудшему случаю - к наиболее неблагоприятному возможному сочетанию исходных данных.

Цель исследований - установление вида зависимости объема Q требуемых вычислений от размера задачи N. Объем вычислений может определяться числом арифметических и логических операций или затратами процессорного времени ЭВМ с заданной производительностью. Размер задачи в общем случае связывают с объемом описания задачи, но в приложениях понятие размера легко наполняется более конкретным содержанием.

Таблица 17.1.

$Q(N)$	Б$_1$	Б$_2$ = 100 Б$_1$	Б$_3$ = 1000 Б$_1$
N	N_1	100 N_1	1000 N_1
N^2	N_2	10 N_2	31,6 N_2
N^3	N_3	4,64 N_3	10 N_3
2^N	N_4	6,64 + N_4	9,97 + N_4

Далее, в теории сложности задач выбора вводят понятия эффективных и неэффективных алгоритмов. К *эффективным* относят алгоритмы с полиномиальной зависимостью ***Q*** от ***N,*** например, алгоритмы с функцией ***Q(N)*** линейной, квадратичной, кубической и др. Для *неэффективных* алгоритмов характерна экспоненциальная зависимость ***Q(N)***.

Важность проведения резкой границы между полиномиальными и экспоненциальными алгоритмами вытекает из сопоставления числовых примеров роста допустимого размера задачи с увеличением быстродействия Б используемых ЭВМ (табл.17.1, в которой указаны размеры задач, решаемых за одно и то же время T на ЭВМ с быстродействием Б; при различных зависимостях сложности ***Q*** от размера ***N)***. Эти примеры показывают, что выбирая ЭВМ в K раз более быстродействующую, получаем увеличение размера решаемых задач при линейных алгоритмах в $K^{1/2}$ раз, при квадратичных алгоритмах в K раз и т.д.

Иначе обстоит дело с неэффективными алгоритмами. Так, в случае сложности 2^N для одного и того же процессорного времени размер задачи увеличивается только на ***lgK/lg2*** единиц. Следовательно, переходя от ЭВМ с Б=1 Гфлопс к суперЭВМ с Б=1 Тфлопс, можно увеличить размер решаемой задачи только на 10, что совершенно недостаточно для практических задач. Действительно, в таких задачах, как, например, синтез тестов для БИС, число входных двоичных переменных может составлять более 100 и поэтому полный перебор всех возможных проверяющих кодов потребует выполнения более 2^{100} вариантов моделирования схемы.

В теории сложности все комбинаторные задачи разделены на классы:

- класс неразрешимых задач, в него входят массовые задачи, решение которых полным перебором принципиально невозможно с точки зрения современных научных представлений; этот класс отделяется от других задач так называемым пределом Бреммермана, оцениваемым величиной $N = 10^{93}$; отметим, что реальный предел неразрешимости значительно ниже;
- класс **P**, к нему относятся задачи, для которых известны алгоритмы решения полиномиальной сложности;
- класс **NP**, включающий задачи, для которых можно за

полиномиальное время проверить правильность решения, т.е. ответить на вопрос, удовлетворяет ли данное решение заданным условиям; очевидно, что **P** включено в **NP**, однако вопрос о совпадении этих классов пока остается открытым, хотя, по-видимому, на этот вопрос будет получен отрицательный ответ;

• класс NP -полных задач, характеризующийся следующими свойствами:

1) для этих задач не известны полиномиальные алгоритмы точного решения;

2) любые задачи внутри этого класса могут быть сведены одна к другой за полиномиальное время. Последнее означает, что если будет найден полиномиальный алгоритм для точного решения хотя бы одной NP-полной задачи, то за полиномиальное время можно будет решить любую задачу этого класса.

Из результатов теории сложности следуют важные практические рекомендации: 1) приступая к решению некоторой комбинаторной задачи, необходимо сначала проверить, не принадлежит ли она к классу NP-полных задач, и если это так, то не следует тратить усилия на разработку алгоритмов и программ точного решения; 2) отсутствие эффективных алгоритмов точного решения массовой задачи выбора отнюдь не означает невозможности эффективного решения индивидуальных задач из класса NP-полных или невозможности получения приближенного решения по эвристическим алгоритмам за полиномиальное время.

17.3. Методы локальной оптимизации и поиска с запретами

Среди приближенных методов решения задачи (17.1) часто используемым является метод локальной оптимизации. Так как пространство D метризовано, то можно использовать понятие α-окрестности $S_\alpha(X_k)$ текущей точки поиска $X_к$. Вместо перебора точек во всем пространстве D осуществляется перебор только в $S_a(X_k)$. Если $F(X) > F(X_k)$ для всех $X_j \in S_\alpha(X_k)$, то считается, что найден локальный минимум целевой функции в точке $X_к$. В противном случае точку X_q, в которой достигается минимум $F(X)$ в $S_\alpha(X_k)$, принимают в качестве новой текущей точки поиска.

Недостатком метода является его явно выраженная локальность - «застревание» в окрестностях локальных экстремумов. Повысить эффективность поиска можно с помощью метода оптимизации с запретами (tabu search). Для этого в $S_\alpha(X_k)$ вводят запреты на попадание в некоторые точки. Обычно это запреты на повторное исследование точек, пройденных на нескольких последних шагах оптимизации. Запрет распространяется и на лучшую точку X_k предыдущего шага, которая может оказаться точкой локального минимума. Тогда на данном шаге перемещение происходит в лучшую незапрещенную точку X_{k+1}, несмотря на то что $F(X_{k+1}) > F(X_k)$.

Тем самым появляется тенденция к выходу из области притяжения локального экстремума.

Системы искусственного интеллекта. В теории интеллектуальных систем синтез реализуется с помощью экспертных систем

$$\text{ЭС} = <\text{БД, БЗ, И}>,$$

где БД - база данных, включающая сведения о базовых элементах; БЗ - база знаний, содержащая правила конструирования вариантов структуры; И - интерпретатор, устанавливающий последовательность применения правил из базы знаний. Системы искусственного интеллекта основаны на знаниях, отделенных от процедурной части программ и представленных в одной из характерных форм. Такими формами могут быть продукции, фреймы, семантические сети. Реально функционирующие в современных САПР системы с базами знаний чаще всего относятся к классу экспертных систем.

Продукция представляет собой правило типа «если *A,* то *B»,* где *A* - условие, а *B* - действие или следствие, активизируемое при истинности *A.* Продукционная база знаний содержит совокупность правил, описывающих определенную предметную область.

Фрейм - структура данных, в которой в определенном порядке представлены сведения о свойствах описываемого объекта. Типичный вид фрейма:

$$< \text{имя фрейма}; x_1 = p_1; x_2 = p_2; \ldots ; x_N = p_N; q_1, q_2, \ldots, q_M >,$$

где x_i - имя i-го атрибута; p_t - его значение; q_l - ссылка на другой фрейм или некоторую обслуживающую процедуру. В качестве $p_:$ можно использовать имя другого (вложенного) фрейма, описывая тем самым иерархические структуры фреймов.

Семантическая сеть - форма представления знаний в виде совокупности понятий и явно выраженных отношений между ними в некоторой предметной области. Семантическую сеть удобно изображать в виде графа, в котором вершины отображают понятия, а ребра или дуги - отношения между ними. В качестве вершин сети можно использовать фреймы или продукции.

Экспертная система является типичной системой искусственного интеллекта, в которой база знаний содержит сведения, полученные от людей-экспертов в конкретной предметной области. Трудности формализации процедур структурного синтеза привели к популярности применения экспертных систем в САПР, поскольку в них вместо выполнения синтеза на базе формальных математических методов осуществляется синтез на основе опыта и неформальных рекомендаций, полученных от экспертов.

17.4. Методы распространения ограничений

Во многих задачах структурного синтеза множество ***D*** допустимых вариантов, задаваемое ограничениями ***W(X) > 0*** и (или) ***Z(X) = 0***, включает сравнительно малое число элементов, и в качестве результатов синтеза принимается любой из этих вариантов. Такое решение задачи часто выполняют с помощью метода распространения ограничений (constraints propagation).

Сущность этого метода заключается в сужении допустимых интервалов управляемых переменных ***X*** с помощью учета (распространения) исходных ограничений на выходные параметры ***W*** и ***Z.***

Для пояснения метода рассмотрим простой пример. Пусть в задаче фигурируют три управляемые целочисленные переменные *x, y, z,* заданы исходные интервалы допустимых значений этих переменных $x \in [1:100]$, $y \in [1:100]$, $z \in [10:100]$, а область ***D*** определена ограничениями

$$x + y \geq 5z, \quad (17.2)$$

$$x \geq y + 5. \quad (17.3)$$

Распространение ограничения (17.2) на интервал переменного z приводит к уменьшению его верхней границы, так как $z \leq (x_{max} + y_{max})/5 = 40$, а с учетом ограничения (17.3) - к его новой корректировке $z \leq 39$, поскольку $y_{max} = 95$, а также к увеличению нижних границ переменных *x* и *y,* так как решение неравенств $x + y \geq 50$ и (17.3) дает $x \geq 28$, $y \geq 22$. Таким образом, получено сужение допустимой области $x \in [28:100]$, $y \in [22:95]$, $z \in [10:39]$.

Метод легко распространяется на задачи с нечисловыми параметрами. В этом случае вместо сокращения числовых интервалов уменьшают мощности допустимых подмножеств значений параметров.

Одним из практических приложений метода распространения ограничений является поиск допустимых вариантов в множестве синтезируемых структур при ограничениях типа (17.1) на совместимость элементов структуры.

Пример. Рассмотрим фрагмент структуры, состоящий из трех компонентов *A, B и C,* причем $A \in \{a_1, a_2, a_3, a_4, a_5\}$, $B \in \{b_1, b_2, b_3, b_4\}$, $C \in \{c_1, c_2, c_3,\}$ Заданы списки допустимых сочетаний компонентов в синтезируемой структуре:

$$L1: a_1, b_1; a_2, b_1; a_4, b_2; a_5, b_3; a_5, b_4;$$

$$L2: b_1, c_1; b_3, c_3; b_4, c_1; b_4, c_2;$$

$$L3: a_2, c_3; a_3, c_2; a_4, c_3; a_5, c_2.$$

Сокращение первого списка выполняется путем поочередного выбора в нем a, фиксации в ***L3*** соответствующих значений $c_к$, а затем в ***L2*** сопряженных с $c_к$ значений b. Если в ***L1*** имеется элемент a_1, в ***L3***, то он

сохраняется в сокращенном списке, остальные сочетания с а из L1 удаляются. В нашем примере, поскольку значения a_1 в ***L3*** нет, то сочетание a_1, b_1 недопустимо и из ***L1*** удаляется. Далее для символа a_2 фиксируем в ***L3*** значение c_3 ему в ***L2*** соответствует только значение b_3.Поэтому a_2, b_1 - также недопустимое сочетание. Обработав подобным образом все списки, получаем результат распространения ограничений в виде ***L1***: a_5,b_4; L2: b_4,c_2; ***L3***: a_5,c_2.

Следовательно, решение состоит из единственной допустимой структуры, включающей компоненты a_5, b_4, c_3.

В общем случае сокращение списков выполняется в итерационном процессе до совпадения их содержимого на двух последних итерациях.

На базе метода распространения ограничений компанией ILOG создан программный комплекс оптимизации и синтеза проектных решений, состоящий из подсистем Solver, Configurator, Scheduler и др.

Контрольные вопросы

1. Приведите пример такой функции для двумерного случая в виде совокупности линий равного уровня.
2. Какие свойства характеризуют класс NP-полных задач?
3. Морфологическая таблица содержит 8 строк и 24 столбца. Сколько различных вариантов структуры представляет данная таблица?
4. Приведите пример И-ИЛИ-графа для некоторого знакомого вам приложения.
5. Приведите примеры продукций из знакомого вам приложения.

Лекция № 18.
Тема Методики проектирования автоматизированных систем. Особенности проектирования автоматизированных систем.

План

18.1. Особенности проектирования автоматизированных систем.

18.2. Открытые системы.

18.3. Эволюционные методы

18.4. Генетический метод комбинирования эвристик.

Ключевые слова: системная интеграция, спиральный модель, каскадный модель, концептуальное проектирование, эскизный проект, прототип, функциональный модель, информационный модель, поведенческий модель, открытие системы, эволюционный метод, генетический метод.

18.1. Особенности проектирования автоматизированных систем.

К проектированию АС непосредственное отношение имеют два направления деятельности: 1) собственно проектирование АС конкретных предприятий (отраслей) на базе готовых программных и аппаратных

компонентов с помощью специальных инструментальных средств разработки; 2) проектирование упомянутых компонентов АС и инструментальных средств, ориентированных на многократное применение при разработке многих конкретных автоматизированных систем.

Сущность первого направления можно охарактеризовать словами *«системная интеграция»* (другое близкое понятие имеет название *консалтинг).*

Разработчик АС должен быть специалистом в области системотехники, хорошо знать соответствующие международные стандарты, состояние и тенденции развития информационных технологий и программных продуктов, владеть инструментальными средствами разработки приложений (CASE-средствами) и быть готовым к восприятию и анализу автоматизируемых процессов в сотрудничестве со специалистами-прикладниками.

Существует ряд фирм, специализирующихся на разработке проектов АС (например, Price Waterhouse, Jet Info, Consistent Software, Interface и др.)

Второе направление в большей мере относится к области разработки МО и ПО для реализации функций АС - моделей, методов, алгоритмов, программ на базе знания системотехники, методов анализа и синтеза проектных решений, технологий программирования, операционных систем и т. п. Существует ряд общеизвестных технологий (методик) проектирования ПО АС, среди которых прежде всего следует назвать компонентно-ориентированную разработку технологию индустриальной разработки программных систем. Для каждого класса АС (САПР, ERP, геоинформационные системы и т. д.) можно указать фирмы, специализирующиеся на разработке программных (а иногда и программно-аппаратных) систем. Многие из них на основе одной из базовых технологий реализуют свой подход к созданию АС и придерживаются стратегии либо тотального поставщика, либо открытости и расширения системы приложениями и дополнениями третьих фирм.

Как собственно АС, так и компоненты АС являются сложными системами, и при их проектировании нужно использовать один из стилей проектирования:

• *нисходящее (Top-of-Design);* четкая реализация нисходящего проектирования приводит к *спиральной модели* разработки ПО, на каждом витке спирали блоки предыдущего уровня детализируются, используются обратные связи (альтернативой является так называемая *каскадная модель,* относящаяся к поочередной реализации частей системы);

• *восходящее (Bottom-of-Design);*

• *эволюционное (Middle-of-Design).*

Чаще всего применяют нисходящий стиль блочно-иерархического проектирования.

Рассмотрим этапы нисходящего проектирования АС.

Верхний уровень проектирования АС часто называют *концептуальным* проектированием. Концептуальное проектирование выполняют в процессе

предпроектных исследований, формулировки ТЗ, разработки эскизного проекта и прототипирования (согласно ГОСТ 34.601-90, эти стадии называют формированием требований к АС, разработкой концепции АС и эскизным проектом). *Предпроектные исследования* проводят путем анализа (обследования) деятельности предприятия (компании, учреждения, офиса), на котором создается или модернизируется АС. При этом нужно получить ответы на вопросы: что не устраивает в существующей технологии? Что можно улучшить? Кому это нужно и, следовательно, каков будет эффект? Перед обследованием формируются и в процессе его проведения уточняются цели обследования - определение возможностей и ресурсов для повышения эффективности функционирования предприятия на основе автоматизации процессов управления, проектирования, документооборота и т. п. Содержание обследования - выявление структуры предприятия, выполняемых функций, информационных потоков, имеющихся опыта и средств автоматизации. Обследование проводят системные аналитики (системные интеграторы) совместно с представителями организации-заказчика.

На основе анализа результатов обследования строят модель, отражающую деятельность предприятия на данный момент (до реорганизации). Такую модель называют *«As Is» (как есть)*. Далее разрабатывают исходную концепцию АС. Эта концепция включает в себя предложения по изменению структуры предприятия, взаимодействию подразделений, информационным потокам, что выражается в модели «To Be» (как должно быть).

Результаты анализа конкретизируются в ТЗ на создание АС. В ТЗ указывают потоки входной информации, типы выходных документов и предоставляемых услуг, уровень защиты информации, требования к производительности (пропускной способности) и т. п. ТЗ направляют заказчику для обсуждения и окончательного согласования.

Эскизный проект (техническое предложение) представляют в виде проектной документации, описывающей архитектуру системы, структуру ее подсистем, состав модулей. Здесь же содержатся предложения по выбору базовых программно-аппаратных средств, которые должны учитывать прогноз развития предприятия.

В отношении аппаратных средств и особенно ПО такой выбор чаще всего есть выбор фирмы-поставщика необходимых средств (или, по крайней мере, базового ПО), так как правильная совместная работа программ разных фирм достигается с большим трудом. В проекте может быть предложено несколько вариантов выбора. При анализе выясняются возможности покрытия автоматизируемых функций имеющимися программными продуктами и, следовательно, объемы работ по разработке оригинального ПО. Подобный анализ необходим для предварительной оценки временных и материальных затрат на автоматизацию. Учет ресурсных ограничений

позволяет уточнить достижимые масштабы автоматизации, подразделить проектирование АС на работы первой, второй очереди и т. д.

После принятия эскизного проекта разрабатывают *прототип* АС, представляющий собой набор программ, эмулирующих работу готовой системы.

Благодаря прототипированию можно не только разработчикам, но и будущим пользователям АС увидеть контуры и особенности системы и, следовательно, заблаговременно внести коррективы в проект. Как на этапе предпроектных исследований, так и на последующих этапах целесообразно придерживаться определенной дисциплины фиксации и представления получаемых результатов, основанной на той или иной методике формализации спецификаций. Формализация нужна для однозначного понимания исполнителями и заказчиком требований, ограничений и принимаемых решений.

При концептуальном проектировании применяют ряд спецификаций, среди которых центральное место занимают модели преобразования, хранения и передачи информации. Модели, полученные в процессе обследования предприятия, являются моделями его функционирования. В процессе разработки АС модели, как правило, претерпевают существенные изменения (переход от «As Is» к «To Be») и в окончательном виде модель *«To Be»* рассматривают в качестве модели проектируемой АС.

Различают функциональные, информационные, поведенческие и структурные модели. *Функциональная* модель системы описывает совокупность выполняемых системой функций. *Информационная* модель отражает структуры данных - их состав и взаимосвязи. *Поведенческая* модель описываетинформационные процессы (динамику функционирования), в ней фигурируют такие категории, как состояние системы, событие, переход из одного состояния в другое, условия перехода, последовательность событий, осуществляется привязка ко времени. *Структурная* модель характеризует морфологию системы (ее построение) - состав подсистем, их взаимосвязи.

Содержанием последующих этапов нисходящего проектирования (согласно ГОСТ 34.601-90, это стадии разработки технического проекта, рабочей документации, ввода в действие) являются уточнение перечней приобретаемого оборудования и готовых программных продуктов, построение системной среды, детальное инфологическое проектирование баз данных и их первоначальное наполнение, разработка собственного оригинального ПО, которая, в свою очередь, делится на ряд этапов нисходящего проектирования. Эти работы составляют содержание *рабочего проектирования.* После этого следуют закупка и инсталляция программно-аппаратных средств, внедрение и опытная эксплуатация системы.

Особое место в ряду проектных задач занимает разработка проекта корпоративной вычислительной сети, поскольку ТО АС имеет сетевую структуру.

Если территориально АС располагается в одном здании или в нескольких близко расположенных зданиях, то корпоративная сеть может быть выполнена в виде совокупности нескольких локальных подсетей, связанных опорной локальной сетью. Кроме выбора типов подсетей, связных протоколов и коммутационного оборудования приходится решать задачи распределения узлов по подсетям, выделения серверов, выбора сетевого ПО, определения способа управления данными в выбранной схеме распределенных вычислений и т. п.

В случае если АС располагается в удаленных друг от друга пунктах, в частности расположенных в разных городах, то решается вопрос об аренде каналов связи для корпоративной сети, поскольку альтернативный вариант использования выделенного канала в большинстве случаев оказывается неприемлемым вследствие высокой цены. Естественно, что при этом прежде всего рассматривается возможность использования услуг Internet. Возникающие при этом проблемы связаны с обеспечением информационной безопасности и надежности доставки сообщений.

18.2. Открытые системы

Одной из главных тенденций современной индустрии информатики является создание *открытых систем.* Свойство открытости означает, во-первых, переносимость (мобильность) ПО на различные аппаратные платформы, во-вторых, приспособленность системы к ее модификациям (модифицируемость или собственно открытость) и комплексированию с другими системами в целях расширения ее функциональных возможностей и (или) придания системе новых качеств (интегрируемость). Переход к открытым информационным системам позволяет существенно ускорить научно-технический прогресс в результате замены длительной и дорогостоящей разработки новых систем по полному циклу их компоновкой из ранее спроектированных подсистем или быстрой модернизацией уже существующих систем (реинжиниринг).

Открытость подразумевает выделение в системе интерфейсной части (входов и выходов), обеспечивающей сопряжение с другими системами или подсистемами, причем для комплексирования достаточно располагать сведениями только об интерфейсных частях сопрягаемых объектов. Если же интерфейсные части выполнены в соответствии с заранее оговоренными правилами и соглашениями, которых должны придерживаться все создатели открытых систем определенного приложения, то проблема создания новых сложных систем существенно упрощается. Из этого следует, что основой создания открытых систем являются стандартизация и унификация в области информационных технологий.

Значительное развитие концепция открытости получила в области построения вычислительных сетей, что нашло выражение в эталонной модели взаимосвязи открытых систем, поддерживаемой рядом международных стандартов. Идеи открытости широко используются при построении программного, информационного и лингвистического

обеспечений АС; в результате повышается степень универсальности программ и расширяются возможности их адаптации к конкретным условиям.

Аспекты открытости отражены в стандартизации:

• *API {Application Program Interface)* - интерфейсов прикладных программ с операционным окружением, в том числе системных вызовов и утилит операционной системы (ОС), т. е. связей с ОС;

• межпрограммного интерфейса, включая языки программирования;

• сетевого взаимодействия;

• пользовательского интерфейса, в том числе средств графического взаимодействия пользователя с ЭВМ;

• средств защиты информации. Стандарты, обеспечивающие открытость ПО, в настоящее время разрабатываются такими организациями, как ISO (International Standard Organization), IEEE (Institute of Electrical and Electronics Engineers), EIA (Electronics Industries Association) и др.

Стандарты POSIX (Portable Operating System Interface) предназначены для API и составляют группу стандартов ШЕЕ 1003. В этих стандартах содержатся перечень и правила вызова интерфейсных функций, определяются способы взаимодействия прикладных программ с ядром ОС на языке С (что означает преимущественную ориентацию на ОС Unix), даны расширения для взаимодействия с программами на других языках, способы тестирования интерфейсов на соответствие стандартам POSIX, правила административного управления программами и данными и т. п.

Ряд стандартов ISO посвящен языкам программирования. Имеются стандарты на языки С (ISO 9899), Fortran (ISO 153 9), Pascal (ISO 7185) и др.

Среди других стандартов, способствующих открытости ПО АС, следует отметить стандарты графического пользовательского интерфейса, хранения и передачи графических данных, построения баз данных и файловых систем, сопровождения и управления конфигурацией программных систем и др.

Важное значение для создания открытых систем имеют унификация и стандартизация средств межпрограммного интерфейса, или, другими словами, необходимо наличие профилей АС для информационного взаимодействия программ, входящих в АС. *Профилем* открытой системы называют совокупность стандартов и других нормативных документов, обеспечивающих выполнение системой заданных функций.

Так, в профилях АС могут фигурировать язык Express стандарта STEP, спецификация графического пользовательского интерфейса Motif, унифицированный язык SQL обмена данными между различными СУБД, стандарты сетевого взаимодействия, в профили MCAD может входить формат IGES и в случае ECAD - формат EDIF и т. п.

18.3. Эволюционные методы

Эволюционные методы (ЭМ) предназначены для поиска предпочтительных решений и основаны на статистическом подходе к

исследованию ситуаций и итерационном приближении к искомому состоянию систем.

В отличие от точных методов математического программирования ЭМ позволяют находить решения, близкие к оптимальным, за приемлемое время, а в отличие от известных эвристических методов оптимизации характеризуются существенно меньшей зависимостью от особенностей приложения (т. е. более универсальны) и в большинстве случаев обеспечивают лучшую степень приближения к оптимальному решению. Универсальность ЭМ определяется также применимостью к задачам с неметризуемым пространством управляемых переменных (т. е. среди управляемых переменных могут быть и лингвистические).

Важнейшим частным случаем ЭМ являются генетические методы и алгоритмы. *Генетические алгоритмы* основаны на поиске лучших решений с помощью наследования и усиления полезных свойств множества объектов определенного приложения в процессе имитации их эволюции.

Свойства объектов представлены значениями параметров, объединяемыми в запись, называемую в ЭМ *хромосомой.* В ГА оперируют хромосомами, относящимися к множеству объектов - популяции. Имитация генетических принципов - вероятностный выбор родителей среди членов популяции, скрещивание их хромосом, отбор потомков для включения в новые поколения объектов на основе оценки целевой функции - ведет к эволюционному улучшению значений целевой функции (функции полезности) от поколения к поколению.

Среди ЭМ находят применение также методы, которые в отличие от ГА оперируют не множеством хромосом, а единственной хромосомой. Так, метод дискретного локального поиска (его англоязычное название *hillclimbing)* основан на случайном изменении отдельных параметров (т. е. значений полей в записи или, другими словами, значений генов в хромосоме). Такие изменения называют *мутациями.* После очередной мутации оценивают значение *функции полезности F* (fitness function) и результат мутации сохраняется в хромосоме, только если *F* улучшилась. В другом ЭМ под названием «*Моделирование отжига*» (Simulated Annealing) результат мутации сохраняется с некоторой вероятностью, зависящей от полученного значения *F.*

18.4. Генетический метод комбинирования эвристик

Возможны два подхода к формированию хромосом. Первый из них основан на использовании в качестве генов проектных параметров. Например, в задаче размещения микросхем на плате локусы соответствуют посадочным местам на плате, а генами являются номера (имена) микросхем. Другими словами, значением А-го гена будет номер микросхемы в *k-й* позиции.

Во втором подходе генами являются не сами проектные параметры, а номера эвристик, используемых для определения проектных параметров. Так, для задачи размещения можно применять несколько эвристик. По одной

из них, в очередное посадочное место нужно помещать микросхему, имеющую наибольшее число связей с уже размещенными микросхемами, по другой - микросхему с минимальным числом связей с еще не размещенными микросхема-ми и т. д. Генетический поиск в этом случае есть поиск последовательности эвристик, обеспечивающей оптимальный вариант размещения.

Второй подход получил название *метод комбинирования эвристик.* Этот метод оказывается предпочтительным во многих случаях. Например, в задачах синтеза расписаний распределяется заданное множество работ во времени и между обслуживающими устройствами - серверами, т. е. проектными параметрами для каждой работы будут номер сервера и порядковый номер в очереди на обслуживание. Пусть N - число работ, M - число серверов. Если гены соответствуют номерам работ, то в первом подходе в хромосоме нужно иметь $2N$ генов и общее число отличающихся друг от друга хромосом W заметно превышает наибольшее из чисел $N1$ и M^N.

Согласно методу комбинирования эвристик, число генов в хромосоме в 2 раза меньше, чем в первом подходе, и равно N. Поэтому если число используемых эвристик равно K, то мощность множества возможных хромосом уже несравнимо меньше, а именно:

$$W = K^N.$$

Очевидно, что меньший размер хромосомы ведет к лучшей вычислительной эффективности, а меньшее значение W позволяет быстрее найти окрестности искомого экстремума. Кроме того, в методе комбинирования эвристик все хромосомы, генерируемые при кроссовере, будут допустимыми. В то же время при применении обычных генетических методов необходимо использовать процедуры типа PMX для корректировки генов, относящихся к номерам в очереди на обслуживание, что также снижает эффективность поиска.

Контрольные вопросы

1. Дайте формулировку задачи математического программирования.
2. Что такое «множество Парето»?
3. Дайте предложения по постановке задачи компоновки модулей в блоки для ее решения генетическими методами. Какова структура хромосомы?
4. Назовите основные типы промышленных АС и виды их обеспечения.
5. Какие причины привели к появлению и развитию CALS-технологий?
6. Что понимают под комплексной АС?
7. Дайте определение профиля открытой системы.
8. Чем обеспечивается открытость систем?

Лекция № 19.
Тема: Инструментальные средства концептуального проектирования.

План

19.1. Типы CASE-систем

19.2. Спецификации проектов программных систем

19.3. Методики IDEF0 и IDEF3

19.4. Программное обеспечение CASE-систем для концептуального проектирования

Ключевые слова: CASE-система, функциональное проектирование, информационное проектирование, поведенческое моделирование, прототайпер, конвертор, компилятор, репозитория.

19.1. Типы CASE-систем

В современных информационных технологиях важное место отводится инструментальным средствам и средам разработки АС, в частности системам разработки и сопровождения их ПО. Эти технологии и среды образуют системы, называемые *CASE-системами.*

Аббревиатура CASE имеет двоякое толкование, соответствующее двум направлениям использования CASE-систем. Первое из них - *Computer Aided System Engineering* - подчеркивает направленность на поддержку концептуального проектирования сложных систем, преимущественно слабоструктурированных. Далее CASE-системы этого направления будем называть *системами CASE для концептуального проектирования.* Второе направление называют *Computer Aided Software Engineering,* что переводится как автоматизированное проектирование программного обеспечения. Соответствующие CASE-системы называют *инструментальными CASE* или инструментальными средами разработки ПО.

Среди систем CASE для концептуального проектирования различают системы функционального, информационного или поведенческого проектирования. Наиболее известной методикой *функционального проектирования* сложных систем является методика SADT (Structured Analysis and Design Technique), предложенная в 1973 г. Р. Россом и впоследствии ставшая основой стандарта *IDEF0 (Integrated DEFinition 0).*

Системы *информационного проектирования* реализуют методики инфологического проектирования баз данных. Широко используются язык и методика создания информационных моделей приложений, закрепленные в методике IDEF1X. Кроме того, развитые коммерческие СУБД, как правило, имеют в своем составе совокупность CASE-средств проектирования приложений.

Основные положения стандартов IDEF0 и IDEFIX использованы также при создании комплекса стандартов ISO 10303, лежащих в основе

технологии STEP для представления в компьютерных средах информации, относящейся к проектированию и производству в промышленности.

Поведенческое моделирование сложных систем используют для определения динамики функционирования сложных систем. В его основе лежат модели и методы имитационного моделирования систем массового обслуживания, сети Петри, возможно применение конечно-автоматных моделей, описывающих поведение системы как последовательность смены состояний.

Применение инструментальных CASE-систем ведет к сокращению затрат на разработку ПО за счет уменьшения числа итераций и числа ошибок, к улучшению качества продукта вследствие лучшего взаимопонимания разработчика и заказчика, к облегчению сопровождения готового ПО.

Среди инструментальных CASE-систем различают интегрированные комплексы инструментальных средств для автоматизации всех этапов жизненного цикла ПО (такие системы называют *Workbench*) и специализированные инструментальные средства для выполнения отдельных функций (Tools). Средства CASE-систем по своему функциональному назначению принадлежат к одной из следующих групп: 1) средства программирования; 2) средства управления программным проектом; 3) средства верификации (анализа) программ; 4) средства документирования.

К средствам программирования относятся компиляторы с алгоритмических языков; построители диаграмм потоков данных; планировщики для построения высокоуровневых спецификаций и планов ПО (возможно на основе баз знаний, реализованных в экспертных системах); интерпретаторы языков спецификаций и языков четвертого поколения; прототайпер для разработки внешних интерфейсов - экранов, форм выходных документов, сценариев диалога; генераторы программ определенных классов (например, конверторы заданных языков, драйверы устройств программного управления, постпроцессоры); кросс средства; отладчики программ. При этом под языками спецификаций понимают средства укрупненного описания разрабатываемых алгоритмов и программ, к языкам 4GL относят языки для компиляции программ из набора готовых модулей, реализующих типовые функции достаточно общих приложений (чаще всего это функции технико-экономических систем).

Управление программным проектом называют также управлением конфигурациями ПО. Этому понятию соответствуют корректное внесение изменений в программную систему при ее проектировании и сопровождении, контроль целостности проектных данных, управление версиями проекта, организация параллельной работы членов коллектива разработчиков. Использование средств управления конфигурациями позволяет создавать программные системы из сотен и тысяч модулей, значительно сокращать сроки разработки, успешно модернизировать уже поставленные заказчикам системы.

Основой средств управления программным проектом является *репозиторий* - база данных проекта. Именно в репозитории отражена история развития программного проекта, содержатся все созданные версии (исходный программный код, исполняемые программы, библиотеки, сопроводительная документация и т. п.), с помощью репозитория осуществляются контроль и отслеживание вносимых изменений.

Средства верификации служат для оценки эффективности исполнения разрабатываемых программ и определения наличия в них ошибок и противоречий. Различают статические и динамические анализаторы. В статических анализаторах ПО исследуется на наличие неопределенных данных, бесконечных циклов, недопустимых передач управления и т. п. Динамический анализатор функционирует в процессе исполнения проверяемой программы; при этом исследуются трассы, измеряются частоты обращений к модулям и т. п. Используемый математический аппарат - сети Петри, теория массового обслуживания.

В последнюю из перечисленных групп входят документаторы для оформления программной документации, например отчетов по данным репозитория; различные редакторы для объединения, разделения, замены, поиска фрагментов программ и других операций редактирования.

Проектирование ПО с помощью CASE-систем включает в себя несколько этапов. Начальный этап - предварительное изучение проблемы. Результат представляют в виде исходной диаграммы потоков данных и согласуют с заказчиком. На следующем этапе выполняют детализацию ограничений и функций программной системы и полученную логическую модель вновь согласуют с заказчиком. Далее разрабатывают физическую модель, т. е. определяют модульную структуру программы, выполняют инфологическое проектирование баз данных, детализируют граф-схемы программной системы и ее модулей.

Подсистема CASE в составе системной среды САПР предназначена для адаптации САПР к нуждам конкретных пользователей, разработки и сопровождения прикладного ПО. Ее можно рассматривать как специализированную САПР, в которой объектом проектирования являются новые версии подсистем САПР, в частности версии, адаптированные к требованиям конкретного заказчика. Другими словами, такие CASE-подсистемы позволяют пользователям формировать сравнительно с малыми затратами усилий варианты прикладных ПМК из имеющегося базового набора модулей под заданный узкий диапазон конкретных условий проектирования.

CASE-система, как система проектирования ПО, содержит компоненты для разработки структурных схем алгоритмов и «экранов» для взаимодействия с пользователем в интерактивных процедурах, средства для инфологического проектирования баз данных, отладки программ, документирования, сохранения «истории» проектирования и т. п. Наряду с

этим в CASE-подсистему САПР входят и компоненты со специфическими для САПР функциями.

Так, в состав САПР Microstation (фирма Bentley Systems) включена инструментальная среда Microstation Basic и язык MDL (Microstation Development Language) с соответствующей программной поддержкой. Язык MDL - С-подобный, с его помощью можно лаконично выразить обращения к проектным операциям и процедурам. В целом среда Microstation Basic близка по своим функциям к среде MS Visual Basic, в ней имеются генератор форм, редактор, конструктор диалога, отладчик.

САПР Спрут (российская фирма Sprut Technologies), вообще говоря, создана как инструментальная среда для разработки пользователем потоков задач конструкторского и технологического проектирования в машиностроении с последующим возможным оформлением потоков в виде пользовательских версий САПР. Сконструированный поток поддерживается компонентами системы, в число которых входят графические 2D- и 3D-подсистемы, СУБД, продукционная экспертная система, документатор, технологический процессор создания программ для станков с ЧПУ, постпроцессоры.

Наиболее известной CASE-системой в составе САПР в настоящее время является описываемая ниже система CAS.CADE фирмы MatraDatavision, с ее помощью фирма разработала версию Euclid Quantum своей САПР Euclid.

19.2. Спецификации проектов программных систем

Важное значение в процессе разработки ПО имеют средства спецификации проектов ПО. Средства спецификации в значительной мере определяют суть методов CASE.

Способы и средства спецификации классифицируют по базовой методологии, используемой для декомпозиции ПО как сложной системы, и по аспектам моделирования ПО.

Различают два подхода к декомпозиции ПО. Первый способ называют функциональным или структурным. Он основан на выделении функций и потоков данных. Второй способ - объектный, выражает идеи объектно-ориентированного проектирования и программирования.

Аспектами моделирования приложений являются функциональное, поведенческое и информационное описания.

Практически все способы функциональных спецификаций имеют следующие общие черты:

- модель имеет иерархическую структуру, представляемую в виде диаграмм нескольких уровней;
- элементарной частью диаграммы каждого уровня является конструкция вход функция - выход;
- необходимая дополнительная информация содержится в файлах поясняющего текста.

В большинстве случаев функциональные диаграммы являются диаграммами потоков данных *(DFD - Data Flow Diagram).* Блоки (прямоугольники) в DFD соответствуют функциям, дуги - входным и выходным потокам данных. Поясняющий текст представлен в виде «словарей данных», в которых указаны компонентный состав потоков данных, число повторений циклов и т. п. Для описания структуры информационных потоков можно использовать нотацию Баэкуса - Наура.

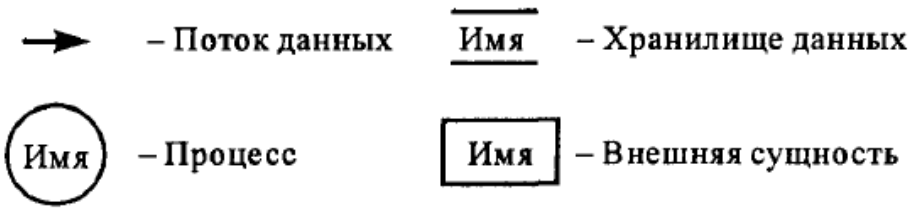


Рис. 19.1. Изображения элементов в нотации Иордана

Одна из нотаций для DFD предложена Е. Йорданом. В ней описывают процессы (функции), потоки данных, хранилища и внешние сущности, их условные обозначения показаны на рис. 19.1.

Разработка DFD начинается с построения диаграммы верхнего уровня, отражающей связи программной системы, представленной в виде единого процесса, с внешней средой. Декомпозиция процесса проводится до уровня, на котором фигурируют элементарные процессы, которые могут быть представлены одностраничными описаниями алгоритмов (мини-спецификациями) на терминальном языке программирования.

Для описания *информационных* моделей наибольшее распространение получили диаграммы сущность - отношение *(ERD - Entity-Relation Diagrams*), в которых предусмотрены средства для описания сущностей, атрибутов и отношений. Спецификации хранилищ данных в CASE, как правило, даются с помощью диаграмм сущность - отношение. Стандартной методикой построения таких диаграмм является IDEF1X.

Поведенческие модели описывают процессы обработки информации. В инструментальных CASE-системах их представляют в виде граф-схем, диаграмм перехода состояний, таблиц решений, псевдокодов (языков спецификаций), процедурных языков программирования, в том числе языков четвертого поколения.

В граф-схемах, как и в диаграммах DFD, блоки используют для задания процессов обработки, но дуги имеют иной смысл - они описывают последовательность передач управления (вместе со специальными блоками управления).

В *диаграммах перехода состояний* узлы соответствуют состояниям моделируемой системы, дуги - переходам из состояния в состояние, атрибуты дуг - условиям перехода и инициируемым при их выполнении действиям. Очевидно, что, как и в других конечно-автоматных моделях, кроме графической формы представления диаграмм перехода состояний можно

использовать также табличные формы. Так, при изоморфном представлении с помощью таблиц перехода состояний каждому переходу соответствует строка таблицы, в которой указываются исходное состояние, условие перехода, инициируемое при этом действие и новое состояние после перехода.

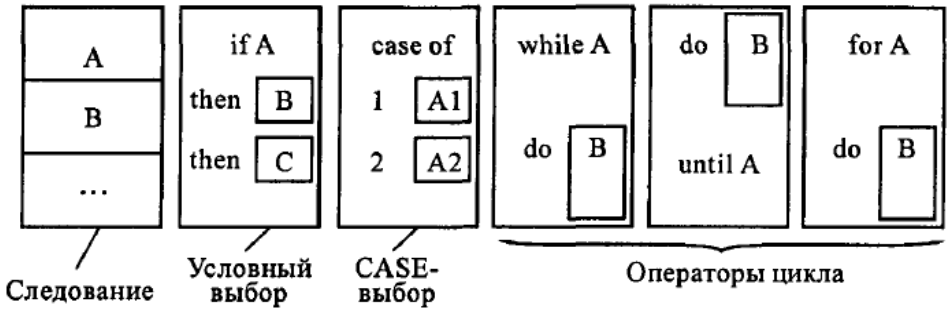


Рис. 19.2. Примеры описания операторов в визуальных языках программирования

Близкий по своему характеру способ описания процессов основан на таблицах (или деревьях) решений. Каждый столбец таблицы решений соответствует определенному сочетанию условий, при выполнении которых осуществляются действия, указанные в нижерасположенных клетках столбца.

Таблицы решений удобны при описании процессов с многократными ветвлениями. В этих случаях помогают также визуальные языки программирования, в которых для описания процессов используют графические элементы, подобные приведенным на рис. 19.2.

В псевдокодах алгоритмы записываются с помощью как средств некоторого языка программирования (преимущественно для управляющих операторов), так и естественного языка (для выражения содержания вычислительных блоков). Используются конструкции (операторы) следования, условные, цикла. Служебные слова из базового языка программирования или из DFD записываются заглавными буквами, фразы естественного языка - строчными.

Языки четвертого поколения предназначены для описания программ как совокупностей заранее разработанных программных модулей. Поэтому одна команда языка четвертого поколения может соответствовать значительному фрагменту программы на языке 3GL. Примерами языков 4GL могут служить Informix-4GL, JAM, NewEra, XAL.

Мини-спецификации процессов могут быть выражены с помощью псевдокодов (языков спецификаций), визуальных языков проектирования или языков программирования.

Объектный подход представлен компонентно-ориентированными технологиями разработки ПО. При объектном подходе ПО формируется из компонентов, объединяющих в себе алгоритмы и данные и взаимодействующих путем обмена сообщениями. Для поддержки объектного

подхода разработан рассматриваемый далее стандартный язык моделирования приложений UML.

19.3. Методики IDEF0 и IDEF3

Взаимосвязанная совокупность методик IDEF для концептуального проектирования разработана по программе Integrated Computer Aided Manufacturing в США. В этой совокупности имеются методики функционального, информационного и поведенческого моделирования и проектирования, в ее состав в настоящее время входит ряд IDEF-методик, часть из которых стандартизирована.

Методики IDEF задают единообразный подход к моделированию приложений, но не затрагивают проблем единообразного представления данных в процессах информационного обмена между разными компьютерными системамии приложениями. Необходимость решения этих проблем в интегрированных АС привела к появлению ряда унифицированных форматов представления данных в межкомпьютерных обменах, среди которых наиболее известными являются форматы IGES, DXF (в машиностроительных приложениях), EDIF (в электронике) и некоторые другие. Ограниченные возможности этих форматов обусловили продолжение работ в направлении создания более совершенных методики представляющих их стандартов. На эту роль в настоящее время претендует совокупность стандартов STEP.

Как отмечено выше, наиболее известной методикой функционального моделирования сложных систем является методика *SADT (Structured Analysis and Design Technique),* положенная в основу стандарта IDEF0.

IDEF0 - это более четко очерченное представление методики SADT. SADT -методика, рекомендуемая для начальных стадий проектирования сложных искусственных систем управления, производства, бизнеса, включающих людей, оборудование, ПО. Начиная с момента создания первой версии методика успешно применялась для проектирования телефонных сетей, систем управления воздушными перевозками, производственных предприятий и др.

Разработку SADT-модели начинают с формулировки вопросов, на которые модель должна давать ответы, т. е. формулируют цель моделирования. Далее строят иерархическую совокупность диаграмм с лаконичным описанием функций.

Недостатки SADT-моделей - слабая формализованное для автоматического выполнения проектных процедур на их основе. Однако наличие графического языка диаграмм, удобного для восприятия человеком, обусловливает полезность и применимость методики SADT.

Описание объектов и процессов в SADT (IDEF0) выполняется в виде совокупности взаимосвязанных блоков (рис. 19.3).

Блоки выражают функции (работы), поэтому их названиями обычно являются глаголы или отглагольные существительные.

Типичные примеры функций: планировать, разработать, классифицировать, измерить, изготовить, отредактировать, рассчитать, продать (или планирование, разработка, классификация, измерение, изготовление, редактирование, расчет, продажа). Число блоков на одном уровне иерархии - не более 6, иначе восприятие диаграмм будет затруднено. Число уровней иерархии не ограничено, но обычно их не более 5. Блоки нумеруются (номер записывается в правом нижнем углу). Дуги (стрелки) отображают множества объектов (данных), их имена - существительные. Управление определяет условия выполнения. Примеры управления: требования, чертеж, стандарт, указания, план. Механизм выражает используемые средства, например: компьютер, оснастка, заказчик, фирма. Входы и выходы могут быть любыми объектами.

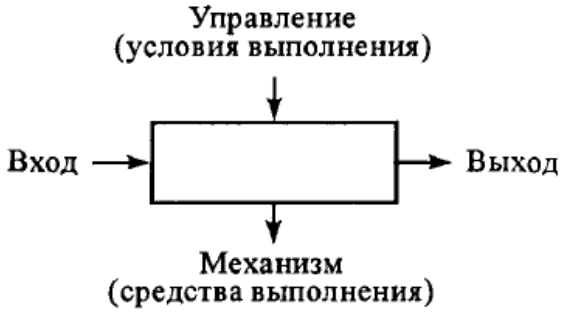


Рис. 19.3. Блок ICOM в IDEF0 диаграммах

Блоки на рис. 19.3 в англоязычной литературе называют блоками *ICOM(Input - Control - Output - Mechanism).*

Рассмотрим пример функциональной модели для процесса создания САПР на предприятии, на котором ранее автоматизация проектирования была развита слабо.

Диаграмма верхнего (нулевого) уровня АО включает единственный блок ICOM «Разработать САПР». В качестве исполнителей фигурируют специализированная организация, занимающаяся проектированием автоматизированных систем и называемая консалтинговой фирмой, а также представители организации-заказчика, объединенные в создаваемый на предприятии отдел САПР.

Диаграмма первого уровня, показанная на рис. 19.4, *а,* включает блоки А1 - обследования предприятия, А2 - проектирования САПР, А3 - реализации САПР и А4 - испытаний системы. Диаграммы следующего второго уровня, раскрывающие первые блоки Al, А2 и А3, представлены на рис. 19.4, б, в и г соответственно (на этих рисунках не отмечены данные, соответствующие внутренним стрелкам диаграмм, а также стрелки условия «финансы»). При обследовании предприятия специалисты консалтинговой фирмы вместе с работниками отдела САПР изучают структуру предприятия, типичные маршруты проектирования, информационные потоки и на этой базе разрабатывают модель «As Is». Далее создается новая модель «To Be» с учетом не только требований автоматизации проектирования, но и будущих

информационных потребностей процессов управления и делопроизводства. Модель «То Be» составляет основу технического предложения на создание САПР.

При проектировании САПР выбирают аппаратно-программную платформу, базовое ПО проектирующих и обслуживающих подсистем, разрабатывают структуру корпоративной сети, определяют типы сетевого оборудования, серверов и рабочих станций, выявляют необходимость разработки оригинальных программных компонентов.

Реализация проекта САПР включает подготовку помещений, монтаж кабельной сети, обучение будущих пользователей САПР, закупку и инсталляцию ТО и ПО.

Разработка SADT-моделей состоит из ряда этапов.

1. Сбор информации. Источниками информации могут быть документы, наблюдение, анкетирование и т. п. Существуют специальные методики выбора экспертов и анкетирования. Создание модели. Используется нисходящий стиль: сначала разрабатываются верхние уровни, затем - нижние.

2. Рецензирование модели. Реализуется в итерационной процедуре рассылки модели на отзыв и ее доработки по замечаниям рецензентов, в завершение собирается согласительное совещание.

Связи функциональной модели, отражающей функции, со структурной моделью, отражающей средства выполнения функций, выражаются с помощью специальных словарей, дающих однозначное толкование вводимых имен ресурсов.

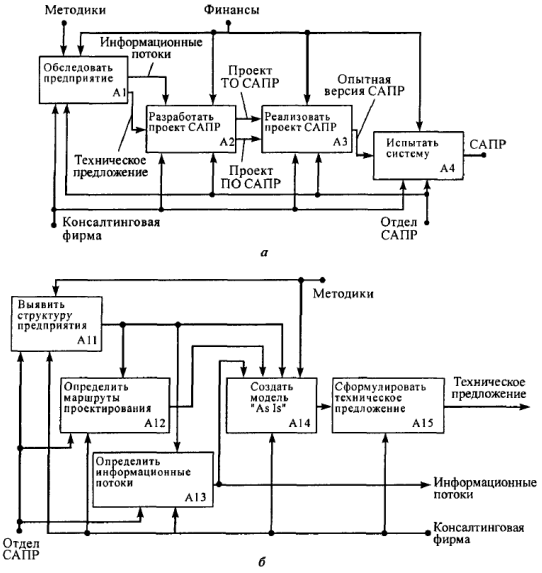


Рис. 19.3. Функциональная модель процесса создания САПР (начало): *а* - IDEFO-диаграмма первого уровня; *б* - IDEFO-диаграмма обследования предприятия; *в* - IDEFO-диаграмма проектирования САПР; *г* - IDEFO-диаграмма реализации проекта САПР

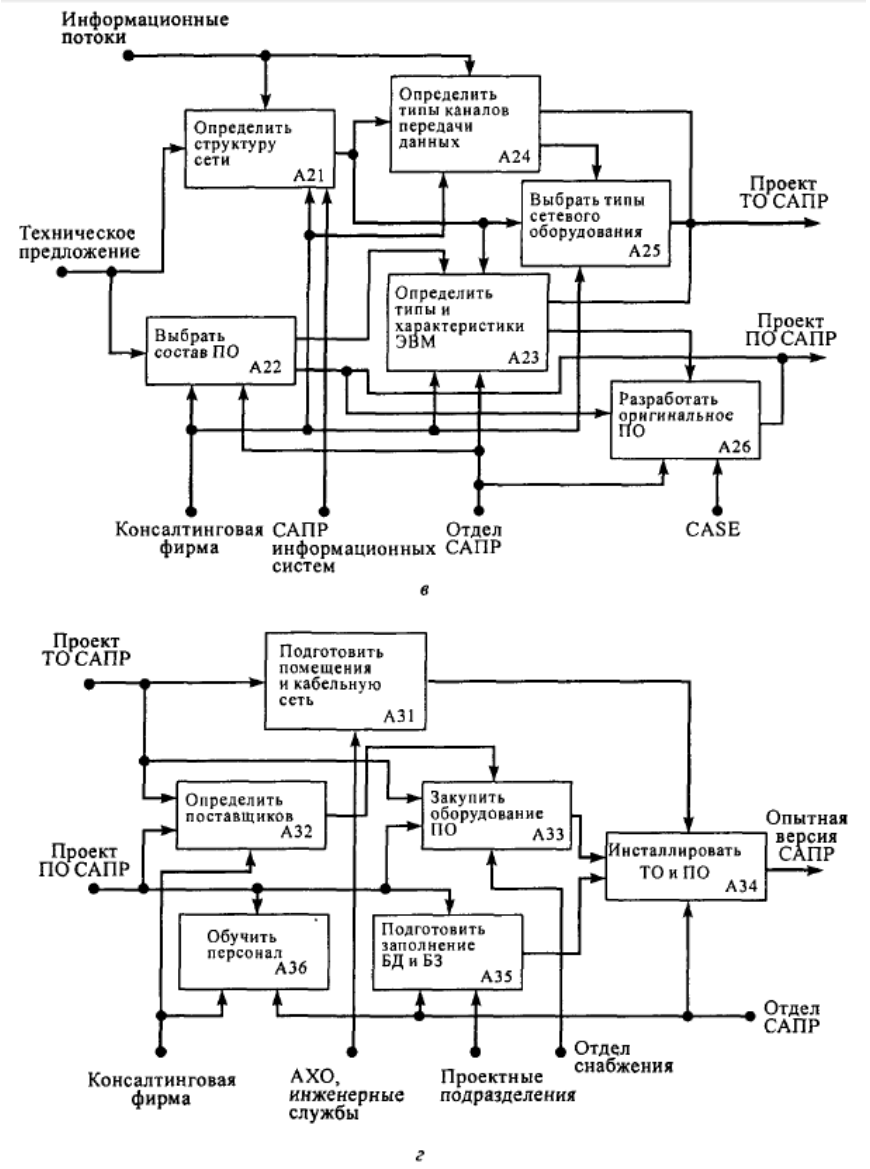


Рис. 19.4. Функциональная модель процесса создания САПР (окончание)

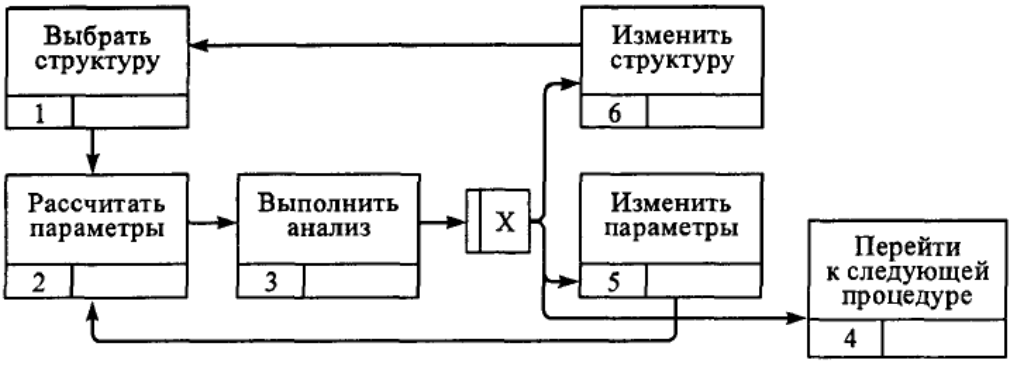


Рис. 19.5. IDEF3-диаграмма последовательности операций

Дальнейшее использование IDEFO-модели - конкретизация задач выбора ресурсов, разработка планов реализации, переход к имитационным моделям и т. п.

Поведенческие аспекты приложений отражает методика IDEF3. Если методика IDEF0 связана с функциональными аспектами и позволяет отвечать на вопросы «Что делает система?», то в IDEF3 детализируются и конкретизируются IDEFO-функции, IDEF3-модель отвечает на вопросы «Как система это делает?» Язык IDEF3 - язык диаграмм, помогающий разработчику моделей наглядно представить моделируемые процессы. В IDEF3 входят два типа описаний:

1) процессно-ориентированные в виде последовательности операций;

2) объектно-ориентированные, выражаемые диаграммами перехода состояний, характерными для конечно-автоматных моделей.

На рис. 19.5 представлен пример процессно-ориентированной IDEF3-диаграммы. Здесь функции (операции) показаны прямоугольниками с горизонтальной чертой, отделяющей верхнюю секцию с названием функции от нижней секции, содержащей номер функции. Связи, отражающие последовательность выполнения функций, изображаются сплошными линиями-стрелками. Для указания разветвлений и слияний связей (их принято называть перекрестками) используют квадраты, у которых одна или обе вертикальные стороны представлены двойными линиями, а внутри квадрата записан один из символов «&», «О» или «Х». При разветвлении эти символы означают реакцию всех, некоторых или только одной из последующих функций на входное воздействие соответственно. Аналогичный смысл имеют символы «&», «О» или «Х» при слиянии - последующая функция начинает выполняться после окончания всех, некоторых или только одной из входных операций.

На рис. 19.6 представлен пример объектно-ориентированной IDEF3-диаграммы. В таких диаграммах имеются средства для изображения состояний системы, активностей, переходов из состояния в состояние и условий перехода.

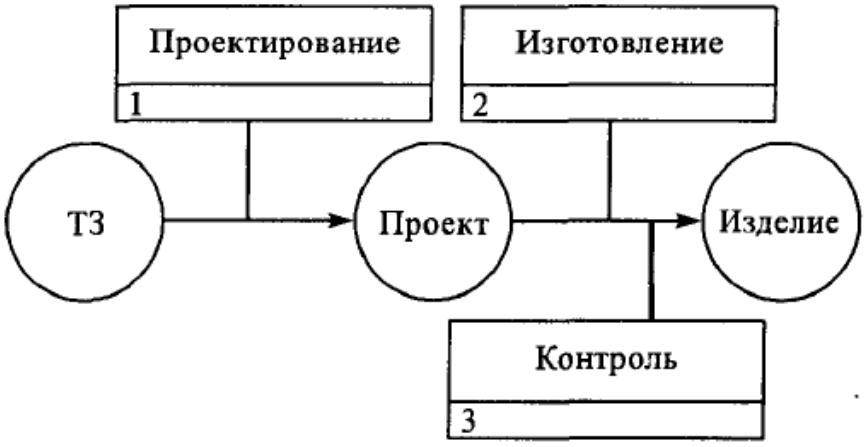


Рис. 19.6. IDEFS-диаграмма перехода состояний

Диаграммы IDEF0 или IDEFS могут быть преобразованы в имитационные модели, если задать дополнительные свойства функций, характеризующие затраты ресурсов. Чаще всего имитационные модели

представляют в виде сетей Петри. Преобразование связано с введением времени в функциональную IDEF0 или в поведенческую IDEF3 -модель с заменой функций переходами, а объектов, отождествляемых со стрелками блоков ICOM, метками в сетях Петри.

19.4. Программное обеспечение CASE-систем для концептуального проектирования

На рынке программных продуктов имеется много CASE-систем для концептуального проектирования АС.

Чаще всего в них поддерживается методология IDEF. В России широко известны программы BPwin, ERwin, OOwin фирмы Platinum Technology, Design/IDEF фирмы Meta Software, CASE - Аналитик фирмы Эйтэкс, Silverrun фирмы CSA и др.

BPwin (Business Processing) служит для разработки функциональных моделей по методике IDEF0.

ERwin предназначена для разработки информационных моделей по методике IDEF 1X. Имеются средства, обеспечивающие интерфейс с серверами баз данных (от пользователя скрыто общение на SQL-языке), перевод графических изображений ER-диаграмм в SQL-формы или в форматы других популярных СУБД. Предусмотрены интерактивные процедуры для связывания дуг IDEF0 с сущностями и атрибутами IDEF IX, т. е. для установления связей между BPwin и ERwin. В систему включены также типичные для CASE средства разработки экранных форм.

OOwin служит для поддержки объектно-ориентированных технологий проектирования информационных систем. Один из способов использования OOwin - детализация объектно-ориентированной модели на базе созданной ER-модели. При преобразовании ER в OO-представление сущности и атрибуты становятся классами (множествами подобных объектов). Классы могут быть дополнены описанием услуг класса, т. е. выполняемых операций, передаваемых и возвращаемых параметров, событий. Другой способ использования OOwin - реинжиниринг, так как модернизация проводится на уровне существующей модели.

Система Design/IDEF (фирма Meta Software) предназначена для концептуального проектирования сложных систем. С ее помощью разрабатываются спецификации, IDEF0- и IDEFlX-диаграммы, словари данных, проводится документирование и проверяется непротиворечивость проектов. Имеется дополнительная система Design/CPN, позволяющая проводить имитационное моделирование на основе моделей, преобразованных в цветные сети Петри.

Другой известной инструментальной средой моделирования приложений является Designer/2000 фирмы Oracle. Модель приложения может быть сгенерирована по ответам пользователя на вопросы системы. Используются собственные методики Oracle, позволяющие строить диаграммы потоков данных, сущность - отношение, иерархические деревья

данных с возможностью их представления в SQL формах, и, следовательно, поддерживается связь с любыми СУБД, работающими в ODBC.

Система Silverrun (фирма Computer Systems Advisors) предназначена для анализа и проектирования информационных систем. Реализовано раздельное функциональное и информационное моделирование. Система включает в себя четыре основные подсистемы: моделирование бизнес-процессов, построение моделей сущность - отношение, инфологическое проектирование реляционных баз данных, управление групповой работой. Имеется интерфейс к Oracle, Informix, Sybase и к ряду других СУБД.

Среди отечественных систем выделяется CASE - Аналитик, в которой выполняется построение диаграмм потоков данных, получение отчетов, генерация макетов документов и др. Имеется интерфейс к ERwin.

Методология объектно-ориентированного анализа и проектирования ПО по методике Г. Буча с использованием языка UML реализована в системах Rational Rose (фирма Rational Software Corporation) и Platinum Paradigm Plus (фирма Platinum Technology). В Rational Rose поддерживается генерация кода по построенным диаграммам классов, обратное моделирование (т. е. построение UML-модели по программному коду на таких языках, как C++, Java, Visual Basic, IDL CORBA), визуальное программирование. Язык UML применяют и в ряде других систем, например в инструментальной среде объектно-ориентированного проектирования ПО objectiF (фирма micro TOOL), в которой автоматически генерируется программный код по графическому UML-описанию.

Ряд программных продуктов, реализующих IDEF-модели, разработаны фирмой KBSI, в частности, ProSim реализует IDEF3, SmartER - IDEF1 и IDEF1X, SmartClass - IDEF4.

Поведенческое моделирование предприятий предусмотрено также в некоторых системах реинжиниринга, например в системе BAANIV.

Для преобразования функциональных или поведенческих моделей в имитационные применяют специальные программы. Так, вместе с программой BPWin для получения имитационных моделей используют программу BPSimulator. Преобразование IDEF0- модель -» сеть Петри реализовано в таких программах, как CPN/Design (фирма Meta Software) со специальным языком программирования ML, ProTem (Software Consultants International Limited) с вариацией типов меток, PACE (Grossenbacher software) с программированием на языке Smalltalk.

Контрольные вопросы

1. Какие функции выполняет сетевое ПО?
2. Перечислите основные подсистемы в системах ERP и их функции.
3. Дайте определение логистики и назовите основные функции логистических систем.
4. В чем заключается назначение систем SCAD A?
5. Что понимают под диаграммой потока данных?

6. Представьте IDEFO-диаграмму верхнего уровня для этапов жизненного цикла промышленной продукции.

7. Приведите пример неспецифического отношения.

8. Постройте IDEFIX-диаграмму для сущностей <студенческая группа>, <студент>, <преподаватель>, <дисциплина>.

9. Перечислите основные особенности БнД в САПР.

10. Что такое <транзакция> в системах обработки данных?

Лекция № 20.
Тема: Построение и порядок проектирования структурных, функциональных, принципиальных и монтажных схем.

План

20.1. Проектирование автоматизированных систем.

20.2.Общие положения, требования и правила при выполнении схем автоматизации.

20.3. Условные графические обозначения технических средств автоматизации.

20.4. Условные буквенно-цифровые обозначения технических средств автоматизации.

20.5. Общие требования к оформлению чертежей и текстовых документов

Ключевые слова: конструктор, техническое задание, подсистема, стандартизация, техническое требование, функциональная схема, структурная схема, принципиальная схема, функциональный цепь.

20.1. Проектирование автоматизированных систем.

Проектирование автоматизированных систем является сложным процессом. Стадии и этапы проектирования, а также состав проектных документов, разрабатываемых при проектировании АС, рассмотрены в предыдущих лекциях.

В связи со сложным интегрированным характером АС при их проектировании, как уже отмечалось, приходиться привлекать весь комплекс нормативных материалов РФ, регламентирующих проектные работы, а именно стандарты и руководящие материалы ЕСКД, ЕСТД, ЕСТПП, ЕСПД, СПДС и другие [4,5].

Практика применения стандартов на АСУ, САПР, АСУ ТП, АСТПП показала, что в них применяется одинаковый понятийный аппарат, имеется много общих объектов стандартизации, однако требования стандартов не согласованы между собой, имеются различия по составу и содержанию работ, различия по обозначению, составу, содержанию и оформлению документов и пр.

На фоне отсутствия единой технической политики в области создания АС многообразие стандартов не обеспечивало широкой совместимости АС при их взаимодействии, не позволяло тиражировать системы, тормозило развитие перспективных направлений использования средств вычислительной техники.

В настоящее время осуществляется переход к созданию сложных АС (за рубежом системы CAD - CAM), включающих в свой состав АСУ технологическими процессами и производствами, САПР - конструктора, САПР - технолога, АСНИ и др. системы. Использование противоречивых правил при создании таких систем приводит к снижению качества, увеличению стоимости работ, затягиванию сроков ввода АС в действие.

Единый комплекс стандартов и руководящих документов должен распространяться на автоматизированные системы различного назначения: АСНИ, САПР, ОАСУ, АСУП, АСУ ТП, АСУ ГПС, АСК, АСТПП, включая их интеграцию.

При разработке документов на создание АС необходимо учитывать следующие особенности АС, как объектов проектирования:

1) техническое задание является основным документом, в соответствии с которым проводят создание АС и приемку его заказчиком;

2) АС, как правило, создают проектным путем с комплектацией изделиями серийного и единичного производства и проведением строительных, монтажных. наладочных и пусковых работ, необходимых для ввода в действие АС;

3) в общем случае АС (подсистема АС) состоит из программно-технических (ПТК), программно-методических комплексов (ПМК) и компонентов технического, программного и информационного обеспечений.

Компоненты этих видов обеспечений, а также ПМК и ПТК должны изготовляться и поставляется, как продукция производственно-технического назначения.

Компоненты могут входить в АС в качестве самостоятельных частей или могут быть объединены в комплексы;

4) создание ЛС в организациях (предприятиях) требует специальной подготовки пользователей и обслуживающего персонала системы;

5) функционирование АС и комплексов обеспечивается совокупностью организационно-методических документов, рассматриваемых в процессе создания как компоненты правового, методического, лингвистического, математического, организационного и др. видов обеспечения. Отдельные решения, получаемые в процессе разработки этих обеспечений, могут реализовываться в виде компонентов технического, программного или информационного обеспечений;

6) совместное функционирование и взаимодействие различных систем и комплексов осуществляется на базе локальных сетей ЭВМ.

7) программно-технические и программно-методические комплексы рассматриваются как сложные изделия, не имеющие аналогов в

машиностроении. Учитывая статус ПТК и ПМК, как продукции производственно-технического назначения, правила и порядок их разработки должен быть аналогичен требованиям, установленным стандартами системы разработки и постановки продукции на производство (СРПП).

Спецификации и соглашения, принятые для локальных сетей ЭВМ обязательны для обеспечения совместимости систем, комплексов и компонентов.

Имеются две группы стандартов, ориентированных непосредственно на проектирование автоматизированных систем. Это группа Т52: «Система технической документации на АСУ» и группа П87: «Информационная технология. Комплекс стандартов на автоматизированные системы», которые образуют «Единый комплекс стандартов автоматизированных систем» (ЕКС АС).

В руководящем документе **РД 50-682-89** этого комплекса устанавливаются назначение, область действия, структура и положение по созданию этого комплекса.

РД 50-682-89 позволяет правильно ориентироваться в структуре ГОСТов по автоматизированным системам.

Назначение ЕКС АС заключается в установлении единых правил и требований, выполнение которых обеспечивает:

- необходимые технический уровень, качество и эффективность функционирования создаваемых АС;
- сокращение затрат и сроков создания АС;
- совместимость различных видов АС и их составных частей;
- типизацию и унификацию в АС, внедрение промышленных технологий создания АС и их составных частей;
- упорядочение процесса создания, развития и функционирования АС.

Документы, входящие в ЕКС АС, **должны устанавливать:**

- терминологию АС;
- классификацию АС и их составных частей;
- порядок создания, функционирования и развития АС;
- требования к составу и содержанию технической документации на АС;
- основные положения и требования к АС в целом;
- технические требования к составным частям АС, создаваемым как продукция производственно-технического назначения;
- требования к интерфейсам, протоколам обмена информацией и другим средствам, обеспечивающим совместимость составных частей АС, а также взаимосвязь различных АС между собой;
- показатели технического уровня и качества АС, методы контроля и испытания систем;

- требования к типовым и унифицированным проектным решениям АС.

Направления и задачи стандартизации при нормативно-техническом обеспечении процессов создания и функционирования АС заключается в следующем:

- установление технических требований к продукции;
- регламентация методов испытаний и правил аттестации и сертификации;
- регламентация правил и порядка разработки;
- установление правил документирования;
- обеспечение совместимости;
- регламентация организационно-методических вопросов функционирования систем.

Проектирование АС в основном, как это видно из ранее приведенных составов документов на создание АС, заключается в разработке ряда схем и текстовых документов. К основным схемам, описывающим проектные решения на создание АС, можно отнести следующие схемы:

- функциональные схемы автоматизации;
- структурные схемы автоматизации;
- принципиальные схемы автоматизации;
- схемы алгоритмов АСУ ТП;
- схемы и таблицы соединений и подключения внешних проводок.

Правила разработки именно этих схем и будут рассмотрены в данном лекционном курсе.

Кроме названных выше схем проектная документация на создание АСУТП содержит важные текстовые документы такие как, техническое задание, пояснительная записка, сметы и др.

Порядок оформления этих документов регламентируется соответствующими ГОСТами, в частности, ГОСТами на оформление текстовых документов и чертежей.

При разработке перечисленных схем, чертежей и текстовых документов необходимо соблюдать ряд общих правил, требований и положений, которые будут приведены далее.

20.2.Общие положения, требования и правила при выполнении схем автоматизации.

Схема - конструкторский документ, на котором показаны в виде условных изображений или обозначений составные части изделия и связи между ними. При выполнении схем используются следующие термины.

Элемент схемы - составная часть схемы, которая выполняет определенную функцию в изделии и не может быть разделена на части, имеющие самостоятельное назначение (резисторы, трансформаторы, диоды, транзисторы и т.п.).

Устройство - совокупность элементов, представляющая единую конструкцию (блок, плата, шкаф, панель и т.п.). Устройство может не иметь в изделии определенного функционального назначения.

Функциональная группа - совокупность элементов, выполняющих в изделии определенную функцию и не объединенных в единую конструкцию (панель синхронизации главного канала и др.).

Функциональная часть - элемент, функциональная группа, а также устройство, выполняющее определенную функцию (усилитель, фильтр).

Функциональная цепь - линия, канал, тракт определенного назначения (канал звука, видеоканал, тракт СВЧ и т.п.).

Линия взаимосвязи - отрезок прямой, указывающий на наличие электрической связи между элементами и устройствами.

Классификацию схем по видам и типам устанавливает **ГОСТ 2.701-84. Виды схем** определяются в зависимости от видов элементов и связей, входящих в состав изделия, и обозначаются буквами русского алфавита. Различают десять видов схем:

- электрическая - Э;
- гидравлическая - Г;
- пневматическая - П;
- газовая - Х;
- кинематическая - К;
- вакуумная - В;
- оптическая - Л;
- энергетическая - Р;
- деления - Е;
- комбинированная - С.

Схемы деления изделия на составные части (буквенное обозначение Е) разрабатывают для определения состава изделия. Комбинированные схемы выполняют, если в состав изделия входят элементы разных видов.

Схемы в зависимости от назначения подразделяют на **типы** и обозначают арабскими цифрами. Установлено восемь типов схем:

- структурная - 1;
- функциональная - 2;
- принципиальная (полная) - 3;
- соединений (монтажная) - 4;
- подключения - 5;
- общая - 6;
- расположения - 7;
- объединенная -0.

На объединенной схеме совмещаются различные типы схем одного вида, например схема электрическая соединений и подключения.

Наименование и код схемы определяются ее видом и типом. Код схемы должен состоять из буквенной части, определяющей вид схемы, и цифровой части, определяющей тип схемы. Например, схема электрическая принципиальная - Э3, схема гидравлическая соединений - Г4 и т. д.

Наименование комбинированной схемы определяется видами схем, входящими в ее состав, и соответствующим типом, например схема электрогидравлическая принципиальная - С3.

Наименование объединенной схемы определяется видом схемы и типами схем, входящими в ее состав, например схема электрическая соединений и подключения - ЭО. При выполнении комбинированных и объединенных схем должны соблюдаться правила, установленные для соответствующих видов и типов схем.

В технических документах, разрабатываемых при проектировании, эксплуатации и исследовании электротехнических устройств, применяют все типы схем, указанные выше, при этом на стадиях эскизного и технического проектирования разрабатывают структурные и функциональные схемы, на стадии рабочего проектирования - принципиальные, соединений, подключения, общие и расположения. Общее количество схем, входящих в комплект конструкторской документации на изделие, выбирается минимальным, но в совокупности они должны содержать сведения в объеме, достаточном для проектирования, эксплуатации, контроля и ремонта изделия. Между схемами одного комплекта осуществляется однозначная связь при помощи буквенно-цифровых позиционных обозначений. Такая связь необходима для быстрого отыскания одних и тех же элементов или устройств, входящих в схемы различного типа.

Общие правила выполнения схем устанавливают **ГОСТ 2.701-84**, **ГОСТ 2.702-75**, **ГОСТ 2.708-81**, **ГОСТ 2.710-81**, **ГОСТ 24.302-80**. Схемы выполняют без соблюдения масштаба, действительное пространственное расположение составных частей не учитывается или учитывается приближенно. Электрические элементы и устройства на схеме изображают в состоянии, соответствующем обесточенному. Элементы и устройства, которые приводятся в действие механически, изображают в нулевом или отключенном положении. При отклонении от этого правила на поле схемы необходимо давать соответствующие указания.

Общие требования к выполнению схем при разработке конструкторских документов на создание АС регламентируются **ГОСТ 24.302-80**.

В этом ГОСТе указывается, что на схемах АС приводят элементы схем и связи между элементами схем, а также необходимые поясняющие надписи. Элементами схемы являются условные графические обозначения объектов с их кодами или наименованиями. Связи между элементами отражают отношения между объектами.

Выделение группы элементов схемы по какому-либо признаку следует выполнять штрихпунктирной линией с поясняющей надписью в левом верхнем углу окаймления.

Линии связи, как правило, должны быть параллельны линиям внешней рамки схемы.

Направления линий связи сверху вниз и слева направо следует принимать за основные, допускается их стрелками не обозначать. В остальных случаях направления линий связи обозначают стрелками. Изображение стрелки – по **ГОСТ 2.307-68**. Слияние линий связи следует обозначать точкой в отличие от пересечения.

Обрывы линий должны быть обозначены. В местах обрывов допускается использовать идентификаторы в виде букв, цифр или букв и цифр.

Толщина линий – по **ГОСТ 2.303-68**.

Расстояние между соседними параллельными линиями связи должно быть не менее **3 мм**. Расстояния между соседними элементами схемы должно быть не менее **10 мм.**

Поясняющие надписи, условные обозначения, сокращения размещают на свободном поле листа (по возможности над основной надписью) и, при необходимости, сводят в таблицу.

В качестве условного графического обозначения элементов структурной схемы применяют**прямоугольник** с соотношением сторон **b = 1,5a**, где **a** выбирают из ряда **20, 25, 35, 40 мм**. Размеры a и b принимают по усмотрению проектировщика. Размеры a и b для всех символов одной схемы следует выбирать одинаковыми.

При выполнении схемы организационной структуры элементами схемы могут быть условные обозначения структурных подразделений, служб, пунктов управления и отдельных должностных лиц, реализующих функции и задачи управления.

При выполнении функциональной схемы автоматизации элементами схемы могут быть условные обозначения приборов и других функциональных средств автоматизации.

Связи на схеме показывают позиционное размещение приборов и устройств относительно управляемого объекта, информационные связи между элементами в процессе функционирования АСУ.

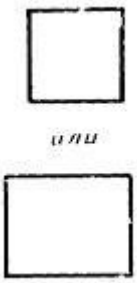
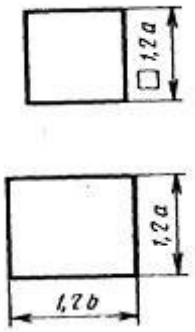
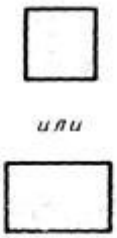
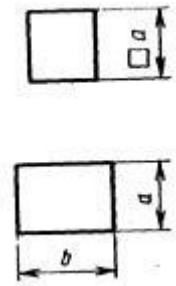
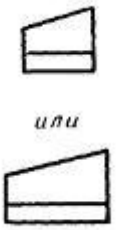
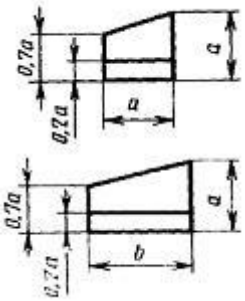
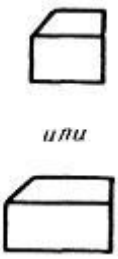
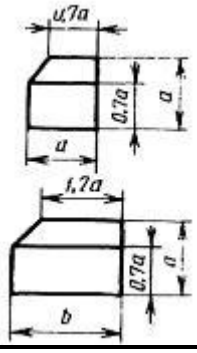
В ГОСТе приводятся требования к выполнению структурной схемы комплекса технических средств, а также требования к выполнению схемы соединений и подключений внешних проводок. Кроме того в **ГОСТ 24.302-80** даются ссылки на другие ГОСТы, которые следует использовать при разработке конкретных схем.

20.3. Условные графические обозначения технических средств автоматизации.

Условные графические обозначения технических средств автоматизации регламентируется **ГОСТ 24.303-80**.

В ГОСТе приводятся наименования устройств (комплексов устройств), изображения и размеры их общих обозначений (**табл. 6.1**). Наименования, изображения и размеры знаков, характеризующих работу устройств, приведены в табл. 20.1.

Таблица 20.1

Наименование устройства	Обозначение	
	Изображение	Размеры
1. Комплекс устройств(общее обозначение)		
2. Устройство (общее обозначение)		
3. Устройство терминальное		
4. Устройство регистрации и первичной обработки информации		

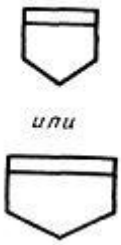
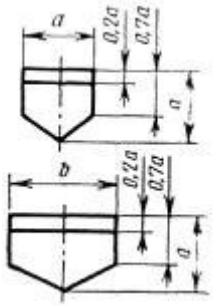
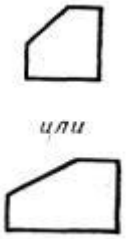
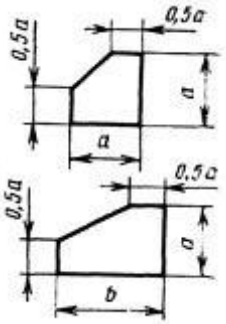
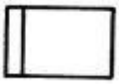
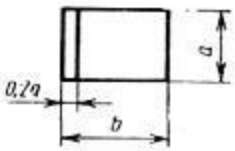
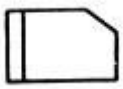
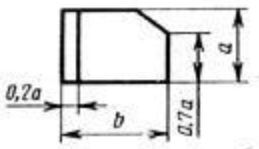
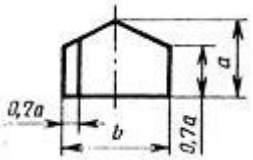
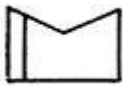
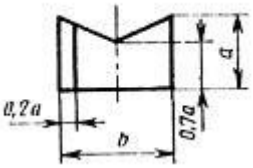
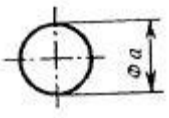
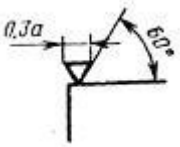
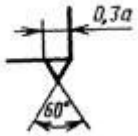
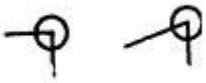
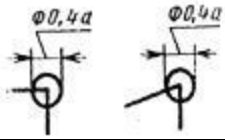
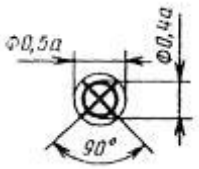
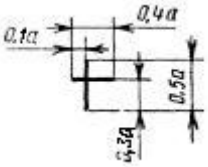
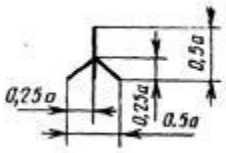
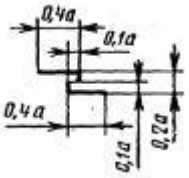
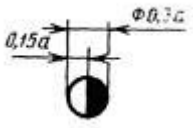
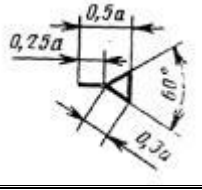
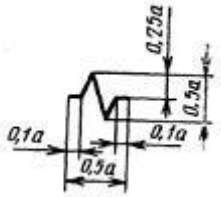
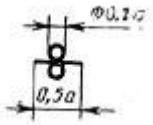
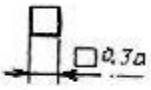
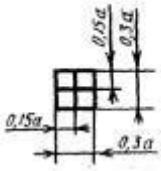
5. Устройство подготовки данных		
6. Кассовый аппарат		
7. Цифровая вычислительная машина общего назначения		
8. Машина перфорационного вычислительного комплекса		
9. Специализированная вычислительная машина		
10. Аналоговая и аналого-цифровая машина		

Таблица 20.2

Наименование устройства	Обозначение	
	Изображение	Размеры
1. Управление		

2. Ввод с машинного носителя		
3. Вывод на машинный носитель		
4. Вычисление		
5. Печать		
6. Защита от ошибок		
7. Коммутация		
8. Концентрация		
9. Сопряжение		
10. Индикация (общее обозначение)	ИН или 	
11. Индикация на электронно-лучевой трубке	ЭЛТ или 	

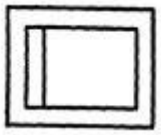
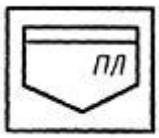
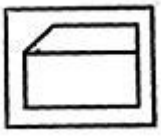
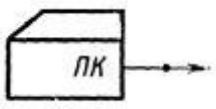
12. Ввод-вывод графической информации		
13. Копирование документации		
14. Хранение информации в оперативном запоминающем устройстве		
15. Хранение информации в постоянном запоминающем устройстве		

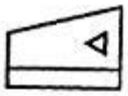
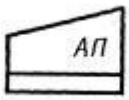
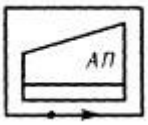
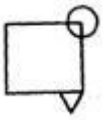
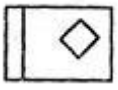
Для изображения комплекса устройств внутри общего обозначения комплекса (символ 1, табл. 20.1) помещают символ устройства того же назначения.

Для изображения устройства управления внутри символа соответствующего устройства помещают общее обозначение устройства управления (символ 1, табл. 20.2). Взаимное расположение символов друг в друге не регламентировано.

Примеры построения обозначений технических средств

Таблица 20.3

Наименование устройства	Обозначение
1. Вычислительный комплекс	
2. Система подготовки данных на перфоленте	
3. Фактурно-бухгалтерский комплекс	
4. Регистратор информации с вводом информации с перфокарты, с выводом информации в канал связи	

5. Устройство отображения информации на электронно-лучевой трубке	
6. Абонентский пункт, состоящий из одного устройства, работающий попеременно как передатчик и как приемник	
7. Передающий абонентский пункт, состоящий из нескольких устройств	
8. Накопитель на магнитной ленте	
9. Устройство управления накопителями	
10. Выводное печатающее устройство	
11. Чертежный автомат (графопостроитель)	
12. Табло индикации	
13. Устройство защиты от ошибок	
14. Устройство сопряжения	
15. Концентратор	
16. Процессор	
17. Вычислительная электронная клавишная машина	
18. Копировально-множительное устройство	
19. Печатающая машина	

20.4. Условные буквенно-цифровые обозначения технических средств автоматизации

Каждый элемент схемы, устройство или функциональная группа элементов должны иметь условное обозначение в соответствии с требованиями **ГОСТ 2.710-81**.

Система условных буквенно-цифровых обозначений, предусматриваемая указанным стандартом, позволяет производить в сокращенной форме запись сведений об элементах, устройствах, функциональных группах (в дальнейшем наряду с этими понятиями будет также применяться ссылки на соответствующие объекты в перечнях элементов, пояснительной записке и т.п.).

Эти же буквенно-цифровые обозначения могут наноситься непосредственно и на изделие, если это предусмотрено в его конструкции.

ГОСТ 2.710-81 предусматривает условные обозначения (классифицирующие символы) следующих типов элементов автоматизированных систем и позиционного расположения элементов (табл. 20.4):

- обозначение устройств высшего уровня - условное обозначение, присвоенное объекту, имеющему схему и перечень элементов; обозначение устройств высшего уровня применяют только в составных обозначениях;
- обозначение функциональной группы - условное обозначение, присвоенное функциональной группе, передающее, как правило, информацию о функциональном назначении функциональной группы;
- обозначение конструктивного расположения - условное обозначение, указывающее место расположения элемента или устройства в изделии;
- позиционное обозначение - условное обозначение, присвоенное каждому элементу и устройству, входящему в состав изделия, и содержащее информацию о виде элемента (устройства), его порядковый номер среди элементов (устройств) данного вида и, при необходимости, указание о функции, выполняемой данным элементом (устройством) в изделии;
- обозначение электрического контакта - условное обозначение, присвоенное электрическому контакту (выводу) элемента или устройства, предназначенному для осуществления электрических соединений или контроля;
- адресное обозначение - условное обозначение, указывающее место на документе, в котором содержится изображение (на схеме) или описание (в таблице) соответствующего элемента (устройства, функциональной группы); адресное обозначение применяют только в составных обозначениях;
- составное обозначение - условное обозначение, состоящее из двух и более условных обозначений различного типа и передающее совокупность сведений, содержащихся в условных обозначениях, входящих в его состав.

Указанные типы условных обозначений позволяют передавать комплексную информацию об объекте: входимость в состав устройства (обозначение высшего уровня), входимость элемента в функциональную группу (обозначение функциональной группы), место расположения элемента (конструктивное обозначение), позиционные обозначения элементов и обозначения электрических контактов, а также место изображения элемента в документации (адресное обозначение). Необходимость применения тех или иных видов обозначений, а также необходимость применения составного обозначения устанавливается разработчиком схемы.

Квалифицирующие символы условных обозначений

Таблица 20.4

Тип условного обозначения	Квалифицирующий символ	Наименование применяемого знака	Примечание
1. Обозначение высшего уровня - устройство	=	Равно	
2. Обозначение высшего уровня - функциональная группа	≠	Не равно	Допускается #
3. Обозначение конструктивного расположения (конструктивное расположение)	+	Плюс	
4. Обозначение элемента (позиционное обозначение)	-	Минус	
5. Обозначение электрического контакта	:	Двоеточие	
6. Адресное обозначение	( )	Круглые скобки	Обозначение заключают в круглые скобки

В принципиальных электрических схемах проектов автоматизации из перечисленных типов условных обозначений, как правило, применяются: позиционные обозначения элементов схем, обозначения электрических контактов (например, для контактов штепсельных разъемов и др.) и составное обозначение.

Составное обозначение образовывается, как правило, из обозначения функциональной группы и позиционного обозначения. В составное обозначение может быть включено и обозначение электрического контакта.

Составное обозначение вводится в сложных схемах, когда целесообразно различные схемы сгруппировать в функциональные группы и (или) выделить какие-либо устройства. Например, в качестве условных обозначений элементов можно применить составное обозначение, образованное из обозначений функциональной группы (ф75) и позиционного обозначения различных элементов схем (= К1, = SB1 и др.). Таким образом, условное обозначение можно записать: **= 75-К1.**

Составное обозначение образовывается последовательной записью условных обозначений различных типов в том порядке, в котором эти типы условных обозначений записаны в **ГОСТ 2.710-81**. Допускается изменять установленную последовательность записи различных типов, когда необходимо, например, передать более полную информацию о вхождении элементов, устройств или функциональных групп в устройства, функциональные группы более высокого уровня.

Например, может быть следующее составное обозначение: **=75-А1-К1** (реле К1 входит в состав устройства А1, которое входит в функциональную группу 75). Перед каждым условным обозначением, входящим в составленное обозначение, должен быть указан так называемый квалифицирующий символ - специальный, установленный стандартом знак, который указывает тип условного обозначения (табл. 20.4).

Запись условного обозначения с квалифицирующим символом, если оно не входит в составное, не требуется, хотя стандарт и допускает записывать его с квалифицирующим символом, если это необходимо для уточнения типа условного обозначения. Рассмотрим подробнее правила построения условных обозначений наиболее широко применяемых в принципиальных электрических схемах проектов автоматизации: функциональной группы, позиционного обозначения и электрического контакта.

20.5. Общие требования к оформлению чертежей и текстовых документов

Обозначение функциональной группы образуют из букв, в сокращенной форме указывающих функциональное назначение (функцию) группы, и порядкового номера. Допускается применять цифровое обозначение функциональной группы, в этом случае его нужно записывать с квалифицирующим символом, например =75. Обозначение функциональной группы указывают у ее изображения сверху или справа. Одинаковым функциональным группам (группам, имеющим тождественные принципиальные схемы) следует присваивать одно и то же условное обозначение. Допускается в условные обозначения одинаковых функциональных групп включать порядковые номера, отделяя их от основного обозначения.

Позиционные обозначения должны быть присвоены всем элементам и устройствам, изображенным на принципиальной электрической схеме. В общем случае позиционное обозначение должно состоять из трех частей, имеющих самостоятельное смысловое значение и записываемых без разделительных знаков и пробелов.

В первой части позиционного обеспечения должен быть указан вид элемента или устройства. Оно должно содержать одну или две буквы латинского алфавита - буквенный код видов элементов.

Во второй части позиционного обозначения должен быть указан порядковый номер элемента (устройства) в пределах элементов (устройств) данного вида. Например, в обозначении реле К1 первая часть представляет собой букву латинского алфавита (код), а цифра - порядковый номер этого реле в схеме. Порядковые номера элементам (устройствам) присваивают, начиная с единиц, в пределах группы элементов (устройств), которым на схеме присвоено одинаковое буквенное позиционное обозначение. Кроме того, порядковые номера элементам (устройствам) присваивают в соответствии с последовательностью расположения элементов или устройств на схеме сверху вниз в направлении слева направо. Это хорошо видно из схем на рис. 20.1, 20.2, 20.3. При необходимости допускается применять последовательность присвоения порядковых номеров в зависимости от размещения элементов в изделии, направления прохождения сигналов или функциональной последовательности процесса. Если в схему вносятся изменения, то последовательность присвоения порядковых номеров может быть нарушена.

В третьей части позиционного обозначения допускается указывать функциональное назначение данного элемента или устройства, использованного в схеме. Для этого применяют буквенные коды функций элементов (коды функционального назначения) в соответствии **ГОСТ 2.710-81**.

Третья часть обозначения в принципиальных электрических схемах проектов автоматизации применяется редко.

Для построения позиционного обозначения в качестве кода вида элементов рекомендуется применять **двух буквенные коды**. Однако в зависимости от конкретного содержания схемы элемент какого-либо типа может быть обозначен и одной буквой - общим кодом вида элемента. Например, если в схеме магнитного пускателя не содержится реле, то пускатель можно обозначить буквой К, хотя пускатель имеет и двухбуквенный код (КМ).

Позиционные обозначения проставляют на схеме рядом с условными графическими обозначениями элементов (устройств) с правой стороны или над ними.

При изображении на схеме элемента или устройства разнесенным способом позиционное обозначение элемента или устройства проставляют около каждой составной части. Иногда целесообразно (если это не усложняет

схему) раздельно изображенные части элементов соединять линией механической связи, указывая тем самым на принадлежность их к одному элементу. В этом случае позиционные обозначения элементов проставляют у одного или обоих концов линии механической связи. Если поле схемы разбито на зоны или схема выполнена строчным способом, то справа от позиционного обозначения или под позиционным обозначением каждой составной части элемента или устройства допускается указывать в скобках обозначение зон или номера строк, в которых изображены все остальные составные части этого элемента или устройства. На схемах питающей сети систем электропитания установок автоматизации в тех случаях, когда они выполняются в однолинейном изображении, около одного условного графического обозначения, заменяющего несколько условных графических обозначений одинаковых элементов, указывают позиционные обозначения всех этих элементов (см. рис. 20.3). Если одинаковые элементы находятся не во всех цепях, изображенных однолинейно, то справа от позиционного обозначения или под ним в квадратных скобках указывают обозначения цепей, в которых находятся эти элементы. На схеме около условных графических обозначений элементов и у линий связи при необходимости можно указывать характеристики элементов цепей (например, номинальные токи предохранителей, плавких вставок, напряжение цепей и др.).

Часто при составлении схемы возникает необходимость у ключей выбора режимов работы, переключателей, приборов, у другой аппаратуры помещать соответствующие надписи, знаки или графические обозначения. В этом случае надписи, знаки или графические обозначения, предназначенные для нанесения на изделие, на схеме заключают в кавычки. Около графических обозначений резисторов и конденсаторов допускается указывать их номиналы с упрощенным способом обозначения единиц измерения.

Обозначение электрического контакта. У всех выводов элементов и устройств, изображенных на схемах, должны быть указаны их обозначения, нанесенные на изделия или установленные в их документации.

Если в конструкции элемента (устройства) и в его документации обозначения выводов не указаны, то нужно условно присвоить им обозначения на схеме, повторяя их затем и в других документах с соответствующими пояснениями.

Если на схеме изображено несколько одинаковых элементов (устройств), то обозначения выводов допускается указывать на одном из них.

При разнесенном способе изображения одинаковых элементов (устройств) обозначения выводов указывают на каждой составной части элемента (устройства).

Для отличия на схеме обозначений выводов от обозначений других элементов, например от обозначений цепей и т.п., допускается записывать обозначения выводов с соответствующим квалифицирующим символом; на принципиальных электрических схемах проектов автоматизации обозначения

выводов, как правило, изображают, как показано на рис. 20.1 и 20.2 (условные точки у выводов с указанием номеров контактов).

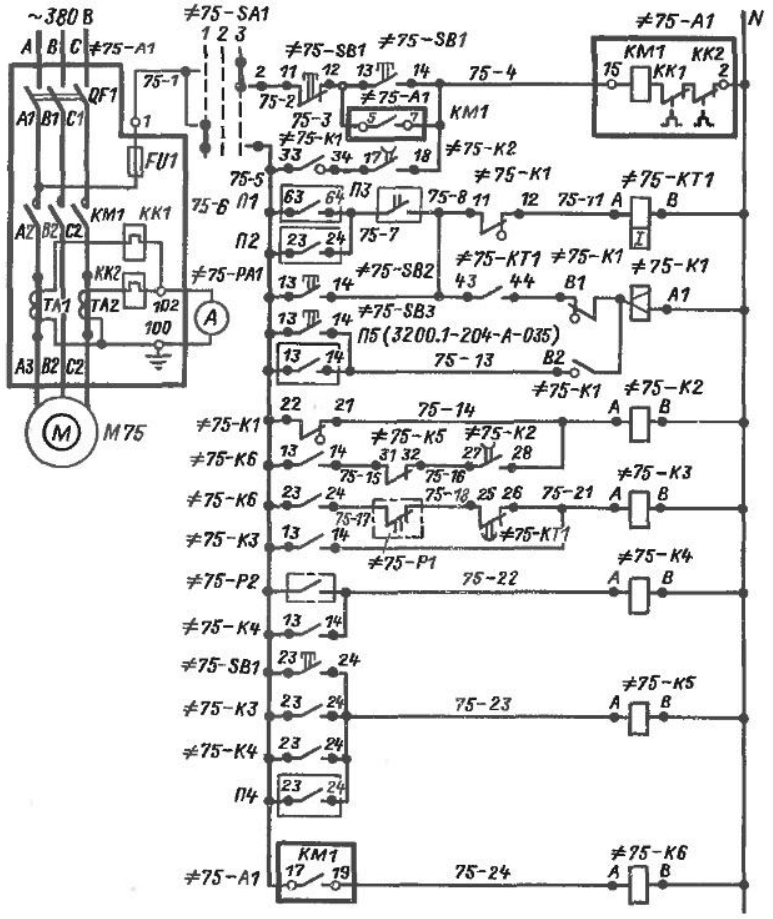
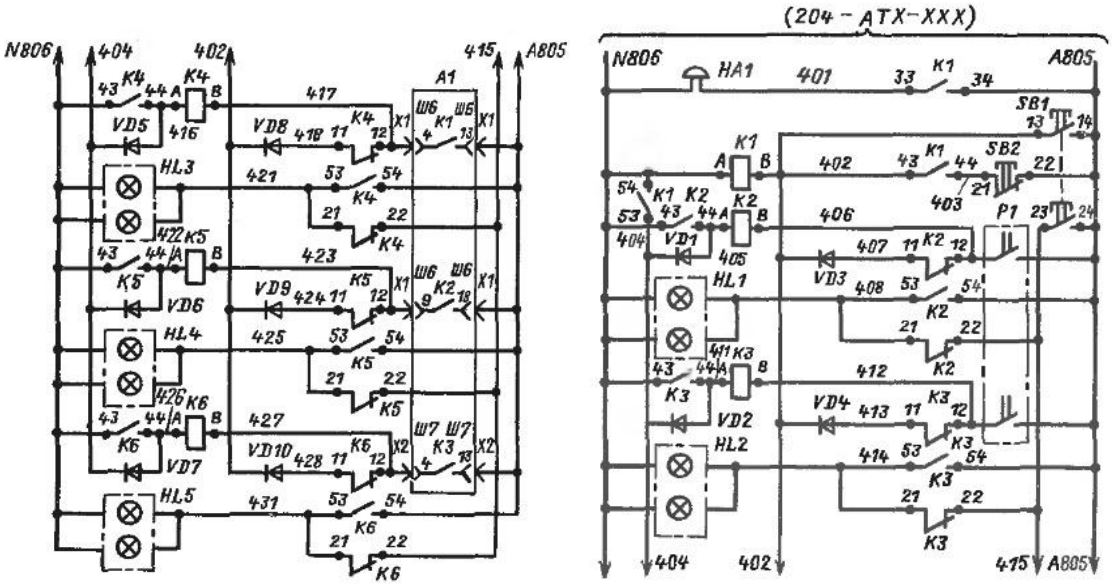


Рис. 20.1. Пример выполнения принципиальной электрической схемы управления



а) б)

Рис. 20.2. Пример выполнения принципиальной электрических схем сигнализации

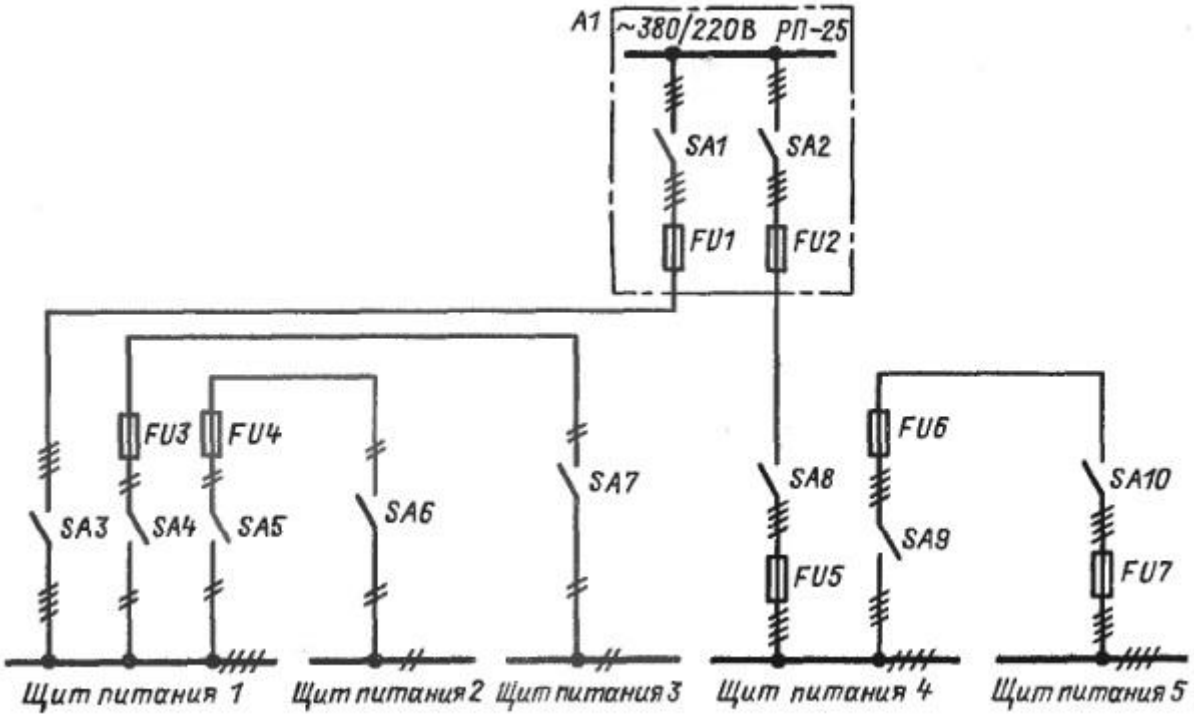


Рис. 20.3. Пример выполнения принципиальной электрической схемы питающей сети (однолинейное изображение)

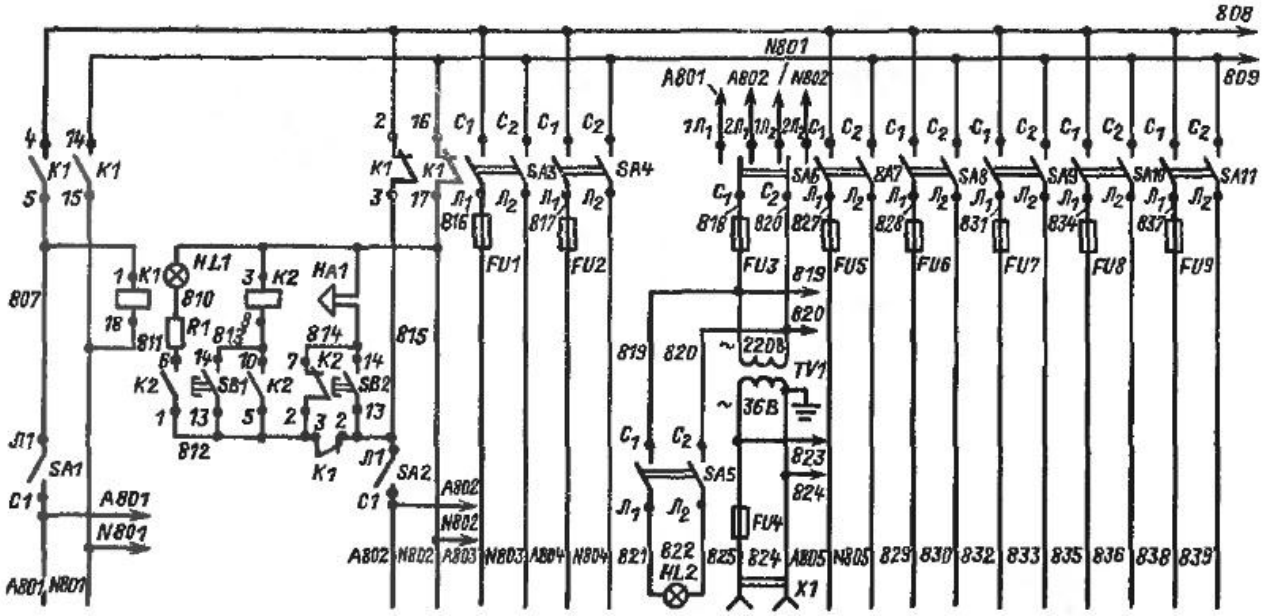


Рис. 20.4. Пример выполнения принципиальной электрической схемы распределительной сети

В **ГОСТ 24.304-82** даны требования к оформлению чертежей, формы документов и формы видеограммы. Приводятся ссылки на необходимые при оформлении чертежей АС ГОСТы.

ГОСТ 24.301-80 устанавливает требования к формам и основным надписями документов со ссылками на соответствующие ГОСТы.

Принимаются такие формы:

- титульный лист;
- заглавные листы различного назначения;

- листы регистрации изменений.

Устанавливается следующий состав документов:

- титульный лист;
- заглавный лист;
- содержание;
- основной текст;
- приложения;
- список литературы;
- лист регистрации изменений.

Приводятся правила оформления каждого из перечисленных документов.

В ГОСТе содержатся требования к построению и изложению текста документа, правила выполнения таблиц, иллюстраций и приложений, требования к оформлению машинописного текста, в частности, указывается, что нельзя размещать наименования разделов и подразделов в нижней части листа, если под ними помещают менее двух строк текста.

Контрольные вопросы

1. Какие особенности надо учитывать при создании АС?
2. В чем заключается назначение ЕКС АС?
3. В чем заключается направление и задачи стандартизации?
4. Какие общие положении и правила выполняется при выполнении схем автоматизации?
5. Какие буквенно – цифровые обозначение существует при автоматизации технических средств?
6. Какие типы условных обозначений применяются в принципиальных электрических схемах?
7. Какие общие требования подставляется к оформлению чертежей и текстовых документов?
8. Какие коды применяются для построения позиционного обозначения?

Лекция № 21.
Тема: STEP – технология.

План

21.1. Общие сведения о стандартах сопровождения промышленной продукции на всех этапах ее жизненного цикла.

21.2. Структура стандартов STEP.

21.3. Методы описания.

21.4. Методы реализации.

21.5. Организации 9 STEP информационных обменов.

Ключевые слова: STEP технология, стандарты Parametrics, стандарты Mandate, семейство стандартов, гипермедийные данные, язык Express, информационные модели, перекодировщик.

21.1. Общие сведения о стандартах сопровождения промышленной продукции на всех этапах ее жизненного цикла.

STEP (Standard for Exchange of Product data) - это совокупность стандартов (под номером ISO 10303), определяющих средства описания (моделирования) промышленных изделий на всех стадиях жизненного цикла. Совокупность стандартов STEP лежит в основе CALS-технологий.

Единообразная форма описаний данных о промышленной продукции обеспечивается введением в STEP языка Express, инвариантного к приложениям. Стандарты STEP не отрицают, а развивают методику информационного моделирования IDEF1X и предполагают возможность совместного использования с методикой функционального проектирования IDEF0 и рядом других международных стандартов (например, со стандартами ISO P-LIB, Mandate, SGML, CDIF и стандартом EIA 649).

Стандарт ISO 10303 состоит из ряда документов (томов).

Том ISO 10303-1 - вводный стандарт, описывающий структуру всей совокупности томов и основные принципы STEP. В следующих группах томов содержатся описания инвариантного к приложениям языка Express, даны методы его реализации, модели, ресурсы, как общие для приложений, так и некоторые специальные (например, геометрические и топологические модели, описание материалов, процедуры черчения, конечно-элементного анализа и т. п.), прикладные протоколы, отражающие специфику моделей в конкретных предметных областях, методы тестирования моделей и объектов.

Удовлетворению требований создания открытых систем в STEP уделяется основное внимание

- специальный раздел посвящен правилам написания файлов обмена данными между разными системами, созданными в рамках STEP-технологии.

Развитие линии стандартов STEP находит выражение в разработке новых стандартов Parts Library (ISO 13584), Parametrics (ISO 14959), Mandate (ISO 15531).

Стандарты Parts Library (P-LIB) содержат обзор и основные принципы представления данных о стандартных компонентах промышленных изделий. В этих стандартах представлены в виде библиотек данные о семействах таких типовых широко используемых компонентов изделий, как болты, подшипники, электронные компоненты и т.п., с целью использования этих данных в системах автоматизированного проектирования. В P-LIB содержатся также правила использования, интерфейса и модификации библиотечных описаний. Цель стандарта -

обеспечить инвариантный для приложений механизм оперирования частями библиотеки.

Благодаря ISO 13584 различные прикладные САПР могут разделять данные из обобщенных баз, беспрепятственно обмениваться данными о типовых компонентах.

Стандарты P-LIB состоят из нескольких частей.

Часть 1 - обзор и основные положения серии стандартов.

Номера с 10 по 19 отведены для частей, содержащих концептуальные положения.

Номера с 20 по 29 выделены для описания логических ресурсов. Здесь разработаны части: 20 - общие ресурсы; 24

- логическая модель поставляемой библиотеки (Logical model of supplier library); 26 - определение поставщиков (Supplier Identification).

Номера 30-39 используются для описания ресурсов внедрения. Здесь разработана часть 31 - интерфейс геометрического программирования (Geometric Programming Interface).

Описание методологии структуризации семейств содержится в части. Протоколам обмена посвящены части, начинающиеся с номера 101. Часть под номером 101 содержит протокол обмена геометрической параметризованной информацией; часть под номером 102 - протокол обмена согласованными с STEP данными.

Стандарты Parametrics введены сравнительно недавно (1996 г.) в связи с тем, что стандарты STEP в недостаточной мере учитывали особенности современных САПР в части представления параметризованных моделей изделий и обмена параметризованными данными.

Рабочая группа ISO по Parametrics решает как краткосрочные, так и перспективные задачи. Первые из них касаются удовлетворения потребностей геометрического проектирования и машинной графики в сегодняшних САПР, в которых широко используются параметризованные модели. Вторые касаются попыток распространения идей параметризации на более ранние этапы проектирования и на более широкий круг моделей и процедур проектирования, имеющих не только геометрический характер [4,5].

Стандарты Mandate посвящены представлению данных, относящихся к функционированию предприятий, управлению территориально распределенными производственными системами, обмену данными о производстве с внешней для предприятия средой.

Часть стандарта, обозначаемая ISO 15531-21, содержит обзор и основные принципы представления данных о промышленной продукции. Содержание этой части характеризуется следующими ключевыми словами: системы промышленной автоматизации и интеграция, промышленные

данные, обмен данными об управлении производством, обмен данными с внешней средой.

Том ISO 15531-31 посвящен обзору и основным принципам использования данных о производственных ресурсах. Излагаются модель, форма и атрибуты представления данных о производственных ресурсах, об управлении их использованием.

Том ISO 15531-41 содержит обзор и основные принципы управления потоками производственных данных.

Семейство стандартов SGML (ISO 8879) предназначено для унификации представления текстовой информации в АС. В цикле проектирования промышленной продукции стандарты SGML обслуживают стадию, на которой выполняется документирование результатов. Стандартная форма документов способствует их правильной передаче, интерпретации и многократному использованию многими системами и пользователями. Стандарты SGML разрабатывались прежде всего применительно к текстовым документам, но их возможности шире, так они применимы для документирования гипермедийных данных.

Роль стандартов SGML конкретизируется следующими направлениями их использования.

1. Единообразное представление структуры данных, включая классификацию и идентификацию типов документов, поддержку различных типов символов и языковых ограничений.

2. Дополнение моделей промышленных изделий, задаваемых в настоящее время стандартами STEP, моделями документов.

3. Обмен данными между различными АС, электронными или традиционными средствами публикации и прежде всего между STEP и SGML средами. Для достижения этой цели SGML-формы должны быть согласованны с формой обменного файла STEP, описываемого в томе ISO 10303-21.

Использование возможностей SGML в STEP-ресурсах осуществляется с помощью информационной структуры SGML_STRING, включаемой в модели на языке Express. Эта структура содержит информацию о требуемом документальном оформлении данных и, следовательно, позволяет выполнять в STEP-среде перечисленные выше функции SGML. Тем самым реализуется интеграция STEP- и SGML-стандартов.

Стандарт SGML устанавливает такие множества символов и правил для представления информации, которые позволяют правильно распознавать и идентифицировать эту информацию. Названные множества описывают в отдельной части документа, называемой декларацией DTD (Document Type Declaration), которую помещают в начале SGML-документа. В DTD указывают соответствие символов и их кодов, максимальные длины используемых идентификаторов, способ представления ограничителей для тегов (примером может служить символ “<” для тегов в HTML), другие

возможные соглашения, синтаксис DTD, а также тип и версия документа. Следовательно, SGML можно назвать способом описания семейства конкретных языков разметки. В частности, подмножествами SGML являются языки разметки XML и HTML.

Стандарт EIA 649 посвящен ***управлению конфигурацией*** изделий. В нем установлены базовые принципы управления конфигурацией и правила управления внесением изменений в документацию, рассматриваются такие вопросы, как идентификация документа, взаимосвязи конфигурации продукта и данных, контроль версий данных и доступа к данным и др. В стандарте вводятся уровни статуса данных, к которым относится документ на том или ином этапе своего жизненного цикла. Возможны уровни рабочих, выпущенных, представленных и утвержденных данных. На уровне рабочих данных с документом работает его составитель (разработчик). На уровне выпущенных данных документ доступен соответствующим подразделениям организации-изготовителя, здесь и далее любое изменение данных требует выполнения специальных согласительных процедур. Представленные данные уже доступны для просмотра заказчикам (потребителям). Статус утвержденных данные получают после одобрения заказчиком.

21.2. Структура стандартов STEP.

Построение открытых распределенных АС для проектирования и управления в промышленности составляет основу современной CALS-технологии. Главная проблема их построения - обеспечение единообразного описания и интерпретации данных, независимо от места и времени их получения в общей системе, имеющей масштабы вплоть до глобальных. Структура проектной, технологической и эксплуатационной документации, языки ее представления должны быть стандартизованными. Тогда становится реальной успешная работа над общим проектом разных коллективов, разделенных во времени и пространстве и использующих разные CAE/CAD/CAM-системы. Одна и та же проектная документация может быть использована многократно в разных проектах, а одна и та же технологическая документация - в разных производственных условиях, что существенно сократит и удешевит общий цикл проектирования и производства. Упрощается эксплуатация систем.

Эти цели поставлены при разработке стандартов STEP. К их разработке под эгидой ISO привлечен ряд ведущих специалистов фирм в разных отраслях промышленности.

Основу STEP составляет язык Express. Это язык унифицированного представления данных и обмена данными в компьютерных средах. Язык инвариантен к приложениям. Хотя он разрабатывался с ориентацией прежде всего на описание жизненных циклов промышленной продукции, области его применения значительно шире.

В STEP используются также следующие основные понятия:

AAM - ***Application Activity Model***; это функциональная модель IDEF0 для определенного приложения;

ARM - ***Application Requirements Model***; это модель данных, представленная обычными средствами приложения;

AIM - ***Application Interpreted Model***; это ARM модель, переведенная в STEP представление;

AP - ***Application Protocol***; это STEP стандарт, отражающий специфику конкретного приложения;

SDAI - ***Standard Data Access Interface***; программный интерфейс к источникам данных (репозиториям) прикладных систем (в том числе к библиотекам моделей CAD/CAM систем) с переводом моделей в STEP файлы, используется в STEP средах для организации обменов между приложениями через общую базу данных STEP.

STEP - это совокупность стандартов и состоит из ряда томов. Тома имеют свои номера ***N*** и обозначаются как "часть N" или ISO 10303-N. Приведем краткую характеристику следующих основных групп томов:

- том ISO 10303-1 - вводный стандарт, выполняющий роль аннотации всей совокупности томов. В этом стандарте вводится ряд терминов, используемых в других стандартах, например, таких как продукт (product), приложение (application), проектные данные (product data), модель (model), модели AAM, АШ, ARM, прикладной протокол (AP), интегрированный ресурс (integrated resource), элемент функциональности (unit of functionality - UoF).

- части 11 - 14 - методы описания (Description methods).
- части 21 - 29 - методы реализации (Implementation methods).
- части 41 - 50 - интегрированные основные ресурсы (Integrated generic resources).
- части 101 - 108 - интегрированные прикладные ресурсы (Integrated application resources).
- части 201 - 236 - прикладные протоколы (Application protocols).
- части 501 - 520 - прикладные компоненты (Application interpreted constructs).

21.3. Методы описания.

Первая группа документов - тома, с номерами в диапазоне с 11 до 19 отведены для описания диалектов языка Express.

N=11: ***Express language reference manual.*** Основное руководство по языку Express. Содержит также описания расширения Express-C базового языка и графического варианта языка Express-G. Базовый язык приспособлен для описания и передачи статических свойств объектов приложений, т.е. параметров структур и ограничений. Поэтому Express-C включает средства описания динамических свойств объектов (добавлено описание событий и транзакций). Для наглядности представления языковых конструкций в Express предусмотрены графические средства изображения моделей, в качестве которых может использоваться специальное дополнение Express-G

(графический Express). Express-G - язык диаграмм, напоминающий язык описания информационных моделей в методике IDEF1X.

N=12: Express-I Language Reference Manual. Express-I - расширение языка, предназначенное для описания отдельных экземпляров данных.

Разрабатываются дополнения, относящиеся к следующим диалектам языка:

- Express-M: ***Mapping definition language***; язык для описания соответствий между сущностями и атрибутами некоторых моделей, представленных в виде схем на языке Express. Например, этими схемами могут быть два разных прикладных протокола, имеющих частично общие данные, или две схемы одного приложения, но созданные разными лицами (при отсутствии соответствующего AP). Одна схема есть схема-источник, другая - целевая схема. Целевых схем может быть несколько при одной схеме-источнике. Предложения Express-M транслируются на язык C, результирующая программа представляет собой совокупность обращений к функциям базы данных SDAI в STEP-среде. Другими словами, транслятор относится к системе SDAI (см. протокол ISO 10303-22), а Express-M можно рассматривать, как язык 4GL для обращений к функциям базы данных SDAI.

Express-X: промежуточный язык, аналогичный Express-M и используемый для описания соответствий между типами данных в заданной исходной Express-схеме и создаваемыми новыми ее вариантами (views); в качестве views могут использоваться форматы с описанием того же множества сущностей, что и в Express-схеме, например, формат IGES (описанию языка Express-X посвящен будущий стандарт ISO 10303-14).

-Express-P: ***Process definition language***; язык диаграмм для представления процессов, методов и коммуникационных структур.

- Express-V: язык, предназначенный для получения ARM представлений из AIM моделей, другими словами, для описания процедур поиска экземпляров Express-объектов, отвечающих заданным условиям, и доступа к ним, например, при создании новых ARM. Эти создаваемые ARM-представления обычно не требуют столь всестороннего описания приложения, как в AIM, и потому могут быть существенно проще. В Express-V имеются: 1) схема-источник (AIM), обычно это прикладной протокол, например, AP203; 2) схема-цель, задающая сущности, которые должны быть в создаваемой частной модели; 3) схема отображения нужных сущностей из источника в цель. На языке Express-V описываются условия (в виде клозов WHEN) такого отображения. берется подходящая уже существующая AIM, как источник, все совпадающие объекты переводятся в ARM, далее описываются оригинальные объекты. Дополнительной возможностью реализаций Express-V является обратное отображение специфики создаваемой ARM в исходную AIM с целью развития прикладных протоколов.

Для возможности применения языка Express должны быть разработаны методы реализации (Implementation Methods), которые могут быть

представлены средствами файлового взаимодействия, построением БД, интерфейсом с языками программирования.

21.4. Методы реализации.

Вторую группу (тома с номерами 21...29) называют “Методы реализации”, она служит для реализации межпрограммного информационного обмена между прикладными системами в STEP-среде. Предусмотрены межпрограммные связи с помощью обменного файла и доступа к БД.

N=21: Clear Text Encoding of the Exchange Structure (physical transfer file format); стандарт устанавливает правила оформления обменного файла. Обменный файл играет в STEP важную роль; если собственно на языке Express определены сущности, то именно в обменном файле задаются экземпляры этих сущностей. Прикладные программы для связи со STEP средой должны читать и генерировать обменные файлы.

N=22: Standard Data Access Interface Specification; содержит описание SDAI - системы представления данных и доступа к данным конкретных прикладных систем (чаще всего это CAD/CAM системы). Данные, участвующие в межпрограммных связях, образуют SDAI-модели. В SDAI системе предусматривается компилятор кода, конвертирующего эти модели в SDAI базу данных, а также функции обращения к этой базе данных. Возможно непосредственное построение прикладных систем, работающих с SDAI базой данных.

Тома с номерами ***N*** = 23.. .29 устанавливают правила обращения к данным в SDAI базе данных на языках программирования C++, C, Java, на языке передачи данных в системах распределенных вычислений IDL, языке разметки XML.

Прикладные протоколы. Прикладным протоколом в STEP называют информационную модель определенного приложения, которая описывает с высокой степенью полноты множество сущностей, имеющихся в приложении, вместе с их атрибутами, и выражена средствами языка Express. Предполагается, что эта модель содержит в себе описание данных любой конкретной задачи соответствующего приложения, т.е. практические информационные модели прикладных задач оказываются частными случаями прикладных протоколов.

Прикладные протоколы в стандарте ISO 10303 содержатся в томах, начиная с N=201. Прикладные протоколы принято обозначать аббревиатурой AP с указанием номера, например, AP203, AP214. Для связи прикладной системы со STEP используемые ею данные должны быть описаны в соответствующем AP.

Список большинства разработанных прикладных протоколов приведен в приложении. Число прикладных протоколов в STEP может расширяться за счет разработки новых протоколов.

В прикладных протоколах широко используются типовые фрагменты информационных моделей, встречающиеся более чем в одном приложении.

Эти фрагменты называют интегрированными общими и прикладными ресурсами.

Типовые фрагменты информационных моделей. Четвертая группа стандартов STEP (тома с номерами 41... 50) “Интегрированные общие ресурсы” описывает общие для приложений ресурсы, под которыми понимаются основные компоненты (building blocks) для моделей прикладных протоколов. Например, описания геометрических объектов в виде поверхностей Безье или 5-сплайнов могут использоваться во многих прикладных протоколах, поэтому эти описания вынесены в группу интегрированных ресурсов.

Тома с номерами 1 01 по 1 99 отведены для документов, относящихся к более специальным средствам, называемым интегрированными прикладными ресурсами (Integrated application resources).

Группа стандартов с номерами, начинающимися с $N = 501$, служит для описания данных о геометрических элементах и моделях некоторых конкретных типовых объектов и конструкций, часто используемых в ряде интегрированных ресурсов и прикладных протоколов [4,5].

Номера и названия томов, описывающих интегрированные и прикладные ресурсы и прикладные компоненты, приведены в приложении.

21.5. Организации 9 STEP информационных обменов.

Возможны обмены через обменный файл и через базу данных SDAI. Эти способы поясняются на рис. 21.1 и 21.2 соответственно.

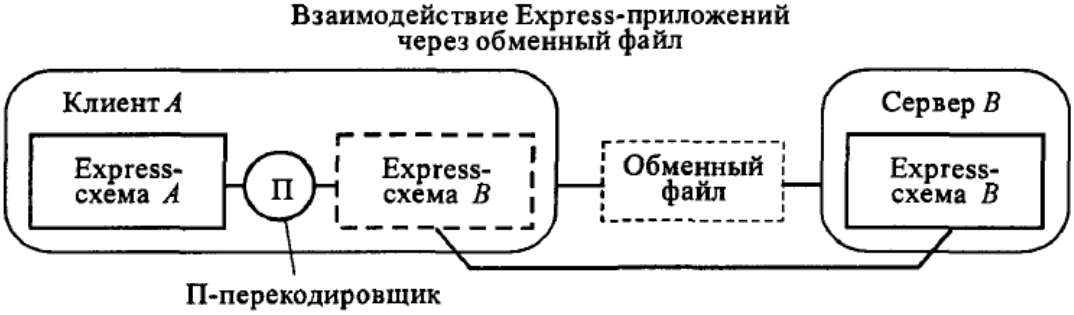


Рис. 21.1. Взаимодействие Express-приложений через обменный файл

Рис. 21.2. Взаимодействие Express-приложений через базу данных SDAI

Обменный файл (см. рис. 21.1) используется при связи двух систем ***A*** и ***B,*** имеющих общие данные с различными обозначениями. Пользователь должен написать перекодировщик (например, на языке Express-X), с помощью которого отождествляются идентификаторы одних и тех же сущностей, имевших разные обозначения в схемах *A* и B.

Связь через базу данных SDAI (рис.21.2) отличается от обмена по схеме рис.

21.1 тем, что здесь имеет место не просто обмен, а разделение данных многими пользователями, и SDAI фактически выступает в роли метамодели для разных САПР.

Описание языка Express в сокращенном виде приведено в следующем параграфе.

Стандарты управления качеством промышленной продукции. Международные стандарты серии ISO 9000 разработаны для управления качеством продукции, их дополняют стандарты серии ISO 14000, отражающие экологические требования к производству и промышленной продукции. Хо - тя эти стандарты непосредственно не связаны со стандартами STEP, их цели - совершенствование промышленного производства, повышение его эффективности - совпадают.

Очевидно, что управление качеством тесно связано с его контролем. Контроль качества традиционно основан на измерении показателей качества продукции на специальных технологических операциях контроля и выбраковкой негодных изделий. Однако есть и другой подход к управлению качеством, основанный на контроле качественных показателей не самих изделий, а проектных процедур и технологических процессов, используемых при создании этих изделий. Такой подход более эффективен и требует меньше затрат. Именно он и положен в основу стандартов ISO 9000.

Под *качеством* продукции в ISO 9000 подразумевается своевременное удовлетворение требований заказчика при приемлемой цене. Документальную систему с руководствами и описаниями процедур достижения качества называют *системой качества* (QS - Quality System).

Система качества обычно представляет собой совокупность трех слоев документов. Слои содержат: 1) описание политики управления для каждого системного элемента (организация, ответственные, контроль); 2) описание процедур управления качеством (что, где, кем и когда должно быть сделано); 3) тесты, планы, инструкции и т.п.

Сертификация предприятий по стандартам ISO 9001-9003 выполняется некоторой уполномоченной внешней организацией. Наличие сертификата качества - одно из важных условий для успеха коммерческой деятельности предприятий.

Стандарты ISO 14000 посвящены проблеме выполнения промышленными предприятиями экологических требований. По своей целевой направленности они близки стандартам управления качеством промышленной продукции ISO 9000, которые служат базой для ISO 14000.

Стандарты ISO 14000 являются также системой управления влиянием на окружающую среду, они, как и ISO 9000, реализуются в процессе сертификации предприятий, задают процедуры управления и контроль документации, аудит, подразумевают соответствующее обучение и сбор статистики. Кроме требований заказчиков и покупателей, здесь воплощаются внутренние требования организации.

Краткое описание стандартов серий ISO 9000 и ISO 14000 дано в приложении.

Контрольные вопросы

1. Назовите причины появления стандартов STEP.
2. Что является предметом стандартизации в CALS-технологиях?
3. Поясните назначение языков разметки. Что такое декларация DTD?
4. Что называют прикладным протоколом в STEP-технологиях? Что такое интегрированные ресурсы?
5. На какие классы подразделяются геометрические модели в протоколе AP203?
6. Представьте на языке Express IDEFlX-диаграмму, построенную для сущностей «студенческая группа», «студент», «преподаватель», «дисциплина».
7. Опишите назначение и структуру обменного файла в языке Express.
8. Для чего нужны разновидности языка Express, такие, как Express-X и Express-V?
9. Дайте характеристику подхода к контролю качества продукции, принятому в стандартах ISO 9000.

ЛИТЕРАТУРА

1. To’layev B. Loyihalash jarayonlarini avtomatlashtirish asoslari. ALTning material va dasturiy ta`minoti.O’quv qo’llanma. –T.: TDTU. 2009, 180 b.
2. To’layev B. Loyihalash jarayonlarini avtomatlashtirish asoslari.Chizmalarni avtomatlashtirilgan ishlab chiqish tizimlari. O’quv qo’llanma. –T.: TDTU. 2009, -103 b.
3. To’layev B. Loyihalash jarayonlarini avtomatlashtirish asoslari.Hisobiy loyihalarni MathCADda bajarish. O’quv qo’llanma. –T.: TDTU. 2009, -154 b.
4. Норенков И.П. Автоматизированное проектирование. Учеб.пособие. –М.: Изд-во МГТУ им. Н.Э. Баумана, 2000. -188с
5. Норенков И.П. Основы автоматизированного проектирования. Учеб.пособие. –М.: Изд-во МГТУ им. Н.Э. Баумана, 2002. -336с
6. Норенков И.П., Трудоношин В.А. Телекоммуникационные технологии и сети. Учеб.пособие. –М.: Изд- во МГТУ им. Н.Э. Баумана, 1998. -287с
7. Частников А.П., Гаврилова Т.А., Белов Д.Л. Разработка экспертных систем. Среда CLIPS. Учеб.пособие. - СПб, BHV 2003. -606с.
8. Макаров И.М., Топчиев Ю.И. Робототехника. История и перспективы. Учеб. Пособие.-М.: Наука, МАИ, 2003. -350с.
9. Сольницев Р.И. Автоматизация проектирования систем автоматического управления:
10. http://www.intuit.ru
11. http://www.prorobot.ru
12. http://ziyonet.uz
13. http://www.knigka.info
14. http://knigi.b111.org

СОДЕРЖАНИЕ