Курганский государственный университет
Кафедра программного обеспечения
автоматизированных систем

К У Р С Л Е К Ц И Й

по дисциплине

ВВЕДЕНИЕ В ПРОГРАММНУЮ ИНЖЕНЕРИЮ

для студентов направления 231000.62

«Программная инженерия»

Лекция 3.1

Визуальное моделирование при анализе и проектировании программных систем

План лекции

1. Задачи и базовые принципы моделирования программных систем
2. Визуализация при моделировании сложных систем
3. Краткая история развития средств визуального моделирования
   - 3.1. Теория множеств
   - 3.2. Теория графов
   - 3.3. Семантические сети
   - 3.4. Диаграммы структурного анализа систем
     - 3.4.1. SADT - Диаграммы функционального моделирования
     - 3.4.2. ERD - Диаграммы "*Сущность – Связь*"
     - 3.4.3. DFD - Диаграммы потоков данных
4. Заключение
5. Контрольные вопросы и задания

Проблема:

Результаты разработки не удовлетворяют требованиям заказчика

Причина:

Разработчиками были приняты неверные проектные решения

Что делать ?

Возврат к пройденным стадиям разработки и частичное (а иногда и полное) перепроектирование системы.

Результат

Программный проект не укладываются в установленные сроки и выделенный бюджет. Проблема не решена.

Как решать эту проблему

или хотя бы свести к минимуму негативный эффект от некорректных проектных решений?

Один из подходов –

использование моделирования проектируемого объекта на всех стадиях проекта.

Задачи моделирования программных систем

Модель - упрощенное представление какого-либо объекта или явления реального мира.

Модели строят для того, чтобы исследовать и лучше понять проектируемую систему.

Моделирование позволяет решать проектировщикам следующие ***основные задачи***:

- Определение структуры системы, как множества взаимосвязанных компонентов.
- Определение поведения системы в различных ситуациях.
- Визуализация системы в некоторых её состояниях.
- Получение шаблона для разработки системы.
- Документирование принимаемых проектных решений

Базовые принципы моделирования

- Каждая модель может быть воплощена на некотором уровне абстракции. Степень детальности рассмотрения свойств объекта в его модели определяется стадией проекта и задачами моделирования.
- Любая модель реализует упрощенный взгляд на реальный объект; лучшая модель - та, что ближе к реальности.
- Нет идеальных моделей. Наилучший подход при разработке сложной системы - использовать несколько почти независимых моделей.
- Выбор модели оказывает определяющее влияние на подход к решению проблемы и на то, как будет выглядеть это решение.

Проектирование – процесс преобразования информационных моделей объекта

Проектирование – это, по существу, производство документации, содержащей описание проектируемого объекта.

Проектирование – это информационный процесс, то есть процесс преобразования информации о проектируемом (еще не реализованном) объекте.

Проектирования – процесс последовательного преобразования информационных моделей проектируемого объекта.

При выполнении крупных проектов сложность моделируемого объекта и, соответственно, объем информации, обрабатываемой проектировщиками, слишком велики для адекватного восприятия проекта одним человеком.

Декомпозиция проекта частично решает проблему «борьбы со сложностью», но создает ряд дополнительных проблем, связанных с интеграцией компонентов системы и организацией согласованной работы нескольких групп разработчиков

Визуализация при моделировании сложных систем

В условиях коллективного проекта и высокой сложности системы для документирования и моделирования процессов её функционирования ***описательные языки оказываются не эффективными,*** так как не обеспечивают требуемого уровня полноты и непротиворечивости описания системы.

Визуальное моделирование - графическое отображение обсуждаемых и принимаемых проектных решений.

Достоинства визуального моделирования:

- Визуализация ***упрощает понимание проекта*** в целом ("*картина стоит ста слов о ней*").
- Визуализация ***помогает согласовать терминологию*** и убедиться в том, что все участники проекта одинаково понимают термины.
- Визуализация ***делает обсуждение конструктивным*** и понятным.
- Визуализация ***дает возможность образного закрепления семантики*** (содержательного смысла) отдельных понятий, что существенно упрощает процесс общения между участниками проекта.

Исторические предшественники:
графические нотации математических моделей

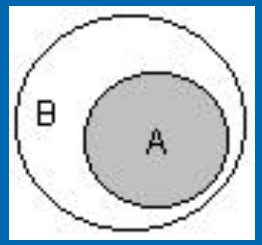
1. Теория множеств

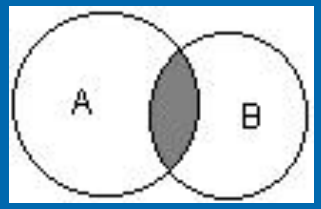
Диаграммы Венна

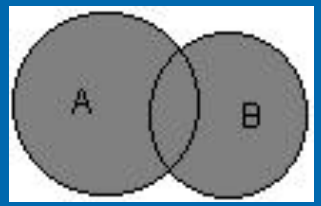
$A \in B$

$A \cap B$

$A \cup B$







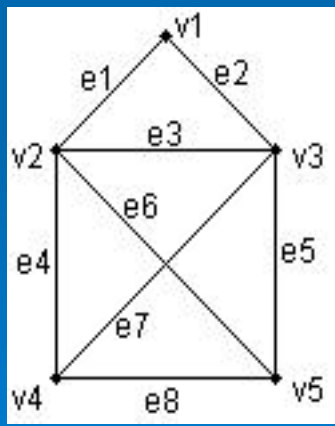
2. Теория графов

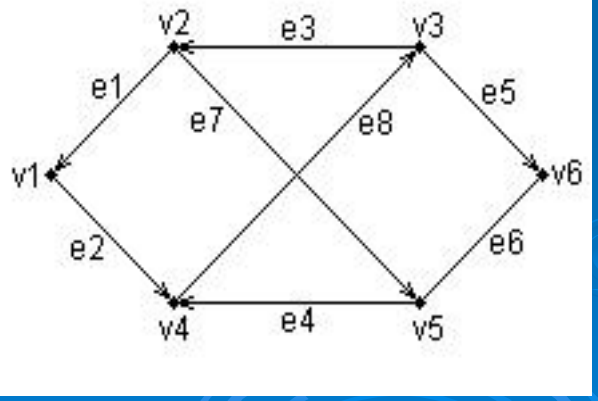
G = (V, E),

V = {v1, v2, ..., vn}

E = {e1, e2, ..., em}

(vi, vj) $\in$ PG – отношение связности





Исторические предшественники: *семантические сети*

Семантические сети получили свое развитие в рамках разработки специальных языков и графических средств для представления знаний человека (проблематика искусственного интеллекта).

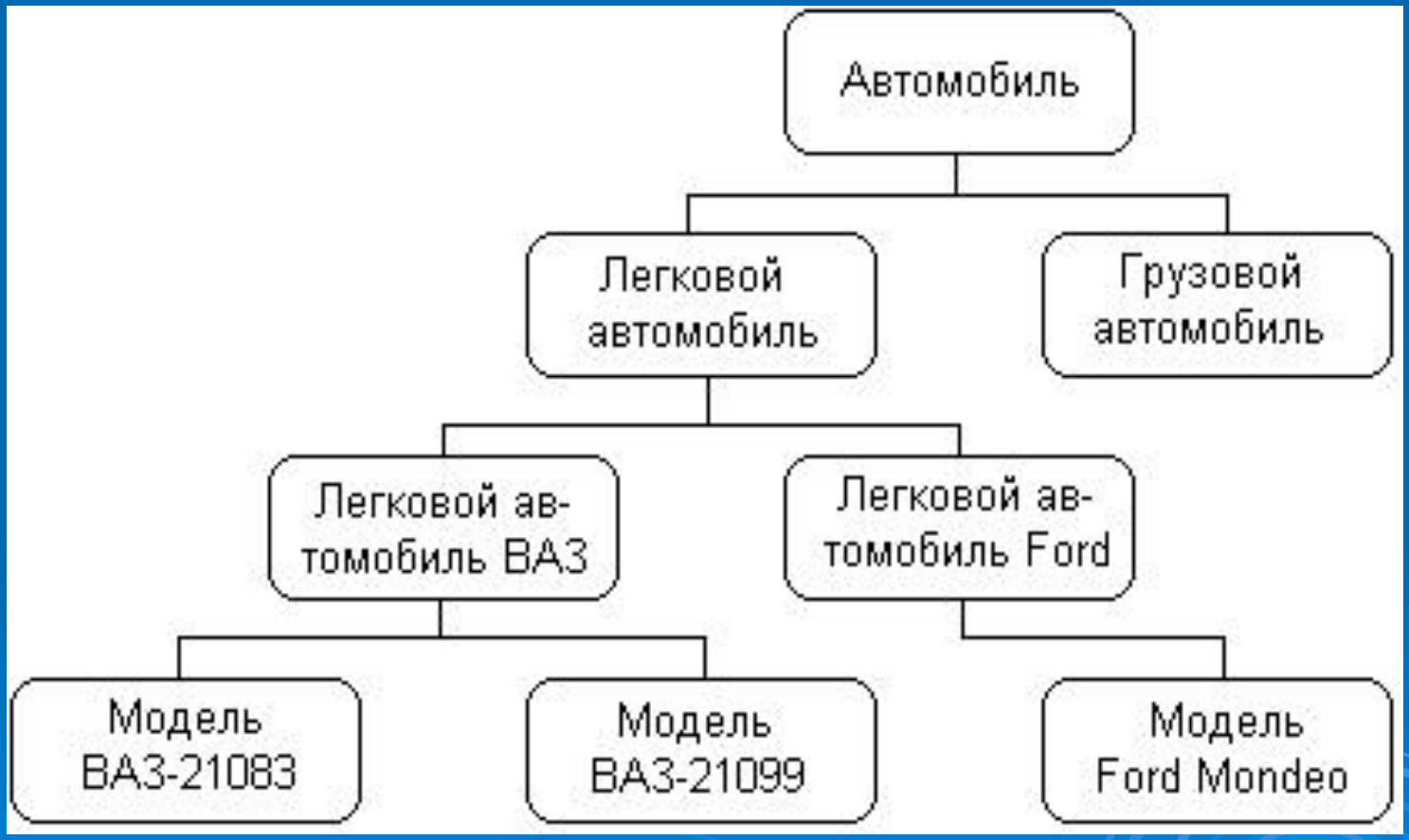
Семантическая сеть - это граф G=(V, E), в котором множество вершин V и множество ребер E разделены на отдельные типы, обладающие специальной семантикой, характерной для той или иной предметной области. Например, множество вершин может соответствовать объектам рассматриваемой предметной области, а множество ребер - различным видам связей между объектами.

Важная особенность графической нотации семантических сетей - наличие специальных обозначений для представления разных типов вершин и ребер графа.

Вершины графа могут изображаться прямоугольниками, овалами, окружностями и другими геометрическими фигурами в зависимости от типа представляемых ими объектов.

Ребра графа изображаются линиями различных видов со стрелками или без них, имеющими специальные обозначения или украшения в виде условных значков.

Пример: ***Фрагмент семантической сети*** для представления иерархии классов предметной области «Автомобили»



Диаграммы структурного анализа систем

Структурный анализ – это метод исследования системы, который начинается с наиболее общего описания с последующей детализацией отдельных аспектов ее поведения и функционирования.

Модель системы представляется иерархической структурой, которая отражает различные уровни абстракции с ограниченным числом компонентов на каждом из уровней.

При выполнении структурного анализа программных систем используют ***три основных вида диаграмм***:

- ***SADT*** (*Structured Analysis and Design Technique*) - диаграммы ***функционального моделирования***
- ***ERD*** (*Entity-Relationship Diagrams*) - диаграммы "***сущность-связь***"
- ***DFD*** (*Data Flow Diagrams*) - диаграммы ***потоков данных***

SADT - Методология функционального моделирования

Методология SADT представляет собой совокупность методов, правил и процедур, предназначенных для построения функциональной модели системы какой-либо предметной области.

Модель SADT позволяет наглядно представить структуру процессов функционирования системы и ее отдельных подсистем, т. е. отображает последователь-ность выполняемых действий и связи между этими действиями.

Базовыми компонентами модели SADT являются:

- **деятельности** (activity)
- **стрелки** (arrow).

Компоненты SADT-диаграмм: «***Деятельности***»

Деятельность (или *процесс*) представляет собой некоторое действие или набор действий, которые имеют фиксированную цель и приводят к некоторому конечному результату.

SADT-модель поддерживает различные виды деятельностей системы, их описание и взаимодействие с другими процессами.

На SADT-диаграммах деятельность изображается прямоугольником, который называется *блоком,* внутри которого записывается содержательное наименование соответствующего процесса.

Компоненты SADT-диаграмм: «***Стрелки***»

Стрелка служит для обозначения некоторого носителя или воздействия, которые обеспечивают перенос информации (или объектов) от одной деятельности к другой.

Модель SADT поддерживает четыре вида (*ICOM-роли*) стрелок. *ICOM* - первые буквы от названий соответствующих ролей:

- I (Input) – вход, т. е. все, что поступает в процесс или потребляется процессом.
- C (Control) – управление или ограничения на выполнение операций процесса.
- O (Output) – выход или результат процесса.
- M (Mechanism) – механизм, который используется для выполнения процесса.

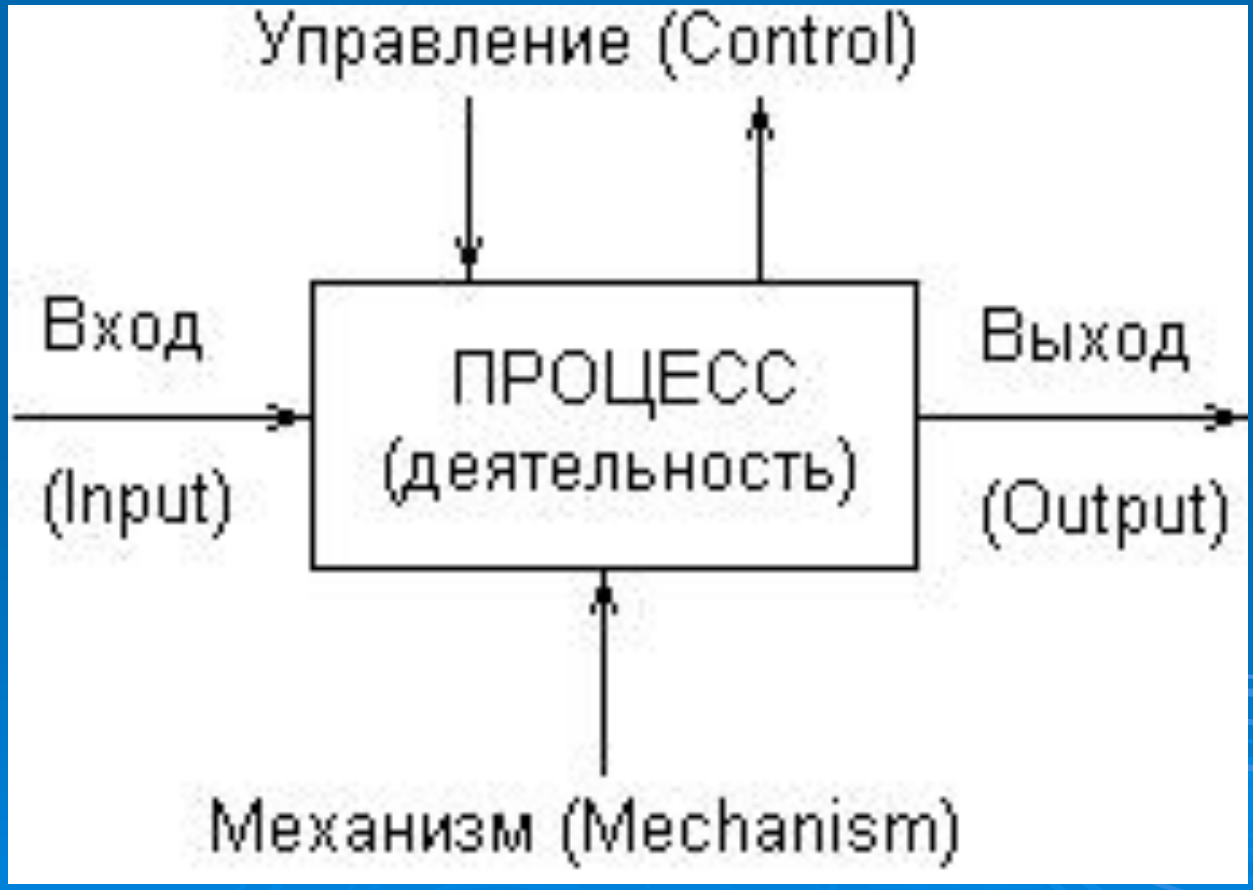
Место соединения *стрелки* с *блоком* (*деятельностью)* определяет тип интерфейса.

Техника построения SADT-диаграмм

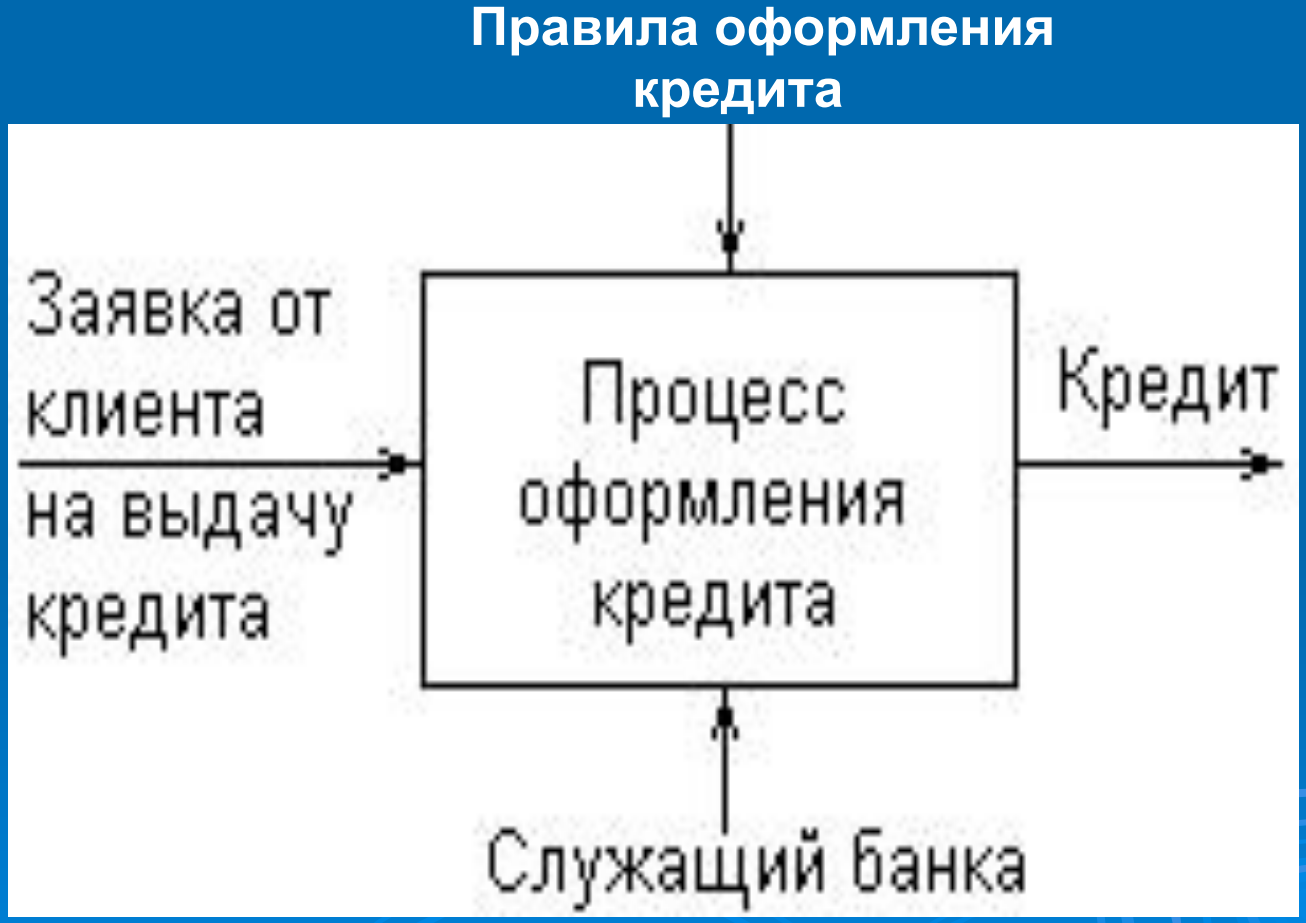
Техника построения SADT-диаграмм однозначно определяет правила изображения стрелок каждого вида ICOM:

- (I) - входная информация, обрабатываемая процессом, изображается *с левой стороны блока*;
- (C) - управляющая информация *входит в блок (или выходит из блока) сверху*;
- (O) - результаты обработки входной информации представляются как выходы процесса и показываются *с правой стороны блока*;
- (M) - механизм представляется стрелкой, которая *входит в блок процесса снизу*. В качестве механизма может выступать человек или компонент автоматизированной системы, которые реализуют данный процесс

Обозначение процессов и ICOM-стрелок на SADT-диаграммах



SADT-диаграмма системы оформления банковского кредита (верхний уровень иерархии)



Модель SADT – иерархическая система диаграмм

Важная особенность методологии SADT - последовательная детализация модели системы: построение модели начинается с представления всей системы в виде простейшей диаграммы, состоящей из одного блока процесса и стрелок ICOM, служащих для изображения основных видов взаимодействия с внешними по отношению к системе объектами.

Исходный процесс, представляющий всю систему как единое целое, подлежит дальнейшей декомпозиции на следующем этапе проектирования.

Частные процессы, полученные в результате декомпозиции исходного процесса, также декомпозируются – и так до некоторого разумного предела, пока уровень детальности рассмотрения свойств процесса не окажется достаточным для его реализации.

В конечном итоге модель SADT представляет собой множество иерархически взаимосвязанных диаграмм, которые представляют сложную систему отдельными ее составными частями.

Очевидно, что на следующем этапе моделирования системы выдачи банковского кредита исходный процесс будет декомпозирован, и в нем будут выделены частные процессы: например, процесс "*Проверка платежеспособности клиента*", процесс "*Запрос кредитной истории клиента*" и процесс "*Оформление кредитного договора*".

ERD - Диаграммы "*Сущность – Связь*"

Диаграммы "сущность-связь" (ERD) предназначены для графического представления моделей данных программных систем и используется на начальных стадиях проектирования баз данных – при разработке концептуальных моделей данных.

Основными компонентами ER-модели являются:

- ***сущность*** (entity)
- ***связь*** (relationship)
- ***атрибуты*** (свойства) сущностей и связей.

Сущность - абстракция *множества реальных объектов* (или процессов) предметной области, информация о которых *существенна* в контексте проектируемой программной системы.

- Каждый реальный объект может быть представлен экземпляром одной (и только одной) сущности.
- Все экземпляры одной сущности имеют *одинаковый набор* (но не значения !) атрибутов.
- Среди экземпляров одной сущности *не должно быть дубликатов* – то есть экземпляров с *одинаковыми значениями всех атрибутов*.

И сущности, и их атрибуты – это именованные объекты модели, каждому из которых должно быть присвоено уникальное имя.

Обозначения сущностей на ER-диаграммах

Для обозначения сущностей используют набор стандартных графических элементов – прямоугольников с дополнительными элементами оформления, внутри которых записывают имена сущностей.

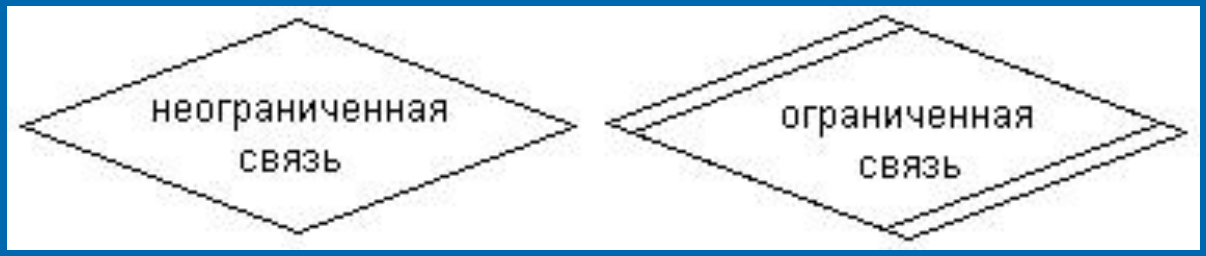
Обозначения связей на ER-диаграммах

Связь - это абстракция некоторого отношения или ассоциации между реальными объектами предметной области, существенного в контексте проектируемой программной системы. Примерами связей могут являться родственные отношения типа "*отец-сын*", производственные отношения типа "*начальник-подчиненный*", отношение вхождения типа "*целое-часть*", отношения типа "*иметь в собственности*", "*обладать свойством*", "*изучать*", "*работать в*" и т.д.

Связь, так же, как и сущность, является именованным объектом ER-модели и тоже может иметь свои собственные атрибуты (свойства связи). Заметим, что в отличие от сущности, связь может и не иметь собственных атрибутов.

Связь на ER-диаграммах обозначается линией, связывающей сущности, в разрыве которой помещается ромб, внутри которого записывается имя связи.

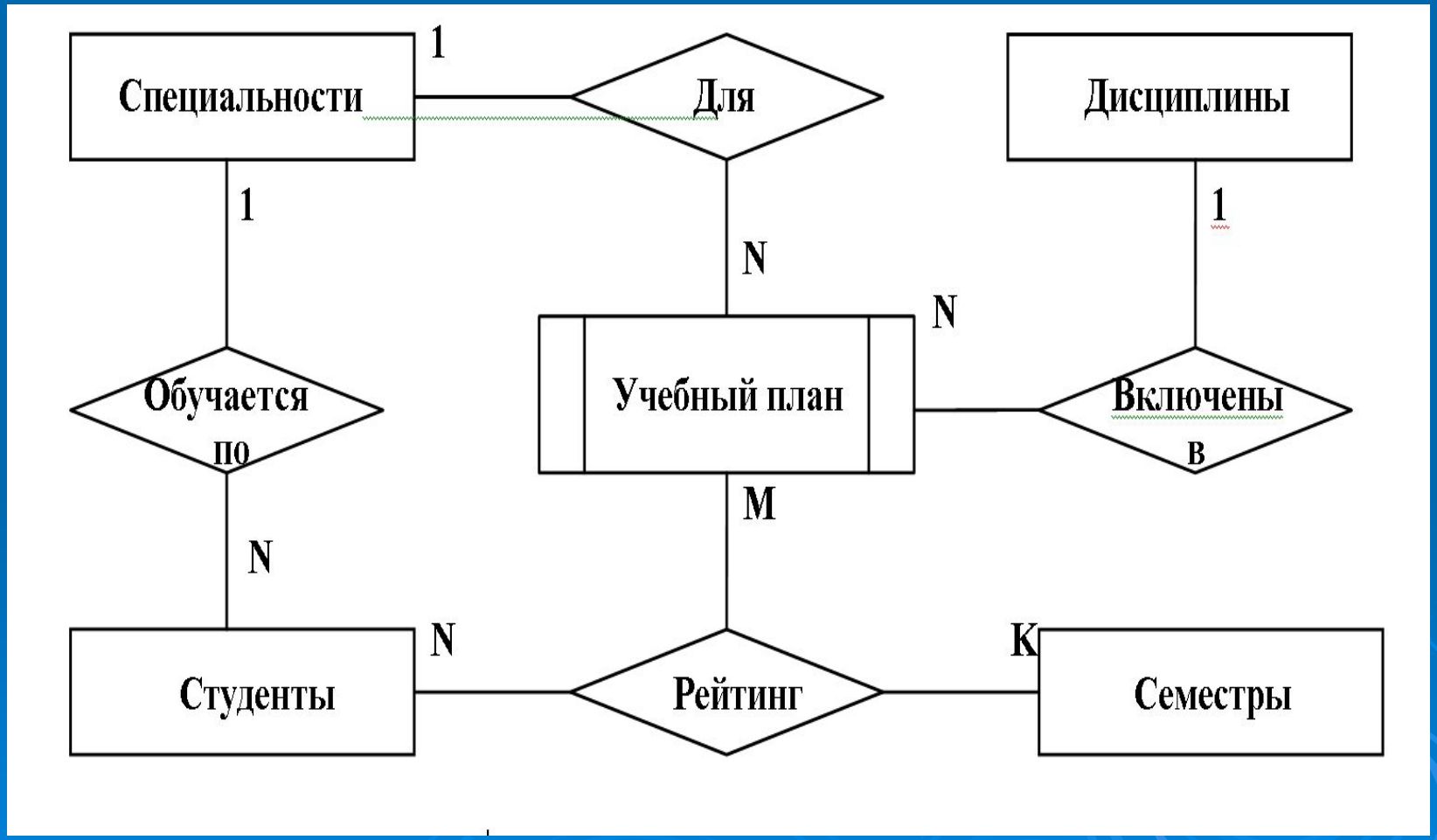
Обозначения связей на ER-диаграммах



Связь отображает не только семантику отношения между сущностями (указанную именем связи), но и дополнительные ***характеристики связи***:

- ***арность***, определяемую количеством сущностей, участвующих в связи;
- ***порядок*** (или *кратность*), определяемый количеством экземпляров сущностей, участвующих в связи;
- дополнительно на диаграмме может быть отражен *аспект* ***обязательности*** участия в связи экземпляров связанных сущностей.

Пример ER-диаграммы:
фрагмент упрощенной концептуальной модели данных
«***Система учета результатов учебного процесса***»



Комментарии

Между сущностями *"Студенты"* и "*Специальности"* определена ***бинарная связь*** "*Обучается по*". Связь имеет ***порядок (кратность) N:1***, что характеризует принятое в ВУЗе правило, согласно которому на одной специальности могут одновременно обучаться несколько студентов, но один студент не может обучаться одновременно более, чем по одной специальности.

Сущность "*Учебный план*" участвует в двух ***бинарных связях порядка N:1*** с сущностями "*Специальности*" и "*Дисциплины*". Каждый экземпляр этой сущности представляет какую-то одну дисциплину учебного плана какой-то одной специальности. Если дисциплина (например, "История отечества", "Физическая культура" или "Информатика") представлена в учебных планах десяти специальностей, то сущность "*Учебный план*" будет содержать десять экземпляров, связанных с одним и тем же экземпляром сущности "*Дисциплины*" и с десятью разными экземплярами сущности "*Специальности*".

"*Учебный план*" - это ***зависимая*** (или ***слабая***) ***сущность***, так как ее экземпляры не могут существовать вне связей с экземплярами двух других сущностей. Возможные атрибуты этой сущности – порядковый номер дисциплины в соответствующем учебном плане, трудоемкость дисциплины в зачетных единицах, объем дисциплины в часах учебных занятий, наличие экзаменов, зачетов, курсовых работ.

Каждый экземпляр ***тернарной связи*** "*Рейтинг*" ***порядка M:N:K*** представляет факт завершения изучения одним студентом одной дисциплины учебного плана его специальности в одном семестре. Эта связь, в отличие от других связей рассматриваемого примера, имеет собственные атрибуты, например, *дата сдачи экзамена* и *итоговый рейтинговый показатель* студента по дисциплине в семестре.

Моделирование системы на последующих стадиях проекта требует более детального её описания, которое должно содержать информацию о поведении или функционировании ее компонентов. Для этих целей возможностей ER-диаграмм оказывается явно недостаточно, и используется другой тип диаграмм, получивших название *диаграмм потоков данных*.

DFD - *Диаграммы потоков данных*

DFD-модель системы – это информационная модель, основными компонентами которой являются потоки данных, которые переносят информацию от одной подсистемы (процесса) к другой подсистеме или процессу.

При этом каждая подсистема/процесс выполняет определенные преобразования входного потока данных и передает результаты обработки информации другим подсистемам/процессам в виде выходных потоков.

Основные компоненты DF-диаграмм:

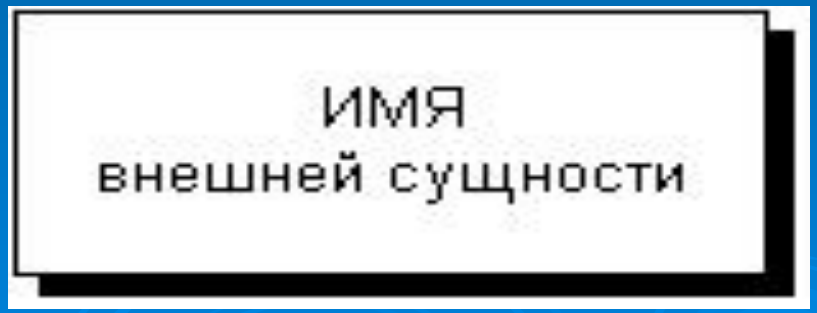
- ***внешние сущности***
- ***системы/подсистемы***
- ***процессы***
- ***накопители (хранилища) данных***
- ***потоки данных***

Компоненты DFD-модели: *Внешние сущности*

Внешняя сущность (*терминатор*) представляет собой любой материальный объект, находящийся за пределами границ рассматриваемой системы, который может выступать в качестве источника информации, обрабатываемой системой, или приемника информации, поступающей от системы.

Примерами внешних сущностей могут служить клиенты организации, заказчики, поставщики, а также датчики системы охранной сигнализации, контроллеры систем управления техническими объектами.

Внешняя сущность обозначается на диаграммах ***прямоугольником с тенью***, внутри которого указывается ее имя.

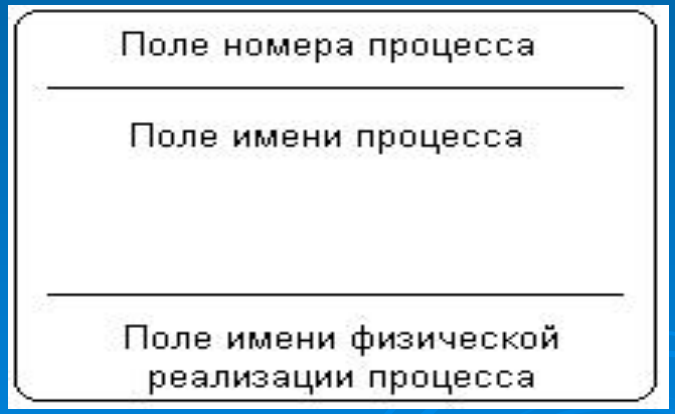


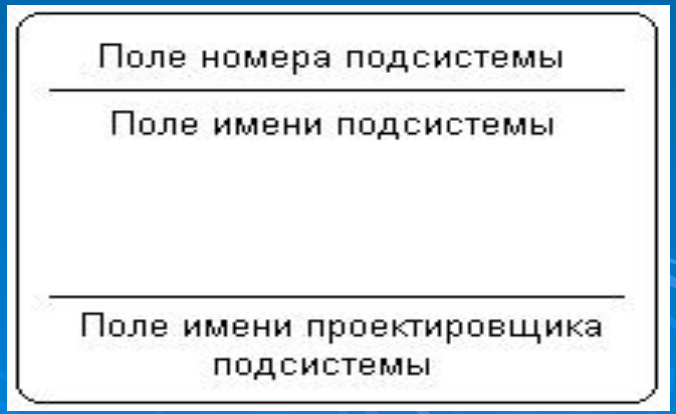
Компоненты DFD-модели: *Процессы и Подсистемы*

Процесс представляет собой совокупность операций по преобразованию входных потоков данных в выходные в соответствии с определенным алгоритмом.

Подсистема – это элемент контекстной диаграммы, представляющей систему в виде иерархической схемы.

Процессы и Подсистемы на диаграмме изображаются прямоугольником с закругленными вершинами, разделенным на три поля.





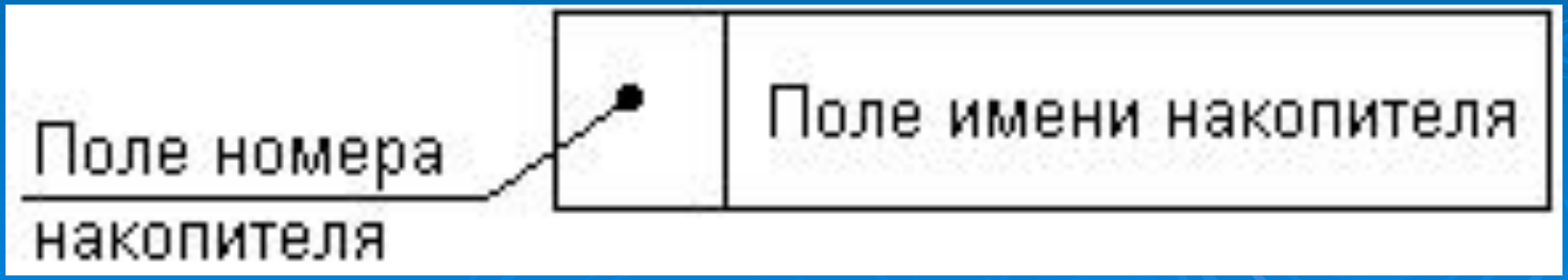
Компоненты DFD-модели: *Накопители*

Накопитель (хранилище) данных представляет собой абстрактное устройство или способ хранения информации, перемещаемой между процессами.

Предполагается, что данные можно в любой момент поместить в накопитель и через некоторое время извлечь.

Накопитель данных на диаграмме потоков данных *изображается прямоугольником с двумя полями.*

- ***Номер*** - должен начинаться с буквы "D" (от англ. Data – данные)
- ***Имя*** – должно характеризовать способ хранения информации.



Компоненты DFD-модели: *Поток данных*

Поток данных представляет информацию, передаваемую через некоторое соединение от источника к приемнику.

Поток данных на диаграмме изображается линией со стрелкой на одном из ее концов, показывающей направление потока.

Поток данных имеет свое собственное имя – содержательную характеристику передаваемой потоком информации.

На следующем слайде приведен пример DF-диаграммы упрощенной модели процесса получения клиентом банка наличных денег по банковской карте.

Внешними сущностями являются клиент банка и, возможно, служащий банка, который контролирует процесс обслуживания клиентов.

Накопителем данных может быть база данных о состоянии счетов клиентов банка.

Отдельные потоки данных отражают характер передаваемой информации, необходимой для обслуживания клиента банка.

Пример DFD-модели

Заключение

1. Исследования и программные разработки в области структурного анализа сложных систем позволили выработать базовые концепции визуального моделирования, создать графическую нотацию для отображения моделей и апробировать ее в CASE-системах.
2. Возможности средств визуального моделирования, разработанных в поддержку методологии структурного системного анализа, оказались сильно ограниченными при переходе к объектно-ориентированным технологиям разработки сложных систем.
3. Основные недостатки этой методологии и поддерживающих ее графических нотаций:
   - отсутствие средств представления сложных алгоритмов обработки данных;
   - недостаточная развитость средств отображения временных характеристик процессов и потоков данных;
   - отсутствие явных средств объектно-ориентированного представления моделей сложных систем.
4. Многие идеи визуального моделирования и элементы графических нотаций моделей структурного анализа были в последствии эффективно использованы при разработке UML - унифицированного языка моделирования, поддерживающего методологию объектно-ориентированного анализа и проектирования сложных систем.

Контрольные вопросы и задания

1. Для чего используют модели сложных систем при их проектировании? Перечислите основные задачи, решаемые проектировщиками систем с помощью моделирования.
2. Какие цели преследует визуальное моделирование систем ?
3. В какой области знаний используются диаграммы Венна ? Приведите пример использования графовых моделей для описания системы.
4. Расшифруйте сокращенные названия диаграмм *SADT, ERD, DFD.*

   Для чего используются диаграммы перечисленных типов ?
5. Какие задачи позволяет решать методология *SADT?* Для чего используются элементы модели *activity* и *arrow*? Что такое ICAM?
6. Опишите *SADT*-диаграммами процесс выдачи книг читателю абонемента публичной библиотеки (на двух уровнях декомпозиции).
7. Перечислите компоненты ER-модели, дайте определения всем компонентам.
8. Разработайте ER-модель данных для учета книжного фонда публичной библиотеки (*вариант:* для читального зала научной литературы).
9. Перечислите компоненты DF-диаграмм, приведите примеры.
10. Разработайте DF-диаграмму процесса выдачи книг читателю абонемента публичной библиотеки.