# Лабораторная работа № 1

Тема: *разработка моделей информационных систем по методологии IDEF0.*
Цель: *изучить методологию моделирования IDEF0, освоить CASE-средство AllFusion Process Modeller для построения диаграмм IDEF0, научиться строить модели информационных систем, используя данную методологию*

## Краткая теория

Методология функционального моделирования IDEF0 – это технология описания системы в целом как множества взаимозависимых действий, или функций. Важно отметить функциональную направленность IDEF0 – функции системы исследуются независимо от объектов, которые обеспечивают их выполнение. "Функциональная" точка зрения позволяет четко отделить аспекты назначения системы от аспектов ее физической реализации. Наиболее часто IDEF0 применяется как технология исследования и проектирования систем на логическом уровне. По этой причине он, как правило, используется на ранних этапах разработки проекта, до IDEF3 моделирования для сбора данных и моделирования процесса "как есть". Результаты IDEF0 анализа могут применяться при проведении проектирования с использованием моделей IDEF3 и диаграмм потоков данных.

IDEF0 сочетает в себе небольшую по объему графическую нотацию (она содержит только два обозначения: блоки и стрелки) со строгими и четко определенными рекомендациями, в совокупности предназначенными для построения качественной и понятной модели системы.

Перед непосредственным построением модели IDEF0 осуществляется выполнение следующих действий:

1. Определение назначения модели – набора вопросов, на которые должна отвечать модель. Должны быть определены границы моделирования, предназначенные для обозначения ширины охвата предметной области и глубины детализации. Границы моделирования являются логическим продолжением уже определенного назначения модели.
2. Указание предполагаемой целевой аудитории, для нужд которой создается модель. От выбора целевой аудитории зависит уровень детализации, с которым должна создаваться модель. Перед построением модели необходимо иметь представление о том, какие сведения о предмете моделирования уже известны, какие дополнительные материалы и (или) техническая документация для понимания модели могут быть необходимы целевой аудитории, какие язык и стиль изложения являются наиболее подходящими.
3. Определение точки зрения, под которой понимается перспектива, с которой наблюдается система при построении модели. Точка зрения выбирается таким образом, чтобы учесть уже обозначенные границы моделирования и назначение модели. Точка зрения остается неизменной для всех элементов модели. При необходимости могут быть созданы и другие модели, отображающие систему с других точек зрения.

Действие, обычно в IDEF0 называемое функцией, обрабатывает или переводит входные параметры (сырье, информацию и т.п.) в выходные. Поскольку модели IDEF0 представляют систему как множество иерархических (вложенных) функций, в первую очередь должна быть определена функция, описывающая систему в целом – контекстная функция (или как ее еще называют – контекстная диаграмма). Функции изображаются на диаграммах как поименованные прямоугольники, или функциональные блоки. Имена функций в IDEF0 задаются с использованием глаголов или отглагольных существительных с поясняющим текстом. Любой блок может быть декомпозирован на составляющие его блоки, образуя тем самым иерархию функциональных блоков модели IDEF0.

Описание любого блока должно, как минимум, включать в себя описание объектов, которые блок создает в результате своей работы ("выхода"), и объектов, которые блок потребляет или преобразует ("вход").

В IDEF0 также моделируются управление и механизмы исполнения. Под управлением понимаются объекты, воздействующие на способ, которым блок преобразует вход в выход. Механизм исполнения – объекты, которые непосредственно выполняют преобразование входа в выход, но не потребляются при этом сами по себе. Для отображения категорий информации, присутствующих на диаграммах IDEF0, существует аббревиатура ICOM, отображающая четыре возможных типа стрелок:

- I (Input) – вход – нечто, что потребляется в ходе выполнения процесса;
- C (Control) – управление – ограничения и инструкции, влияющие на ход выполнения процесса;
- O (Output) – выход – нечто, являющееся результатом выполнения процесса;
- M (Mechanism) – исполняющий механизм – нечто, что используется для выполнения процесса, но не потребляется само по себе.

На рисунке 1 показаны 4 возможных типа стрелок в IDEF0, каждый из типов соединяется со своей стороной функционального блока.

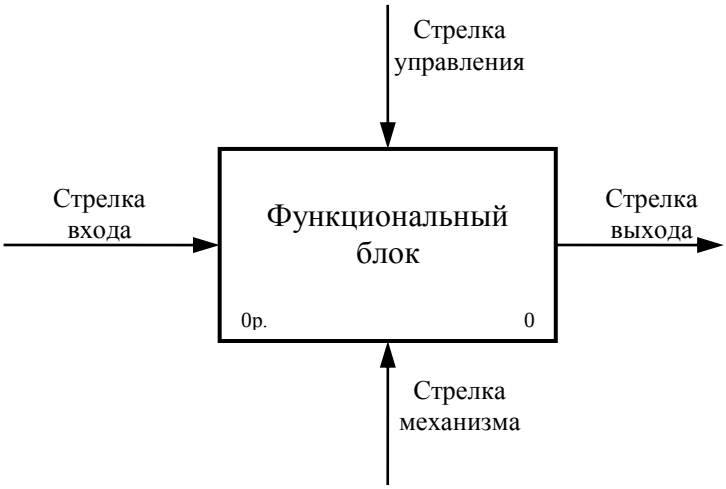


Рисунок 1 – Представление функционального блока на диаграмме IDEF0

Для названия стрелок, как правило, употребляются имена существительные. Стрелки могут представлять собой информацию или объекты, передаваемые между функциональными блоками IDEF0.

Вход представляет собой сырье, или информацию, потребляемую или преобразуемую функциональным блоком для производства выхода. Стрелки входа всегда направлены в левую сторону прямоугольника, обозначающего в IDEF0 функциональный блок. Наличие входных стрелок на диаграмме не является обязательным, так как возможно, что некоторые блоки ничего не преобразуют и не изменяют.

Стрелки управления отвечают за регулирование того, как и когда выполняется функциональный блок, и, если он выполняется, какой выход получается в результате его выполнения. Так как управление контролирует поведение функционального блока для обеспечения создания желаемого выхода, каждый функциональный блок должен иметь, как минимум, одну стрелку управления. Стрелки управления всегда входят в функциональный блок сверху. Влияя на работу блока, управление непосредственно не потребляется и не трансформируется в результате. Управление можно рассматривать как специфический вид входа. В случаях, когда неясно, относить ли стрелку к входу или к управлению, предпочтительно относить ее к управлению до момента, пока неясность не будет разрешена.

Выход – это продукция или информация, получаемая в результате работы функционального блока. Каждый блок должен иметь, как минимум, один выход. Действие, которое не производит никакого четко определяемого выхода, не должно моделироваться вообще. При моделировании непроизводственных предметных областей выходами, как правило, являются данные, в каком-либо виде обрабатываемые функциональным блоком. В этом случае важно, чтобы названия стрелок входа и выхода были достаточно различимы по

своему смыслу. Например, блок "Модификация базы данных" может иметь стрелку "База данных" как на входе, так и на выходе. В такой ситуации входящую стрелку можно назвать "Исходная база данных", а исходящую — "Обновленная база данных".

Механизмы являются ресурсом, который непосредственно исполняет моделируемое действие. С помощью механизмов исполнения могут моделироваться: ключевой персонал, техника и (или) оборудование. Стрелки механизма исполнения могут отсутствовать в случае, если оказывается, что они не являются необходимыми для достижения поставленной цели моделирования.

В методологии IDEF0 существует пять видов соединения функциональных блоков между собой:

- выход – вход. Применяется, когда один из блоков должен полностью завершить работу перед началом работы другого блока;
- выход – управление. Отражает ситуацию преобладания одного блока над другим, когда один блок управляет работой другого;
- выход – механизм исполнения. Встречаются реже и отражают ситуацию, когда выход одного функционального блока применяется в качестве оборудования для работы другого блока;
- выход – обратная связь на управление. Применяются в случаях, когда зависимые блоки формируют обратные связи для управляющих ими блоков;
- выход – обратная связь на вход. Обычно применяется для описания циклов повторной обработки чего-либо.

Методология IDEF0 предусматривает как разбиение, так и соединение стрелок на диаграмме. Разбитые на несколько частей стрелки могут иметь наименования, отличающиеся от наименования исходной стрелки. Исходная и разбитые (или объединенные) стрелки в совокупности называются связанными. Такая техника обычно применяется для того, чтобы отразить использование в процессе только части сырья или информации, обозначаемых исходной стрелкой (рисунок 2). Аналогичный подход применяется и к объединяемым стрелкам.

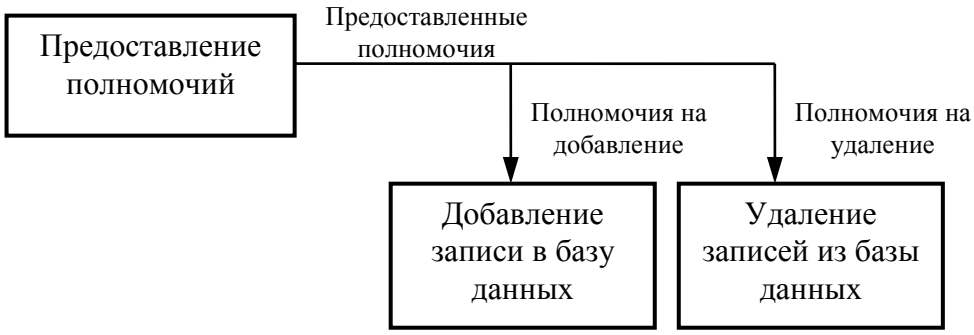


Рисунок 2 – Разбитая на две части и переименованная стрелка

Понятие связанных стрелок используется для управления уровнем детализации диаграмм. Если одна из стрелок диаграммы отсутствует на родительской диаграмме (например, ввиду своей несущественности для родительского уровня) и не связана с другими стрелками той же диаграммы, точка входа этой стрелки на диаграмму или выхода с нее обозначается туннелем. Туннель в данном случае используется как альтернатива загромождению родительских диаграмм помещением на них несущественных для их уровня стрелок. Кроме того, туннели применяются для отражения ситуации, когда стрелка, присутствующая на родительской диаграмме, отсутствует в диаграмме декомпозиции соответствующего блока. Графическое изображение туннелей приведено на рисунке 3.

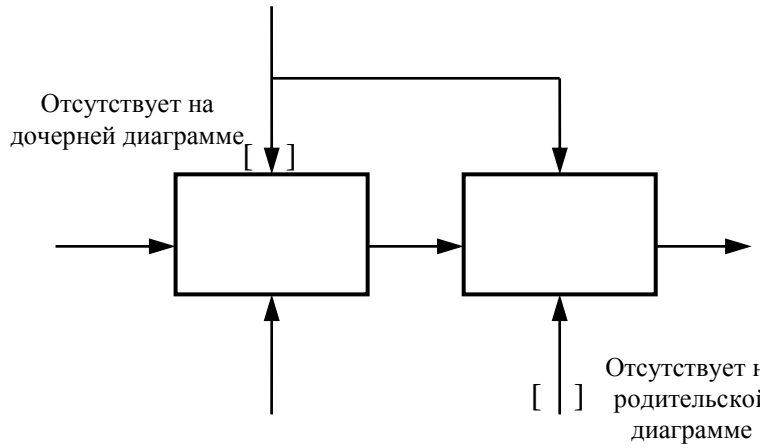


Рисунок 3 – Применение туннелей на диаграммах IDEF0

Построение диаграмм IDEF0 начинается с представления всей системы в виде простейшего компонента – одного блока и дуг, изображающих интерфейсы с функциями вне системы. Поскольку единственный блок отражает систему как единое целое, имя, указанное в блоке, является общим. Это верно и для интерфейсных дуг – они также соответствуют полному набору внешних интерфейсов системы в целом.

Затем блок, который представляет систему в качестве единого модуля, детализируется на другой диаграмме с помощью нескольких блоков, соединенных интерфейсными дугами. Эти блоки определяют основные подфункции исходной функции. Данная декомпозиция выявляет полный набор подфункций, каждая из которых показана как блок, границы которого определены интерфейсными дугами. Каждая из этих подфункций может быть декомпозирована подобным образом в целях большей детализации.

Таким образом, модель IDEF0 представляет собой серию диаграмм с сопроводительной документацией, разбивающих сложный объект на составные части, которые изображены в виде блоков. Детали каждого из основных блоков показаны в виде блоков на других диаграммах. Каждая детальная диаграмма является декомпозицией блока из диаграммы предыдущего уровня. На каждом шаге декомпозиции диаграмма предыдущего уровня называется родительской для более детальной диаграммы.

Синтаксис диаграмм определяется следующими правилами:

- диаграммы содержат блоки и дуги;
- блоки представляют функции;
- блоки имеют доминирование (выражающееся в их ступенчатом расположении, причем доминирующий блок располагается в верхнем левом углу диаграммы);
- дуги изображают наборы объектов, передаваемых между блоками;
- дуги изображают взаимосвязи между блоками.

Дуги, входящие в блок и выходящие из него на диаграмме верхнего уровня, являются точно теми же самыми, что и дуги, входящие в диаграмму нижнего уровня и выходящие из нее, потому что блок и диаграмма изображают одну и ту же часть системы. Пример диаграмм образующих иерархию приведен на рисунке 4.

Каждый блок на диаграмме имеет свой номер. Блок любой диаграммы может быть далее описан диаграммой нижнего уровня, которая, в свою очередь, может быть далее детализирована с помощью необходимого числа диаграмм. Таким образом, формируется иерархия диаграмм. Для того чтобы указать положение любой диаграммы или блока в иерархии, используются номера диаграмм. Например, A21 является диаграммой, которая детализирует блок A21 на диаграмме A2. Аналогично, диаграмма A2 детализирует блок A2 на диаграмме A0, которая является самой верхней диаграммой модели. На рисунке 4.5 показан пример дерева диаграмм.

При построении иерархии диаграмм используются следующие стратегии декомпозиции:

– Функциональная декомпозиция – декомпозиция в соответствии с функциями, которые выполняют люди или организация. Может оказаться полезной стратегией для создания системы описаний, фиксирующей взаимодействие между людьми в процессе их работы. Очень часто, взаимосвязи между функциями весьма многочисленны и сложны, поэтому рекомендуется использовать эту стратегию только в начале работы над моделью системы.
– Декомпозиция в соответствии с известными стабильными подсистемами – приводит к созданию набора моделей, по одной модели на каждую подсистему или важный компонент. Затем для описания всей системы должна быть построена составная модель, объединяющая все отдельные модели. Рекомендуется использовать разложение на подсистемы, только когда разделение на основные части системы не меняется. Нестабильность границ подсистем быстро обесценит как отдельные модели, так и их объединение.
– Декомпозиция по физическому процессу – выделение функциональных стадий, этапов завершения или шагов выполнения. Хотя эта стратегия полезна при описании существующих процессов, результатом ее часто может стать слишком последовательное описание системы, которое не будет в полной мере учитывать ограничения, диктуемые функциями друг другу При этом может оказаться скрытой последовательность управления. Эта стратегия рекомендуется, только если целью модели является описание физического процесса как такового или только в крайнем случае, когда неясно, как действовать.

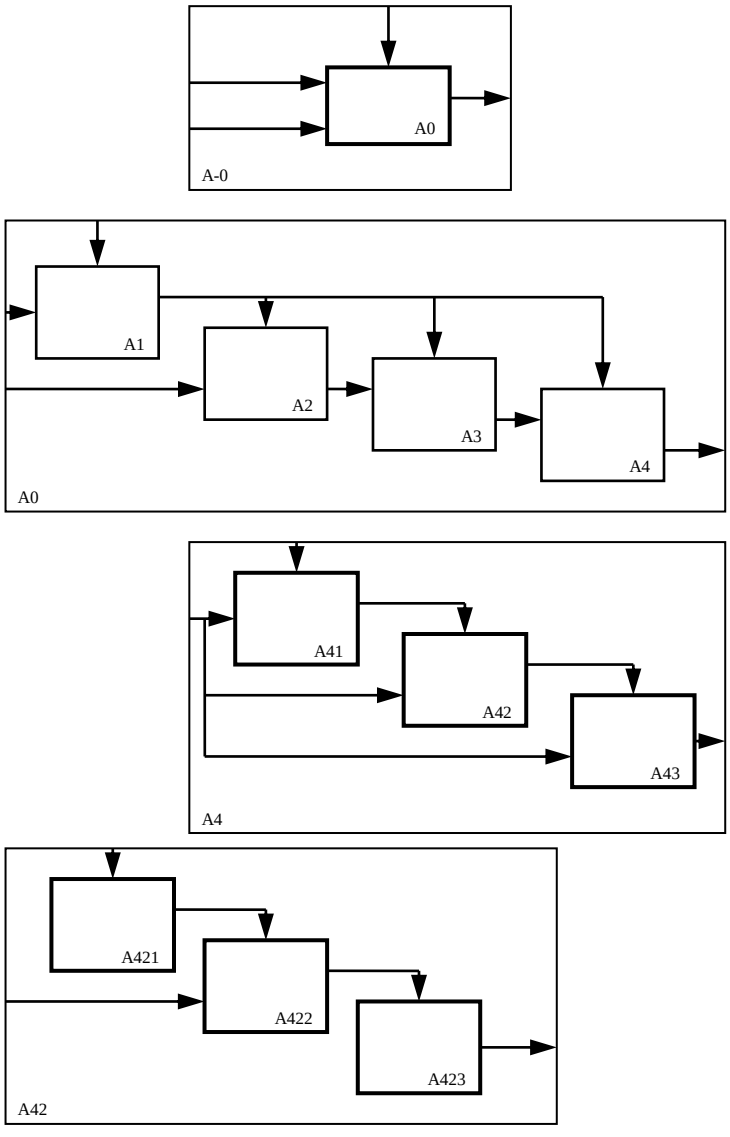


Рисунок 4 – Пример иерархии диаграмм

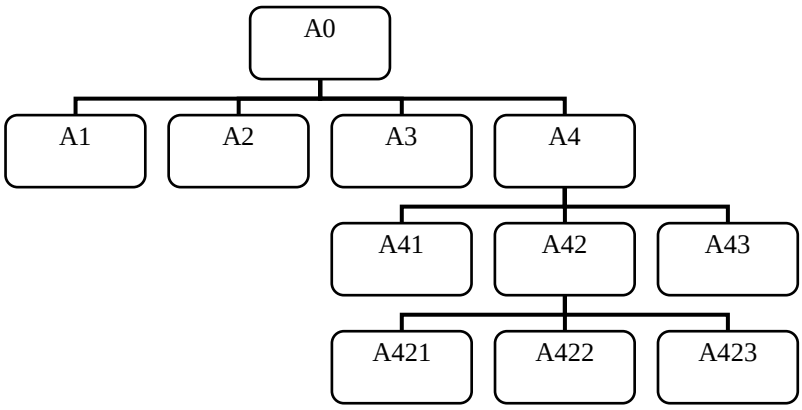


Рисунок 5 – Пример дерева диаграмм

Одна из наиболее частых проблем, возникающих в процессе построения моделей IDEF0, – когда же следует завершить построение конкретной модели? Однозначный ответ на этот вопрос дать невозможно. Только длительная практика позволит приобрести знания, необходимые для принятия правильного решения об окончании моделирования.

Обычно модель строится слоями, большинство из которых не являются глубокими. Чаще всего ограничиваются тремя уровнями. Опыт показывает, что, как правило, создаются несколько диаграмм второго и третьего уровней только для того, чтобы убедиться, что для достижения цели уже первый уровень содержит достаточно информации.

Если модель IDEF0 декомпозируется на глубину 5-6 уровней, то в этом случае на такую глубину декомпозируется обычно один из блоков диаграммы А0. Функции, которые требуют такого уровня детализации, часто очень важны, и их детальное описание дает ключ к секретам работы всей системы. Но хотя важные функции могут нуждаться в глубокой детализации, таких функций при создании одной модели насчитывается, как правило, немного. Модели, обладающие такими функциями, имеют обычно форму зонтика с широким тонким куполом и длинной ручкой, на которой происходит детализация.

Большие аналитические проекты обычно разбиваются на несколько отдельных более мелких проектов, каждый из которых создает модель одного конкретного аспекта всей проблемы. Поэтому вместо одной гигантской модели создается сеть из нескольких небольших моделей. Исключительно большие проекты могут привести к созданию высококачественной модели, состоящей из тысяч блоков, но это случается редко. Большинство систем не требует для адекватного описания моделей такой величины.

Рекомендуется прекращать моделирование, когда уровень детализации модели удовлетворяет ее цели. Опыт показал, что для отдельной модели, которая создается независимо от какой-либо другой модели, декомпозиция одного из ее блоков должна прекращаться, если:

- блок содержит достаточно деталей. Одна из типичных ситуаций, встречающихся в конце моделирования, – это блок, который описывает систему с нужным уровнем подробности. Если блок позволяет ответить на один или несколько вопросов, составляющих цель модели, то дальнейшая декомпозиция может не понадобиться;
- необходимо изменить уровень абстракции, чтобы достичь большей детализации блока. Блоки подвергаются декомпозиции, если они недостаточно детализированы для удовлетворения цели модели. Но иногда при декомпозиции блока выясняется, что диаграмма начинает описывать, как функционирует блок, вместо описания того, что блок делает. В этом случае происходит изменение уровня абстракции – изменение сути того, что должна представлять модель. В IDEF0 изменение уровня абстракции часто означает выход за пределы цели модели и, следовательно, это указывает на прекращение декомпозиции;
- необходимо изменить точку зрения, чтобы детализировать блок. Изменение точки зрения происходит примерно так же, как изменение уровня абстракции. Это чаще всего

характерно для ситуаций, когда точку зрения модели нельзя использовать для декомпозиции конкретного блока, т. е. этот блок можно декомпозировать, только если посмотреть на него с другой позиции. Об этом может свидетельствовать заметное изменение терминологии;

- блок очень похож на другой блок той же модели или на блок другой модели. Иногда встречается блок, чрезвычайно похожий на другой блок модели. Два блока похожи, если они выполняют примерно одну и ту же функцию и имеют почти одинаковые по типу и количеству входы, управления и выходы. Если второй блок уже декомпозирован, то разумно отложить декомпозицию и тщательно сравнить два блока. Если нужны ничтожные изменения для совпадения первого блока со вторым, то внесение этих изменений сократит усилия на декомпозицию и улучшит модульность модели;
- блок представляет тривиальную функцию. Тривиальная функция – это такая функция, понимание которой не требует ни каких объяснений. В этом случае очевидна целесообразность отказа от декомпозиции, потому что роль IDEF0 заключается в превращении сложного вопроса в понятный, а не в педантичной разработке очевидных деталей. Тривиальные функции лучше всего описываются небольшим объемом текста.

## Ход работы

1. Получить у преподавателя индивидуальный вариант задания (информационную систему и ее описание, включающее предметную и функциональную область).
2. Создать проект в CASE-средстве AllFusion Process Modeller (в свойствах проекта указать IDEF0 методологию).
3. Определить контекстный блок информационной системы и отобразить его на контекстной диаграмме.
4. Определить внешние входы, выходы, управления и механизмы, взаимодействующие с данной системой, и отобразить их на контекстной диаграмме.
5. Произвести декомпозицию контекстной диаграммы, представив систему в виде основных функций.
6. Определить на диаграмме A0 связи между функциональными блоками и родительской (контекстной диаграммой).
7. Выполнить декомпозицию каждого из функциональных блоков на диаграмме A0.
8. На диаграммах второго уровня необходимо определить функциональные блоки и связи между ними и родительской диаграммой аналогично п. 6.
9. При необходимости осуществить декомпозицию функциональных блоков второго уровня и построить диаграммы третьего уровня.
10. Используя возможности CASE-средства AllFusion Process Modeller проверить состоятельность построенной модели (модель не должна содержать ошибок).
11. Построить диаграмму дерева модели.
12. Продемонстрировать построенную модель преподавателю.

## Контрольные вопросы

1. Для чего предназначена методология функционального моделирования (IDEF0)?
2. На каком этапе ЖЦ ПО используется методология моделирования IDEF0?
3. Какие элементы могут присутствовать на диаграммах IDEF0?
4. Как на диаграммах IDEF0 обозначаются функциональные блоки?
5. Поясните назначение и правила отображения стрелок в ICOM-модели?
6. Какие виды комбинированных стрелок (соединений между функциональными блоками) могут использоваться на диаграммах IDEF0?
7. В каком случае применяется комбинированная стрелка «выход – вход»?

8. В каком случае применяется комбинированная стрелка «выход – управление»?
9. В каком случае применяется комбинированная стрелка «выход – механизм»?
10. В каком случае применяется комбинированная стрелка «выход – обратная связь по управлению»?
11. В каком случае применяется комбинированная стрелка «выход – обратная связь по входу»?
12. Какое количество функциональных блоков может присутствовать на диаграмме IDEF0?
13. Как должны быть расположены блоки на диаграмме?
14. Какие правила должны соблюдаться при декомпозиции стрелок на диаграмме IDEF0?
15. Как осуществляется построение иерархии диаграмм IDEF0?
16. Что должна содержать контекстная диаграмма IDEF0?
17. Перечислите стратегии декомпозиции диаграмм в методологии IDEF0.
18. Что такое туннель, и в каких случаях он может применяться на диаграммах IDEF0?
19. Перечислите критерии завершения декомпозиции в методологии IDEF0?
20. Какие дополнительные виды диаграмм присутствуют в методологии IDEF0?
21. Для чего используются FEO-диаграммы в методологии IDEF0?