

Лекция 6. Методология функционального моделирования IDEF0

Доц. каф. ИТ Глазунова Л.В.

Семейство стандарта IDEF

- **IDEF0** – методология функционального моделирования. С помощью графического языка IDEF0, разрабатываемая ИС представляется в виде набора взаимосвязанных функций (функциональных блоков – в терминах IDEF0). Как правило, моделирование средствами IDEF0 является первым этапом изучения любой системы;
- **IDEF1X** - относится к типу методологий «Сущность-взаимосвязь» (ER – EntityRelationship) и, как правило, используется для моделирования реляционных баз данных, имеющих отношение к рассматриваемой системе; Используется как одна из методологий в пакете ER Win.
- **IDEF3**– методология документирования процессов, происходящих в системе, которая используется, например, при исследовании технологических процессов на предприятиях.

Основы методологии IDEF0

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

IDEF0-модели состоят из трех типов документов: графических диаграмм, текста и глоссария. Эти документы имеют перекрестные ссылки друг на друга. Каждая диаграмма является единицей описания системы и располагается на отдельном листе.

Графическая диаграмма – главный компонент IDEF0-модели, содержащий блоки, дуги, соединения блоков и дуг и ассоциированные с ними отношения.

Методология может содержать 4 типа диаграмм:

- **Контекстная диаграмма** (диаграмма верхнего уровня), являясь вершиной древовидной структуры диаграмм, показывает назначение системы (основную функцию) и ее взаимодействие с внешней средой. В каждой модели может быть только одна контекстная диаграмма. После описания основной функции выполняется функциональная декомпозиция, т. е. определяются функции, из которых состоит основная.
- Далее функции делятся на подфункции и так до достижения требуемого уровня детализации исследуемой системы. Диаграммы, которые описывают каждый такой фрагмент системы, называются **диаграммами декомпозиции**.
- **Диаграмма дерева узлов** показывает иерархическую зависимость функций (работ), но не связи между ними. Их может быть сколько угодно, поскольку дерево можно построить на произвольную глубину и с произвольного узла.
- **Диаграммы для экспозиции** строятся для иллюстрации отдельных фрагментов модели с целью отображения альтернативной точки зрения на происходящие в системе процессы (например, с точки зрения руководства организации).

Правила построения модели включают:

- ограничение количества блоков на каждом уровне декомпозиции (правило 3-6 блоков);
- связность диаграмм (номера блоков);
- уникальность меток и наименований (отсутствие повторяющихся имен);
- синтаксические правила для графики (блоков и дуг);
- разделение входов и управлений (правило определения роли данных).
- отделение организации от функции, т.е. исключение влияния организационной структуры на функциональную модель.

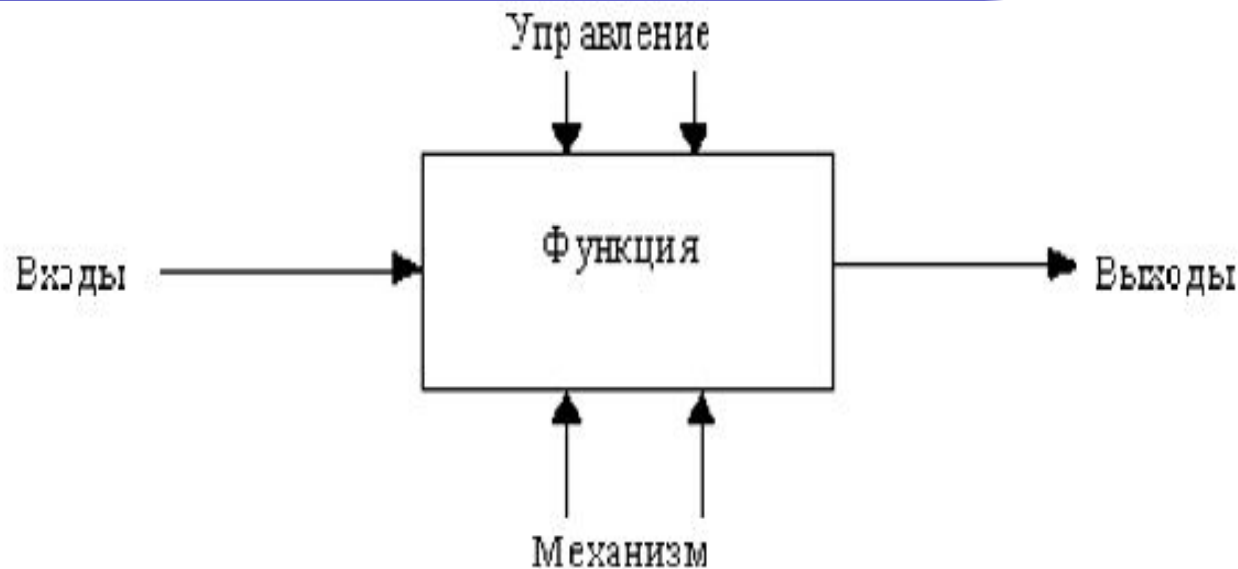
В IDEF0 реализованы три базовых принципа моделирования процессов:

- **Принцип функциональной декомпозиции** когда сложная бизнес-функция может быть представлена в виде совокупности элементарных функций.
- **Принцип ограничения сложности** - диаграммами должны быть разборчивыми и удобочитаемыми. Суть принципа ограничения сложности состоит в том, что количество блоков на диаграмме должно быть не менее двух и не более шести. Практика показывает, что соблюдение этого принципа приводит к тому, что функциональные процессы, представленные в виде IDEF0 модели, хорошо структурированы, понятны и легко поддаются анализу.
- **Принцип контекстной диаграммы.** Моделирование делового процесса начинается с построения контекстной диаграммы. На этой диаграмме отображается только один блок - главная бизнес-функция моделируемой системы.

Основные элементы диаграмм:

1. **Функциональный блок (Activity Box)** графически изображается в виде прямоугольника и олицетворяет собой некоторую конкретную функцию в рамках рассматриваемой системы. Внутри каждого блока помещается его имя и номер. Номер блока размещается в правом нижнем углу. По требованиям стандарта название каждого функционального блока должно быть сформулировано в глагольном наклонении, т.е. быть глаголом или глагольным оборотом.
2. **Интерфейсная дуга (Arrow)** отображает элемент системы, который обрабатывается функциональным блоком или оказывает влияние на функцию, отображенную данным функциональным блоком. Каждая интерфейсная дуга должна иметь свое уникальное наименование (Arrow Label). По требованию стандарта, наименование должно быть оборотом существительного. С помощью интерфейсных дуг отображают различные объекты, в той или иной степени определяющие процессы, происходящие в системе.

Функциональный блок

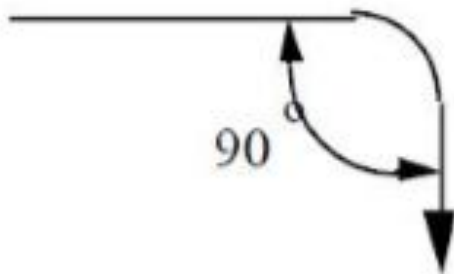


3. **Принцип декомпозиции (Decomposition)** применяется при разбиении сложного процесса на составляющие его функции. При этом уровень детализации процесса определяется непосредственно разработчиком модели. Декомпозиция позволяет постепенно и структурировано представлять модель системы в виде иерархической структуры отдельных диаграмм, что делает ее менее перегруженной и легко усваиваемой

Интерфейсная дуга (Arrow)



Прямолинейный отрезок стрелки



Ломаный сегмент стрелки.
Дуга сопряжения -90 град.



Ветвление стрелок



Слияние стрелок

Правила построения диаграммы IDEF0

- Функциональный блок (или Функция) преобразует Входы в Выходы (т.е. входную информацию в выходную), Управление определяет, когда и как это преобразование может или должно произойти. Исполнители непосредственно осуществляют это преобразование.
- С дугами связаны надписи (или метки) на естественном языке, описывающие данные, которые они представляют. · Дуги показывают, как функции между собой взаимосвязаны, как они обмениваются данными и осуществляют управление друг другом.
- Выходы одной функции могут быть Входами, Управлением или Исполнителями для другой.
- Дуги могут разветвляться и соединяться.
- Функциональный блок, который представляет систему в качестве единого модуля, детализируется на другой диаграмме с помощью нескольких блоков, соединенных между собой интерфейсными дугами.
- Эти блоки представляют основные подфункции (подмодули) единого исходного модуля.

Правила построения диаграммы IDEF0

- Данная декомпозиция выявляет полный набор подмодулей, каждый из которых представлен как блок, границы которого определены интерфейсными дугами.
- Каждый из этих подмодулей может быть декомпозирован подобным же образом для более детального представления.
- Каждая подфункция может содержать только те элементы, которые входят в исходную функцию. Кроме того, модель не может опустить какие-либо элементы, т.е., как уже отмечалось, родительский блок и его интерфейсы обеспечивают контекст. К нему нельзя ничего добавить, и из него не может быть ничего удалено.
- 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 представляют одну и ту же часть системы.

Контекстная диаграмма

Модель IDEF0 всегда начинается с представления системы как единого целого – одного функционального блока с интерфейсными дугами, простирающимися за пределы рассматриваемой области.

Такая диаграмма с одним функциональным блоком называется **контекстной диаграммой**, и обозначается идентификатором “**A-0**”.

Поскольку единственный блок представляет всю систему как единое целое, имя, указанное в блоке, является **именем проекта**. Это верно и для интерфейсных дуг - они также представляют **полный набор внешних интерфейсов системы в целом**.

Контекстная диаграмма устанавливает область моделирования и ее границу.

Контекстная диаграмма

Контекстная диаграмма А-0 также должна содержать краткие утверждения, определяющие **точку зрения** должностного лица или подразделения, с позиций которого создается модель, и **цель**, для достижения которой ее разрабатывают.

Формулировка цели выражает причину создания модели, т.е. содержит перечень вопросов, на которые должна отвечать модель, что в значительной мере определяет ее структуру. Наиболее важные свойства объекта обычно выявляются на верхних уровнях иерархии; по мере декомпозиции функции верхнего уровня и разбиения ее на подфункции, эти свойства уточняются.

Точка зрения определяет, что и в каком разрезе можно увидеть в пределах контекста модели. Изменение точки зрения, приводит к рассмотрению других аспектов объекта.

Пример. Предметная область: деятельность предприятия по сборке и продаже компьютеров

Контекстная диаграмма «Деятельность предприятия по сборке и продаже компьютеров»

Название процесса	Деятельность предприятия по сборке и продаже компьютеров
Цель процесса	Воспроизвести бизнес-процессы, происходящие на предприятии
Точка зрения	С позиции директора как лица, знающего структуру предприятия в целом
Входы	•Заказы клиентов - список компьютеров и их конфигурация, которые клиент желает приобрести •Комплектующие от поставщиков - комплектующие, полученные от поставщиков, из которых собираются компьютеры и ноутбуки

Контекстная диаграмма «Деятельность предприятия по сборке и продаже компьютеров»

Управление	• Законодательство - различные законодательные документы, которыми руководствуется предприятие в процессе своей деятельности • Правила и процедуры - различные правила и процедуры, которыми руководствуется предприятие в процессе своей деятельности (например, правила сборки и тестирования компьютеров, процедура общения с клиентами и т.п.)
Механизмы	• Бухгалтерская система • Персонал
Выход	• Готовая продукция - собранные компьютеры и ноутбуки • Заказы поставщикам - список комплектующих, которые предприятие закупает у поставщиков • Оплата за комплектующие - деньги поставщикам за комплектующие • Маркетинговые материалы - прайс-листы, рекламки и т.п.

Контекстная диаграмма «Деятельность предприятия по сборке и продаже компьютеров»



Диаграммы декомпозиции

Каждая подфункция моделируется отдельным блоком. Каждый **родительский блок** подробно описывается **дочерней диаграммой** на более низком уровне. Все дочерние диаграммы должны быть в пределах области контекстной диаграммы.

Дочерняя диаграмма. Единственная функция, представленная на контекстной диаграмме верхнего уровня, может быть разложена на основные подфункции посредством создания дочерней диаграммы. В свою очередь, каждая из этих подфункций может быть разложена на составные части посредством создания дочерней диаграммы следующего, более низкого уровня. Каждая дочерняя диаграмма содержит дочерние блоки и стрелки, обеспечивающие дополнительную детализацию родительского блока.

Родительская диаграмма - диаграмма, которая содержит один или более родительских блоков. Каждая обычная (не контекстная) диаграмма является также дочерней диаграммой, поскольку, по определению, она подробно описывает некоторый родительский блок. Таким образом, любая диаграмма может быть как родительской диаграммой, так и дочерней

Диаграммы декомпозиции для системы «Деятельность предприятия по сборке и продаже компьютеров»

Первый уровень декомпозиции состоит из 4 блоков:

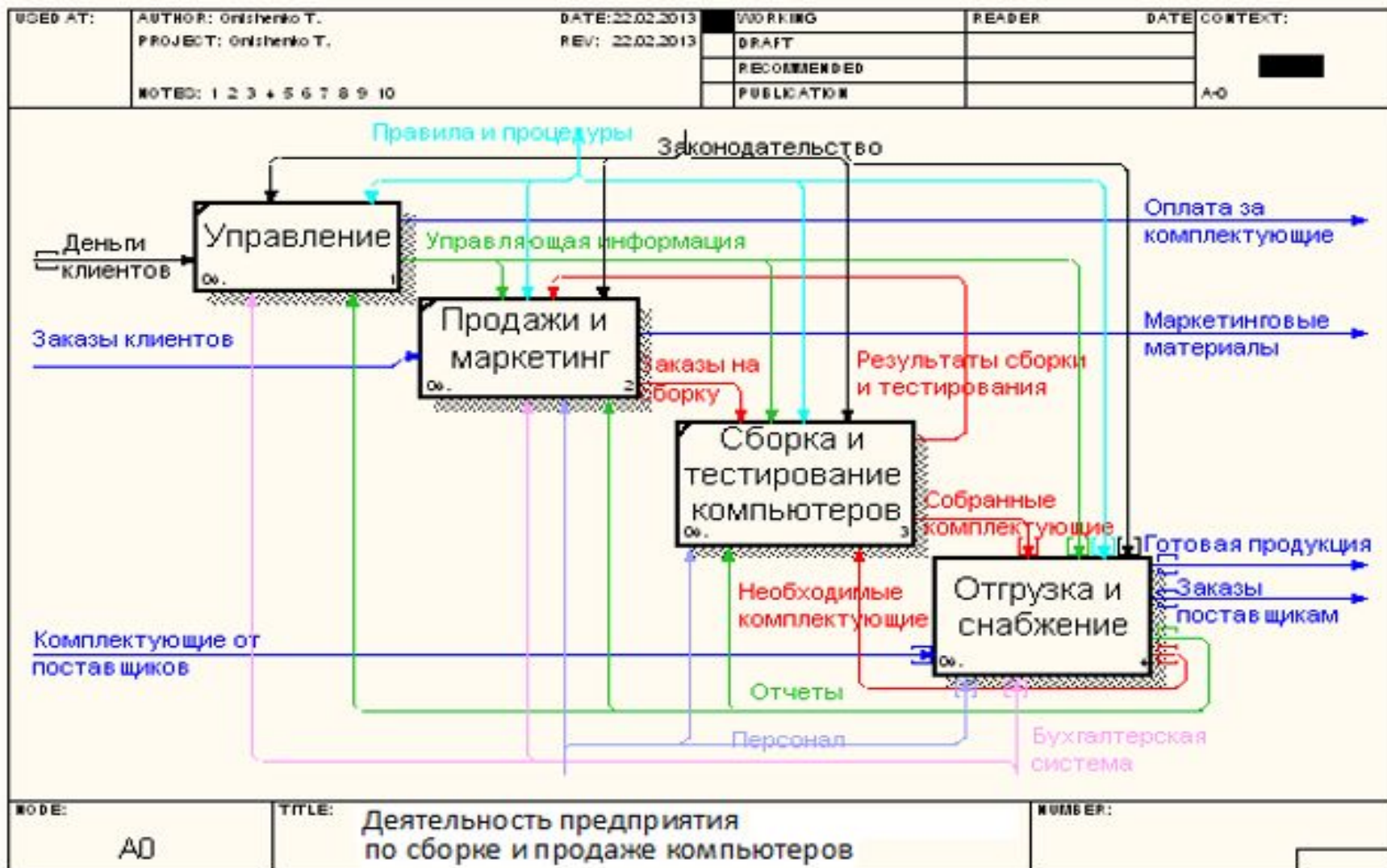
1. Управление
2. Продажи и маркетинг
3. Сборка и тестирование
4. Отгрузка и снабжение

Управление включает в себя общее управление предприятием, финансами, кадрами, бухгалтерию и т.п.

Продажи и маркетинг - это работа с клиентами, презентации, выставки, реклама, маркетинговые исследования и т.д.

Отгрузка и снабжение предполагает снабжение предприятия необходимыми комплектующими, хранение и отгрузка готовой продукции

Диаграмма декомпозиции первого уровня A0



Второй уровень декомпозиции блока «Сборка и тестирование компьютеров» состоит из:

1. Отслеживание расписания, управление сборкой и тестированием
2. Сборка компьютеров
3. Тестирование компьютеров

Диаграмма декомпозиции блока «Сборка и тестирование компьютеров» А3

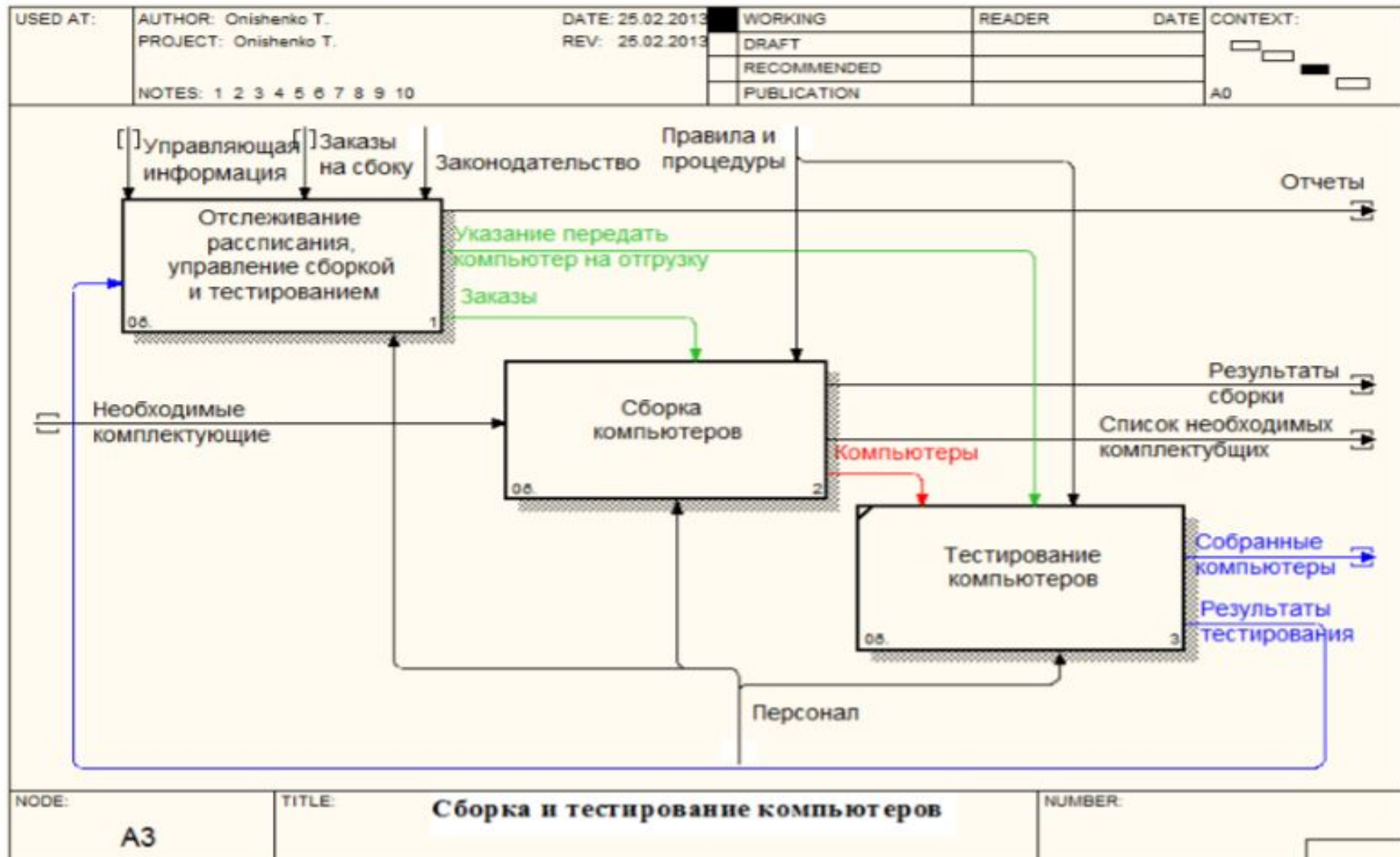
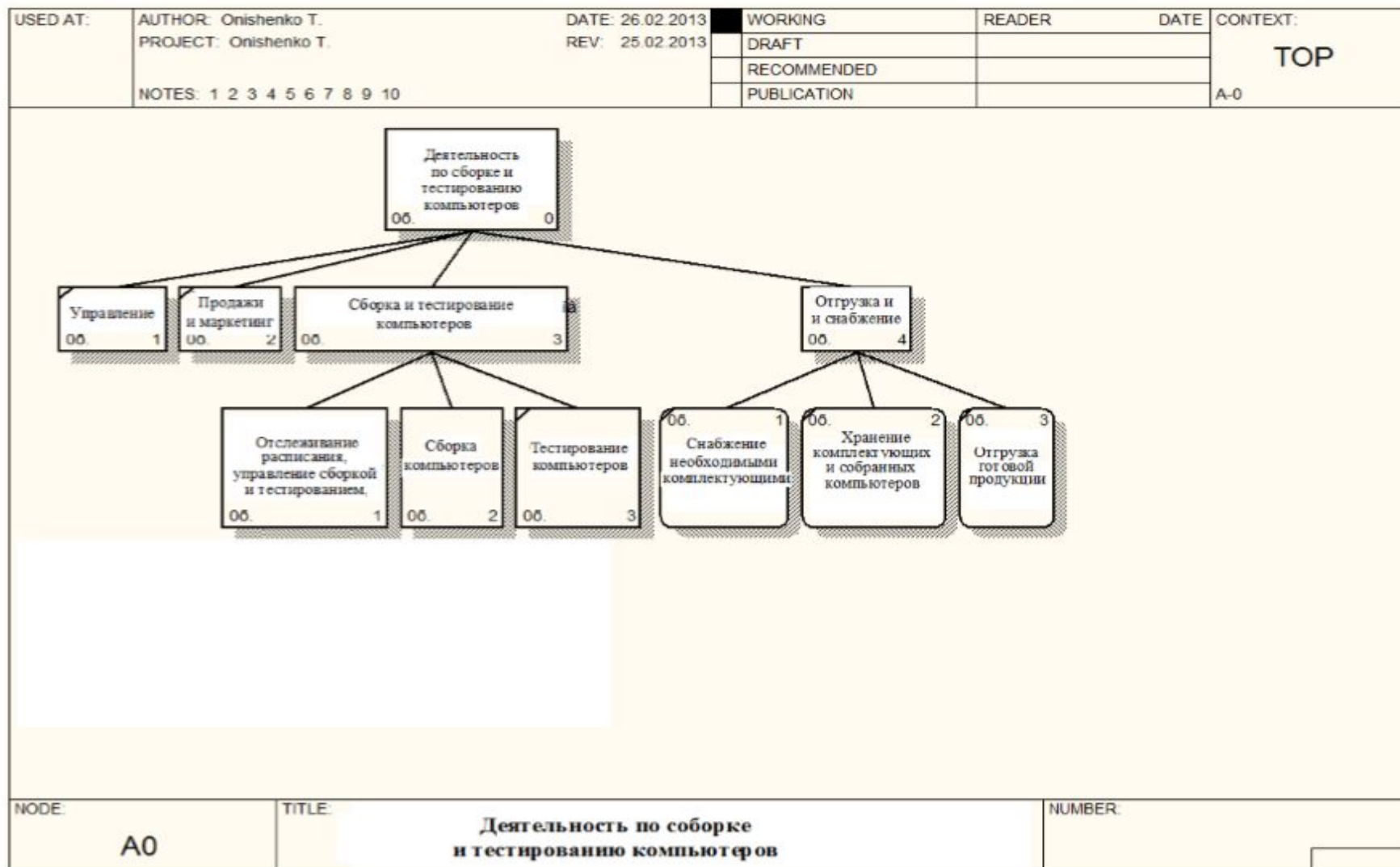


Диаграмма дерева узлов

Разработанная модель IDEF0 со всеми уровнями структурной декомпозицией может быть представлена на единственной диаграмме в виде **дерева узлов, дополняющего перечень узлов.**

Для изображения этого дерева нет стандартного формата. Единственное требование состоит в том, что вся иерархия узлов модели должна быть представлена наглядно и понятно.

Диаграмма дерева узлов для системы «Деятельность предприятия по сборке и продаже компьютеров»



Типы связей между блокам

- **Иерархическая связь** (связь «часть» – «целое») имеет место между функцией и подфункциями, из которых она состоит

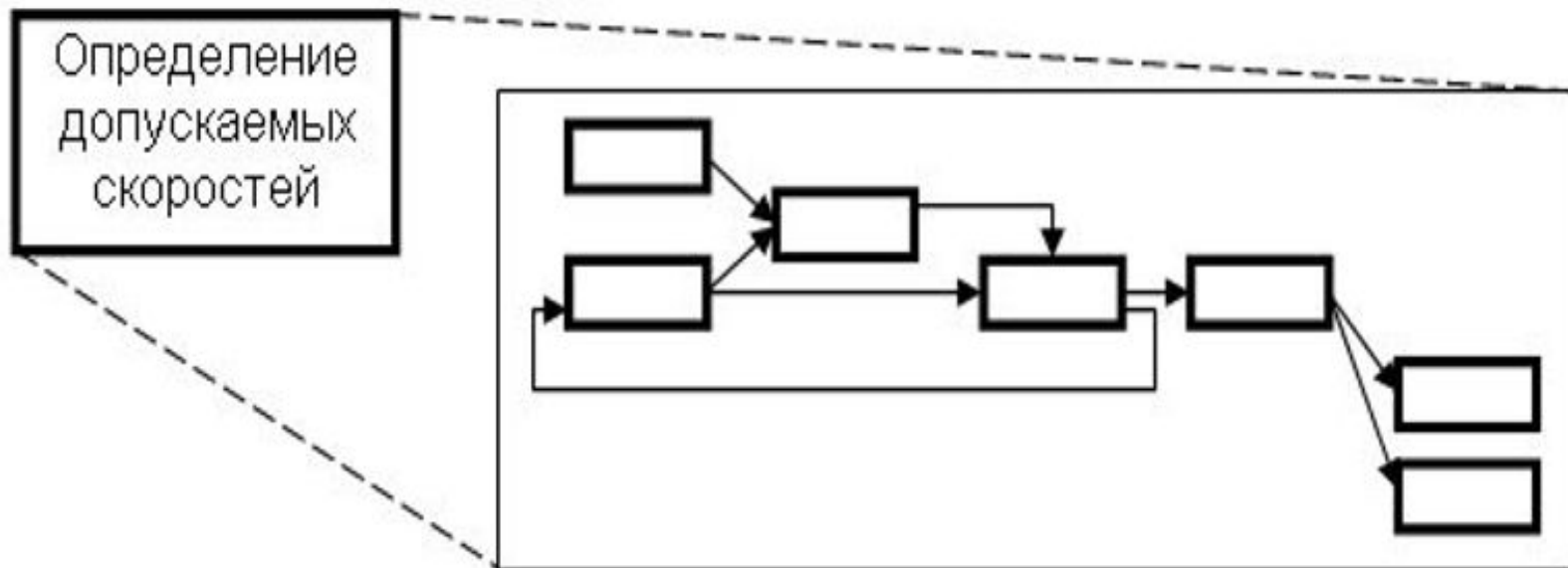


Рис. 4.7. Иерархическая связь

Типы связей между блокам

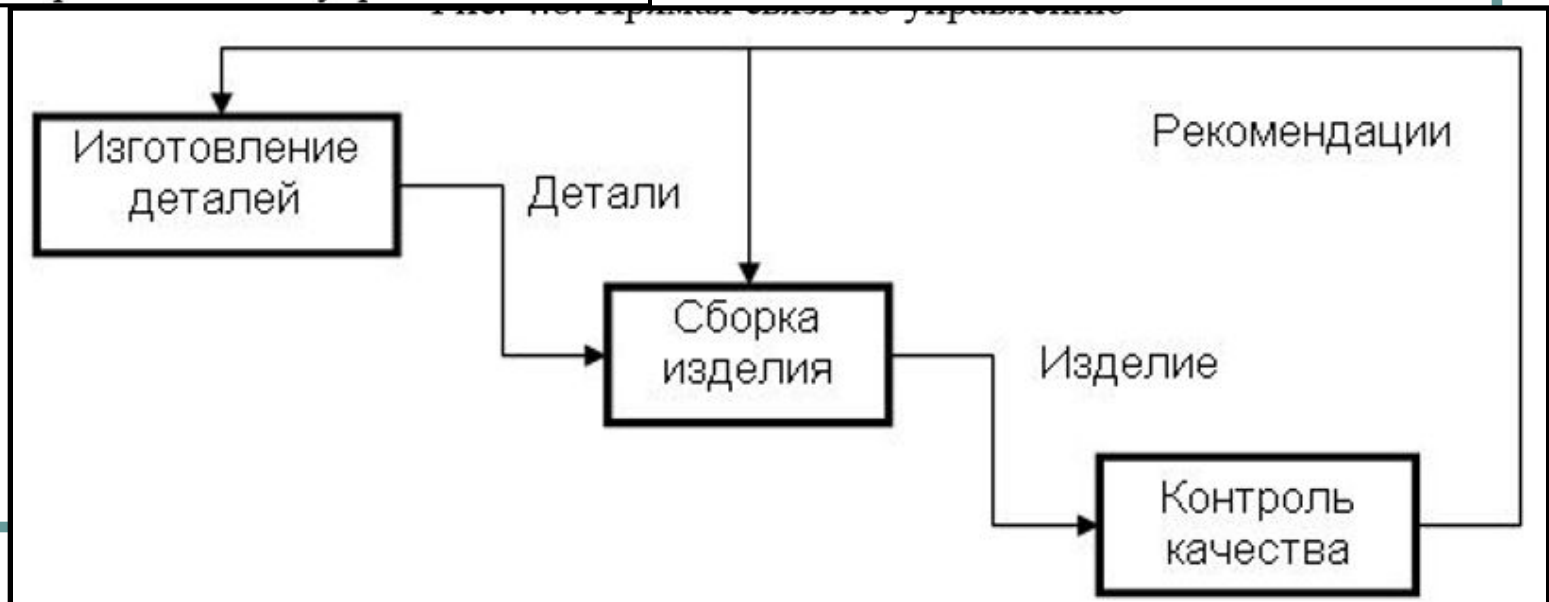
- **Регламентирующая (управляющая, подчиненная) связь** отражает зависимость одной функции от другой, когда выход одной работы направляется на управление другой. **Функцию, из которой выходит управление, следует считать регламентирующей или управляющей, а в которую входит – подчиненной.** Различают **прямую связь по управлению**, когда управление передается с вышестоящей работы на нижестоящую, и **обратную связь по управлению**, когда управление передается от нижестоящей к вышестоящей

Регламентирующая (управляющая, подчиненная) связь



Рис. 4.8. Прямая связь по управлению

Рис Обратная связь по управлению



Типы связей между блокам

- **Функциональная (технологическая) связь** имеет место, когда выход одной функции служит входными данными для следующей функции. С точки зрения потока материальных объектов данная связь показывает технологию (последовательность работ) обработки этих объектов. Различают **прямую связь по входу**, когда выход передается с вышестоящей работы на нижестоящую, и **обратную связь по входу**, когда выход передается с нижестоящей к вышестоящей.
- **Потребительская (последовательная) связь** имеет место, когда выход одной функции служит механизмом для следующей функции. Таким образом, одна функция потребляет ресурсы, вырабатываемые другой
- **Логическая связь** наблюдается между логически однородными функциями. Такие функции, как правило, выполняют одну и ту же работу, но разными (альтернативными) способами или, используя разные исходные данные (материалы).

Функциональная (технологическая) связь

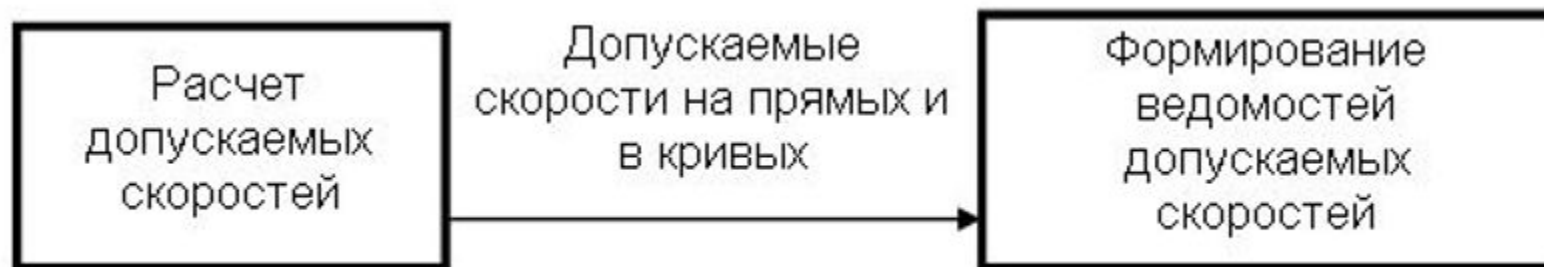


Рис. 4.10. Прямая связь по входу

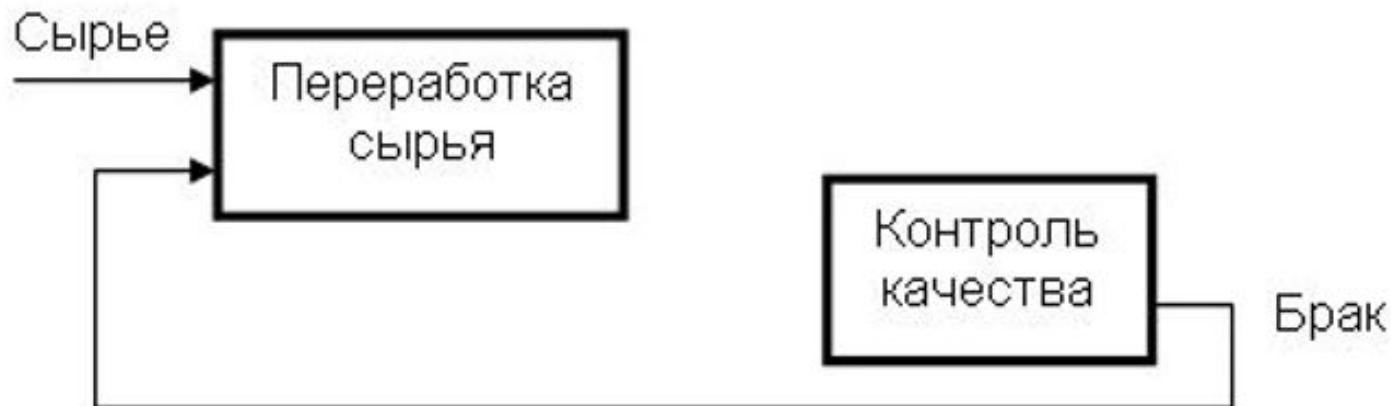


Рис. 4.11. Обратная связь по входу

Потребительская (последовательная) и логическая связи



Рис. 4.12. Потребительская связь

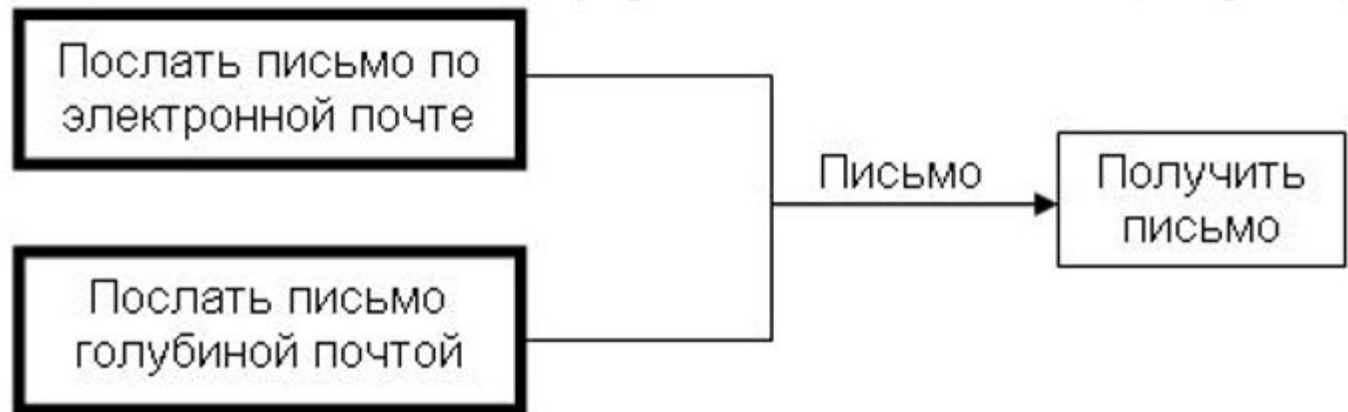


Рис. 4.13. Логическая связь

Типы связей между блокам

- **Коллегиальная (методическая) связь** имеет место между функциями, алгоритм работы которых определяется одним и тем же управлением. Аналогом такой связи является совместная работа сотрудников одного отдела (коллег), подчиняющихся начальнику, который отдает указания и приказы (управляющие сигналы). Такая связь также возникает, когда алгоритмы работы этих функций определяются одним и тем же методическим обеспечением (СНИП, ГОСТ, официальными нормативными материалами и т. д.), служащим в качестве управления.
- **Ресурсная связь** возникает между функциями, использующими для своей работы одни и те же ресурсы. Ресурсно-зависимые функции, как правило, не могут выполняться одновременно.
- **Информационная связь** имеет место между функциями, использующими в качестве входных данных одну и ту же информацию
- **Временная связь** возникает между функциями, которые должны выполняться одновременно до или одновременно после другой функции

Коллегиальная (методическая) и ресурсная связи

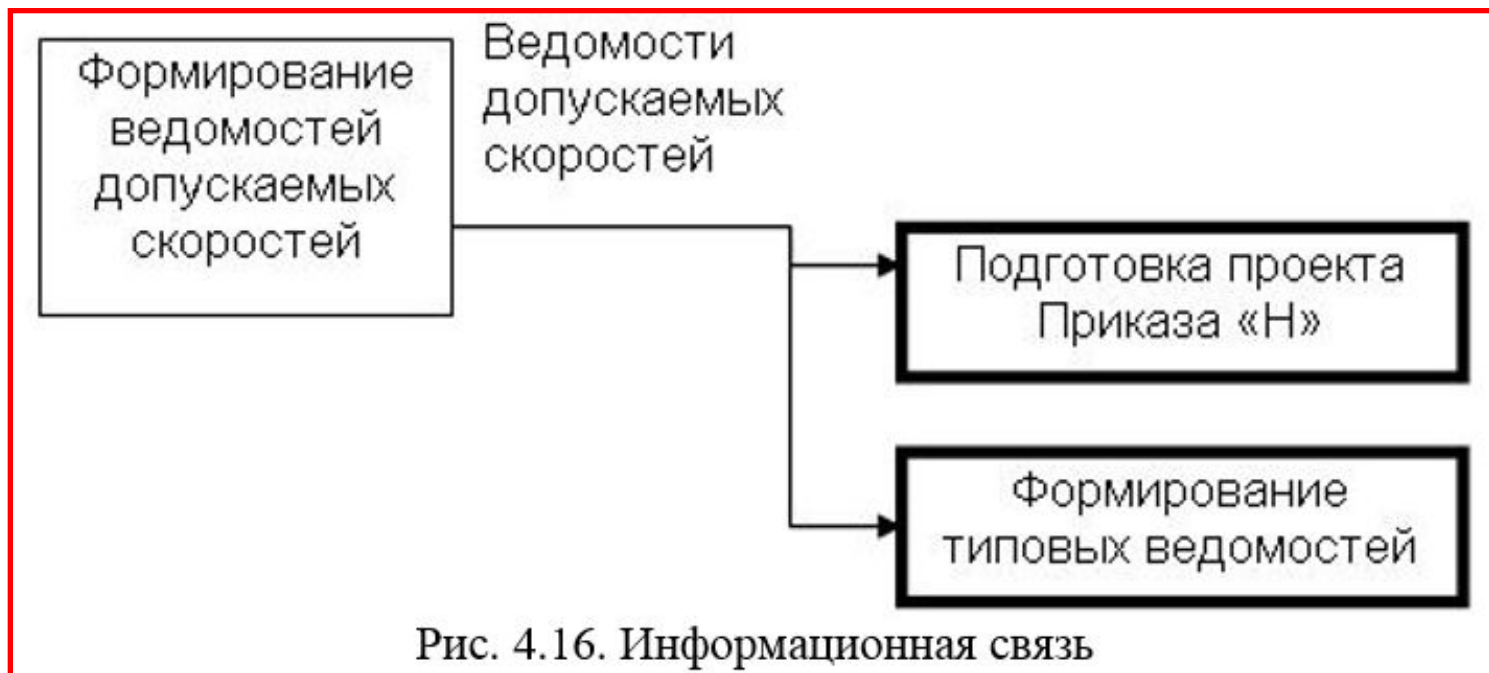


Рис. 4.14. Методическая связь



Рис. 4.15. Ресурсная связь

Информационная связь



Временная связь

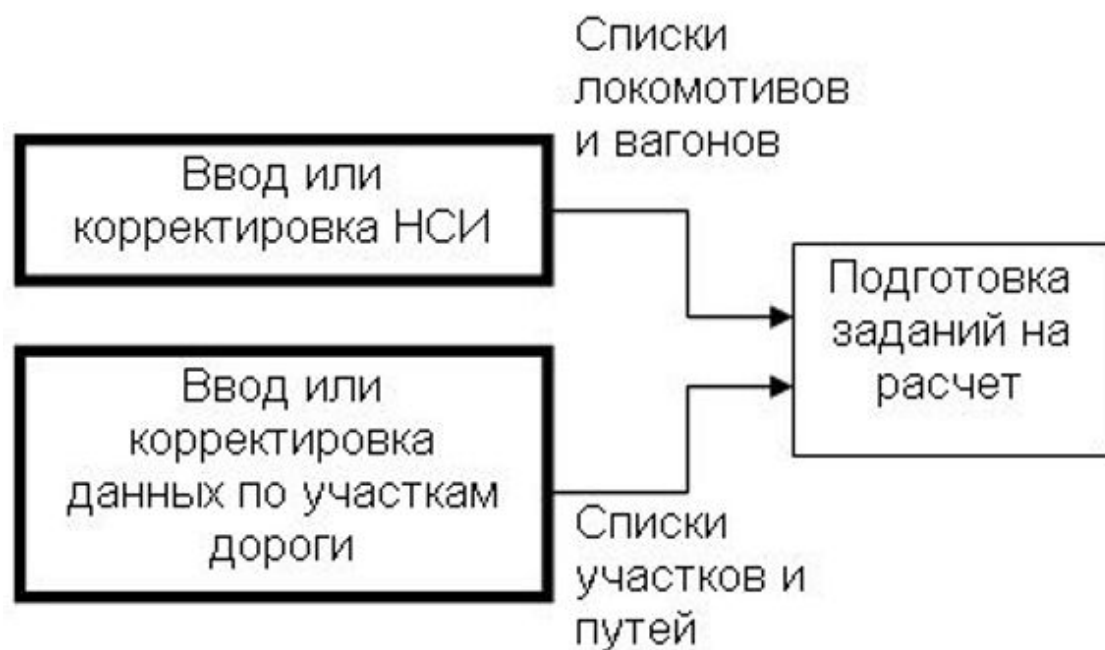
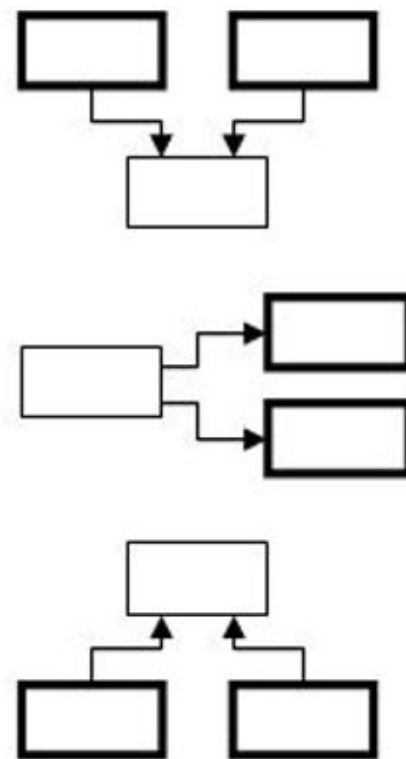


Рис. 4.17. Временная связь



Типы связей между блокам

- **Случайная связь** возникает, когда конкретная связь между функциями мала или полностью отсутствует



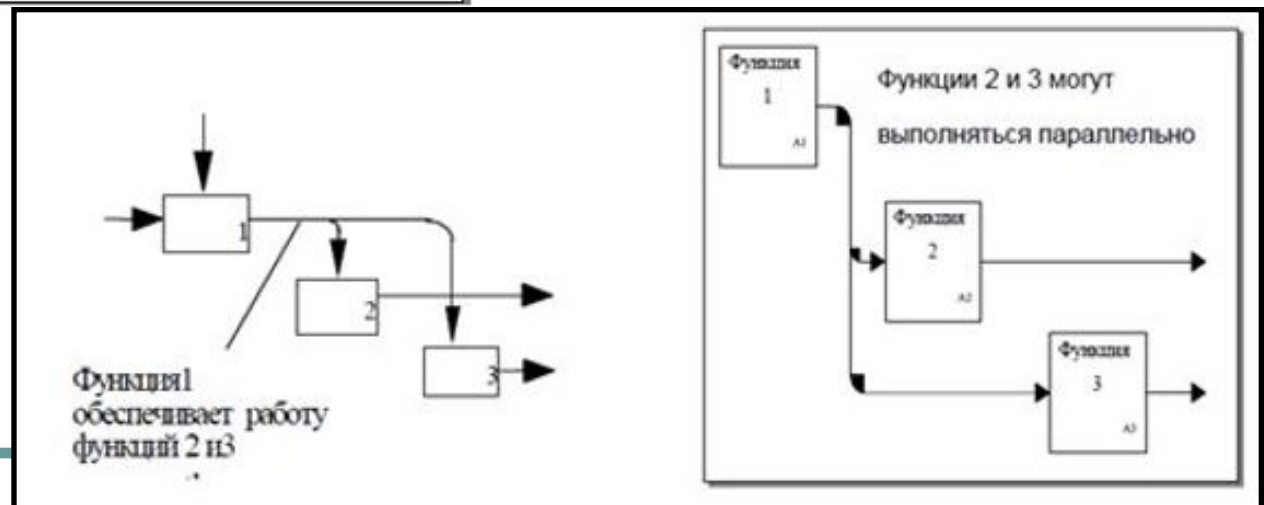
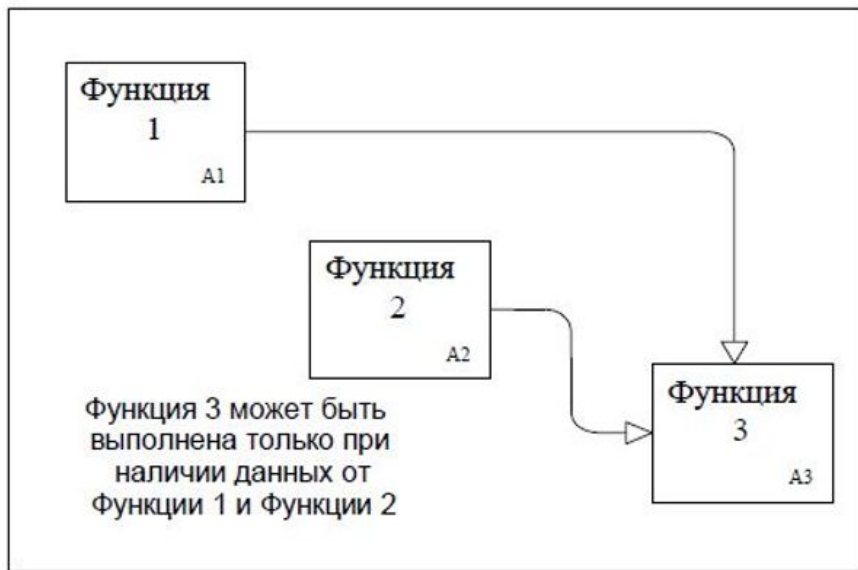
Рис. 4.18. Случайная связь

Значение стрелок на диаграммах

Стрелки на диаграммах IDEF0 , представляя данные или материальные объекты, **одновременно задают своего рода ограничения (условия)**. Входные и управляющие стрелки блока, соединяющие его с другими блоками, по сути описывают условия, которые должны быть выполнены для того, чтобы реализовалась функция, записанная в качестве имени блока. Различные функции в модели могут быть выполнены параллельно, если удовлетворяются необходимые ограничения (условия).

Ветвление и слияние стрелок призвано уменьшить загруженность диаграмм графическими элементами (линиями). Чтобы стрелки и их сегменты правильно описывали связи между блоками – источниками и блоками – потребителями, используется **механизм меток**. Метки связываются с сегментами посредством **ТИЛЬД**

Стрелки на диаграммах IDEF0 задают своего рода ограничения (условия).

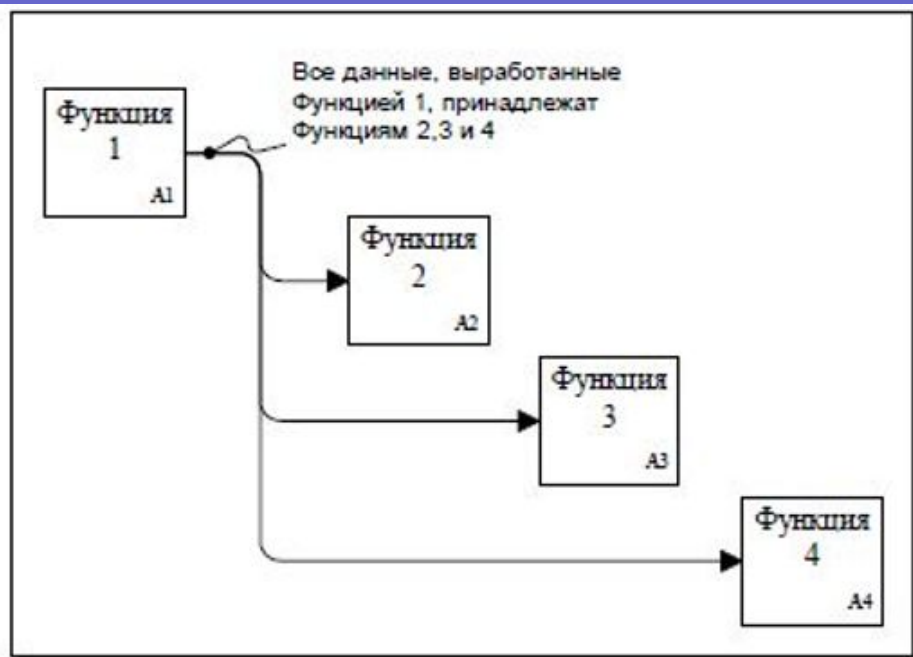


Механизм меток

При этом между сегментами возникают определенные отношения, описанные ниже:

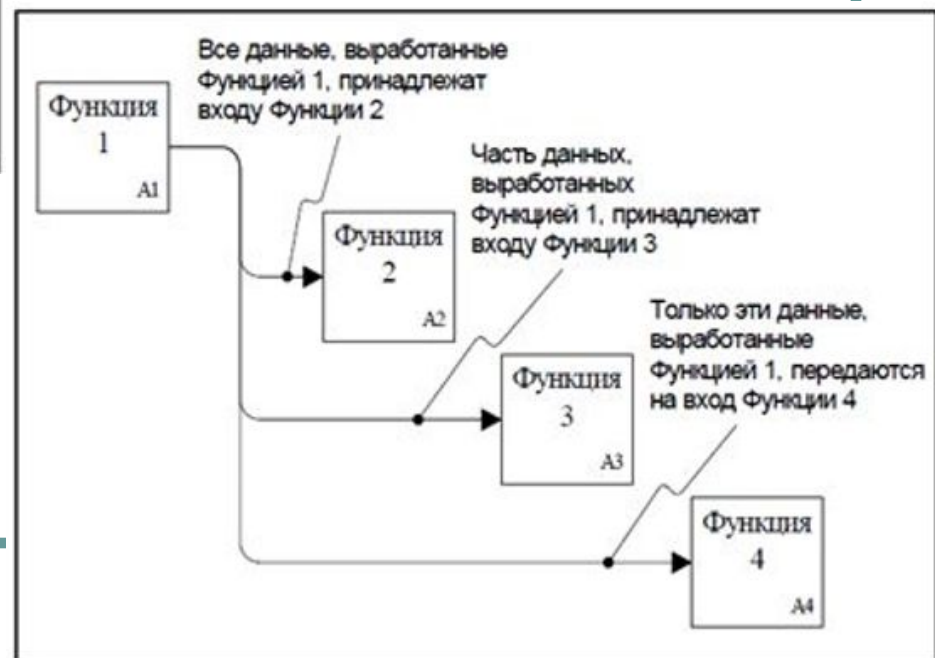
- A. непомеченные сегменты содержат все объекты, указанные в метке стрелки перед ветвлением (т.е. все объекты принадлежат каждому из сегментов);
- B. сегменты, помеченные после точки ветвления, содержат все объекты, указанные в метке стрелки перед ветвлением, или их часть, описываемую меткой каждого конкретного сегмента;
- C. при слиянии непомеченных сегментов объединенный сегмент стрелки содержит все объекты, принадлежащие сливаемым сегментам и указанные в общей метке стрелки после слияния;
- D. при слиянии помеченных сегментов объединенный сегмент содержит все или некоторые объекты, принадлежащие сливаемым сегментам и перечисленные в общей метке после слияния; если общая метка после слияния отсутствует, это означает, что общий сегмент передает все объекты, принадлежащие сливаемым сегментам;

Механизм меток



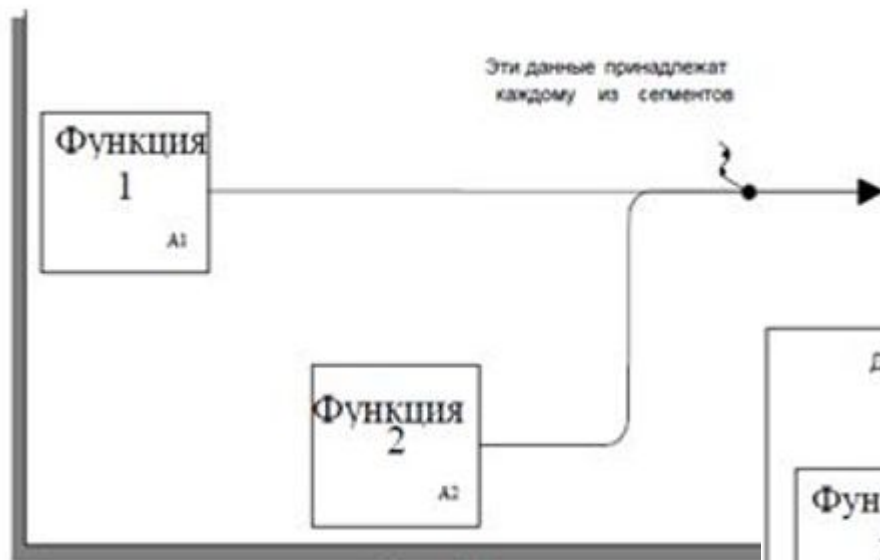
Отношение А.

Отношение В.

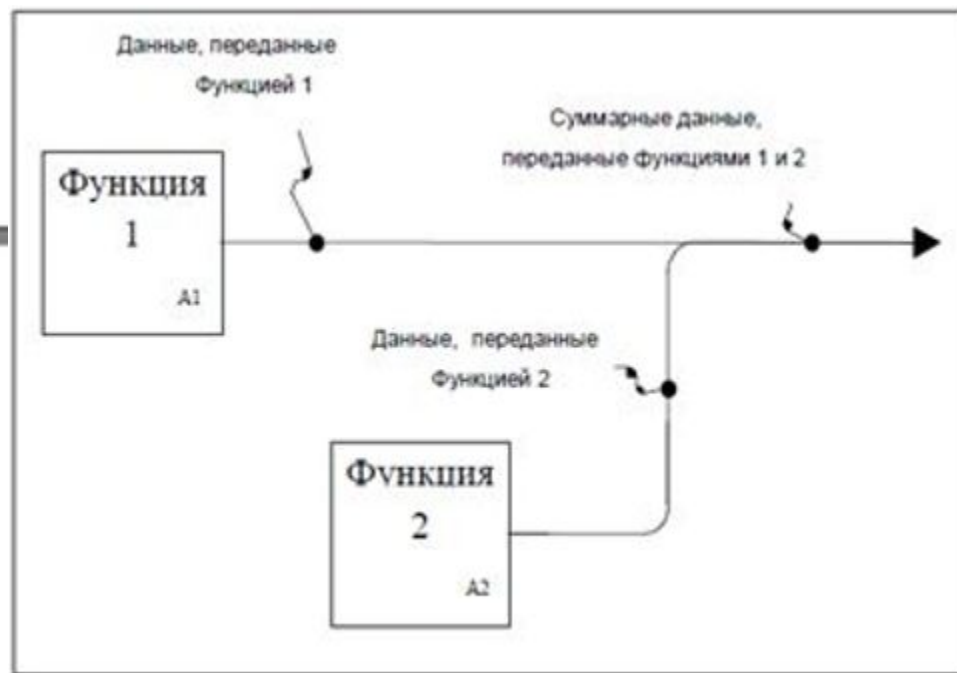


Механизм меток

Отношение С.



Отношение D.



Отношения блоков на диаграммах

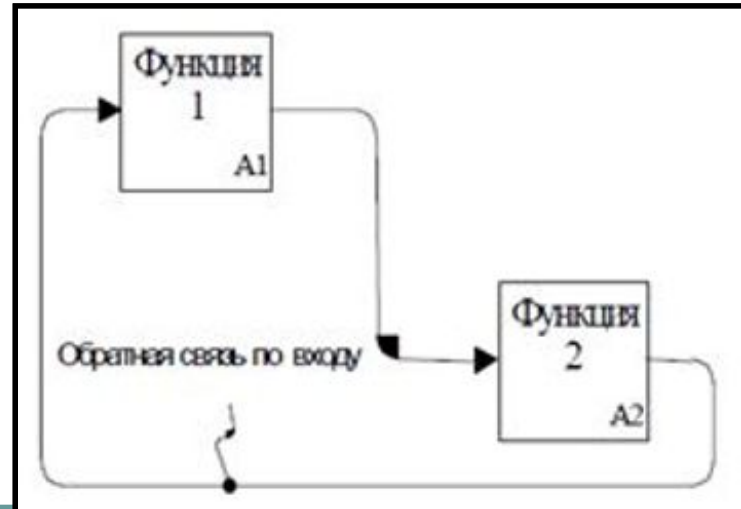
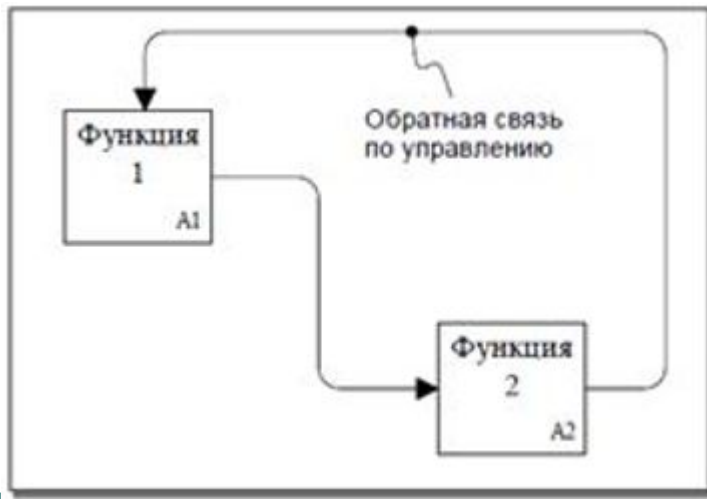
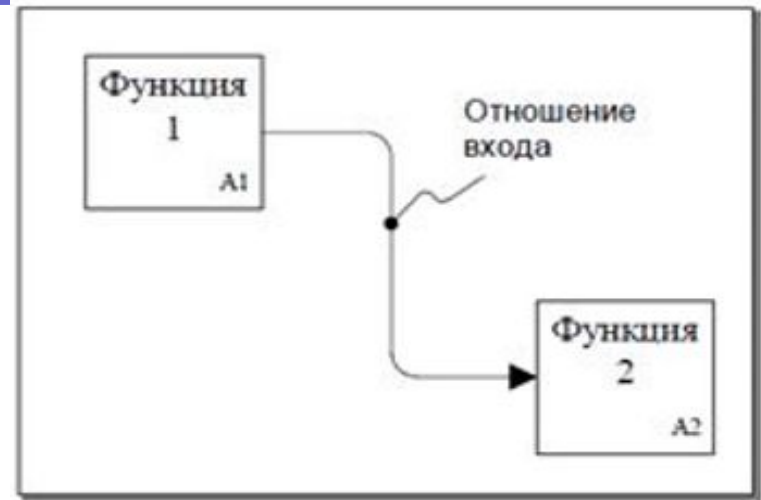
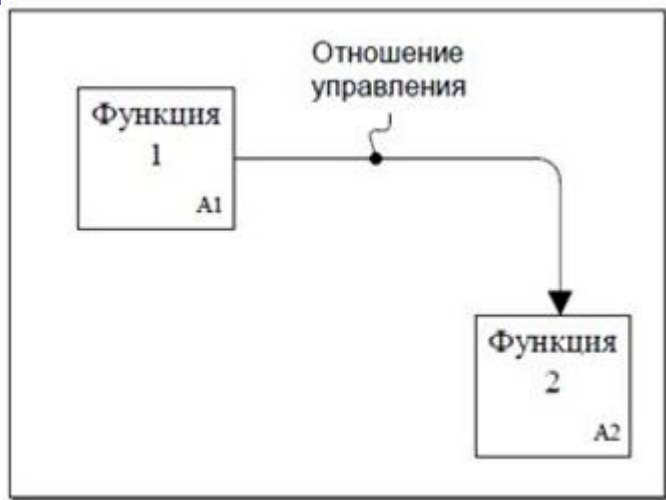
В методологии IDEF0 существует **6 (шесть) типов связей** между блоками в пределах одной диаграммы:

- **Доминирование** – предполагается, что блоки, расположенные на диаграмме выше и левее, «доминируют» над блоками, расположенными ниже и правее. «Доминирование» понимается как влияние, которое один блок оказывает на другие блоки диаграммы.
- **Управление** - возникает тогда, когда выход одного блока служит управляющим воздействием на блок с меньшим доминированием;
- **Выход – вход** - возникает при соединении выхода одного блока с входом другого блока с меньшим доминированием;

Отношения блоков на диаграммах

- **Обратная связь по управлению** - представляет итерацию (выход функции влияет на будущее выполнение других функций с большим доминированием, что впоследствии влияет на исходную функцию) и возникает тогда, когда выход некоторого блока создает управляющее воздействие на блок с большим доминированием;
- **Обратная связь по входу** - представляет итерацию, когда выход блока становится входом другого блока с большим доминированием.
- **Выход – механизм** - отражают ситуацию, при которой выход одной функции становится средством достижения цели для другой, т.е. возникают при отображении в модели процедур пополнения и распределения ресурсов, создания или подготовки средств для выполнения функций системы (например, приобретение или изготовление требуемых инструментов и оборудования, обучение персонала, организация физического пространства, финансирование, закупка материалов и т.д.;

Управление, выход – вход, обратная связь по управлению, обратная связь по входу



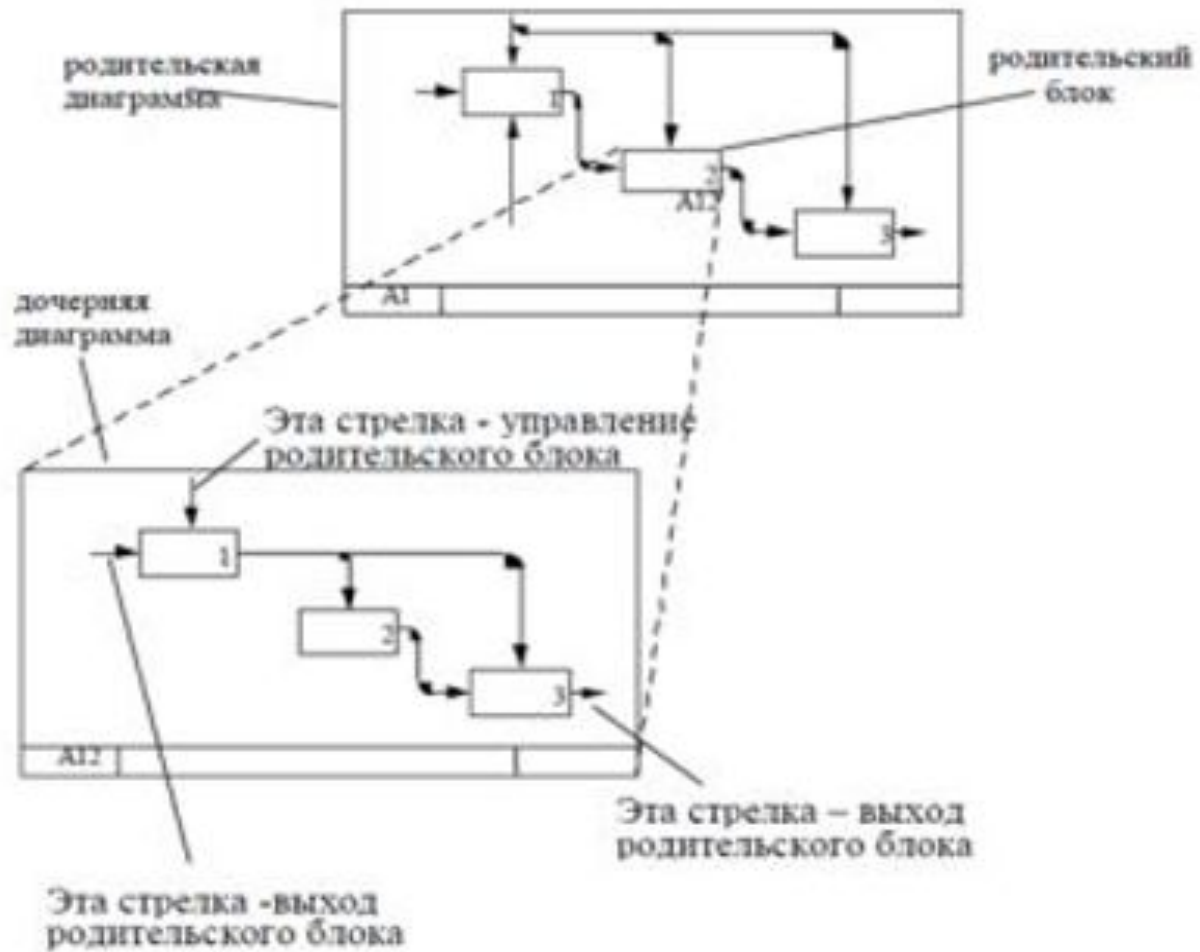
Отношения между блоками диаграммы и другими диаграммами (окружающей средой).

Отношения между блоками диаграммы и другими диаграммами, являющимися по отношению к рассматриваемой диаграмме окружающей средой (окружением), описываются **граничными стрелками**.

На обычной (не контекстной) диаграмме граничные стрелки представляют входы, управления, выходы или механизмы родительского блока диаграммы. Источник или потребитель граничных стрелок можно обнаружить, только изучая родительскую диаграмму.

Все граничные стрелки на дочерней диаграмме (за исключением стрелок, помещенных в тоннель должны соответствовать стрелкам родительского блока.

Граничные стрелки.

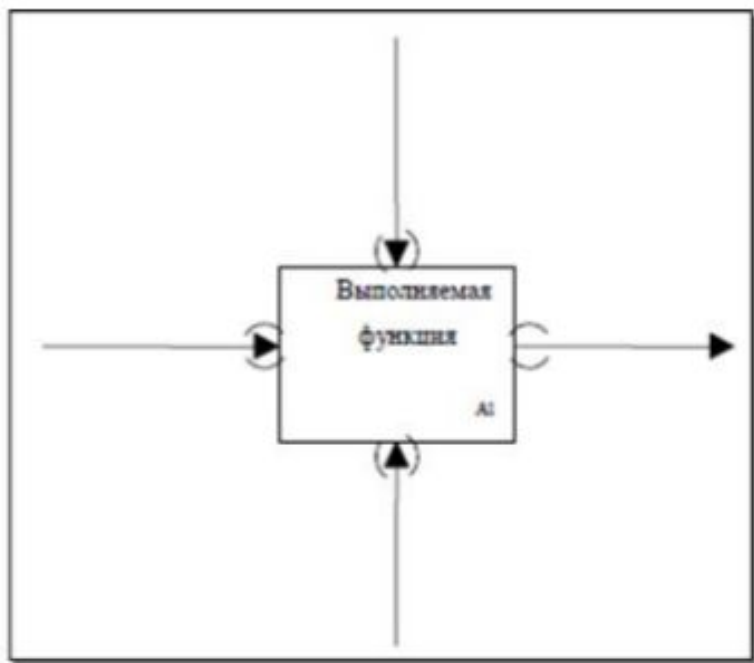


Стрелки, помещенные в «туннель».

Туннель - круглые скобки в начале и/или окончании стрелки. Туннельные стрелки означают, что данные, выраженные этими стрелками, **не рассматриваются на родительской диаграмме и/или на дочерней диаграмме.**

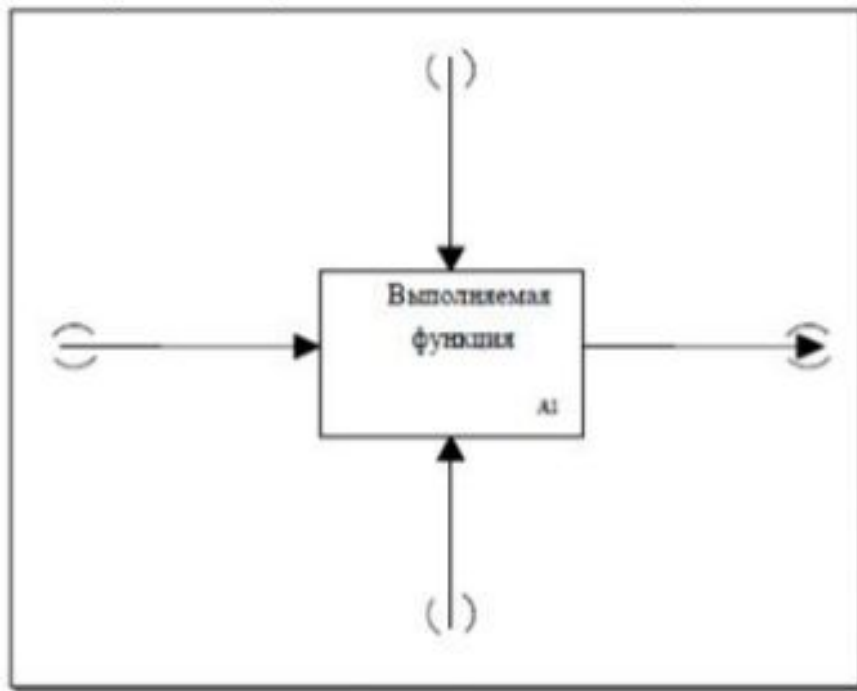
- А. Стрелка, помещенная в туннель там, где она **присоединяется к блоку**, означает, что данные, выраженные этой стрелкой, **не обязательны на следующем уровне декомпозиции.**
- В. Стрелка, помещаемая в туннель **на свободном конце** означает, что **выраженные ею данные отсутствуют на родительской диаграмме.**

Стрелки, помещенные в «туннель».



Туннель вариант А.

Туннель вариант В.



Стрелки, помещенные в «туннель».

