## Практическая работа № 3.

## Метод функционального моделирования SADT.

**2.2.1. ОБЩИЕ СВЕДЕНИЯ**

Метод SADT разработан Дугласом Россом (SoftTech, Inc.) 1973 г. Данный метод успешно использовался в военных, промышленных и коммерческих организациях США для решения широкого круга задач, таких, как долгосрочное и стратегическое планирование, автоматизированное производство и проектирование, разработка ПО для оборонных систем, управление финансами и материально-техническим снабжением и др. Метод SADT поддерживается Министерством обороны США, которое было инициатором разработки стандарта IDEF0 (Icam DEFinition) — подмножества SADT, являющегося основной частью программы ICAM (Integrated Computer Aided Manufacturing — интегрированная компьютеризация производства), проводимой по инициативе ВВС США. IDEF0 был утвержден в качестве федерального стандарта США, его подробные спецификации можно найти на сайте http:/www.idef.com.

Метод SADT представляет собой совокупность правил и процедур, предназначенных для построения функциональной модели объекта какой-либо предметной области. ***Функциональная модель***SADT отображает функциональную структуру объекта, т.е. производимые им действия и связи между этими действиями. Основные элементы этого метода основываются на следующих концепциях:

· графическое представление блочного моделирования. Графика блоков и дуг SADT-диаграммы отображает функцию в виде блока, а интерфейсы входа-выхода представляются дугами, соответственно входящими в блок и выходящими из него. Взаимодействие блоков друг с другом описывается посредством интерфейсных дуг, выражающих "ограничения", которые, в свою очередь, определяют, когда и каким образом функции выполняются и управляются;

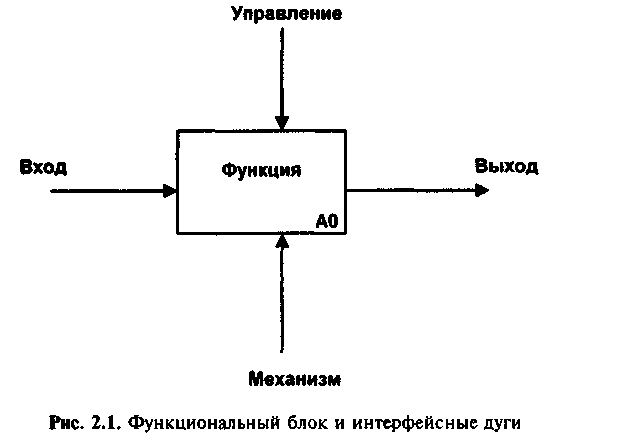
· строгость и точность. Выполнение правил SADT требует достаточной строгости и точности, не накладывая в то же время чрезмерных ограничений на действия аналитика. Правила SADT включают: ограничение количества блоков на каждом уровне декомпозиции (правило 3—6 блоков), связность диаграмм (номера блоков), 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 административной структуры организации на функциональную модель.

Метод SADT может использоваться для моделирования самых разнообразных систем и определения требований и функций с последующей разработкой информационной системы, удовлетворяющей этим требованиям и реализующей эти функции. В существующих системах метод SADT может применяться для анализа функций, выполняемых системой, и указания механизмов, посредством которых они осуществляются.

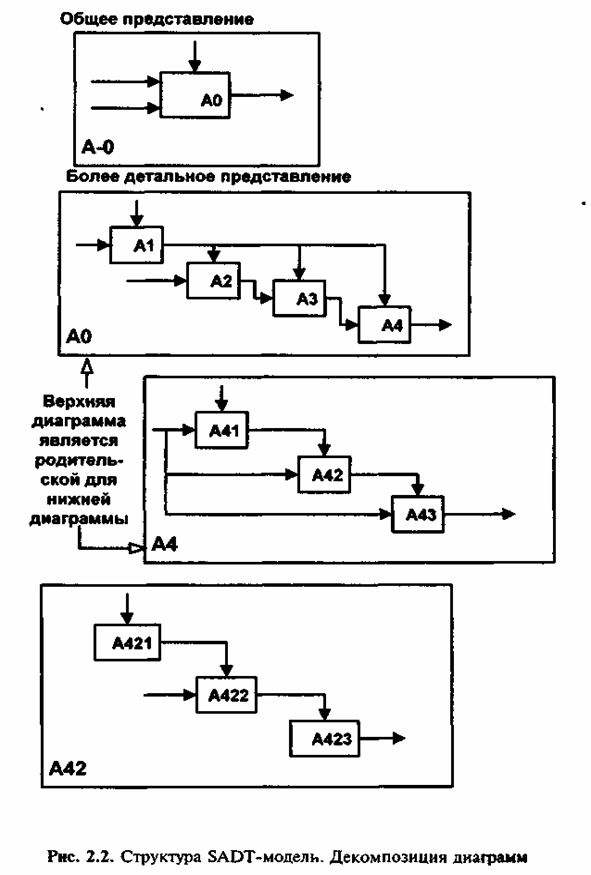
**2.2.2. СОСТАВ ФУНКЦИОНАЛЬНОЙ МОДЕЛИ**

Результатом применения метода SADT является модель, которая состоит из диаграмм, фрагментов текстов и глоссария, имеющих ссылки друг на друга. Диаграммы — главные компоненты модели, все функции организации и интерфейсы на них представлены как блоки и дуги соответственно. Место соединения дуги с блоком определяет тип интерфейса *Управляющая информация*входит в блок сверху, в то время как *входная информация,*которая подвергается обработке, показана с левой стороны блока, а **результаты (выход)** показаны с правой стороны. *Механизм*(человек или автоматизированная система), который осуществляет операцию, представляется дугой, входящей в блок снизу (рис 2 1).



Одной из наиболее важных особенностей метода SADT является постепенное введение все больших уровней детализации по мере создания диаграмм, отображающих модель.

На рис. 2.2, где приведены четыре диаграммы и их взаимосвязи, показана структура SADT-модели. Каждый компонент модели может быть декомпозирован на другой диаграмме. Каждая диаграмма иллюстрирует "внутреннее строение" блока на родительской диаграмме.



**2.2.3. ПОСТРОЕНИЕ ИЕРАРХИИ ДИАГРАММ**

Построение SADT-модели 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.

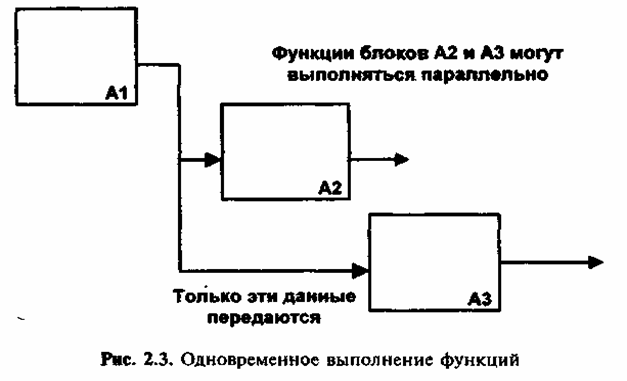
Затем блок, который представляет систему в качестве единого модуля, детализируется на другой диаграмме с помощью нескольких блоков, соединенных интерфейсными дугами. Эти блоки определяют основные подфункции исходной функции. Данная декомпозиция выявляет полный набор подфункций, каждая из которых показана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

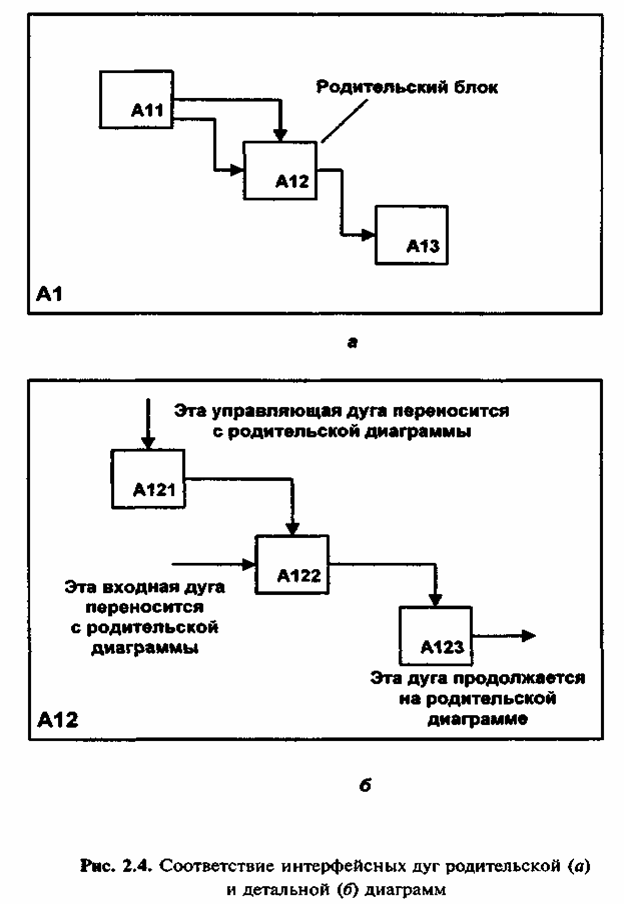
Во всех случаях каждая подфункция может содержать только те элементы, которые входят в исходную функцию. Кроме того, модель не может опустить какие-либо элементы, т.е., как уже отмечалось, родительский блок и его интерфейсы обеспечивают контекст. К нему нельзя ничего добавить, и из него не может быть ничего удалено.

Модель SADT 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.

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.

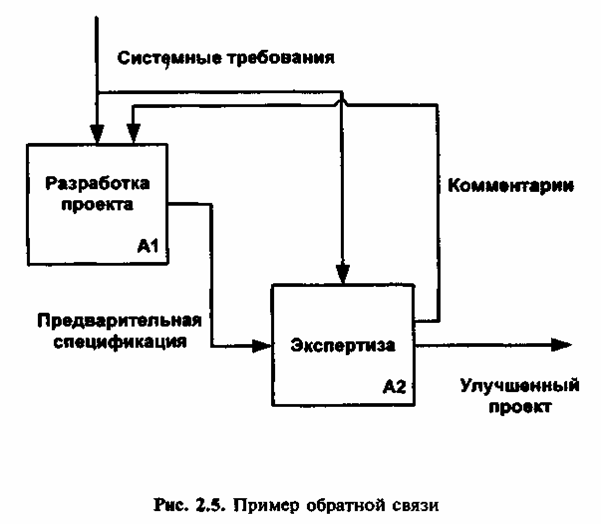
На рис. 2.3–2.5 приведены различные варианты выполнения функций и соединения дуг с блоками.



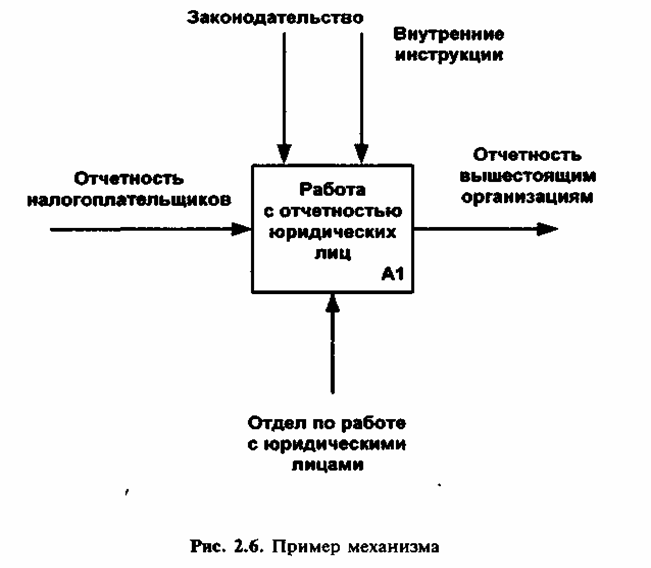


Некоторые дуги присоединены к блокам диаграммы обоими концами, у других же один конец остается неприсоединенным. Неприсоединенные дуги соответствуют входам, управлениям и выходам родительского блока. Источник или получатель этих пограничных дуг может быть обнаружен только на родительской диаграмме. Неприсоединенные концы должны соответствовать дугам на исходной диаграмме. Все граничные дуги должны продолжаться на родительской диаграмме, чтобы она была полной и непротиворечивой.

На SADT-диаграммах не указаны явно ни последовательность, ни время. Обратные связи, итерации, продолжающиеся процессы и перекрывающиеся (по времени) функции могут быть изображены с помощью дуг. Обратные связи могут выступать в виде комментариев, замечаний, исправлений и т.д. (рис. 2.5).



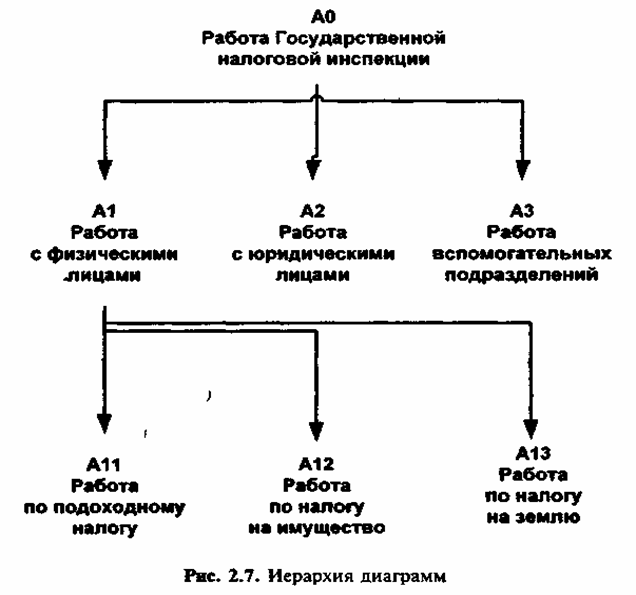
Как было отмечено, механизмы (дуги с нижней стороны) показывают средства, с помощью которых осуществляется выполнение функций. Механизм может быть человеком, компьютером или любым другим устройством, которое помогает выполнять данную функцию (рис. 2.6).



Каждый блок на диаграмме имеет свой номер. Блок любой диаграммы может быть 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, А21 является диаграммой, которая детализирует блок А21 на диаграмме А2.

Аналогично диаграмма А2 детализирует блок А2 на диаграмме АО, которая является самой верхней диаграммой модели. На рис. 2.7 показан пример дерева диаграмм.



**2.2.4. ТИПЫ СВЯЗЕЙ МЕЖДУ ФУНКЦИЯМИ**

Одним из важных моментов при моделировании бизнес-процессов организации с помощью метода SADT является точная согласованность типов связей между функциями. Различают по крайней мере связи семи типов (в порядке возрастания их относительной значимости):

· случайная;

· логическая;

· временная;

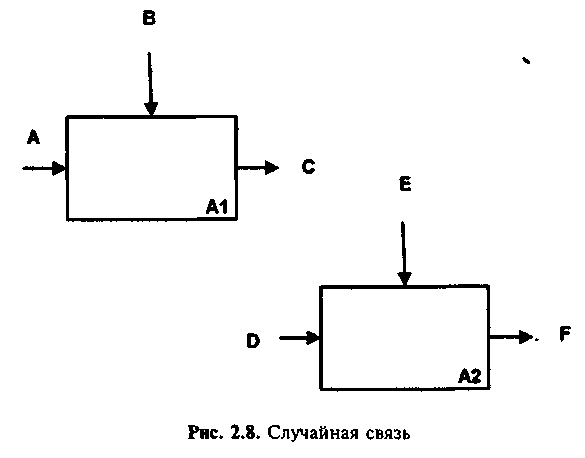
· процедурная;

· коммуникационная;

· последовательная;

· функциональная.

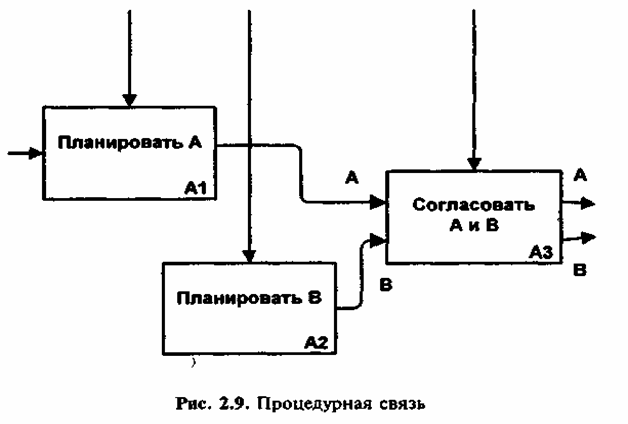
*Случайная связь —*показывает, что конкретная связь между функциями незначительна или полностью отсутствует. Это относится к ситуации, когда имена данных на SADT-дугах в одной диаграмме имеют слабую связь друг с другом. Крайний вариант этого случая показан на рис. 2.8.



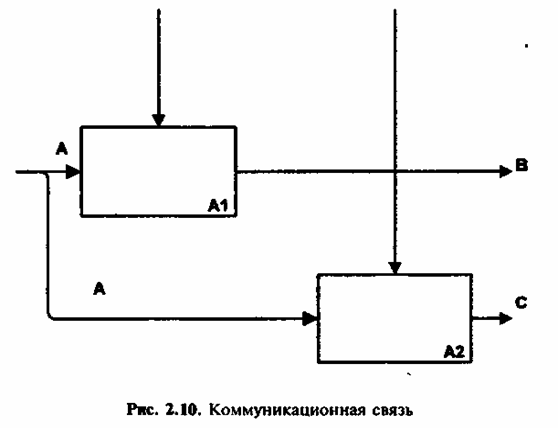
*Логическая связь —*данные и функции собираются вместе благодаря тому, что они попадают в общий класс или набор элементов, но необходимых функциональных отношений между ними не обнаруживается.

*Временная связь —*представляет функции, связанные во времени, когда данные используются одновременно или функции включаются параллельно, а не последовательно.

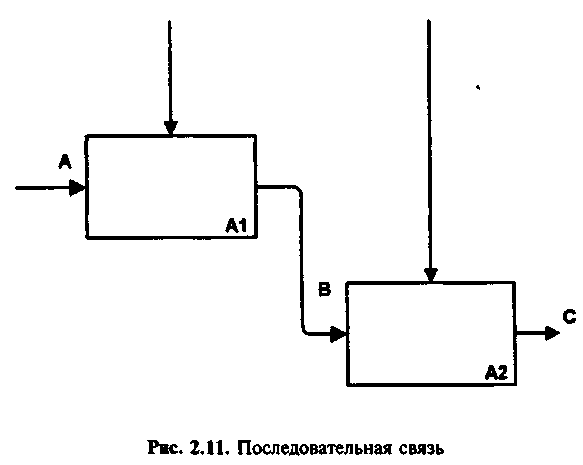
*Процедурная связь*(рис 2.9) – функции сгруппированы вместе благодаря тому, что они выполняются в течение одной и той же части цикла или процесса.



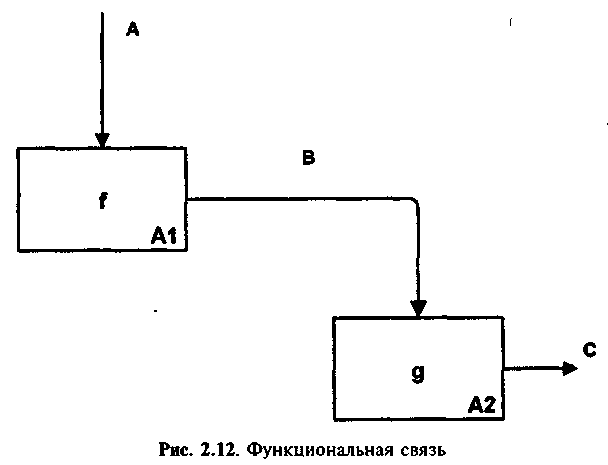
*Коммуникационная связь —*функции группируются благодаря тому, что они используют одни и те же входные данные и/или производят одни и те же выходные данные (рис. 2.10).



*Последовательная связь —*выход одной функции служит входными данными для следующей функции. Связь между элементами на диаграмме является более тесной, чем в рассмотренных выше случаях, поскольку моделируются причинно-следственные зависимости (рис. 2.11).



*Функциональная связь —*все элементы функции влияют на выполнение одной и только одной функции. Диаграмма, являющаяся чисто функциональной, не содержит чужеродных элементов, относящихся к последовательному или более слабому типу связи. Одним из способов определения функционально связанных диаграмм является рассмотрение двух блоков, связанных через управляющие дуги, как показано на рис. 2.12.



В математических терминах необходимое условие для простейшего типа функциональной связи (см. рис. 2.12) имеет следующий вид:

C = g(B) = g(f(A)).

В табл. 2.1 представлены все типы связей, рассмотренные выше. Важно отметить, что уровни 4—6 устанавливают типы связей, которые разработчики считают важнейшими для получения диаграмм хорошего качества.

*Таблица 2.1*

**Описание типов связей**

|  |  |  |  |
| --- | --- | --- | --- |
| **Уровень значимости** | **Тип связи** | **Характеристика типа связи** | |
| **для функций** | **для данных** |
| 0 | Случайная | Случайная | Случайная |
| 1 | Логическая | Функции одного и того же множества или типа (например, "редактировать все входы") | Данные одного и того же множества или типа |
| 2 | Временная | Функции одного и того же периода времени (например, "операции инициализации") | Данные, используемые в каком-либо временном интервале |
| 3 | Процедурная | Функции, работающие в одной и той же фазе или итерации (например, "первый итерации проход компилятора") | Данные, используемые во время одной и той же фазы или итерации |
| 4 | Коммуникационная | Функции, использующие одни и те же данные | Данные, на которые воздействует одна и та же деятельность |
| 5 | Последовательная | Функции, выполняющие последовательные преобразования одних и тех же данных | Данные, преобразуемые последовательными функциями |
| 6 | Функциональная | Функции, объединяемые для выполнения одной функции | Данные, связанные с одной функцией |