**Задание № 7**

**Методология функционального моделирования SADT**

**(Structured Analysis and Design Technique)**

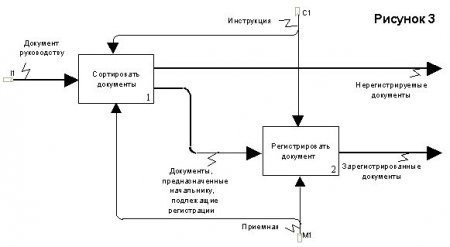
**Функциональна модель IDEF0** — методология и графическая нотация, предназначенная для формализации и описания бизнес-процессов. Отличительной особенностью IDEF0 является её акцент на соподчинённость объектов. В IDEF0 рассматриваются логические отношения между работами, а не их временна́я последовательность (WorkFlow).Также отображаются все сигналы управления, которые на ПДП не отображались. Данная модель является одной из самых прогрессивных моделей и используется при организации бизнес проектов и проектов, основанных на моделировании всех процессов как административных, так и организационных.

Методология SADT может быть использована для **анализа функций**, выполняемых системой, а также для указания **механизмов**, посредством которых они осуществляются.

Методология SADT представляет собой совокупность **методов, 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

**Задание № 7.1**

Построить **верхнего уровня иерархии функциональную модель** объекта проектирования и определить подсистемы проектируемой ИС.



Функциональная модель состоит из **диаграмм, фрагментов текстов, рисунков** и **глоссария**, имеющих ссылки друг на друга.

**Текст** обычно представляет собой рассказ об одной из частей диаграммы.

**Рисунки** - это картинки, поясняющие отдельные моменты.

**Глоссарий**- набор определений объектов и функций, представленных на диаграмме.

**Диаграммы** - главные компоненты модели, все функции ИС и интерфейсы на них представлены как **блоки** и дуги. Место соединения дуги с блоком определяет тип интерфейса. Управляющая информация входит в блок сверху, в то время как информация, которая подвергается обработке, показана с левой стороны блока, а результаты выхода показаны с правой стороны. **Механизм** (человек или автоматизированная система), который осуществляет операцию, представляется дугой, входящей в блок снизу

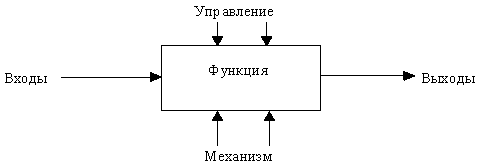


Рис.1. Функциональный блок и интерфейсные дуги.

Построение SADT-модели заключается в выполнении следующих действий:

• сбор информации об объекте, определение его границ;

• определение цели и точки зрения модели;

• построение, обобщение и декомпозиция диаграмм;

• критическая оценка, рецензирование и комментирование.

**Задание № 7.2**

Выбрать **стратегию декомпозиции** и осуществить декомпозицию функциональной модели верхнего уровня иерархии

**Стратегии декомпозиции**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, однако, 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 (таких, например, как работа промышленного предприятия), 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. Одна из наиболее частых проблем, возникающих в процессе построения SADT-моделей, — когда же следует завершить построение конкретной модели? На этот вопрос не всегда легко ответить, хотя существуют некоторые эвристики для определения разумной степени полноты. Здесь представлены правила, которыми пользуются опытные аналитики для определения момента завершения моделирования. Они носят характер рекомендаций. Только длительная практика позволит приобрести знания, необходимые для принятия правильного решения об окончании моделирования. Рекомендуется прекращать моделирование, когда уровень детализации модели удовлетворяет ее цель. 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. Проверить достаточность деталей обычно совсем легко, необходимо просто спросить себя, отвечает ли блок на все или на часть вопросов, составляющих цель модели. Если блок помогает ответить на один или более вопросов, 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 (т.е. изменение способа описания системы). В SADT 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 (т.е. сходные функции уточняются согласованным образом).

• Блок представляет тривиальную функцию. Тривиальная функция — это такая функция, понимание которой не требует никаких объяснений. В этом случае очевидна целесообразность отказа от декомпозиции, потому что роль SADT заключается в превращении сложного вопроса в понятный, а не в педантичной разработке очевидных деталей. В таких случаях декомпозиция определенных блоков может принести больше вреда, чем пользы. Тривиальные функции лучше всего описываются небольшим объемом текста. Следует заметить, что "тривиальный" не означает "бесполезный". Тривиальные функции выполняют очень важную роль, поясняя работу более сложных функций, а иногда и соединяя вместе основные подсистемы. Поэтому при анализе не следует пропускать тривиальные функции. Наоборот, их существование должно быть зафиксировано и они должны быть детализированы, как и любые другие функции. Однако следует предостеречь от больших затрат времени на анализ тривиальных функций системы. Усиленное внимание к мелочам может привести к созданию модели, которой будет недоставать абстракции, что сделает ее трудной для понимания и использования. Общее число уровней в модели (включая контекстный) не должно превышать 5-6. Практика показывает, что этого вполне достаточно для построения полной функциональной модели современного предприятия любой отрасли.

**Этапы построения функциональной модели**

1.Построение SADT-модели начинается с представления всей системы в виде простейшей компоненты - одного блока и дуг, изображающих интерфейсы с функциями вне системы. Поскольку единственный блок представляет всю систему как единое целое, имя, указанное в блоке, является общим. Это верно и для интерфейсных дуг - они также представляют полный набор внешних интерфейсов системы в целом.

2.Затем блок, который представляет систему в качестве единого модуля, **детализируется** на другой диаграмме с помощью нескольких блоков, соединенных интерфейсными дугами. Эти блоки представляют основные **подфункции** исходной **функции**. Данная декомпозиция выявляет полный набор подфункций, каждая из которых представлена как блок, границы которого определены интерфейсными дугами. Каждая из этих подфункций может быть декомпозирована подобным образом для более детального представления.

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 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. 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.

Каждый блок на диаграмме имеет свой номер. Для того, чтобы указать положение любой диаграммы или блока в иерархии, используются номера диаграмм. Например, А21 является диаграммой, которая детализирует блок 1 на диаграмме А2. Аналогично, А2 детализирует блок 2 на диаграмме А0, которая является самой верхней диаграммой модели.

|  |
| --- |
|  |

Рис. 2. Иерархия диаграмм

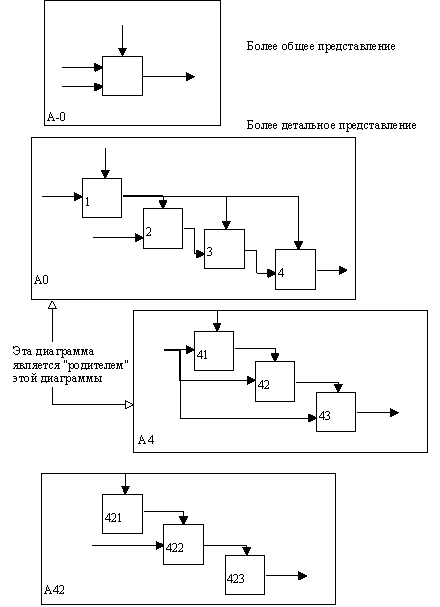


Рис. 3. Структура SADT-модели. Декомпозиция диаграмм

На SADT-диаграммах не указаны явно ни последовательность выполнения функций, ни время.

Таблица. Синтаксис диаграмм

|  |  |  |
| --- | --- | --- |
|  | Блок | Дуга |
| Графический символ | Функциональные блоки изображаются прямоугольниками. | Дуги на SADT-диаграмме изображаются одинарными линиями со стрелками на концах. |
| Текстовое обозначение | Поскольку блоки отображают функции системы, то в названии блоков используют глаголы или глагольные обороты (рассчитать, выполнить задание и т.д.). | Дуги изображают объекты (или данные), поэтому они описываются существительными или существительными с определениями (набор инструментов, чертеж и т.д.) |
| Особенности использования | Блоки на SADT-диаграмме никогда не размещаются случайным образом. Они размещаются по степени важности, как ее понимает автор диаграммы. В SADT этот относительный порядок называется ***доминированием***. Доминирование понимается как влияние, которое один блок оказывает на другие блоки диаграммы. . Наиболее доминирующий блок обычно располагается в верхнем левом углу диаграммы, а наименее доминирующий - в правом нижнем углу диаграммы. | 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.  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.  Дуга в SADT редко изображает один объект. Обычно она символизирует набор объектов. Например, дуга "рабочий комплект" отражает "техническое задание", "чертеж", "план-график", некоторое "сырье и заготовки". Так как дуги представляют собой наборы объектов, они могут иметь множество начальных точек (источников) и конечных точек (назначений). Поэтому дуги могут разветвляться и соединяться различными сложными способами. Пример: дуга принятое задание разветвляется на дугу штамп "принято", которая является управляющей информацией для блока "управлять выполнением задания" и дугу деталь с биркой, которая является входной для того же блока. |

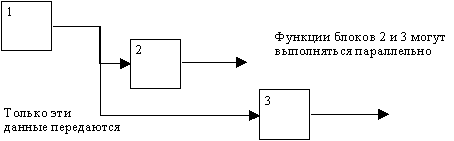


Рис. 4. Одновременное выполнение

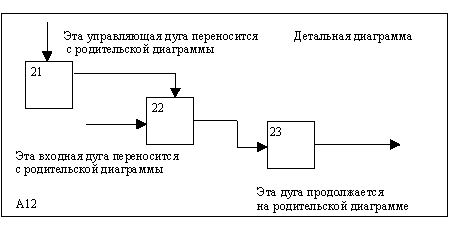
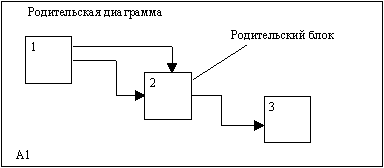


Рис. 5. Соответствие должно быть полным и непротиворечивым

|  |
| --- |
|  |

Рис. 6. Пример обратной связи

|  |
| --- |
|  |

Рис. 7. Пример механизма

|  |
| --- |
|  |

Рис. 8. Пример SADT-диаграммы

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Типы связей между функциями Одним из важных моментов при проектировании ИС с помощью методологии SADT является точная согласованность типов связей между функциями. Различают по крайней мере семь типов связывания:   |  |  | | --- | --- | | **Тип связи** | **Относительная значимость** | | Случайная | 0 | | Логическая | 1 | | Временная | 2 | | Процедурная | 3 | | Коммуникационная | 4 | | Последовательная | 5 | | Функциональная | 6 |   **(0) Тип случайной связности**: наименее желательный.  Случайная связность возникает, когда конкретная связь между функциями мала или полностью отсутствует. Это относится к ситуации, когда имена данных на SADT-дугах в одной диаграмме имеют малую связь друг с другом. Крайний вариант этого случая показан на рисунке 9.  Рис. 9. Случайная связность  **(1) Тип логической связности.** Логическое связывание происходит тогда, когда данные и функции собираются вместе вследствие того, что они попадают в общий класс или набор элементов, но необходимых функциональных отношений между ними не обнаруживается.  **(2) Тип временной связности.** Связанные по времени элементы возникают вследствие того, что они представляют функции, связанные во времени, когда данные используются одновременно или функции включаются параллельно, а не последовательно.  **(3) Тип процедурной связности.** Процедурно-связанные элементы появляются сгруппированными вместе вследствие того, что они выполняются в течение одной и той же части цикла или процесса. Пример процедурно-связанной диаграммы приведен на рисунке 10.  Рис. 10. Процедурная связность  **(4) Тип коммуникационной связности.** Диаграммы демонстрируют коммуникационные связи, когда блоки группируются вследствие того, что они используют одни и те же входные данные и/или производят одни и те же выходные данные (рисунок 11).  **(5) Тип последовательной связности.** На диаграммах, имеющих последовательные связи, выход одной функции служит входными данными для следующей функции. Связь между элементами на диаграмме является более тесной, чем на рассмотренных выше уровнях связок, поскольку моделируются причинно-следственные зависимости (рисунок 12).  **(6) Тип функциональной связности.** Диаграмма отражает полную функциональную связность, при наличии полной зависимости одной функции от другой. Диаграмма, которая является чисто функциональной, не содержит чужеродных элементов, относящихся к последовательному или более слабому типу связности. Одним из способов определения функционально-связанных диаграмм является рассмотрение двух блоков, связанных через управляющие дуги, как показано на рисунке 13.  Рис. 11. Коммуникационная связность  Рис. 12. Последовательная связность  В математических терминах необходимое условие для простейшего типа функциональной связности, показанной на рисунке 13, имеет следующий вид:  C = g(B) = g(f(A))  Ниже в таблице представлены все типы связей, рассмотренные выше. Важно отметить, что уровни 4-6 устанавливают типы связностей, которые разработчики считают важнейшими для получения диаграмм хорошего качества.  Рис. 13. Функциональная связность   |  |  |  |  | | --- | --- | --- | --- | | **Значимость** | **Тип связности** | **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 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 | |

**Основные элементы методологии**

Основные элементы методологии 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 описываются 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 организационной структуры на функциональную модель.