Лекция 15-18

Темы:

15Модели анализа и определения спецификаций. Компоненты полной спецификации методологии структурного анализа.

16 Диаграммы переходных состояний SDT

17 Диаграммы переходных состояний SDT. Разбор ошибок в построении схем.

18Функциональные диаграммы.

Все функциональные спецификации разрабатываемого программного обеспечения описывают перечень функций и состав обрабатываемых данных. Они различаются только системой приоритетов (акцентов), которая используется разработчиком в процессе анализа требований и определения спецификаций.

Так, диаграммы переходов состояний определяют некоторые аспекты поведения программного обеспечения во времени, диаграммы потоков данных — направление и структуру потоков данных, а концептуальные диаграммы классов — отношение между основ ными понятиями предметной области.

На рис. 2.1 показана классификация моделей, используемых в качестве спецификаций разрабатываемого программного обеспечения. **В рамках структурного подхода на этапе анализа и определения спецификаций используют три типа моделей**:

1.ориентированные на функции,

2.ориентированные на данные

3. ориентированные на потоки данных.

Каждый тип модели целесообразно использовать для своего специфического класса программных разработок. Разные модели описывают проектируемое программное обеспечение с разных сторон.

Рис. 2.1. Классификация моделей программного обеспечения, используемых на этапе определения спецификаций

Методологии структурного анализа и проектирования, основанные на моделировании потоков данных, обычно используют комплексное представление проектируемого программного обеспечения в виде совокупности следующих моделей:

■ диаграммы переходов состояний (SDT — State Transition Diagrams), характеризующие поведение системы во вре мени;

■ функциональные диаграммы (SADT — Structured Analisis and Design Technique);

■ диаграммы потоков данных (DFD — Data Flow Diagrams), описывающие взаимодействие источников и потребите лей информации через процессы, которые должны быть реализованы в системе;

■ диаграммы «сущность—связь» (ERD — Entity—Relation ship Diagrams), описывающие базы данных разрабатыва емой системы.

Компоненты спецификации методологий структурного анализа приведены на рис. 2.2. Спецификацию процесса обычно представляют в виде кратко го текстового описания, схем алгоритмов, псевдокодов. Словарь данных — это краткое описание основных понятий, используемых при составлении спецификаций.

Часть 16

диаграммы переходов состояний SDT

State & Transition Diagram (сокращенно S&T) — схема состояний и переходов. Техника для визуализации ТЗ. Она наглядно показывает, как некий объект переходит из одного состояния в другое.

Вот объект находился в состоянии A, потом произошло какое-то действие, и он попал в состояние B. Потом он попадет в состояние C и другие... Принцип не меняется, было одно состояние, стало другое.

Диаграммы переходов состояний (*STD*) предназначены для моделирования и документирования аспектов систем, зависящих от времени или реакции на событие. Они позволяют осуществлять декомпозицию управляющих процессов и описывают отношения между входными и выходными управляющими потоками для управляющего процесса-предка.

С помощью STD-диаграмм можно моделировать последующее функционирование системы на основе ее предыдущего и текущего функционирования. **Моделируемая система в любой заданный момент времени находится точно в одном из конечного множества состояний**.

Объекты std

Состояние - условие устойчивости для системы: способность системы сохранять свои функции без их произвольного изменения. *Имя состояния* - отражает реальную ситуацию, в которой находится система.

Цель - определяет будущее состояние по прошлому и текущему.

Переход - определяет перемещение системы из одного состояния в другое. *Имя перехода* - идентифицирует события, которые являются причиной перехода.

Событие - состоит из какого-либо управляющего потока (внешнего, внутреннего) и происходит при выполнении некоторого условия. При этом следует отметить:

1. Не все события вызывают переходы.
2. События не всегда вызывают переходы.
3. События не всегда вызывают переход в одно и то же состояние.

Условие - событие, вызывающее переход и названое именем перехода.

С переходом из одного состояния в другое может связываться действие или совокупность действий.

Действие при переходе - операция, которая выполняется при переходе. Действие может быть или физическим, или управляющим потоком.

Пример STD диаграммы для работы банкомата приведен на рисунке 5.

На диаграмме элементы нотации обозначаются следующим образом:

- *состояния* – как узлы (например**, Ожидание)**;

- *переходы* – как дуги (например**, Корректный пароль**);

- ***условия***- идентифицируются именем перехода (например,**Корректный пароль**);

- ***действия*** - отклики на события, которые "привязываются" к переходам, записываются под условием (например,**Обеспечить требуемый сервис**).

STD имеет только одно начальное состояние. Но система может иметь большое количество завершающих состояний.

Рисунок 5 - Диаграмма переходов состояний (STD) для банкомата

Рекомендации. При построении STD желательно выполнять правила:

- строить диаграмму на наиболее высоком уровне;
- детализировать (обеспечивать иерархичность диаграмм);
- использовать те же термины (имена событий, действий, потоков,

Часть 17

Мы рисуем:

- кружочки — состояния объекта;
- стрелочки — то, благодаря чему из состояния A в состояние B.

Это действие, но его может совершить не только пользователь, но и система сама. Например, задача запустилась автоматически в 10 часов вечера.

Такая схема позволяет нам сразу визуально оценить, какие переходы вообще возможны и что надо протестировать. Ведь нам надо протестировать и эту стрелку, и эту... Так что стрелочки — это наши готовые тест-кейсы!

Схема состояний и переходов относится к техникам тест-дизайна. Значит, про неё спрашивают на собеседованиях.

Как рисовать диаграмму

Очень важно: S&T рисуется на объект! Один объект. В идеале — на объект, имеющий аналог в базе данных продукта.

Шаг 1. Вы выбираете объект в своём проекте (рабочем или учебном, не суть).

Шаг 2. Думаете, какие у него состояния. Они не должны пересекаться, то есть: объект не может быть разом в двух состояниях, и при этом он всегда хоть в каком-то одном есть

Шаг 3. Рисуете эти состояния кружочками.

Шаг 4. Соединяете их стрелочками. Стрелочки - это действия, их надо подписать.

Шаг 5. Смотрите, что получилось и анализируете, есть ли у объекта другие состояния? А другие действия между текущими состояниями? Переход на шаг 2.

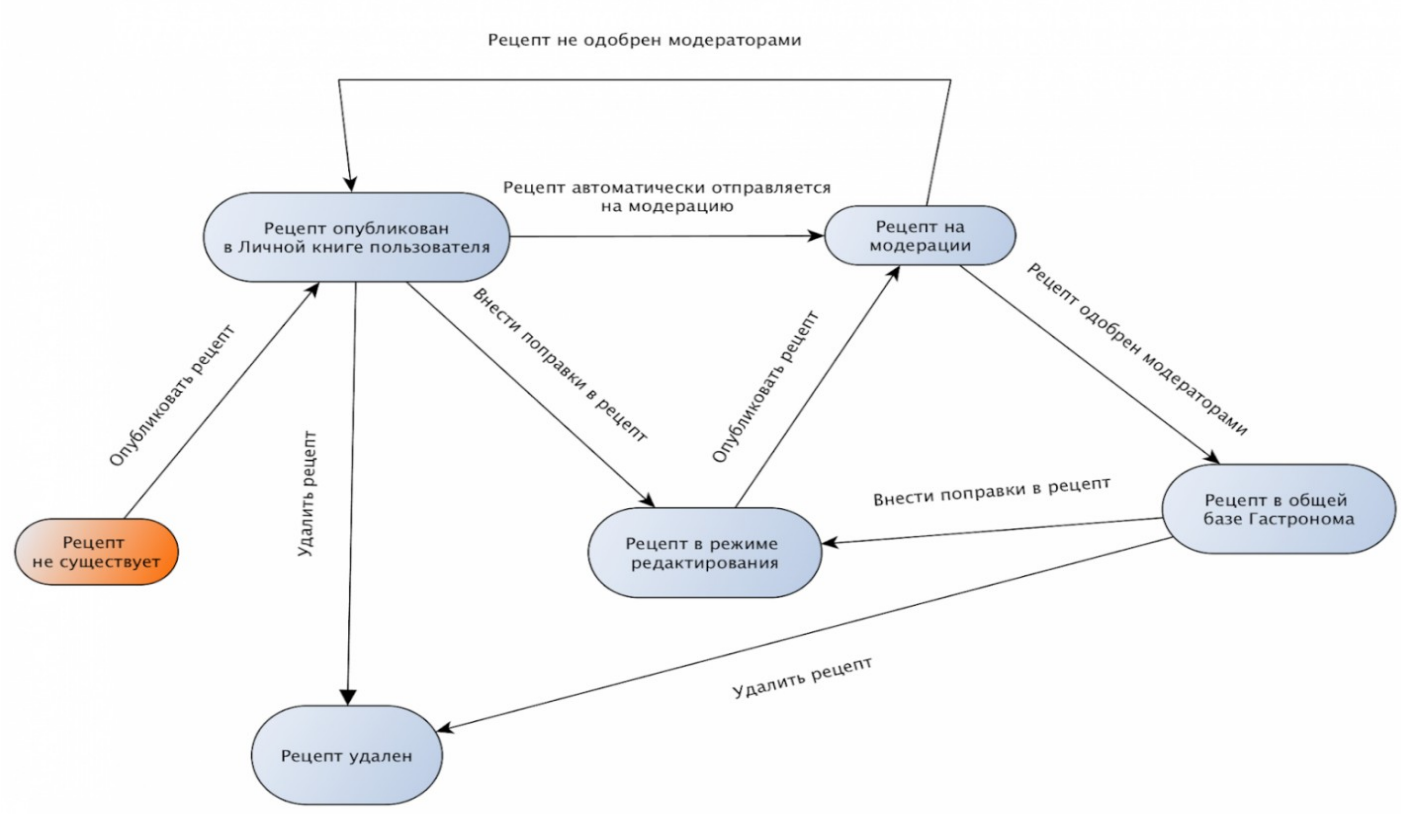
Чтобы начать, задайте себе вопросы:

1. Какой конкретно объект вы выбрали? Как он называется? *(только один!)*
2. Какие у этого объекта есть состояния?

Основное определение состояния — "набор доступных и недоступных действий с объектом". Продукт всегда должен знать, в каком состоянии каждый его объект. Вообще, когда будете думать об объектах и состояниях, старайтесь представлять их аппаратную реализацию.

Объект — это практически всегда строка в базе данных, старайтесь абстрагироваться от интерфейса вообще, и представляйте те действия, которые вы могли бы делать с объектом прямыми запросами в базу.

Вот пример хорошей диаграммы:



Типовые ошибки при составлении карты

1. Вместо объекта — GUI

Очень важно: S&T рисуется на объект! Это не зарисовка графического интерфейса «открыта страница такая, открыта страница сякая»... Если вы описываете разные странички GUI — это уже не S&T.

Зарисовывать страницы смысла обычно нет. Это как при рисовании майнд-карты — мы не рисуем графический интерфейс, мы описываем функционал. То, зачем пользователь вообще пришел на сайт. Это намного полезнее!

Другой вариант той же ошибки: искать билет — (результаты поиска) — открыть форму покупки — (форма открыта) — ввести данные кредитной карты — (данные введены).

2. Несколько объектов в одной карте

На прошлой картинке у нас несколько объектов: результаты поиска, форма, данные. И все — плохие. **Потому что там мы явно что-то покупаем, вот это «что-то» и есть объект!**

Но когда мы описываем покупку, тоже легко скатиться в несколько объектов в одной карте: "пицца в корзине", "заказ оформлен".

Товар тоже очень часто путают, потому что есть два варианта:

1. как на авито — продается конкретная вещь: "Нет на сайте", "Продается", "Продан".
2. просто "товарная позиция", как какие-нибудь носки в магазине одежды: "Отсутствует", "В наличии", "Ожидается поступление" и так далее.

Если речь о сайте типа авито, то объект лучше выбрать "объявление", будет логичнее. А вот если мы покупаем пиццу — это будет товарной позицией.

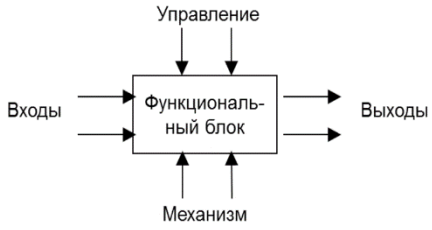
3. Несколько одинаковых состояний

Вспомните пример с тортиком:

1. Купила.
2. Поставила в холодильник на потом.
3. Передумала, достала, надкусала.
4. Снова передумала, решила съесть целиком, осилила половину.
5. Расстроилась, решила не доедать вообще и выкинуть.

Половину пунктов можно объединить. Ведь состояние торта не меняется от того, купили вы его только что или час назад, сидите любуетесь на него, переставляете с места на место или убираете в холодильник:

- 1-2 это торт куплен;
- 3-4 в процессе уничтожения;
- 5 выброшен.



Часть 18

Результатом применения методологии SADT является модель, которая состоит из диаграмм, фрагментов текстов и глоссария, имеющих ссылки друг на друга.

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

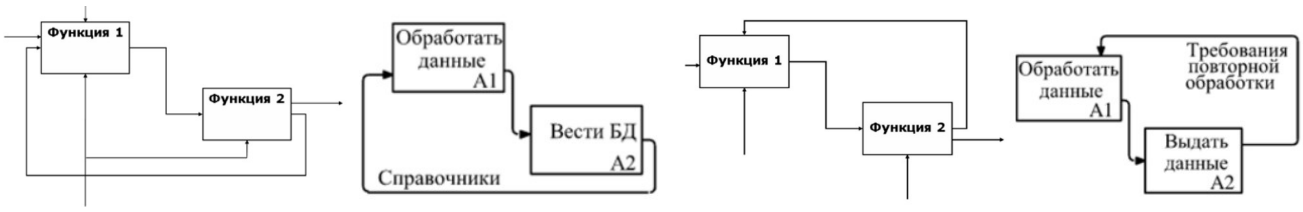
Основные элементы модели SADT**:**

1. **входа** (входят в левую грань работы) - изображают данные или объекты, изменяемые в ходе выполнения работы;
2. **управления** (входят в верхнюю грань работы) - изображают правила и ограничения, согласно которым выполняется работа;
3. **выхода** (выходят из правой грани работы) - изображают данные или объекты, появляющиеся в результате выполнения работы;
4. **механизма** (входят в нижнюю грань работы) - изображают ресурсы, необходимые для выполнения работы, но не изменяющиеся в процессе работы (например, оборудование, людские ресурсы и т.д.);
5. **вызова** (выходят из нижней грани работы) - изображают связи между разными диаграммами или моделями, указывая на некоторую диаграмму, где данная работа рассмотрена более подробно.

Таким образом, **IDEF0-модель состоит из** набора иерархически связанных диаграмм

Типы связей между блоками:

1.Выход-вход 2.**вход-управление** **3.выход-механизм**



5.обртная связь по выходу_ **4.обратная связь по упрвлению**

1.Отношение входа возникает тогда, когда выход одного блока становится входом для одного из последующих блоков. Такие отношения являются наиболее часто встречающимися связями.

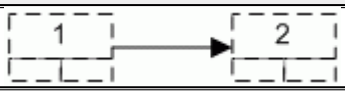
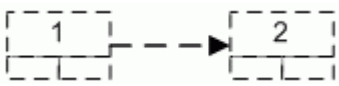
2.Отношения управления — когда выход одного блока непосредственно влияет на работу какого-либо последующего блока.

3.Связь "выход — механизм" встречается нечасто и отражает ситуацию, при которой выход одного блока становится средством достижения цели для другого блока. Данная связь характерна при распределении источников ресурсов: физического пространства, оборудования, финансирования, материалов, инструментов, обученного персонала

4,5.Обратные связи по управлению и по входу предназначены для представления итерации или рекурсии.

4.**Обратная связь по управлению** возникает тогда, когда выход некоторого блока влияет на работу какого-либо предыдущего блока.

5.**Обратная связь по входу** применяется тогда, когда нужно показать, что выход одного блока становится входом для какого- либо из предшествующих блоков, например в случае возврата проекта документа на доработку.

Название связи	Вид связи	Смысл связи
Связь предшествования		Обозначает, что вторая работа начинает выполняться после завершения первой работы.
Связь отношения		Обозначает, что вторая работа может начаться и даже закончиться до того момента, когда закончится выполнение первой работы.
Связь потоков объектов		Одновременно обозначает временную последовательность работ и материальный

		либо информационный поток. В данном случае вторая работа начинает выполняться после завершения первой работы. При этом выходом первой работы объект название которого надписано над стрелкой (в данном случае документ). Эта связь также обозначает, что объект порождаемый первой работой, используется в последующих работах.

*стрелка с двумя наконечниками показывает, что работы будет продолжаться и в других процессах /схемах, чаще всего горизонтально используется

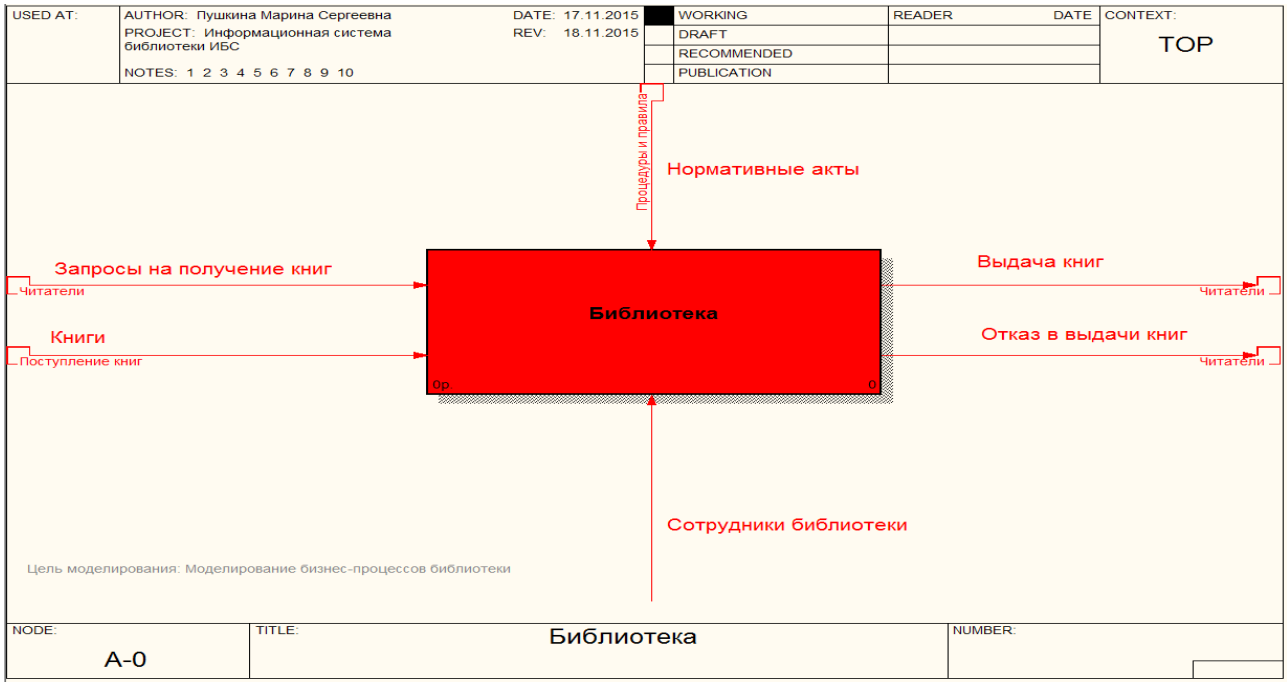
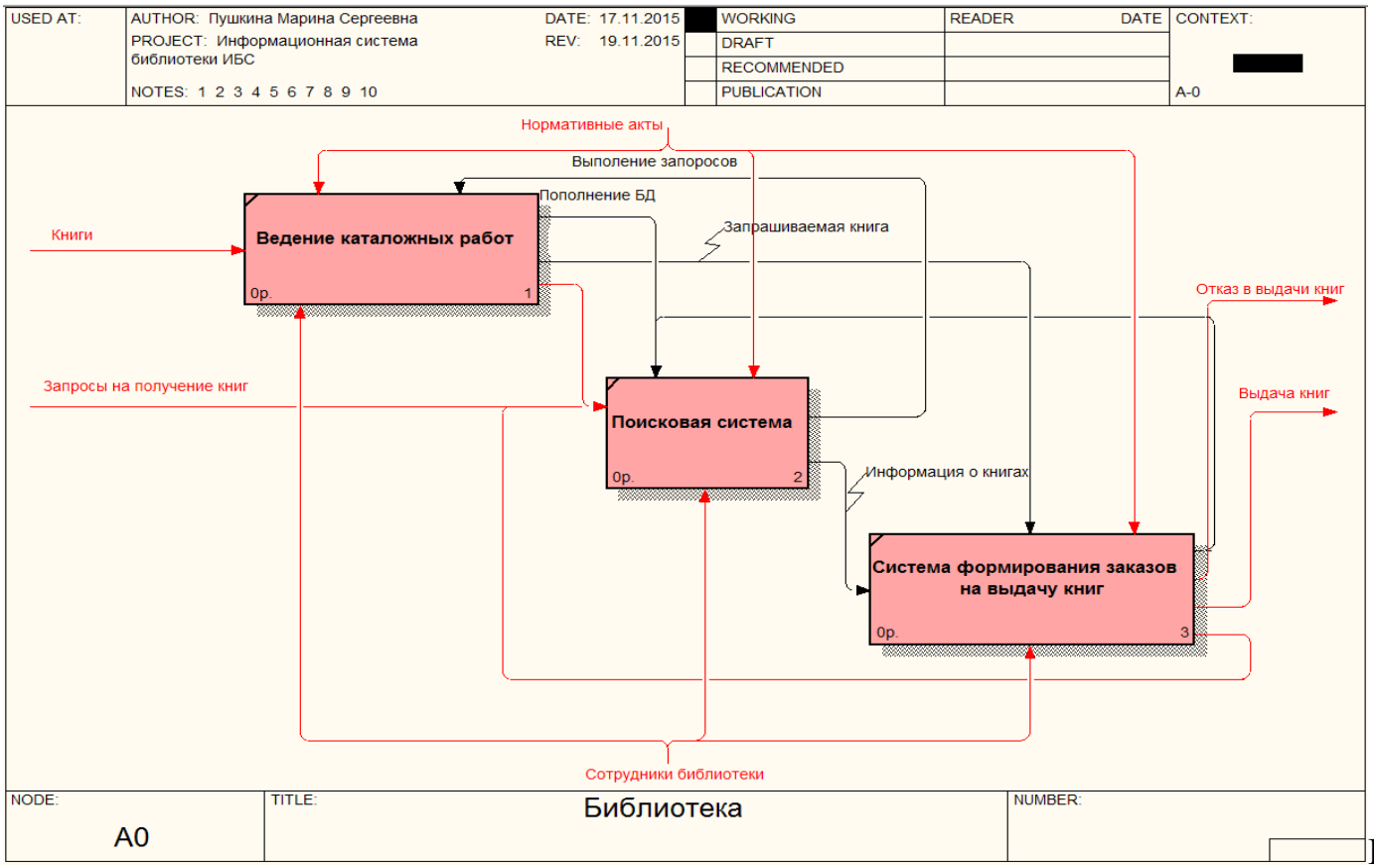


Рисунок 1.1 – Контекстная диаграмма библиотеки



Рис

унок 1.2 – Взаимодействие основных компонентов системы