

## Глава 7. Методика DFD. Диаграммы потоков данных

### 7.1. Методология DFD

Методика IDEF0 может дополняться не только диаграммами потока работ (IDEF3), но и *диаграммами потоков данных (Data Flow Diagramming, DFD)*.

Нотация диаграмм потоков данных позволяет отображать на диаграмме как шаги бизнес-процесса, так и поток документов и управления (в основном управления, поскольку на верхнем уровне описания процессных областей значение имеет передача управления). Также на диаграмме можно отображать средства автоматизации шагов бизнес-процессов. Обычно используется для отображения третьего и ниже уровня декомпозиции бизнес-процессов (первый уровень — перечень бизнес-процессов (IDEF0), а второй — функции, выполняемые в рамках бизнес-процессов (IDEF3)).

Области применения диаграмм потоков данных:

- моделирование функциональных требований к проектируемой системе;
- моделирование существующего процесса движения информации;
- описание документооборота, обработки информации;
- дополнение к модели IDEF0 для более наглядного отображения текущих операций документооборота;
- проведение анализа и определения основных направлений реинжиниринга информационной системы.

Диаграммы DFD могут дополнить то, что уже отражено в модели IDEF0, поскольку они описывают потоки данных, позволяя проследить, каким образом происходит обмен информацией как внутри системы между бизнес-функциями, так и системы в целом с внешней информационной средой.

Методика DFD удобна для описания не только бизнес-процессов (как дополнение к IDEF0), но и программных систем:

- DFD-диаграммы создавались как средство проектирования программных систем (в то время как IDEF0 — средство проектирования систем вообще), поэтому DFD имеют более богатый набор элементов, адекватно отражающих их специфику (например, хранилища данных являются прообразами файлов или баз данных);
- наличие мини-спецификаций DFD-процессов нижнего уровня позволяет преодолеть логическую незавершённость IDEF0 (моделирование обрывается на некотором достаточно низком уровне, когда дальнейшая детализация модели становится бессмысленной) и построить полную функциональную спецификацию разрабатываемой системы.

С помощью DFD-диаграмм требования к проектируемой информационной системе разбиваются на функциональные компоненты (процессы) и представляются в виде сети, связанной потоками данных. Главная цель декомпозиции DFD-функций — продемонстрировать, как каждый процесс преобразует свои входные данные в выходные, а также выявить отношения между этими процессами.

На схемах бизнес-процесса отображаются:

- функции процесса;
- входящая и исходящая информация при описании документов;
- внешние бизнес-процессы, описанные на других диаграммах;
- точки разрыва при переходе процесса на другие страницы.

При моделировании DFD система рассматривается как сеть связанных между собой функций, т. е. как совокупность сущностей. Методология основана на идее нисходящей иерархической организации. Целью является преобразование общих, неясных знаний о требованиях к системе в точные определения. Внимание фокусируется на потоках данных. Методология поддерживается традиционными нисходящими методами проектирования.

### 7.1.1. Варианты методологии DFD

Существует два основных варианта методологии DFD: *методология Гейна–Сарсона (Gane–Sarson)* и *методология Йордана–Де Марко (Yourdon–DeMarko)*.

Главной отличительной чертой методологии Гейна–Сарсона является наличие этапа моделирования данных, определяющего содержимое хранилищ данных (БД и файлов) в DFD. Этот этап включает построение списка элементов данных, располагающихся в каждом хранилище данных; анализ отношений между данными и построение соответствующей диаграммы связей между элементами данных; представление всей информации по модели в виде связанных нормализованных таблиц.

Кроме того, эти методологии отличаются нотацией.

Обе методологии основаны на простой концепции нисходящего поэтапного разбиения функций системы на подфункции:

- 1) формирование контекстной диаграммы верхнего уровня, идентифицирующей границы системы и определяющей интерфейсы между системой и окружением;
- 2) формирование списка внешних событий, на которые система должна реагировать (после интервьюирования эксперта предметной области), для каждого из которых строится *пустой процесс (Bubble)* в предположении, что его функция обеспечивает требуемую реакцию на это событие (в большинстве случаев включает генерацию выходных потоков и событий);
- 3) проведение детализации для каждого из пустых процессов.

Помимо нотаций Йордана–Де Марко и Гейна–Сарсона для элементов DFD-диаграм могут использоваться и другие условные обозначения (OMT, SSADM и т. д.). Все они обладают практически одинаковой функциональностью и различаются лишь в деталях.

Методология DFD позволяет уже на стадии функционального моделирования определить базовые требования к данным. В этом случае совместно

используются методологии DFD и IDEF1X.

Диаграммы DFD могут быть построены с использованием традиционного структурного анализа, подобно тому, как строятся диаграммы IDEF0:

- 1) строится физическая модель, отображающая текущее состояние дел;
- 2) полученная модель преобразуется в логическую модель, которая отображает требования к существующей системе;
- 3) строится модель, отображающая требования к будущей системе;
- 4) строится физическая модель, на основе которой должна быть построена новая система.

Альтернативным является подход, применяемый при создании программного обеспечения, называемый *событийным разделением (Event Partitioning)*, в котором различные диаграммы DFD выстраивают модель системы.

- 1) Логическая модель строится как совокупность процессов и документирования того, что эти процессы должны делать.
- 2) С помощью модели окружения система описывается как взаимодействующий с событиями из внешних сущностей объект. *Модель окружения (Environment Model)* обычно содержит описание цели системы, одну контекстную диаграмму и список событий. Контекстная диаграмма содержит один блок, изображающий систему в целом, внешние сущности, с которыми система взаимодействует, ссылки и некоторые стрелки, импортированные из диаграмм IDEF0 и DFD. Включение внешних ссылок в контекстную диаграмму не отменяет требования методологии чётко определить цель, область и единую точку зрения на моделируемую систему.
- 3) *Модель поведения (Behavior Model)* показывает, как система обрабатывает события. Эта модель состоит из одной диаграммы, в которой каждый блок изображает каждое событие из модели окружения, могут быть добавлены хранилища для моделирования данных, которые необходимо запоминать между событиями. Потоки добавляются для связи

с другими элементами, и диаграмма проверяется с точки зрения соответствия модели окружения.

### 7.1.2. Мини-спецификация

*Мини-спецификация* — это алгоритм описания задач, выполняемых процессами, множество всех мини-спецификаций является полной спецификацией системы. Мини-спецификации содержат номер и/или имя процесса, списки входных и выходных данных и тело (описание) процесса, являющееся спецификацией алгоритма или операции, трансформирующей входные потоки данных в выходные.

Проектные спецификации строятся по DFD и их мини-спецификациям автоматически. Наиболее часто для описания проектных спецификаций используется методика структурных карт Джексона, иллюстрирующая иерархию модулей, связи между ними и некоторую информацию об их исполнении (последовательность вызовов, итерацию). Существует ряд методов автоматического преобразования DFD в структурные карты.

## 7.2. Синтаксис и семантика моделей DFD

Диаграммы потоков данных применяются для графического представления (Flowchart) движения и обработки информации. Обычно диаграммы этого типа используются для проведения анализа организации информационных потоков и для разработки информационных систем. Каждый блок в DFD может разворачиваться в диаграмму нижнего уровня, что позволяет на любом уровне абстрагироваться от деталей.

DFD-диаграммы моделируют функции, которые система должна выполнять, но почти ничего не сообщают об отношениях между данными, а также о поведении системы в зависимости от времени — для этих целей используются диаграммы сущность-связь и диаграммы переходов состояний.

Основные объекты нотации DFD:

- *блоки (Blocks)* или *работы (Activities)* — отображают процессы обработки и изменения информации;
- *стрелки (Arrows)* или *потоки данных (Data Flow)* — отображают информационные потоки;
- *хранилища данных (Data Store)* — отображают данные, к которым осуществляется доступ; эти данные используются, создаются или изменяются работами;
- *внешние ссылки (External References)* или *внешние сущности (External Entity)* — отображают объекты, с которыми происходит взаимодействие.

*Работы DFD* — графическое изображение операции по обработке или преобразованию информации (данных). Входная информация преобразуется в выходную. Работы именуются глаголами в неопределённой форме с последующим дополнением.

По нотации Гейна–Сарсона DFD-блок изображается прямоугольником со скруглёнными углами (рис. 7.1). Каждый блок должен иметь уникальный номер для ссылки на него внутри диаграммы. Номер каждого блока может включать префикс, номер родительского блока (A) и номер объекта, представляющий собой уникальный номер блока на диаграмме. Например, работа может иметь номер A.12.4.



Рис. 7.1. Работа DFD

Во избежание пересечений линий потоков данных один и тот же элемент

может на одной и той же диаграмме отображаться несколько раз; в таком случае два или более прямоугольников, обозначающих один и тот же элемент, могут идентифицироваться линией, перечёркивающей нижний правый угол.

*Поток данных* — механизм, использующийся для моделирования передачи информации между участниками процесса информационного обмена (функциями, хранилищами данных, внешними ссылками). По нотации Гейна–Сарсона поток данных изображается стрелкой между двумя объектами DFD-диаграммы, предпочтительно горизонтальной и/или вертикальной, причём направление стрелки указывает направление потока. Каждая стрелка должна иметь источник и цель. В отличие от стрелок IDEF0-диаграммы, стрелки DFD могут входить или выходить из любой стороны блока.

Стрелки описывают, как объекты (включая данные) двигаются из одной части системы в другую. Поскольку в DFD каждая сторона блока не имеет чёткого назначения, в отличие от блоков IDEF0-диаграммы, стрелки могут подходить и выходить из любой грани.

В DFD-диаграммах для описания диалогов типа команды-ответа между операциями применяются двунаправленные стрелки между функцией и внешней сущностью и/или между внешними сущностями. Стрелки могут сливаться и разветвляться, что позволяет описать декомпозицию стрелок. Каждый новый сегмент сливающейся или разветвляющейся стрелки может иметь собственное имя.

Иногда информация может двигаться в одном направлении, обрабатываться и возвращаться обратно. Такая ситуация может моделироваться либо двумя различными потоками, либо одним двунаправленным потоком. На поток данных можно ссылаться, указывая процессы, сущности или накопители данных, которые поток соединяет. Каждый поток должен иметь имя, расположенное вдоль или над стрелкой, выбранное таким образом, чтобы в наибольшей степени передавать смысл содержания потока пользователям, которые будут рассматривать диаграмму потоков данных.

*Хранилище данных* — графическое представление потоков данных, им-

портируемых/экспортируемых из соответствующих баз данных, изображает объекты в покое. Является неким прообразом базы данных информационной системы организации.

Хранилища данных предназначены для изображения некоторых абстрактных устройств для хранения информации, которую можно в любой момент времени поместить или извлечь из них, безотносительно к их конкретной физической реализации.

Хранилища данных используются:

- в материальных системах (там, где объекты ожидают обработки, например в очереди);
- в системах обработки информации для моделирования механизмов сохранения данных с целью дальнейших операций.

По нотации Гейна–Сарсона хранилище данных обозначается двумя горизонтальными линиями, замкнутыми с одного края (рис. 7.2). Каждое хранилище данных должно идентифицироваться для ссылки буквой D и произвольным числом в квадрате с левой стороны, например D5.

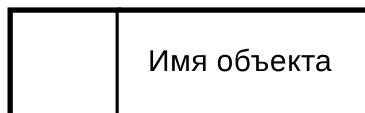


Рис. 7.2. Хранилище данных DFD

В модели может быть создано множество вхождений хранилищ данных, каждое из которых может иметь одинаковое имя и ссылочный номер. Для того чтобы не усложнять диаграмму потоков данных пересечениями линий, можно изображать дубликаты накопителя данных дополнительными вертикальными линиями с левой стороны квадрата. Хранилища данных не изменяют потоки данных, а служат только для хранения поступающих объектов.

*Внешняя сущность* — объект диаграммы потоков данных, являющийся источником или приёмником информации извне модели. Внешние сущности изображают входы и/или выходы, т. е. обеспечивают интерфейс с внешними

объектами, находящимися вне моделируемой системы. Внешними ссылками системы обычно являются логические классы предметов или людей, представляющие собой источник или приёмник сообщений, например, заказчики, конструкторы, технологи, производственные службы, кладовщики и т. д.

По нотации Гейна–Сарсона пиктограмма внешней ссылки представляет собой оттенённый прямоугольник, верхняя левая сторона которого имеет двойную толщину и обычно располагается на границах диаграммы (рис. 7.3). Внешняя ссылка может идентифицироваться строчной буквой Е в левом верхнем углу и уникальным номером, например Е5. Кроме того, внешняя ссылка имеет имя.



Рис. 7.3. Внешняя сущность DFD

Одна и та же внешняя ссылка может быть использована многократно на одной или нескольких диаграммах. Обычно к такому приёму прибегают для того, чтобы не рисовать слишком длинных и запутанных стрелок. Каждая внешняя сущность имеет префикс.

При рассмотрении системы как внешней функции часто указывается, что она находится за пределами границ моделируемой системы. После проведения анализа некоторые внешние ссылки могут быть перемещены внутрь диаграммы потоков данных рассматриваемой системы или, наоборот, какая-то часть функций системы может быть вынесена и рассмотрена как внешняя ссылка. Преобразования потоков данных во внешних сущностях игнорируются.

*Межстраничные ссылки (Off-Page Reference)* — инструмент нотации DFD, описывающий передачу данных или объектов с одной диаграммы модели на

другую. Стрелка межстраничной ссылки имеет идентифицирующее имя, номер и изображение окружности.

Помимо этого, для каждого информационного потока и хранилища определяются связанные с ними элементы данных. Каждому элементу данных присваивается имя. Также для него может быть указан тип данных и формат. Именно эта информация является исходной на следующем этапе проектирования — построении модели «сущность–связь». При этом, как правило, информационные хранилища преобразуются в сущности. Проектировщику остаётся только решить вопрос с использованием элементов данных, не связанных с хранилищами.

В качестве примера построим схему бизнес-процесса «Выдача диплома» (рис. 7.4).

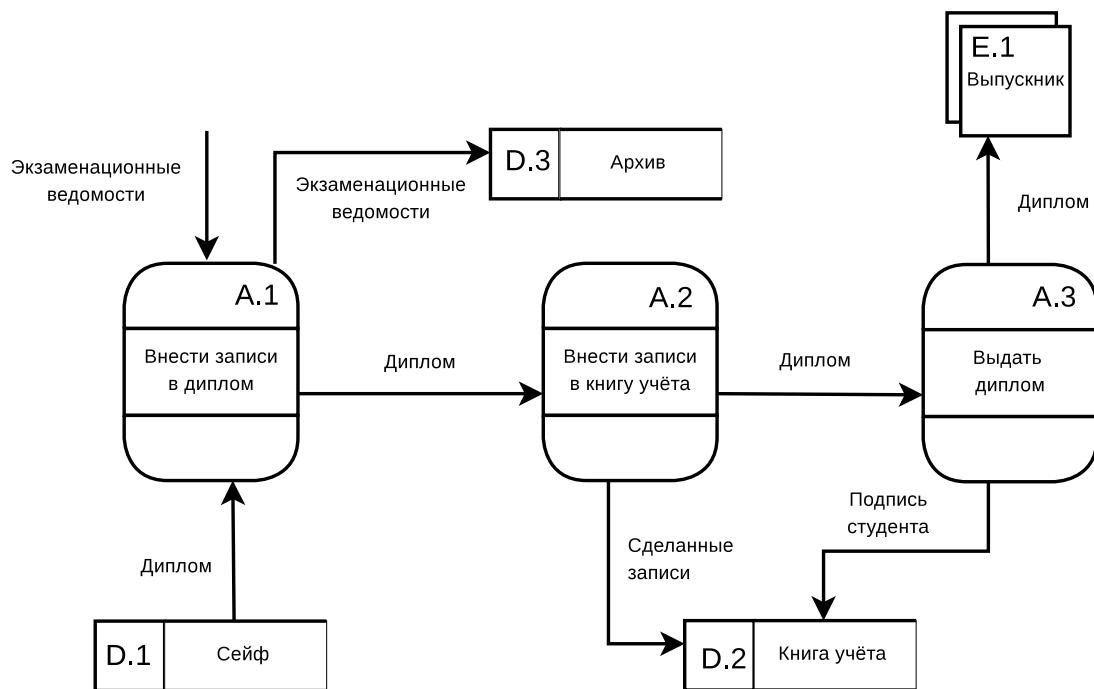


Рис. 7.4. DFD-схема «Выдача диплома»