Лабораторная работа № 2

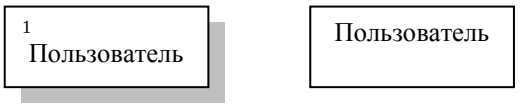
Тема: *разработка моделей информационных систем по методологии потоков данных*
Цель: *изучить методологию моделирования потоков данных, освоить CASE-средство AllFusion Process Modeller для построения диаграмм потоков данных, научиться строить модели информационных систем, используя данную методологию*

Краткая теория

Диаграммы потоков данных (Data Flow Diagram – DFD) моделируют систему как набор действий, соединенных друг с другом стрелками. Диаграммы потоков данных также могут содержать два новых типа объектов: объекты, собирающие и хранящие информацию – хранилища данных и внешние сущности – объекты, которые моделируют взаимодействие с теми частями системы (или другими системами), которые выходят за границы моделирования. В отличие от стрелок в IDEF0, которые иллюстрируют отношения, стрелки в DFD показывают, как объекты (включая и данные) реально перемещаются от одного действия к другому. Это представление потока вкупе с хранилищами данных и внешними сущностями обеспечивает отражение в DFD-моделях таких физических характеристик системы, как движение объектов (потоки данных), хранение объектов (хранилища данных), источники и потребители объектов (внешние сущности). Построение DFD-диаграмм в основном ассоциируется с разработкой программного обеспечения, поскольку нотация DFD изначально была разработана для этих целей. Графические изображения объектов на DFD-диаграммах в различных методологиях могут различаться. В данной лекции будут рассматриваться две основные нотации: Гейна – Сарсона (Gane – Sarson) и Йордана – Де Марко (Yourdon – DeMarco).

В соответствии с данным методом модель системы определяется как иерархия диаграмм потоков данных, описывающих асинхронный процесс преобразования информации от ее ввода в систему до выдачи потребителю. Источники информации (внешние сущности) порождают информационные потоки (потоки данных), переносящие информацию к подсистемам или процессам. Те, в свою очередь, преобразуют информацию и порождают новые потоки, которые переносят информацию к другим процессам или подсистемам, накопителям данных или внешним сущностям – потребителям информации. Диаграммы верхних уровней иерархии (контекстные диаграммы) определяют основные процессы или подсистемы с внешними входами и выходами. Они детализируются при помощи диаграмм нижнего уровня. Такая декомпозиция продолжается, создавая многоуровневую иерархию диаграмм, до тех пор, пока не будет достигнут уровень декомпозиции, на котором детализировать процессы далее не имеет смысла. На диаграммах потоков данных могут присутствовать следующие виды элементов: внешние сущности, процессы, накопители данных, потоки данных.

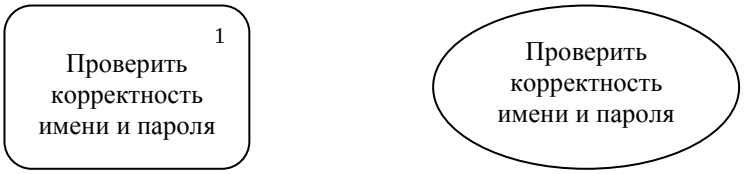
Внешняя сущность представляет собой материальный объект или физическое лицо, источник или приемник информации (например, заказчики, персонал, поставщики, клиенты, склад). Определение некоторого объекта или системы в качестве внешней сущности указывает на то, что она находится за пределами границ анализируемой системы. В процессе анализа некоторые внешние сущности могут быть перенесены внутрь диаграммы анализируемой системы, если это необходимо, или, наоборот, часть процессов может быть вынесена за пределы диаграммы и представлена как внешняя сущность. Графическое изображение внешней сущности в нотациях Гейна – Сарсона и Йордана – Де Марко приведено на рисунке 1.



а) б)

Рисунок 1 – Графическое изображение внешней сущности
а) – нотация Гейна – Сарсона, б) – нотация Йордана – Де Марко

Процесс представляет собой преобразование входных потоков данных в выходные в соответствии с определенным алгоритмом. Физически процесс может быть реализован различными способами: это может быть подразделение организации (отдел), выполняющее обработку входных документов и выпуск отчетов; программа; аппаратно реализованное логическое устройство и т.д. Процесс на диаграмме потоков данных изображается, как показано на рисунке 2. Номер процесса служит для его идентификации. В поле имени вводится наименование процесса в виде предложения с активным недвусмысленным глаголом в неопределенной форме (вычислить, рассчитать, проверить, определить, создать, получить), за которым следуют существительные в винительном падеже.



а) б)

Рисунок 2 – Графическое изображение процесса
а) – нотация Гейна – Сарсона, б) – нотация Йордана – Де Марко

В некоторых CASE-средствах используется модификация нотации Гейна – Сарсона, приведенная на рисунке 3. В этой нотации в нижней секции указывается механизм, выполняющий данный процесс.

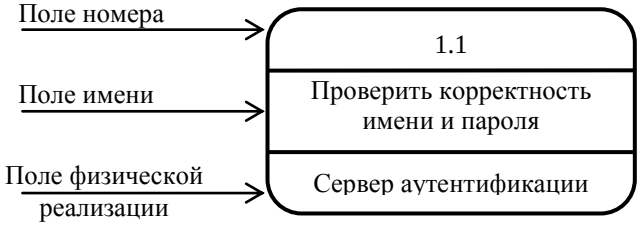


Рисунок 3 – Модифицированная нотация Гейна – Сарсона для изображения процесса

Накопитель данных – это абстрактное устройство для хранения информации, которую можно в любой момент поместить в накопитель и через некоторое время извлечь, причем способы помещения и извлечения могут быть любыми. Накопитель данных может быть реализован физически в виде ящика в картотеке, таблицы в базе данных, файла на носителе информации и т.д. Накопитель данных на диаграмме потоков данных изображается, как показано на рисунке 4.

а) б)

Рисунок 4 – Графическое изображение хранилища данных
а) – нотация Гейна – Сарсона, б) – нотация Йордана – Де Марко

Накопитель данных в нотации Гейна – Сарсона идентифицируется буквой «D» и номером, уникальным в пределах модели. Имя накопителя выбирается из соображения наибольшей информативности для проектировщика. Накопитель данных в общем случае является прообразом будущей базы данных, и описание хранящихся в нем данных должно соответствовать информационной модели.

Поток данных определяет информацию, передаваемую через некоторое соединение от источника к приемнику. Реальный поток данных может быть информацией, передаваемой по кабелю между двумя устройствами, пересылаемыми по почте письмами, магнитными лентами

или дискетами, переносимыми с одного компьютера на другой, и т.д. Поток данных на диаграмме изображается линией, оканчивающейся стрелкой, которая показывает направление потока (рисунок 5). Каждый поток данных имеет имя, отражающее его содержание.

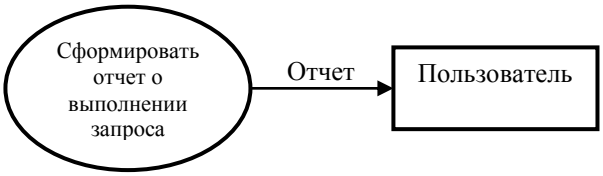


Рисунок 5 – Графическое изображение потока

Поскольку все стороны обозначающего функциональный блок DFD прямоугольника равнозначны (в отличие от IDEF0), стрелки могут начинаться и заканчиваться в любой части блока. В диаграммах потоков данных также используются двунаправленные стрелки, которые нужны для отображения взаимодействия между блоками.

Стрелки на DFD-диаграммах могут быть разбиты (разветвлены) на части, и при этом каждый получившийся сегмент может быть переименован таким образом, чтобы показать декомпозицию данных, переносимых данным потоком. Стрелки могут и соединяться между собой (объединяться) для формирования так называемых комплексных объектов. Пример разделения и объединения стрелок приведен на рисунке 6.

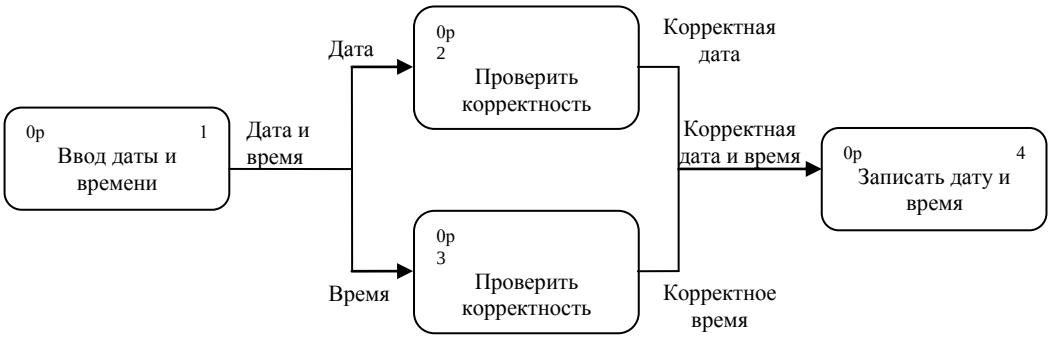


Рисунок 6 – Пример декомпозиции и объединения потока данных

Построение иерархии диаграмм потоков данных

Главная цель построения иерархии DFD заключается в том, чтобы сделать описание системы ясным и понятным на каждом уровне детализации, а также разбить его на части с точно определенными отношениями между ними. Для достижения этого целесообразно пользоваться следующими рекомендациями:

- размещать на каждой диаграмме от 3 до 6—7 процессов (аналогично IDEF0). Верхняя граница соответствует человеческим возможностям одновременного восприятия и понимания структуры сложной системы с множеством внутренних связей, нижняя граница выбрана по соображениям здравого смысла: нет необходимости детализировать процесс диаграммой, содержащей всего один или два процесса;
- не загромождать диаграммы несущественными на данном уровне деталями;
- декомпозицию потоков данных осуществлять параллельно с декомпозицией процессов. Эти две работы должны выполняться одновременно, а не одна после завершения другой;
- выбирать ясные, отражающие суть дела, имена процессов и потоков, при этом стараться не использовать аббревиатуры.

Первым шагом при построении иерархии DFD является построение контекстных диаграмм. Обычно при проектировании относительно простых систем строится единственная контекстная диаграмма со звездообразной топологией, в центре которой находится так называемый главный процесс, соединенный с приемниками и источниками информации, посредством которых с системой взаимодействуют пользователи и другие внешние системы.

Перед построением контекстной DFD необходимо проанализировать внешние события (внешние сущности), оказывающие влияние на функционирование системы. Количество потоков на контекстной диаграмме должно быть по возможности небольшим, поскольку каждый из них может быть в дальнейшем разбит на несколько потоков на следующих уровнях диаграммы.

Для проверки контекстной диаграммы можно составить список событий, который должен состоять из описаний действий внешних сущностей (событий) и соответствующих реакций системы на них. Каждое событие должно соответствовать одному или более потокам данных: входные потоки интерпретируются как воздействия, а выходные потоки – как реакции системы на входные потоки.

Если для сложной системы ограничиться единственной контекстной диаграммой, то она будет содержать слишком большое количество источников и приемников информации, которые трудно расположить на листе бумаги нормального формата, и, кроме того, единственный главный процесс не раскрывает структуры такой системы. Признаками сложности (в смысле контекста) могут быть: наличие большого количества внешних сущностей (десять и более), распределенная природа системы, многофункциональность системы с уже сложившейся или выявленной группировкой функций в отдельные подсистемы.

Для сложных систем строится иерархия контекстных диаграмм. При этом контекстная диаграмма верхнего уровня содержит не единственный главный процесс, а набор подсистем, соединенных потоками данных. Контекстные диаграммы следующего уровня детализируют контекст и структуру подсистем.

Иерархия контекстных диаграмм определяет взаимодействие основных функциональных подсистем между собой, с внешними входными и выходными потоками данных и внешними объектами (источниками и приемниками информации), с которыми взаимодействует система.

Разработка контекстных диаграмм решает проблему строгого определения функциональной структуры системы на самой ранней стадии ее проектирования, что особенно важно для сложных многофункциональных систем, в создании которых участвуют разные организации и коллективы разработчиков.

После построения контекстных диаграмм полученную модель следует проверить на полноту исходных данных об объектах системы и изолированность объектов (отсутствие информационных связей с другими объектами).

Для каждой подсистемы, присутствующей на контекстных диаграммах, выполняется ее детализация при помощи DFD. Это можно сделать путем построения диаграммы для каждого события. Каждое событие представляется в виде процесса с соответствующими входными и выходными потоками, накопителями данных, внешними сущностями и ссылками на другие процессы для описания связей между этим процессом и его окружением. Затем все построенные диаграммы сводятся в одну диаграмму нулевого уровня.

Каждый процесс на DFD, в свою очередь, может быть детализирован при помощи DFD или (если процесс элементарный) спецификации. При детализации должны выполняться следующие правила:

- *правило балансировки* – при детализации подсистемы или процесса детализирующая диаграмма в качестве внешних источников или приемников данных может иметь только те компоненты (подсистемы, процессы, внешние сущности, накопители данных), с которыми имеют информационную связь детализируемые подсистема или процесс на родительской диаграмме;
- *правило нумерации* – при детализации процессов должна поддерживаться их иерархическая нумерация. Например, процессы, детализирующие процесс с номером 12, получают номера 12.1, 12.2, 12.3 и т.д.

Спецификация процесса должна формулировать его основные функции таким образом, чтобы в дальнейшем специалист, выполняющий реализацию проекта, смог выполнить их или разработать соответствующую программу.

Спецификация является конечной вершиной иерархии DFD. Решение о завершении детализации процесса и использовании спецификации принимается аналитиком исходя из следующих критериев:

- наличия у процесса относительно небольшого количества входных и выходных потоков данных (2—3 потока);
- возможности описания преобразования данных процессом в виде последовательного алгоритма;
- выполнения процессом единственной логической функции преобразования входной информации в выходную;
- возможности описания логики процесса при помощи спецификации небольшого объема (не более 20-30 строк).

Спецификации должны удовлетворять следующим требованиям:

- для каждого процесса нижнего уровня должна существовать одна и только одна спецификация;
- спецификация должна определять способ преобразования входных потоков в выходные;
- нет необходимости (по крайней мере, на стадии формирования требований) определять метод реализации этого преобразования;
- спецификация должна стремиться к ограничению избыточности – не следует переопределять то, что уже было определено на диаграмме;
- набор конструкций для построения спецификации должен быть простым и понятным.

Фактически спецификации представляют собой описания алгоритмов задач, выполняемых процессами. 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. Известно большое количество разнообразных методов, позволяющих описать тело процесса. Соответствующие этим методам языки могут варьироваться от структурированного естественного языка или псевдокода до визуальных языков моделирования.

Структурированный естественный язык применяется для читабельного, достаточно строгого описания спецификаций процессов. Он представляет собой разумное сочетание строгости языка программирования и читабельности естественного языка и состоит из подмножества слов, организованных в определенные логические структуры, арифметических выражений и диаграмм.

В состав языка входят следующие основные символы:

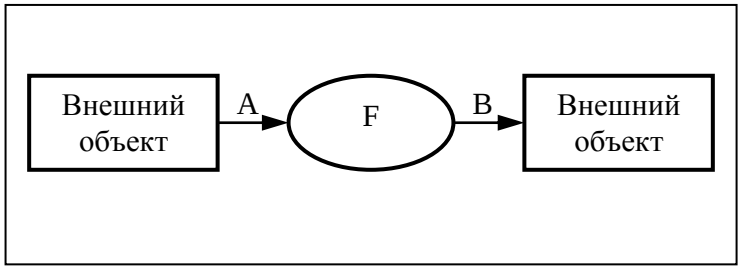
- глаголы, ориентированные на действие и применяемые к объектам;
- термины, определенные на любой стадии проекта ПО (например, задачи, процедуры, символы данных и т.п.);
- предлоги и союзы, используемые в логических отношениях;
- общеупотребительные математические, физические и технические термины;
- арифметические уравнения;
- таблицы, диаграммы, графы и т.п.;
- комментарии.

К управляющим структурам языка относятся последовательная конструкция, конструкция выбора и итерация (цикл). При использовании структурированного естественного языка приняты следующие соглашения:

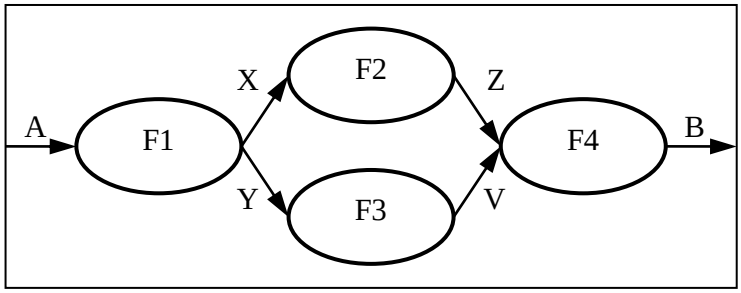
- логика процесса выражается в виде комбинации последовательных конструкций, конструкций выбора и итераций;
- глаголы должны быть активными, недвусмысленными и ориентированными на целевое действие *(заполнить, вычислить, извлечь,* а не *модернизировать, обработать);*
- логика процесса должна быть выражена четко и недвусмысленно.

При построении иерархии DFD переходить к детализации процессов следует только после определения содержания всех потоков и накопителей данных, которое описывается при помощи структур данных. Для каждого потока данных формируется список всех его элементов данных, затем элементы данных объединяются в структуры данных, соответствующие более крупным объектам данных (например, строкам документов или объектам предметной области). Каждый объект должен состоять из элементов, являющихся его атрибутами. Структуры данных могут содержать альтернативы, условные вхождения и итерации. Условное вхождение означает, что данный компонент может отсутствовать в структуре (например, структура «данные о страховании» для объекта «служащий»). Альтернатива означает, что в структуру может входить один из перечисленных элементов. Итерация означает вхождение любого числа элементов в указанном диапазоне (например, элемент «имя ребенка» для объекта «служащий»). Для каждого элемента данных может указываться его тип (непрерывные или дискретные данные). Для непрерывных данных могут указываться единица измерения (кг, см и т.п.), диапазон значений, точность представления и форма физического кодирования. Для дискретных данных может указываться таблица допустимых значений.

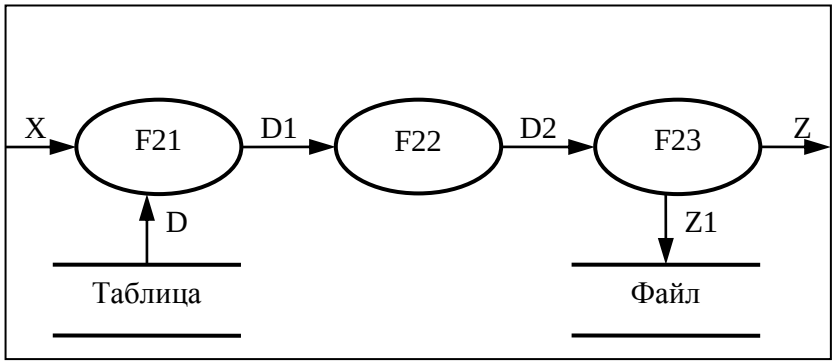
После построения законченной модели системы ее необходимо верифицировать (проверить на полноту и согласованность). В полной модели все ее объекты (подсистемы, процессы, потоки данных) должны быть подробно описаны и детализированы. Выявленные не детализированные объекты следует детализировать, вернувшись на предыдущие шаги разработки. В согласованной модели для всех потоков данных и накопителей данных должно выполняться правило сохранения информации: все поступающие куда-либо данные должны быть считаны, а все считываемые данные должны быть записаны. На рисунке 7 приведен пример построения иерархии диаграмм.



Контекстная диаграмма (уровень-0)



Декомпозиция контекстной диаграммы (уровень-1)



Декомпозиция процесса F2 (уровень-2)

Рисунок 7 – Пример иерархии диаграмм потоков данных

Ход работы

1. Получить у преподавателя индивидуальный вариант задания (информационную систему и ее описание, включающее предметную и функциональную область).
2. Создать проект в CASE-средстве AllFusion Process Modeller (в свойствах проекта указать DFD методологию).
3. Определить контекстный блок информационной системы и отобразить его на контекстной диаграмме.
4. Определить внешние сущности, взаимодействующие с данной системой, и отобразить их на контекстной диаграмме.
5. Определить информационные потоки между внешними сущностями и моделируемой системой.
6. Произвести декомпозицию контекстной диаграммы, представив системы в виде взаимодействующих процессов.
7. Определить на диаграмме A0 информационные потоки между процессами и между процессами и верхним уровнем (диаграмма A-0).
8. Выполнить декомпозицию каждого из процессов на диаграмме A0.
9. На диаграммах второго уровня необходимо определить и построить хранилища данных, с которыми работает моделируемая информационная система.
10. Определить и построить на диаграммах второго уровня информационные потоки между процессами, верхним уровнем и хранилищами данных.
11. При необходимости осуществить декомпозицию процессов второго уровня и построить диаграммы третьего уровня.
12. Для всех элементов диаграмм второго и третьего уровней, не имеющих декомпозиции, (процесс или хранилище данных) составить спецификации.
13. Используя возможности CASE-средства AllFusion Process Modeller проверить состоятельность построенной модели (модель не должна содержать ошибок).
14. Продемонстрировать построенную модель преподавателю.

Контрольные вопросы

1. Для чего предназначена методология моделирования потоков данных?
2. На каком этапе ЖЦ ПО используется методология моделирования потоков данных?
3. Какие элементы могут присутствовать на диаграммах потоков данных?
4. Для чего на диаграммах потоков данных предназначены внешние сущности и как они обозначаются?
5. Для чего на диаграммах потоков данных предназначены процессы и как они обозначаются?
6. Для чего на диаграммах потоков данных предназначены хранилища данных и как они обозначаются?
7. Для чего на диаграммах потоков данных предназначены потоки данных и как они обозначаются?
8. Поясните семантику и нотацию слияния и разделения потоков данных в методологии DFD.
9. В чем сходство и различие методологий DFD и IDEF0?
10. Какие предъявляются рекомендации при построении диаграмм DFD?
11. Как осуществляется построение иерархии диаграмм DFD?
12. В чем заключается предназначение контекстной диаграммы, и какие элементы могут на ней отображаться?
13. Какие правила должны соблюдаться при декомпозиции элементов диаграмм DFD?
14. Что такое спецификация процесса и для чего она предназначена?

15. По каким критериям можно принять решение о прекращении декомпозиции и составлении спецификации процесса?
16. Какие требования предъявляются к спецификации процесса?
17. На каком языке осуществляется задание спецификации процесса, и какие требования к его составу предъявляются?