Структурный подход к проектированию ПС

1.1 Сущность структурного подхода

Сущность структурного подхода к разработке ИС заключается в ее декомпозиции (разбиении) на автоматизируемые функции: система разбивается на функциональные подсистемы, которые в свою очередь делятся на подфункции, подразделяемые на задачи и так далее. Процесс разбиения продолжается вплоть до конкретных процедур. При этом автоматизируемая система сохраняет целостное представление, в котором все составляющие компоненты взаимоувязаны. При разработке системы "снизу-вверх" от отдельных задач ко всей системе целостность теряется, возникают проблемы при информационной стыковке отдельных компонентов.

Все наиболее распространенные методологии структурного подхода базируются на ряде общих принципов. В качестве двух базовых принципов используются следующие:

- принцип "разделяй и властвуй";
- принцип иерархического упорядочивания.

Выделение двух базовых принципов не означает, что остальные принципы являются второстепенными, поскольку игнорирование любого из них может привести к непредсказуемым последствиям (в том числе и к провалу всего проекта). Основными из этих принципов являются следующие:

- принцип абстрагирования;
- принцип формализации;
- принцип непротиворечивости;
- принцип структурирования данных.

В структурном анализе используются в основном две группы средств, иллюстрирующих функции, выполняемые системой и отношения между данными. Каждой группе средств соответствуют определенные виды моделей (диаграмм), наиболее распространенными среди которых являются следующие:

- SADT (Structured Analysis and Design Technique) модели и соответствующие функциональные диаграммы;
- DFD (Data Flow Diagrams) диаграммы потоков данных;
- ERD (Entity-Relationship Diagrams) диаграммы "сущность-связь".

На стадии проектирования ИС модели расширяются, уточняются и дополняются диаграммами, отражающими структуру программного обеспечения: архитектуру ПО, структурные схемы программ и диаграммы экранных форм.

1.2 Метод SADT

Методология SADT разработана Дугласом Россом, на ее основе разработана, в частности, известная методология IDEF0 (Icam DEFinition), которая является основной частью программы ICAM (Интеграция компьютерных и промышленных технологий), проводимой по инициативе ВВС США.

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. Основные элементы этой 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;
- связность диаграмм;
- уникальность меток и наименований;
- синтаксические правила для графики;
- разделение входов и управлений.
- отделение организации от функции, т.е. исключение влияния организационной структуры на функциональную модель.

1.2.1 Состав функциональной модели

Результатом применения методологии SADT является модель, которая состоит из диаграмм, фрагментов текстов и глоссария, имеющих ссылки друг на друга. 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 (рисунок 1.1).

Рисунок Ошибка! Текст указанного стиля в документе отсутствует.**.1 – Функциональный блок и интерфейсные дуги**

На рисунке 4.2, 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.

1.2.2 Иерархия диаграмм

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.

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.

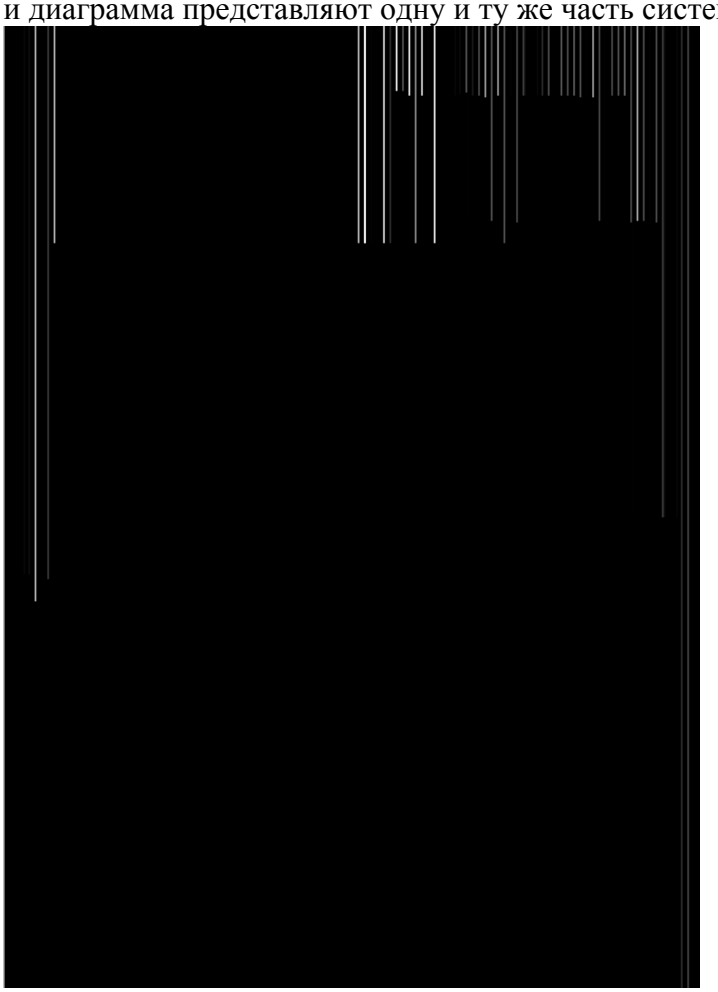


Рисунок Ошибка! Текст указанного стиля в документе отсутствует.**.2 – Структура SADT-модели. Декомпозиция диаграмм**

На рисунках 1.3 - 1.5 представлены различные варианты выполнения функций и соединения дуг с блоками.

Рисунок Ошибка! Текст указанного стиля в документе отсутствует.**.3 – Одновременное выполнение**

Рисунок Ошибка! Текст указанного стиля в документе отсутствует.**.4 – Соответствие должно быть полным и непротиворечивым**

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. (рисунок 1.5).

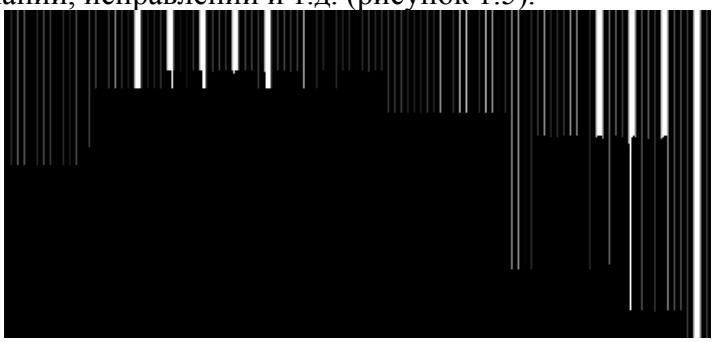


Рисунок Ошибка! Текст указанного стиля в документе отсутствует.**.5 – Пример обратной связи**

Как было отмечено, механизмы (дуги с нижней стороны) показывают средства, с помощью которых осуществляется выполнение функций (рисунок 4.6).

Рисунок Ошибка! Текст указанного стиля в документе отсутствует.**.6 – Пример механизма**

Каждый блок на диаграмме имеет свой номер. Блок любой диаграммы может быть далее 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, A21 является диаграммой, которая детализирует блок 1 на диаграмме A2. Аналогично, A2 детализирует блок 2 на диаграмме A0, которая является самой верхней диаграммой модели. На рисунке 1.7 показано типичное дерево диаграмм.

Рисунок Ошибка! Текст указанного стиля в документе отсутствует.**.7 – Иерархия диаграмм**

1.2.3 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:

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	Коммуникационнная	Функции, использующие одни и те же данные	Данные, на которые воздействует одна и та же

Значимость	Тип связности	Для функций	Для данных
			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

1.3 Диаграммы потоков данных

В основе данной методологии (методологии Gane/Sarson) лежит построение модели анализируемой ИС – проектируемой или реально существующей. В соответствии с методологией модель системы определяется как иерархия диаграмм потоков данных (ДПД или DFD), описывающих асинхронный процесс преобразования информации от ее ввода в систему до выдачи пользователю. Диаграммы верхних уровней иерархии определяют основные процессы или подсистемы ИС 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 такой уровень декомпозиции, на котором процесс становятся элементарными и детализировать их далее невозможно.

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. Таким образом, основными компонентами диаграмм потоков данных являются:

- внешние сущности;
- системы/подсистемы;
- процессы;
- накопители данных;
- потоки данных.

1.3.1 Внешние сущности

Внешняя сущность представляет собой материальный предмет или физическое лицо, представляющее собой источник или приемник информации. В процессе анализа некоторые внешние сущности могут быть перенесены внутрь диаграммы анализируемой ИС, если это необходимо, или, наоборот, часть процессов ИС может быть вынесена за пределы диаграммы и представлена как внешняя сущность.

Внешняя сущность обозначается квадратом (рисунок 1.8), расположенным как бы "над" диаграммой и бросающим на нее тень, для того, чтобы можно было выделить этот символ среди других обозначений:

Рисунок Ошибка! Текст указанного стиля в документе отсутствует.**.8 – Внешняя сущность**

1.3.2 Системы и подсистемы

При построении модели сложной ИС она может быть представлена в самом общем виде на так называемой контекстной диаграмме в виде одной системы как единого целого, либо может быть декомпозирована на ряд подсистем.

Подсистема (или система) на контекстной диаграмме изображается следующим образом (рисунок 1.9).

Рисунок Ошибка! Текст указанного стиля в документе отсутствует.**.9 – Подсистема**

Номер подсистемы служит для ее идентификации. В поле имени вводится наименование подсистемы в виде предложения с подлежащим и соответствующими определениями и дополнениями.

1.3.3 Процессы

Процесс представляет собой преобразование входных потоков данных в выходные в соответствии с определенным алгоритмом. Процесс на диаграмме потоков данных изображается, как показано на рисунке 1.10.

Рисунок Ошибка! Текст указанного стиля в документе отсутствует.**.10 –Процесс**

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, например:

- "Ввести сведения о клиентах";
- "Выдать информацию о текущих расходах";
- "Проверить кредитоспособность клиента".

Информация в поле физической реализации показывает, какое подразделение организации, программа или аппаратное устройство выполняет данный процесс.

1.3.4 Накопители данных

Накопитель данных представляет собой 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 на диаграмме потоков данных изображается, как показано на рисунке 1.11.

Рисунок Ошибка! Текст указанного стиля в документе отсутствует.**.11 – Накопитель данных**

Накопитель данных идентифицируется буквой "D" и произвольным числом. Имя накопителя выбирается из соображения наибольшей информативности для проектировщика.

1.3.5 Потоки данных

Поток данных определяет информацию, передаваемую через некоторое соединение от источника к приемнику. Поток данных на диаграмме изображается линией, оканчивающейся стрелкой, которая показывает направление потока (рисунок 1.12). Каждый поток данных имеет имя, отражающее его содержание.

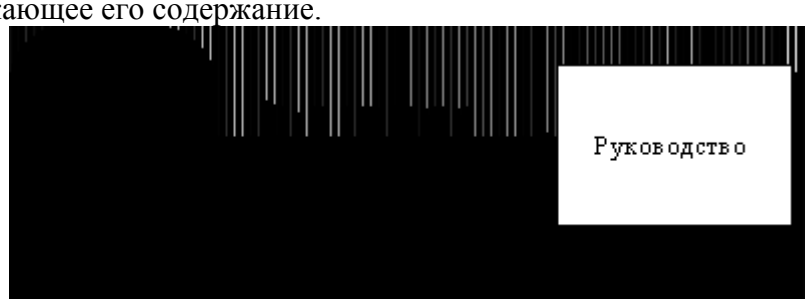


Рисунок Ошибка! Текст указанного стиля в документе отсутствует.**..12 – Поток данных**

1.3.6 Построение иерархии диаграмм потоков данных

Первым шагом при построении иерархии ДПД является построение контекстных диаграмм. Обычно при проектировании относительно простых ИС 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.

Признаками сложности системы могут быть:

- наличие большого количества внешних сущностей;
- распределенная природа системы;
- многофункциональность системы с уже сложившейся или выявленной группировкой функций в отдельные подсистемы.

Для сложных ИС 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.

Разработка контекстных диаграмм решает проблему строгого определения функциональной структуры ИС на самой ранней стадии ее проектирования, что особенно важно для сложных многофункциональных систем, в разработке 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.

Для каждой подсистемы, присутствующей на контекстных диаграммах, выполняется ее детализация при помощи ДПД. Каждый процесс на ДПД, в свою очередь, может быть детализирован при помощи ДПД или миниспецификации. При детализации должны выполняться следующие правила:

- правило балансировки;
- правило нумерации.

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

Миниспецификация является конечной вершиной иерархии ДПД. Решение о завершении детализации процесса и использовании миниспецификации принимается аналитиком исходя из следующих критериев:

- наличия у процесса относительно небольшого количества входных и выходных потоков данных;

- 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 миниспецификации небольшого объема.

При построении иерархии ДПД переходить к детализации процессов следует только после определения содержания всех потоков и накопителей данных, которое описывается при помощи структур данных. Структуры данных конструируются из элементов данных и могут содержать альтернативы, условные вхождения и итерации. Условное вхождение означает, что данный компонент может отсутствовать в структуре. Альтернатива означает, что в структуру может входить один из перечисленных элементов. Итерация означает вхождение любого числа элементов в указанном диапазоне. Для каждого элемента данных может указываться его тип (непрерывные или дискретные данные). Для непрерывных данных может 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. В полной модели все ее объекты должны быть подробно описаны и детализированы. Выявленные недетализированные 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.

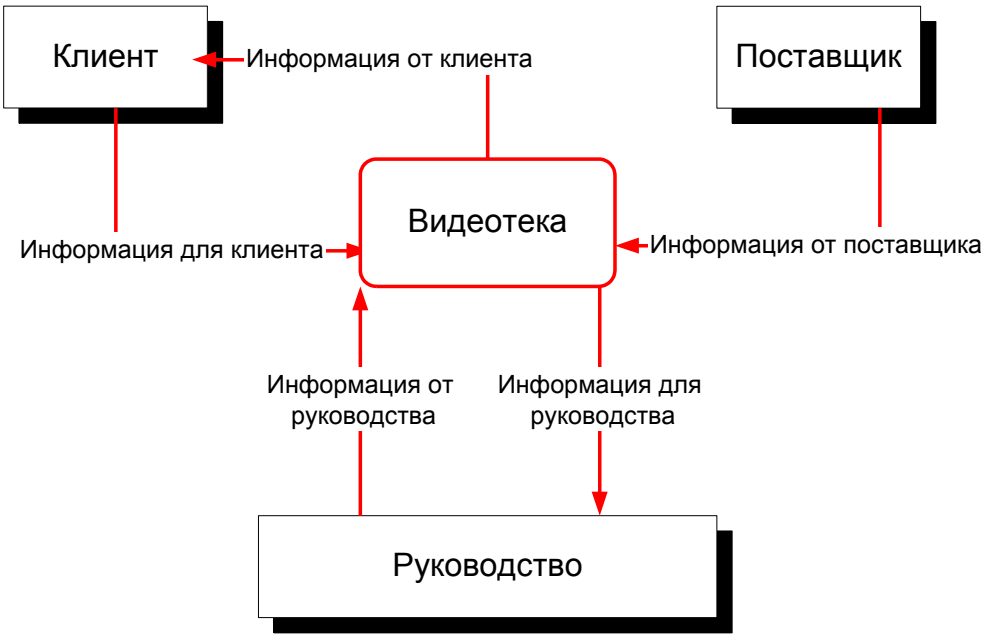
В качестве предметной области используется описание работы видеотеки, которая получает запросы на фильмы от клиентов и диски, возвращаемые клиентами. Запросы рассматриваются администрацией видеотеки с использованием информации о клиентах, фильмах и дисках. При этом проверяется и обновляется список арендованных дисков, а также проверяются записи о членстве в видеотеке. Администрация контролирует также возвраты дисков, используя информацию о фильмах, дисках и список арендованных дисков, который обновляется. Обработка запросов на фильмы и возврат дисков включает следующие действия: если клиент не является членом видеотеки, он не имеет права на аренду. Если требуемый фильм имеется в наличии, администрация информирует клиента об арендной плате. Однако, если клиент просрочил срок возврата имеющихся у него дисков, ему не разрешается брать новые фильмы, пока он не сдаст старые. Когда диск возвращается, администрация рассчитывает арендную плату плюс пеня за несвоевременный возврат.

Видеотека получает новые диски от своих поставщиков. Когда новые диски поступают в видеотеку, необходимая информация о них фиксируется. Информация о членстве в видеотеке содержится отдельно от записей об аренде дисков.

Администрация видеотеки регулярно готовит отчеты за определенный период времени о членах видеотеки, поставщиках дисков, выдаче определенных дисков и дисках, приобретенных видеотекой.

1.4 Использование методологии DFD

Перед построением контекстной DFD необходимо проанализировать внешние события (внешние объекты), оказывающие влияние на функционирование видеотеки. Эти объекты взаимодействуют с ИС путем информационного обмена с ней.



Начальная контекстная диаграмма

Список событий строится в виде матрицы (ELM) и описывает различные действия внешних сущностей и реакцию ИС на них.

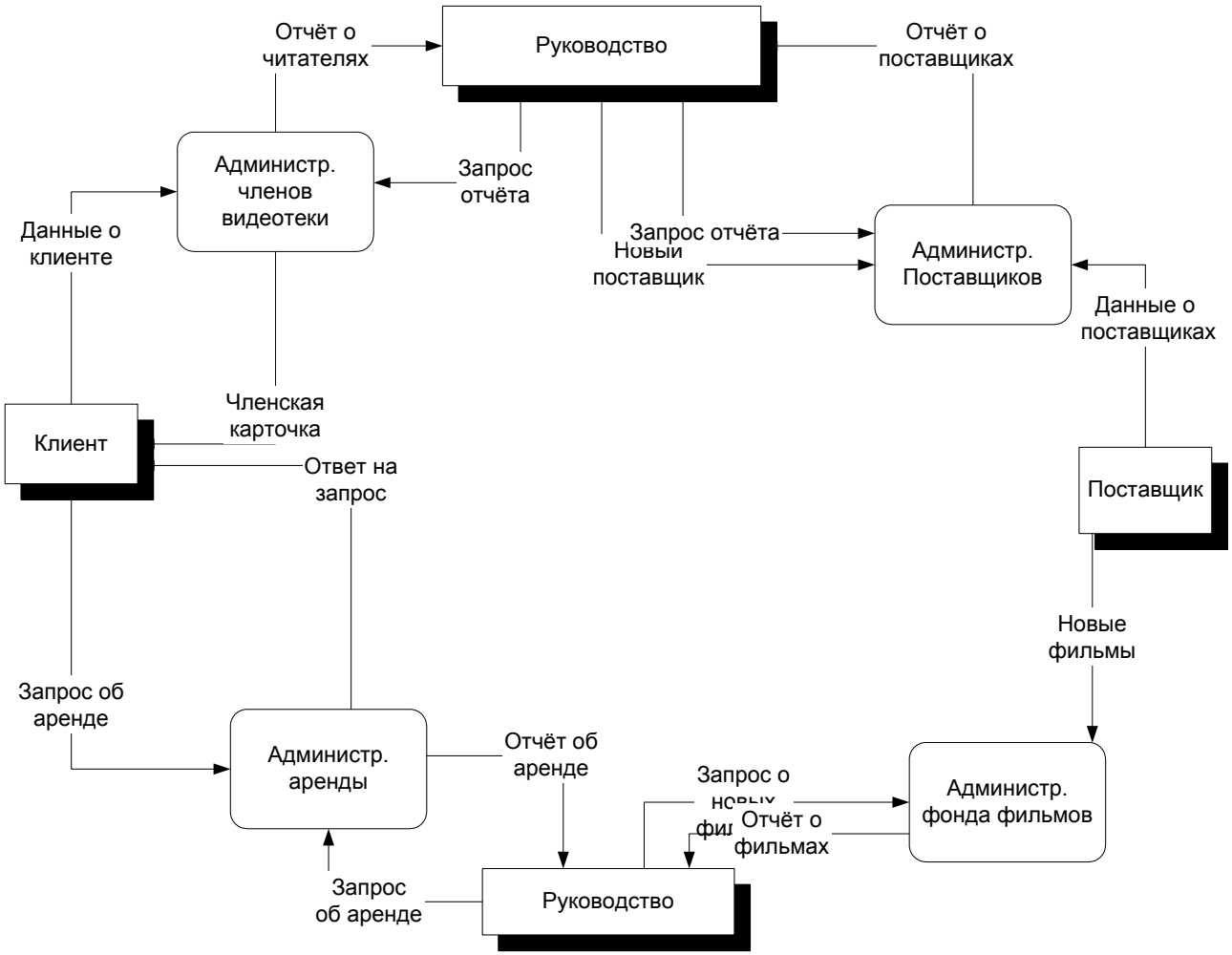
Матрица списка событий имеет следующий вид:

№	Описание	Реакция
1	Клиент желает стать членом видеотеки	Регистрация клиента в качестве члена видеотеки
2	Клиент сообщает об изменении адреса	Регистрация измененного адреса клиента
3	Клиент запрашивает аренду фильма	Рассмотрение запроса
4	Клиент возвращает фильм	Регистрация возврата
5	Руководство предоставляет полномочия новому поставщику	Регистрация поставщика
6	Поставщик сообщает об изменении адреса	Регистрация измененного адреса поставщика
7	Поставщик направляет фильм в видеотеку	Получение нового фильма
8	Руководство запрашивает новый отчет	Формирование требуемого отчета для руководства

Для завершения анализа функционального аспекта поведения системы строится полная контекстная диаграмма, включающая диаграмму нулевого уровня. При этом процесс "видеотека" декомпозируется на 4 процесса, отражающие основные виды административной деятельности видеотеки. Существующие "абстрактные" потоки данных между терминаторами и процессами трансформируются в потоки, представляющие обмен данными на более конкретном уровне. Список событий показывает, какие потоки существуют на этом уровне: каждое событие из списка должно формировать некоторый поток.

Потоки на диаграмме верхнего уровня	Потоки на диаграмме нулевого уровня
Информация от клиента	Данные о клиенте, Запрос об аренде
Информация для клиента	Членская карточка, Ответ на запрос об аренде
Информация от руководства	Запрос отчета о новых членах, Новый поставщик, Запрос отчета о поставщиках, Запрос отчета об аренде, Запрос отчета о фильмах

Потоки на диаграмме верхнего уровня	Потоки на диаграмме нулевого уровня
Информация для руководства	Отчет о новых членах, Отчет о поставщиках, Отчет об аренде, Отчет о фильмах
Информация от поставщика	Данные о поставщике, Новые фильмы



Контекстная диаграмма