

Институт Интеграции Международных  
Образовательных Программ

Кыргызско-Американский Факультет  
информационных технологий и Интернет

# Курсовая работа

По дисциплине: CASE-Технологии

Тема: Анализ группы тестирования ПО в  
ООО "Фининфор"

**Выполнил:** гр. КИС-М-04  
Похилько А.Ф.

**Проверила:** Мукашова Ж.Б.

## Содержание

<b>1. ВВЕДЕНИЕ .....</b>	<b>3</b>
<b>2. ОПИСАНИЕ СТРУКТУРНЫХ МЕТОДОЛОГИЙ И НОТАЦИЙ.....</b>	<b>4</b>
2.1.    ФУНКЦИОНАЛЬНЫЙ АНАЛИЗ IDEF0.....	4
2.2.    ДИАГРАММЫ ПОТОКОВ ДАННЫХ DFD .....	4
2.3.    ER-МОДЕЛЬ БАЗЫ ДАННЫХ.....	4
<b>3. MICROSOFT VISIO КАК CASE-СРЕДСТВО .....</b>	<b>5</b>
3.1.    ОБОСНОВАНИЕ ВЫБОРА .....	5
3.2.    VISIO 2003: ОСНОВНЫЕ ФАКТЫ.....	5
3.2.1.  Наглядное представление и оформление процессов, систем и взаимосвязей .....	5
3.2.2.  Более понятное представление данных .....	5
3.2.3.  Сохранение в виде веб-страницы.....	5
3.2.4.  Создание диаграмм по существующим данным .....	6
3.2.5.  Удобство проектирования и анализа.....	6
<b>4. АНАЛИЗ ГРУППЫ ТЕСТИРОВАНИЯ.....</b>	<b>7</b>
4.1.    Точка зрения анализа.....	7
4.2.    ФУНКЦИОНАЛЬНЫЙ АНАЛИЗ .....	7
4.2.1.  Контекстная диаграмма.....	7
4.2.2.  Диаграмма первого уровня .....	8
4.2.3.  Детализация процесса тестирования ПО.....	8
4.3.    Потоки данных .....	9
4.3.1.  Контекстная диаграмма.....	9
4.3.2.  Детали.....	10
4.4.    СТРУКТУРА ДАННЫХ.....	10
4.4.1.  Структура (таблицы, связи) базы данных после генерирования программных кодов .....	11
4.5.    ИЕРАРХИЯ ДИАГРАММ .....	12
<b>5. ЗАКЛЮЧЕНИЕ .....</b>	<b>13</b>
ПРИЛОЖЕНИЕ I. КОНТЕКСТНАЯ ДИАГРАММА IDEF0 .....	14
ПРИЛОЖЕНИЕ II. ДИАГРАММА ПЕРВОГО УРОВНЯ IDEF0 .....	15
ПРИЛОЖЕНИЕ III. ДЕТАЛИЗАЦИЯ ПРОЦЕССА A4 IDEF0.....	16
ПРИЛОЖЕНИЕ IV. КОНТЕКСТНАЯ ДИАГРАММА DFD .....	17
ПРИЛОЖЕНИЕ V. ДИАГРАММА ПЕРВОГО УРОВНЯ DFD.....	18
ПРИЛОЖЕНИЕ VI. ДИАГРАММА СУЩНОСТЬ-СВЯЗЬ.....	19
ПРИЛОЖЕНИЕ VII. SQL-ЛИСТИНГ.....	20

## 1. Введение

Целью работы является анализ группы тестирования программного обеспечения в ООО «Фининфор» (г. Москва). Для достижения этой цели будут использованы средства и подходы, изученные в рамках дисциплины CASE-Технологии на факультете КАФ-Интернет.

Прежде всего, скажем несколько слов о тестировании. Процесс тестирования ПО – это проверка продукта на соответствие требованиям заказчика и принятым стандартам, а также некоторым критериям качества, предлагаемым современными методологиями разработки программного обеспечения. Как только проект, над которым работает компания, достигает уровня сложного и крупного комплекса, поддерживать качество выпускаемого продукта становится невозможным без создания специальной службы. В этой ситуации образуется группа тестирования – сотрудники, которые занимаются вопросами качества ПО профессионально. Качество выпускаемого продукта оказывает существенное влияние на успех компании, так как продукты сомнительного качества не будут востребованы на рынке и компания проиграет конкурентную борьбу.

В рассматриваемой компании тестировщики до сих пор представляли собой просто группу сотрудников, отношения с ними остальных сотрудников не были определены и зафиксированы, равно как и задачи этой группы. Ожидается, что выполняемая работа окажет поддержку при решении следующих задач:

1. выделение группы тестирования в полноценное подразделение;
2. создание документов, регулирующих работу группы в компании (положение о группе тестирования, должностные инструкции сотрудников);
3. определение отношений группы с остальными подразделениями компании.

Появление хорошо организованной группы тестирования в компании должно улучшить качество выпускаемого продукта и, как следствие, оказать положительное влияние на имидж компании в глазах заказчика и его бизнес.

## 2. Описание структурных методологий и нотаций

### 2.1. Функциональный анализ IDEF0

IDEF0-моделирование – это метод или техника анализа сложных систем, состоящих из множества взаимосвязанных функций или деятельности. Техника моделирования имеет чисто функциональную ориентацию. Функции системы анализируются **независимо от объектов**, которые исполняют эти функции. Чисто функциональная перспектива дает возможность четко отделить источники значений (входы и выходы) от источников исполнения (функциональности).

### 2.2. Диаграммы потоков данных DFD

Диаграммы потоков данных (DFD) являются основным средством моделирования функциональных требований проектируемой системы. С их помощью эти требования разбиваются на функциональные компоненты (процессы) и представляются в виде сети, связанной потоками данных. Главная цель DFD - продемонстрировать, как каждый процесс преобразует свои входные данные в выходные, а также выявить отношения между этими процессами. Основными компонентами диаграмм потоков данных являются:

- внешние сущности;
- процессы;
- потоки данных
- хранилища данных.

DFD является структурной методологией, поддерживающей принцип иерархии и декомпозиции, результатом анализа является иерархия диаграмм, описывающая структуру процессов системы и потоки данных в ней.

В данной работе для диаграмм DFD применяется нотация Йордана-Де Марко, в которой процессы изображаются окружностями, а не прямоугольниками.

### 2.3. ER-модель базы данных

Цель ER-моделирования данных состоит в обеспечении разработчика ИС концептуальной схемой базы данных в форме одной модели или нескольких локальных моделей, которые относительно легко могут быть отображены в любую систему баз данных.

Наиболее распространенным средством моделирования данных являются диаграммы "сущность-связь" (ERD). С их помощью определяются важные для предметной области объекты (сущности), их свойства (атрибуты) и отношения друг с другом (связи). ERD непосредственно используются для проектирования реляционных баз данных.

### **3. Microsoft Visio как CASE-средство**

#### **3.1. Обоснование выбора**

Для построения всех видов диаграмм будет использовано средство MS Visio (версия Professional из пакета Office 2003 и специальная версия Architect для экспорта ER-диаграммы в структуру БД). Выбор средства обоснован его универсальностью, в особенности тем, что нет необходимости для построения нового типа диаграмм изучать новое программное средство. Благодаря функциям Reverse Engineering и генераторам кода в версии Architect приложение представляет собой полноценное CASE-средство.

Visio 2003 помогает пользователям, занимающихся бизнесом и техническим специалистам, наглядно представлять, оформлять и передавать информацию о процессах и системах, повышая эффективность их совместной работы. Microsoft Office Visio 2003 используется для построения схем и диаграмм различного типа, а также наглядного представления бизнес-процессов. Ориентированный на широкий круг пользователей, Visio 2003 поможет оптимизировать работу организации, исключить ненужные операции, повысить гибкость и эффективность деятельности.

#### **3.2. Visio 2003: основные факты**

##### **3.2.1. Наглядное представление и оформление процессов, систем и взаимосвязей**

Использование фигур Microsoft SmartShapes® в сочетании с удобными возможностями поиска, благодаря которым легко найти нужную форму на компьютере или в Интернете, позволяет значительно упростить построение диаграмм в Visio 2003.

Visio 2003 содержит средства построения диаграмм, необходимые широкому кругу пользователей: специалистам в области бизнеса, руководителям проектов, специалистам по сбыту и маркетингу, диспетчерам, специалистам по информационным технологиям, разработчикам программного обеспечения, администраторам баз данных и веб-узлов, руководителям производства и инженерам.

##### **3.2.2. Более понятное представление данных**

Visio расширяет возможности системы Microsoft Office System, облегчая понимание и обмен сложными идеями, системами и процессами путем их графического представления. Фигуры Visio отличаются от простых рисунков тем, что они обладают набором настраиваемых свойств для отображения ключевых данных в контексте диаграммы.

##### **3.2.3. Сохранение в виде веб-страницы**

Позволяет сохранить диаграмму Visio в одном из веб-форматов и обеспечивает доступ к данным изображения, используя пользовательский интерфейс обозревателя (UI). Интерфейс создаваемых в Visio веб-страниц отличается привлекательностью, современностью и доступностью, предоставляя широкие возможности для совместной работы с данными в организации и за ее пределами.

### **3.2.4. Создание диаграмм по существующим данным**

Автоматическое создание диаграмм баз данных из баз данных SQL Server или Access, создание диаграмм программного обеспечения на языке UML из проектов Microsoft Visual Studio® .NET, создание веб-схем существующих веб-узлов, шкал времени из Microsoft Excel или Microsoft Project, календарей из Microsoft Outlook® и организационных диаграмм из Microsoft Excel или Microsoft Exchange Server. Данные диаграмм Visio можно извлекать в формате XML и других форматах, экспортировать в файлы Excel, Microsoft Word, SQL Server и другие типы файлов для интеграции с бизнес-процессами и системами.

### **3.2.5. Удобство проектирования и анализа**

Visio 2003 служит для создания деловых и технических диаграмм для лучшего понимания и организации сложных процессов и систем.

- Быстрое построение диаграмм путем перетаскивания готовых фигур Microsoft SmartShapes®.
- Использование ориентированных на различную профессиональную тематику средств для создания широкого диапазона деловых и технических диаграмм в масштабах организации.
- Создание диаграмм общего типа из текущих данных.
- Доступ к контекстной справке и шаблонам, регулярно обновляемым через Интернет.
- Наглядное представления данных и эффективная совместная работа
- Визуальное представление бизнес-процессов при помощи диаграмм различных типов и уровней сложности.

## 4. Анализ группы тестирования

### 4.1. Точка зрения анализа

Для применения структурных методологий анализа, прежде всего нам необходимо зафиксировать точку зрения моделирования.

*В данной работе анализ проводится с точки зрения организатора работы группы тестирования – **руководителя группы**. Особенностью точки зрения является влияние обстановки первичной организации работы группы.*

Для выбранной точки зрения достаточен уровень детализации, на котором рассматривается факт проверки соответствия ПО спецификациям, без необходимости (на данном этапе) разбивать процесс проверок на элементы непосредственного тестирования.

### 4.2. Функциональный анализ

#### 4.2.1. Контекстная диаграмма

Функциональный анализ в нотации IDEF0 мы начали, как всегда, с построения контекстной диаграммы системы (см. Приложение I), главным результатом которого явилось определение входов, выходов, управляющих воздействий и инструментов **системы в целом**.

Обнаружилось, что *процесс тестирования ПО* происходит на основании должностных инструкций и методологий тестирования (*управляющие воздействия*), в нем участвует группа тестирования и применяются инструменты тестирования (*инструменты*). В качестве *входов* процесса обозначились:

- **спецификации ПО** – описание изменений и дополнений в ПО, служащие эталоном поведения системы, с которым сравнивается работа поступившего в тестирование билда;
- **ПО к тестированию** – билд (сборка) программного обеспечения, проверяемая на соответствие заявленному в спецификации описанию;
- **экземпляры БД** – получаемые от заказчика экземпляры реальных БД, позволяющие создать в тестовой лаборатории среду, близкую к эксплуатационной;
- **сообщения об ошибках** – поступающие от пользователей и сотрудников проектной группы сообщения о новых ошибках, а также сообщения, изменившие свой статус на интересующий группу тестирования (новое, исправлено).

На *выходе*, в качестве результатов работы представлены:

- **отчеты о тестировании ПО** – документы, содержащие результаты тестирования и перечень проведенных тестов, начиная от простого подтверждения успешного тестирования, заканчивая развернутым отчетом;

- **сообщения об ошибках** – сообщения о новых обнаруженных в системе ошибках, подтвержденные сообщения об ошибках, протестированные исправленные ошибки.

#### **4.2.2. Диаграмма первого уровня**

На диаграмме первого уровня рассмотрена детализация главного процесса системы «Протестировать ПО». В качестве составляющих процесса представлены пять функций:

##### **4.2.2.1. Поставить задачу автоматизации тестирования (A1)**

Задание автоматизации тестирования формулирует руководитель группы и передает его специалисту по автоматизированному тестированию для реализации. Задание формируется исходя из спецификации ПО и содержит в себе указание цели автоматизации, перечня автоматизируемых действий и их описание.

##### **4.2.2.2. Автоматизировать тестирование (A2)**

Данная функция исполняет задание, сформированное на этапе A1 и в результате выдает автоматизированный тест, использующийся как инструмент в функции A4. Автоматизация тестирования осуществляется специалистом по автоматизированному тестированию с помощью инструментов тестирования – специализированных программ, позволяющих симулировать работу пользователя с программным обеспечением.

##### **4.2.2.3. Подготовить конфигурацию для тестирования (A3)**

Данный процесс является очень важным, подготовительным шагом к собственно проверке ПО. Полученное ПО к тестированию нуждается в установке на тестовый комплекс, и это требует отдельных знаний и усилий. Также для создания качественной тестовой платформы требуются реалистичные базы данных, получаемых от заказчика, задача подготовки и развертки которых также является отдельным этапом работы специалистов.

##### **4.2.2.4. Проверить работу ПО (A4)**

Центральный процесс в группе тестирования – это проверка ПО. В данном процессе участвуют специалисты по ручному и автоматизированному тестированию, работающие на тестовой платформе с помощью инструментов тестирования. В результате их работы обрабатываются сообщения об ошибках и поведение тестовой платформы сверяется со спецификации ПО. Обнаруживаемые в процессе проверки дефекты появляются на выходе процесса, а весь процесс фиксируется в протоколах тестирования, которые передаются на этап A5.

##### **4.2.2.5. Сформировать отчет о тестировании (A5)**

На данном этапе руководитель группы проводит обзор результатов, полученных сотрудниками в процессе проверки ПО, формирует итоговый отчет о тестировании, который передается за рамки группы, руководителю проекта.

#### **4.2.3. Детализация процесса тестирования ПО**

Основной процесс A4 состоит из 6 подпроцессов (Приложение III), каждый из которых в качестве управляющего воздействия использует должностные инструкции.

#### **4.2.3.1. Исполнить автоматизированные тесты (A41)**

Автоматизированное тестирование осуществляется *специалистом по автоматизированному тестированию* на *тестовой платформе* с использованием заранее разработанных *автоматизированных тестов* и с использованием *инструментов тестирования*. В качестве входных данных используется спецификация ПО, и по ней проверяется поведение тестовой платформы. Результатом работы являются отметки о прохождении автоматизированных тестов и сообщения об обнаруженных ошибках, передаваемые на вход следующих процессов.

#### **4.2.3.2. Провести ручное тестирование (A42)**

Ручное тестирование аналогично автоматизированному проверяет поведение *тестовой платформы* и его соответствие *спецификации*. Выполняет это *специалист по ручному тестированию*, результатом его работы становятся новые обнаруженные ошибки и перечень выполненных действий

#### **4.2.3.3. Сформировать новое сообщение об ошибке (A43)**

На вход данного этапа поступают обнаруженные на этапах A41 и A42 новые ошибки, которые требуется внести в базу данных ошибок. Это отдельный процесс формирования баг-репорта (сообщения об ошибке), при котором необходимо оценить критичность ошибки,

#### **4.2.3.4. Заполнить протокол тестирования (A44)**

Протокол тестирования – это отчетный документ *специалиста*, проводившего проверку ПО, в котором записаны проделанные действия или исполненные автоматизированные тесты. Документ позволяет руководителю группы оценить полноту проверки ПО и принять решение о готовности ПО к поставке заказчику.

#### **4.2.3.5. Проверить исправленную ошибку (A45)**

Помимо обновлений функций ПО группе тестирования необходимо подтверждать факт исправления обнаруженных ранее ошибок. Сообщения об исправленных ошибках поступают в группу и проверяются специалистом по ручному тестированию на текущей тестовой платформе, после чего им проставляется признак подтверждения исправления или сообщение отправляется на дальнейшее исправление с уточнением деталей сохраняющейся ошибки в программе.

#### **4.2.3.6. Проверить новое сообщение об ошибке (A46)**

Проверка новых сообщений об ошибке является важным шагом к их исправлению. В процессе проверки выясняется способ ее воспроизвести и добавляются некоторые другие данные (файлы, записи системного журнала, скриншоты), которые помогут разработчику локализовать и исправить ошибку. После воспроизведения и дополнения сообщение об ошибке переходит в состояние «подтверждено».

### **4.3. Поток данных**

#### **4.3.1. Контекстная диаграмма**

На контекстной диаграмме потоков данных выявлены 6 потоков данных, связанных с процессом тестирования ПО, а также хранилище данных – БД ошибок (багтрек). Основной

информационной сущностью является сообщение об ошибке, из них состоит большинство информационных потоков. Ближайшими внешними сущностями для группы тестирования являются разработчики и клиенты, а также менеджер проекта.

#### **4.3.2. Детали**

С точки зрения потоков данных процесс проверки ПО распадается на 4 элемента, причем хранилище данных сюда не входит, так как относится к работе всей проектной группы, а не только тестирования.

##### **4.3.2.1. Проверить исправление ошибки (1.1)**

На вход данного процесса попадают сообщения об ошибках, статус которых «Исправлено». После проверки исправления ошибка переводится в статус «Подтверждено» и направляется в хранилище для доступа других сотрудников. Также в процессе проверки могут быть обнаружены новые ошибки, которые аналогичным образом помещаются в хранилище со статусом «новый».

##### **4.3.2.2. Проверить новое сообщение об ошибке (1.4)**

Сообщение об ошибке помещается в хранилище сразу в момент ее обнаружения. Однако для исправления необходимо описание гарантированного воспроизведения ошибочной ситуации и это составляет значительную часть работы специалистов группы тестирования. Итогом исследований специалиста является решение – подтвердить ошибку, или отвергнуть ее. Отвергается ошибка в случае невозможности ее воспроизвести или в случае неверного истолкования поведения системы.

##### **4.3.2.3. Протестировать ПО вручную (1.2)**

В рамках ручного тестирования ПО поведение поступившей на тестирование версии сверяется с заявленным в спецификации и все расхождения вызывают появление новых сообщений об ошибках. Независимо от их наличия формируется отчет о тестировании, содержащий перечень выполненных проверок и их результаты.

##### **4.3.2.4. Протестировать ПО автоматизировано (1.3)**

Автоматизированное тестирование преследует те же цели, что и ручное, выдает такие же результаты, но использует методы и приемы автоматизации, позволяющие сократить затраты времени и человеческих ресурсов на проведение однообразных и заранее известных тестовых воздействий.

#### **4.4. Структура данных**

Основной информационной сущностью, которая циркулирует в системе, является сообщение об ошибке. В результате атрибутизации для нее выявлены следующие поля:

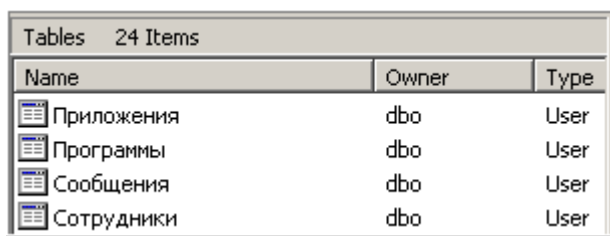
- **ID** – номер для ссылок
- **Продукт** – программный, в котором обнаружена ошибка
- **Краткое описание** – краткое описание ошибки или ее название
- **Описание** – детальное описание ошибки, способов ее воспроизведения

- **Статус** – состояние, в котором находится сообщение (новое, на исправлении, на тестировании, закрытое)
- **Критичность** – уровень приоритета сообщения
- **Ответственный исполнитель** – ответственный за исправление сотрудник
- **Дата создания** – дата внесения сообщения в БД
- **Дата исправления** – дата исправления ошибки

В целях нормализации БД, из основной сущности «Сообщения» вынесены вторичные «Сотрудники» и «Программы и направления», и связаны с главной сущностью отношениями «Один-ко-многим». Итоговую модель данных можно увидеть в приложении VI.

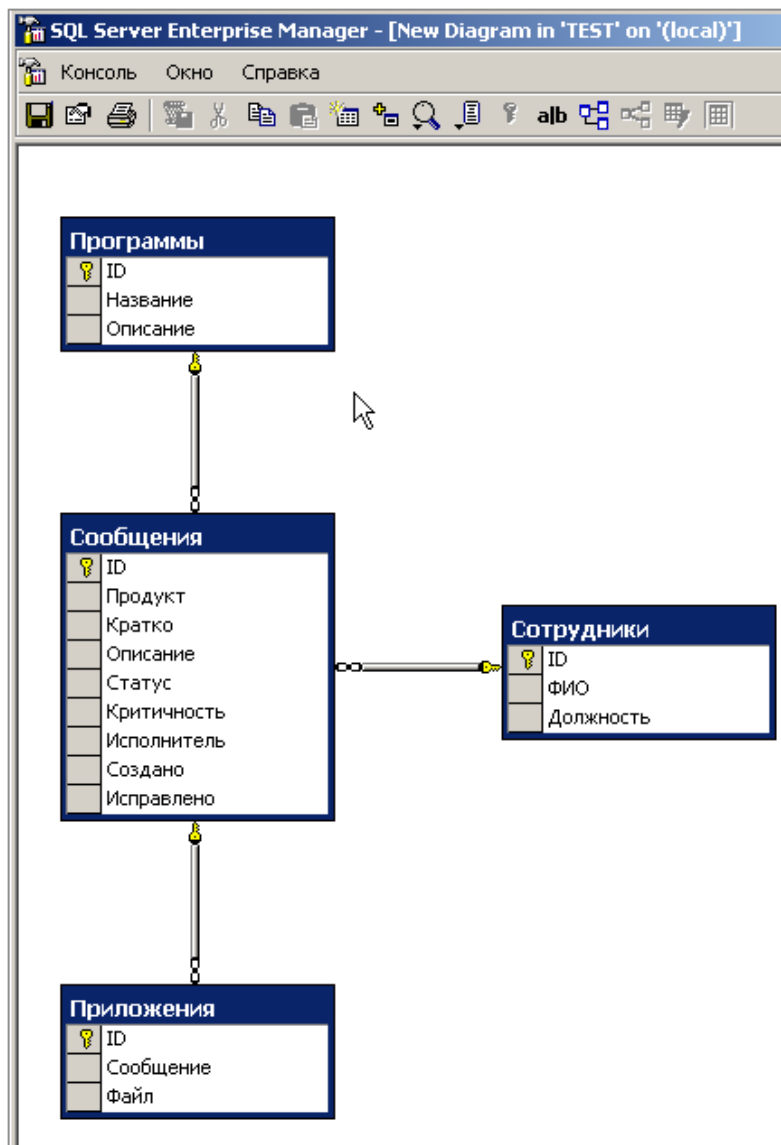
#### 4.4.1. Структура (таблицы, связи) базы данных после генерирования программных кодов

В результате использования встроенных средств MS Visio Architect мы получили SQL-код (приложение VII), генерирующий структуру данных для СУБД MS SQL Server. В результате исполнения кода на сервере мы получили таблицы БД:



Name	Owner	Type
Приложения	dbo	User
Программы	dbo	User
Сообщения	dbo	User
Сотрудники	dbo	User

связанные между собой согласно проектной модели:



#### 4.5. Иерархия диаграмм

В результате применения всех заявленных методик анализа мы получили следующую иерархию диаграмм:

- Контекстная диаграмма IDEF0
  - Диаграмма первого уровня IDEF0
    - Детализация процесса А4 "Протестировать ПО"
- Контекстная диаграмма DFD
  - Диаграмма первого уровня DFD
  - ER-диаграмма

*Все диаграммы можно увидеть в приложениях в конце документа*

## 5. Заключение

Итак, завершен первый этап анализа работы крупы тестирования ПО в компании «Фининфор». Это именно первый этап, так как по его результатам очевидно, что работу эту необходимо продолжать. Группа тестирования является молодым и развивающимся подразделением, никаких нормативных документов в отношении группы нет, и планируется написание *Положения о группе тестирования*, которое как раз будет основываться на результатах проведенной работы.

Важным итогом анализа является определение ролей сотрудников внутри группы, на основе которого планируется разработать должностные инструкции сотрудников. Это поможет каждому из работников четче понимать свою роль в общем процессе и эффективней влиять на качество конечного программного продукта.

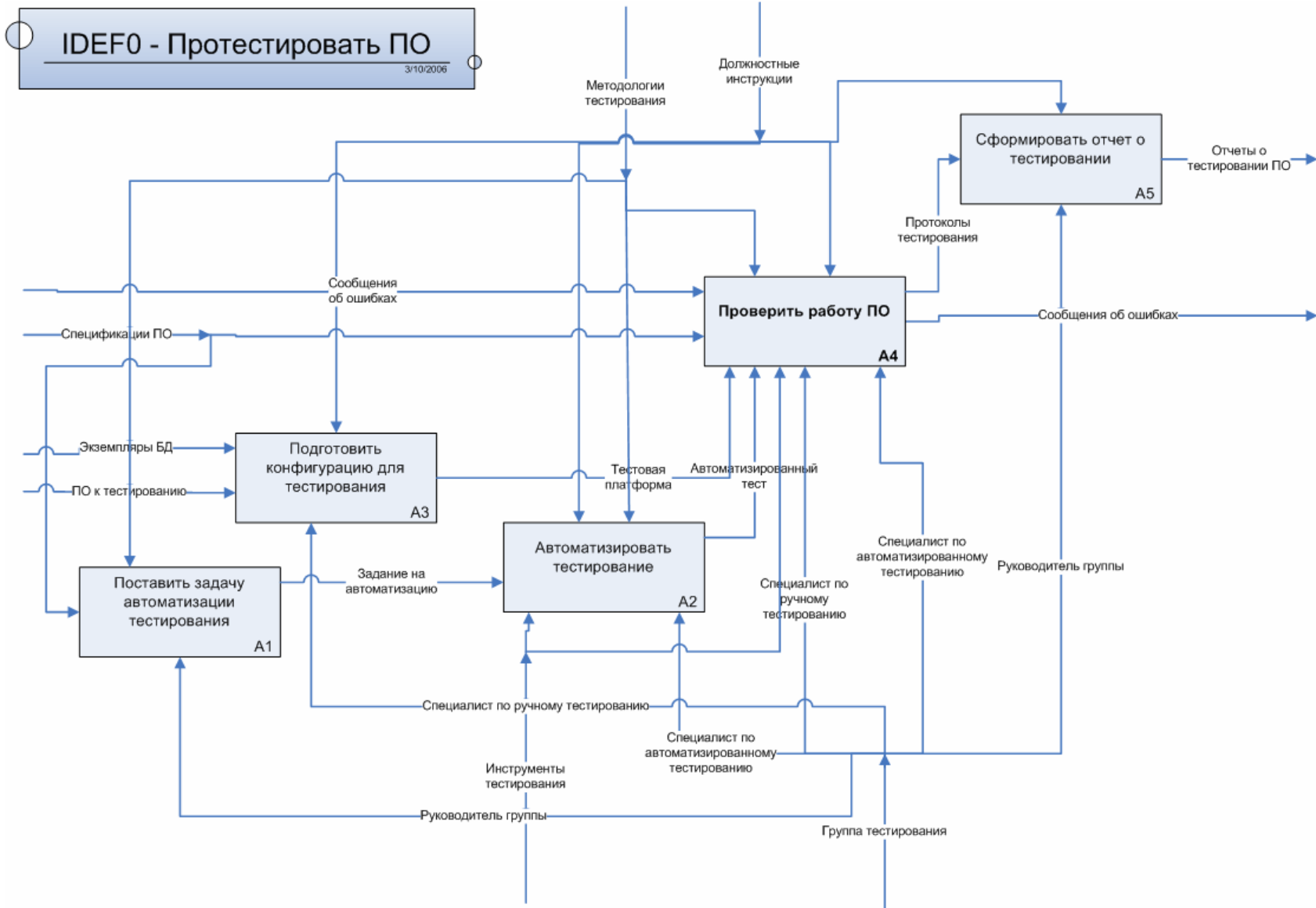
С точки зрения обучения курсу *CASE-технологий* выполнение работы дало возможность осваивать новые знания на живом примере, что является хорошим стимулом, однако этим же была вызвана задержка в завершении работы. В процессе работы явственно обнаружилось, что знания еще недостаточно глубоки, в особенности не хватает опыта структурного проектирования, но обнаружившиеся положительные эффекты от применения методологий структурного анализа дают уверенность, что опыт их применения будет увеличиваться.

*март 2006*

Приложение I. Контекстная диаграмма IDEF0

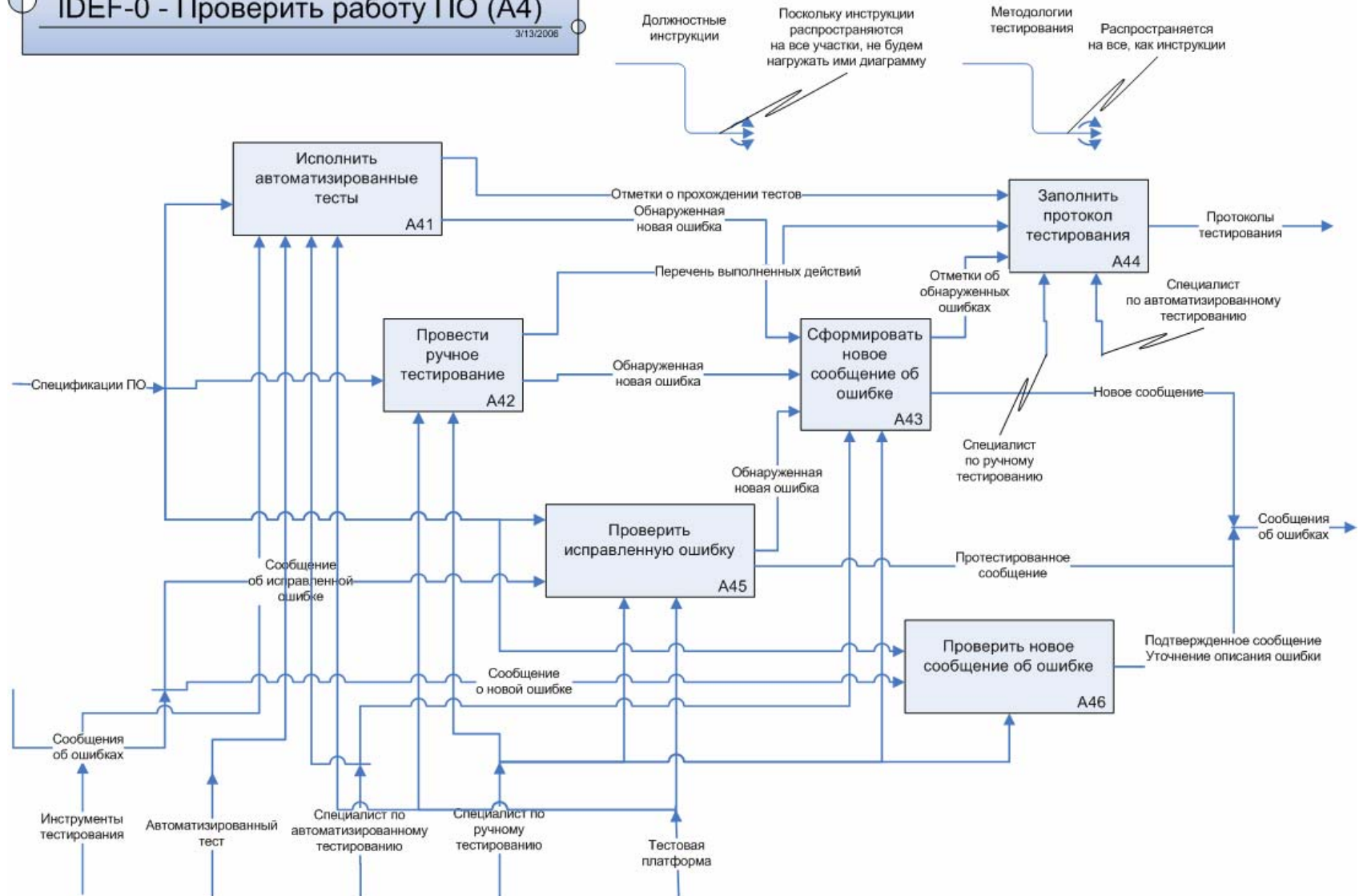


## Приложение II. Диаграмма первого уровня IDEF0

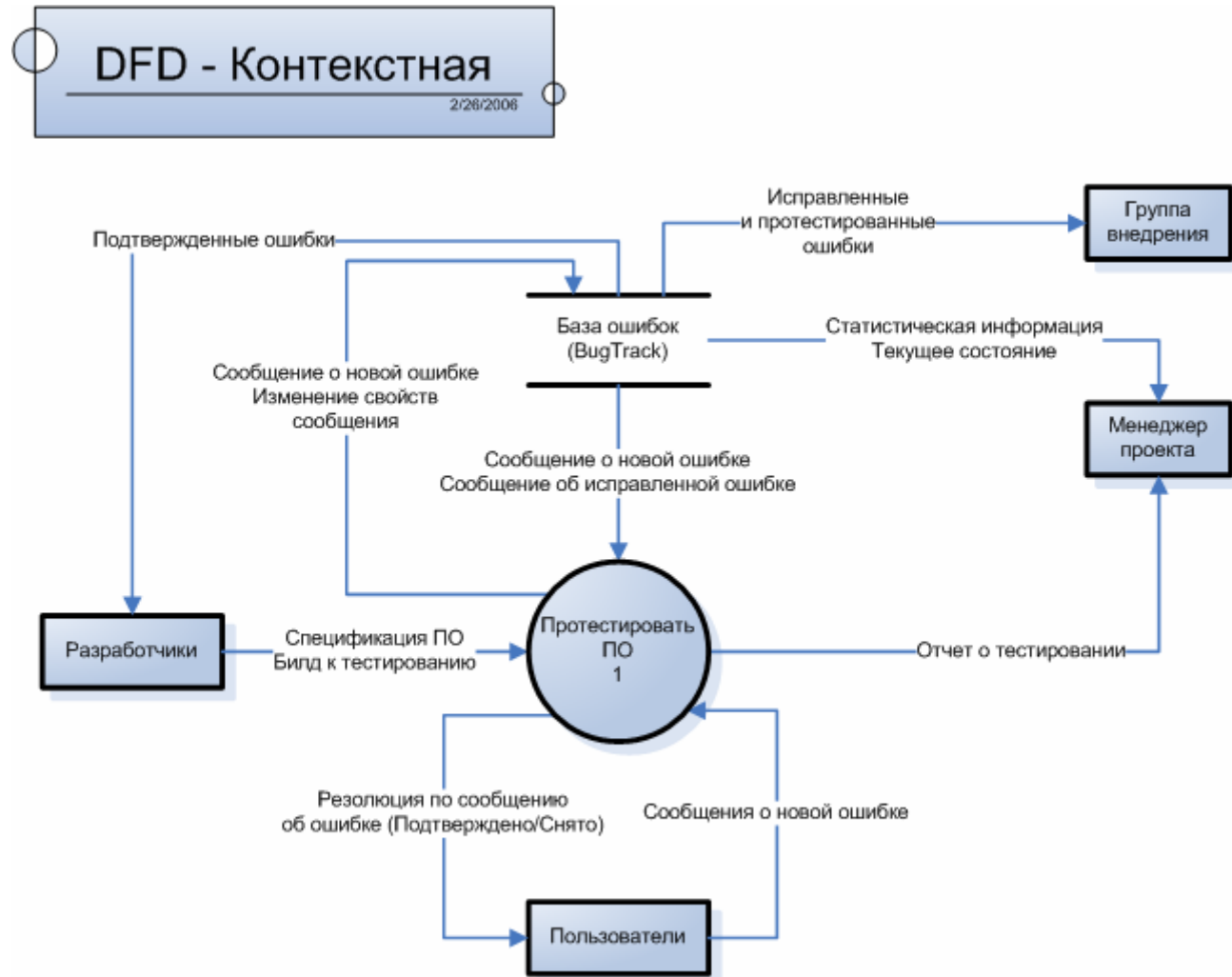


### Приложение III. Детализация процесса A4 IDEF0

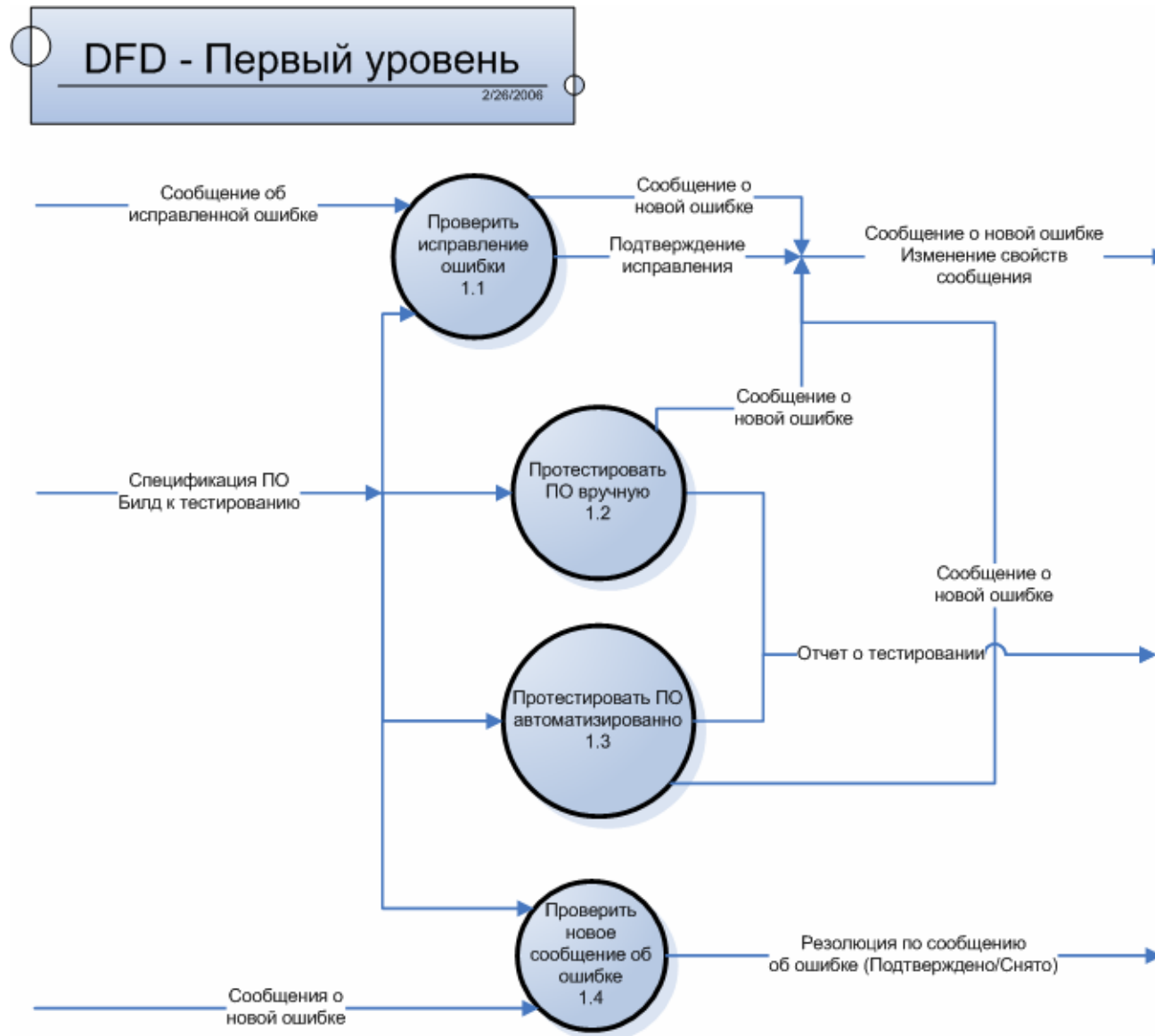
IDEF-0 - Проверить работу ПО (A4)  
3/13/2008



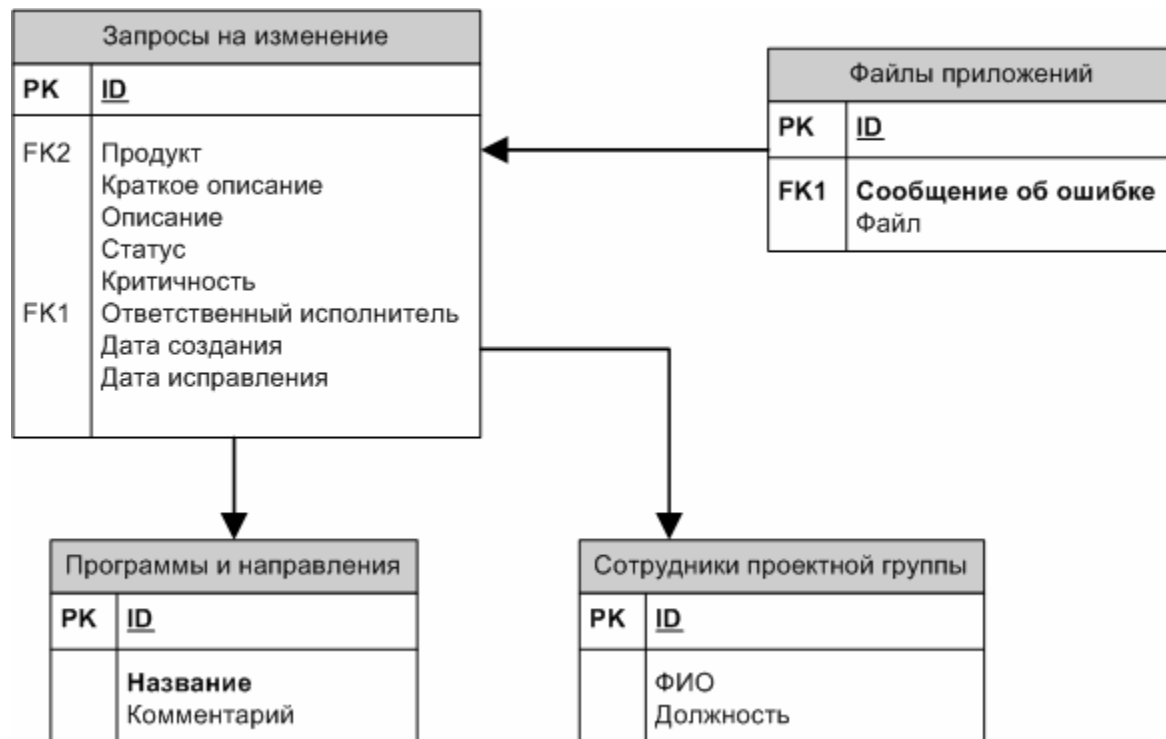
Приложение IV. Контекстная диаграмма DFD



Приложение V. Диаграмма первого уровня DFD



Приложение VI. Диаграмма сущность-связь



## Приложение VII. SQL-листинг

```
if exists (select * from dbo.sysobjects where id = object_id(N'[dbo].[FK_Сообщения_Программы]') and
OBJECTPROPERTY(id, N'IsForeignKey') = 1)
ALTER TABLE [dbo].[Сообщения] DROP CONSTRAINT FK_Сообщения_Программы
GO
if exists (select * from dbo.sysobjects where id = object_id(N'[dbo].[FK_Приложения_Сообщения]') and
OBJECTPROPERTY(id, N'IsForeignKey') = 1)
ALTER TABLE [dbo].[Приложения] DROP CONSTRAINT FK_Приложения_Сообщения
GO
if exists (select * from dbo.sysobjects where id = object_id(N'[dbo].[FK_Сообщения_Сотрудники]') and
OBJECTPROPERTY(id, N'IsForeignKey') = 1)
ALTER TABLE [dbo].[Сообщения] DROP CONSTRAINT FK_Сообщения_Сотрудники
GO
if exists (select * from dbo.sysobjects where id = object_id(N'[dbo].[Приложения]') and
OBJECTPROPERTY(id, N'IsUserTable') = 1)
drop table [dbo].[Приложения]
GO
if exists (select * from dbo.sysobjects where id = object_id(N'[dbo].[Программы]') and
OBJECTPROPERTY(id, N'IsUserTable') = 1)
drop table [dbo].[Программы]
GO
if exists (select * from dbo.sysobjects where id = object_id(N'[dbo].[Сообщения]') and
OBJECTPROPERTY(id, N'IsUserTable') = 1)
drop table [dbo].[Сообщения]
GO
if exists (select * from dbo.sysobjects where id = object_id(N'[dbo].[Сотрудники]') and
OBJECTPROPERTY(id, N'IsUserTable') = 1)
drop table [dbo].[Сотрудники]
GO
CREATE TABLE [dbo].[Приложения] (
    [ID] [int] IDENTITY (1, 1) NOT NULL ,
    [Сообщение] [int] NOT NULL ,
    [Файл] [varbinary] (8000) NOT NULL
) ON [PRIMARY]
GO
CREATE TABLE [dbo].[Программы] (
```

```

        [ID] [int] NOT NULL ,
        [Название] [varchar] (50) COLLATE Cyrillic_General_CS_AS NOT NULL ,
        [Описание] [varchar] (50) COLLATE Cyrillic_General_CS_AS NULL
    ) ON [PRIMARY]
GO
CREATE TABLE [dbo].[Сообщения] (
    [ID] [int] IDENTITY (1, 1) NOT NULL ,
    [Продукт] [int] NOT NULL ,
    [Кратко] [varchar] (50) COLLATE Cyrillic_General_CS_AS NOT NULL ,
    [Описание] [text] COLLATE Cyrillic_General_CS_AS NULL ,
    [Статус] [varchar] (50) COLLATE Cyrillic_General_CS_AS NOT NULL ,
    [Критичность] [int] NULL ,
    [Исполнитель] [int] NULL ,
    [Создано] [datetime] NOT NULL ,
    [Исправлено] [datetime] NULL
) ON [PRIMARY] TEXTIMAGE_ON [PRIMARY]
GO
CREATE TABLE [dbo].[Сотрудники] (
    [ID] [int] IDENTITY (1, 1) NOT NULL ,
    [ФИО] [varchar] (50) COLLATE Cyrillic_General_CS_AS NOT NULL ,
    [Должность] [varchar] (50) COLLATE Cyrillic_General_CS_AS NOT NULL
) ON [PRIMARY]
GO
ALTER TABLE [dbo].[Приложения] WITH NOCHECK ADD
    CONSTRAINT [PK_Приложения] PRIMARY KEY CLUSTERED
    (
        [ID]
    ) ON [PRIMARY]
GO
ALTER TABLE [dbo].[Программы] WITH NOCHECK ADD
    CONSTRAINT [PK_Программы] PRIMARY KEY CLUSTERED
    (
        [ID]
    ) ON [PRIMARY]
GO
ALTER TABLE [dbo].[Сообщения] WITH NOCHECK ADD
    CONSTRAINT [PK_Сообщения] PRIMARY KEY CLUSTERED

```

```

    (
        [ID]
    ) ON [PRIMARY]
GO
ALTER TABLE [dbo].[Сотрудники] WITH NOCHECK ADD
    CONSTRAINT [PK_Сотрудники] PRIMARY KEY CLUSTERED
    (
        [ID]
    ) ON [PRIMARY]
GO
ALTER TABLE [dbo].[Приложения] ADD
    CONSTRAINT [FK_Приложения_Сообщения] FOREIGN KEY
    (
        [Сообщение]
    ) REFERENCES [dbo].[Сообщения] (
        [ID]
    )
GO
ALTER TABLE [dbo].[Сообщения] ADD
    CONSTRAINT [FK_Сообщения_Программы] FOREIGN KEY
    (
        [Продукт]
    ) REFERENCES [dbo].[Программы] (
        [ID]
    ),
    CONSTRAINT [FK_Сообщения_Сотрудники] FOREIGN KEY
    (
        [Исполнитель]
    ) REFERENCES [dbo].[Сотрудники] (
        [ID]
    )
GO

```