МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования
«Воронежский государственный технический университет»

Кафедра систем автоматизированного проектирования и информационных систем

МЕТОДИЧЕСКИЕ УКАЗАНИЯ

к лабораторным работам по дисциплине "Эффективное внедрение информационных систем" для студентов направления 09.03.02 «Информационные системы и технологии» очной формы обучения

Воронеж 2024

Составитель: канд. техн. наук Э.И. Воробьев
УДК 681.3.

Методические указания по выполнению лабораторных работ по курсу " Эффективность внедрения ИС" 09.03.02 "Информационные системы и технологии" дневной формы обучения / Воронеж. гос. техн. ун-т.; Сост. Э. И. Воробьев. Воронеж, 2024.

Настоящие методические указания посвящены рассмотрению основных подходов к разработке приложений для мобильных устройств при выполнении лабораторных работ, порядку их выполнения.

Предназначено для студентов 3-4 курса

Рецензент
Ответственный за выпуск И.О. зав. кафедрой
САПРИС П.Ю. Гусев

Рекомендовано методическим семинаром кафедры САПРИС и методической комиссией ФИТКБ Воронежского государственного технического университета в качестве методических материалов

Лабораторная работа № 1

Разработка сценария внедрения программного продукта для рабочего места

Цели: научиться определять цели и задачи внедрения программного продукта, разбиению команды разработчиков на рабочие группы.

Теоретические сведения

Внедрение программного обеспечения — процесс настройки программного обеспечения под определенные условия использования, а также обучения пользователей работе с программным продуктом.

Цели внедрения программного продукта:

1. Обеспечение стабильной повторяемости результатов с заданным качеством.
2. Снижение квалификационных требований к участникам IT-проектов.
3. Сокращение сроков выполнения IT-проектов.

После определения целей, следует сконцентрироваться на действиях, на каждом из этапов работы, преимущественно на процессе исполнения

Коллективная разработка. Одним из основных вопросов коллективной разработки является разделение труда.

Модели разбиения коллектива на рабочие группы:

- иерархическая модель;
- матричная модель (равноправные соисполнители);
- бригада главного программиста;
- программирование в парах;
- ядерная модель;
- общинная модель разработки.

Методические указания

1. Сформулируйте цели и задачи внедрения вашего программного продукта.
2. Сформулируйте этапы внедрения программного продукта.
3. Организуйте разбиение вашего коллектива (всей группы целиком) в соответствии с классификацией разбиения коллектива на рабочие группы.
4. По группам обсудите и распределите обязанности и задания конкретно для каждого участника.
5. Обсудите, разработайте и оформите в электронном виде техническое задание разработки вашего программного продукта.
6. В ранее созданных вами группах распределите функции управления.
7. Определите ответственных и исполнителей управления.
8. Продумайте и опишите все процессы управления.

Контрольные вопросы

1. Назовите возможные цели и задачи внедрения программного обеспечения.
2. Назовите основные рабочие группы в команде разработчиков.
3. Дайте функциональную характеристику каждой группе.
4. Назовите возможные цели и задачи внедрения ПО.
5. Назовите основные рабочие группы в команде разработчиков.
6. Дайте функциональную характеристику каждой группе.
7. Что такое методологии внедрения?
8. Назовите известные вам методологии управления
9. Что такое управление сроками проекта? Для чего оно нужно?
10. Из каких процессов оно состоит?
11. В чем заключается управление стоимостью?
12. Из каких процессов оно состоит?
13. В чем заключается управление рисками?

Лабораторная работа №2.

Построение функциональной модели бизнес-процессов предприятия

Порядок выполнения работы

1 Построить функциональную модель бизнес-процессов предприятия.
2 Оформить работу.
3 Осуществить защиту работы.

Ход выполнения работы

1 Запустите BPwin(кнопка Пуск/BPwin).
2 Если появляется диалог ModelMart Connection Manager, нажмите на кнопку Cancel.
3 Щелкните по кнопке . Появляется диалоговое окноI would like to. Вне-сите в текстовое поле Name имя модели«Деятельность компании» и выберите Type– Business Process (IDEF0). Нажмите кнопку ОК.
4 Откроется диалоговое окно Properties for New Models (Свойства новой модели). Введите в текстовое поле Author (Автор) имя автора модели, в текстовое поле Author initials– его инициалы. Нажмите последовательно кнопки Apply и ОК.
5 Автоматически создается незаполненная контекстная диаграмма.
6 Обратите внимание на кнопку на панели инструментов. Эта кнопка включает и выключает инструмент просмотра и навигации Model Explorer (Браузер модели). Model Explorer имеет три вкладки– Activities ( ), Diagrams ( ) и Objects ( ). Во вкладке Activities щелчок правой кнопкой по объекту в браузере модели позволяет выбрать опции редактирова-ния его свойств. Откройте окно свойств модели и ознакомьтесь с ними.
7 Если непонятно, как выполнить то или иное действие, можно вызвать

контекстную помощь(клавиша F1) или воспользоваться меню Help.

8 Перейдите в меню Model/Model Properties. Во вкладке General диалого-вого окна Model Properties в текстовое поле Model name следует внести имя модели«Деятельность компании», в текстовое поле Project – имя проекта «Модель деятельности компании» и, наконец, в текстовое поле Time Frame (Временной охват) – AS-IS (Как есть).

9 Во вкладке Purpose диалогового окна Model Properties в текстовое поле Purpose(цель) внесите данные о цели разработки модели«Моделировать текущие(AS-IS) бизнес-процессы компании», в текстовое поле Viewpoint(точка зрения) – «Директор».

10 Во вкладке Definition диалогового окна Model Properties в текстовое поле Definition(Определение) внесите«Это учебная модель, описывающая деятельность компании» и в текстовое поле Scope(охват) – «Общее управление бизнесом компании: исследование рынка, закупка компонентов, сборка, тести-рование и продажа продуктов».

11 Перейдите на контекстную диаграмму и правой кнопкой мыши щелк-ните по прямоугольнику, представляющему в нотации IDEF0 условное графи-ческое обозначение работы. В контекстном меню выберите опцию Name. Во вкладке Name внесите имя«Деятельность компании».

12 Во вкладке Definition диалогового окна Activity Properties в текстовое поле Definition(Определение) внесите«Текущие бизнес-процессы компании». Текстовое поле Note (Примечания) оставьте незаполненным.

13 Создайте ICOM-стрелки на контекстной диаграмме(таблица1). Для это-го выберите инструментArrow на панели инструментов, щелкните мышью на границе источника стрелки, отпустите мышь, щелкните мышью на подсвеченной границе приемника стрелки. Свойства стрелок(Arrow Name, Arrow Definition) задайте в окне Arrow Properties, которое вызывается в контекстном меню.

Таблица1 – Стрелки контекстной диаграммы

Название стрелки (Arrow Name)	Определение стрелки (Arrow Definition)	Тип стрелки (Arrow Type)
Звонки клиентов	Запросы информации, заказы, техподдержка и т. д.	Input
Правила и процедуры	Правила продаж, инструкции по сборке, процедуры тестирования, критерии производительности и т. д.	Control
Проданные продукты	Настольные и портативные компьютеры	Output
Бухгалтерская система	Оформление счетов, оплата счетов, работа с заказами	Mechanism

14 С помощью кнопки внесите текст в поле диаграммы– точку зрения и цель.

15 Создайте отчет по модели. В меню Tools/Reports/Model Report задай-те опции генерирования отчета(установите все галочки) и нажмите кноп-ку Preview (Предварительный просмотр).

Контрольные вопросы

1 Какие свойства имеет модель?

2 Какие вкладки имеет инструментModel Explorer (Браузер модели) и что каждая из них отображает?

3 Назовите типы стрелок в диаграммеIDEF0.

4 Какие виды текста можно внести в поле диаграммы?

5 Назовите опции генерирования, которые включает в себя отчетModel Report.

Лабораторная работа №3.

Язык SQL. Реинжиниринг бизнес-процессов предприятия при внедрении интегрированной информационной системы

Порядок выполнения работы

1 По указанию преподавателя из предложенного перечня процессов пред-приятия выбрать один.

2 Разбить процесс на основные операции, описать их, установить взаимо-связи, исполнителей.

3 Выделить основные элементы процесса(входы, выходы, ресурсы и т. д.) и описать их.

9

4 Выделить основных участников процесса и описать их полномочия по участию в процессе.

5 Провести идентификацию процесса в различных видах классификаций и обосновать свои выводы.

6 Построить модельSADT выбранного процесса и приложить к рисунку поясняющую спецификацию в виде таблицы.

7 Оформить работу.

8 Осуществить защиту работы.

Контрольные вопросы

1 Дайте определение процесса и назовите основные группы про-цессов предприятия.

2 Назовите основные элементы процесса и дайте их определения.

3 Назовите классификационные типы процессов предприятия и определи-те их основные отличительные признаки.

4 Назовите основные принципы выделения процессов на предприятии.

5 Назовите основные правила описания процесса.

Лабораторная работа №4.

Реализация бизнес-логики процессов интегрированной информационной системы предприятия

Порядок выполнения работы

1 Выбрать способ организации бизнес-логики информационной системы.

Существует три основных способа организации бизнес-логики программ-ной системы: функциональный, объектный и смешанный. Каждый из способов

имеет свои преимущества и недостатки, и задача разработчика– выбрать способ, оптимальный для данного проекта.

Сценарий транзакции(функциональный подход).

Это простейший подход к описанию бизнес-логики. За термином«сценарий транзакции» кроется функция системы, реализующая определенную функцию прикладной логики. Например, расчет стоимости заказа, формируемого в системе.

При таком подходе программная система представляет собой набор функций, в котором каждая функция соответствует операции, которую приложение

выполняет для пользователя. Это старая добрая процедурная модель, хорошо

известная разработчикам. Проектируется иерархия функций, возможно повторное использование функций нижних уровней. Существуют давно и хорошо отработанные методологии, а также стандарты функционального моделирования, с помощью которых можно проектировать функциональную структуру

программной системы в сложных случаях.

Несомненное достоинство подхода– простота реализации в программном

коде. Однако у функционального подхода есть и определенные недостатки.

С возрастанием сложности прикладной логики становится чрезвычайно сложно формировать структуру, не содержащую дублированных функций. А попытка переделать имеющуюся сложную структуру при изменении прикладной логики быстро приводит к тому, что функциональная структура выходит из-под контроля разработчиков. Все это в конечном счете осложняет развитие и поддержку системы.

Модель предметной области(объектный подход).

Этот подход наиболее сложен в реализации, но и наиболее продуктивен.

Объектная модель предметной области разрабатывается, по крайней мере, для

основных понятий. Например, для интернет-магазина могут быть созданы объекты «клиент», «товар», «корзина», «заказ» и т. д. Вместо того чтобы помещать

бизнес-логику расчета стоимости в одну или несколько процедур, для каждого объекта реализуется функциональность исходя из зоны ответственности этого объекта. Например, объект «заказ» содержит алгоритм вычисления стоимости заказа на основе цен входящих в него товаров, при этом алгоритмы определения цен на товары реализуются отдельно, в объектах «товар». Если для разных

товаров применяются разные алгоритмы вычисления цены, это достаточно легко реализовать в модели за счет создания разных типов товаров.

Достоинством модели предметной области традиционно признается гибкость. Объектный подход предоставляет в распоряжение разработчика множество способов моделирования отношений и распределения ответственности между объектами, что позволяет настраивать бизнес-логику в очень широких пределах. Кроме того, важной особенностью объектного подхода является то, что объекты могут близко соответствовать понятиям прикладной области, что в принципе позволяет разработчику и заказчику«говорить на одном языке».

Помимо этого, подавляющее большинство типовых решений– шаблонов

проектирования– создано именно для применения в объектной модели. Шаблоны позволяют разрабатывать объектные структуры с заданными свойствами.

В частности, создатели шаблонов уделяют значительное внимание последствиям применения шаблонов и возможностям внесения изменений в объектные структуры, предлагаемые шаблонами.

Однако, чтобы применять объектный подход необходимо выработать определенный, своеобразный стиль мышления. Новичкам приходится затрачивать много времени даже на то, чтобы разобраться в имеющейся объектной структуре (например, в шаблоне), не говоря уже о самостоятельном проектировании объектной структуры. Не всем удается преодолеть барьер освоения этой

технологии; многие разработчики, даже работая на объектно-ориентированном языке и применяя объектные библиотеки, фактически используют процедурный стиль программирования.

Другая особенность модели предметной области– необходимость создания сложных программных механизмов взаимодействия объектных структур с реляционными базами данных. Разработка такого механизма обычно обходится дорого(хотя на этот счет тоже имеются типовые решения).

Модуль таблицы(смешанный подход).

Перечисленные выше методы организации бизнес-логики представляют собой два «крайних» случая. Третий случай– смешанный, гибридный подход, сочетающий в себе определенные достоинства(и недостатки) функционального и объектного подходов.

Типовое решение «модуль таблицы» предусматривает, как и в модели предметной области, отдельные объекты для товаров, заказов и т. д. Однако в отличие от «настоящей» модели предметной области в модуле таблицы для работы со всеми (к примеру) заказами, содержащимися в базе данных, применяется только один объект. Именно этот единственный объект содержит логику обработки заказов. А чтобы работать с отдельным заказом, следует указывать его уникальный идентификатор.

Недостатком данного подхода является сложность разработки и сложность восприятия кода. С точки зрения прозрачности, читабельности кода модуль таблицы заметно проигрывает «настоящей» модели предметной области (конечно, предполагается, что разработчик в каждом случае использует максимум возможностей сделать код простым и понятным). При несложной логике предметной области модуль таблицы проигрывает в простоте реализации и функциональному подходу.

Основные преимущества модуля таблицы– простота взаимодействия с базой данных и гибкость структурирования бизнес-логики по сравнению с функциональным подходом. Возможно также применение шаблонов проектирования. Однако по сравнению с моделью предметной области гибкость и возможности применения шаблонов в модуле таблицы существенно ограничены.

Целесообразность применения модуля таблицы также зависит от уровня поддержки структуры множества записей в конкретной среде разработки (включая удобство отображения множества записей на базу данных и на элементы графического интерфейса). Развитая поддержка набора данных (DataSet) в средеMicrosoft.Net во многих случаях оправдывает применение этой платформы разработки.

Особенности подходов.

Перечислим основные особенности рассмотренных типовых решений, позволяющие сравнивать эти решения между собой и выбирать подход, соответствующий задаче.

Сценарий транзакции(функциональный подход): предельно прост, плохо работает при сложной логике, затрудняет развитие системы.

Модель предметной области(объектный подход): наиболее сложный и затратный, позволяет реализовать сложную бизнес-логику, дает возможность «цивилизованно» развивать систему; для простых систем подход целесообразен, только если уже «обкатан» разработчиком.

Модуль таблицы (смешанный подход): обеспечивает определенное сочетание простоты и гибкости, однако гибкость ограничена, а эффективность подхода зависит от среды разработки.

На основе рассмотренных особенностей можно сформулировать несколько критериев и предложить упрощенную схему выбора подхода на основе качественной оценки критериев.

Качественные критерии и схема выбора.

Особенности типовых решений показывают, что решение по выбору подхода можно принять на основе качественной оценки следующих критериев:

- опыт команды разработчиков;
- стабильность бизнес-логики(ожидаемая стабильность функциональных требований);
- сложность бизнес-логики(функциональная сложность задач, решаемых программной системой);
- среда разработки(удобство работы с множеством записей).

Схема принятия решения на основе этих критериев приведена на рисунке1.

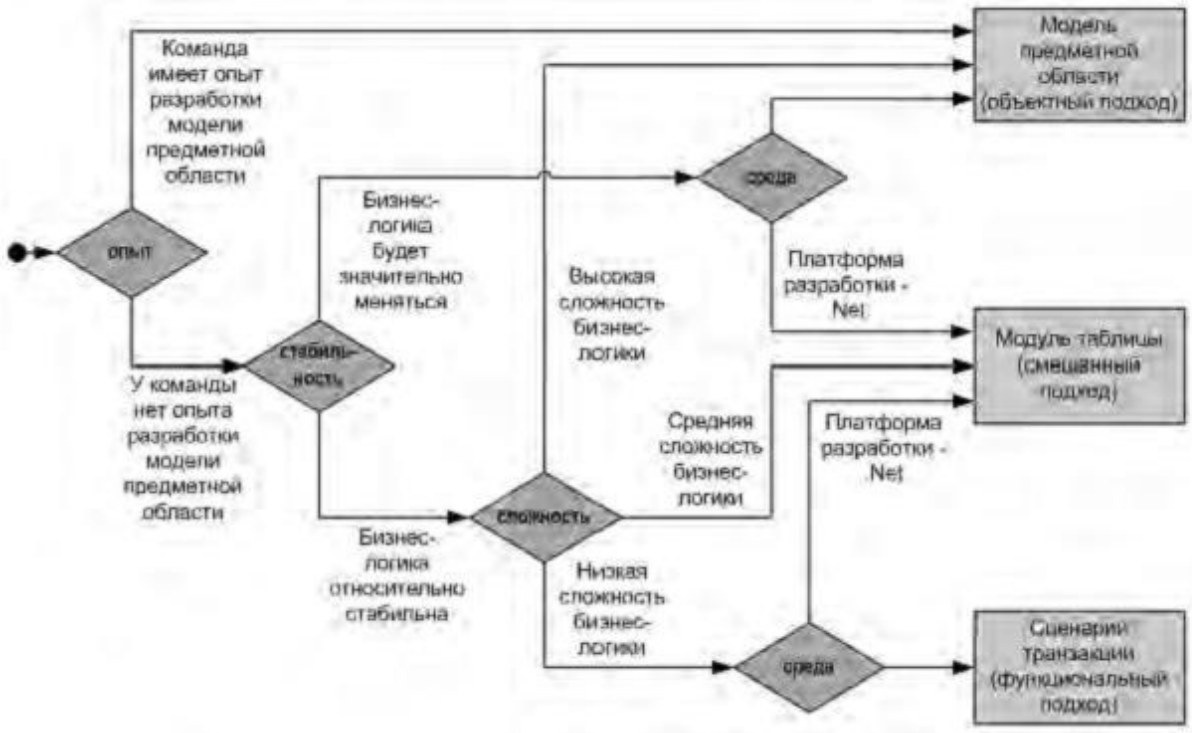

Рисунок1 – Схема принятия решения

Упрощенная схема выбора типового решения для реализации бизнес-логики.

На схеме хорошо видно, что в некоторых случаях выбор сделать очень легко. Например, когда приложение простое, изменений не предвидится, объ-ектная модель является для разработчика слишком сложной, можно применить простой функциональный подход. С другой стороны, команда разработчиков, хорошо владеющих объектным подходом, будет строить объектную модель предметной области во всех случаях.

Пограничные ситуации требуют оценки стабильности и сложности разработки. На предлагаемом уровне это не должно вызвать затруднений.

Данная методика предназначена в первую очередь для оперативного «прокручивания» в голове с целью избежать грубых ошибок при выборе программного решения. Подобная систематизация также представляется полезной как первый шаг программиста к сознательному и ответственному проектированию приложений в противоположность «проектированию на лету».

В реальных проектах, как обычно, все обстоит сложнее. Например, вопрос организации бизнес-логики может теряться на фоне массы других вопросов разработки архитектуры. Однако это не значит, что решение нужно пускать на самотек. Приведенная схема показывает, что можно предельно упростить принятие решения, не теряя при этом возможность выявить «подводные камни».

2 Реализовать бизнес-логику процессов интегрированной информационной системы предприятия.

3 Оформить работу.

4 Осуществить защиту работы.

Контрольные вопросы

1 Опишите функциональный способ организации бизнес-логики программной системы.

2 Опишите смешанный способ организации бизнес-логики программной системы.

3 Опишите объектный способ организации бизнес-логики программной системы.

4 Опишите достоинства и недостатки основных способов организации бизнес-логики программной системы.

8 Лабораторная работа

Лабораторная работа № 5

Тема Разработка (подготовка) документации и отчетных форм для внедрения программных средств

Цель: научиться разрабатывать документацию для внедрения программного продукта

Теоретические сведения

При разработке программных средств (ПС) создается и используется большой объем разнообразной документации.

Она необходима как средство передачи информации между разработчиками программного продукта, как средство управления разработкой, и как средство передачи пользователям информации, необходимой для применения и сопровождения.

На создание этой документации приходится большая доля стоимости программного средства.

Эту документацию можно разбить на две группы:

1. документы управления разработкой программного средства;
2. документы, входящие в состав программного средства.

Документы управления разработкой ПС управляют и протоколируют процессы разработки и сопровождения ПС, обеспечивая связи внутри коллектива разработчиков ПС и между коллективом разработчиков и менеджерами ПС - лицами, управляющими разработкой ПС.

Эти документы могут быть следующих типов:

- планы, оценки, расписания;
- отчеты об использовании ресурсов в процессе разработки;
- стандарты;
- рабочие документы;
- заметки и переписка.

Документы, входящие в состав ПС, описывают программы ПС как с точки зрения их применения пользователями, так и с точки зрения их разработчиков и сопроводителей. Эти документы будут использоваться не

только на стадии эксплуатации ПС, но и на стадии разработки для управления процессом разработки (вместе с рабочими документами).

Эти документы образуют два комплекта с разным назначением:

- пользовательская документация ПС (П-документация).
- документация по сопровождению ПС (С-документация).

Пользовательская документация программных средств

Пользовательская документация объясняет пользователям, как они должны действовать, чтобы применить разрабатываемое ПС. Она необходима, если ПС предполагает какое-либо взаимодействие с пользователями.

К такой документации относятся документы, которыми должен руководствоваться пользователь при инсталляции, при применении для решения своих задач и при управлении.

Эти документы частично затрагивают вопросы сопровождения, но не касаются вопросов, связанных с модификацией программ.

Можно считать типовым составом следующий состав пользовательской документации для достаточно больших ПС:

- общее функциональное описание;
- руководство по инсталляции;
- инструкция по применению;
- справочник по применению;
- руководство по управлению.

Документация по сопровождению программных средств

Документация по сопровождению описывает ПС с точки зрения ее разработки. Эта документация необходима, если ПС предполагает изучение того, как оно устроено (сконструировано), и модернизацию его программ.

Документация по сопровождению ПС можно разбить на две группы:

1. документацию, определяющую строение программ и структур данных ПС и технологию их разработки;
2. документацию, помогающую вносить изменения в ПС.

Документация первой группы содержит итоговые документы каждого технологического этапа разработки ПС.

Она включает следующие документы:

- внешнее описание ПС;
- описание архитектуры ПС, включая внешнюю спецификацию каждой ее программы (подсистемы).
- для каждой программы ПС описание ее модульной структуры, включая внешнюю спецификацию каждого включенного в нее модуля;
- для каждого модуля спецификацию и описание его строения;
- тексты модулей на выбранном языке программирования;
- документы установления достоверности ПС.

Документация второй группы содержит руководство по сопровождению ПС, которое описывает особенности реализации ПС.

В нем также фиксируются, какие части ПС являются аппаратно- и программно-зависимыми.

Документирование ППП

Создание и использование пакета прикладных программ (ППП) от формирования концепции и требований к первой версии до изъятия его из

эксплуатации сопровождается документированием объектов и процессов жизненного цикла ППП.

По своему назначению документацию ППП можно классифицировать как:

1. технологическую документацию процесса разработки, включающую подробные технические описания для специалистов, ведущих проектирование, разработку и сопровождение ППП, обеспечивающую возможность отчуждения, детального освоения, развития и корректировки ими программ и баз данных на всем жизненном цикле ППП;

2. эксплуатационную (пользовательскую) документацию программного продукта, создаваемую для конечных пользователей пакета и позволяющую им осваивать и квалифицированно применять его для решения конкретных прикладных задач.

Технологическая документация включает:

- проектную документацию;
- документацию тестирования компонентов и комплексов про-грамм;
- документацию испытаний ППП;
- документацию сопровождения и управления конфигурацией ППП.

В состав проектной документации входят:

- отчет по обследованию предметной области, для которой предназначен разрабатываемый ППП, с описанием комплекса задач;
- описание концепции проектирования;
- техническое задание на проектирование;
- план-график работ;
- спецификации эскизного и технического проекта;
- документация на разработанные программные модули пакета;
- общее описание программного обеспечения, используемого при разработке и функционировании пакета.

В состав документации тестирования входят:

- исходные данные для проведения тестирования (методы тестирования, тестовые наборы, эталонные значения, реальные ресурсы тестирования - временные, аппаратно-программные, людские, критерии полноты и качества тестирования);
- программа (сценарии) тестирования;
- журнал тестирования;
- итоговый отчет о результатах тестирования.

В состав документации испытаний входят:

- программа испытаний;
- описание методов и методик испытаний;
- протоколы испытаний;
- акт завершения работ;
- акт приемки ППП в эксплуатацию.

В состав документации сопровождения управления конфигу-рацией входят:

 отчеты пользователей о выявленных дефектах и предложения по корректировке программ;
 журнал выявленных дефектов и предложений по совершенствованию и развитию версии ППП;
 журнал подготовленных и утвержденных корректировок, а также реализованных изменений в новой версии пакета;
 отчет о результатах эксплуатации снятой с сопровождения версии пакета;
 журнал тиражирования и характеристик базовых версий, поддерживаемых сопровождением.

Пользовательская документация включает в себя:
 паспорт на программное средство;
 общее описание информационной системы (ИС), в составе которой будет использоваться ППП;
 руководство администратора программного средства,
 руководства оперативных пользователей с требованиями к уровню подготовки пользователя, описание функций.

Методические указания

I. Разработайте регламент внедрения программного продукта, который состоит из следующих пунктов:

1. Титульный лист
2. Общие положения, в которых перечислены основные задачи разработки регламента:
2.1. Определения
2.2. Участники деятельности
2.3. Этапы разработки и внедрения
3. Постановка задачи и запуск проекта
4. Техническое задание должно содержать в себе, следующую информацию:
4.1. цель автоматизации;
4.2. наименование и краткую характеристику системы;
4.3. назначение и функции предмета разработки;
4.4. требования к предмету разработки, в том числе к функциональным характеристикам, надежности, справочной информации и

др.;

4.5. требования к видам обеспечения, требования к информационному обеспечению, условия работы;
4.6. порядок выполнения работ по созданию системы с указанием содержания работ;
4.7. особые требования к проведению приемки работ;
4.8. условия взаимодействия с другими проектами;
4.9. другая необходимая информация.
5. Порядок выполнения работ и внедрения программных продуктов

Контрольные вопросы

1. Перечислите пользовательскую документацию?

2. Для каких целей разрабатывают «Техническое задание»?
3. Перечислите ГОСТы в области разработки документации программного обеспечения.