МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное бюджетное образовательное учреждение высшего образования
«Кузбасский государственный технический университет имени Т. Ф. Горбачева»

Кафедра информационных и автоматизированных производственных систем

Составители
О. Н. Ванеев
О. А. Ивина

УСТРОЙСТВО И ФУНКЦИОНИРОВАНИЕ ИНФОРМАЦИОННОЙ СИСТЕМЫ

Методические материалы

Рекомендовано цикловой методической комиссией специальности
СПО 09.02.07 Информационные системы и программирование
в качестве электронного издания для использования
в образовательном процессе

Кемерово 2018

Рецензенты

Сыркин И. С. – кандидат технических наук, доцент кафедры информационных и автоматизированных производственных систем ФГБОУ ВО «Кузбасский государственный технический университет имени Т. Ф. Горбачева»

Чичерин И. В. – кандидат технических наук, доцент, зав. кафедрой информационных и автоматизированных производственных систем ФГБОУ ВО «Кузбасский государственный технический университет имени Т. Ф. Горбачева»

Ванеев Олег Николаевич, Ивина Оксана Анатольевна

Устройство и функционирование информационной системы: методические материалы [Электронный ресурс] для студентов специальности СПО 09.02.07 Информационные системы и программирование очной формы обучения / сост. О. Н. Ванеев, О. А. Ивина; КузГТУ. – Электрон. издан. – Кемерово, 2018.

Методические материалы для МДК.06.03 «Устройство и функционирование информационной системы» описывают содержание практических, лабораторных и самостоятельных занятий, перечень вопросов на защиту выполненных работ.

РАБОТА №1 ЛАБОРАТОРНАЯ. ФОРМИРОВАНИЕ ПРЕДЛОЖЕНИЙ О РАСШИРЕНИИ ИНФОРМАЦИОННОЙ СИСТЕМЫ

1. ЦЕЛЬ И ЗАДАЧИ РАБОТЫ

Целью выполнения лабораторных работ является: получить опыт практической работы в системе управления версиями файлов Subversion (SVN). Получить навыки создания репозитория, научиться отслеживать, противоречащих друг другу изменения. Получить практику командной работы над проектом.

2. КРАТКИЕ ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

Каждый программный/программно-аппаратный проект проходит в своей жизни несколько последовательных стадий: постановка задачи, поиск вариантов решения, написание кода и его отладка, поддержка проекта. При этом многие разработчики и даже команды разработчиков вручную фиксируют историю изменений проекта (при его разработке и поддержке), осуществляют формирование готовых релизов (временных слепков состояния проекта), ветвление проекта (если нужна его адаптация под конкретные условия ТЗ или нужно проверить пару идей). Естественно, такая работа с изменяющейся информацией чревата большой путаницей и частыми ошибками. Для автоматизации этой работы и были разработаны системы управления версиями файлов. Существует достаточно большое количество систем управления версиями файлов: CVS, SVN, Bazzar, Git и т. д. Выбор той или иной системы определяется личными предпочтениями разработчика и его требованиями. Но с точки зрения пользователя все эти системы похожи и решают одну задачу: контроль и управление версиями файлов.

Системы управления версиями файлов берут на себя работу по управлению версиями и ревизиями исходных файлов проекта, то есть:

1) больше не нужно помнить обо всех исправленных ошибках и модификациях, все это есть в системе;

2) нет смысла складировать ревизии на жестком диске, все это уже лежит в репозитории системы;

3) нет смысла плодить много папок проектов с незначительными отличиями и помнить, в чем эти отличия;

4) нет смысла переносить проекты вручную, достаточно получить их из репозитория в новом месте;

5) нет смысла вручную поддерживать когерентность последних библиотек в проектах, которые их используют.

Все это возможно благодаря тому, что эти системы автоматически ведут историю изменений файла. Эта история, в пределах одного репозитория, может развиваться только в одном направлении. Проще говоря, репозиторий никогда ничего не забывает. Единственное, что требуется от пользователя, это аккуратно заполнять комментарии к изменениям версий файлов для быстрой навигации по репозиторию. То есть, системы управления версиями файлов нужны для того, чтобы автоматизировать рутинную работу по отслеживанию изменений файлов и облегчить жизнь разработчику. Но использование систем управления версиями файлов не заменяет документирование проекта, а всего лишь помогает быстро и просто получить ответы на вопросы, кто, когда, что и зачем изменял в файлах.

Subversion (SVN) – бесплатная система управления версиями файлов с открытым исходным кодом. SVN позволяет управлять файлами и каталогами, а также сделанными в них изменениями во времени. SVN предоставляет следующие возможности:

1. Контроль изменений каталогов. SVN использует «виртуальную» файловую систему с возможностями управления версиями, которая способна отслеживать изменения во времени целых структур каталогов.

2. Настоящая история версий. SVN делает возможным добавление, удаление, копирование и переименование как файлов, так и каталогов. При этом каждый вновь добавленный файл начинает жизнь с чистого листа, сохраняя собственную историю изменений.

3. Атомарная фиксация изменений. Каждый набор изменений либо попадает в хранилище целиком, либо не попадает туда вовсе. То есть если при фиксации изменений проекта произошла ошибка при обработке файла, то изменения всего проекта не будут зафиксированы.

4. Метаданные с версиями. Каждый файл и каталог имеет собственный набор свойств, представленных в виде названия и значения. Вы можете создавать и сохранять любые необходимые пары

названий свойств и их значений. Свойства файлов точно так же находятся под управлением системы, как и их содержимое.

5. Единый способ работы с данными. SVN обнаруживает различия между файлами с помощью специального бинарного алгоритма, который одинаково работает как с текстовыми, так и с бинарными файлами. Файлы записываются в хранилище в сжатом виде независимо от их типа, а различия между отдельными версиями могут передаваться по сети в обоих направлениях.

6. Эффективные ветки и метки. SVN создает ветки и метки путем простого копирования проекта, используя механизм, похожий на жесткие ссылки в файловых системах. Благодаря этому операции по созданию веток и меток занимают немного времени.

Основные термины и команды SVN:

• Репозиторий (repository) – централизованное хранилище исходных кодов, рабочих материалов и документации. Любое количество клиентов подключается к хранилищу и читает или записывает эти файлы.

• Рабочая копия (working copy/WC) – обычное дерево каталогов на диске компьютера, содержащее набор файлов для работы над проектом. Изменения в рабочей копии недоступны для других пользователей репозитория до тех пор, пока они не будут зафиксированы.

• Trunk (ствол) – основное направление разработки.

• Branch (ветка) – направление разработки, которое существует независимо от другого направления, но имеет с ним общую историю. Ветка всегда начинается как копия чего-либо и движется от этой точки, создавая свою собственную историю.

• Tag (метка) – срез состояния проекта в определенный момент времени. Отделяется от основного проекта явно, через создание отдельной папки.

• Revision – номер ревизии репозитория, в пределах репозитория номер ревизии является уникальной величиной.

• Checkout – команда, которая выполняет начальное извлечение файлов проекта из репозитория в WC и помещает эти файлы под контроль SVN.

• Commit – команда, которая выполняет фиксацию изменений файлов проекта в WC в репозиторий.

• Update – команда, которая выполняет синхронизацию файлов проекта в WC с файлами проекта из репозитория.

• Revert – команда, которая выполняет отмену любых изменений в файлах проекта в WC.

• Merge – команда, которая выполняет слияние файлов из разных веток проекта и помещает результат слияния в WC.

• Conflict – ситуация, возникающая при фиксации изменений, когда одни и те же файлы изменяли несколько разработчиков.

• Resolve – набор правил по разрешению возникающих конфликтов.

• Import – команда для быстрого копирования дерева файлов в репозиторий.

• Export – команда для быстрого копирования файлов проекта из репозитория.

• Switch – команда, которая выполняет переключение WC на другую ветку разработки.

• Create, Add, Delete, Copy, Move, Rename – команды для управления файлами и папками в репозитории или WC.

Рассмотрим работу с SVN рассмотрена на основе демонстрационного репозитория demo_repo и тестового проекта demo_project, который изначально содержит один файл readme.txt.

Репозиторий SVN. Основными объектами при работе с SVN являются рабочая копия (WC) и репозиторий. В репозитории SVN хранятся все структуры папок и файлов. Репозиторий хранит все изменения, зафиксированные в нем, с момента создания. Для отслеживания изменений во времени каждой операции, которая изменяет содержимое репозитория, присваивается уникальный «номер ревизии», запоминаются время и автор изменения. Все ревизии папок и файлов в репозитории доступны любому пользователю. SVN запоминает все изменения, даже самые небольшие. Для облегчения поиска нужного состояния проекта рекомендуется заполнять строку комментария к любым изменениям в репозитории. Пустые строки комментария к фиксации изменений оставлять не принято. Данные виды фиксаций очень сложно отслеживать и искать. В случае удаления папок или файлов из репозитория они будут удалены только в текущей ревизии. И в случае необходимости папки и файлы могут быть легко восстановлены.

Основной областью работы пользователя является рабочая копия. Любые изменения папок, файлов и их содержимого в рабочей копии недоступны для других пользователей до тех пор, пока эти изменения не будут зафиксированы в репозитории. Добавлять, перемещать и удалять папки и файлы проекта лучше в рабочей копии. Использовать для этих целей репозиторий не рекомендуется. Использовать возможности репозитория для целей управления папками и файлами можно только в случае, если нужное действие сложно сделать в рабочей копии. К таким действиям относится копирование и перемещение папок и файлов вне пределов рабочей копии. При работе с репозиторием помните, что у папок и файлов есть история изменений. Если вы удалите файл и создадите новый файл с таким же именем, это будут два разных файла, с непересекающейся историей.

Создание репозитория Репозиторий SVN может быть, как сетевым, так и локальным. Сетевые репозитории нужны для использования командой разработчиков или в случае, когда необходим доступ к репозиторию с другого компьютера. Локальные репозитории удобно использовать тогда, когда разработчик один. Локальный репозиторий можно сделать в любом месте на любом жестком диске. Для этого нужно создать папку репозитория, в нашем случае это D:\demo_repo, и выполнить в ней команду Create repository here, репозиторий создан.

Можно приступать к работе над проектом. Важное замечание. Операции над папками репозитория, в обход команд клиента SVN, могут привести к повреждению репозитория и возможной потере информации в нем. Репозиторий использует специальный алгоритм хранения файлов. При обновлении версии файла он не создает новый файл, а создает файл разницы между этими двумя файлами. Эта разница и сохраняется в репозитории. Это сделано для экономии места и экономии трафика (если используется сетевой репозиторий).

Работа с репозиторием. Работа с репозиторием является обязательной составляющей работы с проектами, находящимися под контролем SVN. Для работы с репозиторием используется браузер репозитория (repo-browser). Чтобы им воспользоваться, заходим в корневую папку любого жесткого диска и путем нажатия правой кнопки мыши запускаем браузер.

Для просмотра репозитория нужно указать, к какому именно репозиторию будет обращение. Префикс *file:///* указывает на то, что репозиторий *D:\demo_repo* является локальным. Наш репозиторий пока пуст.

Создание проекта. Для того чтобы поместить свои папки и файлы под контроль SVN, необходимо создать первоначальный проект. Сделать это можно двумя способами.

Создание проекта в репозитории. Этот способ подходит для создания изначально пустого проекта. Заходим в репозиторий. В корневой папке репозитория, используя команду, создаем корневую папку нашего проекта *demo_project.* Обязательно оставляем комментарий к действиям по изменению репозитория. Подобную последовательность шагов проделываем для обязательных папок, назначение которых описано ниже, в итоге получим пустой проект в репозитории.

Импорт проекта. Этот способ больше подходит для случая, когда есть готовый проект и нужно поместить его под контроль SVN. Возьмем тестовый проект demo_project1. С помощью команды Import импортируем его в репозиторий. Не забываем про указание имени папки, в которую мы импортируем проект, и о комментарии к изменению репозитория. После окончания импорта в окне протокола видно, что происходило с папками и файлами проекта. все папки и файл проекта были добавлены в репозиторий одной командой. При импорте проекта в репозиторий будут добавлены все файлы, которые находятся в папках проекта. То, что импорт проекта завершен удачно, не означает, что папка, которую вы использовали для импорта проекта, является теперь рабочей копией. Любые изменения в этой папке не могут быть зафиксированы в репозитории SVN. При импорте проекта убедитесь в том, что проект с таким именем не существует, и в том, что вы создаете проект в корневой папке репозитория.

При импорте проекта в репозиторий будут добавлены все файлы, которые находятся в папках проекта. Поэтому перед импортом проверьте проект на предмет чистки от ненужных файлов. Какие файлы не имеет смысла хранить в SVN, будет рассмотрено далее. При создании проектов очень удобно использовать шаблоны проектов. В таком случае, используя шаблон, вы импортируете проект в

репозиторий. Это избавит вас от необходимости каждый раз создавать повторяющиеся от проекта к проекту папки и файлы.

Если вы неправильно создали проект (не те имена папок, не туда поместили по ошибке), то сначала удалите командой *Delete* все неправильно созданное в репозитории и только потом создавайте проект заново. Или воспользуйтесь командой *Move* для того, чтобы перенести папки проекта в другое место (в *TortoiseSVN* в браузере репозитория команда *Move* выполняется по технологии drag-and-drop).

Создание рабочей копии. Для того чтобы начать работу с проектом, нужно создать рабочую копию. Для этого создаем корневую папку проекта на диске. Заходим в эту папку и используем команду *Checkout*. С помощью браузера репозитория выбираем интересующий нас проект, папки и файлы в этом проекте. С помощью браузера ревизий *Show log* выбираем нужный номер ревизии. В данном случае выбираем *Head* (головная/последняя) ревизия. Нажимаем на *OK*. Все, рабочая копия создана и находится под контролем SVN, о чем говорит иконка SVN на файлах и папках. Теперь можно свободно изменять, удалять, модифицировать папки и файлы проекта, добавлять новые папки и файлы, не опасаясь того, что ваши изменения будут мешать работе ваших коллег. При *Checkout* проекта из репозитория в проекте создаются служебные папки ***.svn***. Изменять или удалять их нельзя. В случае повреждения служебных папок информация о сделанных вами изменениях не может быть зафиксирована в репозитории. Узнать, к какому репозиторию и проекту относится папка или файл, можно из свойств папки/файла, которая находится под контролем репозитория. Если нужно передать куда-то файлы проекта, воспользуйтесь командой *Export*. Эта команда не создает служебных папок.

Ознакомимся с некоторыми командами, которые может выполнять клиентская часть. Наиболее часто используемые разработчиками команды:

- *svn update* – обновляет содержимое локальной копии до самой последней версии из репозитория.
- *svn commit* – отправляет все изменения локальной копии в репозиторий.
- *svn add <файл/папка>* – включить файл/папку в локальную копию проекта

- *svn move <файл/папка1> <папка2>* – переместить файл/папку1 в папку2
- *svn copy <файл/папка> <папка>* – скопировать файл/папку1 в папку2
- *svn delete <файл/папка>* – удалить файл/папку из локальной копии проекта
- *svn list <URL>* – просмотр каталога репозитория
- *svn log <файл> -- verbose* – история изменения файла по ревизиям
- *svn cat -- revision <номер_ревизии> <файл>* – отображение содержимого файла из данной ревизии
- *svn iff --revision <номер_ревизии1: номер_ревизии2> <файл>* – отображение изменений файла между двумя ревизиями
- *svn status* – отображение изменений в локальной копии относительно репозитория

3. ПОРЯДОК ВЫПОЛНЕНИЯ РАБОТЫ

Перед выполнением лабораторной работы, необходимо создать на одном из компьютеров (или сервере) пустой проект так чтобы он был доступен всем выполняющим работу.

1. Скачать проект из общего репозитория группы.
2. Добавить в проект свою папку с исходными файлами.
3. Внести добавленную папку под SVN.
4. Изменить метод ввода/вывода, так чтобы ввод дешифруемого сообщения осуществлялся из файла input.txt, а вывод осуществлялся в файл output.txt каталога, в котором находиться программа.
5. Внести результаты под SVN.
6. Оформить внесённые изменения в виде патча, т. е. в каталоге patch создать два каталога: каталог, содержащий только изменённые файлы, и каталог backup содержащий эти же файлы до изменений.
7. Выполнить команду Update на папке общего проекта. Убедиться в наличии файлов run.bat и group_result.txt.
8. Добавить в run.bat строчки, запускающие ваше приложение и записывающие результат работы из [вашего каталога]/output.txt в общий файл group_result.txt

9. Скорректировать hook отвечающий за обработку вносимых под SVN файлов таким образом, чтобы операция commit выполнялась только если в качестве комментария указана фамилия студента, выполняющего работу.

10. Внести результаты под SVN.

11. Получить лог изменений файла group_result.txt.

В отчёте зафиксировать все произведённые действия. Приложить копии экрана (Print Screen), с краткими пояснениями. Сделать выводы по основным пунктам лабораторной работы.

4. КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Что такое Хранилище (Repository)?
2. Что такое Рабочая копия (Working Copy)?
3. Что такое версия?
4. Понятия управления контроля версиями, применения контроля версий.
5. Чем помогают системы контроля версий типа SVN?
6. Опишите механизм контроля версий.

РАБОТА №2 ПРАКТИЧЕСКАЯ. ОБСЛУЖИВАНИЕ ЛОКАЛЬНОЙ СЕТИ

1. ЦЕЛИ И ЗАДАЧИ РАБОТЫ

Целью работы является получение практических навыков по выполнению экспорта настроек командной среды разработки.

2. КРАТКИЕ ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

Современные интегрированные среды. Точно настроив интегрированную среду разработки по своему вкусу, вы можете сохранить эти настройки на будущее. Для этого можно экспортировать настройки интегрированной среды в файл или даже передать ряду инсталляций системы Visual Studio 2013, чтобы во всех установленных системах были одинаковые настройки. Точно настроив интегрированную среду разработки по своему вкусу, вы можете сохранить эти настройки на будущее. Для этого можно экспортировать настройки интегрированной среды в файл или даже передать ряду инсталляций системы Visual Studio 2013, чтобы во всех установленных системах были одинаковые настройки

Для того чтобы экспортировать выбранную конфигурацию, выберите команду Tools --> Import and Export Settings, чтобы запустить мастер Import and Export Settings Wizard, как показано на рисунке ниже. На первом этапе работы этого мастера следует выбрать настройку Export, а также настройки, которые следует сохранить при выполнении процедуры экспорта.

Можно экспортировать множество сгруппированных настроек, открывая раздел Options следует сохранить настройки Debugging и Projects, а также конфигурации Text Editor и Windows Forms Designer. Маленькие пиктограммы с восклицательным знаком свидетельствуют о том, что некоторые настройки не были предназначены для экспорта по умолчанию, поскольку они содержат информацию, которая может нарушить конфиденциальность информации. В этом разделе вы должны сделать свой выбор вручную, если действительно хотите, чтобы эта информация была сохранена в резервном файле. Выбрав настройки, которые хотите экспортировать, переходите к следующему этапу работы мастера, который может за-

нять несколько минут, в зависимости от того, сколько настроек вы экспортируете.

Импортировать файл настроек очень легко. Для этого используется тот же самый мастер, но теперь на первом этапе вам следует выбрать команду Import. Вместо простой перезаписи текущей конфигурации мастер позволяет вам сначала сохранить резервную копию текущих настроек.

Затем можете выбрать существующий файл конфигурации из списка. Это тот же самый список файлов, из которого вы выбираете настройки при первом запуске системы Visual Studio 2013. Кроме того, можете просмотреть файлы настроек, созданные вами заранее. Выбрав файл настроек, можете импортировать только разделы конфигурации или всю ее целиком.

Мастер по умолчанию исключает несколько разделов, таких как External Tools или Command Aliases, чтобы вы не могли непреднамеренно уничтожить пользовательские настройки. Убедитесь, что вы выбрали эти разделы, если хотите восстановить все настройки.

Если вы просто хотите восстановить одну из конфигураций системы Visual Studio 2013, заданных по умолчанию, выполните *команду Reset All Settings* на первом этапе работы мастера и не выполняйте весь процесс импорта.

Visual Studio предоставляет возможность поделиться настройками с членами команды, с которой вы работаете. Это полезно в тех случаях, например, когда происходит редактирование одних и тех же файлов (используя Team Foundation Server). Каждый программист по-своему оформляет код, использует символы табуляции и т. д. Когда разные члены команды работают с одним файлом, простое редактирование файла может привести к незначительным изменениям, не влияющим на работу приложения (лишние пробелы, переносы строк и т. д.) Однако, когда эти файлы добавляются в репозиторий исходного кода (TFS, Git, ...), эти изменения могут вызвать проблемы.

Если вы работаете с командой разработчиков возможно создание единого файла настроек. В разделе настроек Environment --> Import and Export Settings вы можете установить *флажок Use Team Settings File.* Когда этот флажок установлен, должен быть указан путь, где будет храниться общий файл настроек.

Настройки синхронизации. Одним из нововведений среды Visual Studio 2013 является ее синхронизация с облачными службами. Вы можете войти в Visual Studio с учетной записью Microsoft и настройки Visual Studio будут синхронизированы на всех ваших компьютерах. Этот процесс синхронизации не распространяется на все настройки Visual Studio. При отсутствии дополнительных настроек с вашей стороны, синхронизация затрагивает следующие параметры:

- Настройки разработчика (это те параметры, которые были указаны при установке Visual Studio и первом ее запуске).
- Настройка цветовой темы (Environment --> General).
- Настройки шрифтов и цвета (Environment --> Fonts and Colors).
- Горячие клавиши (Environment --> Keyboard).
- Параметры запуска (Environment --> Startup).
- Настройки текстового редактора (Text Editor).
- Все пользовательские псевдонимы команд.

Можно изменить настройки синхронизации или вообще полностью выключить. Для этого, в диалоговом окне Options выберете узел Environment --> Synchronized Settings, где можно увидеть уровень детализации, доступный для вас. Чтобы полностью отключить синхронизацию, убедитесь, что флажок включения синхронизации не установлен для всех категорий параметров.

Эти многочисленные настройки позволяют настраивать редактирование кода, добавлять элементы управления в вашу форму и даже выбирать методы при отладке кода. Настройки, доступные в диалоговом окне Options, также позволяют вам управлять процессом создания приложений и даже задавать собственные комбинации клавиш для выполнения команд. Также можно обращаться к диалоговому окну Options для настройки процессов компиляции, отладки и написания макросов.

3. ПОРЯДОК ВЫПОЛНЕНИЯ РАБОТЫ

Необходимо настроить интегрированную среду разработки по заданию преподавателя и сохранить данные настройки

1. Экспортируйте настройки интегрированной среды в файл Выберите команду Tools --> Import and Export Settings. На первом

этапе работы этого мастера следует выбрать настройку Export, а также настройки, которые следует сохранить при выполнении процедуры экспорта.

В разделе Options сохраните настройки Debugging и Projects, а также конфигурации Text Editor и Windows Forms Designer. Выбрав настройки, которые хотите экспортировать, переходите к следующему этапу работы мастера

2. Импортируйте файл настроек. Для этого используется тот же самый мастер, но теперь на первом этапе вам следует выбрать команду Import. Затем можете выбрать существующий файл конфигурации из списка. Н этом этапе просмотрите файлы настроек, созданные вами заранее. Выбрав файл настроек, можете импортировать только разделы конфигурации или всю ее целиком.

Для восстановления настроек убедитесь, что выбрали разделы External Tools или Command Aliases, которые могут быть исключены Мастером по умолчанию.

Для восстановления одну из конфигураций системы Visual Studio 2013, заданных по умолчанию, выполните *команду Reset All Settings* на первом этапе работы мастера. На этом процесс импорта будет завершен.

3. Поделитесь настройками с членами команды, с которой вы работаете.

4. Создайте единый файл настроек. В разделе настроек Environment --> Import and Export Settings установите *флажок Use Team Settings File*. Укажите путь, где будет храниться общий файл настроек.

5. Измените настройки синхронизации:

- Настройки разработчика.
- Настройка цветовой темы.
- Настройки шрифтов и цвета.
- Горячие клавиши.
- Параметры запуска.
- Настройки текстового редактора.
- Все пользовательские псевдонимы команд.

6. Выключите настройки синхронизации. Для этого, в диалоговом окне Options выберете узел Environment --> Synchronized Settings, убедитесь, что флажок включения синхронизации не установлен для всех категорий параметров.

4. КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Настройка интегрированной среды.
2. Экспорт выбранной конфигурации Visual Studio 2013.
3. Как восстановить конфигурацию Visual Studio 2013?
4. Импорт заданных конфигураций Visual Studio 2013.
5. Visual Studio 2013 – синхронизация с облачными службами.
6. Как можно изменить настройки синхронизации Visual Studio 2013?

РАБОТА №3 ЛАБОРАТОРНАЯ. ОБСЛУЖИВАНИЕ СИСТЕМЫ БУХГАЛТЕРСКОГО УЧЁТА

1. ЦЕЛЬ И ЗАДАЧИ РАБОТЫ

Изучить особенности сопровождения информационных систем бухгалтерского учета и материально-технического снабжения. Получить опыт инсталляции системы 1с Предприятие и создания информационной базы.

2. КРАТКИЕ ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

Установка системы 1С Предприятие и создание информационной базы.

В архитектуре программного комплекса 1С:Предприятия выделяется следующие составляющие:

- платформа;
- информационная база;
- база данных;
- конфигурация.

Платформа – набор компонентов, обеспечивающих возможность работы с системой. Информационная база – это конкретное описание того, какие данные будут использоваться, алгоритмов работы с данными и сами данные. Одна платформа может быть связана с несколькими информационными базами.

Для того, чтобы установить новый экземпляр прикладного решения на основе 1С Предприятие, то есть экземпляра программного комплекса, обеспечивающего решение отдельной задачи по учёту деятельности, необходимо:

- установить платформу;
- создать информационную базу.

Одна платформа может поддерживать работу с несколькими информационными базами. Установка платформы производится на основе отдельного экземпляра установочного пакета, обеспечиваемого ключом защиты.

Отдельную информационную базу можно рассматривать как экземпляр обеспечивающий ведение отдельного учёта.

Создание новой информационной базы выполняется при выборе команды «добавить» в окне запуска системы 1С Предприятие.

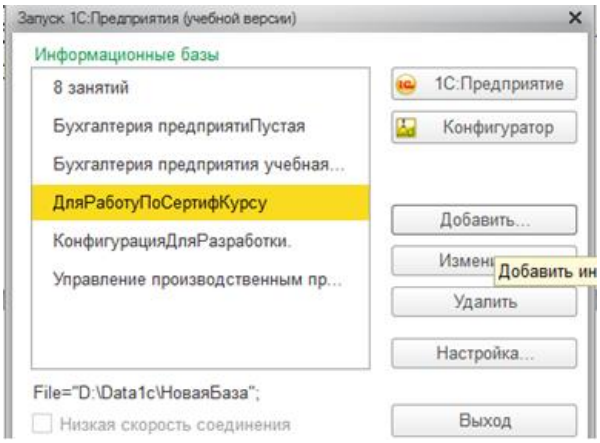


Рис.1.1. Вызов команды создания новой информационной базы

При этом не нужно путать создание новой информационной базы с настройкой системы на каталог существующей информационной базы.

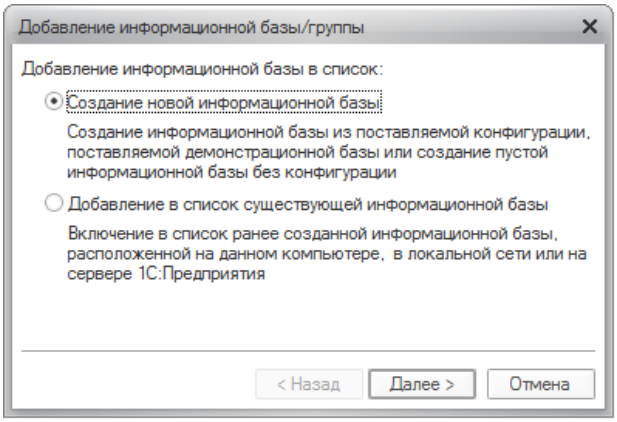


Рис.1.2. Окно команды добавление новой информационной базы

Создание информационной базы может производиться:

- на основе ранее разработанного прикладного решения, представленного в виде комплекта поставки;
- на основе ранее установленного шаблона;
- на основе файла конфигурации;
- информационная база может быть создана полностью вручную.

При установке информационной базы на *основе поставляемых компонентов* создаётся новый шаблон в каталоге шаблонов. Шаб-

лон можно считать тем образцом, на основе которого может быть создана новая информационная база. Шаблон включает файл конфигурации, описывающий используемые объекты в информационной базе и методы работы ними – «1Cv8.cf». Кроме того, при установке создаётся файл обновления предшествующих совместимых информационных баз до устанавливаемой конфигурации – «1Cv8.cfu», файл с выгрузкой демонстрационной базы – «1Cv8.dt», каталоги с дополнительными обработками и внешними отчётами.

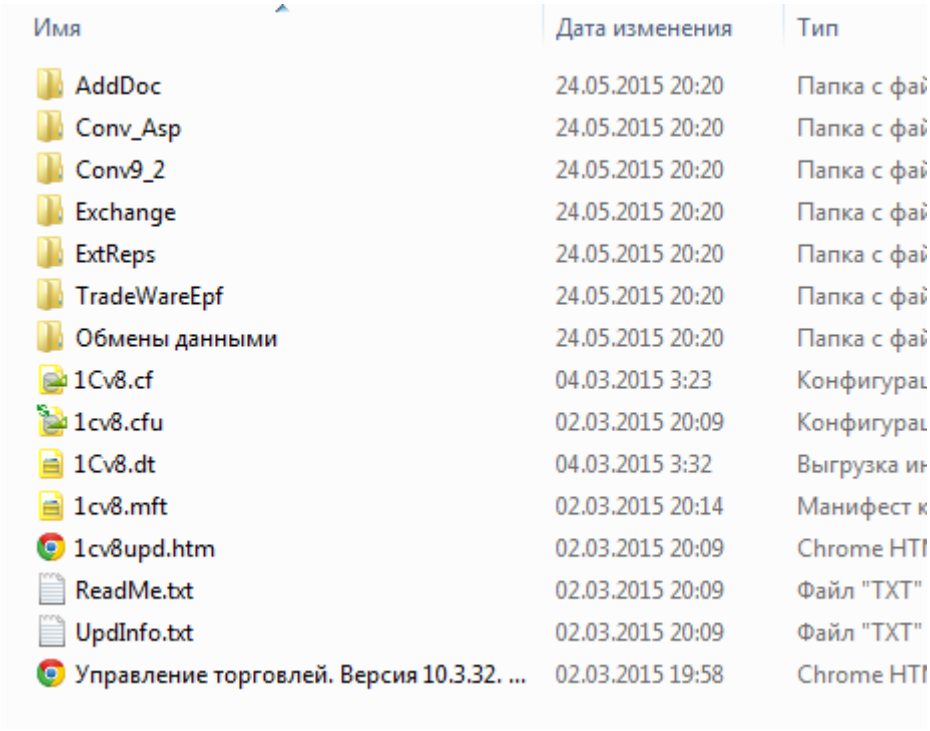


Рис.1.3. Содержимое каталога отдельного шаблона

Положение каталогов шаблонов настраивается в окне запуска кнопка настройка.

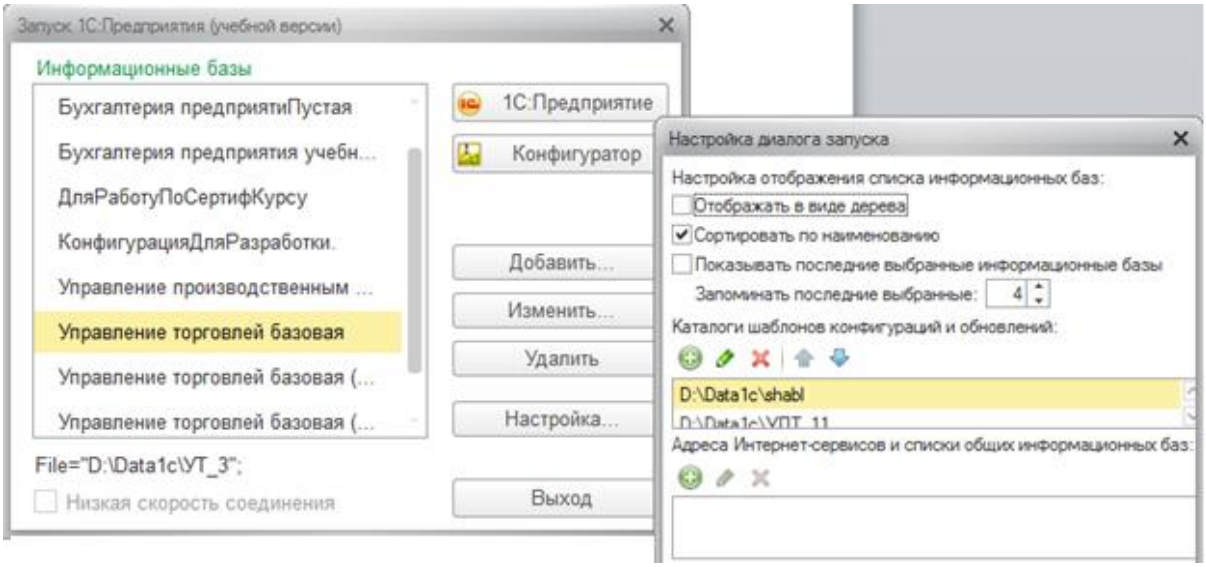


Рис.1.4. Настройка каталогов шаблонов

Новый шаблон в папку шаблонов можно установить и вручную, скопировав из любого источника. При этом необходимо соответственно настроить при этом каталог шаблонов.

Создание новой информационной базы может производиться на основе одного из существующих шаблонов.

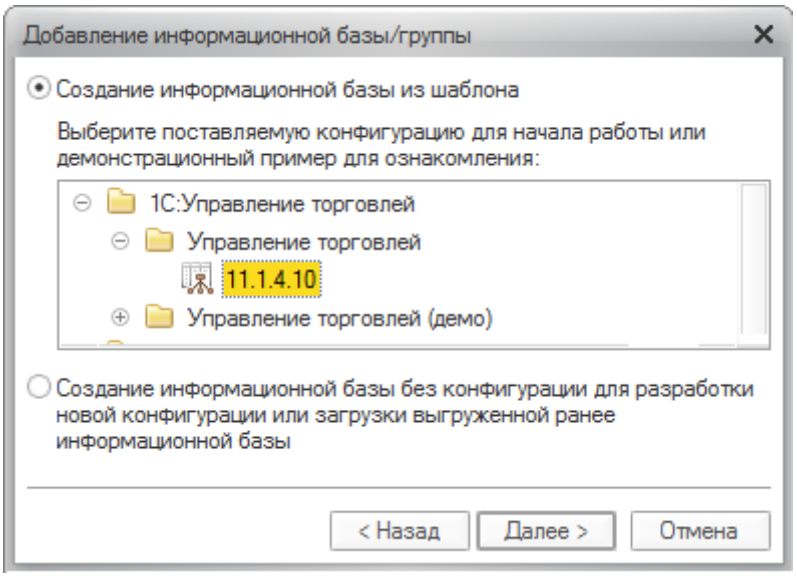


Рис.1.5. Создание новой информационной базы на основе шаблона

Для создаваемой информационной базы необходимо указать каталог, где будут находиться данные информационной базы.

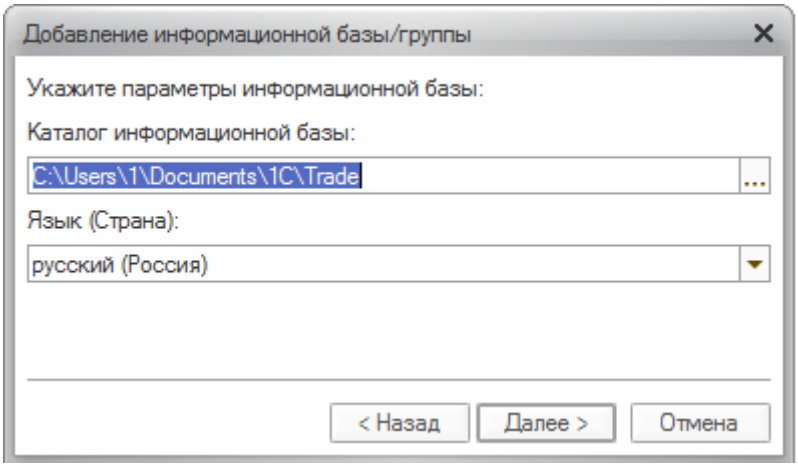


Рис.1.6. Задание каталога создаваемой информационной базы

Создание информационной базы на основе файла конфигурации. Создать новую информационную базу необходимого прикладного решения (конфигурации) можно и на основе файла конфигурации 1Cv8.cf. Данный файл можно получить из рабочего прикладного решения командной в режиме конфигуратора *Конфигурация/Сохранить файл конфигурации*. В данном файле содержится описание документов и всех элементов, на основе которых построено прикладное решение. Предварительно информационная база должна быть создана как база данных без шаблона, можно использовать также ранее существующую ненужную информационную базу, так как все данные будут уничтожены.

После создания информационной базы необходимо в режиме конфигуратора задать пользователей системы.

После установки можно зайти в режим предприятия под учётной записью созданного пользователя.

Работа в режиме предприятие в конфигурации «Управление торговлей».

«Управление торговлей» – является одной из часто используемых прикладных решений на основе платформы 1с8.Х.

Основными объектами среды выполнения, то есть объектами, которые видит, и с которыми работает конечный пользователь являются:

Документы – соответствуют хозяйственной операции.

В конфигурациях бухгалтерского учёта существуют элементы более детальные, чем документы – операции и проводки.

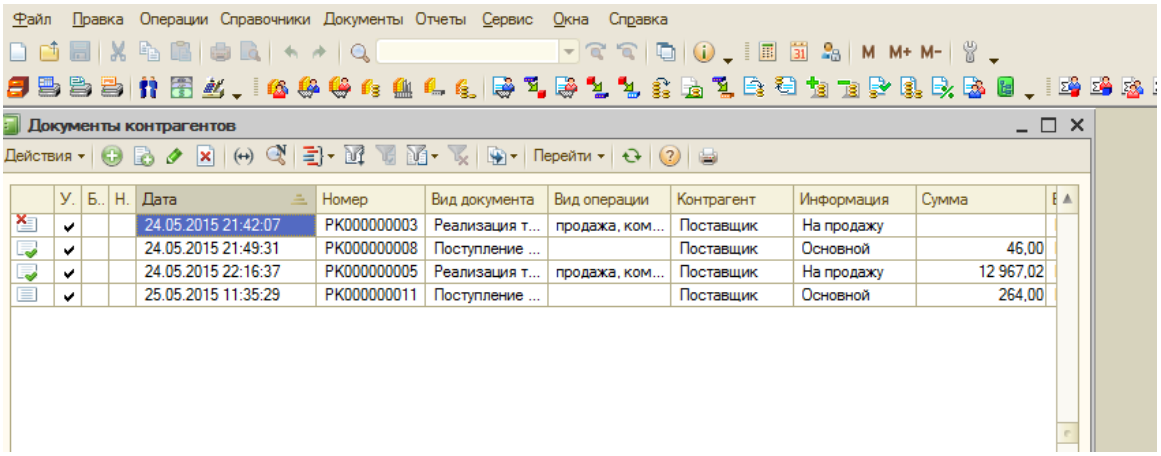
Они будут рассматриваться при изучении особенностей конфигураций бухгалтерского учёта данной конфигурации.

Справочники – агрегатный объект, использующийся для хранения однотипных сложных объектов. Справочники можно рассматривать, как базовые классификаторы, используемые в системе. *Справочники содержат перечень элементов, которые являются атрибутами, документов (контрагенты, номенклатура, материалы, сотрудники).*

Константа – объект, хранящий единичные данные. В константах могут храниться основные параметры работы прикладного решения.

Перечисления – агрегатный объект, содержащий несколько элементов одного типа.

Журналы документов – регистрационная книга, реестр, где отображаются документы (проводки, операции). *Обычно используется несколько типов журналов (отдельные для документов определённого типа), кроме того, всегда присутствует общий журнал, отображающий всех типов. Журнал можно использовать для создания, корректировки документов, выборки документов требуемого типа.*



У.	Б..	Н.	Дата	Номер	Вид документа	Вид операции	Контрагент	Информация	Сумма
✔			24.05.2015 21:42:07	РК000000003	Реализация т...	продажа, ком...	Поставщик	На продажу	
✔			24.05.2015 21:49:31	РК000000008	Поступление ...		Поставщик	Основной	46,00
✔			24.05.2015 22:16:37	РК000000005	Реализация т...	продажа, ком...	Поставщик	На продажу	12 967,02
✔			25.05.2015 11:35:29	РК000000011	Поступление ...		Поставщик	Основной	264,00

Рис.1.7. Журнал документов «документы контрагентов»

Отчёты – объекты, используемые для отображения сводной информации о движении и остатках ресурсов. Отчёт не меняет содержания информационной базы.

Обработки – объекты среды, используемые для выполнения некоторых вспомогательных действий. Отличие обработок от отчётов в том, что обработки может менять содержание

информационной базы и обработки обычно не имеет печатной формы.

Пользователь в режиме предприятия может создавать новые элементы справочников, новые документы, редактировать их, помечать на удаление, удалять.

Новые типы документов, новые типы справочников, константы, перечисления пользователь в среде выполнения создавать не может, они определяются прикладным решением.

Одним из принципов работы 1с Предприятия является принцип двух этапного удаления элементов среды -

- пометка на удаление.
- удаление.

При пометке на удаление элемент исключается из использования, его нельзя использовать для ссылок, в качестве значения атрибутов.

Удаление производится как отдельная операция, выполняемая в монопольном режиме (*Операции/Удаление объектов, помеченных на удаление*). Правами выполнять данную операцию владеет пользователь административного уровня. При удалении объекта среды выполнения предварительно выявляются его ссылки и связи (например, использование данного объекта в качестве значения атрибута другого объекта). Если на удаляемый объект есть ссылки из других, не удаленных объектов, то удаление будет запрещено.

Роль документов.

Документы отображают движение ресурсов различного типа. Документы можно:

* Создавать.
* Сохранять.
* Проводить.
* Отменять проведение документа и операции.
* Помечать на удаление.
* Удалять.

При создании документа определяется его содержание, но в базе оно не сохраняется.

При сохранении документ сохраняется в ИБ, но движение ресурсов не выполняется.

При проведении документа выполняется движение ресурсов, связанных с документом.

При отмене проведения, отменяются движения ресурсов, связанных с документом, документ при этом сохраняется в информационной базе.

В журнале документов (Журнал – перечень документов определенного типа) проведенные документы отмечаются зелёным флажком – v.

Задание начальных параметров учёта

Для начала работы с системой необходимо задать начальные параметры учёта. Это действие выполняется командой – *Сервис/НастройкаПараметровУчёта.*

Необходимо задать валюты ведения учёта (закладка «Валюты»).

Валюты необходимо задать для регламентированного учёта, то есть учёта в соответствии с законодательством, по которому производится налоговый учёт и управленческого учёта. Управленческий учёт ведётся в соответствии с внутренними задачами управления.

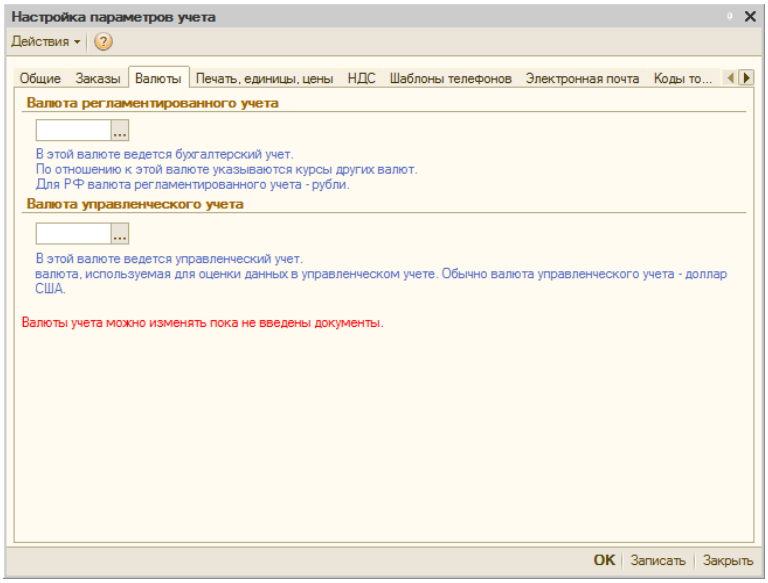


Рис.1.8. Настройка параметров учёта

Для задания валют учёта необходимо предварительно заполнить справочник «Валюты».

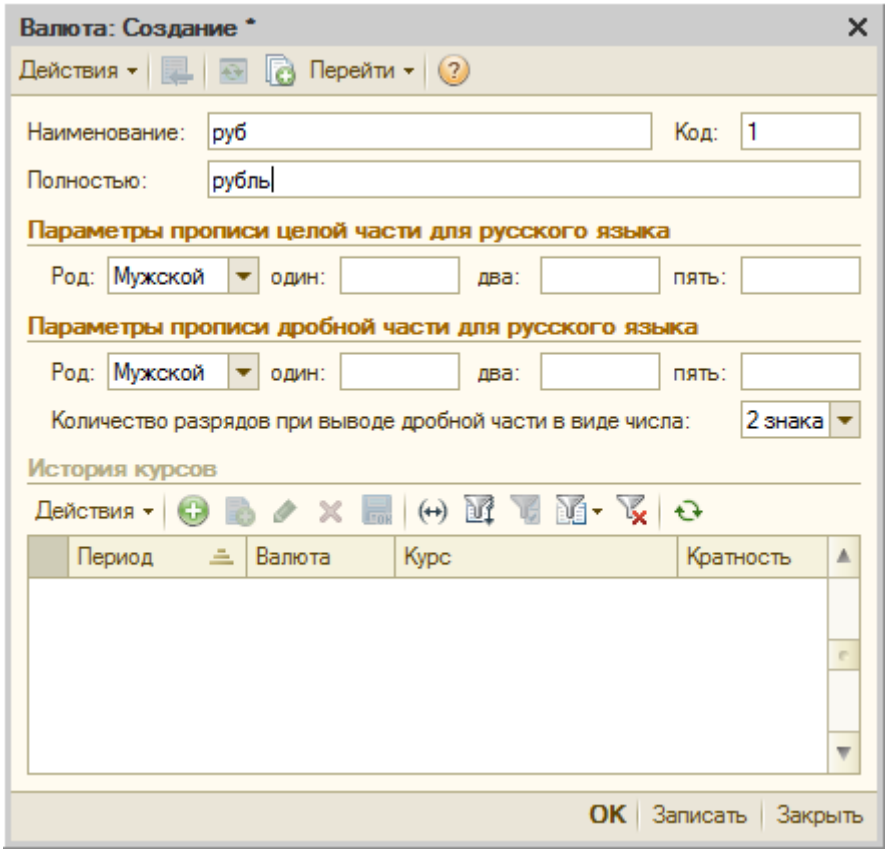


Рис.1.9. Заполнение справочника «Валюты»

Для обеспечения работы системы необходимо также задать префиксы штрих кодов. Хоть мы не будем работать со считывателями штрих кодов (а может и будем). Но задать их надо.

3. ПОРЯДОК ВЫПОЛНЕНИЯ РАБОТЫ

1. Выполнить установку 8.3 учебная версия. (если не установлена). Установка лежит в каталоге обмена на сервере.
2. Создать пустой каталог в доступном дисковом пространстве. (На съёмном носителе лучше не работать). Создать пустую информационную базу без шаблона. Настроить её на созданный каталог.
3. Выполнить загрузку файла конфигурации из файла 1Cv8.cf.

Файл лежит на сервере в каталоге учебного курса. Подкаталог CFUT10.32.

4. В режиме конфигурации задать два пользователя. Один с полными правами с полными интерфейсов, другой с ….
5. Войти в режим предприятие под пользователем с полными правами.

6. Задать значение основных справочников и констант необходимых для работы.

a. Константа «Валюты». Задайте рубли. (рис. 1.9).

7. Войти в настройки параметров учёта (Сервис…

a. Задать валюту учёта. Бухгалтерского и управленческого.

b. Задать дату ведения учёта НДС в соответствии с постановлением Правительства РФ от 26 декабря 2011 №1137 (Любая дата раньше текущей)

c. В закладке Коды товаров задать префиксы штрих кода для штучного товара и для других товаров.

8. Задайте для организации используемую учётную политику. Сервис/Настройка учёта/Учётная политика (Налоговый учёт).

9. Задайте данные об организации. Справочник Организации (Группа справочников – Предприятия) достаточно задать наименование организации и используемый для неё префикс. (Организаций может быть несколько, но достаточно одной).

10. Задайте необходимую номенклатуру товаров.

11. При создании единиц номенклатуры используется дополнительный справочник «Вид номенклатуры» Вид номенклатуры используется для более детального анализа деятельности предприятия. Примеры Видов номенклатуры – товар для продажи, товар для комплектации. Для создаваемых единиц видов номенклатуры необходимо выбрать их соотнесение с возможными Видами номенклатуры.

12. При создании единиц номенклатуры необходимо установить базовые единицы измерения. Для этого необходимо заполнить справочник единицы измерения. Задайте необходимые единицы измерения (штука, литр, кг.…).

13. Кроме базовой единицы измерения на закладке Единицы можно задать другие единицы измерения, используемые с данной единицей номенклатуры, например, ящик, пакет, упаковка с коэффициентами пересчёта. Использование других единиц облегчает работу с документами. Обычно внутренний учёт идёт в базовых единицах, документы поставки в других.

14. Задайте возможные типы цен номенклатуры. Справочники/Номенклатура/ТипыЦен. Должны быть заданы цена поступления, как базовая и розничная рассчитываемая на основании базовой по коэффициенту.

15. Создайте места хранения. Один склад тип – оптовый.

16. Создать контрагентов, с которыми предполагается взаимодействовать. Создайте одного поставщика, одного покупателя. У поставщика соответственно установите флажок «Покупатель». У поставщика «Поставщик». При создании контрагентов для договора, по которому ведётся взаиморасчёт с поставщиком необходимо установить тип расчёта «По договору в целом».

17. Создать документы для поступления (документ – поступление).

18. Создать документ продажи поступившей продукции (Документы/Продажи/Реализации товаров и услуг). (Для того, чтобы можно было установить цену в документе продажи для пользователя необходимо установить возможность корректировки цен – Сервис/Пользователи/Настройка дополнительных прав пользователей/Документы – Редактирование цен и скидок в документе.

19. Сделать отчёт Отчёты/Запасы/Ведомость по товарам на складах.

4. КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Какими методами может быть создана новая информационная база некоторого прикладного решения?

2. Что такое шаблон информационной базы? Для чего он используется где расположен?

3. Назовите основные объекты среды выполнения прикладного решения 1С Предприятия.

4. Какая процедура удаления экземпляров объектов принята в среде 1С Предприятие?

5. Какую роль играют документы в прикладном решении 1С Предприятие?

6. Какие действия можно выполнит с документом в среде выполнения?

7. Какие начальные параметры учёта должны быть заданы в новой информационной базе прикладного решения Управление Торговлей?

РАБОТА №4 ПРАКТИЧЕСКАЯ. ОБСЛУЖИВАНИЕ СИСТЕМЫ ТЕХНОЛОГИЧЕСКОЙ ПОДГОТОВКИ ПРОИЗВОДСТВ

1. ЦЕЛЬ И ЗАДАЧИ РАБОТЫ

Получение практических навыков разработки документов в различных офисных пакетах и сопоставления возможностей данных пакетов.

2. КРАТКИЕ ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

Наиболее удобным языком моделирования бизнес-процессов является IDEF0, предложенный более 40 лет назад Дугласом Россом (SoftTech, Inc.) и называвшийся первоначально SADT – Structured Analysis and Design Technique1. В начале 70-х годов XX века вооруженные силы США применили подмножество SADT, касающееся моделирования процессов, для реализации проектов в рамках программы ICAM (Integrated Computer-Aided Manufacturing). В дальнейшем это подмножество SADT было принято в качестве федерального стандарта США под наименованием IDEF0.

В IDEF0 система представляется как совокупность взаимодействующих работ или функций. Такая чисто функциональная ориентация является принципиальной – функции системы анализируются независимо от объектов, которыми они оперируют. Это позволяет более четко смоделировать логику и взаимодействие процессов организации.

Под моделью в IDEF0 понимают описание системы (текстовое и графическое), которое должно дать ответ на некоторые заранее определенные вопросы.

Моделируемая система рассматривается как произвольное подмножество. Произвольное потому, что, во-первых, мы сами умозрительно определяем, будет ли некий объект компонентом системы, или мы будем его рассматривать как внешнее воздействие, и, во-вторых, оно зависит от точки зрения на систему. Взаимодействие системы с окружающим миром описывается как вход (нечто, что перерабатывается системой), выход (результат деятельности системы), управление (стратегии и процедуры, под управлением которых

производится работа) и механизм (ресурсы, необходимые для проведения работы). Находясь под управлением, система преобразует входы в выходы, используя механизмы.

Процесс моделирования какой-либо системы в IDEF0 начинается с определения контекста, т. е. наиболее абстрактного уровня описания системы в целом. В контекст входит определение субъекта моделирования, цели и точки зрения на модель.

Под субъектом понимается сама система, при этом необходимо точно установить, что входит в систему, а что лежит за ее пределами, другими словами, мы должны определить, что мы будем в дальнейшем рассматривать как компоненты системы, а что как внешнее воздействие. На определение субъекта системы будет существенно влиять позиция, с которой рассматривается система, и цель моделирования – вопросы, на которые построенная модель должна дать ответ, другими словами, первоначально необходимо определить область моделирования. Описание области как системы в целом, так и ее компонентов является основой построения модели. Хотя предполагается, что в течение моделирования область может корректироваться, она должна быть в основном сформулирована изначально, поскольку именно область определяет направление моделирования и когда должна быть закончена модель. При формулировании области необходимо учитывать два компонента – широту и глубину. Широта подразумевает определение границ модели – мы определяем, что будет рассматриваться внутри системы, а что снаружи. Глубина определяет, на каком Уровне детализации модель является завершенной. При определении глубины системы необходимо не забывать об ограничениях времени: трудоемкость построения модели растет в геометрической прогрессии от глубины декомпозиции. После определения границ модели предполагается, что новые объекты не должны вноситься в моделируемую систему; поскольку все объекты модели взаимосвязаны, внесение нового объекта может быть не просто арифметической добавкой, но в состоянии изменить существующие взаимосвязи. Внесение таких изменений в готовую модель является, как правило, очень трудоемким процессом (так называемая проблема «плавающей области»).

Цель моделирования. Модель не может быть построена без четко сформулированной цели. Цель должна отвечать на следующие вопросы:

• Почему этот процесс должен быть замоделирован?
• Что должна показывать модель?
• Что может получить читатель?

Формулировка цели позволяет команде аналитиков сфокусировать усилия в нужном направлении. Примерами формулирования цели могут быть следующие утверждения: «Идентифицировать и определить текущие проблемы, сделать возможным анализ потенциальных улучшений», «Идентифицировать роли и ответственность служащих для написания должностных инструкций», «Описать функциональность предприятия с целью написания спецификаций информационной системы» и т. д.

Хотя при построении модели учитываются мнения различных людей, модель должна строиться с единой точки зрения. Точку зрения можно представить, как взгляд человека, который видит систему в нужном для моделирования аспекте. Точка зрения должна соответствовать цели моделирования. Очевидно, что описание работы предприятия с точки зрения финансиста и технолога будет выглядеть совершенно по-разному, поэтому в течение моделирования важно оставаться на выбранной точке зрения. Как правило, выбирается точка зрения человека, ответственного за моделируемую работу в целом.

Для описания логики взаимодействия информационных потоков более подходит IDEF3, называемая также WorkFlow Diagramming – методологией моделирования, использующая графическое описание информационных потоков, взаимоотношений между процессами обработки информации и объектов, являющихся частью этих процессов. Диаграммы WorkFlow могут быть использованы в моделировании бизнес-процессов для анализа завершенности процедур обработки информации. С их помощью можно описывать сценарии действий сотрудников организации, например, последовательность обработки заказа события, которые необходимо обработать за конечное время. Каждый сценарий сопровождается описанием процесса и может быть использован для документирования каждой функции.

IDEF3 – это метод, имеющий основной целью дать возможность аналитикам описать ситуацию, когда процессы выполняются в определенной последовательности, а также описать объекты, участвующие совместно в одном процессе.

Техника описания набора данных IDEF3 является частью структурного анализа. В отличие от некоторых методик описаний процессов, IDEF3 не ограничивает аналитика чрезмерно жесткими рамками синтаксиса, что может привести к созданию неполных или противоречивых моделей.

IDEF3 может быть также использован как метод создания процессов. IDEF3 дополняет IDEF0 и содержит все необходимое для построения моделей, которые в дальнейшем могут быть использованы для имитационного анализа.

Каждая работа в IDEF3 описывает какой-либо сценарий бизнес процесса и может являться составляющей другой работы. Поскольку сценарий описывает цель и рамки модели, важно, чтобы работы именовались отглагольным существительным, обозначающим процесс действия, или фразой, содержащей такое существительное.

Точка зрения на модель должна быть задокументирована. Обычно это точка зрения человека, ответственного за работу в целом. Также необходимо задокументировать цель модели – те вопросы, на которые призвана ответить модель.

В IDEF3 работы изображаются прямоугольниками с прямыми углами и имеют имя, выраженное отглагольным существительным, обозначающим процесс действия, одиночным или в составе фразы, и номер (идентификатор); другое имя существительное в составе той же фразы обычно отображает основной выход (результат) работы (например, *«Изготовление изделия»*). Часто имя существительное в имени работы меняется в процессе моделирования, поскольку модель может уточняться и редактироваться. Идентификатор работы присваивается при создании и не меняется никогда. Даже если работа будет удалена, ее идентификатор не будет вновь использоваться для других работ. Обычно номер работы состоит из номера родительской работы и порядкового номера на текущей диаграмме.

Связи. Связи показывают взаимоотношения работ. Все связи в IDEF3 однонаправленны и могут быть направлены куда угодно, но обычно диаграммы IDEF3 стараются построить так, чтобы связи были направлены слева направо. В IDEF3 различают три типа стрелок, изображающих связи, стиль которых устанавливается во вкладке Style диалога Arrow Properties (пункт контекстного меню Style).

Старшая (Precedence) стрелка – сплошная линия, связывающая единицы работ (UOW). Рисуется слева направо или сверху вниз.

Показывает, что работа-источник должна закончиться прежде, чем работа-цель начнется.

Стрелка отношения (Relational Link) – пунктирная линия, использующаяся для изображения связей между единицами работ (UOW), а также между единицами работ и объектами ссылок.

Потоки объектов (Object Flow) – стрелка с двумя наконечниками, применяется для описания того факта, что объект используется в двух или более единицах работы, например, когда объект порождается в одной работе и используется в другой.

Старшая связь и поток объектов. Старшая связь показывает, что работа-источник заканчивается ранее, чем начинается работа-цель. Часто результатом работы-источника становится объект, необходимый для запуска работы-цели. В этом случае стрелку, обозначающую объект, изображают с двойным наконечником. Имя стрелки должно ясно идентифицировать отображаемый объект. Поток объектов имеет ту же семантику, что и старшая стрелка.

Отношение показывает, что стрелка является альтернативой старшей стрелке или потоку объектов в смысле задания последовательности выполнения работ – работа-источник не обязательно должна закончиться прежде, чем работа-цель начнется. Более того, работа-цель может закончиться прежде, чем закончится работа-источник.

Перекрестки (Junction). Окончание одной работы может служить сигналом к началу нескольких работ, или же одна работа для своего запуска может ожидать окончания нескольких работ. Перекрестки используются для отображения логики взаимодействия стрелок при слиянии и разветвлении или для отображения множества событий, которые могут или должны завершены перед началом следующей работы. Различают перекрестки слияния (Fan-in Junction) и разветвления (Fan-out Junction) стрелок. Перекресток не может использоваться одновременно для слияния и для ветвления. Для внесения перекрестка служит кнопка  (добавить на диаграмму перекресток – Junction) в палитре инструментов. В диалоге Junction Type Editor необходимо указать тип перекрестка.

Все перекрестки на диаграмме нумеруются, каждый номер имеет префикс J. Можно редактировать свойства перекрестка при помощи диалога Junction Properties (вызывается из контекстного

меню). В отличие от IDEF0 и DFD в IDEF3 стрелки могут сливаться и разветвляться только через перекрестки.

Объект ссылки. Объект ссылки в IDEF3 выражает некую идею, концепцию или данные, которые нельзя связать со стрелкой, перекрестком или работой. Для внесения объекта ссылки служит кнопка  (добавить в диаграмму объект ссылки – Referent) в палитре инструментов.

Объект ссылки изображается в виде прямоугольника, похожего на прямоугольник работы. Имя объекта ссылки задается в диалоге Referent Properties (пункт контекстного меню Name), в качестве имени можно использовать имя какой-либо стрелки с других диаграмм или имя сущности из модели данных. Объекты ссылки должны быть связаны с единицами работ или перекрестками пунктирными линиями. Официальная спецификация IDEF3 различает три стиля объектов ссылок – безусловные (unconditional), синхронные (synchronous) и асинхронные (asynchronous). BPwin поддерживает только безусловные объекты ссылок. Синхронные и асинхронные объекты ссылок, используемые в диаграммах переходов состояний объектов, не поддерживаются.

Спецификация объекта. Есть возможность детально специфицировать объекты диаграммы. Можно ввести имя и класс объекта, указать устойчивость его существования (persistence) и возможность наличия у него нескольких экземпляров. *Именование объекта.* Имя каждого присутствующего на диаграмме объекта можно задать непосредственно на диаграмме или в окне его спецификации. Дать название объекту можно следующим образом. Щелкните правой кнопкой мыши на объекте на диаграмме последовательности или диаграмме кооперации. В появившемся меню выберите пункт Open Specification (Открыть спецификацию). Введите имя объекта в поле Name (Имя). Каждый объект на диаграмме должен иметь уникальное имя. В дальнейшем с помощью этого поля можно изменить имя объекта. Или выделите объект на диаграмме последовательности или диаграмме кооперации. Нажмите правую кнопку мыши так, чтобы курсор был виден на объекте. Введите имя объекта. Если объект уже был соотнесен с классом, описание, введенное в окне документирования, будет добавлено и к классу. В противном случае оно будет связано только с объектом. Описание объекта не повлияет на генерацию кода, описание класса будет присутствовать и в ко-

де. Описание объекта можно просмотреть в отчете, генерируемом командой File> Print Specifications (Файл > Печать спецификаций) меню модели. По умолчанию объекту назначается класс Unspecified (не определен).

При назначении объекту класса можно либо указать уже существующий класс модели, либо создать новый класс. К моменту генерации кода все объекты должны быть соотнесены с классами. Для соотнесения объекта с существующим классом щелкните правой кнопкой мыши на диаграмме последовательности или диаграмме кооперации. В появившемся меню выберите пункт Open Specification (Открыть спецификацию). В раскрывающемся списке классов выберите имя класса или введите его с клавиатуры. После соотнесения класса с объектом название класса появится на диаграмме за именем объекта и двоеточием. Или выделите класс в логическом представлении браузера. Перетащите класс из браузера на объект диаграммы. После соотнесения класса с объектом название класса появится на диаграмме за именем объекта и двоеточием. Если удалить класс, с которым соотнесен объект, имя класса сохранится на диаграмме рядом с объектом, но будет заключено в скобки. Разорвать соотнесение объекта с классом можно следующим образом. Щелкните правой кнопкой мыши на объекте на диаграмме последовательности или диаграмме кооперации. В появившемся меню выберите пункт Open Specification (Открыть спецификацию). В раскрывающемся списке классов выберите пункт (Unspecified) (Не определен). Если нужно создать для объекта новый класс, щелкните правой кнопкой мыши на объекте на диаграмме последовательности или диаграмме кооперации. В появившемся меню выберите пункт Open Specification (Открыть спецификацию). В раскрывающемся списке классов выберите пункт New (Создать). Перед вами появится окно спецификации для нового класса. Если необходимо убедиться, что все объекты соотнесены с классами, выберите в меню модели пункт Report> Show Unresolved Objects (Отчет> Показать свободные объекты). Появится список всех объектов, которые еще не были соотнесены с классами. Для того чтобы на диаграмме выводилось только имя объекта, щелкните правой кнопкой мыши на объекте на диаграмме последовательности или диаграмме кооперации. В появившемся меню выберите пункт Open Specification (Открыть спецификацию). Введите имя объекта в поле Name. В рас-

крывающемся списке классов выберите пункт (Unspecified) (Не определен). Если нужно показать на диаграмме имя объекта и имя его класса, щелкните правой кнопкой мыши на объекте на диаграмме последовательности или диаграмме кооперации. В появившемся меню выберите пункт Open Specification (Открыть спецификацию). В раскрывающемся списке классов выберите имя класса или введите его с клавиатуры. Если необходимо работать только с именем класса, щелкните правой кнопкой мыши на объекте на диаграмме последовательности или диаграмме кооперации. В появившемся меню выберите пункт Open Specification (Открыть спецификацию). Удалите имя объекта из поля его имени. Теперь на диаграмме для объекта будет показано только имя соответствующего класса. Как и прежде, перед ним будет двоеточие.

Определение устойчивости объекта. Для определения устойчивости объекта щелкните правой кнопкой мыши на объекте на диаграмме последовательности или диаграмме кооперации. В появившемся меню выберите пункт Open Specification (Открыть спецификацию). Установите переключатель Persistence в значение Persistent, Static или Transient. Для работы с множественными экземплярами объекта щелкните правой кнопкой мыши на объекте на диаграмме последовательности или диаграмме кооперации. В появившемся меню выберите пункт Open Specification (Открыть спецификацию). Установите или сбросьте флажок Multiple Instances (Множественные экземпляры). На диаграмме кооперации для этого объекта будет показана соответствующая пиктограмма (значок множественного или одиночного объекта). На диаграмме последовательности всегда выводится пиктограмма одиночного объекта.

3. ПОРЯДОК ВЫПОЛНЕНИЯ РАБОТЫ

1. Определить содержание базового бизнес-процесса и, если необходимо, связанных с ним БП, отобразить содержание БП в виде диаграмм деятельности.

a. Выявить объекты, участвующие в выполнении БП.

b. Выявить бизнес-правила, выполнения отдельных элементов БП.

c. Выделить автоматизируемые элементы БП.

2. Для объектов, выявленных в бизнес-процессах сформировать классы.

a. Создать классы объектов предметной в соответствующем пакете обозревателя модели.

b. Задать для объектов предметной области, участвующих в выполнении рассматриваемых бизнес процессов (соответствующие диаграммы деятельности) классификаторы из состава сформированных классов.

c. Построить диаграмму классов предметной области.

3. Выявить типы данных предметной области.

4. Отобразить выявленные автоматизируемые элементы БП на отдельной диаграмме в соответствующим пакете обозревателя модели.

5. Создать отчёт по выполненной работе.

Заданием для работы является отображение содержания одного из процессов деятельности предприятия, рассмотренного в прошлой практической работе.

4. КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Какие элементы и диаграммы используются для отображения содержания бизнес-процессов?
2. Каким образом выявляются классы объектов предметной области?
3. Каким образом выявляются элементы, подлежащие автоматизации? Какими элементами и на каких диаграммах они отображаются?

РАБОТА №5 ПРАКТИЧЕСКАЯ. РАЗРАБОТКА ТЕХНИЧЕСКОГО ЗАДАНИЯ НА СОПРОВОЖДЕНИЕ СИСТЕМЫ ОПЕРАТИВНОГО УЧЁТА ДВИЖЕНИЯ ТОВАРОВ

1. ЦЕЛЬ И ЗАДАЧИ РАБОТЫ

Целью работы является получение практических навыков формирования технического задания. В связи с этим задачами работы является: формулировка темы работы; уточнение целей разработки системы; формулировка требований к системе;

2. КРАТКИЕ ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

Техническое задание (ТЗ) является основным документом, в котором определяются требования к автоматизированной системе и порядок её создание. В нём закрепляется итоги начальной фазы разработки системы, в результате которой должны быть определены границы системы и принято решение о разработки системы. Техническое задание составляется совместно с заказчиком системы и является главным документом, на основе которого производится приёмка системы. Содержание технического задания определено в ГОСТ 34.602-89.

Согласно ГОСТ ТЗ должно содержать следующие подразделы, которые могут быть на подразделы.

1. Общие сведения.
2. Назначение и цели создания системы.
3. Характеристика объектов автоматизации.
4. Требования к системе.
5. Состав и содержание работ по созданию системы.
6. Требования к составу и содержанию работ по подготовке объекта автоматизации к вводу системы в действие.
7. Требования к документированию.
8. Источники разработки.
9. В ТЗ могут включаться приложения.

В зависимости от вида, назначения, специфических особенностей объекта автоматизации и условий функционирования системы допускается оформлять отдельные разделы ТЗ в виде приложений,

вводить дополнительные разделы, исключать или дополнять. Для ТЗ ГОСТ предлагает пример титульного листа (см. прил.1)

Раздел 1. Общие сведения включает:

Общие сведения о системе

1.1. Полное наименование системы и её условное обозначение.

1.2. Указание на заказчика и разработчика системы

1.3. Перечень документов на основе которых производится разработка.

1.4. Плановые сроки начала и окончания работа по созданию системы.

1.5. Сведения об источниках и порядке финансирования работ.

1.6. Порядок оформления и предъявления заказчику результатов работ по созданию системы.

Полное наименование системы должно указывать на объекты автоматизации, то есть на процессы, которые автоматизирует система, то есть связано с темой автоматизации. Например, «Система формирования заказов на доставку», однако от данного принципа возможны отклонения. Условное обозначение должно облегчать описание системе, быть легко произносимым, и может нести в себе рекламные функции. Например, система «Быстрый Документ», «Документ доставки». Условное обозначение может быть составлено в виде аббревиатуры полного наименования.

В *перечень документов*, на основании которых создается система, включается указание на договора, приказы, соглашения, на основе которых производится разработки.

Порядок оформления и предъявления заказчику результатов работ по созданию системы подразумевает возможность предоставления предварительных версий, компонентов системы до окончания всех работ.

Раздел 2. Назначение и цели создания системы

Назначение системы подразумевает общее указание на процесс, автоматизация которого обеспечивается системой.

Формулировка цели создания системы должна характеризовать показатели работы автоматизируемого процесса, на которых повлияет использование системы. Цель должна указывать на конкретный показатель и величину его изменения.

Раздел 3. Характеристика объекта автоматизации должна включать описание процесса, который будет автоматизировать раз-

рабатываемая система. Общая характеристика процесса, его входы, выходы, особенности выполнения, влияние на деятельность предприятия в целом.

Раздел 4. Требования к системе.

Раздел включает описание требований, характеризующих систему как единое целое и характеристику отдельных функций.

4.1. Требования к системе в целом содержит следующие подпункты.

Требования к структуре и функционированию системы. В данном подразделе должна быть приведена общая характеристика структуры разрабатываемой системы. Тип архитектуры, используемое системное программное обеспечение, приведена общая характеристика основных программных и аппаратных компонент и особенности связей между ними.

В подразделе *«Требования к численности и квалификации персонала системы и режиму его работ»* необходимо привести численность, квалификация и режим работы персонала, который должен обеспечивать работоспособность системы.

В подразделах *«Требования к надежности» и «Требования безопасности»* необходимо привести конкретные показатели надёжности и требования безопасности, которые должна обеспечивать система.

«Требования к патентной чистоте» должны оговариваться лицензии, указываться тип свободного программного обеспечения, используемый при разработке.

Пункт *«Требования по стандартизации и унификации»)* должен указывать на стандарты, используемые при разработке. И используемые принципы унификации проектных и программных решений.

Дополнительные требования включают: требования к оснащению системы устройствами для обучения персонала (тренажерами, другими устройствами аналогичного назначения) и документацией на них; требования к сервисной аппаратуре, стендам для проверки элементов системы; требования к системе, связанные с особыми условиями эксплуатации; специальные требования по усмотрению разработчика или заказчика системы.

4.2. Требование к функциям (задачам).

В подразделе **«Требование к функциям (задачам)»,** выполняемым системой, приводят: по каждой подсистеме перечень функций (в том числе обеспечивающих взаимодействие частей системы), подлежащих автоматизации.

При создании системы в две или более очереди (итерации) – перечень функциональных подсистем, отдельных функций, вводимых в действие в каждой итерации. Для каждой функции приводится её описание, временной регламент реализации, требования к качеству реализации функции, к форме представления выходной информации, характеристики необходимой точности и времени выполнения, достоверности выдачи результатов, перечень и критерии отказов для каждой функции, по которой задаются требования по надежности. *Описание функций при согласовании с заказчиком можно привести в виде моделей UML или другой нотации.*

4.3. Требования к видам обеспечения.

В подразделе **«Требования к видам обеспечения»** в зависимости от вида системы приводят требования к математическому, информационному, лингвистическому, программному, техническому, метрологическому, организационному, методическому и другим видам обеспечения системы.

Для математического обеспечения системы приводят требования к составу, области применения (ограничения) и способам, использования в системе математических методов и моделей, типовых алгоритмов и алгоритмов, подлежащих разработке. Например, математические методы и алгоритмы, используемые для шифрования/дешифрования данных.

Для информационного обеспечения системы приводят требования к составу, структуре и способам организации данных систем, к информационному обмену между компонентами системы, к информационной совместимости со смежными системами, по применению систем управления базами данных.

Для лингвистического обеспечения системы приводят требования к применению в системе языков программирования высокого уровня, языков взаимодействия пользователей и технических средств системы, а также требования к кодированию и декодированию данных, к языкам ввода-вывода данных, языкам манипулиро-

вания данными, средствам описания предметной области (объекта автоматизации), к способам организации диалога.

Для программного обеспечения системы приводят перечень покупных программных средств, а также требования, к независимости программных средств от используемых аппаратных средств и операционной среды, к качеству программных средств, а также к способам его обеспечения и контроля.

Для технического обеспечения системы приводят требования к видам технических средств, в том числе к видам комплексов технических средств, программно-технических комплексов и других комплектующих изделий, допустимых к использованию в системе, к функциональным, конструктивным и эксплуатационным характеристикам средств технического обеспечения системы.

Для организационного обеспечения приводят требования к структуре и функциям подразделений, участвующих в функционировании системы или обеспечивающих эксплуатацию к организации функционирования системы и порядку взаимодействия персонала системы и персонала объекта автоматизации к защите от ошибочных действий персонала системы.

4.4. Требования к информационному обмену между компонентами системы.

Входящие в состав АС Кадры подсистемы в процессе функционирования должны обмен информацией на основе открытых форматов обмена данными, используя для этого входящие в их состав модули информационного взаимодействия. Форматы данных будут разработаны и утверждены на этапе технического проектирования.

4.5. Требования информационной совместимости со смежными системами.

Состав данных для осуществления информационного обмена по каждой смежной системе должен быть определен Разработчиком на стадии «Проектирование. Разработка эскизного проекта. Разработка технического проекта» совместно с полномочными представителями Заказчика. Система не должна быть закрытой для смежных систем и должна поддерживать возможность экспорта данных в смежные системы через интерфейсные таблицы или файлы данных. Система должна обеспечить возможность загрузки данных, получаемых от смежной системы.

4.6. Требования по использованию общесоюзных и зарегистрированных республиканских, отраслевых классификаторов, унифицированных документов и классификаторов, действующих на данном предприятии.

При разработке системы должны быть применены классификаторы, разработанные для кодирования информации (документ «инструкция по кодированию информации для «информационной системы наркологической службы кемеровской области»)

4.7. Требование к программному обеспечению.

Например, «Система должна работать в среде операционной системы Windows XP sp3 и более поздними версиями» .

Для разработки системы не должно использоваться дополнительное покупное программное обеспечение.

Раздел 5. Состав и содержание работ по созданию (развитию) системы.

Раздел «Состав и содержание работ по созданию (развитию) системы» должен содержать перечень стадий и этапов работ по созданию системы в соответствии с ГОСТ 24.601, сроки их выполнения, перечень организаций – исполнителей работ, ссылки на документы, подтверждающие согласие этих организаций на участие в создании системы, или запись, определяющую ответственного (заказчик или разработчик) за проведение этих работ. В данном разделе также приводят:

1) перечень документов, по ГОСТ 34.201-89, предъявляемых по окончании соответствующих стадий и этапов работ;

2) вид и порядок проведения экспертизы технической документации (стадия, этап, объем проверяемой документации, организация-эксперт);

3) программу работ, направленных на обеспечение требуемого уровня надежности разрабатываемой системы (при необходимости);

4) перечень работ по метрологическому обеспечению на всех стадиях создания системы с указанием их сроков выполнения и организаций-исполнителей (при необходимости).

Раздел 6. Порядок контроля и приемки системы

Указывают:

1) виды, состав, объем и методы испытаний системы и ее составных частей (виды испытаний в соответствии с действующими нормами, распространяющимися на разрабатываемую систему);

2) общие требования к приемке работ по стадиям (перечень участвующих предприятий и организаций, место и сроки проведения), порядок согласования и утверждения приемочной документации;

3) статус приемочной комиссии (государственная, межведомственная, ведомственная).

Раздел 7. Требования к составу и содержанию работ по подготовке объекта автоматизации к вводу системы в действие

Необходимо привести перечень основных мероприятий и их исполнителей, которые следует выполнить при подготовке объекта автоматизации. В перечень основных мероприятий включают:

1) приведение поступающей в систему информации (в соответствии с требованиями к информационному и лингвистическому обеспечению) к виду, пригодному для обработки с помощью ЭВМ;

2) изменения, которые необходимо осуществить в объекте автоматизации;

3) создание условий функционирования объекта автоматизации, при которых гарантируется соответствие создаваемой системы требованиям, содержащимся в ТЗ;

4) создание необходимых для функционирования системы подразделений и служб;

5) сроки и порядок комплектования штатов и обучения персонала.

Раздел 8. Требования к документированию

Приводят:

1) согласованный разработчиком и Заказчиком системы перечень подлежащих разработке комплектов и видов документов, соответствующих требованиям ГОСТ 34.201-89 и НТД отрасли заказчика; перечень документов, выпускаемых на машинных носителях; требования к микрофильмированию документации;

2) требования по документированию комплектующих элементов межотраслевого применения в соответствии с требованиями ЕСКД и ЕСПД;

3) при отсутствии государственных стандартов, определяющих требования к документированию элементов системы, дополнительно включают требования к составу и содержанию таких документов.

Раздел 9. Источники разработки

В этом разделе должны быть перечислены документы и информационные материалы (технико-экономическое обоснование, отчеты о законченных научно-исследовательских работах, информационные материалы на отечественные, зарубежные системы-аналоги и др.), на основании которых разрабатывалось ТЗ и которые должны быть использованы при создании системы.

3. ПОРЯДОК ВЫПОЛНЕНИЯ РАБОТЫ.

1. Уточнить постановку задачи с преподавателем. Предварительно определить требования к системе.

2. В общей модели разработки создать пакет верхнего уровня (view) «формирование требований», в котором будут отображаться работы по формированию требований к системе. В пакете верхнего уровня создать подпакеты:

2.1. состав требований;

2.2. диаграммы вариантов использования;

2.3. реализация требований;

2.4. требования к пользовательскому интерфейсу.

3. Создать в подпакете «состав требований» пользовательскую диаграмму «требования к системе». На диаграмме с помощью элементов «requirement» (группа «castom» на панели инструментов) отобразить функциональные требования к системе. В свойствах каждого элемента описать, что подразумевается под данным функциональным требованием. При необходимости, разделит функциональные требования на подсистемы.

4. Разработка диаграмм вариантов использования.

4.1. Создать в подпакете «диаграммы вариантов использования» диаграммы вариантов использования (use case diagramc) для каждой выделенной подсистемы.

4.2. Вынести на созданную диаграмму вариантов использования из обозревателя модели требования, определённые в пункте 2.

4.3. Определить для каждого требования сценарии его реализующие. Отобразить на диаграмме сценарии с помощью элемента «use case». Соединить элементы «use case» соответствующими им требованиями отношениями зависимость со стереотипом «поддер-

живает». При необходимости структурировать варианты использования, задав между ними отношения включения и расширения.

4.4. Определить внешние сущности (пользователи и внешние системы), участвующие в функционировании системы. Отобразить внешние сущности с помощью элемента «actor».

4.5. Соединить внешние сущности с элементами «use case» отношением ассоциации. Отношение должно отображать возможность актанта запускать на выполнение соответствующий сценарий или получать от него значимый результат.

5. Определение содержание сценариев выполнения функций.

5.1. Декомпозировать сценарии выполнения функций в виде диаграмм деятельности.

5.1.1. Выявить объекты сущности и граничные объекты, участвующие в выполнении сценариев.

5.1.2. Описать объекты сценариев.

6. На основе выявленных в сценариях граничных объектов сформировать создать проект экранных форм и описать требования к пользовательскому интерфейсу.

7. Сформулировать предварительные требования к хранилищу данных.

Заданием для работы является формулировка требований к системе, автоматизирующие процессы, выявленные в прошлой работе.

4. КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Что такое техническое задание?

2. Какие подразделы выделяются в техническом задании?

3. Что включается в первый и второй подразделы технического задания? Как формулируется назначение и цель системы?

4. Что является объектом автоматизации?

5. Какие виды требований выделяются в техническом задании?

6. Где в техническом задании отображаются функциональные требования к системе?

7. Какие виды обеспечений выделяются у информационной системы в техническом задании?

РАБОТА №6 ПРАКТИЧЕСКАЯ. ОБСЛУЖИВАНИЕ СИСТЕМЫ ВИДЕОНАБЛЮДЕНИЯ

1. ЦЕЛЬ И ЗАДАЧИ РАБОТЫ

Целью работы является освоение содержание процесса проектирования информационной системы. В связи с этим задачами работы является:

- Определение классов объектов, реализующих функциональность системы.
- Определение взаимодействий объектов, реализующих функциональность системы.
- Определение методов классов объектов.
- Выявление объектов, требующих длительное хранение.

2. КРАТКИЕ ТЕОРЕТИЧЕСКИЕ ПОЛОЖЕНИЯ

Все вычислительные системы (ВС) по категориям решаемых ими задач и (в некоторой степени) по характеру взаимодействия с пользователем можно разделить на две большие группы:

1) универсальные ВС (высокопроизводительные системы, рабочие станции);

2) информационно-управляющие системы.

Информационно-управляющие системы (ИУС) – это вычислительные системы, в которых:

– есть взаимодействие с объектом контроля или управления;

– присутствует в виде ограничения временной фактор (время выступает в качестве одного из ограничений, определяющих организацию вычислительного процесса).

Универсальные вычислительные системы, в отличие от ИУС, предназначены для решения задач, которые перед ними ставит некий оператор (непосредственно сидящий перед дисплеем или работающий за удаленным терминалом). Система может обрабатывать задания, в том числе и в пакетном режиме. Оператору при этом не обязательно получить реакцию на решение задачи в какой-то жестко ограниченный временной период. Универсальные системы устроены таким образом, чтобы они хорошо могли решать максимально широкий круг задач.

Те подсистемы универсальных ВС, которые отвечают за выполнение задач реального времени (РВ), строятся по принципам ИУС.

Одной из основных черт ИУС является то, что это всегда специализированные системы. Они проектируются для решения конкретной задачи независимо от того, что будет лежать в основе их реализации: персональный компьютер (ПК), специализированный микроконтроллер, вычислительная сеть или что-то иное. Разработчик ИУС вынужден затрагивать разные уровни вычислительной системы: от программной надстройки до аппаратуры.

ИУС рассматриваются как область вычислительной техники (ВТ), в которой системы можно поделить на две большие категории:

– встроенные вычислительные системы;

– распределенные информационно-управляющие системы.

Определение систем реального времени:

1. Система называется системой реального времени (СРВ), если правильность ее функционирования зависит не только от логической корректности вычислений, но и от времени, за которое эти вычисления производятся.

2. Система работает в реальном времени, если ее быстродействие адекватно скорости протекания физических процессов на объектах контроля или управления. Система управления должна собрать данные, произвести их обработку в соответствии с заданными алгоритмами и выдать управляющее воздействие за такой промежуток времени, который обеспечивает успешное выполнение поставленных перед системой задач.

Основные требования к СРВ:

– требования по времени;

– возможность параллельного выполнения нескольких задач;

– предсказуемость;

– важно максимальное время отклика на событие, а не среднее;

– особые требования в вопросах безопасности;

– возможность безотказной работы в течение длительного периода времени.

СРВ должны реагировать на периодические и непериодические типы внутренних и внешних событий. Принадлежность системы к классу СРВ никак не связана с ее быстродействием. Исходные требования к времени реакции системы и другим временным пара-

метрам определяются или техническим заданием на систему, или просто логикой ее функционирования. Быстродействие СРВ должно быть тем больше, чем больше скорость протекания процессов на объекте контроля и управления. Чтобы определить необходимое быстродействие для систем, имеющих дело со стационарными процессами, часто используют теорему Котельникова, из которой следует, что частота дискретизации сигналов должна быть как минимум в 2 раза выше граничной частоты их спектра. При работе с широкополосными по своей природе переходными процессами часто применяют быстродействующие АЦП с буферной памятью, куда с необходимой скоростью записывается реализация сигнала, которая затем анализируется и/или регистрируется вычислительной системой. При этом требуется закончить всю необходимую обработку до следующего переходного процесса, иначе информация будет потеряна. Подобные системы иногда называют системами квазиреального времени.

Классификация информационно-управляющих систем реального времени

Различают системы *жесткого* и *мягкого* реального времени:

1. Системой жесткого реального времени называется система, где неспособность обеспечить реакцию на какие-либо события в заданное время является отказом и ведет к невозможности решения поставленной задачи. Время реакции в системах жесткого реального времени должно быть минимальным. Большинство систем жесткого реального времени являются системами контроля и управления. Такие СРВ сложны в реализации, так как для них предъявляются особые требования в вопросах безопасности.

2. Точного определения для систем мягкого реального времени не существует, поэтому отнесем к ИУС мягкого реального времени системы, не попадающие в категорию жестких. Так как система мягкого реального времени может не успевать решать поставленные задачи в заданное время, возникает проблема определения критериев успешности (нормальности) ее функционирования. Вопрос этот совсем не простой, так как в зависимости от функций системы это может быть максимальная задержка в выполнении каких-либо операций, средняя своевременность обработки событий и т. п. Более того, эти критерии влияют на то, какой алгоритм планирования задач является оптимальным.

Также СРВ можно разделить на системы, *специализированные* и *универсальные*:

1. Специализированной СРВ называется система, где конкретные временные требования априори определены. Такая система должна быть специально спроектирована для удовлетворения этих требований.

2. Универсальная СРВ должна уметь выполнять произвольные, заранее не определенные временные задачи без применения специальной техники. Разработка таких систем является самой сложной задачей, хотя обычно требования, предъявляемые к таким системам, мягче, чем требования для специализированных систем.

Для описания классов необходимо отметить, как выделяются классы, подлежащие описанию. По этому вопросу в литературе приводится множество соображений, советов, рекомендаций и даже принципов. Само разнообразие подходов свидетельствует о том, что среди них нет универсального и применимого во всех случаях. Можно выделить три приема выделения классов, самых простых (можно сказать, даже примитивных), а потому, по нашему мнению, самых действенных и широко применимых:

- словарь предметной области;
- реализация вариантов использования;
- образцы проектирования.

Словарь предметной области – это набор основных понятий (сущностей) данной предметной области

Рассмотрите внимательно текст технического задания (или иного документа, лежащего в основе проекта) и выделите в содержательной части имена существительные – все они являются кандидатами на то, чтобы быть названиями классов (или атрибутов классов) проектируемой системы. Разумеется, после этой простой операции к полученному списку нужно применить фильтр здравого смысла и опыта, отсекая ненужное.

Рассмотрим теперь, как выявляются классы в процессе реализации вариантов использования. Если при реализации вариантов использования применяются диаграммы взаимодействия, то в этом процессе в качестве побочного эффекта выделяются некоторые классы непосредственно, поскольку на диаграммах кооперации и последовательности основными сущностями являются объекты, которые по необходимости нужно отнести к определенным классам.

Использование диаграмм деятельности также может подсказать, какие классы нужно определить в системе, особенно если на диаграмме деятельности указывается поток объектов. Однако, если сценарии вариантов использования описываются на псевдокоде, то выделить классы значительно труднее. Фактически, если варианты использования реализуются на псевдокоде или диаграммами деятельности вне связи с объектами, то выявление объектной структуры системы просто откладывается «на потом». Иногда это может быть вполне оправдано – например, архитектор, моделирующий систему, прежде чем начать проектирование основной структуры классов, хочет более глубоко вникнуть в логику бизнес-процессов незнакомой ему предметной области.

Диаграмма классов является основным средством моделирования структуры UML. Класс в UML является основной структурной единицей. Диаграммы классов наиболее информационно насыщены по сравнению с другими типами канонических диаграмм UML, инструменты генерируют код в основном по описанию классов, структура классов точнее всего соответствует окончательной структуре кода приложения.

На диаграммах классов в качестве сущностей применяются, прежде всего, классы, как в своей наиболее общей форме, так и в форме многочисленных стереотипов и частных случаев: интерфейсы, типы данных, процессы и др. Кроме того, в диаграмме классов могут использоваться (как и везде) пакеты и примечания. Сущности на диаграммах классов связываются главным образом отношениями ассоциации (в том числе агрегирования и композиции) и обобщения. Отношения зависимости и реализации на диаграммах классов применяются реже.

Класс – один из самых «богатых» элементов моделирования UML. Описание класса может включать множество различных элементов, и чтобы они не путались, в языке предусмотрено группирование элементов описания класса по *разделам.* Стандартных разделов три:

- раздел имени – наряду с обязательным именем может содержать также стереотип, кратность и список свойств;
- раздел атрибутов – содержит список описаний атрибутов класса;

- раздел операций – содержит список описаний операций класса.

Как и все основные сущности UML, класс обязательно имеет имя, а стало быть раздел имени не может быть опущен. Прочие разделы могут быть пустыми. Наряду со стандартными разделами, описание класса может содержать и произвольное количество дополнительных разделов. Семантически дополнительные разделы эквиваленты примечаниям. Если инструмент умеет что-то делать с информацией в дополнительных разделах, пусть делает. В любом случае инструмент обязан сохранить эту информацию в модели.

Класс, а также отдельные элементы его описания могут иметь произвольные заданные пользователем ограничения и именованные значения.

Кратность класса задается по общим правилам. Наиболее распространенный случай неограниченной кратности (т. е. класс может иметь произвольное значение экземпляров) подразумевается по умолчанию и никак не отражается на диаграмме классов. Другой распространенный случай – нулевая кратность – обычно представляется с помощью стандартного стереотипа «utility.

Связь между объектами (экземплярами классов) в программе может быть организована самыми разными способами. Например, в объекте одного класса может хранится указатель на объект другого класса. Другой вариант: объект одного класса является контейнером для объектов другого класса. Связь не обязательно является непосредственно хранимым физическим адресом. Этот адрес может динамически вычисляться во время выполнения программы на основании другой информации. Например, если объекты представлены как записи в таблице базы данных, то связь означает, в записи одного объекта имеется поле, значением которого является первичный ключ записи другого объекта (из другой таблицы).

Диаграммы классов содержат множество деталей. Для практически значимых систем диаграммы классов в конечном итоге получаются довольно сложными. Пытаться прорисовать сложную диаграмму классов сразу «на всю глубину» нерационально – слишком велик риск «утонуть» в деталях. Удачная модель структуры сложной системы создается за несколько (может быть даже за несколько десятков) итераций, в которых моделирование структуры перемежается моделированием поведения. Удобнее:

- Описывать структуру удобнее параллельно с описанием поведения. Каждая итерация должна быть небольшим уточнением как структуры, так и поведения.
- Не обязательно включать в модель все классы сразу. На первых итерациях достаточно идентифицировать очень небольшую (10%) долю всех классов системы.
- Не обязательно определять все свойства класса сразу. Начните с имени – операции и атрибуты постепенно выявятся в процессе моделирования поведения.
- Не обязательно показывать на диаграмме все свойства класса. В процессе работы диаграмма должна легко охватываться одним взглядом.
- Не обязательно определять все отношения между классами сразу. Пусть класс на диаграмме «висит в воздухе» – ничего с ним не случится.

3. ПОРЯДОК ВЫПОЛНЕНИЯ РАБОТЫ

В качестве примера рассмотрим построение диаграммы последовательности для моделирования процесса телефонного разговора с использованием обычной телефонной сети. Объектами в этом примере являются: два абонента, a и b, два телефонных аппарата end, коммутатор и сам разговор как объект моделирования. При этом как коммутатор, так и разговор являются анонимными объектами.

На первом этапе располагаем выбранные объекты на предполагаемой диаграмме. Заметим, что абонентов мы будем рассматривать как актеров, причем первый из них – a – играет активную роль, а второй – b – пассивную роль. Поэтому первый получает фокус управления сразу после своего появления в системе, а второй имеет только линию жизни. Коммутатор также имеет постоянную активность, что изображается его фокусом управления. Разговор как объект появляется только после установки соединения и уничтожается с его прекращением. Поэтому он будет изображен позже на этой же диаграмме последовательности.

Процесс взаимодействия в этой системе начинается с поднятия трубки телефонного аппарата первым абонентом. Тем самым он посылает сообщение телефонному аппарату c, которое переводит этот аппарат в активное состояние и вызывает действие – подачу тоно-

вого сигнала в телефонную трубку для первого абонента. Следующее действие также инициируется первым абонентом – набор цифр телефонного номера. Это представлено в форме итеративного сообщения со знаком «*» слева от его имени.

Заметим, что поднятие телефонной трубки и набор цифр номера являются физическими действиями и поэтому изображаются в форме простых асинхронных сообщений. После набора цифр номера телефона аппарат с рекурсивно вызывает процедуру посылки коммутационных импульсов на коммутатор. Последний инициирует создание нового объекта в моделируемой системе – телефонного разговора. После создания анонимный объект «разговор» сразу получает фокус активности и посылает сообщение телефонному аппарату b на выполнение действия – звонка вызова. При этом второй абонент снимает трубку (асинхронное сообщение), тем самым устанавливается прямое соединение между абонентами, a и b. После того как абоненты опустят трубки, разговор заканчивается. Тем самым объект «разговор» уничтожается. Окончательный вариант диаграммы последовательности может содержать некоторые временные ограничения и комментарии. Назначение отдельных сообщений соответствуют рассмотренным действиям.

Диаграмма последовательности рассмотренного примера представлена на рис. 6.1

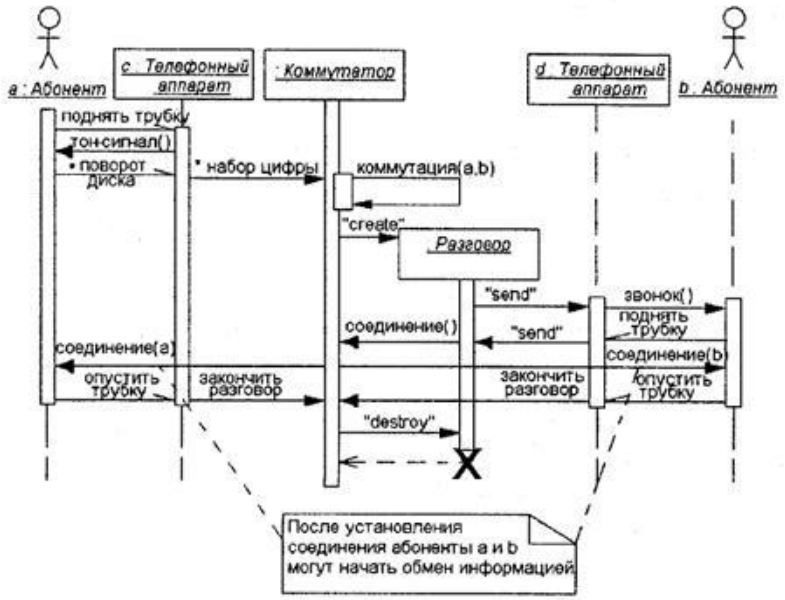


Рис.6.1 Диаграмма последовательности
для моделирования телефонного разговора

Заданием для практической работы является разработка модели проектирования для системы, выданной в качестве задания к предшествующей практической работе.

4. КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Содержание модели проектирования.
2. Какие задачи решаются в процессе проектирования информационной системы?
3. Каким образом определяются классы информационной системы?
4. Каким образом в модели проектирования отображается реализация варианта использования?
5. Как строится диаграмма последовательностей?
6. Как определить состав операций классов информационной системы на основе диаграммы последовательностей, в которой участвуют объекты класса?

РАБОТА №7 ПРАКТИЧЕСКАЯ. ОПРЕДЕЛЕНИЕ ПОКАЗАТЕЛЕЙ БЕЗОТКАЗНОСТИ СИСТЕМЫ

1. ЦЕЛЬ И ЗАДАЧИ РАБОТЫ

Приобретение студентами практических навыков проведения стоимостного анализа и создания свойств, определяемых пользователем.

2. КРАТКИЕ ТЕОРЕТИЧЕСКИЕ ПОЛОЖЕНИЯ

Стоимостной анализ

Как правило, в процессе моделирования строится несколько моделей TO-BE, из которых по какому-либо критерию выбирается лучшая. Для того чтобы определить качество созданной модели с точки зрения эффективности бизнес-процессов, необходима система метрики.

Аналитику предоставляются два инструмента для оценки модели:

– стоимостной анализ (Activity Based Costing)

– свойства, определяемые пользователем (User Defined Properties)

ABC включает следующие основные понятия:

– **объект затрат** – причина, по которой работа выполняется; обычно, основной выход работы, стоимость работ есть суммарная стоимость объектов затрат;

– **движитель затрат** – характеристика входов и управлений работы, которые влияют на то, как выполняется и как долго длится работа;

– **центры затрат**, которые можно трактовать как статьи расхода.

Результаты стоимостного анализа могут существенно повлиять на очередность выполнения работ. ABC позволяет оценить стоимостные и временные характеристики системы.

Свойства, определяемые пользователем (UDP)

Если стоимостных показателей недостаточно, имеется возможность внесения собственных метрик – свойств, определенных пользователем. UDP позволяет произвести дополнительный анализ.

Свойства, определяемые пользователем предназначены для оценки работ модели без суммирующих подсчётов по критериям, вытекающим из особенностей моделируемой предметной области.

Каждой работе можно поставить в соответствие несколько свойств. Список доступных свойств хранится в словаре (UDP Dictionary). На свойства можно ссылаться по ключевому слову (словарь UDP Keyword List). Одно ключевое слово может соответствовать нескольким свойствам и одно свойство может иметь несколько ключевых слов.

Имеется возможность задания 18 различных типов свойств, в том числе управляющих команд и массивов.

Таблица 7.1.

Типы UDP и их использование

Тип	Использование
Text	При задании свойства стрелки или работы просто вносится текст, например это может быть просто дополнительное пояснение
Paragraph Text	Текст в несколько строк
Integer	Целое число, например значение свойства «Количество баллов»
Command	Командная строка.
Character	один символ
Date mm/dd/yy (yy)	дата
Real Number	действительное число, например значение свойства «Потребление электроэнергии, кВт-ч»
Text List (Single selection)	Массив строк. Значения свойства этого типа должны быть определены в диалоге UDP Dictionary (поле Value). Объекту модели можно присваивать только одно значение из предварительно заданного списка
Integer List (Single selection)	Массив целых чисел. Значения свойства этого типа должны быть определены в диалоге UDP Dictionary (поле Value). Объекту модели можно присваивать только одно значение из предварительно заданного списка
Command List	Массив команд. Значения свойства этого типа должны быть определены в диалоге UDP Dictionary (поле Value). Объекту модели можно присваивать только одно значение' из предварительно заданного списка
Date List mm/dd/yy (yy)	Массив дат. Значения свойства этого типа должны быть определены в диалоге UDP Dictionary (поле Value). Объекту модели можно присваивать только одно значение из предварительно заданного списка

Тип	Использование
Real Number List (Single selection)	Массив действительных чисел. Значения свойства этого типа должны быть определены в диалоге UDP Dictionary (поле Value). Объекту модели можно присваивать только одно значение из предварительно заданного списка
Character List (Single selection)	Массив символов. Значения свойства этого типа должны быть определены в диалоге UDP Dictionary (поле Value). Объекту модели можно присваивать только одно значение из предварительно заданного списка
Text List (Multiple selections)	Массив строк (множественный выбор). Значения свойства этого типа должны быть определены в диалоге UDP Dictionary (поле Value). Объекту модели можно присваивать одновременно несколько значений из предварительно заданного списка
Integer List (Multiple selections)	Массив целых чисел (множественный выбор). Значения свойства этого типа должны быть определены в диалоге UDP Dictionary (поле Value). Объекту модели можно присваивать одновременно несколько значений из предварительно заданного списка
Date List (Multiple selections)	Массив дат (множественный выбор). Значения свойства этого типа должны быть определены в диалоге UDP Dictionary (поле Value). Объекту модели можно присваивать одновременно несколько значений из предварительно заданного списка
Real Number List (Multiple selections)	Массив действительных чисел (множественный выбор). Значения свойства этого типа должны быть определены в диалоге UDP Dictionary (поле Value). Объекту модели можно присваивать одновременно несколько значений из предварительно заданного списка
Character List (Multiple selections)	Массив символов (множественный выбор). Значения свойства этого типа должны быть определены в диалоге UDP Dictionary (поле Value). Объекту модели можно присваивать одновременно несколько значений из предварительно заданного списка

Задание стоимости работ

При проведении стоимостного анализа сначала задаются единицы измерения времени и денег.

Для задания единиц измерения следует вызвать диалог Model Properties (меню Model/Model Properties) вкладка ABC Units (рис. 7.1). Каждому центру затрат необходимо дать описание в Cost Center (пункт Dictionary/Cost Center).

Для задания стоимости работы (для каждой работы на диаграмме) следует щелкнуть правой кнопкой мыши по работе и на всплывающем меню выбрать Costs. Во вкладке Costs (рис. 7.2) указывается частота проведения данной работы в рамках общего процесса (Frequency) и продолжительность (Duration). Затем выбирается в списке один из центров затрат и в окне Costs задать его стоимость. Аналогично назначаются суммы по каждому центру затрат.

Если в процессе назначения стоимости возникает необходимость внесения дополнительных центров затрат, диалог Cost Center Editor вызывается прямо из диалога Activity Cost соответствующей кнопкой.

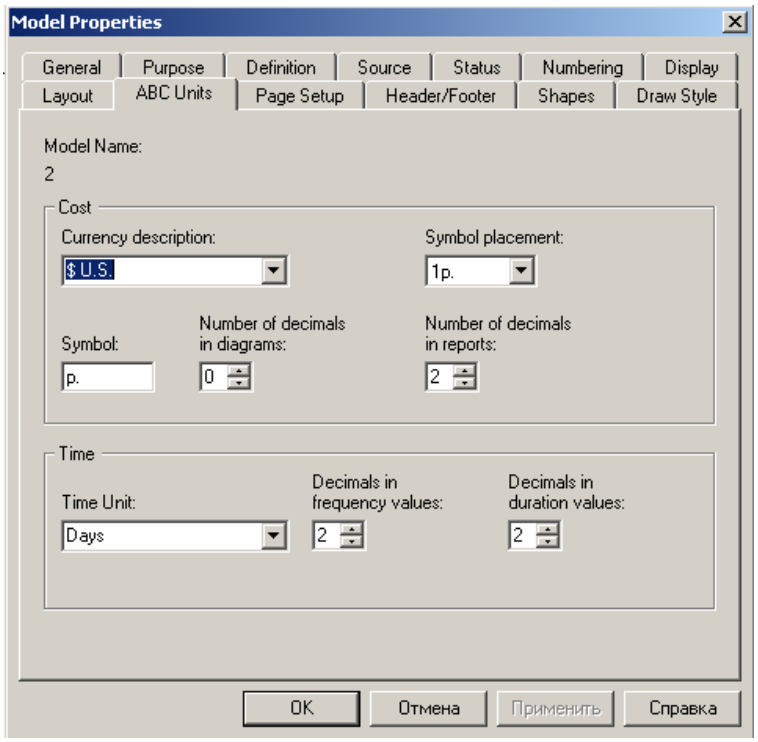


Рис.7.1 Настройка единиц измерения валюты и времени

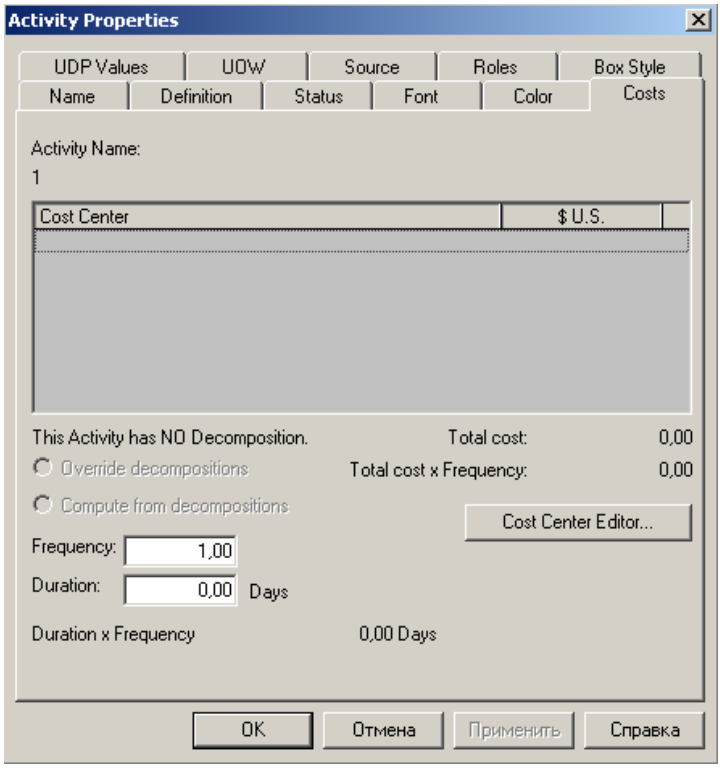


Рис.7.2 Задание стоимости работ в диалоге Activity Cost

Общие затраты по работе рассчитываются как сумма по всем центрам затрат. При вычислении затрат вышестоящей (родительской) работы сначала вычисляется произведение затрат дочерней работы на частоту работы (число раз, которое работа выполняется в рамках проведения родительской работы), затем результаты складываются. Если во всех работах модели включен режим Compute from Decompositions, подобные вычисления автоматически проводятся по всей иерархии работ снизу вверх.

Такой принцип подсчета справедлив, если работы выполняются последовательно.

Если схема выполнения более сложная (например, работы производятся альтернативно), можно отказаться от подсчета и задать итоговые суммы для каждой работы вручную (Override Decompositions). В этом случае результаты расчетов с нижних уровней декомпозиции будут игнорироваться, при расчетах на верхних уровнях будет учитывать сумма, заданная вручную.

Результаты стоимостного анализа наглядно представляются на специальном отчете Activity Cost Report (меню Tools/Report/Activity Cost Report). (рис. 7.3).

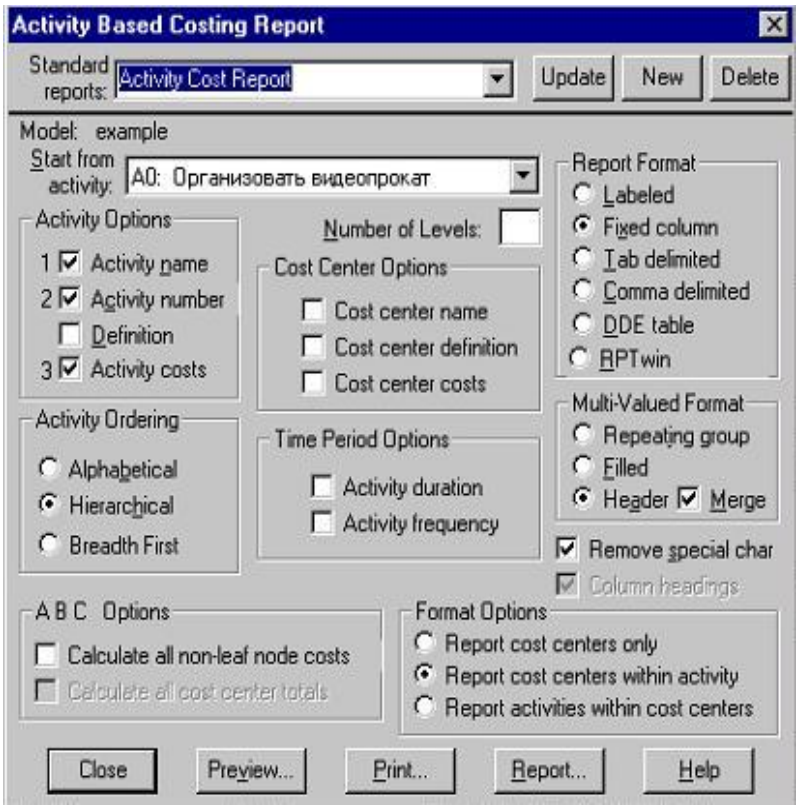


Рис.7.3 Диалог ройки отчета по стоимости работ

Результаты отображаются и непосредственно на диаграммах. В левом нижнем углу прямоугольника работы может показываться либо стоимость, либо продолжительность, либо частота проведения работ

По умолчанию на диаграмме отображается стоимость работы, если требуется изменить настройку, то необходимо воспользоваться диалогом настройки свойств модели (Model/Model Properties/Display – ABC Data и ABC Units);

Задание свойств UDP

Для описания UDP служит диалог UDP Dictionary Editor (меню Model/UDP Dictionary Editor), показанный на рис. 2.4.

Для задания нового свойства (UDP) следует в словаре UDP Dictionary перейти к нижней строке списка и дважды щёлкнуть по полю Name, ввести имя, указать тип свойства и выбрать необходимые ключевые слова.

В качестве свойств может рассматриваться любая мера, позволяющая качественно или количественно оценить работу или связь, например:

– оценка качества или чего-то другого путём выбора из списка возможных значений (очень высокое, высокое, среднее, низкое, очень низкое) и т. п.

– потребляемая мощность или подобная количественная оценка;

– сопровождающая документация, спецификации и т. п.

По результатам анализа, основанного на свойствах, определяемых пользователем, можно сформировать отчёт (Tools/Report/Diagram Object Report) задав в поле User-Defined property необходимые свойства для вывода. Свойства для вывода задаются с помощью кнопки UDP Filter.

3. ПОРЯДОК ВЫПОЛНЕНИЯ РАБОТЫ

1. Согласуйте с преподавателем необходимый набор стоимостный центров и свойств пользователя.
2. Задайте для процессов стоимость их выполнения по стоимостным центрам.
3. Сформируйте отчёт по стоимости работ.
4. Задайте свойства пользователя для процессов декомпозиции.
5. Оформите отчёт и представьте его преподавателю.

Заданием для работы является процесс, рассмотренный в первой практической работе.

4. КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Какие виды инструментов для оценки модели вы знаете?
2. Что такое объект затрат?
3. Что такое центры затрат? (стоимостные центры)
4. Какие характеристики для процесса задаются при ABC анализе?
5. Для чего предназначены свойства, определяемые пользователем?
6. Кратко описать типы свойств.
7. Как рассчитываются общие затраты по работе?
8. Что рассматривается в качестве свойств UDP?

РАБОТА №8 ПРАКТИЧЕСКАЯ. ОПРЕДЕЛЕНИЕ ПОКАЗАТЕЛЕЙ БЕЗОТКАЗНОСТИ СИСТЕМЫ

1. ЦЕЛЬ И ЗАДАЧИ РАБОТЫ.

Целью работы является получение практических навыков выполнения реинжиниринга бизнес-процессов.

2. КРАТКИЕ ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

Прямым проектированием (Forward engineering) называется процесс преобразования модели в код путем отображения на некоторый язык реализации. Процесс прямого проектирования приводит к потере информации, поскольку написанные на языке UML модели семантически богаче любого из существующих объектно-ориентированных языков. Фактически именно это различие и является основной причиной, по которой мы, помимо кода, нуждаемся и в моделях. Некоторые структурные свойства системы, такие как кооперации, или ее поведенческие особенности, например, взаимодействия, могут быть легко визуализированы в UML, но в чистом коде наглядность теряется.

Обратным проектированием (Reverse engineering) называется процесс преобразования в модель кода, записанного на каком-либо языке программирования.

В результате этого процесса вы получаете огромный объем информации, часть которой находится на более низком уровне детализации, чем необходимо для построения полезных моделей. В то же время обратное проектирование никогда не бывает полным. Как уже упоминалось, прямое проектирование ведет к потере информации, так что полностью восстановить модель на основе кода не удастся, если только инструментальные средства не включали в комментариях к исходному тексту информацию, выходящую за пределы семантики языка реализации. Пример, представленный на рис. 8.1, был создан с помощью обратного проектирования библиотеки классов языка Java.

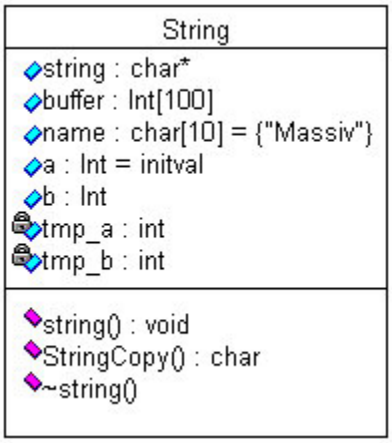


Рис.8.1 Класс, полученный в результате обратного проектирования

Процесс обратного проектирования делится на два этапа: анализ и генерацию модели.

На первом этапе производятся все подготовительные операции по анализу текста программы на отсутствие синтаксических ошибок. Второй этап – преобразование кода в модель.

Все операции выполняются независимо, что дает больший маневр для разработчика, который, например, хочет провести только синтаксический разбор теста, без генерации модели.

Соответственно при отсутствии ошибок в файле можно приступить к генерации модели. В целях оптимизации времени генерации в предусмотрено три способа проведения обратного проектирования, каждый из которых может охватить и превосходно выполнить определенный сегмент работ.

- FirstLook – приближенная пробежка по телу программы.
- Detailed Analysis – детальный анализ проекта.
- RoundTrip – комбинация двух вышеперечисленных способов. Позволяет безболезненно строить и перестраивать разрабатываемые приложения по принципу круговой разработки.

Нашей целью будет получение графической модели из класса на языке программирования. Обратите внимание на комментарии. Каждая строка снабжена комментарием. Смысл обратного проектирования состоит не только в том, чтобы корректно нарисовать модель, но и для правильного описания спецификации каждой составляющей класса. За основу программы возьмем следующий класс:

```
//It's main class class string public:
char *string; //Structure's pointer
int buffer[100]; //Temporary buffer
char name[10]={«Massiv»}; //Name of data
int a; //Integer
int b; //Integer void string(void); //constructor
void ~string(void); //destructor
char StringCopy(char *, //Buffer char *, //sourcel char *); //source2 private:
int tmp_a;
int tmp_b;
```

3. ПОРЯДОК ВЫПОЛНЕНИЯ РАБОТЫ

Заданием для работы является исходный код, содержащий описание классов. Классы должны содержать операции и их реализацию.

Выполнение должно включать следующие этапы.

1. Анализ кода и выделение состав классов.
2. Отображение классов в среде моделирования. Для каждого класса в отчёте по работе должно быть сделано пояснение – на основе какого программного объекта он выявлен.
3. Выявление переменных классов и отображение их в среде моделирования. Для каждой переменной класса в отчёте по лабораторной работе должно быть сделано пояснение.
4. Выявление операций классов. Для каждого класса необходимо выявить операции и сделано пояснение о свойствах операции, её параметрах и уровне в иерархии наследования.

4. КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Что из себя представляет обратное проектирование?
2. Для чего выполняется обратное проектирование?
3. Что из себя представляет результат обратного проектирования?
4. Какие модели можно построить в результате обратного проектирования?
5. Существуют ли инструментальные средства для обратного проектирования?

РАБОТА №9 ПРАКТИЧЕСКАЯ. ОПРЕДЕЛЕНИЕ ПОКАЗАТЕЛЕЙ ДОЛГОВЕЧНОСТИ СИСТЕМЫ

1. ЦЕЛЬ И ЗАДАЧИ РАБОТЫ

Целью работы является получение практических навыков разработки требований безопасности к информационной системе.

2. КРАТКИЕ ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

Критерии оценки безопасности ИС отображены в стандарте ISO/IEC 15408 «Критерии оценки безопасности информационных технологий» (издан 1 декабря 1999 года) относится к оценочным стандартам.

Стандарт содержит два основных вида требований безопасности:

- функциональные – соответствуют активному аспекту защиты – предъявляемые к функциям безопасности и реализующим их механизмам;
- требования доверия – соответствуют пассивному аспекту – предъявляемые к технологии и процессу разработки и эксплуатации.

Угрозы безопасности в стандарте характеризуются следующими параметрами:

- источник угрозы;
- метод воздействия;
- уязвимые места, которые могут быть использованы;
- ресурсы (активы), которые могут пострадать.

Для структуризации пространства требований, в стандарте введена иерархия класс – семейство – компонент – элемент.

Классы определяют наиболее общую, «предметную» группировку требований (например, функциональные требования подотчетности).

Семейства в пределах класса различаются по строгости и другим тонкостям требований.

Компонент – минимальный набор требований, фигурирующий как целое.

Элемент – неделимое требование.

Между компонентами могут существовать зависимости, которые возникают, когда компонент сам по себе недостаточен для достижения цели безопасности.

Подобный принцип организации защиты напоминает принцип программирования с использованием библиотек, в которых содержатся стандартные (часто используемые) функции, из комбинаций которых формируется алгоритм решения.

Стандарт позволяет с помощью подобных библиотек (компонент) формировать два вида нормативных документов: профиль защиты и задание по безопасности.

Профиль защиты представляет собой типовой набор требований, которым должны удовлетворять продукты и/или системы определенного класса (например, операционные системы на компьютерах в правительственных организациях).

Задание по безопасности содержит совокупность требований к конкретной разработке, выполнение которых обеспечивает достижение поставленных целей безопасности.

Функциональный пакет – это неоднократно используемая совокупность компонентов, объединенных для достижения определенных целей безопасности.

Базовый профиль защиты должен включать требования к основным (обязательным в любом случае) возможностям. Производные профили получаются из базового путем добавления необходимых пакетов расширения, то есть подобно тому, как создаются производные классы в объектно-ориентированных языках программирования.

Функциональные требования

Все функциональные требования объединены в группы на основе выполняемой ими роли или обслуживаемой цели безопасности. Всего в «Общих критериях» представлено 11 функциональных классов, 66 семейств, 135 компонентов.

«Общие критерии» включают следующие классы функциональных требований:

- Идентификация и аутентификация.
- Защита данных пользователя.
- Защита функций безопасности (требования относятся к целостности и контролю данных сервисов безопасности и реализующих их механизмов).

- Управление безопасностью (требования этого класса относятся к управлению атрибутами и параметрами безопасности).
- Аудит безопасности (выявление, регистрация, хранение, анализ данных, затрагивающих безопасность объекта оценки, реагирование на возможное нарушение безопасности).
- Доступ к объекту оценки.
- Приватность (защита пользователя от раскрытия и несанкционированного использования его идентификационных данных).
- Использование ресурсов (требования к доступности информации).
- Криптографическая поддержка (управление ключами).
- Связь (аутентификация сторон, участвующих в обмене данными).
- Доверенный маршрут/канал (для связи с сервисами безопасности).

Рассмотрим содержание одного из классов.

Класс функциональных требований «Использование ресурсов» включает три семейства:

- Отказоустойчивость. Требования этого семейства направлены на сохранение доступности информационных сервисов даже в случае сбоя или отказа. В стандарте различаются активная и пассивная отказоустойчивость. Активный механизм содержит специальные функции, которые активизируются в случае сбоя. Пассивная отказоустойчивость подразумевает наличие избыточности с возможностью нейтрализации ошибок.
- Обслуживание по приоритетам. Выполнение этих требований позволяет управлять использованием ресурсов так, что низкоприоритетные операции не могут помешать высокоприоритетным.
- Распределение ресурсов. Требования направлены на защиту (путем применения механизма квот) от несанкционированной монополизации ресурсов.

Аналогично и другие классы включают наборы семейств требований, которые используются для формулировки требований к системе безопасности.

«Общие критерии» – достаточно продуманный и полный документ с точки зрения функциональных требований и именно на

этот стандарт безопасности ориентируются соответствующие организации в нашей стране и в первую очередь Гостехкомиссия РФ.

Требования доверия

Вторая форма требований безопасности в «Общих критериях» – требования доверия безопасности.

Установление доверия безопасности основывается на активном исследовании объекта оценки.

Форма представления требований доверия, та же, что и для функциональных требований (класс – семейство – компонент).

Всего в «Общих критериях» 10 классов, 44 семейства, 93 компонента требований доверия безопасности.

Классы требований доверия безопасности:

- Разработка (требования для поэтапной детализации функций безопасности от краткой спецификации до реализации).
- Поддержка жизненного цикла (требования к модели жизненного цикла, включая порядок устранения недостатков и защиту среды разработки).
- Тестирование.
- Оценка уязвимостей (включая оценку стойкости функций безопасности).
- Поставка и эксплуатация.
- Управление конфигурацией.
- Руководства (требования к эксплуатационной документации).
- Поддержка доверия (для поддержки этапов жизненного цикла после сертификации).
- Оценка профиля защиты.
- Оценка задания по безопасности.

Применительно к требованиям доверия (для функциональных требований не предусмотрены) в «Общих критериях» введены оценочные уровни доверия (их семь), содержащие осмысленные комбинации компонентов.

Степень доверия возрастает от первого к седьмому уровню. Так, оценочный уровень доверия 1 (начальный) применяется, когда угрозы не рассматриваются как серьезные, а оценочный уровень 7 применяется к ситуациям чрезвычайно высокого риска.

3. ПОРЯДОК ВЫПОЛНЕНИЯ РАБОТЫ

1. Проанализировать заданную систему.
2. Сформулировать 5 функциональных требований безопасности актуальных для системы рассматриваемого типа.
3. Сформулировать 5 требований доверия к безопасности, характерных для системы рассматриваемого типа.
4. Создать отчёт по работе, содержащий перечень сформированных требований.

В качестве задания для работы преподавателем задаётся одна из систем, знакомая студенту, например:

- система контроля входа в корпус;
- система записи на приём в поликлинику;
- система пожарного контроля в корпусе;
- ИС КузГТУ.

4. КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Какие виды требований включает стандарт ISO/IEC 15408?
2. Чем отличаются функциональные требования от требований доверия?
3. В чем заключается иерархический принцип «класс – семейство – компонент – элемент»?
4. Какова цель требований по отказоустойчивости информационных систем?
5. Сколько классов функциональных требований?

РАБОТА №10 ПРАКТИЧЕСКАЯ. ОПРЕДЕЛЕНИЕ ЕДИНИЧНЫХ ПОКАЗАТЕЛЕЙ ДОСТОВЕРНОСТИ ИНФОРМАЦИИ В СИСТЕМЕ

1. ЦЕЛЬ И ЗАДАЧИ РАБОТЫ

Целью выполнения работы является получение практических навыков разработки технического задания на создание информационной системы. Для выполнения целей работы должны быть решены следующие задачи.

- Студенты должны ознакомиться с содержанием технического задания.
- Получить навыки формулировки целей разработки информационной системы.
- Формулировки требований к ИС.

2. КРАТКИЕ ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

Общие положения

1.1. Техническое задание (ТЗ) является основным документом, определяющим требования и порядок создания (развития или модернизации – далее создания) информационной системы (далее ИС), в соответствии с которым проводится разработка ИС и ее приемка при вводе в действие.

1.2. ТЗ разрабатывают на систему в целом, предназначенную для работы самостоятельно или в составе другой системы.

1.3. Требования к ИС в объеме, установленном настоящим стандартом, могут быть включены в задание на проектирование вновь создаваемого объекта информатизации. В этом случае ТЗ не разрабатывают.

1.4. Включаемые в ТЗ требования должны соответствовать современному уровню развития информационных технологий и не уступать аналогичным требованиям, предъявляемым к лучшим современным отечественным и зарубежным аналогам. Задаваемые в ТЗ требования не должны ограничивать разработчика системы в поиске и реализации наиболее эффективных технических, технико-экономических и других решений.

1.5. В ТЗ включают только те требования, которые дополняют требования к системам данного вида и определяются спецификой конкретного объекта, для которого создается система.

1.6. Изменения к ТЗ оформляют дополнением или подписанным заказчиком и разработчиком протоколом. Дополнение или указанный протокол являются неотъемлемой частью ТЗ на ИС. На титульном листе ТЗ должна быть запись «Действует с ...».

2. Состав и содержание

2.1. ТЗ содержит следующие разделы, которые могут быть разделены на подразделы:

1) общие сведения;

2) назначение и цели создания (развития) системы;

3) характеристика объектов;

4) требования к системе;

5) состав и содержание работ по созданию системы;

6) порядок контроля и приемки системы;

7) требования к составу и содержанию работ по подготовке объекта разработки к вводу системы в действие;

8) требования к документированию;

9) источники разработки.

В ТЗ могут включаться приложения.

2.2. В зависимости от вида, назначения, специфических особенностей проекта и условий функционирования системы допускается оформлять разделы ТЗ в виде приложений, вводить дополнительные, исключать или объединять подразделы ТЗ.

В ТЗ на части системы не включают разделы, дублирующие содержание разделов ТЗ в целом.

2.3. В разделе «Общие сведения» указывают:

1) полное наименование системы и ее условное обозначение;

2) шифр темы или шифр (номер) договора;

3) наименование компаний разработчика и заказчика (пользователя) системы и их реквизиты;

4) перечень документов, на основании которых создается система, кем и когда утверждены эти документы;

5) плановые сроки начала и окончания работы по созданию системы;

6) сведения об источниках и порядке финансирования работ;

7) порядок оформления и предъявления заказчику результатов работ по созданию системы (ее частей), по изготовлению и наладке отдельных средств (технических, программных, информационных) и программно-технических (программно-методических) комплексов системы.

2.4. Раздел «Назначение и цели создания (развития) системы» состоит из подразделов:

1) назначение системы;

2) цели создания системы.

2.4.1. В подразделе «Назначение системы» указывают вид деятельности системы (управление, проектирование и т. п.) и перечень объектов информатизации (объектов), на которых предполагается ее использовать.

2.4.2. В подразделе «Цели создания системы» приводят наименования и требуемые значения технических, технологических, производственно-экономических или других показателей объекта информатизации, которые должны быть достигнуты в результате создания ИС, и указывают критерии оценки достижения целей создания системы.

2.5. В разделе «Характеристики объекта информатизации» приводят:

1) краткие сведения об объекте информатизации или ссылки на документы, содержащие такую информацию;

2) сведения об условиях эксплуатации объекта автоматизации.

2.6. Раздел «Требования к системе» состоит из следующих подразделов:

1) требования к системе в целом;

2) требования к функциям (задачам), выполняемым системой;

3) требования к видам обеспечения.

Состав требований к системе, включаемых в данный раздел ТЗ на ИС, устанавливают в зависимости от вида, назначения, специфических особенностей и условий функционирования конкретной системы.

2.6.1. В подразделе «Требования к системе в целом» указывают:

- требования к структуре и функционированию системы;
- требования к численности и квалификации персонала системы и режиму его работы;
- показатели назначения;

- требования к надежности;
- требования безопасности;
- требования к эргономике и технической эстетике;
- требования к эксплуатации, техническому обслуживанию, ремонту и хранению компонентов системы;
- требования к защите информации от несанкционированного доступа;
- требования по сохранности информации при авариях;
- требования к защите от влияния внешних воздействий;
- требования к патентной чистоте;
- требования по стандартизации и унификации;
- дополнительные требования.

2.6.1.1. В требованиях к структуре и функционированию системы приводят:

1) перечень подсистем, их назначение и основные характеристики, требования к числу уровней иерархии и степени централизации системы;

2) требования к способам и средствам связи для информационного обмена между компонентами системы;

3) требования к характеристикам взаимосвязей создаваемой системы со смежными системами, требования к ее совместимости, в том числе указания о способах обмена информацией (автоматически, пересылкой документов, по телефону и т. п.);

4) требования к режимам функционирования системы;

5) требования по диагностированию системы;

6) перспективы развития, модернизации системы.

2.6.1.2. В требованиях к численности и квалификации персонала на ИС приводят:

- требования к численности персонала (пользователей) ИС;
- требования к квалификации персонала, порядку его подготовки и контроля знаний и навыков;
- требуемый режим работы персонала ИС.

2.6.1.3. В требованиях к показателям назначения ИС приводят значения параметров, характеризующие степень соответствия системы ее назначению.

2.6.1.4. В требования к надежности включают:

1) состав и количественные значения показателей надежности для системы в целом или ее подсистем;

2) перечень аварийных ситуаций, по которым должны быть регламентированы требования к надежности, и значения соответствующих показателей;

3) требования к надежности технических средств и программного обеспечения;

4) требования к методам оценки и контроля показателей надежности на разных стадиях создания системы в соответствии с действующими нормативно-техническими документами.

2.6.1.5. В требования по безопасности включают требования по обеспечению безопасности при поставке, наладке, эксплуатации и обслуживании системы.

2.6.1.6. В требования по эргономике и технической эстетике включают показатели ИС, задающие необходимое качество взаимодействия человека с машиной и комфортность условий работы персонала.

2.6.1.7. В требования к защите информации от несанкционированного доступа включают требования, установленные действующей в отрасли и информационной среде заказчика.

2.6.1.8. В требованиях по сохранности информации приводят перечень событий: аварий, отказов технических средств (в том числе – потеря питания) и т. п., при которых должна быть обеспечена сохранность информации в системе.

2.6.1.9. В требованиях по патентной чистоте указывают перечень стран, в отношении которых должна быть обеспечена патентная чистота системы и ее частей.

2.6.1.10. В дополнительные требования включают специальные требования по усмотрению разработчика или заказчика системы.

2.6.2. В подразделе «Требование к функциям (задачам)», выполняемым системой, приводят:

- по каждой подсистеме перечень функций, задач или их комплексов (в том числе обеспечивающих взаимодействие частей системы), подлежащих автоматизации;
- при создании системы в две или более очереди – перечень функциональных подсистем, отдельных функций или задач, вводимых в действие в 1-й и последующих очередях;
- временной регламент реализации каждой функции, задачи (или комплекса задач);

- требования к качеству реализации каждой функции (задачи или комплекса задач), к форме представления выходной информации, характеристики необходимой точности и времени выполнения, требования одновременности выполнения группы функций, достоверности выдачи результатов;
- перечень и критерии отказов для каждой функции, по которой задаются требования по надежности.

2.6.3. В подразделе «Требования к видам обеспечения» в зависимости от вида системы приводят требования к математическому, информационному, лингвистическому, программному, техническому, метрологическому, организационному, методическому и другие видам обеспечения системы.

2.6.3.2. Для информационного обеспечения системы приводят требования:

1) к составу, структуре и способам организации данных в системе;

2) к информационному обмену между компонентами системы;

3) к информационной совместимости со смежными системами;

4) по применению систем управления базами данных;

5) к структуре процесса сбора, обработки, передачи данных в системе и представлению данных;

6) к защите данных;

7) к контролю, хранению, обновлению и восстановлению данных;

2.6.3.3. Для лингвистического обеспечения системы приводят требования к применению в системе языков программирования высокого уровня, языков взаимодействия пользователей и технических средств системы, а также требования к кодированию и декодированию данных, к языкам ввода-вывода данных, языкам манипулирования данными, средствам описания предметной области, к способам организации диалога.

2.6.3.4. Для программного обеспечения системы приводят перечень покупных программных средств, а также требования:

1) к зависимости программных средств от операционной среды;

2) к качеству программных средств, а также к способам его обеспечения и контроля;

2.6.3.5. Для технического обеспечения системы приводят требования:

1) к видам технических средств, в том числе к видам комплексов технических средств, программно-технических комплексов и других комплектующих изделий, допустимых к использованию в системе;

2) к функциональным, конструктивным и эксплуатационным характеристикам средств технического обеспечения системы.

2.6.3.6. В требованиях к метрологическому обеспечению приводят:

1) предварительный перечень измерительных каналов;

2) требования к точности измерений параметров и (или) к метрологическим характеристикам измерительных каналов;

3) требования к метрологической совместимости технических средств системы;

4) перечень управляющих и вычислительных каналов системы, для которых необходимо оценивать точностные характеристики;

5) требования к метрологическому обеспечению технических и программных средств, входящих в состав измерительных каналов системы, средств, встроенного контроля, метрологической пригодности измерительных каналов и средств измерений, используемых при наладке и испытаниях системы;

6) вид метрологической аттестации (государственная или ведомственная) с указанием порядка ее выполнения и организаций, проводящих аттестацию.

2.6.3.7. Для организационного обеспечения приводят требования:

1) к структуре и функциям подразделений, участвующих в функционировании системы или обеспечивающих эксплуатацию;

2) к организации функционирования системы и порядку взаимодействия персонала ИС и персонала объекта информатизации;

3) к защите от ошибочных действий персонала системы.

2.7. Раздел «Состав и содержание работ по созданию (развитию) системы» должен содержать перечень стадий и этапов работ по созданию системы, сроки их выполнения, перечень организаций – исполнителей работ, ссылки на документы, подтверждающие согласие этих организаций на участие в создании системы, или запись, определяющую ответственного (заказчик или разработчик) за проведение этих работ.

В данном разделе также приводят:

1) перечень документов, предъявляемых по окончании соответствующих стадий и этапов работ;

2) вид и порядок проведения экспертизы технической документации (стадия, этап, объем проверяемой документации, организация-эксперт);

3) программу работ, направленных на обеспечение требуемого уровня надежности разрабатываемой системы (при необходимости);

4) перечень работ по метрологическому обеспечению на всех стадиях создания системы с указанием их сроков выполнения и организаций-исполнителей (при необходимости).

2.8. В разделе «Порядок контроля и приемки системы» указывают:

1) виды, состав, объем и методы испытаний системы и ее составных частей;

2) общие требования к приемке работ по стадиям, порядок согласования и утверждения приемочной документации;

2.9. В разделе «Требования к составу и содержанию работ по подготовке объекта автоматизации к вводу системы в действие» необходимо привести перечень основных мероприятий и их исполнителей, которые следует выполнить при подготовке проекта к вводу ИС в действие.

В перечень основных мероприятий включают:

1) приведение поступающей в систему информации (в соответствии с требованиями к информационному и лингвистическому обеспечению);

2) создание условий функционирования проекта, при которых гарантируется соответствие создаваемой системы требованиям, содержащимся в ТЗ;

3) создание необходимых для функционирования системы подразделений и служб;

4) сроки и порядок комплектования штатов и обучения персонала.

2.10. В разделе «Требования к документированию» приводят:

1) согласованный разработчиком и Заказчиком системы перечень подлежащих разработке комплектов и видов документов; перечень документов, выпускаемых на машинных носителях;

2) при отсутствии государственных стандартов, определяющих требования к документированию элементов системы, дополнительно включают требования к составу и содержанию таких документов.

2.11. В разделе «Источники разработки» должны быть перечислены документы и информационные материалы, на основании которых разрабатывалось ТЗ и которые должны быть использованы при создании системы.

3. ПОРЯДОК ВЫПОЛНЕНИЯ РАБОТЫ

Работа выполняется на основе предшествующей работы по формированию требований. В результате выполнения работы должен быть оформлен документ «Техническое задание».

Заданием для работы является разработка технического задания для ИС, являющейся темой работы по формулировке требований.

4. КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Что такое техническое задание?
2. Какие подразделы выделяются в техническом задании?
3. Что включается в первый и второй подразделы технического задания? Как формулируется назначение и цель системы?
4. Что является объектом автоматизации?
5. Какие виды требований выделяются в техническом задании?
6. Где в техническом задании отображаются функциональные требования к системе?
7. Какие виды обеспечений выделяются у информационной системы в техническом задании?

РАБОТА №11 ЛАБОРАТОРНАЯ. СБОР ИНФОРМАЦИИ ОБ ОШИБКАХ. ФОРМИРОВАНИЕ ОТЧЕТА ОБ ОШИБКАХ

1. ЦЕЛЬ И ЗАДАЧИ РАБОТЫ

Целью выполнения работы является получение практических навыков разработки руководства пользователя информационной системы.

2. КРАТКИЕ ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

Руководство пользователя формируется IT-специалистом документации на систему и её.

Для руководства пользователя разработано ряд стандартов, в частности ГОСТ РД 50-34.698-90. В соответствии с данными стандартом руководство пользователя содержит следующие разделы.

Введение.

1. Назначение и условия применения.
2. Подготовка к работе.
3. Описание операций.
4. Аварийные ситуации.
5. Рекомендации по освоению.

Содержание разделов будет описано на основе конкретного примера.

1. Введение

В разделе «Введение» указывают:

- *область применения;*
- *краткое описание возможностей;*
- *уровень подготовки пользователя;*
- *перечень эксплуатационной документации, с которой необходимо ознакомиться пользователю.*

1.1. Область применения

Требования настоящего документа применяются при:

- предварительных комплексных испытаниях;
- опытной эксплуатации;
- приемочных испытаниях;

- промышленной эксплуатации.

1.2. Краткое описание возможностей

Информационно-аналитическая система Корпоративное Хранилище Данных (ИАС КХД) предназначена для оптимизации технологии принятия тактических и стратегических управленческих решений конечными бизнес-пользователями на основе информации о всех аспектах финансово-хозяйственной деятельности Компании.

ИАС КХД предоставляет возможность работы с регламентированной и нерегламентированной отчетностью.

При работе с отчетностью используется инструмент пользователя Oracle Discoverer Plus, который предоставляет следующие возможности:

- формирование табличных и кросс-табличных отчетов;
- построение различных диаграмм;
- экспорт и импорт результатов анализа;
- печать результатов анализа;
- распространение результатов анализа.

1.3. Уровень подготовки пользователя

Пользователь ИАС КХД должен иметь опыт работы с ОС MS Windows (95/98/NT/2000/XP), навык работы с ПО Internet Explorer, Oracle Discoverer, а также обладать следующими знаниями:

- знать соответствующую предметную область;
- знать основы многомерного анализа;
- понимать многомерную модель соответствующей предметной области;
- знать и иметь навыки работы с аналитическими приложениями.

Квалификация пользователя должна позволять:

- формировать отчеты в Oracle Discoverer Plus;
- осуществлять анализ данных.

1.4. Перечень эксплуатационной документации, с которой необходимо ознакомиться пользователю

- Информационно-аналитическая система «Корпоративное хранилище данных».
- Информационно-аналитическая система «Корпоративное хранилище данных».

2. Назначение и условия применения Oracle Discoverer Plus

В разделе «Назначение и условия применения» указывают:

- виды деятельности, функции, для автоматизации которых предназначено данное средство автоматизации;
- условия, при соблюдении (выполнении, наступлении) которых обеспечивается применение средства автоматизации в соответствии с назначением (например, вид ЭВМ и конфигурация технических средств, операционная среда и общесистемные программные средства, входная информация, носители данных, база данных, требования к подготовке специалистов и т. п.).

Oracle Discoverer Plus в составе ИАС КХД предназначен для автоматизации подготовки, настройки отчетных форм по показателям деятельности, а также для углубленного исследования данных на основе корпоративной информации хранилища данных.

Работа с Oracle Discoverer Plus в составе ИАС КХД возможна всегда, когда есть необходимость в получении информации для анализа, контроля, мониторинга и принятия решений на ее основе.

Работа с Oracle Discoverer Plus в составе ИАС КХД доступна всем пользователям с установленными правами доступа.

3. Подготовка к работе

В разделе «Подготовка к работе» указывают:

- состав и содержание дистрибутивного носителя данных;
- порядок загрузки данных и программ;
- порядок проверки работоспособности.

3.1. Состав и содержание дистрибутивного носителя данных

Для работы с ИАС КХД необходимо следующее программное обеспечение:

Internet Explorer (входит в состав операционной системы Windows);

Oracle JInitiator устанавливается автоматически при первом обращении пользователя к ИАС КХД.

3.2. Порядок загрузки данных и программ

Перед началом работы с ИАС КХД на рабочем месте пользователя необходимо выполнить следующие действия:

1. Необходимо зайти на сайт ИАС КХД ias-dwh.ru.

2. Во время загрузки в появившемся окне «Предупреждение о безопасности», которое будет содержать следующее – хотите установить и выполнить «Oracle JInitiator». Нажимаем на кнопку «Да».

3. После чего запуститься установка Oracle JInitiator на Ваш компьютер.

4. Выбираем кнопку Next и затем OK.

3.3. Порядок проверки работоспособности

Для проверки доступности ИАС КХД с рабочего места пользователя необходимо выполнить следующие действия:

1. Открыть Internet Explorer, для этого необходимо кликнуть по ярлыку «Internet Explorer» на рабочем столе или вызвать из меню «Пуск».

2. Ввести в адресную строку Internet Explorer адрес: ias-dwh.ru и нажать «Переход».

3. В форме аутентификации ввести пользовательский логин и пароль. Нажать кнопку «Далее».

4. Убедиться, что в окне открылось приложение Oracle Discoverer Plus.

В случае если приложение Oracle Discoverer Plus не запускается, то следует обратиться в службу поддержки.

4. Описание операций

В разделе «Описание операций» указывают:

- описание всех выполняемых функций, задач, комплексов задач, процедур;
- описание операций технологического процесса обработки данных, необходимых для выполнения функций, комплексов задач (задач), процедур.

Для каждой операции обработки данных указывают:

- наименование;
- условия, при соблюдении которых возможно выполнение операции;
- подготовительные действия;
- основные действия в требуемой последовательности;
- заключительные действия;
- ресурсы, расходуемые на операцию.

В описании действий допускаются ссылки на файлы подсказок, размещенные на магнитных носителях.

4.1. Выполняемые функции и задачи

Oracle Discoverer Plus в составе ИАС КХД выполняет функции и задачи, приведенные в таблице ниже:

1. Функция – обеспечивает многомерный анализ в табличной и графической формах.

1.1. Задача – визуализация отчетности.

Описание – В ходе выполнения данной задачи пользователю системы предоставляется возможность работы с выбранным отчетом из состава преднастроенных.

1.2. Задача – Формирование табличных и графических форм отчетности

Описание – В ходе выполнения данной задачи пользователю системы предоставляется возможность формирования собственного отчета в табличном или графическом виде на базе преднастроенных компонентов.

4.2. Описание операций технологического процесса обработки данных, необходимых для выполнения задач

Ниже приведено описание пользовательских операций для выполнения каждой из задач.

Задача: «Визуализация отчетности»

Операция 1: Регистрация на портале ИАС КХД

Условия, при соблюдении которых возможно выполнение операции:

- Компьютер пользователя подключен к корпоративной сети.
- Портал ИАС КХД доступен.
- ИАС КХД функционирует в штатном режиме.

Подготовительные действия:

На компьютере пользователя необходимо выполнить дополнительные настройки, приведенные в п. 3.2 настоящего документа.

Основные действия в требуемой последовательности:

1. На иконке «ИАС КХД» рабочего стола произвести двойной щелчок левой кнопкой мышки.

2. В открывшемся окне в поле «Логин» ввести имя пользователя, в поле «Пароль» ввести пароль пользователя. Нажать кнопку «Далее».

Заключительные действия:

- Не требуются.

Ресурсы, расходуемые на операцию:

- 15–30 секунд.

Операция 2: Выбор отчета

Условия, при соблюдении которых возможно выполнение операции:

- Успешная регистрация на Портале ИАС КХД.

Подготовительные действия:

- Не требуются.

Основные действия в требуемой последовательности:

1. В появившемся окне «Мастер создания рабочих книг» поставить точку напротив пункта «Открыть существующую рабочую книгу».
2. Выбрать нужную рабочую книгу и нажать кнопку «Откр.»:

Заключительные действия:

1. После завершения работы с отчетом необходимо выбрать пункт меню «Файл».
2. Далее выбрать пункт «Закрыть».

Ресурсы, расходуемые на операцию:

15 секунд.

Задача: «Формирование табличных и графических форм отчетности»

Заполняется по аналогии.

5. Аварийные ситуации

В разделе «Аварийные ситуации» указывают:

1. Действия в случае несоблюдения условий выполнения технологического процесса, в том числе при длительных отказах технических средств.
2. Действия по восстановлению программ и/или данных при отказе магнитных носителей или обнаружении ошибок в данных.
3. Действия в случаях обнаружении несанкционированного вмешательства в данные.
4. Действия в других аварийных ситуациях.

В случае возникновения ошибок при работе ИАС КХД, не описанных ниже в данном разделе, необходимо обращаться к сотруднику подразделения технической поддержки ДИТ (HelpDesk) либо к ответственному Администратору ИАС КХД.

Описание возможных ошибок приведено ниже.

Класс ошибки – Портал ИАС КХД

Ошибка – Сервер не найден. Невозможно отобразить страницу

Описание ошибки – Возможны проблемы с сетью или с доступом к порталу ИАС КХД.

Требуемые действия – Для устранения проблем с сетью обратиться к сотруднику подразделения технической поддержки (HelpDesk). В других случаях к администратору ИАС КХД.

Ошибка – Требуется ввести действительное имя пользователя

Описание ошибки – При регистрации на портале ИАС КХД не введено имя пользователя

Требуемые действия – Ввести имя пользователя.

Ошибка – Требуется ввести пароль для регистрации

Описание ошибки – При регистрации на портале ИАС КХД не введен пароль.

Требуемые действия – Ввести пароль.

Класс ошибки – Сбой в электропитании рабочей станции.

Ошибка – Нет электропитания рабочей станции или произошел сбой в электропитании.

Описание ошибки – Рабочая станция выключилась или перезагрузилась.

Требуемые действия – Перезагрузить рабочую станцию. Проверить доступность сервера ИАС КХД по порту 80, выполнив следующие команды:

- нажать кнопку «Пуск»;
- выбрать пункт «Выполнить»;
- в строке ввода набрать команду telnet ias_dwh.ru 80;
- если открылось окно Telnet, значит соединение возможно;
- Повторить попытку подключения (входа) в ИАС КХД.

3. ПОРЯДОК ВЫПОЛНЕНИЯ РАБОТЫ

1. Ознакомится с теоретическими сведениями. Изучить состав типового руководства пользователя.
2. Создать руководство пользователя для заданной системы.
3. Оформить руководство пользователя в виде отчёта по работе.

Заданием для работы является разработка руководства пользователя для системы, рассмотренной в предшествующей работе.

4. КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Для чего разрабатывается руководство пользователя?
2. Кто разрабатывает руководство пользователя?
3. Перечислите основные разделы руководства пользователя.
4. Что включается в описание операции?
5. Типы ошибок при работе ИАС КХД и пути их решения.

РАБОТА №12 ЛАБОРАТОРНАЯ. ВЫЯВЛЕНИЕ И УСТРАНЕНИЕ ОШИБОК ПРОГРАММНОГО КОДА ИНФОРМАЦИОННЫХ СИСТЕМ

1. ЦЕЛЬ И ЗАДАЧИ РАБОТЫ

Целью работы является получение теоретических навыков по работе со средствами автоматического создания документации.

2. КРАТКИЕ ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

Для создания документации в процессе разработки ИС используются разнообразные средства формирования отчетов, а также компоненты издательских систем. Обычно средства документирования встроены в конкретные CASE-средства. Исключением являются некоторые пакеты, предоставляющие дополнительный сервис при документировании. Из них наиболее активно используется SoDA (Software Document Automation).

Продукт SoDA предназначен для автоматизации разработки проектной документации на всех фазах ЖЦ ПО. Он позволяет автоматически извлекать разнообразную информацию, получаемую на разных стадиях разработки проекта, и включать ее в выходные документы. При этом контролируется соответствие документации проекту, взаимосвязь документов, обеспечивается их своевременное обновление. Результирующая документация автоматически формируется из множества источников, число которых не ограничено.

SoDA не зависит от применяемых инструментальных средств. Связь с приложениями осуществляется через стандартный программный интерфейс API. Переход на новые инструментальные средства не влечет за собой дополнительных затрат по документированию проекта.

SoDA содержит набор шаблонов документов, определяемых стандартом на программное обеспечение DOD 2167A. На их основе можно без специального программирования создавать новые формы документов, определяемые пользователями.

Пакет включает в себя графический редактор для подготовки шаблонов документов. Он позволяет задавать необходимый стиль, фон, шрифт, определять расположение заголовков, резервировать места, где будет размещаться извлекаемая из разнообразных источ-

ников информация. Изменения автоматически вносятся только в те части документации, на которые они повлияли в программе. Это сокращает время подготовки документации за счет отказа от перегенерации всей документации.

SoDA реализована на базе издательской системы FrameBuilder и предоставляет полный набор средств по редактированию и верстке выпускаемой документации. Разные версии документации могут быть для наглядности отмечены своими отличительными признаками. В системе создаются таблицы требований к проекту, по которым можно проследить, как реализуются эти требования. Разные виды документации, сопровождающие различные этапы ЖЦ, связаны между собой, и можно проследить состояние проекта от первоначальных требований до анализа, проектирования, кодирования и тестирования программного продукта.

Итоговым результатом работы системы SoDA является готовый документ (или книга). Документ может храниться в файле формата SoDA (FrameBuilder), который получается в результате генерации документа. Вывод на печать этого документа (или его части) возможен из системы SoDA.

Среда функционирования SoDA – ОС типа UNIX на рабочих станциях Sun SPARCstation, IBM RISC System/6000 или Hewlett Packard HP 9000 700/800.

SoDA требует по крайней мере 32 MB оперативной памяти, 100-300 MB для установки и 64 MB рабочего пространства

3. ПОРЯДОК ВЫПОЛНЕНИЯ РАБОТЫ

1. Ознакомится с теоретическими положениями.
2. Запустить систему RatinalSoda.
3. Создать пример документации.

4. КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Для чего нужны средства автоматизации документирования.
2. Особенности системы Ratinal SoDA.
3. Что является итоговым результатом работы системы SoDA.
4. Из чего состоит пакет для подготовки шаблонов документов.
5. Какие средства используются для формирования отчетов в системе Ratinal SoDA

РАБОТА №13 ВЫПОЛНЕНИЕ ОБСЛУЖИВАНИЯ ИНФОРМАЦИОННОЙ СИСТЕМЫ В СООТВЕТСТВИИ С ПОЛЬЗОВАТЕЛЬСКОЙ ДОКУМЕНТАЦИЕЙ

1. ЦЕЛЬ И ЗАДАЧИ РАБОТЫ

Ознакомление с процедурой составления пользовательской (эксплуатационной) документации к программному продукту.

2. КРАТКИЕ ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

Эксплуатационная документация должна обеспечивать отчуждаемость ПС от их первичных разработчиков, адекватно отражать требуемое внешнее качество и качество в использовании, а также возможность освоения и эффективного применения ПС достаточно квалифицированными специалистами. Она применяется непосредственными пользователями в соответствии с функциональным назначением ПС, а также заказчиками, покупателями и поставщиками программных продуктов. Состав этой документации формируется с использованием части технологических документов с учетом требований заказчиков или потенциальных пользователей ПС. Содержание эксплуатационных документов должно предотвращать или исключать возможность некорректного использования комплекса программ пользователями за пределами условий эксплуатации, при которых поставщиком гарантируются требуемые и утвержденные характеристики качества функционирования ПС. При формировании эксплуатационных документов ПС, кроме базовых стандартов жизненного цикла могут использоваться ряд ведомственных нормативных документов и фирменных руководств.

Эксплуатационная документация включает в себя:

- документация администрирования при применении ПС;
- документация операторов-пользователей при применении программного средства;
- документация обучения специалистов применению ПС.

Документация администрирования при эксплуатации информационной системы должна обеспечивать поддержку первичной инсталляции, штатного функционирования и восстановления программ и данных после сбоев и отказов. Управляющая деятельность админи-

стратора состоит в манипулировании управляемыми объектами и должна описываться, анализироваться и регламентироваться совокупностью требований и документов. Для этого необходима полная документация о компонентах информационной системы (компьютерах, сетевых устройствах), которые имеют свои особенности в управлении с помощью специальных программных компонентов, поддерживающих администрирование и управление системой.

К основным функциям системы администрирования, документы для которых подлежат разработке и оцениванию, относятся:

- планирование использования памяти и производительности вычислительной системы в рабочем режиме применения ПС, оперативное управление и распределение ресурсов информационной системы;
- инсталляция и генерация рабочих версий ПС для оперативных пользователей;
- управление и учет внешней среды при выполнении адаптации и реконфигурации конкретного ПС;
- выявление, регистрация и накопление данных о сбоях и дефектах функционирования программ и данных;
- управление средствами защиты информации и санкционированного доступа пользователей, анализ попыток взлома системы защиты, восстановление программ и информации баз данных при искажениях;
- сбор и обобщение статистики о качестве функционирования ПС в составе системы обработки информации.

Документация операторов-пользователей должна обеспечивать корректную и квалифицированную эксплуатацию комплекса программ во всем диапазоне его характеристик, предписанных требованиями заказчика и зафиксированных метриками в использовании. Объектами разработки и оценивания являются документы на процедуры и компоненты интерфейса с внешней средой и с пользователями, определяющие инициализацию соответствующих операций, их ход и результаты, а также комфортность их выполнения. Должно быть предусмотрено достаточное качество идентификации ошибочных действий и ситуаций, а также стандартизированной формы сообщений об ошибках пользователей.

Приобретение, поставка, разработка, функционирование и сопровождение программных средств в значительной степени зависит

от квалификации специалистов. Поэтому эксплуатационной документацией обязательно должно поддерживаться эффективное обучение персонала с целью его подготовки к приобретению, поставке, применению и сопровождению программного средства. Процесс обучения специалистов, контроль и учет результатов обучения с оцениванием достигнутой ими квалификации должен гарантировать, что соответствующие категории обученного персонала готовы для выполнения запланированных действий и решения задач с определенным программным средством.

3. ПОРЯДОК ВЫПОЛНЕНИЯ РАБОТЫ

Результатом выполнения данной работы является пользовательская документация для системы, рассмотренной в предшествующей работе.

4. КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Назначение пользовательской документации.
2. Состав и содержание эксплуатационных документов.
3. Основные функции системы администрирования, для которой необходимы эксплуатационные документы.
4. Приобретение, поставка, разработка, функционирование и сопровождение программных средств
5. Особенности составления документации для операторов-пользователей

РАБОТА №14 ПРАКТИЧЕСКАЯ. РЕИНЖИНИРИНГ БИЗНЕС-ПРОЦЕССА МЕТОДОМ ИНТЕГРАЦИИ

1. ЦЕЛЬ И ЗАДАЧИ РАБОТЫ

Целью работы является получение практических навыков реинжиниринга бизнес-процессов.

2. КРАТКИЕ ТЕОРЕТИЧЕСКИЕ ПОЛОЖЕНИЯ

Реинжиниринг (reengineering) – перепроектирование, перестройка, реконструкция, реорганизация. Реинжиниринг используется периодически для радикальных преобразований, обеспечивающих существенное повышение эффективности (улучшение показателей деятельности компании в десятки раз). А в период между этими «скачками» применяются методы постепенного улучшения для «настройки» усовершенствованных процессов (технологии управления качеством)).

Причины, когда возникает необходимость реинжиниринга:

- диверсификация товаров и услуг (ориентация на различные сегменты рынка), вызывающая многообразие бизнес-процессов;
- изменение сектора рынка (переход от массового производства к работе по индивидуальным заказам, требующая высокую степень адаптации базового бизнес-процесса к потребностям клиента);
- внедрение новых технологий (инновационных проектов), затрагивающих все основные бизнес-процессы предприятия;
- расширение кооперативных связей с партнерами предприятия и поставщиками материалов, обусловливающих альтернативность построения бизнес-процесса.

Принципы БИ.

Важнейшими принципами инжиниринга(реинжиниринга) бизнес-процессов являются:

- «интеграция» соединение отдельных бизнес процессов в единый бизнес процесс, с точки зрения управления, представления.
- «горизонтальное сжатие процесса» – несколько рабочих процедур объединяются в одну;
- «вертикальное сжатие процесса» – сплющивание вертикали;

• «распараллеленность процесса» – шаги процесса выполняются в естественном порядке, работа выполняется в том месте, где это целесообразно.

Важным акцентом в принципах реинжиниринга является использование новых информационных технологий. Информационные технологии (ИТ) играют критически важную роль в BPR. Они рассматриваются как важная составная часть производственных процессов. Но реинжиниринг – это не то же самое что автоматизация, при которой с помощью ИТ автоматизируются существующие бизнес-процессы со всеми их недостатками. Реинжиниринг использует ИТ для автоматизации новых, реконструированных процессов. Более того, сама реконструкция бизнес-процессов, как правило, становится возможной только при использовании информационных технологий, так, как ИТ зачастую меняют сущность процессов.

3. ПОРЯДОК ВЫПОЛНЕНИЯ РАБОТЫ

1. Построить в одной из нотаций диаграмму «as-is» отображающую исходное содержание заданного бизнес-процесса.
2. Выделить деятельности (процессы второго уровня) интеграция которых возможна.
3. Сформировать интегрированные деятельности, на основе выявленных групп интегрируемых процессов. Описать данные процессы.
4. Построить диаграмму «to-be» отображающую полученное содержания рассматриваемого бизнес-процесса.

4. КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Определение бизнес-процесса.
2. Что такое реинжиниринг, когда он выполняется?
3. Какие принципы реинжиниринга выделяются?
4. В чём состоит принцип интеграции?

РАБОТА №15 ЛАБОРАТОРНАЯ. ФОРМИРОВАНИЕ ПРЕДЛОЖЕНИЙ ПО РЕИНЖИНИРИНГУ ИНФОРМАЦИОННОЙ СИСТЕМЫ «ОРГАНАЙЗЕР»

1. ЦЕЛЬ И ЗАДАЧИ РАБОТЫ

Изучить основные этапы и основы проектирования реинжиниринга информационных систем.

2. КРАТКИЕ ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

В рамках процесса реинжиниринга ИС (в независимости от методологии) принято выделять следующие наиболее существенные этапы:

- формирование команды реинжиниринга;
- сбор претензий к системе;
- создание спецификации требований к системе;
- актуализация структурных моделей системы;
- генерация альтернатив реинжиниринга системы;
- выбор оптимальной альтернативы;
- реализации выбранной альтернативы.

Одним из наиболее важных этапов является формирование команды реинжиниринга ИС. Команда обязательно должна иметь лидера, который будет принимать стратегические решения, и координатора, который будет организовывать их реализацию. Также в команду должны входить: специалисты по информационным технологиям вообще, специалисты по ИТ в этой ИС, разработчики, представители групп пользователей предприятия заказчика. Последние будут отражать интересы большинства пользователей информационной системы, а также будут способствовать внедрению обновленной системы на предприятии. Пользователи сопротивляются нововведениям и поэтому необходима их моральная подготовка, которую удобнее провести при посредничестве таких представителей групп пользователей.

Рекомендуется также включать в команду реинжиниринга независимых экспертов. Численный состав команды рекомендуется ограничивать 10 членами. При необходимости сбора более многочисленной команды, необходимо выделить «ядро» (не более 10 че-

ловек) и «окружение». Члены команды РИС со стороны предприятия заказчика должны быть временно освобождены от должностных обязанностей. Перед началом проекта реинжиниринга информационной системы необходимо получить поддержку руководства предприятия. Без выделения ресурсов (своевременного и в необходимом объеме) реинжиниринг обречен на провал. При этом часто возникает задача обоснования для руководства предприятия необходимости именно реинжиниринга информационной системы, а также невозможности ограничиться только отдельными исправлениями в информационной системе. Решения команды РИС должны иметь статус приказа руководителя предприятия. Всем членам команды реинжиниринга необходимо разъяснить цели и задачи проекта, его особенности и ограничения.

На следующем этапе проводится сбор и обработка претензий пользователей информационной системы. От конечных пользователей можно получить конкретные замечания по функционированию ИС, а от руководства – пожелания в плане стратегического развития информационной системы. Следует помнить, что эти замечания еще не являются требованиями к системе, а служат лишь симптомами ее несовершенства.

Пользователи и руководство могут сообщить лишь о видимых проявлениях. В основном, они сводятся к замечаниям по удобству пользовательского интерфейса информационной системы, быстродействию и полноте реализации отдельных функций системы. Кроме того, обычно ни пользователи, ни руководство не видят в целом, в комплексе проблему с информационной системой. Процедур сбора претензий на сегодняшний день существует достаточно много (анкетирование, интервьюирование, наблюдение за работой и т. п.). Следует отметить, что обычно пользователи занимают пассивную позицию и необходимо их дополнительно стимулировать. Собранные претензии могут противоречить друг другу. Нужно установить их приоритеты и обоснованность. Поэтому перед составлением спецификации требований необходима обработка собранных претензий.

Разработчики должны выяснить, что лежит в основе этих претензий, какие глубинные недостатки информационной системы их порождают. Составление спецификации требований можно проводить по известным методикам, например.

Спецификация для реинжиниринга информационной системы должна позволить с одной стороны сохранить общее назначение системы, а с другой позволить существенно развить систему, не потеряв этого назначения. При составлении спецификации рекомендуется использовать требования, сформулированные в техническом задании, по которому была разработана существующая информационная система.

Требования можно разделить на два типа: функциональные и нефункциональные. К функциональным требованиям относятся:

- требования к бизнес-функциям (высокоуровневые цели предприятия или заказчика информационной системы);
- требования к целям и задачам информатизации (цели и задачи, которые помогает решать информационная система);
- требования к функциям информационной системы (что необходимо реализовать);
- системные требования (могут включать также требования к квалификации персонала).
- К нефункциональным относятся:
- бизнес-правила (корпоративная политика, постановления правительства, стандарты и т. п.);
- атрибуты качества информационной системы (характеристики, важные для пользователей и разработчиков);
- внешний интерфейс (не только пользовательский, но и программный);
- ограничения (дополнительные требования).

Очевидно, для перепроектирования информационной системы требуются ее структурные модели (функциональная, информационная, архитектурная, объектно-ориентированная и т. п.). Причем, в актуальном состоянии, т. е. модели, описывающие информационную систему в том виде и в том состоянии, в котором она существует и эксплуатируется. Однако часто приходится сталкиваться с отсутствием таких структурных моделей или их несоответствием системе. Поэтому, перед перепроектированием требуется актуализация структурных моделей информационной системы. Такое расхождение между информационной системой и описывающими ее структурными моделями обычно вызвано некачественной работой исполнителей. Любые изменения в информационной системе, прежде всего, должны быть отражены в ее структурных моделях.

Можно выделить следующие причины, являющиеся типовыми для такой ситуации:

Разработчики, ссылаясь на острую нехватку времени, корректируют только саму информационную систему, оставляя корректировку структурных моделей «на потом». В действительности, это – некачественная работа, ведущая к проблемам при реинжиниринге системы.

Изменения информационной системы намеренно не отражаются в ее структурных моделях (по разным мотивам, например, «модели нужны были для создания системы, а теперь она живет своей жизнью»). Это в корне неверно. Структурные модели теряются («пропадают») или уничтожаются (не обязательно по злому умыслу, возможно в результате нештатной ситуации). Такие ситуации необходимо исключить (например, путем регулярного резервного копирования моделей). Разработчики «уносят с собой» структурные модели информационной системы (обычно при увольнении). Здесь –ошибка руководителя проекта и кадровой службы предприятия.

До сих пор встречаются и такие проекты, в ходе которых структурные модели информационной системы не строятся вовсе (хранятся «в головах разработчиков», рисуются на бумаге, которая позже теряется). В этом случае приходится проводить сложный анализ информационной системы с целью восстановления ее структурной модели. Очень важную роль играют комментарии разработчиков к структурным моделям (как на диаграммах, так и в виде вспомогательного текста), а также комментарии в текстах программ. Поскольку зачастую разработчикам сложно разобраться даже в собственных текстах, написанных какое-то время назад.

Современные CASE-средства (такие как Rational Rose, Altova UModel, AllFusion ERWin Data Modeler, AllFusion Business Process Modeler) предоставляют не только широкие возможности построения структурных моделей информационной системы, но и некоторые функции для восстановления структурных моделей из исходных текстов (Rational Rose) или базы данных (AllFusion ERWin Data Modeler) распространенной СУБД. Кроме того, в некоторые современные средства разработки уже включены инструменты для поддержки реинжиниринга (например, в составе Borland Delphi 2009,

имеется инструмент для рефакторинга программного обеспечения информационной системы).

Генерация альтернатив (вариантов) реинжиниринга является наименее формализуемой (на сегодняшний день) операцией, требующей творческого подхода. Распространение получили следующие способы:

- генерация альтернатив реинжиниринга ИС в целом;
- генерация и комбинирование частных вариантов;
- использование шаблонов.

Первый способ требует, чтобы альтернатива описывала реинжиниринг информационной системы в целом, учитывая все требования к системе. Как правило, удается разработать небольшое количество таких альтернатив. Это способ требует привлечения разработчиков высшей квалификации, способных охватить информационную систему в целом («одним взглядом»). Разработать такую альтернативу крайне сложно, поскольку требуется не только учитывать все требования к информационной системе, но и представлять себе систему целиком.

Второй способ генерации базируется на комбинировании частных вариантов разрешения требований к информационной системе. Он основан на методе морфологического ящика. Для каждого требования разрабатывается максимальное количество разных частных вариантов его разрешения. При этом альтернатива реинжиниринга ИС в целом представляет собой такое подмножество частных вариантов, которое разрешает все требования к информационной системе.

Генерация вариантов для отдельных требований – более простая задача, чем генерация вариантов для всей ИС сразу. Комбинирование вариантов может быть автоматизировано.

Третий способ является промежуточным между первым и вторым. Он предполагает выбор в качестве основы (шаблона) некоторой готовой альтернативы и последующей коррекции ее для условий конкретного проекта реинжиниринга конкретной информационной системы. Шаблон может описывать общую концепцию реинжиниринга информационной системы на общем уровне и требовать конкретизации. Как вариант, шаблон может также вполне конкретно описывать реинжиниринг основной части информационной системы (например, ее базовых функций) и требовать дополнения

для разрешения второстепенных требований к ИС. Такой способ требует наличия «базы» шаблонных решений. Выбор оптимальной альтернативы представляет собой решение многокритериальной задачи принятия решения в условиях риска9. Для этого необходимо сначала определить критерии оптимальности, а затем оценить альтернативы (сначала частные варианты, если альтернативы генерируются по второму способу).

Сложность выбора заключается в том, что альтернативы могу быть несравнимы. Чаще всего оценка проводится по таким показателям, как:

- стоимость реализации альтернативы;
- степень разрешения требования (ожидаемый эффект);
- сложность реализации альтернативы;
- время (длительность) реализации альтернативы.

Для получения таких оценок приходится прибегать к экспертным методам. Статистику использовать трудно, поскольку условия выполнения проектов (и сами информационные системы) сильно различаются. Риск заключается в том, что значения показателей могут отклоняться от запланированных. Моделировать риск удобно путем описания показателей случайными величинами с некоторым законом распределения.

Поскольку альтернатива представляет собой некоторое подобие комплекса работ, можно пользоваться β-распределением. Это позволяет потребовать от экспертов всего по двух оценок: минимального и максимального возможных значений показателя. При использовании второго способа генерации альтернатив показатели самих альтернатив можно рассчитывать формальными методами на основе показателей частных вариантов.

Часто руководствуются следующими «граничными» альтернативами:

- альтернатива с минимальной стоимостью;
- альтернатива с минимальной сложностью реализации;
- альтернатива с максимальным разрешением требований;
- альтернатива с минимальным временем реализации.

Скорее всего, ни одна из перечисленных альтернатив не будет реализована, но они служат для предварительной оценки параметров проекта реинжиниринга информационной системы. Получение оценки длительности реализации альтернативы является одной из

наиболее сложных задач, поскольку один и тот же набор работ может быть выполнен по разным графикам (сетевым графикам).

Показатель длительности (в отличие от показателя стоимости, например) не позволяет выполнять операцию сложения. Вообще, оценка длительности работы затруднительна сама по себе. Однако в настоящее время затраты на приобретение оборудования относительно невелики (по разным оценкам, порядка 10%). Поэтому можно считать, что основные затраты – на зарплату разработчиков. Зная их квалификацию и почасовую ставку, можно грубо оценить продолжительность работы.

Для реализации выбранной альтернативы необходимо составить технический проект, подробно описывающий необходимые технические решения. Это связано с тем, что альтернатива представляет собой опорный вариант, описывающий по большей части концепцию будущего решения. При реализации альтернативы приходится сталкиваться с целым рядом трудностей, таких как: необходимость перехода со старой информационной системы на новую (включая переустановку программного обеспечения, конвертирование и перенос данных и т. п.); необходимость обучения (переобучения) пользователей; необходимость подготовки окружения информационной системы (бизнес-процессов, смежных систем и т. п.); необходимость поддержки двух версий информационной системы во время перехода со старой версии на новую. Для успешного перехода необходимо учесть режим эксплуатации информационной системы.

3. ПОРЯДОК ВЫПОЛНЕНИЯ РАБОТЫ

Рассмотреть основные этапы и основы проектирования реинжиниринга информационных систем на примере информационной системы «Органайзер»

4. КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Основные этапы реинжиниринга ИС.
2. Выбор, состав и влияние команды реинжиниринга на работу проекта.
3. Значение сбора и обработки претензий пользователей информационной системы.

4. Спецификация для реинжиниринга информационной системы.

5. Функциональные и нефункциональные требования спецификации ИС.

6. Структурные модели, необходимые для перепроектирования информационной системы.

7. Современные CASE-средства для построения структурных моделей информационной системы.

8. Типы вариантов альтернатив реинжиниринга ИС.

СОДЕРЖАНИЕ САМОСТОЯТЕЛЬНОЙ РАБОТЫ

Цель самостоятельной работы обучающихся – получить новые знания по дисциплине «Устройство и функционирование информационной системы».

Самостоятельная работа необходима для формирования у обучающихся способности самостоятельно решать задачи профессиональной деятельности, формирования умения и навыков планирования времени, формирования стремления развиваться и совершенствоваться.

Виды самостоятельной работы обучающихся указаны в табл. 1.

Таблица 1

Виды самостоятельной работы

№ п/п	Вид СРС
1	Исследование средств управления проектами.
2	Построение модели управления качеством процесса изучения модуля «Устройство и функционирование информационной системы».
3	Разработка руководства по инсталляции программного продукта для ИС.
4	Разработка сетевого графика проекта реализации ИС

УЧЕБНО-МЕТОДИЧЕСКИЕ МАТЕРИАЛЫ ПО ДИСЦИПЛИНЕ

Основная литература

1. Федорова, Г, Н. Разработка, внедрение и адаптация программного обеспечения отраслевой направленности. – Москва: НИЦ ИНФРА-М, 2019. – 336 с. – Режим доступа: http://znanium.com/go.php?id=989682. – Загл. с экрана.

Дополнительная литература

1. Рудаков, А. В. Технология разработки программных продуктов [Электронный ресурс]: учебник для студентов учреждений среднего профессионального образования, обучающихся по специальности «Программное обеспечение вычислительной техники и автоматизированных систем»: [профессиональный модуль ПМ.03 «Участие в интеграции программных модулей» (МДК.03.01)] / А. В. Рудаков. – Москва: Академия, 2017. – 208 с. – Режим доступа: http://www.academia-moscow.ru/catalogue/4831/362819/. – Загл. с экрана.

2. Казанский, А. А. Программирование на visual c# 2013. [электронный ресурс]. – Москва: Юрайт, 2018. – 191 с. – Режим доступа: https://biblio-online.ru/book/programmirovanie-na-visual-c-2013-414752. – Загл. с экрана.

Программное обеспечение и интернет-ресурсы

1. https://www.intuit.ru/studies/courses/497/353/info – Учебный курс Введение в программную инженерию; https://www.intuit.ru/studies/courses/532/388/info Учебный курс Управление развитием ИС

2. https://iiba.ru/category/overview-of-all-methods-of-work-with-requirements/ обзор методик работы с требованиями http://www.enter-agile.com/2010/01/user-story-primer-intro.html Основы пользовательских историй

СОДЕРЖАНИЕ