

МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение
высшего образования
**«Московский государственный университет
информационных технологий, радиотехники и электроники»**
МИРЭА

ИКБСП **Направление 09.03.02**

Кафедра КБ-4 **Квалификация БАКАЛАВР**

Тема диплома:

Разработка информационной системы метрологического учета средств измерений

Выполнил: Брильков Александр Анатольевич

Оглавление

ВВЕДЕНИЕ……..3
1. ИССЛЕДОВАТЕЛЬСКИЙ РАЗДЕЛ……..5
1.1. Характеристика объекта исследования……..5
1.2. Экономическая сущность задачи……6
1.3. Обоснование необходимости и цели использования вычислительной техники для решения задачи……8
1.4. Постановка задачи……..14
1.5. Обзор существующих разработок и обоснование выбора технологии проектирования……..15
2. СПЕЦИАЛЬНЫЙ РАЗДЕЛ……..20
2.1. Выбор средств разработки……20
2.2. Характеристика базы данных……26
2.3. Системы классификации, применяемые в ИС метролога……..33
2.4. Разработка проекта автоматизации……..38
2.5. Ожидаемые риски и их описание……49
3. ТЕХНОЛОГИЧЕСКИЙ РАЗДЕЛ……52
3.1. Характеристика входной и выходной информации……52
3.2. Программное обеспечение задачи (комплекса задач, ИП)……56
3.3. Руководство пользователя……63
4. ОБОСНОВАНИЕ ЭКОНОМИЧЕСКОЙ ЭФФЕКТИВНОСТИ ПРОЕКТА…….68
4.1. Выбор и обоснование методики расчёта экономической эффективности……68
4.2. Расчёт показателей экономической эффективности проекта……72
ЗАКЛЮЧЕНИЕ……..78
СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ……..80
ПРИЛОЖЕНИЕ. ЛИСТИНГ ПРОГРАММНЫХ МОДУЛЕЙ……...83

ВВЕДЕНИЕ

Автоматизация учета и контроля метрологического обеспечения средств измерений в настоящее время является одним из главных вопросов решаемых метрологическими службами предприятий.

Основная задача метрологической службы заключается в учете средств измерений, планировании и контроле работ по обеспечению единства измерений согласно требованиям Федерального закона от 26 июня 2008 г. N 102-ФЗ «Об обеспечении единства измерений».

Выполнение вышеуказанной задачи представляет собой рутинную работу с большими перечнями средств измерений. Метрологическая служба постоянно контролирует состояние технических средств, своевременно выполняя их поверку и калибровку, составляя годовые графики и месячные планы работ, предоставляя отчетность вышестоящему руководству. Учитывая, что количество технических средств постоянно меняется – вводятся в эксплуатацию новые и списываются отработавшие свой ресурс – вести качественный учет средств измерений без использования автоматизации достаточно сложно. Поэтому основным аспектом эффективного управления большим количеством средств измерений является информационная осведомлённость.

Так как результаты поверок и калибровок нуждаются в учете, со временем насущным вопросов в организациях стала автоматизация метрологического учета. Именно этому вопросу и посвящена данная дипломная работа.

Объектом рассмотрения дипломной работы является метрологический отдел, предметом – деятельность сотрудников метрологического отдела по метрологическому учету.

Целью работы является разработка информационной системы метрологического учёта средств измерений.

Задачами дипломной работы являются:

- дать характеристику объекта управления;

- выбор комплекса задач автоматизации;
- спроектировать и создать базу данных;
- разработать и реализовать программное средство обработки данных;
- описание разработанной информационной системы;
- расчет экономической эффективности.

1 ИССЛЕДОВАТЕЛЬСКИЙ РАЗДЕЛ

1.1 Характеристика объекта исследования

Объектом исследования является отдел метрологии. На рисунке 1.1 представлена схема организационной структуры отдела метрологии.

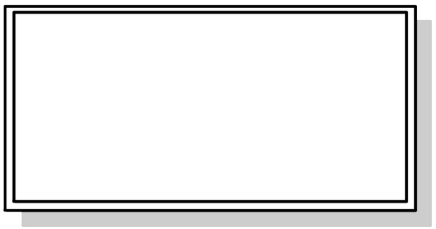


Рисунок ИССЛЕДОВАТЕЛЬСКИЙ РАЗДЕЛ.1 Схема организационной структуры отдела метрологии

Свою деятельность отдел осуществляет в соответствии с планом работы, утверждённым директором предприятия:

- ведёт учёт средств измерений (СИ) и нормативной документации (НД);
- пополняет и актуализирует фонд нормативных документов;
- обеспечивает калибровку и поверку СИ;
- организует работу по нормативному и метрологическому обеспечению единства и требуемой точности диагностических исследований;
- координирует работу подразделений предприятия по нормативному и метрологическому обеспечению измерений;
- осуществляет контроль за материально-техническим оснащением подразделений, использованием и состоянием СИ, применением стандартных образцов;

- составляет планы работ и отчётов по итогам работы отдела;
- оформляет договора на выполнение работ, связанных с функциями отдела.

Начальник отдела метрологии:

- Осуществляет руководство деятельностью отдела метрологии.
- Определяет функции и распределяет обязанности между сотрудниками отдела метрологии в пределах утвержденных на них должностных инструкций.
- Дает указания, обязательные для исполнения сотрудниками отдела метрологии.
- Вносит на рассмотрение руководства вопросы, отнесенные к компетенции отдела метрологии.
- Подписывает служебную документацию в пределах своей компетенции.
- В рамках деятельности отдела метрологии координирует работу и оказывает методическую помощь структурным подразделениям предприятия.
- Обеспечивает взаимодействие отдела метрологии со структурными подразделениями предприятия и привлекает к работе специалистов других структурных подразделений предприятия по согласованию с их руководителями.
- Способствует созданию необходимых условий труда и благоприятного морально-психологического климата в коллективе. Обеспечивает соблюдение работниками отдела метрологии трудовой дисциплины.

1.2 Экономическая сущность задачи

Основным предназначением метрологического отдела является проведение калибровок и поверок средств измерения полигона. Так как рассматриваемое предприятие является испытательным полигоном, то в силу специфики выполняемой деятельности на полигоне используется большое количество различного измерительного оборудования. От того, насколько точно проводятся различные измерения, зависит результат деятельности всего полигона и

проводимых им испытаний, направление дальнейшего развития испытываемой техники. То есть выполняемые отделом задачи являются очень важными и значимыми для всего полигона.

Порядок проведения калибровок и поверок определен законодательством РФ.

Калибровка средств измерений (калибровочные работы) - совокупность операций, выполняемых с целью определения и подтверждения действительных значений метрологических характеристик и (или) пригодности к применению средства измерений, не подлежащего государственному метрологическому контролю и надзору.

Калибровка средств измерений производится в соответствии с нормативными документами, регламентирующими проведение калибровочных (поверочных) работ. Средства измерения представляются на калибровку в сроки, установленные графиками калибровки. Положительные результаты калибровки средств измерений действительны в течение межкалибровочного интервала.

Поверка средств измерений - совокупность операций, выполняемых отделом метрологии с целью, определения и подтверждения соответствия средств измерений установленным техническим требованиям.

Результатом поверки является подтверждение пригодности средства измерений к применению или признание средства измерений непригодным к применению. Если средство измерений по результатам поверки признано пригодным к применению, то на него или техническую документацию наносится оттиск поверительного клейма или выдается "Свидетельство о поверке".

Если средство измерений по результатам поверки признано непригодным к применению, оттиск поверительного клейма гасится, "Свидетельство о поверке" аннулируется, выписывается "Извещение о непригодности" или делается соответствующая запись в технической документации.

СИ подвергаются первичной, периодической, внеочередной, инспекционной и экспертной поверке.

Первичной поверке подлежат СИ при выпуске из производства и ремонта, при ввозе по импорту. Первичной поверке подлежит, как правило, каждый экземпляр

СИ, но допускается и проведение выборочной поверки.

Периодической поверке подлежат СИ, находящиеся в эксплуатации или на хранении, через определенный межповерочный интервал. Периодическую поверку должен проходить каждый экземпляр СИ.

Периодическую поверку СИ, предназначенных для измерения (воспроизведения) нескольких величин или имеющих несколько диапазонов измерений, но используемых для измерения (воспроизведения) меньшего числа величин или на меньшем числе диапазонов измерений допускается на основании решения главного метролога производить только по тем требованиям нормативных документов по поверке, которые определяют пригодность СИ для применяемого числа величин и применяемых диапазонов измерений. Соответствующая запись должна быть сделана в эксплуатационных документах.

Первый межповерочный интервал устанавливается при утверждении типа СИ. Отдел метрологии обязан вести учет результатов периодической поверки и разрабатывать рекомендации по корректировке межповерочных интервалов.

Экономической сущностью разработки подсистемы метрологического учета является снижение экономических затрат (трудовых и стоимостных) по учету проведения калибровок и поверок за счет автоматизации данного процесса.

1.3 Обоснование необходимости и цели использования вычислительной техники для решения задачи

В результате деятельности отдела метрологии в области осуществления поверок и калибровок скапливается большое количество документации, протоколов и других форм документов, каждый из которых содержит необходимую информацию.

На основании данной информации необходимо строить регулярные отчеты, которые содержат сведения о количестве проведенных поверок и калибровок, их результатах, количестве поверенного и откалиброванного измерительного

оборудования с разбивкой по типам СИ, подразделениям, периодам поверок и калибровок.

Рассмотрим существующую технологию работы метролога метрологического отдела.

Для анализа существующей модели технологии деятельности метролога разработаем модель «Как есть», с использованием методологии SADT и программного средства AllFusion Process Modeler r7. Данное программное средство является развитием BPWin.

Модель «Как есть» позволяет выяснить, «что мы делаем сегодня» перед тем, как «перепрыгнуть» на то, «что мы будем делать завтра». Анализ функциональной модели позволяет понять, где находятся наиболее слабые места, в чем будут состоять преимущества новых бизнес-процессов и насколько глубоким изменениям подвергнется существующая структура организации бизнеса. Детализация бизнес-процессов позволяет выявить недостатки организации даже там, где функциональность на первый взгляд кажется очевидной. Признаками неэффективной деятельности могут быть бесполезные, неуправляемые и дублирующиеся работы, неэффективный документооборот (нужный документ не оказывается в нужном месте в нужное время), отсутствие обратных связей по управлению (на проведение работы не оказывает влияния ее результат), входу (объекты или информация используются нерационально).

Контекстная диаграмма является вершиной древовидной структуры диаграмм и представляет собой самое общее описание системы и ее взаимодействия с внешней средой.

На рисунке 1.2 приведена диаграмма, описывающая верхний уровень процесса деятельности метролога предприятия.



Рисунок ИССЛЕДОВАТЕЛЬСКИЙ РАЗДЕЛ.2 Характеристика деятельности метролога предприятия

Декомпозиция данного процесса приведена на рисунке 1.3.

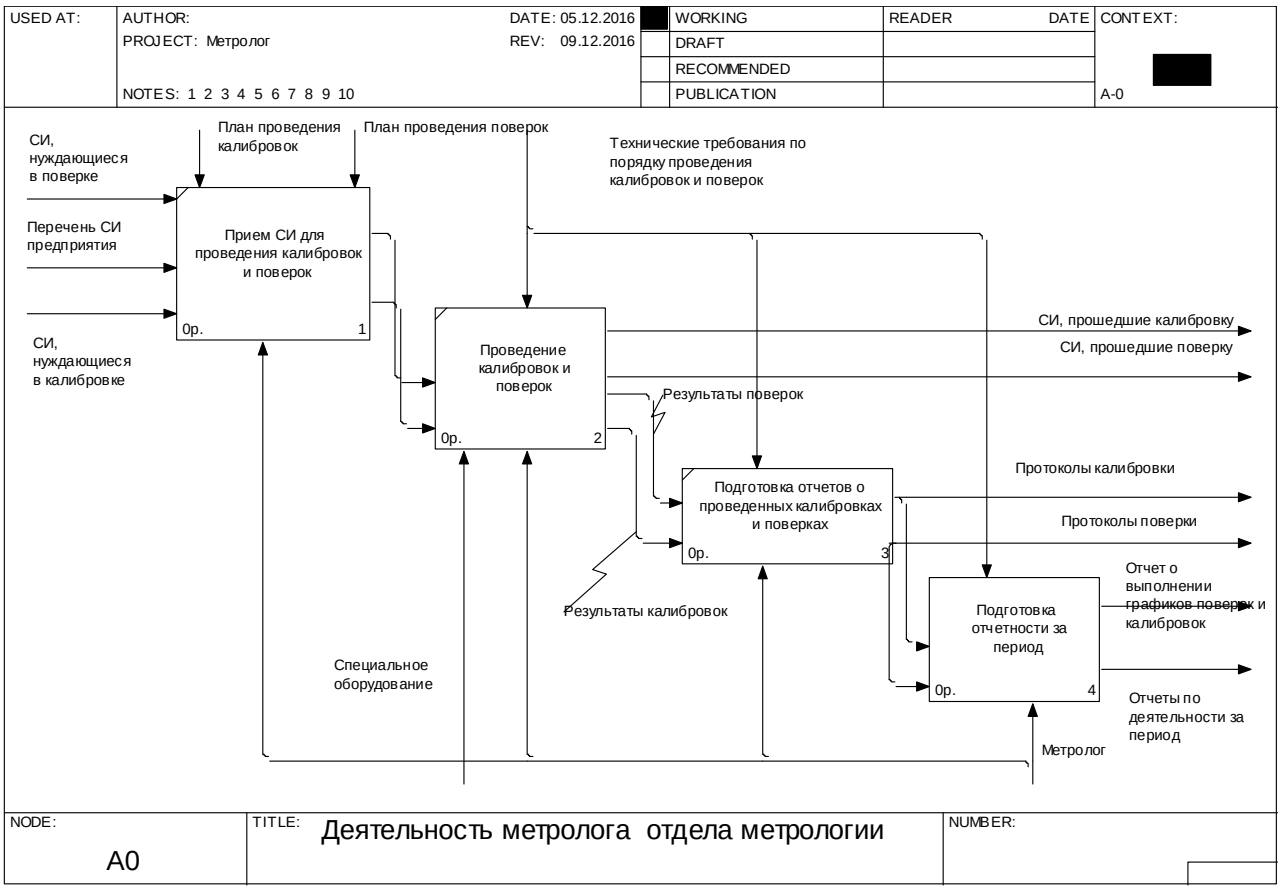


Рисунок ИССЛЕДОВАТЕЛЬСКИЙ РАЗДЕЛ.3 Декомпозиция процесса деятельности метролога

В соответствии с приведенной диаграммой, метролог осуществляет свою деятельность в следующем порядке:

- Прием средства измерения для осуществления поверки и калибровки в соответствии с планами проведения калибровок и поверок;
- Непосредственное проведение операция калибровок и поверок;
- Подготовку протоколов по результатам проведения калибровок и поверок;
- Подготовка отчетных документов по итогам выполненных мероприятий.

К числу таких документов относятся:

- Общий отчет о количестве выполненных поверок и калибровок;
- Статистический отчет по результатам проведения калибровок и поверок;

- Отчет о выполнении графиков проведения калибровок и поверок.

С точки зрения рассматриваемой задачи автоматизации наибольший интерес представляет процесс подготовки отчетности. Декомпозиция данного процесса показана на рисунке 1.4.

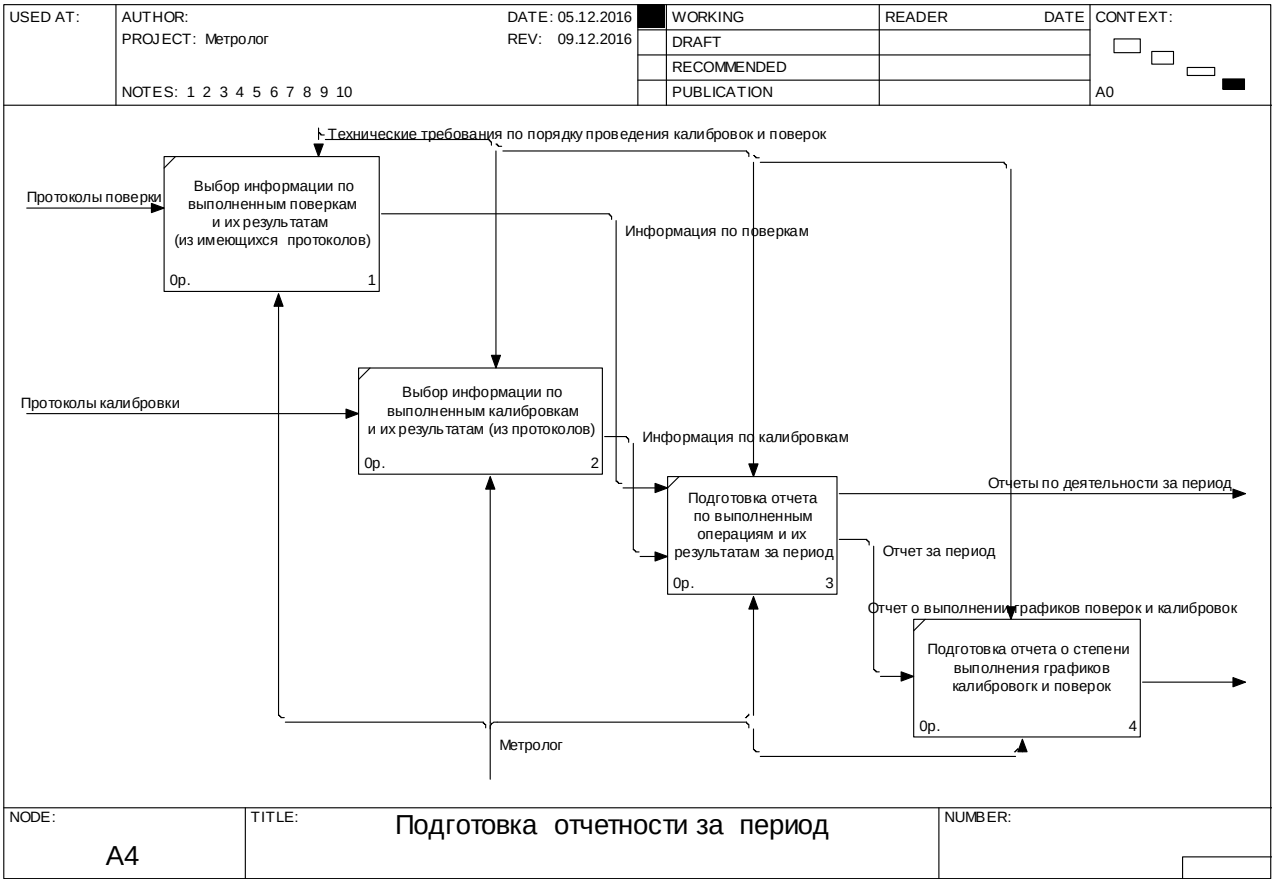


Рисунок ИССЛЕДОВАТЕЛЬСКИЙ РАЗДЕЛ.4 Декомпозиция процесса подготовки отчетности

Процесс подготовки отчетности состоит из следующих подпроцессов:

- Выбор информации из протоколов о проведении калибровок;
- Выбор информации из протоколов о проведении поверок;
- Подготовка отчета по выполненным операциям за период;
- Подготовка отчета о степени выполнения графиков.

Для того, чтобы получить данную информацию, метрологу необходимо просмотреть большое количество протоколов, графиков, планов и других документов. Данные документы хранятся исключительно в бумажной форме, Что затрудняет параметрический поиск информации и замедляет работу метролога. В

связи с этим необходимо автоматизировать учет проведения операций по калибровкам и поверкам, учет протоколов и результатов данных операций, а также составления отчетов.

В настоящее время существующая технология учета выполняемых метрологом операций обладает следующими недостатками:

- Обработка документов в бумажной форме, из чего вытекает большая трудоёмкость выполняемых операций;
- Отсутствие возможности интеграции системы расчета с другими информационными системами, используемыми на предприятии;
- Отсутствие возможности автоматизированного построения отчетов.

Для того, чтобы устранить имеющиеся недостатки, необходимо разработать и внедрить автоматизированную систему метролога метрологического отдела.

Автоматизированная система должна обеспечивать выполнение следующих функций информационной поддержки:

1) сбора исходной информации о средствах измерения;
2) ввода информации о выполненных калибровках и поверках;
3) формирования информации, необходимой для автоматизации подготовки отчетов.

Автоматизированная система метролога должна обеспечивать выполнение следующих функций ведения нормативно-справочной информации:

1) ведения справочника средств измерения;
2) ведения справочника типов калибровок и поверок;
3) сохранения «истории» проведения поверок и калибровок.

Автоматизированная система метрологадолжна обеспечивать выполнение следующих функций администрирования:

1) обеспечения целостности и доступности хранимой, обрабатываемой и передаваемой информации;
2) архивирования данных;
3) резервного копирования и восстановления данных;

4) восстановления работоспособности после устранения причин, вызвавших аварийную ситуацию;

5) информирования персонала, отвечающего за обслуживание автоматизированной системы метролога, о возникновении аварийной ситуации.

1.4 Постановка задачи

Целью автоматизированного варианта решения задачи является уменьшение трудоёмкости при выполнении метрологом своих обязанностей. Назначением системы является учет проведенных поверок и калибровок с формированием следующие отчетов:

- Отчет по выполненным поверкам;
- Отчет по выполненным калибровкам;
- Отчет за период по выполнению графика калибровок и поверок;
- Список средств измерения со сроками проведения калибровок и поверок.

Кроме того, внедрение разработки системы позволит достичь:

- сокращение времени обработки входной информации и получения результатных показателей;
- повышение степени достоверности выходной информации;
- снижение трудоемкости и количества ошибок при решении задачи.

В программе должны быть реализованы следующие функции:

- автоматизация ввода показателей из первичных документов;
- хранение форм первичных документов;
- хранение и выдача нормативно-справочной информации по запросу пользователя;
- обеспечение целостности хранимой информации.

Выполнение перечисленных функций будет осуществлено следующим образом:

1. Автоматизация ввода, контроля, загрузки данных первичной информации в базу данных с использованием экранных форм:

- окно просмотра и изменения в списках средств измерения;
- окно создания общего списка СИ;
- окно формирования протоколов калибровок и поверок;
- окно учета проведения калибровок и поверок;
- формирования отчетной документации;

2. Ведение условно-постоянной информации, хранящейся в таблицах базы данных:

- редактирование справочных и учетно-оперативных данных;
- добавление учетных записей;
- удалении учетных записей;

3. Выдача справочной информации:

- информация по СИ;
- информация по срокам поверок и калибровок.

4. Отчеты на запросы пользователей:

- формирование итоговой отчетности проведенным операциям;
- формирование оперативных отчетов для метролога.

1.5 Обзор существующих разработок и обоснование выбора технологии проектирования

Для того, чтобы определить, можно ли использовать для автоматизации задачи уже имеющееся на рынке программное обеспечение, а именно:

- Softlogic.Метрология
- Автоматизированная система управления метрологической службой (АСУ МС) версии 6.1
- Программный комплекс "Дельта-СИ"
- Автоматизированное рабочее место метролога - АРМ "Метролог"

Softlogic.Метрология позволяет:

- вести единый журнал приборов;

- учитывать как собственные приборы, так и приборы внешних организаций;
- планировать работы по поверке и калибровке;
- планировать работы по техническому обслуживанию;
- рассчитывать трудочасы персонала и планировать нагрузку;
- автоматически формировать отчеты, сертификаты, хронометражные карты, протоколы;
- подключать внешние измерительные приборы и отслеживать состояние внешних влияющих факторов.

К основным возможностям системы относятся:

- Единый реестр организаций (обслуживание, производитель и т.п.)
- Единый реестр сотрудников (образование, аттестации, допуски, участки)
- Настраиваемая организационная структура и привязка сотрудников к участкам и подразделениям
- Постановка задач и контроль исполнения
- Единый журнал учета средств измерений (настраиваемые типы приборов, состояния, периодов калибровок, поверок и техобслуживания и т.п.)
- Ведение истории работы с прибором, истории выдачи сертификатов и т.п.
- Автоматическое формирование предписаний (предписания об изъятии и т.п.)
- Учет эталонных приборов, нормативных документов, регламентов, требований, кодификаторов, норм времени и т.п.
- Автоматическое формирование планов работ (техническое обслуживание, калибровки, поверки и т.п.)
- Формирование и печать различных документов (графики работ, графики технического обслуживания поверок и калибровок, единый реестр средств измерений, план работ персонала, отчет о выполненных работах)
- Открытый интерфейс и API для доработки и интеграции с внешними устройствами и системами Гибкие возможности для настройки (типы,

этапы, циклы работ, справочники, шаблоны, роли пользователей, доступ к данным, печатные формы и т.п.)

АСУ МС предназначена для автоматизации и управления деятельности метрологических служб предприятий по учету средств измерений (СИ). Система автоматизирует все основные функции метрологической службы: учет СИ, планирование, организацию и контроль выполнения обслуживания, анализ состояния приборного парка и т.д.

Программный комплекс имеет в своем составе модуль «МетрЭкспорт», который прошел испытание на совместимость с АИС "Метрконтроль" и соответствует «Требованиям к совместимости программно-аппаратных средств метрологической службы юридического лица с автоматизированной информационной системой учета результатов поверки Ростехрегулирования (АИС «Метрконтроль»)» от 17 мая 2006г. и «Дополнениям и Изменениям» к ним от 12 декабря 2008г. Таким образом, система обеспечивает обмен данными с АИС «Метрконтроль» и может применяться метрологическими службами, аккредитованными на право поверки, для передачи данных о поверке средств измерений в автоматизированную систему учета поверочной деятельности Федерального агентства по техническому регулированию и метрологии.

В программном комплексе также реализованы такие важнейшие функции как:

- Разграничение прав доступа пользователей к информации о средствах измерений;
- Обмен данными между МСЮЛ, аккредитованного на право поверки, и центральным узлом АИС «Метрконтроль»;
- Штриховое кодирование;
- Учёт клейм, выполненных в виде наклеек;
- Набор готовых стандартных отчетов метрологических работ;
- Экспорт отчетных данных и данных по любым экранным формам в MS Excel, MS Word.

ПК “Дельта-СИ” предназначен для автоматизированного учета состояния средств измерений (СИ) и позволяет автоматизировать следующие функции

метрологических служб:

- составление планов поверки (калибровки) СИ;
- учет аттестации приборов;
- формирование производственных заданий поверителям и ремонтному персоналу;
- архивирование событий в процессе эксплуатации и ремонта СИ;
- составление и печать отчетов;
- получение статистической информации о состоянии СИ;
- мониторинг парка СИ.

Целью АРМ «Метролог» является создание базы данных обо всех СИ используемых на предприятии с возможностью оперативного и полного получения сведений о СИ с автоматическим сведением во всевозможные отчеты.

Функции программы АРМ Метролог:

- Учет наличия, движения и состояния приборов. Планирование любых видов метрологического контроля и ремонта.
- Контроль за проведением метрологического и технического обслуживания.
- Анализ состояния и применения приборного парка.
- Накопление и обработка статистики по отказам, результатам поверок, характеристикам приборов, движению приборов.

Сравнение возможностей рассмотренных систем приведено в таблице 1.1.

Таблица ИССЛЕДОВАТЕЛЬСКИЙ РАЗДЕЛ.1 Сравнение возможностей рассмотренных систем

Возможность	Softlogic.Метрология	АСУ МС	Дельта-СИ	АРМ "Метролог"
Учет парка СИ	Да	Да	Да	Да
составление планов поверки (калибровки) СИ	Да	Да	Да	Да
учет аттестации приборов	Да	Да	Да	Да
получение статистической информации о	Да	Да	Да	Да

состоянии СИ				
Экспорт отчетных данных	нет	Да	Да	нет
Формирование документов для аккредитации МС	нет	нет	да	Да
Контроль соблюдения графиков	Да	Да	Да	да
Стоимость, рублей	250000	280000 (за 5 рабочих мест)	320000	бесплатно

Для рассматриваемой задачи необходима только одна основная функция – учет проведения поверок и калибровок с возможностью составления отчетов. В рассмотренных системах данные возможности присутствуют, но одновременно вместе с необходимыми функциями в пакете программного обеспечения могут приобретаться и многие другие, которыми, возможно, метрологу при выполнении своих обязанностей воспользоваться и не придется, так как одни функции явно избыточны, другие продублированы уже используемым программным обеспечением.

В связи с этим нет экономического смысла в покупке уже готового решения, наиболее выгодным предоставляется вариант самостоятельной разработки с учетом всех тонкостей деятельности предприятия, в частности, метролога.

2 СПЕЦИАЛЬНЫЙ РАЗДЕЛ

2.1 Выбор средств разработки

2.1.1 Выбор языка программирования

Одним из важных требований, предъявляемых к проектированию информационных систем, эксплуатируемых совместно на технологической базе весьма ограниченных возможностей, является большая их однородность, позволяющая обеспечить совместимость, мобильность, переносимость.

Сегодня в индустрии разработки бизнес-приложений основными языками программирования являются C# (Microsoft) и Delphi (Embarcadero Technologies).

C# — объектно-ориентированный язык. Разработан в 1998-2001 годах группой инженеров под руководством Андерса Хейлсберга в компании Microsoft как основной язык разработки приложений для платформы.NET Framework и впоследствии был стандартизирован как ECMA-334 и ISO/IEC 23270. Компилятор C# входит в стандартную установку .NET Framework [3.1].

При выборе ПО для разработки будем рассматривать следующие характеристики: надёжность; эффективность; стоимость; понятность пользователю; простота настройки; простота оформления экранных и печатных форм.

При выборе среды разработки рассмотрим следующие языки:

- Microsoft Visual C# 2008 Express Edition;
- Delphi 7.

Microsoft Visual C# 2008 Express Edition является мощной системой программирования, позволяющей быстро и эффективно создавать приложения для Microsoft Windows.

C# — это типизированный, объектно-ориентированный, простой и в то же время мощный язык программирования, который позволяет разработчикам создавать многофункциональные приложения. При использовании возможностей .NET Framework Visual C# 2008 позволяет создавать приложения

Windows, веб-службы, инструменты, компоненты и элементы управления баз данных, а также многое другое.

Delphi 7 можно назвать знакомой средой для визуальной разработки, основанной на Object Pascal. Этот пакет имеет поддержку новейших перспективных технологий веб-служб. Несет в себе предварительную версию средств для работы с Microsoft .NET Framework. Благодаря Delphi 7 разработчики, применяющие Delphi, могут получить дополнительные навыки программирования для .NET, отлаживать свои приложения для работы под управлением .NET, не теряя при этом навыков и знаний в области проектирования под Windows

Характерной чертой пакета становится полная поддержка уже созданных и новый Web-сервисов, встроенная модельно-управляемая разработка и возможность предварительного просмотра для Microsoft .NET Framework. Применяя Delphi 7 Studio более 1 миллиона разработчиков, программирующих на Delphi, могут уже сейчас улучшать своих навыки работы с .NET и реализовывать приложения для .NET с применением уже созданных наработок и навыков взаимодействия на Windows-платформу.

Основные преимущества и возможности Borland Delphi 7 Studio заключаются в:

- Создание корпоративных приложений с применением MDA — упрощает процесс создания, позволяя разработчикам ускорять процесс от проектирования к установке, минимизирую количество кода, а также время, нужное для разработки;
- RAD Visual Web Development — позволяет визуально разрабатывать Web-приложения в среде Delphi 7 Studio и исключает задачи создания общего сервера Application Mode;
- Встроенная межплатформенная поддержка Linux — Delphi 7 Studio будет идти в комплекте с Delphi language version of Borland Kylix 3, первой высокоэффективной, визуальной средой разработки (IDE) для упрощенного создания БД, GUI-, Web-приложений, а также Web-сервисов для ОС Linux;
- Доступность составления отчетов в рамках компании — позволяет разработчикам реализовывать межплатформенные отчеты, определяя

эффективность функционирования программ;

- Royalty-Free DataSnap Multi-tier Application Deployment (ранее MIDAS) — обновленное лицензирование гарантирует разработчикам целостных, одиночных, клиент-серверных и многоуровневых приложений реализовать разработку без скрытой платы за все время рабочего использования;
- Создание ПО для Windows XP — Delphi 7 Studio имеет поддержку Windows XP Theme, позволяющую разработчикам разрабатывать ПО, использующее положительные стороны интерфейса Windows XP User.

Сегодня Delphi — мощная технология создания высокопроизводительных приложений под Windows с развитым интерфейсом, позволяет достичь высокой скорости разработки за счет снижения времени кодирования согласно концепции Rapid Application Development с помощью визуальной среды разработки. В рамках данного подхода можно разрабатывать системы различного типа: настольные приложения, приложения для сенсорных экранов, платежных терминалов, а также приложения для баз данных, включая многозвенные, web-приложения и приложения, взаимодействующие с облачной инфраструктурой [3.2].

В Embarcadero Delphi 2010 усовершенствованная многоуровневая платформа DataSnap обеспечивает расширенный доступ к стандартным протоколам обмена данными, включая новые протоколы JSON и REST, поддерживающие взаимодействие с другими сервисами. Удобный и гибкий набор компонентов функционирует в среде последних версий Windows, что позволяет разрабатывать современные программные решения для цифровых планшетов, сенсорных панелей и информационных терминалов с пользовательским интерфейсом и поддержкой сенсорного ввода.

Для разработки программы был выбран язык программирования C#.

На данный момент язык программирования C# набирает очень большой темп, и нет столь простого и многофункционального языка, как Си шарп. В нем собраны все достоинства разных языков. Быстродействие выполнения приближается к языку Assembler.

Язык Си шарп имеет 300 000 библиотек разных функций, которые работают с максимальным быстродействием.

2.1.2 Выбор СУБД

Подбор наиболее подходящей под задачи СУБД всегда представляет из себя тяжелую задачу с различными требованиями и ключевыми элементами, но является чуть ли не самым важным этапом всей разработки приложений БД.

Для реализации поставленных задач можно использовать любую современную СУБД. Для удобного взаимодействия пользователя с системой необходимо будет тщательно продумать интерфейс системы - чтобы он был простой и в то же время функциональный.

Для оптимального функционирования базы данных необходимо будет правильно определить логические взаимосвязи между таблицами. Таким образом, на разработку самой базы данных и основы клиентского приложения может быть затрачено значительное время.

Точно определив, какие именно данные нам нужны, каким образом они будут храниться в памяти и какая должна быть система доступа к данным, мы тем самым решили только вопрос управления данными. Кроме этого нужен еще простой способ автоматизации решения предстоящих типовых задач.

Выбор системы управления баз данных (СУБД) является одним из важных этапов при разработке приложений баз данных. Выбранный программный продукт должен удовлетворять как текущим, так и будущим потребностям предприятия, при этом следует учитывать финансовые затраты на приобретение необходимого оборудования, самой системы, разработку необходимого программного обеспечения на ее основе, а также обучение персонала. Кроме того, необходимо убедиться, что новая СУБД способна принести предприятию реальные выгоды.

Проектируемая БД должна использоваться для накопления, хранения и обработки данных. Узким местом при разработке базы данных для информационно-аналитических систем является хранение большого количества элементов - вариантов решений. Решение задачи оптимизации при глобальном поиске в

информационно-аналитических системах, а так наличие малого количества запросов от самих пользователей информационно-аналитической системы позволяют не устанавливать жесткие требования к производительности СУБД. Таким образом, все выше перечисленное можно определить следующие требования к СУБД:

- Кроссплатформенность СУБД;
- Полная совместимость с выбранной средой разработки (PHP);
- Простота использования и внедрения;
- Распространенность и популярность СУБД. При использовании малораспространенной СУБД в будущем могут возникнуть проблемы с поддержкой и развитием ИС, что так же наложит дополнительные затраты на перенос накопленных данных.
- Надежность, позволяющая обеспечить решение задачи постоянного наполнения и обеспечения сохранность данных.

Для выбора СУБД выделены несколько групп критериев:

- Моделирование данных
- Особенности архитектуры и функциональные возможности
- Контроль работы системы
- Особенности разработки приложений
- Производительность
- Надежность
- Требования к рабочей среде
- Смешанные критерии

Основным принципом выбора СУБД следует считать определение программного продукта, в наибольшей мере соответствующего предъявляемым требованиям. Эту задачу решить не очень просто. Во-первых, к СУБД предъявляется большое число требований, которые с течением времени изменяются, во-вторых, СУБД имеют большое число параметров, что затрудняет их сравнение. Кроме того, информация о СУБД часто носит рекламный характер, не позволяющий сделать правильное суждение.

На уровне технических характеристик разнообразие СУБД еще больше, чем на качественном уровне. К техническим характеристикам относятся:

- общие параметры (операционная среда, потребность в оперативной памяти, ограничения на максимальный объем БД и др.);
- ограничения на операции над данными;
- типы данных;
- возможности средств формулировки и выполнения запросов;
- работа в многопользовательских средах;
- инструментальные средства разработки приложений.

Оценка производительности производится методом тестирования с помощью эталонных тестов из набора AS3AP (ANSI SQL Standard Scalable and Portable). В них контролируется широкий спектр часто встречающихся операций БД и моделируются однопользовательские и многопользовательские среды.

В Таблице 2.1 приведена сравнительная таблица трех распространенных систем управления базами данных, конкурирующих на рынке программного обеспечения по основным показателям.

Таблица СПЕЦИАЛЬНЫЙ РАЗДЕЛ.2 Сравнение СУБД

Показатели	Microsoft SQL Server 2008	баллы	MySQL 5.1	баллы	PostgreSQL 8.4	баллы
Поддерживаемые операционные системы	Windows Desktop/Server	3	Windows Desktop/Server , Linux, Unix, Mac	5	Windows1 Desktop/S22erver, Linux, Unix, 2Mac	5
Поддержка даты и времени	Да	3	Да (но без временной зоны)	2	Да	3
Аутентификация	Средствами БД и Active Directory	3	Средствами БД	2	Много разных методов, включающих	1

					предыдущие	
Производительность планировщика запросов для сложных запросов	Средняя (умеет параллельные запросы «из коробки»)	1	Очень хорошая	5	Плохая	1
Итого		10		14		10

Для проекта, описывающего разработку ИС метролога, наиболее приемлема СУБД MySQL.

При выборе программного обеспечения для разработки будем рассматривать следующие характеристики: надёжность; эффективность; стоимость; понятность пользователю; простота настройки; простота оформления экранных и печатных форм.

2.2 Характеристика базы данных

Создание ИС в кратчайшие сроки всегда сочеталась с рядом неудач, многие из которых были вызваны неимением полной спецификации всех требований; неимением адекватной методологии разработки; низкой степенью разделения проекта на различные составные части, поддающиеся эффективному управлению и контролю.

Для решения этих проблем применялся подход к разработке ПО, базирующийся на понятии жизненного цикла ИС (или жизненного цикла создания ПО).

База данных становится основным компонентом ИС, следовательно, жизненный цикл ИС так или иначе связан с жизненным циклом системы БД, поддерживающей его. Жизненный цикл ИС включает в себя планирование, подготовку и анализ требований, проектирование, подбор СУБД, создание приложений, доработку прототипа, реализация, тестирование, модификацию данных и сопровождение.

Рассмотрим наиболее важные вопросы проектирования ИС и выбора СУБД, от правильного решения которых зависит успех всего проекта в целом.

Разработка БД. Ключевым этапом создания ИС является этап разработка БД, целями которой становятся:

- Указание данных и связей между ними, нужных для всех основных областей применения рассматриваемой системы и любых групп ее пользователей;
- Проектирование модели данных, способной поддерживать реализацию любых требуемых транзакций, связанных с обработкой информации;
- Создание предварительного варианта проекта, структура которого удовлетворит все основные требования, предъявляемые к скорости работы системы.

Есть два основных подхода к разработке систем БД: нисходящий и восходящий. Восходящий идеален для проектирования простых БД с малым количеством атрибутов. Применение этого подхода затруднено при разработке БД с большим количеством атрибутов, установить среди которых все возможные функциональные зависимости сложно. При разработке сложных систем БД обосновано применение нисходящего подхода, хорошо показавшего себя в концепции модели «сущность-связь». В этом случае работа идет от идентификации сущностей и связей между ними, которые очень важны для данной разработки.

Весь процесс разработки БД имеет три фазы: концептуальное, логическое и физическое проектирование. Любая фаза заключается в создании необходимой модели данных, являющейся источником данных для последующей фазы. Главное значение в этом процессе возложено на концептуальную модель данных, создаваемую на основе параметров, записанных в спецификации пользовательских требований. Концептуальное проектирование БД не связано с такими подробностями ее реализации, как тип используемой целевой СУБД, тип применяемой вычислительной платформы и т.п., но качество концептуальной модели тут уже становится решающим фактором, который позволяет выразить трудозатраты на реализацию системы, ее производительность и последующий успех. Опыт разработки и применения ИС показывает, что ошибки, возникшие на этом этапе, самые трудно выявляемые и трудно устраняемые, так как они встречаются в основном на следующих этапах разработки системы – при разработке или сопровождении.

На этапе логического проектирования концептуальная модель данных переходит в логическую модель, реализуемую на основе выбранной модели

хранения данных целевой СУБД. Другими словами, этот этап показывает, какая СУБД будет применяться в качестве целевой - сетевая, иерархическая, реляционная или объектно-ориентированная. Но на этом этапе пропускаются все остальные аспекты исходной СУБД - например, отдельные особенности физической организации хранения данных. Логическая модель, показывающая особенности представления о реализуемой системе сразу нескольких типов пользователей, становится глобальной логической моделью данных. Есть два базовых подхода для разработки глобальной логической модели данных: централизованный и метод интеграции представлений. Если разрабатывается крупная ИС, предпочтительнее и эффективнее применять второй подход, когда глобальная логическая модель данных создается методом слияния отдельных моделей, показывающих представления разных групп пользователей.

На этапе физического проектирования принимаются решения о способах реализации создаваемой БД. Поэтому физическое проектирование сильно связано с конкретной СУБД. Между логическим и физическим проектированием имеется постоянная обратная связь, поскольку все решения, внедренные на этапе физического проектирования для увеличения производительности системы, также влияют на структуру логической модели данных. Главной целью физического проектирования БД можно назвать описание способа физической реализации логического проекта БД.

Диаграмма базы данных приведена на рисунке 2.1.

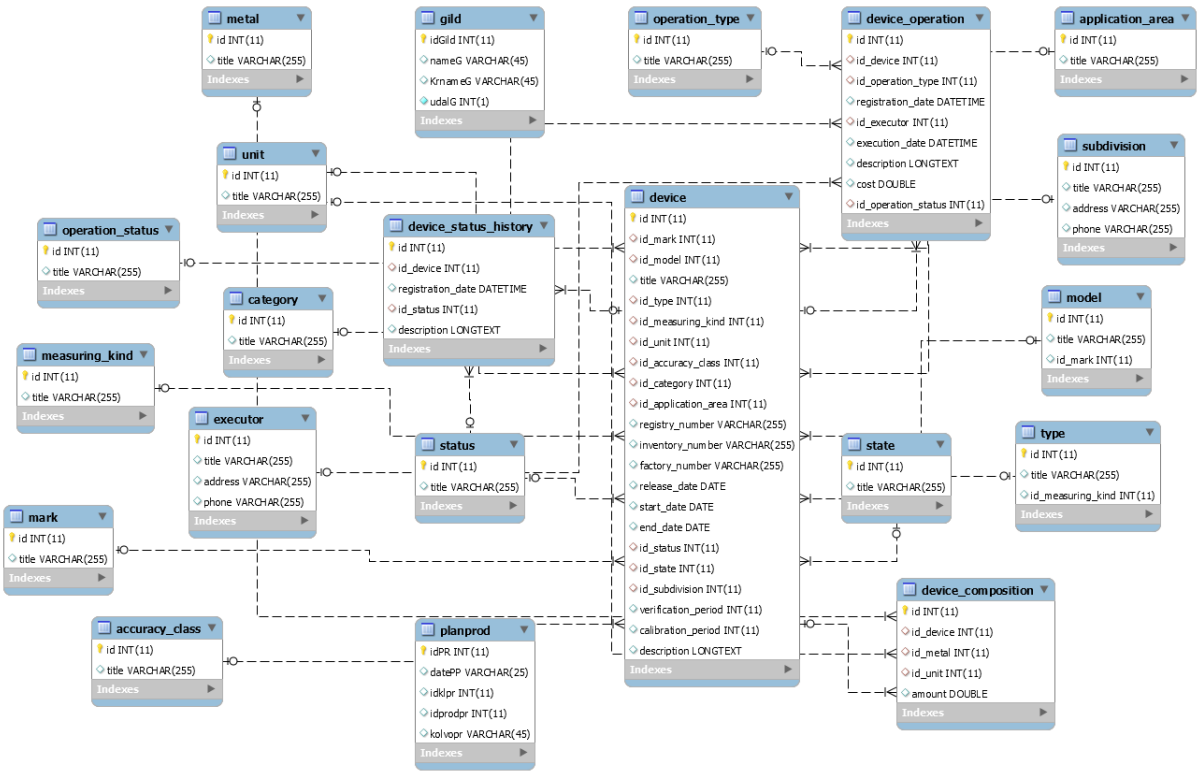


Рисунок СПЕЦИАЛЬНЫЙ РАЗДЕЛ.5 ER-модель базы данных

Характеристика каждой таблицы приведена ниже.

Таблица СПЕЦИАЛЬНЫЙ РАЗДЕЛ.3 Таблица «device»

Содержит данные по устройствам измерения.

№	Название поля	Описание поля
1	id	Код (первичный ключ)
2	id_mark	Производитель
3	id_model	Модель
4	title	Название
5	id_type	Тип СИ
6	id_measuring_kind	Вид измерений
7	id_unit	Единица измерения
8	id_accuracy_class	Класс точности
9	id_category	Категория СИ
10	id_application_area	Сфера применения
11	registry_number	Код СИ (гос. реестр)
12	inventory_number	Инвентарный номер

13	factory_number	Заводской номер
14	release_date	Дата выпуска
15	start_date	Дата начала использования
16	end_date	Дата окончания использования
17	id_status	Статус СИ
18	id_state	Состояние СИ
19	id_subdivision	Подразделение
20	verification_period	Период поверки (месяцы)
21	calibration_period	Период калибровки (месяцы)
22	description	Описание

Таблица СПЕЦИАЛЬНЫЙ РАЗДЕЛ.4 Таблица «device_operation»

Содержит данные по работам, которые проводились для устройства измерения.

№	Название поля	Описание поля
1	id	Код работы (первичный ключ)
2	id_device	Устройство
3	id_operation_type	Тип работы
4	registration_date	Дата регистрации работы
5	id_executor	Исполнитель
6	execution_date	Дата выполнения
7	description	Описание
8	cost	Стоимость
9	id_operation_status	Статус операции

Таблица СПЕЦИАЛЬНЫЙ РАЗДЕЛ.5 Таблица «device_composition»

Содержит данные по драгоценным металлам, которые имеются в устройстве.

№	Название поля	Описание поля
1	id	Код (первичный ключ)
2	id_device	Устройство
3	id_metal	Металл
4	id_unit	Единица измерения

5	amount	Количество

Таблица СПЕЦИАЛЬНЫЙ РАЗДЕЛ.6 Таблица «subdivision»

Справочник «Отдел».

№	Название поля	Описание поля
1	id	Код (первичный ключ)
2	title	Название
3	address	Адрес
4	phone	Телефон

Таблица СПЕЦИАЛЬНЫЙ РАЗДЕЛ.7 Таблица «executor»

Справочник «Исполнители работ».

№	Название поля	Описание поля
1	id	Код (первичный ключ)
2	title	Название
3	address	Адрес
4	phone	Телефон

Таблица СПЕЦИАЛЬНЫЙ РАЗДЕЛ.8 Таблица «mark»

Справочник «Производители СИ».

№	Название поля	Описание поля
1	id	Код (первичный ключ)
2	title	Название

Таблица СПЕЦИАЛЬНЫЙ РАЗДЕЛ.9 Таблица «model»

Справочник «Модели СИ».

№	Название поля	Описание поля
1	id	Код (первичный ключ)
2	title	Название

Таблица СПЕЦИАЛЬНЫЙ РАЗДЕЛ.10 Таблица «unit»

Справочник «Единицы измерения».

№	Название поля	Описание поля
1	id	Код (первичный ключ)
2	title	Название

Таблица СПЕЦИАЛЬНЫЙ РАЗДЕЛ.11 Таблица «type»

Справочник «Типы СИ».

№	Название поля	Описание поля
1	id	Код (первичный ключ)
2	title	Название

Таблица СПЕЦИАЛЬНЫЙ РАЗДЕЛ.12 Таблица «measuring_kind»

Справочник «Вид измерений».

№	Название поля	Описание поля
1	id	Код (первичный ключ)
2	title	Название

Таблица СПЕЦИАЛЬНЫЙ РАЗДЕЛ.13 Таблица «accuracy_class»

Справочник «Класс точности».

№	Название поля	Описание поля
1	id	Код (первичный ключ)
2	title	Название

Таблица СПЕЦИАЛЬНЫЙ РАЗДЕЛ.14 Таблица «category»

Справочник «Категория СИ».

№	Название поля	Описание поля
1	id	Код (первичный ключ)
2	title	Название

Таблица СПЕЦИАЛЬНЫЙ РАЗДЕЛ.15 Таблица «application_area»

Справочник «Сфера применения».

№	Название поля	Описание поля
1	id	Код (первичный ключ)
2	title	Название

Таблица СПЕЦИАЛЬНЫЙ РАЗДЕЛ.16 Таблица «status»

Справочник «Статус СИ».

№	Название поля	Описание поля
1	id	Код (первичный ключ)
2	title	Название

Таблица СПЕЦИАЛЬНЫЙ РАЗДЕЛ.17 Таблица «state»

Справочник «Состояние СИ».

№	Название поля	Описание поля
1	id	Код (первичный ключ)
2	title	Название

Таблица СПЕЦИАЛЬНЫЙ РАЗДЕЛ.18 Таблица «subdivision»

Справочник «Подразделение».

№	Название поля	Описание поля
1	id	Код (первичный ключ)
2	title	Название

Таблица СПЕЦИАЛЬНЫЙ РАЗДЕЛ.19 Таблица «operation_status»

Справочник «Статус выполнения работы».

№	Название поля	Описание поля
1	id	Код (первичный ключ)
2	title	Название

Таблица СПЕЦИАЛЬНЫЙ РАЗДЕЛ.20 Таблица «operation_type»

Справочник «Тип работы».

№	Название поля	Описание поля
1	id	Код (первичный ключ)
2	title	Название

2.3 Системы классификации, применяемые в ИС метролога

Для обеспечения эффективного поиска, обработки на ЭВМ и передачи по каналам связи технико-экономических данных, их нужно представить в цифровом

виде. С этой целью данные сначала упорядочивают (классифицируют), а затем формализуют (кодируют) с помощью классификатора.

Под классификацией понимается разделение множества объектов на подмножества по их различным и схожим чертам в соответствии с установленными методами. Классификация фиксирует определенные связи между классами объектов. Под объектом можно подразумевать любой предмет, процесс, явление нематериального или материального свойства. Система классификации могут группировать объекты и выделять определенные классы, которые затем характеризуются рядом общих свойств. Поэтому совокупность правил распределения объектов множества на подмножества можно считать системой классификации.

Свойство или характеристика объекта классификации, позволяющее найти его сходство или различие с другими объектами классификации, названо признаком классификации. Множество или подмножество, которое включает в себя часть объектов классификации по одному или нескольким признакам, называется классификационной группировкой.

Классификатор, это ни что иное, как документ, который позволяет осуществлять формализованное описание данных в ИС, имеющей наименования объектов, наименования классификационных группировок, а также их кодовые обозначения.

По сфере действия классификаторов их делят на: международные, общегосударственные, отраслевые и локальные.

Международные классификаторы находятся в составе Системы международных экономических стандартов и служат для передачи данных между компаниями из разных стран мирового сообщества.

Общегосударственные классификаторы распространяют свое влияние внутри страны для организации процессов передачи и обработки данных между экономическими системами государственного уровня.

Отраслевые классификаторы применимы для реализации процедур обработки данных и ее передачи организациями внутри отрасли.

Локальные классификаторы применяются внутри отдельно взятой компании или предприятия.

В системе используется 5 видов кодирования, предназначенные для однозначной идентификации сотрудников, средств измерения, операций, статусов операций. Виды системы кодирования, указаны в Таблице 2.20.

Таблица СПЕЦИАЛЬНЫЙ РАЗДЕЛ.21 Используемые системы кодирования

Кодируемое множество объектов	**Длина кода**	**Мощность кода**	**Система кодирования**	**Система классификации**	**Вид классификатора**
Тип операции	2	99	порядковая	иерархическая	общесистемный
Оборудование	4	9999	порядковая	иерархическая	общесистемный
Операции	4	9999	порядковая	иерархическая	общесистемный
Отдел	2	99	порядковая	иерархическая	общесистемный
Сотрудники	5	99999	порядковая	иерархическая	общесистемный
Должности	1	9	порядковая	иерархическая	общесистемный
Тип пользователя	2	99	порядковая	иерархическая	общесистемный
Тип средства измерения	2	99	порядковая	иерархическая	общесистемный
Средства измерения	4	9999	порядковая	иерархическая	общесистемный
Статус	2	99	порядковая	иерархическая	общесистемный

Рассмотрим более подробно состав каждого классификатора.

Классификатор Тип операции.

Данный классификатор был введён для идентификации типов операций.

1. система кодирования – порядковая
2. количество признаков классификации – 1
3. объекты кодирования: Тип операции
4. структурная формула классификатора

Классификатор оборудования

Данный классификатор вводится для идентификации оборудования, применяемого для производства проверок и калибровок средств измерения.

1. система кодирования – порядковая

2. количество признаков классификации – 1

3. объекты кодирования: оборудование

4. структурная формула классификатора

Классификатор операций

Данный классификатор вводится для идентификации операций, которые производятся при работе со средствами измерений.

1.система кодирования – порядковая,

2. количество признаков классификации – 1,

3. объекты кодирования: операции

4. структурная формула классификатора

Классификатор отделов

Данный классификатор вводится для идентификации отделов компании.

1. система кодирования – порядковая

2. количество признаков классификации – 1

3. объекты кодирования: отделы

4. структурная формула классификатора

Классификатор должностей

Данный классификатор вводится для идентификации должностей сотрудников компании.

1. система кодирования – порядковая

2. количество признаков классификации – 1

3. объекты кодирования: должности

4. структурная формула классификатора

Классификатор сотрудника

Данный классификатор вводится для идентификации сотрудников, которые производят операции над средствами измерений и которые сдают средства измерения на калибровку и поверку.

1.система кодирования – порядковая,

2. количество признаков классификации – 1,

3. объекты кодирования: сотрудники

4. структурная формула классификатора

Классификатор типов пользователей

Данный классификатор вводится для идентификации типов пользователей, имеющих доступ к системе.

1. система кодирования – порядковая

2. количество признаков классификации – 1

3. объекты кодирования: типы пользователей

4. структурная формула классификатора

Классификатор типов средств измерения

Данный классификатор вводится для идентификации типов средств измерений, калибровка и поверка которых учитывается в системе.

1. система кодирования – порядковая
2. количество признаков классификации – 1
3. объекты кодирования: типы средств измерения
4. структурная формула классификатора

Классификатор средств измерения

Данный классификатор вводится для идентификации средств измерений, калибровка и поверка которых учитывается в системе.

1. система кодирования – порядковая
2. количество признаков классификации – 1
3. объекты кодирования: средства измерения
4. структурная формула классификатора

Классификатор статусов операций

Данный классификатор вводится для идентификации статусов операций, которые производятся над средствами измерений.

1. система кодирования – порядковая
2. количество признаков классификации – 1
3. объекты кодирования: статусы операций
4. структурная формула классификатора

2.4 Разработка проекта автоматизации

Методология проектирования ИС включает в себя описание процесса создания и сопровождения систем в виде жизненного цикла (ЖЦ) ИС, отождествляя его с некоторой последовательностью стадий и исполняемых на них процессов. Для каждой стадии выявляется состав и последовательность производимых работ, итоговые результаты, методы и средства, нужные для реализации работ, ответственность и роль участников и т.д. Подобное формальное описание ЖЦ ИС дает возможность спланировать и подготовить процесс совместной разработки и поддерживать управление этим процессом.

Жизненный цикл (ЖЦ) ИС представляется, как ряд событий, случающихся с системой с момента ее внедрения и до окончания использования.

Модель ЖЦ отражает различные состояния системы, от момента возникновения необходимости в данной ИС и до момента ее окончательного вывода из эксплуатации. Модель жизненного цикла представлена некой структурой, которая содержит процессы, действия и задачи, реализуемые в ходе создания, работы и сопровождения ПО в течение всей жизни системы, от выявления требований до окончания ее использования.

Сегодня известны и применимы следующие модели жизненного цикла:

- Каскадная модель включает в себя последовательную реализацию всех этапов проекта в заранее определенном порядке. Начало следующего этапа говорит о полном завершении работ на предыдущем этапе.
- Поэтапная модель с периодичным контролем. Создание ИС реализовано в виде итераций с циклами обратной связи между этапами. Межэтапные проверки позволяют учесть реально существующее взаимовлияние итогов разработки на различных этапах; ЖЦ каждого из этапов продлевается на весь срок разработки.
- Спиральная модель. На любом витке спирали выполняется генерация очередной версии продукта, корректируются требования проекта, выражается его качество и планируются работы уже следующего витка. Особое внимание при этом обращается на начальные этапы разработки - анализ и проектирование, где

возможность создания тех или иных технических решений обосновывается и проверяется благодаря построению прототипов.

Каскадный подход отлично зарекомендовал себя в процессе создания относительно простых ИС, когда в самом начале разработки можно с большой точностью и полнотой составить все требования к системе. Главным недостатком такого подхода является то, что основной процесс разработки системы не может полностью уложится в такие жесткие рамки, постоянно есть потребность в возврате к уже завершенным этапам для уточнения или изменения ранее принятых решений. В итоге реальный процесс разработки ИС становится соответствующим поэтапной модели с периодичным контролем.

Для разработки документооборота материально-технического оборудования выбираем каскадную модель.

Все стадии создания системы предусматривают выполнение некоторого объема работ, представляемых в виде процессов ЖЦ. Процесс выражается как совокупность объединенных действий, изменяющих входные данные в выходные. Описание любого процесса состоит из перечня решаемых задач, исходных данных и итоговых результатов.

Есть целый ряд стандартов, определяющих ЖЦ ПО, а в отдельных случаях и процессы разработки.

Среди самых известных стандартов выделяют следующие:

- ГОСТ 34.601-90 - распространяется на АИС и указывает в себе стадии и этапы их создания. Также в нем имеется описание содержания работ на всех этапах. Стадии и этапы работы, отраженные в стандарте, зачастую соответствуют каскадной модели жизненного цикла.
- ISO/IEC 12207:1995 - стандарт на процессы и реализацию жизненного цикла. Применяется ко всем видам заказного ПО. Стандарт не имеет описания стадий, фаз и этапов.
- Custom Development Method по созданию прикладных ИС - технологический материал, углублённый до уровня заготовок проектных документов, которые рассчитаны на применение в проектах совместно с Oracle.

Используется CDM для типовой модели ЖЦ (имеются все работы/задачи и этапы), а также для случаев "быстрой разработки" (Fast Track) или "облегченного подхода", которые будут оптимальны в малых проектах.

- Rational Unified Process (RUP) включает в себя итеративную модель разработки, имеющую четыре фазы: старт, анализ, создание и использование. Все эти фазы могут быть разделены на этапы (итерации), по итогу которых имеется версия для внутреннего или внешнего использования. Реализация четырех основных фазы считается циклом разработки, и любой такой цикл завершается созданием версии системы. В случае, если работа над проектом не прекращается и после этого, полученный продукт продолжает оптимизироваться и снова проходит те же фазы. Суть реализации в рамках RUP - это разработка и сопровождение моделей на базе UML.
- Microsoft Solution Framework (MSF) похож на RUP, так же имеет четыре фазы: исследование, построение, создание, стабилизация, является итерационным, включает в себя применение объектно-ориентированного моделирования. MSF в отличии от RUP в сильнее ориентирован на создание бизнес-приложений.
- Extreme Programming (XP). Экстремальное программирование (самая молодая среди остальных методологий) было реализовано в 1996 году. В основе методологии лежит командная работа, четкая коммуникация между исполнителем и заказчиком в течение всего срока проекта, а сама разработка реализуется методом последовательной доработки прототипов.
- Стандарт ISO/IEC серии 15288.

В связи с небольшим объемом разрабатываемой автоматизированной системы и ее характером необходимо использовать именно стандарт ISO/IEC серии 15288.

Задача внедрения информационной системы включает в себя создание (адаптацию) и запуск в продуктивную эксплуатацию элементов информационной системы. Так как разработка информационной системы будет осуществляться своими силами, то и внедрение будет происходить без привлечения посторонних специалистов.

Этап внедрения планируется разбить на следующие подэтапы:

1. Предпроектное обследование. В ходе обследования выявляются основные информационные потоки на предприятии и сверяется база основной нормативно-справочной документации. Главным требованием в данном случае является наличие всех необходимых для функционирования корпоративных информационных систем справочников и классификаторов и соответствие принципових организации требованиям системы. В ходе выполнения этапа обязательно должны быть проанализированы на полноту корпоративные стандарты учета и отчетности. На данном этапе также производится диагностирование проблем, которые могут возникнуть при внедрении, разрабатывается и согласовывается настройка справочников и классификаторов системы в соответствии с сформулированными требованиями. При необходимости, принимаются решения об изменении существующих практик учета или функциональных моделей. По результатам этапа формируется подписываемый всеми участниками проекта внедрения документ, который описывает все выявленные проблемы и намечает пути их ликвидации.

2. Построение информационно-функциональной модели деятельности предприятия, описание и оптимизация процессов, подвергающихся автоматизации. Моделирование должно проводиться хорошо обученными сотрудниками рассматриваемого предприятия с привлечением высококвалифицированных консультантов и с привязкой созданной модели к стандартам бизнеса и к будущей системе.

3. Адаптация ИС на предприятии. В ходе этапа производится настройка системы тестирование отдельных модулей и функций группой внедрения. На данном этапе также очень важно наличие корпоративных стандартов, так как именно они являются основой настроек системы.

4. Опытная эксплуатация информационной системы. Осуществляется для тестирования полного соответствия функциональности, полученной в результате настройки системы, требованиям предприятия. На этом этапе сохраняется двойной ввод данных в старую и новую системы. В ходе опытной эксплуатации: генерируются стандартные отчеты (с помощью ИС и обычными способами) и производится верификация данных; система постепенно вводится в эксплуатацию,

по отдельным участкам учета; документируются инструкции по ведению рабочих мест и корректируются должностные инструкции участников учетного процесса. В отдельных подразделениях предприятия в систему вводятся фактические данные (в ограниченном объеме) и последовательно тестируются бизнес-функции путем моделирования реальных ситуаций деятельности предприятия (в условиях, максимально приближенных к действительности). Отрабатывается взаимная работа подразделений на основе тестовых пилотных примеров. Конечные пользователи (сотрудники отдела ИТ) обучаются работе с настроенной системой непосредственно на своих рабочих местах. После обучения конечных пользователей отрабатывается интегрированный пилотный пример и полностью моделируется деятельность предприятия. На основе результатов выполнения пилотного примера руководством предприятия принимается решение о переводе ИС в промышленную эксплуатацию.

Этап эксплуатации подразумевает под собой непосредственное использование информационной системы для выполнения ею тех функций, для которых она предназначена.

Работы, ожидаемые на этапе эксплуатации, можно разделить на две группы: плановые и неплановые.

К плановым работам будут относится такие работы, как:

- инсталляция программного обеспечения;
- базовая настройка и проверка работоспособности компонентов устанавливаемой системы;
- устранение недостатков в конфигурации системы;
- проверка надежности работы системы;
- окончательная донастройка.

Данные работы будут проводиться той же группой, что и на ранних этапах. В состав этой группы входят сотрудники технического отдела - технические специалисты и системные администраторы. Модель процессов MSF учитывает частые изменения проектных требований. Она основывается на том, что создание решения включает в себя короткие циклы, создающие поступательные движения от простейших версий решения к его итоговому виду.

В модели MSF есть 5 фаз ЖЦ: создание концепции, построение плана, разработка, тестирование, внедрение.

Этап создания концепции состоит из заложения одной из фундаментальных основ успеха проекта – сбор и сплочение проектной группы на базе выработки единого видения. Проектная группа должна четко понимать, что она хочет сделать для заказчика и выразить цель таким образом, чтобы 100% мотивировать и заказчика, и проектную команду. В таком случае в роли заказчика выступает сам разработчик. Создание высокоуровневого взгляда на цели и варианты проекта расценивается как ранняя стадия планирования. Она готовит почву для процессов разработки детальных планов, которые осуществятся непосредственно во время фазы планирования.

Главными задачами этапа создания концепции становится сборка ядра проектной группы и подготовка документа с описанием рамок проекта и общими требованиями. Составление видения проекта и специфицирование его рамок не является одним и тем же, хотя для успеха проекта важны оба компонента. Видение – это неограниченное представление о том, каким хотелось бы видеть решение. Рамки же задают понятные границы того, что из предложенного этим видением будет возможно реализовать в условиях уже созданных проектных ограничений.

За время проведения этапа подготовки концепции реализуется определение и анализ бизнес требований. Подробнее эти требования уже анализируются во время этапа планирования.

Веха «Концепция утверждена» – в ней проектная группа составляет соглашение об общих задачах проекта, доступных в решении функциональности и конкретных временных рамках.

Итоги:

- Базовое описание и рамки проекта;
- Составление структуры проекта.

Этап планирования включает в себя основную работу по составлению планов проекта. Он подразумевает составление проектной группой функциональной

спецификации, создание дизайнов, доработку рабочих планов, оценку затрат на проект и срока доработки разных составляющих проекта.

В начале этапа планирования проектная группа изучает и документирует проектные требования. Они делятся на 4 общих категории: бизнес-требования, потребительские требования, требования по использованию и системные требования, касающиеся решения в целом. В рамках проектирования решения и разработки его функциональной спецификации важно следить за соответствием между указанными требованиями и проектируемой функциональностью. Это соответствие не так важно и будет взаимно-однозначным. Оно является одним из способов отслеживания корректности дизайна и его полноты для достижения поставленных перед решением целей.

Процесс проектирования можно назвать систематическим способом продвижения от абстрактных задач к конкретным техническим деталям. Он начинается с методичного анализа профилей пользователей, описывающих различные типы пользователей (включая персонал сопровождения) и их должностные обязанности. Значительная часть этой работы реализуется во время фазы подготовки концепции. Затем составляется набор сценариев применения, в каждом из которых прогоняется выполнение какой-либо операции конкретным типом пользователя. В итоге каждый сценарий использования разделяется на последовательность специфических действий, называемых примерами использования, необходимыми для выполнения пользователем с целью реализации операции.

Есть 3 уровня процесса проектирования: логический дизайн, концептуальный дизайн и физический дизайн. Работа над логическим дизайном начинается через определенный промежуток времени после начала работы над концептуальным дизайном, а создание физического дизайна начинается через некоторый промежуток времени после начала работы над логическим.

Результаты процесса проектирования описываются в функциональной спецификации. Функциональные спецификации представляют вид и поведение

каждой компоненты решения. Также для всех составляющих имеется их архитектура и дизайн.

Функциональная спецификация используется для множества целей. Главные из них – это:

- Команды разработчикам о том, что они должны будут реализовать;
- База для оценки объема работы;
- Полноценное соглашение с заказчиком о том, что нужно делать;
- Оптимизация деятельности всей проектной команды.

Как только разработана базовая версия функциональной спецификации, начинается детальное планирование. Руководитель проектной группы готовит план и участвует в командных сессиях планирования. Примеры планов состоят из плана внедрения, плана тестирования, плана использования, плана мер безопасности, плана обучения.

Далее проектная группа совместно оценивает планы и находит взаимозависимости между ними. Все планы синхронизируются и выражаются в виде сводного плана проекта.

Члены проектной группы, входящие в разные ролевые кластеры, подсчитывают нужное для выполнения запланированных задач время и готовят календарный график сдачи результатов. Далее идет синхронизация календарных графиков с последующим их внедрением в сводный календарный график проекта.

Веха «Планы проекта утверждены» отражает соглашение между проектной группой. Промежуточные вехи этапа планирования успешно пройдены, подготовленные календарные графики реалистичны, все роли распределены и ответственности в команде указаны должным образом. Функциональные спецификации, сводный план и сводный календарный график проекта становятся основой для принятия альтернативных решений в будущем.

Утвержденные планы, таблицы, графики и т.д. составляют начальную версию проекта. Она состоит из всех соглашений, принятых на основе общего мнения с учетом 3 плановых показателей проекта: ресурсы, время и функциональность

решения. После создания и утверждения базовой версии проекта, проектная группа начинает ее разрабатывать.

Изменения в исходной базовой версии проекта строго отслеживаются. Это не означает, что все принятые во время этапа планирования решения неизменны – в ходе этапа разработки проектная группа должна изучить и формально утвердить или опровергнуть все предлагаемые корректировки базовой версии.

Итоги:

- Описание функционала;
- График и план выполнения проекта.

Этап разработки включает в себя создание проектной группой компонент решения (документацию и программный код). Но часть этой работы может продолжаться также на этапе тестирования, если такая необходимость имеется. Данный этап также включает в себя создание инфраструктуры.

Активность проектной команды на этом этапе не ограничена составлением кода – все ролевые кластеры участвуют в разработке и тестировании решения.

Веха «Разработка завершена» становится итогом всего этапа разработки. К моменту ее наступления разработка всех компонентов завершена, и решение готово к тестированию и стабилизации. Компания может оценить решение и определить все оставшиеся проблемы и нерешенные вопросы, которые важно уладить до выпуска решения.

Итоги:

- Готовый исходный код приложений;
- Скрипты конфигурации и инсталляции;
- Окончательная доработка функционала;
- Обоснование решения;
- Сценарии и описание тестов.

Этап тестирования подразумевает проведение проверки разработанного решения. При этом внимание приковано к его работе в реалистичной модели производственной среды. Проектная группа занята нахождением и устранением ошибок, а также подготовкой решения к выпуску.

После того, как появляется версия, достаточно стабильная для того, чтобы быть кандидатом для выпуска, производится первоначальное внедрение решения.

Этап тестирования завершается вехой «Готовность решения утверждена». В состоянии, которые имеется к данному моменту, решение может полноценно внедряться в производственную среду. Также к моменту наступления этой вехи проектная группа заканчивает разрешение всех существенных проблем и реализует внедрение решения. Ответственность за постоянное управление и поддержку решения теперь лежит на команде сопровождения.

Итоги:

- Готовый продукт;
- Готовая документация;
- Обоснование решения;
- Итоги всех тестов;
- Код работающего приложения;
- Проектная документация;
- Результаты пройденного этапа.

Этап внедрения включает в себя внедрение решения проектной группой, тестирование внедренного решения, передачу работы персоналу поддержки и сопровождения. По итогам внедрения проектная группа анализирует итоги работы и удовлетворенность заказчика.

В момент этого этапа по ходу переноса компонент решения из среды тестирования в производственную среду также идут работы по тестированию всего комплекса.

Веха «Внедрение завершено» является окончанием этапа внедрения. К этому времени решение уже дает заказчику некоторую бизнес-отдачу, а проектная группа заканчивает свою деятельность.

Решение разрабатывается стабильным и четко удовлетворяющим выработанным критериям успешности. Стабильность решения показывает также готовность систем его использования и обслуживания.

Итоги:

- ИС эксплуатации и поддержки;
- Процессы и процедуры;
- Журналы протоколов, базы знаний и отчеты;
- Версии проектных документов, массивы информации и программный код, созданный во время проекта;
- Отчет об окончании проекта;
- Итоговые версии всех проектных документов;
- Уровень удовлетворенности заказчика и потребителей;
- Описание других шагов.

В процессе использования персонал компании должен четко следовать всем инструкциям, которые относятся к созданной ИС. В случае проблем или вопросов, персонал обращается в службу поддержки. Эта служба проанализирует конкретную ситуацию, и примет меры для возобновления работы.

2.5 Ожидаемые риски и их описание

В процессе ЖЦ создаваемой ИС не избежать разного рода рисков.

Выделим главные риски, характерные для каждого этапа ЖЦ нашей ИС и обозначим меры их минимизации.

Этап подготовки концепции – есть риск сознания концепции, которую в дальнейшем будет почти невозможно реализовать. В подготовке концепции должны быть описаны главные функции разрабатываемой ИС. Главное выделить основу, и в дальнейшем развивать разработанную систему.

Для предотвращения реализации рисков на этапе подготовки концепции, важно четко понимать свои возможности. Для предотвращения переоценки собственных сил, изначально нужно создать общую концепцию, в которой уже будут заложены только базовые функции системы. И по мере углубления в эту разработку можно увеличивать дополнительные функции.

Этап планирования имеет риск неверного планирования, создание очень оптимистичных планов проекта, где компания не сможет уложится, вследствие чего придется увеличивать время разработки, что может привести к удорожанию проекта

в целом. К этапу планирования нужно отнестись очень важно, отслеживать каждый шаг и анализировать реалистичность результатов.

Для минимизации риска на этапе планирования нужно во время планирования заложить в график поправки на некоторые задержки в выполнении тех или иных действий. Так нужно постараться создать гибкий график, который бы не корректировался из-за задержки или опережения.

Этап разработки содержит риск того, что разработка некоторого модуля будет связана с большими трудностями, а отдельная функция будет мешать продвижению разработки. На данном этапе нужно вовремя выявить проблемный модуль или функцию и по возможности упростить ее, заменить другой или удалить полностью из проекта.

Чтобы предотвратить риск разработки сложного модуля, принимается несколько решений: либо разбивать данный модуль на несколько и решать указанные задачи по отдельности, либо упрощать сложный модуль, если это является единственным вариантом преодоления риска.

Этап тестирования имеет риск выявления большого количества ошибок в программном коде, что повлечёт большие затраты на доработку и устранение всех выявленных ошибок. Нельзя предсказать, какой количество ошибок будет найдено и как много времени будет нужно на их устранение.

Для минимизации рисков на этапе тестирования необходимо данному этапу отвести максимально возможное время, выделенное на создание системы, поскольку в зависимости от того, как качественно будет реализован продукт, зависит, примет ли заказчик данную разработку или нет.

Этап внедрения может быть очень длительным, если заказчик каким-то образом будет не доволен созданным продуктом, а также сам персонал автоматизируемой компании может отрицательно относиться к внедрению нового ПО.

Для минимизации рисков на данном этапе важно произвести качественное обучение персонала еще до начала внедрения, подготовить службу сопровождения и поддержки, понять, какие проблемы имеют место быть в процессе внедрения и быть

готовым к их решению. Постоянно отвечать на вопросы персонала по поводу возникающих у него проблем, открыть горячую линию для решения возможных проблем.

3 ТЕХНОЛОГИЧЕСКИЙ РАЗДЕЛ

3.1 Характеристика входной и выходной информации

3.1.1 Характеристика нормативно-справочной и входной оперативной информации

Входной информацией для разрабатываемой информационной системы являются данные сотрудников, об отделах и должностях, сведения о типах средств измерения, сведения об операциях и их статусах. Эти данные поступают как в цифровом, так и в печатном виде.

В систему вносятся следующие данные о пользователях:

- Фамилия
- Имя
- Отчество
- Дата рождения
- Логин
- Пароль
- Должность
- Тип пользователя

Для обеспечения работы системы предусмотрены справочники, приведенные в таблице 3.1.

Таблица ТЕХНОЛОГИЧЕСКИЙ РАЗДЕЛ.22 Перечень используемых справочников

название справочника	ответственный за ведение	средний объём справочника в записях	среднюю частоту актуализации	средний объем актуализации, %
Статусы операций	Администратор	45	1 раз в месяц	10
Тип средства измерения	Администратор	150	1 раз в год	10
Тип пользователя	Администратор	250	1 раз в год	10
Тип операции	Администратор	5	1 раз в год	10
Статусы	Администратор	45	1 раз в месяц	10
Средства измерения	Администратор	45	1 раз в месяц	10
Отдел	Администратор	45	1 раз в месяц	10
Должности	Администратор	45	1 раз в месяц	10

Сотрудники	Администратор	45	1 раз в месяц	10
Оборудование	Администратор	45	1 раз в месяц	10

Все документы могут поступать как бумажной, так и в электронной формах.

Состав справочников приведен ниже.

- Справочник Оборудование:
- Тип_СИ
- Заводской_номер
- Дата_начала_использования
- Примечание
- Отдел
- Справочник Сотрудники:
- Фамилия
- Имя
- Отчество
- Дата_рождения
- Логин
- Пароль
- Должность
- Тип_пользователя
- Справочник Должности:
- Название
- Справочник Отдел:
- Название
- Справочник Средства измерения:
- Номер_госреестра
- Название
- Изготовитель
- Справочник Статусы
- Название
- Справочник Тип операции

- Название
- Справочник Тип пользователя
- Название
- Справочник Тип средства измерения
- Название
- Справочник Статусы операций:
- Операция
- Статус
- Дата

Оперативная информация учитывается в таблице Операции со следующим реквизитным составом:

- Оборудование
- Дата_завершения
- Тип_операции
- Сотрудник
- Описание

3.1.2 Характеристика результатной информации

В качестве выходных данных в разработанной ИС формируются следующие документы:

- Список средств измерения;
- Список сотрудников;
- Список оборудования;
- Отчет по выполненным поверкам за период.
- Отчет по выполненным калибровкам за период.
- Отчет по выполнению графика поверок;
- Отчет по выполнению графика калибровок;
- Список средств измерения со сроками проведения калибровок и поверок.

Список средств измерения содержит код, номер и название средства измерения, список сотрудников – ФИО, должность сотрудника, его логин для доступа в систему. Список оборудования содержит такие реквизиты, как код, модель, заводской номер, название, дату начала использования.

Остальные отчеты формируются за период.

В системе учтены две основные операции – калибровка и поверка средств измерения. Именно по эти операциям формируются отчеты.

Отчет по выполненным поверкам за период содержит следующие реквизиты:

- № п/п
- Дата поступления
- Дата выполнения
- Тип СИ (модель)
- Наименование СИ Заводской номер
- Принадлежит отделу
- Ответственный за выполнение работы

Отчет по выполненным калибровкам за период содержит следующие реквизиты:

- № п/п
- Дата поступления
- Дата выполнения
- Тип СИ (модель)
- Наименование СИ Заводской номер
- Принадлежит отделу
- Ответственный за выполнение работы

Отчет по выполнению графика поверок характеризует степень выполнения запланированных операций по поверке СИ.

Отчет по выполнению графика калибровок характеризует степень выполнения запланированных операций по калибровке СИ.

Список средств измерения со сроками проведения калибровок и поверок содержит следующие реквизиты:

- № п/п СИ
- Тип СИ (Модель)
- Заводской номер
- Дата поступления
- Срок (до)
- Тип операции

3.2 Программное обеспечение задачи (комплекса задач, ИП)

3.2.1 Общие положения (дерево функций и сценарий диалога)

Основная функции разрабатываемой информационной системы, сложна и требуют иерархического представления. Для более наглядного представления функции системы необходимо декомпозировать на составляющие его функции.

Представим основную функцию системы в дерева функций.

На рисунке 3.1. представлено дерево функций системы.



Рисунок ТЕХНОЛОГИЧЕСКИЙ РАЗДЕЛ.6 Дерево функций системы

Первый уровень представляет основную функцию разрабатываемой системы – учёт информации о производимых операциях со средствами измерения. В ходе выполнения этой функции пользователь вводит данные о средстве измерения, необходимой операции, ее результате после выполнения, ответственном лице.

Второй уровень системы представляет функцию получения вышеперечисленных и показанных на схеме отчетов. Кроме получения отчетов в виде экранной формы, присутствует возможность экспорта отчетов в форматы pdf, xls, opendoc.

Служебные функции содержат функции настройки доступа к ИС, а также авторизационных данных, работу со справочниками – добавление и редактирование данных.

Сценарии диалога системы для пользователя приведен на рисунке 3.2.



Рисунок ТЕХНОЛОГИЧЕСКИЙ РАЗДЕЛ.7 Сценарий диалога для пользователя

После авторизации, которая заключается в вводе логина и пароля, пользователь получает доступ к главному меню программы. Выбирая необходимый пункт, пользователь либо вводит данные о плане проведения операций, либо управляет содержанием справочников, либо получает в необходимом формате отчетные документы.

3.2.2 Структурная схема пакета (дерево вызова процедур и программ)

Схема вызова программных модулей представлена на Рисунке 3.3.



Рисунок ТЕХНОЛОГИЧЕСКИЙ РАЗДЕЛ.8 Схема вызова программных модулей

Описание программных модулей представлено в Таблице 3.2.

Таблица ТЕХНОЛОГИЧЕСКИЙ РАЗДЕЛ.23 Описание программных модулей

№ п/п	**Наименование модуля**	**Функции модуля**
1)	ПМ Авторизация	Выполняет авторизацию пользователей при их доступе в систему
2)	ПМ Главное меню	Производит вызов меню, для каждого вида пользователей - своего
3)	ПМ работы со справочниками	Содержит определенные процедуры, позволяющие осуществлять редактирование

№ п/п	Наименование модуля	Функции модуля
		справочников
4)	ПМ получения отчетов	Содержит заранее определенные процедуры формирования отчетных документов

3.2.3 Описание программных модулей

Модуль данных проекта представлен на рисунке 3.4.

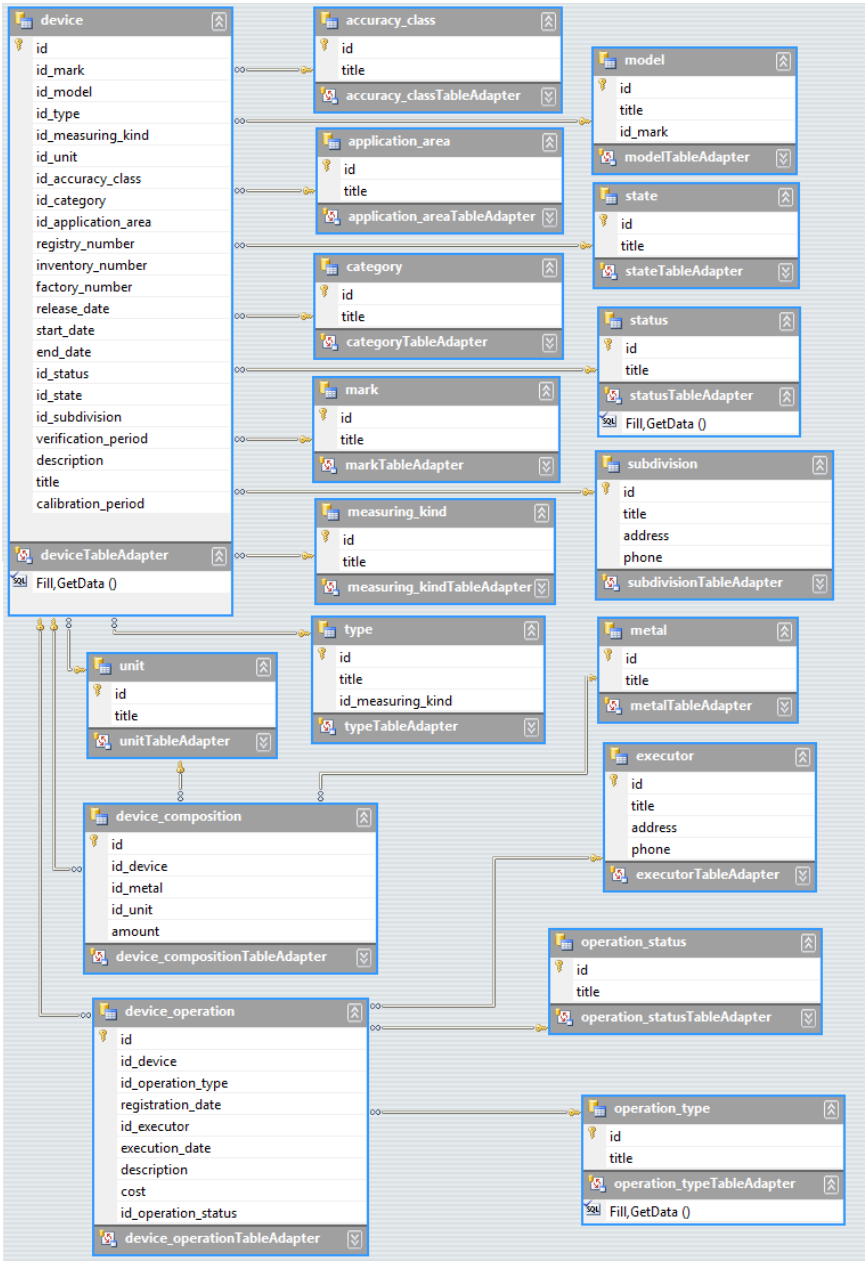


Рисунок ТЕХНОЛОГИЧЕСКИЙ РАЗДЕЛ.9 Модуль данных проекта

Объединяет в себе классы для обращения к таблицам базе данных. Для каждой интересующей нас таблицы в этом классе мы создается компоненты «TableAdapter» и «DataTable». Компонент «DataTable» определяет список полей набора данных, а компонент «TableAdapter» определяет четыре базовых sql-запроса (select, insert, update, delete) для работы с конкретной таблицей БД.

Диаграмма классов для модуля данных (рисунок 3.5):

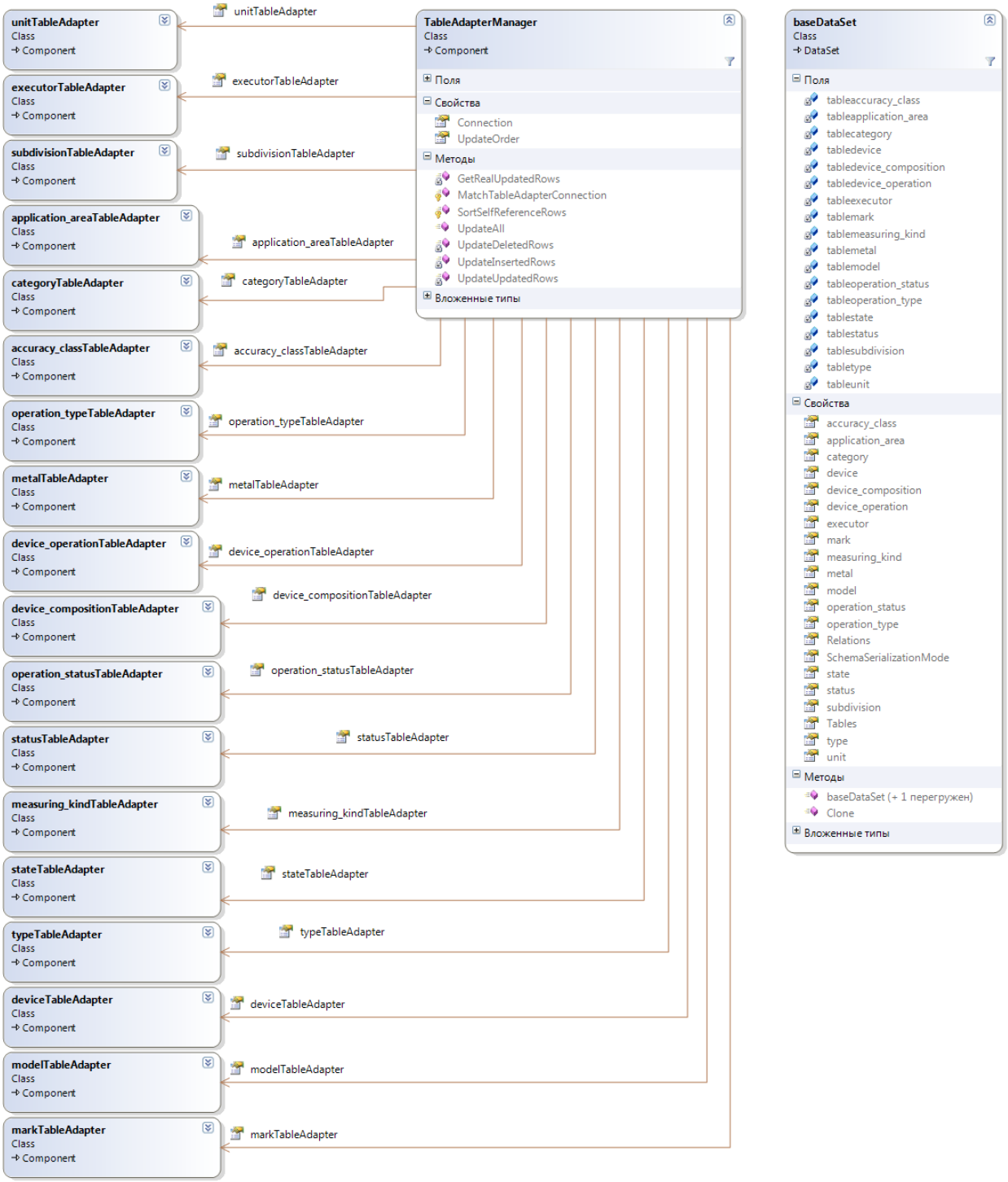


Рисунок ТЕХНОЛОГИЧЕСКИЙ РАЗДЕЛ.10 Диаграмма классов для модуля данных

BaseDataSet

Объединяет в себе наборы данных. Для каждой интересующей нас таблицы базы данных в этом классе мы создается компоненты «DataTable». Компонент «DataTable» определяет структуру набора данных.

TableAdapterManager

Объединяет в себе компоненты-адаптеры, созданные разработчиком внутри модуля данных.

deviceTableAdapter

Класс-адаптер для таблицы базы данных «device» (список всех СИ). Реализует в себе четыре sql-запроса (select, insert, update, delete) к целевой таблице. Позволяет выгружать данные из таблицы базы данных в компоненты, наследуемые от класса «System.Data.DataTable», через который программист может редактировать таблицу базы данных. Метод «Fill» адаптера заполняет «DataTable». Метод «Update» адаптера заносит данные из «DataTable» в базу данных.

device_operationTableAdapter

Класс-адаптер для таблицы базы данных «device_operation» (список работ проводимых над СИ). Реализует в себе четыре sql-запроса (select, insert, update, delete) к целевой таблице. Позволяет выгружать данные из таблицы базы данных в компоненты, наследуемые от класса «System.Data.DataTable», через который программист может редактировать таблицу базы данных. Метод «Fill» адаптера заполняет «DataTable». Метод «Update» адаптера заносит данные из «DataTable» в базу данных.

device_compositionTableAdapter

Класс-адаптер для таблицы базы данных «device_composition» (описание содержания драгоценных металлов в СИ). Реализует в себе четыре sql-запроса (select, insert, update, delete) к целевой таблице. Позволяет выгружать данные из таблицы базы данных в компоненты, наследуемые от класса «System.Data.DataTable», через который программист может редактировать таблицу базы данных. Метод «Fill» адаптера заполняет «DataTable». Метод «Update» адаптера заносит данные из «DataTable» в базу данных.

и т.д. (согласно структуре базы данных) для всех остальных классов, имеющих суффикс «TableAdapter».

3.3 Руководство пользователя

Рассмотрим порядок работы в системе.

После запуска программы пользователь видит главное меню программы.

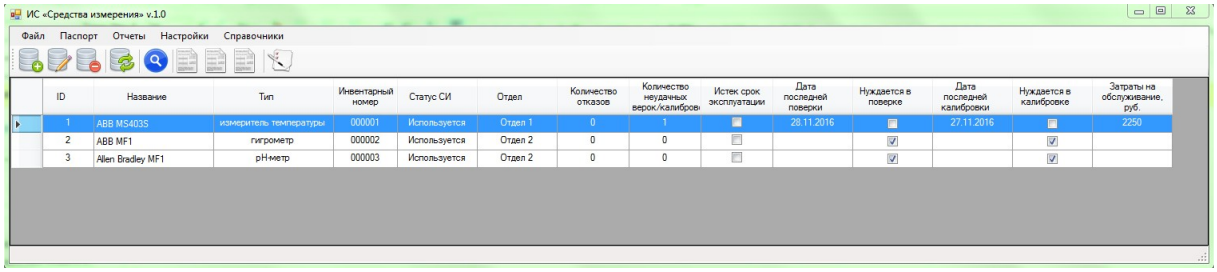


Рисунок ТЕХНОЛОГИЧЕСКИЙ РАЗДЕЛ.11 Главное меню программы

Каждая экранная форма содержит стандартные элементы интерфейса, которые имеют пояснительную подпись.

Для формирования плана проведения операций пользователю необходимо нажать кнопку «Добавить».

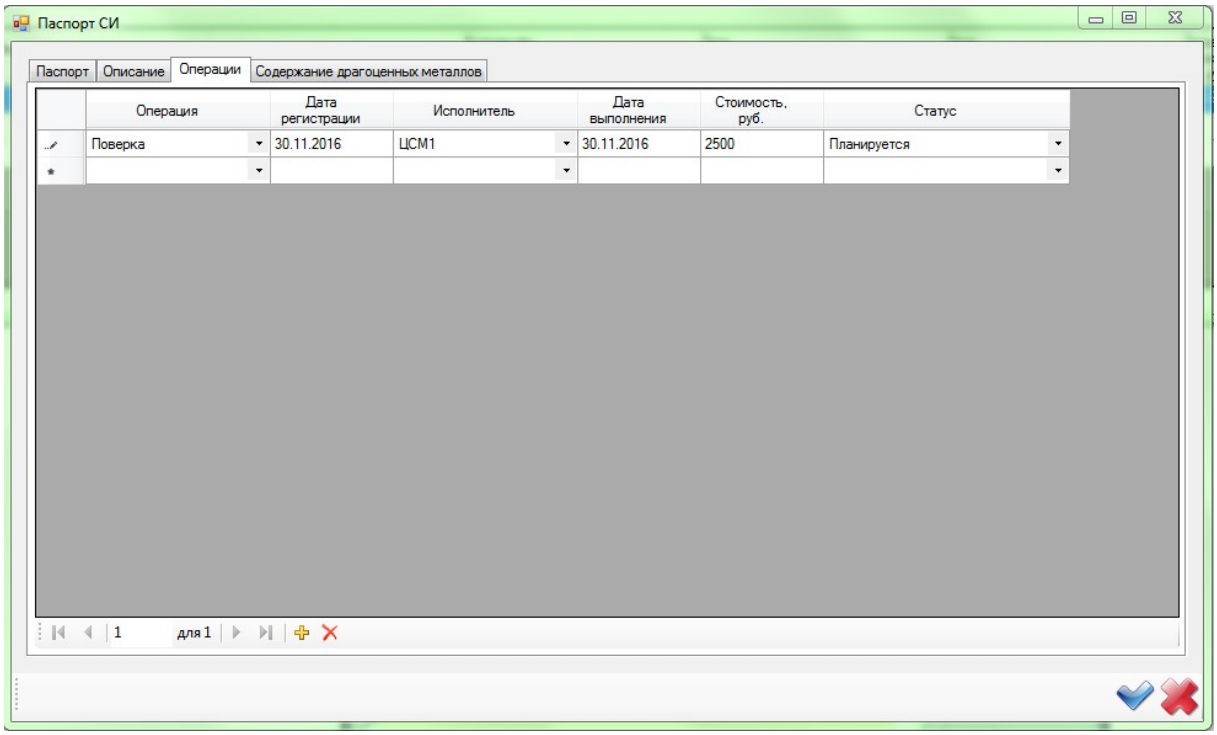


Рисунок ТЕХНОЛОГИЧЕСКИЙ РАЗДЕЛ.12 Добавление данных об операции с СИ

Последовательно вводя данные о СИ, типе операции, ответственном лице, описание (при необходимости), пользователь формирует паспорт СИ. Учет проведения операций осуществляется путем изменения ее статуса (рисунок 3.8).

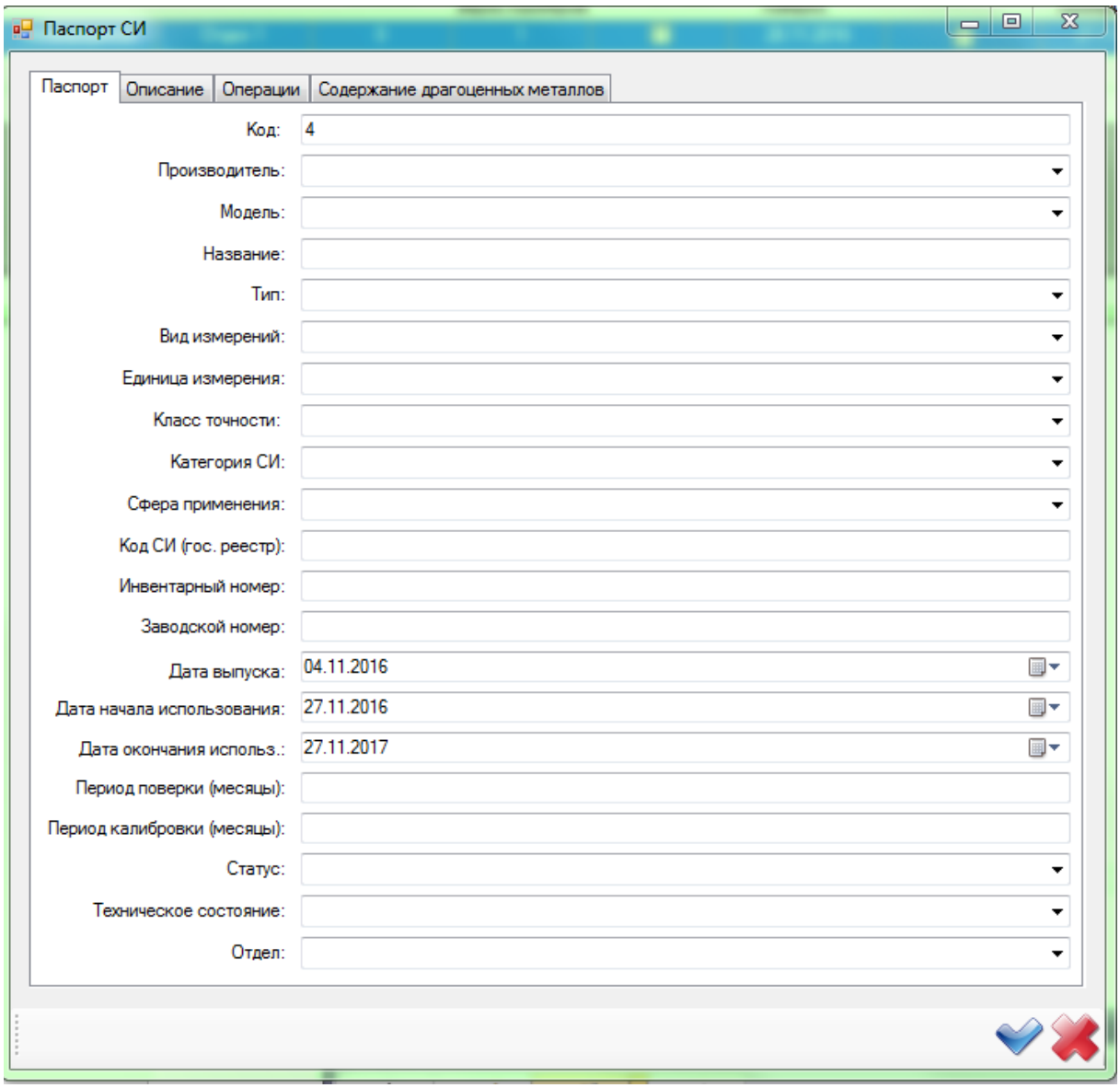


Рисунок ТЕХНОЛОГИЧЕСКИЙ РАЗДЕЛ.13 Паспорт СИ

После чего статус операции меняется. Ввод данных в справочники осуществляется после выбора справочника из выпадающего справочника (рисунок 3.9).

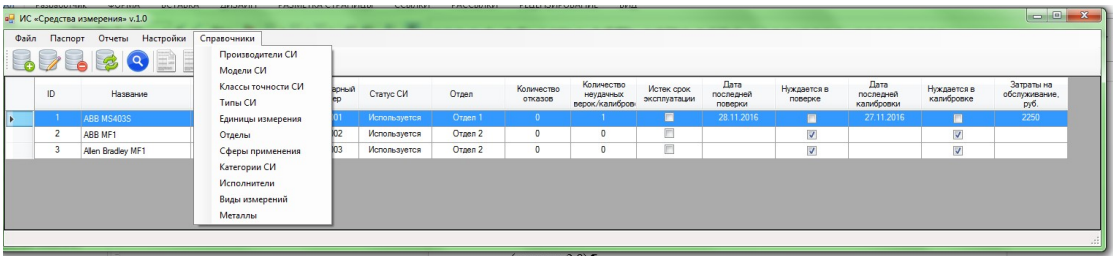


Рисунок ТЕХНОЛОГИЧЕСКИЙ РАЗДЕЛ.14 Выбор справочника

Для построения отчета необходимо перейти в соответствующий пункт меню и выбрать период получения отчета:

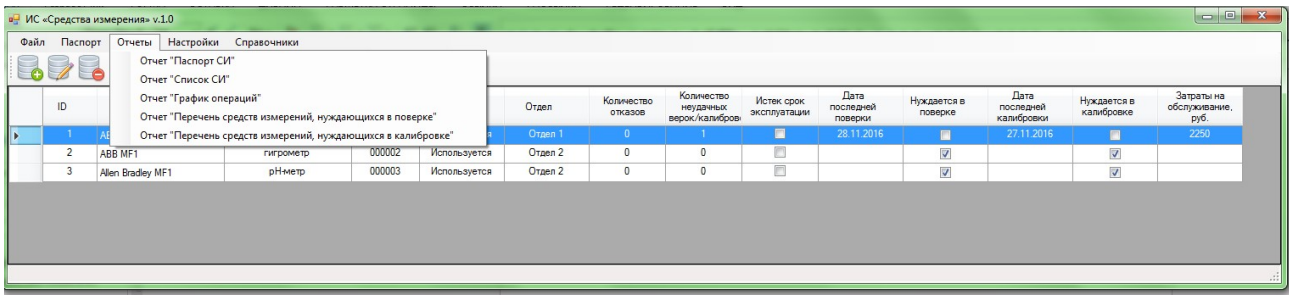
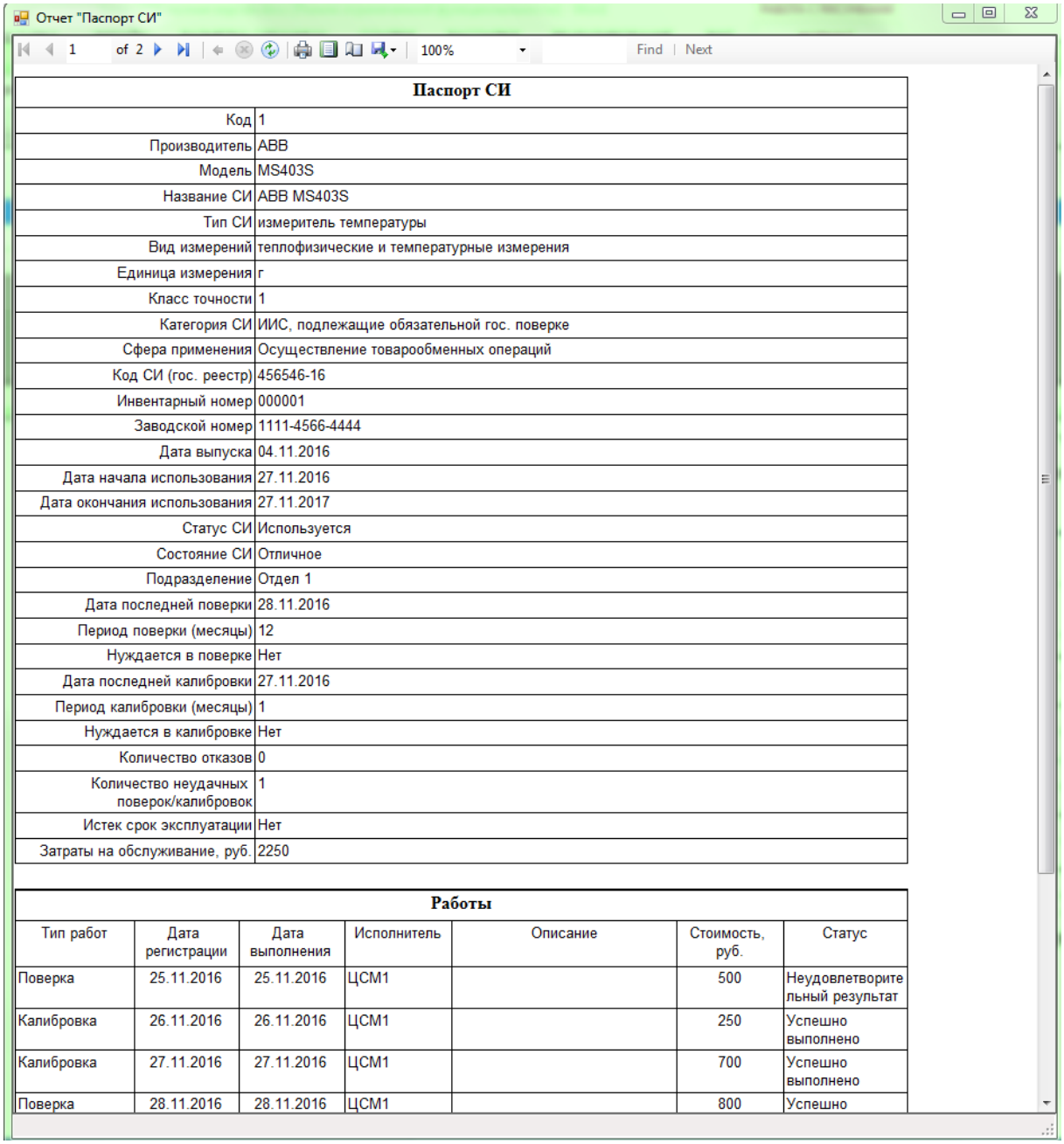


Рисунок ТЕХНОЛОГИЧЕСКИЙ РАЗДЕЛ.15 Список отчетов

3.4 После выбора периода необходимо нажать кнопку с наименованием интересующего отчета, в результате чего сформируется экранная форма отчета.



Паспорт СИ

Код	1
Производитель	ABB
Модель	MS403S
Название СИ	ABB MS403S
Тип СИ	измеритель температуры
Вид измерений	теплофизические и температурные измерения
Единица измерения	г
Класс точности	1
Категория СИ	ИИС, подлежащие обязательной гос. поверке
Сфера применения	Осуществление товарообменных операций
Код СИ (гос. реестр)	456546-16
Инвентарный номер	000001
Заводской номер	1111-4566-4444
Дата выпуска	04.11.2016
Дата начала использования	27.11.2016
Дата окончания использования	27.11.2017
Статус СИ	Используется
Состояние СИ	Отличное
Подразделение	Отдел 1
Дата последней поверки	28.11.2016
Период поверки (месяцы)	12
Нуждается в поверке	Нет
Дата последней калибровки	27.11.2016
Период калибровки (месяцы)	1
Нуждается в калибровке	Нет
Количество отказов	0
Количество неудачных поверок/калибровок	1
Истек срок эксплуатации	Нет
Затраты на обслуживание, руб.	2250

Работы

Тип работ	Дата регистрации	Дата выполнения	Исполнитель	Описание	Стоимость, руб.	Статус
Поверка	25.11.2016	25.11.2016	ЦСМ1		500	Неудовлетворительный результат
Калибровка	26.11.2016	26.11.2016	ЦСМ1		250	Успешно выполнено
Калибровка	27.11.2016	27.11.2016	ЦСМ1		700	Успешно выполнено
Поверка	28.11.2016	28.11.2016	ЦСМ1		800	Успешно

Рисунок ТЕХНОЛОГИЧЕСКИЙ РАЗДЕЛ.16 Отчет «Паспорт СИ»

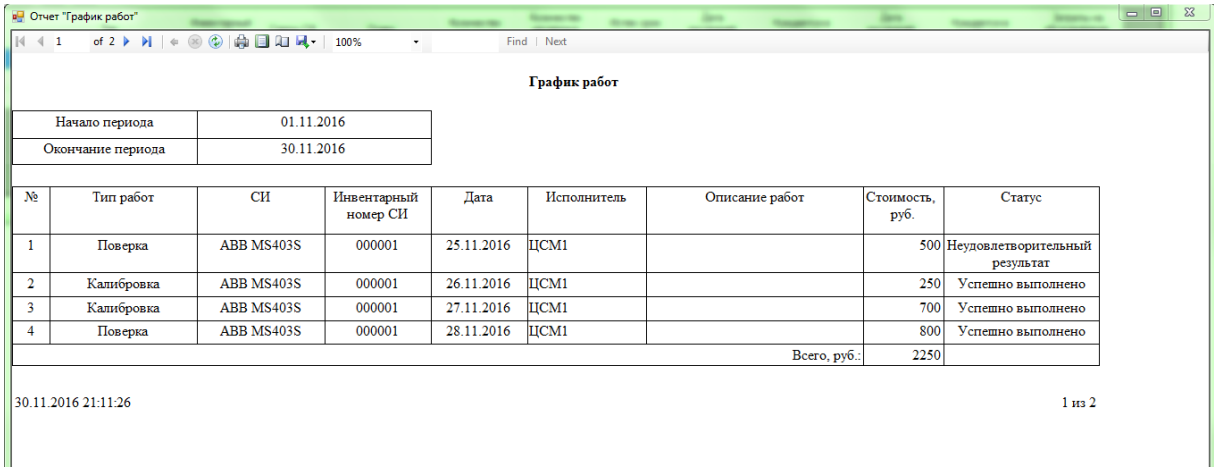


График работ

Начало периода	01.11.2016
Окончание периода	30.11.2016

№	Тип работ	СИ	Инвентарный номер СИ	Дата	Исполнитель	Описание работ	Стоимость, руб.	Статус
1	Поверка	ABB MS403S	000001	25.11.2016	ЦСМ1		500	Неудовлетворительный результат
2	Калибровка	ABB MS403S	000001	26.11.2016	ЦСМ1		250	Успешно выполнено
3	Калибровка	ABB MS403S	000001	27.11.2016	ЦСМ1		700	Успешно выполнено
4	Поверка	ABB MS403S	000001	28.11.2016	ЦСМ1		800	Успешно выполнено
						Всего, руб.:	2250	

30.11.2016 21:11:26 1 из 2

Рисунок ТЕХНОЛОГИЧЕСКИЙ РАЗДЕЛ.17 Отчет «График работ»

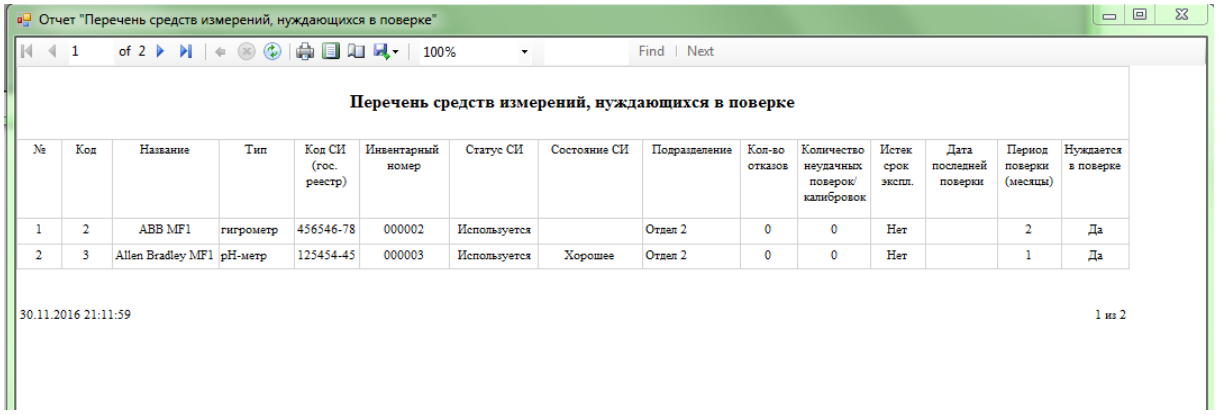


Рисунок ТЕХНОЛОГИЧЕСКИЙ РАЗДЕЛ.18 Перечень средств, нуждающихся в поверке

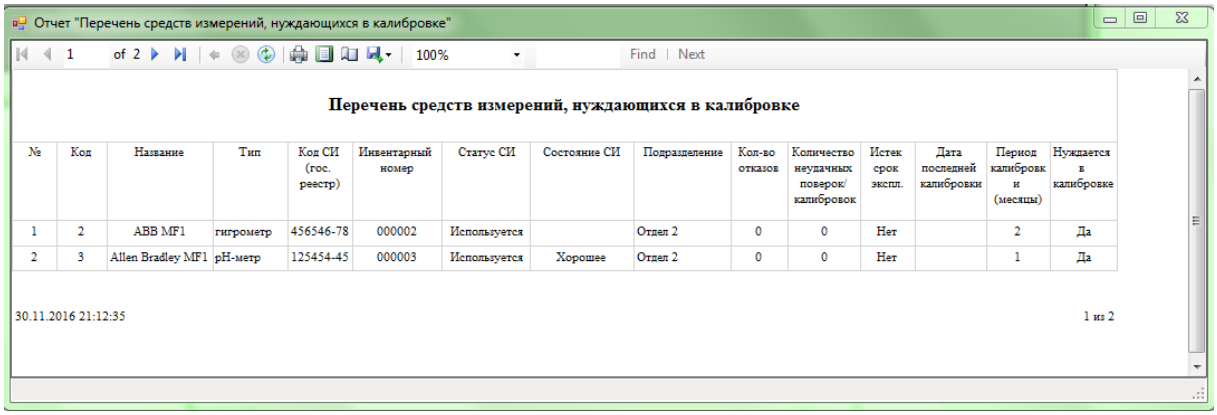


Рисунок ТЕХНОЛОГИЧЕСКИЙ РАЗДЕЛ.19 Перечень средств, нуждающихся в калибровке

4 ОБОСНОВАНИЕ ЭКОНОМИЧЕСКОЙ ЭФФЕКТИВНОСТИ ПРОЕКТА

4.1 Выбор и обоснование методики расчёта экономической эффективности

Под оценкой экономической рентабельности ИС принято понимать процесс, включающий в себя обоснование, измерение, определение того, насколько нужным и эффективным в экономическом плане стало для компании внедрение ИС. При всем этом, полезность рассматривается в виде обычного денежного эквивалента той разницы доходов/расходов компании в результате инвестиций в новую ИС.

Методом подсчета эффективности ИС можно назвать способ или средство для проведения полноценной оценки ИС. Такие способы обычно состоят как из формальных, так и неформальных процедур, при этот неформальными являются не основанные на цифровых данных, субъективные и быстрые процедуры оценки. Формальные же более рациональные, объективные, базирующиеся на простых данных механизмы оценки.

Сегодня внедрение современной ИС – дело достаточно дорогое. Нахождение компаний в рыночном секторе требует, как минимум, анализа возможных экономических последствий, а лучше – полной оценки эффективности каждого шага в процессе преобразования системы управления компанией.

Оценить экономическую эффективность ИС – дело сложное и трудоемкое, требует множества не только технических, но и экономических навыков. И лишь совокупность этих двух составляющих позволит привести к достоверному и точному результату анализа.

Продвигать на рынке ИС в условиях бешеной конкуренции нельзя без оценки ожидаемой экономической эффективности системы. Кроме того статистическая оценка качества и рентабельности внедрения часто (от 40-70% случаев) характеризует неудачным внедрением.

Специалисты разработки, продвижения и сопровождения ИС должны уметь проводить предварительную экспертизу нового проекта. Также их задачей является постоянный мониторинг системы на совпадение внедряемых технологий и

стратегий развития компании. И процесс оценки затрат и достигаемого эффекта должен быть непосредственно процессом – процедурой, которую важно проводить на протяжении всего этапа разработки и внедрения системы, чтобы потом реально влиять на продолжение роста.

Обычно выделяют следующие этапы оценивания экономической эффективности ИС:

- Классическая оценка эффективности в соотношении затрат и результатов;
- Подсчет совокупной стоимости владения данной системой;
- Оценка ИС как инвестиционного проекта;
- Установка системы сбалансированных показателей для оценивания экономического роста.

Оценку можно рассмотреть на основе всех единых принципов (технического, технологического, отраслевого, финансового, регионального). И к этому можно отнести:

- Обзор проекта на всем жизненном цикле;
- Моделирование финансовых потоков;
- Выявление условий сравнения нескольких проектов;
- Эффективность и положительность;
- Учет временных затрат;
- Учет будущих затрат и поступлений от проекта;
- Анализ ситуации с проектом и без него;
- Учет всех возможных последствий проекта;
- Анализ различных участников проекта;
- Многоэтапное построение оценки;
- Влияние инвестиционных проектов;
- Влияние инфляции;
- Возможное действие рисков и неопределенностей.

Данные коммерческой эффективности проекта могут отразить последствия внедрения ИС. Но для расчета рыночной эффективности проекта принято использовать следующие:

- Чистая прибыль;
- Чистый дисконтированный доход;
- Стандартная норма доходности;
- Параметры доходности затрат и инвестиций;
- Период окупаемости.

В итоге исходя из всего сказанного можно делать вывод, что сам процесс расчета экономической эффективности очень сложен и не всегда однозначен. Для каждого случая есть свой индивидуальный подход. Но нельзя забывать о некоторых методиках, которые исключают человеческий фактор и снижают погрешность отсутствия нужных данных.

В нашем случае, т.к. система создается не для продажи, а для реализации на одном предприятии, нужно рассчитать экономическую эффективность исходя из снижения затрат на производство. Во время ее расчета нужно сравнить результаты обработки данных, полученные при стандартном бизнес-процессе и после установки разработанной системы. Прямая эффективность автоматизированной обработки демонстрируется в показателе уменьшения экономических стоимостных затрат на обработку данных. В процесс оценки прямой эффективности с единицах стоимости проводятся две группы показателей – снижение трудозатрат и снижение стоимости.

Для расчета изменения трудозатрат на обработку данных применяется такая система параметров:

1. Абсолютный показатель уменьшения трудозатрат на обработку информации

$$\Delta T=T_0-T_1 \qquad (4.1.1)$$

где T_0 является годовой трудоемкостью обработки данных в базисном варианте;

T_1 определяется как годовая стоимость обработки данных в проектируемом варианте.

2. Параметр снижения трудозатрат

$K_т=(\Delta T/T_0)*100$ (%) (4.1.2)

3. Величина снижения трудозатрат, показывающая рост производительности труда в обработке данных.

$Y_т=T_0/T_{1в}$ (4.1.3)

К показателям стоимости обычно относят абсолютное уменьшение стоимости (ΔC), параметр относительного уменьшения стоимости затрат (K_C), индекс уменьшения стоимости затрат (Y_C):

1. Параметр снижения стоимости затрат

$\Delta C=C_0-C_1$ (4.1.4)

где C_0 определяется как годовая стоимость обработки данных в базисном варианте;

C_1 является годовой стоимостью обработки данных в проектируемом варианте.

2. Величина эффективности затрат:

$K_c=(\Delta C/C_0)*100$ (%) (4.1.5)

3. Параметр изменения стоимости затрат

$Y_c=C_0/C_1$ (4.1.6)

Кроме приведенных выше показателей рекомендуется также рассчитывать время окупаемости затрат на реализацию проекта автоматизированной обработки данных ($T_{ок}$), рассчитываемый обычно в годах, частях года или месяцах года:

$Tок = K_П /\Delta C$ (4.1.7),

где $K_П$ представляют затраты на реализацию проекта (проектирование и внедрение).

А также расчетный коэффициент рентабельности капитальных затрат:

$Eр = 1/Tок$ (4.1.8)

4.2 Расчёт показателей экономической эффективности проекта

В соответствии с выбранной выше методикой произведем расчет основных показателей экономической эффективности разработанной информационной системы.

Для этого необходимо оценить объемы обрабатываемой информации в базовом и проектном варианте. Для расчета стоимостных показателей учтем, что заработная плата сотрудника составляет около 45000 рублей. Так как в рабочей неделе содержится 5 рабочих дней по 8 часов, рабочих дней в месяце в среднем – 22, стоимость одного часа работы составит:

45000/(22*8)=267 рублей в час.

Как было указано выше, основным документом в автоматизированной задаче является наряд на работы. Объем данного документа в бумажном эквиваленте составляет около 20 документострок. В рабочий день оформляется около 10 нарядов, что в год составляет:

20*10*20*12=48000 документострок.

Кроме того, при работе сотрудника необходимо учитывать следующие документы (таблица 4.1):

Таблица ОБОСНОВАНИЕ ЭКОНОМИЧЕСКОЙ ЭФФЕКТИВНОСТИ ПРОЕКТА.24

Список и объем обрабатываемых документов

Наименование	Объем в день, д/с	Объем в год, д/с
Список средств измерения;	150	39600
Список сотрудников;	25	6600
Список оборудования;	10	2640
Отчет по выполненным поверкам за период.	300	79200
Отчет по выполненным калибровкам за период.	250	66000
Отчет по выполнению графика поверок;	20	5280
Отчет по выполнению графика калибровок;	250	66000
Список средств измерения со сроками проведения калибровок и поверок.	150	39600

Кроме того, учтем также, что до использования разработанной системы существуют накладные расходы, выражающиеся в необходимости закупки

канцелярских принадлежностей, а именно (в год):

Бумага – 30 пачек по 120 рублей = 3600 рублей, в час – 3600/(20*8*12)=1,875 рублей.

А при использовании разработанной системы необходимо учесть амортизацию компьютера, которая составит 25% от его первоначальной стоимости (16000 рублей) при сроке эксплуатации 5 лет, то есть в час:

4000/(20*12*8)=2 рубля в час.

Тогда операции технологического процесса при базовом и проектном варианте за год и их характеристики представлены в таблице 4.1 и таблице 4.2

Таблица ОБОСНОВАНИЕ ЭКОНОМИЧЕСКОЙ ЭФФЕКТИВНОСТИ ПРОЕКТА.25

Базовый вариант

№ п/п	Наименование операций технологического процесса решения комплекса задач	Оборудование	Ед. Изм.	Объем работы в год	Норма выработки / (опер/в час.)	Тру доемкость	Среднечасовая зарплата специалиста (руб.)	Часовая норма аморти-зации (руб. за час) / ст. 1 маш.часа (руб.)	Часовая стоимость накладных расходов (руб.)	Стоимостные затраты для ручных операций
1.	Список средств измерения;	нет	д/с	39600	10	3960	267,86	0	1,88	1060714,29
2.	Список сотрудников;	нет	д/с	6600	10	660	267,86	0	1,88	176785,71
3.	Список оборудования;	нет	д/с	2640	50	52,8	267,86	0	1,88	14142,86
4.	Отчет по выполненным поверкам за период.	нет	д/с	79200	50	1584	267,86	0	1,88	424285,71
5.	Отчет по выполненным калибровкам за период.		д/с	66000	100	660	267,86	0	1,88	176785,71
6.	Отчет по выполнению графика поверок;		д/с	5280	250	21,12	267,86	0	1,88	5657,14
7.	Отчет по выполнению графика калибровок;		д/с	66000	250	264	267,86	0	1,88	70714,29
8.	Список средств измерения со сроками проведения калибровок и поверок.		д/с	39600	150	264	267,86	0	1,88	70714,29
	Итого:					**7465,92**				**1999800,00**

Таблица ОБОСНОВАНИЕ ЭКОНОМИЧЕСКОЙ ЭФФЕКТИВНОСТИ ПРОЕКТА.26

Проектный вариант

№ п/п	Наименование операций технологического процесса решения комплекса задач	Оборудование	Ед. Изм.	Объем работы в год	Норма выработки / (опер/в час.)	Трудоемкость	Средне-часовая зарплата специалиста (руб.)	Часовая норма амортизации (руб. за час)	Часовая стоимость накладных расходов (руб.)	Стоимостные затраты для ручных операций
1.	Список средств измерения;	ПК	д/с	39600	10	3960	267,8571	2	2	1060714,29
2.	Список сотрудников;	ПК	д/с	6600	10	660	267,8571	2	2	176785,71
3.	Список оборудования;	ПК	д/с	2640	10	264	267,8571	2	2	70714,29
4.	Отчет по выполненным поверкам за период.	ПК	д/с	79200	250	316,8	267,8571	2	2	84857,14
5.	Отчет по выполненным калибровкам за период.	ПК	д/с	66000	300	220	267,8571	2	2	58928,57
6.	Отчет по выполнению графика поверок;	ПК	д/с	5280	350	15,086	267,8571	2	2	4040,82
7.	Отчет по выполнению графика калибровок;	ПК	д/с	66000	350	188,57	267,8571	2	2	50510,20
8.	Список средств измерения со сроками проведения калибровок и поверок.	ПК	д/с	39600	300	132	267,8571	2	2	35357,14
	Итого:					5756				1541908,16

Абсолютный показатель снижения трудовых затрат на обработку информации:

ΔТ=7465,92 - 5756,46=1709,46часов

Коэффициент снижения трудовых затрат

$К_т$=(1709/7465)*100%=22 %

Показатель снижения стоимостных затрат

ΔС=1999800,00 -1541908,16=457891,84рублей

Рассчитаем затраты на внедрение и проектирование системы, для чего учтем заработную плату разработчиков. При средней заработной плате 27000 , привлечении двух разработчиков и общем сроке разработки 20 рабочих дней (месяц), принимаем стоимость разработки – 54000 рублей.

Срок окупаемости затрат на внедрение проекта машинной обработки информации:

$Т_{ок}$ = 54000/457891=0,14 года или около 2 месяцев.

Рассчитаем расчетный коэффициент эффективности капитальных затрат:

Ер = 1 / Ток=1/0,11 =0,7

На рис. 4.1. приведена диаграмма сравнения базового и проектного вариантов трудовых затрат, на рис. 4.2 – стоимостных затрат.



Рис. 4.1 Диаграмма сравнения базового и проектного варианта трудовых затрат обработки информации.

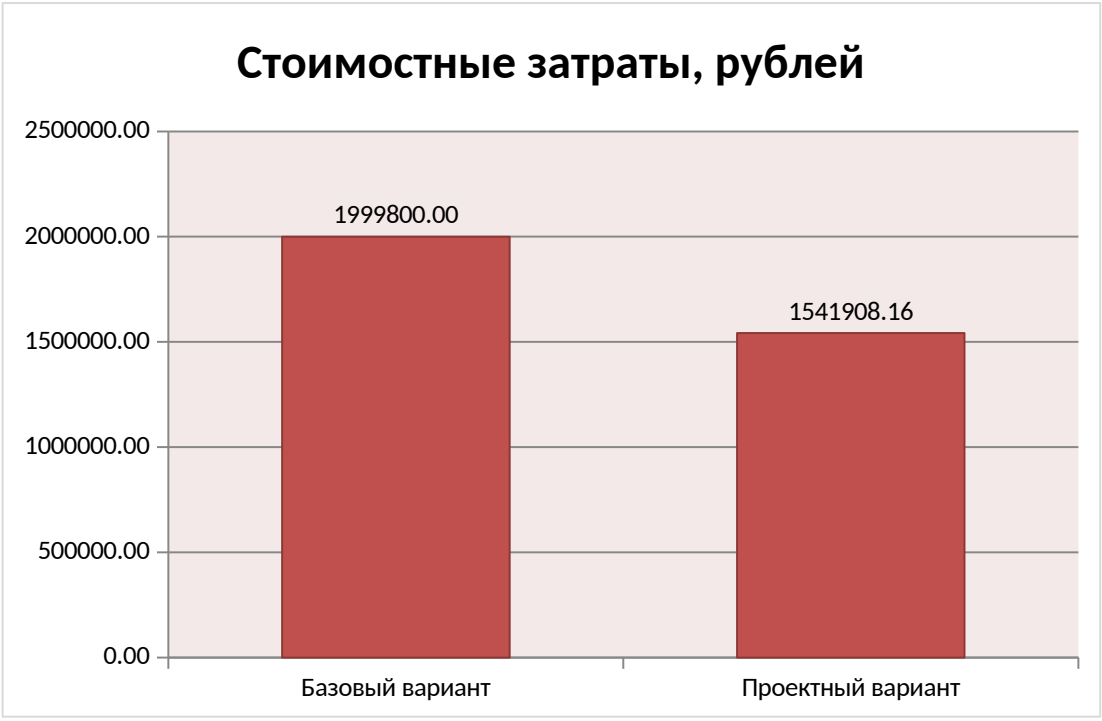


Рис. 4.2 Диаграмма сравнения базового и проектного варианта

ЗАКЛЮЧЕНИЕ

В результате проведенной работы ясно, что основные задачи дипломного проекта — анализ деятельности предприятия, выявление существующих недостатков в текущей технологии управления предприятием, разработка информационной подсистемы метрологического учета, в дипломной работе выполнены.

В данной дипломной работе рассмотрена деятельность метрологического отдела предприятия. В работе рассмотрена организационно-штатная структура управления предприятием, особо выделены функции выполнения метрологического учета.

В ходе анализа деятельности предприятия было выяснено, что узким местом является подготовка отчетности по выполненным метрологическим операциям со средствами измерения – поверками и калибровками. Было выяснено, что в результате использования некорректных бизнес-процессов на предприятии крайне неэффективно используется рабочее время сотрудников метрологического отдела.

В работе подробно описана экономическая сущность задачи, проведен анализ существующих программных аналогов, проведена формализация расчета показателей.

Далее проведены обоснования по техническому, информационному, технологическому и программному обеспечению задачи.

При рассмотрении требований к техническому обеспечению стало ясно, что добавления какого-либо аппаратного обеспечения в целях выполнения автоматизируемой задачи в существующую информационную систему не требуется.

Далее приведен примерный состав справочников и классификаторов системы, а также, в ходе сравнения и анализа, выбраны язык программирования и система управления базами данных.

В проектной части в ходе описания информационного обеспечения задачи описана информационная модель, классификаторы и системы кодирования, дана

характеристика входной, нормативно-справочной информации, результатной информации.

При описании программного обеспечения задачи приведено дерево функций системы, сценарий диалога пользователя и системы, дана характеристика базы данных, в том числе показана ее ER-диаграмма и приведено описание реквизитного состава каждой таблицы (наименование поля, его тип и назначение), дано описание структурной схемы пакета и программных модулей.

Разработанная информационная система доступна пользователям с любым уровнем подготовки, не требует специальных знаний для ее использования.

Разработанный проект в полной мере соответствует требованиям задания на дипломное проектирование.

В ходе разработки использовался язык программирования C#, популярная система управления базами данных Mysql, что позволяет внедрить разработанную систему на любом предприятии с аналогичными рассмотренному бизнес-процессами.

.

СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ

1. Алекс Эллайн: C++. От ламера до программера (включая C++11), - М., Издательство: Питер, 2015 г., 480 стр.
2. Аникеев С. П., Маркин Н.В.: Разработка приложений баз данных в Delphi. Самоучитель, - М.,Диалог-МИФИ, 2013 г., 160 с.
3. Арнольд Виллемер: Программирование на C++, - М., Эксмо, 2013 г., 528 с.
4. Брюс Тейт: Семь языков за семь недель. Практическое руководство по изучению языков программирования, - М., Издательство: ДМК-Пресс, 2014 г., 384 стр.
5. Васвани В.А.: MySQL: использование и администрирование, СПб, Питер, 2011 г., 368 с.
6. Васильев А.В: Самоучитель C++ с примерами и задачами, - М., Наука и Техника, 2015 г., 480 с.
7. Вигерс, Битти: Разработка требований к программному обеспечению, - М., Издательство: BHV, 2014 г., 736 стр.
8. Герберт Шилдт: C++ для начинающих, - М., Эком, 2011 г., 640 с.
9. ГолощаповА. Р.: MicrosoftVisualStudio 2011, - М., BHV, 2011 г., 544 с.
10. ГурвицГ.Е.: Microsoft Access 2011. Разработка приложений на реальном примере, - М., BHV, 2011 г., 424 с.
11. Дейтел, Дейтел: Как программировать на Visual C# 2012. Включая работу на Windows 7 и Windows 8, - М., Питер, 2014 г., 864 с.
12. Джо Майо: Самоучитель MicrosoftVisualStudio 2011, - М., BHV, 2011 г., 464 с.
13. Исаев Г.А.: Информационные системы в экономике. Учебник, - М., Омега-Л, 2013 г., 462 с.
14. Исаев Г.А.: Проектирование информационных систем. Учебное пособие, - М., Омега-Л, 2015 г., 424 с.

15. ИэнГриффитс: Программирование на C# 5.0, - М., Эксмо, 2014 г., 1135 с.

16. Курлов А.А, Петров Е.А.: Методология информационной аналитики, - М.,Проспект, 2014 г., 384 с.

17. Мартынов Н.А: Программирование для Windows на C\C++. В 2-х томах, - М., Бином, 2013 г., 480 с.

18. Ошероув Р.О: Искусство автономного тестирования с примерами на C#, - М., ДМК-Пресс, 2014 г., 360 с.

19. Паттерсон, Хеннесси: Архитектура компьютера и проектирование компьютерных систем М. Классика ComputersScience, Спб Питер, , 2012 г., 784 с.

20. Пирогов В.А.: Информационные системы и базы данных: организация и проектирование, - М., BHV, 2009 г., 528 с.

21. Полубенцева - М. П.: C/C++ Процедурное программирование, - М., BHV, 2014 г., 432 с.

22. Потопахин В.С.: Искусство алгоритмизации, - М., Издательство: ДМК-Пресс, 2014 г., 320 стр.

23. Пугачев Е.К., Шериев Ш.К., Кичинский Е.А.: Разработка приложений для Windows 8 на языке C#, - М., BHV, 2013 г., 416 с.

24. Роберт Дж. Мюллер, Проектирование баз данных и UML, - М., Лори, 2013 год, 432 с.

25. Роберт Лафоре: Объектно-ориентированное программирование в C++, - М., Питер, 2013 г., 928 с.

26. СурядныйА.Е.: Microsoft Access 2011. Лучший самоучитель, - М., Астрель, 2012 г., 448 с.

27. Таненбаум А.А, Бос Е.В.: Современные операционные системы, Спб, Питер, 2015 г., 1120 стр

28. Тахагхогхи, Вильямс: Руководство по MySQL, - М., BHV, 2011 г., 544 с.

29. Тимофеев В. А: Самоучитель C++ как он есть, - М., Бином, 2009 г., 336 с.

30. Тюгашев А. Е.: Языки программирования. Учебное пособие. Стандарт третьего поколения, - М., Издательство: Питер, 2014 г. 336 стр.

31. Ховард, Лебланк, Виега: Как написать безопасный код на C++, Java, Perl, PHP, ASP.NET, - М., ДМК-Пресс, 2014 г., 288 с.

32. Чистов Д.А: Экономическая информатика (для бакалавров). Учебное пособие, - М., Кнорус, 2014 г., 512 с.

33. Эндрю Стиллмен: Изучаем C#, - М., Питер, 2014 г., 816 с.

34. Энтони Молинаро, SQL. Сборник рецептов, - М., Символ-Плюс,2011 г., 672 с.

35. Энтони Уильямс: Параллельное программирование на C++ в действии. Практика разработки многопоточных программ, - М., ДМК-Пресс, 2014 г., 672 с.

ПРИЛОЖЕНИЕ. ЛИСТИНГ ПРОГРАММНЫХ МОДУЛЕЙ

```
namespace Project
{
    partial class ApplicationAreaForm
    {
            this.tsbCancel = new System.Windows.Forms.ToolStripButton();
            this.tsbOk = new System.Windows.Forms.ToolStripButton();
            this.baseDataSet = new Project.baseDataSet();
            this.application_areaBindingSource = new
System.Windows.Forms.BindingSource(this.components);
            this.application_areaTableAdapter = new
Project.baseDataSetTableAdapters.application_areaTableAdapter();
            this.tableAdapterManager = new
Project.baseDataSetTableAdapters.TableAdapterManager();
            ((System.ComponentModel.ISupportInitialize)
(this.baseDataSet)).BeginInit();
            ((System.ComponentModel.ISupportInitialize)
(this.application_areaBindingSource)).BeginInit();
            ((System.ComponentModel.ISupportInitialize)
(this.application_areaDataGridView)).BeginInit();
            this.SuspendLayout();
            //
            // tsMain
            //
            this.tsMain.Dock = System.Windows.Forms.DockStyle.Bottom;
            this.tsMain.ImageScalingSize = new System.Drawing.Size(32, 32);
            this.tsMain.Items.AddRange(new System.Windows.Forms.ToolStripItem[] {
            this.tsbCancel,
            this.tsbOk});
            this.tsMain.Location = new System.Drawing.Point(0, 292);
            this.tsMain.Name = "tsMain";
            this.tsMain.Size = new System.Drawing.Size(523, 39);
            this.tsMain.TabIndex = 13;
            this.tsMain.Text = "toolStrip1";
            //
            // tsbCancel
            //
            this.tsbCancel.Alignment =
System.Windows.Forms.ToolStripItemAlignment.Right;
            this.tsbCancel.DisplayStyle =
System.Windows.Forms.ToolStripItemDisplayStyle.Image;
            this.tsbCancel.Image =
global::Project.Properties.Resources.button_cancel_7517;
            this.tsbCancel.ImageTransparentColor = System.Drawing.Color.Magenta;
            this.tsbCancel.Name = "tsbCancel";
            this.tsbCancel.Size = new System.Drawing.Size(36, 36);
            this.tsbCancel.Text = "Отмена";
            this.tsbCancel.Click += new System.EventHandler(this.tsbCancel_Click);
            //
            // tsbOk
            //
            this.tsbOk.Alignment = System.Windows.Forms.ToolStripItemAlignment.Right;
            this.tsbOk.DisplayStyle =
System.Windows.Forms.ToolStripItemDisplayStyle.Image;
            this.tsbOk.Image = global::Project.Properties.Resources.apply_1005;
            this.tsbOk.ImageTransparentColor = System.Drawing.Color.Magenta;
            this.tsbOk.Name = "tsbOk";
            this.tsbOk.Size = new System.Drawing.Size(36, 36);
            this.tsbOk.Text = "Сохранить";
```

```
            this.tsbOk.Click += new System.EventHandler(this.tsbOk_Click);
            //
            // baseDataSet
            //
            this.baseDataSet.DataSetName = "baseDataSet";
            this.baseDataSet.SchemaSerializationMode =
System.Data.SchemaSerializationMode.IncludeSchema;
            //
            // application_areaBindingSource
            //
            this.application_areaBindingSource.DataMember = "application_area";
            this.application_areaBindingSource.DataSource = this.baseDataSet;
            //
            // application_areaTableAdapter
            //
            this.application_areaTableAdapter.ClearBeforeFill = true;
            //
            // tableAdapterManager
            //
            this.tableAdapterManager.accuracy_classTableAdapter = null;
            this.tableAdapterManager.application_areaTableAdapter =
this.application_areaTableAdapter;
            this.tableAdapterManager.BackupDataSetBeforeUpdate = false;
            this.tableAdapterManager.categoryTableAdapter = null;
            this.tableAdapterManager.device_compositionTableAdapter = null;
            this.tableAdapterManager.device_operationTableAdapter = null;
            this.tableAdapterManager.device_status_historyTableAdapter = null;
            this.tableAdapterManager.deviceTableAdapter = null;
            this.tableAdapterManager.executorTableAdapter = null;
            this.tableAdapterManager.markTableAdapter = null;
            this.tableAdapterManager.measuring_kindTableAdapter = null;
            this.tableAdapterManager.metalTableAdapter = null;
            this.tableAdapterManager.modelTableAdapter = null;
            this.tableAdapterManager.operation_statusTableAdapter = null;
            this.tableAdapterManager.operation_typeTableAdapter = null;
            this.tableAdapterManager.stateTableAdapter = null;
            this.tableAdapterManager.statusTableAdapter = null;
            this.tableAdapterManager.subdivisionTableAdapter = null;
            this.tableAdapterManager.typeTableAdapter = null;
            this.tableAdapterManager.unitTableAdapter = null;
            this.tableAdapterManager.UpdateOrder =
Project.baseDataSetTableAdapters.TableAdapterManager.UpdateOrderOption.InsertUpdateDe
lete;
            //
            // application_areaDataGridView
            //
            this.application_areaDataGridView.AutoGenerateColumns = false;
            this.application_areaDataGridView.ColumnHeadersHeightSizeMode =
System.Windows.Forms.DataGridViewColumnHeadersHeightSizeMode.AutoSize;
            this.application_areaDataGridView.Columns.AddRange(new
System.Windows.Forms.DataGridViewColumn[] {
            this.dataGridViewTextBoxColumn1,
            this.dataGridViewTextBoxColumn2});
            this.application_areaDataGridView.DataSource =
this.application_areaBindingSource;
            this.application_areaDataGridView.Dock =
System.Windows.Forms.DockStyle.Fill;
            this.application_areaDataGridView.Location = new System.Drawing.Point(0,
0);
            this.application_areaDataGridView.Name = "application_areaDataGridView";
            this.application_areaDataGridView.Size = new System.Drawing.Size(523,
292);
            this.application_areaDataGridView.TabIndex = 14;
```

```
            //
            // dataGridViewTextBoxColumn1
            //
            this.dataGridViewTextBoxColumn1.DataPropertyName = "id";
            this.dataGridViewTextBoxColumn1.HeaderText = "id";
            this.dataGridViewTextBoxColumn1.Name = "dataGridViewTextBoxColumn1";
            this.dataGridViewTextBoxColumn1.Width = 64;
            //
            // dataGridViewTextBoxColumn2
            //
            this.dataGridViewTextBoxColumn2.DataPropertyName = "title";
            this.dataGridViewTextBoxColumn2.HeaderText = "title";
            this.dataGridViewTextBoxColumn2.Name = "dataGridViewTextBoxColumn2";
            this.dataGridViewTextBoxColumn2.Width = 350;
            //
            // ApplicationAreaForm
            //
            this.AutoScaleDimensions = new System.Drawing.SizeF(6F, 13F);
            this.AutoScaleMode = System.Windows.Forms.AutoScaleMode.Font;
            this.ClientSize = new System.Drawing.Size(523, 331);
            this.Controls.Add(this.application_areaDataGridView);
            this.Controls.Add(this.tsMain);
            this.Name = "ApplicationAreaForm";
            this.StartPosition = System.Windows.Forms.FormStartPosition.CenterScreen;
            this.Text = "Сферы применения";
            this.Load += new System.EventHandler(this.ApplicationAreaForm_Load);
            this.tsMain.ResumeLayout(false);
            this.tsMain.PerformLayout();
            ((System.ComponentModel.ISupportInitialize)(this.baseDataSet)).EndInit();
            ((System.ComponentModel.ISupportInitialize)
(this.application_areaBindingSource)).EndInit();
            ((System.ComponentModel.ISupportInitialize)
(this.application_areaDataGridView)).EndInit();
            this.ResumeLayout(false);
            this.PerformLayout();

        }

        #endregion

        private System.Windows.Forms.ToolStrip tsMain;
        private System.Windows.Forms.ToolStripButton tsbCancel;
        private System.Windows.Forms.ToolStripButton tsbOk;
        private baseDataSet baseDataSet;
        private System.Windows.Forms.BindingSource application_areaBindingSource;
        private baseDataSetTableAdapters.application_areaTableAdapter
application_areaTableAdapter;
        private baseDataSetTableAdapters.TableAdapterManager tableAdapterManager;
        private System.Windows.Forms.DataGridView application_areaDataGridView;
        private System.Windows.Forms.DataGridViewTextBoxColumn
dataGridViewTextBoxColumn1;
        private System.Windows.Forms.DataGridViewTextBoxColumn
dataGridViewTextBoxColumn2;
    }
}



namespace Project
{
    partial class CategoryForm
    {
        /// Required method for Designer support - do not modify
```

```
        /// the contents of this method with the code editor.
        /// </summary>
        private void InitializeComponent()
        {
            this.components = new System.ComponentModel.Container();
            this.tsMain = new System.Windows.Forms.ToolStrip();
            this.tsbCancel = new System.Windows.Forms.ToolStripButton();
            this.tsbOk = new System.Windows.Forms.ToolStripButton();
            this.baseDataSet = new Project.baseDataSet();
            this.categoryBindingSource = new
System.Windows.Forms.BindingSource(this.components);
            this.categoryTableAdapter = new
Project.baseDataSetTableAdapters.categoryTableAdapter();
            this.tableAdapterManager = new
Project.baseDataSetTableAdapters.TableAdapterManager();
            this.categoryDataGridView = new System.Windows.Forms.DataGridView();
            this.dataGridViewTextBoxColumn1 = new
System.Windows.Forms.DataGridViewTextBoxColumn();
            this.dataGridViewTextBoxColumn2 = new
System.Windows.Forms.DataGridViewTextBoxColumn();
            this.tsMain.SuspendLayout();
            ((System.ComponentModel.ISupportInitialize)
(this.baseDataSet)).BeginInit();
            ((System.ComponentModel.ISupportInitialize)
(this.categoryBindingSource)).BeginInit();
            ((System.ComponentModel.ISupportInitialize)
(this.categoryDataGridView)).BeginInit();
            this.SuspendLayout();
            //
            // tsMain
            //
            this.tsMain.Dock = System.Windows.Forms.DockStyle.Bottom;
            this.tsMain.ImageScalingSize = new System.Drawing.Size(32, 32);
            this.tsMain.Items.AddRange(new System.Windows.Forms.ToolStripItem[] {
            this.tsbCancel,
            this.tsbOk});
            this.tsMain.Location = new System.Drawing.Point(0, 334);
            this.tsMain.Name = "tsMain";
            this.tsMain.Size = new System.Drawing.Size(529, 39);
            this.tsMain.TabIndex = 10;
            this.tsMain.Text = "toolStrip1";
            //
            // tsbCancel
            //
            this.tsbCancel.Alignment =
System.Windows.Forms.ToolStripItemAlignment.Right;
            this.tsbCancel.DisplayStyle =
System.Windows.Forms.ToolStripItemDisplayStyle.Image;
            this.tsbCancel.Image =
global::Project.Properties.Resources.button_cancel_7517;
            this.tsbCancel.ImageTransparentColor = System.Drawing.Color.Magenta;
            this.tsbCancel.Name = "tsbCancel";
            this.tsbCancel.Size = new System.Drawing.Size(36, 36);
            this.tsbCancel.Text = "Отмена";
            this.tsbCancel.Click += new System.EventHandler(this.tsbCancel_Click);
            //
            // tsbOk
            //
            this.tsbOk.Alignment = System.Windows.Forms.ToolStripItemAlignment.Right;
            this.tsbOk.DisplayStyle =
System.Windows.Forms.ToolStripItemDisplayStyle.Image;
            this.tsbOk.Image = global::Project.Properties.Resources.apply_1005;
            this.tsbOk.ImageTransparentColor = System.Drawing.Color.Magenta;
```

```
            this.tsbOk.Name = "tsbOk";
            this.tsbOk.Size = new System.Drawing.Size(36, 36);
            this.tsbOk.Text = "Сохранить";
            this.tsbOk.Click += new System.EventHandler(this.tsbOk_Click);
            //
            // baseDataSet
            //
            this.baseDataSet.DataSetName = "baseDataSet";
            this.baseDataSet.SchemaSerializationMode =
System.Data.SchemaSerializationMode.IncludeSchema;
            //
            // categoryBindingSource
            //
            this.categoryBindingSource.DataMember = "category";
            this.categoryBindingSource.DataSource = this.baseDataSet;
            //
            // categoryTableAdapter
            //
            this.categoryTableAdapter.ClearBeforeFill = true;
            //
            // tableAdapterManager
            //
            this.tableAdapterManager.accuracy_classTableAdapter = null;
            this.tableAdapterManager.application_areaTableAdapter = null;
            this.tableAdapterManager.BackupDataSetBeforeUpdate = false;
            this.tableAdapterManager.categoryTableAdapter =
this.categoryTableAdapter;
            this.tableAdapterManager.device_compositionTableAdapter = null;
            this.tableAdapterManager.device_operationTableAdapter = null;
            this.tableAdapterManager.device_status_historyTableAdapter = null;
            this.tableAdapterManager.deviceTableAdapter = null;
            this.tableAdapterManager.executorTableAdapter = null;
            this.tableAdapterManager.markTableAdapter = null;
            this.tableAdapterManager.measuring_kindTableAdapter = null;
            this.tableAdapterManager.metalTableAdapter = null;
            this.tableAdapterManager.modelTableAdapter = null;
            this.tableAdapterManager.operation_statusTableAdapter = null;
            this.tableAdapterManager.operation_typeTableAdapter = null;
            this.tableAdapterManager.stateTableAdapter = null;
            this.tableAdapterManager.statusTableAdapter = null;
            this.tableAdapterManager.subdivisionTableAdapter = null;
            this.tableAdapterManager.typeTableAdapter = null;
            this.tableAdapterManager.unitTableAdapter = null;
            this.tableAdapterManager.UpdateOrder =
Project.baseDataSetTableAdapters.TableAdapterManager.UpdateOrderOption.InsertUpdateDe
lete;
            //
            // categoryDataGridView
            //
            this.categoryDataGridView.AutoGenerateColumns = false;
            this.categoryDataGridView.ColumnHeadersHeightSizeMode =
System.Windows.Forms.DataGridViewColumnHeadersHeightSizeMode.AutoSize;
            this.categoryDataGridView.Columns.AddRange(new
System.Windows.Forms.DataGridViewColumn[] {
            this.dataGridViewTextBoxColumn1,
            this.dataGridViewTextBoxColumn2});
            this.categoryDataGridView.DataSource = this.categoryBindingSource;
            this.categoryDataGridView.Dock = System.Windows.Forms.DockStyle.Fill;
            this.categoryDataGridView.Location = new System.Drawing.Point(0, 0);
            this.categoryDataGridView.Name = "categoryDataGridView";
            this.categoryDataGridView.Size = new System.Drawing.Size(529, 334);
            this.categoryDataGridView.TabIndex = 11;
            //
```

```
            // dataGridViewTextBoxColumn1
            //
            this.dataGridViewTextBoxColumn1.DataPropertyName = "id";
            this.dataGridViewTextBoxColumn1.HeaderText = "id";
            this.dataGridViewTextBoxColumn1.Name = "dataGridViewTextBoxColumn1";
            this.dataGridViewTextBoxColumn1.Width = 64;
            //
            // dataGridViewTextBoxColumn2
            //
            this.dataGridViewTextBoxColumn2.DataPropertyName = "title";
            this.dataGridViewTextBoxColumn2.HeaderText = "title";
            this.dataGridViewTextBoxColumn2.Name = "dataGridViewTextBoxColumn2";
            this.dataGridViewTextBoxColumn2.Width = 350;
            //
            // CategoryForm
            //
            this.AutoScaleDimensions = new System.Drawing.SizeF(6F, 13F);
            this.AutoScaleMode = System.Windows.Forms.AutoScaleMode.Font;
            this.ClientSize = new System.Drawing.Size(529, 373);
            this.Controls.Add(this.categoryDataGridView);
            this.Controls.Add(this.tsMain);
            this.Name = "CategoryForm";
            this.StartPosition = System.Windows.Forms.FormStartPosition.CenterScreen;
            this.Text = "Категории СИ";
            this.Load += new System.EventHandler(this.CategoryForm_Load);
            this.tsMain.ResumeLayout(false);
            this.tsMain.PerformLayout();
            ((System.ComponentModel.ISupportInitialize)(this.baseDataSet)).EndInit();
            ((System.ComponentModel.ISupportInitialize)
(this.categoryBindingSource)).EndInit();
            ((System.ComponentModel.ISupportInitialize)
(this.categoryDataGridView)).EndInit();
            this.ResumeLayout(false);
            this.PerformLayout();

        }

        #endregion

        private System.Windows.Forms.ToolStrip tsMain;
        private System.Windows.Forms.ToolStripButton tsbCancel;
        private System.Windows.Forms.ToolStripButton tsbOk;
        private baseDataSet baseDataSet;
        private System.Windows.Forms.BindingSource categoryBindingSource;
        private baseDataSetTableAdapters.categoryTableAdapter categoryTableAdapter;
        private baseDataSetTableAdapters.TableAdapterManager tableAdapterManager;
        private System.Windows.Forms.DataGridView categoryDataGridView;
        private System.Windows.Forms.DataGridViewTextBoxColumn
dataGridViewTextBoxColumn1;
        private System.Windows.Forms.DataGridViewTextBoxColumn
dataGridViewTextBoxColumn2;
    }
}

namespace Project
{
    public partial class DeviceForm : Form
    {
        private int id = 0;

        public DeviceForm()
        {
            InitializeComponent();
```

```
        }

        public int ID
        {
            get
            {
                return id;
            }
            set
            {
                id = value;
            }

        }


        private void DeviceForm_Load(object sender, EventArgs e)
        {
            // TODO: данная строка кода позволяет загрузить данные в таблицу
"baseDataSet.subdivision". При необходимости она может быть перемещена или удалена.
            this.subdivisionTableAdapter.Fill(this.baseDataSet.subdivision);
            // TODO: данная строка кода позволяет загрузить данные в таблицу
"baseDataSet.state". При необходимости она может быть перемещена или удалена.
            this.stateTableAdapter.Fill(this.baseDataSet.state);
            // TODO: данная строка кода позволяет загрузить данные в таблицу
"baseDataSet.status". При необходимости она может быть перемещена или удалена.
            this.statusTableAdapter.Fill(this.baseDataSet.status);
            // TODO: данная строка кода позволяет загрузить данные в таблицу
"baseDataSet.operation_status". При необходимости она может быть перемещена или
удалена.

this.operation_statusTableAdapter.Fill(this.baseDataSet.operation_status);
            // TODO: данная строка кода позволяет загрузить данные в таблицу
"baseDataSet.executor". При необходимости она может быть перемещена или удалена.
            this.executorTableAdapter.Fill(this.baseDataSet.executor);
            // TODO: данная строка кода позволяет загрузить данные в таблицу
"baseDataSet.operation_type". При необходимости она может быть перемещена или
удалена.
            this.operation_typeTableAdapter.Fill(this.baseDataSet.operation_type);
            // TODO: данная строка кода позволяет загрузить данные в таблицу
"baseDataSet.metal". При необходимости она может быть перемещена или удалена.
            this.metalTableAdapter.Fill(this.baseDataSet.metal);
            // TODO: данная строка кода позволяет загрузить данные в таблицу
"baseDataSet.device_composition". При необходимости она может быть перемещена или
удалена.

this.device_compositionTableAdapter.Fill(this.baseDataSet.device_composition);
            // TODO: данная строка кода позволяет загрузить данные в таблицу
"baseDataSet.device_operation". При необходимости она может быть перемещена или
удалена.

this.device_operationTableAdapter.Fill(this.baseDataSet.device_operation);
            // TODO: данная строка кода позволяет загрузить данные в таблицу
"baseDataSet.category". При необходимости она может быть перемещена или удалена.
            this.categoryTableAdapter.Fill(this.baseDataSet.category);
            // TODO: данная строка кода позволяет загрузить данные в таблицу
"baseDataSet.accuracy_class". При необходимости она может быть перемещена или
удалена.
            this.accuracy_classTableAdapter.Fill(this.baseDataSet.accuracy_class);
            // TODO: данная строка кода позволяет загрузить данные в таблицу
"baseDataSet.unit". При необходимости она может быть перемещена или удалена.
            this.unitTableAdapter.Fill(this.baseDataSet.unit);
```

```
            // TODO: данная строка кода позволяет загрузить данные в таблицу
"baseDataSet.measuring_kind". При необходимости она может быть перемещена или
удалена.
            this.measuring_kindTableAdapter.Fill(this.baseDataSet.measuring_kind);
            // TODO: данная строка кода позволяет загрузить данные в таблицу
"baseDataSet.type". При необходимости она может быть перемещена или удалена.
            this.typeTableAdapter.Fill(this.baseDataSet.type);
            // TODO: данная строка кода позволяет загрузить данные в таблицу
"baseDataSet.application_area". При необходимости она может быть перемещена или
удалена.

this.application_areaTableAdapter.Fill(this.baseDataSet.application_area);
            // TODO: данная строка кода позволяет загрузить данные в таблицу
"baseDataSet.model". При необходимости она может быть перемещена или удалена.
            this.modelTableAdapter.Fill(this.baseDataSet.model);
            // TODO: данная строка кода позволяет загрузить данные в таблицу
"baseDataSet.mark". При необходимости она может быть перемещена или удалена.
            this.markTableAdapter.Fill(this.baseDataSet.mark);
            // TODO: данная строка кода позволяет загрузить данные в таблицу
"baseDataSet.device". При необходимости она может быть перемещена или удалена.
            this.deviceTableAdapter.Fill(this.baseDataSet.device);

            if (id != 0)
            {
                deviceBindingSource.Filter = "id = '" + id + "'";
            }
            else
            {
                DataRowView Row = deviceBindingSource.AddNew() as DataRowView;
                id = (int)Row["код"];
            }

            if (this.deviceBindingSource.Current != null)
            {
                baseDataSet.deviceRow R = (this.deviceBindingSource.Current as
DataRowView).Row as baseDataSet.deviceRow;
                if (R.Isrelease_dateNull() == true)
                {
                    release_dateDateTimePicker.CustomFormat = " ";
                    release_dateDateTimePicker.Format = DateTimePickerFormat.Custom;
                }

                if (R.IsNull("start_date") == true)
                {
                    start_dateDateTimePicker.CustomFormat = " ";
                    start_dateDateTimePicker.Format = DateTimePickerFormat.Custom;
                }

                if (R.Isend_dateNull() == true)
                {
                    end_dateDateTimePicker.CustomFormat = " ";
                    end_dateDateTimePicker.Format = DateTimePickerFormat.Custom;
                }
            }
        }

        private void tsbOk_Click(object sender, EventArgs e)
        {
            this.Validate();
            this.deviceBindingSource.EndEdit();
            this.tableAdapterManager.UpdateAll(this.baseDataSet);
            this.DialogResult = System.Windows.Forms.DialogResult.OK;
        }
```

```
        private void tsbCancel_Click(object sender, EventArgs e)
        {
            this.DialogResult = System.Windows.Forms.DialogResult.Cancel;
        }

        private void release_dateDateTimePicker_ValueChanged(object sender, EventArgs
e)
        {
            (sender as DateTimePicker).Format = DateTimePickerFormat.Short;
        }

        private void device_operationBindingSource_AddingNew(object sender,
AddingNewEventArgs e)
        {

        }
    }
}

namespace Project
{
    public partial class ExecutorForm : Form
    {
        public ExecutorForm()
        {
            InitializeComponent();
        }

        private void executorBindingNavigatorSaveItem_Click(object sender, EventArgs
e)
        {

        }

        private void ExecutorForm_Load(object sender, EventArgs e)
        {
            // TODO: данная строка кода позволяет загрузить данные в таблицу
"baseDataSet.executor". При необходимости она может быть перемещена или удалена.
            this.executorTableAdapter.Fill(this.baseDataSet.executor);

        }

        private void tsbCancel_Click(object sender, EventArgs e)
        {
            DialogResult = System.Windows.Forms.DialogResult.Cancel;
        }

        private void tsbOk_Click(object sender, EventArgs e)
        {
            this.Validate();
            this.executorBindingSource.EndEdit();
            this.tableAdapterManager.UpdateAll(this.baseDataSet);
            DialogResult = System.Windows.Forms.DialogResult.OK;
        }
    }
}
```