Санкт-Петербургский государственный политехнический университет

Факультет экономики и менеджмента

Кафедра информационных систем в экономике и менеджменте

ДИПЛОМНЫЙ ПРОЕКТ

Тема: Разработка модулей интеграции информационно-измерительной системы Neftemer с экономическими информационными системами потенциального заказчика.

Студент гр. 6074/41 О.В. Ненашев

Санкт-Петербург

2011

Санкт-Петербургский государственный политехнический университет

Факультет экономики и менеджмента

Кафедра информационных систем в экономике и менеджменте

Проект допущен к защите
Зав. кафедрой, проф., д.э.н.
_____И.В. Ильин
«____»_____ 2011 г.

ДИПЛОМНЫЙ ПРОЕКТ

Тема: Разработка модулей интеграции информационно-измерительной системы Neftemer с экономическими информационными системами потенциального заказчика.

Направление:	080800 -	Прикладная информатика
Специальность:	080801 -	Прикладная информатика в экономике
Специализация:	080810 -	Информационные системы в инвестиционной деятельности

Выполнил студент гр. 6074/41	_____	О.В. Ненашев
Руководитель проекта, к.в.н., доцент	_____	А.Б. Анисифоров
Рецензент директор ООО “Комплекс-Ресурс”	_____	Д.В. Кратиров

Санкт-Петербург

2011

Санкт-Петербургский государственный политехнический университет

Факультет экономики и менеджмента

Кафедра Информационные системы в экономике и менеджменте

УТВЕРЖДАЮ
«___»_____ 2011 г.
Зав. кафедрой д.э.н., профессор
_____ И.В. Ильин

ЗАДАНИЕ

на выполнение дипломного проекта

студенту Ненашеву Олегу Вячеславовичу

1. Тема дипломного проекта *Разработка модулей интеграции информационно-измерительной системы Neftemer с экономическими информационными системами потенциального заказчика.*

2. Срок сдачи студентом законченного проекта *25.05.2011*

3. Исходные данные для дипломного проекта *Внутренняя документация компании ООО "Комплекс-Ресурс", открытая информация из сети Интернет, комплекс стандартов по ЕСПД*

4. Содержание дипломного проекта (перечень подлежащих разработке вопросов*) Характеристика ООО "Комплекс-Ресурс"; Обзор предыдущих версий системы; Анализ требований заказчика; Обзор стандартных средств межпрограммного взаимодействия ИИС; Формирование требований к интерфейсной части системы; Организация процесса разработки системы в ООО "Комплекс-Ресурс"; Разработка архитектуры системы; Разработка механизмов взаимодействия ИС с экономическими ИС предприятия; Разработка спецификации на программную часть системы с обзором используемых технологий и программных средств; Разработка и реализация коммуникационных модулей системы.*

5. Перечень графического материала *Структура ООО "Комплекс-Ресурс"; Диаграмма развёртывания предыдущих вариантов системы; Принципы построения распределённой ИС*; *Архитектура разработанной информационной системы; Диаграмма развёртывания системы на предприятии в различных конфигурациях; Архитектура системы с указанием модулей, разработкой которых занимался Ненашев О.В.; Статистические данные для оценки объёма выполненной работы.*

6. Консультанты (с указанием относящихся к ним разделов)
Анисифорова Л.О.(доцент каф. ИСЭМ) - архитектура ИС; Кратиров Д.В. (директор ООО "Комплекс-Ресурс") - общее согласование представляемого материала; Ростова О.В. (к.э.н, доцент) – Нормоконтроль

7. Календарный график работы:
Выбор темы дипломного проекта ……..… до 20.03
Выдача задания на проект ……...……до 25.03
Отчет студента о работе:
- анализ литературных источников …… до 01.04
- написание проекта и промежуточное представление руководителю рабочих материалов …….. до 01.05
- сдача готового проекта руководителю ……..... до 20.05
- сдача проекта для утверждения допуска к защите ...…….. до 25.05
- защита дипломного проекта ..…….…….…. до 10.11

8. Дата выдачи задания *"6" апреля 2011*

Руководитель _____ А.Б. Анисифоров
(подпись) (Ф.И.О.)

Задание принял к исполнению _____ (дата)
_____*О.В. Ненашев* (подпись студента)

РЕФЕРАТ

Отчёт, 102 стр., 18 рис., 19 табл., 31 ист., 2 прил.

БАЗЫ ДАННЫХ, ИНФОРМАЦИОННО-ИЗМЕРИТЕЛЬНАЯ СИСТЕМА, МОНИТОРИНГ ДОБЫЧИ НЕФТИ, СИСТЕМЫ СОВМЕСТНОЙ РАЗРАБОТКИ, РАЗРАБОТКА ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ, СИСТЕМНАЯ АРХИТЕКТУРА, ЭКОНОМИЧЕСКАЯ ИНФОРМАЦИОННАЯ СИСТЕМА

В работе рассмотрены некоторые этапы разработки распределённой информационной системы по анализу потоков нефти ИС Neftemer. Данная система была разработана фирмой ООО “Комплекс-Ресурс” в рамках проекта “Канада”. ИС Neftemer является сложным программно-аппаратным комплексом, но в работе уделено внимание только разработке общей архитектуры системы, организации процессов разработки и отдельным модулям, при помощи которых осуществляется взаимодействие с информационными системами заказчика.

В работе описаны принципы добычи и мониторинга нефти, приведена информация о компании ООО “Комплекс-Ресурс”, проекте “Канада”. Приведена классификация проекта и указаны задачи, которые решал автор во время разработки системы. На основании требований заказчика разработаны архитектура ИИС Neftemer и спецификации на разработку отдельных модулей.

В ходе работы в ООО “Комплекс-Ресурс” внедрены средства организации совместной работы: системы управления задачами, версиями, хранения библиографии. Данные системы интегрированы в общий процесс разработки. Рассмотрены вопросы построения расширяемой архитектуры ИС Neftemer, подробна рассмотрены сервер данных системы и внешняя база данных.

Произведён анализ соответствия разработанной системы поставленным требованиям, экономической эффективности внедрения разработанной системы и средств организации совместной разработки. Указаны основные пути дальнейшего развития системы. По итогам работы сделан вывод, что система соответствует поставленным техническим требованиям и маркетинговым требованиям, благодаря чему она быть внедрена у потенциального заказчика.

СОДЕРЖАНИЕ

ПЕРЕЧЕНЬ СОКРАЩЕНИЙ

БД	-	Блок Детектирования
ГК	-	Главный Контроллер Блока Детектирования в ИИС Neftemer
ЕСПД	-	Единая Система Программной Документации
ИИС	-	Информационно-Измерительная Система
ИУС	-	Информационно-Управляющая Система
ИС	-	Информационная Система
КК	-	Коммуникационный Контроллер (Аппаратный модуль ИИС Neftemer)
ООО	-	Общество с Ограниченной Ответственностью
ОС	-	Операционная система
ПК	-	Персональный Компьютер
ПО	-	Программное Обеспечение
СКВ	-	Система Контроля Версий
СПБГПУ	-	Санкт-Петербургский Государственный Политехнический Университет
СПКРПО	-	Система Поддержки Командной Разработки Программного Обеспечения
СПО	-	Системное Программное Обеспечение
СУБД	-	Система Управления Базами Данных
ЭВМ	-	Электронная Вычислительная Машина
ЭИС	-	Экономическая Информационная Система
CRM	-	Customer Relationship Management System Система управления взаимоотношениями с клиентами
ECM	-	Enterprise Content Management - Система управления информационными ресурсами предприятия
ERP	-	Enterprise Resource Planning System - Система планирования ресурсов предприятия
IDE	-	Integrated Development Environment - Интегрированная среда разработки
I2C	-	Inter-Integrated Circuit Interface - Интерфейс взаимодействия между интегральными системами
MVS2005	-	Microsoft Visual Studio 2005
MS	-	Microsoft Corporation

SCADA	-	Supervisory Control And Data Acquisition Диспетчерское управление и сбор данных
SQL	-	Structured Query Language - Язык структурированных запросов
TCP	-	Transmission Control Protocol
UML	-	Unified Modeling Language – Унифицированный язык моделирования
VCS	-	Version Control System – Система контроля версий
VPN	-	Virtual Private Network - Виртуальная частная сеть
XML	-	eXtensible Markup Language - Расширяемый язык разметки
XSD	-	XML Schema Description – Спецификация структуры XML-документов
WebDAV	-	Web-based Distributed Authoring and Versioning – Сетевой протокол файлового обмена с поддержкой совместного доступа и версионирования.

ВВЕДЕНИЕ

В настоящее время нефтяная отрасль является одной из самых значимых в мировой экономике. Нефть - это уникальная смесь множества углеводородов различного строения, её компоненты основные компоненты используются для производства топлива, асфальтов и пластмасс. В тоже время, нефть содержит множество миноритарных составляющих, которые также используются в химической промышленности [8, с.8]. Поэтому, нефть действительно является “чёрным золотом” для экономики.

К составу нефти предъявляются достаточно жёсткие требования, которые вызваны экологическими нормами и технологическими процессами. Поэтому, учёт количества и качества нефти - одна из важнейших задач, которая встаёт перед нефтедобывающими компаниями.

В современном мире конечная экспертиза состава нефти производится в процессе её продажи. Однако транспортировка и продажа нефти большими объёмами, из-за чего происходит смешивание сырья с различных участков добычи. При этом становится невозможным выявление причины возникающих отклонений от требований. Поэтому, большинство нефтедобывающих компаний производят дополнительный контроль на месте добычи. При этом, должны производиться централизованные обработка и хранение данных.

Для решения подобных задач строятся распределённые информационные системы, которые одновременно решают как информационно-измерительные, транспортные и экономические задачи. Одной из подобных систем является программно-аппаратный комплекс “ИИС Neftemer”, разработанный в ООО “Комплекс-Ресурс”, составляющим которой посвящён данный проект.

Автор дипломного проекта работал в ООО “Комплекс-Ресурс” на должности инженера-программиста в течение двух с половиной лет. В течение этого времени он участвовал во всех этапах разработки системы. Дипломный проект содержит информацию о некоторых составляющих системы, которые имеют непосредственное отношение к экономическим информационным

системам (ЭИС) и наиболее интересны с точки зрения специальности “Прикладная информатика в экономике”.

Разработка ИИС “Neftemer” велась в соответствии с современными методологиями проектирования (Scrum, ICONIX и др.) и принципами построения распределённых информационных систем. В рамках проекта был составлен полный комплекс программной документации, а разработанная система прошла комплекс приёмо-сдаточных испытаний у заказчика. Таким образом, основным результатом дипломного проекта является сложная информационная система, для которой был завершён основной цикл разработки и подтверждена практическая применимость системы.

Дипломный проект состоит из пяти частей. В первой части приведена информация о принципах добычи нефти, ООО “Комплекс-Ресурс”, описаны причины появления проекта “Канада”, в ходе которого велась разработка ИИС Neftemer.

Во второй части рассмотрен процесс формирования концепции системы. Проведён анализ требований, на основании которых были сформированы задачи на разработку системы.

Третья часть описывает процесс внедрения систем для поддержки совместной разработки в ООО “Комплекс-Ресурс”. Решение данной задачи было необходимым с учётом сложности проекта и распределённости команды разработки Были внедрены системы управления задачами, контроля версий исходных кодов и управления библиографией. Также на базе Redmine был построен аналог CRM-системы (Customer Relationship Management System) для взаимодействия с заказчиками.

Чётвертая часть описывает процесс разработки отдельных элементов системы: внешних баз данных и нескольких модулей интеграции с системами заказчика. Кратко описаны реализации отдельных модулей, выбранные средства разработки и подходы к тестированию системы.

В пятой части подводятся общие итоги разработки. Произведена оценка соответствия системы требованиям заказчика, анализ экономической эффективности внедрения систем совместной разработки, а также описан предполагаемый эффект от дальнейшего внедрения ИИС Neftemer. Также произведён анализ вклада автора в проект “Канада”.

В рамках контракта был заключён договор о неразглашении информации, поэтому в дипломном проекте невозможно привести подробную информацию о заказчике проекта, его информационных системах и пр. Также в отчёте и приложениях отсутствуют архитектура конечной системы, техническое задание на разработку, UML-спецификация и программная документация.

Тем не менее, практически весь комплекс документации на систему был разработан автором дипломного проекта при взаимодействии с заказчиком и сторонними специалистами. Успешная сдача первых версий системы заказчику подтверждает качество данной документации.

1. ВВЕДЕНИЕ В ДИПЛОМНЫЙ ПРОЕКТ

В данном разделе собрана вводная информация, необходимая для дальнейшего восприятия дипломного проекта неподготовленным читателем. В пункте 1.1 описан процесс добычи нефти и организация информационных систем на современных предприятиях. Эти данные лежат в основе всех информационных систем, работающих в нефтедобывающей отрасли.

В пунктах 1.1.4 и 1.3 рассказано об ООО “Комплекс-Ресурс” и проекте “Канада”, которому посвящён дипломный проект. Указана история фирмы, предыстория проекта. Также проведён анализ ранее существовавших систем, по результатам которого сделан вывод о необходимости разработки новой системы.

В конце раздела приведена классификация проекта, указаны этапы развития, ресурсы и участники проекта. Также приведён краткий список задач, решение которых входило в обязанности автора данного проекта.

1.1. Введение в предметную область

1.1.1. Основные характеристики нефти[1]

Нефть (от перс. neft) — горючая маслянистая жидкость со специфическим запахом, распространённая в осадочной оболочке Земли. Она образуется вместе с газообразными углеводородами на глубинах более 1,2—2 км. Вблизи земной поверхности нефть преобразуется в густую мальту, полутвёрдый асфальт и другие [2].

Нефть относится к группе горных осадочных пород вместе с песками, глинами, известняками, каменной солью и др. Она обладает одним важным свойством – способностью гореть и выделять тепловую энергию. Среди других горючих ископаемых она имеет наивысшую теплотворную способность. Это делает её одним из наиболее ценных природных ресурсов.

[1] Материалы данного пункта являются компиляцией статей из [2] и [23]

Нефть сильно варьируется по плотности: от лёгкой (0,65—0,70 г/см3) до тяжёлой (0,98—1,05 г/см3). Пластовая нефть, находящаяся в залежах на значительной глубине, в различной степени насыщена газообразными углеводородами. По химическому составу нефть весьма разнообразна, и говорить о среднем составе можно лишь условно. Менее всего колеблется элементный состав: 82,5—87% углерода, 11,5—14,5% водорода, 0,05—0,35 кислорода, 0,001—5,3% серы и 0,001—1,8% азота. На рис. 1.1 приведена классификация нефтей, которая используется при переработке нефти [23].

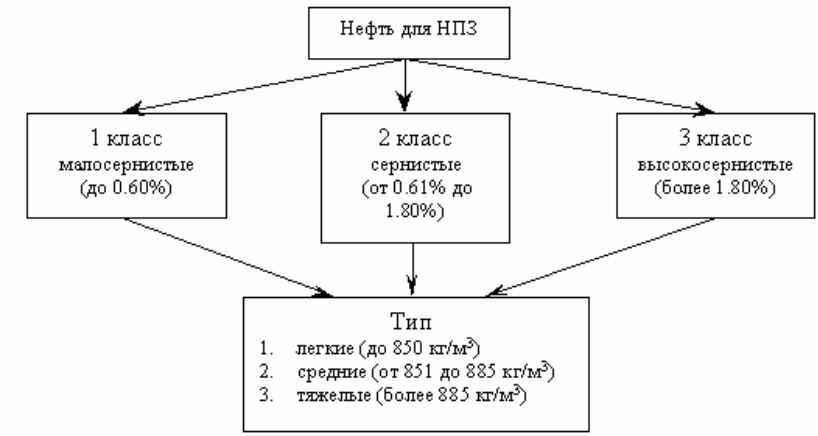

Рис. 1.1. Классификация нефтей с различных месторождений

Нефть содержит большое число различных примесей (воду, природных газ, минеральную серу и т.д.), которые должны быть отфильтрованы при добыче. Тем не менее, в случае возникновения неисправностей часть примесей может попасть в товарную нефть и отгружена заказчику. Это не только ухудшает качество поставляемых продуктов, но и может привести к серьёзным неполадкам у потребителей (нарушение технологического процесса, аварии). Таким образом, любое нефтедобывающее предприятие заинтересовано в том, чтобы поставляемая им нефть соответствовала заявленным характеристикам.

При добыче и транспортировке нефть смешивается, и становится невозможным локализовать причину возникающих отклонений. Поэтому, в

большинстве случаев анализаторы нефти ставятся непосредственно на точках добычи, что позволяет оперативно выявлять неисправности и предотвращать попадание некачественного продукта в общие транспортные сети.

1.1.2. Организация добычи нефти на современных предприятиях

В нефтедобывающей отрасли преобладают крупные компании, так как организация добычи и транспортировки нефти требует существенных затрат. Учитывая то, что залежи нефти рассеяны по всему мир, подобным компаниям приходится строить сложные логистические системы. Пример подобной компании приведён на рис. 1.2.

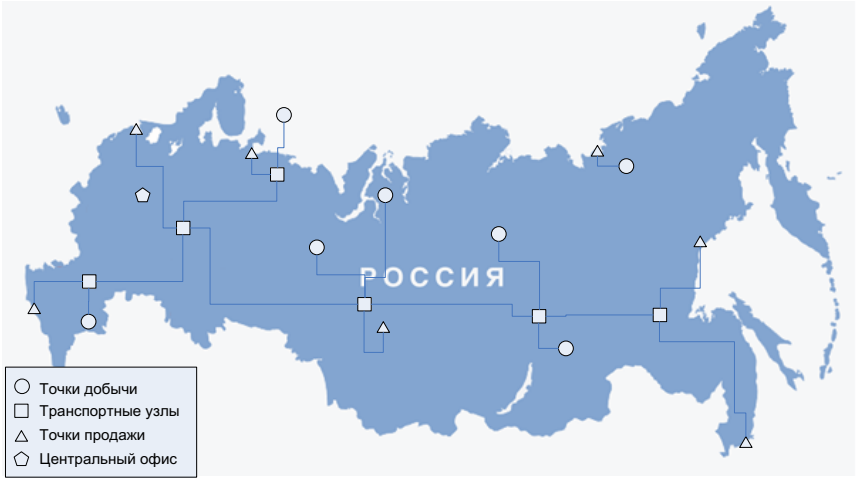

Рис. 1.2. Пример распределённой нефтедобывающей компании

Видно, что управление подробной системой требует правильно организованных бизнес процессов и невозможно без общего управления всеми узлами системы. Для решения подобных задач в современном мире используются специализированные информационные системы, которые будут рассмотрены позднее.

В рамках дипломного проекта нам также интересна внутренняя организация точек добычи. Участки добычи располагаются на месторождениях,

каждое из которых представляет собой ряд нефтесодержащих слоёв (пазух), доступ к которым ведётся раздельно (см. рис. 1.3а). В итоге, на каждом участке присутствует множество скважин различного назначения, датчиков (анализаторы потока нефти, метеорологическое оборудование и т.п.) и исполнительных механизмов (насосы, вентили и пр.), и необходима управление всей инфраструктурой на участке в соответствии с задачами, поставленными верхним уровнем управления (ИИС корпоративного уровня)

Для управления добычей на объекте ставятся локальные ИИС, которые связываются с верхним уровнем управления через оборудование телеметрии (рис. 1.3б). Таким образом, информационно-измерительные системы предприятия имеют иерархическую структуру, и на каждом уровне решаются свои задачи управления и обработки данных.

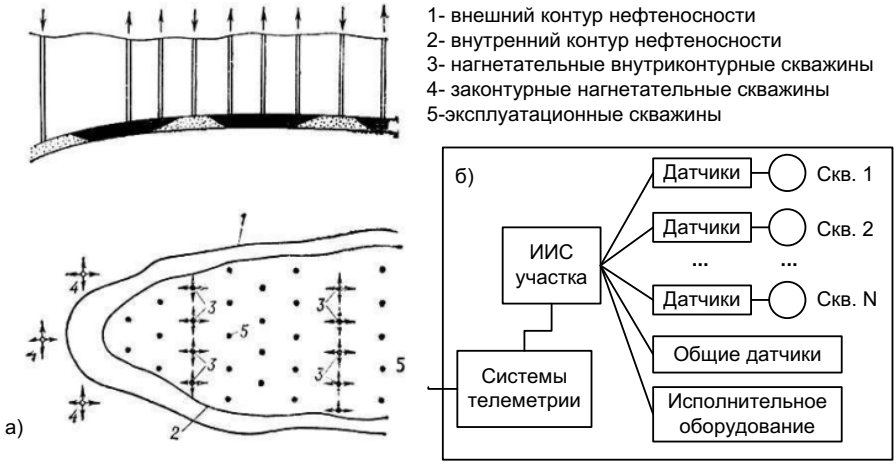

Рис. 1.3. Организация добычи нефти на отдельных участках
(а - организация добычи [2], б - организация локальной ИИС)

1.1.3. Информационные системы, используемые в отрасли

Добывающие предприятия нуждаются не только в учёте добываемых ресурсов, но и контроле их добычи и транспортировки в реальном времени. Типовыми задачами таких систем являются:

- мониторинг потоков нефти в реальном времени (оценка объёмов, качества и т.д.);
- регулировка объёмов добычи в зависимости от возможностей транспортной системы и потребностей;
- представление данных на диспетчерских пультах;
- сбор и сохранение истории параметров нефти.

Перечисленные задачи решаются информационно-измерительными системами (ИИС). Также в рамках предприятия может существовать обратная связь верхних уровней системы с добывающими комплексами (например, для передачи требований по объёму добычи). В этом случае система относится к классу информационно-управляющих систем (ИУС).

Для решения задач сбора данных и управления существует особый класс программного обеспечения, называемый SCADA-системами (Supervisory Control And Data Acquisition - Диспетчерское управление и сбор данных). Примерами подобных систем являются InTouch, Vijeo Citect, MasterSCADA, Sitex и многие другие [10].

SCADA-системы предоставляют интегрированную среду разработки (IDE) и инструментарий, с помощью которого пользователь может построить своё приложение, включающее как логику сбора и обработки данных, так и средства визуализации. В соответствии с [5], наиболее распространёнными возможностями SCADA-систем являются:

- анимированная визуализация системы на операторской панели;
- интерактивные элементы управления на операторской панели;
- подсистема построения графиков, позволяющая сохранять и по запросу воспроизводить историю изменения технологических параметров;
- подсистема событий, дающая возможность ведения и отображения протокола событий;
- контроль и обработка исключительных событий;

- программирование поведения системы при помощи специальных языков (наиболее популярны VBA, C#, C-подобные языки).

Таким образом, использование SCADA-систем позволяет значительно ускорить разработку верхнего уровня информационно-измерительных систем.

1.1.4. Интеграция информационных систем между собой

Как видно из предыдущего пункта, одновременно на предприятии используются две группы информационных систем. Одна решает экономические задачи, а вторая – информационно-измерительные. Данные группы систем должны взаимодействовать между собой.

Например, промышленная ИС должна передавать в экономические системы информацию об объёме добычи на различных кластерах, чтобы можно было обеспечить транспортировку нефти, хранение и продажу нефти. Статистика по добыче необходима для проведения анализа и планирования работы предприятия. С другой стороны, экономическая ИС должна передать в промышленную систему планы по добыче нефти. Таким образом, обе ИС должны взаимодействовать друг с другом в реальном времени, а также иметь доступ к истории работы. Пример организации совместной работы систем приведён на рис. 1.4.

В современном мире нефтедобывающие предприятия используют средства анализа потоков нефти от различных производителей. При этом несколько различных систем может эксплуатироваться одновременно, что может быть вызвано как ограниченными возможностями отдельных средств, так и стремлением предприятия диверсифицировать процесс добычи.

Производители средств анализа нефти предлагают решения по интеграции их в общую систему: модули сбора и первичной обработки данных, средства телеметрии и даже полноценные ИИС. Большинство заказчиков при этом стремятся использовать единую информационную систему, что позволяет

эффективно управлять предприятием. Поэтому, встаёт задача интеграции частных ИИС с внутренними информационными системами предприятия.

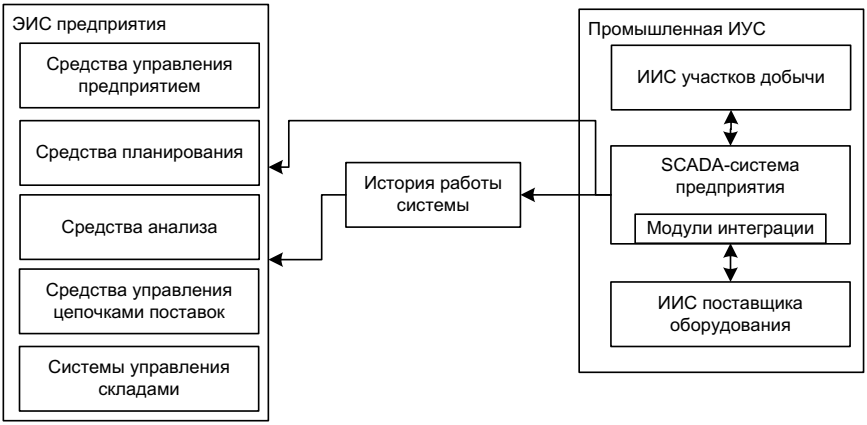

Рис. 1.4. Взаимодействие ЭИС и промышленной ИУС предприятия

Обычно подобная задача решается на уровне SCADA-системы предприятия. При добавлении устройства от нового поставщика в данную систему добавляется модуль, который обеспечивает управление ИИС поставщика и формирование данных, необходимых для функционирования внутренних систем.

1.2. Введение в проект “Канада”

1.2.1. Ограничения по разглашению информация

В рамках работы по проекту “Канада” автором был заключен ряд соглашений по неразглашению информации. В связи с этим, невозможно предоставить в дипломном проекте всю информацию по проекту и техническим особенностям его реализации.

В процессе написания написании диплома автор консультировался с руководством ООО “Комплекс-Ресурс”, и в итоге бы сформирован перечень тем, которые не могут быть разглашены в дипломном проекте или в процессе защиты. В соответствии с данным перечнем запрещается:

- разглашение имени компании-заказчика системы или любых технических характеристик используемых им систем (кроме согласованного списка);
- приведение проектной документации в качестве приложений;
- использование любых фотографий или скриншотов ИИС Neftemer или демонстрационного стенда;
- приведение промышленных конфигураций системы или элементов UML-спецификации на систему или её составляющие;
- описание внутреннего протокола взаимодействия модулей системы;
- приведение исходных кодов баз данных, математического и диагностического ПО системы, а также функционально-полных описаний остальных модулей.

Таким образом, в дипломном проекте невозможно подробно описать процесс разработки и реализации программного обеспечения. Данные элементы в дипломном проекте будут приведены урывочно, но подробно будут описаны используемые методологии и подходы к построению архитектур.

1.2.2. Краткая информация о фирме ООО “Комплекс-Ресурс”

История фирмы началась в конце 70-х годов, когда Кратировым В.А. был предложен новый метод анализа потока нефти с использованием радиоактивного гамма-излучения [13]. Основной особенностью предлагаемого метода была его неинвазивность, т.е. не требовалось встраивать датчики внутрь транспортной инфраструктуры, что сильно упрощало монтаж и обслуживание оборудования. Идея нашла поддержку в фирме Беларусь-Нефть, которая спонсировала дальнейшие исследования.

В конце 1988 года, была разработана первая версия анализатора нефти, названная “Пульсаром”. Она тестировалась в нескольких нефтедобывающих компаниях, но с развалом СССР были потеряны и заказчики. Тем не менее, в

1991 году Кратировым В.А. было основано ООО “Комплекс-ресурс”, которое занялось разработкой промышленного варианта анализатора на той же технологии.

В 1997-м году анализатор нефти был впервые сертифицирован для промышленного использования. В 2001-м году началось эксплуатационное тестирование датчиков в Усинске (Республика Коми). В настоящее время эксплуатация данных устройств расширена и одновременно используется около двухсот анализаторов. Также проводится эксплуатационное тестирование датчиков за рубежом (в Канаде, Бразилии и некоторых других странах). На рис. 1.5 приведён пример одного из основных измерительных модулей системы, называемый Блоком Детектирования (БД). Данные блоки осуществляют сбор первичных данных, которые впоследствии приводятся к показателям потока нефти.

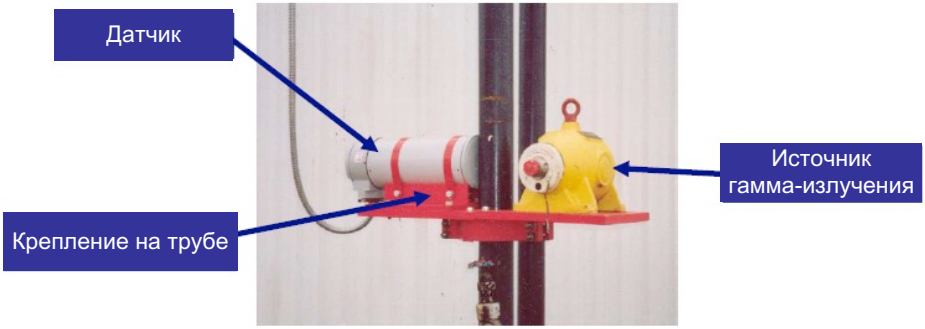

Рис. 1.5. Фотография анализатора нефти, производимого ООО “Комплекс-Ресурс

В начале 2005-го года в Крэнфилдском университете (Великобритания) было проведено тестирование измерителей ООО “Комплекс-Ресурс” на соответствие западным стандартам измерительного оборудования. Результаты тестирования оказались удовлетворительными для всех типов нефти, и, с учётом относительно невысокой себестоимости оборудования, стала очевидной перспектива применения измерителей на западном рынке.

Для продвижения анализаторов нефти в Великобритании ООО “Комплекс-Ресурс” совместно с De Flow Ltd и Крэнфилдским университетом было основано совместное предприятие, названное Neftemer Ltd. В данной фирме анализатор получил название “Neftemer Multiphase Meter”

В настоящее время фирма активно развивается. У неё имеются представительства во многих нефтедобывающих странах. На рис. 1.6 показано территориальное расположение представительств Neftemer Ltd [28].

Рис. 1.6. Карта расположения представительств ООО “Комплекс-Ресурс”

Фирма постоянно совершенствует свои датчики для улучшения точности показателей потока нефти и повышения надёжности устройства. Также ведётся разработка ПО для сбора данных и эталонов для поверки промышленных версий анализаторов.

ООО “Комплекс-Ресурс” предлагает различные решения для анализа нефти, предназначенные для работы в различных условиях: от полярных месторождений в Канаде и Сибири до подводной добычи. Блоки Neftemer имеют необходимую сертификацию и в настоящее время промышленно используются в России и Канаде. Кроме того, производится эксплуатационное тестирование блоков в нескольких странах.

В марте 2010-го года Neftemer Ltd и Крэнфилдский университет получили премию за “Лучшую инновацию” от EEEgr (East of England Energy Group) за разработку системы по определению и контролю нежелательных включений газа и жидкостей в потоке нефти [25]. Была подтверждена коммерческая эффективность системы, и следует ожидать её дальнейшего развития.

До участия в проекте “Канада”, которому посвящена данная работа, автор дипломной работы занимался разработкой ПО, предназначенного для сбора данных с блоков детектирования (БД) для ООО “Комплекс-Ресурс”.

1.2.3. Требования, поставленные потенциальным заказчиком

В 2006-м году ООО “Комплекс-Ресурс” заключило контракт с крупной международной нефтедобывающей компанией на поставку нескольких измерительных комплексов для тестовой эксплуатации. По итогам эксплуатации компания проявила интерес к дальнейшему использованию системы. К сожалению, существовавшие на тот момент версии информационной системы не соответствовали потребностям компании-заказчика.

По итогам был сформирован ряд требований к информационной системе, при выполнении которых та компания была согласна заказать несколько десятков измерительных комплексов. Точная сумма контракта не известна, но надо отметить, что аналогичные измерительные комплексы других производителей стоят десятки, а то и сотни тысяч долларов (например, [18] или [31]). Естественно, подобные возможности были крайне интересны для небольшой фирмы, и было заключено предварительное соглашение, по которому заказчик раскрыл часть информации о своих информационных системах и предоставил требования к информационной системе.

В соответствии с требованиями, ИИС должна собирать и обрабатывать информацию с множества удалённых точек добычи. При этом заказчик предоставляет транспортный канал для передачи данных, но задачи передачи данных через него ложатся на разработчика. Также система должна иметь свою

базу данных и предоставлять информацию о текущем состоянии системы в реальном времени. Также должны быть предоставлены терминальные программы для расширенного анализа потоков нефти, что должно демонстрировать преимущества анализаторов перед конкурентами. Ниже будут рассмотрены только общие требования к системе.

Территориальное расположение комплексов

Фирма-заказчик ведёт разработку месторождений во многих регионах мира с разными природными условиями. При этом применяются различные технологии добычи нефти, да и сам её состав сильно варьируется в зависимости от месторождения.

Заказчик хотел бы протестировать работу анализаторов нефти в различных условиях, поэтому планируется одновременно установить системы в нескольких точках земного шара. Были предъявлены следующие требования:

- анализаторы нефти располагаются в районах добычи группами до десяти единиц;
- в каждом районе добычи предоставляется помещение для размещения оборудования с обеспечением условий в индустриальном диапазоне (-40 .. +80 ^{0}C);
- в вычислительном центре фирмы предоставляется серверное оборудование для размещения баз данных системы и дополнительного программного обеспечения;
- обеспечивается доступ оборудования с кластеров во внутреннюю сеть компании с туннелированием через VPN (Virtual Private Network - виртуальная частная сеть);
- не гарантируется постоянное подключение кластеров к сети, и должна быть обеспечена возможность автономной работы.

Также был поставлен ряд требований, ограничивающих возможности доступа к системам.

- предоставляется удалённый доступ к серверу для проведения диагностики и контроля данных;
- взаимодействие с внешним миром из виртуальной локальной сети невозможно;
- физический доступ оборудованию на добывающих кластерах затруднён, обновление ПО и сервисное обслуживание должны по-возможности производиться удалённо.

Таким образом, заказчик требует построить распределённую информационную систему, используя для этого виртуальные сети. Примерная схема подобной системы приведена на рис. 1.7.

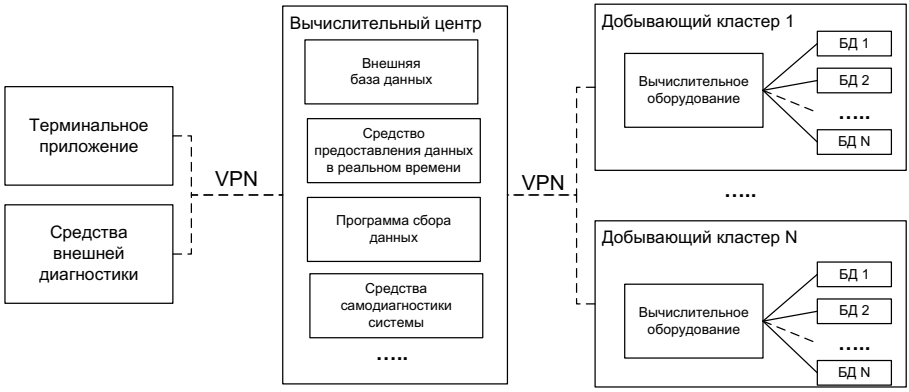

Рис. 1.7. Сетевая структура ИИС

Требования по предоставлению данных в реальном времени

Для извлечения данных реального о состоянии потока нефти заказчик предполагает использовать ту же систему, что и для всех видов анализаторов, используемых им в настоящее время. Для сбора данных используется одна из наиболее популярных SCADA-систем, которая поддерживает множество стандартизованных интерфейсов и протоколов доступа. Поэтому, при разработки заказчик требует лишь реализовать свой сервер данных на базе стандартных протоколов. Заказчиком заявлены следующие требования:

- поддержка интерфейсов Ethernet и RS-485 и протоколов Modbus TCP и Modbus RTU (соответственно);
- предоставление информации о скорости потока нефти, содержании воды и некоторых примесей с задержкой не более 30 секунд;
- наличие операторских панелей на пунктах добычи с отображением текущего состояния потока и возможностью управления;
- наличие клиентского приложения, которое может подключаться к серверу данных через сетевой интерфейс.

Требования по удалённому управлению

Несмотря на то, что ИИС предназначена прежде всего для сбора данных, в ней должны быть предоставлены и некоторые функции управления. В ней должны быть заложены следующие возможности:

- возможность запуска процедур самодиагностики;
- удаленное обновление ПО и конфигураций;
- возможность запуска синхронизации локальных данных со съёмными носителями;
- возможность удалённого перезапуска и аварийного отключения модулей системы.

Требования по хранению данных

Заказчик использует свои SCADA-системы и средства хранения данных, которые интегрированы в его бизнес-процессы. Так как одновременно используется множество различных типов анализаторов, предоставляющих разнородную информацию, то заказчик использует и сохраняет в своих системах только необходимую ему информацию.

В то же время, заказчику хотелось бы иметь всю полезную информацию, которые может предоставить блоки детектирования ООО “Комплекс-Ресурс”. Поэтому, в дополнение к имеющейся системе заказчик потребовал предоставить

и собственные средства хранения и апостериорной обработки данных. Были предъявлены следующие требования:

- использование реляционной базы данных как хранилища данных на базе существующих решений;
- предоставление заказчику информации о внутренней архитектуре базы данных и технической документации;
- наличие клиентского приложения для администрирования содержимого базы данных;
- возможность импорта и экспорта данных в файлы с открытым внутренним форматом;
- наличие пользовательского клиента с предоставлением истории, возможностями хранения заметок и генерации отчётов.

Общие выводы

В соответствии с требованиями можно заключить, что заказчику требуется информационная система, которая реализует все этапы обработки, транспортировка, хранения и представления данных. Несомненно, подобная система будет достаточно сложна, и её реализация с требуемым уровнем качества потребует больших трудозатрат.

В то же время, заказчиком предоставлена практически полная свобода по технической реализации системы и её отдельных модулей. Основным ограничением тут являются время и стоимость разработки, а также стоимость аппаратных комплектующих. Но надо понимать, что стоимость самих блоков детектирования, используемых в самой системе, и стоимость остальных модулей системы может достигать нескольких тысяч долларов, практически не влияя на общую стоимость владения, что даёт достаточно большой простор для разработки и позволяет сосредоточиться на качестве и надёжности системы.

1.2.4. Требования, поставленные руководством ООО “Комплекс-Ресурс”

Поставленные заказчиком требования во многом являются общими для отрасли. Следует ожидать, что любая информационная система будет обладать схожими характеристиками. Однако, в каждой компании бизнес-процессы и архитектура информационных систем обладают своими особенностями, поэтому в любом случае потребуется доработка системы. В связи с этим, руководством ООО “Комплекс-Ресурс” был сформирован ряд требований к системе, призванных повысить её гибкость и снизить затраты на её адаптацию к ИС заказчиков. Также был поставлен ряд бизнес-требований, связанных с экономическими ожиданиями и возможностями предприятия.

Технические требования к системе

Разработка должна быть направлена на то, чтобы минимизировать издержки на внедрение, поддержку и модификацию системы. Руководство фирмы поставило следующие требования:

- поддержка всей номенклатуры датчиков и анализаторов, поставляемых ООО “Комплекс-Ресурс”;
- возможность получения информации о состоянии всей системы в любой момент времени (опция);
- обеспечение гибкости системы программными средствами;
- обеспечение высокой надёжности системы;
- наличие средств диагностики и самодиагностики;
- иерархическая модульная организация системы;
- разделение вычислительной и транспортной составляющих;
- использование единого интерфейса для связи модулей между собой;
- ориентация на ОС семейств Windows NT и Windows CE.

Детальная проработка технических требований легла на автора дипломного проекта и группу разработки. Данные вопросы будут описаны во второй главе дипломного проекта.

Бизнес-требования к системе

Фирма “Комплекс-Ресурс - относительно небольшое предприятие. Поэтому, оно обладает ограниченными финансовыми ресурсами и не может позволить себе большие затраты на разработку до начала продаж. Таким образом, вначале должна появиться демонстрационная версия системы, которую можно будет представить заказчику и привлечь его к инвестициям в дальнейшую разработку.

Демонстрационная версия системы должна соответствовать требованиям заказчика, но при этом можно временно отложить затраты на разработку. Можно указать следующие факторы:

- нет нужды в выборе оптимальных аппаратных комплектующих, системного программного обеспечения (СПО);
- не требуется оптимизация программного обеспечения под программную платформу;
- не требуется сертификация информационной системы;
- разработку средств повышения надёжности можно отложить.

Высвободившиеся ресурсы руководство фирмы потребовало направить на более глубокую проработку пользовательских интерфейсов и внешнего вида системы, чтобы повысить привлекательность системы для пользователей.

1.2.5. Исследование исходного состояния информационной системы

К моменту выхода на западный рынок в компании был разработан ряд информационно-измерительных систем для различных отечественных заказчиков. Эксплуатация данных систем продолжается до сих пор [29], и их надёжность была неоднократно подтверждена. Пример архитектуры одной из

систем приведён на рис. 1.8 [29]. В тоже время, все разработанные системы были ориентированы на определённых заказчиков с невысокими требованиями к функциональности.

Фактически, старые системы представляют собой локализованные Информационно-измерительные комплексы, которые лишь предоставляют информацию о состоянии потока нефти в реальном времени. Данные системы подключаются к телеметрическому оборудованию пользователя, и все задачи по дальнейшим обработке и хранению информации ложатся на информационные системы заказчика.

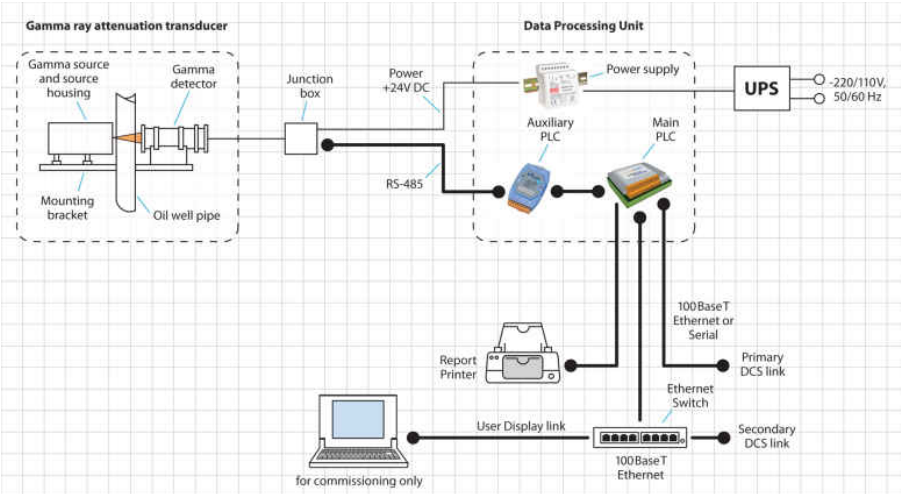

Рис. 1.8. Пример одной из исходных архитектур измерительной системы

Перечисленные недостатки не являются фатальными, и возможно расширение функциональности на базе существующей системы. Поэтому, было произведено исследование аппаратного и программного обеспечения, которое позволило оценить целесообразность использования существующих наработок.

Недостатки архитектуры

Архитектура старых систем предполагает использование минимального числа элементов системы, каждый из которых реализует определённые

функциональные требования. При этом, из-за недостатка вычислительных возможностей старых аппаратных модулей, используется каскадная обработка данных. Например, на рис. 1.8 обработка данных осуществляется в три этапа: в самом блоке детектирования (Gamma detector), основном и вспомогательном контроллерах (PLC). Очевидно, что перечисленные модули могут быть заменены одним с более высокой производительностью, но их объединение затруднено непереносимостью программного обеспечения и жёстко заданными интерфейсами, к которым подключаются внешние модули и диагностическое оборудование.

Говорить об архитектуре ПО в рамках существующих систем тяжело, так приложения для различных модулей разработаны разными исполнителями без использования типовых решений и методологий проектирования. В итоге, программное обеспечение системы плохо структурировано и повторное использование наработок по архитектуре не имеет смысла.

Недостатки аппаратной платформы

Основным недостатком аппаратной базы у ИИС прошлых версий является, прежде всего, ориентация на одного заказчика и его специфическую аппаратуру. Также нельзя забывать, что для сертификации измерительной системы в России требуется проходить целый ряд испытаний. Сам процесс сертификации может занимать дольше, чем разработка, поэтому замена или модификация аппаратных компонентов в большинстве случаев коммерчески невыгодна [14].

С этими проблемами столкнулись и продукты ООО “Комплекс-Ресурс”. Будучи разработанными в конце 90-х годов, аппаратная платформа системы практически не менялась, и к моменту исследования морально устарела. Например, блоки обработки данных, показанные на схеме рис. 1.8 позволяют обрабатывать данные только с нескольких блоков детектирования, т.к. в противном случае не хватает ни производительности самих модулей, ни

скорости передачи данных через промышленный интерфейс RS-485. Некоторые компоненты в настоящее время сняты с производства, и их дальнейшее использование требует дополнительных затрат и вносит существенные риски.

Можно перечислить и другие недостатки, но данные вопросы не являются предметом рассмотрения в дипломном проекте. По итогам исследования был сделан вывод, что аппаратная платформа неприменима в новых системах.

Недостатки программного обеспечения

Программное обеспечение ИИС условно можно разделить на следующие категории: приложения хранения данных, математическое, коммуникационное, управляющее и диагностическое ПО. Далее кратко рассмотрим все из них.

Исследование подтвердило, что точность измерений соответствует требованиям заказчика, т.е. математическое обеспечение, заложенное в систему, применимо для использования в последующих версиях. Поскольку сложность реализации математического аппарата велика, то данные модули, составляющие около половины программного кода исходной системы, имеет смысл сохранить в последующих версиях.

Для потенциального заказчика наиболее важны коммуникационные возможности системы. Существующая архитектура и реализация малопригодны для дальнейшего использования. Ниже перечислены основные недостатки:

- жёсткая привязка к сетевым интерфейсам;
- специфические протоколы доступа к данным;
- отсутствие средств централизованной обработки данных и хранения результатов обработки;
- невозможность получения данных о текущем состоянии системы из-за недостаточной скорости обработки данных;
- невозможность удалённого управления системой.

Управляющее программное обеспечение, использованное в системе, имеет ряд нареканий, связанных с корректностью обработки сложных ситуаций

и протоколированию работы системы. Тем не менее, оно может быть достаточно легко доработано. А вот диагностическое ПО в системах практически отсутствует. В ряде реализаций возможен вывод протоколов через специальные интерфейсы, но нет никаких возможностей самодиагностике или отладки, что критично для автономных систем.

Для сохранения результатов измерений системы используют текстовые файлы примитивны форматов. С учётом большого потока данных (каждый БД генерирует отчёты размером 1 Гбайт/сутки), использование подобного подхода в новой системе невозможно. Целесообразно использовать бинарные форматы или применять технологии сжатия данных для малозначимых данных

Кроме перечисленного, в программной реализации имеется ряд проблем, которые сильно затрудняют повторное использование исходных кодов. К таким недостаткам относятся:

- наличие большого числа ошибок;
- несмотря на использование C++, код
- непереносимость кода из-за явного использования системных вызовов и ассемблерных вставок;
- использование сторонних библиотек неизвестного происхождения без исходных кодов и информации о порядке лицензионного использования;
- отсутствие программной документации на исходные коды.

Перечисленные в предыдущем пункте недостатки говорят о невысоком качестве разработанных продуктов. Вероятно, причиной этому являлись невысокий уровень разработчиков и некорректная постановка процесса разработки на предприятии. Надо отметить, что разработка рассмотренного ПО велась сторонними разработчиками, а не сотрудниками ООО “Комплекс-Ресурс”.

1.2.6. Необходимость разработки новой информационной системы

По итогам исследования существующих ИИС был сделан вывод, что их доработка не является целесообразной, так как требуется модификация почти всех компонентов системы. В то же время, возможно использование некоторых её составляющих.

Корневым недостатком исходных систем признана их архитектура, которая позволяет успешно решать частные задачи, но непригодна для построения гибкой системы, что необходимо для распространения системы на западном рынке.

В итоге, принято решение вести разработку новой системы, которая бы соответствовала требованиям текущего заказчика и была бы максимально гибкой, чтобы её внедрение у прочих заказчиков требовало малых затрат. Для разработки данной системы в рамках ООО “Комплекс-Ресурс” был создан проект “Канада”, о котором будет подробно рассказано в следующем пункте.

1.3. Описание проекта “Канада”

История проекта

Работы по проекту “Канада” начались летом 2009-го года. Основной целью проекта является разработка информационно-измерительной системы (ИИС) для сбора данных с анализаторов нефти (производимых ООО “Комплекс-Ресурс”), расчёта параметров потока, хранение и предоставление результатов пользователю. Данная информационная система получила рабочее название “ИИС Neftemer”.

Проектируемая система позиционировалась как универсальная и совместимая с уже существующими информационными системами заказчиков. Поэтому, при разработке предъявлялись серьёзные требования по её конфигурируемости и переносимости. Кроме того, часть системы должна была работать автономно, и предъявлялись требования к её надёжности.

После анализа требований и возможностей руководством фирмы было решено производить разработку самостоятельно, создав для этого отдельную группу разработки в Санкт-Петербурге. Команда разработчиков была собрана в августе 2009-го года. В неё также был переведён автор данного отчёта, ранее работавший над другими проектами в ООО “Комплекс-Ресурс”. Изначально в разработке системы участвовало пять человек, но со временем состав группы менялся.

Автор дипломного проекта участвовал в нём до конца декабря 2010 года, после чего вышел из проекта по собственному желанию. В настоящее время проект продолжает развиваться, хотя команда разработчиков сменилась практически полностью, а некоторые ветки были заморожены из-за недостаточного бюджета.

Надо отметить, что на момент начала проекта присутствовала только абстрактная концепция системы, а часть требований и решений выбирались уже в процессе реализации проекта. Некоторые этапы данного проекта будут рассмотрены в следующих пунктах.

Классификация проекта

В таблице 1.1 приведена классификация проекта “Канада” в соответствии с рекомендациями Института Управления Проектами (PMI). Эта классификация является наиболее популярной в странах Америки, на которые ориентирован проект. Подробную информацию по классификации можно найти в [16]. Надо отметить, что является типовой при оценке сложности проектов в соответствии с методологией ICONIX, которая была использована при проектировании системы [17, с.215].

Следует учитывать, что до начала проекта “Канада” ООО “Комплекс-Ресурс” не занималась самостоятельной разработкой ПО на уровне информационных систем. Поэтому, очевидные из классификации недостатки

организации можно объяснить отсутствием опыта предприятия в разработке сложного программного обеспечения и недостаточным бюджетом.

Таблица 1.1

Классификация проекта “Канада”

Параметр	Значение	Комментарий
Тип	Технический	Проект предусматривает разработку технической системы
Класс	Мультипроект	Проект состоит из нескольких отдельных подпроектов, связь между которыми невелика[1]
Масштаб	Малый	Объём инвестиций менее 1 миллиона долларов
Вид	Смешанный	Проект носит как элементы проектной и сетевой организации
Длительность	Среднесрочный	Срок реализации проекта – около двух лет
Организационная форма	Проектная	Для реализации проекта создана отдельная группа, причём участники группы разработки не задействованы в других проектах фирм
Модель жизненного цикла	Спиральная	Интервал между внутренними релизами – 3 месяца
Стиль руководства	Делегирование полномочий	Большая часть задач по разработке и принятию решений переносится на группу разработки
Методы мотивации	Отсутствуют	
Организационная культура	Отсутствует	Разработка по большей части ведётся удалённо
Методологии разработки	Отсутствуют	Scrum, ICONIX

Участники проекта

Разработка проекта участвовало большое число юридических и физических лиц. Число непосредственных разработчиков было невелико, но в рамках проекта приходилось постоянно взаимодействовать с представителями

[1] Например, разработка встраиваемой части системы практически не связана с графическими интерфейсами пользователя.

заказчика и представителями других групп ООО “Комплекс-Ресурс”. В таблице 1.2 приведён список основных участников проекта.

Таблица 1.2

Участники проекта “Канада”

Группа	Участники	Комментарий
Инициатор	ООО Комплекс-Ресурс	
Заказчик	Не разглашается	В соответствии с соглашением запрещено любое упоминание заказчика
Куратор проекта	Кратиров Д.В.	Генеральный директор ООО “Комплекс-Ресурс”
Руководитель проекта	Не разглашается	В соответствии с личным пожеланием
Исполнители	Группа разработки в СПб	Состояла из 5 человек (в том числе в неё входил автор данного документа)
Поставщики	ООО “Комплекс-Ресурс”	Анализаторы нефти; отладочное оборудование; предоставление офиса для проведения встреч;
	sourcerepo.com	Хостинг для системы управления проектами
	Поставщики оборудования	Информация не подлежит разглашению
	Microsoft Corporation	ОС Windows CE, среды разработки
	Cranfield University	Системное тестирование ИИС на соответствие стандартам отрасли

Особенностью проекта “Канада” является территориальная удалённость его участников (см. рис. 1.9). Особенно важно то, что группа разработчиков находилась вне основного офиса фирмы, что затрудняет постоянное взаимодействие между участниками. С учётом этого большую часть обмена информацией приходится совершать через Интернет.

Большую часть времени группа разработки работало удалённо, поэтому организация совместной разработки была одним из важнейших вопросов на подготовительном этапе разработки. В ООО “Комплекс-Ресурс” необходимые информационные системы отсутствовали, и задачи по их внедрению пришлось

решать группе разработки. Подробная информация по данному вопросу приведена в разделе 3 дипломного проекта.

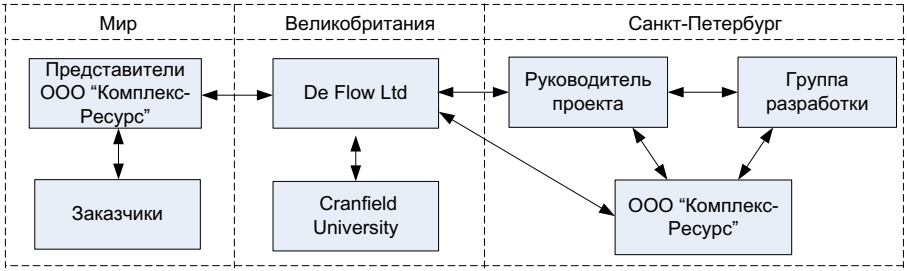

Рис. 1.9. Территориальное распределение участников проекта

Этапы развития проекта

Проект “Канада” был разбит на несколько этапов. На каждом из них расширялась архитектура и функциональность системы. Работа велась параллельно сразу по нескольким направлениям. В таблице 1.3 приведён список этапов разработки и краткая информация о задачах, которые выполнялись на каждом из этапов на каждом из этапов.

По таблице видно, что развитие системы идёт эволюционным путём. При этом на этапах производился частичный реинжиниринг ранее разработанных программных модулей и изменение архитектуры самой системы. Внесение изменений, в первую очередь, изменением требований к системе со стороны заказчика.

Модификация требований к системе входит в задачи управления требованиями, несмотря на возникающие неудобства, является составной и необходимой частью разработки сложных систем. Данный процесс предусмотрен спиральной моделью проектирования, которая была выбрана в рамках проекта.

Таблица 1.3

Этапы разработки в рамках проекта “Канада”

№	Сроки	Цель этапа	Основные задачи
1	август 2009-октябрь 2009	Сбор требований к системе и реализация тестовой системы сбора данных	- Анализ требований и разработка общей архитектуры информационной системы; - Внедрение систем организации совместной разработки на предприятии; - Разработка прототипов программы сбора данных и операторской панели для системы; - Выбор аппаратной платформы и СПО для реализации промышленного варианта системы; - Разработка ТЗ на основные модули и их согласование с заказчиком;
2	октябрь 2009-апрель 2010	Разработка демонстрацион-ной версии системы	- Разработка архитектуры модулей ИИС; - Разработка основных модулей для встраиваемых частей системы; - Разработка баз данных для коммуникационного контроллера и корпоративного сервера; - Разработка конвертеров данных для перевода данных в универсальные форматы; - Разработка демонстрационного варианта терминального приложения Neftemer Toolbox;
3	апрель 2010 - август 2010	Разработка основных модулей ИИС	- Разработка и согласование ТЗ на дополнительные модули системы и внесение изменений; - Реинжиниринг ранее разработанных модулей; - Разработка модулей управления конфигурациями и обновления программного обеспечения; - Разработка средств самодиагностики системы;
4	август 2010 - февраль 2011	Тестирование и доводка существующей реализации системы	- Поддержка существующей реализации встраиваемой части системы; - Разработка программной и сопроводительной документации; - Доработка средств работы с данными в составе комплекса терминальной программы; - Сдача системы заказчику для эксплуатационного тестирования.
5	будущее	Расширение функционал ьности	- Разработка дополнительных модулей системы по требованиям заказчика; - Внедрение ИИС у заказчика;

1.4. Задачи, поставленные перед автором дипломного проекта

Как уже было сказано, у автора данной работы уже имелся некоторый опыт работы с блоками детектирования ООО “Комплекс-Ресурс”. Поэтому, в самом же начале проекта мне были поручены достаточно важные задачи.Список основных задач, которые поручались мне при выполнении проекта, приведён в таблице 1.4.

Таблица 1.4

Задачи, поставленные перед автором дипломного проекта

Этап	Задачи Ненашева О.В.
1	- участие в подборе команды разработчиков для проекта; - внедрение системы управления проектом; - разработка архитектуры ИИС Neftemer - прототипирование измерительных приложений комплекса;
2	- разработка технических заданий и спецификаций на модули системы; - разработка программных модулей для встраиваемых частей системы; - выбор комплектующих для аппаратной части системы; - участие в разработке базы данных;
3	- разработка конечной архитектуры ИИС; - разработка внутреннего протокола для обмена данными между программными модулями системы; - разработка ТЗ и спецификаций на модули системы; - доработка ранее разработанных модулей под новую архитектуру; - программная реализация встраиваемых модулей системы;
4	- поддержка разработанных программных модулей; устранение ошибок; - разработка программной документации;
5	- консультирование участников проекта;

В среднем на протяжении проекта объём нагрузки составлял около 30 рабочих часов в неделю. На четвёртом этапе автором была завершена разработка порученных ему элементов системы, после чего он принял решение о выходе из проекта с продолжения научно-исследовательской работы в СПБГПУ.

На пятом этапе автором осуществлялось (и осуществляется) только консультирование руководителя проекта, так как дальнейшая разработка системы в настоящее заморожена, а заказчикам поставляется ранее выпущенный в июне 2011 года релиз системы.

2. РАЗРАБОТКА ОБЩЕЙ АРХИТЕКТУРЫ СИСТЕМЫ

В данной главе кратко рассмотрены подготовительные этапы разработки концепции системы. Исходными данными являлись требования, поставленные заказчиком и руководством ООО “Комплекс-Ресурс” (см. п. 1.2.3 и 1.2.4). Был произведён анализ данных требований, по результатам которого были разработаны детальные требования и общая архитектура системы.

После формирования общей концепции системы была произведена оценка сложности проекта, что позволило определить необходимые ресурсы и сформировать бюджет проекта. На данном этапе автор дипломного проекта выполнял роль технического консультанта, так как окончательное принятие решений осуществлялось руководителем.

2.1. Терминология, используемая при разработке системы

В пункте 1.2.3 достаточно чётко описаны требования заказчика к информационно-измерительной системе. На их основании можно разделить систему на четыре основные составляющие:

- блоки детектирования и подключенные к ним модули первичной обработки данных;
- модули сбора и обработки данных, расположенные в районе добычи;
- центральная система сбора и обработки данных (корпоративный сервер);
- терминальные системы.

В дальнейшем будем именовать программное обеспечение данных уровней информационно-измерительным (ИИ), коммуникационным (К), серверным (С) и терминальным (Д). Кроме того, введём понятия системного и диагностического ПО.

К системному ПО будут относиться приложения, обеспечивающие работу системы, но не реализующие бизнес-логику системы. Примерами подобного ПО являются операционная система, загрузчики системы, средства резервного копирования данных, средства обновления программного обеспечения.

Диагностическое ПО предназначено для контроля работы остальных модулей: выявления сбоев и ошибок, контроль целостности данных и сетевых соединений между модулями.

Естественно, в каждой из составляющих системы может присутствовать программное обеспечение любого типа, но для нас прежде всего важно распределение ролей между программными модулями.

2.2. Анализ поставленных требований

Анализ поставленных требований является первым этапом любой разработки. В рамках проекта “Канада” на данный этап было отведено около полутора месяцев. За этот срок были разработаны и согласованы технические требования к системе, на базе которых была разработана общая архитектура.

В работе нет возможности привести подробный анализ всех требований, так как данные информация во многом не подлежит разглашению, а общий размер документации составляет около двух сотен страниц. Поэтому, в пункте приведены основополагающие этапы анализа.

Уровни обработки данных

Таким образом, имеется иерархия из трёх уровней обработки данных, и требуется определить, на каких уровнях будет вестись обработка. Надо отметить, что анализаторы нефти генерируют достаточно большой объём сырых данных (около 8 Кбайт/секунду), а сложность алгоритмов обработки очень велика, так как для получения точных результатов требуется собирать статистику за часы работы. Поэтому, целесообразно вести обработки на нижних уровнях, передавая только результирующие показатели.

К сожалению, это противоречит требованию руководства фирмы о том, что вся информация о системе при соответствующих настройках должна попадать во внутреннюю базу данных, и придётся предусмотреть возможность передачи полной информации.

Тем не менее, всю обработку, которая даёт значимые показатели по потоку нефти и её составу, необходимо вести на уровне добывающего кластера, так как в требованиях не гарантируется устойчивое подключение к корпоративному серверу, а данные необходимо представлять и на кластерах добычи.

Взаимодействие между программными модулями

В соответствии с требованиями, требуется обеспечить гибкость системы. Наиболее важно при этом обеспечить независимость программной и аппаратной частей. Для этого требуется обеспечить следующее:

- программное обеспечение должно быть переносимо между аппаратными реализациями;
- программное обеспечение должно иметь модульную архитектуру;
- модули должны связываться друг с другом не только внутри одного устройства, но и будучи расположенными на различных устройствах.
- должна быть возможность запуска нескольких одинаковых модулей с независимыми конфигурациями на одном устройстве.

Выполнение подобных требований позволяет с лёгкостью масштабировать систему на разработку.

Средства диагностики

Так как система задумывается как высоконадёжная, то должна быть возможность контроля всех процессов, происходящих в системе. Прежде всего, это означает, что в каждом аппаратном устройстве должны быть реализованы средства диагностики, а каждый программный модуль должен иметь диагностический интерфейс.

Диагностические модули внутри системы должны образовывать дополнительную иерархию, чтобы обеспечить распространение ошибок и

информации об исключительных событиях. Также это даёт возможность централизованного контроля ошибок.

В рамках системы решено ввести дополнительный модуль, который будет обеспечивать верхний уровень контроля. В работе подобный модуль называется “супервизором”. Он будет запускаться на каждом устройстве системы и контролировать поведение всех программных модулей устройства, а также супервизоров устройств нижнего уровня иерархии.

Постановка требований к внешней базе данных системы

В соответствии с требованиями из п. 1.2.3, ИИС должна хранить историю своей работы в реляционной базе данных. Учитывая то, что в перспективе заказчик может использовать данную базу для своих целей (например, для расширенного анализа), желательно использовать типовые решения.

Наиболее распространённым подходом к реализации баз данных является использование сторонней системы управления базами данных (СУБД), которая реализует функциональность работы с реляционным хранилищем, включает средства администрирования и интерфейс взаимодействия с СУБД.

Большинство СУБД обладают сетевым интерфейсом доступа, что позволяет взаимодействовать с ними удалённо. Для формирования и извлечения данных используются специальные языки управления запросами. В настоящее время наиболее популярным языком управления запросами к базе данных является SQL (Structured Query Language – “язык структурированных запросов”). Он поддерживается практически всеми реляционными СУБД, и при его использовании можно обеспечить высокую переносимость структуры базы между различными системами. Это важно для демонстрационной версии системы, когда не принято окончательное решение по реализации базы данных.

По итогам работы к СУБД демонстрационной версии ИИС Neftemer были сформированы следующие требования:

- Должны существовать драйвера JDBC и ODBC для данной СУБД;

- СУБД должна выполняться в режиме сервиса;
- Должен присутствовать надежный механизм ограничения доступа к базе данных (управление правами доступа);
- СУБД должна эффективно работать с большими объемами данных;
- Необходима поддержка хранимых процедур и представлений для реализации стандартных алгоритмов обработки хранимых данных;
- Необходима поддержка триггеров для расширения функциональности базы данных (например, автоматическая обработка поступающих данных);
- Для развития системы до многопользовательского варианта необходима поддержка защиты данных от одновременного доступа.

2.3. Разработка общих элементов архитектуры ИИС Neftemer

Выбор внутреннего протокола системы и сетевого интерфейса

По результатам анализа из предыдущего пункта, требуется выбрать интерфейс или набор интерфейсов, через которые элементы системы будут взаимодействовать между собой. Учитывая сложность системы и аппаратную независимость, нежелательно использовать фиксированный интерфейс.

По итогам анализа было принято определить протокол верхнего уровня, который возможно будет реализовать на базе произвольного физического. За основу решено взять стек протоколов TCP/IP, реализации которого существуют практически для всех технических интерфейсов (например, Ethernet, RS-485, I2C). Использование TCP/IP как несущего протокола является типовым решением и рекомендовано в соответствии со многими стандартами и методологиями (например, [22]).

Протокол TCP гарантирует передачу данных в сети, реализует квитирование при передаче данных и контроль целостности получаемых данных [15]. Данный протокол является одним из наиболее распространённых в современном мире, и на его основе построено множество прикладных

протоколов, используемых в сети Интернет (например, HTTP, FTP, SMTP, RDP и другие).

Тем не менее, одного лишь TCP недостаточно для построении системы. Требуется обеспечить передачу команд и данных между различными модулями. Для этого также существует множество решений, одним из которых является использование расширяемых протоколов Modbus TCP или Modbus RTU. Использование данных протоколов для предоставления данных в реальном времени является одним из требований заказчика, поэтому целесообразно их использовать во всей системе.

Modbus является семейством открытых протоколов взаимодействия в технических системах, и в настоящее время де-факто является стандартом для SCADA-систем и интеллектуальных датчиков. Например, датчики ООО “Комплекс-Ресурс” частично реализуют протокол Modbus RTU. Семейство протоколов Modbus предоставляет:

- шинную архитектуру взаимодействия (1 клиент – много серверов);
- набор команд для регистрового доступа к памяти серверов;
- набор диагностических команд;
- набор зарезервированных пользовательских операций, на базе которых может быть реализована дополнительная функциональность.

Более подробное описание протокола Modbus можно найти в документации (например, [24]). В рамках разработки потребуется реализовать команды доступа к данным, команды управления и диагностические приложения.

По истокам обсуждений решено открыть заказчику протокол взаимодействия со всеми модулями системы, а не только с сервером данных реального времени. Такое решение позволит упростить использование системы заказчиком и, при необходимости, разработку собственных приложений, использующих элементы системы.

Подход к развёртыванию системы и механизмы конфигурирования

Для системы с переменной архитектурой важно обеспечить возможность гибкой настройки системы. Типовым решением является использование единого описания всей системы, на базе которого производится автоматическое или автоматизированное развёртывание её модулей. В терминах системы подобное описание называется конфигурационным файлом системы. Несмотря на сложность реализации подобного механизма, он позволяет экономить время на внедрении ИИС Neftemer.

Предлагаемый подход, кроме требований к гибкости системы, также и учитывает и необходимость обновления программного обеспечения, при котором должна быть возможность замены не только исполняемых, но и конфигурационных файлов системы. Итак, система управления конфигурациями решает состоит из следующих программных модулей:

- модуль обновления ПО;
- модуль запуска программного обеспечения;
- средства контроля корректного запуска.

Основной опасностью при обновлении системы является её возможный отказ в случае наличия ошибок в конфигурационном файле. Например, изменение настроек внешнего интерфейса в аппаратном модуле может привести к тому, что устройство перестанет откликаться на внешние запросы. Это приведёт к тому, что удалённое исправление системы будет невозможным, что может быть фатально из-за ограниченной доступности модулей на добывающем кластере. Исправление подобных ошибок имеет огромную стоимость из-за нарушения работы добывающего комплекса и необходимости отправки специалистов на место аварии.

Для исключения подобных ситуаций следует предусмотреть механизм автоматической отмены внесённых изменений в случае отсутствия связи с внешними уровнями иерархии. Для приятия подобных решений необходим

анализ состояния всех программных модулей, и для этих целей решено использовать супервизор. Пример подобной архитектуры приведён на рис. 2.1. Далее рассмотрим некоторые из модулей, участвующих в механизме развёртывания ИИС Neftemer.

Для развёртывания системы по описанию из конфигурационного файла решено использовать специальное системное приложение, называемое в работе “Загрузчик ПО”. Оно должно реализовывать следующую функциональность:

- считывание конфигурации из специальных файлов;
- контроль настроек и выявление ошибок;
- запуск программных модулей;
- передача настроек в программные модули;
- контроль корректного запуска модулей.

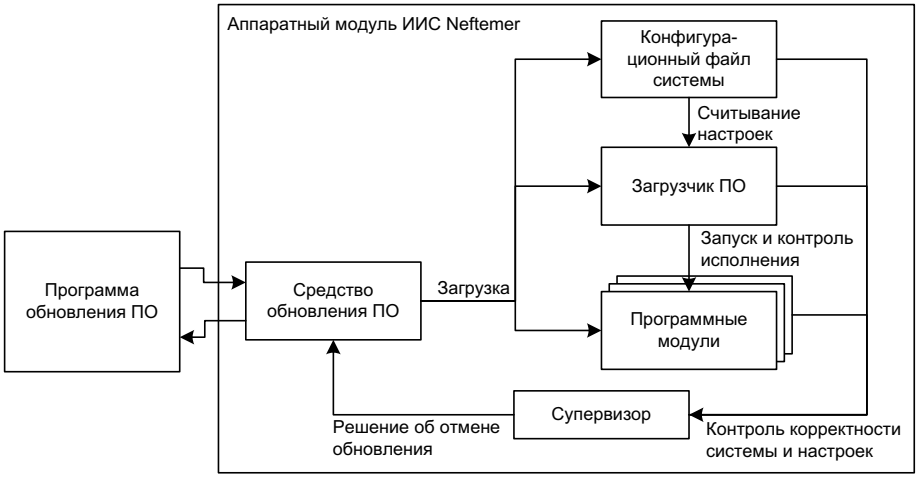

Рис. 2.1. Управление развёртыванием системы

На надёжность ИИС также влияет и системное программное обеспечение, используемое на различных программных модулях. С учётом независимой работы модулей и необходимости использование кода на промышленных и универсальных ЭВМ (для серверных решений и демонстрационной системы), целесообразно вести использовать программные решения на базе операционных

систем. Это позволяет использовать встроенные в ОС средства повышения надёжности, но при этом потребуется обеспечить переносимость программного обеспечения между операционными системами.

2.4. Разработка архитектуры системы

Распределение функциональности между аппаратными модулями

При разработке демонстрационной версии системы решено вести разработку в рамках сетевой архитектуры, которая была затребована заказчиком (см. рис. 1.7). Решено использовать единый сервер для всех ИС Neftemer на предприятии. На нм будут размещены приложения для хранения данных и их представления в реальном времени. Система будет взаимодействовать с терминальными ПК и измерительной частью системы.

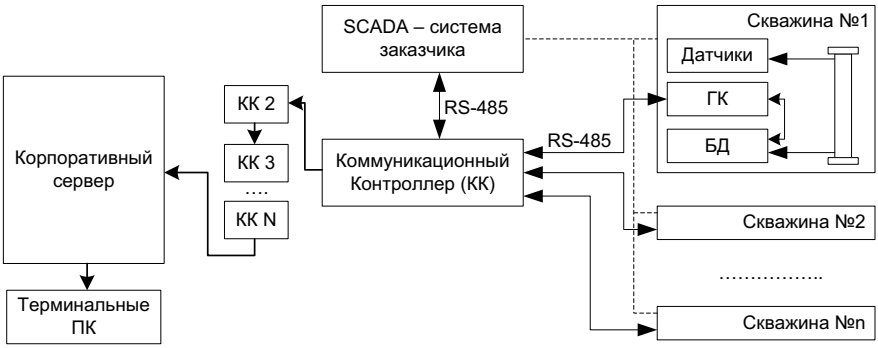

Рис. 2.2. Схема соединения устройств в системе

На каждом добывающем участке предполагается разместить так называемый Коммуникационный Контроллер (КК), который должен исполнять функции ИИС участка. Всё взаимодействие между верхним уровнем управления и БД будет осуществляться через данный модуль. Также он будет выполнять задачи по локальному хранению и представлению данных. разработанная архитектура предполагает, что на КК системы можно выстроить в иерархию, которая обеспечит дополнительные уровни передачи данных и контроля (например, на уровне группы участков в рамках одного месторождения).

К каждому БД в системе решено добавить дополнительный модуль, который на рис. 2.2 называется Главным Контроллером (ГК). Целью данного модуля является первичная обработка информации, что позволяет снизить трафик между КК и БД в десять раз. Также модуль реализует мониторинг работы БД и его средства самодиагностики. Модуль ГК предполагается разместить внутри блоков детектирования, т.е. не возникнет проблем с их физическим размещением на участке добычи.

Перечень основных модулей информационной системы

По результатам проведённого в предыдущем пункте анализа был сформирован перечень программных модулей, которые должны быть реализованы в демонстрационной версии системы. В таблицах 2.1-2.4 приведена краткая информация о каждом из приложений. Программы условно разбиты на четыре категории:

- встраиваемые приложения ИС Neftemer;
- тестовые и диагностические приложения комплекса;
- средства коммуникации;
- средства визуализации и пользовательских интерфейсов.

В класс встраиваемых приложений входят все программы, которые включаются в цепочку обработки, хранения и передачи данных. Также в неё входят средства повышения надёжности системы. Кроме перечисленных в таблице 2.1 программ, в систему также входят локальные базы данных и средства операционной системы, о которых будет рассказано позднее.

Диагностические приложения системы не предполагают использования в рамках промышленных версий ИС Neftemer, но необходимы в процессе разработки, так как разработчики не имеют постоянного доступа к отладочным стендам системы.

Таблица 2.1

Встраиваемые приложения ИИС Neftemer

Название программы	Основные функции
Загрузчик ПО	- запуск программного обеспечения по заданной конфигурации системы.
Супервизор	- контроль остальных программ системы; - мониторинг ресурсов системы.
Сервер логов	- сбор логов со всех программ системы
Средство обновления ПО	- загрузка файлов и конфигураций в системы; - замена ПО с контролем версий и целостности; - хранение старых конфигураций.
Сервер ГК	- сбор сырых данных с МКБД; - обработка сырых данных.
Сервер КК	- сбор данных и результатов с КК; - получение температур, давления и др. с датчиков
Менеджер архивов	- сбор данных из системы и их архивация; - предоставление информации об архивах; - синхронизация данных с носителями.
Конвертер данных	- Преобразование данных в системе; - диспетчеризация преобразования данных.
Программа экспорта в БД	- сбор и экспорт данных в базу данных системы в реальном времени; - мониторинг использования ресурсов СУБД.

Таблица 2.2

Коммуникационные приложения ИИС Neftemer

Название программы	Основные функции
Входной шлюз RS-485	- приём и маршрутизация запросов по интерфейсу RS-485 в режиме ведомого устройства; - управляемая блокировка команд и адресов (для SCADA)
Выходной шлюз RS-485	- туннелирование внутреннего сетевого протокола через Modbus и передача по интерфейсу RS-485; - разделяемый доступ нескольких клиентов к одному физическому порт (управление приоритетами и т.п.)
Входной шлюз TCP/IP	- маршрутизация адресов; - контроль доступа; - аутентификация, шифрование внутреннего протокола;

Таблица 2.3

Тестовые и диагностические приложения ИИС Neftemer

Название программы	Основные функции
Частотомер	- сбор данных с МКБД; - консольный интерфейс для мониторинга МКБД; - предоставление данных через сервер; - формирование тестовых файлов с сырыми данными.
Эмулятор ГК	- эмуляция протокола и режимов работы ГК; - загрузка сырых данных из текстовых файлов; - загрузка результатов из файлов или их генерация.
Эмуляторы БД	Данные приложения эмулируют поведение блоков детектирования различных версий: - эмуляция протокола и режимов работы МКБД; - загрузка данных из текстовых файлов; - протоколирование приходящих запросов.
Эмулятор датчиков	- формирование тестовых показаний по давлению и температуре.
Монитор логов	- подключение к любому серверу ИИС Neftemer; - считывание логов с сервера; - вывод логов в тестовом виде и их сохранение.
Монитор состояния	- подключение к супервизорам аппаратных модулей; - считывание, визуализация и сохранение состояния модулей; - перехват сообщений о критических ошибках в системе;

Таблица 2.4

Программы визуализации ИИС Neftemer

Название программы	Описание
Neftemer Toolbox	- Предоставление средств работы с базой данных и ИИС Neftemer в едином графическом интерфейсе; - Визуализация архивов, хранящихся в базе данных системы; - Визуализация текущего состояния системы и отображение данных в реальном времени; - Управление конфигурациями системы и обновлением ПО;
Графический интерфейс КК	- Основной визуализатор коммуникационного контроллера, рассчитанный на VGA-дисплей
Встраиваемый интерфейс КК	- Визуализатор для жидкокристаллического индикатора; - Индикация состояния программы на индикаторах.

Структура ПО демонстрационной версии системы

В соответствии с разработанной архитектурой, для каждого аппаратного модуля системы была сформирована архитектура программного обеспечения системы. По соглашению о неразглашении информации, в дипломном проекте нельзя приводить информацию о системном программном обеспечении системы, поэтому на схемах приведены только модули, входящие в комплекс разработанного программного обеспечения. На рис. 2.3 схема ПО, используемая в главном контроллере ИС Neftemer.

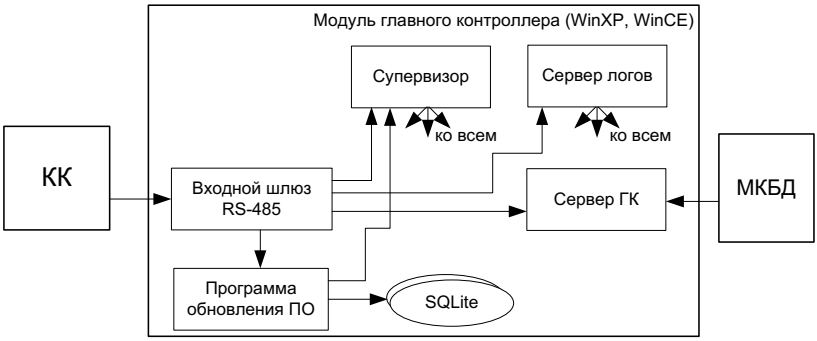

Рис. 2.3. Схема демонстрационной конфигурации главного контроллера

Коммуникационный контроллер может иметь несколько конфигураций. На рис. 2.4 приведена наиболее полная конфигурация системы, когда КК осуществляет внутреннюю обработку данных и их сохранение в локальной базе данных. Подобный подход используется в случае автономной работы системы без прямого подключения к корпоративному серверу. Передача данных возможно через архивы, генерируемые конвертером данных.

Архитектура отладочной конфигурации

Для отладки системы во время разработки программного обеспечения неудобно работать с множеством аппаратных устройств. Поэтому, после отладки интерфейсов взаимодействия целесообразно перейти к конфигурации, когда весь комплекс приложений развёртывается на ПК разработчика. Схема подобной конфигурации приведена на рис. 2.5.

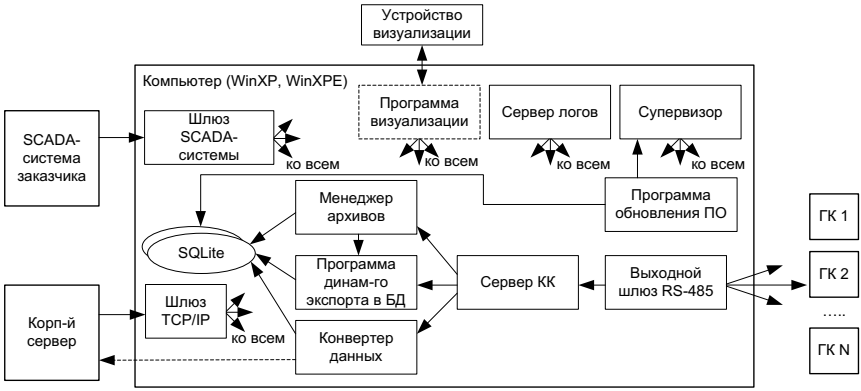

Рис. 2.4. Схема демонстрационной конфигурации коммуникационного контроллера ИС Neftemer

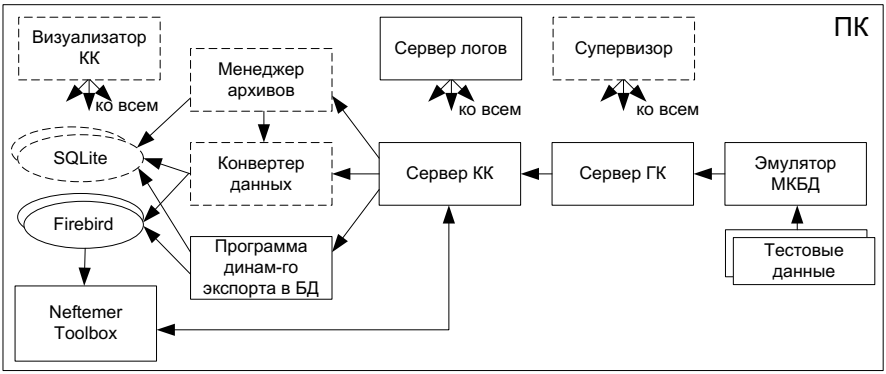

Рис. 2.5. Отладочная конфигурация ИИС Neftemer

В отладочной конфигурации Neftemer Toolbox одновременно исполняет функции SCADA-системы и клиента базы данных потенциального заказчика ИИС Neftemer. При этом используются интерфейсы и алгоритмы работы, подобные приложения заказчика.

2.5. Подготовка к выполнению проекта

При разработке ИИС Neftemer требуется в максимально короткий срок представить демонстрационную версию системы, которая соответствует требованиям заказчика и демонстрирует основные преимущества системы. Руководством ООО “Комплекс-Ресурс”.

Оценка сложности проекта

В пунктах 2.2-2.4 была сформирована общая концепция всей системы. Проектируемая система достаточно сложна с технической точки зрения. Можно выделить следующие причины:

- распределённость системы;
- многомодульность;
- платформенная независимость программных модулей (может использоваться различный аппаратный базис, операционная система также может меняться);
- средства повышения надёжности и самодиагностики.

Дополнительную сложность вносит необходимость обеспечения гибкости системы в течение всего процесса разработки. С учётом разработанной архитектуры, время разработки всей системы одним человеком было оценено в несколько лет. Львиная доля сложности состоит в разработке программного обеспечения, поэтому для соблюдения разумных сроков потребуется несколько разработчиков.

Формирование ресурсов проекта

Как видно из предыдущего пункта, разрабатываемая информационная система достаточно сложна, и её разработка исходными средствами ООО “Комплекс-Ресурс” оказалось невозможной. Руководством фирмы было принято решение сформировать отдельную команду разработчиков, в которую был переведён автор дипломного проекта, так как на тот момент у него имелся опыт разработки терминального ПО для блоков детектирования.

Информация о бюджете проекта разглашению не подлежит. Надо отметить, что на первом этапе проект реализуется на средства ООО “Комплекс-Ресурс” без участи заказчика, поэтому проект был сильно ограничен в средствах. Для экономии средств было решено отказаться от аренды помещения для команды разработчиков и вести разработку удалённо. Для проведения встреч

ООО “Комплекс-Ресурс” согласилось предоставлять помещение в своём центральном офисе.

После формирования команды разработки был сформирован фонд, в котором хранились средства на закупку необходимого программного и аппаратного обеспечения. Данный фонд контролировался руководителем проекта, а формирование списка необходимых средств было поручено команде разработки. Надо отметить, что в рамках разработки использовалось только лицензионное программное обеспечение, средства на закупку которого были выделены из фонда.

Формирование команды разработки

В начале проекта команда разработки включала руководителя проекта, автора дипломного проекта и ещё одного представителя ООО “Комплекс-Ресурс”, который должен был заниматься аппаратной частью системы. Прежде всего, в команде не хватает разработчиков программного обеспечения. В соответствии с [6], команда разработки программного обеспечения должна включать следующие роли:

- менеджер программного продукта;
- менеджер команды;
- системный архитектор;
- инженер-программист;
- инженер по тестированию;
- специалист по удобству пользования;
- технический писатель.

Бюджет проекта не позволяет нанять полноценную команду на всё время выполнения проекта, поэтому от некоторых ролей пришлось отказаться, а другие - совместить. Руководством ООО “Комплекс-Ресурс” было принято решение, что функции двух менеджеров будет совмещать руководитель проекта, а тестированием программных модулей будут заниматься разработчики. Это

противоречит большинству методологий командной разработки (например, Scrum [4] и Microsoft Solution Framework(MSF) [3]). Тем не менее, это – распространённая практика для малых компаний.

От специалиста по удобству пользования решено было отказаться, задачи системного архитектора и технического писателя решено передать автору дипломного проекта (за исключением Neftemer Toolbox). Таким образом, потребовалось набрать разработчиков, которые при этом были бы компетентны в смежных задачах.

Бюджет проекта позволял нанять двух-трёх полноценных разработчиков, что руководству казалось неэффективным из-за возможных “простоев” на этапе тестирования и приёмки. Поэтому, было принято решение временно нанимать людей для решения частных задач. К таким задачам были отнесены:

- разработка комплекса приложений Neftemer Toolbox;
- исследование систем организации совместной работы и обеспечения общего офисного пространства;
- реализация базы данных для демонстрационной системы;
- прототипирование графического интерфейса КК с использованием SCADA-системы;
- разработка пользовательских интерфейсов для встраиваемых частей системы.

Перечисленные задачи не критичны по надёжности и производительности, поэтому их можно поручать разработчикам с малым опытом работы или в качестве дополнительной работы. В итоге, в решении данных задач в команду было взято пять студентов СПБГПУ.

Сроки выполнения проекта

С учётом ресурсов предприятия был сформирован предварительный план работ по ИИС Neftemer. Проект был разбит на несколько этапов, состав и предполагаемые сроки исполнения которых приведены в таблице 2.5.

Таблица 2.5.

Предполагаемые сроки исполнения проекта “Канада”

Этап	Срок исполнения	Содержание
1	3 мес.	- Проектирование системы - Прототипирование программных модулей
2	9 мес.	Разработка демонстрационной версии системы: - Разработка аппаратной составляющей системы; - Разработка полнофункциональных версий основного встраиваемого ПО системы; - Разработка демонстрационных версий прочих модулей. - Сдача системы заказчику для тестирования.
3	6 мес.	- Доработка программных модулей системы; - Разработка промышленных конфигураций системы; - Разработка документации на систему
4	12 мес.	- Поддержка заказчика проекта; - Доработка программных модулей под требования заказчика.

В процессе разработки системы сроки несколько изменились, так как к системе были предъявлены новые требования, а сложность некоторых задач была недооценена. Реальный график выполнения проекта приведён в пункте 1.3.

2.6. Организация процесса разработки ПО в ООО “Комплекс-Ресурс”

Для разработки проекта “Канада” была собрана команда новая разработчиков, и по итогам формирования ресурсов проекта руководством принято решение, что разработка проекта будет вестись удалённо. Фактически, это означает, что команда будет собираться крайне редко, и встаёт проблема контроля затраченного времени, статуса задач и совместного взаимодействия.

Не имея опыта в разработки программного обеспечения, руководство проекта не предусмотрело вопрос организации совместной разработки до формирования группы. Однако, большинство разработчиков (в т.ч. и автор проекта) настояли на внедрении системы управления проектами. Использование данной системы необходимо по следующим причинам:

- согласование планов работ по связанным задачам;
- отслеживание исполнения задач участниками группы разработки;
- контроль версий исходных кодов и разрешение конфликтов;
- учёт возникающих ошибок и контроль их исправления.

Руководство проекта признало необходимость подобных систем и выделило ресурсы для их внедрения. При этом было предложено реализовать не систему управления задачами, необходимую группе разработки, а полноценную систему организации совместной разработки, которая бы покрывала все потребности ООО “Комплекс-Ресурс”. Данному вопросу посвящена следующая глава дипломного проекта.

3. ВНЕДРЕНИЕ СИСТЕМ ОРГАНИЗАЦИИ СОВМЕСТНОЙ РАЗРАБОТКИ В ООО “КОМПЛЕКС-РЕСУРС”

В пункте 2.6 был сделан вывод о необходимости систем организации совместной разработки в ООО “Комплекс-Ресурс”. При этом преследовалось две цели: упрощение совместной разработки в условиях территориального распределения команды и облегчение взаимодействия с заказчиками системы.

В разделе описан процесс внедрения информационных систем. Сформулированы требования к системам, проведён обзор и анализ существующих решений, обоснован выбор системы управления задачами Redmine и прочих связанных с ней систем.

3.1. Формирование требований к комплексу информационных систем

В самом начале разработки встал вопрос внедрения комплексной системы управления проектами в ООО “Комплекс-Ресурс”. Руководством были составлены требования к подобной системе, основные из которых приведены ниже:

1) Иерархическая структура проектов
2) Централизованное хранение документов
3) Организация общего доступа с версионированием файлов
4) Защищённый доступ к системе
5) Интеграция с MS Office (в т.ч. MS Project)
6) Возможность письменного ведения дискуссий
7) Простота использования
8) Интеграция с системами контроля версий исходных кодов
9) Стоимость сервиса - не более 50$ в месяц

Кроме перечисленных требований, группе для ведения эффективной командной разработки требовались система контроля версий для исходных кодов и система управления задачами.

Использование внешних сервисов объясняется тем, что в ООО “Комплекс-Ресурс” отсутствует инфраструктура, необходимая для локального размещения информационных систем. Как минимум, для этого потребовалось бы предоставить выделенный сервер и внешний статический IP-адрес. Также

потребовалось бы установить и настроить программного обеспечения, после чего пришлось бы вести обслуживание системы в течение всего проекта. Очевидно, что затраты на организацию подобной системы существенно превысили бы поставленное ограничение (~50$ в месяц), а надёжность всего комплекса была бы значительно ниже, чем у сторонних сервисов. Исходя из перечисленного, решено реализовать необходимую функциональность при помощи стороннего хостинга или веб-сервисов. Второй вариант при этом является предпочтительным из-за низких затрат на администрирование.

Одним из участников проекта (Васильевым Н.Н.) был проведён обзор и первичный анализ существующих решений [1]. Исследование показало, что существуют Web-сервисы, которые удовлетворяют большинству поставленных требований (например, JIRA Confluence, Assembla, MS SharePoint и пр.). Однако, стоимость их использования в требуемой конфигурации в разы превышает ограничение в 50$, а сложность администрирования требует дополнительных затрат на поддержку систем.

После анализа предложенных вариантов, руководством было принято решение об отказе от комплексной системы управления проектами, а необходимую функциональность обеспечить при помощи отдельных Web-приложений. Решено выбрать и внедрить следующие информационные системы:

- система управления задачами;
- средство ведения библиографии;
- репозитории исходных кодов проекта.

Задачи формирования требований к каждой из систем, их выбор и внедрение были переданы группе разработки. Далее мы рассмотрим требования, поставленные для каждой из систем.

Требования к системе управления задачами

Система управления задачами является главной составляющей любой системы поддержки совместной разработки. С точки зрения разработчиков, система должна предоставлять следующую функциональность:

- хранение списков текущих и выполненных задач;
- группировка задач;
- возможность комментирования и обсуждения задач;
- интеграция с системой контроля версий;
- учёт времени, затраченного на решение задачи;
- наличие Web-сервисов, предоставляющих выбранную систему;

Система управления задачами используется не только разработчиками, но и руководством проекта. С учётом распределённой команды, важно обеспечить средства для контроля качества и сроков исполнения задач. Следующие возможности являются необходимыми для управления проектом:

- учёт времени, затраченного на каждую задачу;
- возможность задания сроков исполнения;
- управление правами доступа к системе.

Надо отметить, что современные системы управления задачами обладают гораздо большими возможностями. Поэтому, их анализ должен производиться с анализом дополнительных возможностей, предоставляемых системой.

Требования к системам контроля версий

При совместной разработке происходит одновременная модификация исходных кодов многими разработчиками. Поэтому, важно обеспечить отслеживание этих изменений и устранение возникающих конфликтов. Для решения подобных задач используются специальные приложения, называемые системами контроля версий (англ. - VCS). Минимальными требованиями к разрабатываемой системе являются:

- версионирование текстовых и бинарных форматов данных;
- представление истории изменений в текстовом или графическом виде;
- средства для объединения параллельных изменений и разрешения конфликтов;
- поддержка системой управления задачами выбранной системы контроля версий;
- возможность автономной работы с содержимым репозитория без доступа к сети Интернет;

Также было бы полезным обеспечить обратную связь репозитория с системой управления задачами. Например, при внесении изменений в исходные коды было бы удобно добавлять ссылки на задачи, к которым относится вносимое изменение.

Требования к системе управления библиографией

Система управления библиографией потребовалась для формирования коллекций документов и обеспечения совместной работы с ними. В процессе работы к системе управления библиографией были сформированы следующие требования:

- группировка элементов и добавление связей между ними;
- возможность привязки файлов к описаниям;
- расположение на удалённом веб-сервисе;
- возможность работы с библиографией без подключения к сервису;
- хранение нескольких версий одного документа;
- поддержка совместного доступа с разрешением конфликтов.

3.2. Выбор систем организации взаимодействия

Для каждого из рассматриваемых типов информационных систем был произведён обзор и анализ существующих решений, на основании которых были выбраны наиболее подходящие решения. Следует отметить, что данный пункт

относится к этапу проекта, который осуществлялся в конце 2009-го года, и с тех пор ситуация во многом изменилась.

Система управления задачами

При выборе системы управления задачами (иначе - систем отслеживания ошибок) ставилась задача обеспечить максимальное соответствие требованиям, указанным в предыдущем пункте. Были рассмотрены многие решения, для которых предоставляются веб-сервисы. В таблице 3.1 приведена сравнительная информация по тем из них, что соответствуют поставленным требованиям.

По таблице 3.1 видно, что коммерческие решения (Fogbugs, JIRA, Assembla и др.) более соответствуют поставленным требованиям с точки зрения общей информационной системы, но их стоимость превышает установленные пределы. Поэтому, от их использования решено отказаться.

Среди приложений с открытым исходным кодом основной проблемой является поддержка множественных проектов, которая отсутствовала в большинстве рассмотренных систем. Именно это стало причиной отказа от Bugzilla и Trac, которые в настоящее время являются наиболее популярными системами управления задачами при разработке ПО [12].

Из оставшихся систем наиболее привлекательными кажутся Bitbucket и Redmine. При этом Redmine предоставляет гораздо большую функциональность, но её потенциальными недостатками на момент выбора были невысокая распространённость (сейчас ситуация изменилась) и небольшое число участников разработки. Тем не менее, существовавшая на тот момент функциональность полностью соответствовала поставленным требованиям, и решено было использовать её в процессе разработки ИИС Neftemer.

Таблица 3.1

Сравнение характеристик популярных систем управления задачами[1]

Название	Ст-ть сервиса, $/месяц	Преимущества	Недостатки
Bitbucket	12	- Поддержка множественных проектов и репозиториев; - Неограниченный размер файлового хранилища на корпоративных тарифах;	- Обеспечивается только минимальная функциональность;
Bugzilla	5	- Поддержка многими веб-сервисами; - Интеграция с MS Project (сторонние средства); - Интеграция с IDE;	- Недружественный интерфейс; - Нет поддержки множественных проектов;
Trac	5	- Наиболее популярная из open-source систем; - Большое число плагинов и дополнений; - Простота администр-ия;	- Отсутствует поддержка множественных проектов и репозиториев;
Redmine	7	- Ориентация на иерархическую организацию проектов; - Расширяемая функциональность; - Встроенные средства коммуникации;	- Малая распространённость; - Плохая организация файлового хранилища;
FogBugz	75	- Полнофункциональный баг-трекер;	- Высокая цена; - Проприетарная VCS;
Comindwork	99	- Все предыдущие плюсы; - Интеграция с офисным пакетом MS Office; - Интеграция с MS Project; - Средства генерации отчётов;	- Высокая стоимость; - Высокая сложность администрирования; - Высокая цена; - Высокая сложность администрирования;
Assembla	99		
JIRA	150		

[1] Цены приведены по состоянию на август 2009 года. В настоящее время цены на многие продукты существенно снизились. Например, стоимость сервисов Comindwork и Assembla в минимально необходимой комплектации составляет 40-50$ в месяц, и, если бы решение принималось в текущий момент, их следовало бы рассмотреть более детально.

Основной причиной для выбора стала поддержка множества проектов с древовидной иерархией, произвольного числа репозиториев, а также поддержка форумов, блогов, и wiki-страниц внутри единого веб-интерфейса. Обзор данной системы приведён в приложении 1. Ещё одним основанием стала невысокая цена решения - всего 7.5$ в месяц в качестве веб-сервиса, предоставляемого Sourcerepo.com. В стоимость также входит резервное копирование данных и предоставление файлового хранилища [26].

В версии Redmine 1.0.2, которая появилась во время разработки проекта, появилась возможность привязки изменений к задачам с возможностью указания текущего состояния и затраченного на неё времени. В дальнейшем это позволило автоматизировать генерацию отчётов по состоянию проекта, которые предоставлялись руководству ООО “Комплекс-Ресурс” и заказчикам.

Система контроля версий (VCS)

Основным из требований к является возможность автономной работы с доступом к информации о предыдущих версиях хранимых данных. Подобные возможности являются свойством отдельного класса СКВ, называемых распределёнными системами контроля версий. К ним относятся такие системы, как Mercurial, Git, Bazaar и другие.

Система управления проектами Redmine по-умолчанию поддерживает системы контроля версий Subversion, Mercurial и Git. При помощи дополнений к Redmine с данной системой возможно использовать практически множество других VCS [27]. Таким образом, требуется сделать выбор между Mercurial и Git, которые обладают практически идентичной функциональностью и одинаково поддерживаются в выбранной системе контроля версий. По совместному решению разработчиков, было принято решение использовать Mercurial, так как все разработчики имели опыт работы с ним. Краткое описание данной системы контроля версий приведено в приложении 1.

Системы управления библиографией

Проведённый обзор показал, что среди дешёвых решений только две системы управления библиографией (Mendeley и Zotero) поддерживают хранение данных на удалённом сервере и совместную работу с библиографией. Обе системы поддерживаются различными научными центрами США, имеют большое число пользователей и схожую функциональность.

По состоянию на 2010 год, приложение Mendeley не поддерживало операционные системы семейства Windows и офисный пакет MS Office. Поскольку для оформления документации в ООО “Комплекс-Ресурс” используется именно ПО Microsoft, то для управления библиографией решено использовать Zotero. Подробный обзор данной системы приведён в в приложении 1. Выбранная система соответствует всем требованиям, поставленным в пункте 3.1, а также имеет ряд дополнительных возможностей:

- Поддержка внешних хранилищ файлов (через WebDAV);
- Хранение заметок и категоризация документов;
- Автоматическая генерация библиографии с использованием пользовательских стилей;
- Поиск по документам и полям;
- Хранение заметок о документах [30].

Zotero в качестве веб-сервиса поддерживается Центром Истории и Новых Носителей Данных (CHNM), который ведёт разработку данного продукта. Предоставляется бесплатных хостинг с размером хранилища данных в 100 Мб и несколько платных тарифов с более высокой ёмкостью. С учётом того, что хранилище файлов может быть реализовано на базе хостинга Sourcerepo.com, в Zotero требуется хранить только метаданные и заметки о книгах, для чего ста мегабайт должно хватить с избытком. Поэтому, использование данного решения для проекта не требует дополнительных затрат.

В рамках проекта приложение Zotero показало себя как удобное средство для работы с библиографией, которое сильно упрощает разработку технической документации, так как может быть адаптировано практически под любые стандарты. В качестве примера, список использованных источников и внутритекстовые ссылки в данном дипломном проекте сгенерированы с помощью Zotero.

3.3. Внедрение информационных систем в ООО “Комплекс-Ресурс”

3.3.1. Внедрение системы управления задачами Redmine

Система Redmine была развёрнута на базе интернет-сервиса Sourcerepo.com. Был выбран минимальный тарифный план, который поддерживает произвольное число пользователей и проектов. Стоимость обслуживания составила 7.5$ в месяц, что значительно ниже поставленных требований. При необходимости, тарифный план может быть изменён в любой момент без нарушения работы системы и необходимости переноса данных. На этапе внедрения системы перед автором были поставлены следующие задачи:

- создание иерархии проектов, подготовка репозиториев;
- конфигурирование системы и обеспечение максимального удобства работы с ней;
- внесение в конфигурации основных задач и контрольных точек проекта;
- регистрация пользователей и настройка прав доступа к системе настройка системы уведомлений;
- наполнение wiki-страниц и форумов системы;
- администрирование содержимого системы (контроль актуальности задач, внесение изменений, поддержка пользователей и т.п.)[1].

[1] Администрирование системы управления проектами как внешнего ресурса (взаимодействие с Sourcerepo.com, контроль целостности, резервное

При подготовке системы было принято решение разделить проекты в соответствии с логической архитектурой системы. При этом, не требовалось выделять каждый модуль в отдельный проект, так как Redmine поддерживает дополнительные средства группировки. Проекты также не разбиваются по этапам, так как это удобно делать при помощи механизма “Версий”. В итоге, построена древовидная иерархия проектов (см. рис. 3.1).

Модули форумов, новостей, wiki-страниц, документов и файлов были активированы только для верхних уровней иерархии. Это было сделано для того, чтобы упростить администрирование и поиск необходимых данных. Все перечисленные модули имеют внутренние средства группировки, поэтому подобный подход не усложнит работу с системой.

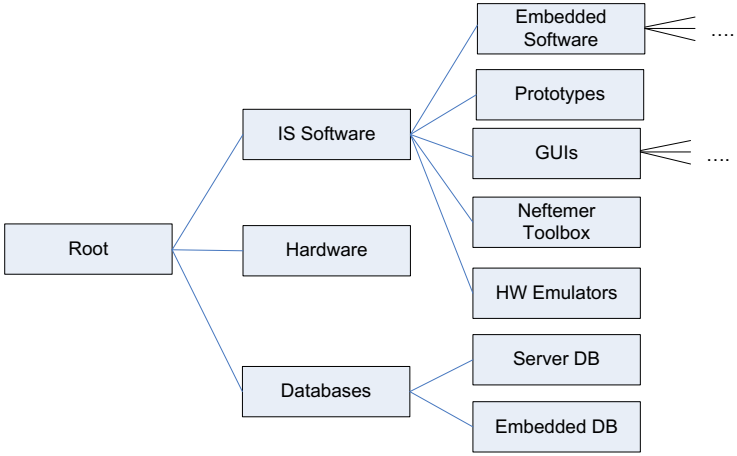

Рис. 3.1. Структура проектов в Redmine

Следующей задачей стал выбор архитектуры репозиториев проекта. Автор проекта считает, что в рамках проекта “Канада” было бы удобным использование единого репозитория со сложной внутренней структурой. В случае с Mercurial это позволило бы вести независимую разработку в отдельных

копирование) осуществлялось Н.Н. Васильевым. Администрирование содержимого системы на первых этапах осуществлялось совместно.

ветках, но при необходимости объединять ревизии (например, для формирования релизовой сборки программ). Однако, руководителем проекта было принято решение создать несколько репозиториев, изолировав независящие друг от друга части (т.е. те части, где зависимость ограничивается спецификацией). В итоге был сформировано несколько основных репозиториев (см. табл. 3.2.) и несколько дополнительных репозиториев, для частных задач, которые в дипломном проекте не рассматриваются.

Таблица 3.2

Структура репозиториев проекта “Канада”

Название	Проект	Описание
NM_Software	IS Software	- Основной репозиторий для встраиваемой части системы. Содержит исходные коды на C/C++/C#, Описания конфигураций, Отладочные конфигурации системы и тесты для программного обеспечения.
NM_Toolbox	Neftemer Toolbox	- Репозиторий инструментального ПО для терминального компьютера. Содержит преимущественно проекты на Java: Графические интерфейсы, конвертеры данных из предыдущих систем, генераторы тестовых данных и т.п.
NC_GUI	GUIs	- Содержит проекты, реализующие пользовательские интерфейсы для встраиваемых частей системы в различных конфигурациях аппаратной части.
NM_Databases	Databases	- Репозиторий, который содержит SQL-скрипты, позволяющие восстановить архитектуру баз данных и системные таблицы.

В процессе конфигурирования системы оказалось, что трёх базовых типов задач (Feature, Bug и Support) недостаточно для выполнения операций. Поэтому, были добавлены дополнительные типы задач. Последним этапом подготовки системы для использования было внесение пользователей. Для этого было создано 4 группы пользователей: гость, администратор, менеджер, разработчик и наблюдатель. Права доступа для каждой из групп приведены в таблице 3.4. Тем не менее, все участники группы разработки числились в проектах с правами

доступа не ниже наблюдателя, т.е. каждый мог знать о состоянии разработки по всем направлениям.

Таблица 3.3

Типы задач в системе управления проектами Redmine

Тип задачи	Описание
Feature	- Добавление новой функциональности
Refactoring	- Внесение в исходные коды изменений, не влияющих на функциональность программы, но улучшающих архитектуру исходных кодов.
Bug	- Исправление ошибок в программном обеспечении.
Support	- Поддержка участников проекта (консультирование, подготовка данных и тестовых сборок, …).
Testing	- Тестирование программных и аппаратных составляющих системы.
Documentation	- Задачи, связанные с разработкой и исправлением программной документации.
Research	- Исследовательские задачи, связанные с исследованием, обзором и анализом существующих решений, формированием предложений по архитектуре системы и т.п.
Administration	Задачи, связанные с поддержкой системы Redmine и прочих составляющих инфраструктуры проекта.

Таблица 3.4

Уровни доступа в системе управления проектами Redmine

Уровень доступа	Возможности
Гость	(Незарегистрированный вход) Доступ к приветственной странице проекта и разрешением на отправку запросов на авторизацию. Просмотр и модификация любого другого содержимого запрещены.
Наблюдатель	- Просмотр и комментирование всех задач проекта; доступ к файлам и документам для считывания; создание собственных задач для других разработчиков (например, фиксация ошибки в программе).
Разработчик	- Доступ к файлам, документам и репозиториям с правом записи; Редактирование wiki-страниц
Менеджер	- Права разработчика + регулирование прав доступа к проекту и репозиторию; удаление файлов; создание новых версий и групп задач.
Администратор	- Полный доступ к проекту; блокировка пользователей
Суперпользователь	- Конфигурирование Redmine, назначение администраторов; удаление пользователей;

Назначить исполнителем задачи можно только разработчика, менеджера и администратора. При этом администратору могут быть назначены только задачи типов “Support” и “Administration”. Однако, система Redmine позволяет назначать несколько ролей каждому из пользователь, а итоговый уровень доступа определяется по их совокупности.

При регистрации каждый пользователь системы должен предоставить свой адрес электронной почты (корпоративная почта в ООО “Комплекс-Ресурс отсутствует”), а пользователи репозиториев (разработчик и выше) – публичные ssh-rsa2 ключи, необходимые для установления шифрованного канала связи с системой контроля версий. После проверки администратором и задания будущей роли на указанный адрес отправляется временный пароль, и пользователь может зайти в систему. За время участия автора в проекте в Redmine было зарегистрировано около двадцати пользователей с различными правами доступа.

В рамках внедрения Redmine были решены и задачи внесения содержимого в систему. Использовались как форумы, так и wiki-страницы, которые были связаны с дополнительной документацией на систему.

На верхнем уровне (проект Root) были активированы модули новостей, календаря, диаграмм Ганта и учёта времени. Модуль новостей использовался для передачи сообщений об изменениях и важных событиях, а прочие модули были использованы как дополнительные средства визуализации задач и процесса разработки.

3.3.2. Интеграция информационных систем

После настройки Redmine потребовалось обеспечить взаимодействие системы с Zotero и программным обеспечением персональных компьютеров разработчиков. На рис. 3.2 приведена схема взаимодействия различных программных продуктов. Подробная информация по организации взаимодействия будет приведена далее.

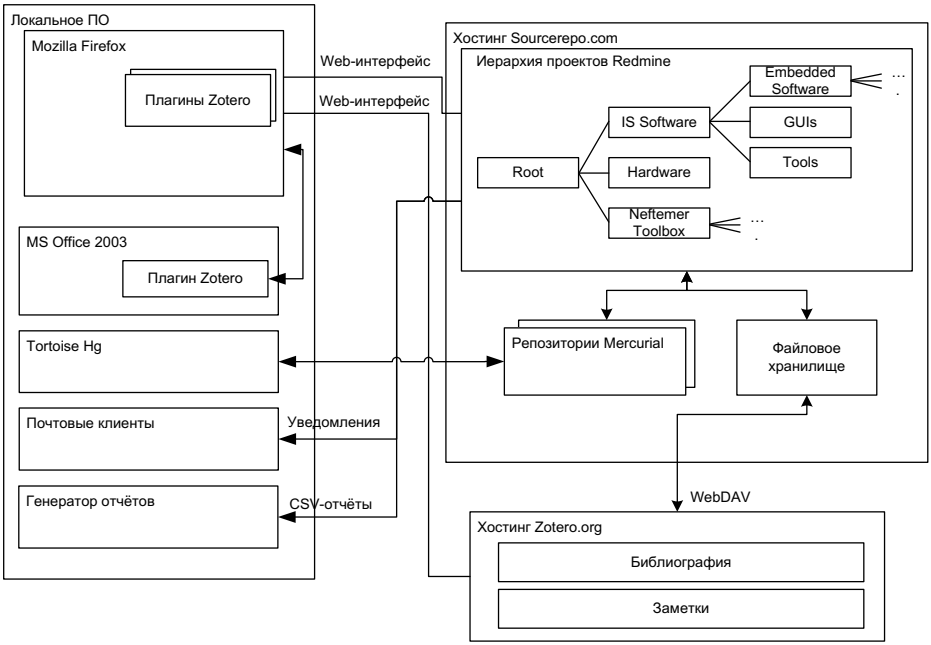

Рис. 3.2. Схема взаимодействия информационных систем поддержки совместной разработки

Взаимодействие пользователя с информационными системами

Все используемые системы (Redmine, Mercurial и Zotero) имеют свои web-интерфейсы, поэтому для работы с ними во многих случаях достаточно любого браузера. Через него пользователь может:

- работать с системой управления Redmine (вся функциональность реализуется через веб-интерфейс);
- просматривать историю изменений в репозиториях и получать различные версии исходных кодов;
- просматривать документацию в Zotero и оставлять заметки.

Таким образом, доступа через браузер достаточно для людей, не вносящих изменения в исходные коды или документацию (например, для заказчика или руководства проекта). Поскольку все службы расположены на

внешних сервисах, то они доступны пользователю при наличии доступа к сети Интернет. Для уведомления пользователей Redmine поддерживает уведомления по почте, и в случае использования почтовых клиентов уведомления об изменениях в системе контроля задач приходят практически мгновенно.

Для доступа к ресурсам требуется пройти аутентификацию, логины и пароли предоставляются администратором проекта и могут быть заблокированы в любой момент. Безопасность доступа обеспечивается за счёт шифрованного доступа (во всех сервисах используется HTTPS). Подобных подход является типовым решением и используется практически во всех коммерческих web-сервисах.

Для полноценной работы с репозиториями Mercurial должно быть установлено специальное приложение. Оно обладает консольным интерфейсом, но для него существует множество графических клиентов (например, Tortoise Hg), которые сильно упрощают анализ изменений.

Интеграция Zotero с прочими системами

В соответствии с разработанной архитектурой, Zotero использует файловое хранилище на хостинге Sourcerepo.com. Для этого используется специализированный протокол WebDAV (Web-based Distributed Authoring and Versioning), который обеспечивает совместный доступ к файлам с разрешением конфликтов [19]. Для поддержки WebDAV в Redmine был добавлен специальный плагин. При этом, для Zotero пришлось создать отдельного пользователя, который имел доступ только к файловому хранилищу.

Для Zotero недостаточно web-интерфейса, так как в нём сильно ограничены возможности группировки и комментирования документов. Поэтому, для редактирования библиотеки удобно использовать набор плагинов для браузера Mozilla Firefox, которые реализуют пользовательский интерфейс Zotero (см. приложение 1) и обеспечивают его интеграцию со всеми остальными приложениями (например, офисными пакетами).

3.4. Организация взаимодействия с заказчиками

Очевидно, что на этапе разработки и эксплуатационного тестирования системы потребуется обеспечить поддержку заказчика: корректировку требований к системе, добавление новой функциональности, исправление ошибок и т.д. Все перечисленные задачи решаются с помощью коллаборативных CRM-систем [9], и в рамках проекта были рассмотрены возможности их внедрения в ООО “Комплекс-Ресурс”. Бюджет проекта недостаточен для использования интегрированных ERP-систем, поэтому решено использовать обособленную CRM-систему с поддержкой минимально-необходимой функциональности. Было выделено три возможных подхода:

- использование CRM-системы на внешнем хостинге;
- использование CRM-системы на базе стороннего веб-сервиса;
- реализация подобия ERP-системы на базе Redmine.

Прежде всего, нужно сформировать требования к функциональности CRM-системы. После обсуждения с руководством проекта и представителями заказчика был сформирован следующий список требований к возможностям системы:

- подача сообщений об обнаруженных ошибках;
- контролем процесса устранения обнаруженных недостатков;
- подачи предложений по добавлению функциональности в последующих версиях и внесению изменений в текущую версию;
- получение технической поддержки через систему;
- сохранение истории вопросов и ответов.

Выбор варианта реализации CRM-функциональности

Поверхностный обзор проприетарных CRM-систем показал, что их совокупная стоимость не соответствует выделенным ресурсам. Однако, существует большое число открытых CRM-систем (например, SugarCRM,

ADempiere или VTiger). При их использовании потребовалось бы платить только за хостинг, а в случае использования подобных систем на базе веб-сервиса упростилось бы администрирование системы.

Альтернативным вариантом является использование системы управления задачами Redmine, которая уже была внедрена для организации совместной разработке. С помощью механизмов разграничения прав, можно предоставить заказчику доступ только к отдельному проекту, в котором он сможет взаимодействовать с другими участниками проекта. Первичный анализ показал, что с помощью Redmine можно реализовать всю требуемую функциональность. Поэтому, с целью экономии ресурсов был выбран именно этот вариант.

Реализация модулей взаимодействия с клиентами в Redmine

Для реализации CRM в Redmine был создан дополнительный проект. В нём были отключены все модули кроме тех, что описаны в таблице 3.5. Предполагается иметь общий для всех заказчиков CRM-проект с общей информацией, а также ряд закрытых проектов, доступ к каждому из которых имеет только один заказчик.

Таблица 3.5

Средства, предоставляемые заказчику в рамках CRM-подпроекта

Модуль	Возможности
Новости	- Уведомления о выходе новых версий ИИС Neftemer; - Сообщения о прочих значимых событиях.
Задачи	- Добавление задач и отчётов об ошибках; - Отслеживание изменений по созданным заявкам; - Отправка заявок о поддержке;
Форумы	- Обсуждение общих вопросов с разработчиками ИИС Neftemer;
Wiki-страницы	- Часто задаваемые вопросы; - Краткие описания модулей системы; - Руководства по работе с системой Redmine
Документы	- Руководства пользователя; - Прочая документация, необходимая заказчику;
Файлы	- Релизовые сборки программного обеспечения - Файлы конфигураций

Совмещение в одной системе внутренней разработки и CRM-систем для взаимодействия с заказчиком требует больших усилий с точки зрения информационной безопасности, так как некоторые детали проекта (в основном, математическое обеспечение системы) раскрывать нельзя.

Для обеспечения безопасной работы с заказчиками в Redmine был создан дополнительный тип пользователей – Customer. Он обладает ограниченным доступом к проекту CRM, а остальные проекта для него невидимы и недоступны для считывания. При этом, изменение прав пользователя с заказчика на любую из проектных ролей возможно только для аккаунта Суперпользователя, доступ к которому имеет только руководитель проекта. Все сообщения, выкладываемые в проекте CRM, проходят предварительную проверку. Подобный подход не может полностью защитить от утечки информации, но исключает её случайное попадание в общий доступ.

Можно заключить, что подобная система действительно позволяет выполнять необходимые функции CRM-систем. Подход был успешно опробован при взаимодействии с потенциальным заказчиком в течение разработки дополнительных модулей и тестирования системы заказчиком.

4. РАЗРАБОТКА ИНФОРМАЦИОННОЙ СИСТЕМЫ И КОММУНИКАЦИОННЫХ МОДУЛЕЙ

4.1. Выбор средств разработки

Конфигурационные файлы системы

Конфигурационный и информационный файлы решено представлять в формате XML (eXtended Markup Language – расширяемый язык разметки). Это даёт следующие возможности:

- обработка файлов стандартными программами и библиотеками;
- масштабируемость файлов;
- возможность валидации стандартными средствами.

Использование стандартных средств работы с XML позволяет достаточно легко создавать и редактировать файлы даже при отсутствии программ-визуализаторов с поддержкой настройки. Большинство средств (FoxEdit, XMLSpy, XmlEditor и др.) поддерживают контекстные подсказки, верификацию и т.п., поэтому в демонстрационной версии системы не требуется разработка специальных средств конфигурирования системы.

Для поддержки разработки и верификации для файлов разработаны XSD-схемы с описанием основных структур и элементов. В настоящее время они не позволяют контролировать все возможные ошибки, но в будущем это будет исправлено.

Выбор сред исполнения и языков программирования

В соответствии с разработанной архитектурой, программное обеспечение системы должно быть кроссплатформенным, надёжным, производительным. В тоже время, должно быть минимизировано время программной реализации модулей демонстрационной системы.

Есть два подхода к программной реализации: с использованием языков, компилируемых в машинные коды системы или же использованием дополнительной среды исполнения (виртуальные машины, интерпретаторы и

т.п.). Исполняемая среда предоставляет набор средств, которые позволяют повысить надёжность системы (контроль доступа к памяти, преобразований типов и пр.) и снизить время разработки, что важно для демонстрационной версии. В тоже время подобный подход снижает производительность программ, что повышает требования к аппаратной части системы. Соответственно, это приводит к росту стоимости системы в целом.

По итогам проведённого анализа решено на C++ реализовать те программные модули, которые без существенных изменений будут перенесены из демонстрационной версии системы в промышленную версию. Для всех прочих модулей решено использовать C# и среду исполнения .NET Compact Framework 2.0.

C++ был выбран потому, что является одним из наиболее распространённых языков программирования. Сам по себе он является кроссплатформенным, и для него существует множество кроссплатформенных библиотек, решающих частные задачи (например, считывание и разбор конфигурационных файлов). Кроме того, у всех участников проекта имелся опыт программирования на данном языке.

Для демонстрационных элементов ИИС Neftemer .NET взят потому, что все используемые операционные системы поддерживают его встраиваемую версию (.NET Compact Framework версии 2.0). Функциональность данной версии несколько отличается от .NET Framework 2.0, который являлся наиболее свежей версией в 2005-м году (при создании CE 6.0), но её хватает для решения большинства задач. Ограничения связаны в основном с графическими средствами, но для встраиваемых интерфейсов достаточно примитивной графики.

Язык C# для реализации ПО на .NET был выбран потому, что среда разработки MVS2005 содержит набор средств для быстрого проектирования и рефакторинга кода на данном языке. Использование C# не приводит к повторной реализации внутренних библиотек системы, так как платформа .NET

поддерживает C++ через CLR (Common Language Runtime). Все библиотеки впоследствии удалось реализовать таким образом, что они были совместимы с CLR, а также C++ для Windows XP и Windows CE, у которых программные интерфейсы (API) несколько различаются.

Выбор средств для программной реализации

Поскольку демонстрационный стенд использует операционные системы Windows Server, Windows XP Embedded и Windows CE 6.0, то для программной реализации элементов системы предпочтительно использовать средства разработки, поставляемые компанией Microsoft. Разработка для всех трёх ОС возможна в рамках одной интегрированной среды разработки (IDE – Integrated Development Environment) – Microsoft Visual Studio 2005 (далее MVS2005). Использование более новых версий данной IDE невозможно, так как средства разработки для CE 6.0 поставляются в отдельном дополнении, доступном только для данной среды разработки. С точки зрения разрабатываемой системы, MVS2005 позволяет:

- вести разработку на языках C++ и C#;
- производить отладку программного обеспечения (в т.ч. удалённую);
- редактировать конфигурационные файлы и их спецификаций;
- производить модульное и интеграционное тестирование программных компонентов системы;
- производить верификацию конфигурационных файлов.

Таким образом, выбранная среда разработки обеспечивает большую часть необходимой функциональности. Некоторая дополнительная функциональность была реализована при помощи расширений (см. табл. 4.1.). Для реверс-инжиниринга кода и построения его поведенческих описаний используется среда Enterprise Architect 8, которая была использована при проектировании системы (см. п. 4.3).

Таблица 4.1

Список расширений, используемых в MVS2005

Название плагина	Возможности / применение
Windows Embedded CE 6.0 Tools	
MVC CE Platform Builder IDE	Встроенная среда для разработки пользовательских конфигураций операционной системы Windows CE.
MVS Doxy	Плагин интеграции с генератором программной документации Doxygen. Используется для автоматической генерации шаблонов программной документации.
MVS Spellchecker	Проверка правописания в комментариях к программному коду.
MVS Mercurial Integration	Средство для интеграции MVS2005 с системой контроля версий Mercurial, используемой в проекте.

Для редактирования XML- и XSD-файлов проекта была использована среда EditiX XML 2010. По сравнению с MVS2005, она предоставляет более широкий спектр функциональности при разработке XSD-спецификаций. Также в данной среде имеется возможность автоматической генерации текстовой спецификации на формат файлов и средство выявления недокументированных опций или потенциальных ошибок в схеме [ссылка].

Выбор библиотек

При разработке программного обеспечения было использовано множество готовых библиотек и типовых решений, что упростило разработку кода и его портирование на другие платформы. В таблице 4.2 приведён перечень основных библиотек, которые были использованы в рамках рассматриваемых в дипломе составляющих ИИС Neftemer.

Существуют библиотеки, которые реализуют сетевое взаимодействие по протоколу Modbus и его внутреннее адресное пространство. Тем не менее, в проекте решено было от них отказаться и реализовать свою библиотеку, так как в рамках разработанной архитектуры требуются возможности, которые плохо поддерживаются существующими решениями. К ним относятся:

- расширяемый набор команд протокола;

- расширяемое адресное пространство протокола;
- независимость от физической реализации протокола;
- наличие реализаций для .NET и C++.

Таблица 4.2

Список библиотек, использованных при разработке на C++

Название плагина	Возможности / применение
STL	Библиотека шаблонов для C++, в настоящее время входящая в стандарт языка. Прежде всего, включает средства для работы с контейнерами и сложными типами данных.
Boost	Типовая библиотека расширений для C++, которая включает множество типовых решений. В программном обеспечении ИИС Neftemer были использованы: - средства создания многопоточных приложений; - средства работы с динамической памятью; - средства обработки текстовых файлов; - средства модульного тестирования исходных кодов.
POCO	Библиотека для создания сетевых кроссплатформенных приложений. Из неё были использованы средства для работы с TCP-сокетами
TinyXML	Парсинг и генерация XML файлов. Используется для считывания конфигурационных файлов системы

Разработка собственной интерфейсной библиотеки не потребует много времени, так как в рамках предыдущих работ в ООО “Комплекс-ресурс” автором была разработана библиотека со схожей функциональностью.

Для встраиваемых приложений, реализованных на C#, дополнительных библиотек не требуется, так как вся необходимая функциональность реализуется средствами .NET Compact Framework.

4.2. Разработка средств конфигурирования системы

При проектировании формировании концепции системы предполагалось, что будет использоваться единый конфигурационный файл. Однако, существуют элементы описания, которые не влияют на функционирование системы. Например, к ним относятся аннотации для системных сообщений, типов команд

устройств. Данная информация меняется дожжет меняться даже в рамках единой конфигурации (например, при локализации системы). Поэтому, было принято решение сделать описание системы в двух файлах, которые в работе называются конфигурационным и информационным. Далее кратко рассмотрим каждый из них.

Конфигурационный файл системы

Конфигурационный файл включает информацию о системе, на которой развёртывается система, и настройки для всех приложений модуля, которые запускаются с помощью Загрузчика ПО. Системная информация включает:

К системной информации относятся:

- серийный номер, адрес и имя устройства;
- информация о конфигурации (имя, версия);
- информация о системе: операционная система, используемая архитектура аппаратной части (например, x86);
- путь к директории, хранящей конфигурацию системы.

После системной информации следует список запускаемых приложений. В конфигурационном файле хранится следующая информация:

- список и запускаемых программ, их версии (для контроля);
- порядок и режим запуска программ;
- время, отводимое на инициализацию приложения;
- режимы контроля программы;
- описание внешнего интерфейса программы (используемый физический интерфейс и его параметры).

Конфигурационный файл, в первую очередь, обрабатывается модулем запуска ПО и супервизором системы. После запуска, каждая программа обращается к файлу конфигурации и считывает свои настройки, а также системную информацию.

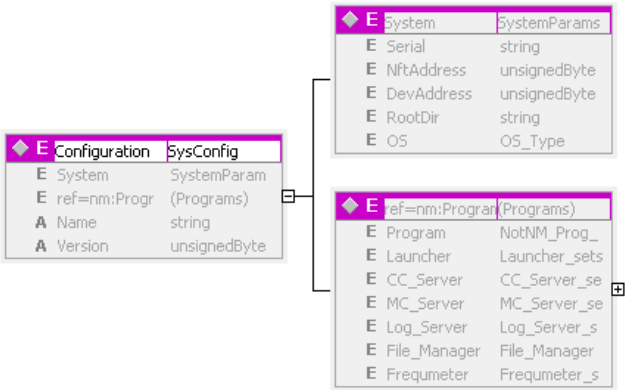

Рис. 4.1. Схема верхнего уровня конфигурационного файла ИИС Neftemer

Каждое приложение обладает уникальным номером, который передаётся при запуске программ. Это позволяет запускать несколько приложений одного типа. После запуска программа считывает необходимые ей настройки и должна за отведённое время инициализировать модуль самодиагностики, к которому подключается супервизор системы. Супервизор в системе всегда имеет нулевой идентификатор и запускается первым.

Информационный файл системы

Информационный файл системы ориентирован на то, чтобы предоставить пользователю и сторонним средствам максимально полную информацию о системе. Это может быть использовано, например, для представления справочной информации в программах визуализации. Информационный файл включает в себя следующие элементы (см. схему на рис. 3.2):

1) Общая информация о системе
   - версия;
   - текстовое описание;
   - ссылки на документацию;
2) Информация о производителе
3) Информация о заказчике
4) Описание элементов системы
   - программы и их модули;
   - информация о базовых модулях;
   - типы и коды событий;
   - связи между программами и модулями;
5) Стандартные конфигурации

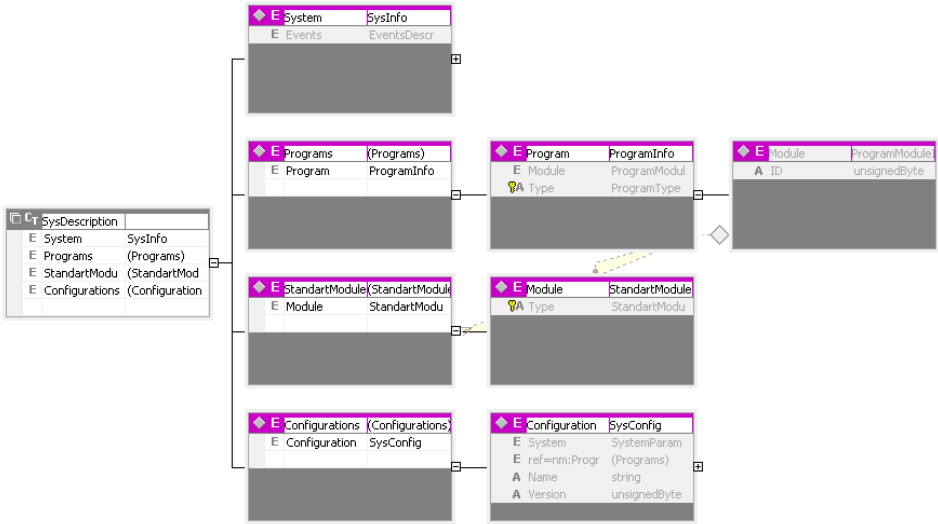

Рис. 4.2. Схема верхних уровней информационного файла ИИС Neftemer

Как и в программном обеспечении, используется иерархическое построение программных модулей. Для каждого из них возможно перечислить список включаемых модулей нижнего уровня и один модуль, поведение которого расширяется (например, сервера данных приложений строятся на базе абстрактного модуля, реализующего сервер во внутреннем протоколе системы).

Каждая программа представляет собой список включаемых модулей. При этом для каждого из них имеется уникальный идентификатор, что позволяет точно определить источник сообщения при обработке данных, полученных по сетевому протоколу.

В конце информационного файла системы следует перечень стандартных конфигураций. Этот блок зарезервирован для того, чтобы предоставить пользователю возможность выбора конфигураций в терминальном приложении NM_Toolbox, которое было разработано в рамках проекта.

Средства верификации XML-файлов

Для верификации XML-файлов была разработана их спецификация на языке XSD. В данной спецификации ставилась цель минимизировать риск ошибки со стороны пользователя, поэтому осуществлялись контроль структуры файла. проверка задаваемых значений (с помощью диапазонов и перечислений), зависимости между элементами описания.

Самым сложным элементом верификатора является проверка зависимостей между модулями в информационном файле системы, так как для этого недостаточно возможностей, предоставляемых тегами XSD-описания. Пришлось использовать дополнительные скрипты, с помощью которых выявлялись ссылки на несуществующие модули, нарушения древовидной иерархии (зацикливание ссылок, отсутствие ссылки на абстрактный модуль Root в описании).

Также с помощью XSD-схем были задокументированы поля описания, что позволило при редактировании XML-файлов показывать контекстную справку с описанием конфигурации. Примеры конфигурационных файлов системы и их XSD-спецификаций приведены в приложении 2.

4.3. Программная реализация элементов системы

TODO #1: Включить согласованный текст пункта

4.4. Разработка внешней базы данных

TODO #2: Включить согласованный текст пункта

4.5. Подготовка программной документации на систему

Разработка документации на систему (в т.ч. и программной) являлась одной из основных задач автора дипломного проекта. Так как большая часть документов не подлежало распространению за пределами ООО “Комплекс-Ресурс”, к документации не было предъявлено жёстких требований кроме полноты и единой стилистики оформления. Тем не менее, в рамках работы решено разработать пакет документации на систему, близкий к рекомендациям

SWEBOK (Software Engineering Body of Knowledge) и требованиям ЕСПД (Единая система программной документации). Был выбран следующий перечень необходимой документации:

- подробные спецификации на программные модули в формате UML;
- технические задания на разработку основных модулей системы;
- руководства пользователя;
- руководства разработчика (описания внутренней архитектуры проектов, процесса развёртывания средств разработки и т.п.);
- программы тестирования системы и её составляющих;
- документация на исходные коды системы.

Всего было разработано около тысячи страниц текстовой документации.

При проектировании системы и разработке спецификаций использовалась единая среда разработки в UML - Enterprise Architect 8. Данная среда обладает встроенной поддержкой методологии ICONIX, что позволило точно следовать рекомендациям данного подхода. Среда Enterprise Architect 8 включает средства генерации текстовой документации на систему, что позволило сформировать автоматически сгенерировать документы для разрабатываемого комплекса программной документации.

Кроме указанного выше, к программной документации на систему также относятся XSD-спецификации конфигурационных файлов системы, документация на исходные коды, сгенерированная при помощи Doxygen.

5. АНАЛИЗ РЕЗУЛЬТАТОВ РАЗРАБОТКИ

В данном разделе подводятся основные итоги проекта “Канада”. Произведён анализ соответствия разработанного программного обеспечения поставленным требованиям. При этом отдельно рассмотрены архитектура системы и её реализация. Также произведён анализ результатов внедрения систем поддержки совместной разработки в ООО “Комплекс-Ресурс”. Для всех элементов произведена оценка экономической эффективности разработки.

В пункте 5.2 произведён анализ вклада автора в проект “Канада”. К сожалению, в соответствии с требованиями руководства проекта, невозможно рассказать о вкладе других участников проекта и указать их заслуги в разработке системы.

В пункте 5.4 указаны перспективы дальнейшего развития системы и общие результаты проекта.

5.1. Анализ соответствия разработанных модулей поставленным требованиям

При разработке архитектуры были предприняты различные меры, чтобы обеспечить гибкость, конфигурируемость и надёжность системы. В частности, была реализована модульная архитектура системы с единым протоколом взаимодействия между модулями, который может быть реализован на базе любого физического интерфейса. Учитывая то, что вся система развёртывается на базе единого файла конфигурации, описывающего отдельные модули, можно говорить о высокой степени гибкости системы и адаптируемости системы.

Проведённая разработка подтвердила ожидания. Были опробованы различные конфигурации системы с развёртыванием на одном или нескольких устройствах под управлением различных операционных систем. Все конфигурации успешно запустились, а неверные настройки были отвергнуты ещё на этапе верификации конфигурационного файла системы.

Надёжность системы обеспечена за счёт использования встроенных в программные модули средств самодиагностики и программных модулей.

Наиболее критичные участки, связанные с заменой конфигураций и исполняемых файлов, защищены дополнительными средствами контроля, которые в случае неисправности позволяют вернуть аппаратный модуль к исходному состоянию.

Надёжность доставки данных обеспечена за счёт использования протокола связи с двойным квитированием и контрольными суммами, которые с большой вероятностью позволяют выявить ошибку при передаче данных. Все модули поддерживают внутреннюю буферизацию для всех типов передаваемых данных. Поэтому, при нормальной работе системы потеря данных практически невероятна, но во внутреннем протоколе предусмотрено выявление потерь данных и передача информации о них на верхний уровень. Там данные могут быть аппроксимированы, и в большинстве случаев ошибка не скажется на работе системы.

5.2. Анализ вклада автора в разработку системы

Вклад в организацию разработки

Автор проекта был инициатором внедрения системы управления проектами в ООО “Комплекс-Ресурс”. Была обоснована необходимость использования, после чего автор проекта и Н.Н. Васильев совместно провели анализ различных вариантов и выбрали вариант, соответствующий требованиям. Были выбраны и внедрены система

В рамках работы была произведена настройка систем, а с помощью дополнительных сервисов удалось обеспечить не только управление задачами, но и хранение библиографий, что было необходимо ввиду большого числа документов и дополнительных материалов. Также на базе Redmine удалось построить примитивную CRM-систему коллаборационной направленности. Эта система позволила упростить взаимодействие с заказчиком, что несомненно будет дополнительным плюсом при принятии решения о заказе ИИС Neftemer.

Перечисленные действия позволили упростить взаимодействие команды разработки и сделать сам процесс более прозрачным. Несомненно, благодаря этому удалось существенно повысить качество разработанной системы.

Вклад в разработку системы

На рис. 5.1 показана структура отладочной версии системы, которая была создана на третьем этапе проекта. На ней отмечена большая часть модулей системы, для которых была выполнена детальная проработка или реализация.

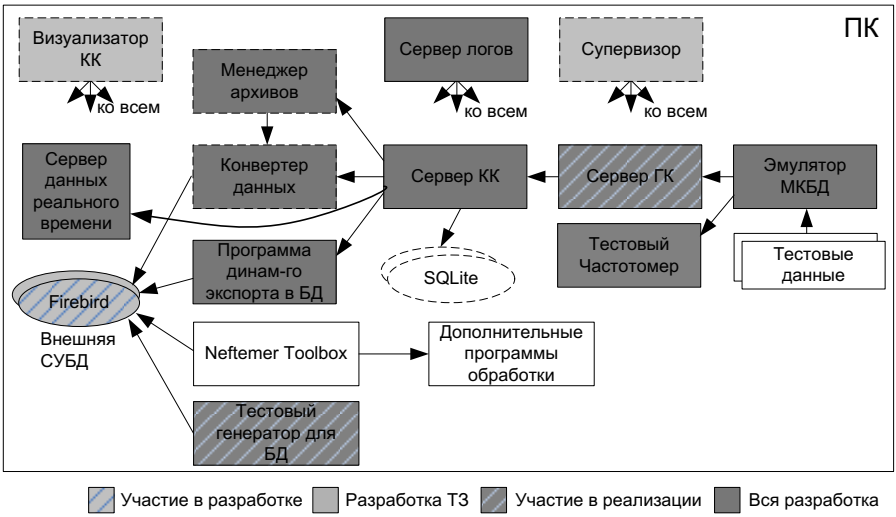

Рис. 5.1. Вклад Ненашева О.В. в разработку ИИС Neftemer на примере тестового комплекса

На схеме отображены далеко не все спроектированные компоненты, но большая часть оставшихся была реализована только на уровне спецификаций для промышленной версии ИИС Neftemer или же не подлежит разглашению.

Как видно из рис. 5.1, большая часть модулей для встраиваемой части системы была частично или полностью разработана автором проекта. При этом была разработана архитектура системы, внутренний протокол обмена и решены все проблемы взаимодействия между модулями.

Кроме разработки и реализации системы, автор занимался автор проекта занимался разработкой полного комплекса программной документации, близкого по своему составу к требованиям ЕСПД. За время проекта было разработано около тысячи страниц программной документации. Также было произведено документирование исходных кодов системы в соответствии с Doxygen. Общий объём исходных кодов проекта, реализованных автором проекта, составил около 120 тысяч строка.

В данном пункте перечислены не все задачи, которыми автор занимался во время проекта. Тем не менее, надо заключить, что вклад автора в разработку системы достаточно велик.

5.3. Анализ экономической эффективности разработки

Оценка экономической эффективности разработки ИС Neftemer

Разработанная информационная система не может принести предприятию мгновенную выгоду. Фактически, с заменой старых информационных систем на ИС Neftemer ООО “Комплекс-Ресурс” не планирует повышать цены на поставляемые анализаторы нефти (система является приложением к ним), а экономическая отдача от системы может заключаться только в увеличении спроса. В случае потенциального заказчика, при успешном проведении эксплуатационного тестирования ожидается, что им будет осуществлён заказ на несколько десятков аппаратных комплексов (комплект оборудования для одного участка добычи) в течение трёх лет после двух лет тестирования. Предполагается, что перечисление средств будет идти равномерно.

К сожалению, в соответствии с договором о неразглашении информации, нельзя привести никаких экономических показателей, но можно сказать, что на проект “Канада” была затрачена сумма, приблизительно равная стоимости двух измерительных комплексов. В соответствии с оценками, внутренняя норма доходности ООО “Комплекс-Ресурс” составляет 12%, а доходность при продаже комплексов - 30%. Таким образом, можно получить следующую оценки

эффективности проекта, используя в качестве условной единицы стоимость одного комплекса. В соответствии с рекомендациями из [7], была произведена оценка экономических показателей проекта. При производстве систем оплата производится при получении устройств, поэтому в денежных потоках сразу приведены ожидаемые доходы при поставке 10 комплексов в год. В таблице 5.1 приведены данные по денежным потокам, на основе которых вычислялись оценки экономической эффективности.

Таблица 5.1

Денежные потоки проекта “Канада”

Год	2009	2010	2011	2012	2013	2014
Этап	Разработка		Тестир-ование	Поставки системы		
Денежный поток, ед.	-1.250	-0.700	-0.700	3.000	3.000	3.000
Прив-я стоим-ть (2009), ед.	-1.250	-0.625	-0.558	2.135	1.907	1.702
Прив-я стоим-ть (2014), ед.	-2.203	-1.101	-0.983	3.763	3.360	3.000
Накопленная сумма, ед.	-1.250	-1.950	-2.650	0.350	3.350	6.350
Накопленная сумма диск., ед.	-1.250	-1.875	-2.433	-0.298	1.609	3.311

Оценки экономических показателей проекта, полученные при анализе системы, приведены в таблице 5.2.

Таблица 5.2

Оценки экономической эффективности проекта “Канада”

Индекс	Полное название	Значение
ARR	Бухгалтерская доходность проекта	121,6%
NPV	Чистая приведённая стоимость	3.311 ед.
NFV	Чистая приведённая стоимость в будущем	5.835 ед.
PI	Индекс рентабельности проекта	4.31
IRR	Внутренняя норма рентабельности	46,3%
$T_{ок.}$	Время окупаемости проекта	2.83 г.
$T_{ок\ прив.}$	Время окупаемости с учётом дисконтирования	4.185 г.
-	Минимальные среднегодовые поставки для окупаемости разработки	4.2

Полученные оценки говорят прежде всего о том, что проект обладает огромной нормой доходности. Причиной этому, прежде всего, является относительно низкая себестоимость датчиков, которая позволяет продавать

системы с большой наценкой при сохранении конкурентоспособности. Затраты на разработку системы окупятся в том случае, если после эксплуатационного тестирования удастся дополнительно продавать хотя бы 4.2 системы в год. По оценкам руководства фирмы, достижение такого объёма поставок возможно даже за счёт российского рынка (компания не ограничена одним поставщиком), поэтому риск неполучения прибыли практически отсутствует. К сожалению, в работе невозможно привести обоснование подобных оценок.

Эффективность внедрения системы управления проектами

Внедрение систем управления проектами в ООО “Комплекс-Ресурс” несомненно принесло экономические результаты, так как повысилась эффективность разработки системы, т.е. уменьшились затраты на разработку при допущении, что в случае без внедрения систем проект дошёл бы до аналогичного состояния. Тем не менее, тяжело получить численные оценки вклада. Существуют различные методы оценки, но большинство экспертов сходятся к тому, что использование информационных систем повышает эффективность на 20-50% в зависимости от квалификации разработчиков [11].

Предположим, что эффективность разработки после внедрения информационных систем возросла на 25%. При этом, на проведение обзоров и внедрение систем в первый год проекта было затрачено около 10% ресурсов, т.е. 0.125 единиц. На поддержку системы в последующие годы затрачена пренебрежительно малая сумма. На основании данных оценок произведён пересчёт показателей проекта, сравнение показателей для двух вариантов приведено в таблице 5.3.

По таблице видно, что индекс рентабельности проекта упал почти на 20%, а сроки окупаемости проекта приблизились к его планируемому времени жизни проекта. Несмотря на допущения, можно предположить, что правильная организация процесса разработки сильно улучшила эффективность проекта в целом.

Таблица 5.3

Показатели проекта “Канада” при отсутствии систем управления проектами

		Оценки	
Индекс	Полное название	С ИС	Без ИС
ARR	Бухгалтерская доходность проекта	121,6%	95.49%
NPV	Чистая приведённая стоимость	3.311 ед.	2.674 ед.
NFV	Чистая приведённая стоимость в будущем	5.835 ед.	4.713 ед.
PI	Индекс рентабельности проекта	4.310	3.674
IRR	Внутренняя норма рентабельности	46.3%	36.5%
$T_{ок.}$	Время окупаемости проекта	2.83 г.	3.19 г.
$T_{ок\ прив.}$	Время окупаемости с учётом дисконтирования	4.19 г.	4.96 г.
-	Минимальные среднегодовые поставки для окупаемости разработки	4.2	5.83

5.4. Итоги и перспективы дальнейшего развития системы

В настоящее время разработаны и реализованы все основные модули ИИС Neftemer, а также демонстрационные версии приложений для серверной и терминальной частей системы. Заказчику системы была сдана демонстрационная версия, которая в настоящий момент проходит эксплуатационное тестирование.

Разработка специализированных модулей была временно остановлена с целью фокусировки ресурсов на средствах хранения и обработки данных, разработка которых была поручена другим разработчикам. Данная ситуация вполне приемлема для демонстрационных версий системы, но в случае промышленного применения ИИС Neftemer потребуется вернуться к доработка заложенных в архитектуру компонентов системы. К сожалению, это будет происходить без участия автора дипломного проекта, так как в настоящее время он перешёл на другое место работы.

Тем не менее, есть основания полагать, что система будет и дальше успешно развиваться и без участия автора проекта, так как на модули разработана достаточно полная программная документация, а дальнейшая разработка требует в основном реализации отдельных программных модулей по готовым спецификациям. При этом, у будущих разработчиков будут в

распоряжении библиотеки, которые реализуют внутренние протоколы, архитектуру программных модулей, средства диагностики и многое другое. Потребуется лишь дополнить необходимую функциональность.

Ещё одним направлением развития системы является поиск новых заказчиков. Для этого придётся вести разработку дополнительных модулей и конфигураций. Поскольку система изначально проектировалась как расширяемая, то подобные модификации не должны вызвать трудностей.

В любом случае, ИС Neftemer будет привлекательной для любого заказчика, так как её стоимость в несколько раз ниже существующих аналогов, а использование неинвазивных блоков детектирования позволяет существенно снизить затраты на установку и при необходимости обеспечить мобильность оборудования. Для успеха подобной системе требуется лишь обеспечить надёжные поставки блоков детектирования и предоставить информационную систему, которая соответствует требованиям заказчика. Текущая система вполне соответствует подобным требованиям, и в дальнейшие задачи ООО “Комплекс-Ресурс” входит сопровождение и модификация системы в соответствии с изменением потребностей заказчика.

Заключение

В дипломном проекте было рассказано лишь о малой части задач, которые были доверены автору в рамках проекта “Канада”. В процессе работы была построена сложная информационная система (ИС Neftemer), в задачи которой входили сбор, обработку, хранение и предоставление показателей по потокам нефти, что необходимо для эффективной работы нефтедобывающего предприятия.

В СПБГПУ автор проходил параллельное обучение по двум специальностям “Прикладная информатика в экономике” и “Управление в технических системах”. В ходе работы были использованы навыки, полученные при обучении по обеим специальностям, была подтверждена квалификация автора в областях системной архитектуры, архитектуры информационных систем, разработки программного обеспечения. В течение проекта автор участвовал в следующих задачах:

- организации и планировании проекта “Канада”;
- разработке требований и спецификаций к системе;
- внедрении систем поддержки совместной разработки;
- разработке программного и аппаратного обеспечения;
- сдаче системы и разработке программной документации;
- поддержке пользователей и сопровождении системы;
- управлении проектам.

Таким образом, за время работы в ООО “Комплекс-Ресурс” автор участвовал во всех основных процессах жизненного цикла информационной системы, что является весьма ценным опытом. В дипломный проект вошли такие части, как разработка общей архитектуры информационной системы, модулей интеграции разработанной ИИС Neftemer с экономическими информационными системами заказчика, а также внедрение систем совместной разработки в ООО “Комплекс-Ресурс”.

На начале своей работы автор участвовал в организации проекта “Канада”: анализе поставленных требований, планировании, формировании ресурсов, подборе команды разработчиков и организации процесса разработки. Конечно, конечные решения принимались руководителем проекта, но по многим вопросам мнение автора было учтено. Таким образом, нашли применение навыки по управлению проектами, которые были получены в процессе обучения в СПБГПУ по обеим специальностям.

Внедрение систем поддержки совместной разработки позволило взглянуть на ЭИС с иной стороны, чем они рассматривались во время обучения. Фактически, была внедрена система, функциональность которой можно встретить во многих ERP-системах: управление задачами, планирование ресурсов, систему управления взаимоотношениями с клиентами (CRM) и систему управления информационными ресурсами предприятия (ECM).

При разработке программного обеспечения были использованы современные средства и методологии разработки, а также общепринятые типовые решения (базы данных, открытые библиотеки, стандартизованные протоколы обмена), что обеспечило высокое качество разработки при невысоких затратах. Одним из основных достижений автор считает разработку архитектуры и спецификации на систему при помощи UML и ICONIX, но из-за соглашения о неразглашении информации привести сколь либо подробную информацию по этим задачам невозможно.

Надо отметить, что в рамках работы решались не только инженерные, но и исследовательские задачи. Однако данные задачи относились к тематики, не связанной с информационными системами в экономике и менеджменте, и в дипломе они рассмотрены не были.

Демонстрационная версия ИИС Neftemer была успешно завершена и сдана заказчику, где прошла ряд испытаний, которые подтвердили её соответствие поставленным требованиям. Пользуясь документацией на систему и минимальной поддержкой, заказчик смог интегрировать систему со своими

ERP- и SCADA-системами. На момент выхода автора из проекта шла подготовка к эксплуатационному тестированию системы, после чего была бы продолжена разработка полнофункциональной версии системы.

Если потенциальный заказчик системы примет решения о заказе ожидаемого числа измерительных комплексов, то, в соответствии с оценками из пятой главы, рентабельность проекта “Канада” окажется гораздо выше внутренней нормы доходности ООО “Комплекс-Ресурс”. Но даже в том случае, если заказчик полностью откажется от закупки систем, разработанная система позволит получить преимущество при работе с другими заказчиками (в том числе и российскими). Поэтому, можно сделать вывод, что разработанная система с большой вероятностью окупит вложенные в неё средства. Несомненно, подобный результат во многом является заслугой разработчиков ИС Neftemer.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. Васильев Н.Н. Обзор систем организации общей работы. Внутренняя документация ООО Комплекс-Ресурс. ООО «Комплекс-Ресурс», 2009.

2. Вассоевич Н.В. Нефть. // Большая Советская Энциклопедия. 1970. № 13.

3. Колесов А. Введение в методологию Microsoft Solutions Framework. // BYTE/Россия. 2004. С. 46-51.

4. Кон М. Scrum: гибкая разработка ПО. Санкт-Петербург: Вильямс, 2011. 576 с.

5. Прошин Д.И. Проблемы выбора инструментальных средств построения SCADA-систем. // Информатизация и Системы Управления в Промышленности. 2010.

6. Скопин И.Н. Основы менеджмента программных проектов. Москва: Интернет-университет информационных технологий, 2004. 336 с.

7. Царёв В.В. Оценка экономической эффективности инвестиций. Санкт-Петербург: Питер, 2004. 464 с.

8. Щелкачев В.Н. Важнейшие принципы нефтеразработки. 75 лет опыта. Москва: Издательство РГУ Нефти и Газа им.Губкина, 2004. 608 с.

9. Юрьев В.Н. Информационные системы в экономике: Учебник. Санкт-Петербург: Изд-во Политехн. ун-та,, 2006. 538 с.

10. Bailey D. Practical SCADA for industry. London: Elsevier, 2003. 320 c.

11. Black S. How to Compete: The Impact of Workplace Practices and Information Technology on Productivity. // The Review of Economics and Statistics. 2001. C. 434-445.

12. Cabot J. Tools for Teams: A Survey of Web-Based Software Project Portals. Toronto: University of Toronto, 2009. 141 c.

13. Kratirov V. Neftemer – A Versatile And Cost Effective Multiphase Meter. Saint Andrews: 24th North Sea Flow Measurement Workshop, 2006.

14. Madisetti V.K. Reengineering legacy embedded systems. // Design & Test of Computers, IEEE. 2002. № 16. С. 38–47.

15. Postel J. Internet Protocol - DARPA Internet Program Protocol Specification. RFC 791. Los Angeles: USC/Information Sciences Institute, 1981.

16. Project Management Institute. A guide to the project management body of knowledge (PMBOK® Guide). Newtown Square Pa.: Project Management Institute, 2008. Вып. 4th ed.

17. Rosenberg D. Agile development with ICONIX process : people, process, and pragmatism. Berkeley CA: Apress, 2005. 624 c.

18. Scheers L. Multiphase Flow Metering Per Well – Can it be Justified? Saint Andrews, 2002.

19. Whitehead J. WebDAV: IETF Standard for Collaborative Authoring on the Web. // IEEE Internet Computing. 1998. С. 34-40.

20. Граничин О.Н. INTUIT.ru: Курс: Информационные системы предприятия. Лекция №11: Информационные системы планирования ресурсов и управления предприятием: ERP-системы. [Электронный ресурс]. URL: http://www.intuit.ru/department/itmngt/itmangt/11/1.html (просмотрено: 31.10.2011).

21. Ненашев О.В. Проект по разработке средства реинжиниринга устройства. [Электронный ресурс]. URL: https://nenhome-apps.sourcerepo.com/redmine /nenhome/projects/vhdlreengineering/wiki (просмотрено: 13.03.2011).

22. Орлик С. Проектирование программного обеспечения. Основы Программной Инженерии (по SWEBOK). [Электронный ресурс]. URL: http://swebok.sorlik.ru/2_software_design.html.

23. Шевырев В. Состав нефти и её классификация. [Электронный ресурс]. URL: http://www.mnpu.ru/?menu=docs&sub=svojstva (просмотрено: 31.10.2011).

24. Bies L. Modbus protocol, specifications and in depth tutorial. [Электронный ресурс]. URL: http://www.lammertbies.nl/comm/info/modbus.html (просмотрено: 30.10.2011).

25. EEE Group. Slug controller scoops top innovation award - News. [Электронный ресурс]. URL: http://www.eeegr.com/news/info.php?refnum=2272&startnum= (просмотрено: 13.03.2011).

26. Hosting Playground Inc. SourceRepo. Main page. [Электронный ресурс]. URL: http://sourcerepo.com/ (просмотрено: 30.10.2011).

27. Lang J.-P. Redmine Documentation. [Электронный ресурс]. URL: http://www.redmine.org/projects/redmine/wiki (просмотрено: 13.03.2011).

28. Neftemer Ltd. Neftemer's Worldwide Offices. [Электронный ресурс]. URL: http://neftemer.com/offices.html (просмотрено: 13.03.2011).

29. Neftemer Ltd. The Neftemer Multiphase Meter. [Электронный ресурс]. URL: http://neftemer.com/multiphase-meters.html (просмотрено: 16.10.2011).

30. Roy Rosenzweig Center for History and New Media. Zotero documentation. [Электронный ресурс]. URL: http://www.zotero.org/support/ (просмотрено: 15.10.2011).

31. Total Temperature Instrumentation Inc. Magmeters / Electromagnetic Flow Meters | Instrumart. [Электронный ресурс]. URL: http://www.instrumart.com/ ProductList.aspx?CategoryID=4139&source=flowmeters (просмотрено: 31.10.2011).

ПРИЛОЖЕНИЕ 1. КРАТКИЙ ОБЗОР ИСПОЛЬЗУЕМЫХ СРЕДСТВ И ТЕХНОЛОГИЙ

1. Система управления задачами Redmine

Redmine - это система управления проектами и отслеживания задач. Redmine написан на Ruby и представляет собой приложение на основе широко известного веб-фреймворка Ruby on Rails. Распространяется согласно GNU GPL, т.е. возможна его свободная модификация и доработка. Система включает следующие возможности:

- ведение множества проектов в рамках одной оболочки;
- гибкая система доступа, основанная на ролях;
- система отслеживания ошибок;
- диаграммы Ганта и календарь;
- ведение новостей проекта, документов и управление файлами;
- хранение документации;
- поддержка wiki-страниц, форумов и блогов;
- учёт временных затрат;
- настраиваемые произвольные поля для инцидентов, временных затрат, проектов и пользователей;
- интеграция с системами управления версиями (SVN, CVS, Git, Mercurial, Bazaar и Darcs);
- оповещение пользователей об изменениях с помощью RSS-потоков или электронной почты;
- возможность формирования отчётов в стандартных форматах практически по всем поддерживаемым областям (pdf, csv, rtf);
- защищённый доступ к системе;
- наличие большого числа плагинов.

Таким образом, система удовлетворяет всем поставленным требованиям за исключением интеграции с MS Office. В плане же разработки По система предоставляет полную и даже избыточную функциональность.

На следующих страницах приведены некоторые скриншоты окон из системы управления задачами Redmine, которая использовалась в рамках магистерского проекта Ненашева О.В. (см. [21]). Они не относятся к системе, используемой в Neftemer Ltd, и не отражают никакой конфиденциальной информации.

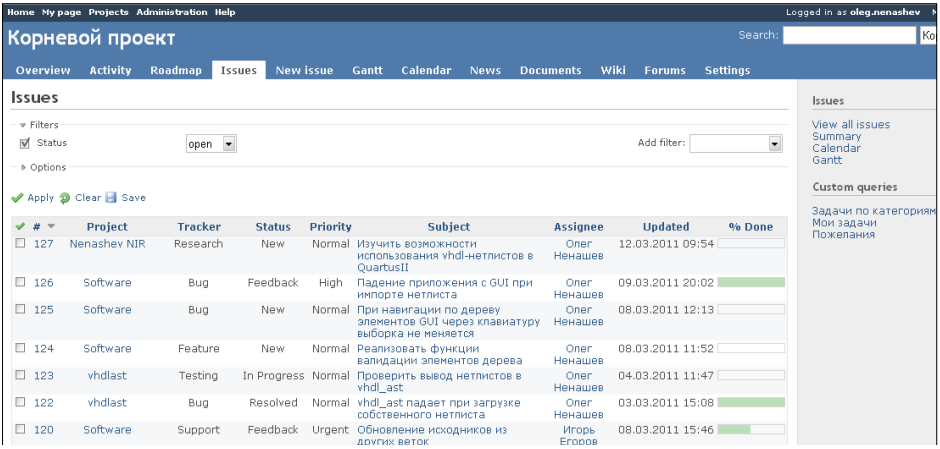

Рис. П1.1. Окно просмотра задач в трекере Redmine

Redmine поддерживает большое число wiki-страниц и позволяет их связывать с отдельными задачами, группами задач или друг с другом. При этом вся навигация легко осуществляется через браузер.

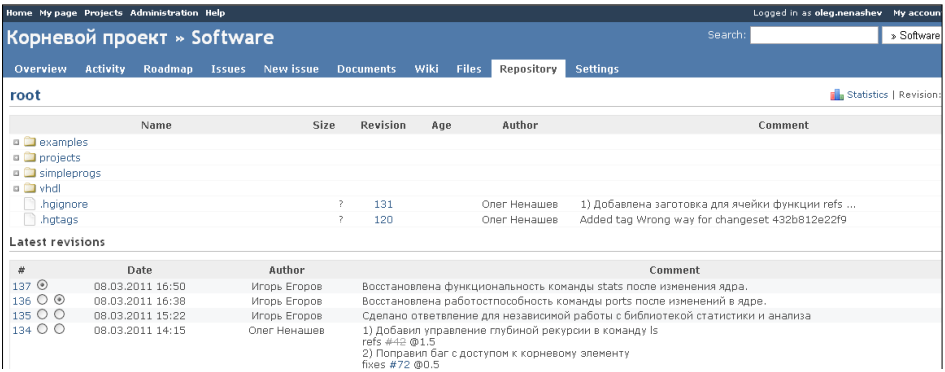

Рис. П1.2. Окно просмотра репозитория

Пользователь имеет возможность задания и сохранения параметров фильтрации задач. Таким образом, любой пользователь может адаптировать просмотрщик под свои нужды.

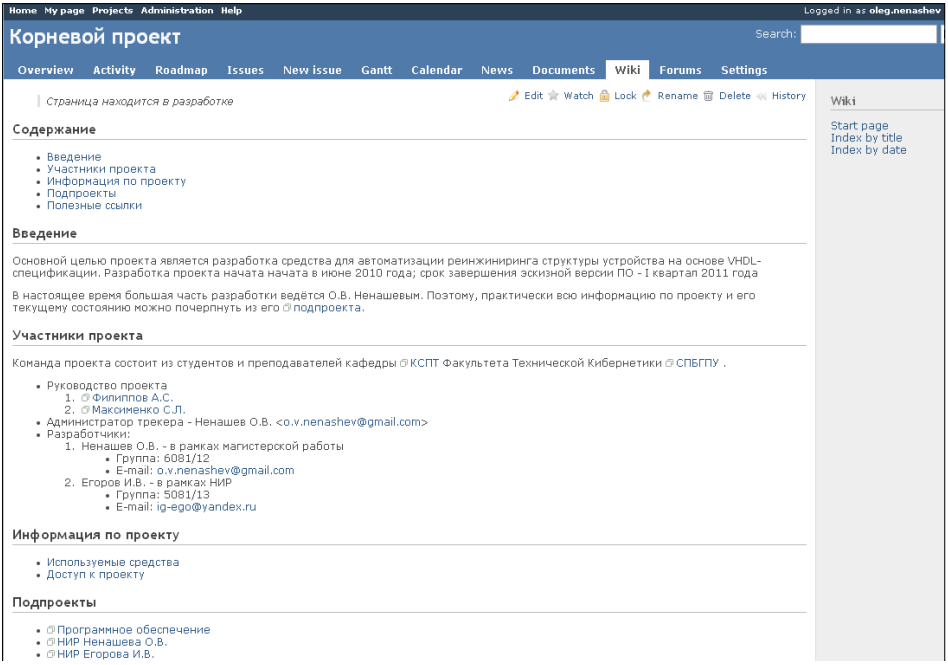

Рис. П1.3. Окно wiki-информации по проекту

Средство поддерживает автоматические ссылки на задачи и затраты времени при формировании версии исходного кода. При этом формируются обратные ссылки из задач. Также средство поддерживает сравнение различных версий кода.

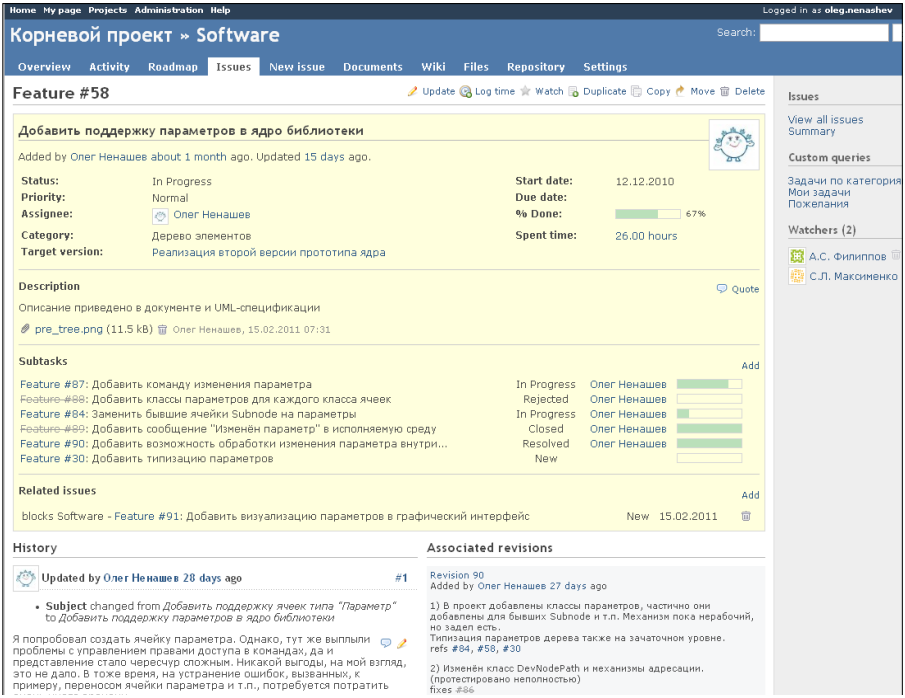

Рис. П1.4. Окно просмотра отдельной задачи

2. Средство ведения библиографии Zotero

Zotero - система управления библиографической информацией с открытым исходным кодом. Она разработана в американском Центре Истории и новых Носителей Данных имени Розенцвейга. Система представляет собой web-приложение, которое включает следующие возможности:

- Хранение информации на удалённом сервере (в т.ч. поддержка пользовательских хранилищ данных);
- Возможность организации совместной работы через механизмы групп с управлением правами доступа;
- Интеграция с текстовыми процессорами (MS Word, OpenOffice.org, TeX, и пр.)

- Синхронизация с сервером, перенос на другой компьютер, сохранение библиотеки на переносных носителях;
- Сохранение библиографической информации на лету с сайтов Google Scholar, Google Books, Amazon.com, ScienceDirect, Springerlink и др.
- Автоматическое извлечение информации об источнике из файлов форматов *.pdf, *.odt, *.doc(x);
- Автоматическая загрузка об по идентификаторам источниках по ISBN, ISSN, DOI;
- Поддержка пользовательских расширений.

В качестве примера, список использованных источников в данном дипломном проекте сформирован с помощью Zotero и его плагина к Microsoft Word.

ПРИЛОЖЕНИЕ 2. ПРИМЕРЫ ИСХОДНЫХ КОДОВ И ПРОГРАММНОЙ ДОКУМЕНТАЦИИ НА СИСТЕМУ

В соответствии с договором о неразглашении информации, невозможно предоставить полные исходные коды или описания из программной документации. Тем не менее, было получено разрешение на представление некоторых примеров, которые не имеют отношения к промышленной конфигурации системы или не несут угрозы информационной безопасности ООО “Комплекс-Ресурс”

1. Примеры конфигурационных файлов системы и их спецификаций

Ниже приведён пример реально используемого конфигурационного файла системы. Он используется для отладки работы модуля сбора данных с блока детектирования ИИС Neftemer.

В конфигурации подключаются модуль сбора данных и программный эмулятор блока детектирования, который связывается с БД через эмулятор последовательного интерфейса COM (он логически совместим с RS-485, который используется в демонстрационной версии системы).

Эмулятор БД не входит в основную систему, и при его разработке было принято решение не включать в данный тестовый модуль инфраструктуру самодиагностики. В связи с этим загрузчик ПО не может контролировать данный модуль, и он запускается как обычный исполняемы файл (тип NonSystem). Нарушения в его работе фиксируются с помощью лог-файлов, которые генерирует эмулятор.

Листинг П2.1. Пример конфигурационного файла ИС Neftemer

```
<?xml version="1.0" encoding="utf-8"?>
<!--Файл с описанием настроек тестовой конфигурации-->
<!--Файл конфигурации обрабатывается супервизором. Порядок запуска
программ определяется их порядком в файле-->
<!--Номер программы соответствует номеру в файле
xsi:noNamespaceSchemaLocation="NM_Configuration.xsd"-->
<Configuration  Name="Test frqm" Version="0"
      xsi:schemaLocation="NM ../xsd_include/NM_Configuration.xsd"
```

```
        xmlns="NM"
        xmlns:nm="NM"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xmlns:xs="http://www.w3.org/2001/XMLSchema">
 <annotation>
   <DevComment>Тестовая конфигурация для проверки модуля сбора
</DevComment>
 </annotation>
 <System>
   <Serial>0000ABCD</Serial>
   <NftAddress>1</NftAddress>
   <DevAddress>1</DevAddress>
   <RootDir>E:\Neftemer\Target</RootDir>
   <OS>Windows XP</OS>
 </System>
 <Programs>
   <Program Type="NonSystem">
     <Call>
       <Command>NM_MCBD03v15_Emulator.exe</Command>
       <Delay>100</Delay>
     </Call>
   </Program>
   <Program Type="Frequmeter" Number="100">
     <Call>
       <Command>Frequmeter_debug.exe</Command>
       <Delay>100</Delay>
     </Call>
     <Server>
       <Port>504</Port>
       <AcceptRemote>true</AcceptRemote>
       <Transmit>false</Transmit>
     </Server>
     <MCBD>
       <Interface>
         <COM-Port>2</COM-Port>
         <Baudrate>19200</Baudrate>
         <Timeout>100</Timeout>
       </Interface>
       <BD>
         <HardwareVersion>BD-03</HardwareVersion>
         <SoftwareVersion>15</SoftwareVersion>
         <ID>2</ID>
         <Mode>0</Mode>
```

```
        <CountPeriod>4</CountPeriod>
        <Diffcount>false</Diffcount>
      </BD>
    <Operation>
      <!--  NDA. Не разглашается -->
     </Operation>
    </MCBD>
    <Output useBinary="false">
     <Directory>results</Directory>
    </Output>
  </Program>
 </Programs>
</Configuration>
```

Листинг П2.2. Фрагмент информационного файла системы

```
<?xml version="1.0" encoding="UTF-8"?>

…

 <System>
  <Events>
   <Event Type="Message" Code="0">
    <annotation>
     <Summary>TODO</Summary>
     <DevComment>Сообщения передают информацию об штатном изменении
состояния системы</DevComment>
    </annotation>
   </Event>
   <Event Type="Warning" Code="1" >
    <annotation>
            <DevComment>Исключительные ситуации, не несущие опасности
для системы</DevComment>
    </annotation>
   </Event>
   <Event Code="2" Type="Error">
    <annotation>
     <Summary>TODO</Summary>
     <DevComment>Ошибки и сбои, нарушающие работу
системы</DevComment>
    </annotation>

….
```

```
   <Program Type="CC_Server">
     <annotation>
       <Summary>Здесь будет краткая информация о программе (для
всплывающих подсказок и т.п.)</Summary>
       <DevComment>Здесь перечислена информация о программе сервера
главного контроллера</DevComment>
     </annotation>
     <!--Идентификатор 0 относится к самой программе (если сообщение не
назначить модулю)-->
     <Module ID="0" Name="CC Server">
       <annotation>
         <Summary>CC Server programm</Summary>
         <DevComment>Модуль для хранения сообщений о самой
программе</DevComment>
       </annotation>
       <Events></Events>
     </Module>

…

  </Configurations>
</SystemDescription>
```

Листинг П2.3. Фрагмент xsd-спецификации на информационный файл системы

```
<xs:schema targetNamespace="NM" xmlns="NM" xmlns:nm="NM"
xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified"
attributeFormDefault="unqualified">

…

    <xs:annotation>
      <xs:documentation>Корневой элемент информационного файла
системы</xs:documentation>
    </xs:annotation>
  </xs:element>
  <xs:complexType name="SysDescription">
    <xs:annotation>
      <xs:documentation>Структура с полным описанием
системмы</xs:documentation>
    </xs:annotation>
    <xs:sequence>
      <xs:element name="System" type="nm:SysInfo">
        <xs:annotation>
```

```
        <xs:documentation>Общее описание системы</xs:documentation>
      </xs:annotation>
      <xs:key name="EventKey">
        <xs:selector xpath=".//nm:Event" />
        <xs:field xpath="@Type" />
      </xs:key>
….
  <xs:complexType name="ProgramInfo">
    <xs:annotation>
      <xs:documentation>Информация о программе</xs:documentation>
      <xs:appinfo>Включает комментарии и список внутренних модулей с
информацией о них</xs:appinfo>
    </xs:annotation>
    <xs:complexContent>
      <xs:extension base="nm:ItemInfo">
        <xs:sequence>
          <xs:element maxOccurs="32" minOccurs="0" name="Module"
type="nm:ProgramModuleInfo" />
        </xs:sequence>
        <xs:attribute name="Type" type="nm:ProgramType" />
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
</xs:schema>
```

2. Примеры исходных кодов на C++

Ниже приведены примеры исходных кодов двух составляющих исходных кодов: алгоритма взаимодействия с БД в рамках ИС Neftemer (с контролем данных и диагностикой) и пример заголовочного файла шаблонного буфера данных системы, который используется во всех модулях системы (в том числе и в сервере данных реального времени). К сожалению, невозможно привести полные исходные коды, но данные фрагменты демонстрируют его внутреннюю архитектуру и средства самодокументирования исходного кода. Используемый формат описаний поддерживается генератором программной документации Doxygen.

Листинг П2.4. Фрагмент программного кода, реализующего алгоритм надёжного взаимодействия с блоками детектирования ИС Neftemer

```
#include "MCBD.h"
```

```
/// Поток опроса МКБД
///
/// \param[in] args Указатель на модуль, для которого был создан поток
unsigned _stdcall MCBD::Comm_thread(void *args)
{
	MCBD::OperResult res;
	MCBD::OperResult opres;
	long test;

	// Получение указателя на модуль опроса
	MCBD_Module* BD=(MCBD_Module*) args;

	// Цикл взаимодействия с МКБД
	while (true)
	{
		/// Обработка общих флагов управления
		if (BD->f_CriticalError) return 0;

		// Остановка модуля
		if (BD->m_Flags.f_stop_Module)
		{
		   BD->m_Flags.f_stop_Module=false;
		   BD->Change_State(MCBD::DISABLED,"Module stopped");
		   return 0;
		}

		// Остановка взаимодействия с МКБД
		if (BD->m_Flags.f_stop_BD)
		{
		  BD->m_Flags.f_stop_BD=false;
		  BD->Change_State(MCBD::STOPPED,"BD Stopped");
		}

		// Проверка ошибок с переводом в BROKE при необходимости
		BD->Check_Errors();

		// Автомат состояний
		switch (BD->m_State)
		{
		// Состояние ожидания команды
		//// STOPPED ////
		case MCBD::STOPPED:
		  // Пришла команда начать измерения
```

```
                    if (BD->m_Flags.f_start_measurements)
                       BD->Change_State(MCBD::INIT,"Initialization command got");
                    else Sleep(1000);
                    break;

                  //// INIT ////
                  case MCBD::INIT:
                           opres=BD->Initialize_BD();
...
};
```

Листинг П2.5. Фрагмент заголовочного файла для циклического буфера данных с множественным доступом.

```
#ifndef MultiBuffer_H
#define MultiBuffer_H

#include "MultiBufferReader.h"
#include "MultiBufferWriter.h"

#define MAXCHANNELS 32

/// Элемент контейнера MultiBuffer
template <class T>
class MultiBufferElem
{
public:
        ULONGID;                ///< Идентификатор элемента
        T               elem;   ///< Хранимый элемент
};

/// \brief Класс-очередь с независимым считыванием по нескольким каналам
///
/// Буфер реализует хранилище данных определённого типа с возможностью
независимого
/// считывания из нескольких каналов. Сами каналы имеют функции для
фильтрации данных, например, по времени.
template <class T>
class MultiBuffer:public MultiBufferWriter<T>
{
        MultiBufferElem<T>*     m_Array;        ///< Массив элементов
```

```
        MultiBufferReader<T>*   m_Channels[MAXCHANNELS];  ///< Массив
считывателей каналов

        int m_LowPtr;           ///< Указатель нижней границы данных
        int m_HighPtr;          ///< Указатель верхней границы данных
        int m_Elem_numb;        ///< Число элементов в буфере

        ULONG m_NextID; ///< Указатель на идентификатор следующего
записываемого элемента
        ULONG m_LowID; ///< Указатель на минимальный идентификатор
элемента из хранимых в буфере
        int m_BufferSize;       ///< Максимальное число хранимых элементов

public:

…

}
```