Министерство образования и науки Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего профессионального образования
ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ СИСТЕМ УПРАВЛЕНИЯ И РАДИОЭЛЕКТРОНИКИ (ТУСУР)
Кафедра комплексной информационной безопасности электронно-вычислительных систем (КИБЭВС)

УТВЕРЖДАЮ
заведующий каф.КИБЭВС
_____ А.А. Шелупанов
"___"_____ 2013г.

КОМПЬЮТЕРНАЯ ЭКСПЕРТИЗА

Отчет по групповому проектному обучению
Группа КИБЭВС-1208

Ответственный исполнитель
Студент гр. 520-1
_____ Никифоров Д. С.
"___"_____ 2013г.

Научный руководитель
Аспирант каф.КИБЭВС
_____ Гуляев А. И.
"___"_____ 2013г.

Томск – 2013

РЕФЕРАТ

Курсовая работа содержит 45 страницы, 10 рисунков, 2 таблицы, 11 источников, 1 приложение.

КОМПЬЮТЕРНАЯ ЭКСПЕРТИЗА, ФОРЕНЗИКА, ЛОГИ, QT, XML, GIT, LATEX, ЖУРНАЛЬНЫЕ ФАЙЛЫ, SKYPE, PIDGIN, WINDOWS, СЕРТИФИКАЦИЯ.

Цель работы — создание автоматизированной системы, предназначенной для экспертизы образов жёстких дисков.

Задачей, поставленной на данный семестр, стало написание автоматизированного экспертизного комплекса, имеющего следующие возможности:

1) сбор и анализ событий системных журналов операционной системы;
2) сбор и анализ информации из журналов истории браузеров;
3) сбор и анализ истории переписки мессенджеров;
4) сбор и анализ событий журнальных файлов приложений;
5) обнаружение сетевых параметров системы;
6) поиск файлов по имени.

Результатами работы в данном семестре являются:

- разработка архитектуры проекта;
- использование в разработке системы контроля версий GIT;
- использование Qt — кроссплатформенной библиотеки C++;
- изучение вопроса сертификации для возможности внедрения данного комплекса в гос. структуры, занимающиеся информационной безопасностью;
- использование системы компьютерной вёрстки TEXдля написания документации.

Пояснительная записка выполнена в текстовом редакторе Vim.

Список исполнителей

Моргуненко А.В. – документатор.

Никифоров Д.С. – программист, ответственный исполнитель, ответственный за написание части системы, работающей с логами системы.

Поляков И.Ю. – программист, ответственный за написание части системы, работающей с логами мессенджеров.

Пономарёв А.К. – аналитик.

Содержание

1 Введение

Компьютерно-техническая экспертиза – это самостоятельный род судебных экспертиз, относящийся к классу инженерно-технических экспертиз, проводимых в следующих целях: определения статуса объекта как компьютерного средства, выявление и изучение его роли в рассматриваемом деле, а так же получения доступа к информации на электронных носителях с последующим всесторонним её исследованием [1]. Компьютерная экспертиза помогает получить доказательственную информацию и установить факты, имеющие значение для уголовных, гражданских и административных дел, сопряжённых с использованием компьютерных технологий. Для проведения компьютерных экспертиз необходима высокая квалификация экспертов, так как при изучении представленных носителей информации, попытке к ним доступа и сбора информации возможно внесение в информационную среду изменений или полная утрата важных данных.

Компьютерная экспертиза, в отличие от компьютерно-технической экспертизы, затрагивает только информационную составляющую, в то время как аппаратная часть и её связь с программной средой не рассматривается.

На протяжении предыдущих семестров нами были рассмотрены такие направления компьютерной экспертизы, как исследование файловых систем, сетевых протоколов, организация работы серверных систем, механизм журналирования событий. Также нами были изучены основные задачи, которые ставятся перед сотрудниками правоохранительных органов, которые проводят компьютерную экспертизу, и набор чуществующих утилит, способных помочь эксперту в проведении компьютерной экспертизы. Было выявлено, что существует множество разрозненных программ, предназначенных для просмотра лог-файлов системы и таких приложений, как мессенджеры и браузеры, но для каждого вида лог-файлов необходимо искать отдельную программу. Так как ни одна из них не позволяет эксперту собрать воедино и просмотреть все логи системы, браузеров и мессенджеров, было решено создать для этой цели собственный автоматизированный комплекс, которому на данный момент нет аналогов.

2 Назначение и область применения

Разрабатываемый комплекс предназначен для автоматизированного сбора информации из журналов операционных систем и приложений.

3 Постановка задачи

На данный семестр были поставлены следующие задачи:

- определить средства для разработки;
- разработать архитектуру проекта;
- реализовать структуру проекта;
- определить единый формат выходных данных;
- реализовать несколько модулей.

Задачи по проектированию модулей:

1) сбор и анализ истории переписки пользователей системы Windows из Pidgin и Skype;
2) написание автоматизированных модулей для сбора истории переписки пользователей системы Windows из таких программ как Pidgin и Skype;
3) сбор и анализ событий журнальных файлов Windows;
4) написание автоматизированного модуля для сбора событий журнальных файлов Windows.

4 Новые инструменты

4.1 Система контроля версий Git

Для разработки программного комплекса для проведения компьютерной экспертизы решено использовать Git.

Git — распределённая система управления версиями файлов. Проект был создан Линусом Торвальдсом для управления разработкой ядра Linux, как противоположность системе управления версиями Subversion (также известная как «SVN») [2].

Необходимость использования системы версий, очевидна. Так как в группе несколько программистов и тестер, мы имеем:

- возможность удаленной работы с исходными кодами;
- возможность создавать свои ветки, не мешая при этом другим разработчикам;
- доступ к последним изменениям в коде, т.к. все исходники хранятся на сервере git.keva.su ;
- исходные коды защищены, доступ к ним можно получить лишь имея RSA-ключ;
- возможность откатиться к любой стабильной стадии проекта.

Основные постулаты работы с кодом в системе Git:

- каждая задача решается в своей ветке;
- коммитим сразу, как что-то получили осмысленное;
- в master мержится не разработчиком, а вторым человеком, который производит вычитку и тестирование изменения;
- все коммиты должны быть осмысленно подписаны/прокомментированы.

Для работы над проектом нами был поднят собственный репозиторий на сервере git.keva.su. Адреса репозиториев следующие:

Исходные файлы проекта:

git clone git@git.keva.su:gpo.git gpo.git

Репозиторий для тестирования проекта:

git clone git@git.keva.su:gpo-testdata.git gpo-testdata.git

4.2 Система компьютерной вёрстки TEX

TEX — это созданная американским математиком и программистом Дональдом Кнутом система для вёрстки текстов. Сам по себе TEX представляет собой специализированный язык программирования.Каждая издательская система представляет собой пакет макроопределений этого языка.

LATEX — это созданная Лэсли Лэмпортом издательская система на базе TEX'a[3]. LATEX позволяет пользователю сконцентрировать свои услия на содержании и структуре текста, не заботясь о деталях его оформления.

Для подготовки отчётной и иной документации нами был выбран LATEX так как совместно с системой контроля версий Git он предоставляет возможность совместного создания и редактирования документов. Огромным достоинством системы LATEX то, что создаваемые с её помощью файлы обладают высокой степенью переносимости [4].

Совместно с LATEX часто используется BibTEX — программное обеспечение для создания форматированных списков библиографии. Оно входит в состав дистрибутива LATEX и позволяет создавать удобную, универсальную и долговечную библиографию. BibTEX стал одной из причин, по которой нами был выбран LATEX для создания документации.

4.3 Qt - кроссплатформенный инструментарий разработки ПО

Qt - это кроссплатформенная библиотека C++ классов для создания графических пользовательских интерфейсов (GUI) от фирмы TrollTech. Эта библиотека полностью объектно-ориентированная, что обеспечивает легкое расширение возможностей и создание новых компонентов. Ко всему прочему, она поддерживает огромнейшее количество платформ.

Qt позволяет запускать написанное с его помощью ПО в большинстве современных операционных систем путём простой компиляции программы для каждой ОС без изменения исходного кода. Включает в себя все основные классы, которые могут потребоваться при разработке прикладного программного обеспе-

чения, начиная от элементов графического интерфейса и заканчивая классами для работы с сетью, базами данных и XML. Qt является полностью объектно-ориентированным, легко расширяемым и поддерживающим технику компонентного программирования.

Список использованных классов фраемворка QT

- iostream
- QChar
- QCryptographicHash
- QDir
- QDirIterator
- QFile
- QFileInfo
- QIODevice
- QList
- QRegExp
- QString
- QTextStream
- QtSql/QSqlDatabase
- QVector
- QXmlStreamReader
- QXmlStreamWriter
- Conversations

Класс QXmlStreamWriter представляет собой XML писателя с простым потоковым.

Класс QXmlStreamReader представляет собой быстрый синтаксически корректный XML анализатор с простым потоковым API.

QVector представляет собой класс для создания динамических массивов.

Модуль QtSql/QSqlDatabase помогает обеспечить однородную интеграцию БД в ваши Qt приложения.

Класс QTextStream предоставляет удобный интерфейс для чтения и записи текста.

QTextStream может взаимодействовать с QIODevice, QByteArray или QString. Используя потоковые операторы QTextStream, вы можете легко читать и записывать слова, строки и числа. При формировании текста QTextStream поддерживает параметры форматирования для заполнения и выравнивания полей и форматирования чисел. [5]

Класс QString предоставляет строку символов Unicode.

QString хранит строку 16-битных QChar, где каждому QChar соответствует один символ Unicode 4.0. (Символы Unicode со значениями кодов больше 65535 хранятся с использованием суррогатных пар, т.е. двух последовательных QChar.)

Unicode - это международный стандарт, который поддерживает большинство использующихся сегодня систем письменности. Это расширение US-ASCII (ANSI X3.4-1986) и Latin-1 (ISO 8859-1), где все символы US-ASCII/Latin-1 доступны на позициях с тем же кодом.

Внутри QString использует неявное совместное использование данных (копирова при-записи), чтобы уменьшить использование памяти и избежать ненужного копирования данных. Это также позволяет снизить накладные расходы, свойственные хранению 16-битных символов вместо 8-битных.

В дополнение к QString Qt также предоставляет класс QByteArray для хранения сырых байт и традиционных нультерминальных строк. В большинстве случаев QString - необходимый для использования класс. Он используется во всем API Qt, а поддержка Unicode гарантирует, что ваши приложения можно будет легко перевести на другой язык, если в какой-то момент вы захотите увеличить их рынок распространения. Два основных случая, когда уместно использование QByteArray: когда вам необходимо хранить сырые двоичные данные и когда критично использование памяти (например, в Qt для встраиваемых Linux-систем).[6]

Класс QRegExp предоставляет сопоставление с образцом при помощи регулярных выражений.

Регулярное выражение, или ”regexp”, представляет собой образец для поиска соответствующей подстроки в тексте. Это полезно во многих ситуациях, например:

Проверка правильности – регулярное выражение может проверить, соответствует ли подстрока каким-либо критериям, например, целое ли она число или не содержит ли пробелов. Поиск – регулярное выражение предоставляет более мощные шаблоны, чем простое соответствие строки, например, соответствие одному из слов mail, letter или correspondence, но не словам email, mailman, mailer, letterbox и т.д. Поиск и замена – регулярное выражение может заменить все вхождения подстроки другой подстрокой, например, заменить все вхождения & на &, исключая случаи, когда за & уже следует amp;. Разделение строки – регулярное выражение может быть использовано для определения того, где строка должна быть разделена на части, например, разделяя строку по символам табуляции.

QFileInfo - Во время поиска возвращает полную информацию о файле.

Класс QDir обеспечивает доступ к структуре каталогов и их содержимого.

QIODevice представляет собой базовый класс всех устройств ввода/вывода в Qt.

Класс QCryptographicHash предоставляет способ генерации криптографических хэшей. QCryptographicHash могут быть использованы для генерации криптографических хэшей двоичных или текстовых данных.В настоящее время MD4, MD5, и SHA-1 поддерживаются.[6]

QChar обеспечивает поддержку 16-битных символов Unicode.

4.3.1 Автоматизация поиска журнальных файлов

Для сканирования образа на наличие интересующих лог файлов использовался класс QDirIterator. После вызова происходит поочередный обход по каждому файлу в директории и поддиректории. Проверка полученного полного пути

к файлу осуществляется регулярным выражением, если условие выполняется, происходит добавление в список обрабатываемых файлов.

4.3.2 Реализация сохранения результатов работы программного комплекса в XML

Сохранение полученных данных происходит в ранее выбранный формат XML(Extensible Markup Language). Для этого используется класс QXmlStreamReader и QxmlStreamWriter. Класс QXmlStreamWriter представляет XML писателя с простым потоковым API.

QXmlStreamWriter работает в связке с QXmlStreamReader для записи XML. Как и связанный класс, он работает с QIODevice, определённым с помощью setDevice().

Сохранение данных реализованно в классе WriteMessage. В методе WriteMessage структура которого представлена на UML диаграмме в разделе Архитектура.

5 Технические характеристики

5.1 Требования к аппаратному обеспечению

Минимальные системные требования:

- процессор 1ГГц Pentium 4;
- оперативная память 512 Мб;
- место на жёстком диске – 9 Гб.

5.2 Требования к программному обеспечению

Для корректной работы разрабатываемого программного комплекса на компьютере должна быть установлена операционная система ubuntu 12.04 или выше, данная система должна иметь набор библиотек QT.

5.3 Выбор единого формата выходных файлов

Для вывода результата был выбран формат XML-документов, так как с данным форматом лего работать при помощи программ, а результат работы данного комплекса в дальнейшем планируется обрабатывать при помощи программ.

XML - eXtensible Markup Language или расширяемый язык разметки. Язык XML представляет собой простой и гибкий текстовый формат, подходящий в качестве основы для создания новых языков разметки, которые могут использоваться в публикации документов и обмене данными [7]. Задумка языка в том, что он позволяет дополнять данные метаданными, которые разделяют документ на объекты с атрибутами. Это позволяет упростить программную обработку документов, так как структурирует информацию.

Простейший XML-документ может выглядеть так:

<?xml version="1.0"?>
<list_of_items>
<item id="1"\><first/>Первый</item\>
<item id="2"\>Второй <subsub_item\>подпункт 1</subsub_item\></item\>
<item id="3"\>Третий</item\>
<item id="4"\><last/\>Последний</item\>
</list_of_items\>

Первая строка - это объявление начала XML-документа, дальше идут элементы документа <list_of_items> - тег описывающий начало элемента list_of_items, </list_of_items> - тег конца элемента. Между этими тегами заключается описание элемента, которое может содержать текстовую информацию или другие элементы (как в нашем примере). Внутри тега начала элемента так же могут указывать атрибуты элемента, как например атрибут id элемента item, атрибуту должно быть присвоено определенное значение.

6 Разработка программного обеспечения

6.1 Архитектура

6.2 Основной алгоритм

В ходе разарботки был применен видоизменнённый шаблон проектирования Factory method.

Данный шаблон относится к классу порождающих шаблонов. Шаблоны данного класса - это шаблоны проектирования, которые абстрагируют процесс инстанцирования (создания экземпляра класса). Они позволяют сделать систему независимой от способа создания, композиции и представления объектов. Шаблон, порождающий классы, использует наследование, чтобы изменять инстанцируемый класс, а шаблон, порождающий объекты, делегирует инстанцирование другому объекту. Пример организации проекта при использовании шаблона проектирования Factory method представлен на рисунке 6.1.

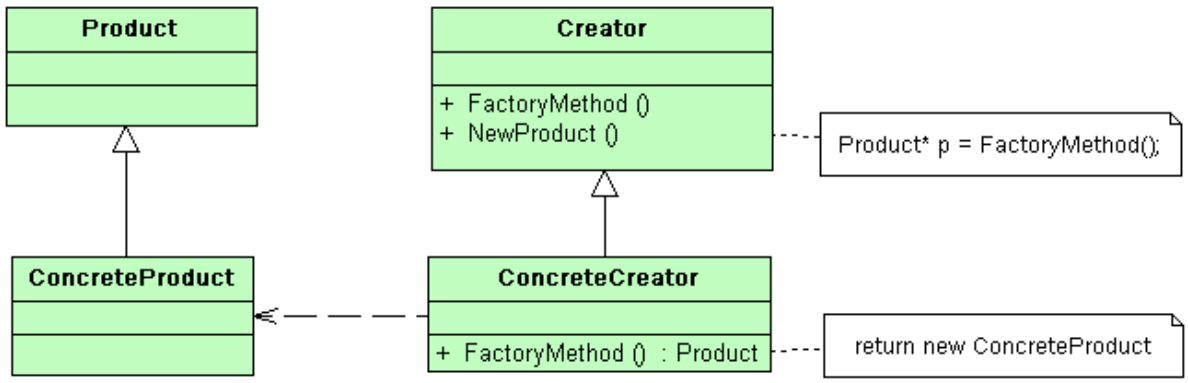

Рисунок 6.1 – Пример организации проекта при использовании шаблона проектирования Factory method

Использование данного шаблона позволило нам разбить наш проект на независимые модули, что весьма упростило задачу разработки, так как написание алгоритма для конкретного таска не влияло на остальную часть проекта. При разработке был реализован базовый класс для работы с образом диска. Данный клас предназначался для формирования списка настроек, определения операци-

онной системы на смонтированном образе и инстанционировании и накапливание всех необходимых классов-тасков в очереди тасков. После чего каждый таск из очереди отправлялся на выполнение. Блоксхема работы алгоритма представлена на рисунке 6.2.

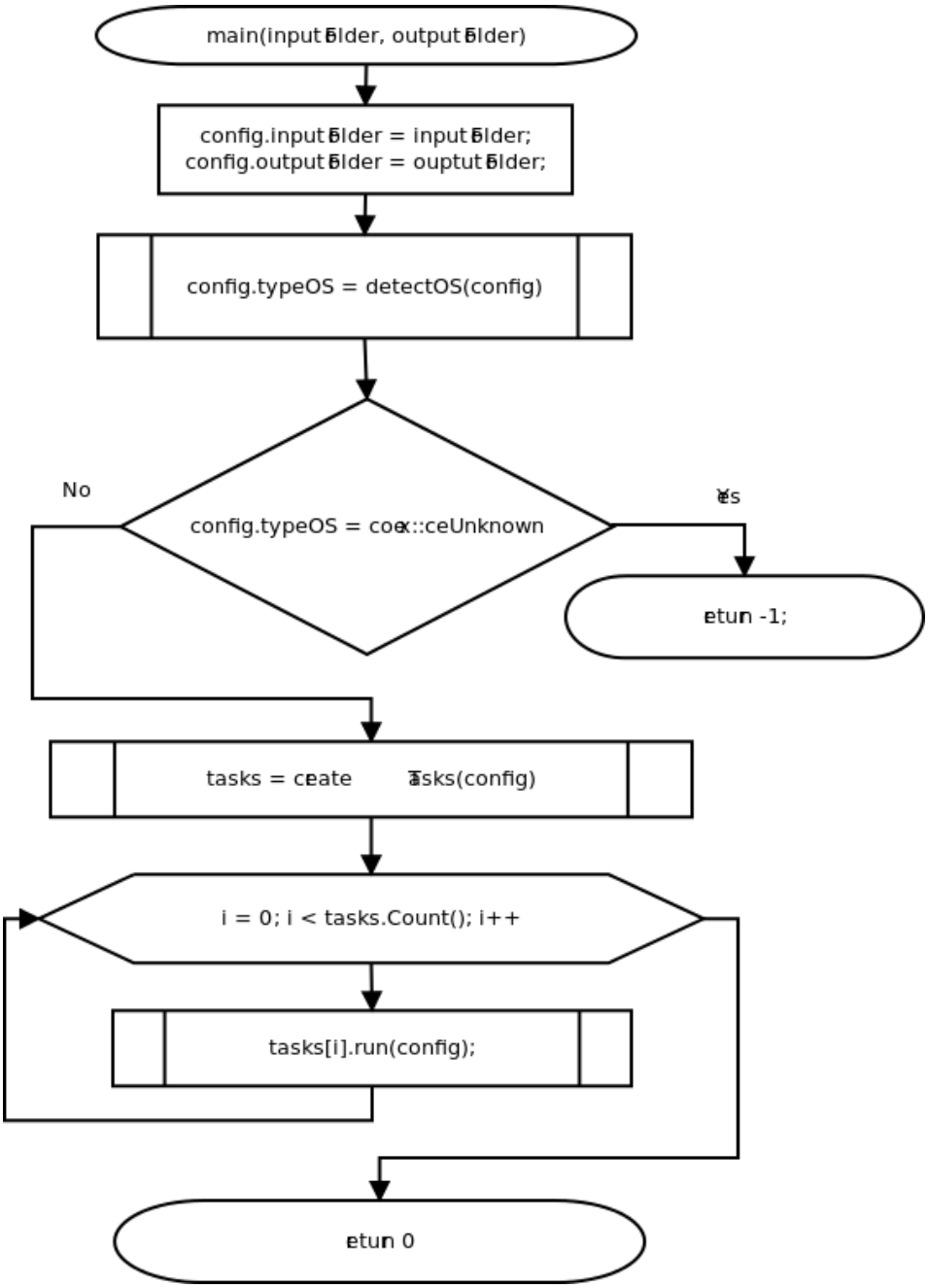

Рисунок 6.2 – Алгоритм работы с образом диска

Каждый класс-таск порождался путем наследования от базового абстрактного класса который имеет 8 методов и 3 атрибута:

1) QString manual() - возвращает справку о входных параметрах данного таска;

2) void setOption(QStringList list) - установка флагов для поданных на вход параметров;

3) QString command() - возвращает команду для инициализации таска вручную;

4) bool supportOS(const coex::typeOS &os) - возвращает флаг, указывающий на возможность использования данного таска для конкретной операционной системы;

5) QString name() - возвращает имя данного таска;

6) QString description() - возвращает краткое описание таска;

7) bool test() - предназначена для теста на доступность таска;

8) bool execute(const coex::config &config) - запуск таска на выполнение;

9) QString m_strName - хранит имя таска;

10) QString m_strDescription - хранит описание таска;

11) bool m_bDebug - флаг для параметра –debug;

На данный момент в проекте используется восемь классов. UML-диаграмма классов представлена на рисунке 6.3.

Классы taskSearchSyslogsWin, taskSearchPidginWin и taskSearchSkypeWin - наследники от класса task являются тасками. Класс winEventLog и _EVENTLOGRECORD предназначины для конвертации журнальных файлов операционной системы Windows XP, а класс writerMessages для преобразования истории переписки.

6.2.1 Описание основных функций типичного модуля системы

Данный модуль является классом-наследником от некоторого абстрактного класса используемого как основу для всех модулей программы (шаблон проектирования Factory method). Модуль содержит в себе 8 методов и 3 атрибута:

QString manual() - возвращает справку о входных параметрах данного таска

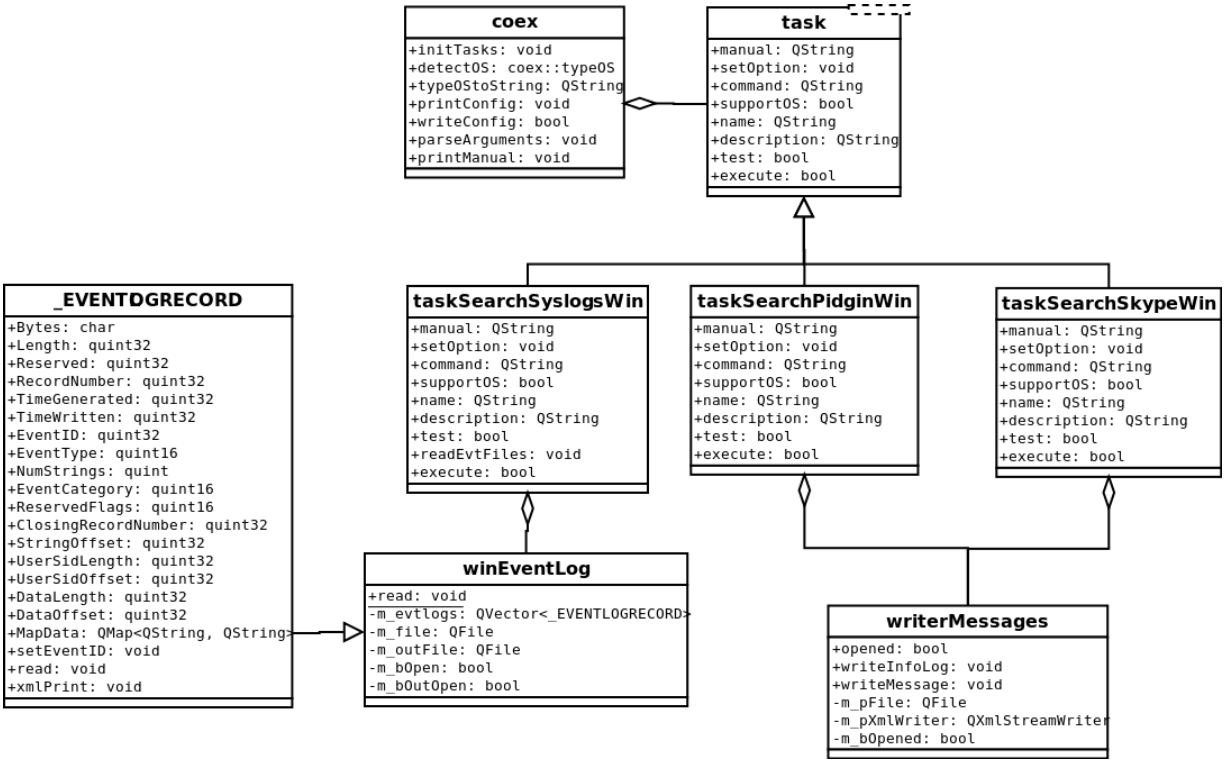

Рисунок 6.3 – UML-диаграмма классов

void setOption(QStringList list) - установка флагов для поданных на вход параметров

QString command() - возвращает команду для инициализации такска вручную

bool supportOS(const coex::typeOS &os) - возвращает флаг указывающий на возможность использования данного таска для конкретной операционной системы

QString name() - возвращает имя данного таска

QString description() - возвращает краткое описание такска

bool test() - предназначена для проверки работоспособности таска

bool execute(const coex::config &config) - запуск таска на выполнение

QString m_strName - хранит имя таска

QString m_strDescription - хранит описание таска

bool m_bDebug - флаг для параметра –debug

6.3 Сбор и анализ системных журналов Windows

Анализ журналов операционных систем может помочь при решении многих задач. Примером таких задач может быть попытка восстановления системы после поломок, поиск причин неполадок системы, просмотр журналов с целью выявления активности приложений в определенные периоды и т.д. и т.п. Также, анализ журнальных файлов является неотъемлемой частью компьютерно-технических экспертиз.

При исследовании компьютеров задача анализа журнальных файлов ставится на этапе «выявление и изучение его роли в рассматриваемом деле», так как именно в журнальных файлах операционной системы хранится информация о действиях производимых на данном компьютере.

Для проведения компьютерных экспертиз существует множество специализированных программных средств, как платных так и свободно распространяемых. Примером такого средства является «OSForensics». Данный инструмент позволяет проводить множество различных исследований таких как просмотр содержимого оперативной памяти, поиск подозрительных файлов на жестком диске, составления списка программ установленных на исследуемом компьютере. Но бесплатная версия не предоставляет никаких средств по работе с журналами операционной системы. Данный факт подтолкнул к созданию собственного программного средства для проведения компьютерных экспертиз.

В данной работе описывается работа по созданию модуля для данного программного средства, позволяющего в автоматическом режиме находить, читать и конвертировать журнальные файлы операционной системы Windows XP в XML-документы, понятные для человека и легко обрабатываемые при помощи электронно-вычислительных средств.

6.3.1 Общие сведения о журнальных файлах

В операционной системе Windows XP по умолчанию есть четыре журнала:

1) Журнал приложений

2) Журнал безопасности

3) Журнал установки

4) Журнал системы

Каждый из этих журналов содержит определенный тип информации. Журналы приложений содержат информацию о запуске и остановке процессов приложений, изменении статуса каждого приложения, а также предупреждения и ошибки связанные с приложениями.

Журнал безопасности содержит информацию о входах в систему, и события связанные с безопасностью системы, например превышение количества попыток неправильно введенных подряд паролей.

Журнал установки содержит информацию об установке и обновлении компонентов системы.

Журнал системы хранит информацию о системных событиях. Например изменение схемы энергопотребления или различного рода предупреждения и ошибок.

Каждый журнал хранится в соответствующем файле с расширением .evt. Эти файлы хранятся в папке %windows%/system32/config. Эти файлы имеют специальную структуру.

6.3.2 Структура журнальных файлов операционной системы Windows XP

Все журнальные файлы операционной системы Windows XP имеют единую структуру. Файл представляет собой последовательность записей бинарных данных. Каждая запись – это структура имеющая семнадцать полей [8].

Первые 4 байта содержат длину события в байтах, после длинны идет 4 байтный системный код сообщения. Затем 4 байтовый номер записи, после него дата создания записи, время создания, идентификатор события, тип события и так далее. Ниже приведен список полей записи, предназначенной для считывания одного события из журнального файла.

– uint32 Length;

- uint32 Reserved;
- uint32 RecordNumber;
- uint32 TimeGenerated;
- uint32 TimeWritten;
- uint32 EventID;
- uint16 EventType;
- uint16 NumStrings;
- uint16 EventCategory;
- uint16 ReservedFlags;
- uint32 ClosingRecordNumber;
- uint32 StringOffset;
- uint32 UserSidLength;
- uint32 UserSidOffset;
- uint32 DataLength;
- uint32 DataOffset;
- byte[] Data.

Из сторонних источников [8] стало известно о пяти типах событий (поле EventType) (значение - тип события):

1) 0x0001 - Error event;
2) 0x0010 – Failure Audit event;
3) 0x0008 - Success Audit event;
4) 0x0004 - Information event;
5) 0x0002 - Warning event.

У поля EventID удалось определить четыре значения:

1) 0x00 – Success;
2) 0x01 – Informational;
3) 0x02 – Warning;
4) 0x03 – Error.

Среди множества полей записи события были выделены поля содержащие информацию о типе события, времени возникновения события и создания записи, пользователя от имени которого была сделана запись, а также поле Data - поле с бинарными данными, в которых записана подробная информация о событии.

6.3.3 Алгоритм работы модуля

Модуль выполняет две задачи: поиск журнальных файлов на образе диска и конвертацию каждого файла в XML-документ. Первая задача выполняется при помощи библиотек QDir и QDirIterator из Qt Framework.

QDir — библиотека позволяющая работать с конкретной директорией. Создав объект данного типа с указанием директории мы получим доступ к этой директории в программе и сможем работать в ней (просматривать содержимое; удалять, создавать или копировать файлы; создавать поддиректории). Данный объект так же поддерживать разные наборы фильтров выходных данных которые могут отсеивать ненужную информацию.

QDirIterator — библиотека, предназначенная для работы с файловой системой начиная с определенной директории как точки входа. Создав объект данного типа с указанием директории мы получим все пути которые существуют в файловой системе и начинаются с указанной директории. Данный объект поддерживает фильтрацию которая помогает выделять только необходимую информацию, исключая то, что нас не интересует, например можно вывести список только тех файлов, которые находятся в данной директории или поддиректориях, или исключить вывод символьных ссылок. Объекты данного типа используются для поиска файлов или папок на образе исследуемого диска.

Так же данные библиотеки позволяют создавать объект QFile, который позволяет работать с файлом, путь к которому передается как параметр при создании, данный объект позволяет получить базовую информацию о файле, такую как относительный или абсолютный путь до этого файла, размер файла, тип файла или его имя. Так же позволяет перемещать или копировать данный файл. Поиск работает по алгоритму представленному на рисунке 6.1.

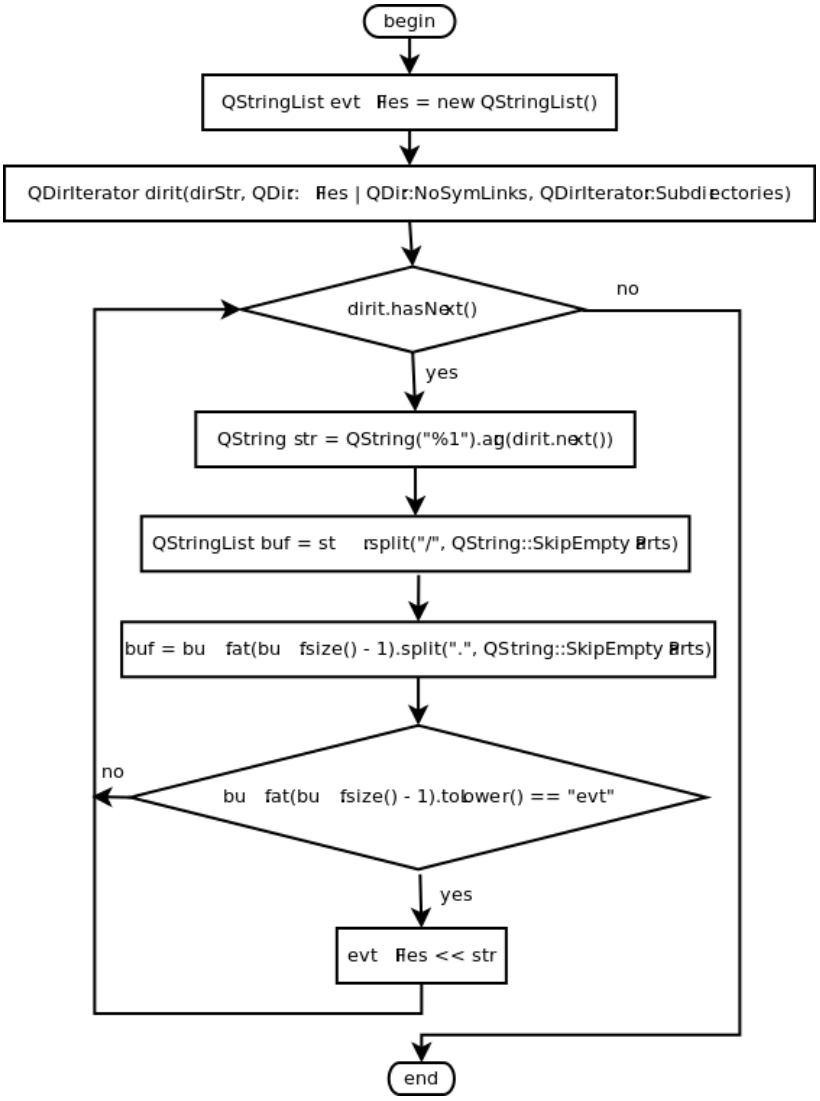

Рисунок 6.1 – Алгоритм поиска файлов с расширением *.evt

На выходе данного алгоритма мы получаем объект QStringList который содержит пути до всех найденных .evt файлов. Каждый экземпляр коллекции строк данного объекта передается в конструктор объекта winEventLog, который и конвертирует указанный файл в XML-документ. Алгоритм работы конвертора представлен на рисунке 6.2.

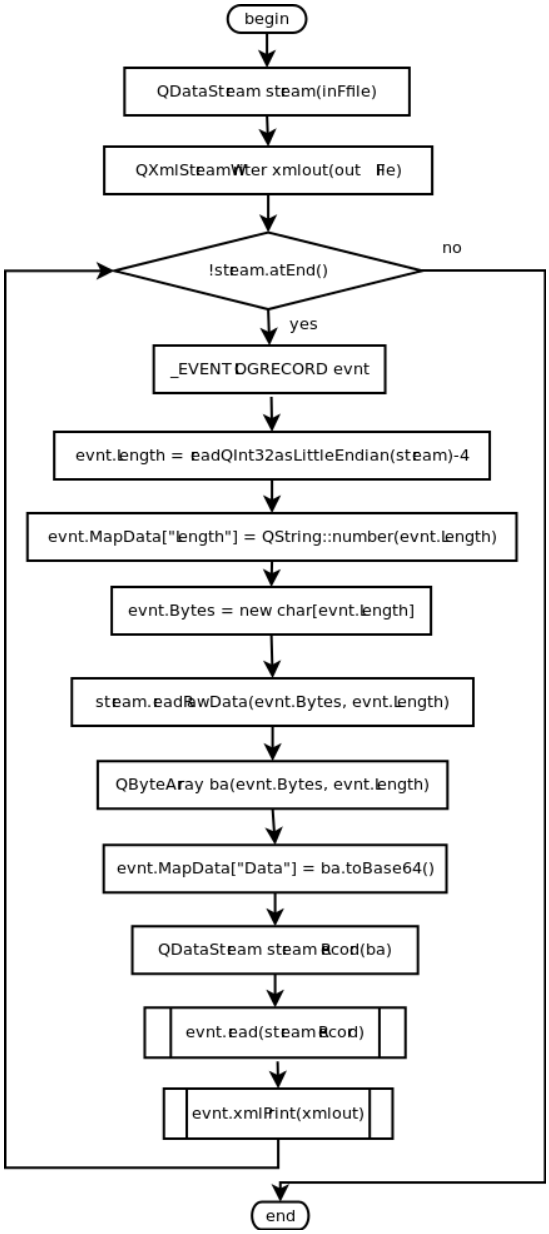

Рисунок 6.2 – Алгоритм конвертирования файлов *.evt в формат XML

6.4 Сбор и анализ истории переписки мессенджеров

Опишем задачу более подробно, разделив её на подзадачи:

1) определить места хранения файлов с личной перепиской пользователя;
2) определить формат файлов переписки;
3) разбор найденных файлов;
4) автоматизировать процесс поиска файлов;
5) производить сохранение полученных информации формат XML.

6.4.1 Определение местоположения логов переписки мессенджера Skype

По умолчанию файлы располагаются в директориях, для разных операционных систем возможны незначительные изменения (таблица 1).

Таблица 1 – Директории хранения логов Skype

Windows Vista/7/8	C:\Users\username\ AppData\Roaming\Skype
Windows XP	C:\Documents and Settings\username\ Application Data\Skype
Linux(Ubuntu)	/home/username/.Skype

Основная интересующая нас информация находится в main.db [9]. Алгоритм поиска логов представлен на рисунке 6.1.

6.4.2 Определение местоположения логов переписки мессенджера Pidgin

Определение месторасположения файлов переписки происходит следующим образом: для примонтированного образа запускается модуль, который сужает область поиска, сканируя только нужные директории образа.

По умолчанию файлы располагаются в директориях, для разных операционных систем возможны незначительные изменения (таблица 2).

Таблица 2 – Директории хранения логов pidgin

Windows Vista/7/8	C:\Users\username\AppData\ Roaming\.purple\logs
Windows XP	C:\Documents and Settings\username\ Application Data\.purple
Windows 98/ME	C:\Windows\Profiles\username\.purple
Linux(Ubuntu)	/home/username/.purple/logs

Основная интересующая нас информация хранится в файлах с маской име-

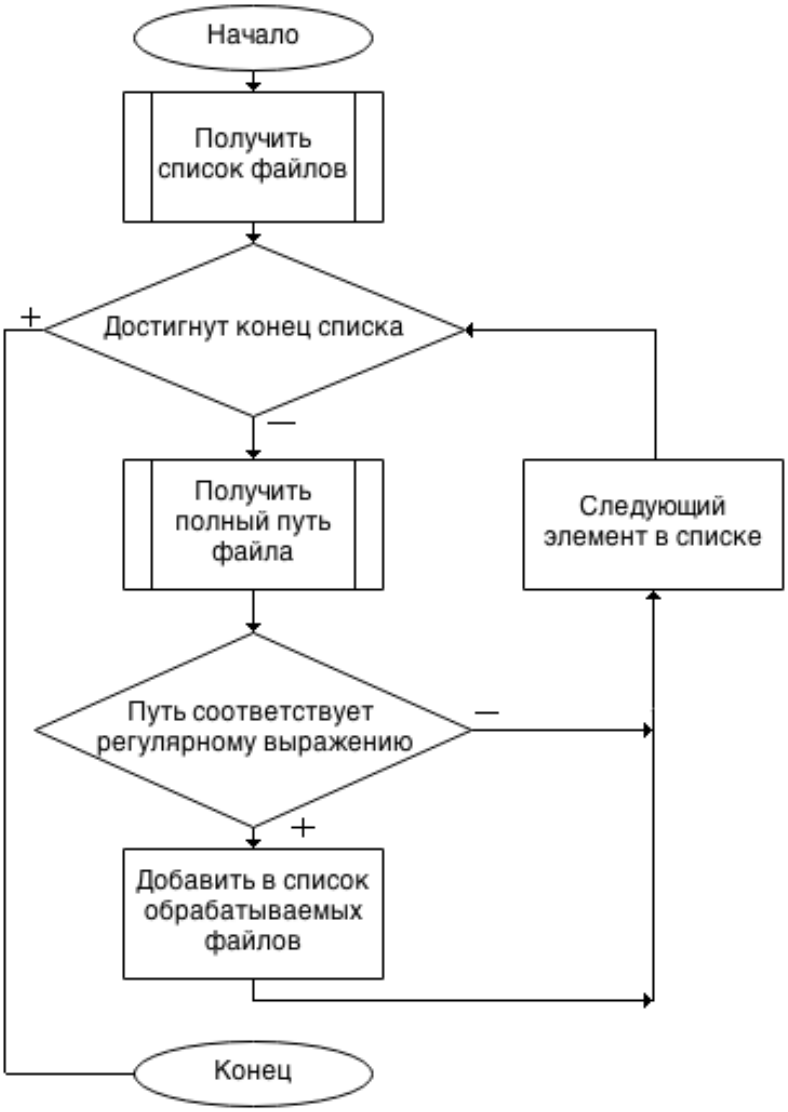

Рисунок 6.1 – Блок-схема алгоритма поиска логов

ни YEAR-MONTH-DATE.TIME.html (например 2013-03-02.004915+0700NOVT.htm).

6.4.3 Формат хранения переписки мессенджера Skype

Приложение «skype» хранит переписку на локальных машинах пользователей или же возможна синхронизация с машин других пользователей [9]. Формат хранения: реляционная база данных основная на СУБД SQLite. Но база данных «main.db» не является единственным местом хранения информации, Skype сохраняет сведения о работе программы во временных файлах (chatsync). Эти файлы имеют расширение «.dat» и цифробуквенные имена (например «0172b0a519e2c584»)[1

6.4.4 Формат хранения переписки мессенджера Pidgin

Приложение «pidgin» поддерживает перечисленные ниже протоколы: [11]

- ”AIM”
- ”Bonjour”
- ”Gadu-Gadu
- ”Google Talk”
- ”Groupwise”
- ”ICQ”
- ”IRC”
- ”MSN”
- ”MXit”
- ”MySpaceIM”
- ”SILC”
- ”SIMPLE”
- ”Sametime”
- ”XMPP”
- ”Yahoo!”
- ”Zephyr”

Соответственно, все подлюченные аккаунты всех вышеперечисленных протоколов хранятся как лог файлы на локальной машине пользователя в формате HTML и TXT. По умолчанию лог файлы хранятся в .HTML файле. Настройки программы, пользователя и подключенных аккаунтов хранятся в XML.

Примечание: Кроме account.xml - он хранит нешифрованные пароли для всех подключенных чатов [11].

6.4.5 Алгоритм работы модуля Skype

Структура рассматриваемого файла.
main.db содержит 18 таблиц. и ещё немного статистики.

- ”DbMeta”
- ”Contacts”
- ”LegacyMessages”
- ”Calls”
- ”Accounts”
- ”Transfers”
- ”Voicemails”
- ”Chats”
- ”Messages”
- ”ContactGroups”
- ”Videos”
- ”SMSes”
- ”CallMembers”
- ”ChatMembers”
- ”Alerts”
- ”Conversations”
- ”Participants”
- ”VideoMessages”

Таблицы которые были рассмотрены, на данный момент:

1) Contacts
2) Messages

3) Chats

4) Calls

5) CallMembers

6) Conversations

В таблице Contacts находятся все контакты, причем даже те, что были удалены, и уже не показываются в клиенте.

select skypename, fullname, languages, country, city from contacts

В таблицах Calls и CallMembers содержатся, соответственно, история звонков и их участников.

select calls.id as "ID разговора", coalesce(contacts.displayname, accounts.fullname) as "Инициатор", strftime('%d.%m.%Y %H:%M:%S',calls.begin_timestamp, 'unixepoch', 'localtime') as "Дата начала", time(calls.duration, 'unixepoch') as "Длительность", callmembers.dispname as "Подключенный участник", strftime('%d.%m.%Y %H:%M:%S' 'unixepoch', 'localtime') as "Дата подключения", time(callmembers.call_duration, 'unixepoch') as "Длительность подключения" from calls inner join callmembers on calls.id = callmembers.call_db_id left join contacts on calls.host_identity = contacts.skyp left join accounts on calls.host_identity = accounts.skypename

И, наконец, в таблицах Conversations и Messages содержатся данные переписки и сами сообщения.

select conversations.id as "ID переписки", conversations.displayname as "Участники переписки", messages.from_dispname as "Автор сообщения", strftime('%d.%m.%Y %H:%M:%S',messages.timestamp, 'unixepoch', 'localtime') as "Время сообщения", messages.body_xml as "Текст сообщения" from conversations inner join messages on conversations.id = messages.convo_id order by messages.timestamp

Для доступа ко всему содержимому базы достаточно иметь доступ к самому файлу — содержимое базы никак не шифруется и не защищается, так что любой человек, который сможет получить доступ к вашему профилю Windows, сможет найти список контактов, просмотреть историю звонков и прочитать всю переписку.

6.4.6 Алгоритм работы модуля Pidgin

Структура рассматриваемого файла.

У каждого лог файла есть заголовок находящийся между тегов title. В котором записан ID_Chat, дата начала переписки, логин пользователя и используемый протокол.

Затем идет ”тело” в котором описывается обмен сообщениями в формате, время, автор сообщения и сообщение.

Пример ”Заголовка” <head><meta http-equiv = ”content-type” content= ”text/html; charset=UTF-8”> <title> Conversation with 0dpkhcz6clufs2kozj82uqif30@public.talk.google.com at Чт. 24 окт. 2013 23:40:21 on user.fox@gmail.com/ (jabber)</title></head>

Пример ”Тела”   (23:44:52) user.fox@gmail.com/95C9F047: Hello)

Точно так же из полученного списка найденные файлы поочередно открываются для чтения. Разбор открытого файла решено осуществлять при помощи регулярных выражений, описанных в класс QRegExp.

Парсинг логов pidgin представлен на рисунке 6.2.

6.4.7 Структура логов Skype, сохранённых в XML

Все документы XML начинаются с пролога (prolog). Пролог сообщает, что документ написан на XML, а также указывает, какая версия XML при этом использовалась.

Следующий элемент Messages с атрибутом Messenger, ему присваивается имя рассматриваемого приложения.

Для каждого файла БД, находящейся в обрабатываемом списке, создается элемент info account содержащий атрибуты skypeName, fullName, emails, ipcountry которым присваивается, соответственно: логин пользователя Skype, его полное имя, email (не шифрованный), геолокация ip-адреса.

Далее следует элемент contact содержащий атрибуты skypeName, fullName,

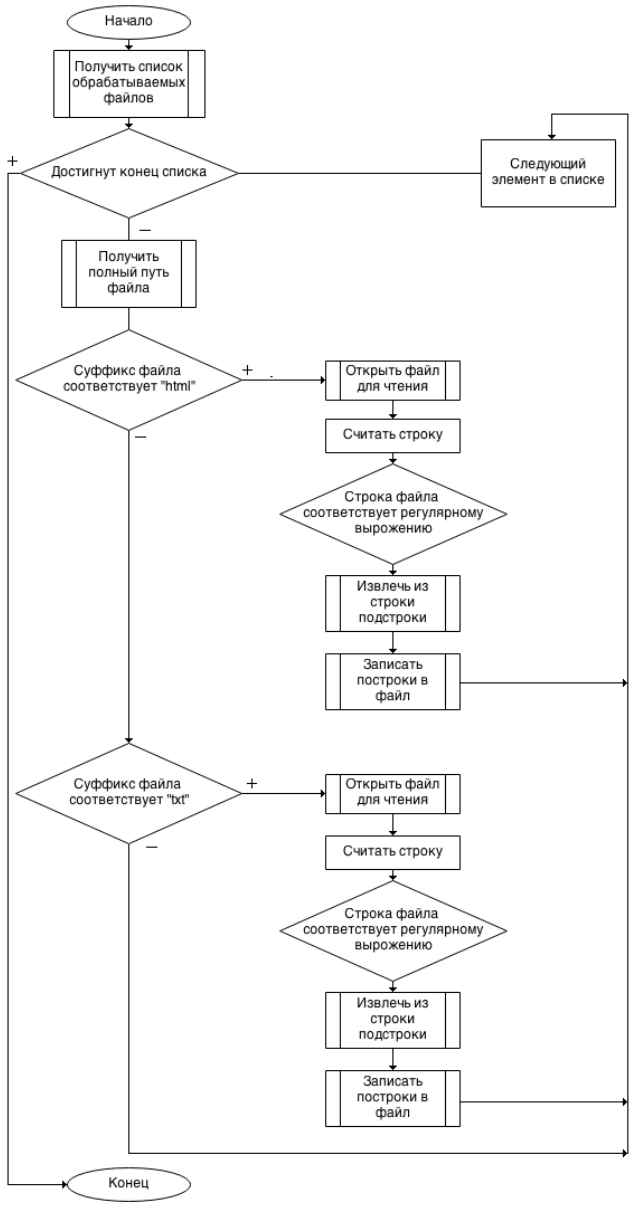

Рисунок 6.2 – Блок-схема алгоритма парсинга логов

languages, country, city.

6.4.8 Структура логов Pidgin, сохранённых в XML

Все документы XML начинаются с пролога (prolog). Пролог сообщает, что документ написан на XML, а также указывает, какая версия XML при этом использовалась.

Следующий элемент Messages с атрибутом Messenger, которому присваивается имя рассматриваемого приложения.

Далее следует элемент INFO содержащий атрибуты chathID, account, data, protocol которым присваивается, соответственно: идентификатор чата, полная

дата в формате День.Число Месяц. Год Час:Мин:Сек, аккаунт с которого происходил обмен сообщениями и протокол используемый для передачи сообщений.

Последний элемент – MESSAGE, содержащий атрибуты author, dataTime, message.

На текущий момент полностью реализован импорт контактной книги из приложения Skype, импорт звонков и переписки находится в режиме разработки.

7 Некоторые аспекты сертификации программных средств объектов информатизации по требованиям информационной безопасности

Под информационной безопасностью объектов информатизации в общем случае понимается такое их состояние, при котором исключается нанесение неприемлемого ущерба субъектам информационных отношений при применении последними средств информатизации. Следовательно, для оценки информационной безопасности объектов информатизации важно оценить сначала степень информационной безопасности средств информатизации, применяемых на объектах информатизации, в том числе всех их компонентов.

Одной из важнейших составляющих любого объекта информатизации являются применяемые программные средства различного назначения, которые существенно влияют на информационную безопасность объекта информатизации в целом.

В настоящее время достаточно полно регламентирована законодательными актами и распорядительными документами организация работ по сертификации средств защиты информации. Разработаны и представлены в виде нормативных документов (Государственных стандартов и Руководящих документов Гостехкомиссии России) требования к защиты информации автоматизированных систем и их компонентов (вычислительных и программных средств). Отработаны методы их проверки и создано достаточно большое количество программных средств для проведения испытаний. Однако в основной их части требования касаются средств и методов защиты информации от несанкционированного доступа, а также проверок на отсутствие закладных деструктивных элементов в таких программных

средствах.

В то же время информационная безопасность средств информатизации определяется не только их защищенностью от несанкционированного доступа. Одной из важнейших составляющих является выполнение средством информатизации заданных функций в различных условиях функционирования, в том числе при воздействии внешних деструктивных факторов.

Действительно, стержневой характеристикой качества любого средства информатизации является его функциональная пригодность. Ибо, если средство информатизации не решает в заданном объеме и с заданным качеством установленных для него задач, то нет смысла обеспечивать защиту от несанкционированного доступа и нецелесообразно его применять по назначению, так как только по этой причине результаты его использования могут привести к непредсказуемым последствиям и нанести пользователю неприемлемый ущерб.

В рамках Системы сертификации «Росинфосерт» разрабатывается подход к сертификации средств информатизации по требованиям информационной безопасности, сущность которого основана на действующей нормативно-методической базе Системы сертификации «Росинфосерт». Эта нормативно-методическая база дорабатывается с учетом требований Федерального закона «О техническом регулировании», в том числе в части вычислительных и программных средств, а также компьютерных систем в целом.

В ФЗ «О техническом регулировании» определены типы технических регламентов, в которых должны быть сформулированы требования по обеспечению биологической, механической, пожарной, промышленной, химической и др. видов безопасности применительно к продукции, работам и услугам. К сожалению, не попали в этот список требования по обеспечению информационной безопасности. Очевидно, Законодатель считает ее составной частью всех остальных видов безопасности.

Согласно закону, безопасность продукции определяется ее характеристиками, безопасность работ и предоставляемых услуг определяется применением безопасной продукции и безопасностью методов использования этой продукции при

работах и предоставлении услуг. Фактически безопасность продукции является одной из составляющих ее качества.

Таким образом, информационная безопасность средств информатизации должна оцениваться в рамках общей оценки их качества, как один из показателей качества продукции.

При рассмотрении вопросов информационной безопасности исследуется и оценивается безопасность информационных ресурсов (данных, программ и их совокупности), которые, в свою очередь нельзя рассматривать в отрыве от вычислительных средств, на которых они размещены и реализованы. То есть предметом рассмотрения должны быть вычислительные средства с реализуемыми на них информационными ресурсами.

Таким образом, требования информационной безопасности следует применять к многоуровневому программно-вычислительному комплексу как единому целому (рисунок 7.1):

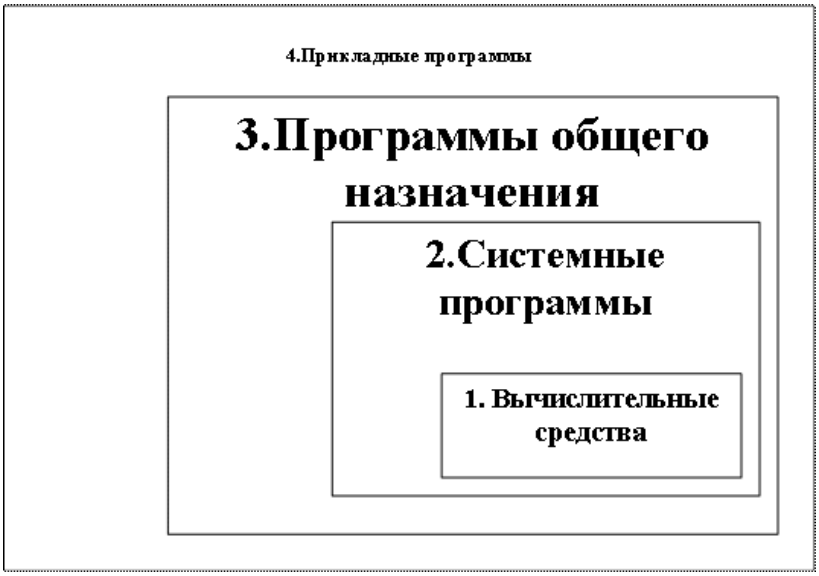

Рисунок 7.1 – Многоуровневый программно-вычислительный комплекс

В характеристики качества (в том числе и в информационную безопасность) такого комплекса компоненты каждого уровня вносят свою составляющую. Например, недостаточная надежность технических компонентов вычислительных средств может компенсироваться программно-алгоритмическими решениями. В конечном счете, следует рассматривать в качестве основного показателя качества

комплекса безопасность его функционирования.

Поскольку 100% безопасности функционирования любого комплекса определенной структуры быть не может, то можно говорить о безопасности комплекса лишь в вероятностном смысле. Это в полной мере относится к компонентам любого уровня.

В общем случае для прикладных программных средств безопасное функционирование означает:

- полное и точное выполнение всех заданных функций;
- обеспечение целостности и сохранности;
- обеспечение защиты от неправильных действий пользователя, от некорректных входных данных, от случайных сбоев вычислительных средств;
- простой и удобный интерфейс.

Эти составляющие обеспечивают как аппаратные средства, так и операционная среда, Однако, центральным моментом оценки качества прикладных программных средств должна являться оценка их собственных функциональных характеристик, но, прежде, чем провести оценку качества прикладных программных средств, следует убедиться в том, что характеристики остальных составляющих комплекса соответствуют предъявляемым к ним требованиям, в том числе требованиям информационной безопасности.

В свете всего сказанного предлагается следующая этапность оценки:

1-й этап – осуществляется оценка выполнения требований к качеству комплекса на первом уровне. Здесь исследуются и оцениваются характеристики преимущественно технических средств с использованием широкой номенклатуры специальных или специализированных тестов.

2-й этап – осуществляется оценка уже программно – аппаратного комплекса, включающего средства 1-го и 2-го уровней (технические средства и системные программные средства). При исследованиях и оценках используются имитаторы (в том числе программные) сигналов внешних устройств и функционирования программ общего назначения.

3-й этап – осуществляется оценка программно – аппаратного комплекса, включающего средства 1-го, 2-го и 3-го уровней (технические средства, системные программные средства и программные средства общего назначения). При оценках также используются независимые имитаторы сигналов внешних устройств, функционирования программ общего назначения и прикладных программ.

И, наконец, на 4-м этапе осуществляется оценка характеристик качества всего программно-вычислительного комплекса в целом. При этом используются результаты всех предыдущих этапов, что повышает достоверность и доверие к полученным оценкам.

Обязательные требования к продукции (работам и услугам) по действующему законодательству Российской Федерации устанавливаются Техническими регламентами, принимаемыми в качестве Федеральных законов. Сегодня необходимость технического регламента, устанавливающего обязательные требования по информационной безопасности к программно-вычислительным комплексам очевидна. При этом основным инструментом контроля соблюдения таких требований является сертификация (подтверждение соответствия). Место Системы сертификации в рамках проведения единой технической политики России в области решения задач информатизации поясняется рисунком 7.2.

Регистрация в реестре Системы сертифицированной продукции и выдача сертификата соответствия заявителю. Применение результатов сертификации продукции можно проиллюстрировать примером проведения тендера на поставку средств информатизации для государственных нужд. Схема проведения такого тендера представлена на рисунке 7.3.

Отбор участников тендера целесообразно проводить на основе анализа результатов их деятельности, одним из объективных показателей которой является сертификат соответствия системы менеджмента качества организации требованиям международных стандартов ИСО 9001 – 2001, а также наличие в номенклатуре выпускаемой продукции сертифицированных продуктов, характеристики которых представлены в Реестре Системы. Такой подход ставит барьер недобросовестным поставщикам и некачественной продукции на рынок средств инфор-

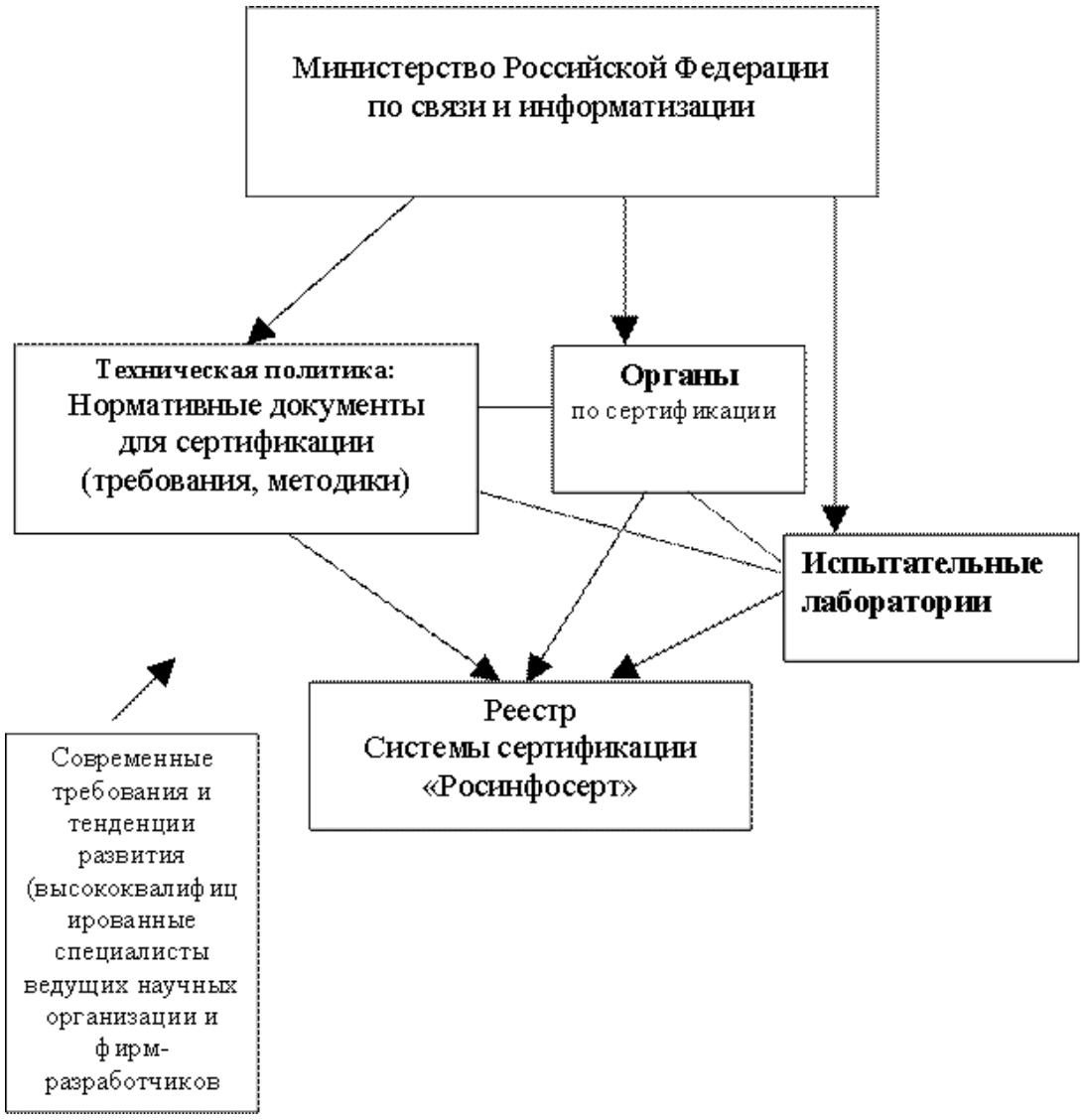

Рисунок 7.2 – Система сертификации при проведении единой технической политики

матизации России.

В этом случае нормативно-правовое обеспечение применения сертификации для реализации обязательных требований информационной безопасности составляют следующие документы:

1) Соглашение о взаимодействии в области сертификации средств информатизации между Минсвязи России и субъектом равного уровня. Такое соглашение является необязательным, но желательным документом, который регламентирует взаимоотношения руководства Системы сертификации, ее органов и испытательных лабораторий с потенциальными поставщиками и потребителями средств информатизации, впрямую не

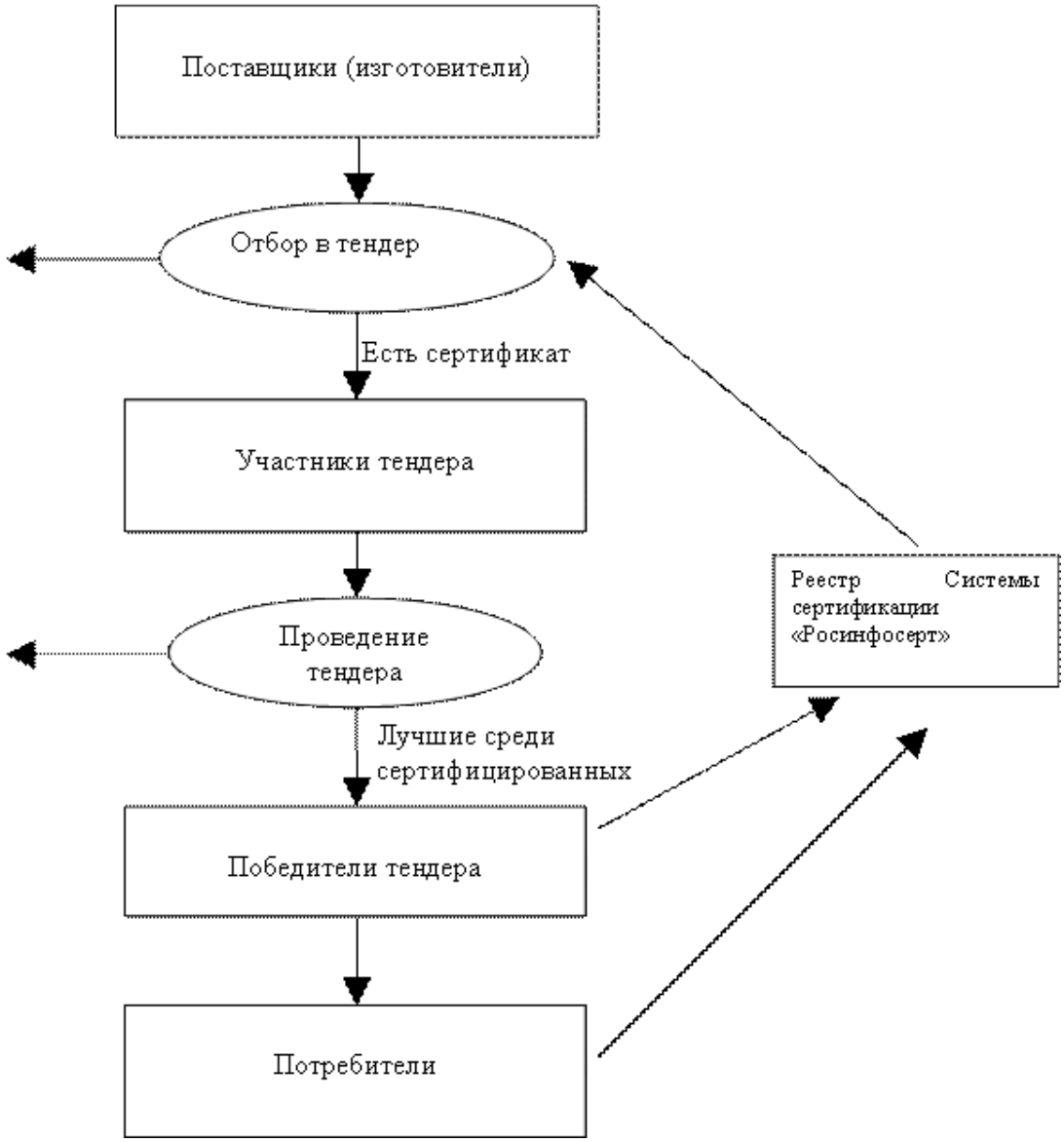

Рисунок 7.3 – Схема проведения тендера на поставку средств информатизации для государственных нужд

подчиняющимися Минсвязи России.

2) Распоряжение (постановление, приказ) субъекта об утверждении Положения о порядке использования средств информатизации для решения своих задач.

3) Положение о порядке использования субъектом средств информатизации, устанавливающее систему показателей и правил отбора и применения поставляемых для государственных нужд средств информатизации.

4) Нормативный документ для сертификации, содержащий состав характеристик средства информатизации, их допустимые значения и способы

оценки. Этот документ носит статус стандарта организации, утверждается Минсвязи России по согласованию с субъектом.

8 Цели следующего семестра

1) более глубокое изучение Git для лучшего и более правильного отображения истории;
2) учимся работать в отдельных ветках Git и сливать с master;
3) учимся работать с bugtracker;
4) осуществление переноса архитектуры на плагины;
5) тестирование системы;
6) подготовка deb-пакета, лицензии;
7) написание публикации.

9 Заключение

В данном семестре нашей группой была выполнена часть работы по созданию автоматизированного программного комплекса для проведения компьютерной экспертизы, проанализированы дальнейшие перспективы и поставлены цели на следующий семестр.

Список использованных источников

1 Федотов Н. Н. Форензика - компьютерная криминалистика. — Юрид. мир, 2007. — P. 432. — ISBN: 9785911590154.

2 Chacon S. Pro Git : professional version control. — 2011. — URL: http://progit.org/ebook/progit.pdf.

3 Львовский С. Набор и вёрстка в системе LaTeX. — МЦНМО, 2006. — P. 448. — ISBN: 5-94057-091-7.

4 Котельников И. А.; Чеботаев П. З. LaTeX 2_ε по-русски. — Сибирский Хронограф, 2004. — P. 489.

5 Qt Documentation [Электронный ресурс]. — 2013. — URL: http://qt-project.org/doc/.

6 Всё о кроссплатформенном программировании - Qt [Электронный ресурс]. — 2013. — URL: http://doc.crossplatform.ru/qt/.

7 Справочник по XML-стандартам [Электронный ресурс], Microsoft. — URL: http://msdn.microsoft.com/ru-ru/library/ms256177(v=vs.110).aspx.

8 Windows Event Log Format [Электронный ресурс]. — 2005. — URL: http://www.whitehats.ca/main/members/Malik/malik_eventlogs/malik_eventlogs.html.

9 Официальное сообщество Skype [Электронный ресурс]. — URL: http://community.skype.com/.

10 Юрьевич М. И. Компьютерно-техническая экспертиза. — 2007 – 2013. — URL: http://computer-forensics-lab.org/.

11 Официальное сообщество Pidgin. — URL: https://developer.pidgin.im/.

Приложение А
(Обязательное)
Компакт-диск

Компакт-диск содержит:

– электронную версию пояснительной записки в форматах *.tex и *.pdf;

– актуальную версию программного комплекса для проведения компьютерной экспертизы.