Оглавление

[Введение 4](#_Toc453026326)

[1 Теоретическая часть квалификационной работы 5](#_Toc453026327)

[1.1 О медицинских информационных системах 5](#_Toc453026328)

[1.2 Обзор существующих медицинских информационных систем 7](#_Toc453026329)

[1.3 Предпосылки разработки модуля "Статистика" 8](#_Toc453026330)

[1.4 Обзор средств разработки 9](#_Toc453026331)

[1.4.1 Язык программирования для разработки 9](#_Toc453026332)

[1.4.2 Библиотеки элементов интерфейса 11](#_Toc453026333)

[1.4.3 Система управления базами данных 12](#_Toc453026334)

[1.5 Выбор средств разработки 13](#_Toc453026335)

[1.6 Структура базы данных МИС САМСОН 14](#_Toc453026336)

[2 Практическая часть квалификационной работы 15](#_Toc453026337)

[2.1 Техническое задание 15](#_Toc453026338)

[2.2 Структура модуля 16](#_Toc453026339)

[2.3 Интерфейс и функциональные возможности 18](#_Toc453026340)

[3 Социально-экономическая часть 19](#_Toc453026341)

[3.1 Конкурентный анализ использования медицинских информационных систем 19](#_Toc453026342)

[3.2 Эффективность использования модуля «Статистика» 19](#_Toc453026343)

[3.3 Расчеты экономической эффективности использования модуля 19](#_Toc453026344)

[Заключение 20](#_Toc453026345)

[Список литературы 21](#_Toc453026346)

[Приложение А. Техническое задание 22](#_Toc453026347)

# Введение

В соответствии с приказом Министерства здравоохранения Российской Федерации от 28 апреля 2011 года № 364 «Об утверждении концепции создания единой государственной информационной системы в сфере здравоохранения» подавляющее число медицинских учреждений разных форм собственности, занимающихся лечением пациентов, внедряют информационные медицинские системы (далее МИС). Основная задача МИС – это выполнение функции сбора, хранения, обработки, выдачи информации.

Цель дипломного проекта: разработка модуля «Статистика» комплекса САМСОН для РНЦ ВТО им. Илизарова.

При разработке модуля «Статистика» были поставлены следующие задачи:

* Проанализировать предметную область.
* Провести анализ требований и определить необходимую функциональность разрабатываемого модуля.
* Определить программные средства разработки.
* Спроектировать и разработать модуль.

# Теоретическая часть квалификационной работы

## О медицинских информационных системах

В современном мире большинство медицинских учреждений, заинтересовано в эффективном обслуживании пациентов.

В числе многочисленных задач в области управления здравоохранением наиболее актуальными являются следующие: «оперативное получение достоверных первичных данных об объемах и качестве медицинской помощи, оказываемой медицинскими организациями; планирование обоснованных затрат на оказание гарантированных объемов медицинской помощи в соответствии со стандартами качества»1.

Поэтому очень популярными становятся медицинские информационные системы, предлагающие сбор, хранение и обработку информации, связанную с пациентами клиники.

Медицинская информационная система (МИС) – система автоматизации документооборота для лечебно-профилактических учреждений, в которой объединены система поддержки принятия медицинских решений, электронные медицинские карты о пациентах, данные медицинских исследований в цифровой форме, данные мониторинга состояния пациента с медицинских приборов, средства общения между сотрудниками, финансовая и административная информация.[[1]](#footnote-1)

<http://cyberleninka.ru/article/n/elektronnaya-istoriya-bolezni-vazhneyshee-zveno-meditsinskih-informatsionnyh-sistem>

<http://www.postmodern.ru/?tree_id=413>

<http://www.idmz.ru/idmz_site.nsf/pages/vit2016_1.htm>

<http://www.kmis.ru/site.nsf/pages/2009_obzor_kmis.htm>

## Обзор существующих медицинских информационных систем

## Предпосылки разработки модуля "Статистика"

однократный ввод и многократное использование первичной информации

создание прикладных информационных систем по модели «программное обеспечение как услуга  
  
ГАРАНТ.РУ: <http://www.garant.ru/products/ipo/prime/doc/4092541/#ixzz4ApF1E6Xj>

В рамках научно-исследовательских организаций обеспечивается автоматизация процессов доступа к информации, имеющейся в Системе.  
  
ГАРАНТ.РУ: <http://www.garant.ru/products/ipo/prime/doc/4092541/#ixzz4ApFm9fwa>

система ведения интегрированной электронной медицинской карты, а также создаваемых на ее основе специализированных регистров по отдельным нозологиям и категориям граждан, в том числе обеспечивающая персонифицированный учет медицинской помощи и лекарственного обеспечения;  
  
ГАРАНТ.РУ: <http://www.garant.ru/products/ipo/prime/doc/4092541/#ixzz4ApGNqBde>

аналитические системы, а также системы автоматизированного контроля и поддержки принятия управленческих решений на основе анализа первичных данных.  
  
ГАРАНТ.РУ: <http://www.garant.ru/products/ipo/prime/doc/4092541/#ixzz4ApGRQ5Iy>

## Обзор средств разработки

### Язык программирования для разработки

Выбор языка разработки не такая уж и простая задача, так как язык программирования должен отвечать требованиям производительности, безопасности, скорости разработки на языке, распространенности, документированности и ещё большому количеству требований разработчика.

В связи с этим для разработки было отобрано несколько кандидатов: C#, Ruby, Java, Javascript.

C# -- ООП язык программирования разработанный в компании Microsoft. C# имеет статическую, строгую типизацию. К плюсам данного языка программирования можно отнести встроенную поддержку .NET -- графического фреймворка позволяющего создавать кроссплатформенные приложения (не зависящие от операционной системы). Поддержка событийно-ретинированного программирования. Унифицированная система типизации. К недостаткам можно отнести слабую поддержку ОС отличных от Windows, довольно сложный синтаксис, относительно невысокую производительность.

Java -- объектно-ориентированный язык с статической типизацией и кроссплатформенной реализацией. Код на java транслируется в специальной байт-код и выполняется виртуальной машиной, поэтому язык является интерпретируемым и может выполняться на любой платформе. К преимуществам языка можно отнести автоматического управление памятью, богатый набор средств фильтрации исключительных ситуаций, наличие классов позволяющий выполнять различные http запросы и обрабатывать ответы, встроенную поддержку многопоточности. Так же Java позволяет разрабатывать приложения не только для компьютеров, но и для мобильных устройств. Благодаря этому вокруг языка сформировано большое количество разработчиков, которые помогают делать этот язык лучше. Ситуация с GUI фреймворками как и у всех кроссплатформенных языков не очень хороша, тут и вездесущий qt, и wxWidgets, и не очень популярный AWT на основе которого реализовано множество других фреймворков графического интерфейса для java.

К недостаткам java модно отнести не очень быструю скорость выполнения программы из-за отказа от арифметики указателей, неявных преобразований типов с потерей точности. Так же в java полностью отсутствует возможность управления памятью, объект будет удален только тогда, когда на него не будет существовать указателей и, интерпретатор запустит очередную «сборку муосра»

Ruby -- интерпретируемый язык программирования высокого уровня. Ruby, это тщательно сбалансированный язык. Его создатель Юкури Мацумото, объединил части своих любимых языков, для того чтобы сформировать новый язык, в котором парадигмы функционального и императивного программирования будут сбалансированы. Ruby получил свою популярность благодаря фреймворку Ruby on Rails который реализует среду разработки веб приложений. Преимущества Ruby:

* кроссплатформенный
* может внедряться в HTML разметку (как PHP)
* реализует «чистую» концепцию объектно-ориентированного программирования
* независимая от операционной системы реализация многопоточности
* итераторы
* расширяется за счет библиотек, написанных на C или Ruby
* встроенный отладчик
* огромное сообщество разработчиков
* централизованная система управления пакетами для языка

Недостатки:

* низкая производительность
* слабая русскоязычная вещей, описывающих поведение выше начального уровня

Javascript – прототипно-орейнтированный сценарный язык программирования. Контролируется нотацией ECMAScript. JavaScript обычно используется для выполнения сценариев внутри браузера, однако только этим использование языка не ограничивается. Так, например, node.js разрешает использовать JavaScript на сервере, позволяя обрабатывать входящие запросы пользователя и работать с базой данных. Также появился фреймворк для построения десктопных графических приложений на JavaScript который называется Electron. Достоинства языка:

* массовость
* динамическая типизация
* поддерживается огромным количеством устройств, телефоны, планшеты, компьютеры, всем что имеет браузер
* применение во многих областях

Недостатки:

* необходимость обеспечения кроссбраузерности
* система наследования вызывает трудности после перехода с ООП языка

### Библиотеки элементов интерфейса

При разработке приложений мы сталкиваемся с необходимостью отображения графического интерфейса для пользователя. Так-как проблему, связанную с отображением пользовательского интерфейса, приходится решать часто и это занимает много времени, то для того что бы ускорить процесс разработки в большинстве проектов используются библиотеки элементов интерфейса.

В каждой системе, имеющей графический интерфейс, присутствует набор стандартных простых элементов интерфейса с возможностью работы с ними. Такие простые элементы называются «виджетами», они могут быть вложены друг в друга что позволяет строить сложные пользовательские интерфейсы. Библиотеки элементов интерфейса осуществляют «высокоуровневый» доступ к элементам интерфейса операционной системы, позволяя не заботится о размещении и компоновкой элементов друг с другом. Также большинство библиотек имеют независимую от операционной системы реализацию, что позволяет программисту разрабатывать приложение не под определённую операционную систему, а под все распространенные сразу.

Нам необходимо что бы приложение работало на любой операционной системе, поэтому будут рассматриваться только кроссплатформенные библиотеки графического интерфейса.

Qt – свободный, кроссплатформенный инструмент для построения графического интерфейса на C++, также есть «привязки» к другим языкам программирования таким как: Python, Ruby, Java, PHP и другим.

Отличительной особенностью qt является использование системы сигналов и слотов, которые реализуют шаблон проектирования, называемый «наблюдателем». Концепция заключается в том, что компонент может посылать сигнал о событии (выделение текста, нажатия кнопки и др.), а другие компоненты могут принимать эти сигналы с помощью специальных функций называемых слотами.

Ещё один плюс использование qt является наличие качественной документации. Статьи документации имеют множество качественных примеров, благодаря которым значительно упрощается понимание материала. Исходный код библиотеки qt подробно комментирован, хорошо отформатирован, и легко читается, что также упрощает изучение данного фреймворка.

Другая библиотека графического интерфейса это FLTK (Fast, Light Toolkit). Она также является кроссплатформенной и имеет свободную лицензию. Изначально библиотека разрабатывалась для поддержки 3D графики, поэтому имеет встроенный интерфейс к OpenGL, но хорошо подходит и для построения обычного пользовательского интерфейса. Библиотека использует независимые от операционной системы модель отображения, поэтому приложения выглядят в операционной системе неестественно. Также к недостаткам можно отнести малое число виджетов по сравнению с другими графическими библиотеками. Данный фреймворк плохо документирован из-за чего сложен в изучении.

GTK+ -- популярный кроссплатформенный фреймворк, имеет простой в использовании API. GTK+ написан на Си, но является объектно-ориентированным. Причиной выбора языка Cи было желание строить интерфейсы для других языков программирования. Благодаря этому под GTK+ можно писать на большом количестве языков программирования (Erlang, Haskell, Perl, PHP, Lua, Lisp, Ruby и др.). При использовании GTK+ графический интерфейс может конфигурироваться пользователем или программистом, при этом осуществляется настройка не только цветов и шрифтов, но и способы отображения различных элементов. К недостаткам данной графической библиотеки можно отнести слабую документацию и скудное количество, и качество примеров. В русском сегменте программистов про GTK мало кто знает, поэтому на русском документации почти нет.

### Система управления базами данных

База данных (БД) – набор предназначенных для совместного использования логически связанных данных. База данных определятся однократно, а затем используется одновременно несколькими пользователями.

Система управления базами данных (СУБД) – это совокупность программных и лингвистических средств благодаря которым пользователи могут осуществлять управление созданием и использованием баз данных.

СУБД по способу доступа разделяются на:

* Файл-серверные
* Клиент-серверные

## Выбор средств разработки

Для разработки приложения «Статистика» был выбран скриптовый язык программирования Ruby. Ruby является открытым программным обеспечением и распространяется свободно под лицензией GNU General Public License. Этот язык программирования имеет большое сообщество разработчиков и поэтому в нем имеется множество готовых решений, благодаря которым процесс разработки значительно увеличивается.

Для построения графического интерфейса был выбран фреймворк Qt. Qt отлично показывает себя на распространенных платформах таких как windows, os x и графической среде gnome.

Для доступа к базам данных используется библиотека ActiveRecord. Данная библиотека реализует шаблон проектирования Active Record. Принцип работы AR: пусть в базе данных существует некоторая таблица. Для этой таблицы создается специальный класс AR, который является представлением таблицы как класса в программе. Таким образом каждый экземпляр данного класса соответствует одной записи таблицы, при чтении полей объектом считывается значение записи таблицы базы данных. Благодаря реализации Active Record в Ruby объявление двух классов и связей между ними будет выглядеть так:

class Client < ActiveRecord::Base

has\_many :clientAddress

end

class ClientAddress < ActiveRecord::Base

belongs\_to :client

end

И это всё. Это работает без единой строки конфигурации. В данном примере предполагается что в базе данных есть таблицы Client и ClientAddress. В ClientAddress есть внешний ключ с именем client\_id. Если соблюдать данное соглашение, то все будет работать само.

Как видно из примера использование Active Record значительно упрощает работу с базой данных и соответственно скорость работы программиста. Ему больше не нужно заботится о реализации доступа к базам данных.

## Структура базы данных МИС САМСОН

# Практическая часть квалификационной работы

## Техническое задание

Наименование системы: Модуль «Статистика» комплекса САМСОН.

Плановые сроки начала и окончания работы: Рабочий проект должен быть создан до конца июня 2016. Возможны доработки и исправления, но основной функционал к этому времени должен быть готов.

Источники и порядок финансирования: Финансирование данного проекта не предусмотрено. Все расходы берет на себя заказчик.

Порядок оформления и предъявления заказчику результатов работ: Работы по разработке модуля «Статистика» комплекса САМСОН идут поэтапно, каждый следующий этап устанавливается после выполненного по факту предыдущего.

Назначение системы: Данный проект создается для предпринимателей, чтобы они могли осуществлять расширенный поиск и проводить анализ результатов внутри медицинской информационной системы САМСОН.

В данном проекте будет реализован следующий функционал:

1. Поиск
2. Фильтрация
3. Сортировка
4. Экспорт

Подробное техническое задание находится в приложении А.

## Структура модуля

Модели, Контроллеры, Файлы

Модели

Client – модель содержит список пациентов

Event – модель содержит события происходящие с пациентами (запись на прием, стационарное лечение, экстренное поступление и др.)

Action – модель содержит действия выполняемые над пациентами (движение, поступление, выписка, терапия, фотосессия, услуги и др.)

Organisation – модель содержит организации и ЛПУ

OrgStructure – модель содержит подразделения, отделения, участки организаций

Kladr – модель содержит классификацию адресов России

Contract – Договор, содержит код оплаты

Контроллеры

Search\_window занимается логикой работы с интерфейсом пользователя. Содержит множество экшенов отвечающих за отображение справочников из базы данных, извлечение из интерфейса пользователя необходимых данных для составления запроса и экспорт таблиц во внешние файлы

S11ActiveRecord реализует интерфейс доступа к базе данных для приложения.

Файлы

./database/S11\_settings.rb – содержит настройки доступа к БД

./database/S11.rb – содержит код который осуществляет подключение объектных моделей бд к приложению

./database/tables/\*.rb – файлы содержат объектные модели, необходимые для доступа к таблицам базы данных

./module/adv\_qt.rb – расширяет класс qt для более простой работы с фреймвоком

./module/crutch.rb – содержит исправления необходимые для верного построения запросов к базе данных

./module/export.rb – описывает класс для экспорта данных из приложения в внешний мир

./Application.rb –занимается инициализацией и отображением главного окна приложения

./main.rbw –подключает все модули приложения и запускает приложение

./Search\_window.rb – занимается обработкой пользовательского интерфейса

./ui\_search\_window.rb – пользовательский интерфейс

В дополнение к этому для нормальной работы приложения необходимы gem’ы (внешние модули):

«qtbindings» – модуль, осуществляющий «привязку» qt к ruby.

«activerecord» – соединяет классы таблиц реляционных баз данных, чтобы упростить создание и использование объектов, данные которых требуют хранения в базе данных. Объекты Active Record содержат сохраненные данные и описывают поведение, для работы с этими данными.

«squeel» – оптимизирует activerecord позволяя писать запросы меньшим количеством строк.

## Интерфейс и функциональные возможности

# Социально-экономическая часть

## Конкурентный анализ использования медицинских информационных систем

## Эффективность использования модуля «Статистика»

## Расчеты экономической эффективности использования модуля

# Заключение

# Список литературы

Bodnar, J. (29 05 2016 г.). *Ruby Qt tutorial*. Получено из ZetCode, tutorials for programmers: http://zetcode.com/gui/rubyqt/

Gerasimov. (29 05 2016 г.). *САМСОН - Официальный сайт проекта , САМСОН-ВИСТА , Электронная регистратура , Электронная история болезни , Медстатистика , Типовая МИС , МИС , Автоматизация ЛПУ , самозапись, Стационар , поликлиника , больница, Запись на прием к врачу - Главная*. Получено из САМСОН - Официальный сайт проекта: http://samson-rus.com/

*MySQL.RU .:. Одобрено лучшими российскими программистами*. (29 05 2016 г.). Получено из MySQL.RU: http://www.mysql.ru/docs/man/

*Ruby — Викиучебник*. (05 29 2016 г.). Получено из Викиучебник: https://ru.wikibooks.org/wiki/Ruby

*Rusrails: Интерфейс запросов Active Record*. (29 5 2016 г.). Получено из Rusrails: Ruby on Rails по-русски: http://rusrails.ru/active-record-query-interface

*Разработка/Руководства/Введение в Qt4 на Ruby — KDE TechBase*. (29 05 2016 г.). Получено из KDE TechBase: https://techbase.kde.org/Development/Tutorials/Qt4\_Ruby\_Tutorial/ru

Фултон, Х. (2014). *Программирование на языке Ruby.* Litres.

Шлее, М. (2012). *Qt 4.8. Профессиональное программирование на C++.* БХВ-Петербург.

# Приложение А. Техническое задание

НЕЗАБУДЬ ВОТКНУТЬ ТЗ СЮДА

1. 1 Приказ Министерства здравоохранения и социального развития РФ от 28 апреля 2011 г. № 364 “Об утверждении концепции создания единой государственной информационной системы в сфере здравоохранения”  
     
   ГАРАНТ.РУ: <http://www.garant.ru/products/ipo/prime/doc/4092541/#ixzz4ApBMsQoR>

   https://ru.wikipedia.org/wiki/Медицинская\_информационная\_система [↑](#footnote-ref-1)