Оглавление

[Введение 4](#_Toc453802112)

[1 Теоретическая часть квалификационной работы 4](#_Toc453802113)

[1.1 О медицинских информационных системах 4](#_Toc453802114)

[1.2 Обзор существующих медицинских информационных систем 4](#_Toc453802115)

[1.3 Предпосылки разработки модуля "Статистика" 4](#_Toc453802116)

[1.4 Обзор средств разработки 4](#_Toc453802117)

[1.4.1 Язык программирования для разработки 4](#_Toc453802118)

[1.4.2 Библиотеки элементов интерфейса 4](#_Toc453802119)

[1.4.3 Система управления базами данных 4](#_Toc453802120)

[1.5 Выбор средств разработки 4](#_Toc453802121)

[1.6 Структура базы данных МИС САМСОН 4](#_Toc453802122)

[2 Практическая часть квалификационной работы 4](#_Toc453802123)

[2.1 Техническое задание 4](#_Toc453802124)

[2.2 Структура модуля 4](#_Toc453802125)

[2.3 Интерфейс и функциональные возможности 4](#_Toc453802126)

[2.4 Расчеты экономической эффективности использования модуля 4](#_Toc453802127)

[Заключение 4](#_Toc453802128)

[Список литературы 4](#_Toc453802129)

[Приложение А. Техническое задание 4](#_Toc453802130)

# 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

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.

Медицинская информационная система должна обеспечивать так же вывод информации о группах пациентов, эта информация необходима для анализа работы отделений медицинского центра и общей работы лечебно-профилактического учреждения (ЛПУ)

Внедрение МИС так же скрывает в себе проблемы, связанные со стрессом для персонала, непривыкшему к такой системе, и как следствие возможный саботаж. Такие решения приводят к замедлению работы при обслуживании пациентов и как следствие увеличению стоимости.

Однако с внедрением системы может возрасти уровень сервиса, в том числе доступный объем справочной информации для пациента.

## Обзор существующих медицинских информационных систем

Единая государственная информационная система в сфере здравоохранения министерства здравоохранения российской федерации. Подсистема мониторинга реализации государственного задания по оказанию высокотехнологичной медицинской помощи.

Областью применения информационно-аналитической системы Министерства здравоохранения Российской Федерации являются процессы сбора, обработки и анализа данных в сфере здравоохранения.

Подсистема мониторинга реализации государственного задания по оказанию высокотехнологичной медицинской помощи предназначена для автоматизации процессов сбора, обработки и анализа данных об оказании ВМП гражданам Российской Федерации.

Так как цель данной дипломной работы «Разработка модуля «Статистика»», то обзор будет осуществлен по возможностям других систем вывода статистики.

Система представляет следующие способы для вывода статистической информации:

Выбор колонок для отображения (Рисунок 1)

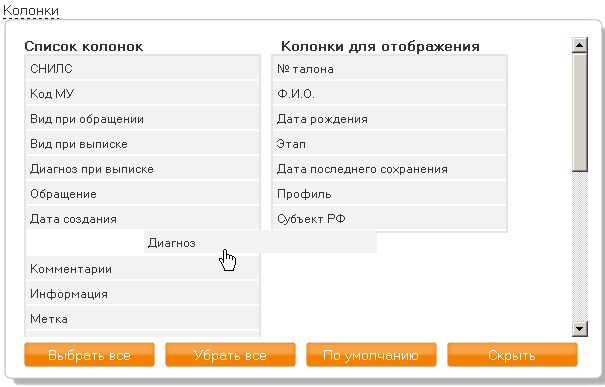


Рисунок 1 - Выбор колонок для отображения

Так же в системе существует возможность расширенного поиска представляющая поиск по полям, отображенным на рисунке (Рисунок 2).

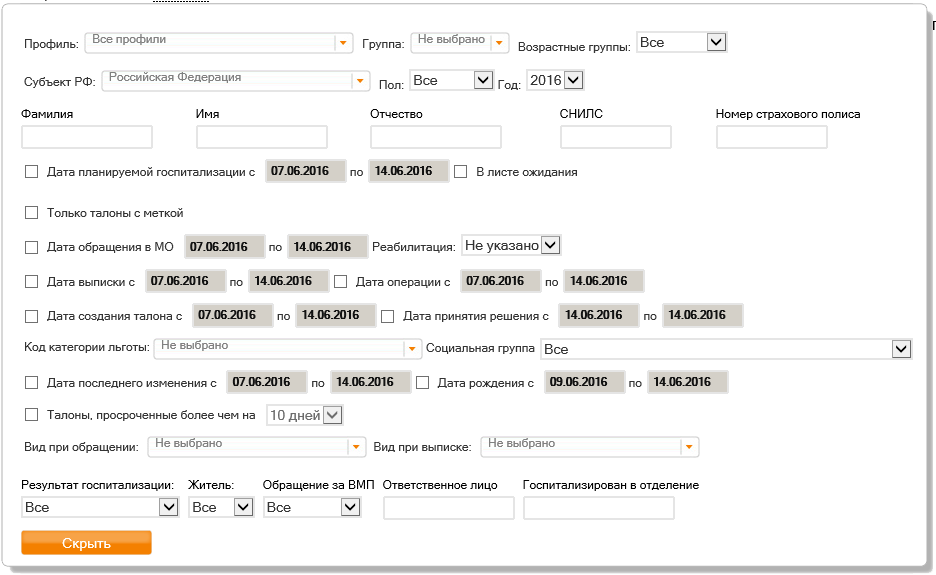


Рисунок 2 - Расширенный поиск

Информационная система "Медведь". Российская платформа Медведь автоматизирует работу органов управления здравоохранением регионального масштаба и предоставляет электронные услуги населению.

Функции платформы Медведь:

* Сбор и хранение персональных данных.
* Статистическая отчетность по утвержденным государственным формам отчетности.
* Нерегламентированная аналитическая отчетность на основе произвольных запросов клиента.
* Представление ключевых параметров системы здравоохранения на рабочем месте руководителя для наглядного визуального анализа.

<http://mis-region.ru/about/naznachenie-sistemy/> как оформить ссылку на список?

Медицинская информационная система «Медведь» представляет возможность вывода статистической информации в виде заранее подготовленных отчетных форм. Такой подход не всегда удобен, так как исключает возможность получения определенной информации из медицинской системы. Система сбора отчетности МИС «Медведь» может быть внедрена одним из двух способов. Способы отличаются наличием в лечебно-профилактическом учреждении собственной системы. Диаграмма внедрения представлена на рисунке (Рисунок 3).

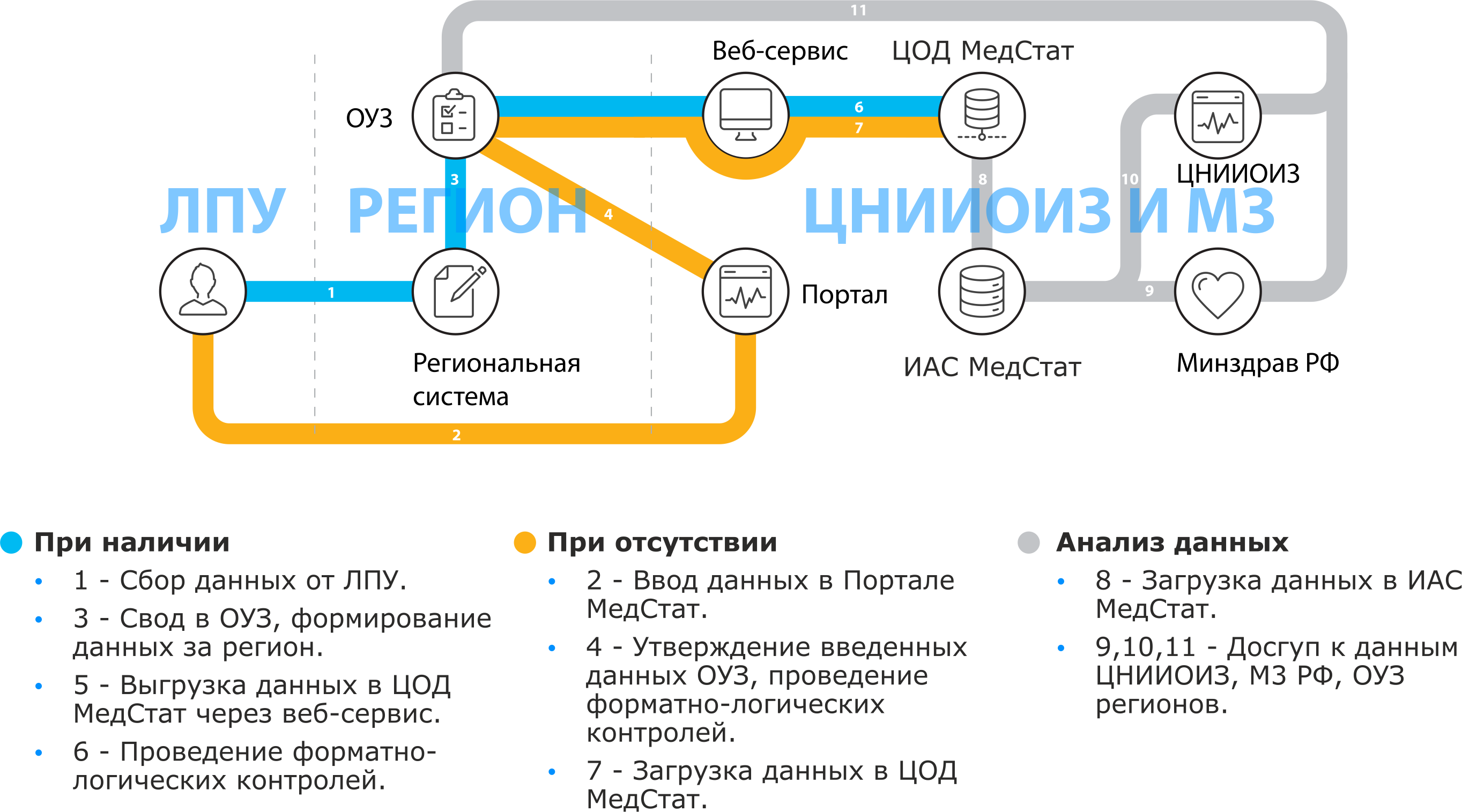


Рисунок 3 - Медведь внедрение

## Предпосылки разработки модуля "Статистика"

Приложение к программе «Самсон» позволит решить многочисленные проблемы, связанные с учетом и анализом состава больных: пролеченных в стационаре, больных, находящихся на лечении в стационаре, а также амбулаторных пациентов.

1. Ведение базы данных «Нозологический и функциональный каталог».

С 1970 года в архиве Центра Илизарова существует информационно-поисковая система "Нозологический и функциональный каталог". До 1998 года она существовала в виде картотеки, затем была создана электронная версия. В нее вносятся сведения из истории болезни по каждому этапу лечения больного. (Ф.И.О. больного, возраст, пол, регион проживания, отделение, дата выписки, заключительный клинический диагноз, код по МКБ-10[[1]](#footnote-1), название операции, архивный номер). Далее каталог дополняется информационными систематическими сведениями: нозологическая форма заболевания, сегмент, локализация и т.д.

Нозологический каталог используется в целях подбора пациентов по заданным параметрам для научно-исследовательской работы ФГБУ РНЦ «ВТО» им. акад. Г.А. Илизарова: подготовки докторских и кандидатских диссертаций, монографий, научных статей. Используется он и в клинической работе: для подготовки аналитических отчетов о работе стационара.

Информация из историй болезни пролеченных пациентов переносится в ИПС в ручном режиме. В связи с тем, что объем информации достаточно большой, это занимает значительное количество времени, не исключает возможные ошибки и неточности.

1. Заполнение «Журнала учета стационарных больных».

Журнал учета больных, поступивших на стационарное лечение в профильное отделение, в настоящее время заполняется в ручном режиме старшей медицинской сестрой отделения. Вся информация, необходимая для ведения журнала вносится в МИС «Самсон» на этапе оформления истории болезни.

1. Группировка данных по произвольно заданным параметрам для составления оперативных отчетов о работе клинических отделений и стационара в целом, проведения анализа работы учреждения.

В течение предыдущих лет для полноценного анализа работы клиники заполнялась база данных «Медицинская статистика», которая содержала информацию из «Статистической карты выбывшего больного» по тридцати одному пункту. С активным внедрением МИС «Самсон» дублирование информации оказалось излишне затратным по времени. Однако обнаружилось, что «Самсон» формирует только заранее созданные отчеты без возможности редактирования схемы запроса. Извлечь необходимую информацию оказалось невозможным. В качестве примера, центр Илизарова, в отличие от медицинских учреждений муниципального здравоохранения, принимает на лечение пациентов из большинства субъектов Российской Федерации и иностранных больных. Поэтому, довольно часто необходимо получать информацию о пролеченных больных в разрезе регионов с указанием прочих статистических данных, например, источника финансирования лечения, продолжительности лечения, половозрастного состава и т.д. однократный ввод и многократное использование первичной информации.

## Обзор средств разработки

### Языки 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 языков: C#, Ruby, Java, Javascript.

C# -- ОО (Объектно-ориентированный) 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);
* поддержку событийно-ретинированного программирования;
* унифицированную систему 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 сообщество разработчиков, которые помогают делать этот язык лучше. С GUI фреймворками как и у всех кроссплатформенных языков существуют определенные проблемы, тут и распространенный Qt, и wxWidgets, и используемый реже AWT на основе которого реализовано множество других фреймворков графического интерфейса для java.

К недостаткам java можно 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 операцию освобождения памяти.

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 русскоязычное описание нюансов.

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

Клиент-серверная СУБД позволяет обмениваться клиенту и серверу минимально необходимыми объёмами информации. При этом основная вычислительная нагрузка ложится на сервер. Клиент может выполнять функции предварительной обработки перед передачей информации серверу, но в основном его функции заключаются в организации доступа пользователя к серверу.[[2]](#footnote-2)

В большинстве случаев клиент-серверная СУБД гораздо менее требовательна к пропускной способности компьютерной сети, чем файл-серверная СУБД, особенно при выполнении операции поиска в базе данных по заданным пользователем параметрам, т.к. для поиска нет необходимости получать на клиент весь массив данных: клиент передаёт параметры запроса серверу, а сервер производит поиск по полученному запросу в локальной базе данных. Результат выполнения запроса, который обычно на несколько порядков меньше по объёму, чем весь массив данных, возвращается клиенту, который обеспечивает отображение результата пользователю.

## 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 разработки приложения. Программисту больше не нужно заботится о реализации доступа к базам данных.

## Структура базы данных МИС САМСОН

Медицинская информационная система «САМСОН» использует для хранения три базы данных: s11, mes, kladr.

База данных s11 содержит информацию о пролеченных пациентах, их адреса, документы, связи с другими пациентами, этапы лечения, действия, совершаемые над пациентом и другие данные. Также база s11 содержит информацию о персонале учреждения, структуру учреждения, заключенные контракты с другими учреждениями, патенты, и огромное количество справочников.

База s11 содержит 350 таблиц. В тексте данной работы структура показана не будет, но она содержится в файле s11\_strict.pdf который приложен к данной работе. Основные таблицы приведены на рисунке (Рисунок 1).

Таблица Client содержит информацию о пациентах, которые когда-либо обращались в ЛПУ.

Таблица OrgStructure содержит информацию о подразделениях, участках и отделениях внутри учреждения.

Таблица Event содержит события, связанные с обращениями пациента (плановое поступление, экстренное поступление, стоматология, проф. осмотр, обращение в регистратуру, запись на прием и.др.).

Таблица Action содержит действия, происходящие после обращения пациента (забор крови, перевод в другое отделение, рентгенография, различные операции и анестезии, и многое другое).

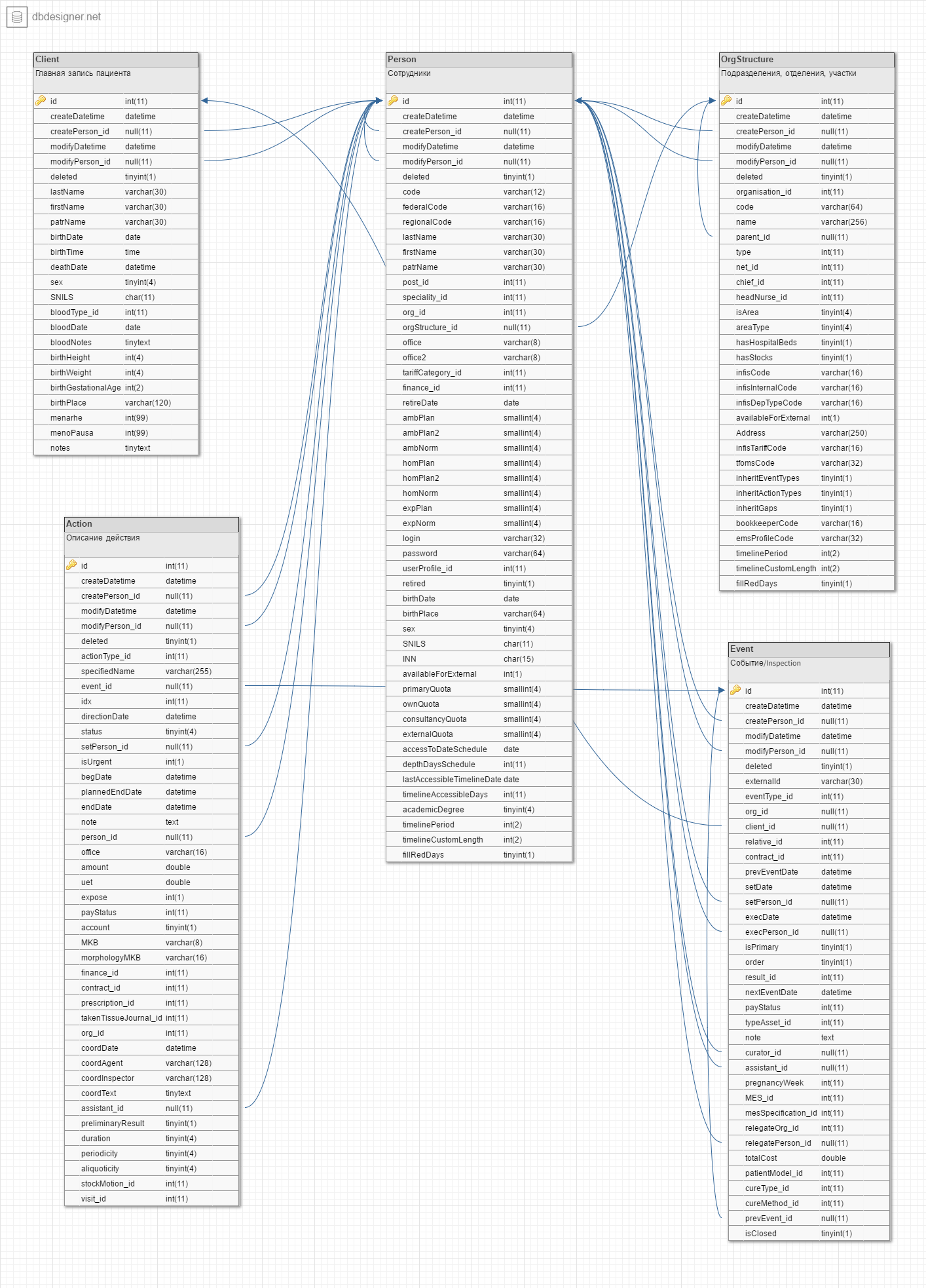


Рисунок 4 - Схема данных s11

Для хранения справочника адресов используется бд KLADR, состоящая из 14 таблиц: ALTNAMES, DOMA, eisHouse, FLAT, infisAREA, infisREGION, infisSTREET, infisSTREETYP, infisToEis, KLADR, OKATO, SOCRBASE, STREET, tmpOKATO. Таблицы хранятся в СУБД MyISAM, в которой нет поддержки внешних ключей, поэтому на схеме данных нет связей (Рисунок 2). Связи в данной таблице организуются только логически, по полю CODE, которое содержит строку с кодом, например, AABBBBBBCCCCCCCCDDDDE. Где AA – код региона, BBBBBB – код населённого пункта, CCCCCCCC – код улицы, DDDD – код домов, E – незначащая цифра, защищающая поле NAME

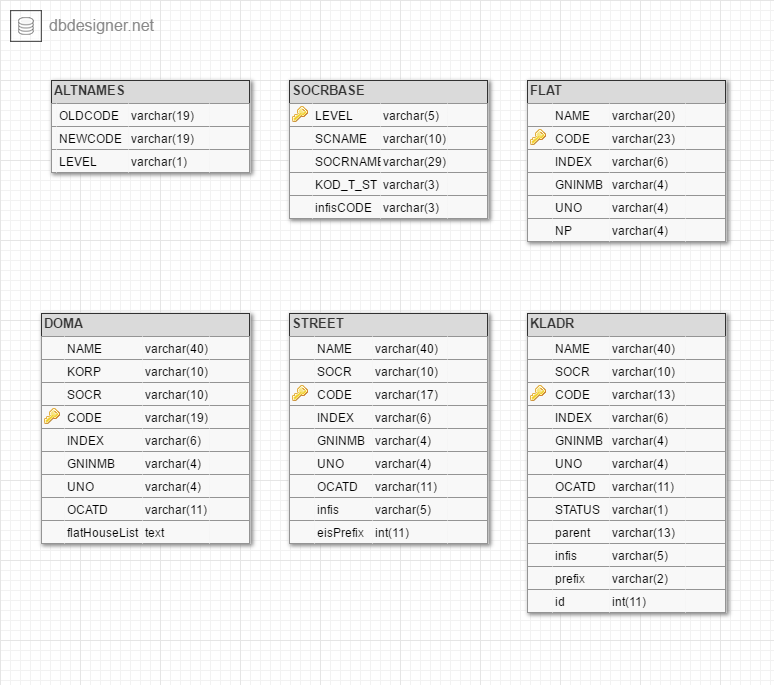


Рисунок 5 - Схема данных KLADR

Таблица ALTNAMES (Таблица 1) содержит сведения о изменении кодов.

Таблица 1 - kladr.ALTNAMES

|  |  |  |
| --- | --- | --- |
| Столбец | Тип | Комментарии |
| OLDCODE | varchar(19) | Старый код |
| NEWCODE | varchar(19) | Новый код |
| LEVEL | varchar(1) | Уровень объекта |

Таблица DOMA (Таблица 2) содержит записи с объектами номеров домов, улиц городов и населенных пунктов.

Таблица 2 - kladr.DOMA

|  |  |  |
| --- | --- | --- |
| Столбец | Тип | Комментарии |
| NAME | varchar(40) | Номера домов, владений (в виде списка и/или диапазонов) |
| KORP | varchar(10) | Корпус дома |
| SOCR | varchar(10) | Сокращенное наименование типа объекта |
| CODE | varchar(19) | Код |
| INDEX | varchar(6) | Почтовый индекс |
| GNINMB | varchar(4) | Код ИФНС (ИМНС) |
| UNO | varchar(4) | Код территориального участка ИФНС |
| OCATD | varchar(11) | Код ОКАТО |
| flatHouseList | text | Список домов |

Таблица FLAT (Таблица 3) содержит номера квартир.

Таблица 3 - kladr.FLAT

|  |  |  |
| --- | --- | --- |
| Столбец | Тип | Комментарии |
| NAME | varchar(20) | Номера квартир (в виде списка и/или диапазонов) |
| CODE | varchar(23) | Код |
| INDEX | varchar(6) | Почтовый индекс |
| GNINMB | varchar(4) | Код ИФНС (ИМНС) |
| UNO | varchar(4) | Код территориального участка ИФНС |
| NP | varchar(4) | Номер подъезда дома |

Таблица KLADR (Таблица 4) содержит районы, города, поселки городского типа, сельсоветы, сельские населенные пункты.

Таблица 4 - kladr.KLADR

|  |  |  |
| --- | --- | --- |
| Столбец | Тип | Комментарии |
| NAME | varchar(40) | Название населенного пункта |
| SOCR | varchar(10) | Сокращение |
| CODE | varchar(13) |  |
| INDEX | varchar(6) |  |
| GNINMB | varchar(4) | Номер налоговой |
| UNO | varchar(4) | код участка |
| OCATD | varchar(11) | Код по ОКАТО |
| STATUS | varchar(1) | Статус месности |
| parent | varchar(13) | ид родителя |
| infis | varchar(5) |  |
| prefix | varchar(2) | нормер региона |
| id | int(11) |  |

Таблица SOCRBASE (Таблица 5) содержит записи с краткими именами типов адресных объектов.

Таблица 5 - kladr.SOCRBASE

|  |  |  |
| --- | --- | --- |
| Столбец | Тип | Комментарии |
| LEVEL | varchar(5) | Уровень объекта данного типа |
| SCNAME | varchar(10) | Сокращенное наименование типа объекта |
| SOCRNAME | varchar(29) | Полное наименование типа объекта |
| KOD\_T\_ST | varchar(3) | Код типа объекта |
| infisCODE | varchar(3) |  |

Таблица STREET (Таблица 6) содержит улицы городов и населенных пунктов.

Таблица 6 - kladr.STREET

|  |  |  |
| --- | --- | --- |
| Столбец | Тип | Комментарии |
| NAME | varchar(40) | Наименование |
| SOCR | varchar(10) | Сокращенное наименование типа объекта |
| CODE | varchar(17) |  |
| INDEX | varchar(6) | Почтовый индекс |
| GNINMB | varchar(4) | Код ИФНС |
| UNO | varchar(4) | Код территориального участка ИФНС |
| OCATD | varchar(11) | Код ОКАТО |
| infis | varchar(5) |  |
| eisPrefix | int(11) |  |

База данных MES содержит значения медико-экономических стандартов. Таблицы базы данных стандартов медицинской помощи: MES, MES\_bloodPreparation, MES\_equipment, MES\_ksg, MES\_limitedBySexAge, MES\_medicament, MES\_mkb, MES\_nutrient, MES\_service, MES\_visit, ModelDescription, mrbBloodPreparation, mrbBloodPreparationType, mrbEquipment, mrbEquipmentGroup, mrbMedicament, mrbMedicamentDosageForm, mrbMedicamentGroup, mrbMESGroup, mrbModelAgeGroup, mrbModelAidCase, mrbModelAidPurpose, mrbModelCategory, mrbModelContinuation, mrbModelDiseaseClass, mrbModelExpectedResult, mrbModelInstitutionType, mrbModelSertificationRequirement, mrbModelStateBadness, mrbNutrient, mrbNutrientGroup, mrbService, mrbServiceGroup, mrbService\_Contents, mrbSpeciality, mrbVisitType. Структура Рисунок 3

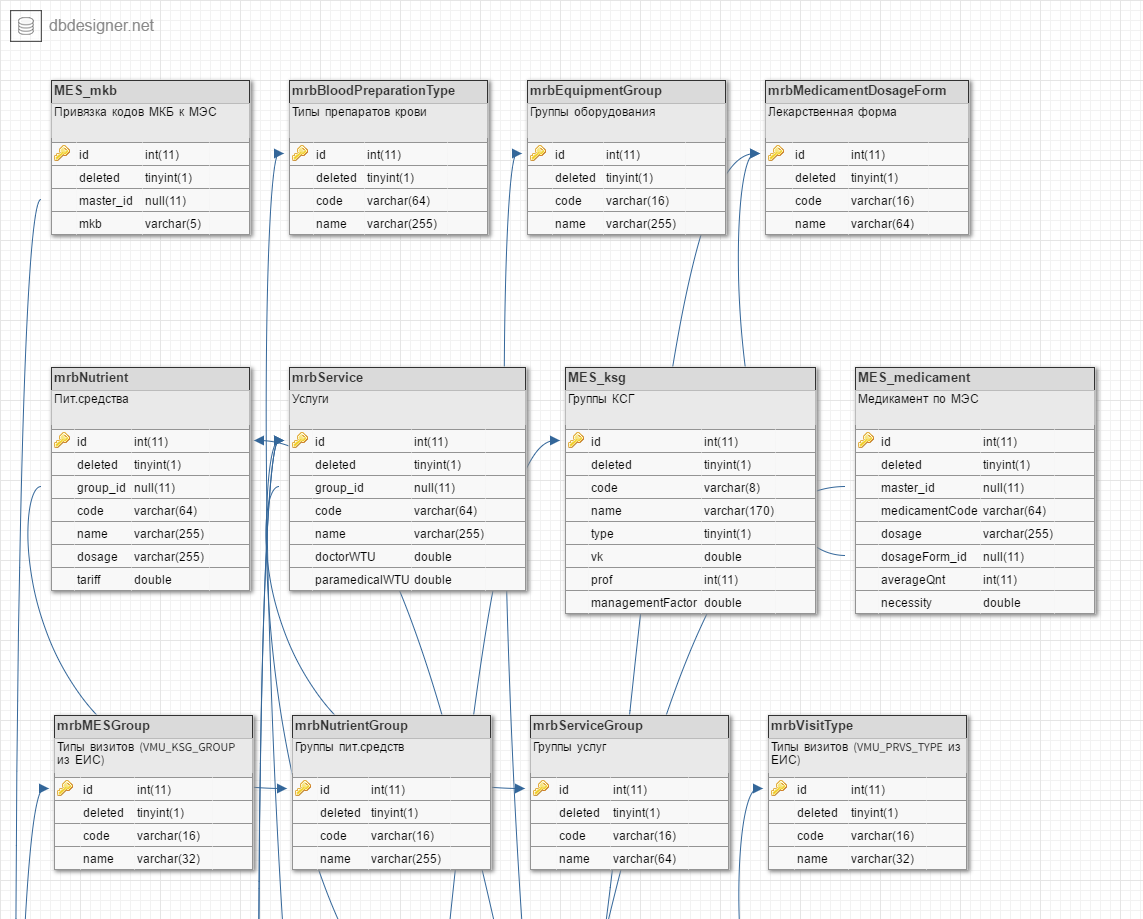


Рисунок 6 - Схема данных MES

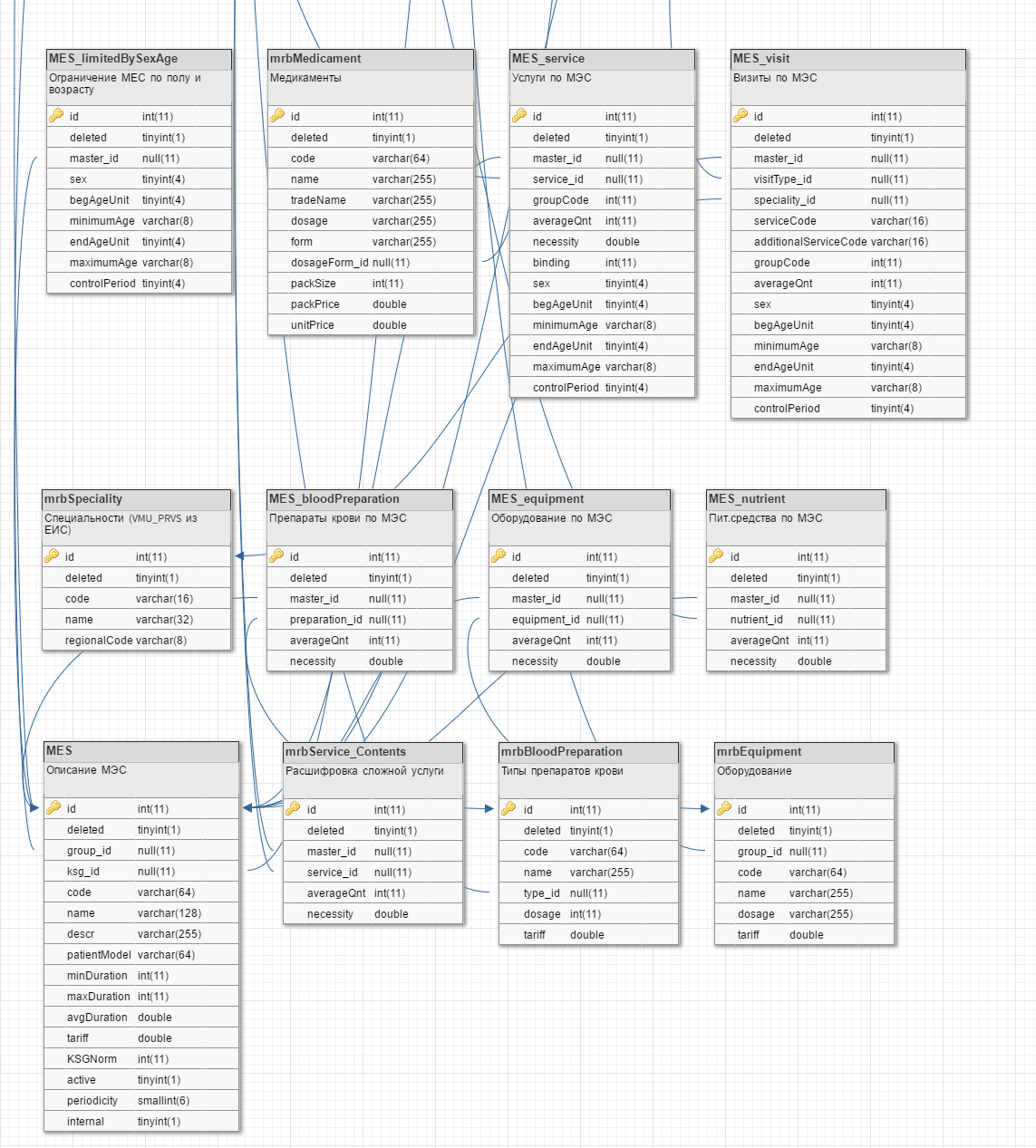


Рисунок 7 - Схема данных MES (продолжение)

В таблице MES (Таблица 7) хранятся описания МЭС.

Таблица 7 - mes.MES

|  |  |  |
| --- | --- | --- |
| **Столбец** | **Тип** | **Комментарии** |
| id | int(11) |  |
| deleted | tinyint(1) |  |
| group\_id | int(11) | {mrbMESGroup} |
| ksg\_id | int(11) | Группа КСГ |
| code | varchar(64) | Код |
| name | varchar(128) | Наименование |
| descr | varchar(255) | Описание |
| patientModel | varchar(64) | Модель пациента |
| minDuration | int(11) | Минимальная длительность |
| maxDuration | int(11) | Максимальная длительность |
| avgDuration | double | Средняя длительность |
| tariff | double | Тариф |
| KSGNorm | int(11) | Норматив визитов |
| active | tinyint(1) |  |
| periodicity | smallint(6) | Периодичность в днях |
| internal | tinyint(1) | является внутренним |

В таблице MES\_bloodPreparation (Таблица 8) хранится описание препаратов крови по МЭС

Таблица 8 - mes.MES\_bloodPreparation

|  |  |  |
| --- | --- | --- |
| **Столбец** | **Тип** | **Комментарии** |
| id | int(11) |  |
| deleted | tinyint(1) |  |
| master\_id | int(11) | Применяется для МЭС {MES} |
| preparation\_id | int(11) | Препарат {mrbBloodPreparation} |
| averageQnt | int(11) | среднее число препарата на курс лечения |
| necessity | double | частота назначения препарата в рамках стандарта |

В таблице MES\_equipment (Таблица 9) содержится описание оборудования по МЭС

Таблица 9 - mes.MES\_equipment

|  |  |  |
| --- | --- | --- |
| **Столбец** | **Тип** | **Комментарии** |
| id | int(11) |  |
| deleted | tinyint(1) |  |
| master\_id | int(11) | Применяется для МЭС {MES} |
| equipment\_id | int(11) | Оборудование {mrbEquipment} |
| averageQnt | int(11) | среднее число единиц препарата на курс лечения |
| necessity | double | частота назначения препарата в рамках стандарта |

В таблице MES\_ksg (Таблица 10) содержится описание клинико-статистических групп.

Таблица 10 - mes.MES\_ksg

|  |  |  |
| --- | --- | --- |
| **Столбец** | **Тип** | **Комментарии** |
| id | int(11) |  |
| deleted | tinyint(1) | Отметка удаления записи |
| code | varchar(8) | Код КСГ |
| name | varchar(170) | Наименование КСГ |
| type | tinyint(1) | Тип КСГ 0-Неизвестный,1-Тер.,2-Комб.,3-Хир.,4-Проч. |
| vk | double |  |
| prof | int(11) |  |
| managementFactor | double | Управленческий коэффициент |

В таблице MES\_limitedBySexAge (Таблица 11) содержится ограничения медико-экономических стандартов по полу и возрасту.

Таблица 11 - mes.MES\_limitedBySexAge

|  |  |  |
| --- | --- | --- |
| **Столбец** | **Тип** | **Комментарии** |
| id | int(11) |  |
| deleted | tinyint(1) | Отметка удаления записи |
| master\_id | int(11) | МЕС {MES} |
| sex | tinyint(4) | Пол (0-неопределено, 1-М, 2-Ж) |
| begAgeUnit | tinyint(4) | Применимо для минимального возраста (0-любой, 1-Д, 2-Н, 3-М, 4-Г) |
| minimumAge | varchar(8) | Минимальный возраст (yy-mm-dd) |
| endAgeUnit | tinyint(4) | Применимо для максимального возраста (0-любой, 1-Д, 2-Н, 3-М, 4-Г) |
| maximumAge | varchar(8) | Максимальный возраст (yy-mm-dd) |
| controlPeriod | tinyint(4) | Период контроля{0-Текущая дата,1-Конец текущего года,2-Конец предыдущего года} |

В таблице MES\_medicament (Таблица 12) содержится медикаменты применяемые для определенных МЭС.

Таблица 12 - mes.MES\_medicament

|  |  |  |
| --- | --- | --- |
| **Столбец** | **Тип** | **Комментарии** |
| id | int(11) |  |
| deleted | tinyint(1) |  |
| master\_id | int(11) | применяется для МЭС {MES} |
| medicamentCode | varchar(64) | Код |
| dosage | varchar(255) | дозировка |
| dosageForm\_id | int(11) | Лекарственная форма {mrbMedicamentDosageForm} |
| averageQnt | int(11) | СЧЕ - среднее число единиц медикамента (таблеток, ампул и т.д.) на курс лечения в стационаре |
| necessity | double | ЧН - частота назначения медикамента (потребность) в рамках настоящего стандарта |

В таблице mes.MES\_mkb (Таблица 13) содержится привязки кодов МКБ к МЭС.

Таблица 13- mes.MES\_mkb

|  |  |  |
| --- | --- | --- |
| **Столбец** | **Тип** | **Комментарии** |
| id | int(11) |  |
| deleted | tinyint(1) |  |
| master\_id | int(11) | применяется для МЭС {MES} |
| mkb | varchar(5) | Код диагноза по МКБ |

В таблице MES\_nutrient (Таблица 14) содержаться информация о питательных средствах по МЭС.

Таблица 14 - mes.MES\_nutrient

|  |  |  |
| --- | --- | --- |
| **Столбец** | **Тип** | **Комментарии** |
| id | int(11) |  |
| deleted | tinyint(1) |  |
| master\_id | int(11) | Применяется для МЭС {MES} |
| nutrient\_id | int(11) | Пит.средство {mrbNutrient} |
| averageQnt | int(11) | СЧЕ - среднее число единиц препарата на курс лечения |
| necessity | double | ЧН - частота назначения препарата в рамках настоящего стандарта |

В таблице mrbNutrient (Таблица 15) содержится информация о питательных средствах.

Таблица 15 - mes.mrbNutrient

|  |  |  |
| --- | --- | --- |
| **Столбец** | **Тип** | **Комментарии** |
| id | int(11) |  |
| deleted | tinyint(1) |  |
| group\_id | int(11) | Группа {mrbNutrientGroup} |
| code | varchar(64) | Код |
| name | varchar(255) | Наименование |
| dosage | varchar(255) | дозировка |
| tariff | double | Стоимость |

В таблице mrbService (Таблица 16) содержатся услуги.

Таблица 16 - mes.mrbService

|  |  |  |
| --- | --- | --- |
| **Столбец** | **Тип** | **Комментарии** |
| id | int(11) |  |
| deleted | tinyint(1) |  |
| group\_id | int(11) | Группа услуг {mrbServiceGroup} |
| code | varchar(64) | Код |
| name | varchar(255) | Наименование |
| doctorWTU | double | УЕТвр - усл. единицы трудозатрат врача |
| paramedicalWTU | double | УЕТср - усл. единицы трудозатрат сред. перс. |

В таблице mrbService\_Contents (Таблица 17) содержаться расшифровки для сложных услуг.

Таблица 17 - mes.mrbService\_Contents

|  |  |  |
| --- | --- | --- |
| **Столбец** | **Тип** | **Комментарии** |
| id | int(11) |  |
| deleted | tinyint(1) |  |
| master\_id | int(11) | Главная услуга {mrbService} |
| service\_id | int(11) | Часть услуги {mrbService} |
| averageQnt | int(11) | СЧЕ - среднее число единиц на курс лечения |
| necessity | double | ЧН - частота назначения (потребность) в рамках настоящего стандарта |

В таблице MES\_service (Таблица 18) содержаться услуги по МЭС.

Таблица 18 - mes.MES\_service

|  |  |  |
| --- | --- | --- |
| **Столбец** | **Тип** | **Комментарии** |
| id | int(11) |  |
| deleted | tinyint(1) |  |
| master\_id | int(11) | применяется для МЭС {MES} |
| service\_id | int(11) | услуга {mrbService} |
| groupCode | int(11) | Группировка услуг |
| averageQnt | int(11) | СК - среднее кол-во предоставления услуги (кратность) пациенту за период госпитализации |
| necessity | double | ЧП - частота предоставления услуги (потребность) в рамках настоящего стандарта |
| binding | int(11) | Пр.об. - признак объединения услуг в единую технологическую совокупность |
| sex | tinyint(4) | Пол (0-не определено, 1-М, 2-Ж) |
| begAgeUnit | tinyint(4) | Применимо для минимального возраста (0-любой, 1-Д, 2-Н, 3-М, 4-Г) |
| minimumAge | varchar(8) | Минимальный возраст (yy-mm-dd) |
| endAgeUnit | tinyint(4) | Применимо для максимального возраста (0-любой, 1-Д, 2-Н, 3-М, 4-Г) |
| maximumAge | varchar(8) | Максимальный возраст (yy-mm-dd) |
| controlPeriod | tinyint(4) | Период контроля{0-Текущая дата,1-Конец текущего года,2-Конец предыдущего года} |

В таблице MES\_visit (Таблица 19) содержаться визиты по МЭС

Таблица 19 - mes.MES\_visit

|  |  |  |
| --- | --- | --- |
| **Столбец** | **Тип** | **Комментарии** |
| id | int(11) |  |
| deleted | tinyint(1) |  |
| master\_id | int(11) | применяется для МЭС {MES} |
| visitType\_id | int(11) | Тип визита {mrbVisitType} |
| speciality\_id | int(11) | Специальность {mrbSpeciality} |
| serviceCode | varchar(16) | Код услуги |
| additionalServiceCode | varchar(16) | Дополнительный код услуги (ID\_SERVICE\_DATA) |
| groupCode | int(11) | Группировка визитов |
| averageQnt | int(11) | СК - среднее кол-во визитов (кратность) за период госпитализации |
| sex | tinyint(4) | Пол (0-не определено, 1-М, 2-Ж) |
| begAgeUnit | tinyint(4) | Применимо для минимального возраста (0-любой, 1-Д, 2-Н, 3-М, 4-Г) |
| minimumAge | varchar(8) | Минимальный возраст (yy-mm-dd) |
| endAgeUnit | tinyint(4) | Применимо для максимального возраста (0-любой, 1-Д, 2-Н, 3-М, 4-Г) |
| maximumAge | varchar(8) | Максимальный возраст (yy-mm-dd) |
| controlPeriod | tinyint(4) | Период контроля{0-Текущая дата,1-Конец текущего года,2-Конец предыдущего года} |

В таблице ModelDescription (Таблица 20) содержится описание моделей.

Таблица 20 - mes.ModelDescription

|  |  |  |
| --- | --- | --- |
| Столбец | Тип | Комментарии |
| id | int(11) |  |
| idx | int(11) | индекс для сортировки |
| name | varchar(64) | название |
| fieldIdx | int(11) | индекс в split(".") |
| tableName | varchar(64) | Название таблицы |

В таблице mrbBloodPreparation (Таблица 21) содержаться типы препаратов крови

Таблица 21 - mes.mrbBloodPreparation

|  |  |  |
| --- | --- | --- |
| **Столбец** | **Тип** | **Комментарии** |
| id | int(11) |  |
| deleted | tinyint(1) |  |
| code | varchar(64) | Код |
| name | varchar(255) | название |
| type\_id | int(11) | Тип препарата {mrbBloodPreparationType} |
| dosage | int(11) | Дозировка |
| tariff | double | Тариф |

В таблице mrbEquipment (Таблица 22) содержится оборудование.

Таблица 22 - mes.mrbEquipment

|  |  |  |
| --- | --- | --- |
| **Столбец** | **Тип** | **Комментарии** |
| id | int(11) |  |
| deleted | tinyint(1) |  |
| group\_id | int(11) | Группа {mrbEquipmentGroup} |
| code | varchar(64) | Код |
| name | varchar(255) | Наименование |
| dosage | varchar(255) | дозировка |
| tariff | double | Стоимость |

В таблице mrbMedicament (Таблица 23) содержаться информация о медикаментах.

Таблица 23 - mes.mrbMedicament

|  |  |  |
| --- | --- | --- |
| **Столбец** | **Тип** | **Комментарии** |
| id | int(11) |  |
| deleted | tinyint(1) |  |
| code | varchar(64) | Код |
| name | varchar(255) | МНН - международное непатентованное название медикамента |
| tradeName | varchar(255) | Торговое наименование лек.средства |
| dosage | varchar(255) | дозировка |
| form | varchar(255) | форма выпуска |
| dosageForm\_id | int(11) | Лекарственная форма {mrbMedicamentDosageForm} |
| packSize | int(11) | Количество единиц в упаковке |
| packPrice | double | Стоимость упаковки |
| unitPrice | double | Стоимость единицы |

Таблицы mrbMedicamentGroup, mrbModelAgeGroup, mrbModelAidCase, mrbModelAidPurpose, mrbModelCategory, mrbModelContinuation, mrbModelDiseaseClass, mrbModelExpectedResult, mrbModelInstitutionType, mrbModelSertificationRequirement, mrbModelStateBadness, mrbNutrientGroup, mrbServiceGroup, mrbVisitType, mrbMESGroup, mrbEquipmentGroup, mrbBloodPreparationType, mrbMedicamentDosageForm имеют структуру аналогичную MES\_ksg (Таблица 10).

# 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.

Данный проект создается для заведующих отделений и отдела мониторинга высокотехнологичной медицинской помощи, 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

Модулем «Статистика» может пользоваться любой пользователь, имеющий на своём компьютере приложение «Самсон-виста», так как модуль «Статистика» получает доступ к базе данных из настроек «САМСОН». Общий вид приложения представлен на рисунке (Рисунок 5).

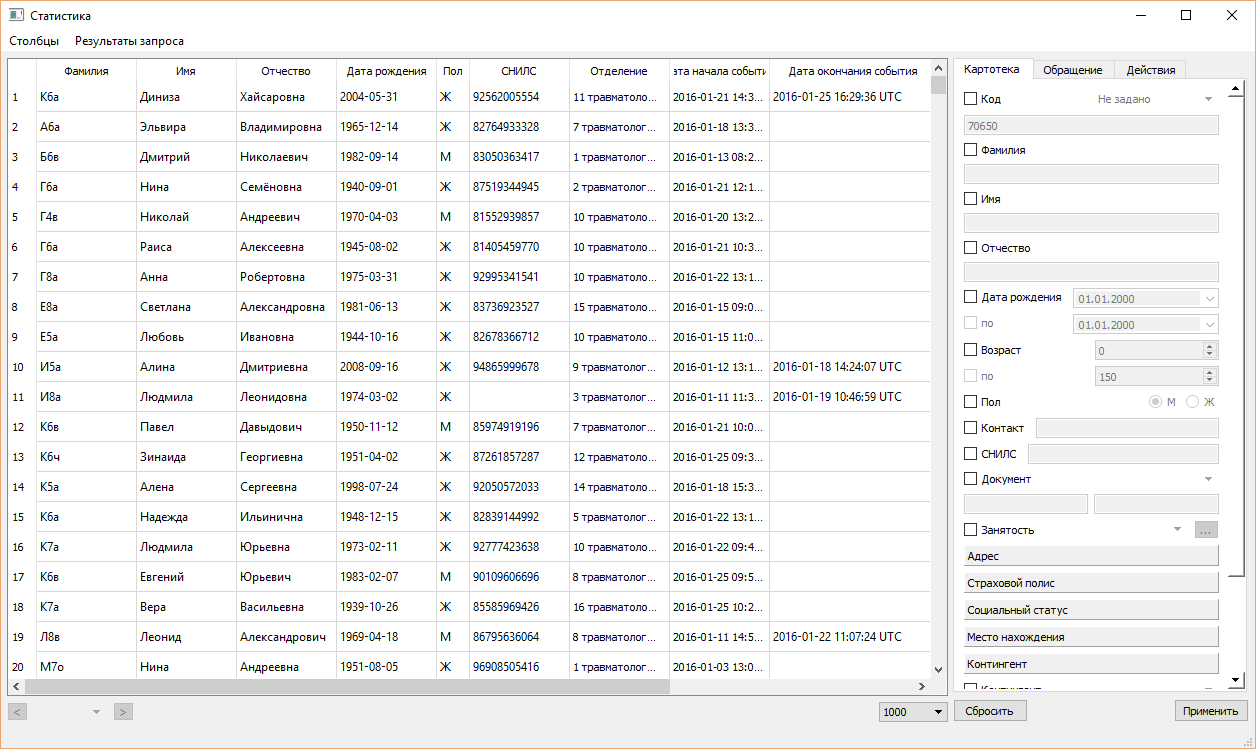


Рисунок 8 - Общий вид модуля "статистика"

При запуске приложения пользователю необходимо ввести свои учетные данные соответствующие учетным данным из системы «САМСОН» (Рисунок 6).

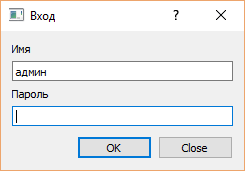


Рисунок 9 - Авторизация

При вводе неверных данных будет выдано соответствующее сообщение (Рисунок 7)

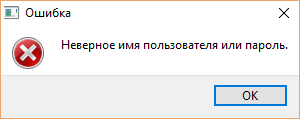


Рисунок 10 - Ошибка авторизации

В приложении есть возможность задавать необходимые пользователю столбцы (Рисунок 18). При выборе нужного столбца он отобразится после нажатия кнопки «Применить» находящейся в нижней правой части экрана.

После нажатия кнопки «Применить» выполнится запрос к базе данных, результат запроса будет отображен в главной таблице окна, а в строке состояния появится надпись о общем количестве записей, которые вернул запрос (Рисунок 11).

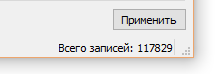


Рисунок 11 - Общее число записей запроса

Сортировка пациентов производится при нажатии на заголовок столбца который нужно отсортировать.

Также у пользователя есть возможность экспортировать таблицу в файл для последующего открытия в его любимом редакторе (Рисунок 8).

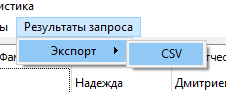


Рисунок 12 - Экспорт

Фильтрация пациентов производится при выборе фильтров в правой части окна.

При установке галочки в контроле «Код» становится доступна фильтрация по кодам пациента (Рисунок 9). У пациента может быть несколько кодов: его уникальный идентификатор в базе данных «САМСОН», фото номер, архивный номер и идентификаторы других систем учета пациентов.

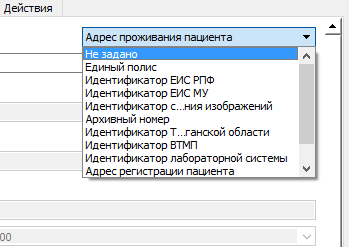


Рисунок 13 - Коды пациента

Фильтрация доступна и по всем основным значениям (Рисунок 18): имени, фамилии, отчеству, дате рождения, возрасту, полу, СНИЛС.

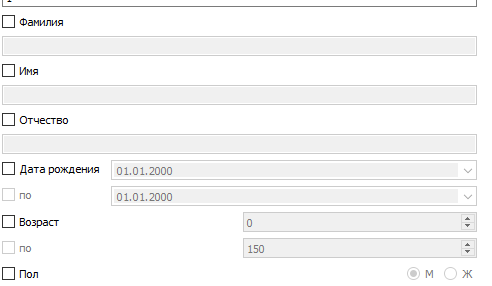


Рисунок 14 - Основные фильтры

Если у пациента внесены документы в базу, то его можно найти по ним (Рисунок 11).

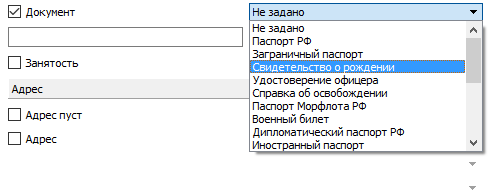


Рисунок 15 - Фильтрация по документам

Можно найти пациентов по определенной области, городу, району, улице, дому (Рисунок 12). Существует возможность выбора адреса регистрации или проживания.

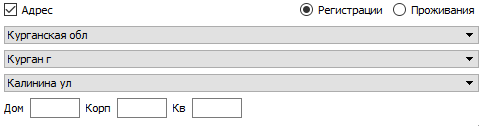


Рисунок 16 - Фильтр по адресам

Вкладка обращение позволяет фильтровать пациентов по их обращениям в ЛПУ (Рисунок 13).

Контрол «Тип» при первой установке галочки подгружает список типов событий (Рисунок 14).

Также на вкладке обращение можно указать даты назначения, выполнения и следующей явки (Рисунок 15).

После установки галочки в контроле «Подразделение» из базы данных будут загружены все подразделения и построена древовидная структура (Рисунок 16)

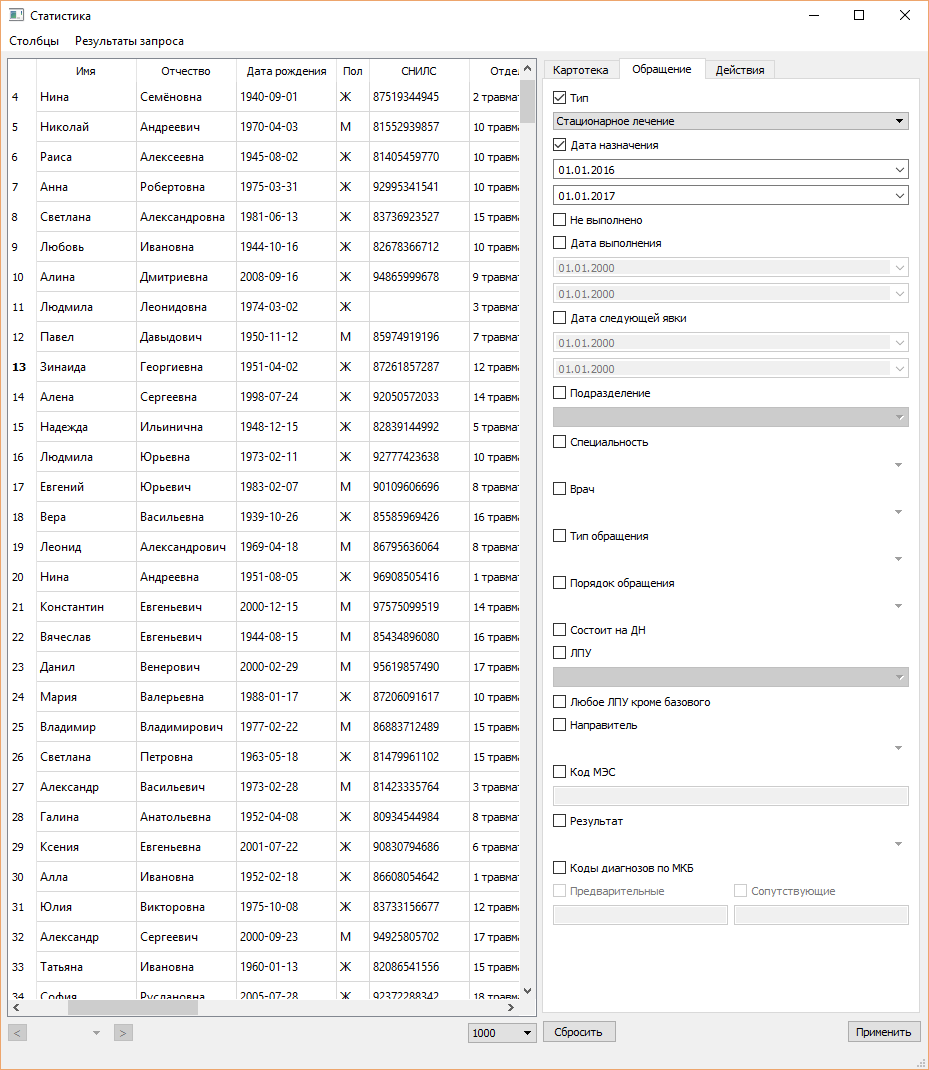


Рисунок 17 - Вкладка обращение

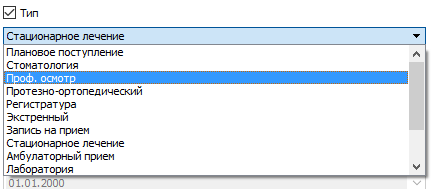


Рисунок 18 - Типы обращений

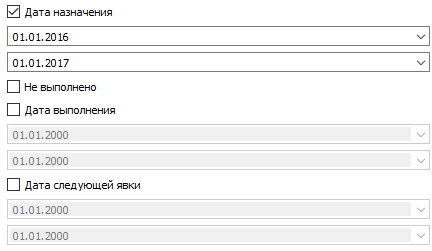


Рисунок 19 - Обращение даты

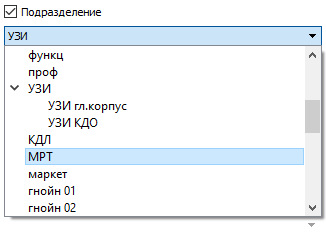


Рисунок 20 - Обращение, подразделение

Вкладка действия содержит фильтры для поиска действий совершаемых над пациентом

Тип действия отвечает за выбор действия над пациентом, основная группировка типов действия это: «Статус», «Лечение», «Диагностика» и «Прочие мероприятия». После выбора одного из типов из базы данных подгружается более подробные фильтры (Рисунок 17).

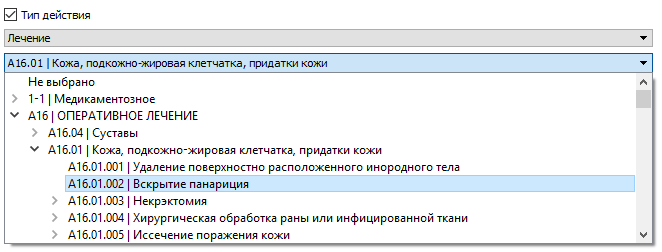


Рисунок 21 - Действия, типы

Подразделение назначившего, специальность назначившего и назначил на вкладке действий позволяет находить всех пациентов у определенных врачей.

Коды диагнозов по МКБ позволяют фильтровать результаты поиска по кодам диагнозов.

Кнопка «Сброс» выключает все фильтры и устанавливает их значения по умолчанию.

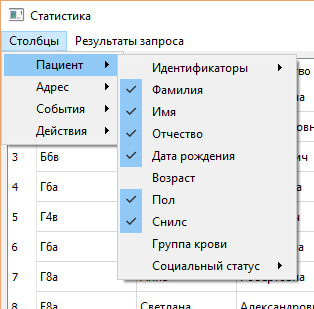


Рисунок 22 - Выбор столбцов для отображения

В нижней части таблицы есть возможность выбора количества отображения строк (Рисунок 19).

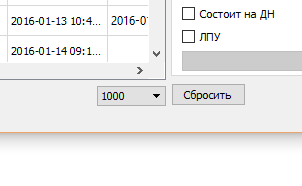


Рисунок 23 - Сколько строк нужно отобразить

Если в результате запроса строк для отображения больше чем указано в настройках, то активируется панель с нумерацией страниц, которая позволяет либо переходить на прошлую или следующую страницы, либо выбрать произвольную (Рисунок 23).

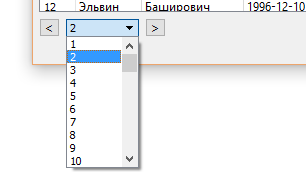


Рисунок 24 - Постраничная навигация

У пользователя есть возможность поменять размер таблицы, сделать её больше или меньше за счет уменьшения области фильтров. Таблицу с результатами или область фильтров можно скрыть совсем если что-то из них в данный момент ненужно (Рисунок 20).

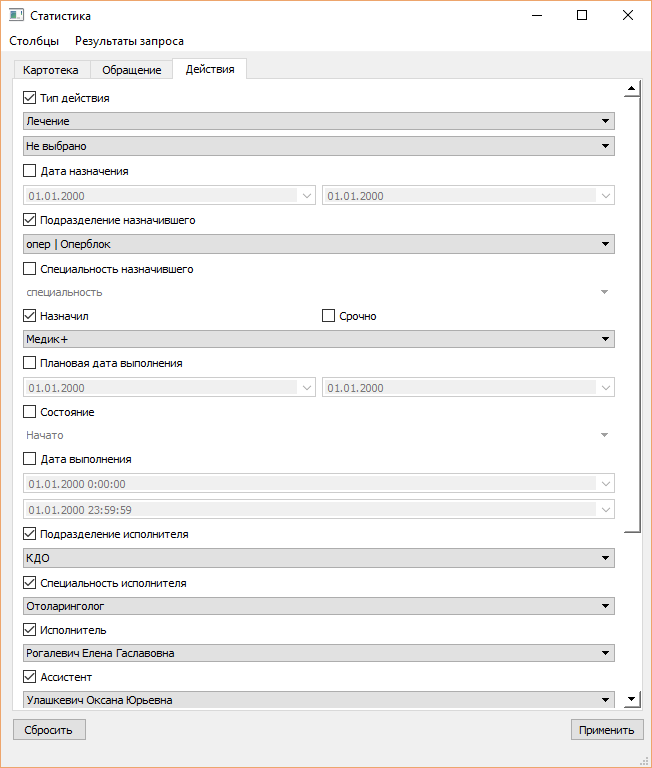


Рисунок 25 - Скрытие таблицы результатов

При закрытии приложения настройки отображения сохраняются в файл настроек находящийся в папке с приложением. Настройками отображения являются размещение и размер окна, флаг о полноэкранном режиме, настройки, связанные с отображением страниц и отображаемые столбцы. Во время открытия приложения сохраненные настройки считываются из файла и применяются к пользовательскому интерфейсу.

## Расчеты экономической эффективности использования модуля

При разработке модуля вывода статистики информационной системы «САМСОН» были задействованы два человека: студент, пишущий диплом и руководитель дипломного проекта.

Студент в проекте отвечает за анализ предметной области, разработку внутренних алгоритмов модуля, проектирование интерфейса и отладку программы. Руководитель занимается контролем хода работ и консультирует студента при необходимости. Этапы разработки приведены в таблице (Таблица 24).

Трудоемкость работ определялась с учетом срока окончания работ, объемом выполняемых функций. Выбор комплекса работ по разработке проекта производится в соответствии с ГОСТ 19.102-77 «Единая система программной документации», устанавливающего стадии разработки.

Таблица 24 - Этапы разработки

|  |  |  |  |
| --- | --- | --- | --- |
| Стадия разработки | Длительность работ (в днях) | | |
| минимум | максимум | ожид. |
| 1. Техническое задание | | | |
| * 1. Обоснование необходимости разработки программы | | | |
| * + 1. Постановка задачи | 1 | 2 | 1 |
| * + 1. Сбор исходных материалов | 2 | 3 | 2,4 |
| * + 1. Выбор и обоснование критериев эффективности качества разрабатываемой программы | 11 | 2 | 7,4 |
| * + 1. Обоснование необходимости проведения научно-исследовательских работ | 1 | 1 | 1 |
| * 1. Научно-исследовательские работы | | | |
| * + 1. Определение структуры входных и выходных данных. | 3 | 4 | 3,4 |
| * + 1. Предварительный выбор методов решения задач. | 10 | 15 | 12 |
| * + 1. Обоснование целесообразности применения ранее разработанных программ | 2 | 3 | 2,4 |
| * + 1. Определение требований к техническим средствам. | 1 | 2 | 1,4 |
| * + 1. Обоснование принципиальной возможности решения поставленной задачи | 1 | 1 | 1 |
| * 1. Разработка и утверждение технического задания | | | |
| * + 1. Определение требований к программе. | 3 | 5 | 3,8 |
| * + 1. Разработка технико-экономического обоснования разработки программы. | 1 | 2 | 1,4 |
| * + 1. Определение стадий, этапов и сроков разработки программы и документации на неё. | 1 | 2 | 1,4 |
| * + 1. Выбор языков программирования. | 1 | 2 | 1,4 |
| * + 1. Определение необходимости проведения научно-исследовательских работ на последующих стадиях. | 1 | 2 | 1,4 |
| * + 1. Согласование и утверждение технического задания. | 1 | 2 | 1,4 |
| 1. Эскизный проект | | | |
| * 1. Разработка эскизного проекта | | | |
| * + 1. Предварительная разработка структуры входных и выходных данных. | 7 | 10 | 8,2 |
| * + 1. Уточнение методов решения задачи. | 2 | 3 | 2,4 |
| * + 1. Разработка общего описания алгоритма решения задачи | 10 | 12 | 11 |
| * + 1. Разработка технико-экономического обоснования. | 2 | 3 | 2,4 |
| * 1. Утверждение эскизного проекта | | | |
| * + 1. Разработка пояснительной записки. | 1 | 2 | 1,4 |
| * + 1. Согласование и утверждение эскизного проекта. | 1 | 2 | 1,4 |
| 1. Рабочий проект | | | |
| * 1. Разработка программы | | | |
| * + 1. Программирование и отладка программы. | 34 | 42 | 37 |
| * 1. Разработка программной документации | | | |
| * + 1. Разработка программных документов в соответствии с требованиями ГОСТ 19.101-77. | 2 | 3 | 2,4 |
| * 1. Испытания программы | | | |
| * + 1. Разработка, согласование и утверждение порядка и методики испытаний. | 1 | 2 | 1,4 |
| * + 1. Проведение предварительных государственных, межведомственных, приёмо-сдаточных и других видов испытаний. | 2 | 3 | 2,4 |
| * + 1. Корректировка программы и программной документации по результатам испытаний. | 2 | 2 | 2 |

Продолжительность работ определяется по формуле (Формула 1):

Формула 1 - Ожидаемая продолжительность работы

* -- ожидаемая продолжительность работ;
* -- соответственно наименьшая и наибольшая по мнению эксперта длительность работы.

Суммарные затраты на проектирование и разработку приложения определяются суммой всех затраченных средств, а именно затраты на использование машинного времени и расходы на материалы. Оклад разработчиков учитывать в данном проекте не будем.

Так как разработанный программный продукт должен быть создан на ПК, к суммарным затратам на разработку необходимо добавить и использование машинного времени, начисляемое по формуле (Формула 2).

Формула 2 - стоимость использования ЭВМ

Где,

* – машинное время компьютера, необходимое для разработки программного продукта;
* – стоимость 1 часа машинного времени.

Рассчитаем стоимость 1 часа использования ПК по формуле ():

Где,

* V – вольты;
* A – амперы;
* С – стоимость 1 кВт-ч

Учитывая время на проектирование приложения, представленное в таблице (Таблица 24) и стоимость разработку программного продукта на компьютере за один час, расходы будут равны.

Приведем расчет затрат на материалы используя данные представленные в таблице (Таблица 25).

Таблица 25 - Затраты на материалы

|  |  |  |  |
| --- | --- | --- | --- |
| Наименование | Количество шт. | Стоимость руб./ед. изм. | Общая стоимость (в рублях) |
| Печать черно-белых листов | 70 | 3 | 210 |
| CD-R диск | 1 | 40 | 40 |
| Итого |  |  | 250 |

Итого сумма затрат на разработку приложения будет составлять N.

Эксплуатационные расходы по созданному модулю отсутствуют, поэтому общая стоимость внедрения и разработки будет равна .

# Заключение

В ходе работы над проектом удалось достичь всех поставленных целей. Была проанализирована медицинская информационная система «САМСОН», выявлены проблемы связанные с выводом информации из неё. На основе этих проблем был разработан модуль «Статистика», который позволяет пользователю выбирать необходимые для отображения «Колонки» и устанавливать параметры поиска.

Благодаря разработанному модулю упрощено составление свободных отчетов по работе клинических отделений и стационара в целом, проведения анализа работы учреждения. Упрощена возможность переноса пациентов в архивную базу данных «Нозологический и функциональный каталог». Исключена необходимость заполнения «Журнала учета стационарных больных».

Интерфейс приложения получился простой и удобный для пользователя. Уровень подготовки пользователей должен соответствовать начальной компьютерной грамотности, предполагающей наличие умений и навыков работы с ПК.

# 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

<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>

1.3 Предпосылки разработки модуля "Статистика"

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>

1. Международная классификация болезней 10-го пересмотра [↑](#footnote-ref-1)
2. https://ru.wikipedia.org/wiki/Клиент-серверная\_СУБД [↑](#footnote-ref-2)