Оглавление

[Введение 2](#_Toc454137764)

[Архитектура 3](#_Toc454137765)

[Обзор технологий 5](#_Toc454137766)

[Client-Server 5](#_Toc454137767)

[WCF 5](#_Toc454137768)

[Microservices 5](#_Toc454137769)

[WPF 5](#_Toc454137770)

[RDBS 5](#_Toc454137771)

[ORM 6](#_Toc454137772)

[DI и IoC 6](#_Toc454137773)

[Обзор библиотек 8](#_Toc454137774)

[Stimulsoft 8](#_Toc454137775)

[NLog 8](#_Toc454137776)

[MashApp 8](#_Toc454137777)

[Ninject/Autofac/WinsdorCasstle 8](#_Toc454137778)

[EF 8](#_Toc454137779)

[MySQL/PostgreSQL 8](#_Toc454137780)

[История изменений 9](#_Toc454137781)

# Введение

В данном документе описывается архитектура проекта, рассказывается об основных концепциях, выбранных технологиях и библиотеках.

# Архитектура

Система делится на два компонента – сервер и клиент. Сервер отвечает за хранение информации, подключение пользователей. Клиентская часть осуществляет взаимодействие с оборудованием (кардиомонитором, кроватью). Через клиентское приложение так же осуществляется заведение информации о пользователях, проведение сеансов и управление ими. Для улучшения масштабирования предполагается разделение обязанностей по разным компонентам и разнесение функций по разным слоям:

* DAO (слой данных): на этом слое осуществляется получение и обработка данных;
* Слов бизнес-логики: на этом уровне будут проводиться все вычисления, построение отчетов.

В системе предусмотрена развитая система логирования. Это позволит сохранять как техническую информацию о неполадках, так и запоминать действия пользователей.

Общая схема архитектуры представлена на рисунке 1.

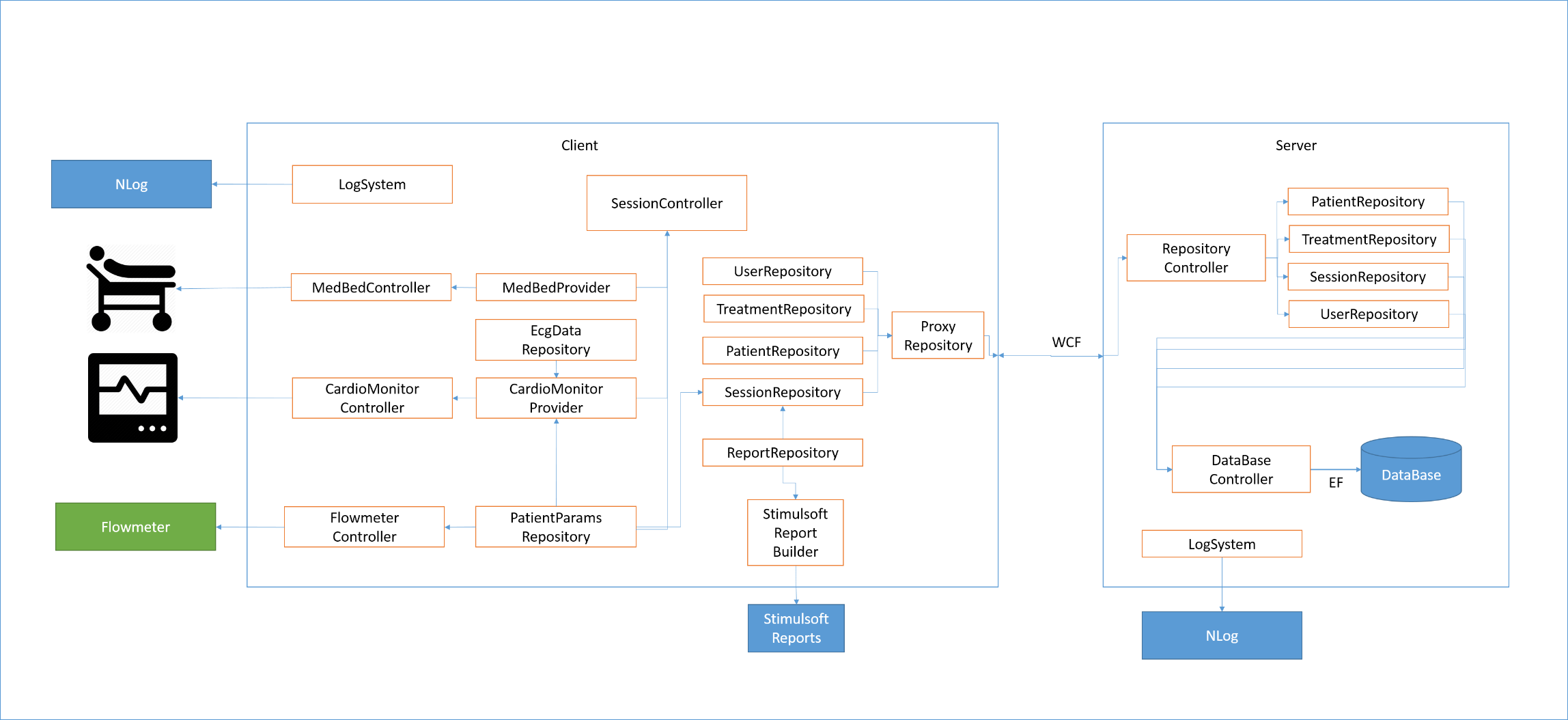


Рисунок 1. Архитектура системы

# Обзор технологий и концепций

## Client-Server

Клиент-сервер (англ. Client-server) — вычислительная или сетевая архитектура, в которой задания или сетевая нагрузка распределены между поставщиками услуг, называемыми серверами, и заказчиками услуг, называемыми клиентами. Физически клиент и сервер — это программное обеспечение. Обычно они взаимодействуют через компьютерную сеть посредством сетевых протоколов и находятся на разных вычислительных машинах, но могут выполняться также и на одной машине. Программы — сервера, ожидают от клиентских программ запросы и предоставляют им свои ресурсы в виде данных (например, загрузка файлов посредством HTTP, FTP, BitTorrent, потоковое мультимедиа или работа с базами данных) или сервисных функций (например, работа с электронной почтой, общение посредством систем мгновенного обмена сообщениями, просмотр web-страниц во всемирной паутине).

Преимущества:

* Отсутствие дублирования кода программы-сервера программами-клиентами;
* Так как все вычисления выполняются на сервере, то требования к компьютерам, на которых установлен клиент, снижаются;
* Все данные хранятся на сервере, который, как правило, защищён гораздо лучше большинства клиентов. На сервере проще организовать контроль полномочий, чтобы разрешать доступ к данным только клиентам с соответствующими правами доступа.

Недостатки:

* Неработоспособность сервера может сделать неработоспособной всю вычислительную сеть. Неработоспособным сервером следует считать сервер, производительности которого не хватает на обслуживание всех клиентов, а также сервер, находящийся на ремонте, профилактике и т. п.;
* Поддержка работы данной системы требует отдельного специалиста — системного администратора.

## WCF

## WPF

Windows Presentation Foundation (WPF) — система для построения клиентских приложений Windows с визуально привлекательными возможностями взаимодействия с пользователем, графическая (презентационная) подсистема в составе .NET Framework (начиная с версии 3.0), использующая язык XAML.

В основе WPF лежит векторная система визуализации, не зависящая от разрешения устройства вывода и созданная с учётом возможностей современного графического оборудования. WPF предоставляет средства для создания визуального интерфейса, включая язык XAML (Extensible Application Markup Language), элементы управления, привязку данных, макеты, двухмерную и трёхмерную графику, анимацию, стили, шаблоны, документы, текст, мультимедиа и оформление.

Графической технологией, лежащей в основе WPF, является DirectX, в отличие от Windows Forms, где используется GDI/GDI+. Производительность WPF выше, чем у GDI+ за счёт использования аппаратного ускорения графики через DirectX.

WPF предоставляет широкий спектр возможностей по созданию интерактивных настольных приложений:

* Привязка данных;
* Стили;
* Шаблоны элементов управления;
* Шаблоны данных;
* Ресурсы.

## RDBS

База данных — представленная в объективной форме совокупность самостоятельных материалов (статей, расчётов, нормативных актов, судебных решений и иных подобных материалов), систематизированных таким образом, чтобы эти материалы могли быть найдены и обработаны с помощью электронной вычислительной машины (ЭВМ)

Реляционная база данных (англ. Relational database system) – это совокупность отношений, содержащих всю информацию, которая должна храниться в БД. Однако пользователи могут воспринимать такую базу данных как совокупность таблиц.

## ORM

ORM (англ. Object-Relational Mapping, рус. объектно-реляционное отображение) — технология программирования, которая связывает базы данных с концепциями объектно-ориентированных языков программирования, создавая «виртуальную объектную базу данных». Существуют как проприетарные, так и свободные реализации этой технологии.

В объектно-ориентированном программировании объекты в программе представляют объекты из реального мира. В качестве примера можно рассмотреть адресную книгу, которая содержит список людей с нулём или более телефонов и нулём или более адресов. В терминах объектно-ориентированного программирования они будут представляться объектами класса «Человек», которые будут содержать следующий список полей: имя, список (или массив) телефонов и список адресов.

Суть задачи состоит в преобразовании таких объектов в форму, в которой они могут быть сохранены в файлах или базах данных, и которые легко могут быть извлечены в последующем, с сохранением свойств объектов и отношений между ними. Эти объекты называют «хранимыми» (англ. persistent). Исторически существует несколько подходов к решению этой задачи.

С точки зрения программиста система должна выглядеть как постоянное хранилище объектов. Он может просто создавать объекты и работать с ними как обычно, а они автоматически будут сохраняться в реляционной базе данных.

На практике всё не так просто и очевидно. Все системы ORM обычно проявляют себя в том или ином виде, уменьшая в некотором роде возможность игнорирования базы данных. Более того, слой транзакций может быть медленным и неэффективным (особенно в терминах сгенерированного SQL). Все это может привести к тому, что программы будут работать медленнее и использовать больше памяти, чем программы, написанные «вручную».

Но ORM избавляет программиста от написания большого количества кода, часто однообразного и подверженного ошибкам, тем самым значительно повышая скорость разработки. Кроме того, большинство современных реализаций ORM позволяют программисту при необходимости самому жёстко задать код SQL-запросов, который будет использоваться при тех или иных действиях (сохранение в базу данных, загрузка, поиск и т. д.) с постоянным объектом.

## DI и IoC

Инверсия управления (англ. Inversion of Control, IoC) — важный принцип объектно-ориентированного программирования, используемый для уменьшения зацепления в компьютерных программах. Также архитектурное решение интеграции, упрощающее расширение возможностей системы, при котором контроль над потоком управления программы остаётся за каркасом.

Модули верхнего уровня не должны зависеть от модулей нижнего уровня. И те, и другие должны зависеть от абстракции.

Абстракции не должны зависеть от деталей. Детали должны зависеть от абстракций.

Одной из реализаций IoC в применении к управлению зависимостями является внедрение зависимостей (англ. dependency injection). Внедрение зависимости используется во многих фреймворках, которые называются IoC-контейнерами.

Если сравнить с более низкоуровневыми технологиями, IoC-контейнер — это компоновщик, который собирает не объектные файлы, а объекты ООП (экземпляры класса) во время исполнения программы. Очевидно, для реализации подобной идеи было необходимо создать не только сам компоновщик, но и фабрику, производящую объекты. Аналогом такого компоновщика (естественно, более функциональным) является компилятор, одной из функций которого является создание объектных файлов. В идее компоновки программы во время исполнения нет ничего нового. Предоставление программисту инструментов внедрения зависимостей дало значительно большую гибкость в разработке и удобство в тестировании кода

# Обзор библиотек

## Stimulsoft

Stimulsoft Reports.Ultimate - это комплексное решение для построения отчетности для платформы .NET Framework.

* Преимущества системы:
* Мощная система экспорта отчетов, которая поддерживает множество различных форматов.
* Простое и одновременно мощное ядро генератора отчетов.
* Простой и функциональный дизайнер отчетов.

## NLog

NLog является свободнойплатформой логгирования для .NET с богатыми маршрутизацией и управлением журналами. NLog может обрабатывать диагностические сообщения, испускаемые из любого языка .NET, дополнить их контекстной информацией (дата и время, степень тяжести, потоки, процессы, окружающая среда), формат определяется в соответствии с необходимыми предпочтениями и отправить в одну или несколько конечный целей.

Преимущества:

* Легкость конфигурирования;
* Поддержка шаблонов;
* Расширяемость.

## MahApps.Metro

Это инструментарий для создания приложений WPF в Metro-стиле. Много возможностей библиотека предоставляет в чистом виде.

## Autofac

Autofac является контейнером IoC для Microsoft .NET. Он управляет зависимостями между классами, так что остается возможность легкового внесения изменений. Это достигается путем обработки регулярных классов .NET в качестве компонентов.

## EF

Entity Framework (EF) — это объектно-реляционный модуль сопоставления, позволяющий разработчикам .NET работать с реляционными данными с помощью объектов, специализированных для доменов. Это устраняет необходимость в написания большей части кода для доступа к данным, который обычно требуется разработчикам

## PostgreSQL

PostgreSQL является мощной c открытым исходным кодом объектно-реляционной системой управления базами данных. Он имеет более чем 15 лет активного развития и проверенной архитектуре, которая заслужила ему высокую репутацию надежности, целостности данных и корректность. Она работает на всех основных операционных систем, включая Linux, UNIX (AIX, BSD, HP-UX, SGI IRIX, Mac OS X, Solaris, Tru64) и Windows. Она полностью соответствует ACID, имеет полную поддержку внешних ключей, объединений, представлений, триггеров и хранимых процедур (на нескольких языках). Она включает в себя большинство SQL: 2008 типы данных, в том числе INTEGER, числовыми, BOOLEAN, CHAR, VARCHAR, ДАТА, ИНТЕРВАЛ, и TIMESTAMP. Он также поддерживает хранение больших двоичных объектов, включая рисунки, звуки или видео. Он имеет собственные интерфейсы программирования для C / C ++, Java, .NET, Perl, Python, Ruby, Tcl, ODBC, среди прочего, и исключительной документации.

# История изменений

Таблица 1. История изменений документа

|  |  |  |  |
| --- | --- | --- | --- |
| № | Автор изменений | Дата | Комментарий |
| 1 | Маркелов М.А. | 17.06.2016 | Создан документ. Кратко описана архитектура, перечислены используемые технологии и библиотеки. |
| 2 |  |  |  |
| 3 |  |  |  |
|  |  |  |  |