МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение

высшего образования

«**Вятский государственный университет**»

Факультет автоматики и вычислительной техники

Кафедра электронных вычислительных машин

Отчет

Комплекс лабораторных работ по дисциплине

«Проектирование информационных систем»

Выполнил студент группы ИВТ-42 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/Никитин Н.Д./

Проверил доцент кафедры ЭВМ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/Караваева О.В./

Киров 2017

# Введение

С развитием информационных технологий, компьютеры с их расширенными функциональными возможностями, активно применяются в различных сферах человеческой деятельности, связанных с обработкой информации, представлением данных.

В современном обществе, которое функционирует в жестких рыночных условиях, своевременная обработка информации способствует совершенствованию организации производства, оперативному и долгосрочному планированию, прогнозированию и анализу хозяйственной деятельности, что позволяет успешно конкурировать на рынке. Каждая организация стремится минимизировать затраты времени, материальных, трудовых ресурсов в ходе своей деятельности и упростить процесс обработки информации. Эти задачи можно решить с использованием автоматизированных информационных систем.

Современные крупные проекты информационных систем характеризуются, как правило, следующими особенностями:

* сложность описания (достаточно большое количество функций, процессов, элементов данных и сложные взаимосвязи между ними), требующая тщательного моделирования и анализа данных и процессов;
* наличие совокупности тесно взаимодействующих компонентов (подсистем), имеющих свои локальные задачи и цели функционирования (например, традиционных приложений, связанных с обработкой транзакций и решением регламентных задач, и приложений аналитической обработки (поддержки принятия решений), использующих нерегламентированные запросы к данным большого объема);
* необходимость интеграции существующих и вновь разрабатываемых приложений;
* функционирование в неоднородной среде на нескольких аппаратных платформах.

Использование баз данных и информационных систем становится неотъемлемой составляющей деловой деятельности современного человека и функционирования преуспевающих организаций. В связи с этим большую актуальность приобретает освоение принципов построения и эффективного применения соответствующих технологий и программных продуктов: систем управления базами данных, CASE-средств автоматизации проектирования и других.

# Описание предметной области

В данном комплексе лабораторных работ в качестве исследуемой системы рассматривается приложение для подсчета физической работоспособности.

# Техническое задание

## Наименование и область применения

Наименование разрабатываемого программного продукта: приложение для подсчета физической работоспособности.

Областью применения являются медицинские учреждения.

## Назначение разработки

Основным назначением разработки является автоматизация и облегчение работы врачей, производящих расчет физической работоспособностей у пациентов.

## Требования к программе или программному изделию

### Требования к функциональным характеристикам

Программа должна обеспечивать возможность выполнения следующих функций:

Режим функционирования:

* автономный – используется в случае отсутствия потребности в высокой точности для расчета примерного значения затрачиваемой энергии при выполнении упражнения. В этом случае подключаемое устройство отсутствует, значение угла между голенью и полом и бедром, и голенью вводится вручную и используются для всех приседаний;
* комплексный – основной режим работы, происходит обмен данными с подключаемым устройством, расчет ведется для каждого приседания отдельно;
* повторный расчет – данные вводятся из файла, полученного при одном из предыдущих измерений, режим предназначен для повторения вычислений.

Постоянные параметры, вводимые вручную:

* рост пациента;
* масса тела пациента;
* время выполнения упражнения.
* таблицы длин сегментов тела для различных половозрастных и соматотипических групп, хранящиеся в виде отдельных файлов формата «\*.json» с возможностью редактирования из программы.
* длины сегментов тела, вводимые вручную или подставляемые из таблиц по типу пациента.
* углы между голенью и полом и между голенью и бедром, получаемые из измерительного устройства или водимые вручную.

2.3.2 Выходные данные программы

Выходными данными программы являются:

* протокол выполнения упражнения с указанием углов, вычисленной мощности для каждого приседания и упражнения в целом, выводимый на экран и сохраняемый в виде файла формата «\*.json»;
* массив исходных данных, полученный от устройства и данные, введенные вручную или подставленные из таблиц, представленный в виде бинарного файла, который пригоден для загрузки в программу для повторных вычислений.

2.3.3 Требования к модулю справочной информации

Предусмотреть разработку руководства пользователя по работе с программой, в котором должна содержаться необходимая информация по подключению программы к устройству и вводу исходных данных из таблиц, а также редактированию таблиц.

Инструкции должны иметь графические пояснения в виде скриншотов (где это возможно).

## 2.4 Требования к надёжности

### 2.4.1 Требования к обеспечению надежного функционирования программы

Надежное (устойчивое) функционирование программы должно быть обеспечено выполнением заказчиком совокупности организационно-технических мероприятий, перечень которых приведен ниже:

* Поддержка заряда технических средств;
* Следование указаниям руководства эксплуатации от производителей датчиков;
* Эксплуатация устройства в разрешенных производителем условиях;
* Следование руководству пользователя от разработчика.

Со стороны разработчика:

* Создание руководства пользователя, описывающего принципы работы с приложением;
* Создание устойчивого программного обеспечения.

### 2.4.2 Время восстановления после отказа

Время восстановления после отказа, вызванного сбоем в работе технических средств (иными внешними факторами), не фатальным сбоем (не крахом) операционной системы, не должно превышать 30-ти минут при условии соблюдения условий эксплуатации технических и программных средств.

Время восстановления после отказа, вызванного неисправностью технических средств, фатальным сбоем (крахом) операционной системы, не должно превышать времени, требуемого на устранение неисправностей технических средств и переустановки программных средств.

### 2.4.3 Отказы из-за некорректных действий пользователей системы

Отказы возможны вследствие некорректных действий пользователя при взаимодействии с программой. Несовместимостью стороннего программного обеспечения. Аппаратные конфликты. В данном случае разработчик не несет ответственности за действия пользователя, а также не имеет возможности заявить время восстановления после отказа в случае возникновения ошибок, вызванных некорректными действиями пользователя.

## 2.5 Требования к составу и параметрам технических средств

В состав технических средств должен входить IBM-совместимый персональный компьютер, который включает в себя:

* процессор типа Рentium и выше;
* оперативную память объёмом 1 Гб и более.

Для печати отчётов необходим принтер.

## 2.6 Требования к информационной и программной совместимости

Системные программные средства, используемые программой, должны быть представлены лицензионной локализованной версией операционной системы Windows 7 и выше.

Исходные коды программы должны быть реализованы на языке высокого уровня.

## 2.7 Требования к программной документации

Разрабатываемые программные модули должны быть самодокументированы, то есть тексты программ должны содержать все необходимые комментарии, а имена переменных и функций должны отражать суть происходящего

В состав сопровождающей документации должны входить пояснительная записка, содержащая описание разработки, руководства пользователя и администратора.

## 2.8 Стадии и этапы разработки

Разработка должна быть проведена в четыре стадии: разработка технического задания, проектирование, реализация и тестирование программного обеспечения.

На стадии разработки технического задания должен быть выполнен этап разработки и утверждения настоящего технического задания.

На стадии проектирования должны быть выполнены перечисленные ниже этапы:

* проектирование динамической структуры системы;
* проектирование функционирования системы;
* проектирование статической структуры системы;
* разработка алгоритмов функционирования системы;
* проектирование структуры базы данных.

На стадии реализации программного обеспечения должны быть выполнены перечисленные ниже этапы:

* реализация структуры базы данных;
* выбор языка программирования;
* реализация структуры интерфейса;
* реализация программного кода;
* разработка руководств пользователя и администратора.

На стадии тестирования программного обеспечения должны быть выполнены перечисленные ниже этапы:

* тестирование на основе метода чёрного ящика;
* тестирование на основе метода белого ящика.

# 3 Структурный подход

Для проведения анализа и реорганизации бизнес-процессов предназначено CASE-средство AllFusion ERwin Process Modeler, поддерживающее методологии: IDEF0 (функциональная модель), IDEF3 (Workflow Diagram), DFD (DataFlow Diagram).

Построение модели системы начинается с описания функционирования системы в целом.

## 3.1 Описание системы

Пациенту необходимо заполнить регистрационную форму, включающую данные об идентификационном номере, которыми может являться номер паспорта или полиса медицинского страхования. Кроме того пациенту необходимо сообщить свои антропометрические данные: рост и вес, так как без них расчет провести будет невозможно.

Врачу необходимо зарегистрировать пациента, внести в соответствующие текстовые поля данные пациента. Так же врачу необходимо проверить, не проходил ли пациент данное обследование прежде. Данная проверка проводится с помощью сверки идентификационного номера пациента. Кроме того до начала исследований необходимо закрепить все необходимые датчики и провести с пациентом краткий инструктаж об использовании системы.

Врач может засечь конкретный промежуток времени для выполнения приседаний, или позволить пациенту приседать пока тот не устанет. В первом случае, врачу необходимо использовать секундомер или специальные часы.

После выполнения упражнения датчики врач помогает отсоединить датчики, фиксирует полученные данные в специальном протоколе и прикрепляет его к истории болезни или выдает протокол пациенту. Кроме того по полученным результатам можно будет сделать заключение о допустимых физических нагрузках, возможности работы на предприятиях где требуется физический труд и прочее, зависящее от компетентности врача.

В случае некорректности полученных данных или возникновения ошибок влияющих на полученный результат исследования, врачу необходимо провести исследование еще раз дав пациенту перерыв для восстановления сил.

Входными данными системы являются регистрационные данные пациента (Имя, фамилия), антропометрические данные пациента (рост, вес и тд), а так же формулы для расчета и данные с датчиков.

Построение модели описанной системы начинается с контекстной диаграммы, которая рассматривает систему с точки зрения стороннего наблюдателя

Контекстная диаграмма представлена на рисунке 1.

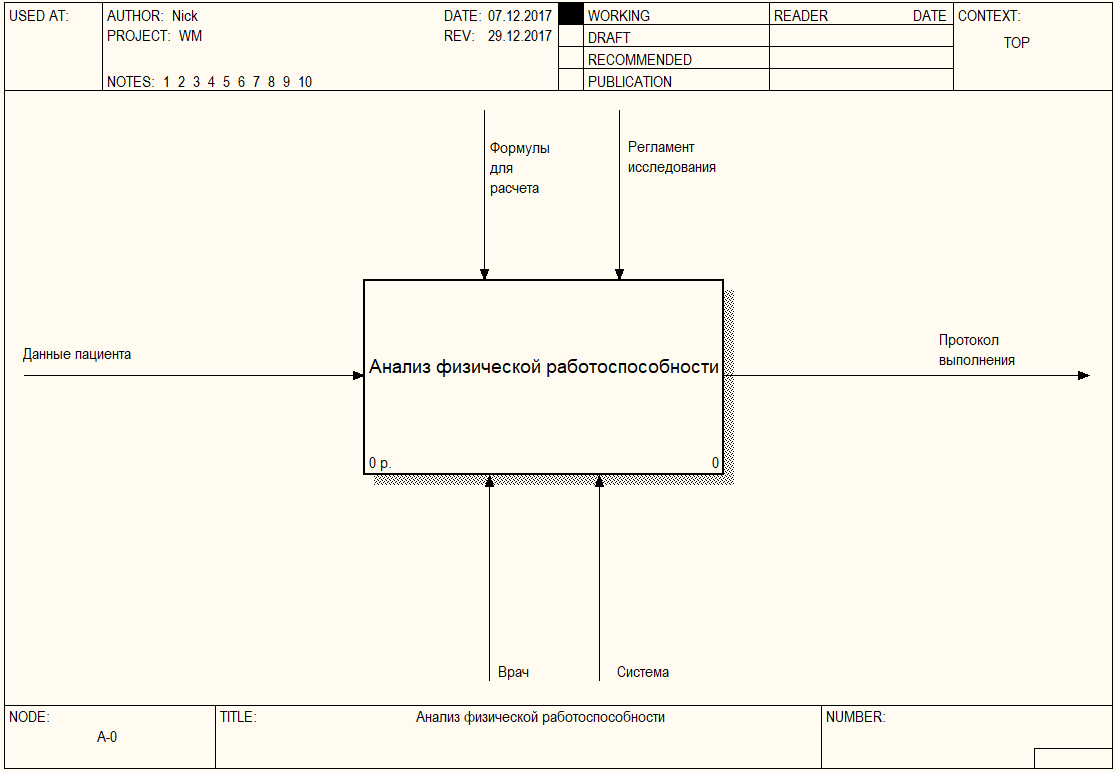


Рисунок 1 – Контекстная диаграмма IDEF0

Функционирование системы описывается в терминах:

* вход:
  + данные пациента;
* выход
  + протокол выполнения;
* управление
  + формулы для расчета;
  + регламент исследования;
* механизм
  + врач;
  + система.

Медицинское обследование расчета физической работоспособности включает в себя целый комплекс рутинных действий, записывания данных пациент, измерения расстояний от точки в приседе, до точки в вертикальном положении. Данную задачу можно оптимизировать с помощью программы, которая с помощью специального оборудования и формул высчитывает физическую работоспособность пациента, а так же запоминает данные о пациенте.

Диаграмма декомпозиции функции анализа физической работоспособности представлена на рисунке 2.

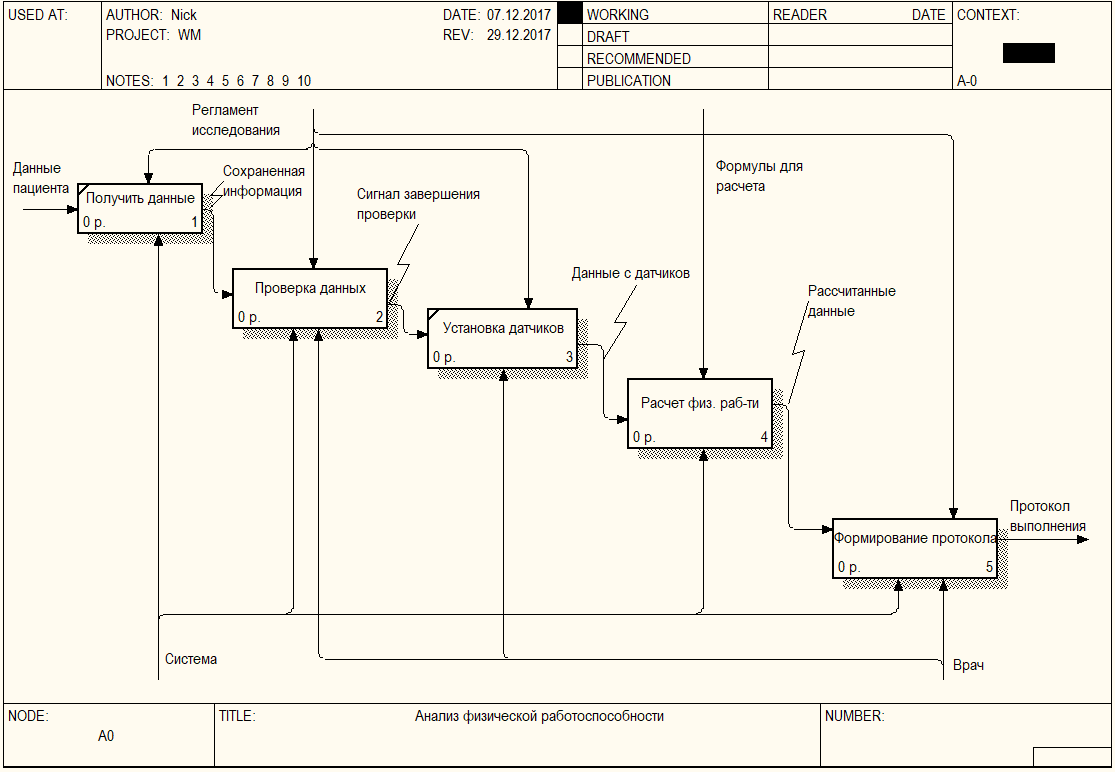


Рисунок 2 – Декомпозиция программы анализа физической работоспособности

Проверка данных пациента необходима для унификации файлов исследования, в данный файл входит id пациента, его персональные данные, а так же антропометрические данные. Врач может при необходимости изменить антропометрические и персональные данные пациента. Так же врачу необходимо вводить дату проведения исследования и сохранять протокол выполнения упражнений пациентом. Соответственно к системе предъявляются требования обеспечения пересылки данных между локальным хранилищем и базой данных.

Для наглядности лучше использовать диаграмму DFD, чтобы показать на них потоки передаваемых данных.

DFD-диаграмма проверки данных представлена на рисунке 3.

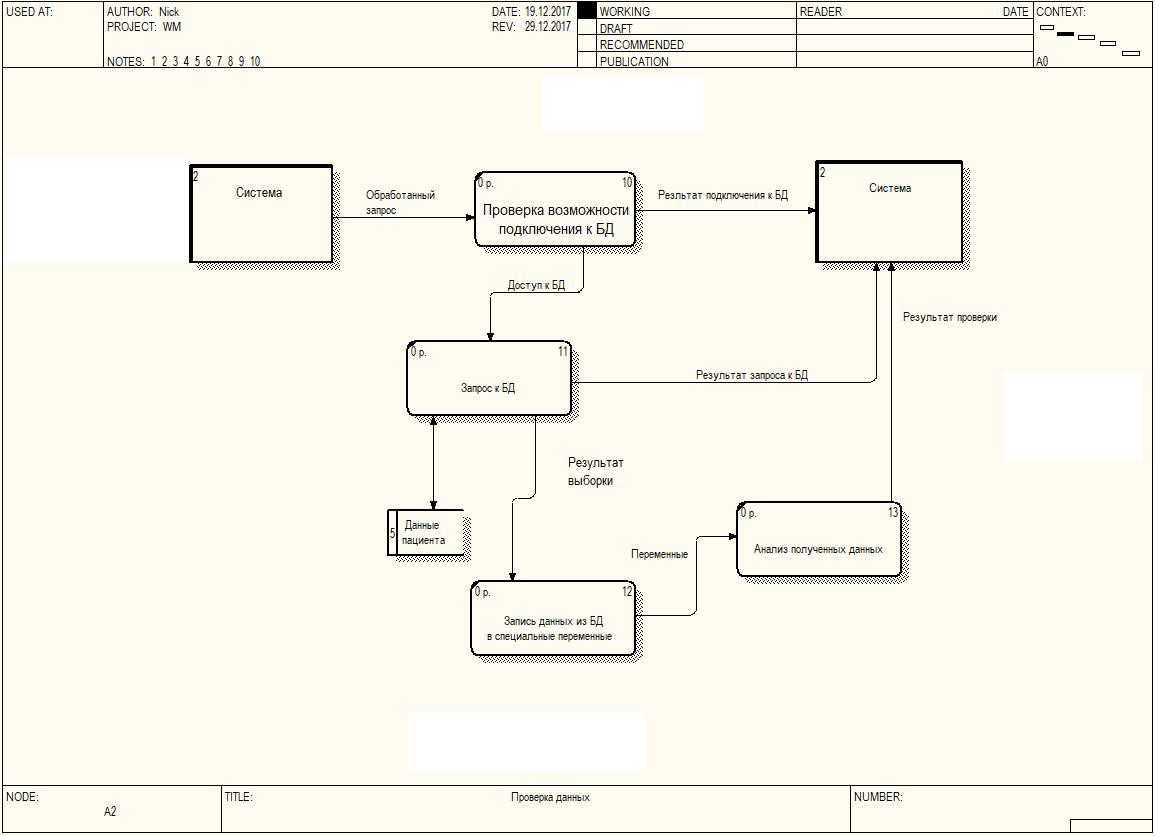


Рисунок 3 – DFD диаграмма проверки данных

При сохранении данных об исследования необходимо проверить возможность сохранения файла пациента в базе данных, в случае, если база данных по каким-то причинам не позволяет это сделать, то программа должна предусматривать возможность сохранения данных в локальном хранилище.

Демонстрация работы данного механизма представлена на DFD диаграмме формирования протокола.

DFD-диаграмма формирования протокола представлена на рисунке 4.

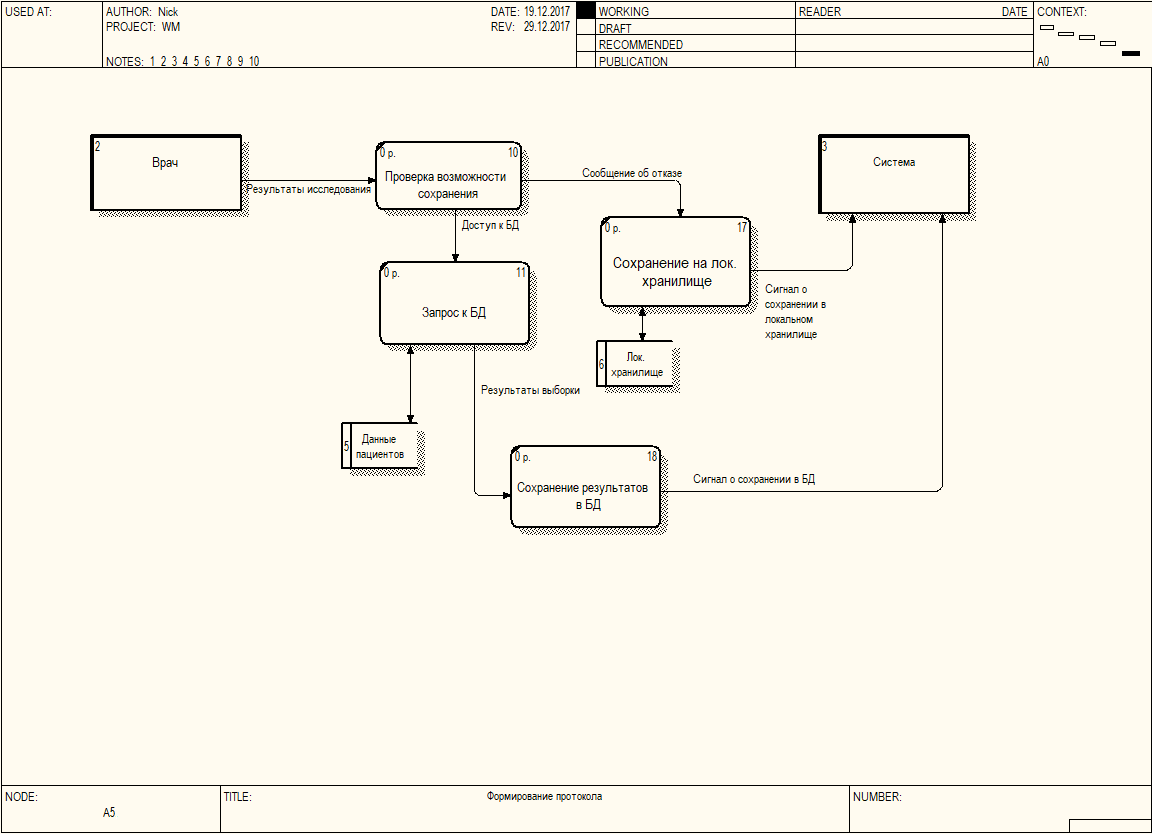


Рисунок 4 – DFD диаграмма формирования протокола

Дальнейшую декомпозицию целесообразнее представить в виде диаграмм IDEF3 – так как они отражают порядок выполнения действий.

При получении данных пациенту необходимо сообщить свои регистрационные данные: id, ФИО, а также антропометрические данные для исследования: рост, вес и др. При необходимости, или при ошибке в ходе заполнения специальных полей для ввода, врач может повторить процедуру получения данных. После получения данных, они сохраняются на время работы программы в специальные переменные.

Данные действия описаны в диаграмме получения данных, приведенной на рисунке 5.

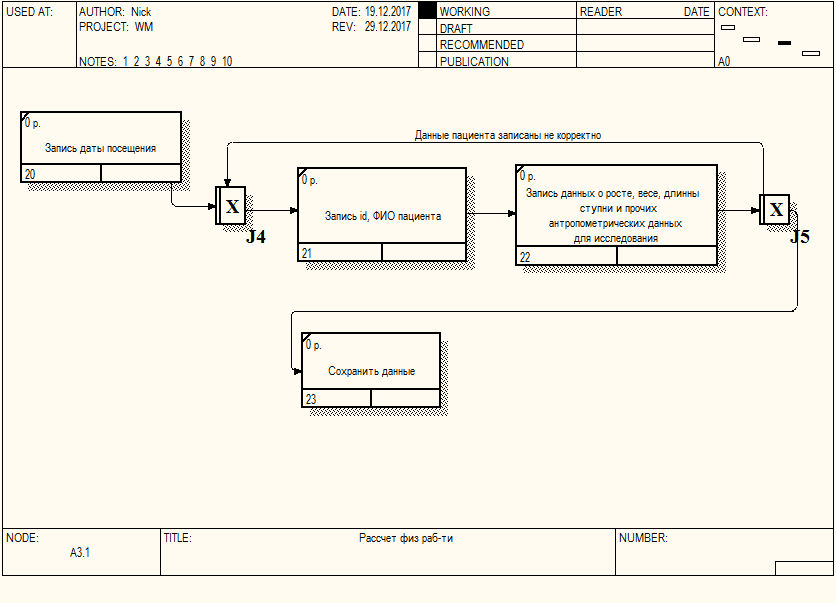


Рисунок 5 – Получение данных

Одним из самых важных этапов, является установка датчиков. С помощью их система получает данные необходимые для расчета физической работоспособности. Важно при установке датчиков убедится в том, что устройства полностью готово к работе, установлен стойкий сигнал связи с компьютером, на котором запущена программа, а так же важно синхронизировать датчики, иначе данные получаемые с них могут быть не корректными. После выполнения всех проверок необходимо уведомить пациента о том, что ему можно начинать выполнять упражнение с помощью сообщения на экране компьютера или с помощью звукового сигнала.

Данные процесс описан на диаграмме установка датчиков, представленной на рисунке 6.

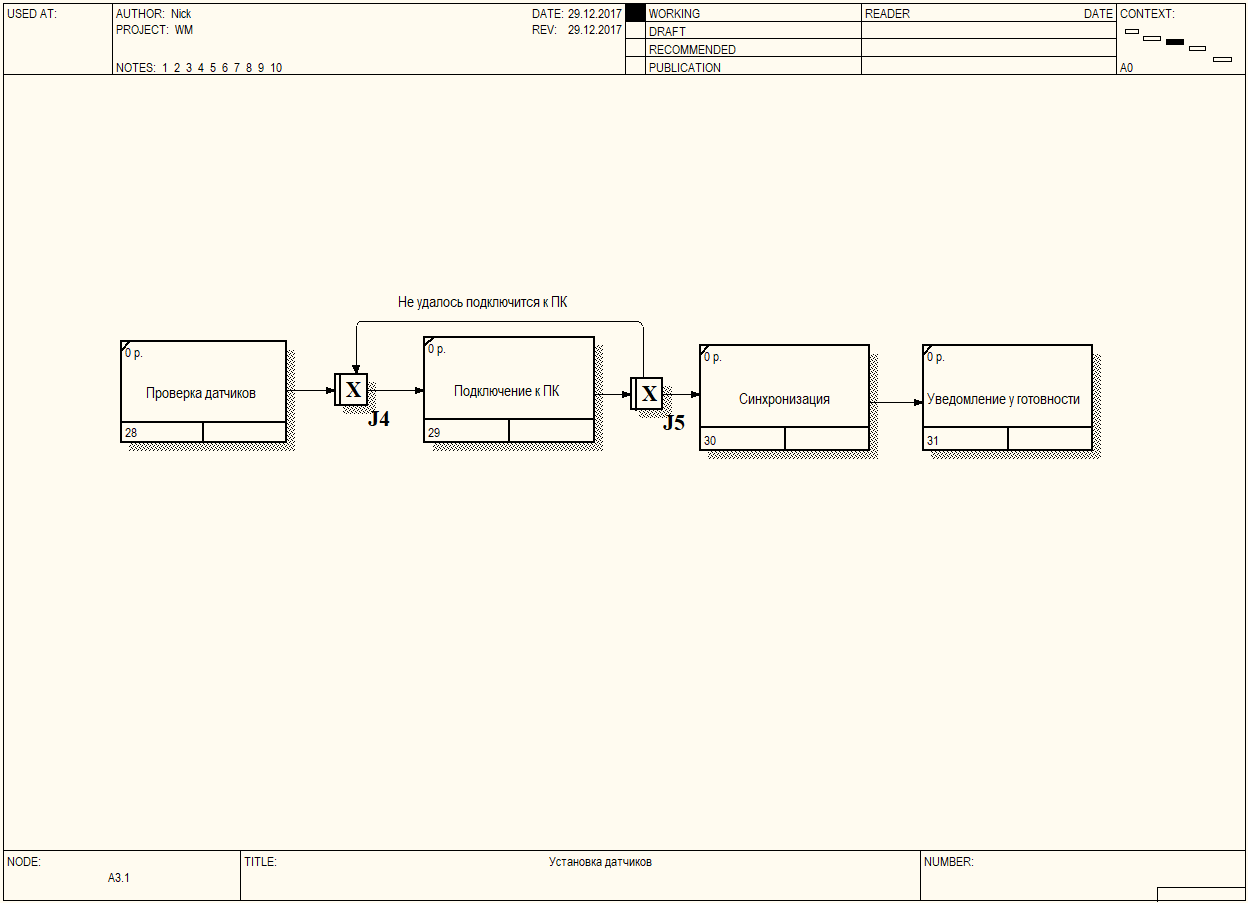


Рисунок 6 – Подключение датчиков

Для работы приложения необходима база данных.

Базу данных наглядно можно представить в виде ER-модели данных.

Базовыми понятиями ER-модели данных (ER — Entity-Relationship) являются:

— сущность;

— атрибут;

— связь.

Сущность — это класс однотипных объектов, информация о которых должна быть учтена в модели. Сущность имеет наименование, выраженное существительным в единственном числе, и обозначается в виде прямоугольника с наименованием.

Атрибут сущности — это именованная характеристика, являющаяся некоторым свойством сущности. Наименование атрибута должно быть выражено существительным в единственном числе (возможно, с описательными оборотами или прилагательными). Атрибуты изображаются в прямоугольнике, обозначающем сущность.

Ключ сущности — это не избыточный набор атрибутов, значения которых в совокупности являются уникальными для каждого экземпляра сущности. При удалении любого атрибута из ключа нарушается его уникальность. Ключей у сущности может быть несколько. На диаграмме ключевые атрибуты отображаются подчеркиванием.

Связь — это отношение одной сущности к другой. Каждая связь имеет одно или два наименования. Наименование обычно выражается неопределенной формой глагола. Каждое из наименований относится к своему концу связи. Иногда наименования не пишутся ввиду их очевидности.

Связь типа один-к-одному означает, что один экземпляр первой сущности связан точно с одним экземпляром второй сущности.

Связь типа один-ко-многим означает, что один экземпляр первой сущности связан с несколькими экземплярами второй сущности.

Связь типа многие-ко-многим означает, что каждый экземпляр первой сущности может быть связан с несколькими экземплярами второй сущности, и наоборот. Этот тип связи является временным, допустимым на ранних этапах разработки модели. В дальнейшем такую связь необходимо заменить двумя связями типа один-ко-многим путем создания промежуточной сущности.

Каждая связь может иметь одну из двух модальностей связи:

— модальность «может» (обозначается пунктирной линией);

— модальность «должен» (обозначается прямой линией).

Связь может иметь разную модальность с разных концов.

Для корректной работы программы необходимо хранить данные о пациентах.

Пациент:

* ФИО
* Дата рождения
* ID пациента

Обследование:

* Идентификатор обследования
* Дата обследования
* Идентификатор пациента
* Рост
* Вес
* Длина туловища
* Длина плеча
* Длина предплечья
* Длина кисти
* Длина бедра
* Длина голени
* Высота стопы
* Высота головы

Данный способ разделения данных позволяет разбить данные необходимые для сверки регистрационных данных пациентов, и проведения исследований.

Необходимо установить связи между данными таблицами: пациент может пройти множество обследований, неизменным являются идентификаторы пациента и даты обследований.

Соответствующая диаграмма представлена на рисунке 7.



Рисунок 7 – Схема базы данных

# 4 Объектно-ориентированный подход

ООП – это подход, описывающий систему с точки зрения взаимодействия объектов.

При объектно-ориентированном подходе первой обычно строят диаграмму вариантов использования, показывающую совокупность вариантов использования и исполнителей, а также отношения между ними.

Действующими лицами являются пациент и врач использующий специальную программу.

При выполнении операции получения данных участвуют пациент и врач, в выполнение упражнений задействован только пациент и специализированные датчики, при формировании заключительных результатов, врач, проанализировав протокол может поставить диагноз или сообщить пациенту свои рекомендации.

Установку датчиков производит врач, настраивая оборудование и подгоняя крепеж датчиков для удобства пациента.

Диаграммы вариантов использования представлены на рисунках 8-11.

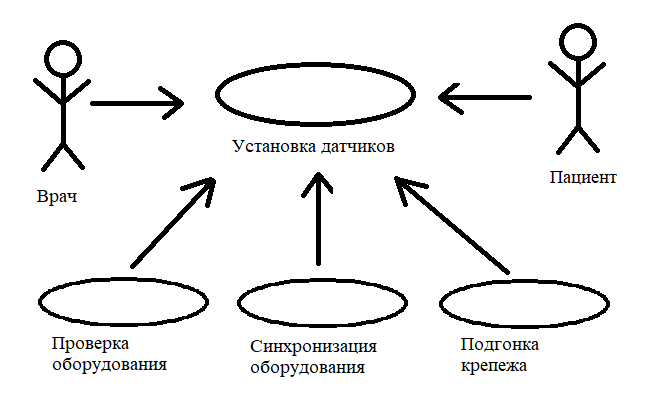


Рисунок 8 – Диаграмма вариантов использования

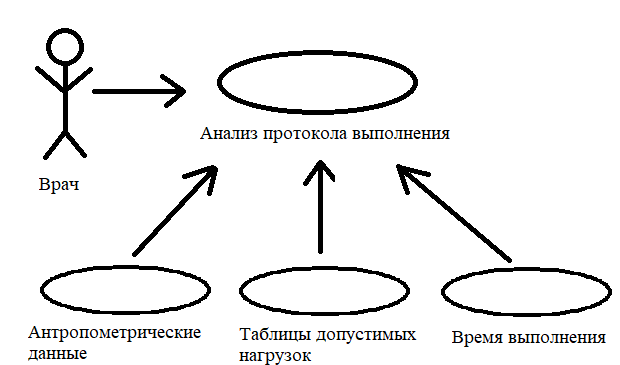


Рисунок 9 – Диаграмма вариантов использования

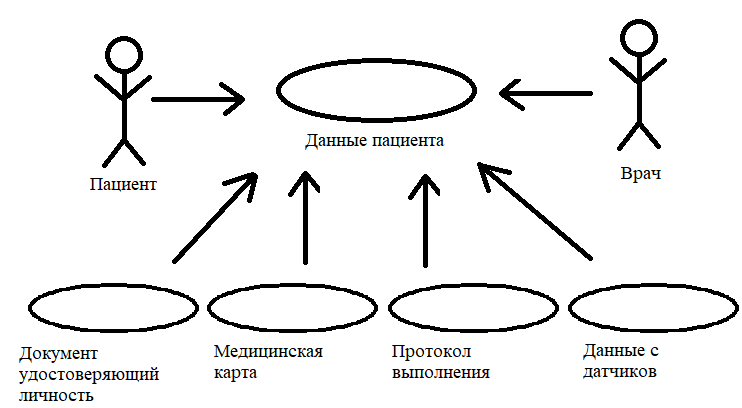


Рисунок 10 – Диаграмма вариантов использования

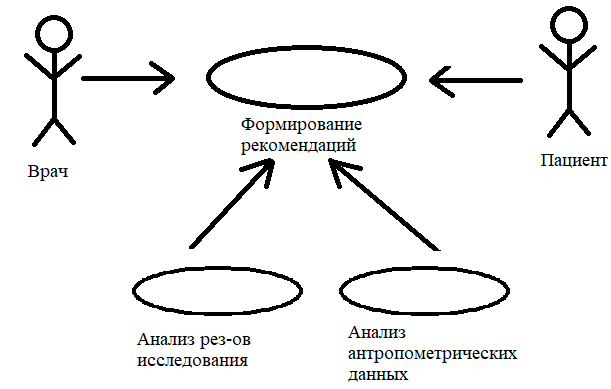


Рисунок 11- Диаграмма вариантов использования

Диаграммы взаимодействия моделируют взаимодействия между объектами системы.

На диаграмме взаимодействия отображают один из процессов обработки информации в варианте использования.

Для представления на диаграмме взаимодействия выбран процесс получения данных о пациенте.

Пациент сообщает свои регистрационные данные, используя документ удостоверяющий личность (паспорт, права, регистрационный талон и тд), после сообщает свои антропометрические данные необходимые для исследования. Врач записывает сообщаемые данные в специальные поля для ввода информации в соответствующих формах программы. В случае ошибки или указания неверных данных операция может быть проведена повторно.

На рисунке 12 представлена диаграмма взаимодействия.

Рисунок 12 – Диаграмма взаимодействия

На основе диаграммы взаимодействия строится диаграмма кооперации. Кооперативная диаграмма организована вокруг самих объектов. Здесь отображается поток данных. Графически такая диаграмма представляет собой граф из вершин и ребер. Она представлена на рисунке 13.

Рисунок 13 – Кооперативная диаграмма

Элементы на диаграмме состояний отображают жизненный цикл одного объекта, начиная с момента его создания и заканчивая разрушением.

Главное назначение диаграммы состояний — описать возможные последовательности состояний и переходов, которые в совокупности характеризуют поведение моделируемой системы в течение всего ее жизненного цикла.

В случае посещения пациентом медицинского учреждения, ему необходимо сообщить свои регистрационные данные, после проверки данных, врач получит файл пациента или создаст новый. В данный файл необходимо записать антропометрические данные или проверить необходимость их корректировки, если пациент проходил ранее данное обследование. После занесения всех данных, врач подключает датчики проверяя их работоспособность, в случае успешного подключения система должна уведомить об этом сообщением на экране пк или звуковым сигналом. Если подключение не прошло успешно, датчики посылают сообщение об ошибке с помощью специальных индикаторов, врачу нужно попытаться исправить неисправность оборудования, если поломка не устранима, то обследование считается законченным. В случае успешного подключения или починки датчиков врачом начинается процесс получения данных о углах наклона тела пациента, программа формирует протокол выполнения и по завершению выполнения упражнений выдает его на экран пк.

Диаграмма состояний представлена на рисунке 14.

Рисунок 14 – Диаграмма состояний

Диаграмма классов — это диаграмма языка UML, на которой представлена совокупность декларативных или статических элементов модели, таких как классы с атрибутами и операциями, а также связывающие их отношения.

С помощью диаграмм классов разработчики могут видеть и планировать структуру системы еще до фактического написания кода, благодаря чему с самого начала могут понять, хорошо ли спроектирована система.

В диаграмме классов описаны классы и методы, используемые системой для проведения операций сервисного обслуживания.

Класс Доктор содержит такие поля как логин, пароль. Класс доктор содержит такие методы, как запрос данных пациента, ремонт датчиков.

Пациент содержит поля регистрационных и антропометрических данных, а так же такие методы как, отправка регистрационных и антропометрических данных.

Класс датчик содержит данные об углах наклона пациента, кроме того он содержит методы отправки сообщения об успешности подключения к пк и отправки кода ошибки подключения.

Система содержит в себе поля данных пациента, данные датчиков и протокол выполнения упражнений. Классу системе соответствуют методы отправки результатов исследования, получения информации от датчиков, получения информации о регистрационном процессе пациента (был он зарегистрирован ранее или нет), а так же вывод сообщений о успешности процесса сохранения данных.

Диаграмма классов представлена на рисунке 15.



Рисунок 15 – Диаграмма классов

Вывод

В результате выполнения данного комплекса лабораторных работ изучены структурный и объектно-ориентированный подходы к проектированию информационных систем. Также разработаны различные диаграммы, упрощающие последующее написание программного продукта.

Каждый из рассматриваемых подходов имеет свои преимущества и недостатки. В зависимости от решаемых задач более целесообразно использовать тот или иной подход.

К достоинствам структурного подхода можно отнести:

— возможность проведения глубокого анализа бизнес-процессов, выявления узких мест;

— применение диаграмм IDEF0, IDEF3 и DFD обеспечивает логическую целостность и полноту описания, необходимую для достижения точных и непротиворечивых результатов.

Недостатком структурного подхода может быть ситуация, когда при увеличении количества уровней представления анализ и модификация моделей становятся затруднительными.

К достоинствам объектно-ориентированного подхода можно отнести:

— сравнительная легкость, наглядность, эффективность моделей;

— возможность автоматической генерации кода на основе построенных моделей.

Недостатки объектно-ориентированного подхода:

— невозможность проведения детального анализа бизнес-процессов;

— неполнота и незавершенность некоторых видов диаграмм, возможность их неверной интерпретации.

Таким образом, основным фактором для выбора подхода является решаемая задача. Также немаловажным является то, насколько аналитик знаком со структурным или объектным подходом.

Для решения данной задачи оптимальным является структурный подход.