




## Лабораторная работа №1

### Планирование и организация проекта по разработке программного средства. Организация инфраструктуры разработки проекта

#### Цели выполнения лабораторной работы:

- выявить проблему разработки, определить цели и задачи программного средства
- выбрать методологию разработки ПС и организовать рабочую инфраструктуру, необходимую для разработки программного средства

|  Задание | Этапы выполнения задания   |
|---|--|
| 1 Проанализировать предметную область и выявить проблему разработки                       | <ol style="list-style-type: none"><li>1 Проанализировать предметную область выбранной темы:<ul style="list-style-type: none"><li>■ провести анализ бизнеса и его продукта;</li><li>■ выявить ключевые заинтересованные стороны (стейкхолдеры), имеющие отношение к проекту по разработке ПС и определить их ключевые потребности;</li><li>■ проанализировать интересы и влияние каждого стейкхолдера на проект;</li></ul></li><li>2 Сформулировать проблему разработки</li><li>3 Определить цель(-ли) и задачи разработки</li></ol>  |
| 2 Организовать процесс разработки программного средства                                   | <ol style="list-style-type: none"><li>1 Выбрать и обосновать проектную методологию разработки ПС</li><li>2 Выбрать средства для работы над проектом по разработке программного средства</li></ol>  |
| 3 Сформировать отчет по лабораторной работе   | <p>Содержание отчета:</p> <ol style="list-style-type: none"><li>1 Титульный лист (приложение А)</li><li>2 Анализ предметной области<ol style="list-style-type: none"><li>2.1 Анализ бизнеса и его продукта</li><li>2.2 Анализ заинтересованных сторон</li><li>2.3 Реестр заинтересованных сторон</li><li>2.4 Описание проблемы разработки</li></ol></li><li>3 Формулировка цели разработки</li><li>4 Перечень задач, необходимых для реализации проекта по разработке программного средства</li><li>5 Обоснование выбора методологии разработки программного средства</li><li>6 Описание инфраструктуры по разработке программного средства</li><li>7 Выводы</li></ol> |



## Краткие теоретические сведения и методические указания к Заданию 1

Как и любой живой организм, программа имеет свой жизненный цикл (англ. Software development Life Cycle – SDLC).

В отечественной и зарубежной литературе существует множество определений понятия «жизненный цикл». Одно из них: «Жизненный цикл (рисунок 3) – совокупность взаимосвязанных процессов создания и последовательного изменения состояния продукции от формирования к ней исходных требований до окончания ее эксплуатации или потребления» [1, стр.28].

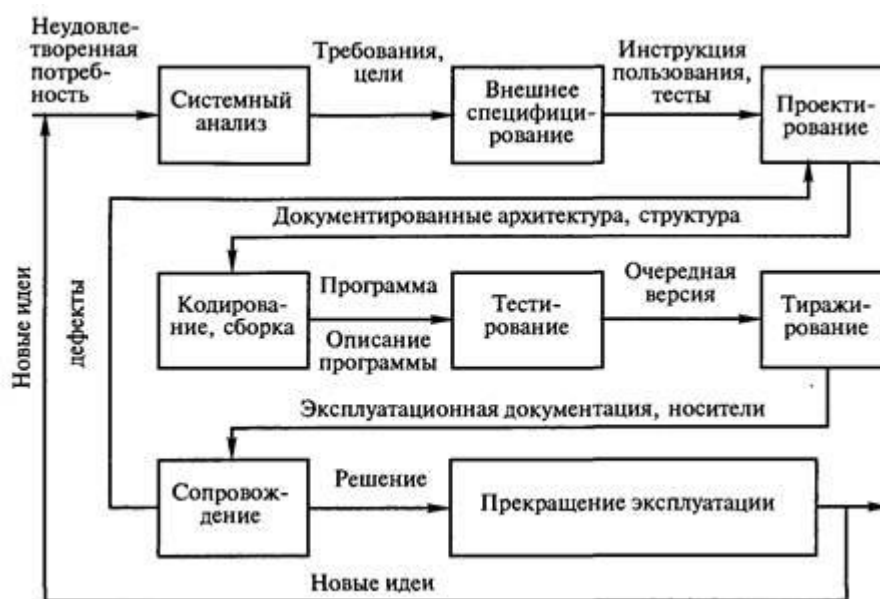


Рисунок 3 – Жизненный цикл программных изделий [1, стр.28]

Из рисунка 3 следует, что жизненный цикл программы состоит из следующих этапов:

- 1 Выявление неудовлетворенной потребности или проблемы разработки.
- 2 Системный анализ выявленной потребности и формулировка цели (-ей) и задач разработки, а также определение требований к будущему программному обеспечению.
- 3 Внешнее верифицирование служит для создания «видения» программы, т.е. формирование общего направления работы на следующих этапах. Кроме того, на данном этапе может быть разработан пользовательский интерфейс будущего ПО, а также инструкции по его использованию.
- 4 Проектирование необходимо для создания полной спецификации программного изделия, начиная от постановки задачи, заканчивая планом разработки отдельных частей программы.
- 5 Кодирование и сборка, результатом которого является версия программы, готовая для установки, а также ее описание.



6 Тестирование – в рамках данного этапа разработанное программное изделие проходит различные проверки на соответствие ранее определенным требованиям.

7 Тиражирование подразумевает составление документации по эксплуатации программы для различных пользователей и подготовку способов доставки программы конечному пользователю.

8 Сопровождение предполагает установку разработанного ПО на машины конечного пользователя с целью выявления ошибок и выпуска обновлений для устранения выявленных замечаний. Также данный этап используется для усовершенствования базовой версии программы.

9 Прекращение эксплуатации означает не только факт удаления программы из машин конечных пользователей, но и период времени, в течение которого программа может использоваться некоторыми пользователями.

Помимо данной интерпретации жизненного цикла на практике также может быть использована модель жизненного цикла, представленная на рисунке 4.

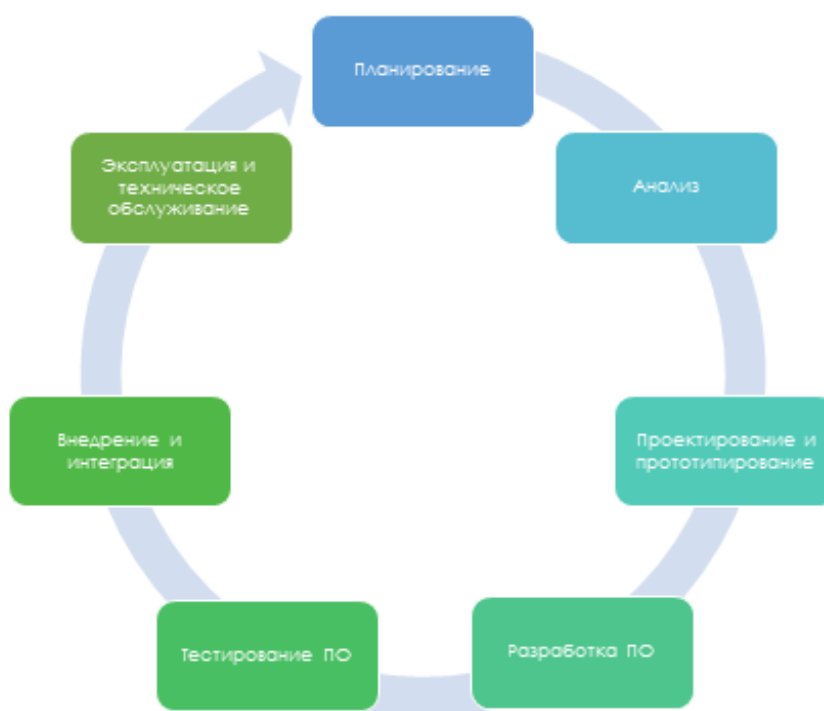


Рисунок 4 – Этапы жизненного цикла программного обеспечения

**Планирование.** Этап планирования, еще его называют этапом технико-экономического обоснования – это этап, на котором будет осуществлено построение плана работы над предстоящим проектом. Данный этап решает следующие задачи:

- а) определение проблемы и контекста существующих систем (т.е. существующих аналогов) для определения целей нового, создаваемого ПО;
- б) оценка стоимости реализации ПО;



в) обеспечение работы ресурсами для реализации ПО.

Из вышесказанного следует, что основными выходными данными этапа планирования будут являться сроки и стоимость реализации плана по разработке ПО.

**Анализ.** На данной стадии собираются воедино все детали нового ПО, а также обсуждаются начальные идеи прототипов как программных, так и пользовательских будущего ПО.

На этапе анализа будут:

- определены системные требования прототипов;
- осуществлен поиск альтернатив уже существующим прототипам;
- определены пользовательские требования.

Результатом данного этапа можно назвать составление спецификации системных требований к ПО (англ. SRS – system requirements specification).

**Проектирование и прототипирование.** Работа на данном этапе начинается с создания прототипов пользовательского интерфейса. В зависимости от тематики проекта могут быть созданы: диаграммы, графические схемы программы, эскизы и т.д. Помимо работы над пользовательским интерфейсом, также осуществляется работа по проектированию способа хранения данных, системных и сетевых интерфейсов. Основным результатом работы на данной стадии является создание документа, содержащего детальное описание всех требований к разрабатываемому ПО. Данный документ является отправной точкой для разработчиков, которые будут реализовывать проект.

**Разработка ПО.** Название данного этапа обуславливает его основную задачу – разработать ПО согласно требованиям, определенным на предыдущих стадиях. Перед непосредственным написанием программного кода разработчики выбирают программные компоненты для наиболее эффективной и оптимальной реализации проекта. С целью выбора наиболее подходящих для разработки компонентов необходимо проанализировать, какие компоненты на данный момент уже используются для решения задач, определенных требованиями документов из предыдущих стадий, и выбрать наиболее приемлемые. Зачастую такой анализ позволяет не только сократить сроки разработки ПО, но и снизить его стоимость, т.к. в некоторых случаях дешевле будет приобрести стороннее решение и адаптировать его под поставленные задачи. В случае, когда на рынке отсутствуют решения, позволяющие решить поставленные задачи, необходимо разработать и реализовать с нуля способы, позволяющие удовлетворить все пользовательские требования.

**Тестирование ПО.** Несмотря на все усилия, прилагаемые на стадиях проектирования и разработки ПО, реализовать верно абсолютно все пользовательские требования невозможно. Поэтому непосредственно после стадии разработки следует стадия тестирования – это своеобразная проверка на соответствие реализованной функциональности описанным пользовательским требованиям. Кроме того, тестирование позволяет повысить качество разрабатываемого ПО и сократить количество ошибок, выявляемых на стадиях внедрения и эксплуатации.



На сегодняшний день существует множество техник тестирования. Количество применяемых при тестировании техник напрямую зависит от назначения и масштаба проекта. Также стоит помнить, что применение нескольких видов тестирования на маломасштабных проектах может привести к увеличению стоимости реализации проекта. Поэтому стоит со всей серьезностью подойти к выбору той или иной техники тестирования, а также необходимому количеству техник тестирования.

**Внедрение и интеграция.** В рамках данного этапа в начале осуществляется установка разработанного ПО в тестовое окружение, которое максимально приближено к тем условиям, которые на данный момент существуют у конечного пользователя. Это необходимо для того, чтобы проверить работоспособность реализованной функциональности. Затем разработанное ПО устанавливается непосредственно на рабочие машины конечных пользователей, осуществляется обучение персонала работе с новым ПО и интеграция ПО в рабочие процессы компании-заказчика. Также на данном этапе происходит оценка полученных системных показателей с показателями, определенными на стадии планирования (например, скорость работы модулей ПО, скорость ответа от БД и проч.).

**Эксплуатация и техническое обслуживание.** Данный этап может включать в себя: поддержку ПО в работоспособном состоянии, исправление ошибок, возникающих в ходе эксплуатации, расширение функциональности ПО, выпуск обновлений ПО и другие действия, направленные на сохранение работоспособности ПО.

На начальном этапе – этапе планирования проекта **выявляются проблемы и потребности в продукте, собираются исходные данные, определяются цели и задачи проекта.**

Первое, что необходимо сделать – провести **анализ бизнеса и его продукта, определить стратегические цели, выявить и проанализировать круг заинтересованных лиц.**

При формализации стратегических целей необходимо учесть, что они должны удовлетворять SMART-критериям.

---

**Техники анализа бизнеса и его продукта:** Business Model Canvas, Diagram Ishikawy, SWOT-анализ, PESTLE-анализ, MOST-анализ, GAP-анализ, MoSCoW-анализ.



**Техники выявления и анализа заинтересованных лиц:** Матрица заинтересованных лиц, Stakeholder map, Луковичная диаграмма, Матрица RACI, Матрица интересов и влияния, Матрица оценки уровня вовлечения стейкхолдеров.

---



**В разделе 2 Анализ предметной области** в обязательном порядке следует:



- продемонстрировать использование техник анализа бизнеса и его продукта (не менее 2-х);
- продемонстрировать использование техник идентификации (не менее 1-й) и анализа (Матрица RACI, Матрица интересов и влияния) заинтересованных лиц;
- сформировать Реестр заинтересованных сторон на основе проведенных исследований в соответствии с таблицей 1;
- осуществить постановку целей по методу SMART.

Таблица 1 – Реестр заинтересованных сторон

| Поле                            | Алгоритм заполнения  |
|---------------------------------|--|
| ID                              | Уникальный идентификатор требования  |
| Имя                             | Фамилия и имя заинтересованного лица   |
| Роль в проекте                  | Проектная роль (пользователь, эксперт, спонсор, член команды и т.п.)                                 |
| Должность                       | Занимаемая заинтересованным лицом должность  |
| Отдел / подразделение           | Отдел или подразделение, где работает заинтересованное лицо  |
| Непосредственный начальник      | Прямой начальник заинтересованного лица  |
| Контактная информация           | Телефон, e-mail и прочее – вся известная контактная информация                                       |
| Предпочитаемый вид коммуникаций | Электронная почта / телефон / онлайн-конференция и т.п.  |
| Главные ожидания                | Главные ожидания заинтересованного лица по проекту   |
| Главные требования              | Главные требования заинтересованного лица по проекту   |
| Влияние на проект               | Влияние на проект ПО в баллах по шкале 1–10 (где 1 – минимальное влияние; 10 – максимальное влияние) |
| Отношение к проекту             | Противник / Сторонник / Нейтрал  |
| Интерес к проекту               | Возможно, заинтересованное лицо ХОЧЕТ принять участие в проекте как эксперт или в иной форме         |
| Комментарий                     | Необходимые комментарии  |

Стоит подчеркнуть особую важность правильного определения проблемы, т.к. неверная формулировка проблемы разработки в конечном итоге приводит к ошибкам, исправление которых в дальнейшем не представляется возможным. Здесь встает закономерный вопрос: как правильно определить проблему, т.е. то, для чего в будущем будет нужно программное обеспечение?





Для ответа на данный вопрос необходимо обратиться к системному анализу, а в частности к теории принятия решений. Согласно данной теории, **проблема** – это несоответствие между существующим и требуемым (целевым) состоянием системы при данном состоянии среды в рассматриваемый момент времени [2]. Другими словами, разница между тем что есть сейчас и тем, что должно быть, и называется проблемой.

После того, как проблема была выявлена, можно сформулировать цель (или цели). **Цель** – образ желаемого будущего (субъективная цель) или будущее реальное состояние (объективная цель) [3]. Цель является способом, инструментом для устранения различия между текущим состоянием объекта (системы) и желаемым состоянием, что приводит к решению изначально выявленной проблемы.

Далее необходимо осуществить декомпозицию цели на более мелкие части, которые обуславливают формулировку набора задач, необходимых для достижения конечной цели и, как итог, решения проблемы в целом. То, в какой мере будут решены задачи, определяет степень достижения цели и степень решения проблемы.

На практике для выявления проблемы можно воспользоваться формой, представленной в таблице 2.

Таблица 2 – Форма для выражения определения проблемы [4]

| Элемент                            | Описание   |
|------------------------------------|--|
| <b>Проблема</b>                    | [описание проблемы]  |
| <b>воздействует на</b>             | [перечень сторон, на которых оказывает влияние данная проблема]                          |
| <b>результатом чего является</b>   | [описание воздействия данной проблемы на заинтересованных лиц и/или бизнес-деятельность] |
| <b>Выигрыш от</b>                  | [описание предлагаемого решения]   |
| <b>Может состоять в следующем:</b> | [перечень основных преимуществ, представляемых решением]                                 |

Пример заполнения формы приведен в таблице 3.

Продолжая работу с примером из таблицы 2, можно сформулировать цель разработки ПО следующим образом: необходимо создать такое программное изделие, которое позволило бы получить доступ к медицинским данным населения и осуществлять их обработку.

На основании выявленной цели, можно определить набор задач, которые требуется решить для достижения поставленной цели. Например, набор задач может быть следующим:

- составить список функциональных и нефункциональных требований;
- спроектировать пользовательский интерфейс;
- спроектировать структуру данных и архитектуру программы;
- разработать и протестировать итоговую версию программы.



Таблица 3 – Пример заполнения формы для выражения определения проблемы [4]

| Элемент                            | Описание   |
|------------------------------------|--|
| <b>Проблема</b>                    | Отсутствие доступа к интегрированным данным о состоянии здоровья населения   |
| <b>воздействует на</b>             | <ul style="list-style-type: none"><li>– персонал медицинских организаций, оказывающих медицинскую помощь населению;</li><li>– граждан, обращающихся за оказанием медицинской помощи</li></ul>  |
| <b>результатом чего является</b>   | <ul style="list-style-type: none"><li>– высокая вероятность принятия неверного или необоснованного медицинского решения о лечении пациента;</li><li>– отсутствие у пациентов сводной информации о состоянии своего здоровья и результатах оказанных медицинских услуг</li></ul>  |
| <b>Выигрыш от</b>                  | создания единого хранилища медицинских данных о состоянии здоровья населения и интерфейса к нему   |
| <b>Может состоять в следующем:</b> | <ul style="list-style-type: none"><li>– обеспечении граждан доступом к интегрированной информации о состоянии собственного здоровья;</li><li>– обеспечении персонала медицинских организаций основой для принятия обоснованного медицинского решения о лечении пациента;</li><li>– предоставлении возможности своевременного выявления тенденций в состоянии здоровья населения и обеспечении основы для принятия управленческих решений в сфере здравоохранения;</li><li>– и т.д.</li></ul> |

**i Краткие теоретические сведения и методические указания к Заданию 2**

Определив цели и задачи, можно перейти к выбору подхода (методологии) разработки программного средства. На текущий момент существует множество методологий разработки, поэтому при выборе того или иного подхода необходимо учитывать масштаб программы, квалификацию разработчиков и т.д.

Выделяют следующие группы подходов к разработке [1, стр.89]:

**1 Подходы со слабой формализацией** не используют явных технологий и их можно применять только для очень маленьких проектов, как правило, завершающихся созданием демонстрационного прототипа. К таким подходам относят так называемые ранние технологические подходы.

**2 Строгие (классические, жесткие, предсказуемые)** подходы рекомендуется применять для средних, крупномасштабных и гигантских





проектов с фиксированным объемом работ. Одно из основных требований к таким проектам – предсказуемость.

**3 Гибкие (адаптивные, легкие)** подходы рекомендуется применять для небольших или средних проектов в случае неясных или изменяющихся требований к системе. При этом команда разработчиков должна быть ответственной и квалифицированной, а заказчики должны принимать участие в разработке.

Практически все существующие **проектные методологии** разработки программного обеспечения можно отнести к одному из следующих типов:

- **методологии, ориентированные на план.** Такие методологии направлены на минимизацию неопределенности решения до реализации в целях максимального контроля и минимизации рисков (например, Waterfall). На основе данных методологий реализуются проекты высокой сложности, с большими рисками, или проекты, на которых затруднен доступ к заинтересованным лицам;

- **методологии, ориентированные на изменения.** Направлены на быструю поставку бизнесу рабочего продукта за короткие итерации в ситуациях, когда четкое видение будущего решения не определено (Scrum, Kanban, RAD, RUP, XP, MSF и др.). Как правило, на основе данных методологий реализуются проекты с меньшим риском и возможностью прямого участия заинтересованных лиц и сбора регулярной обратной связи.

Необходимо подчеркнуть, что неверный выбор методологии разработки может привести к финансовым и временным потерям и, тем самым, снижению прибыли компании-разработчика, поэтому важно верно определить методологию разработки.

Этап создания инфраструктуры для разработки программного продукта не менее важен и критичен, чем выбор методологии. Значимость данного этапа обуславливается будущими рисками срывов сроков реализации, потерей выявленных дефектов программы и, как следствие, неполное соответствие конечного программного продукта начальным требованиям заказчика. Также некачественная организация процесса разработки может привести к конфликтам внутри коллектива, что негативно скажется на продуктивности работников.

Обычно под инфраструктурой понимают набор технических и программных средств, которые необходимы для разработки и доставки ПО. Сюда входят способы хранения исходного кода программы и логов, хранения и обработки требований, организации взаимодействия с заказчиком и внутри команды разработки, демонстрации и тестирования разработанного ПО, обеспечения безопасности и масштабирования программы, поставки разработанного программного обеспечения заказчику и прочее.

В рамках выполнения лабораторной работы необходимо представить перечень всех программных утилит, которые будут использоваться при разработке программного средства в виде таблицы 4.



Таблица 4 – Инфраструктура разработки программного средства

| Наименование утилиты | Направление использования   |
|----------------------|-----------------------------|
| 1MeisterTask         | Система управления задачами |
| 2...                 | ...                         |

Также можно представить выбранную инфраструктуру в графическом виде (рисунок 5).

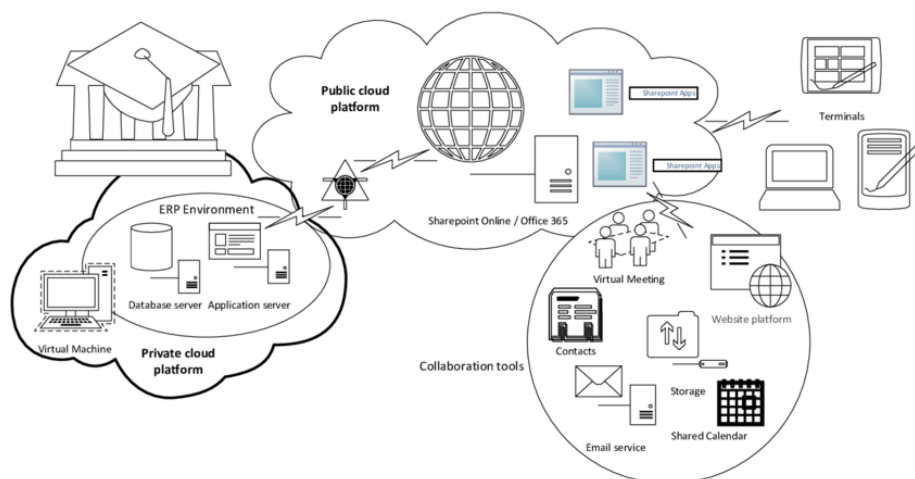


Рисунок 5 – Пример частичной организации инфраструктуры проекта

## Контрольные вопросы к лабораторной работе №1

- 1 Из каких этапов состоит жизненный цикл программного обеспечения?
- 2 Какие задачи решают на этапе планирования проекта по разработке программного обеспечения?
- 3 В чем разница планирования проектов, разрабатываемых по гибким и жестким методологиям?
- 4 Что представляет собой анализ предметной области?
- 5 В чем заключается анализ бизнеса и его продукта и какие техники анализа используют?
- 6 Что важно учитывать при постановке целей по SMART?
- 7 Как выявить ключевые заинтересованные стороны, имеющие отношение к проекту по разработке программного обеспечения?
- 8 Что такое матрица RACI и когда ее следует создавать?
- 9 Какие аспекты следует учесть при определении проблемы программной разработки?
- 10 Что является результатом этапа анализа программного обеспечения?
- 11 Какие проектные методологии ориентированы на изменения?
- 12 Когда следует использовать Waterfall model, а когда Scrum?
- 13 Что понимают под IT-инфраструктурой?




## Лабораторная работа №2

Разработка экономического обоснования проекта.

Построение моделей AS-IS и TO-BE

### Цели выполнения лабораторной работы:

- определить экономическую целесообразность разработки проектного решения
- проанализировать и описать текущее состояние целевых бизнес-процессов, причины проведения изменений и будущее состояние

|  Задание   | Этапы выполнения задания   |
|---|--|
| 1 Выполнить экономическое обоснование проектного решения  | Выполнить экономическое обоснование проекта по разработке программного средства по одному из типовых вариантов: <ul style="list-style-type: none"><li>▪ вариант 1 – по индивидуальному заказу в рамках аутсорсинговой модели;</li><li>▪ вариант 2 – в виде продукта для массового рынка (продуктовая модель);</li><li>▪ вариант 3 – для внедрения в собственные бизнес-процессы разработчика</li></ul> |
| 2 Проанализировать функциональную модель текущего состояния бизнес-процессов, выявить их узкие места и определить преимущества новых бизнес-процессов | <ol style="list-style-type: none"><li>1 Выявить основные (целевые) бизнес-процессы предметной области</li><li>2 Построить модели AS-IS и TO-BE целевых бизнес-процессов с использованием нотации BPMN</li><li>3 Выполнить графический материал (модели «AS-IS» и «TO-BE») в виде плаката и оформить согласно требованиям ЕСПД и ЕСКД</li></ol>   |
| 3 Сформировать отчет по лабораторной работе   | Содержание отчета: <ol style="list-style-type: none"><li>1 Титульный лист (приложение А)</li><li>2 Экономическое обоснование разработки проектного решения</li><li>3 Анализ и моделирование целевых бизнес-процессов предметной области</li><li>4 Графический материал: AS-IS и TO-BE модели процессов предметной области в нотации BPMN</li><li>5 Выводы</li></ol>                                    |



## **Краткие теоретические сведения и методические указания к Заданию 1**

В рамках выполнения лабораторной работы все проектные решения по разработке программного средства сведены к 3-м типовым вариантам.

**Вариант 1.** По индивидуальному заказу в рамках аутсорсинговой модели.

**Вариант 2.** В виде продукта для массового рынка (продуктовая модель).

**Вариант 3.** Для внедрения в собственные бизнес-процессы разработчика.

Студентом проводится экономическое обоснование проектного решения только **для одного типового варианта**. Конкретная методика экономического обоснования выбирается исходя из тематики варианта лабораторной работы.

Для полноценного экономического обоснования проекта необходимо четко сформулировать следующие моменты, важные для коммерциализации проекта:

- кто будет пользователем программного средства;
- какую проблему пользователя будет решать программное средство;
- какие существуют конкурентные программные аналоги;
- почему разрабатываемое программное средство окажется для пользователя более предпочтительным по сравнению с существующими аналогами.



Методические указания по экономическому обоснованию проектного решения по разработке программного средства в рамках 3-х типовых вариантов подробно представлены в источнике [5].

В отчете раздел **Экономическое обоснование проектного решения** в зависимости от выбранного типового варианта проектного решения необходимо представить в одной из следующих редакций:

### **Вариант 1**

#### **1 ЭКОНОМИЧЕСКОЕ ОБОСНОВАНИЕ РАЗРАБОТКИ ПРОГРАММНОГО СРЕДСТВА, РАЗРАБАТЫВАЕМОГО ПО ИНДИВИДУАЛЬНОМУ ЗАКАЗУ**

- 1.1 Характеристика разработанного по индивидуальному заказу программного средства
- 1.2 Расчет затрат на разработку и цена программного средства, созданного по индивидуальному заказу
- 1.3 Расчет результата от разработки и использования программного средства, созданного по индивидуальному заказу



#### 1.4 Расчет показателей экономической эффективности разработки и использования программного средства

### Вариант 2

## 1 ЭКОНОМИЧЕСКОЕ ОБОСНОВАНИЕ РАЗРАБОТКИ И РЕАЛИЗАЦИИ ПРОГРАММНОГО ПРОДУКТА НА МАССОВОМ РЫНКЕ

- 1.1 Характеристика программного средства, разрабатываемого для реализации на рынке
- 1.2 Расчет инвестиций в разработку программного средства для его реализации на рынке
- 1.3 Расчет экономического эффекта от реализации программного средства на рынке
- 1.4 Расчет показателей экономической эффективности разработки и реализации программного средства на рынке

### Вариант 3

## 1 ЭКОНОМИЧЕСКОЕ ОБОСНОВАНИЕ РАЗРАБОТКИ ПРОГРАММНОГО СРЕДСТВА ДЛЯ СОБСТВЕННЫХ НУЖД

- 1.1 Характеристика программного средства, разрабатываемого для собственных нужд
- 1.2 Расчет инвестиций в разработку программного средства для собственных нужд
- 1.3 Расчет экономического эффекта от использования программного средства для собственных нужд
- 1.4 Расчет показателей экономической эффективности разработки и использования программного средства в организации

### **i** Краткие теоретические сведения и методические указания к Заданию 2

При выполнении Задания 2 следует проанализировать и описать:

- текущее состояние целевых бизнес-процессов;
- причины проведения изменений;
- будущее состояние.

Для того чтобы систематизировать текущие бизнес-процессы и описать их состояние необходимо построить **модель «AS-IS»**, которую называют функциональной и выполняют с использованием популярных графических нотаций.

Для создания верхнего уровня модели бизнес-процессов, как правило, используют нотацию IDEF0, обеспечивающую наиболее общее описание объекта моделирования. Нотация IDEF0 используется для построения структурных моделей, показывающих как устроен бизнес организации и раскрывающих информацию о целевых группах бизнес-процессов и их



взаимосвязях. Следует заметить, что данную нотацию можно использовать для моделирования на любом уровне.

На нижнем уровне (операционный уровень выполнения бизнес-процесса, модели WorkFlow) используют нотации BPMN, Процесс, Процедура, ЕРС.

---

#### На практике выделяют бизнес-процессы уровня:



- предприятия (верхний) – бизнес-процессы, которыми управляют топ-менеджеры;
  - управлений/департаментов (средний) – бизнес-процессы на уровне крупных функциональных подразделений предприятия;
  - уровня подразделений/отделов (средний) – функции подразделений и отделов;
  - уровня операций на рабочих местах (нижний) – функции, выполняемые на рабочих местах.
- 

Модель «AS-IS» должна отражать всю полноту информации о бизнес-процессах, существующих на момент обследования предприятия/организации/отдела и позволяет понять, каким образом выполняются бизнес-процессы, а также выявить их узкие места и в конечном итоге разработать мероприятия по их оптимизации/цифровизации.

Анализ функциональной модели позволяет выявить, где находятся наиболее слабые места, в чем будут состоять преимущества новых бизнес-процессов и насколько глубоким изменениям подвергнется существующая структура организации бизнеса.



---

Модель AS-IS должна быть максимально приближена к действительности и основываться на реальных потоках бизнес-процессов, а не на их идеализированном представлении.

Вначале необходимо изучить, как работает предприятие/организация/отдел, выделить целевые бизнес-процессы, сделать их описание и только потом строить модель AS-IS.

---

Пример краткого описания бизнес-процесса предметной области приведен в приложении В.

**Узкие места в модели «AS-IS»** – работы/функции/операции и связи, которые снижают эффективность процесса, увеличивают его трудоемкость и стоимость.

Признаки узких мест<sup>4</sup>:

– бесполезные, неуправляемые и дублирующие работы/функции/операции;

---

<sup>4</sup> Маклаков, С.В. Моделирование бизнес-процессов с BPwin 4.0 [Электронный ресурс]. – Режим доступа: <https://coollib.com/b/157463-sergey-vladimirovich-maklakov-modelirovanie-biznes-protssosov-s-bpwin-40/read>





- нарушение временных рамок выполнения бизнес-процесса;
- неэффективный документооборот (нужный документ не оказывается в нужном месте в нужное время);
- отсутствие обратных связей по управлению (на проведение работы не оказывает влияния ее результат), входу (объекты или информация используются нерационально) и т.д.



Неплохой пример описания слабых мест бизнес-процессов рассматривается в источнике [6].

Для снижения затрат (финансовых, материальных, людских, временных), повышения эффективности функций, которые оказались неэффективными и требовали изменений разрабатывается **модель «ТО-ВЕ»**. Модель «ТО-ВЕ» строится для новых и модифицированных бизнес-процессов.

В рамках выполнения лабораторной работы следует построить функциональные модели («AS-IS» и «ТО-ВЕ») целевых бизнес-процессов предметной области с использованием стандарта BPMN 2.0.

Следует заметить, что данная нотация широко используется бизнес-аналитиками, разработчиками и может применяться не только для моделирования бизнес-процессов, но и для их исполнения.

Пример BPMN-модели бизнес-процессов предметной области представлен на рисунке 6.

**Выполнение графического материала (модели «AS-IS» и «ТО-ВЕ») в виде плаката и оформление согласно требованиям ЕСПД и ЕСКД.** При оформлении плакатов помимо требований ЕСПД и ЕСКД нужно учесть рекомендации, приведенные ниже.

Основная надпись чертежа (плаката) должна содержать поля, указанные на рисунке 7 (размеры основной надписи указаны в ГОСТ 2.104-2006, размещенном по ссылке в источнике [7]).

На рисунке 7 стоит пояснить следующие элементы:

- параметр 1 – регистрационный номер будет частично соответствовать номеру группы (например, для группы 914301 регистрационный номер будет 91431);
- параметр 2 – порядковый номер темы, состоит из двух цифр и будет соответствовать порядковому номеру в списке группы;
- параметр 3 – порядковый номер графического формата, также состоит из двух цифр (для лабораторных работ доступны следующие номера: 01 – AS-IS модель процессов предметной области в нотации BPMN; 02 – ТО-ВЕ модель процессов предметной области в нотации BPMN;

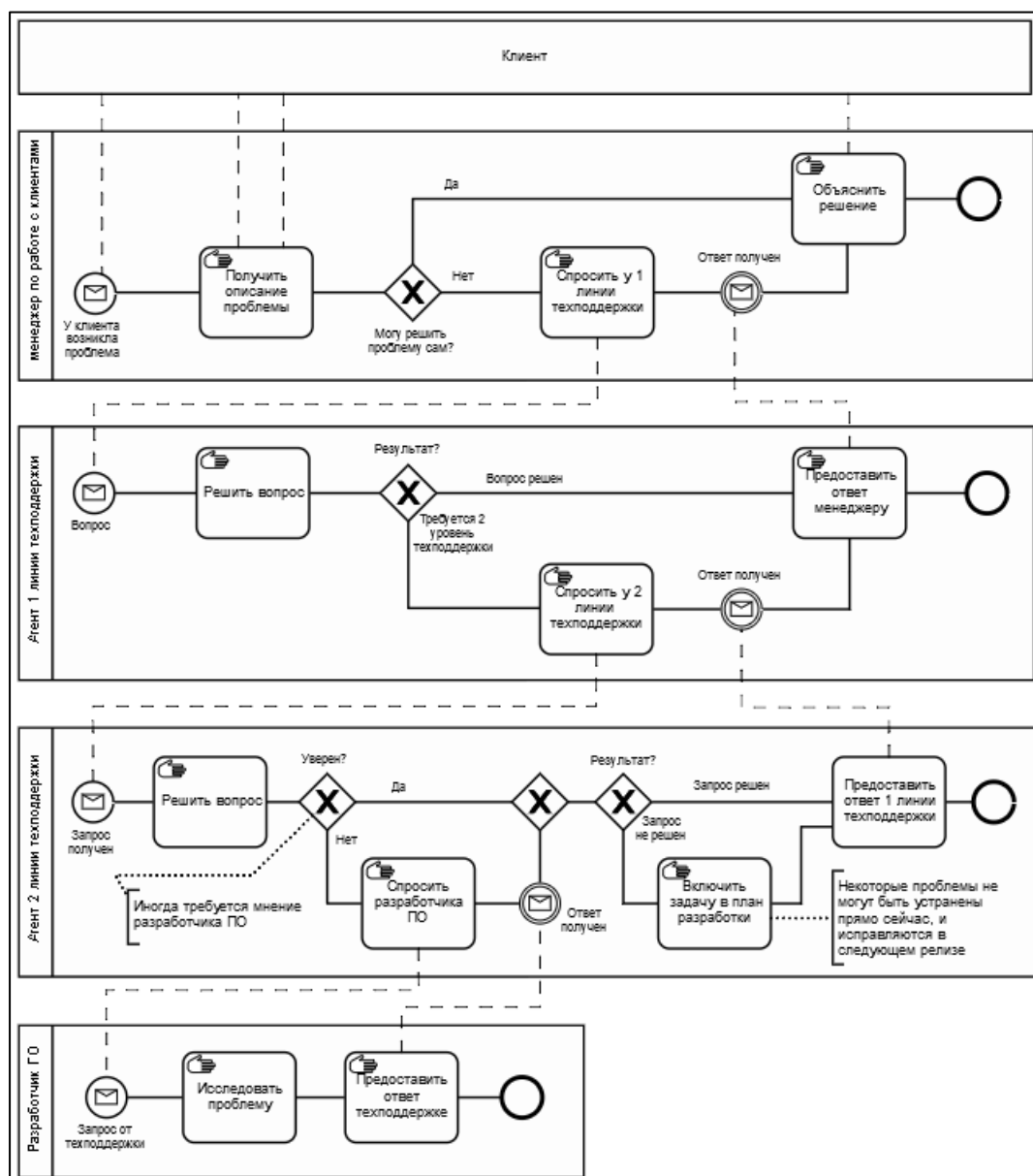


Рисунок 6 – Пример BPMN-модели процессов предметной области

|         |        |          |          |      |   |     |       |         |
|---------|--------|----------|----------|------|---|-----|-------|---------|
|         |        |          |          |      | <b>ГИУР.91431-01 90 01</b>                                      |     |       |         |
|         |        |          |          |      | 1/ 2/ 3/  |     |       |         |
| Изм.    | Лист   | № докум. | Подп.    | Дата | <b>AS-IS модель процессов предметной области в нотации BPMN</b> | Лит | Масса | Масштаб |
| Разраб. | Иванов |          | 23.11.22 | Т    |   |     |       |         |
| Пров.   | Петров |          | 01.12.22 |      |   |     |       |         |
|         |        |          |          |      |   |     |       |         |
|         |        |          |          |      | 4/  |     |       |         |
|         |        |          |          |      | Лист Листов 1   |     |       |         |
|         |        |          |          |      | ПИКС, гр.914301   |     |       |         |

Рисунок 7 – Основная надпись чертежа (плаката)



– параметр 4 – название документа, всегда начинается с имени существительного (названия документов см. в скобках к описанию параметра 3).

## **Контрольные вопросы к лабораторной работе №2**

- 1 Какие аспекты важны для коммерциализации проекта по разработке программного средства?
- 2 В чем заключается экономическое обоснование проектного решения по разработке программного средства по индивидуальному заказу в рамках аутсорсинговой модели?
- 3 В чем заключается экономическое обоснование проектного решения по разработке программного средства в виде продукта для массового рынка?
- 4 В чем заключается экономическое обоснование проектного решения по разработке программного средства для внедрения в собственные бизнес-процессы разработчика?
- 5 Что такое функциональная модель предметной области?
- 6 Для чего проводят анализ функциональной модели?
- 7 Чем различаются модели «AS-IS» и «TO BE»?
- 8 Что представляют собой узкие места в модели «AS-IS» и как их выявляют?
- 9 Какие нотации моделирования используются для построения процессных моделей?
- 10 Для чего используется нотация моделирования BPMN?




## Лабораторная работа №3

Определение функционального назначения и контекста, составление обзора продукта. Формирование технического задания на разработку программного средства

### Цели выполнения лабораторной работы:

- сформировать техническое задание на разработку программного средства (раздел 1)
- сделать постановку задачи на разработку программного средства

|  Задание | Этапы выполнения задания  |
|---|---|
| 1 Сформировать раздел 1 технического задания на разработку                                | 1 Провести обзор существующих аналогов разрабатываемого программного средства<br>2 Определить функциональное назначение и контекст программного средства<br>3 Сформировать раздел 1 технического задания на разработку  |
| 2 Осуществить постановку задачи на разработку   | На основании анализа предметной области и спецификации требований осуществить постановку задачи на разработку программного средства   |
| 3 Сформировать отчет по лабораторной работе   | Содержание отчета:<br>1 Титульный лист (приложение А)<br>2 Таблица аналогов<br>3 Таблица функциональности аналогов<br>4 Таблица функциональности программного средства<br>5 Техническое задание на разработку программного средства (раздел 1)<br>6 Постановка задачи на разработку программного средства<br>7 Выводы |



### Краткие теоретические сведения и методические указания к Заданию 1

Суть сравнительного анализа аналогов разрабатываемого ПС заключается в выделении их достоинств и недостатков, определении ключевых тенденций развития ПО в данной предметной области.

В рамках выполнения лабораторной работы необходимо: провести обзор аналогов на основании требований, изложенных в чек-листе (рисунок 8); представить результаты анализа в соответствии с шаблонами таблиц 5 – 7.




| <div></div> <div><b>ЧЕК-ЛИСТ</b><br/><b>ОБЗОР ФУНКЦИОНАЛЬНОСТИ АНАЛОГОВ</b><br/><b>ПРОГРАММНОГО СРЕДСТВА</b></div> |   |
|---|---|
| <b>ВЫБОР АНАЛОГОВ</b>   | <ul style="list-style-type: none"><li>○ Концентрируемся на актуальных аналогах, входящих в ТОП (обязательно на основе материалов аналитических агентств, исследовательских и консалтинговых компаний, специализирующихся на рынках информационных технологий, например, Tadviser, Gartner, IDC и др.)</li><li>○ Выбираем для обзора не менее 3-х аналогов</li><li>○ Формируем таблицу аналогов:<br/>Название аналога<br/>Компания-разработчик<br/>URL официального сайта компании-разработчика<br/>URL источников для обзора</li></ul>                          |
| <b>ФОРМИРОВАНИЕ ТАБЛИЦЫ ФУНКЦИОНАЛЬНОСТИ АНАЛОГОВ</b>   | <ul style="list-style-type: none"><li>○ Переходим на официальный сайт компании-разработчика и анализируем информационные блоки с требуемым контентом</li><li>○ Формируем таблицу функциональности аналогов:<br/>Аналог (компания-разработчик)<br/>Назначение аналога<br/>Функциональность аналога<br/>Стоимость<br/>Класс аналога<br/>URL страницы с представленной функциональностью</li><li>○ Включаем в список все представленные функции</li><li>○ Систематизируем представленные в таблице сервисы по автоматизации процессов предметной области</li></ul> |
| <b>ФОРМИРОВАНИЕ ТАБЛИЦЫ ФУНКЦИОНАЛЬНОСТИ ПРОГРАММНОГО СРЕДСТВА</b>  | <ul style="list-style-type: none"><li>○ Отбираем функции, которые необходимо реализовать в разрабатываемом программном средстве</li><li>○ Выявляем ключевые тенденции развития программного обеспечения в предметной области</li><li>○ Добавляем свои уникальные функции и дополнения, чтобы программное средство стало привлекательным для конечного пользователя</li><li>○ Формируем таблицу функциональности программного средства с учетом ролей пользователей</li></ul>  |

Рисунок 8 – Чек-лист «Обзор функциональности аналогов программного средства»



Таблица 5 – Аналоги для управления рекрутингом

| Название аналога | Компания-разработчик | URL официального сайта компании-разработчика                    | URL источников для обзора   |
|------------------|----------------------|---|---|
| 1 Mirapolis HCM  | ООО «Мираполис-ИНС/И | <a href="https://www.mirapolis.ru">https://www.mirapolis.ru</a> | <a href="https://www.tadviser.ru/index.php/">https://www.tadviser.ru/index.php/</a> HRM |
| 2 . . .          |                      |   |   |

Таблица 6 – Функциональность аналогов для управления рекрутингом

| Аналог<br>(компания-разработчик)  | Назначение аналога  | Функциональность аналога  | Стоимость  | Класс аналога | URL   |
|---|---|---|------------|---------------|---|
| <b>1 Mirapolis HCM</b><br>(ООО «Мираполис», блок <b>Подбор</b> – в составе Mirapolis HCM) | Автоматизация полного цикла подбора с момента появления потребности до выхода на работу нового сотрудника | <b>1 Единая база и удобный поиск</b><br>кандидатов для внешнего и внутреннего подбора <ul style="list-style-type: none"> <li>Единая база для внешнего и внутреннего подбора</li> <li>Быстрая загрузка резюме из электронной почты или рабочих сайтов с помощью плагинов</li> <li>Разграничение доступа для управления видимостью кандидатов и отдельных событий</li> <li>Гибкий поиск и фильтры (например, по опыту работы, образованию, навыкам и знаниям) позволяют быстро найти нужных кандидатов в базе любого размера</li> <li>Тэги, группировки кандидатов по разным критериям сделают поиск быстрее и создавать лонглисты кандидатов</li> </ul> <b>2 Распознавание резюме</b> <ul style="list-style-type: none"> <li>Резюме с job-сайтов</li> <li>E-mail сообщения</li> <li>Файлы word или pdf</li> <li>любой неструктурированный текст, например, из соцсетей или мессенджеров</li> </ul> <b>3 Тесты и оценка кандидатов</b> <ul style="list-style-type: none"> <li>Проведение тестов или практических заданий на любом этапе подбора</li> <li>Оценка кандидатов по навыкам и компетенциям</li> </ul> | По запросу | HRM-система   | <a href="https://www.mirapolis.ru/recruit/">https://www.mirapolis.ru/recruit/</a> |





| Аналог<br>(компания-разра-<br>ботчик)   | Назначение<br>аналога | Функциональность аналога  | Стои-<br>мость | Класс аналога | URL |
|---|-----------------------|---|----------------|---------------|-----|
| <b>1 Mirapolis HCM</b><br>(ООО «Мираполис»,<br>блок <b>Подбор</b> – в со-<br>ставе Mirapolis HCM) |                       | <ul style="list-style-type: none"><li>• Формирование средней оценки несколькими согласующими</li><li>• Рейтинг кандидатов на основе</li><li>• результатов оценки и тестов</li><li>• Хранение истории тестов и оценок кандида-<br/>тов</li></ul> <b>4 Личные кабинеты</b> <ul style="list-style-type: none"><li>• Кабинет Заказчика: список кандидатов с ука-<br/>занием этапов воронки подбора, запланиро-<br/>ванные и проведенные интервью, форма об-<br/>ратной связи о кандидате, рейтинг кандида-<br/>тов, комментарии для рекрутера, согласо-<br/>вание условий найма</li><li>• Кабинет Рекрутера: настраиваемые филь-<br/>тры кандидатов по этапам воронки подбора и<br/>вакансиям, звонки и отправка сообщений<br/>кандидату с основного экрана, комментиро-<br/>вание, присвоение тегов, просмотр важной<br/>информации без переключения в карточку<br/>кандидата, запланированные действия</li><li>• Кабинет Руководителя подбора:<ul style="list-style-type: none"><li>• согласование и распределение вакансий, ин-<br/>дикация сроков по вакансиям и действиям,<br/>отображение дедлайнов, KPI и загрузки ре-<br/>крутеров</li></ul></li><li>• Кабинет Кандидата: Информация о вакан-<br/>сиях, подача заявки, доступ к необходимым<br/>материалам, прохождение тестирования, об-<br/>мен документами (оффер, анкета кандидата,<br/>документы для оформления)</li></ul> <b>5 Массовый подбор</b> <ul style="list-style-type: none"><li>• Автоматический отбор кандидатов по за-<br/>данным критериям</li><li>• Массовые операции над кандидатами -<br/>приглашения на интервью, проведение<br/>тестирования и опросов</li><li>• Интеграция с голосовыми и чат ботами, си-<br/>стемами видео-интервью, автодозвона до<br/>кандидатов</li></ul> |                |               |     |



| Аналог<br>(компания-разра-<br>ботчик)   | Назначение<br>аналога | Функциональность аналога   | Стои-<br>мость | Класс аналога | URL |
|---|-----------------------|--|----------------|---------------|-----|
| <b>1 Mirapolis HCM</b><br>(ООО «Мираполис»,<br>блок <b>Подбор</b> – в со-<br>ставе Mirapolis HCM) |                       | <b>6 Настройка воронки подбора</b> <ul style="list-style-type: none"> <li>• Настройка маршрута согласования заявки на подбор</li> <li>• Последовательное или параллельное согласование</li> <li>• Разные процессы согласования заявок (например, на штатную и внештатную должность, с отклонением от установленной ЗП и без)</li> <li>• Создание и изменение собственных этапов воронки подбора и сроков их прохождения</li> <li>• Фиксированные этапы воронки или с возможностью выбора следующего при согласовании</li> <li>• Создание и настройка уведомлений к любым этапам и событиям подбора</li> </ul> <b>7 Аналитика</b> <ul style="list-style-type: none"> <li>• Библиотека готовых отчетов и возможность создания собственных</li> <li>• Дашборды по ключевым показателям</li> <li>• Наиболее эффективные источники подбора</li> <li>• Эффективность команды подбора и каждого участника</li> <li>• Скорость закрытия вакансий</li> <li>• Стоимость подбора и множество других</li> <li>• параметров, необходимых для управления процессом подбора</li> </ul> <b>8 Возможность интеграции</b> <ul style="list-style-type: none"> <li>• Плагины для парсинга резюме из электронной почты и при поиске на job- сайтах</li> <li>• Интеграция со всеми job-сайтами</li> <li>• Интеграция с ip-телефонией и электронной почтой</li> </ul> |                |               |     |



| Аналог<br>(компания-разра-<br>ботчик)   | Назначение<br>аналога | Функциональность аналога   | Стои-<br>мость | Класс аналога | URL |
|---|-----------------------|--|----------------|---------------|-----|
| <b>1 Mirapolis HCM</b><br>(ООО «Мираполис»,<br>блок <b>Подбор</b> – в со-<br>ставе Mirapolis HCM) |                       | <ul style="list-style-type: none"><li>• Интеграция с учетными системами, такими как IC, SAP, Босс Кадровик, Парус и другими</li><li>• Широкие возможности API позволяют интегрироваться с чат-ботами, системами для проведения видео-интервью, автодозвонами и другими системами</li></ul> |                |               |     |
| <b>2. ...</b>   |                       |  |                |               |     |



Таблица 7 – Функциональность программного средства с учетом ролей пользователей

| Функции  | Доступность   |                |
|--|---|----------------|
|  | пользователю 1  | пользователю 2 |
| <b>основные:</b>                                   |   |                |
| поддержка ЭЦП                                      | да  | нет            |
| создание / изменение регистрационных карточек      | да, есть встроенный редактор  |                |
| ...  |   |                |
| <b>дополнительные:</b>                             |   |                |
| создание шаблонов документов, для вывода на печать | нет   | да             |
| полнотекстовый поиск по вложенным файлам           | да, поддерживаются форматы MS Office, Adobe PDF, HTML, XML, текстовые файлы |                |
| ...  | ...   | ...            |

**Техническое задание (ТЗ)** на разработку является отправной точкой в реализации проекта по разработке любого программного изделия. На данный момент существуют стандарты, как зарубежные (ISO IEC IEEE 29148-2018), так и отечественные (группа стандартов серии ГОСТ 19 и ГОСТ 34), определяющие элементы и содержание технического задания.

В рамках лабораторной работы необходимо составить раздел 1 технического задания (а именно раздел «Введение»). Пример полного шаблона ТЗ<sup>5</sup> представлен на рисунке 9.

Рассмотрим кратко составные части задания<sup>6</sup>:

- **функциональное назначение** – описание назначения программного средства;
- **область применения (иначе границы проекта)** – сфера деятельности, в которой предполагается применение разрабатываемого программного средства;
- **общий взгляд на программное изделие** – описать контекст и происхождение программного изделия;
- **функции программного изделия** – показать основные (самые глобальные) функции, которые должно предоставлять разрабатываемое ПО;
- **характеристики пользователей** – описать группы пользователей, которые будут использовать разработанное программное изделие и влиять на удобство использования (указать опыт, уровень образования и прочие факторы, которые могут стать ограничениями при разработке ПО);

<sup>5</sup> Детальное пояснение каждого пункта ТЗ представлено в источниках [8, гл.9.6] и [9, гл.10].

<sup>6</sup> Полу жирным выделены пункты ТЗ, которые являются обязательными при составлении ТЗ



- 1 Введение
  - 1.1 Функциональное назначение
  - 1.2 Область применения
  - 1.3 Обзор продукта
    - 1.3.1 Общий взгляд на программное изделие
    - 1.3.2 Функции программного изделия
    - 1.3.3 Характеристики пользователей
    - 1.3.4 Операционная среда
  - 1.4 Ограничения дизайна и реализации
  - 1.5 Предположения и зависимости
  - 1.6 Ссылки
- 2 Спецификация требований
  - 2.1 Внешние интерфейсы
    - 2.1.1 Пользовательские интерфейсы
    - 2.1.2 Интерфейсы ПО
    - 2.1.3 Интерфейсы оборудования
    - 2.1.4 Коммуникационные интерфейсы
  - 2.2 Функции
    - 3.2.x Название функции программного средства
      - 3.2.x.1 Описание
      - 3.2.x.2 Функциональные требования
  - 2.3 Требования к юзабилити
  - 2.4 Требования к производительности
  - 2.5 Требования к данным
    - 2.5.1 Логическая модель данных
    - 2.5.2 Словарь данных
    - 2.5.3 Отчеты
    - 2.5.4 Получение, целостность, хранение и утилизация данных
  - 2.6 Ограничения проектирования
  - 2.7 Системные атрибуты программного обеспечения
    - 2.7.1 Надежность
    - 2.7.2 Доступность
    - 2.7.3 Безопасность
    - 2.7.4 Удобство сопровождения/эксплуатации
    - 2.7.5 Переносимость
- 3 Верификация
- 4 Требования по интернационализации и локализации
- 5 Вспомогательная информация
- 6 Приложения
  - 6.1 Словарь терминов
  - 6.2 Сокращения и аббревиатуры

Рисунок 9 – Пример ТЗ на разработку программного изделия

– **операционная среда** – привести описание всех ограничений, которые могут влиять на работу будущего программного средства (например, аппаратные ограничения, версии используемых ОС, существующие политики безопасности, интерфейсы взаимодействия со сторонними компонентами, требования к качеству разработанного ПО, географическое расположение пользователей и прочее);

– **ограничения дизайна и реализации** – указать языки программирования (ЯП), библиотеки, модули и прочие компоненты, которые были использованы или разработаны заказчиком ранее и которые необходимо использовать при разработке исследуемого проектного решения;

– **предположения и зависимости** – представить все суждения, которые предполагаются верными в отсутствии знаний или доказательств иного, а также



описание всех внешних факторов и компонентов, которые могут повлиять на работу и которые находятся вне контроля разрабатываемого программного средства;

- **ссылки** – указать ссылки на все документы, из которых была взята информация для составления ТЗ (также можно разместить данный раздел в п. 5);

- **внешние интерфейсы** – предоставить информацию, которая позволит быть уверенным, что разработанное программное средство будет правильно взаимодействовать с пользователями, а также с внешними как программными, так и аппаратными компонентами (обязательным является заполнение **п. 2.1.2. Интерфейсы ПО**);

- **функции** – описать функциональные требования к разрабатываемому программному средству;

- **требования к юзабилити** – описать требования к удобству использования программного средства;

- **требования к производительности** – указать требования, предъявляемые к различным системным операциям;

- **требования к данным** – описать все детали данных, которые система будет использовать в качестве входных (обязательными для заполнения являются **пункты 2.5.1 Логическая модель данных и 2.5.2 Словарь данных**);

- **ограничения проектирования** – указать ограничения, которые накладывают на проектирование программного средства внешние стандарты и нормативные документы;

- **системные атрибуты программного обеспечения** – определить такие атрибуты программного средства, как надежность, доступность, переносимость и прочее;

- **верификация** – представить способы, методы и методики проверки соответствия разработанного ПО требованиям заказчика (допускается использование нескольких видов тестирования совместно);

- **требования по интернационализации и локализации** – описать возможности, позволяющие использовать разработанное программное средство в странах (районах), которые отличаются от локации его создания;

- **вспомогательная информация** – включить в данный подраздел любую другую информацию, которая не вошла в предыдущие разделы, но является важной для разработки программного средства;

- **словарь терминов** – определить перечень специальных терминов, использованных в ТЗ;

- **сокращения и аббревиатуры** – представить специфические сокращения и аббревиатуры, использованные в ходе разработки ТЗ и программного средства.





## **① Краткие теоретические сведения и методические указания к Заданию 2**

Основная цель постановки задачи – указать назначение и цели создания программного средства (таблица 8). Цель, в свою очередь, показывает, что заказчик хочет видеть на выходе.

Таблица 8 – Описание назначения и целей создания программного средства

| <b>ПОСТАНОВКА ЗАДАЧИ</b> |  |
|--------------------------|--|
| <b>Назначение:</b>       | Указать перечень объектов автоматизации, на которых предполагается использовать программу/систему, перечень автоматизируемых органов (пунктов) управления объекта автоматизации и управляемых ими объектов (здесь указать в каких подразделениях предусматривается устанавливать программу/систему и привести в разрезе подразделений перечень автоматизируемых бизнес-процессов верхнего уровня). |
| <b>Цели создания:</b>    | Наименования и требуемые значения технических, технологических, производственно-экономических или других показателей объекта автоматизации, которые должны быть достигнуты в результате создания программы; критерии оценки достижения целей создания программы/системы.   |

**Пример 1.** Описание постановки задачи для предметной области «Разработка информационно-аналитической системы «Корпоративное хранилище данных» (КХД)»

**Назначение:** КХД предназначена для повышения оперативности и качества принимаемых управленческих решений сотрудниками Заказчика.

Основным назначением КХД является автоматизация информационно-аналитической деятельности в бизнес-процессах Заказчика.

В рамках проекта автоматизируется информационно-аналитическая деятельность в следующих бизнес-процессах:

- анализ финансово-хозяйственной деятельности;
- информационная поддержка процессов бюджетирования;
- ...

**Цели создания:** КХД создается с целью:

- обеспечения сбора и первичной обработки исходной информации, необходимой для подготовки отчетности по показателям деятельности;
- создания единой системы отчетности по показателям деятельности;
- повышения качества (полноты, точности, достоверности, своевременности, согласованности) информации;
- ...



В результате создания хранилища данных должны быть улучшены значения следующих показателей:

- время сбора и первичной обработки исходной информации;
- количество информационных систем, используемых для подготовки аналитической отчетности;
- время, затрачиваемое на информационно-аналитическую деятельность;
- ...

**Пример 2.** Описание постановки задачи для предметной области «Обновление приложения Smart TV и Android TV»

Приложение Smart и Android TV являются продуктами холдинга «Москва Медиа», через которые осуществляется дистрибуция аудиовизуального контента. Программы передач каналов Москва 24 и Москва Доверие в виде архивов, а также прямой эфир (только Москва 24) доступны в приложении.

Основная задача продуктов – информирование локальной (московской) аудитории о происходящих событиях в регионе, а также репрезентации контента, который удовлетворяет широкие потребности данной аудитории. В связи с этим необходимо обновить существующие версии приложений и улучшить их функционал. Предполагается, что обновленные версии приложений будут более востребованы для конечного пользователя и соответствовать интересам холдинга Москва Медиа.

Предполагаемое время работ по обновлению приложений – август/сентябрь 2017 года.

Предполагаемое время релиза приложений – 4 квартал 2017 года.

### Контрольные вопросы к лабораторной работе №3

- 1 Для чего необходимо Техническое задание в проекте?
- 2 В чем заключаются принципиальные отличия Технического задания по ГОСТам 19 и 34?
- 3 Какой раздел Технического задания на разработку можно считать основным и почему?
- 4 Что представляет собой Обзор продукта?
- 5 Как правильно сделать постановку задачи на разработку программного средства?




## Лабораторная работа №4

### Анализ требований и разработка спецификации требований к программному средству

#### Цели выполнения лабораторной работы:

- составить документ о концепциях и границах
- сформировать техническое задание на разработку программного средства (разделы 2-6)

|  Задание | Этапы выполнения задания  |
|---|---|
| 1 Разработать документ о концепциях и границах  | 1 Изучить типы требований по К. Вигерсу и смоделировать их<br>2 Оформить основные бизнес-требования в виде документа о концепции и границах проекта (vision and scope document)<br>3 Представить пользовательские требования в виде вариантов использования |
| 2 Разработать разделы 2-6 технического задания на разработку                              | 1 Разработать спецификацию требований (функциональных и нефункциональных) к программному средству<br>2 Сформировать разделы 2-6 технического задания на разработку  |
| 3 Сформировать отчет по лабораторной работе   | 1 Титульный лист (приложение А)<br>2 Документ о концепции и границах (vision and scope document).<br>3 Техническое задание на разработку программного средства (разделы 2-6)<br>4 Выводы  |



#### Краткие теоретические сведения и методические указания к Заданию 1

В основе любой программной системы, а также процесса ее разработки лежат требования к программному обеспечению.



**Требование** – описание того, какие функции и с соблюдением каких условий программная система должна выполнять в процессе решения полезной для пользователя задачи.

Требования должны быть: полные (достаточные); однозначные (недвусмысленные); необходимые (зачем-то кому-то); осуществимые (в данных



условиях); приоритизированные (по некоторому критерию); проверяемые (можно ли проверить реализацию); модифицируемые; корректны.

Основные типы требований по К. Вигерсу представлены на рисунке 10 [9].

Как видно из рисунка 10, требования к ПО рассматриваются на трех уровнях – бизнес-требования, пользовательские и функциональные требования. В каждой из перечисленных групп требований имеются свои нефункциональные требования.

**Бизнес-требования** – это высокоуровневая бизнес-цель организации или заказчиков системы.

Бизнес-требования выражают цель, ради которой разрабатывается продукт (зачем вообще он нужен, какая от него ожидается польза).

#### Примеры бизнес-требований:

1 Программное средство должно обеспечивать возможность поиска кандидатов по должности, специальности и географическому региону.

2 Нужен инструмент, в реальном времени отображающий наиболее выгодный курс покупки и продажи валюты.

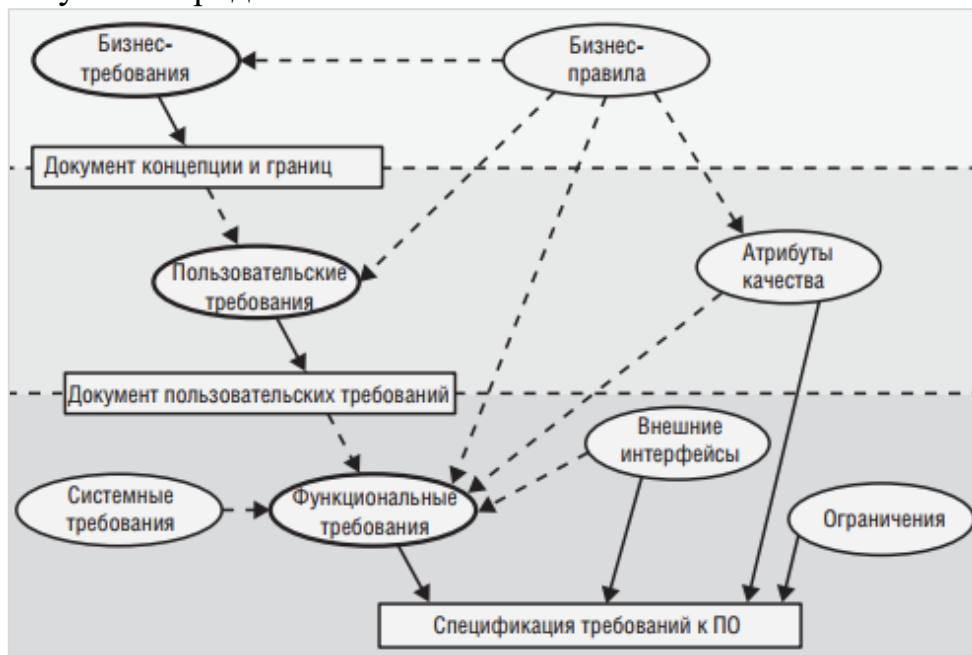


Рисунок 10 – Классификация требований по К. Вигерсу

3 Увеличить среднее эффективное рабочее время каждого сотрудника на 15 минут в день в течение 6 месяцев после первого выпуска системы.

Как правило, бизнес-требования выявляются на этапе анализа предприятия. Результатом выявления бизнес-требований является **Документ о концепции и границах** (vision and scope document).

Документ о концепции и границах (vision and scope document) собирает бизнес-требования в единый документ, который подготавливает основу для последующей разработки продукта.

Концепция и границы – два базовых элемента бизнес-требований.



**Концепция продукта** (product vision) сжато описывает конечный продукт, который достигнет заданных бизнес-целей, т.е. что продукт представляет собой сейчас, и каким он станет впоследствии.

**Границы проекта** (project scope) показывают, на какую часть конечной концепции продукта будет направлен текущий проект или итерация

---

Рекомендуемый шаблон для описания концепции продукта (видение решения)

Для <целевая аудитория>,

**которая** <описание потребностей или возможностей>,

**этот продукт** <имя разрабатываемого продукта>



**является** <категория продукта>,

**который** <ключевое преимущество, основная причина для использования>.

**В отличие от** <основные конкуренты, текущая система или бизнес-процесс>,

**наш продукт** <перечисление основных отличий и преимуществ разрабатываемого продукта>

---

### Пример описания концепции продукта

Для сотрудников, **которым** нужно заказывать еду в кафе, **данная** Cafe Ordering System **является** приложением для смартфона, **которое** принимает индивидуальные или групповые заявки на питание, взимает оплату и инициирует доставку готовых блюд к указанной компании.

**В отличие от** имеющихся в настоящее время служб заказа по телефону, **наш продукт** позволит сотрудникам не приходить в кафетерий, чтобы получать заказанные блюда, что сэкономит им время, и кроме того, увеличится ассортимент доступных им блюд.



Для моделирования границ проекта можно использовать контекстную диаграмму, диаграмму потоков данных (DFD).

---

Рекомендуемый шаблон документа о концепции и границах представлен ниже.

## 1 Бизнес-требования

1.1 Исходные данные

1.2 Возможности бизнеса

1.3 Бизнес-цели

1.4 Критерии успеха

1.5 Положение о концепции проекта (Видение решения)

1.6 Бизнес-риски

1.7 Предположения и зависимости

## 2 Рамки и ограничения проекта

2.1 Основные функции

2.2 Объем первоначально запланированной версии

2.3 Объем последующих версий



## 2.4 Ограничения и исключения

### 3 Бизнес-контекст

#### 3.1 Профили заинтересованных лиц

#### 3.2 Приоритеты проекта

#### 3.3 Особенности развертывания



Отличный пример оформления **Документа о концепции и границах проекта** приведен в Приложении В источника [9].

В рамках лабораторной работы следует представить **Документ о концепции и границах** следующей структуры:

### 1 Бизнес-требования

#### 1.1 Исходные данные

#### 1.2 Возможности бизнеса

#### 1.3 Бизнес-цели

#### 1.4 Положение о концепции проекта (Видение решения)

### 2 Рамки и ограничения проекта

### 3 Бизнес-контекст

При формировании разделов данного документа следует ознакомиться с пояснительным материалом, представленным на страницах 91-110 источника [9], а также воспользоваться примером, приведенным в Приложении В [9].

В разделе 2 следует представить только подраздел 2.1 Основные функции.

В разделе 3 следует представить только подраздел 3.1 Профили заинтересованных лиц.

**Пользовательские требования** описывают задачи, которые пользователь может выполнять с помощью разрабатываемой системы.

#### **Примеры пользовательских требований:**

1 Бухгалтер может генерировать отчеты о наличии и движении МЦ в разрезах.

2 Менеджер по снабжению должен управлять информацией о поставщиках.

3 Диспетчер имеет возможность регистрировать новый заказ в системе.

Пользовательские требования оформляются в виде *вариантов использования, пользовательских историй, пользовательских сценариев*.



**Диаграмма вариантов использования** – это наиболее общее представление функционального назначения системы с точки зрения получения значимого результата для пользователя.

Цели использования Use Case диаграммы:

– определить общие границы и контекст моделируемой предметной области;

– сформулировать общие требования к функциональному поведению





проектируемой системы;

- разработать исходную концептуальную модель системы для ее последующей детализации в форме логических и физических моделей;
- подготовить исходную документацию для взаимодействия разработчиков системы с ее заказчиками и пользователями.

### Принять во внимание при разработке диаграммы вариантов использования:

- Поведение системы следует описывать только с помощью основных вариантов использования, отражающих уровень базовых пользовательских требований.

Частая ошибка: «Выдать деньги» в банкомате не является вариантом использования, а «Снять деньги» – это вариант использования, который включает в себя последовательность действий по выдаче денег.



- Use Case диаграмма не описывает требования подробно, не показывает последовательность выполнения варианта. Диаграмма Use Case используется для выяснения требований к системе, фиксации этих требований в форме, которая позволит проводить их дальнейшее моделирование и разработку.
- Не злоупотреблять многократным использованием стереотипов «include» и «extend». Декомпозировать вариант использования до функций – грубейшая ошибка!
- Примеры неудачной и корректной диаграммы вариантов использования – рисунок 11.

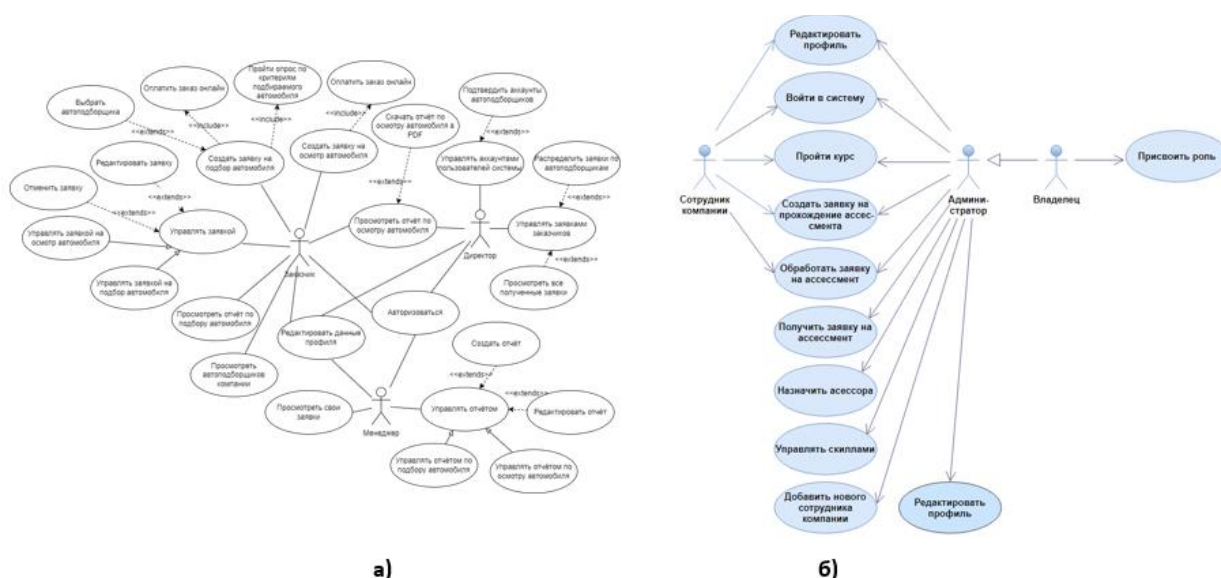


Рисунок 11 – Примеры а) неудачной и б) корректной диаграммы вариантов использования

В рамках лабораторной работы для описания пользовательских требований необходимо представить **спецификацию вариантов использования**. Для этого



необходимо:

- разработать UML диаграмму вариантов использования, содержащую не менее **десяти** Use Case, исключая тривиальные операции работы с БД (добавление, удаление, редактирование записей в БД);
- представить описание вариантов использования.

### Принять во внимание!



На диаграмме вариантов использования не представлять категорию пользователей «Администратор», за исключением случаев, когда программное средство разрабатывается в целях создания рабочего места администратора

Описать варианты использования можно с помощью:

- 1 Шаблона, представленного в Приложении В источника [9].
- 2 Сценария использования. Пример заполненного шаблона описания варианта использования сценарием представлен в таблице 9.

Таблица 9 – Пример описания варианта использования сценарием

| УС-1 Зарегистрироваться в системе                                   |  |
|---|--|
| Действующее лицо  | Слушатель  |
| Краткое описание варианта использования                             | Данный вариант использования описывает вход слушателя в систему регистрации курсов   |
| Входные условия   | Отсутствуют  |
| Основной поток действий при исполнении варианта использования       | 1 Слушатель хочет войти в систему регистрации курсов.<br>2 Система запрашивает имя пользователя и пароль.<br>3 Слушатель вводит имя и пароль.<br>4 Система проверяет имя и пароль, после чего открывается доступ в систему   |
| Альтернативный поток действий при исполнении варианта использования | 1 Если во время выполнения основного потока обнаружится, что слушатель ввел неправильное имя и/или пароль, система выводит сообщение об ошибке.<br>2 Система запрашивает имя пользователя и пароль.<br>3 Слушатель вводит имя и пароль.<br>4 Система проверяет имя и пароль, после чего открывается доступ в систему |
| Выходные условия  | 1 Если вариант использования выполнен успешно, слушатель входит в систему.<br>2 В противном случае состояние системы не изменяется   |



## Краткие теоретические сведения и методические указания к Заданию 2

**Системные требования** описывают высокоуровневые требования к программному обеспечению, содержащему несколько или много взаимосвязанных подсистем и приложений.

### Примеры пользовательских требований:

- 1 Приложение является десктопным.
- 2 Для работы приложение использует интерпретатор PHP.
- 3 Приложение является кроссплатформенным.

**Функциональные требования** – положение о фрагменте требуемой функциональности или поведения системы при определенных условиях.

### Примеры функциональных требований:

- 1 Система должна автоматически выполнять резервное копирование данных ежедневно в указанный момент времени.
- 2 Электронный адрес пользователя, вводимый при регистрации, должен быть проверен на соответствие требованиям RFC822.

Функциональное требование строится по формуле:



«ID->Условие->Система должна->действие»

При этом ID включает аббревиатуру – сокращение от названия элемента (обычно на английском) и порядковый номер с учетом иерархии.

Для документирования функциональных требований может быть использован шаблон, представленный таблицей 10 либо в Приложении В источника [9].

Таблица 10 – Шаблон для документирования функциональных требований

| ID  | Название функции | Роль | Описание | Функциональные требования |
|-----|------------------|------|----------|---------------------------|
| UC1 |                  |      |          | FR1-1<br>FR1-2<br>...     |
| UC2 |                  |      |          | FR2-1<br>FR2-2<br>...     |

**Нефункциональные требования** – описание свойств или характеристик, которые система должна демонстрировать, или ограничения, которые она должна соблюдать.

Нефункциональные требования описывают свойства системы (удобство использования, безопасность, надёжность, расширяемость и т. д.), которыми она должна обладать при реализации своего поведения.

К группе нефункциональных требований относят: *бизнес-правила*,



*атрибуты качества, ограничения, внешние интерфейсы, требования к данным.*

**Бизнес-правило** – это положение, определяющее или ограничивающее какие-либо стороны бизнеса.

Бизнес-правила описывают особенности принятых в предметной области и/или непосредственно у заказчика процессов, ограничений и иных правил.

Эти правила могут относиться к бизнес-процессам, правилам работы сотрудников, нюансам работы ПО и т.д. Это могут быть корпоративные регламенты, политики, стандарты, законодательные акты, которые подразумевают организацию структуры бизнеса, контролируют ведение бизнеса или влияют на бизнес.

Выделяют следующие типы бизнес-правил: факт, ограничение, активатор, вычисление, вывод, термин.

#### **Примеры бизнес-правил:**

1 Никакой документ, просмотренный посетителями сайта хотя бы один раз, не может быть отредактирован или удалён.

2 Подключение к системе извне офиса запрещено в нерабочее время.

3 Интернет-магазин обрабатывает заказы покупателей только до 19:00.

**Атрибуты качества** – устанавливают параметры, которые будут использоваться для оценки работы системы.

Атрибутов качества большое количество, но для любого проекта реально важными являются лишь некоторые их подмножества: надежность, переносимость, масштабируемость, удобство и др.

#### **Примеры атрибутов качества:**

1 95% новых пользователей должны суметь успешно ввести заказ без ошибок с первой попытки.

2 Внесённые в текст статьи изменения не должны быть утеряны при нарушении соединения между клиентом и сервером.

3 Система должна выводить пользователю сообщение о подтверждении в среднем за 3 секунды и не более чем через 6 секунд после того, как пользователь отослал информацию системе.

**Ограничения** – условия или требования, которые накладываются на доступные разработчику возможности проектирования или разработки системы.

Ограничения представляют собой факторы, ограничивающие выбор способов и средств (в том числе инструментов) реализации продукта.

Например, производительность, платформа реализации и развертывания, протоколы взаимодействия с внешними системами и т.п.

#### **Примеры ограничений:**

1 Система должна быть реализована на платформе .NET Framework 4.0.

2 Не допускается использование Flash при реализации клиентской части приложения.

3 Система должна использовать текущую версию СУБД Oracle, являющуюся корпоративным стандартом.

**Внешние интерфейсы** – интерфейсы взаимодействия с внешними



системами или операционной средой.

### **Примеры внешних интерфейсов:**

1 Обмен данными между клиентской и серверной частями приложения при осуществлении фоновых AJAX-запросов должен быть реализован в формате JSON.

2 Интернет-магазин должен иметь интеграцию с PayPal для оплаты товаров пользователями.

3 Система должна опрашивать систему учета запасов кафетерия для определения наличия запрашиваемого блюда.

Виды интерфейсов: *интерфейс пользователя, программные интерфейсы, интерфейсы оборудования, коммуникационные интерфейсы.*

**Требования к данным** – определяют список данных, которые используются для описания элементов в разрабатываемой системе.

Часто сюда относят описание базы данных и особенностей её использования.

### **Примеры требований к данным:**

1 Все данные системы, за исключением пользовательских документов, должны храниться в БД под управлением СУБД MySQL; пользовательские документы должны храниться в БД под управлением СУБД MongoDB.

2 Информация о кассовых транзакциях за текущий месяц должна храниться в операционной таблице, а по завершении месяца переноситься в архивную.

Пользовательские, функциональные и нефункциональные требования обычно включаются в SRS, с которым работают заинтересованные лица.



В рамках выполнения лабораторной работы составление спецификации требований к программному средству будет состоять в формировании разделов 2-6 ТЗ на разработку (см. рисунок 8).

Детальное пояснение содержания данных разделов ТЗ представлено в гл. 9.6 источника [8] и гл.10, Приложении В источника [9].

Для документирования бизнес-правил может быть использован шаблон, представленный таблицей 11.



Таблица 11 – Шаблон для документирования бизнес-правил

| Идентификатор | Определение правила   | Тип правила <sup>7</sup> | Статическое или динамическое | Источник           |
|---------------|---|--------------------------|------------------------------|--------------------|
| BR-1          | Периоды доставки – это 15-минутные интервалы, начинающиеся каждую четверть часа     | Факт                     | Динамическое                 | Менеджер кафетерия |
| BR-2          | Доставка всех заказов должна быть завершена между 10:00 и 14:00 по местному времени | Ограничение              | Динамическое                 | Менеджер кафетерия |

Для документирования нефункциональных требований может быть использован шаблон, представленный таблицей 12.

Таблица 12 – Шаблон для документирования нефункциональных требований

| Идентификатор требования                 | Описание  |
|--|---|
| <b>Пользовательские интерфейсы:</b>      |   |
| UI-1                                     | Экраны Cafeteria Ordering System должны соответствовать «Process Impact Internet Application User Interface Standard, Version 2.0»        |
| UI-2                                     | Система должна обеспечивать ссылку на справку на каждой HTML-странице, объясняющую, как пользоваться этой страницей                       |
| <b>Интерфейсы программного средства:</b> |   |
| SI-1                                     | Cafeteria Ordering System должна передавать количество единиц заказанных блюд системе учета запасов кафетерия через программный интерфейс |
| SI-2                                     | Cafeteria Ordering System должна опрашивать систему учета запасов кафетерия для определения наличия запрашиваемого блюда                  |
| ...                                      | ...   |

<sup>7</sup> Вигерс с.197



## Контрольные вопросы к лабораторной работе №4

- 1 Что представляет собой Документ о концепции и границах (vision and scope document) и когда его следует разрабатывать?
- 2 На каких уровнях рассматривают требования к программному обеспечению?
- 3 Какие существуют методы выявления требований?
- 4 На каком этапе определяются бизнес-требования?
- 5 В чем отличия бизнес-правил от требований к системе?
- 6 На какие типы классифицируют бизнес-правила?
- 7 Что такое SRS и каковы ее ключевые элементы?
- 8 В чем принципиальная разница между требованием и потребностью с точки зрения бизнес-анализа?
- 9 Что представляют собой нефункциональные требования и как их фиксировать?
- 10 Какой уровень детализации традиционно применяется при описании варианта использования?





## ПРИЛОЖЕНИЕ Б (обязательное)

### Варианты заданий для выполнения лабораторных работ

1. Проектирование и разработка программного средства поддержки процессов управления байкшерингом.
2. Проектирование и разработка программного средства управления предприятием по ремонту электротехники и продаже комплектующих на базе анализа бизнес-процессов.
3. Проектирование и разработка программного средства тестирования знаний студентов на базе Quiz-сервиса.
4. Проектирование и разработка программного средства поддержки процессов бронирования в индустрии гостеприимства.
5. Проектирование и разработка программного средства реализации интегрированных моделей и алгоритмов обработки данных по учету продаж розничной и оптовой торговли предприятия.
6. Проектирование и разработка программного средства реализации информационного процесса оформления и обработки заказов в интернет-магазине.
7. Проектирование и разработка программного средства поддержки организации управления персоналом на предприятиях сферы ресторанного бизнеса.
8. Проектирование и разработка программного средства реализации мобильного сервиса внутренней навигации в помещении.
9. Проектирование и разработка программного средства планирования туристических маршрутов с возможностью их оптимизации.
10. Проектирование и разработка программного средства реализации бизнес-модели free-to-play.
11. Проектирование и разработка программного средства автоматизации процессов оформления заказа и учета медицинского оборудования клиники.
12. Проектирование и разработка программного средства реализации онлайн-сервиса медицинского обслуживания в поликлинике.
13. Проектирование и разработка программного средства поддержки интеллектуального процесса подбора оптимального варианта объекта недвижимости.
14. Проектирование и разработка программного средства учета и анализа повременно-премиальной оплаты труда работников предприятия.
15. Проектирование и разработка программного средства поддержки процессов мониторинга и анализа дезинсекционных мероприятий санитарно-профилактических служб.
16. Проектирование и разработка программного средства реализации мобильного сервиса для мониторинга авиарейсов в реальном времени.
17. Проектирование и разработка программного средства автоматизации процедур мониторинга и управления процессами складского учета предприятия.
18. Проектирование и разработка программного средства автоматизации процессов заказа и бронирования авиабилетов на основе информационных и телекоммуникационных технологий.



19. Проектирование и разработка программного средства автоматизация процедур мониторинга техник тестирования программного обеспечения с оценкой трудозатрат.

20. Проектирование и разработка программного средства поддержки системы управления клининговой компанией и автоматизация процедуры мониторинга заявок клиентов.

21. Проектирование и разработка программного средства автоматизации процедур анализа и оценки эффективности инвестиционных проектов.

22. Проектирование и разработка программного средства оптимизации процесса взаимодействия банка с физическими лицами и его программная реализация для сектора обслуживания по вкладам.

23. Проектирование и разработка программного средства поддержки бизнес-процессов автосервиса с разработкой модуля управления взаимодействием с клиентами.

24. Проектирование и разработка программного средства реализации онлайн сервиса по организации культурных мероприятий и бронированию билетов.

25. Проектирование и разработка программного средства управления системой дисконта предприятия.

26. Проектирование и разработка программного средства автоматизации процедур мониторинга и управления командой IT-проекта.

27. Проектирование и разработка программного средства автоматизации функций оперативного учета и отчетности по лизинговым операциям

28. Проектирование и разработка программного средства обработки запросов пользователей и ранжирования информации в интернет-магазине.

29. Проектирование и разработка программного средства визуализации информационных процессов деятельности киностудии.

30. Проектирование и разработка программного средства автоматизации бизнес-процессов фитнес-студии и мониторинга эффективности выполнения заказов.

31. Проектирование и разработка программного средства поддержки и анализа процессов управления документооборотом поликлиники.

32. Проектирование и разработка программного средства поддержки онлайн сервиса управления взаимоотношениями с клиентами в ветеринарных центрах

33. Проектирование и разработка программного средства поддержки процесса перекрестных продаж в B2B сегменте.

34. Проектирование и разработка программного средства управления онлайн-подписками на печатные издания.

35. Проектирование и разработка программного средства поддержки процесса создания стартового пакета для развертывания Web-приложений на основе технологий Facebook.

36. Проектирование и разработка программного средства поддержки кроссплатформенного карточного игрового приложения с элементами монетизации.



37. Проектирование и разработка программного средства реализации сервиса визуализации данных на основе оптимизационных моделей и экспертного оценивания.

38. Проектирование и разработка программного средства поддержки процессов реализации краудфандинговых проектов в сфере международного туризма.

39. Проектирование и разработка программного средства сбора статистических данных о прохождении слушателями обучающего курса.

40. Проектирование и разработка программного средства автоматизации процессов оценки профессиональных компетенций персонала в IT-компаниях на основе технологии бизнес-консалтинга.

41. Проектирование и разработка программного средства динамического планирования генерального бюджета структурных подразделений предприятия.

42. Проектирование и разработка программного средства управления взаимодействием транспортных компаний и организаций-грузовладельцев в логистических системах.

43. Проектирование и разработка программного средства учета и контроля товарных запасов розничного магазина оператора сотовой связи.

44. Проектирование и разработка программного средства автоматизации управления жилыми комплексами и бизнес-центрами на основе IoT-технологии.