АО «НИЦ СПб ЭТУ»

Утвержден
СМК 7.3.2-10ПР-ЛУ

ДОКУМЕНТИРОВАННАЯ ПРОЦЕДУРА

Проектирование архитектуры программного обеспечения

СМК 7.3.2-10ПР

Подпись и дата	
Инв. № дубл.	
Взам. инв. №	
Подпись и дата	
Инв. № подл.	

Введена в действие приказом от 15.10.2024 № 101/1-О
Всего листов (страниц) 41
Экземпляр №

Содержание

1 Область применения

Настоящая документированная процедура устанавливает порядок выполнения процесса проектирования архитектуры программного обеспечения.

Настоящая документированная процедура направлена на выполнение требований ГОСТ Р ИСО 9001-2015 (8.3), ГОСТ РВ 0015-002-2020 (8.3), ОСТ 134-1028-2012 изм. 2 (8.3) и развивает положения документа СМК 4.2.2-01РК-2006.

Настоящая документированная процедура распространяется на порядок проектирования архитектуры программного обеспечения изделий, разрабатываемых акционерным обществом «Научно-инженерный центр Санкт-Петербургского электротехнического университета» (далее – АО «НИЦ СПб ЭТУ»).

Настоящая документированная процедура обязательна для применения во всех подразделениях АО «НИЦ СПб ЭТУ» при выполнении работ по контрактам (договорам) на выполнение опытно-конструкторской работы (составной части опытно-конструкторской работы), научно-исследовательской работы (составной части научно-исследовательской работы) в том числе при взаимодействии с предприятиями-соисполнителями.

2 Нормативные ссылки

В настоящей документированной процедуре использованы нормативные ссылки на следующие стандарты и документы системы менеджмента качества:

ГОСТ 19.101-2024 Виды программ и программных документов

ГОСТ 19781-90 Обеспечение систем обработки информации программное. Термины и определения

ГОСТ Р 57100-2016 Системная и программная инженерия. Описание архитектуры

ГОСТ Р 56136-2014 Управление жизненным циклом продукции военного назначения. Термины и определения

ГОСТ Р ИСО 9001-2015 Системы менеджмента качества. Требования

ГОСТ РВ 0015-002-2020 Система разработки и постановки на производство военной техники. Системы менеджмента качества. Требования

ОСТ 134-1028-2012 изм. 2 Ракетно-космическая техника. Требования к системам менеджмента качества предприятий, участвующих в создании, производстве и эксплуатации изделий

СМК 4.2.2-01РК-2006 Руководство по качеству

СМК 4.2.3-01ПР Управление документацией системы менеджмента качества

СМК 7.3.1-02ПР Управление проектной деятельностью

СМК 7.3.2-06ПР Разработка программного обеспечения. Порядок выполнения

СМК 7.3.3-02ПР База знаний. Регламент работы

СМК 7.3.4-04ПР Детальное проектирование. Порядок выполнения

Положение о дистанционной (удаленной) работе работников акционерного общества «Научно-инженерный центр Санкт-Петербургского электротехнического университета»

П р и м е ч а н и е – При пользовании настоящим стандартом целесообразно проверить действие ссылочных стандартов (классификаторов) в информационной системе общего пользования – на официальном сайте Федерального агентства по техническому регулированию и метрологии в сети Интернет или по ежегодному информационному указателю «Национальные стандарты», который опубликован по состоянию на 1 января текущего года, и по выпускам ежемесячного информационного указателя «Национальные стандарты» за текущий год. Если на документ дана недатированная ссылка, то следует использовать документ, действующий на текущий момент, с учетом всех внесенных в него изменений. Если заменен ссылочный документ,

на который дана датированная ссылка, то следует использовать указанную версию этого документа. Если после принятия настоящего документа в ссылочный документ, на который дана датированная ссылка, внесено изменение, затрагивающее положение, на которое дана ссылка, то это положение применяется без учета данного изменения. Если ссылочный документ отменен без замены, то положение, в котором дана ссылка на него, применяется в части, не затрагивающей эту ссылку.

3 Термины определения

В настоящей документированной процедуре применены следующие термины с соответствующими определениями:

3.1 **артефакт:** По СМК 7.3.3-01ПР.

3.2 **архитектура (системы):** Основные понятия или свойства системы в окружающей среде, воплощенной в ее элементах, отношениях и конкретных принципах ее проекта и развития [ГОСТ Р 57100, статья 3.2].

3.3 **база знаний:** По СМК 7.3.3-02ПР.

3.4 **брокер:** Сервис централизованного обмена сообщениями.

3.5 **внешняя сущность:** Объект за пределами моделируемой системы, который является отправителем или получателем данных.

3.6 **диаграмма компонентов:** Концептуальная картина взаимодействия между различными системами (подсистемами).

3.7 **диаграмма развертывания:** Графическое изображение устройств, процессов и связей между ними.

3.8 **жизненный цикл:** Совокупность явлений и процессов, повторяющаяся с периодичностью, определяемой временем существования типовой конструкции изделия от ее замысла до утилизации или конкретного экземпляра изделия от момента завершения его производства до утилизации [ГОСТ Р 56136, статья 3.16].

3.9 **коммуникационная ассоциация**: Путь связи между узлами.

3.10 **основной метод процесса проектирования:** Визуальное моделирование, которое позволяет получить полное представление о всей проектируемой системе и отдельных ее компонентах.

3.11 **паттерн:** Шаблон проектирования (повторяемая архитектурная конструкция в сфере проектирования программного обеспечения, предлагающая решение проблемы проектирования в рамках некоторого часто возникающего контекста).

3.12 **порт:** Точка входа или выхода компонента.

3.13 **поток данных:** Непосредственно данные, которые входят в процессы и хранилища или выходят из них.

3.14 **предоставляемый интерфейс:** Возможности или функции, которые предоставляет конкретный компонент для взаимодействия с другими компонентами или системами.

3.15 **программа:** Совокупность команд и данных, обеспечивающая выполнение заданной последовательности действий средствами вычислительной техники [ГОСТ 19.101, статья 3.1].

3.16 **программный комплекс:** Программа, состоящая из двух или более программных компонентов и/или программных комплексов, выполняющих взаимосвязанные функции [ГОСТ 19.101, статья 3.3].

3.17 **программное обеспечение:** Совокупность программ системы обработки информации и программных документов, необходимых для эксплуатации этих программ [ГОСТ 19781, статья 1].

3.18 **протоколы информационного взаимодействия (протоколы):** Описание и последующее согласование особенностей взаимодействия между различными системами или подсистемами одной системы.

3.19 **процесс проектирования:** Систематический способ продвижения от абстрактных концепций к конкретным техническим деталям.

3.20 **процесс:** Функция или действия по обработке данных.

3.21 **подсистема:** Часть системы с некоторыми связями и отношениями.

3.22 **руководитель проекта:** По СМК 7.3.1-02ПР.

3.23 **система управления версиями:** Программное обеспечение для облегчения работы с изменяющейся информацией. Система управления версиями позволяет хранить несколько версий одного и того же документа, при необходимости возвращаться к более ранним версиям, определять, кто и когда сделал то или иное изменение.

3.24 **системный аналитик проекта:** По СМК 7.3.1-02ПР.

3.25 **системный архитектор проекта:** По СМК 7.3.1-02ПР.

3.26 **спецификация развертывания:** Файл конфигурации.

3.27 **требуемый интерфейс:** Интерфейс, который необходим компоненту от своего окружения для выполнения заявленной функциональности, контракта или поведения.

3.28 **узел:** Физическая сущность, которая включает один или несколько компонентов, подсистем или исполняемых файлов.

3.29 **узел-контейнер:** Виртуальная единица, представляющая собой машину, которая работает над запуском контейнеров с приложениями.

3.30 **устройство:** Узел, который используется для представления физического вычислительного ресурса в системе.

3.31 **хранилище данных:** Источник, приемник или промежуточное хранилище данных внутри моделируемой системы.

3.32 **proto-файл:** Текстовый формат с определённым строгим синтаксисом, определяющий интерфейс взаимодействия модулей программного обеспечения.

4 Сокращения

В настоящей документированной процедуре приняты следующие сокращения:

АНЦ	– астрономический научный центр;
АПК АДМ-О	– объектовый аппаратно-программный комплекс обработки данных мониторинга околоземного космического пространства;
АПК ОДМ-У	– удаленный аппаратно-программный комплекс обработки данных мониторинга околоземного космического пространства;
АПУ	– автоматизированный пункт управления;
АРМ	– автоматизированное рабочее место;
БД	– база данных;
ГрСК	– гринвичская система координат;
ДПО	– департамент программного обеспечения;
ДУТиБА	– департамент управления требованиями и бизнес-анализа;
ИБ	– информационная безопасность;
МУ РС	– модуль управления расширенного состава;
НЗ	– носитель записи;

ОБА	– отдел бизнес-анализа;
ОКР	– опытно-конструкторская работа;
ОС	– операционная система;
ОС СН	– операционная система специального назначения;
ПК	– программный комплекс;
ПК ИВ	– программный комплекс информационного взаимодействия;
ПО	– программное обеспечение;
СлИБ	– служба информационной безопасности;
СМК	– система менеджмента качества;
СН	– сеанс наблюдения;
СР	– средство регистрации;
СУБД	– система управления базами данных;
СЧ	– составная часть;
ТЗ	– техническое задание;
ТП	– техническое проектирование;
ТТЗ	– тактико-техническое задание;
ФХ	– файловое хранилище;
ЭП	– эскизное проектирование;
BPMN	– Business Process Model and Notation;
DFD	– Data Flow Diagrams;
IP	– Internet Protocol;
OSI	– Open Systems Interconnection;
SVN	– свободная централизованная система управления версиями;
TCP/IP	– Transmission Control Protocol;
VPP	– Visual paradigm project.

5 Ответственность

Начальник департамента программного обеспечения несет ответственность за своевременное и качественное выполнение работ подчиненных подразделений, участвующих в разработке архитектуры ПО.

Начальник департамента управления требованиями и бизнес-анализа несет ответственность за своевременное и качественное выполнение работ подчиненных подразделений, участвующих в разработке архитектуры ПО.

Системный архитектор проекта несет ответственность за соблюдение требований настоящей документированной процедуры в целом и отвечает за технические решения на основе материалов системного аналитика проекта:

– потоки данных;

– развёртывание;

– разделение на компоненты;

– протоколы взаимодействия;

– реализация требований.

Системный аналитик проекта несет ответственность за соблюдение требований настоящей документированной процедуры в части предоставления исходных данных для проектирования системы:

– анализ бизнес-процессов;

– анализ профилей пользователей;

– разработка сценариев использования системы;

– выявление и фиксация требований;

– определение модели предметной области.

Руководитель проекта осуществляет контроль за соблюдение системным архитектором проекта и системным аналитиком проекта требований настоящей документированной процедуры в части:

– разработки функциональной спецификации;

– соблюдения ограничений проекта;

– соглашений с заказчиком о способе и порядке реализации требований;

– синхронизации работы всей проектной команды.

6 Проектирование архитектуры программного обеспечения

6.1 Общие положения

6.1.1 В процессе проектирования архитектуры ПО производится формирование и анализ функциональных требований будущей системы (набор функций, предоставляемых системой), нефункциональных требований (надёжность, производительность, расширяемость, совместимость и другие), а также анализ существующих ограничений, которые могут возникнуть в процессе разработки системы или ее подсистем.

6.1.2 Процесс проектирования архитектуры ПО начинается с методичного анализа профилей пользователей, которые описывают различные типы пользователей (включая персонал сопровождения) и их рабочие функции. Часть этой работы может проводиться во время процесса анализа (выработки концепции). Так как выбранная модель разработки (в соответствии с документом СМК 7.3.2-06ПР) ориентирована на варианты использования, то в качестве метода описания требований к системе, а также в качестве естественной единицы для дальнейшего планирования и оценки выполнения работ выступают сценарии использования.

6.1.3 Сценарии использования позволяют выявлять реальные потребности будущих пользователей системы и отслеживать полноту описания этих требований. В наборе сценариев использования моделируется выполнение какой-либо операции определенным типом пользователя (например, подготовка оператором программного комплекса условно-постоянных данных по пуску). Далее каждый сценарий использования разбивается на последовательность специфических действий, называемых вариантами использования, которые необходимо выполнить пользователю для осуществления операции.

6.1.4 Исходными данными для проектирования архитектуры ПО являются результаты анализа предметной области, которые дают понимание требований и содержат общую структуру разрабатываемой системы. Основными результатами анализа, необходимыми для проектирования ПО можно считать:

– сценарии использования;

– аналитическая модель предметной области;

– нефункциональные требования.

6.1.5 Описание архитектуры ПО имеет следующие назначения:

– создание общего представления о системе «с нуля» любому заинтересованному лицу;

– средство трассировки реализации исходных требований. Описывает связь функциональных/нефункциональных требований и компонентов, реализующих их;

– средство коммуникации для описания точки соприкосновения интересов разных заинтересованных сторон. В процессе разработки стороны приводятся архитектором во взаимодействие для согласования и разрешения противоречивых требований;

– базис для системного анализа и проектирования.

6.1.6 Основной метод процесса проектирования архитектуры ПО позволяет получить представление о таких компонентах, как:

– бизнес процессы автоматизируемой предметной области;

– требования к будущей системе;

– состояния и процессы взаимодействия моделируемых объектов;

– логическое представление системы, позволяющее определить структуру разрабатываемых классов.

6.1.7 Средства моделирования архитектуры ПО используют языки моделирования (UML/BPMN/DFD), представляющие собой набор графических нотаций для визуализации, спецификации, конструирования и документирования ПО.

6.1.8 Описанием архитектуры системы на различных этапах пользуются практически все участники проектной команды, поэтому различные архитектурные представления должны позволить найти соответствующей проектной роли ответы именно на свои вопросы, касающиеся различных архитектурных аспектов. Наиболее заинтересованные проектные роли:

– архитекторы;

– аналитики;

– конструкторы;

– разработчики;

– тестировщики;

– пользователи.

6.1.9 Результатом проектирования является архитектура ПО и наброски модели реализации. Когда в ходе этапа разработки архитектура ПО окончательно стабилизируется, и требования будут четко определены, на первый план выйдут задачи реализации.

6.2 Общий вид процесса проектирования архитектуры программного обеспечения

6.2.1 В общем виде процесс проектирования архитектуры ПО представляется BPMN-диаграммой (приложение А).

6.2.2 Основные процессы проектирования архитектуры ПО представлены в разделе 8 (8.5) документа СМК 7.3.4-04ПР.

6.3 Архитектура системного уровня

Архитектура ПО имеет несколько представлений:

– логическое представление системы;

– процессное представление системы;

– физическое представление системы.

6.3.1 Логическое представление системы

6.3.1.1 Описание системы должно быть представлено как список высокоуровневых компонентов, соответствующих удовлетворению отдельных функциональных требований

или их групп, представляющий собой самостоятельные модули для развёртывания. От них должна идти детализация структуры описанием включаемых компонентов до уровня, определяемого стадией разработки архитектуры. Также логическое представление помогает в трассировке распределения функциональных обязанностей внутри системы и обратной трассировки от компонента до функционального (нефункционального) требования, в реализации которого он участвует. Логическое представление дает общее видение о системе и подробно описывает:

– организацию компонентов системы;

– взаимодействие компонентов системы;

– зависимость компонентов системы друг от друга.

Логическое представление позволяет системному архитектору проекта увидеть, работает ли система, достигает ли она своих целей. Компонент представляет отдельную часть системы, которая выполняет определенную функцию или имеет определенную роль. Он может быть программным модулем, классом, библиотекой, сервисом, физическим устройством и т. д. Выделяя компоненты, следует помнить, что компоненты автономны. Компоненты – элементы модульной системы, т. е. могут быть заменены альтернативой.

6.3.1.2 Логическое представление системы чаще всего выполняется при помощи диаграммы компонентов. Диаграмма компонентов дает концептуальную картину взаимодействия между различными системами (частями одной системы). Общие принципы разработки диаграммы компонентов:

– компоненты должны иметь интерфейсы (требуемые и/или предоставляемые). Интерфейсы также показывают связи и зависимости в архитектуре системы:

1) требуемый интерфейс указывает на интерфейсы, которые компонент ожидает от других компонентов;

2) предоставляемый интерфейс на диаграмме компонентов указывает на то, какие возможности или функции предоставляет конкретный компонент для взаимодействия с другими компонентами или системами;

– порт может быть безымянным, может иметь имя, тип (у порта может быть указана его кратность);

– компоненты могут иметь иерархическую связь. Компонент более высокого уровня (система, подсистема) может включать компоненты более низкого уровня (подсистемы, модули).

6.3.1.3 Основные элементы диаграммы компонентов приведены на рисунке 1.

6.3.1.4 Пример диаграммы компонентов приведен на рисунке 2.

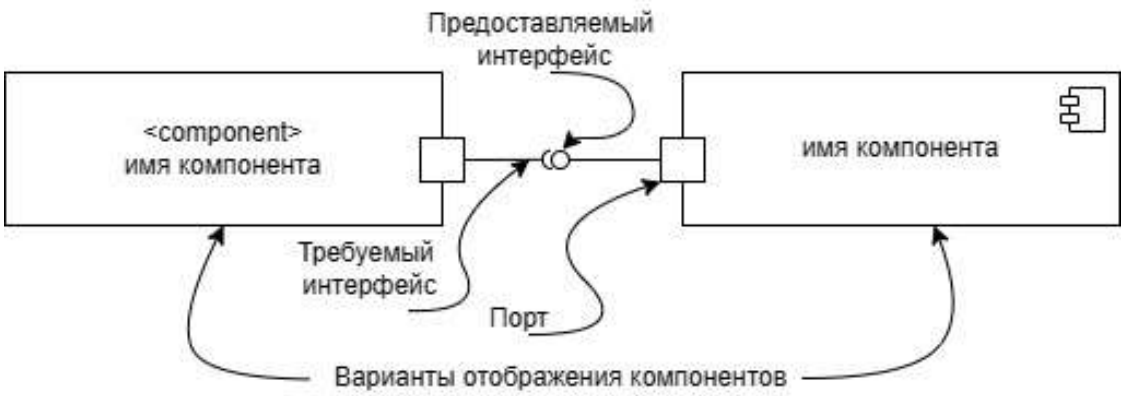

Рисунок 1

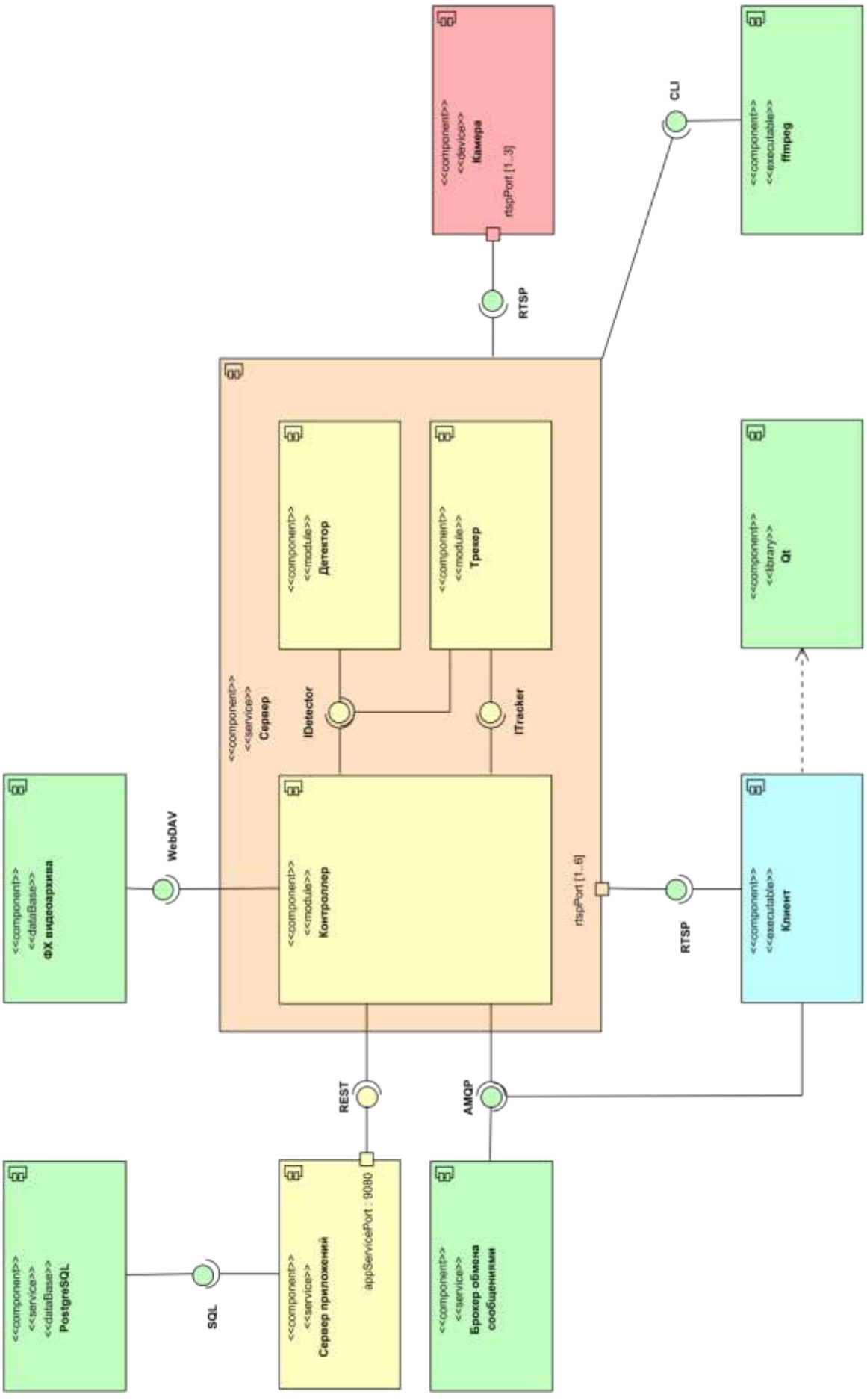
<<component>>
<<device>>
Камера
rtspPort [1..3]
RTSP
CLI
<<component>>
<<executable>>
ffmpeg
<<component>>
<<module>>
Детектор
<<component>>
<<module>>
Трекер
<<component>>
<<library>>
Qt
<<component>>
<<service>>
Сервер
IDetector
ITracker
<<component>>
<<dataBase>>
ФХ видеоархива
WebDAV
<<component>>
<<module>>
Контроллер
rtspPort [1..6]
RTSP
<<component>>
<<executable>>
Клиент
REST
AMQP
<<component>>
<<service>>
PostgreSQL
SQL
<<component>>
<<service>>
Сервер приложений
appServicePort : 9080
<<component>>
<<service>>
Брокер обмена
сообщениями

Рисунок 2

6.3.1.5 Рекомендации по оформлению диаграмм компонентов.

Для наглядности и удобочитаемости рекомендуется:

– изменять цвета диаграмм компонентов, устанавливаемые в Visual Paradigm по умолчанию;

– задавать различные цвета:

1) для компонентов верхнего уровня и внутренних компонентов;

2) для компонентов стандартных библиотек, приложений и разрабатываемых компонентов;

3) для компонентов АО «НИЦ СПб ЭТУ» и компонентов сторонних взаимодействующих организаций.

6.3.1.6 Диаграмма компонентов дополняется описанием компонентов в базе знаний проекта. Описание компонента может состоять из следующих разделов:

– имя компонента – название компонента согласно документации, или внутреннее имя (например, имя файла с соответствующим модулем);

– описание – краткая характеристика компонента (назначение, выполняемые роли, возможно ссылки на технические решения, согласно которым он спроектирован, процессы, для поддержки которых он используется, и алгоритмы, которые в нём выполняются);

– реализуемые функции – список идентификаторов функций из функционального описания, которые реализуются данным компонентом;

– включаемые компоненты – список компонентов, входящих в состав описываемого компонента. Элементы должны представлять собой ссылки на документы, отвечающие настоящим требованиям;

– используемые компоненты – список компонентов, используемых данным компонентом в процессе работы, от которых есть прямая зависимость.

6.3.2 Процессное представление системы

6.3.2.1 Процессное представление системы имеет целью продемонстрировать, как каждый процесс преобразует свои входные данные в выходные, а также выявить отношения между этими процессами. Процессное представление позволяет определить порядок хранения данных в процессе обработки, а также состав и формат данных, передаваемых между процессами, что позволяет сформировать протоколы взаимодействия между различными подсистемами разрабатываемой системы и между разрабатываемой системой и внешними взаимодействующими системами.

6.3.2.2 Процессное представление системы чаще всего выполняется при помощи диаграммы потоков данных DFD (далее – диаграмма DFD). Диаграмма DFD описывает внешние по отношению к системе источники и адресаты данных, логические функции системы, потоки данных и хранилища данных, к которым осуществляется доступ. Диаграммы DFD обеспечивают удобный способ описания передаваемой информации, как между частями моделируемой системы, так и между системой и внешним миром.

6.3.2.3 Основные элементы диаграммы DFD:

– внешняя сущность;

– процесс;

– поток данных;

– хранилище данных.

6.3.2.4 Диаграмма DFD не привязана к хронологии выполнения сценариев и к логическому представлению системы (компонентам), а ориентирована на моделирование потоков и функций преобразования данных.

6.3.2.5 Диаграмма DFD направлена на логическое представление передачи и обработки данных, а не на физические потоки данных. Например, в случае с брокером не следует отражать процесс передачи данных через брокер, заводя на него все потоки. Достаточно указать на стрелке потока «AMQP».

6.3.2.6 Основные элементы диаграммы DFD приведены на рисунке 3.

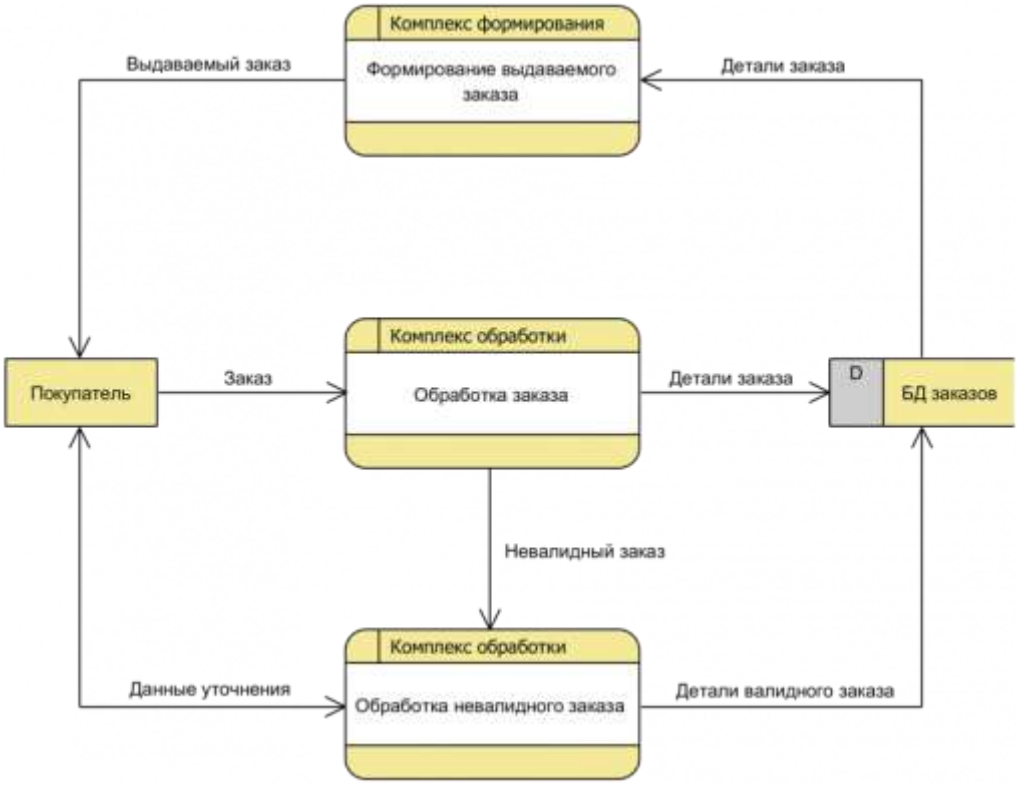

Рисунок 3

6.3.2.7 Пример диаграммы DFD приведен на рисунке 4.

6.3.2.8 Рекомендации по оформлению диаграмм DFD:

– для элементов процессов указывать его размещение (location) и т. п.;

– для потоков данных указывать характеристику потока (протокол передачи и формат передаваемых данных);

– при необходимости задавать различные цвета:

1) для процессов различного размещения;

2) для процессов и хранилищ данных АО «НИЦ СПб ЭТУ» и процессов и хранилищ данных сторонних взаимодействующих организаций;

– избегать применения слишком большого количества цветов (больше трех или четырех);

– в случае большой и сложной системы:

1) выделять процессы одной группы не цветами, а группировкой внутри вспомогательных визуальных элементов. Например, именованных прямоугольников;

2) разрабатывать диаграмму на значимый сценарий работы системы, чтобы избежать перегрузки процессами обработки данных, которые никогда в одно время в системе не существуют.

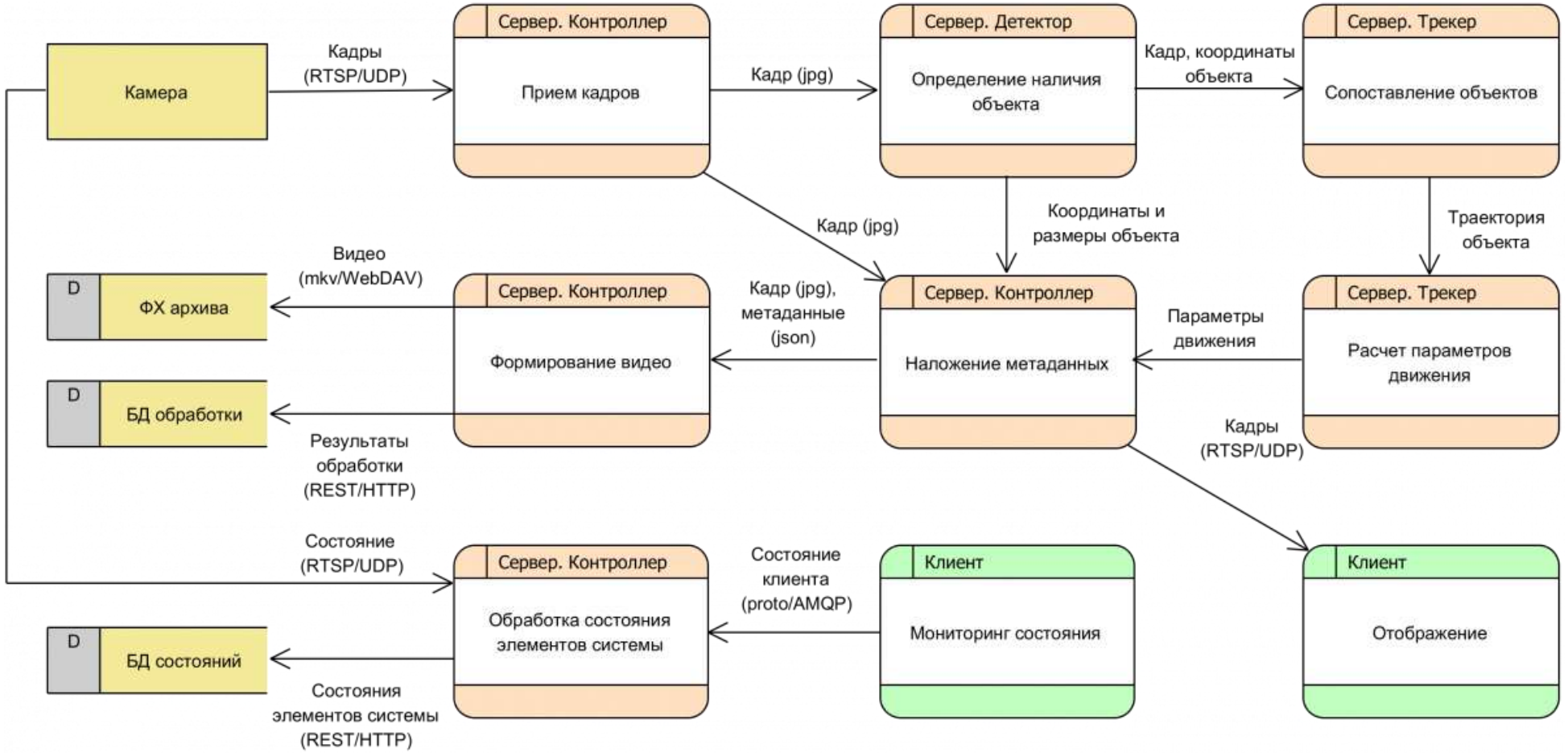
Камера
Кадры (RTSP/UDP)
Сервер. Контроллер
Прием кадров
Кадр (jpg)
Сервер. Детектор
Определение наличия объекта
Кадр, координаты объекта
Сервер. Трекер
Сопоставление объектов
Кадр (jpg)
Координаты и размеры объекта
Траектория объекта
D
ФХ архива
Видео (mkv/WebDAV)
Сервер. Контроллер
Формирование видео
Кадр (jpg), метаданные (json)
Сервер. Контроллер
Наложение метаданных
Параметры движения
Сервер. Трекер
Расчет параметров движения
D
БД обработки
Результаты обработки (REST/HTTP)
Кадры (RTSP/UDP)
Состояние (RTSP/UDP)
Сервер. Контроллер
Обработка состояния элементов системы
Состояние клиента (proto/AMQP)
Клиент
Мониторинг состояния
Клиент
Отображение
D
БД состояний
Состояния элементов системы (REST/HTTP)

Рисунок 4

6.3.2.9 Графическое описание диаграммы DFD дополняется текстовым в базе знаний проекта. Описание элементов диаграммы DFD может состоять из следующих разделов:

– ссылки на протоколы информационного взаимодействия с внешними сущностями;

– ссылки на описание схем БД, строения ФХ и т. д. для хранилищ данных;

– ссылки на форматы временных файлов;

– ссылки на алгоритмы преобразования данных.

6.3.3 Физическое представление системы

6.3.3.1 Цель физического представления системы – продемонстрировать общую конфигурацию или топологию распределенной системы и отразить информацию о размещении различных артефактов по отдельным узлам системы.

Представление описывает:

– физические узлы, необходимые для размещения на них исполнимых компонентов системы;

– связи между узлами реализации системы на этапе ее исполнения (например, зависимость работы одного узла от другого);

– среды исполнения и общесистемное ПО, необходимое для реализации функций системы.

6.3.3.2 Физическое представление системы чаще всего выполняется при помощи диаграммы развертывания. Диаграмма развертывания содержит графические изображения устройств, процессов и связей между ними.

В отличие от диаграмм логического представления, диаграмма развертывания является единой для системы в целом, поскольку должна всецело отражать особенности ее реализации. При разработке диаграмм развертывания выявляются узкие места системы и планируются пути реконфигурации её топологии для достижения требуемой производительности.

6.3.3.3 Основные элементы диаграммы развертывания:

– узел – узел может быть аппаратным или программным элементом. Узлы могут быть показаны на диаграмме в виде типа или в виде экземпляра. Если один узел содержит другие узлы, то такой узел называют узлом-контейнером;

– артефакт – может включать другие артефакты. Артефакты связаны между собой пунктирными стрелками зависимостей (возможно, через интерфейсы). Такие связи обозначают, что один компонент использует услуги другого. Если артефакт соответствует компоненту, рекомендуется для этого артефакта добавлять стереотип «component»;

– коммуникационная ассоциация;

– устройство;

– спецификация развертывания описывает множество свойств, которые определяют параметры выполнения артефакта компонента, развертываемого на некотором узле.

6.3.3.4 Основные элементы диаграммы развертывания приведены на рисунке 5.

6.3.3.5 Пример диаграммы развертывания приведен на рисунке 6.

6.3.3.6 Рекомендации по оформлению диаграмм развертывания.

При оформлении диаграмм рекомендовано:

– изменять цвета диаграмм, устанавливаемые в Visual Paradigm по умолчанию;

– задавать различные цвета:

1) для устройств, узлов и артефактов верхнего и вложенных уровней;

2) для артефактов стандартных библиотек, приложений и разрабатываемых артефактов;

3) для устройств, узлов и артефактов АО «НИЦ СПб ЭТУ» и сторонних взаимодействующих организаций;

– учитывать в том числе общесистемное ПО, важное для функционирования системы (web-сервера, системы обнаружения вторжений и т. п.).

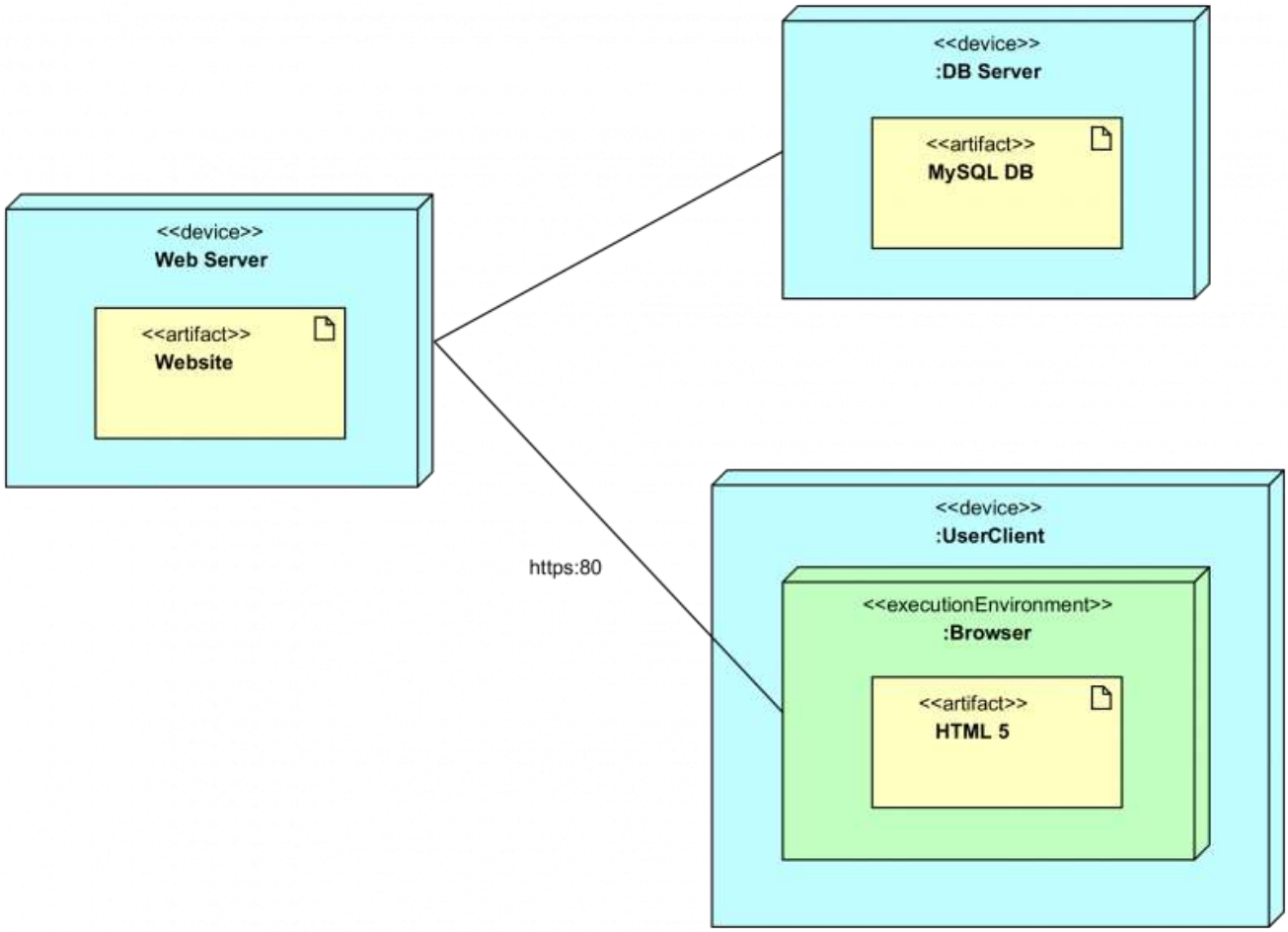

Рисунок 5

6.3.3.7 Диаграмма развертывания дополняется текстом в базе знаний проекта:

– для устройств может быть дана спецификация аппаратного обеспечения;

– спецификация развертывания может быть детализирована в базе знаний.

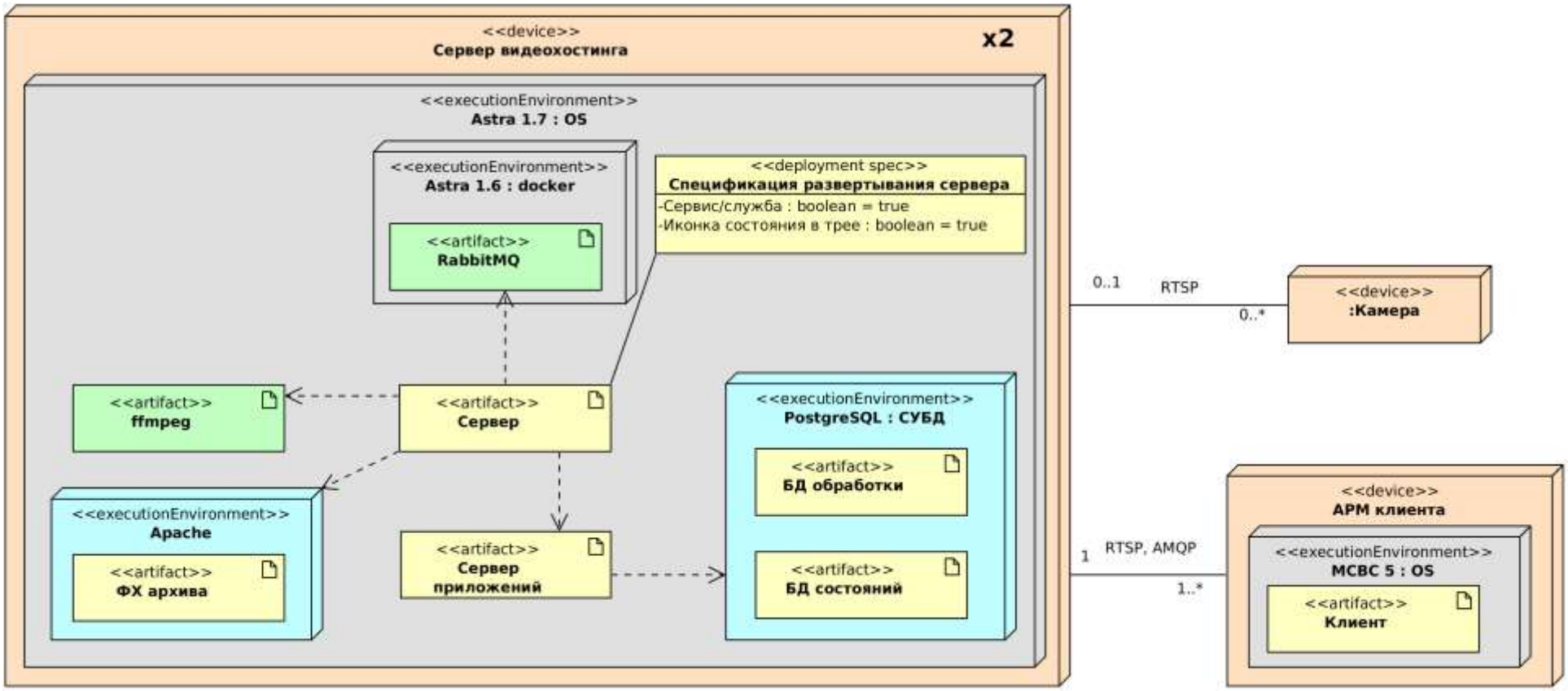
<<device>>
Сервер видеохостинга
x2
<<executionEnvironment>>
Astra 1.7 : OS
<<executionEnvironment>>
Astra 1.6 : docker
<<artifact>>
RabbitMQ
<<deployment spec>>
Спецификация развертывания сервера
-Сервис/служба : boolean = true
-Иконка состояния в трее : boolean = true
<<artifact>>
ffmpeg
<<artifact>>
Сервер
<<executionEnvironment>>
PostgreSQL : СУБД
<<artifact>>
БД обработки
<<executionEnvironment>>
Apache
<<artifact>>
ФХ архива
<<artifact>>
Сервер
приложений
<<artifact>>
БД состояний
0..1
RTSP
0..*
<<device>>
:Камера
1
RTSP, AMQP
1..*
<<device>>
АРМ клиента
<<executionEnvironment>>
МСВС 5 : OS
<<artifact>>
Клиент

Рисунок 6

6.3.4 Протоколы информационного взаимодействия

6.3.4.1 Протокол информационного взаимодействия (далее – протокол) описывает порядок взаимодействия между различными системами, подсистемами одной системы и т. п. Типовое содержимое разделов протокола приведено в таблице 1.

Т а б л и ц а 1

Раздел	Подраздел	Содержимое	Примечание
Введение	Назначение	Цели разработки протокола. Например, для управления системой и получения различной информации (указывается, какой)	–
	Область применения	Взаимодействующие стороны (системы, подсистемы и т. п.)	–
	Основания для разработки	Руководящий документ (например, техническое задание), на основании которого разрабатывается протокол	–
	Срок ввода в действие и время действия	Сроком ввода в действие протокола обычно считается дата его утверждения. Время действия – период проведения испытаний и период эксплуатации системы	–
	Порядок корректировки	Порядок внесения изменений, например, протокол может уточняться и дополняться по согласованию сторон путем выпуска дополнений или иной порядок	–
Порядок взаимодействия	–	Сценарии взаимодействия: инициатор взаимодействия, порядок обмена сообщениями и т. п. При описании сценария взаимодействия излагается цель взаимодействия и кратко описывается последовательность взаимодействия. Каждый сценарий должен содержать: описание, диаграмму взаимодействия, при необходимости описание диаграммы для подробного пояснения отдельных элементов. На диаграмме обязательно показываются все элементы внешнего взаимодействия своей системы. Взаимодействия между элементами других систем и элементы внутреннего взаимодействия своей системы показываются, когда это	Диаграмма взаимодействия выполняется в нотации UML Sequence diagram

Раздел	Подраздел	Содержимое	Примечание
		нужно для целостного понимания всей схемы взаимодействия	
Технология информационного обмена	–	Тип взаимодействия (сетевое взаимодействие / обмен файлами). Описание реализации стандарта сетевой модели OSI (7 уровней) или сетевой модели TCP/IP (4 уровня). Принципы и особенности взаимодействия, такие как наличие брокера, тип паттерна взаимодействия и т. п. Методы аутентификации / авторизации, уровни конфиденциальности, шифрование SLA: объем передаваемой информации, количество запросов в секунду, допустимая задержка ответа	Раздел при необходимости может быть разбит на подразделы в зависимости от технологий информационного обмена, этапов взаимодействия, элементов взаимодействующих систем (подсистем) и т. п. В этом случае описание реализации стандарта сетевой модели и описание информационного взаимодействия выполняется отдельно для каждого подраздела
Приложения	Состав и описание форматов данных	Исходные тексты. Например, proto-файлы, json-схемы, xsd-схемы. При необходимости, таблицы описания сообщений информационного обмена	Могут быть добавлены иные вспомогательные данные по согласованию взаимодействующих организаций. Таблицы описания сообщений информационного обмена обычно добавляются для удобства анализа протокола сотрудниками согласовывающих организаций, не владеющими знаниями скриптовых языков описания исходных текстов сообщений. Кроме того, первоначальное табличное описание сообщений информационного обмена позволяет учесть все нюансы при формировании артефактов верификации сообщений (json-схем, xsd-схем, элементов proto-файлов)
	Примеры сообщений	Например, json-файлы, xml-файлы	–

6.3.4.2 Для описания раздела технологий информационного обмена существуют следующие рекомендации:

– в большинстве случаев подходит модель описаний сети TCP/IP с 4 уровнями деления. Модель OSI с 7 уровнями следует использовать для сложных случаев сетевого взаимодействия;

– при выделении сетевых портов следует их добавить в таблицу используемых портов, находящуюся в базе знаний АО «НИЦ СПб ЭТУ»;

– при выборе портов следует избегать ephemeral ports с целью минимизации рисков конфликта со средствами антивирусной защиты приложений;

– описывать имеет смысл уровни, действительно влияющие на прикладное ПО или реализацию протокола взаимодействия. Например, для протокола взаимодействия программного уровня достаточно указать транспортный или прикладной протокол, а при описании взаимодействия между системами – целесообразно согласовывать и физический уровень взаимодействия;

– при описании технологий защиты информации следует (в зависимости от наличия подобных ограничений) уделить внимание:

1) согласованию уровней конфиденциальности как при передаче файлов через файловую систему, так и данных по сети, в том числе между разными ОС, которые по-разному могут интерпретировать одни и те же метки конфиденциальности;

2) требованиям к аутентификации/авторизации, исходящим из класса системы и методов обеспечения единого пространства пользователя.

6.3.4.3 Пример протокола информационного взаимодействия «Протокол REST/JSON» приведен в приложении Б.

Пример протокола информационного взаимодействия «Протокол TCP/Protobuf» приведен в приложении В.

6.4 Логическое представление системы

6.4.1 Логическое представление системы имеет целью продемонстрировать набор компонентов и их взаимосвязей, ответственных за закрытие всех функциональных и нефункциональных требований системы. Описание системы должно быть представлено как список компонентов верхнего уровня. Компонент представляет собой самостоятельный модуль со своим контрактом и реализует:

– удовлетворение отдельных функциональных требований;

– отдельные аспекты функционирования системы: представление данных, доступ к данным (ввод/вывод/конвертирование), транспорт/взаимодействие, GUI, математика, логгирование, конфигурирование и т. д.

От компонентов верхнего уровня должна идти детализация структуры с описанием включаемых компонентов до уровня, определяемого стадией разработки архитектуры. Логическое представление дает общее видение о системе и подробно описывает как:

– организованы компоненты;

– компоненты взаимодействуют;

– компоненты зависят друг от друга;

– библиотечные и внешние зависимости используются для достижения целей системы.

Компонент представляет отдельную часть системы, которая выполняет определенную функцию или имеет определенную роль. Он может быть программным модулем, классом, библиотекой, сервисом и т. д. Компоненты могут быть заменены альтернативой с соблюдением программных контрактов взаимодействия.

6.4.2 Логическое представление системы осуществляется при помощи диаграммы компонентов. Основные элементы диаграммы компонентов описаны в разделе логического представления системы и не меняют своей семантики. В части диаграммы компонентов существенно:

– описание портов, которые определяют «внешнее» (по отношению к программному модулю) взаимодействие компонентов – с компонентами других программных модулей системы или с другими системами. Наличие портов на диаграмме предполагает существование протокола взаимодействия – стандартного (ftp, WebDAV) или специализированного;

– интерфейс – как абстракция набора возможностей/функций компонента в конечном итоге будет являться классом-интерфейсом в терминах конкретного стека (или диаграммы классов);

– описание использования общих библиотечных решений (сторонних или внутренних) с использованием соответствующих стереотипов.

Пример диаграммы компонентов представлен на рисунке 7.

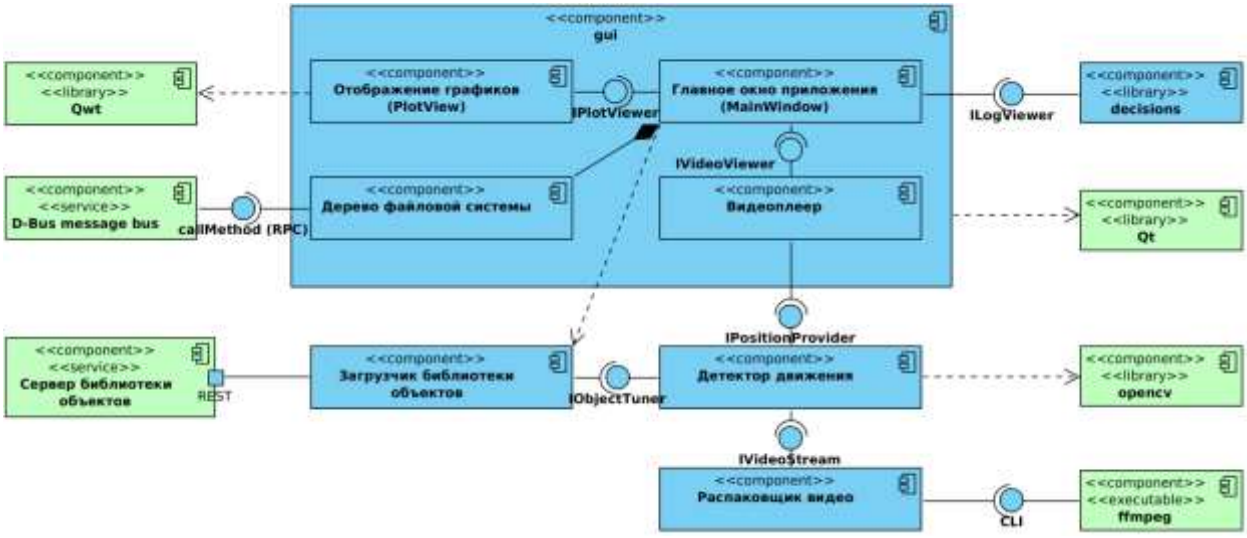

Рисунок 7

6.4.3 Для наглядности и удобочитаемости рекомендуется оформлять диаграммы компонентов следующим образом:

– задавать различные цвета:

1) для компонентов АО «НИЦ СПб ЭТУ» и компонентов сторонних взаимодействующих организаций;

2) для логично выделяемых групп элементов, если это приведёт к более лёгкому восприятию диаграммы (например, компоненты, отвечающие за безопасность);

– полноценно использовать механизм стереотипов для детализации вида компонента:

1) «service»;

2) «library»;

3) «executable» и др.;

– задействовать при необходимости порты с целью отразить сетевое взаимодействие с внешними компонентами;

– помнить о возможности наличия у компонента различных интерфейсов и, если это важно, отражать их использование иными компонентами.

6.4.4 Диаграмма классов является нотацией более низкого уровня и применяется для логического представления обычно частей системы, требующих особого фокуса при проектировании. Стоит отметить, что диаграмма классов представляет ценность именно на этапе согласования нетривиальных архитектурных решений, так как после написания кода может быть сгенерирована автоматизированными средствами документирования. Такими фокусами могут быть:

– структуры данных (состав атрибутов, вложенность, множественность, владение) для ключевых «управляющих» структур;

– распределение ответственности сущностей (состав методов), взаимосвязи между сущностями (интерфейсы, наследование, владение);

– применение типовых паттернов проектирования или других архитектурных конструкций для реализации функционального или нефункционального требования.

6.4.5 Диаграмма классов – это UML-диаграмма, которая описывает систему, визуализируя различные типы объектов внутри системы и виды статических связей, которые существуют между ними. Она также иллюстрирует операции и атрибуты классов. Динамика должна быть отражена на диаграммах других видов (диаграммах последовательности, состояний, коммуникации и т. д.).

6.4.6 Элементы диаграммы классов:

– класс – ключевой элемент (описание класса содержит имя и члены класса (поля/атрибуты и методы) с заданием видимости);

– взаимосвязи объектов классов – зависимость, ассоциация (агрегация, композиция);

– взаимосвязи классов – обобщение (наследование), реализация;

– зависимость – слабая форма отношения использования, при которой изменение в спецификации одного влечёт за собой изменение другого (обратное не обязательно).

6.4.7 Графическое описание логического представления дополняется текстом в базе знаний проекта. Описание компонента может состоять из следующих разделов:

– описание компонента и его ответственности;

– особенности использования программного интерфейса какого-либо компонента или самого компонента;

– ссылки на протоколы взаимодействия с внешними сущностями;

– ограничения на версии внешних зависимостей;

– обоснование принятых архитектурных решений.

6.5 Исследование технологий

6.5.1 В процессе проектирования архитектуры ПО возникает необходимость выбора технологических решений в условиях наличия нескольких альтернатив. Систематизация принципов и порядка проведения исследования технологий преследует следующие цели:

– осуществление осознанного выбора на основе объективных критериев;

– обоснованное повторное использование результатов исследований между проектами;

– повышение качества инфраструктурной поддержки единых решений за счёт фокусировки на ограниченном числе технологий.

При выборе технологии важно обеспечить надежность, достоверность и воспроизводимость результатов исследований. Проведение исследования технологий и инструментов требует системного подхода, чтобы исследование являлось всесторонним и целостным.

6.5.2 Порядок проведения исследования технологий включает в себя следующие шаги:

– определение цели – чаще всего необходимость выбора технологии вызвана конкретной проектной задачей, вытекающей из требований к проекту, например, «Выбор сетевого протокола обмена данными …» или «Выбор технологии автоматизированного тестирования приложений на Swing». Постановка цели исследования предполагает понимание конечного результата, нужного проекту от выбора технологии, и чаще всего переносится напрямую в формулировку задачи в системе управления проектами;

– формирование набора критериев – определить и согласовать с заинтересованными лицами (например, системный архитектор проекта) критерии оценки технологий;

– определение веса критерия – в сложных случаях может понадобиться определить не только сами критерии, но и их важность для проекта в виде целого числа, отражающего приоритетность критерия:

1) сбор данных – сравнить технологии по выбранным критериям;

2) формирование набора технологий-кандидатов;

3) анализ технологий по выбранным критериям – может осуществляться как на основе документации, так и через прототипирование в условиях отсутствия нужных данных в открытом доступе;

4) анализ отзывов – проанализировать отзывы и мнения других пользователей технологий, чтобы получить представление об их преимуществах и недостатках в решаемых задачах;

5) оценка рисков – обозначить риски применения в различных условиях;

– проверка технической реализуемости – в случаях, когда нет уверенности в информации из анализируемых источников или есть подозрения в её неприменимости в конкретных условиях проекта, может понадобиться подтверждение достижения цели исследования с применением конкретной технологии, например, посредством прототипирования;

– подведение итога – сформировать вывод на основе результатов исследования. Результаты исследования могут быть использованы в нескольких проектах, в том числе возможна ситуация выбора различных технологий на основе результатов одного и того же исследования в условиях различных требований проекта, поэтому формулировки вывода могут содержать условия при которых рекомендуется использовать ту или иную технологию. Также требуется определить наиболее подходящую для проекта, в рамках которого проведено исследование, технологию, аргументировать выбор. Привести рекомендации по ее применению и по ограничениям использования;

– утверждение результатов – результат исследования доводится/обсуждается системным архитектором проекта АО «НИЦ СПб ЭТУ» с целью повышения качества и всесторонности с одной стороны, а также применения единых решений с другой. Утверждение и доведение происходит через канал в системе обмена мгновенными

сообщениями (в соответствии с документом «Положение о дистанционной (удаленной) работе АО «НИЦ СПб ЭТУ»») и комментариях к странице исследования.

6.5.3 Обилие и разноплановость инженерных задач не позволяет сформировать полный или универсальный набор критериев оценки. В то же время, можно выделить рекомендации и примеры критериев различных классов, на которые в значительной части случаев можно обратить внимание.

Рекомендации:

– при возможности, формулировать критерий как бинарный (по которому можно оценить технологию в терминах да/нет). Например, «Поддержка мандатных меток»;

– минимизировать отмеченные особенности технологий в тексте или примечаниях, если для них можно сформулировать критерий. Например, риск отсутствия обновлений библиотеки можно сформулировать в виде критерия «Дата последней версии» или «Поддерживаемость»;

– набор выбранных критериев рекомендуется заранее согласовать с заинтересованными лицами, так как добавление даже одного нового критерия после первичного исследования, приведёт к ещё одной итерации исследования всех технологий-кандидатов.

Примеры критериев:

– лицензия распространения;

– открытый исходный код;

– стоимость;

– масштабируемость;

– поддерживаемость;

– производительность;

– качество (например, распознавания текста);

– качество документации;

– простота реализации/использования;

– поддерживаемые ОС;

– поддерживаемые версии зависимостей (python, jdk, clang, gcc и т. п.);

– дата последнего выпуска версии.

Примеры критериев безопасности:

– уязвимость (наличие критических уязвимостей, не устраняемых долгое время);

– поддержка мандатного/дискреционного разграничения;

– наличие сертификата соответствия (Федеральная служба по техническому и экспортному контролю и др.);

– шифрование.

6.5.4 Принципы проведения исследования включают в себя следующие элементы:

– воспроизводимость – например, при написании тестов или иного исходного кода при исследовании технологии, он фиксируется в выделенном репозитории исходных кодов в специализированной группе проектов research, предназначенной для хранения артефактов исследований с целью возможности воспроизведения и расширения на новые технологии-кандидаты при актуализации исследования в будущем. При расчете количественных данных, полученных практическим путем, необходимо указать на каких данных проводилось исследование;

– документирование – исследование описывается в ясной и краткой форме, используя визуальные средства (таблицы, диаграммы сравнения);

– нейтральность – придерживаться нейтральности и объективности при оценке технологий, чтобы избежать предвзятости и ошибок;

– тщательность – быть тщательным и внимательным при сборе и анализе данных о технологиях, чтобы обеспечить точность и надежность результатов;

– учет требований проекта – учитывать требования проекта при окончательном технологическом выборе.

6.5.5 При оформлении результатов исследования следует руководствоваться следующей описанной структурой.

6.5.5.1 Цель исследования.

Без дополнительного вложенного заголовка в начале страницы исследования:

– кратко описать цель проводимого исследования;

– при необходимости дать краткое описание терминов/сокращений используемых далее;

– указать ссылку на задачу в системе управления проектами для возможности дополнительного анализа причин исследования, просмотра исполнителей, дополнительных комментариев.

Пример:

Необходимо исследовать инструменты оптического распознавания символов для использования в проектах АО «НИЦ СПб ЭТУ» в рамках задачи. Исследования проводились на ОС СН «Astra Linux Special Edition», релиз «Смоленск» (версия 1.7) для документов на русском языке.

6.5.5.2 Сводная таблица результатов.

Заинтересованные в исследовании лица просматривают страницу сверху, поэтому расположение итоговой таблицы целесообразно именно до описания подробностей исследуемых технологий – это основной и наиболее представительный результат сравнительного анализа. Цель сведения данных в таблицу – обеспечить оперативный анализ результатов исследования.

Таблица должна содержать:

– перечень инструментов/технологий, подвергаемых исследованию;

– перечень критериев оценки;

– на пересечении критерия и технологии оценку соответствия;

– ссылку на страницу технологии, при наличии.

Рекомендации к таблице:

– критерии по умолчанию лучше располагать в столбцах, а технологии – в строках, так как в общем случае технологий может быть больше, чем критериев;

– оценку бинарных критериев рекомендуется отмечать знаками «+» или «-», а не только цветовым выделением ячейки с целью удобного экспортирования статьи в .odt /.pdf форматы;

– целесообразно добавить столбец примечаний для дополнительной важной краткой информации/особенности технологии;

– при наличии в статье более полного описания и оценки технологии рекомендуется делать внутреннюю ссылку с названия технологии в таблице для быстрого перехода к описанию, а уже официальный сайт размещать в подробном описании технологии;

– желательно добавить цветовое разграничение для быстрого визуального поиска наилучшего результата.

Пример оформления сводных таблиц результатов представлен в таблицах 2 и 3.

Т а б л и ц а 2

Библиотека	Движок	Открытый исходный код	Дата последнего релиза	Зависимости	Качество распознавания
pytesseract	Tesseract	+	15.10.2023	Python >= 3.6, Pillow,libtesseract, tesseract-ocr, pytesseract	0.896
pyocr	Tesseract или CuneiForm	+	17.09.2023	Python >= 3.4, Pillow, libtesseract или tesseract-ocr или Cuneiform, pyocr	0.955
dedoc	Tesseract	+	20.11.2024	Python >= 3.7, libreoffice, djvulibre-bin, unzip, unrar, libtesseract-dev, dedoc	0.959
EasyOSR	EasyOSR	+	24.09.2024	Python >= 3.7, torch, torchvision>=0.5, Pillow, scikit-image, easyocr	0.783
ABBYY FineReader	ABBYY FineReader Engine	–	–		
SETERE OCR		–	01.11.2024	libqt5webengine5, libqt5webenginewidgets5, libpodofo0.9.6	0.879
PaddleOCR	Paddle Paddle	+	22.10.2024	Python >= 3.7, Pillow, requests, opencv-python, scikit-image, paddle-ocr	0.349

Т а б л и ц а 3

Критерий	Samba	NFS	FTP	SFTP	WebDav
Поддержка мандатной модели в ОС СН «Astra Linux»	+	–	–	–	+
Поддержка мандатной модели в ОС МСВС	+	+	–	–	+
Абстракция сетевой ФС	+	+	–	Частично через sshfs	Частично через davfs

Окончание таблицы 3

Критерий	Samba	NFS	FTP	SFTP	WebDav
Количество сетевых соединений с клиента	Число пользователей	1	Число соединений	Число соединений	Число соединений
Аутентификация	+	Доверие UID клиента	+	+	+
Шифрование передаваемых данных	Опционально	–	–	+	При работе с SSL
Докачивание файлов	Через rsync	Через rsync	+	+	+
Простота отслеживания ошибок соединения			+		+
Корректная обработка потери соединения с сервером	Подвисание/ отключение[1)]	Подвисание	+	+	+
Наличие сервера в ОС СН «Astra Linux»	+	+	+	+	+
Наличие сервера в ОС МСВС	Файлы <= 2 Гбайт	+	+	+	
Поддержка ACL	+	+	+	+	–
[1)]Может быть решено за счёт применения autofs.					

6.5.5.3 Дополнительная информация.

Для более полного описания исследуемых аспектов технологий/инструментов может возникнуть необходимость в дополнительной информации в менее формализованном виде, нежели сравнительная таблица. В этом случае выделяется под отдельным заголовком информация по соответствующей технологии в произвольном виде. Дублировать данные, отражённые в таблице, при дополнительном описании не требуется. Дополнительные данные могут потребоваться для:

– обоснования принятия решения;

– краткого описания исследуемых инструментов, их принципов работы;

– приведения примеров использования в виде небольших фрагментов кода или ссылок на него;

– описания исследования инструментов на реальном проекте АО «НИЦ СПб ЭТУ» или прототипе;

– описания ограничений или возникших трудностей при проведении исследования.

6.5.5.4 Вывод.

В завершении исследования необходимо:

– подвести итог проведенного исследования, указать наилучшее решение по исследуемым критериям;

– дать рекомендации по использованию или дальнейшему изучению.

Пример 1:

В результате исследования проведен сравнительный анализ средств, позволяющих проводить статический анализ избыточности Java-кода, в частности, позволяющих находить неиспользуемые классы. Поскольку на рынке не найдены средства, обеспечивающие из коробки полный анализ избыточности на уровне проекта, то приведенные средства нуждаются в дополнительной конфигурации и/или написании кода. По результатам проверки средств на реальном проекте, лучший результат показало средство jQAssistant, также стоит отметить, что средство jQAssistant работает значительно быстрее, чем прототип и позволило выявить вложенные неиспользуемые классы.

Пример 2:

Для тестирования Swing предпочтительно использовать AssertJ или Sikuli в зависимости от стабильности интерфейса пользователя и необходимости сравнения сложных графических компонентов.

7 Управление документом

Управление настоящей документированной процедурой осуществляется в соответствии с документом СМК 4.2.3-01ПР.

Ответственным за актуализацию настоящей документированной процедуры является начальник департамента программного обеспечения.

Приложение А
(справочное)
Схема процесса проектирования

А.1 Схема процесса проектирования приведена на рисунке А.1.

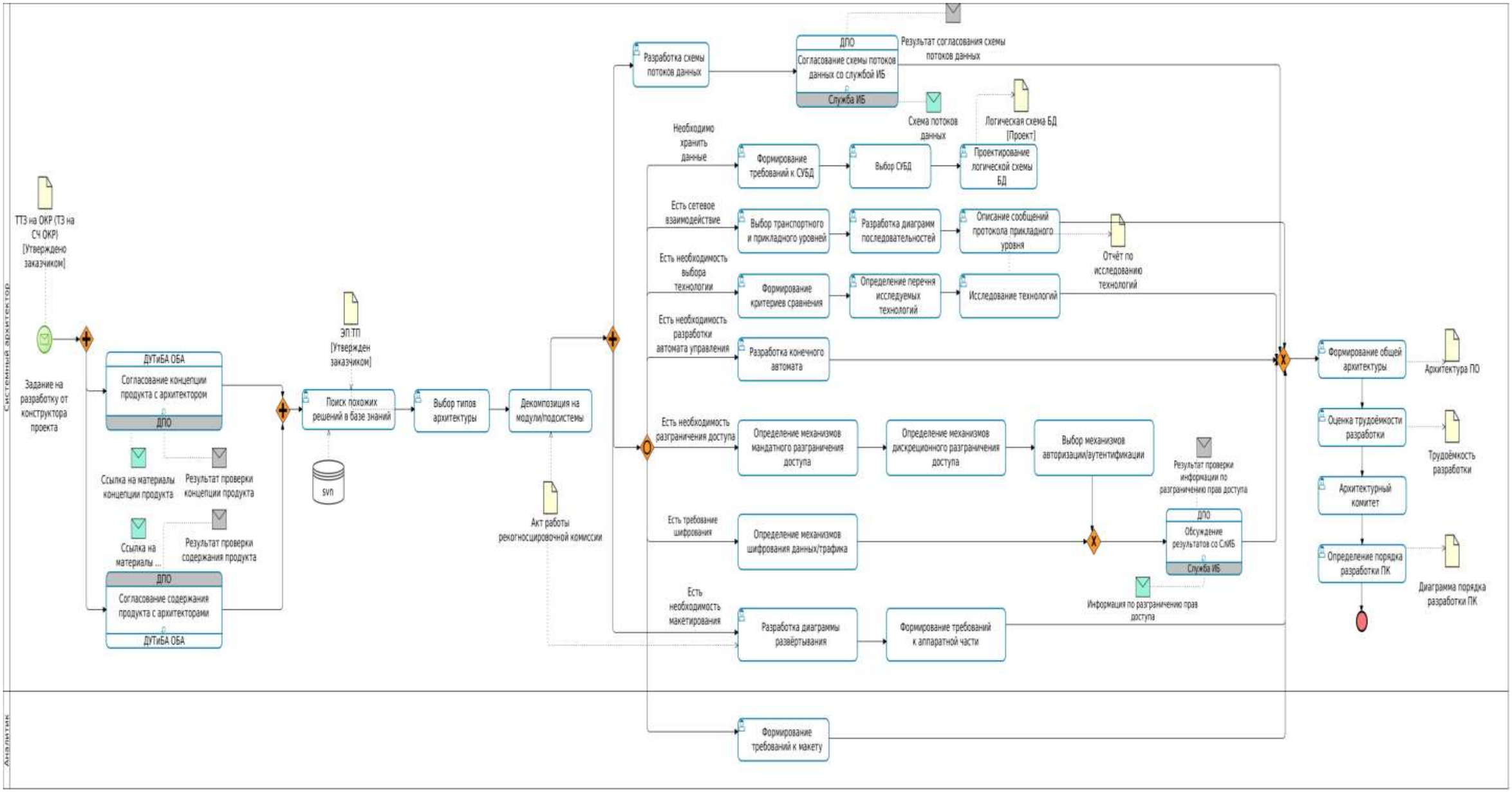

Рисунок А.1

Приложение Б

(справочное)

Пример описания протокола «Протокол REST/JSON»

Порядок взаимодействия.

Перечень сценариев взаимодействия:

– взаимодействие при формировании и доведении заданий по основному назначению для радиотехнических комплексов;

– взаимодействие при уточнении и доведении заданий по основному назначению для радиотехнических комплексов;

– взаимодействие при получении и обработке зарегистрированной информации радиотехнических комплексов;

– взаимодействие при формировании и доведении запросов на проведение сеансов наблюдения оптико-электронных средств и получения результатов планирования сеансов наблюдения;

– взаимодействие при получении и обработке зарегистрированной информации оптико-электронного комплекса.

Планирование сеансов наблюдения.

Взаимодействие АПК ОДМ осуществляется в целях формирования, доведения и согласования запросов на проведение сеансов наблюдения оптических средств и получения от органов управления АНЦ результатов планирования (ответов на запросы) оптических средств:

– запрос подготавливается на АПК ОДМ-У и передается представителям АНЦ на НЗ для предварительного анализа и согласования;

– после согласования запроса этот запрос на НЗ предоставляется в автоматизированный пункт управления для его последующей передачи после регистрации в журнале входящих документов на О-часть в МУ РС;

– после передачи запроса в МУ РС (после автоматического получения уведомления) АПК ОДМ-О организует копирование документа в свое ФХ для последующей обработки программными комплексами О-части;

– результаты планирования (ответы на запросы) выкладываются в МУ РС способами внутреннего протокола АНЦ и параллельно доводятся до АПК ОДМ-У при помощи НЗ. После автоматического получения уведомления о наличии ответа на запрос АПК АДМ-О организует копирование документа в свое ФХ для последующей обработки программными комплексами О-части;

– сценарий считается выполненным, когда в ФХ на АПК ОДМ-О будут сохранены запрос на проведение сеансов наблюдения оптических средств и соответствующий ему ответ с результатами планирования;

– порядок обмена сообщениями представлен на диаграмме (взаимодействующие части нашей системы показаны светло-зеленым цветом).

Схема взаимодействия представлена на рисунке 1.

Детализация действий:

– подготовка/уточнение запроса на проведение СН – оператор АПК ОДМ-У готовит файл запроса request.json;

– запись файла запроса (request.json) на НЗ – запись файла на НЗ (CD-ROM) производится средствами АПК ОДМ-У;

– запрос на проведение СН (request.json) на НЗ – файл на НЗ передается должностному лицу АНЦ:

1) результат анализа корректности запроса (согласование) – вербальный ответ от представителя АНЦ;

2) подготовка НЗ с файлом запроса – запись файла на НЗ (CD-ROM) производится средствами АПК ОДМ-У;

3) предоставление файла запроса на НЗ для последующей передачи в МУ РС – файл на НЗ передается должностному лицу АНЦ с обязательной фиксацией факта передачи в журнале входящих документов;

– информация о появлении нового запроса – информация появляется в результате автоматически выполняемых периодических запросов ПК ИВ по протоколу ftp содержимого каталогов МУ РС:

1) формирование задания копирования файла запроса – задание копирования создается оператором АПК ОДМ-О с использованием ПК ИВ;

2) запрос файла request.json;

3) копирование файла запроса request.json в ФХ – копирование файла в ходе выполняемых заданий копирования ПК ИВ по протоколу ftp из каталога МУ РС в ФХ АПК ОДМ-О;

– информация о появлении нового ответа – информация появляется в результате автоматически выполняемых периодических запросов ПК ИВ по протоколу ftp содержимого каталогов МУ РС;

– формирование задания копирования файла ответа – задание копирования создается оператором АПК ОДМ-О с использованием ПК ИВ:

1) запрос файла response.json;

2) копирование файла ответа response.json в ФХ – копирование файла в ходе выполняемых заданий копирования ПК ИВ по протоколу ftp из каталога МУ РС в ФХ АПК ОДМ-О;

– файл ответа (response.json) на НЗ – от должностного лица АНЦ на НЗ получается файл response.json с результатами планирования применения оптических средств в соответствии с нашим запросом;

– сохранение файла ответа в ФХ – выполнение задания копирования ПК ИВ с НЗ в ФХ.

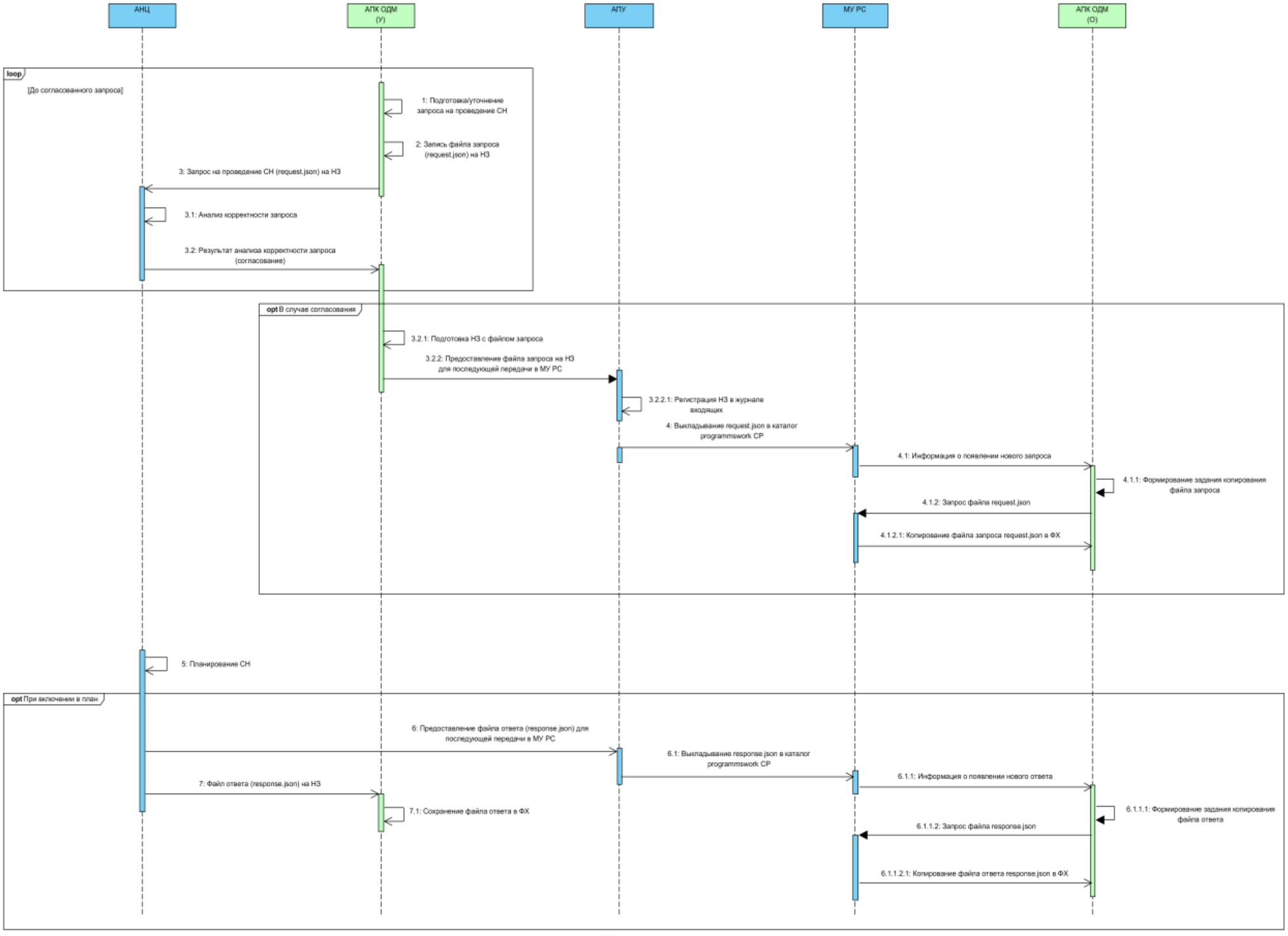
АНЦ
АПК ОДМ (У)
АПУ
МУ РС
АПК ОДМ (О)
loop
[До согласованного запроса]
1: Подготовка/уточнение запроса на проведение СН
2: Запись файла запроса (request.json) на НЗ
3: Запрос на проведение СН (request.json) на НЗ
3.1: Анализ корректности запроса
3.2: Результат анализа корректности запроса (согласование)
opt В случае согласования
3.2.1: Подготовка НЗ с файлом запроса
3.2.2: Предоставление файла запроса на НЗ для последующей передачи в МУ РС
3.2.2.1: Регистрация НЗ в журнале входящих
4: Выкладывание request.json в каталог programmswork СР
4.1: Информация о появлении нового запроса
4.1.1: Формирование задания копирования файла запроса
4.1.2: Запрос файла request.json
4.1.2.1: Копирование файла запроса request.json в ФХ
5: Планирование СН
opt При включении в план
6: Предоставление файла ответа (response.json) для последующей передачи в МУ РС
6.1: Выкладывание response.json в каталог programmswork СР
6.1.1: Информация о появлении нового ответа
6.1.1.1: Формирование задания копирования файла ответа
6.1.1.2: Запрос файла response.json
6.1.1.2.1: Копирование файла ответа response.json в ФХ
7: Файл ответа (response.json) на НЗ
7.1: Сохранение файла ответа в ФХ

Рисунок 1

Технология информационного обмена.

В соответствии со стандартом TCP/IP протокол взаимодействия предполагает следующие параметры различных уровней:

– канальный уровень:

1) физический канал передачи данных: медный кабель витая пара категории CAT5e или выше. Электрические характеристики интерфейса соответствуют стандарту Ethernet 1000BASE-T, IEEE 802.3ab. Скорость передачи данных от 10 Мбит/с;

2) порядок подключения: кабель подключается в порт маршрутизатора, настроенный на маршрутизацию трафика в соответствии с параметрами сетевого адресного пространства объекта установки;

– транспортный уровень: протокол взаимодействия: TCP;

– прикладной уровень: протокол взаимодействия: HTTP.

Состав и описание форматов данных представлен в виде json-схемы сообщения request_scheme.json:

```
{
   "$schema": "http://json-schema.org/draft-07/schema#",
   "type": "object",
   "title": "Запрос на проведение сеанса наблюдения оптико-электронного комплекса",
   "properties": {
      "metadata" : {
         "type": "object",
         "title": "Метаданные запроса",
         "properties": {
            "message_id": {
               "title": "Уникальный идентификатор сообщения",
               "$ref": "#/$defs/uuid"
            },
            "message_type": {
               "type": "string",
               "title": "Тип сообщения",
               "enum": [ "REQUEST", "REQUEST_CANCEL" ],
               "comment": "REQUEST - Запрос на проведение СН; REQUEST_CANCEL
- Отмена запроса"
            },
            "originator": {
               "type": "string",
               "title": "Источник формирования",
               "enum": [ "SKPPAD", "APK_ODM" ]
            },
            "creation_date": {
               "title": "Время создания/изменения запроса (UTC)",
               "$ref": "#/$defs/timestamp"
            }
         },
         "required": [ "message_id", "message_type", "originator", "creation_date" ]
      },
      "data" : {
         "type": "object",
```

```
"title": "Данные запроса",
"properties": {
  "request_id": {
    "title": "Уникальный идентификатор запроса",
    "$ref": "#/$defs/uuid"
  },
  "object_id": {
    "type": "string",
    "title": "Номер объекта наблюдения",
    "pattern": "^(?:[0-9]{5})$"
  },
  "start_time": {
    "title": "Дата и время начала сеанса наблюдения (UTC)",
    "$ref": "#/$defs/timestamp"
  },
  "end_time": {
    "title": "Дата и время окончания сеанса наблюдения (UTC)",
    "$ref": "#/$defs/timestamp"
  },
  "priority": {
    "type": "integer",
    "title": "Приоритет",
    "enum": [ 0, 1 ],
    "comment": "0 – космический аппарат, 1 - объект по основному назначению"
  },
  "track": {
    "type": "array",
    "title": "Типовая траектория движения объекта наблюдения в ГрСК",
    "minItems": 1,
    "uniqueItems": true,
    "items": {
      "type": "object",
      "title": "Точка типовой траектории",
      "properties": {
        "time": {
          "type": "integer",
          "title": "Время относительное, с",
          "minimum": 0
        },
        "X": {
          "type": "number",
          "title": "Координата X"
        },
        "Y": {
          "type": "number",
          "title": "Координата Y"
        },
```

```
                        "Z": {
                            "type": "number",
                            "title": "Координата Z"
                        },
                        "X_DOT": {
                            "type": "number",
                            "title": "Первая производная по координате X"
                        },
                        "Y_DOT": {
                            "type": "number",
                            "title": "Первая производная по координате Y"
                        },
                        "Z_DOT": {
                            "type": "number",
                            "title": "Первая производная по координате Z"
                        }
                    },
                    "required": [ "time", "X", "Y", "Z", "X_DOT", "Y_DOT", "Z_DOT" ]
                }
            },
            "targeting_site_id": {
                "type": "integer",
                "title": "Идентификатор средства регистрации, для которого рассчитаны целеуказания",
                "minimum": 1
            },
            "targets": {
                "type": "array",
                "title": "Массив целеуказаний по объекту наблюдения",
                "minItems": 1,
                "uniqueItems": true,
                "items": {
                    "type": "object",
                    "title": "Точка целеуказания",
                    "properties": {
                        "time": {
                            "type": "integer",
                            "title": "Время относительное, с",
                            "minimum": 0
                        },
                        "azimuth": {
                            "type": "number",
                            "title": "Азимут, градусы"
                        },
                        "elevation": {
                            "type": "number",
                            "title": "Угол места, градусы"
                        }
```

```
                    },
                    "required": [ "time", "azimuth", "elevation" ]
                  }
                }
              },
              "required": [ "request_id", "object_id", "start_time", "end_time", "priority" ],
              "if": {
                "required": [ "targets" ]
              },
              "then": {
                "required": [ "targeting_site_id" ]
              },
              "else": {
                "not": { "required": ["targeting_site_id"] }
              }
            }
          },
          "required": [ "metadata", "data" ]
          "$defs": {
            "timestamp": {
              "type": "string",
              "pattern":                                                    "^(\\d{4})-(\\d{2})-
(\\d{2})([T](\\d{2}):(\\d{2})(:(\\d{2}(?:\\.\\d+)?))?)?Z$"
            },
            "uuid": {
              "type": "string",
              "pattern":    "(?i)^(?:[0-9a-f]{8}-[0-9a-f]{4}-[1-7][0-9a-f]{3}-[89ab][0-9a-f]{3}-[0-
9a-f]{12})$"
            }
          }
        }
```

Формат структуры сообщения заявки на проведение сеанса наблюдения представлен в таблице 1.

Объект типа Metadata – метаданные сообщения представлены в таблице 2.

Т а б л и ц а 1

Имя поля	Тип поля	Тип данных	Описание поля	Пример заполнения	Признак обязательности	Дополнительные свойства	Возможные значения	Примечания
metadata	object	Объект типа Metadata	Метаданные сообщения	–	+	–	–	–
data	object	Объект типа Data	Данные сообщения	–	+	–	–	–

Т а б л и ц а 2

Имя поля	Тип поля	Тип данных	Описание поля	Пример заполнения	Признак обязательности	Дополнительные свойства	Возможные значения	Примечания
message_id	string	string	Уникальный идентификатор сообщения (UUID)	«15C57ABA-7DA1-43CC-947A-3334BA0A710A»	+	–	–	Для каждого нового сообщения генерируется новый message_id
message_type	string	string	Тип сообщения	«REQUEST»	+	–	«REQUEST», «REQUEST_CANCEL»	Запрос на проведение СН. Отмена запроса
originator	string	string	Источник формирования	«SKPPAD»	+	–	«SKPPAD», «APK_ODM»	–
creation_date	string	datetime	Время создания/изменения запроса (UTC)	«2022-10-23T22:05:00Z»	+	–	–	–

Пример сообщения request.json:


```
{
   "metadata" : {
      "message_id": "018e41ec-5e62-767d-a5b1-85974cc81c58",
      "message_type": "REQUEST",
      "originator": "APK_ODM",
      "creation_date": "2022-10-23T22:06:17Z"
   },
   "data" : {
      "request_id": "018cfa24-32ab-727b-9d1a-61a7badd1a35",
      "object_id": "29522",
      "start_time": "2022-10-24T10:05:00Z",
      "end_time": "2022-10-24T10:20:00Z",
      "priority": 1,
      "track": [
         {
            "time": 1,
            "X": -3272962,
            "Y": 5584462,
            "Z": 3213615,
            "X_DOT": 3147.5,
            "Y_DOT": -1945.5,
            "Z_DOT": 6576.5
         }
      ],
      "targeting_site_id": 13288,
      "targets": [
         {
            "time": 1,
            "azimuth": 2.82362910523035,
            "elevation": 0.179180535411965
         }
      ]
   }
}
```

Приложение В
(справочное)
Пример описания протокола «Протокол TCP/Protobuf»

Порядок взаимодействия представлен на рисунке 1.

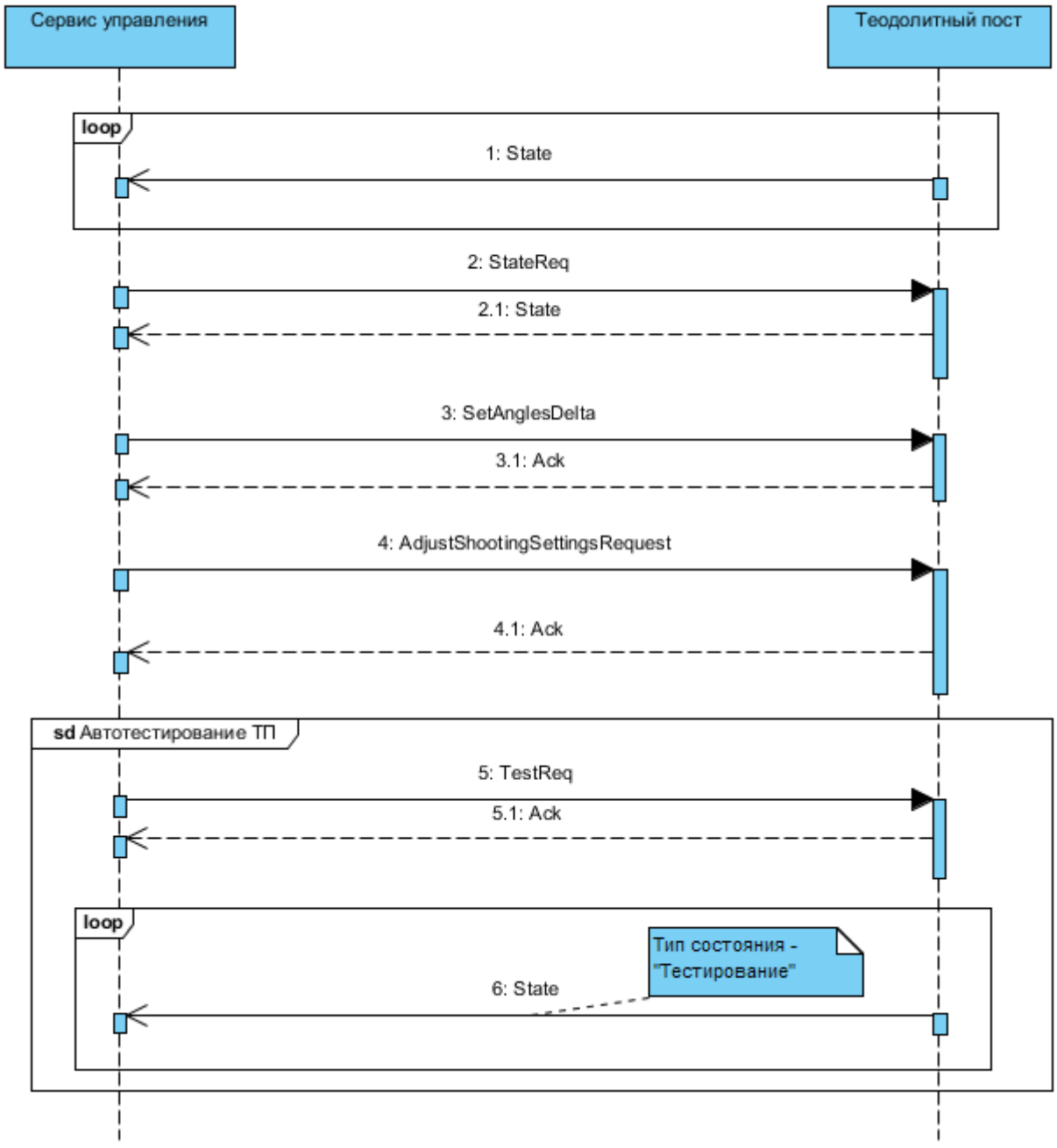

Рисунок 1

Технология информационного обмена.

В соответствии со стандартом OSI протокол взаимодействия предполагает следующие параметры различных уровней:

– физический уровень:

1) тип разъёма: 8P8C;

2) стандарт интерфейса: RJ45;

– канальный уровень:

1) формат кадра (может быть описан в таблице);

2) параметры вычисления CRC:

- вычисляется для всего пакета, включая маркер и длину;
- используется 16-bit CRC-CCITT 0x1021 с начальным значением 0;
- настройки CRC в терминах boost::crc:.

```
boost::crc_optimal < 16, 0x1021, 0, 0, false, false >
```

– сетевой уровень – протокол взаимодействия: IP версии 4, стандарт RFC 791;

– транспортный уровень – протокол взаимодействия: TCP.

Формат сообщений обмена – protobuf версии 2, описанный в приложении. Ограничения:

– до 100 запросов состояния в секунду;

– задержка ответа на любой запрос не более 100 мс.

Особенности взаимодействия:

– сразу после соединения теодолитный пост начинает слать сообщение «State»;

– по приёму любой команды (запрос состояния командой не является):

1) в случае успешного приёма, расшифровки и принятия команды на исполнение, теодолитный пост возвращает сообщение *Ack;*

2) в случае ошибки расшифровки или принятия команды на исполнение, теодолитный пост возвращает сообщение *Nak* и устанавливает соответствующую строку ошибки в поле *what;*

– все сообщения посылаются через агрегирующее сообщение *CommandEnvelope*, путём установки одного из его опциональных полей.

Приложения.

Состав и описание форматов данных представлен ниже.

```
syntax = "proto2";

import "timesync-messages.proto";
package theodolite.messages;

/* СООБЩЕНИЯ ПРОТОКОЛА ИНФОРМАЦИОННОГО ВЗАИМОДЕЙСТВИЯ */

/* Конверт, который включает все пересылаемые сообщения. Может содержать только одно сообщение + идентификатор запроса. */
message CommandEnvelope {
  required uint32 request_id = 1; // Идентификатор запроса (счётчик)
  optional Ack ack = 2;
  optional Nak nak = 3;

  optional StateReq state_req = 4;
  optional State state = 5;
  optional RotateToCoordinatesReq rotate_to_coordinates_req = 6;
  optional AdjustShootingSettingsRequest adjust_shooting_settings_req = 7;
  optional TestReq test_req = 8;
```

```
        optional SetPidRegulatorCoeffReq set_regulator_coeff = 9; // Установить коэффициенты PID-регулятора (для серийного теодолитного поста)
        optional SetAnglesDelta set_angles_delta = 10; // Установить смещения углов относительно истинного 0
    }

    /* Положительное подтверждение (положительная квитанция) на запрос (англ. Positive Acknowledgement (ACK)) */
    message Ack {
    }

    /* Отрицательное подтверждение (англ. Negative AcKnowledgment (NAK)) */
    message Nak {
      required string what = 1; // Причина отказа (произвольная строка с описанием ошибки)
    }

    …
```

Лист регистрации изменений									
Изм.	Номера листов (страниц)				Всего листов (страниц) в документе	Номер документа	Входящий номер сопроводительного документа и дата	Подпись	Дата
	измененных	замененных	новых	аннулированных					