Общество с ограниченной ответственностью "Инфоком-С"

Цифровая платформа управления пространственными данным «Плато»

Описание платформы

Всего листов: 29

2021 г.

АННОТАЦИЯ

В настоящем документе приведены общие сведения о цифровой платформе управления пространственными данными региона «Плато», ее функциональное назначение, описание логической структуры, используемые технические средства, способ вызова, состав входных и выходных данных (далее – Платформы).

Платформа предназначена для реализации широкого комплекса задач по управлению геопространственными данными, а также для построения государственных или корпоративных инфраструктур пространственных данных.

Оформление программного документа «Описание программы» произведено по требованиям ЕСПД (ГОСТ 19.101-77 [1)], ГОСТ 19.103-77 [2)], ГОСТ 19.104-78* [3)], ГОСТ 19.105-78* [4)], ГОСТ 19.106-78* [5)], ГОСТ 19.402-78* [6)], ГОСТ 19.604-78* [7)]).

[1)] ГОСТ 19.101-77 ЕСПД. Виды программ и программных документов

[2)] ГОСТ 19.103-77 ЕСПД. Обозначение программ и программных документов

[3)] ГОСТ 19.104-78* ЕСПД. Основные надписи

[4)] ГОСТ 19.105-78* ЕСПД. Общие требования к программным документам

[5)] ГОСТ 19.106-78* ЕСПД. Общие требования к программным документам, выполненным печатным способом

[6)] ГОСТ 19.402-78* ЕСПД. Описание программы

[7)] ГОСТ 19.604-78* ЕСПД. Правила внесения изменений в программные документы, выполненные печатным способом

СОДЕРЖАНИЕ

1. ОБЩИЕ СВЕДЕНИЯ

1.1. Наименование и обозначение комплекса программ

1.1.1. Наименование: Цифровая платформа управления пространственными данными «Плато» (сокращенно – Платформа).

1.2. Программное обеспечение, необходимое для функционирования платформы

Платформа функционирует на базе следующего общего и системного программного обеспечения:

1) Docker;
2) PostgreSQL с расширением PostGIS;
3) Mongo DB;
4) Hasura;
5) Vue;
6) DotNetCore;
7) Graylog;
8) NGINX.

1.3. Язык программирования, на котором написана платформа

Языки программирования, используемые фреймворки и СПО Комплекса программ оркестровки сервисов:

1) Язык программирования C#;
2) Язык программирования Python;
3) Язык программирования JavaScript;

2. ФУНКЦИОНАЛЬНОЕ НАЗНАЧЕНИЕ

2.1. Назначение платформы

Цифровая платформа управления пространственными данными «Плато» представляет собой программный комплекс, реализующий широкий комплекс задач по управлению геопространственными данными, предназначенный для построения государственных или корпоративных инфраструктур пространственных данных.

Выполняемые функции:

- управление моделью данных о территории на основе онтологии;
- управление источниками данных;
- управление интерфейсом прикладных приложений, развертываемых на базе единой платформы;
- управления картографическими данными;
- управление аналитическими данными;
- управление публикацией данных для внешних потребителей;
- разграничение доступа к данным.

2.2. Сведения о функциональных ограничениях на применение

2.2.3 Технические характеристики

В таблице 1 перечислены технические характеристики Платформы.

Таблица 1 – Технические характеристики Платформы

№	Характеристика	Значение
1	Максимальное количество одновременного взаимодействия с сервисами	> 100 шт.
4	Максимальное ожидание отклика сервиса поставщика	<= 3 мин.
7	Максимальная скорость получения данных по одному каналу	>= 10 Мб/с

8	Максимальное количество одновременных потоков получения данных продукта	>= 10 шт.
11	Максимальная скорость передачи данных продуктов опубликованных в виде OGC-сервиса	>= 2 Мб/с
12	Максимальная скорость передачи данных продуктов опубликованных в виде файла	>= 2 Мб/с
13	Минимальная пропускная способность канала связи с сетями передачи данных, использующих протоколы TCP/IP, включая информационно-телекоммуникационную сеть «Интернет»	>= 100 Мб/с
14	Минимальная частота запросов к API компонент ИС ФОИВ, включая: Росгидромет, МЧС, Минприроды, Минсельхоз, Росреестр, Росрыболовство, Россельхознадзор, и организациями РАН	>= 10 шт./сут.
15	Минимальная частота запросов к API компонент систем и комплексов	>= 10 шт./сут.
16	Максимальное возможное выделяемое для виртуальных ЭВМ общее количество вычислительных ядер	>= 150 шт.
17	Максимальное возможное выделяемое для виртуальных ЭВМ общий размер ОЗУ	>= 1000 Гб
18	Максимальное возможное выделяемое для виртуальных ЭВМ общий размер ПЗУ	>= 50 Тб
19	Максимальная скорость передачи данных в систему по одному каналу от поставщиков	>= 10 Мб/с
20	Максимальное количество одновременных потоков передачи данных в систему от поставщиков	>= 10 шт.

21	Максимальный размер передаваемого пакета данных от поставщика	>= 4 Гб
22	Средний объем данных хранимых потребителем для использования в качестве исходных данных	>= 3 Гб
31	Средний размер системного сообщения	>= 300 байт
32	Средняя скорость создания резервной копии БД	<= 10 Мб/с
33	Минимальное количество хранимых резервных копий БД	>= 2 шт.
34	Максимальная частота отправки запросов к серверу	<= 50 шт./с
35	Максимальное время ожидания ответа на запрос от сервисов системы	<= 5 с

3. ОПИСАНИЕ ЛОГИЧЕСКОЙ СТРУКТУРЫ

3.1. Алгоритм программы

Алгоритм работы Платформы описывается следующими основными сценариями взаимодействия между программными компонентами:

- сценарий «Создание модели данных о территории»;
- сценарий «Создание прикладных приложений»;
- сценарий «Управления аналитическими данными».

3.1.1. Сценарий «Управления аналитическими данными»

Сценарий «Регистрация сервиса выполнения заказа» описывает программы, процессы и потоки данных при выполнении создания онтологической модели данных на основе базовой модели данных о территории

Для управления Моделью был разработаны следующие компоненты Платформы:

- сервис данных о территории;
- визуальный компонент «Конструктор модели данных»

Визуальный компонент «Конструктор модели данных» в процессе функционирования осуществляет ввод параметров конструктора – семантической модели путем генерации OWL-файла, либо загрузку OWL-файла, полученного из внешних источников.

Визуальный компонент «Конструктор модели данных» обладает базовым набором функций, позволяющих отображать онтологическую модель данных, а также выгружать Модель посредством доступа к сервису данных о территории, а также передавать сервису данные о вносимых в Модель изменениях.

Сервис данных о территории в процессе функционирования осуществляет:

- получение онтологии данных в формате OWL-файла, и SPARQL-запросов;
- верификацию вносимых изменений путем запроса прав доступа к сервису разграничения доступа (на основе модели доступа на основе атрибутов «Attribute-Based Access Control» (ABAC));

- запись Модели (или редактируемых элементов Модели) в Хранилище;

- верификацию вносимых изменений путем проверки соответствия OWL-файла и модели Хранилища, которая реализуется путем SPARQL-запросов по свойствам объектов и сравнения с результатами соответствующей выборки из Хранилища;

- выполняет поиск существующих эквивалентов при добавлении в модель нового класса на основе:

- совпадения имени класса;

- вычисления реляционной, атрибутивной и иерархической меры близости с выводом значений в составе массива отладочных данных в визуальный компонент «Конструктор модели данных».

Сценарий «Управления аналитическими данными» представлен на рисунке 1.

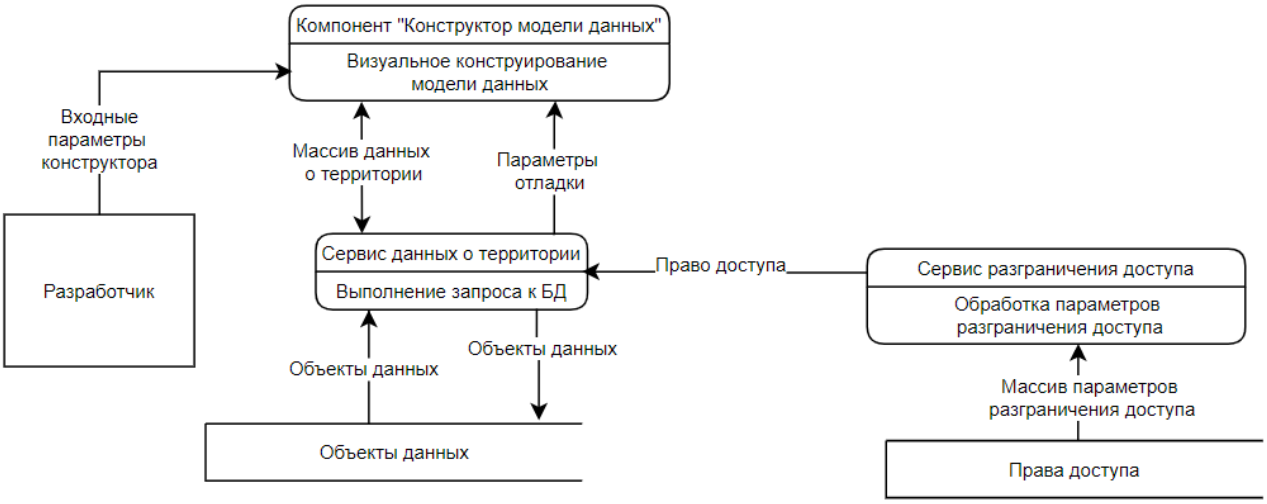

Рисунок 1 – Сценарий «Управления аналитическими данными»

3.1.2. Сценарий «Создание прикладных приложений»

Сценарий «Создание прикладных приложений» описывает программы, процессы и потоки данных при управлении сервисом оператором.

Визуальный компонент «Конструктор источников данных» реализует пользовательский интерфейс, осуществляющий навигацию по семантическому графу и добавление выбранных DataProperties, ObjectProperties в формируемый источник данных.

Сформированный источник данных добавляется в репозиторий сервиса источников данных и становится доступным для других компонентов Платформы.

Визуальный компонент «Конструктор приложений» реализует пользовательский интерфейс, осуществляющий формирование манифеста приложения путем выбора основных его компонент в визуальном редакторе экранной формы приложения. Далее, сохраненный манифест приложения обрабатывается сервисом приложений и возвращает в «Конструктор приложений» параметры отладки. После выполнения, приложение публикуется в репозитории и становится доступным для других компонентов Платформы. В процессе функционирования прикладных приложений сервис приложений осуществляет информационное взаимодействие с визуальными компонентами пользовательского интерфейса, извлечение данных о территории (от сервиса данных о территории) в соответствии с описанными в манифесте сценариями обработки данных, а также с сервисом разграничения доступа путем запроса прав доступа (на основе модели доступа на основе атрибутов «Attribute-Based Access Control» (ABAC)).

Сценарий «Создание прикладных приложений» представлен на рисунке 2.

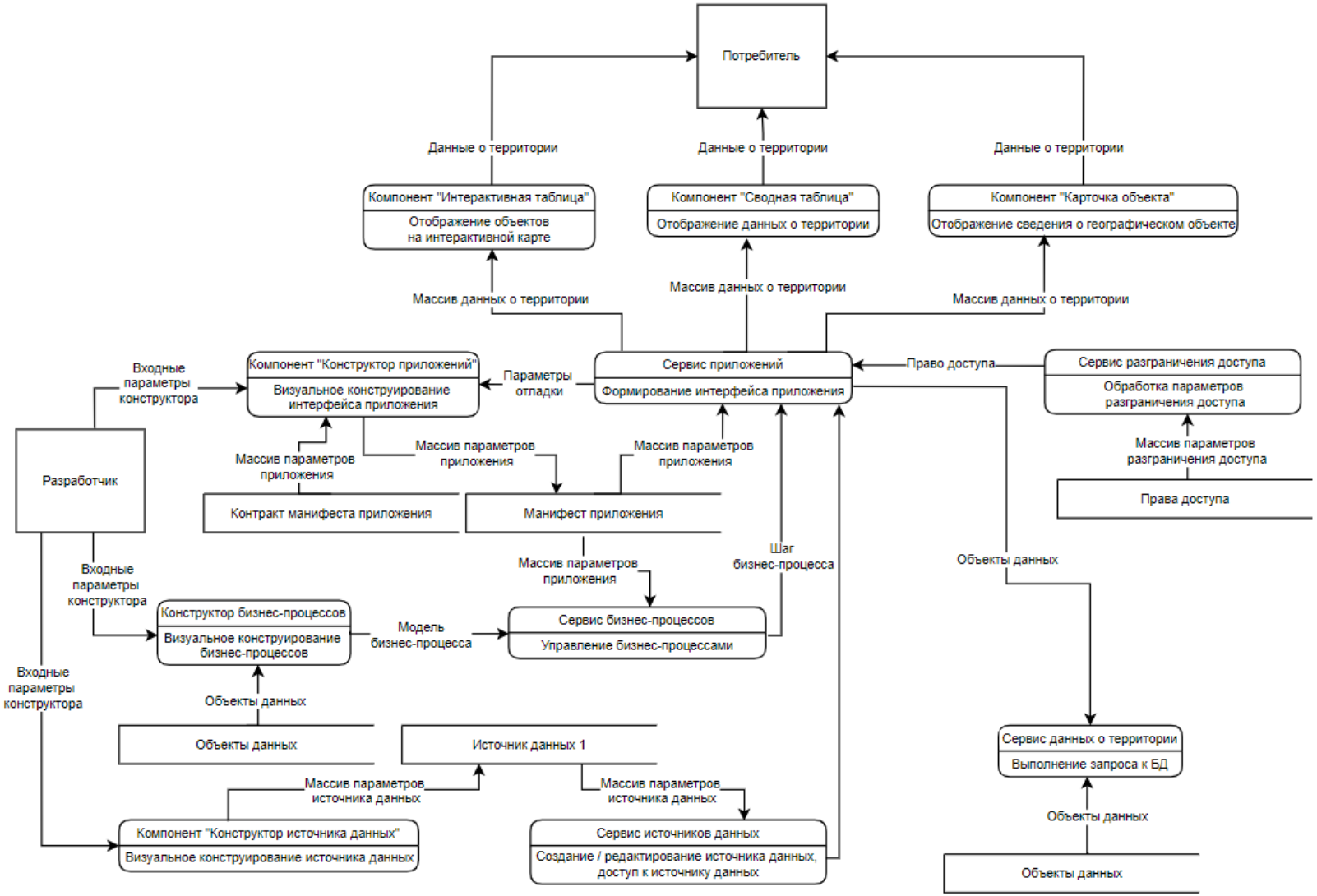

Рисунок 2 – Сценарий «Создание прикладных приложений»

3.1.2 Сценарий «Управления аналитическими данными»

Сервис аналитики в процессе функционирования осуществляет получение параметров аналитической модели от визуальных компонент «Сводная таблица», «Отчет», «Интерактивная карта», «Метрика» или Хранилища, источников данных для проведения аналитического расчета и выполняет выпорку исходных данных путем запроса к сервису данных о территории.

Также, в зависимости от расчетной модели и результатов обработки данных сервис аналитики может возвращать сводные данные сервису данных о территории для обогащения или корректировки Модели в соответствии со сценариями корректировки Модели, описанными в алгоритме обработки аналитических данных:

- новые атрибуты – например, значения показателей, рассчитанных в результате статистических или аналитических расчетов – прогнозируемых значений временных рядов, индекс пространственной автокорреляции, значениями коэффициентов корреляции между различными атрибутами и т.п.;

- новые классы – в результате применения алгоритмов кластеризации над экземплярами выбранных классов (подклассов);

- новые связи между экземплярами и классами – добавление экземпляров в класс в результате выполнения классификации.

Сценарий «Управления аналитическими данными» представлен на рисунке 3.

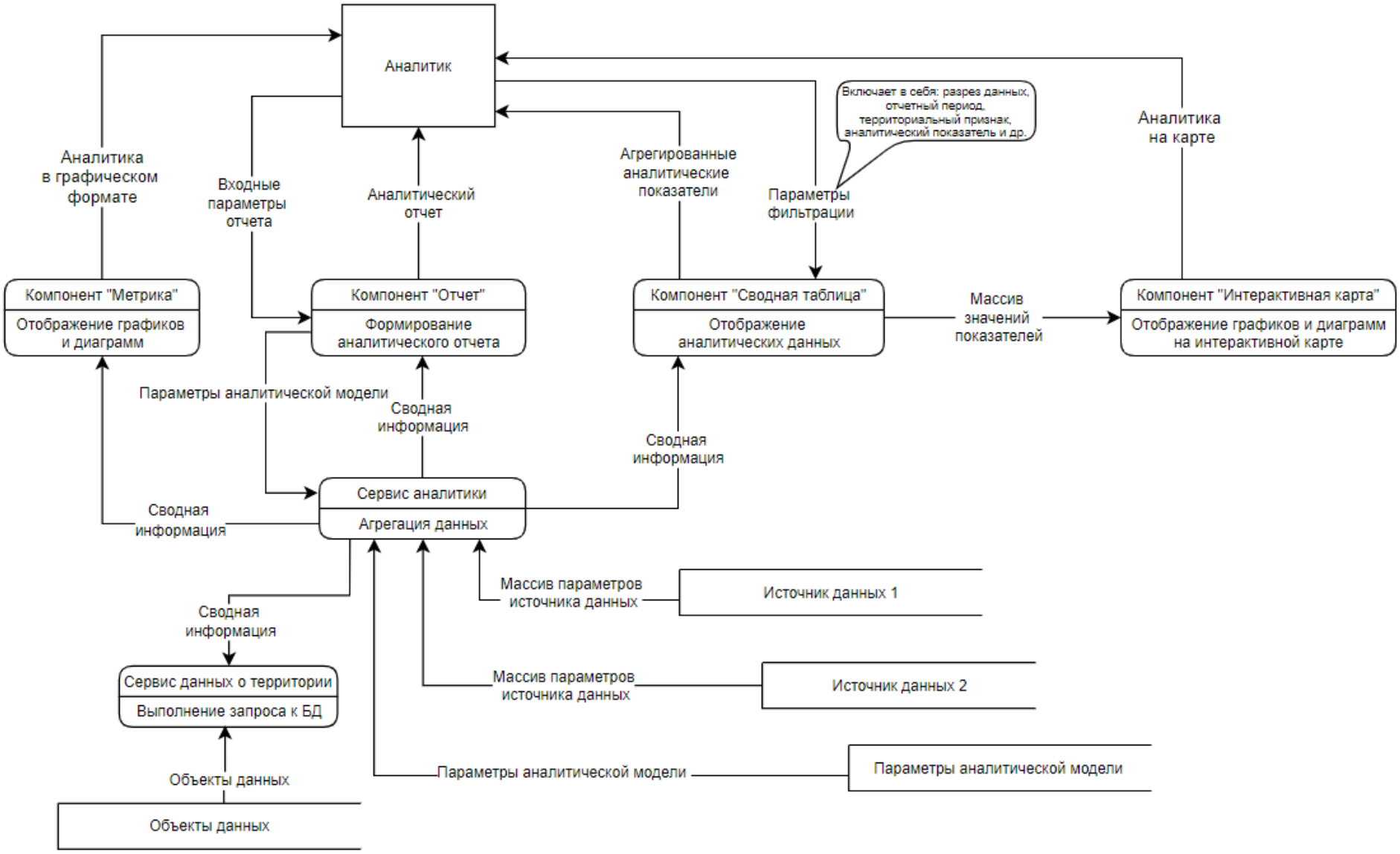

Рисунок 3 – Сценарий «Управления аналитическими данными»

3.2. Используемые методы

Используемые методы основаны на возможностях готового открытого прикладного программного обеспечения, входящего в состав Платформы. Перечень методов приведен в таблице 2.

Таблица 2 – Перечень методов, способствующих повышению уровня качества программного обеспечения

№	Наименование метода	Описание метода
1.	RAD	Концепция создания средств разработки программных продуктов, уделяющая особое внимание быстроте и удобству программирования, созданию технологического процесса. Применение данного метода при разработке информационной системы позволяет программисту максимально быстро создавать компьютерные программы, а Исполнителю максимально быстро и в срок выполнить работы по непосредственному программированию системы.
2.	Метод шаблонного проектирования (паттерны проектирования)	Объектно-ориентированное программирование с использованием шаблонов проектирования позволяет облегчить работу проектировщиков и разработчиков программного обеспечения и ускорить процесс разработки итоговых документов. Использование данного метода в значительной мере способствует повышению качества выполнения соответствующей части работ проекта за счет: унификации проводимых работ; повышения скорости создания форм; удобочитаемости предлагаемых шаблонов.
3.	Проблемно-ориентированное программирование	Использование данного метода приводит к созданию программных абстракций, которые называются моделями предметных областей. В эти модели входит сложная бизнес-логика, устраняющая промежуток между реальными условиями области применения продукта и кодом, что повышает уровень проектируемой системы
4.	Обобщенное программирование	Применение комбинированного подхода к программированию позволяет значительно повысить качество программного кода и качество информационной системы в целом.

5.	Процедурное программирование	Данная методика программирования является классической и используется при программировании информационной системы.
6.	Метод экстремального программирования	Применение данного метода позволяет сделать процесс программирования более гибким, что повысит качество программного кода и качество информационной системы в целом
7.	Управление конфигурацией	Метод предусматривает гибкую настройку и анализ конфигурации. Использование данного метода в области информационных технологий в значительной мере повышает качество проработки вопросов, сокращает время на анализ и обработку информации.
8.	Метод разработки регламентов	Данный метод позволяет разработать порядок взаимодействия между различными объектами или по работе с информационной системой. Использование данного метода позволяет регламентировать правила и требования по работе с системой в формальной форме
9.	Метод работы с VPN	Данный метод применяется при настройке серверных компонент прикладного решения
10.	Метод работы с WEB-технологиями	Веб-технологии дают существенный выигрыш во времени при сборе и формировании мониторинговой информации о внедрении модели управления образовательным учреждением с учетом направлений, рассылке запросов, нормативной, методической и распорядительной документации, позволяют осуществлять поиск и передачу информации. Данный метод существенно повысит качество выполняемых работ за счет сокращения временных затрат на рассылку необходимых документов и сбор информации.
11.	Метод сценариев	Данный метод даст возможность оценить наиболее вероятный ход развития событий и возможные последствия принимаемых решений; определить возможные тенденции развития, взаимосвязи между действующими факторами, сформировать картину возможных состояний, к которым может прийти ситуация под влиянием тех или иных воздействий.
12.	Метод графов	Объектные графы обеспечивают простой способ учёта взаимных связей в множестве объектов, и не обязательно, чтобы эти связи в точности проецировались в классические связки объектно-ориентированного программирования
13.	Инструментальные средства OLAP	Метод обработки данных, заключающаяся в подготовке суммарной (агрегированной) информации на основе больших массивов данных, структурированных по многомерному принципу необходимо на этапе разработки модели информационной системы

14.	Метод ARIS	Метод моделирования «сущность-связь» является основой проектирования баз данных любых информационных систем. На основе анализа предметной области, проведенного одним из методов структурного анализа (SADT, DFD, IDEF3), выявляются сущности, связи и атрибуты, что позволяет спроектировать базу данных, адекватно описывающую предметную область информационной системы
15.	Построение ER-моделей (разработка диаграмм "сущность-связь")	Данная методика позволяет описывать концептуальные схемы предметной области. ER-модель используется при высокоуровневом (концептуальном) проектировании баз данных. С её помощью можно выделить ключевые сущности и обозначить связи, которые могут устанавливаться между этими сущностями.
16.	Построение информационно-логических моделей	Применение данного метода позволяет отобразить данные предметной области в виде совокупности информационных объектов и связей между ними. Эта модель представляет данные, подлежащие хранению в базе данных.
17.	Прототипирование	Использование данного метода при разработке информационных систем заключается в создании прототипов систем, каждый следующий из которых более близок к требованиям Заказчика к информационной системе
18.	Метод анализа информационных потоков	Анализ информационных потоков позволяет выявить схему работы объектов управления, обеспечивает информационное отображение объекта управления, взаимосвязь между его элементами, структуру и динамику информационных потоков. Использование данного метода позволяет провести качественный анализ документации, связанной с информационными потоками, тем самым выполнить требования к автоматизируемым объектам.
19.	Метод визуального проектирования	Применение данного метода необходимо в том случае, когда возникает принципиальная трудность адекватной формализации процесса создания отдельных частей информационной системы. Данный метод расширяет возможности программирования, что позволяет повысить качество программного продукта в целом и обеспечить выполнения требований к заказчику.
20.	Метод визуализации	Позволит разработать наиболее оптимальный, с точки зрения восприятия информации и понимания основного смысла, макет программы.
21.	Метод автоматического	Метод автоматического протоколирования работы программы позволяет автоматически получать

	протоколирования работы программы	подробные протоколы выполняемых программой действий. Использование данного метода позволяет оперативно обнаруживать узкие места в разрабатываемой архитектуре и проводить оптимизацию этих участков, повышая тем самым качество конечного программного продукта.
22.	Журналирование	Данная методика позволяет производить запись в хронологическом порядке операций обработки данных. Применение данной методики позволяет проводить контроль за выполнением работ по реализации проекта, а также применять в информационной системе функционал по контролю за действиями пользователей системы.
23.	Метод авторизации	Предоставление определённому лицу или группе лиц прав на выполнение определённых действий; а также процесс проверки (подтверждения) данных прав при попытке выполнения этих действий является основным инструментом при работе с пользователями информационной системой.
24.	Метод асинхронного запроса	Данный метод используется для интеграции системы с другими автоматизированными учетными системами. Позволяет организовать передачу данных с проверкой на корректность
25.	Метод верификации	Использование данного метода в значительной мере способствует повышению качества выполнения соответствующей части работ проекта за счет использования процедуры формирования финальных версий документов при формировании отчетов о проведении мероприятий по внедрению и эксплуатации информационной системы.
26.	Метод интерактивного тестирования программного кода	Метод проверки кода в интерактивном режиме позволяет исправлять ошибке в процессе проверки. Использование данного позволяет ускорить процесс тестирования.
27.	Метод объектно-ориентированного программирования	Применение метода объектно-ориентированного программирования позволяет разрабатывать сложные и в тоже время качественные программы. Объектно-ориентированное программирование (ООП) – парадигма программирования, в которой основными концепциями являются понятия объектов и классов или прототипов. В настоящее время именно эта парадигма программирования используется в подавляющем большинстве промышленных проектов.
28.	Метод оптимизации запросов СУБД	Метод оптимизации запросов заключается в модификации исходных запросов к БД с целью повышения быстродействия программы. Использование

		данного метода позволяет повысить качество работы за счет увеличения скорости работы программного продукта.
29.	Метод оптимизации производительности программного кода	Метод оптимизации производительности программного кода заключается в модификации исходного кода с целью повышения быстродействия программы. Использование данного метода позволяет повысить качество работы за счет увеличения скорости работы программного продукта.
30.	Метод программирования	Под программированием понимается весь спектр активностей, связанных с созданием и поддержанием в рабочем состоянии программных продуктов. Применение данного методологического подхода является одной из необходимых стадий разработки программных продуктов и интерактивных моделей.
31.	Метод работы со случаем (кейс)	Использование этого метода обеспечит получение новой информации, необходимой для качественной разработки документов по проекту
32.	Метод разработки регламентов взаимодействия	Данный метод позволяет разработать совместно Исполнителю и Заказчику разработать порядок взаимодействия по широкому кругу вопросов. Повышает качество выполнения работ и формирования списка требований Заказчика к Исполнителю и наоборот
33.	Метод разработки диаграммы активности	Использование данного метода необходимо для унифицированной разработки процесса разработки программного продукта с использованием нотация языка UML.
34.	Метод синхронного запроса	Данный метод используется для интеграции системы с другими автоматизированными учетными системами. Позволяет организовать простую передачу данных.
35.	Метод системного анализа	Использование данного методологического подхода при разработке документов позволит структурировать, качественно проанализировать и сформировать информацию и предполагаемый состав разрабатываемых документов, что повысит их качество. Использование данного метода в значительной мере будет способствовать глубине проработки представляемых материалов для достижения целей и выполнения задач проекта с высоким качеством и в установленные сроки. Научный метод познания, представляющий собой последовательность действий по установлению структурных связей между переменными или элементами исследуемой системы.
36.	Метод системного анализа при	Использование данного метода обеспечивает комплексность разрабатываемой концепции.

	разработке концепции проекта	
37.	Метод шаблонного проектирования (паттерны проектирования)	Объектно-ориентированное программирование с использованием шаблонов проектирования позволяет облегчить работу проектировщиков и разработчиков программного обеспечения и ускорить процесс разработки итоговых документов. Использование данного метода в значительной мере способствует повышению качества выполнения соответствующей части работ проекта за счет: унификации проводимых работ; повышения скорости создания форм; удобочитаемости предлагаемых шаблонов.
38.	Метод эргономического тестирования	Использование этого метода обеспечит полноту, достоверность, корректность и обоснованность информации, необходимых для качественной разработки документов, позволит всесторонне оценить результат выполнения работ по проекту
39.	Метод эргономичного дизайна	Метод позволяет оценивать характеристики пользовательского интерфейса, диалог системы с пользователем в процессе взаимодействия с системой и в условиях существенного влияния на него факторов внешней среды, сопоставлять результаты работ с руководящими и нормативными документами по эргономике и проекту
40.	Метод юзабилити-тестирования	Использование данного метода в значительной мере способствует повышению качества и оперативности выполнения проекта за счет: представления широкому кругу людей получения наглядной объективной информации о существующем состоянии исследуемого объекта, ходе выполнения и результатах реализации проекта; интерактивного обсуждения о ходе выполнения и результатах реализации проекта; экономии на транзакционных издержках (отсутствие необходимости рассылки почтой, перевозки, проведения семинаров, совещаний, опросов и т.д.); получения комплексного представления о ходе реализации проекта; экономии времени на осмысление и обработку данных; возможности применения электронно-вычислительной техники с целью повышения скорости представления данных.
41.	Нагрузочное тестирование	Применение данного метода позволяет проводит тестирование с точки зрения определения или сбор показателей производительности и времени отклика программно-технической системы или устройства в ответ на внешний запрос с целью установления соответствия требованиям, предъявляемым к данной

		системе (устройству). Данный метод позволяет повысить качество проведения тестирования.
42.	Событийно-ориентированное программирование	Применение данного метода расширяет возможности программирования, что делает процесс создания информационной системы более гибким, выполняя требования к информационной системе.
43.	Создание баз данных	Использование методов позволяет собирать и хранить информацию по различным сторонам проекта, упорядочивать ее и предлагать для оперативного использования, экономя время и средства проекта.
44.	Стресс-тестирование	Применение одной из форм тестирования, которая используется для определения устойчивости системы в условиях превышения пределов нормального функционирования, позволяет проверить соответствие системы требованиям Заказчика к устойчивости
45.	Тестирование	Использование данного метода в значительной мере способствует повышению качества и оперативности выполнения проекта за счет: представления широкому кругу людей получения наглядной объективной информации о существующем состоянии исследуемого объекта, ходе выполнения и результатах реализации проекта; интерактивного обсуждения о ходе выполнения и результатах реализации проекта; экономии на транзакционных издержках; получения комплексного представления о ходе реализации проекта; экономии времени на осмысление и обработку данных; возможности применения электронно-вычислительной техники с целью повышения скорости представления данных. Метод востребован при проведении независимой оценки качества реализации проводимых в рамках проекта работ
46.	Тестирование по документации (формальное тестирование)	Применение данной методики тестирования программного обеспечения позволит повысит качество проверки программного обеспечения на наличие ошибок и достижения поставленных целей.
47.	Тестирование при сдаче	Применение данной методики тестирования программного обеспечения позволит повысит качество проверки программного обеспечения на наличие ошибок и достижения поставленных целей.
48.	Тестирование производительности	Применение данной методики тестирования программного обеспечения позволит повысит качество проверки программного обеспечения на наличие ошибок и достижения поставленных целей.
49.	Тестирование совместимости	Применение данной методики тестирования программного обеспечения позволит повысит качество

		проверки программного обеспечения на наличие ошибок и достижения поставленных целей.
50.	Тестирование стабильности	Применение данной методики тестирования программного обеспечения позволит повысит качество проверки программного обеспечения на наличие ошибок и достижения поставленных целей.
51.	Пузырьковый метод сортировки	Использование данного метода эффективно для небольших алгоритмов. Данный метод лежит в основе более совершенных алгоритмов сортировки.
52.	Метод расчета контрольных сумм по алгоритму хеширования ГОСТ Р 34.11-2012 с длиной хэш-кода 512 бит	Применение данного метода позволяет проверять целостности данных при их передаче с помощью криптографический алгоритм вычисления хеш-функции с размером блока входных данных и размером хеш-кода в 512 бит.
53.	Метод контроля целостности данных по алгоритму CRC	Применение данного метода позволяет проверять целостности данных при сохранении. CRC является практическим приложением помехоустойчивого кодирования, основанным на определённых математических свойствах циклического кода
54.	Метод обеспечения сжатия данных по алгоритму LZW	Преимущество использования данного алгоритма является сжатия данных без потерь. Один из самых универсальных алгоритмов сжатия данных на сегодняшний день.
55.	Метод загрузки данных с помощью кеширования	Применение данного метода позволяет эффективно использовать повторно ранее полученные данные, тем самым ускоряет процесс загрузки информации.
56.	Метод приоритетов для формирования и выполнения очередей заданий	Использование данной методики позволяет устанавливать приоритет на основе статических и динамических характеристик элементов.
57.	Метод очереди fifo	Применение данного метода осуществляет способ организации и манипулирования данными относительно времени и приоритетов. Принцип выполнения данного метода в следующем - тот, кто приходит первым, тот и обслуживается первым, пришедший следующим ждёт, пока обслуживание первого не будет закончено, и так далее.
58.	Метод использования направленных ациклических графов (DAG)	Применение данного метода осуществляет собой топологическую сортировку, где каждый объект или фрагмент данных находится в определённом порядке. Конструкция DAG состоит из вершин, соединяемых рёбрами. Основной алгоритм DAG называется

		топологическим распределением, где каждое ребро направлено от более раннего ребра к более позднему.
59.	Технология тестирования "Черного ящика"	Данная технология описывает порядок проведения тестирования "черного ящика", используемого для обеспечения комплекса мероприятий проверки качества информационной системы

3.4. Связи программы с другими программами

Платформы предназначена для функционирования в изолированной инфраструктуре компании ООО «Инфоком-С». Взаимодействия с другими программами и внешними ИС не предусмотрено.

4. ИСПОЛЬЗУЕМЫЕ ТЕХНИЧЕСКИЕ СРЕДСТВА

Для функционирования Платформы необходимы технические средства, приведенные в таблице 3.

Таблица 3 – Требования к характеристикам технических средств, обеспечивающих функционирование комплекса программ

Характеристика, единица измерения	Значение	
	мин.	опт.
Частота ядра ЦП, ГГц	3	3
Количество логических вычислительных ядер ЦП, шт.	4	8
Информационная емкость ОЗУ, Гб	8	16
Информационная емкость ПЗУ, Гб	256	512
Максимальная пропускная способность передачи данных сети, Mbs	100	1000

5. ВЫЗОВ И ЗАГРУЗКИ

Загрузка компонентов Платформы осуществляется путем выполнения команды ПО Docker **docker compose up** в директории с файлом конфигурации сервисов с именем docker-compose.yml

После загрузки Платформа переходит в дежурный режим ожидая вызова запросов от компонент. Вызов осуществляется с помощью согласованных HTTP-запросов, содержащие необходимую информацию для начала выполнения заложенных функций программ, путем взаимодействия с API.

Для начала работы с порталом Платформы необходимо в браузере перейти по адресу http://plato.infolan.org/.

Вызов компонентов Платформы осуществляются с помощью согласованных HTTP-запросов, содержащие необходимую информацию для начала выполнения заложенных функций программ, путем взаимодействия с API сервиса приложений, сервиса источников данных и сервиса данных (о территории).

Для выполнения сценария «Создание модели данных о территории» пользователь должен открыть конструктор модели данных о территории http://plato.infolan.org/#/app_model. Для выполнения сценария «Создание прикладных приложений» пользователь должен открыть конструктор приложений и источников данных на странице http://plato.infolan.org/#/app_dev. Для выполнения сценария «Управления аналитическими данными» пользователь должен интерфейс пользователя приложения для управления аналитическими данными на странице http://plato.infolan.org/#/app10.

Вызов компонентов Платформы публикации файлов по стандартам HTTP/FTP и Платформы публикации по стандартам OGC осуществляются с помощью согласованных HTTP-запросов, содержащие необходимую информацию для начала выполнения заложенных функций программ, путем взаимодействия с API сервера приложений.

6. ВХОДНЫЕ ДАННЫЕ

В качестве входных данных Платформы выступают:

1) Модель данных в формате OWL;
2) Параметры приложения, вводимые разработчиком
3) Параметры источника данных, вводимые разработчиком;
4) Параметры аналитической модели, вводимые аналитиком.

7. ВЫХОДНЫЕ ДАННЫЕ

В качестве выходных данных Платформы выступают:

1) Данные о приложении. Сведения о прикладном приложении, такие как: название, пользовательское описание, поддерживаемы команды, описание функциональных возможностей, идентификационные данные сервиса.

2) Данные о сервисе. Сведения о сервисе, такие как: название, пользовательское описание, поддерживаемы команды, описание функциональных возможностей, идентификационные данные сервиса.

3) Аутентифицирующая сервис информация. Идентификационные данные сервиса необходимые для получения идентификатора сессии во внешней системе авторизации и аутентификации.

4) Данные о регистрации сервиса. Сведения о состоянии регистрации сервиса выполнения заказов, содержат общую информацию о сервисе, результат авторизации и параметры активной сессии.

5) Команда управления. Формализованная команда управления сервисом, содержит системной имя команды и параметры команды вида ключ- значение.

6) Запрос данных. Универсальный запрос данных, основан на формализованных командах управления сервисами выполнения заказов. Содержит информацию о сервисе, у которого необходимо выполнить запрос, наименовании запроса и параметрах запроса в формате ключ-значение.

7) Исходные данные. В качестве исходных данных могут выступать растровые и векторные данные, бинарные файлы, структурированные файлы, архивы файлов, которые являются результатом заказа, полученного от сервисов поставщиков. Форматы исходных данных: ShapeFile, GML, KML, GeoTIFF, GEOJSON, XML, ZIP.

8) Геоданные. В качестве геоданных выступают файлы в векторных и растровых форматах, а также специализированные базы данных для хранения географических объектов. Форматы геоданных: ShapeFile, GML, KML, GeoTIFF, GEOJSON.

9) Метаданные. В качестве метаданных выступает информация описывающая исходные данные.

10) Запрос на публикацию файла. Формализованный запрос, содержит параметры публикации исходных данных в виде файлов для предоставления итогового результата выполнения заказа. Параметры передаются в формате ключ-значение.

11) Запрос на публикацию геоданных. Формализованный запрос, содержит параметры публикации геоданных по стандарту OGC для предоставления итогового результата выполнения заказа. Параметры передаются в формате ключ-значение;

12) Аналитические данные. Данные в текстовом формате, представленные в виде сводных таблиц, а также файлы в векторных и растровых форматах, отображающие результат аналитической обработки данных над пространственными объектами.

Лист регистрации изменений									
Номера листов (страниц)					Всего листов (страниц) в докум.	№ документа	Входящий № сопр. докум. и дата	Подп.	Дата
Изм.	изменен-ных	заменен-ных	новых	аннулиро-ванных					