ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ УЧРЕЖДЕНИЕ НАУКИ
ИНСТИТУТ ДИНАМИКИ СИСТЕМ И ТЕОРИИ УПРАВЛЕНИЯ
ИМЕНИ В.М. МАТРОСОВА
СИБИРСКОГО ОТДЕЛЕНИЯ РОССИЙСКОЙ АКАДЕМИИ НАУК

На правах рукописи

Фёдоров Роман Константинович

Сервис-ориентированная информационно-аналитическая среда композиции сервисов обработки пространственных данных

Специальность 2.3.5 – Математическое и программное обеспечение вычислительных систем, комплексов и компьютерных сетей

Диссертация на соискание ученой степени
доктора технических наук

Научный консультант –
академик И. В. Бычков

Иркутск – 2024

Оглавление

ВВЕДЕНИЕ

Актуальность. Для современного этапа решения крупномасштабных фундаментальных и прикладных задач актуально повышение эффективности и надежности процессов обработки и передачи данных в вычислительных системах, комплексах и компьютерных сетях путём внедрения сервис-ориентированной архитектуры (СОА, англ. service-oriented architecture, SOA), на основе которой реализуют интеллектуальные технологии и машинное обучение, а также системы обработки больших объемов пространственных данных (ПД).

Сервис-ориентированная архитектура является развитием подходов, применявшихся при создании пакетов прикладных программ (ППП), а затем параллельных и распределенных вычислительных систем, и базируется на теоретических и практических результатах исследований ведущих российских и зарубежных ученых, в том числе С.М. Абрамова, А.И. Аветисяна, А.П. Афанасьева, В.Б. Бетелина, И.В. Бычкова, А.В. Бухановского, Вл.В. Воеводина, В.П. Гергеля, Б.М. Глинского, В.П. Иванникова, И.А. Каляева, В.Н. Коваленко, Д.А. Корягина, В.В. Коренькова, В.Д. Корнеева, И.И. Левина, А.И. Легалова, Л.В. Массель, Ю.Е. Малашенко, В.Э. Малышкина, А.В. Манциводы, Г.А. Опарина, Г.И. Радченко, Г.М. Ружникова, С.И. Смагина, И.А. Соколова, Л.Б. Соколинского, О.В. Сухорослова, В.В. Топоркова, А.Г. Феоктистова, В.Г. Хорошевского, Б.Н. Четверушкина, М.В. Якобовского, A. Ballatore, G. Baryannis, J. Bih, D. Churches, F. Curbera, E. Deelman, L. Di, D. Edmond, M. Farnaghi, D. Fellows, I. Foster, A. Friis-Christensen, G. Gombas, R. Haines, A. Harrison, M. Hinz, A. Hofstede, W. Huang, J. Pereira, G. Juve, Y.-K. Kwok, H. Li, Yu. Liyang, R. Lucchi, M. Lutz, D. Martin, L. Miao, T. Nixon, N. Ostländer, S. Scheider, P. Schut, D. Silva, Z. Sun, H. Topcuoglu, D. Ulutaş Karakol, K. Vahi, K. Wolstencroft, W. Yang, P. Yue, P. Zhao, H. Zhi-Wei, Q. Zhu, C. Zhuang и других известных специалистов.

В 70-х годах прошлого столетия ППП включали различные реализации численного решения вычислительных задач и средств системного обеспечения (программных и языковых). ППП позволяли исследователю получить целый набор различных взаимосвязанных методов, ориентированных на определенную предметную область. Взаимодействие программ происходило только в рамках

пакета через файловую систему с использованием обменных форматов, либо через заданные программные интерфейсы. За рубежом аналогом ППП является система управления научными рабочими процессами (англ., Workflow Management System – WMS). При этом схема или план решения задачи в ППП коррелирует с вышеупомянутым понятием научного рабочего процесса.

В связи с активным использованием Интернета развивались методы интеграции и взаимодействия пакетов программ через сети передачи данных (СПД), что привело к созданию нового направления «распределенных пакетов программ». В том числе были реализованы методы планирования и выполнения распределенных пакетов программ на гетерогенных вычислительных ресурсах и на суперкомпьютерах. Но взаимодействие программ через СПД производилось, в основном, в рамках пакета.

СОА появилась в конце 1980-х и берёт своё начало в идеях, изложенных в CORBA, DCOM, DCE и т. д. Реализация методов в виде сервисов упрощает использование программного обеспечения (ПО), которое сводится к вызову сервиса через Интернет, используя его программный интерфейс. Программные системы, реализованные с помощью данного подхода, обладают следующими преимуществами: сервисы кроссплатформенны, тестируемы, доступны по сети. В рамках СОА нет необходимости устанавливать, конфигурировать, обновлять программное обеспечение. Сервисы имеют низкий порог вхождения для их использования (пользователю нет необходимости изучать подробности реализации метода, достаточно изучить интерфейс).

В настоящее время тенденция развития WMS базируется на разработке сервис-ориентированных научных процессов. Кроме того, в связи с высокой актуальностью задач, решаемых, например, в рамках экологического мониторинга, активно развиваются специализированные системы управления сервис-ориентированными научными процессами в области геоинформатики. Известными примерами таких систем являются BPEL Designer Project (https://projects.eclipse.org/projects/soa.bpel) и GeoJModelBuilder (https://github.com/geoprocessing/GeoJModelBuilder).

В современном мире активно растет количество сервисов, реализующих предоставление, обработку и публикацию данных. Например, это сервисы предоставления данных дистанционного зондирования земли, обработки пространственных данных, расшифровки генома и т. д. Созданные сервисы значительно упрощают решение многих задач. В области обработки ПД определены и активно используются стандарты Open Geospatial Consortium (OGC). При этом возникает ряд сложных научно-технических задач нахождения сервисов, построения их связей между собой, проверки корректности совместного выполнения этих сервисов и др.

Объединение сервисов, т. е. создание их композиции, позволяет решать большое количество задач. СОА значительно упрощает и ускоряет интеграцию программного обеспечения, созданного разработчиками из разных предметных областей, за счет упрощения и стандартизации интерфейсов. Композиции сервисов обеспечивают повышение уровня автоматизации решения задач, начиная от ввода данных и заканчивая публикацией результатов.

В то же время создание композиций сервисов является нетривиальной задачей. Наличие большого количества сервисов, с одной стороны, увеличивает возможности исследователей, а с другой стороны, значительно усложняет поиск нужных сервисов для решения конкретной задачи. Составление найденных сервисов в композиции в некоторых случаях может оказаться комбинаторно сложной задачей, при этом возможна генерация достаточно большого числа альтернативных композиций сервисов. Все эти альтернативы необходимо будет оценить, выбрать наиболее релевантные для решаемой задачи, а также их проверить на возможность взаимодействия. Часто композиция из двух потенциально возможных сервисов не реализуема из-за того, что структуры данных и форматы входных и выходных данных отличаются.

Композиции сервисов позволяют объединить результаты работы исследователей, но создание таких композиций в силу сложного процесса требует высокой квалификации. Поэтому актуальной научной проблемой является разработка новых моделей, алгоритмов, методов и технологии для создания сервис-

ориентированной информационно-аналитической среды (СОИАС) с целью повышения эффективности процессов подготовки и проведения научных экспериментов по решению задач в области геоинформатики за счет автоматизации построения и применения композиций сервисов. Здесь под эффективностью понимается сокращение накладных расходов (стоимостных, временных и др. трудозатрат) на выполнение экспериментов, а также обеспечение новых функциональных возможностей, необходимость которых обуславливается наличием вышеупомянутых сложных научно-технических задач.

Цель исследования заключается в повышении эффективности процессов подготовки и проведения научных экспериментов на основе сервис-ориентированной парадигмы за счет автоматизации построения и применения композиций сервисов, реализующих методы анализа и обработки ПД.

Основные задачи диссертационного исследования

1. Провести анализ инструментальных средств, технологий и существующих информационно-аналитических сред с использованием сервисов обработки ПД и композиций сервисов.
2. Исследовать и разработать модель СОИАС, обеспечивающую создание и обмен композициями сервисов между пользователями.
3. Разработать метод создания композиций сервисов обработки ПД.
4. Разработать методы выполнения композиции сервисов обработки ПД.
5. Разработать основные компоненты среды, реализующие модель СОИАС.
6. Провести апробацию СОИАС на задаче поддержки междисциплинарных научных исследований Байкальской территории.

Объектом исследования является распределенная гетерогенная информационно-вычислительная среда, функционирующая на основе СОА.

Предметом исследования являются модели, методы, алгоритмы и программное обеспечение автоматизации построения и применения композиций сервисов в распределенной гетерогенной информационно-вычислительной среде.

Методы исследования. В работе использовались методы информационного моделирования, теории графов, системного и объектно-ориентированного программирования, проектирования баз данных, построения распределенных комплексов проблемно-ориентированных программ, веб-технологий, планирования выполнения композиций сервисов в статических и динамических средах.

Научную новизну диссертации представляют следующие результаты исследования, выносимые на защиту и расширяющие существующий базис теории и практики сервис-ориентированных вычислений:

1) создана модель сервис-ориентированной информационно-аналитической среды обработки пространственных данных междисциплинарных исследований, которая в сравнении с подобными моделями обеспечивает оценку композиций сервисов на основе многопользовательской статистики их применения;

2) разработан метод создания композиций сервисов, базирующийся на применении предложенной модели, который в отличие от существующих методов проводит комплексный анализ метаданных, онтологий, экспертных знаний и статистики применения сервисов, что позволяет находить композиции сервисов на основе комбинации данных;

3) разработан оригинальный программный инструмент создания сервисов ввода и публикации реляционных данных, обеспечивающий предоставление метаданных, пользовательский и программный интерфейс редактирования данных, поддержку передачи данных WPS сервисам. Создание сервисов данных впервые производится на основе иерархической модели данных с возможностью задания асинхронного вычисления значений атрибутов с помощью сервисов. Создаваемые сервисы можно сразу включать во множество композиций;

4) разработан оригинальный программный компонент выполнения композиций сервисов, заданных на процедурном языке, с обработкой промежуточных данных с помощью средств языка и его библиотек, для которого в отличие от других подходов обеспечивается формирование DAG с помощью процедурного языка и одновременно его планирование и выполнение с учетом добавляемых в процессе выполнения новых заданий в гетерогенной динамической вычислительной среде;

5) разработан комплекс программных компонентов, реализующий модель сервис-ориентированной информационно-аналитической среды, который обеспечил создание композиций сервисов и их обмен между пользователями.

Соответствие диссертации паспорту научной специальности. Тема и основные результаты диссертации соответствуют следующим областям исследований паспорта специальности 2.3.5 – «Математическое и программное обеспечение вычислительных систем, комплексов и компьютерных сетей»:

- модели, методы, архитектуры, алгоритмы, языки и программные инструменты организации взаимодействия программ и программных систем;

- модели, методы, алгоритмы, облачные технологии и программная инфраструктура организации глобально распределенной обработки данных.

Теоретическая значимость работы заключается в развитии методов и средств автоматизации построения и применения композиций сервисов на основе комплексного анализа метаданных, онтологий, экспертных знаний и статистики применения сервисов. Основные результаты диссертационного исследования использованы при выполнении государственных заданий и научных исследований:

- **гранта Министерства науки и высшего образования РФ** на выполнение крупного научного проекта по приоритетным направлениям научно-технического развития «Фундаментальные основы, методы и технологии цифрового мониторинга и прогнозирования экологической обстановки Байкальской природной территории» (№ 075-15-2020-787);

- **проектов Программы фундаментальных исследований Президиума РАН:** программа II.1, тема «Разработка и экспериментальное исследование эффективности методов оценки антропогенного воздействия на окружающую среду» ФНМ-51, 49, 27 (2012 г., 2013-2015 гг., 2018-2019 гг.);

- **проектов РФФИ** 18-07-00758_а (2018-2020 гг.), 17-57-44006 Монг_а (2018-2020гг.), 17-47-380007_р-а (2017-2019 гг.), 17-29-05089_а (2017-2019 гг.), 16-57-44034 Монг_а (2016-2017 гг.), 16-07-00554-а (2016-2018 гг.), 16-07-00411_а (2016-

2018 гг.), 15-47-04348_а (2015-2017 гг.), 14-07-00166_а (2014-2016 гг.), 14-47-04125 р_сибирь_а (2014-2016 гг.), 13-07-12080 офи_м (2014-2016 гг.), 13-05-41105 РГО_а (2013-2014 гг.);

- **проекта Программы фундаментальных исследований Отделения нанотехнологий и информационных технологий РАН** № 4.1 (2012-2014 гг.);

- **проектов междисциплинарных интеграционных программ СО РАН** № 17, 131 (2012-2014 гг.);

- **проектов междисциплинарных интеграционных программ СО РАН и ДВО РАН** № 73, 74 (2012-2014 гг.);

- **базовых проектов Программы фундаментальных исследований СО РАН**:

1. № IV.38.1.2. «Методы и технологии облачной сервис-ориентированной цифровой платформы сбора, хранения и обработки больших объёмов разноформатных междисциплинарных данных и знаний, основанные на применении искусственного интеллекта, модельно-управляемого подхода и машинного обучения» (2020-2021 гг.);

2. № IV.38.1.2 «Методы и технологии создания распределенной сервисно-ориентированной среды сбора, хранения, обработки больших объёмов разноформатных междисциплинарных научных данных и знаний, основанные на конструктивных средствах спецификации, порождающем программировании и интеллектуализации» (2017-2019 гг.);

3. № IV.38.2.3 «Новые методы, технологии и сервисы обработки пространственных и тематических данных, основанные на декларативных спецификациях и знаниях» (2013-2015 гг.);

4. № IV.31.2.4 «Методы и технологии разработки программного обеспечения для анализа, обработки и хранения разноформатных междисциплинарных данных и знаний, основанные на применении декларативных спецификаций форматов представления информации и программных систем» (2010-2012 гг.).

Практическая значимость. Предложенные в рамках диссертационной работы методы, модели, технологии, алгоритмы и программное обеспечение позволяют снизить трудозатраты и сократить сроки разработки программного обеспечения за счет автоматизации построения композиций сервисов обработки междисциплинарных пространственных данных. Комплекс программных компонентов, реализующий модель сервис-ориентированной информационно-аналитической среды, активно используется на практике. В ходе выполнения различных проектов созданы более 200 сервисов предоставления данных, более 40 сервисов обработки данных и 250 сервисов публикации данных. Развернуты 6 различных геопорталов, ориентированных на различные предметные области и коллективы:

1) Геопортал ИДСТУ СО РАН;
2) Информационно-аналитическая система по фиторазнообразию Байкальской Сибири;
3) Атлас ИГ СО РАН;
4) Геопортал ИЗК СО РАН;
5) Очаги распространения иксодовых клещей;
6) Информационная система (ИС) «L.».

В рамках перечисленных геопорталов сформированы композиции сервисов, объединяющие сервисы данных, сервисы обработки и публикации, созданные разными коллективами. Практическая значимость результатов подтверждена полученными актами внедрения комплекса программных компонент ИГ СО РАН, СИФИБР СО РАН, ПАБСИ КНЦ РАН, ИППЭС КНЦ РАН, Самарский университет им. Королева, НЦ ПЗСРЧ. Автором в составе коллектива получено 8 свидетельств о регистрации программ для электронных вычислительных машин (ЭВМ).

Достоверность и обоснованность полученных в диссертации результатов подтверждается корректным применением классических методов исследования, анализом адекватности разработанных моделей и алгоритмов, решением прикладных и тестовых задач, индексацией полученных результатов в РИНЦ, Web of Science, Scopus, активной эксплуатацией СОАИС большим числом пользователей.

Апробация результатов исследования. Основные результаты диссертационного исследования докладывались на следующих научных мероприятиях: Международная конференция «3th Russia and Pacific Conference on Computer Technology and Applications» (Владивосток, 2017 г.); Международная научно-практическая конференция «Использование современных информационных технологий в ботанических исследованиях» (Апатиты, 2017 г.); Всероссийская конференция «Обработка пространственных данных в задачах мониторинга природных и антропогенных процессов» (Новосибирск, 2017 г., Бердск, 2019 г.); XVIII Всероссийская конференция молодых ученых по математическому моделированию и информационным технологиям (Барнаул, 2013 г.; Новосибирск, 2017 г.); Национальный суперкомпьютерный форум (НСКФ, Переславль-Залесский, 2016 г.); Российско-монгольская конференция молодых ученых по математическому моделированию, вычислительно-информационным технологиям и управлению (Иркутск-Ханх (Монголия), 2013, 2015, 2016 гг.); III Всероссийская конференция «Математическое моделирование и вычислительно-информационные технологии в междисциплинарных научных исследованиях» (Иркутск, 2013 г.); Всероссийская конференция «Ляпуновские чтения» (Иркутск, 2014-2017 гг.); Международная конференция «Системный анализ и информационные технологии» (Иркутск, САИТ, Светлогорск, 2015 г., Иркутск, 2019 г.); Scientific-practical Workshop Information Technologies: Algorithms, Models, Systems (ITAMS, Иркутск, 2018-2021 гг.); Всероссийская конференция с международным участием «Информационные и математические технологии в науке и управлении» (ИМТ, Иркутск, 2009, 2011, 2013, 2014, 2016, 2019-2021 гг.); Международная конференция «Современные проблемы дистанционного зондирования Земли из космоса» (Москва, 2019); Всероссийская конференция «Обработка пространственных данных в задачах мониторинга природных и антропогенных процессов» (Бердск, 2019 г.); Международная конференция «Математические и информационные технологии» (MIT-2011, Врнячка Баня, Сербия, 2011 г.); «Международная географическая конференция» (Иркутск, п. Листвянка, 2018 г.); VIII Всероссийская конференция «Безопасность и мониторинг природных и техногенных систем» (Красноярск, 2023 г.); Всероссийская конференция «Информационные технологии в управлении» (Санкт-Петербург, 2020 г.); 16-я Международная конференция «Системный анализ,

управление и навигация» (Евпатория, 2011 г.); Всероссийская конференция «Распределенные информационно-вычислительные ресурсы. Наука – цифровой экономике» (DICR, Новосибирск, 2017 г.); Международный научно-технический конгресс «Интеллектуальные системы и информационные технологии» (Дивноморское, 2022 г.), а также семинарах ИДСТУ СО РАН.

Личный вклад автора. Все выносимые на защиту научные положения получены соискателем лично. Из совместных исследований в диссертацию включены только те результаты, которые принадлежат непосредственно автору. Модель сервис-ориентированной информационно-аналитической среды, метод создания композиций сервисов [138-144], программный инструмент создания сервисов ввода и публикации реляционных данных, комплекс программных компонентов, реализующий модель сервис-ориентированной информационно-аналитической среды [145 - 160], программный компонент выполнения композиций сервисов, заданных на процедурном языке, [161-167], содержащие непосредственный творческий вклад автора на всех этапах – от постановки задач до разработки правил, моделей, программных компонентов, соискателем разработаны лично. Программная реализация модуля интерпретации сценариев выполнена совместно с Шумиловым А.С. Часть компонентов и сервисов СОИАС, которые непосредственно не касаются создания композиций сервисов, но необходимы для полноценной работы среды, разработаны совместно с коллегами: Авраменко Ю.В., Хмельновым А.Е., Поповой А.К., Ветровым А.А., Парамоновым В.В. Совместно с научным консультантом Бычковым И.В. и Ружниковым Г.М. выделены проблемы, сформулированы направления, и обобщены результаты исследований, сформулированы научные выводы и практические рекомендации.

Публикации. Результаты диссертационного исследования отражены в 45 научных работах. Основные публикации представлены в 4 монографиях, 9 статьях в российских журналах, рекомендованных ВАК для опубликования научных результатов диссертации, а также в 32 статьях, проиндексированных в международных базах цитирования Web of Science и Scopus. Автором в составе коллектива получено 8 свидетельств о регистрации программ для электронных вычислительных машин (ЭВМ) [192 – 201].

Структура диссертации. Диссертация состоит из введения, пяти глав, заключения, библиографии из 201 наименования, списка принятых сокращений и 12 приложений. Объем основного текста работы – 195 страниц, включая 5 таблиц и 81 рисунок. Общий объем диссертации 271 страница.

Автор выражает глубокую благодарность научному консультанту академику И.В. Бычкову за чуткое неустанное сопровождение в работе, помощь в преодолении трудностей и постановку нестандартных, увлекательных и интересных задач, Ружникову Г.М. за полную поддержку всех идей и начинаний, своим коллегам – соавторам за помощь в разработке компонентов, полезные советы и конструктивную критику, а также сотрудникам ИДСТУ СО РАН за обсуждение и полезные замечания при выполнении диссертационной работы.

ГЛАВА 1. ОБЗОР

1.1. Сервис-ориентированная архитектура

Определения

Развитием распределенных вычислений является сервис-ориентированный подход, в рамках которого различные программы, алгоритмы, источники данных публикуются в виде независимых сервисов. Сервис – это механизм, позволяющий получать доступ к одному или нескольким программным средствам с использованием специального интерфейса в соответствии с ограничениями, правилами и описанием. Сервисы могут выполнять вычисления, осуществлять управление устройствами, предоставлять доступ к данным, публиковать данные, запускать выполнение цепочек других сервисов. Разновидностью сервисов являются веб-сервисы, доступ к которым осуществляется через сеть Интернет с использованием протокола HTTP(S). Такой подход к реализации программного продукта в виде совокупности веб-сервисов получил широкое распространение и имеет очевидные преимущества: веб-сервисы кроссплатформенны, легко тестируются, доступны из любой точки сети Интернет, доступ к сервисам, как правило, стандартизирован [1]. Использование сервисов позволяет обеспечивать логическое разделение приложения на модули, которые можно реализовывать разными языками программирования, на различном аппаратном обеспечении (серверах), с использованием инструментов взаимодействия, мониторинга и хранения данных.

В 1960-х годах началось активное развитие многопроцессорных систем и параллельных вычислений, а также совершенствование аппаратного обеспечения (мэйнфреймы, суперкомпьютеры). В этот момент появилась необходимость передачи большого количества данных между вычислительными узлами, что обусловило развитие коммуникационных технологий. Первоначально передача данных производилась в пределах локальных сетей организаций, но уже в 1970-1980-х годах отдельные взаимодействия между локальными сетями организаций перерастают в глобальную сеть Интернет, появляются распределенные вычислительные системы (РВС), архитектуры параллельных вычислений, интерфейсы передачи сообщений между участниками РВС. В 1990-х годах

появляются GRID-технологии, которые реализуют централизованное предоставление ресурсов для решения различного рода вычислительных задач. GRID – это система, координирующая распределенные ресурсы посредством стандартных, открытых, универсальных протоколов и интерфейсов для обеспечения нетривиального качества обслуживания [2, 3].

В 2006 году появляются облачные вычисления (англ. cloud computing) – модель обеспечения удобного сетевого доступа по требованию к некоторому множеству конфигурируемых вычислительных ресурсов. Облачные вычисления характеризуются масштабируемостью и виртуализацией. Виртуализация позволяет абстрагировать и унифицировать предоставляемые программные и аппаратные ресурсы для конечных пользователей. За счет виртуализации упрощается развертывание специализированного программного обеспечения, что позволяет автоматически масштабировать его на необходимом для решения задач ресурсах за приемлемое время, а также перераспределять нагрузку на свободные ресурсы. Развитие облачных вычислений привело к появлению возможности оперативного предоставления вычислительных ресурсов и обеспечения доступа для развертывания и масштабирования сервисов. В это же время началось создание большого количества стандартов интерфейсов сервисов, в том числе ориентированных на работу в сети Интернет (веб-сервисов). Пакеты прикладных программ, обычно выполняемых в рамках одного вычислительного узла, стало возможным запускать на наборе узлов, разнесенных географически, принадлежащих разным организациям и соединенных сетью Интернет.

Большой вклад в разработку методов и теоретических основ организации распределенных вычислений и выполнения пакетов прикладных программ внесли Foster I., Lamport L., Lynch N., Hansen P.B., Бычков И.В., Горбунов-Посадов М.М., Самарский А.А., Матросов В.М., Бахманн П., Опарин Г.А., Феоктистов А.Г. и др. Таким образом, отдельные вычислительные модули стало возможным размещать на распределенных вычислительных узлах в виде веб-сервисов и использовать их для решения сложных задач в рамках РВС.

Сервис-ориентированный подход обладает рядом преимуществ:

- сервисы могут быть повторно использованы, что уменьшает стоимость разработки новых систем в силу наличия уже готовых модулей;

- использование сервисов предполагает организацию обмена сообщений между ними, что повышает мобильность и взаимозаменяемость сервисов;
- на основе контроля и анализа процесса обмена сообщениями можно обнаруживать атаки, обеспечивать необходимый уровень безопасности при передаче сообщений, осуществлять преобразование сообщений, осуществлять балансировку нагрузки на вычислительные узлы, на которых развернуты сервисы.

В настоящее время сервис-ориентированная архитектура часто используется для построения сложных многофункциональных систем, таких как системы международной онлайн торговли, системы банковского взаимодействия, системы предоставления государственных сервисов. Например «Система межведомственного электронного взаимодействия» (СМЭВ) [4], которая упрощает и централизует процесс получения государственных услуг населением. При этом каждый участник СМЭВ предоставляет свой набор стандартизированных сервисов, которые используют заранее оговоренные структуры данных и способы передачи сообщений.

Протоколы взаимодействия сервисов

Как правило, большие сервис-ориентированные системы характеризуются тем, что их внутренние сервисы взаимодействуют на основе единого набора протоколов. OASIS (Organization for the Advancement of Structured Information Standards) [5] является некоммерческой организацией, разрабатывающей и внедряющей протоколы для сервис-ориентированных сред (Unified Business Language, UBL), электронной торговли, документооборота и т. д. OASIS также разработал известные стандарты UDDI (Universal Description Discovery and Integration), WS-BPEL (Web-Services Business Process Execution Language), WSS (Web Service Security).

Чаще всего сервисы инкапсулируют какие-либо вычислительные алгоритмы. Пользователь не знает внутреннюю логику работы сервиса, ориентируется только на его описание и взаимодействует с сервисом только через его стандартизированный интерфейс. Сервисы различаются по длительности, сложности, требованиям к программным и аппаратным ресурсам и т. д. Сервисы, вне зависимости от входных параметров на любой корректный или некорректный входной параметр должны

возвращать результат работы или сообщение с указанием ошибки в рамках стандарта интерфейса.

Говоря о сервис-ориентированной архитектуре, необходимо рассмотреть несколько аспектов работы сервисов – форматы представления данных, способы передачи данных между сервисами, способы описания сервисов, а также способы отображения доступных сервисов.

Существуют два основных формата представления данных в сервис-ориентированной архитектуре – XML (eXtensible Markup Language) и JSON (JavaScript Object Notation). XML характеризуется большей сложностью синтаксиса и размером собственных данных формата, нежели JSON, но предоставляет большие возможности для задания отображения данных с помощью стандарта XSLT, а также поддерживает пространства имен [6]. Оба формата позволяют производить проверку сообщений с помощью стандартов XML Schema и JSON Schema, соответственно. Третьим вариантом передачи данных является передача бинарных данных (текстовая информация, сгенерированное сервисом изображение).

Между сервисами передача данных осуществляется путем включения набора данных в тело сообщения или передачи URL (Uniform Resource Locator) файла данных. Передача данных внутри сообщений, формат которых определяется стандартом интерфейса сервиса, обычно реализовывается для данных малого объема (строковые, числовые данные, URL-адреса). Если же сервис в качестве результата возвращает данные большого объема, например, набор изображений высокого разрешения, то обычно передается URL данных, не включая непосредственный массив данных в тело сообщения. Передача данных больших объемов посредством их URL-адресов позволяет уменьшить нагрузку на сеть (данные запрашиваются принимающей стороной по мере необходимости), отдельный URL-адрес файла позволяет загружать файл несколько раз, не совершая запросов к самому сервису.

Протоколом удаленного вызова программных методов является XML-RPC (XML Remote Procedure Call), который использует XML-файлы для удаленного вызова сервисов и передачи данных. Основной идеей данного протокола является представление передаваемых структур данных в виде XML-документов, содержащих как строковые и числовые параметры, так и массивы и другие

структуры данных. Основным преимуществом XML-RPC является его понятность и легкость в работе.

Стандарт SOAP (Simple Object-Access Protocol) [7] развил идею инкапсуляции запросов к удаленным сервисам с использованием XML. SOAP использует стандарт WSDL, что делает его интеграцию и использование удобным и регламентированным. SOAP может использоваться как с HTTP(S), так и с использованием альтернативных способов передачи данных, например, через электронную почту. SOAP обеспечивает высокую степень защищенности сервисов, он может использоваться как с базовой HTTP(S) авторизацией, так и со стандартами WS-Security [9]. SOAP-сообщение обычно состоит из четырех частей: конверта, определяющего начало и конец сообщения, заголовка, содержащего опциональные атрибуты, тела и информации о возможных ошибках. В то же время большое количество возможностей, а также специфика построения самих SOAP-документов делают стандарт менее удобным в работе, чем XML-RPC.

В отличие от стандартизированных способов задания интерфейсов сервисов, таких как XML-RPC или SOAP, в последнее время все чаще используется стиль построения архитектуры веб-сервисов REST (REpresentational State Transfer) [10]. Основными отличиями данного стиля являются:

- клиент-серверная архитектура (единый интерфейс между клиентом и сервером);
- отсутствие состояния (вызов сервиса зависит только от передаваемых параметров, не от предыдущих вызовов);
- единообразие интерфейса (идентификация ресурсов посредством уникальных URL-адресов, взаимодействие с ресурсами через представление).

Часто взаимодействие с ресурсами, предоставляемыми через REST-интерфейс, осуществляется с использованием определенных в стандарте HTTP команд GET, POST, DELETE и т. д., что упрощает понимание и использование такого рода сервисов. Дополнительно в процессе обмена сообщениями между клиентом и сервисом используются HTTP-коды, определяющие статус ответа.

Для организации работы с веб-сервисами, ориентированными на обработку пространственных данных, часто используется XML-стандарт WPS (Web Processing Service), разработанный консорциумом OGC [10]. Стандарт WPS охватывает как

аспекты обнаружения и описания сервиса, так и непосредственный вызов и контроль выполнения сервисов. В стандарте имеется три вида запросов: GetCapabilities, DescribeProcess, Execute. Запрос GetCapabilities возвращает описание WPS-службы (программной системы, организующей и координирующей процесс выполнения различных зарегистрированных в ней сервисов) и список доступных для выполнения сервисов. На основе идентификаторов сервисов, представленных в ответе на запрос GetCapabilities, в ответ на запрос DescribeProcess возвращается описание конкретного сервиса, включающее в себя название, описание, идентификаторы и типы входных и выходных параметров сервиса. На основании информации о конкретном сервисе совершается запрос Execute, в ответ на который клиенту может прийти**:**

1) сырые данные, произведенные сервисом (текст, изображение, видеофайл);

2) XML-документ с результатами выполнения (данные встраиваются в документ в явном виде);

3) XML-документ со ссылками на результаты работы сервиса (клиент получает набор URL-адресов, ведущих на сохраненные результаты работы сервиса, которые будут доступны неограниченное время после завершения работы удаленного сервиса);

4) XML-документ со статусом выполнения сервиса – стандарт WPS разрешает сколь угодно долгое выполнение сервисов. В случае, если сервис выполняется длительное время, в документе со статусом выполнения сервиса указывается процент выполнения сервиса, а также адрес XML-файла, в который будут помещены результаты выполнения сервиса по мере завершения его работы.

Обычно WPS-сервис представляет собой атомарную функцию, выполняющую какие-либо вычисления. Задание взаимодействия между WPS-сервисами позволяет организовывать повторяющиеся рабочие процессы. WPS-сервис может самостоятельно вызвать набор других WPS-сервисов или принимать на вход список сервисов, которые необходимо выполнить в качестве параметров Execute запроса.

Метаданные сервисов

Среди основных способов описания сервисов можно выделить стандарт WSDL (Web Service Description Language). Стандарт WSDL описывает, как получить доступ к сервису и какие операции он может выполнять. WSDL изначально использовался для документирования сервисов, использующих SOAP в качестве формата обмена данными [11]. WSDL состоит из четырех основных элементов:

1) типы данных, используемых в работе сервиса (структура данных задается в соответствии со стандартом XML Schema, с помощью которого в дальнейшем можно производить проверку передаваемых данных);

2) список сообщений используемых сервисов. Каждое сообщение может содержать несколько частей по аналогии с параметрами вызываемых функций в языках программирования;

3) методы, предоставляемые сервисом;

4) форматы сообщений и детали протоколов для каждого метода, предоставляемого сервисом.

Для проверки корректности передаваемых данных используется стандарт XML Schema. Последняя версия WSDL 2.0 делает возможным описание REST-сервисов в рамках WSDL.

Сложность и перегруженность стандарта WSDL послужила толчком для создания более простого стандарта WADL (Web Application Description Language), специализирующегося на описании ресурсов, предоставляемых каким-либо сервисом, а также на описании отношений, установленных между ними. В какой-то степени WADL играет для REST-сервисов ту же роль, что WSDL играет для сервисов, работающих на основе SOAP. WADL менее распространен, чем WSDL, тем более что последние версии стандарта WSDL также поддерживают описание REST-сервисов.

Появление единой схемы семантической разметки [12], созданной в 2011 году компаниями Google, Microsoft и Yahoo, позволило встроить метаданные в пользовательское описание сервисов [13]. Основной целью Schema.org является помощь в создании метаданных для улучшения качества работы поисковиков. В контексте поиска сервисов эта схема семантической разметки рассматривается как универсальная онтология, применяемая для описания сервисов в виде, пригодном

для машинной обработки. Указанная онтология применяется, например, в технологии JSON-LD («JavaScript Object Notation for Linked Data» — объектная нотация JavaScript для связанных данных) [14], которая позволяет передавать связанные данные в удобном для программной обработки виде. JSON-LD использует понятие контекста, который связывает свойства объектов в JSON-документе с элементами онтологии.

Применение онтологий для описания сервисов

Учитывая большое количество сервисов и разнообразие выполняемых ими операций при построении композиций, необходима автоматизация обработки метаданных. Автоматическая обработка метаданных требует определения концептов, используемых в предметной области, и установления семантических взаимосвязей между классами и ограничений на эти связи, а также алгоритмические элементы, отражающие процессы обработки данных. Автоматическая обработка метаданных выполняется в рамках онтологического моделирования, которое нашло широкое применение в рамках технологий программирования для семантической паутины [15]. Для онтологического моделирования геообработки применяется упорядоченный набор онтологий:

1) общая онтология;
2) онтология геопространственной области;
3) онтология пространственных данных;
4) онтология процессов геообработки.

Общая онтология является основным верхним уровнем словаря для описания общих понятий, не зависящих от домена. Это общий язык, который используют все другие онтологии. Для описания понятий используется язык OWL (Web Ontology Language) [16], разработанный консорциумом W3C. Язык OWL позволяет описывать классы и отношения между ними. В основе языка – представление действительности в модели данных «объект – свойство». Каждому элементу описания в этом языке (в том числе свойствам, связывающим объекты) ставится в соответствие URI. В качестве основы для определения понятий верхнего уровня и утверждения об этих понятиях используется словарь Dublin Core [17]. Одним из

важных отношений OWL для составления композиций является частное – общее. Так язык OWL позволяет определить иерархию классов, включая суперклассы, под которыми объединяются подклассы, что позволяет применять правило ко всем классам. Информация об иерархии классов при поиске композиций сервисов может быть полезна для анализа соответствия данных. Иерархия классов собирается в файле OWL из элементов subClassOf, например:

<owl:Class rdf:ID="A-Road">

<rdfs:subClassOf>

<owl:Class rdf:ID="Roadway"/>

</rdfs:subClassOf>

</owl:Class>

Класс A-Road является дочерним по отношению к классу Roadway. Атрибуты классов онтологии должны быть определены в предоставленном файле OWL со ссылкой на выбранный класс или один из его суперклассов.

<owl:DatatypeProperty rdf:ID="Length">

<rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#float"/>

<rdfs:domain rdf:resource="#Roadway"/>

</owl:DatatypeProperty>

rdf:ID определяет имя атрибута. rdfs:range указывает тип атрибута с использованием типа схемы XML. rdfs:domain указывает класс, в котором определен атрибут.

Онтология геопространственной области обеспечивает семантическое описание верхнего уровня наук о земле как отправная точка для организации знаний о геообработке. В статье [18] предлагается общая онтология SWEET, концепты которой включены в каталог Global Change Master Directory (GCMD) [19], содержащий более 28 000 описаний наборов данных, известных как DIF. Формат каталога совместим со стандартом Федерального комитета географических данных

и международным стандартом ISO 19115 [20]. Также концепты SWEET используются в тезаурусе типов объектов Александрийской цифровой библиотеки и программном обеспечении Earth Science Modeling Framework [21] с открытым исходным кодом для моделирования климата, прогнозирования погоды и других приложений для наук о Земле.

Онтология геообработки охватывает знания о:

1) пространственно-временные факторы, т. е. пространство, время и единицы измерения;
2) физические факты, напр. физические явления, физические свойства и физические вещества;
3) дисциплины, например, научные области и проекты;
4) платформы данных, например, приборы и датчики.

Онтология геопространственных данных задает концепты и отношения между ними и концептами типов данных «DataType» для пространственных данных. Она напрямую связывает данные с научными дисциплинами посредством включения онтологии геопространственной области. Существуют широко используемые стандарты метаданных, такие как ISO 19115 [20], FGCDC [22]. Онтология геопространственных данных также добавляет семантику в метаданные, что позволяет пользователю находить данные, не зная точных ключевых слов метаданных, поскольку термины в запросе имеют эквивалентное определение в онтологии геопространственной области. Чтобы обеспечить унифицированное представление метаданных, определяются семантические отношения между концептами в разных стандартах метаданных, например, между «disjoint» и «sameAs». Таким образом, не существует четкой границы между различными стандартами метаданных. Пользователь может использовать любой термин из любого стандарта метаданных для запроса данных, описанных в любом из поддерживаемых онтологией стандартов метаданных.

В статье [15] разработана онтология для описания сервисов геообработки данных на основе OWL-S [23]. Эта онтология классифицирует сервисы

геообработки, их внутреннюю структуру и документирует отношения и ограничения среди них, путем включения следующих концептов:

- научная дисциплина: область, в которой может быть применен сервис геообработки, например, солнечное излучение или поверхность Земли. Определения берутся из онтологии геопространственной области;
- методология/алгоритм: методология или алгоритм, используемый в анализе данных сервиса геообработки, например, алгоритмы ISODATA и MINDISTANCE;
- входные данные: тип данных определяется в онтологии геопространственных данных;
- выходные данные: тип и свойства выходных данных, например, время работы и точность. Тип данных из онтологии геопространственных данных.

Open Geospatial Consortium активно разрабатывает стандарты Semantic Sensor Web (Семантическая сеть сенсоров) для работы с данными дистанционного зондирования Земли [24]. Семантическая сеть сенсоров — это набор технологий Семантической паутины (Semantic web), адаптированный для работы с данными, поставляемыми разнородными сенсорами. Количество сенсоров растет и требует автоматизации поиска и использования. Доступ ко всем возможностям разрабатываемой семантической сети сенсоров определяется наличием возможности работать с онтологиями, в частности «Semantic Sensor Network Ontology» [25]. Применение этих стандартов и технологий является перспективным из-за активного использования данных сенсоров (в том числе дистанционного зондирования Земли (ДЗЗ)) в разрабатываемой среде.

Любой сервис определяется его входами, выходами (параметры) и выполняемой операцией. Все входы, выходы и операция могут быть описаны с помощью метаданных и онтологий. Тем не менее для формирования композиций требуется учитывать множество факторов:

- форматы данных параметров;
- схемы используемых данных;
- используемые единицы измерения;

- качество и полнота данных;
- для пространственных данных – используемая пространственная проекция, пространственный охват;
- период сбора данных и т. д.

Следовательно, полное описание сервиса является достаточно сложной задачей, которая полностью основывается на существующих онтологиях. Знания, определенные онтологиями, могут не содержать необходимой информации. Следовательно, полное описание сервиса, которое позволит создавать композиции, не всегда возможно.

Сервисы предоставления и редактирования данных

Сервисы обработки данных могут быть реализованы на разных языках программирования, работать на разном аппаратном обеспечении и т. д., что предполагает возможное различие в хранении данных. Поэтому передача данных между сервисами должна проводиться в соответствии со стандартами. В работе рассматривается стандарт OGC WPS, в котором имеется поддержка основных типов данных и пространственных данных. Рассмотрим основные типы данных, передача которых регламентирована в рамках стандарта (рисунок 1.1).

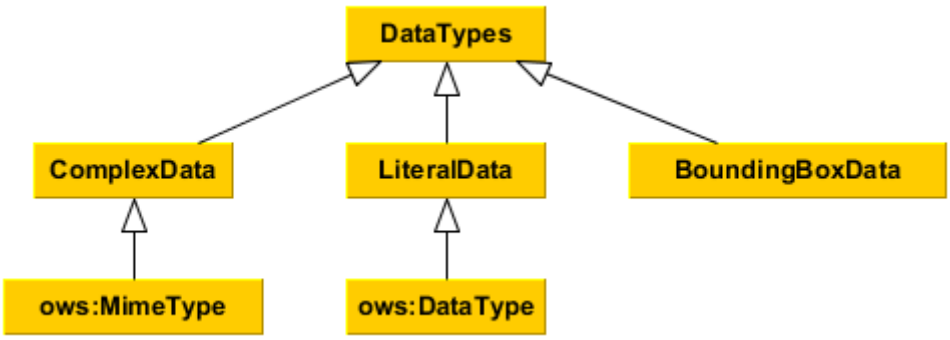


Рисунок 1.1 – Типы данных WPS сервисов

LiteralData – тип данных, передаваемый в виде строки, тип значения специфицируется с помощью ows:DataType (например, строка, число, дата и т. д.).

ComplexData – тип данных для передачи и получения структуры данных, такие, как GML или двоичные данные. Сложные данные могут быть встроены в

запрос/ответ или находиться на удаленном ресурсе (в качестве ссылки). Структура ComplexData состоит из иерархии структур данных. Структура данных верхнего уровня описывает передаваемые данные и включает:

- MimeType – идентифицирует формат в соответствии с ows:MimeType;
- кодировка – необязательная ссылка (URI) на кодировку;
- схема – необязательная ссылка (URI) на документ схемы XML.

BoundingBoxData – тип данных, с помощью которого можно передать экстент области обработки.

В большинстве случаев сбор данных осуществляется в реляционном виде из-за удобной структуры данных для анализа и отображения.

Автоматизация всех этапов на основе SOA значительно ускоряет решение задач. Однако предоставление данных (т. е. поиск, выбор, сбор, передача данных, определение схемы) в основном проводится с участием человека, что значительно замедляет решение задач. В связи с этим выдвигаются новые требования к предоставлению данных:

1) постоянный программный доступ к данным в сети Интернет с регламентацией доступа;
2) предоставление метаданных, описывающих точку доступа, схему данных, единицы измерения и т. д. Эта информация необходима для поиска сервисов и определения соответствия между данными и входными параметрами сервисов.

Кроме того, требуется реализация всего жизненного цикла данных, включая:

3) возможность создания новых данных и соответствующих сервисов как пользователями, так и программными системами. Промежуточные и конечные результаты работы сервисов часто требуется сохранять, что может потребовать создания сервисов. Требуется также удаление данных и сервисов. Создание новых сервисов должно основываться на использовании схемы данных. Схемы данных для многих задач предметных областей являются устоявшимися. Сервисы обработки принимают на вход данные с определенной схемой данных.

Результатом их работы являются данные с другой схемой. Для формирования композиций сервисов является важным соответствие схем данных входных и выходных параметров сервисов. Сервисы предоставления данных должны создаваться в соответствии с заранее определенными схемами данных;

4) реализацию редактирования данных пользователями;
5) публичный API для программной модификации данных;
6) поддержку пространственных данных.

Часто данные разных исследователей могут быть об одном и том же объекте исследования и обладают общим набором характеристик, но в тоже время имеется своя специфика и отличие в виде дополнительных полей, которые необходимы для реализации целей конкретного исследователя. Хотя по структуре данные исследователей могут отличаться, тем не менее они содержат общую часть, которая может быть проанализирована одними и теми же методами. В терминах объектно-ориентированного программирования (ООП) эти данные имеют общего родителя. Поэтому реализация механизмов наследования для сервисов предоставления данных является также желательной.

7) поддержку механизмов наследования.

Рассмотрим соответствие сформулированным требованиям существующего программного обеспечения. На сегодняшний день активно создаются информационные системы, которые производят сбор данных разных предметных областей в Интернет. Как наиболее успешные проекты, можно отметить: онлайн-энциклопедию Википедия [26]; орнитологический ресурс eBird [27]; свободную карту мира OpenStreetMap [28]. Широкую известность среди ботаников России получил использующий краудсорсинг открытый атлас и онлайн-определитель растений России и сопредельных стран Плантариум [29]. Ныне действует ряд глобальных информационных ресурсов, ориентированных на поддержку научных исследований биоразнообразия: Global Biodiversity Information Facility [30], Species2000 [31], Catalogue of Life [32]; фиторазнообразия: The Plant List [33], Tropicos® [34] и др. В России существуют информационные системы особо охраняемых природных территорий и объектов России (ООПТ) [35, 36],

Зоологического института РАН [37] и др. Сибирь охватывает региональный проект Электронный атлас Биоразнообразие животного и растительного мира Сибири [38], Байкальскую Сибирь – БД «Флора Байкальской Сибири» [39]. Система DINA [40] разработана как слабосвязанный набор нескольких веб-модулей. Концептуальной основой этой модульной экосистемы является набор программных интерфейсов (API) для программирования веб-приложений, гарантирующих интероперабельность ее компонентов.

Для получения метеорологических данных посредством API можно воспользоваться следующими сервисами:

- Метеосервис [41];
- OpenWeatherMap [42];
- World Weather API and Weather Forecast [43];
- AccuWeather [44].

Пользователи часто используют универсальные системы, позволяющие создавать реляционные данные:

- Rows [45] – система сбора данных в виде онлайн таблиц;
- Google Таблицы [46] – сервис, который позволяет создавать, обновлять и редактировать таблицы и делиться данными онлайн;
- NextGIS [47] – сервис, позволяющий создать пространственные данные в Интернет, реализует Веб ГИС.

Таблица 1.1. Сравнение программного обеспечения сбора данных

ПО	Метаданные	Редактирование пользователями	Наследование	Публичный API CRUD	Создание сервисов
Википедия	+	+	+	+	-
eBird	+	+	-	+	-
OpenStreetMap	+	+	-	+	-
Плантариум	-	+	-	-	-

GBIF	+	-	-	-	-
Species2000	+	+	-	+	-
ИС Зоологического института	-	-	-	-	-
ИС ООПТ России	-	+	-	-	-
ИС Биоразнообразие животного и растительного мира Сибири»	+	+	-	+	-
БД «Флора Байкальской Сибири»	-	-	-	представление данных	-
meteoservice.ru	-	-	-	представление данных	-
openweathermap.org	-	-	-	представление данных	-
worldweatheronline.com	-	-	-	представление данных	-
accuweather.org	-	-	-	представление данных	-
Rows	+	+	-	-	-
Google Таблицы	-	+	-	-	-
NextGIS	+	+	-	+	-
Geoserver	+	+	-	+	-
Dina	+	+	-	+	-

В таблице 1.1 рассмотрено программное обеспечение, активно используемое исследователями для сбора научных данных в реляционном виде. Отметим, что ни одно из них не удовлетворяет всем сформулированным требованиям.

Сервисы обработки данных

На сегодняшний день выделим тенденцию роста количества сервисов обработки данных. Активно создаются сервисы анализа данных дистанционных методов земли, обработки пространственных данных, анализа текстов и т. д. Для разработчиков существует несколько платформ создания WPS сервисов, которые предоставляют общие и соответствующие стандартам методы использования существующих библиотек и алгоритмов в качестве WPS сервисов. Среди них выделим ZOO-Project [48] для создания сервисов на основе динамически подгружаемых библиотек, 52°North [49] для сервисов на языке Java, PyWPS [50] для создания сервисов на языке Python, ArcGIS Server [51] также предоставляет возможность публиковать их методы в виде WPS, Geoserver [52], Tethys [53]. Все эти платформы позволяют создавать новые сервисы и публикуют алгоритмы с существующих ГИС и библиотек, фреймворков таких как GRASS [54], ArcGIS Server, Sextante [55], WPS4R [56], GDAL [57]. На основе ZOO-Project реализовано более 500 различных сервисов.

Многие из сервисов выполняют аналогичные функции. Например, сервисы построения кратчайших маршрутов предоставляют Google, Yandex, OSM и т. д. Отличаются сервисы качеством результатов, стоимостью услуг и т. д. Отметим, что многие исследователи в своей работе используют сервисы обработки данных, отображения данных, в том числе в виде карт. Обмен опытом использования сервисов (решаемые задачи, значения параметров и т. д.) был бы очень полезен исследователям.

1.2. Анализ подходов к созданию композиций сервисов

Обнаружение сервисов

При большом количестве сервисов, используемых для решения различных задач, возникает проблема обнаружения и повторного их использования. Существует два подхода к организации обнаружения сервисов – централизованный и децентрализованный [58].

Одним из известных централизованных методов реализации обнаружения сервисов является использование стандарта UDDI (Universal Description Discovery

and Integration), который является XML-стандартом и определяет формат предоставления информации о зарегистрированных сервисах. При использовании UDDI реестров сервисов пользователь должен самостоятельно производить поиск требуемого сервиса в списке зарегистрированных на основании какого-либо критерия. Изначально UDDI разрабатывался для организации глобального хранилища сервисов, в котором различные научные и коммерческие организации могли бы регистрировать и предоставлять свои сервисы и одновременно использовать сервисы, зарегистрированные другими участниками. Структура реестра UDDI дает возможность реализовать полный цикл управления сервисами «публикация – поиск – связывание», ориентированный прежде всего на автоматическую динамическую реконфигурацию вычислительного процесса вследствие изменяющейся обстановки. На основе информации реестра возможно автоматически поддерживать заданный уровень качества функционирования за счет автоматического выбора рабочей конфигурации при отказе (или невозможности использования) отдельных ее сервисов. К сожалению, данная идея не нашла широкой поддержки, и крупные участники рынка постепенно стали предлагать собственные форматы организации реестров сервисов, более приспособленных к их потребностям.

Среди децентрализованных методов обнаружения сервисов одним из популярных является стандарт WS-discovery [59], разработанный совместными усилиями компаний Microsoft, IBM и т. д. и стандартизированный консорциумом OASIS. Основным отличием WS-discovery от UDDI является то, что UDDI ориентирован на создание глобального, межкорпоративного и межгосударственного индекса сервисов, а WS-discovery нацелен на обнаружение сервисов в рамках локальной сети организации. WS-discovery предполагает рассылку сообщений всем участникам сети. Сообщения содержат координаты сервиса и краткую информацию о его предназначении и параметрах. Децентрализованный метод обнаружения сервисов получил широкое распространение, например, в операционных системах семейства Windows.

Композиция сервисов

Часто решение задачи может требовать многократного вызова последовательности (цепочки) сервисов, при этом пользователю приходится придерживаться определенного порядка вызова сервисов, вводить одни и те же значения параметров для каждого сервиса и т. д. В этом случае для автоматизации работы пользователя требуется композиция сервисов [60], задающая вычислительный процесс. Композиция сервисов состоит из других (атомарных или составных) сервисов, взаимодействие между которыми производится с помощью управляющих конструкций, таких как последовательность, ветвление и циклы.

Выделяются два типа задания композиций сервисов: абстрактный и конкретный. Вычисления в рамках конкретного типа должны быть проведены на предопределенных ресурсах. Вычисления, описываемые абстрактным типом, абстрагированы от конкретных ресурсов. Они могут выполняться с помощью различных программных приложений, реализующих одинаковые функции, и на любых ресурсах, подходящих по своим характеристикам требованиям. Учитывая динамичный характер распределенных вычислительных сред, предпочтительно использовать абстрактный тип.

Обычно композиции сервисов задаются с помощью языков для формального описания бизнес-процессов, например BPEL (Business Process Execution Language) [61], BPMN (Business Process Modeling Notation) [62], BPML (Business Process Management Language) [63] и XPDL (XML Process Definition Language) [64]. Также для задания композиций сервисов активно используются популярные языки программирования, например, Python, JavaScript. Существует ряд систем workflow («поток задач»), позволяющих формировать композиции сервисов в различных предметных областях: Pegasus [65], Kepler [66], Swift [67], KNIME [68], Taverna[69], Galaxy[70], Trident [71] and Triana [72], Everest (Mathcloud) [73], CLAVIRE [74]. Для обработки пространственных данных активно используется Geo-Processing Workflows [75]. Общепризнанной формализацией композиции сервисов является направленный ациклический граф (Directed acyclic graph, DAG) [76, 77], в котором вершинами являются вызовы сервисов, а дугами — зависимости между сервисами по данным. Выделяются два типа рабочих процессов: абстрактный и конкретный.

Вычисления в рамках конкретного рабочего процесса должны быть проведены на предопределенных ресурсах. Вычисления, описываемые абстрактным рабочим процессом, абстрагированы от конкретных ресурсов. Они могут выполняться с помощью различных программных приложений, реализующих одинаковые функции, и на любых ресурсах, подходящих по своим характеристикам требованиям. Учитывая динамичный характер распределенных вычислительных сред, предпочтительно использовать абстрактные рабочие процессы.

Создание композиции сервисов – это сложный процесс, требующий навыков программирования от пользователя, поиска подходящих сервисов среди их большого количества, знания языков задания композиций сервисов, больших временных издержек на разработку и отладку.

Одна важнейших особенностей создания композиции сервисов – это спецификация того, как результаты конкретного сервиса могут быть приняты входами других сервисов, т. е. сопоставление данных. Композиция сервисов обычно выполняется как обратный вывод, т. е. от целевого состояния к исходному состоянию. Существует несколько подходов к автоматизации создания композиций сервисов [78].

1. Интерактивное построение композиций сервисов на основе знаний. В этом подходе пользователь вручную строит композицию сервисов, а система вывода и знания (метаописания, онтологии) используется для поиска подходящих сервисов и значений параметров [79, 80, 81].

2. Автоматическое построение композиций сервисов. В этом подходе производится (полу) автоматическое обнаружение и составление необходимых сервисов на основе семантического анализа и логического вывода [82, 83, 84].

3. Автоматическая композиция Web-сервисов с учетом планирования выполнения. В этом подходе [85, 86] входные данные представляются сервисами, которые семантически аннотированы с использованием онтологий, например Web Ontology Language for Services (OWL-S) [23, 87], сервисы обработки аннотированы с помощью Web Service Modeling Ontology (WSMO) [88]. Выбор среди

сгенерированных цепочек сервисов может проводиться на основе оценки качества обслуживания (QoS) с использованием алгоритмов, таких как генетические алгоритмы или алгоритмы теории игр [89, 90].

Для поиска подходящих сервисов и значений параметров в статье [91] используется Пролог. Онтологическая информация, записанная на OWL или OWL-S, преобразуется в тройки RDF и загружается в базу знаний. Машина вывода имеет встроенные аксиомы и правила вывода OWL. Эти аксиомы и правила применяются к фактам в базе знаний, чтобы найти все соответствующие следствия, такие как отношения наследования между классами, которые могут быть не явно заданы.

В статье [92] предлагается подход, в котором для поиска подходящих WPS сервисов используется язык декларативного логического программирования Datalog. Формализована база знаний, включающая типы пространственных данных, выполняемые операции, ограничения на проекцию и т. д. Логический вывод запроса Datalog производит поиск сервиса, удовлетворяющий множеству ограничений.

В статье [93] для поиска подходящей композиции сервисов предлагается использовать SPARQL. Предварительно композиции сервисов описываются с помощью онтологической модели. Далее с помощью запросов формируются дополнительные знания о типах входных и выходных данных.

Методы автоматической композиции сервисов поддерживают автоматическое обнаружение, выбор и привязку составных сервисов с минимальным вмешательством человека [94, 95]. Построение плана и автоматическая композиция сервисов была активной областью для автоматической компоновки веб-сервисов [96]. Наиболее известным алгоритмом в этом подходе является GraphPlan [97]. GraphPlan динамически строит цепочки услуг на основе семантических параметров, соответствующих отношениям между услугами и изменениям значений QoS [98, 99, 100]. В ранних исследованиях композиции сервисов часто автоматически составлялись путем связывания их входных и выходных параметров [84, 101, 102, 103, 104, 104]. Однако эти подходы имеют существенный недостаток. Если два разных сервиса имеют одинаковые входные и выходные параметры, отличить их непросто, хотя они выполняют разные операции над данными.

В последние годы многие исследования подчеркивали важность семантической информации о сервисе при композиции сервисов геообработки, утверждая, что при составлении сервисов необходимо обнаружение сервисов с соответствующими функциями [105, 106, 107, 108]. Входы и выходы между соседними сервисами должны совпадать не только с точки зрения параметров, но также с точки зрения их синтаксиса и семантики. Онтология стала распространенным инструментом для описания семантической информации сервисов геообработки, а онтологические подходы в настоящее время являются наиболее распространенными методами автоматической компоновки сервисов геообработки.

В этом направлении создавались онтологии пространственных операций для классификации пространственных данных и сервисов, определили ряд правил связывания пространственных операций [109]. Проводилась классификация сервисов геообработки с помощью созданных онтологий. Для поиска и сопоставления сервисов использовались семантические веб-технологии [110]. В статье [111, 112] предложили концепцию онтологии типов данных и построение на ее основе композиции сервисов [113, 114]. Также были предприняты некоторые усилия по автоматизации составления сервисов геообработки с помощью методов глубокого обучения [115, 116].

Следует отметить, что автоматическое построение композиций сервисов на основе семантического анализа, логического вывода и планирования является сложной задачей по следующим причинам:

- большая часть сервисов имеет недостаточно метаинформации, либо она отсутствует или не формализована;
- требуется проработанная онтология предметной области, ее построение не тривиальный процесс, в котором нужно рассмотреть все возможные связи между сервисами. Достаточно сложно выполнить описание некоторых сервисов. В статье [83] приводится пример, в котором сервис, выполняющего деление, может возвращать плотность, если его входные данные представляют собой массу и объем, или скорость, если задано расстояние и период времени;

- требуется согласованность метаинформации сервисов и онтологии предметной области.

Обычно при решении конкретной задачи пользователь составляет некоторую последовательность выполнения сервисов на основе их неформализованных описаний и примеров решений подобных задач. Одна и та же последовательность выполнения сервисов может повторяться многократно. Автоматизация выполнения последовательности сервисов позволит упростить и ускорить работу пользователя. Выполняя последовательность сервисов, пользователь задает неявную информацию о связях между ними, используемых значениях параметров. Эта информация может быть полезна для автоматического составления композиций сервисов. Поэтому актуальна задача автоматизации формирования композиций сервисов на основе статистических данных о применении сервисов пользователями.

Оценка качества сервисов

Качество обслуживания (Quality of Service, QoS) является критически важным фактором при выборе Web-сервисов для сложных бизнес-процессов [117]. Поэтому аспекты надежности, доступности и другие аспекты QoS фиксируются в стандартах, в частности, в стандарте Business Process Execution Language for Web-Services (BPEL4WS). Необходимость составлять спецификации QoS для Web-сервисов объясняется двумя обстоятельствами. С одной стороны, клиентам нужно качественное обслуживание, а с другой – поставщики услуг стремятся к достижению оптимального баланса между удовлетворенностью пользователей и уровнем загрузки системы. Например, наиболее важные транзакции, такие как обработка банковского платежа, должны выполняться с высоким приоритетом. Показатели QoS для Web-сервисов можно разделить на пять категорий: быстродействие, ресурсоемкость, функциональная надежность, свойства транзакционности, безопасность [118]. Эти качества реализуются на различных уровнях в приложении: уровне экземпляра класса сервиса, уровне класса сервиса и системном уровне. При формировании композиций сервисов показатели QoS используются в качестве ограничений при выборе сервисов.

1.3. Выполнение сервисов

Функционирование многопользовательской СОА среды требует обеспечения эффективного и надежного выполнения сервисов на множестве ограниченных ресурсов. Композиции сервисов задают упорядоченность и связанность их выполнения, которые необходимо учитывать при планировании. Рассмотрим существующие алгоритмы планирования.

Эвристические алгоритмы нахождения расписания

Эвристические алгоритмы нахождения расписания можно разбить на два класса: статические и динамические алгоритмы. Статические алгоритмы предполагают, что расписание выполнения строится перед непосредственным выполнением скрипта, т. е. все данные о времени выполнения заданий, зависимостях между заданиями, а также временных затратах на передачу данных между вычислительными узлами известны заранее. Динамические алгоритмы предполагают назначение задания на вычислительные узлы по мере их появления, решения планировщика принимаются по мере выполнения композиции.

В настоящее время существует большое количество эвристик нахождения расписания выполнения композиций, задаваемых в виде DAG. Обычно эвристики используются в списковых алгоритмах. Списковые алгоритмы предполагают сортировку списка заданий на основе определенной величины, рассчитываемой с помощью эвристики. Назначение заданий происходит в том порядке, в каком задания расположены в списке. Если несколько заданий в списке имеют одинаковое значение эвристики, в таком случае выбор назначаемого задания происходит случайным образом. Алгоритм планирования делится на этап расстановки приоритетов (рангов) для заданий и этап назначения заданий на вычислительные узлы. Одними из самых эффективных списковых алгоритмов являются алгоритмы HEFT (Heterogeneous Earliest Finish Time) [119], SDBATS (Standard Deviation Based Algorithm for Task Scheduling) [120], PEFT (Predict Earliest Finish Time) [121] и HSIP (Heterogeneous Scheduling with Improved Task Priority) [122].

Распространенным эвристическим алгоритмом является алгоритм Heterogeneous Earliest Finish Time (HEFT) [119]. Оценка времени выполнения каждого задания определяется как сумма среднего времени выполнения данного

задания на всех вычислительных узлах и наибольшего значения оценки времени работы заданий, от которых напрямую зависит данное задание, т. е. оценка определяется по формуле $rank(t_i) = \overline{w_i} + max_{t_j \in succ(t_i)}(\overline{c_{i,j}} + rank(t_j))$, где $\overline{w_i}$ – среднее время выполнения задания на всех вычислительных узлах, а $\overline{c_{i,j}}$ – средняя стоимость передачи данных между заданиями t_i и t_j для всех пар вычислительных узлов. Расчет оценки времени работы начинается с заданий, не имеющих зависящих. В случае, когда время выполнения одного задания на разных узлах сильно различается и время передачи данных между заданиями сравнительно больше, HEFT составляет расписание, используя худший вариант. Данный алгоритм на основании произведенных сравнений [123] был признан самым эффективным на данный момент из алгоритмов, не использующих специальных техник.

Алгоритм SDBATS исключает недостаток алгоритма HEFT, используя для вычисления оценки времени выполнения задания не среднее время выполнения задания на узлах, а среднеквадратичное отклонение значений времени выполнения задания на разных вычислительных узлах. Данное изменение в свою очередь приводит к составлению крайне неточных расписаний в случае, если время, необходимое для передачи данных, довольно большое. Кроме того, алгоритм SDBATS предполагает запуск входных заданий на всех доступных вычислительных узлах, что ухудшает расписание в случае, если время выполнения одного и того же задания сильно различается на разных вычислительных узлах.

Другим часто встречающимся алгоритмом является алгоритм PEFT [121]. PEFT предлагает использовать вместо рассчитанных для каждого задания приоритетов специальные таблицы OCT (Optimistic Cost Table), представляющие собой матрицы, в которых строки отображают задания, содержащиеся в рассматриваемом DAG, а столбцы – вычислительные узлы, на которых могут выполняться рассматриваемые сервисы. Каждый элемент матрицы $OCT(t_i, r_j)$ содержит максимальное из всех значений ранга для дочерних (напрямую зависимых от t_i) заданий при условии, что выполнение происходит на вычислительном узле r_j. В алгоритме на каждом этапе назначения задания на вычислительный узел производится оценка возможных временных значений для решения на некоторое

количество шагов вперед. Алгоритм PEFT составляет более эффективные расписания по сравнению с общепризнанным алгоритмом HEFT.

Алгоритм HSIP является одним из последних списковых алгоритмов, показывающих лучшие результаты в составлении расписаний как на искусственных, так и реальных задачах. HSIP акцентирует внимание на гетерогенной природе распределенной вычислительной среды и предлагает подсчет эвристики не только с учетом нормального распределения, но и с учетом затрат на передачу данных между вычислительными узлами. Алгоритм также предлагает дупликацию начальных заданий графа с учетом различающегося времени выполнения заданий на разных вычислительных узлах. В соответствии с расчетами, приведенными в статье [120], алгоритм HSIP на данный момент является самым эффективным статическим эвристическим алгоритмом для построения расписания выполнения DAG.

Метаэвристические алгоритмы нахождения расписания

Среди метаэвристических алгоритмов часто используемыми алгоритмами являются генетические алгоритмы, а также алгоритмы муравьиной колонии и имитации отжига.

Метаэвристики — это общие эвристики, позволяющие находить близкие к оптимальным решения различных задач оптимизации за приемлемое время. Метаэвристики пытаются объединить основные эвристические методы в рамках алгоритмических схем более высокого уровня, направленных на эффективное изучение пространства поиска. Метаэвристики включают две категории: метаэвристики локального поиска и эволюционные алгоритмы.

Генетические алгоритмы позволяют находить приближенные решения из широкого пространства поиска, применяя эволюционные принципы, причем они обычно находят более точные решения, нежели эвристические алгоритмы, но проигрывают по времени выполнения [124]. Генетические алгоритмы работают по принципу улучшения требуемых свойств каждого следующего получаемого поколения в соответствии с принципами природной генетики. Алгоритмы такого рода в своей работе используют набор отдельных элементов (популяция) и набор операторов, аналогичных биологическим и определенным над популяцией, используемых для манипуляций с популяцией в целях оптимизации и нахождения

приемлемого решения. В соответствии с эволюционными теориями только самые приспособленные элементы популяции выживают и производят потомство, наследующее требуемые характеристики. Генетический алгоритм переносит проблему планирования на набор строк (хромосом), каждая строка представляет потенциальное решение. Три основных аспекта использования генетических алгоритмов заключаются в следующем: определение целевой функции, определение и реализация генетического представления и определение и реализация генетических операторов.

Основной идеей алгоритма муравьиной колонии является симуляция поведения муравьев при поиске пропитания. Когда группа агентов (муравьев) ищет пропитание, они используют особый тип химического вещества для связи друг с другом, обычно называемый феромоном. Изначально муравьи начинают поиск еды случайным образом, в случайном направлении. При нахождении еды муравей оставляет определенную величину феромона на пути к источнику еды. По мере работы алгоритма большинство муравьев выбирает наикратчайшее расстояние до источника еды на основании концентрации феромонов, сосредоточенных на пути к источнику. Основным преимуществом такого рода алгоритма является механизм позитивной реакции, внутреннего параллелизма и расширяемости. Недостаток алгоритма – явление, когда на каком-то этапе работы алгоритма все муравьи приходят к одному и тому же источнику еды (решению), однако данное решение может быть лишь локальным минимумом. Алгоритм муравьиной колонии возможно применить практически к любой комбинаторной задаче [125].

Изначально алгоритм симуляции отжига был разработан по аналогии с химическими процессами, происходящими при отжиге металла. Отжиг включает в себя нагревание и остывание металла в целях изменения его физических свойств в результате изменений в его внутренней структуре. По мере остывания металла структура фиксируется, при этом сохраняя свои приобретенные свойства. В симуляции отжига постоянному изменению подвергается температура, при которой происходит процесс отжига. Изначально температура достаточно высока, затем она плавно понижается по мере работы алгоритма. Чем выше температура, тем чаще алгоритму будет разрешаться принимать решения, которые хуже текущего решения. Это дает алгоритму возможность преодолевать локальные оптимумы, которые

находятся в начале вычисления. По мере понижения температуры шанс принятия заведомого неверного решения уменьшается, тем самым алгоритм больше концентрируется на области поиска, которая, скорее всего, приведет к нахождению оптимального решения. Постепенное понижение температуры обосновывает основное преимущество алгоритма – высокую близость решения к оптимальному при наличии большого изначального числа локальных оптимумов [126].

Алгоритм PSO (Particle swarm optimization) был опубликован в 1995 году и является алгоритмом, моделирующим поведение стаи, решающей оптимизационную проблему в определенном пространстве поиска [127]. Поведение стаи базируется на социально-биологических принципах и обеспечивает понимание социального поведения, а также находит свое применение в решении задач из области информационных технологий и инженерии. Алгоритм PSO предполагает, что даются некая функция, позволяющая оценить найденное решение, и структура сети, соединяющая отдельных участников стаи, т. е. определяется связанность каждого участника с другими участниками. Затем случайным образом задается начальная популяция участников стаи. Далее запускается процесс движения участников стаи, на каждом из этапов производится оценка полученного решения, в случае получения удовлетворительной оценки участники стаи запоминают свое положение. Информация о текущем положении участника и о его положении, где решение было удовлетворительным, доступна его соседям. Передвижения участников стаи осуществляются на основании успешных передвижений соседей. В итоге обычно популяция участников стаи сходится, давая удовлетворяющее решение. Каждый участник стаи представляет собой решение оптимизационной проблемы. Расположение участника обуславливается как его собственным опытом, так и положением его соседей. Когда соседями участника стаи становится вся стая, тогда расположение рассматриваемого участника лучшее и является глобально лучшим решением оптимизационной задачи. Если же используется меньшее число соседей, то тогда лучшее решение рассматриваемого участника является локально лучшим решением оптимизационной задачи. Производительность каждой частицы измеряется с помощью целевой функции, которая устанавливается в зависимости от самой оптимизационной проблемы.

Активно развивается SAT (propositional SATisfiability problem) подход [128], который применяется в том числе в задачах планирования. SAT предполагает составление булевой формулы, описывающей искомый план. Далее специальный решатель пытается найти такие значения переменных в формуле, при которых формула была бы истинной. Найденные значения переменных дают план, гарантирующий выполнение задач. Существующие решатели достаточно быстро производят поиск решения. Проблема применения данного подхода заключается в том, что нахождение оптимального плана производится многократным запуском решателя с модификацией булевой формулы таким образом, чтобы предыдущие решения не приводили формулу к истине. Немаловажным фактором является то, что часто в задаче поиска плана нахождение точного решения плана некритично, в то время как непосредственное время поиска имеет большое значение.

Стоить отметить, что большинство алгоритмов планирования разрабатывалось с расчетом на использование в высокопроизводительных вычислениях, когда вычислительные узлы почти гарантировано способны выполнять сервисы и чьи характеристики, а также их степень занятости, заранее известны. С развитием сервис-ориентированного подхода характеристики среды изменились в силу ее гетерогенности, которая заключается в том, что вычислительные узлы, на которых развернуты сервисы, обладают различной производительностью. То же самое касается сетевой инфраструктуры, в которую интегрированы вычислительные узлы. В совокупности эти факторы дают различное время выполнения для одного и того же сервиса с одними и теми же входными параметрами на разных вычислительных узлах.

Гибридные алгоритмы планирования

В последнее время стали все чаще появляться гибридные алгоритмы, сочетающие разные техники, например, алгоритм Hybrid Evolutionary Workflow Scheduling Algorithm for Dynamic Heterogeneous Distributed Computational Environment [129], сочетающий HEFT и генетический алгоритм. Данный алгоритм способен адаптироваться к изменениям вычислительной среды, что особенно важно в распределенных вычислительных средах, когда работа отдельных вычислительных узлов и каналов связи не гарантируется, а время выполнения

сервисов и передачи данных между ними также изменяется со временем. Данный алгоритм учитывает изменения, происходящие в вычислительной среде, например, отключение вычислительного узла или добавление вычислительных сервисов к композиции ведет к мгновенной перестройке расписания, в то время как несоответствие фактического и ожидаемого времени выполнения сервиса будет учтено только в момент следующей плановой перестройки расписания.

Параллельная обработка пространственных данных

При обработке пространственных данных часто возникает ситуация, когда объемы обрабатываемых данных затрудняют их обработку в силу более высоких требований к аппаратной части или временным затратам. Для уменьшения времени выполнения вычислений или более оптимального использования аппаратных мощностей разрабатываются новые и развиваются существующие подходы выполнения распределенных вычислений.

Одним из наиболее популярных подходов обработки данных является реализация вычислений на основе модели распределенных вычислений MapReduce [130]. Свободно доступная реализация MapReduce – Apache Hadoop [131] позволяет осуществлять контроль и управление вычислительными узлами, а также предоставляет такие средства, как распределенную файловую систему (единое файловое пространство для выполняемых сервисов), и собственную распределенную СУБД. Расширение Spatial Hadoop [132] ориентировано на работу с большими массивами пространственных данных. Особый интерес привлекает к себе гибридный подход HadoopDB [133], состоящий из распределённой СУБД Postgres и Apache Hadoop. Суть HadoopDB состоит в связывании нескольких одноузловых систем баз данных с использованием Apache Hadoop в качестве координатора задач и сетевого коммуникационного слоя MapReduce.

В работе [134] авторы предлагают на основе Apache Hadoop и MapReduce систему обработки изображений с автоматическим распараллеливанием данных между вычислительными узлами. Система состоит из двух частей Apache Hadoop и реализованных обработчиков изображений с аппаратно программным интерфейсом (API), встроенных в пакет ImageProcessingLibrary. Система ориентирована на обработку коллекции независимых изображений. В статье [135] авторы

использовали Hadoop-GIS для обработки пространственных данных, причем входные данные содержат множество полигональных или других объектов и данные распределяются между вычислительными узлами блоками. При таком подходе возникает ситуация, когда один объект содержится в двух и более блоках одновременно и необходимо определить, к какому блоку его отнести. Для решения этой задачи авторы предлагают два способа. Первый – исключить эти объекты из обработки и потерять малую часть данных. Второй – произвести дополнительные вычисления по обработке конфликтных ситуаций. В этом случае достигается точный результат, а общее время работы увеличивается незначительно. Основное отличие данной работы заключается в том, что в силу специфики распределенной системы нет возможности настроить общую распределенную систему хранения и передачи данных. Описанный подход по обработке конфликтных ситуаций в [131] является перспективным, он будет более подробно рассмотрен, улучшен и адаптирован под цели данной работы.

1.4. Инфраструктура пространственных данных

Геопортал – это программно-технологическое обеспечение для работы с пространственными данными. Его основная задача – обеспечение пользователя средствами и сервисами хранения и каталогизации, публикации и загрузки пространственных данных, поиска и фильтрации по метаданным, интерактивной веб-визуализации, доступа к пространственным данным на основе картографических веб-сервисов.

Существует много разных определений термина «геопортал» – от несколько упрощенного его восприятия как «средства веб-картографирования», в том числе – с использованием спутниковых снимков, распространенного в системах информационно-справочного типа (например, сервисы Google Maps и Яндекс Карты), до более широкого его определения как одного из элементов инфраструктуры пространственных данных (ИПД).

Выводы

В рамках исследования выполнен анализ существующих направлений развития технологий распределенных вычислений, методов и подходов к созданию

композиций сервисов и их выполнения, который показал, что в настоящее время активно развивается SOA, появляется большое количество сервисов. В связи с этим, возникает необходимость решения следующих задач: поиск сервисов и реализация их взаимодействия, т. е. создание композиций сервисов. Композиции сервисов позволяют объединить методы и данные, созданные разными специалистами, и решать сложные задачи. Поиск и создание новых композиций сервисов является комбинаторно сложной задачей, в которой необходимо проанализировать большое количество сервисов и предоставить наиболее релевантные их комбинации. Существующие методы и подходы на основе анализа метаданных и онтологий не позволяют значительно упростить поиск из-за частого отсутствия метаданных и трудоемкости разработки онтологий. Создание композиций сервисов ограничено сложностью взаимодействия сервисов по данным из-за различий в программном интерфейсе доступа к данным, в структуре данных, используемых справочников и т. д. Поэтому является актуальной разработка сервис-ориентированной информационно-аналитической среды обработки междисциплинарных пространственных данных, повышающей эффективность научных исследований за счет создания композиций сервисов и организации обмена ими.

ГЛАВА 2. МЕТОД СОЗДАНИЯ КОМПОЗИЦИЙ СЕРВИСОВ

2.1. Анализ требований к проведению научных экспериментов на основе сервис-ориентированной парадигмы

В соответствии с поставленной целью повышения эффективности процессов подготовки и проведения научных экспериментов на основе сервис-ориентированной парадигмы за счет автоматизации построения и применения композиций сервисов, реализующих методы анализа и обработки ПД, рассмотрев текущее состояние современных информационных технологий, выделим основные требования к среде.

В научных исследованиях существует множество вычислительных методов, которые можно разработать в виде сервисов. Способов и приемов разработки программных интерфейсов сервисов и протоколов взаимодействия с ними большое количество. В работе рассматриваются сервисы, которые являются слабо связанными, заменяемыми программными компонентами, оснащенными стандартизированными интерфейсами для взаимодействия по стандартизированным протоколам. Сервисы обработки потоковых данных в данной работе не рассматриваются. Каждый сервис имеет идентификатор, обычно URI, множество входных и выходных параметров, реализует протокол взаимодействия с сервисом, определяющим передачу параметров. Большинство сервисов обработки пространственных данных являются сервисами без хранения состояния, т. е. результат работы сервиса зависит только от входных параметров. В связи с этим выполнение такого сервиса достаточно просто повторить. Для этого достаточно указать те же самые значения параметров и вызвать сервис. Сервисы обычно хранят результаты работы длительное время. Поэтому если вызов сервиса производился ранее, то результаты его работы можно получить без повторного вызова. Это обосновывает необходимость хранения истории вызовов сервисов (заданий) для ускорения выполнения композиции сервисов.

Совместное применение настольного ПО вместе с сервисами имеет некоторые сложности. Например, включение методов настольного приложения в композицию сервисов является не тривиальным из-за сложности запуска, часто отсутствия

программного интерфейса и необходимости участия человека, что может привести к значительному увеличению времени вычислений. Поэтому необходимо по возможности исключить применение настольного ПО. Среда должна обеспечивать на основе сервисов все этапы исследований. Это требование является реализуемым для множества задач из-за появления большого набора открытых библиотек, методы которых можно представить в виде сервисов. С точки зрения получения конечного для пользователя результата композиция сервисов должна содержать как минимум три сервиса: предоставления данных, обработки и публикации результатов.

Сервис-ориентированная среда должна обеспечить функциональную расширяемость с помощью включения в работу новых сервисов. В связи с необходимостью перевода методов открытых библиотек в WPS сервисы среда должна обеспечить с помощью инфраструктурных компонент разработку сервисов и их развертывание, что может обеспечить значительное расширение набора сервисов среды. При появлении нового сервиса необходимо предоставить его метаданные всем участникам среды. Среди исследователей в рамках одной группы или сообщества исследователей необходимо организовать специализированный обмен метаданными сервисов. Обычно представление метаданных выполняется с помощью регистрации сервиса в каталоге, где сохраняются все метаданные, необходимые для применения: адрес, параметры, типы данных, описание и т. д. Метаданных о сервисе должно быть достаточно, чтобы обеспечить его запуск, передачу параметров (протокол взаимодействия) и поиск. Метаданные должны позволять проводить поиск сервиса с учетом его входных и выходных данных. Кроме того, в рамках среды требуется обеспечить пользовательский интерфейс вызова сервисов и передачу необходимых данных.

Для того чтобы пользователи среды могли использовать новые сервисы, они должны удовлетворять некоторым правилам взаимодействия (протоколу). Протокол взаимодействия сервисов должен быть стандартизован. Обработка данных для научных исследований может занимать длительное время. Например, расчет модели распространения загрязнений атмосферного воздуха может занимать несколько часов или дней. Поэтому одним из требований к протоколам вызова сервисов является поддержка длительного выполнения. Большая часть данных содержит пространственную привязку, которая накладывает свои ограничения на обработку

данных сервисами. Поэтому наиболее подходящим стандартом, удовлетворяющим перечисленным требованиям, является Web Processing Service, разработанный Open Geospatial Consortium. Стандарт WPS на случай длительного выполнения предполагает, что сервис возвращает ссылку, где можно отслеживать процент выполненной работы и получить результаты.

Существуют сложные сервисы, решающие некоторую последовательность задач и применяющие совокупность методов. Обычно они содержат не только вычислительные методы, но и методы отображения данных и ввода. Часто применение отдельного метода в таких сервисах невозможно, так как отсутствует программный интерфейс. Поэтому разбиение сложного сервиса на ряд атомарных сервисов (т. е. которые нельзя разделить на еще более мелкие сервисы) позволяет использовать их по отдельности, увеличивает количество возможных композиций сервисов и количество решаемых задач. Атомарность сервисов приводит к тому, что для решения задачи необходимо вызвать множество сервисов с указанием соответствующих параметров, для автоматизации вызова которых требуется композиция. Положительным моментом является возможность гибко менять сервисы, например сервис отображения данных. Поэтому в рамках среды следует разрабатывать атомарные сервисы.

В сервис-ориентированной среде любой сервис, который невозможно скомбинировать с другими, практически становится бесполезным. Поэтому важным моментом является то, что создание сервиса должно рассматриваться с учетом возможного взаимодействия с другими сервисами. Сервисы для входных параметров требуют данные определенных типов, т. е. с заданной структурой, единицами измерений, используемых справочников, точностью и т. д. Стандарт WPS обеспечивает метаданные, описывающие типы входных и выходных данных. Однако вероятность того, что типы выходных данных одного сервиса будут соответствовать входным данным другого сервиса, является низкой. Для формирования композиций сервисов требуется унификация типов данных. Проблема унификации типов данных достаточно сложная и может быть частично решена за счет предоставления инструментария в рамках среды для конвертации

данных, перепроецирования, определения общих типов данных и методов создания сервисов под определенные типы данных.

Среди всех типов данных выделим реляционные таблицы, с помощью которых ведется сбор множества данных. Они довольно часто используются в качестве входных данных вычислительных сервисов. Реляционные таблицы характеризуются схемой данных, т. е. набором атрибутов, их типов и т. д. Часто разные коллективы исследователей ведут сбор для одних и тех же данных, но при этом используют отличающиеся схемы. Вычислительные сервисы могут иметь отличия в схемах входных данных. Все это приводит к тому, что данные пользователей невозможно передать вычислительному сервису без конвертации. Конвертация – достаточно сложный процесс, часто требующий ручного труда, например, из-за различий в названиях атрибутов, используемых справочников и т. д. Значительно проще изначально собирать данные в рамках определенной схемы, чем проводить потом их конвертацию. Публикация схем данных позволит создавать сервисы ввода данных по заданным схемам и, соответственно, сервисы обработки данных, что позволит без обработки передавать между ними данные. Поэтому необходимо упростить создание сервисов ввода данных, реализующих заданную схему.

Атомарность сервисов для новых задач требует формирования композиций сервисов. Каждая композиция является сервисом. Формирование композиций является достаточно сложным процессом, в котором необходимо выбрать сервисы, определить последовательность их применения и задать параметры. Количество сервисов постоянно растет. Часто сервисы выполняют аналогичные функции с небольшими отличиями в наборе параметров, используемых структурах данных, точности, скорости работы и т. д. Формирование композиции сервисов является комбинаторно сложной задачей, в которой необходимо перебрать большое количество сочетаний сервисов. Каждое сочетание сервисов необходимо проверить на согласованность по передаваемым данным, т. е. требуется организация автоматизированного поиска композиции сервисов с оценкой релевантности сервисов поисковому запросу (решаемой задачи).

На основе представленных метаданных и описаний пользователи производят выбор сервисов, наиболее подходящих под решаемую задачу. Общее количество сервисов обработки пространственных данных является достаточно большим. Поэтому требуется разработать механизм поиска, который помогает пользователям найти нужный сервис. Механизм поиска сервисов должен учитывать требования пользователя к входным и выходным данным и ранжировать сервисы по релевантности, т. е. по степени соответствия, актуальности и достоверности запросу пользователя.

Пользователь может провести несколько вычислительных экспериментов, в рамках которых он применяет разные композиции сервисов. Вычислительные эксперименты могут быть как успешными, так и неуспешными. В большинстве случаев успешность эксперимента может оценить только пользователь, т. е. не существует формализованной процедуры, которая автоматически определяет успешность эксперимента для любых задач. Успешность эксперимента можно оценивать только по косвенным признакам. Одним из таких признаков является публикация пользователем его результатов в открытом доступе. При публикации результатов необходимо представить их в удобном для человека виде. Например, в картографии существуют методы создания карт, в которых специалисты задают стили отображения данных на карте. Задание стилей отображения – это сложный процесс, занимающий много времени и требующий применения экспертных знаний. С одной стороны, маловероятно, что пользователь будет тратить свои усилия на публикацию результатов неуспешного эксперимента, с другой – публикация результатов говорит о том, что пользователь считает их полезными. Публикация данных в рамках СОА осуществляется также с помощью сервисов. Поэтому сервисы публикации данных необходимо выделить среди общего множества сервисов для того, чтобы автоматизировать оценку композиций.

Информация о применении пользователями сервисов может быть полезной при их ранжировании. Реализация СОАИС позволяет получать опыт применения сервисов множеством пользователей. Поэтому требуется сформировать новые критерии оценки композиций сервисов, которые учитывают полезность с точки

зрения пользователя (автора результата) и востребованность другими пользователями.

Одним из важных условий развития сервис-ориентированной среды является доступность сервисов, т. е. сервисы должны быть развернуты на работающих в Интернет вычислительных узлах со свободными вычислительными мощностями. Наличие множества готовых для применения сервисов значительно ускоряет научные исследования, позволяет исследователю за короткое время перебрать множество различных методов. Большинство разработчиков сервисов не имеют собственных таких вычислительных узлов. Существует возможность развертывания сервисов на платных ресурсах. Однако научные сервисы обычно используются узким набором специалистов и на нерегулярной основе. Соответственно, сервисы редко могут приносить постоянный доход, который можно использовать на оплату ресурсов. Поэтому для развития среды требуется предоставлять вычислительные узлы для развертывания сервисов с предустановленным программным обеспечением, упрощающим создание сервисов, удовлетворяющих стандартам.

Сервис-ориентированная среда должна обеспечить работу множества пользователей с распределенными сервисами на ограниченных вычислительных ресурсах. На программных и аппаратных платформах, на которых развернуты сервисы, существуют различные системы управления, мониторинга, балансировки нагрузки и т. д. Доступ к этим систем обычно отсутствует. Поэтому при реализации среды необходимо опираться только на наличие программного интерфейса, с помощью которого можно вызвать сервис и производить обработку. Предполагается, что узел, на котором развернут сервис, должен обрабатывать только одно задание в одну единицу времени. Одновременное выполнение двух и более заданий на одном узле может привести к увеличению суммарного времени выполнения из-за использования общих ресурсов узла (процессорного времени, файловой системы и т. д.). Программно получать состояние узлов (обрабатывают ли они в этот момент времени задание) для неконтролируемых узлов нет возможности из-за предположения, что имеется только доступ на основе стандарта WPS, REST, RPC и т. д. Поэтому следует не допускать запуск одновременно нескольких заданий, отслеживать выполнение заданий и фиксировать их завершение.

Для создания композиций сервисов требуется метод, который учитывает перечисленные требования и позволяет связывать различные сервисы между собой для достижения целей пользователей. При этом метод должен учитывать непроработанность метаданных сервисов, недостаточную онтологическую формализацию данных и процессов и давать возможность разрабатывать новые сервисы или средства обработки данных для создания композиций сервисов.

2.2. Вычислительная модель композиции сервисов

Работа метода основывается на новой вычислительной модели композиции сервисов, определяемая структурой

$CM =< Z, S, O, A, T, CR, F >$ со следующими обозначениями.

Z – множество параметров. В стандарте WPS OGC обработки пространственных данных параметры связаны с определенными типами данных.

S – множество сервисов. В работе рассматриваются только сервисы без хранения состояния (stateless), т. е. результат выполнения сервисов не должен зависеть от предыдущих вызовов. Этого достаточно для решения широкого круга задач. Сервис $s \in S$ – это упорядоченная тройка $s=<name, Z^{Inp}, Z^{Out}>$, где *name* – это имя сервиса, $Z^{Inp} \subset Z$ – множество входных параметров, $Z^{Out} \subset Z$ – множество выходных параметров сервиса. Среди выходных параметров сервиса имеется параметр успешности выполнения сервиса. Далее будем обозначать $s.Z^{Inp}$ и $s.Z^{Out}$ параметры, принадлежащие определенному сервису. Сервисы разделяются на сервисы получения данных $S^d \subset S$, вычислительные сервисы $S^c \subset S$ и сервисы публикации данных $S^p \subset S$.

$O \subset S \times S$ – множество отношений между сервисами, определяет возможность передачи данных между сервисами.

A – множество пользователей, которые взаимодействуют с СОИАС через применение сервисов S. Набор применяемых ими сервисов может варьироваться.

T – множество заданий выполнения сервисов из S. Каждое задание $t \in T$ определяется упорядоченной четверкой $t=<a, s, V_I, V_O>$, т. е. вызов сервиса $s \in S$ пользователем $a \in A$ с множеством значений V_I входных параметров $s.I$ и результаты V_O выходных параметров $s.O$.

CR – множество вычислительных узлов, на которых выполняются сервисы. Каждый сервис может быть активирован на заранее определенных узлах. Автоматическое выделение новых узлов для сервисов в работе не рассматривается.

F – множество функций ранжирования сервисов.

Отметим, что композиция сервисов является также сервисом с заданным именем, входными и выходными параметрами. Общепризнанным стандартом определения композиций является использование направленного ациклического графа (Directed Acyclic Graph, (DAG)) s_i=DAG_i=<T_i,E_i >, $s_i \in S$, в котором вершинами являются задания $t \in T_i$, а ребра e=<v_k,v_l > , $e \in E_i$ показывают передачу данных от выходного параметра одного задания к входному параметру другого задания, где $v_k \in V_I^i$, $v_l \in V_O^j$, $t_i =< a_i, s_i, V_I^i, V_O^i >$, $t_j =< a_j, s_j, V_I^j, V_O^j >$. Поскольку возможна передача данных одновременно по нескольким параметрам, то ребер между двумя вершинами (заданиями) также может быть несколько. В литературе чаще всего передача данных между заданиями обычно обозначается одним ребром. Для создания готовой к выполнению композиции необходимо определить все передаваемые данные между параметрами.

Определим типы данных, используемые для обработки пространственных данных. Частично примем обозначения, принятые в реляционной алгебре:

$D = \{D_1, \dots, D_l\}$ – множество скалярных типов, определенных в стандарте WPS;

$R = \{R_1, \dots, R_m\}$ – множество схем отношений;

$d = \{d_1, \dots, d_k\}$ – множество таблиц (отношений). В качестве параметров могут использоваться реляционные таблицы $d_i \in Z$.

Введем множество типов параметров $TZ = D \cup R$. Для каждого параметра определен тип $tz_j = type(z_i)$, $tz_j \in TZ$.

Для формирования композиций сервисов требуется информация о возможности передачи данных выходного параметра сервиса для использования во входном параметре другого сервиса. Однако соответствие типов параметров не является достаточным, например, тип «число» может иметь смысл – высота над

уровнем моря, а в другом случае – температура. Поэтому соответствие типов является необходимым условием, но не достаточным. Получение информации о возможности передачи данных между параметрами может производиться на основе анализа метаданных, онтологий, экспертных знаний и статистики применения сервисов и т. д. Поэтому вводим понятие метки, которое позволяет абстрагироваться от способа получения информации о возможной передаче данных между параметрами. Метка используется для идентификации возможности передачи данных, т. е. если выходной параметр и входной параметр имеет одну и ту же метку, то между ними возможна передача данных. Каждому параметру может соответствовать ноль и более меток. Метками могут являться онтологические концепты, заданные с помощью URI, или некоторые сгенерированные уникальные значения.

$m \in M$ – множество меток. Для каждого параметра z_j определено подмножество меток M_j. Проверка возможности передачи данных между сервисами в рамках модели выполняется на основе соответствия типов параметров и меток. Для двух сервисов s^i и s^j возможна передача данных от параметра $z^i \in s^i.Z^{Out}$ к параметру $z^j \in s^j.Z^{Inp}$, если типы параметров совпадают $type(z_i) = type(z_j)$ и существует метка $m \in M$, которая имеется у обоих параметров ($m \in M_i$ и $m \in M_j$).

Выполнение этого условия не гарантирует корректное выполнение сервиса s^j. Оно позволяет значительно сузить количество возможных композиций сервисов.

2.3. Основные этапы метода

Процесс создания композиции сервисов представлен на рисунке 2.1 и состоит из трех основных этапов.

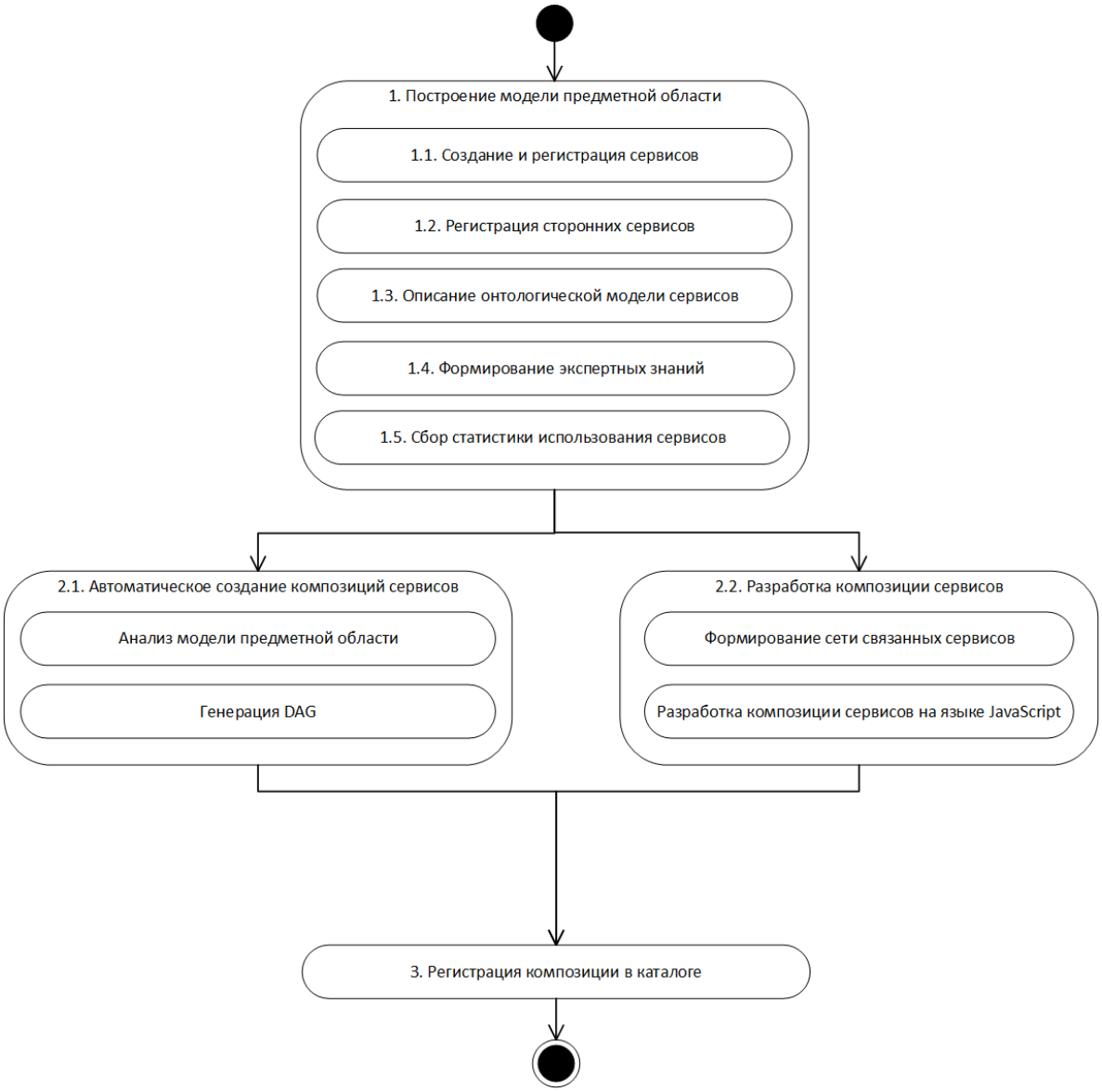


Рисунок 2.1 – Схема метода формирования композиций сервисов

Этап 1. Построение модели предметной области

Состоит из 5 шагов, выполняемых независимо друг от друга.

1.1. Создание и регистрация сервисов – предполагается разработка вычислительных сервисов, сервисов данных и сервисов публикации данных. Вычислительные сервисы разрабатываются в соответствии со стандартом WPS на вычислительных ресурсах среды или на любых других серверах, к которым имеется сетевой доступ. Для создания сервисов данных используется компонент «Фабрика сервисов ввода и редактирования реляционных данных», который поддерживает все основные этапы работы с табличными данными. «Фабрика сервисов отображения

графиков» и «Фабрика сервисов отображения пространственных данных» предназначены для создания сервисов публикации данных. Все создаваемые сервисы регистрируются в каталоге, обладают программным интерфейсом и готовы стать частью композиции сервисов.

1.2. Регистрация сторонних сервисов, т.е. осуществляется подключение WPS сервисов, созданных на сторонних вычислительных ресурсах. В каталоге сервисов указывается необходимая для запуска сервиса информация и метаданные.

1.3. Описание онтологической модели сервисов предназначено для определения возможности передачи данных между параметрами сервисов на основе онтологий. В рамках описания онтологической модели сервисов предполагается определение концептов для параметров сервисов. Каждому параметру может соответствовать ноль и более меток. Метками могут являться онтологические концепты, заданные с помощью URI. Использование одного и того же концепта для выходного параметра сервиса и для входного параметра другого сервиса предполагает возможность передачи данных между ними и присвоение одной метки для них. Одним из основных отношений, которые применяются для формирования меток, является отношение частное – общее. В онтологии для задания этого отношения между концептами используется элемент subClassOf. Например, если сервис имеет входной параметр z_j с заданным некоторым концептом k, то в качестве входных данных могут использоваться данные сервисов, которые имеют концепты, являющиеся частными по отношению к k .

1.4. Формирование экспертных знаний – используется для определения возможности передачи данных пользователем. Разработка онтологий – достаточно сложный процесс. Часто с помощью онтологий не все сущности и отношения предметной области формализованы. Поэтому в качестве меток параметров сервисов могут использоваться некоторые сгенерированные уникальные значения, назначаемые пользователем для конкретных сервисов. Например, сервис построения карты плотности может применяться для всех точечных данных. Соответственно, выходным параметрам, которые предоставляют векторный точечный слой, можно

назначить ту же саму метку, что и у входного параметра сервиса построения карты плотности.

1.5. Сбор статистики выполнения сервисов, т.е. производится сбор информации о произведенных вызовах сервисов, которые включают название сервиса и его адрес, значения входных и выходных параметров, время выполнения сервисов, успешность выполнения, ошибки выполнения и т. д. Данные сохраняются в таблице (таблица 2.1), где значения параметров хранятся в формате JSON.

Таблица 2.1. Данные о вызовах сервисов

Id задания	Id сервиса	Статус	Входные параметры	Выходные параметры	Начало	Конец
3986	309	Success	{"table":"ticki"}	{"file":"http://84	18:20	18:22
3989	307	Failed	{"disp":"20"}		18:21	
3985	308	Success	{"file":"http://84..."}	{"res":"http://84	18:21	18:22
3987	307	Success	{"file":"http://84..."}	{"dist":"28”,	19:18	19:19

Далее данные этой таблицы будем обозначать как *Log* – множество заданий (успешных вызовов сервисов), совершенных пользователями. Причиной неудачного вызова сервиса могут быть некорректные значения параметров, поэтому данные фильтруются и оставляются только успешные вызовы.

Производится анализ статистических данных, определение передачи данных между параметрами сервисов и присвоение меток. Определение связи между двумя вызовами сервисов может производиться на основе анализа значений параметров. Множество параметров можно разделить на входные и выходные. Предполагается, если значение выходного параметра совпадает со значением входного параметра, то между вызовами сервисов может быть передача данных. Значения параметров могут быть строковыми или числовыми, а также идентификаторами ресурсов URI, обычно это URL ссылки на файлы, сервисы предоставления данных и т. д. Каждый URI, используемый для получения данных WPS сервисом, однозначно идентифицирует передаваемые данные. Соответственно, по URI параметрам можно однозначно определить, что данные передаются от одного вызова сервиса другому. Несколько сложнее со строковыми и числовыми параметрами, совпадение значений которых может быть случайным. На текущий момент строковые и числовые параметры не рассматриваются.

Приведем алгоритм проверки передачи данных между двумя вызовами сервисов.

Input: *ti*, *tj* – два задания

Output: flag – наличие связи между заданиями

```
1   function islinked(t_i, t_j)
2   flag <- False
3   for each v_m in t_i.VO:
4     for each v_n in t_j.VI:
5       If (typeof v_m = URI and typeof v_n = URI and v_m =v_n) then
6               flag <- true
7         v_n.linkedwith <- v_m
8       end if
9     end for
10  end for
11  return flag
```

Данный алгоритм кроме наличия связи между вызовами сервисов определяет, по каким параметрам они связаны.

Этап 2. Создание композиций сервисов

После построения модели предметной области на этапе 2 применяются два альтернативных способа создания композиций:

- автоматическое формирование композиций сервисов, позволяет создать композицию сервисов на основе анализа графа связности заданий;
- разработка композиций сервисов пользователем с помощью языка JavaScript. Этот способ требуется при необходимости обработки промежуточных данных с помощью средств языка, или при заранее неизвестном количестве заданий в композиции DAG_i, которые формируются в зависимости от данных.

Этап 3. Регистрация композиции сервисов в каталоге

Последним этапом является сохранение полученных композиций сервисов в каталоге для последующего применения пользователями.

2.4. Алгоритм формирования сети связанных сервисов

Одним из способов, упрощающих разработку композиции сервисов, является формирование сети связанных сервисов [136]. Сеть связанных сервисов определяет возможные связи сервисов по передаваемым данным, т. е. дуги этой сети задают

возможность элементарной композиции – цепочки выполнения из двух сервисов, когда результаты работы одного сервиса могут быть применены в качестве входных данных другого сервиса. Будем обозначать сеть сервисов с помощью графа $SG = < S, L >$, в котором вершинами являются сервисы $s \in S$, а ребра $l = <s_k, s_l>$, $l \in L$ показывают возможность передачи данных от одного сервиса к другому.

Для формирования сети связанных сервисов используют метаинформацию [137, 23] и онтологии [15]. В частности, осуществляется построение семантической сети на основе WSDL. Следует отметить, что автоматическое построение сети связанных сервисов является сложной задачей по следующим причинам:

- сервисы имеют недостаточно метаинформации, либо она вообще отсутствует или рассчитана на человека;
- параметры некоторых сервисов могут быть связаны со многими другими сервисами. Например, сервисы алгебры растров могут использоваться во многих композициях сервисов для анализа пространственных данных. Связывание трех последовательных сервисов, где промежуточным сервисом являются сервисы алгебры изображений, требует проработанной онтологии предметной области. Ее построение – достаточно сложный процесс, в котором необходимо рассмотреть все возможные сущности и связи;
- требуется согласованность метаинформации сервисов и онтологии предметной области.

Для получения недостающей информации при построении сети связанных сервисов разработан алгоритм, основанный на анализе статистических данных о применении сервисов пользователями. Построение сети связанных сервисов сводится к задаче определения передачи данных между параметрами сервисов (causal link) $s.Z^{Inp}$ и $s.Z^{Out}$.

Разработан следующий алгоритм построения сети связанных сервисов.

Input: Log – множество заданий

Output: SG – связи между сервисами

```
for each ti in Log:
      for each tj in Log:
            for each vm in ti.VO:
                  for each vn in tj.VI:
                        If (typeof vm = file and typeof vn = file and vm = vn) then
                              add (ti, tj) into SG
                        end if
                  end for
            end for
      end for
end for
```

Данный алгоритм производит поиск сочетания вызовов сервисов, в которых файл, являющийся результатом работы одного сервиса, передается в качестве параметра другому сервису. Предложенный метод построения сети связанных сервисов апробирован на небольшом фрагменте данных о применении сервисов пользователями. Источники данных представлены в виде сервисов, у которых присутствуют только исходящие дуги. На рисунке 2.2 представлена часть полученной сети. Несвязанные сервисы (одиночные вершины) не отображены. Необходимо отметить, что при таком подходе семантика всех параметров сервисов не определяется. Среди положительных моментов укажем, что вместе с сетью связанных сервисов получаем связанность параметров по данным, возможные значения параметров, частотные характеристики использования пользователями сервисов и их композиций, время выполнения сервисов и т. д.

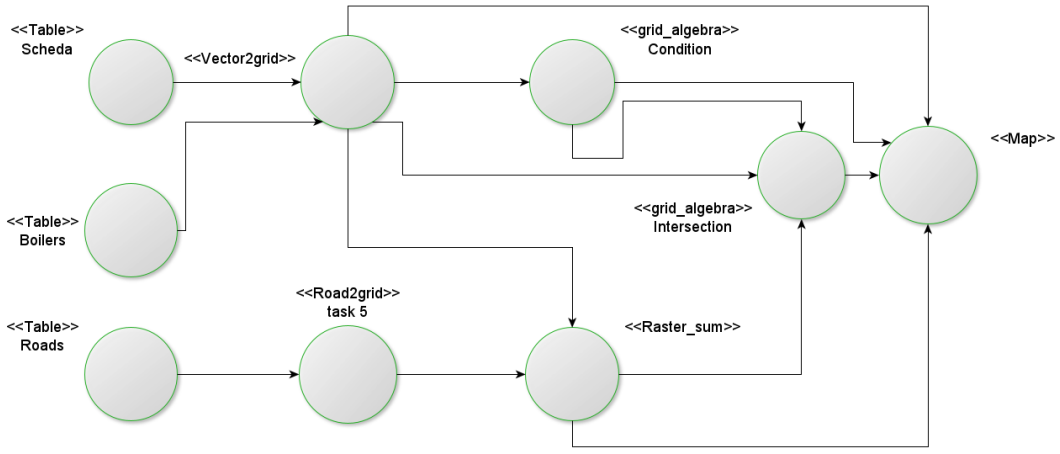


Рисунок 2.2 – Полученная сеть сервисов

Представленный метод позволяет сформировать сеть связанных сервисов на основе статистических данных применения сервисов. Сеть связанных сервисов значительно упрощает разработку композиции сервисов, предоставляя информацию о возможных цепочках из двух сервисов. Преимуществом данного алгоритма является то, что связь между двумя сервисами подтверждается работоспособным примером, демонстрирующим передачу данных от одного сервиса другому. Кроме того, вместе с сетью связанных сервисов получаем возможные значения параметров, частотные характеристики использования пользователями сервисов и их композиций, время выполнения сервисов и т. д. Анализ полученных данных является перспективным в следующих направлениях:

- автоматическое формирование композиции сервисов для типичных операций пользователя;
- определение альтернативных композиций сервисов;
- распознавание семантики параметров сервисов;
- подсказка разработчику возможных значений параметров и т. д.

2.5. Автоматическое формирование композиций сервисов

В СОА решение задачи может требовать многократного вызова сервисов, при этом пользователю приходится придерживаться определенной последовательности вызова сервисов, вводить одни и те же значения параметров для каждого сервиса и т. д. При многовариантных экспериментах это становится достаточно трудоемким процессом и требует много времени. Автоматизация процесса позволит значительно сократить время проведения эксперимента и сформирует готовый инструмент анализа и обработки данных для других пользователей. Автоматическое создание композиций производится на основе статистических данных, экспертных знаний и онтологий. Каждая композиция задается графом *DAG*, с заданными параметрами заданий и связями по данным. Создаваемые композиции сохраняются в каталоге сервисов, где пользователь может найти нужную композицию в соответствии со своим запросом. Автоматическое создание композиций состоит из следующих шагов:

Шаг 1. Создание графа заданий и их связей (граф DAG_{gen}).

Шаг 2. Декомпозиция графа заданий.

Шаг 3. Редукция множества композиций сервисов

Создание графа заданий и их связей производится на основе анализа таблицы заданий. Разработан следующий алгоритм построения графа DAG_{gen}.

Input: *Log* – множество заданий

Output: DAG_{gen} – граф заданий и их связей

```
1  for each ti in Log:
2       for each tj in Log:
3            If islinked(ti,tj)then
4                 add (ti, tj) into DAGgen
5                 newlabel() -> label
6                 addlabel(label, ti)
7                 addlabel(label, tj)
8            end for
9  end for
```

Функция **islinked** проверяет возможность передачи данных между двумя заданиями. После выполнения алгоритма из списка выполненных сервисов получим общий граф DAG_{gen} (рисунок 2.3), где все вызовы пользователей сервисов t_i связаны по данным.

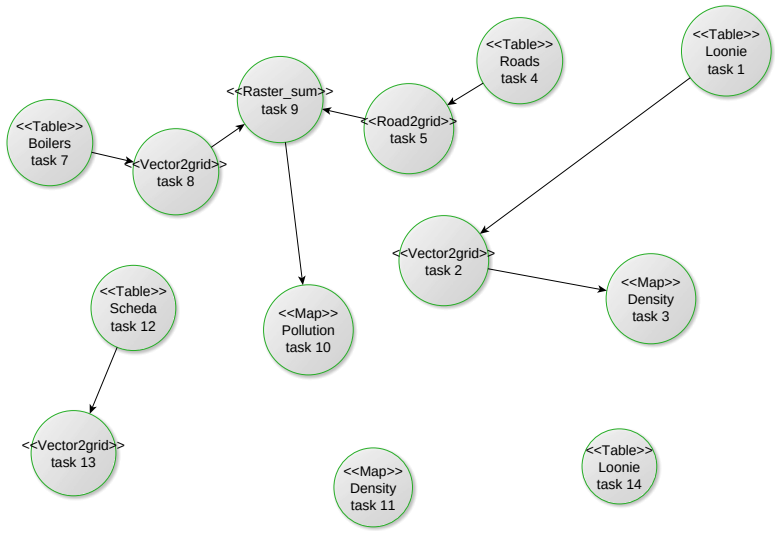


Рисунок 2.3 – Граф заданий DAG_{gen}

Дополнительно получаем возможные значения параметров сервисов, являются ли они результатами работы других сервисов или их ввел пользователь. Функция **islinked** определяет возможность передачи данных на основе меток, которые формируются с помощью метаданных сервисов, экспертных знаний и онтологий.

Далее производится декомпозиция графа заданий. Граф DAG_{gen} содержит задания, применяемые для решения множества разных задач. Следовательно, его необходимо декомпозировать и выделить отдельные подграфы, соответствующие решаемым задачам. Пользователь может выполнить некоторую последовательность сервисов, задающую последовательность заданий. Если задания связаны по данным, то они образуют связанный подграф D_i (компонента связности графа) в DAG_{gen}. Выполнение композиции сервисов всегда приводит к созданию связанного подграфа D_i. Если задания, сформированные композицией сервисов, не связаны, тогда такую композицию сервисов можно разделить. Для получения подграфа, соответствующего решению задачи, недостаточно найти компоненты связанности в DAG_{gen}, так как в подграфе связанности могут быть задания получения данных. Например, задания могут быть сервисами получения данных ДЗЗ или сервисами предоставления векторных и реляционных данных. Эти данные могут использоваться для решения большого количества различных задач. Например, данные ДЗЗ Landsat 8, которые можно получить с помощью сервисов, используются для задач оценки подтоплений из-за разлива рек, динамики изменения ледников, классификации породного состава лесов и т. д. Каждому решению задачи будет соответствовать свой подграф в DAG_{gen}, и они будут все связаны между собой через задание получения данных ДЗЗ, т. е. объединяться в компоненты связности. Поэтому при выделении подграфов, соответствующих только одной задаче, необходимо не рассматривать связи через задания получения данных, т. е. через сервисы данных $S^d \subset S$. Учет заданий, выполняемых сервисами предоставления данных, при выполнении анализа может быть полезен для определения параметров композиций сервисов, т. е. сервисов, которые могут быть заменены на другие при их повторном выполнении.

Для поиска связанных подграфов D_i применяется алгоритм поиска в ширину. При этом сложность поиска является линейной от суммы числа вершин и числа

рёбер графа DAG_{gen}. Это обосновывается направленностью графа и отсутствием циклов.

Input: *nodes* – граф DAG_{gen}

Output: subgraphs – массив связанных подграфов

```
1   Stack <-[];
2   while(nodes is not empty):// перебираем все задания
3         new subgraph
4         add subgraph into subgraphs
5         first <- nodes.pop()
6         add first into subgraph
7         add first into stack
8         while(stack is not empty){//перебор всех связанных заданий
9               cur <- stack.pop()
10              for each n in cur.nexts:
11                    if n exists in nodes then
12                          add n into subgraph.nodes
13                          add n into stack
14                          remove n in nodes
15                    end if
16              end for
17              for each n in cur.prevs:
18                    if n exists in nodes then
19                          add n into subgraph.nodes
20                          add n into stack
21                          remove n in nodes
22                    end if
23              end for
24        end while
25  end while
```

В результате выполнения алгоритма получаем множество связанных подграфов D_i (рисунок 2.4). На рисунке задания получения данных выделены пунктирной линией, задания публикации данных имеют более темную заливку, каждый *Di* является частным случаем реализации композиции сервисов.

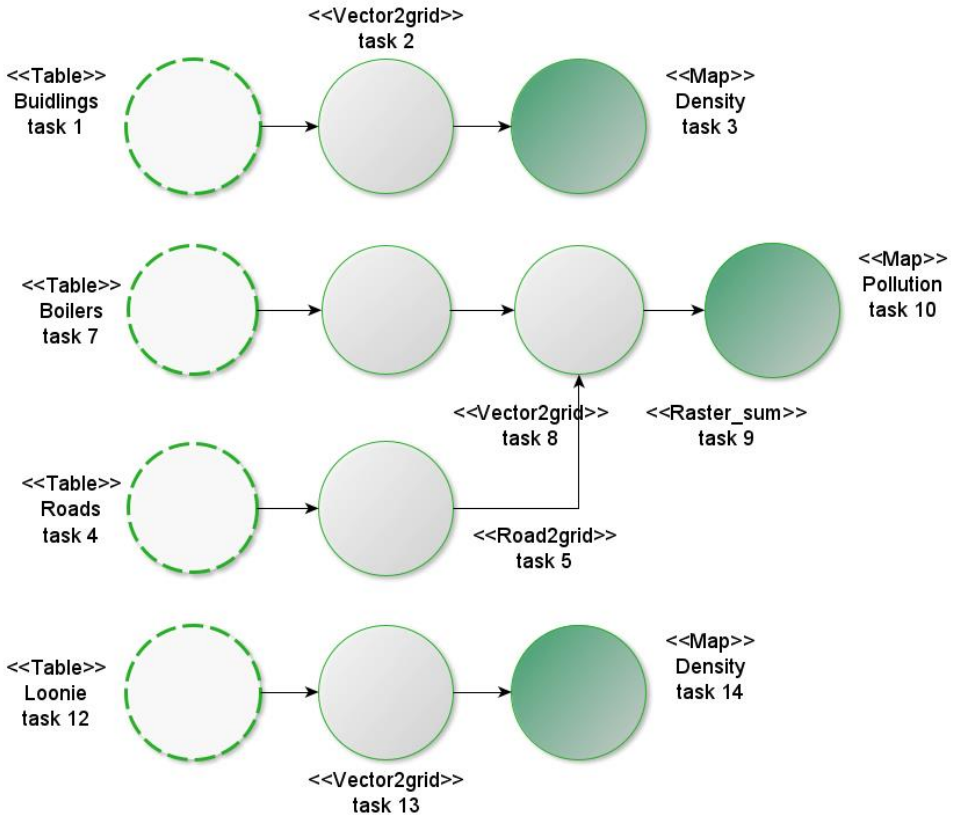


Рисунок 2.4 – Связанные подграфы D_i

Редукция множества композиций сервисов

Алгоритм поиска может получить достаточно много связанных подграфов D_i. Часть подграфов может соответствовать экспериментам, которые привели к недостаточно хорошим результатам. Также среди них может оказаться достаточно много дублирований, которые возникают из-за многократного решения задачи на разных данных и с разными параметрами. Поэтому возникает задача редуцирования множества D_i и разработки методов оценки, позволяющих ранжировать и выделить наиболее полезные композиции сервисов.

Композиции сервисов, состоящие из одной вершины (одного вызова сервиса), являются тривиальными, и для них нет необходимости формировать композиции. Поэтому подграфы D_i, состоящие из одной вершины, сразу отбрасываем. Одним из эвристических критериев необходимости связанного подграфа заданий может быть получение конечного для пользователя результата. Предполагается, что результат будет конечным, если пользователь его публикует. Например, на основе полученных данных создаются карты, графики, таблицы и т. д. В общем, среди всех связанных

подграфов выделяются D_i, состоящие как минимум из двух вершин и имеющие хотя бы одно задание публикации данных, т. е. вызова сервиса $s \in S^p$.

Поиск эквивалентных подграфов

Часто методики применения сервисов многократно повторяются на разных наборах данных. Соответственно, пользователь может повторять одну и ту же последовательность вызовов сервисов. При этом могут меняться входные данные, часть которых предоставляется сервисами. Это приводит к созданию нескольких подграфов D_i, в которых производятся вызовы одних и тех же вычислительных сервисов, отличающихся значениями параметров и сервисами предоставления данных. Заменим в связанных подграфах D_i задания t на соответствующие сервисы s. Ребра оставим без изменений, и в новом графе SD_i эти ребра вместо заданий будут связывать сервисы. Повторение одной и той же последовательности вызовов сервисов приведет к созданию изоморфных подграфов SD_i относительно сервисов и связей по данным без учета сервисов предоставления данных. Проверка на изоморфность двух графов осуществляется нижеследующим алгоритмом.

Алгоритм выделения изоморфных графов

Графы SD_i являются направленными, что значительно упрощает их сравнение. Предварительно для каждого графа добавляется вспомогательное начальное задание, несвязанное с сервисом, с которого начинается выполнение композиции. Все задания, не зависящие по данным от других, добавляются к начальному заданию. Сравнение двух графов начинается с начальных заданий и выполняется рекурсивно. Два задания являются изоморфными, если название сервисов совпадают и являются изоморфными все зависящие от них задания.

Input: n1, n2 – графы SD_i.
Output: true или false, являются графы изоморфными или нет.

```
1   function isonodes(n1,n2)
2   if n1.nexts.length <> n2.nexts.length then
3     return false
4   end if
5   for each nx1 in n1.nexts:
6     found <- false
7     for each nx2 in n2.nexts:
8       if nx1.service = nx2.service and isonodes(nx1,nx2) then
9         found <- true
10        break
11      end if
12    end for
13    if not found then
14      return false
15    end if
16  end for
17  return true
```

Изоморфные подграфы SD_i не всегда соответствуют одной и той же композиции сервисов. Например, один и тот же набор сервисов может применяться для получения разных результатов при изменении входных параметров. При дешифрировании данных ДЗЗ могут использоваться параметры, задающие различные пороговые значения. Их изменения могут привести к получению принципиально разных результатов. С другой стороны, изменение параметров не всегда означает применение другой композиции вычислительных сервисов. Например, композиция сервисов может быть применена для другой территории, т. е. изменен параметр, задающий территорию, но решает ту же самую задачу. Возникает проблема определения, соответствуют ли два изоморфных графа решению одной задачи. Одним из способов решения этой проблемы является определение соответствия семантики результатов изоморфных графов. Автоматическое определение семантики результатов является нетривиальной задачей, поэтому предлагается воспользоваться экспертными знаниями пользователей. В рамках среды пользователи могут указать, что два изоморфных графа друг другу соответствует с помощью присваивания меток.

В результате выполнения алгоритма множество компонентов связности разбивается на ряд подмножеств $IDAG_i$, где в каждом подмножестве все компоненты связности изоморфны. В результате выполнения алгоритма множество связанных подграфов разбивается на ряд подмножеств $IDAG_i$, где в каждом подмножестве все связанные подграфы изоморфны. Количество подграфов D_j в $IDAG_i$ является

оценкой частоты использования этой композиции сервисов. Далее проводится отбор изоморфных подграфов $IDAG_i$, у которых количество вершин более одного и имеется хотя бы один вызов сервиса для публикации данных.

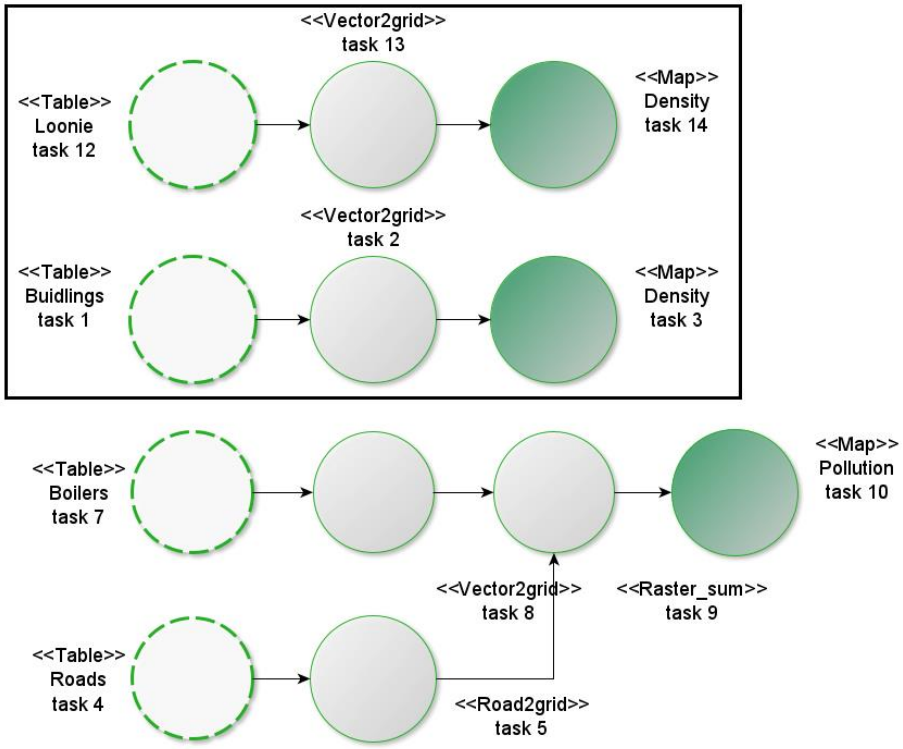


Рисунок 2.5 – Изоморфные компоненты связности выделены прямоугольником

На рисунке 2.5 приводится результат определения двух изоморфных подграфов SD_i, которые имеют одинаковую метку «Density» на сервисе публикации карт «Map».

Подграфы SD_i являются искомыми композициями сервисов. Для того чтобы выполнить композицию сервисов, представленную подграфом SD_i, необходимо пользователю задать все свободные входные параметры и переопределить сервисы предоставления данных. Все эти данные могут быть введены пользователем на генерируемой форме выполнения композиции сервисов. В качестве значений по умолчанию предлагаются наиболее часто используемые значения среди всех изоморфных подграфов SD_i.

Изоморфные подграфы SD_i с одинаковой семантикой результатов объединяются (рисунок 2.6). Количество $count(SD_i)$ изоморфных подграфов SD_i является оценкой частоты использования этой композиции сервисов. Частота использования может применяться, например, для отображения композиций сервисов в порядке, при котором отображаются сначала наиболее часто используемые.

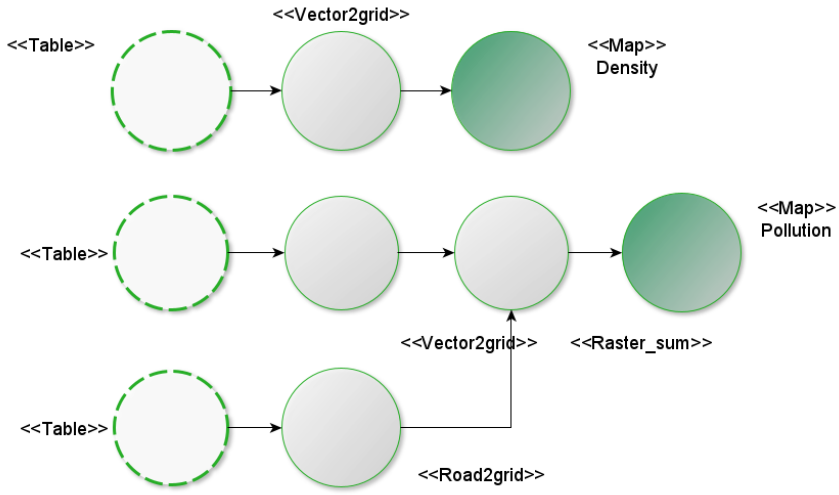


Рисунок 2.6 – Композиции сервисов

2.6. Постановка задачи поиска композиции сервисов

Модель CM позволяет решать задачу создания композиций сервисов. Поиск композиций сервисов должно проводиться в соответствии с некоторым запросом пользователя. Пусть Q – множество запросов, запрос $q \in Q$ – это упорядоченная пара q=<*Input*,*Output*>, где

- *Input* – описание начального сервиса;
- *Output* – описание конечного сервиса.

Нужно найти композицию сервисов s, представленную с помощью графа DAG, где вершины – это вызовы сервисов (задания), ребра – передача данных между заданиями. *Input* будет начальным сервисом, т. е. не будет получать данные от других сервисов, а *Output* конечным сервисом, т. е. не будет передавать свои данные другим сервисам композиции. Все вызовы сервисов (задания) в композиции образуют связанный граф. Формирование композиций является более сложной задачей, чем поиск последовательности сервисов (service chain). Так как часто

сервисы имеют несколько параметров, данные которых могут быть получены от нескольких заданий (сервисов).

В запросе *Input* или *Output* может быть не задан. Например, когда есть определенные данные, и нужно узнать, что можно из них получить. Либо наоборот, известно, что нужно получить, но не знаем, как и из каких данных.

Запросу $q \in Q$ может соответствовать достаточно большое количество композиций сервисов. Поэтому их требуется ранжировать (упорядочить) по степени соответствия запросу. Способы ранжирования сервисов зависят от целей пользователя, ограничений и т. д. Функция $f \in F,\ f(q,s) \to [0,+\infty]$ сопоставляет каждому сервису и запросу число – вычисленную релевантность запросу пользователя (ranking function). Эта функция учитывает соответствие различных признаков сервиса запросу.

2.7. Реализация поиска и эвристический метод ранжирования композиций сервисов

Создание возможных композиций производится при наступлении определенных событий, например, регистрация сервиса, выполнение сервиса и т. д. Создание композиций сервисов – достаточно длительный процесс, включающий несколько этапов. На текущий момент предварительное создание композиций сервисов является предпочтительнее, чем во время запроса пользователя из-за необходимости быстрого ответа. В то же время хранение всех созданных композиций сервисов на текущий момент не представляет сложности.

Поиск композиции сервисов производится в каталоге сервисов. Представим реализацию метода поиска композиции сервисов в рамках заданной модели. Запрос пользователя q=<*Input,Output*> состоит из двух частей: описания начального и конечного сервисов. В рамках программной реализации модели описание может быть задано в виде текста или указанием конкретных сервисов из каталога. В случае описания в виде текста производится поиск по ключевым словам в каталоге сервисов. Первоначально производится поиск *Input* сервисов, затем поиск *Output* сервисов, если они заданы. В общем случае будем обозначать множество найденных

сервисов S^{Input} для начальных, а S^{Output} для конечных сервисов. Если пользователь указал конкретные сервисы, то эти множества состоят из одного элемента.

Далее производится фильтрация композиций $DAG_i=<T_i,E_i>$, после которой остаются только композиции, для которых выполняются следующие условия:

1) $\exists s^{start} \in S^{Input}$, для которого имеется задание $\exists t^{start} =< a, s^{start}, V_I, V_O >$, и это задание принадлежит композиции $t^{start} \in T_i$;
2) $\exists s^{finish} \in S^{Output}$, для которого имеется задание $\exists t^{finish} =< a, s^{finish}, V_I, V_O >$, и это задание принадлежит композиции $t^{finish} \in T_i$;
3) существует путь из t^{start} в t^{finish}.

Если S^{Input} или S^{Output} – пустое множество, тогда требуем выполнения только условия 2 или 1, соответственно. Если количество найденных в каталоге сервисов равно нулю, то поиск прекращается и выводится сообщение пользователю, что сервисы не найдены.

Количество найденных композиций может быть большим. В этом случае требуется ранжировать найденные композиции для удобства пользователей. Функции ранжирования сервисов являются эвристическими. На текущий момент предложена функция, которая производит оценку по частоте использования композиции, т. е. по количеству изоморфных подграфов (см. раздел 2.5)

$$f(q,s) = count(DAG_i).$$

Данная оценка позволяет выделить композиции сервисов, которые активно применяются пользователями.

Реализация поиска в геопортале

Существует множество предметных специалистов, которые производят сбор больших массивов данных с помощью сервисов. Поэтому в пользовательском интерфейсе предоставляется функция поиска, где в качестве *Input* будет сервис получения данных $s \in S^d$. В результате предметные специалисты могут оперативно получать все возможные композиции сервисов (методы), которые можно применить к их данным. Реализация такого поиска упрощает применение композиций. При

этом список композиций сервисов будет пополняться по мере появления новых сервисов, добавления метаданных, онтологий или применения сервисов на аналогичных таблицах.

2.8. Сравнение предлагаемой модели, алгоритмов и методов

Построение композиции сервисов включает автоматизацию следующих шагов:

1) поиск сервисов;
2) построение связей (определение параметров, по которым производится взаимодействие по данным);
3) проверка совместимости параметров;
4) построение композиций сервисов (создание выполняемого описания, например, DAG).

Применение статистики позволяет распространять среди пользователей успешный опыт проведения вычислительных экспериментов. Выполнено сравнение существующих систем управления научными рабочими процессами (WMS) с предлагаемой СОИАС с точки зрения формирования композиций сервисов. Сравнение приведено в таблице (таблица 2.2).

Таблица 2.2. Сравнение СОИАС с другими подходами

Подходы	Поддержка автоматизации				
	Поиск сервисов	Построение связей	Проверка совместимости параметров	Построение композиций сервисов	Использование статистики
BPEL Designer Project (https://projects.eclipse.org/projects/soa.bpel)	+	-	-	-	-

GeoJModelBuilder (https://github.com/geoprocessing/GeoJModelBuilder).	+	-	-	-	-
Everest (Сухорослов О.В.)	+	-	-	-	-
CLAVIRE (Бухановский А.В.)	+	+/-	-	-	-
DiVTB (Радченко Г.И.)	+	-	-	-	-
Pegasus (https://github.com/pegasus-isi/pegasus)	+	-	-	-	-
Предлагаемая среда (СОИАС)	+	+	+	+	+

Результаты сравнения показывают, что предлагаемая реализация СОИАС на основе предложенной модели, алгоритмов и методов имеет ряд преимуществ при создании композиции сервисов.

Выводы

Разработана вычислительная модель композиции сервисов, которая позволяет создавать композиции сервисов. Модель впервые позволяет формировать композиции сервисов и их обмен на основе статистических данных использования сервисов пользователями, метаданных сервисов и онтологий. В модель введено множество функций ранжирования сервисов, которые позволяют оценить сервисы в зависимости от запроса и выбранных критериев. Предложен метод создания композиций сервисов. Использование метода может значительно упростить работу пользователя и автоматизирует часто повторяющиеся его действия. В результате применения модели и метода создания композиций опыт применения пользователем сервисов автоматически распространяется среди всех пользователей.

Результаты исследований, представленные в данной главе, опубликованы в [138-144].

ГЛАВА 3. АЛГОРИТМИЧЕСКАЯ И ПРОГРАММНАЯ РЕАЛИЗАЦИЯ СЕРВИС-ОРИЕНТИРОВАННОЙ ИНФОРМАЦИОННО-АНАЛИТИЧЕСКОЙ СРЕДЫ

3.1. Архитектура и основные компоненты

Разработана архитектура, реализующая вычислительную модель композиции сервисов CM. Архитектура обеспечивает функциональную расширяемость за счет сервисов и дает возможность пользователю работать полностью удаленно. Для удобства описание архитектуры представлено последовательно от общего к детальному. На рисунке 3.1 она представлена с точки зрения пользователя. Пользователь работает со средой с помощью браузера. Рассмотрим представленные на рисунке компоненты.

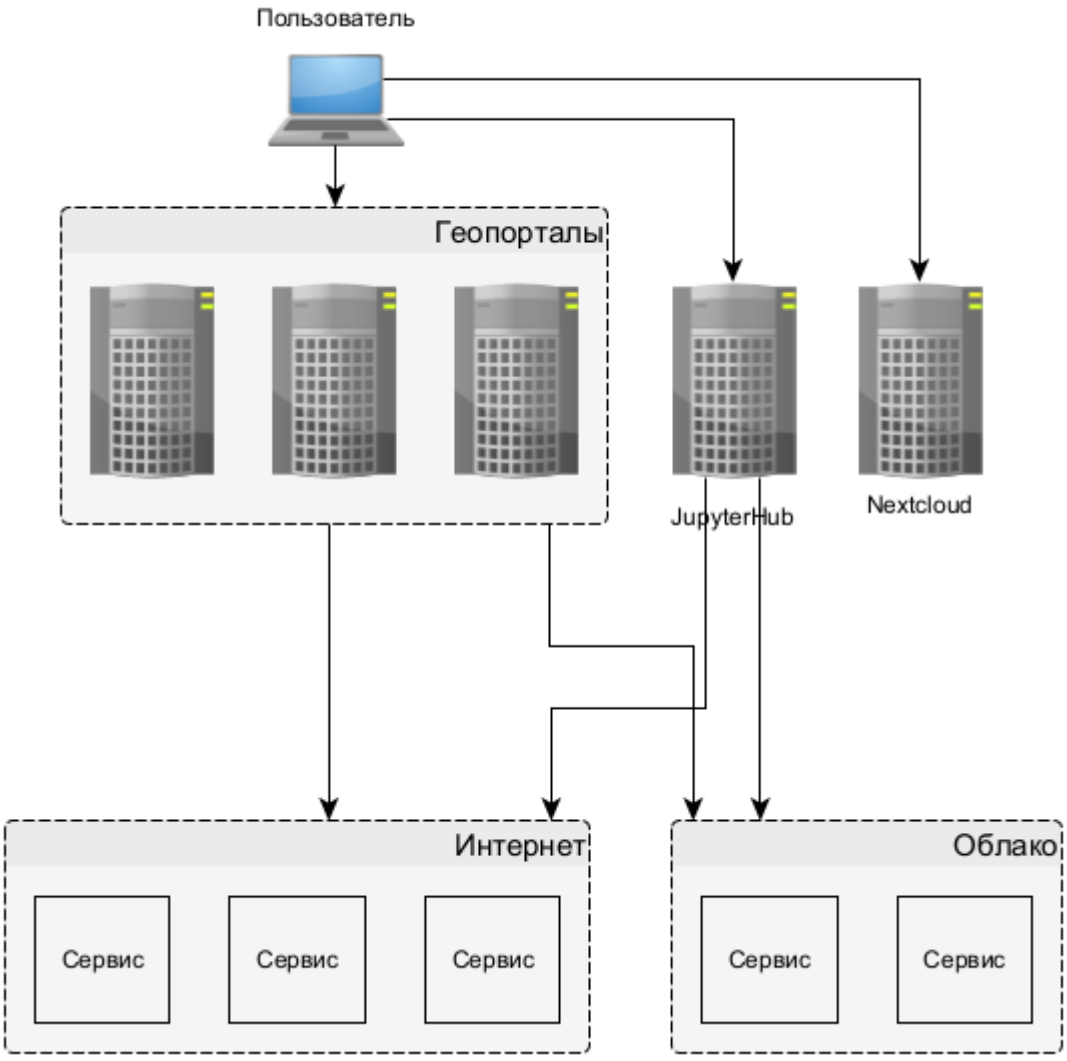


Рисунок 3.1 – Архитектура облачной среды

Типовой геопортал реализует функции ввода, редактирования, отображения и анализа данных с помощью сервисов. Геопортал является точкой входа для поиска и использования сервисов данных и геообработки. Для упрощения работы пользователей для каждой предметной области создается отдельный геопортал, который предоставляет быстрый доступ к наиболее часто используемым сервисам.

JupyterHub – многопользовательский сервер, предназначенный для анализа данных в среде Jupyter notebook. Позволяет проводить интерактивное программирование и создавать отчеты-ноутбуки. Предоставляет доступ (рисунок 3.2) к вычислительным узлам с графическими ускорителями и современным методам обработки данных. В JupyterHub подключается пользовательская директория системы хранения данных, предоставляются базовые пространственные данные, космоснимки и т. д.

Nextcloud – свободно распространяемая программная система (https://nextcloud.com/) для управления файловой системой хранения данных. При работе с сервисами геообработки часто приходится работать с данными в виде файлов. Выделяется каждому пользователю в системе хранения данных директория. Nextcloud предоставляет функцию регистрации пользователей и является сервером авторизации пользователей на основе протокола OAuth 2.0 для всех компонентов среды.

«Сервисы» — методы обработки данных, развернутые на вычислительных узлах облачной среды или в Интернет. Методы должны быть реализованы в соответствии со стандартом WPS. В рамках среды предоставляются виртуальные машины с предустановленным программным обеспечением для реализации WPS сервисов.

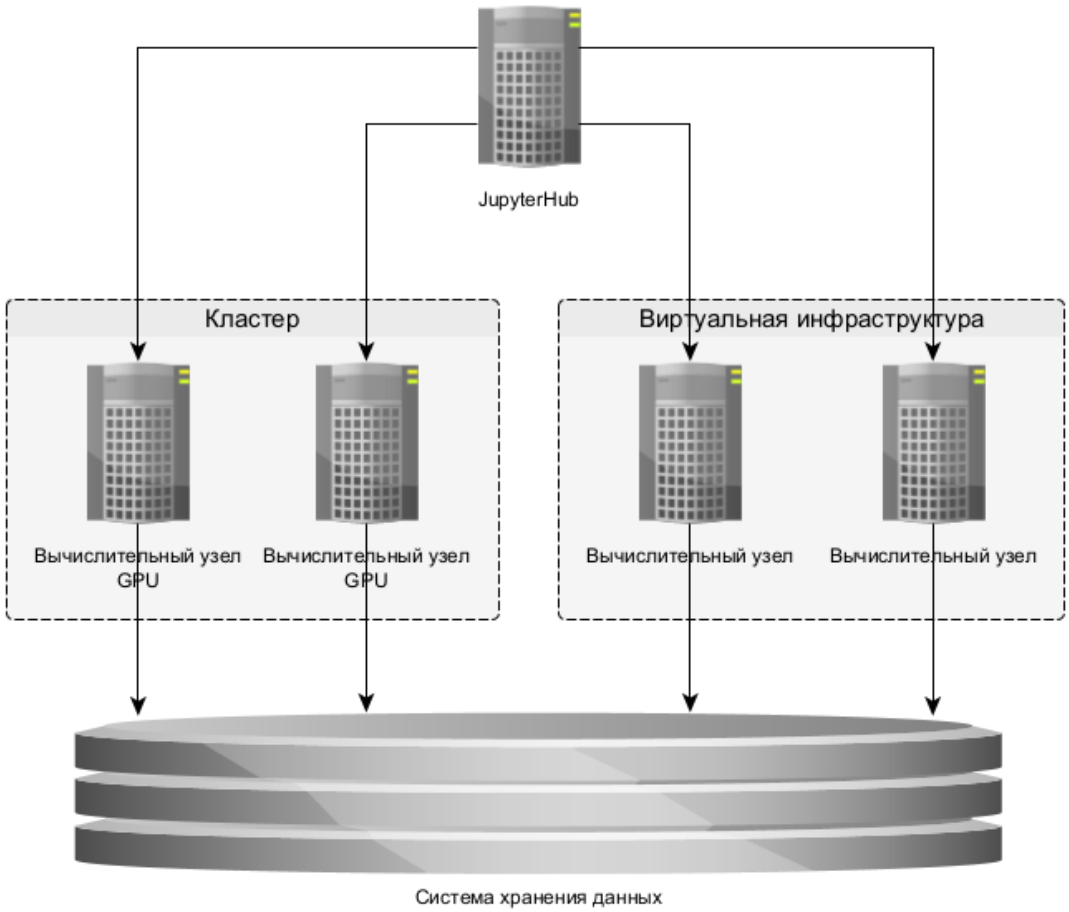


Рисунок 3.2 – JupyterHub

Пользователь использует Nextcloud для работы с файлами, геопортал – для сбора/публикации данных и применения существующих сервисов, а JupyterHub – для разработки новых методов. Далее рассмотрим некоторые компоненты подробнее.

3.2. Типовой геопортал

Разработан типовой геопортал на основе программной платформы Node.js, реализующий общие для информационных систем сбора и анализа данных функции на основе WPS сервисов. Исходный код геопортала выложен на Gitlab (https://gitlab.com/fromul/geoservices) под лицензией MIT. Геопортал обеспечивает единую точку доступа ко всем ресурсам и сервисам обработки. Его основными функциями являются формирование готовых наборов карт, загрузка и выгрузка данных, регламентация доступа к ним, создание собственных данных, встроенный картографический веб-клиент для просмотра картографических веб-служб, применение сервисов обработки данных и т. д. В основном геопортал воспринимают

как средство веб-картографирования, в том числе – с использованием спутниковых снимков. В разрабатываемой среде под геопорталом понимается средство не только для визуализации, но и для сбора и анализа данных, т. е. геопортал является полноценной информационной системой (ИС), автоматизирующей работу исследователей, с развитыми функциями пространственного анализа.

Создание единого и общего (универсального) для всех исследователей геопортала не оправдалось на практике. Наличие множества данных или таблиц, не относящихся к сфере деятельности исследователя, значительно усложняет внедрение и использование геопортала. ИС должна быть простой для пользователя, т. е. при входе на геопортал он должен сразу получить все необходимое для своей работы. Чтобы это реализовать, необходимо создавать геопорталы, ориентированные на конкретных предметных специалистов. Пользовательский интерфейс геопорталов должен упрощать работу пользователей и, желательно, соответствовать их ожиданиям (привычкам). Пользовательский интерфейс типового геопортала может быть модифицирован под требования пользователей с помощью шаблонизатора PUG.

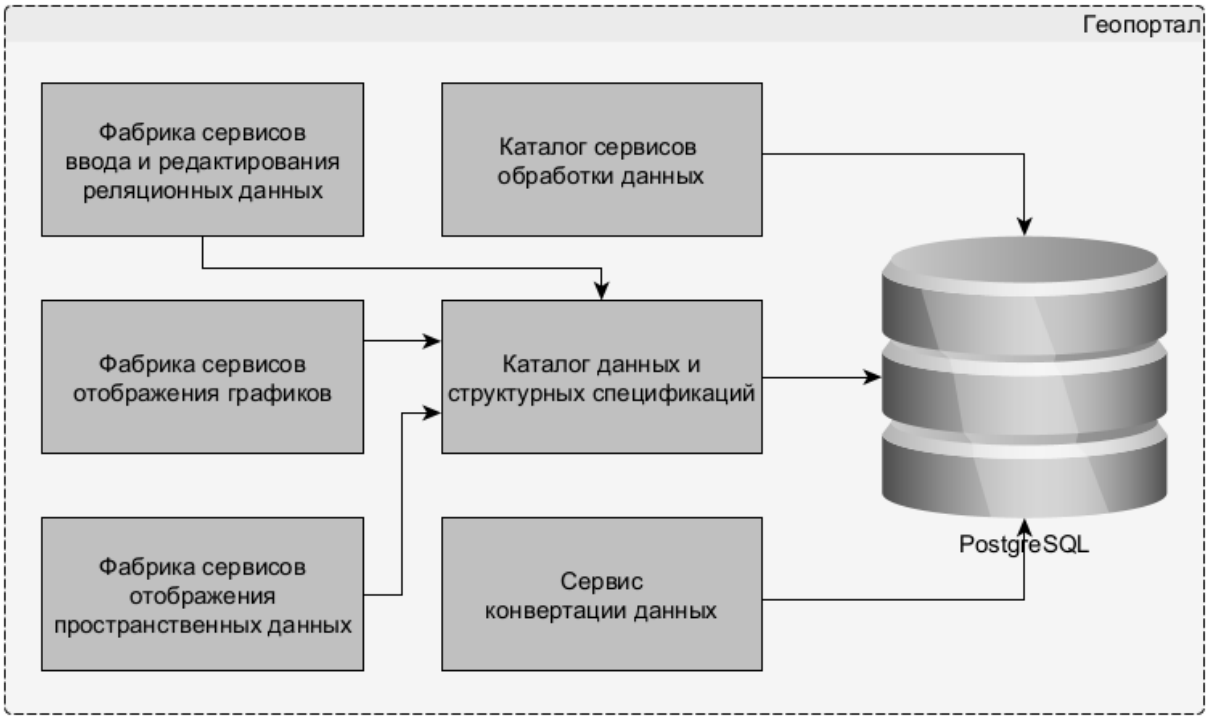


Рисунок 3.3 – Структура геопортала

На рисунке 3.3 представлена структура геопортала.

Компоненты «Фабрика сервисов ввода и редактирования реляционных данных», «Фабрика сервисов отображения графиков», «Фабрика сервисов отображения пространственных данных» предназначены для создания новых сервисов, каждый из которых обладает программным интерфейсом и готов стать частью композиции сервисов. Созданные сервисы автоматически регистрируются в соответствующих каталогах.

«Каталог сервисов обработки данных» предназначен для хранения метаданных сервисов. Он обеспечивает поиск сервисов и их выполнение.

«Каталог данных и структурных спецификаций» предназначен для хранения метаданных сервисов для таблиц. Каталог обеспечивает поиск данных.

«Сервис конвертации данных» предназначен для преобразования реляционных данных с одной структуры к другой с функциями нормализации данных. Позволяет загрузить данные из разных форматов: CVS, SHAPE, MIF и т. д.

«PostgreSQL» – свободно распространяемая СУБД [13]. Для обработки пространственных данных в СУБД установлено расширение PostGIS. Каждый пользователь геопортала может создавать таблицы. СУБД может функционировать на каждой виртуальной машине геопортала или на общей СУБД.

3.3. Фабрика сервисов ввода и редактирования реляционных данных

Фабрика сервисов ввода и редактирования реляционных данных (зарегистрирована под названием «Фарамант») предназначена для создания таблиц и сервисов работы с ними. Сервисы обеспечивают пользовательский интерфейс для ввода и редактирования разных по структуре пользовательских таблиц и реализуют добавление, чтение, изменение и удаление записей таблицы (CRUD) с поддержкой некоторых сложных типов данных, встречающихся в исследованиях. Сервисы предоставляют программный интерфейс для обеспечения композиции с другими сервисами. Это дает возможность интегрировать данные сервисов в информационную среду. Процесс создания сервисов тоже автоматизирован, так как таблицы могут использоваться для автоматической публикации результатов

вычислений. Соответственно, геопортал предоставляет сервис для создания новых сервисов ввода и редактирования реляционных данных.

Работа сервисов основывается на модели таблицы. Модели таблиц упорядочиваются в виде иерархий и реализуются механизмы наследования и полиморфизма в терминах объектно-ориентированного подхода. Применение модели таблицы позволяет:

1) создавать таблицы на основе готовых и устоявшихся моделей предметной области;
2) унифицировать по структуре таблицы разных пользователей, содержащие общую модель или унаследованные от нее другие модели;
3) создавать WPS-сервисы не к конкретным таблицам, а к моделям, т. е. применять их к любым таблицам, созданным по данной модели;
4) проводить анализ и создавать отчеты по пользовательским таблицам, созданным на основе общей модели.

Большая часть данных является пространственно привязанной. Создаваемые сервисы позволяют работать с пространственными атрибутами: вводить, отображать на карте, выполнять пространственные операции и т. д. Большинство сервисов обработки пространственных данных поддерживают стандарт WPS. Табличные данные часто используются в качестве входных данных для сервисов. Реализована поддержка этого стандарта разработанными сервисами, что значительно повышает интероперабельность сервисов, возможность создания композиций сервисов.

Большинство существующих CRUD систем реализует работу в один момент времени только с одной записью таблицы. В то же время многие сущности могут быть представлены в реляционной базе данных в нормализованном виде с помощью множества записей, находящихся в разных отношениях. Например, образец гербария представляет дерево записей, где в корне – информация о месте, времени находки, а ветками являются виды растений, которые были найдены на этом месте. Информацию о таких сущностях удобнее для пользователя отображать и редактировать на одной форме. Далее такие сущности будем называть документами. Поэтому реализация CRUD геопортала проводит операции не на уровне записи, а на

уровне документа. На программном уровне редактирование документа проводится в рамках одной транзакции, что позволяет улучшить контроль корректности и непротиворечивости данных.

В общем, весь набор функций сервисов представлен на рисунке 3.4 с помощью диаграммы использования.

Рисунок 3.4 – UML, диаграмма использования сервисов ввода и редактирования реляционных данных

Наиболее подходящей схемой для решения задачи ввода и редактирования реляционных данных является Model-View-Controller (MVC, «Модель-Представление-Контроллер») – разделение данных приложения и управляющей логики на три отдельных компонента: модель, представление и контроллер, для того чтобы модификация каждого компонента могла осуществляться независимо. Это позволяет использовать один и тот же программный код представления и контроллера для разных моделей.

Архитектура компонента (рисунок 3.5) – клиент-сервер, типичная для Web-технологий. Клиентские компоненты работают на основе браузера.

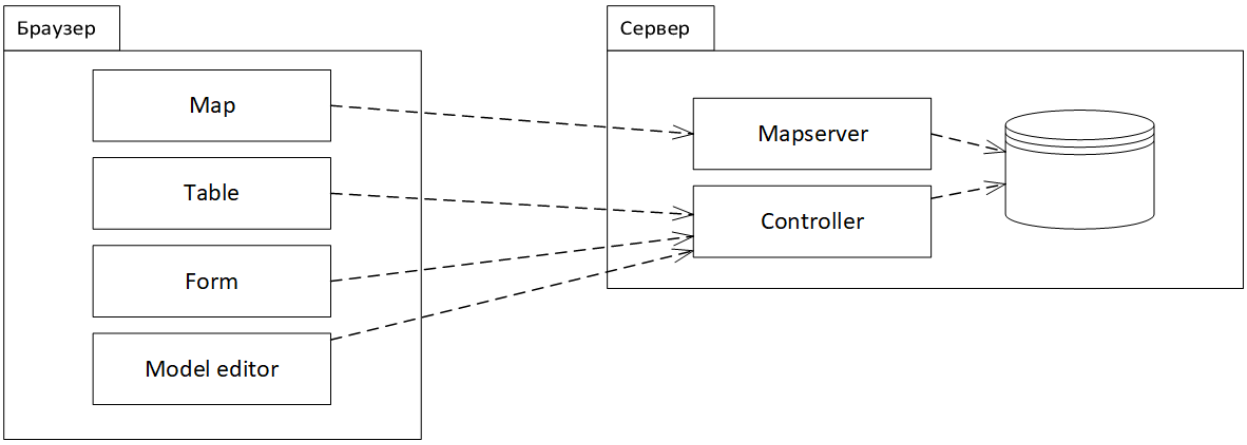


Рисунок 3.5 – Архитектура системы «Фарамант»

«PostgreSQL» – СУБД с установленным расширением PostGIS для обработки пространственных данных. Несмотря на древовидную структуру документа, используется реляционная база данных, что позволяет использовать язык запросов SQL.

«Controller» – модуль, который в соответствии с заданными политиками безопасности на основе структурных спецификаций реализует создание таблиц, формирует запросы к таблицам: добавление, изменение, удаление записей таблицы и т. д.

«Mapserver» – проводит на стороне сервера генерацию изображений слоев карт для пространственных атрибутов в соответствии со стандартом WMS.

Пользовательский интерфейс (Представление) в браузере реализуется компонентами:

«Table» – модуль, формирующий пользовательский интерфейс отображения множества документов в виде таблицы, по всем атрибутам можно выполнять сортировку и фильтрацию данных;

«Form» – модуль, предназначенный для генерации пользовательского интерфейса, обеспечивающий редактирование таблицы на основе её модели;

«Map» – модуль, отвечающий за формирование пользовательского интерфейса для отображения карты на основе библиотеки с открытым исходным кодом (Leaflet);

«Model editor» – модуль, предназначенный для создания и редактирования модели таблицы.

Модель таблицы

Модель описывает используемую структуру документа и пользовательский интерфейс и хранится в формате JSON. Структуру документа можно представить как множество реляционных кортежей, представленных в виде иерархии (рисунок 3.6). В корне иерархии находится всегда один главный кортеж.

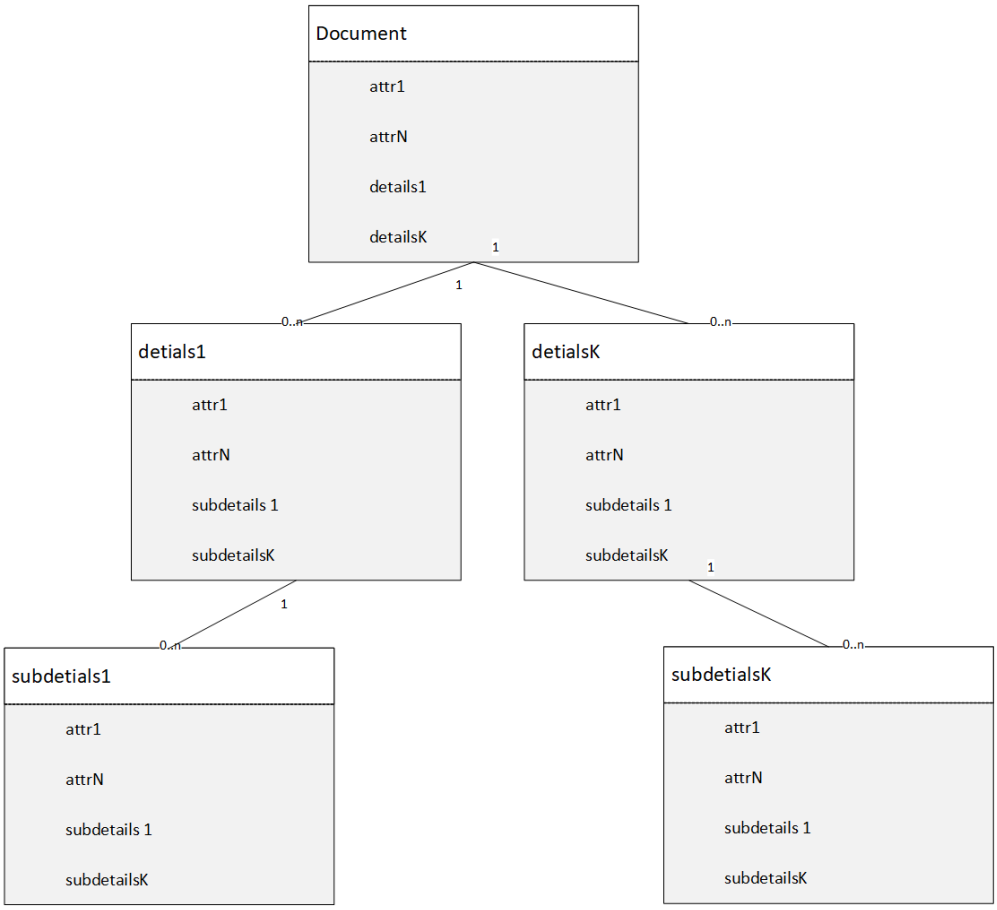


Рисунок 3.6 – Структура документа

Каждый кортеж имеет имя, заголовок и множество атрибутов. Каждый атрибут в свою очередь специфицируется названием, именем в базе данных, типом данных, единицами измерения (для числовых данных), элементом управления и его свойствами. Атрибут специального типа details задает отношение один ко многим. С помощью этого атрибута строится иерархия документа.

Элементы управления реализуют специфичные методы, необходимые для формирования пользовательского интерфейса добавления, редактирования и

отображения данных атрибута. Например, с помощью элементов управления реализуется взаимодействие с картой, таблицами справочников и т. д. Свойства элемента управления позволяют настраивать пользовательский интерфейс в зависимости от характеристик данных, например, единицы измерения для числовых данных или определять тип пространственных данных. Разработано более двадцати различных элементов управления, позволяющих работать со стандартными типами данных: number, string, date, boolean и т. д. Рассмотрим подробнее элементы управления. Каждый элемент управления реализует методы для ввода, отображения, фильтрации данных. Набор элементов управления является расширяемым. Достаточно унаследовать базовый класс Widget (рисунок 3.7). Метод assign() предназначен для генерации пользовательского интерфейса для ввода и редактирования данных атрибута. Метод getUserVal() предназначен для формирования строки со значением для отображения пользователю. Например, если это атрибут является ссылкой на другую таблицу (reference key), то пользователю отображается строковое значение, полученное из ссылаемой таблицы. Методы viewPropForm() и getProp() предназначены для формирования специфичных свойств, необходимых для реализации элементов управления. Например, в свойствах атрибута структурной спецификации может содержаться список возможных значений атрибута или имя таблицы справочника. Каждому элементу управления установлен в соответствие тип данных СУБД PostgreSQL.

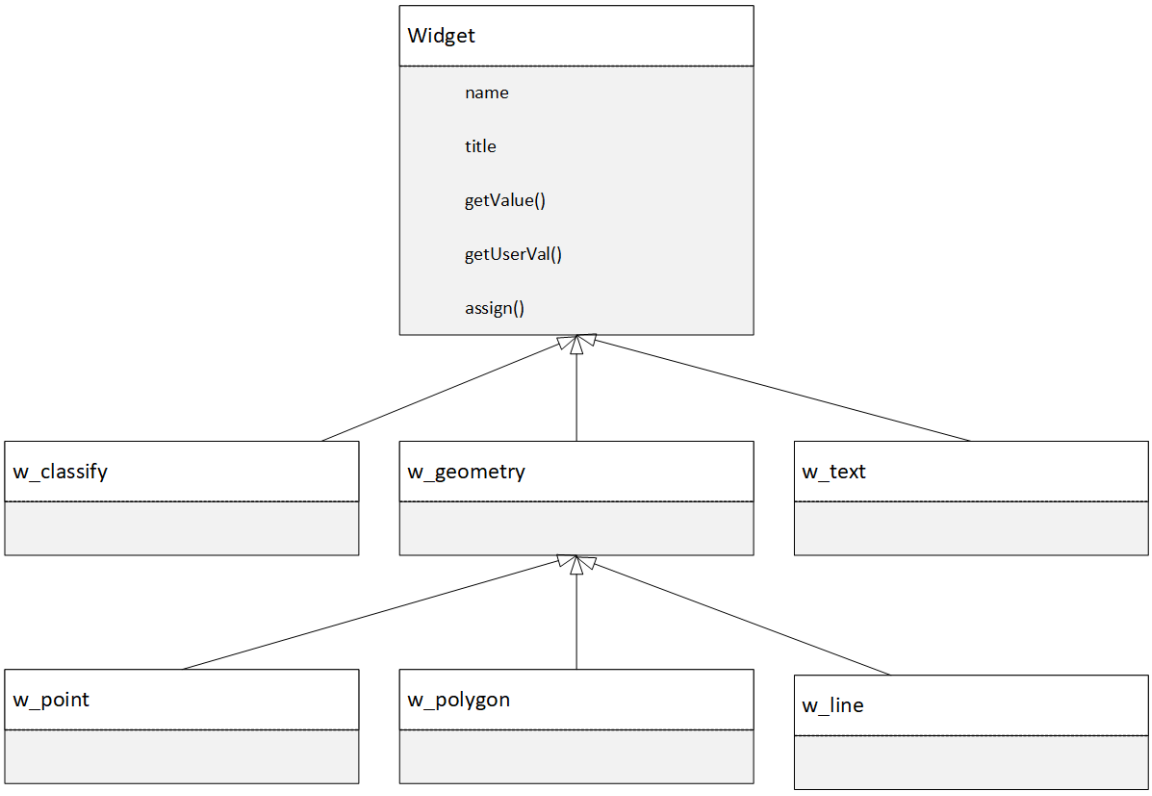


Рисунок 3.7 – Диаграмма классов компонентов управления

Использование элементов управления позволяет пользователю без программирования создавать гибкий и удобный пользовательский интерфейс для широкого класса реляционных таблиц. Рассмотрим реализации некоторых из них:

W_Number, W_Text – элементы управления для числовых и текстовых данных. Проводится проверка на ввод числовых данных.

W_Point, W_Line, W_Polygon – эти элементы управления предназначены для работы с пространственными данными и позволяют вводить, изменять координаты точечных, линейных и площадных объектов. При наличии у таблицы атрибутов, использующих перечисленные элементы управления, создаются соответствующие слои на карте. Ввод и редактирование данных можно осуществлять в режимах: последовательно указывая координаты всех точек пространственного примитива; с помощью мыши.

W_Classify – для ввода атрибутов, ссылающихся на таблицы без иерархической зависимости. Данный элемент управления позволяет использовать в качестве справочника любую таблицу, зарегистрированную в каталоге. Если в

таблице справочника имеется атрибут с пространственными данными, то создается соответствующий слой на карте. В рамках системы подготовлен ряд таблиц, содержащих унифицированные классификаторы. Например, международная классификация болезней МКБ-10, международный указатель научных названий растений (The International Plant Names Index, IPNI), таксономический классификатор биологического разнообразия животных России (А.Л. Лобанов, Зоологический институт РАН) и т. д.

W_Details – предназначен для создания множества подчиненных кортежей. На форме ввода элемент управления отображается в виде дочерней таблицы.

Ввод и редактирование данных документа осуществляется в ячейках таблицы или на сгенерированной форме с использованием указанных в модели элементов управления. Для пространственных атрибутов дополнительно создаются картографические слои для отображения данных на карте. Каждая таблица должна обладать удобным пользовательским интерфейсом, который можно настроить под требования пользователя. Модель документа содержит шаблоны форм ввода и печати документа.

Создание сервисов ввода и редактирования

Модели документов с метаданными (авторы, дата обновления и т. д.) хранятся в каталоге таблиц. Для создания таблицы пользователю необходимо создать модель в виде структурных спецификаций с помощью редактора модели либо использовать готовую. На основе модели таблицы создаются таблицы в СУБД PostgreSQL, генерируется пользовательский интерфейс и определяется логика работы сервисов.

Разработан REST интерфейс для создания сервисов:

http[s]://hostname:port/dataset/<имя метода>,

savestructure – метод создает сервис и изменяет модель таблицы для существующего сервиса. Метод является POST запросом и в теле запроса передает единственный параметр tablejson, который является моделью таблицы. Если идентификатор сервиса задан в модели, то производится изменение сервиса, иначе

– создание нового. Результат выполнения запроса приходит в формате JSON. Метод возвращает модель таблицы, в которой указан адрес нового сервиса.

Во всех методах ниже имеется параметр идентификатор сервиса dataset_id.

- drop – метод производит удаление сервиса. Сервис может быть удален только его создателем.
- clear – метод производит удаление всех данных сервиса. Метод может быть вызван только создателем сервиса.
- user/list – получение списка пользователей сервиса.
- user/add – добавление пользователей. Параметры userid – идентификатор пользователя, accesstype – тип доступа.
- user/delete – удаление пользователя. Параметры userid – идентификатор пользователя.

Разработанный набор REST методов позволяет программно создать сервисы ввода и редактирования, готовые к работе.

REST интерфейс сервисов

Каждый документ является JSON объектом, в котором значения атрибутов заданы с помощью пар <имя атрибута>:<значение>. Все методы работают на основе GET и POST запросов. Все запросы реализуют механизм Cross-Origin Resource Sharing (CORS), чтобы дать возможность браузеру пользователя получать разрешения на доступ к выбранным сервисам на источнике (домене), отличном от того, что сайт использует в данный момент. Это позволяет использовать REST интерфейс с любых сайтов. Получение документов производится с помощью следующего метода:

http[s]://hostname:port/dataset/list?f=<tableId>&count_rows=true &iDisplayStart=0&iDisplayLength=10[&<sort>][&<filter>]

Параметры метода:

- tableId – идентификатор таблицы;
- count_rows – подсчет количество записей;

- iDisplayStart, iDisplayLength – необходимы для организации постраничного вывода, обозначают порядковый номер начального документа и количество документов в запросе;
- <filter> – используется для фильтрации данных, задается как множество пар вида: f_<fieldname>=<value>;
- <sort> – используется для сортировки данных, задается как множество пар вида: s_<fieldname>=<order>, порядок следования пар используется для задания приоритета сортировки.

Результат выполнения запроса приходит в формате JSON.


```
▼ object {4}
    sEcho : null
    iTotalRecords : 25767
    iTotalDisplayRecords : 25767
  ▼ aaData [10]
    ▼ 0 {14}
        id  : 40170
        path : [\"/LANDSAT_8/2020_11/LO81320232020311LGN00/LO08_L1TP_132023_20201106_20201111_01_
               T1_B11.TIF\"]
        sensor : LANDSAT_8
        borders : MULTIPOLYGON(((105.96857 54.18695,109.58659 54.18695,109.58659
                  51.9852,105.96857 51.9852,105.96857 54.18695)))
        cloudiness : 75.8
        exists : null
        url : [\"http://storage.googleapis.com/gcp-public-data-
              landsat/LO08/01/132/023/LO08_L1TP_132023_20201106_20201111_01_T1/LO08_L1TP_132023_
              20201106_20201111_01_T1_B11.TIF\"]
        idate : 2020-11-06 00:00:00
        classname : null
        is_deleted : false
        created_by : 50f7a1d80d58140037000006
        edited_by : null
        edited_on : 2022-02-09T17:40:08.512Z
        created_on : 2022-02-10 01:40:08.512196+08
    ▶ 1 {14}
    ▶ 2 {14}
    ▶ 3 {14}
```


Массив aaData содержит список документов. Атрибут iTotalDisplayRecord содержит общее количество документов.

Добавление документа производится с помощью следующего метода:

http[s]://hostname:port/dataset/add?f=<tableId>&document=<document>

<document> – это объект документ или массив документов. Если это массив, то одной операцией можно добавить несколько документов.

Редактирование документа производится с помощью следующего метода:

http[s]://hostname:port/dataset/update?f=<tableId>&document=<document>&f_id=<f_id>

<document> – это объект документ. Документ идентифицируется атрибутом f_id, который определяет значение первичного ключа.

Удаление документа производится с помощью следующего метода:

http[s]://hostname:port/dataset/delete?f=<tableId>&f_id=<f_id>

Документ идентифицируется атрибутом f_id.

Интеграция с WPS сервисами

Для интеграции сервисов данных с WPS сервисами разработаны сервисы конвертации table2shp и table2json, которые отличаются форматом вывода данных таблиц. table2shp сохраняет данные в формате SHP. Данные, которые не содержат пространственные атрибуты, нельзя сохранить в формате SHP. Для них используется сервис table2json. Оба этих сервиса имеют одинаковый набор параметров:

1) идентификатор таблицы;
2) правила фильтрации данных и группировки.

С помощью этих сервисов можно получить данные за заданный период или с применением групповых функций.

Иерархия таблиц

В разных областях исследований ученые часто объединяют усилия для сбора данных. Тем не менее, у каждого ученого свои цели исследований и, соответственно, требуется собирать данные, немного отличные по набору атрибутов от общего набора. В этом случае обычно ученые собирают данные в отдельных таблицах. Сбор данных в разных таблицах приводит к сложности их совместного использования, так

как требует сложной процедуры объединения. В рамках сервисов предлагается использовать механизм наследования таблиц PostgreSQL. С помощью этого механизма создается иерархия таблиц (рисунок 3.8).

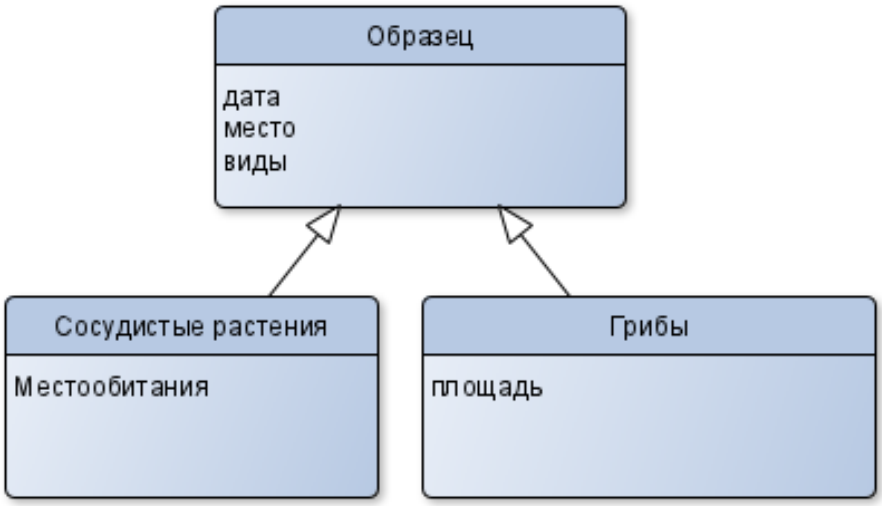


Рисунок 3.8 – Иерархия таблиц

Создается некая базовая таблица, например, «Образец», которая содержит общий для всех набор атрибутов. Создаются дочерние таблицы, например «Сосудистые растения» и «Грибы», которые наследуют все атрибуты базовой таблицы и дополнительно содержат специфичные для набора данных атрибуты. Для каждой таблицы создается своя модель документа. Разработан метод, позволяющий скопировать модель базовой таблицы, а затем ее модифицировать. Можно добавить новые атрибуты, удалить существующие, изменить свойства элементов управления и т. д.

При добавлении записи в дочерней таблице автоматически появляется запись в базовой таблице. При редактировании записи изменения автоматически отображаются в базовой таблице. Механизм наследования позволяет каждому пользователю видеть только свой набор данных. Базовая таблица собирает записи со всех пользователей по общему набору атрибутов.

Асинхронное вычисление значений атрибутов

В документе значения атрибутов могут автоматически вычисляться с помощью специальных выражений, которые могут использовать данные сервисов. Вычисление значений атрибутов происходит при наступлении следующих событий:

- создание формы документа;
- сохранение документа;
- дублирование документа.

Вычисление атрибутов документа производится на основании выражения на специальном языке. Первоначальная идея использовать язык JavaScript для задания выражений была отвергнута из-за потенциальной угрозы безопасности – встраивание вредоносного кода. Выполнение выражений должно полностью контролироваться. Для создания парсера использован Jison генератор, который создает код разбора выражений и их вычислений на языке JavaScript. Синтаксис выражений приводится в приложении Б. Выражения могут состоять из:

- арифметических операторов;
- оператора конкатенации строк;
- оператора получения значения из дерева документа jspath (аналог XPath для JSON);
- оператора получения значения из таблицы геопортала;
- оператора получения значения с помощью REST сервисов;
- логических констант true и false;
- скобок.

Выражения по умолчанию вычисляются на стороне сервера. Выражения, вычисляемые при создании или дублировании документа, интерпретируются на стороне браузера. Из-за наличия в выражениях оператора получения значения из таблицы геопортала и оператора получения значения из REST сервиса вычисление выражение проходит асинхронно.

Пользовательский интерфейс ввода и редактирования данных

Рисунок 3.9 – Пример редактирования данных на форме и в ячейке таблицы

Основываясь на модели, формируется интерфейс ввода и редактирования таблицы (рисунок 3.9). Пространственные атрибуты отображаются на карте. Ввод и редактирование данных осуществляется в таблице или на форме. Для каждого атрибута используется элемент управления, указанный пользователем. По всем атрибутам можно выполнять сортировку и фильтрацию.

Пользователь, создавший таблицу, автоматически назначается её владельцем. Для каждой таблицы владелец может настроить следующие права доступа:

- Просмотр и редактирование только владельцем. В этом случае только владелец может просматривать и изменять таблицу. При просмотре каталога таблиц другими пользователям таблица и данные не отображаются.
- Просмотр и редактирование определенными пользователями. Владелец может предоставить права доступа любому из зарегистрированных пользователей. Таблица в каталоге таблиц будет отображаться всем пользователям, а данные будут отображаться только указанным пользователям.
- Просмотр – всем. Редактирование – определенными пользователями. Все пользователи геопортала могут просматривать содержимое таблицы, редактирование разрешается только определенным пользователям.
- Просмотр и редактирование – всем пользователям. Просмотр и редактирование разрешается всем зарегистрированным пользователям.

Для каждой записи таблицы автоматически устанавливается владелец записи, дата создания записи, пользователь, изменивший запись, признак модерации. По умолчанию пользователь не может просматривать и редактировать записи таблицы.

Для каждого пользователя владелец может установить права доступа:

- Редактирование собственных записей. В этом случае пользователь может создавать, редактировать и просматривать свои записи в таблице.
- Просмотр записей других пользователей.
- Редактирование записей других пользователей.
- Модератор. Пользователю разрешается редактировать записи других пользователей и устанавливать признак, что запись прошла модерацию.

Если владелец не установил «Просмотр и редактирование только владельцем», то таблица отображается в каталоге таблиц. Любой зарегистрированный пользователь может отправить запрос владельцу на получение дополнительных прав.

Модуль разработки форм ввода

Генерация форм редактирования документов производится модулем «Form». В процессе генерации последовательно перебираются все атрибуты, указанные в модели. Для каждого атрибута используется соответствующий элемент управления для создания полей ввода на форме или в ячейке таблицы. Существуют два способа задания расположения элементов управления на форме:

- по умолчанию в порядке следования атрибутов в модели;
- на основе шаблона формы.

Шаблон формы – это HTML код с указанием положения вставки элементов управления для ввода атрибутов. Создается шаблон в специальном редакторе или в блокноте (рисунок 3.10). Указание положения элементов управления атрибутов осуществляется с помощью специальных тегов вида «#fieldname#», где fieldname это имя атрибута в БД, либо с помощью HTML контейнеров , у которых идентификатор совпадает с именем атрибута. Применение контейнеров позволяет управлять отображением заголовков полей.

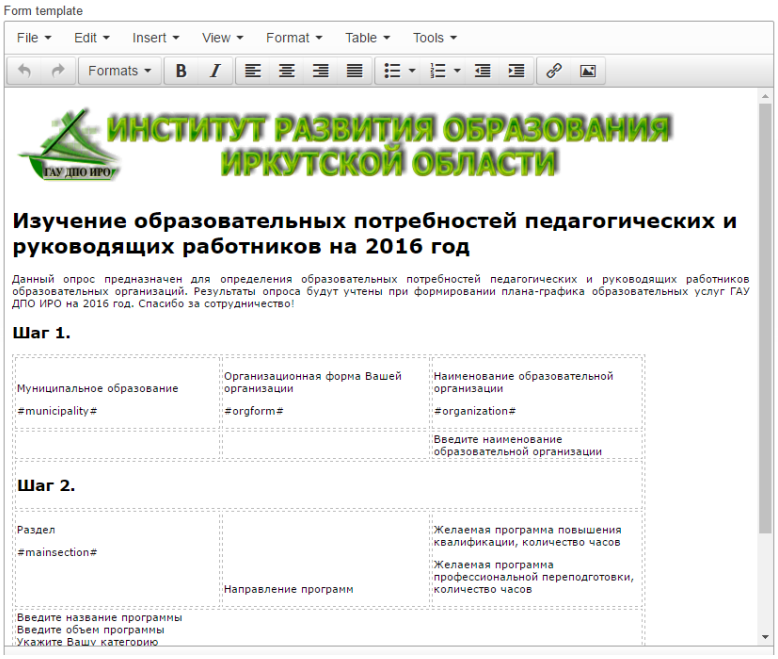


Рисунок 3.10 – Шаблон формы ввода анкеты формируется в WYSIWYG редакторе

На значения атрибутов документов часто налагается ряд ограничений, заданных предметной областью. Например, дата произошедшего события не может быть ранее текущей даты. Существуют более сложные ограничения, например, в адресе населенный пункт должен указываться из списка населенных пунктов, находящихся в выбранном регионе. Автоматическая проверка ограничений при вводе и редактировании данных может значительно улучшить корректность документов.

В рамках сервисов предлагается возможность задания зависимостей между элементами управления для ограничения множества возможных значений и определение условия видимости элементов управления. Реализация ограничений в виде отображения только возможных значений ускорит и упростит ввод данных. Например, в задаче сбора образовательных потребностей пользователь должен указать организацию, в которой он работает. Учитывая количество организаций, эффективнее чтобы пользователь указал муниципальное образование и организационную форму организации, и только затем выбрал организацию из отфильтрованного списка. Реализует зависимость элемент управления «classify». Для организации фильтрации текущего элемента управления «classify» необходимо

указать, по каким атрибутам таблицы справочника производить фильтрацию, и какие значения используются для фильтрации.

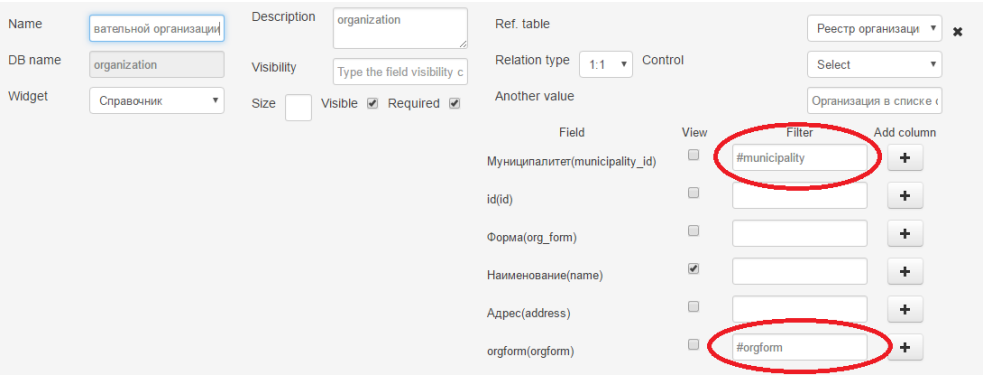


Рисунок 3.11 – Фильтрация данных на форме. Овалами выделены значения, по которым производится фильтрация соответствующих атрибутов

Если в качестве значения указывается «#fieldname», то значение для фильтрации автоматически берется из соответствующего атрибута. На рисунке 3.11 представлен атрибут, определяющий организацию. Для этого атрибута используется таблица справочник «Реестр организаций». Фильтрация реестра организаций производится по атрибутам «Муниципалитет» и «orgform». Соответственно, в выпадающем списке будут отображаться организации только из определенного муниципального образования и определенной организационной формы. Их идентификаторы автоматически берутся из соответствующих элементов управления. При изменении муниципального образования список организаций автоматически обновляется. Пока значения фильтрации не определены, т. е. пользователь не указал значения атрибутов, элемент управления недоступен для ввода.

Следующей возможностью управления логикой работы формы является управление видимостью элементов управления атрибутов. Например, при отсутствии соответствующей образовательной программы пользователь должен указать желаемое название. Для хранения программы обучения создан атрибут «program_name», для него используется элемент управления «Текст». В свойстве элемента управления можно задать условие видимости в поле «Visibility» (рисунок 3.12). В примере указано условие: section==-1 || program==-1 || retrain==-1, это

означает, что элемент управления отображается, если хотя бы в одном из атрибутов section, program, retrain указано «Другое».

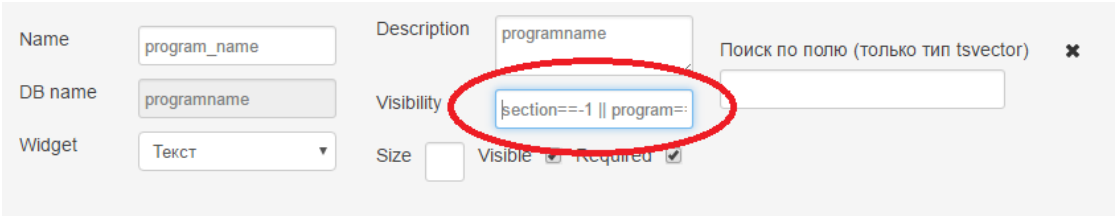


Рисунок 3.12 – Условие видимости элемента управления атрибута

Условие выражается в соответствии с синтаксисом языка JavaScript. Может содержать только имена атрибутов, константы и специальные символы (например, =). При изменении значения любого атрибута производится проверка условия видимости. Если в шаблоне формы для атрибута задан , то в этом случае можно скрыть атрибут вместе с заголовком.

Хранение истории документов

В рамках сервисов реализовано сохранение истории изменения документов. Владелец таблицы в настройках таблицы может указать необходимость сохранения истории. При выполнении CRUD команд с помощью REST сервисов документы передаются в виде JSON объектов, которые сериализуются в строки. В специальной таблице сохраняется строка, представляющая документ, дата и время изменения, действие и пользователь. Имеется специальная форма, на которой пользователь может посмотреть историю документа, и при необходимости вернуть его к предыдущему состоянию.

Анализ данных

В рамках создаваемых сервисов реализованы методы обработки реляционных данных, которые позволяют провести первичный анализ данных и отобразить результаты в виде таблицы и карты. Это методы фильтрации и сортировки данных, а также группировки данных. При реализации этих методов ставилась задача упростить для пользователя их использование.

Фильтрация табличных данных

Для фильтрации данных в рамках интерфейса отображения таблицы созданы фильтры для всех типов данных. Для каждого поля пользователь может определить свои правила фильтрации (рисунок 3.13).

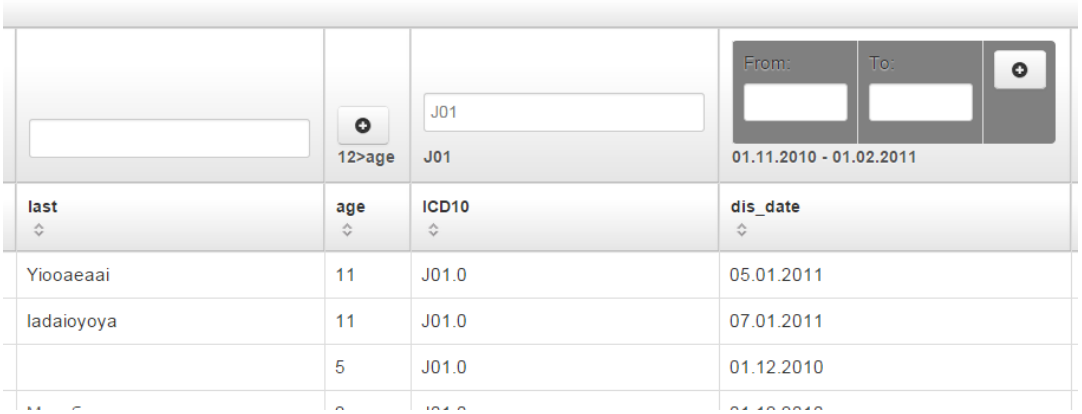


Рисунок 3.13 – Применение фильтров в таблице

Результаты фильтрации отображаются в таблице и на карте. Набор правил в контроллере транслируется в условие предложения WHERE SQL запроса. Все правила по полям объединяются по логическому «И». Внутри поля можно определить несколько значений фильтраций, которые для поля объединяются по логическому «ИЛИ». Так для строкового поля ICD10 можно установить несколько значений. Соответственно, останутся все записи, у которых имеются перечисленные значения. Для полей типа ДАТА пользователь может задать диапазоны дат. При этом одно из значений может опустить. Для числовых значений пользователь может указать любые знаки сравнения (рисунок 3.14).

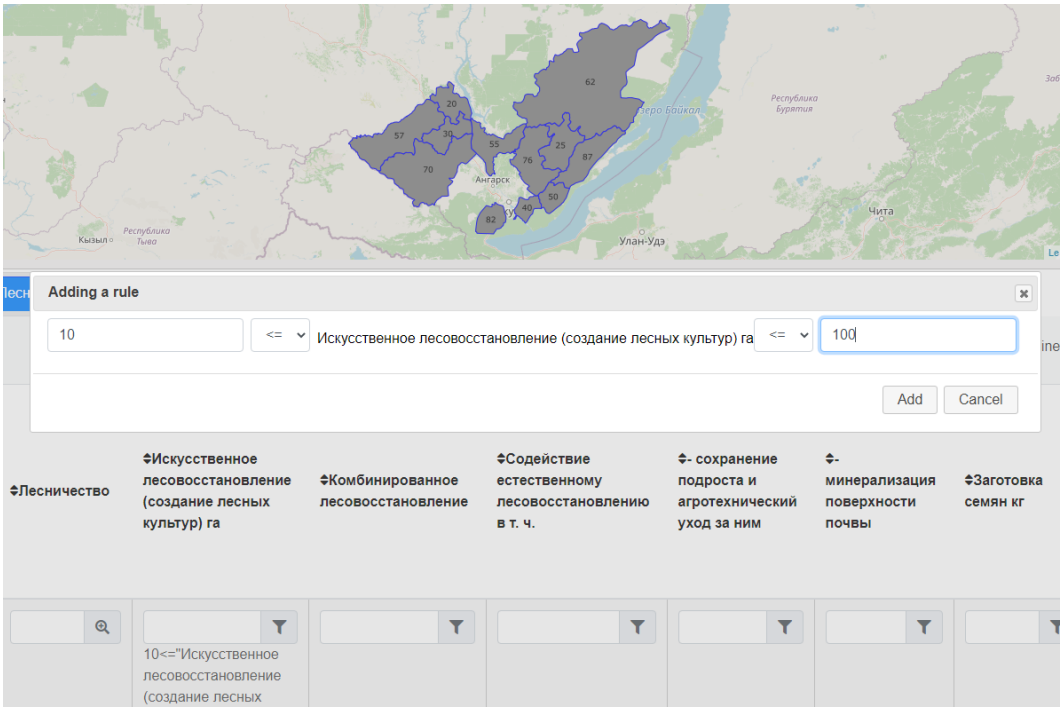


Рисунок 3.14 – Применение фильтра для числовых полей

Для пространственных атрибутов предусмотрена фильтрация с помощью указываемых пользователем полигональных объектов (рисунок 3.15) с применением одного из двух пространственных операторов:

- contain – остаются объекты, находящиеся внутри указанных полигональных объектов;
- within – остаются объекты, содержащие указанные полигональные объекты.

Пользователь может указать полигональные объекты следующими способами:

- на карте с помощью мыши;
- выбрать полигоны, задающие границы объектов административного деления;
- загрузить полигон в формате KML.

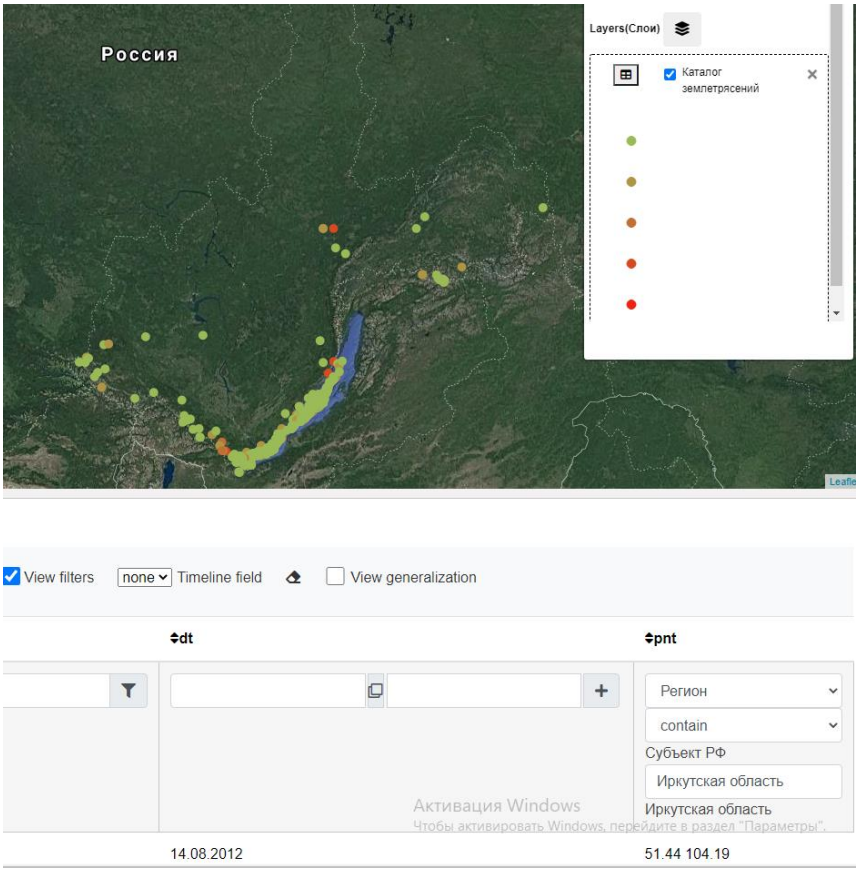


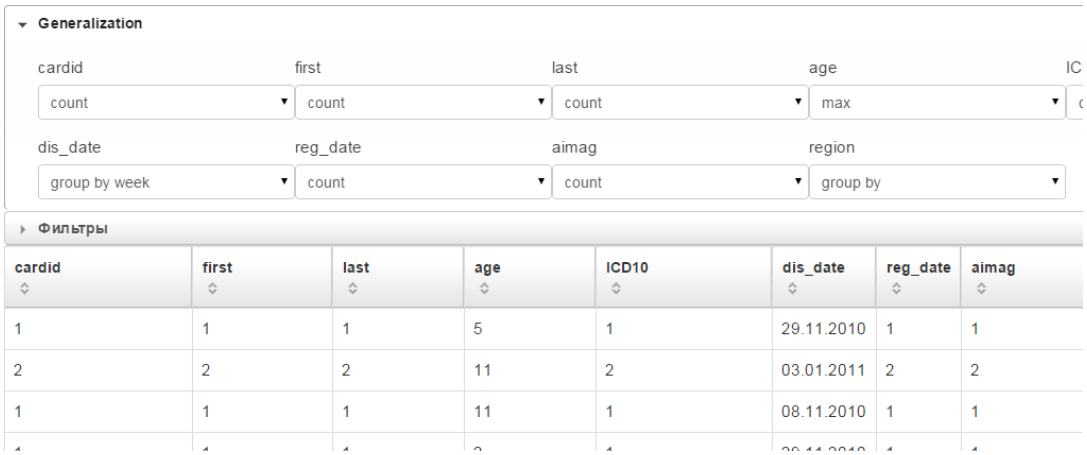
Рисунок 3.15 – Фильтрация по Иркутской области данных каталога землетрясений

Группировка данных с использованием подготовленных таблиц измерений

Группировка данных производится с помощью SQL и позволяет получить обобщенную информацию, т. е. средние, максимальные, минимальные значения атрибутов и количество записей по группам (рисунок 3.16). Например, для задачи сбора данных гербариев пользователь может подсчитать количество образцов по видам растений. В общем случае пользователю необходимо указать атрибуты, по которым производится группировка данных, и применяемые групповые функции. Группировка записей может производиться для всех типов полей по равенству значений. Результат группировки данных отображается в таблице и не влияет на карту. Так как имеется информация о типах данных атрибутов, то предусмотрена возможность специальной группировки для пространственных атрибутов, даты и времени (хранятся в одном атрибуте). Это обусловлено тем, что почти все данные имеют привязку ко времени и пространственное положение, например, в рамках административного деления. Для атрибутов типа ДАТА можно провести группировку записей (документов) по декадам, годам, кварталам, месяцам, неделям,

дням и часам. По пространственным атрибутам можно выполнить группировку по регионам и районам.

Для строковых полей и полей типа ДАТА можно выбрать групповую функцию подсчета количества записей в группе. Для числовых полей можно выбрать одну из функций, рассчитывающих максимальное, минимальное, среднее значение атрибута или количество по группам записей. При группировке данных используются заданные пользователем фильтры.



cardid	first	last	age	ICD10	dis_date	reg_date	aimag
1	1	1	5	1	29.11.2010	1	1
2	2	2	11	2	03.01.2011	2	2
1	1	1	11	1	08.11.2010	1	1

Рисунок 3.16 – Обобщение табличных данных

Развертывание геопортала в облачной инфраструктуре

В соответствии с необходимостью создания множества геопорталов, ориентированных на определенные предметные области, разработан алгоритм развертывания геопортала на основе облачной инфраструктуры. Алгоритм состоит из следующих шагов:

- развертывание виртуальной машины облачной среды на основе шаблонов (в терминологии VMware это «template», в Openstack это «image»), где заранее установлен и сконфигурирован типовой геопортал и присвоение DNS имени, что позволяет получить общедоступный сайт в сети Интернет;
- разработка таблиц для хранения данных пользователей на основе структурных спецификаций;
- модификация пользовательского интерфейса.

Шаблон содержит подключение к сетевому диску, содержащему базовые пространственные данные. Через сетевой диск производится передача данных

между сервисами, находящимися в облачной среде, что позволяет значительно ускорить передачу данных.

3.4. Фабрика сервисов отображения пространственных данных

Компонент «Фабрика сервисов отображения пространственных данных» обеспечивает создание WMS сервиса (рисунок 3.17).

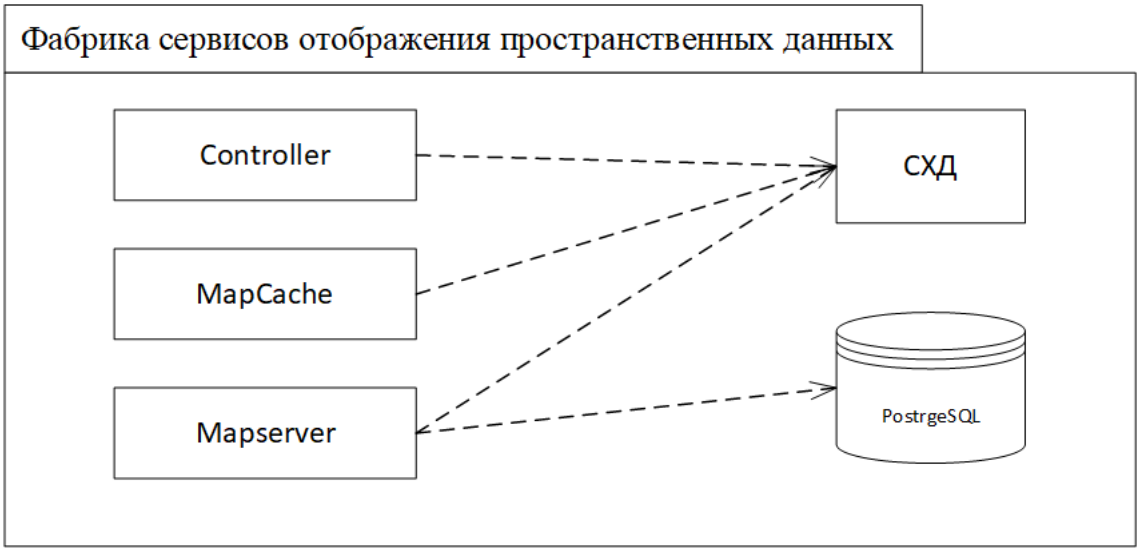


Рисунок 3.17 – Создание WMS сервиса

«Mapserver» – проводит на стороне сервера генерацию изображений слоев карт в соответствии со стандартом WMS. Выделен на отдельный вычислительный узел из-за необходимости использования 80 порта, так как многие организации блокируют доступ к остальным портам;

«MapCache» – проводит кэширование карт для получения скоростного доступа. Кэширование можно настроить для карт, которые медленно отображаются, например, из-за большого объема данных.

«Controller» – реализует интерфейс сервиса, подготавливает данные и формирует map файл, в котором содержатся настройки для Mapserver.

REST интерфейс «Controller»

Разработано два метода для отображения растровых и векторных данных в виде файлов и таблиц, созданных сервисами ввода и редактирования реляционных данных. Оба метода работают на основе GET и POST запросов. Создание WMS сервиса для таблицы производится с помощью следующего метода:

http[s]://hostname:port/map/createmap/list?f=<tableId>&style=<mapstyle>&fieldname=<fieldname>[&<filter>][&<grouping>]

Входные параметры метода:

- tableId – идентификатор таблицы;
- fieldname – имя атрибута, тип которого должен быть MULTIPOINT, MULTILINE или MULTIPOLYGON;
- style – стили отображения слоя;
- <filter> – используется для фильтрации данных, задается как множество пар вида: f_<fieldname>=<value>;
- <grouping> – используется для группировки данных, задается как множество пар вида: g_<fieldname>=<value>.

Возвращает сервис следующие параметры:

- wms_link – URL WMS сервера;
- wms_layer_name – имя слоя.

Создание слоя для растровых или векторных файлов производится с помощью следующего метода:

http[s]://hostname:port/map/createfilemap/list?source=<source>&style=<style>&fieldname=<fieldname>[&<filter>][&<grouping>]

Входные параметры метода:

- source – идентификатор таблицы;
- title – имя слоя;
- style – стили отображения слоя;
- keywords – ключевые слова;
- Возвращает сервис:
- wms_link – URL WMS сервера;
- wms_layer_name – имя слоя.

На рисунке 3.18 показано взаимодействие сервисов ввода и редактирования реляционных данных, представленных модулями Table и Map, с фабрикой сервисов

отображения пространственных данных. Модуль Map обращается к модулю «Controller», который создает WMS. Далее модуль Map обращается напрямую к Mapserver.

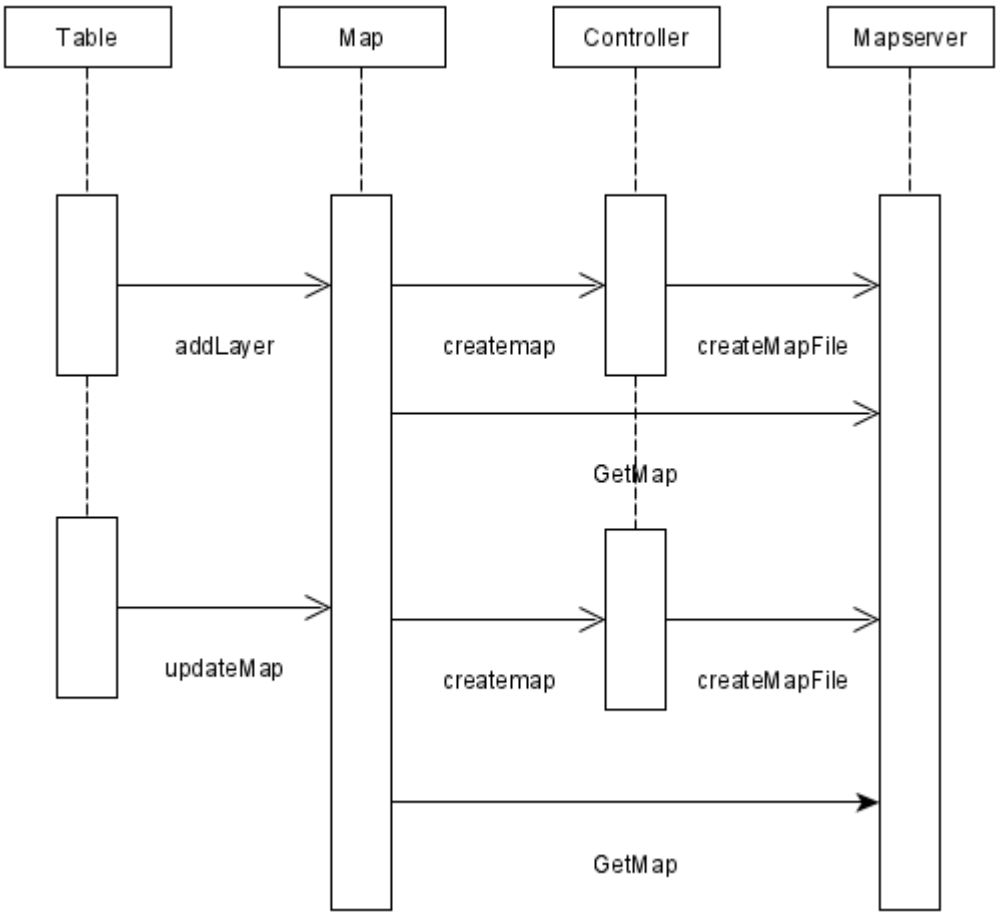


Рисунок 3.18 – Диаграмма последовательности отображения данных на карте

Mapserver производит генерацию растровых изображений, используя специальный файл настроек (MAP), в котором прописываются настройки доступа к данным, используемые проекции, стили отображения данных и т. д. В MAP файле сохраняется запрос к PostgreSQL на получение данных. Для каждого пользователя создается MAP файл с уникальным именем. Это позволяет разным пользователям одновременно просматривать данные с различными настройками фильтров и стилей отображения. Изменения фильтров или стилей отображения приводят к автоматическому изменению MAP файла.

Формирование легенды слоя

Формирование легенды производится в браузере с помощью специального редактора стилей (рисунок 3.19). Легенда формируется для каждого слоя (атрибута) отдельно и хранится в формате JSON. Формат хранения стилей SLD поддерживается только частично в связи с низкой скоростью отображения карт при его использовании. В соответствии со стандартом WMS стили указываются в виде ссылки в URL адресе. Поэтому Mapserver должен для каждого тайла (фрагмента карты) выполнить его загрузку на сервер, что значительно увеличивает время отрисовки слоя. Для отображения слоев с помощью Mapserver стили в формате JSON конвертируются в map файл.

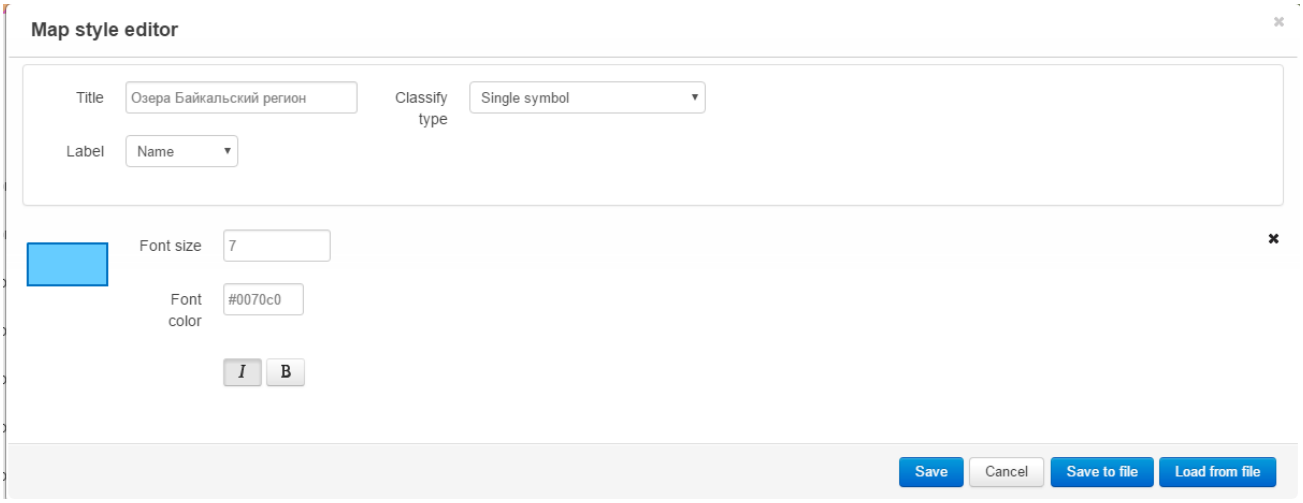


Рисунок 3.19 – Редактор стилей отображения слоев

Базовым элементом формирований стилей слоев является условный знак. Условный знак может быть векторным, растровым изображением, символом шрифта и диаграммой. Разработан редактор условного знака (рисунок 3.20).

Применение векторных изображений для точечных символов

Условный знак может состоять из нескольких слоев. Для каждого слоя можно выбрать один из существующих контуров. Можно указать видимость условного знака (opacity), цвет (color), размер (size), угол поворота (angle), смещение (dx, dy). Порядок слоев можно менять. В рамках геопортала имеется таблица контуров, в которой можно задать собственный контур в виде набора пар точек.

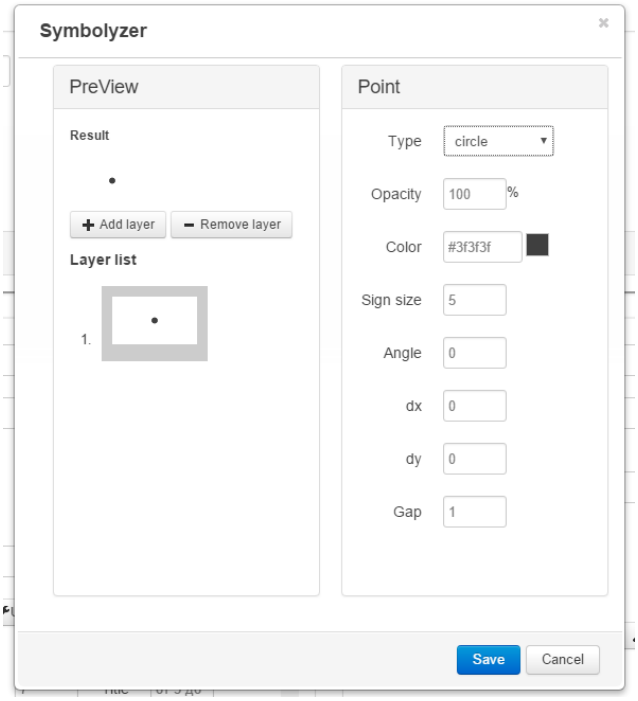


Рисунок 3.20 – Редактор условного знака

Пользователь может загрузить в геопортал растровое изображение и использовать его в качестве условного знака. Поддерживаются форматы: SVG, PNG.

Использование символов шрифта для условного знака

Во многих ГИС в качестве условного знака можно использовать шрифты TrueType. В рамках геопортала эта возможность также реализована. Пользователь может использовать символы шрифта ГИС MapInfo.

Формирование легенды на карты на основе классификации данных

Редактор стилей позволяет классифицировать данные и назначить способ отображения (рисунок 3.21). Каждый класс задается интервалами значений атрибутов и условным знаком. В редакторе стилей можно задать количество классов, список атрибутов, интервалы значений атрибутов и условный знак. Также можно воспользоваться специальными инструментами, которые автоматизируют назначение условных знаков для классов. Инструменты на основе заданного шаблона могут автоматически задать цвет или размер условного знака класса случайным образом либо в соответствии в соответствии с диапазоном значений атрибутов. Можно указать порядок значений атрибутов (сортировку).

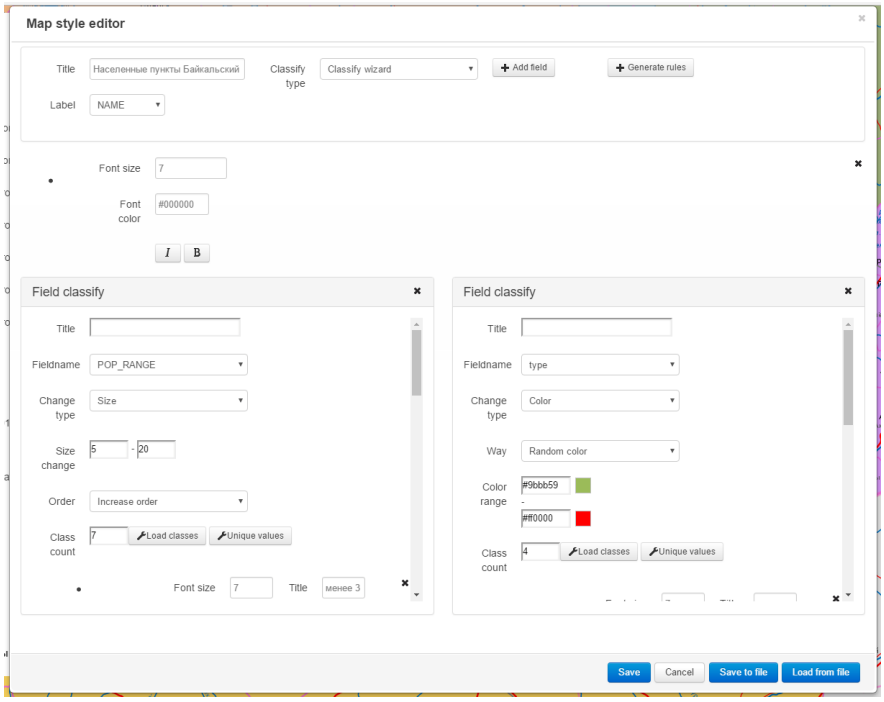


Рисунок 3.21 – Формирование стилей по двум атрибутам

Создание условного знака диаграмма

Поддерживается два типа диаграмм (рисунки 3.22, 3.23): круговые диаграммы и столбчатые. Пользователю необходимо указать, по каким атрибутам формируются диаграммы. Для каждого атрибута можно задать цвет и заголовок. Можно задать изменение размера диаграммы в соответствии со значением указанного атрибута.

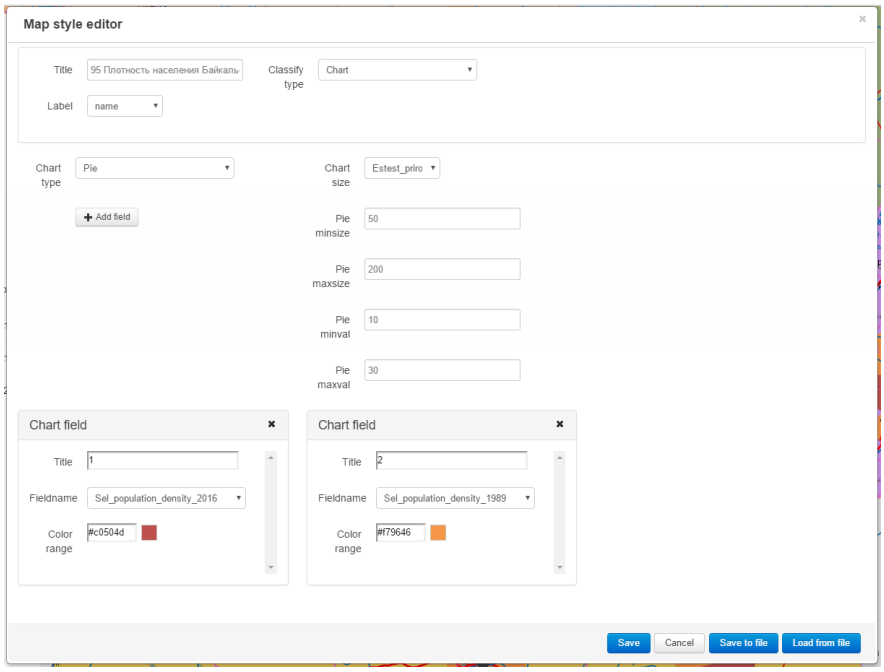


Рисунок 3.22 – Создание диаграмм

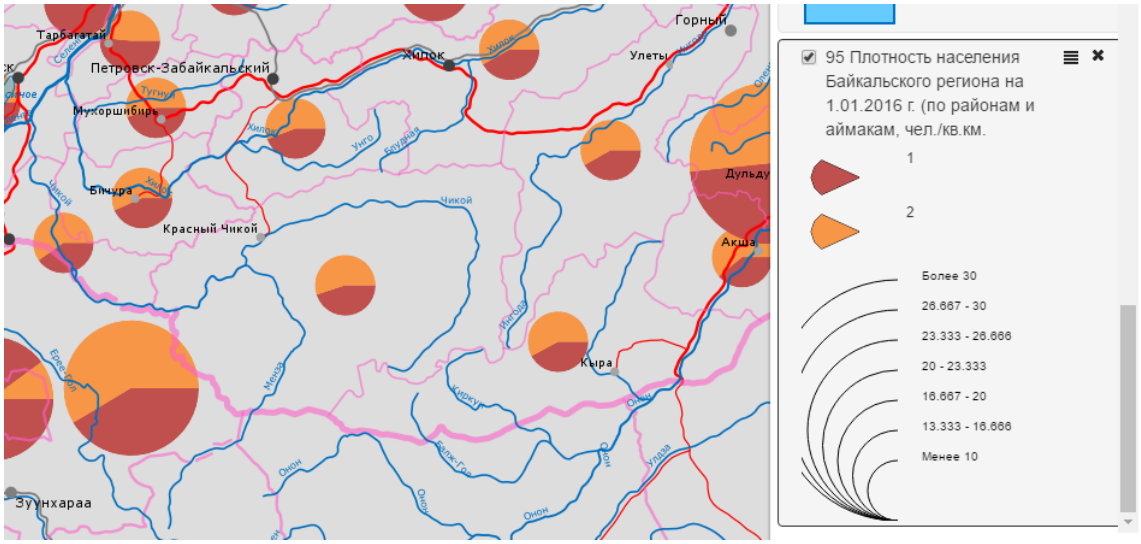


Рисунок 3.23 – Отображение диаграмм

Кэширование

Для ускорения отображения карт применяется MapCache, осуществляющий кэширование растровых тайлов (растровые изображения фрагментов карт, передаваемые сервером браузеру для отображения). MapCache предварительно отрисовывает все карты и сохраняет в файловой системе (кэширует). Далее вместо повторной отрисовки MapCache читает тайл из файловой системы и передает его браузеру. За счёт этого механизма процесс отрисовки значительно ускоряется. Кэширование карты – достаточно долгий процесс. Поэтому кэширование используется только для статических данных, т. е. которые не меняются.

3.5. Фабрика сервисов отображения диаграмм и графиков

Фабрика предназначена для создания сервисов отображения графиков на основе реляционных данных, предоставляемых сервисами ввода и редактирования (п. 3.3). В результате работы сервиса формируется ресурс с графиком, доступный по ссылке. Сервис имеет программный интерфейс, включенный в каталог сервисов. Программный интерфейс разработан таким образом, чтобы можно было создавать графики с заданным отображением для разных данных. Например, диаграмма отображения частоты событий по годам может создаваться для данных регистрации укусов клещей или находок инвазивных растений. Способ отображения данных задается моделью. Данные и модель отображения графика передаются сервису по

разным параметрам, что позволяет для одной модели использовать разные данные или для одной таблицы – разные способы отображения данных.

Создание сервиса отображения графика для таблицы производится с помощью следующего POST метода:

http[s]://hostname:port/wps/execute?id=<createDiagram>

Параметры метода:

- dataset_id – идентификатор таблицы;
- FilterValues – параметры фильтрации данных;
- groupdata – параметры группировки данных;
- model – JSON параметр, содержит модель отображения графика;
- name – название графика;
- keywords – ключевые слова;
- description – описание графика;
- id – идентификатор графика, указывается, если нужно поменять существующий график.

Параметры dataset_id, FilterValues, groupdata определяют исходные данные фабрики. Для создания модели разработан редактор (рисунок 3.24).

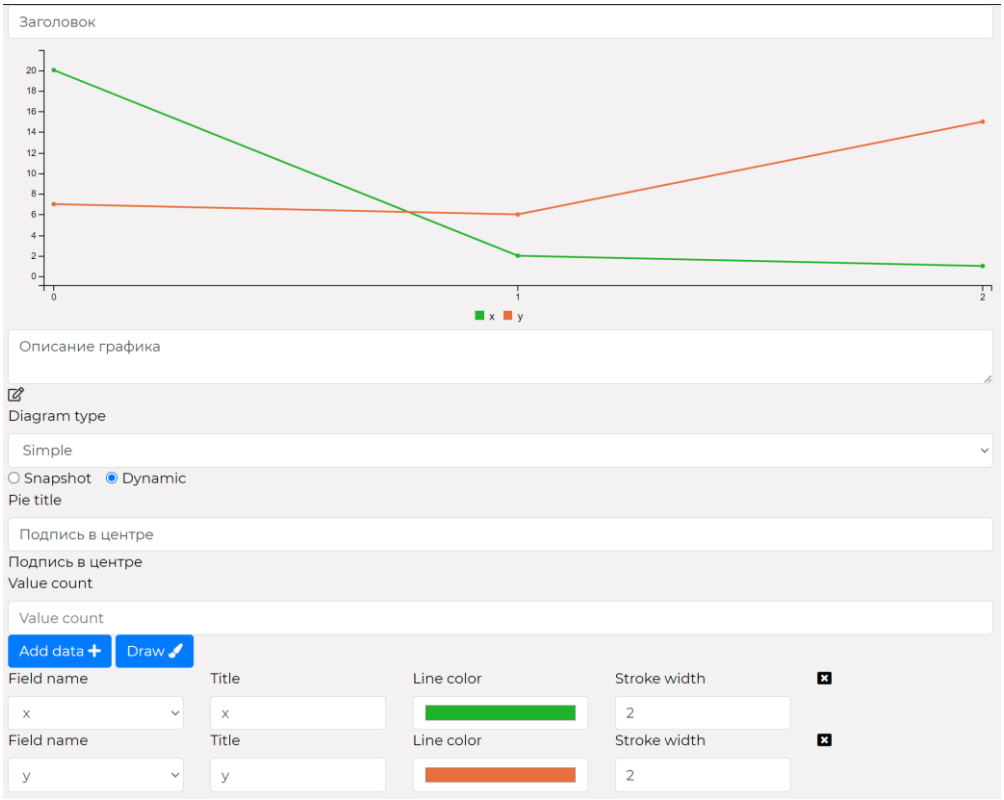


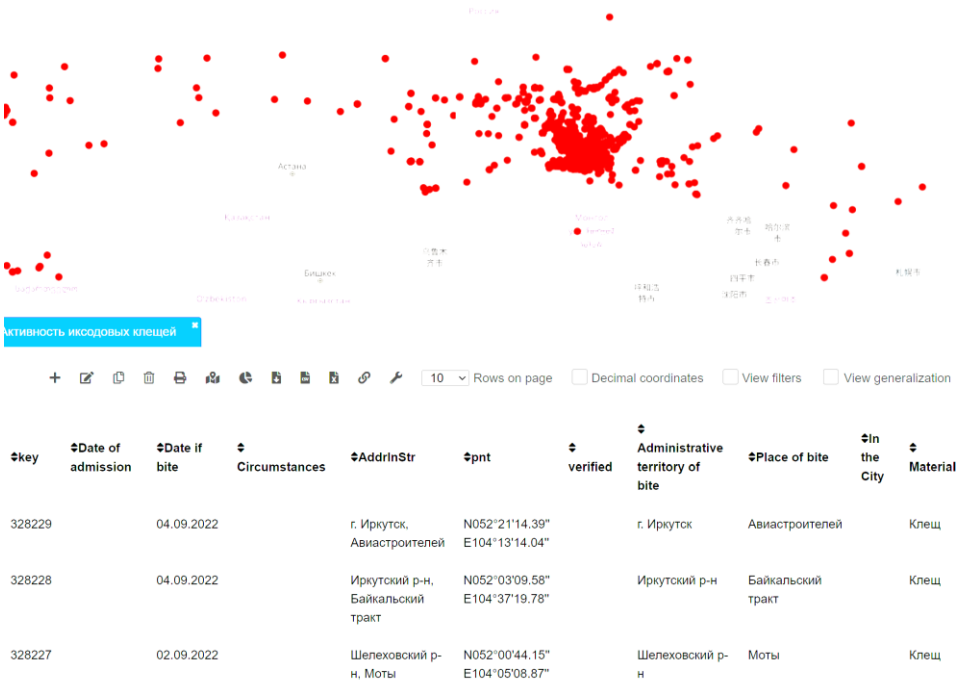
Рисунок 3.24 – Редактор модели графика

Поддерживается 5 видов диаграмм:

- линейный;
- столбчатый;
- временной ряд;
- круговой;
- XY зависимость.

В модели указывается, какие данные используются, вид графика, стили отображения, способ загрузки данных. Данные могут загружаться динамически в момент отрисовки графика, либо данные являются статичными и сохраняются в момент создания графика. Для динамически загружаемых данных в момент отображения пользователь должен иметь доступ к данным таблицы.

В качестве примера использования фабрики можно привести таблицу регистрации укусов иксодовых клещей (рисунок 3.25), которая ведется НЦ ПЗСРЧ.



key	Date of admission	Date if bite	Circumstances	AddrInStr	pnt	verified	Administrative territory of bite	Place of bite	In the City	Material
328229		04.09.2022		г. Иркутск, Авиастроителей	N052°21'14.39" E104°13'14.04"		г. Иркутск	Авиастроителей		Клещ
328228		04.09.2022		Иркутский р-н, Байкальский тракт	N052°03'09.58" E104°37'19.78"		Иркутский р-н	Байкальский тракт		Клещ
328227		02.09.2022		Шелеховский р-н, Моты	N052°00'44.15" E104°05'08.87"		Шелеховский р-н	Моты		Клещ

Рисунок 3.25 – Регистрация укусов иксодовыми клещами

Для представленных данных создана диаграмма (рисунок 3.26), отображающая распределение регистрации укусов клещами по годам. Первоначально данные в таблице сгруппированы и отфильтрованы, а затем переданы в редактор модели.

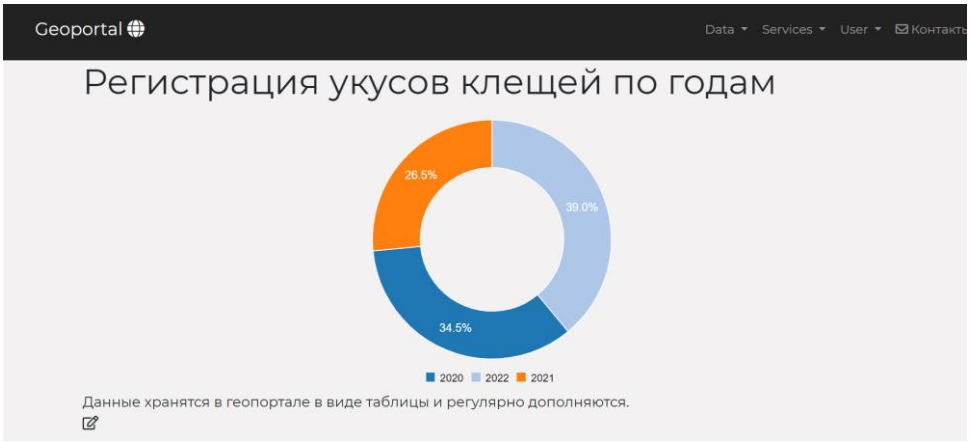


Рисунок 3.26 – Круговая диаграмма распределения регистрации укусов клещами по годам

3.6. Сервис конвертации реляционных данных

Сервис предназначен для загрузки реляционных данных из файлов различных ГИС форматов, поддерживаемых библиотекой GDAL/OGR. Кроме того, поддерживаются данные в формате CSV. Производится нормализация данных к существующим справочникам. Формат CSV поддерживается разными системами, в том числе Microsoft Excel. Модуль производит разбор текстового файла в формате CSV, извлекает названия атрибутов и данные. Можно указать кодировку импортируемых данных и используемый разделитель. Далее необходимо сопоставить атрибуты результирующей таблицы и импортируемой таблицы из текстового файла. В процессе загрузки можно обработать импортируемые значения. Пользователь должен выбрать функцию конвертации данных для каждого атрибута. На текущий момент реализованы следующие функции:

1) Copy – производит копирование без обработки значения из указанного атрибута импортируемой таблицы в атрибут результирующей таблицы. Если атрибут результирующей таблицы является ссылкой на таблицу справочник, то производится поиск соответствующего значения в таблице справочник и берется значение первичного ключа;

2) CopyDate – производит конвертацию даты из текстового формата в формат базы данных. Производится распознавание используемого формата даты импортируемого атрибута. Если формат не распознался, то пользователь может указать его вручную;

3) CopyValue – используется для того, чтобы при загрузке в указанном атрибуте всех импортируемых записей было присвоено значение – константа;

4) CopyPoint – используется для определения атрибута типа «Точка» (геокодирование). На входе метода указывается два атрибута, содержащие широту и долготу точки;

5) Geocoding – производит геокодирование адреса, т. е. определение координат точечного объекта по адресу на основе сервиса OSM.

Для всех значений производится удаление пробелов в начале и в конце строки. Числовые значения проверяются на разделитель целой и дробной части. При геокодировании и выполнении запросов к БД производится кэширование значений для ускорения работы модуля.

3.7. Каталог данных и структурных спецификаций

Каталог реализует следующие функции:

1) регистрация сервисов данных;
2) поиск сервисов данных;
3) редактирование ключевых слов.

Каталог создан с помощью Фабрики сервисов ввода и редактирования пространственных данных, соответственно, предоставляет программный и пользовательский интерфейс. Для каждого сервиса в каталоге хранятся метаданные в соответствии с ядром Dublin Core и структурные спецификации модели, необходимые для использования сервисов. Ключевые слова, которые входят в ядро метаданных, могут использоваться в качестве меток для сервисов. Кроме того, ключевые слова используются для поиска сервисов пользователем. Ключевые слова имеют отношения между собой. Одно из важных отношений, применяемое для поиска сервисов, является отношение «общее – частное». Чтобы использовать это отношение, ключевые слова хранятся в среде. Для хранения и редактирования ключевых слов создана таблица (рисунок 3.27), содержащая следующие атрибуты:

- keyword – ключевое слово на английском;
- ruskeyword – ключевое слово на русском;
- weight – вес, необходим для задания порядка в дереве;
- parent – родительское ключевое слово;
- domain – предметная область.

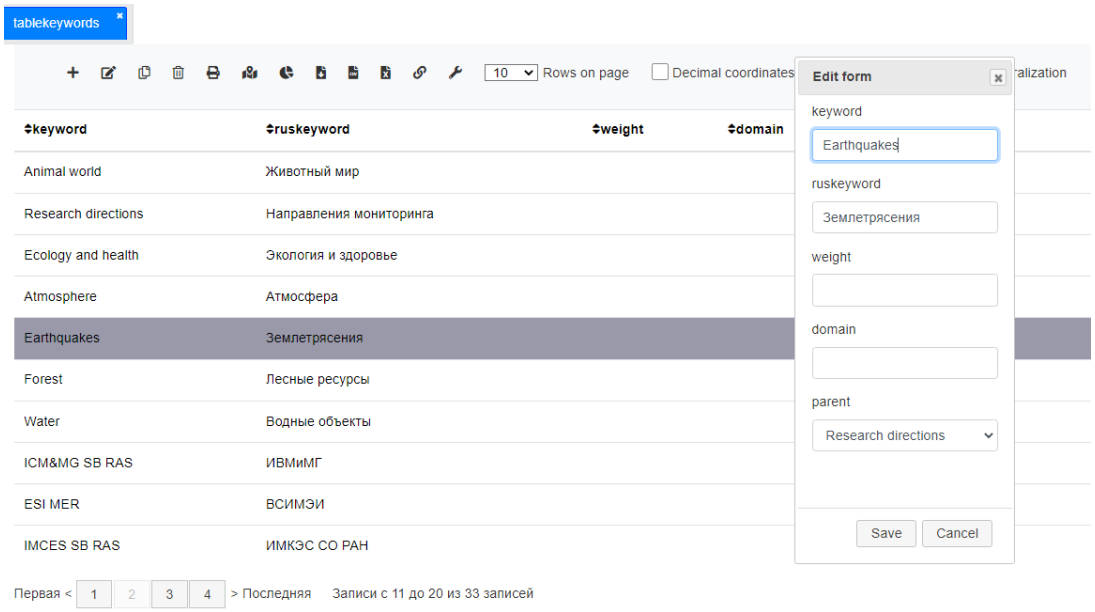


Рисунок 3.27 – Таблица ключевых слов

При регистрации сервиса данных пользователь указывает набор метаданных (рисунок 3.28).

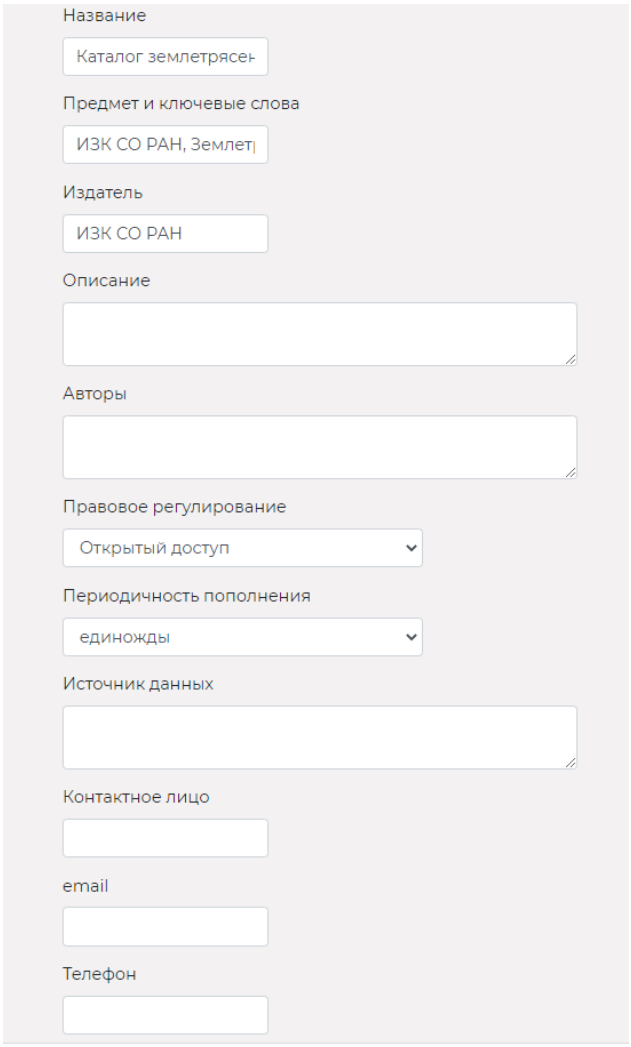


Рисунок 3.28 – Форма редактирования метаданных

Каталог сервисов данных и структурных спецификаций предоставляет функции поиска сервисов. Поиск производится по ключевым словам. Ключевые слова представлены в виде иерархий, которые упорядочивают их от общего к частному. Выбор ключевого слова в дереве приводит к поиску ресурсов и сервисов по оператору «OR» по выбранному ключевому слову и всем его дочерним ключевым словам. Это, например, позволит найти по ключевому слову «Организации» цифровой ресурс, в котором в метаданных указано только ключевое слово «ИДСТУ СО РАН», которое является дочерним по отношению к «Организации». Кроме того, каждое ключевое слово может быть задано на двух языках. Эти варианты тоже включаются с использованием «OR». Выбор пользователем нескольких ключевых слов приводит к поиску ресурсов, в котором ключевые слова объединяются оператором «AND», что приводит к результату, содержащему все ключевые слова.

Например, выбор ключевых слов «Лесные ресурсы» и «ИДСТУ СО РАН» приводит к поиску цифровых ресурсов, где имеются оба ключевых слова (рисунок 3.29).

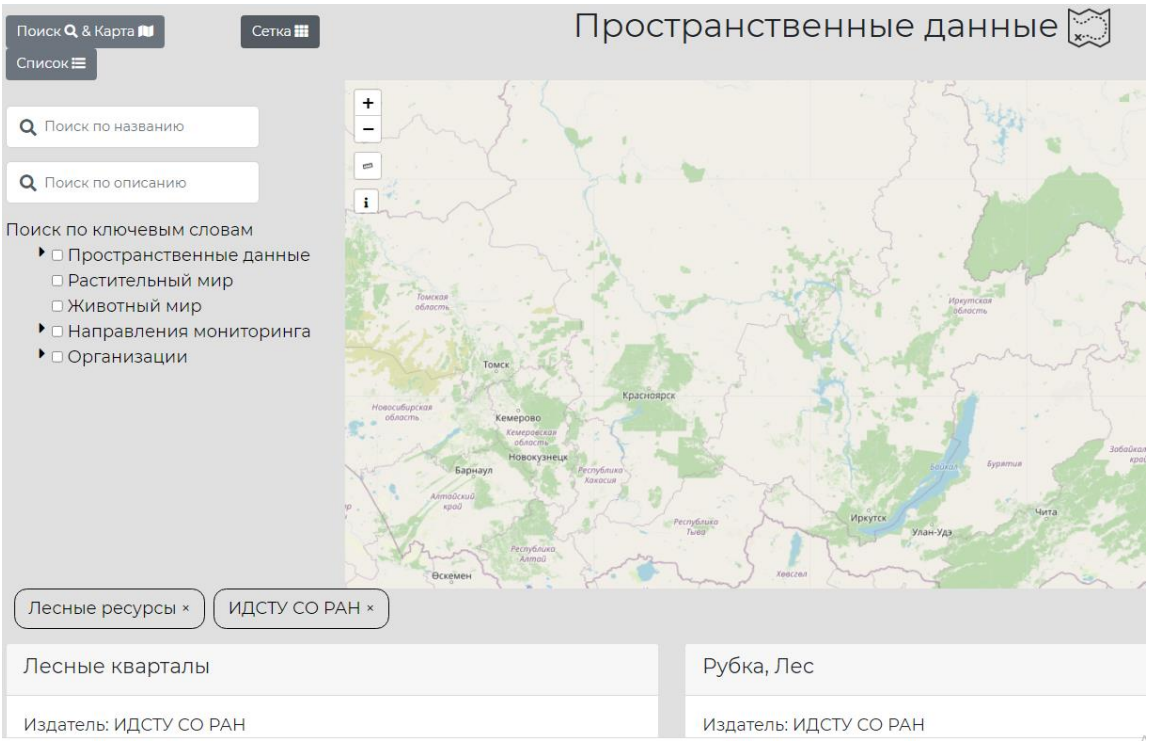


Рисунок 3.29 – Пример поиска сервисов данных по двум ключевым словам

Для реализации поиска цифровых ресурсов и выполнения сложных запросов (с использованием дерева различных операторов, в том числе «И», «ИЛИ» и т. д.) реализована поддержка спецификации GraphQL. Условие фильтрации записей таблицы формируется в виде JSON объекта, представленного ниже. Условие для атрибута формируется в виде объекта {Attribute : {Operator : Value}}. Для логических операторов «AND» и «OR» количество операндов составляет два или более, они представлены в виде массива. Реализованы операторы отрицания, строковые операторы «CONTAINS» проверки содержания строки и т. д. Пример GraphQL запроса:

Запрос на GraphQL на стороне сервера транслируется в предложение WHERE языка SQL. Спецификация GraphQL достаточно просто позволяет расширять набор операторов, полностью поддерживая предыдущие версии запросов.

Распределенные данные и метаданные

REST интерфейс сервисов ввода и редактирования реляционных данных реализует кроссдоменные запросы (CORS), что позволяет в браузере напрямую обращаться к интерфейсу сервисов другого геопортала. Реализованный элемент управления Classify, применяя данную возможность, может использовать данные таблиц, находящихся на других серверах. Наличие такого элемента управления позволяет создавать данные, связанные с данными на разных серверах. Чаще всего такой подход применяется при использовании справочников. Это позволяет использовать один справочник для разных таблиц, решая проблему совместного использования данных. Необходимо отметить, что набор данных справочников

может постоянно меняться. В этом случае значительно проще не отслеживать изменения, а использовать его удаленно.

При реализации каталога метаданных обычно преследуют следующие цели:

1) сбор большого объема информации о сервисах;
2) обеспечение корректности и информативности метаданных.

Пользователи в первую очередь регистрируют сервисы, в которых они заинтересованы, формируя наборы метаданных в рамках их предметной области. Соответственно, единый каталог должен вестись множеством специалистов из разных предметных областей, что усложняет обеспечение корректности метаданных и своевременное обновление. Универсальный каталог, содержащий метаданные для всех предметных областей, довольно неудобно использовать из-за большого количества сервисов. Кроме того, метаданные сервисов могут быть расширены дополнительной информацией, например, для упрощения их поиска пользователями. Поэтому для хранения метаданных (спецификаций) используется децентрализованная модель (рисунок 3.30). Реализованные в типовом геопортале каталог сервисов обработки данных и каталог данных предоставляют REST интерфейс, позволяющий программно запрашивать метаданные, в том числе получать изменения.

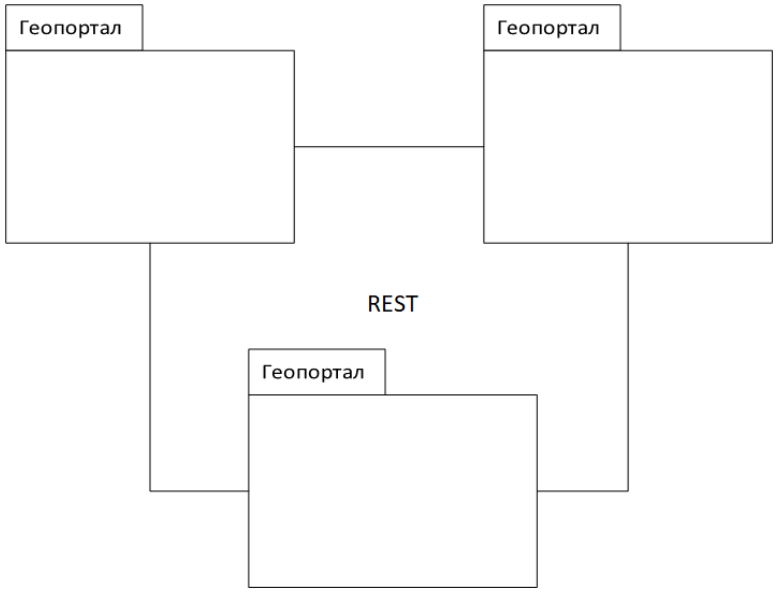


Рисунок 3.30 – Обмен метаданными

REST интерфейс позволяет искать метаданные по всем полям, например, по описанию, авторам, владельцам, ключевым словам и т. д. Используя дату последнего изменения, в каталогах метаданных можно получать все последние изменения метаданных.

3.8. Каталог сервисов обработки данных

Каталог сервисов предназначен для сбора метаданных сервисов, их хранения, поиска, выполнения, сбора статистики и композиции сервисов. Сбор статистики и методы композиции сервисов описаны в главе 2. Поиск нужных сервисов и их использование являются сложной задачей из-за их распределенности, большого количества, частого отсутствия метаданных и т. д. Поэтому каталог сервисов является ключевым элементом среды, позволяющим расширять набор методов обработки и анализа усилиями независимых разработчиков. Каталог создан с помощью Фабрики сервисов ввода и редактирования пространственных данных, соответственно, предоставляет программный и пользовательский интерфейс.

Поиск сервисов

Поиск сервисов (рисунок 3.31) производится по следующим атрибутам: названию, описанию, ключевым полям и типу. Реализация поиска описана в п. 2.9.

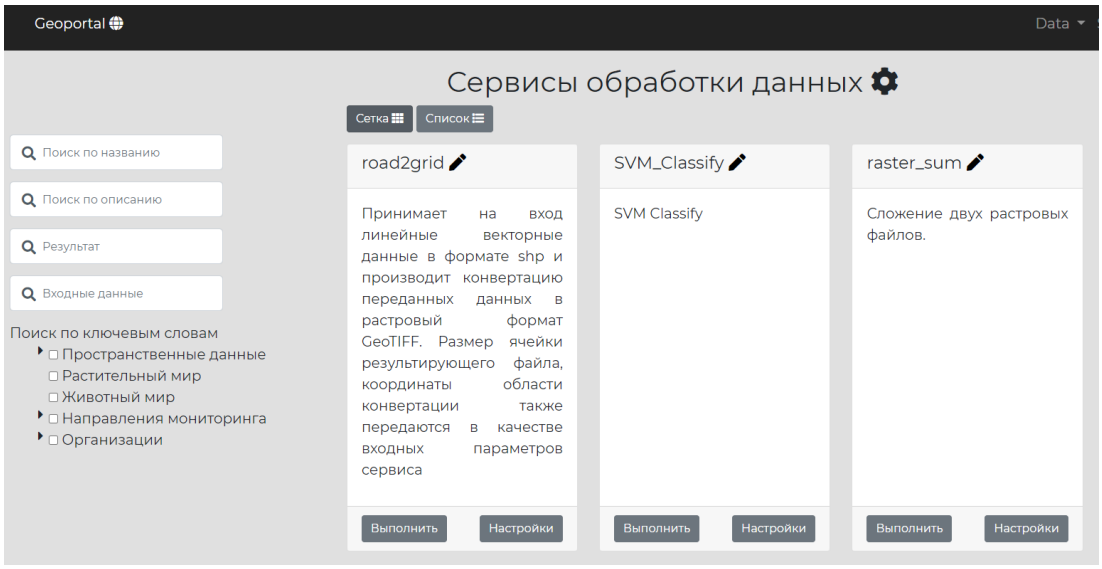


Рисунок 3.31 – Поиск сервисов

Регистрация сервисов

Разработана специальная форма ввода данных, представленная на рисунке 3.32, для регистрации сервисов и проверки корректности введенной информации.

Каждый сервис в системе задается такими параметрами, как имя, тип сервиса (WPS, REST, DAG, JavaScript сценарий), список вычислительных узлов, на которых данный сервис может выполняться.

В зависимости от типа сервиса форма регистрации сервиса требует разный набор атрибутов. При регистрации JavaScript сценария создается текстовое поле для ввода JavaScript кода. При регистрации *DAG* композиции создается текстовое поле для ввода *DAG* в виде JSON. Для REST сервисов вводится адрес.

Отличается регистрация WPS сервисов из-за поддержки в стандарте предоставления метаданных. На первом шаге регистрации WPS сервиса вводится адрес провайдера сервисов. Затем форма регистрации запрашивает по введенному адресу информацию о поддерживаемых сервисах с помощью метода GetCapabilities. Далее система загружает описание параметров выбранного пользователем сервиса и отображает как входные, так и выходные параметры. Для каждого входного параметра пользователь выбирает элемент управления (система элементов управления является частью геопортала), с помощью которого будут вводиться параметры для данного сервиса.

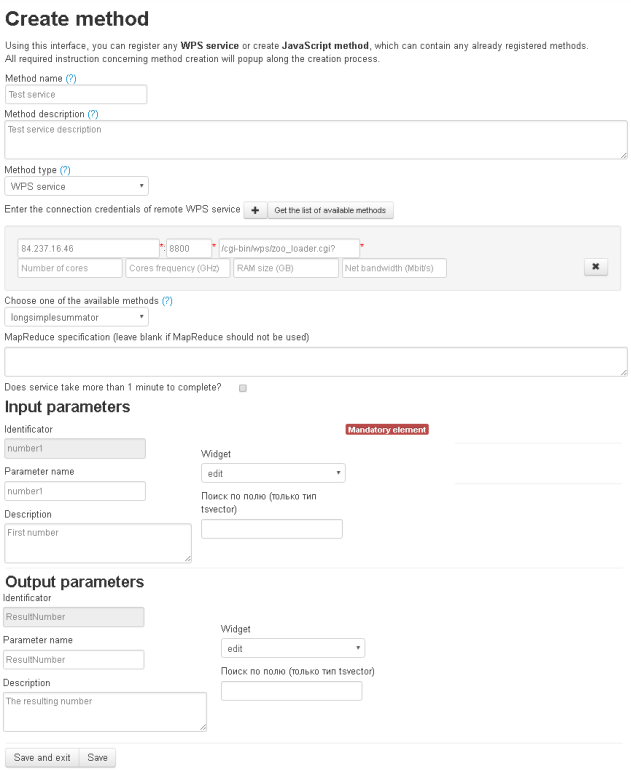


Рисунок 3.32 – Форма регистрации сервиса

Для типов REST, DAG, JavaScript сценарий, входные и выходные параметры указываются вручную. Пользователь создает параметр и указывает его имя и элемент управления. С каждым элементом управления связан тип, который присваивается параметру. В зависимости от элемента управления указывается дополнительная информация, которая поможет пользователю корректно ввести данные.

На форме пользователь определяет, работает ли сервис длительное время. Подсистема выполнения сервисов ожидает ответ сервиса в течение 60 секунд. Если ответ за это время не будет получен, то считается, что сервис не работает. Для сервиса, помеченного как длительный, при запросе на выполнение подсистема выполнения будет добавлять в запрос определенные параметры. При указании этих параметров сервис сразу возвращает ответ, содержащий ссылку на XML-файл, в котором по мере выполнения сервиса будет отображаться процент выполнения задания, а по завершению WPS-сервиса – результат работы.

Запуск сервисов

Для запуска сервисов пользователем разработана форма (рисунок 3.33), состоящая из четырех секций: описание сервиса, секция выполнения, консоль и история. Форма для всех типов сервисов является одинаковой. Описание сервиса содержит метаданные, введенные при регистрации сервиса. Секция выполнения предназначена для ввода пользователем входных данных сервиса и, если имеются результаты сервиса в виде файлов, то вводятся пути сохранения результатов. Поля ввода генерируются на основе описания сервиса. Для параметров, требующих файловые данные, автоматически создается элемент управления, который позволяет выбрать файл из системы хранения данных (СХД) и генерирует временную ссылку для скачивания файла сервисом. Для реляционных входных данных создается элемент управления, который позволяет пользователю выбрать таблицу, указать фильтры. Таблица будет сохранена в формате SHAPE и будут сгенерированы временные ссылки на все файлы этого формата. Для результатов сервиса в виде файлов после завершения работы производится загрузка данных и сохранение в СХД по пути, указанным пользователем. Реализация этих элементов управления и методов работы с таблицами и файлами интегрирует данные, находящиеся в СХД и

реляционных таблицах, с WPS сервисами. Консоль содержит информацию о ходе выполнения сервиса и возникающих ошибках.

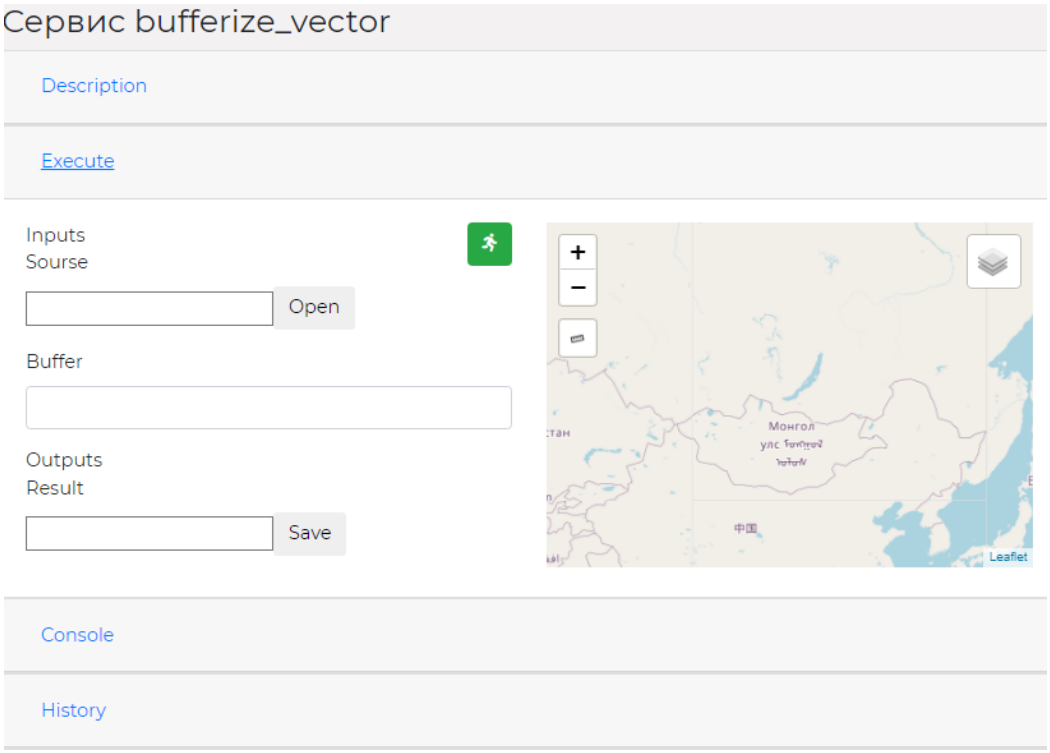


Рисунок 3.33 – Запуск сервисов

Раздел история (рисунок 3.34) отображает все задания сервиса в виде таблицы, с указанием значений входных и выходных параметров, успешности выполнения, времени работы, вывода консоли.

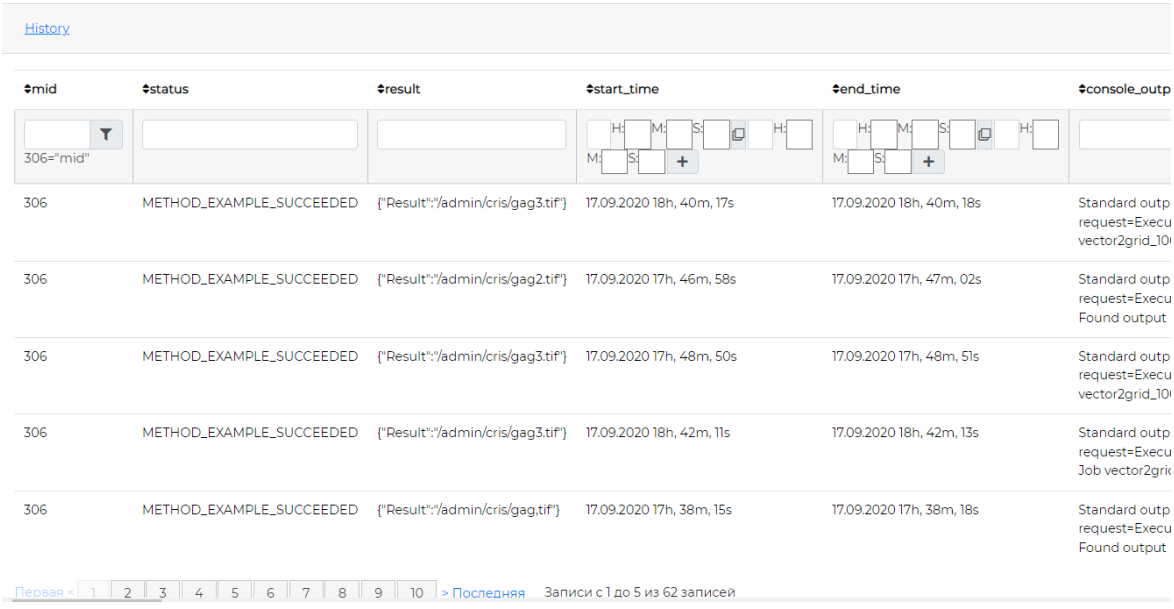


Рисунок 3.34 – История выполнения сервиса

При необходимости можно выбрать в таблице любое задание, указанные параметры будут введены в секции выполнения. Задание можно будет повторить или выполнить с модификацией параметров. При длительном выполнении сервиса форму можно закрыть, а результаты посмотреть потом в истории.

3.9. Базовые пространственные данные

В основе базовых пространственных данных (БПД) лежат готовые наборы цифровых данных. Основным назначением базовых пространственных данных является пространственная привязка различных тематических данных. Соответственно, нужно обеспечить их использование для создания новых данных, связывания с существующими данными и проведения анализа. Хранение, предоставление и использование базовых пространственных данных осуществляется с помощью сервисов ввода и редактирования. В рамках среды созданы слои БПД, соответствующих уровням управления территориями в Российской Федерации: федеральные округа, регионы, районы.

Созданные таблицы, содержащие базовые пространственные данные, могут использоваться для создания новых слоев. Для этого применяется элемент управления w_classify, который взаимодействует с соответствующей таблицей как с таблицей-справочником и позволяет использовать различные его атрибуты, в том числе пространственные. При использовании пространственных атрибутов автоматически добавляется слой на карту, для которого можно применить различные стили с учетом значений атрибутов.

Базовые пространственные данные могут использоваться для анализа данных. В рамках сервисов реализована фильтрация и группировка данных, используя административное деление. Чтобы реализовать фильтрацию и группировку данных, выполнена проверка топологии базовых пространственных данных и исправление ошибок.

3.10. Сервис-ориентированная технология проведения научных экспериментов

В рамках СОА результаты деятельности множества научных коллективов представляются в виде сервисов, что позволяет упростить их использование и

распространить в научном сообществе. Решение крупномасштабных сложных задач требует создания композиций сервисов, объединяющих результаты множества научных коллективов. В рамках СОИАС разработана сервис-ориентированная технология проведения научных экспериментов, которая состоит из следующих этапов.

Этап 1. Построение модели предметной области, которая состоит из разработки вычислительных сервисов, сервисов данных и сервисов публикации данных, регистрации сторонних сервисов, описания онтологической модели сервисов, формирования экспертных знаний и сбора статистики выполнения сервисов.

Этап 2. Создание композиции сервисов, реализующей вычислительный алгоритм. В рамках СОИАС существует два способа создания композиций сервисов. Наиболее простой – это автоматическое создание композиций на основе статистических данных использования сервисов. Для создания композиции достаточно применить сервисы. Сформированная композиция может ускорить многократное повторение последовательности выполнения сервисов на разных данных. Для более сложных задач с заранее не известным количеством заданий с обработкой промежуточных данных применяется разработка композиций на языке JavaScript.

Этап 3. Проведение расчетов на вычислительных ресурсах среды. Ввод параметров и запуск композиций производится с помощью каталога сервисов. Планирование и выполнение композиций осуществляется с помощью подсистемы выполнения WPS сервисов.

Этап 4. Публикация результатов, анализ результатов расчетов. Применяются сервисы публикации данных, которые позволяют отобразить данные в виде таблиц, графиков и карт. Результаты расчетов могут быть загружены на локальный компьютер и проанализированы в десктопном ПО.

Разработанная технология базируется на вышеупомянутой модели, алгоритмах, методах и программных компонентах, а также ресурсах распределенной

вычислительной среды в рамках единой технологической цепочки решения научных и прикладных задач в области геоинформатики. Технология позволяет использовать множество сервисов, разработанных различными коллективами, для решения сложных задач.

Выводы

В главе представлено разработанное в соответствии с вычислительной моделью композиции сервисов CM алгоритмическое и программное обеспечение. Программное обеспечение состоит из:

- оригинального программного инструмента «Фабрика сервисов ввода и редактирования реляционных данных». Создаваемые с помощью инструмента сервисы предоставляют метаданные, пользовательский и программный интерфейс редактирования данных, поддержку передачи данных WPS сервисам. Создание сервисов данных впервые производится на основе иерархической модели данных с возможностью задания асинхронного вычисления значений атрибутов с помощью сервисов. Создаваемые сервисы можно сразу включать во множество композиций;

- компонента «Фабрика сервисов отображения пространственных данных», предназначенного для создания новых WMS сервисов, каждый из которых обладает программным интерфейсом и готов стать частью композиции сервисов. Созданные сервисы автоматически регистрируются в соответствующих каталогах;
- каталога данных и структурных спецификаций, реализующего регистрацию сервисов данных, поиск сервисов и редактирование ключевых слов;
- каталога сервисов обработки данных, предназначенного для сбора метаданных сервисов, их хранения, поиска, выполнения и интеграции сервисов;
- сервиса конвертации реляционных данных, который позволяет объединить в композицию сервисы с разными по структуре данными;
- базовых пространственных данных, которые могут использоваться для анализа данных и создания новых данных с помощью сервисов ввода и редактирования.

Созданные программные компоненты нацелены на создание композиций сервисов.

Результаты исследований, представленные в данной главе, опубликованы в [145 - 160]. Разработанные сервисы ввода и редактирования данных в отличие от существующих ориентированы на обработку вводимых данных различными WPS сервисами и позволяют:

- создавать реляционные таблицы с произвольным количеством атрибутов, в том числе впервые – в соответствии со спецификацией входных данных сервисов;
- использовать при вводе данных унифицированные классификаторы, в том числе базовые пространственные данные;
- применять созданные таблицы в качестве входных данных в WPS-сервисах без специальной публикации в соответствии со стандартами в Интернет;
- отображать данные на основе стандарта WMS;
- организовать совместную работу пользователей над одной таблицей.

ГЛАВА 4. ВЫПОЛНЕНИЕ КОМПОЗИЦИЙ СЕРВИСОВ

4.1. Общая постановка задачи выполнения композиции сервисов

При увеличении количества пользователей или сложности решения задач выполнение сервисов сталкивается с ограничениями на вычислительные ресурсы, которые могут быть гетерогенными, т. е. обладать разными характеристиками по мощности, памяти и т. д. Композиция сервисов часто представлена сложным графом заданий. Задания могут быть зависимы по данным друг от друга и выполняться различное время на разных вычислительных узлах. Последовательное выполнение заданий композиций сервисов может привести к неэффективному использованию вычислительных ресурсов. Поэтому СОАИС должна обеспечить эффективное распределение вычислительных ресурсов при выполнении сервисов.

В тексте диссертации ранее рассматривалось описание композиций в виде направленного ациклического графа (Directed Acyclic Graph). Для решения некоторых задач такой способ определения композиций не подходит, поскольку часто невозможно сформировать композицию сервисов в виде *DAG* из-за необходимости промежуточной обработки данных. Например, результатом работы сервиса является таблица, а на вход другого сервиса требуется передать отдельную запись таблицы. Для композиции двух сервисов требуется получить необходимую запись, преобразовать к нужному виду для второго сервиса. Создание отдельного сервиса для выполнения подобных действий приведет к лишним временным затратам. Такие сервисы будут использоваться только для этого сочетания сервисов из-за специфичности обработки. Кроме того, в DAG все задания (вызовы сервисов) заранее определены, но в некоторых задачах число заданий может зависеть от данных. Альтернативным способом определения композиций сервисов является использование процедурных языков программирования. Например, в Everest (Mathcloud) [73] для задания композиций можно использовать язык Python. Применение процедурного языка позволяет обработать данные и вызвать сервисы в процессе выполнения программы (далее сценария). Количество заданий может быть большим, например, десятки тысяч. Вручную создать DAG с таким количеством заданий достаточно сложно.

Поэтому предлагается применение процедурного языка программирования для задания композиций, что позволяет проводить обработку промежуточных данных с помощью средств языка и его библиотек, использовать промежуточные данные в управляющих конструкциях языка. Применение процедурного языка значительно упрощает создание новых композиций сервисов. Существующие программные решения, использующие языки программирования для формирования композиций сервисов, как правило, не производят автоматическое планирование вызовов сервисов в условиях распределенной гетерогенной среды; процесс выполнения таких композиций сервисов неустойчив к изменениям, происходящим в среде. Методы планирования применяются к композициям сервисов, заданных только с помощью DAG.

Предполагается, что вычислительный узел, на котором развернут сервис, должен обрабатывать только одно задание в одну единицу времени. На практике запуск одновременного выполнения двух ресурсоемких заданий может привести к значительному увеличению общего времени выполнения. Поэтому на множестве вычислительных узлов *CR* необходимо отслеживать, чтобы два задания одновременно не выполнялись. Для реализации этого условия необходимо централизованное управление выполнением, которое реализует постановку заданий на очередь и планирование вычислений. В среде централизованное управление осуществляет подсистема выполнения WPS сервисов.

При запуске сервиса (рисунок 4.1) в зависимости от его типа имеются следующие способы формирования заданий:

1) WPS сервис или REST: будет сформировано одно задание, которое каталогом сервисов будет отправлено подсистеме выполнения WPS сервисов;
2) DAG: будет сформировано множество заданий, которые также будут отправлены подсистеме выполнения;
3) JavaScript сценарий: каталог сервисов отправляет программу интерпретатору JavaScript сценариев, который формирует задания в процессе выполнения программы.

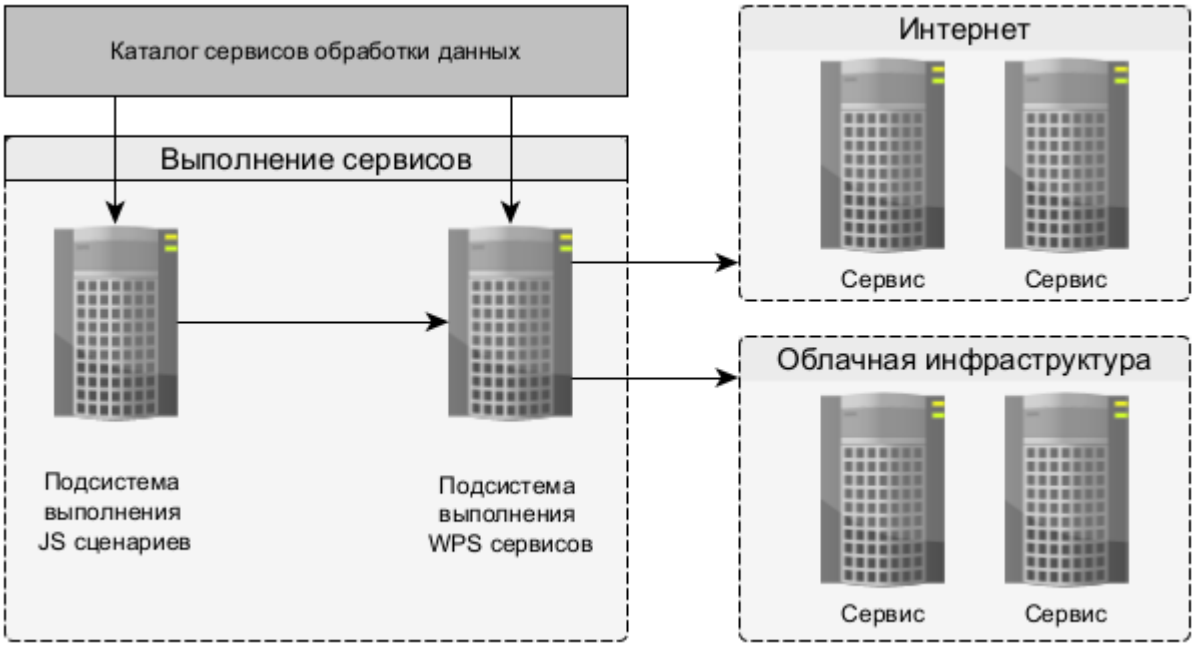


Рисунок 4.1 – Выполнение WPS сервисов

Подсистема выполнения WPS сервисов осуществляет их планирование и выполнение, обеспечивая эффективное по времени распределение заданий по вычислительным узлам. Подсистема предоставляет REST интерфейс для передачи заданий и получения результатов. Постановка задания на выполнение производится с помощью следующего POST метода:

http[s]://hostname:port/addtasks

Параметры метода:

- workflow – необязательный строковый параметр, содержащий идентификатор композиции сервисов;
- Tasks – JSON, содержащий список заданий и их зависимости (DAG);
- callback – адрес REST сервиса, которому нужно передать результаты выполнения композиции.

Метод возвращает статус успешности выполнения постановки в очередь.

Получение результата выполнения задания осуществляется следующим POST методом:

http[s]://hostname:port/taskinfo

Параметр метода task_id – идентификатор задания.

Метод возвращает статус выполнения сервиса. Если сервис закончил работу, то в ответ добавляются его результаты.

4.2. Применение процедурных языков программирования для задания композиций сервисов

Основная идея, заложенная в задание композиций сервисов с помощью процедурных языков программирования, заключается в том, что в процессе выполнения программы (сценария) на процедурном языке программирования формируется *DAG*, который одновременно выполняется с учетом зависимостей по данным и существующих вычислительных ресурсов. Планирование выполнения текущего состояния *DAG* осуществляется существующими оптимизирующими методами. План выполнения *DAG* пересчитывается при добавлении новых заданий, завершении выполняющихся заданий, изменении списка вычислительных ресурсов и т.д. В качестве языка программирования для задания композиций сервисов выбран JavaScript. Основными характеристиками, по которым выбран язык JavaScript, являются:

1) наличие реализаций интерпретатора языка с открытым исходным кодом, что позволяет организовать безопасное выполнение сценария на сервере;
2) наличие большого количества библиотек обработки данных.

Выполнение программы (интерпретация JavaScript) производится интерпретатором, созданным на основе V8 – движок JavaScript с открытым исходным кодом. Интерпретация сценария и выполнение DAG осуществляется параллельно. Планирование и выполнение DAG осуществляет подсистема выполнения WPS сервисов. Интерпретатор в процессе выполнения сценария передает подсистеме выполнения новые задания с зависимостями и обработанные в сценарии данные, в том числе промежуточные. Подсистема выполнения предоставляет интерпретатору промежуточные данные при необходимости их обработки или использования в управляющих конструкциях (рисунок 4.2).

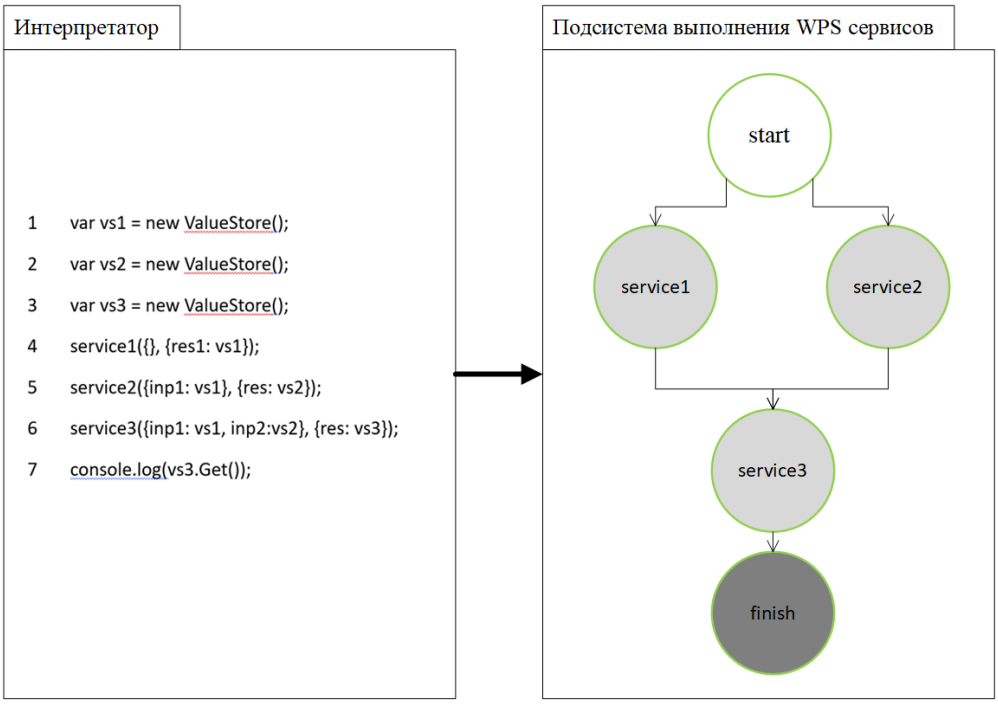


Рисунок 4.2 – Взаимодействие интерпретатора и модуля DAG

Формирование узлов графа DAG (заданий)

Передача заданий модулю DAG осуществляется с помощью вызова определенных функций, соответствующих зарегистрированным в каталоге сервисам (функции-обертки). При вызове функций-оберток происходит добавление задания в очередь с указанием сервиса, его характеристик, передаваемых и получаемых данных. Функция-обертка одного и того же сервиса может вызываться произвольное количество раз, в каждом случае это будет отдельное задание со своими зависимостями от других заданий.

Формирование ребер DAG, т. е. зависимостей между заданиями по данным, производится при добавлении узла DAG. Задание может использовать данные, полученные из следующих источников:

1) другие задания;
2) обработанные данные в коде сценария;
3) начальные данные, переданные пользователем.

Каждая функция-обертка имеет описание входных и выходных параметров, которое можно автоматически получить в соответствии со стандартом WPS из каталога сервисов. При вызове функции-обертки достаточно сформировать

зависимости входных параметров. Зависимости выходных параметров формируются автоматически при добавлении заданий.

Формирование зависимостей на основе анализа текста программы и выявления потоков данных является нетривиальной задачей, так как результаты выполнения сервисов могут помещаться в переменные, значения переменных затем могут переопределяться и т. д. В случае, когда источником данных является другое задание, сложность заключается в идентификации соответствия данных входных параметров регистрируемого задания и выходных параметров других заданий, и отсутствие их модификации в коде сценария.

Поэтому предлагается при вызове функции-обертки в качестве параметров передавать объекты специального класса ValueStore (рисунок 4.3), выступающие в роли контейнеров для данных и позволяющие однозначно идентифицировать передаваемые и получаемые данные. При генерации функций-оберток автоматически формируется интерфейс, требующий использования контейнеров для всех параметров. Контейнер данных может использоваться в качестве входных параметров для нескольких заданий. Он может использоваться только один раз в качестве выходного параметра функции-обертки, либо его значение задается в коде сценария, т. е. данные в него могут быть записаны единственный раз. Каждый контейнер имеет свой идентификатор, генерируемый автоматически. Запись данных осуществляется методом Set() в коде сценария или модулем DAG при завершении выполнения сервиса. Получение данных из контейнера производится с помощью метода Get(). Свойство isReady возвращает TRUE, если данные помещены в контейнер. Если контейнер получил или должен получить данные из другого задания DAG, то свойство source содержит ссылку на эту вершину.

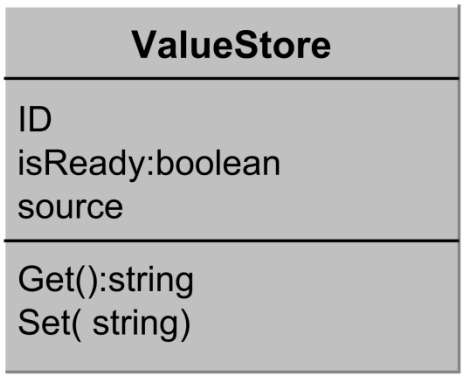


Рисунок 4.3 – Класс ValueStore

Рассмотрим определение зависимостей заданий на примере:

```
1  var vs1 = new ValueStore();
2  var vs2 = new ValueStore();
3  var vs3 = new ValueStore();
4  service1({}, {res1: vs1});
5  service2({inp1: vs1}, {res: vs2});
6  service3({inp1: vs1, inp2:vs2}, {res: vs3});
7  console.log(vs3.Get());
```

При вызове функции-обертки service1 в качестве выходного параметра указан объект vs1. Объект vs1 используется в качестве входного параметра при вызове функций-оберток service2 и service3. Объект vs2 используется в качестве входного параметра при вызове функции-обертки service3. В результате получается граф *DAG*, представленный на рисунке 4.4.

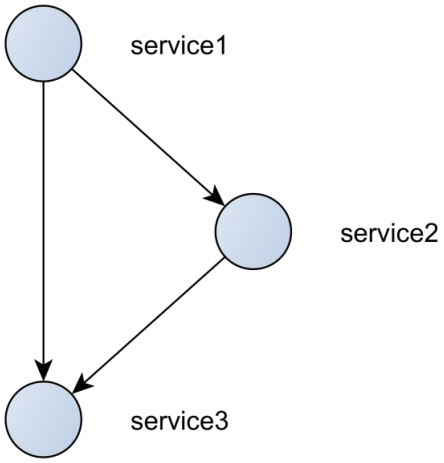


Рисунок 4.4 – Граф, созданный по сценарию

Определение зависимости между заданиями по данным происходит следующим образом: при выполнении функции-обертки сервиса происходит анализ входных параметров функции-обертки. Для каждого контейнера происходит запоминание его идентификатора и проверка, использует ли какое-либо задание данный контейнер в качестве выходного параметра. В случае если проверка находит такую вершину, происходит добавление ребра в граф, т. е. обнаружена зависимость между заданиями по данным.

Алгоритм формирования зависимостей задания

Paramlist – массив параметров добавляемого узла
numParameters – количество параметров добавляемого узла
ValueStorelist – массив ValueStore
numValueStore – количество элементов массива ValueStore
elist – массив зависимостей

```
1   elist = []
2   for I <- 1, numParameters do            //Цикл по входным параметрам узла
3           in_param <- paramlist[i]
4           if in_param.ValueStore.isReady  then //если значение параметра задано,
5                   continue                    //то переходим к следующему параметру
6           for j <- 1, numValueStore do        //Цикл по ValueStorelist
7                   cur_VS <- ValueStorelist[j]
8                   if in_param.ValueStore.ID = cur_VS.ID and cur_VS.source <> NULL then
9                           elist[elist.length+1] = cur_VS.source
```

Получение данных внутри сценария для последующей обработки средствами выбранного языка программирования производится с помощью метода Get класса ValueStore. Если на момент вызова метода Get данные еще не готовы, то метод блокирует выполнение сценария. Выполнение сценария композиции продолжится только после задания данных в контейнере, так как возможна ситуация, что в зависимости от результатов выполнения какого-либо сервиса будет происходить выбор определенной ветви сценария. Задание входных данных сервисов производится с помощью метода Set() (не производит блокирование).

Анализ формирования DAG в процессе выполнения сценария

Рассмотрим, каким образом управляющие конструкции языка программирования влияют на составление направленного ациклического графа зависимостей заданий по данным. Любой код сценария может быть представлен в структурном виде, т. е. процесс его выполнения определяется только при помощи трех структур управления: последовательность, ветвление и циклы [54]. Рассмотрим влияние этих управляющих структур на составление *DAG*.

1. Последовательные вызовы команд могут содержать вызовы функций-оберток, добавляющие в *DAG* новые задания соответствующих сервисов, как проиллюстрировано на рисунке 4.5. Зависимости между сервисами вычисляются в момент вызова функций-оберток. Новое задание может зависеть только от предыдущих. При последовательном вызове функций-оберток блокирование сценария не требуется.

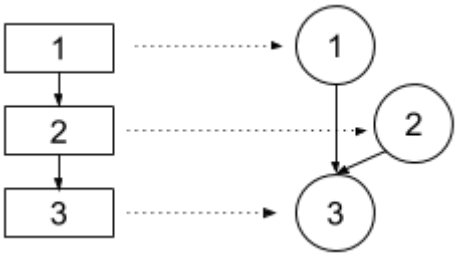


Рисунок 4.5 – Добавление задания в DAG в случае последовательного вызова сервисов

2. Ветвления представляют собой условные операторы, которые выполняют код в зависимости от определенных условий. Наполнение DAG будет зависеть от результата ветвления. Если ветвление происходит на основе результатов одного или нескольких сервисов, функции-обертки которых были вызваны ранее, то ветвление не произойдет до тех пор, пока необходимые для принятия решения результаты сервисов не будут получены, использование результатов работы сервисов в ветвлениях является блокирующим для выполнения композиции. На рисунке 4.6 проиллюстрирован процесс формирования DAG, где добавление заданий 3 или 4 происходит в зависимости условия, которое вычисляется на основе данных задания 2.

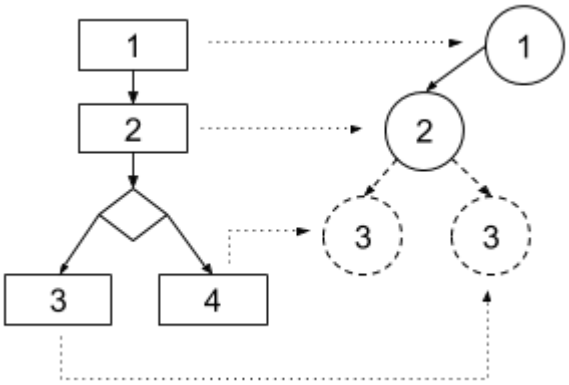


Рисунок 4.6 – Добавление задания в DAG в случае ветвления

3. Циклы представляют собой повторяющийся набор действий. Если в коде цикла присутствуют вызовы каких-либо сервисов, тогда каждая итерация цикла приводит к добавлению в *DAG* новых заданий (рисунок 4.7). Если в цикле не производится обработка результатов новых заданий, то, следовательно, отсутствуют блокирующие операции и цикл можно выполнить без остановок. Если же внутри повтора осуществляется работа с результатами выполнения заданий, то процесс

наполнения *DAG* будет разделен на периоды ожидания результатов выполнения сервисов с предыдущей операции.

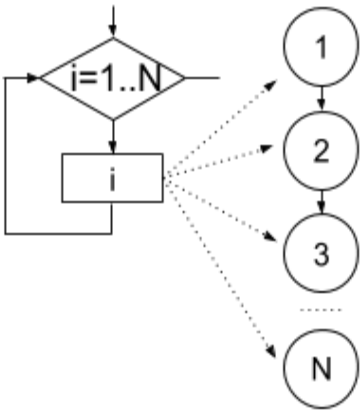


Рисунок 4.7 – Добавление задания в *DAG* в случае повтора

4.3. Параллельное выполнение сервисов обработки пространственных данных на основе подхода MapReduce

Обработка пространственных данных часто требует привлечения значительных вычислительных ресурсов. Существует широкий выбор программных систем, реализующих программную модель MapReduce, позволяющую параллельное выполнение обработки на множестве вычислительных узлов. Обработка пространственных данных органично и эффективно производится в рамках модели MapReduce. Однако методы, которые могут быть использованы в рамках этой модели, должны работать в рамках определенной программной инфраструктуры, которая не подходит для WPS сервисов.

В рамках сервис-ориентированной информационно-аналитической среды требуется реализация модели MapReduce для параллельной обработки пространственных данных с помощью WPS сервисов. Обработка данных в рамках модели MapReduce состоит из двух шагов: Map и Reduce. В шаге Map пространственные данные необходимо разделить по пространственному положению на N частей и произвести обработку каждой части с помощью WPS сервиса, затем на шаге Reduce собрать полученные данные. Требуется решить вопрос разделения и сборки пространственных данных и управления распределенным вычислением сервисов без программирования. Также стоит отметить, что в большинстве реализаций программной модели MapReduce необходим контроль над

вычислительными узлами, на которых будет развертываться система, в то время как при большом количестве сервисов, работающих на совершенно разных как аппаратных, так и программных платформах под управлением сторонних разработчиков, осуществлять прямое управление и настройку узлов практически невозможно. Для решения данной проблемы предлагается схема обработки растровых изображений в рамках модели распределенных вычислений MapReduce, которая позволяет использовать инструменты пространственной обработки в распределенной вычислительной среде без их модификации.

Рассмотрим особенности обработки растровых пространственных данных. Некоторым инструментам пространственного анализа для корректной работы достаточно обработать каждый пиксель входного растра независимо от других пикселей. В этом случае достаточно будет разделить входные данные на N равных частей и произвести обработку, затем собрать полученные данные. Другие инструменты работают с каждым пикселем входного растра и его окрестностью размера nxm, поэтому для корректной обработки входные данные придется разделить с некоторым перекрытием и определить, как поступать с повторяющимися результатами на шаге Reduce, т. е. усреднить результат, выбрать экстремальное значение, объединить, вычесть и т. д.

Операции над пространственными данными, которые используются в шагах Map и Reduce, повторяются для различных инструментов геообработки ввиду общности обрабатываемых данных. Предлагается метод, включающий в себя обработчики для операций Map и Reduce и спецификации, на основе которых будет происходить процесс распределения и сбора данных среди вычислительных узлов.

Программно обработчики шагов Map и Reduce представляют собой библиотеки, встраиваемые в подсистему выполнения WPS-сервисов. При выполнении сервиса подсистема читает спецификацию сервиса. Далее происходит анализ вычислительных узлов, поддерживающих выполнение данного сервиса. В зависимости от настроек распределения входных данных, определенных в спецификации, производится разделение входных данных с последующим формированием новых заданий обработки и их передача подсистеме выполнения

WPS сервисов для выполнения на вычислительных узлах. Reduce обработчик является также заданием, которое зависит от заданий обработки по данным. Соответственно, его выполнение происходит, как только последняя копия сервиса завершает свою работу, все результаты работы копий скачиваются, и происходит процесс сборки результата в соответствии с правилами, определёнными в спецификации. Файлы, получающиеся в результате процесса сборки, передаются дальнейшим заданиям композиции. На рисунке 4.8 изображена схема работы метода обработки пространственных данных.

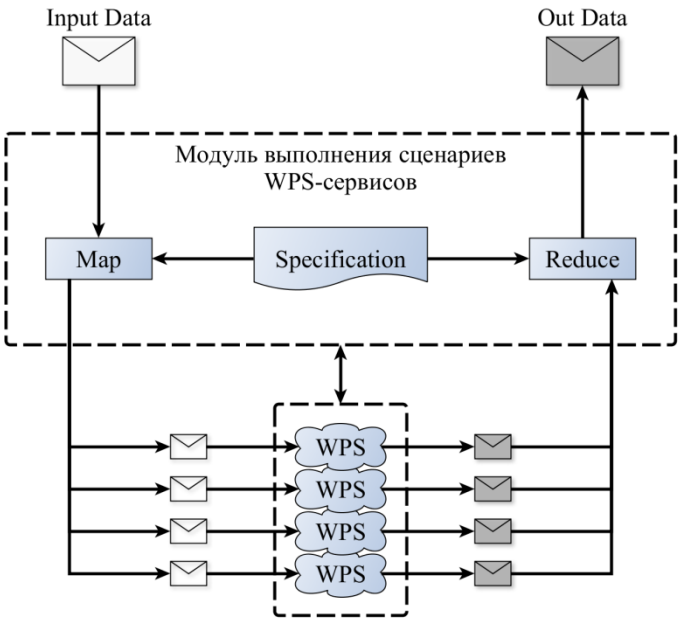


Рисунок 4.8 – Схема обработки растровых изображений в рамках модели MapReduce

Приведем описание некоторых элементов данного метода. Обработчик операции Map включает реализованные функции чтения спецификаций, на их основе формируются параметры для распределения растровых данных между вычислительными узлами. Для разделения данных формируются запросы/параметры запуска утилиты GDAL TRANSLATE, предназначенной для конвертации растров, с возможностью получения части растра. Обработчик операции Reduce включает реализованные функции чтения спецификаций, реализованные обработчики сбора данных. Обработчики данных реализуют стандартные функции обработки конфликтных ситуаций, возникающих в процессе сбора данных. Конфликтные ситуации возникают, например, при сборе частей

мозаики растра в одно целое. К таким ситуациям можно отнести поступление повторяющихся или неоднозначных данных. В этом случае обработчик применяет к ним операцию, указанную в спецификации. В текущей версии доступны следующие операции: max – установить максимальное значение из двух перекрывающихся пикселей, min – установить минимальное значение из двух перекрывающихся пикселей, avg – вычислить среднее значение из двух перекрывающихся пикселей.

Спецификации написаны в формате JavaScript Object Notation (JSON). Настройки спецификаций позволяют указывать минимальные и максимальные размеры ячейки для обработки, позволяя операции Map самостоятельно определять размер ячеек для оптимальной загрузки вычислительных узлов вызываемыми сервисами (в таком случае обработчик также сообщается число вычислительных узлов). В таком случае расчет ячейки производится на основе стратегии равномерной загрузки вычислительных узлов, то есть обработчик стремится занять как можно большее число узлов, при этом максимизируя размер ячейки и минимизируя число вызовов сервиса на каждом узле в целях минимизации расходов на соединение и передачу данных. Спецификации для операции Map содержат следующую информацию: ширина и высота ячейки данных, ширину полосы перекрывающихся пикселей для соседних ячеек. Спецификации для операции Reduce содержат название метода, применяемого на шаге сбора полученных результатов, для обработки перекрывающихся пикселей.

Отличительной чертой данного метода является возможность использования инструментов обработки пространственных данных в распределенной вычислительной среде без их модификации. Разработаны и реализованы обработчики для операций Map и Reduce, которые позволяют управлять процессом распределения и сбора обрабатываемых данных независимо от инструментов пространственной обработки. На основе спецификаций можно определить способ распределения и сбора обрабатываемых данных.

4.4. Реализация WPS-сервисов на вычислительном кластере «Академик В.М. Матросов»

С развитием параллельных вычислений и использования суперкомпьютеров появляется все больше программных систем, реализующих уникальные методы обработки и анализа данных, требующие высокой производительности. На сегодняшний день эти методы используются в основном только разработчиками. В связи с этим является актуальным предоставление доступа к таким программным системам в виде WPS сервисов для расширения круга пользователей. В рамках среды реализован доступ к кластерным программным системам в соответствии со стандартом WPS. Доступ реализован на основе службы Zoo Project, которая реализует основные функции стандарта WPS. В ИДСТУ СО РАН создан вычислительный кластер «Академик В.М. Матросов» ИСКЦ СО РАН. Вычислительный кластер предоставляет REST-интерфейс, позволяющий ставить в очередь, запускать и отслеживать задачи. Разработано специальное расширение для Zoo Project, которое на основе конфигурационных файлов выполняет запросы к REST интерфейсу вычислительного кластера. Запуск вычислительного процесса на кластере и получение результата его работы осуществляется в несколько этапов (рисунок 4.9):

1) пользователь вводит начальные параметры WPS-сервиса в каталоге сервисов и передает выполнение сервиса подсистемы выполнения WPS сервисов;
2) подсистема выполнения WPS сервисов осуществляет вызов WPS сервиса, реализованного на основе Zoo Project;
3) Zoo Project инициирует выполнение длительного процесса и выполняет запрос к REST-интерфейсу вычислительного кластера;
4) задача ставится в очередь на выполнение на кластере, Zoo Project периодически сверяется со статусом выполнения задачи и извещает об этом клиента (пользователя);
5) по завершению задачи Zoo Project возвращает результат в соответствии со стандартом WPS.

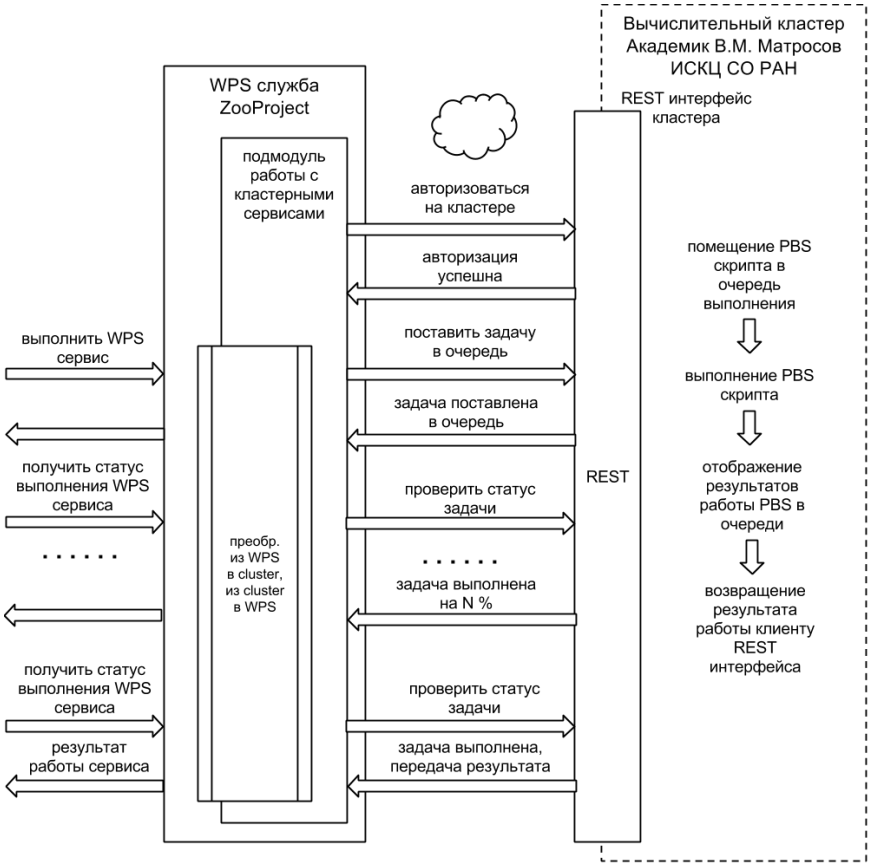


Рисунок 4.9 – Схема предоставления доступа к кластерным программным системам в виде WPS сервисов

Опишем более подробно основные функции программного REST интерфейса для Torque PBS. На текущий момент доступ к REST интерфейсу возможен только из локальной вычислительной сети ИНЦ СО РАН. Все запросы необходимо выполнять по протоколу HTTPS. Данные передаются в JSON-формате. Ответ сервер возвращает также в JSON-формате. Для авторизации пользователя необходимо отправить POST-запрос на адрес https://<address>/v1/auth.

Для добавления задачи в очередь (запуск PBS-скрипта) необходимо выполнить POST-запрос на адрес https:// <address>/v1/queue/task.

Тело POST-запроса:

{

“script_path”:”examples\/cpi_example_job.pbs”,

“arguments”: {

“param1”: ”value1”,

“param2”: ”value2”,

“param3”: “value3”

}

}

Где параметры: script_path — относительный путь до PBS-скрипта, param1, param2 и т. д. — это параметры, которые передаются запускаемой программной системе. В случае успешного добавления задачи REST интерфейс вернет следующий ответ:

{

“success”: “true”,

“message”: “14575.master”,

}

В случае если задача не была добавлена:

{

“success”:”false”,

“message”:”qsub: script file ‘examples\/cpi_example_job.pbs1’ cannot be loaded – No such file or directory.”

}

Для просмотра детальной информации о задаче необходимо выполнить GET-запрос на адрес https:// <address>/v1/queue/task/{id}, где параметр: {id} – числовой идентификатор задачи. Ответ в формате JSON приведен в приложении 3.

Таким образом, достигается главная цель – включение в вычислительную среду программных систем кластера «Академик В.М. Матросов» в виде стандартных WPS-сервисов.

4.5. Подсистема выполнения WPS сервисов

Подсистема выполнения WPS сервисов производит планирование, непосредственный вызов и получение результатов выполнения сервисов. Структура подсистемы представлена на рисунке 4.10. Взаимодействие с каталогами сервисов и JavaScript интерпретаторами производится с помощью модуля HTTP сервер, который реализует REST интерфейс. Добавление задания через REST приводит к постановке задания в очередь DAG, содержащего задания разных пользователей. При постановке производится запрос к базе данных для получения характеристик вычислительных узлов, на которых данный сервис может выполняться, и времени работы задания.

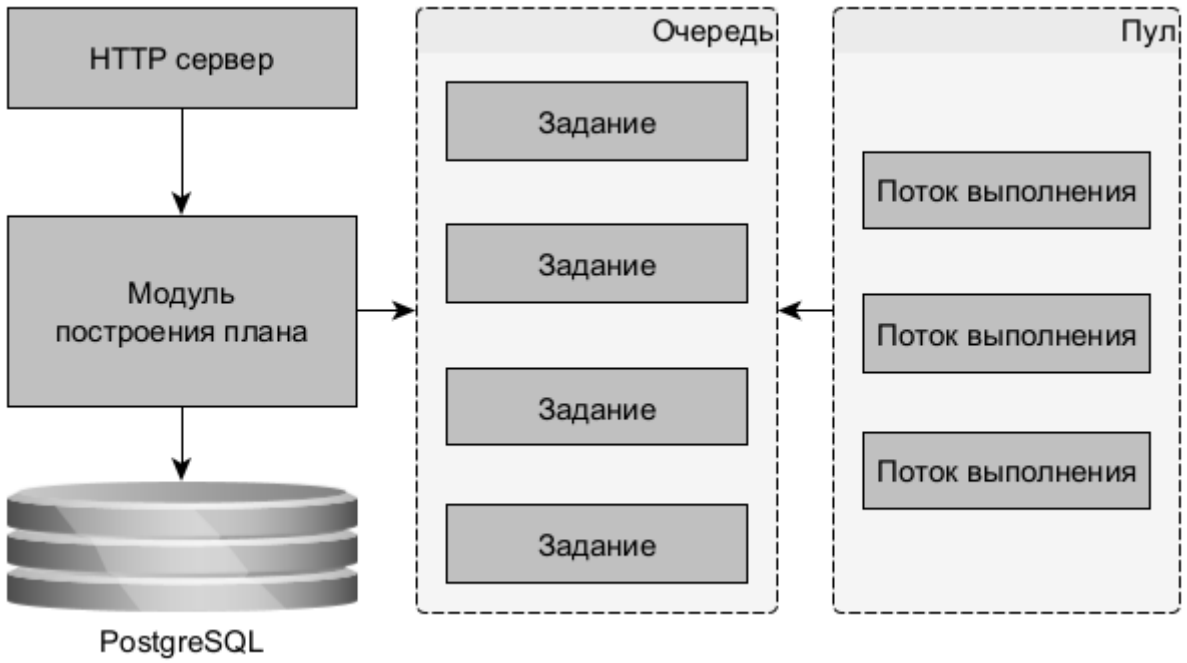


Рисунок 4.10 – Структура подсистемы выполнения WPS сервисов

Для нахождения плана, приближенного к оптимальному по времени выполнения, используется модификация спискового эвристического алгоритма составления расписания HEFT (Heterogeneous Earliest Finish Time), реализующего

метод поиска в глубину. Особенностями алгоритма, служащими для ускорения нахождения приемлемого плана, являются сортировка вершин DAG, вычислительных узлов и отсечение неперспективных ветвей графа поиска решения, использование информации о текущем состоянии вычислительной среды для более быстрого и точного нахождения плана. Поток выполнения запускает задания, проверяет их состояние, отправляя запрос на получение файла статуса выполнения. Потоков выполнения может быть несколько для того, чтобы параллельно запускать и отслеживать несколько сервисов.

При выполнении DAG необходимо учитывать изменение состояния вычислительной среды и добавление новых вершин в DAG. Все эти изменения требуют перестроения плана. Целесообразно проводить перестроение только при определенных событиях:

- включение или отключение вычислительного узла, поддерживающего выполнение хотя бы одного сервиса из DAG;
- добавление нового задания в DAG;
- отличие фактического времени выполнения одного из заданий DAG от ожидаемого;
- завершение выполнения задания с ошибкой.

При наступлении события необходимо сформировать новый план, минимизирующий время выполнения текущего состояния DAG. Разработанный в данных целях алгоритм можно представить в виде блок-схемы на рисунке 4.11.

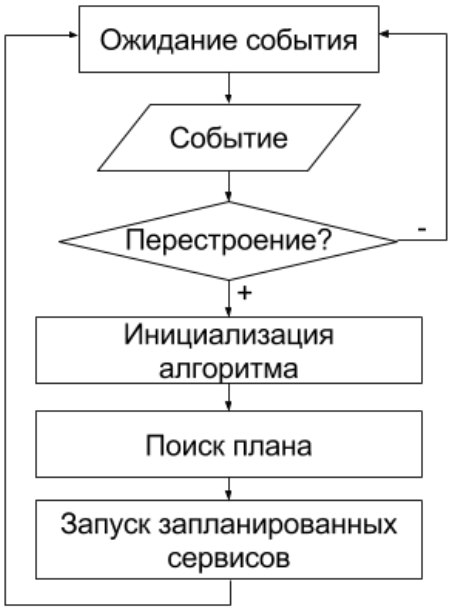


Рисунок 4.11 – Алгоритм работы модуля *DAG*

При регистрации события происходит определение необходимости перестроения плана. Например, в случае добавления задания в *DAG* и назначения его на менее загруженный вычислительный узел общее время плана не меняется, перестроение не требуется.

Выводы

В рамках среды разработан оригинальный программный компонент выполнения сервисов обработки пространственных данных, который позволяет распределять задания по вычислительным ресурсам, повышает надежность выполнения сервисов. Распределение вычислительных ресурсов основывается на модификации спискового эвристического алгоритма составления расписания HEFT (Heterogeneous Earliest Finish Time). Ставится цель минимизации времени выполнения. Программный компонент учитывает изменения состояния вычислительной среды, например, сбой работы узлов, изменение времени выполнения задания, добавление новых заданий и т. д.

С целью упрощения создания композиций сервисов предложено использование процедурного языка программирования для вызова сервисов и обработки промежуточных данных с помощью средств языка и его библиотек. Вызов сервисов производится через подсистему выполнения WPS сервисами, что позволяет впервые сочетать использование существующих методов планирования

выполнения композиций и параллельной обработки пространственных данных и гибкость создания на процедурном языке композиций сервисов. Результаты исследований, представленные в данной главе, опубликованы в [161-167].

ГЛАВА 5. СЕРВИС-ОРИЕНТИРОВАННАЯ ИНФОРМАЦИОННО-АНАЛИТИЧЕСКАЯ СРЕДА ОБРАБОТКИ ПРОСТРАНСТВЕННЫХ ДАННЫХ

СОИАС создана на основе сетевой и вычислительной инфраструктуры ИДСТУ СО РАН. В рамках среды создано 7 геопорталов, ориентированных на разные предметные области. СОИАС обеспечивает множество проектов, среди которых наиболее крупный «Фундаментальные основы, методы и технологии цифрового мониторинга и прогнозирования экологической обстановки Байкальской природной территории», работающий с 2020 года. Он объединяет 13 институтов Сибирского отделения РАН. В рамках проекта постоянно развивается геопортал «Цифровой мониторинг Байкальской природной территории» для проведения комплексного экологического мониторинга и прогнозирования на основе цифровых платформ, комплекса математических и информационных моделей, сервисов и методов машинного обучения, обеспечивающих сбор, хранение, обработку, анализ больших массивов разнородных пространственных данных. Геопортал находится по адресу https://geos.icc.ru/.

На основе геопортала производится сбор и создание сервисов по следующим блокам:

1. Формирование концептуальных основ инструментальной, инфраструктурной и прикладных цифровых платформ экологического мониторинга.

2. Формирование концептуальных основ мониторинга экстремальных природных явлений и антропогенных выбросов в атмосфере.

3. Формирование концептуальных основ мониторинга гидрологических режимов водоемов.

4. Формирование концептуальных основ оценки экологических рисков состояния растительного покрова.

5. Формирование концептуальных основ мониторинга экстремальных геологических и эколого-геохимических процессов.

6. Формирование концептуальных основ медико-экологического и эпидемиологического мониторинга.

На основе СОИАС разработаны и апробированы три методики метода создания композиций в разных предметных областях.

5.1. Методика создания композиций сервисов для ботанических исследований

Построение модели предметной области

Ботаническими исследованиями занимается достаточно много коллективов, находящихся в разных городах. Большая часть из них производит сбор данных (гербарных листов) о распространении различных видов. Данные гербарных листов определены стандартами TDWG Darwin Core и Access to Biological Collections Data (ABCD) и состоят из следующих полей: название вида, местонахождение, местообитание, даты сбора, определения, коллектор, автор определения, гербарная коллекция, координаты, число хромосом, полевой номер, примечания и т. д. Эти коллективы работают обособленно и редко взаимодействуют с друг другом. Для интеграции данных разных коллективов создана модель данных Core collection, с набором полей TDWG и ABCD. С помощью Фабрики сервисов ввода и редактирования реляционных данных на основе предложенной модели созданы сервисы данных, которые позволили коллективам организовать распределенный сбор данных. Для каждого сервиса данных определены пользователи и их права.

Большинство исследователей дополнительно к этому набору полей собирают специфичные данные, необходимые для проведения их исследований. При добавлении новых полей сформированы пользовательские дочерние модели (рисунок 5.1).

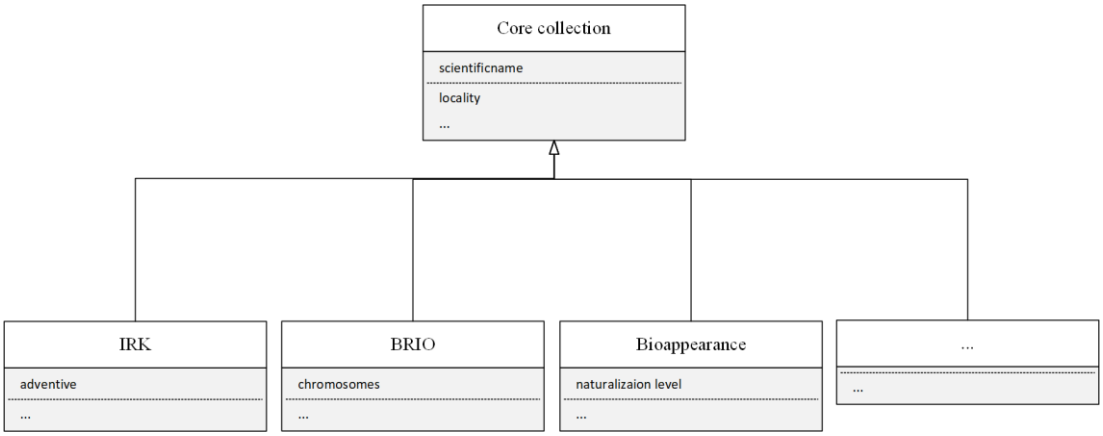


Рисунок 5.1 – Модели данных ботанических исследований

Редактор модели позволяет скопировать модель с существующего сервиса или унаследовать модель, а затем добавить специфичные поля. При копировании модели создается независимый сервис со своим набором прав доступа (обычно создается на отдельном геопортале). При наследовании создается сервис, собираемые данные которого автоматически добавляются в родительскую таблицу. Для реализации наследования сервисов в системе используется механизм наследования таблиц СУБД PostgreSQL. Применение механизма наследования таблиц и реализация наследования моделей таблиц позволяет учитывать специфику работы исследователей и в то же время формировать общий массив данных.

Исследователей достаточно сложно объединить на одном геопортале из-за различий в требованиях к пользовательскому интерфейсу и желания проводить сбор и хранение данных только в рамках своего коллектива. Поэтому для каждого коллектива исследователей развернут отдельный геопортал (рисунок 5.2) со своей иерархией таблиц.

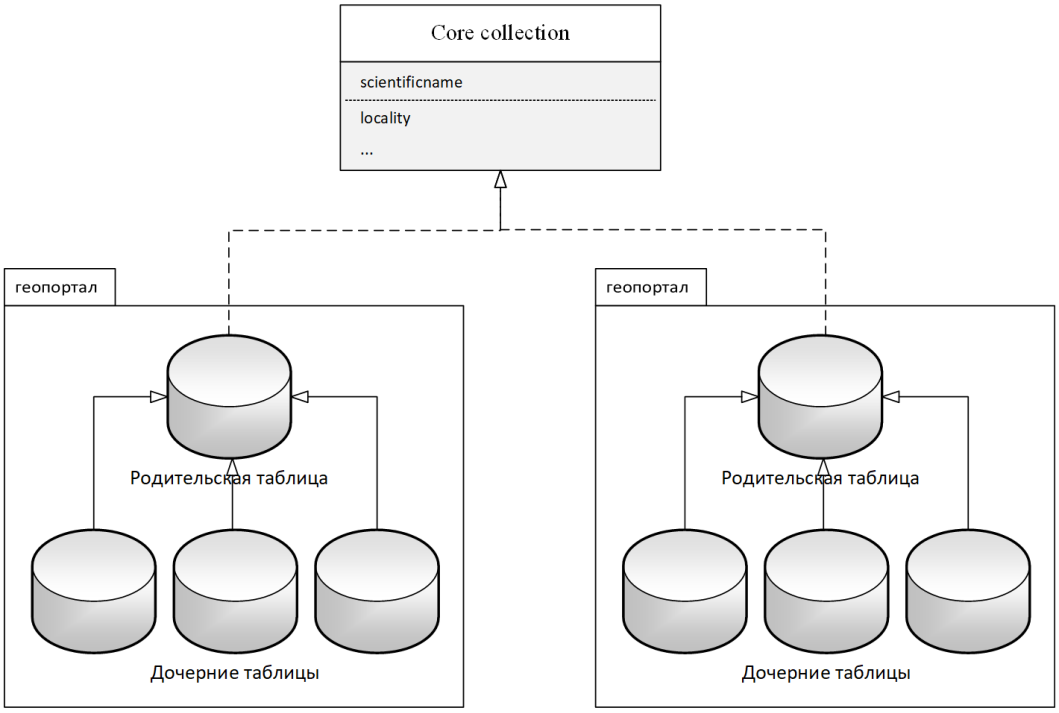


Рисунок 5.2 – Использование модели Core collection

Использование общей модели позволяет быстро создать новые таблицы, учитывающие пожелания исследователей, унифицировать структуру собираемых данных, а затем объединить данные и проводить совместный анализ. С помощью Фабрики сервисов ввода и редактирования реляционных данных созданы следующие сервисы:

1) БД «Гербарий сосудистых растений Сибирского института физиологии и биохимии растений СО РАН» (IRK);
2) БД «Гербарий мхов Сибирского института физиологии и биохимии растений СО РАН» (IRK);
3) БД «Адвентивные виды сосудистых растений Южной Сибири»;
4) БД «Первые находки чужеродных видов растений на территории Байкальской Сибири (Хронология, география)»;
5) БД «Флора сосудистых растений Заповедного Прибайкалья»;
6) БД «Рыбы»;
7) БД «Лишайники»;
8) БД «Печеночные»;
9) БД «Мохообразные»;

10) БД «Грибы» и т. д.

Разработка сервисов обработки данных

Сервисы данных, создаваемые с помощью Фабрики сервисов ввода и редактирования реляционных данных, представляют данные с помощью REST интерфейса. Большинство методов обработки используют входные данные в виде файлов в различных форматах. Реализованы следующие сервисы конвертации данных:

1) table2shp – сохраняет данные в формате SHAPE, который является наиболее распространенным для пространственных данных;
2) table2json – сохраняет данные в формате GeoJSON, который используется для композиций сервисов, заданных с помощью JavaScript;
3) table2csv – конвертирует данные в текстовый формат Comma-Separated Values (CSV);
4) table2xls – используется для экспорта данных в формат Excel с указанием необходимых типов данных.

Все сервисы сохраняют данные с учетом прав, заданных фильтров и возможности группировки данных.

Пространственный анализ распространения видов является наиболее часто используемым для проведения ботанических исследований. Для проведения пространственного анализа на основе библиотеки GDAL разработаны сервисы операций алгебры GRID raster_sum, raster_subtraction, raster_division и raster_multiplication, которые над файлами в формате GeoTIFF выполняют такие операции как: сложение, вычитание, умножение и деление растров. Все сервисы на выходе также получают растровый файл в формате GeoTIFF. Сервис newrastr создает новый растровый файл по заданным пользователем размерам и параметрам ячеек.

Для растеризации точечных векторных данных разработан сервис vector2grid, который принимает на вход данные в формате SHAPE и производит конвертацию данных в растровый формат GeoTIFF. Размер ячейки результирующего файла,

координаты области конвертации также передаются в качестве входных параметров сервиса. Сервис производит подсчет количества точек в ячейках, а если задано поле таблицы, то производится суммирование значений указанного поля по всем точкам ячейки. Данный сервис используется, например, для подсчета числа видов в ячейке, анализа конфигурации и плотности ареала видов.

Разработан сервис elevation_slope_aspect, который производит обработку данных SRTM (рельеф). Данные SRTM (Shuttle Radar Topographic Mission) доступны бесплатно. Исходные данные распространяются квадратами размером 1x1 градус. Название квадрата данных соответствует координатам его левого нижнего угла. Например: файл данных с именем N45E136 соответствует территории 45 гр.с.ш. 136 гр.в.д. Применяется SRTM во многих задачах, в частности для вычисления уклона (анг – slope) и экспозиции (анг – aspect). Уклон представляет скорость изменения высоты для каждой ячейки цифровой модели рельефа DEM (Digital Elevation Model). Экспозиция устанавливает направление максимального уклона в каждой ячейке. Использование данных SRTM обычно производится в следующей последовательности действий: определить названия требуемых файлов данных и скачать, подготовить данные к обработке (склеить, обрезать, представить в виде одного файла и т. д.), обработать данные инструментами геоинформационных систем (ГИС). При этом единицы измерения высот отличаются от единиц измерения системы координат. Инструменты построения карты уклонов рассчитаны на использование одних и тех же единиц измерения координат и высот. Поэтому их применение дает некорректный результат. Пользователю требуется произвести изменение системы координат. Процесс получения карты уклонов занимает значительное время. WPS-сервис автоматизирует обработку SRTM данных, позволяет сократить временные затраты, связанные с выполнением одних и тех же повторяющихся действий. Перед применением сервиса необходимо задать входные параметры: extent – размер входного растра, elevation – имя выходного растра, slope – имя выходного растра уклонов, aspect – имя выходного растра направлений уклонов, cell_size – размер ячейки (масштаб) результирующего растра. Размер входного растра определяется автоматически через взаимодействие с интерактивной картой, пользователь с помощью инструмента Rectangle (рамка)

выделяет интересующую область. Сервис находит соответствующие области файлы исходных данных. Исходные SRTM данные хранятся в системе хранения данных. Кроме того, пользователь указывает имена выходных файлов и размер ячейки. Далее пользователь запускает выполнение сервиса, в процессе которого производится склеивание и обрезание исходных файлов (формирование входного растра), перерасчет масштаба в соответствии с нужным пользователю, обработка входного растра инструментами уклон и экспозиция. Когда сервис завершит вычисления, пользователь получает три выходных файла в формате GeoTiff в системе координат WGS84: выходной растр, растр уклонов, растр направлений уклонов. Эти файлы сохраняются на сервер по пути, указанному пользователем. На рисунке 5.3 показано как выглядит сервис уклон/направление уклона и представлен результат работы сервиса.

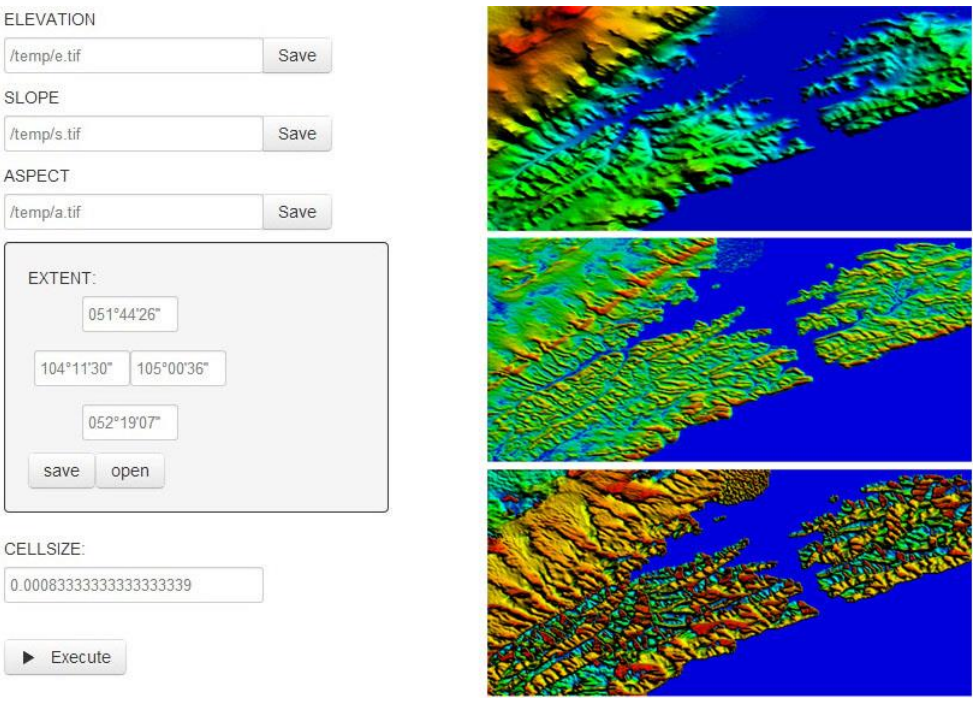


Рисунок 5.3 – Слева интерфейс сервиса, справа результат работы сверху вниз: исходные данные, уклон, экспозиция)

Анализ данных ботанических исследований часто проводится совместно с данными ДЗЗ. Разработан WPS-сервис NDVI, работа с которым происходит следующим образом. Пользователь указывает расположение файлов каналов

космоснимков и путь, по которому будет сохранен результирующий файл. Выходной файл содержит значения NDVI в диапазоне [-1,1], по желанию можно изменить диапазон выходных значений на диапазон [0,255], что удобно для представления в градациях серого или диапазон [0,200], что подходит для формирования карты цветов. На рисунке 5.4 показано, как выглядит интерфейс запуска WPS-сервиса NDVI.

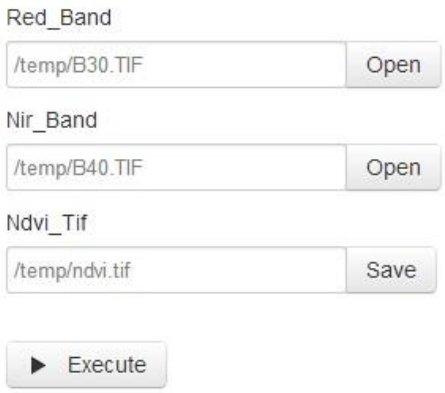


Рисунок 5.4 – Интерфейс запуска сервиса NDVI

Сервис интерполяции точечных данных на ячейки регулярной сетки методом естественных соседей. На входе сервиса слой точечных векторных объектов. На выходе – интерполируемые значения в ячейках регулярной сетки, в формате GeoTIFF. Пользователь может задать размер ячейки и область обработки. Сервис используется для интерполяции данных о распространении видов в регионе.

Описание онтологической модели и формирование экспертных знаний

Для всех сервисов данных, созданных на основе модели Core Collection, указана метка http://rs.tdwg.org/ontology/voc/Collection. Все метки базового сервиса автоматически указываются в метаданных дочерних сервисов. В качестве экспертных знаний для таблиц определены следующие метки:

1) geoportal:mark:relationaltable – данная метка присваивается всем сервисам, созданных с помощью Фабрики сервисов ввода и редактирования реляционных данных, а также входному параметру сервисов table2json, table2csv и table2xls. Применение этой метки позволяет предлагать пользователям воспользоваться одним из способов сохранения данных таблицы.

2) geoportal:mark:geometryrelationaltable – данная метка присваивается всем сервисам, созданным с помощью Фабрики сервисов ввода и редактирования реляционных данных и содержащим атрибут с типом GEOMETRY. Данная метка указана у входного параметра сервиса table2shp. Данные таблицы могут быть переданы сервису table2shp, который сохраняет пространственные данные в формате SHAPE.

Автоматическое создание композиций

При изучении того или иного вида растений является важным исследование ареала его распространения. Одним из способов анализа пространственного распределения объектов является построение карты плотности (тепловая карта). С помощью сервисов данных специалистами собирается информация о находках различных видов растений в реляционные таблицы. Для создания карты плотности распространения вида в рамках методики пользователь использовал следующие сервисы:

1) table – сервис данных, настроена фильтрация для получения данных одного вида и на определенную территорию;
2) table2shp – сохранены отфильтрованные данные в векторный формат SHAPE (пользователь указал сервис данных и файл результата);
3) vector2grid – выполнен подсчет количества растений в ячейках регулярной сетки, который сохранен в формате GeoTIFF (пользователь указал размер ячейки размером 20 х 30 км, файл в формате SHAPE, файл результата);
4) Map – создана карта ареала распространения вида (пользователь указать файл в формате GeoTIFF, название карты, файл стилей отображения данных на карте).

Информация об использовании сервисов сохранена в таблицу *Log*. Далее произведен анализ модели предметной области. На основе анализа автоматически создана композиция сервисов (рисунок 5.5).

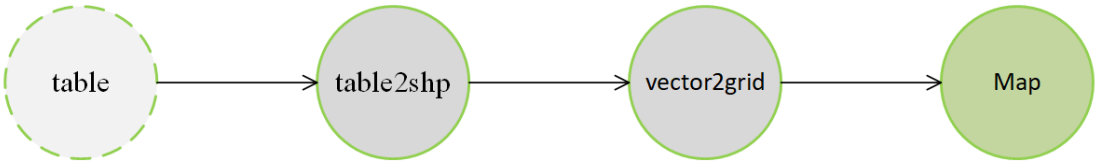


Рисунок 5.5 – Композиция сервисов для построения ареала распространения

Созданная композиция сервисов сохранена в каталог сервисов. Входными параметрами композиции сервисов являются:

1) сервис данных, при создании композиции все сервисы данных считаются параметрами и их можно поменять;
2) размер ячейки, по умолчанию предлагается размер 20 х 30 км;
3) название карты;
4) файл стилей отображения данных на карте, пользователь может указать собственный файл или использовать ранее заданный.

Значения следующих параметров указывать пользователю не требуется: имя выходного векторного файла SHAPE сервиса table2shp; имя входного векторного файла SHAPE сервиса vector2grid; имя выходного файла в формате GeoTIFF сервиса vector2grid; имя входного файла в формате GeoTIFF сервиса Map.

Для сравнения эффективности автоматизации выполнения сервисов произведем подсчет условных действий пользователя. Будем учитывать количество действий поиска сервисов и ввод параметров. Первый сервис требует три действия, второй и третий – четыре. Итого имеем 11 действий пользователя для создания карты ареала распространения вида.

При выполнении данной композиции пользователю необходимо выбрать композицию сервисов и указать таблицу, т. е. всего два действия пользователя. Конечно, любые свободные параметры сервисов могут быть изменены пользователем. В этом случае количество действий будет равно 5. Построение карты ареала распространения может выполняться для разных видов, по разным годам. Использование автоматически созданной композиции сервисов может значительно упростить работу пользователя и автоматизировать часто повторяющиеся его действия.

Композиция сервисов прогнозирования изменения флористического состава Байкальского региона

На основе методики создана композиция сервисов для прогнозирования изменения флористического состава Байкальского региона.

Изменение флористического состава территории возможно в результате двух процессов – исчезновения редких аборигенных видов и внедрения чужеродных адвентивных видов. Последний процесс носит название биологического загрязнения. Вопросы мониторинга, прогнозирования, предотвращения такого загрязнения являются одними из важнейших для сохранения биоразнообразия, так как наиболее агрессивные из чужеродных видов – инвазионные, способны вытеснять или препятствовать возобновлению видов природной, аборигенной флоры, а также наносить ущерб народному хозяйству и здоровью людей. Процесс внедрения чужеродных растений и исчезновения редких видов зависят от социально-экономического развития региона. Так, считается, что распространение чужеродных видов зависит от развитости транспортной сети.

Однако применение математического аппарата для мониторинга и тем более прогнозирования этих процессов с учетом различных факторов не развито. Так, отнесение чужеродных видов к инвазионным и классификация инвазионных видов по степени биологической опасности основана на экспертной глазомерной оценке и является описательной. Для примера к первой категории отнесены виды-«трансформеры», которые активно внедряются в естественные и полуестественные сообщества, изменяют облик экосистем, нарушают сукцессионные связи, выступают в качестве эдификаторов и доминантов, образуя значительные по площади одновидовые заросли, вытесняют и (или) препятствуют возобновлению видов природной флоры. В разделении чужеродных видов по уровню биологической опасности крайне велика доля субъективизма.

Для оценки изменения флористического состава использовались сервисы данных, в которых на тот момент было 66 тыс. записей. Выбран ряд модельных видов для создания прототипа сервиса прогнозирования изменения флористического состава.

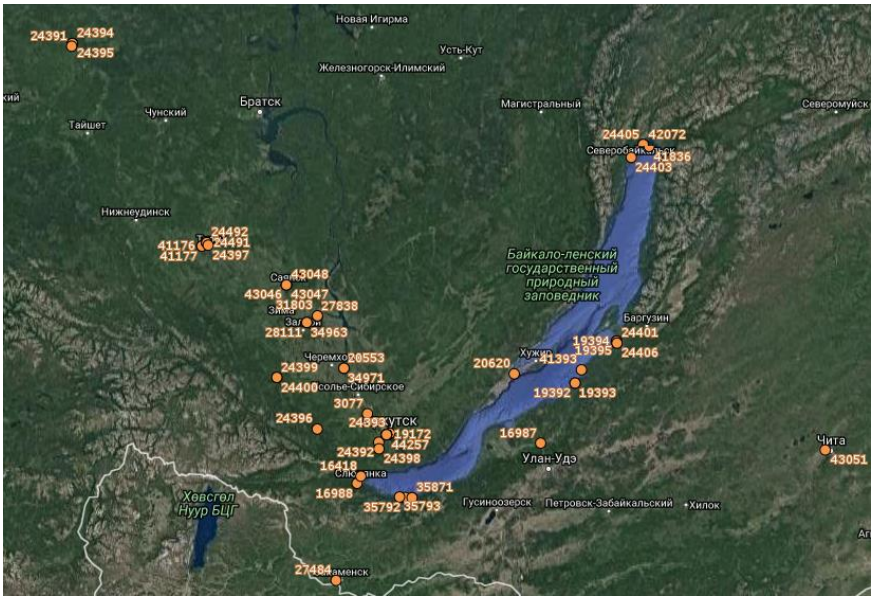


Рисунок 5.6 – Карта распространения вида «ячмень гривастый»

К таковым отнесены инвазионные виды наиболее высокого уровня агрессивности: наземный вид ячмень гривастый (Hordeum jubatum) (рисунок 5.6), водный вид элодея канадская (Elodea canadensis) и дичающий из культуры вид недотрога желеконосная (Impatiens glandulifera). Созданы карты распространения этих видов по БПТ. Также созданы карты по всем видам, признанным экспертами инвазионными – 17 видов, и потенциально инвазионными – 92 вида.

Важно своевременно выделять виды, которые представляют угрозу естественным экосистемам и/или народному хозяйству и здоровью людей. Одним из главных признаков, по которым чужеродный вид может быть отнесен к биологическим загрязнителям, является его распространение в регионе. Также по распространению в регионе определяется и уровень биологической опасности.

В качестве прецедента принято распространение двух видов трансформеров (Hordeum jubatum и Elodea canadensis – 1 категория, наиболее активные биологические загрязнители. В БПТ к трансформерам отнесены только они). Предметными специалистами решено считать, что чужеродный вид, достигающий такого же соотношения числа ячеек, в которых он встречается к числу, исследованных ячеек, что и у этих видов, можно рассматривать как трансформер. В идеале рассматривать соотношение числа ячеек, в которых встречается вид, к общему числу ячеек на территории. Однако в связи с тем, что территория БПТ в

ботаническом плане исследована достаточно неравномерно, первым этапом было выявление наиболее представительных с точки зрения флористической изученности участков. Выявление осуществляется в два этапа:

1) формируется регулярная сеть размером 20 х 30 км, в ячейках которой вычисляется количество видов;

2) выделяются ячейки с количеством видов больше заданного порога.

Для выявления порога, при котором ячейку можно считать репрезентативной, подсчитано ожидаемое число видов сосудистых растений на территории БПТ для площади 600 км^2. Оно высчитано по уравнению Аррениуса

$$y = a \times x^z, \qquad (1)$$

где y – кол-во видов на площади x, a, z – константы (z описывает пространственное разнообразие флоры, a – количество видов растений, свойственное единице площади, $z = 0.102$, $a = 309$ значения выбраны для таежной зоны Сибири (Малышев, 1975 г.)). На площадь 600 км^2 с использованием уравнения Арениуса получаем ожидаемое число видов для ячейки, равное 593. К числу исследованных, репрезентативных ячеек отнесены ячейки, в которых выявлено не менее 50 % видов от ожидаемого числа, соответственно, порог будет равняться 296.

На основе разработанного сервиса создана карта (Рисунок 5.7 –)

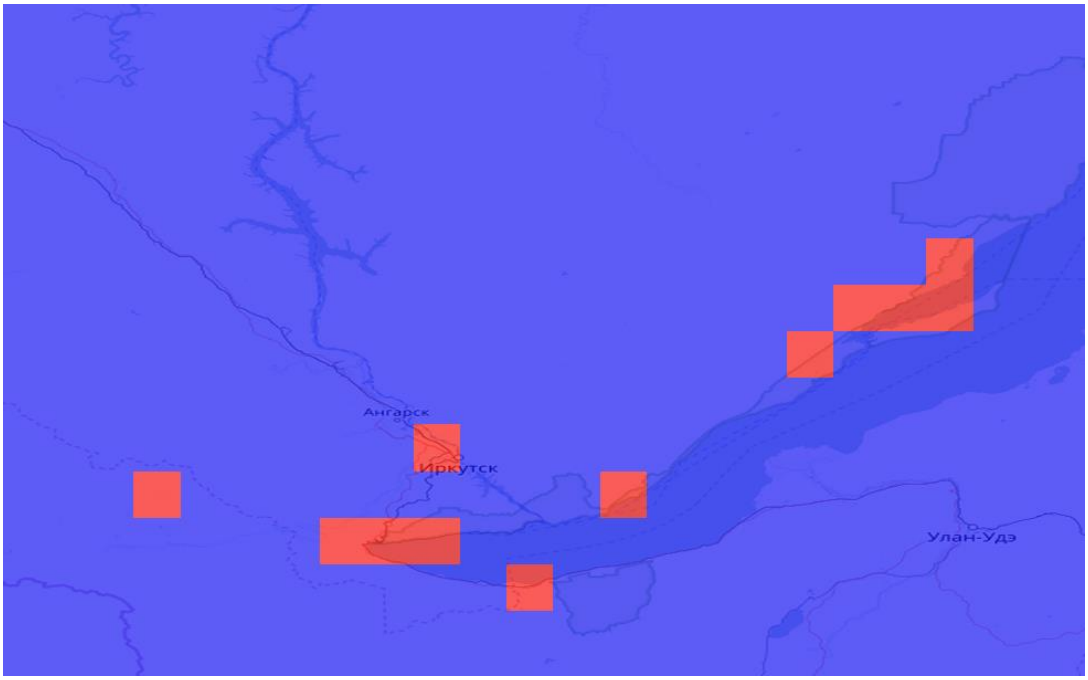


Рисунок 5.7 – Карта флористической изученности

Решено считать, что чужеродный вид, достигающий такого соотношения числа ячеек, в которых он встречается, к числу флористически изученных ячеек, как у Hordeum jubatum для наземных или Elodea canadensis для водных видов можно рассматривать как вид – трансформер. Такой вид нуждается в сборе дополнительных данных и мониторинге. Также требуют сбора дополнительных данных и мониторинга виды, распространение которых в репрезентативных ячейках имеют близкую плотность к модельным видам.

Для выполнения сравнительного анализа распространения вида применены сервисы, разбитые на следующие этапы:

1) формирование регулярной сетки, где в каждой ячейке ставится флаг наличия исследуемого вида (Рисунок 5.8 –);

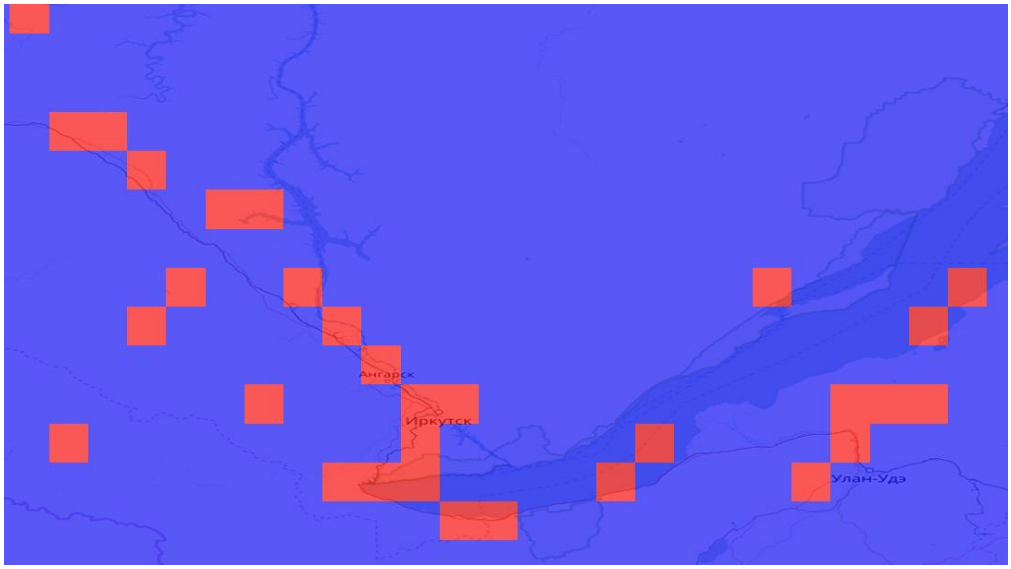


Рисунок 5.8 – Распространение вида Hordeum jubatum

2) для каждой ячейки регулярной сетки производится операция логического «и» с соответствующей ячейкой карты флористической изученности. В результате получаем карту с ячейками, где в ячейки установлен флаг при наличии вида и флористической изученности (рисунок 5.9);

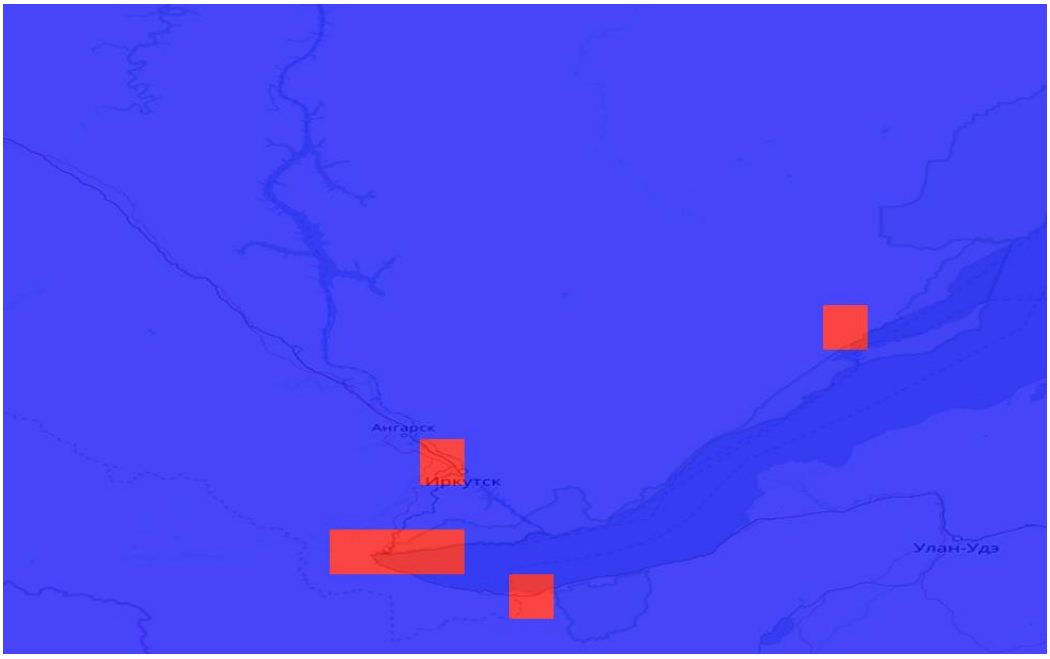


Рисунок 5.9 – Наличие вида Hordeum jubatum в флористически изученных ячейках

3) производится подсчет ячеек с установленным флагом и вычисление отношения с количеством ячеек, флористически изученных (рисунок 5.10).

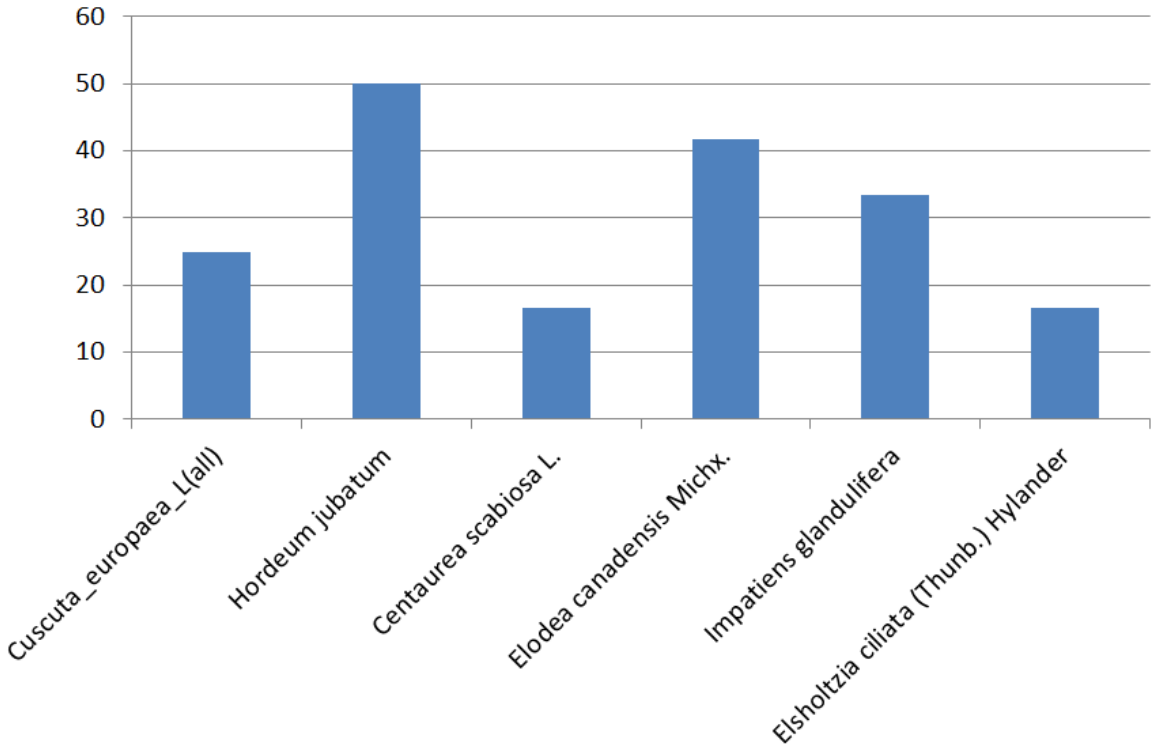


Рисунок 5.10 – Сравнение видов по распределению

В результате последовательных вызовов сервисов сформирована статистика. На основе анализа статистики автоматически создана композиция сервисов, представленная на рисунке 5.11.

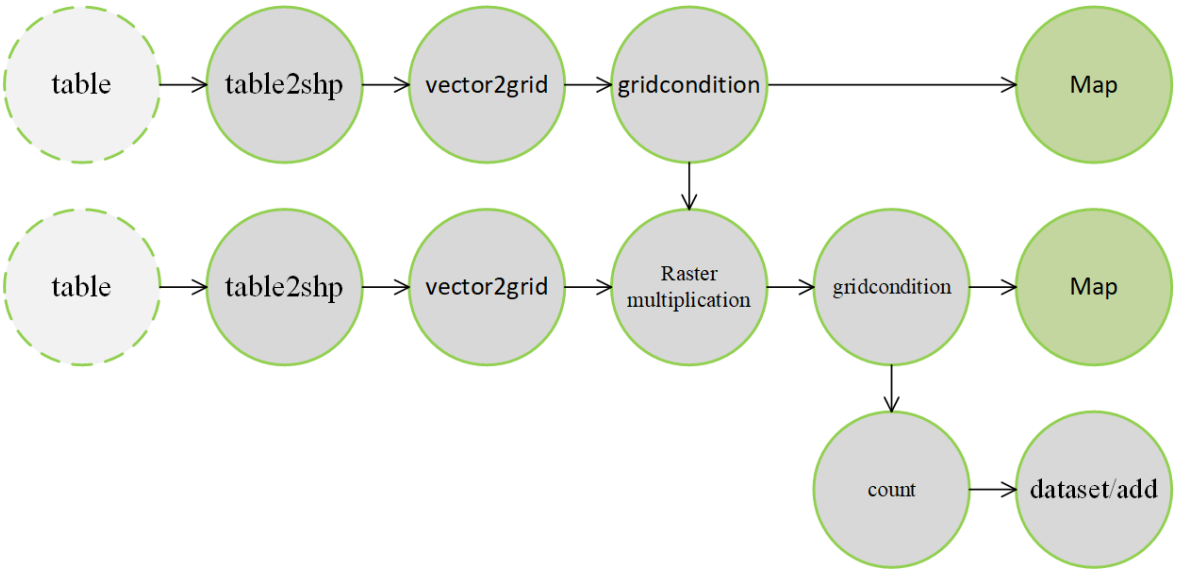


Рисунок 5.11 – Композиция сервисов анализа распространения вида

Распространение композиций сервисов

Сохранение созданной композиции в каталоге сервисов позволяет любому пользователю повторить последовательность сервисов с теми же параметрами на других данных. Указанные при построении модели метки сервисам данных позволяют рекомендовать пользователям сервисы. Применение композиции сервисов для сервиса данных с меткой http://rs.tdwg.org/ontology/voc/Collection автоматически присваивает эту метку входному параметру композиции. Это позволяет для всех сервисов данных, имеющих такую же метку, определить возможность передачи данных на вход созданной композиции. В пользовательском интерфейсе ввода данных реализован список сервисов и их композиций, для которых табличные данные могут переданы. Соответственно, применение композиции приводит к тому, что композиция автоматически попадает в список для всех сервисов данных с такой меткой и распространяется среди всех пользователей, производящих сбор данных (гербарных листов).

5.2. Методика создания композиций сервисов для моделирования загрязнений воздуха

Построение модели предметной области

Создание сервисов данных

На основе сервисов ввода и редактирования для задач мониторинга Байкальской природной территории созданы следующие сервисы данных:

1) Roads – сеть автодорог города Улан-Батора с указанием загруженности;
2) Heatings – печное отопление города Улан-Батора с указанием количества выделяемых вредных веществ;
3) Power stations – ТЭЦ, котельные города Улан-Батора с указанием количества выделяемых вредных веществ;
4) данные атмосферно-почвенного измерительного комплекса (АПИК) в Иркутске;
5) содержание SO2 (Ангарск, Иркутск, Шелехов, Байкальск, Селенгинск, Улан-Удэ, Гусиноозерск);
6) содержание NO2 (Ангарск, Иркутск, Шелехов, Байкальск, Селенгинск, Улан-Удэ, Гусиноозерск);

7) содержание O3 (Ангарск, Иркутск, Шелехов, Байкальск, Селенгинск, Улан-Удэ, Гусиноозерск);
8) содержание CO (Ангарск, Иркутск, Шелехов, Байкальск, Селенгинск, Улан-Удэ, Гусиноозерск);
9) физико-химические показатели оз. Гусиное по сезонам;
10) концентрации металлов в донных отложениях;
11) концентрации металлов в воде;
12) БД по содержанию примесей в атмосферном воздухе населенных мест в пожароопасные периоды Иркутской области, Забайкальского края;
13) БД спутникового мониторинга термальных точек (лесных пожаров) по данным прибора AVHRR спутников серии NOAA (POES).

Разработка сервисов обработки данных

roadToGrid – сервис расчета плотности линейных объектов в ячейках регулярной сетки. На входе сервиса слой векторных объектов. На выходе – общая длина участков линейных объектов, находящихся в ячейках регулярной сетки в формате GeoTIFF. Пользователь может задать размер ячейки и прямоугольную область обработки. Сервис для каждой ячейки регулярной сетки подсчитывает длину всех линейных объектов, проходящих в ее рамках.

Автоматическое создание композиций сервисов для инвентаризации загрязнений воздуха в г. Улан-Батор

Предложенная методика апробирована на задаче создания композиции сервисов инвентаризации загрязнения четырьмя веществами (сажа, парниковые газы SO2, NO3 и CO) для заданного региона. Композиция сервисов учитывает загрязнения от точечных источников (домов, юрт, котельных) и линейных объектов (дороги) при известном среднем загрязнении за определенный период от точечного объекта и общего объема загрязнений автотранспорта.

Загрязнение атмосферного воздуха – одна из наиболее важных проблем города Улан-Батор. Основной целью работы является разработка стратегии решения проблемы загрязнения воздуха в городе Улан-Батор на основе современных технологий и методов математического моделирования. Для формирования

прогноза концентрации было предложено использовать камерную модель распространения примесей в атмосфере. Для идентификации модели была собрана и рассчитана информация по расположению источников выбросов и их интенсивности (ТЭЦ, юрты и автотранспорт). При проведении сценарных расчетов начальное состояние системы было определено на уровне ПДК. Рассматривалось два слоя, выбросы от ТЭЦ учитывались во втором слое, а остальные выбросы – в первом. Расчеты проводились на период времени от трех суток до трех месяцев, причем рассматривался только отопительный сезон, когда выбросы от юрт и ТЭЦ максимальны. В сценарных расчетах учитывались следующие варианты направления ветра: безветренная погода, восточный, северный, северо-западный, северо-восточный. Для заданного региона инвентаризация загрязнений проводилась по следующим веществам: сажа, парниковые газы SO2, NO3 и CO. Целью инвентаризации является набор растровых файлов, каждая ячейка которых содержит значение загрязнения местности по всем интересующим загрязняющим веществам, а также значение совокупного загрязнения (всего 5 растровых файлов).

Созданы сервисы данных, содержащие положение ТЭЦ, котельных, печного отопления, их мощности и оборудование. На основе статистических данных концентраций выбрасываемых загрязняющих веществ для каждого типа оборудования рассчитаны суммарные выбросы для каждого источника. Для учета загрязнений, получаемых от автомобильного транспорта, использовались статистические данные объема продаваемого топлива. В литературе найдены оценки средних выбросов на единицу топлива по веществам. Используя эти оценки, рассчитаны суммарные выбросы от всего автотранспорта, которые необходимо пространственно распределить по сети автодорог с учетом их загрузки. Выделено несколько классов загруженности дорог. Например, концентрация выбросов многополосной дороги в центре города будет отличаться в несколько раз от однополосной грунтовой дороги. Для каждого класса экспертно задается коэффициент, который задает это отношение.

Для расчета выделяемых загрязнений от тепловых электростанций (ТЭЦ) и котельных была использована последовательность сервисов. Для этого пользователем применены сервисы *PowerStation*, *table2shp* и *vectorToGrid*. Сервис

PowerStation предоставил данные с координатами и суммарными выбросами, которые были сохранены в формат SHAPE при помощи сервиса *table2shp*. Затем сервис *vectorToGrid* произвел суммирование загрязнений от точечных источников по ячейкам регулярной сетки в формате GeoTIFF. Для расчета выделяемых загрязнений от печного отопления также была применена аналогичная последовательность для сервиса данных *Heatings*.

Для расчета вклада автомобильного транспорта в загрязнение воздуха в ячейках регулярной сетки используется сервис *roadToGrid*. На вход сервиса предоставляется сеть автодорог формате SHAPE с указанием коэффициентов загруженности и суммарные выбросы. Сеть автодорог получена с помощью сервиса *table2shp* и сервиса данных *Roads*. Сервис *roadToGrid* производит распределение суммарных выбросов по ячейкам регулярной сетки с учетом общей длины автодорог в каждой ячейке и их коэффициентов загруженности. Результат работы сервиса предоставляется в формате GeoTIFF.

Для получения конечного результата (файла в формате GeoTIFF) суммируются загрязнения, полученные от печного отопления, котельных, ТЭЦ и автотранспорта с помощью сервиса *raster_sum*. Далее результат публикуется в виде карты с помощью сервиса *Map*.

На основании действий пользователя (выполнение сервисов) сформирована статистика. На основе статистики применения сервисов автоматически создана композиция сервисов (рисунок 5.12).

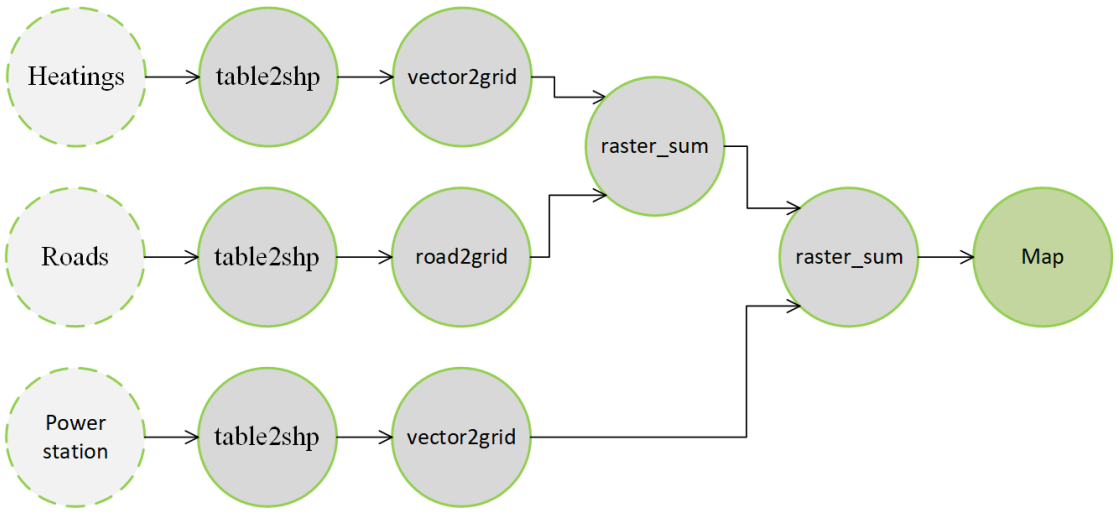


Рисунок 5.12 – Композиция сервисов инвентаризации загрязнений воздуха

Созданная композиция сервисов имеет следующие входные параметры:

1) сервис данных – сеть автодорог с указанием загруженности;
2) сервис данных – печное отопление с указанием количества выделяемых вредных веществ;
3) название атрибута сервиса данных печного отопления, в котором указывается количество выделяемых вредных веществ;
4) сервис данных – ТЭЦ и котельные с указанием количества выделяемых вредных веществ;
5) название атрибута сервиса данных ТЭЦ и котельных, в котором указывается количество выделяемых вредных веществ;
6) размер ячейки создаваемых растровых файлов;
7) экстент области, для которой необходимо произвести расчет;
8) суммарные выбросы автотранспорта.

Результатом выполнения композиции является набор растровых файлов, где каждый файл содержит распределение и концентрации одного из веществ. На рисунке 5.13 изображено распределение одного из веществ на интерактивной карте. Сервис отображения пространственных данных окрашивает растр в зависимости от концентрации загрязняющих веществ и правил, заданных в легенде.

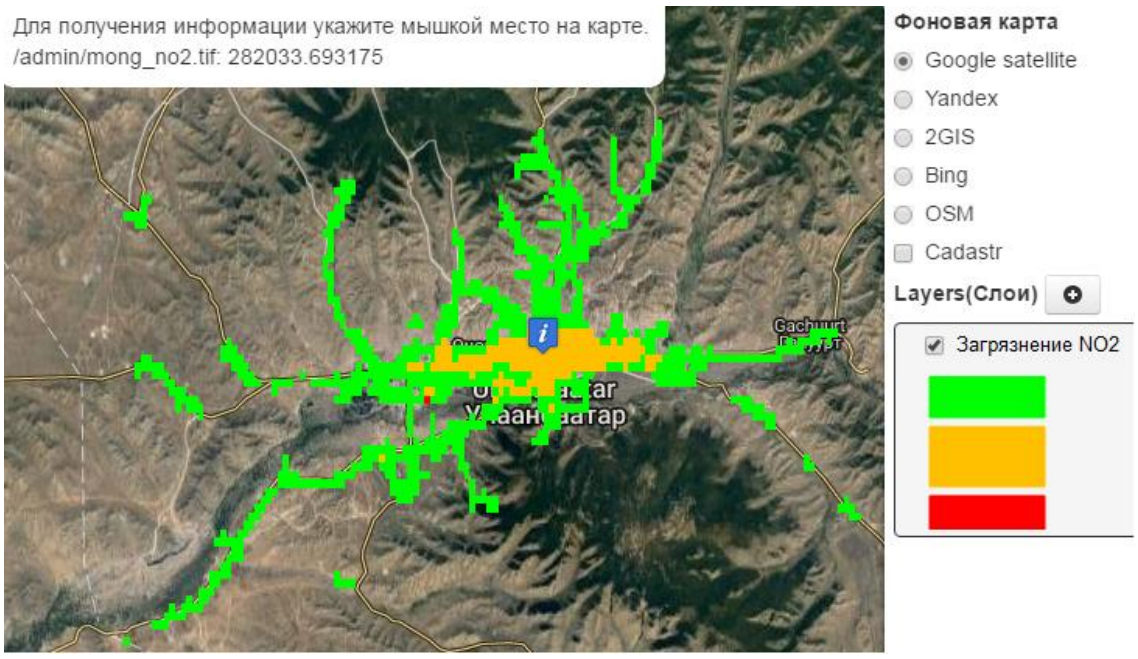


Рисунок 5.13 – Результаты выполнения композиции

Автоматически созданная композиция сервисов применялась многократно для расчета различных сценариев, учитывающих смену типа оборудования в разных районах города. Подсистема выполнения значительно уменьшила время расчета сценария в распределенной вычислительной среде, что позволило увеличить количество возможных сценариев.

5.3. Методика создания композиций сервисов на основе JavaScript

Разработка композиции сервисов с помощью языка JavaScript применялась для расчета временной доступности инфраструктурных объектов. Решение этой задачи требовало реализации обработки промежуточных данных с помощью средств языка. Кроме того, количество заданий в композиции заранее неизвестно и зависит от исходных данных.

Расчет временной доступности требуется при проектировании дорожной сети и расчете ее пропускной способности, планировании маршрутов общественного транспорта, выборе расположения строительства объектов инфраструктуры и т. д. В результате расчета получается растровый файл, в каждой ячейке которого находится время, необходимое, чтобы добрать до ближайшего объекта. Расчет времени производится с учетом дорожной сети, пешей доступности, различных преград, таких как водные объекты, здания, карьеры и т. д. Также необходимо учитывать путепроводы, проходящие через них, например, мосты. Для расчета временной доступности различных объектов инфраструктуры разработаны отдельные сервисы, установленные на нескольких вычислительных узлах локальной облачной инфраструктуры.

Данная композиция сервисов использовалась для оценки доступности школ. Пользователю по каждой точке необходимо предложить несколько ближайших школ. Для ускорения поиска ближайших школ требуется построить карту временной доступности для каждой школы отдельно.

Построение модели предметной области

Создание сервисов данных

Созданы сервисы данных с помощью Фабрики сервисов ввода и редактирования реляционных данных. Произведена загрузка данных из топоосновы в формате SHAPE и из открытых источников.

Roads – сервис данных, предоставляющий дорожную сеть с указанием скорости движения.

Barrier – сервис данных, предоставляющий в виде полигональных объектов различные преграды, на которых не возможно движение, например, реки и озера.

Schools – сервис данных, предоставляющий в виде точек инфраструктурные объекты, до которых производится расчет временной доступности.

Разработка сервисов обработки данных

Для решения задачи использовались следующие сервисы обработки данных, разработанные на основе библиотеки GDAL:

shp_to_geojson_converter – конвертирует SHP файл в формат GeoJSON, который может быть преобразован в стандартный объект JavaScript;

geojson_to_shp_converter – конвертирует данные в формате GeoJSON в SHP файл;

bufferize_vector – создает буферные зоны вокруг точечных объектов переданного SHP файла.

Для задачи специально разработан сервис road_analysis, который выполняет анализ доступности точечных объектов на основании данных о дорожной сети и преград. Результатом работы сервиса является регулярная сетка в растровом формате данных GeoTIFF, где в каждой ячейке содержится время достижения от этой точки до ближайшего объекта инфраструктуры. Работа сервиса основывается на алгоритме Дейкстры. Время прохождении ячейки рассчитывается на основе заданных пользователем ее размеров, скорости движения фрагментов дорожной

сети. В ячейках, в которых отсутствует дорожная сеть и преграды, время прохождения рассчитывается, используя скорость движения пешехода (5 км/ч).

Разработка композиции сервисов на языке JavaScript

Расчет временной доступности необходимо выполнить для каждого образовательного учреждения отдельно. Сервис данных Schools предоставляет информацию о положении образовательных учреждений. С помощью сервиса table2json данные можно сохранить в файл. Затем требуется произвести перебор всех организаций и для каждого отдельно создать растр временной доступности.

Разработка композиции сервисов на языке JavaScript производится в каталоге сервисов. Перед заданием кода сценария, задающую композицию, пользователю необходимо определить входные и выходные параметры. В случае рассматриваемой задачи входными параметрами будут:

1) сервис данных с точечными объектами инфраструктуры;
2) сервис данных с указанием преград с полигональными объектами;
3) сервис данных с линейными объектами, содержащий объекты дорожной сети;
4) область производимого расчета (расчет производится для тех объектов инфраструктуры, которые попадают в выделенную область);
5) размер ячейки.

В качестве выходного параметра пользователь указывает папку, в которую необходимо сохранить результаты работы.

После определения входных и выходных параметров композиции сервисов каталог автоматически формирует модуль и шаблон функции, которая будет вызываться при использовании композиции сервисов. При задании кода композиции сервисов пользователь использует специальное текстовое поле с подсветкой синтаксиса языка и ему предоставляется список доступных для использования сервисов.

Код композиции сервисов представлен ниже. Первоначально производится сохранение данных в формат GeoJSON. Затем формируется цикл по всем объектам, представленным в GeoJSON. Каждый объект сохраняется в файл формата SHAPE.

Далее строится буферная область вокруг объекта в виде растра и вызывается сервис road_analysis, который строит искомый растр.

```
function school_avialablity(input, output){
        var dir = input.dir.substring(0, input.dir.lastIndexOf("\\")+1);
        var schools_json = new ValueStore();
        table2json({source: input.schools}, {ResultJSON: schools_json});
        var schools = JSON.parse(schools_json.get());
        for(var I = 0; I < schools.features.length; i++){
                var sch = new ValueStore();
                var sch_jsn={ "type": "FeatureCollection",   "features": [school_json.features[i]]}
                sch.set(sch_jsn);
                var shape = new ValueStore();
                geojson_to_shp_converter({json:sch }, {shape: shape });
                var tiffbuf = new ValueStore();
                bufferize_vector({input: shape, extent: input.extent, cellsize: input.cellsize },
                        {buf: tiffbuf });
                var resltfile = new ValueStore();
                resltfile.set(dir+"\\"+i+".TIFF");
                road_analysis({targets: tiffbuf, roads: input.roads, barrier: input.barrier,
                        extent: input.extent, cellsize: input.cellsize }, {res: resltfile })
        }
}
```

На рисунке 5.14 показана форма ввода входных параметров сервиса. Перед выполнением необходимо задать входные параметры:

- параметр schools определяет сервис данных, содержащий объекты, до которых рассчитывается временная доступность. Количество объектов, для которых выполняется расчет, возможно ограничить, воспользовавшись фильтрами;
- входные параметры barrier и roads являются сервисами данных, содержавшие линейные объекты с объектами дорожной сети и естественных преград соответственно;

- параметр extent определяет область, для которой выполняется расчет временной доступности объектов, область можно выделить на карте (рисунок 5.15);
- параметр cellsize определяет размер ячейки.

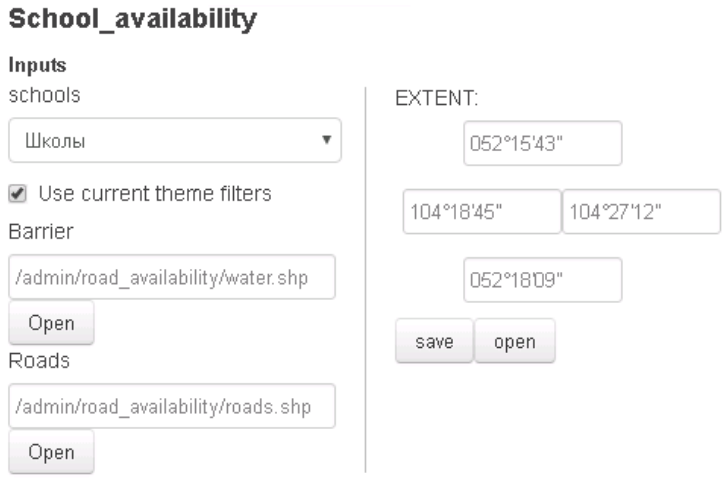


Рисунок 5.14 – Входные параметры сервиса

Рисунок 5.15 – Выбор объектов на карте

Результатом выполнения сценария является набор растровых файлов, соответствующих рассматриваемым объектам образовательной инфраструктуры.

Полученные растровые файлы могут быть отображены на интерактивной веб-карте Геопортала.

На рисунке 5.16 представлен автоматически создаваемый ациклический граф зависимости заданий по данным. DAG строится по мере выполнения сценария. Например, задание с идентификатором 1001 соответствует вызову функции shp_to_geojson_converter, результаты которой необходимы для запуска функции geojson_to_shp_converter в теле цикла. В теле цикла вызываются задания geojson_to_shp_converter, bufferize_vector, road_analysis, что соответствует последовательностям заданий 1002 – 1003 – 1004, 1005 – 1006 – 1007 и т. д.

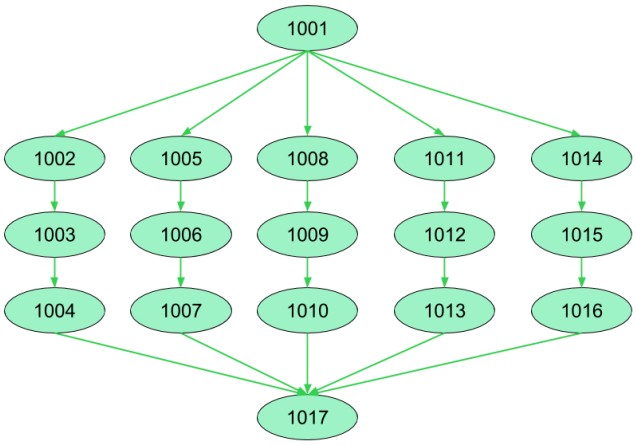


Рисунок 5.16 – Направленный ациклический граф зависимости заданий по данным

Так как среда, в пределах которой развернуты распределенные вычислительные сервисы, является гетерогенной, в процессе выполнения будут искусственно отключаться некоторые вычислительные узлы и сервисы в целях имитации изменений вычислительной среды. При наступлении событий, перечисленных выше, должно осуществляться перепланирование выполнения сервисов.

В случае отсутствия изменений в вычислительной среде по мере выполнения сценария, расписание выглядит согласно представленному на рисунке 5.17. На рисунке видно, что вычислительные узлы не простаивают, задания компактно выполняются на всех четырех вычислительных узлах.

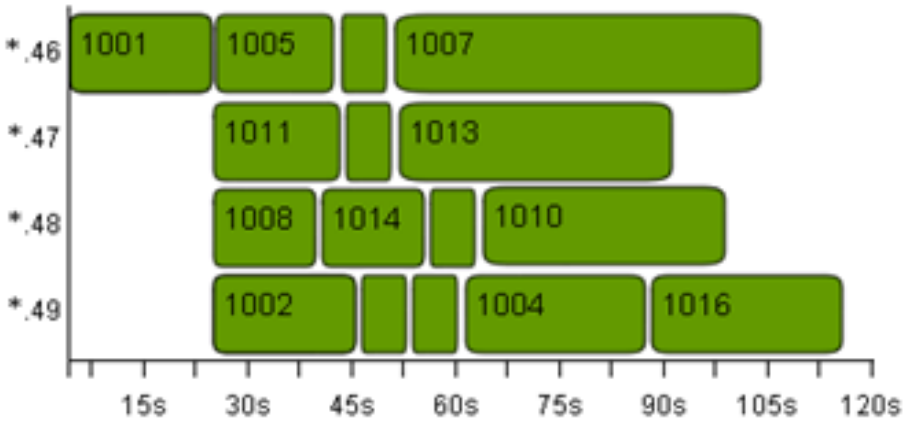


Рисунок 5.17 – Расписание выполнения композиции сервисов

При отключении двух из четырех вычислительных узлов алгоритм планирования обрабатывает данную ситуацию и перестраивает расписание. На расписании, представленном на рисунке 5.18, видно, что два вычислительных узла были отключены некоторое время. По истечении некоторого времени с момента обнаружения неисправности вычислительного узла алгоритм пробует выполнить запрос на удаленном вычислительном узле и в случае ответа узла сервисы снова назначаются на этот вычислительный узел.

Рисунок 5.18 – Расписание в случае отключения вычислительных узлов

В случае, когда вычислительный узел не в состоянии выполнить определенный сервис (например, когда узлы .46 и .47 имеют некорректные копии сервиса bufferize_vector, выполняющегося наименьшее время), расписание перестраивается с учетом невыполнимости определенного сервиса на некоторых узлах. На рисунке 5.19 видно, что узлы .46 и .47 не выполняют сервис bufferize_vector.

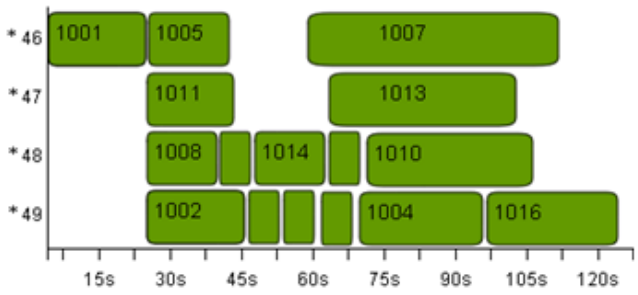


Рисунок 5.19 – Расписание в случае невыполнимости сервиса на некоторых узлах

По мере завершения сценария результаты расчета загружаются на Геопортал и визуализируются с помощью интерактивной веб-карты. На рисунке 5.20 представлен скриншот демонстрации результатов расчета временной доступности для образовательного учреждения. С помощью инструментов по работе с интерактивной картой геопортала ячейки отображаемых растровых файлов были окрашены в соответствии с попаданием их значений в интервалы 0-10 минут, 10-20 минут, и т. д. Стоит отметить, что на растровых файлах присутствуют пустые ячейки, которые соответствуют естественным преградам (водоемы).

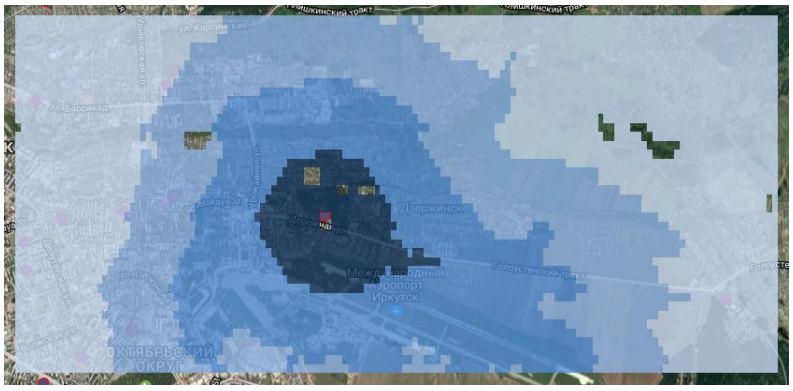


Рисунок 5.20 – Визуализация расчета временной доступности

Текущую постановку задачи невозможно решить, используя только существующие сервисы. Набор сервисов изначально разработан для решения других задач. Решение текущей задачи требует обработки промежуточных данных. Разработка отдельного сервиса для обработки промежуточных данных является неоправданной по двум причинам:

1) сервис будет специфичным, и маловероятно, что будет использоваться в других задачах;

2) разработка отдельного сервиса является более трудоемкой по сравнению с применением средств языка JavaScript.

Применение языка JavaScript для создания композиций сервисов упростило решение задачи за счет возможности обработки данных с помощью средств языка JavaScript, позволило динамическое определение множества задач и применение методов планирования выполнения композиции сервисов, улучшило надежность выполнения композиции за счет перепланирования выполнения в случае сбоев вычислительных узлов.

5.4. Методика создания композиций сервисов для обработки данных дистанционного зондирования Земли

Байкальская природная территория обладает обширными и частично труднодоступными площадями. Большая площадь, сложный рельеф и слабая доступность БПТ обосновывают актуальность использования данных ДЗЗ (спутники миссии Sentinel-2, Канопус-В) для ведения цифрового экологического мониторинга. Данные спутников миссии Sentinel-2 (семейство спутников дистанционного зондирования Земли Европейского космического агентства, созданное в рамках проекта глобального мониторинга окружающей среды и безопасности Copernicus) активно используются для мониторинга растительности, лесных и водных ресурсов, использования земель, ликвидации последствий стихийных бедствий и т. д. Данные ДЗЗ, полученные с космических аппаратов Sentinel-2A, 2B, имеют разрешение от 10 до 60 м в видимой, ближней инфракрасной (VNIR) и коротковолновой инфракрасной (SWIR) зонах спектра, включающих в себя 13 спектральных каналов, что гарантирует отображение различий в состоянии растительности, в том числе и временные изменения. Наличие в миссии двух спутников позволяет проводить повторные съемки каждые 5 дней на экваторе и каждые 2–3 дня в средних широтах. Доступ к снимкам предоставляется геологической службой USGS через браузер или через запрос к хранилищу Google Cloud.

Создание сервисов данных

С помощью Фабрики сервисов ввода и редактирования реляционных данных разработаны следующие сервисы данных:

1) каталог данных ДЗЗ;
2) описание классов подстилающей поверхности ДЗЗ;
3) обучающая выборка.

Каталог данных ДЗЗ

Поиск и загрузка необходимого снимка в службе USGS занимает продолжительное время. Поэтому для организации удобного поиска космоснимков и быстрого доступа к ним в рамках каталога данных ДЗЗ, добавлена возможность отображения снимков на карте и реализованы компоненты загрузки космоснимков в СХД. Компоненты загрузки реализованы в виде скрипта, запускаемого с заданной периодичностью. При запуске скрипт позволяет указать параметры облачности, выбрать временной промежуток. При этом в каталог данных ДЗЗ добавляется следующая информация: дата съемки, тип сенсора, процент облачности, путь в СХД, ссылка на первоисточник. Каждый снимок представлен на карте в виде ограничивающего прямоугольника. Отображаются космоснимки в двух режимах (для Sentinel-2 это native RGB и RGB на основе каналов B12, B8A, B04). Реализован поиск снимков (рисунок 5.21) с учетом сенсора, облачности, положения и даты.

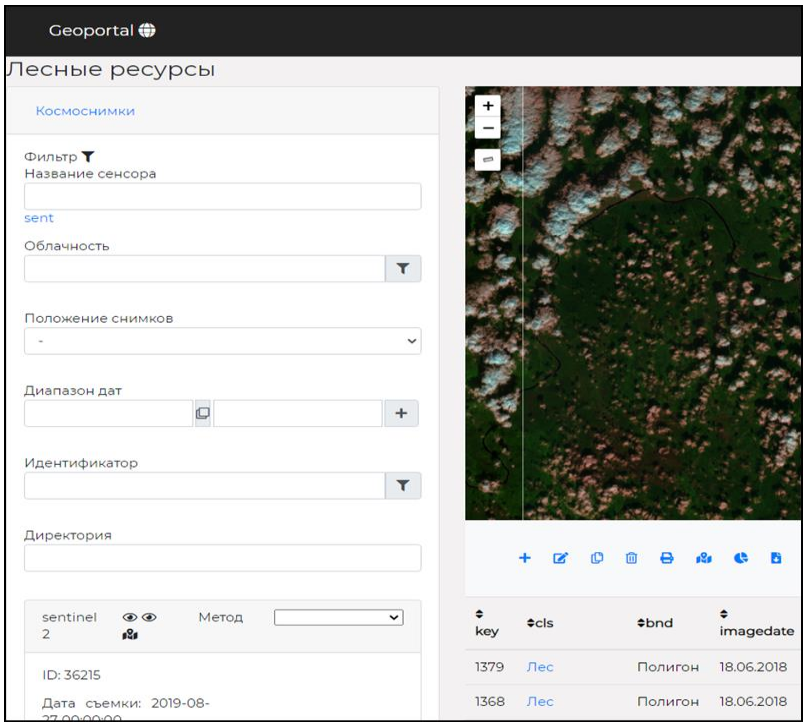


Рисунок 5.21 – Форма поиска снимка

Поиск по положению позволяет найти снимки внутри экстента, либо наоборот, искать снимки, которые содержат указанный пользователем прямоугольник. Добавлен поиск в каталоге снимка по его идентификатору.

На конец 2023 года в каталоге собрано более 100 тысяч космоснимков Sentinel-2 и Landsat-8 территорий Иркутской области и Республики Бурятия общим объёмом более 180 терабайт.

Данные беспилотных летательных аппаратов (БПЛА) используются при формировании обучающей выборки для классификации космоснимков. Качество обучающей выборки критически важно для корректной работы методов классификации. Наличие в разметке полигонов, содержащих ошибки в классе, например, хвойный лес вместо лиственного или неточные границы, приводит к значительному ухудшению работы классификаторов. Разметка производится на основе данных Sentinel-2. Некоторые классы довольно сложно различать на космоснимках, поэтому для формирования качественной выборки используются данные беспилотных летательных аппаратов. Затем полученные данные обрабатываются с помощью программного обеспечения Agisoft для получения ортофотопланов (рисунок 5.22), которые помещаются в СХД и регистрируются в каталоге. После этого данные БПЛА можно использовать в качестве подложки для уточнения разметки космоснимков.

Рисунок 5.22 – Фрагмент ортофотоплана

Обучающая выборка

Данные ДЗЗ (космоснимки) любой территории, в том числе Байкальской природной территории, имеют специфические спектральные и текстурные характеристики, поэтому при мониторинге активно применяются методы классического обучения с учителем [31], позволяющие учитывать эти характеристики. Эти методы дают результаты высокой точности (до 99%) при сегментации космоснимков среднего разрешения и позволяют для каждого пикселя космоснимка определить его класс – растительность, открытая почва, вода, антропогенные объекты и т. д. Применение этих методов для исследования задач классификации космоснимков требует подготовки обучающей выборки на заданную территорию. Исследования проводились для БПТ и прибрежных территорий озера Байкал, включая особо охраняемые территории. В открытом доступе на исследуемую территорию нет наборов данных нужной точности о типе земной поверхности.

В рамках проекта применены современные методы обработки данных ДЗЗ и реализован сервис классификации космоснимков Sentinel-2 с учетом специфики БПТ.

В современных исследованиях по оценке и прогнозированию состояния окружающей среды территорий [30] активно используются мультиспектральные космоснимки со спутников Sentinel-2, которые имеют самое высокое разрешение (10 метров) из свободно распространяемых, что позволяет различать типы поверхности (вода, лес, дороги, здания) и отслеживать лесные пожары, вырубки. Съемка территорий спутниками Sentinel-2 проводится в среднем каждые 5 дней. Космоснимки состоят из 13 спектральных каналов, четыре из которых с разрешением 10 метров, остальные – 20 и 60 метров. Для классификации все каналы снимков исследуемой территории приведены к разрешению 10 метров. Для поддержки исследований создан каталог космоснимков. Пример исходных данных представлен на рисунке 5.23.

Рисунок 5.23 – Sentinel-2, южная часть БПТ

При формировании обучающей выборки было необходимо определить набор классов, решающих следующие задачи:

- классифицировать всю исследуемую территорию с максимально возможной точностью;
- определить классы, наиболее полезные для решения практических задач.

Природные особенности ландшафтов Байкальской природной территории, большая часть которой занята горной тайгой, а в средней части преобладает лесостепной ландшафт, обусловили состав обучающей выборки. Лесообразующие породы включают в себя хвойные (сосна, лиственница, кедр) и лиственные (береза, осина) деревья.

В высокогорном поясе и поймах рек распространены кустарниковые заросли, поэтому в обучающей выборке лесные ресурсы представлены 5 классами: хвойный, лиственный и смешанный лес, редколесье, кустарники. На БПТ проводятся активные заготовительные рубки леса, в том числе и нелегальные, для их учета к набору добавлен класс «Вырубки».

Значительную и важную часть исследуемой БПТ занимают водные объекты и горные системы, для обработки которых добавлены классы «Голая скала» и «Вода». На исследуемой территории развит аграрный комплекс, который на текущий момент представлен классами «Однолетние сельскохозяйственные культуры» и «Пастбище». Для оценки антропогенного влияния на территорию добавлен класс «Жилая зона».

На космоснимках БПТ часто присутствуют облака, влияющие на качество классификации. Маски облаков, представленные в наборе данных Sentinel-2, недостаточно точны, поэтому для учёта их влияния добавлен класс «Облака».

Построение обучающей выборки проводилось с помощью сервисов ввода и редактирования реляционных данных (рисунок 5.24) на основе данных Sentinel-2 и полевых исследований.

Рисунок 5.24 – Разметка космоснимка полигонами с заданными классами

Разметка осуществлялась полигональными объектами с указанием класса, директории космоснимка, даты съемки. В форме ввода автоматически подставляется путь к снимку, его идентификатор и дата снимка (рисунок 5.25).

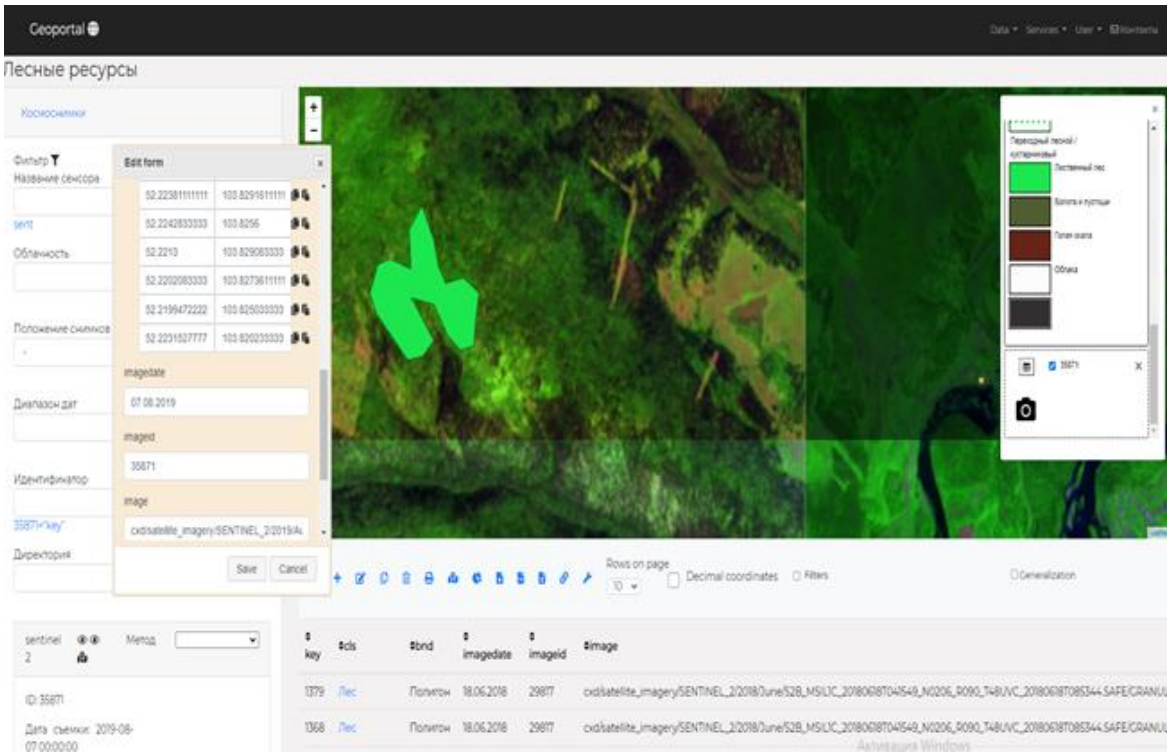


Рисунок 5.25 – Подсистема разметки космоснимков

Всего размечены 134 космоснимка Байкальской природной территории, количество полигональных объектов – 2246 шт., на площади более 5900 кв.км.

(примерно 230 млн. пикселей, одна территория может быть размечена для нескольких снимков). Все снимки летнего периода за 2018-2020 года.

Идентификация классов лесов проводилась на основе комбинации каналов SWIR (2185,7-2202,4 мкм), NIR (864.0-864.7 мкм) и RED (664.6-664.9 мкм), позволяющей отделять породы по интенсивности оттенка зеленого цвета на полученном изображении [79].

Сервис классификации данных Sentinel-2 на основе нейронной сети ResNet50

Для классификации космоснимков Sentinel-2 применялась сверточная нейронная сеть ResNet50. Архитектура сети представлена на рисунке 5.26.

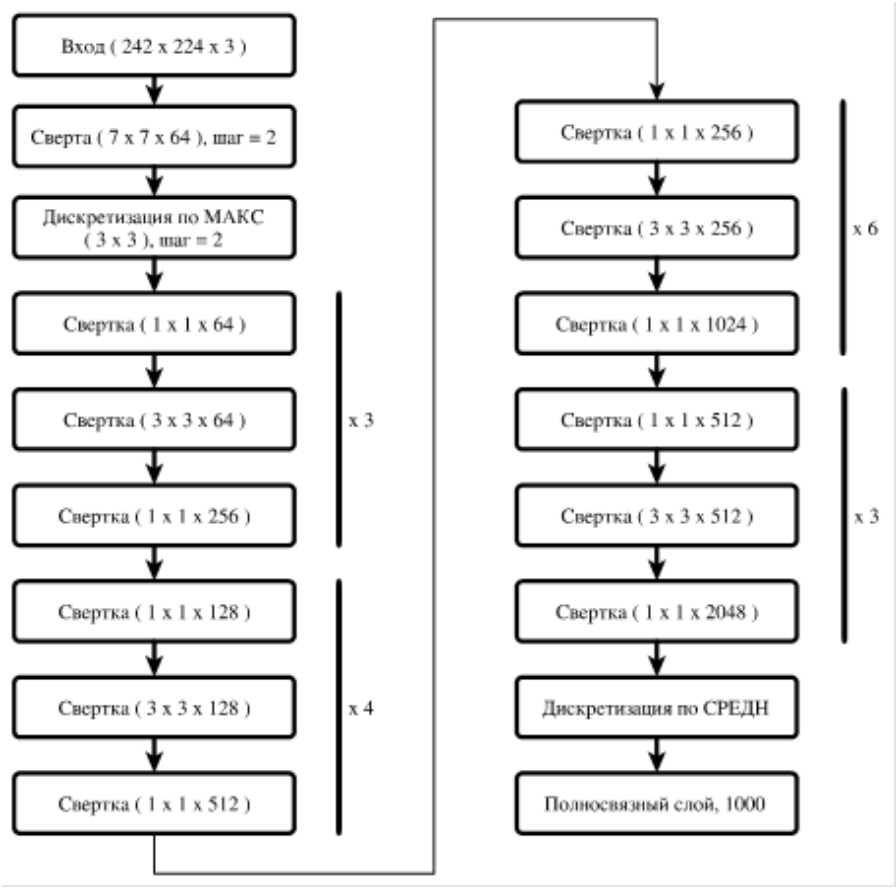


Рисунок 5.26 – Архитектура сети ResNet50

Входными данными для нейронной сети являются тензоры. Размерность тензоров определена окрестностью, на основе которой сеть должна принимать решение и количеством слоев. Окрестностью выбрана ячейка размером 64x64 пикселя, т. е. 640 на 640 метров. Классификация редколесья, переходного леса

требует такой размерности. Количество слоев – 14, 13 из них – исходные слои Sentinel-2, 14 слой – это локальные бинарные шаблоны (LBP), подсчитанные на слое B2. Добавление слоя LBP позволило улучшить качество классификации по некоторым классам до 10 %. Добавление различных индексов, например, NDWI, не привело к улучшению результата. Далее тензор будем называть образцом. В будущем предполагается включение дополнительных слоев, таких как высота над уровнем моря, уклон, экспозиция, полученных на основе данных SRTM. Кроме того, планируется включение метеоданных, полученных на основе обработки данных развиваемой сети метеостанции и данных ДЗЗ. Метеоданные и рельеф могут значительно улучшить классификацию лесных ресурсов и почвенного покрова.

Разработан алгоритм преобразования разметки космоснимков в обучающую выборку (набора образцов), который состоит из следующих этапов:

1. Подготовка космоснимков. Приводятся все каналы к разрешению 10x10 метров (gdal_translate), производится вычисление LBP (локальных бинарных шаблонов).
2. Подготовка маски для каждого космоснимка. Создается векторный файл с объектами в проекции космоснимка. Включаются векторные объекты, размеченные только для этого космоснимка. Формируется маска объектов с помощью утилиты gdal_rasterize.
3. Сканирование масок космоснимков с определенным шагом (рисунок 5.27). Если центральный пиксель не фон, то формируется образец на основе космоснимка и сохраняется в виде тензора.

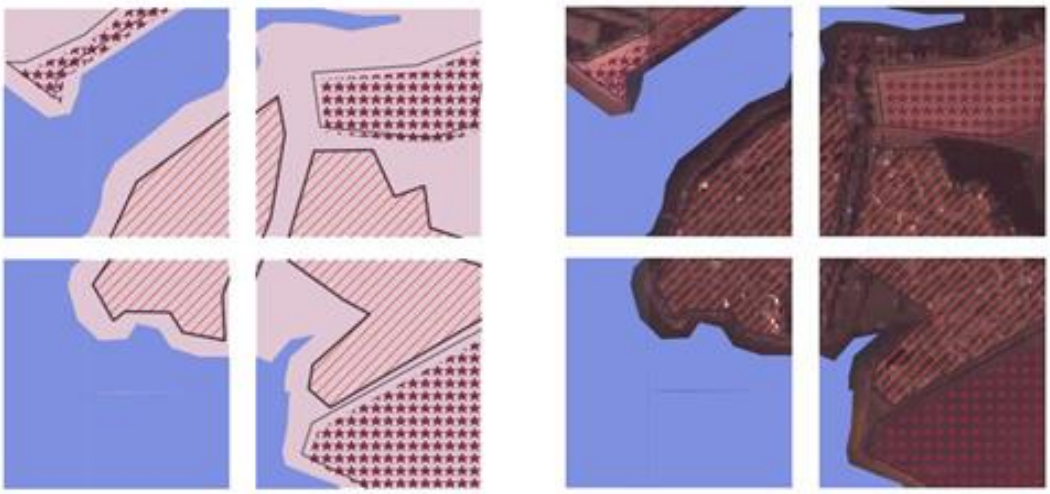


Рисунок 5.27 – Совмещение маски (слева) с космоснимком (справа)

На текущий момент по выполненной разметке сформировано следующее количество образцов по классам (таблица 5.1):

Таблица 5.1. Количество образцов по классам

Идентификатор	Класс	Количество
1	AnnualCrop	321675
6	Pasture	338398
8	Residential	6403
10	SeaLake	444956
11	Mixed forest	117786
12	Woodland	19854
13	Logging forest	18465
15	Coniferous forest	49503
17	Transitional woodland shrub	21694
18	Leaved forest	88975
21	Bare rock	111192
26	Cloud	50396

По каждому классу получено разное количество образцов. Поэтому требуется балансировка обучающей выборки, чтобы каждый класс был представлен примерно одинаковым количеством образцов.

Кроме того, требуется балансировка образцов в рамках одного класса. Например, в разметке представлены преимущественно плотные облака. Поэтому сверточная нейронная сеть нацелена классифицировать именно их и недостаточно точно выделяет перистые облака. Для балансировки в рамках класса используется кластеризация образцов. Для каждого канала образца выделяются следующие характеристики:

- среднее значение канала;
- среднеквадратическое отклонение.

Далее проводится кластеризация и для каждого кластера отбирается равное количество образцов. Применение кластеризации обучающей выборки также позволило улучшить качество классификации.

Общее число настраиваемых параметров ResNet50: 23 538 338. Используется алгоритм оптимизации: Adam, betas=(0,9; 0,99); функция потерь: FlattenedLoss; Обучение на 80 эпохах на компьютере со следующими характеристиками: процессор Intel(R) Core (TM) i9-9900X CPU с тактовой частотой 3.50GHz, объем ОЗУ 125 Гб, две графических карты NVIDIA GeForce RTX 2080SUPER 8ГБ проводится в течение 80 минут. Классификация одного снимка занимает 9 минут. Результаты классификации сохраняются в формате GeoTIFF.

Разработан WPS сервис, выполняющий классификацию космоснимка. Для реализации сервиса был выбран сервер Flask-PyWPS – одна из реализаций стандарта Web Processing Service (WPS) на основе Python. Выбор Flask-PyWPS обусловлен тем, что позволяет совмещать его с библиотеками обработки изображений, такими как GDAL, OpenCV и т.д. Сервис классификации принимает на вход путь к директории, где содержатся растровые файлы в формате GEOTIFF, соответствующие 13 слоям космоснимка Sentinel-2, приведенные к единому пространственному разрешению. Результатом работы сервиса является сегментированное изображение, где в каждом пикселе указан его класс. Из исходного космоснимка сервис вычисляет слой с локальными бинарными шаблонами и формирует множество тензоров формы Nx64x64x14, где N – число образцов размера 64x64 по 14 слоям, и подает его на вход

нейронной сети. Результаты классификации сохраняются в файл (рисунок 5.28) в формате GEOTIFF [35-36].

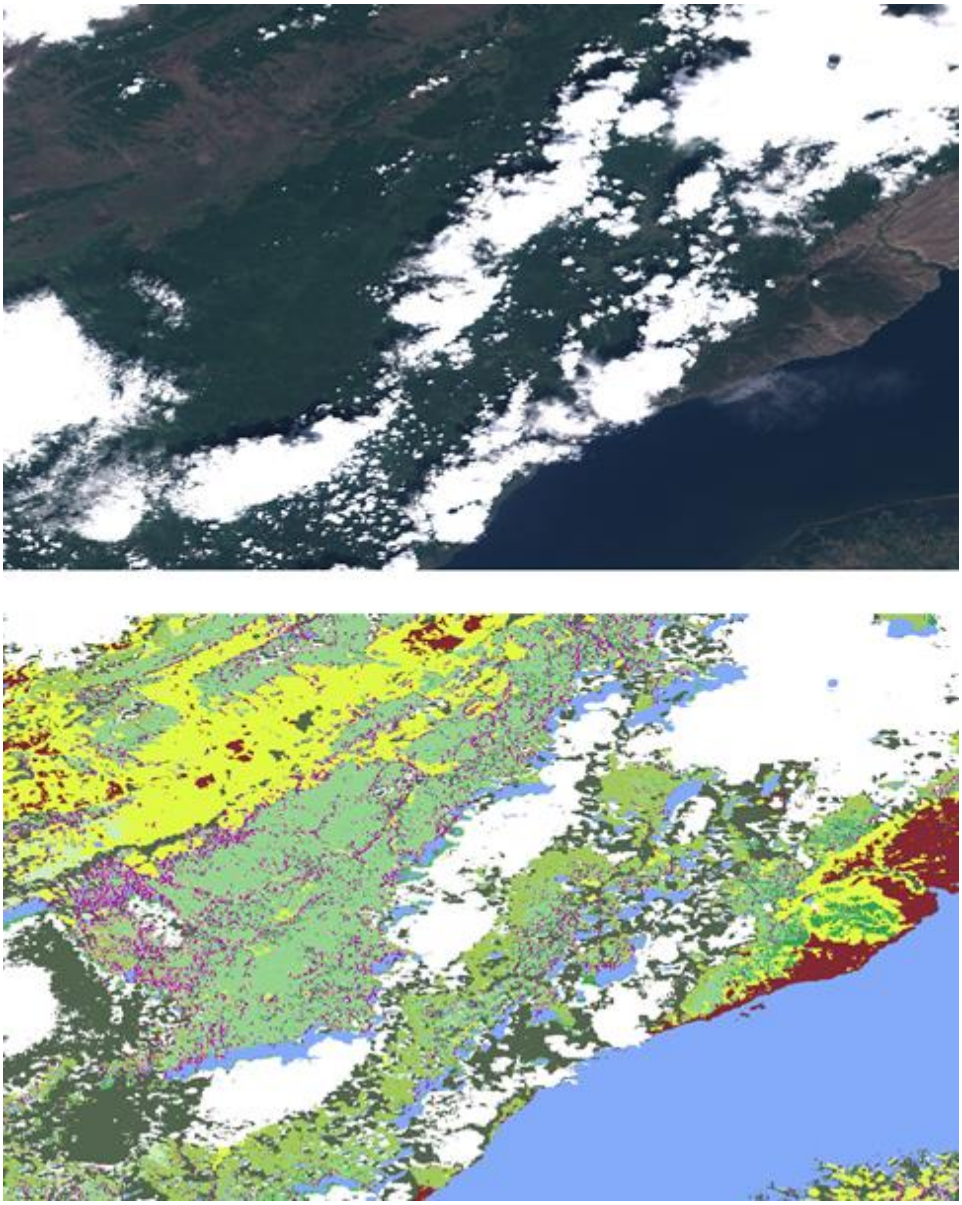


Рисунок 5.28 – Результаты классификации космоснимка (сверху), нейронной сетью (снизу)

Сервис оценки результатов классификации

Оценить точность классификации можно по различным метрикам. Одной из наглядных и включающих множество статистических данных метрик является матрица ошибок. Матрица ошибок представляет собой инструмент, использующий кросс-табуляцию. На ее основе можно увидеть, как соотносятся значения совпадающих классов, полученные из различных источников. В качестве

источников могут выступать, например, проверяемый растр (результат классификации) и опорный (точная/ручная разметка). Для построения матрицы могут использоваться все ячейки растра (пиксели) или выборка ячеек, формируемая случайно. Для построения матрицы ошибок использовалась библиотека pycm. Был реализован алгоритм выбора проверочных ячеек, в котором для каждого изображения проверочные ячейки соответствуют пользовательской разметке. Разработан сервис оценки результатов классификации. Сервис принимает на вход два изображения: результат классификации и маску классов. Алгоритм производит подсчет точности по 61 параметру, среди которых можно выделить ACC(Accuracy), AUC (Area under the ROC curve), PPV (Precision or positive predictive value), TPR (Sensitivity, recall, hit rate, or true positive rate) [36].

Для оценки результатов работы классификатора произведена разметка космоснимков, не участвовавших в обучении, и сформирована верификационная выборка. По результатам классификации проведено попиксельное сравнение с выполненной разметкой. В таблице 4 представлены результаты сравнения по классам. Средняя точность (вероятность верной классификации) составляет 0,9414.

Таблица 5.2. Результаты классификации на верификационной выборке

Класс	**Точность**
Пастбище	0,99976
Жилая зона	0,999868537
Вода	0,99697472
Смешанный лес	0,77193426
Редколесье	0,687642153
Вырубки	0,993470135
Хвойный лес	0,989869998

Переходный лес / кустарники	0,99976
Лиственный лес	0,972972973
Голая скала	0,94368231
Облака	0,999965272
Среднее по классам	0,941489124

Произведено сравнение работы классификаторов на верификационной выборке. Результаты сравнения показывают, что классификатор на основе ResNet50 работает лучше, чем на RandomForest, потому что сверточная нейронная сеть принимает решение по окрестности 64x64 пикселя. По результатам сравнения было выявлено, что классификаторы плохо отделяют классы «Редколесье», «Лиственный лес» и «Вырубки». Чтобы различать эти классы, требуется анализ серии космоснимков, полученных в разное время года. Разница в точности классификации на данных обучающей и верификационной выборок обосновывают необходимость расширения разметки и классификации серии космоснимков.

Полученные высокие оценки точности классификации позволяют применять предложенный метод по некоторым классам для решения актуальных задач Байкальской природной территории, в частности, для мониторинга состояния лесного фонда, оценки влияния изменений климата на ландшафт, анализа динамики застройки, инвентаризации сельхозугодий и т. д.

Выводы

Комплекс программных компонентов, реализующий модель сервис-ориентированной информационно-аналитической среды, активно используется на практике. Созданы более 200 сервисов данных, более 40 сервисов обработки данных. Развернуты 7 геопорталов, ориентированных на различные предметные области и коллективы. На основе созданных сервисов сформированы композиции сервисов, объединяющие сервисы данных, сервисы обработки и публикации. Впервые

разработана среда, которая обеспечивает создание композиций сервисов и их обмен между пользователями. Результаты исследований, представленные в данной главе, опубликованы в [168 - 209].

ЗАКЛЮЧЕНИЕ

В ходе выполнения диссертационного исследования проведен анализ инструментальных средств, технологий и существующих информационно-аналитических сред с использованием сервисов обработки пространственно-временных данных и средств создания композиций сервисов. Выявлено, что поиск новых композиций сервисов является комбинаторно сложной задачей, в которой необходимо проанализировать большое количество сервисов и предоставить наиболее релевантные их комбинации. Создание композиций сервисов ограничено сложностью взаимодействия сервисов по данным из-за различий в программном интерфейсе доступа к данным, в структуре, в используемых справочниках и т. д. Существующие методы и подходы на основе анализа метаданных и онтологий не позволяют значительно упростить поиск и создание композиций сервисов, поэтому разработка сервис-ориентированной информационно-аналитической среды обработки междисциплинарных пространственных данных, повышающей эффективность научных исследований за счет создания композиций сервисов и организации обмена ими, является актуальной задачей для развития междисциплинарных исследований. В процессе ее решения получены следующие новые результаты:

- вычислительная модель сервис-ориентированной информационно-аналитической среды обработки пространственно-временных данных междисциплинарных исследований, которая в сравнении с подобными моделями обеспечивает оценку композиций сервисов на основе многопользовательской статистики их применения;
- оригинальные алгоритмы и метод автоматического создания композиций сервисов на основе статистических данных использования сервисов, базирующиеся на применении предложенной модели. Использование метода может значительно упростить работу пользователя и автоматизирует часто повторяющиеся его действия. В результате применения модели и метода создания композиций опыт применения пользователем сервисов автоматически распространяется среди всех пользователей;

- оригинальный программный компонент выполнения композиций сервисов, заданных на процедурном языке, с обработкой промежуточных данных с помощью средств языка и его библиотек, в процессе выполнения которого формируется DAG, для которого одновременно обеспечивается динамическое планирование с применением алгоритма HEFT и выполнение в гетерогенной динамической вычислительной среде;
- основные компоненты среды, реализующие модель СОИАС и метод автоматического создания композиций сервисов, в частности, оригинальные программные инструменты «Фабрика сервисов ввода и редактирования реляционных данных», «Фабрика сервисов отображения пространственных данных», каталог данных и структурных спецификаций, каталог сервисов обработки данных, сервис конвертации реляционных данных;
- апробация СОИАС на задаче поддержки междисциплинарных научных исследований Байкальской территории. Комплекс программных компонентов, реализующий модель сервис-ориентированной информационно-аналитической среды, активно используется на практике. Созданы более 200 сервисов данных, более 40 сервисов обработки данных. Развернуты 6 различных геопорталов, ориентированных на различные предметные области и коллективы. На основе созданных сервисов сформированы композиции сервисов, объединяющие сервисы данных, сервисы обработки и публикации. Впервые среда обеспечивает создание композиций сервисов и их обмен между пользователями.

Результаты диссертации апробированы, опубликованы, обсуждены на семинарах и конференциях, использованы для решения ряда важных практических задач. В настоящее время разработанные программные компоненты специалистами из различных научных и образовательных организаций.

Предложенная в диссертации сервис-ориентированная технология проведения научных экспериментов допускает свое естественное развитие и обобщение применительно к другим предметным областям научных исследований.

ЛИТЕРАТУРА

1. Bih J. Service oriented architecture (SOA) a new paradigm to implement dynamic e-business solutions // Ubiquity. 2006. Vol. 4. P. 1–17. DOI: 10.1145/1162511.1159403
2. Foster I., Kesselman C., Tuecke S. The anatomy of the Grid: Enabling scalable virtual organizations // The International Journal of High Performance Computing Applications. 2001. Vol. 15, № 3. P. 200-222.
3. Foster I. What is the Grid? A Three Point Checklist // GRID today. 2002. Vol. 1. P. 32-36.
4. Набатов Д.Г. Проблемы межведомственного электронного взаимодействия // Тр. Ин-та государства и права Российской академии наук. 2013. № 2. С. 230-239.
5. OASIS [Электронный ресурс]. – Режим доступа: https://www.oasis-open.org/ (дата обращения 25.02.2024).
6. Lin B., Chen Y., Chen X., Yu Y. Comparison between JSON and XML in Applications Based on AJAX // Proceedings of 2012 International Conference on Computer Science and Service System. 2012. P. 1174-1177.
7. SOAP [Электронный ресурс]. – Режим доступа: https://www.w3.org/TR/soap/ (дата обращения 25.02.2024).
8. Lawrence K. Web Services Security: SOAP Message Security 1.1 (WS-Security 2004) // Security. 2006. Vol. 2003, № 2. P. 76.
9. Webber J. REST in practice // Lecture Notes in Computer Science. 2010. Vol. 6285. P. 7.
10. Schut P. OpenGIS ® Web Processing Service // Open Geospatial Consortium. 2007. № 6. P. 1-3.
11. Curbera F., Duftler M., Khalaf R., Nagy W., Mukhi N., Weerawarana S. Unraveling the Web services web: An introduction to SOAP, WSDL, and UDDI // IEEE Internet Computing. 2002. Vol. 6, № 2. P. 86-93.
12. Schema.org [Электронный ресурс]. – Режим доступа: https://schema.org (дата обращения 25.02.2024).
13. Liyang Yu. A Developer’s Guide to the Semantic Web: 2nd edition. Springer Publishing Company Incorporated, 2014. 608 p.
14. A JSON-based Serialization for Linked Data [Электронный ресурс]. – Режим доступа: https://www.w3.org/TR/json-ld (дата обращения 25.02.2024).

15. Di L., Zhao P., Yang W., Yue P. Ontology-Driven Automatic Geospatial-Processing Modeling Based on Web-Service Chaining // Proceedings of the Sixth Annual NASA Earth Science Technology Conference. College Park, MD, USA, 2006. P. 27-29.
16. Web Ontology Language (OWL) [Электронный ресурс]. – Режим доступа: https://www.w3.org/OWL/ (дата обращения 25.02.2024).
17. Dublin Core Metadata Initiative [Электронный ресурс]. – Режим доступа: https://www.dublincore.org/ (дата обращения 25.02.2024).
18. Raskin R., Pan M., Mattmann C. (2004). Enabling Semantic Interoperability for Earth Science Data [Электронный ресурс]. – Режим доступа: https://www.researchgate.net/publication/2950805_Enabling_Semantic_Interoperability_for_Earth_Science_Data (дата обращения 25.02.2024).
19. Global Change Master Directory (GCMD) Keywords [Электронный ресурс]. – Режим доступа: https://www.earthdata.nasa.gov/learn/find-data/idn/gcmd-keywords (дата обращения 25.02.2024).
20. ISO 19115 Geographic information – Metadata [Электронный ресурс]. – Режим доступа: https://www.iso.org/ru/standard/53798.html (дата обращения 25.02.2024).
21. Collins N., Theurich G., DeLuca C., Suarez M., Trayanov A., Balaji V., Li P., Yang W., Hill C., da Silva A. Design and Implementation of Components in the Earth System Modeling Framework // International Journal of High Performance Computing Applications. 2005. Vol. 19, № 3. P. 341-350. DOI: 10.1177/1094342005056120. https://zenodo.org/record/1235570.
22. FGDC [Электронный ресурс]. – Режим доступа: https://www.fgdc.gov/standards (дата обращения 25.02.2024).
23. OWL-S: Semantic Markup for Web Services [Электронный ресурс]. – Режим доступа: https://www.w3.org/Submission/OWL-S/ (дата обращения 25.02.2024).
24. Publishing and Using Earth Observation Data with the RDF Data Cube and the Discrete Global Grid System [Электронный ресурс]. – Режим доступа: https://w3c.github.io/sdw/eo-qb (дата обращения 25.02.2024).
25. Semantic Sensor Network Ontology [Электронный ресурс]. – Режим доступа: https://www.w3.org/TR/vocab-ssn/ (дата обращения 25.02.2024).
26. Wikipedia [Электронный ресурс]. – Режим доступа: https://www.wikipedia.org (дата обращения: 25.02.2023).

27. eBird [Электронный ресурс]. – Режим доступа: http://ebird.org (дата обращения: 25.02.2024).
28. OpenStreetMap [Электронный ресурс]. – Режим доступа: https://www.openstreetmap.org (дата обращения: 25.02.2024).
29. Плантариум [Электронный ресурс]. – Режим доступа: http://www.plantarium.ru (дата обращения: 25.02.2024).
30. Global Biodiversity Information Facility [Электронный ресурс]. – Режим доступа: http://data.gbif.org (дата обращения: 25.02.2024).
31. Species2000 [Электронный ресурс]. – Режим доступа: http://www.species2000.org (дата обращения: 25.02.2024).
32. Catalogue of Life [Электронный ресурс]. – Режим доступа: http://www.catalogueoflife.org (дата обращения: 25.02.2024).
33. The Plant List [Электронный ресурс]. – Режим доступа: http://www.theplantlist.org (дата обращения: 06.10.2016).
34. Tropicos [Электронный ресурс]. – Режим доступа: http://www.tropicos.org/Home.aspx (дата обращения: 25.02.2024).
35. Информационная поисковая система по фауне и флоре заповедников России [Электронный ресурс]. – Режим доступа: http://www.sevin.ru/natreserves (дата обращения: 25.02.2024).
36. ООПТ России [Электронный ресурс]. – Режим доступа: http://oopt.aari.ru (дата обращения: 25.02.2024).
37. Информационно-поисковые системы и БД Зоологического института РАН [Электронный ресурс]. – Режим доступа: http://www.zin.ru/proj_r.htm (дата обращения: 25.02.2024).
38. Биоразнообразие животного и растительного мира Сибири [Электронный ресурс]. – Режим доступа: http://www.nsc.ru/win/elbib/bio/#db (дата обращения: 25.02.2024).
39. Флора Байкальской Сибири [Электронный ресурс]. – Режим доступа: http://www.flora.baikal.ru (дата обращения: 25.02.2024).
40. Macklin J, Glöckler F, Hoffmann J, Ronquist F, Daume S, Haston E (2017) DINA: Open Source and Open Services – A Modern Approach for Sustainable Natural History

Collection Management Systems // Proceedings of TDWG. 2017. DOI: https://doi.org/10.3897/tdwgproceedings.1.20216 (дата обращения: 25.02.2024).

41. Метеосервис [Электронный ресурс]. – Режим доступа: meteoservice.ru (дата обращения: 25.02.2024).
42. OpenWeatherMap [Электронный ресурс]. – Режим доступа: openweathermap.org (дата обращения: 25.02.2024).
43. World Weather API and Weather Forecast [Электронный ресурс]. – Режим доступа: https://www.worldweatheronline.com/ (дата обращения: 25.02.2024).
44. AccuWeather [Электронный ресурс]. – Режим доступа: https://www.accuweather.com/ (дата обращения: 25.02.2024).
45. Rows [Электронный ресурс]. – Режим доступа: http://www.rows.com (дата обращения: 06.10.2022).
46. Google Таблицы [Электронный ресурс]. – Режим доступа: https://www.google.ru/intl/ru/sheets/about/ (дата обращения: 25.02.2024).
47. NextGIS [Электронный ресурс]. – Режим доступа: https://nextgis.ru/ (дата обращения: 25.02.2024).
48. ZOO-Project [Электронный ресурс]. – Режим доступа: http://www.zoo-project.org/ (дата обращения: 25.02.2024).
49. 52°North Web Geoprocessing Service [Электронный ресурс]. – Режим доступа: https://wiki.52north.org/Geoprocessing/52nWebProcessingService (дата обращения: 25.02.2024).
50. PyWPS [Электронный ресурс]. – Режим доступа: https://pywps.org/ (дата обращения: 25.02.2024).
51. ArcGIS Server [Электронный ресурс]. – Режим доступа: https://enterprise.arcgis.com/ru/server/latest/publish-services/windows/wps-services.htm. (дата обращения: 25.02.2024).
52. Geoserver [Электронный ресурс]. – Режим доступа: https://docs.geoserver.org/2.22.x/en/user/services/wps/operations.html. (дата обращения: 25.02.2024).
53. Xiaohui Q., Li Z., Ames D.P., Nelson E.J., Nathan S. Simplifying the deployment of OGC web processing services (WPS) for environmental modelling – Introducing

Tethys WPS Server // Environmental modelling & software. 2019. Vol. 115, № 2. P. 38-50.

54. GRASS [Электронный ресурс]. – Режим доступа: https://grass.osgeo.org/ (дата обращения: 25.02.2024).
55. Sextante [Электронный ресурс]. – Режим доступа: https://sextantegis.com/index.html (дата обращения: 25.02.2024).
56. Hinz M., Nüst D., Proß B., Pebesma E. Spatial Statistics on the Geospatial Web // Short paper, AGILE. 2013.
57. GDAL [Электронный ресурс]. – Режим доступа: https://gdal.org/ (дата обращения: 25.02.2024).
58. Garofalakis J., Panagis Y., Sakkopoulos E., Tsakalidis A. Contemporary Web Service Discovery Mechanisms // Journal of Web Engineering. 2006. Vol. 5, № 3. P. 265-90.
59. Nixon T. Web Services Dynamic Discovery (WS-Discovery), version 1.1 // OASIS. 2009. P. 1-50. [Электронный ресурс]. – Режим доступа: http://docs.oasis-open.org/ws-dd/discovery/1.1/os/wsdd-discovery-1.1-spec-os.pdf (дата обращения: 25.02.2024).
60. Edmond D., ter Hofstede A. Service composition for electronic commerce // Proceedings of the Pacific Asia Conference on Information Systems (PACIS-2000). Hong Kong, 2000.
61. Web Services Business Process Execution Language Version 2.0 (5 May 2003) [Электронный ресурс]. – Режим доступа: http://docs.oasis-open.org/wsbpel/2.0/OS/wsbpel-v2.0-OS.html (дата обращения: 25.02.2024).
62. Business Process Modeling Notation [Электронный ресурс]. – Режим доступа: http://www.omg.org/spec/BPMN/2.0.2/ (дата обращения: 25.02.2024).
63. Pereira J.L., Silva D. Business Process Modeling Languages: A Comparative Framework // Advances Intelligent Systems and Computing. 2016. Vol. 444. DOI: 10.1007/978-3-319-31232-3_58.
64. Huser V., Rasmussen L. V., Oberg R., Starren J. B. Implementation of workflow engine technology to deliver basic clinical decision support functionality // BMC medical research methodology. 2011. Vol. 11. P. 43. DOI:10.1186/1471-2288-11-43. — PMID 21477364.

65. Deelman E., Vahi K., Juve G. Pegasus, a workflow management system for science automation // Future Generation Computer Systems. 2015. Vol. 46. P. 17-35.
66. Ludäscher B., Altintas C., Berkley C., Higgins D., Jaeger E., Matthew J., Edward A.L., Tao J. Zhao Y. Scientific Workflow Management and the Kepler System // Special Issue: Workflow in Grid Systems. Concurrency and Computation: Practice & Experience. 2006. Vol. 18(10). P. 1039-1065.
67. Wilde M., Hategan, M., Wozniak J.M. Swift: A language for distributed parallel scripting // Parallel Computing. 2011. Vol. 37(9). P. 633-652.
68. Berthold M.R., Cebron N., Dill F. The konstanz information miner // SIGKDD Explorations. 2009. Vol. 11. P. 26-31.
69. Wolstencroft K., Haines R., Fellows D. The Taverna workflow suite: designing and executing workflows of Web Services on the desktop, web or in the cloud // Nucleic Acids Research. 2013. Vol. 41(W1). P. 557-561.
70. Blankenberg D., Kuster G.V., Coraor N. Galaxy: A Web-Based Genome Analysis Tool for Experimentalists // Current Protocols in Molecular Biology. 2010. № 89. P. 1-21.
71. Simmhan Y., Barga R., Ingen C. Building the trident scientific workflow workbench for data management in the cloud // Advanced Engineering Computing and Applications in Sciences (ADVCOMP). 2009. P. 41-50. DOI: 10.1109/ADVCOMP.2009.14.
72. Churches D., Gombas G., Harrison A. Programming scientific and distributed workflow with Triana services: Research articles // Concurrency and Computation: Practice and Experience. 2006. Vol. 18(10). P. 1021-1037.
73. Smirnov S., Sukhoroslov O., Volkov S. Integration and Combined Use of Distributed Computing Resources with Everest // Procedia Computer Science. 2016. Vol. 101. P. 359-368.
74. Бухановский А.В., Васильев В.Н., Виноградов В.Н., Смирнов Д.Ю., Сухоруков С.А., Яппаров Т.Г. CLAVIRE: Perspective Technology for Second Generation Cloud Computing // Приборостроение. Современные тенденции развития распределенных вычислений. 2011. Т. 54, № 10. С. 7-13.
75. Chen N.C., Di L.P., Yu G.N., Gong J.Y. Geo-processing workflow driven wildfire hot pixel detection under sensor web environment // Computers & geosciences. 2010. Vol. 36, № 3. P. 362-372.

76. Kwok Y.-K., Ahmad I. Static scheduling algorithms for allocating directed task graphs to multiprocessors // ACM Computing Surveys. 1999. Vol. 31, № 4. P. 406-471.
77. Xie G., Li R., Xiao X., Chen Y. A High-Performance DAG Task Scheduling Algorithm for Heterogeneous Networked Embedded Systems // Proceedings of IEEE 28th International Conference Advanced Information Networking and Applications. 2014. P. 1011-1016.
78. Zhi-Wei H., Cheng-Zhi Q., A-Xing Z., Peng L., Yi-Jie W., Yun-Qiang Z. From Manual to Intelligent: A Review of Input Data Preparation Methods for Geographic Modeling // ISPRS International journal of geo-information. 2019. Vol. 8, № 9. article 376. DOI: 10.3390/ijgi8090376.
79. Di L., Zhao P., Yang W., Yue P. Ontology-Driven Automatic Geospatial-Processing Modeling Based on Web-Service Chaining // Proceedings of the Sixth Annual NASA Earth Science Technology Conference. College Park, MD, USA. 2006. P. 27-29.
80. Zhao P., Di L., Yu G., Yue P., Wei Y., Yang W. Semantic Web-based geospatial knowledge transformation // Computers & Geosciences. 2009. Vol. 35, № 4. P. 798-808.
81. Scheider S., Ballatore A. Semantic typing of linked geoprocessing workflows // International Journal of Digital Earth. 2017. Vol. 11. P. 113-138.
82. Jiang J., Zhu A.X., Qin C.Z., Zhu T., Liu J., Du F., Liu J., Zhang G., An Y. CyberSoLIM: A cyber platform for digital soil mapping // Geoderma. 2016. № 263. P. 234–243.
83. Lutz M., Lucchi, R., Friis-Christensen A., Ostländer N. A Rule-Based Description Framework for the Composition of Geographic Information Services // Proceedings of the International Conference on GeoSpatial Sematics. Mexico City, Mexico, 29–30 November. 2007. P. 114-127.
84. Lutz M. Ontology-based descriptions for semantic discovery and composition of geoprocessing services // GeoInformatica. 2007. Vol. 11. P. 1-36.
85. Yue P., Di L., Yang W., Yu G., Zhao P., Gong J. Semantic Web Services-based process planning for earth science applications // International Journal of Geographical Information Science. 2009. Vol. 23. P. 1139-1163.

86. Farnaghi M., Mansourian A. Automatic composition of WSMO based geospatial semantic web services using artificial intelligence planning // Journal of Spatial Science. 2013. Vol. 58. P. 235-250.
87. Martin D., Burstein M., Hobbs J., Lassila O., McDermott D., McIlraith S., Narayanan S., Paolucci M., Parsia B., Payne T. OWL-S: Semantic markup for web services // W3C Member Submission. 2004.
88. Roman D., Keller U., Lausen H., Bruijn J.D., Stollberg M., Polleres A., Feier C., Bussler C., Fensel D. Web Service Modeling Ontology // Applied ontology. 2005. № 1. P. 77-106.
89. Li H., Zhu Q., Yang X., Xu L. Geo-information processing service composition for concurrent tasks: A QoS-aware game theory approach // Computers & Geosciences. 2012. Vol. 47. P.46-59.
90. Yue P., Tan Z., Zhang M. GeoQoS: Delivering Quality of Services on the Geoprocessing Web // Proceedings of the OSGeo's European Conference on Free and Open Source Software for Geospatial (FOSS4G-Europe 2014). Bremen, Germany. 15–17 July 2014.
91. Liping D., Peisheng Z., Wenli Y., Peng Y. Ontology-driven Automatic Geospatial-Processing Modeling based on Web-service Chaining. 2006. [Электронный ресурс]. – Режим доступа: https://www.researchgate.net/publication/237432891_Ontology-driven_Automatic_Geospatial-Processing_Modeling_based_on_Web-service_Chaining (дата обращения: 25.02.2024).
92. Fitzner D., Hoffmann J. and Klien E. Functional description of geoprocessing services as conjunctive datalog queries // Geoinformatica. 2011. Vol. 15(1). P. 191-221.
93. da Silva L.M., Braga R., Campos F. Composer-Science: A semantic service based framework for workflow composition // e-Science projects Information Sciences. 2012. Vol. 186. — № 1. P. 186-208
94. Xing H., Chen J., Wu H., Hou D. A web service-oriented geoprocessing system for supporting intelligent land cover change detection // International Journal of Geoinformatics. 2019. Vol. 8, № 50. P.1-19.
95. Fonte C.C., Minghini M., Patriarca J., Antoniou V., See L., Skopeliti A. Generating up-to-date and detailed land use and land cover maps using OpenStreetMap and

GlobeLand30 detection // International Journal of Geoinformatics. 2017. Vol. 6, № 125.

96. Baryannis G., Plexousakis D. Automated web service composition: State of the art and research challenges // Technical Report ICS-FORTH/TR-409. 2010. P. 82. [Электронный ресурс]. – Режим доступа: https://www.academia.edu/16602430/Automated_Web_Service_Composition_State_of_the_Art_and_Research_Challenges (дата обращения: 25.02.2024).
97. Blum A.L., Furst M.L. Fast planning through planning graph analysis // Artificial Intelligence. 1997. Vol. 90. P. 281-300.
98. Mena F.M., Ucan R.H., Cetina V.U., Ramirez F.M. Web service composition using the bidirectional Dijkstra algorithm // IEEE Transactions on Reliability. 2016. Vol. 14. P. 2522-2528.
99. Meyer H., Weske M. Automated service composition using heuristic search // Proceedings of the International Conference on Business Process Management, Vienna, Austria, 5-7 September. 2006. P. 81-96.
100. Yan Y., Chen M., Yang Y. Anytime QoS optimization over the PlanGraph for web service composition // Proceedings of the 27th Annual ACM Symposium on Applied Computing. Trento, Italy, 26–30 March. 2012. P. 1968-1975.
101. Brogi A., Corfini S. Behaviour-aware discovery of Web service compositions // International Journal of Web Services Research. 2007. Vol. 4. P. 1-25.
102. Brogi A., Corfini S., Montes J.F.A., Delgado I.N. A Prototype for Discovering Compositions of Semantic Web Services // Proceedings of the SWAP. Pisa, Italy. 2016. [Электронный ресурс]. – Режим доступа: https://www.yumpu.com/en/document/view/4779564/a-prototype-for-discovering-compositions-of-semantic-web-services (дата обращения: 25.02.2024).
103. Thakkar S., Knoblock C.A., Ambite J.L., Shahabi C. Dynamically composing web services from on-line sources // Proceedings of the AAAI-2002 Workshop on Intelligent Service Integration. Edmonton, AB, Canada, 29 July 2002. P. 1-7.
104. Yue P., Di L., Yang W., Yu G., Zhao P. Semantics-based automatic composition of geospatial Web service chains // Computers & Geosciences. 2007. Vol. 33. P. 649-665.

105. Cruz S.A., Monteiro A.M., Santos R. Automated geospatial web services composition based on geodata quality requirements // Computers & Geosciences. 2012. Vol. 47. P. 60-74.

106. Huang W., Harrie L. Towards knowledge-based geovisualisation using Semantic Web technologies: A knowledge representation approach coupling ontologies and rules // International Journal of Digital Earth. 2020. Vol.13. P. 976-997.

107. Sun Z., Yue P., Lu X., Zhai X., Hu L. A task ontology driven approach for live geoprocessing in a service-oriented environment // Transactions in GIS. 2012. Vol. 16. P. 867-884.

108. Zhuang C., Xie Z., Ma K., Guo M., Wu L. A task-oriented knowledge base for geospatial problem-solving // International journal of geoinformatics. 2018. Vol. 7, № 11. P. 867-884.

109. Li W., Song M., Tian Y. An ontology-driven cyberinfrastructure for intelligent spatiotemporal question answering and open knowledge discovery // International Journal of Geoinformatics. 2019. Vol. 8, № 11. P. 496.

110. Karakol U.D., Cömert Ç. Architecture for semantic web service composition in spatial data infrastructures // Survey Review. 2022. Vol. 54. P. 1-16.

111. Малых А.А., Манцивода А.В. Онтологии, метаданные и семантическое программирование // Вестник НГУ. Серия: Математика, механика, информатика. 2007. Vol. 7, № 2. P. 29-51.

112. Scheider S., Meerlo R., Kasalica V., Lamprecht A.-L. Ontology of core concept data types for answering geo-analytical questions // Journal of Spatial Information Science. 2020, № 20. P. 167-201.

113. Scheider S., Nyamsuren E., Kruiger H., Xu H. Geo-analytical question-answering with GIS // International Journal of Digital Earth. 2021. Vol. 14. P. 1-14.

114. Xing H., Liu C., Li R., Wang H., Zhang J., Wu H. Domain Constraints-Driven Automatic Service Composition for Online Land Cover Geoprocessing // International Journal of Geoinformatics. 2022. Vol. 11. 629. DOI: https://doi.org/10.3390/ijgi11120629.

115. Miao L., Liu C., Fan L., Kwan M.-P. An OGC web service geospatial data semantic similarity model for improving geospatial service discovery // Open Geosciences. 2021. Vol. 13. P. 245-261.

116. Wei Z., Gui Z., Zhang M., Yang Z., Mei Y., Wu H., Liu H., Yu J. Text GCN-SW-KNN: A novel collaborative training multi-label classification method for WMS application themes by considering geographic semantics // Big Earth Data. 2021. Vol. 5, № 1. P. 66-89. DOI: https://doi.org/10.1080/20964471.2021.1877434.

117. Cardoso J., Sheth A., Miller J., Arnold J., Kochut K. Quality of service for workflows and web service processes // Journal of Web Semantics. 2004. Vol. 1. P. 281-308.

118. Ньюкомер Э. Веб-сервисы. СПб.: Питер. 2003. 256 с.

119. Topcuoglu H., Hariri S., Wu M. Performance-effective and low-complexity task scheduling for heterogeneous computing // Parallel distributed systems. 2002. Vol. 13, № 3. P. 260-274.

120. Munir E., Mohsin S., Hussain A., Nisar M., Ali S. SDBATS: A Novel Algorithm for Task Scheduling in Heterogeneous Computing Systems // Proceedings of Parallel and Distributed Processing Symposium Workshops. 2013. P. 43-53.

121. Arabnejad H., Barbosa J.G. List scheduling algorithm for heterogeneous systems by an optimistic cost table // IEEE Transactions on Parallel and Distributed Systems. 2014. Vol. 25, № 3. P. 682-694.

122. Wang G., Guo H., Wang Y. A novel heterogeneous scheduling algorithm with improved task priority // Proceeding of IEEE 17th International Conference on High Performance Computing and Communications. 2015. P. 1826-1831.

123. Canon L.-C., Jeannot E., Sakellariou R., Zheng W. Comparative evaluation of the robustness of DAG scheduling heuristics // Grid Computing. 2008. P. 73-84.

124. Gupta S., Kumar V., Agarwal G. Task Scheduling in Multiprocessor System Using Genetic Algorithm // Proceedings of Second International Conference on Machine Learning and Computing. 2010. P. 267-271.

125. Dorigo M., Stützle T. Ant Colony Optimization // Intelligence Magazine IEEE. 2004. P. 319.

126. Bertsimas D., Tsitsiklis J. Simulated Annealing // Statistical Science. 1993. Vol. 8, № 1. P. 10-15.

127. Kennedy J., Eberhart R. Particle swarm optimization // Proceedings of IEEE International Conference. 1995. Vol. 4. P. 1942-1948.

128. Claessen K., Een N., Sheeran M., Sorensson N., Voronov A., Akesson K. SAT-solving in practice // Proceeding of 9th International Workshop on Discrete Event Systems. 2008. P. 61-67.

129. Nasonov D., Butakov N., Balakhontseva M., Knyazkov K., Boukhanovsky A. Hybrid evolutionary workflow scheduling algorithm for dynamic heterogeneous distributed computational environment // Journal of Applied Logic. 2017. Vol. 24. P. 50–61.

130. Dean J., Ghemawat S. MapReduce: Simplified Data Processing on Large Clusters // Sixth Symposium on Operating System Design and Implementation. 2004. P. 137-149.

131. Hadoop. Режим доступа: http://hadoop.apache.org/ (дата обращения 18.04.2023).

132. Spatialhadoop. Режим доступа: http://spatialhadoop.cs.umn.edu/ (дата обращения 18.04.2012).

133. Abouzeid A., Bajda-Pawlikowski K., Abadi D., Silberschatz A., Rasin A. HadoopDB: An Architectural Hybrid of MapReduce and DBMS Technologies for Analytical Workloads // Proceedings of the 35th VLDB Conference. 2009.

134. Созыкин А.В., Гольдштейн М.Л. Система обработки изображений с автоматическим распараллеливанием на основе MapReduce // Вестник Южно-Уральского государственного университета. 2012. № 27 (286). С. 109-118.

135. Aji A., Wang F., Vo H., Lee R., Liu Q., Zhang X., Saltz J. Hadoop: GIS: A High Performance Spatial Data Warehousing System over MapReduce // The 39th International Conference on Very Large Data Bases. 2013. Vol. 6, № 11. P. 1009-1020.

136. Lecue F., Leger A. A formal model for semantic Web service composition // The Semantic Web – ISWC. 2006. Vol. 4273. P. 385-398.

137. Kopecky, J., Vitvar T., Bournez C., Farrell J. SAWSDL: Semantic annotations for WSDL and XML schema // IEEE INTERNET COMPUTING. 2007. Vol. 11, № 6. P. 60-67.

138. Федоров Р.К., Шумилов А.С., Бычков И.В., Ружников Г.М. Компоненты среды WPS-сервисов обработки геоданных // Вестник Новосибирского гос. ун-та. Сер. Информационные технологии. 2014. Т. 12, № 3. С. 16-24.

139. Фёдоров Р.К., Бычков И.В., Ружников Г.М. Формирование композиций сервисов на основе статистических данных пользователей // Вестник

Новосибирского гос. ун-та. Сер. Информ. технологии. 2021. Т. 19, № 2. С. 115-130. DOI: 10.25205/1818-7900-2021-19-2-115-130.

140. Fedorov R.K., Shumilov A.S., Voskoboynikov M.L. Analysis of service calls for construction of the semantic network of services // Proc. 1st Scientific-Practical Workshop on Information Technologies: Algorithms, Models, Systems (ITAMS-2018). 2018. Vol. 2221. P. 20-24. URL: http://ceur-ws.org/Vol-2221/paper4.pdf (Scopus).
141. Fedorov R. Building service composition based on statistics of the services use // CEUR Workshop Proceedings: Proc. of 2nd Scientific-Practical Workshop Information Technologies: Algorithms, Models, Systems (ITAMS-2019). 2019. Vol. 2463. P. 40-46. (Scopus).
142. Feoktistov A., Gorsky S., Kostromin R., Fedorov R., Bychkov I. Integration of web processing services with workflow-based scientific applications for solving environmental monitoring problems // ISPRS International Journal of Geo-Information. 2022. Vol. 11, № 1 DOI: 10.3390/ijgi11010008 (Web of Science Q2).
143. Bychkov I.V., Feoktistov A.G., Gorsky S.A., Kostromin R.O., Fedorov R.K. Automating the Integration of Services for the Web Processing of Environmental Monitoring Data with Distributed Scientific Applications // Optoelectronics, Instrumentation and Data Processing. 2022. Vol. 58, №4. P. 373–379. DOI: 10.3103/S8756699022040045 (Scopus Q3)
144. Воскобойников М.Л., Федоров Р.К., Ружников Г.М. Автоматизация вызовов WEB-сервисов на мобильном устройстве // Вестник Бурятского гос. ун-та. Математика, информатика. 2019. № 2. С. 83-94. DOI: 10.18101/2304-5728-2019-2-83-94.
145. Попова А.К., Ружников Г.М., Бычков И.В., Хмельнов А.Е., Шигаров А.О., Гаченко А.С., Федоров Р.К., Фереферов Е.С., Новицкий Ю.А. Интеграция информационно-аналитических ресурсов и обработка пространственных данных в задачах управления территориальным развитием: моногр. Новосибирск. 2011. 1 С.
146. Бешенцев А.Н., Гаченко А.С., Батуев А.Р., Плюснин В.М., Федоров Р.К., Хмельнов А.Е., Бычков И.В., Ружников Г.М., Сороковой А.А., Воронин В.И. Сервисы и инфраструктура пространственных данных междисциплинарныхнаучных исследований геосистем и биоразнообразия Прибайкалья и Забайкалья: моногр. Барнаул. 2011. 11 С.
147. Маджара Т.И., Бычков И.В., Ружников Г.М., Хмельнов А.Е., Федоров Р.К., Парамонов В.В., Шигаров А.О., Фереферов Е.С., Гаченко А.С., Михайлов А.А.,

Шумилов А.С., Авраменко Ю.В. Инфраструктура информационных ресурсов и технологии создания информационно-аналитических систем территориального управления: моногр. 2016.

148. Бычков И.В., Ружников Г.М. и др. Фундаментальные основы, методы и технологии цифрового мониторинга и прогнозирования экологической обстановки Байкальской природной территории: моногр. Новосибирск. 2022.
149. Бычков И.В., Ружников Г.М., Хмельнов А.Е., Федоров Р.К., Маджара Т.И., Шигаров А.О., Дорж Т., Нергуй Б. Технологические основы развития инфраструктуры пространственных данных Монгольской академии наук // Вычислительные технологии. 2013. Т. 18, № 5. С. 16–26.
150. Федоров Р.К., Шумилов А.С., Бычков И.В., Ружников Г.М. Компоненты среды WPS-сервисов обработки геоданных // Вестник Новосибирского гос. ун-та. Сер. Информационные технологии. 2014. Т. 12, № 3. С. 16-24.
151. Бычков И.В., Ружников Г.М., Парамонов В.В., Шумилов А.С., Фёдоров Р.К. Инфраструктурный подход к обработке пространственных данных в задачах управления территориальным развитием // Вычислительные технологии. 2018. Т. 23, № 4. С. 15-31. DOI: 10.25743/ICT.2018.23.16488.
152. Бычков И.В., Федоров Р.К., Фереферов Е.С. Инструментальные компоненты цифровой платформы экологического мониторинга Байкальской природной территории // Вычислительные технологии. 2023. Т. 28. № 6. С. 95-107
153. Бычков И.В., Маджара Т.И., Новопашин А.П., Фереферов Е.С., Феоктистов А.Г., Федоров Р.К. Информационно-вычислительные ресурсы ИРНОК: инфраструктура, данные, приложения // Вычислительные технологии. 2023. Т. 28. № 3. С. 117-135
154. Fedorov R.K., Shumilov A.S., Ruzhnikov G.M. Geoportal cloud // CEUR Workshop Proceedings. 2017. Vol. 2033. P. 305-308 (Scopus).
155. Федоров Р.К., Шумилов А.С. Создание и публикация WPS-сервисов на основе облачной инфраструктуры // Вестник Бурятского гос. ун-та. 2015. № 4. С. 29-35.
156. Хмельнов А.Е., Гаченко А.С., Фереферов Е.С., Ружников Г.М., Федоров Р.К. Интеграционный подход создания региональной инфраструктуры пространственных данных // Вестник Бурятского гос. ун-та. 2015. № 3. С. 39-48.
157. Федоров Р.К., Федорова Е.Н., Ружников Г.М. Технология сбора и анализа реляционных данных // Открытое образование. 2016. Т. 20, № 5. С. 35-40.
158. Бычков И.В., Плюснин В.М., Ружников Г.М., Хмельнов А.Е., Федоров Р.К., Гаченко А.С. Создание инфраструктуры пространственных данных для управления регионом // География и природные ресурсы. 2013. № 2. С. 146–151.

159. Шумилов А.С., Фёдоров Р.К., Ружников Г.М., Ветров А.А., Михайлов А.А. Интернет-система ввода и редактирования пространственных данных «Фарамант»: Свидетельство о государственной регистрации программ для ЭВМ № 2014610274 от 09.01.2014. М.: Федеральная служба по интеллектуальной собственности, патентам и товарным знакам, 2014.

160. Фёдоров Р.К. Программная система выполнения WPS сервисов: Свидетельство о государственной регистрации программ для ЭВМ № 2024611171 от 20.12.2023. М.: Федеральная служба по интеллектуальной собственности, патентам и товарным знакам, 2023.

161. Федоров Р.К., Бычков И.В., Шумилов А.С., Ружников Г.М. Система планирования и выполнения композиций веб-сервисов в гетерогенной динамической среде // Вычислительные технологии. 2016. Т. 21, № 6. С. 18-35.

162. Бычков И.В., Ружников Г.М., Фёдоров Р.К., Шумилов А.С. Выполнение JAVASCRIPT-композиций WPS-сервисов в распределенной гетерогенной среде // Вычислительные технологии. 2019. Т. 24, № 3. С. 44-58. DOI: 10.25743/ICT.2019.24.3.004.

163. Бычков И.В., Федоров Р.К., Шумилов А.С., Ружников Г.М. Система планирования и выполнения композиций веб-сервисов в гетерогенной динамической среде: Свидетельство о государственной регистрации программ для ЭВМ № 2016663724 от 18.10.2016. М.: Федеральная служба по интеллектуальной собственности, патентам и товарным знакам, 2016.

164. Федоров Р.К., Бычков И.В., Шумилов А.С., Ружников Г.М. Среда выполнения сервисов и их сценариев: Свидетельство о государственной регистрации программ для ЭВМ № 2017617913 от 17.07.2017. М.: Федеральная служба по интеллектуальной собственности, патентам и товарным знакам, 2017.

165. Фёдоров Р.К. Программная система выполнения WPS сервисов: Свидетельство о государственной регистрации программ для ЭВМ № 2024611171 от 20.12.2023. М.: Федеральная служба по интеллектуальной собственности, патентам и товарным знакам, 2023.

166. Федоров Р.К., Шумилов А.С., Авраменко Ю.В. Обработка векторных данных с помощью спецификаций в соответствии с моделью MapReduce // Вестник Бурятского гос. ун-та. Математика, информатика. 2017. №2. С. 12-19. DOI: 10.18101/2304-5728-2017-2-12-19.

167. Шумилов А.С., Федоров Р.К. Задание графа зависимостей для композиций сервисов с помощью JavaScript сценариев // География и природные ресурсы. 2016. № 6. С. 160-163.

168. Бычков И.В., Ружников Г.М., Федоров Р.К., Попова А.К., Будээбазар А., Балт Б., Уганбаатар Д. Цифровая трансформация экологического мониторинга оз. Хубсугул и Прихубсугулья // Вычислительные технологии. 2022. Т. 27. № 5. С. 14-29

169. Bychkov I.V., Plyusnin V.M., Ruzhnikov G.M., Fedorov R.K., Khmel'nov A.E., Gachenko A.S. The creation of a spatial data infrastructure in management of regions (exemplified by Irkutsk oblast) // Geography and Natural Resources. 2013. Vol. 34, № 2. P. 191–195. DOI: 10.1134/S1875372813020133 (Scopus).

170. Paramonov V., Fedorov R., Ruzhnikov G., Shumilov A. Web-Based Analytical Information System for Spatial Data Processing // Communications in Computer and Information Science. 2013. Vol. 403. P. 93-101 (Web of Science Emerging Sources Citation Index).

171. Bychkov I.V., Ruzhnikov G.M., Paramonov V.V., Shumilov A.S., Fedorov R.K., Sanjaa B. Infrastructural approach to spatial data processing in applications to territorial development management // CEUR Workshop Proceedings. 2017. Vol. 2033. P. 7-9 (Scopus).

172. Bychkov I.V., Fedorov R.K., Avramenko Y.V., Shumilov A.S., Shigarov A.O., Ruzhnikov G.M. et al. Information-analytical environment supporting interdisciplinary research of natural resources in the Baikal region // CEUR Workshop Proceedings, Proc. 1st Scientific-Practical Workshop on Information Technologies: Algorithms, Models, Systems (ITAMS-2018). 2018. Vol. 2221. P. 42-52. URL: http://ceur-ws.org/Vol-2221/paper8.pdf (Scopus).

173. Fedorov R.K., Shumilov A.S. Service compositions in problems of urban planning // Proc. 1st Scientific-Practical Workshop on Information Technologies: Algorithms, Models, Systems (ITAMS-2018). 2018. Vol. 2221. P. 1-6. URL: http://ceur-ws.org/Vol-2221/paper1.pdf (Scopus).

174. Bychkov I.V., Ruzhnikov G.M., Paramonov V.V., Shumilov A.S., Fedorov R.K., Levi K.G., Demberel S. Infrastructural approach and geospatial data processing services in the tasks of territorial development management // Proceedings 1st Intern. Geographical Conference of North Asian Countries on China-Mongolia-Russia Economic Corridor: Geographical and Environmental Factors and Territorial Development Opportunities. 2018. Vol. 190, № 1. DOI: 10.1088/1755-1315/190/1/012048 (Web of Science Emerging Sources Citation Index).

175. Fereferov E.S., Gachenko A.S., Hmelnov A.E., Fedorov R.K. Information Technologies for Monitoring of Anthropogenic Impacts to Lake Baikal // Proceedings for First Scientific-practical Workshop Information Technologies: Algorithms, Models, Systems (ITAMS-2018). 2018. Vol. 2221. P. 61-69. URL: http://ceur-ws.org/Vol-2221/paper10.pdf (Scopus).

176. Avramenko Y.V., Fedorov R.K. The technology of classification geospatial data based on WPS standard // Proceedings 1st Scientific-Practical Workshop on Information Technologies: Algorithms, Models, Systems (ITAMS-2018). 2018. Vol. 2221. P. 37-41. URL: http://ceur-ws.org/Vol-2221/paper7.pdf (Scopus).

177. Bychkov I.V., Gachenko A.S., Hmelnov A.E., Fedorov R.K., Fereferov E.S. Geological Information System of environmental and human intervention impact assessment on bodies of water of the Irkutsk region // Proceedings 1st Intern. Geographical Conference of North Asian Countries on China-Mongolia-Russia Economic Corridor: Geographical and Environmental Factors and Territorial Development Opportunities. 2018. Vol. 190, № 1. DOI: 10.1088/1755-1315/190/1/012027 (Web of Science Emerging Sources Citation Index).

178. Avramenko Yu., Fedorov R. The method of extracting the contours of objects from satellite images // CEUR Workshop Proceedings: Proceedings of 2nd Scientific-Practical Workshop Information Technologies: Algorithms, Models, Systems (ITAMS-2019). 2019. Vol. 2463. P. 70-75 (Scopus).

179. Bychkov I.V., Ruzhnikov G.M., Khmelnov A.E., Fedorov R.K., Madzhara T.I. Digital monitoring of the ecosystem of Lake Baikal // CEUR Workshop Proceedings. Volume: All-Russian Conference «Spatial Data Processing for Monitoring of Natural and Anthropogenic Processes» (SDM 2019). 2019. Vol. 2534. P. 8-14 (Scopus).

180. Bychkov I.V., Ruzhnikov G.M., Khmelnov A.E., Fedorov R.K., Madzhara T.I., Popova A.K. Digital monitoring of Lake Baikal and its coastal area // CEUR Workshop Proceedings: Proceedings of 2nd Scientific-Practical Workshop Information Technologies: Algorithms, Models, Systems (ITAMS-2019). 2019. Vol. 2463. P. 13-23. DOI: 2-s2.0-85073518338 (Scopus).

181. Bychkov I., Ruzhnikov G., Paramonov V., Mikhailov A., Fedorov R., Klyuchevskii A., Dem'yanovich V., Demberel S. The Framework of the Digital Environment for Analysing of Seismic Hazards of Lithosphere Blocks in Baikal-Mongolian Region // CEUR Workshop Proceedings: Proceedings of 2nd Scientific-Practical Workshop Information Technologies: Algorithms, Models, Systems (ITAMS-2019). 2019. Vol. 2463. P. 84-92 (Scopus).

182. Batuev A.R., Batuev D.A., Beshentsev A.N., Bogdanov V.N., Dashpilov T.B., Korytniy L.M., Tikunov V.S., Fedorov R.K. Atlas information system for providing socio-economic development of the Baikal region // InterCarto, InterGIS. 2019. Vol. 25. P. 66-80. DOI: 10.35595/2414-9179-2019-1-25-66-80 (Scopus).

183. Fedorov R.K., Kitov A.D., Avramenko Y.V. Automation of forming a database of the glaciers based on remote sensing // CEUR Workshop Proceedings: All-Russian Conf. «Spatial Data Processing for Monitoring of Natural and Anthropogenic Processes» (SDM 2019). 2019. Vol. 2534. P. 207-211 (Scopus).

184. Ruzhnikov G.M., Klyuchevskii A.V., Paramonov V.V., Mikhailov A.A., Fedorov R.K., Dem'yanovich V.M., Dembrel S. Service-oriented Information and Analytical the System of Estimation of Influence of the Lithosphere Model on Dynamic Parameters of Rocky Soil Oscillations From Earthquakes of Southern Baikal Region // CEUR Workshop Proceedings: Proceedings Information Technologies in Earth Sciences and Applications for Geology, Mining and Economy (ITES&MP-2019). 2019. Vol. 2527. P. 42-46 (Scopus).

185. Bychkov I., Feoktistov A., Gorsky S., Edelev A., Sidorov I., Kostromin R., Fereferov E., Fedorov R. Supercomputer Engineering for Supporting Decision-making on Energy Systems Resilience // Proceedings of the 14th IEEE International Conference on Application of Information and Communication Technologies (IEEE, 2020). 2020. P. 1-6 (Scopus).

186. Bychkov I.V., Ruzhnikov G.M., Fedorov R.K., Khmelnov A.E., Popova A.K. Digital environmental monitoring technology Baikal natural territory // CEUR Workshop Proceedings: 3rd Scientific-Practical Workshop Information Technologies: Algorithms, Models, Systems (ITAMS-2020). 2020. Vol. 2677 (Scopus).

187. Avramenko Y.V., Fedorov R.K. Applied digital platform for remote sensing data processing // CEUR Workshop Proceedings: 3rd Scientific-Practical Workshop Information Technologies: Algorithms, Models, Systems (ITAMS-2020). 2020. Vol. 2677 (Scopus).

188. Bychkov I.V., Ruzhnikov G.M., Fedorov R.K., Popova A.K. Digital platform for forest resources monitoring in the BAIKAL natural territory // Journal of Physics: Conference Series: 13th Multiconference on Control Problems (MCCP 2020). 2021. Vol. 1864, № 1. DOI: 10.1088/1742-6596/1864/1/012111 (Scopus).

189. Bychkov I.V., Ruzhnikov G.M., Fedorov R.K., Popova A.K. A platform approach to the organization of digital forest monitoring of the Baikal natural territory // IOP Conference Series: Earth and Environmental Science: 11th Intern. Conf. and Early Career Scientists School on Environmental Observations, Modeling and Information

Systems (ENVIROMIS 2020). 2021. Vol. 611. P. 012056. DOI: 10.1088/1755-1315/611/1/012056 (Scopus).

190. Bychkov I.V., Ruzhnikov G.M., Fedorov R.K., Khmelnov A.E., Popova A.K. Organization of digital monitoring of the Baikal natural territory // IOP Conference Series: Earth and Environmental Science. 2021. DOI: 10.1088/1755-1315/629/1/012067 (Scopus).
191. Bychkov I.V., Ruzhnikov G.M., Fedorov R.K., Popova A.K. Digital technologies for forest monitoring in the Baikal natural territory // CEUR Workshop Proceedings: 4th Scientific-Practical Workshop Information Technologies: Algorithms, Models, Systems (ITAMS-2021). 2021. Vol. 2984. P. 1-5 (Scopus).
192. Avramenko Y.V., Popova A.K., Fedorov R.K. Cloud service of Geoportal ISDCT SB RAS for machine learning // CEUR Workshop Proceedings: 4th Scientific-Practical Workshop Information Technologies: Algorithms, Models, Systems (ITAMS-2021). 2021. Vol. 2984. P. 6-10 (Scopus).
193. Bychkov I.V., Paramonov V.V., Ruzhnikov G.M., Mikhailov A.A., Fedorov R.K., Klyuchevskii A.V., Dem'yanovich V.M., Demberel S. Russian-Mongolian scientific initiative for assessing the seismic hazards of the Baikal region and Mongolia // CEUR Workshop Proceedings: 4th Scientific-Practical Workshop Information Technologies: Algorithms, Models, Systems (ITAMS-2021). 2021. Vol. 2984. P. 112-119 (Scopus).
194. Zubova E., Kashulin N., Terentyev P., Melekhin A., Fedorov R.K., Shalygin S. Occurrence of fish species in the inland water of Murmansk Region (Russia): research in 1972-2021 // Biodiversity Data Journal. 2021. Vol. 9. DOI: 10.3897/BDJ.9.e68131 (Web of Science Q3).
195. Bychkov I.V., Ruzhnikov G.M., Fedorov R.K., Popova A.K., Avramenko Y.V. Classification of Sentinel-2 satellite images of the Baikal Natural Territory // Computer Optics. 2022. Vol. 46, № 1. P. 90-96. DOI: 10.18287/2412-6179-CO-1022. URL: http://www.computeroptics.ru/KO/PDF/KO46-1/460111.pdf (Web of Science Q2).
196. Федоров Р.К., Парамонов В.В., Ружников Г.М., Данчинова Г.А., Хасантинов М.А., Ляпунов А.В. WEB-сервис импорта данных из реляционных таблиц в CSV формате: Свидетельство о государственной регистрации программ для ЭВМ № 2017615230 от 05.05.2017. М.: Федеральная служба по интеллектуальной собственности, патентам и товарным знакам, 2017.
197. Данчинова Г.А., Ляпунов А.В., Хаснатинов М.А., Болотова Н.А., Манзарова Э.Л., Соловаров И.С., Петрова И.В., Шулепова С.Ю., Лазарева Е.Л., Федоров Р.К., Парамонов В.В. Информационно-справочная система «Обращаемость людей, пострадавших от присасывания иксодовых клещей на Территории

Бурятии» (ИСС «Бурятия-клещи»): Свидетельство о государственной регистрации программ для ЭВМ № 2017620505 от 04.05.2017. М.: Федеральная служба по интеллектуальной собственности, патентам и товарным знакам, 2017.

198. Федоров Р.К., Авраменко Ю.В. Программа расчета спектральных индексов и построения композитного отображения для космоснимков Sentinel-2: Свидетельство о государственной регистрации программ для ЭВМ № 2022615805 от 04.04.2022. М.: Федеральная служба по интеллектуальной собственности, патентам и товарным знакам, 2022.
199. Фёдоров Р.К., Авраменко Ю.В. WPS сервис классификации космоснимков Sentinel-2: Свидетельство о государственной регистрации программ для ЭВМ № 2024610919 от 21.12.2023. М.: Федеральная служба по интеллектуальной собственности, патентам и товарным знакам, 2023.
200. Бычков И.В., Горский С.А., Еделев А.В., Костромин Р.О., Сидоров И.А., Феоктистов А.Г., Фереферов Е.С., Федоров Р.К. Поддержка управления живучестью систем энергетики на основе комбинаторного подхода // Известия Российской академии наук. Теория и системы управления. 2021. Т. 6, № 6. С. 122-135. DOI: 10.31857/S000233882106007X.
201. Федоров Р.К., Шумилов А.С. Сценарий расчета временной доступности объектов образования // Вестник Бурятского гос. ун-та. Математика, информатика. 2017. №2. С. 20-32. DOI: 10.18101/2304-5728-2017-2-20-32.
202. Антонов И.А., Федоров Р.К., Башалханов И.А. Анализ пространственного распределения поселений рыжих лесных муравьев в Байкальском регионе // Журнал Сибирского федерального ун-та. Сер. Биология. 2019. Т. 12, № 4. С. 385-397. DOI: 10.17516/1997-1389-0309.
203. Антонов И.А., Федоров Р.К., Гаченко А.С., Агафонова Т.А. Интеграция базы данных по хвоегрызущим насекомым Байкальской Сибири в среду геопортала // Известия Иркутского гос. ун-та. Сер. Биология. Экология. 2013. Т. 6, № 2. С. 159–162.
204. Верхозина А.В., Кривенко Д.А., Мурашко В.В., Федоров Р.К., Казановский С.Г., Шумилов А.С. Информационно-аналитическая система по фито-разнообразию Байкальской Сибири // Известия Иркутского гос. ун-та. Сер. Биология. Экология. 2016. Т. 9, № 3. С. 11-28.
205. Парамонов В.В., Михайлов А.А., Ружников Г.М., Фёдоров Р.К., Ключевский А.В., Демьянович В.М. Информационно-аналитическая система оценки сейсмического потенциала кайнозойских активных разломов Монголо-

байкальского региона // Вестник Бурятского гос. ун-та. Математика, информатика. 2020. № 4. С. 26-39. DOI: 10.18101/2304-5728-2020-4-26-39.

206. Фереферов Е.С., Минаев В.В., Михайлов А.А., Гаченко А.С., Власова Н.В., Воробьева Н.В., Федоров Р.К., Хмельнов А.Е. Информационно-аналитическая система мониторинга и оценки антропогенного воздействия на экологию прибрежной зоны озера Байкал // География и природные ресурсы. 2016. № 6. С. 174-178.

207. Фёдоров Р.К., Авраменко Ю.В. WPS-сервисы обработки данных дистанционного зондирования Земли // Вестник Бурятского гос. ун-та. 2014. № 9(1). С. 12-15.

208. Bychkov I.V., Gachenko A.S., Hmelnov A.E., Fedorov R.K., Fereferov E.S. Geological Information System of environmental and human intervention impact assessment on bodies of water of the Irkutsk region // Proceedings 1st Intern. Geographical Conference of North Asian Countries on China-Mongolia-Russia Economic Corridor: Geographical and Environmental Factors and Territorial Development Opportunities. 2018. Vol. 190. № 1 DOI: 10.1088/1755-1315/190/1/012027 (Web of Science Emerging Sources Citation Index).

209. Zubova E., Kashulin N., Terentyev P., Melekhin A., Fedorov R.K., Shalygin S. Occurrence of fish species in the inland water of Murmansk Region (Russia): research in 1972-2021 // Biodiversity Data Journal. 2021. Vol. 9. DOI: 10.3897/BDJ.9.e68131 (Web of Science Q3).

ПРИЛОЖЕНИЕ А. Основные обозначения и сокращения

AaaS - Application as a Service – приложение как услуга

API - Application Programming Interface – программный интерфейс приложения

AWF – Abstract Workflow – абстрактный WF

CWF – Concrete Workflow – конкретный WF

DAG –Directed Acyclic Graph – направленный ациклический граф

OWL – Web Ontology Language – язык описания онтологий

Provenance – журналирование процесса исполнения

RDF – Resource Description Framework – язык описания ресурсов

REST - REpresentational State Transfer – «передача состояния представления»

RO – Research Object – объект исследования

SaaS – Software as a Service – программное обеспечение как услуга

SOA – Service-Oriented Architecture – Сервисно-Ориентированная Архитектура

UML – Unified Modeling Language – универсальный язык моделирования

VSO – Virtual Simulation Objects – виртуальный моделирующий объект

WCF – windows communication foundation – технология обмена данными

WF - workflow – поток задач

WFMS – Workflow Management System – система управления исполнением КП

XML — Extensible Markup Language – расширяемый язык разметки

БЗ – База знаний

ИП – Интеллектуальная Поддержка

КП – Композитное Приложение

МОА – Модельно-Ориентированная Архитектура

ООП – Объекто-Ориентированное Программирование

ОТ – Общественный Транспорт

ПО – Программное Обеспечение

ПРИЛОЖЕНИЕ Б. СИНТАКСИС ВЫРАЖЕНИЙ ДЛЯ ВЫЧИСЛЕНИЯ ЗНАЧЕНИЙ АТРИБУТОВ СЕРВИСОВ ВВОДА И РЕДАКТИРОВАНИЯ ДАННЫХ

```
/* description: Parses end executes mathematical expressions. */

/* lexical grammar */
%lex
%%

\s+                   /* skip whitespace */
[0-9]+("."[0-9]+)?\b  return 'NUMBER'
\'([+-/|/%*$\^!" <>=.a-zA-Z0-
9_{}&\?\[\]]*)\'           yytext = yytext.slice(1,-1);     return 'STRING'
"*"                    return '*'
"/"                    return '/'
","                    return ','
"-"                    return '-'
"+"                    return '+'
"^"                    return '^'
"("                    return '('
")"                    return ')'
"PI"                   return 'PI'
"true"                 return 'true'
"false"                return 'false'
"E"                    return 'E'
"jspath"               return 'jspath'
"getrefval"            return 'getrefval'
"loadrest"             return 'loadrest'
"img_attr"             return 'img_attr'
<<EOF>>                return 'EOF'
.                      return 'INVALID'

/lex

/* operator associations and precedence */

%left '+' '-'
%left '*' '/'
%left '^'
%left UMINUS

%start expressions

%% /* language grammar */

expressions
    : e EOF
        {return $1;}
```

```
        ;

    e
        : e '+' e
            {$$ = $1+$3;}
        | e '-' e
            {$$ = $1-$3;}
        | e '*' e
            {$$ = $1*$3;}
        | e '/' e
            {$$ = $1/$3;}
        | e '^' e
            {$$ = Math.pow($1, $3);}
        | '-' e %prec UMINUS
            {$$ = -$2;}
        | '(' e ')'
            {$$ = $2;}
        | 'jspath' '(' STRING ')'
            {
                if (typeof exports === 'undefined' )
                    $$ = fieldcalc.getdocvalue($3);
                else
                    $$ = global.fieldcalc.getdocvalue($3);
            }
        | 'loadrest' '(' e ')'
            {
                if (typeof exports === 'undefined' )
                    $$ = fieldcalc.loadrest($3);
                else
                    $$ = global.fieldcalc.loadrest($3);
            }
        | 'img_attr' '(' e ')'
            {
                if (typeof exports === 'undefined' )
                    $$ = img_attr($3);
                else
                    $$ = $3;
            }
        | 'getrefval' '(' e ',' e ',' e ')'
            {
                if (typeof exports === 'undefined')
                    $$ = fieldcalc.getrefval($3,$5,$7);
                else
                    $$ = global.fieldcalc.getrefval($3,$5,$7);
            }
        | NUMBER
            {$$ = Number(yytext);}
        | 'false'
            {$$ = false;}
```

```
        | 'true'
            {$$ = true;}
        | STRING
            {$$ = yytext;}
        | E
            {$$ = Math.E;}
        | PI
            {$$ = Math.PI;}
        ;
```

ПРИЛОЖЕНИЕ В. ГЕОПОРТАЛ ИНФОРМАЦИОННО-АНАЛИТИЧЕСКАЯ СИСТЕМА ПО ФИТОРАЗНООБРАЗИЮ БАЙКАЛЬСКОЙ СИБИРИ

С 2012 года работает и постоянно развивается геопортал «Информационно-аналитическая система по фиторазнообразию Байкальской Сибири» для проведения инвентаризации данных гербарных коллекций и аналитической поддержки ботанических исследований. Геопортал (рисунок В.1) находится по адресу http://biodiv.isc.irk.ru/. Разрабатывается совместно с отделом СИФИБР СО РАН «Биоразнообразие и природные ресурсы. ИАС предназначена для интеграции, хранения, обработки, комплексного использования и анализа данных по биоразнообразию. Проблема, которую помогает решать геопортал, заключается в том, что в гербарных хранилищах России находится более 17 млн. гербарных листов в основном в бумажном виде. При этом нередко отсутствуют каталоги коллекций, как правило, их роль выполняют бумажные картотеки или настольные электронные базы данных (БД). Крайне редко БД коллекций доступны в сети Интернет, еще реже к учетным записям в них привязаны фотографии или сканы образцов. В связи с этим анализ данных биоразнообразия является трудоемким и затратным по времени. При разработке геопортала ставилась задача перевода данных гербарных коллекций в цифровой вид и публикация их в Интернет. Для перевода в цифровой вид использовались сервисы ввода и редактирования реляционных данных. Выделена виртуальная машина на базе Ubuntu 18 TLS. Развернут типовой геопортал. Система призвана обеспечить интеграцию разных по составу данных для проведения широкомасштабного анализа данных о биологическом разнообразии, эффективный информационный обмен между организациями, учеными, сервисами, отображение информации в виде таблиц, карт, пространственный и статистический анализ данных.

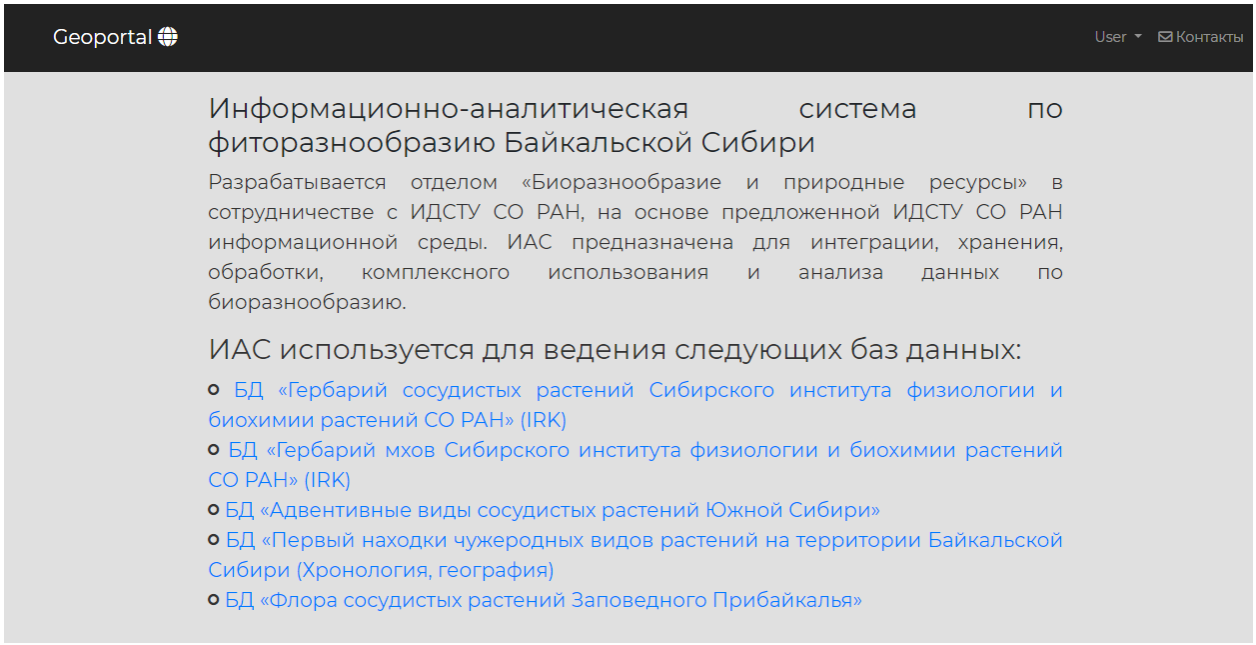


Рисунок В.1 – Главная страница Информационно-аналитической системы по фиторазнообразию Байкальской Сибири

Структура данных для хранения гербариев определена этикетками гербарных листов и состоит из следующих полей: название вида, местонахождение, местообитание, даты сбора, определения, коллектор, автор определения, гербарная коллекция (в БД также внесены сведения о дублетах, переданных в другие коллекции), координаты, число хромосом, полевой номер, примечания и т. д. Необходимо отметить, что большинство исследователей дополнительно к этому набору полей собирают специфичные данные, необходимые для их исследований. Поэтому применение структурных спецификаций позволило довольно быстро учитывать пожелания исследователей, создавать новые таблицы и добавлять новые поля. Созданы следующие таблицы:

11) БД «Гербарий сосудистых растений Сибирского института физиологии и биохимии растений СО РАН» (IRK);
12) БД «Гербарий мхов Сибирского института физиологии и биохимии растений СО РАН» (IRK);
13) БД «Адвентивные виды сосудистых растений Южной Сибири»;
14) БД «Первый находки чужеродных видов растений на территории Байкальской Сибири (Хронология, география);

15)БД «Флора сосудистых растений Заповедного Прибайкалья».

Ввод и редактирование данных осуществляется в ячейках таблицы или на сгенерированной форме (отдельными карточками для каждой учетной записи) (рисунок В.2). Количество полей в таблице составляет более двадцати. В стандартной форме сервисов ввода и редактирования элементы управления ввода полей отображаются вертикальным списком и занимают достаточно много места. Поэтому разработан специальный шаблон формы, на котором выполнена компоновка элементов управления для уменьшения времени ввода данных, наиболее важные и часто заполняемые поля вынесены в начало, уменьшено экранное расстояние между элементами.

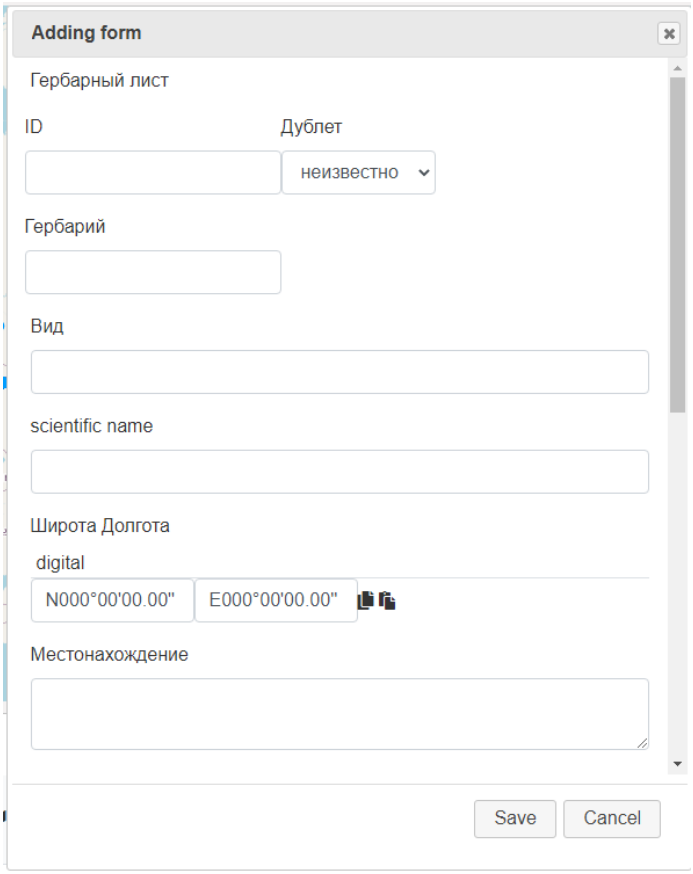


Рисунок В.2 – Пример редактирования данных на форме

Для полей с географическими координатами создаются картографические слои на карте (рисунок В.3.), а точки на карте являются ссылками на записи таблицы, информация о которых хранится в ИАС. На карте возможно использование различных слоев в виде подложки – Google satellite, Google road map, Yandex map, 2 GIS, Bing satellite, OpenStreetMap (OSM). В системе имеются слои, которые также

можно использовать совместно с данными гербарных листов, например, административное деление РФ, почвенные карты, кадастровое деление РФ и т. д.

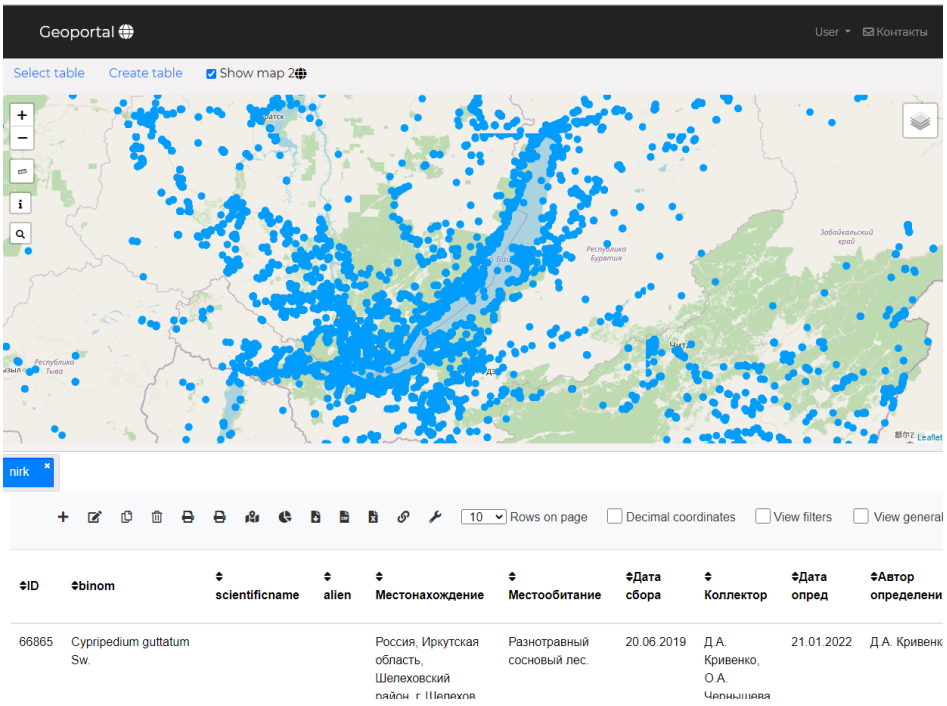


Рисунок В.3 – Карта, отображающая распространение видов

Также к записи могут быть добавлены изображения (рисунок В.4). Пользователь может загрузить их из локальной машины, либо выбрать в СХД.

Рисунок В.4 – Добавление изображений к учетным записям

Реализована печать данных (рисунок В.5) на основе шаблонов печати, отвечающих потребностям пользователей. Выполнена компоновка по 6 этикеток на странице.

ID 66870
Гербарий СИФИБР СО РАН (IRK)
Paris quadrifolia L.
N051°33'06.20" E105°05'34.70"
Россия, Республика Бурятия, Кабанский район, южный берег оз. Байкал, окраина п. Танхой, левый берег р. Осиновка, выс. 458 м над ур. моря.
Смешанный лес.
20.06.2021 Leg. Д.А. Кривенко, О.А. Чернышева
23.01.2022 Det. Д.А. Кривенко

ID 66867
Гербарий СИФИБР СО РАН (IRK)
Medicago lupulina L.
N051°27'31.85" E104°47'49.57"
Россия, Республика Бурятия, Кабанский район, юго-восточное побережье оз. Байкал, окрестности р. Малый Мамай.
У полотна железной дороги.
08.09.2003 Leg. М.М. Иванова
Det. М.М. Иванова

ID 66868
Гербарий СИФИБР СО РАН (IRK)
Myosotis imitata Serg.
N051°53'34.25" E104°49'39.96"
Россия, Иркутская область, Иркутский район, правый берег р. Ангара, близ п. Никола, Байкальский тракт.
Обочина дороги.
04.06.2018 Leg. Д.А. Кривенко, О.А. Чернышева
22.01.2022 Det. Д.А. Кривенко

ID 66869 Дублет
Гербарий СИФИБР СО РАН (IRK)
Myosotis imitata Serg.
N051°53'34.25" E104°49'39.96"
Россия, Иркутская область, Иркутский район, правый берег р. Ангара, близ п. Никола, Байкальский тракт.
Обочина дороги.
04.06.2018 Leg. Д.А. Кривенко, О.А. Чернышева
22.01.2022 Det. Д.А. Кривенко

Рисунок В.5 – Форма печати данных гербарных листов

Основными данными ИАС по флоре Байкальской Сибири (Иркутская область, Республика Бурятия, Забайкальский край) послужила БД Гербария СИФИБР СО РАН (IRK). Кроме того, имеется ряд других источников данных.

Одной из задач специалистов при работе с данными гербарных листов является геокодирование, т. е. определение координат по словесному описанию местоположения растения. В этом описании встречаются различные топонимы. Для упрощения геокодирования разработан инструмент, который обеспечивает поиск по топонимам: населенным пунктам, рекам и т. п. Реализована возможность присваивания значения поля для группы записей. Применяется эта возможность для присваивания одних и тех же координат записям с одинаковым местоположением

При помощи ИАС прежняя настольная БД Гербария СИФИБР СО РАН, функционировавшая на Access, перешла на интернет платформу, обеспечивающую надежное хранение данных и обрела новые функциональные возможности. С ней доступна работа с любого компьютера, подключенного к сети Интернет, любого количества пользователей с регламентацией доступа к просмотру, редактированию и обработке данных. Есть возможность работы с несколькими таблицами одновременно, на фоновой карте отображаются точки из всех открытых таблиц. Они доступны обработке одним набором фильтров независимо от набора полей и объекта биологического разнообразия. Так, распространение животных и растений может рассматриваться совместно для выявления общих закономерностей.

Кроме того, данные по биоразнообразию могут анализироваться совместно с любыми другими данными. Например, данные по распространению инвазионных видов могут анализироваться совместно друг с другом (распространение инвазионных растений и инвазионных животных), с данными по распространению эндемичных видов, а также с данными по изменению плотности населения в регионе, антропогенной нарушенности территории, величине и направлению транспортного потока, климатическими данными.

Благодаря созданию ресурса время, затрачиваемое на работы с гербарной коллекцией, ведению ее БД, инвентаризацию флор и анализу информации существенно сократилось за счет снижения объемов технической рутинной работы. Так, этикетирование новых гербарных образцов сейчас является одновременно и

внесением учетной записи в БД, а присвоение учетной записи географических координат ведет к автоматическому нанесению точки на топографическую карту или тематические карты различного рода. Разработанная система фильтров используется для создания выборок по любому количеству полей таблицы в зависимости от поставленных целей. Общее количество записей во всех таблицах составляет свыше 194 тыс. и постоянно пополняется.

ПРИЛОЖЕНИЕ Г. ГЕОПОРТАЛ АТЛАС ИГ СО РАН

Совместно с Институтом географии СО РАН разработан геопортал «Экологический атлас Байкальского региона» для представления эколого-географической информации в виде единой многоаспектной и многоуровневой проблемно-ориентированной информационно-картографической системы. В рамках геопортала созданы разномасштабные аналитические, комплексные и интегральные экологические карты на участок Мирового природного наследия «Озеро Байкал»; буферную экологическую зону Байкальской природной территории (российская и монгольская части); на Байкальскую природную территорию в целом; на субъекты Российской Федерации Байкальского региона (Республика Бурятия, Иркутская область, Забайкальский край); на муниципальные районы Байкальского региона; на города и урбанизированные территории. Геопортал находится по адресу http://atlas.isc.irk.ru/. На рисунке Г.1 представлено главное окно геопортала. Для удобства просмотра карта занимает все пространство окна браузера. Все другие компоненты располагаются поверх карты и скрываются, кроме панели инструментов. Слева находится список карт. Справа расположена легенда карты. Справа внизу отображаются координаты мышки в проекции WGS 84 и текущий масштаб.

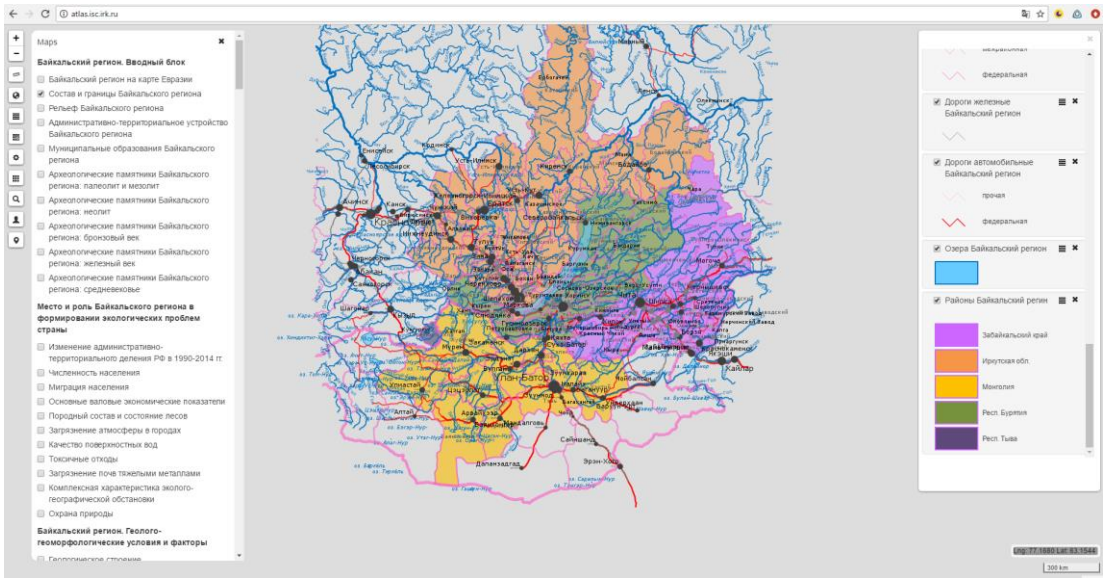


Рисунок Г.1 – Главное окно геопортала

В рамках разработки геопортала проведено развитие системы импорта данных и компонентов отображения карты. Для загрузки данных в ГИС форматах был модернизирован модуль импорта данных. Включена поддержка всех форматов пространственных данных, поддерживаемых библиотекой GDAL/OGR. Рассмотрим более подробно развитие компонентов отображения карт.

Реализовано формирование легенды, которое производится в браузере с помощью специального редактора стилей (рисунок Г.2). Легенда формируется для каждого слоя отдельно и хранится в описании слоя. Для хранения легенды используется формат JSON. Формат хранения стилей SLD поддерживается только частично в связи с низкой скоростью отображения карт при его использовании. В соответствии со стандартом WMS стили указываются в виде ссылки в URL слоя. Поэтому Mapserver должен для каждого тайла (фрагмента карты) выполнить его загрузку на сервер, что значительно увеличивает время отрисовки слоя. Для отображения слоев с помощью Mapserver стили в формате JSON конвертируются map файл.

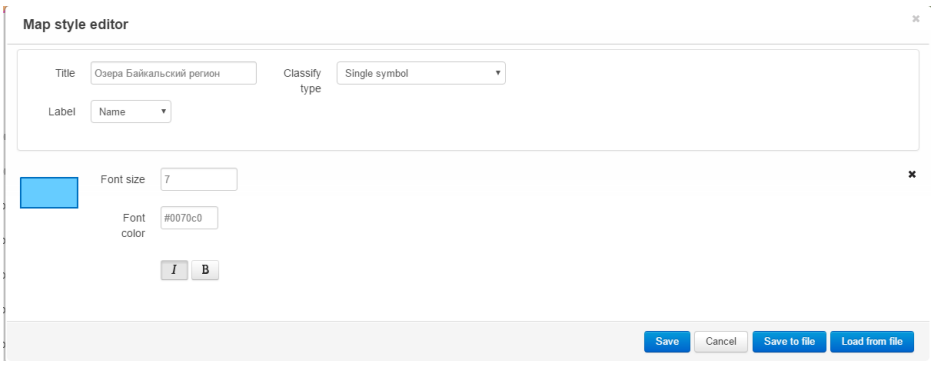


Рисунок Г.2 – Редактор стилей отображения слоев

Базовым элементом формирований стилей слоев является условный знак. Условный знак может быть векторным, растровом изображением, символ шрифта и диаграммой.

Условный знак может состоять из нескольких слоев. Для каждого слоя можно выбрать один из существующих контуров. Далее нужно указать видимость условного знака (opacity), цвет (color), размер (size), угол поворота (angle), смещение

(dx, dy). Порядок слоев можно менять. В рамках геопортала имеется таблица контуров, в которой можно задать собственный контур в виде набора пар точек.

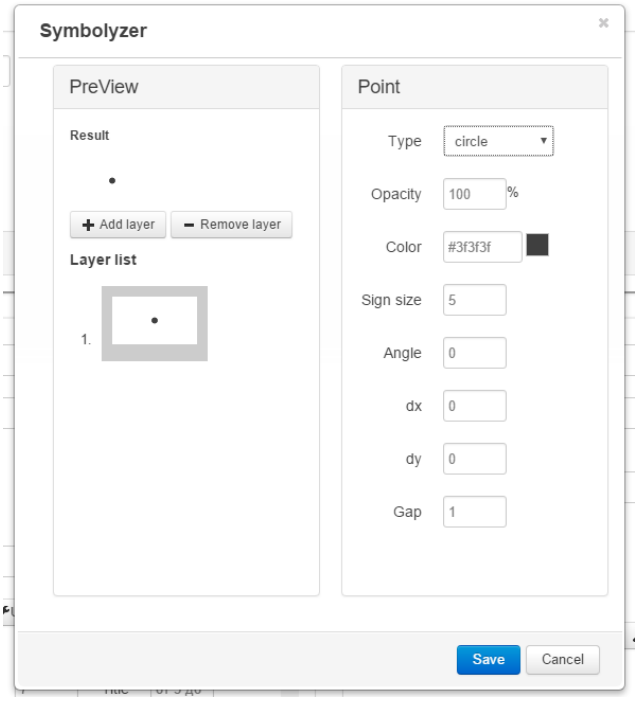


Рисунок Г.3 – Редактор условного знака

Пользователь может загрузить в геопортал растровое изображение и использовать его в качестве условного знака. Поддерживаются форматы: svg, png. Во многих ГИС в качестве условного знака можно использовать шрифты TrueType. В рамках геопортала эта возможность также реализована. Пользователь может использовать символы заранее загруженных шрифтов. На текущий момент используется шрифт ГИС MapInfo.

Редактор стилей (рисунок Г.4) позволяет классифицировать данные по нескольким атрибутам. Каждому классу назначается способ отображения, который зависит от значений атрибутов. Для каждого атрибута необходимо указать, изменяется цвет или размер. Для цвета и размера задается диапазон, в рамках которого они назначаются классам. Диапазоны значений атрибутов задаются автоматически или вручную. Можно указать порядок значений атрибутов (сортировку).

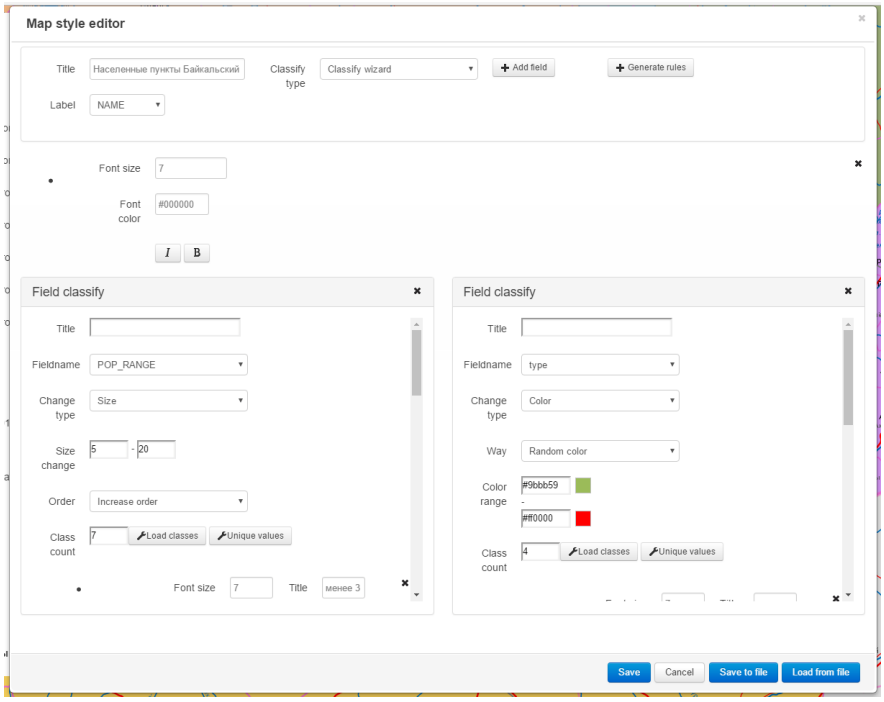


Рисунок Г.4 – Формирование стилей по двум атрибутам

Поддерживается два типа диаграмм (рисунок Г.5, Г.6): круговые диаграммы и столбчатые. Пользователю необходимо указать, по каким атрибутам формируются диаграммы. Для каждого атрибута можно задать цвет и заголовок. Можно задать изменение размера диаграммы в соответствии со значением указанного атрибута.

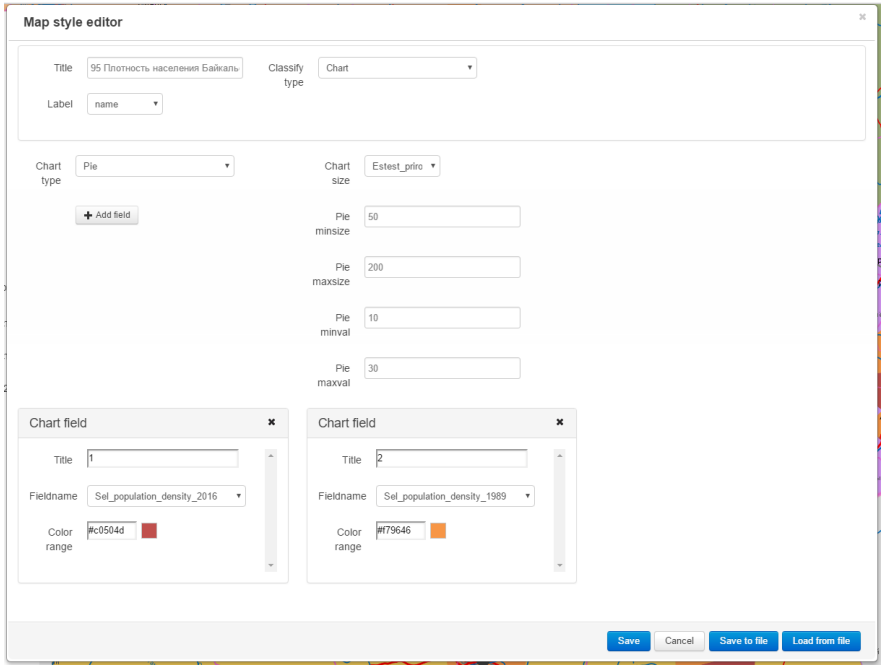


Рисунок Г.5 – Создание диаграмм

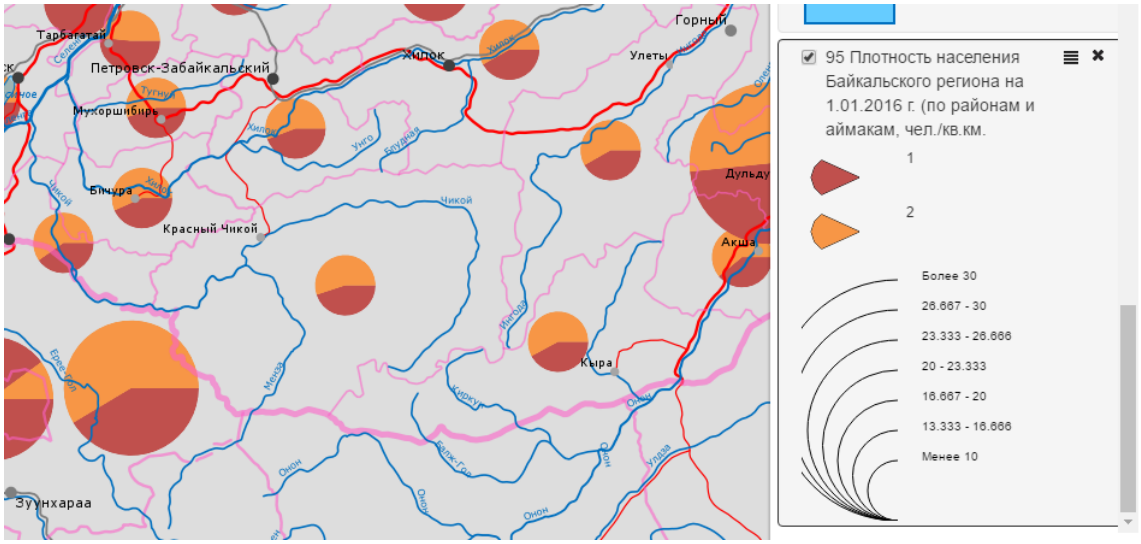


Рисунок Г.6 – Отображение диаграмм

В рамках геопортала создана специальная таблица для хранения карт. Каждая запись этой таблицы содержит информацию о карте. Карта – это логическая единица, которая определяет набор слоев и их порядок, название карты (отображается в левой панели), раздел, требуемый масштаб и центр окна и карта подложка. В качестве карты подложки могут использованы:

1) Google satellite
2) Yandex
3) 2GIS
4) Bing
5) OSM
6) None
7) Cadastr

Зарегистрированный пользователь может создать свою карту.

Для ускорения отображения карт применяется MapCache, осуществляющий кэширование растровых тайлов (растровые изображения фрагментов карт, передаваемые сервером браузеру для отображения). MapCache предварительно отрисовывает все карты и сохраняет в файловой системе (кэширует). Далее вместо повторной отрисовки MapCache копирует файл и передает его браузеру. За счёт этого механизма процесс отрисовки значительно ускоряется. Кэширование карты

достаточно долгий процесс. Поэтому при разработке карт этот механизм не используется.

ПРИЛОЖЕНИЕ Д. ГЕОПОРТАЛ ИНФОРМАЦИОННАЯ СИСТЕМА «L.»

Первая версия геопортала Информационная система (ИС) «L.» (рисунок Д.1) запущена в 2018 г. Геопортал (https://isling.org) предназначен для внесения, хранения, организации и анализа данных из разных источников (наблюдения, образцы коллекций, литература и т. п.), разного типа (текст, изображения, данные по ДНК и т. п.) по всем группам живых организмов. Целью L. является обеспечение любого исследователя инструментарием полного цикла работы с данными – от работы в коллекциях до организации и анализа. На этапе разработки геопортала добавлена функция регистрации пользователей. Для вхождения в определенную исследовательскую группу права пользователей повышаются администратором, но вносить стандартные образцы может любой зарегистрировавшийся.

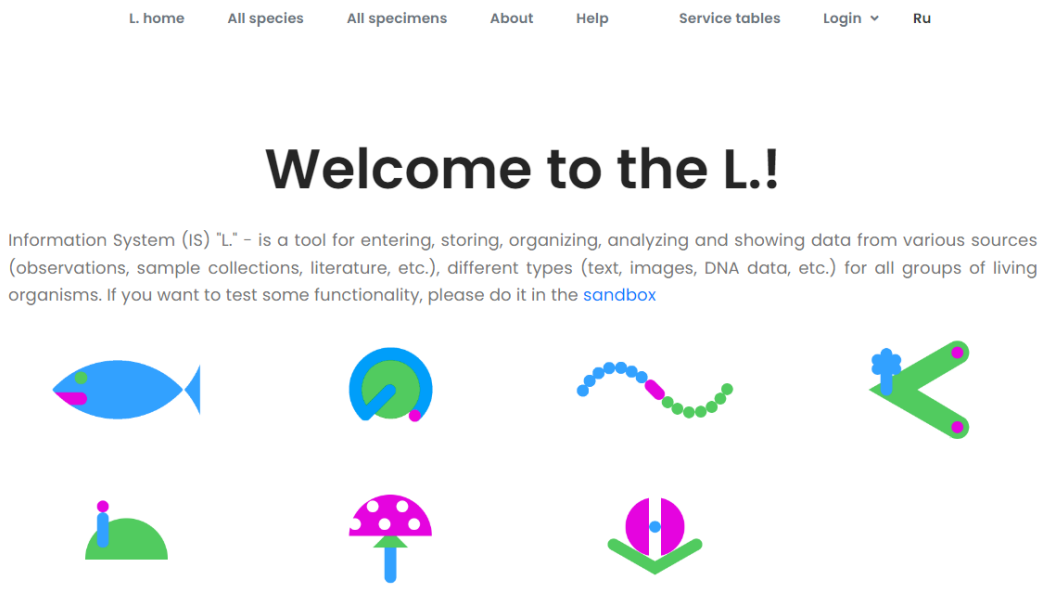


Рисунок Д.1 – Информационная система (ИС) «L.»

Большое внимание в L. уделено персонализации форм и представлений образцов. Для каждой исследовательской группы экранная форма образца (и его представление как отдельного образца, так и в списке образцов или видов) может выглядеть по-разному – дополнительные поля можно добавлять как к образцу в целом, так и к видам в образце.

Реализована возможность внесения в один образец видов организмов из любой биологической группы – образец может содержать, например, мхи,

лишайники, все симбионты лишайников, паразиты лишайников, бактерии которыми больны паразиты лишайников, вирусы, поразившие этих бактерий и т. п. – и все эти организмы можно учитывать из этого образца в соответствующих групповых анализах.

Помимо обычного вывода списка видов, списка образцов, печати этикеток (персонализированной для каждой коллекции) реализован экспорт данных в форматах KML и CSV. Также возможна организация долговременных выборок – подпроектов, представляющих собой списки видов по определенной территории или таксономической группе или ограниченными группой различных фильтров.

ИС L. способна обслуживать тематические подпроекты без необходимости переноса собранных данных в «специально сделанные для этого сайты». Например, ведение Красной книги страны или региона организуется всего лишь созданием постоянной выборки с фильтром по региону и отношению видов к охраняемым. Аналогично решаются вопросы флор и фаун.

Кроме того, реализован ряд дополнительных функций геопортала для удобства работы пользователей:

1) при внесении образца, старые синонимы автоматически заменяются на текущие;
2) возможен ввод координат в любом формате (с встроенной проверкой, не позволяющей внести некорректные координаты) или с помощью мыши на карте;
3) добавлена возможность прикрепления ссылок на другие (связанные) образцы или на литературу к любому образцу и выбора лицензии для фотографий;
4) реализовано прикрепление det.name к виду в образце: у каждого вида в образце может быть свой автор определения (переопределения – в истории образца);
5) реализовано автозаполнение вида хранения (store species – вид, «по которому» образец лежит в коллекции), store species заполняется первым видом из списка видов в образце и может быть внесен вручную при необходимости (например, если хранится по старой системе);
6) отображается связанность дочерних (копий) и материнских образцов. Например, можно узнать о переопределении материнского образца со страницы дублета (и наоборот).

ПРИЛОЖЕНИЕ Е. ГЕОПОРТАЛ ИЗК СО РАН

Совместно с Институтом земной коры СО РАН разработан геопортал (рисунок Е.1) «Сейсмотектоническое и инженерно-геологическое районирование», обеспечивающего сбор, хранение и обработку данных, описывающих геологические процессы в байкало-монгольском регионе на территориях активного природопользования. Для бесперебойного функционирования с возможностью удаленного доступа к системе через глобальную сеть Интернет выделен виртуальный сервер, с назначенным URI: http://gp.icc.ru. Геопортал является точкой доступа к сервисам геообработки и пространственным данным. Базой любого сейсмического районирования является каталог землетрясений исследуемой территории. Для территории России и Байкальской природной территории таким набором является открытый Специализированный каталог землетрясений для задач общего сейсмического районирования территорий Российской Федерации (СКЗ), который содержит около 32 000 событий. СКЗ был загружен на геопортал. В геопортале созданы следующие таблицы:

1) Каталог землетрясений;
2) Результаты оценки динамических параметров очагов землетрясений, вычисленные с помощью рекуррентных интервалов;
3) Эпицентры землетрясений с магнитудами M за период 1900-2014 гг. по каталогу NEIC;
4) Механизмы очагов сильных (M 5.5) землетрясений Монголо-Байкальского сейсмического пояса (1950-2012 гг.);
5) Эпицентры землетрясений;
6) Результаты оценки динамических параметров очагов землетрясений, вычисленные с помощью рекуррентных интервалов;

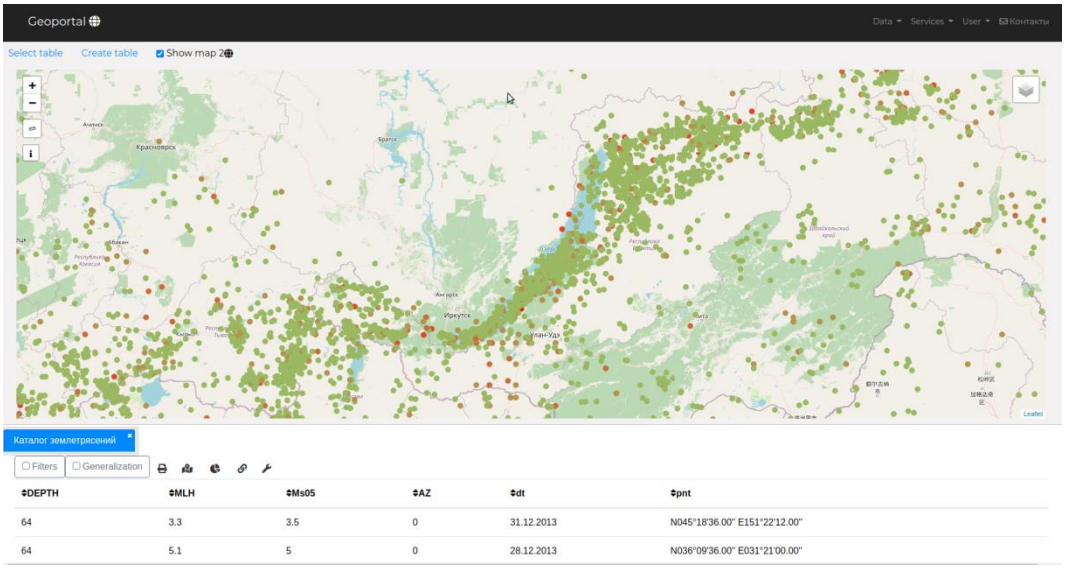


Рисунок Е.1 – Отображение точек землетрясений

WPS сервис сейсмического районирования

Данные приведённого каталога позволяют произвести оценку сейсмичности Байкальской природной территории. Для этого на геопортале был реализован WPS сервис на основе макросейсмического уравнения. Сервис работает следующим образом - для каждой точки карты по заданной сетке находится максимальная балльность по уравнению перебором всего каталога. В результате получается массив максимальной балльности для региона. По этому массиву в дальнейшем можно построить карту сейсмической опасности, которая будет представлять собой изолинии максимальной балльности. WPS сервис предназначен для решения задач сейсмического макрорайонирования.

Сервис работает следующим образом – для каждой точки карты по заданной сетке находится максимальная балльность по макросейсмическому уравнению

$$I = 1.5\,M - S\lg\sqrt{(\Delta^2 + H^2)} + C$$

где I – сила толчка в баллах, M – магнитуда $(K=4+1.8M)$, Δ – расстояние (км), значение глубины гипоцентра везде 10 км.) перебором всего каталога землетресений. В результате получается массив максимальной балльности для региона. По этому массиву в дальнейшем можно построить карту сейсмической опасности, которая будет представлять собой изолинии максимальной балльности [42-44].

ПРИЛОЖЕНИЕ Ж. ГЕОПОРТАЛ «ОЧАГИ РАСПРОСТРАНЕНИЯ ИКСОДОВЫХ КЛЕЩЕЙ»

Геопортал развернут по адресу http://tbd.icc.ru/. Работа ведется совместно с Центром диагностики и профилактики клещевых инфекций с 2012 года. На территории Иркутской области отмечается высокая активность иксодовых клещей. Так, например, в 2012 году зарегистрировано 7148 обращений по поводу укусов иксодовыми клещами в медицинско-профилактические учреждения Иркутской области. Большинство из обратившихся проживают в южных районах Иркутской области (рисунок Ж.1). Однако единичные случаи обращений фиксируются на всей территории Иркутской области во всех 33 муниципальных районах.

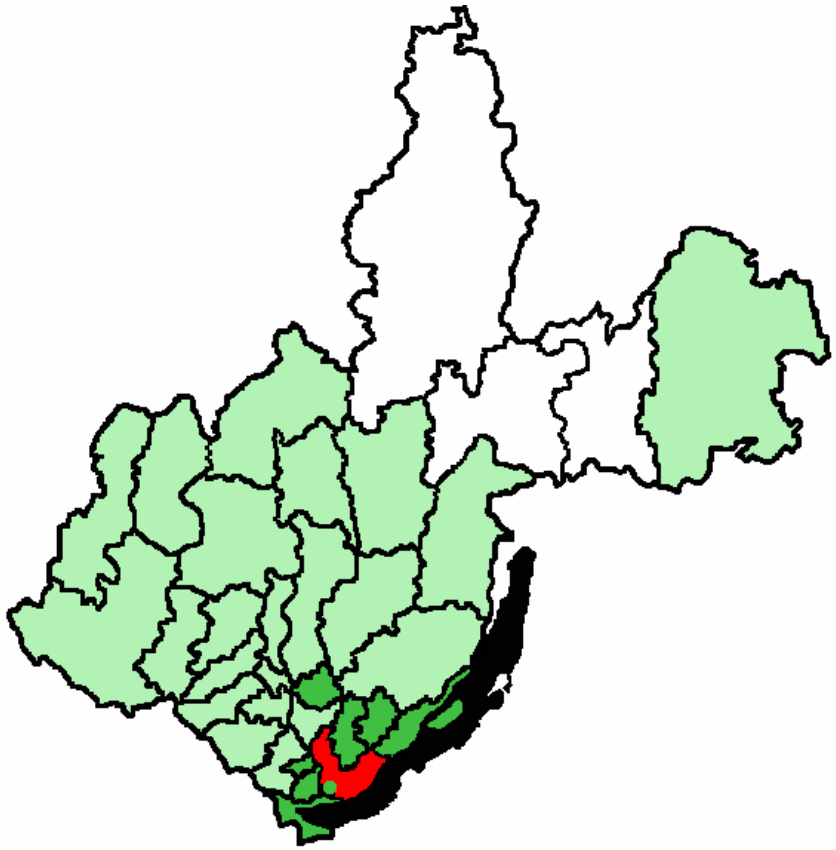


Рисунок Ж.1 – Пространственная структура обращаемости населения за помощью после укусов клещей

Полученная информация о случаях укусов людей клещами, лабораторных исследованиях, а также проведённой профилактической и лечебной работе,

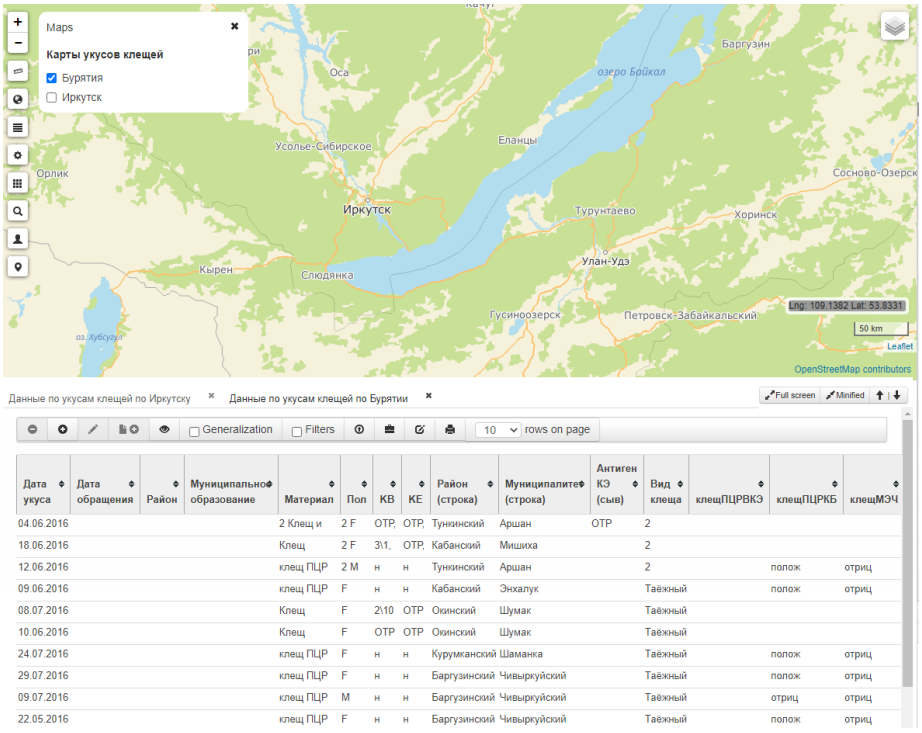
накапливается на протяжении многих лет. Можно выделить следующие проблемы, касающиеся её хранения и обработки:

Большие объемы разнородных данных, в том числе на бумажных носителях (требуется сканирование и распознавание). Такой способ хранения затрудняет анализ всей имеющейся информации по активности иксодовых клещей в Иркутской области.

Имеющиеся базы данных локализованы в медицинских, научных учреждениях и их структурных подразделениях. Такое состояние затрудняет обмен и комплексирование накопленной информации.

Слабая интеграция. Практически отсутствует прямой обмен данными в учреждениях, занимающихся анализом и профилактикой клещевых инфекций, не проводится работа по созданию единого информационного пространства. Отсутствует возможность удаленного доступа к имеющимся базам данных, не существует единой базы данных, содержащей сведения по распространению и активности иксодовых клещей.

Всё перечисленное обосновывает актуальность создания геопортала (рисунок Ж.2), позволяющий интегрировать имеющуюся разноформатную пользовательскую информацию, проводить анализ данных, генерировать различные отчеты, в т.ч. в виде тематических карт, предоставлять информацию об ареалах распространения и активности клещей, обеспечивать построение зависимостей между факторами, влияющими на распространение клещей и переносимые ими заболевания, проводить анализ данных с учетом пространственной и временной составляющих, прогнозировать очаги распространения иксодовых клещей в зависимости от года и периода.



Дата укуса	Дата обращения	Район	Муниципальное образование	Материал	Пол	КВ	КЕ	Район (строка)	Муниципалитет (строка)	Антиген КЭ (сыв)	Вид клеща	клещПЦРВКЭ	клещПЦРКБ	клещМЭЧ
04.06.2016				2 Клещ и	2 F	ОТР,	ОТР,	Тункинский	Аршан	ОТР	2			
18.06.2016				Клещ	2 F	3\1,	ОТР,	Кабанский	Мишиха		2			
12.06.2016				клещ ПЦР	2 M	н	н	Тункинский	Аршан		2		полож	отриц
09.06.2016				клещ ПЦР	F	н	н	Кабанский	Энхалук		Таёжный		полож	отриц
08.07.2016				Клещ	F	2\10	ОТР	Окинский	Шумак		Таёжный			
10.06.2016				Клещ	F	ОТР	ОТР	Окинский	Шумак		Таёжный			
24.07.2016				клещ ПЦР	F	н	н	Курумканский	Шаманка		Таёжный		полож	отриц
29.07.2016				клещ ПЦР	F	н	н	Баргузинский	Чивыркуйский		Таёжный		полож	отриц
09.07.2016				клещ ПЦР	M	н	н	Баргузинский	Чивыркуйский		Таёжный		отриц	отриц
22.05.2016				клещ ПЦР	F	н	н	Баргузинский	Чивыркуйский		Таёжный		полож	отриц

Рисунок Ж.2 – Основная страница геопортала «Очаги распространения иксодовых клещей»

Проведены работы по формализации географических названий ареалов активности иксодовых клещей и осуществлена их пространственная привязка. Предложены технологические решения по нормализации разнородных пользовательских данных и привязки их к стандартизованным справочникам в т.ч. с иерархической структурой. Созданы сервисы проведения пространственного анализа данных. Все разработанные сервисы интегрированы и представлены единым программным комплексом. Так, было проведено обобщение алгоритмов геокодирования для автоматического или полуавтоматического приведения разнородных данных к единым справочникам с иерархической структурой для осуществления совместного анализа.

Центром диагностики и профилактики клещевых инфекций были предоставлены обезличенные данные об укусах иксодовыми клещами по результатам обращений пострадавших за период 2013-2020 гг. Данные используются для отображения на тематических картах, а также для оценки и прогнозирования активности иксодовых клещей на территории БПТ. Данные о предполагаемом месте укуса (со слов пострадавших) указываются произвольно со

слов укушенного. Названия территорий в ряде случаев содержат опечатки, народные названия и т. п. В связи с этим была решена актуальная задача, по созданию сервиса, обеспечивающего геокодирование названий местности – назначения географических идентификаторов для имеющихся названий.

Для проведения гекодирования были приведены подготовительные работы и создан RESTful Web сервис, осуществляющий гекодирование, используя нечеткое сравнение строк. Подготовительные работы заключались в экспертном анализе названий местности, которая не идентифицировалась по результатам запросов в OpenStreetMap API (https://www.openstreetmap.org) и определения её координат на карте. Эксперт также определял альтернативное, корректное название местности. Рассматривались названия, встречающиеся четыре и более раза. Менее часто встречающиеся названия были исключены из рассмотрения, т. к. они не являются значимыми и не влияют на полноту предоставления о территориях, на которых люди подвергаются укусам клещей. Также из рассмотрения были исключены зарубежные территории. Во время подготовительных работ было принято решение не проводить исправление исходных названий. Решения было принято в связи с тем, что подобные ошибки повторяются из года в год и предлагаемый подход позволяет ускорить процесс геокодирования новых данных. Пример гекоодированных неформализованных (народных) названий местности показан на рисунке Ж.3.

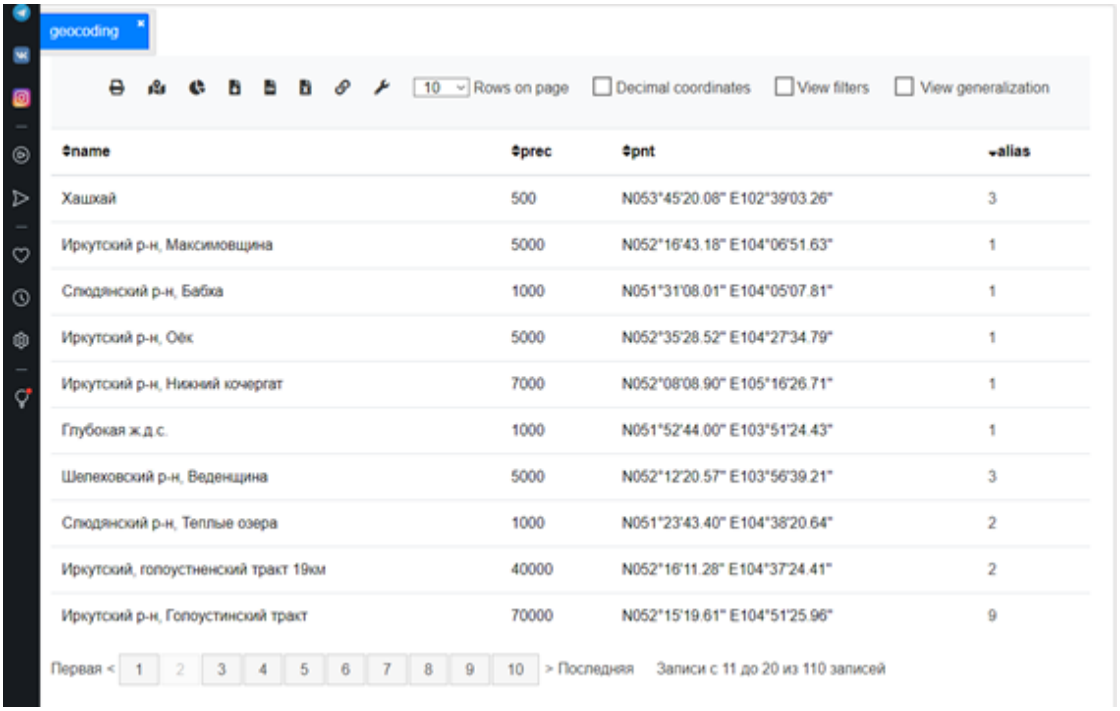


Рисунок Ж.3 – Пример геокодирования неформализованных названий местности

Если эксперт замечал, что для одной и той же местности используются различные названия, то он мог задать список альтернативных названий для данной местности. Пример такого списка приведен на рисунке Ж.4. Фрагмент карты, иллюстрирующей результаты геокодирования, представлен на рисунке Ж.5.

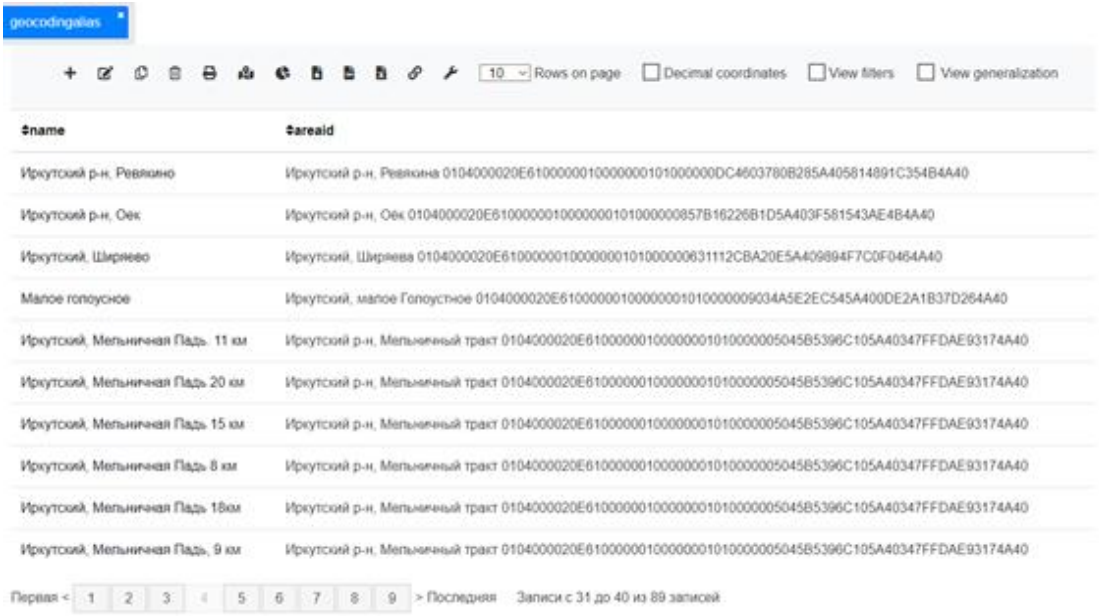


Рисунок Ж.4 – Пример таблицы альтернативных названий местности

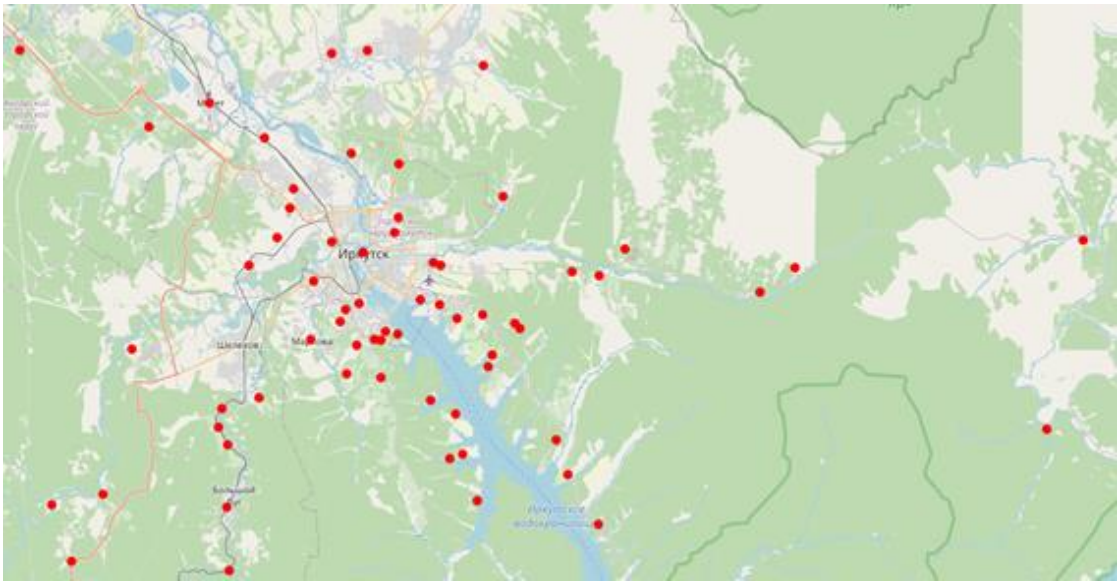


Рисунок Ж.5 – Фрагмент карты с результатами геокодирования мест укусов иксодовыми клещами

Созданные таблицы в дальнейшем используются для проведения операций геокодирования разработанным сервисом. Сервис представляет собой Flask (https://flask.palletsprojects.com) приложение. Рассмотрим основные принципы его работы.

В качестве входных данных по HTTP-протоколу сервис получает GET-запрос с названием местности. Если в названии не обозначен регион, то строка дублируется и к названию одной добавляется «Иркутская область», а другой «Республика Бурятия». После чего по API OpenStreetMap делается запрос к ресурсу, с целью получить координаты запрошенной местности. В полученных результатах существует оценка полноты совпадения. Все результаты, имеющие точность менее 50% не учитываются. Затем делается запрос к таблице геокодированных экспертами названий и их альтернатив (алиасов). Полнота совпадения названий рассчитывается на похожесть строк методом триграмм. Нечеткое сравнение также позволяет нивелировать некоторые ошибки (опечатки) в названиях местности. Все совпадения более 50% также попадают в результирующую выборку. И этой выборок, для обозначения местности используется с самым большим коэффициентом похожести. Результат возвращается в виде JSON. Общая схема работы сервиса представлена на рисунке Ж.6.

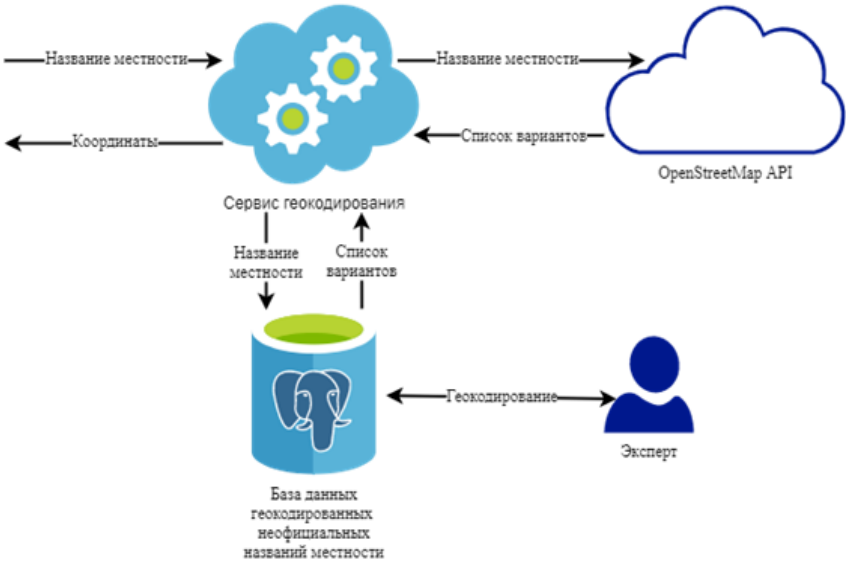


Рисунок Ж.6 – Общая схема работы сервиса

При помощи сервиса были обработаны 35618 записей. Из них геокодировано 31042 записей. Из нераспознанных записей часть не содержит информации о предполагаемом географическом месте укуса; указаны географические названия он относящиеся к Иркутской области и Республики Бурятия; названия населенных пунктов, которые невозможно идентифицировать без знания административного района.

Сервис пространственного статистического анализа данных

Основной информацией, поступающей из ФГБУ «НЦ ПЗСРЧ» СО РАМН являются сведения о дате, предположительном месте укуса клещом, его виде, поле, возрасте пострадавшего, наличие страхового полиса и прививки. Однако, в базу данных КИАС можно добавить множество тематических данных, имеющих пространственную привязку, позволяющих выявлять зависимости между различными факторами, влияющими на активность данного вида паукообразных. Примеры таких данных – информация об осадках и температуре воздуха, данные о рельефе местности и растительном покрове и т. п. Использование различных данных позволит выявлять факторы, влияющие на активность иксодовых клещей.

Так, например, на рисунке Ж.7 показана сезонная динамика обращаемости населения с укусами клещей. По результатам анализа видно, что пик активности

иксодовых клещей приходится на май месяц для нечетных годов и на июнь месяц для четных годов.

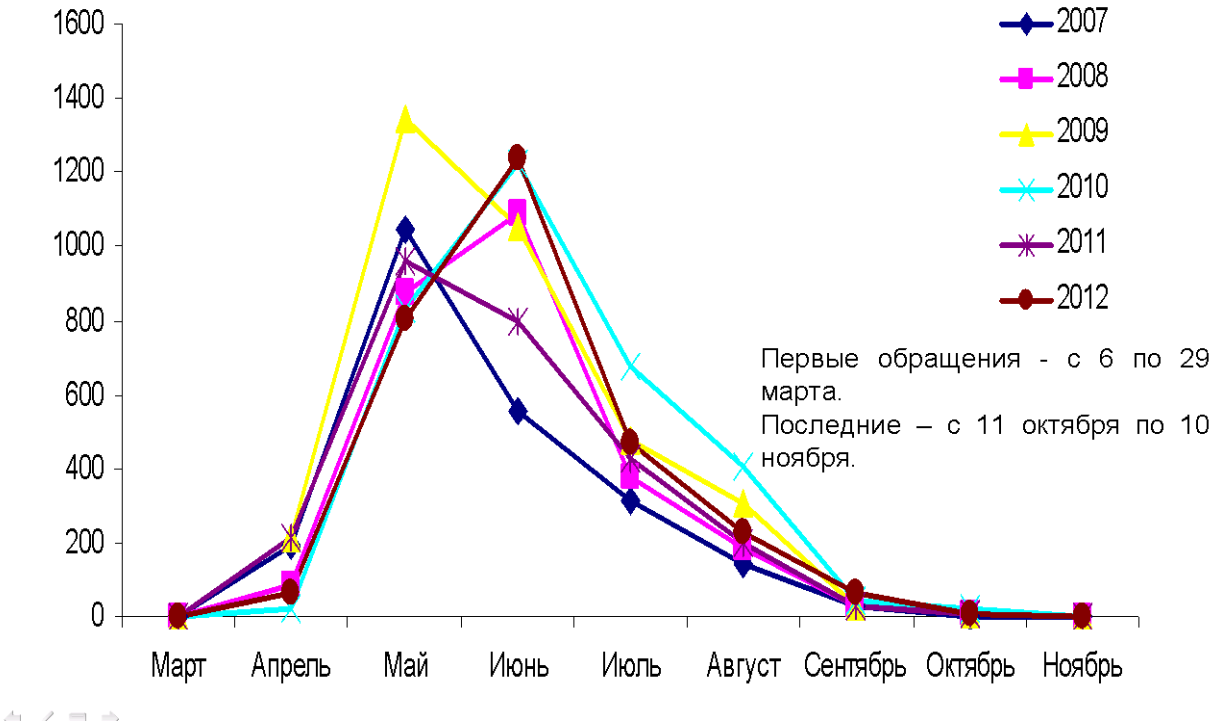


Рисунок Ж.7 – Сезонная динамика обращаемости населения с укусами клещей

Для обеспечения высоких показателей мобильности модулей КИАС относительно адаптируемости и мобильности было принято решение реализовывать аналитические функции в виде WPS-методов. Сервисы статистического анализа данных используют разработанную кроссплатформеную программную библиотеку, реализующую статистические методы регрессии и корреляции. Библиотека оформлена в виде, допускающим регистрацию как WPS-метод для ZOO-project сервера. Программный код библиотеки реализован на языке C++ в среде Visual Studio 2008. Для компиляции использован компилятор C++ compiler из среды Visual Studio. Однако структура модуля и используемые в нем программные библиотеки позволяют выполнять компиляцию для различных программных сред с использованием компилятора GCC (GNU Compiler Collection).

Практическое использование данных методов позволяет оценить зависимость между различными тематическими данными, имеющими географическую привязку. Данные для анализа должны быть представлены в формате GeoTIFF.

Для обработки данных используются математические методы библиотеки ALGLIB Free edition (http://alglib.net). Данная библиотека является широкораспространенной, и содержит большой набор методов для проведения статистического и алгебраического анализа данных.

Разработанная программная библиотека содержит два основных метода:

- corr – позволяет находить коэффициент корреляции между двух массивов данных;
- regVal – позволяет находить коэффициенты регрессии между двумя массивами данных.

Обрабатываются данные, организованные в виде структур fMatrix3D.

Метод corr содержит 3 параметра и возвращает значение типа float по результатам выполненной работы. Параметры метода:

- m1 – указатель типа fMatrix3D представляющий первый массив данных для корреляционного анализа;
- m2 – указатель типа fMatrix3D представляющий второй массив данных для корреляционного анализа;
- method – указатель типа char на метод корреляционного анализа. Поддерживаются значения «person» - вычисления коэффициента корреляции методом Пирсона; «spearman» – вычисление коэффициента ранговой корреляции методом Спирмана.

В методе реализована предварительная проверка анализа размерности сравниваемых структур. В случае, если размерности не совпадают, то метод возвращает значение «2». В случае если при выполнении метода происходит какая-

либо ошибка, то метод возвращает значение «-2». В случае успешного выполнения возвращаемое значение принадлежит диапазону от -1 до 1.

Результат корреляции значений передается клиенту, а также записывается в лог-файл.

Метод regVal содержит два параметра передающихся по значению (in-параметры):

- m1 – указатель типа fMatrix3D представляющий первый массив данных для регрессионного анализа;
- m2 – указатель типа fMatrix3D представляющий второй массив данных для регрессионного анализа;

а также 2 параметра передающихся по ссылке (out-параметры) возвращающих значения регрессионного анализа:

- параметр v типа real_1d_array (одномерный массив, структура данных используемая в библиотеке alglib), представляющий коэффициенты регрессионного анализа;
- параметр NVars типа ae_int_t (целочисленный тип данных, используемый в библиотеке alglib) определяющий число независимых переменных в регрессии.

Метод возвращает значение типа bool. Возможные варианты “true” – если метод выполнен без ошибок и “false” в противном случае.

Анализируемые данные проходят предварительную проверку на эквивалентности размерности данных.

Результаты корреляционного анализа передаются сервером клиенту, а также записываются в специализированный лог-файл.

Для применения существующих в ГИС методов анализа удобно представить изучаемую территорию в виде набора ячеек регулярной сетки. В ГИС такое представление называется GRID моделью. Основным форматом GRID модели,

используемой в разрабатываемой ГИС, является GeoTiff. Общий алгоритм проведения моделирования разбивается на ряд шагов, которые выполняются с помощью сервисов, реализующих стандарты программного взаимодействия геоинформационных систем.

ПРИЛОЖЕНИЕ З. ПРОТОТИП ГЕОПОРТАЛА ИРКУТСКОЙ ОБЛАСТИ

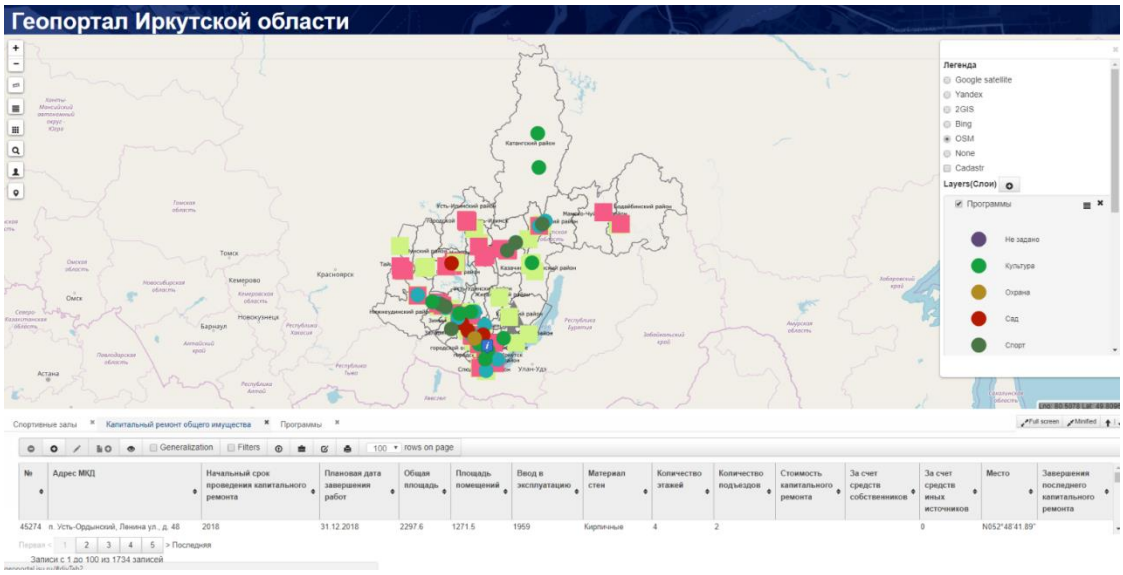


Рисунок З.1 – Прототип геопортала Иркутской области

Цели создания:

- Распространение географически привязанной информации о состоянии территорий Иркутской области и деятельности органов государственной власти на территории Иркутской области среди населения;

- Создание общего географического информационно-аналитического ресурса для специалистов Администрации Иркутской области (закрытая версия портала).

Наполнение информацией:

Прототип геопортала Иркутской области (рисунок З.1.) демонстрируется на базе геопортала ИГУ, разрабатываемого с использованием системы Фарамант. На геопортале ИГУ будут собраны слои, подготавливаемые специалистами ИГУ (географы, биологи, экономисты, и т. д.). Для демонстрации возможностей системы на геопортал были импортированы данные, предоставленной сотрудниками Администрации Иркутской области. В результате был создан ряд слоёв, позволяющих проиллюстрировать основные возможности системы с использованием конкретных данных:

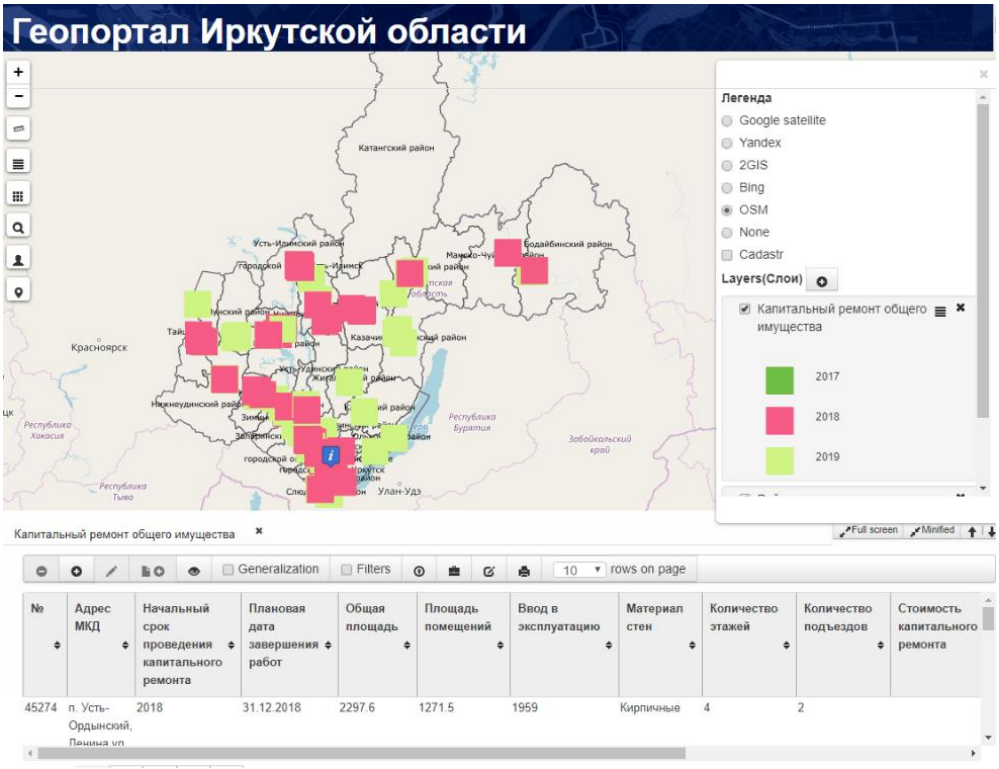


Рисунок 3.2 – Капитальный ремонт общего имущества

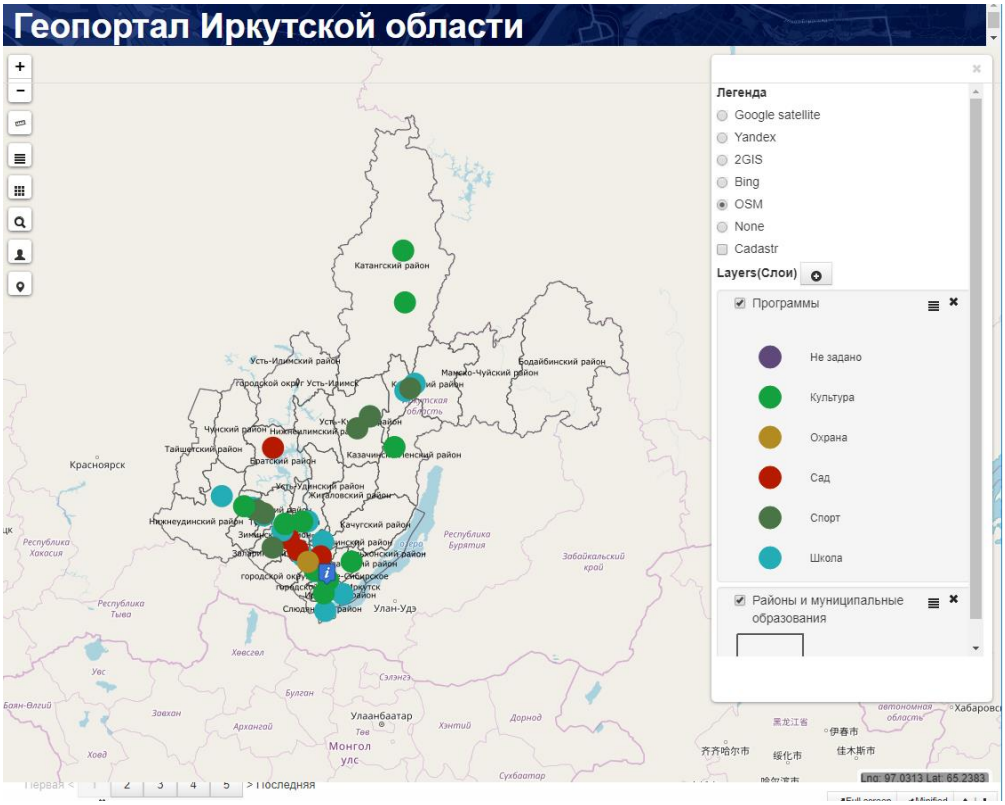


Рисунок 3.3 – Программы

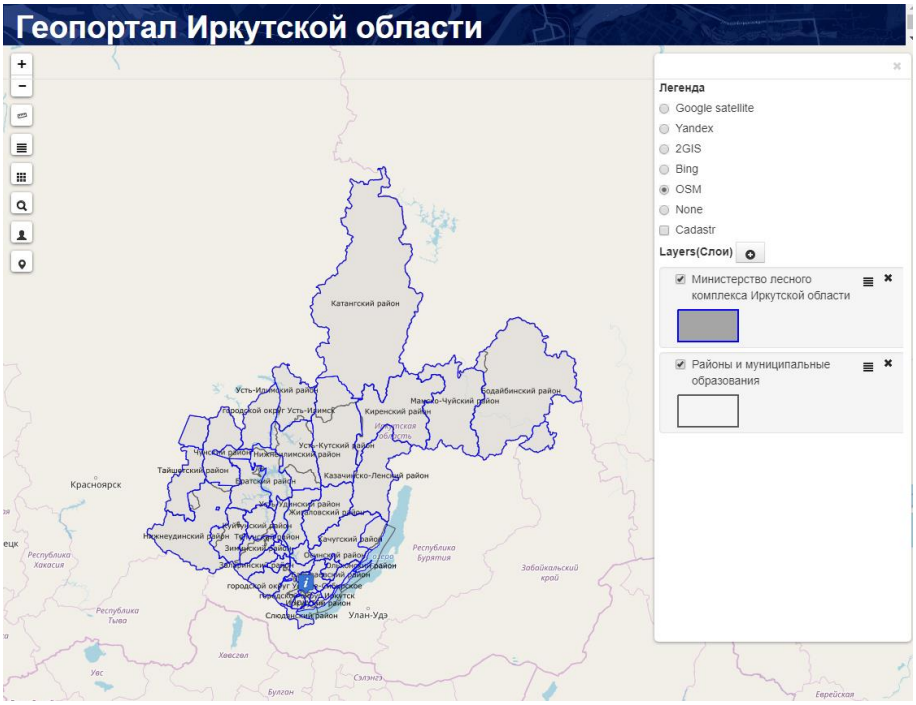


Рисунок 3.4 – Министерство лесного комплекса Иркутской области

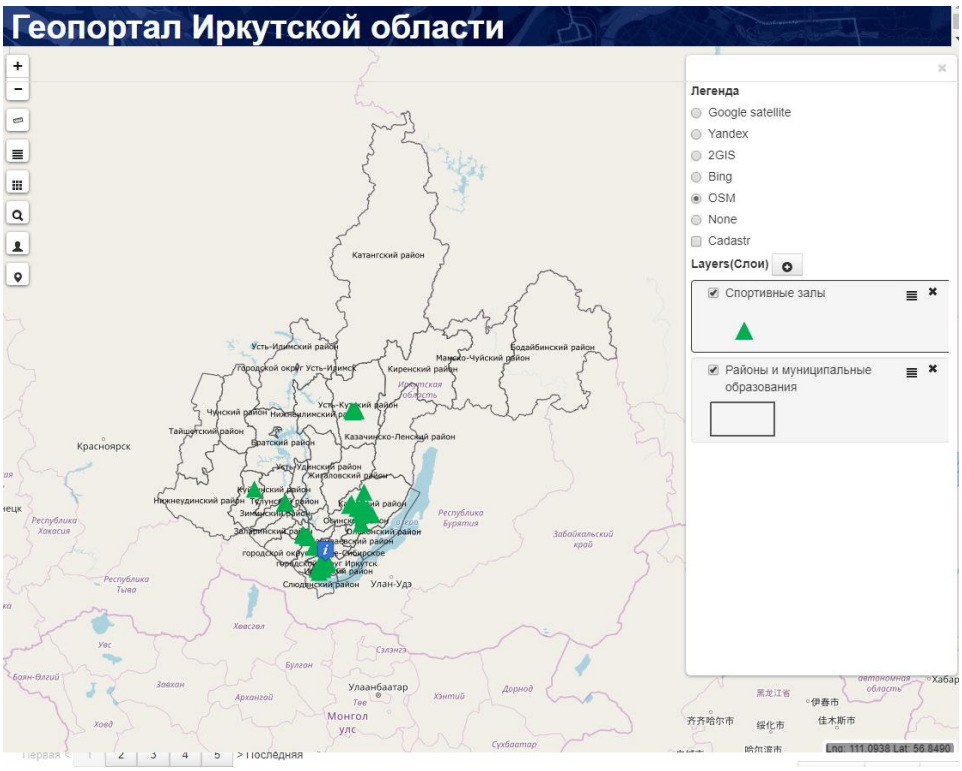


Рисунок 3.5 – Спортивные залы

ПРИЛОЖЕНИЕ И. КЛАССИФИКАЦИЯ ТЕРРИТОРИИ МЕТОДОМ ОПОРНЫХ ВЕКТОРОВ

Классификация территории методом опорных векторов

В настоящее время одним из основных источников информации о структуре и параметрах растительного покрова выступают данные экспедиционных исследований и дистанционного зондирования (ДДЗ). Одним из активно развивающихся методов анализа растительного покрова с использованием ДДЗ является метод, основанный на установлении взаимосвязей между спектральными характеристиками снимка и биологическими параметрами лесного покрова (запас фитомассы, видовой состав и др.) [Владимиров, 2012; Кузьменко и др., 2013; Владимиров и др. 2014]. Общеизвестно, что спектральные кривые природных объектов определяются тремя зонами спектра: зеленой, красной и ближней инфракрасной. Характеристики отраженного излучения в значительной степени определяется содержанием хлорофилла и уровнем влагообеспеченности зеленых фракций древесной растительности. В настоящее время, активно используются относительные показатели, получаемые на основе спектральных индексов, коррелирующих с уровнем обеспеченности растений хлорофиллом - нормализованный разностный вегетационный индекс NDVI [Владимиров, 2012; Кузьменко и др., 2013; Владимиров и др. 2014].

Как правило, процесс анализа данных ДЗЗ и составления карт занимает много времени, т. к. большая часть работы проводится специалистом вручную. В данной статье предлагается технология обработки пространственных данных методом опорных векторов, которая упрощает процесс составления тематических карт. Метод опорных векторов успешно используется в обработке пространственных данных и основан на построении оптимальной разделяющей гиперплоскости в пространстве признаков. Гиперплоскость максимально удаленная от ближайших к ней точек разделяемых классов считается оптимальной. На практике чаще используется разделение двух классов. Поэтому, для того чтобы выделить определенный класс чаще всего строят гиперплоскость, разделяющую попарно с каждым другим классом.

Для начала необходимо сформировать обучающие выборки, по количеству разделяемых классов. Например, при наличии n классов чтобы выделить один из них необходимо составить $n-1$ обучающих выборок.

Формирование обучающей выборки производиться на серии изображений и положении прецедентов наличия или отсутствия классифицируемых данных в формате SHAPE. В атрибутивной части SHAPE файла должна содержаться информация о классе прецедента. На основе входных данных для каждого прецедента формируются наборы векторов признаков. Каждая ячейка изображения проверяется на нахождение в одном из полигональных объектов SHAPE файла. Если ячейка находится внутри полигонального объекта, то формируется вектор признаков с указанием класса прецедента и добавляется в обучающую выборку. Компонентами вектора признаков служат значения пикселей из различных наборов данных GRID. В завершении обучения мы получаем файл модели классификации. На рисунке И.1 приведена схема получения компонент вектора признаков.

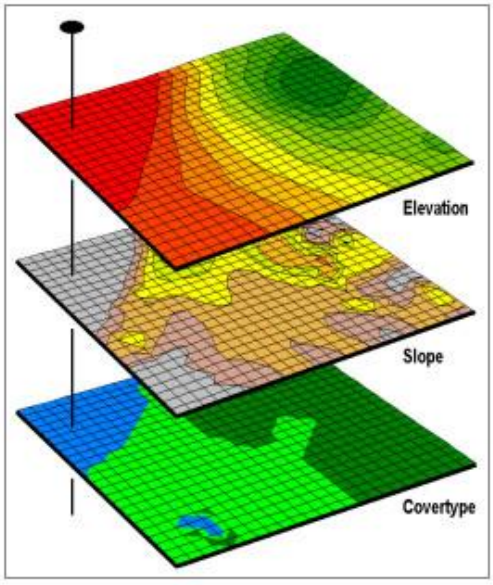


Рисунок И.1 – Получение компонент вектора признаков

Классификация одного класса производится на основе $n-1$ файлов моделей классификации. На вход классификатору подается серия изображений, на выходе получаем серию изображений. Классификация методом опорных векторов ставит в соответствие каждому классу одно из значений «1» или «–1», обозначающих отношение представителя к одному из двух классов. В данной работе вместо этих

значений «1», «–1» используется значение расстояния от разделяющей гиперплоскости до каждого представителя классов. Такой подход позволяет более информативно отобразить результаты.

Для применения метода опорных векторов в рамках геопортала разработана технология, которая состоит из следующих этапов: формирование обучающей выборки, создание классификатора, классификация и отображение результатов. Рассмотрим каждый из этапов более подробно.

Формирование обучающей выборки производится с помощью таблицы подсистемы ввода и редактирования реляционных данных геопортала. На этом этапе необходимо создать таблицу (рисунок И.2) как минимум с двумя атрибутами: положение прецедента (точечный тип геометрии) и номер класса.

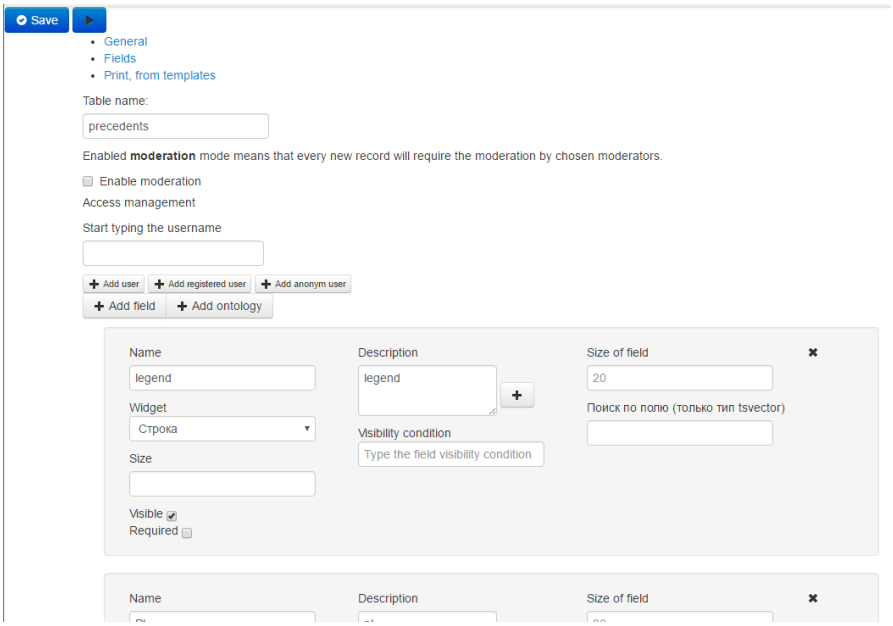


Рисунок И.2 – Создание таблицы

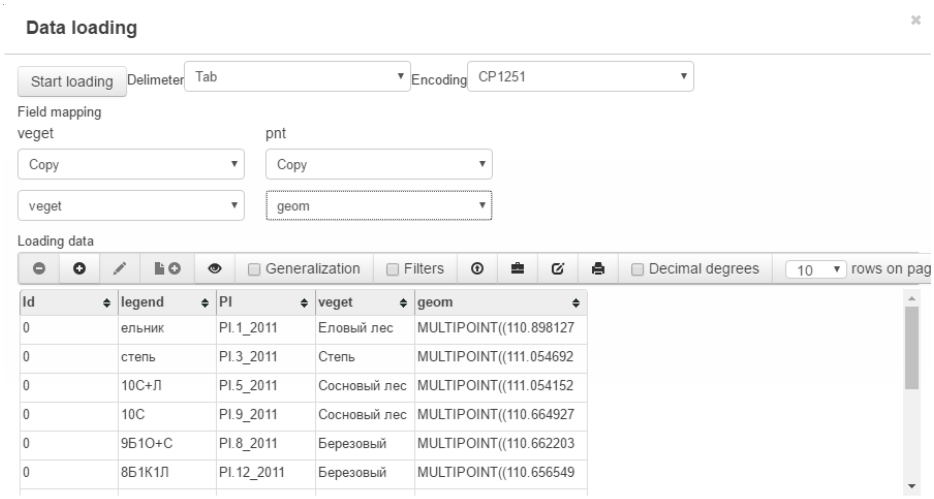


Рисунок И.3 – Загрузка данных

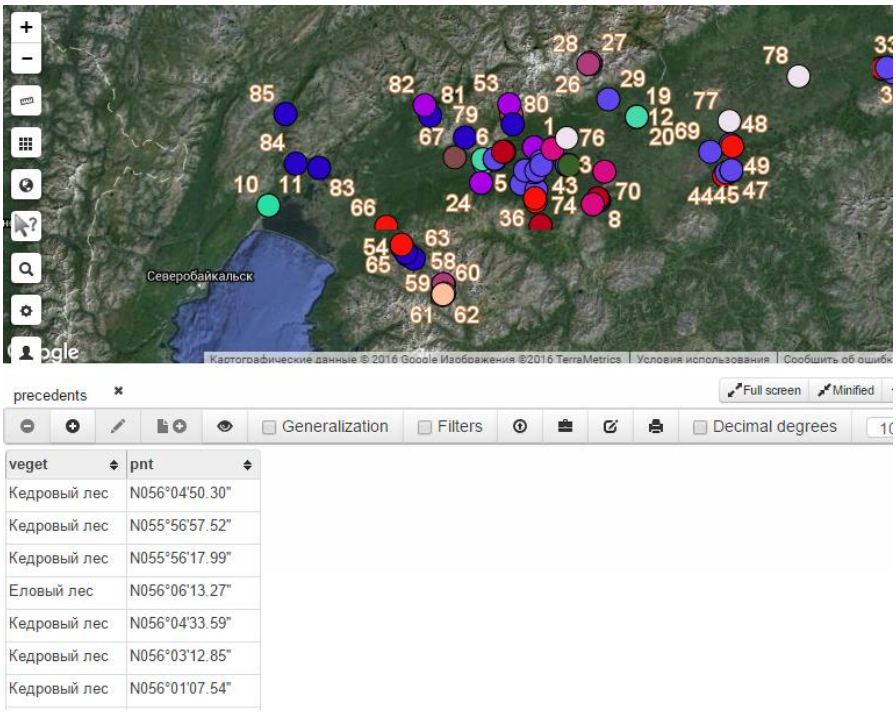


Рисунок И.3 – Прецеденты

Отображение результатов, чтобы получить результаты классификации, необходимо произвести сложение полученных значений расстояний для каждого пикселя результирующего изображения из серии. В конце мы получим изображение с одним из выделенных классов. В качестве экспериментальных данных использовались изображения северной части озера Байкал, а задача классификации стояла в определении территории с растительным покровом. Серия изображений состояла из 4 изображений со значениями NDVI, elevation, slope, aspect.

Обсудим достоинства предложенной технологии обработки данных ДЗЗ методом опорных векторов. Данная технология позволяет ускорить процедуру обработки данных ДЗЗ, за счет предоставления пользователю функций различных средств обработки данных ДЗЗ в виде набора WPS-сервисов. Для обработки данных пользователю необходимо лишь заполнить входные поля WPS-сервиса и запустить его выполнение. Использование WPS-сервиса SVM автоматизирует повторяющиеся операции, которые пользователю приходится выполнять, имея в своем арсенале совокупность ПО по обработке данных ДЗЗ. В данном случае к повторяющимся операциям относятся: манипуляции, связанные подготовкой и обработкой данных при помощи метода опорных векторов.

ПРИЛОЖЕНИЕ К. JSON ОТВЕТ REST ИНТЕРФЕЙСА ДЛЯ TORQUE PBS


```
{
  «success»:»true»,
  “task”:{
    “jid”:”14133.master”,
    “name”:”slt_p-npho_m40_3m_3”,
    “id”:14133,
    “owner”:”logolova@access1”,
    “owned”:”false”,
    “walltime”:”168:00:00”,
    “tstart”:”2022.10.19 11:10:32”,
    “runtime”:”148:56:08”,
    “state”:”Running”,
    “tpercent”:89,
    “server”:”master”,
    “queue”:”batch”,
    “errorfile”:”access1:\/home\/ebelogolova\/calc\/Gau\/AR\/M-P\/slt_p-npho_m40_3m_3.e14133”,
    “outputfile”:”access1:\/home\/ebelogolova\/calc\/Gau\/AR\/M-P\/slt_p-npho_m40_3m_3.o14133”,
    “mail”:””,
    “script”:”\/home\/ebelogolova\/calc\/Gau\/AR\/M-P\/slt_p-npho_m40_3m_3.pbs”,
    “mem”:”25935748kb”,
    “vmem”:”56238812kb”,
    “mail_begin”:”false”,
    “mail_end”:”false”,
    “mail_abort”:true
  }
}
```

ПРИЛОЖЕНИЕ Л. СВИДЕТЕЛЬСВА О РЕГИСТРАЦИИ ПРОГРАММ

РОССИЙСКАЯ ФЕДЕРАЦИЯ

СВИДЕТЕЛЬСТВО

о государственной регистрации программы для ЭВМ

№ 2014610274

Интернет-система ввода и редактирования пространственных данных «Фарамант»

Правообладатель: ***Федеральное государственное бюджетное учреждение науки Институт динамики систем и теории управления Сибирского отделения Российской академии наук (ИДСТУ СО РАН) (RU)***

Авторы: ***Фёдоров Роман Константинович (RU), Ружников Геннадий Михайлович (RU), Шумилов Александр Сергеевич (RU), Ветров Александр Анатольевич (RU), Михайлов Андрей Анатольевич (RU)***

Заявка № **2013660173**

Дата поступления **06 ноября 2013 г.**

Дата государственной регистрации в Реестре программ для ЭВМ ***09 января 2014 г.***

Руководитель Федеральной службы по интеллектуальной собственности

Б.П. Симонов

РОССИЙСКАЯ ФЕДЕРАЦИЯ

СВИДЕТЕЛЬСТВО

о государственной регистрации программы для ЭВМ

№ 2022615805

Программа расчёта спектральных индексов и построения композитного отображения для космоснимков Sentinel-2

Правообладатель: ***Федеральное государственное бюджетное учреждение наук Институт динамики систем и теории управления имени В.М. Матросова Сибирского отделения Российской академии наук (RU)***

Авторы: ***Фёдоров Роман Константинович (RU), Авраменко Юрий Владимирович (RU)***

Заявка № **2022615500**
Дата поступления **04 апреля 2022 г.**
Дата государственной регистрации
в Реестре программ для ЭВМ ***04 апреля 2022 г.***

Руководитель Федеральной службы
по интеллектуальной собственности

Ю.С. Зубов

РОССИЙСКАЯ ФЕДЕРАЦИЯ

СВИДЕТЕЛЬСТВО

о государственной регистрации программы для ЭВМ

№ 2024610919

WPS сервис классификации космоснимков Sentinel-2

Правообладатель: ***Федеральное государственное бюджетное учреждение науки Институт динамики систем и теории управления имени В.М. Матросова Сибирского отделения Российской академии наук (RU)***

Авторы: ***Фёдоров Роман Константинович (RU), Авраменко Юрий Владимирович (RU)***

Заявка № **2023689239**
Дата поступления **25 декабря 2023 г.**
Дата государственной регистрации
в Реестре программ для ЭВМ ***16 января 2024 г.***

Руководитель Федеральной службы
по интеллектуальной собственности

Ю.С. Зубов

РОССИЙСКАЯ ФЕДЕРАЦИЯ

СВИДЕТЕЛЬСТВО

о государственной регистрации программы для ЭВМ

№ 2016663724

Система планирования и выполнения композиций веб-сервисов в гетерогенной динамической среде

Правообладатель: ***Федеральное государственное бюджетное учреждение науки Институт динамики систем и теории управления имени В.М. МАТРОСОВА Сибирского отделения Российской академии наук (RU)***

Авторы: ***Фёдоров Роман Константинович (RU), Шумилов Александр Сергеевич (RU), Бычков Игорь Вячеславович (RU), Ружников Геннадий Михайлович (RU)***

Заявка № **2016660937**

Дата поступления **18 октября 2016 г.**

Дата государственной регистрации

в Реестре программ для ЭВМ ***15 декабря 2016 г.***

Руководитель Федеральной службы по интеллектуальной собственности

Г.П. Ивлиев

РОССИЙСКАЯ ФЕДЕРАЦИЯ

СВИДЕТЕЛЬСТВО

о государственной регистрации программы для ЭВМ

№ 2017615230

Web-сервис импорта данных из реляционных таблиц в CSV формате

Правообладатели: ***Федеральное государственное бюджетное учреждение науки Институт динамики систем и теории управления имени В.М. Матросова Сибирского отделения Российской академии наук (RU), Федеральное государственное бюджетное научное учреждение «Научный центр проблем здоровья семьи и репродукции человека» (RU)***

Авторы: ***Фёдоров Роман Константинович (RU), Парамонов Вячеслав Владимирович (RU), Ружников Геннадий Михайлович (RU), Данчинова Галина Анатольевна (RU), Хаснатинов Максим Анатольевич (RU), Ляпунов Александр Валерьевич (RU)***

Заявка № **2017612094**

Дата поступления **15 марта 2017 г.**

Дата государственной регистрации в Реестре программ для ЭВМ ***05 мая 2017 г.***

Руководитель Федеральной службы по интеллектуальной собственности

Г.П. Ивлиев

РОССИЙСКАЯ ФЕДЕРАЦИЯ

СВИДЕТЕЛЬСТВО

о государственной регистрации программы для ЭВМ

№ 2017617913

Среда выполнения сервисов и их сценариев

Правообладатель: ***Федеральное государственное бюджетное учреждение науки Институт динамики систем и теории управления имени В.М. Матросова Сибирского отделения Российской академии наук (RU)***

Авторы: ***Фёдоров Роман Константинович (RU), Шумилов Александр Сергеевич (RU), Бычков Игорь Вячеславович (RU), Ружников Геннадий Михайлович (RU)***

Заявка № **2017613155**

Дата поступления **10 апреля 2017 г.**

Дата государственной регистрации в Реестре программ для ЭВМ ***17 июля 2017 г.***

Руководитель Федеральной службы по интеллектуальной собственности

Г.П. Ивлиев

РОССИЙСКАЯ ФЕДЕРАЦИЯ

СВИДЕТЕЛЬСТВО

о государственной регистрации программы для ЭВМ

№ 2024611171

Программная система выполнения WPS сервисов

Правообладатель: ***Федеральное государственное бюджетное учреждение науки Институт динамики систем и теории управления имени В.М. Матросова Сибирского отделения Российской академии наук (RU)***

Автор(ы): ***Фёдоров Роман Константинович (RU)***

Заявка № **2023689223**
Дата поступления **25 декабря 2023 г.**
Дата государственной регистрации
в Реестре программ для ЭВМ ***18 января 2024 г.***

Руководитель Федеральной службы
по интеллектуальной собственности

Ю.С. Зубов

ПРИЛОЖЕНИЕ М. АКТЫ О ВНЕДРЕНИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ УЧРЕЖДЕНИЕ НАУКИ
СИБИРСКИЙ ИНСТИТУТ ФИЗИОЛОГИИ И БИОХИМИИ РАСТЕНИЙ
СИБИРСКОГО ОТДЕЛЕНИЯ РОССИЙСКОЙ АКАДЕМИИ НАУК
(СИФИБР СО РАН)
664033, Иркутск-33, а/я 317. Для телеграмм: Иркутск-33, Физиология.
Тел. 42-67-21, Факс: (3952) 51-07-54, ИНН 3812010449
E-mail: matmod@sifibr.irk.ru

от 11.06.2024г. /15335-32/2115/158

Акт о внедрении результатов научного исследования

Результаты диссертационной работы Фёдорова Романа Константиновича «Сервис-ориентированная информационно-аналитическая среда композиции сервисов обработки пространственных данных» внедрены с 2012 г. и успешно используются для сбора и анализа данных о биоразнообразии и ведения базы данных гербарной коллекции.

В частности, подсистема Геопортал обеспечивает возможность сбора данных в виде таблиц с произвольным набором полей, разграничение прав пользователей, использование различных фильтров по полям, группировку данных, подсчет обобщенных показателей, отображение данных на карте и применение различных WPS сервисов обработки и анализа данных и т.д.

Использование указанных результатов позволяет организовать распределенную работу, повысить оперативность сбора данных, расширить набор методов анализа собираемых данных.

Зав. отдела «Биоразнообразие и биологические ресурсы», в.н.с., к.б.н. А.В. Верхозина

Директор СИФИБР СО РАН, Д.б.н. В.И. Воронин

МИНОБРНАУКИ РОССИИ
Федеральное государственное
бюджетное учреждение науки
ИНСТИТУТ ГЕОГРАФИИ
им. В.Б. Сочавы
Сибирского отделения
Российской академии наук
(ИГ СО РАН)
Улан-Баторская ул., д. 1
Иркутск, 664033
Тел. (3952) 42-69-20
Факс (3952) 42-27-17
E-mail: postman@irigs.irk.ru
ОКПО 03533731, ОГРН 1023801757220
ИНН/КПП 3812011724/381201001
11.12.25 № 0335-01-279/225
На № _____ от _____

Акт

о внедрении результатов диссертационного исследования Фёдорова Романа Константиновича «Сервис-ориентированная информационно-аналитическая среда композиции сервисов обработки пространственных данных», представленного на соискание ученой степени доктора технических наук по научной специальности 2.3.5 – Математическое и программное обеспечение вычислительных систем, комплексов и компьютерных сетей

Настоящим актом подтверждаем внедрение результатов диссертационного исследования Фёдорова Романа Константиновича «Сервис-ориентированная информационно-аналитическая среда композиции сервисов обработки пространственных данных» в Федеральном государственном бюджетном учреждении науки Институт географии им. В.Б. Сочавы Сибирского отделения Российской академии наук (ИГ СО РАН) с целью создания информационной системы «Экологический атлас Байкальского региона» http://atlas.isc.irk.ru/.

Подсистемы среды использовались для сбора и обработки данных, создания и публикации тематических карт. Использование указанных результатов позволяет организовать распределенную работу, повысить оперативность сбора данных.

Директор ИГ СО РАН, д.г.н. И.Н. Владимиров

ФГАОУ ВО
«САМАРСКИЙ
НАЦИОНАЛЬНЫЙ
ИССЛЕДОВАТЕЛЬСКИЙ
УНИВЕРСИТЕТ ИМЕНИ
АКАДЕМИКА
С. П. КОРОЛЕВА»
Биологический
факультет

КАФЕДРА ЭКОЛОГИИ, БОТАНИКИ И ОХРАНЫ ПРИРОДЫ
443086, Россия,
г. Самара,
Московское шоссе, 34

SAMARA NATIONAL
RESEARCH UNIVERSITY
Biological Faculty

DEPARTMENT OF ECOLOGY, BOTANY AND NATURE PROTECTION
34, Moskovskoye shosse,
Samara, 443086, Russia

tel. (846)334-54-43
fax (846) 334 57 22
E-mail:
lkavelenova@mail.ru

По месту требования

№ _____ от "___" _____ _____г.

Акт о внедрении результатов научного исследования

Результаты диссертационной работы Фёдорова Романа Константиновича «Сервис-ориентированная информационно-аналитическая среда композиции сервисов обработки пространственных данных» внедрены в учебный процесс на кафедре экологии, ботаники и охраны природы Самарского университета с 2019 г. и успешно используются для сбора и анализа данных о видах мохообразных и лишайников, а также для организации данных по образцам коллекций лихенологического и бриологического гербария Самарского университета – SMR (L), SMR (M) и SMR (H) соответственно.

В частности, подсистема Геопортал обеспечивает следующие возможности:

– сбор данных в виде таблиц с произвольным набором полей;
– разграничение прав пользователей;
– использование различных фильтров по полям;
– группировка данных;
– подсчёт обобщённых показателей;
– отображение данных на карте;
– применение различных WPS-сервисов обработки и анализа данных.

Использование указанных результатов позволяет организовать распределённую работу, повысить оперативность сбора данных, расширить набор методов анализа собираемых данных.

Указанные результаты включены в курс «Спецпрактикум по экологии» направления подготовки 06.04.01 Биология профиль Экология и природоохранная биология кафедры экологии, ботаники и охраны природы Самарского университета.

Зав. кафедрой экологии,
ботаники и охраны природы,
д.б.н., профессор — Л. М. Кавеленова

Декан биологического факультета,
к.б.н., доцент — И. Д. Романова

МИНОБРНАУКИ РОССИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ
БЮДЖЕТНОЕ УЧРЕЖДЕНИЕ НАУКИ
ФЕДЕРАЛЬНЫЙ
ИССЛЕДОВАТЕЛЬСКИЙ ЦЕНТР
«КОЛЬСКИЙ НАУЧНЫЙ ЦЕНТР
РОССИЙСКОЙ АКАДЕМИИ НАУК»

**ПОЛЯРНО-АЛЬПИЙСКИЙ
БОТАНИЧЕСКИЙ САД-ИНСТИТУТ
им. Н. А. АВРОРИНА**

(ПАБСИ КНЦ РАН)

ул. Ботанический сад, г. Кировск,
Мурманская обл., Россия, 184256
факс (815-55) 6-16-58
тел. (815-55) 6-33-50, факс (815-55) 79-549
e-mail: pabgikscras@mail.ru; http://www.pabgi.ru

По месту требования

11.12.2023 № 186.08-01-339

Акт о внедрении результатов научного исследования

Настоящим, Полярно-альпийский ботанический сад-институт им. Н. А. Аврорина – обособленное подразделение Федерального государственного бюджетного учреждения науки Федерального исследовательского центра «Кольский научный центр Российской Академии наук» (ПАБСИ КНЦ РАН) подтверждает, что результаты диссертационной работы Фёдорова Романа Константиновича «Сервис-ориентированная информационно-аналитическая среда композиции сервисов обработки пространственных данных» внедрены в ПАБСИ КНЦ РАН с 2019 г. и успешно используются для сбора, организации и анализа данных по образцам цианопрокариот, мохообразных и лишайников, хранящихся в KPABG.

В частности подсистема Геопортал обеспечивает возможность сбора данных в виде таблиц с произвольным набором полей, разграничение прав пользователей, использование различных фильтров по полям, группировку данных, подсчет обобщенных показателей, отображение данных на карте и применение различных WPS сервисов обработки и анализа данных и т.д.

Использование указанных результатов позволяет организовать распределенную работу, повысить оперативность сбора данных, расширить набор методов анализа собираемых данных.

Директор ПАБСИ КНЦ РАН Е. А. Боровичев

МИНОБРНАУКИ РОССИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ
БЮДЖЕТНОЕ УЧРЕЖДЕНИЕ НАУКИ
ФЕДЕРАЛЬНЫЙ
ИССЛЕДОВАТЕЛЬСКИЙ ЦЕНТР
«КОЛЬСКИЙ НАУЧНЫЙ ЦЕНТР
РОССИЙСКОЙ АКАДЕМИИ НАУК»

**ИНСТИТУТ ПРОБЛЕМ
ПРОМЫШЛЕННОЙ ЭКОЛОГИИ СЕВЕРА**

(ИППЭС КНЦ РАН)

Академгородок, д. 14а, г. Апатиты,
Мурманская обл., Россия, 184209
Тел.: (815 55) 6-10-93, (815 55) 79-594,
Факс: (815 55) 7-49-64
E-mail: makarov@inep.ksc.ru

от 11.12.2023 г. № 186.05-02/512

По месту требования

Результаты диссертационной работы Фёдорова Романа Константиновича «Сервис-ориентированная информационно-аналитическая среда композиции сервисов обработки пространственных данных» внедрены в Институте проблем промышленной экологии Севера – обособленном подразделении Федерального государственного бюджетного учреждения науки Федерального исследовательского центра «Кольский научный центр Российской академии наук» (ИППЭС КНЦ РАН) с 2019 г. и успешно используются для сбора и анализа данных о видах рыб, мохообразных и лишайников, а так же для организации данных по образцам гербария ИППЭС КНЦ РАН (INEP).

В частности подсистема Геопортал обеспечивает возможность сбора данных в виде таблиц с произвольным набором полей, разграничение прав пользователей, использование различных фильтров по полям, группировку данных, подсчет обобщенных показателей, отображение данных на карте и применение различных WPS сервисов обработки и анализа данных и т.д.

Использование указанных результатов позволяет организовать распределенную работу, повысить оперативность сбора данных, расширить набор методов анализа собираемых данных.

Директор ИППЭС КНЦ РАН, д.т.н. **Д. В. Макаров**

Институт проблем промышленной экологии Севера (ИППЭС КНЦ РАН)

«УТВЕРЖДАЮ»
Директор ФГБНУ «Научный центр проблем здоровья семьи и репродукции человека»,
чл.-корр. РАН, д.м.н., проф., д.м.н.
_____Л.В. Рычкова
«____»_____2024 г.
М.П.

АКТ О ВНЕДРЕНИИ

1. ***Наименование предложения для внедрения:*** «Сервис-ориентированная информационно-аналитическая среда композиции сервисов обработки пространственных данных»
2. ***Кем предложен:*** ***ФГБУН ИДСТУ СО РАН***
   Автор: Фёдоров Р.К.
3. ***Источник информации***: Результаты диссертационной работы на соискание ученой степени доктора технических наук
4. ***Где и когда внедрено***: ФГБНУ НЦ ПЗСРЧ в течение 2020-2024 гг.
   Общее количество наблюдений: 48066
5. ***Результаты внедрения***: Разработанная автором Подсистема "Геопортал" информационно-аналитической среды обеспечивает сбор данных об активности иксодовых клещей в форме таблиц с разграничением прав пользователей, а также позволяет проводить анализ данных с использованием инструментов отображения информации на карте, фильтрации и группировки данных, подсчета обобщенных показателей, а также различных WPS-сервисов и т.д.
6. ***Эффективность внедрения:*** (информационная, профилактическая): Использование полученных результатов позволяет организовать распределенную работу подсистемы «Геопортал», повысить оперативность сбора данных, расширить набор методов анализа собираемых данных для геоинформационной системы «ГИС Мониторинг клещевых инфекций» и, как следствие, улучшить качество и эффективность принятия управленческих решений.
7. ***Заключение:*** Предлагаемая подсистема «Геопортал» проста в использовании, не требует специальных мер соблюдения техники безопасности, поэтому может быть рекомендована к внедрению в Центре инновационной медицины (ЦИМ) ФГБНУ НЦ ПЗСРЧ.

Руководитель ЦИМ ФГБНУ НЦ ПЗСРЧ,
к.м.н. Л.М. Лазарева