
ТЕХНОЛОГИИ
ИНТЕРНЕТ-
КАРТОГРАФИРОВАНИЯ

Министерство науки и высшего образования
Российской Федерации
Федеральное государственное бюджетное
образовательное учреждение высшего образования
«Пермский государственный
национальный исследовательский университет»

Р. К. Абдуллин, А. И. Пономарчук

ТЕХНОЛОГИИ ИНТЕРНЕТ-КАРТОГРАФИРОВАНИЯ

Допущено методическим советом Пермского государственного национального исследовательского университета в качестве учебного пособия для студентов, обучающихся по направлению подготовки бакалавров «Картография и геоинформатика» и направлению подготовки магистров «Математико-картографическое моделирование геосистем и комплексов»

Пермь 2020

УДК 528.942
ББК 26.17
А139

А139 Абдуллин Р. К.
Технологии интернет-картографирования : учебное пособие / Р. К. Абдуллин, А. И. Пономарчук; Пермский государственный национальный исследовательский университет. – Пермь, 2020. – 132 с.: ил

ISBN 978-5-7944-3521-4

Учебное пособие «Технологии интернет-картографирования» посвящено применению веб-ГИС технологий для создания картографических веб-приложений.

В пособии подробно описаны существующие возможности публикации пространственных данных в сети Интернет и актуальные технологии разработки клиентской части картографических веб-приложений. Кроме того, в издании рассмотрены вопросы развития веб-картографии как области ГИС, технические основы создания веб-приложений, включающие описание принципов работы сети Интернет, классической архитектуры веб-приложений, в том числе картографических, существующих технологий разработки на стороне сервера и клиента.

Пособие предназначено для студентов, обучающихся по направлению подготовки бакалавров «Картография и геоинформатика», а также по направлению подготовки магистров «Математико-картографическое моделирование геосистем и комплексов». Оно также может быть полезно и для специалистов, работающих в области наук о Земле, заинтересованных в использовании веб-ГИС технологий в исследованиях.

УДК 528.942
ББК 26.17

Печатается по решению ученого совета географического факультета Пермского государственного национального исследовательского университета

Рецензенты: факультет картографии и геоинформатики Московского государственного университета геодезии и картографии (и.о. декана факультета, доцент кафедры картографии МИИГАиК, канд. техн. наук Загребин Г. И.);
ведущий научный сотрудник кафедры картографии и геоинформатики географического факультета МГУ им. М. В. Ломоносова, канд. геогр. наук Самсонов Т. Е.

ISBN 978-5-7944-3521-4

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

В настоящее время в области картографии и геоинформационных технологий наблюдается тенденция интеграции геоинформационных систем (ГИС) и сети Интернет, что способствует развитию относительно нового направления – веб-картографии. Совершенствование технологий веб-картографирования сопровождается вовлечением в процесс обмена пространственной информацией все большего числа людей благодаря повышению доступности пространственных данных, а также оперативности их публикации и обновления.

Представляемое учебное пособие предназначено для обеспечения теоретическим материалом и использования при проведении практических занятий по дисциплинам «Web-картографирование» и «Технологии Интернет-картографирования» для студентов 4 курса бакалавриата географического факультета, обучающихся по направлению «Картография и геоинформатика», и студентов 1 курса магистратуры, обучающихся по программе «Математико-картографическое моделирование геосистем и комплексов».

Излагаемый в пособии материал должен помочь студентам в освоении базовых методов и технологий публикации пространственных данных в сети Интернет, а также разработки картографических веб-приложений.

Пособие рассчитано на студентов, получивших знания в области геоинформатики и геоинформационных технологий во время изучения предшествующих веб-картографии дисциплин.

Учебное пособие включает четыре раздела, которые позволяют получить представление о том, как развивалось веб-картографирование и что оно представляет собой на современном этапе развития:

- современное состояние веб-картографии и история ее развития;
- технические основы создания картографических веб-приложений;
- способы публикации пространственных данных в сети Интернет;
- технологии разработки клиентской части картографических веб-приложений.

В первом и втором разделах пособия в основном излагаются теоретические аспекты веб-картографии, в третьем и четвертом разделах рассматриваются как теоретические основы решения определенного класса задач, так и приводятся примеры решения конкретных задач с иллюстрациями и пошаговыми инструкциями для их выполнения. Эти инструкции и примеры составлены как для использования коммерческого программного обеспечения компании ESRI (ArcGIS Server), так и для программного обеспечения с открытым исходным кодом (GeoServer). Также пособие содержит инструкции по работе с облачными платформами для публикации в сети Интернет пространственных данных – ArcGIS Online и NextGIS.com.

1. МИГРАЦИЯ ГИС В СЕТЬ ИНТЕРНЕТ

1.1. Понятие веб-картографии, ее основные задачи

Как известно, первые геоинформационные системы (ГИС) возникли в конце 50-х – начале 60-х гг. XX в. Их появление было вызвано созданием вычислительной техники. Однако широко они стали использоваться только начиная с 1980-х гг., когда на рынки вышли первые коммерческие программные продукты (ARC/INFO, Intergraph и др.) [2]. Изначально ГИС развивались как настольные программные продукты, но с активным развитием компьютерных сетей (в частности, сети Интернет) в 1990-е гг., появились и сетевые версии ГИС. Именно с их появлением и возникло новое направление в области геоинформационных технологий под названием веб-картография. Наряду с термином «веб-картография» в научных работах активно используется понятие «веб-картографирование» (web mapping), причем последнее гораздо чаще встречается в англоязычных источниках.

Веб-картография – это область компьютерных технологий, связанная с доставкой пространственных данных конечному пользователю с помощью вычислительных сетей [6]. Приставка «веб» употребляется потому, что в качестве среды могут использоваться любые вычислительные сети, а не только Интернет, хотя он занимает главенствующее положение. В этом пособии упор сделан на возможностях публикации и использования пространственных данных в сети Интернет. Веб-картография является одним из направлений геоинформационных технологий в целом, можно сказать, что она возникла путем интеграции ГИС и Интернета (Веба), превратив ГИС в веб-приложение. В результате этого появились термины «Интернет-ГИС» и «Веб-ГИС»; в англоязычных источниках используется термин «Web-GIS», который появился не так давно, но гораздо чаще встречается термин «web mapping services» (картографические веб-сервисы).

Основными задачами веб-картографии являются:

- визуализация (пространственное представление) данных и облегчение работы с ними путем формирования запросов к ним по атрибутам, прокладки маршрутов и использования других функций, основанных на местоположениях объектов (LBS – Location Based Services). Эти операции с данными, например, дают пользователям ответы на вопросы: «Что и где находится?», «Как быстрее добраться?» и др.;
- распространение пространственных данных. С помощью картографических веб-приложений можно организовать загрузку данных и обмен ими, а также совместно использовать их. Кроме

того, распространение пространственных данных является целью создания геопорталов;

- формирование массива пространственных данных. Пользователи могут применять технологии веб-ГИС для создания массивов географических данных. Например, проект OpenStreetMap (OSM) создан пользователями Интернета, которые собирают данные путем дешифрирования космических снимков и с помощью портативных GPS-приемников;
- пространственный анализ данных. Отметим, что картографические веб-приложения вышли за пределы простой визуализации данных и в них могут быть доступны различные аналитические функции, например: картометрические функции, поиск оптимального маршрута, статистический анализ, построение профиля и др.

1.2. Предпосылки возникновения веб-картографии

Веб-картография как отдельное направление, появилась в 1993 г. на стыке геоинформатики и сетевых компьютерных технологий. Ее возникновение напрямую связано с качественными преобразованиями, которые произошли в 1990-е гг. в компьютерных сетях и в частности в сети Интернет. Среди них – появление протокола передачи гипертекста HTTP (Hypertext Transfer Protocol), языка разметки гипертекста HTML (Hypertext Markup Language) и стандартного указателя ресурса в сети URL (Uniform Resource Locator), создание веб-сервера и веб-браузера. Благодаря этим нововведениям стал возможным переход ГИС на новый уровень развития. Поэтому прежде чем рассматривать происхождение и эволюцию веб-картографии, стоит остановиться на истории развития сети Интернет и ГИС.

1.2.1. Развитие сети Интернет и Всемирной паутины

В целом появление сети Интернет считается одной из крупных вех развития человека, сопоставимой с изобретением печатного станка [12].

Возникновение Интернета связывают с необходимостью обмена информацией в период холодной войны в случае отказа некоторых узлов связи или их уничтожения в результате ядерного удара. По этой причине в 1960-е гг. Агентство перспективных исследований Министерства обороны США (ARPA) запустило проект по созданию сети удаленных друг от друга компьютеров. Таким образом, в 1969 г. в рамках проекта была создана сеть под названием ARPANET, которая считается прародительницей сети Интернет. ARPANET объединила главные вычислительные системы четырех университетов на западе США: Стэнфордского, Калифорнийского в Санта-Барбаре, Калифорнийского в Лос-Анджелесе и Университета штата Юта, ознаменовав тем самым рождение Интернета. Постепенно к созданной сети добавлялись новые компьютеры: в первую очередь устройства государственных органи-

заций, университетов и научно-исследовательских институтов. Это привело к увеличению числа узлов сети с 4 до 57 к 1975 г. Сети этих организаций были соединены между собой, образовав тем самым сеть сетей, или Интернет.

Отметим также, что в 1976 г. под руководством американского ученого Винтона Серфа был разработан стек протоколов передачи данных в сети под названием TCP/IP, который и сейчас является стандартом для передачи данных в Интернете. Протокол – это совокупность правил, устанавливающих формат и процедуры обмена информацией между несколькими устройствами: двумя или более.

К концу 1989 г. к Интернету уже было подключено более 100 тыс. компьютеров по всему миру. Однако он приобрел популярность только в 1990-х гг., что объясняется коренным изменением концепции Интернета и изобретением ряда протоколов работы в сети.

До 1990-х гг. спектр служб Интернета ограничивался в основном электронной почтой E-mail, передачей новостей Usenet, передачей файлов протоколом FTP, а также протоколом удаленного управления компьютерами Telnet. Таким образом, Интернет был сложен в использовании, его содержимое значительно уступало по разнообразию тому, что имеется в нем сегодня. Поэтому основными пользователями Интернета были специалисты, работающие в научно-исследовательских институтах и государственных организациях.

Однако в 1990 г. Тим Бернерс-Ли, научный сотрудник Европейской организации по ядерным исследованиям в Женеве (CERN), качественно изменил концепцию использования Интернета. Задавшись целью найти простой способ совместного использования документов и обмена ими со своими коллегами, он изобрел протокол передачи гипертекста HTTP, язык разметки гипертекста HTML и стандартный указатель ресурса в сети URL. Также Бернерс-Ли разработал первый в мире веб-сервер и веб-браузер, назвав свое изобретение World Wide Web (WWW), или – на русскоязычный лад – Всемирной паутиной, или Вебом [12]. Веб называют лицом Интернета.

Несмотря на то что термины «Интернет» и «Всемирная паутина» (Веб) для многих являются синонимами, они означают разные вещи. Интернет – это огромная сеть сетей, которая соединяет миллионы компьютеров во всем мире. Компьютеры, подключенные к Интернету, могут общаться друг с другом с помощью ряда протоколов, например: HTTP (протокол передачи гипертекста), SMTP (протокол передачи электронной почты), FTP (протокол передачи файлов) и др. В то же время Веб – это система связанных гипертекстовых документов и программ, к которым можно получить доступ в Интернете, в основном с помощью протокола HTTP. Большинство пользователей привлекает в Интернете информационное содержание Веба и разнообразная деятельность, которой можно в нем заниматься.

Веб по-прежнему стремительно развивается и расширяется, переходя из проводных сетей в беспроводные благодаря растущей популярности мобильного доступа, а также развитию технологий беспроводной связи и сотовых сетей нового поколения[12].

Также заметим, что с начала 2000-х гг. говорят о новой концепции Веба – Веб 2.0. Авторство этого термина приписывается Тиму О'Рейли, американскому издателю, т. к. в 2005 г. он выпустил статью, где развивал названную концепцию. Под Веб 2.0 он понимает методику проектирования

систем, которые благодаря учёту сетевых взаимодействий становятся лучше при увеличении количества пользующихся ими людей. Особенность Веб 2.0 заключается в привлечении пользователей к наполнению и многократной выверке информационного материала [35].

Основными принципами Веб 2.0 являются [35]:

- использование коллективного разума пользователей Веба для формирования информации и контента, развития и улучшения веб-проектов и сервисов, создания сообществ;
- использование Веба как платформы для разработки программного обеспечения и облачных вычислений. В данном случае большинство платформ предоставляют свои ресурсы в виде веб-сервисов и веб-служб – программ, доступ к которым осуществляется через протокол HTTP. Использование Веба как платформы выражено в появлении технологии SaaS (Software as a Service) – программное обеспечение как сервис, где возможности программного обеспечения поставляются как веб-службы или веб-приложения;
- использование AJAX (Asynchronous JavaScript and XML) – подхода к построению пользовательских интерфейсов веб-приложений, при котором при обновлении данных веб-страница не перезагружается, а асинхронно загружает нужные пользователю данные, т. е. происходит обмен данными браузера и веб-сервера в фоновом режиме. Использование AJAX стало наиболее популярно после того, как компания Google начала активно использовать его при создании своих сайтов, таких как Gmail и Google Maps;
- создание мэшапов (от англ. mash-up) – веб-приложений, объединяющих данные из нескольких источников в один интегрированный сервис. Например, при объединении картографических данных Google Maps с данными о недвижимости с другого сайта получается новый уникальный веб-сервис, изначально не предлагаемый ни одним из источников данных;
- высокая эргономичность и дружественность интерфейсов веб-приложений;
- использование ключевых слов и тегов.

Таким образом, современный Веб построен на основе описанной концепции Веб 2.0, которая нашла свое отражение и в распространенных картографических веб-приложениях, например: Google Maps, Google Earth, «Яндекс.Карты» и мн. др.

1.2.2. Появление и развитие ГИС

Согласно классическому определению, геоинформационная система – это информационная система, обеспечивающая сбор, хранение, обработку, доступ, отображение и распространение пространственно-координиро-

ванных данных [1]. Появление ГИС относят к 60-м гг. XX в. К предпосылкам их возникновения относят активное развитие компьютерных технологий и совершенствование средств периферии, накопление обширного объема географической информации (аэроснимки, статистика и др.), потребность в упорядочивании имеющейся информации в базах данных, необходимость обеспечения доступности этих материалов для широкого круга пользователей, необходимость принятия оперативных решений в разных областях и др. [2, 9] В истории развития ГИС традиционно выделяется четыре периода [11].

Начальный период (или «пионерный период») становления ГИС относится к 60-м гг. XX в. На этом этапе развития ГИС использовались в основном для решения узкоспециализированных задач, к которым можно отнести инвентаризацию земельных и других ресурсов, информационно-справочные задачи, обработку статистической информации и др., и в основном разрабатывались научными и ведомственными организациями. Первые геоинформационные системы появились в Швеции в середине 60-х гг. [9] Работы шведской школы геоинформатики концентрировались вокруг ГИС земельно-учетной специализации, в частности Шведского земельного банка данных, предназначенного для автоматизации учета земельных участков (землевладений) и недвижимости. Однако наиболее ярким примером этого периода было создание в 1963–1971 гг. Канадской ГИС (КГИС) под руководством Р. Томлинсона, которая стала одним из примеров крупной универсальной региональной ГИС национального уровня. Эта система создавалась для анализа данных инвентаризации земель Канады в области рационализации землепользования [11]. Вторая половина 60-х гг. XX в. знаменательна также работами Гарвардской лаборатории машинной графики и пространственного анализа. Здесь было создано программное обеспечение, которое стало классическим в сфере автоматизированного картографирования того времени.

Следующий этап развития ГИС относится к 70-м гг. XX в., и его часто называют государственным периодом. Особенностью создаваемых в это время ГИС была их закрытость, поскольку они в большинстве случаев были направлены на решение государственных и ведомственных задач. На этом этапе также стали формироваться первые фундаментальные принципы ГИС: были сформулированы понятия пространственного объекта и его описания позиционными и атрибутивными характеристиками, разработаны технологии цифрования карт как основного источника информации в ГИС, операции манипулирования пространственными данными, в основе которых лежали алгоритмы вычислительной геометрии. Для этого периода характерно проявление взаимодействия методов и средств ГИС и автоматизированного картографирования. Особенно быстрым был прогресс геоинформационных и картографических технологий в США, проявившийся в деятельности Геологической службы США и Бюро переписей. В конце 70-х гг. под эгидой Международного географического союза была выполнена инвентаризация прикладных ГИС и программных средств, обеспечивающих работу с пространственными данными, средств компьютерной графики и картографии [9].

Третий этап развития ГИС начался в 80-е гг. XX в., когда получили широкое распространение дешевые персональные компьютеры. Этот этап именуют эпохой зрелости и коммерческо-профессиональным периодом. Он характеризуется появлением комплексных программных решений для

создания ГИС. Разработка коммерческих программных средств ГИС, связанная в немалой степени с возможностями вычислительных средств и персональных ЭВМ, существенно меняет всю геоинформационную индустрию, появление которой связывается именно с этим периодом [11].

Создание ГИС стало основываться не на уникальных программных и аппаратных средствах собственной разработки, а на адаптации функциональных возможностей довольно универсальных программных продуктов применительно к анализируемым проблемам. Именно в это время массово создавались ГИС на платформе персональных компьютеров. С этим периодом также связывают появление специализированного программного обеспечения для разработки ГИС. Так, в Институте изучения систем окружающей среды (ESRI, Inc., США) был создан программный продукт ARC/INFO (на его основе в 2000-е гг. появилось программное обеспечение ArcView и ArcGIS), важнейшей особенностью которого стала независимость от платформ и операционных систем. Компанией Intergraph Corp. был разработан продукт ERDAS. Производители ГИС-пакетов предоставляли свои программные продукты бесплатно или с большими скидками целому ряду научных и образовательных организаций, что способствовало скорейшему освоению и использованию ГИС-технологий, позволило быстрее увидеть и оценить их перспективы.

В 80-е гг. существенно расширяется круг решаемых задач: геоинформационные технологии проникают во все новые сферы науки, производственной деятельности и в образование. Осваиваются принципиально новые источники массовых данных для ГИС: данные дистанционного зондирования, данные глобальных систем позиционирования. Цифровые методы обработки изображений интегрируются с системами автоматизированной картографии и ГИС, создавая предпосылки для единой программной среды [11].

Четвертый период развития ГИС, именуемый массовым периодом, начался в 90-е гг. XX в. и продолжается до сих пор. Для этого этапа развития характерно использование ГИС не только коммерческими организациями и органами государственной власти всех уровней, как это было ранее, но и обычными индивидуальными пользователями. С этим напрямую связано активное применение ГИС при решении практических задач во многих предметных областях. При этом все больше проектов стало выполняться с использованием не персональных компьютеров, а серверных решений и компьютерных сетей. Также названный период характеризуется появлением интеллектуальных систем и мультимедиатехнологий, активным ведением работ в области моделирования, в области создания инфраструктур пространственных данных, применением нейронных сетей для классификаций и прогнозирования. На четвертом этапе активно развиваются направления, связанные с созданием мобильных ГИС, интеллектуализацией систем, созданием имитационных моделей, разработкой сценариев развития в ГИС, а также интеграцией самих информационных систем с новыми технологиями, использующими пространственные данные [2].

Еще одна важная особенность современного этапа развития ГИС – появление веб-картографии, что вызвано массовым распространением ГИС и ростом популярности сети Интернет в это время.

1.3. Происхождение и эволюция веб-картографии

Несмотря на то что для большинства пользователей сети Интернет появление веб-картографии ассоциируется с выходом на рынок продуктов компании Google в 2005 г., в реальности картографические веб-приложения появились значительно раньше. Возникновение веб-картографии и первой веб-ГИС относят к 1993 г., когда Научно-исследовательский центр в Пало-Альто (PARC) корпорации Xerox впервые запустил веб-сервис – визуализатор карт Xerox PARC Map Viewer (рис 1.1). Веб-сайт сервиса предоставлял простейшие функции работы с картой, включающие в себя изменение масштаба, выбор слоев и преобразование проекции. Визуализатор карт позволял пользователям в интерактивном режиме отправлять HTTP-запросы из веб-браузера к серверу, который выполнял картографические операции, создавал новую карту и отправлял ее браузеру, сделавшему запрос. Пользователь получал фрагменты карт на экране в формате GIF. Так впервые был применен подход к работе с геоинформацией в веб-браузере, продемонстрировавший возможность использования ГИС любым веб-пользователем без ее локальной установки, – преимущество, которого не имеют традиционные настольные ГИС. Поэтому именно названное приложение с его функциональной концепцией стало родоначальником большинства более поздних версий веб-ГИС.

Поняв преимущества этой идеи, геоинформационное сообщество стало быстро переходить на использование функций ГИС в веб-браузерах. В результате появились многочисленные приложения веб-ГИС. Таким образом, наступил период интенсивного развития веб-картографии.

Так, в 1994 г. была выпущена первая онлайн-версия Национального атласа Канады, который представлял собой интерактивный картографический веб-сайт. Функциональные возможности атласа позволяли выбирать для отображения различные слои данных, такие как дороги, реки, административные границы и природоохранные зоны и др. При этом пользователю была доступна возможность выбора подходящего символа для отображения данных [12].

В 1995 г. Калифорнийский университет в Санта-Барбаре при участии ряда других организаций создал Александрийскую цифровую библиотеку, а Геологическая служба США (USGS) – веб-портал Национальной клиринговой палаты геопространственных данных. Эти два веб-приложения позволили пользователям искать карты и спутниковые снимки по ключевым словам и в границах задаваемого участка. Названные приложения направлены на совместное использование пространственных данных большим числом пользователей и являются одними из первых примеров геопорталов [12].

В 1995 г. Бюро переписи населения США выпустило свой картографический веб-сервис с данными TIGER, который позволял искать и отображать на карте внушительный объем демографической информации штатов, округов и городов из национальной базы данных переписи населения [12].

Также в 1995 г. в Калифорнийском университете в Беркли было разработано программное обеспечение GRASSLinks. GRASS (Geographic Resources Analysis Support System – система поддержки анализа географических ресурсов) изначально была настольным ГИС-инструментом, и ее функции не

были доступны в Вебе. GRASSLinks представлял собой интерфейс взаимодействия между веб-сервером и GRASS, позволяющий пользователям выбирать в веб-браузере слои данных и направлять запросы веб-серверу. Сервер направлял запрос в GRASS, где производились операции буферизации, наложения, классификации и картографирования, после чего результаты возвращались пользователю. GRASSLinks стал одним из первых примеров, показавших, что веб-ГИС может выходить за рамки картографии и поиска, выполняя сложный пространственный анализ [12].

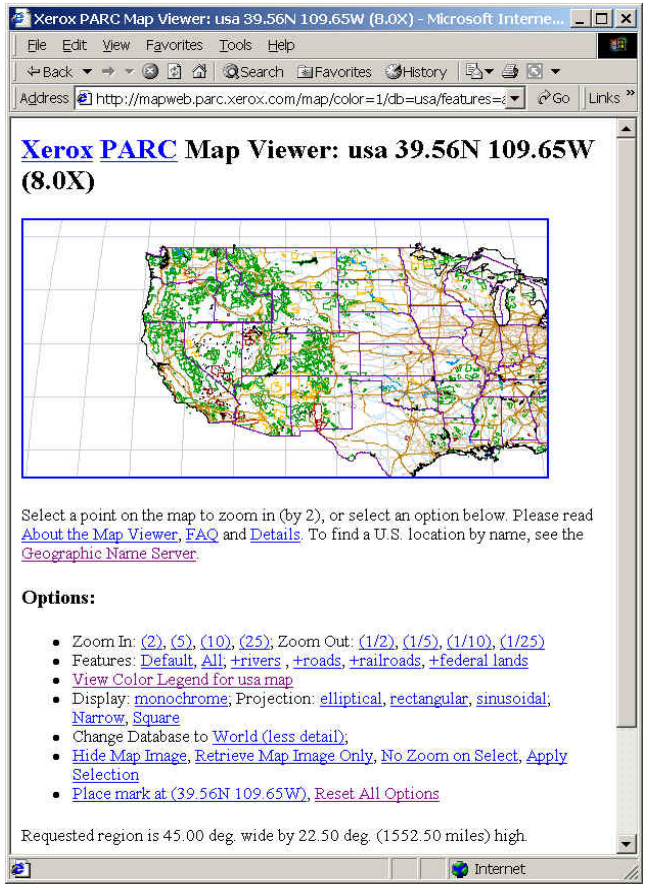

Рис. 1.1. Интерфейс картографического веб-сервиса Xerox PARC Map Viewer

В 1996 г. компания MapQuest (США) запустила свое картографическое веб-приложение. Оно позволило пользователям просматривать карты, искать поблизости компании, находить оптимальный маршрут до желаемого пункта назначения и планировать поездки. Это приложение стало одним из первых потребительских картографических веб-сайтов, который до сих пор популярен [12].

Таким образом, можно отметить, что на ранних этапах становления веб-картографии (до 1998 г.) отличительными особенностями большинства картографических веб-приложений были их локальность (малый географический охват), узкая тематическая направленность и ограниченная функциональность, что значительно сужало круг их потенциальных пользователей.

Один из первых решительных шагов в направлении популяризации веб-ГИС был сделан в 1998 г. в Великобритании, когда по адресу www.streetmap.co.uk был запущен сервис, который, в отличие от своих предшественников, не был ориентирован на визуализацию локального участка земной поверхности и насыщение ее узкотематической информацией. Создатели данного веб-ресурса пошли иным путем – они опубликовали простейшую топографическую информацию, но покрыли ей всю территорию Великобритании. Именно благодаря такому подходу этот сервис стал очень популярным. Пользователи сервиса без особого труда могли определить местоположение торгового центра, дома и любого другого объекта, зная, например, его почтовый индекс, а затем отправить готовую схему проезда на печать [6].

Также 1998 г. был ознаменован появлением специального программного обеспечения, позволяющего любому пользователю сети создавать собственные картографические веб-сервисы, – MapServer [6]. Примерно в это же время четкое понимание перспектив веб-ГИС привело к тому, что крупные компании – производители геоинформационного программного обеспечения (ESRI, Intergraph) приняли решение о разработке собственных специальных коммерческих приложений для создания веб-ГИС. Так, компанией ESRI был выпущен программный продукт ArcIMS.

К важным событиям в развитии веб-картографии можно отнести появление в 2000 г. первой версии стандарта картографического сервиса WMS (Web Map Service), который обеспечивает передачу географически привязанных растровых изображений, генерируемых картографическим сервером.

Следующим переломным моментом в развитии веб-картографии стал 2005 г., когда компания Google запустила два глобальных картографических веб-сервиса: Google Maps и Google Earth. Стоит отметить, что ни один из запущенных ранее картографических веб-сервисов не мог отличиться глобальным охватом. Кроме того, в организации сервисов был использован принципиально новый подход вместо, используемого ранее, – классического. При классическом подходе пользователь посылает запрос на сервер, ждет обработки и получает обратно сгенерированное в реальном режиме времени изображение, а при подходе, использованном компанией Google, данные были заранее подготовлены для немедленной передачи по сети, что в сочетании с технологиями AJAX позволило добиться быстрой работы с картами и бесшовного покрытия данными при навигации.

После появления продуктов компании Google были созданы и другие глобальные картографические веб-сервисы (Microsoft Bing, Yahoo, «Яндекс. Карты», OpenStreetMap и др.), благодаря которым становятся общедоступными космические снимки высокого разрешения, навигационные сервисы, сведения о заторах на дорогах и т. п. В целом создание глобальных картографических веб-сервисов ознаменовало переход к современному этапу развития веб-картографии.

Современный период также характеризуется появлением программного обеспечения и средств разработки (интерфейсы программирования

веб-приложений (API), специальные библиотеки, облачные инфраструктуры и др.), необходимых для создания картографических веб-сервисов. Благодаря этому современные картографические веб-приложения по своим функциональным возможностям постепенно приближаются к настольным ГИС, в то время как первые картографические веб-сервисы ограничивались визуализацией данных. Важным событием этого этапа развития веб-картографии является окончательное формирование набора базовых стандартов картографических веб-сервисов (или веб-служб) Открытым геопространственным консорциумом – Open Geospatial Consortium (OGC). OGC – международная некоммерческая организация, ведущая деятельность по разработке стандартов в сфере пространственных данных и сервисов.

В целом современное время характеризуются колоссальным интересом к веб-картографии и ее возможностям, а также значительным ростом числа сервисов, использующих картографические веб-технологии, и расширением их пользовательской аудитории. Немаловажную роль в этом сыграло и то, что современные картографические веб-сервисы сочетают в себе принципы концепции Веб 2.0.

Также к важным тенденциям, наблюдаемым в последние годы в области веб-картографии, относятся: появление большого числа бесплатных проектов реализующих концепцию предоставления предобработанных данных (проекты, предоставляющие доступ к открытым данным), увеличение возможностей персонификации сервисов, возможность совмещения собственных данных с существующими сервисами, глобальность сервисов, все большая интеграция таких служб в повседневную жизнь, развитие краудсорсинга.

1.4. Основные преимущества и недостатки использования картографических веб-приложений

Картографические веб-приложения и веб-ГИС имеют ряд преимуществ по сравнению с их настольными аналогами. Это в первую очередь связано со снятием такого существенного ограничения, как отсутствие доступности удаленных от пользователя геоинформационных ресурсов. К таким преимуществам можно отнести [12]:

- **Глобальный охват.** Разработчик может предоставить доступ к картографическому веб-приложению любому пользователю, находящемуся в любой точке мира, где есть подключение к сети Интернет. Пользователь может работать с веб-приложением, используя домашний компьютер или сотовый телефон.

- **Большое число пользователей.** Традиционная настольная ГИС, как правило, используется в каждый момент времени только одним пользователем, в то время как веб-ГИС может использоваться одновременно большим числом пользователей. Но это может требовать от веб-ГИС более высокого уровня производительности и масштабируемости.

- **Кроссплатформенные возможности.** Большинство клиентов картографических веб-приложений – это веб-браузеры (например, самые популярные: Internet Explorer, Google Chrome, Mozilla Firefox, Opera, Safari), которые есть практически на каждом персональном компьютере или мобильном устройстве, работающем под управлением разных операционных систем (например, Microsoft Windows, Linux, macOS, Android, iOS и др.). Таким образом, и веб-ГИС может функционировать на устройствах с различными операционными системами, чего не скажешь о любой настольной ГИС.

- **Низкая стоимость в среднем на клиента.** Подавляющая часть интернет-содержимого предоставляется конечным пользователям бесплатно. Это справедливо и для веб-ГИС. Как правило, пользователю не нужно покупать программное обеспечение или платить за использование веб-ГИС. Организации, в которых многим пользователям необходим доступ к ГИС, также могут сократить расходы, используя веб-ГИС. Вместо покупки и установки настольной ГИС для каждого пользователя организация может установить всего одну веб-ГИС, и эта единственная система будет использоваться одновременно множеством пользователей. При этом доступ сотрудников к веб-ГИС можно обеспечить как в офисе, так и дома и в полевых условиях (во время выезда, в командировке).

- **Снижение требований к рабочему устройству пользователя.** Современные традиционные настольные ГИС для своей работы предъявляют существенные требования к аппаратному обеспечению устройств: объему оперативной памяти, свободного места на жестком диске, производительности видеокарты и др. В то же время при использовании веб-ГИС, особенно с тонким клиентом, основная нагрузка по обработке данных ложится на сервер и требования к устройству пользователя значительно снижаются.

- **Простота использования для конечных пользователей**. Настольные ГИС в большей степени предназначены для специалистов, прошедших соответствующее обучение. В свою очередь, картографические веб-приложения, представляющие собой обычную веб-страницу, предназначены для широкой аудитории. Разработчики веб-ГИС стремятся сделать приложение максимально простым, интуитивно понятным и удобным. В результате веб-ГИС оказываются намного проще в использовании, чем их настольные аналоги.

- **Единовременное обновление.** Чтобы обновить настольную ГИС до новой версии, пакет обновлений необходимо установить на каждый компьютер. В случае веб-ГИС одно обновление охватывает всех клиентов, что намного упрощает задачу администрирования. Поскольку программа и данные обновляются на сервере, большинство клиентов веб-ГИС получают возможность использования обновленной версии веб-приложения автоматически. Это обеспечивает легкость обслуживания веб-ГИС

и значительно улучшает возможности ее актуализации, благодаря чему она подходит для доставки информации в режиме реального времени.

- **Разнообразие применений.** В отличие от настольной ГИС, использование которой ограничено некоторым числом ГИС-специалистов, веб-ГИС может использоваться любым сотрудником организации и вообще любым пользователем. Такая широкая аудитория имеет разнообразные потребности, и веб-ГИС используются для самых разных целей. Например, для информирования о расположении точек доступа Wi-Fi, отображения на карте мест событий из новостей. Также важным способом применения картографических веб-сервисов является краудсорсинг и создание ГИС с общественным участием (ГИСОУ), целью которых выступает привлечение широкой публики к производству пространственных данных и участие в принятии решений при помощи ГИС. Среди известных работающих краудсорсинговых проектов можно назвать OpenStreetMap (OSM) – openstreetmap.org, участники которого формируют веб-карту на основе дешифрирования космических снимков и глобальных систем позиционирования (GPS/ГЛОНАСС). Похожий проект – «Народная карта Яндекс», n.maps.yandex.ru. Интеграция краудсорсинга и картографических веб-сервисов может использоваться в разных областях, например: при управлении городской территории, когда жители города могут сообщать о проблеме и ее местоположении, при сборе информации о расположении несанкционированных свалок, перекрытии дорог и др.

Перечисленные преимущества картографических веб-сервисов порождают и некоторые проблемы, стоящие перед их разработчиками. Например, легкость использования веб-ГИС не только способствует участию в них широкого круга пользователей, но и напоминает проектировщикам веб-ГИС о необходимости учитывать при ее разработке неподготовленность пользователей Интернета и должностных лиц, не знакомых с ГИС-технологиями. Многочисленность пользователей обеспечивает более широкое внедрение ГИС, однако в то же время требует от веб-ГИС масштабируемости, т. е. способности сохранять хорошую производительность при росте числа пользователей.

Также к недостаткам картографических веб-приложений можно отнести ограниченность функциональных возможностей. Она связана с тем, что содержание и набор функций веб-ГИС в первую очередь формируются разработчиками веб-приложения.

Еще одним недостатком картографических веб-приложений, относящимся не к их пользователям, а к разработчикам, является сложность их создания. Для создания собственной веб-ГИС необходимо иметь дорогостоящее оборудование, а также владеть навыками программирования и разработки веб-страниц. Но эта проблема может быть частично решена использованием облачных инфраструктур пространственных данных (ArcGIS Online, NextGIS и др.), о которых речь пойдет в гл. 3.

2. ТЕХНИЧЕСКИЕ ОСНОВЫ СОЗДАНИЯ ВЕБ-ПРИЛОЖЕНИЙ

2.1. Принципы работы сети Интернет и Всемирной паутины

Интернет – это всемирная система объединенных компьютерных сетей, или сеть сетей. Под компьютерной сетью понимается система соединения компьютеров и другого вычислительного оборудования (серверы, маршрутизаторы и др.) при помощи каналов связи. Компьютерные сети различаются по нескольким признакам: по территориальному охвату, архитектуре, топологии, скоростям передачи данных, назначению и др. Так, по территориальному охвату выделяют локальные сети, или Local Area Network (LAN), и глобальные (распределенные), или Wide Area Network (WAN). Локальные сети объединяют компьютеры, находящиеся в одном или нескольких близко расположенных зданиях, и обычно используются в рамках одной организации (корпорации, учреждения). Глобальные сети предназначены для объединения отдельных компьютеров и локальных сетей, расположенных на значительном удалении (сотни и тысячи километров) друг от друга. Глобальные сети объединяют пользователей, находящихся по всему миру, используя при этом самые разнообразные каналы связи.

Под архитектурой компьютерной сети понимают концепцию, определяющую взаимосвязь, структуру и функции взаимодействия рабочих станций в сети. Обычно при создании компьютерных сетей используют три вида архитектуры: клиент – сервер, одноранговую (пиринговую, децентрализованную) и терминал – главный компьютер (рис. 2.1). При использовании клиент-серверной архитектуры задания или сетевая нагрузка распределены между поставщиками услуг, называемыми серверами, и заказчиками услуг, называемыми клиентами. Одноранговая архитектура основывается на равноправии участников сети, и часто в такой сети отсутствуют выделенные серверы, а каждый узел является клиентом и выполняет функции сервера. Архитектура «терминал – главный компьютер» – это концепция информационной сети, в которой вся обработка данных осуществляется одним или группой главных компьютеров. Рассматриваемая архитектура предполагает две функциональные роли оборудования: главный компьютер, где осуществляется управление сетью, хранение и обработка данных и терминалы, предназначенные для передачи главному компьютеру команд на организацию сеансов и выполнение заданий, ввода данных для выполнения заданий и получения результатов. Главный компьютер через мультиплексоры передачи данных (МПД) взаимодействует с терминалами.

Топология компьютерной сети – это описание физических соединений (схема соединения) компьютеров в сети каналами связи. Топология бывает полносвязной, линейной, древовидной (иерархической), кольцевой, шиной, смешанной и др.

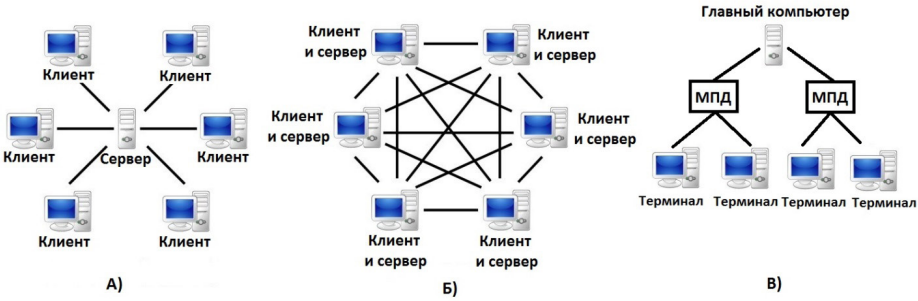

Рис. 2.1. Виды архитектуры компьютерных сетей:
А) «клиент – сервер»; Б) одноранговая; В) «терминал – главный компьютер»

Сеть Интернет относят к глобальной сети, объединяющей локальные сети разной архитектуры и топологии. В отличие от локальных сетей глобальная сеть Интернет включает подсеть связи (или территориальную сеть связи, систему передачи информации), к которой подключаются локальные и региональные сети, отдельные компьютеры и терминалы (средства ввода и отображения информации).

Подсеть связи состоит из каналов передачи информации и коммуникационных узлов (маршрутизаторов). Каналы связи физически строятся на основе витых пар коаксиальных и оптических кабелей или эфира. Коммуникационные узлы предназначены для передачи данных по сети, выбора оптимального маршрута передачи информации по каналам связи, коммутации пакетов информации и реализации ряда других функций с помощью компьютеров и соответствующего программного обеспечения, имеющихся в коммуникационном узле.

Компьютеры, за которыми работают пользователи-клиенты, называются рабочими станциями, а компьютеры, являющиеся источниками ресурсов сети, предоставляемых пользователям, называются серверами. Таким образом, работа пользователей с Интернетом построена главным образом на основе архитектуры клиент – сервер. В Интернете нет единого пункта регистрации, поэтому рабочие станции пользователей подключаются к сети через поставщиков услуг – провайдеров.

Информация в сети Интернет передается блоками данных (пакетами). Каждый пакет снабжен адресами посылающего и принимающего компьютеров. На принимающем компьютере пакеты собираются в целое сообщение. Для передачи пакетов процедуры используются протоколы передачи данных. Протокол – это совокупность правил, устанавливающих формат и процедуры обмена информацией между двумя или несколькими устройствами.

Работа сети Интернет основана на использовании семейств коммуникационных протоколов TCP/IP (Transmission Control Protocol/ Internet Protocol). Так, за адресацию и гарантию того, что коммуникационный узел определит наилучший маршрут доставки пакета, отвечает протокол IP. Управление передачей данных реализуется протоколом TCP, который разбивает сообщение на пакеты и собирает принимаемое сообщение из пакетов. Протокол TCP следит за целостностью переданного пакета и контролирует доставку всех пакетов сообщения [10].

При обмене данными в Интернете необходимо, чтобы каждый компьютер, подключенный к сети, имел свой уникальный адрес. В сетях с протоколом TCP/IP для идентификации сетей и компьютеров используются IP-адреса. IP-адрес – это уникальный сетевой адрес узла в компьютерной сети, построенной по протоколу IP. IP-адрес имеет длину 4 Б, для удобства чтения каждый байт при записи отделяется точкой, например 194.85.160.21. IP-адрес включает в себя класс сети, номер сети и номер компьютера в ней. Класс сети определяет, сколько разрядов IP-адреса будет отведено для адресации сети, а сколько – для отдельных компьютеров в ней. Чтобы получить собственный адрес, локальная сеть должна быть зарегистрирована в Международном центре сетевой информации – InterNIC (Network Information Center), который выдает IP-адреса.

Цифровые адреса хороши для организации взаимодействия компьютеров, однако для пользователей предпочтительнее использование имен, т. к. их легче запоминать. Поэтому в Интернете введена доменная система имен (Domain Name System – DNS).

Доменный адрес (доменное имя) – это уникальный символический идентификатор подключенного к Интернету компьютера (хоста), зарегистрированный в DNS, например microsoft.com.

Протокол TCP/IP работает только с IP-адресами и не может непосредственно использовать доменные имена. Преобразованием имен в IP-адреса занимается система DNS. По существу, она представляет собой базу данных, в которой зафиксировано однозначное соответствие доменных имен и IP-адресов. Получая введенное пользователем доменное имя, коммуникационный узел посылает запрос одному из серверов DNS с целью преобразования этого имени в эквивалентный IP-адрес. Если сервер DNS не имеет информации об имени, он возвращает IP-адрес другого сервера DNS, способного ответить на запрос.

Доменное имя включает в себя идентификатор узла и идентификатор самого домена (домен верхнего уровня), указывающий на местоположение или тип организации, которой принадлежит сервер. Например, домен верхнего уровня com имеют коммерческие предприятия США. Также каждая страна имеет свой двухбуквенный домен верхнего уровня, например: uk – Великобритания, de – Германия, ru и рф – Россия.

Интернет в целом предоставляет пользователям свои услуги в виде сетевых сервисов или служб. Разные сервисы имеют разные протоколы. Протоколы служб Интернета называются прикладными протоколами. Так, например, известны протоколы передачи файлов – FTP и протокол удаленного управления компьютерами – Telnet и др. Самым используемым протоколом Интернета, как уже было упомянуто в разд. 1.2 пособия, является протокол передачи гипертекста – HTTP (Hypertext transfer protocol), благодаря которому функционирует Всемирная паутина – служба World Wide Web (WWW).

Гипертекст – это система текстовых документов с перекрестными ссылками. Веб позволяет включать в эти документы не только текст, но и графику, аудио- и видеоинформацию и ссылки на другие ресурсы Интернета. При этом объекты, на которые сделаны ссылки, могут находиться на удаленных компьютерах. Такое свойство делает Веб единой информационной структурой. Гипертекстовые документы Веба называют веб-страницами. Комплекс веб-страниц и других ресурсов, представляющий собой единый информаци-

онный блок, называется веб-узлом, или веб-сайтом.

На данный момент веб-сайты во Всемирной паутине в большинстве случаев представляют собой веб-приложения. Под веб-приложением понимается приложение, построенное на основе архитектуры «клиент – сервер», в котором в роли сервера выступает веб-сервер, а в роли клиента – веб-браузер [12]. Их базовая структура описана в следующем разделе.

Как упоминалось ранее, для адресации компьютеров и обращения к серверам в Интернете используются IP-адреса и доменные имена. А для доступа к отдельным ресурсам в Интернете (в том числе Вебе), расположенных на серверах (например, к веб-страницам, файлам рисунков и т. п.), используется URL (Uniform Resource Locator – стандартный указатель ресурса), который часто называют веб-адресом. Каждая веб-страница имеет уникальный URL, определяющий ее местоположение во Всемирной паутине, содержащей миллиарды веб-страниц. Базовый синтаксис URL представлен на рис. 2.2.

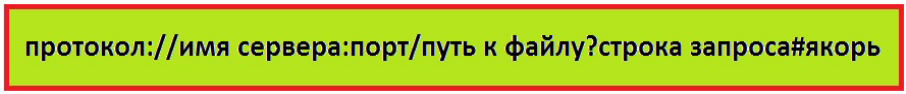

Рис. 2.2. Базовый синтаксис URL-адреса

Протокол указывает на протокол передачи данных между клиентом и сервером, например: HTTP, HTTPS, FTP и др. Имя сервера или IP-адрес указывает на веб-сервер, до которого нужно добраться. Иногда перед ним стоит имя пользователя и пароль, которые могут использоваться для подключения к серверу, в формате «имя_пользователя:пароль@». Номер порта не обязателен: когда он не указан, используется порт протокола по умолчанию. Например, портом по умолчанию для HTTP является 80, а для HTTPS – 443. Путь к файлу содержит имена папок пути и файла ресурса на веб-сервере. Строка запроса не обязательна: она является способом передать параметры скрипту на веб-сервере. Строка запроса обычно содержит одну или более пар имя-значение, между которыми стоит знак «=». Между этими парами ставится знак «&». Якорь в URL задает место или раздел внутри веб-страницы [3].

2.2. Архитектура веб-приложений

Веб-приложения обычно строятся на основе клиент-серверной архитектуры, в которой задания или сетевая нагрузка распределены между поставщиками услуг (серверами) и потребителями услуг (клиентами). Фактически клиент и сервер – это программное обеспечение. Типичным веб-клиентом является веб-браузер. Также сервером называют компьютер, на котором исполняется серверное программное обеспечение.

К преимуществам архитектуры «клиент – сервер» можно отнести то, что программный код, выполняемый на стороне сервера, не дублируется программой-клиентом, все вычисления выполняются на стороне сервера, поэтому требования к аппаратному обеспечению на стороне клиента снижаются, сервер, как правило, защищен лучше большинства клиентов, поэтому

на нем можно обеспечить соответствующий контроль прав доступа для программ-клиентов.

У такой архитектуры: высокая стоимость оборудования, в случае неработоспособности сервера прекращает работать вся сеть, для поддержания работоспособности сервера необходимо нанимать сотрудников – администраторов сервера.

Помимо двухуровневой архитектуры, веб-приложения могут иметь трех- и многоуровневую архитектуру. Наиболее часто веб-приложения имеют трехуровневую (трехзвенную) архитектуру, представленную ярусами данных, логики и представления (рис. 2.3).

Рис. 2.3. Трехуровневая архитектура веб-приложений

Стоит отметить, что все компоненты веб-приложения, в том числе и картографического, физически можно развернуть как на одном компьютере, так и на разных.

Уровень представления составляет клиентская часть веб-приложения, которая реализует интерфейс пользователя. Главная функция интерфейса – формирование запросов к серверу и отображение понятных пользователю результатов.

Уровень логики представлен веб-сервером и серверами приложений. Веб-сервер – программа на удаленном компьютере, принимающая запросы от многих клиентов и генерирующая результаты для клиентов в согласованных форматах. Для выполнения специальных запросов эта программа обращается к серверам приложений, выступая для них в качестве клиента. Сервер приложений – программа на удаленном компьютере, принимающая запросы от веб-сервера (или основного сервера) и генерирующая результаты для их включения в результаты общих запросов клиентов для основного сервера. Обычно сервер приложений предназначен для исполнения программ и скриптов, на которых построены веб-приложения, и его основная задача – обеспечение создания динамических страниц.

Уровень данных обеспечивает хранение данных. Он представлен сервером баз данных и другими хранилищами, например каталогами файловой системы.

Основной рабочий процесс веб-приложения состоит из нескольких шагов. Сначала с помощью веб-клиента – как правило, браузера – пользователь инициирует запрос к веб-серверу, печатая URL в адресной строке или переходя по ссылке на какой-либо веб-странице. Далее веб-сервер получает

запрос, выполняет синтаксический разбор URL, находит соответствующий документ или выполняет скрипт, по необходимости обращается к серверу приложений, в том числе к серверу баз данных. Затем веб-сервер возвращает клиенту результат, который может содержать документ, результат работы скрипта и результат, полученный от сервера приложений. Ответ обычно имеет формат HTML (сгенерированная сервером веб-страница с мультимедийным контентом: аудио-, видеопотоками, изображениями, информацией из базы данных и другими данными). Веб-клиент (браузер) получает ответ от веб-сервера, обрабатывает, отрисовывает его и представляет пользователю [12].

Сейчас набирает популярность новый подход к разработке веб-приложений, называемый AJAX. При использовании AJAX страницы веб-приложения не перезагружаются целиком, а лишь догружают необходимые данные с сервера, что делает их более интерактивными и производительными. Также в последнее время активно применяется технология WebSocket. При ее использовании постоянные запросы от клиента к серверу уже не отправляются, а создается двунаправленное соединение, при котором сервер может отправлять данные клиенту без запроса от последнего. Таким образом, появляется возможность динамически управлять содержимым страницы (контентом) в режиме реального времени.

При работе веб-приложения запросы и ответы осуществляются путем их передачи в сети с использованием протокола HTTP. Протокол HTTP устанавливает набор правил и процедур, которые используются веб-клиентами и веб-серверами для обмена сообщениями друг с другом (запросы и ответы). Например, используя HTTP, веб-сервер знает, какую информацию помещать в заголовок и тело ответного сообщения, а веб-клиент знает, чего ожидать в них.

HTTP-сообщения (запросы и ответы) имеют заголовок и тело. В заголовке HTTP-ответа обычно размещается информация об управлении кэшем (буфером памяти компьютера с возможностью быстрого доступа к данным, хранящимся в нем) – задаются директивы кэширования, которым клиент должен следовать (например, не кэшировать ответ), дается информация о типе содержимого (тип MIME (Multipurpose Internet Mail Extensions) указывает, является ли тело сообщения текстом HTML, изображением JPEG, звуковым файлом MP3 или другим типом данных), о коде состояния (указывает статус ответа, например: 200 – успешное выполнение запроса, 403 – доступ запрещен, 500 – ошибка сервера и др.).

Протокол HTTP отличается простотой, незапоминаемостью состояния и гибкостью [12]. Простота заключается в том, что обычно сеанс HTTP состоит из трех шагов: (1) клиент устанавливает соединение с веб-сервером и посылает ему запрос; (2) сервер обрабатывает запрос клиента, клиент в это время ожидает ответ сервера; (3) сервер посылает клиенту ответное сообщение с кодом состояния в заголовке ответа и данными (если имеются) в теле ответа, после чего закрывает соединение. Когда сервер ответил на запрос клиента, соединение между клиентом и сервером обычно разрывается и забывается. Серверы не сохраняют информацию о клиенте в промежутке между запросами, что позволяет значительно снизить нагрузку на сервер при работе с множеством клиентов. В этом и проявляется незапоминаемость состояния. Гибкость протокола HTTP позволяет передавать данные любого типа с указанием типа содержимого в HTTP-заголовке.

При использовании обычного протокола HTTP передаваемые между клиентом и сервером данные могут быть перехвачены. Для решения этой проблемы был разработан протокол HTTPS (Secure Hypertext Transfer Protocol – безопасный протокол передачи гипертекста), который исключает возможность «прослушки» благодаря использованию шифрования. Это обеспечивает защиту при передаче данных в Интернете, который по большей части является незащищенной сетью. HTTPS часто используют для передачи конфиденциальной информации, такой как персональные профили пользователей, пароли для входа в систему, данные кредитных карт.

2.3. Технологии разработки веб-приложений

Существуют различные технологии разработки веб-приложений, применяемые на стороне веб-сервера (бэкенд разработка), веб-клиента (фронтенд разработка) и для связи между ними (рис. 2.4) [12].

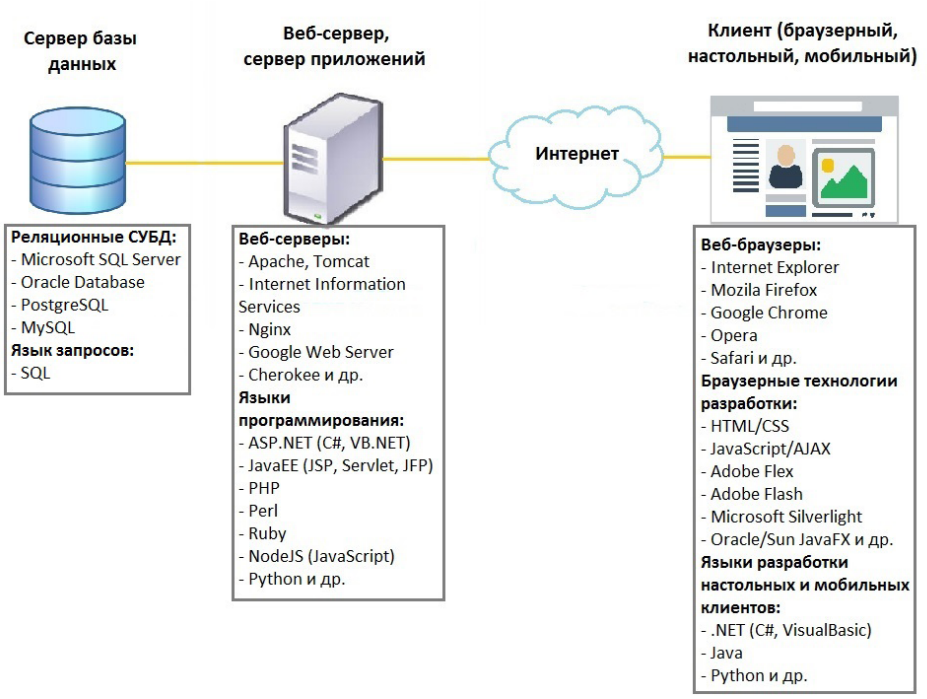

Рис. 2.4. Технологии разработки веб-приложений

2.3.1. Технологии разработки на стороне сервера

Веб-сервером называется как физический сервер (компьютер), так и программное обеспечение, выполняющее функции веб-сервера. В данном случае под веб-сервером понимается программное обеспечение, позволяющее принимать HTTP-запросы от клиентов и возвращать им HTTP-ответы с какими-либо данными: как правило, HTML-страницей, изображениями, файлами, медиа-потоком и др. Если говорить простыми словами, то веб-серверы формируют веб-сайты, содержимое которых читает пользователь, и предоставляют данные для использования другими веб-приложениями. Также веб-серверы являются контейнерами, позволяющими выполнение в них некоторых скриптов, таких как JSP (Java Server Pages), Java Servlet и ASP.NET, PHP и др. Веб-серверы имеют дополнительные функции, включающие в себя автоматизацию работы веб-страниц, ведение журнала обращений пользователей к различным ресурсам, аутентификацию и авторизацию пользователей, поддержку динамически генерируемых страниц (с использованием AJAX), поддержку протокола HTTPS для защищенных соединений с клиентами.

Примерами наиболее популярных веб-серверов (как программных продуктов) являются:

- Apache HTTP Server и Tomcat от Apache Software Foundation. Apache – веб-сервер с открытым кодом, отличающийся быстродействием и надежностью. Является кроссплатформенным программным обеспечением, поддерживает операционные системы Microsoft Windows, Linux, Mac OS, BSD и другие Unix-подобные системы. Apache – широко используемое программное обеспечение для веб-сервера, поддерживающее язык программирования Java и другие языки, если установлены соответствующие модули (PHP, Python, Ruby и др.). Также Apache позволяет использовать СУБД для идентификации пользователей. Настройка веб-сервера осуществляется при помощи текстовых файлов. Tomcat – контейнер сервлетов (скриптов на языке Java) с открытым исходным кодом, позволяет запускать веб-приложения и содержит программы для собственного конфигурирования. Он может использоваться в качестве самостоятельного веб-сервера и в сочетании с веб-сервером Apache HTTP Server;
- IIS (Internet Information Services, ранее Internet Information Server) от Microsoft. IIS – это проприетарный веб-сервер для работы в среде Microsoft Windows, поставляемый вместе с операционной системой Windows Server. Это второй по популярности в мире веб-сервер. Поддерживает набор языков программирования, на которых основывается платформа для разработки веб-приложений ASP.NET (языки C#, Visual Basic.NET, J#) от компании Microsoft, и другие языки, если установлены соответствующие модули расширения. Отличается легкостью в освоении и графическим интерфейсом управления;

- Nginx от компании NGINX Inc. – это веб-сервер и почтовый прокси-сервер с открытым исходным, работающий на Unix-подобных операционных системах (тестировалась сборка и работа на FreeBSD, OpenBSD, Linux, Solaris, Mac OS и др.), также имеется сборка для работы под Microsoft Windows. Nginx позиционируется производителем как простой, быстрый и надежный сервер, не перегруженный функциями. Для работы серверной части веб-приложений Nginx использует технологию FastCGI (клиент-серверный протокол взаимодействия веб-сервера и приложения), которая совместима со многими языками программирования, например: C/C++, C#, Python, Java, PHP и др.

Другими менее используемыми веб-серверами являются lighttpd, Google Web Server, Resin, Cherokee, Sambar Server, Rootage, Oracle/Sun Java Web Server и др.

Программы на стороне веб-сервера выполняются внутри контейнеров сервера веб-приложений. К основным языкам программирования, которые используются для написания данных программ, относятся следующие:

- языки, применяемые в платформе разработки веб-приложений ASP.NET от компании Microsoft. ASP.NET является важной частью среды .NET Framework. Среда .NET имеет технологии для программной разработки как на стороне сервера, так и на стороне клиента (в том числе веб-браузера и настольного клиента). ASP.NET содержит набор библиотек, облегчающих пользователям написание веб-страниц ASP.NET (имеют расширение .aspx) на языках C# и Visual Basic (VB.NET);

- PHP – язык сценариев, применяемый для создания веб-страниц, в частности динамических веб-сайтов. Язык разрабатывается группой энтузиастов в рамках проекта с открытым кодом;

- Java, в частности платформа программирования JavaEE (Java Platform Enterprise Edition). JavaEE включает в себя ряд технологий, которые предназначены для разработки веб-приложений на стороне сервера. Среди них Servlet, предназначенная для обслуживания запросов веб-клиентов, JSP, использующаяся для динамической генерации веб-страниц на стороне сервера, и JSF (Java Server Faces), которая является серверным фреймворком для разработки веб-приложений на языке Java. Веб-приложения, разработанные на Java, могут выполняться в Apache, Tomcat, Oracle/Sun Java Web Server и серверах веб-приложений IBM WebSphere, GlassFish и др.

Для разработки серверной части веб-приложений используются и такие языки программирования, как JavaScript (платформа Node.js), Perl, Ruby (фреймворк Ruby on Rails), Python и др. Стоит отметить, что программы, написанные на перечисленных языках, способны выполняться на различных веб-серверах. Исключением являются программы, разработанные с использованием платформы Node.js. Для их выполнения при помощи Node.js создается собственный веб-сервер.

Также разработчиками веб-приложений активно используются технологии, которые понимают серверные программы, написанные на любых

языках программирования. В качестве примера такой технологии можно привести CGI/FastCGI.

Для хранения данных, используемых в некоторых веб-приложениях, в том числе картографических, обеспечения их целостности, организации доступа к ним и управления ими часто применяются серверные реляционные СУБД, или серверы баз данных. Для реализации подключения к серверу баз данных используется строка подключения, которая содержит IP-адрес сервера (или URL-адрес), имя базы данных, имя пользователя базы данных и пароль к ней.

Для управления данными в реляционных СУБД используется специальный язык – SQL (Structured Query Language – язык структурированных запросов), являющийся набором операторов, инструкций и вычисляемых функций. Этот язык не зависит от конкретной СУБД и соответствует единому стандарту, который постоянно совершенствуется. В разных СУБД существуют различные процедурные расширения функционала SQL, реализующие дополнительные функции для работы с данными, например: математические функции, не включенные в стандарт, или функции для работы с пространственными данными.

Одна из ключевых особенностей языка SQL заключается в том, что с его помощью формируются запросы, описывающие, какую информацию из базы данных необходимо получить, а пути решения этой задачи СУБД определяет сама.

Большинство современных СУБД являются многопользовательскими, т. е. они могут предоставлять одновременный доступ к информации, хранящейся в базах данных, сразу для нескольких пользователей. При этом у каждого пользователя есть свой набор прав и ограничений при работе с данными. Например, одному пользователю разрешается только просматривать информацию из базы данных, а другому – и просматривать и редактировать. Можно также настроить права доступа не только для всей информации в базе данных, но и для каждой отдельной таблицы.

Среди серверных СУБД существуют как платные (Microsoft SQL Server и Oracle Database), так и свободные (PostgreSQL и MySQL).

Особенность СУБД Microsoft SQL Server (MSSQL) состоит в возможности работы в основном под управлением операционных систем семейства Windows (под операционные системы Linux и Unix имеется неполная поддержка). Также в качестве языка запросов MSSQL использует Transact-SQL (SQL со значительно расширенной функциональностью).

Oracle Database – самая распространенная корпоративная СУБД. Отличается высокой стоимостью и высокой надежностью. Oracle Database способна работать под управлением различных операционных систем: Microsoft Windows Server, Linux, Solaris, Unix-подобных систем и др.

Преимуществами PostgreSQL являются ее открытость (т. е. это свободная СУБД с открытым исходным кодом), поддержка многочисленных стандартов SQL и форматов данных, возможность расширения функциональности посредством написания скриптов на различных языках программирования. Как и Oracle Database, СУБД PostgreSQL реализована для множества операционных систем. Как и PostgreSQL, MySQL является свободной СУБД с открытым исходным кодом, способна работать под управлением множества операционных систем и поддерживает большое количество форматов данных.

2.3.2. Технологии разработки на стороне клиента

Веб-клиенты включают разнообразные программы: как те, что выполняются в браузере, так и настольные и мобильные приложения (например, известные картографические настольные и мобильные программы Google Earth, Google Maps, ArcGIS Explorer), выполняемые вне браузера. Веб-браузер – наиболее популярный вид веб-клиентов.

Веб-браузер – это программное приложение для извлечения и представления информационных ресурсов Всемирной паутины. Технически веб-браузер является клиентом, реализующим спецификации HTTP, HTML и JavaScript, т. е. он умеет общаться с веб-серверами, отображать текст с HTML-разметкой, интерпретировать и выполнять программный код, написанный на языке JavaScript.

Наиболее популярными веб-браузерами являются Internet Explorer (IE), Edge, Google Chrome, Mozilla Firefox, Opera, Apple Safari. Существуют небольшие различия в том, как эти браузеры поддерживают спецификации HTML и JavaScript: в зависимости от веб-браузера одна и та же веб-страница может выглядеть и вести себя по-разному. Поэтому проектировать и разрабатывать веб-приложение нужно так, чтобы оно корректно отображалось в большинстве популярных браузеров на разных устройствах.

На стороне веб-браузера применяются следующие технологии фронтенд-разработки:

- HTML и CSS. HTML (HyperText Markup Language – язык разметки гипертекста) – стандартизированный язык разметки документов (веб-страниц) во Всемирной паутине, который интерпретируется веб-браузерами. Полученный результат интерпретации в форматированном виде отображается на экране устройства пользователя в понятной для него форме. Текстовые документы, содержащие разметку на языке HTML, обычно имеют расширения .html или .htm. Язык HTML описывает содержание документа при помощи набора структурных и семантических элементов, называемых тегами или дескрипторами. Существует несколько версий языка HTML, последняя из которых HTML5. Эта версия создана для улучшения уровня поддержки мультимедиатехнологий, удобочитаемости и простоты синтаксического анализа кода. Благодаря улучшению поддержки мультимедиа в HTML5 появилась возможность создавать и управлять графическими и мультимедийными объектами на веб-страницах без использования сторонних плагинов, например без Adobe Flash Player.

Для оформления внешнего вида HTML-документов используется формальный язык стилей CSS (Cascading Style Sheets – каскадные таблицы стилей). Таблицы стилей CSS можно писать как внутри самого html-документа, так и в отдельном подключаемом к нему файле с расширением .css. Файлы HTML и CSS хранятся на сервере и по запросу передаются клиенту для интерпретации в браузере. Более подробно работа с HTML разметкой и таблицами стилей CSS описана в гл. 3;

- JavaScript – язык скриптов (фрагментов кода), интерпретируемых и исполняемых веб-браузером, которые позволяют сделать веб-страницы динамическими и интерактивными. Это самый популярный язык скриптов, используемый на многих миллионах веб-страниц. JavaScript не стоит путать с языком Java, хотя его базовый синтаксис намеренно сделан похожим на Java и C++, чтобы сократить число новых понятий при его изучении. JavaScript прост, благодаря чему с ним могут работать люди с начальными навыками программирования, и безопасен, т. к. имеет ограниченный доступ к локальному жесткому диску клиентского компьютера, на котором он выполняется. JavaScript обычно не зависит от платформы, но может выполняться по-разному в различных браузерах. Разработчик, создающий кросс-браузерное приложение (корректно работающее во многих браузерах), должен учитывать возможную несовместимость. Существует также комбинация HTML, CSS и подмножества JavaScript, называемая DHTML (динамический HTML), которая используется для придания веб-сайтам относительно простой интерактивности. При разработке клиентской части картографических веб-приложений могут использоваться различные библиотеки и интерфейсы программирования приложений (API – Application Programming Interface), созданные специально для этих целей, например: ArcGIS API for JavaScript, Leaflet и др. API сервиса предоставляет набор готовых классов, процедур, функций и структур или констант, с помощью которых разработчики могут создавать собственные приложения. Скрипты, написанные на языке JavaScript, так же, как HTML- и CSS-документы, хранятся на сервере и отправляются пользователю для исполнения в браузере. Синтаксис и возможности использования языка JavaScript для разработки веб-приложений рассмотрены в гл. 4;

- AJAX (Asynchronous JavaScript and XML – асинхронный JavaScript и XML) – группа методов веб-разработки, используемых для создания более быстрых и интерактивных браузерных веб-приложений. AJAX использует асинхронное взаимодействие между браузером и веб-сервером, благодаря чему данные для веб-страницы можно выбирать с сервера в фоновом режиме (асинхронно), не влияя на текущее отображение и поведение страницы. Без AJAX, как правило, необходимо переходить со страницы на страницу и ожидать загрузки новой страницы перед тем, как начать взаимодействовать с ней. С AJAX вы можете оставаться на странице, пока необходимые данные извлекаются с сервера и затем анализируются и обновляются на той же странице. Главным достижением AJAX является улучшение взаимодействия с пользователем. С точки зрения методологии разработки AJAX позволяет отделять данные от форматирования, что является предпочтительным при проектировании программного обеспечения. Таким образом, при использовании AJAX страницы веб-приложения не перезагружаются целиком, а лишь догружают необходимые данные с сервера, что делает их более интерактивными и производительными;

- Adobe Flex – это среда с открытым кодом для создания высокоинтерактивных веб-приложений, называемых также насыщенными интернет-приложениями (Rich Internet Applications – RIAs), которые единообразно развертываются во всех основных браузерах. Она может передавать сложные анимационные и переходные эффекты, что превращает ее в мощный язык реализации насыщенных веб-приложений. Flex использует MXML – язык для описания компоновки и поведения пользовательского интерфейса, а также ActionScript – объектно ориентированный язык программирования со строгим контролем типов данных для создания клиентской логики. Программы, разработанные на Flex, могут выполняться в веб-браузерах с подключенным модулем Adobe Flash Player или вне их – в кросс-платформенной среде выполнения Adobe AIR. Благодаря этому приложения Flex совместимы со всеми основными браузерами и платформами;

- Adobe Flash – мультимедийная платформа компании Adobe Systems для разработки насыщенных веб-приложений или мультимедийных презентаций. Широко используется для создания рекламных баннеров, анимации, игр и для воспроизведения на веб-страницах видео- и аудиозаписей. Платформа включает в себя ряд средств разработки, а также программу для воспроизведения flash-контента – Adobe Flash Player, хотя flash-контент умеют воспроизводить и многие плееры сторонних производителей. Adobe Flash позволяет работать с векторной, растровой и с трехмерной графикой, используя при этом графический процессор, и поддерживает двунаправленную потоковую трансляцию аудио и видео;

- Microsoft Silverlight – кросс-браузерный и кросс-платформенный подключаемый модуль для реализации RIAs веб-приложений, который использует графику, анимацию или видео в среде .NET и позволяет запускать приложения, содержащие анимацию, векторную графику, аудио- и видеоролики. В нем применяются XAML (Extensible Application Markup language – расширяемый язык разметки приложений) – для разработки пользовательского интерфейса и языки программирования среды .NET, такие как C# и VB.NET, – для разработки бизнес-логики. Программы Microsoft Silverlight могут выполняться в веб-браузерах с подключенным модулем Microsoft Silverlight или как настольные приложения в среде выполнения WPF (Windows Presentation Foundation);

- Oracle/Sun JavaFX – программная платформа Oracle/Sun, предназначенная для создания и развертывания RIAs, которые могут выполняться на самых разнообразных устройствах. JavaFX полностью интегрирована со средой выполнения Java (Java Runtime Environment – JRE), что обеспечивает выполнение приложений этой платформы в браузере, на настольном компьютере или в мобильном телефоне, если в них имеется соответствующая JRE;

- Cappuccino – технология (фреймворк) для создания веб-приложений. Главная особенность Cappuccino – это созданный специально

для него язык Objective-J, являющийся надстройкой над языком JavaScript и предоставляющий свой API. Язык Objective-J при загрузке на стороне клиента интерпретируется и работает как обычный JavaScript.

2.3.3. Форматы обмена данными между сервером и клиентом

Согласно современным принципам разработки веб-приложений рекомендуется отделять данные от форматирования и обмен данными от их представления. От формата данных, используемого при обмене сообщениями между сервером и клиентом, может сильно зависеть нагрузка на канал связи и, соответственно, эффективность приложения. Для обмена данными в Вебе существует три основных формата [12]: XML, JSON и AMF.

XML (Extensible Markup Language) – расширяемый язык разметки, похожий по написанию на язык HTML. У формата XML есть несколько преимуществ. Во-первых, язык XML позволяет определять собственные теги и атрибуты. Во-вторых, этот формат не зависит от платформы, поскольку файлы XML представляют собой простой текст. В-третьих, XML самоописателен (теги и атрибуты в файле – слова обычного языка). Файлы такого формата могут быть обработаны автоматически, потому что они хорошо структурированы. Также файлы XML могут быть сверены с их схемой для проверки. Благодаря этим преимуществам XML является самым часто используемым форматом обмена данными в Вебе. Однако у него есть и некоторые недостатки. Он громоздок, поскольку использует теги из обычных слов для разметки данных. Кроме того, анализ файлов XML (т. е. извлечение значений данных из определенных узлов XML-документа) не отличается эффективностью, особенно в JavaScript – наиболее широко используемом во Всемирной паутине языке скриптов.

JSON (JavaScript Object Notation) – это облегченный формат обмена данными между компьютерами. Как сказано в определении Стандарта скриптового языка программирования ECMA (Европейской ассоциации производителей компьютеров), он является производным от литералов JavaScript. JSON более компактен, чем XML, его конструкции легче разбираются в JavaScript, для которого JSON является внутренне используемым типом данных. Основное применение JSON – программирование веб-приложений, где он служит альтернативой XML. Пример использования XML и JSON для описания одной и той же информации представлен на рис. 2.5.

AMF (Action Message Format) – это двоичный формат, разработанный Adobe и используемый главным образом для обмена данными между клиентскими приложениями на Adobe Flex/Flash и веб-серверами. AMF является собственным типом данных Flex, что делает в нем его обработку более эффективной по сравнению с JSON. По сравнению с XML, который также представляет собой собственный тип данных во Flex, AMF намного компактней, благодаря чему его передача и обработка более эффективны. AMF чаще всего используется во Flex-приложениях, но существуют и его реализации в PHP, Java и .NET.

XML

```
<?xml version="1.0" encoding="UTF-8" ?>
<person>
 <name>Иван</name>
 <age>37</age>
 <mother>
  <name>Ольга</name>
  <age>58</age>
 </mother>
 <children>
  <child>Маша</child>
  <child>Игорь</child>
  <child>Таня</child>
 </children>
 <married>true</married>
 <dog null="true" />
</person>
```

JSON

```
{
 "person":{
  "name":"Иван",
  "age":37,
  "mother":{
   "name":"Ольга",
   "age":58
  },
  "children":[
   "Маша",
   "Игорь",
   "Таня"
  ],
  "married":true,
  "dog":null
 }
```

Рис. 2.5. Пример описания информации в форматах XML и JSON

2.4. Особенности архитектуры картографических веб-приложений

2.4.1. ГИС-серверы

Базовая архитектура картографических веб-приложений соответствует архитектуре обычных веб-приложений, но к этой архитектуре добавляются компоненты ГИС (рис. 2.6). Наиболее важным компонентом является ГИС-сервер. ГИС-сервер – это комплекс программного обеспечения, предназначенного для публикации пространственных данных, карт и алгоритмов (моделей) их обработки в локальных или глобальных сетях (в том числе в сети Интернет), а также обработки запросов к ним. Он представляет собой разновидность сервера приложений.

Рабочий процесс картографического веб-приложения начинается с того, что пользователь посылает запрос к веб-серверу по Интернету посредством HTTP, используя приложение веб-ГИС с помощью клиента, которым может быть веб-браузер, настольное или мобильное приложение. Веб-сервер

направляет относящиеся к ГИС запросы в ГИС-сервер. ГИС-сервер обрабатывает запрос, который может заключаться в отображении карты, поиске данных или выполнении анализа. Данные, карта или другой сформированный ГИС-сервером результат посылается веб-сервером посредством HTTP обратно клиенту. Наконец, клиент отображает результат пользователю, тем самым завершая цикл «запрос-ответ».

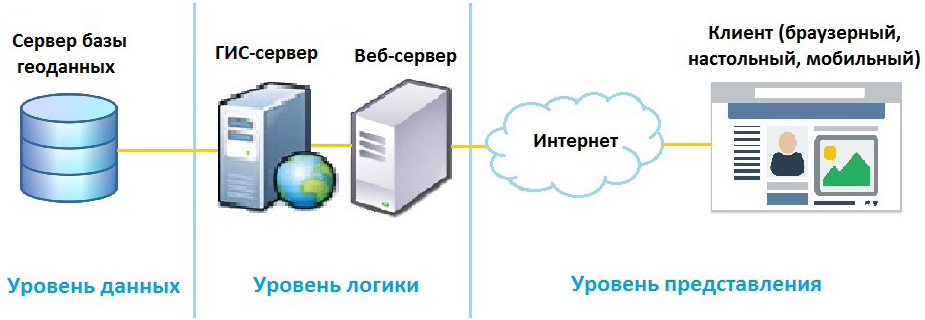

Рис. 2.6. Архитектура простейшего картографического веб-приложения

Функциональность, возможность настройки под пользователя, масштабируемость и производительность ГИС-сервера имеют решающее значение для успешной работы картографического веб-приложения. Существует несколько наиболее популярных программных продуктов ГИС-серверов, как коммерческих, так и открытых.

Среди коммерческих наиболее популярны продукты ArcGIS Server от компании ESRI (США), Location Intelligence Module (LIM), входящий как компонент в платформу MapInfo Spectrum Platform от компании MapInfo Corp. (США); также к этой категории можно отнести платформу NextGIS Web от ООО NextGIS, в которой есть серверная часть.

ArcGIS Server – коммерческий программный продукт (ГИС-сервер), предназначенный для публикации пространственных данных и работы с ними в сети Интернет, посредством картографических веб-сервисов. Он способен функционировать под управлением операционных систем Windows и Linux. Для хранения публикуемых пространственных данных и организации многопользовательского доступа к ним ArcGIS Server поддерживает взаимодействие с разными СУБД. При этом в ArcGIS имеется собственная технология ArcSDE, которая позволяет хранить в СУБД специфичные структуры данных, такие как топология, классы отношений и растры, и настроить поддержку многопользовательского редактирования данных [4].

При разработке клиентской части картографических веб-приложений, использующих сервисы, опубликованные с помощью ArcGIS Server, можно обратиться к нескольким специально созданным API: ArcGIS API for JavaScript, for Flex, for Silverlight (однако стоит отметить, что с 1 июня 2016 г. API for Flex и for Silverlight не развиваются, поэтому рекомендуется использовать API for JavaScript). Для разработки настольных и мобильных клиентских ГИС-приложений также есть API под разные языки программирования и платформы (ArcGIS Runtime SDK).

Для управления параметрами ArcGIS Server существует визуальный

веб-интерфейс, доступный через браузер, и интерфейс, доступный через настольное приложение ArcCatalog. Настройка отображения слоев и публикация картографических веб-сервисов осуществляется при помощи настольного программного продукта ArcGIS Desktop, или ArcGIS Pro.

Стоит также отметить, что ArcGIS Server является главным компонентом портала ArcGIS Enterprise (Portal for ArcGIS). Портал ArcGIS Enterprise позволяет обеспечить общий доступ ко всем ресурсам, созданным в одной организации, при помощи отдельного сайта и настольных программных продуктов ArcGIS. Также ArcGIS Enterprise включает в себя инструменты, позволяющие простым пользователям создавать собственные веб-приложения без использования программирования.

Кроме того, использовать веб-сервисы, опубликованные на ArcGIS Server, можно и в облачной инфраструктуре ArcGIS Online, реализованной по принципу SaaS (программное обеспечение как сервис). ArcGIS Online может напрямую выступать в качестве клиента для сервисов ArcGIS Server, при этом ArcGIS Online также предоставляет отдельные инструменты и шаблоны, позволяющие создавать собственные простые клиентские веб-приложения. Более подробная информация о работе с ГИС-сервером ArcGIS Server и с ArcGIS Online изложена в гл. 3.

LIM – это один из компонентов коммерческой платформы MapInfo Spectrum Platform, являющийся сервером инфраструктуры пространственных данных (ГИС-сервером). Геоинформационная платформа Spectrum способна работать в различных операционных системах, включая Windows, Linux и UNIX-подобные системы. Эта платформа поддерживает все распространенные типы пространственных баз данных, включая возможность прямой записи и чтения пространственных объектов из разных СУБД, в том числе Oracle, Microsoft SQL Server, PostgreSQL и др. [7]

Визуализация и публикация картографических веб-сервисов, а также работа с ними может осуществляться при помощи встроенного веб-приложения (ГИС-портал), а также настольного программного обеспечения MapInfo. Для разработки собственных веб-приложений используются языки технологии .NET.

NextGIS Web – картографическая платформа, состоящая из серверной и клиентской части. Серверная часть, предназначенная для хранения и отрисовки данных, написана на языке программирования Python и работает под управлением операционной системы Linux. Клиентская часть предоставляет пользовательский веб-интерфейс для интерактивного управления геоданными и взаимодействия с ними через карту. Вся конфигурация системы хранится внутри базы данных PostgreSQL с модулем расширения PostGIS. Платформа может интегрироваться с настольной ГИС QGIS.

Среди свободного программного обеспечения наиболее используемыми ГИС-серверами являются GeoServer, MapServer и QGIS Server.

GeoServer – ГИС-сервер с открытым исходным кодом, способный работать на операционных системах семейств Windows, Linux, UNIX, Solaris, macOS и др. Он поддерживает большое число форматов пространственных данных и спецификаций картографических веб-сервисов. Так же как и большинство ГИС-серверов, GeoServer имеет возможность подключения к базам пространственных данных, работающих под управлением СУБД PostgreSQL (PostGIS), Oracle, MySQL, MS SQL Server и др. Для управления ГИС-сервером

и для публикации пространственных данных и настройки их отображения GeoServer имеет визуальный интерфейс, доступный в окне обычного веб-браузера. При настройке символов для отображения слоев используется специальный язык – Styled Layer Descriptor (SLD) [22]. Создание клиентской части картографических веб-приложений, использующих для работы с пространственными данными GeoServer, обычно осуществляется при помощи специальных JavaScript-библиотек, таких как Leaflet и OpenLayers. Подробно о работе с GeoServer написано в гл. 3.

MapServer – кроссплатформенный программный продукт с открытым исходным кодом [27]. Он может работать под управлением различных операционных систем: Windows, Linux, macOS, Solaris и др. Так же как и перечисленные ранее ГИС-серверы, MapServer имеет возможность интеграции с большинством современных СУБД: MS SQL Server, Oracle, MySQL, PostgreSQL и др. Главной сложностью при работе с MapServer, особенно для неопытных пользователей, является его настройка, осуществляемая при помощи текстовых файлов. Сам по себе MapServer представляет собой программу, работающую на основе технологии CGI. Эта программа активна на веб-сервере, она динамически создает запрошенные клиентом карты в виде изображений и генерирует содержащие их веб-страницы (рис 2.7). Сама карта изначально создается в специальном текстовом файле конфигурации, называемом Mapfile (имеет расширение .map). Этот файл определяет экстент карты, перечень слоев, их проекции и символы, а также сообщает программе MapServer, где находится источник данных и куда выводить изображения. Mapfile создается при помощи собственного языка, описанного на сайте MapServer [27]. Для создания полнофункциональных веб-приложений или настольных приложений MapServer также предоставляет интерфейс MapScript, который позволяет использовать языки программирования Perl, PHP, Java, Ruby C#, Python и др. Таким образом, MapServer, в отличие от перечисленных ранее ГИС-серверов, предоставляет в основном интерфейсы разработки веб-приложений на стороне сервера.

QGIS Server – еще одно программное обеспечение (ГИС-сервер) с открытым исходным кодом, работающее в операционных системах семейства Windows, Linux, FreeBSD. Как и MapServer, QGIS Server представляет собой приложение FastCGI/CGI, которое работает совместно с веб-сервером (например, Apache или др.) [5]. Он интегрируется с настольной ГИС QGIS и поэтому поддерживает подключения к тем же СУБД: PostgreSQL (PostGIS), MS SQL Server, DB2 и др. Через QGIS также имеется возможность публикации карт и пространственных данных на QGIS Server и инструменты их подключения для просмотра. При публикации карт на QGIS Server, как и у MapServer, создается Mapfile в каталоге на веб-сервере, где установлен ГИС-сервер. Из Mapfile на сервере при запросах пользователями генерируются изображения карты. Стоит отметить, что QGIS Server практически не имеет собственных API для разработки веб-приложений, особенно клиентской части, у него есть только поддержка плагинов на языке Python и поддержка библиотеки Qt (язык C++).

ГИС-серверы предоставляют доступ к опубликованным пространственным данным, картам и моделям в виде веб-служб, называемых картографическими веб-сервисами. Веб-служба – это программа, которая выполняется на веб-сервере и предоставляет программные интерфейсы другим программам через Всемирную паутину [12]. Существуют общепринятые стандарты (протоколы) картографических веб-сервисов (веб-служб), в

соответствии с которыми предоставляются пространственные данные в сети Интернет. Эти стандарты поддерживаются всеми существующими ГИС-серверами.

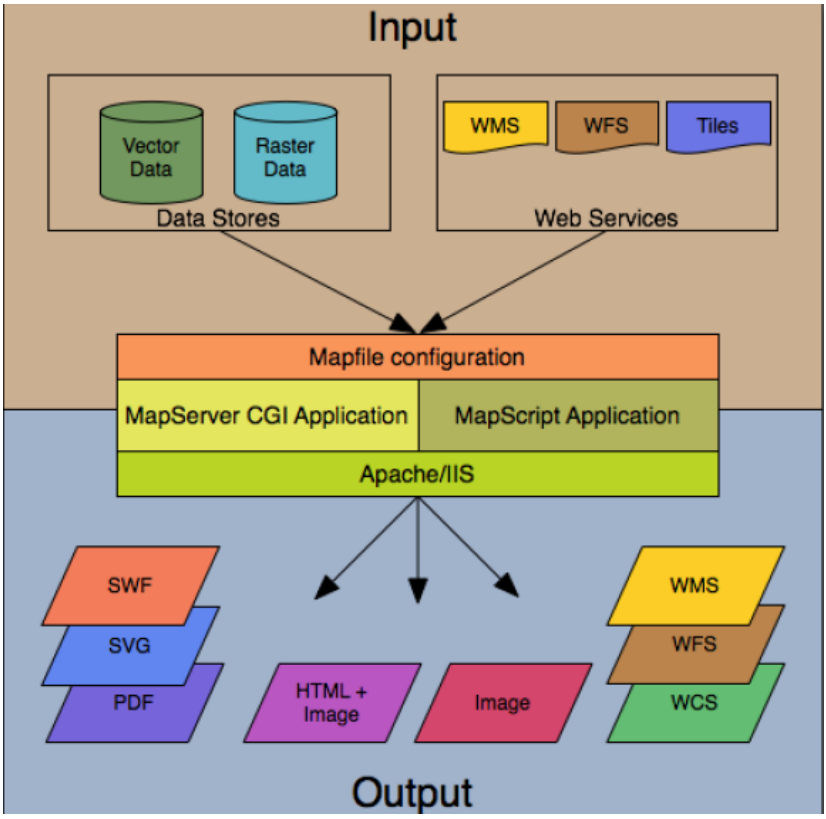

Рис. 2.7. Базовая архитектура приложений MapServer

2.4.1.1. Стандарты картографических веб-сервисов

Разработкой общих принципов и стандартов в области веб-картографии, в том числе стандартов картографических веб-служб, занимается международная некоммерческая организация Открытый геопространственный консорциум (OGC – Open GIS Consortium). OGC основан в 1994 г., его членами являются крупные коммерческие, академические и государственные организации, занимающиеся разработкой и исследованиями в области информационных технологий, например: Boeing, Oracle, ESRI, MapInfo, Intergraph, Google и др.

OGC разработаны следующие стандарты картографических веб-служб: Web Map Service (WMS), Web Feature Service (WFS), Web Coverage Service (WCS), Web Processing Service (WPS), Web Map Tile Service (WMTS), Catalog Service for

the Web (CSW), OpenGIS Location Service (OpenLS), Sensor Web Enablement (SWE) и др. [8, 12]

WMS – стандарт картографического веб-сервиса, определяющий параметры для запроса и представления карт в сети Интернет, которые формируются в виде графических изображений (в форматах PNG, JPEG, GIF и др.). В стандарте WMS определены параметры карты у сервера. Запрос может включать следующие параметры: перечень слоев, экстент, систему координат, размер и формат изображения. При использовании WMS можно получить информацию об объектах на карте, включающей координаты точек объектов из выбранных слоев и их атрибуты. Также стандарт WMS определяет операцию запроса метаданных картографического веб-сервиса, содержащих его имя, экстент, слои и их свойства, систему координат, краткое описание и др.

WFS – стандарт картографического веб-сервиса, определяющий параметры для передачи векторных графических пространственных данных в сети Интернет. Изначально спецификация WFS поддерживала только чтение векторных данных и их атрибутов, а более поздняя версия – WFS-T (WFS Transactional) – поддерживает запись (редактирование) пространственных данных, т. е. операции вставки, удаления, обновления атрибутивных и пространственных данных, находящихся на сервере. Стандарт WFS также определяет такие операции, как запрос метаданных о веб-сервисе, включая его содержимое и возможности, типы пространственных объектов, которые он может предоставлять, и поддерживаемые им операции. При использовании WFS возможно запросить структуру типов пространственных объектов, которые он поддерживает, заблокировать один или несколько объектов на время редактирования. Картографические веб-сервисы WFS можно применять для картографических задач и поиска, а также для отсечения, проецирования, архивирования и отправки географических данных.

WCS – стандарт картографического веб-сервиса, который поддерживает получение пространственных данных в виде растровых покрытий, включая данные дистанционного зондирования Земли, цифровые модели рельефа (ЦМР) и др. В отличие от стандарта WMS, который позволяет получить визуальное представление векторных и растровых данных в виде изображения, WCS возвращает исходные растровые данные. Стандарт WCS определяет ряд основных операций: получение метаданных веб-сервиса, запрос полного описания одного или нескольких покрытий, предоставляемых WCS, запрос покрытия в пределах указанных географических областей и промежутков времени, в указанной системе координат и в указанном формате.

WPS – спецификация веб-сервиса геообработки (геопроцессинга), описывающая правила для входящих и исходящих данных. Геопроцессинг может включать любой алгоритм, расчет или модель, оперирующие векторными или растровыми пространственными данными.

WTMS – спецификация, расширяющая возможности WMS для создания кэшированных картографических сервисов (рис. 2.8). Кэширование – это процесс, при котором сервер заранее формирует и сохраняет набор квадратных фрагментов изображений карт (тайлов) для каждого заданного масштабного уровня для их быстрой передачи клиенту с целью визуализации.

CSW – стандарт сервиса каталога для представления в Вебе. Спецификация CSW предназначена для обмена пространственными данными, она

поддерживает публикацию метаданных, пространственных данных и поиск среди них. Есть два вида CSW: только для чтения и с возможностью редактирования.

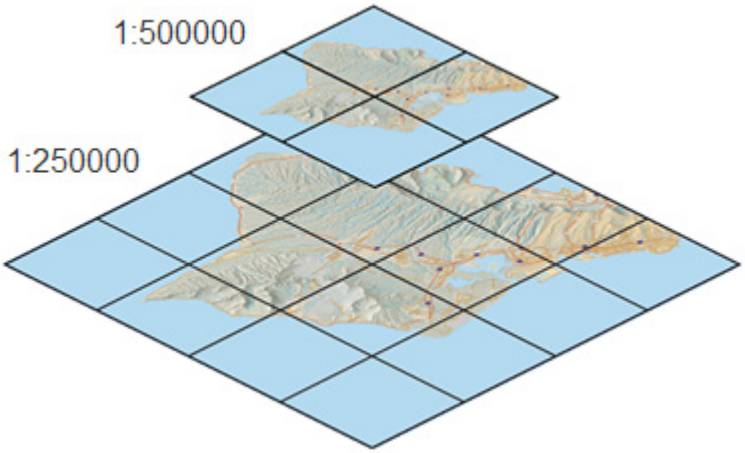

Рис. 2.8. Процесс кэширования данных для двух масштабных уровней

OpenLS – стандарт, определяющий интерфейсы веб-служб, действующих на основе местоположения клиента (LBS – Location Based Services). Спецификация OpenLS включает в себя ряд различных типов сервисов, в том числе веб-службы построения маршрутов, сервисы навигации, сервисы географических указателей (поиск точек интереса, например магазинов и др. услуг), сервисы геокодирования, службы представления (для отображения точек интереса, маршрутов и другой информации на устройстве клиента).

SWE – стандарт сервиса, который поддерживает и отображает информацию с датчиков в сети Интернет, например, данные о температуре воздуха и др. Стандарт SWE включает в себя ряд других стандартов: Sensor Observation Service (SOS) – службу съема показаний с датчиков, Sensor Planning Service (SPS) – службу планирования работы датчиков и Sensor Alert Service (SAS) – службу экстренных сообщений датчиков.

2.4.1.2. Стандарты форматов пространственных данных

Помимо стандартов картографических веб-сервисов, OGC разработаны стандарты форматов пространственных данных, которые могут быть использованы для их передачи от веб-служб к клиентам. Среди них Geography Markup Language (GML – язык разметки географии), Keyhole Markup Language (KML/KMZ), GeoPackage (GPKG), GeoRSS и др. Стоит отметить, что эти форматы не являются обязательными для использования, пространственные данные клиентам можно передавать и в других форматах, которые поддерживает та или иная ГИС, например: векторные shape-файлы, растры в формате TIFF и т. п.

GML – стандарт формата данных, который описывает, как использовать XML для передачи географической информации. Этот формат позволяет описывать геометрию и свойства пространственных объектов, систему координат, топологические отношения. Он часто используется в сервисах WFS при передаче пространственных объектов, запрошенных от сервиса пользователем.

KML/KMZ – формат описания пространственных объектов, созданный на основе языка разметки XML. Файл формата KML может включать в себя описание множества пространственных объектов разного содержания (точки, линии, полигоны, изображения, подписи, 3D-модели и др.) в значениях координат (широты и долготы). Также KML может содержать параметры пространственных объектов (например, высоту, направление, наклон), условные знаки для них, связанные с объектами тексты, изображения и ссылки на другие ГИС-сервисы. Файлы KML вместе с сопровождающими пиктограммами условных знаков и фотографиями часто сжимаются в архивы формата ZIP и имеют расширение имени KMZ.

GeoPackage – открытый формат хранения пространственных данных (векторных объектов, наборов матричных листов изображений и растровых карт в различных масштабах, схем, сетевых данных, облаков точек, стилей и метаданных), реализованный в виде расширения для базы данных SQLite. Файл формата GPKG содержит пространственные данные в виде таблиц и метаданные с заданными определениями, описанием схемы данных и ограничениями.

GeoRSS. RSS (Really Simple Syndication или Rich Site Summary) – это средство поддержки веб-каналов для публикации часто обновляемой информации, такой как записи в блогах или заголовки новостей. Технология RSS характеризуется простотой, режимом реального времени и агрегаторами (программами чтения) потоков. RSS-форматы определяют несколько тегов XML, которые указывают на название, дату публикации, авторскую аннотацию и дают URL-ссылку на полный текст упомянутой статьи. RSS позволяет пользователю подписываться на любые новости и веб-сайты. Агрегаторы новостей RSS регулярно проверяют обновления потоков, на которые подписан пользователь. С расширением использования RSS появилась необходимость не только знать, что происходит, но и где это происходит. В связи с этим появился стандарт GeoRSS, который позволяет кодировать информацию о местоположении в RSS и других XML-сообщениях. Стандарт GeoRSS определяет три формата: W3C Geo, OGC GeoRSS-Simple и GeoRSS-GML. W3C Geo позволяет описывать только точки в системе координат WGS-84. OGC GeoRSS-Simple может описывать простые виды геометрии (точки, линии, полигоны) в WGS-84 и их дополнительные свойства, например: тип и название объекта, высоту, радиус, отношения с другими объектами и др. GeoRSS-GML поддерживает более широкий спектр видов пространственных объектов и разные системы координат по сравнению с OGC GeoRSS-Simple. GeoRSS-GML является профилем GML [12].

2.4.2. Базы пространственных данных

Важной частью любого картографического веб-приложения является качественная база пространственных данных, которая представляет собой среду хранения и управления данными разного типа: векторными, растровыми, сетевыми данными, 3D-данными и др. База пространственных данных может быть как небольшой персональной, так и большой корпоративной (многопользовательской). При организации картографического веб-сервиса лучше использовать многопользовательские базы данных, т. к. они позволяют настроить одновременный доступ множеству пользователей к данным, а также и к их редактированию. При этом обеспечивается безопасность, целостность и резервирование данных, а также осуществляется контроль версий при одновременном и последовательном редактировании. Многопользовательские базы пространственных данных обычно создаются и работают под управлением различных реляционных (табличных) СУБД (Microsoft SQL Server, Oracle Database, PostgreSQL, MySQL, Informix, IBM DB2 и др.).

Для организации возможности работы с пространственными данными в среде реляционных СУБД в геоинформационных программных продуктах существуют свои механизмы и расширения. Так, например, в линейке программных продуктов ArcGIS представлена технология ArcSDE, которая обеспечивает поддержку версионного (многопользовательского) редактирования баз пространственных данных практически в любой реляционной СУБД [4]. Подобный механизм в виде расширения для СУБД PostgreSQL под названием PostGIS используется и в открытой геоинформационной системе QGIS [31].

К качеству базы пространственных данных картографического веб-приложения предъявляется ряд условных требований:

- хранение богатой коллекции пространственных данных централизованным или распределенным способом;
- применение правил и отношений в организации данных;
- поддержка пространственных реляционных моделей (топологии, сетевые данные) по необходимости;
- обеспечение целостности пространственных данных;
- возможность работы в среде многопользовательского доступа и редактирования и поддержка версии данных;
- поддержка определяемых пользователем типов пространственных объектов;
- наличие средств для надежной защиты данных, резервного копирования и восстановления;
- сохранение высокой производительности при увеличении объема данных и числа одновременно подключенных пользователей.

2.4.3. Клиенты картографических веб-приложений

Клиентская часть картографического веб-приложения является интерфейсом взаимодействия конечного пользователя со всей системой. Как было отмечено ранее, клиентами картографического веб-приложения, помимо обычных веб-браузеров, могут выступать настольные, мобильные и серверные приложения.

Браузерные клиенты первых картографических веб-приложений были достаточно просты и медлительны. Но с появлением и использованием языка JavaScript и различных технологий, таких как AJAX, Flash, Flex, Silverlight, JavaFX, стало возможным создавать веб-приложения с более дружественным, динамичным и насыщенным интерфейсом (RIA). Также современные браузерные клиенты благодаря перечисленным технологиям могут выполнять большинство функций современных настольных ГИС и таким образом даже способствуют появлению облачных ГИС, например ArcGIS Online.

Настольные клиенты веб-ГИС представляют собой приложения, разработанные с помощью соответствующих языков программирования, например: языков платформы .NET (C#, VB.NET), Java, C++ и др. Особенность настольных приложений, в отличие от браузерных, состоит в том, что они могут иметь доступ к локальным ресурсам компьютера пользователя (файловой системе, периферийным устройствам). В качестве примеров настольных клиентов можно привести настольные ГИС, которые способны подключаться к картографическим веб-службам: ArcGIS Desktop, QGIS и др. При этом настольные ГИС могут проводить некоторые операции с сервисами, недоступные для браузерных клиентов. Также в качестве настольных клиентов картографических веб-сервисов можно рассматривать такие известные программы как Google Earth, SASPlanet и ряд других.

Мобильные клиенты, которые позволяют использовать веб-ГИС в полевых условиях и передавать информацию о местоположении на основе глобальных систем позиционирования, становятся все более популярными. Их можно разделить на две основные категории: клиенты на основе веб-браузера и клиенты на основе нативных приложений (т. е. приложений, созданных под определенную платформу: Android, iOS, Windows Phone и др.). Примерами нативных мобильных клиентских приложений являются ArcGIS Mobile, ArcGIS for iPhone, Microsoft Bing Maps Mobile, Google Earth Mobile и MapQuest Mobile.

Обычно клиенты картографических веб-приложений создаются для работы с одним приложением, но существуют и такие, которые позволяют работать сразу с несколькими различными картографическими веб-сервисами. Подобные клиенты называются геобраузерами. Геобраузерами называют визуализаторы карт, с помощью которых можно просматривать стандартные картографические веб-службы и данные, доступные на разных ресурсах сети Интернет [12]. Они часто представляют собой настольные приложения. В качестве примеров геобраузеров можно привести Carbon Project Gaia [20], ArcGIS Explorer Desktop, SASPlanet [34].

Настольные ГИС (например, ArcGIS Desktop, QGIS и др.) тоже можно считать геобраузерами, поскольку они позволяют просматривать и запрашивать разные веб-службы, соответствующие стандартам OGC.

Еще одним популярным видом клиента веб-ГИС является виртуальный

глобус. Виртуальный глобус – это трехмерная программная модель (представление) Земли или других миров. Он позволяет перемещаться в виртуальной среде, меняя точку и направление зрения. Многие из таких глобусов берут данные из Веба и поэтому называются онлайновыми виртуальными глобусами.

Онлайновые виртуальные глобусы, как правило, можно использовать бесплатно. Популярными их примерами являются Google Earth, NASA World Wind, Google Earth, ArcGIS Explorer Desktop, Microsoft Bing Maps и др. Большинство интерактивных виртуальных глобусов являются также и геобраузерами, поскольку на отображаемую в них поверхность можно накладывать разнообразные стандартизированные источники пространственных данных и отображать их.

2.4.3.1. Тонкие и толстые клиенты

При проектировании картографических веб-приложений необходимо учитывать несколько факторов, влияющих на их качество и быстродействие. Эти факторы связаны с нагрузкой, которую испытывают ГИС-сервер (при обращении множества пользователей), база пространственных данных (при совершении частых операций чтения и записи данных) и интернет-соединение (при передаче больших объемов данных). Также важным фактором является ограниченная ГИС-функциональность браузерного клиента. Существуют различные пути решения перечисленных проблем, включающие оптимизацию веб-служб, например за счет предварительного кэширования, резервирование, балансировку нагрузки и эффективное использование соединения с Интернетом, правильное распределение рабочей нагрузки между клиентом и сервером.

Основное соображение при проектировании клиент-серверного приложения – распределение рабочей нагрузки между клиентом и сервером. В зависимости от этого различают архитектуры веб-приложений с тонким и толстым клиентом.

В архитектуре с тонким клиентом большая часть работы возлагается на сервер, а клиенту остается меньшая часть. Клиент только посылает запросы пользователя на сервер, который выполняет всю обработку. Применимо к картографическим веб-приложениям сервер создает карту, выполняет анализ данных и геообработку. Далее сервер генерирует веб-страницу в формате HTML, в которую встраиваются изображения карты в графических форматах (png, jpeg, gif), и возвращает ее клиенту для отображения. Таким образом, например, работают картографические веб-приложения, использующие MapServer.

Преимущество такого подхода заключается в снижении требований к производительности компьютеров пользователей, поскольку ресурсоемкие и сложные («тяжелые») операции выполняются сервером. Кроме того, помимо веб-браузера, пользователю не нужно устанавливать никакого дополнительного программного обеспечения, даже модуля к браузеру.

Основными недостатками использования архитектуры с тонким клиентом для картографических веб-приложений являются, во-первых,

большая нагрузка ГИС-сервер, во-вторых, ограниченная интерактивность клиентской части приложения, поскольку ее интерфейс в таком случае строится с помощью простого HTML с ограниченным использованием JavaScript.

Толстый клиент является противоположностью тонкому. Архитектура с толстым клиентом возлагает выполнение большинства функций на клиента, а не на сервер. Достигается это обычно с помощью подключаемого модуля веб-браузера или настольного клиентского приложения, которые выполняются локально на клиентском компьютере. Толстый клиент запрашивает исходные данные с сервера, а затем самостоятельно отрисовывает карту, данные или выполняет их анализ.

Положительная сторона этой архитектуры состоит, во-первых, в быстром взаимодействии с пользователем, поскольку программа выполняется локально на компьютере пользователя. При этом данные также могут размещаться локально. Вторым положительным моментом является меньшая нагрузка на сервер, связанная с уменьшением числа циклов «запрос-ответ» между клиентом и сервером.

К недостаткам архитектуры с толстым клиентом можно отнести неудобства, связанные с установкой браузерных модулей подключения или настольных приложений. Например, политика компании в целях безопасности может запрещать установку модулей и приложений. Также недостатками являются возможные ограничения передачи больших объемов данных через Интернет, связанные с «шириной» канала связи, и ограничения, связанные с вычислительной мощностью компьютера клиента. Так, не на каждом клиенте возможно реализовать сложные операции геообработки.

Учитывая все перечисленные преимущества и недостатки каждой из архитектур, можно сделать вывод о том, что при проектировании картографических веб-приложений необходим некий баланс в распределении нагрузки между сервером и клиентом. Так, более простые ГИС-операции должны выполняться на клиенте, а более сложные – на сервере. Популярная стратегия проектирования картографических веб-приложений предусматривает деление нагрузочных операций на несколько категорий.

Согласно этой практической стратегии, составляющие обычного картографического веб-приложения рекомендуется разделять на базовые карты, рабочие слои и инструменты [12].

Базовые карты обычно заранее отрисовываются сервером или клиентом. Опыт показывает, что отображение базовых карт лучше делать на сервере. Базовая карта, как правило, сравнительно статична и в обычных условиях обновляется редко. Она обычно служит картографической основой для отображения рабочих тематических слоев. Для обеспечения более быстрой работы картографического сервиса базовые карты следует кэшировать, т. е. сервер должен заранее отрисовывать и сохранять изображения карт заданных масштабных уровней, чтобы потом их можно было быстро визуализировать, просто передавая эти изображения клиенту.

Данные рабочих слоев обычно невелики по объему или отображаются только при определенных масштабах. Поток данных направляется клиенту, который затем осуществляет их визуализацию и управляет ими (в отличие от базовых слоев, которые обычно визуализирует сервер). Это связано с тем, что рабочие слои, как правило, являются динамическими, т. е. достаточно часто подвергаются изменениям, и поэтому они не могут предварительно кэширо-

ваться. Также пользователям нужно взаимодействовать с этими слоями для выполнения с ними различных операций. Им нужно, чтобы эти слои обеспечивали быстрый отклик, показывая дополнительную информацию по щелчку мыши, могли редактироваться через веб-интерфейс, использовались при операциях геообработки и т. д. Динамическая отрисовка и взаимодействие с пользователем возлагаются на клиента и могут быть реализованы с помощью браузерных API, таких как интерфейсы ArcGIS для JavaScript, Flex, Silverlight, библиотеки Leaflet, OpenLayers. Иногда объем данных рабочего слоя слишком велик для эффективной визуализации браузером. В этом случае данные должны визуализироваться сервером.

Что касается рабочих инструментов геообработки, то сравнительно простые операции выполняются клиентом, а сложные – сервером. При этом для выполнения процессов обработки данных на клиенте все необходимые данные также должны находиться на нем, а при запуске инструментов на сервере – соответственно, на самом сервере.

Типичными примерами простых операций являются графическое отображение результатов анализа и интерполяция данных на основе набора точек. К сложным операциям можно отнести поиск ближайшего объекта определенного типа и построение кратчайшего маршрута до него, расчет речного стока, нахождение наилучшего места для размещения какого-либо объекта путем наложения ряда слоев данных и т. д.

2.5. Геопорталы

Большая часть картографических веб-приложений и сервисов, существующих в сети Интернет, имеет какую-либо определенную тематику (например, экологическая, гидрологическая, планы городов, справочники организаций и т. п.) и определенный перечень возможностей для работы с картой, включающий просмотр данных, их простой анализ и т. д. Однако существует и особая разновидность картографических веб-приложений, называемых геопорталами. Они имеют более широкую функциональность по сравнению со стандартными картографическими веб-приложениями. Геопорталы способны концентрировать в себе множество веб-служб и пространственных данных как разной, так и единой тематики и предоставлять функциональные возможности для осуществления поиска среди них, для их просмотра, получения и загрузки, анализа. Также геопорталы предоставляют ресурсы для разработки других картографических веб-приложений и т. д.

Геопортал – это веб-сайт, который обеспечивает единую точку доступа к пространственным данным, веб-службам и другим географически привязанным ресурсам.

В основу классификации геопорталов обычно положены территориальный и тематический принципы.

По территориальному охвату геопорталы делятся на глобальные, национальные (федеральные), региональные и локальные. Часто геопорталы, особенно национального и регионального охвата, могут напоминать национальные и региональные комплексные атласы. При этом спектр их возмож-

ностей, конечно же, значительно шире, чем у их традиционных печатных аналогов.

По тематике геопорталы могут быть универсальными или иметь определенную специализацию, такую как климат, транспорт, реагирование на чрезвычайные ситуации и др.

В качестве примеров геопорталов можно привести геопорталы поиска и заказа данных дистанционного зондирования Земли (серверы NASA, DigitalGlobe, Геопортал Роскосмоса, Космоснимки.ру и др.), региональные геопорталы субъектов России (Геопортал Республики Коми, Геопортал Республики Татарстан и др.), федеральные тематические геопорталы (Публичная кадастровая карта, Федеральная государственная информационная система территориального планирования (ФГИС ТП) и др.).

Геопорталы способствуют передаче пространственной информации от поставщиков, которые ей владеют или распоряжаются, к потребителям, которые в ней нуждаются. Поставщики пространственных ресурсов публикуют на геопорталах метаданные (данные о данных), т. е. описания пространственных наборов данных, веб-служб и документов и других ресурсов. Потребитель может просматривать содержимое геопорталов или осуществлять на них поиск с учетом заданных критериев (ключевые слова, тип содержимого, формат, пространственный экстент, информация о местоположении, временной интервал), чтобы обнаружить относящиеся к его теме пространственные ресурсы и оценить, насколько они подходят для решения поставленной задачи. Затем пользователь может загрузить онлайн-данные, подключиться и использовать веб-службы или обратиться к поставщику с просьбой о предоставлении данных. Качественные геопорталы позволяют искать содержимое не только по ключевым словам, но и по пространственно-временным характеристикам информации, например: по заданной пользователем области пространства, моменту или интервалу времени [12].

Метаданные обычно предоставляются в формате XML и включают в себя идентификационную информацию (имя набора данных, его краткое описание, назначение и ключевые слова), информацию о качестве (точность позиционирования, полнота данных и их связность), информацию о пространственной привязке (система координат и пространственный экстент), временную информацию (дата получения данных и срок их годности).

Для удобства работы с метаданными, выполнения их автоматической обработки и осуществления поиска по заданным критериям необходимы их стандартизация и приведение к общему формату. Международным информационным сообществом определен ряд стандартов метаданных.

Так, существует стандарт под названием «Дублинское ядро» (по названию города Дублина (штат Огайо), где в 1995 г. он был создан). Это простой стандарт для описания информационных ресурсов, относящихся к различным областям знаний. Дублинское ядро утверждено Международной организацией по стандартизации (ISO) как стандарт ISO 15836. Его главным преимуществом является простота и легкость использования. Спецификация имеет два уровня: простой и компетентный. Простое Дублинское ядро состоит всего из 15 элементов в 3 группах, которые описывают ресурс: название, создатель, тема, описание, тип, дата, формат и др. Компетентное Дублинское ядро добавляет 3 дополнительных элемента (аудитория, происхождение и правообладатель) и группу квалификаторов элементов, которые уточняют

их семантику. Дублинское ядро – самый популярный стандарт метаданных, широко использующийся для описания многих видов ресурсов, включая пространственные. Однако из-за ограниченного набора элементов он часто нуждается в расширении [12].

Также международное сообщество в лице Технического комитета 211 ISO разработало ряд международных стандартов для представления географической информации и метаданных. Стандартом ISO 19115 определяется схема для описания географических наборов данных, а стандартом 19119 – веб-служб. В словаре данных стандарта насчитывается более 400 элементов метаданных. Эти два стандарта являются стандартами содержания, а в стандарте ISO 19139 приведен общий способ XML-кодирования метаданных, необходимый для реализации стандартов содержания [12].

Большинство стандартов метаданных сложны, поскольку они должны учитывать характеристики всех видов пространственных ресурсов. Хотя такие инструменты, как ArcGIS Desktop, помогают создавать метаданные, соответствующие стандартам ISO и пр., процесс этот остается непростым для многих публикаторов пространственных данных.

С распространением концепции Веб 2.0 обмен данными уже происходит не только между представителями научного и профессионального сообщества. С увеличением легкости передачи данных возникает потребность в облегчении их описания, чтобы потенциальные потребители могли их находить. В связи с этим появляется необходимость в создании так называемых метаданных 2.0, ориентированных на пользователя, легких в создании, понятных для всех [12].

Движение в направлении метаданных 2.0 отражается в ставших в последнее время популярными тегах и хештегах. Тег и хештег – это ключевые слова или термины, которые описывают объект, такой как фотография, видео или инструмент. Они активно используются в социальных сетях, видеохостингах и других веб-ресурсах. Тем самым конструируются гибкие и легкие метаданные, по которым можно легко находить нужные фото- и видеоматериалы. Однако такие метаданные часто неформальны и неполны с профессиональной точки зрения, в то же время они позволяют пользователям оперативно делиться информацией, а другим легко находить и оценивать ресурсы с точки зрения пригодности для их целей.

В зависимости от области применения могут использоваться как стандартизированные типы метаданных, так и простые метаданные 2.0. Стандартизированные метаданные важны для описания профессионалами и организациями их геоинформационных ресурсов. В то же время для поиска и оценки применимости этих ресурсов другими профессионалами метаданные 2.0 поощряют участие широкого круга пользователей в этом процессе и поэтому во многих ситуациях могут также использоваться в геопорталах.

Из вышенаписанного следует, что база метаданных (каталог) является основным компонентом любого геопортала. Она может реализовываться в распределенной или централизованной архитектуре (рис. 2.9). Геопортал с распределенным каталогом для выполнения поискового запроса пользователя должен выполнять распределенные операции поиска во многих каталогах, передавая им этот запрос. В геопортале с централизованным каталогом все записи метаданных содержатся в одном хранилище. Каждый из каталогов участников может загружаться в центральный каталог посред-

ством периодического сбора (синхронизации) метаданных. Поставщик может зарегистрировать свой каталог на геопортале и указать частоту, с которой геопортал должен проверять обновления его базы метаданных.

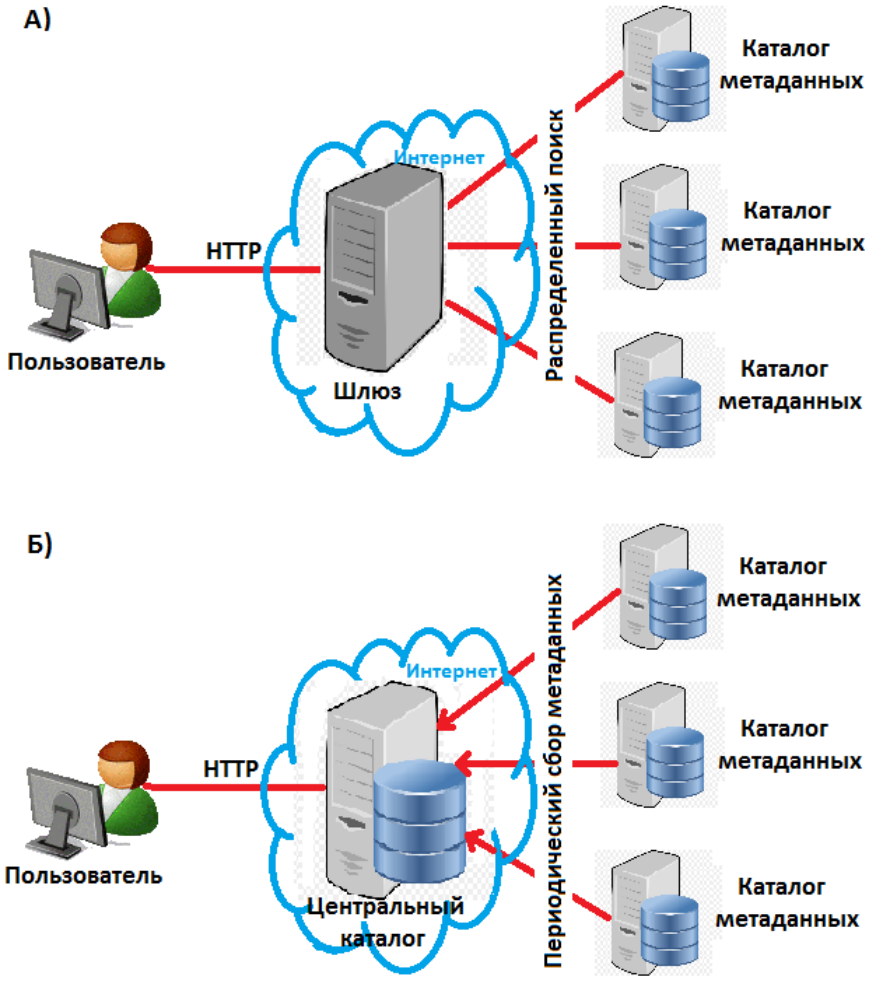

Рис. 2.9. Варианты организации базы метаданных геопортала: А) с распределенной архитектурой; Б) с централизованной архитектурой

3. ПУБЛИКАЦИЯ ПРОСТРАНСТВЕННЫХ ДАННЫХ

Как было отмечено выше, первым массовым и доступным программным решением для публикации пространственных данных в сети Интернет являлся MapServer [27]. В настоящее время существует множество продуктов (как серверных, так и пользовательских ГИС), обеспечивающих подготовку пространственных данных в различных форматах, пригодных для непосредственного использования на стороне веб-браузера в целях отображения картографической информации. Некоторые пользовательские (настольные) ГИС, например QGIS [33], предоставляют ограниченные возможности для публикации отдельных слоев или даже полных карт (картографических проектов) в виде набора HTML-страниц и вспомогательных файлов. Подготовленный таким образом набор файлов может быть открыт в любом стандартном браузере для получения веб-карты с включением графики и минимальных средств для просмотра.

Наибольшую гибкость, функциональность и масштабируемость при публикации пространственных данных разного типа и в разных форматах (включая стандартные протоколы для передачи данных по сети) обеспечивают специализированные серверные решения, называемые обычно ГИС-сервером. Основная задача ГИС-сервера состоит в формировании картографических изображений и их передаче по запросам, поступающим от соответствующего веб-сервера, в виде графических файлов или потоков данных. Ниже будут описаны два таких сервера: коммерческий ArcGIS Server и свободно распространяемый продукт с открытыми исходными кодами GeoServer.

3.1. Публикация пространственных данных средствами ArcGIS Server

ArcGIS Server представляет собой кросс-платформенное решение, что обеспечивается использованием JRE (Java Runtime Environment, или среда исполнения Java-программ). За счет этого ArcGIS Server может быть установлен на серверах под управлением операционных систем Windows и Linux (использование других операционных систем компанией ESRI не поддерживается). Ниже (разд. 3.1.1) описан процесс его установки под Linux, при этом все последующие действия по настройке и публикации с точки зрения пользователя выполняются одинаково и не зависят от платформы.

Разработка картографических веб-приложений с использованием программных средств и технологий компании ESRI главным образом базируется на использовании ArcGIS Server. Краткое описание данного программного продукта представлено в разд. 2.4.1. Основным назначением ArcGIS Server является публикация пространственных данных и инструментов геообработки в локальной или глобальной сети, а также предоставление доступа к ним в виде сервисов. ArcGIS Server может использоваться как в рамках портала ArcGIS Enterprise (Portal for ArcGIS), где он является главным компонентом, так

и в качестве отдельного продукта (т. е. автономно). Портал ArcGIS Enterprise позволяет обеспечить общий доступ ко всем ресурсам, созданным в одной организации, при помощи отдельного сайта и настольных программных продуктов ArcGIS. Автономное использование ArcGIS Server встречается чаще всего, при этом сервер используется как поставщик ресурсов и данных.

3.1.1. Особенности установки ArcGIS Server

Руководство по установке и развертыванию ArcGIS Server на компьютерах и серверах под управлением операционных систем Windows и Linux представлено в справке на веб-ресурсах компании ESRI [4]. При выборе программной архитектуры следует иметь в виду:

- что реальное решение на базе ArcGIS Server представляет собой не единый программный продукт, а сборку из взаимодействующих компонентов, на версии которых могут накладываться жесткие ограничения;
- при обновлении версий (что является обязательным требованием для льготных программ лицензирования) требуется одновременное обновление всех компонентов от компании ESRI, включая ArcGIS Desktop;
- новые версии ArcGIS Server, как правило, требуют обновления сторонних компонентов – в частности, перехода на новые версии систем управления пространственными базами данных, среды исполнения Java и т. п.

Операционная система Windows для компании ESRI, безусловно, является приоритетной платформой (пользовательские продукты ArcGIS Desktop разрабатываются только для Windows). В связи с этим процесс установки серверных продуктов (ArcGIS Server и сопутствующих компонентов) здесь в значительной мере автоматизирован. Для Linux процесс установки разбивается на ряд этапов, реализация которых выполняется вручную и зависит как от используемой версии Linux, так и от конкретного набора компонентов. Несмотря на эти сложности, использование Linux может оказаться предпочтительней с точки зрения стоимости лицензий на операционные системы Windows Server (зависящей в числе прочего от количества процессорных ядер и оперативной памяти), а также общей управляемости программного комплекса. Ниже рассмотрены моменты, касающиеся установки ArcGIS Server на ОС Linux.

Операционная система. Согласно данным компании ESRI на странице загрузки, для версии ArcGIS Server (на момент установки это 10.7.1, последняя версия – 10.8) были успешно протестированы такие разновидности серверных вариантов Linux, как Ubuntu (LTS 18.04, LTS 16.04), CentOS Linux (7, 6), Scientific Linux (7, 6), Oracle Linux (7, 6), Red Hat Enterprise (7, 6) и SUSE Linux Enterprise (12). В частности (описываемые ниже технические детали и версии продуктов соответствуют установке, выполненной авторами данного пособия), использован open source дистрибутив CentOS Linux 7.0, обеспечивающий бинарную совместимость с коммерческим дистрибутивом Red Hat

Enterprise Linux Server 7.0, традиционно используемым компанией ESRI. Для установки ArcGIS Server с нуля можно рекомендовать подходящий дистрибутив Ubuntu Server LTS, предоставляющий, возможно, наибольший уровень поддержки среди некоммерческих версий Linux.

Среда исполнения ArcGIS Server. Программная архитектура предусматривает использование *среды исполнения JRE* (Java Runtime Environment) и *контейнера сервлетов* Apache Tomcat (что очень близко к архитектуре реализации GeoServer, описанной в разд. 3.2.1). Для ArcGIS Server 10.7.1 в качестве среды исполнения достаточен свободный пакет JDK (Java Development Kit) в версии 1.8 и контейнер сервлетов Apache Tomcat подходящей версии. Установка пакетов осуществляется последовательно с использованием стандартных средств операционной системы CentOS и подробно здесь не рассматривается.

Основной компонент ArcGIS Server. Этот компонент загружается из личного кабинета клиента ESRI в виде архива, который разворачивается в домашнем каталоге выделенного пользователя ags. Установка пакета и его последующее выполнение также осуществляется под именем непривилегированного пользователя ags, что обеспечивает определенный уровень безопасности для операционной системы. Установка производится в режиме командной строки с использованием файла авторизации с расширением .prvc. Файл может быть сгенерирован в личном кабинете или получен по запросу от службы технической поддержки. Команда установки

```
./Setup -m silent -l yes -a authorization.prvc
```

явно задает путь к файлу авторизации. После успешной установки (может длиться до 20 мин. и дольше) управление сервером может осуществляться в графическом режиме через веб-интерфейс менеджера

```
https://<host>:6443/arcgis/manager
```

где в качестве параметра <host> может использоваться доменное имя или IP-адрес. При первом обращении по этому адресу запускается ArcGIS Server Setup Wizard и выполняются первичные настройки, в частности задаются имя нового сайта, имя администратора и его пароль. В случае каких-либо проблем с авторизацией появится сообщение об отсутствии авторизации, а просмотреть список авторизованных продуктов (компонентов и расширений) и сроки действия их лицензий можно с помощью команды

```
/authorizeSoftware -s
```

Просмотр состояния, остановка, запуск и перезапуск сервера осуществляются также от имени пользователя ags командами

```
systemctl status arcgisserver.service
systemctl stop arcgisserver.service
systemctl start arcgisserver.service
systemctl restart arcgisserver.service
```

Более подробные инструкции по установке ArcGIS Server и сопутству-

ющих компонентов представлены по ссылке [4].

Компонент ArcGIS Web Adaptor. Этот компонент обеспечивает интеграцию существующего веб-сайта с одним или несколькими экземплярами ArcGIS Server с одновременным повышением безопасности за счет скрытия от внешнего пользователя элементов внутренней структуры. Установка производится в режиме командной строки из распакованного архива. Команда установки с опциями по умолчанию имеет вид (при обновлении версии сначала удаляется предыдущая):

```
./Setup -m silent -l yes
```

Последующее после установки конфигурирование Web Adaptor выполняется в режиме командной строки с параметрами, указанными согласно [4].

Система управления пространственными базами данных. В качестве таковой целесообразно использовать наиболее развитую в функциональном отношении некоммерческую СУБД PostgreSQL (с открытыми исходными кодами) с расширениями, превращающими ее в пространственную СУБД. Заметим, что компания ESRI предлагает применять специальные сборки – версии PostgreSQL, скомпилированные уже с расширениями и доступные для загрузки также в личном кабинете. Для ArcGIS 10.7.1 минимальной версией является PostgreSQL 9.6.12 (PostGIS 2.3), допустимы также PostgreSQL 10.7 (PostGIS 2.4) и PostgreSQL 11.2.8 (PostGIS 2.5.1). Целесообразно обращаться к самой новой версии в расчете на то, что ее пригодность сохранится максимально долго при последующих обновлениях. В данном случае выбираем версию PostgreSQL 10.7 (PostGIS 2.4), используемую в других проектах, в частности совместно с GeoServer.

После загрузки и распаковки архива с выбранной версией PostgreSQL процесс установки инициируется командой

```
sudo ./postgresql-10.7-1-linux-x64.run
```

выполняемой с полномочиями суперпользователя. В интерактивном режиме задается пароль администратора postgres, устанавливаются дополнительные инструменты командой строки и приложение pgAdmin 4 для управления СУБД через веб-интерфейс. Первоначальные настройки предусматривают автоматический запуск сервера PostgreSQL при загрузке операционной системы и *только локальный* доступ к серверу. Как правило, требуется ограниченный удаленный доступ и с других машин, что регулируется соответствующими настройками. В этом отношении используемый вариант PostgreSQL ничем не отличается от стандартного, поэтому детали конфигурирования здесь не рассматриваются. Соответствующие руководства легко найти на сайте проекта [32].

Следует иметь в виду, что процесс перехода на другую версию PostgreSQL (что фактически становится обязательным каждые 1–2 года) должен сопровождаться подготовкой резервных копий (бэкапов) всех баз геоданных на старой версии и их последующего восстановления на новой версии. Создание и восстановление бэкапов выполняется средствами командной строки из соответствующих версий PostgreSQL. Пример создания бэкапа для условной базы геоданных gdb:

```
pg_dump -U postgres -W -h localhost -Fc gdb > gdb.dump
```

Пример восстановления:

```
pg_restore -U postgres -W -h localhost -d gdb gdb.dump
```

Следует также иметь в виду, что при восстановлении на уровне отдельных баз данных их владельцы, если они отличаются от postgres, должны быть заранее созданы перед восстановлением.

Обновление ArcGIS Server и сопутствующих компонентов при льготных условиях лицензирования приходится выполнять ежегодно, и фактически данный процесс (подробно описанный по ссылке [4]) сводится к последовательной переустановке компонентов всего комплекта в порядке **удаление предыдущей версии ⟶ установка новой версии**.

3.1.2. Компоненты ArcGIS Server и их настройка

При установке ArcGIS Server создается так называемый сайт ArcGIS Server. Основной частью, ядром сайта ArcGIS Server является непосредственно сам ГИС-сервер (ArcGIS Server). Он производит графическое отображение карт, запуск инструментов, обработку запросов к данным и выполняет любые другие действия, доступные в виде сервисов. При этом ГИС-сервер ArcGIS Server способен предоставить доступ к сервисам сразу после установки по стандартному протоколу HTTP с использованием порта 6080 (по умолчанию). На один сайт ArcGIS Server может быть добавлено несколько ГИС-серверов, установленных на разные компьютеры. Это достигается благодаря тому, что при установке ГИС-сервера существует возможность как создания нового сайта ArcGIS Server, так и присоединения к существующему сайту.

Помимо ГИС-сервера, сайт ArcGIS Server имеет два других важных компонента: хранилище конфигурации (configuration store) и директории сервера (server directories). Хранилище конфигурации – это директория, которая содержит все основные свойства сайта и его сервисов (например, информацию о веб-сервисах, пользователях, ролях, данных и настройках безопасности). На сайте, имеющем несколько компьютеров с ГИС-серверами, они получают доступ к хранилищу конфигурации из общей сетевой директории.

Директории сервера представляют собой физические директории в сети, специально используемые сайтом ArcGIS Server для хранения и записи определенного вида информации. Существуют директории сервера для хранения кэша, выходных данных, задач, системных файлов, загрузок, входных данных, KML-файлов и индексов. На сайте, содержащем несколько компьютеров с ArcGIS Server, это должна быть общая сетевая директория.

Как уже было сказано, для интеграции ArcGIS Server с имеющимся веб-сервером (Apache, IIS, WebSphere, WebLogic и др.) используется Web Adaptor. Он облегчает обмен данными между веб-сервером и ArcGIS Server, давая ряд дополнительных преимуществ: обеспечивает единую точку доступа на сайт

ArcGIS Server через общий URL-адрес (по выбранному администратором порту, отличному от стандартного 6080, и имени сайта). Web Adaptor получает запросы клиентов к картографическим веб-сервисам и отправляет их на серверы, входящие в состав сайта ArcGIS Server. Один Web Adaptor может быть настроен только на один сайт ArcGIS Server, но один сайт ArcGIS Server может быть настроен для многих конечных точек доступа с использованием разных экземпляров Web Adaptor. Также Web Adaptor обеспечивает более высокий уровень безопасности, предоставляя возможность блокировки доступа к функциям администрирования для внешних пользователей (ограничивается доступ к приложениям ArcGIS Server Manager и ArcGIS Server Administrator Directory). Общая архитектура развертывания ArcGIS Server с использованием Web Adaptor представлена на рис. 3.1.

Физически все описанные компоненты можно развернуть и на одном компьютере (в том числе и на обычной локальной машине), что, например, удобно для тестирования приложений при их разработке.

При создании нового сайта ArcGIS Server во время установки программного обеспечения создается учетная запись администратора, которая предназначена для текущей настройки сайта и его дальнейшего администрирования, а также (при необходимости) для настройки нового сайта.

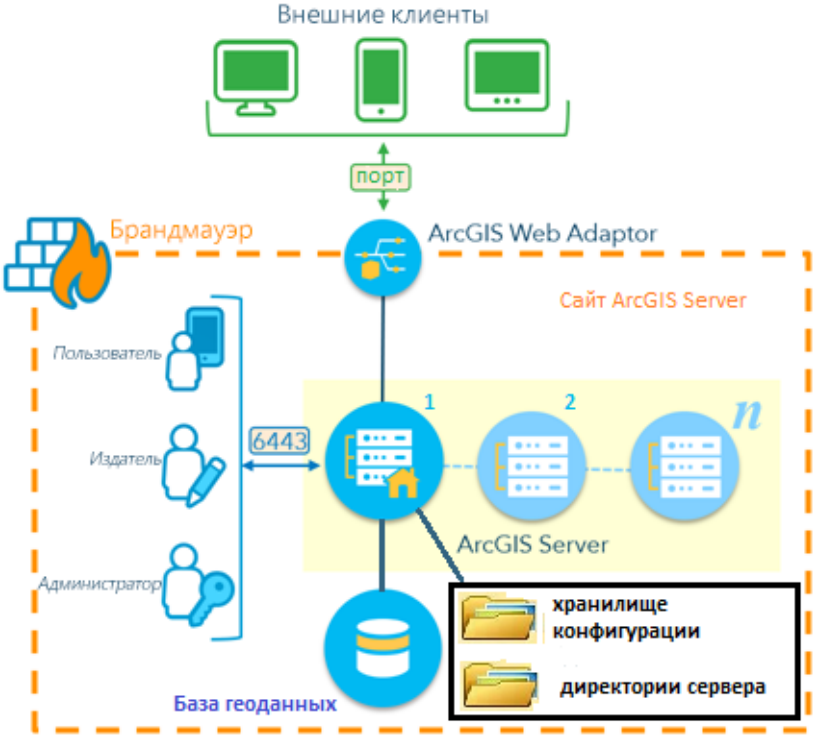

Рис. 3.1. Типичная схема архитектуры ArcGIS Server

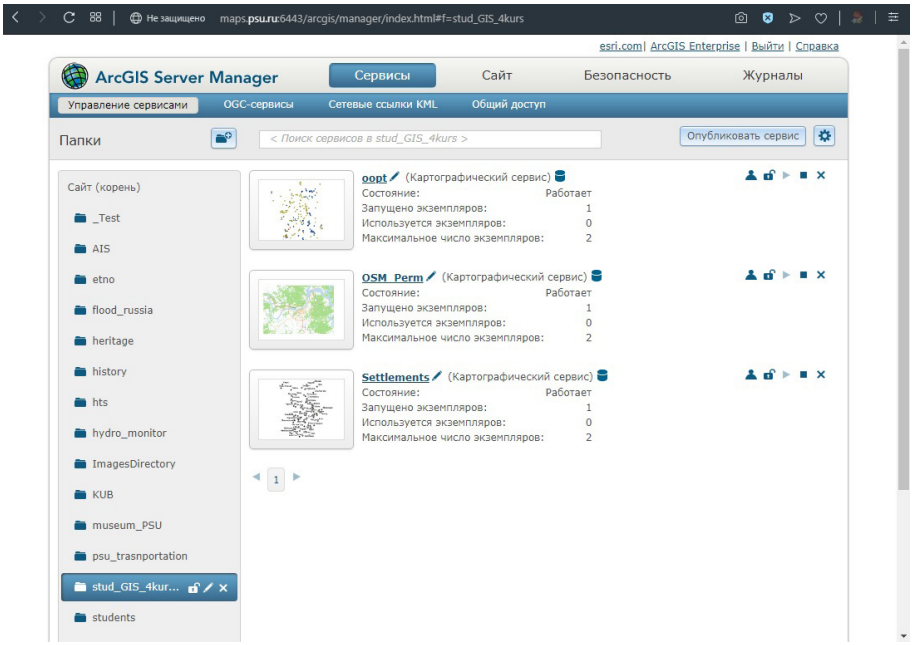

Рис. 3.2. Веб-интерфейс пользователя приложения для администрирования ArcGIS Server Manager

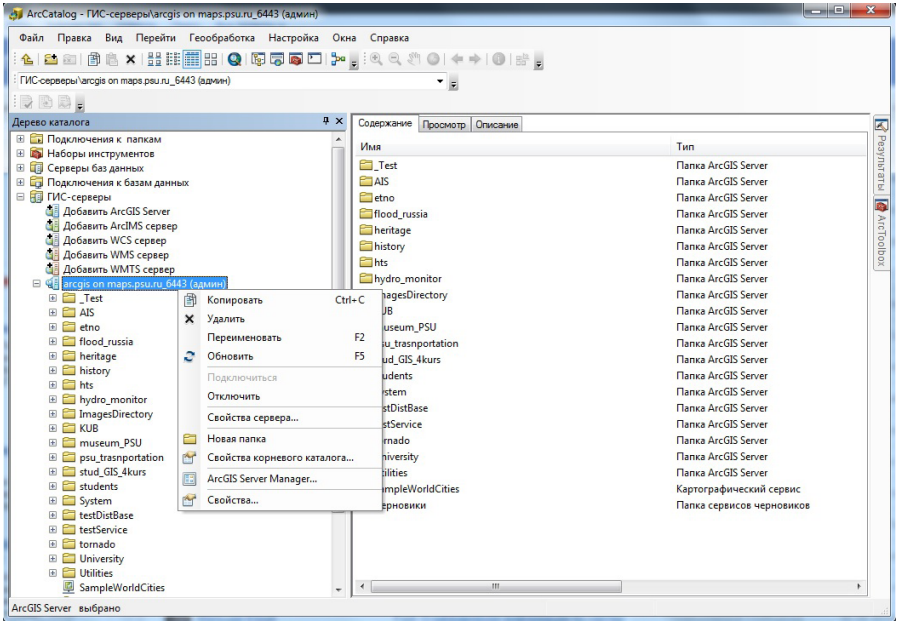

Рис. 3.3. Управление ArcGIS Server через приложение ArcCatalog

После установки ArcGIS Server его администрирование осуществляется через приложение ArcGIS Server Manager, которое имеет интерфейс доступа через веб-браузер по адресу вида https://<host>:6443/arcgis/manager (рис. 3.2), где <host> – адрес сервера. Также часть функционала ArcGIS Server Manager доступна через настольное приложение ArcCatalog при добавлении подключения к ArcGIS Server в дереве каталога в группе **ГИС-серверы → Добавить ArcGIS Server** (рис. 3.3).

В ArcGIS Server Manager можно управлять (запускать, устанавливать, удалять) опубликованными веб-сервисами (службами), также в нем имеются возможности настройки сайта ArcGIS Server (настройка директорий хранения сервисов, настройка хранилища конфигурации, регистрация хранилищ и баз геоданных, авторизация дополнительных модулей и расширений ArcGIS, настройка Web Adaptor и др.), возможности управления безопасностью и просмотра журнала ArcGIS Server.

Кроме того, для установок конфигурации сайта ArcGIS Server используется веб-каталог администрирования ArcGIS Server Administrator Directory, или ArcGIS REST API. Доступ к этому веб-каталогу осуществляется по адресу вида https://<host>:6443/arcgis/admin (рис. 3.4). ArcGIS REST API позволяет создавать скрипты для выполнения стандартных задач администрирования сервера, такие как добавление компьютера (ГИС-сервера) на сайт, публикация сервиса, добавление разрешений и т. д. Веб-каталог ArcGIS Server Administrator Directory предлагает интерактивный доступ к этому API. При помощи данного API можно полностью управлять сайтом ArcGIS Server на языке программирования Python.

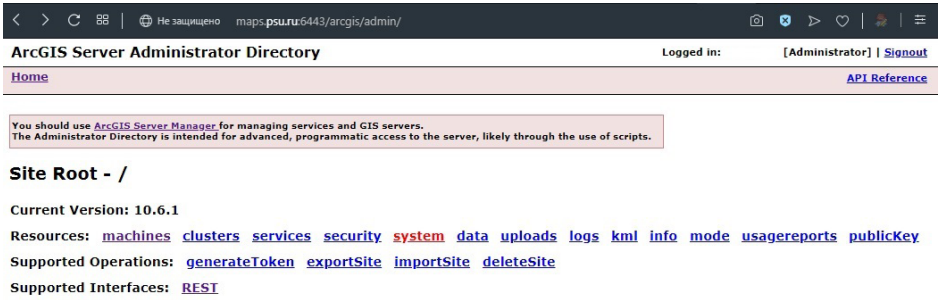

Рис. 3.4. Внешний вид веб-каталога администрирования ArcGIS Server Administrator Directory в окне веб-браузера

Еще одной важной и наиболее часто используемой составляющей сайта ArcGIS Server является каталог сервисов – Services Directory (рис. 3.5). Он открыт для пользователей (т. е. не требует учетных записей для просмотра) и доступен через веб-браузер по URL-адресу формата http://<host>:8080/arcgis/rest/services (для доступа может использоваться протокол https и другой порт, например 8080 или 6443). Каталог сервисов представляет собой веб-приложение, в котором содержится перечень всех опубликованных на ArcGIS Server ГИС-ресурсов в виде веб-сервисов (служб). Среди таких ресурсов могут быть отдельные слои, карты, инструменты геообработки, локаторы адресов и др.

Эти веб-сервисы взаимодействуют с пользовательскими веб-приложениями через архитектуру, основанную на Representational State Transfer (REST). Можно получить информацию и совершить операции на сервере с использованием URL-запросов согласно заданному формату. При использовании Services Directory по адресной строке браузера можно отследить, как разные URL-запросы используются для получения информации с помощью REST. Каждый ресурс, опубликованный на ArcGIS Server, имеет уникальный URL-адрес, который используется для доступа к нему, например из клиентского веб-приложения.

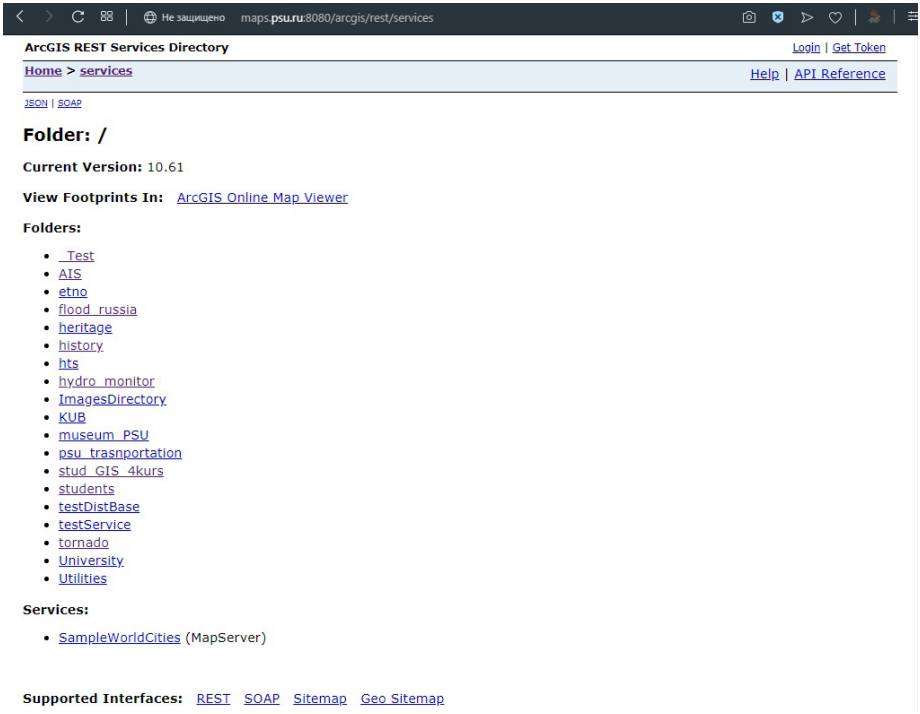

Рис. 3.5. Внешний вид каталога сервисов (Services Directory) ArcGIS Server в окне веб-браузера

3.1.3. Работа с пространственными данными

Хранение пространственных данных, публикуемых в сети Интернет при помощи ArcGIS Server, можно осуществлять в разных хранилищах на сервере. Такими хранилищами могут выступать обычные каталоги, например с набором шейп-файлов, облачные хранилища, каталоги растров, файловые базы геоданных, а также многопользовательские базы геоданных, реализуемые при помощи СУБД. При публикации пространственных данных либо использующих их карт необходима регистрация каталогов и баз геоданных (в том числе многопользовательских) в ArcGIS Server. Это необходимо для того,

чтобы избежать повторного копирования данных на сервер при их публикации, а также для реализации возможностей редактирования геоданных в настольной ГИС без их повторной публикации (т. е. зарегистрированные данные после внесения в них изменений автоматически обновятся и на ArcGIS Server).

Регистрация каталогов и баз данных на ArcGIS Server осуществляется либо в ArcGIS Server Manager в меню **Сайт** → **ГИС-сервер** → **Хранилища данных** (рис. 3.6), либо в приложении ArcCatalog через подключение к ArcGIS Server, где в контекстном меню **Свойства сервера** имеется вкладка **Хранилище данных**.

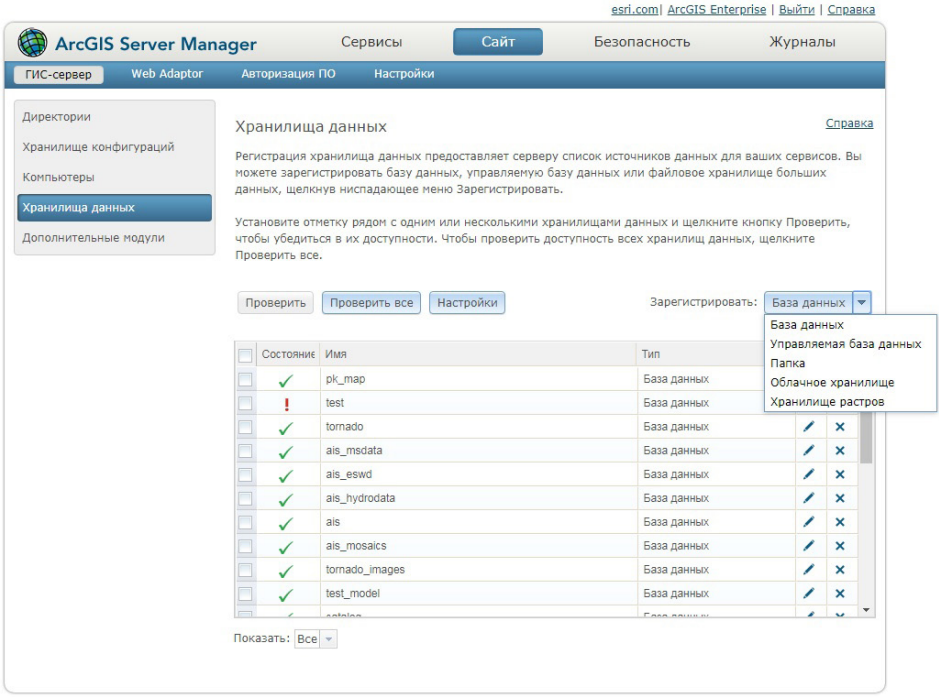

Рис. 3.6. Регистрация хранилищ данных через ArcGIS Server Manager

В связке с ArcGIS Server предпочтительнее использовать многопользовательские базы геоданных, реализованные под управлением СУБД. В ArcGIS Server они называются просто базами данных. К преимуществам их использования относятся: возможность организации одновременного доступа к данным и их редактирования нескольким (многим) пользователям, применение настраиваемых бизнес-правил и отношений внутри базы данных, возможность хранения топологических отношений, геометрических сетей, сетевых наборов данных.

Как упоминалось ранее, ArcGIS использует собственный механизм хранения пространственных данных в многопользовательских базах геоданных внутри СУБД (обычно используются Microsoft SQL Server,

PostgreSQL, Oracle), который называется ArcSDE. Создание средствами СУБД пространственной многопользовательской базы геоданных осуществляется в настольных программах ArcGIS при помощи инструмента геообработки ArcToolbox **Создать многопользовательскую базу геоданных** (группа **Управление данными → Администрирование базы геоданных**). В инструменте необходимо указать СУБД, которая используется, имя сервера, где установлена СУБД, имя базы данных, которая создана внутри СУБД, имя и пароль администратора базы данных в СУБД, имя (обычно sde) и пароль администратора создаваемой пространственной базы геоданных, файл авторизации, содержащий сведения о лицензии ArcGIS Server (рис. 3.7).

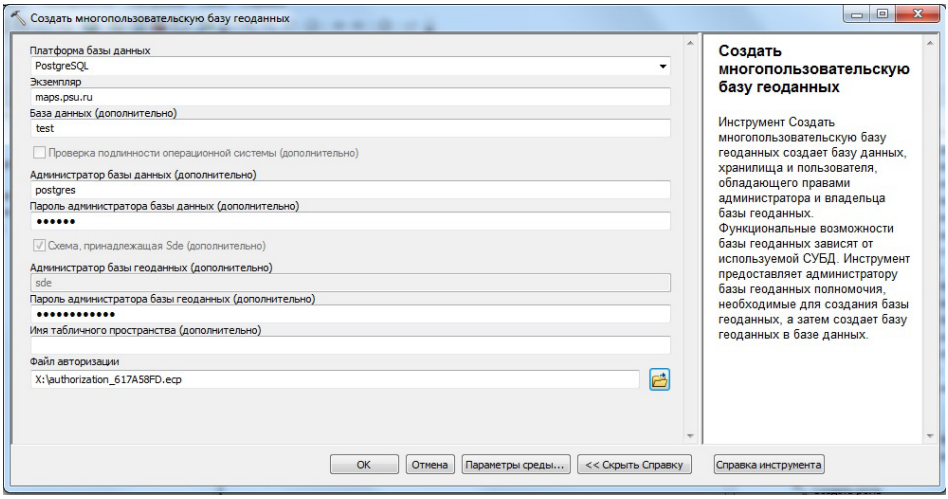

Рис. 3.7. Создание пространственной многопользовательской базы геоданных test внутри СУБД PostgreSQL

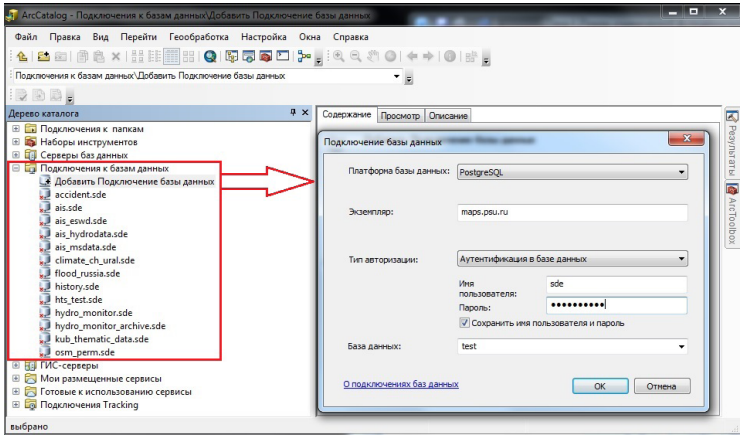

Рис. 3.8. Реализация подключения к многопользовательской базе геоданных через ArcCatalog

После того как многопользовательская база геоданных создана, ее нужно зарегистрировать на ArcGIS Server указанными выше способами. После этого базу можно наполнять пространственными данными из других источников и создавать внутри нее, например, новые классы пространственных объектов, связи и т. п. Для этого необходимо через приложение ArcCatalog создать подключение к этой базе данных в **Дереве каталога** в разделе **Подключения к базам данных** → **Добавить подключение базы данных** (рис. 3.8). После этого база данных будет отображаться в списке подключенных баз и дальнейшая работа с ней в ArcGIS будет похожа на работу с обычной базой геоданных. Еще одним важным моментом, касающимся работы с такой базы данных, является то, что для редактирования средствами ArcMap классов пространственных объектов и таблиц необходима их регистрация как версионных данных, это можно сделать в ArcCatalog в контекстном меню **Управлять** → **Регистрировать как версионный** каждого редактируемого слоя.

Подготовка пространственных данных, карт, инструментов геообработки и т. д., а также их публикация на ArcGIS Server осуществляется через настольные приложения ArcGIS Desktop или ArcGIS Pro.

3.1.4. Процесс публикации данных и сервисов

При работе с пространственными данными и при создании карты для публикации в виде картографического веб-сервиса в приложении ArcMap необходимо предварительно настроить порядок слоев, символы их отображения, задать диапазоны масштабов видимости слоев, если это необходимо, и т. д. Публикация готовой карты осуществляется с использованием меню **Файл** → **Опубликовать как** → **Сервис**. При этом после входа в это меню можно опубликовать новый сервис, либо перезаписать ранее опубликованный сервис, либо сохранить файл определения сервиса, который в дальнейшем можно использовать для его публикации через ArcGIS Server Manager. При выборе публикации нового сервиса далее создается или указывается подключение к ArcGIS Server (если оно было создано ранее в ArcCatalog, то оно будет доступно для выбора), затем выбирается существующая или создается новая папка, где сервис будет опубликован. При этом в этой папке сервис будет доступен после публикации в каталоге сервисов (Services Directory). После выбора папки появится окно редактора сервисов (рис. 3.9), в котором можно указывать и менять ряд параметров для публикации сервиса, включая возможности, которые будут доступны на нем (табл. 3.1).

В этом окне можно отметить, какие возможности будут доступны для сервиса (например, доступ по стандартам OGC, сетевой анализ и др.), можно инициировать создание тайлов для разных масштабных уровней, включить и настроить кэширование сервиса, активизировать возможность создания объектов на карте в формате KML, настроить доступ к сервису в облачной инфраструктуре ArcGIS Online, создать описание сервиса, проверить его на наличие ошибок, предварительно посмотреть, как будет выглядеть карта, и т. д. Затем, когда указаны все необходимые параметры, осуществляется публикация сервиса при помощи опции **Опубликовать**. После публикации сервис становится доступен в Services Directory (рис. 3.5).

Публикация инструмента геообработки на ArcGIS Server осуществля-

ется после его запуска из окна **Результаты** (меню **Геообработка** → **Результаты**), в котором будет зафиксирован этот сеанс работы инструмента. В окне **Результаты** необходимо вызвать контекстное меню сеанса запуска инструмента геообработки и выбрать **Опубликовать как** → **Сервис геообработки**.

Публикация базы геоданных осуществляется из ArcCatalog с помощью ее контекстного меню **Опубликовать как сервис геоданных**.

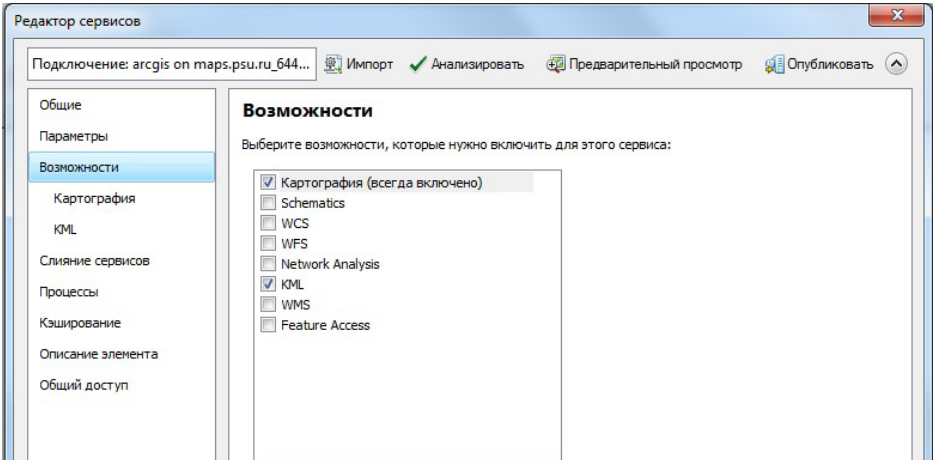

Рис. 3.9. Окно редактора картографического сервиса перед его публикацией

Выше были перечислены только наиболее часто используемые типы сервисов ArcGIS Server, но на самом деле он поддерживает публикацию гораздо большего числа типов сервисов, представленных в табл. 3.1. При этом число поддерживаемых для публикации типов сервисов зависит также от наличия установленных на ArcGIS Server дополнительных модулей (например, ArcGIS Image Server, ArcGIS GeoEvent Server). У каждого типа сервиса есть свой перечень возможностей в окне редактора сервисов при публикации. Более подробно с типами сервисов можно ознакомиться в справке [4].

Отдельно стоит упомянуть сервис геометрии, который предназначен для выполнения картометрических измерений (измерений координат, расстояний, площадей). Сервис геометрии не нуждается в публикации, т. к. он устанавливается вместе с ArcGIS Server по умолчанию и доступен в Services Directory в каталоге **Utilities**. Он используется на всех публикуемых в ArcGIS Server картах.

Ранее упоминалось, что для публикации сервисов на ArcGIS Server и для их использования в ArcGIS Desktop необходимо иметь соответствующую учетную запись. С ее помощью создается подключение к ArcGIS Server из приложения ArcCatalog (меню **Дерева каталога**: **ГИС-серверы** → **Добавить ArcGIS Server**). Существуют разные типы учетных записей (табл. 3.2): администратора, издателя и пользователя, которые имеют разный набор функций при работе с ArcGIS Server.

Таблица 3.1.
Типы сервисов, доступные для публикации на ArcGIS Server

Тип сервиса	Требуемый ГИС-ресурс для публикации	Доступные возможности
Сервис геокодирования	Локатор адресов (.loc, .mxs, пакетный локатор)	1. Доступ к локатору адресов
Сервис геоданных	Файл (.sde) подключения файловой базы геоданных или базы данных к базе геоданных	1. Доступ к содержанию базы геоданных для запросов, извлечения и репликации данных. 2. Создание сервисов Web Coverage Service (WCS), Web Feature Service (WFS), совместимых со спецификацией OGC
Сервис геообработки	Результат геообработки из окна Результаты в ArcGIS Desktop	1. Доступ к моделям геообработки. 2. Создание сервисов Web Processing Service (WPS), совместимых со спецификацией OGC

Тип сервиса	Требуемый ГИС-ресурс для публикации	Доступные возможности
Картографический сервис (кэшированный, динамический)	Документ карты (.mxd)	1. Доступ к содержанию карты, например слоям и их атрибутам. 2. Доступ к векторным объектам на карте (сервис объектов). 3. Организация доступа к данным в формате Keyhole Markup Language (функция KML). 4. Сетевой анализ (необходим модуль ArcGIS Network Analyst) 5. Функции для создания, получения доступа, редактирования и обработки опубликованных схем сетей, а также анализ инженерных сетей (необходим дополнительный модуль ArcGIS Utility Network Management). 6. Доступ к слоям набора данных участков. 7. Просмотр, создание, обновление и редактирование схематических представлений (необходим модуль Schematics). 8. Создание сервисов Web Coverage Service (WCS), Web Feature Service (WFS), Web Map Service (WMS), Web Map Tile Service (WMTS), совместимых со спецификацией OGC

Тип сервиса	Требуемый ГИС-ресурс для публикации	Доступные возможности
Сервис изображений (кэшированный, динамический), доступный при наличии модуля ArcGIS Image Server	Набор растровых данных, набор данных мозаики или файл слоя, ссылающийся на набор растровых данных или набор данных мозаики	1. Доступ к составляющим набора растровых данных или набора данных мозаики, включая значения пикселов, метаданные и каналы. 2. Создание сервисов Web Coverage Service (WCS), Web Map Service (WMS), Web Map Tile Service (WMTS), совместимый со спецификацией OGC
Сервис сцены	3D-сцена из ArcGIS Pro	1. Работа с 3D-ресурсами
Сервис глобуса	Документ глобуса приложения ArcGlobe (.3dd)	1. Работа с 3D-сервисами на уровне глобуса
Потоковый сервис, доступный при наличии модуля ArcGIS GeoEvent Server	Компоненты сервисов дополнительного модуля GeoEvent Server, публикация осуществляется из GeoEvent Manager	1. Предоставление данных клиентам на картографическом сервисе в потоковом режиме
Сервис векторных листов	Пакет векторных листов из приложения ArcGIS Pro (.vtpk).	1. Обеспечение общего доступа и совместного использования векторных листов в пользовательских приложениях
Сервис Workflow Manager	База данных, полученная с помощью инструмента геообработки Создать базу данных Workflow	1. Доступ к возможностям управления рабочими процессами в качестве веб-сервиса

Таблица 3.2.
Типы учетных записей для доступа к ArcGIS Server

Типы подключений ArcGIS Server (Роль)	**Функции**	**Точки подключения к сайту ArcGIS Server**
Администратор	Администрирование сайта ArcGIS Server, публикация веб-сервисов на ArcGIS Server	– ArcGIS Server Manager (полная функциональность); – Services Directory; – ArcGIS Server Administrator Directory (полная функциональность)
Издатель	Публикация веб-сервисов на ArcGIS Server	– ArcGIS Server Manager (ограниченная функциональность); – Services Directory; – ArcGIS Server Administrator Directory (ограниченная функциональность)
Пользователь	Только использование веб-сервисов ArcGIS Server	– Services Directory (если есть права на доступ к ней)

3.2. Публикация пространственных данных средствами GeoServer

Наряду с классическим MapServer [27] значительное распространение среди продуктов с открытым исходным кодом получил GeoServer [22], к особенностям которого относится использование среды исполнения JRE, а также наличие развитого графического интерфейса и возможность визуального контроля публикуемых пространственных данных. Опора на JRE (аналогично рассмотренному выше ArcGIS Server) обеспечивает кроссплатформенность (впрочем, этим же свойством обладает и MapServer, не использующий JRE) и облегчает интеграцию с другими приложениями, выполняющимися в JRE. Дискуссии о преимуществах и недостатках решений на основе MapServer и GeoServer идут до сих пор, но мы в данном пособии ограничиваемся только рассмотрением возможностей GeoServer.

3.2.1. Особенности установки GeoServer

В терминах архитектуры программных систем GeoServer представляет собой *сервлет*, разработанный на языке Java, т. е. специальную программу, требующую для запуска и выполнения наличия сервера – *контейнера сервлетов*, функционирующего в соответствующей *среде исполнения Java* (JRE). С известной долей условности можно говорить, что *контейнер сервлетов* является упрощенной версией *сервера приложений*. В контексте пособия будет использоваться первый термин, соответствующий функциональности программ Apache Tomcat и Jetty, упоминаемых ниже. В качестве среды исполнения Java достаточно использовать минимальный вариант JRE (не путать с расширенным вариантом JDK – Java Development Kit, предназначенным для разработки Java-приложений). Контейнером сервлетов для запуска GeoServer может быть Apache Tomcat, Jetty, JBoss и т. п. Подобная архитектура обеспечивает кроссплатформенность, т. е. возможность запуска одного и того же Java-кода в различных операционных системах, включая Windows, Linux и macOS. Все, что для этого требуется, – установить соответствующую операционной системе версию JRE, установить контейнер сервлетов и загрузить само Java-приложение. Имеются две версии Java-кода приложения GeoServer:

- сервлет, или web archive (файл с расширением .war), предназначенный для запуска в контейнере сервлетов;
- платформно-независимый код (Platform Independent Binary), требующий наличия только среды исполнения JRE, поскольку он уже включает в себя минимальный вариант контейнера сервлетов на основе Jetty.

Последняя стабильная версия программы GeoServer доступна по ссылке http://geoserver.org/release/stable/ (на момент установки – версия 2.17.1 от 08.06.2020; упоминаемые ниже номера версий соответствуют установке, выполненной авторами данного пособия для GeoServer 2.16.2 от 22.01.2020). Здесь представлены варианты Java-приложений, исходный код, архивы с

документацией для пользователя и разработчика, а также пакеты расширений и дополнительные модули (плагины), обеспечивающие повышение функциональности сверх базовой установки, в частности путем добавления форматов входных и выходных пространственных данных.

Вне зависимости от используемого варианта первым шагом всегда является установка (или проверка наличия в системе) нужной версии JRE. Рекомендуется использовать версию 8 от компании Oracle, хотя для установки под Linux пригодна и «свободная» версия JRE из OpenJDK (Open Java Development Kit), поддерживаемая как open source продукт. Ссылки для загрузки приводятся в соответствующих разделах пользовательской документации. После установки JRE можно сразу же использовать версию платформно-независимого кода (Platform Independent Binary), положив файл в нужный каталог используемой операционной системы.

Ранее для операционной системы Windows предлагался упрощенный вариант автоматизированной установки Windows installer, но с версии 16.0 такая возможность не предоставляется. Тем не менее установка GeoServer на локальной машине под управлением Windows достаточно проста и может быть рекомендована для освоения базовых возможностей продукта и тестирования решений на его основе. Для многопользовательской работы целесообразно использовать выделенный сервер на основе свободной ОС (например, на одной из разновидностей Linux или BSD)[1]. Обсуждение деталей установки выходит за рамки данного пособия, хотя предполагается: на нашей рабочей системе установлен GeoServer версии не ниже 2.16.2 и пространственная СУБД PostgreSQL версии не ниже 10.12.

3.2.2. Первоначальные настройки GeoServer

В этом разделе описаны действия, направленные на обеспечение безопасности и реализацию наиболее востребованных возможностей GeoServer, не претендуя на охват всего спектра функциональности, обеспечиваемого как базовыми компонентами в составе Java-приложения GeoServer, так и многочисленными расширениями функций, форматов и сервисов.

Отделение данных от программных компонентов GeoServer. После стандартной установки GeoServer мы получаем файловую структуру каталога geoserver, включающую каталоги data, META-INF, WEB-INF и файл index.html. Уникальные данные и настройки GeoServer сосредоточены в каталоге **data**, который целесообразно изолировать от общей структуры с целью упрощения процесса последующих обновлений. Это можно реализовать копированием или переносом каталога с данными в другое место файловой системы, а доступ к ней организовать путем коррекции переменной окружения GEOSERVER_DATA_DIR в конфигурационном файле ../geoserver/WEB-INF/web.xml. Эти действия рекомендуется выполнить сразу же после установки GeoServer; после перезапуска сервера мы получим рабочее окружение, идентичное первоначальному.

---

1 При работе над данным пособием использовалась система под управлением Ubuntu Server 18.04 LTS (Linux) с установленными на ней программами GeoServer версии 2.2.16.2 и СУБД PostgreSQL 10.12.

Общее описание интерфейса. После запуска сервера доступ к его графическому (административному) интерфейсу осуществляется через веб-браузер по адресу http://<host>:8080/geoserver/ с указанием параметров аутентификации, используемых по умолчанию (login = admin, password = geoserver). Окно (рис. 3.10) включает разделы **About & Status, Data, Services, Settings, Tile Caching, Security, Demos и Tools**, отдельно справа – **Service Capabilities.**

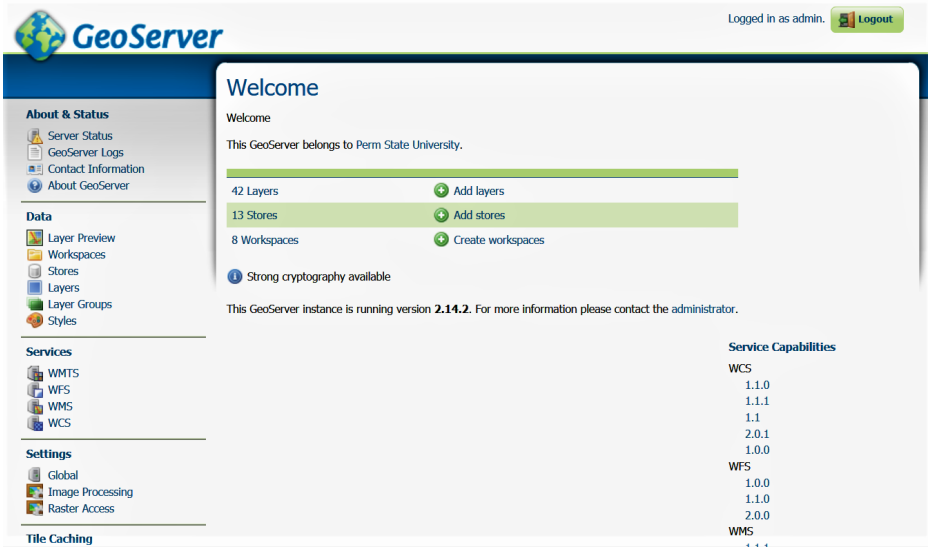

Рис. 3.10. Фрагмент окна GeoServer после подключения с правами администратора

Раздел **About & Status** содержит общую информацию о сервере, включая его версию, технические параметры и текущее состояние, а также информацию о владельце ресурса – организации и контактных реквизитах администратора.

Раздел **Data** является основным при работе с пространственными данными и обеспечивает доступ к следующим возможностям:

- создание рабочих пространств (Workspaces), позволяющих структурировать публикуемые на GeoServer данные по темам и проектам и задавать для них различные права доступа. Фактически рабочее пространство является пространством имен, позволяющим иметь объекты с одинаковыми внутренними именами; уникальность имени обеспечивается сочетанием имени рабочего пространства с внутренним именем объекта. Каждому рабочему пространству соответствует свой файловый каталог в структуре данных. В интерфейсе программы можно применять фильтр по рабочему пространству, ограничивая список объектов только «своими»;
- создание хранилищ данных (Stores) различного типа: векторных слоев в виде шейп-файлов или таблиц пространственных баз данных, растров и мозаик растров в различных форматах,

данных сторонних ГИС-серверов и т. п. В большинстве своем это подключения к источникам пространственных данных – то, с чего начинается практическая работа;

- использование стандартного набора стилей оформления (Styles) или добавление своих стилей, определяющих вид публикуемых пространственных данных. Применение стилей позволяет создавать изображения, имитирующие любые символы, требуемые при создании карт;
- публикация пространственных данных (Layers) в виде слоев, генерируемых с использованием данных из хранилищ (Stores) и заданных стилей оформления (Styles). Опубликованные данные могут быть доступны в виде протоколов (сервисов WMS, WFS, WCS и т. п.) или готовых графических или иных файлов (например, PNG, GeoTIFF, KML, GeoJSON, Shapefile и мн. др.), которые могут быть выгружены по запросу;
- организация опубликованных слоев в виде групп (Layer Groups) с указанием иерархии и условий отображения. Данная возможность имитирует соответствующую функциональность геоинформационных систем и существенно упрощает процесс публикации на веб-серверах;
- просмотр результатов публикации (Layer Preview) и их выгрузка в различных форматах. Для просмотра используются средства стандартной библиотеки OpenLayers, что обеспечивает возможность визуального контроля результатов сразу после публикации.

Разделы **Services, Settings** и последующие включают настройки с заданными по умолчанию параметрами, которые вполне отвечают стандартным вариантам использования. Имеет смысл сразу же после начальной установки актуализировать данные подраздела **Contact Information** и обязательно настроить параметры безопасности.

Настройки безопасности. При первом же входе в систему с параметрами по умолчанию предлагается сменить пароль администратора. Пароль (master password) приходится вводить дважды, т. к. он используется и как пароль администратора (admin), и как средство защиты ключей доступа. Если смена пароля не выполнена сразу, это можно сделать в любой момент через пункт меню **Security → Passwords**.

Также сразу рекомендуется применить средства сильного шифрования, в ответ на приглашение программы, установив значение **Strong PBE** (Password Based Encryption). Данная настройка также может быть выполнена в любой момент через пункт меню **Security → Settings**.

Права доступа к данным регулируются понятием ролей и групп (Roles, Groups) способом, который будет описан в следующем подразделе. Доступ к административному интерфейсу GeoServer (логин) осуществляется с указанием имени и пароля пользователя, который, в свою очередь, может иметь несколько ролей и/или групп с разными правами. В свежей установке системы имеется только пользователь admin с полными правами доступа.

3.2.3. Создание рабочего пространства и настройка окружения

После начальной установки внутри каталога data (в дальнейшем для удобства он будет называться структурой данных GeoServer) с данными и настройками имеется еще один подкаталог data, предназначенный, согласно рекомендациям документации, для хранения файловых геоданных. Просмотр показывает, что здесь хранятся наборы демонстрационных файлов, входящих в поставку GeoServer. Целесообразно создать здесь же[2] подкаталог common для общих файловых данных, относящихся к группе создаваемых проектов.

Для создания и публикации слоев общих данных следует также сформировать рабочее пространство common. Это позволит нам отделить эти данные от прочих, задав соответствующие права доступа для роли, выделенной под работу с ними. Для этого в разделе Workspaces основного меню достаточно ввести имя common, а в качестве URI (Uniform Resource Identifier) указать значение типа http://<host>/common/[3], после чего в структуре данных GeoServer появится подкаталог workspaces/common/, в котором будут храниться конфигурационные файлы с настройками.

При наличии большого количества пространственных данных и пользователей с правом доступа к веб-интерфейсу GeoServer может потребоваться введение ограничений как на чтение, так и на изменение публикуемых слоев данных. В качестве примера ниже показаны создание и настройка двух ролей: READER с правом просмотра всех данных и COMMON_CREATOR с правом редактирования рабочего пространства common. Общий порядок действий описан ниже.

1. Через пункт меню **Security → Users, Groups, and Roles → Roles → Add new role** следует добавить две указанные выше новые роли.

2. Через пункт меню **Security → Data** заменить правило

```
*.*.r for * (all roles)
```

на

```
*.*.r for ADMIN, GROUP_ADMIN, READER
```

В результате правило по умолчанию «доступ на чтение всего для всех» (первая часть до точки указывает рабочее пространство, вторая – слой пространственных данных) заменяется на правило, предоставляющее «доступ на чтение всего» только для явно указанных ролей.

3. Через пункт меню **Security → Data → Add new rule** создать правила для рабочего пространства common и новой роли **COMMON_CREATOR**, последовательно вводя параметры:

---

2 При создании новых подкаталогов нужно указывать в качестве владельца пользователя, от имени которого выполняется GeoServer. Возможность записи в этот каталог пространственных данных (шейп-файлов, растровых данных и т. п.) должна обеспечиваться соответствующими правами доступа средствами используемой операционной системы.

3 URI не обязательно должен соответствовать реальной ссылке, для GeoServer это просто идентификатор.

1) правило на *чтение*:
– workspace: common;
– layer and groups: *;
– access mode: Read;
– roles: COMMON_CREATOR.

2)правило на *редактирование* – все аналогично, кроме параметра **Access mode: Write**;

3) правило на *создание новых объектов* – все аналогично, кроме параметра **Access mode: Admin**.

Если при создании нового пользователя (меню **Security → Users, Groups, and Roles → Roles → Add new role**) назначить ему только роль **READER**, то после регистрации в системе ему будут доступны для чтения всего два раздела: **Data → Layer Preview** (все слои) и **Demos** (демонстрационные данные). Если пользователю назначить единственную роль **COMMON_CREATOR**, то ему будут доступны (для чтения, редактирования и создания) все подразделы **Data**, относящиеся *только* к слоям и данным соответствующей рабочей области. Таким образом, комбинирование ролей может обеспечить весьма гибкую структуру доступа к данным.

При просмотре списков слоев можно использовать фильтрацию путем ввода соответствующих строк и подстрок в поле **Search** текущего окна, например **Data → Layer Preview**. В качестве критерия поиска (фильтрации) можно использовать название рабочего пространства: так, после ввода topp будут видны только слои структуры рабочего пространства topp:*.

Для первичного просмотра опубликованных данных предназначен инструмент на основе библиотеки OpenLayers (меню **Data → Layer Preview** и далее – **Common Formats → OpenLayers**), обеспечивающий не только визуализацию слоя с параметрами по умолчанию, но и выбор стиля, масштаба, отображение атрибутивных данных пространственного объекта при его выделении курсором мыши, фильтрацию по атрибутам и т. п. В поставке GeoServer имеются и более комплексные примеры готовых фрагментов: в частности, ссылка http://<host>:8080/geoserver/www/wfs-t.html выдает начальное окно (рис. 3.11) веб-приложения, созданного на основе OpenLayers.

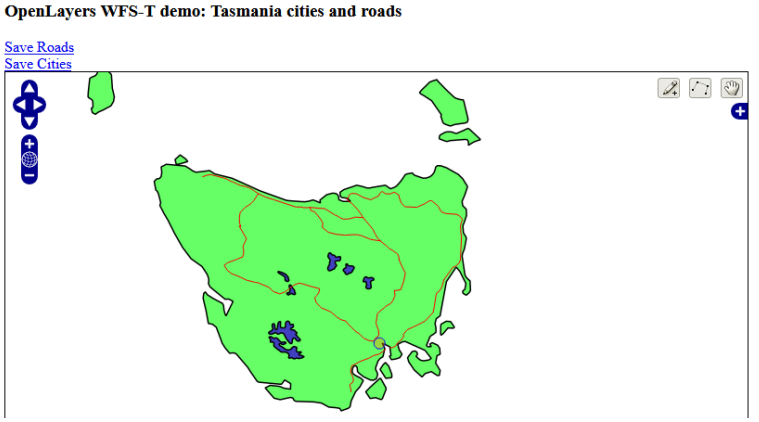

Рис. 3.11. Веб-приложение с элементами пользовательского интерфейса

Элемент меню **Common Formats** обеспечивает возможность выгрузки слоев пространственных данных в различных форматах, некоторые из которых были перечислены в разд. 3.2.2. Отдельный интерес представляют демонстрационные данные для групповых слоев (завершают список просмотра Layers Preview), показывающие возможность публикации фрагментов веб-карты в виде групп, соответствующих функциональности настольных ГИС типа ArcGIS Desktop, QGIS и т. п. Использование стилей дает возможность управлять не только видом отображения, но и диапазоном масштабов видимости для каждого отдельного слоя группы.

3.2.4. Публикация векторных данных

Наиболее распространенным способом публикации векторных данных является сервис (протокол) WMS[4], обеспечивающий представление данных в виде готового фрагмента карты – графического изображения в заданной проекции и стиле. Для указания проекции в программе GeoServer используется числовой номер SRS (Spatial Reference System), совпадающий с кодом регистра EPSG Dataset [19]. Так, географическая система координат WGS-84 имеет код EPSG:4326 (соответственно, номер SRS = 4326), а используемая в веб-картографии проекция Spherical Mercator (Pseudo Mercator) – код EPSG:3857 (SRS = 3857). Целесообразно сразу ограничить список возможных преобразований координат и оставить только те проекции, которые могут быть реально востребованы для конкретного экземпляра GeoServer. Это можно сделать через меню Services → WMS, введя в окно Limited SRS list список кодов: 4326, 3857, 4269, 32640, 28410, 900913. Значения 32640 и 28410 относятся соответственно к проекциям UTM zone 40N и Gauss-Kruger zone 10, а код 4269 – к проекции NAD83, используемой на территории США. Последний оставлен для возможности работы с демонстрационными данными, входящими в поставку GeoServer. Код EPSG-900913 соответствует проекции Google Maps Global Mercator, часто используемой в известных картографических сервисах и проектах[5].

Сразу после установки доступны все наиболее распространенные типы векторных данных, включая такие, как Directory of spatial files, GeoPackage (формат данных, хранимых в базах данных SQLite), PostGIS (PostGIS Database), PostGIS (JNDI), Properties (доступ к файлам с параметрами векторных данных), Shapefile (отдельные шейп-файлы), а также Web Feature Server (доступ к данным сервисов WFS из внешних источников). Наиболее практичными представляются хранилища векторных пространственных данных с типами Directory of spatial files (каталог шейп-файлов, объединенных одной проекцией и экстентом), GeoPackage и PostGIS (таблицы пространственных данных, хранимых в СУБД PostGIS/PostgreSQL).

4 Большая гибкость может быть обеспечена путем использования протокола WFS, позволяющего выполнять отображение векторных пространственных данных средствами клиентского приложения. Протокол WMS используется в первую очередь сервисами, предоставляющими «подложки» или базовые покрытия на основе открытых данных Open Street Map или мозаик аэрокосмических снимков.

5 В программу GeoServer включен тайловый сервис GeoWebCache, по умолчанию использующий сетки, основанные на проекциях EPSG:4326 EPSG:900913, наиболее распространенных в веб-картографии.

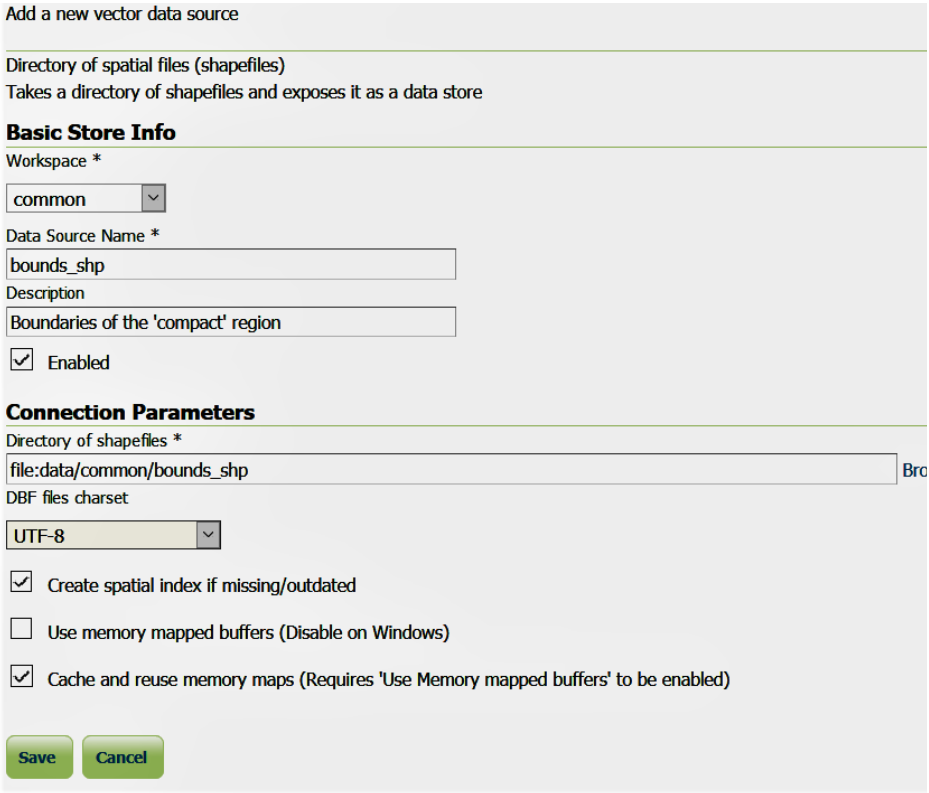

Рис. 3.12. Создание источника данных «Каталог шейп-файлов»

Каталог шейп-файлов является наиболее простым вариантом для быстрой публикации новых векторных данных, полученных из разных источников. Основные шаги процедуры публикации включают:

1. размещение шейп-файлов в пределах доступности программы GeoServer (например, в подкаталоге data структуры данных GeoServer);
2. создание источника пространственных данных с типом Directory of spatial files в соответствующем рабочем пространстве;
3. последовательная публикация шейп-файлов из каталога с назначением им соответствующих стилей отображения;
4. создание новых стилей и их назначение опубликованным слоям в качестве альтернативных (каждому пространственному объекту присваивается стиль «по умолчанию» и дополнительно может быть назначено любое количество пользовательских стилей, используемых при публикации по мере необходимости);

5. контроль результатов путем просмотра через инструмент Layer Preview.

Рассмотрим процесс на примере публикации границ некоторой области, используемой для учебных проектов. Исходное место размещения каталога шейп-файлов – data/common/bounds_shp/ в структуре данных GeoServer. Для создания источника данных (Data Store) следует выполнить действия:

1. через пункт меню **Data** → **Stores** → **Add new Store** выбрать тип Directory of spatial files (shapefiles), указать имя нового источника (поле **Data Source Name**) bounds_shp и рабочее пространство common. Целесообразно (но не обязательно) заполнить поле **Description** (рис. 3.12);
2. ввести ссылку на источник (каталог шейп-файлов) в поле **Directory of shapefiles** непосредственно или с помощью инструмента **Browse...**;
3. задать кодировку DBF-файлов в значение UTF-8 и сохранить параметры кнопкой **Save**. После этого появится окно со списком шейп-файлов, готовых к публикации (рис. 3.13);

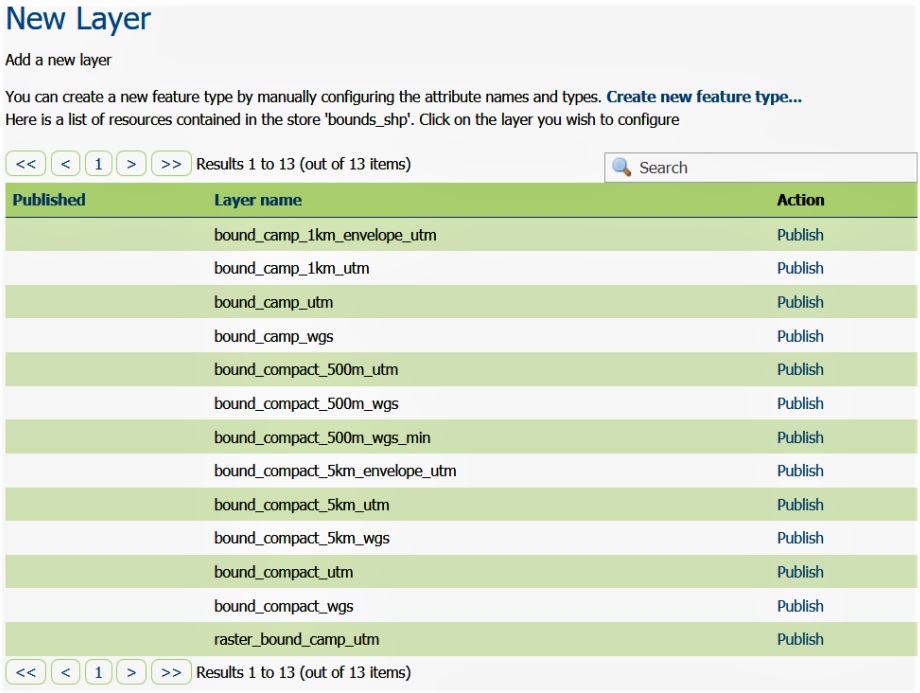

Рис. 3.13. Список шейп-файлов для публикации

4. основной (целевой) проекцией для публикации является проекционная система координат (далее – СК) WGS 1984/UTM zone 40N (EPSG:32640). В используемом примере имеются шейп-файлы

как в системе географических координат WGS 1984 (EPSG:4326), так и в целевой проекционной СК. Процедура публикации таких данных несколько отличается: в первом случае выполняется перепроецирование, во втором пространственные данные берутся «как есть»;

5. выбрать первый слой bound_compact_wgs в географических координатах (рис. 3.14), указать название в поле Title (опционально, по умолчанию значение поля совпадает с именем файла) и краткие пояснения (поле Abstract, заполняется также опционально);

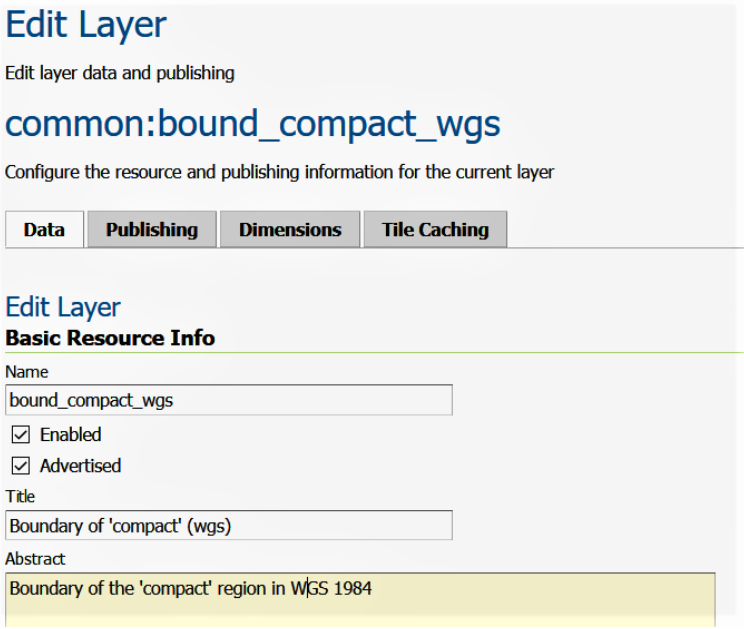

Рис. 3.14. Начало публикации шейп-файла

6. назначить параметры в секции **Coordinate Reference Systems** (рис. 3.15): исходную СК (определяется автоматически по исходному шейп-файлу) EPSG:4326 и целевую EPSG:32640 (выбирается из списка поиском по числовой части), а также требование проецирования (опция Reproject native to declared) из исходной СК в целевую. Экстент публикуемых данных заполняется автоматически после нажатия кнопок **Compute from data и Compute from native bounds**. После сохранения настроек слой публикуется со стилем по умолчанию (в данном случае – polygon). Дополнительные стили могут быть добавлены к слою позднее из числа доступных на вкладке **Publishing** через пункт меню **Data → Layers → Edit Layer**.

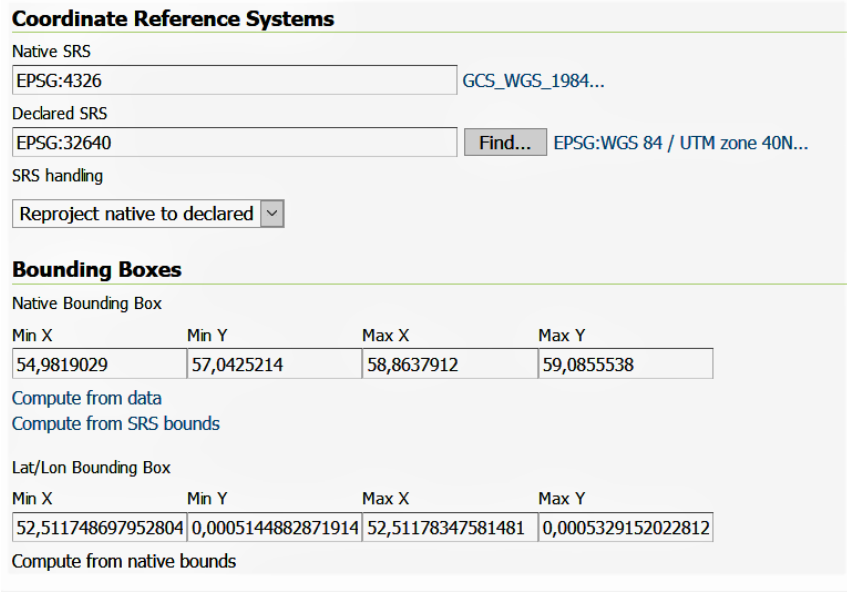

Рис. 3.15. Задание параметров перепроецирования

7. выбрать второй слой bound_compact_utm в проекционных координатах, соответствующих целевой СК, ввести название и краткие пояснения;

8. ввести код целевой СК (в данном случае он совпадает с кодом исходной EPSG:32640), так что перепроецирование не будет требоваться (опция **Force declared**). Экстент публикуемых данных также заполняется после нажатия кнопок и представлен в виде проекционных (в метрах) и географических (в градусах, минутах и секундах) координат.

После публикации обоих слоев (с точки зрения пользователя сервисами они абсолютно идентичны и выдают результат в одной и той же целевой СК) они становятся доступны для просмотра через Layer Preview и после последующих коррекций – через список опубликованных слоев (рис. 3.16) для добавления новых стилей и дополнительных параметров. Стили могут создаваться в формате SLD средствами встроенного текстового редактора GeoServer. В качестве основы удобно брать файлы стилей SLD, предварительно настроенные и экспортированные из QGIS[6]. Альтернативной формой для GeoServer является также использование каскадных стилей [22] (см. разд . Tutorial: Styling data with CSS).

6 Существует также формат YSLD, использующий язык разметки YAML. Этот формат более удобен для ручного редактирования, но для его использования требуется установка плагина YSLD Styling.

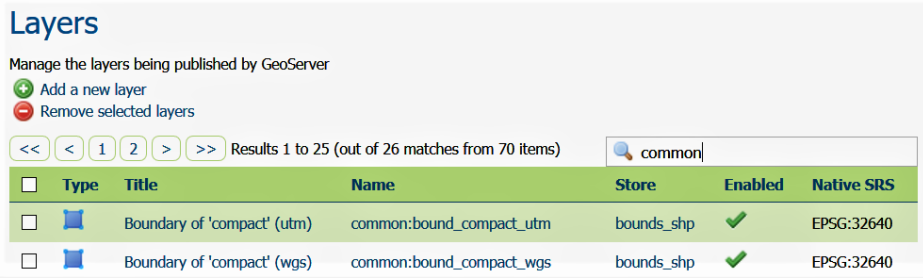

*Рис. 3.16. Опубликованные слои в списке **Layers***

Векторные данные могут публиковаться на основе источника Shapefile, позволяющего указать *только один* шейп-файл. Процедура добавления и публикации аналогична описанной выше.

Важным средством являются групповые (составные) слои, формируемые через меню **Data → Group Layers** путем объединения уже опубликованных слоев в группы. На этапе создания составного слоя назначаются порядок отображения слоев (сверху вниз, а не снизу вверх, как это принято в ГИС) и стили из числа добавленных ранее. С учетом возможностей стилей групповой слой обеспечивает полноценное картографическое отображение пространственных данных любой сложности, включая настройку изменения вида символов и их видимости в зависимости от текущего масштаба.

3.2.5. Публикация растровых данных

Организация работы с растровыми данными требует серьезного внимания из-за их значительного объема. В базовую поставку GeoServer включен ряд растровых источников, наиболее употребительными из которых являются типы GeoTIFF, WorldImage и ImageMosaic. Первый тип представляет собой отдельный файл, и для его публикации достаточно указать место его расположения. Тип WorldImage предусматривает файл в одном из графических форматов (jpeg, png, tiff и т. п.), сопровождаемый текстовым файлом (world file) с элементами пространственной привязки. Тип ImageMosaic обеспечивает «сшивку» нескольких растров в один с генерацией индексного файла с границами растров. На рис. 3.17 показан пример – мозаичное покрытие центральной части территории Пермского края фрагментами растров, полученных по спутниковым снимкам SPOT-5.

Перечисленных «базовых» типов вполне достаточно для небольших проектов, использующих ограниченные объемы данных, а следующая ниже часть подраздела ориентирована на более ресурсоемкие задачи.

Рис. 3.17. Увеличенный фрагмент мозаичного покрытия спутниковыми снимками SPOT-5 (составитель мозаики – С. И. Перминов)

Помимо указанных выше типов, в базовую установку включены такие типы хранилищ растровых данных, как ArcGrid и GeoPackage (mosaic). Целесообразно сразу увеличить этот список, добавив соответствующие расширения для текущей версии GeoServer из библиотеки GDAL (Geospatial Data Abstraction Library – свободно распространяемая библиотека работы с растровыми и векторными данными)[7]. Для этого нужно зайти на страницу загрузки, в разделе Extensions → Coverage Format последовательно выбрать позиции GDAL, Image Pyramid, JDBC Image Mosaic, JPEG2K и загрузить соответствующие архивные файлы:

- geoserver-2.16.0-gdal-plugin.zip(общее расширение, обеспечивающее работу с библиотекой GDAL для других расширений);
- geoserver-2.16.0-pyramid-plugin.zip (тип данных ImagePyramid для пирамид растров, хранимых в файловой системе);
- geoserver-2.16.0-imagemosaic-jdbc-plugin.zip (тип ImageMosaicJDBC для растров, хранимых в базах данных);
- geoserver-2.16.0-jp2k-plugin.zip (тип JPEG-2000 для растров, хранимых в виде файлов с расширением jp2).

Согласно указаниям документации, установка плагинов в систему сводится к распаковке архивов, выделению файлов с расширением .jar и переносу их в каталог .../webapps/geoserver/WEB-INF/lib/ по месту его расположения в файловой системе. После перезапуска GeoServer в список доступных типов растровых данных добавятся три новых: ImageMosaicJDBC, ImagePyramid и JP2K (Direct).

7 Помимо расширений для GeoServer, на уровне операционной системы должны быть установлены библиотеки GDAL с исполняемыми программами, в том числе на языке Python (для указанной выше операционной системы все они входят в пакеты gdal-bin, python-gdal и python3-gdal).

Быстродействие при передаче растровых данных (например, по протоколу WMS) от сервера к клиенту определяется не столько пропускной способностью каналов связи (ограниченных имеющейся инфраструктурой), сколько правильной организацией хранения и обработки данных на стороне сервера. Оптимальная структура должна обеспечивать быстрое извлечение нужного фрагмента данных (соответственно пространственной области) с нужным пространственным разрешением (соответственно масштабу). Как правило, размер набора тайлов, покрывающих запрошенную область в требуемом масштабе, намного меньше размера исходного растра. Для выполнения операций по преобразованию исходных растров в требуемые форматы и структуры требуются библиотека GDAL (Geospatial Data Abstraction Library [21]) и расширения для GeoServer, наиболее важные из которых указаны выше. Ниже представлены некоторые операции, которые могут быть выполнены с помощью программ из библиотеки GDAL[8].

Список программ на сайте библиотеки GDAL [21] включает в себя назначение и форматы использования утилит, обеспечивающих операции как с растровыми, так и с векторными данными. Эти утилиты наряду с PROJ и GEOS являются программной основой, на которой построены практически все свободные и часть коммерческих ГИС, а также связанные с ними сервисы. Ниже дан перечень наиболее распространенных утилит.

Получение списка поддерживаемых форматов данных:

```
gdalinfo --formats
```

Выводится список растровых форматов (в текущей версии – 136).

Получение данных о растровом файле:

```
gdalinfo raster_name
```

Например, для файла в формате GeoTIFF выводятся: информация о системе координат (название проекции, код EPSG, параметры эллипсоида, единицы измерения и т. п.), координаты верхнего левого угла, размеры ячейки растра, проекционные и географические координаты углов прямоугольника, порядок хранения данных в файле и сведения о каналах.

Преобразование файла из RGB (24 бита) в формат 8-битного цвета:

```
rgb2pct …
```

Операция сокращает объем «типичного» растрового файла примерно втрое, используя *палитру* – таблицу соответствия первичных цветов (яркости по каналам Red, Green, Blue) значениям от 0 до 255. Использованная для преобразования палитра отображается в выводе команды gdalinfo.

---

8 Средствами командной строки в любой операционной системе после установки библиотеки GDAL.

Преобразование форматов растровых файлов:

```
gdal_translate …
```

Утилита обеспечивает изменение структуры данных (например, переход от структуры striped к структуре tiled, рекомендуемой для больших файлов), размера ячеек, обрезание по заданному экстенту, позволяет менять формат данных, выполнять перепроецирование и т. п.

Создание мозаики растровых файлов:

```
gdal_merge …
```

Объединение смежных растров («сшивка») в один большой файл.

Создание пирамидальной структуры растровых файлов:

```
gdal_retile …
```

Утилита обеспечивает создание структуры или изменение имеющейся структуры пирамидных слоев (размер тайлов, количество уровней и т. п.). На вход подается каталог с растровыми файлами, на выходе генерируется структура в соответствии с заданным количеством уровней.

Типовые сценарии использования программ GDAL. Для создания (переконфигурирования) исходных структур растровых файлов и их подготовки к публикации средствами GeoServer можно рекомендовать два типовых сценария работы[9], которые могут применяться последовательно.

Сценарий 1. Формирование мозаичного покрытия (coverage) из имеющегося набора растровых файлов. Пусть исходные файлы покрывают большую территорию, чем нужно. В таком случае порядок действий может состоять из следующих шагов:

1. публикация исходной мозаики на GeoServer и получение индексного шейп-файла с границами отдельных гранул;
2. наложение контура с границами требуемой области на индексный шейп-файл и определение гранул, которые будут необходимы;
3. удаление лишних гранул с сохранением области прямоугольной формы и объединение оставшихся с помощью утилиты gdal_merge;
4. при необходимости – преобразование полученного файла с помощью утилиты gdal_translate: изменение структуры хранения данных, изменение глубины цвета (переход от 24 бит к палитре 8 бит, утилита rgb2pct) и т. п. Часть преобразований может быть выполнена на предыдущем этапе;

9 Конечно, это неполный, но опробованный на практике набор вариантов.

5. вырезание (средствами утилиты gdal_translate) фрагмента прямоугольной формы из получившейся прямоугольной области;
6. разделение фрагментов на файлы меньшего размера. Может выполняться в виде цикла операций утилиты gdal_translate, обеспечивающих последовательное вырезание фрагментов нужного размера и структуры;
7. публикация результата на GeoServer в виде растрового покрытия с использованием типа данных ImageMosaic с параметрами рис. 3.18. Пример визуализации результата средствами Layer Preview показан выше (рис. 3.17).

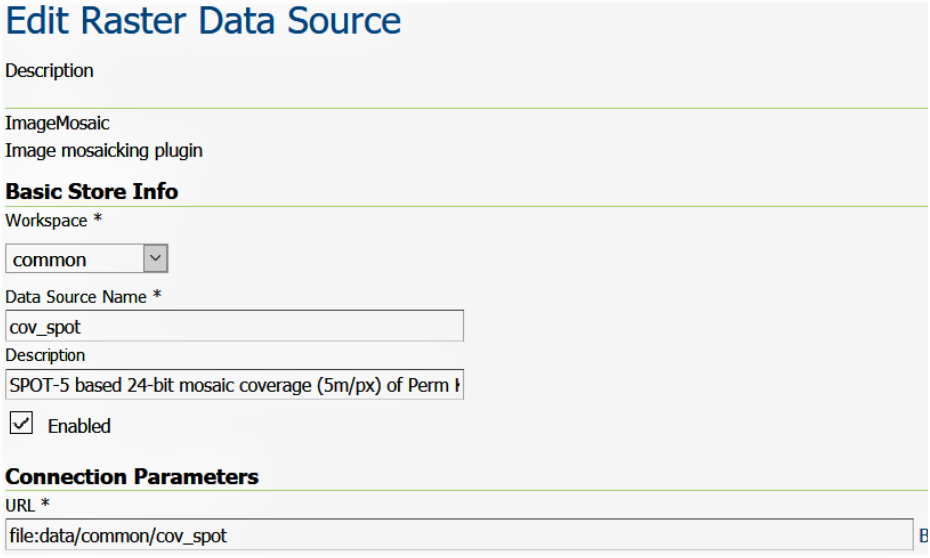

Рис. 3.18. Параметры источника данных типа ImageMosaic

Следует иметь в виду, что оптимальные параметры (размеры файлов и их структуру) следует подбирать опытным путем. Общие соображения говорят о том, что с точки зрения быстродействия дисковой системы хранения один большой файл лучше, чем много мелких. Практически же существуют ограничения используемой файловой системы и прочие факторы. Сжатие файлов также снижает быстродействие.

После публикации на GeoServer отображение слоя по запросу формируется достаточно долго, что обусловлено исходным размером файловой структуры. Динамическая генерация тайлов на стороне GeoServer занимает время. Очевидным способом повышения быстродействия является генерация статической структуры пирамидных слоев, рассмотренная ниже.

Сценарий 2. Формирование структуры пирамидных (pyramid) слоев на основе имеющегося мозаичного покрытия. Результат предыдущего сценария используется в качестве входных данных, а задача состоит в генерации пирамидных слоев до уровня, обеспечивающего максимально быстрое отображение всей территории. Порядок действий может включать следующие шаги:

1. генерация структуры пирамидных слоев средствами утилиты gdal_retile с указанием каталога с файлами растрового покрытия (результат сценария 1) и количества уровней. Последнее определяет степень двойки: число уровней 3 означает, что будет создано еще 3 слоя, уменьшающих масштаб (детализацию) исходного слоя в 2, 4 и 8 раз соответственно;
2. публикация результата на GeoServer в виде набора пирамидных слоев с использованием типа данных ImagePyramid. Параметры источника данных по форме совпадают с параметрами на рис. 3.18, визуализация результата средствами Layer Preview аналогична показанной выше на рис. 3.17.

Итоговым результатом применения пирамидной структуры растровых данных является существенно более высокая (практически мгновенная) скорость формирования изображения, что обусловлено ясной причиной: для отображения всей области используется минимальное количество файлов верхнего уровня, соответствующих самому грубому масштабу[10]. При использовании пирамидной структуры файлов с глубиной цвета 8 бит (цветовая палитра) формирование изображения происходит еще быстрее.

3.2.6. PostgreSQL как хранилище данных

Использование баз пространственных данных (spatial databases) в качестве хранилища данных обеспечивает неограниченную масштабируемость с точки зрения как объемов данных, так и количества одновременно работающих (в том числе в режиме редактирования) пользователей. При необходимости можно использовать продвинутые возможности систем управления базами данных (далее – СУБД) для создания высокопроизводительных систем любого уровня. Применение СУБД для небольших проектов оправдано в связи с тем, что структура баз данных обеспечивает целостность данных и многопользовательский режим работы. В связке с GeoServer целесообразно использовать PostgreSQL [32] с расширениями PostGIS [31], добавляющими функциональность работы с пространственными данными[11]. В литературе такая пространственная СУБД часто обозначается как связка PostgreSQL/PostGIS, или кратко – PostGIS.

---

10 Не следует путать тайлы (типичные размеры – 256 x 256 или 512 x 512 пикселей) и значительно более крупные файлы, создаваемые при генерации пирамидной структуры. Действующая структура данных обеспечивает баланс между скоростью операций чтения (сравнительно небольшое количество файлов) и скоростью передачи по сети (небольшой размер реально передаваемых тайлов).

11 Для этих же целей можно использовать коммерческие СУБД MS SQL, Oracle и многие другие с соответствующими пространственными расширениями.

Описание процедур установки и настройки пространственной СУБД выходит за рамки данного пособия. Тем не менее следует перечислить ряд шагов, сопровождающих использование PostgreSQL/PostGIS в качестве источника пространственных данных для GeoServer:

1. Установка PostgreSQL средствами операционной системы [32] в версии, совместимой с версией GeoServer. Загрузка и установка пространственных расширений согласно указаниям [31].

2. Настройки безопасности PostgreSQL. Как минимум это смена пароля суперпользователя postgres и добавление пользователей с правами создания пространственных баз данных и отдельно – с правами только на чтение[12]. Могут также потребоваться разрешения на доступ с внешних машин, т. к. по умолчанию возможно только локальное подключение к PostgreSQL.

3. Опционально – установка программного обеспечения для визуального управления PostgreSQL, например свободно доступной программы pgAdmin IV с интерфейсом в веб-браузере. Видимый на рис. 3.19 фрагмент интерфейса pgAdmin IV показывает возможность подключения к серверам не только в связке PostgreSQL/PostGIS (полностью свободное решение), но и к СУБД на основе PostgreSQL с пространственными расширениями от компании ESRI (решение, используемое в качестве источника пространственных данных для ArcGIS Server).

4. Подготовка и импорт пространственных данных в СУБД. Может осуществляться различными способами, включая графические инструменты и программы, запускаемые из командной строки.

Перечислим некоторые из способов импорта шейп-файлов в пространственную СУБД PostgreSQL/PostGIS:

- графический инструмент pgShapeloader из комплекта программ OpenGeo Suite [28], устанавливаемого под Windows, Linux и macOS. Может запускаться прямо из интерфейса pgAdmin IV, также входящего в комплект;

- графический инструмент shp2pgsql-gui, устанавливаемый в составе пакета PostGIS [31] под Windows, Linux и macOS;

- инструмент командной строки shp2pgsql, также устанавливаемый в составе пакета PostGIS [31] под Windows, Linux и macOS. Может использоваться вне графического интерфейса операционной системы, что удобно для серверов под управлением Linux. Фактически это программа, используемая в упомянутых выше графических инструментах;

- расширение DB Manager, входящее в базовую поставку QGIS [33], при импорте шейп-файлов также использующее в качестве базы программу shp2pgsql.

---

12 Последнее выполняется после создания баз данных. Пользователи с ограниченными правами нужны для того, чтобы предоставлять доступ только в режиме просмотра и только к «разрешенным» данным.

Как видим, основой импорта везде является программа shp2pgsql, обновляемая согласованно с версиями PostgreSQL/PostGIS. В составе PostGIS имеются также утилиты экспорта pgsql2shp данных в формат шейп-файлов, а также утилита raster2pgsql для импорта (загрузки) растров.

Программа shp2pgsql генерирует SQL-скрипт, который затем исполняется на стороне СУБД (посредством pgAdmin IV, вкладки SQL или иным способом) при соответствующих параметрах подключения (рис. 3.19). В качестве примеров приведем формат команд, который может быть использован для подготовки к импорту шейп-файлов из разд. 4.2.4:

```
shp2pgsql -s 4326 -I bound_compact_utm wgs.bound > bound_wgs.sql
shp2pgsql -s 32640 -I bound_compact_utm utm.bound > bound_utm.sql
```

Параметр «-s» означает номер SRS, задаваемый явно, а параметр «-I» генерирует пространственный индекс. Первый аргумент указывает на имя шейп-файла (без расширения), второй – на полное имя пространственной таблицы в базе геоданных (далее – БГД). Префикс имени (wgs., utm.) соответствует имени схемы – элементу логической структуры БГД, связанному в данном примере с разными системами координат. Результат выполнения команды (SQL-скрипт) записывается в текстовом виде и может быть использован для загрузки данных в БГД средствами СУБД, например через клиентское приложение командной строки psql, обеспечивающее всю необходимую функциональность.

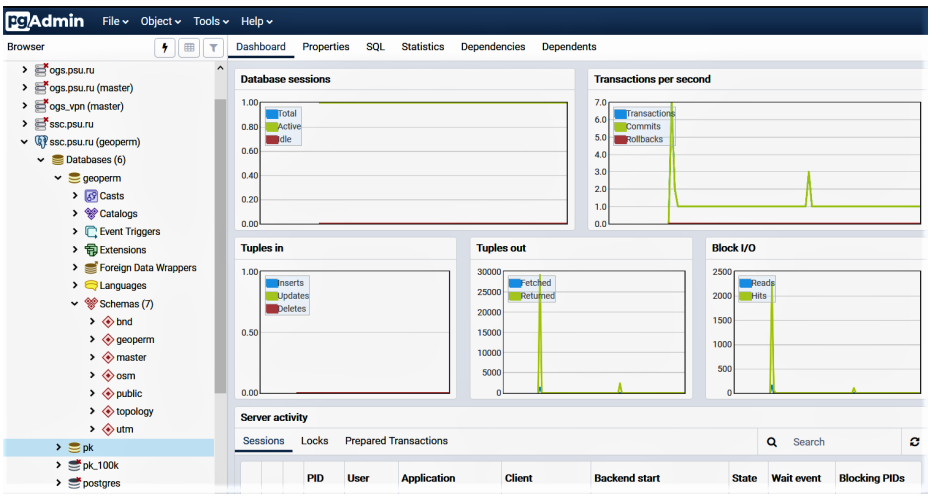

Рис. 3.19. Фрагмент интерфейса программы pgAdmin IV

На этапе публикации данных из PostgreSQL/PostGIS в интерфейсе программы GeoServer предусмотрена возможность создания *представлений* на основе SQL-запросов. Фактически это те же пространственные таблицы, которые могут быть сформированы динамически на основе одной или нескольких связанных исходных таблиц. Ниже дан пример простейшего пред-

ставления, реализующего функцию статического фильтра по значению criteria в поле field_name:

```
SELECT *
FROM table_name
WHERE field_name='criteria'
```

Представления могут включать параметры, значения которых нужно задавать в момент запроса. В свою очередь, на стороне СУБД есть свои возможности по управлению данными для публикации, включая как создание комплексных выборок с элементами агрегирования и сложных вычислений, так и ограничение доступа на уровне отдельных схем, таблиц, полей и записей (строк).

3.2.7. Дополнительные возможности публикации данных

Программа GeoServer предоставляет достаточно много дополнительных функций, не описанных в данном пособии. Помимо групповых слоев (разд. 4.2.4), просмотров, фильтров и слоев с параметрами (см. завершающую часть разд. 4.2.6), GeoServer может использоваться в качестве посредника для публикации данных из внешних источников. Внешние источники (протоколы WMS, WFS и т. п.), в свою очередь, могут использоваться для объединения с собственными данными.

3.3. Использование облачных картографических платформ для публикации пространственных данных и создания картографических веб-приложений

Еще одним популярным способом публикации геопространственных данных в сети Интернет является использование облачных картографических платформ. Наиболее часто используемые из них – ArcGIS Online (URL-адрес для доступа: https://arcgis.com) [13] и NextGIS.com (URL-адрес для доступа: https://my.nextgis.com) [14]. Облачные картографические платформы построены по модели SaaS (программное обеспечение как сервис).

3.3.1. Создание картографических веб-приложений с использованием облачной платформы ArcGIS Online

ArcGIS Online, как и все продукты ArcGIS, является разработкой компании ESRI и часто используется как одна из составляющих портала ArcGIS Enterprise (Portal for ArcGIS) при наличии у пользователя (обычно компании) соответствующей лицензии на программные продукты линейки ArcGIS [13]. В этом случае ArcGIS Online используется как система общего доступа ко всем опубликованным в организации ресурсам, связанным с пространственными данными. Однако часть функциональных возможностей

платформы ArcGIS Online доступна для любого пользователя на бесплатной основе в некоммерческих целях.

Для работы с данной платформой необходима учетная запись пользователя. Учетные записи для доступа в ArcGIS Online бывают двух типов: персональные и корпоративные. Они существенно отличаются друг от друга функциональными возможностями, доступными пользователям (табл. 3.3). К корпоративной учетной записи можно получить доступ при покупке лицензии, а к персональной – при простой регистрации на сайте.

Таблица 3.3.
Сравнительная характеристика учетных записей ArcGIS Online

Параметры сравнения	**Учетные записи**	
	Корпоративная	**Персональная**
Доступ, совместное использование и управление пространственными ресурсами:	+	+
• поиск карт, данных и приложений	+	+
• доступ к базовым картам, сервисам изображений	+	+
• доступ к веб-картам с помощью браузера, мобильных устройств или настольного приложения ArcGIS Desktop	+	+
• встраивание карт в веб-страницы, блоги, приложения	+	+
• создание групп для совместной работы с ресурсами	+	+
• гибкая настройка доступа к ресурсам (личный, внутри групп или общедоступный)	+	+
Добавление собственных данных на веб-карту	+	+
Добавление сервиса геометрии в карты и приложения	+	+
Хранение карт, данных и приложений в облаке компании ESRI	+	Лимит 2 GB
Создание карт по табличным данным в Microsoft Excel	+	
Добавление сервиса геокодирования в карты и приложения	+	
Доступ к Portal for ArcGIS и Web Mapping API	+	
Публикация картографических веб-сервисов в облаке компании ESRI	+	
Управление ролями пользователей, доступом и настройками безопасности	+	

Параметры сравнения	Учетные записи	
	Корпоративная	Персональная
Отслеживание параметров использования облака	+	
Настройка домашней страницы организации на ArcGIS Online	+	
Добавление ресурсов организации на домашнюю страницу ArcGIS Online	+	
Создание пользовательского URL-адреса для домашней страницы организации	+	
Техническая поддержка	+	

После входа в ArcGIS Online для просмотра собственных карт, приложений и данных необходимо перейти на вкладку **Ресурсы** (рис. 3.20), где представлены все доступные ресурсы пользователя, включающие карты, трехмерные сцены, приложения, слои данных и т. д. При переходе в меню каждого ресурса становятся активными функции их редактирования, просмотра и управления доступа пользователей к ним.

Также на вкладке **Ресурсы** имеются возможности создания (меню **Создать**) карт, трехмерных сцен, а также веб-приложений с использованием карт и 3D-сцен, созданных в ArcGIS Online как самим пользователем, так и другими пользователями, предоставившими общий доступ к своим ресурсам.

Кроме того, ArcGIS Online можно использовать как облачное хранилище для пространственных данных (меню **Добавить** вкладки **Ресурсы**), например для хранения векторных данных (в формате ESRI Shapefile), растров и других файлов. При этом некоторые данные, например векторные, можно непосредственно отобразить на карте в ArcGIS Online и создать на его основе собственный ресурс.

При работе над общими ресурсами существует возможность создания групп пользователей (вкладка **Группы**). Благодаря использованию групп можно настроить доступ к некоторым ресурсам только для определенных пользователей.

Важной функцией, которая доступна пользователям ArcGIS Online, является возможность создания веб-приложений без непосредственного написания кода. Для этого на первом этапе необходимо создать карту(ы) или сцену(ы) (меню **Создать** → **Карта** (или **Сцена**) вкладки **Ресурсы** либо выбрать вкладку **Карта** или **Сцена**). При создании карты сначала необходимо ввести ее название и описание. Затем в открывшемся вьюверe (который тоже является веб-приложением) необходимо наполнить карту содержимым через меню **Добавить**. В качестве содержимого можно использовать собственные слои с компьютера в форматах ESRI Shapefile в виде ZIP-архива, GPX, GeoJSON, CSV и TXT (меню **Добавить слой из файла**), слои и карты, опубликованные на ArcGIS Server (меню **Добавить слой из Интернета**), созданные другими пользователями ArcGIS Online слои, к которым имеется общий доступ (меню **Поиск слоев**).

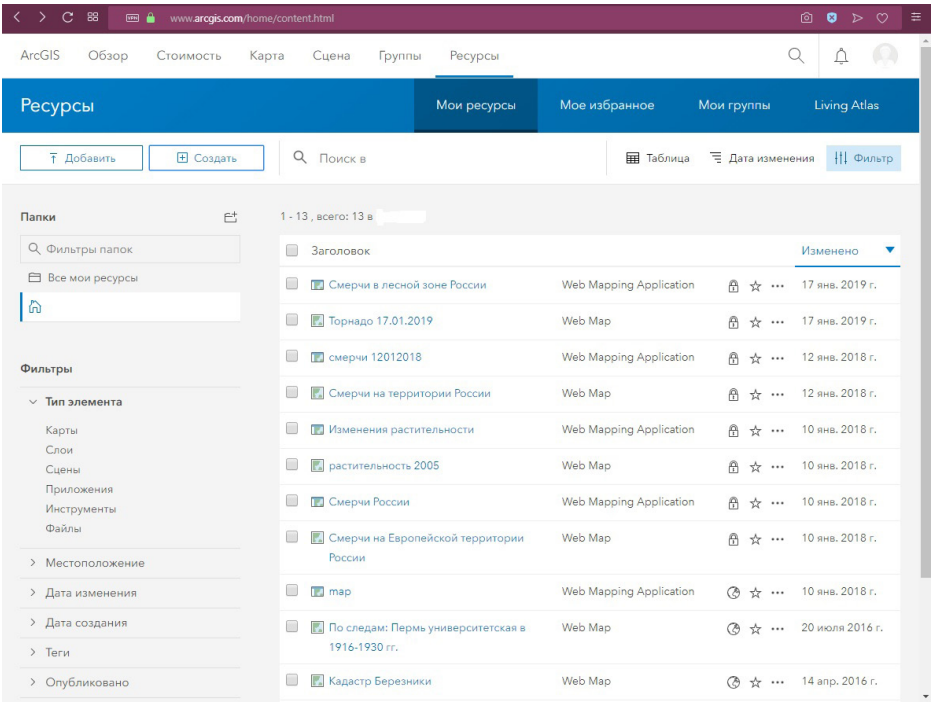

Рис. 3.20. Внешний вид меню Ресурсы ArcGIS Online

Настройка слоев осуществляется на вкладке **Содержание** и включает в себя выбор способа отображения данных, выбор символов, настойку диапазонов масштабов видимости, настройку всплывающих окон при идентификации объектов и т. д. После настройки карту необходимо сохранить. Аналогичным образом можно создать 3D-сцену. После сохранения карта становится доступной во вкладке **Ресурсы** ArcGIS Online. Ее можно просмотреть через стандартный вьювер ArcGIS Online.

Далее на базе созданных карт и сцен создается клиентское веб-приложение (меню **Создать** → **Приложения** на вкладке **Ресурсы**). При использовании персональной учетной записи ArcGIS Online создание веб-приложения доступно только на основе имеющихся шаблонов, которые разбиты по разным тематикам, например: для отображения данных, для построения карт историй, для организации сбора и редактирования данных, для сравнения нескольких карт, для отображения 3D-сцен и т. д. После выбора шаблона указывается одна или несколько карт или сцен (в зависимости от типа шаблона), которые будут доступны для просмотра и работы в веб-приложении. Далее необходимо настроить приложение в специальном окне (рис. 3.21).

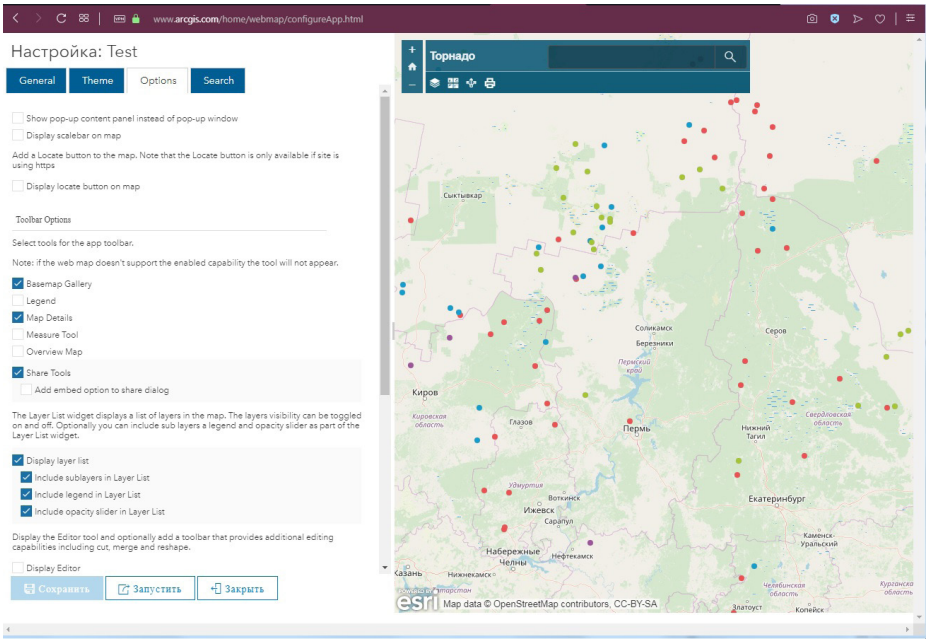

Рис. 3.21. Окно настройки веб-приложения, созданного с использованием шаблона ArcGIS Online

В настройках есть возможность выбора цветового оформления панелей и кнопок приложения, а также необходимых опций для работы с картой, таких как отображение легенды карты и списка слоев, добавления инструментов измерений расстояний и площадей, масштабной линейки, обзорной карты, инструмента определения местоположения пользователя и галереи базовых карт.

После настройки и сохранения веб-приложение станет доступным во вкладке **Ресурсы** ArcGIS Online. Ему будет присвоен уникальный идентификатор (id), состоящий из латинских букв и цифр, и уникальная ссылка (URL-адрес) для его просмотра, включающая идентификатор. Стоит отметить, что картам и сценам, созданным в ArcGIS Online, также присваиваются аналогичные идентификаторы. Они позволяют искать карты и сцены и использовать их при создании веб-приложений на базе библиотеки ArcGIS API for JavaScript.

Чтобы созданное в ArcGIS Online веб-приложение было доступно другим пользователям, необходимо настроить общий доступ к нему с помощью меню. Там же можно и просмотреть приложение в окне браузера, при этом в адресной строке веб-браузера будет отображаться присвоенная приложению ссылка. Пример веб-приложения, созданного с использованием одного из шаблонов ArcGIS Online, приведен на рис. 3.22. Данное приложение позволяет получить информацию о смерчах, произошедших на территории России.

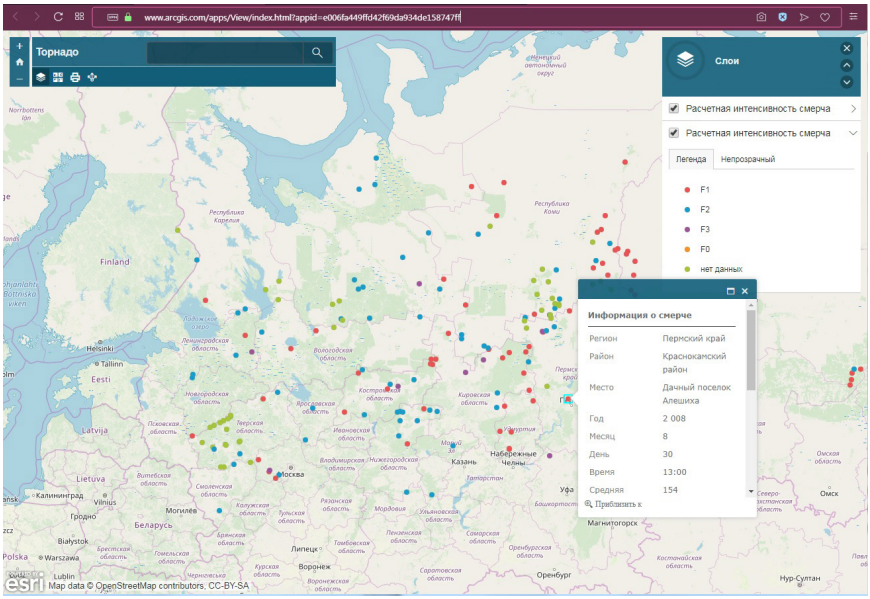

Рис. 3.22. Веб-приложение, созданное с использованием шаблона ArcGIS Online

При использовании корпоративной учетной записи для создания веб-приложений можно также обратиться к конструктору, который называется Web AppBuilder for ArcGIS (рис. 3.23). Он также позволяет создавать веб-приложения без написания кода. При этом доступна функция экспорта созданного приложения в виде набора файлов с исходным кодом для дальнейшей доработки.

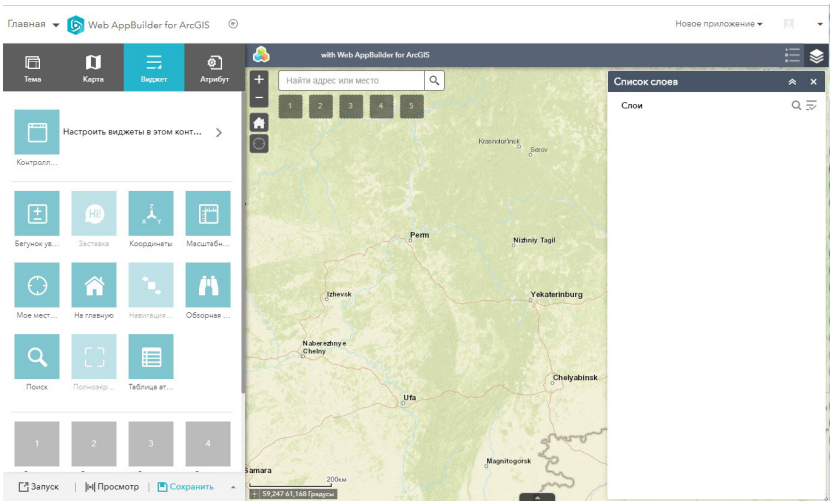

Рис. 3.23. Окно конструктора картографических веб-приложений Web AppBuilder for ArcGIS

При использовании Web AppBuilder for ArcGIS существует возможность выбора тем оформления приложений, настройки цветового решения, размещения различных виджетов в окне приложения. Виджеты представляют собой визуальный элемент интерфейса приложения (кнопка, окно, блок), который позволяет получить оперативный доступ к какому-либо действию. Например, существует виджет в виде кнопки для перехода к домашнему экстенту карты, виджет для просмотра таблицы атрибутов векторного слоя и т. п. После настройки и сохранения веб-приложение станет доступным во вкладке **Ресурсы** ArcGIS Online.

3.3.2. Создание картографических веб-приложений с использованием облачной платформы NextGIS.com

Работа с облачной платформой NextGIS.com несколько похожа на работу с ArcGIS Online.

К основным функциям и возможностям облачной платформы NextGIS.com относятся [14]:

- загрузка и хранение растровых (GeoTiff) и векторных (ESRI Shape в виде ZIP-архива, GeoJSON) пространственных данных;
- подключение пространственных данных из внешних баз (PostGIS) и картографических веб-сервисов;
- отображение пространственных данных на создаваемых пользователем веб-картах, а также их публикация с помощью стандартных протоколов;
- настройка стилей визуализации пространственных данных;
- редактирование атрибутов загруженных пространственных данных, добавление их описания и фотографий;
- работа с пространственными данными через мобильное приложение NextGIS Mobile и настольное приложение NextGIS QGIS;
- создание собственных клиентских приложений с использованием NextGIS API.

Для доступа к облачной платформе необходимо создать учетную запись (аккаунт).

У NextGIS.com существует три варианта использования: один бесплатный (Free) и два платных (Mini и Premium). При использовании бесплатной версии отсутствует возможность закрывать данные от других пользователей NextGIS.com, существуют ограничения по количеству слоев и карт (до 30), пользователей, а также отсутствует возможность использования данных на других сайтах. В варианте Mini можно создавать неограниченное количество слоев и карт и использовать карты на сторонних сайтах. План Premium имеет функции настройки расширенного управления доступом, при помощи которых можно защитить данные от сторонних пользователей. В дополнение к этому тариф Premium позволяет создавать неограниченное

число слоев и карт, использовать собственный домен и элементы оформления, а также дает большую производительность и функциональность.

После создания учетной записи NextGIS.com необходимо создать веб-ГИС, внутри которой будет происходить дальнейшая работа с пространственными данными (их хранение, создание на их основе карт и т. д.), и выбрать один из трех тарифных планов. После создания веб-ГИС доступ к ресурсам пользователя осуществляется при авторизации по адресу вида username.nextgis.com, где username – это имя учетной записи пользователя.

При входе в учетную запись пользователь попадает в административный интерфейс (рис. 3.24), внутри которого можно создавать различные ресурсы. Среди них – векторные и растровые слои, полученные путем загрузки данных, WMS-сервисы и WFS-сервисы, слои из базы данных PostGIS, а также веб-карты, которые создаются из загруженных слоев.

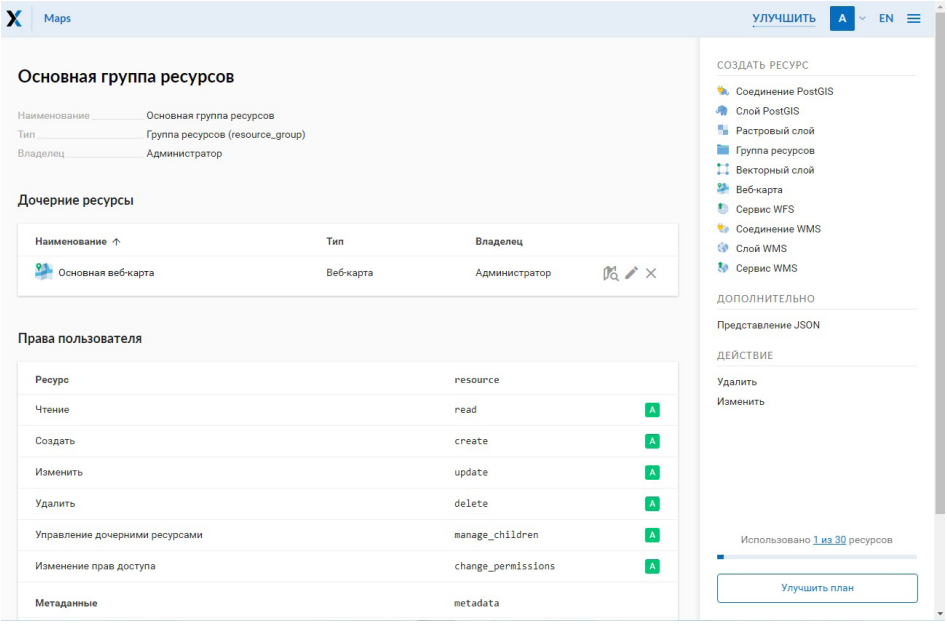

Рис. 3.24. Административный интерфейс веб-ГИС-платформы NextGIS.com

Для создания веб-карты необходимо сначала загрузить векторные и растровые слои для нее в виде отдельных ресурсов. Создание любых ресурсов и наполнение их данными происходит с помощью панели справа **Создать ресурс**. Для загрузки векторного слоя необходимо перейти в меню **Векторный слой** (рис. 3.25). Загрузка векторных данных доступна в форматах ESRI Shapefile (ZIP-архив) и GeoJSON.

Поскольку в NextGIS.com хранение данных отделено от их представления (как и в ArcGIS Online), то после загрузки векторного слоя ему необходимо задать стиль отображения. При этом следует учитывать ряд моментов.

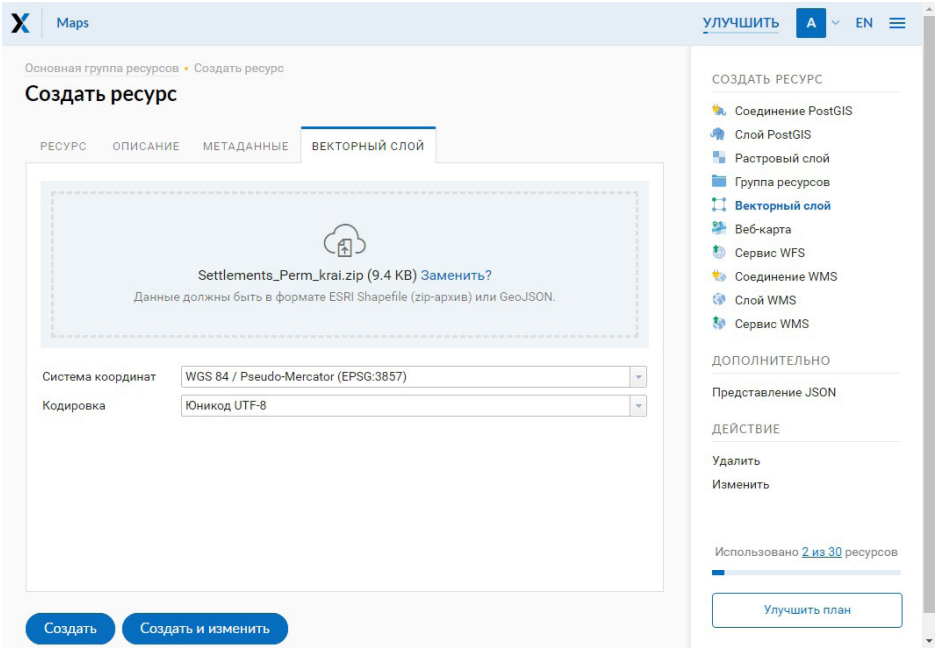

Рис. 3.25. Меню загрузки векторных слоев в платформе NextGIS.com

Во-первых, созданный стиль тоже является ресурсом и для него доступны операции удаления, изменения, переименования и т. д. Во-вторых, у одного загруженного слоя может быть несколько разных стилей. И, в-третьих, при создании веб-карты используются не слои (данные), а их представления в виде стилей. Создание стиля доступно в меню слоя. При этом NextGIS.com поддерживает два типа стилей для векторных данных: стили MapServer (которые используются для оформления карт на ГИС-сервере MapServer) и стили QGIS. Первые пишутся вручную на специальном языке тегов, используемом MapServer при описании карт в Mapfile (см. разд. 2.4.1). Главным недостатком этих стилей является отсутствие визуального редактора для них. Стили QGIS создаются в настольной ГИС QGIS при их сохранении в формате QML. Стоит отметить, что такие стили удобно настраивать и редактировать в интерфейсе QGIS. Поэтому предпочтительнее использовать стили QGIS.

Растровые данные загружаются аналогично векторным с помощью меню **Растровый слой**. При этом для загрузки доступны только трехканальные (RGB) растры в формате GeoTIFF. Для отображения растрового слоя также нужно создать стиль. Но в отличие от векторного растровый стиль создается из палитры самого растра и при этом не имеет опций для его настройки и изменений.

Для удобства работы все ресурсы, относящиеся к одной веб-карте или сервису, можно объединить в группу (поместить в один каталог). Для создания группы используется меню **Группа ресурсов**.

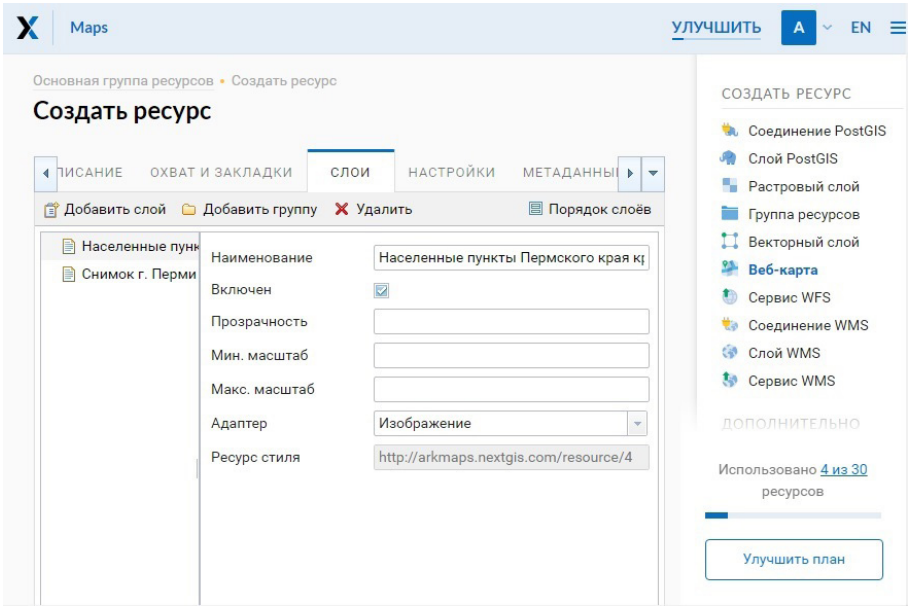

Рис. 3.26. Меню настройки слоев веб-карты в NextGIS.com

После загрузки слоев из можно создать веб-карту при помощи меню **Создать ресурс → Веб-карта**. При входе в данное меню необходимо указать название веб-карты и на вкладке **Слои** добавить на нее созданные ранее представления слоев в виде стилей (рис. 3.26). При использовании платных тарифов доступны также настройки ограничения доступа к карте, выбора базовой карты (подложки). По умолчанию в качестве подложки используются данные OpenStreetMap в виде картографического сервиса. После создания веб-карта добавляется в список ресурсов, где имеется функция ее просмотра в веб-интерфейсе приложения NextGIS.com (рис. 3.27).

При необходимости использования загруженных пространственных данных в сторонних веб-приложениях и настольных ГИС создаются веб-сервисы, соответствующие стандартам OGC. NextGIS.com позволяет предоставлять данные в виде двух веб-сервисов: WMS и WFS. Процесс создания веб-сервисов аналогичен процессу создания веб-карты.

Главный недостаток NextGIS.com состоит в том, что эта платформа не имеет простых для рядового пользователя возможностей создания отдельных веб-приложений, таких как, например, набор шаблонов или Web AppBuilder в ArcGIS Online.

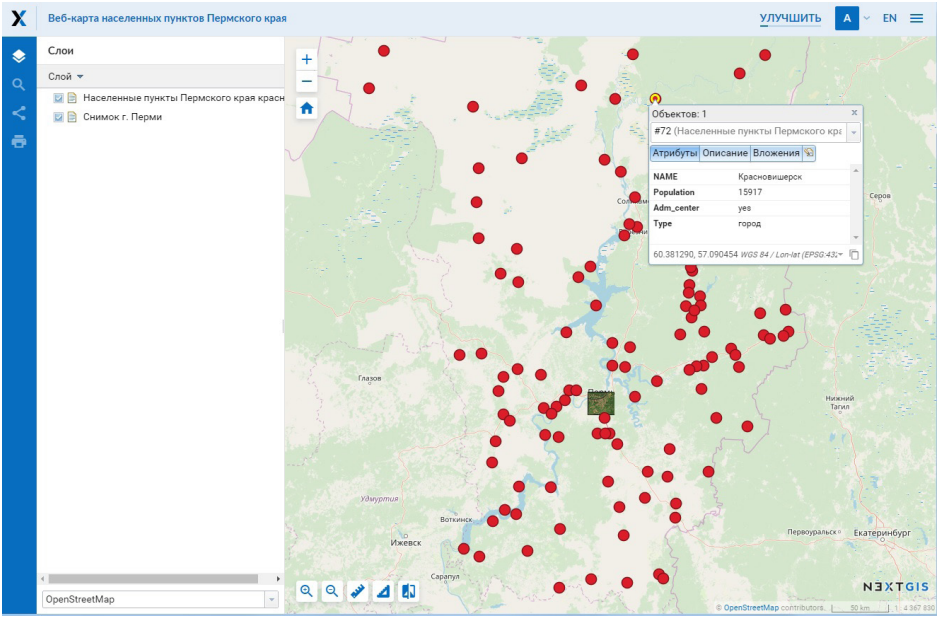

Рис. 3.27. Пример веб-карты, созданной в NextGIS.com с использованием растрового и векторного слоев

4. ТЕХНОЛОГИИ РАЗРАБОТКИ КЛИЕНТСКОЙ ЧАСТИ КАРТОГРАФИЧЕСКИХ ВЕБ-ПРИЛОЖЕНИЙ

При разработке картографических веб-приложений на стороне клиента необходимо знание минимального набора технологий. К ним относятся стандартные технологии формирования графического пользовательского интерфейса (GUI – Graphical User Interface) – язык разметки HTML и таблицы стилей CSS, а также технологии придания интерактивности веб-приложениям, к которым в первую очередь относится язык программирования JavaScript. Для работы в веб-приложениях с картографической информацией используются специальные библиотеки, написанные на языке JavaScript. Наиболее известные и используемые из них – OpenLayers, Leaflet, Mapbox GL, ArcGIS API for JavaScript и др. Далее в главе приводится описание минимального набора (стека) технологий, необходимого для разработки картографических веб-приложений на стороне клиента.

4.1. Язык разметки HTML и таблицы стилей CSS

HTML (HyperText Markup Language – язык разметки гипертекста) – это стандартизированный язык разметки документов (веб-страниц) во Всемирной паутине. Он был разработан британским ученым Тимом Бернерсом-Ли в 1989–1991 гг. в стенах Европейского Центра ядерных исследований в Женеве (Швейцария).

Документы HTML являются текстовыми документами, имеющими расширение .html или .htm. Любой HTML-документ состоит из набора отдельных элементов, начало и конец которых обозначается при помощи специальных конструкций, называемых тегами (или дескрипторами). Большинство элементов начинается открывающим тегом, обозначающимся как <имя тега>, а заканчивается закрывающим – </имя тега>. Также есть элементы, имеющие только открывающие теги, например: тег перевода строки –
 или тег для прорисовки горизонтальной черты – <hr>. Регистр, в котором набрано имя элемента, в HTML значения не имеет. Элемент HTML-документа может быть как пустым, так и может содержать текст, изображения, ссылки, другие элементы (теги) и т. п.

Элементы HTML могут иметь атрибуты, которые расширяют их возможности, позволяют изменять свойства и поведение элементов. Они бывают глобальными, применяемыми ко всем элементам HTML-документа, и собственными. Атрибуты прописываются в открывающем теге элемента, каждый атрибут имеет собственное зарезервированное имя и набор допустимых значений. Конструкция элемента с атрибутами записывается следующим образом:

<имя тега атрибут_1=”значение” атрибут_2=”значение” ...></имя тега>

(например,

текст).

В данном случае у элемента для создания ссылок <a> заданы атрибуты href (адрес, по которому пользователь при нажатии на ссылку) и title (текст, обозначенный как ссылка, на веб-странице).

Аналогично атрибутам у HTML-элементов могут быть и события. События – это специальные глобальные атрибуты, используемые в тегах для вызова обработчиков событий, написанных на различных языках сценариев (чаще всего JavaScript) и вызываемых тогда, когда на странице происходит какое-либо действие. События позволяют сделать веб-страницу динамической. Примером события может быть клик по элементу, например по кнопке (событие onclick). Конструкция элемента с назначенным событием записывается следующим образом:

<имя тега событие_1=”вызываемый скрипт_1” событие_2=”вызываемый скрипт_2” ...></имя тега>.

Перечень всех HTML-элементов (тегов) и поддерживаемые ими атрибуты и события можно посмотреть, например, на сайте http://htmlbook.ru [15]. Основные часто используемые элементы представлены далее в настоящем разделе.

Интерпретаторами языка HTML выступают веб-браузеры, которые преобразуют текст и отображают документ в форматированном виде. При этом в разных браузерах веб-страницы могут отображаться по-разному. Поэтому при разработке приложений необходимо учитывать эту особенность.

Стоит отметить, что на сегодняшний день существует несколько версий языка HTML. Наиболее используемыми из них являются HTML 4.01 (опубликована в 1999 г.) с расширяющим ее возможности XHTML 1.0 – Extensible HyperText Markup Language и HTML5 (опубликована в 2014 г.). Стандарты версий HTML закрепляются Консорциумом Всемирной паутины (World Wide Web Consortium – W3C). Отличия версии HTML5 в первую очередь состоят в улучшении поддержки мультимедиатехнологий в браузере без применения дополнительных модулей и плагинов (таких как Flash Player). Для реализации этого в 5-й версии были добавлены новые элементы (теги), такие как <audio>, <video>, <canvas> и др. Еще одна особенность HTML5 по сравнению с 4-й версией – появление новых элементов для обогащения семантического (смыслового) содержания документов. Это такие элементы, как <section>, <article>, <header>, <footer> и др. Также язык HTML5 объединил в себе синтаксические нормы HTML и XHTML, которые ранее существовали отдельно друг от друга.

В стандартной структуре HTML-документа выделяются некоторые общие элементы. Так, каждый HTML-документ, отвечающий спецификации HTML, обязан начинаться со строки декларации используемой версии. Для этого используется элемент <!DOCTYPE>. Этот элемент необходим для того, чтобы браузер понимал, как следует правильно интерпретировать текущую веб-страницу, поскольку HTML существует в нескольких версиях. Кроме того, имеется XHTML, похожий на HTML, но отличающийся от него по синтаксису.

Для версии HTML 4.01 данный элемент выглядит приблизительно следующим образом:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN" "http://www.w3.org/TR/html4/strict.dtd">.
```

Для стандарта HTML версии 5 строка выглядит значительно проще: <!DOCTYPE html>. Стоит отметить, что элемент <!DOCTYPE> не имеет закрывающего тега.

После тега объявления версии языка HTML обозначается начало и конец документа с помощью тегов <html> и </html>. Остальные элементы, находящиеся внутри данных тегов, образуют дерево документа, или объектную модель документа, – DOM (Document object model) [24]. При этом элемент <html> является корневым элементом дерева (рис. 4.1). Таким образом, HTML-документ имеет иерархическую структуру.

В связи с иерархической структурой HTML-документа DOM-элементы находятся между собой в определенных отношениях, согласно которым выделяют родительские, дочерние и сестринские элементы, а также элементы-потомки и предки.

Предок – элемент, который заключает в себе другие элементы. На рис. 4.1 предком для всех элементов является <html>. В то же время элемент <body> является предком для всех содержащихся в нем тегов: <h1>, , <h2>, <img>.

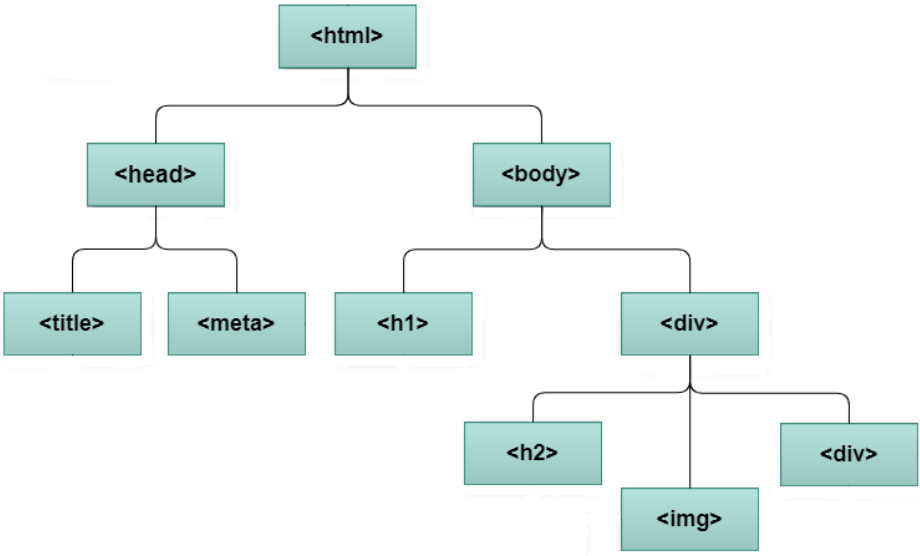

Рис. 4.1. Пример простой структуры DOM-дерева HTML-документа

Потомок – элемент, расположенный внутри одного или более типов элементов. Например, на рис. 4.1 <body> является потомком <html>, а элемент <h1> – потомком одновременно <body> и <html>.

Родительский элемент – элемент, непосредственно связанный с другими элементами более низкого уровня и находящийся на дереве выше их. На рис. 4.1 <html> является родительским только для <head> и <body>. Тег – родительский для другого , <h2> и <img>.

Дочерний элемент – элемент, непосредственно подчиненный другому элементу более высокого уровня. На рис. 4.1 только элементы <h1> и являются дочерними по отношению к <body>.

Сестринский элемент – элемент, имеющий общий родительский элемент с рассматриваемым. В этом случае рассматриваемый и сестринский элемент – элементы одного уровня. На рис. 4.1 <head> и <body> – элементы одного уровня и так же, как и <title> и <meta>, являются сестринскими.

Далее, внутри элемента <html> располагаются теги заголовка <head></head> и тела <body></body> документа.

В заголовке HTML-документа содержатся теги и текст, представляющие собой техническую информацию о веб-странице (заголовок, описание, кодировка, ключевые слова для поисковых систем, подключаемые стили оформления страницы и скрипты и др.). Эта информация не показывается напрямую на веб-странице при ее отображении в браузере (за исключением данных из элемента <title>), но она помогает понять браузеру, как следует обрабатывать и отображать веб-страницу.

В заголовок HTML-документа обязательно помещается элемент <title>, который содержит заголовок веб-страницы, отображаемый в веб-браузере при ее просмотре (часто – в названии вкладки веб-браузера). Другие элементы, обычно дочерние по отношению к <head>, являются необязательными. К ним относятся <meta>, <style>, <link>, <script>.

С помощью элементов <meta> (не имеет закрывающего тега) можно задать описание содержимого веб-страницы, автора документа, ключевые слова для поисковых систем, кодировку символов и др. В заголовке HTML-документа может содержаться несколько элементов <meta>, поскольку в зависимости от используемых атрибутов (name, content, http-equiv, charset) и их значений они несут разную информацию. Так, например, для указания авторства документа можно использовать конструкцию следующего вида: <meta name="author" content ="Ivan Ivanov">, для описания ключевых слов – <meta name="keywords" content="веб-картография, ГИС, html-разметка">, для установки кодировки UTF-8 на странице – <meta http-equiv="Content-Type" content="text/html; charset=utf-8">, для автоматической перезагрузки страницы через 30 сек. — <meta http-equiv="refresh" content="30">.

Внутри элемента <style> задаются стили на языке CSS, используемые для оформления внешнего вида HTML-документа. Применение стилей, написанных на языке CSS, будет рассмотрено далее.

Элемент <link> (не имеет закрывающего тега) используется для подключения внешних файлов таблиц стилей, написанных на языке CSS (имеют расширение .css), а также файлов шрифтов. Количество элементов <link> не ограничено. При подключении внешнего файла используется относительная ссылка, определяющая положение файла по отношению к HTML-документу. Пример подключения файла стилей style.css может выглядеть следующим образом:

<link rel="stylesheet" href="style.css" type="text/css">.

В этом примере при помощи атрибута href указывается ссылка на внешний файл стилей, и по ней можно определить, что файл style.css расположен в одном каталоге с HTML-документом, в котором он подключается.

Элемент <script> необходим для описания скриптов (программ), используемых при работе веб-страницы. Внутри данного тега может содержаться как сам текст скрипта на определенном языке (чаще всего JavaScript), так и ссылка на внешний файл с программой. Для указания языка, на котором написан скрипт, используется атрибут type, например <script type="text/javascript"></script>. При этом в HTML5 атрибут type можно опустить, т. к. он по умолчанию принимает значение text/javascript, если не указан явно. При подключении внешнего файла скрипта, как и при подключении стилей, используется относительная ссылка, которая указывается при помощи атрибута src.

В теле HTML-документа (<body>) располагается все содержимое веб-страницы, которое отображается при ее просмотре в браузере. Это содержимое может быть представлено различными элементами, которые условно делятся на несколько типов: элементы для группировки содержимого (блочные элементы), строчные элементы, теги для формирования списков, теги для формирования таблиц, теги для фреймов, теги для создания форм, теги для встраиваемого содержимого. Далее представлены основные часто используемые элементы. Некоторые из этих элементов можно использовать только при написании веб-страниц на HTML5.

Для группировки содержимого можно использовать следующие основные элементы:

- – определяет параграфы (абзацы) в тексте;
- <address></address> – предназначен для хранения информации об авторе и может включать в себя любые элементы HTML вроде ссылок, текста, выделений и т. д.;
- <hr> – рисует горизонтальную линию, которая по своему виду зависит от используемых атрибутов;
- <pre></pre> – определяет блок предварительно форматированного текста (выводит текст с пробелами между словами и переносами);
- <blockquote></blockquote> – предназначен для выделения длинных цитат внутри документа;
- <h1></h1>,...,<h6></h6> – определяют текстовые заголовки разного уровня, которые показывают относительную важность секции, расположенной после заголовка (заголовок 1 – самый крупный, 6 – самый мелкий);
- – универсальный блочный контейнер для разделов HTML-документа, применяется в тех случаях, где нужны блочные элементы без дополнительных свойств;
- <figure></figure> – независимый контейнер для такого контента, как изображения, диаграммы и т. п.;
- <figcaption></figcaption> – заголовок для элемента <figure>;

- <main></main> – контейнер для уникального основного содержимого HTML-документа (веб-страницы сайта);
- <section></section> – задает раздел документа, может применяться для блока новостей, контактной информации, глав текста, вкладок в диалоговом окне и др.;
- <header></header> – задает «шапку» сайта или раздела, в которой обычно располагается заголовок (название сайта);
- <footer></footer> – задает «подвал» сайта или раздела, в котором обычно располагается информация об авторе сайта, дата документа, контактная информация, правовая информация (копирайты);
- <nav></nav> – задает навигацию по сайту. Если на странице несколько блоков ссылок, то в элемент <nav> обычно помещают приоритетные ссылки.

К строчным элементам оформления веб-страницы относятся:

- <a> – элемент, предназначенный для создания ссылок на другие веб-страницы или на закладки (якори) внутри текущей страницы;
- – позволяет установить жирное начертание для текста;
- <big></big> – увеличивает размер текста на одну единицу по сравнению с обычным текстом;
-
 – устанавливает перевод текста на новую строку в том месте, где находится данный тег;
- – применяется для задания акцента определенному тексту (в браузерах такой текст обычно отображается курсивом);
- – устанавливает выделение текста курсивом;
- <del></del> – используется для выделения текста, который был удален в новой версии документа;
- <ins></ins> – предназначен для выделения текста, который был добавлен в новую версию документа;
- <s></s> – применяется для зачеркивания текста;
- <small></small> – уменьшает размер текста на одну единицу по сравнению с обычным текстом;
- – предназначен для определения строчных элементов документа. В отличие от блочных элементов этот тег позволяет выделять часть информации внутри других тегов и устанавливать для нее свой стиль;
- – применяется для задания акцента определенному тексту (в браузерах такой текст обычно отображается жирным шрифтом);
- – предназначен для отображения шрифта в виде нижнего индекса;

- – предназначен для отображения шрифта в виде верхнего индекса;
- <u></u> – применяется для подчеркивания текста.

К элементам формирования списков относятся:

- – используется для создания маркированного списка;
- – используется для создания нумерованного списка;
- – определяет отдельный элемент маркированного или нумерованного списка внутри тегов <ul> и <ol>;
- <dd></dd>, <dt></dt> и <dl></dl> – набор тегов для создания списка определений разных терминов. Каждый такой список начинается с контейнера <dl>, куда входит тег <dt>, создающий термин, и тег <dd>, задающий определение этого термина.

Для формирования таблиц используются теги:

- <table></table> – тег для создания таблицы, состоящей из строк и ячеек, которые задаются с помощью тегов <tr> и <td>;
- <tr></tr> – используется для создания строк таблицы;
- <td></td> – используется для создания ячеек таблицы внутри строк;
- <th></th> – используется для создания заголовков столбцов таблицы;
- <caption></caption> – используется для создания заголовка таблицы.

Следующая группа HTML-элементов предназначена для создания фреймов на веб-страницах. Фреймы необходимы для разделения веб-страницы на отдельные области, которые обычно содержат навигацию по сайту и его контент. В каждую из таких областей загружается самостоятельные веб-страница или документ, определяемые с помощью тега <frame>. Механизм фреймов позволяет открывать веб-страницу (или документ) в одном фрейме по ссылке, нажатой в совершенно другом фрейме. Для создания фреймов используются следующие элементы:

- <frameset></frameset> – определяет структуру фреймов на веб-странице, т. е. задает способ разметки страницы на отдельные области (данный тег может заменить собой элемент <body> на веб-странице);
- <frame></frame> – определяет свойства отдельного фрейма, на которые делится веб-страница. Этот элемент должен располагаться в контейнере <frameset>;
- <iframe></iframe> – создает плавающий фрейм (контейнер), который находится внутри обычного документа, он позволяет загружать в область заданных размеров любые другие независимые документы.

Еще одна группа элементов необходима для создания форм на веб-страницах (рис. 4.2). Формы предназначены главным образом для обмена данными между пользователями и сервером. Стоит отметить, что формы являются довольно сложным элементом интерфейса пользователя.

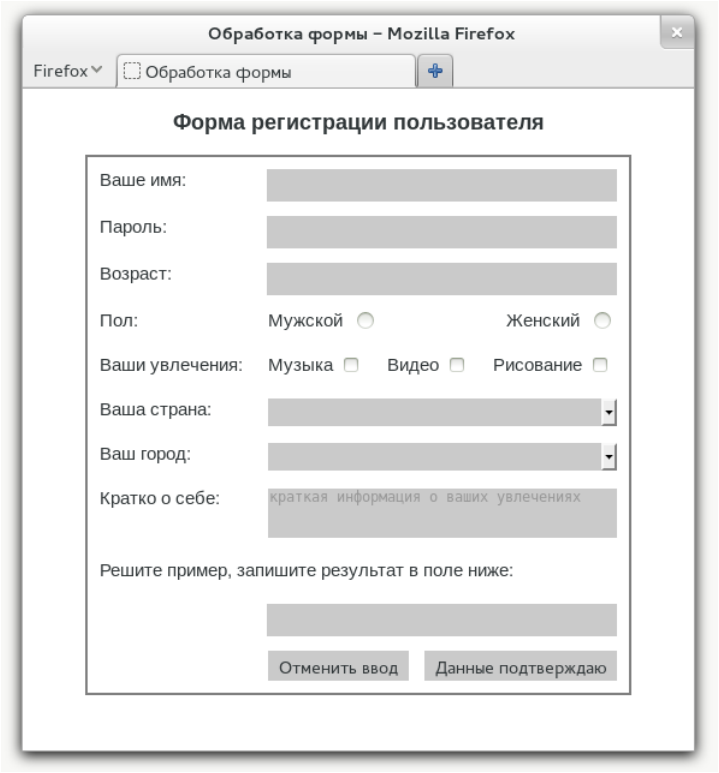

Рис. 4.2. Пример формы на веб-странице

К элементам, используемым для создания форм, можно отнести:

- `<form></form>` – предназначен для создания формы на веб-странице;
- `<input></input>` – является одним из разносторонних элементов формы, позволяет создавать разные элементы интерфейса и обеспечивает взаимодействие с пользователем. Тег `<input>` используется преимущественно для создания текстовых полей, различных кнопок, переключателей и флажков, которые задаются при помощи атрибута `type`;
- `<select></select>` – позволяет создать элемент интерфейса в виде раскрывающегося (выпадающего) списка (меню), а также список с одним или множественным выбором;
- `<option></option>` – определяет отдельные пункты списка, создаваемого с помощью элемента `<select>`;
- `<textarea></textarea>` – представляет собой элемент формы для создания области, в которую можно вводить несколько строк текста. В отличие от тега `<input>` в текстовом поле допустимо делать

переносы строк, которые сохраняются при отправке данных на сервер;

- <label></label> – устанавливает связь между определенной меткой, в качестве которой обычно выступает текст, и элементом формы (<input>, <select>, <textarea>). Такая связь необходима, чтобы изменять значения элементов формы при нажатии курсором на текст. Кроме того, с помощью <label> можно устанавливать горячие клавиши на клавиатуре и переходить по ним на активный элемент подобно ссылкам;
- <button></button> – создает на веб-странице кнопки, которые позволяют запускать выполнение скриптов или осуществлять отправку данных на сервер из форм.

Для встраивания какого-либо содержимого на веб-страницу, например мультимедиа или других файлов, используется ряд элементов:

- <img> – используется для отображения на веб-странице изображений в графических форматах;
- <map></map> – предназначен для создания карты изображений, позволяющей хранить в одном изображении несколько ссылок, служит контейнером для элементов <area>;
- <area> – определяют активные области для карт-изображений <map>. Такие области устанавливают невидимые зоны на изображении, являющиеся ссылками на HTML-документы;
- <embed> – используется для загрузки и отображения объектов (например, видеофайлов, flash-роликов, некоторых звуковых файлов и т. д.), которые исходно браузер не может распознать. Как правило, такие объекты требуют подключения к браузеру специальных модулей (плагинов) или запуска вспомогательной программы;
- <object></object> – сообщает браузеру, как загружать и отображать объекты, которые исходно браузер не может распознать;
- <article></article> – задает содержание сайта вроде новости, статьи, записи блога, форума или др.;
- <audio></audio> – добавляет и воспроизводит аудиозаписи на веб-странице и управляет их настройками;
- <video></video> – добавляет и воспроизводит видеозаписи на веб-странице и управляет их настройками.

При написании документов на языке HTML часто возникает необходимость добавления комментариев в код. Закомментированный текст внутри HTML-разметки выделяется следующим образом: <!-- текст комментария -->.

Как уже было отмечено ранее, для описания внешнего вида элементов веб-страниц используется язык CSS (Cascading Style Sheets – каскадные таблицы стилей). Хотя этот язык преимущественно используется как средство

описания, оформления внешнего вида веб-страниц, написанных с помощью языков разметки HTML и XHTML, он может также применяться к любым XML-документам, например: к SVG (Scalable Vector Graphics – масштабируемая векторная графика) или XUL (XML User Interface Language – язык разметки для создания динамических пользовательских интерфейсов на основе XML).

Каскадные таблицы стилей описывают правила форматирования элементов HTML-документа с помощью свойств и их допустимых значений. Для каждого элемента можно использовать ограниченный набор свойств, а остальные из них не будут оказывать на него никакого влияния. Среди таких свойств можно назвать цветовое оформление элементов, задание шрифтов, описание размера и расположения элемента на веб-странице и мн. др.

Конструкция для объявления стиля для элементов веб-страницы при помощи CSS состоит из двух частей: селектора и блока объявлений (рис. 4.3).

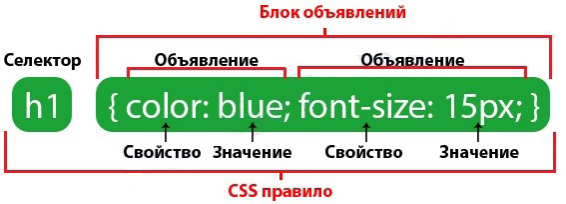

Рис. 4.3. Объявление стилей при помощи CSS

Селектор сообщает браузеру, какой именно HTML-элемент веб-страницы нужно форматировать, а блок объявлений – как его форматировать. Блок объявлений помещается в фигурные скобки и включает в себя одно или несколько объявлений, разделяемых знаком «;». Каждое объявление также состоит из двух частей: имени свойства (например, color – свойство, обозначающее цвет текста) и его значения (например, green – зеленый цвет). Список всех свойств, которые используются в CSS, и их возможные значения можно изучить на ресурсе http://htmlbook.ru [15] или на другом аналогичном сайте.

В качестве селекторов могут выступать названия элементов HTML-документа и их атрибутов, а также имена их идентификаторов и классов, которые задаются разработчиком веб-страницы при помощи атрибутов id и class соответственно. Также при помощи селекторов можно задавать не только одиночные элементы, но и элементы, являющиеся потомками, дочерними или сестринскими. В зависимости от элементов, которым задается стиль, выделяется несколько видов селекторов.

Универсальный селектор, обозначающийся символом «*», относится сразу ко всем элементам веб-страницы. Например, стиль *{margin:0; background-color: red;} обнулит внешние отступы для всех элементов сайта, а для их фона применит красный цвет.

Селектор элемента позволяет форматировать сразу все элементы веб-страницы. Например, правило h1{font-family: Arial; font-size: 11pt;} задаст общий стиль для всех заголовков уровня <h1>.

Селекторы класса позволяют задавать стили для одного и более элементов с одинаковым именем класса, размещенных в разных местах страницы или на разных страницах сайта. Имя класса для элементов зада-

ется с помощью атрибута class, например . В приведенном примере блочный элемент имеет имя класса mapLegend. В CSS селектор класса обозначается следующим образом: .имя_класса{объявления}. Например, стиль .mapLegend{border-color: black;} сделает у всех элементов класса mapLegend внешнюю границу черного цвета. Стоит также отметить, что один HTML-элемент может относиться сразу к нескольким классам, при этом в значении атрибута class имена этих классов указываются через пробел.

Селектор идентификатора позволяет форматировать один конкретный элемент веб-страницы, имеющий уникальный (встречающийся только один раз на веб-странице) идентификатор, имя которого задается значением атрибута id, например . В CSS селектор класса обозначается следующим образом: #имя_идентификатора{объявления}. Например, стиль #mapLayers{width: 300px; height: 100px;} задает элементу с идентификатором mapLayers размеры 100 пикселей по высоте и 300 пикселей по ширине.

Селекторы потомков (контекстные селекторы) применяют стили к элементам, расположенным внутри элемента-контейнера. При задании стиля в селекторе сначала указывается элемент-предок, затем через пробел – потомок. Например, стиль ul li{font-style: italic;} выберет все элементы <li>, являющиеся потомками всех элементов <ul> в HTML-документе. Если нужно отформатировать потомки определенного элемента, этому элементу следует задать стилевой класс или идентификатор. Примеры таких конструкций:

- div#paragraph1 p.note {color: red;} – этот стиль будет применен ко всем элементам <p> с классом note, которые являются потомками блочного элемента , имеющего идентификатор paragraph1;
- p.first a{color: green;} – данный стиль будет применен ко всем ссылкам, потомкам абзаца с классом first;
- p. first a{color: green;} – если добавить пробел, то будут стилизованы ссылки, расположенные внутри любого тега класса first, который является потомком элемента <p>;
- .first a{color: green;} – данный стиль применится к любой ссылке, расположенной внутри другого элемента, обозначенного классом first.

Селекторы дочерних элементов. Дочерний элемент является прямым потомком содержащего его элемента. У одного элемента может быть несколько дочерних элементов, а родительский элемент у каждого элемента только один. Селектор дочерних элементов позволяет применить стили только если дочерний элемент идет сразу за родительским элементом и между ними нет других элементов, т. е. если дочерний элемент больше ни во что не вложен. Конструкция селектора дочерних элементов в CSS выглядит следующим образом:

```
родительский_элемент > дочерний_элемент{объявления}.
```

Например, стиль div#list1 > li{color: yellow} будет применен ко всем элементам li, являющимся дочерними по отношению к элементу с идентификатором list1.

Селекторы сестринских элементов позволяют выбрать элементы из группы элементов одного уровня, т. е. имеющих общего родителя. В конструкции этого селектора в CSS сестринские элементы разделяются знаками «+» или «~». Например, стиль h1 + p{font-size: 14pt;} будет применен ко всем первым абзацам, идущим за любым элементом <h1> веб-страницы, а стиль h1 ~ p{font-size: 14pt;} применится ко всем абзацам, являющимся сестринскими по отношению к любому элементу <h1>.

Селекторы атрибутов форматируют элементы на основе имени атрибута или значения атрибута. Конструкции селекторов атрибутов могут быть различными, но стоить отметить, что имя атрибута в селекторе указывается в квадратных скобках – []. Например, стиль [href]{color: red;} будет применен ко всем элементам, у которых задан атрибут href, а стиль img[title="рисунок"] {height: 300px;} отформатирует все изображения, которые имеют название «рисунок».

Селекторы псевдоклассов. Псевдоклассы – это классы, фактически не прикрепленные к HTML-тегам. Они позволяют применить CSS-правила к элементам при совершении события или к элементам, подчиняющимся определенному правилу. Также псевдоклассы дают возможность задавать стиль дочерним элементам в соответствии с параметром, указанном в круглых скобках. Перечень всех псевдоклассов можно посмотреть в CSS-справочнике [15]. Например, CSS-правило a:hover{font-family: Times New Roman;} позволяет изменять семейство шрифта при наведении указателем мыши на гиперссылку, а стиль table:nth-child(even){background-color: green;} позволит все четные строки таблицы, которые являются дочерними элементами, отобразить зеленым цветом.

Селектор псевдоэлементов задает стиль элементов, не определенных в дереве элементов документа, а также генерировать содержимое (с помощью свойства :content), которого нет в исходном коде текста. Синтаксис использования псевдоэлементов следующий:

```
селектор:Псевдоэлемент{объявления}.
```

Существует пять псевдоэлементов. Они позволяют выбирать первую букву (:first-letter), первую строку (:first-line), применять стиль к выделенному пользователем тексту (:selection), а также добавлять содержимое и применять стили до (:before) и после (:after) определенного элемента. Например, стиль p:first-letter{font-family: Arial; color: red;} отформатирует первую букву всех абзацев под шрифт Arial красного цвета.

Стоит отметить, что один и тот же стиль можно сразу применить к нескольким элементам одновременно. Для этого в левой части CSS-правила необходимо через запятую перечислить селекторы всех элементов, к которым применяется стиль.

Чтобы использовать стили, написанные на языке CSS, на конкретной веб-странице, их следует расположить либо в HTML-документе, либо в отдельном текстовом файле с расширением .css. Всего существует четыре способа внедрения таблиц стилей в HTML-документ (рис 4.4).

При использовании первого способа (рис 4.4, А)) таблицы стилей размещаются в отдельном файле и затем подключаются к HTML-документу при помощи тега <link>, у которого в атрибуте href задается относительная ссылка

на файл со стилями (т. е. указывается расположение файла относительно HTML-документа). При этом тег <link> размещается в заголовке (<head>) HTML-документа (рис 4.4).

Второй способ (рис 4.4, Б)) внедрения таблиц стилей на веб-страницу также подразумевает их размещение в отдельном файле, который может быть подключен к HTML-документу с помощью директивы @import url(). Данную директиву необходимо разместить между тегами <style> и </style> в заголовке HTML-документа. В скобках директивы указывается относительная ссылка на файл .css (рис 4.4).

Третий способ (рис 4.4, В)) позволяет описать таблицы стилей в самом HTML-документе внутри тегов <style> и </style>, расположенных в заголовке HTML-документа (рис 4.4).

Четвертый способ (рис 4.4, Г)) позволяет описывать таблицы стилей у конкретного элемента (тега) HTML-документа посредством атрибута style (рис 4.4).

Применение CSS к HTML-документам основано на принципах наследования и каскадирования. Принцип наследования заключается в том, что свойства CSS, объявленные для элементов-предков, почти всегда наследуются элементами-потомками.

Принцип каскадирования применяется в случае, когда какому-то элементу HTML одновременно поставлено в соответствие более одного правила CSS, т. е. когда происходит конфликт значений этих правил. Чтобы разрешить такие конфликты, вводятся правила приоритета. Наиболее низким приоритетом обладают стили браузера, а наиболее высоким – стили, заданные с использованием описанных ранее селекторов. При этом приоритеты между селекторами определяются при помощи расчета показателя специфичности. Чем больше этот показатель, тем выше приоритет стиля. Показатель специфичности состоит из четырех параметров: a) встроенный стиль или из внешнего файла (если встроенный, то a = 1), b) равен количеству идентификаторов, c) равен количеству классов, псевдоклассов и селекторов атрибутов, d) равен количеству селекторов элементов и псевдоэлементов (рис. 4.5).

Но самым высоким приоритетом обладают стили, объявленные с помощью сопроводительного правила !important, которое добавляется к значению определенного свойства CSS-правила.

Создавать веб-страницы при помощи языка HTML и таблицы стилей CSS можно в любом текстовом редакторе или среде разработки IDE (Integrated Development Environment). Код можно писать даже в стандартном «Блокноте» Windows, но его использовать неудобно, т. к. в нем нет подсветки синтаксиса указанных языков. Поэтому для этих целей лучше использовать редакторы и среды, имеющие эту подсветку. Подсветка синтаксиса языков позволяет избегать ошибок и делает чтение кода более удобным. В качестве примеров таких редакторов можно привести NotePad++, Atom и др., а также среды разработки Visual Studio Code, NetBeans и др. Перечисленные продукты имеют бесплатные лицензии.

А)

```
<!DOCTYPE html>
<html>
   <head>
      .....
      <link rel="stylesheet" href="style.css">
   </head>
   <body>
      .....
   </body>
</html>
```

Б)

```
<!DOCTYPE html>
<html>
   <head>
      .....
      <style media="all">
         @import url(style.css);
      </style>
   </head>
</html>
```

В)

```
<!DOCTYPE html>
<html>
   <head>
      .....
      <style>
         body {
            color: red;
         }
      </style>
   </head>
   <body>
      .....
   </body>
</html>
```

Г)

```
<!DOCTYPE html>
<html>
   <head>
      .....
   </head>
   <body>
      <p style="font-size: 20px; color: green; font-family: arial">
         .....
      </p>
   </body>
</html>
```

Рис. 4.4. Способы внедрения таблиц стилей CSS в HTML-документ: А) размещение стилей в отдельном файле с подключением к HTML-документу при помощи тега <link>; Б) размещение стилей в отдельном файле с подключением к HTML-документу посредством директивы @import url(); В) описание стилей в HTML-документе в тегах <style>; Г) описание стилей у конкретного элемента (тега) HTML-документа посредством атрибута style

Селектор	a, b, c, d	Число
span	0, 0, 0, 1	1
div span	0, 0, 0, 2	2
.class	0, 0, 1, 0	10
#id span	0, 1, 0, 1	101
#id .class	0, 1, 1, 0	110

Рис. 4.5. Пример таблицы специфичности разных селекторов стилей

4.2. Язык программирования JavaScript

Язык программирования JavaScript (JS) может использоваться для разработки как серверной, так и клиентской частей веб-приложений.

При создании серверной части веб-приложений на JavaScript наиболее часто используется программная платформа Node.js, которая превращает JavaScript из узкоспециализированного языка в язык общего назначения. Для исполнения JavaScript на сервере используется специальная программа, которая называется движком JavaScript. Движок JavaScript предназначен для компиляции, т. е. перевода программы, написанной на языке программирования, в машинный код перед ее исполнением. Платформа Node.js использует движок JavaScript, который называется V8.

При создании клиентской части веб-приложений JavaScript нашел более широкое применение, и на данный момент он является самым популярным средством ее разработки. Это связано с тем, что JavaScript позволяет придавать веб-страницам интерактивность и написанные на нем программные коды могут исполняться любыми веб-браузерами без установки дополнительных модулей и расширений, поскольку в них уже есть встроенные JavaScript-движки (например, браузеры Google Chrome и Opera имеют JavaScript-движок V8), способные считывать и исполнять JavaScript-код. Следовательно, веб-браузеры являются интерпретаторами языка JavaScript.

При использовании в веб-браузерах JavaScript представляет собой язык сценариев, или скриптов. Скрипт – это программный код, который не требует компиляции перед выполнением.

Особенностью веб-браузеров, упрощающей разработку веб-приложений с использованием JavaScript, является то, что они имеют в своем составе так называемую консоль разработчика (рис. 4.6). Она позволяет разработчику находить ошибки при просмотре веб-страницы в браузере, а также производить отладку кода. При этом все ошибки выводятся в консоль в виде сообщений с указанием строки кода, где произошла ошибка. Для запуска консоли в каждом браузере используются собственные горячие клавиши. Например, в Google Chrome клавиша F12, а в Opera – сочетание клавиш Ctrl+Shift+I. Также консоль может использоваться разработчиком для вывода в нее какой-либо информации, например, значения переменной, или результата выполнения функции и т. п. Вывод информации в консоль осуществляется при помощи команды `console.log();`.

Для написания кода на языке JavaScript необходимо использовать редакторы кода. В основном они те же, что используются для создания веб-страниц. Это могут быть обычные редакторы (NotePad++, Atom и др.), а также среды разработки IDE, например: Visual Studio Code, NetBeans, Visual Studio (бесплатная версия Community) и др. Преимущество использования IDE состоит в том, что она позволяет загружать проект, который может состоять из множества файлов, переключаться между этими файлами, предлагает автодополнение по коду всего проекта (например, дополнение названий функций, переменных и поддерживаемых ими свойств и т. д.), а также может быть интегрирована с системой контроля версий файлов проекта (например, такой как Git), средой для тестирования и другими инструментами на уровне всего проекта.

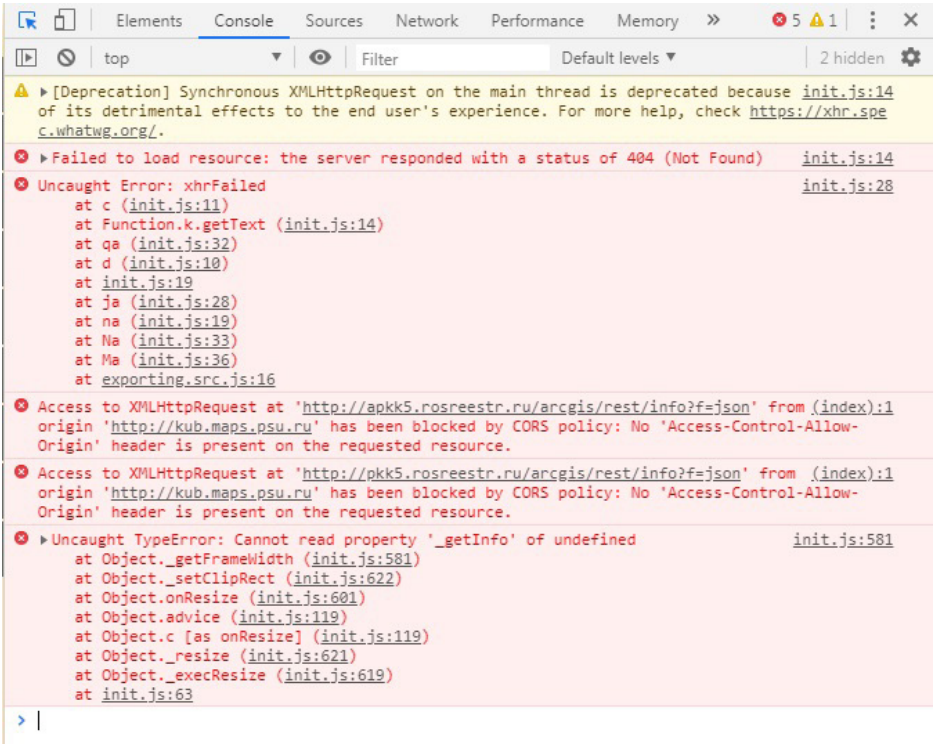

Рис. 4.6. Общий вид консоли разработчика веб-браузера

Применение JavaScript-сценариев на веб-странице возможно реализовать несколькими способами.

Во-первых, скрипты можно поместить непосредственно в HTML-документ внутри элемента <script>, который рекомендуется размещать в заголовке веб-страницы (<head>). Контейнеров <script> в одном HTML-документе может быть произвольное число. Для указания типа скриптов обычно используется атрибут type тега <script> с соответствующим значением: <script type="text/javascript">. Однако указывать данный атрибут необязательно, т. к. по умолчанию его значением является JavaScript.

Классическим примером может выступать скрипт, выводящий модальное окно (окно в графическом интерфейсе пользователя, которое блокирует работу пользователя с приложением до тех пор, пока пользователь не закроет его) с классической надписью "Hello, World!" внутри браузера:

```
<script type="text/javascript">
    alert('Hello, World!');
</script>
```

Во-вторых, JavaScript-сценарии можно применять в виде обработчиков событий. Как упоминалось в разд. 4.1, у HTML-элементов имеются события (подобно атрибутам), которые срабатывают при определенных условиях, например при наведении указателя мыши на элемент или при клике

по нему. Можно добавить необходимое событие в HTML-элемент, а в качестве значения этого события указать функцию, написанную на JavaScript. При этом функцию можно описать в заголовке HTML-документа внутри элемента <script>, как указано в предыдущем абзаце. Например, в конструкции <button onclick='alert("нажал кнопку")'>Кнопка</button> у кнопки задано событие клика по ней (onclick), при выполнении которого выведется модальное окно браузера с надписью «Нажал кнопку».

Третий способ использования JavaScript-сценариев на веб-странице предполагает написание скрипта в отдельном файле, имеющем расширение .js, и подключение его к HTML-документу при помощи абсолютной или относительной ссылки на данный файл, указанной в атрибуте src элемента <style>. Конструкция подключение файла выглядит следующим образом:

```
      <head>
            <script type=»text/javascript»
src=»Путь_к_файлу_со_скриптом»>
            </script>
      </head>
```

Стоит также упомянуть, что у элемента <script> имеются и другие важные атрибуты, такие как async и defer. Атрибут async позволяет браузеру запускать скрипт асинхронно, это значит, что указанный в атрибуте src файл будет выполняться без ожидания загрузки и отображения веб-страницы. В то же время и страница не ожидает результата выполнения скрипта, а продолжает загружаться как обычно. Наличие атрибута defer откладывает выполнение скрипта до тех пор, пока вся веб-страница не будет загружена.

Далее будут описаны основные особенности работы с языком программирования JavaScript.

Язык JavaScript поддерживает объектно ориентированный стиль программирования. Это значит, что скрипты можно представить как совокупность объектов, каждый из которых является экземпляром определенного класса (элемент, описывающий абстрактный тип данных и его реализацию), а классы образуют иерархию наследования. У каждого объекта есть применимые к нему свойства (атрибуты) и функции (методы). Объектом выступает любая переменная.

Язык JavaScript поддерживает несколько *типов переменных*: числовой (number), строковый (string), логический (boolean), нулевой (null), неопределенный (undefined), массив (array), объект (object), функция (function).

Числовые переменные бывают двух типов: целые числа и числа с плавающей запятой. Строковые переменные используются для хранения набора любых символов, заключенных в двойные или одинарные кавычки. Логические переменные имеют только два значения (true (истина) и false (ложь)) и используются для проверки условий. Переменные нулевого типа имеют одно значение (null), которое применяется для представления несуществующих объектов. Неопределенный тип переменной (undefined) означает отсутствие первоначального значения переменной, а также несуществующее свойство объекта. Этот тип автоматически присваивается при создании пустой переменной. Массив представляет собой индексированный набор элементов и используется для хранения нескольких значений разных типов. Он обознача-

ется при помощи квадратных скобок, а входящие в него элементы указываются через запятую (пример массива – [1, 5, 25.7, "word"]). Нумерация элементов массива начинается с 0 (в примере нулевой элемент массива равен 1, а третий – «word»). Объекты используются для хранения коллекций различных значений и задаются при помощи фигурных скобок (пример объекта – {'first': 23, 'second': 50}). Функции представляют собой коллекции инструкций. Более подробно функции будут описаны далее в настоящем разделе.

Переменные в JavaScript можно объявлять при помощи трех ключевых слов: let, var, const. При объявлении переменной можно сразу задать и ее значение. Слово const используется для объявления констант, т. е. неизменяемых переменных, например const beginDate = "01.01.2019". Попытка изменения (перезаписи) значения (в примере значения – 01.01.2019) такой переменной приведет к ошибке.

Изменяемые переменные, значения в которых можно менять неограниченное число раз, объявляются при помощи слов let и var. Например, объявление переменных с присваиванием им значений var x = 75 или let y = "ГИС" (переменной x присвоено число, y – строка). Как уже было ранее упомянуто, если объявить переменную без присваивания значения, то она будет иметь тип undefined.

Между let и var есть некоторые различия. При объявлении любой переменной в JavaScript она помещается в область видимости. Если переменная объявлена с помощью let, то ее область видимости ограничивается блоком, в котором она объявлена, т. е. ее можно использовать только в этом блоке. Переменная, объявленная через var, имеет области видимости внутри функции, в которой она объявлена, или внутри всего скрипта (глобальная область видимости), если она создана вне какой-либо функции. Например, код

```
if (true) {
        var test = true;
}
alert(test);
```

запустит модальное окно с надписью true. А код

```
if (true) {
        let test = true;
}
alert(test);
```

выдаст ошибку, поскольку переменная test объявлена в блоке условий (if) и за его пределами не видна.

Еще одним отличием является то, что при помощи слова var внутри скрипта можно несколько раз создавать переменную с одним и тем же именем. В больших скриптах разработчик может не заметить, что у него уже есть переменная с некоторым именем, и создать ее повторно. Это может привести к некорректной работе программы. Использование ключевого слова let для объявления переменных позволяет избежать подобной ошибки, т. к. с помощью этого слова можно создать переменную с определенным именем внутри скрипта только один раз.

Использование слова let является более новым способом создания переменных, чем var, и разработчики рекомендуют по возможности использовать именно его [16].

При задании имен переменных в JavaScript существуют следующие ограничения: имя переменной должно содержать только латинские буквы, цифры и символы «$» и «_», имя переменной не должно начинаться с цифр.

Стоит отметить, что JavaScript является нетипизированным языком, поэтому тип данных для конкретной переменной при ее объявлении указывать не нужно (как в типизированных языках). Тип данных переменной зависит от значений, которые она принимает. Тип переменной может динамически изменяться в процессе совершения операций с данными, в том числе и при перезаписи значений переменной. Преобразование типов выполняется автоматически в зависимости от того, в каком контексте они используются. Однако существует строго типизированный язык TypeScript, который является надстройкой над JavaScript. Код, написанный на TypeScript, может легко компилироваться в код на JavaScript.

Поскольку каждая переменная является объектом, то у нее имеются свойства и функции, вызов которых осуществляется через точку. Например, есть массив let x = [1, 2, 3, 4, 5], он обладаем свойством – длиной (количеством элементов) массива (length). Чтобы вывести в модальное окно браузера длину массива, можно записать код

```
let x = [1, 2, 3, 4, 5];
let lenMassive = x.length;
alert(lenMassive);
```

У языка JavaScript есть и другие синтаксические особенности, заключающиеся в правильном написании выражений, операторов, инструкций, функций и т. д. Стоит отметить, что язык JavaScript имеет общепринятый стандарт написания кода, который называется ECMAScript (сокращенно – ES). Этот стандарт постоянно дополняется. На момент написания данного пособия последним принятым стандартом является ES9 (или ES2018). При обновлении стандарта к возможностям языка добавляются новые операторы, методы, стили написания выражений и т. п. Более подробно синтаксические особенности языка JavaScript можно изучить, например, в онлайн-учебнике, расположенном по адресу https://learn.javascript.ru.

Выражение – это фрагмент программы, который интерпретатор может вычислить, вернув его значение. Самым простым выражением является объявление переменной, но чаще всего выражения строятся при помощи операторов, которые используются в арифметических действиях, логических выражениях, операциях сравнения и присваивания и др. Разные выражения обычно пишут в разных строках и отделяют друг от друга знаком «;» (см. пример выше).

Список используемых в JavaScript операторов в порядке убывания их приоритета:

- . (доступ к свойству), [] (доступ к свойству), () (вызов функции), new (создание нового объекта);
- ++ (инкремент – увеличивает переменную на 1 единицу),

-- (декремент – уменьшает переменную на 1 единицу), - (унарный минус – меняет знак переменной на противоположный), + (унарный плюс), ~ (поразрядное дополнение), ! (логическое дополнение), delete (удаление свойства), typeof (определение типа данных в переменной), void (возврат неопределенного значения);

- * (умножение), / (деление), % (остаток от деления);
- + (сложение), - (вычитание), + (конкатенация строк);
- << (побитовый сдвиг влево), >> (побитовый сдвиг вправо с расширением знакового разряда), >>> (побитовый сдвиг вправо с дополнением нулями);
- < (меньше), <= (меньше или равно), > (больше), >= (больше или равно), instanceof (проверка типа объекта), in (проверка наличия свойства);
- == (проверка на равенство), != (проверка на неравенство), === (проверка на идентичность), !== (проверка на неидентичность);
- = (присваивание), *=, /=, +=, -=, <<=, >>=, >>>=, &=, ^=, |= (присваивание с операцией).

Инструкция в JavaScript – это фрагмент программы, приказывающий браузеру что-либо сделать. Такие фрагменты включают в себя условные инструкции, циклы (конструкции итерации), функции и др. Основные инструкции:

- *конструкция проверки условия* (if else) выглядит следующим образом:

```
if (выражение-условие){инструкция_1}
else{инструкция_2}
```

Для проверки *нескольких условий* используется конструкция switch-case:

```
switch(выражение){
case_1: инструкция_1;
        break;
case_2: инструкция_2;
        break;
        …
case_x: инструкция_x;
        break;
default: инструкция_x+1
}
```

Инструкция switch последовательно сравнивает записанное в ней значение выражения со всеми вариантами, перечисленными во всех case. Если установлено соответствие, то switch выполняется до ближайшего break (прерывает дальнейшее выполнение кода), в default указываются остальные варианты. Пример ниже вернет значение «Как раз!» в модальное окно браузера:

```
var a = 2+2;
switch (a){
case 3:
       alert('Мало');
       break;
case 4:
       alert('Как раз!');
       break;
case 5:
       alert('Много');
       break;
default:
       alert('Такие значения мне не известны');
}
```

- *конструкции циклов* используются для последовательного выполнения одного и того же действия со списком объектов (с элементами массивов). Циклы бывают трех типов: циклы по параметру (for), циклы с предусловием (while) и циклы с постусловием (do/while).

Цикл по параметру записывается следующим образом:

```
for (начало, условие, шаг){инструкции}.
```

Например, цикл

```
var r = [1,2,3,4,5];
for (var i=0; i<t.length; r++) {
       console.log(i[r]);
}
```

выведет в консоль браузера поочередно значения всех элементов массива r.

Цикл с предусловием:

```
начало; while(условие){инструкции}.
```

Например,

```
i=1;
while(i<5){
       i=i+1;
}
```

Цикл с постусловием:

```
начало; do{инструкции}while(условие).
```

Например,

```
i=1;
do{
        i=i+1;
} while(i<5)
```

- *конструкция обработки исключений* (try/catch/finally). Блок try содержит инструкции, исключения которых обрабатываются блоком catch. По сути, данную конструкцию можно перевести как «попробуй сделать определенное действие, если не получилось, то сделай другое». После блока catch может находиться блок finally, содержащий код очистки, который будет выполнен независимо от того, что происходит в блоке try. Конструкция обработки исключений записывается следующим образом:

```
try{инструкции} catch(e){инструкции} finally{инструкции}
```

Стоит отметить, что данная конструкция очень часто используется для поиска ошибок. Так, если посмотреть на запись конструкции, то в круглых скобках блока catch указана переменная e, которая как раз и принимает значение исключения (ошибки), когда в блоке try не выполняются инструкции. Например, код

```
var S = 2019;
try{
        console.log(S);
}
catch(e){
        console.log(e);
}
```

выведет в консоль браузера значение переменной S. Если бы переменная S не была объявлена, то данный код выведет в консоль ошибку, содержащуюся в переменной e (эта ошибка – S is not defined, т. е. переменная S не объявлена, ее нет);

- *конструкции объявления функций*. Функции являются основными блоками программы. Функции объявляются при помощи ключевого слова function. Есть два способа объявления функций, и эти способы по-разному влияют на работу веб-сайта. Первый способ – объявление функции через переменную (или часто встречающееся в литературе function definition expression):

```
var имя_функции = function (параметры){инструкции}
```

Второй способ – просто использование ключевого слова (или часто встречающееся в литературе function declaration statement):

```
function имя_функции(параметры){инструкции}
```

Как видно из представленных конструкций, в круглых скобках указываются параметры (аргументы), которые сможет принимать данная функция, а в

фигурных – инструкции, которые составляют тело функции.

Вызов функции, т. е. выполнение ее кода, осуществляется следующим образом:

```
имя_функции(параметры или их значения).
```

Для возврата результата выполнения функции используется ключевое слово return. Директива return может находиться в любом месте тела функции. Как только выполнение доходит до этого места, функция останавливается и значение возвращается в вызвавший ее код.

Пример создания функции и ее вызова с определенными параметрами:

```
function myFunc(a, b){
        return a+b;
}
myFunc(2, 3);
```

После выполнения функция вернет значение, равное 5.

При использовании функций также стоит вспомнить про глобальные и локальные переменные и их области видимости. Так, локальные переменные (т. е. объявленные внутри функции) видны только внутри этой функции. Глобальные переменные (т. е. объявленные снаружи всех функций) видимы для любой функции, но только если их не перекрывают локальные переменные, имеющие такое же имя.

Еще одной особенностью языка JavaScript, которую необходимо учитывать при разработке, является асинхронность. Это значит, что код, написанный на JavaScript, не всегда выполняется последовательно, как, например, код, написанный на языке Python. При исполнении JavaScript-кода вызов команд осуществляется последовательно (сверху вниз), однако каждая следующая вызываемая команда не дожидается окончательного выполнения предыдущей. Проблемы асинхронности JavaScript можно решать при помощи упоминавшихся конструкций async/await и defer, а также конструкций Callback (записывает функции, которые вызываются по завершению асинхронного действия, применяется конструкция XMLHttpRequest) и Promise (позволяет выполнить определенный код только тогда, когда будет исполнен другой код, передающий свой результат первому). Кроме того, необходимо учитывать порядок инициализации (считывания) объектов в JavaScript. Так, при выполнении кода в браузере сначала инициализируются функции, объявленные при помощи обычного слова function, а только затем переменные. Например, функция function myFunc1(){} будет считана раньше, чем функция var myFunc2 = function (){}.

Написание программного кода на любом языке программирования следует сопровождать комментариями. Они необходимы разработчикам для более легкого и быстрого понимания текста кода (т. е. помогают им разобраться в том, что делает каждая функция, за что отвечает каждая переменная, и т. д.). Также комментарии можно иногда использовать и для отладки кода. Например, можно временно закомментировать участки кода, чтобы понять, какая строка или функция вызывает ошибку или не возвращает результат. Комментарии могут находиться в любом месте скрипта, и они

никак не влияют на его выполнение, т. к. полностью игнорируются интерпретатором.

В языке JavaScript для создания комментариев отвечают специальные символы. Так, для написания однострочного комментария используется двойная косая черта (//), а для создания многострочных комментариев – конструкция /* текст_комментария */.

При разработке клиентской части веб-приложений на JavaScript обычно работают с элементами HTML-документа, т. е. с DOM-деревом. При помощи JavaScript можно получить какой-либо элемент и проводить с ним различные манипуляции, например: добавить внутрь элемента новый элемент, удалить элемент, изменить его стиль, наполнить его содержимым и т. д. Также при помощи JavaScript можно отлавливать определенные действия, происходящие на веб-странице. Такие действия называются событиями. JavaScript – это язык событий. Примерами событий могут быть: наведение курсора на какой-либо элемент веб-страницы, клик по экрану, двойной клик и т. д.

Все события можно отследить и задать для них какие-либо функции. Это можно сделать с помощью упоминавшихся ранее обработчиков событий, например

```
<button onclick='alert("нажал кнопку")'>Кнопка</button>
```

или

```
<button onclick='myFunc()'>Кнопка</button>
<script>
var eFunc = function(){
        var x = 75;
        var y = 27;
        alert(x+y);
};
</script>
```

Другой способ задания функций на события представляет собой получение при помощи JavaScript-элементов DOM-дерева и указания им событий и функций на эти события.

Для того чтобы получить все DOM-дерево элементов веб-страницы, необходимо использовать команду document (например, можно ее ввести в консоли браузера при открытой веб-странице и нажать **Enter**). Также все дерево элементов веб-страницы можно вызвать, если использовать команду window. При этом window – это объект, в котором document – один из подобъектов, т. е. window шире, чем document. Однако чаще для работы с DOM-элементами используют объект document. Для получения списка функций, методов и свойств, которые поддерживает тот или иной объект, в консоли браузера или в среде разработки можно ввести имя объекта и точку (например, document.«свойства и методы»). Так, у объекта document в таком списке есть методы для получения элементов из DOM-дерева веб-страницы. Среди них наиболее часто используемыми методами получения элементов являются:

- по его идентификатору (атрибуту id) – getElementById();
- по его имени (атрибуту name) – getElementsByName();
- по названию элемента (тега) – getElementsByTagName();

- по названию класса (атрибуту class) – get ElementsByClassName();
- по селекторам CSS – querySelectorAll() и querySelector().

Например, код let x = document.getElementById('firstCell') позволяет получить из HTML-документа элемент с идентификатором 'firstCell' и присвоить его переменной x.

Получение атрибутов и событий HTML-элементов в JavaScript осуществляется оператором . Например, код изменения идентификатора элемента может выглядеть следующим образом:

```
let x = document.getElementById('firstCell');
x.id = 'newCell';
```

Также для получения атрибутов элементов можно использовать метод getAttribute(), а для их изменения – метод setAttribute(). Например, изменить стиль элемента из прошлого примера можно таким образом:

```
x.setAttribute('style', 'color:red')
```

По-другому этот пример можно записать так: x.style.color = 'red'.

Настройка событий осуществляется подобным образом. Например, код смены цвета фона элемента на зеленый при клике по нему можно записать так:

```
x.onclick = function(){
        x.setAttribute('style', 'background-color:green');
}
```

Также регистрацию событий на элементе можно реализовать при помощи метода addEventListener():

```
x.addEventListener(''click'', function(){
        x.setAttribute('style', 'background-color:green');
})
```

Важной особенностью при работе с DOM-деревом является то, что код JavaScript может начать выполняться раньше, чем в браузер загрузится дерево элементов HTML-страницы. Это еще одно проявление асинхронности. Чтобы избежать подобного поведения, можно отследить событие загрузки DOM-дерева и начать выполнять код только после этого. Данное событие записывается следующим образом:

```
document.addEventListener(«DOMContentLoaded», function(){});
```

Для удобства работы с деревом элементов HTML-страницы при помощи JavaScript можно использовать дополнительные библиотеки (представляют собой набор классов и функций), например: jQuery, Dojo и др. Они меняют в сторону упрощения базовый синтаксис JavaScript для обращения к элементам. Помимо перечисленных, существует и множество других JavaScript-библиотек, которые расширяют возможности разработки веб-приложений и используются для разных целей. Так, для разработки пользовательских интерфейсов часто применяют библиотеки React.js, Vue.js, Angular.js, Ember.js, Bootstrap,

для обработки массивов – Underscore.js, для построения диаграмм и графиков – Chart.js, Plotly.js и т. д.

4.3. Разработка клиентского веб-приложения с использованием интерфейса ArcGIS API for JavaScript

Интерфейс программирования приложений ArcGIS API for JavaScript представляет собой JavaScript-библиотеку, предоставляющую набор классов и функций, предназначенных для создания клиентских веб-приложений на основе опубликованных веб-сервисов. Данная библиотека наилучшим образом подходит для разработки веб-приложений с использованием пространственных данных, опубликованных при помощи программного обеспечения ArcGIS Server. Библиотека и документация по ее использованию доступны в сети Интернет на ресурсе для разработчиков (ArcGIS for Developers), расположенном по адресу https://developers.arcgis.com/javascript/ [17].

Существует две версии данной библиотеки (3.x и 4.x), которые регулярно обновляются. Эти версии отличаются друг от друга следующим: во-первых, 4-я версия имеет классы и модули, предназначенные для работы с трехмерными (3D) данными, во-вторых, в этих версиях названия классов, отвечающих за одну и ту же функцию, могут отличаться и, в-третьих, существуют некоторые отличия между ними в написании выражений.

В начале разработки клиентского картографического веб-приложения создается HTML-файл, в тело `<body>` которого помещаются первичные элементы необходимые для работы будущего приложения (например, блочные элементы `<div>` для помещения карты, легенды и т. п., кнопки `<button>` для активации некоторых функций и т. д.). Элементы можно добавлять по ходу разработки приложения. Также для созданных элементов необходимо настроить стили при помощи CSS.

Для начала работы с библиотекой ArcGIS API for JavaScript в рамках разработки картографического веб-приложения необходимо импортировать библиотеку и используемые в ней таблицы стилей CSS. Для этого в HTML-документе в заголовке страницы `<head>` указываются конструкции:

- для 3-й версии:

```
<link rel="stylesheet" href="https://js.arcgis.com/3.x/esri/css/esri.css">
<script src="https://js.arcgis.com/3.x/"></script>;
```

- для 4-й версии:

```
<link rel="stylesheet" href="https://js.arcgis.com/4.x/esri/css/main.css">
<script src="https://js.arcgis.com/4.x/"></script>;
```

где `x` – текущий выпуск библиотеки.

Затем в тегах `<script></script>` в заголовке веб-страницы `<head>` или в отдельном файле с расширением .js (который подключается к HTML-документу) пишется код приложения.

Скрипт, использующий библиотеку ArcGIS API for JavaScript, начинается с подключения используемых модулей (соответствуют классам) библиотеки

при помощи команды require (также можно использовать команду import) и создания функции, аргументами которой являются классы подключаемых модулей. Внутри тела созданной функции пишется основной код веб-приложения, который в том числе использует подключенные модули. Общая конструкция подключения модулей библиотеки ArcGIS API for JavaScript выглядит следующим образом:

```
require([«модуль_1», «модуль_2», «модуль_N»],
function(имя_класса_1, имя_класса_2, имя_класса_N){
        //код для функционирования веб-приложения
});
```

Перечень всех модулей и входящих в них классов можно посмотреть на сайте каждой из версий библиотек ArcGIS API for JavaScript на вкладке API Reference (рис. 4.7).

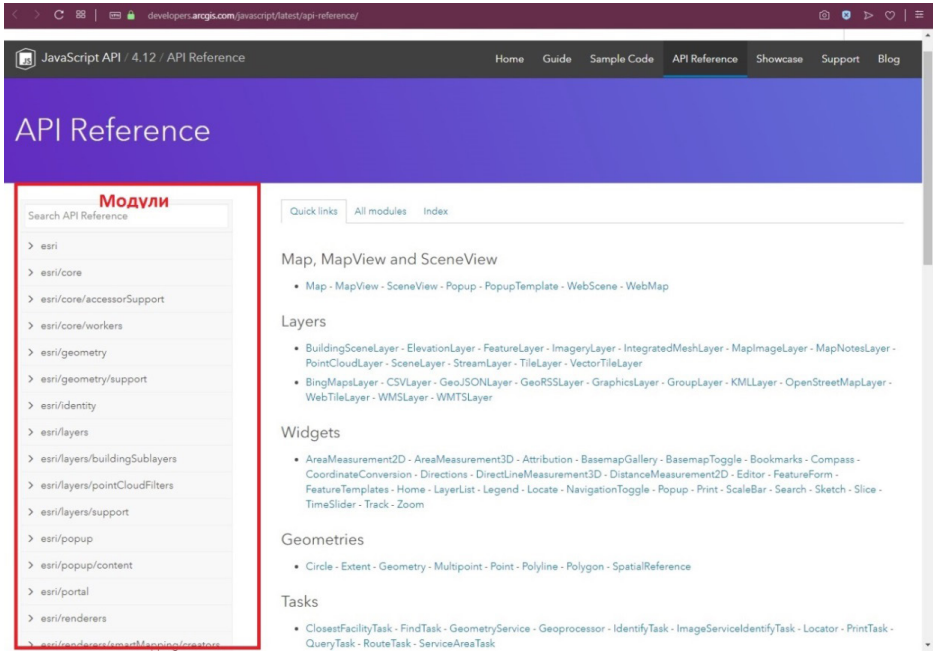

Рис. 4.7. Модули библиотеки ArcGIS API for JavaScript, версия 4.12

По каждому классу на сайте библиотеки есть подробная справка, включающая информацию о том, как подключить модуль, как выглядит конструкция написания класса, какие свойства (properties) и методы (methods) имеет каждый класс, а также какие события (events) можно отследить у объекта, созданного с использованием того или иного класса. Для свойств указаны и возможные значения, которые они могут принимать. Также внутри справки имеются ссылки на примеры (samples) использования этого класса в клиентских веб-приложениях.

Главным при создании картографического веб-приложения, является

класс, предназначенный для создания карты (class Map). В 4-й версии библиотеки ArcGIS API for JavaScript помимо класса Map необходимо также использовать класс вида карты: MapView – для плоских двумерных карт или SceneView – для трехмерных изображений. Пример создания карты веб-приложения в 4-й версии библиотеки:

```
require([«esri/Map», «esri/views/MapView»],
function(Map, MapView){
        var myMap = new Map({
                basemap: osm
        });
        var view = new MapView({
                container: «divMap»,
                map: myMap
        });
})
```

В данном коде создаются новые экземпляры классов (объекты) Map и MapView, которые присваиваются переменным myMap и view соответственно. В фигурных скобках каждого объекта указываются его свойства и значения этих свойств. Например, у объекта карты указано свойство basemap (базовая карта), которое имеет значение osm (соответствует картографическому веб-сервису OpenStreetMap). У вида карты в свойствах указаны блочный элемент HTML-страницы, в который необходимо поместить вид (свойство container), и объект карты, который необходимо добавить к виду (свойство map).

Для добавления картографических сервисов на карту в виде слоев используются классы, расположенные в группе модулей "esri/layers". Например, для добавления карты с перечнем слоев и оформлением, опубликованной на ArcGIS Server в виде картографического сервиса WMS, можно использовать класс MapImageLayer (в 3-й версии библиотеки – ArcGISDynamicMapServiceLayer), а для добавления отдельных векторных слоев опубликованной карты – класс FeatureLayer (слой доступен по стандарту WFS). Добавление картографических сервисов WMS, опубликованных, например, на GeoServer (см. разд. 3.2), используется класс WMSLayer, сервисов WFS – WFSLayer и т. д. При реализации возможности добавления на веб-карту пользователем собственных векторных объектов используется класс GraphicLayer.

Например, в представленном коде (использована 4-я версия библиотеки):

```
require(["esri/Map", "esri/views/MapView", "esri/layers/
MapImageLayer"],
function(Map, MapView, MapImageLayer){
        var myMap = new Map({
                basemap: satellite
        });
        var view = new MapView({
                container: "divMap",
                map: myMap
```

```
            });
            var layer1 = new MapImageLayer({
                  url:"http://maps.psu.ru:8080/arcgis/rest/
services/stud_GIS_4kurs/OSM_Perm/MapServer ",
                  opacity: 0.5
            });
            myMap.add(layer1);
      })
```

– к объекту Map в виде отдельного слоя добавляется карта территории г. Перми при помощи класса MapImageLayer. Указанная карта г. Перми создана на основе векторных данных OpenStreetMap и включает 53 слоя. Она опубликована в виде веб-сервиса на ArcGIS Server, установленном на сервере maps.psu.ru, и доступна по ссылке в его директории сервисов Services Directory. Ссылка на сервис указывается в свойстве url объекта MapImageLayer. При помощи метода add() класса Map сервис в виде слоя добавляется на карту myMap поверх ее базовой карты (значение satellite свойства basemap – космические снимки ArcGIS Imagery).

Для отображения на карте векторных объектов какого-либо слоя из опубликованного на ArcGIS Server картографического сервиса используется класс FeatureLayer. Добавление с помощью данного класса векторного слоя населенных пунктов из картографического сервиса, использованного в предыдущем примере, выглядит следующим образом:

```
      require(["esri/Map", "esri/views/MapView", "esri/layers/
FeatureLayer"],
      function(Map, MapView, FeatureLayer){
            var myMap = new Map({
                  basemap: satellite
            });
            var view = new MapView({
                  container: "divMap",
                  map: myMap
            });
            var layer2 = new FeatureLayer({
                  url:"http://maps.psu.ru:8080/arcgis/rest/
services/stud_GIS_4kurs/OSM_Perm/MapServer/39",
                  opacity: 0.5,
                  title: "Населенные пункты"
            });
            myMap.add(layer2);
      })
```

Как видно из примера, в ссылке свойства url указывается порядковый номер слоя, добавляемого из картографического сервиса. Используемый слой населенных пунктов имеет на ArcGIS Server в Services Directory порядковый номер 39.

Для реализации возможностей рисования векторных объектов на карте используется класс GraphicLayer (графический слой) и классы для добавления (рисования) объектов в этот слой, расположенные в 4-й версии библио-

теки в группе модулей "esri/views/draw", а в 3-й версии – в "esri/toolbars".

У слоев, добавленных при помощи класса MapImageLayer, изменять стиль оформления можно только в ArcGIS Desktop с дальнейшей перезаписью картографического сервиса. А у слоев, созданных при помощи классов FeatureLayer или GraphicLayer, доступна возможность изменения стиля оформления. Для этого используются классы, расположенные в группе модулей "esri/symbols", если задаются единые стили для всех объектов, и в группе "esri/renderers", если необходимо использовать какие-либо способы картографического отображения, например количественный фон или др.

Для настройки идентификации объектов слоев, а также создания запросов к их атрибутивным таблицам обычно используют классы IdentifyTask и QueryTask, расположенные в группах "esri/tasks".

К часто используемым классам библиотеки ArcGIS API for JavaScript стоит отнести те, которые позволяют создать легенду (Legend), масштабную линейку (ScaleBar), кнопку домашнего экстента карты (Home и HomeButton), панель управления слоями карты (LayerList), инструменты измерения площадей, расстояний, координат (Measurement), всплывающие окна (Popup), галерею базовых карт (BasemapGallery) и др. Перечисленные классы доступны в группе модулей "esri/widgets" в 4-й версии и в группе "esri/dijit" 3-й версии библиотеки.

Стоит также отметить, что при разработке веб-приложений с помощью ArcGIS API for JavaScript можно использовать готовые примеры, которые приведены на сайте библиотеки. Эти примеры доступны как в описании каждого модуля (класса), так и отдельно на вкладке Sample Code.

4.4. Разработка клиентского веб-приложения с использованием интерфейсов OpenLayers и Leaflet

Описанный в предыдущем разделе интерфейс ArcGIS API for JavaScript представляет собой лишь один из вариантов, обеспечивающих создание веб-приложений (веб-карт с возможностями интерактивного управления), исполняемых в любой операционной системе благодаря кроссплатформенности языка JavaScript. Из наиболее известных и популярных среди разработчиков – интерфейсы OpenLayers [29] и Leaflet [25], являющиеся библиотеками с открытым исходным кодом (open source), написанными на языке JavaScript. Среди коммерческих продуктов весьма популярно решение Google Maps API, реализуемое в рамках Google Maps Platform [23] и предлагающее разные варианты платного использования. Принципы организации и вид исходного кода веб-приложений на основе упомянутых здесь программных интерфейсов схожи, а отличия проявляются в разной функциональности и удобстве использования.

Ниже будут рассмотрены простейшие примеры использования двух библиотек (OpenLayers и Leaflet), развиваемых в рамках проектов с открытыми исходными кодами. Первый проект имеет более длительную историю развития, несколько большую функциональность и более значительные объемы кода библиотек. Второй проект более компактен и прост в изучении, в целом имеет меньше функций и настроек, обеспечивая простоту

разработки базовых элементов картографического интерфейса и, соответственно, меньшие объемы кода. Библиотека Leaflet лежит в основе открытой библиотеки Mapbox GL JS, используемой сервисами картографической платформы Mapbox, которые развиваются компанией с одноименным названием [26]. Платформа позволяет разрабатывать и обеспечивать онлайн-функционирование мобильных и картографических веб-приложений на основе обновляемых пространственных данных.

В обоих вариантах можно вообще не копировать библиотеки, ограничиваясь использованием гиперссылок на текущие версии библиотек, размещенных в онлайн-репозиториях. Тем не менее для повышения эффективности разработки и эксплуатации рекомендуется использовать среду исполнения приложений на JavaScript (например, Node.js) и сопутствующие инструменты.

4.4.1. Библиотека Leaflet

Для того чтобы обеспечить отображение карты через веб-интерфейс (окно стандартного браузера), в какой-либо элемент HTML-страницы (как правило, это раздел или блок, выделяемый тегом) встраивается фрагмент кода с описанием объекта Map и сопутствующих элементов (например, тайловый объект, сгенерированный соответствующим картографическим сервисом и используемый в качестве подложки). Код включает также параметры начального отображения: координаты центра, масштаб, размер окна и т. п.

Ниже приводятся типовые фрагменты, задающие отображение в окне браузера различных растровых покрытий (подложек), формируемых из данных соответствующих тайловых сервисов. Так, размер отображаемой в окне браузере карты описывается вставкой в описание стиля кода

```
        html, body, #map {
          height: 800px;
          width: 100%;
        }
```

(высота окна в пикселах, ширина – все, что доступно).

Географические границы максимальной области просмотра задаются координатами нижнего левого и верхнего правого углов прямоугольника:

```
        var bounds = L.latLngBounds(L.latLng(57, 55), L.latLng(59,
57));
        var map = new L.Map('map', {
          center: new L.LatLng(58.00,56.00),
      zoom: 11,
          minZoom: 9,
          maxZoom: 17,
          maxBounds: bounds
        });
```

Здесь же указываются координаты центра и масштаб увеличения, которые используются при открытии окна и при его обновлении. Границы области, минимальный и максимальный масштабы определяют предельно допустимые диапазоны значений.

Далее представлены два варианта отображения окрестностей г. Перми, использующие тайловые сервисы от Google: спутниковое покрытие и карта рельефа; оба слоя используются в качестве подложек по выбору пользователя. Соответствующие фрагменты кода:

```
        var baseMaps = {
            «Google.Карта»:  L.tileLayer('http://{s}.google.com/vt/
lyrs=p&x={x}&y={y}&z={z}',{
              subdomains:['mt0','mt1','mt2','mt3']
            }).addTo(map),
            "Google.Спутник":  L.tileLayer('http://{s}.google.com/vt/
lyrs=s&x={x}&y={y}&z={z}',{
              subdomains:['mt0','mt1','mt2','mt3']
            })
        };
```

(слои, соответствующие двум альтернативным вариантам подложек).

Поверх выбранной подложки по командам пользователя могут накладываться другие слои (один, несколько или ни одного). В следующем примере показано два слоя: картографическое покрытие (групповой слой) и отдельный слой, настроенные и опубликованные на локальном сервере для отображения в виде тайлов, передаваемых по протоколу WMS:

```
        var overlayMaps = {
           «Адм. границы»: L.tileLayer.wms(«http://localhost:8080/
geoserver/ows?», {
              layers: 'common:pk_osm_adm6_district_pol',
              format: 'image/png',
              transparent: true,
              styles: 'ng_boundary_pol'
          }).addTo(map),

          «Карта OSM»: L.tileLayer.wms(«http://localhost:8080/geos-
erver/ows?», {
              layers: 'common:hybrid_grp',
              format: 'image/png',
              transparent: true
          })
        };
```

(оба слоя могут накладываться поверх подложки).

Для отображения последнего слоя дополнительно указывается стиль, отличный от используемого по умолчанию. Отображение слоев завершается командой[13]

13 Данная команда представляет собой одну строку кода, для удобства восприятия разделенную на две строки с помощью обратного слэша.

```
        L.control.layers(baseMaps, overlayMaps, \
          {collapsed: false}).addTo(map);
```

добавляющей на карту интерактивный элемент – меню управления слоями.

4.4.2. Библиотека OpenLayers

Структура HTML-страницы аналогична описанной выше. Размеры карты здесь также описываются добавлением в описание стиля строк

```
        .map {
            height: 800px;
            width: 100%;
        }
```

(параметры аналогичны тем, которые были заданы в примере для Leaflet).

Далее представлен фрагмент кода, обеспечивающего отображение окрестностей г. Перми в виде подложки. В качестве источника используется тайловый сервис с данными от проекта OpenStreetMap [30]:

```
        var bounds = [55.0, 57.0, 57.0, 59.0];
        var map = new ol.Map({
          target: 'map',
          layers: [
            new ol.layer.Tile({
              source: new ol.source.OSM()
            })
          ],
          view: new ol.View({
            center: ol.proj.fromLonLat([56.00, 58.00]),
            zoom: 11,
            minZoom: 9,
            maxZoom: 17,
             extent: ol.proj.transformExtent(bounds, 'EPSG:4326',
'EPSG:3857')
          })
```

Экстент и масштабы соответствуют примеру из предыдущего раздела. Формат команды для создания подложки несколько отличается от предыдущего примера (слой "Google.Карта" наиболее близок к нему по содержанию и отличается отдельными настройками и – в меньшей степени – данными). Мы видим, что объем кода для создания подложки здесь несколько больше. Аналогичным образом могут быть настроены слои, накладываемые поверх подложки.

Среди разработчиков в целом превалирует мнение, что библиотека OpenLayers вследствие длительного периода развития является более

стабильной и функциональной и требует взамен большего количества времени на ее освоение. В свою очередь, сторонники Leaflet делают акцент на значительно большей простоте в решении типовых задач и – за счет активного развития и появления новых плагинов – на возможности реализовать аналогичную функциональность. Решение вопроса о лучшем выборе остается за разработчиком.

Как обычно, наиболее доступным и быстрым способом создания веб-приложений оказывается использование готовых примеров и при необходимости обращение к документации. Все это размещается на сайтах соответствующих библиотек [25, 29].

ЗАКЛЮЧЕНИЕ

Веб-картографирование – быстро развивающаяся отрасль, и некоторые ранее актуальные технические моменты создания картографических веб-приложений становятся устаревшими. Поэтому авторы данного пособия постарались описать основные концептуальные принципы и подходы к проектированию и созданию картографических веб-сервисов, которые останутся актуальными в ближайшие годы. Они преимущественно описаны в гл. 1–2. Также для лучшего понимания материала в гл. 3 и 4 учебного пособия были включены некоторые актуальные на момент написания пособия технические детали. Они касаются основных возможностей публикации картографических веб-сервисов, работы с конкретным программным обеспечением (ArcGIS Server, GeoServer) и облачными ресурсами (ArcGIS Online и NextGIS.com), а также разработки клиентских картографических веб-приложений с использованием API: ArcGIS API for JavaScript, Leaflet и OpenLayers. Указанные программы и ресурсы на протяжении нескольких лет остаются доминирующими в области создания профессиональных веб-ГИС как в России, так и за рубежом.

Кроме того, стоит отметить, что на данный момент требования к ГИС-специалистам, выдвигаемые потенциальными работодателями, достаточно часто включают в себя знания, умения и навыки работы с веб-ГИС-технологиями. Таким образом, изложенный в пособии материал будет полезен студентам для получения компетенций, ожидаемых от современного ГИС-специалиста.

Авторы пособия надеются, что оно вызовет интерес не только у студентов, обучающихся по направлению подготовки «Картография и геоинформатика» и по программе магистратуры «Математико-картографическое моделирование геосистем и комплексов», но и у всех обучающихся географического факультета, которые интересуются возможностями применения ГИС-технологий при решении научных и практических задач.

ЛИТЕРАТУРА

1. Геоинформатика. Толковый словарь основных терминов. М.: ГИС-ассоциация, 1999. 204 с.
2. Геоинформатика: учебник для студентов вузов / Е. Г. Капралов, А. В. Кошкарев, В. С. Тикунов и др.; под ред. В. С. Тикунова. М: Издательский центр «Академия», 2005. 480 с.
3. Документ RFC 3986, стандарт URI. URL: https://tools.ietf.org/html/rfc3986 (дата обращения: 01.06.2020).
4. Документация компании ESRI – платформа ArcGIS Enterprise. URL: https://enterprise.arcgis.com/ru/ (дата обращения: 12.06.2020).
5. Документация по продукту QGIS Server. URL: https://docs.qgis.org/3.10/en/docs/user_manual/working_with_ogc/ogc_server_support.html (дата обращения: 16.06.2020).
6. Дубинин М. Ю., Костикова А. М. Веб-ГИС. Компьютерра. 2008. № 749.
7. Компания EstiMap (дистрибьютер программных продуктов MapInfo). URL: http://www.esti-map.ru (дата обращения: 10.06.2020).
8. Костикова А. М. Классификация картографических веб-сервисов. URL: http://gis-lab.info/qa/ogc-intro.html (дата обращения: 16.06.2020).
9. Лурье И. К. Геоинформационное картографирование Методы геоинформатики и цифровой обработки космических снимков: учебник. 2-е изд., испр. М.: КДУ, 2010. 424 с.
10. Олифер В. Г., Олифер Н. А., Компьютерные сети. Принципы, технологии, протоколы: учебник для вузов. 4-е изд. СПб.: Питер, 2010. 943 с.
11. Основы геоинформатики: в 2 кн. / Е. Г. Капралов, А. В. Кошкарев, В. С. Тикунов и др.; под ред. В. С. Тикунова. М.: Издательский центр «Академия», 2004. Кн. 1. 352 с.
12. Пиньде Фу, Цзюлинь Сунь. Веб-ГИС. Принципы и применение. Редлендз (Калифорния): Esri Press, 2010. 356 с.
13. Руководство пользователя ArcGIS Online. URL: https://www.esri.com/ru-ru/arcgis/products/arcgis-online/resources (дата обращения: 15.06.2020).
14. Руководство пользователя NextGIS.com. URL: https://docs.nextgis.ru/docs_ngcom/source/toc.html (дата обращения: 10.06.2020).
15. Самоучитель HTML и CSS htmlbook.ru. URL: http://htmlbook.ru (дата обращения: 01.06.2020).

16. Современный учебник JavaScript. URL: https://learn.javascript.ru (дата обращения: 01.06.2020).

17. ArcGIS API for JavaScript v4.x. URL: https://developers.arcgis.com/javascript/ (дата обращения: 10.06.2020).

18. Colin Henderson. Mastering GeoServer. Packt Publishing, 2014. 420 p.

19. EPSG Geodetic Parameter Dataset. URL: http://www.epsg-registry.org (дата обращения: 16.06.2020).

20. Gaia Geospatial Platform. URL: http://thecarbonproject.azurewebsites.net/Products/Gaia (дата обращения: 01.06.2020).

21. GDAL library documentation. URL: https://gdal.org (дата обращения: 16.06.2020).

22. GeoServer: open source server for sharing geospatial data. URL: http://geoserver.org (дата обращения: 16.06.2020).

23. Google Maps Platform: URL: https://cloud.google.com/maps-platform/ (дата обращения: 16.06.2020).

24. HTML5book: сайт для изучения веб-технологий. URL: https://html5book.ru. (дата обращения: 10.06.2020).

25. Leaflet: an open-source JavaScript library for mobile-friendly interactive maps. URL: https://leafletjs.com (дата обращения: 16.06.2020).

26. Mapbox: maps and location for developers. URL: https://www.mapbox.com/ (дата обращения: 16.06.2020).

27. MapServer: open source web mapping. URL: https://www.mapserver.org (дата обращения: 16.06.2020).

28. OpenGeo Suite User Manual. URL: http://93.187.166.52:8081/opengeo-docs/ (дата обращения: 16.06.2020).

29. OpenLayers. URL: https://openlayers.org (дата обращения: 16.06.2020).

30. OpenStreetMap. URL: https://www.openstreetmap.org (дата обращения: 01.06.2020).

31. PostGIS: Spatial and Geographic objects for PostgreSQL. URL: http://postgis.net (дата обращения: 16.06.2020).

32. PostgreSQL: Open Source Relational Database. URL: https://www.postgresql.org (дата обращения: 16.06.2020).

33. QGIS. Свободная географическая информационная система с открытым кодом. URL: https://qgis.org (дата обращения: 16.06.2020).

34. SASGIS. Веб-картография и навигация. URL: http://www.sasgis.org (дата обращения: 01.06.2020).

35. Tim O'Reilly. What Is Web 2.0. Design Patterns and Business Models for the Next Generation of Software. URL: https://www.oreilly.com/pub/a/web2/archive /what-is-web-20.html (дата обращения: 01.06.2020).

Учебное издание

Абдуллин Ринат Камилевич
Пономарчук Алексей Иванович

ТЕХНОЛОГИИ ИНТЕРНЕТ-КАРТОГРАФИРОВАНИЯ

учебное пособие

Редактор **В. А. Филимонова**
Корректор **А. В. Смирнова**
Компьютерная верстка **А. Н. Ташкинова**

Подписано в печать 11.09.2020. Формат 60×84/16.

Усл. печ. л. 7,67. Тираж 100 экз. Заказ 222

Издательский центр Пермского государственного
национального исследовательского университета
614990, г. Пермь, ул. Букирева, 15