$R(3) = t_n(3) - t_p(3) = 15 - 11 = 4$; $R(4) = t_n(4) - t_p(4) = 19 - 19 = 0$;

$R(5) = t_n(5) - t_p(5) = 30 - 28 = 2$; $R(6) = t_n(6) - t_p(6) = 24 - 24 = 0$;

$R(7) = t_n(7) - t_p(7) = 38 - 38 = 0$.

Критический путь проходит по событиям 1-2-4-6-8, его продолжительность равна 38. События 1, 2, 4, 6, 8 являются критическими и имеют нулевые резервы времени.

1. Галюк, А.Д. Управление проектами: курс лекций. – Екатеринбург: Изд-во УрГУПС, 2014. – 107 с.
2. Красс, М.С. Математика в экономике: математические методы и модели. Учебник для бакалавров / М.С. Красс, Б.П. Чупрынов; под ред. М.С. Красса. – Изд. 2-е, испр. и доп. – М.: Юрайт, 2017. – 541 с.

УДК 004.04

**А.В. Понизов, М.А. Серов, Т.А. Галаган**

## РАЗРАБОТКА ВЕБ-ПРИЛОЖЕНИЯ ДЛЯ ОБРАБОТКИ ГНСС-ДАННЫХ С ИСПОЛЬЗОВАНИЕМ МИКРОСЕРВИСНОЙ АРХИТЕКТУРЫ

***В работе рассматриваются этапы проектирования и особенности реализации web-приложения микросервисной архитектуры, которое предназначено для обработки ГНСС-данных, используемых для анализа геодинамической активности.***

***Ключевые слова: обработка данных, ГНСС-данные, распределенные вычисления, LXC-контейнеры, C#.***

## DEVELOPMENT OF WEB-APPLICATION FOR PROCESSING GNSS-DATA BASED ON MICROSERVICE ARCHITECTURE

***The paper discusses the design and implementation features web application of a micro-service architecture that is intended for processing GNSS-data for the analysis of geodynamic activity.***

***Key words: data processing, GNSS-data, distributed calculations, LXC-containers, C #.***

**DOI: 10/22250/jasu.7**

### Введение

Одним из важных аспектов современной геологии является анализ активности различных тектонических структур. Основным подходом для изучения смещений земной поверхности на Дальнем Востоке России является космическая геодезия с использованием глобальных навигационных спутниковых систем (ГНСС) [1, 2, 4]. Рассматриваемое веб-приложение предназначено для того, чтобы автоматизировать и упростить процесс, используемый в рамках метода, увеличить количество обрабатываемой информации и предоставить удобный интерфейс для запуска и параметрической настройки процесса.

Существуют отдельные программные продукты для решения конкретных задач в рамках ГНСС-метода. Однако именно отсутствие комплексных продуктов затрудняет работу специалистов,

поскольку различные программные продукты написаны на разных языках, имеют разные интерфейсы и форматы данных, а также поддерживаются различными коллективами разработчиков. Зачастую это приводит к невозможности использовать различные продукты для сложной обработки данных и/или применять другой программный продукт для дальнейшей обработки результатов, полученных в другом продукте.

Основными компонентами веб-приложения являются: микросервисы, выполняющие процедуры сбора, систематизации и первичной обработки геодезических данных; микросервисы математической обработки данных, включающие: приведение к условиям однородной среды, нормирование, осреднение в квадратах, расчет показателей в равнозначных градациях (баллах), расчет суммарного балла, построение картосхем в изолиниях; графический веб-интерфейс для формирования задач обработки информации и непосредственной настройки процесса обработки.

Преимущество микросервисной архитектуры заключается в возможности обеспечения совместной работы слабосвязанных и легко изменяемых модулей – микросервисов [3]. Для обеспечения стабильной и надежной передачи данных между микросервисами используются средства веб-фреймворка ASP .NET Core компании Microsoft. Он базируется на платформе .NET Core 3.1, открытой и кроссплатформенной, с длительным циклом поддержки и подробной документацией.

Для обеспечения плавной работы интерфейса и отсутствия перезагрузок при переходе между частями приложения используется фреймворк Angular. Он позволяет создавать «одностраничные приложения» (single page application), а также обладает большим набором готовых элементов интерфейса (Angular Material). К его преимуществам можно отнести и компонентную базу фреймворка, развиваемую авторами, что гарантирует стабильную работу при последующих обновлениях.

Основой для запуска микросервисов служит технология LXC (Linux Extended Container), удобный доступ к которой поставляется вместе с платформой виртуализации Proxmox VE.

**Особенности разработки веб-приложения на основе микросервисной архитектуры**

Архитектура приложения состоит из нескольких микросервисов, общающихся между собой по протоколу HTTP. Взаимодействие компонентов показано на рис. 1.

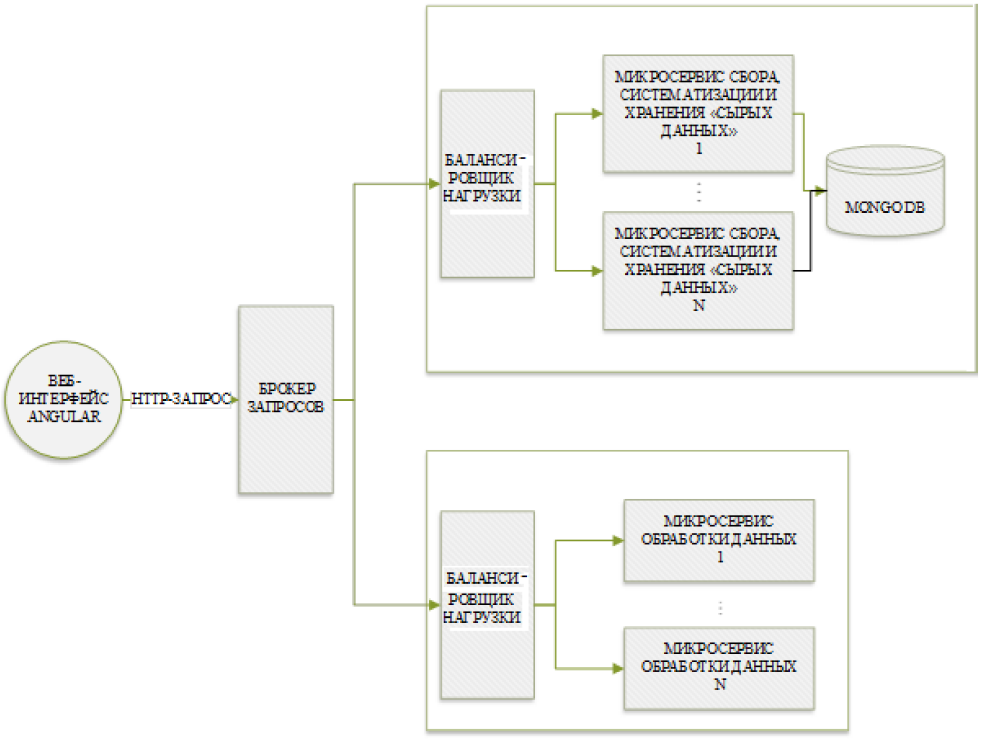


*Рис. 1.* Взаимодействие компонентов web-приложения.

Как видно из рис. 1, в системе, помимо микросервисов и веб-интерфейса, имеются брокер запросов и балансировщики нагрузки.

Брокер запросов представляет собой ASP .NET Core приложение, которое принимает HTTP-запросы от веб-интерфейса переадресует их нужной (в зависимости от запроса) группе потребителей (микросервисов), где их принимает балансировщик нагрузки. Схема работы брокера запросов представлена на рис. 2.

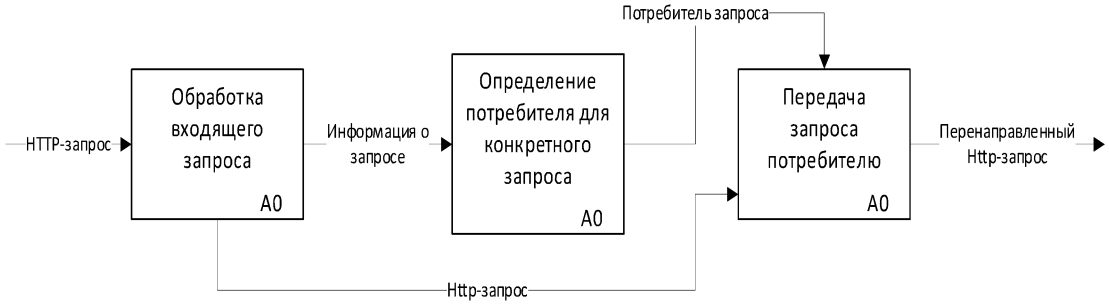


*Рис. 2.* Схема работы брокера запросов.

В качестве балансировщика нагрузки используется веб-сервер Apache 2 с соответствующими настройками. Это позволяет ему на основе информации о загруженности микросервисов равномерно распределять между ними задачи.

Основой каждого экземпляра микросервиса является LXC-технология виртуализации на уровне операционной системы, которая не использует виртуальные машины, а создает виртуальное окружение с собственным пространством процессов и сетевым стеком. К ее преимуществам относят: безопасность, простоту создания и управления, а также низкие ресурсные издержки на обеспечение виртуализации (в отличие от полноценных виртуальных машин). Схема контейнеризации сервисов показана на рис. 3.

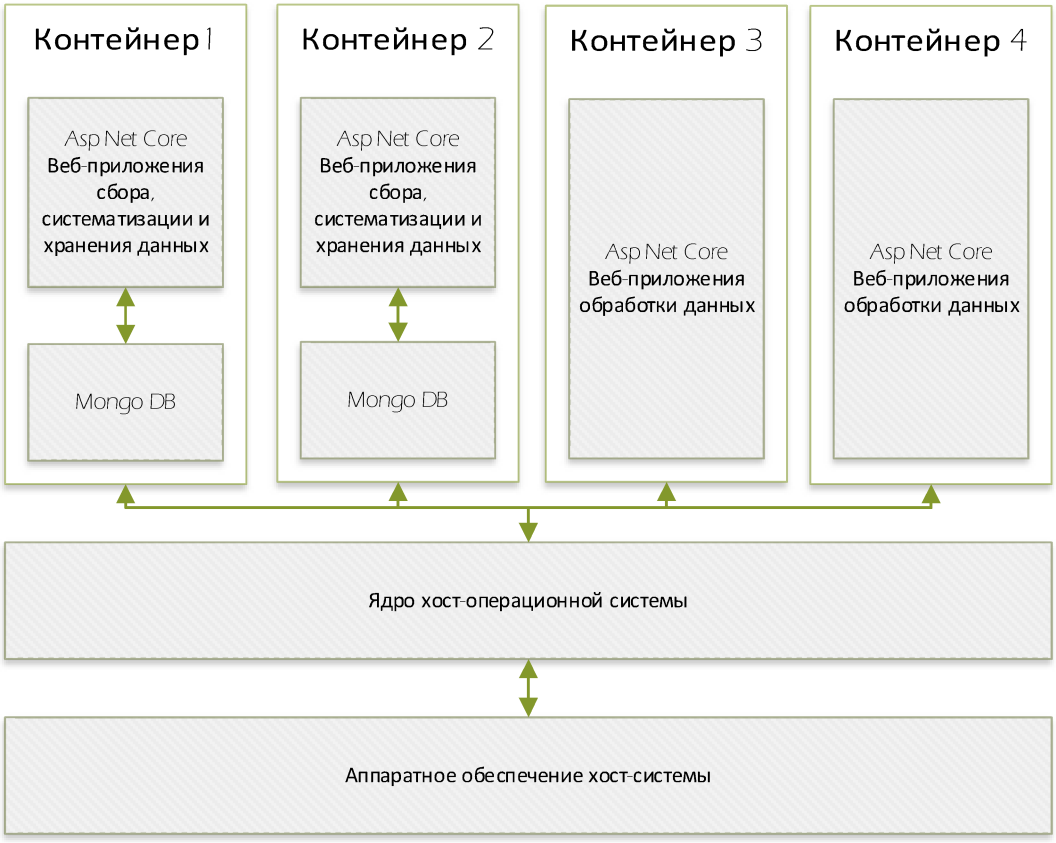


*Рис. 3.* Схема контейнеризации сервисов.

Из рис. 3 видно, что контейнеры в отличие от виртуальных машин включают в себя только приложение и его компоненты. Это позволяет эффективнее использовать оперативную память, ресурсы процессора и место на жестком диске.

Микросервисы сбора представлены в виде ASP .NET Core-приложения и выполняют функции систематизации и хранения информации, занимаются загрузкой из различных источников данных, необходимых для проведения расчетов, а также их первичной обработкой (конвертирование в открытый формат Rinex). Для хранения данных используется Mongo DB. Причины использования NoSQL базы данных заключены в неструктурированности исходной информации. В свою очередь, Mongo DB не вносит ограничений на типы данных, имеет хорошую горизонтальную масштабируемость. База данных содержит коллекции, которые включают данные с отдельных ГНСС-станций.

Микросервисы обработки данных представлены в виде ASP .NET Core-приложения и реализуют функционал математической обработки исходных данных.

Для работы с приложением пользователь должен лишь осуществить выбор ГНСС-станции и диапазон времени, за который необходимо выполнить вычисления, в веб-интерфейсе. Дальнейшая работа предполагает прохождение пользователем трех этапов. Экранная форма формирования задачи представлена на рис. 4. После завершения всех этапов кнопка «Начать» инициирует начало выполнения задачи.

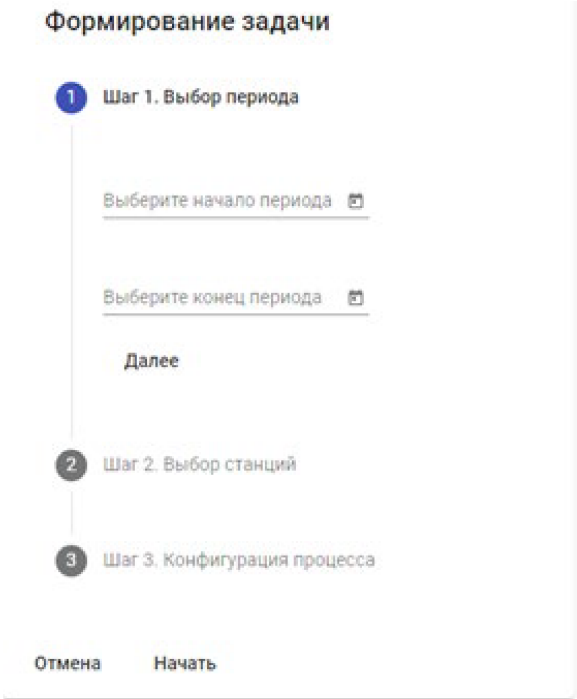


*Рис. 4.* Экранная форма формирования задачи.

Дальнейшая работа происходит автоматически. Веб-интерфейс отправляет эти данные брокеру запросов, который выполняет следующие задачи:

1) просматривает пришедший запрос, определяет исполнителей и переадресует им дальнейший инструкции в виде HTTP-запроса;

2) получает ответ от микросервисов обработки, систематизации, хранения данных в виде файлов исходных данных для следующего этапа;

3) сообщает веб-интерфейсу о завершении очередного этапа и переходе к следующему;

4) формирует запрос к микросервисам математической обработки данных, вкладывает в него все необходимые данные и переходит в режим ожидания ответа;

5) после получения ответа передает результаты в веб-интерфейс.

Итоговый результат доступен пользователю через веб-интерфейс.

**Заключение**

Разработанные с использованием платформы .Net Core 3.1 веб-компоненты приложения являются кроссплатформенными и обладают высокой стабильностью. Кроссплатформенность – гарантия длительной эксплуатации компонент, а в случае необходимости – замены технологии контейнеризации.

Разработанный микросервис сбора, систематизации и хранения информации дал возможность автоматизировать процесс загрузки данных из множества отдельных источников, обеспечил их первичную систематизацию и централизованное хранение в NoSQL базе данных.

Запуск нескольких экземпляров микросервиса обработки данных дал возможность повысить скорость и объемы обработки информации.

Использование микросервисной архитектуры гарантирует горизонтальную масштабируемость приложения, что позволяет наращивать вычислительные мощности за счет запуска новых экземпляров существующих микросервисов, без обновления аппаратной составляющей центра обработки данных. Всё, что необходимо конечному специалисту, – это выполнить настройку процесса, запустить его и просмотреть результаты.

В рамках клиентской части был реализован интерфейс конфигурирования задачи обработки, который предлагает пользователю пошаговую настройку задачи обработки данных, при этом на каждом этапе проверяет корректность вводимых данных.

1. Жижерин, В.С., Серов, М.А., Холобуда, С.П. Моделирование геодинамических процессов Верхнего Приамурья на основе GPS данных // Успехи современного естествознания. – 2018. – № 11. – С. 103-108.

2. Ханчук, А.И., Коновалов, А.В., Сорокин, А.А. [и др.]. Инструментальное и информационно-технологическое обеспечение сейсмологических наблюдений на Дальнем Востоке России // Вестник ДВО РАН.– 2011. – № 3. – С. 127-137.

4. Ричардсон, Крис. Микросервисы. Паттерны разработки и рефакторинга. – СПб.: Питер, 2019. – 544 с.

5. Быков, В. Г., Бормотов, В.А., Коковкин, А.А [и др.]. Начало формирования единой сети геодинамических наблюдений // Вестник ДВО РАН. – 2009. – № 4. – С. 83-93.

6. Чамберс, Д. ASP .NET Core. Разработка приложений / Д. Чамберс, Д. Пэкетт, С. Тиммс. – СПб.: Питер, 2018. – 464 с.