Реализация панели администрирования для платформы многоагентного моделирования

А.А. Бурова, С.С. Буров, Д.С. Парыгин

Волгоградский государственный технический университет

attapi343@gmail.com, sergey.burovic@gmail.com, dparygin@gmail.com

Аннотация

В данной статье рассмотрены проектирование и реализация web клиента, выполняющего роль панели администрирования для платформы многоагентного моделирования перемещений и взаимодействий акторов в рамках участка карты города. Вся логика моделирования в данной платформе реализуется непосредственно в модулях, тогда как программная платформа осуществляет лишь её вызов для конкретных, подключенных модулей.
Программная платформа реализована на фреймворке ASP.NET Core версии 5.0.
Для реализации веб клиента был выбран фреймворк Angular версии 11 с использованием компонентов пользовательского интерфейса Ant Design.

Ключевые слова: симуляция, город, C#, модульность, клиент-сервер, панель администрирования

Библиографическая ссылка: Бурова А.А., Буров С.С., Парыгин Д.С. Реализация панели администрирования для платформы многоагентного моделирования // Информационное общество: образование, наука, культура и технологии будущего. Выпуск 5 (Труды XXIV Международной объединенной научной конференции «Интернет и современное общество», IMS-2021, Санкт-Петербург, 24 – 26 июня 2021 г. Сборник научных статей). — СПб.: Университет ИТМО, 2021. С. 19-27. DOI: 10.17586/2587-8557-2021-5-19-27

1. Введение

Существует множество платформ, позволяющих проводить моделирование с графическим пользовательским интерфейсом. Рассмотрим три платформы в таблице 1: Ant Road Planner [1], NetLogo [2] и AnyLogic [3].

Таблица 1. Обзор платформ моделирования с панелью администрирования

Название платформы	Редактор базовой карты	Динамическое изменение свойств моделирования	Модульность	Отображение результатов моделирования	Динамическое изменение диапазона моделирования
Ant Road Planner	+	-	-	+	-
NetLogo	+	-	-	+	+
AnyLogic	+	-	-	+	-

Графический пользовательский интерфейс в таких платформах зачастую имеет ряд ограничений и работает неотрывно от самой платформы, становясь из-за этого ее непосредственной частью и накладывая ряд ограничений на эксплуатацию самой платформы.

Целевым компонентом данной работы является приложение, представляющее собой клиент-серверную платформу для моделирования перемещений и взаимодействий акторов в рамках участка карты города [4, 5]. Построение модели ведется на основе многоагентного подхода [6, 7]. Программа осуществляет расширение функционала моделирования за счет подключаемых к ней модулей (инициализирует их) и отображает функционал на графической карте города. Разработка различных модулей ведется по сей день, и система постоянно расширяется.

Модули содержат в себе всю логику и правила моделирования. Изначально модули не поставляются в составе данного программного комплекса, каждый из них является отдельной обособленной библиотекой классов. Однако модули принимают непосредственное участие в работе программного комплекса в случае подключения одного или нескольких из них. К примеру, модуль управления данными об объектах на онлайн-карте города [8] производит парсинг данных из файла с форматом OSM XML и преобразует их к необходимой структуре для хранения в списке объектов основной программы и дальнейшего использования их другими модулями.

Система администрирования требуется для исполнения различных сценариев моделирования в реальном времени с расширяемым свойствам списком свойств самой модели и акторов и подключаемыми компонентами для получения ответа на множественный вопрос "что, если".

В связи с этим, целью работы является разработка системы распределенного администрирования поведения акторов и свойств модели города в реальном времени. При этом в качестве ключевого компонента концепции создания такой системы было решено сфокусироваться на предоставлении возможности пользователю влиять на ход моделирования путем изменения свойств моделей и акторов в реальном времени.

2. Постановка задачи

Существующее приложение моделирует перемещения акторов на основе модулей, однако имеет ряд недостатков и ограничений, часть которых должна быть устранена в созданной панели администрирования.

Объектом настоящего исследования является процесс администрирования моделирования и получения результатов.

Предмет исследования – методы распределенного администрирования поведения акторов и свойств модели города в реальном времени.

Диаграмма бизнес-процессов (“BPMN AS IS”) в случае с использованием приложения отображена на рисунке 1.

Одной из проблем текущих систем администрирования моделирования является отсутствие подходов к обработке и представлению состояния акторов в реальном времени. Системы отображают результат моделирования без возможности приостановить моделирование или перезапустить его во время текущего моделирования уже с другими модулями и/или свойствами.

В некоторых системах моделирования, таких как Ant Road Planner после начальной настройки свойств моделирования и запуска моделирования нет возможности динамически изменить какие-либо свойства модели в целом и акторов в частности в реальном времени. Только после отработки моделирования и получения результатов можно заново запустить моделирование на измененных свойствах.

Проблемой также можно назвать отсутствие динамического изменения диапазона обрабатываемых акторов на основе текущего пользовательского представления. Часто в начале моделирования выбирается конкретный фиксированный участок города, который уже в ходе моделирования невозможно масштабировать без остановки и перезапуска моделирования целиком.

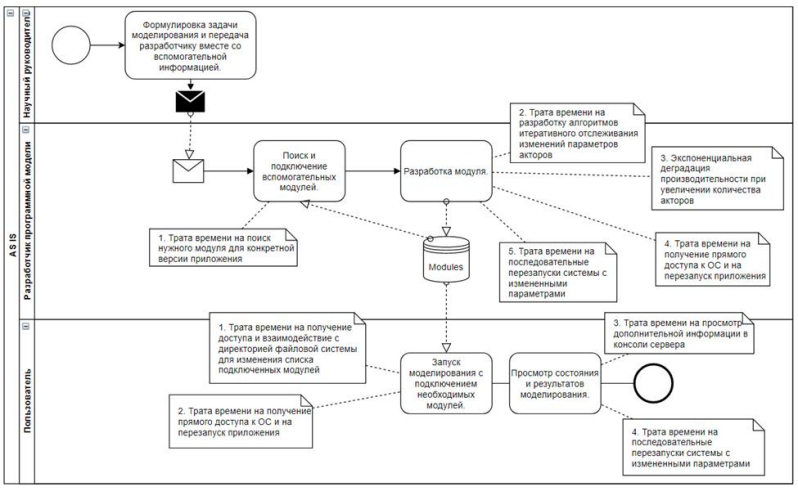

Рис. 1. BPMN AS IS

В связи с этим была поставлена задача разработать систему распределенного администрирования поведения акторов и свойств модели города в реальном времени с предоставлением возможности пользователю менять ход моделирования путем изменения свойств моделей и акторов. Система должна динамически менять диапазон обрабатываемых акторов на основе текущего пользовательского представления.

Для реализации веб клиента был выбран фреймворк Angular [9] версии 11 с использованием компонентов пользовательского интерфейса Ant Design [10].

3. Архитектура платформы

В архитектуре платформы учтены все потребности обновленных процессов. В данной работе протокол gRPC должен использоваться в одностороннем порядке (от сервера к клиенту), в архитектуру заложены дальнейшие планы по его использованию. Архитектура платформы представлена на рисунке 2.

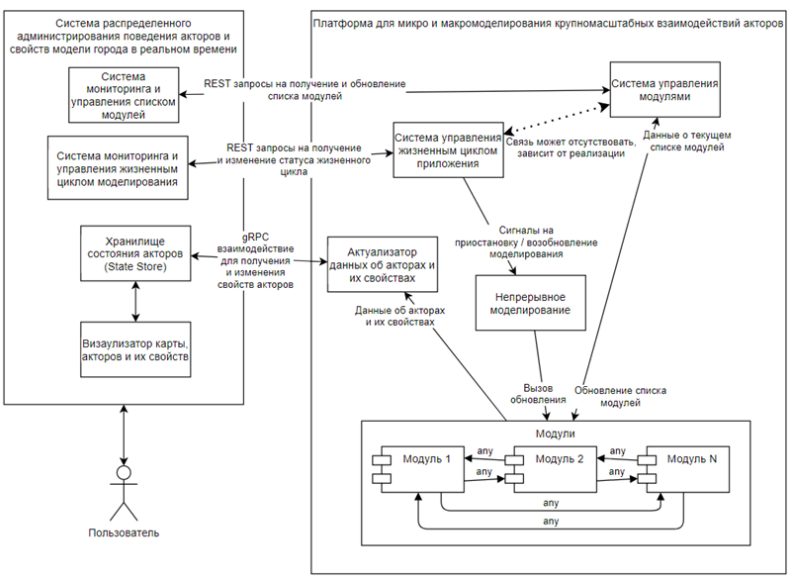

Рис. 2. Архитектура платформы

4. Описание методов применяемых в исследовании

Для решения задачи с разработкой подхода к обработке и представлению состояния акторов в реальном времени был проанализирован proto3 [11] API платформы моделирования:

```
syntax = “proto3”;

option csharp_namespace = “OSMLS.Map”;

package map;

import “google/protobuf/empty.proto”;

Service MapService {
  rpc Updates (google.protobuf.Empty) returns (stream MapFeaturesCluster);
}

message MapFeaturesCluster {
  repeated MapFeature features = 1;
}

message MapFeature {
  string type_full_name = 1;
  string features_geo_json = 2;
  string open_layers_style = 3;
}
```

На основе API можно выделить несколько методов, которые понадобятся для обработки и представления состояния акторов. Одним из таких методов является разделение акторов по их типу (из поля type_full_name) на группы, представляющие собой слои. Разделение на слои открывает ряд возможностей, одной из которых является возможность отключать отображение некоторых типов акторов в случае необходимости.

Другой важной возможностью, а также следующим используемым методом, открываемой с помощью использования слоев, является возможность применения к отдельным слоям конкретных стилей (из поля open_layers_style). Данный стиль представляет собой javascript код, который должен быть выполнен, прежде чем будет получен объект класса Style.

Метод отображения акторов в конкретных координатах должен базироваться на формате GeoJSON [12], т.к. данные именно в таком формате могут быть получены из свойства features_geo_json. Т.к. для решения большинства других подзадач уже используются методы, предоставляемые библиотекой OpenLayers, для работы с GeoJSON форматом может также использоваться методы [13] из данной библиотеки, направленные на работу с данным форматом.

Для решения задачи с разработкой подхода к представлению и изменению свойств модели в целом и акторов в частности в реальном времени может быть использован ряд методов из REST API платформы моделирования. Так, группа методов State позволяет управлять жизненным циклом моделирования, которое может быть запущено, временно приостановлено и остановлено. Метод Assemblies предоставляет возможность добавлять в приложение новые сборки с модулями и с зависимостями модулей, а группа методов Modules позволяет получать весь список модулей, полученный из добавленных сборок, а также предоставляет возможность управлять списком модулей, участвующих в процессе моделирования (конкретные модули из общего списка можно активировать и деактивировать для модели).

Для решения задачи с разработкой подхода динамического изменения диапазона обрабатываемых акторов на основе текущего пользовательского представления должен быть разработан метод клиент-серверного взаимодействия, позволяющий передавать от

клиента к серверу и хранить для каждого из клиентов текущее пользовательское представление (т.е. границы той зоны на экране, которая отображается пользователю).

Таким образом, были кратко описаны методы, которые будут использованы в исследовании при создании целевого метода распределенного администрирования поведения акторов и свойств модели города в реальном времени.

5. Реализация панели администрирования

5.1. Генерация REST API и gRPC-web инфраструктуры

Всю сгенерированную инфраструктуру было принято добавлять по пути src/app/generated, поэтому данный путь был добавлен в стандартный .gitignore файл.

Для получения swagger.json файла, используемого для генерации, вызывается серверный маршрут /swagger/v1/swagger.json. Данный маршрут создается автоматически, с помощью библиотеки Swashbuckle.AspNetCore. После чего данный файл добавляется в корень проекта.

Для генерации инфраструктуры используется npm модуль ng-openapi-gen.

Для генерации typescript gRPC-web инфраструктуры [14] установлен protoc с возможностью доступа к нему по соответствующей команде.

После чего добавлен npm модуль ts-protoc-gen, позволяющий преобразовывать сгенерированный с помощью protoc javascript код в typescript код. Также были добавлены модули для поддержки gRPC-web и google-protobuf:

После чего создан скрипт в package.json файле для генерации инфраструктуры на основе map.proto;

Сгенерированный клиент не требует внедрения с помощью DI и может быть получен напрямую из пространства имен grpc.

5.2. Реализация управления сборками

Управление сборками осуществляется в AssemblyCompositorComponent, дочернем компоненте AssemblyComponent. Для загрузки файлов используется компонент NzUploadModule и AssembliesService.

Вид итогового компонента представлен на рисунке 3. При нажатии на кнопку компонента открывается окно файловой системы с предложением выбрать один или несколько файлов сборок. После выбора, сборки сразу же загружаются на сервер.

Рис. 3. Вид компонента “Add Assemblies”

5.3. Реализация управления модулями

Управление модулями осуществляется в ModuleManagerComponent, дочернем компоненте ModuleComponent. Для этого используется ModulesService.

Вид итогового компонента представлен на рисунке 4. Каждая из кнопок компонента может быть активна или неактивна, в зависимости от текущего состояния модели.

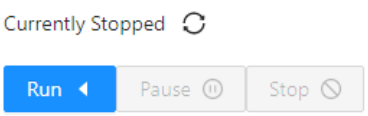

Рис. 4. Вид кнопок при остановке моделировании

5.4. Реализация управления состоянием модели

Управление состоянием модели осуществляется в ModelStateManagerComponent, дочернем компоненте ModelComponent. Для этого используется StateService.

Вид итогового компонента представлен на рисунке 5. Данный компонент поддерживает множественный выбор модулей для модели.

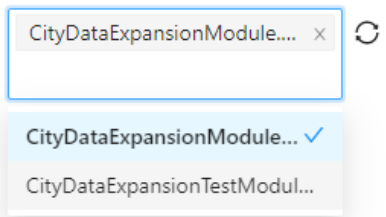

Рис. 5. Компонент с выбранной моделью

После выбора или отмены выбора одного из модулей, изменения сразу же отправляются на сервер.

5.5. Реализация карты и отображения агентов

Для отображения карты и агентов используется библиотека OpenLayers. Она установлена в проект с помощью npm.

Объекты карты отображаются на основе текущего MapFeaturesCluster из map.proto. Для получения данного объекта MapComponent создается gRPC-web клиент, с указанием адреса сервера. MapFeaturesCluster отображается на карте таким образом, что каждый объект типа MapFeature занимает отдельный слой со своим стилем, обозначенным в соответствующем open_layers_style. Стиль извлекается из строковой переменной с помощью функции eval. Итоговый вариант карты вместе с панелью администрирования отображен на рисунке 6.

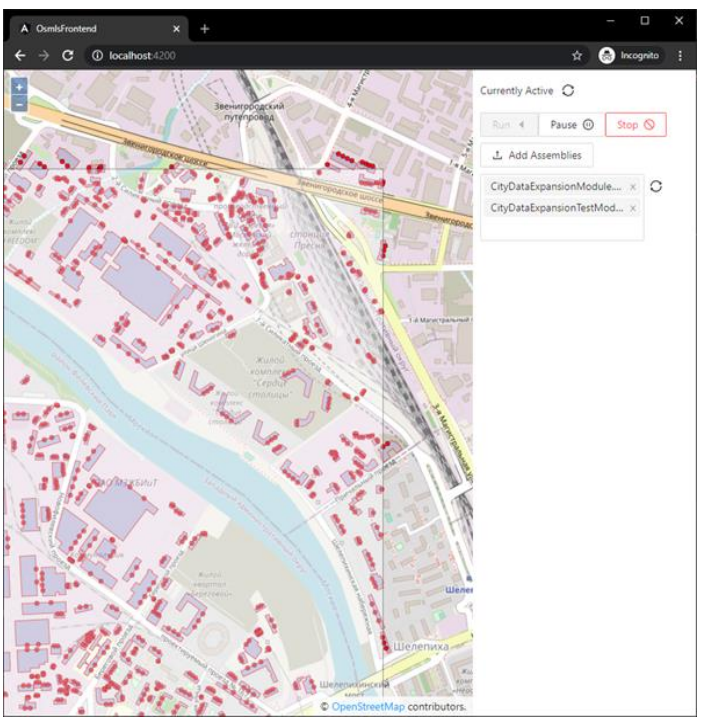

Рис. 6. Карта с панелью администрирования

6. Заключение

Для взаимодействия с платформой моделирования перемещений и взаимодействий акторов в рамках участка карты города был разработан web клиент, выполняющий роль панели администрирования и предоставляющий:

— пользовательское отображение акторов в реальном времени с конкретными стилями на основе типов акторов;
— возможность пользователю управлять списком модулей модели;
— возможность пользователю управлять жизненным циклом моделирования;
— возможность пользователю влиять на ход моделирования, путем изменения свойств модели и акторов;
— информацию для платформы моделирования о координатах и разрешении текущего пользовательского представления для дополнительных возможностей масштабирования и оптимизации.

В сравнении с существующими решениями данная панель администрирования отличается возможностью работы с конкретной платформой моделирования, которая в свою очередь предлагает ряд преимуществ, относительно других решений в сфере моделирования, таких как:

— кроссплатформенность;
— открытый исходный код;
— расширяемость функционала с помощью модулей.

Таким образом, разработанная панель администрирования делает платформу моделирования более доступной для взаимодействия с конечным пользователем, предоставляя графический пользовательский интерфейс, вместо программного (REST и gRPC API).

Авторы выражают благодарность коллегам по лаборатории UCLab за участие в разработке проекта Live.UrbanBasis.com.

Литература

[1] Ant Road Planner // Сервис для симуляции движения пешеходов. URL: https://antroadplanner.ru/ (дата обращения: 08.06.2021).
[2] NetLogo // Агентно-ориентированный язык программирования и интегрированная среда разработки. URL: https://www.netlogoweb.org/ (дата обращения: 08.06.2021).
[3] AnyLogic // Программное обеспечение для имитационного моделирования, разработанное российской компанией The AnyLogic Company. URL: https://www.anylogic.ru/ (дата обращения: 08.06.2021).
[4] Multi-agent Approach to Modeling the Dynamics of Urban Processes (on the Example of Urban Movements) / D. Parygin, A. Usov, S. Burov, N. Sadovnikova, P. Ostroukhov, A. Pyannikova // Communications in Computer and Information Science: Proceedings of the 6th International Conference on Electronic Governance and Open Society: Challenges in Eurasia (EGOSE 2019), St. Petersburg, Russia, 13–14 November 2019. Springer, 2020. Vol. 1135. P. 243–257. DOI: 10.1007/978-3-030-39296-3_18.
[5] Клиент-серверное решение для моделирования массовых перемещений акторов в условиях городской среды / С. С. Буров, А. А. Пьянникова, Е. И. Сетов, Д. С. Парыгин // России – творческую молодёжь: материалы XIII Всероссийской науч.-практ. студ. конф., Камышин, 20–21 апр. 2020 г. / ВолгГТУ. Волгоград, 2020. Т. 3. С. 34–35.
[6] Development of Scenarios for Modeling the Behavior of People in an Urban Environment / A. Anokhin, S. Burov, D. Parygin, V. Rent, N. Sadovnikova, A. Finogeev // Studies in Systems, Decision and Control: Society 5.0: Cyberspace for Advanced Human-Centered Society. – Springer, 2021. Vol. 333. P. 103–114. DOI: 10.1007/978-3-030-63563-3_9.

[7] Парыгин, Д. С. Управляемое данными развитие урбанизированных территорий: моногр. / Д. С. Парыгин ; ВолгГТУ. Волгоград, 2021. 124 с.
[8] Бурова А.А., Буров С.С., Парыгин Д.С., Финогеев А.А., Рент В.Э. Разработка модуля управления данными об объектах на онлайн-карте города // Прикаспийский журнал: управление и высокие технологии. Астрахань: ФГБОУ ВО «Астраханский государственный университет», 2021. № 1. С. 18-27. URL: https://hi-tech.asu.edu.ru/?articleId=1290 (дата обращения: 05.03.2021).
[9] Introduction to the Angular Docs // Angular is a platform for building mobile and desktop web applications. URL: https://angular.io/docs (дата обращения: 05.03.2021).
[10] Ant Design of Angular // NG-ZORRO. URL: https://ng.ant.design/docs/introduce/en (дата обращения: 05.03.2021).
[11] Language Guide (proto3) // Google Developers is Google's site for software development tools and platforms, application programming interfaces (APIs), and technical resources. URL: https://developers.google.com/protocol-buffers/docs/proto3 (дата обращения 04.06.2021).
[12] The GeoJSON Format // The IETF Datatracker is the day-to-day front-end to the IETF database for people who work on IETF standards. URL: https://datatracker.ietf.org/doc/html/rfc7946(дата обращения 04.06.2021).
[13] ol/format/GeoJSON~GeoJSON] // OpenLayers makes it easy to put a dynamic map in any web page. It can display map tiles, vector data and markers loaded from any source. URL: https://openlayers.org/en/latest/apidoc/module-ol_format_GeoJSON-GeoJSON.html (дата обращения 04.06.2021).
[14] Create gRPC-web app with Angular 8 on Windows // Anthony Giretti's .NET blog. URL: https://anthonygiretti.com/2020/03/29/grpc-asp-net-core-3-1-how-to-create-a-grpc-web-client-examples-with-angular-8-and-httpclient/ (дата обращения: 05.03.2021).

Implementation of the Administration Panel for the Multi-Agent Modeling Platform

A.A. Burova, S.S. Burov, D.S. Parygin

Volgograd State Technical University

This article discusses the design and implementation of a web client that acts as an administration system (panel) for a platform for multi-agent modeling of movements and interactions of actors within a city map area. All modeling logic in this platform is implemented directly in modules, while the modeling platform only calls it for specific, connected modules.
The modeling platform is implemented on the ASP.NET Core 5.0 framework.
For the implementation of the web client, the Angular 11 framework was chosen with the Ant Design UI components

Keywords: Simulation, city, C#, modularity, client-server, administration panel

Reference for citation: Burova A.A., Burov S.S., Parygin D.S. Implementation of the Administration Panel for the Multi-Agent Modeling Platform // Information Society: Education, Science, Culture and Technology of Future. Vol. 5 (Proceedings of the XXIV International Joint Scientific Conference «Internet and Modern Society», IMS-2021, St. Petersburg, June 24-26, 2021). - St. Petersburg: ITMO University, 2021. P. 19 – 27. DOI: 10.17586/2587-8557-2021-5-19-27

Reference

[1] Ant Road Planner // Service for simulating the movement of pedestrians. URL: https://antroadplanner.ru/ (access date: 08.06.2021).

[2] NetLogo // Agent-oriented programming language and integrated development environment. URL: https://www.netlogoweb.org/ (access date: 08.06.2021).
[3] AnyLogic // Simulation software developed by the Russian company The AnyLogic Company. URL: https://www.anylogic.ru/ (access date: 08.06.2021).
[4] Multi-agent Approach to Modeling the Dynamics of Urban Processes (on the Example of Urban Movements) / D. Parygin, A. Usov, S. Burov, N. Sadovnikova, P. Ostroukhov, A. Pyannikova // Communications in Computer and Information Science: Proceedings of the 6th International Conference on Electronic Governance and Open Society: Challenges in Eurasia (EGOSE 2019). St.Petersburg, Russia, 13–14 November 2019. Springer, 2020. Vol. 1135. P. 243–257. DOI: 10.1007/978-3-030-39296-3_18.
[5] Client-server solution for modeling mass movements of actors in an urban environment / S. S. Burov, A. A. Pyannikova, E. I. Setov, D. S. Parygin // Russia - creative youth: materials of the XIII All-Russian scientific and practical student conference, Kamyshin, April 20–21. 2020. VSTU. Volgograd, 2020. V. 3. P. 34–35. [In Russian].
[6] Development of Scenarios for Modeling the Behavior of People in an Urban Environment / A. Anokhin, S. Burov, D. Parygin, V. Rent, N. Sadovnikova, A. Finogeev // Studies in Systems, Decision and Control: Society 5.0: Cyberspace for Advanced Human-Centered Society. Springer, 2021. Vol. 333. P. 103–114. DOI: 10.1007/978-3-030-63563-3_9.
[7] Parygin D.S. Data-driven development of urbanized territories: monograph. / D. S. Parygin; VSTU. Volgograd, 2021. 124p. [In Russian].
[8] Burova A.A., Burov S.S., Parygin D.S., Finogeev A.A., Rent V.E. Development of an object data management module on an online city map // Caspian Journal: Management and High Technologies. 2021. № 1. 18-27. URL: https://hi-tech.asu.edu.ru/?articleId=1290 (access date: 05.03.2021). [In Russian].
[9] Introduction to the Angular Docs // Angular is a platform for building mobile and desktop web applications. URL: https://angular.io/docs (access date: 05.03.2021).
[10] Ant Design of Angular // NG-ZORRO. URL: https://ng.ant.design/docs/introduce/en (access date: 05.03.2021).
[11] Language Guide (proto3) // Google Developers is Google's site for software development tools and platforms, application programming interfaces (APIs), and technical resources. URL: https://developers.google.com/protocol-buffers/docs/proto3 (access date: 04.06.2021).
[12] The GeoJSON Format // The IETF Datatracker is the day-to-day front-end to the IETF database for people who work on IETF standards. URL: https://datatracker.ietf.org/doc/html/rfc7946(access date: 04.06.2021).
[13] ol/format/GeoJSON~GeoJSON] // OpenLayers makes it easy to put a dynamic map in any web page. It can display map tiles, vector data and markers loaded from any source. URL: https://openlayers.org/en/latest/apidoc/module-ol_format_GeoJSON-GeoJSON.html (access date: 04.06.2021).
[14] Create gRPC-web app with Angular 8 on Windows // Anthony Giretti's .NET blog. URL: https://anthonygiretti.com/2020/03/29/grpc-asp-net-core-3-1-how-to-create-a-grpc-web-client-examples-with-angular-8-and-httpclient/ (access date: 05.03.2021).