Wojskowa Akademia Techniczna

Wydział Elektroniki

Rodzaj pracy: magisterska
Dyplomant: **mgr inż. Korneliusz Duduś**
Promotor: **ppłk dr inż. Krzysztof Maślanka**

PORÓWNANIE WYDAJNOŚCI INTERFEJSÓW PROGRAMISTYCZNYCH NA PRZYKŁADZIE REST API, GRAPHQL I GRPC

Wprowadzenie

Interfejsy programistyczne zostały opracowane w odpowiedzi na zapotrzebowanie wymiany informacji pomiędzy aplikacjami komputerowymi na poziomie lokalnym lub globalnym. W miarę rozwoju tej dziedziny programowania zaczęto opracowywać coraz więcej przeróżnych rozwiązań ukierunkowanych na odpowiednie przypadki komunikacji. Większość interfejsów programistycznych opiera się na tych samych protokołach komunikacyjnych takich jak na przykład HTTP. Pełnią one swoją funkcję na poziomie warstwy aplikacji, a główne różnice między nimi polegają na sposobie wykorzystania protokołów komunikacyjnych.

Celem niniejszej pracy dyplomowej jest porównanie wydajności interfejsów programistycznych na przykładzie REST API, GraphQL oraz gRPC w zastosowaniu systemów o architekturze mikro usług, będących jednym z najbardziej popularnych modeli architektonicznych stosowanych w dzisiejszych czasach na potrzeby usług zamieszczonych w sieci Internet. Interfejsy programistyczne których porównanie jest celem pracy, zostały wybrane ze względu na ich popularność oraz elastyczność mającą ogromne znaczenie w czasach, gdy dziedzina informatyki oraz teleinformatyki tak dynamicznie się rozwija.

Zadania wykonane na potrzeby pracy magisterskiej:

1. Analiza rozwiązań wspierających architekturę mikrousług.
2. Charakterystyka interfejsów programistycznych REST API, GraphQL i gRPC.
3. Opracowanie koncepcji zastosowania mikrousług w wybranym systemie.
4. Implementacja systemu z architekturą mikrousługową z wybranymi interfejsami programistycznymi.
5. Opracowanie metodyki testów porównawczych i realizacja testów.
6. Podsumowanie i wnioski.

Badania

Pracę poświęcono analizie mocnych i słabych stron omawianych interfejsów programistycznych. W przypadku interfejsu programistycznego REST do jego mocnych stron można zaliczyć: otwartość - pozwalającą na wymianę informacji w łatwy sposób bez konieczności podejmowania poważniejszych kroków w celu poprawnej integracji, łatwość analizy – dane są przesyłane w postaci tekstowej zrozumiałej przez człowieka co znacznie ułatwia analizę przeprowadzonej komunikacji, obsługa przeglądarki – interfejs programistyczny REST jest w pełni wspierany przez przeglądarki internetowe co pozwala na przeprowadzenie wstępnej komunikacji bez konieczności pisania programów.

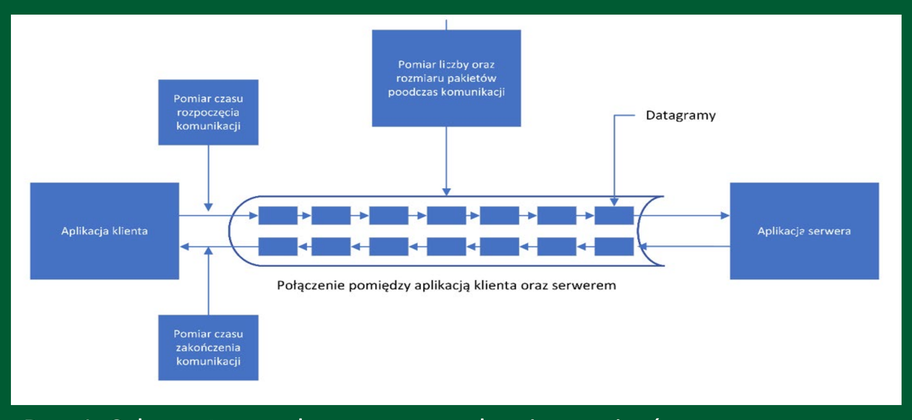
Interfejs programistyczny GraphQL w porównaniu do interfejsu REST, pozwala dodatkowo na obsługę zapytań tworzonych przez użytkownika.

Dzięki takiej funkcjonalności istnieje możliwość ograniczenia ilości punktów dostępowych do jednego oraz zminimalizowania przesyłanych danych do ilości wymaganej.

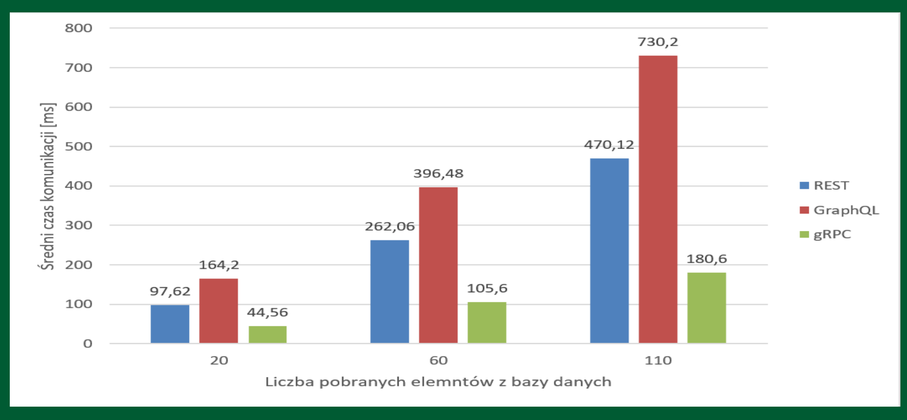
Interfejs programistyczny gRPC dzięki oparciu komunikacji na protokole HTTP wersji 2 pozwala na przesyłanie danych w postaci binarnej, przesyłanie strumieniowe oraz kompresję nagłówków HPACK dzięki czemu staje się on najbardziej wydajnym spośród omawianych interfejsów programistycznych.

Następnie opracowano oraz zaimplementowano systemy informatyczne wykorzystujące omawiane interfejsy programistyczne do przesyłania danych pomiędzy aplikacjami w środowisku lokalnym. Pomiary w środowisku lokalnym przeprowadzono w celu zminimalizowania wpływu czynników zewnętrznych na uzyskane wyniki pomiarów.

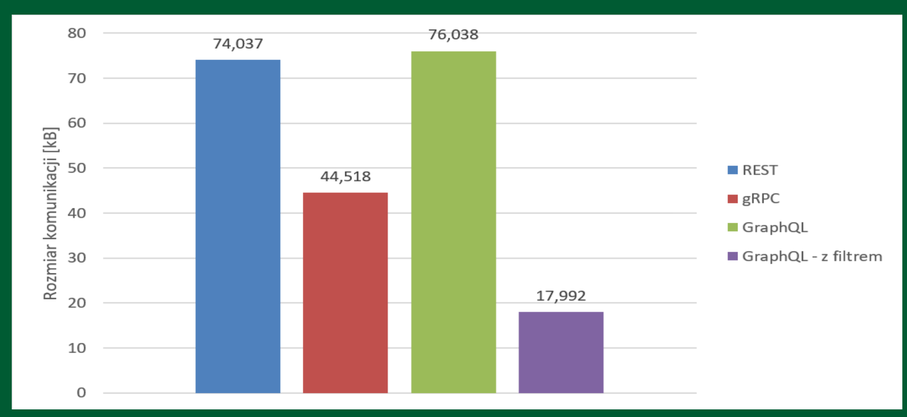
Przygotowano kilka scenariuszy testowych podczas których została zmierzona różnica czasu pomiędzy rozpoczęciem oraz zakończeniem komunikacji w celu wykazania mocnych oraz słabych stron omawianych interfejsów programistycznych. Dodatkowo poza pomiarem czasu potrzebnym na przeprowadzenie komunikacji, zostało zmierzone obciążenie generowane w sieci poprzez badane interfejsy programistyczne w celu zbadania przydatności tych technologii w sieci o ograniczonej przepustowości.



Rys. 1. Schemat sposobu przeprowadzenia pomiarów.



Rys. 2. Całkowity czas komunikacji dla przesyłania strumieniowego.



Rys. 3. Wykres przedstawiający całkowity rozmiar komunikacji.

Wnioski

Na podstawie uzyskanych wyników pomiarów można stwierdzić, iż najbardziej wydajnym interfejsem programistycznym w typowych zastosowaniach jest gRPC. Ze względu na wydajność interfejsu gRPC, automatyczne generowanie kodu oraz możliwość przesyłania informacji za pomocą strumienia danych jest on najlepszym wyborem, spośród badanych interfejsów, do zastosowania w systemach o architekturze mikro usług. Natomiast jako punkty końcowe takiego systemu odpowiedzialne za komunikację z usługami innych firm zalecane jest stosowanie interfejsu REST lub GraphQL.