ИКОНОМИЧЕСКИ УНИВЕРСИТЕТ – ВАРНА

КАТЕДРА „ИНФОРМАТИКА“



**РЕФЕРАТ**

по дисциплината

**„Интернет технологии и комуникации“**

на тема:

**Облачни комуникационни модели в дистрибутирана система за управление на поръчки**

Докторант:

Йордан Йорданов

Научен ръководител:

доц. д-р Павел Петров

Варна, 2022

**Съдържание**

[Списък на съкращенията 2](#_Toc97478350)

[Въведение 3](#_Toc97478351)

[1. Синхронна комуникация между микроуслуги 4](#_Toc97478352)

[1.1 Механизъм за комуникиране чрез трансфер на репрезентативно състояние 4](#_Toc97478353)

[1.2 Механизъм за заявки към отдалечени процедури 5](#_Toc97478354)

[1.3 Недостатъци на синхронната комуникация между микроуслуги 5](#_Toc97478355)

[2. Асинхронна комуникация между микроуслуги 6](#_Toc97478356)

[2.1 Въведение в проблема “Съгласуваност между услугите” 6](#_Toc97478357)

[2.2 Асинхронна комуникация между различните микроуслуги с помощта на посредник на съобщения 7](#_Toc97478358)

[3. Комуникационни модели за достъп до бекенда 7](#_Toc97478359)

[3.1 Директна комуникация на клиент с микроуслуга 7](#_Toc97478360)

[3.2 Шлюз за приложете програмни интерфейси 8](#_Toc97478361)

[3.3 Комуникация в реално време 8](#_Toc97478362)

[Заключение 10](#_Toc97478363)

[Използвана литература 11](#_Toc97478364)

Списък на съкращенията

|  |  |  |
| --- | --- | --- |
|  |  |  |
| REST | Representational State Transfer | |
| gRPC | Google Remote Procedure Call | |
| API | Application Programming Interface | |
| HTTP | Hypertext Transfer Protocol | |
| CNCF | Cloud Native Computing Foundation | |
|  |  | |
|  |  | |

Въведение

**Актуалността на изследваната тема** се обуславя от тенденцията облачните технологии да се превръщат в основна платформа на продуктовите предприятията, които искат да внедряват и изпълняват своите дигитални бизнес услуги.

**Обект на изследване** в настоящия труд е информационна система базирана на архитектура, състояща се от много малки, независими микроуслуги. Всяка микроуслуга се изпълнява в отделен процес като контейнер, който е разположен в клъстер, управляван от инструмент за оркестрация, който отговаря за внедряване и управлението.

**Целта** на реферата е да представи сътрудничеството или т.н. информационно и икономическо взаимодействие между подсистемите, изискващо надеждни модели за комуникация. От архитектурна гледна точка, комуникационните протоколи са важно дизайнерско решение. В и извън клъстера, микро-услугите комуникират чрез технологии за изпращане и получаване на данни.

**Основните точки**, които са поставени за изпълнение на целта, са следните:

-  да се изследват възможните комуникационни модели между клиентско приложение и микро-услугите от бекенда;

-  да се предложат основни принципи и добри практики при изграждането на комуникация в приложения, базирани на облак;

**Основната теза** на изследването е, че едно от главните предизвикателства е да се внедрят бизнес процеси от край до край, като същевременно се поддържа последователност и съгласуваност в микроуслугите. Главна причина са възникнето на частични повреди в отделните компоненти на системата.

# Синхронна комуникация между микроуслуги

Може да разграничим два основни вида комуникация, които са използвани между различните микро-услуги: синхронна и асинхронна комуникация. Синхронната разчита главно на протоколите HTTP/HTTPS. Зависи от комуникацията „заявка-отговор“.

## Механизъм за комуникиране чрез трансфер на репрезентативно състояние

През 2000 г. Рой Филдинг предлага прехвърляне на представително състояние (REST) като архитектурен подход за проектиране на уеб услуги. REST е архитектурен стил за изграждане на разпределени системи, базирани на хипермедия. Той е независим от всеки основен протокол и не е непременно обвързан с HTTP. Въпреки това, най-често срещаните реализации на REST API използват HTTP като протокол на приложението.

Основно предимство на REST е, че той използва отворени стандарти и не обвързва внедряването на API или клиентските приложения с конкретна реализация. Например, уеб услуга REST може да бъде написана в ASP.NET и клиентските приложения могат да използват всеки език или набор от инструменти, които могат да генерират HTTP заявки и да анализират HTTP отговори.

REST-базирана комуникация е една от най-използваните в днешно време. Тя изисква услугите да са настроени с REST API. За постигането на това разработчиците използват HTTP глаголите: ВЗЕМЕТЕ, ПУБЛИКУВАТЕ, ПОСТАВЯТЕ, ИЗТРАЙТЕ (GET, POST, PUT, DELETE) на един и съши URL адрес. Услугите от този вид са само-описващи се, а протоколът може да бъде използван за комуникация от няколко вида:

* Браузър към уеб сървър
* Мобилно приложение към уеб сървър
* Настолно приложение към уеб сървър
* Сървър към сървър

## Механизъм за заявки към отдалечени процедури

gRPC е модерна, високопроизводителна рамка, която развива дистанционно извикване на процедури (RPC) протокол. На ниво приложение, gRPC рационализира съобщенията между клиенти и бек-енд услуги. Произхождащ от Google, gRPC е с отворен код и е част от Cloud Native Computing Foundation екосистема от облачни предложения.

Типичното клиентско приложение на gRPC ще разкрие локална функция да извиква друга функция на отдалечена машина като процес, който реализира бизнес операция. RPC абстрахира мрежовата комуникация и изпълнение от точка до точка.

## Недостатъци на синхронната комуникация между микроуслуги

Всички услуги, осъществяващи синхронна комуникация имат много знания една за друга. Създава се тясна връзка между различните микроуслуги, което нарушава една от предпоставките за използване на микросервизи. Всеки път, когато бъде добавена нова услуга и тя трябва да бъде актуализирана за нещо, което се случва в системата, ще трябва да се направят промени в кода, за да бъде извикана и тази нова услуга. Така добавянето на нови услуги става все по-трудно. С течение на времето системата може да бъде натоварена на места, които не сме забелязали в началото.

Недостатъците на използването на синхронна комуникация ще бъде малък мост, към следващата точка, където ще разгледаме асинхронната комуникация.

# Асинхронна комуникация между микроуслуги

Кратко въведение в асинхронната комуникация за микроуслуги, различните опции за използване. Протокол за комуникация е AMQP. Може да има един или множество приемници на съобщенията.

## Въведение в проблема “Съгласуваност между услугите”

Предизвикателството е да се внедрят бизнес процеси от край до край, като същевременно се поддържа последователност в услугите.  
За да представим проблема напълно, е важно да разгледаме две диаграми на монолитната или ориентираната към услуги версии на системата.

* нито една услуга не трябва да включва таблици/хранилище от друга и никога не трябва да извиква директни заявки към тях
* комуникация, базирана на събития. модел за публикуване-абониране
* Предизвикателството относно комуникацията не е толкова в протоколите, а повече за стила, защото когато възникне повреда – колкото по-свързана е системата, толкова по-големи проблеми ще се получат
* Частични повреди, проектирате на системата, като се вземе предвид общите рискове

## Асинхронна комуникация между различните микроуслуги с помощта на посредник на съобщения

Най-често срещаният подход е, използване на посредник за корпоративни съобщения с опашки и теми за публикуване-абониране. Сервизната шина се използва за отделяне на приложения и услуги един от друг, осигурявайки следните предимства:

* Работа за балансиране на натоварването между конкуриращи се работници
* Безопасно маршрутизиране и прехвърляне на данни и контрол през границите на услуги и приложения
* Координиране на транзакционна работа, която изисква висока степен на надеждност.

# Комуникационни модели за достъп до бекенда

С въвеждането на потребителски интерфейс трябва да представим как клиентските приложения взаимодействат с различните услуги. Ще проучим как предния край на системата достъпва до инфраструктурата на микроуслугите.

## Директна комуникация на клиент с микроуслуга

Използва се, когато различни части от страницата на клиента изискват различни микроуслуги. Обикновено клиентът извиква балансьор на натоварване, който изисква данни от вътрешната микросервизна инфраструктура.

Някои от недостатъците на този подход:

* Прекалено много „обиколки“ в Интернет (извън вътрешната микросервизна мрежа)
* Микроуслугите трябва да бъдат изложени на „външния свят“
* Междусекторни проблеми като удостоверяване и оторизация
* Използване на синхронна комуникация като HTTP
* Различните клиентски приложения изискват различни API (уеб срещу мобилни клиенти)

## Шлюз за приложете програмни интерфейси

Предоставя еднократна крайна точка за група микроуслуги. Наподобява модела за дизайн: „фасадата“. Известен е също като „backend for frontend“. Изгражда се за конкретните нужди на клиента, Действа като пълномощник между клиентите и микроуслугите. Може да осигури удостоверяване, кеширане и други проблеми.

API шлюзът може да се превърне в “анти-модел“ като пълно монолитно приложение: съдържащо твърде много крайни точки, обединяващо всички микроуслуги, унищожавайки техните предимства.

API шлюзовете също трябва да бъдат отделени за всеки клиент, разделен от логически групи въз основа на бизнес граници. протокол за пренос на данни могат да бъдат HTTP или gRPC

## Комуникация в реално време

Комуникацията в реално време може да се постигне с HTTP уеб сокети, изградени с резервни механизми. Използва се, когато изпращаме данни от услугите директно към клиентите

ASP.NET Core има SignalR като комуникационна технология в реално време. SignalR е библиотека за ASP.NET разработчици, която опростява процеса на добавяне на уеб функционалност в реално време към приложенията. Уеб функционалността в реално време е възможността сървърният код да изпраща съдържание към свързани клиенти незабавно, когато стане достъпно, вместо сървърът да чака клиент да поиска нови данни. SignalR може да се използва за добавяне на всякакъв вид уеб функционалност "в реално време". Всеки път, когато потребител обновява уеб страница, за да види нови данни, то, тя е кандидат за използване на SignalR.

Заключение

Комуникацията на услугите се превръща в важно дизайнерско решение при конструирането на облачно приложение. В реферата проучвахме модели на комуникация, които са естествени за облачната инфраструктура: клиентите от предния край комуникират с микроуслуги от задния край, платформи за API шлюз и комуникация в реално време. Разгледахме как комуникират микроуслугите с други бек-енд услуги, както синхронната HTTP комуникация, така и асинхронни съобщения между услугите. Покрихме gRPC, модерна, високопроизводителна рамка, която развива дистанционно извикване на процедури.

Използвана литература

1. БИЖКОВ, Г., КРАЕВСКИ, В. (2007) *Методология и методи на педагогическите изследвания*. УИ „Св. Климент Охридски“.
2. ВАНКОВА, Д. (2014) *Делфи – методът, същност и изследователски опит.* МУ – Варна. Известия на съюза на учените – Варна. с. 59-66.
3. ГАВРАИЛОВ, E. (2014) *Основи на научните изследвания.* УИ ВСУ „Черноризец Храбър“.
4. ГАНЧЕВ, Г., ДЕЛЧЕВ, М. (2013) *Методика на педагогическите изследвания*. [Онлайн] Достъпно на: <https://obuch.info/metodika-na-pedagogicheskite-izsledvaniya.html> [Достъпено: 20 декември 2021].
5. ДИМИТРОВ, Н. (2013) *Въведение в научните изследвания*. „Интелексперт-94“.
6. ИИКТ – БАН. (2020) *Ефективни методи и алгоритми за Монте Карло симулации, анализ на чувствителността и стохастични оптимизации.* [Онлайн] Достъпно на: http://ict.acad.bg/?page\_id=557 [Достъпено: 22 януари 2022].
7. КОРОВКИНА, Н., ЛЕВОЧКИНА, Г. (2022) *Методика подготовки исследовательских работ студентов*. [Онлайн] Достъпно на: https://intuit.ru/studies/courses/11980/1160/info [Достъпено: 15 декември 2021].
8. НЕМИНСКА, Р. (2015) *Методи на интердисциплинарно обучение.* Българско списание за образование. Брой 2. с.115-125.
9. ОРЛОЕВ, Н. (2002) *Методология на научните изследвания*. РУ „А. Кънчев”.
10. ПАПАНЧЕВ, Т. (2015) *Обзор на методите за моделиране на надеждността на електронни изделия.* Сп. „Компютърни науки и комуникации”. БСУ – Бургас. Том 4. с. 34-43.