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**

[Въведение 3](#_Toc97024150)

[1. Синхронна комуникация между микроуслуги 3](#_Toc97024151)

[1.1 Проучване на REST микросервизна архитектура, изградена с .NET Core 3](#_Toc97024152)

[1.2 Разглеждане на gRPC 4](#_Toc97024153)

[1.3 Недостатъци на синхронната комуникация между микроуслуги 5](#_Toc97024154)

[2. Асинхронна комуникация между микроуслуги 5](#_Toc97024155)

[2.1 Въведение в проблема “Съгласуваност между услугите” 5](#_Toc97024156)

[2.2 Асинхронна комуникация между различните микроуслуги с помощта на посредник на съобщения 6](#_Toc97024157)

[3. Комуникационни модели за достъп до бекенда 6](#_Toc97024158)

[3.1 Директна комуникация на клиент с микроуслуга 7](#_Toc97024159)

[3.2 Шлюз за приложете програмни интерфейси 7](#_Toc97024160)

[3.3 Комуникация в реално време 7](#_Toc97024161)

[Заключение 8](#_Toc97024162)

[Използвана литература 9](#_Toc97024163)

Въведение

В архитектурата на облачните системи, комуникационните протоколи се превръщат във важно дизайнерско решение. Най-често системите се реализират базирани на модел на независими микро-услуги. Всяка една от тях се изпълнява в отделен процес, разположен като контейнер в клъстер, управляван от инструмент за оркестрация, който отговаря за внедряването и управлението. В и извън клъстера, микро-услугите комуникират чрез технологии за изпращане и получаване на данни. Важни точки, които трябва да бъдат обмислени са:

-  Натоварване на мрежата

-  Устойчивост, т.е. повторен опит при неуспешна заявка

-   Всяка микро услуга трябва да удостоверява и разрешава повиквания

-   Шифроване/дешифриране на съобщения

Едно от главните предизвикателства е да се внедрят бизнес процеси от край до край, като същевременно се поддържа последователност и съгласуваност в микроуслугите. Защото когато възникне повреда - колкото по-свързани са отделните компоненти в системата, толкова по-големи проблеми се появяват. Често се случат частични повреди, които трябва да бъдат взети в предвид при проектиране на системата.

# Синхронна комуникация между микроуслуги

## Проучване на REST микросервизна архитектура, изградена с .NET Core

Тъй като предишният модул все още е свеж в паметта ви, ние сме готови да започнем да изследваме как различните услуги за нашето решение GloboTicket ще работят заедно. Както ще разберете, различните услуги ще комуникират с помощта на набор от различни комуникационни технологии. Въпреки че GloboTicket е демонстрация и, разбира се, добавих тези различни технологии, за да мога да ви покажа различните опции, дори в решение от реалния живот, ще срещнете това в архитектура на микросервизи. Такова приложение трябва да отговаря на различни бизнес нужди и следователно начинът, по който се обработва комуникацията между различните услуги, зависи от проблема, който се опитваме да решим. В този модул ще се съсредоточим върху синхронната комуникация между някои от услугите на нашето решение GloboTicket. Нека започнем, като разгледаме какво ще изследваме в този модул. Тъй като това е основната тема на този модул, ще започнем с проучване на повече подробности за това как нашите микроуслуги ще комуникират синхронно. Мисля, че това е добро начало, за да привлечем всички към различните възможности, които имаме. Ще разгледаме различните опции за комуникация, но мисля, че REST ще се използва често, така че първо ще разгледаме как можем да използваме REST-базирана комуникация в нашата архитектура. Разбира се, комуникацията в този случай с REST ще изисква да настроим услуга с базиран на REST API.

## Разглеждане на gRPC

Второ, ще трябва да се свържем с услугата от друга микроуслуга и това е, което ще проучим по-нататък. След това ще разгледаме използването на gRPC; gRPC вече се поддържа напълно в .NET Core и в някои случаи може да бъде по-добър избор от обикновения REST. Ще проучим как можем да използваме gRPC и в нашето решение.

## Недостатъци на синхронната комуникация между микроуслуги

И накрая, ще трябва да разгледаме недостатъците на използването на синхронна комуникация в нашата архитектура и това ще бъде малкият мост, да кажем, вече към следващия модул, където ще разгледаме асинхронната комуникация.Представените теоретични постановки водят до заключението, че в основата на методологията и методиката стои *методът* – основна градивна единица на изследователския процес.

# Асинхронна комуникация между микроуслуги

глед Така че синхронната комуникация ще ни помогне, но след предишния модул е ​​ясно, че няма да можем да го използваме за цялата комуникация между нашите микросервизи за GloboTicket. Но в този модул ще решим всичко това, когато изследваме различните услуги, които комуникират асинхронно. Сега, преди да забравя, все още съм Джил Клийрън. Моля, не се колебайте да се свържете с всички въпроси, които може да имате относно този курс. И така, как ще се справим с този модул тогава? Ще започнем с кратко въведение за асинхронната комуникация за микроуслуги, какво е това и какви са различните опции за нашите микроуслуги да го използват?

## Въведение в проблема “Съгласуваност между услугите”

И накрая, ще трябва да решим проблема, който вече засегнахме в предишния модул за евентуална последователност, и ще видим, че тук асинхронната комуникация ще бъде решението. Много интересни теми. Надявам се, че сте готови. Да го направимили предмет на изучаване.

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

Най-често срещаният подход е, че ще използваме някакъв вид автобус и това ще направя първо. След това ще видим как можем да включим в услуга и за нас това ще бъде услугата за плащане, фонов работник и ще се използва за комуникация с външен доставчик на плащане. Асинхронната комуникация може да съществува и под формата на анкета на услуга, която в нашия случай ще бъде анкета на услуга, която съхранява събития, случили се в приложението, и по този начин комуникацията тук също ще бъде асинхронна.

# Комуникационни модели за достъп до бекенда

Прекарахме много време, за да разберем как можем да използваме ASP.NET Core, за да настроим различни микроуслуги за GloboTicket и съм доволен от това, което създадохме досега. Но да имате хубав набор от микроуслуги и да нямате реален начин за взаимодействие с тях, не, това не е много полезно. Ще трябва да внесем потребителски интерфейс, преден край или предни части, които ще взаимодействат с различните услуги, които сме създали. Добре дошли в последния модул от курса по комуникация на ASP.NET Core Microservices тук на Pluralsight. В този модул ще проучим как можем да въведем предния край за достъп до инфраструктурата на микроуслугите. Все още съм Джил Клийрен и ще ви преведа и през този модул. Всъщност имам две теми, които искам да обсъдя точно в този модул.

## Директна комуникация на клиент с микроуслуга

Първо, ще преминем от реализацията по подразбиране, която е известна като комуникация клиент-микроуслуга, до мястото, където искам да бъда, и това е с добавянето на шлюз.

## Шлюз за приложете програмни интерфейси

Във втората и последна тема ще разширим това и ще внесем още едно изискване, а това ще бъде необходимостта от различни клиенти, а това също е за повече от един шлюз. Да започваме.

## Комуникация в реално време

ASP.NET 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" към вашето ASP.NET приложение. Докато чатът често се използва като пример, можете да направите много повече. Всеки път, когато потребител обновява уеб страница, за да види нови данни, или страницата прилага дълга анкета за извличане на нови данни, тя е кандидат за използване на SignalR. Примерите включват табла за управление и приложения за наблюдение, приложения за сътрудничество (като едновременно редактиране на документи), актуализации на напредъка на работата и формуляри в реално време.

Заключение

Научните изследвания са насочени към постижения в науката и създаване на нови знания в дадена област. Научното познание е непрекъснат процес на систематизирано изучаване на обекти и събития. Неговата динамичност се изразява в непрестанни изследвания за постигане на високи резултати и научен прогрес. За този прогрес допринасят множество различни фактори, като добра и целенасочена подготовка, актуалност и практическа приложимост на изследвания проблем и прилагане на подходящи методологии, методики и методи.

Методологията на изследването се отнася до изследователския подход и различните видове дейности за справяне с изследователските предизвикателства. Методологиите предоставят набор от общи и частни изисквания в зависимост от проучваната област. В рамките на методологията могат да се използват различни методики, конкретизиращи начина на прилагане на подходящи изследователски методи. Научните методи, като наблюдение, анализ, интерпретация и обобщаване на резултати, водят до изграждане на научно познание (Wohlin, 2021).

Разгледаните основи на провеждане на научно изследване са от изключителна важност при създаване на научни трудове. Посредством използването на подходящи методологии, методики и методи в научните изследвания води до научен прогрес.

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.
11. ТАСЕВ, Г. (2005) Методични основи на научните изследвания. Как да напиша и защитя дисертацията си? София: Авангард Прима.
12. ХРИСТОВ, Х., ХРИСТОВА, А. (2012) *Модели и гъвкави методи в обучението по компютърни науки*. Наука и технологии. Том 2. с. 59-63.
13. BECK, K. et al. (2001) *Манифест за Agile разработка на софтуер*. [Онлайн] Достъпно на: https://agilemanifesto.org/iso/bg/manifesto.html [Достъпено: 22 януари 2022].
14. BHATTACHERJEE, А. (2012) *Social Science Research: Principles, Methods, and Practices*. University of South Florida.
15. EUROPEAN FORESIGHT PLATFORM – EFP (2022). *Scenario Method* [Online] Available at: http://www.foresight-platform.eu/community/forlearn/how-to-do-foresight/methods/scenario/ [Accessed: 22 January 2022].
16. THOMOPOULOS, N. (2012) *Essentials of Monte Carlo Simulation: Statistical Methods for Building Simulation Models.* Springer Science & Business Media.
17. WOHLIN, C., RUNESON, P. (2021) *Guiding the selection of research methodology in industry–academia collaboration in software engineering*. Information and Software Technology.