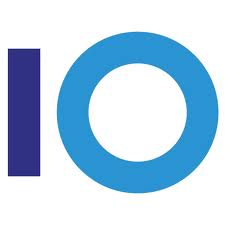
**Универзитет Св. ʻʻКирил и Методијʼʼ – Скопје**

**Факултет за информатички науки и компјутерскo**

**инженерство**

**Дипломска работа**

Анализа на **.NET Core**

Ментор: Кандидат:

Проф. Д-р. Иван Чорбев Војдан Гичаровски

Индекс број: 111040

Скопје, 2017

Содржина

[Апстракт 3](#_Toc473494048)

[Клучни зборови 3](#_Toc473494049)

[Вовед 4](#_Toc473494050)

[Отворен изворен код 5](#_Toc473494051)

[Наменета за повеќе оперативни системи и платформи 6](#_Toc473494052)

[Mono 6](#_Toc473494053)

[Xamarin 7](#_Toc473494054)

[Компоненти 8](#_Toc473494055)

[C# Компајлер – Roslyn 8](#_Toc473494056)

[Алатка за градење софтвер – MSBuild 8](#_Toc473494057)

[Алатки за командна линија – dotnet 9](#_Toc473494058)

[Библиотеки што се користат при извршување на софтвер – CoreCLR 9](#_Toc473494059)

[.NET Стандард 9](#_Toc473494060)

[Библиотеки – CoreFX 9](#_Toc473494061)

[Менаџер на пакети – NuGet 9](#_Toc473494062)

[Виртуелна машина – Docker 9](#_Toc473494063)

[Рамка за развивање веб апликации - ASP.NET MVC 9](#_Toc473494064)

[Kestrel 9](#_Toc473494065)

[Owin 10](#_Toc473494066)

[Интегрирана околина за развој - Visual Studio 2017 RC 10](#_Toc473494067)

[Пример апликации и нивна разлика при развој со .NET Framework 4.6.2 и .NET Core 1.0 10](#_Toc473494068)

# Апстракт

Во оваа дипломска работа се прави анализа на .NET Core рамката изработена од компанијата Microsoft. .NET Core е следната верзија на рамката .NET Framework којашто се користи при изработка на сите типови на апликации, како веб, мобилни и клиентски апликации, со помош на јазикот C#.

Во почетокот на оваа дипломска работа ќе се зборува за историското значење на .NET Core и како тоа влијае на програмерската заедница во целост, но и како влијае конкретно на компанијата Microsoft.

Главна цел е да се опфатат новостите и разликите на .NET Core во однос на неговиот претходник .NET Framework, но исто така и да се објаснат подетално сите негови компоненти и промените кои ги претрпеле со новата верзија на .NET.

## Клучни зборови

.NET Core, .NET Framework, ASP.NET Core, NuGet, Roslyn, .NET Standard, Open source, Cross platform, Docker, OWIN, Kestrel

# Вовед

Денес живееме во дигитално доба каде што компјутерите и мобилните уреди ни помагаат при извршување на секојдневните обврски, ни ги олеснуваат и забрзуваат здодевните задачи како плаќање сметки, пазарење, пишување дипломски задачи, но и ни овозможуваат да се забавуваме и комуницираме со пријателите, можеме да играме игри, да слушаме музика или да гледаме некој интересен филм.

Развитокот на компјутерите не е краток и едноставен, од првиот електричен компјутер со сијалици оспособен за програмирање, до денешните компјутери со микро-транзистори и по неколку гигабајти меморија и огромна процесорска моќ поминати се повеќе од 70 години. Развојот на хардверот и материјалите што се користат за негова изработка, но и самиот процес на изработка напреднале значително за во денешно време да можеме во нашиот џеб да носиме компјутер со поголема процесорска моќ од компјутерот што се користел за лансирање и водење на леталото Аполо 11.

Паралелно како што се развивал хардверот со него се развивал и софтверот којшто го користи. Од првата верзија на Windows развиена во 1985 до денес можеме да забележиме огромен напредок во функционалностите што ги нуди, интеракцијата и начинот на изработка на апликации.

За да може да се изработи една апликација потребни се компајлер за соодветниот јазик кој што се користи за програмирање, кој во тоа време е јазикот Ц, и библиотеки за интеракција со периферните уреди. Во денешно време постојат голем број на јазици за програмирање меѓу кои и јазикот C# којшто е осмислен, изработен и одржуван од страна компанијата Microsoft. Овој јазик сам по себе не може да се користи за да се направи продуктивна апликација, туку мора да се употребува во комбинација со рамката .NET за изработка на апликација способна за извршување и комуникација со периферните уреди.

.NET претставува колекција на библиотеки за менаџмент на меморија, редослед на извршување и преведување на програмскиот код во инструкции за компјутерот на којшто се извршува апликацијата изработена со помош на оваа рамка. Во сржта .NET содржи библиотека наречена CLR (common language runtime) што всушност го преведува напишаниот код во соодветни машински инструкции спремни за извршување. Останатите библиотеки помагаат за интеракција со периферните уреди и ги олеснуваат операциите со различните типови на информации (броеви, букви, реченици, низи).

# Отворен изворен код

Во изминатите години Microsoft беше компанија којашто своите технологии ги изработуваше во тајност, но со новата верзија на .NET и воедно со промената на главниот извршен директор Сатија Надела (анг. Satya Nadella) се реши да го објави изворниот код на технологијата .NET Core, компајлерот и компонентите. Со овој чекор Microsoft придонесе за развој на технологија којашто е најблизу до потребите на развивачите на софтвер коишто ќе ја употребуваат за развој на најразлични апликации.

Кодот е отворен за разгледување и надополнување за сите заинтересирани развивачи, но тие мораат да ги почитуваат правилата поставени од страна тимот на Microsoft што е одговорен за соодветната технологија. Овие правила обично се однесуваат на стилот на пишување код, шаблоните што се употребуваат, стилот на пишување GIT пораки, авторски права на код, процес за поднесување за барање за додавање или модифицирање на код, начин и место за дискусија за проблемите што се треба да се решат, стандард на јавни повици и структури којшто библиотеките мора да го запазат и други правила кои што се специфични за секоја компонента поодделно.

Иако кодот е отворен за модификации стандардот за функционалностите што оваа рамка мора да ги нуди е однапред одреден и напишан како засебна библиотека .NET Standard којашто мора да биде интегрирана, што во суштина значи дека сите рамки што ја интегрираат оваа базична библиотека ќе го имаат истиот јавен интерфејс за своите базични функционалности. Сите останати дополнителни функционалности развиени во рамките на .NET Core ќе бидат дел специфично само од оваа рамка.

Овој стандард го решава проблемот кој се јавува при споделување на библиотеки во различни платформи и рамки. На пример ако еден развивач на софтвер напише библиотека за манипулација на дати и временски зони и таа библиотека е зависна од .NET Standard 1.6 тогаш таа библиотека ќе може да биде искористена во било која апликација што користи рамка (.NET Framework vNext или .NET Core 1.0) што го интегрира соодветниот стандард.

# Наменета за повеќе оперативни системи и платформи

Најзначајната промена на .NET рамката со изработката на верзијата .NET Core е што ќе биде достапна за развој и користење на повеќе платформи и оперативни системи. Во минатото Microsoft се фокусираа да изградат рамка за развој којашто работи ексклузивно за Windows, но сега тие изработуваат рамка којашто ќе може да се извршува на повеќето познати дистрибуции на Linux и на macOS.

Оваа промена има посебно значење за развивачите на софтвер затоа што сега нема да мора да се потпираат на други алатки за програмирање за повеќе платформи одеднаш како Mono, Cordova или Qt, туку ќе можат да ја користат рамката .NET Core.

Исто така оваа промена ги засегнува и клиентите затоа што со оваа промена веб апликациите не мора да бидат хостирани ексклузивно на Windows сервер, туку може и на сервер со Linux оперативен систем доколку има поддршка за .NET Core рамката. Со оглед на тоа што Linux серверите се поевтини, цената на хостирање на ASP .NET апликации би паднала значително на светско ниво.

## Mono

Mono е првиот обид да се произведе платформа со ист јавен стандард како .NET Framework којашто би работела на повеќе различни оперативни системи и уреди како Linux, Solaris, BSD macOS, Android, iOS, Wii, Xbox 360, PlayStation 3. Овој проект е изработен со цел да се олесни развивањето на .NET апликации на Linux оперативниот систем, но и да се даде можност да се развиваат квалитетни апликации со користење на C# јазикот за повеќе платформи одеднаш.

Структурата на Mono платформата е направена според стандардите и шаблонот на .NET Framework коишто ги има објавено Microsoft под стандардите ECMA-334 и ECMA-335, односно развиени се C# компајлер за преведување на C# код во бајт код, CLR (common language runtime) компајлер за статичко, навремено и предвремено компајлирање на бајт код во машиснки код за соодветниот процесор, библиотеки според .NET Standard спецификацијата.

## Xamarin

Xamarin е компанија којашто е основана во 2011 од развивачите на проектот Mono со главна цел да развие платформа и интегрирана работна околина за развивање на софтвер за мобилните оперативни системи Android, iOS и Windows Phone во јазикот C# на било кој оперативен систем.

Овој потфат има повеќе големи значења за софтверската индустрија и за развивачите на софтвер за мобилни уреди. Пред да се развие овој проект развивачите на софтвер беа принудени да развиваат апликации поодделно за секој мобилен оперативен уред што најчесто значи развивање на истата апликација три пати во различни технологии и јазици соодветно за Android, iOS и Windows Phone. Потоа се разви платформата Cordova којшто во суштина апликациите ги прикажува како веб страни, иако овој начин на развивање апликации за повеќе оперативни системи одеднаш заштедува време, најчесто не се практикува затоа што апликациите развиени на овој начин се многу бавни и нивните анимации не се прикажуваат како што треба.

Xamarin работи како обвивка на нативните библиотеки што ги нудат алатките за развивање на трите мобилни платформи што значи дека при компајлирање на напишаниот код во C# тој се преведува во нативен код за соодветната платформа. Со цел за користењето на овој софтвер да биде лесно и за развивачите коишто ги користат оперативните системи Linux и macOS, компанијата Xamarin има развиено интегрирана околина Xamarin Studio за работа и за овие оперативни системи со цел да ја замени околината за развој Visual Studio произведена од Microsoft којашто работи ексклузивно на Windows.

Во февруари 2016 година Microsoft и Xamarin потпишаа договор за откуп со што сега оваа компанија припаѓа на Microsoft. Ова има големо значење за развивачите на софтвер што користат Linux и macOS оперативни системи затоа што Microsoft започна со официјална поддршка на интегрираната околина Xamarin Studio. Со развитокот на .NET Core и неговата одлика дека може да се извршува на повеќе оперативни системи, развивачите на софтвер за прв пат добиваат шанса да можат да развијат веб, клиентски и мобилен софтвер во Xamarin Studio, што наликува со своите функционалности на Visual Studio (но ги нема сите), во јазикот C# на било кој оперативен систем што е поддржан од Xamarin Studio и .NET Core.

# Компоненти

Развивањето на софтвер со помош на рамката .NET Core можеби изгледа како едноставна и лесна работа, но доколку навлеземе подлабоко во компајлирањето и извршувањето на напишаниот код, тестирање и пакување со цел оптимално прикачување на апликацијата на соодветната машина што ќе ја извршува брзо ќе сфатиме дека целата рамка е многу сложена и со цел да се олесни нејзиното одржување и ажурирање таа е поделена во повеќе компоненти.

## C# Компајлер – Roslyn

Како што работата на развивач на софтвер е да ја преведе бизнис логиката побарана од клиентот во соодветен код, така работата на компајлерот е да го преведе тој код во соодветни машински инструкции што може да ги разбере компјутерот, но во овој случај не се случува тоа. Поради тоа што првично програмерите имаат избор да развиваат софтвер со јазиците C# и Visual Basic, решението на Microsoft е овие јазици да се преведат во некој среден јазик (CIL – Common intermediate language) на пониско ниво близу машински код а тој потоа да се изврши со помош на CLR.

.NET Platform Compiler или попознат како Roslyn компајлер е алатката којашто го преведува C# кодот во соодветен CIL код. Со новата верзија на .NET Core доаѓаат и нови функционалности на компајлерот во форма на анализатори на синтакса, семантички анализатори и декомпајлери. Овие новости не се зависни од .NET Core рамката туку од верзијата и синтаксата којашто се поддржува на C# јазикот, но тоа не значи дека веќе постоечките функционалности не може да се подобруваат и да се поправаат доколку има некоја грешка или нелогичност при нивно извршување.

Библиотеките коишто го сочинуваат компајлерот сепак се дел од .NET Core рамката иако не зависат од нејзините функционалности, со тоа се и дел од проектот што го градиме. Во претходните .NET Framework верзии овие библиотеки не беа дел од рамката, туку се додаваа во проектот како NuGet пакети со што се олеснува нивното ажурирање, но во .NET Core рамката тие се дел составен дел и секогаш се користат истите верзии од овие библиотеки во секој проект што ја користи оваа рамка, па со тоа се одржува конзистентност на нивното верзионирање. Најновата верзија на Roslyn компајлерот ја поддржува јазикот C# верзија 6.0.

## Алатка за градење софтвер – MSBuild

MSBuild (Microsoft Build Tools) е алатка за создавање, компајлирање, тестирање, пакување и прикачување на апликации и генерирање на документација. MSBuild е независен од Visual Studio, но Visual Studio го користи MSBuild во позадина за извршување на своите задачи. Тоа значи дека доколку сакаме да изградиме некој продукт можеме да го направиме тоа без воопшто да користиме Visual Studio, туку само алатката MSBuild.

Во првичната алфа верзија на рамката .NET Core се отфрли користењето на конфигурацискаta датотека .csproj со .xproj и project.json, со што се раздели стратегијата за компајлирање на проектот во датотеката .xproj и листата на NuGet пакетите и верзиите на .NET рамката којашто се користи во датотеката project.json. Оваа промена најверојатно нема да трае долго затоа што тимот од Microsoft што ја изработува рамката .NET Core објави дека во верзијата 2.0 ќе се вратат на старата структура на конфигурирање со .csproj датотека.

## Алатки за командна линија – dotnet

## Библиотеки што се користат при извршување на софтвер – CoreCLR

## .NET Стандард

## Библиотеки – CoreFX

## Менаџер на пакети – NuGet

## Виртуелна машина – Docker

## Рамка за развивање веб апликации - ASP.NET MVC

### Kestrel

### Owin

# Интегрирана околина за развој - Visual Studio 2017 RC

# Пример апликации и нивна разлика при развој со .NET Framework 4.6.2 и .NET Core 1.0