LATVIJAS UNIVERSITĀTE

DATORIKAS FAKULTĀTE

**WEBAPPOS .NET IZSAUKUMU ADAPTERIS UN C# STARPKODA ĢENERATORS**

KVALIFIKĀCIJAS DARBS

Autors:

Artis Pauniņš ap19122

Darba vadītājs: Sergejs Kozlovičs

RĪGA 2020

# ANOTĀCIJA

# ABSTRACT

# SATURA RĀDĪTĀJS

# 

# IEVADS

**Nolūks**

Kvalifikācijas darba nolūks ir aprakstīt programmatūras prasības izstrādājamajam adapterim un starpkoda ģeneratoram. Dokumentā tiks detalizēti aprakstītas sistēmas funkcijas un ierobežojumi.

Dokuments paredzēts visām ieinteresētajām pusēm, tostarp izstrādātājiem.

**Darbības sfēra**

WebAppOS operētājsistēmas izsaukumu adapteri ir paredzēti, lai tie spētu izsaukt operētājsistēmas funkcijas. Adapterus veido atsevišķi katrai programmēšanas valodai, tādējādi šie adapteri dos iespēju operētājsistēmas izstrādātājiem izstrādāt jaunas un esošās funkcijas vairākās programmēšanas valodās.

**Saistība ar citiem dokumentiem**

Dokumenta noformējumā ievērotas standarta LVS 68:1996 “Programmatūras prasību specifikācijas ceļvedis” vadlīnijas.

**Pārskats**

# APZĪMĒJUMU SARAKSTS

WebAppOS – operētājsistēma kura ir veidota palaišanai tīmeklī.

gRPC framework - moderna atvērtā koda augstas veiktspējas attālās procedūras izsaukuma sistēma, kas var darboties jebkurā vidē.

Atvērts kods jeb atvērta programmatūra – programmatūra, kas tiek izlaista saskaņā ar licenci, kurā autortiesību īpašnieks lietotājiem piešķir tiesības izmantot, pētīt, mainīt un izplatīt programmatūru un tās pirmkodu jebkuram nolūkam.

Tīmekļa procesors -

# 1. VISPĀRĪGS APRAKSTS

## 1.1 Esošā stāvokļa apraksts

Kā jebkurai programmai un operētājsistēmai, arī WebAppOS ir savas funkcijas. Tā kā WebAppOS ir izstrādāts valodā JAVA, tad arī funkcijas ir rakstītas valodā JAVA. Tas ierobežo iespējas izstrādāt operētājsistēmas funkcijas citās programmēšanas valodās.

WebAppOS ir arī funkcijas, kuras ir rakstītas valodā lua. Lai operētājsistēma varētu palaist šīs funkcijas, tad tīmekļa procesorā ir adapteris, kurš ir veidots kādai konkrētai programmēšanas valodai un šie adapteri spēj atrast un izsaukt funkcijas konkrētā programmēšanas valodā. Izņemot JAVA un lua valodas, nav pieejami adapterī citām programmēšanas valodām, piemēram, C# vai Python.

Uzdevums ir izstrādāt tīmekļa procesoram adapteri, kas ļautu operētājsistēmai izsaukt funkcijas C# programmēšanas valodā.

## 1.2 Pasūtītājs

Sistēmas pasūtītājs ir Latvijas Universitātes Matemātikas un informātikas institūts.

## 1.3 Produkta perspektīva

WebAppOS apraksts

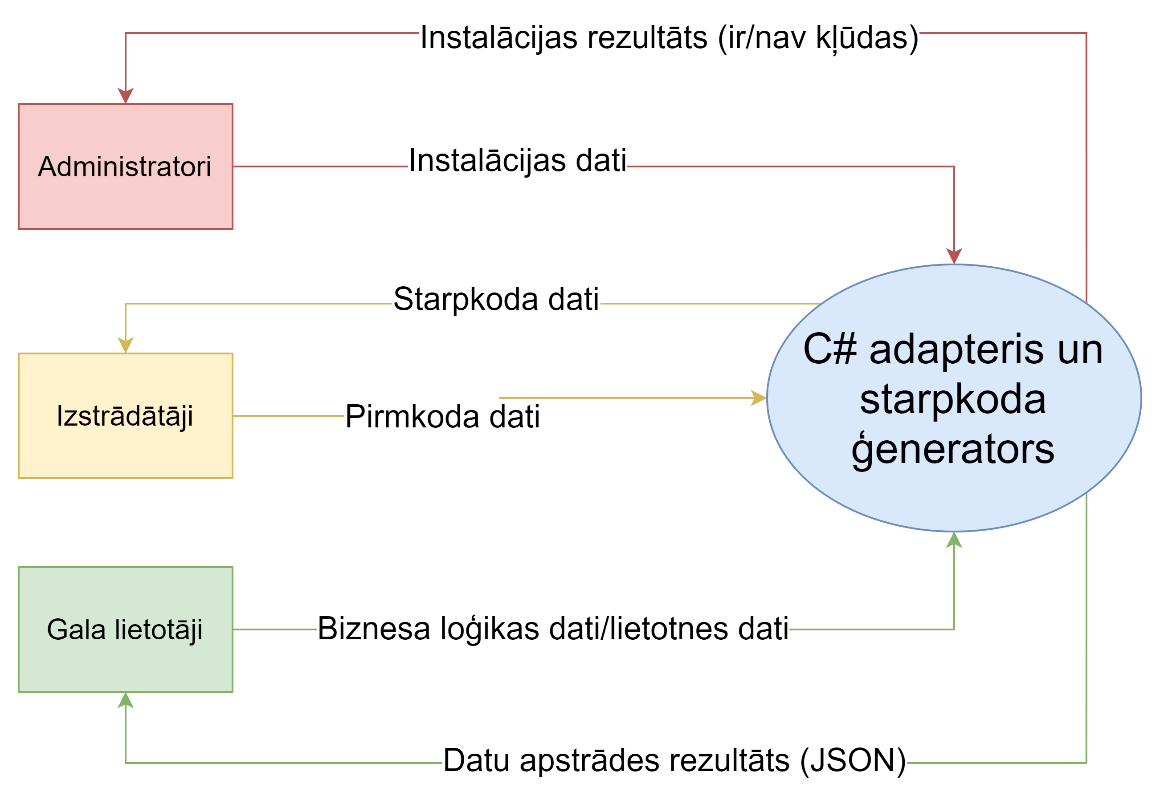
* Galvenās funkcijas
  + WebAppOS nodrošina iespēju izsaukt kodu, kas ir rakstīts dažādās programmēšanas valodās un kas atrodas dažādos tīkla mezglos
  + WebAppOS nodrošina tīmekļa atmiņas sinhronizāciju starp klientu un serveri.
  + WebAppOS ļauj veidot tīmekļa lietotnes tādā pašā veidā, ka veido darbvirsmas lietotnes, nedomājot par tīklu.
* Saskarnes
  + RAAPI – ļauj piekļūt tīmekļa atmiņai {https://webappos.org/dev/raapi/}
  + IWebCalls API – ļauj izsaukt funkcijas, kuras rakstītas citās programmēsanas valodās vai kas atrodas dažādos tīkla mezglos
    - Local – virzienā no WebAppOS uz .NET
    - Remote – virzienā no .NET uz WebAppOS
* Nepieciešams tīkla savienojums ar WebAppOS serveri un vēlamā operētājsistēma ir Linux Ubuntu

Adapteris un kodu ģenerators tiks iekļauti WebAppOS distribūcijā.

## 1.4 Sistēmas lietotāji

Sistēmu izmantos trīs lietotāju grupas:

* Administratori – administratoru uzdevums būs WebAppOS uzstādīšana.
* Izstrādātāji – izstrādātāji veidos programmatūru WebAppOS videi izmantojot izstrādāto adapteri un kodu ģeneratoru
* Gala lietotāji – gala lietotāji izmantos WebAppOS biznesa vajadzībām.



*1.1. att.* **0. līmeņa DPD diagramma**

## 1.5 Darījumprasības

* Dinamiski palaist funkcijas
* LINQ atbalsts
* Programmkodu ģenerēšana
* Nosaukumu telpu (namespace) atbalsts kodu ģenerēšanai
* Rezervētie vārdi
  + class – klašu definēšanai
  + association – asociāciju definēšanai
  + Integer – veselo skaitļu definēšanai
  + String – simbolu virkņu definēšanai
  + Boolean – Būla vērtību glabāšanai
  + Real – reālo skaitļu glabāšanai
  + URL – url definēšanai
  + private – privāto lauku glabāšanai
  + public – publisko lauku glabāšanai
  + BaseObject – klase “BaseObject”, kas ir virsklase visām definētajām klasēm
* Klašu kļūdu pārbaude
  + Vai eksistē virsklase, no kuras grib mantot
  + Virsklase nevar būt pamatklase
  + Katrai klasei ir viena virsklase
  + Visām klasēm ir unikāli vārdi
  + Klašu definīcijai izmanto atslēgvārdu “class”
  + Klašu nosaukumi nevar būt rezervētie vārdi
* Lauku (mainīgo un funkciju) kļūdu pārbaude
  + Laukam ir datu tips “Integer”, “String”, “Boolean” vai “Real”
  + Lauku vārdi ir unikāli, kā arī nedrīkst sakrist ar klases vārdu, bet drīkst sakrist ar virsklases vārdu.
  + Lauku vārdi sākas ar burtu vai apakš svītru un sastāv no burtiem, cipariem un apakš svītrām
  + Lauku vārdi nevar būt rezervētie vārdi
  + Lauki ir atdalīti ar semikolu
  + Funkciju argumentu vārdi drīkst sakrist ar lauku vārdiem, kā arī ar klases vārdu
  + Funkciju arguments sastāv no datu tipa un vārda, un visi argumenti ir atdalīti ar komatu
  + Funkciju argumentu vārdi savā starpā ir unikāli
* Kļūdu pārbaude starp klases un tās virsklašu laukiem
  + Ja apakšklasē un virsklasē ir funkcija ar vienu un to pašu vārdu, tad to drīkst pieļaut, ja funkcijām ir identiski datu tipi un argumenti. URL drīkst būt dažādi
* Asociāciju kļūdu pārbaude
  + Asociācijā starp divām klasēm ir jābūt dažādiem lomu vārdiem skatoties gan no avota klases, gan no mērķa klases, ņemot vērā arī abu klašu virsklases un apakšklases.
* Ja ģenerējamajā pirmkodā tiek konstatēta kaut viena kļūda, tad kods netiek ģenerēts.

## 1.6 Vispārējie ierobežojumi

* Dažādi OS (Windows 10, Windows 11, Linux Ubuntu, MacOS 10, MacOS 11 )
* Dažādu procesoru arhitektūras (AMD64, ARM64)
* Java 11 (WebAppOS) un .NET 5.0 (Adapterim un kodu ģeneratoram)

## 1.7 Pieņēmumi un atkarības

* Izmanto trešo pušu kodu ģeneratorus ar atvērtu pirmkoda licenci
* Uzģenerētajam kodam jādarbojas iepriekš minētajās operētājsistēmās
* Izstrādātājam jāspēj iemācīties pašveidotā programmēšanas valoda
* Izstrādātāji var palaist programmas, izmantojot komandrindu
* Kods pašveidotajā programmēšanas valodā ir sintaktiski pareizs, piemēram, funkcijām nedrīkst likt divus vārdus, bet drīkst pieļaut gadījumus, kad kaut kas nav ievadīts, piemēram, funkcijai nav norādīts ne vārds, ne datu tips. Tādā gadījumā kompilatoram ir jāfiksē to kā kļūdu.

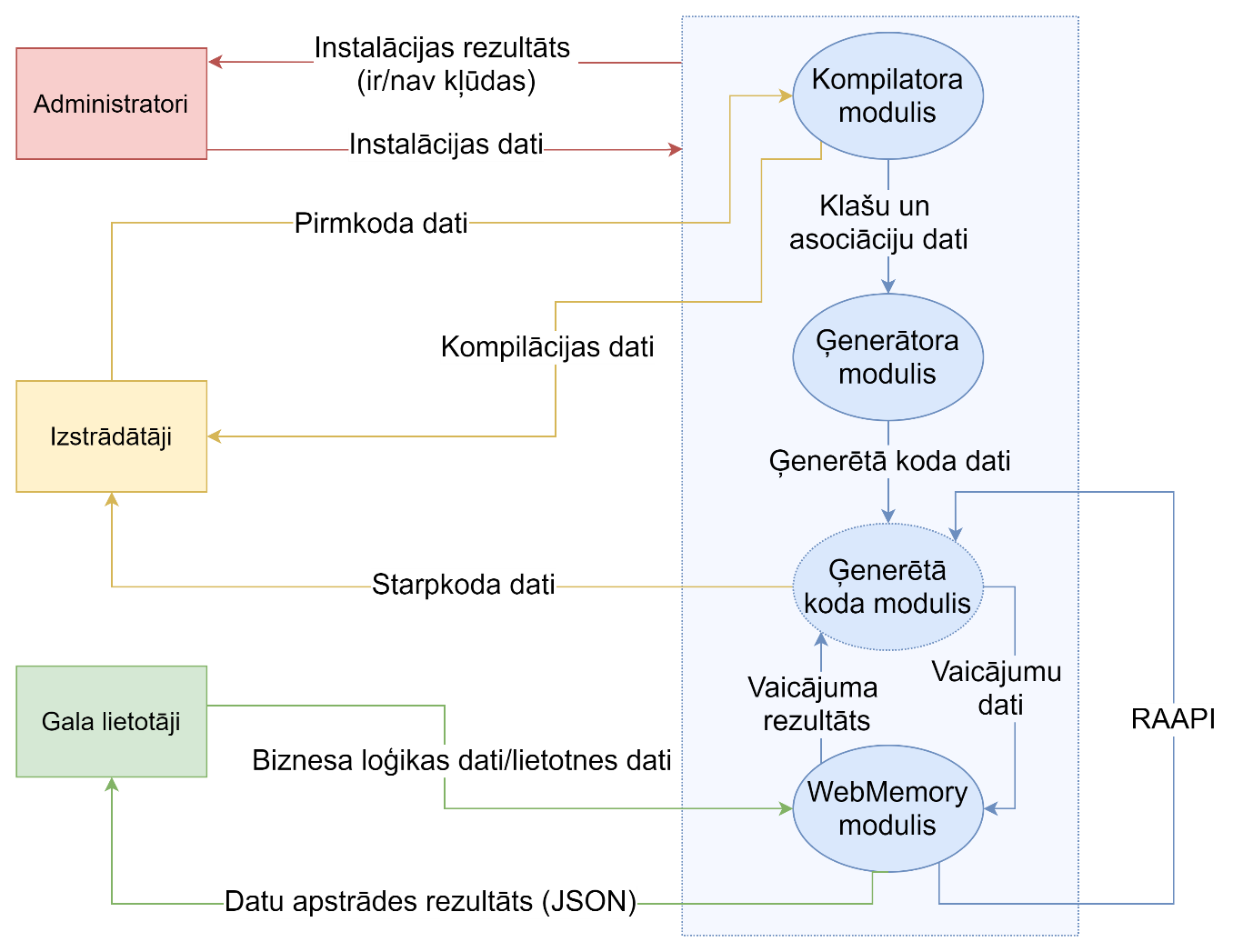
# 2. PROGRAMMATŪRAS PRASĪBU SPECIFIKĀCIJA

## 2.2 Funkcionālās prasības

### 2.2.1 Funkciju sadalījums pa moduļiem/komponentiem

Sistēma sastāv no 4 moduļiem:

* WebMemory modulis – apstrādā klases, objektus, atribūtus un asociācijas, kuras glabājas šajā modulī.
* Ģenerētā koda modulis – uzģenerētā pakotne ar C# klasēm, kas savieno C# kodu ar WebMemory moduli. Tajā būs klases, kuras atbilst datiem, kas atrodas WebMemory modulī.
* Kompilatora modulis – atbild par pirmkoda apstrādi jeb kļūdu atrašanu pirmkodā un pirmkoda datu iegūšanu (klases un to īpašības, metodes, asociācijas).
* Ģeneratora modulis – atbild par pirmkoda ģenerēšanu C# programmēšanas valodā, izmantojot datus, kas iegūti no kompilatora moduļa.

*2.1. att.* **1. līmeņa DPD diagramma**

### 2.2.2 Kodu ģenerēšanas modulis

### 2.2.3 Ģenerētā koda modulis

### 2.2.4 WebMemory modulis

## 2.3 Nefunkcionālās prasības

# 3. PROGRAMMATŪRAS PROJEKTĒJUMA APRAKSTS