**VibeCheck**

**ALEKSANDRA POPOV RA106/2022**

**OGNJEN LIPTAK RA197/2022**

**BENCE-MATE FERČIK RA177/2020**

Table of Contents

[Opis sistema 3](#_Toc207810022)

[Komponenti sistema 3](#_Toc207810023)

[Komunikacija i protokoli 5](#_Toc207810024)

[Procedura pridruživanja 5](#_Toc207810025)

[Kada se uređaj (senzor ili alarm) pokrene 5](#_Toc207810026)

[Kada se kontroler (broker) pokrene 5](#_Toc207810027)

[Kada se korisnička aplikacija pokrene 5](#_Toc207810028)

[Razmena podataka (MQTT) 6](#_Toc207810029)

[Održavanje veze 6](#_Toc207810030)

[Provera živosti brokera (od strane klijenta) 6](#_Toc207810031)

[Provera živosti klijenta (od strane brokera) 6](#_Toc207810032)

[Eksplicitno diskonektovanje 6](#_Toc207810033)

[Lista MQTT tema 7](#_Toc207810034)

[JSON sadržaj 7](#_Toc207810035)

[Plan testiranja 9](#_Toc207810036)

[Povezivost 9](#_Toc207810037)

[Predefinisana Logika Pragova Senzora 9](#_Toc207810038)

[MEMS Akcelerometar (m/s²) 9](#_Toc207810039)

[Tilt Senzor (stepeni) 9](#_Toc207810040)

[Testovi uređaja 9](#_Toc207810041)

[Testovi Aplikacije 10](#_Toc207810042)

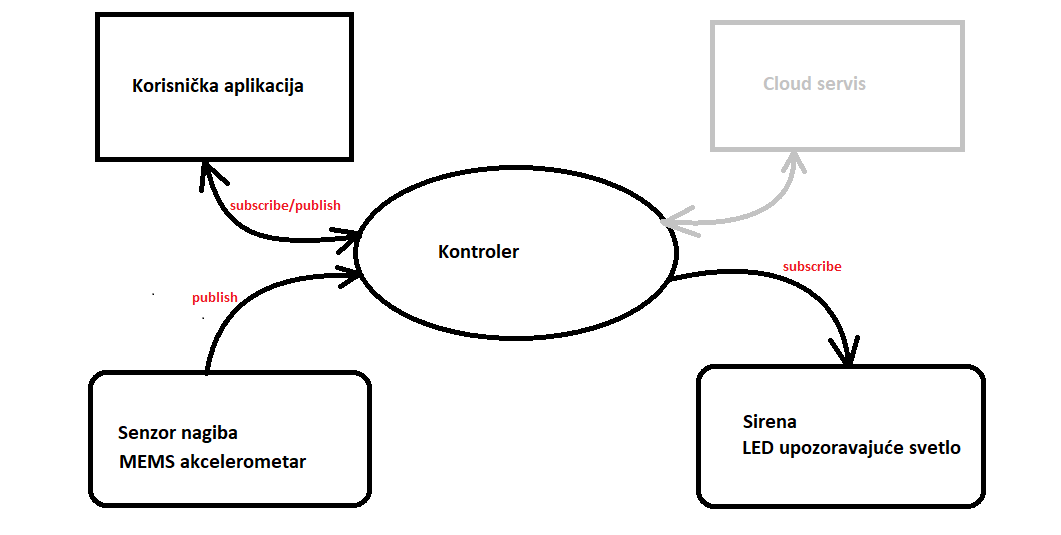
VibeCheck

# Opis sistema

Sistem „VibeCheck” koristi se za merenje vibracija (skraćeno „vibe”) i nagiba tla, posebno kod građevinskih konstrukcija, i aktivira alarme ako nivoi postanu nestabilni.

Ove vibracije mogu izazvati oštećenja struktura kod starijih ili osetljivih objekata, a neke gradnje imaju ograničenja u pogledu vibracija u blizini susednih objekata ili javnih površina. Nagibajuća zgrada je uvek loša vest.

## Komponenti sistema



slika

Kao što se vidi na slici 1, sistem se sastoji od sledećih elemenata:

* MEMS(Micro-Electric-Mechanical-System) akcelerometar:

Meri vibracije na -teoretski-100HZ frekvencijama.  
Jedinica: mm/s.  
Publish-uje podatke sa senzora putem MQTT-a.

* Senzor nagiba

Meri nagib po X i Y osi na niskim frekvencijama.(100mHz).  
Jedinica: °.  
Publish-uje podatke sa senzora putem MQTT-a.

* Sirena:

Aktuator koji se može aktivirati na daljinu.  
Pretplaćuje se na MQTT temu kako bi primao komande za aktivaciju.

* ISPREKIDAN ZVUK: Prekoračen je warning prag **nagiba** ili **vibracija**. Ovo ukazuje na potencijalno pomeranje konstrukcije.
* NEPREKIDAN ZVUK: Prekoračen je alert prag **nagiba** ili **vibracija**. Ovo ukazuje na jake, tekuće vibracije. To je hitnije upozorenje.
* LED upozoravajuće svetlo

Aktuator koji se može aktivirati na daljinu.  
Pretplaćuje se na MQTT temu kako bi primao komande za aktivaciju.

* PORO TREPERENJE: Prekoračen je **warning** prag **nagiba** ili **vibracija**. Ovo ukazuje na potencijalno pomeranje konstrukcije..
* BRZO TREPERENJE: Prekoračen je **alert** prag **nagiba** ili **vibracija**. Ovo ukazuje na jake, tekuće vibracije. To je hitnije upozorenje.
* Korisnička aplikacija

Prikazuje trenutne nivoe vibracija, nagiba i status alarma.  
Omogućava korisniku da ručno postavi nove pragove za vibracije i nagib.  
Objavljuje nove vrednosti pragova i može ručno aktivirati alarm.

* Cloud servis, Mock server

Mock server simulira server na kome se čuvaju informacije o uređajima. On vraća osnovni JSON fajl za novi uređaj koji se povezao.  
Cloud servis je još predviđen za slanje upozorenja korisniku (npr. putem email-a). Neće biti implementiran u ovom projektu.

* Kontroler (broker)

Kontroler je prvenstveno kontrolna jedinica i mozak sistema, ali prosleđivanje informacija je ključni deo njegovog posla.  
Otkriva uređaje putem SSDP-a, preuzima početne informacije o uređaju sa mock servera, primi podatke senzora, prosleđuje podatke i status sistema klijentu, primenjuje kontrolnu logiku (npr. proverava da li vrednosti prelaze pragove) i objavljuje komande aktuatorima.

Apsolutna vrednost nagiba se izračunava na sledeći način:

Postoje 2 praga za obe vrednosti(vibracija i nagib):

* Upozorenje (Warn): Prekoračen je niži prag, obaveštava se aplikacija i uključuju se aktuatori u warning režimu.
* Alarm (Alert): Prekoračen je viši prag, obaveštava se aplikacija i uključuju se aktuatori u alert režimu.

# Komunikacija i protokoli

Sistem koristi dva glavna protokola:

Device Discovery: SSDP (UDP) multicast na adresi 239.255.255.250:1900

Data Exchange: MQTT(mosquitto broker) pomoću kontrolera

Device info: HTTP/1.1 na mock server 127.0.0.1:8080

## Procedura pridruživanja

### Kada se uređaj (senzor ili alarm) pokrene

* Otvori UDP socket i pošalje SSDP NOTIFY-ssdp:alive putem multicasta.   
  Poruka sadrži:

NT: ssdp:projekat // Identifikacija, da ne dobijamo nasumične poruke sa mreže, već samo od projektovanog sistema  
NTP: Tip poruke //ssdp:alive, ssdp:byebye  
USN: Identifikator uređaja.  
LOCATION: URL na mock server za dobijanje detaljnog opisa uređaja (**JSON** format)

* Osluškuje na svom UDP socketu, spreman da odgovori na M-SEARCH zahteve od kontrolera.
* Inicijalizuje libmosquitto
* U slučaju da je uređaj pokrenuo pre Mosquitto brokera, periodično šalje zahtev za ponovno povezivanje.

### Kada se kontroler pokrene

* Inicijalizuje libmosquitto
* U slučaju da je kontroler pokrenuo pre Mosquitto brokera, periodično šalje zahtev za ponovno povezivanje
* Otvori UDP socket i počne da očekuje NOTIFY poruke na multicast adresu 239.255.255.250:1900
* Pošalje SSDP M-SEARCH poruku putem multicasta kako bi "probudio" sve uređaje na mreži i natirao ih da odmah pošalju RESPONSE. Ovo je korisno ako se kontroler pokrene posle uređaja.
* Na osnovu LOCATION polja iz NOTIFY ili RESPONSE poruke, kontroler preuzima osnovne parametre sa mock servera i beleži ih kao registrovane uređaje na mreži.
* Ako je kontroler uspešno registrovao uređaj, tada objavljuje poruku da uređaj postavi stanje svog ContactService-a na T.

### Kada se korisnička aplikacija pokrene

* Šalje SSDP alive poruku na
* Inicijalizuje libmosquitto
* Odmah se pretplaćuje na relevantne teme.
* U slučaju da je korisnička aplikacija pokrenuo pre Mosquitto brokera, periodično šalje zahtev za ponovno povezivanje

## Razmena podataka (MQTT, upravlja se mosquitto)

Odvije se po sledećim koracima:

* mosq = mosquitto\_new(NULL, true, NULL)   
  mosquitto\_connect\_callback\_set (mosq, on\_connect/message/publish/disconnect); – inicializuje se libmosquitto
* mosquitto\_connect(mosq, "127.0.0.1", “localhost”, keepalive\_interval); – konektuje se, i odmah posle:
* mosquitto\_subscribe(mosq, NULL, "topic", 0); subscribeuje se
* mosquitto\_publish(mosq, NULL, "topic", json\_strlen, json\_string, 0, false); - počne da objavljuje podatke

## Održavanje veze

### Provera živosti mosquitto brokera i klijenta

Podešen je keepalive\_interval parametar u mosquitto\_connect().

* Ako klijent nije dobio poruku od mosquitto brokera za nekoliko vreme, automatski šalje PINGREQ poruke i čeka na PINGRESP od strane mosquitto brokera. Ako ne dobije odgovor, pozvaće on\_disconnect callback. //BrokerTimeout
* Slično, ako mosquitto broker nije dobio poruku ili PINGREQ od klijenta, smatra se de je klijent prekidao veze //ClientTimeout

### Provera živosti uređaja (od strane kontrolera)

Kontroler je pretplaćen na temu VibeCheck/+/disconnected. Kada je klijent diskonektovan, LWT funkcija on\_disconnect objavljuje poruku u toj temi I kontroler uklanja ga sa interne liste aktivnih uređaja.

### Eksplicitno diskonektovanje

Ako je klijent želeo da se diskonektuje:

* Klijent se poziva mosquitto\_disconnect()
* Posle toga, klijent odmah šalje NOTIFY SSDP:byebye multicast poruku u mrežu
* Kontroler uklanja ga iz svoje liste, ali zna da je ispravno diskonektovao

# Lista MQTT tema

|  |  |  |  |
| --- | --- | --- | --- |
| **Tema** | **Publisher** | **Subscriber** | **Opis** |
| VibeCheck/sensors/vibration | MEMS senzor | Kontroler | Mere vibracije |
| VibeCheck/sensors/tilt | Senzor nagiba | Kontroler | Mere nagiba |
| VibeCheck/actuators/LED | Kontroler | Led svetlo | Alarm nagiba ili vibracije |
| VibeCheck/actuators/audio | Kontroler | Sirena | Alarm nagiba |
| VibeCheck/threshold/change | Aplikacija | Kontroler | Promena praga |
| VibeCheck/control/status | Kontroler | Aplikacija | Podaci od sensore+status aktuatora |
| VibeCheck/+/disconnected | Svi uređaji LWT | Kontroler | U slučaju da je neki uređaj “nasilno” diskonektovan |
| VibeCheck/+/connected | Kontroler | Svi uređaji | Кontroler је uspešno registrovao uređaj |

## JSON

|  |  |
| --- | --- |
| **Tema** | **JSON format** |
| VibeCheck/sensors/vibration | {  “id”: “vibration\_sensor\_1”,  “group”: “sensor”,,  “ContactService”:  {  “State”: “T/F”  },  “VibrationService”:  {  “Vibration“: “0~ 29.43” // 3g  }  } |
| VibeCheck/sensors/tilt | {  “id“:“sensor\_tilt\_1“,  “group“:“sensor“,  “ContactService”:  {  “State”: “T/F”  },  “TiltService”:  {  “AngleX“: “0~15”,  “AngleY”: “0~15”  }  } |
| VibeCheck/actuators/LED | {  “id”: “Warning\_LED\_1”,  “group”: “actuator”,  “ContactService”:  {  “State”: “T/F”  },  “WarnService”:  {  “Blink”:”OFF/SLOW/FAST”  {  } |
| VibeCheck/actuators/audio | {  “id”: “Warning\_audio\_1”,  “group”: “actuator”,  “ContactService”:  {  “State”: “T/F”  },  “WarnService”:  {  “Siren”:”OFF/INTERMITTENT/STEADY”  {  } |
| VibeCheck/threshold/change | {  “VibrationService”:  {  “WarnThreshold”: “0~29.43”,  “AlertThreshold: “0~29.43”  },  “TiltService”:  {  “WarnThreshold”: “0~15”,  “AlertThreshold”: “0~15”  }  } |
| VibeCheck/control/status | {  “VibrationService”:  {  “Vibration”: “0~29.43”,  “VibrationStatus”: “NORMAL/WARNING/ALERT/OOB”  },  “TiltService”:  {  “Tilt”: “0~15”,  “TiltStatus”: “NORMAL/WARNING/ALERT/OOB”  }  } |
| VibeCheck/+/disconnected | {  “id”:” <client\_id>”,  “group”:”<client\_group>”,  “Stanje”:”OFF”  } |
| VibeCheck/+/connected | {  “id”:” <client\_id>”,  “group”:”<client\_group>”,  “Stanje”:”ON”//problem if off  } |

# Plan testiranja

## Povezivost

|  |  |  |
| --- | --- | --- |
| **Test** | **Okidač** | **Očekivani Rezultat** |
| Kasno otkrivanja uređaja | |  | | --- | | Pokretanje uređaja, zatim kontrolera |  |  | | --- | |  | | Kontroler otkriva postojeći uređaj preko SSDP-a MSEARCH |
| Detekcija nedostupnog brokera | Zaustavljanje brokera | Uređaj detektuje gubitak brokera preko propuštenog PINGRESP-a |
| Ponovno povezivanje brokera | Restart brokera | Uređaj se automatski povezuje i ponovo registruje |
| Ispravno gašenje | Kucanje q | Uređaj šalje SSDP byebye, kontroler uklanja ga iz svoje liste |
| Neispravno gašenje | CTRL+C ili exit terminal | Broker aktivira LWT, uklanja ga iz svoje liste |
| Skalabilnost | Povezivanje više uređaja | Broker rukuje sa više istovremenih konekcija |

## Predefinisana Logika Pragova Senzora (unutar kontrolera)

### MEMS Akcelerometar (mm/s)

|  |  |  |
| --- | --- | --- |
| **Opseg** | **Stanje Sistema** | **Stanje aktuatori** |
| 0-10 | Normalno | - |
| 10-20 | Warning na vibracije | BlinkSlow i IntermittentSiren |
| 20-29.43 | Alarm vibracija | BlinkFast i SteadySiren |
| 30+ | Alarm- van granica | BlinkFast i SteadySiren |

### Tilt Senzor (stepeni)

|  |  |  |
| --- | --- | --- |
| **Opseg** | **Stanje Sistema** | **Stanje aktuatori** |
| 0-0.25 | Normalno | - |
| 0.25-0.5 | Warning na vibracije | BlinkSlow i IntermittentSiren |
| 0.5+ | Alarm vibracija | BlinkFast i SteadySiren |
| 15+ | Alarm- van granica | BlinkFast i SteadySiren |

## Testovi uređaja

* Simulacija senzora: Ručni unos vrednosti za sve opsege
* Validacija pragova: Provera da je warning < alarm u konfiguraciji
* Odgovor aktuatora: Provera da se tačno aktiviraju odgovarajući aktuatori za svako stanje
* Testiranje granica: Testiranje graničnih vrednosti (29.43, 30, -1 i 0.49, 0.5, 15, -0,2)

## Testovi Aplikacije

* Konfiguracija pragova: Postavljanje novih validnih/nevalidnih vrednosti preko UI
* Ručna kontrola aktuatora: Direktno uključivanje/isključivanje sirene i LED-a
* Vizualizacija stanja: UI prikazuje tačan status Sistema
* Rukovanje van granica: Sistem pravilno označava Out of Bounds vrednosti senzora na UI