Tehnologije

Izzivi

Kako komunicirajo naprave?

- Kako prepoznati s katerimi napravami želimo komunicirat?
- Kako določimo format/vsebino komunikacije?
- Kako vemo, da je sporočilo bilo sprejeto?
- Kako bi to implementirali?

Ali obstaja kakšen standard?

Motivacija: https://flespi.com/blog/http-vs-mqtt-performance-tests

https://cedalo.com/blog/mqtt-connection-beginners-guide/

Izbrani protokoli/storitve naprav

- MQTT
  - Odprtokodni standard, ki je nastal pod okriljem podjetja IBM.
- SmartThings
  - Samsungova rešitev za IoT ekosistem
- Home kit
  - Applova rešitev (protokol), za povezovanje naprav.

MQTT ozadje

Protokol MQTT (Message Queuing Telemetry Transport) je namenjen izmenjavi sporočil med napravami (M2M - machine to machine).

Temelji na načtovalskem vzorcu objavi-naroči (angl. publish/subscribe). Za izmenjavo sporočil uporablja protokol TCP/IP, s tem omogoča povezovanje oddaljenih lokacij.

V letu 2019 sprejet MQTT različica 5 standarda. Uradne dokumente najdete tukaj in tukaj.

OASIS - Organization for the Advancement of Structured Information Standards

- OASIS je neprofitni konzorcij ki spodbuja razvoj, **približevanje in sprejetje odprtih standardov**, ter odprte kode za globalno informacijsko družbo.
- Podobno kot organizacija W3C, ki skrbi za osnovne standarde, OASIS skrbi bolj za višje nivojske standarde.
- Konzorcij sestavlja več kot 5000 udeležencev iz več kot 600 organizacij in posameznih članov iz več kot 65 držav.
    - OASIS se odlikuje po preglednem upravljanju in operativnih postopkih. Člani sami določijo tehnični program OASIS, pri čemer uporabljajo lahek proces, ki je izrecno zasnovan za spodbujanje industrijskega soglasja in združevanje različnih prizadevanj.
- Znani standardi: OpenDocument (temelji na Open Office), SAML (Security Assertion Markup Language),...

https://www.oasis-open.org/

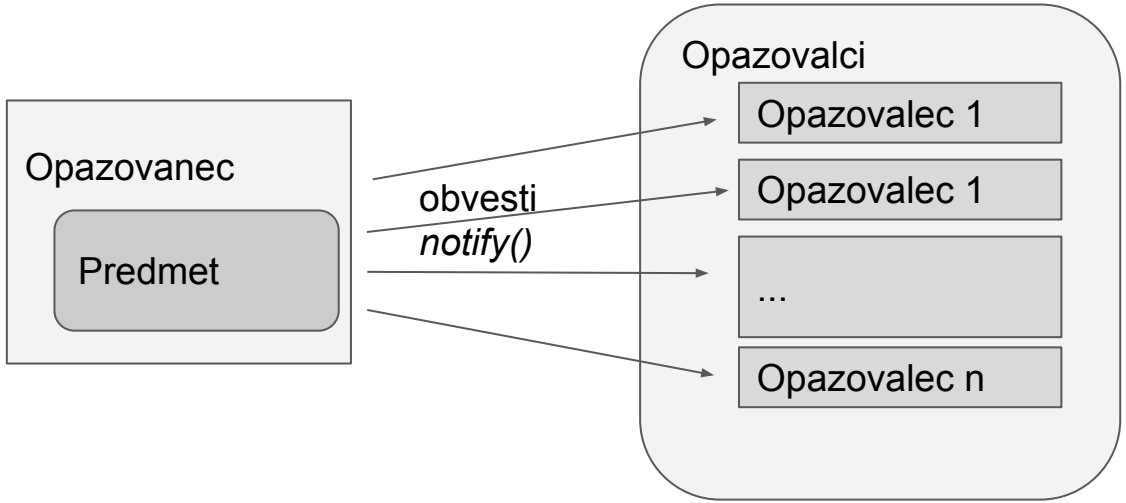
Objavi-naroči - načrtovalski vzorec

Po namenu je podoben načrtovalskemu vzorcu Opazovalec (observer). Glavni cilj načrtovalskega vzorca je komunikacija med objekti po principu Izdajatelj (Publisher), ki ustvari sporočilo (event) in ga posreduje (objavi) naročnikom (Subscribers) sporočila (naročeni).

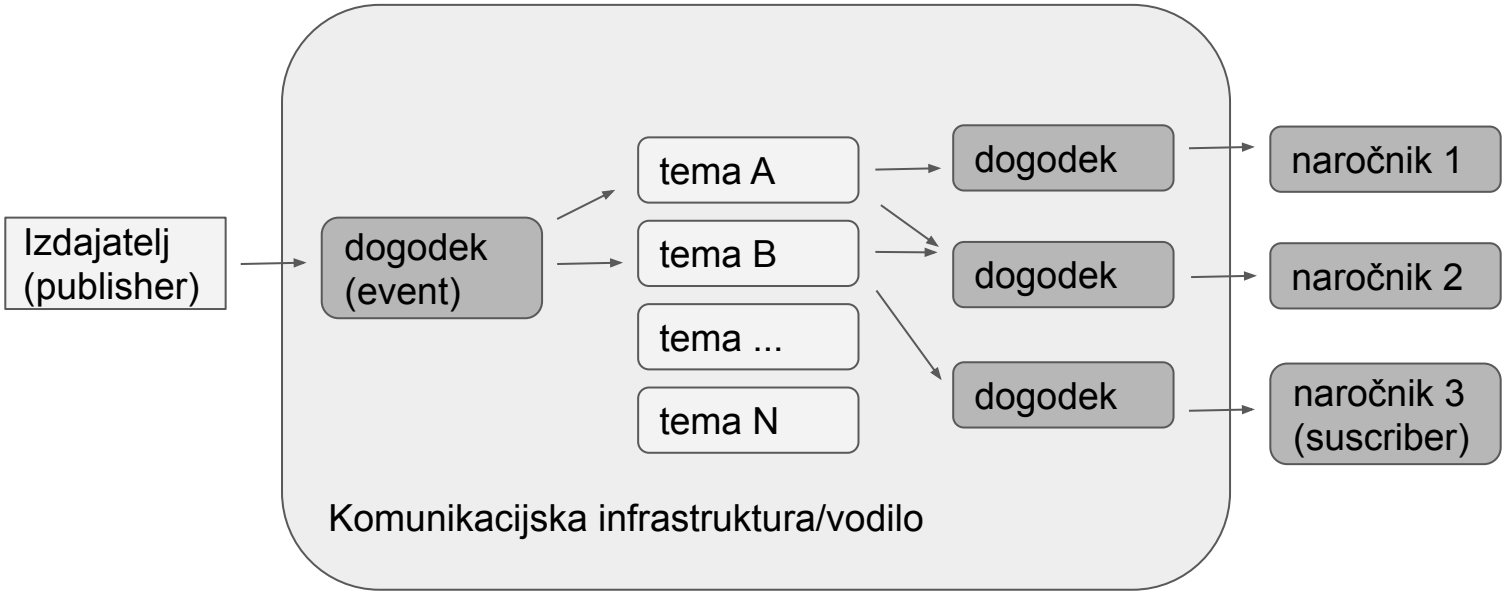
Sporočilo se pošilja med objekti **s pomočjo posrednika** (broker ali message broker). Kadar komunikacija poteka znotraj računalnika, za programsko infrastrukturo po kateri se pošiljajo sporočila pogosto srečamo izraz vodilo sporočil (event bus).

Leta 2014 postane verzija MQTT Version 3.1.1 OASIS Standard

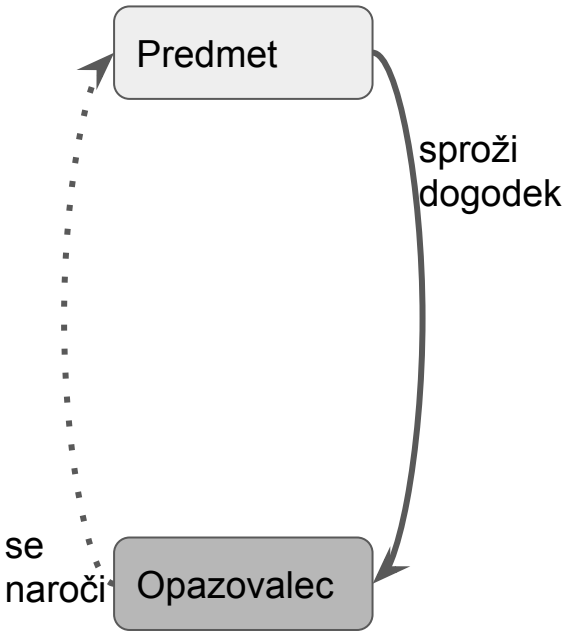
Načrtovalski vzorec Opazovalec



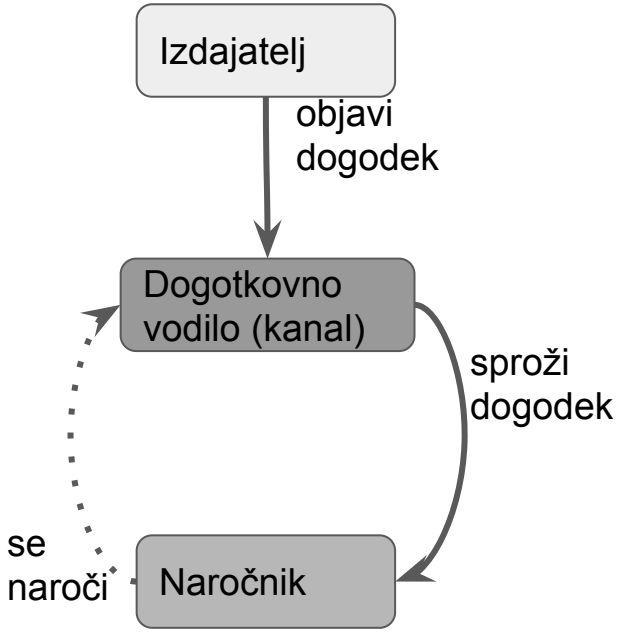
Načrtovalski vzorec Objavi/naroči



Primerjava



Opazovalec



Izdajatelj
naročnik

Primerjava načrtovalskih vzorcev

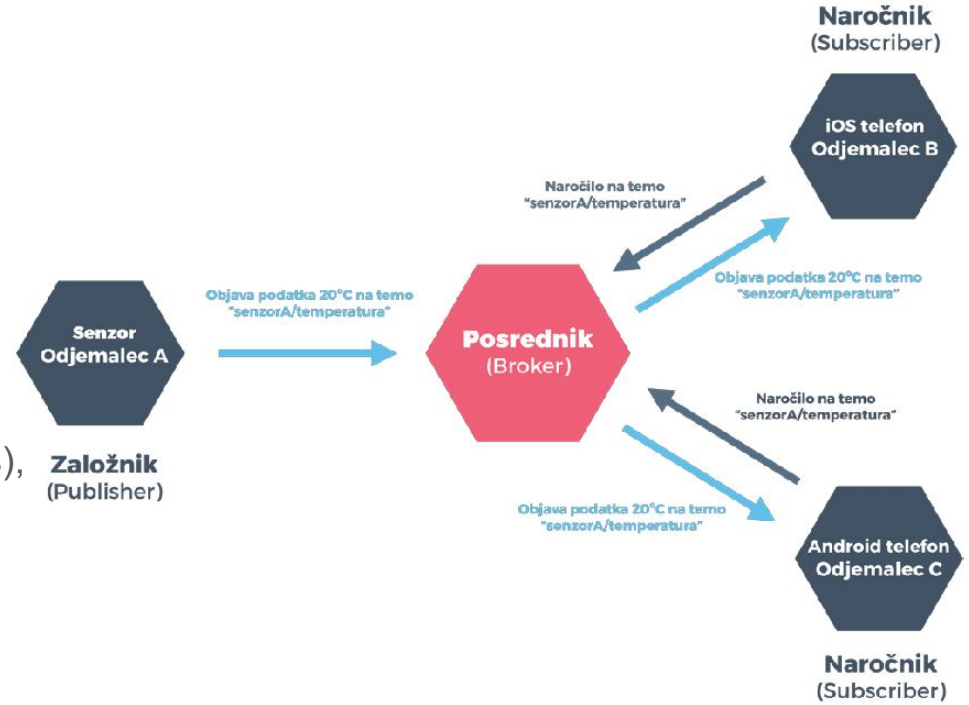
Izdajatelj/naročnik vzorec omogoča še večjo stopnjo šibke sklopljenosti med objekti (loose coupling) od vzorca Opazovalec, saj ni vnaprej določenega dogovora (npr. vmesnika).

Vzorec Opazovalec je implementiran sinhrono, saj se takoj ob obvestilu obvestijo (notify()) vsi opazovalci. Vzorec Izdajatelj/naročnik se implementira asinhrono, s pomočjo vrst sporočil ali posrednikov.

Vzorec Izajatelj/naročnik se lahko uporablja tudi za komunikacijo med aplikacijami ali ločenimi moduli (npr. aktivnostmi).

MQTT

- Osnovne funkcionalnosti so:
  - Poveži (Connect),
  - Objavi (Publish),
  - Naroči (Subscribe).
- Srečujemo se s težavami kot je:
  - kvaliteta storitev (Quality of Service),
  - zadržana sporočila (Retained Messages),
  - stalna seja (Persistent Session),
  - zadna volja in oporoka,
  - ohranjanje povezave (Keep Alive).



Lastnosti protokola MQTT

- Glavne lastnosti:
  - Lahek protokol – *mali stroški* izmenjave podatkov (pasovna širina, hitrost).
    - Distribucija minimalnih paketov podatkov v velikih količinah.
  - Preprosta objava podatkov (en klic).
  - Selektivna izbira poslušanja posameznih dogodkov.
  - Dogodkovno usmerjena arhitektura.
  - Skrb za zanesljivi prenos podatkov (velike oddaljenosti, različna mrežna infrastruktura).
  - Majhne in učinkovite knjižnice, ki omogočajo delo na napravah IoT (energijsko učinkovite).
  - Skrb za varnost in zasebnost podatkov.
  - Velika skalabilnost
    - Možnost paralelizacija pošiljanja, sprejemanja, …
    - Predpomnjenje sporočil.
    - Možnost na tisoče naročnikov.
    - Na tisoče pošiljateljev.

Prostorska, časovna in procesorska sklopljenost

Prostorsko ločevanje (space decoupling) izdajatelj in naročnik ne vesta drug za drugega (ne poznata niti naslova IP in vrat).

Časovna ločenost (time decoupling) izdajatelj in naročnik lahko izmenjata sporočila tudi kadar nista hkrati zagnana (asinhrona komunikacija).

Neodvisno delovanje (synchronization decoupling). Ostale operacije na izdajatelju in naročniku med procesom objavljanja ali sprejemanja ni potrebno prekiniti.

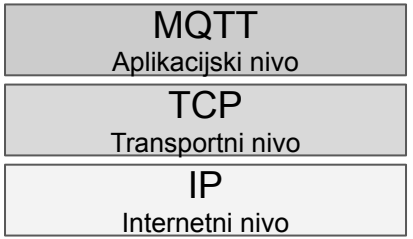
Filtriranje sporočil

Filtriranje je ključno saj ne želimo prejemati, še manj pa pošiljati sporočila, ki nam niso namenjena, to velja iz vidika učinkovitosti, kakor tudi iz vidika varnosti.

- Filtriranje **po temi**
  - Naročnik se naroči na določeno temo (topic) oz. vzorec tem, ki so del sporočil.
  - Teme so predstavljene s hierarhično strukturo.
- Filtriranje po **vsebini sporočila**
  - Na podlagi prejete vsebine posrednik filtrira sporočilo in ga posreduje.
  - Slabost je da mora odjemalec vnaprej vedeti katere vsebine so znane in **zato vsebina ne more biti šifrirana**.
- Filtriranje po **tipu ali podtipu**
  - Podobno kot pri npr. pri programiranju ko pošiljamo vsa izpeljana sporočila iz razreda Exception.

Razlike med MQTT in navadnimi vrstami sporočil (message queue)

- **Vrsta sprejema sporočila dokler se ne odpošljejo naprej.** V vrsti zato nimamo sporočil, ki nimajo odjemalcev. V primeru MQTT je možno sprejeti sporočilo tudi ko ni nobenega naročnika na posamezno temo.
- **Sporočilo v vrsti je namenjeno samo enemu odjemalcu**. V primeru MQTT vsi, ki so naročeni na določeno temo dobijo sporočilo.
- **Vrste sporočil so vnaprej določene in morajo biti zato ustvarjene.** V primeru MQTT se lahko teme določajo sproti (med delovanjem), kar omogoča večjo prilagodljivost.



Povezava 1/2 (MQTT Connection)

Povezujemo se na posrednika.

Ukaz MQTT **CONNECT** vsebuje paket z naslednjimi elementi:

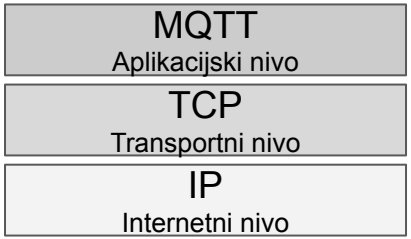
clientId
“clent-121s”
univerzalno ime/identifikator (identifier), ki je obvezno kadar posrednik mora voditi stanje odjemalca

cleanSession
TRUE pomeni da posrednik ne shranjuje zgrešenih sporočil in zbriše staro sejo. V primeru FALSE se shranijo vsa zgrešen sporočila QoS nivoja 1 in 2.

username/password
Uporabniško ime in geslo za avtentikacijo in avtorizacijo odjemalca.

lastWillTopic, lastWillQos, lastWillMessage, lastWillRetain
Podatki o poslednji volji odjemalca, ki se zgodi v primeru nepričakovane prekinitve.

keepAlive
Interval v sekundah, ki določa najdaljši dovoljeni čas med odjemalcem in posrednikom, ki je dovoljen brez komunikacije. ***Odjemalec se zaveže, da bo redno pošiljal PING zahteve*** na katere posrednik prav tako odgovori s PING zahtevo.



Povezava 2/2 (MQTT Connection)

Odgovor na CONNECT MQTT je **CONNACK** in vsebuje

sessionPresent

“TRUE/FALSE”

Pove odjemalcu če ima posrednik pri sebi že trajno sejo (lahko vsebuje zgodovino). Trajno sejo, t.j. TRUE vrne smo če smo pri predhodnih klicih imeli povezavo, ki je imela zastavico cleanSession nastavljeno na false.

returnCode

Vrne ali je bila povezava uspešno vspostavljena.

0 - Povezava sprejeta, 1 - Povezava zavrnjena, nesprejemljiva verzija protokola, 2 - povezava zavrnjena (identifikator odjemalca zavrnjen), 3 - Povezava zavrnjena (nedosegljiv server), 4 - povezava zavrnjena (napačno ime ali geslo), 5 - povezava zavrnjena (ni avtoriziran)

MQTT Objavi

PUBLISH paket vsebuje:

packetId
Številka paketa, ki je v primeru nastavitve QoS = 0 vedno 0 (**Q**uality **o**f **S**ervice).

topicName
Niz, ki predsatvlja hirarhično predstavitev tem. “Celje/Parking/1/empty”

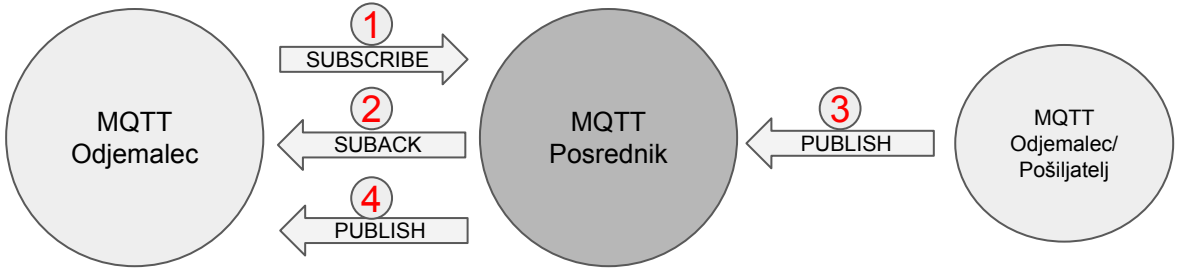
QoS
Nastavitev kvalitete pošiljanja od 0 do 2. Vpliva na povratno informacijo o sprejetju sporočila. Več kasneje.

retainFlag
Pove serverju ali si zapomne to vrednost kot zadnjo pravilno vrednost za določeno temo. Ko se nov odjemalec prijavi dobi to vrednost takoj (npr. temperaturni senzor, status odprtja garaže).

payload
To je poleg **topicName** najpomemnejša nastavitev, saj v njo shranimo vrednost sporočila. Sporočilo je poljubnega tipa (data-agnostic). Ponavadi je to število, niz, lahko pa je json niz, slika...

DUP flag
Zastavica se nastavi v primeru ponovnega pošiljanja sporočila (duplicate). Za zastavico poskrbijo MQTT implementacije. Nastavitev je pomembna pri stopnji QoS dve (točno enkrat).

MQTT Naroči



SUBSCRIBE paket vsebuje:

packetId - Številka paketa.
Lista tem na katere se naročimo in njihov QoS status

QoS1 - število med 0 in 2,
Topic1 - niz ki predstavlja temo 1,
QoS2 - število med 0 in 2,
Topic2 - niz ki predstavlja temo 2, ...

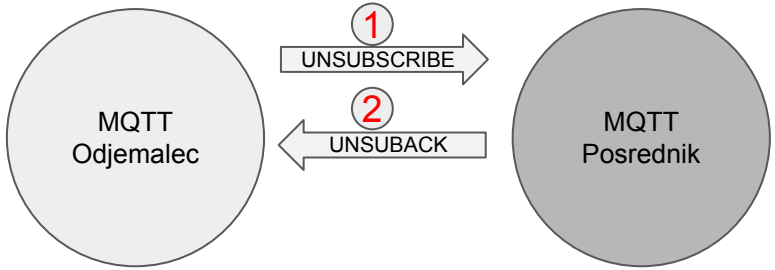
SUBACK odgovor posrednika na zahtevo SUBSCRIBE

packetId - Številka paketa.
returnCode list

Za vsako temo dobimo odgovor o uspešnosti prenosa.

- 0 - Uspešno (Maksimum QoS = 0)
- 1 - Uspešno (Maksimum QoS = 1)
- 2 - Uspešno (Maksimum QoS = 2)
- 128 - Neuspešno

MQTT Odjavi



UNSUBSCRIBE paket vsebuje:

packetId - Številka paketa.
Lista tem na katere se odjavljamo
Topic1 - niz ki predstavlja temo 1,
Topic2 - niz ki predstavlja temo 2, ...

UNSUBACK odgovor posrednika na zahtevo

packetId - Številka paketa.

- Po prejetju paketa lahko predvidevamo da se je odjemalec uspešno odjavil od izbranih tem.

MQTT Teme

Pojem tema (topic) se nanaša na niz v UTF-8, ki ga uporablja posrednik za filtriranje sporočil.

Tema je lahko sestavljena iz podtem, ki predstavljajo posamezne nivoje (levels) tem. Za ločevanje nivojev se uporablja znak “/”.

Primer: Slovenija/Celje/IV Osnovna šola/876256281/water/status

*Presledkom in posebnim znakom v temah se je bolje izogibati

MQTT Nadomestni/maskirni znaki tem

Nadomestni/maskirni znak (wildcard).

Poljubna vrednost na posameznem nivoju znak “**+**”

Poljubna vrednost na vseh nadaljnjih nivojih “**#**”

Teme ki se začnejo z znakom “**$**”

Teme $ so namenjene posredniku in njegovi interni statistiki. Pogosto so te teme v obliki $SYS/.

```
$SYS/broker/clients/connected
$SYS/broker/clients/disconnected
$SYS/broker/messages/sent
$SYS/broker/uptime
$SYS/broker/clients/total
```

lpm/lab/+/luč

```
lpm/lab/spredaj/luc
lpm/lab/spredaj/okno
lab/spredaj/luc/datum
lpm/kabinet1/spredaj/luc/datum
lpm/lab/zadaj/luc
```

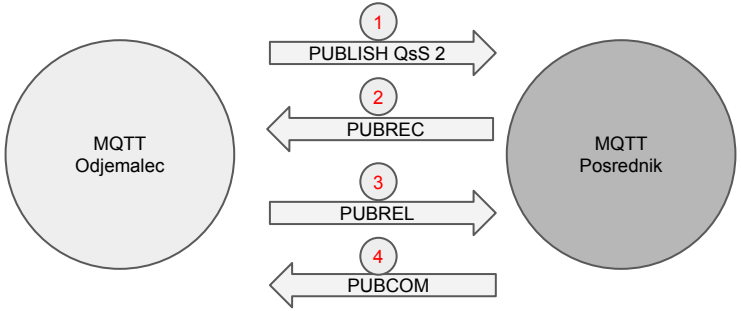
lpm/lab/#

```
lpm/lab/spredaj/luc
lpm/lab/spredaj/okno
lab/spredaj/luc/datum
lpm/kabinet1/spredaj/luc/datum
lpm/lab/zadaj/luc
```

MQTT dobre prakse pri poimenovanju tem

- **Nikoli** ne uporabljaj **vodilne poševnice** “/”. Npr. **/**`hisa/vhod/luc`
- Nikoli ne uporabljajte presledkov v temah. Presledki in nekateri znaki so lahko različno predstavljeni na različnih platformah.
- Določi preprosto in kratko taksonomijo.
- Uporabljaj samo ASCII znake in se izogibaj znakov, ki se ne tiskajo.
- V temo vključi unikaten identifikator pošiljatelja.
- Ne naročaj se na temo # (pridobivanje vseh sporočil).
- Ob ustvarjanju tem, je bolje imeti večjo globino nivojev kot krajšo.

QoS - Kvaliteta storitve 1/2



Je dogovor med prejemnikom in pošiljateljem sporočila, ki določa gotovost prejema sporočila.

Nivo 0 - sporočilo dobiš največ enkrat (lahko tudi da ne prejmeš sporočila, zanesljivost je enaka protokolu TCP). Imenuje se tudi “fire and forget” metoda (PUBLISH QoS 0).
Nivo 1 - sporočilo dobiš najmanj enkrat (lahko da dobiš enako isto sporočilo večkrat - PUBLISH QoS 1). Pošiljatelj zadrži sporočilo in ga pošilja dokler ne dobi uspešnega PUBACK potrdila. Pri večkratnem pošiljanju se nastavi zastavica DUP (duplicate).
Nivo 2 - sporočilo se prejme točno enkrat. V ta namen potrebujemo štiri sporočila ali štiridelno uparitev (handshake) (glej sliko). PUBLISH QoS 2, PUBREC (sprejel), PUBREL (izdaja sporočanja), Izdjatelj lahko sedaj sprazne “buffer” in sporoči PUBCOMP.

QoS - Kvaliteta storitve 2/2

- Sporočilo potuje dvakrat enkrat od pošiljatelja do posrednika, drugič od posrednika do naročnika. Pošiljatelj določi QoS sporočila, vendar če je kvaliteta QoS določena nižja na strani sprejemnika, se upošteva nižja kvaliteta.
- Identifikator je unikaten samo med pošiljateljem in posrednikom. Po končanem sporočilu je lahko identifikator ponovno uporabljen.
- Vsa sporočila QoS 1 in 2 se hranijo v vrsti za odjemalce, ki trenutno niso povezani. Stanja se shranjujejo samo če je nastavljena trajna seja.
- **Priporočila**
    - QoS 0 se uporablja na zanesljivih povezavah; v primerih ko si lahko dovolimo da izgubimo kakšno sporočilo; sporočila se pošiljajo v kratkih intervalih…
    - QoS 1 se uporablja najpogosteje. Zagotoviti je potrebno da odjemaleca ne “zmede” večkratni prejem sporočila.
    - QoS 2 je večkrat počasnejše od QoS 1. Uporablja se izjemoma v primerih kjer lahko večkratni sprejem sporočila škoduje prejemniku.

Trajna seja in čakalne vrste

- V primeru izpada odjemalca (prekinjena povezava do posrednika) se lahko sporočila na katera je naročen odjemalec shranjujejo na posredniku (trajna seja).
- V trajni seji je shranjeno: obstoj seje, vse naročene teme, vsa sporočila (QoS 1 in 2), ki jih odjemalec še ni dobil ali potrdil ali povsem potrdil (QoS 2).
- Trajna seja se ustvari ob nastavitvi povezave cleanSession false.
- Seja in sporočila se zbrišejo ob povezavi z zastavico cleanSession true.
- Odjemalec lahko preveri, če je stalna seja prisotna tako da ob uspešni povezavi prebere odgovor CONNACK, kjer je zastavica sessionPresent nastavljena na TRUE ali FALSE.
- **Lahko se zgodi da določenih odjemalcev ni več. V tem primeru se sporočila nabirajo pri posredniku. (Brisanje več na WWW)**

Zadnje stanje/vrednost sporočila (retain state)

Pogosto se zgodi da nas zanima zadnje stanje izbrane teme. V primeru ko izdajamo sporočila redko npr. vsakih 24 ur ali redkeje se zgodi, da nekdo ki se naroči na posamezno temo, dobi sporočilo po preteku določenega časa. Da omogočimo takojšnje pridobivanje vrednosti zadnje teme ima protokol podporo nastavljanja zadnje teme. To storimo z nastavitvijo zastavice **retainFlag.** Posrednik ohrani samo eno sporočilo na temo. V primeru tem, ki vsebujejo nadomestne znake (+,#) se shrani za vse teme ki spadajo v te kategorije.

Trajna sporočila zbrišemo z novim sporočilom ali pa s praznim sporočilno (payload).

Trajna storočila se uporabljajo vedno kadar želimo da novi ali na ponovno prijavljen odjemalec takoj dobi zadnje sporočilo. *(Naštej nekaj primerov)*

Poslednja volja odjemalca

V primeru, da se odjemalec nepričakovano poslovi od posrednika je mnogokrat potrebno o tem obvestiti zainteresirane odjemalce. Npr. vklopiti nadomestni server ali storitev. To se naredi s poslednjo voljo, ki ni nič drugega kot MQTT sporočilo.

Poslednja volja se nastavi ob povezavi MQTT in vsebuje temo, ratainflag, QoS stopnjo ter vsebino. Vsi ki so naročeni na to temo bodo dobili sporočilo, ki jo ob, nezaželjeni prekinitvi, sproži posrednik.

Do poslednje volje pride ko: posrednik zazna napako I/O napako, odjemalec ne komunicira v obdobju ohranjanja življenja, posrednik zapre odjemalca zaradi napake omrežja.

Dobra praksa je, da se uporablja MQTT za pošiljanje sporočil kot so ONLINE / OFFLINE.

Praktični primer

Knjižnice odjemalec in posrednik. WWW

Varni kanali (http://www.steves-internet-guide.com/mosquitto-tls/).

Drugi načini komunikacije

V kakšnem nivoju komunikacije govorimo?

Hitrost komunikacije?

Zanesljivost komunikacije?

Integracija protokola komunikacije?

Odprtokodni standard?

Posredna komunikacija? Npr. naprava odda zvok, ki ga zazna mikrofon, ...

SmartThings platforma

Podpira višjo stopnjo abstrakcije kot MQTT.

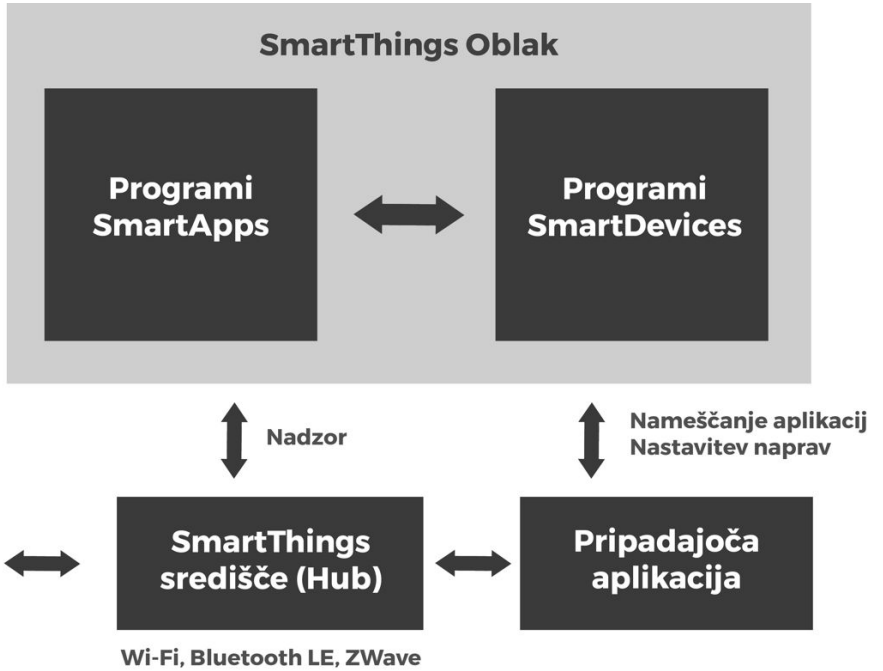
https://partners.smartthings.com/

- Je platforma za:
  - ustvarjanje aplikacij za povezovanje naprav, akcij in zunanjih storitev za avtomatizacijo,
  - integracijo novih naprav v SmartThings ekosistem in
  - objavo aplikacij in integracijo v SmartThings katalog.
- Razvoj aplikacij je omogočen s podporo programskega jezika Groovy.
- Enostavna kontrola naprav

```
def someoneArrived(evt) {
    lights.on()
    music.on()
    sendPush("Dobrodošli doma!")
}
```

Arhitektura SmartThings platforme

- Naprave zavite v ovoj SmartDevice (SmartThings).
- SmartAps se lahko naročajo na dogodke SmatDevices, pošiljajo SMS, naročajo na lokacijo uporabnika itd..
- Programi se izvajajo v SmartThings oblaku.
- Ima lastno varnostno arhitekturo, ki upravlja, do katerih naprav lahko dostopa.
- Izvajanje posameznih aplikacij v peskovniku Koshuke (Grooy nadrazred)

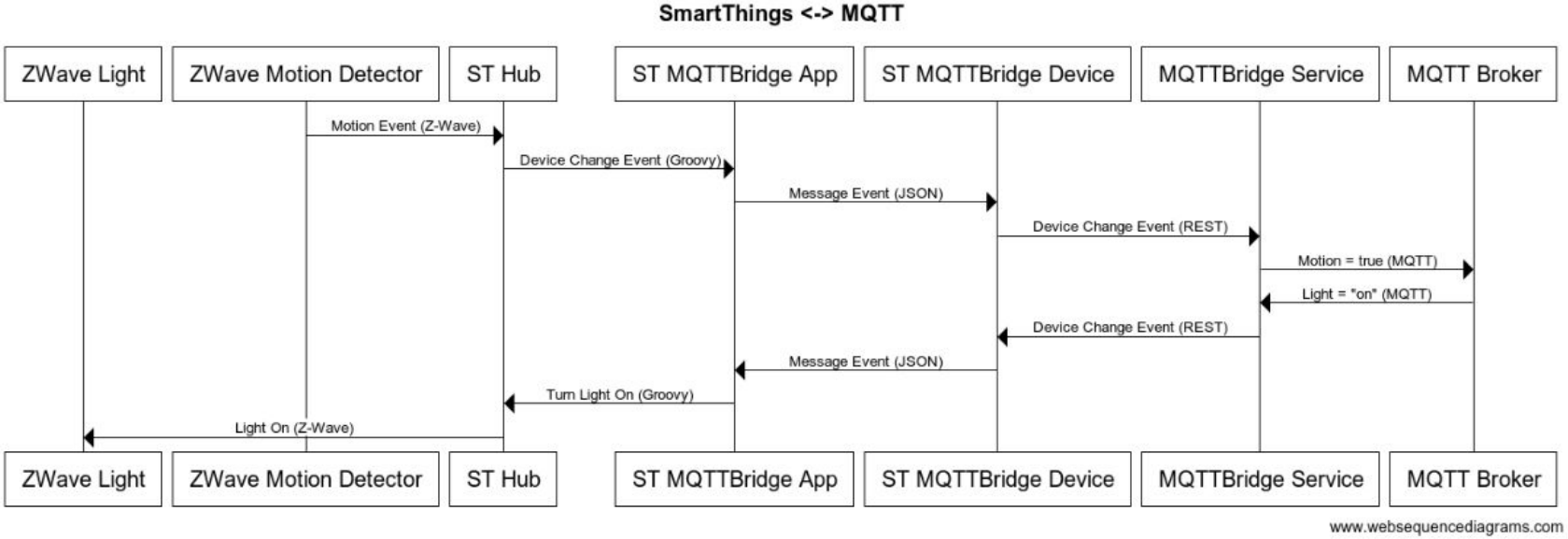


https://partners.smartthings.com/supported-devices

Mostovi

Kako vključiti nezdružjive platform/naprave? Potrebno je dodati nekakšnega tolmača, prevajalca komunikacije, ki predstavlja most med tehnologijami.

Primer: MQTT Bridge to SmartThings (prednosti slabosti)



Apple protokol HomeKit

- HomeKit sestavljata sklopa:
  - Vmesnik za programiranje (angl. Application Programming Interface, krajše API), ki omogoča komunikacijo z dodatki.
  - Protokol HomeKit (angl. HomeKit Accessory Protocol, krajše HAP) omogoča komunikacijo z raznoraznimi fizičnimi napravami (npr. ključavnicami, vrati, termostati, …
- iOS naprava uporablja HAP za odkritje, preiskovanje in interakcijo s HomeKit dodatki, kot so luči, senčila, rampe, ...
- HAP uporablja za komunikacijo z napravami Bluetooth LE ali internet protokol (IP).
- Seja komunikacije med napravami je kriptirana. To omogočajo z vzpostavitvijo seznanitve (angl. Pair Setup) in potrditvijo seznanitve (angl. Pair Verify).

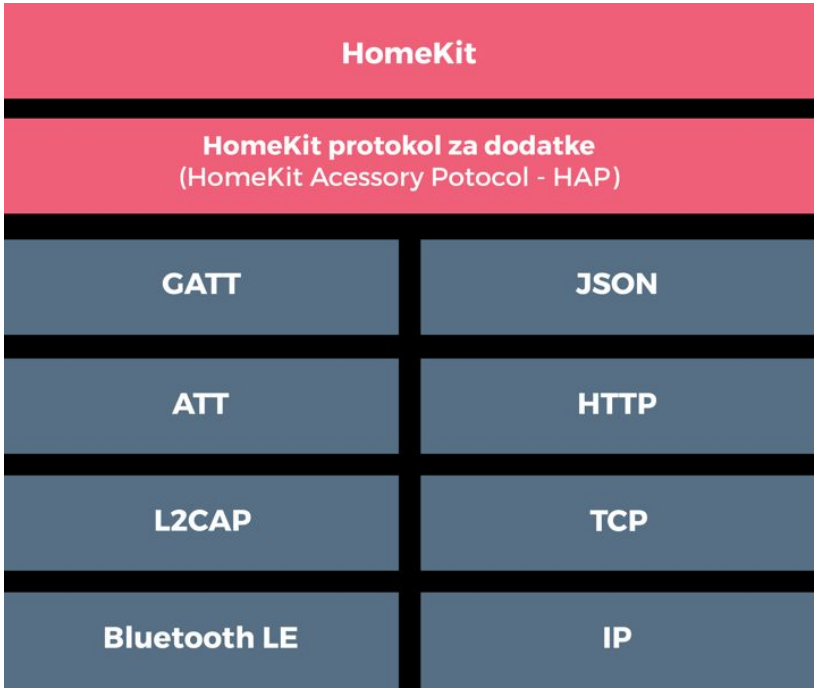
https://www.apple.com/shop/accessories/all-accessories/homekit

Transportni sklad protokola HomeKit

V odvisnosti od načina komunikacije imamo različni transportni sklad, ki je na vrhu poenoten z Homekit protokolom za dodatke.

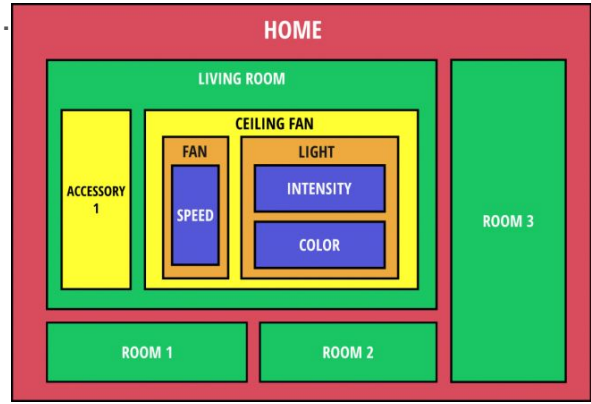
Dva sklada protokolov IP in Bluetooth.

Npr.: Bluetooth->Logical Link Control and Adaptation Protocol (L2CAP)->Attribute Protocol (ATT)->Generic Attribute Profile (GATT)

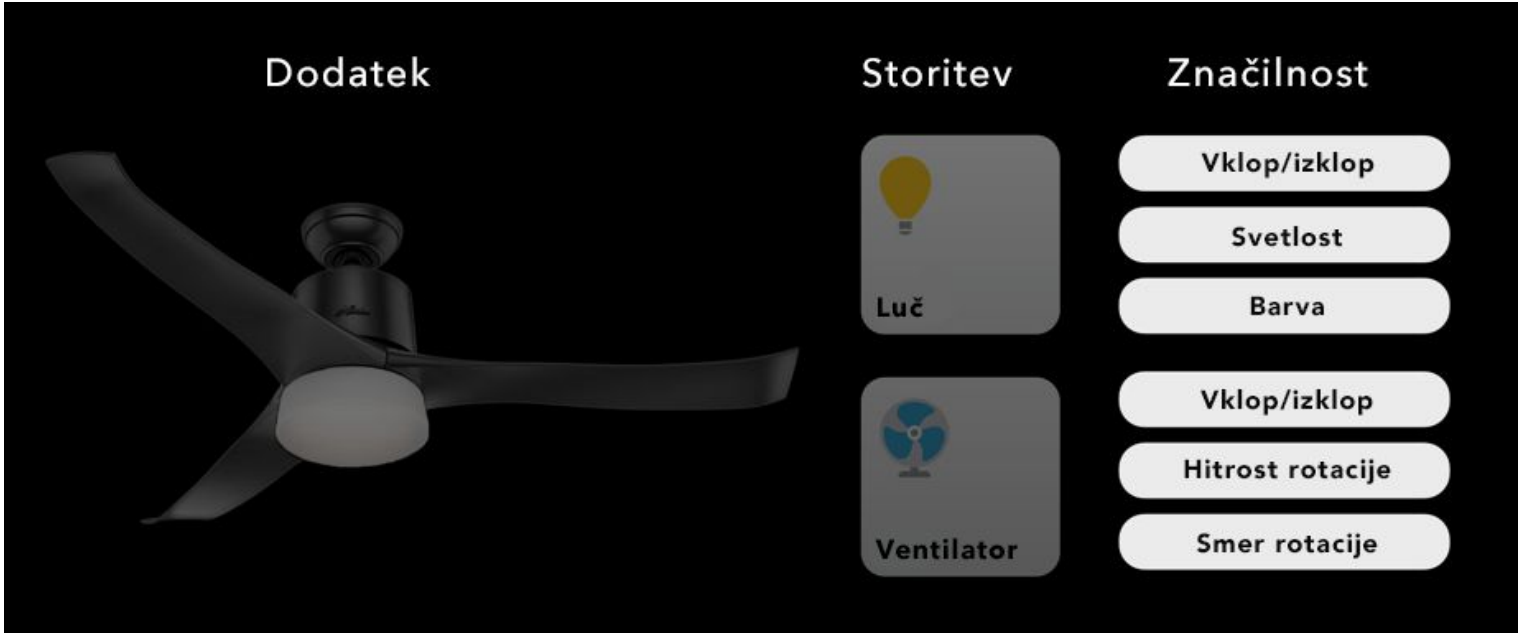


HomeKit elementi 1/2

- Skupna baza omogoča konsistentno komunikacijo dodatkov z različnimi aplikacijami.
  - Vsebuje vse podatke o prostorih, napravah, scenah, prožilcih, …
- Hierarhija objektov vsebuje štiri večje skupine
  - Dom (Home), Cona (Zone), Soba (Room) in Dodatek (Accessory).
  - Dodatek ima svoje storitve in vsaka storitev vsaj eno značilnost.
- Vsaka naprava ima svoje vloge.
  - Glede na vloge imajo naprave različne funkcionalnosti.
  - HAP strežnik razkriva zbirko dodatkov HAP odjemalcem.
  - HAP dodatek je fizična naprava, ki deluje preko HAP strežnika.
  - HAP most je posebna vrsta strežnika, ki skrbi za komunikacijo z drugimi transportnimi protokoli.



HomeKit elementi 2/2



HomeKit varnost

- Poleg osnovnih transportnih mehanizmov varnosti se uporablja veliko različnih algoritmov:
  - **Protokol SRP** (angl. Secure Remote Password) Šifrira in avtorizira izmenjavo ključev pri prvotnem seznanjanju.
  - **Curve25519** Šifrira začetno avtentikacijo za vsako sejo.
  - **Ed25519** Dolgoročni ključi za seznanjanje in avtentikacijo.
  - **HKDF-SHA-512** Empirična izpeljava šifrirnih ključev za vsako sejo.
  - **ChaCha20-Poly 1305** Šifrira in avtorizira HAP podatke.
- Licenca za proizvajalce je dostopna s tajno pogodbo, za razvijalce programske opreme pa je več omejitev.