SBNZ
Smart Home
Pametna Kuća

Članovi Tima

- Luka Šerbedžija SW-32-2018
- Miloš Panić SW-19-2018
- Luka Bjelica SW-21-2018

Motivacija

IoT tehnologije postaju sve popularnije i počinju da se primenjuju u svim aspektima ljudskog života. Oblast korišćenja na koju je stavljen fokus jesu pametne kuće, odnosno kućna automatizacija, čiji je cilj da korisnikove svakodnevne aktivnosti učine lakšim, bezbednijim i efikasnijim. Sistem se oslanja na skup senzora koji prikupljaju podatke i/ili mogu biti upravljani od strane korisnika ili sistema. Kako bi korisnici mogli da izvuku maksimum iz uređaja koji su instalirani, potrebno je da imaju dobar softver koji će im omogućiti efikasan rad sa njima.

Pregled problema

Kućna automatizacija omogućava korisniku pristup kontrolnim uređajima u kući sa bilo kog uređaja bilo gde u svetu. Preciznije opisuje domove u kojima je skoro sve - svetla, uređaji, električne utičnice, sistemi za grejanje i hlađenje - uključeno na mrežu kojom se može daljinski upravljati.

Pored upravljanja uređajima, sistem treba da omogući nadgledanje podataka koje senzori šalju, kako bi korisnik imao uvid u njih. S obzirom da se radi o velikoj količini podataka koji se periodično šalju (*time series data*), sistem treba da vrši agregaciju podataka u pozadini. Na taj način, korisnik bi mogao da ima uvid u tok podataka kroz vreme, što je njemu najbitnije.

Iz perspektive kućne bezbednosti, ovo takođe uključuje postojanje alarmnog sistema, koji je povezan sa svim vratima, prozorima, bravama, detektorima dima, nadzornim kamerama i svim drugim senzorima koji su povezani sa njim.

Činjenica da postoji veliki broj slučajeva primene sistema, kao i veliki broj senzora koji se mogu koristiti, poželjno je da sistem bude konfigurabilan kako bi mogao pokriti svaki od njih.

Postoji veliki broj rešenja ove prirode, ali su oni uglavom ograničeni na rad sa malim skupom senzora koji su proizvedeni od strane određenog proizvođača. Takvi sistemi uglavnom podrazumevaju i ograničenje na korišćenje njihovog softverskog rešenja, bez mogućnosti integracije sa drugima.

Cilj je da rešenje koje razvijemo bude višenamensko i proširivo na rad sa proizvoljnim skupom senzora, konfigurabilno kako bi podržalo rad u različitim kontekstima primene i bazirano na pravilima, što povećava stepen automatizacije i doprinosi lakoći daljeg razvoja sistema.

Metodologija Rada

Postoje dve vrste ulaza u sistem, real-time podaci koji stižu sa senzora i podaci koje korisnik unosi. Od podataka sa senzora očekivani su sledeći parametri:

- potrošnja struje
- temperatura
- vlažnost vazduha
- udeo ugljen dioksida i kiseonika u vazduhu
- statusi uređaja
- prisutnost u prostoru
- jačina zvuka
- koordinate
- i dodatne specifične informacije sa određenih uređaja.

Od uređaja kojima se može upravljati, i koji predstavljaju izlaze is sistema, očekivani su:

- sijalice
- prekidači
- utičnice
- emiteri zvuka alarma
- bojleri
- frižideri
- rerne
- klima uređaji
- radijatori
- lock sistemi (pametne brave)
- veš mašine
- mašine za sudove
- mikrotalasne
- pametni prozori
- kamere
- pametna ogrlica za ljubimce

Kako bi sistem bio upotrebljiv u različitim uslovima, potrebno je omogućiti konfiguraciju pragova vrednosti za svaki parametar za koji se definišu pravila. To se rešava uvođenjem koncepta Konteksta, koji predstavlja određenu konfiguraciju sistema. Kontekst može biti ručno ili automatski uključen, odnosno isključen. Kontekst se može aktivirati određenim pravilom, odnosno reakcijom na određeni događaj, kao i definisanim vremenskim periodom važenja. Sledeći parametari, odnosno ulazi u sistem, se mogu konfigurisati za određeni kontekst:

- min max potrošnja struje
- min max temperatura
- min max vlažnost
- min max CO2
- min max O
- status uređaja (određenog) On/Off
- min max intenziteti pokreta (definicija prisutnosti/odsustva)
- min max jačina zvuka

Postojaće predefinisani konteksti:

- Away From Home, u daljem tekstu AFH

- At Home, u daljem tekstu AH
- Eco
- Extra Safe
- Winter
- Spring
- Summer
- Autumn
- Daily
- Nightly
- Custom

Baza znanja

Legenda:
- K - Kontekst, Kprev - Prethodni kontekst
- C - Konfiguracija, [K] jedna definicija za specifican Kontekst - K
- Konfiguracija(<parametar>) - vrednost parametra iz konfiguracije konteksta
- naziv() u when delu - drools query funkcija
- naziv() u then delu - akcija
- dispatch(eventName) - ispaljivanje eventa

Pravila:

when Kprev != AFH and K = AFH
then takeSensorStatusesSnapshot()

when K = AFH and C[K](max_potrošnja) < potrošnja
then informUser() and activateContext(Eco)

when (K = AFH or K = Night) and outdoorCamDetectedPerson()
then informUser() and startRecording() and dispatch(PotentialRoberyMovement)

when K = AFH and C[K](max_int_pokret) < int_pokret
then informUser() and startRecording() and dispatch(PotentialRoberyMovement)

when K = AFH and C[K](max_zvuk) < zvuk
then informUser() and startRecording() and dispatch(PotentialRoberySound)

when PotentialRoberyMovement and PotentialRoberySound
then activateAlarmSystem() and informUserOfAlarm()

when K = AFH and DoorOrWindowsStatusChanged
then activateAlarmSystem() and informUserOfAlarm()

when any(K) and C[K](max_temp) < temp
then informUser() and dispatch(TemperatureTooHigh)

when any(K) and C[K](min_temp) > temp
then informUser() and dispatch (TemperatureTooLow)

when K = Summer and K = Nightly and TemperatureTooHigh
then openWindows()

when K = Summer and K = Daily and TemperatureTooHigh
then turnOnAC()

when K = Summer and K = Daily and TemperatureTooLow and isACOn()
then turnOffAC()

when K = Summer and K = Daily and TemperatureTooLow and !isACOn()
then openWindows()

when K = Summer and isACOn() and areWindowsOpen()
then closeWindows()

when K = Winter and TemperatureTooHigh
then turnOffHeating()

when K = Winter and TemperatureTooLow
then turnOnHeating()

when K = Winter and isHeatingOn() and areWindowsOpen()
then closeWindows()

when any(K) and C[K](max_CO2) < CO2
then informUser() and dispatch(CO2TooHigh)

when any(K) and C[K](min_humid) > humid
then informUser() and dispatch(HumidTooLow)

when TemperatureTooHigh and CO2TooHigh and HumidTooLow
then activateSprinklers() and informUserOfAlarm() and closeWindows() and turnOffACSupply()

when K = AH and C[K](int_prisutnost_kupatilo) < prisutnost_kupatilo and NotInKupatilo
then turnOffBoiler() and turnLightsOn() and dispatch(IsInKupatilo)

when K = AH and C[K](int_prisutnost_kupatilo) > prisutnost_kupatilo and IsInKupatilo
then turnOnBoiler() and turnLightsOff() and dispatch(NotInKupatilo)

when K = Nightly and K = Eco and WashingMachineIsOn
then turnOnWashingMachine() and dispatch(WashingMachineIsOn)

when C[K](int_prisutnost_ulazna_vrata) < prisutnost_ulazna_vrata

then informUser() and dispatch(MovementOnTheFrontDoor)

when C[K](int_prisutnost_ulazna_vrata) > prisutnost_ulazna_vrata
then dispatch(NothingOnTheFrontDoor)

when K = Nightly and !areFrontDoorLightsOn() and MovementOnTheFrontDoor
then turnFrontDoorLightsOn()

when K = Nightly and areFrontDoorLightsOn() and MovementOnTheFrontDoor
then turnFrontDoorLightsOff()

when K = AFH and doorbellRings()
then informUser()

when K = AH and doorbellRings()
then informUser() and turnVideoCallOn() and dispatch(VideoCallOn)

when NothingOnTheFrontDoor and VideoCallOn
then turnVideoCallOff() and dispatch(VideoCallOff)

when any(K) and tempSensorWentOffline()
then informUser() and dispatch(TempSensorOffline)

when any(K) and co2SensorWentOffline()
then informUser() and dispatch(CO2SensorOffline)

when any(K) and humidSensorWentOffline()
then informUser() and dispatch(HumiditySensorOffline)

when TempSensorOffline and CO2SensorOffline and HumiditySensorOffline
then informUser() and dispatch(PotentialFire)

when any(K) and tempSensorWentOnline()
then informUser() and dispatch(TempSensorOnline)

when any(K) and co2SensorWentOnline()
then informUser() and dispatch(CO2SensorOnline)

when any(K) and humidSensorWentOnline()
then informUser() and dispatch(HumiditySensorOnline)

when TempSensorOnline and CO2SensorOnline and HumiditySensorOnline
then informUser() and dispatch(NotPotentialFire)

when K != PetWalk and petBraceletOutOfRange(C[K](home_radius))
then informUser() and dispatch(PotentialPetMissing)

when K != PetWalk and PotentialPetMissing and petBraceletOutOfRange(C[K](neighborhood_radius))
then informUser() and dispatch(PetMissing)

when K == AFH and PetMissing
then alarmUser() and startSendingRealTimeLocation()

when PetMissing and petBraceletInRange(C[K](neighborhood_radius))
then informUser() and dispatch(PotentialPetCameBack)

when PotentialPetCameBack and petBraceletInRange(C[K](home_radius))
then informUser() and dispatch(PetCameBack)

Primer Forward Chaining-a:

when any(K) and C[K](max_temp) < temp
then informUser() and dispatch(TemperatureTooHigh)

when any(K) and C[K](max_CO2) < CO2
then informUser() and dispatch(CO2TooHigh)

when any(K) and C[K](min_humid) > humid
then informUser() and dispatch(HumidTooLow)

==========

when TemperatureTooHigh and CO2TooHigh and HumidTooLow
then dispatch(PotentialFire)

==========

when PotentialFire and isWindowOpen()
then closeWindow()

when PotentialFire and isACOn()
then turnOffAC()

when PotentialFire
then activateSprinklers() and dispatch(SprinklersActivated)

==========

when SprinklersActivated and isAnyElectricDeviceOn()
then turnOffAllElectricDevices()

when SprinklersActivated over: window(3 min)
then notifyFireStation()

Primer Backward chaining-a:

Agregirani podaci će biti organizovani u obliku strukture stabla. Čvor će predstavljati 24h agregacija dok će listove predstavljati 5min agregacija. Uz pomoć ovog stablja možemo praviti određene izveštaje koji će prikazivati optimalne periode. Optimalni period može biti npr. optimalna potrošnja struje gde ćemo gledati da li je struja manja od onoga što se smatra optimalnom.

Primer CEP-a:

Podaci sa senzora će biti predstavljeni u obliku događaja. Ovakvih događaja bi bilo previše jer senzori mogu slati vrednosti na nivou sekunde, nisu pogodni za analizu i uglavnom nam ne trebaju u ovom obliku. Događaje ćemo agregirati uz pomoć sum, average, min, max, count funkcije na nivou 5, 15, 30, 60 minuta, 1, 2, 4, 8, 24h. Kako bi postupak bio najoptimalniji, rezultati petominutnih agregacija (događaji) će se koristiti za 15 minutne agregacije (događaj) i tako ulančano do 24h. Takođe je bitno da su vremena agregacija zaokružena, npr. da je 5 minutna na 15:00, 15:05, 15:10 a ne 15:08:42, 15:13:42, 15:18:42. Dodatno će biti moguće i grupisanje podataka.

Primer template-a:

Korisnici će biti u mogućnosti da definišu pravila dinamički. Od uslova će im biti omogućeno da rade upite nad kontekstima poput when C uz logičke operatore AND i OR, upite nad poslednjim vrednostima očitanih sa senzora npr. current_temp uz relacione i aritmetičke operatore i pozivanje predefinisanih funkcija kao što su areWindowsOpen() (koji će biti implementirani kao drools query). Što se akcija tiče, korisnik će moći da ispali predefinisani event, postavi aktivaciju narednog konteksta.

Za template će biti napisan mali DSL.