UNIVERZA V LJUBLJANI

FAKULTETA ZA RAČUNALNIŠTVO IN INFORMATIKO

Andrej Kavrečič

Sistem za upravljanje zgradb

UNIVERZITETNI ŠTUDIJSKI PROGRAM PRVE STOPNJE
RAČUNALNIŠTVO IN INFORMATIKA

Mentor: prof. dr. Saša Divjak

Ljubljana, 2012

Univerza *v Ljubljani*
Fakulteta *za računalništvo in informatiko*

Tržaška 25
1001 Ljubljana, Slovenija
telefon: 01 476 84 11
01 476 83 87
faks: 01 426 46 47
01 476 87 11
www.fri.uni-lj.si
e-mail: dekanat@fri.uni-lj.si

Št. naloge: 00006/2011
Datum: 05.09.2011

Univerza v Ljubljani, Fakulteta za računalništvo in informatiko izdaja naslednjo nalogo:

Kandidat: **ANDREJ KAVREČIČ**

Naslov: **SISTEM ZA UPRAVLJANJE ZGRADB**
BUILDING MANAGEMENT SYSTEM

Vrsta naloge: Diplomsko delo univerzitetnega študija prve stopnje

Tematika naloge:

Razvijte programsko opremo za sistem za upravljanje zgradb in svojo rešitev primerjajte s trenutno aktualnimi rešitvami za avtomatizacijo zgradb. Opišite povezovalne protokole, ki v sistemu za upravljanje zgradb služijo za komunikacijo med napravami v zgradbi. Navedene standarde primerjajte med seboj. Opišite uporabljen sistem za avtomatizacijo zgradb in njegove komponente. Predstavite možnosti, ki jih nudi razviti sistem in njegov uporabniški vmesnik.

Mentor:

prof. dr. Saša Divjak

Fakulteta za računalništvo in informatiko
UNIVERZA V LJUBLJANI
2

Dekan:

prof. dr. Nikolaj Zimic

I Z J A V A O A V T O R S T V U

diplomskega dela

Spodaj podpisani/-a **Andrej Kavrečič,**

z vpisno številko **63050141,**

sem avtor/-ica diplomskega dela z naslovom:

SISTEM ZA UPRAVLJANJE ZGRADB

S svojim podpisom zagotavljam, da:

- Sem diplomsko delo izdelal/-a samostojno pod mentorstvom (naziv, ime, priimek) prof. dr. Saša Divjak

- So elektronska oblika diplomskega dela, naslov (slov., angl.), povzetek (slov., angl.) ter ključne besede (slov., angl.) identični s tiskano obliko diplomskega dela

- Soglašam z javno objavo elektronske oblike diplomskega dela v zbirki "Dela FRI".

V Ljubljani, dne 15.5.2012 Podpis avtorja/-ice:_____

Zahvala

Najprej bi se zahvalil mentorju prof. dr. Saši Divjaku, ki je podprl mojo idejo in mi omogočil, da napišem delo iz izbrane teme.

Zahvaljujem se mami, Ireni Kavrečič Holcman za hiter lektorski pregled diplomske naloge.

Za konec pa bi se še zahvalil moji boljši polovici, Ani Simončič, za vsestransko podporo.

Kazalo

POVZETEK **1**

ABSTRACT **2**

1 UVOD **3**

2 KAJ JE BMS? **4**

3 VGRADNJA SISTEMA BMS V ZGRADBO **5**

3.1 STANDARD X10 5
3.2 INSTEON 6
3.3 Z-WAVE 7
3.4 UPB 7
3.5 ZIGBEE 7
3.6 PRIMERJAVA STANDARDOV 8

4 REŠITVE ZA AVTOMATIZACIJE ZGRADB **9**

4.1 HOMESEER 9
4.1.1 Komponente sistema *9*
4.1.1.1 Krmilnik modulov 9
4.1.2 Pregled določenih modulov *10*
4.1.2.1 Modul za luči 10
4.1.2.2 Modul za Ogrevanje / ohlajanje (HVAC) 11
4.1.2.3 Modul za avdio in video opremo 11
4.1.2.4 Modul za varnost in zaščito zgradbe 11
4.1.3 Fibaro - Inovativni moduli *12*

5 PLK KOT NADOMESTILO MODULOV **13**

5.1 SISTEM CYBMS 14
5.1.1 Krmilniki *15*
5.1.2 Skupne lastnosti PCU ter DHU programskega modula *16*
5.1.3 Komunikacija med moduli *16*
5.1.4 Metoda setUpServer() *17*
5.1.5 Delovne spremenljivke *18*
5.1.6 Seznam elementov *20*
5.1.7 Ukazi za medsebojno komunikacijo *21*
5.1.8 Parser ukazov *22*
5.1.9 Identifikacija PM *22*
5.1.10 Uporaba niti *22*
5.1.11 MUTEX *23*
5.2 PODROBNEJŠI OPIS DELOVANJA POSAMEZNEGA PROGRAMSKEGA MODULA 24
5.2.1 Programski modul "PCU" *24*
5.2.1.1 Map addr 24
5.2.1.2 Filter 25
5.2.1.3 Sprejem ukaza REG PLC 25

5.2.1.4 Sprejem ukaza REG VAR 25
5.2.1.5 Sprejem ukaza SET 25
5.2.2 Programski modul "DHU" *25*
5.2.2.1 Povezava s spletni vmesnikom 26
5.2.2.2 Uporaba baze podatkov 26
5.2.2.3 Map_var 27
5.2.2.4 Lookup_var 27
5.2.2.5 Sprejem ukaza GET 27
5.2.2.6 Sprejem ukaza SET 27
5.2.2.7 Sprejem ukaza UPDATE 27
5.3 UPORABNIŠKI VMESNIK 28
5.3.1 Opis menija za upravljanje s sistemom *28*
5.3.1.1 Polje uporabniki 28
5.3.1.2 Polje objekti 28
5.3.1.3 Polje alarmi 29
5.3.1.4 Polji predloge struktur ter predloge prostorov 29
5.3.1.5 Polje Šifranti 29
5.3.1.6 Polje Slike 29
5.3.1.7 Polje dokumenti 29
5.3.1.8 Polje poročilo 29
6 SKLEPNE UGOTOVITVE 30
SEZNAM SLIK IN TABEL 31
SEZNAM SLIK 31
SEZNAM TABEL 31
VIRI 32

Seznam uporabljenih kratic in simbolov

BMS	Sistem za upravljanje zgradb (ang. building management system).
PLK	Programabilni logični krmilnik (ang. programmable logical controller - PLC) je elektronska naprava, ki nadzira senzorje v določenem prostoru ali skupini prostorov in v skladu z vnaprej določenim programov krmili grelna telesa, klimatske naprave, luči, ipd.
PM	Programski modul – ki predstavlja eno komponento sistema in opravlja specifično nalogo.
PCU	Programski modul "plccomm", PLC Communication Unit – za komunikacijo s PLK-ji.
DHU	Programski modul "datahub", Data Hub Unit – logični koordinator operacij.
mySQL	Sistem za upravljanje s podatkovnimi bazami.
GUI	Grafični uporabniški vmesnik.
NAD	Omrežni naslov PLK, ki mu je določen s serijsko številko (ang. Network ADdress)
API	Ang. "Application Programming Interface", to je množica rutin, protokolov ter pripomočkov za izdelavo aplikacij.
CAN	Ang. "Controller Area Network", je protokol namenoma razvit za komunikacijo med mikrokrmilniki, ki temelji na pošiljanju sporočil.

Povzetek

Glavni cilj diplomskega dela je bil razvoj programskega dela rešitev za sistem za upravljanje zgradb ter nato še primerjava sistema s trenutno najbolj aktualnimi rešitvami na trgu avtomatizacije zgradb. V prvem delu so opisani povezovalni protokoli, ki v sistemu za upravljanje zgradb služijo za komunikacijo med napravami v zgradbi. Opis se začne s pregledom standarda *X10,* nato pa mu sledijo še ostali aktualni standardi: *Insteon, Z-Wave, UPB* ter *Zigbee*. Na koncu sledi še skupna primerjava vseh standardov med seboj. V nadaljevanju naloge sledi opis aktualnega sistema za avtomatizacijo zgradb na tržišču, to je *HomeSeer*. Naštete ter opisane so potrebne komponente za delovanje takega sistema ter opis nekaj njihovih najbolj uporabljenih modulov za vodenje naprav. Za primerjavo je dodano podjetje *Fibaro*, ki se je domislilo izdelave novega inovativnega modula. V zadnjem delu diplomske naloge pa je opisana razvita rešitev *cyBMS*. Najprej so opisane komponente sistema, ki jim sledi podrobnejši opis razvitih programskih modulov *PCU* ter *DHU*. Razloženo je njihovo skupno ogrodje in kasneje tudi podrobnejši opis vsakega programskega modula posebej. Kot zaključek je predstavljen *uporabniški vmesnik* in s tem tudi možnosti, ki jih nudi razviti sistem.

Ključne besede:

- avtomatizacija
- moduli
- industrijski programabilni logični kontrolerji
- vodenje
- pametna zgradba

Abstract

The main goal of this thesis was the development of the software solution part for the building management system and the comparison of this system with the most current solutions available on the market for building automation. The first section describes the connectivity protocols that are used for communication between devices in the building with the building management system installed. Description begins with an overview of the *X10* standard and is followed by the remaining protocols: *Insteon*, *Z-Wave*, *UPB* and *Zigbee*. At the end there is a comparison of common features between these protocols. The next section describes one of the current building management systems on the market, the *HomeSeer*. It lists and describes the components needed to make such a system to operate and it also describes some of their commonly used modules for managing devices. For a comparison, the company *Fibaro* and their innovative managing modules are added. In the last part of the thesis there is the description of the developed solution - *cyBMS*. First all of the system components are described. Then a more detailed description of the developed software modules *PCU* and *DHU* follows. First their common framework is explained and then more detailed description of each program module is followed. In conclusion, the *user interface* is presented and thus the capabilities that are offered by the developed system.

Keywords:

- automation
- modules
- industrial programmable logic controllers
- management
- smart house

1 Uvod

Sistem za upravljanje zgradb največkrat srečamo v večjih poslovnih zgradbah. Njegova primarna funkcija je skrb za ogrevanje pozimi ter ohlajanje poleti – vzdrževanje temperature. Sedaj imajo taki sistemi veliko več funkcij, ki so naprednejše in predvsem naredijo tak sistem zanimivejši za vgradnjo v skoraj vsak dom. Ne čudi dejstvo, da se uporaba takega sistema širi tudi na manjše poslovne objekte kot tudi v naprednejša stanovanja. Ljudje postajamo vse bolj zaposleni z ostalimi stvarmi, kot so na primer služba, družina, prostočasne aktivnosti in podobno. V polni zasedenosti našega vsakdanjika in pogoste odsotnosti od doma nam takšni sistemi zagotavljajo kakovostnejše bivanje.

V sodelovanju s podjetjem INDEA d.o.o. smo razvili sistem, ki omogoča upravljanje zgradb na daljavo preko spletnega vmesnika. Celoten sistem je napisan v programskem jeziku C, teče pa na operacijskem sistemu Linux. Trajalo je kar nekaj časa, preden je sistem začel v celoti delovati. Ne zaradi velike kompliciranosti takega sistema, ker to sploh ne velja za tak sistem, temveč zaradi same konsistenčnosti. Prenosi podatkov (branje in pisanje), nastavljanje raznoraznih vrednosti, opomniki, itd. se morajo vedno zgoditi na enak način, da je lahko kot tak sistem sploh uporaben. Pomembno je bilo tudi ravnanje programske opreme ob izpadu električne energije. V kolikor ni sistem, na katerem teče programska oprema, opremljen z UPS-om, se mora sistem ob povrnitvi električnega toka brez kakšnih posebnih težav vrniti v predhodno stanje. Za to je bila potrebna uporaba baze podatkov. Uporabljena je pogosta mySQL baza.

Za doseganje želene stopnje stabilnosti ter konsistentnosti delovanja samega sistema je bilo potrebnih veliko ur testiranja. Za testiranja sistema je bil v programski kodi vključen t.i. "logger", ki je periodično shranjeval v datoteko trenutno stanje sistema, da smo lahko lažje razhroščevali delovanje sistema. Testiranja so bila dolga, sam sistem smo pustili delovati tudi po več dni ob koncih tednov in spremljali napake, ki so se pri tem pojavljale. Iskanje ter detekcija takšnih napak je bila dokaj težavna, saj so se določene napake pojavljale samo ob določeni uri ali pa celo po določenem številu dni delovanja sistema. Ker pa mora biti tak sistem zelo stabilen, smo morali odpraviti vse to.

Programska koda je razdeljena v programske module. Vsak modul skrbi za svojo stran naloge. Določen modul skrbi za komunikacijo s PLK-ji, drugi komunicira s spletnim strežnikom, tretji z bazo itd. Sama razčlemba programske kode na manjše, smiselne celote, nam je dala možnost napisati kodo, ki je berljiva, predvsem pa je lažje predvidevati samo delovanje in posledično je iskanje napak lažje. Z lahkoto bi lahko vse module združili v enega, vendar bi bilo veliko težje ohranjati tako konsistentnost pri delovanju, kot je tu potrebna.

2 Kaj je BMS?

V angleščini je to kratica za "Building Management System", v slovenščini pa "sistem za upravljanje zgradb". Sama kratica se pomensko navezuje na visokotehnološki sistem, ki je načeloma vgrajen v modernejše zgradbe. Slednji pa ima "skrb" za zgradbo, tako da nadzoruje njeno mehansko in električno opremo. Seveda pa mora biti ta sposobna komunikacije s prej omenjenim sistemom. Med to opremo štejemo naslednje:

- osvetlitev
- ogrevanje
- senčenje
- upravljanje klimatske naprave
- požarne sisteme
- varovalne sisteme
- avdio ter video sisteme
- gospodinjske aparate
- namakalne sisteme
- idr.

Kot je razvidno iz seznama, lahko večino naprav avtomatiziramo. Tudi bolj kompliciranе naprave (npr. alarmne sisteme) lahko nadzorujemo. Podpora napravam je iz dneva v dan večja, tako da je res prava redkost, v kolikor obstaja naprava, ki še ne bi bila podprta.

Sistem kot tak je v osnovi sestavljen iz programskega dela na eni strani ter strojnega dela na drugi. Namen takega sistema je avtomatizirati in prevzeti kontrolo nad priključeno opremo ter jo tako uporabljati kar najbolj učinkovito. Lahko rečemo, da gre za neke vrste samostojni računalniški sistem, ki ima vnaprej naložene vrednosti ter prage. Glede na primerjavo med trenutno vrednostjo in nastavljenim pragom nastavlja in spreminja delovanje posameznih naprav priključenih v sistem. Sam nivo kontrole je odvisen predvsem od informacij pridobljenih iz priključenih naprav ter od sistemskih nastavitev, ki določajo, kako naj reagira na spremembe. Akcije, ki se nato izvršijo, so odvisne ali od naravnih pogojev (temperatura, gibanje, svetloba) ali vsiljenih pogojev (urnik, ročni ter daljinski posegi) [1]. Zgradba opremljena s takšnim sistemom lahko porabi manj električne energije, omogoča bolj udobno delo, lahko nudi oddaljen dostop … Možnosti je veliko, zato je sistem kot tak zelo zanimiv ter bo najbrž v bližnji prihodnosti mogoče že kot standard ob gradnji novih, bolj poslovno usmerjenih objektov.

3 Vgradnja sistema BMS v zgradbo

Znano je, da je strošek vgradnje naprednih sistemov v zgradbo, ki omogočajo centralno upravljanje z napravami po zgradbi, trenutno kar velik zalogaj. Že ob sami gradnji zgradbe je pametno razmisliti v naprej in tako vgraditi potrebne sestavne dele, kot so razni krmilniki in dodatne signalne žice. To predstavlja najboljši in na koncu tudi najelegantnejši način, da lahko zgradbo poimenujemo pametna zgradba. Seveda pa to naredi začetno investicijo gradnje zgradbe dražjo, kar pa lahko marsikoga odvrne od tega. Zato dandanes srečujemo zgradbe brez potrebne instalacije, čeprav gre za novogradnje. Na srečo pa tudi v takem primeru obstajajo številne rešitve. Trg so preplavila številna podjetja, ki s pomočjo brezžičnih tehnologij omogočajo avtomatizacijo zgradb. Uporabljeni standardi prenosa signalov med napravami so različni, in sicer vse od najstarejšega standarda X10 do novejših. Ti so bili razviti, da bi zadostovali vse večjim potrebam. To so: INSTEON, Z-Wave, UPB … Potrebno je razmisliti, da moramo vsaki napravi, ki jo želimo upravljati, priskrbeti neke vrste vmesnik oziroma modul, ki se priključi med napravo in omrežno napetostjo. Ta modul pa s pomočjo določenega komunikacijskega standarda omogoča komunikacijo z ostalimi v omrežju ali pa sprejema direktne ukaze iz centralnega krmilnika namenjenega takim modulom.

3.1 Standard X10

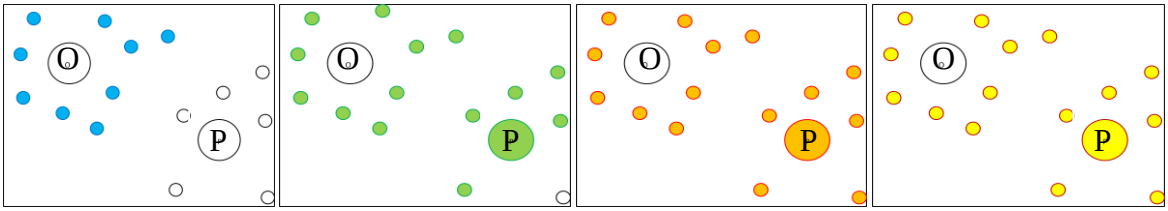
Razvit je bil leta 1975 pri podjetju Pico Electronics. Je mednarodni standard za komunikacijo elektronskih naprav namenjenih avtomatizaciji zgradb. Za delovanje in komunikacijo primarno uporablja omrežno napetost, za komunikacijo z brezžičnimi napravami pa so na voljo tudi kratki radijski signali (RF). Signali so digitalnih oblik. Torej, ko naprava ni priključena v omrežno napetost, komunikacija steče preko radijskih signalov. Primer takih naprav so na primer krmilniki X10 v obliki obeskov, brezžične tipkovnice, protivlomni moduli, itd. Komunikacija preko omrežne napetosti uporablja omrežno frekvenco za oddajo signalov. Vsakič ko ta prečka ničlo, je poslan en bit informacije. Sporočilo je v osnovi sestavljeno iz naslova ter komande. Protokol prenašanja signalov je enak tako za omrežno napetost kot za radijske signale. Po protokolu mora biti sporočilo sestavljeno iz: 4 biti za številko zgradbe, 4 biti za številko naprave ter 4 biti za vrsto ukaza [7]. Bolj enostavne in cenejše X10 naprave omogočajo samo enosmerno komunikacijo. To pomeni, da ob prejetju sporočila, le-tega ne potrdijo, zato veljajo dvosmerne X10 naprave za bolj robustne, vendar je njihova cena vsaj dvakrat višja od enosmernih. X10 standard ima tudi nekaj pomanjkljivosti. Včasih se lahko zgodi, da se določena sporočila izgubijo (v primeru istočasnega pošiljanja), zato velja pravilo, da se lahko pošlje le eno sporočilo naenkrat. Hitrost pošiljanja tipičnega sporočila znaša 0.75 s, kar je lahko v nekaterih primerih prepočasi. Problem je najbolj opazen, če uporabljamo dvosmerne X10 naprave. Skupno število uporabljenih naprav v omrežju je

lahko le 256 (16 x 16). Nazadnje pa velja omeniti še to, da X10 protokol ne omogoča kodiranja signala, kar pomeni, da lahko sosed z enake omrežne napetosti sproži spremembo stanja v kateri izmed naših X10 naprav.

3.2 INSTEON

Enako kot X10 protokol se tudi INSTEON za komunikacijo poslužuje omrežne napetosti ter radijske frekvence. Gre za tehnologijo, razvito posebej za avtomatizacijo domačega doma. Razvili so jo pri podjetju SmartLabs. V kolikor imamo v zgradbi še kakšno napravo, ki podpira le X10 tehnologijo, to ne predstavlja problema, saj INSTEON deluje tudi s starejšimi tehnologijami. Naprava podprta z INSTEON tehnologijo lahko oddaja, sprejema ter ponovno pošlje sprejeti signal. Vse to počne brez potrebe po nekem osrednjem krmilniku. Pomembna lastnost te tehnologije, s katero se loči od ostalih, je ta, da vsaka naprava sprejeto sporočilo ponovno pošlje vsem napravam v omrežju, ne glede na to, komu je bilo sprva namenjeno [6].

Prikaz ponavljanja sporočil:



Slika 1. Zaporedje akcij. O-oddajnik, P-prejemnik.

Zaporedje akcij: Oddajanje, 1. ponovitev, 2. ponovitev, 3. ponovitev.

Tako vsaka dodatna naprava le okrepi pokritost omrežja. Drugi standardi za pošiljanje sporočil večinoma uporabljajo usmerjevalne tabele, kar je v primerjavi z INSTEON-om slabše. Vsaka naprava ima svojo enolično prepoznavno številko (ID). Vsa sporočila so kodirana. To pomeni, da se prenašajo v kodirani obliki, kar naredi sistem varnejši pred vsiljivci. V kolikor želimo naprednejše funkcije (npr. vodenje naprav preko spletnega vmesnika), obstajajo številni krmilniki, ki omogočajo centralno vodenje vseh naprav v zgradbi. To pomeni, da lahko kar preko mobilnega telefona (z internetnim dostopom) krmilimo naprave, čeprav se lahko takrat na primer nahajamo še v službi. S PDA-jem se preprosto preko mobilnega ali brezžičnega omrežja povežemo s centralnim krmilnikom in mu naložimo, kaj mora postoriti. Po sprejetju ukaza krmilnik nato pošlje sporočilo ustreznim modulom.

3.3 Z-Wave

V nasprotju s prejšnjima dvema tehnologijama (X10 in INSTEON) Z-wave ne omogoča komunikacije preko omrežne napetosti, temveč samo preko radijskih valov [6]. Komunikacijski protokol je optimiziran za pošiljanje kratkih sporočil med napravami. Deluje na frekvenci okrog 900 Mhz. Ta višina frekvence je bila izbrana, da pri delovanju ne bi prihajalo do motenj zaradi delovanja drugi naprav v zgradbi. Primer najpogostejših naprav so razna WiFi omrežja, ki delujejo na višjih frekvencah. Za vsako napravo, ki jo želimo voditi s pomočjo te tehnologije, moramo zanjo kupiti neke vrste vmesnik. Nato je potreben še nakup centralnega krmilnika, sicer omrežje ne deluje. Na eno takšno omrežje lahko priklopimo do največ 232 naprav. V primeru, da jih želimo več, dokupimo dodaten krmilnik ter vse krmilnike povežemo in jih tako združimo med seboj. Vsakič, ko dodamo nov modul v sistem, je potrebno le-tega prijaviti v samo omrežje. Postopek običajno zahteva pritisk tipk na vmesniku v določenem zaporedju. Podoben postopek nas čaka takrat, ko želimo določeni modul oziroma napravo odstraniti. Razlog tiči v tem, da Z-wave krmilnik najprej določi jakost signala med napravami in se nato na njeni podlagi odloči, kakšno mrežo bo spletel. Vsaka naprava namreč predstavlja določen vozel v omrežju. Tehnologijo so podprla številna podjetja (preko 150) in tako ustvarila konzorcij.

3.4 UPB

Mnogi ga imenujejo "X10 na steroidih" [11]. Za kratico se skriva ime "Universal Powerline Bus". Razvilo ga je podjetje Powerline Control System (PCS). Prenos sporočil poteka samo preko omrežne napetosti, vendar je v primerjavi z X10 standardom, UPB veliko hitrejši in zmogljivejši. Komunikacija velja za robustno z možnostjo programiranja številnih dodatnih, zelo naprednih funkcij. Čas prenosa sporočila je samo 0.3 s, pri X10 pa znašala 0.75 s. Sporočila so poslana v kodirani obliki, zato je UPB varnejši kot X10, ki pošilja sporočila brez šifriranja. Možna je uporaba UPB vmesnikov popolnoma brez dodatnega krmilnika, vsaj do takrat, dokler je omrežje manjše. Tako je namestitev hitrejša in stroški niso enormni. V kolikor pa je omrežje večje in/ali želimo uporabo naprednejših funkcij, še vedno obstaja možnost nakupa krmilnika. Podobno kot dražje različice X10 naprav, UPB naprave že v osnovi omogočajo dvosmerno komunikacijo. Tako je zagotovljena večja pravilnost delovanja sistema.

3.5 Zigbee

Ta komunikacijski protokol se ponaša predvsem z nizko porabo energije. Uporablja radijske valove, temelji pa na IEEE 802 standardu. Ustvarjen je bil z namenom, da bi bil enostaven ter cenovno ugoden v primerjavi s konkurenti. Sama nizka poraba energije izhaja predvsem iz

tega, da naprave opremljene s tem protokolom večino svojega časa preživijo v t.i. "stand-by" načinu delovanja. To je mogoče, saj naprava za prehod iz mirovanja v delovni način potrebuje le 25 ms. To je kot nalašč namenjeno za prenosne, brezžične naprave, ki se napajajo samo preko baterije. Moduli opremljeni s tem protokolom potrebujejo za svoje delovanje osrednji krmilnik – t.i. koordinator. Topologija omrežja je lahko bodisi zvezda bodisi drevo. V primeru drevesa je krmilnik v vlogi korena. Ostale naprave pa sestavljajo njegove liste.

3.6 Primerjava standardov

Za lažji pregled ter primerjavo protokolov med seboj je v spodnji tabeli primerjava [6] najbolj pomembnih lastnosti, po katerih se razlikujejo.

	X10	**INSTEON**	**Z-wave**	**UPB**	**Zigbee**
Cena	nizka	srednja	srednja	drago	srednja
Zanesljivost	zadovoljiva	zelo dobra	dobra	dobra	dobra
Dodajanje modulov izboljša delovanje omrežja	ne	da	ne	ne	ne
Medij komunikacije	električno omrežje in RF	električno omrežje in RF	RF	električno omrežje	RF
Vozli ponovijo sporočilo	ne	da	ne	ne	ne
Krmilnik neobvezen	da	da	ne	da	ne
Največje število naprav na omrežje	256	16,777,216	232	250	256
Podpira X10	da	da	ne	ne	ne
Topologija	drevo	/	"Source routed mesh"	drevo	zvezda ter drevo
Enkripcija	ne	da	da	da	da

Tabela 1. Primerjava lastnosti protokolov

4 Rešitve za avtomatizacije zgradb

4.1 HomeSeer

Gre za enega izmed vodilnih podjetji za avtomatizacijo domov ter drugih kontrolnih sistemov [8]. Ustanovljeno je bilo leta 1998. S svojimi zelo izpopolnjenimi ter kvalitetno narejenimi produkti postavljajo mejnike drugim konkurentom. Njihov sistem je zelo enostaven za uporabo, saj lahko brez večjih težav krmilimo skoraj vsako napravo v stanovanju. Sistem HomeSeer omogoča oddaljen dostop do centralnega krmilnega sistema, kar pomeni, da lahko kar preko spletnega brskalnika upravljamo z napravami v naši zgradbi ali pa samo pregledujemo stanje. S pomočjo makrov lahko ustvarimo zaporedje akcij, ki naj se zgodijo ob določenem času ali pa ob določenem pojavu. Tako lahko na primer, ko prispemo domov, senzor gibanja zazna avtomobil ter sproži naš vnaprej nastavljen makro, ki nato vklopi vhodne luči ter pošlje signal garažnim vratom, da se odprejo. Sistem HomeSeer je združljiv z velikim številom povezovalnih standardov, kot so X10, INSTEON, Z-Wave … Upravljanje sistema je možno tudi preko pametnih mobilnih telefonov z operacijski sistemi Android ter iOS.

Za vzpostavitev takega sistema moramo za vsako elektronsko napravo, ki jo imamo v zgradbi, kupiti modul, ki se v večini primerov priklopi med napravo in omrežno napetostjo. Vsak tak modul ponuja več funkcij delovanja. V kolikor je to navadna žarnica za osvetljevanje prostora, lahko poleg najbolj osnovnega vklop/izklop luči uporabljamo tudi t.i. dim funkcijo, ki omogoča nastavitev svetlosti same žarnice. Moduli za HVAC (heating, ventilation, and air conditioning) naprave omogočajo številne naprednejše funkcije. Najbolj osnovna funkcija je vzdrževanje temperature. V primeru izredno nizke ali izredno visoke temperature nam lahko preko elektronske pošte ali SMS sporočila pošljejo sporočilo. Ob primeru zaznave požara ali dima se sistem HVAC izključi. Vse, kar je potrebno za delovanje, je primeren modul ter določen set makrov, ki sproži te akcije.

4.1.1 Komponente sistema

4.1.1.1 Krmilnik modulov

Za delovanje vseh teh modulov je potreben neke vrste krmilnik. Na izbiro imamo dve različni možnosti. Lahko se odločimo samo za nakup programske opreme, ki deluje na našem najbolj uporabljenem operacijskem sistemu, ali pa kupimo celotno elektronsko napravo. Pri napravi gre za krmilnik, na katerem teče podobna programska oprema, ki je na voljo ločena, vendar s to razliko, da ima tak krmilnik več naprednejših funkcij kot samostojna programska oprema.

Samostojni krmilnik je bolj kompaktna rešitev, saj je po dimenzijah zelo majhen in tako primeren za postavitev v skoraj vsako sobo. Najzmogljivejši ima tudi daljinski upravljalec.
Lastnosti HomeSeer PRO 100 - S3 krmilnika:

- Konfiguracija in spremljanje ter dodeljevanje ukazov poteka preko spletnega brskalnika.
- Možnost dostopa preko PDA-jev na za to prirejeno spletno stran.
- Funkcija HSProtect skrbi za varnost, preprečuje morebitne vdore v operacijski sistem.
- Funkcija HSSentry avtomatsko "resetira" krmilno enoto v primeru zaznave strojne ali programske težave.
- Možnost oddaljenega dostopa.
- Zelo majhna poraba, okrog 10 W pod normalnimi pogoji delovanja.
- Zelo majhna velikost: 24 cm * 18 cm * 5 cm.

Programsko opremo je mogoče prirediti svojim željam. Na voljo je sistemski API. Ob pregledu dokumentacije je možno narediti svoj "plug-in" in tako bodisi razširiti možnosti delovanja sistema ali pa mogoče samo spremeniti videz okolja.

4.1.2 Pregled določenih modulov

Za napravo, ki jo želimo krmiliti, je potrebno poiskati pripadajoči modul. Brez primernega modula ne moremo uporabljati vseh funkcij, ki jih omogoča dotična naprava, kaj šele kakšnih naprednejših. Najosnovnejši modul omogoča vklop/izklop in funkcijo "dim". Pri podjetju HomeSeer imajo zanimiv slogan, ki se glasi: "Česa ne moreš krmiliti s pomočjo HomeSeer modulov?" [9].

4.1.2.1 Modul za luči

Podpira skorajda vsako žarnico, od navadnih sobnih pa vse do raznih reflektorjev.

Omogoča:

- Upravljanje z lučmi, kjerkoli se nahajate s pomočjo vašega pametnega telefona.
- Avtomatski vklop in izklop luči ob naravnih dogodkih (sončni vzhod, sončni zahod).
- Dim funkcijo.
- Vklop luči v sili.
- Kontrolo luči s pomočjo glasu.

4.1.2.2 Modul za Ogrevanje / ohlajanje (HVAC)

Modul upravlja gretje in ohlajanje prostorov po zgradbi s pomočjo dodatnih komunikacijskih termostatov.
Omogoča:

- Ohranjanje želene temperature v prostoru.
- Nastavitev dnevnih ali tedenskih urnikov (s pomočjo krmilnika).
- Avtomatski izklop puhala v primeru nevarnosti ali požara.
- Možnost uravnavanja termostatov s pomočjo glasu.

4.1.2.3 Modul za avdio in video opremo

S pomočjo modula na infrardeče žarke (IR) lahko krmilimo avdio in video opremo v zgradbi.

- Kontrola AV sistemov kar preko PDA-jev.
- Avtomatski vklop na priljubljen televizijski program, ko se le ta prične.
- Avtomatski izklop AV opreme, ko v sobi ni nikogar (s pomočjo senzorja gibanja).
- Avtomatski vklop AV opreme za simulacijo oseb v zgradbi (primerno za roparje).
- Možnost glasovnega upravljanja.

4.1.2.4 Modul za varnost in zaščito zgradbe

Za boljšo varnost HomeSeer omogoča krmiljenje raznih alarmnih sistemov. Vključitev takega modula v sistem omogoča vrsto novih, zanimivih možnosti:

- Vklop in izklop alarmnega sistema s pomočjo PDA-ja.
- Avtomatski izklop alarmnega sistema ob vpisanem PIN-u.
- Avtomatski vklop alarmnega sistema, ko ni nikogar v zgradbi.
- V primeru kakršnekoli zaznave pošlje elektronsko sporočilo ali SMS.
- Upravljanje določenih možnosti kar preko glasu.
- Odklepanje ter zaklepanje vrat na daljavo.
- Oddaljeno odklepanje vrat v primeru prihoda delavca ali čistilke.
- Zagotovitev, da so vrata zaprta, preden se jih zaklene.
- Avtomatsko odklene vsa vrata v primeru nevarnosti ali v sili.
- Možnost obvestila preko SMS v primeru odprtja določenih vrat.

V ponudbi je še veliko modulov. Opisani so samo najbolj pogosto uporabljeni ter najbolj uporabni moduli. Na voljo je še modul za zalivanje vrta, za porabo vode, za porabo električne energije, za detekcijo gibanja in okupacije, za spremljanje temperature, vlažnosti, svetlobe, za videonadzor …

4.1.3 Fibaro - Inovativni moduli

Pri podjetju **Fibaro** gre za manjšo skupino mladih raziskovalcev, ki je sprva dodobra preučila sam trg avtomatizacije zgradb. Dobri dve leti so preučevali dotedanje rešitve in ugotavljali, kako jih izboljšati. Njihova prva prodajalna produktov je s prodajo pričela leta 2011.

Slika 2. Modul Fibar

Prvi produkt, ki so ga splavili na trg, so bili moduli, ki so nujno potrebni za krmiljenje naprav. Do sedaj so pri vseh ostalih podjetjih moduli bili fizično narejeni na način, da so se priklopili med vtičnico naprave ter samim napajalnim kablom naprave. **Fibaro** se je domislil izdelave miniaturnega modula [10], velikosti 40 mm * 36 mm * 15 mm, tako da ga je možno skriti v samo steno zgradbe, in sicer kar za vtičnico. Taka "skrita" namestitev modulov omogoča uporabniku ohraniti videz navadne vtičnice in tako ne posega v sam dizajn notranje opreme oziroma prostora. Resnično ni najbolj estetsko, če vidimo module na vsaki vtičnici.
Sam modul je sestavljen iz večplastnega PCB-ja in čeprav gre za izredno majhne velikosti, kar pomeni, da so pri sami izdelavi imeli težave z raznoraznimi interferencami, je tak modul praktično ekvivalenten ostalim "zunanjim". Prikrajšana mu ni prav nobena funkcija. Poleg revolucionarnega modula ima podjetje v ponudbi še vse ostalo, kar "pride zraven". Centralni krmilnik se imenuje Fibaro HomeCenter2, ki ima naslednje dobre lastnosti:

- Pametna strojna arhitektura - najhitrejša naprava od vse konkurence.
- Enostaven, uporabniku prijazen modul.
- Hitra in enostavna začetna konfiguracija.
- Geo lokalizacija.
- Zelo majhna poraba energije.
- Zgodovina dogodkov.

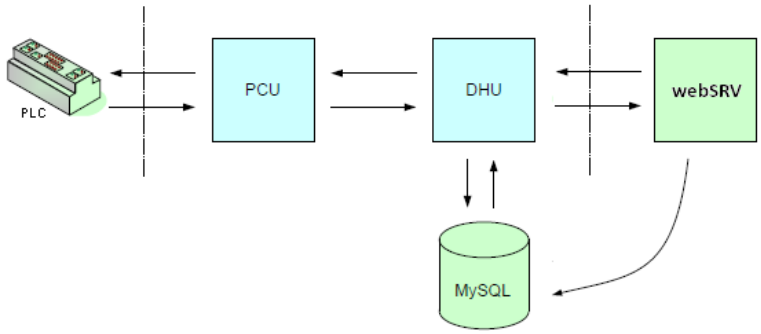
5 PLK kot nadomestilo modulov

Postavljanje dodatnih elementov (moduli) v zgradbe je morda v začetku smotrno dejanje in opravičuje sam nakup, vendar ima tudi to svoje pomanjkljivosti. V kolikor želimo avtomatizirati le del naprav v zgradbi in nimamo v mislih nakupa modula prav za vsako napravo ter ne razmišljamo o kakih naprednejših funkcijah in stoodstotni stabilnosti sistema, potem se lahko lotimo vgradnje. Če samo pomislimo, da za vsako, ampak res vsako napravo potrebujemo svoj modul, število vseh kaj hitro naraste na 30 naprav ali celo več. Odvisno od same velikosti zgradbe in njene namembnosti. Vsak tak modul namreč povprečno stane med 30 € in 50 €. Za kompleksnejše naprave pa še veliko več. Če pomnožimo povprečno ceno modula s pričakovanim številom naprav, ki jih želimo krmiliti, skupni strošek ni več zanemarljiv. Poleg stroška pa imamo tako ob vsaki napravi še eno napravo (modul) več, kar pomeni, da je verjetnost, da se bo kaj pokvarilo ali začelo narobe delovati veliko večja. Večina modulov omogoča krmiljenje preko radijskih frekvenc, kar pomeni, da lahko malo bolj iznajdljivi nepridipravi kaj hitro prevzamejo kontrolo nad našimi napravami in tako ogrozijo celotno delovanje. Z odločitvijo, da bi avtomatizirali zgradbo s pomočjo modulov, se posredno podredimo tudi standardnemu dizajnu teh modulov. V primeru lepše urejenih prostorov, modul bele barve, postavljen med vtičnico in kablom naprave pokvari dizajn ter tako sam izgled notranje ureditve prostora. Da ne pozabimo na centralno vodenje vseh modulov – krmilnik. Cena le-tega kaj hitro preraste 1000 €, zmogljivejši so ocenjeni celo nad 3000 € [3].

Hitro je opaziti, da je za resnejše, večje ter bolj kompleksne sisteme potrebno ubrati drugi pristop. Ob sami gradnji se je potrebno vprašati, ali želimo imeti avtomatizirano delovanje zgradbe ali ne. Veliko bolje je vgraditi ter tako integrirati ta sistem že ob gradnji. Po omrežju je potrebno speljati dodatne signalne žice poleg napajalnih. Vse skupaj pa mora biti speljano v t.i. kontrolno omarico. V njej se nahaja eden ali več specializiranih krmilnikov – PLC. Večje poslovne zgradbe lahko imajo takih krmilnikov po 50. Vsak PLC je sposoben krmiliti določeno število naprav, odvisno od modela. Povprečno pa vsaj okrog 20 navadnih ter po 5 bolj kompleksnih naprav. Ko imamo v zgradbi več naprav, preprosto zvežemo dodaten PLC paralelno. Vse krmilje se tako nahaja v kontrolni omarici in tako ni več tistih modulov. Vse je tudi skrito očem. Taki sistemi se uporabljajo v poslovnih zgradbah ter novogradnjah. Veljajo za veliko bolj stabilne, saj so krmilniki narejeni za industrijsko uporabo, ni nepotrebnega omrežja modulov, vse je vodeno centralno. Sistem tako deluje hitro in zanesljivo. Programska oprema v operacijskem sistemu krmilnika skrbi za vodenje naprav po navodilih oziroma željah uporabnika. Slednji lahko vodi sistem preko za to narejene krmilne plošče ali pa preko programske opreme na računalniku. Krmilniki imajo po večini še dodaten omrežni vhod.

5.1 Sistem cyBMS

Sama rešitev, ki smo jo razvijali, se imenuje po imenu uporabljenega krmilnika ter imenu sistema za upravljanje z zgradbami – **Cy**bro **B**uilding **M**anagement **S**ystem. Rešitev je bila izdelana pod vodstvom podjetja INDEA d.o.o.



Slika 3. Shema sistema cyBMS

Iz slike je razvidno, da se na eni strani nahaja PLK, na drugi strani pa spletni strežnik. Za komunikacijo med tema dvema stranema pa skrbita dva programska modula. Programski modul PCU in DHU [2]. Rešitev, ki smo jo opisali ter hkrati tudi razvijali, zavzema vse tisto, kar se nahaja desno od PLK-ja. Torej oba programska modula ter spletno stran, ki je nameščena na spletni strežnik. Krmilnik je imel že vnaprej naloženo potrebno programsko kodo, tako da se jih je v osnovi moralo samo nastaviti glede na okolje, v katerem so se znašli. To zavzema nastavljanje parametrov ter spreminjanje raznih mejnih vrednosti. Nameščena koda vsebuje že vse funkcije, ki so potrebne za krmiljenje naprav priključenih nanj. Glavna naloga je torej bila prenos informacij ter podatkov iz spletnega strežnika na ustrezne PLK-je ter tudi v obratni smeri, prebrane podatke prenesti v spletni strežnik za poznejše obdelave in preglede.

Programski modul DHU deluje na višjem nivoju – logičnem, dostopa do baze podatkov in nudi podatke spletnemu servisu. Njegova naloga je, da hrani vrednosti spremenljivk, skrbi za "logiranje" in proženje alarmov. S tem ko hrani trenutne vrednosti spremenljivk, omogoča hitrejše poizvedbe, saj odpade potreba po vsakokratnem dostopanju do krmilnikov, predvsem v primeru več sočasnih uporabnikov portala. Programski modul PCU je namenjen komunikaciji s krmilniki in kot sam deluje na nivoju objekta v absolutnih koordinatah (NAD + lokacija v pomnilniku PLK) in ni povezan z bazo. Velja pravilo, da lahko en DHU komunicira z več PCU-ji. Obratno naj načeloma ne bi bilo možno. Oba programska modula DHU ter PCU sta napisana v programskem jeziku C ter tečeta na operacijskem sistemu Linux. Za slednji operacijski sistem smo se odločili predvsem zaradi cene licenciranja in stabilnosti sistema. Programski jezik C pa smo izbrali zaradi že razpoložljive knjižnice in prenosljivosti

na druge platforme. Kasneje se je izkazalo, da bi bilo verjetno bolje izbrati C++ zaradi večje preglednosti objektne izvorne kode.

5.1.1 Krmilniki

V svetu industrijske avtomatike že vrsto let prevladujejo programabilni logični krmilniki (PLK). Gre za posebno vrsto mikroračunalnikov, ki krmilijo naprave v raznih industrijskih procesih. Razpolagajo z določenim številom digitalnih ter analognih vhodov in izhodov. Naprave, ki jih želimo krmiliti, priklopimo na sam krmilnik preko vnaprej pripravljenih vhodov ter izhodov. Kanalov je več. Za krmiljenje priključenih naprav pa PLK-ji vsebujejo pristojen program oziroma t.i. krmilno kodo. Slednja se izvaja znotraj krmilnega cikla. Ta krmilni cikel je kombinacija končnega algoritma in krmilnika, ki ta algoritem izvaja v neskončnost. Razumemo lahko, da si vsaka, na krmilnik priključena naprava lasti določen prostor v krmilnem ciklu. V tem prostoru pa se nahaja njena krmilna koda, ki upravlja z napravo ter javlja povratne informacije nazaj v krmilnik.

Za naš projekt smo izbrali PLK, ki ga izdeluje slovensko podjetje Robotina. Pri izbiri ni šlo zgolj za naključje, temveč za to, da podjetji Indea ter Robotina sodelujeta med seboj. Izbran je bil model družine CyBro [5]. Sam krmilnik je grajen na modularen način, kar se od takega krmilnika tudi pričakuje. Krmilnik omogoča tudi priključitev raznih dodatnih razširitvenih modulov (senzor temperature in vlage, relejna enota, senzor osvetlitve, modul za sobo, vremensko postajo ...). Večinoma so to dodatni vhodi in izhodi, obstaja pa tudi priročen LCD prikazovalnik, preko katerega lahko upravljamo s krmiljenjem. Seveda je bolj namenjen končnemu uporabniku kot pa samemu razvoju. Na sistem se lahko priključi tudi zaslon občutljiv na dotik, na katerem se prav tako spremlja in nastavlja določene parametre.

Slika 4. Krmilnik

V primeru, da se znajdemo v situaciji, kjer imamo za krmiljenje veliko množico senzorjev, stikal, ipd., se lahko kaj hitro zgodi, da samo en krmilnik ne zadostuje, kljub temu da omenjeni krmilnik ni tako zelo nesposoben. V takšnem primeru se preprosto uporabi več krmilnikov, ki jih povežemo med seboj. Kot primer naj navedem, da za krmiljenje vseh

prostorov večjega poslovnega objekta lahko naš razviti sistem cyBMS uporablja preko 50 takšnih krmilnikov. Komunikacija med PLK cyBro in moduli, ki so vezani na krmilnik, poteka po protokolu CAN. Za komunikacijo med samimi krmilniki pa se uporablja protokol Ethernet TCP/IP. Programski jezik (ter struktura jezika), s katerim se ga programira, je zelo podoben Pascal-u. Vsak krmilnik ima svojo enolično identifikacijsko številko NAD. Uporabi se jo za naslavljanje točno določenega krmilnika. Uporabna je predvsem takrat, ko se v sistemu uporablja več krmilnikov hkrati. Izhaja še iz časov, ko so bili krmilniki povezani z RS485 vmesnikom.

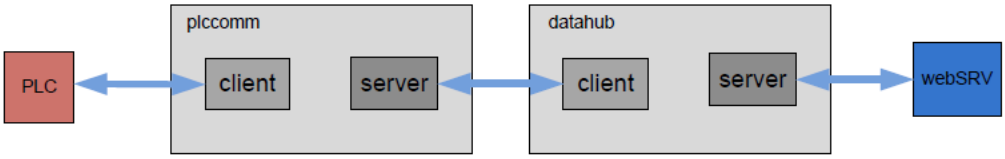
Slika 5. Krmilnik z zaslonom

5.1.2 Skupne lastnosti PCU ter DHU programskega modula

Programska modula (PM) si v dobršni meri delita skupno ogrodje. Razlog je ta, da sta bila pisana vzporedno in sta bila v povojnih preizkusih celo identična, saj se je najprej usposabljalo komunikacijo med moduloma ter zunanjim svetom. Modula uporabljata tudi podobne funkcije, procedure ter knjižnice. Seveda te iste funkcije opravljajo čisto drugačne naloge na svojstven način, čeprav je osnova enaka. Če strnemo vso zadevo, lahko rečemo, da je programski modul po imenu DHU tako bolj prilagojen komunikaciji z uporabniškim vmesnikom, PCU pa komunikaciji s samim krmilnikom(-i).

5.1.3 Komunikacija med moduli

Oba programska modula komunicirata med seboj, prav tako vsak izmed njiju komunicira še z zunanjim modulom. V splošnem zgleda povezovalna shema modulov nekako tako:



Slika 6. Povezovalna shema modulov

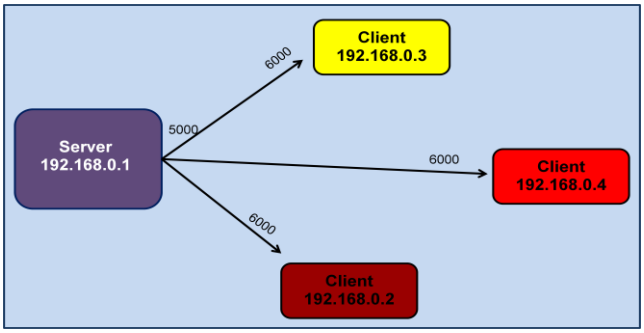
Na sliki sta v vsakem modulu vrisana po dva pravokotnika, ki nakazujeta, da tako eden kot drugi programski modul uporabljata strežniški ter klientni proces. Strežniški proces se

vzpostavi v metodi **setUpServer()**. DHU lahko sprejme več povezav s strani spletnega servisa, PCU pa je lahko povezan z več PLK-ji hkrati. Skoraj bi lahko rekli, da gre za zrcalna dejanja.

5.1.4 Metoda setUpServer()

Uporablja se generičen TCP strežnik. Je prirejen specifiki uporabljenih zahtev. Direktna komunikacija poteka preko protokola TCP/IP. V metodi se konfigurira strežnik, ki bo nato čakal na sprejem klientov na določenih vratih. Najprej se torej ustvari TCP vtič, potem se ta vtič veže na določena vrata, kjer strežnik posluša.

Slednji generični model "client-server" deluje po protokolu TCP/IP "unicast", kar pomeni, da se stvari pošiljajo izključno na točno določen IP ter pripadajoča vrata.



Slika 7. Shema modela "client-server"

Glavni koraki metode:

- Uporablja protokol TCP / IP.
- Ustvari se TCP vtič.
- Določi se številka vrat.
- Sprejema vsako novo povezavo.
- Vse se zapiše v dnevnik – log.
- Zažene se zanka.
- Na določen interval se preverja nove povezave.
- Ob primeru nove povezave se ustvari nova nit.
- Vsak klient je svoja nit.

5.1.5 Delovne spremenljivke

To so spremenljivke, ki predstavljajo imena senzorjev, aktuatorjev tj. tistih naprav, s katerimi imamo dejansko opravka. Iz njih se bere ter krmili preko številnih PLK-jev. Spremenljivke, s katerimi upravljata programska modula, se delita na dve skupini: logične in absolutne spremenljivke. Kot že omenjeno, ta delitev izhaja iz tega, ker programski modul DHU deluje na logičnem, PCU pa na absolutnem nivoju oz. prostoru.

Za lažje razumevanje vzamemo na primer senzor temperature v zgradbi s številko 1 na parceli s številko 2:

Logična spremenljivka	*Absolutna spremenljivka*
Parcela2. zgradba1.temp1	Cybro-3301.temp[1]

```
/*
     map_var

     pretvori logicno spremenljivko v absolutno = DB --> PLC
*/
int map_var(char *logvar, char *absvar)
{
    char q[500];
    int ret=0;

    MYSQL_RES *res;
    MYSQL_ROW row;

    strcpy(absvar,"*");

    sprintf(q,"SELECT plcvar FROM V_nametrsf WHERE logvar='%s' COLLATE UTF8_BIN",logvar);
    ret = execute_SQL(q,&res);

    if(!ret)
    {
       if ((row = mysql_fetch_row(res)) != NULL)
          strcpy(absvar, row[0]);
       else
          ret = 3;
    }

    mysql_free_result(res);
    if(ret) slprintf(LOG_ERR,"SQL error(%d):\n%s\n",ret,q);
    return ret;
}
```

Slika 8. Metoda map_var

V grobem lahko rečemo, da absolutna spremenljivka vsebuje ime PLK-ja ter njegovo identifikacijsko številko - štirimestno NAD številko. Poleg tega pa vsebuje še pot oziroma interno/lokalno ime spremenljivke dotičnega modula. Ker med DHU-jem ter PCU-jem poteka komunikacija, se pred vsakim pošiljanjem delovnih spremenljivk le-te pretvorijo. Vse

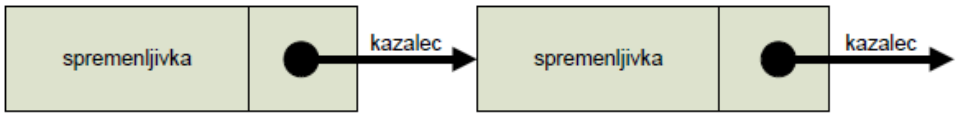
pretvorbe spremenljivk se izvajajo znotraj modula DHU. Pred pošiljanjem se v metodi MAP_VAR pretvorijo logične spremenljivke v absolutne.
Ob prejemanju pa se obratna pretvorba izvede v metodi LOOKUP_VAR.

```
/*
    look_var

    pretvori absolutno spremenljivko v relativno
*/
int look_var(char *absvar, char *logvar)
{
    char q[2048],rs[50];
    int ret=0, n;

    MYSQL_RES *res;
    MYSQL_ROW row;

    strcpy(logvar,"");

    sprintf(q,"SELECT logvar FROM V_nametrsf WHERE plcvar='%s' COLLATE UTF8_BIN",absvar);
    ret = execute_SQL(q,&res);

    if(!ret)
    {
        n = 0;
        row = mysql_fetch_row(res);
        while(row)
        {
            strcpy(rs, row[0]?row[0]:"");
            if(rs[0])
            {
                n++;
                if(logvar[0]) strcat(logvar,";");
                strcat(logvar,rs);
            }
            row = mysql_fetch_row(res);
        }
        ret = n;
    }
    else
    {
        slprintf(LOG_ERR,"ERR - look_var SQL: %s",mysql_error(conn));
    }
    mysql_free_result(res);
    return ret;
}
```

Slika 9. Metoda look_var

5.1.6 Seznam elementov

Za potrebe shranjevanja ukazov ter imen delovnih spremenljivk oba PM-a uporabljata seznam, ki je napisan prav posebej za delo z dotičnimi spremenljivkami. Poleg osnovne shrambne funkcije vsebuje seznam še dodatne funkcije in procedure, ki olajšujejo delo s takšnim seznamom.



Slika 10. Seznam s kazalci

Iz slike se lahko kaj hitro ugotovi, da gre za zaporedni seznam elementov, kjer je vsak element sestavljen iz dveh delov. Prvi del je namenjen shrambi delovne spremenljivke ter tudi njene vrednosti in tipa ukaza. V ta namen so definirane tri spremenljivke z omejeno dolžino tipa **char**. Drugi del pa je kot običajno pri zaporednih seznamih uporabljen za kazalec, ki kaže na naslednji element v seznamu.

Na sliki je prikazana koda strukture, opaziti je deklaracije funkcij ter procedur, ki so napisane za delo s seznamom. Za dodajanje elementa v seznam je napisana funkcija **sps_add**, ki znotraj kliče funkcijo **sps_last**, katera vrne kazalec na zadnji element v seznamu, da ve, kam dodati element. V primeru, da želimo najti določeno spremenljivko, uporabimo funkcijo za iskanje **sps_findVar**. Napisane so tudi procedure za brisanje elementa (**sps_delete**), spreminjanje vrednosti elementa (**sps_change**) in dodatne procedure, ki so (bile) namenjene bolj testiranju delovanja pri samem razvoju sistema - **sps_print** ter **sps_printAll.**

```
1  #ifndef __ELEMNTI_H__
2  #define __ELEMNTI_H__
3
4  #include <stdio.h>
5  #include <stdlib.h>
6  #include <string.h>
7
8  struct element{
9      int vrednost;
10     char cmd[10]; // ukaz
11     char var[50]; // spremenljivka
12     char val[10]; // vrednost
13     struct element* next;
14 };
15
16 typedef struct element* elementp;
17
18 struct element *zacetek;
19
20 elementp sps_last(elementp p);
21 elementp sps_add(elementp p, elementp *z);
22 elementp sps_findVar(char*varname, elementp *z);
23 void sps_change(elementp p, char *c,char *vr,char *vl);
24 int sps_delete(elementp p, elementp *z);
25 void sps_print(elementp q);
26 void sps_printAll(elementp *z);
27
28 #endif
```

Slika 11. Metode seznama elementov

5.1.7 Ukazi za medsebojno komunikacijo

Komunikacija poteka z besedilnimi sporočili. Tako sporočilo je v splošnem sestavljeno iz treh delov: ukaza, delovne spremenljivke ter vrednosti delovne spremenljivke. Obstajajo tudi izjeme. Za vsa poslana besedilna sporočila veljajo naslednja pravila [2]:

- Vsako sporočilo se mora zaključiti s CR (ASCII 13).
- Sporočilo ima kontrolni in podatkovni del.
- V kontrolnem delu so ukazi in osnovni parametri ločeni s presledkom v eni vrstici.
- Podatkovni del je lahko razdeljen na več vrstic.
- Vrstice ločimo z LF (ASCII 10).
- Rezultati se začnejo z nizom OK ali ERR.

Spodnja tabela vsebuje vse definirane ukaze ter doda še primer uporabe.

Ukaz	**Namen**	**Primer**
GET	Poizvedba vrednosti spremenljivke	GET parcela1. Zgradba 2.temp1
SET	Nastavitev vrednosti spremenljivke	SET parcela1. zgradba1. ogrevanje 22
REG	Registracija nove spremenljivke	REG parcela1. zgradba1. stikalo 1
UPDATE	Osvežitev vrednosti spremenljivke	UPDATE parcela1. zgradba1. ogrevanje 23
USER	Pošiljanje upo. imena	USER Janko
PASS	Pošiljanje gesla	PASS geslo
EXIT	Prekinitev povezave	EXIT

Tabela 2. Prikaz definiranih ukazov ter primeri uporabe

V komunikaciji, ki poteka med programskima moduloma, se uporablja celoten nabor ukazov s to razliko, da DHU ne uporablja ukaza UPDATE, PCU pa ne uporablja ukaza REG. Med programskim modulom DHU in spletnim vmesnikom pa sta uporabljena samo ukaza GET in SET.

5.1.8 Parser ukazov

V večini programov se uporabljajo t.i. "parserji" – ločevalniki nizov. Tudi predstavljeni primer ni izjema, saj tudi sam vsebuje ločevalnik nizov določene vrste. Aktiven je takrat, ko se ukazi prenašajo med programskima moduloma. Le-ti gredo najprej skozi proceduro *parseParameters,* ki dani niz razdeli na posamezne dele po določenih pravilih.

Primer ukaza:
UPDATE parcela1.zgradba1.ogrevanje 23

Zgornji ukaz procedura najprej razdeli na tri dele, nato posamezen del shrani v dvodimenzionalno polje z ukazi. Ker mora veljati pravilo enoličnosti, obstaja tudi funkcija, ki je sposobna pretvoriti razdeljene dele nazaj v splošen ukaz.

5.1.9 Identifikacija PM

Pred začetkom vsake komunikacije se izvede identifikacija – preverjanje pristnosti klienta, da se pravilno upošteva določene omejitve pri dostopanju, branju in nastavljanju parametrov. Na strani strežniškega procesa se preverjajo pravice na osnovi zapisov v bazi podatkov. Ko se želi klient (DHU, PCU, GUI) prijaviti na strežniški proces, pošlje ukaz USER, ki mu takoj sledi uporabniško ime in ukaz PASS, ki mu takoj sledi geslo.
Pravilno sestavljeno besedilno sporočilo je prikazano na spodnjem primeru. Kot že omenjeno, je na koncu vsakega ukaza znak ASCII 13.

Primer:
USER Janko
PASS ******

V primeru, da se tako uporabniško ime kot geslo na strani strežnika ujemata, lahko tedaj uspešno prijavljeni klient upravlja s sistemom, seveda z izstavljanjem ukazov GET, SET, itd. V nasprotnem, negativnem primeru, pa strežnik pošlje rezultat "ERR" – kot string ter takoj prekine povezavo.

5.1.10 Uporaba niti

Za lažjo modularnost oziroma lažje razumevanje delovanja programskih modulov kar oba PM-ja uporabljata po svoj set niti. PM DHU vsebuje nit, ki skrbi samo za povezavo med njim ter PCU-jem. Njene osnovne naloge so vzpostavitev povezave, prevedba pristnosti ob vzpostavitvi povezave ter v primeru izgube povezave ponovno poizkušanje vzpostavitve

povezave. Za oba PM-ja pa je značilna skupna nit, ki čaka na nove povezave in ob vsaki na novo vzpostavljeni povezavi kreira novo nit, ki nato z njo rokuje. Vsaka, tako na novo nastala nit, skrbi za komunikacijo ter prenos informacij med procesi. Druga skupna nit pa ima nalogo, da periodično bere ukaze iz seznamov ter jih izvršuje. Uporaba niti je poleg vsega ostalega poenostavila še nadzor nad obnašanjem programske kode ter omogočila večjo razširljivost.

5.1.11 MUTEX

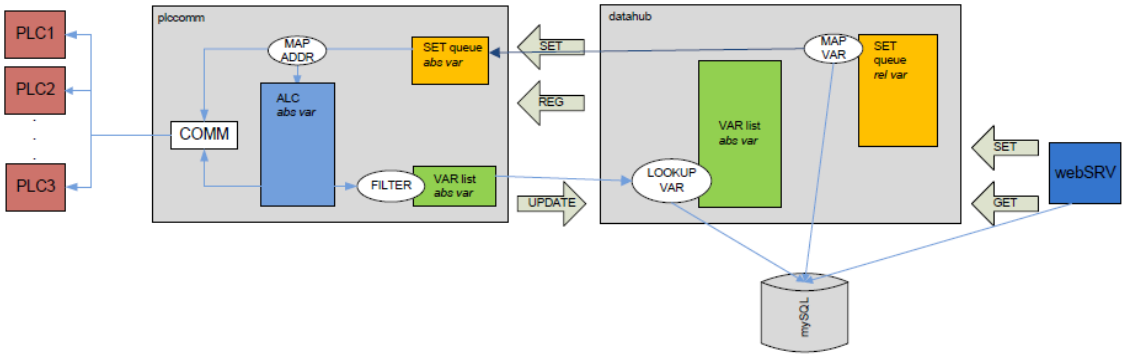
Zaradi uporabe niti v PM prihaja do težav z dostopanjem do skupnih resursov. Dve niti na primer ne smeta istočasno dostopati do seznama delovnih spremenljivk. Podobno je s preverjanjem gesel pri preverjanju pristnosti. Za rešitev teh problemov smo izbrali in nato uporabili semafor tipa "mutex", ki poskrbi, da prva nit, ki je prišla na vrsto zaklene objekt, ki ga potrebuje, ostale niti pa morajo počakati, da se dostop do objekta sprosti. Gre za dokaj preprosto obliko zaklepanja skupnih resursov.

```
1  #define MUTEX_IN  pthread_mutex_lock(&mymutex);
2  #define MUTEX_OUT pthread_mutex_unlock(&mymutex);
3
4  ...
5
6  MUTEX_IN
7  elementp p = listadd;
8  while(p)
9  {
10     if(!strcmp(avn,p->absvar))
11     {
12         // update value
13         sps_change(p,"UPD",p->var,val,typ);
14
15         // log by logname and plcname
16         if(!log_abs) log_var(p->var,p->absvar,val); // zapise vrednost v zgodovino
17     }
18     p = p->next; // naslednja
19 }
20 MUTEX_OUT
```

Slika 12. Primer semaforja

Na sliki je primer zaklepanja kritičnih delov kode, in sicer gre za primer spreminjanja vrednosti v seznamu. Modul želi spremeniti določeno vrednost elementa v seznamu s pomočjo procedure "*sps_change*". V primeru, da ne bi obstajale niti, ne bi bilo možnosti do istočasnega dostopa do seznama. Ker pa uporabljamo niti, je potrebno seznam pred istočasnim dostopom zavarovati. Tako mora vsaka nit z makrom "MUTEX_IN" trenutno zakleniti dostop do seznama ter ob spremembi vrednosti elementa (v tem primeru) seznam tudi odkleniti z makrom "MUTEX_OUT". Nujno je poskrbeti, da nit ne pade v neskončno zanko, ko je dobila vstop do seznama, saj bi tako onemogočila dostop vsem ostalim.

5.2 Podrobnejši opis delovanja posameznega programskega modula



Slika 13. Prikaz posameznih modulov

5.2.1 Programski modul "PCU"

Zadolžen je za komunikacijo s PLK-ji. Torej je to zadnja postaja na poti webSRV, ki skrbi tudi za njihovo medsebojno komunikacijo (v primeru, da jih je več). Ob samem zagonu PM-ov se poveže z vsemi priključenimi PLK-ji in prebere spisek ter naslove spremenljivk in njihove vrednosti. Vse prebrane naslove shrani v seznam ALC, kjer se nato nahajajo vse spremenljivke od vseh, na PCU priključenih PLK-jev. Zadeva je rešena tako, da si PCU za vsak priključen PLK ustvari novo nit in znotraj ustvarjene niti operira z določenim PLK-jem. Seznam delovnih spremenljivk VARlist osvežuje ciklično z intervalom dolžine 1s, tako da seznam predstavlja dejansko stanje senzorjev, stikal ipd. po stanovanju, poslovnem ali industrijskem objektu.

5.2.1.1 Map addr

Pri tem gre za metodo, ki absolutna imena spremenljivk v obliki "NAD.spremenljivka" pretvori v obliko IP.addr. Preslikava se izvede s pomočjo ACL seznama, kjer se nahaja spremenljivki pripadajoč naslov (addr). Ta predstavlja naslov spremenljivke v PLK-ju.

Primer ACL seznama:

```
;CPU CyBro-2 7431
;Addr Id    Array Offset Size Scope  Type  Name
0000  00000 1     0      1    global bit   cybro_ix00
0001  00000 1     0      1    global bit   cybro_ix01
0002  00000 1     0      1    global bit   cybro_ix02
0003  00000 1     0      1    global bit   cybro_ix03
0004  00000 1     0      1    global bit   cybro_ix04
0012  00000 1     0      1    global bit   clock_10ms              10ms clock (5ms+5ms).
0013  00000 1     0      1    global bit   clock_100ms             100ms clock (50ms+50ms).
```

```
0014  00000 1     0      1    global bit   clock_1s              1s clock (0.5s+0.5s).
0015  00000 1     0      1    global bit   clock_10s             10s clock (5s+5s).
0016  00000 1     0      1    global bit   clock_1min            1min clock (30s+30s).
```

Slika 14. Seznam ACL, primer

5.2.1.2 Filter

Gre za metodo, ki iz liste ALC pobere samo tiste spremenljivke, ki jih je DHU predčasno registriral. Prebrane spremenljivke shrani v seznam VAR list. Spremenljivke se iz tega seznama ob spremembi pošiljajo v DHU z uporabo sporočila UPDATE.

5.2.1.3 Sprejem ukaza REG PLC

Ob zagonu modul DHU registrira vse PLK-je tako, da v PCU pošlje ukaz REG PLC. Ko PCU dobi tak ukaz, se ustvari nova nit, ki operira samo znotraj obsega določenega PLK-ja. V isti niti operira tudi zgoraj opisana metoda Filter.
Prejeti ukazi: SET, REG PLC, REG VAR

5.2.1.4 Sprejem ukaza REG VAR

Po uspešni registraciji PLK-jev sledi še registracija spremenljivk, ki pripadajo določenemu PLK-ju. Tako nato nit, ki je dodeljena za rokovanje z določenim PLK-jem, točno ve, katere spremenljivke mora spremljati.

5.2.1.5 Sprejem ukaza SET

Ko se registracija PLK-jev in spremenljivk zaključi, je PCU pripravljen na sprejem ukazov tipa SET. Ob prejetju takega ukaza, spremenljivko, ki sledi ukazu, pretvori v naslov v PLK-ju, ki ga dobi iz prej ustvarjenega ALC seznama.

5.2.2 Programski modul "DHU"

Slednji PM se nahaja med samim spletnim strežnikom in zadnjim modulom v vrsti – PCU-jem. Glavna naloga je skrb za sprejemanje ukazov iz spletnega strežnika ter njihovo pretvorbo in pošiljanje naprej modulu PCU. Med drugim dostopa tudi do baze podatkov.

Poglavitne funkcije so:

- Vzdrževanje povezave med spletnim vmesnikom in PCU-jem.

- Sprejemanje ukazov iz spletnega strežnika.
- Pošiljanje ukazov PCU.
- Hranjenje ukazov v seznamu zahtev.
- Pretvorba imen spremenljivk iz log->abs in obratno.

Takoj ob zagonu se izvedejo ukazi tipa REG.

- Prijavi krmilnike (REG PLC) – vsak PLK ima svojo nit.
- Prijavi spremenljivke (REG VAR).

Postopek registracije se izvede tudi ob ponovni vzpostavitvi povezave s PCU.

5.2.2.1 Povezava s spletni vmesnikom

Takoj ob zagonu PM se najprej konfigurira ter postavi 2 strežnika:

- prvi čaka povezave PCU modulov;
- drugi pričakuje zahteve spletnega strežnika.

Njegov opis delovanje je razložen v prejšnjih poglavjih.

5.2.2.2 Uporaba baze podatkov

Za pretvorbo spremenljivk logičnega tipa v absolutni in obratno se DHU poslužuje baze podatkov. Gre za enostavno tabelo z določenimi entitetami.

V bazi podatkov so shranjeni podatki:

- o strukturi objekta (prostori),
- spisek krmilnikov,
- spisek uporabnikov,
- dnevnik vrednosti vseh ali izbranih spremenljivk,
- poročila,
- grafi,
- pravila za proženje alarmov,
- pravila obveščanja ob proženju alarmov,
- drugi specifični dodatki objekta (posebna poročila in obdelave,...).

5.2.2.3 Map_var

To je procedura, ki logične spremenljivke pretvori v absolutne. Za pretvorbo je potrebna baza podatkov. Procedura s pomočjo imena logične spremenljivke izvede poizvedbo v bazi podatkov, da dobi pripadajoče absolutno ime spremenljivke. Ime vsebuje tudi ime krmilnika, kjer se nahaja iskani senzor. Ko je pretvorba izvršena, DHU lahko pošlje spremenljivko v PCU.

5.2.2.4 Lookup_var

Ko DHU prejme spremenljivko iz PCU-ja, je ta absolutna, zato jo je treba prevesti v logično. Algoritem pretvorbe je inverzen zgoraj opisanemu. Pretvorjena spremenljivka se nato shrani v seznam VAR, kjer se nahajajo ažurirane vrednosti vseh senzorjev.

5.2.2.5 Sprejem ukaza GET

V primeru, da uporabnik želi izvedeti vrednost določenega senzorja v stanovanju, se preko spletnega vmesnika pošlje ukaz GET ter pripadajoča spremenljivka iskanega senzorja. DHU poišče prejeto spremenljivko v seznamu "VAR_list" ter prebere vrednosti in z ukazom ?ime? vrne vrednost spletnemu vmesniku.

5.2.2.6 Sprejem ukaza SET

Ko sprejme ukaz SET, ga shrani v čakajoči seznam (SET_queue). Od tam gre pretvorjena spremenljivka v absolutnem tipu v PCU ravno tako z ukazom SET. Tam se njena vrednost posreduje pripadajočemu PLK-ju.

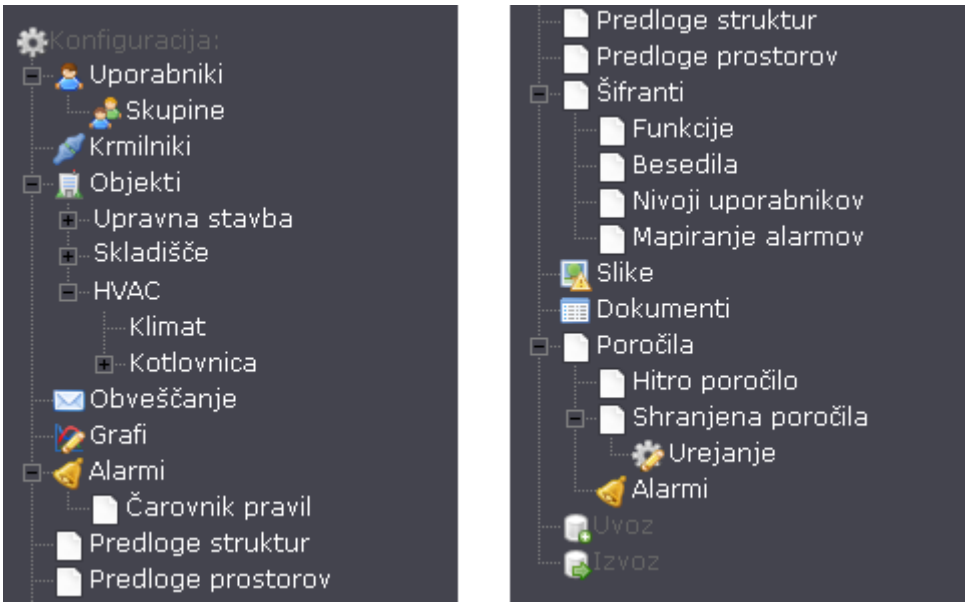
5.2.2.7 Sprejem ukaza UPDATE

Ukaz pošlje PCU skupaj z novimi vrednostmi spremenljivk. Razen ob prvem polnjenju, ko se vpisujejo začetne vrednosti. Takoj ko PCU zazna spremembo v vrednosti spremenljivke, to s pomočjo besedilnega sporočila pošlje v DHU, kjer prejeta nova vrednost prepiše staro v seznamu VAR.

5.3 Uporabniški vmesnik

Celotni sistem končni uporabnik upravlja le preko uporabniškega vmesnika, ki temelji na spletnih straneh. Večji del vmesnika je napisan v programskem jeziku PHP, poslužuje pa se tudi "javascript-a" ter ostalih prijemov. Vizija pri izdelovanju izgleda strani je temeljila predvsem na profesionalnosti, preglednosti, skladnosti in seveda intuitivnosti.

5.3.1 Opis menija za upravljanje s sistemom



Slika 15. Prikaz menija

5.3.1.1 Polje uporabniki

To je prva stvar v drevesni strukturi menija. V sistemu so vnaprej definirane **Skupine** uporabnikov. Vsaka skupina ima določene pravice upravljanja s sistemom. Nekatere skupine lahko samo pregledujejo stanje, druge lahko upravljajo s senzorji v določenem nadstropju ali sobi, tretje pa imajo popoln dostop – administrator. Pod **Krmilniki** se nahaja tabela s podatki o vseh krmilnikih, ki so v zgradbi. Možno je pregledati njihovo stanje, jih vklopiti/izklopiti, idr.

5.3.1.2 Polje objekti

Predstavljajo posamezne fizične objekte zgradbe, kjer se nahajajo senzorji. Če kliknemo na **Skladišče**, dobimo vse možne akcije (vklop luči, pregled temperature, idr.), ki jih lahko opravimo v tem prostoru. Enako velja za **Upravna stavba.** Pod **HVAC** najdemo kontrolo nad

hlajenjem prostorov ter kontrolo nad kotlovnico zgradbe. Pod **Obveščanje** določimo naslovne podatke ter tip.

5.3.1.3 Polje alarmi

To predstavlja funkcijo, ki omogoča, da nastavimo mejno vrednost določenega senzorja, in če jo ta prekorači, se sproži alarm, ki uporabnika obvesti o dogodku. Podobno kot Obvestilo je lahko poslano preko elektronske pošte ali preko SMS sporočila na mobilni telefon. Tip obvestila določimo pod **Obveščanje**.

5.3.1.4 Polji predloge struktur ter predloge prostorov

Predloge so napisane za lažjo in hitrejšo začetno postavitev sistema, ki končnemu uporabniku služijo kot neko izhodišče. Ob dodajanju novega prostora izberemo najprimernejšo predlogo prostora ter najprimernejšo predlogo strukture. Kombinacijo na koncu še dodelamo.

5.3.1.5 Polje Šifranti

V tem polju se nahajajo seznami šifrantov. Lahko ugotovimo na primer, katera oznaka spada določenemu senzorju.

5.3.1.6 Polje Slike

Tukaj so shranjene vse slike, ki so potrebne za prikaz spletnega vmesnika (gumbi, senzorji, ikone, ipd.). Možno jih je seveda naknadno spremeniti ter tako prirediti podobo vmesnika svojim lastnim željam.

5.3.1.7 Polje dokumenti

Tukaj se nahajajo razni napotki ter navodila za ravnanje z določenim senzorjem ali napravo. Znotraj menija lahko naložimo nove, ki so nato dostopni preko menija **Senzorjev** pod pripadajočo napravo.

5.3.1.8 Polje poročilo

V času delovanja sistema se beležijo vrednosti vseh priključenih senzorjev ter naprav, tako da jih je v tem meniju možno pregledati. Možen je prikaz vrednost poljubne naprave v zadnjih x dneh. Kaj naj si sistem vse zapomni, se določi v podmeniju **Urejanje**.

6 Sklepne ugotovitve

Avtomatizacija zgradb prinaša ogromno prednosti ter številne nove možnosti upravljanja z napravami po zgradbi. Vprašanje je, kako naj se lotimo avtomatizacije? Na trgu je dandanes ogromno rešitev za avtomatizacijo zgradb. Večina rešitev je popolnoma enakega tipa, le določeni proizvajalci delajo razlike. Rešitev s pomočjo modulov je primerna za avtomatizacijo manjšega števila naprav. V kolikor želimo avtomatizirati celo zgradbo, je potreben drugače pristop. Prava rešitev je centralno vodenje s programabilnimi logičnimi krmilniki. V večini primerov je tak način dražji, vendar je nesporno boljši v vseh pogledih.

Realizacija sistema za upravljanje z zgradbami cyBMS je bila uspešno izvedena. Uspešno je bila napisana programska koda, ki vodi krmilnike ter prenaša podatke s spletnega vmesnika do krmilnikov ter nazaj. Dosežena je bila želena stabilnost in konsistenčnost delovanja sistema. Sistem kot tak ima veliko funkcij delovanja, ki se jih vse nastavi v spletnem vmesniku. Napisan je tako, da lahko brez večjih težav po potrebi dodamo poljubno novo funkcijo. Naslednja faza pri tem sistemu bo zagotovo prevedba programske kode v enega izmed objektnih jezikov, najbrž bo to kar C++. Sistem kot tak je že v uporabi v raznih poslovnih prostorih. Naj omenim poslovno zgradbo Rotonda v Ljubljani. Tam je v omrežju preko 80 krmilnikov, ki imajo nadzor nad večino naprav po prostorih, krmili pa jih realiziran sistem za upravljanje zgradb. Pri tako velikem številu PLK-jev je možnost, da pride do kake izmed napak veliko večja, vendar se je na koncu izkazalo, da so se naša testiranja splačala in kot izgleda, sam sistem deluje brez kakršnih koli posebnosti. Je pa v tem primeru to največje število uporabljenih PLK-jev v našem sistemu doslej. Upamo in želimo si, da temu sledijo še večji poslovni prostori in seveda nadgradnje ter izpopolnjevanje našega sistema.

Seznam slik in tabel

Seznam slik

Slika 1. Zaporedje akcij. O-oddajnik, P-prejemnik. 6
Slika 2. Modul Fibar 12
Slika 3. Shema sistema cyBMS 14
Slika 4. Krmilnik 15
Slika 5. Krmilnik z zaslonom 16
Slika 6. Povezovalna shema modulov 16
Slika 7. Shema modela "client-server" 17
Slika 8. Metoda map_var 18
Slika 9. Metoda look_var 19
Slika 10. Seznam s kazalci 20
Slika 11. Metode seznama elementov 20
Slika 12. Primer semaforja 23
Slika 13. Prikaz posameznih modulov 24
Slika 14. Seznam ACL, primer 25
Slika 15. Prikaz menija 28

Seznam tabel

Tabela 1. Primerjava lastnosti protokolov 8
Tabela 2. Prikaz definiranih ukazov ter primeri uporabe 21

Viri

[1] "Pametne zgradbe. Znanstvena fantastika v zgradbi prihodnosti", *Računalniške novice*, št. 8/XVII, str. 16-21, 2012.

[2] Razvojna dokumentacija Indea – interno gradivo

[3] Home Automation and Smart Metering, *Technology First, Farnell*, Journal IX, str. 16-17, 2010.

[4] (2012) Specifikacije. Dostopno na: http://www.indea.si

[5] (2012) Komponente. Dostopno na: http://www.robotina.si/

[6] (2012) Primerjava INSTEON standarda z ostalimi. Dostopno na: http://www.netropolus.com/LEARNINGINSTEONvsZWave.htm

[7] (2011) Protokol X10. Dostopno na: http://www.x10.com/support/basicx10.htm

[8] (2012) Rešitev HomeSeer. Dostopno na: http://www.homeseer.com/

[9] (2012) Spletna trgovina HomeSeer produktov. Dostopno na: http://store.homeseer.com/store/

[10] (2012) Rešitev Fibaro. Dostopno na: http://www.fibaro.com/the-fibaro-system

[11] (2011) Protokol UPB v primerjavi z X10. Dostopno na: http://pulseworx.com/upb/x10_compare_upb_.htm