**ТЕХНОЛОГИЧНО УЧИЛИЩЕ ЕЛЕКТРОННИ СИСТЕМИ**

**към ТЕХНИЧЕСКИ УНИВЕРСИТЕТ - СОФИЯ**

**ДИПЛОМНА РАБОТА**

Тема: Система за автоматизация на съществуващи ОВК инсталации

Дипломант: Научен ръководител:

*Ванеса Стойновa маг. инж. Любомир Циров*

СОФИЯ

2018

# УВОД

До скоро, системи обхващащи цели помещения с опция за централизиран автоматичен контрол, можеха да бъдат открити само в индустрията или в луксозните частни домове. Типично тези системи обхващаха само светлините и климатиците, а възможностите отвъд тези базови функции за наблюдение и контрол, бяха изключително редки или ограничени само до едно място в сградата, откъдето това беше възможно да се прави. В този смисъл, може да се смята, че автоматизацията не е някаква революционна стъпка, а такава е наскоро появилата се възможност всички компоненти на домa или офиса да бъдат свързани в мрежа, със собствен IP aдрес, с което да могат да бъдат достъпени и управлявани отвсякъде с помощта на интернет.

С помощта на системата за автоматизация на съществуващи ОВК (Отопление, вентилация и климатизация) инсталации потребителя може да наблюдава и променяте настройките за температура чрез умен термостат, уеб интефрейс, мобилно приложение или друго. Идеална е за хора, които трябва да пътуват много за работата си или които обичат да пътуват дълго. Например, за продължителни пътувания през зимата, потребителя може да намали температурата на къщата, докато отсъства, за да намали консумацията на отоплителните уреди.

През пролетта и есента, температурата може да варира драматично в течение на един ден. Например, през пролетните месеци не е необичайно да бъде 4 градуса сутрин и 19 градуса следобед. Поради тази причина може да се наложи да се работи с отоплението в ранните сутрешни часове, но не и следобед. С помощта на системата за автоматизация клиента може да контролира настройките за отопление и охлаждане, за да компенсира извънредните сезонни температурни колебания. Всеки път, когато се увеличава енергийната ефективност на дома или офиса, се намаляват разходите за сметките за ток.

# ПЪРВА ГЛАВА

# ПРЕГЛЕД НА ПОДОБНИ ПРОДУКТИ, СЪЩЕСТВУВАЩИ РЕШЕНИЯ И ПОДХОДЯЩИ ТЕХНОЛОГИИ

## ПРЕГЛЕД НА ПОДОБНИ ПРОДУКТИ

Пазара в днешно време предлага голямо разнообразие от системи за автоматизация на ОВК инсталации и подобни продукти. Производителите на електроника предлагат огромен набор от „умни“ устройства и приложения, много от които имат широка гама от иновативни опции. Една от най-големите фирми в производството на „умни“ устройства е Nest.

### Nest Learning Thermostat

****

**Фиг. 1.1 Умен термостат Nest**

През2011 година компанията Nest Lab разработва своя първи термостат. Той е различен, модерен и работи изцяло на нов принцип. Термостатът Nest Learning е първият по рода си, който получава сертификат ENERGY STAR. Той се учи от ежедневно потребление на енергия на потребителя.

Термостатър е със скромни размери: на височина – 8.4 см, а на широчина 3.2 см Разполага със сензори за температура, влажност, близка активност, далечна активност и допълнителен индикатор за светлина. Работи с 95% от 24V нагряващи и охлаждащи системи, включително газ, ток, въздухопоток, топлинна помпа, масло, топла вода, соларни и геотемални.

Nest научава температурата, която потребителя иска и се програмира сам в рамките на една седмица. Автоматично се изключва, при отсъствието на хора в дома, за да спестява енергия. Термостатът засича потребителя в другия край на помещението и веднага показва времето, часа и температурата. Допълнителна фукнция е това, че спокойно може да се контролира и дистанционно чрез браузър на компютър или специално разработените приложения за iPhone, iPad и Android. Чрез тях потребителя може да провери историята на използваната енергия (колко и за какво е използвана). Поддържа няколко езика – английски, немски, френски, испански и италиански.

С Nest Sense термостатът Nest изучава дома: като колко време отнема да се затопли, колко e проветрива е и какво е времето. Също така научава за отоплителната и/или охладителната система на дома и как да я направи още по-ефективна.

**Nest Learning Thermostat** спестява над 4 милиарда kWh енергия в милиони домове по света. Независими учени доказали, че спестява между 12 ~ 14% от сметката за отопление и 15-16% от сметката за охлаждане. **[1]**

### CoolMasterNet

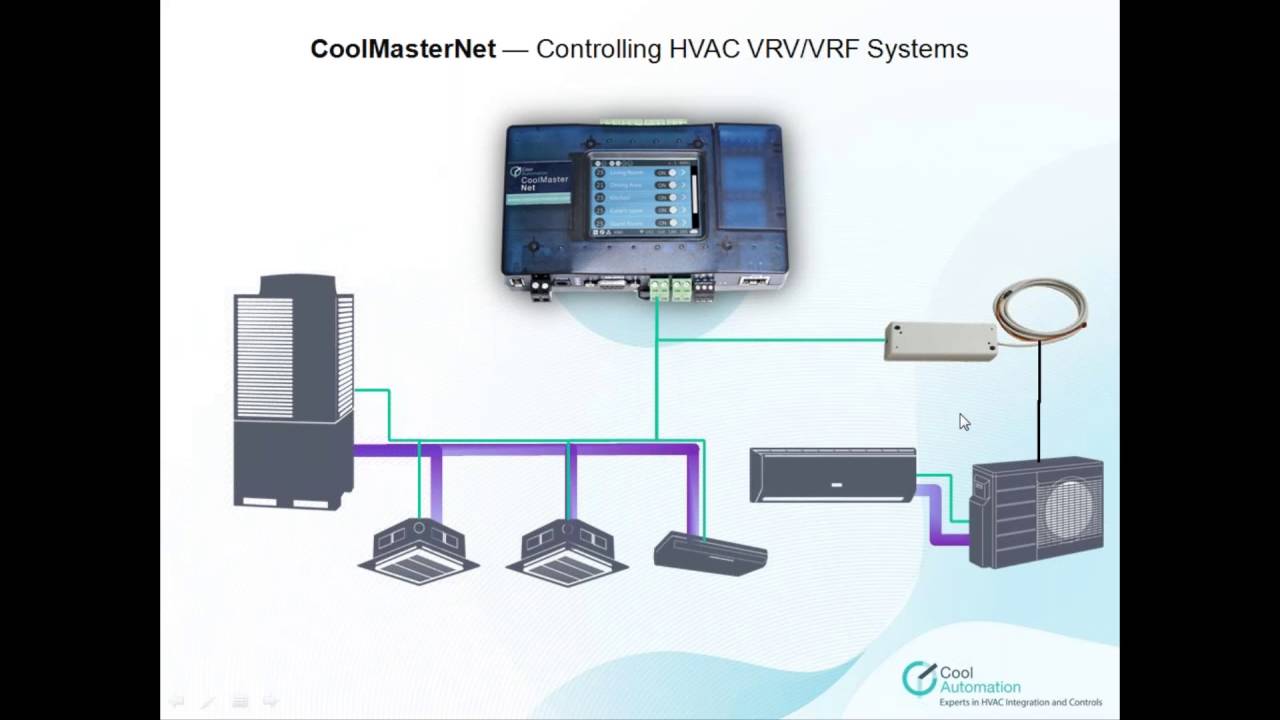


Фиг. 1.2.1 комуникационен мост CoolMasterNet

CoolMasterNet е уникален комуникационен мост, който безпроблемно свързва усъвършенствани системи за климатизация (VRF), системи за домашна автоматизация и системи за управление на сградите (BMS). CoolMasterNet осигурява общи интерфейси RS232 и RS485, включително Ethernet порт, изискващ най-често срещаните софтуерни команди за безпроблемен процес на интеграция.

Мостът CoolMasterNet може лесно да се свърже към терминалите на комуникационната шина за климатизация VRV / VRF, за да контролира и наблюдава най-важните параметри на климатичната система.

Интегратори за домашна автоматизация и инсталатори на ОВК - Могат лесно да свързват разширени инсталации, като игнорират своите сложни комуникационни протоколи. Решението „Plug & Play“ на CoolMasterNet дава възможност за прост процес на интеграция. Доставчици на ОВК услуги - имат достъп до най-важните параметри на инсталациите, необходими за диагностични и обслужващи цели.

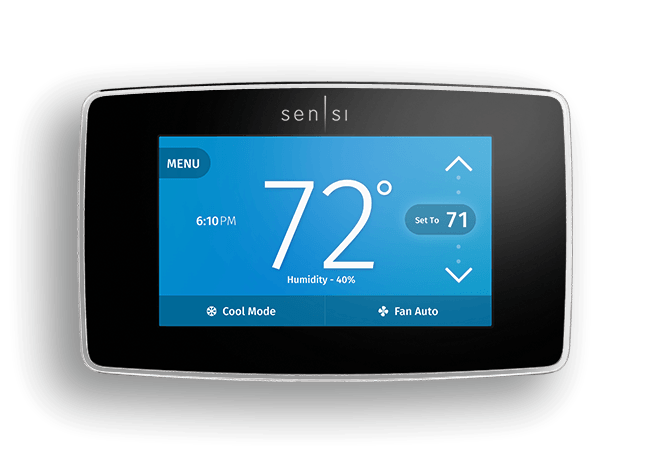


Фиг. 1.2.2

CoolMasterNet поддържа повечето съществуващи марки на VRF с една хардуерна платформа. Поддържанни марки са: Daikin, Mitsubishi, Toshiba, Fujitsu, LG, Hitachi, Gree и др.

С CoolMasterNet, свързан с интернет като самостоятелно устройство - CoolRemote се превръща в уеб базирано решение за управление, което позволява на крайните потребители да управляват и наблюдават VRV / VRF / Split HVAC устройства от фиксирани или мобилни платформи, напр. смарт телефони, таблети и персонални компютри и отвсякъде. **[2]**

### Sensi™ Touch Wi-Fi Thermostat



Фиг 1.3 Умен термостат Sensi

Емерсoн е водещ доставчик на решения за отопление и охлаждане, който пуска във вход следващото развитие на своя национално признат интелигентен термостат. Новият Sensi ™ Touch Wi-Fi® термостат съчетава доказаната интелигентна домашна технология с цветен сензорен дисплей и елегантен, минималистичен външен вид, който ще бъде у дома във всяко модерно жилищно пространство.

Новият SensiTouch Wi-Fi термостат на Emerson, заедно с актуализирана версия на класическия Sensi ™ Wi-Fi термостат, вече е съвместим с платформата за управление на смартфони Apple HomeKit, осигурявайки прост и сигурен контрол чрез приложението Apple Home на iPhone, iPad или Apple Watch. Новите термостати Sensi ще надграждат наследството на марката за свързване на прецизен контрол на температурата и удобство, за да се осигури най-високо ниво на домашен комфорт.

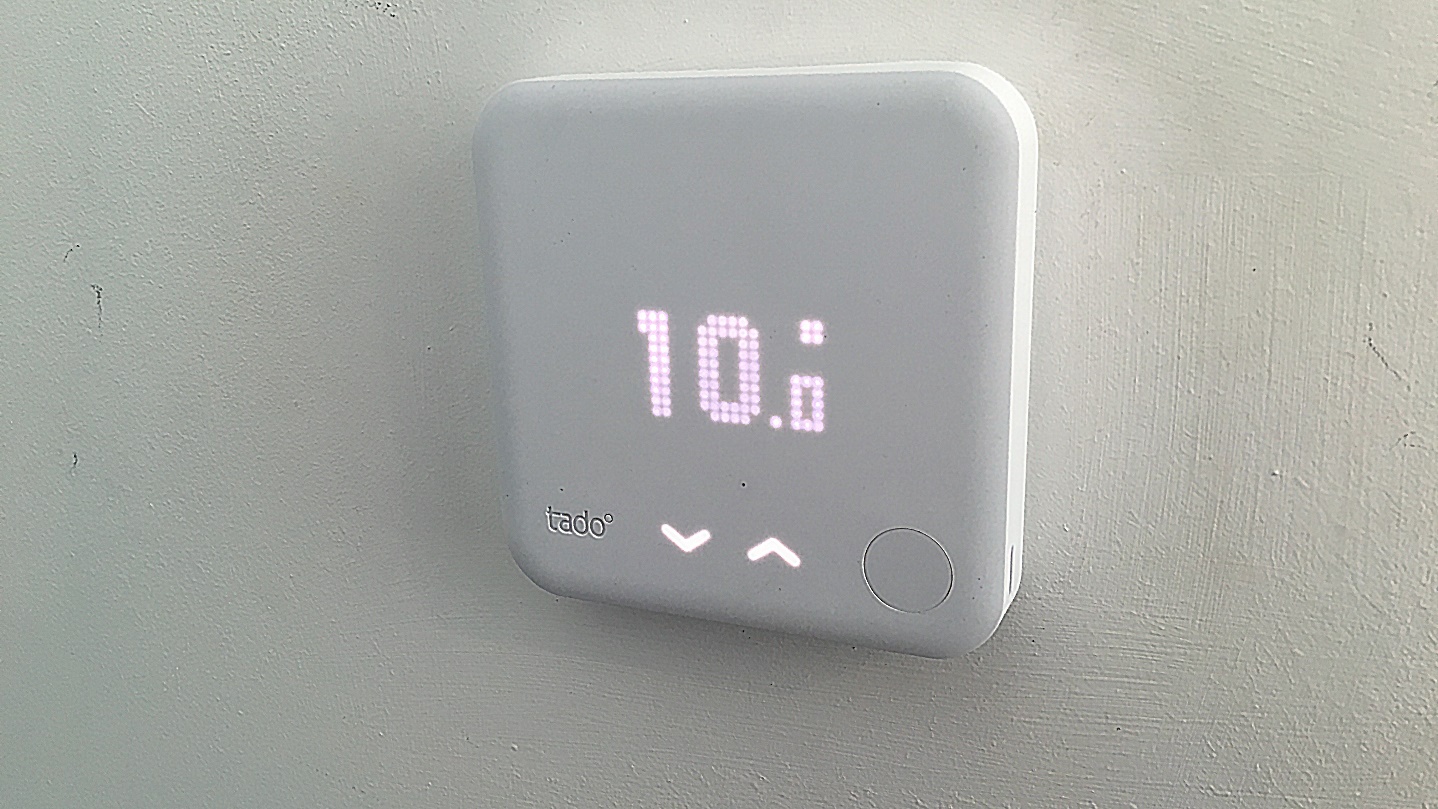
Sensi Touch Wi-Fi термостат предлага най-доброто в своя клас инсталация както за собствениците на жилища, така и за изпълнителите, с инструкции стъпка по стъпка и видео уроци в приложението Sensi за смартфони. Потребителите могат да използват новата функция за разпознаване на близост с геофенсинг или опции за гъвкав график, за да постигнат икономия на енергията, без да компрометира техния комфорт. Термостатът използва интелигентни сигнали, за да уведоми клиентите за екстремни температури и влажност.

Sensi Touch Wi-Fi термостатът има цветен сензорен екран с висока разделителна способност и чист дизайн, който отговаря на модерния дом. Термостатът също има осветени клеми за по-голяма видимост и лесни бутони, които премахват необходимостта от отвертка. Допълнителните опции на менюто позволяват на потребителите да персонализират информацията на началния екран и да изберат непрекъснато задно осветяване за меко осветление в тъмни коридори или стаи. Актуализираният термостат Sensi Wi-Fi не изисква обикновен проводник (c-wire) за повечето инсталации.

С помощта на Apple HomeKit потребителя може бързо и лесно да сглоби, надеждно да сдвои Sensi термоста си със своя iPhone или iPad, да го добави към домашното си устройство, да се интегрира с други аксесоари HomeKit чрез създаването на сцени, и да започне да контролира температурата в дома си чрез приложението Apple Home, Siri или контролния център. **[3]**

### Tado Smart Thermostat

Tado пуска на пазара първия си интелигентен термостат през 2014 година. Оттогава развитието на продукта, актуализираното приложение и добавянето на индивидуален контрол на радиатора драстично подобрява развитието на платформата. Сега, термостатът Tado Smart е един от най-мощните и умни отоплителни устройства, които може даден клиент да закупи. Към настоящия момент **Tado Smart Thermostat е иновативен и съвременен безжичен термостат, който се грижи за уюта на потребителя вместо него.**



Фиг 1.4 Умен термостат Tado

**Състои се от два компонента: термостат и мост.** Мостът комуникира безжично с термостата и е свързан в задната част на маршрутизатора на потребителя.

Бутонът на термостата извежда атрактивен бял LED дисплей, който може да покаже температурата на околната среда и състоянието на връзката. Самият мост разполага със индикатори за състояние "връзка", "рутер" и "интернет".

С управлението на няколко помещения и на новия алгоритъм за отопление, tado ° може да спести още повече разходите за отопление и да направи живота на потребителя по-комфортен. Подобреният интерфейс на самия термостат и новите Smart Schedules в приложението правят управлението на отоплението още по-лесно.

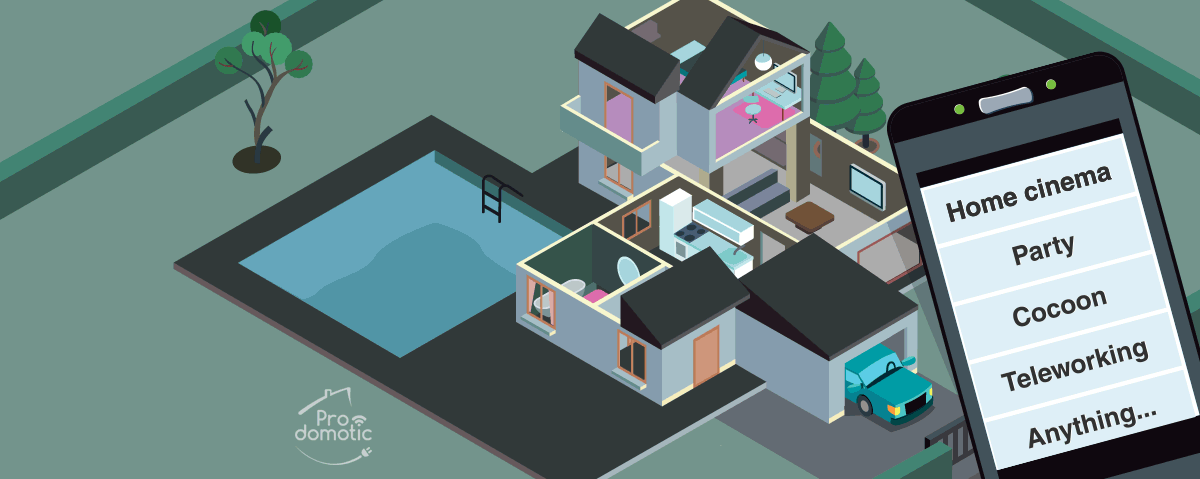
Подходът на Tado към интелигентното отопление е базиран на откриване на присъствие, използвайки местоположението на мобилния телефон на потребителя, за да открие дали е у дома или не. Приложението може да проследява няколко смартфона и може да поддържа потребителски акаунт. Откриването на присъствие е отлично, като приложението също проследява местоположението на потребителя, така че може да започне да включва отоплението, докато той се прибира.

Тадо наскоро осъвременява системата, за да промени начина на работа на режимите на отопление. Със старата система можеше да се настроят температурите за ден и нощ ръчно и в режим "Отдалечен". Новата система добавя по-подробен график за дома, подобно на традиционния термостат. Например, може да се включи отоплението сутрин, но да се изключи през деня, когато е по-топло времето. Контролът се осъществява чрез уеб браузър, приложение за смартфони (Android или iOS) и дисплея на термостата. Чрез уеб браузъра или приложението може да се зададе стандартното действие за температурни промени, направени чрез термостата. Опциите са: да се запазят температурната промяна до следващата планирана точка на превключване, да се направи промяната постоянна или да се зададе време за изчакване по подразбиране - например два часа. Като цяло приложението дава пълен контрол над отоплението, независимо от местоположението на потребителя. **[4]**

### Pro-domotic

Системата за домашна автоматизация ще позволи на потребителя да контролира всеки тип топлинен източник. Pro-Domotic работи с термостат на стая, което позволява да се зададе конкретна температура във всяка . Може да се контролира нивото на петролния резервоар и дневната консумация на нефт или газ.

Системата за домашна автоматизация дава възможност на потребителя да си представи десетки сценарии за всеки момент от ежедневиет. Друго предимство на домашната автоматизация е възможността да се управляват за големи помещения две различни температури.



Фиг. 1.5 Pro-domotic

От приложението Pro-domotic потребителят може да развие жизнени сценарии. Например отоплението ще бъде автоматично настроено на комфортен режим, когато има човек у дома, в икономичен режим за кратки отсъствия (работни дни) и режим за защита от замръзване за дълги отсъствия (ваканции). Разбира се, може по всяко време да се възстанови контрола върху оборудването.

Със система за домашна автоматизация може да се наблюдава консумацията на енергия, за да се контролират разходите и да се оптимизира потреблението, като същевременно се запази комфорта.

Благодарение на енергийното табло може да се проследи консумацията на различни съоръжения: електричество, вода, газ, слънчева енергия и отопление.

Потребителя може да интегрира сензори за температура към превключвателите, за да получи сигнали при необичайно ниво на температурата. Тази функция е много полезна за помещения, които съдържат електронно оборудване или предмети, които не могат да издържат на екстремни температури. Например потребителят има хладилна винена изба, която трябва да остане при постоянна температура. Ще бъде изпратено предупреждение към неговия смартфона, когато настъпи внезапна промяна в температурата.

Pro-Domotico включва термостати и температурни сензори в превключватели. Контролът на топлината се извършва от техническата област, което позволява отсъствието на термостатични вентили. Няма да се видждат термостатите или термостатичните вентили, които понякога може да не се харесат на потребителя. **[5]**

## СРАВНЕНИЕ НА СЪЩЕСТВУВАЩИ РЕШЕНИЯ

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **System name** | **Nest** | **CoolMasterNet** | **SensiTouch** | **Tado** | **Pro-domotic** |
| **System type** | Smart thermostat | Communication bridge | Smart thermostat | Smart thetmostat | Smart automation SW system |
| **Approx. Price** | 194 € | 1,560 € | 161,74 € | 249.00€ | Pricing not available |
| **Network Interfaces** | WiFi | Ethernet/ WiFi | WiFi | WiFi | WiFi |
| **Allows monitoring** | ✔ | ✔ | ✔ | ✔ | ✔ |
| **Allows control** | ✔ | ✔ | ✔ | ✔ | ✔ |
| **Allows expansions** | ✘ | ✔ | ✘ | ✔ | ✔ |
| **Smartphone app** | ✔ | ✘ | ✔ | ✔ | ✔ |
| **Web Interface via browser** | ✔ | ✔ | ✔ | ✔ | ✔ |
| **REST API** | ✘ | ✘ | ✘ | ✘ | ✘ |

Табл.

Разгледаните продукти разполагат с модул за комуникация по интернтер. Стойността им варира от 160 евро до 1,560 евро, което е голяма разлика. Умните термостати предлагат отадлечен контрол и наблюдение само на температурата, докато комуникационния мост и системата Pro-domotic дават възможност на потребителя за контрол и наблюдение и на други параметри/функции като : сигурност, осветление и др. Стойноста

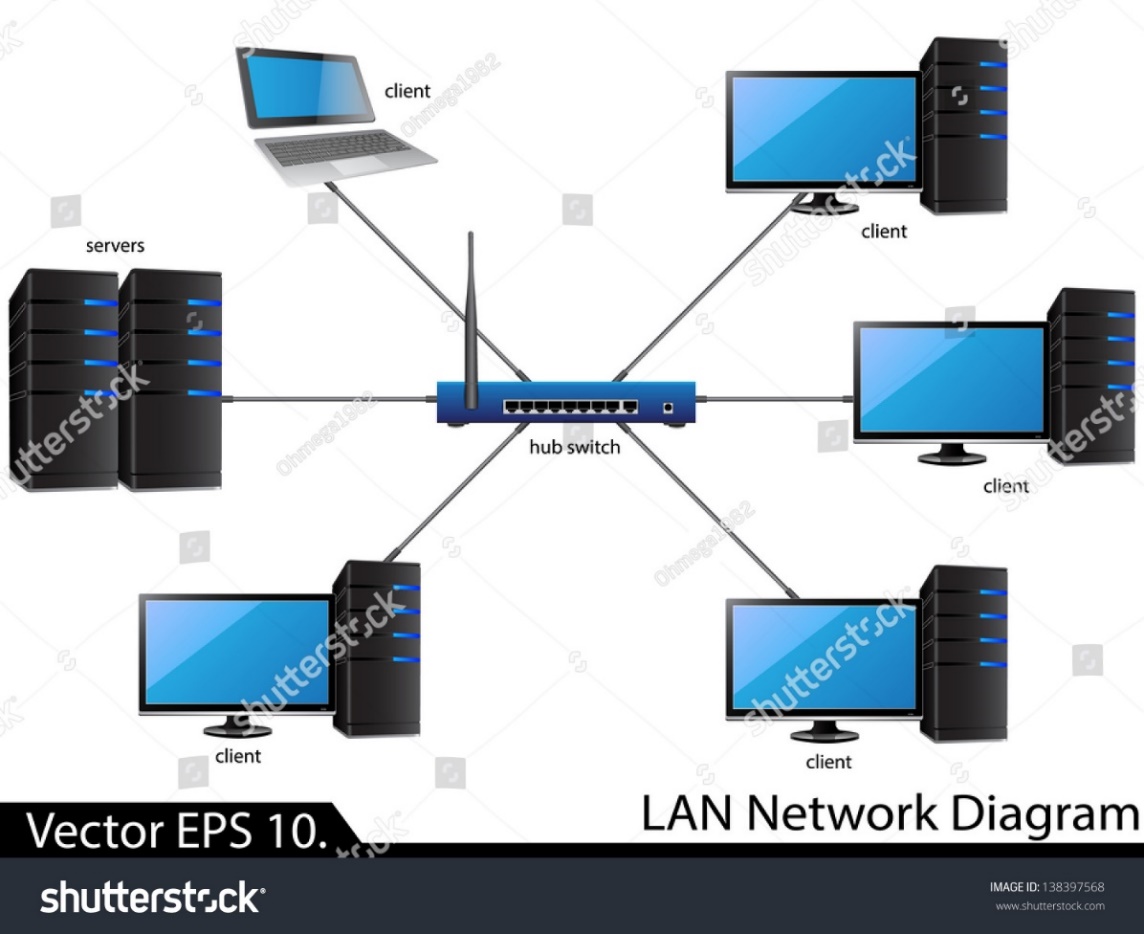
## 1.3 ПОДХОДЯЩИ ТЕХНОЛОГИИ

### 1.3.1 Local area networks

Локална мрежа е вид малка компютърна мрежа, обслужваща компютри и други устройства (напр. мрежови принтери или скенери), свързани помежду си. В днешно време най-разпространени са технологиите на свързване Ethernet или Wi-Fi, за разлика от миналото, когато са били използвани предимно ARCNET, Token ring и др.

Чрез локалната мрежа се осъществява:

* Споделяне на информация между компютрите. Всеки потребител в мрежата може да получи достъп до всички или само до някои от устройствата, директориите или отделни файлове, в зависимост от това с какви права разполага (пълни права, само за четене и др.)
* Свързване на приложения, които са инсталирани на различни компютри в мрежата, напр. програма, която работи на един компютър, би могла да използва помощни програми, които са инсталирани на друг компютър.
* Автоматична синхронизация на файлове в цялата мрежа. Повечето мрежови софтуери предоставят възможност някои от файловете след обработка да бъдат обновени на всички компютри, където има техни копия. **[6]**



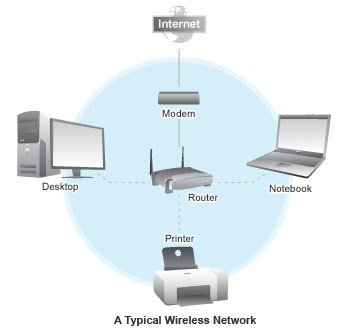
Фиг 1.6 Схема на локална мрежа

### 1.3.2 Wi Fi

Wi Fi е популярна технология, която позволява електронни устройства да обменят данни или да се свързват с Интернет безжично, използвайки радиовълни. Това е технология на безжичната мрежа (WLAN) базирана на спецификациите от серията IEEE 802.11. Първоначално тя е лицензирана от Wi-Fi Alliance. Била е разработена, за да бъде използвана от преносимите изчислителни устройства, като преносими компютри, в локални мрежи (LAN), но сега все повече се използва и за други услуги, включително Internet и VoIP, игри, базово свързване на потребителска електроника, като телевизори и DVD устройства или цифрови камери.

През 1997 година IEEE пускат първия оригинален 802.11 стандарт, който не се приема сериозно поради ниската си максимална скорост. Това продължава така до 1999 година, когато IEEE пускат 802.11b стандарта, който се разпространява широко и зад него застават 3Com, Aironet, Harris Semiconductor, Lucent, Nokia и Symbol Technologies, които основават независима организация, наречена Wireless Ethernet Compatibility Alliance (WECA). Целта на WECA е да промотира, тества и сертифицира Wi-Fi технологията и продуктите. През 2003 година WECA си сменя името на Wi-Fi Alliance, при което Wi-Fi се превръща в синоним на “Безжичен Интернет”. През същата година IEEE пускат 802.11g стандара, който също както и 802.11b работи на 2.4 Ghz, но с много по-висока скорост – 24.7 Mbps в секунда.

Най-новият стандарт в безжичните мрежи е 802.11n, с който може да бъдат достигнати скорости от типа на 140 мегабита в секунда. Тези скорости са постигнати благодарение на Multiple Input and Multiple Output (MIMO) технологията. При нея се използва противоположна идея на a/b/g/. Докато при 802.11a/b/g/ стандартите, предметите, които отразяват сигналите (като стените например) предизвикват интерференция, при MIMO тези отражения се използват. Това става чрез интегрирането на няколко антени, разделени на няколко модули. При предаването на пакетите, информацията се разделя на части, а всеки модул я изпраща към съответния модул на приемащата антена.



Фиг 1.7 Схема на безжична мрежа

Работата на Wi-Fi рутера е да приема сигнал от единици и нули от Ethernet връзката си, да я преобразува в радио вълни и да я изпраща към адаптера на лаптопа на потребителя . Той пък я приема, чрез антената си, декодира я в единици и нули и я представя. Когато трябва да върне информация, всичко минава по обратния път.  
В затворени пространства, безжична мрежа може да предава данни в разстояние от 20 до 45 метра, поради високата интерференция. На открито обаче такава мрежа може да работи от разстояние до над 300 метра.

**Wi-Fi Protected Access (WPA)** е основният протокол за сигурност. С прости думи казано, чрез WPA можете да се поставят трудноразбиваеми (но съвсем не и невъзможно) пароли, които почти със сигурност ще спре опитите на околните за безплатен Интернет. **[7]**

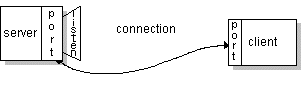
### 1.3.2 IP communication ( sockets)

Сокетите представляват крайна точка на двупосочна комуникация между две програми-сървърна програма и клиентска програма.  
Обикновено един сървър работи въру специална машина, която в Интернет пространството се индентифицира с име или IP адрес. Сървърът чака(“слуша”) на даден порт клиет да се свърже с него. Порт- представлява число от 0 до 65536 и е уникален идентификатор на дадена услуга или приложение. За да се свърже даден клиент с даден сървър е необходимо клиентът да знае името или IP адреса на дадения сървър и порта, на който сървърът „слуша”. След като клиентът се свърже със сървъра е необходимо той да се индентифицира с нов собствен порт – уникален номер, който той ще използва по време на дадената връзка.

[](http://javac.bg/wp-content/uploads/2015/11/mreji2.gif)

Фиг. 1.8 Заявка от клиента към сървъра

Ако всичко премине успешно, сървърът приема връзката с клиента като запазва неговия адрес и порт номер.

[](http://javac.bg/wp-content/uploads/2015/11/mreji3.gif)

Фиг. 1.9 Връзка между сървъра и клиента

След изграждане на дадената връзка, сървърът се връща към първоначалното си състояние, в което слуша за нови клиенти. **[8]**

### 1.3.3 MQTT

MQ Telemetry Transport (MQTT) е отворен протокол, който позволява на устройствата да изпълняват команди с помощта на съобщения. С MQTT сравнително лесно можете да настройвате програми на машина с инсталиран Linux, които да събират информация и да я публикуват в мрежата, като я подават на малки, ограничени откъм ресурси микроконтролери, които да видят тази информация и да извършат командата. Наскоро появилите се WiFi микроконтролери, като например ESP8266 правят цялото това нещо изключително достъпно.

Преимуществото на това да използвате съобщения е, че устройствата могат да „слушат“ за интересни неща и да изпратят всякаква информация, която смятат за важна. Също така, няма нужда всяко устройство да знае за другите свързани в мрежата. Например, станцията за времето, може да направи публикация – каква е температурата, влажността, скоростта и посоката на вятъра и останалите „неща“ могат да се абонират, за да се възползват от информацията. Въпреки че има много начини за изпращане на съобщения през Linux, MQTT би ви позволил да пратите съобщения към и от своите Arduinо или mbed смарт устройства. Ако сте заинтересовани да си купите IoT или smart device, най-вероятно ще искате да знаете дали messaging използва open standard, също като MQTT.

MQTT e направен от OASIS и е open standard. Много имплементации на MQTT са налични. **[9]**

# ВТОРА ГЛАВА

# ИЗИСКВАНИЯ КЪМ ПРОДУКТА, ОБЩА СХЕМА НА ПРОДУКТА, ИЗБОР НА КОМПОНЕНТИ И ПРОЕКТИРАНЕ НА БЛОКОВИ СХЕМИ НА ОТДЕЛНИТЕ ЗВЕНА

## 2.1 Изисквания към програмния продукт

Програмният продукт ще се състои от два компонента – централно звено и сезнорно звено. Централното звено ще приема информация от сензорното звено, която ще бъде обработена и предоставена на потребителя по подходящ начин. Работата на сензорното звено е да може да отчита информация и да я предоставя на централното звено при нейното поикване.

### 2.1.1 Хардуерни изисквания

2.1.1.1 Централно звено

- Да разполага с достатъчно оперативна памет

- Да разполага с етернет порт

- Да разполага сWi Fi модул

- Да разполага с Micro USB конектор

- Да разполага със следните интерейси – GPIO/UART/SPI/I2C

` 2.1.1.2 Сензорно звено

- Да може да свързва безжично към дадена мрежа

- Да разполага със следните интерфейс за свързване на сензори - - SPI/I2C/Analog

- Захранване – батерия АА или micro USB конеткор

### 2.1.2 Софтуерни изисквания

2.1.2.1 Софтуер на централното звено

- Да осъществява IP комуникация

- Да има база данни

- Да поддържа HTTP интерфейс

- Да поддържа MQTT интерфейс

2.1.2.2 Софтуер на сензорното звено

- Да има real-time operating system (RTOS)

- Да поддържа MQTT интерфейс

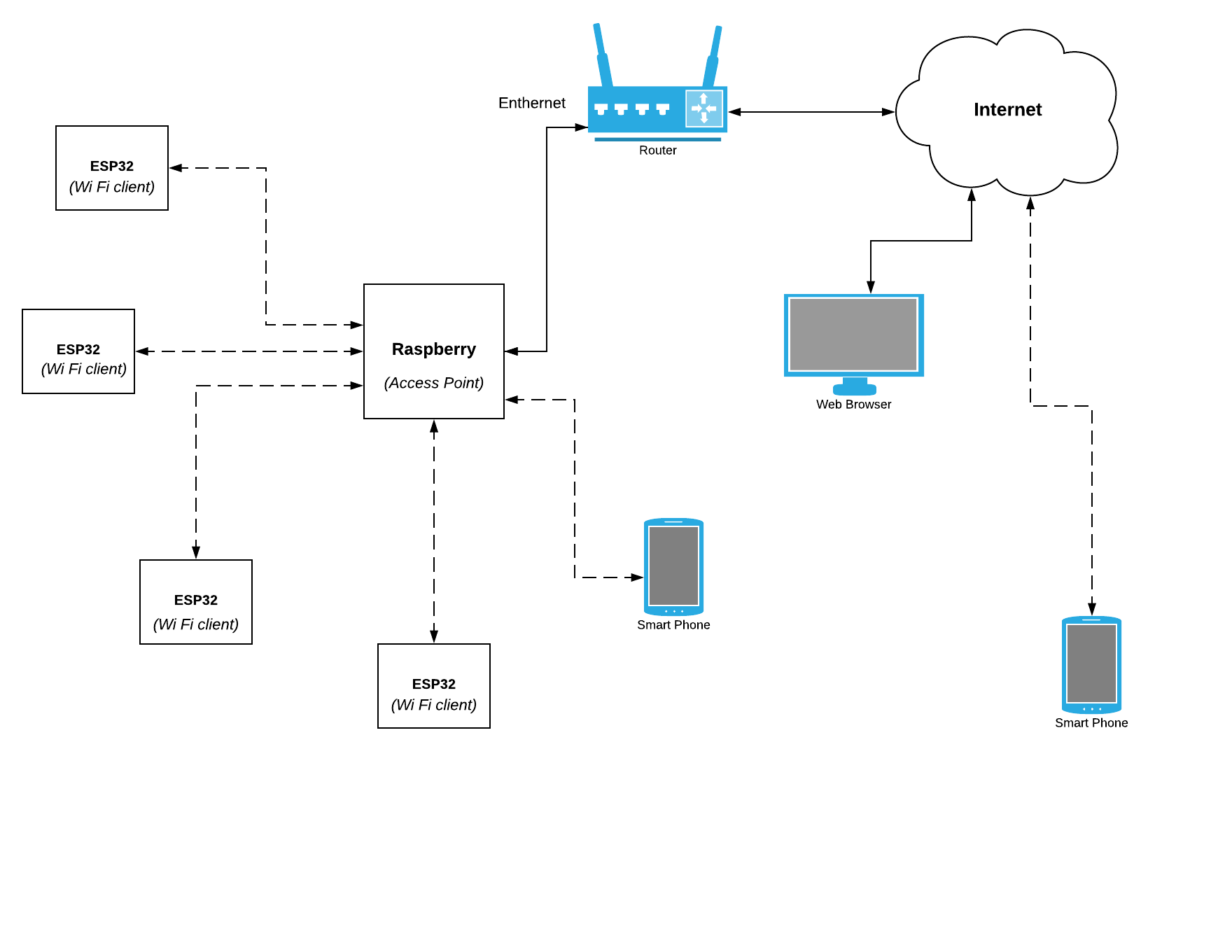
- Да има работещ енергоспестяващ режим

2.1.2.3 Софтуер на уеб интерфейса

- Да предоставя възможност за наблюдение

- Да предоставя възможност за контрол

## 2.2 Общи схеми на продукта



Фиг 2.1 Обща схема на продукта

Централното звено ще изпълнява роля на access point, за да може да си комуникира с всяко едно сензорно звено. Сенорните звена ще отчитат и изпращат информация на централното звено, където тя ще бъде обработвана и качвана на сървъра, който ще се намира на централното звено. Сървъра ще обновява информацията в уеб интефейса от където потребителя да може да наблюдава .

## 2.3 Избор на компоненти

### 2.3.1 Избор на хардуерна платформа

2.3.1.1 Централно звено

От разгледаните платки в таблица 2.1 Raspberry Pi 3 платката разполага с най- малко памет, но за сметка на това поддържа най-голям и разнооббразен брой операционни системи. Неговата тактова честота, с която се определя количеството операции, които може да направи процесора за секунда време, също е най-ниската от разгледаните платни в таблицата. Броя на ядрата при всички мини компютри е равен.

Разполагат с основните интерфейси, които отговаят на хардуерните изисквания за централното звено, добра скорост за пренос на данни и съвременен стандарт.

Платформата Raspberry Pi 3 е с най-голяма популярност и с най-мaсово на брой разработени проекти. Ниската цена е не по-малко значим фактор.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Board Name** |  | **Raspberry Pi 3 B** | **Аsus Tinker Board** | **OrangePI Plus2** | **Odroid-C2** |
| **Operating system support** |  | Raspbian (Debian), Ubuntu, Fedora, RISC OS, Windows 10 IoT Core и др. | ThinkerOS, Armbian, Android | Android, Lubuntu, Debian, Rasberry Pi Image | Ubuntu, Android и други Linux дистрибуции |
| **CPU** | Arch | 64-bit ARM Cortex-A53 | 32-bit ARM Cortex-A17 | Cortex-A7 | 64-bit ARMv8 |
| Cores | Quad Core | Quad Core | Quad Core | Quad Core |
| Frequency | 1.2GHz | 1.8GHz | 1.8GHz | 1.5GHz |
| **Memory** | RAM | 1 GB | 2GB | 2GB | 2GB |
| Storage | MicroSD slot | MicroSD slot | Onboard storage 16 GB + microSD slot | MicroSD slot |
| **Interfaces** | UART | ✔ | ✔ | ✔ | ✔ |
| GPIO | ✔ | ✔ | ✔ | ✔ |
| I2C | ✔ | ✔ | ✔ | ✔ |
| SPI | ✔ | ✔ | ✔ | ✔ |
| Ethernet | 10/100 Mbps | 10/100 Mbps | 10/100/1000 Mbps | 10/100/1000 Mbps |
| **Wi Fi** | Supported versions | 802.11.b/g/n | 802.11 b/g/n | 802.11 b/g/n | 802.3 |
| Antenna type | integrated chip | IPEX antenna header | integrated chip | integrated chip |
| **Power supply** |  | 5V от DC адаптер / GPIO конектор | 5V от DC адаптер | 5V от DC адаптер | 5V от DC адаптер |
| **Price** |  | 79.00 лв | 120.00 лв | 131.00 лв | 136.00лв |

Табл. 2.1 Сравнение за хардуер за централното звено

2.3.1.2 Сензорно звено

За сензорно звено избирам модула NodeMCU v3. В таблица 2.2 са сравнени няколко модула, отговарящи на хардуерните изисквания. NodeMCU също като ESP-WROOM-02 е базиран на ESP8266 микроконтролера, но NodeMCU разполага както с повече RAM памет, така и с повече flash памет. Паметта е за сметка на цената

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Module Name** |  | **ESP-WROOM-02** | **ESP-WROOM-32** | **Sparrow v3** | **Genuino MKR1000** | **NodeMCU** |
| **Microcontroller** | Name | EP8266 | ESP32 | ATmega128RFA1 | SAMD21 | EP8266 |
| Frequency | 160 MHz | 40 MHz | 16 MHZ | 48 MHz | 160 MHz |
| Processing core | Tensilica L106 32-bit | Xtensa 32-bit LX6 | 8-bit ATmega128RFA1 | 32-bit Cortex-M0 | Tensilica L106 32-bit |
| **Memory:** |  |  |  |  |  |  |
| RAM | 50 KB | 520 KB | 16KB | 32KB | 128 KB |
| Flash | 2 MB | 4 MB | 128 KB | 256 KB | 4 MB |
| **Interfaces** |  |  |  |  |  |  |
| SPI | ✔ | ✔ | ✔ | ✔ | ✔ |
| I²S | ✔ | ✔ | ✔ | ✘ | ✔ |
| I²C | ✔ | ✔ | ✔ | ✔ | ✔ |
| UART | ✔ | ✔ | ✔ | ✔ | ✔ |
| GPIO | ✔ | ✔ | ✔ | ✔ | ✔ |
| DAC | 1 × 10-bit | 2 × 8-bit | 8 × 10 bit | 1 × 10-bit | ✘ |
| ADC | 1 × 10-bit | 8-bit | 8-bit | 8/10/12 bit | ✔ |
| **Size** |  | 18 mm x 20 mm | 25.2mm x 18mm | NA | 61.5 mm x 25 mm | 48mm x 25mm |
| **Price** |  | 3.32 € | 4.42 € | NA | $34.99 | 14.00 лв |

Табл 2.2 Сравнение на хардуер за сензорно звено

### 2.3.2 Избор на софтуерна платформа

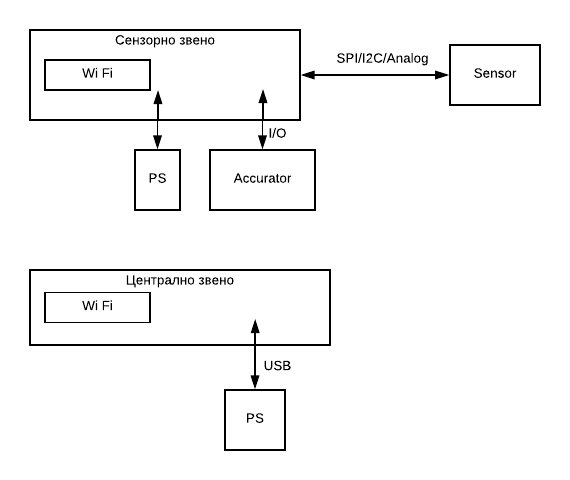
1. Софтуерна платформа на цетралното звено - Raspbian

Raspbian е Linux дистрибуция. Вместо нова операционна система Raspbian е модифицирана версия на популярното дистрибуторство Debian Squeeze Wheezy. Той се изпълнява на патентована версия на Linux ядрото. Операционната система е в активно развитие. Тя е силно оптимизирана за нископроизводителните ARM процесори на линия Raspberry Pi. Raspbian е компилирани пакети с изходен код, взети от отделните хранилища на съответните пакет. Най-важната разлика на Raspbian е, че е изградена с поддръжка на твърда плаваща точка, която драстично подобрява производителността. Пакетите обикновено се предоставят като източник и могат да се компилират с всеки компилатор, в който случай всеки пакет (очевидно понастоящем в района на 35000) е трябвало да се построи специално за Raspberry Pi, използвайки компилатор с твърд флоат (и някои други оптимизации ).**[10]**

2.Софтуерна платформа на сензорното звено – Lua

Lua е мощен, ефикасен, лек, скриптов език. Поддържа процедурно програмиране, обектно-ориентирано програмиране, функционално програмиране, програмиране, базирано на данни, и описание на данните. Използва се в много индустриални приложения (напр. Adobe Photoshop Lightroom), с акцент върху вградените системи (напр. Middleware Ginga за цифрова телевизия в Бразилия) и игри (например World of Warcraft и Angry Birds). **[11]**

## 2.4 Проектиране на блокови схеми на отделните звена



Фиг 2.3 Блокова схема на централното и сензорното звено

Сензорното звено има вграден Wi Fi модул, с който ще осъществява връзка с централното звено. На някой от следните три интерфейса – SPI, I2C, Analog – ще бъде свързан самия сензор, който ще отчита информацията за времето. За захранването има два варианта – от батерии или с DC адаптер. Централното звено разполага само с вграден Wi Fi модул и DC адаптер за захранване.

# ТРЕТА ГЛАВА

# ПРОЕКТИРАНЕ НА ПРОГРАМНОТО ОСИГУРЯВАНЕ НА ПРОДУКТА

## 3.1 Проектиране на комуникацията между отделните звена

Комуникацията между отделните звена ще се осъществява по следния начин: централното звено публикува съобщение към брокера. Брокера го получава и го изпраща на заинтересуваните клиенти, които са се абонирала за дадената тема, на която е публикувано съобщението. Сензорното звено получава съобщението и изпраща на брокера потвърждение, което да предаде на централното звено, за да може то да знае, че изпълнението съобщението било успешно .

Етапите, през които системата ще минава са два – „Discover” режим и нормален режим на работа. При стартиране на системата се изпълнява първо „Discover” режима, с който системата се конфигурира. След успешна конфигурация звената преминават към нормален режим на работа.

### 3.1.1 Дървовидна струткура на съобщенията

Всяко сензорно звено се абонира за темата със заглавие system\_name/nodeID/config, където nodeID е техния уникален индентификатор. По този начин няма опасност сензонрото звено да получи съобщение, което да е предназначено за друго сензорно звено. Дадено сензорно звено не знае за съществуването на други сезнорни звена. Неговата работа е да обработва съобщенията от централното звено, които са предназначени за него.

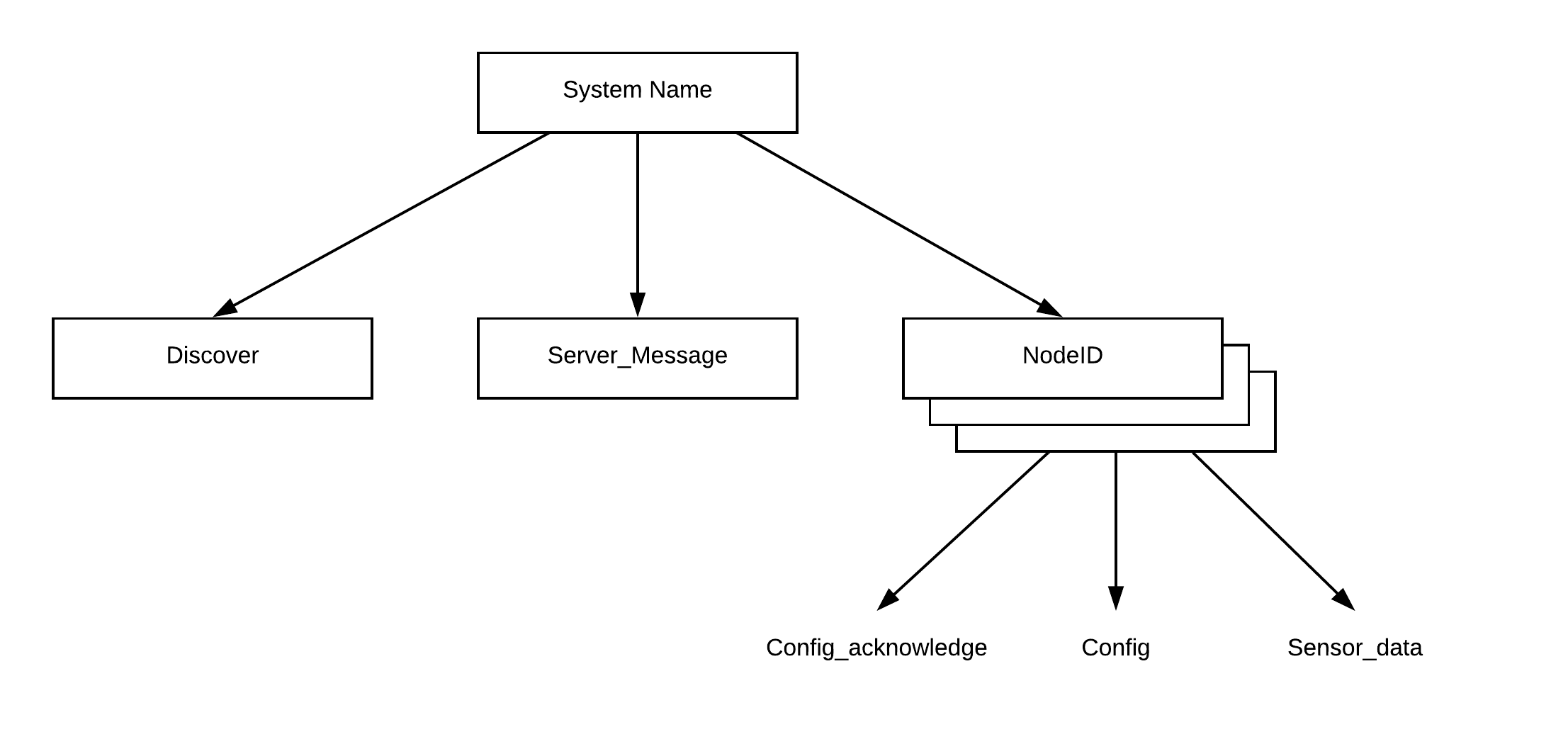
Централното звено в началото се абонира само за темата със заглавие /system\_name/discover и чака докато получи съобщение. При наличието на съобщение, централното звено разполага с уникалния индентификатор на сензорно звено, което е в същата мрежа. Към настоящия момент звеното може да се абонира за следните теми със заглавия system\_name/nodeID/data и system\_name/nodeID/config\_ack.

На темата system\_name/nodeID/data централното звено ще получава съобщение с информация за температурата и влажността на помещението, в което се намира дадено сензорно звено. На темата system\_name/nodeID/config\_ack звеното ще получава потвърждение за успешно изпълнение на съобщение съответно от темата system\_name/nodeID/config.

На темата system\_name/nodeID/config сезнорното звено получава информация, с която да се конфигура. Информацията, която съдържа едно съобщението за темата config, са режим на опциите на работа и съответните параметри за дадения режим. Видовете режим на опциите на работа са „Без работа“, „Сензор“, „Реле“ и „Сензор и Реле“. Опцията „Без работа“ се задава по подразбиране, когато дадено сензорно звено бъде регистрирано от централното. Единствената работа на звеното е да се събужда периодично, за да проверява дали няма съобщение от централното звено за него. Опциите „Сензор“, „Реле“ и „Сензор и Реле“ управляват съответно сензора, релето и сенозра и релето на звеното. Параметърът, който е необходим за всеки режим на опцията, е време за сън, който определя зколко ще е продължителността на енергоспестяващия режим на звеното, а за режима на опциите „Реле“ и „Сензор и Реле“ е неоходим допълнителен параметър, който да управлява релето. Този параметът също се задава от потребителя, за да може да регулира температурата в ададено помещение.

Центалното звено се абонира за още една тема със следното заглавие - /system\_name/server\_message. На тази тема централното звено ще получава съобщение от сървъра. Съобщението може да съдържа информация за довабяне на сензорно звено, премахване на сензорно звено или конфигуриране на дадено сензорно звено.

На фигура 3.1.1 е показана дървовидна структура на съобщенията.



Фигура 3.1.1 Topic Tree

### 3.1.2 Discover режим

Първата работа на сензорното звено е да осъществи връзка с мрежата, в която се намира брокера. След успешна установена комуникация следващата задача е да се свърже към смия брокер, който ще отговаря за комуникацията между сензорното звено и централното звено. Брокерът е ядрото на всеки publish/subscribe протокол. В зависимост от конкретнaта имплементация, брокерът може да се справи с хиляди съпътстващо свързани MQTT клиенти.

За целта сензорното звено инициализира глобален MQTT клиент обект с функцията mqtt.Client(), която като параметри приема уникален индентификатор за обекта, време за свъзраване (conenection timeout), потребителско име и парола, които брокера ще поиска при опит за осъществяване на връзка с него. За да бъде осъщесвена връзка с брокера ползвам функцията mqttBroker:connect(), която приема като параметри IP адреса на брокера, порта, на който слуша брокера, (тези параметри трябва да са известни на сензорното звено) функция, която да бъде извикана след успешно осъществена връзка, в случая се извиква функцията mqttSub(), и функция, която да бъде изпълнена в случай когато връзка с брокера не може да бъде осъществена. При успешно осъществяване на връзка с брокера функцията връща true, в противен случай fale.

*-- Establish a connection to the MQTT broker with the configured parameters.*

*function conn()*

*print("Making connection to MQTT broker")*

*mqttBroker:connect(tgtHost, tgtPort, 0, function(client) print ("connected")*

*mqttSub() end, function(client, reason) print("failed reason: "..reason) end)*

*end*

Фиг 3.1.2 Функция за осъществяване на връзка с MQTT брокера

Във функцията mqttSub() сензорното звено се абонира, ползвайки mqttBroker:subscribе(), за темата със следното заглавие – /system\_name/nodeID/config. Функцията mqttBroker:subscribе() приема като аргументи – темата, за която звеното ще бъде абонирано, целочислено число определящо качеството на обслужване и функция, която да бъде да извикана след успешното абониране (в случая това е main() функцията.

*function mqttSub()*

*mqttBroker: subscribe(configTopic = 0,*

*function(conn)*

*print("Subscribing for topic: " .. configTopic)*

*main()*

*end)*

*end*

Фиг. 3.1.3 Функция за абониране

Качеството на обслужване (QoS level) отговаря за начина на доставяне на съобщение. Нулата обозначава, че съобщението ще бъде доставено най-много веднъж, единицата, че съобщението ще бъде доставено поне веднъж, а двойката, че съобщението ще бъде доставено точно един единствен път

От фиг. 3.1.3 се вижда, че след абонирането сензорното звено извивка main() функцията, където първото нещо, което се изпълнява, е публикуването на уникалния индентификатор на сензорното звено , чрез извикване на функцията pubEvent(), която приема два аргумента – съобщението, което ще бъде публикувано, и темата, на която да го изпрати. Освен тази два аргумента на самата функция, която отговаря за самото публикуване на съобщение, се подават още два аргуметна - QoS level и retain flag. Този процес се повтаря на всяка секунда докато сензорното звено не получи съобщение от централното звено, с което да прекъсне процеса.

*-- Function pubEvent() publishes the data to the defined queue.*

*function pubEvent(pubValue, topicQueue)*

*-- Print a status message*

*print("Publishing to " .. topicQueue .. ": " .. pubValue)*

*mqttBroker:publish(topicQueue, pubValue, 0, 0) -- publish*

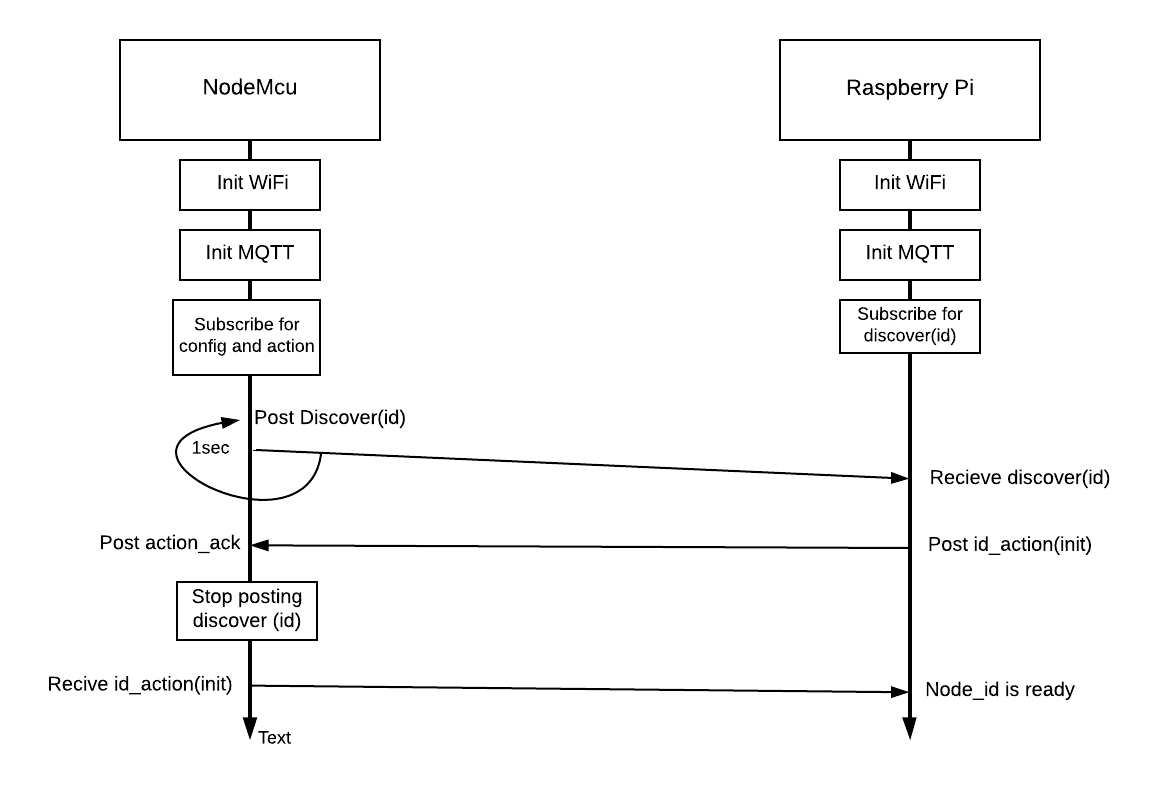
*end*

Фиг. 3.1.4 Функцията за публикуване на съобщения

В същото време от страна на централното звено неговата първа задача е да осъществи интернет връзка, било то с етернет кабел или WiFi. Следващата стъпка е да се свърже с брокера, който се намира на самото звено. За целта първо се създава MQTT клиент с фукнцтията MQTTClient\_create(), който да бъде готов за свързване към брокера. Функцията приема като аргументи указател към клиент от тип MQTTClient handle (валидна препратка към клиента), терминиран стринг, който специфира IP адреса на брокера, кой който ще се свърже клиента (в случая, понеже брокера е на същата машина, IP адреса е 127.0.0.1), уникален индентификатор на клиента, вид на persistence, който ще бъде използван от клиента ( в случая   
MQTTCLIENT\_PERSISTENCE\_NONE. Ако машината или системата, на която клиентът работи, се изключи или блокира текущото състояние на всички съобщения в процес на изпращане или получаване се губи и някои съобщения може да не бъдат доставени) и persistence\_context ( в случай, че клиентът използва MQTTCLIENT\_PERSISTENCE\_NONE persistence, този аргумент не се използва и неговата стойност трябва да бъде NULL).

За да осъществи връзка с броека централното звено използва функцията MQTTClient\_connect(), на която подава валиден клиентски индентификатор и указател към валидна структура от тип MQTTClient\_connectOptions. Структурата дефинира няколко конфигурации, които контролират начина, по който клиентът се свързва към MQTT сървъра. Ключовите данни в структурата, която звеното ползва, са потребителското име и паролата за брокера. Централното звено проверява дали успешно е осъществена комуникация като проверява дали върната стойност от изпълнението на фукнцията е MQTTCLIENT\_SUCCESS.

Следващата стъпка на централното звено е да се абонира за темата със заглавие /system\_name/discover. Oсъществява се с функцията MQTTClient\_Subscribe(), която приема следните аргументи - валидна препратка към клиента, темата, за която звеното да се абонира, и качеството на обслужване. След успешното абониране централното звено чака докато получи съобщение с уникалния индентификатор на някое сензорно звено. При получаване на ID-то, звеното се абонира за темите /system\_name/nodeID/data и /system\_name/nodeID/config\_ack. След успешно абониране за дадените теми централното звено изпраща съобщение инит на /system\_name/nodeID/config с функцията MQTTClient\_publishMessage(), която приема аргументи валидна препратка към MQTT клиент, темата, на която да бъде публикувано съобщението, дължина на съобщението, указател към съобщението, качество на обслужването, the съобщението и указател към променлива от тип MQTTClient\_deliveryToken (This is populated with a token representing the message when the function returns successfully)



Фиг. 3.1.4 Времева диаграма на Discover режима

В инит съобщението централното звено задава на сензорното звено режим на опция „Без работа“ и време за сън 5 минути. След получаването на съобщението сензорното звено спира да публикува съобщението за discover темата и заспива. Преди да премине към енергоспестяващия режим изпраща съобщение на /system\_name/nodeID/config\_ack, което съобщава на централното звено, че модула е готов за работа. Сензорното звено преминава към нормален режим на работа.

### 3.1.3 Нормален режим на работа

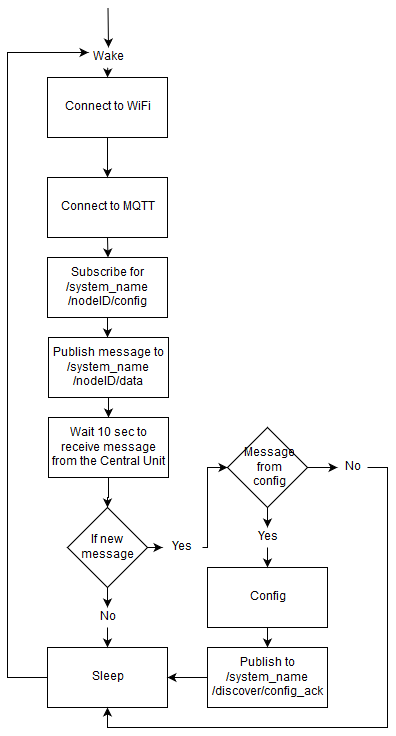
След успешно приключване на discover режима сензорното звено преминава към нормален режим на работа. Нормалния режим се дефинира със следните функции – wake(), getSensorData(), sendMessage(),messageArrived (), configMessage(), sleep().

Функцията wake() събужда процесора и модула за WiFi на сензорното звено, осъществява връзка с брокера и се абонира отново за темата със заглавие /system\_name/nodeID/config. Aко режима на опциите е “Сензор“ или „Сензор и реле“ се извиква функцията getSensorData(), с която се взима информация за температурата и влажността на помещението, в което се намира звеното, и се записва. Извиква се функцията pubEvent(), на която се подават данните от сенозра и темата /system\_name/nodeID/data.

Но ако режима на опциите е „Реле“ или „Без работа“ функцията няма бъде извикана getSensorData(). Но следва изикването на . Целта на съобщението е да уведоми централното звено, че даденото сензорно звено се е събудило.

Сензорното звено изчаква 10 секунди, за да има време централното звено да му ипзрати съобщение. Ако в този период звеното получи съобщение, ще се изпълни функцията messageArrived(), в противен случай звеното направо преминава към енергоспестяващия режим. Фунцкията messageArrived() обработва съобщението и го изпраща на функцията configMessage(), която поема изпълнението на съобщението. След успешното приключване на изпълненито сензорното звено изпраща съобщение на темата със заглавие /system\_name/nodeID/config\_ack, с което централното звено разбира, че процеса на изпълнение е успешен.

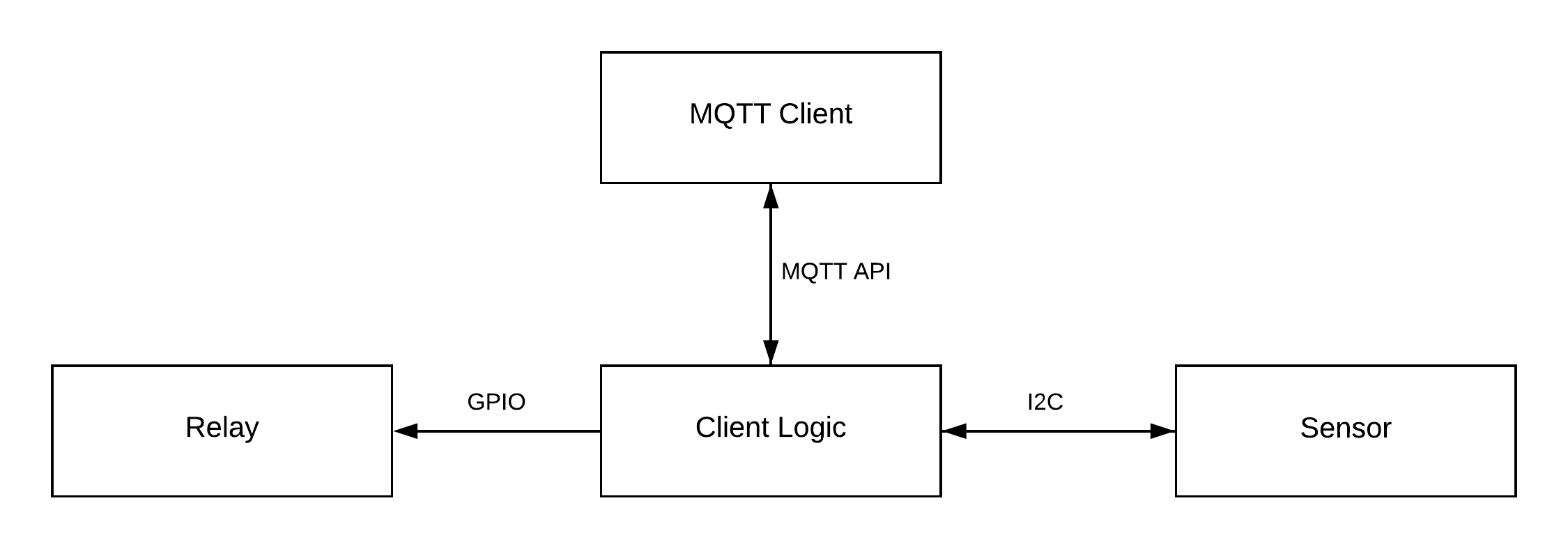
След това сензорното звено преминава към енергоспестяващия режим докато не стане време да се „събуди“. Целия този алгоритъм, описан на фиг. 3.1.5, се изпълнява отново и отново докато сензорното звено не бъде премахнато от уеб интерфейса от потребителя.



Фиг. 3.1.5 Алгоритъм на normal operation режима

## 3.2 Проектиране на сензорното звено

Компонентите, които съставят сензорното звено са следните – MQTT клиент, който е отговорен за получаването и изпращането на съобщения от/към брокера, клиентска логика, която поема съобщенията, тяхната обработка и управлението на сензора и релето, сензор, с който се отчита информация за помещението, и реле, с което се контролира температурата в даденото помещение. Комуникацията между MQTT клиента и логиката се осъществява по интерфейса MQTT API, между релето и логиката по GPIO (като пина на релето се конфигура да бъде вход), а между сензора и локигата по I2C интерфейса.



Фиг. 3.2.1 Статичен излгед на сензорното звено

### 3.2.1 Свързване на сензор към сезнорното звено

Сензора за температура и влажност, който използвам за целите на дипломна работата, е si7021. Модулът разполага с I2C интерфейс. I2C използва само две двупосочни отворени канализационни линии, Serial Data Line (SDA) и Serial Clock Line (SCL). SDA е свързан към GPIO4 на NodeMcu-то, който отговоря на пин с индекс 2, а SCL е свързан към GPIO5 на NodeMcu-то – с индекс на пина 1.

### 3.2.2 Получаване на данни от сензора

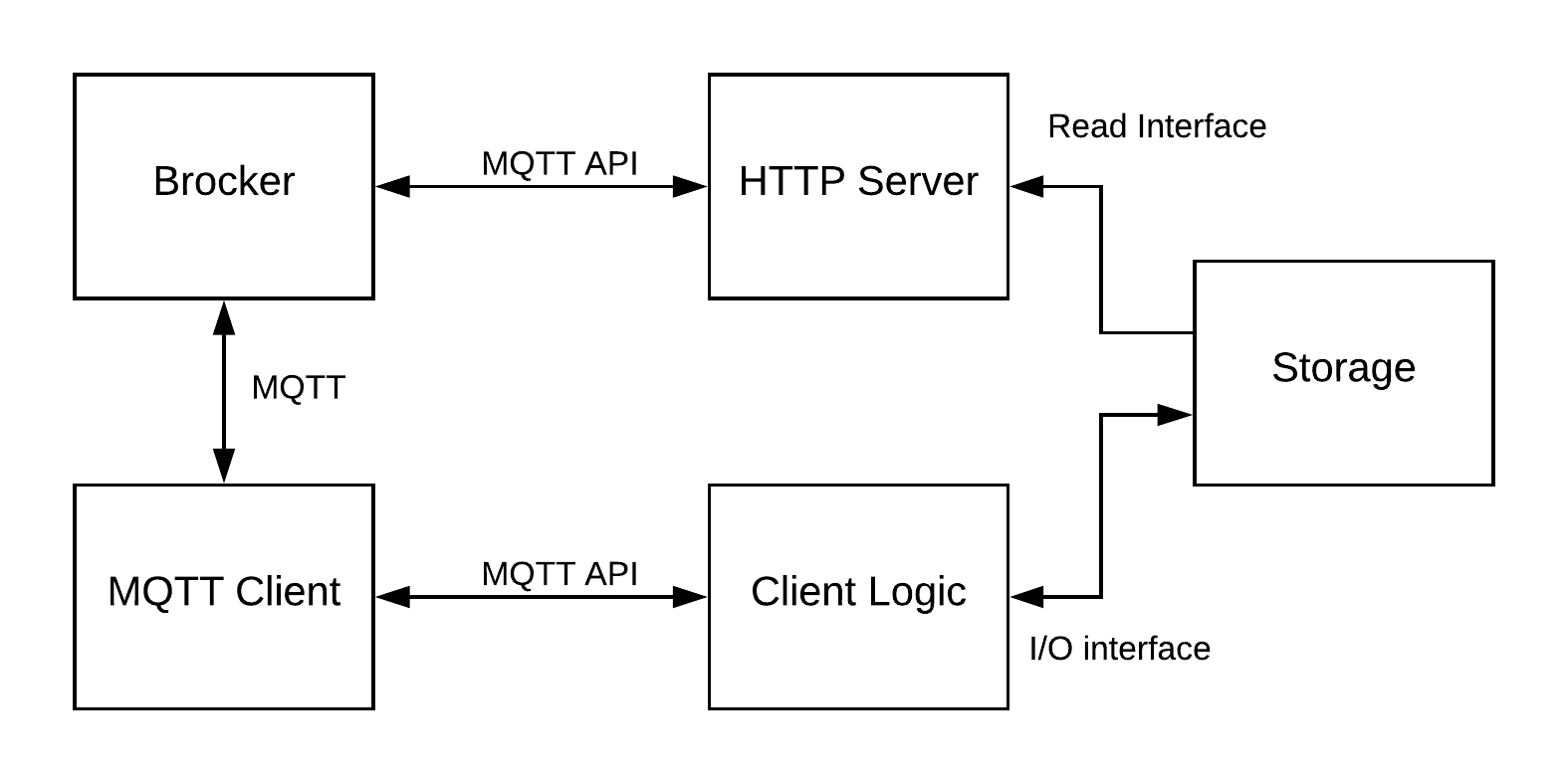
За да използва сезнорното звено прикрепения към него сензор използвам функцията Si7021.init(), с която звеното настройва I2C интерфейса на si7021, като за целта трябва да се подадат два параметъра – единият параметър е индекса на на пина, към който е свързан е SDA, а другият параметър е индекса на пина, към който е свързан SCL. След като сезнора е настроен данните могат да бъдат прочетени с функцията Si7021.read(). Тяхното получаване на сензорното звено става чрез две функции: si7021.getHumidity(), която взима информация за влажността, която сензора отчина и si7021.getTemperature(), взима взима информация за стойността на температурата.

### 3.2.3 Енергоспестяващ режим :

Сензорното звено разполага с 3 на брой енергоспестяващи режима - modem sleep, light sleep и deep sleep. Modem sleep режима спира WiFi предаването, но поддържа връзката и работата на процесора, Light sleep режима изключва WiFi-а и процесора, но се събужда периодично, за да получава съобщения, а deep sleep режима напълно изключва безжичната връзка и процесора, оставяйки само таймер, за да събуди процесора. Deep sleep режима изисква външна връзка между GPIO16 и RST, за да може да събуди чипа. Но това събуждане води до ресетване на чипа и програмата се рестартира.

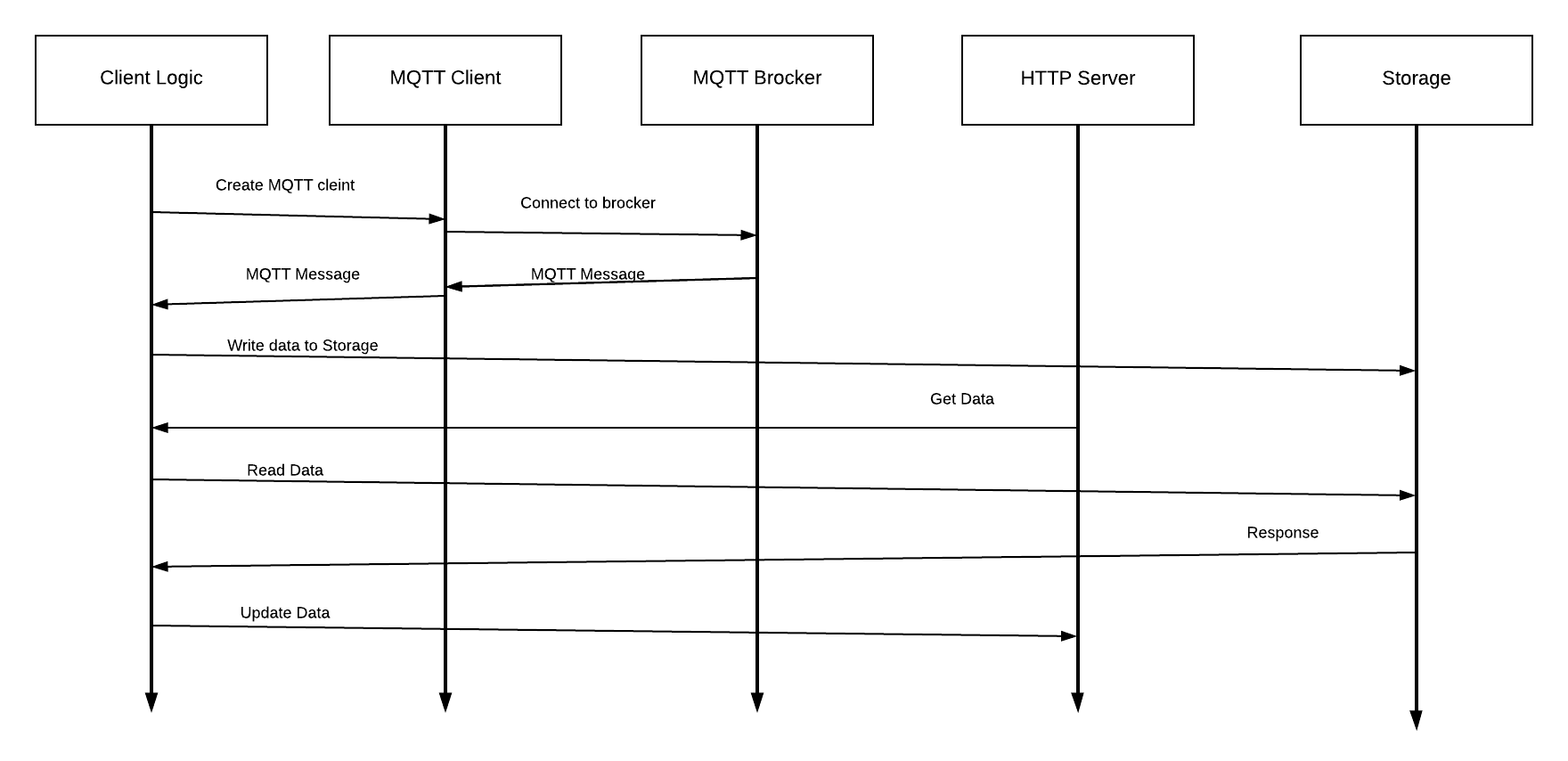
За целите на дипломната работа използвам light sleep режима, чрез функцията node.sleep(). Функцията приема единствен аргумент конфигурация, в която са дефинирани - продължителността на режима, пин, от който да бъде прекъснат режима и функция, която да се извика след събуждането на процеора и включването на безжичната връзка - wake(). Зададената по подразбиране продължителност е 5 минути. Потребителя има право да промени продължителноста. С изпълнението на wake() фунцкията сензорното звено осъществява връзка с брокера, абонира се отново за темата със заглавие /system\_name/nodeID/config, изпраща съобщение на темата /system\_name/nodeID/data, в което, ако на звеното е зададен режим „Сензор“ или режим „Сензор и реле“, ще има информация за температурата и влажността на помещението, но ако е зададен друг режиж, просто се съобщава, че звеното се е събудило.

### 3.3 Проектиране на Централното звено



Фиг. 3.3.1 Статичeн изглед на софтуера на централното звено

На фиг. 3.3.1 са дадени компонентите, които съставят централното звено и интерфейсите между тях. Брокера е софтуер, който се изпълнява като демон и се грижи за разпределянето на съобщенията между клиентите. MQTT клиента изпраща и получава дадените съобщения и ги предава на клиентската логика. Клиентската логика се грижи за обработката на съобщенията и управлението на различните компоненти. Хранилището пази информацията от сензорните звена, а HTTP сървъра отговаря за представянето на данните по подходящ начин на потребителя и предаването на команди от потребителя към логиката. HTTP сървъра може само да чете данни от хранилището, докато логиката може да чете и да записва инфомрация.

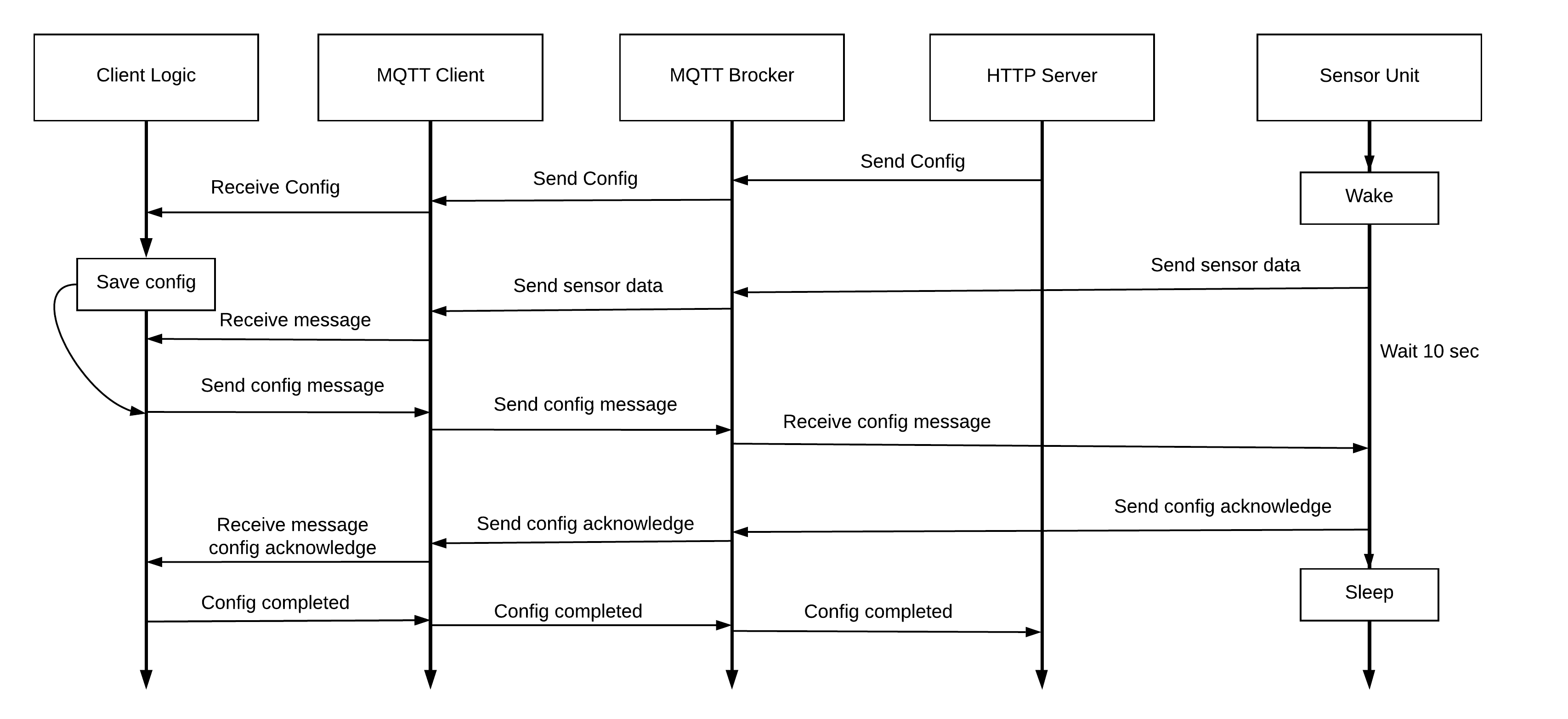


Фиг. 3.3.2 Динамичен изглед на централното звено

При стартирането на централното звено се създава MQTT клиент, който да осъществи връзка с брокера. Следва абониране за темите /system\_name/discover и system\_name/server\_message звеното пуска нишка, която да следи за съобщения от брокера.

При получаване на съобщение нишката обработва и определя от коя тема е съобщението и пуска нова нишка, която да се заеме с неговото изпълнението.

Когато потребителя поиска да добави сензорно звено, уеб интерфейса ще се обърне към сървръра, който ще се обърне към логиката, за да му предостави списък с всички налични звена от хранилището, които би могъл да добави. При избора на сезнорно звено, което да бъде добавено, потребителя задава конфигурация – име на звеното, на колко време да получава информация от сензорното режим на опцията и параметър за температурата, която да поддържа (в случай, че режима на опцията е различен от „Реле“ или „Сензор и Реле“ този параметър отсъства). Тази информация централното звено записва в инит съобщението и го публикува на темата /system\_name/nodeID/config. Целият този процес се изпълнява в отделна нишка. След публикуването на съобщението централното звено очаква да получи потвърждение от сензорното звено. След получаването на исканото съобщение, клиентската извиква функцията pthread\_exit(NULL), с която да да сложи край на текущата нишка. Към настоящия момент има една работеща нишка, която чака да бъде получено съобщението от брокера.



Фиг. 3.3.3 Предаване на конфигурация от сървъра към сензорно звено

На фиг. 3.3.3 е представена псоледователността на действията при задаване на конфигурация от потребителя за дадено сензорно звено. Сървъра изпраща съобщение до брокера, което се предава към MQTT клиента и после на клиентската логика. Логиката запазва конфигурацията в за даденото сензорно звено в буфер. Ако има нова конфигурация за даденото звено, то тя ще се запише на мястото на старата. Когато сензорното звено се събуди от енергоспестяващия режим изпраща съобщение до централното звено, с което да го уведоми, че е излезнал от режима. Ако има конфигурация за звеното, логиката я изпраща на брокера посредством MQTT клиента, а брокера на сензорното звено. В отговор звеното изпраща потвърждение за изпълнение на конфигурацията и заспива.

Ако няма съобщение за сензорното звено, то логиката не изпраща нищо. Звеното изчаква 10 секунди и при липса на съобщение заспива.

### 3.4 Проектиране на Уеб интерфейса

За проектирането на уеб интерфейса използвам езика ruby и фреймуърка Sinatra. Ruby e динамичен език за програмиране с отворен код, фокусиран върху лесната употреба и продуктивноста, притежава елегантен и лесен за четене и писане синтаксис.

Sinatra е безплатен и отворен фреймуърк за приложения написан на Ruby. Това е алтернатива на други Ruby уеб приложения като Ruby on Rails, Merb, Nitro и Camping. Sinatra не следва типичния модел модел-view-контролер, използван в други рамки, като Ruby on Rails. Вместо това фреймуърка се фокусира върху „бързото създаване на уеб приложения в Ruby с минимални усилия“.

Уеб интерфейса се състои от един руби и четири erb файла. ERB (Embedded RuBy) е функция на Ruby, която позволява удобно да се генерира всякакъв вид текст, във всякакво количество, от шаблони. Самите шаблони съчетават обикновен текст с Ruby код за променлива, замяна и контрол на потока, което ги прави лесни за писане и поддръжка. ERB файловете са realTime.erb, statistics.erb, config.erb и layout.erb.

Файла layout.erb съдържа html код, който отговаря за оформлението на уеб интерфейса. Във файловете realTime.erb, statistics.erb и config.erb са имплементирани функционалностите, който уеб интерфейса предоставя на потребителя - Данни на живо, Статистика и Конфигурация. Руби файла съдържа HTTP сървъра, който манипулира данните по подходящ начин прилага HTTP методи и комуникира с брокера.

Връзката между HTTP сървъра и клиентската логика се осъществява по следния начин: HTTP Сървъра създава MQTT клиент, който да осъществи връзка с брокера, клиента изпраща съобщение на темата със заглавие /system\_name/server\_message, което брокера предава на логиката, която се заема с неговото изпълнение. В съобщението може да има една от следните команди: добавяне на сензорно звено, конфигурация на сензорно звено или премахване на сензорно звено.

За добавяне на сензорно звено в съобщението трябва да има информация за режима на работа на сензорното звено и съответните параметри за режима. Премахването на звеното се осъществява като в съобщението за логиката е зададен режим IDLE

*<% @sensors.each do |sensor| %>*

*<div class=“span6“>*

*<div class=““ style=“max-width: 300px; padding: 19px 29px 29px; margin: 0 auto 20px; background-color: #fff; border: 1px solid #e5e5e5; -webkit-border-radius: 5px; -moz-border-radius: 5px; border-radius: 5px; -webkit-box-shadow: 0 1px 2px rgba(0,0,0,.05); -moz-box-shadow: 0 1px 2px rgba(0,0,0,.05);“>*

*<h3> <%= sensor[:name] %> </h3>*

*<h4> Temp: <%= sensor[:temp] %> </h4>*

*<h4> Humidity: <%= sensor[:hum] %> </h4>*

*</div>*

*</div>*

*<% end %>*

Фиг. 3.4.1 Фунцкионалност данни на живо

За предоставянето на информация за отделните сензорни звена от базата данни се извличат всички звена, чиито статус на опциите е различен от IDLE (Без работа), заедно с техните данни за помещението. Информацията за тях се записва в обекта sensors, който при визуализацията на данни за потребителя бива обходен като се взимат параметрите за името на сензорното звено, зададено от потребителя, и температурата и влажността на помещението, в което се намира звеното.

Функционалността „Статистика“ предоставя на потребителя възможност за дадено сензорно звено да получи информация за изменението на температурата и влажността в рамките на деня, на седмицата или на годината под формата на диаграма. Потребителя избира от уеб интерфейса сензорното звено, за което иска да получи информация. Сървъра взима информацията от базата данни за определения период (по подразбиране се взима информацията, която е отчетена в рамките на деня) и я записва в обектa data. Функционалността „Статистика“ предоставя две диаграми – едната за температура, а другата за влажност.

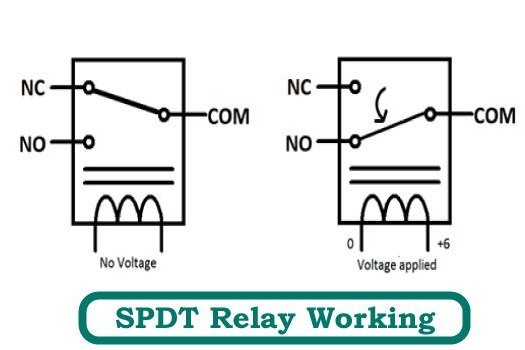
Функционалостта „Конфигурация“ предоставя на потребителя списък извлечен от базата данни за всички налични звена в мрежата на брокера, както и информация за състоянието и режима на звеното. При промяна на режим или параметрите на даден режим сървъра обработва и записва данните в съобщение, което да предаде на логиката посредством брокера.

# ЧЕТВЪРТА ГЛАВА

# ПРОЕКТИРАНЕ НА РАБОТОСПОСОБЕН МОДЕЛ

Сензорното звено хардуерно се изгражда от четири компонента – безжичен модул nodeMcu, сензор за температура и влажност si7021, реле и прототипна платка Breadboard.

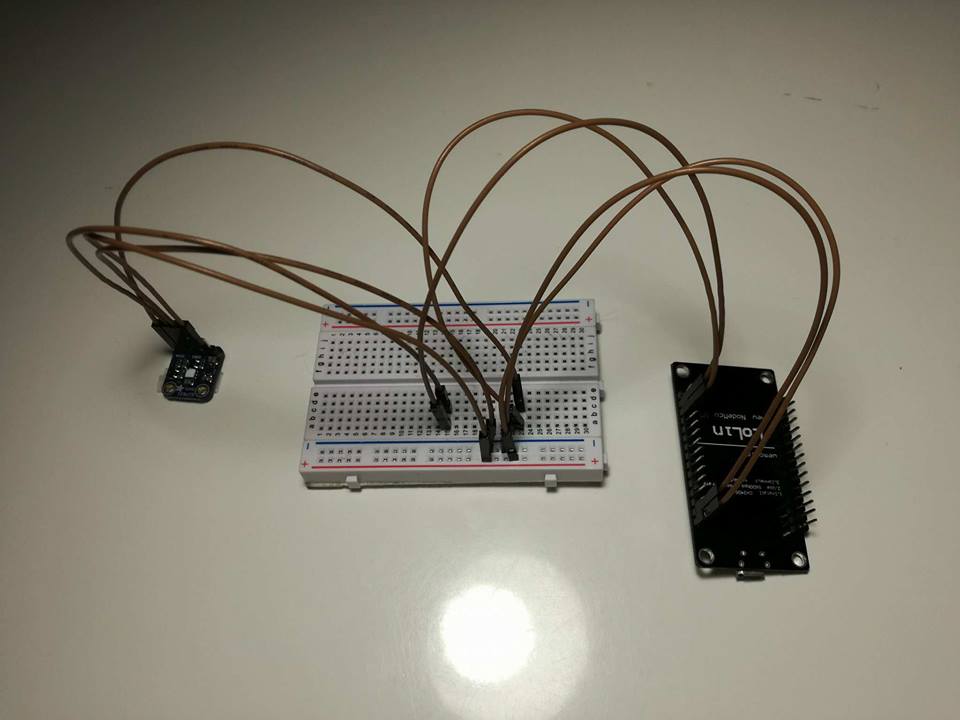
Релето е електронен превключвател, предназначен за комутация на електрическа верига. Основните части на електромагнитното реле са електромагнит, котва и превключвател. Електромагнитът представлява електрически проводник, намотан на бобина със сърцевина от феромагнитен материал. Котвата е пластина от магнитен материал, която чрез лостче управлява контактите. При пропускане на електричен ток през намотката на електромагнита възниква магнитно поле което притегля котвата към сърцевината и тя превключва контактите.



Фиг. 4.1 Начин на работа на релето

Свързване на хардуерните компоненти:

С джъпмери земята на безжичния модул, сензора и релето се свързват към минусово гнездо на прототипната платка, а тяхното захранване - към положително гнездо на платката. Релето разполага с пин вход, който е свързан към дигиталния пин D4 на безжичната платка (GPIO, който се здава да бъде изход). Сензора разполага с два пина SCL и SDL, които са свързани към дигиталните пинове съответно D1 и D2 на безжичната платка.



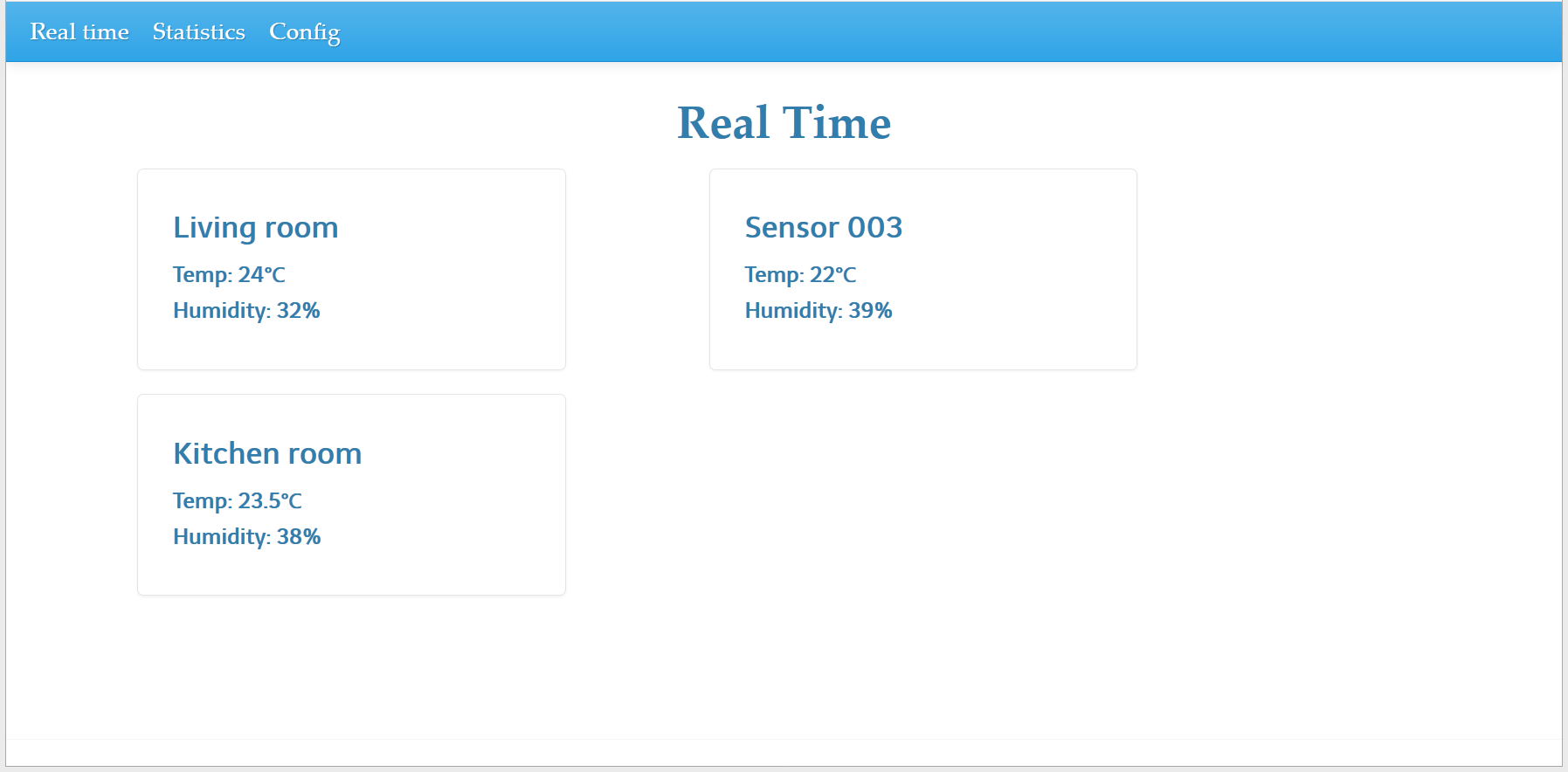
Фиг. 4.2 Работоспособен модел

# ПЕТА ГЛАВА

# РЪКОВОДСТВО НА ПОТРЕБИТЕЛЯ

## 5.1 Данни на живо

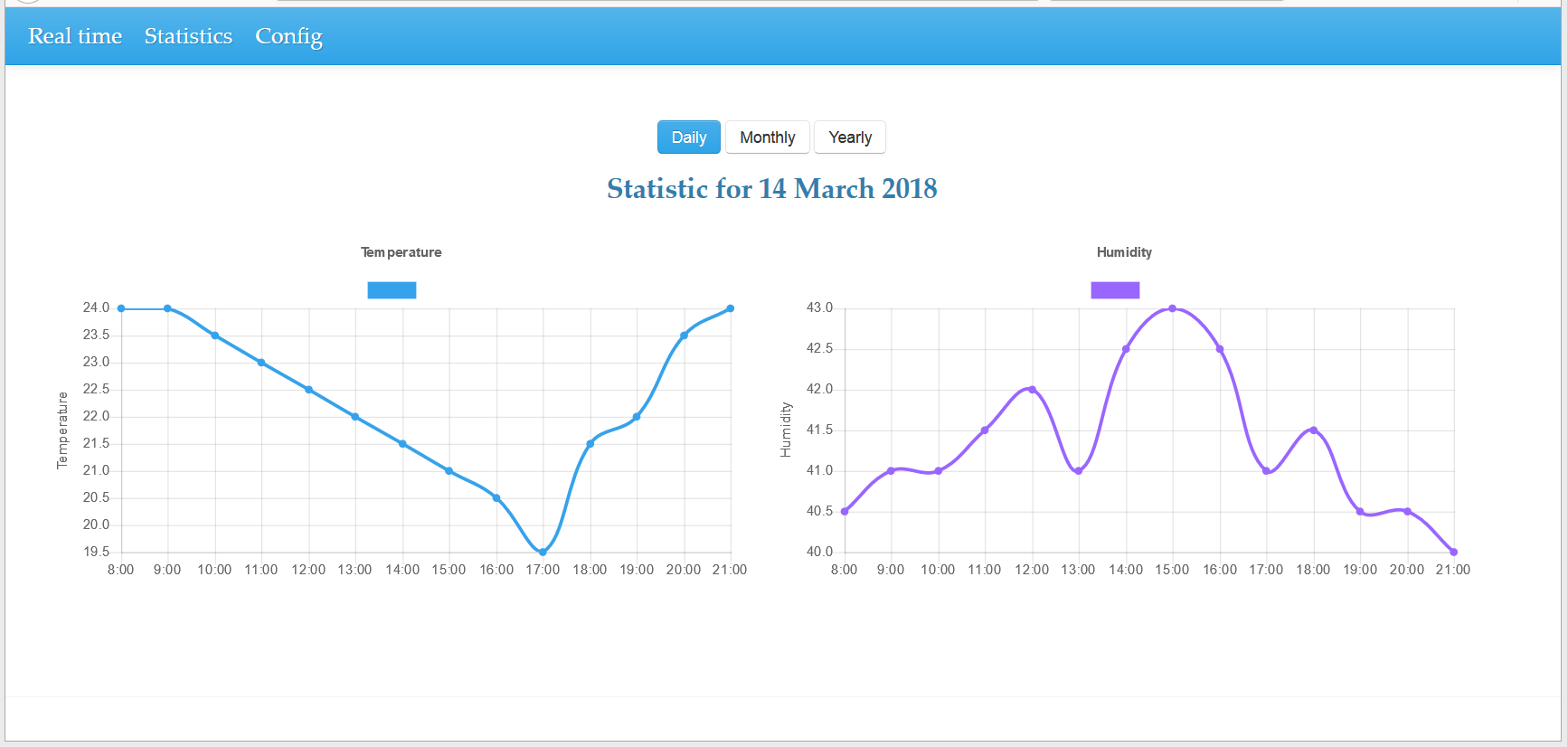
На фиг. 5.1 е показан изглед на функционалността данни на живо. Функционалността предоставя на потребителя списък от активните сезнорни звена. Всяко сензорно звено предоставя информация на потребителя за температура и влажност на даденото помещение, в което се намира звеното.



Фиг. 5.1 Данни на живо

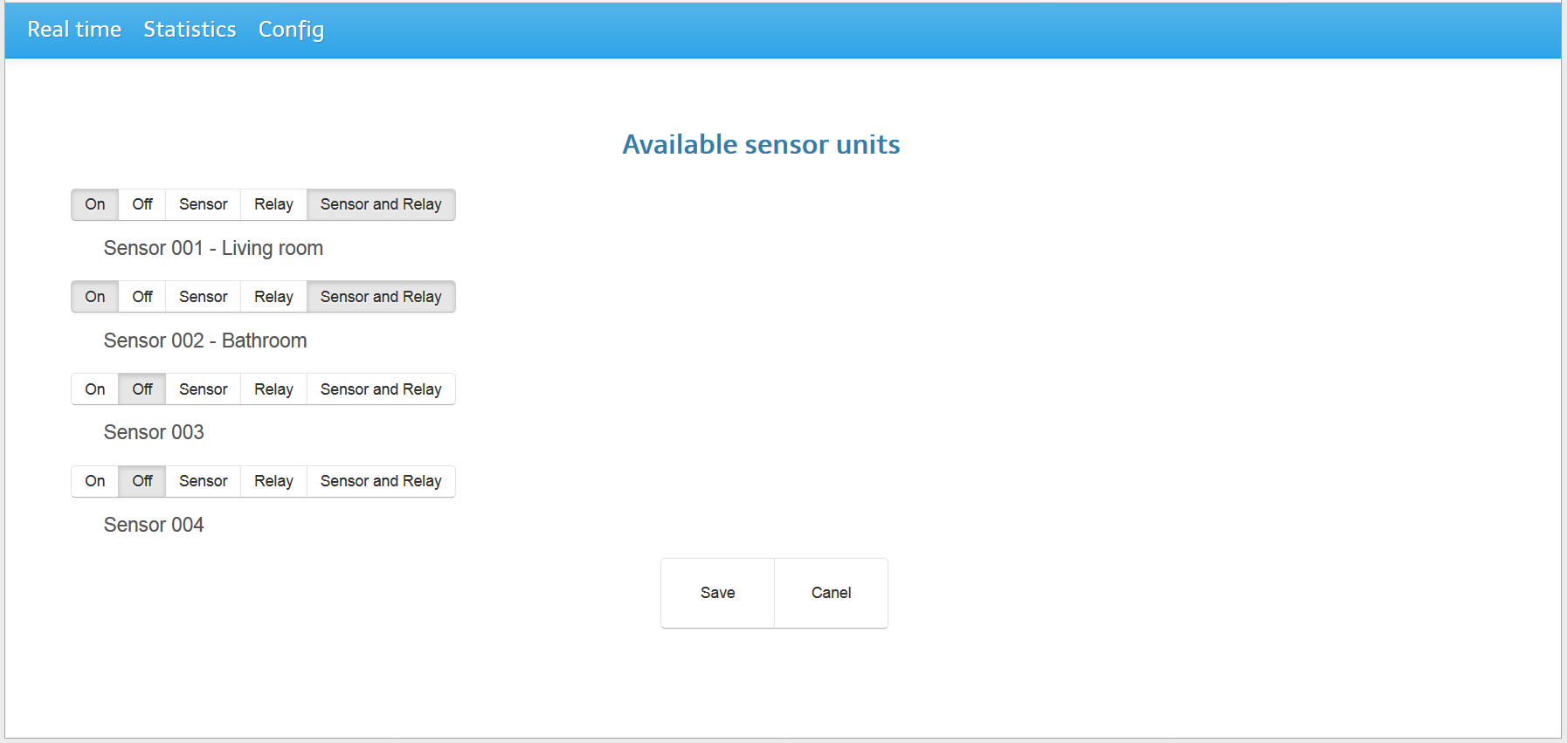
## 5.2 Статистика

Функционалността статистика е показана на фиг. 5.2. Тя предоставя потребителя две диаграми. Едната показва изменението на температурата за даден период, а другата изменението на влажността за същия период. Потребителя има възможност да проследи информацията в рамките на ден, седмица или година.



Фиг 5.2 Статистика за деня

## 5.3 Конфигурация



Фиг. 5.3 Конфигурация

Конфигурацията показана на фиг. 5.3 предоставя на потребителя списък с всички сензорни звена в мрежата на брокера, информация дали те са включени или не и в какъв режим на работа са. С бутоните On и Off потребителя лесно може да включва и изключва дадено звено, а с бутоните „Sensor”, „Relay” и „Sensor And Realy” да управлява режима на работа .

# ЗАКЛЮЧЕНИЕ

Основната цел на дипломната разбота е да улесни потребителя в неговото ежедневие с регулирането на параметрите за комфортност на неговия дом или офис . Настоящата разработка предоставя решенеие на проблема на потребителя чрез система за отдалечен контрол и наблюдение с изчистен и прост уеб интерфейс.

Като бъдещо развитие на системата са планирани следните функционалсности:

* Поддръжка на потребителски профил (предоставяне на данни за потребителския профил и тяхното управление)
* Мобилно приложение
* Поддръжка на други параметри като: осветеност, вентилация, повече от едно реле на дадено сензорно звено и т.н.
* Допълнително захранване (в случай, когато токът спре)

# ИЗПОЛЗВАНА ЛИТЕРАТУРА

**[1]** [https://nest.com/thermostats/nest-learning-thermostat/tech-specs/#](https://nest.com/thermostats/nest-learning-thermostat/tech-specs/)

**[2]** <https://coolautomation.com/products/coolmasternet/#link_tab-1417938236-1-25>

**[3]** <https://sensicomfort.com/shop/products/thermostats/touch-thermostat>

**[4]** <https://www.tado.com/bg/products/smart-thermostat-starter-kit>

**[5]** <http://www.pro-domotic.com/en/content/home-automation-heating-air-conditioning>

**[6]**<https://bg.wikipedia.org/wiki/%D0%9B%D0%BE%D0%BA%D0%B0%D0%BB%D0%BD%D0%B0_%D0%BC%D1%80%D0%B5%D0%B6%D0%B0>

**[7]** <http://news.laptop.bg/statii/bezzhichni-internet-mrezhi-wi-fi-vsichko-za-t/>

**[8]** <http://javac.bg/?p=301>

**[9]** <https://dev.bg/open-internet-of-things-%D1%87%D1%80%D0%B5%D0%B7-mqtt/>

**[10]** <https://en.wikipedia.org/wiki/Raspbian>

**[11]** <https://www.lua.org/about.html>

# ПРИЛОЖЕНИЕ

# СЪДЪРЖАНИЕ

[УВОД 4](#_Toc509479636)

[ПЪРВА ГЛАВА 5](#_Toc509479637)

[ПРЕГЛЕД НА ПОДОБНИ ПРОДУКТИ, СЪЩЕСТВУВАЩИ РЕШЕНИЯ И ПОДХОДЯЩИ ТЕХНОЛОГИИ 5](#_Toc509479638)

[1.1 ПРЕГЛЕД НА ПОДОБНИ ПРОДУКТИ 5](#_Toc509479639)

[1.1.1 Nest Learning Thermostat 5](#_Toc509479640)

[1.1.2 CoolMasterNet 7](#_Toc509479641)

[1.1.3 Sensi™ Touch Wi-Fi Thermostat 9](#_Toc509479642)

[1.1.4 Tado Smart Thermostat 10](#_Toc509479643)

[1.1.5 Pro-domotic 12](#_Toc509479644)

[1.2 СРАВНЕНИЕ НА СЪЩЕСТВУВАЩИ РЕШЕНИЯ 14](#_Toc509479645)

[1.3 ПОДХОДЯЩИ ТЕХНОЛОГИИ 15](#_Toc509479646)

[1.3.1 Local area networks 15](#_Toc509479647)

[1.3.2 Wi Fi 16](#_Toc509479648)

[1.3.2 IP communication ( sockets) 18](#_Toc509479649)

[1.3.3 MQTT 19](#_Toc509479650)

[ВТОРА ГЛАВА 21](#_Toc509479651)

[ИЗИСКВАНИЯ КЪМ ПРОДУКТА, ОБЩА СХЕМА НА ПРОДУКТА, ИЗБОР НА КОМПОНЕНТИ И ПРОЕКТИРАНЕ НА БЛОКОВИ СХЕМИ НА ОТДЕЛНИТЕ ЗВЕНА 21](#_Toc509479652)

[2.1 Изисквания към програмния продукт 21](#_Toc509479653)

[2.1.1 Хардуерни изисквания 21](#_Toc509479654)

[2.1.2 Софтуерни изисквания 21](#_Toc509479655)

[2.2 Общи схеми на продукта 22](#_Toc509479656)

[2.3 Избор на компоненти 23](#_Toc509479657)

[2.3.1 Избор на хардуерна платформа 23](#_Toc509479658)

[2.3.2 Избор на софтуерна платформа 25](#_Toc509479659)

[2.4 Проектиране на блокови схеми на отделните звена 26](#_Toc509479660)

[ТРЕТА ГЛАВА 28](#_Toc509479661)

[ПРОЕКТИРАНЕ НА ПРОГРАМНОТО ОСИГУРЯВАНЕ НА ПРОДУКТА 28](#_Toc509479662)

[3.1 Проектиране на комуникацията между отделните звена 28](#_Toc509479663)

[3.1.1 Дървовидна струткура на съобщенията 28](#_Toc509479664)

[3.1.2 Discover режим 30](#_Toc509479665)

[3.1.3 Нормален режим на работа 35](#_Toc509479666)

[3.2 Проектиране на сензорното звено 37](#_Toc509479667)

[3.2.1 Връзване на сензор към сезнорното звено 37](#_Toc509479668)

[3.2.2 Получаване на данни от сензора 37](#_Toc509479669)

[3.2.3 Енергоспестяващ режим : 38](#_Toc509479670)

[3.3 Проектиране на Централното звено 39](#_Toc509479671)

[3.4 Проектиране на Уеб интерфейса 42](#_Toc509479672)

[ЧЕТВЪРТА ГЛАВА 45](#_Toc509479673)

[ПРОЕКТИРАНЕ НА РАБОТОСПОСОБЕН МОДЕЛ 45](#_Toc509479674)

[ПЕТА ГЛАВА 46](#_Toc509479675)

[РЪКОВОДСТВО НА ПОТРЕБИТЕЛЯ 46](#_Toc509479676)

[5.1 Данни на живо 46](#_Toc509479677)

[5.2 Статистика 46](#_Toc509479678)

[5.3 Конфигурация 47](#_Toc509479679)

[ЗАКЛЮЧЕНИЕ 48](#_Toc509479680)

[ИЗПОЛЗВАНА ЛИТЕРАТУРА 49](#_Toc509479681)

[ПРИЛОЖЕНИЕ 50](#_Toc509479682)

[СЪДЪРЖАНИЕ 51](#_Toc509479683)