OULUN AMMATTIKORKEAKOULU

Kalle Vuorinen

PLANTUI PRON TUOTEKEHITYS JA MIKROKONTROLLEREIDEN OHJELMOINTI JA KÄYTTÄMINEN KOMENTOYKSIKKÖNÄ

PLANTUI PRON TUOTEKEHITYS JA MIKROKONTROLLEREIDEN OHJELMOINTI JA KÄYTTÄMINEN KOMENTOYKSIKKÖNÄ

Kalle Vuorinen
Opinnäytetyö
Kevät 2017
Tietotekniikan koulutusohjelma
Hyvinvointiteknologian suuntautumisvaihtoehto
Oulun ammattikorkeakoulu

TIIVISTELMÄ

Oulun ammattikorkeakoulu
Tietotekniikan koulutusohjelma, hyvinvointiteknologian suuntautumisvaihtoehto

---

Tekijä: Kalle Vuorinen
Opinnäytetyön nimi: Plantui Pron tuotekehitys ja mikrokontrollereiden ohjelmointi ja käyttäminen komentoyksikkönä
Työn ohjaajat: Jukka Jauhiainen ja Kaisa Orajärvi
Työn valmistumislukukausi ja -vuosi: 2017 kevät Sivumäärä: 65

Opinnäytetyö toteutettiin koosteopinnäytetyönä, jota otettiin kokeiluun tietotekniikan koulutusohjelmassa vuodesta 2014 alkaen. Opinnäytetyön ensimmäisen osan aiheena oli tutustua mikrokontrollereiden ohjelmointiin ja käyttöön komentoyksikkönä. Tämän lisäksi työssä käsiteltiin erilaisia mikrokontrolleripohjia sekä sitä, mistä osista mikrokontrolleri koostuu. Opinnäytetyön ensimmäisessä osassa tutustutaan mikrokontrollereiden ohjelmoinnin perusteisiin Arduinon avulla.

Ensimmäisen osan tarkoitus oli tutustua jonkin itseä kiinnostavan osa-alueen teoriaan. Työssä käytiin erilaisia sensoriteknologisia ja sulautettuja järjestelmiä läpi saatavilla olevien lähteiden kautta. Ensimmäisen osan kirjoittaminen oli mielenkiintoista, sillä aikaisemmin oli ollut vain käytännön ymmärtämistä aihealueesta ja nyt sen ympärille kehittyi teoriaosaaminen. Työnjälkeen ymmärtää paremmin kokonaiskuvan mikrokontrollereiden rakenteesta ja toiminnasta.

Opinnäytetyön yhdistetyn toisen ja kolmannen osan aiheena oli kehittää Plantui Smart Gardenista IoT-prototyyppi. Työ toteutettiin Plantui Oy:lle yritysprojektin aiheesta. Yritys oli suunnittelemassa laitteellensa älykästä järjestelmää tukemaan sen toimintoja. Toinen ja kolmas osa keskittyy laitekehitykseen, sovelluskehitykseen sekä projektinhallintaan Plantui Pron kehitystyössä.

Työssä saatiin käytännönkokemusta siitä, mitä projektityö on ja millaisia haasteita useamman henkilön kanssa työskentely tuo. Työssä piti ottaa huomioon kaikkien kehittyminen omalla osa-alueellaan ja sen, että jokaista uutta ominaisuutta vasten piti ensin tehdä tutkimustyötä ja kokeiluja ennen kuin pystyi konkretisoimaan asian prototyyppikehitykseen. Opinnäytetyön lopputuloksena toteutettu prototyyppi on yksi mallin laite- sekä sovelluspuolen mahdollisuuksille, joita yritys voi halutessaan hyödyntää kehittäessään kaupallisen version mallista.

Asiasanat: Asioiden Internet, IoT, tuotekehitys, web-kehitys, laitekehitys, Scrum, projektinhallinta

ABSTRACT

Oulu University of Applied Sciences
Degree programme in Information Technology and Telecommunication,
Medical Engineering

Author: Kalle Vuorinen
The names of parts of the thesis: Plantui Pro development and programming of micro controllers and their use as a command unit
Supervisors: Jukka Jauhiainen, Tuula Hopeavuori ja Kaisa Orajärvi
Term and year when the thesis was submitted: Spring 2017 Number of pages: 65

The thesis was carried out in three parts, which was taking to test use in Information Technology degree programme starting from 2014. The subject of the first part of thesis was to do research and understand how micro controllers can be programmed and how a micro controller can work as a controller in a system. Thesis will also cover different micro controller boards and what kind of components does a micro controller include and how controllers can be programmed.

The purpose of the first part was to get theoretical understanding of an issue that interests the author. Various sensor technological and embedded systems where asserted in the first part. Writing the first part was interesting because before the author had only knowledge on practical level about the subject and now the practical knowledge gained theoretical context to support the professionalism further. After the first part the author has more profound understanding of micro controllers and how they work. The subject felt good and it guided author's choice for the parts two and three.

The subject of the combined second and third part of the thesis was to develop a working IoT prototype from Plantui Smart Garden. The thesis was carried out on the project that was offered by Plantui Ltd. The company was designing an intelligent system to provide IoT support for their main product. The second and third part focuses on hardware development, software development and product management in product development.

The thesis gave practical experience on what project work is and what challenges does working with multiple individuals bring to the table. Individual development and the need for research before an actual feature could be developed was something that needed to be considered. The result of the thesis was a prototype that can be referred as an example for software and hardware development when the company decides to develop the commercial product.

Keywords: Internet of Things, IoT, product development, web development, hardware development, Scrum, project management

SISÄLLYS

1 JOHDANTO

Opinnäytetyö toteutettiin koosteopinnäytetyönä, joka otettiin kokeiluun tietotekniikan koulutusohjelmassa vuodesta 2014 alkaen. Koosteopinnäytetyö tarkoittaa normaalin opinnäytetyön jakamista osiin. Ensimmäinen osa tehtiin keväällä 2014 ja yhdistetty toinen ja kolmas aloitettiin keväällä 2015. Koosteopinnäytetyö oli mahdollista tehdä kolmessa tai kahdessa osassa.

Työn ensimmäisen osa oli luonteeltaan tutkiva selvitystyö. Työssä käytiin erilaisia sensoriteknologisia ja sulautettuja järjestelmiä läpi saatavilla olevien lähteiden kautta. Ensimmäisessä osassa perehdytään mikrokontrollereiden ohjelmointiin ja siihen, millaisia viittauksia ohjelmistossa täytyy olla, jos halutaan käyttää sensoreja hyödyksi toiminnollisuuksissa.

Opinnäytetyön yhdistetyn toisen ja kolmannen osan aiheena oli kehittää Plantui Smart Gardenista IoT-prototyyppi. Työ toteutettiin Plantui Oy:lle yritysprojektin aiheesta. Yritys oli suunnittelemassa laitteellensa älykästä järjestelmää tukemaan sen toimintoja. Toinen ja kolmas osa keskittyy laitekehitykseen, sovelluskehitykseen sekä projektinhallintaan Plantui Pron kehitystyössä.

2 OPINNÄYTETYÖN OSAN 1 AIHE, TAVOITE JA TULOKSET

Opinnäytetyön ensimmäisen osan (liite 1) aiheena oli tutustua mikrokontrollereiden rakenteeseen, ohjelmointiin sekä niiden toimintaan komentoyksikkönä. Tämän lisäksi työssä käsiteltiin erilaisia mikrokontrolleripohjia sekä sitä, mistä osista mikrokontrolleri koostuu. Opinnäytetyön ensimmäisessä osassa tutustuttiin mikrokontrollereiden ohjelmoinnin perusteisiin Arduinon avulla sekä sensoriteknologian perusteisiin mallien avulla. Työn tavoitteena oli selvittää mikrokontrollerin toimintaan tarvittavat komponentit sekä niiden toiminta, kun mikrokontrolleria käskytetään toteuttamaan haluttuja tehtäviä.

Sensoreiden avulla voidaan rakentaa hyvin helposti pelkistettyjä järjestelmiä. Sensorien kirjo on todella laaja ja yhtä sensoria voi käyttää moneen eri tarkoitukseen. Mikrokontrollereita on integroituna sekä avoimena käytettävissä prototyypitykseen. Avoimia mikrokontrollereita ovat esimerkiksi Arduinon mikrokontrollerit sekä Iproto Xi -sarjat. Näiden ohjelmistot ja koodit ovat vapaata lähdekoodia ja siksi hyvin kehitysystävällisiä. Opinnäytetyön ensimmäinen osa raapaisi pintaa mikrokontrollien mahdollisuuksista uusien tuotteiden ja innovaatioiden kehittämisessä.

3 OPINNÄYTETYÖN OSAN 2 + 3 AIHE, TAVOITE JA TULOKSET

Opinnäytetyön yhdistetyn toisen ja kolmannen osan (liite 2) aiheena oli kehittää Plantui Smart Gardenista IoT-prototyyppi. Työ toteutettiin Plantui Oy:lle yritysprojektin aiheesta. Yritys oli suunnittelemassa laitteellensa älykästä järjestelmää tukemaan sen toimintoja. Toinen ja kolmas osa keskittyy laitekehitykseen, sovelluskehitykseen sekä projektinhallintaan Plantui Pron kehitystyössä.

Työn tavoitteena oli ideoida ja suunnitella ratkaisuja Plantui Pro -version kehitykseen. Laitteesta haluttiin tutkia, mitä siitä voidaan mitata ja millä tavalla. Samalla piti arvioida loppukäyttäjälle tuotettavaa lisäarvoa korrelaatiossa mitattaviin suureisiin. Laitekehityksen tueksi käynnistettiin sovelluskehitysprojekti, jonka tavoitteena oli rakentaa laitteelle ja sen uusille ominaisuuksille sovellus. Sovelluksella pitää pystyä näyttämään dataa laitteesta sekä ohjaamaan erilaisia toimintoja laitteessa Internetin välityksellä.

Plantui Oy on vuonna 2012 perustettu Design & Food Tech -yritys, joka keskittyy kasvien kasvatusmenetelmään sisätiloissa vesiviljelyperiaatteella. Kasvatus perustuu Janne Loiskeen visioon älykkäästä puutarhasta, jolla voi kasvattaa kasveja ja yrttejä kaupunkiolosuhteissa.

Plantui Pro -tuotekehitysprojekti koostui sovelluskehitys- ja laitekehitysprojektista. Sovelluskehityksessä mukana olivat Khoa Bùi, An Phạm sekä Long Hoang. Laitekehitysprojektissa mukana olivat Antti Jaara, Mika Schroderus sekä opinnäytetyöstä vastaava.

Laitekehityksen tuloksena oli irrallinen järjestelmä, joka oli ohjattavissa Bluetooth-yhteyden välityksellä erillisellä sovelluksella. Järjestelmältä pystyy kysymään lämpötilan, kosteuden, ilman partikkelien lukumäärän sekä kasvien etäisyyden valokennosta. Järjestelmä pystyy hallitsemaan laitteen valojen intensiteettejä ja näin luomaan käyttäjän valinnan mukaisen valon spektrin.

Sovelluskehityksen tuloksena oli käyttöliittymäkuvaukset sekä toimiva käyttöliittymä kovakoodatulla datalla. Tietokantatauluja ei toteutettu ajanpuutteen vuoksi, vaan keskityttiin käyttöliittymän hiomiseen tilaajan toiveesta. Sovelluskehitysprojekti loppui keväällä 2016 ja opinnäytetyöstä vastaava jatkoi projektia syksyllä 2016. Viimeinen julkaistu versio on julkaistu marraskuussa 2016.

4 YHTEENVETO

Osissa tehty työ oli miellyttävä toteuttaa, koska se aloitettiin jo hyvissä ajoin ja ohjausta sai aina tarvittaessa. Jatkossa olisi hyvä, jos opiskelijalle voitaisiin tarjota koosteopinnäytetyötä mahdollisuutena normaalin opinnäytetyön ohella. Hyödyntämällä koosteopinnäytetyön joustoa saadaan todennäköisesti rytmitettyä opiskelijoiden valmistumista paremmaksi.

Opinnäytetyön ensimmäinen osa antoi tekijälleen hyvin paljon uutta tietoa mikrokontrollereiden mekaanisesta suunnittelusta ja sensoriteknologian moniulotteisuudesta. Teoreettisen työn merkitys korostui yrityslähtöisessä projektityössä, josta muodostui sen työmäärän ansiosta opinnäytetyön toinen ja kolmas osa.

Aihe ensimmäiseen osaan muodostui omasta kiinnostuksesta nanoteknologiaan ja robotiikkaan. Nanoteknologia tuntui liian teoriaperäiseltä, ja käytännönläheisempi ratkaisu oli ottaa aiheeksi mikrokontrollereihin perehtyminen. Opinnäytetyön ensimmäisen osan kirjoittaminen oli mielenkiintoista ja siinä oppi ymmärtämään sensoreiden toimintaa mikrokontrollereiden ja järjestelmien osana.

Jälkimmäinen osa on yhdistetty toinen ja kolmas osa opinnäytetyöstä. Työn aihe tuli ensimmäisestä yrityslähtöisestä tuotekehitysprojektista. Yrityslähtöisestä tuotekehitysprojektista huomattiin pian, että työmäärä vastaa enemmän opinnäytetyötä kuin pelkkää yrityslähtöistä tuotekehitysprojektia. Projektin aihe oli laitekehitys ja opinnäytetyöhön lisättiin tämän lisäksi projektinhallinta ketterässä tuotekehitysprojektissa.

Työssä tutustuttiin ketterän kehitysprojektin hallinnan ja tuotekehityksen prototyypityksen vaiheisiin ja hyötyihin teoriatasolla. Toteutusosassa saatiin käytännön kokemusta laitekehityksen prototyypin työvaiheista sekä sovelluskehityksen tietokanta- ja käyttöliittymäsuunnittelusta sekä ohjelmoinnista moderneilla teknologioilla. Laitekehityksen prototyyppi ja sovelluskehityksen demo olivat työn näkyviä tuloksia.

Yhdistetyn toisen ja kolmannen osan työstäminen antoi paljon uusia kokemuksia projektinhallinnasta ja työskentelystä monialaisessa projektissa, jossa pitää ottaa huomioon loppukäyttäjä. Käytännön harjoitteet ja projektin monimuotoisuus pitivät työn jatkuvasti mielekkäänä ja mielenkiintoisena. Isomman kokonaisuuden hallitseminen ja monen eri vaiheen

toteutuksessa mukana oleminen hyödyttää todella paljon tulevaisuudessa. Työn tuoma kokemus ja tieto tulevat olemaan hyödyllisiä tulevissa haasteissa.

Liite 1

OULUN AMMATTIKORKEAKOULU

Kalle Vuorinen

MIKROKONTROLLEREIDEN OHJELMOINTI JA KÄYTTÄMINEN KOMENTOYKSIKKÖNÄ

Liite 1

MIKROKONTROLLEREIDEN OHJELMOINTI JA KÄYTTÄMINEN KOMENTOYKSIKKÖNÄ

Kalle Vuorinen
Opinnäytetyö osa 1
Kevät 2015
Tietotekniikan koulutusohjelma
Hyvinvointiteknologian suuntautumisvaihtoehto
Oulun ammattikorkeakoulu

SISÄLLYS

SANASTO

Accumulator	– Rekisteri
ALU	– Aritmeettis-looginen-yksikkö, suorittaa kaikki matemaattiset laskelmat
Arduino	– Käytettävä esimerkki mikrokontrolleri (Arduino UNO)
Cache	– Välimuisti, nopeuttaa toimintaa. Välimuistissa pidetään sellaista tietoa, johon viittaaminen on todennäköistä.
CPU	– Prosessorin keskusyksikkö
DRAM	– Sanoista dynamic random access memory. Luku- ja kirjoitusmuistin tyyppi, jossa jokainen bitti tallennetaan kondensaattoriin. Kondensaattorin varaus häviää ajan myötä ja vaatii siksi säännöllistämistä virkistämistä.
GPU	– Graafisten laskelmien suorittava yksikkö
I/O-liitäntä	– Input/Output-liitäntäliitos
IC-piiri	– Mikropiiri eli integroitu piiri
IMU	– Sanoista inertal measurement unit. Yksikkö, joka sisältää kiihtyvyysanturin sekä gyroskoopin.
Master	– Isäntä, voi toimia komennettavana sekä käskevänä yksikkönä.
RAM	– Käyttömuisti
ROM	– Lukumuisti
Slave	– Orja, toimii ainoastaan komennettavana yksikkönä. Ei pysty tuottamaan käskyjä itsenäisesti. Toimii viestien välittäjänä.
SOC	– Järjestelmäpiiri

1 JOHDANTO

Mikrokontrollereiden ohjelmointi ja käyttäminen komentoyksikkönä -opinnäytetyössäni perehdytään mikrokontrollereissa oleviin komponentteihin ja siihen, miten käskytys voidaan toteuttaa ohjelmiston avulla.

Opinnäytetyön tavoitteena on selvittää mikrokontrollerin toimintaan tarvittavat komponentit sekä niiden toiminta, kun mikrokontrolleria käskytetään toteuttamaan haluttuja tehtäviä. Opinnäytetyössä perehdytään tämän lisäksi mikrokontrollereiden ohjelmointiin ja siihen, millaisia viittauksia ohjelmistossa täytyy olla, jos halutaan käyttää sensoreja hyödyksi toiminnollisuuksissa.

Sensoriteknologiaa tutkitaan hyvin pintapuoleisesti. Siinä tutkitaan yleisimpien sensorien lukemaa dataa, jotta siihen pystytään viittaamaan ja hyödyntämään mikrokontrollerissa.

2 MIKROKONTROLLERIT

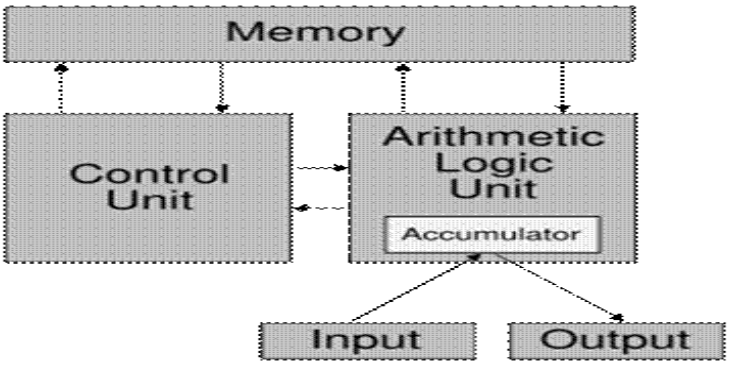
Yleiskäyttöisellä prosessorilla tarkoitetaan mikroprosessoria, jota käytetään yleensä mikrotietokoneissa ja sulautetuissa järjestelmissä. Oleellista tämän tyyppisissä mikrokontrollereissa on se, että niitä käytetään toimituksissa, joissa tarvitaan suurta määrää keskusmuistia. Mikroprosessorit jaetaan kolmeen eri ryhmään: mikroprosessorit, mikro-ohjaimet, signaaliprosessorit.

2.1 Rakenne

Mikrokontrolleri eli IC-piiri on pieni yksittäinen tietokone integroidussa mikropiirissä, joka sisältää prosessorin, muistin ja ohjelmoitavat input/output oheislaitteen eli siirrännän. Pelkästään mikrokontrolleria pystytään kuvaamaan sulautettuna järjestelmänä. Suurin osa käytettävistä mikrokontrollereista on kuitenkin osa jotain eriävää sulautettua järjestelmää.

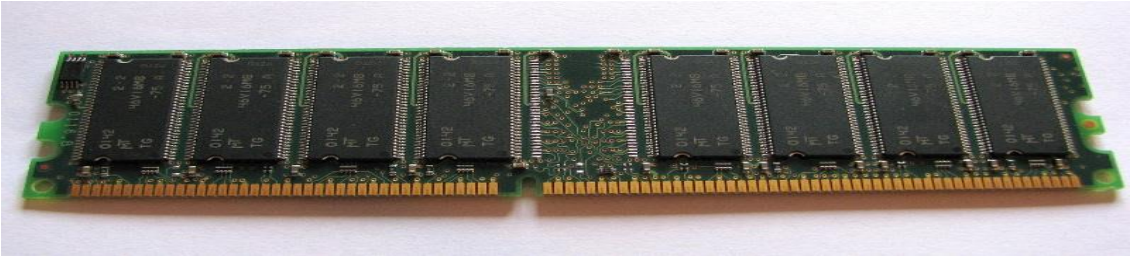
2.1.1 Osat

Suoritin eli prosessori on pääosa mikrokontrollerissa, ja se jakaa käskyt ohjelmoidun ohjelman mukaisesti. Se pystyy tekemään perustoiminnot eli input/ouput-arvon muunnokset pinneissä. Prosessorissa itsessään on aritmetrinen logiikkaosa ALU, joka suorittaa aritmeettisia ja loogisia tehtäviä (kuva 1.). ALU:n laskemat arvot tallentuvat väliaikaiseen rekisteriin eli accumulatoriin, josta ne menevät eteenpäin. Rekistereitä voi olla nykyajan prosessoreissa monia, mutta on myös mahdollista, että niitä on vain yksi. Nykyaikaisia prosessoreita sanotaan mikroprosessoreiksi, koska niissä on yksi integroitu piiri eli IC-piiri. CPU:n sisältävä IC-piiri voi myös sisältää muistin, oheiskäyttöliittymän sekä muita tietokoneen osia. Tällaisia rakenteita kutsutaan jo mikrokontrollereiksi tai sirujärjestelmiksi SoC. Prosessori eli suoritin on osa, joka hakee muistista käskyt ja prosessoi sen mukaiset tehtävät. (1, s. 1,4,7–8)



KUVA 1. CPU:n ja ALU:n toiminta (2)

Muisti on paikka, johon suoritettava ohjelma on tallennettu. Yleisesti ottaen muisti on RAM-muistia (kuva 2), eli se tyhjenee, kun mikroprosessoriin ei kulje enää virta. Keskusmuisti säilyttää kaiken tiedon, jota mikrokontrolleri toiminnassaan tarvitsee. Käytössä on kahdenlaista muistia ROM- ja RAM-muistia. ROM:ssa on tallessa ohjelmat, joiden pitää säilyä virran katkaisun yli eli esimerkiksi käynnistysohjelma. RAM:ssa on tallessa sellaiset tiedot ja ohjelmat, jotka saavat tuhoutua virran katketessa. Kaikki muuttuva ja prosessoitava tieto on RAM:ssa, koska CPU pystyy kirjoittamaan RAM-muistissa olevaa tietoa uusiksi. Prosessorin lukutapaan ei vaikuta se, että lukeeko se RAM-muistista vai ROM-muistista tietoa. (1, s. 7–8.)



KUVA 2. Geneerinen DDR-RAM -muistipalikka (3)

Tulo- ja lähtöliitännät eli input/output-liitännät ovat tarpeelliset mikrokontrollerille, jotta se saa liitäntärajapinnan ympäristöönsä. Liitäntöihin voidaan liittää erilaisia moduuleita ja sensoreita, joilla voidaan lisätä toiminnollisuuksia mikrokontrollerille. Tämä on yleisesti vähiten standardoitu osa, joka tarkoittaa sitä, että ne vaihtelevat huomattavasti varsinkin sulautetuissa järjestelmissä. Varsinaisesti mikrokontrollerit toimivat digitaalisten ja analogisten pinnien avulla, joista osa on input- ja osa output-tuloja. (1, s. 7–8.)

2.1.2 Kytkennät

Mikrokontrolleri vaatii vähintään seuraavan laisen kytkennän toimiakseen: käyttöjännite, kellosignaali, reset-signaali ja I/O-liitäntä ja vähintäänkin yksi kytkin sekä merkkivalo. (1, s. 15.)

2.1.3 Toiminta

Mikroprosessorin toimintaa on helpoin tarkastella siitä lähtevien tietoväylien avulla. Näitä väyliä ovat osoiteväylä, tietoväylä sekä yhdistetty tieto- ja ohjausväylä.

Osoiteväylä on prosessorilta lähtevä yksisuuntainen väylä, jonka avulla prosessori määrää kohdekomponentin ja siellä olevan muistipaikan, jonka kanssa se asioi. Osoiteväylän leveydellä ei ole suoranaista vaikutusta tehokkuuteen vaan se vaikuttaa siihen, kuinka suuri on prosessorin muistiavaruus. Osoiteväylän liitäntöjä merkitään A-kirjaimella eli esimerkiksi A0, A1, A2 …, An, jossa n+1 on osoiteväylän leveys. (4, s. 13–18.)

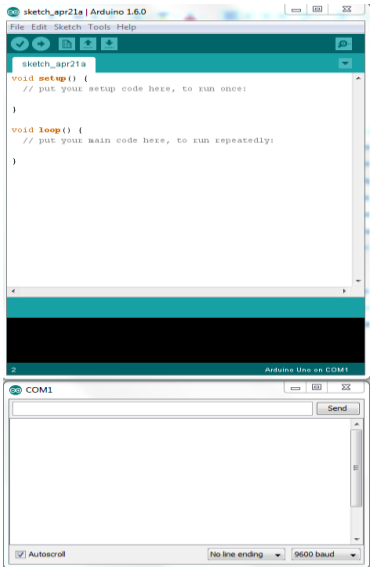
Tietoväylä siirtää ohjelmakoodia ohjelmamuistista prosessorille. Se myös välittää kaiken käsiteltävän tiedon prosessorin, muistin sekä liitäntäpiirien välillä. Tietoväylä toimii kaksisuuntaisesti eli esimerkiksi muistista prosessorille ja prosessorista muistille. Tietoväylän leveys määrittää sen, kuinka suuria määrinä prosessori pystyy samanaikaisesti käsittelemään tietoa yhdellä osoituksella. Tietoväylän leveydellä on erittäin suuri merkitys tehokkuudelle. Tietoväylän liitäntöjä merkitään yleensä D-kirjaimella D0, D1, D2 …, Dn. Tietoväylä kytketään kaikille niille piireille, joihin prosessori kirjoittaa ja josta se lukee tietoa. (4, s. 14.)

Yhdistetty tieto- ja osoiteväylä toimii kahdella eri signaalilla eli tieto- ja osoitesignaalilla. Nämä kaksi signaalia vuorottelevat samoissa liitännöissä. Väylien erottamiseksi toisistaan tarvitaan ulkoinen väyläerotinpiiri. Prosessori syöttää ensin yhdistetylle väylälle osoitteen, joka lukitaan ulkoisiin lukkopiirin lähtöihin. Osoite jää lukkopiirin lähtöihin, kunnes seuraava osoite syötetään lukkopiirille. Kun osoite on lukittu lukkopiirin lähtöihin, toimii yhdistetty väylä normaalisti tietoväylänä. Tätä väylätyyppiä käytetään erityisesti esimerkiksi Intelin mikroprosessoreissa. (4, s. 14.)

2.2 Ohjelmointi

Mikrokontrollerit ohjelmoidaan yleensä C-kielellä, jonka mikrokontrolleri kääntää Assembly-kielelle. Assembly-kielestä käytetään myös termiä symbolinen konekieli. Yleensä käytetään C-kieltä, jonka assembler-kääntäjä kääntää Assembly-kielelle, jonka mukaan mikrokontrolleri toimii. Nykyään käytetään hyvin vähän suoraan Assembly-kielellä kirjoitettavia laitteistoja, sillä se on erittäin hidasta. Hyvänä puolena Assembly-kielessä on, että se antaa täyden määräysvallan ohjelmoijalle. Edistyneimmissä kielissä ongelmana voi olla se, että varsinaisia toimintoja ei ole määritetty ja joudutaan rakentamaan hitaamman kiertoratkaisun, jotta saadaan haluttu ominaisuus tai toiminta toimimaan.

C-kieltä käytetään suurelta osin mikrokontrollereiden, koska se on hyvin lähellä konekieltä ja esimerkiksi Arduinolla on oma sovellus, joka kirjoittaa C-kielen suoraan mikrokontrollerille ja se toimii tämän perusteella. Kuvassa 3 on Arduinon oma sovellus sekä sen komentorivi.



Kuva 3. Arduino ja komentoikkuna

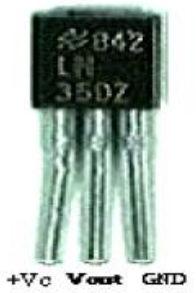
3 SENSORITEKNOLOGIA MIKROKONTROLLEREISSA

3.1 Sensorit

Sensoreja on kuutta eri päätyyppiä, joihin luetellaan muut sensorit. Sensoreita on monia ja varsinkin sensorien alatyyppejä on monia. Sensoreiden kuusi päätyyppiä ovat lämpötila-, paine-, paikka-, nopeus-, voima- ja kiihtyvyysanturi. Jokaista sensorityyppiä voidaan käyttää osana mikrokontrolleria, jotta saadaan haluttu tieto tai toiminto. Seuraavaksi esitellään kukin pääluokka esimerkkeineen.

3.1.1 Lämpötila-anturi

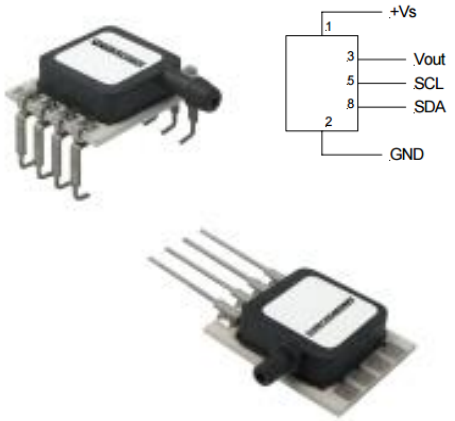
Lämpötila-antureita on vastuslämpötila-anturi sekä termopareja. Yksi käytetyimmistä lämpötila-antureista on National Semiconductor -perheen LM35 IC, joka on vastuslämpötila-anturi. Lämpötila-antureita on monenlaisia. Lämpötila-arvot, missä anturia voidaan käyttää, ja se, käyttääkö sensori Celsius-asteikkoa vai Kelvin-asteikkoa, ovat hyvin anturikohtaisia. Kuvassa 4 esitetty anturi antaa lineaarista lämpötila-arvoa, ja sen toiminta-alue on -55 celsiuasasteesta +150 celsiusasteeseen. (5.)



KUVA 4. LM35 lämpötila-anturi (5)

3.1.2 Paineanturi

Paineantureita on monenlaisia, mutta ne luokitellaan muutamaan pääluokkaan. Suurin luokka on painetta keräävät ja muita tyyppejä ovat resonoiva, lämpötila ja ionisoiva. Yksinkertaisimmin käytettävistä paineantureista on painetta keräävä ilmanpaine-sensori, jolla voidaan mitata esimerkiksi keuhkojen tilavuutta. Kuvassa 5 on esimerkki anturista, joka mittaa painetta ilmanpaineen mukaan.

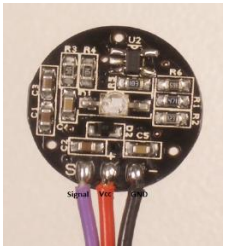


KUVA 5. *HCA-BARO paineanturi* (6)

3.1.3 Paikka-anturi

Paikka-antureista käytetyimpiä ovat nykyään optiset pulssisensorit, joiden avulla pystytään mittaamaan pulssia valon avulla. Kuvassa 6 on esitelty yksi pulssisensori, jonka voi helposti liittää Arduino-mikrokontrolleriin ja mitata esimerkiksi omaa pulssia. Kuvassa 7 on kyseisen sensorin kytkentä esitettynä. S-kirjaimella merkitty napa tulee kytkeä signaalin output-kohtaan, ”+”-merkillä merkitty tulee kytkeä jännitteeseen ja ”-”-merkillä merkitty tulee kytkeä maahan.

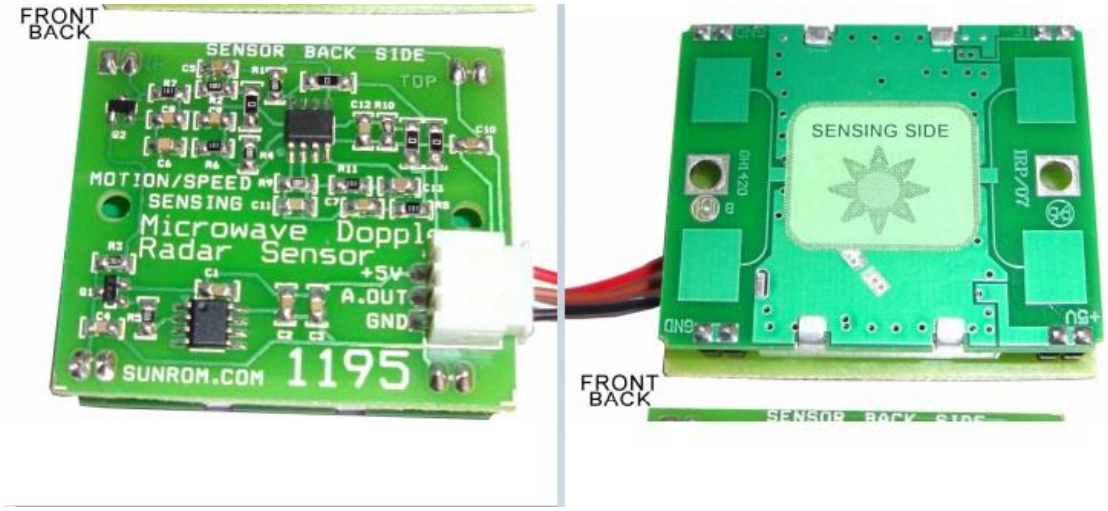
KUVA 6. Pulse Sensor AMPED (7)



KUVA 7. Pulse Sensor AMPED -kytkennät (7)

3.1.4 Nopeusanturi

Nopeusantureista käytetyimpiä ovat tutka-anturit. Tutka-antureita käytetään muun muassa liikenteessä nopeuden valvonnassa. Kuvassa 8 on esimerkki tutka-anturista. Kuvan 8 anturi on Dopplerin tutka-anturi, joka pystyy havaitsemaan liikettä ja liikkeen nopeutta. Tutka-antureissa toimintaperiaate perustuu elektromagneettisen signaalin lähettämiseen ja sen vastaanottamiseen eli niin sanottuun heijastumiseen. Kuvan 8 anturilla voidaan mitata nopeutta ja liikettä 20 metrin etäisyydeltä ja se pystytään kytkemään suoraan analog out –pinniin mikrokontrollerissa. (8.)

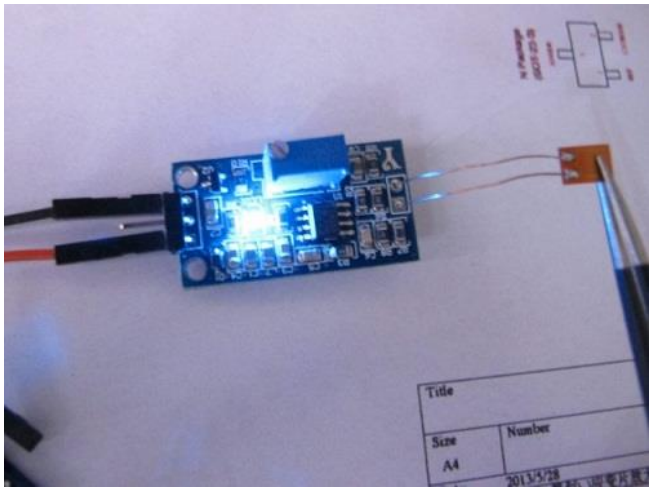


KUVA 8. Doppler *Radar -sensori* (8)

3.1.5 Voima-anturi

Voima-anturia eli esimerkiksi venymäliuska-anturi perustuu materiaalin venymiseen. Materiaalin resistanssi muuttuu venymisen johdosta. Venymäliuska-anturi on yleisin hydrauliikassa ja pneumatiikassa käytetty anturi. Liuskaan on kiinnitetty metallilangasta tai foliosta tehty vastus.

Kuvassa 9 on venymäliuska-anturi osana Arduinon moduulia. Kyseisessä venymäliuskassa on käytetty metallilangasta tehtyä vastusta.



KUVA 9. *Arduinon moduuli, jossa on venymäliuska-anturi* (9)

3.1.6 Kiihtyvyysanturi

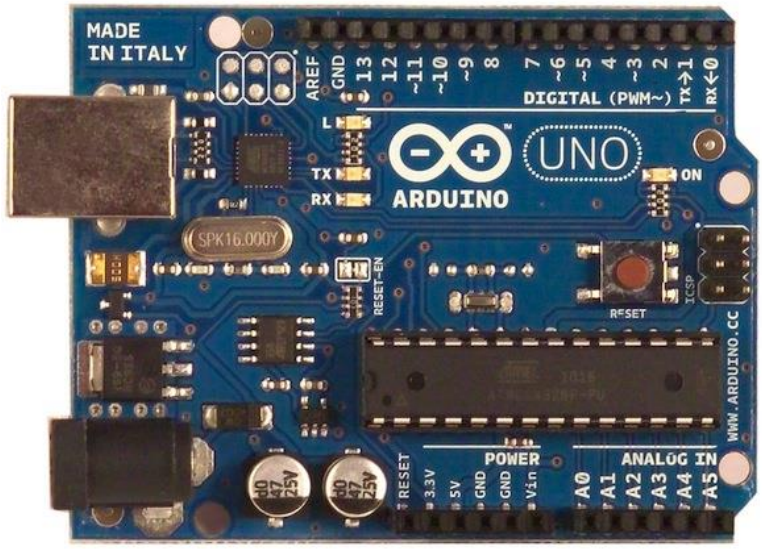
Kiihtyvyysantureita ja gyroskooppeja käytetään erittäin monessa sovelluksessa. Kiihtyvyysanturi mittaa liikettä kolmessa lineaarisessa suunnassa (x-, y-, z-vektori). On myös olemassa kiihtyvyysantureita, jotka mittaavat kiihtyvyyttä vain yhdessä suunnassa. Kiihtyvyysantureita, jotka mittaavat vain yhdensuuntaista liikettä, käytetään esimerkiksi junan nopeuden mittaamisessa. Puhelimissa on nykyään antureita, jotka mahdollistavat näytön kääntymisen samalla kun käännät puhelinta. Kuvassa 10 on käytetty IMU-sensoria, joka on liitetty jo mikrokontrolleriin. (10.)

KUVA 10. SparkFun Triple Axis Accelerometer Breakout - ADXL335 *ja kytkentä mikrokontrolleriin* (10)

3.2 Sensorien käyttö ohjelmassa

Sensoreja käytetään mikrokontrollereiden lisänä, jotta saadaan mitattua jotakin halututtua suuretta tai tapahtumaa. Seuraavissa kappaleissa käydään läpi, miten sensoreihin viitataan Arduinon ohjelmassa.

Sensoreille pitää määrittää ohjelman alussa jokin pinni, jonka kautta ne syöttävät arvoa mikrokontrollerille. Jos pinniä ei määritetä, käyttää Arduino oletettavaa pinniä A0 (Kuva 11).



KUVA 11. *Arduino UNO rakenne* (11)

Arduinolla pystytään mittaamaan suoraan esimerkiksi jännitettä sensorin signaalista, jos se otetaan huomioon ulossyötettävässä arvossa. Arduino mittaa A0 pinnin kautta analogista jännitettä välillä 0–5 V, mutta analogRead-funktio muuttaa sen A/D-muuntimella digitaaliseksi arvoksi 0 – 1023. Saatu arvo saadaan muutettua jännitteeksi kertomalla se jännitteen suhdanteella, kuten kuvassa 12 on tehty. Kuvan 13 ohjelma tulostaa serialille eli komentoikkunaan jännitteen arvon. Sensorien antamaan arvoon viitataan hyödyntämällä pinniä, johon sensori on kytketty.

```
void loop() {
  // read the input on analog pin 0:
  int sensorValue = analogRead(A0);
  // Convert the analog reading (which goes from 0 - 1023) to a voltage (0 - 5V):
  float voltage = sensorValue * (5.0 / 1023.0);
  // print out the value you read:
  Serial.println(voltage);
}
```

KUVA 12. Arduinon jännitelasku ja sensorValue

Sensorit antavat mikrokontrollereille sensorValuen eli sensorin arvon. Sensorin arvo vaihtelee todella paljon ja on myös riippuvainen mikrokontrollerista. Esimerkiksi Arduino antaa analogiseksi arvoksi maksimissaan 1023, jota voidaan verrata esimerkiksi jännitteeseen ja laskea siitä arvoa. Digitaaliset pinnit voivat tuottaa 255 arvoa, joka vastaa 5 volttia.

Sensoriarvoa muuttaessa 0–1023 välillä, se muuttuu jännitteestä mitä pinni lukee. Pitää tehdä toinen muuttuja, joka laskee tietyllä lausekkeella arvojen 0.0-5.0. Eli saadaan lauseke:

float voltage = sensorValue * (5.0/1023.0);

Tämä arvo tulostetaan vielä serial monitoriin käyttämällä Serial.println-komentoa.

Serial.println(voltage);

4 MIKROKONTROLLERI OSANA SULAUTETTUA JÄRJESTELMÄÄ

Mikrokontrolleja on yksi sulautetun järjestelmän tärkein osa. Se määrittelee sulautetun järjestelmän toimimisen. Hyvin yksinkertaiset arkiset asiat ovat sulautettuja järjestelmiä. Esimerkiksi pysäköintilippuautomaatti on sulautettu järjestelmä. Mikrokontrollerin tehtävänä on nimensä mukaisesti kontrolloida laitteistoa, joka siihen on liitetty. Mikrokontrolleri on siis komentoyksikkö, joka pystyy toimimaan omatoimisesti.

Yleensä sulautetut järjestelmät ovat olleet suljettuja ja täten kolmas osapuoli ei ole pystynyt muuttamaan tai tarjoamaan omia ohjelmia niitä varten. Nykyään on myös käytössä avoimia järjestelmiä, joihin voi muuttaa esimerkiksi avointa lähdekoodia käyttäen ohjelmistoja.

Avoimia järjestelmiä ovat nykyään puhelimet, joihin pystyy asentamaan halutun Android-käyttöjärjestelmän. Myös erilaiset mikrokontrollerit voidaan jo itsessään luokitella sulautetuiksi järjestelmiksi, koska niillä on niin paljon toimintoja ilman lisälaitteita. Tällaisia mikrokontrollereita, joita voidaan ohjelmoida ja jotka voidaan luokitella jo itsessään sulautetuksi järjestelmäksi, ovat esimerkiksi Arduino-tuotesarjan mikrokontrollerit, Aistimen Iproto Xi -sarjan mikrokontrollerit sekä monet muut vastaavanlaiset tuotteet. Tämänlaiset laitteistot ovat hyvin yleisessä käytössä, kun rakennetaan prototyyppejä, koska niitä on helppo ja halpa soveltaa.

Suljettuja järjestelmiin voi laskea puhelimien käyttöjärjestelmät, joiden koodi ei ole ”open source” -pohjalla toimivaa eli Windows ja Apple -käyttöjärjestelmillä toimivat puhelimet. Toinen tunnettu laiteluokka suljetusta järjestelmästä on pelikonsolit.

5 YHTEENVETO

Tässä opinnäytetyössä on perehdytty mikrokontrollereiden rakenteeseen ja toimintaan ja erilaisten sensorityyppien toimintaan esimerkkien avulla. Työssä käsiteltiin antureiden käyttöä Arduinon omassa ohjelmassa. Neljännessä pääluvussa käsiteltiin mikrokontrollereiden toimintaa osana sulautettua järjestelmää. Sulautetun järjestelmän käsitteet avoin ja suljettu järjestelmä käsitellään muutaman esimerkin avulla.

Sensoreiden avulla voidaan rakentaa hyvin helposti pelkistettyjä järjestelmiä. Sensorien kirjo on todella laaja ja yhtä sensoria voi käyttää moneen eri tarkoitukseen. Mikrokontrollereita on integroituna sekä avoimena käytettävissä prototyypitykseen. Avoimia mikrokontrollereita ovat esimerkiksi Arduinon mikrokontrollerit sekä Iproto Xi -sarjat. Näiden ohjelmistot ja koodistot ovat ”open source” ja siksi hyvin kehitysystävällisiä. Opinnäytetyön ensimmäinen osa raapaisi pintaa mikrokontrollien mahdollisuuksista uusien tuotteiden ja innovaatioiden kehittämisessä.

LÄHTEET

1. Rantala, Pekka. 2001. Tietokonetekniikka osa 2 Mikrotietokonetekniikka. Kotka: TietoKotka Oy.
2. Prasetya Ryan. ARITHMETIC LOGIC UNIT (ALU) http://kopongkopong.blogspot.fi/2011/09/arithmetic-logic-unit-alu.html Hakupäivä 24.04.2015.
3. DDR RAM kuva http://upload.wikimedia.org/wikipedia/commons/thumb/9/9b/Generic_DDR_Memory_%28Xytram%29.jpg/1920px-Generic_DDR_Memory_%28Xytram%29.jpg Hakupäivä 24.04.2015.
4. Koskinen, Jari. 2004. Mikrotietokonetekniikka Sulautetut järjestelmät. Keuruu: Otavan Kirjapaino Oy.
5. Nilesh R. Patel – Swarup S. Mathurkar & Rahul B. Lanjewar – Ashwin A. Bhandekar. 2013. Microcontroller based drip irrigation system using smart sensor. Saatavissa: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6726064 Hakupäivä 10.04.2015.
6. Barometricsensor Kuva http://www.sensortechnics.com/en/products/pressure-sensors-and-transmitters/barometric-pressure-sensors/hca-baro/ Hakupäivä 24.04.2015.
7. Pulssisensori http://www.makershed.com/products/pulse-sensor-amped-for-arduino Hakupäivä 24.04.2015
8. Tutka http://www.sunrom.com/media/files/p/212/1195-datasheet.pdf Hakupäivä 24.04.2015.
9. Venymäliuska http://www.elecrow.com/images/Products/strain-Pro1.jpg Hakupäivä 24.04.2015.
10. Kiihtyvyys http://www.robotshop.com/blog/en/arduino-5-minute-tutorials-lesson-7-accelerometers-gyros-imus-3634 Hakupäivä 24.04.2015.
11. Arduino. http://cdn.makeuseof.com/wp-content/uploads/2011/11/ArduinoUnoFront1.jpg?6055c3 Hakupäivä 24.04.2015.

Liite 2

OULUN AMMATTIKORKEAKOULU

Kalle Vuorinen

PLANTUI PRON TUOTEKEHITYS

Liite 2

PLANTUI PRON TUOTEKEHITYS

Kalle Vuorinen
Opinnäytetyö osa 2 + 3
Kevät 2017
Tietotekniikan koulutusohjelma
Hyvinvointiteknologian suuntautumisvaihtoehto
Oulun ammattikorkeakoulu

SISÄLLYS

SANASTO

Client	– Sovellus, joka käyttää palvelimen tarjoamia palveluja.
Frontend	– Yleiskäsite teknologiasta, joka mahdollistaa käyttöliittymän ulkoasun.
I/O	– Tulee sanoista Input/Output. Yleinen määritelmä sisään- ja ulosannettavasta kytkennästä.
IoT	– Asioiden Internet tulee englanninkielisestä ilmaisusta ”Internet of Things”. Tarkoittaa Internetin laajentumista laitteisiin ja koneisiin, joita voidaan ohjata ja mitata verkon yli.
Iteraatio	– Sprint iteraatio on yksi 1-4 viikon kehitysjakso, jonka tuloksena on julkaisukelpoinen tuote.
MVP	– Tulee englanninkielisistä termeistä “Minimum viable product”, joka tarkoittaa elinkelpoista vähimmäistuotetta.
Ngrx store	– Tilakone, joka on suunniteltu helpottamaan tilakäsittelyä sovelluksessa.
REST	– Tulee englanninkielisistä sanoista “Representational State Transfer”. REST tarkoittaa asynkronista mallia palvelinrajapinnoilla.
Spektri	– Valon spektri on jakauma valonsäteiden aallonpituuksista.

1 JOHDANTO

Tämän opinnäytetyön toimeksiantajana on turkulainen kasviviljelyn tukipalveluja tuottava Plantui Oy. Plantui Oy:n tuote Plantui Smart Garden on vesiviljelyyn perustuva innovatiivinen minikasvihuone, jolla voidaan kasvattaa myrkyttömiä ja ravinnerikkaampia syötäviä kasveja kuin lähiruokakaupan tuoreosasto pystyy tarjoamaan.

Opinnäytetyön tavoitteena on ideoida ja suunnitella ratkaisuja Plantui Pro -version kehitykseen. Plantui Pro tulee olemaan ulkonäöllisesti ja päätoimiltaan samanlainen kuin Plantui 6. Laitteeseen tullaan toteuttamaan IoT-tuki, joka mahdollistaa lisäominaisuuksien käyttöönoton. Laitteen tueksi kehitetään käyttöliittymäksi web-sovellus, jolla pystytään lukemaan kasvatuksen eri vaiheita sensoridatan avulla. Tavoitteena on myös tuottaa yritykselle lisää positiivista näkyvyyttä opiskelijayhteistyöllä sekä tuottaa lisäarvoa asiakkaille.

Lähtötietomuistiota tehdessä työn tavoitteena oli perehtyä siihen, mitä laitteesta voidaan mitata, millä tavalla mittaukset voidaan suorittaa ja mikä on loppukäyttäjälle tuleva lisäarvo tiedosta. Kick off -tapaamisessa työn tavoitteiksi muotoutui kasvien korkeuden, kosteuden, ilman partikkeleiden ja lämpötilan mittaaminen. Tarkoitus oli rakentaa älypuutarhan rinnalle mittausjärjestelmä, joka toimisi esimerkiksi Arduinolla, Raspberry Pillä tai vastaavalla mikrokontrollerilla.

Projekti koostui laitekehitys- ja web-sovellusprojektista, jotka liittyivät suunnittelun kautta vahvasti toisiinsa. Laitekehitysprojektissa mukana olivat Antti Jaara, Mika Schroderus Mika sekä opinnäytetyöstä vastaava. Web-sovelluskehityksessä mukana olivat Khoa Bùi, An Phạm sekä Long Hoang.

Sensoriteknologisia ratkaisuja suunnitellaan POC-ajattelumallia hyödyntäen. Sensoreita valittaessa on tarkoitus tehdä myös tutkimustyötä mahdollisista sensoreista tuotantokäyttöön. Tuotantokäyttöön tulevista sensoreista pitää ottaa huomioon kustannukset sekä sensorien resistiivisyys korroosiolle. Opinnäytetyössä perehdytään prototyypin toteuttamiseen laite- ja sovelluskehityksen sekä projektin hallinnan kautta.

2 TILAAJAN JA TUOTTEEN ESITTELY

2.1 Plantui Oy

Vuonna 2012 perustettu Plantui Oy on Design & Food Tech -yritys, joka keskittyy kasvien kasvatusmenetelmään sisätiloissa vesiviljelyperiaatteella. Kasvatus perustuu Janne Loiskeen visioon älykkäästä puutarhasta, jolla voi kasvattaa kasveja ja yrttejä kaupunkiolosuhteissa. Ajatus älykkäästä puutarhasta esiteltiin vuonna 2010 Tokyo Design Week -tapahtumassa Japanissa, jossa se sai positiivisen vastaanoton. Palautteen innoittamana Plantui Oy perustettiin useiden sijoittajien ja yhteistyökumppanien avulla, ja päästiin kehittämään ensimmäistä Plantui-tuotetta, joka julkaistiin kesäkuussa 2014 Tanskassa. (1.)

Tällä hetkellä yrityksellä on 10–20 työntekijää ympäri maailmaa. Pääkonttori sijaitsee Logomon kulttuurikeskuksessa Turussa. Tuotteen kehitys tapahtuu yhteistyökumppanin Mativation Oy:n tiloissa Salossa. Tuotanto tapahtuu Puolassa, mutta se on siirtymässä Singaporen tasavaltaan. (2.)

2.2 Plantui Smart Garden

Plantui Smart Garden mahdollistaa tuoreiden yrttien kasvattamisen ympäri vuoden. Laitteeseen laitetaan Plantui-kasvikapselit, vettä ja lannoitetta, jonka jälkeen laite huolehtii itsenäisesti kasvien kasvatuksesta. Valot ja kastelu toimivat täysin automaattisesti. Plantuin älypuutarhassa voi kasvattaa kuutta kasvikapselia kerrallaan. Laitteella voi kasvattaa yrttejä, lehtivihanneksia tai syötäviä kukkia siemenestä satoon 5–8 viikossa. (3.) (Kuva 1.)

KUVA 1. Kasvuvaiheet (4)

2.1 Teknologia

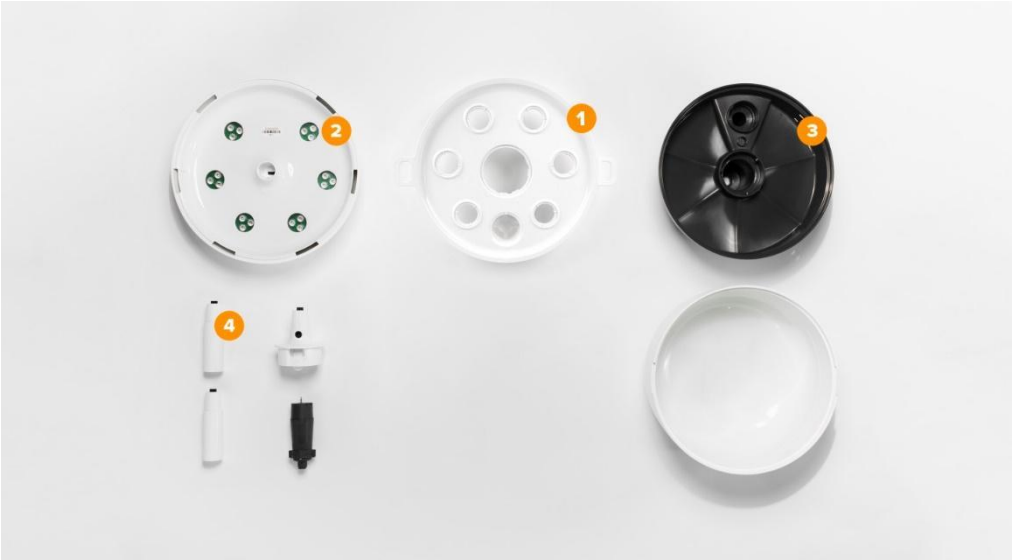
Kasvatusteknologia perustuu Plantui Oy:n kehittämään kasvatusmenetelmään. Patentoitu kasvatusmenetelmä sisältää kasvien yhteyttämiselle parhaan mahdollisen valon spektrin sekä sopivan kastelurytmin ja ravinnekoostumuksen. Plantuin valoista kasvit saavat juuri oikean määrän valoa, mikä mahdollistaa mahdollisimman tehokkaan kasvun. Plantui ohjaa valoja ja kastelua automaattisesti kasvin kasvuvaiheen mukaan. Plantui Smart Gardenin tarkoitus on tuoda ruoka puhtaana, tuoreena, terveellisenä ja lähiruokana ruokapöytään. Plantui ei käytä multaa tai muita haitallisia aineita kasvatuksessa, joten ei tarvitse miettiä, mitä aineita kasvatukseen on käytetty tai mistä ruoka on peräisin. (3; 4.)

2.2 Tuotteen muotoilu ja rakenne

Yksinkertainen ja käytännöllinen muotoilu on keskeinen osa tuotetta. Suunnittelija Janne Loiske sai kansainvälisen Red Dot -palkinnon Plantuin muotoilusta. Red Dot -palkinto on kansainvälisesti tunnustettu laatumerkintä. Tähän mennessä on suunniteltu kolme Plantui Smart Garden -alustaa: Plantui 6, pienempi Plantui 3 ja Muumi Garden lapsille. (5.) (Kuva 2.) Plantui-alustojen perusrakenne koostuu kasvialustasta, valoyksiköstä, vesiyksiköstä ja korkeuspaloista. (Kuva 3.)

KUVA 2. Plantui-kasvualustat (6)

Liite 2



KUVA 3. Tuotteen osat: 1.Kasvialusta, 2.valoyksikkö, 3.vesiyksikkö, 4.korkeuspalat (7)

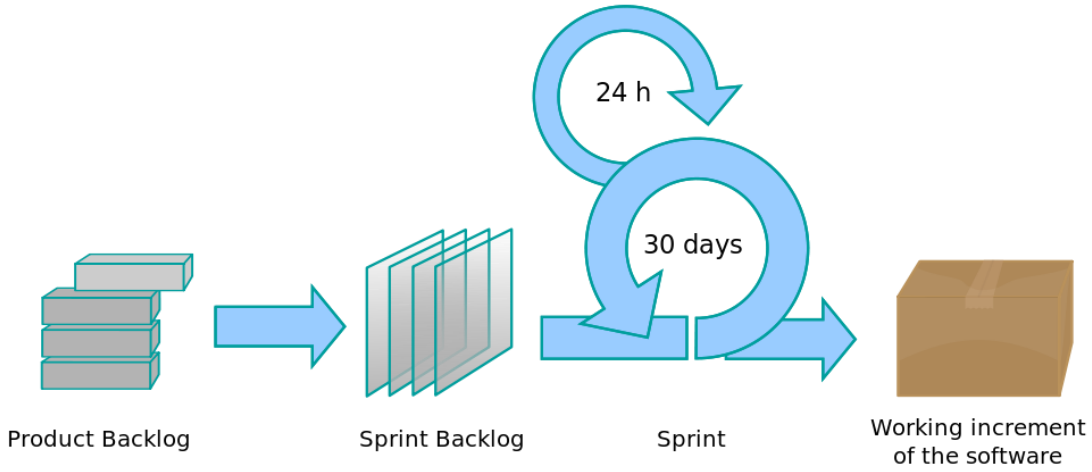
3 PROJEKTINHALLINTA

Projektinhallinta tarkoittaa resurssien organisointia ja hallintaa. Projektinhallinnassa tavoitteena on päättää projektin suunnitteluvaiheessa siihen tulevat asiat ja projektiin käytettävä aika. Projektinhallintamalleja on monia, joista käytetyimmät ovat vesiputousmalli ja ketterän kehityksen tuomat mallit. Vesiputousmallissa pystytään projekti suunnittelemaan niin tarkasti, ettei siihen tule toteutuksen aikana juurikaan muutoksia ominaisuuksiin tai aikatauluun. Monet yritykset ovat havainneet, että vesiputousmalli on toimiva ratkaisu silloin, kun suunnitelmat eivät tuotetta tai palvelua kohtaan muutu kehityksen aikana. (8, s. 3–4.)

Scrum on yleisesti käytetty ketterä projektinhallintamalli. Scrumin ideologia pohjautuu 1–4 viikon kehitysjaksoihin eli sprintteihin. Sprintin ensimmäinen päivä on suunnittelupäivä, jolloin määritetään sprinttiin otettava työmäärä viime sprinttien kehitysvauhdin perusteella. (8, s. 6–8.) (Kuva 4.)

Kehitysjaksoon tuleva työ määritellään etukäteen kehitysjonoon eli tuotteen backlogiin, johon tehtävät käyttäjätarinat määritellään priorisoituun järjestykseen. Scrumin kehitysjakson tarkoituksena on suojata kehitystiimi ulkopuolisilta häiriöiltä, sillä työ on suunniteltu eteenpäin koko kehitysjaksoksi. Scrummaster vastaa ensisijaisesti kehitystiimin koskemattomuudesta ja siitä, että tehtävä työ tulee aina tuotteen omistajan hallinnoiman kehitysjonon kautta. (8, s. 6–8.)

Tuotteen kehitysjono toimii listana, jossa on kaikki haluttavat toiminnallisuudet. Ominaisuuksien listaamisen jälkeen ne priorisoidaan. Priorisoidusta kehitysjonosta eli backlogista, johon on suoritettu kehitysjonon työstö, valitaan tehtävät yhdelle sprintille tehtävälistaan. Kehitysjonon työstö on yksi Scrumiin liittyvistä työvaiheista, jossa kehitystiimi käy yhdessä läpi kehitysjono ja pilkkoo siellä olevia ominaisuuksia pienemmiksi ja toteutettaviksi käyttäjätarinoiksi. (8, s. 6–8.) (Kuva 4.)



KUVA 4. Scrumin prosessi (9)

Scrum-projektihallintaan kuuluu sille ominaisia palavereja ja suunnittelukokouksia. Yksi näistä on kehitysjonon työstö, johon osallistuu niin tuotteen omistaja kuin koko kehitystiimi. Tuotteen kehitysjonoon on sijoitettu priorisoidussa järjestyksessä käyttäjätarinoita, joita kehitystiimi valitsee seuraavaan iteraatioon sprintin suunnittelupalaverissa, joka on iteraation ensimmäisenä päivänä. Sprintin viimeisenä päivänä on sprinttikatselmus, jossa kehitystiimi pitää demon suorituksestaan asianomaisille. (8, s. 7–9.)

Katselmointi ja palautekeskustelu kuluneesta sprintistä pidetään tuotteen omistajan kanssa demon jälkeen. Katselmoinnin jälkeen kehitystiimi pitää keskenään retrospektiivin, jossa käydään onnistuneet ja epäonnistuneet asiat läpi sprintillä. Retrospektiivin tarkoitus on ylläpitää tiimin jatkuvaa kehitystä. Siinä sovitaan yleisiä pelisääntöjä, mahdollisia koulutuksia ja parannusehdotuksia toimintamalleihin. Retrospektiivin tarkoitus on kehittää tiimistä tehokkaampi ja itseohjautuvaisempi. (8, s. 7–12.)

4 LAITEKEHITYS

Laitekehitysprojektin tavoitteisiin kuului tunnistaa mahdollisia mitattavia suureita ja tapaa mitata niitä. Tarkoituksena oli huomioida sensorien kestävyys tuotannossa, mitattavien suureiden hyöty kuluttajalle sekä automatisointi. Opinnäytetyössä perehdyttiin kuuteen sensoripäätyyppiin, joiden pohjalta tutkittiin potentiaalisia sensorimalleja laiteprototyyppiin. Seuraavissa alaluvuissa perehdytään sensorien päätyyppeihin sekä niiden käyttöön mittausjärjestelmässä.

4.1 Sensorit

Sensoreita ja varsinkin sensorien alatyyppejä on monia. Sensoreiden kuusi päätyyppiä ovat lämpötila-, paine-, paikka-, nopeus-, voima- ja kiihtyvyyssensori. Jokaista sensorityyppiä voidaan käyttää osana mikrokontrolleria, jotta saadaan haluttu tieto tai toiminto. Seuraavaksi esitellään sensorien pääluokat esimerkkien avulla.

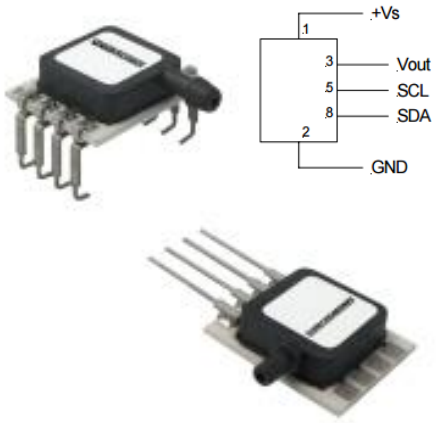
4.1.1 Kaasusensori

Lämpötilasensoreita on vastuslämpötilasensori sekä termopareja. Yksi käytetyimmistä lämpötilasensoreista on National Semiconductor -perheen LM35 IC (kuva 5), joka on vastuslämpötilasensori. Lämpötilasensoreja on monenlaisia. Lämpötila-arvot, missä sensoria voidaan käyttää, ja se, käyttääkö sensori Celsius-asteikkoa vai Kelvin-asteikkoa, ovat hyvin sensorikohtaisia. Kuvassa 5 esitetty sensori antaa lineaarista lämpötila-arvoa, ja sen toiminta-alue on –55 celsiusasteesta +150 celsiusasteeseen. (11.)

KUVA 5. LM35-lämpötilasensori (11)

4.1.2 Painesensori

Painesensoreita on monenlaisia, mutta ne luokitellaan muutamaan pääluokkaan. Suurin luokka on painetta keräävät, ja muita tyyppejä ovat resonoiva, lämpötilaan perustuva ja ionisoiva sensori. Yksinkertaisin käytettävistä painesensoreista on painetta keräävä ilmanpainesensori, jolla voidaan mitata esimerkiksi keuhkojen tilavuutta. Kuvassa 6 on esimerkki sensorista, joka mittaa painetta ilmanpaineen mukaan. (12.)

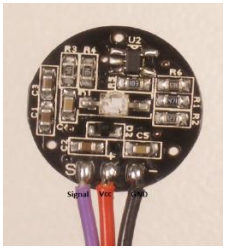


KUVA 6. HCA-BARO-painesensori (12)

4.1.3 Paikkasensori

Paikkasensoreista käytetyimpiä ovat nykyään optiset pulssisensorit, joiden avulla pystytään mittaamaan pulssia valon avulla. Kuvassa seitsemän on esitelty yksi pulssisensori, jonka voi helposti liittää Arduino-mikrokontrolleriin ja mitata esimerkiksi omaa pulssia. Kuvassa 8 on esitetty esimerkkikytkentä sensorista. S-kirjaimella merkitty napa tulee kytkeä signaalin output-kohtaan, ”+”-merkillä merkitty tulee kytkeä jännitteeseen ja ”-”-merkillä merkitty tulee kytkeä maahan. (13.)

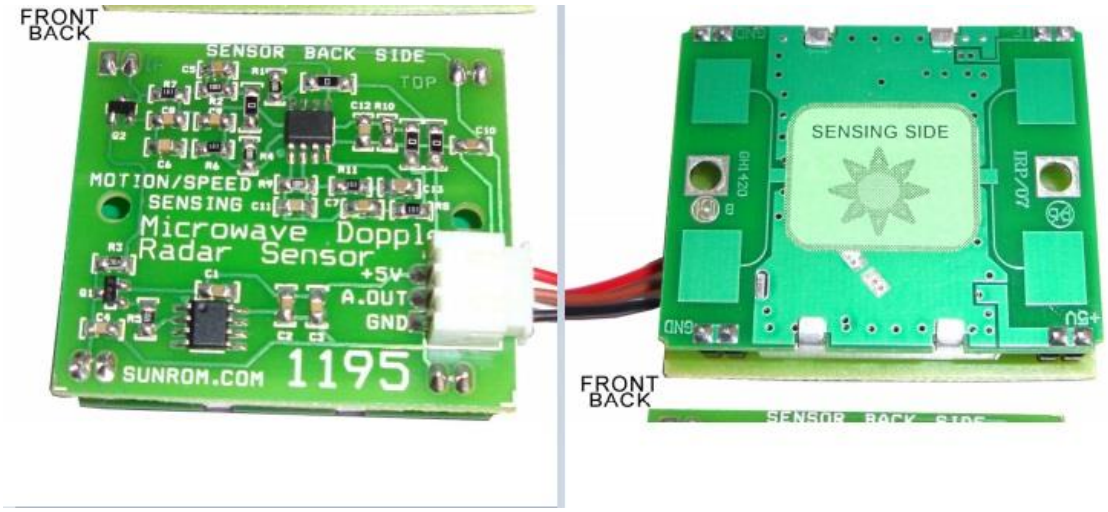
KUVA 7. Pulse Sensor AMPED (13)



KUVA 8. Pulse Sensor AMPED -kytkennät (13)

4.1.4 Nopeussensori

Nopeussensoreista käytetyimpiä ovat tutkasensorit. Tutkasensoreita käytetään muun muassa liikenteessä nopeuden valvonnassa. Kuvassa yhdeksän on esimerkki tutkasensorista. Kuvan sensori on Dopplerin tutkasensori, joka pystyy havaitsemaan liikettä ja liikkeen nopeutta. Tutkasensoreissa toimintaperiaate perustuu sähkömagneettisen signaalin lähettämiseen ja sen vastaanottamiseen eli niin sanottuun heijastumiseen. Kuvan yhdeksän sensorilla voidaan mitata nopeutta ja liikettä 20 metrin etäisyydeltä ja se pystytään kytkemään suoraan mikrokontrollerin pinniin, joka on tarkoitettu analogisen signaalin ulossyötölle. (14.)

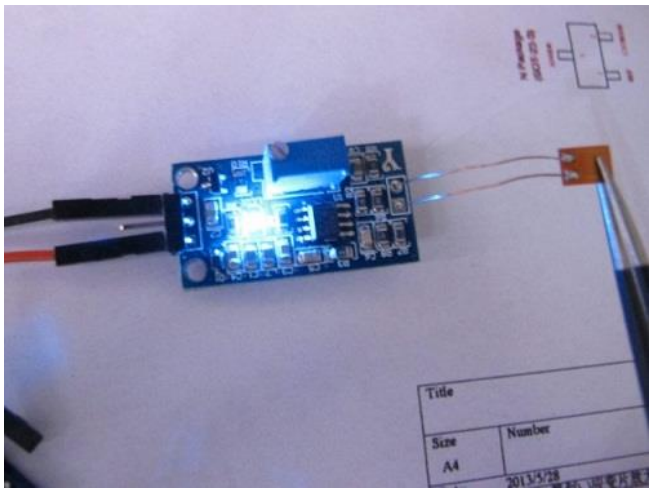


KUVA 9. Doppler-tutkasensori (14)

4.1.5 Voimasensori

Voimasensori eli esimerkiksi venymäliuskasensori perustuu materiaalin venymiseen. Materiaalin resistanssi muuttuu venymisen johdosta. Venymäliuskasensori on yleisin hydrauliikassa ja pneumatiikassa käytetty sensori. Liuskaan on kiinnitetty metallilangasta tai foliosta tehty vastus.

Kuvassa 10 on venymäliuskasensori osana Arduinon moduulia. Tässä venymäliuskassa on käytetty metallilangasta tehtyä vastusta.



KUVA 10. Arduinon moduuli, jossa on venymäliuskasensori (15)

4.1.6 Kiihtyvyyssensori

Kiihtyvyyssensoreita ja gyroskooppeja käytetään erittäin monessa sovelluksessa. Kiihtyvyyssensori mittaa liikettä kolmessa lineaarisessa suunnassa (x-, y-, z-vektori). On myös olemassa kiihtyvyyssensoreita, jotka mittaavat kiihtyvyyttä vain yhdessä suunnassa. Kiihtyvyyssensoreita, jotka mittaavat vain yhdensuuntaista liikettä, käytetään esimerkiksi junan nopeuden mittaamisessa. Puhelimissa on nykyään sensoreita, jotka mahdollistavat näytön kääntymisen samalla kun käännät puhelinta. Kuvassa 11 on käytetty IMU-sensoria, joka on liitetty mikrokontrolleriin. (16.)

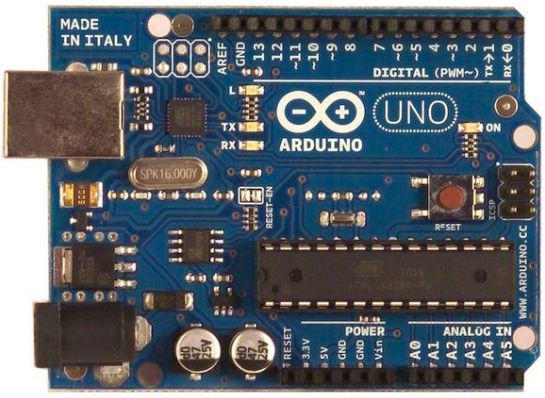
KUVA 11. SparkFun Triple Axis Accelerometer Breakout - ADXL335 ja kytkentä mikrokontrolleriin (16)

4.2 Sensorien ja moduulien käyttö ohjelmassa

Yleiskäyttöisellä prosessorilla tarkoitetaan mikroprosessori, joita käytetään mikrotietokoneissa ja sulautetuissa järjestelmissä. Mikrokontrollereissa on mikroprosessori, joka toimii suorittavana osana mikrokontrolleria. Mikrokontrolleri muodostaa kokonaisuutena tietokoneen, mikä pystyy hallinnoimaan yhtä tai useampaa sensoria riippuen mikrokontrollerin tehoista.

Prototyypitykseen ja pieniin POC-tehtäviin käytetään hyvin yleisesti jotain valmista mikrokontrolleria kuten Arduino. Arduino on edullinen ja helppokäyttöinen mikrokontrolleri, johon on liitetty toimintoja helpottavia toimintoja ja valmiita rajapintoja kytköksille kolmannen osapuolen komponenteille. Arduinoa pystyy komentamaan Arduinon omalla ohjelmalla, joka toimii C-kielellä. Seuraavissa kappaleissa kuvataan sensorien käyttöä Arduinon kanssa. (17.)

Sensoreille pitää määrittää ohjelman alussa jokin pinni, jonka kautta ne syöttävät arvoa mikrokontrollerille. Jos pinniä ei määritetä, Arduino käyttää oletuksena pinniä A0. (Kuva 12.)



KUVA 12. Arduino UNO (17)

Arduinolla pystytään mittaamaan suoraan esimerkiksi jännitettä sensorin signaalista, jos se otetaan huomioon ulossyötettävässä arvossa. Arduino mittaa A0 pinnin kautta analogista jännitettä välillä 0–5 V, mutta analogRead-funktio muuttaa sen A/D-muuntimella digitaaliseksi arvoksi 0 – 1023. Saatu arvo saadaan muutettua jännitteeksi kertomalla se jännitteen suhdanteella, kuten kuvassa 12 on tehty. Kuvan 13 ohjelma tulostaa serialille eli komentoikkunaan jännitteen arvon. Sensorien antamaan arvoon viitataan hyödyntämällä pinniä, johon sensori on kytketty.

```
void loop() {
  // read the input on analog pin 0:
  int sensorValue = analogRead(A0);
  // Convert the analog reading (which goes from 0 - 1023) to a voltage (0 - 5V):
  float voltage = sensorValue * (5.0 / 1023.0);
  // print out the value you read:
  Serial.println(voltage);
}
```

KUVA 13. Arduinon jännitelasku ja sensoriarvo

Sensorit antavat mikrokontrollereille sensorValuen eli sensorin arvon. Sensorin arvo vaihtelee todella paljon ja on myös riippuvainen mikrokontrollerista. Esimerkiksi Arduino antaa analogiseksi arvoksi maksimissaan 1023, jota voidaan verrata esimerkiksi jännitteeseen ja laskea siitä sensorin syöttämä tieto verraten jännitettä datalehdessä oleviin lukemiin. Digitaaliset pinnit voivat tuottaa 255 arvoa, joka vastaa 5:tä volttia.

Sensoriarvo muuttuu välillä 0–1023 jännitteen perusteella. Toisella muuttujalla pystytään laskemaan yhteys jännitteen ja sensoriarvon välillä. Saadaan lauseke:
float voltage = sensorValue * (5.0/1023.0);
Tämä arvo tulostetaan komentoikkunaan käyttämällä Serial.println-funktiota.
Serial.println(voltage);

5 SOVELLUSKEHITYS

Sovelluskehitysprojektin tavoitteena oli tuottaa suunnittelupohjaa tulevaisuuden ratkaisulle sekä toteuttaa yksi mahdollinen malli sovelluksesta, jolla voidaan kontrolloida ja seurata laitteen toimintoja. Opinnäytetyössä perehdyttiin käytettäviin web-teknologioihin perusteiden tasolla, sillä opinnäytetyöstä vastaava oli sovelluskehityksessä mukana suunnittelemassa ratkaisuja sekä toimi kontaktihenkilönä tilaajaan. Varsinaisesta kehitystyöstä vastasi sovelluskehitysprojektin kehitystiimi.

5.1 Käytettävät teknologiat

Teknologiakehitys on jatkuvasti kehittyvä ja sen ympärille rakennetaan jatkuvasti uusia työkaluja ja arkkitehtuurimalleja. Seuraavissa alaluvuissa esitetään yksi arkkitehtuurinen malli sekä kolme modernia teknologiaa sovelluskehityksestä.

5.1.1 RESTful-palvelut

REST (Representational State Transfer) on arkkitehtuurityyli, joka on suunniteltu käytettäväksi asynkronisissa järjestelmissä. Se koostuu rajoitteista, jotka ohjaavat arkkitehtuuristen elementtien toimintaa. Rajoitteita on lainattu muista arkkitehtuurityyleistä ja täydennetty yhdenmukaista rajapintaa käsittelevillä lisärajoitteilla. (18, s. 76–106.)

REST-teknologian rajoitteilla pyritään parantamaan seuraavia arkkitehtuurisia ominaisuuksia: suorituskykyä, skaalautuvuutta, yksinkertaisuutta, muunneltavuutta, näkyvyyttä, siirrettävyyttä ja luotettavuutta. Edellä mainittuja rajoitteita on yhteensä kuusi, joista yksi sisältää neljä lisärajoitetta. Jotta järjestelmää voitaisiin kutsua REST-arkkitehtuurityylin mukaiseksi, sen täytyy noudattaa kaikkia rajoitteita. (18, s. 76–86.)

Asiakas-palvelin

Asiakas-palvelinmalli on ensimmäinen rajoite. Asiakas-palvelimesta käytetään termiä client. Asiakas-palvelin on palvelin, joka toimii vastaanottavana ja reitittävänä käyttöliittymänä järjestelmään. Tällainen voi olla esimerkiksi hyvin yksinkertainen web-sovellus. Rajoite edellyttää, että järjestelmä

koostuu vähintään kahdesta komponentista: clientistä eli web-sovelluksesta ja palvelimesta, jossa voi olla tietokannat rajapintoineen. Clientin tehtävänä on huolehtia käyttöliittymästä ja palvelimen tehtävän on huolehtia laskennasta sekä tiedon tallennuksesta. Vastuiden jako eri osiin parantaa järjestelmän skaalautuvuutta ja mahdollistaa komponenttien kehittämisen irrallisena toisistaan. (18, s. 45, 78.)

Clientin ja palvelimen vuorovaikutus toimii esimerkiksi seuraavasti: palvelin tarjoaa kytköksen tietokantaan ja palveluihin ja se on koko ajan valmiustilassa odottaen kutsuja sen palvelinrajapintoihin. Client käyttää palvelimen palveluja tuottaakseen sisältöä ja logiikkaa. Transaktio tapahtuu seuraavasti: client lähettää palvelimme kyselyn, jonka palvelin käsittelee ja lähettää asianmukaisen vastauksen esimerkiksi HTTP-koodilla. (18, s. 45.)

Tilattomuus

Tilaton keskustelu clientin ja palvelimen välillä on toinen rajoite. Tämä tarkoittaa sitä, että jokaisen pyynnön pitää sisältää kaikki informaatio, mikä on tarpeellista pyynnön suorittamiseksi. Clientin vastuulla on tilatietojen hallinta, johon voi hyödyntää tilateknologioita kuten ngrx store. Tilateknologioiden käyttö varmistaa, että clientillä on kaikki tieto ajantasaista. Tilattomuus parantaa järjestelmän skaalaavuutta, koska palvelin ei käytä resursseja sovelluksen tilan ylläpitoon. Tehtävien jako asiakkaan ja palvelimen välillä parantaa myös vikatilanteiden käsittelyä, jolloin se parantaa järjestelmän luotettavuutta. Tilaton kommunikointi vaatii pyynnöissä enemmän välttämätöntä dataa, jotta palvelin osaa tunnistaa kyseisen clientin. (18, s. 79.)

Välimuisti

Edellä mainittujen rajoitteiden ongelmakohtia paikataan välimuisti-rajoitteella. Rajoite edellyttää, että pyyntöjen vastauksista tulee selvitä, voidaanko vastaus tallentaa välimuistiin vai ei. Jos välimuistin käyttö sallitaan, client voi tallentaa vastauksen ja tarvittaessa hakee sen sieltä uudestaan käyttöön myöhemmin. Näin ollen vältytään tekemästä samaa pyyntöä uudestaan ja vähennetään palvelimen kuormaa ja nopeutetaan järjestelmää kokonaisuudessaan. Välimuistin tarkoitus tässä asiayhteydessä on vähentää ylimääräisten pyyntöjen määrää palauttamalla tiedot suoraan välimuistista palvelinkutsun sijaan. (18, s. 48, 79.)

Palvelimella voi olla käytössään erilaisia puskureita, joilla voidaan nopeuttaa vastauksien palauttamista. Palvelimella voi olla oma välimuistikerros. Tällaista käsitettä kutsutaan elastisuudeksi. Elastisen haun avulla voidaan rakentaa palvelimelle välimuistikerroksia. Välimuistikerroksissa säilytetään useasti kysyttyä tietoa, jotta sitä tietoa ei joka kerta tarvitse kysyä tietokannasta. Välimuistin käyttäminen parantaa suorituskykyä, skaalautuvuutta sekä käyttäjäkokemusta. Se voi kuitenkin heikentää järjestelmän luotettavuutta, koska välimuistiin tallennettu vastaus saattaa toisinaan vanhentua odotettua aikaisemmin. Tällöin välimuistista haettu vastaus ei enää vastaa palvelimelta haettua vastausta. Tämän takia on tärkeää, että palvelin tekee tarkastuksia välimuistiin ja päivittää vanhentunutta tietoa. (18, s. 48, 80.)

Kerroksittainen järjestelmä

Kerroksittainen järjestelmä -rajoite käsittelee järjestelmän muodostamista hierarkkisista kerroksista. Rajoite edellyttää, että kerrosten tulee tarjota palveluita hierarkiassa yläpuolella oleville kerroksille ja hyödyntää hierarkiassa alapuolella olevien kerrosten palveluita. Komponentit voivat kommunikoida vain välittömästi ylä- tai alapuolella olevien kerrosten komponenttien kanssa, koska ne eivät ole tietoisia seuraavista kerroksista. (18, s. 46, 82.)

Kerroksittaisuus helpottaa järjestelmän monimutkaisuuden hallintaa ja edistää eri komponenttien itsenäisyyttä. Sen avulla on myös mahdollista kapseloida vanhoja järjestelmän osia sekä estää uusien palveluiden käyttö vanhentuneilta asiakkailta. Kerroksittaisuus voi lisäksi parantaa järjestelmän skaalautuvuutta ja tietoturvaa. Sen haittapuolena on kuitenkin tiedon prosessoinnin lisääntymisestä aiheutuva viive. (18, s. 83.)

5.1.2 Angular

Angular on AngularJS kirjaston pohjalta jatkettu teknologia. AngularJS ja Angular ovat molemmat frontend-teknologioita ja tarkoitettu yrityskäyttöön suurempien tuotteiden rakentamiseen. Angular kehitettiin AngularJS:stä löydetyn puutteen takia, sillä AngularJS ei mahdollistanut kaksisuuntaista datasidontaa luotettavasti. Tämä esti tietynlaisten sovellusten kehittämisen helposti ja tämän takia kehittäjät joutuivat rakentamaan kiertoteitä, jotka söivät sovelluksien tehoa. Edellä mainitusta syystä Google aloitti Angular-kehityksen yhdessä Microsoftin kanssa. Ajatuksena oli rakentaa sovelluskehikko, joka tukee niin pieniä kuin isojakin sovellusarkkitehtuureja. (19.)

Angular on kehitetty Angularin, Googlen ja Microsoftin yhteistyössä. Angular-teknologiaa pystyy hyödyntämään kolmella eri ohjelmointikielellä, jotka ovat TypeScript, JavaScript ja Dart. (19.)

TypeScript on Microsoftin kehittämä JavaScriptin superkieli. Tämä tarkoittaa siitä, että se on rakennettu JavaScriptin päälle. TypeScriptissä on samoja komentoja kuin JavaScriptissä, mutta siihen on tuotu monia huomattavasti helpottavia tekijöitä kehittämiseen. Yksi tämänlainen etu on tyypitysten määrittäminen, jonka ansiosta kääntäjät pystyvät etsimään virheitä heti, kun kehittäjä tekee niitä, eikä vasta käännös-tilassa. (20.)

5.1.3 Meteor

Meteor tai MeterJS on ilmainen avoimeen lähdekoodiin perustuva JavaScript-viitekehys. Meteor on kirjoitettu NodeJS-kirjastoa hyödyntäen. Meteor mahdollistaa nopean prototyypityksen ja sillä pystytään toteuttamaan heti alusta asti järjestelmäriippumattomia tuotteita. Järjestelmäriippumaton tarkoittaa sitä, että tuotteet toimivat esimerkiksi Androidilla, iOS:llä sekä web-sovelluksina. (21.)

Meteor integroituu hyvin MongoDB:n kanssa ja käyttää Distributed Data -protokollaa eli DDP:tä sekä publish-subscribe -kaavaa sanomien välittämisessä levittääkseen datamuutokset asiakasohjelmille eli clienteille ilman että kehittä kirjoittaa erikseen synkronointikoodia rajapinnaksi. Meteor on riippuvainen jQuery kirjastosta. JQueryä voidaan hyödyntää kaikissa JavaScript pohjaisissa ohjelmissa ja kirjoistoissa. (21.)

5.1.4 MongoDB

MongoDB on järjestelmäriippumaton, avoimeen lähdekoodiin perustuva sekä dokumenttiorientoitunut tietokantaohjelma. Se on luokiteltu NoSQL-tietokannaksi. MongoDB käyttää JSON-tyyppistä viestintää mallien kanssa. MongoDB yhtymä on kehittänyt MongoDB:n ja se on julkaistu GNU Affero General Public -lisenssin ja Apache-lisenssin turvin. MongoDB:n viehättävyys on sen tavassa käsitellä dataa JSON-muodossa. JSON on yksi yleisempiä datamalleja, ja se on erittäin joustava ja on helppokäyttöinen. (22.)

5.1.5 NodeJS

NodeJS perustuu avoimeen lähdekoodiin sekä järjestelmäriippumattomuuteen. NodeJS on JavaScript-teknologiaan perustuva ajonaikainen ympäristö, jonka avulla voidaan kehittää monenlaisia työkaluja ja sovelluksia. NodeJS ei ole JavaScript-kehikko, vaikka monet sen moduuleista on kirjoitettu JavaScriptillä. Ajonaikainen ympäristö tulkitsee JavaScriptiä käyttäen Google Chromen JavaScript-moottoria. (23. s.15–17)

NodeJS:ssä on tapahtumapohjainen arkkitehtuuri, joka kykenee asynkroniseen siirräntään. Nämä suunnitteluvalinnat tähtäävät suoritustehon ja skaalautuvuuden optimointiin verkkosovelluksissa useilla siirräntä toimenpiteillä. NodeJS on helppokäyttöinen kevyt palvelin, johon on helppo implementoida sovelluksia ja on täten hyvä työkalu prototyypitykseen. (23. s.15–17)

5.2 Käyttöliittymä- ja käyttäjäkokemussuunnittelu

Käyttöliittymäsuunnittelun tarkoitus on varmistaa tuotteen helppokäyttöisyys ja ymmärrettävyys kohderyhmälle. Käyttöliittymän toimiessa oletetusti käyttäjä voi keskittyä siihen, mitä haluaa tuotteella tehdä. Hyvä käyttöliittymäsuunnittelulla pystytään välttämään turhia tukipyyntöjä, mutta ennen kaikkea sen tarkoitus on saada asiakkaat tyytyväiseksi, kun he käyttävät tuotetta.

Käytettävyys määritellään viidellä eri laatuosatekijällä, jotka ovat opittavuus, tehokkuus, muistettavuus, virheet ja tyytyväisyys. Opittavuus tarkoittaa käyttäjien kykyä suorittaa perustehtävät, kun he käyttävät järjestelmää ensi kerran. Tehokkuus tarkoittaa aikaa, kuinka nopeasti käyttäjät pystyvät suoriutumaan tehtävistä, kun ovat jo tuttuja järjestelmän kanssa. Muistettavuudella tarkoitetaan käyttöliittymän muistettavuutta eli sitä, kuinka nopeasti käyttäjä saavuttaa taidon suoriutua tehtävistä nopeasti tauon jälkeen. Virheet osatekijä määritetään käyttäjien tekemien virheiden lukumäärän, vakavuuden ja niistä palautumisen perusteella. Tyytyväisyys määritellään käyttäjään mielipiteestä käyttöliittymän mielekkyydestä. (24.)

Käytettävyys on ominaisuus, jolla mitataan kuinka tehokasta ja miellyttävää palvelua tai tuotetta on käyttää. Käyttökokemus tarkoittaa käyttäjän tuntemusta palvelua käytettäessä. Tähän vaikuttavat palvelu ja sen ominaisuudet, käyttötilanne ja -ympäristö, käyttäjän aikaisemmat kokemukset ja mielipiteet palvelun ominaisuuksista etukäteen, sisällöstä, hyödyistä sekä palveluntarjoajasta. (24.)

6 PROTOTYYPPI OSANA TUOTEKEHITYSTÄ

Prototyypitys on nopea tapa kerätä sisäistä tai ulkoista palautetta testiryhmän avulla. Prototyyppi voidaan rakentaa demonstroimaan tuotteen ulkoasua, tuntoa, tuotteen teknistä toimivuutta tai sitä, että missä tuote pitäisi tuottaa. Prototyypitys on yleisesti kallis prosessi ja sitä edeltää usein POC eli Proof of Concept -vaihe, jossa todennetaan tuotteen toteutuskelpoisuus. Tässä osiossa käydään läpi prototyypin hyötyjä ja milloin se on tarpeen. (25. s. 19–20)

6.1 Prototyypin tarpeellisuuden määrittäminen

Ennen kuin prototyypin kehitys aloitetaan, on hyvin tärkeää tietää prototyypin tavoitteet, jotta osataan määritellä tarkkuus mihin asti prototyyppi kehitetään. Prototyyppiä voidaan käyttää seuraaviin tarkoituksiin:

- käyttäjien vaatimuksen kerääminen
- järjestelmän spesifikaatioiden validoiminen
- käyttö- ja/tai suunnitelma virheiden tunnistaminen
- POC-tuotteen toimittaminen
- tuotteen suunnittelun sekavuuden ratkaiseminen
- lisätäkseen rakentavaa palautetta käyttäjän mukaan ottamisella
- läpinäkyvyyden lisääminen tuotekehitykseen
- laadun varmistaminen
- markkinointi-demon luominen. (25. s 19–20.)

Prototyyppi voi olla hyvin yksinkertainen. Esimerkiksi se voi olla yksinkertaisia piirustuksia, joilla demonstroidaan nopeasti sovelluksen arkkitehtuuria, toiminnallisuutta ja käyttöliittymiä. Tämänkaltaiset prototyypit voivat toimia jo hyvinä indikaattoreina asiakkaille, että mikä on oikea suunta kehityksessä ja mikä ei. (25. s 19–22.)

6.2 Prototyypin suunnittelu ja hallinnointi

Mikäli prototyyppi vaatii enemmän kuin yksinkertaisen tehtävän tai piirroksen luonnistuakseen, se pitää suunnitella ja hallinnoida, ennen kuin siihen käytetään aikaa ja rahaa. On huomioitava, että

prototyyppi ei suoranaisesti edistä varsinaisen tuotteen kehitystä. Esimerkiksi jos tehdään prototyyppi ohjelmasta, hyvin usein prototyypin arkkitehtuuria ei ole mietitty lainkaan. Prototyypillä on haluttu saada mahdollisimman nopeasti käyttäjälle näkyviä osia näkyviin, jotta voidaan testata esimerkiksi käyttöliittymää. Tämä tarkoittaa sitä, että varsinaisen tuotteen kehityksessä täytyy uudelleen rakentaa prototyypillä saavutetut toiminnallisuudet. Prototyyppiä ei voi käyttää suoraan pohjana jatkokehitykselle, sillä sitä ei olla suunniteltu arkkitehtuurisesti pitkäaikaiseksi. (25, s. 22.)

Seuraavassa luettelossa havainnollistetaan yksi esimerkki hyvään prototyyppisuunnitteluun. (25, s. 23–24.)

1. Määrittele prototyypin tarkoitus. Onko prototyypillä tarkoitus oppia lisää käyttäjän vaatimuksista, demonstroida toiminnallisuutta johdolle, vahvistaa järjestelmän määrityksiä, ratkaista epäjohdonmukaisuuksia suunnittelun alkuvaiheessa, tutkia ratkaisuja tiettyihin käyttöliittymäongelmiin, integroidakseen muita järjestelmiä vai luoda markkinointidemo.
2. Aseta prototyypille lähestymistapa. Paras prototyyppi on yksinkertaisin prototyyppi, joka pystyy palvelemaan sen tarpeensa. Elinkelpoinen vähimmäistuote -ajattelumalli eli MVP-ajattelumalli on hyvä lähestymistapa prototyyppien rakentamisessa.
3. Alleviivaa kokeellinen suunnitelma. Tämä sisältää kokeellisen arvon tunnistamisen, tehtävien mittauksien tunnistamisen sekä prototyypistä saadun lopputuloksen analysointimenetelmien tunnistamisen.
4. Luo aikataulu hankinnalle, toteuttamiselle ja testaamiselle. Prototyyppi mielletään projektina, joten on äärimmäisen tärkeää huomioida aikataulu edellä mainituille vaiheille.
5. Toteuta prototyypin suunnittelu.

7 TULOKSET

7.1 Projektin hallinta

Projektissa käytettiin projektinhallintatyökaluna Scrumia, joka on yleisesti käytetty muun muassa ohjelmistokehityksessä sen salliman ketteryyden vuoksi. Projekti jaettiin kehitysjaksoihin eli sprintteihin. Yhden sprintin pituus määriteltiin kahdeksi viikoksi. Sprinttien suunnittelupalaverissa sovittiin sprintissä suoritettava työ. Sprintin onnistumiset arvioitiin sprinttikatselmuksessa demon jälkeen. Toimin laitekehityksemme scrummasterina ja olin vastuussa projektin etenemisestä ja tavoitteiden saavuttamisesta kokonaisuudessaan. Khoa Bùi toimi sovelluskehityksen scrummasterina. Sovelluskehityksen kehitystiimin jäseninä olivat Long Hoang ja An Phạm.

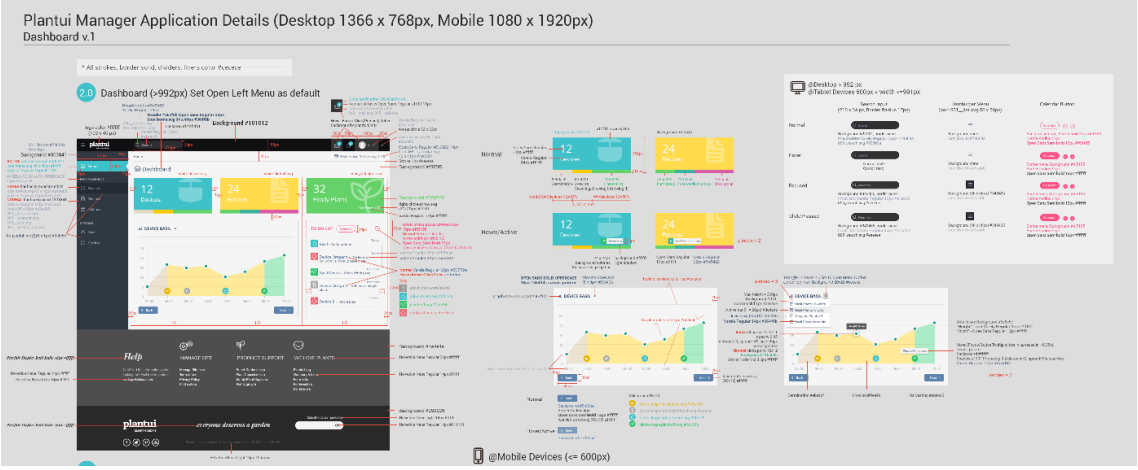
Projektille luotiin tuotteen kehitysjono, joka sisälsi koonnin kaikista käyttäjätarinoista, jotka piti suorittaa projektin valmistumiseksi. Nämä käyttäjätarinat ovat valmiita otettaviksi toteutukseen sprinttiin. Työssä käytettiin Excelillä tehtyä pohjaa työskentelymäärien ja tehtävien seuraamisessa. Excelillä tehty pohja on kätevä ja yksinkertainen työkalu pienessä projektissa verrattuna muihin vastaaviin.

Käytössä oli Basecamp-ohjelma, joka on selainpohjainen projektinhallintatyökalu. Sen kautta pidettiin yhteyttä tilaajaan ja siellä on helppo pitää tehtyjä dokumentteja kaikkien projektiin kuuluvien näkyvillä. Ohjelman kautta on helppo käydä keskustelua mahdollisista ongelmatilanteista usean tahon kanssa. (10.)

Projektissa pidettiin viikoittain ohjelmistotiimin ja laitteistotiimin kanssa palaveri, missä keskusteltiin haasteista, ideoista ja toteutuneesta työstä viime viikon ajalta. Näin molemmilla tiimeillä säilyi tieto projektin kokonaiskulusta. Tuotteen omistajan, Plantui Oy:n, kanssa pidettiin palavereja aina, kun oli haasteita, joihin ei tiimin sisältä löytynyt vastausta. Jokaisen sprintin jälkeen pidettiin sprinttikatselmus, jonka osana oli demo tilaajalle. Mikäli demon aikataulu ei sopinut Plantui Oy:lle, heille pidettiin erillinen raportti projektin edistymisestä.

7.2 UI/UX

Käyttöliittymä- ja käyttökokemussuunnittelu yhteistyössä Plantui Oy:n kanssa tuotti monta eri tapaa katsoa ja käyttää sovellusta. Sovelluksen suunnittelu käytiin kolmessa eri iteraatiossa Plantuin kanssa, jossa jokaisen iteraation jälkeen pidettiin lyhyt demo toteutusideasta yritykselle. Jokaisella palautuskerralla yritys pystyi vaikuttamaan suunnittelun suuntaan haluamallaan tavalla. Tuotteen tilaajat harvoin tietävät, millaisen tuotteen haluavat, kun ovat tilaamassa sen. Haluttuun tuotteeseen päästään aina keskustelemalla ja palautteen kautta jatkuvasti parantaen tuotteen suunnittelua ja käytettävyyttä. Kuvassa 14 esitellään tuotteen toteutuva käyttäjän kaavio, mikä tarkoittaa käyttäjän toimintaa sovelluksessa ja ajatuskartta siitä, mitä milloinkin on mahdollista tehdä.



KUVA 14. Käyttäjän kaavio

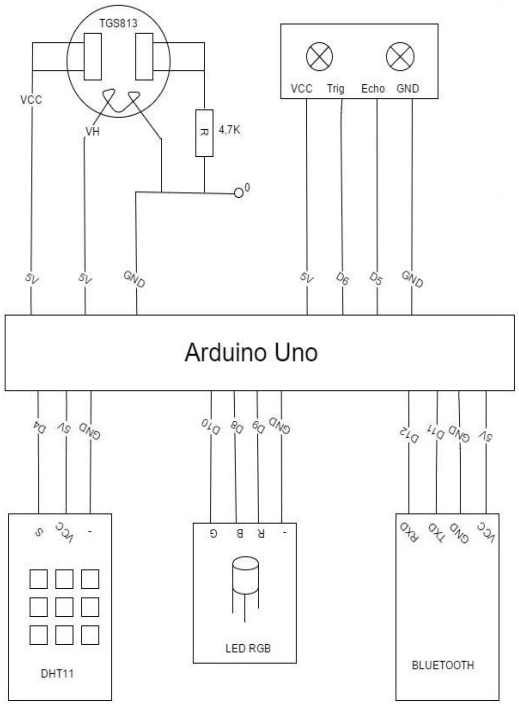
7.3 Laitekehitys

Laitekehityksen iteraatiot sisälsivät suureksi osakseen tutkimusluonteisia käyttäjätarinoita, joissa haettiin aluksi mahdollisia toimitapoja ja jonka jälkeen kehitettiin kuvanmukainen liitäntä (kuva 15), jolla pystyttiin toteuttamaan POC-versio tuotteesta ilman tiedonsiirtoa web-palvelimelle.

KUVA 15. Prototyyppi

Laitekehityksen päällimmäisenä tarkoituksena oli identifioida erilaisia tapoja mitata kasveista erilaisia suureita, jotka voisivat olla hyödyllisiä loppukäyttäjälle. Näihin suureisiin kuului vihermassan mittaaminen, kasvien pituuden, veden määrän, ilman partikkelien sekä lämpötilan mittaaminen. Samanaikaisesti haettiin malleja manipuloimaan laitteen tuottamaa valotusta sekä veden pumppausta erinäköisin kasvireseptein, joiden tarkoituksena on ohjata kasvin kasvatusta laitteessa loppukäyttäjän haluamalla tavalla.

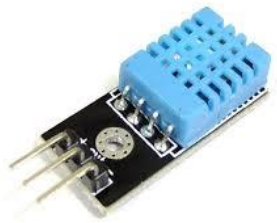
Tutkimme eri tapoja toteuttaa haluttujen määreiden mittaaminen, ja päädyimme tilaamaan kaasusensorin ja sensoripaketin, jossa oli suuri määrä sensoreita. Näistä sensoreista päädyimme muutamaan, joista rakensimme toimivan kokonaisuuden mittaamaan haluttuja suureita. Kuvassa 16 on kuvattu kokonaisvaltainen kytkentäkaavio käytettävien sensorien kanssa. Seuraavissa alaluvuissa käydään läpi valittujen sensorien toimintalogiikkaa ja siitä, mitä niillä voidaan mitata.



KUVA 16. Kytkentäkaavio

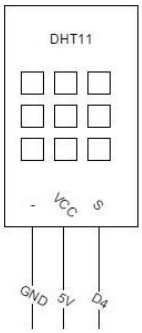
7.3.1 DHT11

DHT11 on lämpötila- ja kosteussensori, joka antaa sensoriarvon digitaalisena signaaliulosanti. Sensori sisältää vastus-tyyppisen kosteusmittaus komponentin ja NTC lämpötila mittauskomponentin. DHT11 on mahdollista kiinnittää 8-bittiseen mikrokontrolleriin. (27.) (Kuva 17.)



KUVA 17. DHT11 sensorimoduuli (27)

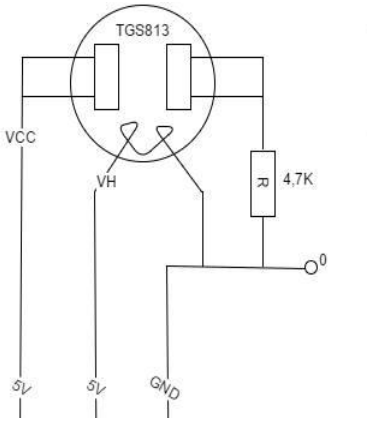
DHT11:n kytkentä on hyvin yksinkertainen, sillä sen signaaliulosanti on kääntäen verrannollinen sen luomaan vastukseen virtaan, mikä muuttuu lämpötilan muuttuessa. DHT11 kytketään maahan, virtaan ja signaali-ulosantiin, joka on tässä tapauksessa Arduinon digitaalinen pinni 4. (Kuva 18.)



KUVA 18. DHT11 kytkentäkaavio

7.3.2 Figaro TGS 813

TGS 813 -kaasusensorissa on korkea herkkyysaste metaanille, propaanille ja butaanille, mikä tekee siitä hyvän luonnollisten kaasujen ja nestekaasujen tarkkailuun. Figaro TGS 813 oli pelkkänä sensorina, eikä moduulina, jolloin se piti ensin kiinnittää vastuksiin ja sitä kautta vasta mikrokontrolleriin. Kaasusensori kytkettiin neljään ulkoiseen syötteeseen: 5 V:n jännite, 5 V:n jännite, maa ja signaali-ulosanti. (Kuva 19.) (28.)



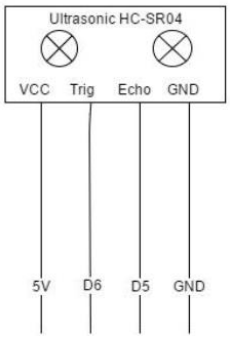
KUVA 19. Figaro TGS 813 sensori ja kytkentäkaavio (27)

7.3.3 Ultraäänisensori HC-SR-04

HC-SR-04 -ultraäänisensorilla voi mitata 2–400 cm:n matkaa 3 mm:n tarkkuudella. Moduuli lähettää automaattisesti kahdeksan 40 kHz:n signaalia ja tunnistaa, mikäli joku näistä signaaleista palaa kaikuna. (Kuva 20.)

KUVA 20. Ultraäänisensori HC-SR-04 (28)

Ultraäänimoduuli on kytketty VCC-pinnistä 5 V:n jännitteeseen, Trig-pinnistä Arduinon D6-pinniin, Echo-pinnistä Arduinon D5-pinniin ja GND-pinnistä maahan. Trig-pinniä hallinnoidaan lähetettäviä signaaleja ja Echo-pinnillä kuunnella näiden signaalien vastetta kaiusta. (Kuva 21.)



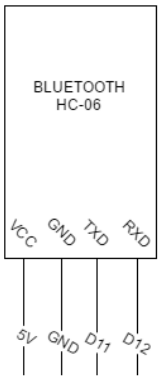
KUVA 21. Ultraäänimoduulin kytkentäkaavio

7.3.4 Bluetooth moduuli HC-06

HC-06 toimii yhteytenä komentoriviin, jonka kautta voidaan lähettää ja vastaanottaa dataa. Käytimme HC-06 luodaksemme demonstroitavan ympäristön ja yhteyden mikrokontrolleriin etänä. Loimme puhelimelle kevyen demo-sovelluksen, jolla käskytimme mikrokontrolleria Bluetoothin kautta. Samalla pystyimme varmistamaan, että saimme oikeanlaista dataa muilta sensoreilta ja pystyimme tarkistamaan niiden kalibrointeja sekä tarkkuuksia käytännössä. (Kuva 22.)

KUVA 22. Bluetooth moduuli HC-06 (29)

Bluetooth moduuli kytkettiin kiinni neljällä pinnillä. Pinnit on merkitty niiden käyttöperiaatteen mukaisesti. VCC tarkoittaa jännitettä ja on kytketty 5 V:n jännitteeseen. GND tarkoittaa maata ja kytketty maahan. TXD tarkoittaa lähetettävää viestiä ja on kytketty Arduinon D11-pinniin, joka on ohjelmoitu lähetettäväksi pinniksi. Vastaavasti RXD tarkoittaa vastaanotettavaa viestiä ja on kytketty Arduinon D12-pinniin, joka on ohjelmoitu vastaanotettavaksi pinniksi. (Kuva 23.)



KUVA 23. Bluetooth moduulin kytkentäkaavio

7.4 Sovelluskehitys

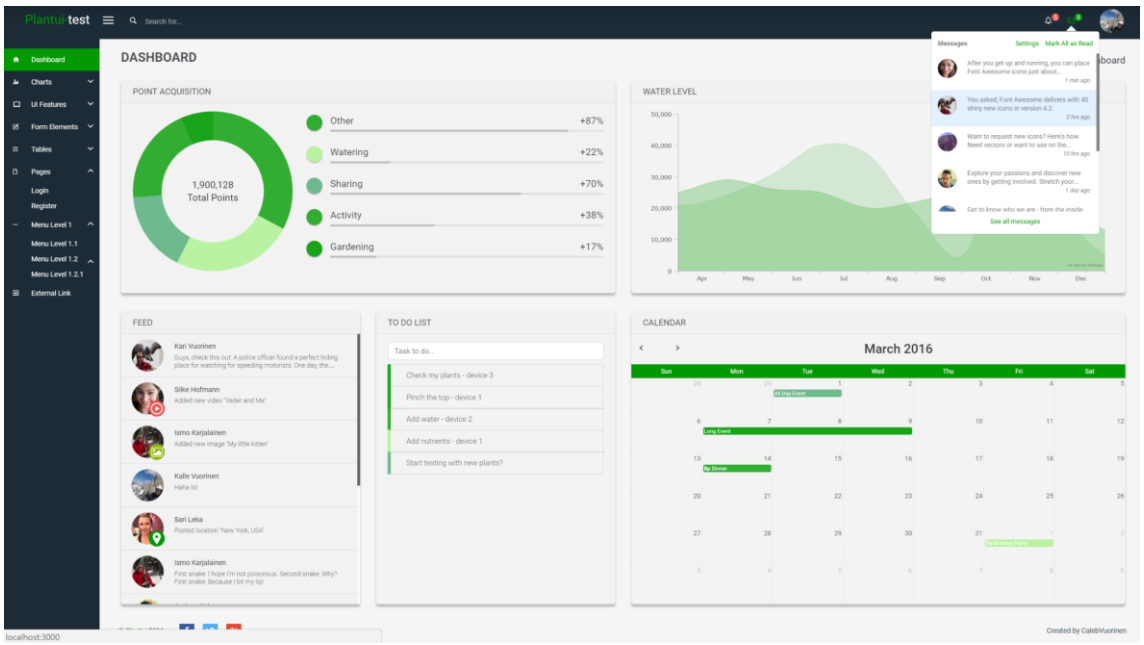
Sovelluskehitys alkoi suunnittelemalla ja tutkimalla moderneja web-teknologioita, joilla prototyyppi voitaisiin toteuttaa. Piti päättää teknologiapaketti, joka ei vaadi liian suurta panosta itse teknologian opetteluun, mutta on silti moderni ja tehokas.

Aloitimme frontend-teknologioiden päättämisestä, sillä ne olivat lähes kaikille projektissa mukana olleille tutumpia. Frontend-teknologioissa päädyttiin nopeasti JavaScript-pohjaisiin ratkaisuihin ja

keskustelimme lähinnä Googlen AngularJS:stä ja Facebookin ReactJS:stä. Haasteina aihioissa oli se, että Angular oli juuri päivittämässä AngularJS:stä Angulariin, joka on kehitetty Googlen sekä Microsoftin toimesta. Valitettavasti Angular ei ehtinyt viralliseen julkaisuversioon tarpeeksi ajoissa ja tästä syystä emme valinneet sitä. (23.) ReactJS on modulaarinen ja hieman enemmän funktionaalinen kieli kuin Angular-perheen ohjelmistokehikot. Funktionaalinen tarkoittaa ohjelmointikielessä sen rakenteellisesta kirjoitustapaa. (30. s. 1–2)

Backend-teknologiassa haettiin ensisijaisesti teknologiaa, joka mahdollistaa skaalaavuuden, sekä helpon editoinnin. Samalla suunnittelimme tietokantaan tarvittavia tietokantatauluja ja sitä, miten asiat yhtyvät keskenään ja mitä tietoja sinne voidaan tarvita. Tietokantataulut ovat salassapitovelvollisuuden alaisia ja tästä syystä niitä ei tuoda julki tässä opinnäytetyössä.

Muutaman palaverin jälkeen päädyttiin käyttämään ReactJS, Bootstrap ja MongoDB -sovelluspakettia kehitystyössä. Web-sovelluksen rakentaminen oli kaksijakoista, sillä siinä piti huomioida myös tietokantaratkaisut. Tietokantakaaviot luotiin yhdessä laitekehityksen ja web-kehityksen kanssa. Tietokantoja ei ehditty hyödyntämään varsinaisessa prototyypissä, sillä toteutuksen aika loppui kesken. Web-sovelluksessa keskityttiin enemmän frontend-kehitykseen ja sen tuomaan arvoon prototyypissä. Kuvassa 24 on esitetty käyttäjän etusivu, kun ohjelmaan on kirjautunut. Varsinainen prototyyppi on saatavilla osoitteessa https://plantuitest.herokuapp.com.



KUVA 24. Web-sovelluksen aloitusnäkymä kirjautuneena

8 YHTEENVETO

Tässä opinnäytetyössä on perehdytty mikrokontrollereiden toimintaan, sensoreiden toimintaan osana mittausjärjestelmää, projektihallintaan ketterässä kehitysprojektissa, käyttöliittymäsuunnitteluun, web-teknologioihin, web-sovelluksen toteuttamiseen sekä prototyypin konkreettiseen käyttämiseen osana tuotekehitystä. Työssä käsiteltiin sensoreiden käyttöä Arduinon omassa ohjelmassa. Neljännessä pääluvussa käsiteltiin sensorien kuusi päätyyppiä sekä niiden käyttö ohjelmassa. Viidennessä pääluvussa käsiteltiin web-sovelluksen nykytrendeihin liittyvät teknologiat sekä käyttöliittymäsuunnittelu. Kuudennessa pääluvussa käytiin prototyypin käyttäminen osana tuotekehitystä.

Sensoreiden avulla voidaan rakentaa hyvin helposti pelkistettyjä mittausjärjestelmiä. Sensorien kirjo on todella laaja ja yhtä sensoria voi käyttää moneen eri tarkoitukseen. Mikrokontrollereita on integroituna sekä avoimena käytettävissä prototyypitykseen. Avoimia mikrokontrollereita ovat esimerkiksi Arduinon mikrokontrollerit sekä Iproto Xi -sarjat. Näiden ohjelmistot ja koodistot ovat avoimesti käytettäviä ja siksi hyvin kehitysystävällisiä.

Web-sovelluksia voidaan kehittää monella eri teknologialla ja nykyajan trendeihin hyödyntäen. On kuitenkin huomioitava, että teknologia kehittyy jatkuvasti ja uusia teknologioita luodaan koko ajan. On siis tärkeää aina tarkastella nykytilanne teknologioiden suhteen, kun ollaan aloittamassa uutta projektia. Tämä mahdollistaa mahdollisimman nopean kehityksen sekä sen, että kehittäjät pysyvät motivoituneina oppiessaan uusia ja parempia teknologioita.

Prototyypin rakentaminen on aina halvempi ratkaisu kuin varsinaisen tuotteen kehitys suoraan. Prototyypittämisellä voidaan tarkastella ja identifioida kuluttajan tarpeita ja käytettävyyteen liittyviä haasteita. Näin ollen pystytään muuttamaan alkuperäistä suunnitelmaa ja toteutuksia kevyemmin ja nopeammin palautteen perusteella.

Opinnäytetyön tavoitteena oli rakentaa yksi yhtenäinen prototyyppi, jota olisi loppuvaiheessa voitu koestaa käyttäen useampaa Plantui Smart Gardenia hyödyksi. Tavoite muuttui kehitystyön aikana ja tiedonsiirtoon tai tietokantaan ei kiinnitetty enää niin paljoa huomiota, vaan kehitystyö suunnattiin näkyvään osioon web-sovelluksessa sekä sensorimallien arvioimiseen laitekehityksessä. Tämän opinnäytetyön lisäksi yritykselle on toimitettu kaksi projektiraporttia englanniksi sekä toimiva sovellus demonstroimiseen.

Projekti oli onnistunut, vaikka sen sisältö muuttui jatkuvasti. Projektin sisällön muuttuminen on olennaista ketterään kehitystyöhön ja se mahdollistaa tuotteen lopputuloksen olevan lähempänä asiakkaan tilaamaa tuotetta. Projektilla onnistuttiin luomaan prototyyppi, jota on jälkeenpäin käytetty referenssinä varsinaiselle kehitystyölle. Yhteistyö on jatkunut projektin jälkeen lyhyinä jaksoina.

Opinnäytetyö oli hyvin mielenkiintoinen ja silmiä avartava. Työ oli ajankohtainen modernien teknologioiden ja projektinhallinnan takia. Se sisälsi erittäin laajan toteutuksen pienestä sensoriarvosta aina näkyvään tuotteeseen verkossa. Työn mielekkyyttä lisäsi liiketoimintaprojektin sitominen projektiin sekä aktiivinen keskustelu yrityksen ja projektiin osallistuneiden kesken. Työ opetti ennen kaikkea vastuuta ja projektinhallintaa projektista, jossa oli mukana useampia henkilöitä.

LÄHTEET

1. Loiske, Janne. Plantui story in short – How Plantui company got started. Plantui. Saatavissa: https://plantui.com/plantui-story-in-short-how-plantui-company-got-started/ Hakupäivä: 16.3.2016
2. Loiske, Janne. 2016. Esitys. Mativation Oy, Salo.
3. Plantui 6 Smart Garden. Plantui. Saatavissa: http://plantui.com/fi/smart-gardens/plantui-6-smart-garden/ Hakupäivä: 16.3.2016
4. Loiske, Janne – Vuorinen, Kari – Ketonen, Timo – Teir, Tore – Valtonen, Ville 8.10.2015. Plantui. Saatavissa: http://plantui.com/wp-content/uploads/2015/10/Plantui_IM_151008.pdf Hakupäivä: 16.3.2016
5. Plantui For Investors. Plantui. Saatavissa: http://plantui.com/for-investors/ Hakupäivä: 16.3.2016
6. Ketonen, Timo. 17.8.2015. Plantui kasvualustat. Aboa Advest. Saatavissa: http://www.aboa-advest.fi/news/plantui-kasvupolulla-aasiassa-ja-euroopassa/ Hakupäivä: 16.3.2016
7. Plantui 6 Smart Garden. Plantui. Saatavissa: http://plantui.com/smart-gardens/plantui-6/ Hakupäivä: 16.3.2016
8. Schwaber, Ken – Sutherland, Jeff. 2016. The Scrum Guide. Saatavissa: http://www.scrum-guides.org/docs/scrumguide/v2016/2016-Scrum-Guide-US.pdf#zoom=100 Hakupäivä: 13.3.2017.
9. Scrum process. 2009. Lakeworks. Saatavissa: https://commons.wikimedia.org/wiki/File:Scrum_process.svg Hakupäivä: 16.3.2016.
10. Basecamp. Saatavissa: https://basecamp.com/ Hakupäivä: 16.3.2016
11. Patel, Nilesh – Mathurkar, Swarup – Lanjewar , Rahul – Bhandekar, Ashwin. 2013. Microcontroller based drip irrigation system using smart sensor. Saatavissa: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6726064 Hakupäivä 13.03.2017.
12. HCA-BARO. Sensortechnics. Saatavissa: http://www.sensortechnics.com/en/products/pressure-sensors-and-transmitters/barometric-pressure-sensors/hca-baro/ Hakupäivä 16.05.2017.
13. Pulse Sensor Amped. World Famous Electronics. Saatavissa: https://pulsesensor.com/pages/pulse-sensor-amped-arduino-v1dot1 Hakupäivä 16.05.2017.

14. Microwave Doppler Radar Sensor for Motion and Speed Sensing. Sunrom Electronics. Saatavissa: http://www.sunrom.com/p/microwave-doppler-radar-sensor-for-motion-and-speed-sensing Hakupäivä 16.05.2017.
15. Strain Gauge Module. MindKits Limited. Saatavissa: http://www.mindkits.co.nz/strain-gauge-module.aspx Hakupäivä 13.03.2017.
16. Coleman Benson. 2012. Arduino 5 Minute Tutorials: Lesson 7 – Accelerometers, Gyros, IMUs. RobotShop. Saatavissa: http://www.robotshop.com/blog/en/arduino-5-minute-tutorials-lesson-7-accelerometers-gyros-imus-3634 Hakupäivä 16.05.2017.
17. Arduino. Saatavissa: https://www.arduino.cc/en/main/arduinoBoardUno Hakupäivä 16.05.2017.
18. Fielding, R. 2000. Architectural Styles and the Design of Network-based Software Architectures. Irvine: University of California. Saatavissa: https://www.ics.uci.edu/~fielding/pubs/dissertation/fielding_dissertation.pdf Hakupäivä 08.01.2017.
19. Angular. Saatavissa: https://angular.io/ Hakupäivä 13.03.2017.
20. TypeScript. Saatavissa: https://www.typescriptlang.org/ Hakupäivä 13.03.2017.
21. Meteor. Saatavissa: https://www.meteor.com/ Hakupäivä 13.03.2017.
22. MongoDB. Saatavissa: https://www.mongodb.com/ Hakupäivä 13.03.2017.
23. NodeJS. Saatavissa: https://nodejs.org/en/ Hakupäivä 13.03.2017.
24. Nielsen, Jakob. 2012. Usability 101: Introduction to Usability. Nielsen Norman Group. https://www.nngroup.com/articles/usability-101-introduction-to-usability/ Hakupäivä 13.03.2017.
25. Frank W. Liou. 2007. Rapid prototyping and engineering applications – A toolbox for prototype development. Taylor & Francis Group. Saatavissa: https://books.google.fi/books?hl=fi&lr=&id=0jbOIWH0EiwC&oi=fnd&pg=PP1&dq=prototype+part+of+development&ots=AqkrptZRBG&sig=Ov5HfWdWWwFdabUGR2OQBhFOoJE&redir_esc=y#v=onepage&q&f=false Hakupäivä 16.05.2017.
26. DHT11 Humidity & Temperature Sensor. 2010. D-Robotics. Datalehti. Saatavissa: http://www.micropik.com/PDF/dht11.pdf Hakupäivä 16.05.2017.
27. TGS 813 - for the detection of Combustible Gases. Figaro. Datalehti. Saatavissa: http://www.figarosensor.com/products/813pdf.pdf Hakupäivä 13.03.2017.
28. Santos, Rui. 20.8.2014. Complete Guide for Ultrasonic Sensor HC-SR04. Saatavissa: http://randomnerdtutorials.com/complete-guide-for-ultrasonic-sensor-hc-sr04/ Hakupäivä 16.05.2017.

29. HC-05 Bluetooth Module. ElectroSome. Saatavissa: https://electrosome.com/shop/hc-05-bluetooth-module/ Hakupäivä 16.05.2017.
30. Michaelson Greg. 2011. An Introduction to Functional Programming Through Lambda Calculus. Courier Corporation. Saatavissa: https://books.google.fi/books?hl=en&lr=&id=gKvwPtvsSjsC&oi=fnd&pg=PR3&dq=functional+programming+thesis&ots=YynTqePweU&sig=-8Wqyxc8ZhqgaKQI-72HWH3fo04&redir_esc=y#v=onepage&q&f=false Hakupäivä 13.03.2017