Kalle Palonen

Palvelinhuoneen lämmöntarkkailujärjestelmä Arduino-mikrokontrollerilla

Metropolia Ammattikorkeakoulu

Insinööri (AMK)

Tietotekniikka

Insinöörityö

30.03.2014

Metropolia

Tekijä Otsikko Sivumäärä Aika	Kalle Palonen Palvelinhuoneen lämmöntarkkailujärjestelmä Arduino-mikrokontrollerilla 36 sivua + 7 liitettä 30.03.2014
Tutkinto	Insinööri (AMK)
Koulutusohjelma	Tietotekniikka
Suuntautumisvaihtoehto	Tietoverkot
Ohjaaja(t)	Yliopettaja Janne Salonen Järjestelmäasiantuntija Henri Siponen

Palvelinhuoneiden olosuhteet ovat nykypäivänä huolellisen tarkkailun alla. Tarkkailuun käytetään erilaisia ohjelmistoja sekä laitteita, jotka ilmoittavat ylläpitäjille tilan kunnon. Järjestelmät ovat usein verkkovalvontaohjelmistoihin liitettyinä.

Opinnäytetyön toimeksiantajana oli Suomen Lähetysseura ry. Toimeksiantona oli lämmöntarkkailujärjestelmän rakentaminen sekä käyttöönotto Suomen Lähetysseura ry:n palvelintiloissa.

Projektin päätavoitteena oli luoda toimiva lämmöntarkkailujärjestelmä. Toisena tavoitteena pidettiin sitä, että järjestelmän rakennus saataisiin dokumentoitua niin tarkasti, että se voidaan rakentaa tarvittaessa uudestaan helposti ja vikatilanteet olisivat helppo korjata.

Menetelminä projektin kuluessa käytettiin käytännön testaamisesta sekä vertailua. Testaamista käytettiin lähinnä koodien tarkistamisessa. Vertailua hyödynnettiin lämpötila-anturin luotettavuuden mittaamisessa sekä kahden IT-infrastruktuurivalvonnan ohjelman vertailussa.

Projektin tuloksena luotiin toimiva lämmöntarkkailujärjestelmä, jota on mahdollisuus tulevaisuudessa laajentaa. Laitteen rakennus dokumentoitiin mahdollisimman tarkasti, joten järjestelmä on tarpeen vaatiessa mahdollisuus rakentaa uudelleen sekä laajentaa.

Avainsanat	Arduino, Mikrokontrolleri, Nagios3, Linux

Author Title Number of Pages Date	Kalle Palonen Server Room Temperature Monitoring System Using Arduino Microcontroller 36 pages + 7 appendices 30 March 2014
Degree	Bachelor of Engineering
Degree Programme	Information Technology
Specialisation option	Information Networks
Instructor(s)	Janne Salonen, Senior Lecturer Henri Siponen, Systems Specialist

Nowadays server room conditions are under careful observation. A variety of software and devices that indicate the status of the conditions to the administrators are used for monitoring. The systems are often connected to IT-infrastructure monitoring software.

The thesis was commissioned by the Finnish Evangelical Lutheran Mission. The task was to build a system for temperature monitoring and implement it in the server room of the Finnish Evangelical Lutheran Mission.

The main objective was to create a fully functional temperature monitoring system. Another objective was to document the system in such detail that it can be easily rebuilt and troubleshooting would be easy to perform.

The methods used during the project were comparison and hands-on testing. Hands-on testing was used mainly in code checking. The comparison was used to test the reliability of the temperature sensor and between two IT-infrastructure monitoring software.

This project was successful in building a fully functioning temperature monitoring system that can also be expanded. Building the system was documented as accurately as possible so, if necessary, the Finnish Evangelical Lutheran Mission have the opportunity to both rebuild and expand the temperature monitoring system.

Keywords	Arduino, Microcontroller, Nagios3, Linux

Sisällys

Liite 6. Nagios templates -tiedosto

Liite 7. Pikaohje järjestelmän rakentamiseen

Lyhenteet

I/O	Input/Output. Tarkoittaa liitäntöjä, jotka voivat toimia syöttöliitäntöinä sekä ulostuloina.
USB	Universal Serial Bus. Sarjaliitin.
SMTP	Simple Mail Transfer Protocol. TCP-pohjainen protokolla, jota käytetään viestien välittämiseen sähköpostipalvelimien kesken.
LTS	Long-Term Support. Pidemmän aikavälin tuki.

1 Johdanto

Insinöörityön tavoitteena on rakentaa Suomen Lähetysseura ry:lle palvelinhuonetta tarkkaileva järjestelmä. Järjestelmän on tarkoitus antaa tietoa palvelinhuoneen lämpötilasta. Mittausjärjestelmä rakennetaan Arduino-mikrokontrollerilla, johon liitetään tarvittavat osat, jotta laite voidaan rakentaa.

Toimiva lämmöntarkkailujärjestelmä tarvitsee toimiakseen Arduino-mikrokontrollerin, USB-kaapelin, lämpösensorin sekä tarvittavat johdot, jotta lämpösensori saadaan liitettyä mikrokontrolleriin. Lämmöntarkkailuun tarvitaan rakennettavan laitteen lisäksi vielä Nagios3-ohjelma, joka on IT-infrastruktuurin valvontaohjelma.

Arduino-mikrokontrolleri liitetään palvelinhuoneessa sijaitsevaan Linuxin Ubuntu-palvelimeen, jossa on asennettuna Nagios-valvontajärjestelmä. Tässä työssä Nagioksen avulla tarkkaillaan palvelinhuoneen lämpötilaa. Nagios ilmoittaa käyttäjilleen, mikäli palvelinhuoneen lämpötila nousee kriittisesti. Ilmoitus näkyy Nagioksen valvontasivuilla sekä lähettää sähköpostin, kun kriittinen lämpötilaraja on ylitetty. Nagios3-valvontaohjelmaa voidaan käyttää myös monessa muussa valvonnassa, kuten palvelimien levytilan valvonnassa.

Työ tehtiin, koska Suomen Lähetysseura ry:llä ei ollut aiemmin palvelinhuoneen lämpötilaa tarkkailevaa järjestelmää. Toimipisteellä aiemmin esiintyneen tapauksen vuoksi katsottiin tarpeelliseksi tehostaa lämpötilan tarkkailua palvelinhuoneessa.

Työssä kerrotaan myös perustietoa Arduino-mikrokontrollerista sekä Nagios-ohjelmasta. Perehdytään Arduinon ominaisuuksiin sekä teknisiin tietoihin. Nagioksen käyttötarkoituksia selvennetään sekä vertaillaan toiseen markkinoilla olevaan IT-infrastruktuurin valvontaohjelmaan nimeltä Centreon. Vertailu ohjelmien välillä muodostetaan Internetistä löytyvien tietojen avulla.

Insinöörityön tavoitteena on luoda toimiva lämmöntarkkailujärjestelmä niin, että sen voi tämän työn lukemalla jäljentää toimivana kuka tahansa. Järjestelmän rakennus kuvataan tämän työn kolmannessa luvussa. Samaisessa luvussa kuvataan mahdollisimman tarkasti myös Linux-palvelimen sekä Nagioksen asennus. Liitteeksi on lisätty niin sanottu pikaohje, joka sisältää selvärakenteisen nopean ohjeen, jonka avulla vastaavan-

laisen järjestelmän voi rakentaa. Liitteen pikaohjeessa ei eri kohtia ole selvennetty yksityiskohtaisesti. Itse työssä järjestelmän rakennus sekä asennus käydään yksityiskohtaisesti vaiheittain läpi.

2 Taustateoria

2.1 Arduino

Arduino on avoimeen lähdekoodiin perustuva elektroninen alusta, johon voi luoda vuorovaikutteisia sovellutuksia. Arduino perustuu joustaviin sekä helposti käytettäviin laitteisiin sekä ohjelmistoihin. Arduino on tarkoitettu kaikkien niiden käyttöön, jotka ovat kiinnostuneita luomaan interaktiivisia ympäristöjä tai esineitä.

Arduinon internetsivuilta löytyy paljon informaatiota aiheesta kiinnostuville. Internetsivuilla on muun muassa opettavia esimerkkikoodeja, jotka on selostettu hyvin. Koodiesimerkkeihin on lisätty kommentteina tietoa, mitä mikäkin kohta suorittaa. Arduinon internetsivut mahdollistavat siis monipuolisen itseopiskelun.

Esimerkeissä on myös kuvia, jonka pohjalta aloittelijan on helppo tehdä yksinkertaisia sovellutuksia Arduinoa hyödyntäen. Esimerkkejä löytyy yksinkertaisista LED-valon vilkutuksesta haasteellisempiin Telnet Clienteihin. [1.]

Aloittelijan on mahdollista edetä esimerkkejä hyväksi käyttäen hyväksi osaajaksi. Arduinon kanssa myös ohjelmointitaidot karttuvat, sillä Arduinon ohjelmointi on lähestulkoon C-ohjelmointikieltä. Tämän kielen opittua hahmottaa jatkossa muitakin ohjelmointikieliä hieman helpommin. Esimerkkikoodien yhteydessä olevat kuvat havainnollistavat aiheita hyvin. Kuvien ansiosta omia kytkentöjä on helppo aloittaa, sillä mallista voi katsoa, miten kytkennät kuuluu tehdä.

Arduino-mikrokontrollerialustoja löytyy useita erimallisia. Arduino-mikrokontrollerista löytyy esimerkiksi pieni ja kätevä Arduino Mini, joka on mikrokontrollereista pienin. Sen käyttömahdollisuudet ovat hieman suppeammat kuin isoimmilla alustoilla, mutta se on käytännöllinen alusta moniin tarkoituksiin. Esimerkiksi pienien sovellutuksien tekoon, joita voi olla muun muassa pienikokoiset robotit tai mittauslaitteet [2].

Tässä projektissa käytetään alustana Arduino Unoa. Arduino Uno on suosituin mikrokontrollerialusta. Se on helppokäyttöinen sekä hinta-laatusuhteeltaan mainio vaihtoehto varsinkin aloittelevalle henkilölle. Arduino Uno -mikrokontrollereita on myynnissä hyvin varustelluissa elektroniikkakaupoissa, esimerkiksi Verkkokauppa.com:ssa.

Arduino Unon valinta oli todella mainio tämän insinöörityön mikrokontrollerialustaksi. Insinöörityön tehtyäni voin suositella Arduino Unon valitsemista ensimmäiseksi Arduino-mikrokontrollerialustaksi. Arduino Unosta saatavilla olevat starter-paketit lisäävät Unon suosiota, sillä ne sisältävät valmiiksi elektroniikkakomponentteja, esimerkiksi LED-valoja sekä kytkentöjä helpottavan kytkentälevyn. Tässä työssä käytettiin Arduino Unon starter-pakettia, joka sisälsi seuraavat tarvikkeet [3]:

- Arduino-mikrokontrollerin
- USB-kaapelin
- kytkentälevyn
- liitäntäjohtoja
- resistoreja
- LED-valoja
- kondensaattoreja.

Työtä varten hankittiin myös lämpötilasensori, joka ei kuulunut edellä mainittuun pakettiin. Lämpötilasensori, jota tässä työssä käytettiin, oli RHT03 (DHT22). Starter-paketti sisältää kaiken tarvittavan, mitä aloitteleva Arduino-ohjelmoija tarvitsee.

2.1.1 Arduinon tekniset tiedot

Tekniseltä puolelta tarkasteltuna Arduino Uno ei ole parhain mikrokontrolleri, mutta sisältää tarvittavat liitännät hieman suurempienkin projektien toteuttamiseen. Arduino Uno pohjautuu ATmega328-mikrokontrolleriin. Uno voi käyttää toiminnassaan 5 – 12 voltin jännitettä, joka on mahdollisuus ottaa USB-väylästä tai ulkoisesta virtalähteestä. Unossa on 14 digitaalista I/O-liitäntää sekä 6 analogista output-liitäntää. [4.]

Tässä työssä virta Arduinoon otetaan USB-väylästä, joten ulkoista virtalähdettä ei tarvita. Tässä on hyvä huomioida se, että USB-väylän jännite ei kaikissa tapauksissa vält-

tämättä riitä Arduinon virtalähteeksi. Tuollaisessa tapauksessa ulkoista virtalähdettä on käytettävä. Ulkoista virtalähdettä on käytettävä myös silloin, mikäli Arduinoa ei siis kiinnitetä tietokoneeseen, esimerkiksi roboteissa.

Arduino Unossa on USB-väylän sekä virtapistokkeen lisäksi myös muita liittimiä sekä komponentteja. Arduinon mikrokontrolleri ATmega328:ssa on komponentteina muun muassa resistoreja eli vastuksia, jotka vastustavat ja tasoittavat Arduinossa kulkevaa tasavirtaa. Nämä ovat tietysti elektroniikassa yleisesti käytettäviä komponentteja. [5.]

Taulukko 1. Mikrokontrollerin teknisiä tietoja.

Mikrokontrolleri	ATmega328
Tulojännite (suositeltu)	7-12V
Tulojännite (rajat)	6-20V
Digitaaliset I/O liitännät	14 (joista 6 tukee PWM ulostuloa)
Analogiset sisääntuloliitännät	6
Tasavirta per I/O liitäntä	40 mA
Tasavirta 3.3V liitännässä	50 mA
Flash muisti	32 KB (ATmega328), josta bootloader käyttää 0.5 KB
SRAM	2 KB (ATmega328)
EEPROM	1 KB (ATmega328)
Kellotaajuus	16 MHz

Unon digitaalisia liitäntöjä voidaan käyttää siis sekä syöttöliitäntöinä että ulostuloliitäntöinä. Tämä tarkoittaa sitä, että Arduino voi vastaanottaa sekä lähettää tietoa digitaalisien liitäntöjen kautta. Arduino Unon digitaaliset liitännät toimivat 5 voltin jännitteellä. [4.]

Kuva 1. Arduino-mikrokontrolleri. [4]

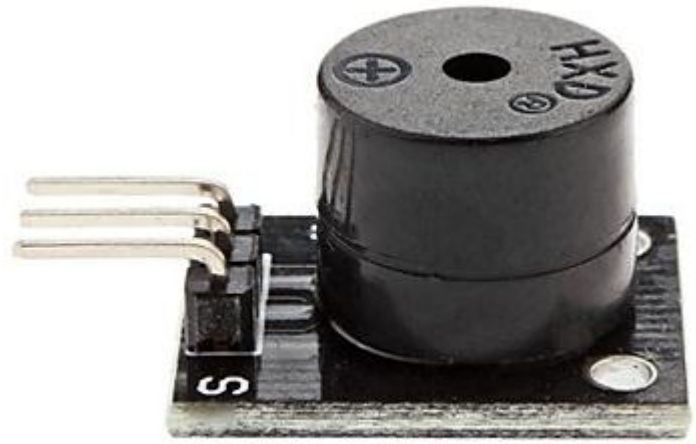
2.1.2 Osat

Arduinoon on saatavilla monenlaisia osia. Arduinon osat voidaan kategorisoida seuraavasti [6]:

- anturit
- lisävarusteet
- moduulit
- näytöt.

Arduinon antureja on tarjolla monenlaisia. Käytössä on muun muassa kiihtyvyysantureita sekä lämpötila-antureita, joissa on kosteusanturi sisällytettynä. Tuollaista lämpötila-anturia käytettiin myös tässä työssä. Arduinoon saatavien anturien avulla ympäristöä voi tarkkailla monin eri tavoin ja siitä saatua tietoa voidaan hyödyntää erilaisissa projekteissa ja sovellutuksissa.

Arduinoon saatavia moduuleja on runsas valikoima. Arduinoon voidaan liittää esimerkiksi summerimoduuli, jonka avulla Arduinosta saadaan tuotettua äänimerkkejä tai toistamaan midi-musiikkia [7].



Kuva 2. Arduinoon liitettävä summerimoduuli. [6.]

Arduinon osia on siis moniin eri tarkoituksiin. Niitä yhdistelemällä voi saada aikaiseksi mielenkiintoisia yhdistelmiä. Esimerkiksi lämpötila-anturin yhteyteen voisi liittää summerimoduulin, joka antaa äänimerkin silloin, kun kriittinen lämpötilaraja on ylitetty. Tai vastaavasti jos Arduino Unoon on kiinnitetty kiihtyvyysanturi ja summerimoduuli, voidaan siitä hyvällä todennäköisyydellä tehdä hälytin, joka hälyttää silloin, kun sitä liikutetaan.

2.2 Nagios-monitorointiohjelma

Nagios on avoimeen lähdekoodiin perustuva IT-infrastruktuurin valvontaohjelma. Nagios-valvontaohjelman avulla yritysten on helpompi seurata IT-infrastruktuurinsa toimintaa. Nagiosta voivat myös yksityishenkilöt käyttää, mikäli omien laitteiden seuranta on tarpeellista ja mielenkiintoa valvomiseen on. Valvontaohjelman ollessa toiminnassa kaikkiin ongelmatilanteisiin pystytään tarttumaan nopeammin ja tarkemmin. [8.]

Nagios on ladattavissa Nagioksen internetsivuilta osoitteessa www.nagios.org. Samoilta internetsivuilta löytyy paljon tietoa liittyen Nagiokseen. Internetsivuilla on muun muassa linkki Nagioksen Youtube -sivustolle, josta löytyy paljon mielenkiintoisia videoita liittyen Nagios IT -infrastruktuurin valvontaohjelmaan.

Nagios-ohjelman perusnäkymä on esitettynä liitteessä 1. Se on selkeä tiedonlähde IT-infrastruktuurin tarkkailemiseen. Vasemmasta laidasta löytyy valikko, josta pääsee tarkkailemaan eri palvelimien tai työasemien tilaa. Nagioksen perusnäkymä tarjoaa myös Nagioksen tietoja. Siitä näkee muun muassa asennetun Nagioksen version.

Liitteessä 3 on kuvattu Nagioksen palvelut. Palvelut-kohta listaa vasemman puoleiseen sarakkeeseen kaikki Nagioksen piiriin liitetyt laitteet, joita on muun muassa localhost- sekä remote-windows-host-liitteen tapauksessa. Laitteiden nimet määräytyvät siitä, miten ne ovat Nagioksen asetuksissa nimetty. Yleensä nimet vastaavat laitteiden oikeita nimiä (hostname). Tässä työssä Nagioksen avulla tarkkaillaan lähinnä palvelimeen liitettyä liitännäistä, joka on liitetty localhost-nimiseen laitteeseen. Localhost on yleensä se palvelin, johon Nagios on asennettu.

Nagiokseen voidaan liittää tarkastuksen alaisuuteen niin ohjelmien kuin laitteiston tietoja. Nagioksella voidaan seurata esimerkiksi palvelinten kovalevyn käyttöä.

Nagioksen toiminta on hyvin yksinkertainen. Nagios valvoo sille asetettua ympäristöä. Nagiokseen voi asettaa valvonnanalaisuuteen palvelimia sekä esimerkiksi yksittäisiä tietokoneita, joita samasta verkosta löytyy. Nagios valvoo siis ympäristöä ja heti, kun epämääräistä toimintaa ohjelman näkökulmasta ilmaantuu se ilmoittaa siitä käyttäjälleen Nagioksen hallintasivulla. [8.]

Nagios voidaan myös asettaa ilmoittamaan ongelmatilanteista sähköpostilla. Tämä voidaan konfiguroida Nagiokseen erikseen. Tämä helpottaa ongelmatilanteiden havainnointia.

Nagios-ohjelma tarjoaa monia ominaisuuksia, jotka parantavat IT-infrastruktuurin seurantaa. Hyvän seurannan takaa Nagioksen laaja monitorointi, joka sisältää muun muassa palveluiden ja verkkoprotokollien seurannan. Näitä on helppo pitää seurannassa keskitetyn näkymän ansiosta, joka kattaa koko IT-infrastruktuurin. [9.]

Nagios on muokattavissa jokaisen tarpeisiin. Avoimeen lähdekoodiin perustuva ohjelma mahdollistaa sen. Nagioksen lähdekoodi on täysin käyttäjän muokattavissa. Nagios on myös hyvin vakaa ja luotettava. Sitä on kehitetty jo yli 10 vuotta ja sitä käyttää arviolta yli miljoona käyttäjää. [9.]

Nagioksen hälytykset pitävät käyttäjät tietoisina mahdollisista häiriötilanteista. Suoraan asennettuna Nagios ilmoittaa ongelmatilanteista vain hallintasivuillaan. Nagios on mahdollista asettaa myös niin, että se lähettää tietyille henkilöille sähköpostihälytyksen, mikäli kriittisiä rajoja on ylitetty. Tässä insinöörityössä on tarkoitus asettaa Nagios ilmoittamaan sähköpostilla, mikäli lämpötila nousee liian korkeaksi tai laskee liian alhaiseksi. Tästä asennuksesta kerrotaan tämän työ asennuskohdassa lisää. Sähköpostihälytykset vähentävät ongelmatilanteisiin kuluvan reagointiajan mielestäni hyvin.

2.2.1 Nagios-liitännäiset

Nagiokseen on olemassa suuri määrä liitännäisiä. Nagioksen liitännäiset tarkoittavat Nagios3-ohjelmaan lisättyjä ominaisuuksia. Liitännäisten avulla voidaan esimerkiksi tarkastaa etätyöaseman tila, eli onko se päällä vai ei. Käytännön tasolla tuollainen esimerkkinä mainittu liitännäinen voidaan muodostaa siten, että kirjoitetaan skripti, joka suorittaa tietyin väliajoin ping-komentoa etätyöasemaan. Ping-komennon vastatessa työasema on päällä ja puolestaan, jos skripti epäonnistuu, työasema ei ole päällä tai ei ole verkossa.

Edellä mainittu esimerkki on käytännöllinen ja hyvä hyöty siinä mielessä, että se kertoo yrityksen IT-osastolle esimerkiksi palvelinten tilan. Tämän ansiosta tiedetään, onko palvelin ylhäällä. Uptime-liitännäiseksi kutsuttu liitännäinen on yksi Nagioksen perusliitännäisistä.

Nagioksen liitännäiset ovat yksittäisiä lisäyksiä Nagios-ohjelmaan. Liitännäiset ovat ohjelmia, jotka on suunniteltu niin, että Nagios suorittaa niitä. Liitännäisiä voidaan tehdä ohjelmointikielillä kuten C-kielellä tai sitten kirjoittaa suoritettavia skriptejä esimerkiksi shell-skriptejä. [10.]

Tässä työssä liitännäinen on bash-skripti, joka muodostettu tähän kyseiseen tarpeeseen. Skriptiin on määritelty muun muassa lämpötiloja, joissa Nagios hälyttää. Tarkemmat tiedot tässä työssä käytettävästä skriptistä löytyy kappaleessa 3.3.2.

2.2.2 Nagioksen vertailu

Nagios ei ole markkinoiden ainoa IT-infrastruktuurin monitorointiohjelma, mutta Nagioksen asema on hyvä. Nagioksen lisäksi saatavilla on esimerkiksi Centreon-niminen monitorointiohjelma. [11.]

Nagios eroaa muihin samanlaisiin ohjelmiin verrattuna enemmän tai vähemmän. Nagiosta on vertailtu muutamiin vastaaviin ohjelmiin, mutta vaikuttaa olevan kaikista paras. [12.]

Nagios eroaa Centreon-ohjelmasta ominaisuuksien puolella selvästi. Se, mitä Centreon-ohjelmassa ei ole mahdollista, on puolestaan Nagioksessa mahdollista. Nagioksessa on muun muassa mahdollisuus muokata käyttäjäkohtaisia ilmoituksia, jota Centreon-ohjelma ei tue. [11.]

Käyttökokemukseni mukaan voin itse suositella Nagios-valvontaohjelman käyttöä. Nagios on helppokäyttöinen sekä asentaa että ylläpitää. Saatavilla olevat monipuoliset liitännäiset lisäävät Nagios-ohjelman mielenkiintoisuutta sekä käytettävyyttä huomattavasti. Laaja verkosto auttaa myös ohjelman parissa aloittelevan henkilön kykyä selvitä ongelmatilanteista. Monilla keskustelupalstoilla riittää keskusteluja Nagiokseen liittyen, josta voi saada reilusti apua ja uusia näkökulmia asiaan.

Centreon on Nagioksen tapaan avoimeen lähdekoodiin perustuva valvontaohjelma. Centreon-ohjelmasta on ladattavissa kaksi eri versiota, vakaa- sekä kehitysversio. Centreon ohjelmasta on ladattavissa myös CES (Centreon Enterprise Server) -standardiversio. [13.]

Heidän internetsivuilta löytyy muun muassa tietoa tuotteesta ja siihen liittyvistä asioista. Sivuilla on esitetty hyvin Centreon-ohjelman ominaisuuksia. Sivuilla myös mainitaan, että Centreon on yhteensopiva Nagios3-ohjelman kanssa. Tämä tarkoittaa sitä, että Nagioksen ydintä voidaan hyödyntää Centreon ohjelmassa. Ohjelmia voidaan siis käyttää samanaikaisesti toisiaan täydentäen. [14.]

Centreon-ohjelmaan löytyy myös vaihtoehtoinen komentorivi, jonka avulla voi tehdä muutoksia Centreon-ohjelman asetuksiin. Komentorivin kautta voi tehdä muun muassa uusien laitteiden lisäykset. [14.]

Centreon tarjoaa monipuolisen käyttöliittymän, josta löytyy päivittäiset statistiikat ja hälytysten tiheydet. Centreon-ohjelman asennuksen mainitaan olevan joustava ja tämän varmasti erilaiset valmiit asetuspohjat tarjoavat. Centreon-ohjelmassa on tietojen mukaan myös hyvät mahdollisuudet rajoittaa käyttäjiltä Centreon-hallintapaneelin näkymää. [14.]

Centreon-ohjelma on laajennettavissa monilla käyttäjien tekemillä laajennuksilla. Centreon-ohjelmaan on saatavilla muun muassa karttapalvelu sekä automaattinen käyttöönottotyökalu. [14.]

Nagioksen ylivertaisuuden vertailuissa selittää varmasti sen käyttölaajuus. Nagios-valvontaohjelmaa käyttää maailmanlaajuisesti yli miljoona käyttäjää. Tämän ansiosta sille on valmiiksi kehitettyjä liitännäisiä sekä lisäosia jo saatavilla. Uusien liitännäisten kehittäminenkin käy helposti, sillä apua löytyy ja malleja on saatavilla Internetissä useita.

Valitettavasti tämän työn yhteydessä Centreon-valvontaohjelmaa ei ollut mahdollista asentaa ja testata, joten vertailu on tietojen pohjalta rakennettu. Centreon-ohjelmasta löytyy monipuolisesti tietoa heidän omilta internetsivuiltaan osoitteesta www.centreon.com.

3 Asennus

Tämän insinöörityön asennus on kolmivaiheinen. Työssä tehdään yksi fyysinen asennus ja kaksi ohjelmistopohjaista asennusta. Työ vaatii Arduino-mikrokontrollerin kytkennät, Linux-palvelimen asennuksen sekä Arduinon ohjelmoinnin.

Ensimmäinen työ, mikä asennukseen liittyy, on Arduinon ohjelmointi. Arduino ohjelmoidaan käyttämällä erillistä Arduino-ohjelmaa. Kappaleessa 3.1 kerrotaan tarkemmin Arduino-mikrokontrollerin ohjelmoinnista.

Toiseksi kerrotaan Arduino-mikrokontrollerin kytkennöistä. Kytkennät tässä työssä ovat melko helpot, mutta varaa virheisiinkin on. Täytyy varmistaa, että kytkennät tulevat

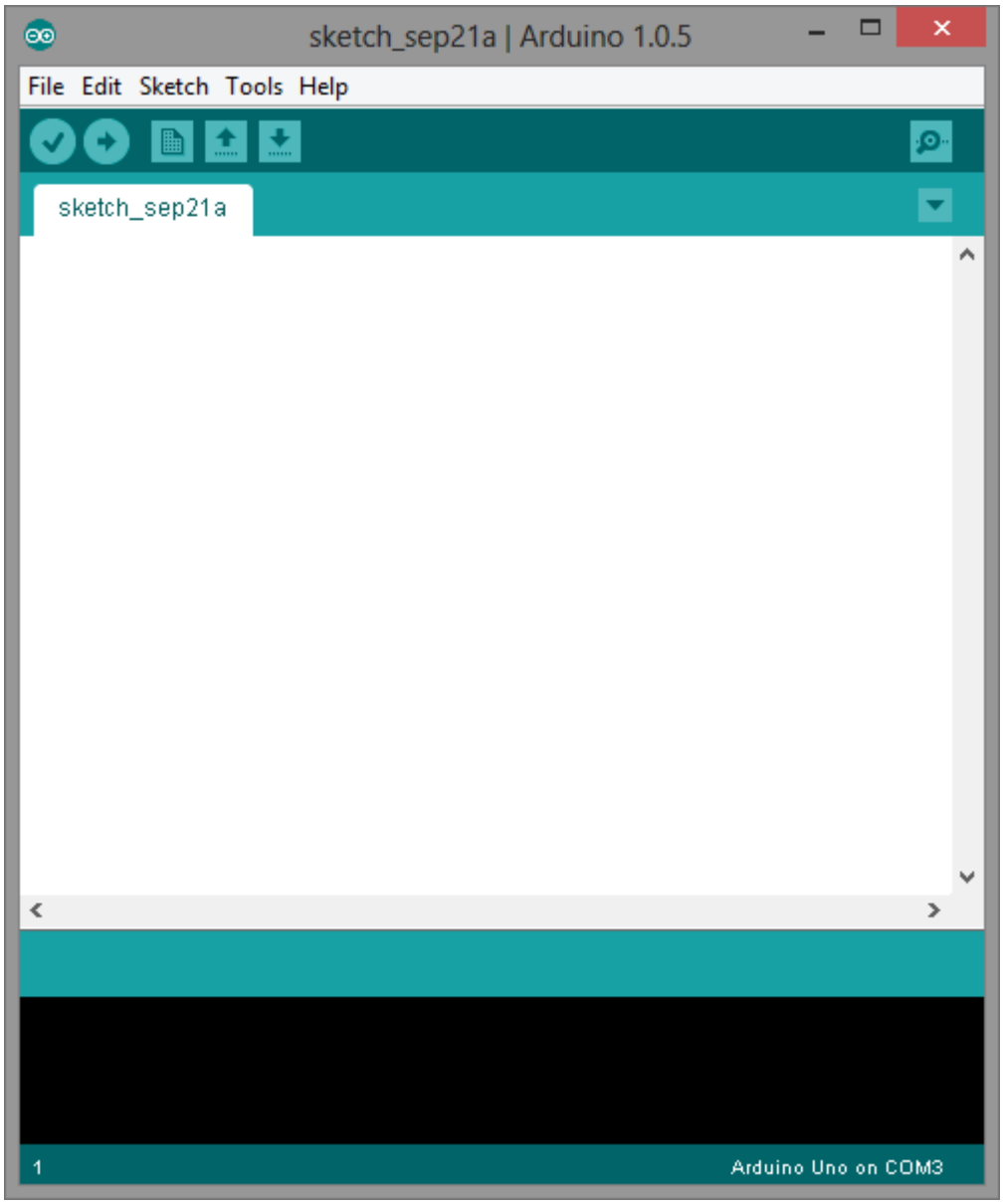
varmasti oikein. Vaarana on, että liialla jännitteellä vioittaa sensoria tai Arduino-mikrokontrollerin liitäntöjä. Tässä vaiheessa vaaditaan siis tarkkuutta, varsinkin haasteellisimmissa kytkennöissä, mutta tarkkaavaksi on hyvä opetella alusta asti.

Kytkentöjen jälkeen on vuorossa Linux-palvelimen sekä Nagios-ohjelman asennukset. Tässä työssä käytettiin Linuxin Ubuntu-palvelinta, joten sen asentaminen selostetaan. Tuolle asennetulle palvelimelle asennetaan Nagios-tarkkailuohjelma. Tämäkin asennus kuvataan myöhemmin tässä työssä.

Asennusvaiheessa on kuvattu testausympäristö. Asennusympäristönä toimiva testauslaitteisto eroaa hieman lopullisesta tuotantoympäristöstä. Erot ovat lähinnä pieniä versioeroavaisuuksia, joten tuotantoon vietäessä tämän asennusosion asennukset toimivat lähestulkoon muokkaamattomina tuotantoympäristössä.

3.1 Arduinon ohjelmointi

Mikrokontrollerialusta Arduinoa ohjelmoidaan ohjelman kautta, joka on jokaiselle käyttäjälleen ilmainen. Ohjelmaan kirjoitetaan koodit, jotka syötetään alustaan USB-portin kautta. Arduinon koodi pohjautuu C-ohjelmointikieleen. Koodi muistuttaa hyvin paljon C-ohjelmointikieltä, eikä vaadi sen osaajilta uuden ohjelmointikielen opettelua. Vasta-alkajille Arduinon ohjelmointikieli saattaa tuntua hankalalta, mutta malliesimerkkejä sekä ohjeita löytyy reilusti. Tämän takia Arduinon ohjelmointi on vasta-alkajillekin helpompaa ja mielenkiintoisempaa.



Kuva 3. Arduinon koodausohjelma.

Arduino-ohjelman voi ladata maksutta Arduinon internetsivujen kautta osoitteesta www.arduino.cc/en/Main/Software. Ohjelma on saatavilla Windows-, Mac- sekä Linux-käyttöjärjestelmille. Ohjelmaan kirjoitetaan koodi, jonka ohjelma tarkistaa ennen kuin sen voi ladata Arduino-mikrokontrolleriin. Ohjelma ilmoittaa virhetilanteista selkeästi, mikä helpottaa käyttäjää. Mikrokontrolleriin ei pysty syöttämään virheellistä koodia. Ohjelma estää virheellisen koodin syöttämisen laitteelle. Koodin voi siirtää mikrokontrolleriin vasta, kun kaikki virheet on koodista korjattu.

Tähän työhön tarvitsi Arduinoon asentaa erillinen kirjasto. Arduino alustaa voidaan laajentaa erilaisten kirjastojen avulla. Arduinon kirjastot laajentavat ympäristöä niin kuin muillakin ohjelmistoalustoilla. Arduinon kirjastoja voi käyttäjä luoda itse omien tarpeidensa mukaisesti. Kirjastoja on valmiina Arduinossa muutamia, mutta niitä voi myös ladata. [15.]

Niin sanottuja vakiokirjastoja ovat muun muassa seuraavat

- EEPROM
- Ethernet
- Firmdata
- SD.

EEPROM-kirjasto mahdollistaa lukemisen sekä kirjoittamisen niin sanottuun pysyvään tallenteeseen. Ethernet-kirjasto puolestaan takaa sen, että Arduino voidaan yhdistää Internetiin Arduino Ethernet Shieldin avulla. Firmdata-kirjaston avulla Arduino pystyy kommunikoimaan tietokoneen ohjelmistojen kanssa käyttäen standardoitua sarjaprotokollaa. SPI-kirjaston olemassaolo sallii sen, että Arduino osaa lukea ja kirjoittaa SD-muistikorteille. [15.]

Tässä työssä ladattiin käyttöön DHT22-niminen kirjasto. Kirjasto toi mahdollisuuden käyttää mikrokontrollerin lämpötila- ja kosteusanturia. Ilman kirjaston lisäystä mikrokontrolleri ei osaa tulkita saatua dataa. Tässä työssä käytettiin vain lämpötila-arvoja, joten nuo kosteusarvot on koodeissa jätetty huomioimatta. DHT22-kirjasto löytyy Internetistä ja on kaikkien ladattavissa, esimerkiksi GitHub-nimiseltä verkkosivustolta.

Ladattu DHT22-kirjasto täytyy tallentaa oikeaan paikkaan, jotta Arduino-ohjelma osaa sen kirjastoihinsa sisällyttää. Se on tallennettava Arduino-kansiosta löytyvän libraries-kansion alle seuraavasti:

C:\Program Files (x86)\Arduino\libraries\DHT22.

Seuraavaksi esitetään Arduinon ohjelmointi vaiheittain. Arduinon koodi löytyy kokonaisena kommenttien kanssa liitteestä 2. Seuraavaksi on esitetty yksittäisiä koodiosia ja selitetty koodin toiminnallisuutta.

```
#include <DHT22.h>
```

Tuolla sisällytetään DHT22-kirjasto koodin käytettäväksi. Arduinon ohjelmoinnissa tarvitsee ensin ottaa käyttöön DHT22.h- sekä stdio.h-kirjastot. Seuraavaksi lisätään koodiin stdio.h-kirjasto.

```
#include <stdio.h>
```

Nuo edellä mainitut ilmaantuvat koodiin automaattisesti, kun kirjastot liitetään Arduinon valikosta käyttöön. Kirjastot löytyvät Arduino-ohjelman Sketch-valikosta. DHT22 kirjaston lisääminen onnistuu myös tätä kautta, mikäli sen kansio on sijoitettu oikeaan paikkaan.

Tämän jälkeen alkaa itse ohjelmointi. Ensin täytyy määrittää, mihin data liitäntään Arduino on liitetty. Tässä työssä Arduino on liitetty digitaaliliitäntään 2. Se määritetään koodiin seuraavasti:

```
#define DHT22_PIN 2
```

Tämän jälkeen koodiin lisätään vielä DHT22 instanssi. Instanssin lisäämisen jälkeen Arduino on ohjelmoitu käyttämään DHT kirjaston tuomia ominaisuuksia. Instanssin lisäämisen avulla Arduino on valmis lukemaan lämpötila-anturin tietoa. [17.]

```
DHT22 OppariDHT22(DHT22_PIN);
```

DHT-instanssin luomisen jälkeen siirrytään alustamaan Arduino. Tässä vaiheessa Arduinoon ohjelmoidaan kaikki tarvittavat toimenpiteet, jotka Arduino suorittaa vain kerran käynnistyessään. Tämän työn kohdalla tässä vaiheessa Arduino ohjelmoidaan käynnistämään sarjaportti, jota kautta tieto Arduinosta siirtyy tietokoneelle.

```
void setup(void)
{
  Serial.begin(9600);
}
```

Arduinon alustamisen ja sarjaportin käynnistämisen jälkeen siirrytään niin sanottuun toisto-kohtaan, jota Arduino toistaa aina päällä ollessaan. Tämä on tärkeä seikka Arduinon ohjelmoinnissa. Täytyy huomata, että setup-kohtaan ei ohjelmoida mitään muuta kuin ne tiedot, joita Arduinon tarvitsee suorittaa vain kerran käynnistyessään.

Koodi saattaa olla muuten oikein, mutta jos jatkuvia komentoja on kirjoitettu Arduinon setup-kohtaan, niin ne eivät valitettavasti toteudu kuin kerran. Tähän voitaisiin esimerkiksi ohjelmoida merkkivalona toimiva LED-lamppu, joka syttyy kerran ja palaa aina siihen asti, kun Arduinoa pidetään päällä.

Toisto-osio on Arduinossa void loop(void) –niminen. Toisto-osioon ohjelmoidaan siis kaikki toistettavat asiat, joita Arduino suorittaa käynnissä ollessaan. Toisto-osio on kuvattu seuraavalla sivulla.

```
void loop(void)
{
   DHT22_ERROR_t errorCode;
   errorCode = OppariDHT22.readData();
   delay(2500);

      if ( errorCode == DHT_ERROR_NONE )
         {
            //Serial.print(OppariDHT22.getTemperatureC());
            char buf[32];
            sprintf(buf, "%hi",OppariDHT22.getTemperatureCInt()/10,

abs(OppariDHT22.getTemperatureCInt()%10));
            Serial.println(buf);
         }
      else
         {
            Serial.println("False");
         }
}
```

Toisto-osioon on ohjelmoitu kaikki se, mitä Arduinon täytyy suorittaa käynnistymisen jälkeen tietyin väliajoin. Aluksi on merkattu niin sanotut muuttuja sekä viive, jonka mukaan Arduino suorittaa toisto-osiossa olevia komentoja. Muuttujaksi on kirjattu errorCode, joka saa tietonsa Arduinolta. Toisto-osion viiveeksi on valittu 2500 millisekuntia. Kyseinen aika valikoitui testaamisen perusteella. Tuolla viiveellä Arduino tulosti lämpötila-arvon lähes jokaisella kerralla, kun testaaminen suoritettiin. Testaamisen perusteella tämä oli siis luotettavin viive.

Tämän jälkeen koodiin on kirjoitettu if-ehtolauseella silmukka, jota Arduino suorittaa käynnistymisen jälkeen taukoamatta. Silmukassa tarkistetaan, että errorCode on DHT_ERROR_NONE. Tämän jälkeen lämpötilatieto tulostetaan kokonaisluvuksi muutettuna. Kaikissa muissa tapauksissa tulostetaan False. Tämä on tarkoituksen mukai-

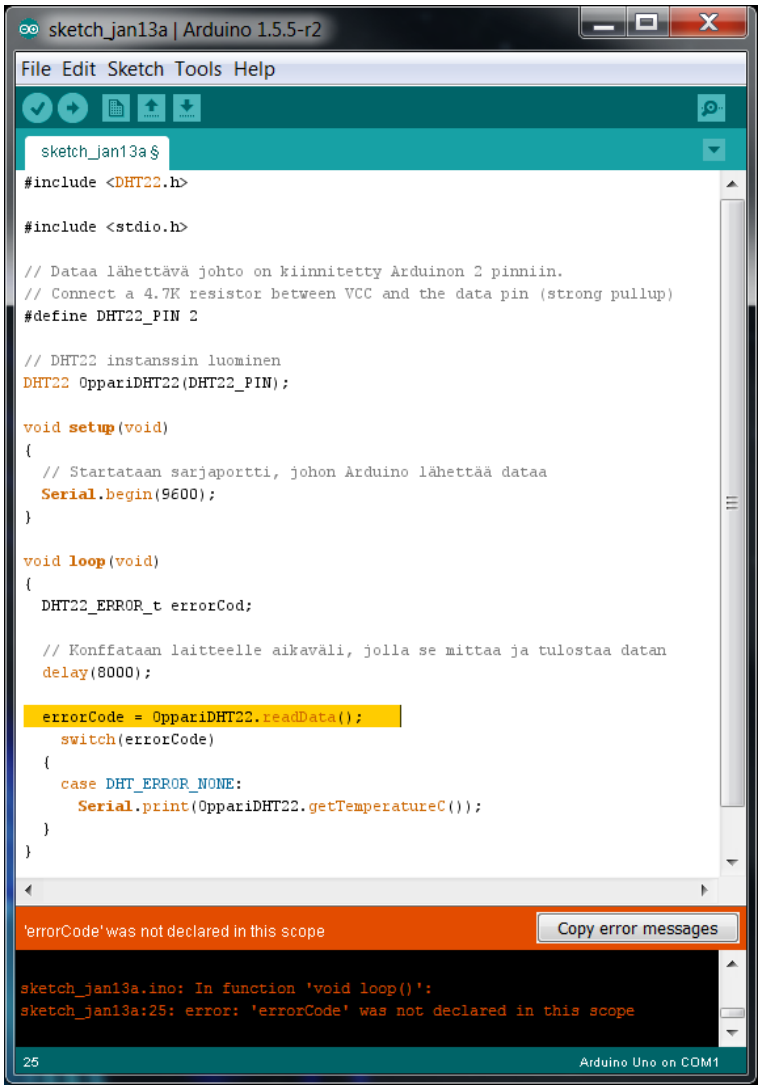
sesti pelkistetty koodi. Vikakoodeja Arduinolle on määritelty useita, mutta tässä järjestelmässä huomioidaan vain se, tulostuuko lämpötilatieto vai ei.

Arduinon ja lämpötilasensorin muita vikasietotiloja ei käytetä sen takia, että niillä ei tulisi olemaan käyttöä. Nagios-monitorointiohjelma huolehtii siitä, että hälytykset saavuttavat käyttäjän mikäli lämpötilatietoa ei saada tai se ylittää kriittiset raja-arvot. Tämän seikan takia Arduinon omia vikatietoja ei tarvitse tulostaa. Vikatiedot liittyvät usein siihen, että Arduino yrittää saada tietoa liian nopeasti tai sitten lämpötila-anturissa on fyysisesti vikaa.

3.1.1 Koodin testaus

Aduino IDE tarkistaa koodin ennen kuin sitä on mahdollista syöttää mikrokontrolleriin. Tästä syystä koodin toimivuuden tarkistus on helppoa. Tässä vaiheessa on hyvä tehdä koodista mahdollisimman helppolukuinen sekä toimiva.

Tällä tavoin koodia on helppo myös toisen tulkita. Ongelmatilanteissakin on helppo toimia, kun koodi on mahdollisimman yksinkertaisesti tehty. Ainakin yksinkertaisissa sovellutuksissa tämä on mahdollista tehdä.



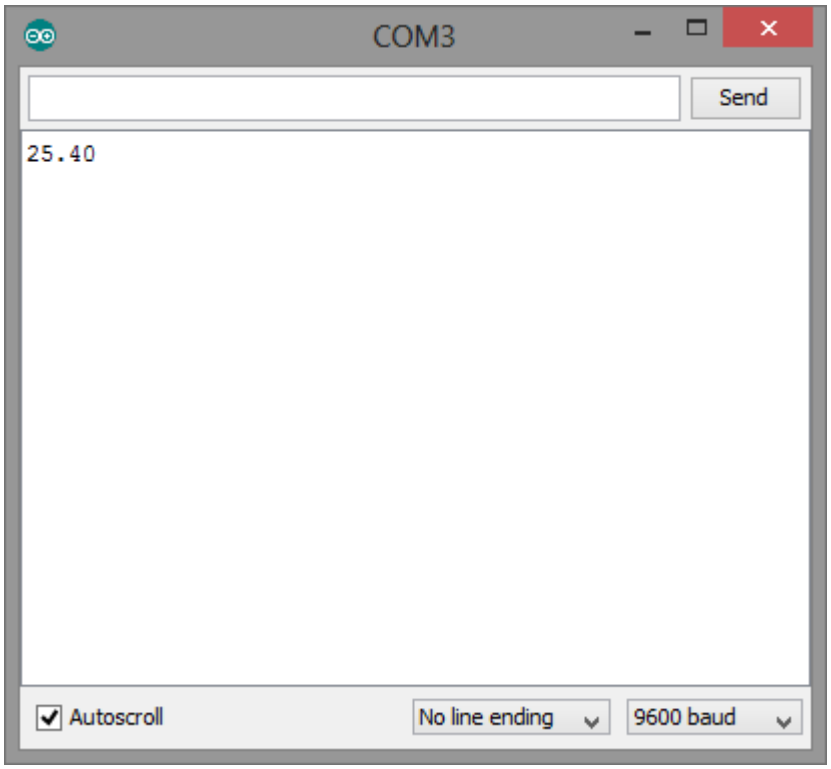
Kuva 4. Virhe koodissa.

Tässä vaiheessa mahdolliset kirjoitusvirheet tulevat esille. Ne voi silti olla hieman epäselviä, sillä esimerkiksi kuvassa 5 esiintyvä kirjoitusvirhe ei ole tuossa korostetussa kohdassa. Täytyy huomata, että tuohon kyseiseen kohtaan viitataan koodissa jo aiemmin, jossa virhekin jo esiintyy.

Edellä mainitun kaltaiset virheet saattavat vaivata aloittelijaa, toisaalta perehtyneemmälle henkilölle tuollainen virhe saattaa ratketa heti. Tässä se on nyt otettu esille esimerkkinä, jottei koodin kirjoittaminen jäisi tuollaiseen ongelmakohtaan kiinni.

3.1.2 Lämpötilatestaus

Arduinon ohjelmoinnin jälkeen lämpötilanmittaus täytyi tarkistaa. Arduinon saamaa tietoa tulostettiin tietokoneelle sarjaportin kautta. Arduinon dataan kuuluu lämpötila sekä huoneen kosteusprosentti. Tässä työssä käytetään vain lämpötilan tulostusta. Arduino tulostaa sarjaportista lämpötilan oletuksena kahden desimaalin tarkkuudella.



Kuva 5. Mikrokontrollerista tulostuva tieto. Kuva on otettu Arduino-sovelluksesta.

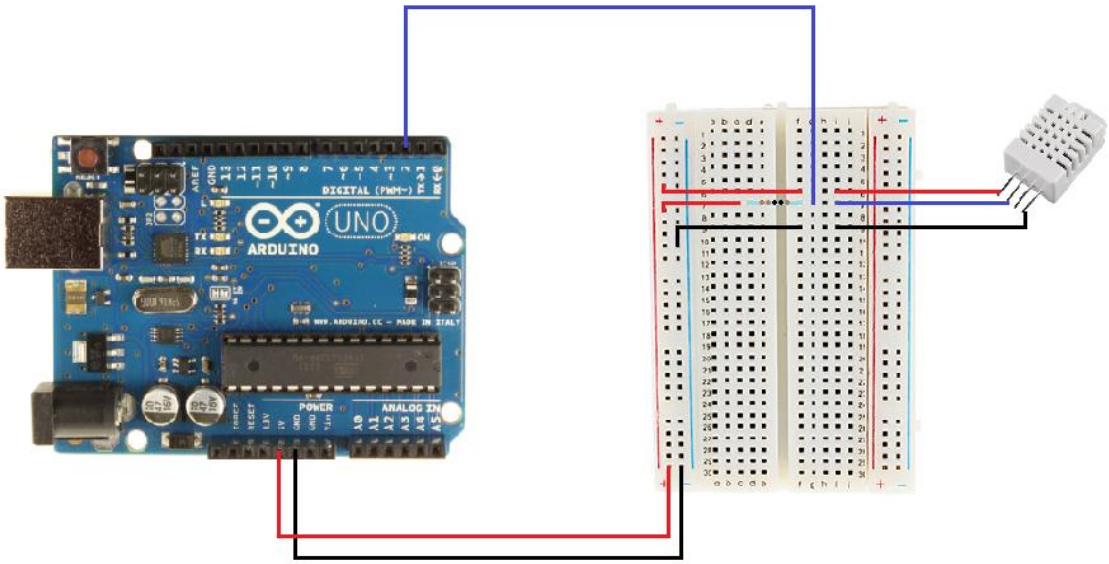
Arduinosta saatua lämpötilaa testattiin eri olosuhteissa ja sen saamat lämpötilat vastasivat toisilla lämpömittareilla saatuja tuloksi. Voidaan siis todeta, että Arduinon saama lämpötila on tähän työhön sopivan tarkka. Kuvassa kuusi esitetään lämpötila vielä kahden desimaalin tarkkuudella. Lopullisessa versiossa lämpötila esitetään ilman desimaaleja, kokonaislukuna.

3.2 Lämpötilamittausjärjestelmän rakennus

Laitteen rakentamisessa käytetään Arduino-mikrokontrolleria. Tässä työssä käytettiin nimenomaan Arduino Uno -mikrokontrolleria. Mikrokontrollerissa on monia liitäntöjä,

muun muassa analogisia sekä digitaalisia. Tässä insinöörityössä hyödynnettiin vain digitaalisia liitäntöjä.

Kytkemiseen tarvitsi mikrokontrollerin, koekytkentälevyn, lämpötila-anturin ja muutaman piuhan sekä yhden 4 700 Ohmin resistorin.



Kuva 6. Arduinon kytkentäkaavio.

Arduinon kytkentä tehtiin koekytkentälevyn avulla. Arduinosta liitettiin sähkö sekä maadoitusjohdot kiinni koekytkentälevyyn, jolloin saatiin virta kulkemaan koekytkentälevyssä. Sähköjohto on kuvattu kuvassa 3 punaisella viivalla. Maadoitusjohto on kuvattu mustalla viivalla.

Ensin on hyvä liittää maadoitusjohdot kiinni. Eli ensin liitetään Arduinon GND-liittimestä piuha kiinni koekytkentälevyn miinusliitäntään. Koekytkentälevyssä virta sekä maadoitusliitännät kulkevat pystysuoraan. A-j-liitännät koekytkentälevyssä ovat kytkettynä toisiinsa vaakasuoraan, joten jos laitetaan virtaa kulkemaan 1-a liitännästä, niin tällöin virtaa kulkee 1-e-liittimeen asti.

Tässä tapauksessa liitettiin maadoitusjohto Arduinosta koekytkentälevyn vasempaan alareunaan miinusmerkin kohdalle. Tämän jälkeen kyseisen pystysuoran liitännöistä lisättiin maadoitusjohto koekytkentälevyn liitäntään 9-f. Näin on saatu maadoitus oike-

aan kohtaan, kun lämpötila-anturin maadoitusliitäntä liitetään koekytkentälevyn liitäntään 9-i.

Tämän jälkeen kiinnitettiin lämpötila-anturi koekytkentälevyyn liitäntään 9-i. Siihen kytkettiin virta yhdistämällä koekytkentälevyyn tuotu punainen sähköjohto koekytkentälevyn liitäntään 6-f. Virtajohto yhdistetään kuvan 3 mukaisesti lämpötila-anturin vasemman laidan liittimeen. Näin ollen lämpötila-anturin vasemman laidan liitin osuu koekytkentälevyn liitäntään 6-i, joten se saa virtaa.

Lisäksi kytketään toinen virtajohto tukemaan lämpötilasensorin toimintaa. Toinen sähköjohto kiinnitetään resistorin kanssa. Sähköjohto kiinnitetään koekytkentälevyn liitäntään 7-c. Tämän jälkeen liitetään 4 700 Ω:n resistori liitännästä 7-d liitäntään 7-f. Näin estetään virtapiikkien läpikulkeutuminen.

Nyt on liitetty tarvittavat sähköjohdot Arduinon sekä koekytkentälevyn välille. Lämpötila-anturi on liitettynä koekytkentälevyn liitäntöihin 6-i → 9-i. Tämän jälkeen täytyy vielä lisätä johto Arduinon digitaalisesta liitännästä koekytkentälevyyn ja sitä kautta lämpötila-anturin data-liitäntään. Arduinon digitaaliliitännästä 2 liitetään kuvan mukaisesti sininen johto koekytkentälevyn liitäntään 7-g. Tämän jälkeen Arduino saa tietoa lämpötila-anturista, kun virrat laitetaan Arduinoon päälle.

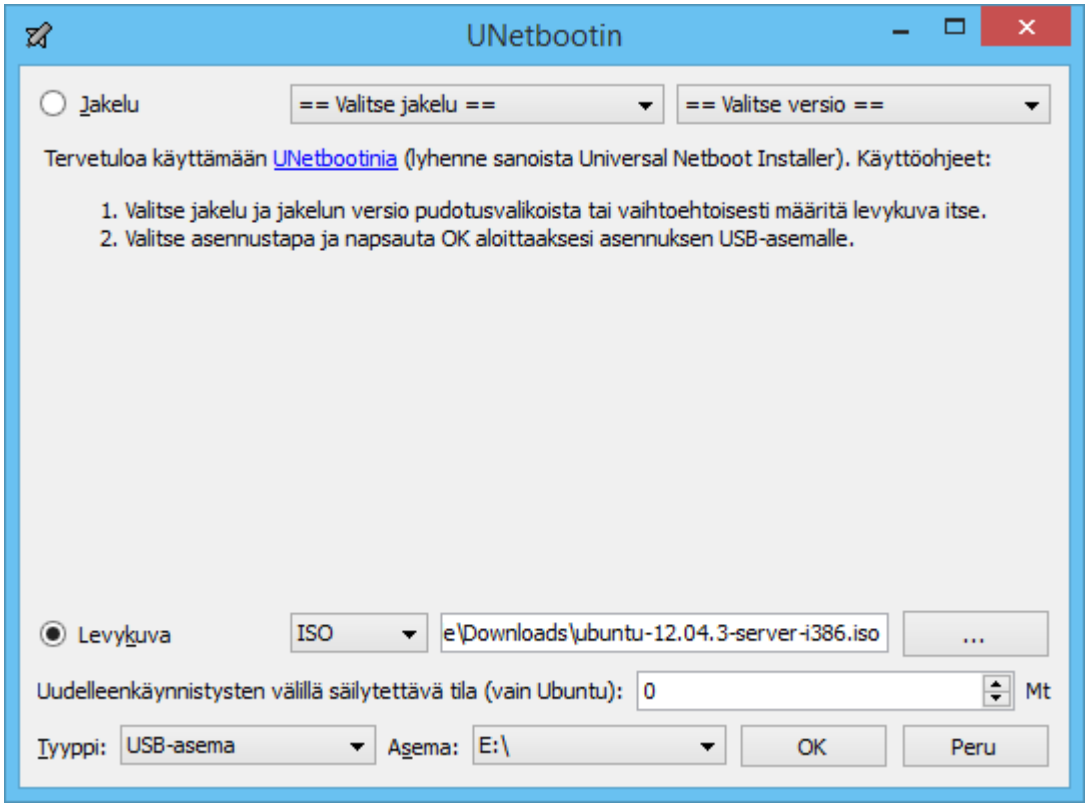
3.3 Linux-palvelimen asennus

Suomen lähetysseuralla on käytössään Ubuntu-palvelin. Tämän työn testiympäristöön valitsin Ubuntun virallisilta internetsivuilta (http://www.ubuntu.com/download/server) Ubuntu palvelimen versio 12.04.3 LTS. LTS-versiossa tukea tarjotaan pidemmälle aikavälille [18.]. Testiympäristöön valittu Ubuntu-palvelin oli 32-bittinen versio.

Ubuntu-palvelimen asennus tehtiin USB-aseman avulla. Tuosta ladatusta ISO-tiedostosta luotiin käynnistyvä USB-asema UNetbootin ohjelman avulla. UNetbootin ohjelman voi ladata maksuttomasti osoitteesta http://unetbootin.sourceforge.net/. Tässä vaiheessa on hyvä olla täysin tyhjä USB-muisti, jotta tuon ladatun ISO-tiedoston voi siihen asentaa. Ohjelma luo USB-asemalle täysin toimivan version Ubuntu-

palvelimesta, jota voi käyttää USB-asemalta tai vaihtoehtoisesti asentaa koneelleen. Tässä työssä palvelin asennettiin kannettavalle tietokoneelle.

Kuvassa 7 on esitetty UNetbootin sovellus. Tässä vaiheessa kannattaa olla tarkkana, että asentaa levykuvan oikeaan asemaan. Tässä tapauksessa USB-asema oli E-asemana, joten asennus tehtiin siihen.



Kuva 7. UNetbootin sovellus.

3.3.1 Nagioksen asennus Linux-palvelimelle

Nagios asennettiin Linux-palvelimelle. Palvelimessa käyttöjärjestelmänä on Linuxin Ubuntu 12.04, johon Nagios3-valvontaohjelma asennetaan. Nagioksen asennus tehtiin komentoriviltä, joka suoritettiin yhdellä komennolla:

```
sudo apt-get install apache2 libapache2-mod-php5 libgd2-xpm-dev nagios3
```

Linux-palvelimelle asennettiin Nagios3-ohjelma. Tuolla komennolla Nagios3-ohjelma on kuin tyhjä taulu ilmoittaen käyttäjälleen vain kyseisen palvelimen tiettyjä tietoja kuten

esimerkiksi sen, että palvelin on päällä. Nagios3-ohjelma voidaan konfiguroida tarkistamaan myös muita samassa verkossa olevia koneita. Tämä on nykyään monissa yrityksissä erittäin tärkeää, sillä yhä useampien yrityksien toiminta riippuu palvelinten sekä erilaisten työasemien toiminnasta.

Tästä syystä niitä on hyvä monitoroida reaaliajassa, jotta suuria ongelmia ei pääse syntymään. Nagios3-ohjelma voidaan määritellä tarkastamaan monia asioita niin käyttäjien työasemia kuin tärkeitä palvelimia. Usein Nagios3-ohjelmaa käytetään juuri palvelimien kunnon tarkkailuun.

Nagiokseen saadaan määritettyä useita palvelimia. Jokaisella palvelimelle voidaan asettaa yksittäisiä valvonnan kohteita, mutta myös samoja määrityksiä on mahdollisuus tehdä kerralla useammalle laitteelle.

Tässä työssä Nagioksen asennus vaati vielä lisäyksiä perusasennuksen jälkeen. Aiemmin mainittu komento (sudo apt-get install apache2 libapache2-mod-php5 libgd2-xpm-dev nagios3) siis asensi Nagioksen palvelimelle. Tässä vaiheessa tarvitsee hieman muokata Nagioksen luomia tiedostoja, jotta lämpötilan tarkkailu onnistuu. Muutoksia täytyy tehdä kahteen tiedostoon:

- /etc/nagios3/commands.cfg
- /etc/nagios3/conf.d/localhost_nagios2.cfg.

Kummankin tiedoston loppuun lisättiin muutama rivi asetuksia. Ensimmäisen tiedoston loppuun lisättiin seuraavat asetukset:

```
define command{
        command_name        Lampotilatarkistus
        command_line        /usr/skriptit/Testi
}
```

Commands.cfg-tiedosto sisältää Nagioksen käskytiedot. Sen mukaan Nagios suorittaa tarkastuksiaan. Tässä tapauksessa luodaan Nagioksen asetuksiin komento, joka suorittaa Testi-nimisen skriptin. Skripti on määritelty tarkistamaan Arduinolta tulevaa dataa. Skripti on Nagioksen liitännäinen, josta kerrotaan seuraavassa luvussa.

Tiedostoon localhost_nagios2.cfg tehdään myös loppuun muutaman rivin lisäys, jotta uusi komento saadaan toimimaan. Kyseiseen tiedostoon luodaan palvelu, joka tarkistaa komentotiedoston ja suorittaa sinne lisätyt komennot.

```
define service{
        use                     generic-service
        host_name    localhost
        service_description     LampotilaTarkistus
        check_command           Lampotilatarkistus
}
```

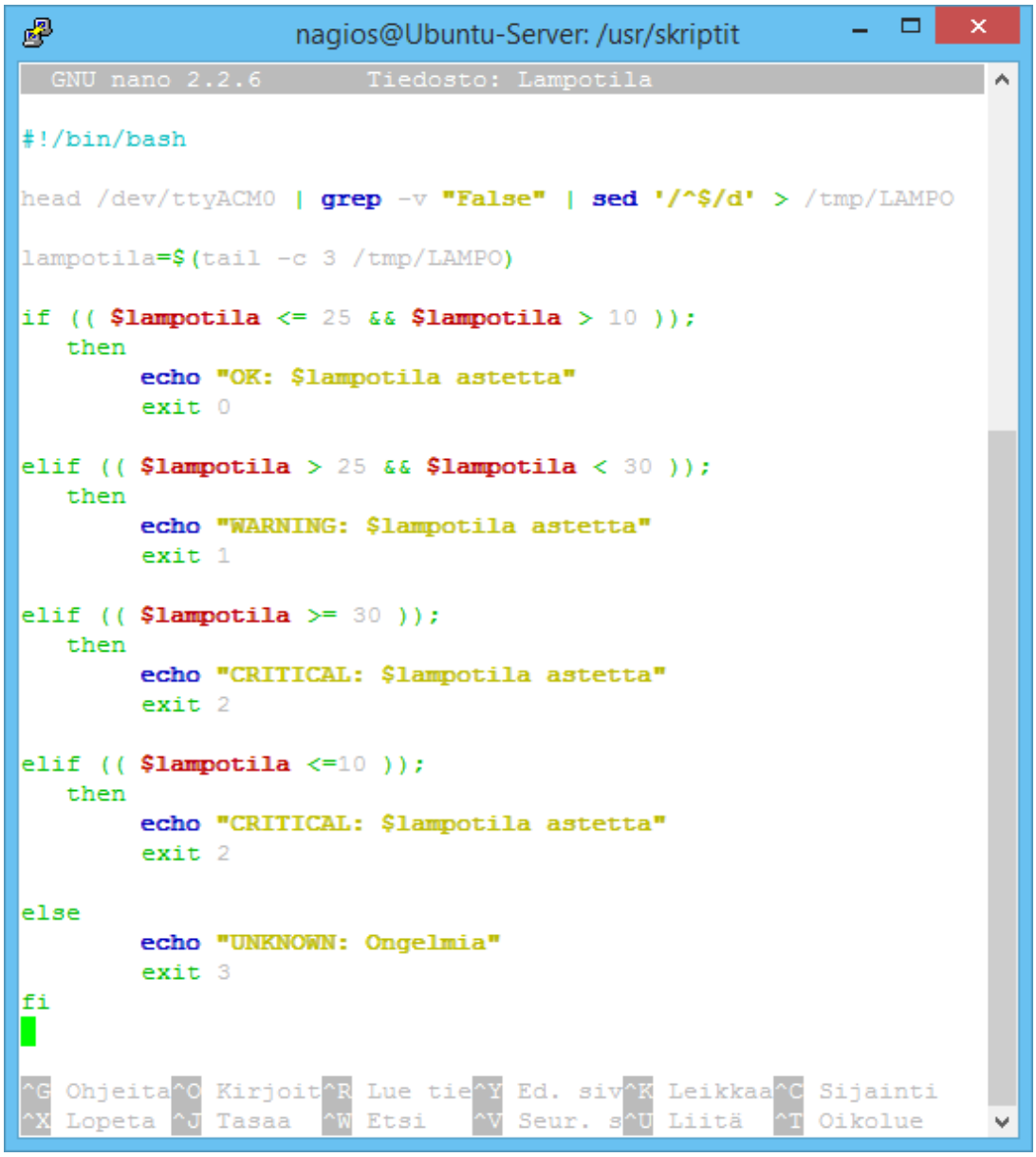
Edellä olevassa koodissa on esitetty tiedostoon localhost_nagios2.cfg lisätyt tiedot. Tässä on tärkeää huomata, että check_command-nimi vastaa täydellisesti commands.cfg-tiedoston command_name-kohtaa. Näiden tietojen täytyy olla identtisesti samat, jotta komennon suorittaminen Nagioksessa onnistuu. Linux-käyttöjärjestelmissä myös isoilla ja pienillä kirjaimilla on merkitystä.

Edellä mainituissa lisäyksissä on hyvä huomata yksi seikka. Tiedostossa commands.cfg oleva command_name on oltava samanniminen kuin tiedostossa localhost_nagios2.cfg oleva check_command. Jos nämä kaksi ovat erinimisiä, ei uusi palvelu toimi. Palvelun määritys hakee tässä tapauksessa Lampotilatarkistus-nimistä komentoa, joten on tärkeää, että tämän niminen komento siis löytyy.

3.3.2 Nagios-liitännäinen

Testiympäristössä Nagiokseen luotiin liitännäinen, joka tarkastaa lämpötilan. Liitännäisen tekeminen vaati hieman Linux skriptien osaamista. Tässä tapauksessa luotiin seuraavanlainen skripti, joka tarkastaa sarjaporttiin tulevan lämpötilalukeman ja näyttää sen valvojalle.

Lämpötilan hakemiseen ei ollut olemassa liitännäistä. Tämän takia kyseinen liitännäinen luotiin itse. Nagiokseen tehtävä liitännäinen vaatii siis skriptin tekemisen sekä hieman muokkauksia Nagioksen tiedostoihin.



Kuva 8. Skripti.

Kuvassa 8 on esitetty skripti, joka tarkistaa Arduinosta tulevan tiedon ja tulostaa sen tässä tapauksessa tiedostoon LAMPO.

Skripti hakee lämpötilatiedon polusta /dev/ttyACM0, johon Arduino on liitetty. Tämän jälkeen siitä luodaan tiedosto nimeltä LAMPO, joka tallennetaan muuttujaan nimeltä lampotila.

Tämän jälkeen skriptissä on if-ehtolause, joka tulostaa tilanteen mukaan lämpötilan sekä oikean tilailmauksen. Skriptissä olevat exit-komennot ovat tärkeässä roolissa.

Niiden avulla Nagios osaa ilmoittaa hallintasivuillaan onko lämpötila OK-, WARNING-, CRITICAL-, tai UNKNOWN-tilassa.

3.3.3 Nagios-sähköpostihälytykset

Nagios-sähköpostihälytysten lisäys vaatii sen, että postfix-palvelu on asennettuna Linux-palvelimeen. Postfix-palvelu oli tässä Linux-palvelinversiossa valmiina, mutta jos se täytyy asentaa, niin seuraavalla komennolla sen saa käyttöön: [19.]

```
sudo apt-get install postfix
```

Edellä esitetyn komennon jälkeen postfix-palvelu on käytössä. Tämän jälkeen täytyy hieman muokata postfix-palvelun asetuksia polussa /etc/postfix/main.cf. Seuraavalla sivulla on esitetty main.cf-tiedosto. Tiedostossa on kolme kohtaa, joihin pitää kiinnittää huomiota. Kohdat ovat:

- myhostname
- myorigin
- relayhost.

Näissä on huomattava, että ensimmäiseen kohtaan on laitettava käytössä olevan Linux-palvelimen nimi. Tämän jälkeen myorigin-kohtaan on hyvä laittaa myös tuo sama Linux-palvelimen nimi. Vaihtoehtoisesti siinä voi antaa olla oletuksen, joka osoittaa tiedostoon /etc/mailname. Jos tuohon kohtaan jättää oletusasetuksen on varmistettava, että tuossa tiedostossa on tietoa, jota postfix palvelu voi käyttää. Jos /etc/mailname-tiedosto on tyhjä tai sitä ei ole postfix-palvelu saattaa toimia epäluotettavasti tai ei ollenkaan. [19.]

```
#myorigin = /etc/mailname

smtpd_banner = $myhostname ESMTP $mail_name (Ubuntu)
biff = no

# appending .domain is the MUA's job.
append_dot_mydomain = no

# Uncomment the next line to generate "delayed mail" warnings
#delay_warning_time = 4h

readme_directory = no

# TLS parameters
smtpd_tls_cert_file=/etc/ssl/certs/ssl-cert-snakeoil.pem
smtpd_tls_key_file=/etc/ssl/private/ssl-cert-snakeoil.key
smtpd_use_tls=yes
smtpd_tls_session_cache_database = btree:${data_directory}/smtpd_scache
smtp_tls_session_cache_database = btree:${data_directory}/smtp_scache

# See /usr/share/doc/postfix/TLS_README.gz in the postfix-doc package for
# information on enabling SSL in the smtp client.

myhostname = Ubuntu-Server
alias_maps = hash:/etc/aliases
alias_database = hash:/etc/aliases
myorigin = Ubuntu-Server
mydestination = $myhostname, localhost.localdomain, localhost
relayhost = posti.saunalahti.fi
mynetworks = 127.0.0.0/8 [::ffff:127.0.0.0]/104 [::1]/128
mailbox_size_limit = 0
recipient_delimiter = +
inet_interfaces = all
```

Edellä esitetyn main.cf-tiedoston viimeiseen alleviivattuun kohtaan on kiinnitettävä erityisesti huomiota. Ilman relayhost-asetuksen muuttamista oikeaksi ei Nagioksen säh-

köpostihälytykset toimi. Ne eivät lähde palvelimelta eteenpäin. Tässä vaiheessa huomasin, että jos tuo kohta on tyhjänä, niin hälytyksiä ei saavu sähköpostitse. Relayhost-asetuskohtaan on laitettava internetpalveluntarjoajan lähtevän postin palvelin (SMTP-palvelin). Tässä tapauksessa palveluntarjoaja oli Saunalahti, joten SMTP-palvelin oli posti.saunalahti.fi. Tämän lisäyksen jälkeen sähköpostihälytys alkoi toimia. [20.]

Postfix-palvelun asetuksien säätämisen jälkeen on hyvä käynnistää postfix-palvelu uudelleen komennolla sudo service postfix restart. Tämä siis käynnistää postfix-palvelun uudelleen, kaikki asetusmuutokset astuvat saman tien voimaan.

Postfix-palvelun asettamisen jälkeen täytyy tehdä vielä muutoksia Nagioksen asetuksiin, jotta sähköpostihälytys saadaan siirtymään oikeaan sähköpostiosoitteeseen. Muutoksia täytyy tehdä kahteen tiedostoon:

- /etc/nagios3/conf.d/nagios2_contacts,
- /etc/nagios3/objects/templates.

Kyseisistä tiedostoista on kuvat liitteinä 5 ja 6. Ensimmäisessä tiedostossa on hyvä kiinnittää huomiota kahteen kohtaan. Ensimmäiseksi täytyy email-kohtaan kirjoittaa toimiva sähköpostiosoite, johon hälytykset halutaan. Tämän jälkeen on syytä varmistaa kontaktiryhmän nimi. Tässä tapauksessa kontaktiryhmä, jonka mukaan hälytykset lähtevät on admins. Tämä on loogisesti valittu, sillä yleensä järjestelmänvalvojat valvovat järjestelmien toimivuutta. Tässä tapauksessa he valvovat myös palvelinhuoneen lämpötilaa. [19.]

Toisessa tiedostossa varmistetaan myös edellä mainittu kontaktiryhmä. Sen täytyy vastata kontaktitiedoston kontaktiryhmää, jotta sähköpostihälytykset saapuvat perille ja oikeisiin sähköpostiosoitteisiin. [19.]

Tiedostojen muokkauksen jälkeen on hyvä käynnistää Nagios-palvelu uudelleen. Tämä suoritetaan komennolla sudo service nagios3 restart. Tämän jälkeen tehdyt muutokset astuvat heti voimaan. Tässä vaiheessa tulee myös ilmi kaikki virheet, joita tiedostoissa on, esimerkiksi kirjoitusvirheet. Palvelu ei käynnisty uudelleen, mikäli virheitä on. Tässä vaiheessa tulee myös ilmoitus siitä, missä virheitä on. Kyseinen seikka helpottaa vianetsintää huomattavasti, ja ongelmat on helppo näin ollen korjata.

Testauksen yhteydessä huomaa myös helposti, mikäli sähköpostihälytyksien yhteydessä jokin menee väärin. Mikäli testaamisen aikana jokin tarkkailtava laite tai palvelu ylittää kriittiset raja-arvot, pitäisi sähköposti-ilmoitus heti saapua määritettyyn sähköpostiosoitteeseen. Tässä vaiheessa kriittisiä virheitä ei enää ole sillä palvelu on saatu jo uudestaan käynnistettyä. Jos ongelmia ilmenee saattaa kyseessä olla kirjoitus virheitä määritetyissä asetuksissa, esimerkiksi sähköpostiosoitteessa.

Tässä on hyvä huomata myös se, että nykyään sähköpostipalvelimet saattavat tulkita Nagioksen hälytykset roskapostiksi, jolloin ne eivät välttämättä välity valvojalle. Testauksen yhteydessä huomattiin, että Windowsin live – sähköpostipalvelu siirsi Nagioksel-ta tulleet sähköpostit suoraan roskapostikansioon. Nagioksen sähköpostihälytykset saapuvat tekaistusta sähköpostiosoitteesta, joka muodostuu asetuksien kautta. Esimerkiksi tämän insinöörityön testivalvontaympäristöstä viestit saapuivat seuraavanlaisesta osoitteesta:

Asennuskappaleessa mainitut koodit sekä kytkennät ovat tuotantoon jääneet versiot. Koodeja sekä asetuksia muutettiin hieman testauksen aikana sekä tuotantoon siirryttäessä. Näistä muutoksista on kirjoitettu selvennyksiä luvuissa 4. Testaus sekä 5. Käyttöönotto.

4 Testaus

Insinöörityötä testattiin testiympäristössä ensin ennen laitteen integroimista tuotantoympäristöön. Testiympäristö rakennettiin vastaamaan oikeata tuotantoympäristöä mahdollisimman tarkasti. Testausympäristö on kuvattu tämän työn asennuskappaleessa. Testausta suoritettiin muutamien päivien ajalla ja tarkistettiin, että järjestelmä toimisi luotettavasti. Testauksen tavoitteena oli löytää järjestelmässä mahdollisesti olevat heikkoudet ja korjata ne mahdollisimman tehokkaasti.

4.1 Laitteiston testaus

Käytössä oli Linux-palvelin, jossa käyttöjärjestelmänä oli Ubuntu Server 12.04. Linux-palvelimelle asennettiin Nagios-tarkkailujärjestelmä. Testiympäristön Linux-palvelimeen

ei asennettu muita sovelluksia. Arduino-mikrokontrolleri kiinnitettiin USB-johdolla kiinni Linux-palvelimeen.

Testauksessa tarkistettiin, että Arduino-mikrokontrolleri toimii. Tarkistettiin, että Arduino käynnistyy ja siihen syötetty koodi toimii myös Linux-ympäristössä. Testauksen yhteydessä huomattiin, että Nagiokseen konfiguroitu liitännäinen ei toiminut. Testatessa Linux-palvelimella tuon luodun liitännäisen skripti toimi hyvin, joten ongelma ei löytynyt sieltä. Pitkällisen pohdinnan jälkeen huomattiin, että Nagios ei pystynyt suorittamaan skriptiä, oikeuksien puutteen vuoksi. Tässä vaiheessa skriptin käyttöoikeudet lisättiin kaikille komennolla chmod 777. Tuo komento antaa kaikille käyttäjille täydet oikeudet haluttuun tiedostoon. Tämä tehtiin vain testiympäristössä, jotta voitiin selvittää, toimiiko järjestelmä niin kuin sen pitäisi toimia.

Oikeuksien lisäyksen jälkeen todettiin, että Nagioksen liitännäinen toimii. Nagios-ohjelma kuuluu Linuxissa oletuksena Others-ryhmään. Tämän takia Nagios ei alun perin saanut haettua tietoa sarjaportista, johon Arduino tietonsa tulostaa. Tämä oli tärkeä tieto saada selville. Nyt ongelmia ei tuotantoon siirtämisessä pitäisi ilmaantua, tietäen minkälaiset oikeudet tulee olla, että Nagios-ohjelma saa tiedot Arduinolta.

Tämän jälkeen testattiin lämpötilailmoitukset Nagioksessa. Lämpötilasensoria lämmitettiin puhaltimella noin 70 celsiusasteeseen. Nagios hälytti tässä vaiheessa lämpötilasta ja ilmoitti lämpötilan olevan 72 celsiusastetta [liite 3.]. Nagioksen hälytys aiheutti myös sähköpostihälytyksen. Tässä vaiheessa testaus onnistui hyvin.

Testauksen tarkoituksena oli myös selvittää, miten Arduinosta tuleva tieto näkyy Linux-palvelimella. Tässä työssä on aiemmin mainittu, että Arduino tulostaa oletuksena kahden desimaalin tarkkuudella lämpötilan. Testauksen yhteydessä ilmeni, että desimaalit olisi hyvä jättää pois. Tässä vaiheessa Arduinon koodia muutettiin hieman. Koodiin muokattiin void loop -kohdasta. Koodissa ollut Serial.print korvattiin seuraavalla koodilla:

```
char buf[32];
sprintf(buf,"%hi",OppariDHT22.getTemperatureCInt()/10,
abs(OppariDHT22.getTemperatureCInt()%10));
Serial.println(buf);
```

Edellä esitetyssä koodin lisäyksessä on luotu ensin taulukko, johon lämpötiloja tallennetaan. Tässä tapauksessa luotiin 32-alkioinen taulukko, mutta pienempikin taulukko olisi varmasti riittänyt. Tässä työssä Arduinolla ei kuitenkaan suoriteta kovin suurta työtä, joten tämä valinta ei vaikuta kriittisesti Arduinon toimintaan. [21.]

Tämän jälkeen tulostusta muokataan sprintf-komennolla. Tässä vaiheessa tulostuvasta tiedosta muokataan desimaalit pois, jonka jälkeen jää vain kokonaisluku. Kun kokonaisluku on saatu muokattua, niin lopussa tuo taulukkoon tallennettu arvo tulostuu ulos ilman desimaaleja.

Testauksen aikana todettiin, että desimaalien poistaminen Arduinon koodissa oli hyvä ratkaisu. Tämän ansiosta palvelimelle tehtävää liitännäistä voitiin hieman yksinkertaistaa.

4.2 Huomioita testauksesta

Testaaminen on hyvä tehdä järjestelmällisesti. Ensin on hyvä määrittää, mikä on Arduinon sijainti Linuxissa. Tässä vaiheessa meni odotettua kauemmin, sillä Arduinon löytäminen laitteistokansion (/dev -kansion) alta oli haastavaa. Hetkellisen tutkailun ja Aduinon useiden kertojen liittämisen jälkeen havaitsin, että Arduino on kiinnittynyt sarjaporttiin /dev/ttyACM0. Tämä on kokeilujeni mukaan aina ollut sama.

Arduino tarvitsee virran kytkemisen jälkeen kaksi sekuntia käynnistyäkseen. Arduinoon liitetty lämpötila-anturi voi lähettää tietoa joka 1-2 sekunti, joten tämäkin aiheuttaa hieman ongelmia. Tämän takia koodiin tehtiin vaihtoehto, joka tulostaa tekstin False. Tämä eliminoitiin liitännäisen avulla pois, jonka jälkeen lämpötila saatiin tulostumaan joka kerralla.

Tässä vaiheessa Nagioksen liitännäiseen lisättiin seuraava komento:

```
head /dev/ttyACM0 | grep –v ”False”
```

Tuolla lisäyksellä saatiin virheellinen tieto poistettua lämpötilatietojen edestä. Tässä vaiheessa huomattiin myös se, että Arduinosta tulostuvassa tiedossa ilmenee tyhjiä rivinvaihtoja, jotka aiheuttavat sen, että Nagioksen hallintasivustolla lämpötilatieto ei näy. Näkyy vain tyhjä rivi. Tämä tapahtuu juuri tuon tyhjän rivinvaihdon seurauksesta sillä hallintasivusto näyttää vain yhden rivin ja lämpötilatieto on siirtynyt toiselle riville. Tässä vaiheessa liitännäistä muokattiin niin, että tyhjät rivivälit poistettiin. Lopullinen komento näyttää seuraavalta:

```
head /dev/ttyACM0 | grep –v ”False” | sed ´/^$/d´ > /tmp/LAMPO
```

Edellä mainittu koodi esittää lopullisen komennon, jonka avulla saatiin muokattua Arduinolta tuleva tieto niin, että minkäänlaista virheellistä tietoa ei tulostu.

Suurelta osin laitteen testaus onnistui hyvin. Pieniä muutoksia jouduttiin tässä vaiheessa tekemään asetuksiin ja koodeihin. Edellä mainitun seikan takia oli hyvä, että järjestelmää testattiin huolella ennen tuotantoon siirtämistä. Ilman huolellista testaamista tuotantoon siirtämisen yhteydessä olisi tullut paljon ongelmia. Tuotantoon siirtämisestä ja sen onnistumisesta kerrotaan luvussa 5.

5 Käyttöönotto

Testattu lämmöntarkkailujärjestelmä liitettiin Suomen Lähetysseuran Linux-palvelimeen. Heillä oli jo valmiiksi tuo palvelin käytössä, joten tässä vaiheessa palvelinta ei tarvinnut enää asentaa.

Linux-palvelimessa oli käytössä Puppet-ohjelma. Sitä käytetään keskitettyyn konfigurointien hallintaan [22.]. Näin ollen laitteiston liittäminen heidän palvelimeen tehtiin hieman erilaisesti, kuin tämän työn testausvaiheessa tehtiin.

Puppet sijaitsi erillisellä palvelimella, johon tässä työssä tehdyt asetukset sijoitettiin. Asetukset laitettiin Puppetilla hieman eri tavalla, mutta lopputuloksena oli se, että Pup-

petiin laitetut asetukset siirtyivät Nagios-palvelimeen ilman ongelmia. Tehdyt asetukset löytyivät nyt myös heidän Nagios-palvelimelta.

Puppet-ohjelma on hyvä, sillä sen avulla asetustiedot sijaitsevat samassa paikassa ja uusien asetusten luominen on helpompaa. Tämän avulla voidaan esimerkiksi palvelinten rikkoutuessa rakentaa täysin samanlainen palvelin toisaalle, sillä asetustiedot sijaitsevat tuolla Puppet-palvelimella.

Seuraavassa käyttöönoton vaiheessa täytyi liittää Arduino heidän Nagios-palvelimeen, jotta nähdään, toimiiko järjestelmä. Arduinon polku heidän Nagios-palvelimella oli sama eli /dev/ttyACM0. Tämä helpotti järjestelmän käyttöönottoa. Tässä vaiheessa täytyi tehdä oikeusmuutos Nagios-palvelimeen. Ryhmälle ”Others” täytyi lisätä lukuoikeudet Arduinon polkuun /dev/ttyACM0. Tämän jälkeen Nagios onnistui lukemaan kyseistä porttia, ja luku ilmestyi Nagios-ohjelmaan.

Muokkausoikeudet tuohon porttiin annettiin vain yhdelle käyttäjälle, joka toimii järjestelmänvalvojana. Tämä tehtiin tietoturvan takia.

Tuotantoympäristön skriptiin tehtiin myös pieniä muutoksia, kun käyttöönotossa huomattiin pieniä epäkohtia. Skripti tallensi alun perin tietoja kansioon /media/LAMPO. Tämä kohta muutettiin niin, että tiedot tallennettiin väliaikaiskansioon tmp eli /tmp/LAMPO. Tämä sen takia, että tiedot joita Arduino tulostaa ovat väliaikaistietoja, joita ei tarvitse tallentaa pitkäksi aikaa. Näin ollen tmp-kansio on riittävä tähän tarkoitukseen, eikä tarvitse palvelimelle luoda turhia kansioita muualle.

Tuotantoympäristö oli siis muulta osin täysin identtinen testiympäristön kanssa. Samat asetukset syötettiin tuotantoympäristöön, kuin mitä tämän työn asennusluvussa on esitetty.

Käyttöönotto sujui kaikilta osin hyvin. Muutamia ongelmia ilmeni, mutta niihin saatiin tehtyä korjaukset käyttöönoton yhteydessä. Esiintyneet ongelmat mainittiin aiemmin ja niihin tehdyt korjaukset on myös mainittu.

6 Kehitysehdotuksia ja lopputulos

Tämän projektin lopulla ilmentyi muutamia seikkoja, joita kyseisessä laitteessa voisi vielä parantaa. Kehitysehdotukset on kerätty tähän, sillä seuraavien ideoiden toteutus olisi vaatinut enemmän resursseja, kuin käytössä oli.

Ensimmäinen kehitysidea liittyy Arduinon komponentteihin. Tässä voitaisiin lisätä Arduinoon useampia lämpötilasensoreja, joiden avulla lämpötilaa voitaisiin tarkastella laajemmalta alueelta palvelinhuoneesta. Voitaisiin tarkastella esimerkiksi lattiatason, keskitilan sekä kattoalueen lämpötilaa. Myös eri kohtia huoneesta olisi hyvä pitää tarkkailussa. Esimerkiksi eri palvelinten kohdilla olisi hyvä olla omat lämpötila-anturit, jotta voidaan tietää aina kyseisten palvelinten läheisyydessä oleva lämpötila. Ilmastointilaitteen läheisyydessä olisi myös otollinen paikka lämpötilan tarkkailulle. Näin voitaisiin huomata nopeammin, mikäli ilmastointilaitteessa ilmenee toimintavikoja eikä puhaltaisi viileää ilmaa. Tämä vaatisi muutaman lämpötilasensorin lisää sekä hieman Arduinon koodin muokkausta. Lisäksi Nagiokseen pitäisi lisätä uudet palvelut uusien lämpötila-alueiden tarkkailemiseen.

Näistä jokaiseen sensoriin voitaisiin määritellä omat lämpötilarajat sekä nimet Nagios3-valvontaohjelmaan. Näin tiedetään, missä vikaa ilmaantuisi, joka puolestaan helpottaa laitteiden tarkistamista.

Toisena kehitysehdotuksena ilmaantui datan seurantaa liittyvä graafin piirtäminen. Tämä tarkoittaa siis sitä, että Arduinon keräämästä tiedosta tulostettaisiin graafia, josta voidaan seurata lämpötilan muutoksia pitemmän aikavälin ajalta. Tämä vaatisi Nagios-järjestelmään muutaman lisäohjelmoinnin, mutta on helposti toteutettavissa. Tämä kehitysidea on Lähetysseuralle ehdotettu, ja he aikovatkin tämän kehitysidean jatkossa toteuttaa.

Lisäksi kehitysehdotuksena voisi mainita sen, että liittää Arduinoon komponentit ilman kytkentälautaa. Tämä ei välttämättä ole tarpeellinen, sillä sitten osia ei pystytä vaihtamaan niin helposti. Tämä ehdotus estäisi mahdolliset kytkentöjen irtoamiset. Esimerkiksi palvelinhuoneen järjestyksen vaihdossa kytkennät saattavat irrota. Tosin tämän työn avulla kytkennät voidaan uusia hyvin yksinkertaisesti.

Arduinoon voisi myös lisätä oman LED merkkivalon, joka olisi sijoitettu näkyvään paikkaan. Tästä voisi olla hieman hyötyä Arduinon päällä pysymisen tarkkailuun. Tämä vaatisi kyllä hieman pitkiä johtoja sekä pienoisen ohjelmamuutoksen Arduinon koodiin. Jos LED-valo sijoitettaisiin näkyvälle paikalle, niin huomattaisiin heti, mikäli Arduinon virransyöttö on katkennut. Tämä kehitysehdotus ei suoranaisesti lisää lämpötilantarkkailuun etuuksia, mutta itse Arduinon päällä pysymisen valvontaa tämä voisi helpottaa.

Lopputuloksena valmistui toimiva lämpötilantarkkailu Suomen Lähetysseura ry:lle. Palvelinsalin Linux-palvelimeen lisättiin Arduino-mikrokontrolleri, johon on liitettynä lämpötilasensori. Arduino on liitetty USB-kaapelilla palvelimeen. Lämmöntarkkailujärjestelmä sijoitettiin palvelinten läheisyyteen palvelinhuoneessa, jotta saadaan mahdollisimman tarkasti mitattua lämpötilaa, joka vallitsee juuri palvelinten kohdalla.

Järjestelmä sijoitettiin jokseenkin korkealle, sillä lämpötila on korkeimmillaan yleensä huoneen yläosissa. Näin saadaan parhain mahdollinen lämpötilatieto kyseisestä tilasta.

Arduino on koodattu niin, että se hakee sensorilta lämpötilatiedon ja välittää sen palvelimelle USB-porttiin. Palvelimella oleva Nagios-valvontaohjelma lukee tuon tiedon ja ilmoittaa sen käyttäjälle. Tieto näkyy käyttäjille Nagioksen hallintasivustolla. Mikäli lämpötila nousee kriittisten arvojen yli, Nagios-valvontaohjelma lähettää siitä automaattisesti sähköpostihälytyksen valittuihin sähköpostiosoitteisiin. Tässä tapauksessa sähköpostihälytykset välittyvät palvelinympäristön valvojalle.

Kaiken kaikkiaan projekti oli todella mielenkiintoinen. Projektin parissa tuli opiskeltua ja opittua uusia asioita ja tarkennettua vanhan asian tietämystä. Linux-palvelimen käyttö tuli varmemmaksi. Linux-palvelinta hallittiin vain komentorivin avulla. Graafista käyttöliittymää ei ollut käytössä. Tämä lisäsi tietämystä erilaisista komennoista, joita Linux-palvelinympäristössä käytetään.

Projektin parissa ohjelmoinnin sekä skriptien kirjoittamisen taito karttui myös. Arduinon ohjelmoinnissa käytettävän C-kielen opiskelua tuli lähes päivittäin, kun mielenkiinto tämän projektin myötä ohjelmointia kohtaan kasvoi. Insinöörityöhön tehdyn liitännäisen ansiosta skriptin luomista sai harjoitella. Skriptin tekemisen yhteydessä Linuxin komennot tulivat entistä tutummiksi.

Lämmöntarkkailujärjestelmä oli insinöörityötä tehdessä kahden viikon testiajalla. Tämän ajan puitteissa ongelmia ei esiintynyt. Jatkossa vianselvitystä kyseisen laitteen parissa tekee Suomen Lähetysseura Ry:n IT-hallinnon työntekijät.

Projektin voidaan sanoa onnistuneen hyvin. Suomen Lähetysseuran IT-hallinto oli tyytyväinen projektin lopputulokseen, ja lämmöntarkkailujärjestelmä jäi heille toimintaan tuotantoympäristöön.

Lähteet

1 Arduino Learning. 2013. http://arduino.cc/en/Tutorial/HomePage. Luettu 15.01.2014.

2 Arduino. 2013. http://www.arduino.cc. Luettu 26.08.2013.

3 Arduino Uno Starter Kit. 2013. http://store.mansteri.com/index.php/fi/arduino/arduino-boards/starter-kit-arduino-uno.html. Luettu 25.11.2013.

4 Arduino Unon tekniset tiedot. 2013. http://arduino.cc/en/Main/arduinoBoardUno. Luettu 26.08.2013.

5 Arduino Datasheet. Verkkodokumentti. 2009. http://www.atmel.com/Images/doc8161.pdf. Luettu 06.02.2014.

6 Arduinon osia. 2013 http://www.miniinthebox.com/fi/arduino_c5028. Luettu 14.09.2013.

7 Arduinon moduuli summeri. 2013. http://www.miniinthebox.com/fi/arduino-summeri-moduuli_p480353.html. Luettu 14.09.2013.

8 Nagios. 2014. http://www.nagios.org/about/overview. Luettu 22.01.2014.

9 Nagioksen ominaisuuksia. 2014. http://www.nagios.org/about/features. Luettu 11.02.2014.

10 Nagios liitännäiset. 2014. http://www.nagios.org/projects/nagiosplugins. Luettu 11.02.2014.

11 Nagioksen vertailu Centreon -ohjelmaan. Verkkodokumentti. 2011. http://assets.nagios.com/datasheets/compare/How_Nagios_Compares_To_Centreon.pdf. Luettu 21.11.2013.

12 Nagios vertailu. 2013. http://exchange.nagios.org/directory/Comparisons. Luettu 21.11.2013.

13 Centreon versiot. 2013. http://www.centreon.com/Home-Download/download-home. Luettu 12.02.2014.

14 Centreon ominaisuudet. 2013. http://centreon.com/Content-Products-IT-network-monitoring/features-in-detail-centreon. Luettu 12.02.2014.

15 Nagios kirjastot. 2013. http://arduino.cc/en/Reference/Libraries. Luettu 20.11.2013.

16 C-kirjastot. 2014. http://www.cplusplus.com/reference/cstdio/. Luettu 01.02.2014.

17 DHT22. 2014. http://playground.arduino.cc/Main/DHTLib#.Uw32wPI_vy0. Luettu 26.02.2014.

18 Ubuntu. 2013. http://www.ubuntu.com/download/server. Luettu 18.01.2014.

19 Nagios sähköpostihälytykset. 2013. http://awaseroot.wordpress.com/2013/02/01/nagios-email-notifications-puppet/. Luettu 15.02.2014.

20 Saunalahden palvelimet. 2014. http://asiakastuki.saunalahti.fi/ohje/365/. Luettu 15.02.2014.

21 Sprintf komento. 2014. http://www.tutorialspoint.com/c_standard_library/c_function_sprintf.htm. Luettu 18.02.2014.

22 Puppetin käyttö. 2012. http://awaseroot.wordpress.com/2012/11/16/how-i-use-puppet-at-work/. Luettu 15.1.2014.

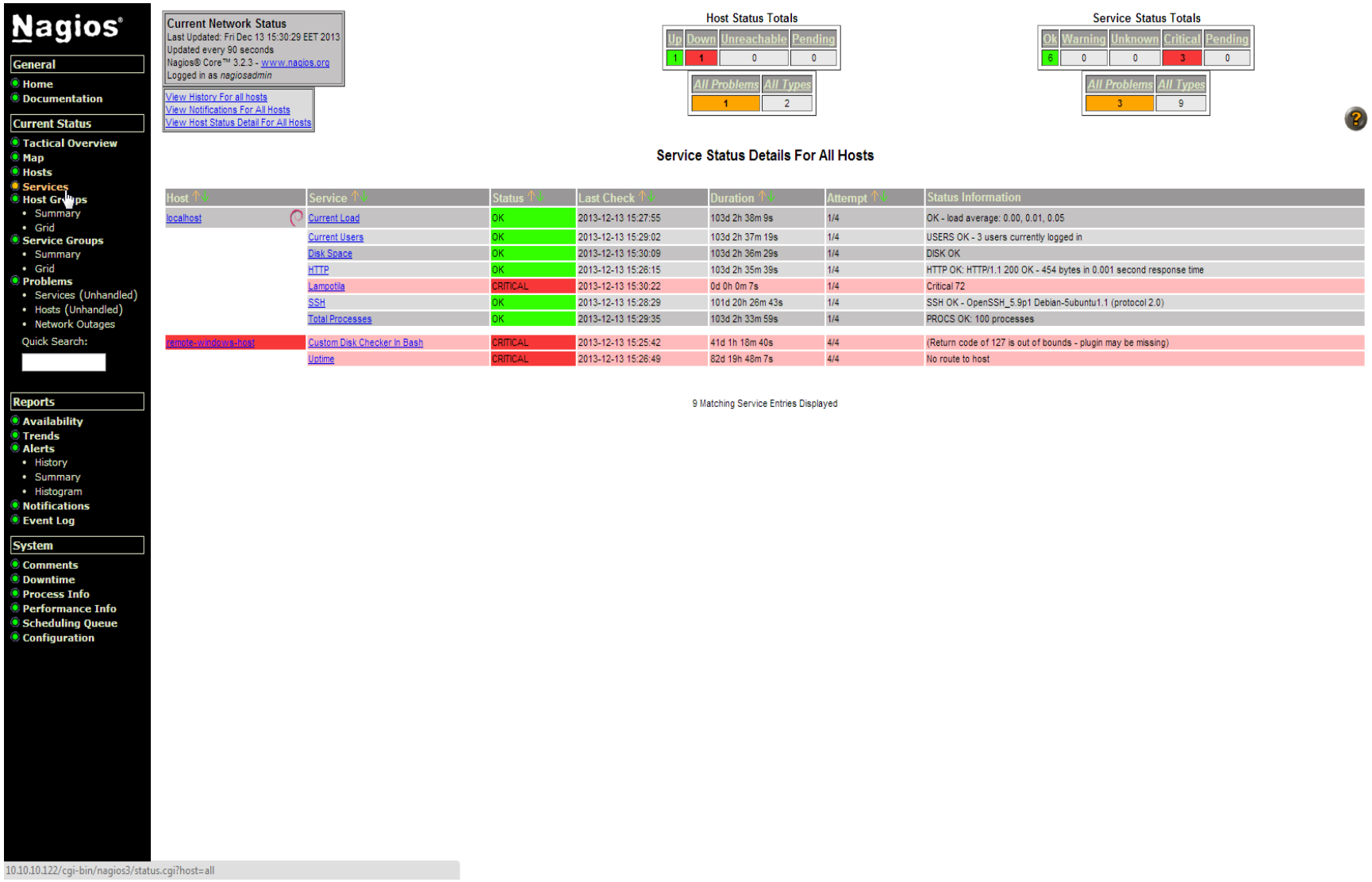
Nagios

Nagios Core
192.168.100.21/nagios3/
Nagios
General
Home
Documentation
Current Status
Tactical Overview
Map
Hosts
Services
Host Groups
Summary
Grid
Service Groups
Summary
Grid
Problems
Services (Unhandled)
Hosts (Unhandled)
Network Outages
Quick Search:
Reports
Availability
Trends
Alerts
History
Summary
Histogram
Notifications
Event Log
System
Comments
Downtime
Process Info
Performance Info
Scheduling Queue
Configuration
Nagios Core™
Nagios® Core™
Version 3.2.3
October 03, 2010
Check for updates
Read what's new in Nagios Core 3
Copyright © 2010 Nagios Core Development Team and Community Contributors.
Copyright © 1999-2009 Ethan Galstad
See the THANKS file for more information on contributors
Nagios Core is licensed under the GNU General Public License and is provided AS IS with NO WARRANTY OF ANY KIND, INCLUDING THE WARRANTY OF DESIGN, MERCHANTABILITY, AND FITNESS FOR A PARTICULAR PURPOSE. Nagios, Nagios Core and the Nagios logo are trademarks, servicemarks, registered trademarks or registered servicemarks owned by Nagios Enterprises, LLC. Usage of the Nagios marks are governed by our trademark policy
Nagios Enterprises
SOURCEFORGE.NET

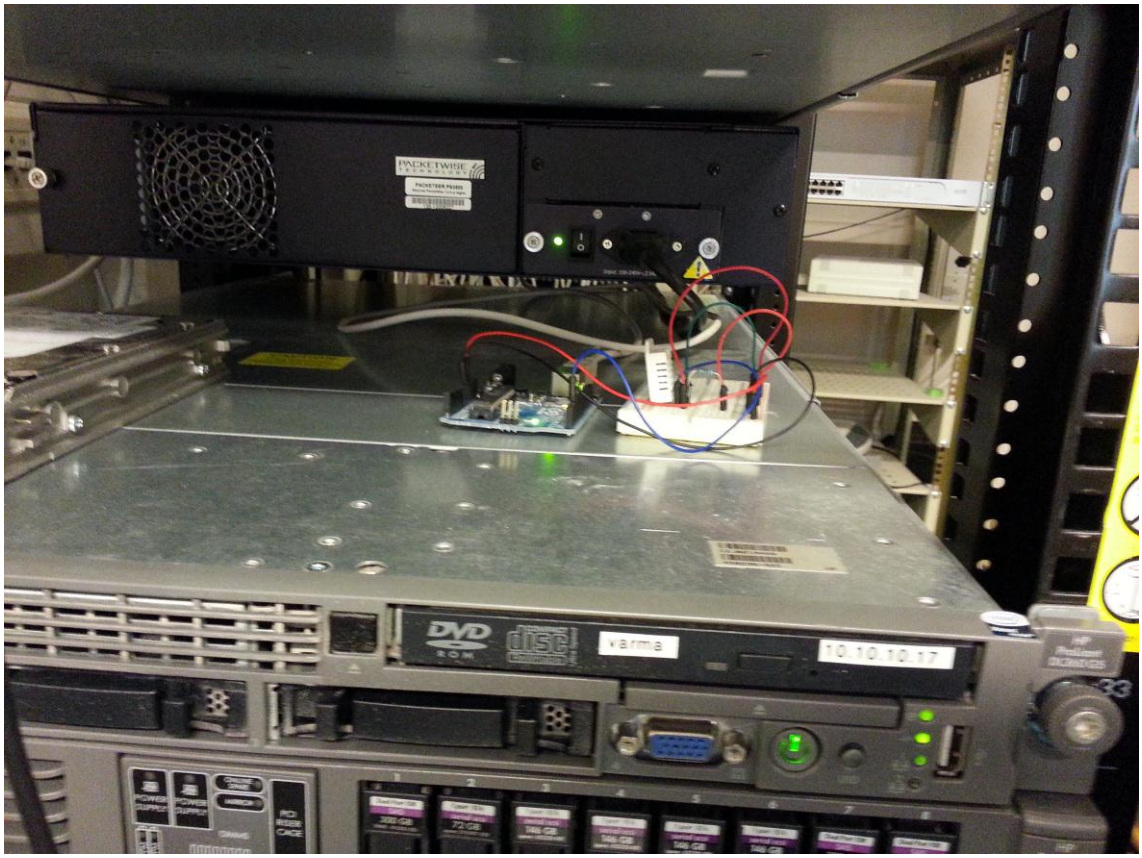
Arduinon koodi

```
#include <DHT22.h>
#include <stdio.h>
// Dataa lähettävä johto on kiinnitetty Arduinon pinniin 2.
#define DHT22_PIN 2

// DHT22 instanssin luominen
DHT22 OppariDHT22(DHT22_PIN);

void setup(void)
{
  // Startataan sarjaportti, johon Arduino lähettää dataa
  Serial.begin(9600);
}
void loop(void)
{
    DHT22_ERROR_t errorCode;
    errorCode = OppariDHT22.readData();
    delay(2500);

        if ( errorCode == DHT_ERROR_NONE )
           {
                char buf[32];
                sprintf(buf, "%hi",
                   OppariDHT22.getTemperatureCInt()/10,
                   abs(OppariDHT22.getTemperatureCInt()%10));
                Serial.println(buf);
           }
        else
           {
                Serial.print("False");
           }
}
```

Lämpötilatestaus



Arduino tuotantoympäristössä



Nagios kontaktitiedosto

```
##############################################################################
# contacts.cfg
##############################################################################

##############################################################################
#
# CONTACTS
#
##############################################################################

define contact{
        contact_name                    root
        alias                           Root
        service_notification_period     24x7
        host_notification_period        24x7
        service_notification_options    w,c
        host_notification_options
        service_notification_commands   notify-service-by-email
        host_notification_commands      notify-host-by-email
        email                           kallepa@live.fi
        }

##############################################################################
#
# CONTACT GROUPS
#
##############################################################################

define contactgroup{
        contactgroup_name               admins
        alias                           Nagios Administrators
        members                         root
        }
```

Nagios templatetiedosto

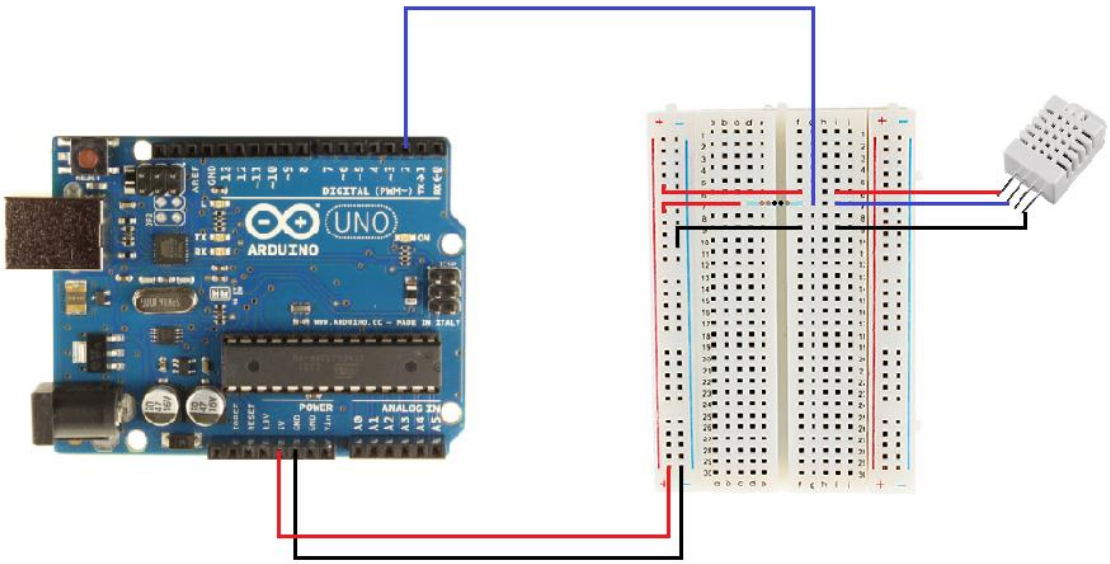
```
# Windows host definition template
define host{
  name windows-server
  use generic-host
  check_period 24x7
  check_interval 5
  retry_interval 1
  max_check_attempts 10
  check_command check-host-alive
  notification_period 24x7
  notification_interval 30
  notification_options d,r
  contact_groups admins
#  hostgroups winhosts
  register 0
}


# Linux host definition template
define host{
  name linux-box
  use generic-host
  check_period 24x7
  check_interval 5
  retry_interval 1
  max_check_attempts 10
  check_command check-host-alive
  notification_period 24x7
  notification_interval 30
  notification_options d,r
  contact_groups admins
  hostgroups linuxhosts
  register 0
}
```

Pikaohje järjestelmän rakentamiseen

1. Lataa ja asenna Arduino ohjelma osoitteesta http://arduino.cc/en/Main/Software#.UxrzZT9_s6Y
2. Lataa Arduino-DHT22 -kirjasto esimerkiksi Github-sivustolta (www.github.com) ja siirrä se Arduinon kirjasto-hakemistoon, joka on mahdollisesti: C:\Program Files (x86)\Arduino\libraries\DHT22.
3. Kirjoita seuraava koodi ohjelmaan ja aja se Arduinoon, joka on liitetty USB-kaapelilla tietokoneeseen:

```
#include <DHT22.h>
#include <stdio.h>
// Dataa lähettävä johto on kiinnitetty Arduinon pinniin 2.
#define DHT22_PIN 2
// DHT22 instanssin luominen
DHT22 OppariDHT22(DHT22_PIN);
void setup(void)
{
// Startataan sarjaportti, johon Arduino lähettää dataa
Serial.begin(9600);
}
void loop(void)
{
DHT22_ERROR_t errorCode;
errorCode = OppariDHT22.readData();
delay(2500);
if ( errorCode == DHT_ERROR_NONE )
{
char buf[32];
sprintf(buf, "%hi",
OppariDHT22.getTemperatureCInt()/10, abs(OppariDHT22.getTemperatureCInt()%10));
Serial.println(buf);
}
else
{
Serial.print("False");
}
}
```

4. Kytke Arduino oheisen kuvan mukaan:



5. Lataa ja asenna Ubuntu palvelin osoitteesta: http://www.ubuntu.com/download/server
6. Asenna Ubuntu palvelimeen Nagios3 ohjelma komennolla: sudo apt-get install apache2 libapache2-mod-php5 libgd2-xpm-dev nagios3
7. Lisää tiedostoon /etc/nagios3/commands.cfg seuraava komento:

```
define command{
        command_name            Lampotilatarkistus
        command_line            /usr/skriptit/Lampotila
}
```

8. Lisää tiedostoon /etc/nagios3/conf.d/localhost_nagios2.cfg seuraava palvelu:

```
define service{
        use                     generic-service
        host_name    localhost
        service_description     LampotilaTarkistus
        check_command           Lampotilatarkistus
}
```

9. Tee oheinen liitännäinen (skripti) valitsemaasi tiedostoon (Esim. /usr/skriptit/Lampotila):

```
#!/bin/bash

head /dev/ttyACM0 | grep -v "False" | sed '/^$/d' > /tmp/LAMPO

lampotila=$(tail -c 3 /tmp/LAMPO)

if (( $lampotila <= 25 && $lampotila > 10 ));
then
        echo "OK: $lampotila astetta"
        exit 0


elif (( $lampotila > 25 && $lampotila < 30 ));
then
        echo "WARNING: $lampotila astetta"
        exit 1

elif (( $lampotila >= 30 ));
then
        echo "CRITICAL: $lampotila astetta"
        exit 2

elif (( $lampotila <=10 ));
then
        echo "CRITICAL: $lampotila astetta"
        exit 2

else
        echo "UNKNOWN"
        exit 3
fi
```

10. Kytke Arduino USB-johdolla kiinni Linux-palvelimeen, jossa Nagios ohjelma on asennettuna.
11. Tarkista Nagios ohjelmasta osoitteessa http://localhost/nagios3 services valikosta Lampotila-palvelun tila. Localhost tarkoittaa Linux-palvelimen IP-osoitetta. Jos kaikki kunnossa pitäisi lämpötila nyt näkyä, mikäli ongelmia tarkista asetukset.
12. Sähköpostihälytyksiä varten asenna Linux-palvelimeen postfix palvelu komennolla sudo apt-get install postfix.
13. Muuta tiedostoa /etc/postfix/main.cf.
    a. Kirjoita myhostname kohtaan Linux-palvelimen nimi.
    b. Kirjoita myorigin kohtaan Linux-palvelimen nimi.
    c. Kirjoita relayhost kohtaan internetpalveluntarjoajasi SMTP-palvelimen osoite.
14. Testaa järjestelmän toiminta.