KYMENLAAKSON AMMATTIKORKEAKOULU
Ohjelmistotekniikan koulutusohjelma

Mikko Aatola

UHF/RFID-KULUNVALVONTASOVELLUS

Opinnäytetyö 2009

TIIVISTELMÄ

KYMENLAAKSON AMMATTIKORKEAKOULU
Ohjelmistotekniikka

AATOLA, MIKKO
UHF/RFID-KULUNVALVONTASOVELLUS

Opinnäytetyö 33 sivua + 1 liitesivu
Työn ohjaaja Laboratorioinsinööri Marko Oras
Toimeksiantaja VISI-RFID Solutions Oy
Marraskuu 2009
Avainsanat etätunnistus, kulunvalvonta, radiolaitteet

Etätunnistus on viime vuosina yleistynyt yhä useammilla aloilla. Erityisesti passiivisten eli ilman paristoa toimivien tunnisteiden ja niitä lukevien laitteiden teknologia on kehittynyt merkittävästi. Lukuetäisyydet ovat kasvaneet, toimintavarmuus parantunut ja tunnisteiden hinnat laskeneet. Passiivisesta tekniikasta onkin monessa sovelluskohteessa tullut varteenotettava kilpailija aktiiviselle tekniikalle.

Tämän opinnäytetyön tarkoituksena oli kehittää runko yksinkertaiselle, passiiviseen UHF-taajuusalueen RFID-teknologiaan perustuvalle kulunvalvontajärjestelmälle. Työn toimeksiantajana oli kyseiseen teknologiaan erikoistunut kotkalainen VISI-RFID Solutions Oy.

Työn alkuvaiheessa tehtiin valinta käytettävästä RFID-lukijasta yhdessä toimeksiantajan kanssa. Lukijavalinta oli jatkon kannalta tärkeä, sillä valitun lukijan ominaisuudet vaikuttivat olennaisesti järjestelmän rakenteen suunnitteluun. Kun käytettävä laitteisto oli selvillä, voitiin alkaa suunnitella järjestelmän ohjelmistopuolta. Valitusta lukijasta oli tehtävä älykäs, eli sen piti pystyä tekemään kulunvalvontapäätökset itsenäisesti ilman yhteyttä ulkopuoliseen tietokantaan. Tätä varten lukijaan kehitettiin ohjelma, joka kykenee ylläpitämään lukijan sisäisiä pääsylistoja. Lisäksi järjestelmään suunniteltiin keskustietokanta sekä hallintaohjelma.

Opinnäytetyön aikataulun puitteissa pyrittiin ensisijaisesti laatimaan suunnitelma kulunvalvontajärjestelmästä ja toisaalta toteuttamaan sen prototyyppi. Tässä onnistuttiin, vaikka toteutus jäikin osittain keskeneräiseksi. Järjestelmän toteutusta jatketaan kehitetyn suunnitelman pohjalta.

ABSTRACT

KYMENLAAKSON AMMATTIKORKEAKOULU
University of Applied Sciences
Software Engineering

AATOLA, MIKKO	UHF/RFID Access Control System
Bachelor's Thesis	33 pages + 1 page of appendices
Supervisor	Marko Oras, Laboratory Engineer
Commissioned by	VISI-RFID Solutions Oy
November 2009	
Keywords	automatic identification, access control, radio equipment

Radio frequency identification has been steadily gaining popularity in an increasing number of fields over the last few years. Especially passive technology, in which the tag itself does not contain a battery, has been developing rapidly in terms of reading distances, reliability and cost. In many applications, passive technology has already become a potent competitor of active technology.

The goal of the project discussed in this paper was to design and develop a base for a simple access control system based on passive UHF frequency band technology. The employer, VISI-RFID Solutions Oy, is a company specialised in passive RFID technology.

At the beginning of the project the decision of the reader to be used was made with the employer. At first some inexpensive readers were considered, which, however, proved to be inadequate for the project in terms of reading speed and programmability. After the decision on the equipment was made, the software part was planned. The reader had to be turned smart, meaning it had to be made capable of making access control decisions on its own. For this purpose, a program was written for the reader that could control and maintain access control lists inside the reader. In addition, a central database and a controlling program were devised.

In the time frame of the project the main goal was to devise a plan for an access control system and to create a prototype of its implementation. This goal was reached, although the prototype was left partially unfinished. The development of the system will be continued based on the plans devised.

ALKUSANAT

Työ tehtiin pääosin vuoden 2009 syksyn aikana, joskin alustavia suunnitelmia tehtiin jo aiemmin. Haluan kiittää VISI-RFID Solutions Oy:n väkeä mielenkiintoisesta ja haastavasta projektista, jonka parissa riittää työtä vielä jatkossakin.

Lisäksi haluan kiittää työn ohjaavaa opettajaa Marko Orasta, sekä koko Ohjelmisto-akatemian väkeä, opiskelutovereitani, sekä perhettäni ja kaikkia muita, jotka jaksoivat uskoa ja kannustaa niin tämän projektin kuin koko opiskelujeni loppuun asti.

Kotka, 26.11.2009

Mikko Aatola

SISÄLLYS

MERKIT, LYHENTEET JA TERMIT

HF	Radiotaajuusalue 3 – 30 MHz (High Frequency, korkea taajuus)
LF	Radiotaajuusalue 30 – 300 kHz (Low Frequency, matala taajuus)
Python	Yleiskäyttöinen, tulkattava korkean tason ohjelmointikieli, jota käytetään usein komentosarjakielenä (skriptauskielenä).
RFID	Radiotaajuinen etätunnistus (Radio Frequency Identification)
SSH	Verkkoprotokolla turvalliseen tiedonsiirtoon (Secure Shell)
TCP	Tietoliikenneprotokolla (Transmission Control Protocol)
UHF	Radiotaajuusalue 300 – 3000 MHz (Ultra High Frequency). RFID-sovelluksissa käytetyt UHF-taajuusalueet ovat 865 – 928 MHz sekä 2,45 GHz.

1 JOHDANTO

Opinnäytetyön tarkoituksena oli luoda runko yksinkertaiselle, mahdollisimman itsenäisesti toimivalle kulunvalvontasovellukselle käyttäen passiiviseen UHF/RFID-tekniikkaan perustuvaa RFID-lukijaa.

Olemassa olevista kulunvalvontasovelluksista löytyy toimivia ratkaisuja monenlaisiin käyttökohteisiin. Kuitenkin työn toimeksiantaja, VISI-RFID Solutions Oy, näki tarvetta yksinkertaiselle ja kevyelle, erityisesti passiiviseen UHF/RFID-tekniikkaan perustuvalle ratkaisulle. Aluksi suunniteltiin järjestelmän rakentamista jonkin edullisimpien hintaluokkien lukijan ympärille, mutta melko pian kävi selväksi, etteivät niiden luentanopeus, -teho ja -varmuus riitä halutunlaiseen kulunvalvontatoimintaan. Lisäksi lopulliseen ratkaisuun valitun lukijan eduksi vaikutti merkittävästi sen kehittynyt sisäinen käyttöjärjestelmä, joka mahdollistaa oman ohjelmakoodin ajamisen suoraan lukijassa.

1.1 Suunnittelu ja lukijan valinta

Järjestelmää ideoitaessa ja suunniteltaessa päädyttiin lopulta ratkaisuun, joka voidaan jakaa kolmeen erilliseen osaan: lukijalaitteessa toimivaan ohjelmaan, PC:llä käytettävään hallintaohjelmaan sekä varsinaiseen lupatietokantaan. (Liite 1, Kaaviokuva RFID-kulunvalvontajärjestelmästä.) Projektia määriteltäessä sovittiin, että tässä opinnäytetyössä keskitytään yksityiskohtaisen kulmien hiomisen sijasta suunnittelemaan kulunvalvontasovellukselle kokonaisuutena toimiva runko, jota voidaan myöhemmin laajentaa esimerkiksi käyttökohteiden asettamien vaatimusten mukaan. Jo ennen projektin aloittamista tiedostettiin, ettei järjestelmän toteutusta tulla opinnäytetyön aikana saamaan valmiiksi. Sen vuoksi keskityttiinkin panostamaan ensisijaisesti järjestelmän suunnitteluun.

Lukijalaitteeksi valittiin Siritin valmistama INfinity 510. Laitevalintaan vaikuttivat positiiviset käyttökokemukset aiemmista projekteista niin lukunopeuden ja -tarkkuuden kuin ohjelmoitavuudenkin osalta. Kuitenkin tämän projektin kannalta tärkein laitteen ominaisuuksista oli mahdollisuus suorittaa lukijassa omaa ohjelmakoodia. Näin sovellus saatiin toimimaan mahdollisimman itsenäisesti, mikä oli alusta asti eräs työn keskeisimmistä tavoitteista.

1.2 VISI-RFID Solutions Oy

Vuonna 2008 perustettu VISI-RFID Solutions Oy on erikoistunut passiiviseen UHF/RFID-teknologiaan perustuviin ratkaisuihin. Yhtiö syntyi Visi Oy:n ja Result Service Finland Oy:n yhteistyön tuloksena. Visi Oy on vuonna 1955 perustettu, videovalvontaan ja radiopuhelintekniikkaan erikoistunut yritys. Vuonna 1998 perustettu Result Service Finland Oy on tulospalveluyritys, joka on erikoistunut passiiviseen UHF/RFID-teknologiaan erityisesti tulospalvelukäytössä, ja mm. patentoinut yhdessä VTT:n kanssa kehittämänsä uudentyyppisen UHF/RFID-antennin (1).

2 RFID-TEKNOLOGIA

Radiotaajuinen etätunnistus eli RFID-tunnistus perustuu lukijalaitteen ja tunnisteiden väliseen kommunikointiin radioaaltojen välityksellä. Koska RFID-lukijat ja -tunnisteet tuottavat ja heijastavat sähkömagneettista säteilyä, ne luokitellaan radiolaitteiksi. Näin ollen ne saavat toimia vain tarkoin rajatuilla radiotaajuusalueilla ja niiden suurin sallittu säteilyteho on rajoitettu. Taajuusalueiden käyttöä ja säteilytehojen rajoja Suomessa kontrolloi Viestintävirasto (2). Eurooppalainen telealan standardisoimisjärjestö ETSI (European Telecommunications Standards Institute) on kehittänyt standardeja, joiden perusteella kunkin valtion telekommunikaatiosta vastaavat viranomaiset voivat laatia omat kansalliset säädöksensä eri taajuusalueiden käytölle. Taajuusalueiden lointi on tärkeää siksi, etteivät RFID-laitteistot aiheuta häiriötä muihin radioaaltoja hyödyntäviin laitteisiin ja sovellutuksiin, kuten radio- ja televisiolähetyksiin, matkaviestintään, teollisuuden tuotantolaitteisiin, tai viranomaisten kuten poliisin käyttämiin radioviestimiin. (3.)

Nykyisin yleisesti käytössä olevat RFID-tunnisteet jaotellaan käytetyn taajuusalueen mukaan neljään ryhmään, jotka on esitelty seuraavalla sivulla olevassa taulukossa (taulukko 1). (4.)

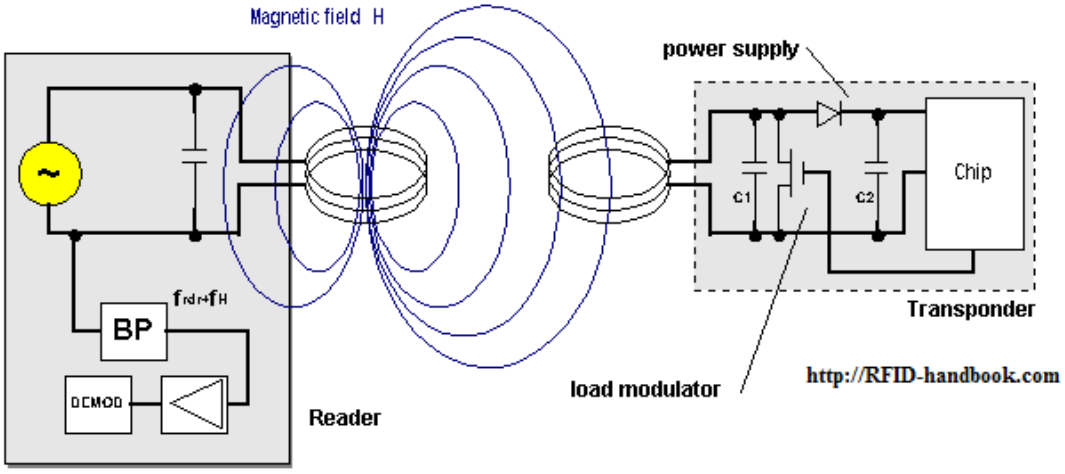
Taulukko 1. Yleisimmät RFID-taajuusalueet

Taajuusalue	RFID-taajuudet
LF (30 – 300 kHz)	125 – 134 kHz
HF (3 – 30 MHz)	13,56 MHz
UHF (300 – 3000 MHz)	865 – 928 MHz
UHF	2,45 GHz

2.1 Tunnisteet

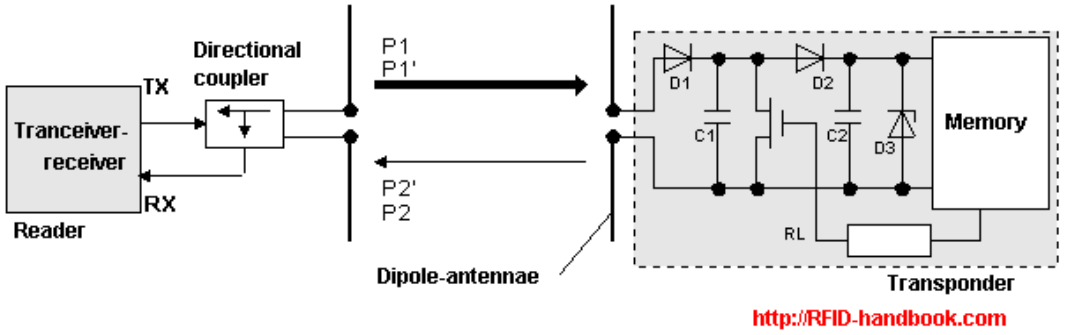
RFID-tunnisteet jaetaan yleensä rakenteensa perusteella kahteen pääryhmään: aktiivisiin ja passiivisiin. Aktiiviset tunnisteet sisältävät pariston, joten ne voivat lähettää tietoa itsenäisesti. Passiivisissa tunnisteissa ei ole omaa virtalähdettä, vaan ne joutuvat turvautumaan ulkoiseen virtalähteeseen, jotta ne voivat lähettää tietoja. Näiden lisäksi on olemassa myös pariston sisältäviä, ns. puolipassiivisia tunnisteita, jotka toimivat muuten kuin passiiviset tunnisteet eli tarvitsevat ulkoista virtaa aktivoituakseen, mutta käyttävät sen jälkeen omaa paristoaan signaalin lähettämiseen saavuttaen näin vahvemman signaalin ja pidemmän kantaman. (4.)

Passiivisten tunnisteiden toiminta voi taajuusalueen mukaan perustua kahteen erilaiseen fysikaaliseen ilmiöön. LF- ja HF-taajuusalueiden tunnisteet perustuvat sähkömagneettiseen induktioon, joka tapahtuu, kun lukija luo oskilloivan magneettikentän, johon tunniste reagoi. Lukija johtaa halutulla käyttötaajuudella (esim. HF-alueella 13,56 MHz) antenniinsa vaihtovirtaa muodostaakseen magneettikentän. Saavuttaessaan tunnisteen antennin tämä magneettikenttä aiheuttaa virran indusoitumisen, ja tämän virran avulla tunnisteen mikropiiri kykenee reagoimaan ja lähettämään tietonsa lukijalle magneettikenttää muuttamalla (kuva 1). (5.)



Kuva 1. Passiivisen induktiivisen tunnisteen toimintaperiaate (5)

UHF-taajuusalueilla toimivat passiiviset tunnisteet puolestaan perustuvat heijastumiseen ja kommunikoivat lukijan kanssa radioaaltoja välittämällä. Lukija lähettää antenninsa kautta korkeataajuisia radioaaltoja, joita tunnisteen antenni vastaanottaa ja heijastaa takaisin (kuva 2). Tunnisteessa itsessään ei ole lähetintä, vaan se sisällyttää omat tietonsa heijastamiinsa aaltoihin. Tämä tekniikka on nimeltään takaisinsironta (eng. backscatter). (5.)



Kuva 2. Passiivisen heijastavan tunnisteen toimintaperiaate (5)

Koska passiivisissa RFID-tunnisteissa itsessään ei ole erillistä virtalähdettä, ne ovat aktiivisia tunnisteita pitkäkestoisempia ja huomattavasti edullisempia. Passiivinen RFID-tekniikka on kehittynyt sekä teknisesti että hintansa puolesta viime vuosina sellaiselle tasolle, että siitä on monissa sovelluskohteissa jo varteenotettavaksi kilpailijaksi aktiiviselle tekniikalle. (6.)

UHF-tekniikan kehittyessä passiivisten tunnisteiden luentaetäisyydet ovat kasvaneet huomattavasti. Vielä muutama vuosi sitten passiivisten tunnisteiden maksimiluentaetäisyydet olivat alle 10 m, mutta nykyään voidaan tehokkaimmilla laitteilla ja laadukkaimmilla tunnisteilla saavuttaa jo lähes 30 m:n lukuetäisyys (7).

2.2 Standardit

RFID-teknologian yleistymistä pitkään hidastanut standardoinnin puute on viime vuosien aikana korjautunut merkittävästi EPCglobal-organisaation perustamisen jälkeen. Organisaatio perustettiin vuonna 2003 ratkaisemaan RFID-teknologian standardointiin liittyviä kysymyksiä. Se osoittautui menestyksekkääksi, ja vuosien 2004–2007 aikana luotiin maailmanlaajuiset EPC-standardit, joista ehkä tunnetuin on tunnisteen ja lukijan välisen tiedonsiirtoprotokollan määrittelevä Gen2-standardi. Gen2 sisällytettiin myös ISO-standardiin ISO/IEC 18000-6 Type C. (8) Nämä kansainväliset standardit takaavat niitä tukevien lukijoiden ja tunnisteiden yhteensopivuuden.

EPC (Electronic Product Code) eli sähköinen tuotekoodi on EPCglobalin hallinnoima numerointistandardi. Sitä pidetään viivakoodin seuraajana. EPC on 96-bittinen koodi, joka yksilöi tunnisteen. Viivakoodiin verrattuna EPC-koodiin mahtuu huomattavasti enemmän tietoa. Kun viivakoodilla on mahdollista tunnistaa vain valmistaja ja tuoteryhmä, EPC:n käyttö mahdollistaa myös jokaisen yksittäisen tuotteen yksilöllisen tunnistamisen. (9.)

Standardin mukaiset EPC-tunnisteet ovat maailmanlaajuisesti yksilöllisiä. Niiden käyttö kuitenkin vaatii lisenssin ostamisen EPCglobal-organisaatiolta, ja tuottaakseen lisäarvoa yritykselle, myös maksullisen jäsenyyden EPC Network -palveluun. (10.)

Gen2-standardin mukaisia tunnisteita voidaan kuitenkin käyttää myös ilman EPC-standardin mukaista numerointijärjestelmää luomalla tunnisteille oma sisäinen numerointijärjestelmä, sillä tunnisteita voidaan helposti koodata myös itse. Tämä on vielä toistaiseksi yleisin ratkaisu uusien RFID-ratkaisuja käyttöön ottavien yritysten kohdalla niin Suomessa kuin ulkomaillakin. (10: 11–12.)

2.3 Sovellusalueet

Radiotaajuisella tunnistuksella on nykyisin käyttökohteita hyvin laajalti monilla eri sovellusalueilla. Teollisuuden ja yritysten kannalta kenties tärkein sovelluskohde ovat logistiset järjestelmät. Etätunnistus helpottaa esimerkiksi tavaraliikenteen ja varastokierron seuraamista, joten sillä voidaan saavuttaa säästöjä mm. prosessien tehostumisena ja hävikkien vähenemisenä. (4.)

Myös kuluttajapuolella RFID on koko ajan yleistymässä. Sitä käytetään jo yleisesti esimerkiksi erilaisissa kulunvalvontajärjestelmissä, kirjastojärjestelmissä ja joukkoliikenteen matkakorttijärjestelmissä. Myös esimerkiksi Suomessakin käyttöön otetut, mikrosirun sisältävät biometriset passit perustuvat RFID-tekniikkaan. (4.)

Vaikka RFID ei teknologiana ole enää uusi, on sen laajempi käyttö alkanut yleistyä vasta lähivuosina. Tähän ovat osaltaan vaikuttaneet niin standardien tuoma varmuus laitteistojen yhteensopivuudesta kuin tekniikan kypsyminen ja hintojen halpeneminen. RFID-teknologian hyödyntäminen lisääntyy koko ajan ja tekniikan edelleen kehittyessä uusia sovellusalueita keksitään koko ajan lisää.

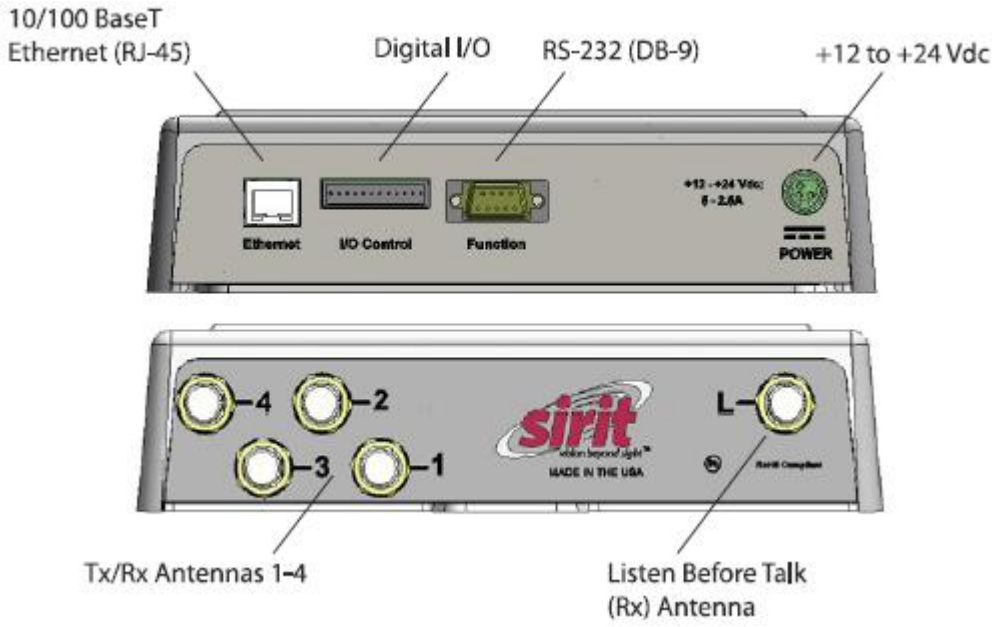
3 SIRIT INFINITY 510 UHF/RFID-LUKIJA

INfinity 510 on kanadalaisen Siritin valmistama UHF/RFID-lukija. Sirit on eräs maailman johtavista RFID-laitevalmistajista. Siritin tuotteiden Suomen maahantuojana toimii Finn-ID.

INfinity 510 tukee UHF-alueen taajuuksia väliltä 860 – 960 MHz, joten se on yhteensopiva useiden eri maiden radiotaajuuksia koskevien säädösten kanssa niin Pohjois-Amerikassa ja Euroopassa kuin Aasiassa ja Tyynenmeren alueellakin. Lukija tukee kaikkia EPCglobal Gen2 ja ISO18000-6C -standardien mukaisten protokollien pakollisia ja valinnaisia ominaisuuksia ja sisältää mahdollisuuden laiteohjelmiston päivittämiseen. Tämä mahdollistaa tulevaisuudessa myös uusien protokollien tuen lisäämisen valmistajan toimittamien ohjelmistopäivitysten muodossa. (11.)

Lukijan liitännät ovat monipuoliset (kuva 3):

- 10/100 BaseT Ethernet-verkkoliitäntä (RJ45-liitin)
- Digitaalinen I/O: 4 lähtöä ja 4 tuloa
- RS-232-sarjaportti (DB9-liitin)
- 4 antenniliitäntää (RP-TNC-liitin)
- 1 LBT (Listen Before Talk) -antenniliitäntä (RP-TNC-liitin)



Kuva 3. INfinity 510 -lukijan taka- ja etupaneelien liitännät (12)

Lukijan neljä antenniliitäntää tukevat monostaattisia antenneja, mikä tarkoittaa, että samalla antennilla voidaan sekä lähettää että vastaanottaa signaaleja. Näiden lisäksi on yksi liitin vastaanottavaa LBT-antennia varten. Asetusten määritystä, hallinnointia ja ulkoisten sovellusten yhteyksiä varten lukijassa on Ethernet-verkkoliitäntä ja sarjaportti. Lukija voi myös lähettää ja vastaanottaa signaaleja digitaalisen I/O-portin kautta. (11.)

Useimmista muista lukijoista INfinity 510:n erottaa mahdollisuus suorittaa ohjelmakoodia suoraan lukijassa. Tämä mahdollistaa lukijan toimimisen itsenäisenä ilman jatkuvasti päällä olevaa verkkoyhteyttä ja erillistä tietokonetta luetun tiedon käsittelyyn.

Projektin alkuvaiheessa, kun lukijavalinta oli tehty, sovellukselle mietittiin, ehdotettiin ja tutkittiin useita erilaisia toteutusvaihtoehtoja. Lupatietokanta piti saada vietyä ja tal-

lennettua lukijan muistiin, mutta lukijan ohjelmointirajapinnoissa ei minkäänlaisia tiedonsiirtokomentoja ollut. Esimerkiksi mahdollisuutta lupatietokantatiedoston siirtämiseen FTP-protokollaa käyttäen suoraan lukijalaitteen muistiin tutkittiin. Nämä vaihtoehdot kuitenkin jouduttiin hylkäämään, koska niiden toteuttaminen ei ollut lukijalaitteen ominaisuuksien ja valmistajan suomien mahdollisuuksien puitteissa käytännössä mahdollista.

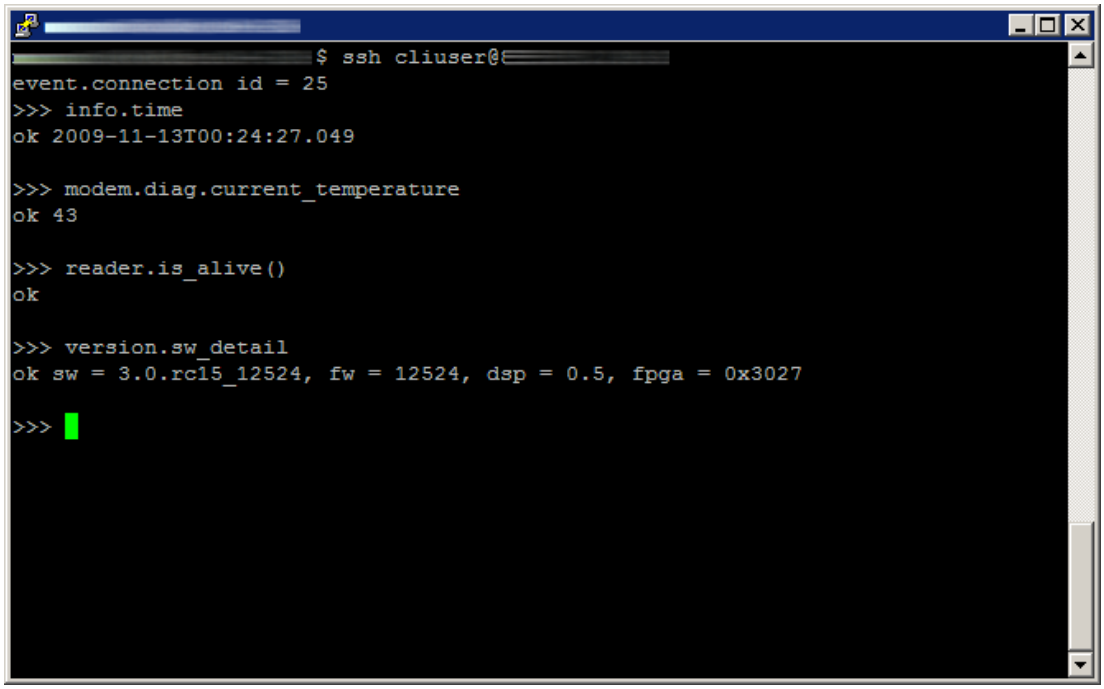
3.1 Protokollat ja rajapinnat

INfinity 510:n ensisijainen käyttötapa on liittää se osaksi Ethernet-verkkoa, jolloin siihen voidaan ottaa yhteys TCP/IP-protokollan välityksellä. Lukijan TCP-portti 50007 on kaksisuuntainen komentokanava ja portti 50008 yksisuuntainen tapahtumakanava.

Sirit käyttää omaa tekstipohjaista protokollaansa lukijan ja ohjelmistojen väliseen kommunikointiin. Lukijalle annetaan komentoja porttiin 50007 muodostettavan kaksisuuntaisen TCP-yhteyden eli ns. komentokanavan kautta. Komennot ovat selväkielistä ASCII-muotoista tekstiä, ja jokainen komento päättyy ASCII-merkkeihin #13 ja #10, eli Windows-tyyliseen rivinvaihtoon. Sirit on dokumentoinut lukijan kommunikointiprotokollan yksityiskohtaisesti kehittäjille suunnatussa Protocol Reference Guide -oppaassaan (13).

Lukijan lähettämien tapahtumien vastaanottamista varten on muodostettava yhteys lukijan porttiin 50008. Tämä ns. tapahtumakanava on yksisuuntainen, eli sen kautta ei voi lähettää komentoja, vaan vain vastaanottaa lukijalta tulevia tapahtumia.

Manuaalista komentojen syöttämistä ja tapahtumien kuuntelua varten lukijaan voidaan ottaa terminaaliyhteys verkkoyhteyden kautta SSH-protokollalla (kuva 4). Verkkoyhteyden lisäksi lukijaan voidaan avata yhteys RS232-sarjaporttin kautta. Yhteystavasta riippumatta kommunikoinnissa käytetään edellä mainittua tekstipohjaista protokollaa.



Kuva 4. SSH-yhteys INfinity 510 -lukijaan

3.2 Komento- ja tapahtumakanavat

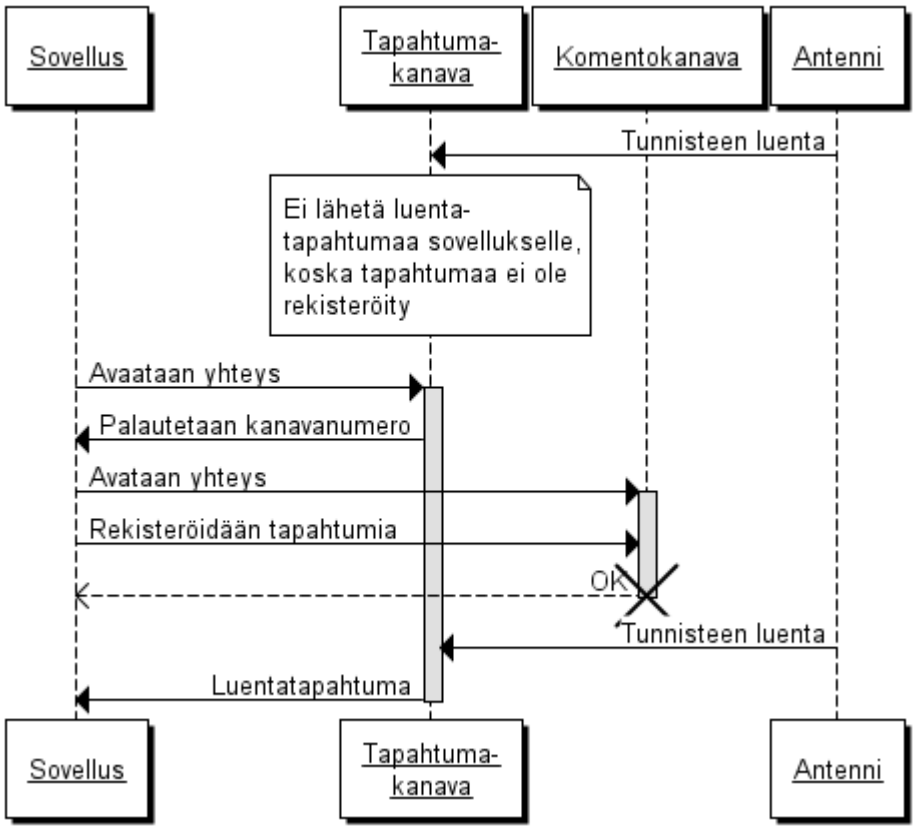
Lukija pitää sisäisesti kirjaa siihen muodostetuista yhteyksistä kanavanumeroiden avulla. Jokaiselle avatulle yhteydelle varataan oma kanavanumeronsa. Ensimmäinen kanava saa numeron 18, toinen 19 jne. Kun lukijaan muodostetaan yhteys, se varaa yhteydelle sisäisen kanavanumeron ja lähettää sen yhteyden avanneelle ohjelmalle. Kun ohjelma on vastaanottanut lukijan yhteydelle varaaman kanavanumeron, se voi käyttää kyseistä kanavanumeroa rekisteröimään kanavan vastaanottamaan tapahtumia lukijalta. Kun lukija esimerkiksi lukee tunnisteen, se lähettää tapahtuman event.tag.arrive kaikkiin niihin tapahtumakanaviin, jotka ovat rekisteröityneet kuuntelemaan tuota kyseistä tapahtumatyyppiä (ks. luku 3.3, Tapahtumapohjainen toimintamalli).

Komentoyhteydelle lukija avaa 2 kanavaa: toisen, josta se ottaa vastaan komentoja ja toisen, jonka kautta se lähettää vastaukset näihin komentoihin. Näitä kanavanumeroita ei kuitenkaan yhteyden avaavassa ohjelmassa yleensä tarvita.

Ulkoisen ohjelman ja lukijan välinen kommunikointi siis aloitetaan yleensä aina seuraavalla tavalla:

1. Avataan tapahtumakanava (TCP-yhteys lukijan porttiin 50008).
2. Luetaan lukijan lähettämä tapahtumakanavan kanavanumero (esim. 18).
3. Avataan komentokanava (TCP-yhteys lukijan porttiin 50007).
4. Rekisteröidään tapahtumakanava kuuntelemaan tarvittavia tapahtumia käyttäen kohdassa 2. saatua kanavanumeroa.

Mikäli sovelluksen ei tarvitse antaa lukijalle enempää käskyjä, vaan vain kuunnella tapahtumia ja reagoida niihin, voidaan komentokanavayhteys näiden vaiheiden jälkeen sulkea. Kuva 5 esittää em. toimenpiteet sekvenssikaaviona.



Kuva 5. Komento- ja tapahtumakanavien toimintaperiaate

3.3 Tapahtumapohjainen toimintamalli

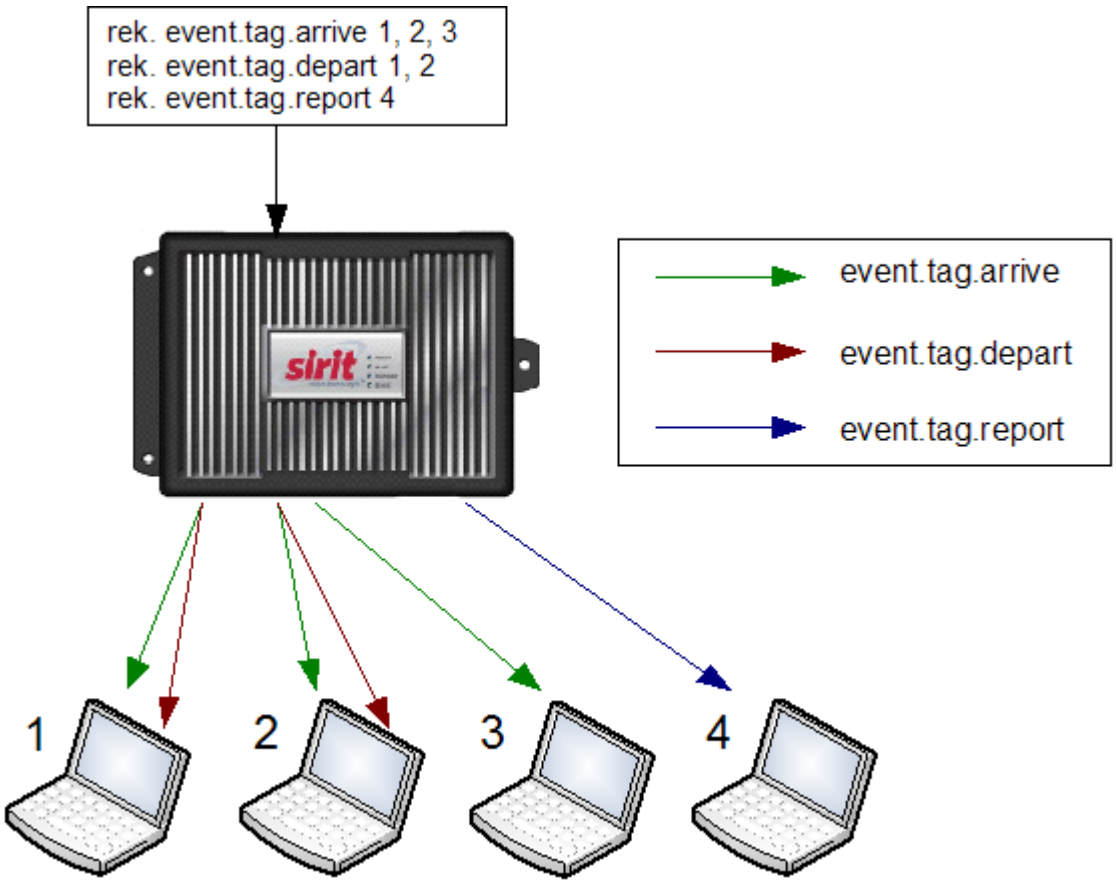
Lukija välittää tietoa ulkoisille ohjelmille lähettämällä niille tapahtumia. Jokaiselle lukijaan yhdistyneelle ohjelmalle ei lähetetä jokaista tapahtumaa, vaan jokainen ohjelma rekisteröi vain ne tapahtumat, joita sen on tarpeen vastaanottaa.

Lukijalta voidaan rekisteröityä vastaanottamaan olemassa olevien tapahtumatyyppien (esim. luentatapahtumat) lisäksi mitä tahansa mielivaltaisia tapahtumia. Toisin kuin valmiita tapahtumatyyppejä, näitä itse määritettyjä tapahtumia ei lukija missään vaiheessa laukaise automaattisesti, vaan ne täytyy laukaista komentokanavan kautta annettavalla reader.trigger_event-komennolla. Tätä ominaisuutta käytettiin hyväksi kulunvalvontaskriptin hallinnoinnissa (ks. luku 4.1.4, Pääsylistojen hallinta).

Havainnoidessaan tunnisteita useimmat lukijat yleensä lähettävät tietoa koko ajan, kun tunniste on antennin kentässä ja ulkoisen ohjelmiston tehtäväksi jää huolehtia siitä, milloin tunniste todetaan havaituksi tai poistuneeksi. INfinity 510 -lukijassa on myös kyseinen ominaisuus; ulkoinen ohjelmisto voidaan rekisteröidä kuuntelemaan event.tag.report-tapahtumia, jolloin lukija toimii juuri edellä kuvatulla tavalla.

Monista muista RFID-lukijoista poiketen Sirit INfinity 510:ssa on kuitenkin mahdollisuus jättää tunnisteiden havainnointi kokonaan lukijan hoidettavaksi. Ulkoisen ohjelmiston tarvitsee vain rekisteröityä vastaanottamaan tunnisteiden saapumis- ja poistumistapahtumia, event.tag.arrive ja event.tag.depart. Tämä on olennainen osa lukijan tapahtumapohjaista toimintamallia, joka vähentää huomattavasti lukijan kanssa kommunikoivien ohjelmistojen työmäärää.

Seuraavalla sivulla oleva esimerkkikuva (kuva 6) havainnollistaa vastaanotettavien tapahtumien rekisteröintiä yleisellä tasolla. Laitteet 1–3 rekisteröidään vastaanottamaan event.tag.arrive-tapahtumia (tunnisteen saapuminen luentakenttään), laitteet 1–2 event.tag.depart-tapahtumia (tunnisteen poistuminen kentästä) ja laite 4 event.tag.report-tapahtumia (jatkuvaa luentaa tunnisteen ollessa kentässä).



Kuva 6. Vastaanotettavien tapahtumien rekisteröinti

4 RFID-KULUNVALVONTAJÄRJESTELMÄ

RFID-kulunvalvontajärjestelmä koostuu yleensä kolmentyyppisistä pääkomponenteista: tunnisteista, lukijoista ja taustajärjestelmästä.

Kulunvalvontajärjestelmissä käytetyt lukijat voidaan jaotella niiden suorittamien toimintojen perusteella kolmeen kategoriaan: peruslukijat (ei-älykkäät), puoliälykkäät ja älykkäät. Peruslukijat hoitavat vain luennan ja lähettävät tunnistetiedot eteenpäin muille hallintalaitteille käsiteltäviksi. Puoliälykkäissä lukijoissa on tarvittavat liitännät porttien, ovien tms. aukaisemiseen, mutta ne eivät itse tee päätöstä luennan hyväksymisestä. Älykkäät lukijat puolestaan tekevät kulunvalvontapäätöksetkin itsenäisesti. Yksi tämän projektin päämääristä oli tehdä INfinity 510 -lukijasta älykäs. (14.)

Tarvittavat liitännät porttien avaamiseen lukijassa jo oli. Älykäs siitä saatiin kuitenkin vasta kehittämällä skripti, joka kykenee tekemään kulunvalvontapäätökset itsenäisesti lukijan sisältämien tietojen perusteella. Skriptin tuli näin ollen myös kyetä vastaanot-

tamaan lupatietoja ja tallentamaan niitä lukijan muistiin, sillä suoraa pääsyä lukijan sisäiseen järjestelmään laitteen valmistaja ei tarjoa.

Lukijan ja sen kulunvalvontatoiminnallisuuden ohella toinen tärkeä osa järjestelmäkokonaisuutta on taustajärjestelmä, joka tässä projektissa koostuu hallintasovelluksesta ja keskustietokannasta. Tietokanta sisältää kaiken järjestelmän käyttämän tiedon, ja hallintasovellus toimii lukijoiden ja tietokannan välillä hallinnoiden sekä suoraan tietokannan tietosisältöä että toisaalta lukijoita tietokannan tietojen mukaisesti.

4.1 Kulunvalvontaskripti

Lukijalaitteeseen ohjelmoidun skriptin toimintaperiaate on yksinkertainen: tunnisteen havaittuaan skriptin tulee tehdä päätös, hylätäänkö luenta vai hyväksytäänkö se. Mikäli tapahtuma hyväksytään, aktivoidaan lukijan digitaalilähtö määrätyksi ajaksi. Lähdön aktivoinnin seuraus puolestaan riippuu käyttötilanteesta eli käytännössä laitetason kytkennöistä. Normaalissa kulunvalvontatapauksessa lähdön aktivointi voi esimerkiksi avata portin.

Lukijassa ajettavat ohjelmat toimivat samalla periaatteella kuin ulkoisetkin ohjelmat, eli ne kommunikoivat lukijan kanssa komento- ja tapahtumakanavien kautta. Lukijassa on sekä Python- että Java-tulkit, eli siinä voidaan suorittaa näillä kielillä kirjoitettuja ohjelmia. Tämän projektin toteutuskieleksi valittiin Python lähinnä olemassa olevan osaamisen takia.

Koska lukijan käyttöjärjestelmä on Linux-pohjainen ja se sisältää Python-tulkin, voidaan sille kirjoittaa ja sillä ajaa normaaleja Python-skriptejä. Lukijoissa on valmiina joitakin valmistajan tekemiä esimerkkiskriptejä, jotka havainnollistavat, minkälaisia asioita niillä voidaan tehdä. Lisäksi niistä selviää, miten lukijan komento- ja tapahtumakanaviin avataan yhteydet ja miten niitä käytetään. Lukijassa on erityinen Python-moduuli, saturn, joka yksinkertaistaa lukijayhteyden avaamisen ja lukijan kanssa kommunikoinnin. Seuraavassa esimerkki lukijayhteyden avaamisesta ja käytöstä lukijassa ajettavassa Python-skriptissä:

```
# Avataan yhteydet
cmd = saturn.Command('localhost',50007)
evt = saturn.Event('localhost',50008)

# Luetaan tapahtumakanavan numero:
evtid = evt.getid()

# Kysytään lukijalta kellonaika
# ja tulostetaan se skriptin lokiin.
print cmd.sendcommand('info.time')

# Siirrytään tapahtumienkuuntelusilmukkaan
# (parametrina tapahtumienkäsittelijäfunktio).
evt.receive(event_handler)
```

Yllä olevassa koodiesimerkissä yhteydet avataan luomalla saturn-moduulin Command- ja Event-funktioiden avulla oliot kullekin yhteydelle. Sen jälkeen luetaan evt-olion getid()-funktiolla tapahtumakanavasta lukijan sille määrittämä tunnistenumero ja talletetaan se muuttujaan myöhempää käyttöä varten. Lukijalle lähetetään komentoja cmd-olion sendcommand()-funktiolla; esimerkkikoodissa lukijalle lähetetään komento info.time, jonka vastauksena lukija palauttaa senhetkisen ajan, joka tulostetaan skriptin lokiin Pythonin print-komennolla.

Viimeisellä rivillä olevalla evt.receive()-komennolla asetetaan skripti ns. tapahtumasilmukkaan. Tämä tarkoittaa sitä, että skripti jää odottamaan lukijan lähettämiä tapahtumia ja kun tapahtuma vastaanotetaan, suoritetaan em. komennon parametrina välitetty, itse kirjoitettu event_handler-tapahtumienkäsittelijäfunktio. Tämä funktio saa automaattisesti parametrin välityksellä lukijalta tulevan tapahtuman, jonka perusteella se päättää jatkotoimenpiteistä. Esimerkiksi tunnisteen luentatapahtumaa käsitellessään funktio tarkistaa tapahtuman parametrien perusteella oikeasta pääsylistasta, sallitaanko pääsy vai hylätäänkö luenta.

Seuraavassa koodikatkelmassa esitetään pelkistetty esimerkki tapahtumankäsittelyfunktion määrittelystä ja rakenteesta.

```
def event_handler(data):
    # Jaetaan tapahtumatiedot taulukkoon.
    event_info = re.split(' ', data)

    if event_info[0] == "event.tag.arrive":
        print time.localtime(), ":", data

    elif event_info[0] == "event.tag.depart":
        print time.localtime(), ":", data
```

Ensimmäisellä rivillä määritetään funktion nimeksi event_handler, ja sille välitettävän parametrin nimeksi data. Varsinaisen funktion ensimmäisellä koodirivillä pilkotaan re.split()-funktiolla data-muuttujan sisältö välilyöntimerkkien kohdilta osiin ja talletetaan näin syntyvä taulukko muuttujaan event_info.

Siritin lukijoiden käyttämän protokollan mukaisesti tapahtumien alkuosa ensimmäiseen välilyöntiin asti määrittää tapahtuman tyypin. Tätä tietoa käytetään hyväksi tapahtumienkäsittelijäfunktiossa, jossa tapahtuman tyyppiä verrataan if-ehtolauserakenteella vuorotellen jokaiseen tapahtumatyyppiin, jota halutaan käsitellä, kunnes oikea tapahtumatyyppi löytyy. Mikäli kyseessä on tapahtumatyyppi, jota funktiota ei ole määritetty käsittelemään, se ohitetaan.

4.1.1 Skriptin asennus ja poisto

Skriptin asentaminen tapahtuu lukijan sisäänrakennetun web-pohjaisen hallintasovelluksen kautta. Sovellukseen tulee kirjautua pääkäyttäjän tunnuksilla (admin), minkä jälkeen skriptejä voidaan hallinnoida Advanced Functions -osiossa olevalla User Application Management -sivulla (kuva 7). Tämän sivun lomakkeilla skripti siirretään lukijaan ja käynnistetään. Skriptille voidaan myös antaa käynnistysparametreja, joiden avulla voidaan tarvittaessa muuttaa skriptin oletusarvoista poikkeavia asetuksia (ks. luku 4.1.3, Käynnistysparametrit).

Kuva 7. Lukijan hallintasovelluksen User Application Management -sivu

Myös skriptin poistaminen lukijasta tapahtuu User Application Management -sivulla. Ennen poistamista on kuitenkin syytä huomioida, että käytettäessä skripti luo lukijan muistiin tiedostoja ja hakemistoja pääsylistoja varten. Näiden poistamista varten skriptiin on tehty erityinen käynnistysparametri cleanup. Kun skripti käynnistetään tällä parametrilla, se poistaa kaikki luomansa tiedostot ja hakemistot ja lopettaa sen jälkeen toimintansa. Tämän jälkeen skripti voidaan poistaa, eikä lukijan muistiin jää mitään kulunvalvontasovelluksen tietoja.

4.1.2 Skriptin rakenne

Skriptin alussa luetaan käynnistyksen yhteydessä annetut parametrit ja alustetaan tarvittavat muuttujat. Tämän jälkeen tarkistetaan lukijan muistista pääsylistojen olemassaolo ja tarvittaessa luetaan ne työmuistiin.

Kun pääsylistat on alustettu, avataan kanavat komentojen antamista ja tapahtumien kuuntelua varten. Lopuksi asetetaan skripti ns. tapahtumasilmukkaan, jossa se kuuntelee lukijan tapahtumia ja tarvittaessa reagoi niihin.

Skriptin ydinosa on tapahtumienkäsittelijäfunktio, jota edellä mainittu tapahtumasilmukka jää suorittamaan. Funktio vastaanottaa kaikki lukijan lähettämät tapahtumat, joita ohjelma on rekisteröity vastaanottamaan, ja käsittelee ne. Tätä ominaisuutta hyväksi käyttäen skriptiä pystytään komentamaan omia tapahtumatyyppejä luomalla (ks. luku 4.1.4, Pääsylistojen hallinta).

4.1.3 Käynnistysparametrit

Käynnistysparametreja muuttamalla voidaan ohjata skriptin toimintaa sitä käynnistettäessä. Jotta skripti olisi mahdollisimman yleiskäyttöinen, tiettyjä tärkeimpiä muuttujien arvoja ei ole kirjoitettu kiinteästi koodiin, vaan ne on määritetty käynnistysparametreiksi. Skriptiä käynnistettäessä parametrit kirjoitetaan niille varattuun kenttään lukijan web-käyttöliittymän User Application Management -sivulla (kuva 7). Tämän skriptin parametrit kirjoitetaan muodossa parametri1/arvo1. Parametreista mikään ei ole pakollinen, vaan mikäli jokin niistä jätetään pois, käyttää skripti puuttuvan parametrin arvona ohjelmakoodiin kirjoitettua oletusarvoa.

Lukijassa on neljä digitaalista lähtöä, mikä tarkoittaa, että sillä voidaan ohjata suoraan neljää ulkoista laitetta, esimerkiksi porttia. Kun lukija lukee tietyltä antennilta sallitun tunnisteen, se lähettää kyseiselle antennille määrätystä digitaalilähdöstä aktivointipulssin. Antenneja vastaavat lähdöt määritetään käynnistysparametrilla dio_pins.

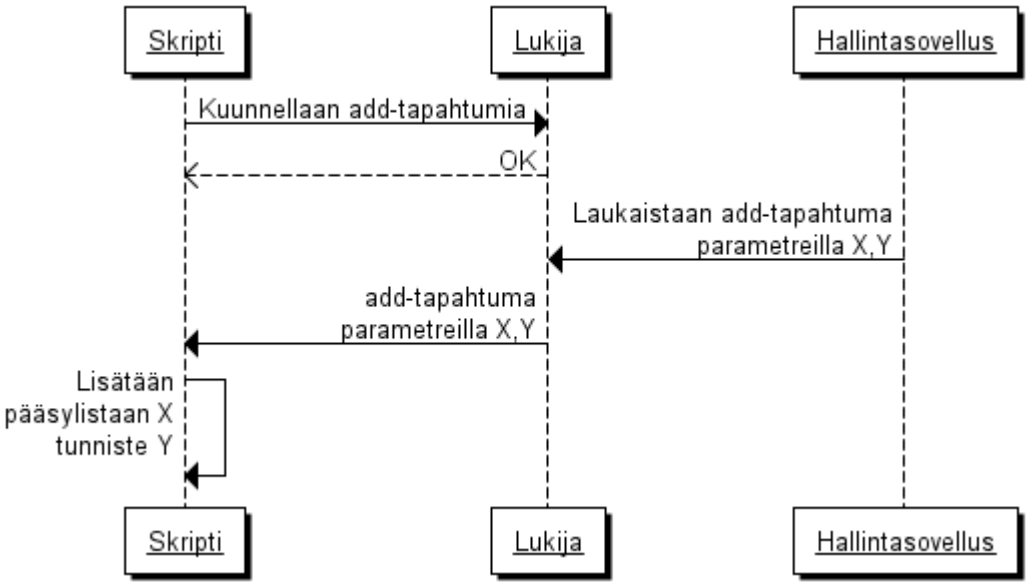
Eri laitteistoista, kytkennöistä ja käyttöympäristöistä riippuen lukijan digitaalilähtöä voidaan joutua pitämään aktiivisena jokin tietty aika. Tämä aika voidaan määrittää millisekunteina parametrilla dio_on_time.

Käytettäessä skripti luo lukijan muistiin tiedostoja ja hakemistoja lupatietokantoja varten. Nämä voidaan poistaa käynnistämällä skripti parametrilla cleanup.

4.1.4 Pääsylistojen hallinta

Lukijan muistissa olevia pääsylistoja hallitaan lukijalle järjestelmän hallintasovelluksen välityksellä lähetettävillä komennoilla. Käytännössä hallintasovelluksen käyttäjä tekee muutoksia järjestelmään graafisen käyttöliittymän kautta ja sovellus hoitaa lukijoiden tietojen päivityksen edellä mainituilla hallintakomennoilla.

Komennot on toteutettu tapahtumapohjaisesti. Esimerkiksi lisättäessä tunniste tiettyyn tunnistuspisteeseen lähettää hallintasovellus kyseiseksi tunnistuspisteeksi määritetylle lukijalle komennon, joka laukaisee tunnisteenlisäystapahtuman. Lukijassa oleva skripti on määritetty kuuntelemaan näitä tapahtumia. Kun lisäystapahtuma laukeaa, skripti havaitsee sen ja päättelee tapahtuman parametreista, mikä tunniste tulee lisätä mille listalle. Alla oleva sekvenssikaavio (kuva 8) havainnollistaa tätä toimintaa.



Kuva 8. Sekvenssikaavio pääsylistojen hallinnasta

Komennot, joilla pääsylistoja hallinnoidaan, on esitelty seuraavalla sivulla olevassa taulukossa (taulukko 2).

Taulukko 2. Lukijan pääsylistojen hallinnointikomennot

Komento	Parametrit	Selitys
vrs_acc.add	antenninumero tunnisteen koodi	Lisää tunnisteen määritetyn antennin sallittujen tunnisteiden listaan
vrs_acc.rem	antenninumero tunnisteen koodi	Poistaa tunnisteen määritetyn antennin sallittujen tunnisteiden listalta
vrs_acc.print_acl	antenninumero	Tulostaa määritetyn antennin pääsylistan sisällön skriptin lokiin.
vrs_acc.save	antenninumero	Tallentaa määritetyn antennin pääsylistan lukijan massamuistiin. Tämä korvaa muistissa jo mahdollisesti olevan pääsylistatiedoston.
vrs_acc.reload	antenninumero	Lataa määritetyn antennin pääsylistan lukijan massamuistista työmuistiin. Mikäli massamuistissa ei ole listaa, lukijan työmuistissa oleva lista jää voimaan.

4.1.5 Tietojen pysyvyys

Skripti asetetaan asennusvaiheessa käynnistymään automaattisesti samalla, kun lukija käynnistyy. Näin varaudutaan mahdollisiin sähkökatkoksiin ja muihin tilanteisiin, joissa lukijasta katkeaa virta ja se joudutaan käynnistämään uudelleen.

Käynnistyessään skripti lukee pääsylistat lukijan massamuistista, joten niitä ei tarvitse päivittää uudelleen hallintasovelluksen kautta lukijan uudelleenkäynnistämisen jälkeen.

Myös kulunvalvontatapahtumat tallentuvat lukijan massamuistiin, kunnes hallintasovellus lukee ne sieltä pois. Näin myös lukijan tapahtumaloki pysyy ajan tasalla mahdollisten vikatilanteiden kohdalla.

4.2 Hallintasovellus

Hallintasovelluksen tehtävänä on olla yhteydessä keskustietokantaan ja sen perusteella pitää yllä lukijoissa toimivien kulunvalvontaskriptien pääsylistoja.

Hallintasovelluksen prototyypin suunnittelu tehtiin Windows-ympäristössä CodeGearin Delphi-ohjelmointikielellä (ent. Borland Delphi). Työn tilaajalla ei työn alkuvaiheessa ollut toiveita tai vaatimuksia ohjelmointikielen suhteen. Aiempia projekteja oli kuitenkin menestyksekkäästi toteutettu Delphillä, joten se oli luonnollinen valinta myös tähän projektiin.

Tämän opinnäytetyön yhteydessä tehty hallintasovellus on vasta konseptivaiheessa oleva prototyyppi, joka on tarkoitettu lähinnä järjestelmän suunnitteluun ja testaukseen. Lopulliseen toiminnallisuuteen ja käyttöliittymän suunnitteluun ei ole panostettu ja nykyisellään sovelluksesta puuttuu vielä joitakin perustoiminnallisuuksia. Sen kehitys jatkuu edelleen.

Lopullinen hallintasovellus tullaan tulevaisuudessa todennäköisesti toteuttamaan webpohjaisena. Prototyypin toteuttaminen Windows-sovelluksena oli kuitenkin olemassa olevan osaamisen ja aikataulun puitteissa mielekkäämpää, sillä lopullisen hallintasovelluksen toteutusmenetelmää ei ole vielä lopullisesti päätetty.

4.2.1 Yhteydet

Prototyyppivaiheen hallintasovelluksen toteuttamisessa suurena apuna oli aiempien projektien yhteydessä kehitetty Delphi-komponentti lukijayhteyden muodostamista, ylläpitoa ja käyttöä varten.

Myös tietokantayhteyttä varten oli olemassa valmiita komponentteja. Tähän valittiin avoimen lähdekoodin ZeosLib-kirjasto, joka on helppokäyttöinen ja sisältää tuen useille erilaisille tietokantajärjestelmille (15). Erilaisten tietokantojen tukea pidettiin tärkeänä, sillä järjestelmässä haluttiin säilyttää mahdollisuus tukea sovelluskohteista riippuen mahdollisimman montaa tietokantajärjestelmää.

4.2.2 Tapahtumaloki

Järjestelmän toimintaperiaate on se, että jokainen tunnistuspiste kerää itsenäisesti tietoja omista tapahtumistaan. Nämä tiedot tallentuvat kunkin lukijan omaan massamuistiin, josta ne täytyy manuaalisesti viedä keskustietokantaan. Mikäli lukijan muisti tulee täyteen, korvataan vanhempia tapahtumia uusilla sitä mukaa, kuin tapahtumia luetaan.

Noin 1000 työntekijää ja 400 ajoneuvoa kattavalla tehdasalueella kertyy normaalina arkipäivänä 7 lukijalta yhteensä n. 3000–4000 luentaa eli keskimäärin 500 luentaa lukijaa kohti (8). INfinity 510 -lukijassa on vapaata muistia n. 5 Mt ja 1 megatavuun mahtuu testien perusteella n. 6000–7000 luentaa. Näin ollen vastaavanlaisessa ympäristössä lukijan muisti tulisi käydä viemässä keskustietokantaan vähintään kahden viikon välein, jotta tietojen menettämiseltä vältyttäisiin.

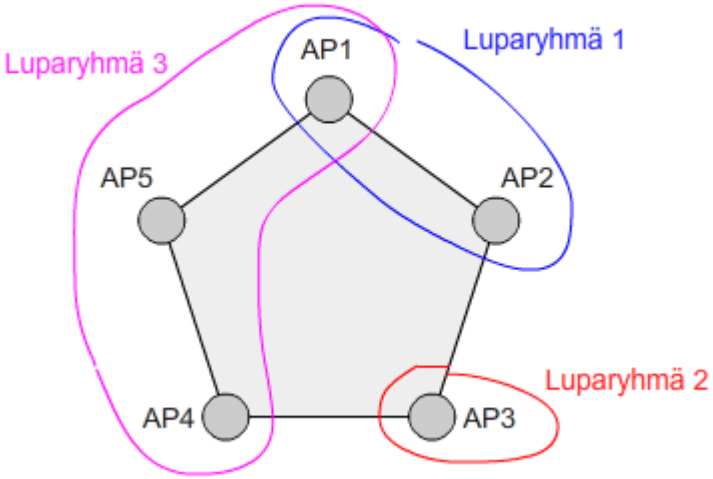
Lukijoiden tapahtumatiedot haetaan keskustietokannan yhteiseen tapahtumalokiin manuaalisesti hallintasovelluksen avulla. Lokiin tallentuu kunkin tapahtuman kohdalla tiedot tunnistuspisteestä, luetusta tunnisteesta, luenta-ajasta sekä hyväksymis- tai hylkäämispäätöksestä. Näin tapahtumia voidaan tarvittaessa seurata mahdollisimman monipuolisilla suodatusvaihtoehdoilla.

4.2.3 Tunnistuspisteet ja luparyhmät

Kohteita (esim. portit, ovet), joita järjestelmällä valvotaan, sanotaan tunnistuspisteiksi. Jokainen lukijan antenni on oma tunnistuspisteensä, joten yksi lukija voi käsittää neljä erillistä tunnistuspistettä, joilla jokaisella on oma toisista riippumaton pääsylistansa. Tunnistuspisteitä hallitaan järjestelmän hallintaohjelmalla, jossa jokaiselle tunnistuspisteelle määritetään lukijan IP-osoite sekä sen lukijaan kytketyn antennin numero, johon tunnistuspiste halutaan määrittää. Näiden lisäksi tunnistuspisteelle annetaan nimi ja kuvaus, jotta ne on helppo tunnistaa.

Järjestelmässä yksittäisille tunnisteille ei määritetä erikseen oikeuksia jokaiseen tunnistuspisteeseen, vaan jokainen tunniste kuuluu johonkin luparyhmään. Kun henkilölle annetaan tunniste ja hänet lisätään järjestelmään, annettu tunniste liitetään johonkin luparyhmään, joka määrittää tunnisteelle annettavat oikeudet. Kuva 9 havainnollistaa

luparyhmäperiaatetta. AP1–5 ovat tunnistuspisteitä (Access Point), joita on käytetty muodostettaessa luparyhmät 1–3.



Kuva 9. Esimerkki luparyhmäjaosta.

4.2.4 Tunnisteiden hallinta

Kun järjestelmä otetaan käyttöön, lisätään siinä käytettävät tunnisteet tietokantaan. Kun järjestelmään lisättävälle henkilölle luovutetaan tunniste, lisätään henkilön tiedot tietokantaan ja määritetään luovutettava tunniste kyseisen henkilön käyttöön.

Yksi tunniste voi kuulua kerrallaan vain yhdelle henkilölle, mutta henkilölle voidaan määrittää useita tunnisteita. Useimmissa tapauksissa pitäisi kuitenkin yhden tunnisteen riittää yhdelle henkilölle, sillä tunnisteen oikeudet voidaan määrittää luparyhmien avulla sopiviksi.

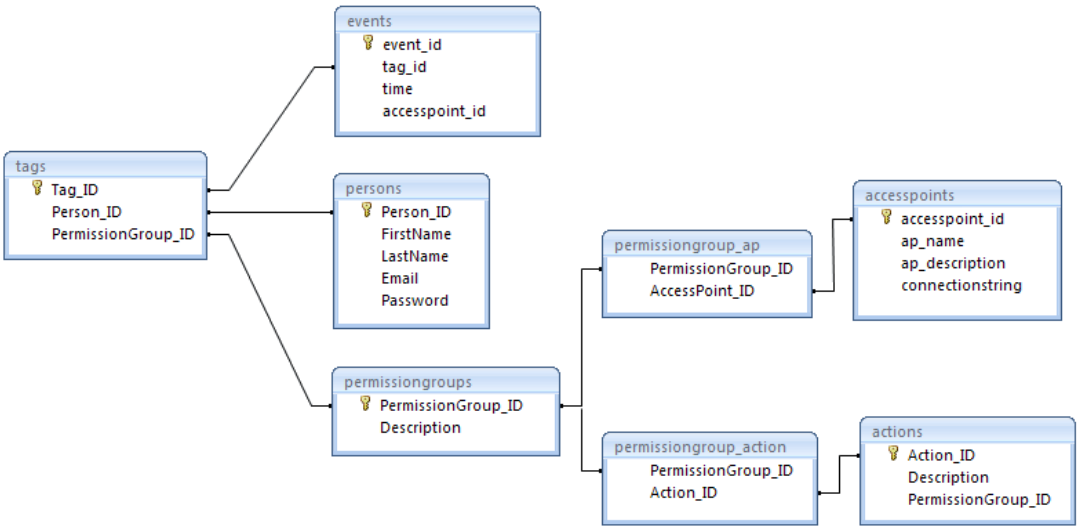
4.3 Tietokanta

Järjestelmän keskustietokannassa on tiedot kaikista järjestelmään määritetyistä tunnisteista ja tunnistuspisteistä sekä henkilöistä, joiden käyttöön tunnisteet on annettu. Näiden lisäksi tietokanta sisältää koko järjestelmän kattavan tapahtumalokin, johon voidaan hakea kunkin lukijan muistissa olevat tiedot. Projektin alkuvaiheessa tiedot täytyy vielä hakea manuaalisesti, mutta tulevaisuudessa tämä tullaan todennäköisesti tekemään automaattisesti.

Tietokantaa suunniteltiin ja testattiin MySQL-tietokantajärjestelmällä, mutta sovellus ei käytä mitään tietokantajärjestelmäkohtaisia erityisominaisuuksia, joten tietokannaksi on mahdollista valita mikä tahansa yleinen relaatiotietokantajärjestelmä.

Tietokannan suunnittelussa pyrittiin välttämään monimutkaista rakennetta ja turhaa tietoa. Tietokannan yksinkertaisuudesta huolimatta sen suunnittelu oli melko haastavaa, koska järjestelmän lopullinen sovelluskohde ei ollut tiedossa, vaan sen tulisi skaalautua mahdollisimman hyvin mahdollisimman moneen tilanteeseen. Kaikkia tilanteita on mahdotonta ottaa huomioon, joten on todennäköistä, että järjestelmän kypsyessä ja ensimmäisten käyttöönottotilanteiden myötä tietokanta tulee vielä jonkin verran muuttumaan. Tavoitteena kuitenkin oli, että perusrakenne pysyisi mahdollisimman paljon muuttumattomana.

Alla oleva kaavio (kuva 10) esittää tietokannan yleisen rakenteen ja taulujen väliset yhteydet.



Kuva 10. Tietokannan taulujen yhteydet

4.3.1 Rakenne

Tietokannan keskeisin taulu, jonka ympärille koko kanta rakentuu, on järjestelmässä käytettävät tunnisteet sisältävä tags-taulu. Se on yhteydessä tunnistekoodin perusteella tapahtumalokiin (events-taulu), henkilön tunnusnumeron perusteella henkilötietotauluun (persons) ja luparyhmätunnuksen perusteella luparyhmätauluun (permissiongroups).

Luparyhmät on suunniteltu käsittämään sekä tunnistuspisteitä että hallintaohjelman toimintoja (accesspoints- ja actions-taulut). Näitä voidaan lisätä aputaulujen (permissiongroup_ap ja permissiongroup_action) avulla kuhunkin luparyhmään tarvittava, rajoittamaton määrä.

5 YHTEENVETO

Työn alussa lukijavalintaan ja järjestelmän suunnitteluun perehdyttiin huolellisesti, jotta saatiin muodostettua selkeä kuva siitä, minkälainen järjestelmästä tehdään ja minkälaista laitteistoa siihen tarvitaan. Kun lopulta saatiin lukijavalinta tehtyä ja sen jälkeen kehitettyä toimiva periaate, jolla lukijaan voitiin tallentaa tietoa skriptin ja tapahtumien avulla, oli lopullisen skriptin toteuttaminen melko suoraviivaista.

Suurin haaste projektissa oli alkuvaikeuksien jälkeen hallintasovelluksen toimintalogiikan suunnittelu ja toteutus. Tämä jäikin etenkin toteutuksen osalta vielä osittain kesken, koska tämän opinnäytetyön aikataulun puitteissa keskityttiin jalostamaan tämänkaltaisen, lukijassa itsenäisesti toimivan kulunvalvontasovelluksen yleinen toimintaperiaate toimivaan kuntoon.

Tämän opinnäytetyön aikana saatiin onnistuneesti luotua toimiva suunnitelma kulunvalvontajärjestelmästä ja hiottua se toimeksiantajan toiveiden mukaiseksi. Lisäksi järjestelmän toteutus aloitettiin ja vietiin jo melko pitkälle.

Jatkokehitystä ajatellen projektissa on vielä paljon työtä. Ensisijaisena tehtävänä on hallintaohjelman käyttöliittymän lopullinen toteutus. Projektin edetessä on käynyt selväksi, että lopullinen hallintasovellus kannattaa tehdä web-pohjaisena, jolloin siitä saadaan mahdollisimman helppokäyttöinen ja käyttöjärjestelmäriippumaton. Lisäksi tapahtumatietojen hakeminen lukijoilta keskustietokantaan olisi syytä automatisoida, jotta järjestelmä voisi toimia itsenäisemmin pidempään.

LÄHTEET

1. Result Service Finland Oy:n antennin esite. Saatavissa: http://www.visi-rfid.fi/media/microstip_antennas_850-870_revC.pdf [viitattu 28.10.2009]

2. Radiotaajuudet. Viestintäviraston Internet-sivut. Saatavissa: http://www.ficora.fi/index/palvelut/palvelutaiheittain/radiotaajuudet.html [viitattu 1.11.2009]

3. RFID-Frequencies. RFID Handbook. Saatavissa: http://rfid-handbook.de/rfid/frequencies.html [viitattu 30.10.2009]

4. RFID. Suomenkielisen Wikipedian artikkeli. Saatavissa: http://fi.wikipedia.org/wiki/RFID [viitattu 30.10.2009]

5. Types of RFID. RFID Handbook. Saatavissa: http://rfid-handbook.de/rfid/types_of_rfid.html [viitattu 30.10.2009]

6. RFID Lab Finland. RFID-teknologia ja sen soveltamismahdollisuudet. Seminaari 2.10.2009. Kotka: Kymenlaakson ammattikorkeakoulu.

7. VISI-RFID Solutions Oy. Saatavissa: http://www.visi-rfid.fi/ [viitattu 29.10.2009]

8. RFID Update: ISO Incorporates Gen2 into RFID Standard. Saatavissa: http://www.rfidupdate.com/articles/index.php?id=1156 [viitattu 30.10.2009]

9. EPC. Suomenkielisen Wikipedian artikkeli. Saatavissa: http://fi.wikipedia.org/wiki/EPC [viitattu 30.10.2009]

10. RFID-tunnistuksen parhaat käytännöt: kuinka toteutan onnistuneen RFID-projektin. Saatavissa: http://www.rfidlab.fi/?file=62 [viitattu 29.10.2009]

11. Sirit INfinity 510 Product Specification Sheet. Saatavissa: http://www.sirit.com/Product_Spec_Sheets/IN510_DS0056_2100709.pdf [viitattu 30.10.2009]

12. INfinity 510 Quick Start Guide, v2.0. Saatavissa: http://www.finn-id.fi/tiedosto/63/ [viitattu 11.6.2009]

13. INfinity 510 Protocol Reference Guide v1.4. Saatavissa: http://www.finn-id.fi/tiedosto/64/ [viitattu 11.6.2009]

14. Access control. Englanninkielisen Wikipedian artikkeli. Saatavissa: http://en.wikipedia.org/wiki/Access_control [viitattu: 31.10.2009]

15. ZeosLib-kirjaston kotisivut. Saatavissa: http://zeos.firmos.at/portal.php [viitattu 17.9.2009]

VISI CONTROL

UHF Passive RFID Access Control System