

Tuomas Huuki

**Testing Automation of Embedded Systems through Web Interfaces**

Notta for Home - System Testing

Thesis

Spring 2010

School of Technology

Information Technology

Embedded Systems



## SEINÄJOEN AMMATTIKORKEAKOULU

### OPINNÄYTETYÖN TIIVISTELMÄ

Koulutusyksikkö: Tekniikan Yksikkö  
Koulutusohjelma: Tietoteknikan koulutusohjelma  
Suuntautumisvaihtoehto: Sulautetut järjestelmät

Tekijä: Tuomas Huuki

Työn nimi: Sulautettujen järjestelmien testaaminen selainkäyttöliittymän kautta

Ohjaaja: Petteri Mäkelä

Vuosi: 2010

Sivumäärä: 45

Liitteiden lukumäärä: 2

---

Tämän opinnäytetyön tarkoituksena oli suunnitella yksinkertainen ja luotettava testausjärjestelmä Notta for Home kodinohjausjärjestelmään. Työ kasittelee ainoastaan tämän järjestelmän testausta, mutta samanlaista menettelyä voidaan käyttää muissakin järjestelmissä. Edellytyksenä on kuitenkin, että järjestelmän selkeimmät virheen aiheuttajat voidaan ohittaa. Käytännössä tämä tarkoittaa helposti vikaantuvia väyliä. Työssä esitellään tapaa jolla testausprosessi voidaan automatisoida palvelemaan järjestelmien käyttöönottajaa, mutta samalla antaa myös tärkeää informaatiota järjestelmän kehittäjälle ja myyjälle.

Koska näin laajan testaamisen toteuttamisen ymmärtäminen edellyttää itse järjestelmän ymmärtämistä, on ensimmäisissä kappaleissa esitelty itse järjestelmä osittain hyvinkin tarkasti. Itse testausjärjestelmää on helppo soveltaa muihinkin kohteisiin jossa testattava järjestelmä koostuu useista komponenteista, mutta jossa itse kokonaisuuden testaaminen halutaan tehdä hyvin yksinkertaisesti ja automatisoidusti. Työn lukija saa lähtökohdat testausprotokollan ideointiin, suunnitteluun ja sen toteutukseen.

Asiasanat: järjestelmätestaaminen, kodinohjaus, protokollat

SEINÄJOKI UNIVERSITY OF APPLIED SCIENCES

**Thesis abstract**

Faculty: School of Technology  
Degree programme: Information Technology  
Specialisation: Embedded Systems

Author: Tuomas Huuki

Title of the thesis: Testing Automation of Embedded Systems through Web Interfaces

Tutor: Petteri Mäkelä

Year: 2010                      Number of pages: 45      Number of appendices: 2

---

The purpose of this thesis was to design a reliable and simple testing protocol for the Notta for Home -home control system. The thesis concentrates specifically on this particular system, but the same methods may be used also in other systems that have to be tested in a manner where the weakest links have to be overridden. The thesis shows a way of automating the testing process to serve the deployer of the system as also the developers and the seller.

Because testing a system as large and complicated as this one requires basic knowledge of the system, the first chapters describe the actual system partially in a very detailed manner. The actual testing environment is easy to implement to other projects where the system consists of many smaller parts, but where the whole system has to be tested automatically and easily. The reader of this thesis gets basic knowledge on the basic concepts, design and implementation of a testing protocol.

Keywords: system testing, home control, protocols

## TABLE OF CONTENTS

<b>TERMS AND ABBREVIATIONS.....</b>	<b>6</b>
<b>FIGURE AND TABLE LIST .....</b>	<b>7</b>
<b>1 INTRODUCTION .....</b>	<b>9</b>
1.1 Company Introduction - Notta Systems Oy .....	10
1.2 Notta for Home.....	12
<b>2 THE TESTING ENVIRONMENT .....</b>	<b>16</b>
2.1 The distribution board .....	16
2.2 The embedded system module .....	20
2.3 The server .....	24
2.4 The panel-pc .....	25
2.5 The base station.....	26
2.6 Basic software architecture .....	27
2.6.1 Software logic .....	28
2.6.2 Server software.....	30
2.6.3 Control Center.....	30
2.6.4 Network.....	31
<b>3 TESTING SOFTWARE .....</b>	<b>32</b>
3.1 Status console.....	33
3.2 Manual testing.....	37
<b>4 TESTING AUTOMATION .....</b>	<b>39</b>
4.1 Data storage server.....	39
4.2 User interface.....	40
4.3 Test protocol .....	41
<b>5 CONCLUSIONS .....</b>	<b>43</b>
<b>REFERENCES.....</b>	<b>44</b>
<b>APPENDIX 1 - PROGRAM FLOW.....</b>	<b>45</b>
<b>APPENDIX 2 - TEST FLOW .....</b>	<b>46</b>

## TERMS AND ABBREVIATIONS

<b>HCS</b>	Home Control System
<b>High-voltage</b>	Devices that are powered by a voltage of 220 volts or more.
<b>Low-voltage</b>	Devices that use an operating voltage of 3.3-24 volts.
<b>Logic module</b>	A physical module that handles connections.
<b>I/O</b>	Physical inputs or outputs.

## FIGURE AND TABLE LIST

FIGURE 1: The Control Center. ....	12
FIGURE 2: The deployment tool. ....	14
FIGURE 3: Bus based devices vs. Notta for Home Topology. ....	17
FIGURE 4: Distribution board. ....	18
FIGURE 5: A logic module without the cover. ....	20
FIGURE 6: Module addressing. ....	21
FIGURE 7: Using digital inputs. ....	22
FIGURE 8: A module cover. ....	24
FIGURE 10: The panel-pc. ....	26
FIGURE 11: The base station. ....	27
FIGURE 12: Basic software architecture. ....	28
FIGURE 13: The Control Center. ....	31
FIGURE 14: The main status window. ....	34
FIGURE 15: The network information window. ....	35
FIGURE 16: The system information window. ....	36

FIGURE 17: The I/O state window.....	37
FIGURE 18: The testing environment.....	39
TABLE 1: The frame format.....	42

# 1 INTRODUCTION

Home control systems are becoming more and more popular today. Also the complexity of these systems is increasing. This means, that every individual component in the system must be adequately tested. Also the whole system must be tested as a fully operating system. The span of these systems range from simple light control to fully automated environment control of the house. Even if a system is used only for lighting control one of the most prominent features is the ability to control all the lights as a group. This usually means that the system has a dedicated switch which allows the user to turn the lights on or off in groups by using only one switch. This is called the home/away switch and is used as the name describes. Modern systems incorporate fire and burglar alarms, video surveillance and much more.

Another new aspect to controlling and keeping an eye on ones home is remote surveillance and remote control. Modern systems make it possible to control the home through a simple web interface over the Internet. Securing the communication channel between the system and the user is a very challenging task. Banks and other secure operators use thousands of dollars each year in certificates and security. Unfortunately this cannot be done for homeowners so other means have to be considered. Also the system must be thoroughly tested for any security leaks and bugs that might hinder the system vulnerable to attacks from the outside. Unfortunately these are common for any web-servers but cannot be allowed to happen in private systems.

## 1.1 Objectives

The purpose of this project was to design an automated testing protocol for the Notta for Home, home control system. The main object was to override all the possible problems that would come up when testing a system that has many



different components that are located in many different environments. The first object was to design a bus that would override the network and its problems. The second object was to design a very simple protocol that could control the different parts of the system independently.

## **1.2 Company Introduction - Notta Systems Oy**

Notta Systems is a company located in Ilmajoki and it is concentrated on a wide range of IT-solutions. Notta Systems was officially established in 2005. The main product of the company is the Notta for Home -home control system. Other areas are web-based solutions, software production, electronics design and subcontracting (Notta Systems, an introduction to the company, 2007, 2). The main technologies used in development include .NET, PHP and Lotus Notes/Domino application development and a wide range of database systems, including MySQL (Notta Systems, an introduction to the company, 2007, 2).

Subcontracting and consulting is split into a different section called partnering. Partnering offers services from specifications and definitions up to personnel training. The partnering workforce is able to implement independent projects or parts of a larger project defined by the customer. It is also possible to hire personnel resources to be used by customers using the customer's tools and practices. At the time of writing this document the key partnering customers are Nokia Oyj and Digia Oyj. (Notta Systems, an introduction to the company, 2007, 4)

Notta Solutions handles the web-based projects and other projects that are not produced for any specific customer, but are widely available products. These also include traditional web sites. Some other examples of these projects are Notta Work Hour management and Church Bell Ringing –automation. (Notta Systems, an introduction to the company, 2007, 5)

### **1.3 Outline of the thesis**

The first chapter gives a broad and simple view of modern home control systems and their features. It also describes one company behind the development of a system. A brief introduction of the company is also given. The second chapter describes the features of the system without going into great detail of the actual technology. The technology behind the system is described in detail in the third chapter. The fourth and fifth chapters deal with software testing and the actual implementation of testing software for the Notta for Home system. The final chapter wraps up all the problems and conclusions that were made during this project.

## 2 NOTTA FOR HOME

The Notta for Home intelligent home control system makes a home a safer place. It improves household economics and usability by automating many of the daily tasks. Notta for Home is like an invisible tireless servant helping in everyday tasks. It combines the many different household systems in to one control point, which is easy to use, administrate and monitor. (Notta for Home website [Referenced 20.11.2009].)

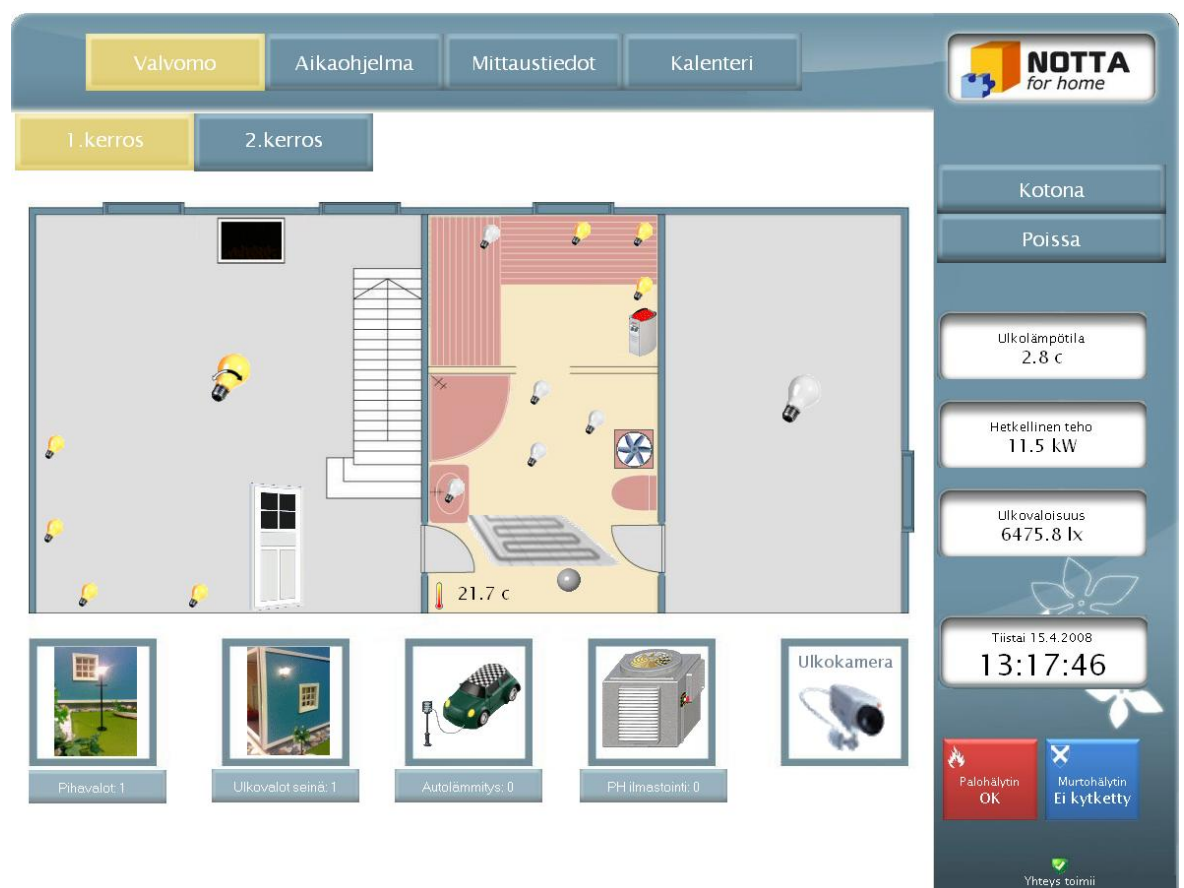


FIGURE 1: The Control Center.

Notta for Home is not a scary and complicated new technology but an easy-to-use tool for everyday life. Notta for Home can be installed into new buildings as well as buildings under renovation. The burglar alarm is a part of the Notta for Home intelligent home control system. If an alarm is triggered a SMS-message is automatically sent to a predefined number or numbers. At the same time, the system starts recording video from the triggered target. It is also possible to monitor a home through a web interface, which allows a view of live video of the home. It also enables the user to check the view of the house over the Internet. (Notta for Home website [Referenced 20.11.2009].)

The delivery includes a distribution board as a ready-to-install system. It is a normal distribution board, which replaces the existing traditional board. The logic modules are expandable as needed. Special cabling needs must be taken into consideration when planning the home cabling. The needed components are planned in co-operation with the electric contractor. The system can be expanded as needed without limits. (Notta for Home website [Referenced 21.11.2009].) The needed functionality for the distribution board is provided by the Notta for Home system controller which includes enough inputs and outputs for a modern house. Adding parallel units can expand the system. (Notta for Home website [Referenced 20.11.2009].)

A Panel-PC with a touch screen is included with the system. It functions as a central control point. The screen can be embedded in to the wall near the front entrance of the house and it displays the current status of the system. All the functions of the house are controlled through this user interface. The panel-pc also displays all the measurement data of the house, for example the temperature values at a given time. (Notta for Home website [Referenced 24.11.2009].)

The different components of the system are linked together with a wireless local area network. A WLAN-base station is included in the delivery, which also enables other computer equipment to be connected to the same network. This makes it possible to use the web browser interface from any computer inside the Notta for Home network. There is also the possibility of using the web-browser interface encrypted over the Internet. (Notta for Home website [Referenced 24.11.2009].)

Alarm relaying is made possible by the included GSM-modem, which can send an SMS-message to any predefined mobile phone. It also enables the controlling of the system with a mobile phone through messaging. Remote use software is provided for Smartphone's. This software is for controlling the whole system and acts as a small control point. The software can be used on any mobile phone supporting java. It supports GSM-data, mobile broadband and WLAN. (Notta for Home website [Referenced 24.11.2009].)

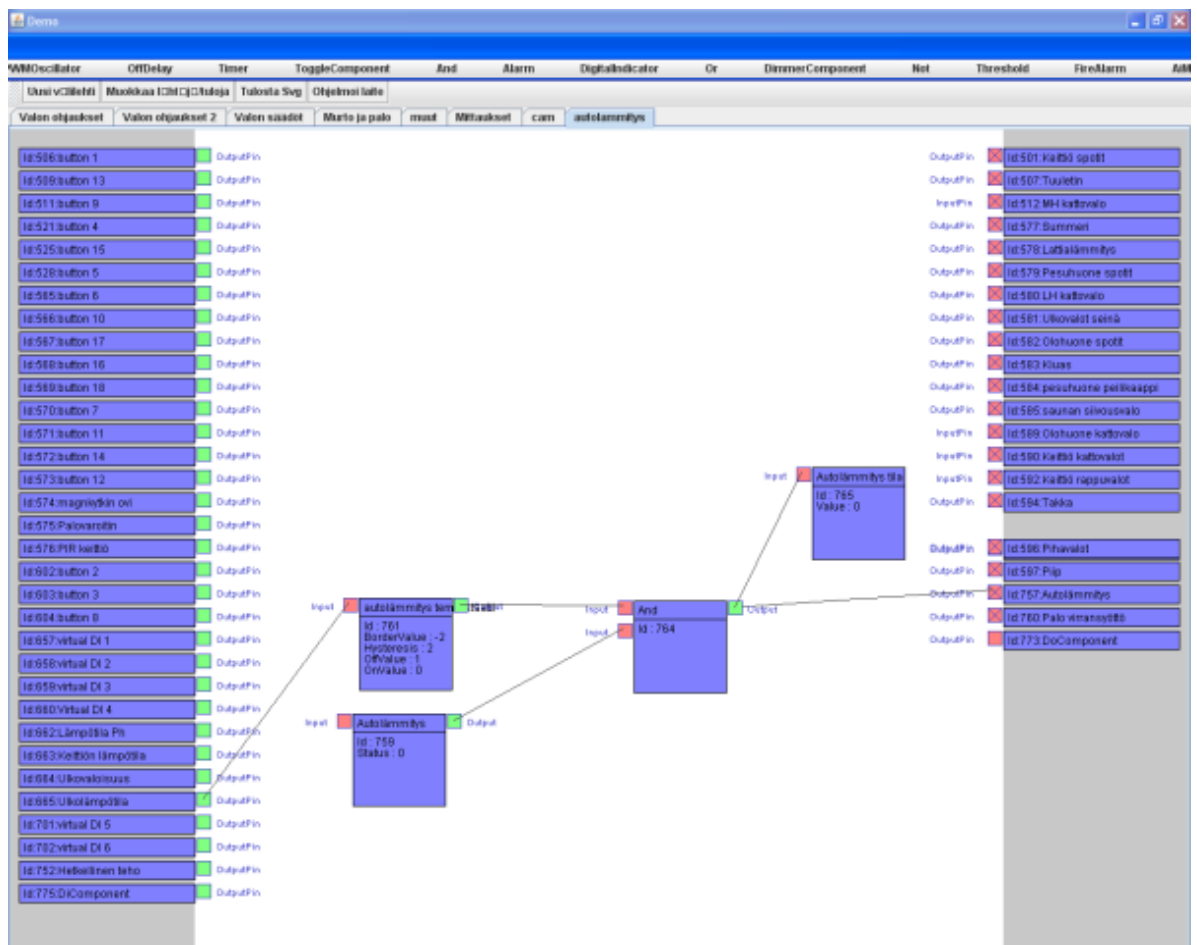


FIGURE 2: The deployment tool.

The deployment tool is the graphical user interface designed to give the system desired functionality effortlessly. Actual deployment is done by moving different types of components to the drawing board and actually connecting them with virtual wires. When creating a component you also give it the parameters depending on the component type, for example temperature, delay, time etc. As

the deployment needs no programming knowledge, the end-user or the electric contractor can do the deployment. (Notta for Home website [Referenced 24.11.2009].)

### **3 THE TESTING ENVIRONMENT**

Contrary to other home control systems, Notta for Home is provided with its own distribution board. The basic design principal is keeping everything centralized in one place and as simple as possible. As a result of this, on-site problem solving is also much faster. (Eteläaho, 2009.)

#### **3.1 The distribution board**

Many of the current systems on the market are bus-based devices, which require special wiring for each device. The problem with these systems is that defects in the system are very hard to locate and as every device on the bus has its own address, conflicts between device addresses are also possible. (Eteläaho, 2009.)

Notta for Home overcomes this problem by using conventional wiring throughout the system. All devices, such as switches and lights are wired to the distribution board in a star topology. In a situation where a device is not functioning correctly, the problem can be easily located to the erroneous device and solved quickly. All measurements can be done straight from the distribution board. (Eteläaho, 2009.)

The only setback from this type of wiring is the length of wiring needed. Good planning of the wiring routes throughout the house normally solves this problem. (Eteläaho, 2009.)

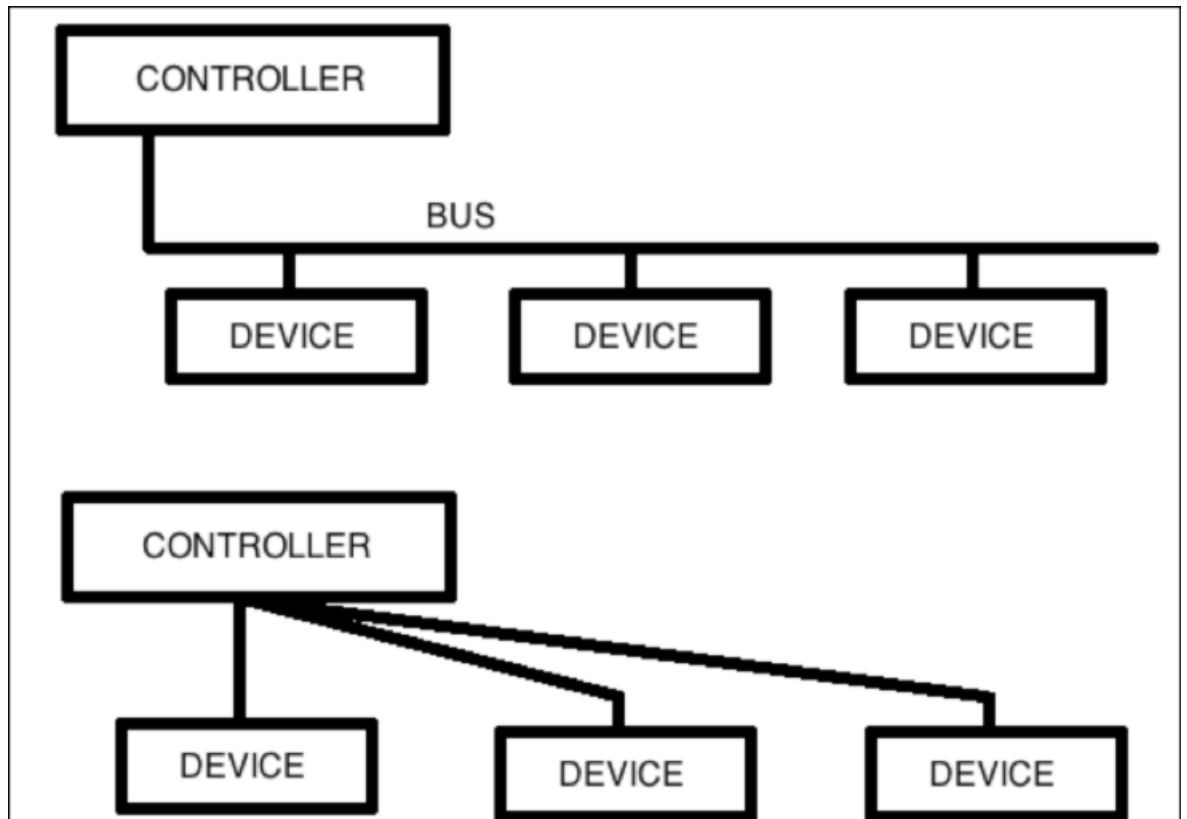


FIGURE 3: Bus based devices vs. Notta for Home Topology.



The distribution board is split into two main sections. The left part houses most of the low-voltage components as the right section houses the high-voltage components, such as fuses and the redundant power supply. The two main sections are split into several subsections. The different subsections are shown in the following figure and explained in numerical order. (Notta for Home hardware specification, 2006, 3.)



FIGURE 4: Distribution board.

The first subsection houses all the row connectors for the low voltage system and includes the main server of the HCS. The server has a power supply that is differentiated from the normal power supply. The server's purpose is explained in its own chapter later on. Row connectors make it easy to wire all the needed connections of the system. Any cables can be simply pushed into the connectors without using any special tools. Each connector is numbered as specified in the electrical planning done by the electrical contractor. (Notta for Home hardware specification, 2006, 4.)

I/O numbering uses two schemes to differentiate the row-connectors and the actual logic modules. This enables the installer to easily locate any inputs without having to know actually where the connection goes inside the system. On the other hand the actual physical input can be easily located from the electric schematics. The logic modules use a numbering according to the type of the I/O. For example, digital outputs are numbered as DI1, DI2, DI3 and so on. Analog outputs correspondingly AO1, AO2 and so on. The row connectors are numbered KA1, KA2, KA3 and so on regardless of the input type. (Notta for Home hardware specification, 2006, 4.)

The second subsection houses all the relays controlled by the software logic. Most of the relays drive high-voltage systems such as lamps, but the system also has support for low-voltage devices and more of the relays can be configured to support these devices. The actual row connectors for the high voltage section reside on the right side of the distribution board on top of the fuses and the ground fault protectors. At the bottom of the third subsection the master embedded system module that provides the main connectivity of the software logic. The module functionality and its connections are described in more detail in the next chapter. This section also houses the dimmers. The number of dimmer packs depends on the configuration of the distribution board and is limited by the physical size of the dimmers. The logic modules drive the dimmers with a 0-10 volt signal and dimmers then adjust the lights accordingly. (Notta for Home hardware specification, 2006, 10.)

All the high-voltage components, connections, fuses and ground fault protectors are located in the second main section. At the top are the row-connectors for all the lights and other devices, below them are the fuses. At the bottom is the redundant uninterruptable power supply. The power supply constantly monitors the main voltage of the system and in case of an blackout automatically switches to backup power. Only the 12- and 24-volt lines have backup power to maximize the operating time. This means, that incase of power loss only the logic modules and specified devices work. These devices are usually the burglar alarm, electronic lock control and emergency lighting. The distribution board is available in many different configurations and sizes, but the basic layout in all distribution boards is the same. (Notta for Home hardware specification, 2006, 12.)

### 3.2 The embedded system module

The heart of the HCS is the embedded system module, which provides the physical connections to the actual distribution board. The embedded system incorporates a proprietary software logic that controls the different functions of the whole system. The physical connections consist of digital and analog inputs, digital and analog outputs, and two serial interfaces for different additional devices. (Notta for Home logic module specification, 2006, 1.)



FIGURE 5: A logic module without the cover.

Each configuration consists of a master module and one or more slave modules. Each module provides a number of physical connections, but the serial interfaces are located only on the master board. The modules can be connected by a ribbon cable in any order, because the actual order of the physical connectors is defined by module addresses. This means that any of the modules inputs and outputs can be ordered as wanted. The master module has the smallest address and the rest of the modules are given an address according to the wanted order. Each module has its own DIP-switches where the address can be defined. (Notta for Home logic module specification, 2006, 5.)

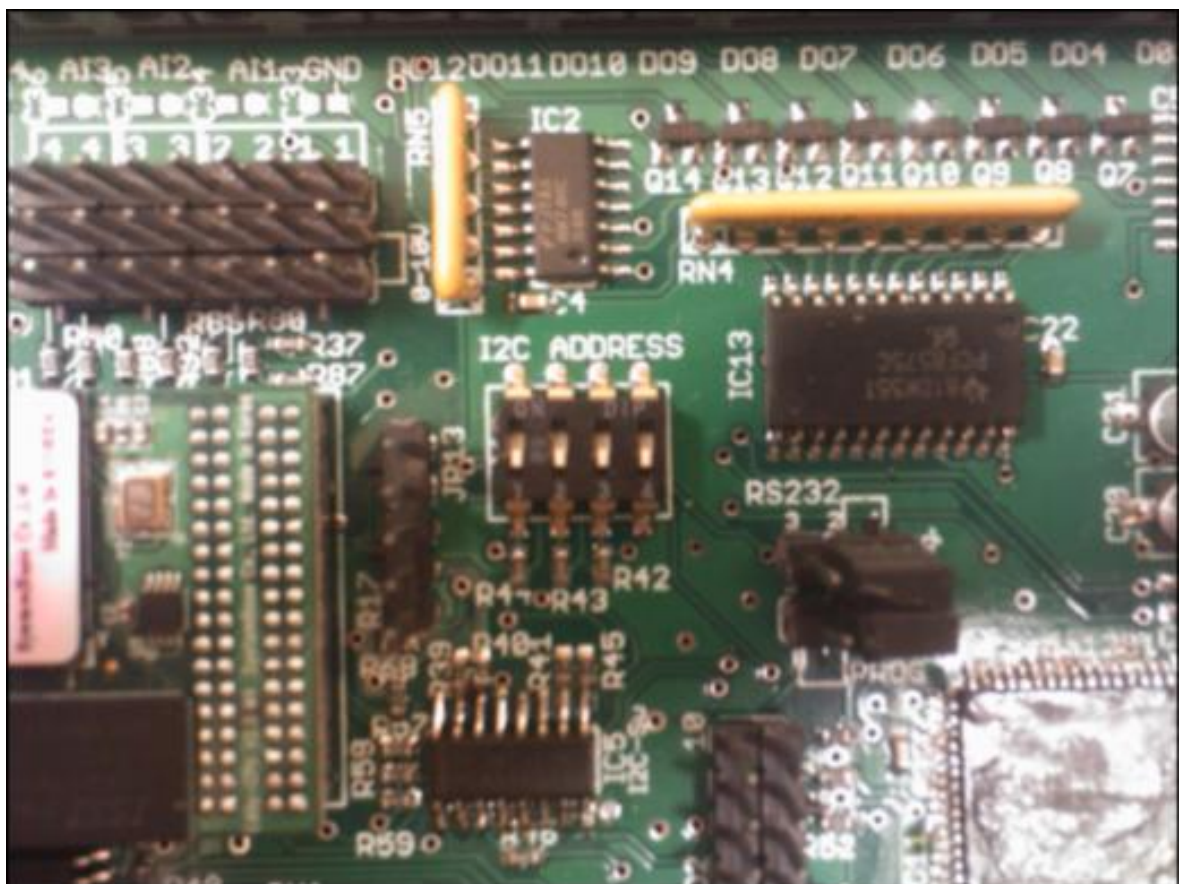


FIGURE 6: Module addressing.

The system is powered by a redundant 24V power supply that automatically switches to battery power if a blackout occurs. This ensures that all vital functionality is maintained even when the rest of the system is down. Vital

functions include devices as emergency lighting and electronic lock control. (Notta for Home logic module specification, 2006, 6)

The purpose of the digital inputs is to provide a connection for simple devices like light switches, motion sensors, magnetic sensors and other devices, which output a simple digital potential free, on/off signal. Devices that use a output with a different voltage potential may also be connected through an external relay. The digital inputs are separated from the other electrical system of the module to ensure maximum compatibility with other devices. The simplest way to use a digital input is demonstrated in the figure below.: (Notta for Home logic module specification, 2006, 8.)

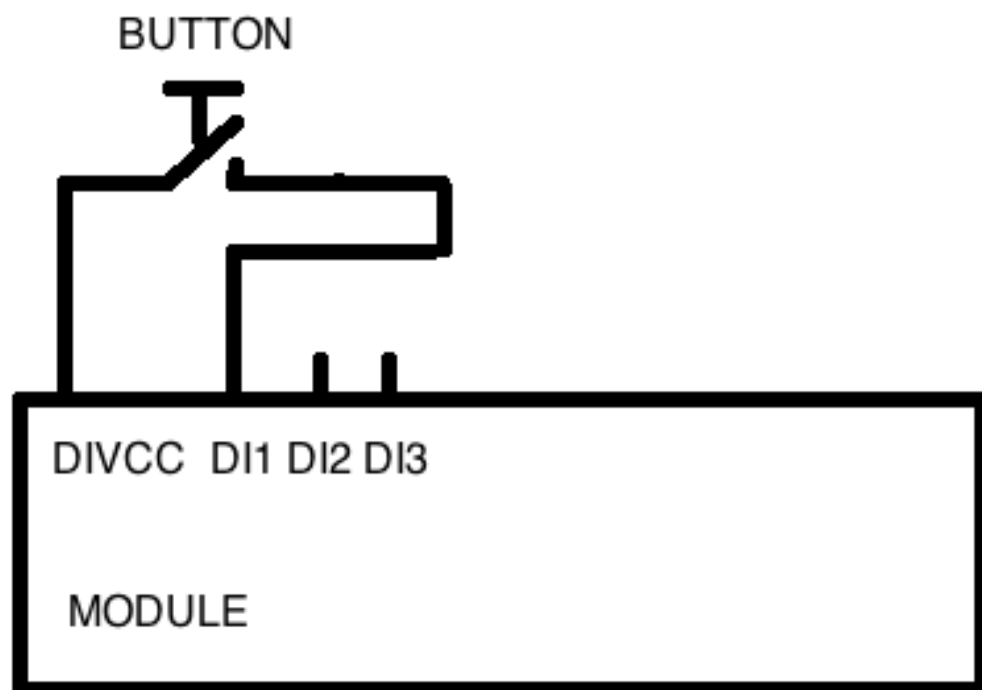


FIGURE 7: Using digital inputs.

Analog inputs serve as physical connections to more complicated devices such as temperature sensors or light sensors. The board has native support for the most commonly used temperature sensor: PT-1000. It can also be configured for a 0-10V input signal simply by changing a few jumper settings on the system board.



Analog inputs are connected similar to digital inputs, except that they share a common ground with the whole system. This ensures maximum accuracy when measuring the potential over analog inputs. (Notta for Home logic module specification, 2006, 9.)

Digital outputs are used to control devices such as relays and latches. Outputs are wired via relays to external devices such as lights. The relays separate the internal 24-volt system from the normal in-house 220-volt system. The actual distribution board also houses 24-volt relay outputs for smaller devices i.e. alarms and electronic door latches. Analog outputs are mainly used for light dimmers. They provide a variable 0-10-volt signal to the control devices. The outputs are also designed to work with any device that can be controlled by a variable voltage signal. (Notta for Home logic module specification, 2006, 17)

The two serial interfaces are used to connect numerous different devices to the system, and provide a way to connect any external device to the system just by supplying the actual software logic a simple driver interface. There are two different types of serial interfaces used on the module. The first one is a normal RS-232 interface, which is intended to be used with the gsm-modem integrated with the system. At this time this is the only use, but if the modem is not used there may be other possibilities also in the future. The second serial interface is a rs-428 interface. This a common interface in many bus-driven devices such as air conditioners and RFID-readers. It provides a versatile interface for all these devices so that they can be integrated in to the system in a seamless fashion. New devices are added by simply providing a driver for these devices. For example new air conditioners can be added by updating the software for the master module. Only the master module has an Ethernet connection that provides connectivity to the server from the modules. This allows the modules to connect to any network, mainly to the Notta for Home internal network. (Notta for Home logic module specification, 2006, 23)

In addition to all the connections mentioned above, there are a few connections that are used in testing and programming of the modules. The most important of these connectors is the test port, which uses a proprietary serial communication

protocol to communicate with the server. Other connectors are used for development and other testing, so they are not explained in this document.

Normally the logic modules are enclosed in a casing that has the I/O marked on the cover. (Notta for Home logic module specification, 2006, 26.)

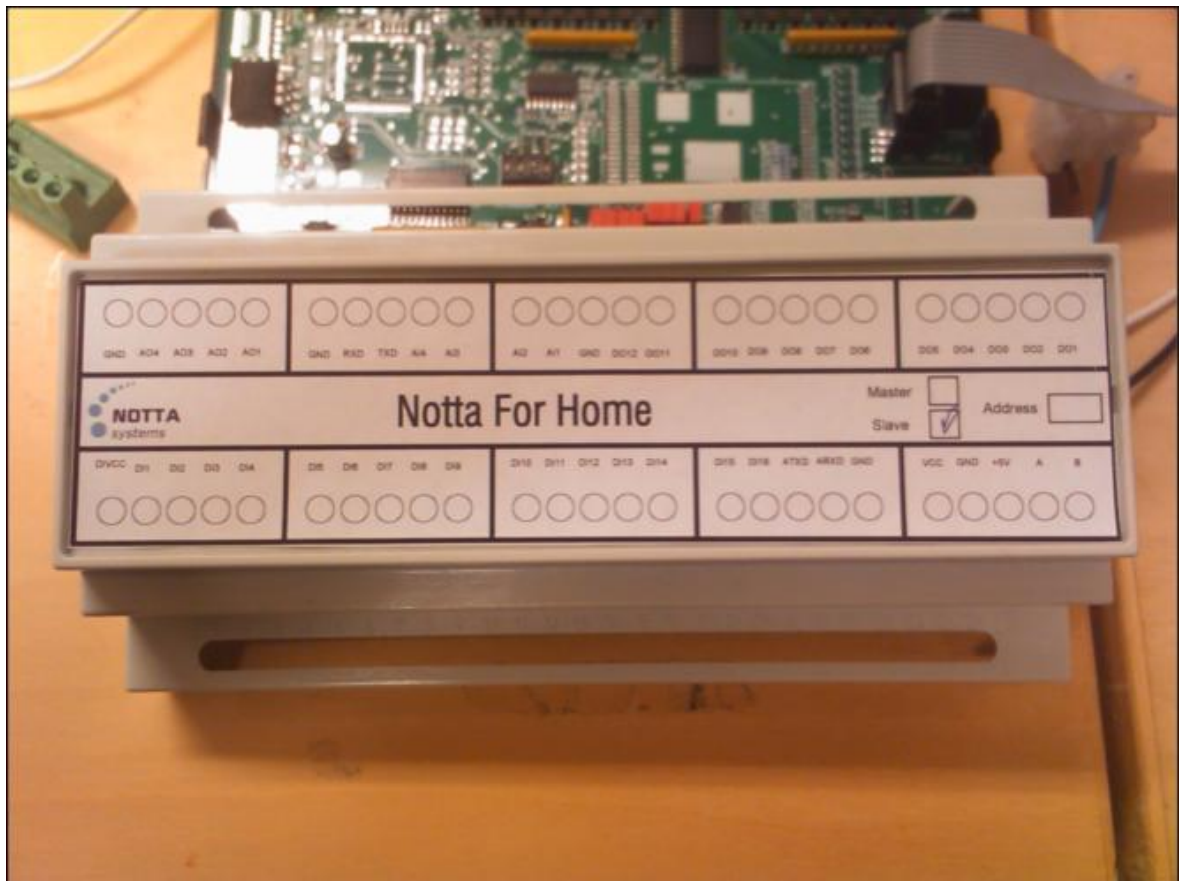


FIGURE 8: A module cover.

### 3.3 The server

The server is based on a mini-itx pc with a solid-state disc and passive cooling. The main reason for this is maximum reliability. With no moving parts the life expectancy of the server hardware is maximized. (Notta for Home server specification, 2008, 1.)

The hardware specifications of the server are:

Motherboard: Intel D201GLY2, with an integrated Celeron 1.2 GHz CPU.

Hard drive: Transcend SSD 4GB

RAM: Kingston DDR800 2GB



FIGURE 9: The Notta for Home main server board.

Because the server is housed inside the distribution board, no external housing is needed. The software and purpose of the server is explained in its own chapter. (notta for Home server specification, 2008, 1.)

### 3.4 The panel-pc

The panel-pc provides a user interface for the end user and is normally located near the entrance of the house. It is a normal pc with an integrated touch screen.



As the server, also this machine has a solid-state drive and passive cooling for maximum reliability. (Notta for Home hardware specification, 2006, 28)

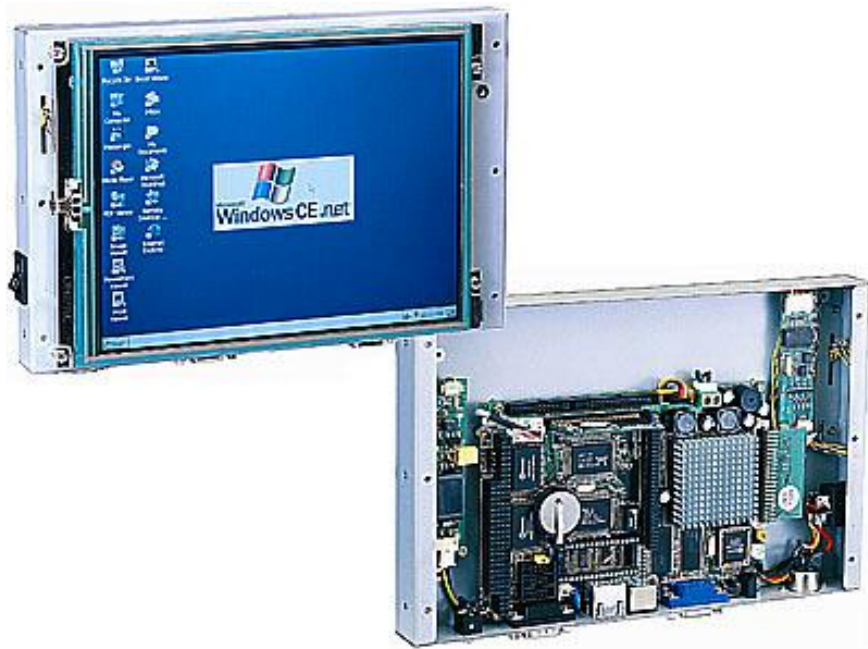


FIGURE 10: The panel-pc.

The current specifications of the panel-pc are:

10.4" TFT LCD Panel PC

Resistive Touch Screen

Vg86-6247-2S VIA Mark 533MHz CPU Board

256MB SDRAM

LAN, COM, Audio, USB

47W External Power Adapter

### 3.5 The base station

All the components are linked together by an Ethernet network. In this case a wlan base station is used for this purpose. The station serves two different purposes. Firstly it differentiates the Notta for Home network from the customers own

network. This makes the internal network much more secure. Secondly it provides a gateway to the internet for updates and remote control of the system. (Notta for Home hardware specification, 2006, 30)

The device currently used as the base station is a Linksys Wlan gateway. The specific model of the station is Linksys WRT54GL version 1.1. (Notta for Home hardware specification, 2006, 30)



FIGURE 11: The base station.

### 3.6 Basic software architecture

The basic architecture of the software can be seen in the figure below. The architecture is designed to be multi-layered so that all the different components of

the system can be developed individually. Each software component is explained here from the lowest level (logic cards) to the user interface (panel-pc). (Notta for Home architecture specification, 2006, 1.)

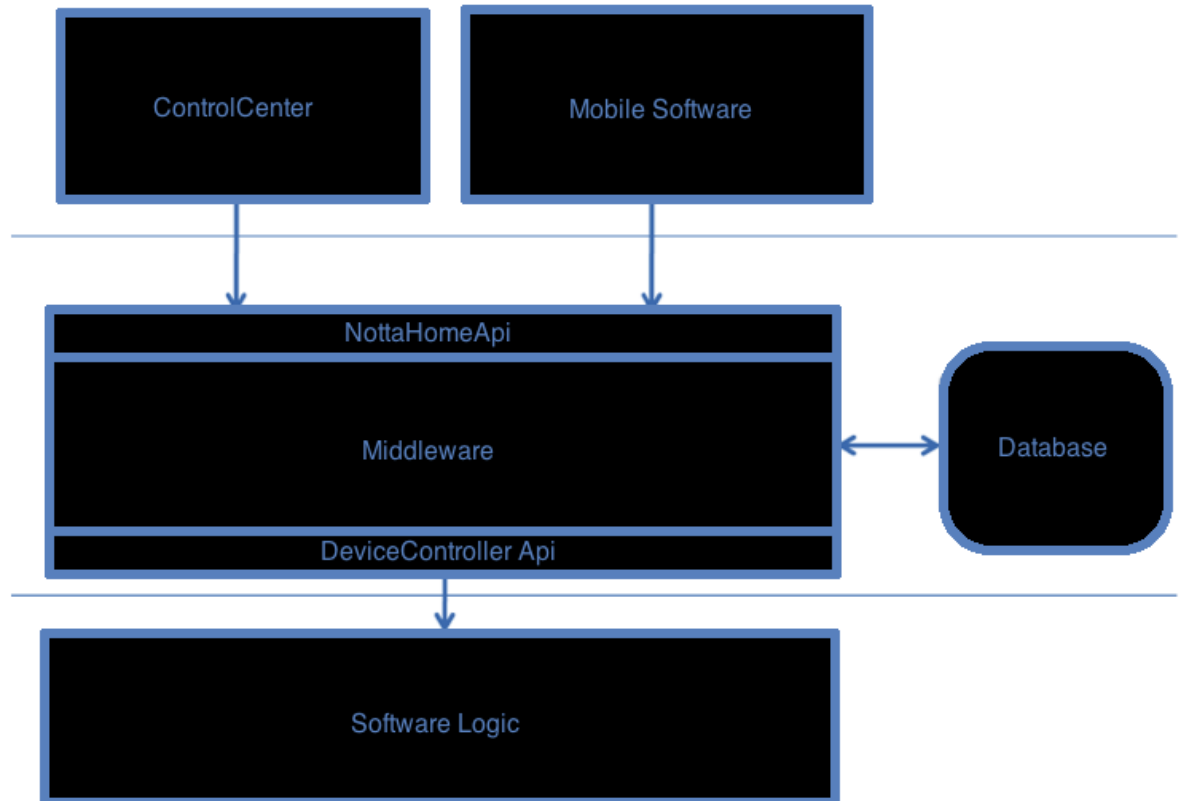


FIGURE 12: Basic software architecture.

### 3.6.1 Software logic

The logic module software (software logic) provides the basic usability of the system. The software consists of a logic program and several drivers. Drivers provide an interface for the actual core logic to interact with several external devices. Each device must have its own driver with a proprietary API. This ensures compatibility with all of the devices and makes it easy to develop new drivers for the logic core. (Notta for Home software specification, 2006, 2.)

The actual logic utilizes an architecture based on a component design. This means, that the most simple form of each device can be represented in the logic. For clarification; a digital input, or simply a switch is represented as a component in the system. This again has its own driver that has been programmed to interact through the API with the actual physical device it is connected to. Similarly, a digital output has the same configuration. If these were to represent a switch and a light, the connection in the logic would be as simple as to connect the switch to the light. This is actually done in that manner in the deployment tool, which is shown later on. (Notta for Home software specification, 2006, 4.)

For more complex components modules are used. Modules are made of two or more components. For example a simple light dimmer is made of an input, output, and a dimmer component. This is then connected to an analog output which is actually used to drive the physical dimmer. Again, a special driver is needed to use the actual physical components. (Notta for Home software specification, 2006, 5)

Two different environments are used on the main logic module board. An ARM9 processor serves as the main processor doing all the complex calculations and a smaller atMega serves as an I/O controller for the physical connections. This is done to keep the slow physical connections separated from the actual software logic that requires more computing power. (Notta for Home software specification, 2006, 7)

Neither of the processors use a widely known operating system, but rather utilize an in-house proprietary system. (Notta for Home software specification, 2006, 11.)

The explained architecture makes it extremely easy to add new devices and their drivers to the system. (Notta for Home software specification, 2006, 12.)

### **3.6.2 Server software**

The server uses a stripped down version of Linux as a base system. To be precise the actual distribution is known as Debian GNU/Linux. The reason for using this distribution is the stableness, usability and updateability of Debian. Debian uses its own packaging system known as dpkg. It also provides an extremely versatile repository system based on http known as apt. These tools allow for easy version tracking and deployment of third party software. (Notta for Home software specification, 2006, 13).

Apache tomcat is used as the main server interface for the Notta for Home server. It provides a web-frontend API to users and developers. This API is used to communicate with the actual embedded logic system. The actual API is a proprietary application but the front-end is available on demand so that users and developers may develop their own control software for the system. This API is known as the Notta for Home Webservice. (Notta for Home software specification, 2006, 15.)

### **3.6.3 Control Center**

Notta for Home provides a Control Center software by default. This is the main user interface of the whole system. The basic layout of the Control Center can be seen below. Lights and other devices are used simply by touching the corresponding icon on the touch screen. (Notta for Home software specification, 2006, 20.)



FIGURE 13: The Control Center

### 3.6.4 Network

All these systems are connected together by a WLAN base station that has an integrated firewall. This allows the network to be monitored continuously, but also provides a secure way to route traffic from the Internet. (Notta for Home software specification, 2006, 21.)

## **4 TESTING SOFTWARE**

The type of testing done in this thesis is called system testing. System testing is classified as testing a whole complete system and how it complies with its specified requirements. (Rex Black, 2002, 10.) Testing is not debugging. Debugging is searching for the cause of the error that has been revealed by testing of by using the software or hardware.

### **4.1 General testing**

Generally, testing can be split into a few type which are named by the function they serve. These parts are called testing levels. The smallest parts test individual parts of code. The largest parts test whole systems that can have thousands of components and millions of lines of code.

#### **4.1.1 Module testing**

Module testing means testing of small and individual pieces of code. It can even be limited to a simple function which must have its behavement analyzed. Module testing is also called unit testing The purpose of module testing is to find simple errors inside and between different program modules (Program testing and tools [Referenced 20.2.2010].) . A simple example would be to test a function that parses a sentence in to words. The test could be run so that a random word generator feeds words into the function and records how the function performs in different situations. This is an extremely important phase of testing because it allows the programmer to evaluate his or her own code.

### **4.1.2 Integration testing**

Integration testing means the phase where all the smaller components of the system are integrated into a larger system one by one and tested. The point of this is to assure that all of the components of the system work together as specified by the application interfaces on architecture specifications of the system. (Program testing and tools [Referenced 20.2.2010].) A good example of this could be a driver that handles user input to a specified device. The parts could be an input parser and the actual driver. The driver could then consist of smaller modules to implement all the functions of the device. Integration testing would start with the input parser and the driver. Then all the functions would be added to the driver one by one.

### **4.1.3 System testing**

System testing assures that the system complies with the given specifications. If large variations are found they are fixed in this phase. System testing also includes performance testing and security. Security considerations must be taken seriously in systems that control physical devices such as locks and doors. Also any unwanted control of ones home would be unacceptable.

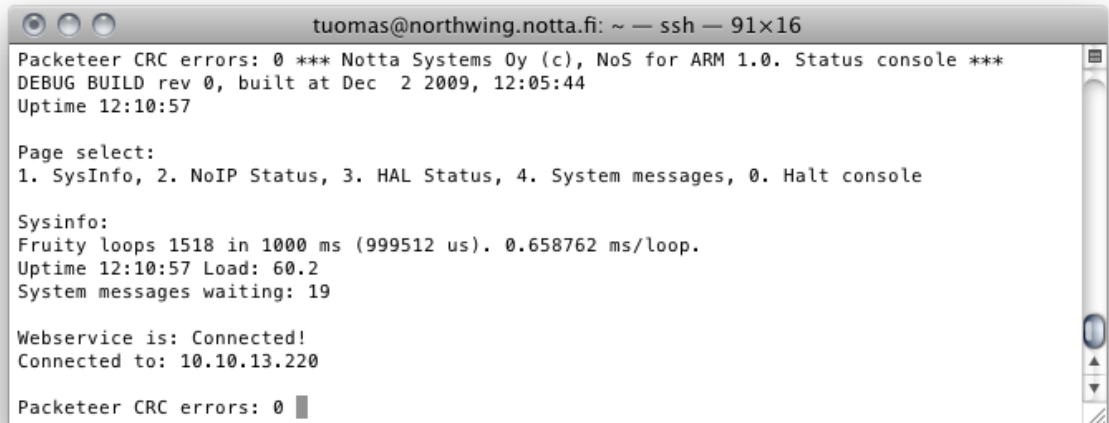
## **4.2 Status console**

Normally the system and the distribution board connections are tested manually. The logic modules provide a simple status console, which displays the current status of the system's internal states. It can be viewed with a special cable connected to the master logic module. Any terminal program can be used to view the output of the console. In the following screenshots an ssh-connection is used to connect to the home server. The actual output is viewed with a simple program named Screen.

The console has four different windows with the first one displaying a general overview of the connections and load. Sysinfo shows the uptime and the current



number of system messages waiting. The purpose of system messages is explained later in this chapter. The next lines show the status of the `web_service` (server) connection and the ip-address of the connected server. The final line shows the number of crc-errors that have occurred between the connection of the server and the logic module.

A screenshot of a terminal window titled "tuomas@northwing.notta.fi: ~ — ssh — 91x16". The terminal displays the following text:

```
Packeteer CRC errors: 0 *** Notta Systems Oy (c), NoS for ARM 1.0. Status console ***
DEBUG BUILD rev 0, built at Dec 2 2009, 12:05:44
Uptime 12:10:57

Page select:
1. SysInfo, 2. NoIP Status, 3. HAL Status, 4. System messages, 0. Halt console

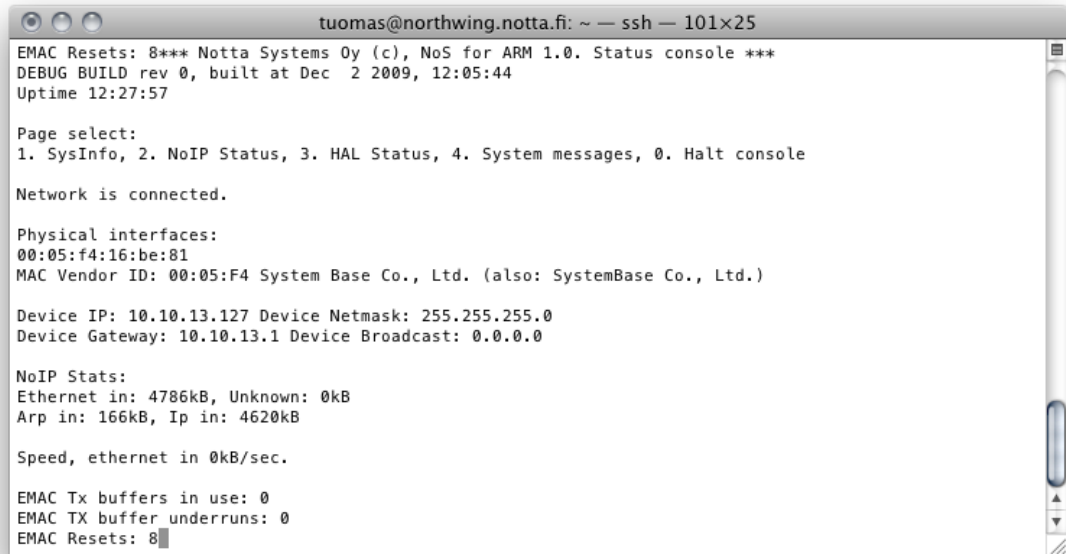
Sysinfo:
Fruity loops 1518 in 1000 ms (999512 us). 0.658762 ms/loop.
Uptime 12:10:57 Load: 60.2
System messages waiting: 19

Webservice is: Connected!
Connected to: 10.10.13.220

Packeteer CRC errors: 0
```

FIGURE 14: The main status window.

The second page displays an overview of the network subsystem. Usually the network configuration is the same on all logic modules but in some special cases it may be modified. This can happen for example in situations where the module is used in an existing network or a fully customized environment.

A screenshot of a terminal window titled "tuomas@northwing.notta.fi: ~ — ssh — 101x25". The terminal displays the following text:

```
EMAC Resets: 8*** Notta Systems Oy (c), NoS for ARM 1.0. Status console ***
DEBUG BUILD rev 0, built at Dec 2 2009, 12:05:44
Uptime 12:27:57

Page select:
1. SysInfo, 2. NoIP Status, 3. HAL Status, 4. System messages, 0. Halt console

Network is connected.

Physical interfaces:
00:05:f4:16:be:81
MAC Vendor ID: 00:05:F4 System Base Co., Ltd. (also: SystemBase Co., Ltd.)

Device IP: 10.10.13.127 Device Netmask: 255.255.255.0
Device Gateway: 10.10.13.1 Device Broadcast: 0.0.0.0

NoIP Stats:
Ethernet in: 4786kB, Unknown: 0kB
Arp in: 166kB, Ip in: 4620kB

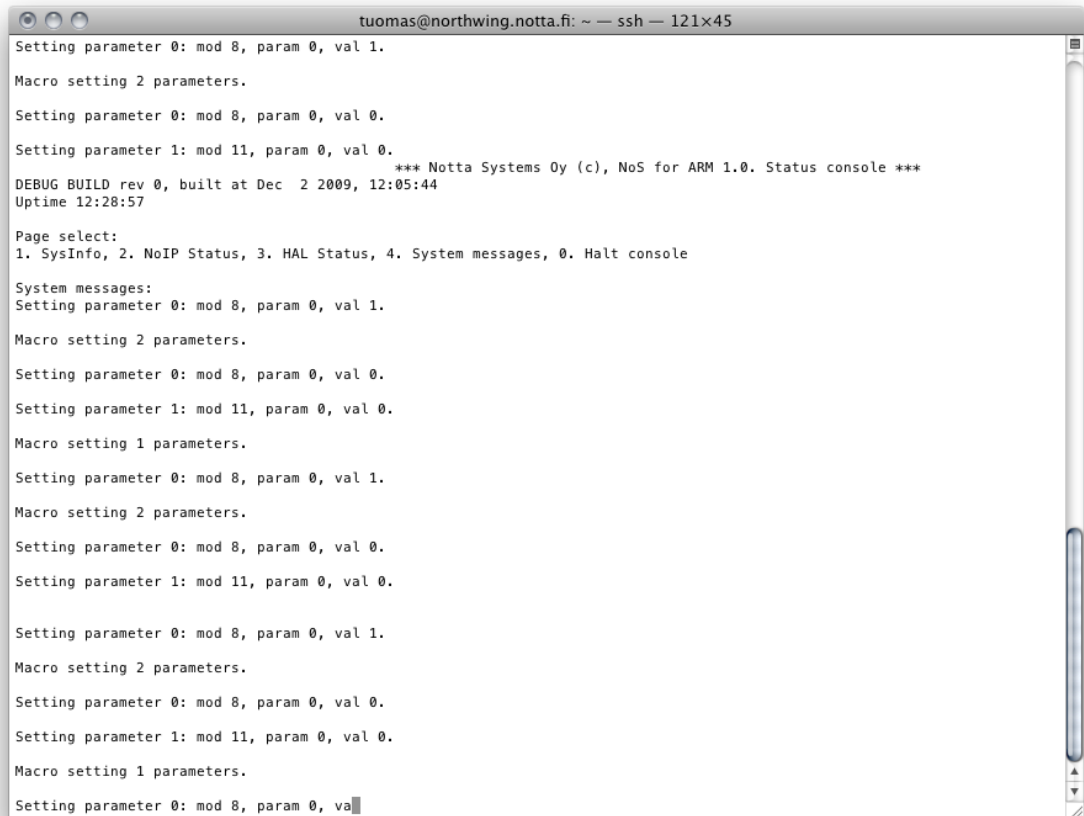
Speed, ethernet in 0kB/sec.

EMAC Tx buffers in use: 0
EMAC TX buffer underruns: 0
EMAC Resets: 8
```

FIGURE 15: The network information window.

For now there is no support for a dhcp server or dynamic configuration and the settings are predefined in all systems. Anyhow, custom environments may still be configured on a special request.

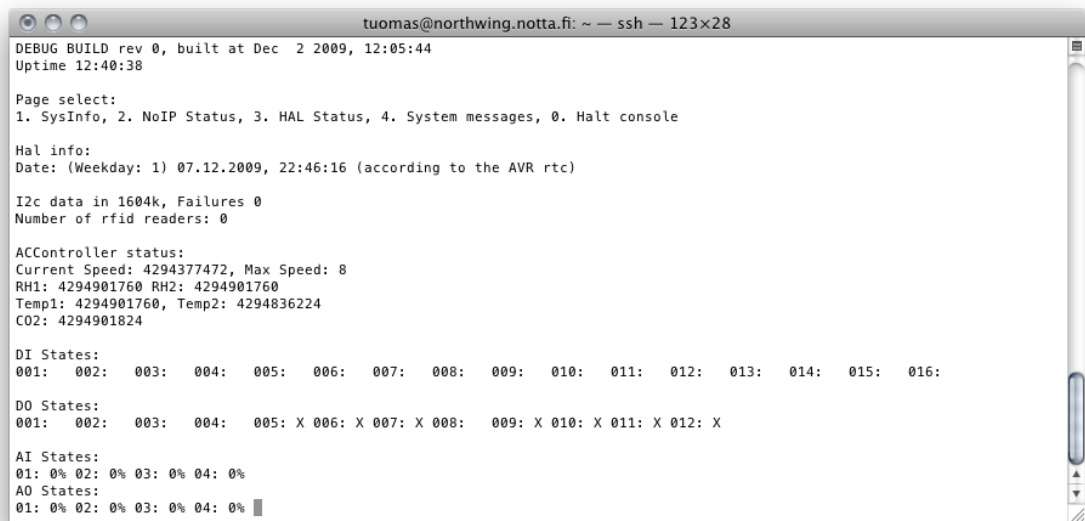
The fourth page displays the system messages, which are used to get an view inside the operating system. All error messages are displayed here.



```
tuomas@northwing.notta.fi: ~ — ssh — 121x45
Setting parameter 0: mod 8, param 0, val 1.
Macro setting 2 parameters.
Setting parameter 0: mod 8, param 0, val 0.
Setting parameter 1: mod 11, param 0, val 0.
DEBUG BUILD rev 0, built at Dec 2 2009, 12:05:44      *** Notta Systems Oy (c), NoS for ARM 1.0. Status console ***
Uptime 12:28:57
Page select:
1. SysInfo, 2. NoIP Status, 3. HAL Status, 4. System messages, 0. Halt console
System messages:
Setting parameter 0: mod 8, param 0, val 1.
Macro setting 2 parameters.
Setting parameter 0: mod 8, param 0, val 0.
Setting parameter 1: mod 11, param 0, val 0.
Macro setting 1 parameters.
Setting parameter 0: mod 8, param 0, val 1.
Macro setting 2 parameters.
Setting parameter 0: mod 8, param 0, val 0.
Setting parameter 1: mod 11, param 0, val 0.
Setting parameter 0: mod 8, param 0, val 1.
Macro setting 2 parameters.
Setting parameter 0: mod 8, param 0, val 0.
Setting parameter 1: mod 11, param 0, val 0.
Macro setting 1 parameters.
Setting parameter 0: mod 8, param 0, va
```

FIGURE 16: The system information window.

The most important page of these four is page three. This page gives an overview of the different I/O components and is used in the systems traditional testing.



```

tuomas@northwing.notta.fi: ~ -- ssh -- 123x28
DEBUG BUILD rev 0, built at Dec  2 2009, 12:05:44
Uptime 12:40:38

Page select:
1. SysInfo, 2. NoIP Status, 3. HAL Status, 4. System messages, 0. Halt console

Hal info:
Date: (Weekday: 1) 07.12.2009, 22:46:16 (according to the AVR rtc)

I2c data in 1604k, Failures 0
Number of rfid readers: 0

ACController status:
Current Speed: 4294377472, Max Speed: 8
RH1: 4294901760 RH2: 4294901760
Temp1: 4294901760, Temp2: 4294836224
CO2: 4294901824

DI States:
001: 002: 003: 004: 005: 006: 007: 008: 009: 010: 011: 012: 013: 014: 015: 016:

DO States:
001: 002: 003: 004: 005: X 006: X 007: X 008: 009: X 010: X 011: X 012: X

AI States:
01: 0% 02: 0% 03: 0% 04: 0%
AO States:
01: 0% 02: 0% 03: 0% 04: 0%

```

FIGURE 17: The I/O state window.

The first row displays the date of the system. The date is synchronized from the server every time the server starts. Next there is the amount of data transferred between the main processor and the smaller processor that acts as the I/O controller. Failures are counted if the controller responds to the commands too slowly, or if a command is not executed at all. The number of tag-readers (RFID-readers is also displayed). AC-Controller status displays the status of the air conditioner driver. As we can see in the screenshot, the controller is not connected because none of the values are valid. Finally there are the status rows for the inputs and outputs.

### 4.3 Manual testing

Normally the testing of the boards is done simply by going through each output and input and setting them manually to a specific state. This is a process that requires a lot of time and labor. All connectivity, including network access must be tested manually by pinging through all the devices.

First each input is verified by connecting buttons to them and going through all the buttons separately. At the same time it has to be verified that the corresponding input is shown in the console. The same must be done with the outputs, except that the system offers no mechanism to override the normal programming and switch the outputs on. Traditionally this has been done by programming the software logic with a simple program that connects the inputs straight to the outputs. LED's are then connected to the outputs to verify that the outputs work. This again requires a lot of time. An additional problem with this is that the amount of inputs and outputs differ. This means that all the connections cannot be tested at the same time.

Analog connections are also tested in the same manner. The analog inputs are switched to 0-10V mode and the outputs are connected to the inputs. Then a simple loopback program is programmed into the system.

These tests only verify the correct operation of the logic modules and the distribution board must be tested in the same manner. When new module- or driver-programs are written for the system all these tests have to be rerun to verify the integrity of the system. After testing all the information must be written into a test report and the report filed manually. Here, even more human error is present. By automating this whole process only a quick check is needed to verify the correct operation of the system and all the testing data is automatically saved from every session.

## 5 TESTING AUTOMATION

The main object of automating the testing in this case was to make the testing easier and faster. The program also provides problem solving options and ensures that the whole system is working properly. The main purpose of this thesis was to design a testing environment and implement the protocol for the software logic.

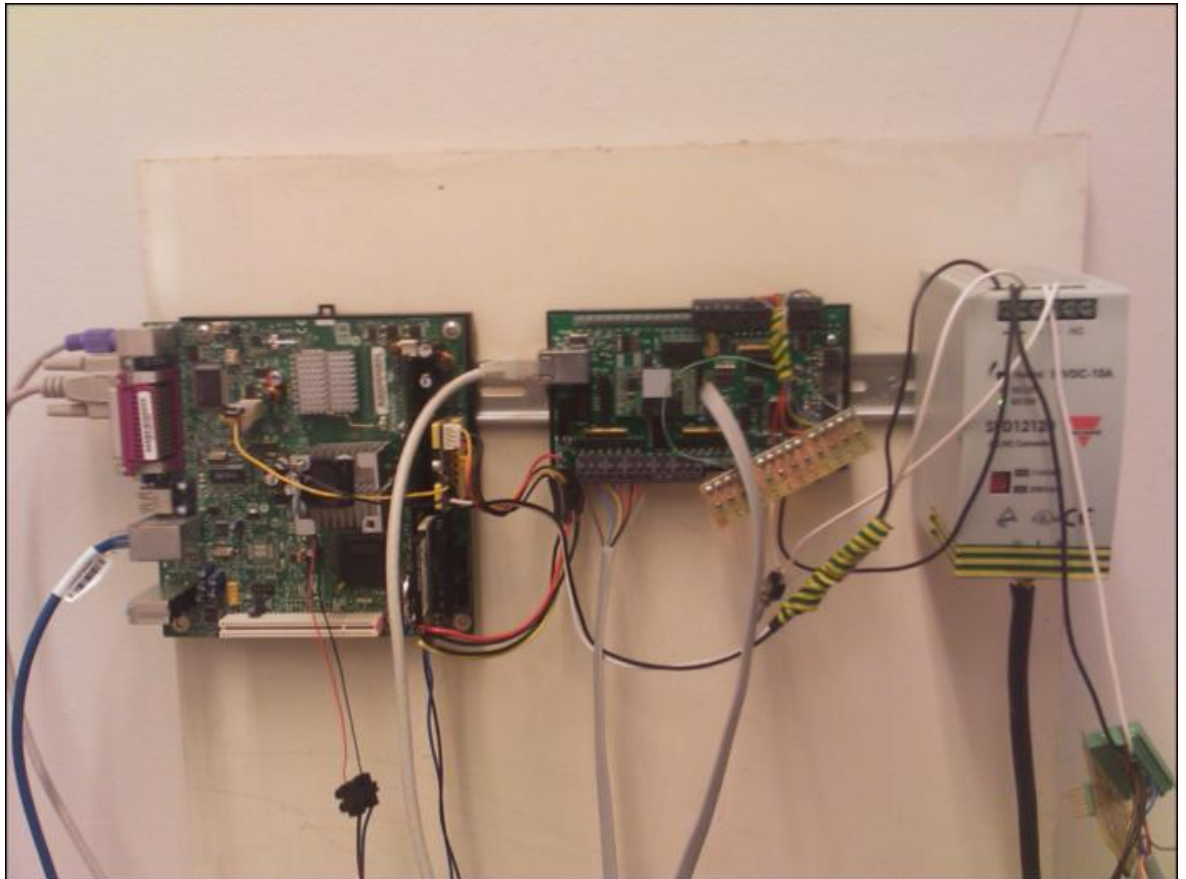


FIGURE 18: The testing environment.

### 5.1 Data storage server

All the test data is stored centrally on one server. The server also handles all the client information that is checked during testing. All sites must be registered with the data server in order to complete and verify the system. This means, that the sites must be registered before the testing is started on location. The data server

also holds software version information, which is checked against the versions installed in the tested systems.

## 5.2 User interface

The user interface used is a special web page hosted on the Notta for Home server. The interface communicates with the logic module through a special cable that is connected to the server's serial port. This makes it possible to override all the possible network problems and provides a secure connection to the logic module. It also eliminates any user errors between other connections because the connection to the module is checked first. The tests are executed in steps where each step is guided by the web interface. A flow chart of the process can be found in appendix 1.

The testing is started by connecting the logic module to the server with a special cable and opening the web page to the testing software. The first phase is filling out the test site information. The information is verified by connecting to the data server. At the same time, the software versions of each component are checked and verified. If a mismatch is found in any component the program displays an alert with the erroneous components and gives instructions on how to update the necessary components to the newest versions. This phase also verifies the connectivity to the logic module.

If all the preceding steps succeed the test software checks the network connectivity of the devices. First the connection to the gateway is checked from the server and then the other connections, such as the connection to the panel-pc and the logic module. Again, if a step fails, the user is notified. Because all configurations might not have all the components of the system, errors in this step are verified from the user. For example, all system configurations do not include the panel-pc in which case the connection to it does not need to be checked.

The amount of modules is checked and the number of I/O is calculated. If it does not match with the initial information given, the user is asked to check the card

cabling and addressing. If the cards match, a basic automated test program is uploaded into the software logic.

After the automated test program has been uploaded, the user is asked to connect all the outputs to the inputs for automated testing. This allows the software to go through each input by switching each output on and checking if the corresponding input state changes. Analog connections are tested in the same manner using the 0-10V configuration. If uploading or automated testing fails the program automatically sends an error report to the data server and asks the user to contact support for assistance. Usually this is due to broken hardware or an invalid configuration which requires more knowledge of the system to be fixed.

The final step is restoring the normal connections to the logic modules and sending the test report to the data server. Finally the actual site program can be uploaded to the software logic and the rest of the connections tested.

### **5.3 Test protocol**

The protocol between the software logic and the testing software is implemented as a simple serial protocol. It utilizes a frame format which consists of a header, body and a simple crc calculation. The crc calculation eliminates any corrupted frames and allows both of the devices to check for any errors in the communication between them.



The frame format is defined as follows:

TABLE 1: The frame format.

		<b>HEADER</b>			
Byte		1	2	3	4
	Start (0xAA)	Reserved	Reserved	Reserved	Reserved
		5	6	7	8
	Command	Command	Command	Command	Command
		<b>BODY</b>			
Byte		9	10	11	12
	Data	Data	Data	Data	Data
		13	14	15	16
	Data	Data	Data	Data	Data
		<b>CRC</b>			
Byte		17	18	19	20
	Reserved	Reserved	Reserved	Reserved	Reserved
		21	22	23	24
	CRC	CRC	CRC	CRC	CRC

By using this protocol the testing software may communicate with the I/O control directly by bypassing the software logic. This allows for checking the states of the I/O without having to rely on any other software. It also grants the usage of the driver functions directly.

The crc is the 32-bit one's complement of the one's complement sum of all the 16-bit words in the frame. For the purposes of computing the checksum, the value of the checksum field is zero.

## **6 CONCLUSIONS**

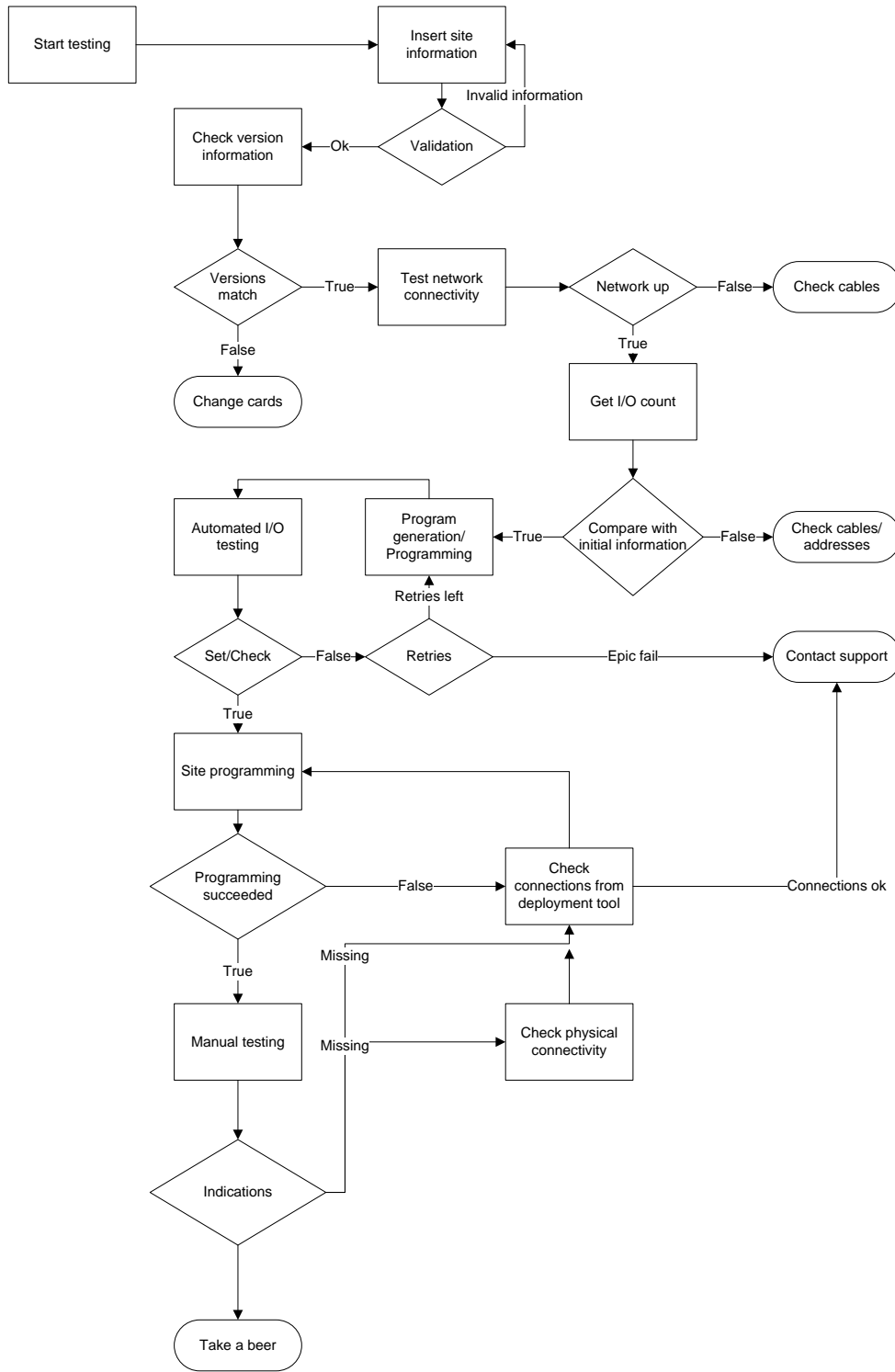
At the time of writing this document the whole environment has not been implemented yet and the actual web-interface is still in development. The protocol designed for the software logic is in early stages and has already been tested. More tests are needed to verify the full functionality of the environment.

During this project many actual software bugs were also found. Bugs, which do not directly relate to this software, were found when developing the testing environment. The importance of testing has also become very clear during this project. After the whole system has been implemented the testing software will speed up the process of discovering and fixing errors.

## REFERENCES

- Black, Rex; (2002). Managing the Testing Process (2nd ed.). Wiley Publishing.
- Notta for Home hardware specification, 2006. [Company internal documentation] Notta Systems Oy.
- Notta for Home logic module specification, 2006. [Company internal documentation] Notta Systems Oy.
- Notta for Home architecture specification, 2006. [Company internal documentation] Notta Systems Oy.
- Notta for Home software specification, 2006. [Company internal documentation] Notta Systems Oy.
- Notta for Home testing plan, 2007. [Company internal documentation] Notta Systems Oy.
- Notta Systems, an introduction to the company, 2007. [Company internal documentation] Notta Systems Oy.
- Notta for Home server specification, 2008. [Company internal documentation] Notta Systems Oy.
- Eteläaho, J. 2009. Notta for Home. [Personal email]. Receiver: Tuomas Huuki. [Viitattu 24.11.2009]
- Notta Systems Oy website [web page] Available at <http://www.notta.fi>
- Notta for Home website [web page] Available at <http://www.notta.fi/home>
- Ohjelmistotekniikat ja siinä käytettävät työkalut [web page] Available at <http://www.mit.jyu.fi/opiskelu/seminarit/ohjelmistotekniikka/testaus>

# APPENDIX 1 - PROGRAM FLOW



## APPENDIX 2 - TEST FLOW

