Curriculum Vitae

Personal Information

Name:	Nicolaas Quirinus Linnenbank
Firstname:	Niek
Address:	Keelweg 33, 5508RP, Veldhoven
Mobile:	06-38769022
E-mail:	nieklinnenbank@gmail.com
Website:	http://www.nieklinnenbank.nl
Date of birth:	April 24, 1988
Nationality:	Dutch
Driving License:	Category B (car)

Education

2009-2011:	Master Parallel Distributed Computer Systems, Vrije Universiteit A'dam (avg: 8.3)
2008-2009:	Hogeschool Utrecht Honours programme
2005-2009:	Bachelor Informatics, Hogeschool Utrecht (cum laude, avg: 8.4)
2000-2005:	Higher General Secondary Education, Minkema College

Employment

2012-now:	Software Designer at ICT Automatisering (customers: ASML, Verkerkgroep)
2011:	Performance Engineer at Rabobank
2010-2011:	Software Developer at SMARTPosition BV, Utrecht (parttime)
2009-2010:	Open Source Consultant at Industrial TSI, Nieuwegein (parttime)
2007-2008:	Traineeship at LogicaCMG in Rotterdam
2006-2007:	System administrator of Linux server at Hogeschool Utrecht (parttime)
2007:	Assistent Java Programming at Hogeschool Utrecht (parttime)
2006-2007:	Assistent Web Design at Hogeschool Utrecht (parttime)

Summary

Niek Linnenbank is an experienced Software Designer who enjoys to design, implement and deliver high-quality (embedded) systems for the customer. His primary expertise is (embedded) operating systems, especially Linux (11+ years), MINIX and his own operating system (FreeNOS).

Skills

- Languages: C/C++, Assembly, Java, C#, PHP, Python, Bash, SQL, Perl, UML, ERD, LaTeX.
- Architectures: Intel x86, Intel SCC, ARM BCM283x, ARM OMAP 3xxx, Zilog Z8.
- Operating Systems: FreeNOS, MINIX, GNU/Linux, FreeBSD, NetBSD, OpenBSD, MacOS.
- Frameworks: Android, Qt4/5, SCRUM, Gstreamer, AIR, .NET, iOS, Eclipse, Java Swing.
- Virtualizers: Xen, VMWare, Bochs, Qemu, kvm, VirtualBox.
- Server Solutions: OpenSSH, MySQL, Apache, Subversion, Samba, BIND, OpenLDAP, Bacula, Zarafa, OTRS, Nagios, Monit, SugarCRM, Asterisk, Postfix.
- Web Development: (X)HTML4/5, XML, CSS, Javascript, Bootstrap, JointJS, MySQL, PHP, Perl/CGI, Servlets/JSP, GIMP and Photoshop.
- Development Tools: Subversion, Git, Jenkins, Clearcase, GNU make, SCons, Bouwer, GNU autotools, GDB, Doxygen.
- Protocols: TCP, IP, UDP, ZWave, DNS, HTTP, IRC, Ethernet, ARP, DHCP, XML/SOAP, FTP, SIP.
- Filesystems: Ext2, Ext3, Minix Filesystem, Linnenbank Filesystem, ProcFS, TmpFS.
- Communication: fluent in Dutch and English, basics of French and German.

Projects Overview

2015:	Implemented Free Niek's Operating System (FreeNOS) on ARM (Raspberry Pi), and added Intel SMP, USB and IP networking support. See http://www.FreeNOS.org
2015:	For Verkerkgroep I designed and implemented a novel domotics platform based on the Raspberry Pi, ZWave, HTML5 and JointJS. With a Bootstrap responsive frontend website the user can configure ZWave devices simply by click-and-drag.
2014:	Created Bouwer, a new Python-based build system for C/C++. It is highly configurable (Kconfig syntax), multi-platform, modular, object-orientated, parallel. See http://nieklinnenbank.nl/bouwer
2014:	For Verkerkgroep I designed a high performance VoIP video conferencing SIP client in C++ for use in their realtime embedded alarm devices.
2012-2014:	For ASML's Motion Control platform I designed and implemented various C device drivers for sensor and actuator I/O boards.
2011:	Started the company "Linnensoft", where I develop mobile games with Box2D for all major platforms, including Android, iOS and Blackberry.
2011:	My PDCS master project is about porting MINIX3 to the Intel Single Chip Cloud Computer, an experimental new architecture aimed at scaling multicore systems to hundreds to thousands of cores on a single chip. Supervised by Andrew S. Tanenbaum.
2011:	Published an Android puzzle game called Plumber which has over 1.5 million downloads. Now sold to Keygames BV.
2010-2011:	Implemented journaling support for the MINIX filesystem, based on previous work done by Stephen Tweedie on ext3fs for Linux.
2010:	As part of the Computer Networks Practical course at the VU I wrote a TCP/IP implementation for MINIX.
2009-2010:	Wrote a driver for the Intel Pro 1000 Gigabit Ethernet adapter on the MINIX operating system.
2009:	My graduate project involves implementing a distributed, highly scalable database on XtreemOS, supervised by Guillaume Pierre.
2008-2009:	As of summer 2008, I am developing an x86 operating system. Thanks to this project, I have learned a lot about everything involved with kernel development. See my homepage and http://www.FreeNOS.org.
2008:	Developed a web-based cluster management application in PHP for my father and his collegues at Hewlett Packard (HP), which displays an overview of available machines on their network for administrative purposes.
2008:	Developed the HUbuntu Linux distribution (a HULK project), together with Coen Bijlsma and Mattijs Hoitink. Also see http://www.hubuntu.nl.
2007-2009:	Member and co-founder of the Hogeschool Utrecht Linux Kennisgroep (HULK). I am involved with several HULK projects, including system administration and development. Visit http://www.hunix.nl/ for more information.
2007-2008:	Started my own open source project, a IRC daemon called Rhino IRCD. (http://sf.net/projects/rhino-ircd)
2006:	Designed and hosted the website for Theo Miltenburg's driving school (http://www.rijschoolmiltenburg.nl)

Projects Details

Verkerk Service Systemen: Embedded Linux and Domotics, 2014-now

Verkerk Service Systemen B.V. is a medium sized healthcare solutions provider with a large customer base in the Netherlands. One of their key products is the VSS automated alarm system, which is used by patients in retirement homes to alert healthcare staff in case of emergency or when they need assistance. The VSS alarm system is composed of several types of embedded Linux devices, each providing different way to display or report alarms, such as (wireless) button or a touch display.

My job as a Software Designer at Verkerk is to gather requirements for new software functionality, design, implement, test and deliver the software to the customer. I work in a team with Hardware Engineers to develop optimal solutions for each requirement and report directly to the Project Coordinator. My responsibilities are to document the design, implementation, test procedures and test results of each of my deliveries and above all ensure maximum quality for each software component.

One of the products I worked on is the VSS SPTouch, an embedded Linux alarm device with a touchscreen display. I designed and implemented a new VoIP SIP client for the VSS SPTouch to provide realtime video conferencing using A-Law/H.264. This allows a nurse to start a realtime video conversation with her client when an alarm is raised, similar to making video calls with Microsoft Skype. The software is fully written in C++ using the Qt4 framework and is deployed at all major customers of Verkerk (200+ locations).

Another project for the VSS SPTouch I worked on is the reliability improvement of the Flash-based filesystem of the SPTouch. Before my work, Verkerk received dozens of broken SPTouch devices each month from customers which spontaneously froze at the customer site and were unable to start. Because of my thorough investigation I discovered the cause (spontaneous bit-errors due to hardware-ECC errata bug) and after extensive testing my solution was deployed at all customer sites. My solution included the migration of the JFFS2 filesystem to UBIFS, and implementing the ECC mechanism in software inside the Linux kernel. This project saved Verkerk thousands of euros and many angry customers.

A newer market for Verkerk is domotics for healthcare. The VSS HOMEbox is a novel domotics platform based on the Raspberry Pi (Raspbian Linux), ZWave, HTML5, JointJS and Verkerk specific wireless equipment (VSS-SAN). With a modern Bootstrap responsive frontend website the user can configure ZWave devices simply by click-and-drag. I have designed and implemented the complete architecture of the VSS HOMEbox. All controller software is written in C++ using the Qt5 framework and the frontend website using NodeJS, JointJS, HTML5 and Bootstrap. The whole system is fully tested, includes a unit tester and is documented extensively in source code and several documents. VSS HOMEbox devices deployed at the customer can be updated remotely (APT) and contain self-monitoring software (Monit) to ensure maximum availability. Several VSS HOMEbox devices have been deployed at a customer pilot project which already run more than half a year without any downtime or failures.

Free Niek's Operating System, 2008-2009, 2015-now

Operating systems are the core of the software stack in modern computer systems. All functions of end-user programs are eventually restricted only by the hardware and operating system. Thanks to the book "Operating Systems: Design & Implementation" by prof. Andrew S. Tanenbaum I became enthousiastic and curious about operating system internals. Therefore I decided to write my own operating system from scratch. All high-level code in FreeNOS (Free Niek's Operating System) is written in C++, because I wanted to learn the language and to experiment with it as a language for an operating system. The most recent FreeNOS v1.0.0 at GitHub is roughly comparable with the first Linux v0.01. In short, FreeNOS provides the following features:

- Virtual memory
- Simple task scheduling
- Inter Process Communication (IPC)
- VGA/Keyboard terminal consoles
- i8250/PL011 serial UART
- PCI, ATA, USB, I2C, SPI bus support drivers
- IP networking stack with UDP, ARP, Ethernet and ICMP protocols

- Filesystems include VFS, ProcFS, TmpFS, Ext2FS and LinnFS
- POSIX, ANSI C, MPI and Standard Algorithm libraries
- Dynamic and Shared memory
- Fully automated unit tester
- All sources documented with Doxygen
- User and kernel code written from scratch in C++/assembly
- Very small microkernel (2K lines)
- Builds with recent GCC, LLVM and SCons versions on POSIX systems

The system is capable of running simple UNIX commands like 'ls', 'ps' and 'uname' on the console prompt. The OS currently supports Intel Pentium x86 processors (PC) and ARMv6-ARMv7 processors (Raspberry Pi models B+ and 2). More information can be found at http://www.FreeNOS.org

ASML: Motion Control Platform (CARM), 2012-2014

The lithography machines of ASML contains many (sub)parts that need to move at extreme precise granularity in order to produce wafers. All moving parts in the NXT/NXE Twinscan machines are controlled by the CARM Motion Control Platform (CARM). CARM is a generic realtime motion control platform written in C with several subsystems including Linux, Solaris and ATCA. My role as Software Engineer in CARM involves developing realtime embedded software (C, Python, Shell) for the CG/CV connectivity layer within CARM, in particular new drivers for I/O boards with actuators and sensors in the ATCA motion control rack. I am responsible for designing, implementing, testing and documenting new drivers and extending current drivers. As Software Engineer I am reporting to the Project Leader (PL), my Team Leader (TL) and several architects. Some examples of projects I worked on are:

- SMA Try-to-concatenate & Fly-By-Repeatability Queue extension for NXT3
- SMA Board Driver Unit Tester in C and Python
- GPAS I/O board driver with ICE support in SSRLv2
- GPAB I/O board driver
- SIOB6 with MMIO GDB for 2nd Tier firmware downloads
- MPAC3 I/O board driver with ADPLL support in SSRLv3
- SMA Configurable SyncPulse Delay extension in CG/CV for NXT3+

With the delivery of the projects described above I am directly involved in the integration of this software at the customer and I provide support on earlier deliveries. Additionally I did troubleshooting and bug fixing for various components within CARM, which means I needed to analyse complex problems and errors, reproduce and finally solve those problems. My deliveries have all been integrated in the NXT1950, NXT1970 and NXE3300 Twinscan machines at Intel and Samsung.

Linnensoft: Mobile Games, 2011-2013

Mobile devices are becoming increasingly popular. Smartphones and tablets are an important piece of our every day life. Because I am enthousiastic about this trend I developed my own apps for Android, iOS and Blackberry since 2010. Additionally, I am experimented with relevant technologies, such as: Adobe AIR, Adobe Illustrator, Cocos2D-X, Box2D and Javascript/HTML5. By using AIR, Cocos2D-X and/or HTML5 I can write my app code once, and publish it on multiple platforms. A successful title I published is a simple puzzle game called "Plumber", which already received over 1.5 million downloads and ranked the top 25 games at Google Playstore in the early years of Android. Plumber was sold to KeyGames B.V. in 2015.

Rabobank: Performance Engineering, 2011

Rabobank is originally a Dutch bank which at present has customers and offices all over the world. Apart from the banking services at local offices, Rabobank also offers online banking services. Of course, Rabobank's online banking system is continuously tested and monitored for various quality aspects, such as user interaction, security, correctness and performance.

The Performance Competence Center (PCC) is responsible for analysing the performance of Rabobank systems and for solving potential performance problems. After my master graduation, I worked for several months at Rabobank as Junior Performance Engineer. It was my job to build an automatically generated VUGen script, using predefined "clickpath" documents. In short, the job included the following steps:

- reading and evaluating the "clickpath"
- ask the projectteam for any missing information
- manual validation of the testsystem
- recording the "clickpath" in Internet Explorer
- generating the VUGen script (C dialect)
- optimizing & optionally modifing the VUGen script
- testing and validating the VUGen script on testserver
- uploading the VUGen script to the load generator
- schedule the performance loadtest
- analysis of the performance results
- advise project members on the results and improvements
- documentation of performance loadtest

With every new version of the Rabobank Online Internet Banking I am responbile to analyse and qualify the whole implementation for performance. As Performance Engineer I need to correctly execute the performance tests and report the results and my advise for improvements to the project members and architects. The result of my work is incorporated in every new release of the Rabobank Online Internet Banking

prof. Andrew S. Tanenbaum (VU): MINIX on the Intel SCC, 2011

Intel is building a new experimental manycore machine in an attempt to address scalability problems on both hardware and software levels. The Single Chip Cloud Computer (SCC) is a minimal manycore prototype with 48 cores and it is quite different from conventional multicore systems. In the SCC the 48 cores are connected via a on-chip mesh network and there is no cache synchronization. In particular the cache incoherency makes the SCC very scalable, but also very complex to program for operating system designers.

My thesis for my master in Parallel Distributed Computer Systems (Vrije Universiteit, pdcs.vu.nl) was to redesign the MINIX3 operating system for the SCC, with the most efficient Inter Process Communication (IPC) as possible. I started by studying the SCC architecture and comparing with SMP multicore systems. Futhermore I changed the design of MINIX3 in order to run it on the SCC. I wrote a new bootloader in C and Assembly and designed, implemented and tested several mechanisms for IPC between processes on different cores in the SCC. The IPC mechanisms included 1-to-1, 1-to-many, many-to-1 communication, Interprocessor Interrupt and polling for notication delivery and Split/Shared-based Message Passing Buffer.

To compare the intercore IPC mechanisms I wrote a program to generate extremly high IPC load and used it to measure the performance of each IPC mechanism. Also I compared the SCC's IPC performance with MINIX on an Intel XEON SMP. Additionally, I designed and implemented various regression tests to fully validate the IPC implementation for correctness and integrity. Finally I wrote my thesis and presented my work to the MINIX3 team and Computer Systems staff. The result of my work is incorporated in the MINIX3 operating system to continue research in the area of manycore operating systems (final mark: 8).

SMARTPosition B.V.: Tracking and Tracing Software, 2010-2011

The computing era made the development possible of many consumer products, but also new technologies. Tracking & tracing equipment can measure the current location of an object on a very fine granularity. SMART-Position is a software company which specializes in developing programs for tracking & tracing products. I worked parttime at SMARTPosition during my master studies at the Vrije Universiteit Amsterdam. The job involved programming, system administration and customer support. Some example projects I worked on are:

- development of PHP/MySQL websites for customers, like adding and modifying forms
- Java indoor positioning using WiFi, mostly configuration, bugfixes and demo's to customers
- Linux server administration (Ubuntu, Apache, Postgre, JBoss, etc)
- Customer support via the main telephone and e-mail

Vrije Universiteit: Journaling Support for the MINIX Filesystem, 2010-2011

When a computer experiences an unexpected shutdown, for example due to a power failure or software crash, the filesystem should always keep it's integrity and availability. Classical filesystems require a complete filesystem scan after an unexpected shutdown to check for corruptions and errors, in order to avoid loss of data. Filesystems which support journaling do not require this step, because the operating system keeps a "logbook" with transactions at runtime.

Support for journaling is an important addition for MINIX3 to fulfill it's mission: providing a reliable and secure operating system. As my project for the course "Operating Systems Practical" I implemented journaling support in MINIX. The project involved the following tasks:

- Analysis of the extended 3 filesystem implemention in Linux
- Deciding which aspects of ext3 can be applied in MINIX
- Detailed technical design of the generic journaling library
- Using comments of David & Raja in the final technical design
- Writing the generic journaling library in C, which offers transactions for block devices (such as disks)
- Build the new JMFS filesystem (based on the existing MFS), which uses libjournal to make transactions of every write operation
- Modified the user applications of MFS to support libjournal, for example fsck and mkfs
- Designed, implemented and ran a large amount of regression tests to verify libjournal for correctness, in particular the output on block-level.

Finally, I tested and analyzed the implementation for performance and documented it in a final report. The result of my work is adopted by the MINIX3 team (final grade: 10).

Industrial TSI B.V.: Open Source Consulting and System Administration, 2009-2010

Open Source Software (OSS) is mostly available free of cost and the code can be analysed and modified by everyone. The open character of OSS can have both advantages and disadvantages. Industrial TSI offers consulting services for selecting, implementing and administrating OSS. My parttime job as Open Source Consultant included mostly (internal) system administration and advicing customers on-site. The OSS I worked with ranged from server to desktop solutions, including: Zarafa, Apache, MySQL, OpenLDAP, OpenOffice, Nagios, OTRS, SOPERA, Bacula and SugarCRM. As part of the system administration of our internal Linux servers and websites I wrote several PHP/Bash scripts, made backups, performed system maintenance and helped collegues with technical problems.

Vrije Universiteit: TCP/IP Implementation for MINIX, 2010

Networks are an essential part of modern computer systems. It is especially important to know the internals of TCP/IP in order to fully understand complex distributed systems. Therefore, for the course "Computer Networks Practical" I wrote my own TCP/IP implementation for MINIX3 from scratch. The implementation was required to pass a broad range of automated TCP/IP validation tests. To build the implementation I did the following:

- Re-read the entire TCP/IP RFC documentation
- Proposed a design for the TCP/IP implementation
- Implemented the code in C and tested it under Qemu, iteratively.
- Wrote my own TCP/IP regression tests for validating corrected in case of packet corruption, packet loss, wrong packet order and/or protocol errors.
- Communication with a real Linux system
- Tested with my own Intel Pro/1000 Gigabit network driver I made earlier in MINIX3

After finishing the code, I documented the implementation in a final report. The result of my work was graded with a 9.

prof. Andrew S. Tanenbaum (VU): Intel Pro/1000 Gigabit Network Driver for MINIX, 2009

MINIX3 (www.minix3.org) is a research operating system from the Vrije Universiteit Amsterdam with a strong focus on reliability and security. Thanks to the microkernel architecture it is possible for MINIX3 to provide higher fault tolerance compared to other operating systems. Proponents of microkernels emphasize the reliability and security advantages, while opponents argue that microkernels suffer more performance overhead. By supporting gigabit networks in MINIX3 we can demonstrate that the extra performance overhead in microkernels in negligible. For my Individual Programming Assignment (IPA) I wrote a driver in C for the Intel Pro/1000 Gigabit network adapter in the MINIX3 operating system. To accomplish that I did the following:

- Read the official Intel documentation of the Intel Pro/1000
- Analyzed the design of existing MINIX3 network drivers
- Analyzed existing Intel Pro/1000 network drivers in Linux, FreeBSD, NetBSD
- Implemented the code in C and tested in Qemu, iterativly
- Tested the driver on three real Intel Pro/1000 variants
- Benchmarked the performance of the driver and analyzed the results

In this project I was able to show that my driver can achieve gigabit network speeds and which MINIX3 component are currently a bottleneck for gigabit speeds. I documented both the implementation and my performance analysis in a final report. The result of my work is adopted in the stable version of MINIX3 and Tomas Hruby continued my work to make all MINIX3 components gigabit-ready. (final grade: 10)

Dr. Guillaume Pierre (VU): Implement Apache HBase on XtreemOS, 2009

XtreemOS (www.xtreemos.eu) is a grid operating system based on Linux, which provides virtual organizations to allow secure sharing of resources. XtreemOS incorporates a scalable distributed filesystem called XtreemFS to share large amounts of storage data in the grid.

Because XtreemFS is capable of providing such large amounts of storage data, it could be interesting for users to use it for more than just data storage, such as connecting a distributed database. Therefore, for my bachelor graduate project I implemented Apache HBase on XtreemOS. The project involved the following tasks:

- Studying, analyzing and learning all components of XtreemOS, XtreemFS, HBase
- Created a software design of the implementation for review
- Wrote the Java code and tested it on the XtreemOS testbed

- User documentation for each command (Linux manual pages)
- Performance benchmark to validate scalability
- Wrote my thesis and gave a final presentation

The result of my work is adopted by the official XtreemOS project in the source code repository (final grade: 9)

Hogeschool Utrecht: HUbuntu Linux Distribution, 2008

The Hogeschool Utrechts trains students of HBO Informatica to become all-round software engineers including knowledge of Java, HTML, PHP, MySQL, Oracle, project management, math, English, customer support and communication. Many technologies and skills are directly applicable in practice. However, a small group of students felt the need to learn more about Linux and open source software. Thus the Hogeschool Utrecht Linux Kennisgroep (HULK) was founded, to promote Linux and open source software in general within the school to both students and teachers. Thanks to the support of the Hogeschool, the HULK could also offer hosting services to students and teachers using Linux and open source for webhosting (Apache/PHP/Mysql), highly available virtualization (DRBD/Xen/iSCSI), version control (Subversion), account management (OpenLDAP) and e-mail (Postfix/Roundcube).

To futher promote Linux at the Hogeschool Utrecht, we created a modified Linux distribution especially adapted for students and teachers at the Hogeschool Utrecht. Together with Coen Bijlsma and Mattijs Hoitink I build the HUbuntu distribution based on Ubuntu. HUbuntu included custom software packages which included software for each course on the school. My tasks in this project included the following:

- Talking with students and teachers to discover their needs
- Buildings scripts with GNU Make/Bash to automatically generate new ISO images
- Designing and making Logo's for HUBuntu to replace the Ubuntu logo's
- Implemented a few software packages for courses
- Tested the ISO on different systems
- Documentation of the system, both technical and end-user
- HUBuntu CD-ROM production with printed labels

At the release of the HUbuntu CD-ROM's we gave a group presentation to the management of the Hogeschool Utrecht. After our graduation, the HUbuntu system has not been actively maintained.

Hewlett Packard: Implement Web-based Cluster Management System, 2008

Digital Corporation (presently known as HP) contributed many important technologies to modern computer systems. One of the most important technologies from Digital include their cluster computing platforms. To provide high performance and high available cluster configurations it is required to develop software suitable for such environments. The HP High Performance Software Support team helps customers with large clusters to solve technical problems.

To gain more insight in the current test cluster systems, I developed a webapplication in PHP/MySQL for Jan Linnenbank and his collegues. The system in capable to track various metrics per system, such as IP addresses, hardware settings and operating system versions. To built the application, I took the following steps:

- Gathered user requirements from Jan Linnenbank
- Made initial user designs for review by Jan
- Iteratively, I developed prototypes of the application, each time applying the feedback from Jan, until the implementation was complete
- Wrote user documentation manual
- Release of the application and maintenance

The final product is still used by the HP High Performance Support team.

Rhino IRC Daemon, 2007-2008

One of the possibilities of the internet include participating in multi-user chat sessions. There are several variants of chat networks and protocols freely available on the internet, including the Internet Relay Chat (IRC) protocol. IRC is based on simple text messages and supports clients for virtually every major operating system. Additionally, the IRC standard supports server-to-server connections, allowing IRC networks to grow significantly.

Because I was an active IRC user at the time, it seemed fun and interesting to learn more about the internals of IRC and the underlying systems, both at the client side and the server side. Since I was enthusiastic about the C programming language and wanted to deepen my knowledge in that area with a larger project. Additionally, I wanted to know more about Linux/UNIX systems, so therefore I decided to write an IRC server in C. To built the system, I took the following steps:

- iteratively (re)designed the software architecture on paper
- (re)write the implementation iteratively
- code debugging & validation using valgrind
- tested the implementation on my own computers and emulators
- wrote end user documentation
- published the code at sourgeforge

In later versions of the Rhino IRC Daemon, I implemented the GNU autotools for automatic building and used the Apache Portable Runtime (from the popular Apache Web Server) for supporting multiple operating systems. The implementation of Rhino IRCD is published on sourgeforce under the GPLv2 license.

LogicaCMG: Wireless Asset Management Monitoring, 2007-2008

LogicaCMG is an international systems integrator with more than 40000 employees in offices located all over the world. LogicaCMG Rotterdam develops the Wireless Asset Management (WAM) system. WAM can be used to track objects (assets) and measure various metrics using different types of sensors, such as warmth, pressure, gps, etc. For example, WAM can be used to track a train wagon and determine how many passangers are currently in the wagon. My traineeship at LogicaCMG included developing a monitoring system in C# for the WAM backend applications. To do that, I took the following steps:

- Created an UML software design
- Iteratively created prototypes of the implementation in C# of the monitoring application and the PHP/HTML frontend (connected via SOAP)
- Tested the application in the staging environment
- Wrote end user documentation
- Wrote a final report of my traineeship and gave a presentation to the management

The final release of the monitoring solutions was able to display the current status of all C# services, physical machines and sensors. In the web GUI the system shows an overview of all systems and SMS/mail alerts are configurable. The result of my work is adopted by the WAM project (final grade: 9)

Theo Miltenburg: Driving School Website, 2006

Since 1989 Theo Miltenburg owns a successful driving school with customers from Harmelen, de Meern, Woerden and Utrecht. Theo's customers initially found his school via newspaper advertisements and through friends and family. For Theo, the popularity of the internet could be an opportunity to further promote his driving school and find new customers. Therefore, he wants a professional website with a small budget. To build the website, I took the following steps:

- Talked with Theo to learn what kind of information he needs to publish
- Proposed various graphical designs for the website
- Iteratively implemented various prototypes of the website in PHP/MySQL, until Theo approved it for publication

- Configured DNS domain & server hosting
- Published the website and maintenance

Apart from developing the website, I maintained the website, domain and server hosting for a couple of months. Currently, Jerry van der Werf has continued the maintenance work. The website is still available at www.rijschoolmiltenburg.nl.