NIEK LINNENBANK

SENIOR EMBEDDED SOFTWARE DESIGNER

ICT High Tech
Part of ICT Group

CITY
Veldhoven

AGE
36 Years

HOBBIES
My Family, Building Voron 3D printers, QEMU, travelling. Also see nieklinnenbank.nl and FreeNOS.org

Profile

Personal goals
To excel in building embedded software in C/C++/assembly for interesting technical customers

Motivation
Developing complex software in a team makes me happy and allows me to be creative and communicative.

Domain knowledge

Medical Devices | STM32 | Linux | Voron | Operating Systems | MINIX | C/C++ | Intel | QEMU Maintainer | FreeNOS | ARM Cortex-A | ARM Cortex-M | Technical Interviewer

Relevant employers

Philips Grooming & Beauty Senior Software Designer	08-2019	Current
3T B.V. Software Designer	03-2018	07-2019
Philips Healthcare Software Designer	07-2016	01-2018

Experiences

Senior Software Designer

08-2019 Current

Philips Grooming & Beauty

Philips Lumea

Philips is a global leader in medical equipment and solutions aiming to improve the lives of 2.5 billion people a year by 2030. Personal Healthcare is a major medical market that fulfills a significant part of that mission. Intense Pulsed Light (IPL) is a key technology provided by Philips for enabling consumers to take care of their personal skin health. IPL treatments allow the safe, clean and (near) permanent removal of body hair. Philips Lumea is the brand name of several IPL-enabled hair removal devices which will become class B medical devices by 2024.

In the IPL Software Team, my role is Senior Software Designer. I am responsible for proposing and implementing new designs for the next generation IPL devices. I work in close co-operation with Hardware Engineers whom develop the IPL circuits and electronics and with our Architect for aligning design and implementation with relevant stakeholder requirements. Additionally, my day-to-day work includes guiding junior team members and my active contribution to discussions around new hardware and features. In the Software team, we work according to the Agile SCRUM / SAFE methodology and apply a strict V-model

approach to software development in our way-of-working. Over the past years at the IPL Software Team, my two main assignments were 1) the design and implementation of a re-usable, modular platform for the IPL C++ firmware 2) the creation of a new BLE-connected IPL device called SunstoneConnected and 3) the introduction of the STM32 MCU architecture to the IPL firmware platform.

Historically, the software of each new IPL product was copied from a previous model and stored into a new GIT archive. Improvements and bugfixes were frequently applied only on a subset or a single product. The state of the IPL code was thus multiple repositories with a large amount of duplicated drivers, application logic and test code which had minor but important differences on a per-product basis. My job was to unify all duplicated code into generic, re-usable modules that can be added to each product that needs it. This work involved the careful step-by-step re-design of logic, interfaces and adding configuration variables where appropriate. Every part that I generalized needed to be fully tested and proven on each product. Most important, the on-going work to bring the first class B IPL model to the market had to be undisturbed. Another essential contribution that I proposed and implemented was the introduction of the CMake build system, to support multiple product builds with flexible configuration. With the new IPL platform in place, I designed and implemented firmware for a new IPL product called SunstoneConnected that supported BLE-connectivity for use with a mobile app. This involved connectivity software on a secondary Nordic NRF-52833 microcontroller, next to our primary Atmel SAMD20J18. Reliable over-the-air upgrade with AES-encrypted upgrade packs was a primary usecase, in addition to providing (real-time) IPL data to the mobile app via the Philips specific Condor protocol on BLE. Finally, as part of the "dual source" project I added the full design and implementation of the IPL firmware based on the STM32G0B1 MCU architecture, which included new drivers for all peripherals (ADC, DAC, GPIO, UART, I2C, Timers, IRQs, Watchdog) and IEC-60335 compliancy self-tests (CPU, Flash, Memory, IRQs, Clock Frequency).

The result of my work is the creation of a re-usable, scalable and modular IPL firmware platform that supports multiple target boards and architectures. Significant amounts of product duplication are unified in a single platform, that enables low-effort creation of new product variants and allows easy platform-wide generic code changes to all targets. The Sunstone product is the first medical class B device now appearing on the market based on this platform, which will be mass produced for consumers world wide. SunstoneConnected project passed successfully the Advance Development (AD) project milestones. It has been received with great enthousiasm by our stakeholders, which were impressed with the technical capabilities, documentation and quality of the software that I delivered. Finally, the STM32 based Sunstone boards successfully passed the rigorous lifetime endurance test of 400k+ flashes and delivered the IPL business a cost saving of more than 19 million euros.

Software Designer

03-2018 07-2019

3T B.V.

ASML Real Time Integrated Computing platform (ARTIC)

ASML is market leader in building lithographic machines for mass production of various semiconductors including processors, memory modules and integrated circuits. For many years ASML machines use embedded I/O boards based on the VME standard. The roles of these embedded I/O boards include realtime sensor signal processing, actuator control and realtime computation of motion control tasks. Due to obsolescence of various electrical components at external manufacturers the existing embedded I/O boards will become end-of-life. ASML Real Time Integrated Computing (ARTIC) is a novel new platform that will replace the 16+ end-of-life boards and generalize them to only 5 generic embedded I/O boards. 3T B.V. is a supplier for ASML that will provide the full hardware and software for the ARTIC boards.

In my role as Software Designer I am responsible for the complete design, implementation and qualification of critical embedded software on the ARTIC boards. Because ASML has outsourced the full development of the ARTIC boards to 3T B.V., both my communication skills and technical documentation skills are very important. Additionally, my pro-active attitude in proposing sensible design decisions is very important for the success of the project. The software that I write in C/C++ and ARM64 assembly in this project have to adhere to the highest quality standards. The software that I write will run on all newly developed ARTIC boards, which will all be incorporated in all modern ASML machines (NXT/NXE) and will eventually have to run 24/7 in the customer factory.

The ARTIC bootloader is a vital part of each ARTIC board. Important modules of the ARTIC bootloader that I write include the full initialization of the NXP LS1046A CPU, the DDR4 training, the fully automated download of boot scripts and software images from ASML's SCH-bootserver and configuration of the Arria 10 FPGA. The ARTIC bootloader is implemented in C and ARM64 assembly and archived in GIT version control. The ARTIC bootloader is based on the open source U-Boot project, which I patched with several ARTIC specific features including SYSLOG support, Spansion S25FS512S QSPI support, and DDR4 warm reset support. In addition to the ARTIC bootloader I also developed several important embedded software modules in C for the Board Management Controller (BMC) microcontroller in the ARTIC common hardware. Several drivers and modules that I designed and implemented include the CDCI6214 clock generator, CPU watchdog, FPGA-I2C driver, TCA9546A I2C switch, TPS549D22 power converter driver, BoardLocationDataProvider and the PowerSupplyControl module.

The ARTIC bootloader is fully documented in the EDS design document, implemented in C and fully qualified in the standard TPS/TAR format of ASML. All software and technical documentation for the ARTIC bootloader comply with the standards and way-of-working of ASML. Also for the BMC I added important parts of the technical documentation in the EDS design document and TPS/TAR qualification. All current ARTIC boards AGCB/AMCB (and future boards) run on the ARTIC bootloader and are successfully integrated in the test/proto systems of multiple development departments at ASML. The ARTIC boards including the embedded software that I wrote will enter production in 2020.

Software Designer

07-2016 01-2018

Philips Healthcare

Philips Azurion iEngineCore

Philips Healthcare is a global market leader providing a large range of healthcare products for customers all over the world. One of the key technologies which Philips provides are X-ray based interventional imaging systems for use the operation room in hospitals, allowing doctors to see realtime inside the body of a patient while doing a live operation. The Philips Azurion platform is the newest interventional X-ray system developed by the Philips Image Guided Therapy department and has all the latest state-of-the-art technologies. A major software component in the Philips Azurion system is iEngineCore, which contains the software for image processing of all X-ray images, real-time display of the images on multiple monitors in parallel and also includes various frontend GUI applications used by the doctors in the operation room.

My job as a Software Designer in the iEngineCore team is to co-design and implement software solutions for the iEngineCore software component. I work together in a team of Software Designers, Engineers, Architects and Testers to develop and extend features for the iEngineCore platform. iEngineCore is a Windows based platform built using C++-11 and various custom implemented Philips frameworks (Sense, SSCF, Simplicity).

A project that I contributed to in the iEngineCore includes Picture-in-Picture support for the TSM, which is a tablet-like device for use in the operating room with a touchscreen. The Picture-in-Picture is an extension to the GUI interface to display a small scaled image next to the full-sized x-ray image. This addition is usefull in a biplane setup of the Azurion system. My work in this project was parts of the implementation and automated tests for Picture-in-Picture support.

Another feature I worked on for iEngine is Series-Flagging support, where I co-designed and implemented the final solution for Philips Azurion. In Philips Azurion, X-ray images are organized as series of images for each patient, where each serie is created for one X-ray exposure session (the docter pressing the pedal). Series-Flagging support allows the docter to mark series which are of interest, similar to a bookmark in a webbrowser. Flagged series can later be exported to a PACS server of USB.

For the newest Philips Azurion release I worked on the Raw Image Storage feature, which is about storing X-ray images which have not been processed yet by the image processing algorithms. Raw images allow easier diagnosis of faults in image processing and optimising image processing for specific usage scenarios by the docter. My job for Raw Image Storage included performance feasability measurements and analysis and co-design of the implementation in the XrayService.

Next to my work in the iEngineCore codebase I also contributed various improvements to the Jenkins continous integration test center, including fully automated endurance tests, A-versus-B performance comparison test support, automated Ethernet and GPU driver settings checks and various data analysis python scripts to check for metrics such as the most frequently occuring build issues.

Software Designer

04-2014 07-2016

Verkerk Service Systemen

VSS SPTouch, HOMEbox, VIPBOX-III

Verkerk Service Systemen B.V. is an medium sized company providing healthcare solutions to a large number of customers in the Netherlands. One of their most notable products is the VSS alarm system. The VSS alarm system is used in particular in retirement facilities to send an alarm to the nurse if an elderly person is in trouble or requires assistance. The VSS alarm system contains several types of embedded Linux devices, each providing a different way of alarm functionality, such as a wireless button or touchscreen.

In my role as Software Designer at Verkerk I gather new requirements from the customer for software functionality, create software designs for those requirements, implement, test and deliver the solution to the customer. I work in a team of Hardware Engineers to define optimal solutions for each requirement and report directly to the Project Coordinator. My

responsibilities include full documentation of the design and implementation, determining and executing appropriate test procedures and above all ensure maximum quality for each software component.

One of the products I have worked on is the VSS SPTouch, an embedded Linux alarm device with a touchscreen. I designed and implemented a complete new VoIP SIP client for realtime video-call conferencing based on A-Law/H.264 and PJSIP. With this solution a nurse can start a realtime video call with her client as a response to an alarm, very similar to Microsoft Skype. The software is completely written in C++ using the Qt4 framework and installed on all customer sizes of Verkerk (200+ locations).

In addition I have worked on significant reliability improvements of the VSS SPTouch, in particular for the NAND Flash-based filesystem. Before my work Verkerk received back several dozens of SPTouch devices each month from customers, which spontanously froze on the location of the customer and could not start up anymore. Thanks to my thorough investigation I succeeded in finding the rootcause (spontenous bit-error faults caused by a hardware-ECC errata bug). After extensive testing my solution was installed on all customer sites, which included the migration of the old JFFS2 filesystem to the newer UBIFS and complete software-based implementation of the ECC error correcting mechanism in the Linux kernel. This project saved Verkerk many thousands of euros and many angry customers.

A newer market for Verkerk is domotics. The VSS HOMEbox is a new domotics platform based on the Raspberry Pi, ZWave, HTML5, JointJS and Verkerk specific wireless protocols (VSS-SAN). With a modern Bootstrap responsive website the user can configure ZWave devices simply by clicking-and-dragging on the screen. I created the architecture, design and full implementation of the VSS HOMEbox. All control software is written in C++ using Qt5 and the frontend website is based on NodeJS, JointJS, HTML5 and Bootstrap. The HOMEbox has a unittester and is extensively documented in both source code and various documents. The VSS HOMEbox can be updated remotely (Debian APT) and contains monitoring software (Monit) for automated fault-recovery, maximizing the reliability of the system. VSS HOMEbox devices have been delivered to the customer on-site, running for more than a year without problems or downtime.

Software Engineer

04-2012 04-2014

ASML

CARM Motion Control platform

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 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 en Python

- GPAS I/O board driver with ICE support in SSRLv2

- GPAB I/O board driver

- SIOB6 met 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.

Company Owner

01-2011 01-2013

Linnensoft

Mobile games for Android, iOS, Blackberry

Mobile devices are becoming increasingly popular. Smartphones and tablets are an important piece of our every day life. Because I am enthusiastic about this trend I develop my own apps for Android, iOS and Blackberry since 2010. A successful title I published is a simple puzzle game called "Plumber", which already received over 1.5 million downloads. Additionally, I am experimenting with the following technologies:

- Adobe AIR, for developing apps with flash
- Adobe Illustrator, professional vector graphics tool
- Cocos2D-X, multi-platform game engine in C++
- Box2D, physics engine in C++
- Javascript/HTML5, using webapplications as mobile apps

By using AIR, Cocos2D-X and/or HTML5 I can write my app code once, and publish it on multiple platforms.

Junior Performance Engineer

09-2011 12-2011

Rabobank Nederland

Performance analysis of Rabobank Internet Banking

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
- 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 responsible 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

Master Student

01-2011 08-2011

prof. Andrew S. Tanenbaum

Implementing MINIX on the Single Chip Cloud Computer

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, including:

- 1-to-1, 1-to-many, many-to-1 communication
- Interprocessor Interrupt and Polling for notification delivery
- Split and Shared Message Passing Buffer (MPB)

To compare the intercore IPC mechanisms I wrote a program to generate extremely 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).

Developer

09-2010 06-2011

SMARTPosition B.V.

Building (web)applications, system administration and customer support

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. SMARTPosition 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

Master Student

06-2010 01-2011

David van Moolenbroek / Raja Appuswamy

Journaling Support for the MINIX Filesystem

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).

Open Source Consultant

08-2009 08-2010

Industrial TSI B.V.

Customer support and system administration

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 included:

- Zarafa

- Apache

- MySQL

- OpenLDAP

- Nagios

- OTRS

- SOPERA

- Bacula

- 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.

Master Student

01-2010 05-2010

David van Moolenbroek / Raja Appuswamy

New TCP/IP implementation for MINIX3

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.

Master Student

09-2009 12-2009

prof. Andrew S. Tanenbaum

Intel Pro/1000 Gigabit Network Driver for MINIX3

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)

Bachelor Student

01-2009 06-2009

Dr. Guillaume Pierre

Implementing a Large-Scale Distributed Database on XtreemOS

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)

Bachelor Student

06-2008 06-2009

Niek Linnenbank

Free Niek's Operating System

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 v0.0.4 is roughly comparable with the first Linux v0.0.1. In short, FreeNOS provided the following features:

- Intel x86 and ARM (Raspberry Pi 1 & 2)
- Virtual memory
- Simple task scheduling
- Inter Process Communication (IPC)
- SMP processing support with MPI interface (Intel only)
- VGA/Keyboard terminal consoles
- i8250 serial UART
- PCI/ATA and basic USB support
- CMOS RTC clock
- VFS, ProcFS, TmpFS, Ext2FS, LinnFS
- POSIX, ANSI C libraries
- Dynamic and Shared memory
- All sources documented with Doxygen
- Fully automated UnitTester with TAP output support
- 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 and newer. More information can be found at http://www.FreeNOS.org.

Later in 2015-2016 I revived the FreeNOS project and extended the code with a working port for the Raspberry Pi 1/2, basic IP networking and USB support and support for SMP processing with a demo MPI program which computes prime numbers in parallel.

Bachelor Student

01-2008 12-2008

Hogeschool Utrecht Linux Kennisgroep

HUbuntu Linux Distribution

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.

Web Developer

01-2008 12-2008

Jan Linnenbank

phpSystemManager

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.

Bachelor Student

01-2007 12-2008

Niek Linnenbank

Rhino IRC Daemon

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.

Bachelor Student

09-2007 02-2008

LogicaCMG

Monitoring of the Wireless Asset Management backend

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)

Web Developer

01-2006 12-2006

Theo Miltenburg

Website Driving School Miltenburg

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.

Education

HBO Informatica + Honours Programme, Hogeschool Utrecht (cum laude, avg: 8.4)

Master Parallel Distributed Computer Systems, Vrije Universiteit Amsterdam (avg: 8.3, see http://pdcs.vu....

Skill-matrix

Programming languages

.NET | Unix shell scripts | C | C# | C++ | CSS | Flash | HTML | Java | Perl | Python | SQL

Tools & Applications

UNIX commandline | Bootstrap | Git | GNU C/C++ Compiler | Jenkins | LaTeX | Make | Oracle Database | Doxygen | Subversion | GDB | GNU tools | OpenGL | PyTorch | Valgrind

OS & Platforms

Android | FreeBSD | FreeNOS (Free Niek's Operating System) | Linux | MacOS | Minix | NetBSD | POSIX | VxWorks | Windows

Hardware & Computers

Soldering & Prototyping | Nordic NRF52833 | Arduino | ARMv8 | STM32 | Atmel SAM D20 | Zilog Z8 | FPGA | Intel SCC | Oscilloscope | Raspberry Pi

Network & Protocol

ARP | DHCP | Ethernet | HTTP | I2C | IP | IPC | MPI | RS-232 | TCP | UART | UDP | USB | XML

Databases

SQL Server | MySQL | SQLite

Methods

Agile / SCRUM / SAFE | Object Oriented | Technical Documentation | Design Patterns | Test Driven Development | UML | V-Model

Other

Technical Interviewer | QEMU Maintainer | Voron 3D Printer Building

Languages

German	Basic
French	Basic
Dutch	Native
English	Fluent