NIELS REIJERS

Lead / Senior Software Developer

@ nielsreijers@gmail.com ☎ +886 975 140 428
www.github.com/nielsreijers
in www.linkedin.com/in/niels-reijers-4609602
www.nielsreijers.com

I have over 25 years' experience in IT, worked as lead developer on large projects, and have designed and implemented many systems from the ground up.

I hold a PhD in computer science from National Taiwan University and published my work in the field's top conference.

Recently I'm mostly doing commercial work, but I'm always keeping an eye on new research developments.

I have strong analytical skills and experience with a wide variety of technologies. A recurring theme in my career has been performance analysis and tuning, from tiny embedded virtual machines, to large scale databases and complex enterprise infrastructures. As hobby projects, I recently started studying category theory and Haskell, and am getting my feet wet in machine learning.

I also enjoy thinking about the softer side of software development. Are we solving the right problems? Which new technologies make us more productive, and which just swamp us with unnecessary choices? Is technology changing ever faster, or is it slowing down? And what does that mean for how we develop software?

EXPERIENCE

Lead / Senior Software Developer

DSW Health Insurance

Apr 2005 – Ongoing Schiedam, The Netherlands

Apr 2005 – Jan 2007: Software developer responsible for maintaining the pharmaceutical claims processing software.

Feb 2007 – Nov 2009: Oversaw the development of the new claims processing system as the lead developer.

- Delivered on time, it has been running reliably for 13 years and proven to be very maintainable, despite being the company's most complex system.
- It produced several spin-off products that became a company-wide standard for other teams, leading to improved efficiency and standardization.

Since 2011: After moving to Taiwan to study Chinese and pursue a PhD, I was granted the opportunity to come back to work as a senior developer for several months every year. This privilege was not typically offered to other employees, and I am grateful for this flexibility.

- Worked on and built many different systems, often with a focus on performance tuning.
- Initiated two high-impact projects:
  - The development of a system for company-wide monitoring of service calls. This provided valuable and previously unavailable real-time insight in the behavior and performance of their systems.
  - Identified a problem with the growing complexity of the deployment system. On my suggestion this was changed to an XML-based 'convention over configuration' solution. This improved maintainability and reduced the burden for developers, who no longer have to write deployments scripts.
- Rebuilt a database conversion that an external party had been working on unsuccessfully for months. My implementation took 1 week to build and ran in minutes rather than hours. This allowed us to convert our data in an acceptable time frame and the project to go live.

C# | MS SQL Server | PowerShell | Python | Jupyter

Postdoctoral Researcher

Academica Sinica, Taiwan Information Security Center

Mar 2020 – May 2021 Taipei, Taiwan

Worked with professor Yuh-Jye Lee on a project on disinformation.

- Using social connections to correct disinformation.
- Created a Chrome plugin to check social media posts with fact checkers and let the user post a comment politely linking to fact checker sites when disinformation is detected.

Golang | HTML | CSS | JavaScript | Chrome Extensions

SKILLS

Critical thinking	●●●●●
Problem solving	●●●●●
Perf. analysis and optimization	●●●●●
SQL (mainly MS SQL)	●●●●●
C, C#	●●●●●
Teamwork	●●●●○
Embedded systems	●●●●○
Scientific writing	●●●●○
Data analysis	●●●○○
Golang	●●○○○
Python, Java, PowerShell	●●○○○
Machine learning	●○○○○
HTML, CSS	●○○○○

EDUCATION

Ph.D. in Computer Science

National Taiwan University, Wireless Networking and Embedded Systems Lab

Sep 2011 – Apr 2018 Taipei, Taiwan

"CapeVM: A Fast and Safe Virtual Machine for Resource-Constrained Internet-of-Things Devices", graded A+.

Chinese Language Course

National Taiwan University, Chinese Language Division

Dec 2009 – May 2011 Taipei, Taiwan

M.Sc. in Computer Science

Delft University of Technology, Parallel and Distributed Systems Group

Sep 1995 – Jun 2002 Delft, The Netherlands

"Location tracking and group communication in FLARE", graded 9/10.

Research done at the Distributed Systems Group in Trinity College Dublin.

Postdoctoral Researcher

National Taiwan University, Wireless Networking and Embedded Systems Lab

📅 Feb 2019 – Aug 2019 📍 Taipei, Taiwan

- Worked with professor Hsueh Chih-Wen on block chain research aimed at reducing energy consumption.
- Reorganised the group meetings to better monitor the students' progress.

C · Block chain

Doctoral Candidate

Intel-NTU Connected Context Computing Center

📅 Sep 2011 – Apr 2014 📍 Taipei, Taiwan

Worked on the WuKong project, a middleware enabling high-level programming of Smart Home applications with heterogeneous architectures.

- Designed and implemented the runtime and communication protocol.
- Led the implementation of physical demos.

C · Embedded systems · Smart Home · Internet-of-Things · Z-Wave

📅 May 2014 – Apr 2018

- Developed a Java Virtual Machine for resource-constrained Internet-of-Things devices such as the Atmel ATMEGA128.
- Implemented on-device ahead-of-time compilation to native code to drastically improve performance compared to existing VMs in this class, which are one to two orders of magnitude slower than optimized C.

Improved the state of the art by:

- Improving performance to close to optimized C and thus reducing energy consumption, while maintaining platform independence.
- Providing a safe execution environment to protect devices from buggy or malicious code, at a cost comparable to existing native code solutions.

The results were published in SenSys 2018, the field's top conference.

C · Embedded systems · AVR assembly · Java · JVM bytecode

Doctoral Candidate

Delft University of Technology, Parallel and Distributed Systems Group

📅 Sep 2002 – Mar 2005 📍 Delft, The Netherlands

Worked on wireless sensor networks.

- A quantitative comparison of localization algorithms. This was later published as a book chapter.
- Developed an efficient algorithm for code distribution based on automatic patching of memory references when blocks of code move.
- Real-world radio measurements which showed the behavior at the link layer to be much more complex than the highly simplified models often used in simulations.

Supervised the lab exercises accompanying the Compiler Construction course.

C · MSP430 assembly · Wireless Sensor Networks · YACC

LANGUAGES

Dutch	●●●●●
English	●●●●●
Mandarin Chinese	●●●○○

INTERESTS

- Cycling, 5k runner
- Beginning Vipassana meditator
- Art
- Volunteer at the Rotterdam International Film Festival, before moving to Taipei
- Taiwanese History
- Prefers CDs over vinyl or streaming
- Led the winning company team in the 2013 Delft University of Technology programming contest

Research Assistant

Trinity College Dublin, Distributed Systems Group

📅 May 2001 – Dec 2001 📍 Dublin, Ireland

- Master thesis research.
- Worked in the "Anois" and "Cortex" European research projects on augmented reality games.
  - Improving GPS localization accuracy.
  - Designed a group communication protocol that allowed the game to continue when the network is partitioned, and recover when the connection is re-established.

C++ | OpenGL | GPS

Software Developer

Millidian

📅 Feb 2000 – Apr 2001 📍 Rotterdam, The Netherlands

- First developer of internet startup Millidian.
- Involved in every aspect of launching our website:
  - Developing both the front end and back end.
  - Building and optimizing the database.
  - Setting up our server infrastructure.

Visual Basic 6 | MS SQL Server | HTML | JavaScript

Software Developer

Broekhuis Solutions

📅 1998 – Jan 2000 📍 Rotterdam, The Netherlands

- Part time work during my Master degree.
- Worked on Microsoft Access projects backed by a SQL Server database.
- Database optimization was a large part of this work.

Microsoft Access | MS SQL Server

ACHIEVEMENTS

To give an impression of my work, below is a more detailed description of some past projects I'm particularly pleased with.

CapeVM

During my PhD I developed a Java Virtual Machine for resource-constrained embedded CPUs. These typically have only a few KB of RAM, and in the order of tens of KBs of flash program memory. Their extremely low power consumption allows them to run for months or years on a single charge, if programmed correctly.

Many proof-of-concept VMs have been developed for these devices, but, as interpreters, they are all one to two orders of magnitude slower than native code. This significantly increases energy consumption and thus negates the main advantage of these CPUs.

My research showed that, even with so little resources, it is possible to translate Java bytecode to native code at load time, on the device. With a number of carefully designed optimizations, this leads to performance several times faster than existing VMs, and within half that of optimized native C.

Besides platform independence, the resulting VM can also provide a safe execution environment and protect the device from buggy or malicious code, at a performance cost that is on-par with or lower than existing native code solutions.

The paper describing the final result was accepted to SenSys 2018, the top conference in the field with a 16% acceptance rate.

Health insurance claims processing

After working for DSW Health Insurance for almost two years, I was made lead engineer for their new claims processing system, which at the time was done by a number of different systems. This project was part of the plan to migrate all legacy systems to .Net.

I led a team of 5 engineers to implement this system, designing many of the core parts of the system myself. The first version, initially processing claims for a single, but complex, type of health care was delivered in just over 2,5 years.

It has since been extended to replace all other claims processing systems. Over the past 13 years it has proven to be very maintainable, requiring no major refactoring, despite being the company's largest and functionally most complex systems.

When we started this project, the company had only recently started using .Net and much of the infrastructure needed to build a system like this was missing. As a result, we produced several spin-off projects that were subsequently adopted as standards by the rest of the company, many of which are still in use today:

- **Background processing engine**: To automatically process incoming claims and several related processes, we developed a uniform way to define these processes, handle incoming tasks, set priorities, configure worker threads, handle errors, etc.
  This engine has since been generalized and made the standard to implement background processing for all other projects within the company.
- **Standard set of code generation templates**: These generate the database and data access layer, which enforced a level of consistency across projects. It also made it easy to generate other features from the entity definitions, such as views to hide privacy sensitive information when developers access production data, or an in-memory database which greatly sped-up unit testing.
- **Simple workflow system**: What pleased me about this spin-off in particular, is how the company was about to start down a traditional waterfall process asking users for the requirements for the new workflow system. I believed this would lead to an unnecessarily complex solution, while a simple to-do list would be sufficient for us.
  This intuition proved to be correct, and we implemented a simple approach with just two tables: Task and TaskType, which has since been used by all subsequent .Net projects with only minor extensions.

Efficient code distribution in wireless sensor networks

While doing my research on wireless sensor networks, reprogramming them by connecting a cable to each device quickly became tedious, so I developed a way to reprogram our nodes remotely.

I developed a simple diff language to update the code using the binary image already present on the device. This required only a fraction of the bandwidth that would be needed to transmit a complete new image.

Similar to my work on CapeVM, a few binary optimizations were important. The key realization was that, after a small code change, most of the resulting binary code is identical, except for a shift in addresses. By including a short list of address ranges and offsets, the receiving node could patch all memory accesses automatically, significantly reducing the size of the patch script.

To date this is my most cited research paper.

Automated deploys

DSW Health Insurance already had an automated deployment system running for a number of years which deployed 200+ modules to a range of target environments. In this system each module provided PowerShell scripts to deploy their code.

It quickly became clear to me that this approach would become unmaintainable over time as the deploy scripts, often copy-pasted from one project to another, were starting to diverge and cause conflicts.

Therefore, I proposed switching to an approach where teams specify what to deploy in a centrally stored XML file, and the deployment system deploys this following fixed conventions that can only be overridden where necessary.

The key insight was that since the deployment system is completely under our own control, it can be very limited in the functionality it offers. If needed, new functionality can be easily added, but ideally there should only be a single way to deploy similar artifacts.

This approach had several advantages:

- It relieved teams from having to write the code to deploy their application.
- It led to greater consistency in how systems are deployed.
- Deployed items are named consistently, making them easier to find and identify.
- The central repository of xmls provides a global view of what is being deployed.
- By modifying the scripts, the way things are deployed can be changed globally, for all modules at the same time. For instance, this has been used to switch to encrypted database connections without requiring any action from development teams.

Service call tracing

While working on designing an automated system to do integration testing of the company's complicate application landscape, it became clear there was no insight into what service calls were being made between systems. This meant we did not know what scenarios should be tested to achieve good test coverage.

At the time the company was almost exclusively using Windows Communication Foundation (WCF). To get an objective measurement of which systems were communicating, I developed a tracing tool that settles into the WCF stack and log all calls sent to, or originating from a server. By deploying this on both the production and testing environment and comparing the data, we were able to determine the test coverage of our integration tests.

Besides being useful to determine test coverage, the previously unavailable insight obtained from this tracing soon proved extremely valuable to monitor performance and analyze production issues.

Later, correlation ids were added to outgoing requests, which allows us to follow a flow over multiple systems, and we added tracing of several other events such web API calls, service bus messages, and the processing engine described earlier.

Although more limited in scope, it often provided insights that could not have been obtained from far more expensive solutions like Dynatrace.

PUBLICATIONS

Journal Papers

Aug 2019 **Improved Ahead-of-Time Compilation of Stack-Based JVM Bytecode on Resource-Constrained Devices**
Niels Reijers and Chi-Sheng Shih, ACM Transactions on Sensor Networks

May 2018 **Making sensor node virtual machines work for real-world applications**
Niels Reijers, J. Ellul and Chi-Sheng Shih, Embedded Systems Letters

Aug 2003 **Distributed Localization in Wireless Sensor Networks: A Quantitative Comparison**
Koen Langendoen and Niels Reijers, Computer Networks (Elsevier), special issue on Wireless Sensor Networks

Conference Papers

Nov 2018 **CapeVM: A Safe and Fast Virtual Machine for Resource-Constrained Internet-of-Things Devices**
Niels Reijers and Chi-Sheng Shih, Proceedings of the 16th ACM Conference on Embedded Networked Sensor Systems (SenSys)

Feb 2017 **Ahead-of-Time Compilation of Stack-Based JVM Bytecode on Resource-Constrained Devices**
Niels Reijers and Chi-Sheng Shih, Proceedings of the International Conference on Embedded Wireless Systems and Networks (EWSN)

Feb 2013 **Design of an Intelligent Middleware for Flexible Sensor Configuration in M2M Systems**
Niels Reijers, Kwei-Jay Lin, Yu-Chung Wang, Chi-Sheng Shih, and Jane Y. Hsu, Proceedings of the 2nd International Conference on Sensor Networks (SENSORNETS)

Aug 2013 **Building Smart M2M Applications Using the WuKong Profile Framework**
Kwei-Jay Lin, Niels Reijers, Yu-Chung Wang, Chi-Sheng Shih, and Jane Y. Hsu, IEEE International Conference on Green Computing and Communications and IEEE Internet of Things and IEEE Cyber, Physical and Social Computing

Dec 2011 **Building Intelligent Middleware for Large Scale CPS Systems**
Niels Reijers, Yu Chung Wang, Chi-Sheng Shih, Jane Y. Hsu and Kwei-Jay Lin, Proceedings of the 2011 IEEE International Conference on Service-Oriented Computing and Applications (SOCA)

Oct 2004 **Link Layer Measurements in Sensor Networks**
Niels Reijers, Gertjan Halkes and Koen Langendoen, First IEEE International Conference on Mobile Ad hoc and Sensor Systems (MASS)

Sep 2003 **Efficient Code Distribution in Wireless Sensor Networks**
Niels Reijers and Koen Langendoen, Second ACM International Workshop on Wireless Sensor Networks and Applications (WSNA)

Apr 2002 **Using group communication to support mobile augmented reality applications**
Niels Reijers, Raymond Cunningham, René Meier, Barbara Hughes, Gregor Gärtner and Vinny Cahill, Proceedings Fifth IEEE International Symposium on Object-Oriented Real-Time Distributed Computing (ISORC)

Workshop

Mar 2014 **Intelligent Middleware for Large Scale Cyber-Physical Systems**
Chi-Sheng Shih and Niels Reijers, half day workshop at 29th ACM Symposium On Applied Computing (SAC), Gyeongju, Korea

Book Chapter

Aug 2005 **Distributed Localization Algorithms**
Koen Langendoen and Niels Reijers, in Embedded Systems Handbook, R. Zurawski (editor), CRC press