David Nilsson

Kungshamra 48
Solna, 17070, Sweden
+46761162066
nilsson.dd@gmail.com

Education

2012-now **MSc in Computer Science (theoretical track), KTH**, *Stockholm, Sweden*.

2009-2012 **BSc in Computer Science , KTH**, *Stockholm, Sweden*.

2006-2009 **Technical track, High School**, *Oskarshamn, Sweden*.

Knowledge

Languages C, C++, Haskell, Java, Python, Verilog

Areas Low-level software development, theoretical computer science, security, functional programming

Architectures X86, AVR, FPGA

Work Experience

September 2014 – now **Software Engineer**, *Microsoft*, Prague.

- Software (development) engineer in backend components for Skype
- Working daily on developing and maintaining large scale backend services
- Primarily dev in C++ but role covers design, implementation, review and testing

January 2014 – June 2014 **Master's thesis**, *Optistring Technologies AB*, Stockholm.

- Researched fault detection in solar panels
- Studied areas in anomality detection and degradation measurements
- Innovated methods for online-detection and classification of partial shading
- Link to publication record

February 2012 – September 2013 **Embedded developer**, *Optistring Technologies AB*, Stockholm.

- Focus on developing next-generation solar inverters in a small but highly skilled startup.
- Design, implementation, verification, and troubleshooting of embedded systems.
- Also work with connecting systems, such as embedded linux and data feeds in python.

January 2012 – February 2012 **Consultant in Embedded development**, Stockholm.

- Design and implementation of embedded software, libraries och visualization tools.
- Work with RF connectivity and various sensors.

Selection of Projects

non-public **PV inverter software**, *Optistring Technologies AB*, ~40k LOC.

Optistring is a swedish startup building next-generation solar power inverters. I entered the company when it was time to build a first prototype for real solar installations. During my time at the company I was the primary developer of the embedded systems, implemented in C. The work covered everything from design, implementation, troubleshooting and such, essentially taking responsibility for part of a product. The system requires several components in order to deliver power output on the grid, such as low-level realtime communication, synchronization at several stages, and handling of various fault conditions. While my primary role was embedded development, I was also heavily involved in developing solutions for streaming logging data, handling databases, and building solutions around embedded Linux systems.

public **Distributed approach to BTC markets**, *Own project*, ~2k LOC.

Project written in Haskell focusing on retrieving bitcoin (digital currency) market feeds. Bitcoin markets have publically available APIs providing orderbook and trade history. This project solves the problem of reliably fetching market data while circumventing regional issues, such as connectivity problems. It is built up around a "proxy layer" which consists of public-facing nodes pooled together. These nodes connect to a backend which handles health measures (such as node load) and regularly transmits requests to the proxy layer. Markets are dynamically loaded in the backend during runtime and then polled regularly through the proxy layer, with results saved in a distributed database (Cassandra). This approach also provides benefits such as dynamic scaling and easy maintenance, since proxy nodes may be removed or added at any time.

non-public **Embedded software for motion platform**, *Consultant work*, ~5k LOC.

Embedded software written in C for sampling motion sensors (accelerometers and gyroscopes) over serial interfaces, and performing RF transmissions to a host application. The data stream was then visualized with various example applications using a library implemented in C++. This project gave some basic experience in embedded development and insight into issues such as power management on constrained systems.

public **Distributed SDE solver**, *Own project*, ~1k LOC.

Distributed approach to solving Stochastic Differential Equations implemented in Haskell. Exposes a high-level interface for composing equations and parameters together with some method of computation, typically distribution over a cluster (using MPI) and a local multi-threaded solver. This project was an experiment into distributed computations and type-level expressibility of complex programs in Haskell.

public **Kattis client**, *Own project*, ~1.5k LOC.

CLI tool used to interface with the Kattis online judge system, which is widely used at KTH and swedish programming competitions. The project was implemented in Haskell and wraps a web client internally in order to automize several tasks and offer a clean and easy interface for the user. It gave insight into functional patterns surrounding complex code and scalable error handling.

public **Reliable blind SQL injections**, *Own project*, 200 LOC.

Example of applying theoretical concepts onto existing problems. Blind SQL injections (injections lacking visible output) are typically detected by sending probes and measuring round-trip times. This project is an implementation in C++ (and blog post) applying bootstrap methods from statistics to send approiate probes, but also gain confidence in results. Eventually ended up in production at a swedish startup building security scanners.

Languages

Fluent in both English and Swedish.

Other

- Portfolio with open source projects - `https://github.com/davnils`
- Blog on computer science - `http://davnils.github.com`