Duarte Nunes

Software Engineer

☎ +55 19 98769-1733
✉ duarte.m.nunes@gmail.com
github.com/duarten
@duarte_nunes
January 19, 1986

Summary

Software engineer with a strong background in systems programming. Currently immersed in distributed systems and high-performance, low-latency software.

Experience

Apr 2019 – Nov 2019 **CTO**, *vectorized.io*.

Developed the initial infrastructure for Redpanda, a Kafka API-compatible, transactional, and safe queue, intended for massive data streams. It is built on top of the Seastar project, an advanced C++ framework for high-performance server applications. Redpanda has no dependencies on external systems, provides over 10x more throughput than the competition at predictable latencies, and allows for proportional cost savings.

Mar 2016 – Apr 2019 **Software Engineer**, *ScyllaDB*.

Worked on ScyllaDB, a high-performance NoSQL database compatible with Apache Cassandra, optimized for modern hardware, and the underlying Seastar project, an advanced C++ framework for high-performance server applications.

Developed several cross-cutting features, including: range tombstones; thrift support; materialized views; Raft and lightweight transactions.

As a maintainer, reviewed contributions to multiple components of the system, enforcing correctness and design constraints.

Aug 2013 – Mar 2016 **Software Engineer**, *Midokura*.

Worked as a software engineer on MidoNet, a virtual network platform for IaaS clouds. Focuses on the architecture and implementation of the core network controller, a distributed and highly concurrent software written in Java and Scala. Responsibilities include:

- defining and implementing network controller features (e.g., resource protection, cross-site peering, L4 load balancing, etc.),
- defining and implementing cross-controller features (e.g., distributed flow state management and replication, port migration, NAT port reservation, etc.)
- performance and scalability improvements (e.g., measuring and accelerating the packet processing pipeline, reducing memory footprint, scaling the on-ramp and off-ramp gateways, re-write of the userspace netlink library, etc.)

Jun 2012 – Jul 2013 **Software Engineer**, *Nokia Siemens Networks.*

Worked as a software engineer and architect on large-scale brownfield Java projects concerning the planning and management of large multi-layer optical networks. Responsibilities included: enforcement of architectural constraints, code review, hands-on knowledge sharing workshops (mainly about concurrent programming and software development), design and implementation of infrastructure modules. Lead the engineering of a concurrent and efficient component to mediate between the planning and evolution of a network and the real-time events generated by its physical, deployed representation.

Jan 2009 – Aug 2013 **Software Engineer**, *CCISEL.*

Member of the CCISEL group at ISEL, doing research, development, education and training. Examples of activities:

2009 - 2013

Did research on concurrent programming and operating systems (mainly Windows Research Kernel); participated in the development the SlimThreading framework, containing synchronization algorithms;

2011-2012

Lectured at the PROMPT post-graduation on the web services and concurrent programming modules;

2011

Did consulting for Talaris, producing a Windows device driver for proprietary ATM hardware;

2010

Gave training at EID during a four-week course on C++ programming tailored for embedded systems programming;

2010

Did consulting for Talaris working on the adaptation of a specialized and proprietary Win32 library for the Linux operating system.

Sept 2011 – Jun 2012 **Software Engineer**, *SAPO, Portugal Telecom.*

Worked as a Software Engineer, employing .NET technologies, on the Service Delivery Broker, used internally at Portugal Telecom and at various customers. Mainly worked on the engine that exchanges the messages between the client and the service, providing cross-cutting features such as: data transformation, routing, authentication, authorization, throttling, load balancing, protocol bridging, caching, etc. Refactored the core runtime towards a parallel and asynchronous implementation and developed the distributed throttling and caching features.

Sept 2010 – Nov 2011 **Teacher Assistant of Computer and Software Engineering**, *ISEL.*

Worked as a teacher assistant for software engineering courses, namely on Concurrent Programming, a 5th semester course where students learn threads (using a C implementation of green threads), Java and .NET synchronization, memory models, asynchronous programming and concurrent programming models, and on Software Laboratory, a 4th semester course where students develop a fully-featured Java web application during the whole semester.

May 2011 – Aug 2011 **Independent Contractor**, *Google*.
In the context of the 2011 edition of Google Summer of Code, successfully developed the project "SlimThreading on Mono" with the purpose of enhancing the threading infrastructure of the Mono open source project using a novel synchronization framework (SlimThreading), which, besides being highly efficient, significantly reduces the dependency on operating system services; also refactored the virtual machine implementation of intrinsic locking.

Speaking Engagements

Fev 02, 2019 **Raft in Scylla: Consensus in an eventually consistent database**, *FOSDEM*, Brussels, Belgium.
This talk will cover the characteristics and requirements of Scylla's Raft implementation, how it enables strongly consistent updates, and how it improves the reliability and safety of internal processes, such as schema changes, node membership, and range movements.

Eventually consistent databases choose to remain available under failure, allowing for conflicting data to be stored in different replicas (later repaired by background processes). Weakening the consistency guarantees improves not only availability, but also performance, as the number of replicas involved in a given operation can be minimized. There are, however, use-cases that require the opposite trade-off. Indeed, Apache Cassandra and Scylla provide Lightweight Transactions (LWT), which allow single-key linearizable updates. The mechanism underlying LWT is asynchronous consensus in the form of the Raft algorithm. In this talk, we'll describe the characteristics and requirements of Scylla's consensus implementation, and how it enables strongly consistent updates. We will also cover how consensus can be applied to other aspects of the system, such as schema changes, node membership, and range movements, in order to improve their reliability and safety. We will thus show that an eventually consistent database can leverage consensus without compromising either availability or performance.

Nov 07, 2018 **Consensus in Eventually Consistent Databases**, *Scylla Summit*, San Francisco, USA.
Eventually consistent databases choose to remain available under failure, allowing for conflicting data to be stored in different replicas (later repaired by background processes). Weakening the consistency guarantees improves not only availability, but also performance, as the number of replicas involved in a given operation can be minimized. There are, however, use-cases that require the opposite trade-off. Indeed, Apache Cassandra and Scylla provide Lightweight Transactions (LWT), which allow single-key linearizable updates. The mechanism underlying LWT is asynchronous consensus. In this talk, we'll describe the characteristics and requirements of Scylla's consensus implementation, and how it enables strongly consistent updates. We will also cover how consensus can be applied to other aspects of the system, such as schema changes, node membership, and range movements, in order to improve their reliability and safety. We will thus show that an eventually consistent database can leverage consensus without compromising either availability or performance.

Oct 24, 2017 **Distributed Materialized Views**, *Scylla Summit*, San Francisco, USA.
Materialized views are tables that store, redundantly, a subset of the columns from a given base table. The partitions in the view typically have a different key, allowing for queries unsupported by the base table.

Supporting materialized views in a multi-master, eventually consistent store such as Scylla presents some challenges. In the absence of a single master serializing updates to each partition, we need a decentralized way of incrementally maintaining materialized views. A solution mustn't negatively affect cluster availability and request latency, as these are important characteristics of Scylla.

Jun 09, 2017 **Scylla: Faster than a Speeding Byte**, *TECHINPORTO*, Oporto, Portugal.
Scylla is a NoSQL database providing Apache Cassandra compatibility, distinguishing itself by supporting millions of operations per second, per node, with predictably low latency, on similar hardware. This talk will introduce Scylla, highlight some of its achievements, and lift the veil on the many design decisions, from programming model to the low-level, mechanical sympathetic details of its memory allocators.

May 19, 2017 **ScyllaDB: NoSQL at Ludicrous Speed**, *J On The Beach*, Malaga, Spain.
ScyllaDB is a NoSQL database providing Apache Cassandra compatibility, distinguishing itself by supporting millions of operations per second, per node, with predictably low latency, on similar hardware. This talk will cover the design decisions and techniques that enable such an achievement.

Jan 2016 **Challenges in Distributed SDN**, *FOSDEM*, Brussels, Belgium.

Nov 2015 **MidoNet and the Open vSwitch Datapath**, *Open vSwitch 2015 Fall Conference*, San Jose, California, USA.

Sept 2015 **Challenges in Distributed SDN**, *LinuxCon Europe*, Dublin, Ireland.

Education

2005 – 2009 **BS, Computer and Software Engineering**, *ISEL - Instituto Superior de Engenharia de Lisboa, Portugal.*

Thesis

title Concurrent Programming Infrastructures

description Obtained the highest grade (20/20) for the final thesis, a project on concurrent programming infrastructures, where he built a user mode scheduler based on the Windows 7 user mode scheduling API and devised a novel technique for user-mode blocking and seamless synchronization of user-mode threads with regular NT threads.

Skills

Disciplines	Concurrent programming, kernel programming, computer networks, virtualization, distributed systems
Programming Languages	C, C++, C#, Rust, Clojure, Java, Scala

Languages

Portuguese	Native or bilingual proficiency
English	Native or bilingual proficiency
Spanish	Professional working proficiency
German	Limited working proficiency
French	Limited working proficiency