SCIENCES & BIOENGINEERING SCIENCES

The Research Group
Software Languages Lab

has the honor to invite you to the public defence of the PhD thesis of

Mathijs Saey

to obtain the degree of Doctor of Sciences

Title of the PhD thesis:

Modular Distribution Strategies in Stream Processing Frameworks: a Meta-Programming Approach

Supervisor:
Prof. dr. Wolfgang De Meuter (VUB)
Co-supervisor:
Prof. dr. Quentin Stiévenart (Université du Québec à Montréal, CA)
The defence will take place on
Monday, September 21, 2026 at 5 p.m.

VUB Etterbeek campus, Pleinlaan 2, Elsene,
In auditorium I.0.01

Members of the jury

Prof. dr. Ann Nowé (VUB, chair)
Prof. dr. Bas Ketsman (VUB)
Prof. dr. Pieter Libin (VUB)
Prof. dr. Annette Bieniusa (RPTU Kaiserslautern-Landau, DE)
Dr. Pascal Weisenburger (Universität St. Gallen, CH)

Curriculum vitae

Mathijs Saey obtained his Masters degree in Computer Science at the Vrije Universiteit Brussel, magna cum laude. He started his PhD at the Software Languages Lab under the supervision of Prof. Dr. Wolfgang De Meuter. During his PhD, he contributed to the "Flamenco" and "Ecopipe" research projects. He is currenty working on the "Basecamp Zero" project for automated testing. His research interests lie in language design and distributed programming. He coauthored three international journal papers (one as first author), two international workshop papers (both as first author) and presented his work at several international conferences and workshops. He supervised 5 Bachelor theses and 7 Master theses.

Abstract of the PhD research

Our world is populated by billions of internet-connected devices which are constantly producing events and sending them to organisations, receiving them as high-volume data *streams*. Many organisations wish to respond to this data with as little delay as possible. *Distributed Stream Processing Frameworks* (DSPFs) are frameworks that facilitate programming these types of applications, which have to handle enormous volumes of data in real time. These frameworks enable applications to scale horizontally by distributing them over several machines connected in a *cluster* configuration.

Crucial for the performance of such applications is the *distribution strategy* that is used to partition data over the cluster nodes. In some DSPFs, like Apache Spark or Flink, this distribution strategy is hardwired into the framework, which can lead to inefficient applications. The other end of the spectrum is offered by Apache Storm, which offers a low-level model wherein programmers can implement distribution strategies on a per-application basis to improve efficiency. However, this model conflates distribution and data processing logic, making it difficult to modify either. As a consequence, today's cluster application developers either have to accept the built-in distribution strategies of a high-level framework or accept the complexity of expressing a distribution strategy in Storm's low-level model.

In this dissertation, we propose Skitter: a novel programming model which uses meta-programming to cleanly disentangle the data processing of a stream processing application from its distribution logic, and to enable the creation of novel distribution strategies in a modular fashion. This enables the expression of distributed stream processing applications in a high-level framework while still retaining the flexibility offered by Storm's low-level model.

We introduce a formal description of Skitter, called featherweightSkitter, and a practical implementation of Skitter as a DSPF, called Skitter.ex. FeatherweightSkitter is used in conjunction with featherweightStorm, a novel formal description of Storm, to prove that any application expressed in Storm can be translated into an equivalent application expressed in featherweightSkitter, showing that the introduced abstractions are "Storm-complete". Skitter.ex is used to show that Skitter enables the implementation of existing distribution strategies from the state of the art in a modular fashion. Our performance evaluation of these strategies shows that they exhibit the expected performance characteristics and that applications written in Skitter.ex obtain throughput rates in the same order of magnitude as Storm.