Master 2 Internship 2026

Optimizing Inter-Function Communication in Serverless Systems

Supervisors: Alessio Pagliari, Marc Shapiro
Contacts: alessio.pagliari@lip6.fr, marc.shapiro@acm.org
Location: LIP6, DELYS Team, Campus Pierre et Marie Curie, 4 Place Jussieu, 75005 Paris

Start: Around February/March 2026
Duration: 6 months

Context

The "serverless" paradigm [1] allows executing functions (small pieces of code) on demand, without managing servers. A platform automatically launches a function when an event arrives, executes the code, returns a result, and then releases the resources. This ephemeral nature facilitates scaling up and down. Platform examples: AWS Lambda[1], Google Cloud Functions[2], and open-source solutions deployable locally such as OpenFaaS[3] or OpenWhisk[4]. Serverless has shown good results for batch data processing, for example MapReduce-style workloads [2]. However, more and more use cases require real-time processing (IoT, monitoring, e-commerce), where data arrives continuously and must be processed without delay by pipelines of interconnected functions [3].

Motivation and challenges. To compose processing pipelines (dataflow [3]), current solutions rely on external messaging systems (Kafka, cloud queues). This approach adds latency (tens to hundreds of ms per hop), requires heavy infrastructure, and imposes costly serialization/deserialization. Direct communication between functions (sockets, channels) would reduce latency and eliminate external dependencies. However, this raises major system challenges that this internship proposes to explore.

Objectives

After a state-of-the-art study, the main objective will be to design and prototype a direct communication system between serverless functions. The student will tackle one or more of the following system challenges:

- **Lifecycle and discovery:** How to synchronize the startup of these ephemeral processing chains: connection requests to services still starting up, termination detection…
- **Scaling and routing:** When functions are replicated, how to balance routing while maintaining processing consistency when certain functions are stateful (i.e. they maintain data between processing events)?
- **Reliability:** How to guarantee the complete functioning of the chain if we consider that functions are not reliable? What mechanisms could be put in place to ensure chain detection and recovery?
- **Performance:** What system optimizations can be implemented to limit the overhead due to this FaaS architecture while achieving latency and throughput compatible with stream computing application requirements (memory mapping, zero-copy algorithms, thread placement…)?

Concretely, the work will include implementing a direct connection mechanism between functions, with an orchestrator that manages the deployment and discovery of complex serverless applications. The final prototype must at least enable communication between multiple functions in a simple pipeline deployed on an open-source serverless platform.

Technologies: open-source platforms (OpenFaaS/OpenWhisk), system extensions in Go/Scala/C/Rust for low-level mechanisms, functions in Python/Go/Java.

Prerequisites

Minimal:

- Fundamentals of distributed systems (consistency, fault tolerance, partitioning).
- System programming (e.g. sockets, threads, Docker containers).
- System and kernel resource management (memory, scheduling…).
- Comfortable with at least one language (C/Go/Python/Java).

Desired (but can be learned during the internship if needed):

- Streaming architectures (Kafka, Flink).
- Notions of consistent hashing, load balancing, service discovery.

[1] https://aws.amazon.com/lambda/
[2] https://cloud.google.com/functions
[3] https://www.openfaas.com/
[4] https://openwhisk.apache.org/

Bibliography

[1] P. Castro, V. Ishakian, V. Muthusamy, and A. Slominski, "The rise of serverless computing," *Communications of the ACM*, 2019.

[2] D. Barcelona-Pons, P. Sutra, M. Sánchez-Artigas, G. Par\ís, and P. Garc\ía-López, "Stateful serverless computing with crucial," *ACM Transactions on Software Engineering and Methodology (TOSEM)*, 2022.

[3] Z. Li, C. Xu, Q. Chen, J. Zhao, C. Chen, and M. Guo, "Dataflower: Exploiting the data-flow paradigm for serverless workflow orchestration," in *Proceedings of the 28th ACM International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS)*, 2023.