SYMBIOMON: A High Performance, Composable Monitoring Service for Online Application Introspection and Adaptation

Srinivasan Ramesh* Robert Ross† Matthieu Dorier† Allen Malony* Philip Carns† Kevin Huck*
*University of Oregon {sramesh, malony, khuck}@cs.uoregon.edu
†Argonne National Laboratory {rbross, carns}@mcs.anl.gov, mdorier@anl.gov

***Abstract*—The construction of high-performance scientific software has shifted from a traditional monolithic message-passing interface (MPI) executable model to a coupled, services-style model in which simulations run alongside a host of distributed HPC data services within the same batch job allocation. Performance analysis of such a distributed system requires collecting, monitoring, aggregating, and exporting performance data from multiple sources online. We propose SYMBIOMON, a low-overhead monitoring service built by composing high-performance microservices. Compared to other monitoring services, SYMBIOMON stands apart in its choice to build and extend components developed using an existing HPC framework instead of building everything from scratch. This design choice ensures that SYMBIOMON is more maintainable, offers significantly more flexibility in resource allocation, and enables the user to toggle functionality as needed freely. SYMBIOMON is currently available online for experimental use.**

***Index Terms*—microservices, monitoring, performance, analysis**

I. Summary

The construction of high-performance computing (HPC) software is undergoing a significant shift from a monolithic message-passing interface (MPI) model to a services-style approach in which several distributed components interact to make progress towards a common scientific goal. The services-style approach is desirable when specialized components are owned, built, and deployed by different scientific teams collaborating on a large project. Machine learning (ML) ensembles, data analysis tasks, visualization tasks, and MPI simulations are examples of these specialized distributed components. Data services play an increasingly vital role in coordinating data exchange, storage, and analysis operations between these distributed components while simultaneously abstracting the complexity of the underlying heterogeneous HPC storage stack.

The Mochi project [1] at Argonne National Laboratory is a multi-year, multi-institution effort aimed at building customizable HPC data services by adopting a composition model to build advanced functionality. Specifically, Mochi services use microservice building blocks to enable the rapid development of customizable distributed services. Concerning the service configuration, the microservice model offers significant flexibility to the user. Two service configurations that are functionally alike can differ vastly in their performance characteristics. Because data services are intimately coupled with MPI simulations and other distributed components, an optimal data service configuration is critical to ensuring the good overall performance of the workflow. It is not easy to estimate the optimality of the current data service configuration, nor is it always clear as to how to improve it.

Due to a large number of moving parts and a high degree of concurrency involved in an HPC environment, observing and reasoning about the performance of a data service configuration requires the collection, aggregation, and online analysis of performance data from multiple sources that include the data service components and the MPI simulation. Further, this data collection and analysis must be enabled while ensuring low operating overhead. In this work, we propose SYMBIOMON, a *composable*, high-performance monitoring service that builds and extends existing Mochi components where necessary, as opposed to building every component from scratch. SYMBIOMON's composition model is attractive for several reasons. First, by clearly separating monitoring and analysis microservice components, SYMBIOMON allows users to toggle functionality as required online. Second, by building upon an existing high-performance RPC framework (Mochi), SYMBIOMON automatically inherits the HPC-platform-specific optimizations applied to Mochi's networking and concurrency layers. Third, composition makes SYMBIOMON arguably more maintainable when compared to other state-of-the-art monitoring service implementations.

SYMBIOMON comprises three core components — the COLLECTOR, AGGREGATOR, and REDUCER microservices. The COLLECTOR exposes the core metric collection API. SYMBIOMON metrics are internally represented as *time-series* event data that can be associated with unique *taglists*. Time-series event data collection is a "cloud-like" approach to service monitoring that offers insight into various types of performance anomalies. However, when applied to extract sub-millisecond events in a large-scale HPC environment, the volume of data collected can quickly overwhelm the storage system. To manage this problem, SYMBIOMON exposes a set of reduction operations. COLLECTOR instances connect to a distributed AGGREGATOR service to store the locally reduced values. The REDUCER instance can perform a second, global reduction on the locally reduced values if needed.

While SYMBIOMON is primarily designed to generate more optimal Mochi data service configurations, we also demonstrate the seamless integration of its online monitoring capabilities to any traditional MPI-based HPC application. Specifically, this is achieved by the use of a TAU [2] plugin. The plugin converts TAU performance data and events

to the corresponding SYMBIOMON metric type. A local COLLECTOR instance on the plugin acts as the access point to the AGGREGATOR service that may be shared by other distributed components running as a part of the same batch job.

REFERENCES

[1] R. Ross, G. Amvrosiadis, P. Carns, C. Cranor, M. Dorier, K. Harms, G. Ganger, G. Gibson, S. Gutierrez, R. Latham *et al.*, "Mochi: Composing data services for high-performance computing environments," *Journal of Computer Science and Technology*, vol. 35, no. 1, pp. 121–144, 2020.

[2] S. Shende and A. Malony, "The tau parallel performance system," *The International Journal of High Performance Computing Applications*, vol. 20, no. 2, pp. 287–311, 2006.