Exploring and Improving Design of Abaco with TLA+

Smruti Padhy, Joe Stubbs
Texas Advanced Computing Center
University of Texas at Austin
e-mail: [spadhy, jstubbs]@tacc.utexas.edu

Abstract

Actor Based Containers (Abaco) is an open-source, distributed computing platform for users to compute atomic, independent workloads, or functions, on cloud infrastructure. It provides a web-based Application Programming Interface (API), and Functions-as-service hosted at the Texas Advanced Computing Center at the University of Texas at Austin. In Abaco, functions are referred to as Actors. The National Science Foundation funded Abaco as a three-year project in September 2017, and the initial production software was released in January 2018. Several major new features have been released since then, based on user feedback and requirements. Since January 2018, hundreds of users have used this platform to perform their science research, approximately 50,000 actors were registered, and nearly 1 million executions have been performed with a total runtime of 25 million seconds. Over these years, different projects have adopted Abaco. For example, the Synergistic Discovery and Design (SD2) project has built RoundTrip, their automated synthetic experimental design system using a complex network of 30 actors; the UT Austin Covid-19 Modelling consortium web portal leverages Abaco for asynchronous task execution; NASA JPL incorporates actors as part of an automated data processing pipeline; etc.

Abaco has been designed as a distributed system and auto scales based on user demands. At the architectural level, it manages hundreds of actors, the message queues for each actor, a large number of executions and workers for each actor. The management of such processes/agents requires a careful design of communication between each of these processes. With the growing adoption of Abaco and Abaco usages in different scientific workflows, we have observed issues in communications between the actor and its workers, specifically for actor update and actor delete API requests. Towards making Abaco more resilient, we revisited the design for actor update and actor delete. We used TLA+ and TLC model checker to explore the design of communication between actors and workers in Abaco for different API requests.

The presentation will provide details of our efforts in understanding and improving the design of Abaco using TLA+ :

- We created and validated TLA+ specifications for the actor creation and auto-scaler module. While writing the specification, we learned how to abstract a web service, REST API, and get insights into the liveness and fairness properties.
- Towards understanding actor update and delete design, we redesigned and validated the actor update and actor delete using TLA+ and TLC model checker. The new design is much cleaner than the old one. We have planned a new implementation using the new design.
- We will highlight how a message queue system such as RabbitMQ used in a distributed system like Abaco can be abstracted in a specification.

In this presentation, we will also present lessons learned during the specification writing and experiences using TLA+ toolbox.