Towards Visualisation of Resilience Assessment for Large-Scale Systems

Elena Troubitsyna and Linas Laibinis
Åbo Akademi University
Turku, Finland
{linas.laibinis, elena.troubitsyna}@abo.fi

Abstract—Ensuring resilience of large-scale systems is a challenging engineering task. It can be greatly facilitated if the analysis of various system characteristics is performed at yearly design stages. Such an analysis helps the designers to assess various architectural alternatives with respect to such parameters as performance, reliability and cost. However, to facilitate experimentation with the created simulation models, it is desirable to visualise them. A visual graphical notation also facilitates model validation and discussion with different engineering teams. In this paper, we outline our approach to creating a graphical notation for simulation models used for analysis of performance and reliability. The initial validation of the proposed approach has demonstrated good potentials.

I. INTRODUCTION

To guarantee resilience and satisfy Service Level Agreement (SLA) that regulates service behaviour with respect to its customers, the service developers should ensure two main properties – fault tolerance and performance. To analyse performance/fault tolerance ratio of the architectural alternatives, it is desirable to create graphical models that can be easily adapted and discussed. On one hand, such a model should explicitly represent the aspects relevant to the resilience assessment, e.g., failure occurrence, off-nominal behavior etc. On the other hand, to facilitate translation of a model into the simulation code, it should be defined in the style compatible with the target simulation framework.

In this paper, we discuss an approach to visualising a discrete-event simulation models in SimPy [1]. The proposed visualisation helps to validate the simulation model and significantly simplified the development of the corresponding simulation models. We discuss the main components of our graphical notation, give a brief overview of the experiment and outline the future work.

II. GRAPHICAL MODEL

The aim of our experimentation with the graphical notation is to define the key concepts required to analyse resilience and enable automated translation of the graphical model into a parameterised SimPy code. Our model visualises the behaviour of the system in a process-oriented style. It is used to represent high-level system architecture, control flow and other details relevant to simulation. Such a representation fits well SimPy style and facilitates automation of translation.

The notation is light-weight and introduces only the core concepts of the domain together with the key artifacts required for analysis of resilience by simulation. The model defines the component interactions, symbolic representation of statistical parameters as well as explicit reactions on faults. The process-oriented model plays the role of a unifying blueprint of the system and serves two main purposes: defining the structure of the simulation models as well as providing an easy-to-comprehend visual representation for a multi-disciplinary engineering team.

III. EXPERIMENTATION

In pursuit of high performance, mechanisms traditionally used for ensuring fault tolerance are adapted to optimise reliability/performance ratio. We analyse different variants of implementing a well-known fault tolerance mechanism write-ahead logging [2] to evaluate fault tolerance/performance ratio. The main principle of WAL is to apply the requested changes to data files in a large distributed database only after they have been logged, i.e., after the log has been stored in the persistent storage. The WAL mechanism ensures fault-tolerance and increases performance, since only the log file should be written to the permanent storage to guarantee that a transaction is committed. We have experimented with using our graphical notation to implement synchronous, semi-synchronous and asynchronous modes of logging in WAL under unreliable logging assumption.

IV. DISCUSSION

Reliance on our graphical notation helped us to create a reusable model that significantly facilitated validation of simulation model and multi-disciplinary team discussions. Currently, we are working on automating translation from graphical notation to parameterised SymPy code.

ACKNOWLEDGMENT

This work is supported by the Finnish National Program Need for Speed http://www.n4s.fi/en/.

REFERENCES