Supporting Continuous Architecting of Data-Intensive Applications

Marcello M. Bersani, Francesco Marconi, Damian A. Tamburri

Politecnico di Milano, Milan, Italy

[*marcellomaria.bersani, francesco.marconi, damianandrew.tamburri*]@*polimi.it*

Andrea Nodari, Pooyan Jamshidi

[*p.jamshidi,a.nodari15*]@*imperial.co.uk*

[a] *Imperial College London, UK*

Abstract

Big data architectures have been gaining momentum in recent years. For instance, Twitter uses stream processing frameworks like Storm to analyse billions of tweets per minute and learn the trending topics. However, architectures that process big data involve many different components interconnected via semantically different connectors making it a difficult task for software architects to refactor the initial designs. As an aid to designers and developers, we developed OSTIA that stands for "On-the-fly Static Topology Inference Analysis"; OSTIA is a tool that allows: (a) recovering big data architectures for the purpose of design-time refactoring while maintaining constraints that would only be evaluated at later stages (deployment, run-time); (b) detecting the occurrence of common anti-patterns across big data architectures; (c) exploiting software verification techniques to evaluate the safety of the elicited architectural models. This paper illustrates OSTIA and evaluates its uses and benefits using industrial and open-source case study research.

Keywords: DevOps, Continuous Architecting, Big Data

1. Introduction

Big data or *data-intensive* applications (DIAs) process large amounts of data for the purpose of gaining key business intelligence through complex analytics using machine-learning techniques [1, 2]. These applications are receiving increased attention in the last years given their ability to yield competitive advantage by direct investigation of user needs and trends hidden in the enormous quantities of data produced daily by the average Internet user. According to Gartner [3] business intelligence and analytics applications will remain a top focus for Chief-Information Officers (CIOs) of most Fortune 500 companies until at least 2017-2018. However, the cost of ownership of the systems that process big data analytics are high due to infrastructure costs, steep learning curves for the different frameworks (such as Apache Storm [4], Apache Spark [5] or Apache Hadoop [6]) typically involved in design and development of big data applications.

A key complexity of the above design and development activity lies in quickly and continuously refining the configuration parameters of of the middleware and service platforms on top which the DIA is running - we defined this process as the *continuous architecting* of data-intensive architectures [7]. The process in question, namely, continuous architecting of data-intensive architectures, is especially complex as the number of middleware involved in DIAs design increases; the more middleware are involved the more parameters need co–evaluation (e.g., latency or beaconing times, caching policies, queue retention and more) - *fine-tuning these "knobs" on so many concurrent technologies requires an automated tool to speed up this heavily manual, trial-and-error continuous fine-tuning process.*

The entry-point for such fine-tuning is the DIA's graph of operations, which we call *topology* - a topology[1] is a directed graph which represents the cascade of operations to be applied on data in a batch (i.e., slicing the data and analysing one partition at the time with the same operations) or stream (i.e., continuous data analysis) processing fashion. The topology can either be known or directly extracted from DIA code.

This paper offers OSTIA, which stands for "On-the-fly Static Topology Inference Analysis" – OSTIA is a tool which retrieves data-intensive topologies and allows to apply three analyses in support of their continuous architecting: (a) *pattern matching* - OSTIA allows detection of known and established design anti-patterns for data-intensive applications; (b) *algorithmic manipulation* - OSTIA provides techniques to unfold, sequence, or wrap highly complex topologies, thus helping designers identify problems in their data-intensive designs; (c) *transparent formal verification* - OSTIA transposes the recovered data-intensive topology models into equivalent formal models for the purpose of verifying temporal properties, such as basic queue-safety clauses [8].

First, during its reverse-engineering step, OSTIA analyses

[1] the term is slightly overridden from the classical notion of topologies existing in DIAs, e.g., Apache Storm topologies.