Automated Architecture Recovery and Consistency Checking for Stream-Based Systems

Pooyan Jamshidi
Imperial College London
London, UK
Email: p.jamshidi@imperial.ac.uk

Andrea Nodari
Imperial College London
London, UK
Email: a.nodari15@imperial.ac.uk

Damian A. Tamburri
Politecnico di Milano
Milan, Italy
Email: damianandrew.tamburri@polimi.it

***Abstract*—Big data architectures have been gaining momentum in the last few years. For example, Twitter uses complex Stream topologies featuring frameworks like Storm to analyse and learn trending topics from billions of tweets per minute. However, verifying the consistency of said topologies often requires deployment on multi-node clusters and can be expensive as well as time consuming. As an aid to designers and developers evaluating their Stream topologies at design-time, we developed OSTIA, that is, "On-the-fly Static Topology Inference Analysis". OSTIA allows reverse-engineering of Storm topologies so that designers and developers may: (a) use previously existing model-driven verification&validation techniques on elicited models; (b) visualise and evaluate elicited models against consistency checks that would only be available at deployment and run-time. We illustrate the uses and benefits of OSTIA on three real-life industrial case studies.**

I. Introduction

Big data applications process vast amounts of data [1] for the purpose of gaining key business intelligence through complex analytics such as machine-learning [2]. Said applications are receiving increased attention in the last few 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. Gartner predicts[1] that said business intelligence and analytics applications will remain top focus for CIOs until at least 2017-2018. However, there are many costs and complexities behind harnessing said applications ranging from very high infrastructure costs to steep learning curves for the many frameworks (e.g., Apache Storm[2], Apache Spark[3] or Hadoop[4] to name a few) involved in designing and developing applications for Big Data.

In our own experience with designing and developing for Big Data, we observed that one such key complexity lies in evaluating architectures' effectiveness, that is, their deployability against maintained consistency with necessary Big data framework restrictions. Making sure this effectiveness is maintained before deployment means saving the time of running trial-and-error experiments while setting up expensive infrastructure needs as well as while fine-tuning complex framework setup options.

In an effort to aid designers and developers to jointly refine their Big data applications at design time, we developed OSTIA, that stands for: "On-the-fly Static Topology Inference Analysis". OSTIA allows designers and developers to quickly start refining their application collaboratively by inferring the application topology though reverse-engineering and architecture recovery [3]. OSTIA currently focuses on Storm, the most famous and established real-time stream processing Big data engine [4]. OSTIA hardcodes intimate knowledge on the development and framework dependence structure necessary for correct Storm topologies for two purposes: (a) realising a visual representation for said topologies; (b) export said topologies for further analysis.

This paper outlines OSTIA, elaborating its major usage scenarios, its benefits while properly discussing and addressing its limitations. Also, we evaluate OSTIA on three industrial case-studies from an open-source social-sensing application to show that OSTIA yields valuable design and development insights using an inference analysis of the three static topologies for said application. We conclude that OSTIA does in fact provide valuable insights for software engineers to increase quality of their Big data design and development featuring Storm.

The rest of the paper is structured as follows. Section II outlines our research problem, research questions and our approach at tackling them. Section III outlines OSTIA, discussing its typical usage scenarios and outlining the main benefits we perceived in using it. Section IV elaborates further on the benefits by providing an actual evaluation of OSTIA using three cases from industrial practice. Section V-B discusses the results and evaluation, also outlining OSTIA limitations and potential threats to its validity. Finally, Sections VI and VII outline related work and conclude the paper.

II. Research Approach

- so we had a focus group to actually elaborate the aprroach
- then we used explorative prototyping to elicit the initial version of the prototype and then refined that by means of case study, we could mention that we used ATC as an action-research source (this is what we are doing now internally in WP2 actually)
- ...

[1] http://www.gartner.com/newsroom/id/2637615
[2] http://storm.apache.org/
[3] http://spark.apache.org/
[4] https://hadoop.apache.org/

III. Research Solution

IV. Evaluation

- we can use the ATC case study as much as we want - that yields already three topologies that we can infer
- any additional case that we can run?
- what do the results show? do we have a way to quickly quantify the time that is saved by using this approach? e.g., the time that is saved in setting up and running the infrastructure and how much would that time saved have costed these could be valuable evaluation insights

V. Discussion

A. Findings and Quality-Improvement Insights

- we should probably enumerate the many issues we have found with the work in WP2 and WP3 to allow some good insights for the people using OSTIA
- also we could discuss the benefits one by one using some usage-scenarios

B. Approach Limitations and Threats to Validity

- here we could elaborate on the limitations of addressing only storm and why we are addressing storm (e.g., the main technology for streaming currently)
- we should probably discuss the fact that the tech is still eclipse-based and how this is not really a limitation after all
- are there any threats to validity? we should probably discuss this as well

VI. Related Work

- mention DICE
- mention work by Len Bass on Big-Data
- other stuff on big data?
- feel free to extend this section with Previous work of course :)

VII. Conclusion

Acknowledgment

Authors' work is partially supported by the European Commission grant no. 644869 (EU H2020), DICE. Also, Damian's work is partially supported by the European Commission grant no. 610531 (FP7 ICT Call 10), SeaClouds.

References

[1] C. K. Emani, N. Cullot, and C. Nicolle, "Understandable big data: A survey." *Computer Science Review*, vol. 17, pp. 70–81, 2015. [Online]. Available: http://dblp.uni-trier.de/db/journals/csr/csr17.html#EmaniCN15

[2] B. Ratner, *Statistical and Machine-Learning Data Mining: Techniques for Better Predictive Modeling and Analysis of Big Data*. CRC PressINC, 2012. [Online]. Available: http://books.google.de/books?id=0KuIJTAznLUC

[3] I. Ivkovic and M. W. Godfrey, "Enhancing domain-specific software architecture recovery." in *IWPC*. IEEE Computer Society, 2003, pp. 266–273. [Online]. Available: http://dblp.uni-trier.de/db/conf/iwpc/iwpc2003.html#IvkovicG03

[4] R. Evans, "Apache storm, a hands on tutorial." in *IC2E*. IEEE, 2015, p. 2. [Online]. Available: http://dblp.uni-trier.de/db/conf/ic2e/ic2e2015.html#Evans15