OpenTOSCA for the 4th Industrial Revolution: Automating the Provisioning of Analytics Tools based on Apache Flink

Michael Falkenthal, Uwe Breitenbücher, Kálmán Képes,
Frank Leymann, Michael Zimmermann
Institute of Architecture of Application Systems,
University of Stuttgart, Germany
{falkenthal, breitenbuecher, kepes, leymann, zimmermann}@iaas.uni-stuttgart.de

Maximilian Christ, Julius Neuffer,
Nils Braun, Andreas W. Kempa-Liehr
Blue Yonder GmbH,
Karlsruhe, Germany
maximilian.christ@blue-yonder.com

Full ACM reference as provided by ACM : Michael Falkenthal, Uwe Breitenbücher, Kálmán Képes, Frank Leymann, Michael Zimmermann, Maximilian Christ, Julius Neuffer, Nils Braun, Andreas W. Kempa-Liehr. 2016. OpenTOSCA for the 4th Industrial Revolution: Automating the Provisioning of Analytics Tools based on Apache Flink. In *Proceedings of the 6th International Conference on the Internet of Things* (IoT '16). ACM, New York, NY, USA, 179-180.
DOI= 10.1145/2991561.2998463 http://doi.acm.org/10.1145/2991561.2998463

BIBTEX:

```
@inproceedings{Falkenthal2016,
    author    =  {Michael Falkenthal, Uwe Breitenb\”{u}cher, Kalman Kepes, Frank
                 Leymann, Michael Zimmermann, Maximilian Christ, Julius Neuffer,
                 Nils Braun, Andreas W. Kempa-Liehr},
    title     =  {OpenTOSCA for the 4\textsuperscript{th} Industrial Revolution:
                 Automating the Provisioning of Analytics Tools based on Apache
                 Flink},
    booktitle =  {Proceedings of the 6\textsuperscript{th} International
                 Conference on the Internet of Things},
    year      =  {2016},
    pages     =  {179-180},
    publisher =  {ACM},
    doi       =  {10.1145/2991561.2998463}
}
```

OpenTOSCA for the 4th Industrial Revolution: Automating the Provisioning of Analytics Tools based on Apache Flink

**Michael Falkenthal, Uwe Breitenbücher,
Kálmán Képes,
Frank Leymann, Michael Zimmermann**
University of Stuttgart
Stuttgart, Germany
falkenthal@iaas.uni-stuttgart.de

**Maximilian Christ, Julius Neuffer, Nils Braun,
Andreas W. Kempa-Liehr**
Blue Yonder GmbH
Karsruhe, Germany
maximilian.christ@blue-yonder.com

ABSTRACT

The 4th industrial revolution entails new levels of data driven value chain organization and management. In industrial environments, the optimization of whole production lines based on machine learning algorithms allow to generate huge business value. Still, one of the open challenges is how to process the collected data as close to the data sources as possible. To fill this gap, this paper presents an OpenTOSCA-based toolchain that is capable of automatically provisioning Apache Flink as a holistic analytics environment altogether with specialized machine learning algorithms. This stack can be deployed as close to the production line as possible to enable data driven optimization. Further, we demonstrate how the analytics stack can be modeled based on TOSCA to be automatically provisioned considering specific mock services to simulate machine metering in the development phase of the algorithms.

ACM Classification Keywords

K.6 Management of Computing and Information Systems; D.2.6 Software Engineering: Programming Environments

Author Keywords

4th Industrial Revolution; Cyber-Physical Systems; Apache Flink; Data Mock Services; Machine Learning; TOSCA.

INTRODUCTION & BACKGROUND

New endeavors to employ smart analytics in manufacturing in order to optimize whole production lines and processes are condensed in the *4th industrial revolution*. Machinery are treated as *cyber-physical systems*, which means that they act, on the one hand, as data sources that hold metering data about specific production stages available. On the other hand, they may also act as actuators that can be controlled to influence the processing of workpieces. One major ambition in terms of the 4th industrial revolution is to leverage the power of *Data Science* and *Machine Learning* techniques in order to identify statistical patterns from sensor data and corresponding meta-information to anticipate future machine states. This enables to automatically detect failures and inaccuracies in the process of production to automatically adapt configurations and adjustments of the machinery. At least decision support on how to manually optimize the production process shall be provided in situations, which are not appropriate for fully automatic adaptions of the machinery. To realize such automated analysis scenarios, an analytics platform along with machine learning algorithms have to be provisioned in a technical infrastructure – typically in the form of a cloud environment. Especially in the development phase of the algorithms, where metering of machines often has to be mocked by specific data mock services, the fully automated provisioning of the analytics stack can increase the efficiency in terms of providing rapid prototypes and fast development cycles. But also in the state of production, fully automated provisioning of analytics stacks can increase the efficiency to adapt algorithms or to provide new relevant ones to quickly react to new production requirements and optimization initiatives.

Apache Flink[1] is a batch and stream processing platform that can be used to compute metering data for optimizing whole production lines. It can be used to deploy and run analytics algorithms implemented in different programming languages, such as Java, Scala and Python. Besides distributing workload over a cluster of compute nodes Apache Flink also supports local computations in a single virtual machine.

This paper shows how an analytics stack based on Apache Flink along with analytics algorithms implemented in Python can be fully automatically provisioned using *OpenTOSCA* [1], a toolchain based on the TOSCA standard [4].

SCENARIO – PREDICTION AND MOCKED DATA DELIVERY SERVICES FOR INDUSTRIAL APPLICATIONS

For a data-driven approach in industrial applications it is important to consider the context of a specific data source. For this purpose, prediction and delivery services communicate via Full-Metadata Format (FMF) files, which combine data source-specific meta-information with tabular measurement data [5]. The prediction service specifies the data, which are needed for its computations, as FMF file header, which is

IoT'16 November 07-09, 2016, Stuttgart, Germany

ACM ISBN 978-1-4503-4814-0/16/11.

DOI: http://dx.doi.org/10.1145/2991561.2998463

[1] https://flink.apache.org

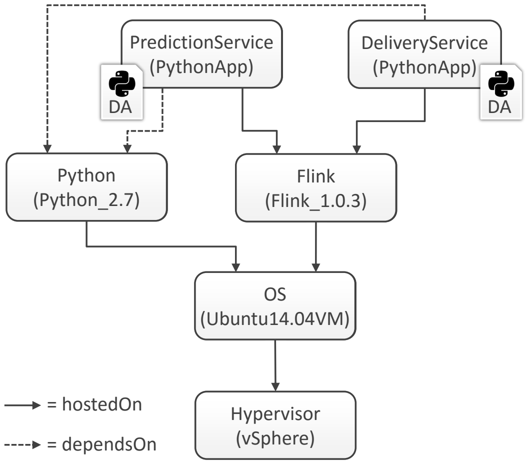

Figure 1. TOSCA topology model of the analytics stack scenario

provided to the delivery service. The delivery service identifies the requested information, retrieves them from the related cyber-physical systems, appends the requested data as CSV table to the FMF file header and provides the result to the prediction service. For this demo, the delivery service mocks the measurement by simulating time series from a stochastic process [3, p. 164], from which the prediction service computes a simple forecast via Flink and provides the result as FMF file to the delivery service.

OVERVIEW ON TOSCA

The structure of the analytics stack is represented by means of a so called *topology model* on basis of the TOSCA standard [4]. The topology model is a node and edge colored graph. Its components are modeled by *node templates*, while the relations between the components are represented by *relationship templates*. Enabling to populate a topology model with different types of components and their specific dependencies, arbitrary *node types* and *relationship types* can be defined. The actual component implementations, i.e., executables such as Python scripts or binaries a component is made of, can be provided by means of so called *deployment artifacts* and related to the respective node template that represents the component. Management capabilities of components are exposed by *management operations*, which are attached by means of *implementation artifacts*. The whole analytics stack can be bundled in a self-contained archive using the TOSCA packaging format, which is called *Cloud Service Archive (CSAR)*.

AUTOMATED DEPLOYMENT DEMONSTRATION

The topology model representing the analytics stack of our scenario is depicted in Fig. 1. In the shown topology the analytics tools and algorithms should be provisioned on a virtual machine hosted on vSphere[2] with the Linux distribution Ubuntu 14.04 running on it. Of course, the stack can also be provisioned on other hypervisor platforms, e.g., OpenStack[3], by only exchanging the corresponding node template. On Ubuntu, the analytics platform Apache Flink as well as Python 2.7 – which is a required dependency of the Python applications modelled in the topology – should be installed. Therefore, the Ubuntu node has an implementation artifact implemented as Web application ARchive (WAR) that exposes management operations in order to transfer files to the virtual machine or to run scripts on it using a SSH connection. The needed information are either specified directly in the topology model by means of properties, such as the credentials for SSH, or are determined during runtime, as for instance the IP-address of the virtual machine. In this scenario, we use the so called *local setup* of Apache Flink to run the instance of the analytics stack on a single machine. To install Apache Flink, the node exposes the management operation *install*, which is implemented using Ansible[4]. Moreover, all required dependencies like the analysis libraries *sklearn*[5] and *pandas*[6] are installed on the virtual machine together with Flink. However, with TOSCA it is also possible to model all dependencies as separate nodes. On the top of the analytics stack are (i) the *DeliveryService* for simulating incoming metering data and (ii) the *PredictionService* for analyzing this data that should be processed with Flink. Both services are implemented as Python applications. The corresponding implementations are attached as deployment artifacts to the respective node templates, hence, the implementations can easily be exchanged.

For the deployment of the described topology, we use OpenTOSCA [1], an open-source implementation of the TOSCA standard. OpenTOSCA allows the automatic provisioning and management of applications. Therefore, OpenTOSCA includes a *plan generator* [2], that can analyze the topology and generate a BPEL workflow describing the order of the management operations needed to be executed to provision the modeled application stack. The generated workflow is automatically deployed on a workflow engine (WSO2 BPS[7]) and executed by OpenTOSCA. OpenTOSCA as well as the shown TOSCA topology are publicly available[8].

ACKNOWLEDGMENTS

This work is partially funded by the BMWi project SePiA.Pro (01MD16013F).

REFERENCES

1. Tobias Binz and others. 2013. OpenTOSCA - A Runtime for TOSCA-based Cloud Applications. In *ICSOC*. Springer, 692–695.
2. Uwe Breitenbücher and others. 2014. Combining Declarative and Imperative Cloud Application Provisioning based on TOSCA. In *IC2E*. IEEE, 87–96.
3. Andreas W. Liehr. 2013. *Dissipative Solitons in Reaction Diffusion Systems*. Springer, Berlin.
4. OASIS. 2013. *Topology and Orchestration Specification for Cloud Applications (TOSCA) Version 1.0*.
5. Moritz K. Riede and others. 2010. On the Communication of Scientific Data: The Full-Metadata Format. *Computer Physics Communications* 181, 3 (2010), 651–662.

[2] http://www.vmware.com/products/vsphere.html

[3] http://www.openstack.org/

[4] https://www.ansible.com

[5] http://scikit-learn.org

[6] http://pandas.pydata.org

[7] http://wso2.com/platform

[8] http://www.opentosca.org/demos/smart-prediction-demo