# Simplifying the Deployment of Big Data Solutions

Ivan Ermilov[1] and Axel-Cyrille Ngonga Ngomo[2]

[1] Instituts für Angewandte Informatik, Hainstraße 11, 04107 Leipzig, Germany,
iermilov@informatik.uni-leipzig.de
[2] Paderborn University, Warburger Str. 100, 33098 Paderborn, Germany
axel.ngonga@upb.de

**Abstract.** The deployment of big data solutions such as Hadoop MapReduce, Apache Spark, and Apache HBase involves rigorous configuration management and tuning of the production environment to satisfy the performance requirements. With the help of BDE SL methodology [1], the Big Data Europe (BDE) project provides common big data components packaged into Docker images, which can be used as building blocks for big data applications. In this demo we showcase the BDE platform for the deployment of Hadoop MapReduce, Apache HBase and Halyard triplestore [2]. The components of the BDE platform include sensible defaults to start with and can be deployed on a single Docker host or a Docker Swarm cluster. Docker Swarm support enables scaling out to a cluster of machines.

**Keywords:** Big Data, DevOps, Triplestore, RDF

*Deployment Scenario.* The deployment scenario includes Apache Hadoop[3], and Apache HBase[4] components, which we packaged into Docker images inside the BDE project. For the one node deployment (c.f. Figure 1) we orchestrate all the components necessary for the Halyard triplestore[5] on a single server or a developer machine without redundancy. This setup can be used for development and testing purposes. In the multi node setup, we distribute components in a cluster to ensure uninterrupted operation of the Halyard triplestore. Namenode, HMaster and Zookeeper are orchestrated in a distributed mode with backups for each of the services, thus the failover is available. Moreover, it is possible to scale the setup horizontally by simply providing new nodes with Datanode, Resource Manager and HRegion.

*Cluster Configuration.* Both one node and multi node deployment are managed using *docker-compose.yml* definitions. In the BDE project, we follow the best practices such as 12 factor apps[6] and make the deployment as granular as

[3] http://hadoop.apache.org/
[4] https://hbase.apache.org/
[5] https://github.com/merck/halyard
[6] https://12factor.net/

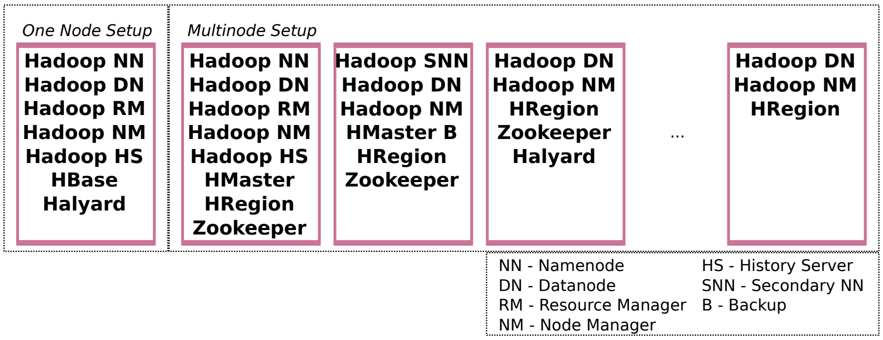

**Fig. 1.** One and multi node deployment of Halyard triplestore.

possible. For example, the minimal deployment of Apache Hadoop consists of two services: *namenode* and *datanode* with configuration injected through environment variables. The complete setup for the orchestration both for one and multiple nodes is available on the Github[7].

To deploy the Halyard triplestore on one node, the user simply executes: `docker-compose up`. For the multinode deployment on Docker Swarm the user needs to adapt docker-compose.yml from our repository and deploy it as a stack: `docker stack deploy -c docker-compose.yml`. In the repository on github we provide an example setup for three node deployment of Halyard.

*Cluster Deployment.* In the demo session we demonstrate the working setup of three node Halyard deployment on remote cluster. We also show how to deploy local setup, perform the RDF data loading and querying with RDF4J console and RDF4J SPARQL endpoint.

*Acknowledgements.* This work was supported by grant from the European Union's Horizon 2020 research Europe flag and innovation program for the project Big Data Europe (GA no. 644564).

## References

1. Ermilov, I., Ngomo, A.C.N., Versteden, A., Jabeen, H., Sejdiu, G., Argyriou, G., Selmi, L., Jakobitsch, J., Lehmann, J.: Managing Lifecycle of Big Data Applications. In: Różewski, P., Lange, C. (eds.) Knowledge Engineering and Semantic Web. No. 786 in Communications in Computer and Information Science, Springer (2017), `https://svn.aksw.org/papers/2017/KESW_BDE_Workflow/public.pdf`
2. Sotona, A., Negru, S.: How to Feed Apache HBase with Petabytes of RDF Data: an Extremely Scalable RDF Store Based on Eclipse RDF4J. In: CEUR-WS, ISWC Demo Track (2017), `http://ceur-ws.org/Vol-1690/paper35.pdf`

[7] `https://github.com/dice-group/docker-halyard`