A Demo of a Software Platform for Ubiquitous Big Data Engineering, Visualization, and Analytics, via Reconfigurable Micro-Services, in Smart Factories

Mirco Soderi, Vignesh Kamath, John G. Breslin
Data Science Institute, National University of Ireland Galway, Ireland {firstname.lastname}@nuigalway.ie

***Abstract*—Intelligent, smart, Cloud, reconfigurable manufacturing, and remote monitoring, all intersect in modern industry and mark the path toward more efficient, effective, and sustainable factories. Many obstacles are found along the path, including legacy machineries and technologies, security issues, and software that is often hard, slow, and expensive to adapt to face unforeseen challenges and needs in this fast-changing ecosystem. Lightweight, portable, loosely coupled, easily monitored, variegated software components, supporting Edge, Fog and Cloud computing, that can be (re)created, (re)configured and operated from remote through Web requests in a matter of milliseconds, and that rely on libraries of ready-to-use tasks also extendable from remote through sub-second Web requests, constitute a fertile technological ground on top of which fourth-generation industries can be built. In this demo it will be shown how starting from a completely virgin Docker Engine, it is possible to build, configure, destroy, rebuild, operate, exclusively from remote, exclusively via API calls, computation networks that are capable to (i) raise alerts based on configured thresholds or trained ML models, (ii) transform Big Data streams, (iii) produce and persist Big Datasets on the Cloud, (iv) train and persist ML models on the Cloud, (v) use trained models for one-shot or stream predictions, (vi) produce tabular visualizations, line plots, pie charts, histograms, at real-time, from Big Data streams. Also, it will be shown how easily such computation networks can be upgraded with new functionalities at real-time, from remote, via API calls.**

***Index Terms*—Software Platform, Micro-service, API, Edge, Fog, Cloud, Big Data, Data Engineering, Data Visualization, Data Analytics, Intelligent Manufacturing, Smart Manufacturing, Reconfigurable Manufacturing, Remote Monitoring**

I. Introduction

Smart and intelligent manufacturing [1], Cloud manufacturing [2], reconfigurable manufacturing [3], remote monitoring [4], stand at the basis of the evolution toward fourth-generation factories that characterize for (i) an improved efficiency, both in terms of economic and environmental costs [5] [6], and (ii) an improved effectiveness, also in terms of resiliency, and throughput, intended as the capacity of satisfying larger shares of (possibly customized) demand in shorter time.

Reconfigurable manufacturing is a cornerstone [7], since (i) it enables the reuse of resources, (ii) it reduces semi-finished products transportation, (iii) it allows to adapt to a changing demand, and (iv) it opens to products customization. Lightweight, portable, loosely coupled, easily monitored, variegated software components, supporting Edge [8], Fog [9], and Cloud [10] computing, that can be (re)created, (re)configured and operated from remote through Web requests in a matter of milliseconds, and that rely on libraries of ready-to-use tasks also extendable from remote through sub-second Web requests, constitute a fertile technological ground on top of which fourth-generation industries can be built.

Under a merely technological point of view, three main technology trends are identified in [11], that are (i) the conformance to well-known standard protocols for data exchange among physical and cyber-physical components; (ii) the spread of graphical user interfaces for system development, monitoring and operation; (iii) the spread of containerization technologies. The demo proposed in this work stands at the intersection of those.

A. Background

Our proposed approach is based on the following building blocks, meant to be deployed as Docker Containers: (i) *Network Factory*, a Node-RED application that exposes APIs to create, upgrade, organize, initialize, start, stop, delete network nodes; (ii) *Service Nodes* [12], computation nodes created through the Network Factory and implemented as Node-RED applications that all expose a common set of APIs for input, output, and task (re)configuration; they are typically configured to perform data readings from heterogeneous sources or data transformations, but they can also expose data on the Web or interface with Artificial Intelligence Servers; (iii) *MQTT brokers*, also possibly created through the Network Factory, they are the means through which Service Nodes exchange data each other; (iv) *Transformation Library* [12], also created and upgraded through API calls made to the Network Factory, it contains a collection of software modules (Node-RED subflows) that are loaded into Service Nodes for execution by means of API calls made to the Service Nodes themselves; (v) *Artificial Intelligence Servers* [11], also created and upgraded through API calls to the Network Factory (possibly even without a server restart), they are Scala + Spark + AKKA HTTP applications that expose a set of APIs to interface with Service Nodes and that offer an extendable set of functionalities including AI-related tasks, Big Data charts generation, and stream transformations.

This publication has emanated from research conducted with the financial support of Science Foundation Ireland under Grant Number SFI/16/RC/3918 (Confirm). For the purpose of Open Access, the author has applied a CC BY public copyright licence to any Author Accepted Manuscript version arising from this submission.

B. Structuring

The paper is structured as follows. Context, motivations, and background research are presented in Section I. Location and usage directions for the software artifacts that are necessary for running and possibly expanding this demo are given in Section II, where details are also given about what is exactly shown in the demo. Future directions are anticipated in Section III. Conclusions are drawn in Section IV.

II. Demo

In this demo it is shown how starting from a virgin Docker Engine, it is possible to (re)build, (re)configure, and operate from remote via API calls, computation networks capable to (i) raise alerts based on configured thresholds or trained ML models, (ii) transform Big Data streams, (iii) produce and store Big Datasets on the Cloud, (iv) train and persist ML models on the Cloud, (v) use trained models for one-shot or stream predictions, (vi) produce tabular visualizations, line plots, pie charts, histograms, at real-time, from Big Data streams. Also, it will be shown how easily such computation networks can be enriched of new functionalities from remote.

A. Getting started

Once that a Docker Engine would be installed and running, the Docker Image msoderi/network-factory will have to be pulled, and a Docker Container started from that. That leads to have a Network Factory available. For the purposes of this demo, the internal port 1880 must be mapped to port 585.

B. Carrying on development activities

For those interested in customizing or extending the Network Factory (out of the scope of this demo), that can be done by restoring the backup of its Docker Volume [1] retrieved from the GitHub repository[2] to the /data folder of a Node-RED Container, and then connecting to the Web interface exposed by the resulting Node-RED application. By default, the Web interface is provided over HTTPS, and self-signed certificates are in use. Docker Volume backups are available for all the building blocks mentioned in Subsection I-A.

C. Consulting the documentation

In addition to the scientific production, some artifacts are available in the GitHub repository for documentation purposes: (i) a Microsoft PowerPoint presentation, named demo.pptx, guides the user step-by-step through this demo; (ii) a preliminary release of the Swagger OpenAPIv2 documentation for Network Factory and Service Nodes is also available in apidoc-networkfactory.yaml and apidoc-servicenode.yaml.

[1]https://github.com/mircosoderi/State-of-the-art-Artifacts-for-Big-Data-Engineering-and-Analytics-as-a-Service/raw/main/networkfactory.tar

[2]https://github.com/mircosoderi/State-of-the-art-Artifacts-for-Big-Data-Engineering-and-Analytics-as-a-Service

D. Importing the Postman API Collection

The Postman API Collection, consisting of 1200+ API calls, stands at the core of this demo. It can be downloaded from the GitHub repository, where it is named demo.postman_collection.json. It has to be imported in a Postman installation to be run.

E. Going through the API collection

API calls from #1 to #5 create a boundary inside of which the nodes will be created. They also instantiate the Transformation Library, MQTT broker, and ACL nodes.

API calls from #6 to #199 build and configure a network that monitors a configured (source) RDB table for new rows, and replicates to a configured (alert) RDB table those that bear a value greater than a configured threshold.

API calls #200 to #206 create and start an Artificial Intelligence Server node, then upgrade it with a task that is needed for this demo, enable a Service Nodes to interface with the server, and reconfigure that Service Node to raise alerts based on a server-side clustering instead of on a configured threshold. The resulting network is depicted in Fig. 1.

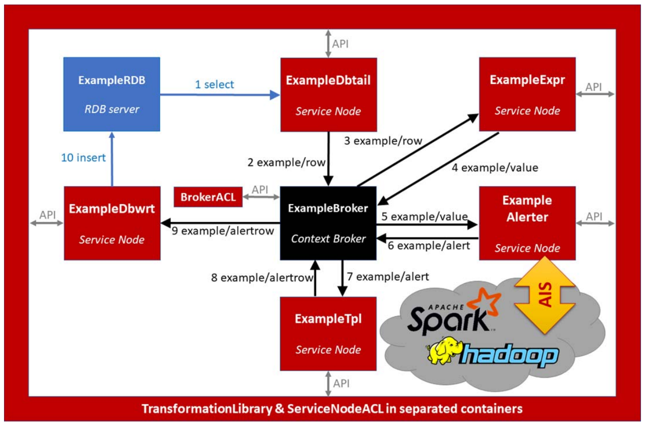

Fig. 1. Network architecture for the clustering-based alerting system

So far, testing could be only based on SQL queries. API calls from #207 to #224 add a new module to the Transformation Library, and then create, configure and start a new Service Node that uses the new module. This way, alerts can be monitored through a Web interface.

API calls #225 to #244 clear everything created so far.

API calls #245 to #495 create, configure, and start the necessary nodes for running the Big Data stream processing examples. These include nodes for data input, data processing, monitoring (ControlRoom), and single-node Apache Kafka and Hadoop instances (for demo purposes). The resulting network architecture is depicted in Fig. 2.

API calls #496 to #550 configure and operate a simple stream forward. API calls #551 to #559 configure and operate a conditional stream forward where messages are produced to the output stream if they bear a value between 50 and 100. API calls #560 to #567 configure and operate a stream

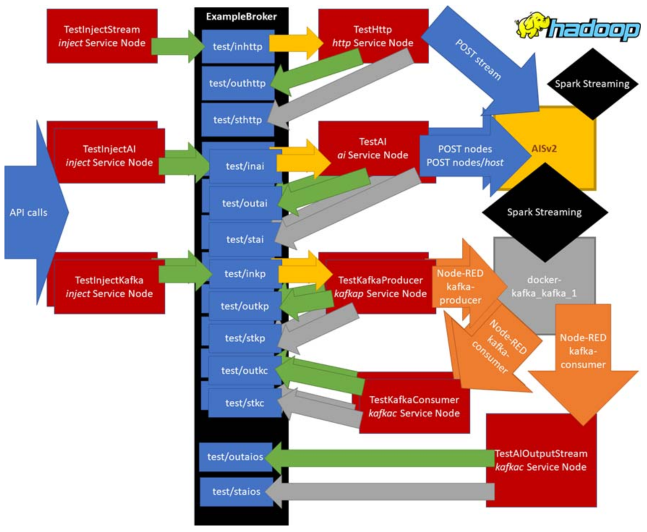

Fig. 2. Network architecture for stream processing and AI-related examples

transformation: an expression is computed for each incoming message, and the result is outputted to the output stream.

API calls #568 to #740 (re)configure and operate the network to produce and persist a libsvm dataset to be used as a training set.

API calls #741 to #761 configure and operate the network to train a clustering model, and then persist and use the trained model for one-shot and stream predictions. API calls #762 to #781 do the same with a classification (logistic regression) model.

API calls #782 to #869 clear everything created so far.

API calls #870 to #1122 build the network for data visualization examples, then configure the same network for producing two similar tabular representations of messages read from a configured Kafka stream. The resulting network is depicted in Fig. 3. Tables are visible at https://localhost:2130/ui.

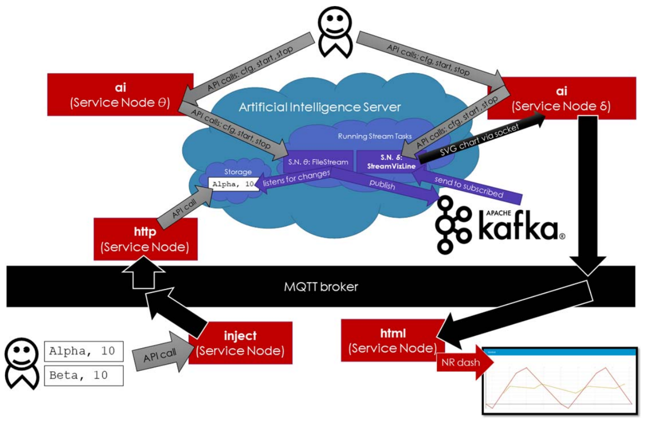

Fig. 3. Network architecture for data visualization examples

API calls #1123 to #1140 configure and operate the network for the real-time generation of a line plot from a Big Data stream. API calls from #1141 to #1160 generate pie charts instead. Finally, API calls #1161 to #1181 produce histograms. All are visible at the above address.

API calls #1182 to #1227 clear everything created so far.

III. Future directions

Two main future directions are identified: (i) enriching the Network Factory of APIs for (re)building user interfaces at real-time; (ii) opening to the possibility of running the network nodes not only on targeted Docker Engines but also on the Cloud, via Kubernetes.

IV. Conclusions

The proposed demo shows how starting with a Docker Engine and a msoderi/network-factory Container run on top of it, it is possible to (re)build, (re)configure, upgrade, and operate from remote easily monitored applications ranging from alerting systems to Big Data stream transformations, AI-based predictions, Big Data charts generation, and possibly much more. That constitutes a fertile technological ground on top of which reconfigurable factories can be built, which leads to costs reduction and greater resiliency and sustainability.

References

[1] Wang, Baicun, et al. ”Smart manufacturing and intelligent manufacturing: A comparative review.” Engineering 7.6 (2021): 738-757.
[2] Ren, Lei, et al. ”Cloud manufacturing: key characteristics and applications.” International journal of computer integrated manufacturing 30.6 (2017): 501-515.
[3] Koren, Yoram, et al. ”Reconfigurable manufacturing systems.” CIRP annals 48.2 (1999): 527-540.
[4] Jonsson, Katrin, Ulrika H. Westergren, and Jonny Holmström. ”Technologies for value creation: an exploration of remote diagnostics systems in the manufacturing industry.” Information Systems Journal 18.3 (2008): 227-245.
[5] Salonitis, Konstantinos, and Peter Ball. ”Energy efficient manufacturing from machine tools to manufacturing systems.” Procedia Cirp 7 (2013): 634-639.
[6] Duflou, Joost R., et al. ”Towards energy and resource efficient manufacturing: A processes and systems approach.” CIRP annals 61.2 (2012): 587-609.
[7] Mehrabi, Mostafa G., A. Galip Ulsoy, and Yoram Koren. ”Reconfigurable manufacturing systems: Key to future manufacturing.” Journal of Intelligent manufacturing 11.4 (2000): 403-419.
[8] Cao, Keyan, et al. ”An overview on edge computing research.” IEEE access 8 (2020): 85714-85728.
[9] Stojmenovic, Ivan, et al. ”An overview of fog computing and its security issues.” Concurrency and Computation: Practice and Experience 28.10 (2016): 2991-3005.
[10] Qian, Ling, et al. ”Cloud computing: An overview.” IEEE international conference on cloud computing. Springer, Berlin, Heidelberg, 2009.
[11] Soderi, Mirco, et al. ”Advanced Analytics as a Service in Smart Factories” IEEE 20th International Symposium on Applied Machine Intelligence and Informatics (SAMI 2022). IEEE, 2022. (forthcoming)
[12] Soderi, Mirco, et al. ”Ubiquitous System Integration as a Service in Smart Factories.” 2021 IEEE International Conference on Internet of Things and Intelligence Systems (IoTaIS). IEEE, 2021.