Crazy Nodes: Towards Ultimate Flexibility in Ubiquitous Big Data Stream Engineering, Visualisation, and Analytics, in Smart Factories

Mirco Soderi(✉) and John G. Breslin

National University of Ireland, Galway, Ireland
{mirco.soderi,john.breslin}@nuigalway.ie

Abstract. Smart Factories characterize as context-rich, fast-changing environments where heterogeneous hardware appliances are found beside of also heterogeneous software components deployed in (or directly interfacing with) IoT devices, as well as in on-premise mainframes, and on the Cloud. This inherent heterogeneity poses major challenges particularly when a high degree of resiliency is needed, and the ubiquitously deployed software components must be replaced or reconfigured at real-time to respond to the most diverse events, ranging from an out-of-range sensor detection, to a new order issued by a customer. In this work, a software framework is presented, which allows to deploy, (re)configure, run, and monitor the most diverse software across all the three layers of the Smart Factory (edge, fog, Cloud), from remote, via API calls, in a standardised uniform manner, relying on containerization technologies, and on a variety of software technologies, frameworks, and programming languages, including Node-RED, MQTT, Scala, Apache Spark, and Kafka. The most recent advances in the framework design, implementation, and demonstration, which led to the introduction of the so-called Crazy Nodes, are presented and motivated. A comprehensive proof-of-concept is given, where user interfaces and distributed systems are created from scratch via API calls to implement AI-based alerting systems, Big Data stream filtering and transformation, AI model training, storage, and usage for one-shot as well as stream predictions, and real-time Big Data visualization through line plots, histograms, and pie charts.

Keywords: Smart manufacturing · Resilient manufacturing · Cloud manufacturing · Artificial intelligence · AI · Distributed system · System of systems · Ubiquitous computing · Stream processing · Stream visualization · Micro-service · Application programming interface · API

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.

T. Margaria and B. Steffen (Eds.): ISoLA 2022, LNCS 13704, pp. 235–240, 2022.
https://doi.org/10.1007/978-3-031-19762-8_18

1 Introduction

1.1 Reconfigurable Manufacturing

Real-time (re)configurability is a key challenge in Smart Manufacturing [1]. The diversity of components is a significant obstacle for interoperability, and then for reconfigurability. On the other hand, a reconfigurable manufacturing allows among the others (i) a more sustainable and efficient manufacturing [2], by eliminating transportation; (ii) low-volume high-value production of customized or even personalised goods [3]; (iii) the alignment of production to demand [4]; (iv) a real-time fine tuning for optimizing energy consumption, mitigating the risk of damages, or increasing the safety of operators, based on sensor detections [5]. A real-world application to pharmaceuticals manufacturing is presented in [6]. Compared to other systems such as Apache AirFlow [7], our proposed framework allows to (i) tune/edit the workflow from remote while data is flowing; (ii) address low-to-high resource devices and the Cloud, uniformly; (iii) implement arbitrary flows; (iv) avail of thousands of ready-to-use modules; (v) implement reconfigurable user interfaces and data charts. However, since APIs are in our framework to instantiate arbitrary applications, the two systems can integrate.

1.2 Framework Overview

The main building blocks of the proposed software framework for the ubiquitous Big Data (stream) engineering, analytics, and visualization as a Service are listed as follows: (i) Network Factory; (ii) Transformation Library; (iii) Service and Crazy Nodes; (iv) AI Servers. The **Network Factory** is a Node-RED application available on the Docker Hub as msoderi/network-factory. It implements APIs that create, start, stop, upgrade, and delete, Docker bridges (fences) and a variety of specialized yet reconfigurable applications in Docker containers. It essentially turns any device equipped with Docker into a totally flexible, real-time remotely reconfigurable equipment. A **Transformation Library** is a containerized Node-RED application that consists in an extendable collection of reusable software modules meant to be loaded to the Service/Crazy Nodes via API calls, and there executed. **Service Nodes** [8], and **Crazy Nodes**, are also containerized Node-RED applications, and expose API calls for configuring input and output MQTT broker instances and topics, and the task to be executed. Depending on the specific module that is loaded into a Node, both input and output can be from/to MQTT brokers, or just one of those. **AI Servers** [9] are containerized Scala + Spark applications. They expose APIs for interfacing with Service and Crazy Nodes. AI Server nodes keep a configuration file for each interfacing Service and Crazy node, and include an extendable library of parallel/stream/AI-related tasks. AI servers also support the remote deployment and execution of Scala expression compiled on the fly when needed, without server restart. The framework is prone to **security issues**, which are mitigated through secure communication protocols, isolation (Docker bridges), and ACL nodes. A comprehensive example system is depicted in Fig. 1.

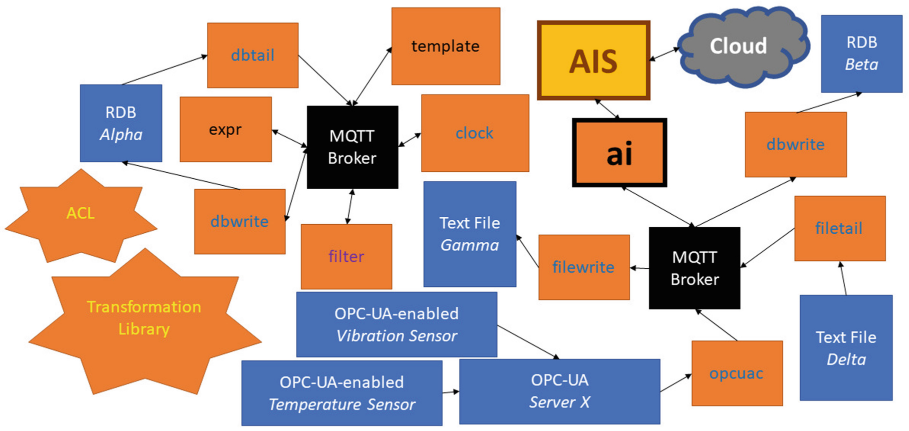

Fig. 1. An example distributed system that can be built through our proposed framework. Service/Crazy nodes are depicted each as an orange rectangle; some of them interface with IoT devices, databases, file systems... (blue labels), while others only interface with configured brokers (black labels), or with AI servers (bold large label). (Color figure online)

1.3 Paper Structure

The challenges that are posed by reconfigurable and resilient manufacturing, as well as its pivotal role in a wide range of use cases and applications, are outlined in Subsect. 1.1. An overview of the software framework is provided in Subsect. 1.2. Service and Crazy Nodes, and their interaction with AI Servers, are described in Sect. 2. A proof of concept is provided in Sect. 3. Conclusions are drawn in Sect. 4.

2 Service and Crazy Nodes

Service Nodes are containerized Node-RED applications. As soon as created through the Network Factory, they all look the same: they are disconnected from any data source, implement the identity function, and expose configuration APIs. Through appropriate calls, MQTT clients are created inside of Service Nodes, and configured to connect to input, output, and status MQTT brokers/topics. At creation time, Service Nodes are bound to a Transformation Library. Through appropriate calls, reusable software modules are copied from the Library to the Service Nodes, where they are then executed. In Fig. 2, the internals of a Service Node, and the ecosystem around it, are depicted. Transformation Library nodes are also created through the Network Factory. They natively include a variety of reusable modules for data input, RDB monitoring, data exchange over OPC-UA, data filtering, data transformation, and others. Service Nodes are not meant to perform Big Data processing, run parallel computations, or implement AI algorithms. Instead, they are suitable for edge computing, data preparing, and

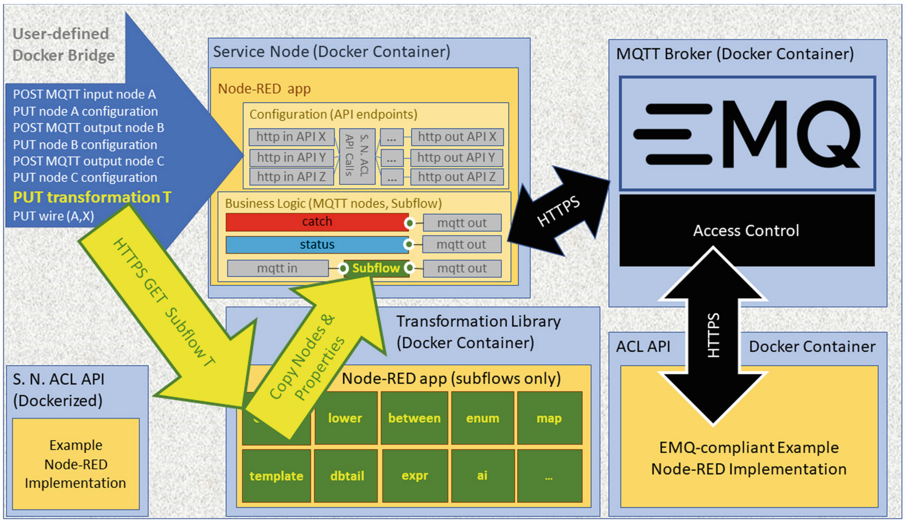

Fig. 2. The internals of a Service Node, and the ecosystem around of it. Instead, in Crazy Nodes, not only you can import/load/copy a task from the Transformation Library, but you can also freely modify the task that is executed inside of a specific node, from remote.

for implementing user interfaces, relying on the Node-RED dashboard module. The "ai" module, natively available in the Transformation Library, can be loaded to Service Nodes from remote to make them capable of interfacing with AI Servers through a variety of communication technologies. Being containerized, Service Nodes can be deployed and run everywhere a Docker Host is available, which makes them device-agnostic, and allows to deploy, configure, and run, heterogeneous software components in a uniform manner.

2.1 Crazy Nodes

Crazy Nodes are maximally reconfigurable Service Nodes. In a Service Node, the task that the node performs can only be one of those available in its associated Transformation Library, and the customization options are only those that the developer of the task has made available. Instead, in Crazy Nodes, in addition to importing reusable modules from the Transformation Library, it is possible to freely add, configure, link, and delete, specific parts in the task that the node executes. All changes have immediate effect and are operated through API calls, which makes Crazy Nodes suitable for (i) event-driven or sensor-driven fine tuning, (ii) incremental implementation, (iii) minor adjustments on the user interface, and remarkably, (iv) it opens to real-time software modifications in response to unforeseen events, which can be seen as the maximum achievable level of resiliency.

2.2 Towards the Cloud

The "ai" module must be loaded into a Service/Crazy Node to enable the interaction with an AI Server. The module implements: (i) configuration APIs, which call corresponding Server APIs for configuring the task that the Server must execute when input come from the Node; (ii) input API, which calls the corresponding Server API and triggers task execution; (iii) output and status APIs, which are asynchronously called by the Server to notify the Service/Crazy Node of status changes, and provide the result if applicable. For stream tasks, the Node only inputs control commands, while data are read/written from/to configured Kafka streams. The interaction is depicted in Fig. 3.

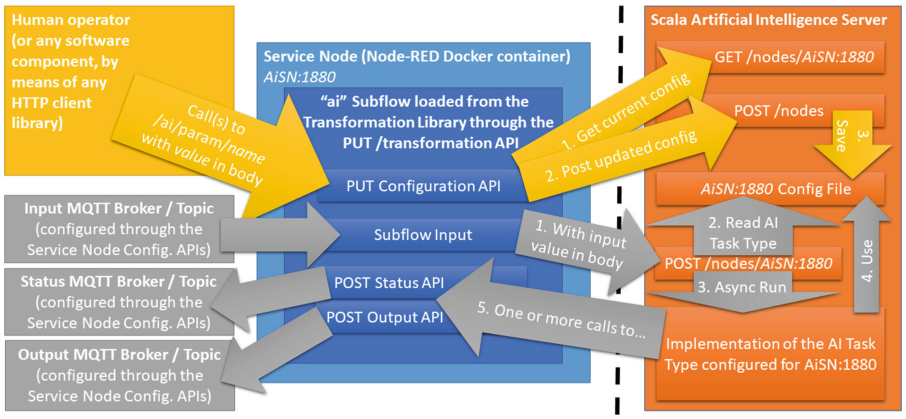
Human operator (or any software component, by means of any HTTP client library)
Call(s) to /ai/param/name with value in body
Input MQTT Broker / Topic (configured through the Service Node Config. APIs)
Status MQTT Broker / Topic (configured through the Service Node Config. APIs)
Output MQTT Broker / Topic (configured through the Service Node Config. APIs)
Service Node (Node-RED Docker container) AiSN:1880
"ai" Subflow loaded from the Transformation Library through the PUT /transformation API
PUT Configuration API
Subflow Input
POST Status API
POST Output API
1. Get current config
2. Post updated config
1. With input value in body
5. One or more calls to...
Scala Artificial Intelligence Server
GET /nodes/AiSN:1880
POST /nodes
3. Save
AiSN:1880 Config File
2. Read AI Task Type
POST /nodes/AiSN:1880
3. Async Run
4. Use
Implementation of the AI Task Type configured for AiSN:1880

Fig. 3. How "ai" Service or Crazy Nodes interact with AI Servers

3 Proof of Concept

As a proof of concept, a former demo [10] has been extended, and API calls have been added for creating and connecting new user interfaces from scratch, without relying on any pre-existing module from the Transformation Library. The resulting proof of concept consists in a Postman collection including 1400+ API calls to build from scratch, configure, operate, destroy, distributed systems for (i) alerting applications, (ii) Big Data engineering, (iii) AI models training/storage/usage, and for (iv) real-time Big Data stream visualization [11]. A step-by-step demo presentation, the Postman export, the OpenAPI documentation, and all software artifacts are available in the GitHub repository[1].

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

4 Conclusions

In this work, a software framework for ubiquitous Big Data (stream) engineering, analytics, and visualization as a Service has been presented. In particular, containerized reconfigurable Node-RED applications named Crazy Nodes have been discussed. The improvements with respect to the former version of those applications, named Service Nodes, have been highlighted. A proof of concept has been proposed where new user interfaces have been constructed from scratch component by component from remote through API calls, and integrated in a pre-existing demo.

References

1. Koren, Y., et al.: Reconfigurable manufacturing systems. CIRP Ann. **48**(2), 527–540 (1999)
2. Bortolini, M., Galizia, F.G., Mora, C.: Reconfigurable manufacturing systems: literature review and research trend. J. Manuf. Syst. **49**, 93–106 (2018)
3. Zennaro, I., Finco, S., Battini, D., Persona, A.: Big size highly customised product manufacturing systems: a literature review and future research agenda. Int. J. Prod. Res. **57**(15–16), 5362–5385 (2019)
4. Lebovitz, R., Graban, M.: The journey toward demand driven manufacturing. In: Proceedings 2nd International Workshop on Engineering Management for Applied Technology. EMAT 2001, pp. 29–35 (2001). IEEE
5. Haghnegahdar, L., Joshi, S.S., Dahotre, N.B.: From IoT-based cloud manufacturing approach to intelligent additive manufacturing: industrial Internet of Things—an overview. Int. J. Adv. Manuf. Technol. **119**, 1–18 (2021). https://doi.org/10.1007/s00170-021-08436-x
6. Adamo, A., et al.: On-demand continuous-flow production of pharmaceuticals in a compact, reconfigurable system. Science **352**(6281), 61–67 (2016)
7. Singh, P.: Airflow. In: Learn PySpark, pp. 67–84. Apress, Berkeley, CA (2019). https://doi.org/10.1007/978-1-4842-4961-1_4
8. Soderi, M., Kamath, V., Morgan, J., Breslin, J.G.: Ubiquitous System Integration as a Service in Smart Factories. In: 2021 IEEE International Conference on Internet of Things and Intelligence Systems (IoTaIS), pp. 261–267 (2021). IEEE
9. Soderi, M., Kamath, V., Morgan, J., Breslin, J.G.: Advanced analytics as a Service in Smart Factories. In: 2022 IEEE 20th Jubilee World Symposium on Applied Machine Intelligence and Informatics (SAMI), pp. 000425–000430 (2022). IEEE
10. Soderi, M., Kamath, V., Breslin, J.G.: A demo of a software platform for ubiquitous big data engineering, visualization, and analytics, via reconfigurable micro-services, in smart factories. In: 2022 IEEE International Conference on Smart Computing (SMARTCOMP), pp. 1–3 (2022). IEEE
11. Soderi, M., Kamath, V., Breslin, J.G.: Toward an API-driven infinite cyber-screen for custom real-time display of big data streams. In: 2022 IEEE International Conference on Smart Computing (SMARTCOMP), pp. 153–155 (2022). IEEE