A Service for Resilient Manufacturing

Mirco Soderi
Data Science Institute
University of Galway
Galway, Ireland
mirco.soderi@universityofgalway.ie

John Gerard Breslin
Data Science Institute
University of Galway
Galway, Ireland
john.breslin@universityofgalway.ie

***Abstract*—In modern industry, adaptation to market changes, as well as prompt reaction to a variety of predictable and unpredictable events, is a key requirement. Ubiquitous computing, real-time analytics, reconfigurable hardware/software components, often coexist in the complex, internally variegated, and often proprietary systems that are traditionally deployed to meet such requirement. However, such tailor-made systems meet only in part the requirements of openness, security, monitorability, geographical distribution, and most of all, remote extendability and changeability, which are crucial for prompt reaction to unforeseen circumstances. In this work, a containerized service application named Network Factory is presented. It enables the remote construction, configuration and operation of resilient computation systems that meet the above-mentioned requirements, and distinguish for their logical simplicity and for the uniform addressing of elaborations and human-computer interfaces, which are achieved through few reconfigurable components and communication mechanisms that are used from the production line up to the Cloud. Source code, documentation, and step-by-step introductory guides are publicly available in a dedicated GitHub repository, and distributed under the CC-BY-4.0 license.**

***Index Terms*—Resiliency, Reconfigurable manufacturing, Ubiquitous computing, Real-time analytics, Open architecture, Security, Monitorability, Changeable software, Geo-Distributed system, Containerization**

I. Introduction

In a world that is becoming more and more unpredictable, being capable to take immediate action to face the unforeseen, either automatically or semi-automatically, ensures a huge competitive advantage to modern industries. Unexpected orders, service disruptions, cyber attacks, natural events, are becoming more and more frequent, and not being able to face them in due time might put factories, communities, and countries, at high risk. A large and variegated set of requirements must be met to be prepared to tackle such extreme challenges (Fig. 1). Failing to meet just one of them may compromise the promptness, the efficiency, or the effectiveness of the countermeasures.

This publication has emanated from research supported in part by a grant from Science Foundation Ireland under Grant Number SFI/16/RC/3918 (Confirm), and also by a grant from SFI under Grant Number SFI 12/RC/2289_P2 (Insight). 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.

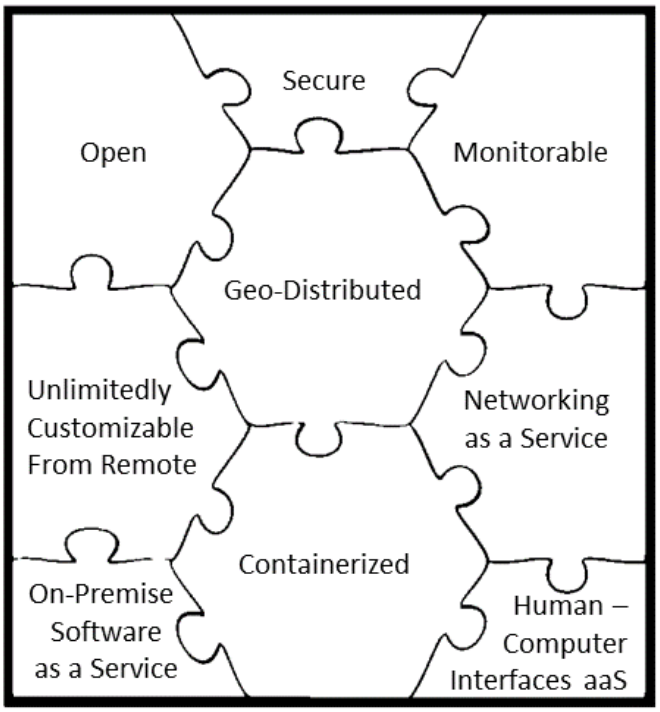

Fig. 1. Requirements

A. Requirements

If immediate action is needed, it cannot be afforded to wait for any third party, which is one of the motivations for preferring open software, protocols and architectures. In many cases, production is physically distributed [1], then for achieving synchronous timely responses it is necessary to have standardized mechanisms in place for remote software installation from shared repositories, configuration, and operation. This is a variant of the Software-as-a-Service (SaaS) [2] delivery model, in which the software runs on targeted devices, instead of on the Cloud. Even beyond, if the issue to be faced is completely unexpected, no ready-to-use software module may be available, which is why it is crucial to have mechanisms in place that enable arbitrary software modifications from remote. To the best of our knowledge, this has not been addressed in any previous research work. This implies dealing with security and stability issues. To mitigate them, the system needs to be

remotely monitorable by design, which is why MQTT brokers and event streaming servers are used for data transferring across atomic components deployed in the same device, or across multiple devices. For being resilient to communication technology disruptions, standardized mechanisms for the remote deployment of alternative communication technologies are needed. Soderi and Breslin [3] have demonstrated how Bluetooth Low Energy (BLE) servers can be instantiated, configured, and operated from remote. Although highly automatized, all production systems have humans in the loop. Then, the capacity to build, configure and publish arbitrary user interfaces from remote makes a big difference. Soderi et al. [4] have presented a demo where customized Big data charts are created and configured from remote. For addressing the most diverse physical devices uniformly, an abstraction layer is required. This is one of the reasons why containerization technologies such as Docker are hugely helpful, and used in our proposal.

B. Related works

Kombaya Touckia et al. [5] have proposed a design for a digital twin framework for reconfigurable manufacturing systems (RMSs) targeted to ensure prompt reactions to unforeseen circumstances. Hajjem et al. [6] have discussed the challenges and requirements of digital twin frameworks for reconfigurable manufacturing systems, and identified the remote installation, configuration and operation of plug and play software as "the path to be followed in order to produce high-quality product and to meet the market requirements". Kurniadi and Ryu [7] have proposed a conceptual framework for IoT-based RMS, emphasizing the role of human operators. Haddou et al. [8] have proposed a digital twin modular framework for reconfigurable manufacturing systems that includes reusable components that recall those that can be created through our proposed Factory Service. Cannata et al. [9] have proposed SOCRADES, a framework for intelligent manufacturing systems based on Service Oriented Architectures (SOA).

C. Paper structure

In Section I the objectives, requirements, and related works are presented. In Section II, the Network Factory is described, its suitability for the outlined purpose is discussed, and directions to get started with it are briefly provided; full details are available in the GitHub repository[1]. In Section III, conclusions are drawn.

II. Network Factory

The Network Factory is a containerized Node-RED application meant to add extensive remote reconfiguration capabilities to any device on which it runs, so enabling system resilience. At today, it includes 30+ APIs (Fig. 2) for (i) logical organization and isolation of the reconfigurable distributed application, (ii) creation, configuration and control of the atomic software components (nodes), (iii) management of module repositories, and (iv) inspection and update of the security mechanisms.

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

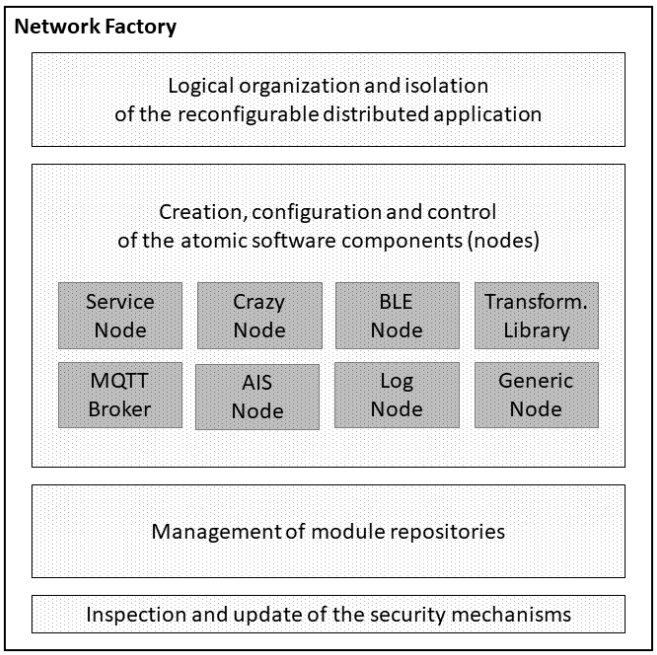

Fig. 2. Network Factory

A. Service

A *fence* is a possibly empty set of containerized applications that (i) refer to each other by name, (ii) cannot be accessed from outside the fence unless ad-hoc configurations are set, and (iii) have mediated access to system resources. Fences contribute to application and hosting system security, and to software maintainability. At present, user-defined Docker networks are used for implementing fences. The Network Factory exposes APIs for fence creation, inspection, modification, and deletion.

The Network Factory exposes APIs for creating, pausing, resuming, starting, and stopping a variety of reconfigurable containerized applications, also named *nodes*, such as Service Nodes [10], Crazy Nodes [11], BLE Nodes [3], Transformation Libraries [10], MQTT brokers [10], Artificial Intelligence Server (AIS) Nodes [12], Log Nodes (Apache Flume server instances), and generic nodes. Although different endpoints are exposed for the creation of the different types of nodes, the execution flow is similar for the creation of any of them (Fig. 3).

Applications created through the Network Factory are modular by design. Service and Crazy Nodes expose an API to set the task to be executed, which accesses an external repository (Transformation Library), retrieves the task implementation and copies it into the Service or Crazy node itself for execution. AIS Nodes have internal tasks libraries. The Network Factory exposes APIs for extending or updating Transformation Libraries and AIS Node internal libraries.

In end, the Network Factory exposes APIs for inspecting and updating the security mechanisms that are applied to any request made to the the Network Factory itself.

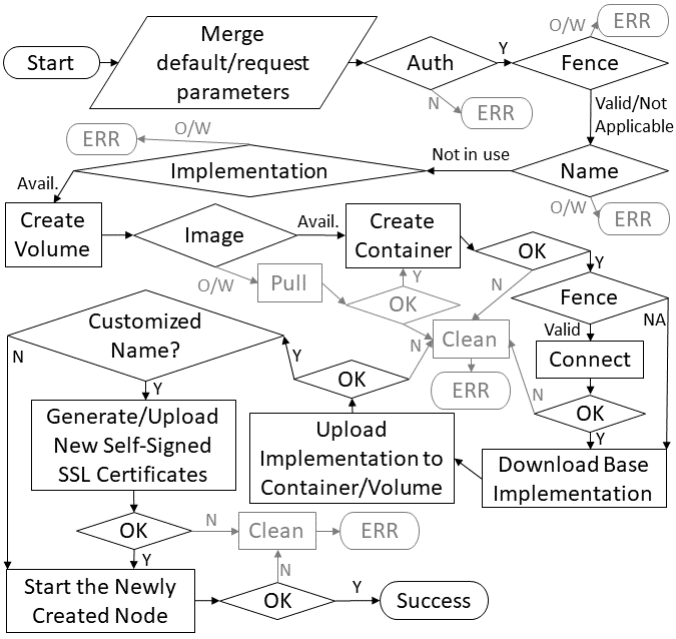

Fig. 3. Node creation

B. Discussion

Based on requirements, the strengths of this work follow. All source code and documentation materials are open-source and distributed under the CC-BY-4.0 licence. All APIs are exposed over HTTPS, with access control delegated to external APIs. All reconfigurable nodes created through the Network Factory run tasks retrieved from remotely extendable libraries. In the specific case of Crazy Nodes, immediate (semi-)automated response to unforeseen events is possible by operating arbitrary modifications on the task implementation from remote. Applications built through the Network Factory are monitorable by design, as nodes are meant to exchange data by means of MQTT brokers or streaming servers, and integration with Apache Flume is supported. An example of communication as a service is provided, which consists in the creation and configuration of a BLE server from remote. User interfaces can be built and updated from remote [4]. All nodes, and the Network Factory itself, are deployed and run in Docker containers, which ensures portability, and isolation. The main weaknesses are: (i) lightweight security protocols [13], MQTT brokers [14], and other, are not yet supported; (ii) integration with Kubernetes is still lacking; (iii) extensive benchmarking and application in production environments is still lacking.

C. Usage

To get started, the reader is invited to go through a Postman collection of extensively documented API requests made to Docker, to the Network Factory and to the created reconfigurable nodes, which also includes information to produce and use customized artifacts. A simple alerting system is created, using a pre-configured threshold first, and introducing a clustering model later. A user interface is also added. Further details can be found in the GitHub repository.

III. Conclusions

A containerized Node-RED application named Network Factory has been presented. It can used by human operators (through appropriate interfaces) and automated agents for creating, configuring/linking, and operating a variety of reconfigurable software components from remote and with unprecedented flexibility. It is meant to be used in reconfigurable manufacturing, and any scenario where timely reactions to unforeseen events are required. It is extensively documented, and distributed open-source under CC-BY-4.0 licence.

References

[1] T. M. Rupp and M. Ristic, "Fine planning for supply chains in semiconductor manufacture," *Journal of materials processing technology*, vol. 107, no. 1-3, pp. 390–397, 2000.

[2] A. Dubey and D. Wagle, "Delivering software as a service," *The McKinsey Quarterly*, vol. 6, no. 2007, p. 2007, 2007.

[3] M. Soderi and J. Gerard, "Ble servers and ubiquitous analytics aas," *AICS 2022 Digital Book of Abstracts*, 2022.

[4] M. Soderi, V. Kamath, and J. G. Breslin, "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)*. IEEE, 2022, pp. 153–155.

[5] J. Kombaya Touckia, N. Hamani, and L. Kermad, "Digital twin framework for reconfigurable manufacturing systems (rmss): design and simulation," *The International Journal of Advanced Manufacturing Technology*, vol. 120, no. 7-8, pp. 5431–5450, 2022.

[6] E. Hajjem, H. H. Benderbal, N. Hamani, and A. Dolgui, "Digital twin framework for reconfigurable manufacturing systems: Challenges and requirements," in *Advances in Production Management Systems. Artificial Intelligence for Sustainable and Resilient Production Systems: IFIP WG 5.7 International Conference, APMS 2021, Nantes, France, September 5–9, 2021, Proceedings, Part II*. Springer, 2021, pp. 553–562.

[7] K. A. Kurniadi and K. Ryu, "Development of iot-based reconfigurable manufacturing system to solve reconfiguration planning problem," *Procedia manufacturing*, vol. 11, pp. 965–972, 2017.

[8] H. Haddou Benderbal, A. R. Yelles-Chaouche, and A. Dolgui, "A digital twin modular framework for reconfigurable manufacturing systems," in *Advances in Production Management Systems. Towards Smart and Digital Manufacturing: IFIP WG 5.7 International Conference, APMS 2020, Novi Sad, Serbia, August 30–September 3, 2020, Proceedings, Part II*. Springer, 2020, pp. 493–500.

[9] A. Cannata, M. Gerosa, and M. Taisch, "Socrades: A framework for developing intelligent systems in manufacturing," in *2008 IEEE International Conference on Industrial Engineering and Engineering Management*. IEEE, 2008, pp. 1904–1908.

[10] M. Soderi, V. Kamath, J. Morgan, and J. G. Breslin, "Ubiquitous system integration as a service in smart factories," in *2021 IEEE International Conference on Internet of Things and Intelligence Systems (IoTaIS)*. IEEE, 2021, pp. 261–267.

[11] M. Soderi and J. G. Breslin, "Crazy nodes: towards ultimate flexibility in ubiquitous big data stream engineering, visualisation, and analytics, in smart factories," in *Leveraging Applications of Formal Methods, Verification and Validation. Practice: 11th International Symposium, ISoLA 2022, Rhodes, Greece, October 22–30, 2022, Proceedings, Part IV*. Springer, 2022, pp. 235–240.

[12] M. Soderi, V. Kamath, J. Morgan, and J. G. Breslin, "Advanced analytics as a service in smart factories," in *2022 IEEE 20th Jubilee World Symposium on Applied Machine Intelligence and Informatics (SAMI)*. IEEE, 2022, pp. 000 425–000 430.

[13] W. Jung, S. Hong, M. Ha, Y.-J. Kim, and D. Kim, "Ssl-based lightweight security of ip-based wireless sensor networks," in *2009 International Conference on Advanced Information Networking and Applications Workshops*. IEEE, 2009, pp. 1112–1117.

[14] W. Pipatsakulroj, V. Visoottiviseth, and R. Takano, "mumq: A lightweight and scalable mqtt broker," in *2017 IEEE International Symposium on Local and Metropolitan Area Networks (LANMAN)*. IEEE, 2017, pp. 1–6.