Service Level Agreements for Safe and Configurable Production Environments

Mohammad Ashjaei*, Kester Clegg†, Lorenzo Corneo‡, Richard Hawkins†,
Omar Jaradat*, Vincenzo Massimiliano Gulisano§, Yiannis Nikolakopoulos§
*Mälardalen University, † University of York, ‡Uppsala University, §Chalmers University

***Abstract*—This paper focuses on Service Level Agreements (SLAs) for industrial applications that aim to port some of the control functionalities to the cloud. In such applications, industrial requirements should be reflected in SLAs. In this paper, we present an approach to integrate safety-related aspects of an industrial application to SLAs. We also present the approach in a use case. This is an initial attempt to enrich SLAs for industrial settings to consider safety aspects, which has not been investigated thoroughly before.**

I. Introduction

With emerging smart factories as part of the recent Industry 4.0 initiatives, several technologies have been exploited to obtain flexibility, adaptability and evolvability in production management. Technologies such as cloud computing, and the recent fog-computing trend, offer a large virtualized computation power to support part of the production complexity. As the mentioned technologies were developed in different context (IT), they provide support for industrial services with technical limitations without fulfilling all industrial requirements. Therefore, several works addressed various aspects of using these technologies in industrial domains, such as overall technological issues [1], integration to the Internet of Things (IoT) [2], robustness and scalability of cloud services [3] and timeliness properties of cloud services [4].

One of the main challenges is that the quality of services that are offered by cloud providers cannot be controlled by cloud consumers. Therefore, the quality of cloud services are negotiated and defined in an agreement. This agreement, which is known as Service Level Agreement (SLA) [5], contains a description of services with various parameters, such as availability. Although the research community on industrial cloud computing has paid a lot of attention on technical aspects of cloud computing, not much research has been done in the area of SLA management, modeling and definition for industrial cloud computing, as identified in [6]. In particular, research in defining parameters that industrial applications are interested in is still very young. For example, parameters related to safety and security are either not defined or they are defined qualitatively. Moreover, given the complicated interactions between various technologies in cloud-based systems, it is crucial for users to have a way of measuring how mature a certain group of technologies is. In this sense, SLAs could easily quantify the technological readiness. Nonetheless, SLAs in the current form are not adequate in covering safety related aspects due to the following challenges:

- Many important services in industrial domains are safety-critical functionalities, which should be reflected in SLAs.
- One of the biggest challenges in safety assurance is dealing with reconfiguration. Safety goals, evidence and argument are highly dependent on assumptions made about operating context and system configuration. The SLAs should define how the reconfiguration can be done.
- Upcoming technologies like osmotic computing [7] demonstrate the dynamicity of service placement and the possibilities for delegation across different parts of the system. This can result in systems having vastly different configurations, which can be defined in SLAs.
- With workloads consisting of a mix of safety critical as well as other production-related functionalities, it is expected that the overall system performance may have to be balanced against safety requirements.

SLAs could be used as a potential means of addressing these challenges by specifying what is expected from each element in an industrial setting. This would require SLAs to define the minimum properties of a system or subsystem that should be maintained in order to assure the safety of the overall system. This will allow for the implementation of any element within the system to change, without affecting the safety of the overall system, as long as it can be guaranteed that the respective SLA is still honoured. This paper is an initial attempt to address the above mentioned challenges by presenting an approach on defining SLAs considering safety-related aspects of industrial applications. Moreover, a concrete case study is presented to better clarify the proposed approach.

The rest of the paper is organized as follows. The next section describes the safety challenge. Sections III, IV and V present three main components of a generic cloud-based system with intra and inter dependencies. Section VI presents a use case and Section VII concludes the paper.

II. Safety and deriving SLAs

There are many systems within factories whose failures, under certain conditions, can lead to human harm or damage to property or the environment, e.g. due to the use of heavy machinery or hazardous substances [8]. Therefore risks associated with the manufacturing processes and the resulting products are analysed, controlled and monitored. In comparison to conventional manufacturing, Industry 4.0 tends to be more re-configurable, modular and dynamic in nature. This poses significant additional safety assurance challenges

since a more re-configurable design often means there is less control over that design, which in turn gives rise to uncertainty (more unpredictable situations or behaviours). Changing or re-configuring the factory will often invalidate the operational or environmental assumptions that are made as part of the safety assurance process. This can impact the evidence that has been generated regarding the operation of the system (which is often valid only in the operational and environmental context in which it is obtained). The evidence might no longer support the developers' claims because it could reflect old development artefacts or old assumptions about operation or the operating environment [9]. This uncertainty implies less confidence in the safety performance of the factory systems.

SLAs can be used to describe the dependencies between the different parts of systems in a "Guarantee-Assumptions" form. This means that an SLA can describe a property important to the safety of the system (parameter, value, behaviour, etc.) together with whatever assumptions are required to guarantee that property. These required properties can be determined from the safety requirements of the factory systems. When constructing an SLA, it is desirable to make the SLA specification as flexible as possible, as this supports the Industry 4.0 desire to change and reconfigure the factory. For example, the safe stopping time for an autonomous vehicle on the factory floor in normal circumstances may be determined from safety analysis of the factory to be three seconds. This requirement could be incorporated into an SLA. However it might be the case that the stopping time is actually longer than this under certain conditions (e.g., in case of slippery or steep floors). Thus, when designing SLAs the possible anticipated changes to the SLA's guarantees must also be considered.

In this paper, we suggest an approach for deriving and refining SLAs for use in safety assurance of factories as presented in Fig. 1.

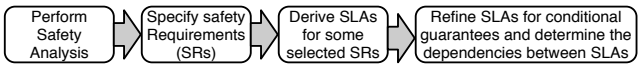

Fig. 1. The derivation process of SLAs.

Deriving and refining of SLAs are done within different parts of a system, which we call them *pillars*. In this paper we identify three pillars: the communication, virtualization and processing pillars. However, the approach can work for more or different pillars based on the system's requirements.

III. The three pillars

This section presents the three pillars that set the basis for the fine-grained details of SLAs requirements.

a) Communication: The communication pillar covers the objective of reliably and timely disseminating messages between several entities of the system (e.g., physical system and cloud/fog devices). In order to ensure safety, monitoring applications must have real-time guarantees. Real-time behaviors are usually provided by selecting suitable communication technologies, protocols and metrics.

b) Virtualization: The virtualization pillar embraces the partitioning of computation or communication services with the aim of supporting isolation and flexibility in a cloud-based system. In computing services, virtualization is realized by hypervisors and virtual machines, or other related techniques such as containers. The networks can also be partitioned for various services using Software Defined networking (SDN).

c) Processing: The processing pillar defines the technology used to analyze the data retrieved from the sensors distributed in a production environment and transform them into valuable information that can be used to take decisions (with or without the involvement of humans) and adjust the production process depending on the information being sensed, as we further elaborate in the use case of Section VI. In the context of Industry 4.0, modern data processing paradigms such as data streaming [10] fit well to the need for *high-throughput* and *low-latency* analysis.

IV. Intra-pillar SLA dependencies

In order to define SLAs that embrace the whole spectrum of requirements, we first focus on requirements that exist within each pillar (independently of dependencies with other pillars, discussed later in Section V).

a) Communication: Communication is responsible of delivering information within the system enforcing safety through real-time behavior. When translating this to SLAs, the goal is to provide metrics that reflect the quality of real-time aspects achieved by the network infrastructure. In particular, potential communication SLAs must take into account information quality (e.g., data freshness) and necessary latency for reacting to events in the physical system (e.g., actuation).

b) Virtualization: Virtualization acts as a technique to provide isolation between services, thus not only the system management reduces but also the temporal and performance metrics can be analyzed without considering the whole system. This means that the performance and temporal aspects of a service can be evaluated in isolation of other services. Considering this aspect in SLAs, the key goal is to provide metrics that can support virtualization in a cloud-based system with required and desired performance metrics. Note the system should consider that the effect of live reconfiguration, either manual or automatic, could adversely affect the virtualization performance and this should be declared in the SLAs.

c) Processing: As discussed in [10], one of the main advantages of processing paradigms such as data streaming is the possibility of transparently (from the programmer perspective) distributing and parallelizing data analysis. The advantage in this case, is the possibility of pushing the analysis to the edge, thus leveraging the considerable cumulative computational power of devices (from embedded to server-like ones) available in large production systems. In this context, the challenge stems from the need of properly mapping the analysis tasks to the existing devices since this affects the overall throughput and latency that can be achieved by the system. When translating this into SLAs, a key question is how deployment and adaptive actions of the data analysis

frameworks can be taken, either manually or via autonomous reconfigurations.

V. Inter-pillar SLA dependencies

As we exemplify in this section, once the requirements of each pillar are defined (as discussed in Section IV) their inter-dependencies must also be addressed in SLAs.

a) Communication: Virtualization and processing decisions heavily depend on data freshness (i.e., the time separating sensing and actions based on it). For virtualization choices, decrements in data freshness could mean, for example, that the allocated resources (i.e. bandwidth) for a particular sub-network are not enough and must be increased in order to meet time and safety requirements. For processing, when analyzing a stream of data coming from several devices for instance, data freshness can be used to prioritize the scheduling of tasks with hard real-time requirements.

b) Virtualization: Virtualization happens on both computation and communication parts of a system. Therefore, it has dependencies with both communication and processing pillars. In case of computation, the main aim is to keep the processing delays in a bounded value as well as availability in bandwidth to operate different functions. A challenging task is to consider the delays and bandwidth availability when reconfiguration of the system occurs. In the communication case, the network can be partitioned into several sub-networks to serve various services. Therefore, latencies of traffic in each sub-network and bandwidth availability in case of reconfiguration in sub-networks should be foreseen.

c) Processing: Throughput and latency of a certain application depend on decisions taken (at the deploy and at run time) about how analysis tasks are mapped to the available computational units in the production environment. In this sense, a tight connection exists with respect to both communication and virtualization, since both will frame the set of possible deployment choices. More concretely, the choice framing stems from the available bandwidth and the latency introduced when delivering information (communication) and from the available computational power (i.e., CPU and memory) available at each unit (virtualization). Together, communication and virtualization decisions will affect how close to the edge the analysis can be pushed.

VI. Use-case study

The use-case, based on a real system implementation, involves an automated factory in which a number of robots are required to move autonomously around the factory in order to transport goods from one location to another. The robots operate as independent agents, meaning they calculate their own required route through the factory without knowledge of the position of other robots operating within the factory (there is no communication between robots). It is necessary from a safety perspective (but also operationally) that the robots achieve their objectives without colliding with other peers. This is achieved through the use of Ultra Wide Band (UWB) radio devices to provide indoor localisation services. The system consists of robots, each one carrying a 'tag', that are tracked by a series of 'anchors' (or beacons) that receive the tags signals and compute their positions. The anchors communicate with the tags via UWB radio. Additionally, the system includes a wifi network located between the anchors and the fog (or cloud) based UWB location service software as shown in Fig. 2.

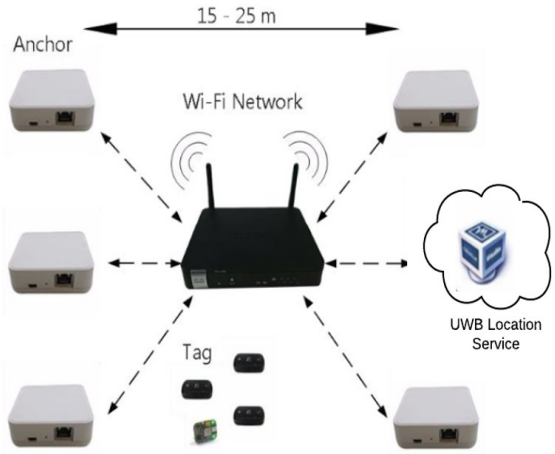

Fig. 2. The system architecture.

In order to ensure that the robots operate safely, it is necessary to enforce an invariant safety property: the tags on the robots must always maintain a minimum separation distance from each other. Although the precise required separation distance between the tags may vary (mainly due to the size of the robots used in the factory), the mechanism by which the minimum separation is achieved, using UWB localisation, remains the same. In this use case, in order to ensure that minimum separation distance between the robots is not violated, it is necessary to enforce a *safe stopping distance (SSD)* between the target and the robot as a safety invariant. The SSD in this use case is equal to the minimum separation distance plus a *decision distance (DD)* and the maximum error distance at a required confidence level. The minimum separation distance is the distance that allows the robot stop safely. Maximum error distance is an error margin due to technological limitations. Therefore, the only part of SSD that can be reflected in parameters in SLAs is the DD, which is the delay of the whole process between sensing the environment and issuing a stop command. In order to define SLA parameters to ensure that the safety invariant is fulfilled, a high level SLA for a maximum DD is defined. We then decompose the SLA into parameters that affect the DD. In this example, the SLA is divided into three pillars, i.e., communication, processing and virtualization components. Figure 3 shows this process.

A. Communication pillar

For distributed real-time systems the communication induced latency must be bounded for the system to work correctly. The application must check if it can meet the real-time requirements given the time-stamp of the data in the sensor. Even if a periodic application produces a result in time,

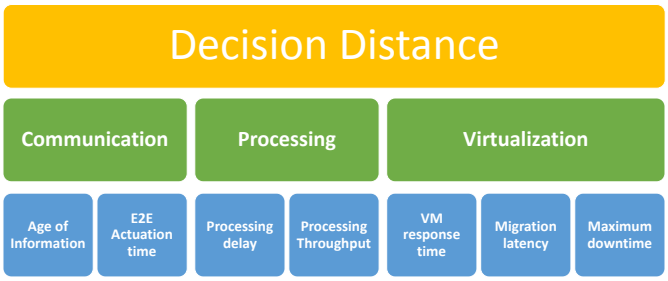

Fig. 3. The break down components for SLA.

the data available may be too old to give a correct real-time response. For this reason, the following metrics for SLAs are defined as below:

- Age of information [11]: the age of information is a metric for measuring data freshness received by the cloud/fog.
- End-to-end actuation time: time elapsed between the sensing of an event in real world and the response to it (e.g., actuation).

B. Virtualization pillar

SLA parameters with respect to virtualization (both communication and computation) to maintain minimum separation distance are as follows:

- Application response time: the response time of applications that reside on the cloud or fog.
- Migration latency: in case there is an application migration between two servers in a data center or between two data centers, the delay of migration should be bounded.
- Maximum downtime: in case of an application migration, there will be a time where the application is not executing, known as *downtime*, which should be bounded.

C. Processing pillar

We consider the aspect of the data processing that needs to take place in order for the positioning system to be updated and potential breaches of the minimum separation distance to be detected. We model the UWB location service as a streaming event-based system where every position update causes a continuous query on the minimum separation distance to be evaluated. The parameters are defined as below:

- Processing delay: the delay between receiving a position update and updating a dynamic map to find any potential breaches in the minimum seperation distance.
- Processing throughput: the number of events that the system is capable of handling per unit of time, without penalizing the processing latency.

The defined parameters should be defined in the SLA with a maxium and minimum acceptable values. Violation of each parameter from the defined range leads to violation of the defined safety variant.

VII. Concluding remarks

In this paper, we suggested an approach to derive and refine SLAs to use in safety assurance of factories. Moreover, we showed that a safety property can be decomposed into various parameters in an SLA with possible dependencies among different components using a case study. In the use case we show three pillars, however in other industrial applications these pillars may vary. The proposed approach is an initial attempt to address challenges related to integrating safety in SLAs. Ongoing work aims at investigating the dependencies of parameters and their reflections to SLAs in such an approach. Moreover, we are aiming at proposing a formal way of capturing the dependencies, possibly developing a supporting tool so to show whether, for each parameter instantiation, the safety invariant is violated.

Acknowledgements

This work is supported by SSF foundation via the project Future Factories in the Cloud (FiC) and in the context of XPRES framework. The authors have equal contributions and listed alphabetically.

References

[1] O. Givehchi, H. Trsek, and J. Jasperneite. Cloud computing for industrial automation systems - a comprehensive overview. In *18th Conference on Emerging Technologies Factory Automation*, September 2013.

[2] P. Persson and O. Angelsmark. Calvin – merging cloud and iot. *Procedia Computer Science*, 52:210 – 217, 2015. The 6th International Conference on Ambient Systems, Networks and Technologies (ANT-2015), the 5th International Conference on Sustainable Energy Information Technology (SEIT-2015).

[3] T. Goldschmidt, A. Jansen, H. Koziolek, J. Doppelhamer, and H. P. Breivold. Scalability and robustness of time-series databases for cloud-native monitoring of industrial processes. In *7th International Conference on Cloud Computing*, June 2014.

[4] T. Hegazy and M. Hefeeda. Industrial automation as a cloud service. *IEEE Transactions on Parallel and Distributed Systems*, October 2015.

[5] Editor: D. Kyriazis. Cloud computing service level agreements - exploitation of research results. *European Commission Directorate General Communications Networks, Content and Technology Unit E2 – Software and Services, Cloud*, 2013.

[6] S. Mubeen, S. Abbaspour Asadollah, A. Papadopoulos, M. Ashjaei, H. Pei-Breivold, and M. Behnam. Management of service level agreements for cloud services in iot: A systematic mapping study. *Journal of IEEE Access*, August 2017.

[7] M. Villari, M. Fazio, S. Dustdar, O. Rana, and R. Ranjan. Osmotic computing: A new paradigm for edge/cloud integration. *IEEE Cloud Computing*, November 2016.

[8] O. Jaradat, I. Sljivo, I. Habli, and R. Hawkins. Challenges of safety assurance for industry 4.0. In *2017 13th European Dependable Computing Conference (EDCC)*, pages 103–106, Sept 2017.

[9] O. Jaradat, P. Graydon and I. Bate. An approach to maintaining safety case evidence after a system change. In *Proceedings of the 10th European Dependable Computing Conference (EDCC)*, UK, 2014.

[10] Ivan Walulya, Yiannis Nikolakopoulos, Vincenzo Gulisano, Marina Papatriantafilou, and Philippas Tsigas. Viper: Communication-layer determinism and scaling in low-latency stream processing. In *Euro-Par 2017: Parallel Processing Workshops*, pages 129–140, Cham, 2018. Springer International Publishing.

[11] S. Kaul, R. Yates, and M. Gruteser. Real-time status: How often should one update? In *2012 Proceedings IEEE INFOCOM*, pages 2731–2735, March 2012.