Towards Comprehensive Safety Assurance in Cloud-based Systems

Oleg Oleinichenko, Christian Drabek, Anna Kosmalska
Frauhofer Insitute for Cognitive Systems IKS
Munich, Germany
email: firstname.lastname@iks.fraunhofer.de

***Abstract*—When a system malfunctions or a required service is not provisioned in a timely manner, this can lead to human injuries or fatalities. Increasingly, safety-critical system operation relies on offloading of functions into the cloud, including real time ones. For this reason, the cloud-based systems that are involved must exhibit a high degree of dependability. Thus, to determine a system's dependability, a comprehensive safety assurance process is needed to allow integration in a development process allowing iterative improvement to tackle complexity and changing requirements. Based on the principles of adaptivity and flexibility we propose a 3-leveled safety analysis process for building up a necessary resilience against disruptions and failures of various scale, nature and operation context dynamism. A multilevel genuine combination of traditional and contemporary safety methods is a key to provide necessary system resilience in a cloud. The right composition, expected yield and applicability of methods that will be best suited for cloud context is a subject of our research.**

***Index Terms*—Safety; Resilience; Dependability; Cloud-based; CPS; RTS; IoT.**

I. Introduction

An ongoing evolution towards an increasingly integrated world comes along with modern Cyber-Physical Systems (CPS) applications paving the way of an ubiquitous automation in the current digital transformation. Contemporary cloud-systems already serve a myriad of services of different purposes in smart cities, agricultural domains, vehicular systems, industrial automation, health-care, robotics, etc [1]. Such applications have certain dependability requirements that need to be fulfilled. Cloud-offloaded functions are scattered across different entities and exhibit dynamic and composite context of operation. They are subject to continuous internal (e.g., algorithmic faults) and external disruption (e.g., connectivity changes) [2]. The developed system must be able to absorb these adverse alterations whilst satisfying its design goals and timeliness requirements for critical Real-Time System (RTS) applications. However, dynamics of operation, complexity and often limited resources [3] pose particular challenge to system's reliability, hence requiring an elaborated resilience assurance strategy [4].

Dependability is the capability to avoid failures that are more frequent and more severe than what is deemed acceptable and ability of a system to supply trusted and available services. There are many dimensions that should be considered when analyzing whether a cloud-based solution is dependable, such as availability, reliability, performability, maintainability. At the same time, there are different ways to implement a dependable system, for instance using fault tolerance algorithms and redundancy techniques [5]. Some of the proposed dependability measures could turn out to be inefficient in a dynamic cloud-based environment, some might provide insufficient coverage or even conflict with other measures diminishing the deemed safety gain [6]. In the end, the system must be performant, whilst providing sufficient level of safety that does not overly burden resource-constrained nodes. Satisfying these requirements in a dependable manner remains largely underexplored. A comprehensive safety assurance process is missing. Although there are some processes proposed for cloud computing and fault tolerance in networks, they do not tackle the problem systematically [7].

In this paper, we propose a 3-leveled comprehensive safety assurance process for resilient and efficient system architectures development that embraces the aspects of design-time as well as run-time assurance aiming to provide high safety coverage of a system. We explain what each level of the process is focused on, its main purpose, methods are meant to be utilized to provide an insight on how it can be applied. The rest of the paper is structured as follows. Section 2 outlines the related work in this area. Section 3 presents the discussion on a proposed comprehensive safety assurance process hierarchy, main steps and validation approach. Section 4 concludes the paper and gives a future outlook.

II. Related Work

This section draws attention to related work of other authors that emphasize the lack of a structured approaches for safety assurance in emerging cloud-based systems suggesting the arising problems, possible ideas on approaching them and shed the light on relevant scientific tracks to study.

The need for a new safety assurance approach is largely dictated by evolution of Internet of Things (IoT) systems. Christos Tsigkanos et al. discuss a road-map [4] for a resilient IoT defining a structure of 4 maturity levels with various degrees of dependability requirements assurance. They discuss such challenges as resilience in a decentralized adaptive edge-centric system, nodes heterogeneity, components coordination for failure robustness, run-time assurance. Instead of traditional self-contained safety mechanisms, they assert that resilience must be built into every core IoT component and thereby unveil the track for future research in this direction.

One step further, they postulate a crucial requirement for any type of an emergent solution - formally analyzable and verifiable models to enable reasoning, starting from the early stages of design to models at run-time. Obtaining assurances on reliable requirements satisfaction in an environment that may adversely change at the system's run-time operation is seen as one of the major obstacles.

The importance of a safety process for complex cloud-based systems is emphasized by a new paradigm in edge-centric cloud-based services - Fog computing. Zeinab Bakhshi et al. present a roadmap [6] for dependability research in this area by bringing inherent system heterogeneity into perspective as a new challenge. Writing about the gap in fulfilling dependability requirements, they claim current approaches differ significantly from each other in terms of the types of faults and errors they address and the method applied leaving the resilience assurance an open question. They admit that relying only on traditional methods and approaches used before to assure safety might be highly inexpedient or even infeasible. They list several important trade-offs that become of paramount importance for novel applications in cloud-based systems, such as resource utilization and fault tolerance or reliability and timeliness. This leads to conclusion that novel cloud-based application will require a range of safety-to-performance Key Performance Indicators (KPIs) as a part of the development process for finding the right balance in each specific case.

Striking the right balance in presence of many challenges in dynamic contexts of cloud-based systems is further discussed by Jose Moura et al. [2]. They stress the importance to pay more attention to the scenario of multiple threats simultaneously affecting the normal operation of a fog system. Therefore, they urge studying diverse operational aspects not only at system run-time but initially at its design, including the protection of the physical infrastructure against weaknesses recently reported. Each resilience-enhancing measure should be validated on whether it inflicts negative effects on the system operation before adopting it stressing the importance of the simulation-oriented practices.

Worth to mention a formalized and easy to grasp definition of a bimodal resilience philosophy for cloud-based CPS aptly described by Somali Chaterji et al. [8]. They outline a roadmap for resilient CPS defining two distentions of thinking about it: *resilience-by-design* and *resilience-by-reaction*. The former encompasses various design and development techniques, the latter refers to techniques that are invoked at run-time only. Resilience-by-design makes the system robust against what can be predicted from existing knowledge, it remains blind to future events though. Resilience-by-reaction in contrast allows recovering after the occurrence of any disruption without over-fitting during the system's design engineering. They propose to use those in a genuine balance as a part of one process and in accordance to the specific operating conditions and application requirements to achieve maximum resilience minimizing residual risk to permissible minimum.

The main focal point of our work is to identify the ways to leverage existing former and modern techniques to propose a process with a cohesive methodology to be able to find such a mixture and build the system that can maintain resilience in the face of disruption, especially in the absence of a central control.

III. Safety Assurance Process

Bringing locally-driven distributed system functionalities into the cloud leverages a great advantage of centralized processing and orchestration. This process fosters self-organization of the original system yielding greater efficiency and flexibility of operation. This, however, increases focus on operational aspects of a such system. Cloud-based system has distinctive characteristics, such as heterogeneous software stacks, often consists of resource-constrained units shared dynamically among other users, laggy or intermittent connectivity between the nodes, its functional devices are often spatially distributed in a composite environment that alters with a diverse rate of change, etc [6]. Enforcing dependability of such system operating in stringent RTS time constrains under emergent cloud environment largely remains the challenge for today's systems. Unlike traditional systems where failures are caused by faults in a specific component of a deterministic system, in cloud-based systems, inherent complexity and unpredictability of the ever-changing environment, aggravated by timeliness requirements, is the main reason of non-sufficient safety measures provisioning [9]. A reference process is therefore important to manage the growing complexity of development activities for cloud-based systems. First of all, such a process must define a clear sequence of procedural phases, their purpose, boundaries and possibly constituent technical methods to enable its implementation. Altogether, the composed process must begin already in the early development stage [4] for safety requirements derivation to influence functional system design choices which have great impact on the overall safety outcome, i.e., the ability of the system to be safe per se. The underlining structure for such process is presented on Figure 1 and can be seen as pyramid-like process targeting different scale of the dependability problems starting from the most fundamental at the bottom

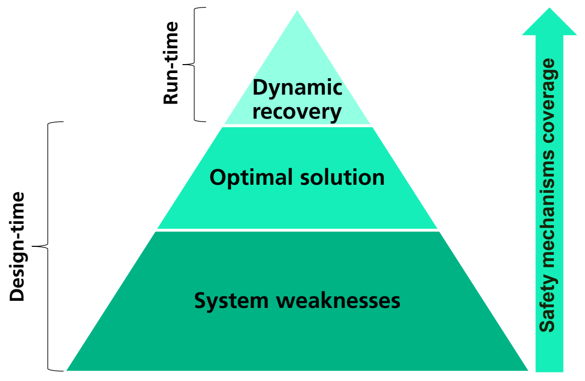

Fig. 1. Underlying structure for the dependable system architecture development (simplified form)

and ending with more sophisticated and research intense at the top. The lower two levels are proactive and confined within design-time domain methods, whereas the third is reactive and consists of run-time domain methods.

A. Composite Approach in Risk Reduction

1) "Level 1 - System weaknesses": It is possible that initial architecture is functionally deficient and contains missing or wrong elements and interfaces. A big aggregation of hazards of a similar root is called *weakness* and this must be addressed before any other minor hazard. This level aims at intercepting large-scale problems of a systematic design nature ensuring full functional sufficiency, i.e., provides the requirements to create a robust system architectural backbone. This level assumes usage of qualitative techniques, such as HAZOP [10] with custom Guide-phrases, STAMP/STPA [11], FMEA [12] and others to provide systematic constellation of failure modes and their consequences onto preliminary architecture draft. The further steps assume proper risk assessment steps (e.g., Hazard Analysis and Risk Assessment(HARA)) to rank the hazards and split them into safety and performance problems.

2) "Level 2 - Optimal solution": In many cloud-based systems, there are many alternatives of defining system operation using various physical and virtual realizations. In order to find the best possible solution, many functional allocations, resource utilization schemes and connectivity scenarios must be evaluated. Through iterative comparison of various compositions, the most optimal (as defined by system goals) utilization pattern is found providing necessary safety, performance and scalability requirements compliance. This level assumes usage of quantitative techniques based on simulation testbeds or Model-Based Dependability Analysis (MBDA) tooling [13] (eg. OSATE [14]) with numeric data for building composite error models.

3) "Level 3 - Dynamic recovery": Not all problems can be effectively resolved using only proactive methods. This is particularly true for cloud-based systems, where overloading its constrained edge nodes excessively with redundancy techniques will consume plenty of resources diminishing the performance gain making the whole system highly inexpedient in use or even subject to malicious intrusions due to added vulnerabilities. This level is aiming at developing an adaptive detection and recovery counter-reaction at run-time against foreseeable and unforeseeable degradation based on monitored attributes across the system [9]. Operation environment and system contextual factors are thoroughly investigated to identify what attributes must be monitored and how to react thereon. This level assumes usage of simulation testbeds and actual cloud-based systems to collect data for that purpose.

B. Detailed Process Outlook

A detailed stepwise procedure of the aforementioned process application is described on Figure 2. A list of performance and safety goals serves as the initial input for the process (step 0) that shapes the entire design. In line with given confidentiality policy, the goals define the tasks and qualities of the system as well as potential hazards and risks. They are used to derive the system requirements (step 1), starting from the top-level tasks that define the overall system architecture and main sub-components. Systematic analysis identifies (step 2) failure modes and system weaknesses in order to refine the requirements (step 3). This is done by suggesting proper countermeasures that are carried out until the level of weaknesses is satisfactorily low. This process, called weakness-driven requirements refinement, helps identify flaws, such as safety hazards, failures, security threats or breach of performance thresholds. Countermeasures are then integrated at the subsystem level. The system requirements limit the options of the configuration space model (step 4), a set of tools to (semi-)formally describe, analyze, and collect information about the system. An integral, yet distinct part of this approach is a safety model, which can be used to carry out safety-oriented analysis and set limits to performance-related optimizations. The models can be instantiated (step 5) and narrowed down in order to further analyze selected weaknesses, among other things. Exploration of the configuration space (step 6) provides a set of solutions. These are essentially system configurations that meet the requirements, from which optimal system configurations are identified and evaluated. Requirements compliance is only one aspect of the success. Detailed analyses and simulations must be used to validate the solutions (step 7) and demonstrate that they fulfill the initial goals of the system and ensure dependability. Validation also allows fault forecasts to be provided and helps to identify critical scenarios. If validation identifies new weaknesses, suitable countermeasures are applied. The system requirements can then be refined for the next iteration. An essential result of the design process is monitoring and recovery concepts, which can be integrated into the operating system. By monitoring properties defined by the requirements (step 8), the state of the system can be determined. If necessary, recovery plans defined by optimal solutions (step 9) for the identified contexts can then be triggered. More advanced concepts and prototypes, including necessary monitors and recovery mechanisms, can help in constituting the system's self-awareness - its ability to determine its own state and its environment - to detect faults and identify possible actions.

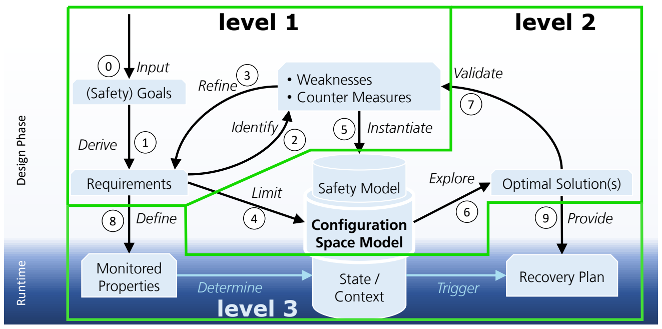

Fig. 2. Underlying structure for the iterative design and evaluation process of the dependable system architecture (detailed form)

C. Validation Approach

To examine adaptive countermeasures alongside other proactive safety mechanisms being built-in earlier, the system operation is adjusted from one extreme to another with a concurrent fault injection. Thereby, one can verify the effectiveness of devised interoperating countermeasures for a given safety-critical scenario to further evolve the safety backbone. Since each application or use-case can have different safety and performance requirements, a *safety-to-performance ratio* must be provided as an insight to track the impact upon environmental and system alteration to strike the desired magnitude in the operation envelope. The particular definition(s) of the *safety-to-performance ratio* is specified individually depending on the use-case and KPI of interest and it mostly depends on finding the right balance between design-time and run-time methods. Additionally, the system countermeasures must be tested progressively with each new process level executed (bottom-up manner) to track the yield of each level as compared to their holistic combination. Relative and absolute scales of high-level safety and performance KPIs (e.g., for automotive domain: collisions, safety distance violations, average velocity, maneuvers executed, etc.) must be used to track the progress gain. For a system to be testable, the minimal thresholds of given KPIs must be defined for selecting required system analysis iterations as well as to ensure their minimal necessary fulfillment at the end of the process. Hence, not only optimal safety-to-performance compromise is found but also significance and applicability of selected safety methods is explored for each individual use-case. Ultimately, this will foster a development of optimal process implementation finding out what methods, in what proportion, to what extent yield the best results in each given case in a cloud environment.

IV. Conclusion and Future Work

In order to achieve resilience in a cloud-based system, understanding and systematically managing dynamic behavior and distributed operations is the key. Hence, all elements of the architecture design process must be accompanied by appropriate safety activities aforehand. Each system design should involve applying an iterative approach to refine the safety goals in coordination with new design decisions. At each level of refinement, assumptions made in the design are explicitly stated and analyzed, so that they may also be later validated. The ultimate system to be safe must incorporate not only robustness aspects covering foreseeable events, but also resilience to cope with unforeseeable evens as well. In this paper, we analyzed the importance of the having a comprehensive process that could address the safety problems of a different scale in a systematic manner. We presented our vision of the 3-leveled process by defining and discussing the aim of each level and the possible methods that could be utilized therefor. We advocate the need of leveraging traditional methods with sophisticated simulation practices within one process in a right proportion and selection to provide the maximal safety coverage to deal with the problems of different scale throughout the entire system development. Thereby, resilience is built into system from a very early phase. We suggest to track a safety-to-performance ratio for optimal decision making when choosing the alternative countermeasures compositions balancing design and run-time mechanisms. Moreover, we underline the importance of adapting the process to particular use-case by exploring applicability and gain of each of its levels' methods. Only a genuine combination of different safety techniques alongside with a cyclic relevant properties monitoring can lead to sufficient safety coverage provisioning in complex and highly dynamic cloud-based systems. Development, testing and refinement of the reference process is an active ongoing activity. Future work will encompass discussions on the application of the proposed process with safety analysis strategies' efficiency evaluation for cloud-based applications.

Acknowledgment

This work was funded by the Bavarian Ministry for Economic Affairs, Regional Development and Energy as part of a project to support the thematic development of the Institute for Cognitive Systems.

References

[1] S. Aggarwal and N. Kumar, "Fog Computing for 5G-Enabled Tactile Internet: Research Issues, Challenges, and Future Research Directions.", Mobile Networks and Applications, 2019 Nov 20, pp.1-28.

[2] J.Moura and D. Hutchison, "Fog computing systems: State of the art, research issues and future trends.", Journal of Network and Computer Applications, 2020 Aug, p.102784.

[3] S. M. Oteafy and H. S. Hassanein, "IoT in the fog: A roadmap for data-centric IoT development.", IEEE Communications Magazine., 2018 Mar 15,56(3), pp.157-163.

[4] C. Tsigkanos, S. Nastic and S. Dustdar, "Towards resilient Internet of Things: Vision, challenges, and research roadmap.", IEEE 39th International Conference on Distributed Computing Systems (ICDCS), 2019 Jul 7, pp. 1754-1764.

[5] A. Avizienis, J.C. Laprie and B. Randell, "Fundamental concepts of dependability.", University of Newcastle upon Tyne, Computing Science, 2001 Apr 19, pp. 7-12.

[6] Z. Bakhshi and G. Rodriguez-Navas, "A preliminary roadmap for dependability research in fog computing.", ACM SIGBED Review, 2020 Jan 6, 16(4), pp.14-19.

[7] L. Paradis and Q. Han, "A survey of fault management in wireless sensor networks.", Journal of Network and systems management, 2007 Jun 1, 15(2), pp.171-190.

[8] S. Chaterji, et al., "Resilient cyberphysical systems and their application drivers: A technology roadmap.", arXiv preprint arXiv:2001.00090, 2019 Dec 20.

[9] S. Burton, J. McDermid, P. Garnet and R. Weaver, "Safety, complexity and automated driving – holistic perspectives on safety assurance.", IEEE Computer Special Issue on Safety, Security, and Reliability of Autonomous Vehicle Software, in press, 2021.

[10] IEC-International Electrotechnical Commission and others, IEC 61882, 2001.

[11] N. Leveson and G. Nancy, "Engineering a safer world: applying systems thinking to safety.", The MIT Press, 2016, p. 560.

[12] IEC-International Electrotechnical Commission and others, IEC 60812, 2003.

[13] S. Kabir, "An overview of fault tree analysis and its application in model based dependability analysis.", Expert Systems with Applications, 2017 Jul 1, 77, pp.114-135.

[14] J. Delange and P. Feiler, "Architecture fault modeling with the AADL error-model annex.", In 2014 40th EUROMICRO Conference on Software Engineering and Advanced Applications, 2014, Aug 27, pp. 361-368.