





Incorporating Resilience to Platforms based on Edge and Fog Computing

Santiago Medina ⁽¹⁾ , Diego Montezanti ⁽¹⁾ , Lucas Gómez D’Orazio ⁽¹⁾, Evaristo Compagnucci ⁽¹⁾, Armando De Giusti ⁽¹⁾⁽²⁾ , Marcelo Naiouf ⁽¹⁾ 

¹ Instituto de Investigación en Informática LIDI (III-LIDI),
Facultad de Informática, Universidad Nacional de La Plata – Comisión de Investigaciones
Científicas de la Provincia de Buenos Aires

² CONICET – Consejo Nacional de Investigaciones Científicas y Técnicas

{smedina, dmontezanti, degiusti, mnaiouf}@lidi.info.unlp.edu.ar
{evaristogaston, lucas.dorazio}@alu.ing.unlp.edu.ar

Abstract. In recent years, Internet of Things (IoT) has become extremely popular because of its ability of sensing information from the environment and processing it in the cloud. Edge and Fog Computing are new paradigms that aim to localize some of the processing near the sensors, helping to cope with high communications latencies and bandwidth bottlenecks. As Wireless Sensor Networks (WSN) and Fog nodes are prone to failure, affecting system reliability and performance, the implementation of resilience strategies becomes essential to ensure reliable delivery of data and system availability during interruptions or in the presence of faults. In this article, we present three lines of research in progress: Redundant Image Processing, Integration of a WSN with an IoT Platform for Intelligent Control and Resilient Monitorization and Control of Robots. We aim to incorporate resilience mechanisms to platforms that integrate Edge, Fog and Cloud Computing, and to evaluate the proposed solutions in terms of the coverage achieved, processing and communication times and power consumption.

Keywords: Edge Computing, Fog Computing, Cloud Computing, Resilience, Internet of Things, Wireless Sensor Networks.

1 Introduction/Motivation

In recent years, the Internet of Things (IoT) has gained a lot of attention because of its characteristic of typically enabling the connection of a significant number of sensor devices that sense information from the environment and share it to a cloud service for processing [1]. This has been extensively used to develop smart applications, such as traffic management, smart houses, monitoring of natural events and human health [2]. Due to the growing popularity of the IoT, the number of Internet-connected devices has increased significantly. As a result, a huge amount of network traffic is generated, which may lead to bottlenecks, and eventually generate limitations in terms of

communication latency with the cloud and network bandwidth [3], so the traditional cloud-based infrastructures are not enough for the current demands of IoT applications [4]. To deal with these issues, in recent years, the paradigms of Fog computing and Edge computing were proposed to alleviate these limitations, by moving some processing capabilities closer to the network edge and away from the central cloud servers [1]. This allows to distribute the computations of the IoT data, and to reduce the communication latency [3].

In IoT systems, data are acquired by wireless sensor networks (WSNs), which are deployed in harsh environments where weather and other factors can cause node failure. In addition, the IoT devices and the nodes in WSNs are heterogeneous, highly distributed, reliant on wireless communications and generally powered by batteries, which are prone to failure [2]. All of this makes the recovery of devices, and the creation of a pattern for Fault Tolerance in IoT, especially difficult [4].

In the Fog layer, some of the computing nodes may be unreliable and fail unexpectedly, affecting the system's reliability. For this reason, mechanisms for handling node failures become essential but are also especially challenging, because when a Fog node fails, moving the computations to neighbor nodes (or to the cloud) may increase the communication latency, and thus affect the system performance [3]. Consequently, the design of an effective fault tolerance mechanism is crucial to ensure reliable delivery of data and to warrant the system availability during interruptions of any kind, or in the presence of faults [2].

Resilience can be introduced at different architectural layers because a fault can occur at any of the layers. Either sensors and actuators, network, or computation and storage nodes, can perform erroneously in their layers [5].

2 Main goals

Our main goal in this project is to incorporate mechanisms aimed at obtaining a certain degree of resilience on a platform that integrates sensor networks with levels of Edge, Fog and Cloud Computing (see Fig. 1).

To do this, we propose to evaluate different scenarios of communication failures on such a 3-tier architecture. The end nodes are sensors connected to microcontrollers, robots, drones, or other devices that can perform tasks remotely controlled by a server.

Faced with possible communication failures between the different levels of the architecture, it is planned to redundantly process information in each of the layers, so that the system can maintain a degree of responsiveness. Through the measurement of response times and energy consumption, we hope to obtain criteria to evaluate the convenience of carrying out the processing at a specific level, but also of having the alternative of transferring it to another layer in presence of a communication failure, in order to provide fault coverage.

In the case of communication failures between robots (or end devices) and servers, validating the status is proposed. This validation can be either periodical or event-driven (like the fulfillment of partial objectives or the reaching of intermediate posi-

tions), by exchanging round trip messages with the server. Therefore, the device could return to the last consistent state and wait for the link to restore. For this, the valid states must be stored in the final device, so that the recovery can be autonomously made.

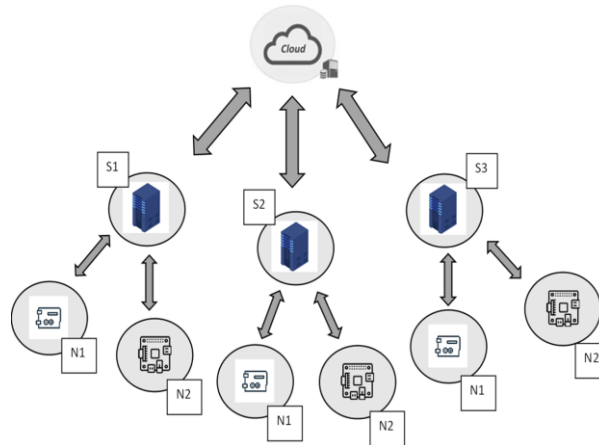


Fig. 1. 3-layer architecture

Accordingly, in both cases, we aim to improve the integrity of the system, allowing it to maintain some basic functionality in the presence of communication failures, or at least to remain in a safe state, by adopting the most appropriate strategy for each case, similar to that explored in [6]. The deliberate provocation of communication failures will allow us to evaluate the robustness of the proposed solutions.

3 Works in progress

3.1 Redundant Image Processing in 3-layer architecture

In this line of research, the focus is on the conceptual integration of resilience aspects in an architecture with Edge, Fog and Cloud Computing levels.

In the 3-layer platform that we propose, the Edge level consists of a node built from an ESP32-CAM [7], which takes pictures when movement is detected. The pictures are transmitted to a Fog server, which can pre-process them and, in turn, transmit them to the Cloud to be able to take a concrete action from the Internet. As each of the three levels has a computational capacity, certain data processing tasks can be performed alternatively in any of them. This makes it possible to implement resilience strategies based on redundancy.

There are two possible scenarios: in normal operation, our goal is to evaluate processing times and communication latencies, in order to decide where to process the

images to detect people or objects. However, in the presence of communication failures, we aim to take advantage of the computing power, whether in the Fog server or in the sensor node itself, to redundantly process the images to maintain functionality. As a consequence, we will be able to characterize the system performance in terms of processing times at each level, coverage obtained against failures and energy consumption.

3.2 Integration of a WSN with an IoT Platform for Intelligent Control of Classrooms

In this line of research, we are studying the deployment and configuration of the IoT platform ThingsBoard [8], to be integrated with a Wireless Sensor Network (WSN) built with CO₂ sensors and energy consumption measurement and control nodes, in the context of a university building with classrooms that are monitored and remotely controlled [9].

The CO₂ sensors in each classroom are connected via WiFi to a Raspberry Pi (which is at the Fog level) that reads the data (via HTTP), and makes some pre-processing before transmitting them to the ThingsBoard server. This Fog server can maintain centralized monitorization of the CO₂ levels across the whole building, and activate alerts in case of undesired values.

The energy consumption measurement and control nodes in each classroom are directly connected via WiFi to the Fog ThingsBoard server, publishing information in MQTT [10] topics. The server can maintain centralized monitorization of power consumption and take concrete actions, such as remotely turning off lights or air conditioning equipment when the classroom is not being used.

3.3 Resilient Monitorization and Control of Robots

Our goal in this line of research is to incorporate resilience to a system that uses Lego Mindstorms EV3 robots [11] to perform specific tasks controlled by a server. We aim to develop an application that controls the functions of the robots through resilient communications.

Currently, we are exploring 3 alternative strategies, in the attempt to maintain the consistency of the system in the presence of failures. Each of them has particular characteristics as regards to detection latency, coverage and workload to be rerun.

1. The server communicates the whole information of the task to be performed or the path to be followed. The robot must store the path and follow it autonomously, communicating with the server upon completion. If the connection is lost during the trajectory, the task cannot be validated. The only resilience strategy is to restart from the beginning if the final point does not match with the expected one. This variant minimizes communication load but maximizes the detection latency.
2. The server communicates partial information of the task or the path. The robot must store the partial path and communicate with the server when reaching the target point, which in turn validates the completed chunk. If the connection is lost, the robot has to use the stored information to return to the starting point, which is the

last valid one. Compared with the previous variant, this one involves more communications but improves the detection latency.

3. The server communicates partial information of the task or the path. In addition, a periodic keepalive signal is transmitted via an independent socket or a different MQTT topic. The robot must store the partial path and communicate with the server when reaching the target point, which in turn validates the completed chunk. In the meanwhile, the status of the connection is monitored with periodical messages. Consequently, if the connection is lost, the robot has to use the stored information to return to the last valid point. This latter alternative involves frequent messages but minimizes the detection latency.

4 Conclusions

Although the research is in an incipient stage, this work proposes to implement mechanisms to provide robustness to a system against communication failures, and to determine the performance of the different resilience strategies in terms of the coverage achieved, processing and communication times and power consumption.

References

1. Mohan, N., & Kangasharju, J. (2016, November). Edge-fog cloud: A distributed cloud for internet of things computations. In 2016 Cloudification of the Internet of Things (CIoT) (pp. 1-6). IEEE.
2. Tong, Y., Tian, L., Lin, L., & Wang, Z. (2020). Fault Tolerance Mechanism Combining Static Backup and Dynamic Timing Monitoring for Cluster Heads. *IEEE Access*, 8, 43277-43288.
3. Karagiannis, V., Desai, N., Schulte, S., & Punnekkat, S. (2020). Addressing the node discovery problem in fog computing. In 2nd Workshop on Fog Computing and the IoT (Fog-IoT 2020). Schloss Dagstuhl-Leibniz-Zentrum für Informatik.
4. Bierzynski, K., Escobar, A., & Eberl, M. (2017, May). Cloud, fog and edge: Cooperation for the future?. In 2017 Second International Conference on Fog and Mobile Edge Computing (FMEC) (pp. 62-67). IEEE.
5. Agrawal, A., & Toshniwal, D. (2021). Fault Tolerance in IoT: Techniques and Comparative Study. *Asian Journal For Convergence In Technology (AJCT) ISSN-2350-1146*, 7(1), 49-52.
6. Montezanti, D., Rucci, E., De Giusti, A., Naiouf, M., Rexachs, D., & Luque, E. (2020). Soft errors detection and automatic recovery based on replication combined with different levels of checkpointing. *Future Generation Computer Systems*, 113, 240-25.
7. Espressif Web Page, https://www.espressif.com/en/news/ESP32_CAM, last accessed 2022/04/28.
8. ThingsBoard Homepage, <https://thingsboard.io>, last accessed 2022/04/27.
9. De Antueno, J., Medina, S., De Giusti, L., & De Giusti, A. (2020). Analysis, Deployment and Integration of Platforms for Fog Computing. *Journal of Computer Science and Technology*, 20(2), e12-e12.
10. MQTT Homepage, <https://mqtt.org/>, last accessed 2022/04/15.
11. ev3dev Homepage, <https://www.ev3dev.org/>, last accessed 2022/04/08.