D1.5 – Final analysis of real-time big data analysis aspects impacting on standards and industrial initiatives

Version 1.0

Document Information

Contract Number	780622
Project Website	https://class-project.eu/
Contractual Deadline	M29, May 2020. (Due to COVID situation this deliverable has been submitted on M31, July 2020)
Dissemination Level	PU
Nature	R
Author(s)	Roberto Cavicchioli (UNIMORE)
Contributor(s)	Paolo Sanfelice (MAS), Jorge Montero (ATOS), Eduardo Quinones (BSC)
Reviewer(s)	BSC

Notices: *The research leading to these results has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No "780622".*

© 2018 CLASS Consortium Partners. All rights reserved.

Change Log

Version	Author	Description of Change
0.1	Roberto Cavicchioli (UNIMORE)	Initial Draft
0.2	Jorge Montero (ATOS)	FIWARE contribution
0.3	Roberto Cavicchioli (UNIMORE)	Update to OpenFog
1.0	Elli Kartsakli (BSC)	Final Version, Ready to EC revision

Table of contents

1 Executive Summary 4
2 Standards and Open Initiatives 4
2.1 OpenFog Applied to Smart Cities and Connected Vehicles.......................................4
2.2 ISO26262 5
2.3 FIWARE 5
3 References..6

1 Executive Summary

This deliverable covers the work done during the second phase of the project within WP1. The deliverable spans 20 months of work and includes the work done in Task 1.3 "*Impact of CLASS on standards and initiatives*" to reach milestone MS3.

Task 1.3 is focused on the evaluation of the impact that real-time big data analytics have on the properties of the safety standards ISO26262 [1] and the FIWARE [2] and OpenFog [3] open initiatives identified in Task 1.1. Regarding ISO26262, this task has verified that the techniques proposed in CLASS fit within the requirements imposed by the standard when developing automotive systems relying on real-time big-data analytics. Similar analyses have been conducted on the FIWARE and OpenFog initiatives.

A comprehensive consideration regarding the impact of big-data techniques applied to edge and cloud platforms on the properties of these standards and open initiatives is provided in this document.

2 Standards and Open Initiatives

2.1 OpenFog Applied to Smart Cities and Connected Vehicles

Deliverable D1.1 [4] introduced the eight pillars upon which the OpenFog Reference Architecture (RA) can benefit smart cities and connected vehicles: *security, scalability, openness, autonomy, RASS (Reliability, Availability, and Serviceability/Safety), agility, hierarchy, programmability* (see Deliverable D1.1 for a description of each pillar).

During Phase 2, the following pillars have been addressed by the CLASS Software Architecture ecosystem: *scalability, programmability, autonomy, agility* and *hierarchy.*

Programmability

The CLASS software architecture includes two well-known programming models upon which the workflows implementing the different data analytics methods across the compute continuum are described, i.e., the map/reduce and the tasking execution models. These two models are complementary and exploit two different types of parallelism: the former exploits structured parallelism and the latter unstructured parallelism. Unfortunately, a programming framework capable of supporting the two models in the same tool does not exist.

With the objective of addressing this important pillar, CLASS has integrated the two models into the same ecosystem. Concretely, we have extended COMPSs to support the map/reduce operations provided by Spark. Deliverable D2.5 [5] provides a detailed description of this integration.

Scalability, autonomy, agility and hierarchy

These highly interconnected pillars refer to the capability of the system to include "independent" computing and storage nodes located within the vehicles and the MASA with decision-making capabilities. The core of the CLASS software architecture includes an *advanced scheduler component* that constantly monitors the status of the computing network. This component has the capability to distribute the workflows based on the changing conditions of the computing network and the real-time properties of the system. If the computing network changes (e.g., a new computing node appears in the network, or the connection to a computing node is lost), the scheduler is changed automatically. This feature is fundamental to guarantee the real-time requirements and the correct operation of the system, as each computing nodes will have the capability of responding without depending on other nodes. Deliverable D2.6 [6] provides a detailed description of the scheduler component.

If the scheduler component determines that the real-time requirements are not fulfilled, the system will be informed so counter-measures can be applied (e.g., scale-up computing resources).

Overall, the CLASS software architecture addresses each of the pillars as follows:

1. *Scalability.* The CLASS software architecture will be constantly monitoring the computing network, selecting the most convenient computing node(s) to fulfil the real-time requirements of the application and scaling up and down the resources when needed.
2. *Autonomy*. The CLASS software architecture is independent of the computing nodes.
3. *Agility.* The architecture has the capability to adapt the workflow to the actual execution conditions and inform the system if the real-time requirements will be fulfilled or not.
4. *Hierarchy.* The scheduling decisions are taken in a hierarchical environment composed of distributed and heterogeneous computing elements.

2.2 ISO26262

As already discussed in D1.3 [7], ISO26262 is the international standard for functional safety of electrical and/or electronic (E/E) systems in production automobiles defined by the International Organization for Standardization (ISO) in 2011. The same possible implementations of the CLASS software architecture mentioned there are still valid.

2.3 FIWARE

As presented in D1.1 [4] and further explained the interactions with CLASS in D1.3 [7], FIWARE is one of the best options to extend the CLASS solutions beyond the project and to be easily integrated on different smart cities with small efforts.

The approach described in D1.3 [7] is still valid, specifically for the "Smart Parking Application" Use Case, where a data model to manage the sensors attached to parking spots in a homogenized way with the help of the Context Brokers provided by FIWARE.

Summarizing, CLASS is aligned with FIWARE to be adapted to multiple smart cities using the approaches described and the previous expertise of many other applications of FIWARE in different scenarios[1] by applying standards for context data management.

[1] https://www.fiware.org/community/impact_stories/

3 References

[1] ISO, "Road vehicles -- Functional safety," ISO, Geneva, Switzerland, 2011.

[2] "FIWARE CATALOGUE," 2019. [Online]. Available: https://www.fiware.org/developers/catalogue/.

[3] "OpenFog Consortium," 2018. [Online]. Available: https://www.openfogconsortium.org/.

[4] CLASS project, "D1.1. Use Case Requirement Specification and Definition and First Description of the Sensing and Data-Sets Collected," 2018.

[5] CLASS project, "D2.5 Final release of Spark and COMPSs integration in CLASS architecture," 2020.

[6] CLASS project, "2.6 Second release of the CLASS software architecture," 2020.

[7] CLASS project, ""D1.3 – First analysis of real-time big data analysis aspects impacting on standards and industrial initiatives"," 2019.