SLA for Cloud processing: use cases for satellite image processing, disaster management and IoT

George-Valentin IORDACHE[1], Cătălin NEGRU[1], Gabriel NEAGU[2], Florin POP[1,2]

[1] University “Politehnica” of Bucharest, Computer Science Department

[2] National Institute for Research and Development in Informatics – ICI Bucharest

george.iordache@cs.pub.ro; catalin.negru@cs.pub.ro; gabriel.neagu@ici.ro; florin.pop@cs.pub.ro

Abstract: Cloud processing is very important in the context of the Cloud computing research area. Various works about Cloud processing are discussed in the literature. This work presents the characteristics of the Service Level Agreement when referring to the Cloud processing. To define and design the Service Level Agreement in relation with Cloud processing we chose three use cases that are part of three different projects that we were involved on. The three projects are: ForestMon, SPERO and ROBIN-Cloud and are further presented in this work. The design of the Service Level Agreement is done differently in the case of each project. Each project is a Cloud related one and the architecture of each project is described in this work. The SLA parameters for Cloud processing in the three cases are discussed in detail.

Keywords: Cloud processing, Cloud processing projects, Service Level Agreement (SLA), SLA parameters, SLA use cases in the Cloud.

SLA pentru procesarea în Cloud: cazuri de utilizare pentru procesarea imaginilor prin satelit, gestionarea dezastrelor și IoT

Rezumat: Procesarea în Cloud este foarte importantă în contextul ariei de cercetare despre Cloud-ul computațional. În literatură sunt discutate diferite lucrări despre subiectul procesare în Cloud. Această lucrare prezintă caracteristicile contractului de Service Level Agreement (SLA) când ne referim la procesare în Cloud. Pentru a defini și a proiecta un contract de SLA în relație cu procesarea în Cloud au fost alese trei cazuri de utilizare care sunt definite în contextul a trei proiecte diferite în care autorii au fost implicați. Numele celor trei proiecte sunt: ForestMon, SPERO și ROBIN-Cloud și sunt prezentate în această lucrare. Proiectarea contractului de SLA se face diferit în cazul fiecărui proiect. Fiecare proiect are ca temă principală procesarea în Cloud și această lucrare descrie arhitectura fiecărui proiect. Parametrii contractului de SLA în cazul procesării în Cloud, în cele trei cazuri, sunt discutați în detaliu.

Cuvinte cheie: procesare în Cloud, proiecte despre procesare în Cloud, Service Level Agreement (SLA), parametri de SLA, cazuri de utilizare a SLA-ului în Cloud.

1. Introduction

The Cloud computing model refers to the process of providing access to different type of IT resources to Cloud service customers for a negotiated cost, and the provider of the service offers the resources online, “with minimum interference” (Aljoumah et al., 2015). One of the research issues when dealing with Cloud computing paradigm is the SLA. The established contract that has two partners: the provider and the customer of the Cloud service.

In this work we present our research related to the importance of the Service Level Agreement (SLA) for Cloud processing. The SLA is a powerful contract that has the purpose of defining the interaction (expressed in parameters with different values and rules) between the Cloud service provider and the Cloud service customer. This work considers three projects (thus the associated use cases of each project) and the SLA parameters that should be defined in each of the three use cases.

We choose to research the characteristics of a Service Level Agreement in three different use cases. The presented use cases are described in terms of their purpose and their architecture. Based on these three use cases we analyse the SLA parameters that should be defined and their practical aspects.

https://doi.org/10.33436/v30i4y202007

We are mainly interested in the Cloud processing, meaning that several operations (such as satellite image processing, satellite data and in-situ data processing, data generated by robots processing) on different types of data are executed with different purposes such as:

- detect deforestation, forest fires;
- disaster prevention, detection, and actions against disasters;
- use the information generated and received by the robots for practical purposes (sending commands to the robots, obstacle detection, room cleaning by robots etc.).

In the presented use cases the processing in the Cloud is the selected option because the Cloud provides the desired number of resources, both hardware and software, for a certain price. The Cloud service providers offer (for a price negotiated between the Cloud service customer and the Cloud service provider), by using the Internet paradigm, the number and type of both physical and virtualized resources necessary for processing the different types of data described above.

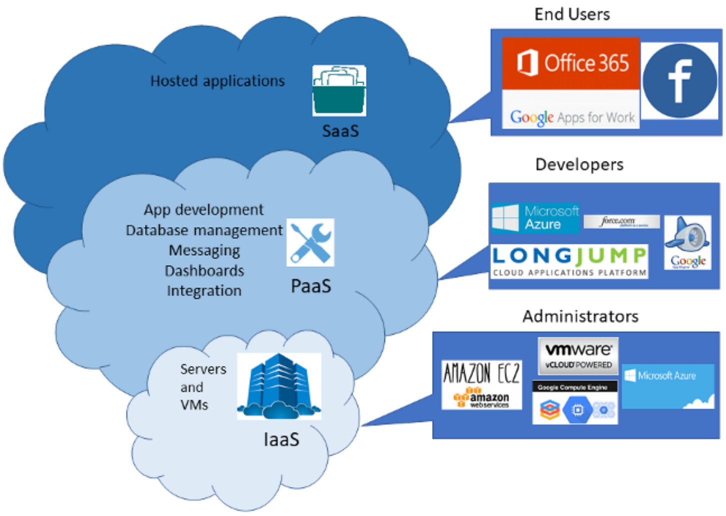

Figure 1. The key categories of Cloud computing services: IaaS, PaaS, SaaS

The three use cases show the importance of Cloud processing in different situations. In the case of the first project that it is discussed (the ForestMon project) the Cloud processing is used for parallelizing and optimizing the image processing algorithms. In the case of the second project (the SPERO project), satellite, aerial data, in-situ data and terrestrial data is stored, archived, analysed and processed in the Cloud since the Cloud offers the necessary amount of resources (as Infrastructure as a Service – basically hardware and virtualized Cloud resources, as Platform as a Service – for example Cloud based operating systems such as Microsoft Azure, Google Apps engine etc., and Software as a Service – for example mail applications, office applications, social networks applications, etc. – see Figure 1). In the case of the third discussed project (the ROBIN Cloud project) the data that is generated comes from intelligent robots and the Cloud-based processing of this data is done at two different architectural sub-levels:

- Edge processing level: in this situation the processing of data that doesn't require a lot of hardware or software resources and since it needs to be processed in real-time is done at the Edge of the Cloud.
- Cloud processing level: in this situation the processing is more complex and the required number of both hardware and software resources is greater than those offered by the Edge level.

2. Related work

The importance of the SLA in the case of the Cloud is discussed in various research papers with focus on modelling SLA parameters, automatic negotiation of the SLA based on the workload SLA, management from various Quality of Service perspectives and so on.

In (Alhamad et al., 2010) a model of SLA for Cloud computing and different SLA parameters for IaaS (see Table 1), PaaS and SaaS are presented. Some of these parameters are also mentioned and analysed in our work. In Table 1 we present these parameters, their usability and their associated measurement units.

Table 1. SLA parameters for IaaS, PaaS, SaaS (Alhamad et al., 2010)

Type of Cloud services	Parameter	Use related to the use cases	Measurement unit
IaaS	CPU capacity	It refers to the computational speed for hardware or VM resources	MIPS
	Memory size	It refers to the capacity of the memory offered to the Cloud service customer	GB, TB
	Boot time	It refers to how fast a computing node boots, and it can be used for its purpose	ms or s
	Storage	It refers to the storage size that is offered to the Cloud service customer	GB, TB, PB
	Scale up	It refers to the maximum number of computational resources offered to the Cloud service customer	Number
	Scale down	It refers to the minimum number of resources offered to the Cloud service customer	Number
	Scale up time	It refers to the necessary time to add and offer a greater number of resour-ces to the Cloud service customer	Time
	Scale down time	It refers to the necessary time to re-duce and offer a lower number of re-sources to the Cloud service customer	Time
	Auto scaling	It refers to whether an automatic increase or decrease in the number of resources is permitted	Yes/No
	Max number can be configured on physical server	It refers to the biggest number of computing resources that can be offered	Number
	Availability	It refers to the time (or percenttage of the total functioning) the service is available to the Cloud service customer	Time or Percentage
	Response time	It refers to the time necessary for the Cloud service to answer a certain request	Time
PaaS	Integration	It refers to what platforms and Cloud services can be integrated with	Yes/No (if it can be inte-grated or not with another platform)
	Scalability	If the service is scalable with an increasing number of users	Yes/No
	Pay as you go billing	If the Cloud computing resources offered by the provider are charged based on the number of resources or on the time the service is available	Yes/No
	Environments of deployment	If the Cloud service is offered online or offline	Yes/No
	Servers	If multiple servers (and their type) are offered by the Cloud service	Yes/No

	Browser	The types of browsers offered by the Cloud service (e.g. IExplorer, Firefox, Chrome, Opera)	The type of browser
	Number of developers	The entities (or Cloud users) that can access the Cloud service platform	Number
SaaS	Reliability	What is the level of confidence a user can have in the service	Number/Rating
	Usability	How easy can the Cloud service customer operate	Number/Rating
	Scalability	If the software offered by the service is scalable with an increasing number of users	Yes/No
	Availability	It refers to the time (or percentage of the total functioning) the service is available to the Cloud service customer	Time/percentage
	Customizability	If the Cloud service can be adapted to various Cloud service customers' requests	Yes/No

In (Son et al., 2013) it is discussed the possibility to negotiate automatically the SLA based on the workload and on the location of the Cloud service to facilitate the allocation of resources. This is important because based on the Cloud processing operation a SLA can be negotiated.

In (Wang et al., 2011) it is presented a platform for SLA with SLA templates that can be implemented between the Cloud service customers and the Cloud service providers. Additionally, a reputation system is analysed. The reputation system is based on the idea that many Cloud service providers and customers vote for the overall reputation score. This is important for the Cloud processing because the SLA negotiation is made easier. By computing a reputation, the confidence of the Cloud service customers in the chosen Cloud service providers can be increased.

In (Serrano et al., 2016) the SLA management from various Quality of Service perspectives (such as performance, availability, reliability and dependability and costs both financial and in energy) is studied and use cases are presented. The importance of this work is given by the attempt to solve three different challenges:

- the solution to generate a SLA in the case of different types of Cloud environments. In the case of Cloud processing the SLA can be generated depending on the use case.
- the description of the SLA in terms of a unified language called CSLA (Cloud Service Level Agreement language); in the case of Cloud processing the unified language offers the possibility to define the SLA easier for both the Cloud service customers and the Cloud service providers.
- use cases for each behaviour of the Cloud service are examined with the purpose of verifying the usefulness of the proposed SLA design for the QoS in Cloud.

2. SLA for the satellite image processing

The ForestMon project addresses the processing of satellite images (that monitor forest areas) with the goal of extracting useful information from them. This information can be represented by changes of the natural landscape that occur rapidly such as: forest fires, deforestations or other forest related disasters.

In this project, the processing of images that come from satellites is intended to be done in Cloud. The parallelization of the processing of the satellite images was done in the MPI language. The purpose of the processing of the satellite images is to obtain a geo-referential image that is of high quality and it is free of snow, clouds and shadows. All these aspects were presented in (Ilie et al., 2019), (Filip et al., 2019).

ForestMon pipeline processing platform for image processing is presented in Figure 2.

The main idea is that satellite images that need to be processed are divided into smaller parts such that a **parallelization** process on the smaller images takes place. The **parallelization** phase has as results several tasks that are added to the **Task Queue**. The **Scheduling engine** takes tasks from the **Task Queue** and sends them to execution to the **Runtime manager**. The results of the execution by the **Runtime manager** are sent to the **Users**. In the same time the **Runtime manager** sends status updates and information about the resource availability to the Scheduling engine to optimize the scheduling process.

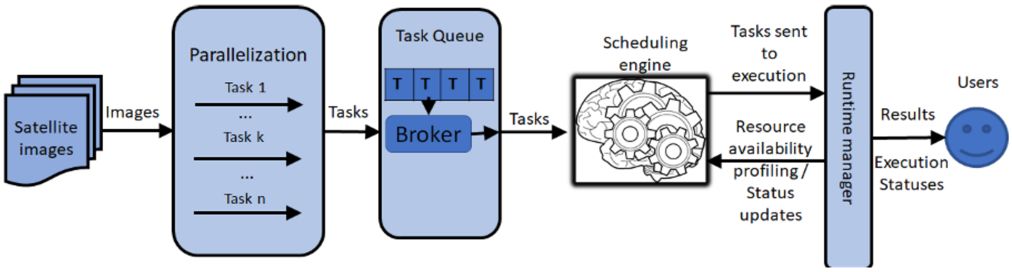

Figure 2. ForestMon platform for image processing in Cloud

From the time point of view the tasks that are components of the parallel program can arrive in real-time or can be queued to be executed in a non-time constraint manner. The case with queued tasks is discussed above and in (Iordache, Pop et al., 2017). In ForestMon the results of the satellite imagery can be used for prediction of disaster or for post disaster evaluations. Thus, it is very important to deal with time-critical tasks in an optimized manner and use the Cloud resources as efficient as possible.

For the parallel processing of images in the Cloud, using the ForestMon architecture, various SLA parameters (based on (Iordache, Paschke et al., 2017)) can be used, such as:

1. **CPU capacity.** For example, in (Ilie et al., 2019), 16 Intel Xeon 64-bit processors running at 2.20 GHz are made available to the Cloud service customer for parallel image processing. In (Filip et al., 2019), 16 x Intel(R) Xeon CPU E5-2695 v4 processors are made available to the Cloud customer.
2. **RAM capacity.** For example, in (Ilie et al., 2019) the RAM capacity is 30 GB of RAM, Level 1 cache of 32K, Level 2 cache of 256K, Level 3 cache of 56320K. In (Filip et al., 2019) the RAM capacity used for parallel image processing is 16GB DDR4 ECC.
3. **Hard Disk capacity.** In (Filip et al., 2019) the Hard disk capacity is of 1TB of RAID5 10k RPM SAS HDD.
4. **Total time of execution.** This parameter is useful when optimizing the execution time by parallelizing in MPI the serial code of the FMask algorithm (the algorithm needed to build the snow, cloud and shadow masks) (Ilie et al., 2019), (Filip et al., 2019). In the case of tasks that come in real-time, some deadlines must be satisfied for the parallel image processing to occur correctly. If the processing of images is a critical Cloud operation (meaning that the process must be done in real-time) it is necessary to have a negotiated number of resources. In the case of different hardware capacities (such as CPU capacity, RAM capacity, Hard Disk capacity) if the minimum number of resources cannot be offered (for example instead of 16 CPUs only 10 can be offered by the Cloud service provider) a penalty must be paid. This penalty should be paid by the Cloud service provider because the total time of execution increases. The penalty can be expressed as a function that depends on the number of resources offered. This function is the following:

$$Penalty(res,c) = \begin{cases} 0 & \text{if } capacity(res) \geq c \\ \alpha + (1-\alpha)\left(\dfrac{c}{capacity(res)}\right)^{p} & \text{otherwise, with } 0 \leq \alpha \leq 1 \end{cases}$$

A graph with different values for α, the ratio $\frac{c}{capacity(res)}$, and with the values 1, 2, 3, 4 for the p power can be seen in Figure 3. The table for the different values of α, of the ratio $\frac{c}{capacity(res)}$, and for *p = 1*, *p = 2*, *p = 3*, p = 4 for the *p* power can be seen in Table 2.

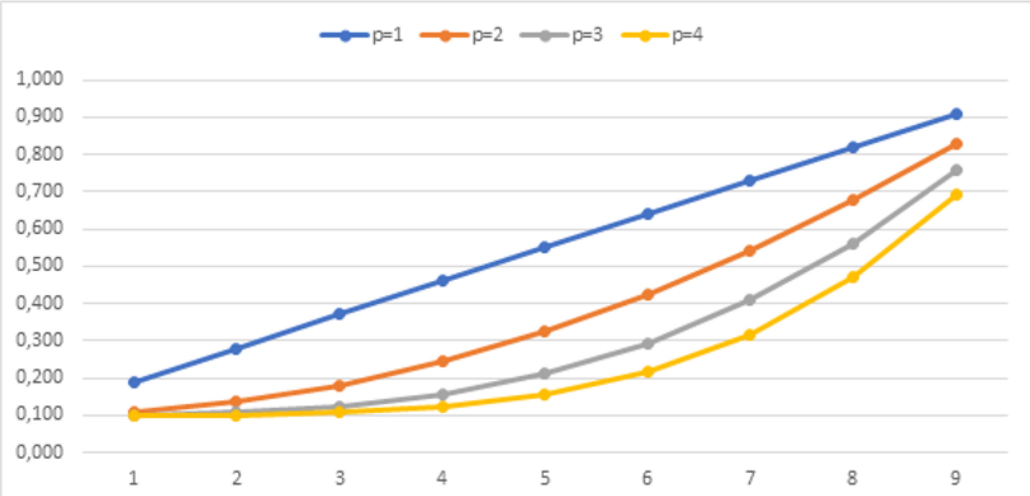

Figure 3. A graph for the penalty function in Cloud

Table 2. The data for the penalty function Cloud for the ForestMon project

$\frac{c}{capacity(res)}$	α	$Penalty(res,c)$ in the case of *p = 1*	$Penalty(res,c)$ in the case of *p = 2*	$Penalty(res,c)$ in the case of *p = 3*	$Penalty(res,c)$ in the case of *p = 4*
0,9	0,1	0,190	0,109	0,101	0,100
0,8	0,2	0,280	0,136	0,107	0,101
0,7	0,3	0,370	0,181	0,124	0,107
0,6	0,4	0,460	0,244	0,158	0,123
0,5	0,5	0,550	0,325	0,213	0,156
0,4	0,6	0,640	0,424	0,294	0,217
0,3	0,7	0,730	0,541	0,409	0,316
0,2	0,8	0,820	0,676	0,561	0,469
0,1	0,9	0,910	0,829	0,756	0,690

From the graph we can see that if we want to penalize the Cloud service provider severely (for example in the case when we need in real time a certain CPU capacity) then we can choose a p power of 1. In the case if a certain number of resources is not necessary in real-time then the penalty should be less severe (for example in the case p = 4).

5. **Accessibility in case of problems.** In the ForestMon case the users don't have access to the Cloud infrastructure.

3. SLA for the disaster management platforms

The SPERO (http://spero.ici.ro/about-spero/) project has the purpose to solve the emergency situations caused by natural disasters such as: fires, floods, storms, volcanic eruptions, earthquakes, tornadoes and other geologic processes, different crisis such as industrial accidents, and humanitarian crisis such as search and rescue situations.

In this project the goal is to develop an integrated system (an architecture that has both hardware and software components) for the management of data and information related with crisis and disasters, as support of their monitoring and for decision making in response to them. Additionally, because the important events (crises and disasters) need to be prevented (if it is possible), analysed and measures need to be taken to deal with these situations, data from different sources (such as satellite data, in-situ, aerial and terrestrial photos) will be integrated. The usual operations on this data will be archiving, searching through, viewing and processing it.

An architecture of the SPERO infrastructure is presented in Figure 4:

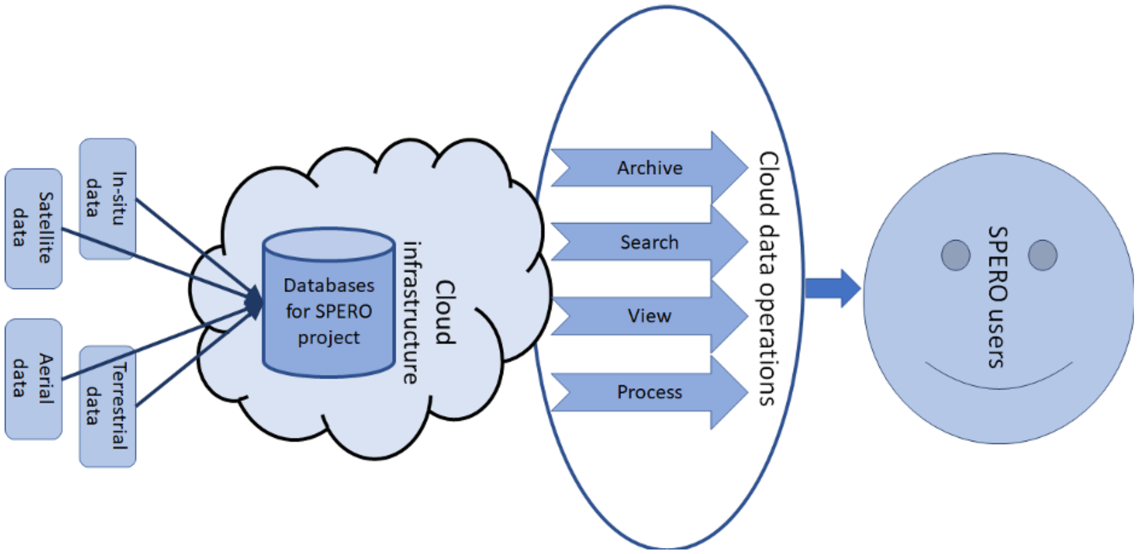

Figure 4. Cloud architecture for the SPERO project

The SLA related characteristics for the SPERO project are described next. The data and information (for example metadata, analysed data etc.) from different data sources is collected and saved in a common Cloud based database. Thus, the SLA parameters that need to be defined by the Cloud service provider are hardware related ones such as **Hard disk capacity**, **CPU capacity**, **RAM capacity** of the Virtual Machine(s) that holds the database. The database offered by the Cloud service provider should be used at a minimum price (we refer to the SLA **Resource prices/COST** parameter negotiated between the customer and the provider of the Cloud service). In addition, the Cloud service needs to be functional during a certain **period of operation**. This SLA parameter it is negotiated based on the needs of the Cloud service customer.

The data (and the information associated with it) is also processed to visualize important features, search through it, classify it and analyse it. In this case the SLA parameters that are defined are related to the correct functioning of the processing Cloud based infrastructure. If the data needs to be analysed in parallel, then the SLA parameter **number and types of VMs** needs to be defined. If the processing of the data needs to be in near-real time, then it is important to negotiate SLA parameters such as: **Service availability** (periods when the service is functional – usually when we know that new data arrives in the system), **Percentage of Disponible Resources** (when data arrives in the Cloud system and needs to be processed a certain percent of resources needs to be available) and the **Total Time of Execution** (needs to be reduced by using optimal processing algorithms, and even parallelize them).

Different types of tools for monitoring the data, alerting the users of the SPERO architecture and drawing conclusions after a major incident need to be developed. The SLA parameters in this case are related to the correct functioning of the Cloud based applications: the **Total Time of Execution** of each application needs to be minimized, **Number and types of VMs** (that are necessary for running the tools), the minimum number of resources that need to be available must be specified – (the **Percentage of Disponible Resources** SLA parameter).

In SPERO the access to information is done in a multi-tenant fashion. We can define three types of entities:

1. The first entity is given by the data collection system. This system needs to save the data in the Cloud database in real-time thus we can say that the importance of this entity is 1 (the most important).

2. The second entity as importance (2 as importance) is the entity that processes the data. The importance of this entity is given by the fact that once we have the data, we can do processing on it.
3. The third type of entities are the tools that monitor, analyse and alert the users of these tools. The tools work on the processed data, thus they are the least important.

In the case of Cloud multi-tenancy, various SLA parameters (see (Iordache, 2019)) need to be defined such as:

- **Percentage of Available Resources Allocated** to a **Tenant k** (in the case the importance of the tenant is greater, then the resources need to be allocated by the Cloud service provider to satisfy the tenants in the order of their importance: this means that if new data arrive in the SPERO system the most important is to save it in the Cloud databases; next as importance is the processing of the data and the least important the analysis of this data);
- **Percentage of Correctly Executed Tasks** on a certain resource $\mathbf{R_k}$. In the case of multi-tenancy in SPERO, it is possible that the access to a resource is concurrent, thus it is very important that the tasks that come to the SPERO Cloud system are executed correctly. If some tasks are executed incorrectly, usually the Cloud service provider needs to pay some penalties;
- **Percentage of Availability** of the Cloud service for a **Tenant k**. If it is the case of concurrent access to the Cloud service of multiple tenants, then each tenant should use the service for a given period and afterwards the access to the Cloud service is given to another tenant and so on.

4. SLA in the IoT systems

The ROBIN-Cloud project goal is to interconnect intelligent robot systems with Cloud infrastructures with the purpose of designing and implementing complex systems. These systems can be used by their customers to solve different practical situations to make their life easier (for example in the case of IoT systems (Alexandru et al. 2018), (Neagu et al., 2017), (Savu et al., 2017), (Stojanovic et al., 2015). Additionally, the Cloud can be used by the robots to save, process and analyse data that comes from the intelligent robots. The reason behind the utilization of the Cloud is the fact that the robots can generate a lot of data (for example from video camera or from different sensors) that needs to be transferred to the Cloud environment because the robots have reduced storage and processing capabilities (Filip et al., 2018).

The ROBIN-Cloud project is motivated by the increasing use of robots in our society and by the increasing size of data (called in some cases Big Data (Florian et al., 2016), (Alexandru et al. 2018), (El-Shafeiy et al., 2017)) generated by the robot systems. Cloud Robotics is based on the idea that the complex and big size data that comes from the robots can be stored in the Cloud. The stored data can be processed, analysed and used in different other operations (such as predictions, learning, knowledge retrieving) can be computed, by using complex algorithms (for example Machine Learning algorithms). The use of Cloud is motivated by the extended interest of different Cloud users (for example business companies) in utilizing the data collected by the Cloud infrastructure. The architecture of the ROBIN-Cloud project considers the following requirements (see Figure 5):

- part of the processing (the one that reduces network latency, necessitates real-time processing capabilities and require a considerable amount of processing power) is made at the Edge of the Cloud;
- and the rest of the processing (the most complex one) is done at the Cloud Level. This level of processing is for batch tasks (the ones that take a great amount of time to be executed and/or that require many resources).

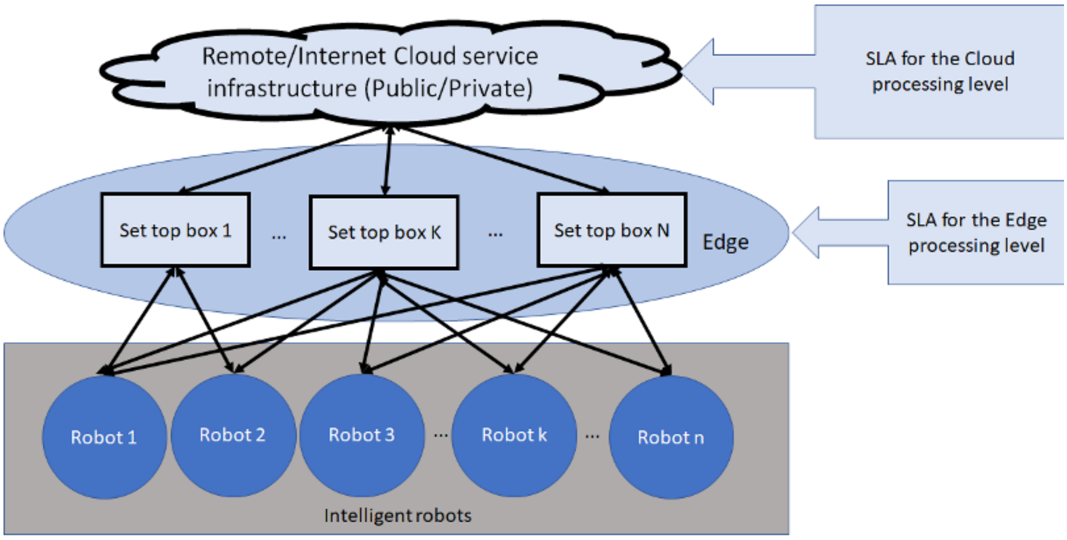

Figure 5. The architecture for the ROBIN Cloud project

In the ROBIN Cloud use case, the SLA parameters that should be taken into consideration must be defined both in the case of the Cloud processing level and in the case of the Edge processing level.

At Edge processing level the parameters that should be considered are presented Table 3:

Table 3. SLA parameters that measure the performance of an Edge computing infrastructure

No.	Name	Unit
1.	Total Execution Time	ms
2.	Total Execution Time for a tenant	ms
3.	Percentage of Disponible Resources	percentage
4.	Service availability	Time
5.	Maximum downtime	Hours
6.	Edge computing infrastructure failure rate	Number
7.	Periods of operation	Time
8.	Latency times	ms
9.	Accessibility in case of problems	Yes/No
10.	Number and types of nodes	Number and type
11.	Consumed energy	Watt

The Edge computing paradigm can be used to schedule DAGs of tasks that come from the Intelligent robots (Ţigănoaia et al., 2019). For example, a robot can send to the Edge computing infrastructure several images that can be divided for processing in smaller images. After the processing takes place the smaller images are combined into the original images' sizes (and sent for example to the Cloud processing level for more complicated processing operations). As discussed in (Iordache, 2019) the parameters that appear in the case of Edge processing level are:

1. The **Total Execution Time** for a certain DAG. Based on the limited processing power at the Edge level and on the real-time processing capabilities it is very important to optimize the **Total Execution Time** of a certain DAG. Additionally, if different robots send tasks to be processed at the Edge processing level by the same resource, we can discuss about the minimizing the **Total Execution Time for a tenant** because every task sent by a robot needs to be executed on the shared Edge infrastructure. The minimization of the **Total Execution Time** and of the **Total Execution Time for a tenant** can be achieved by choosing and implementing the appropriate processing Edge level algorithm;
2. The **Percentage of Disponible Resources** for a given tenant (every robot that sends data shares the same Edge infrastructure) refers to the fact that for the correct execution of the tasks

sent by the robots a certain number of resources from the total number of resources needs to be available to each robot;

3. The Edge computing **Service Availability** denotes the fact that the Edge infrastructure might not be available for certain periods of time because of several causes such as: very small network bandwidth, data loss, malfunctioning of computational issues, not enough storage or communication resources;
4. The **Maximum Downtime** can be expressed in the SLA contract and can be planned or unplanned. In the case of the unplanned downtime the Edge processing service needs to be restarted after the system (hardware and network) failures are fixed. The system failures are detected usually by monitoring the service;
5. The **Edge computing infrastructure failure rate** refers to the malfunctioning of Cloud resources and the failure rate is expressed as the number of failures in a period. This parameter needs to be minimized by proper administration of the Edge computing infrastructure;
6. The **Periods of operation** refers to when (as a time interval) the Edge computing service is up and running;
7. The **Latency times** refer to the time necessary for a bit of data to travel between the source robot that generates the data and a computing node of the Edge infrastructure. This parameter is important because by minimizing it the communication inside the architecture can be achieved;
8. The **Accessibility in case of problems** parameter is usually No because the users of the Edge/Cloud infrastructure should not modify it;
9. The **Number and types of nodes** parameter has a value given by the number of computing nodes and their types and it is agreed when designing the SLA contract based on the needs (such as processing requirements, number of robots) of the customers of the Edge computing infrastructure;
10. The **Consumed energy** is important from the point of view of the provider of the Edge infrastructure because the resources of this infrastructure have limited computing power thus energy in the case of these resources needs to be saved and by saving energy the overall costs of the Cloud service provider are reduced.

In the case of the Cloud computing level (the last level where the data can be stored and processed) the SLA parameters that need to be agreed upon depend on the infrastructure of the Cloud service. As use case we can think of the Cloud infrastructure as a Federated Multi-Cloud one. For example, the Cloud service provider will rent (from other Cloud service providers) Cloud infrastructures for data storage, for processing and analysing it, and for knowledge retrieval. Thus, the offloading of the data to the Cloud needs to be described in the SLA between the customers and the providers of the Cloud service. Some of the SLA parameters that need to be defined are (see Table 4):

Table 4. SLA parameters that measure the performance of the ROBIN-Cloud Cloud computing service

No.	Name	Unit
1.	Maximum discovery time	ms
2.	Total Execution Time	ms
3.	Percent of Correctly Executed Cloud Tasks on the Multi-Cloud Federation Architecture	percentage
4.	Percentage of Disponible Resources	percentage
5.	Maximum Time of Incorrect Functioning	time
6	Costs	currency

1. The **Maximum discovery time** SLA parameter defines the time that is necessary for the main provider of the Cloud service to offer for processing purpose its resources to the other Cloud service providers. This parameter is important because it is possible that the providers of the Cloud service are situated in different geographical regions thus the discovery between them takes time;

2. The **Total Execution Time** can be minimized by choosing one or different Cloud service providers that have the capabilities to execute the tasks in a minimum amount of time. For example, when talking about the ROBIN-Cloud use case, we can choose some Cloud service provider that processes the images (that come from the intelligent robots) in a minimum amount of time;
3. The **Percent of Correctly Executed Cloud Tasks on the Multi-Cloud Federation Architecture** defines the ratio between the number of tasks that can be executed correctly over the total number of tasks. In the case of ROBIN-Cloud project some Cloud providers may not be able to execute correctly some tasks because of different reasons (such as: invalid permissions to run some tasks, the scheduler of the Multi-Cloud Federation cannot schedule some tasks, to big time of execution on certain Multi-Cloud Federation entity, the number of tasks exceed the capacity of the Multi-Cloud Federation, partial or total stoppage of some Cloud service(s) etc.);
4. The **Percentage of Available Resources** is another important parameter of the Multi-Cloud Federation; this parameter depends on the number of resources that can be offered by each Cloud service provider that is part of the Multi-Cloud Federation. In the case of the ROBIN-Cloud project some resources cannot be available to some robots because of various reasons such as:
   - they are used for some processing and because the processing takes too long the resources cannot be available to other robots,
   - some resources are stopped or are not available to a certain robot,
   - concurrent access to the same resources of different robots that send data to the Multi-Cloud Federation;
5. The **Maximum Time of Incorrect Functioning** of the Multi-Cloud Federation can be minimized in the case of the ROBIN-Cloud project by monitoring continuously the Multi-Cloud Federation architecture and detecting incidents such as:
   - stoppage (for different reasons such as maintenance) or malfunctioning of a certain resource,
   - concurrent access to the same Cloud provider resources;
6. The **Costs** of designing a Multi-Cloud Federation infrastructure of the ROBIN-Cloud project are important because there is a trade-off between the number of resources and the processing capabilities of each resource. The Cloud service providers use the necessary processing capabilities of the intelligent robots to design the Multi-Cloud Federation architecture.

5. Conclusions and future work

In this work we present three different use cases (where the Cloud research paradigm is utilized) of SLA and its characteristics. These three use cases show that the SLA for the Cloud is important when dealing with real life situations. The presented parameters prove that reliable research in the area of SLA design for various Cloud processing use case can be done.

As future work we plan to implement the discussed SLA parameters in other real scenarios with the purpose of comparing the results with the situations already considered. It is important for the Cloud service provider and for the customer of the Cloud service to describe their interactions in different situations.

REFERENCES

1. Alexandru, A., & Coardoș, D. (2018). *Utilizarea Tehnologiilor Big Data şi IoT în Domeniul Sănătăţii*. Revista Română de Informatică și Automatică – RRIA (Romanian Journal of Information Technology and Automatic Control), 28, 61-84.

2. Alhamad, M., Dillon, T., & Chang, E. (2010, April). *Conceptual SLA framework for cloud computing*. In 4th IEEE International Conference on Digital Ecosystems and Technologies (pp. 606-610). IEEE.

3. Aljoumah, E., Al-Mousawi, F., Ahmad, I., Al-Shammri, M., & Al-Jady, Z. (2015). *SLA in cloud computing architectures: A comprehensive study*. Int. J. Grid Distrib. Comput, 8(5), 7-32.

4. El-Shafeiy, E. A., & El-Desouky, A. I. (2017). *A big data framework for mining sensor data using Hadoop*. Studies in Informatics and Control, 26(3), 365-376.

5. Filip, I. D., Ghita, B., Pop, F., Iordache, G. V., Negru, C., & Dobre, C. (2018, June*). EdgeMQ: Towards a Message Queuing Processing System for Cloud-Edge Computing: (Use Cases on Water and Forest Monitoring)*. In 2018 17th International Symposium on Parallel and Distributed Computing (ISPDC) (pp. 46-52). IEEE.

6. Filip, I. D., Negru, C., Pop, F., Stoica, A., & Serban, F. (2019, September). *Distributed processing platform for large datasets: satellite imagery usecase*. In 2019 IEEE 15th International Conference on Intelligent Computer Communication and Processing (ICCP) (pp. 63-70). IEEE.

7. Florian, V., & Neagu, G. (2016). *Abordări şi soluţii specifice în managementul, guvernanţa şi analiza datelor de mari dimensiuni (Big Data)*. Revista Română de Informatică şi Automatică – RRIA (Romanian Journal of Information Technology and Automatic Control), 26(1), 5-22.

8. Ilie, A. T., Filip, I. D., Postoaca, A. V., Negru, C., Pop, F., Stoica, A., & Serban, F. (2019, May). *Faster and scalable parallel processing solution to remove visual obstacles from satellite imagery*. In 2019 22nd International Conference on Control Systems and Computer Science (CSCS) (pp. 194-201). IEEE.

9. Iordache, G. (2019, May). *An Analysis of Service Level Agreement Parameters and Scheduling in Multi-Tenant Cloud Systems*. In 2019 22nd International Conference on Control Systems and Computer Science (CSCS) (pp. 140-145). IEEE.

10. Iordache, G., Paschke, A., Mocanu, M., & Negru, C. (2017). *Service level agreement characteristics of monitoring wireless sensor networks for water resource management (slas4water)*. Studies in Informatics and Control, 26(4), 379-386.

11. Iordache, G. V., Pop, F., Esposito, C., & Castiglione, A. (2017, May). *Selection-based scheduling algorithms under service level agreement constraints*. In 2017 21st International Conference on Control Systems and Computer Science (CSCS) (pp. 134-140). IEEE.

12. Neagu, G., Vrejoiu, M. H., Preda, S. A., Stanciu, A. (2017). *IoT platforms - Current solutions and evolution trends*. Revista Română de Informatică şi Automatică – RRIA (Romanian Journal of Information Technology and Automatic Control), 27(3), 5-18.

13. Savu, D., Tomescu, M., & Băjenaru, L. (2017). *Internetul lucrurilor – o nouă paradigmă a conectării în Internet*. Revista Română de Informatică şi Automatică – RRIA (Romanian Journal of Information Technology and Automatic Control), 27(1), 5-14.

14. Serrano, D., Bouchenak, S., Kouki, Y., de Oliveira Jr, F. A., Ledoux, T., Lejeune, J., & Sens, P. (2016). *SLA guarantees for cloud services*. Future Generation Computer Systems, 54, 233-246.

15. Son, S., Jung, G., & Jun, S. C. (2013). *An SLA-based cloud computing that facilitates resource allocation in the distributed data centers of a cloud provider*. The Journal of Supercomputing, 64(2), 606-637.

16. Stojanovic, N., & Stojanovic, D. (2015). *A hybrid MPI+ OpenMP application for processing big trajectory data.* Studies in Informatics and Control, ISSN 1220-1766, 24(2).

17. Ţigănoaia, B., Iordache, G., Negru, C., & Pop, Fl. (2019). *Scheduling in CloudSim of Interdependent Tasks for SLA Design*. Studies in Informatics and Control, 28(4), 477-484.

18. Wang, M., Wu, X., Zhang, W., Ding, F., Zhou, J., & Pei, G. (2011, December). *A conceptual platform of SLA in cloud computing*. In 2011 IEEE Ninth International Conference on Dependable, Autonomic and Secure Computing (pp. 1131-1135). IEEE.

George-Valentin IORDACHE a absolvit în 2006 studiile de licență în știința calculatoarelor și tehnologia informației în cadrul Universității Politehnica din București, Departamentul de Calculatoare, România, apoi a finalizat în 2008 studiile de master în știința calculatoarelor în cadrul Universității Stony Brook, New York, SUA. În 2020, George-Valentin Iordache a primit titlul de doctor în știința calculatoarelor de la Universitatea Politehnica din București, România. În timpul doctoratului, el a investigat proiectarea contractelor cu diverse constrângeri la nivel de serviciu în sistemele Cloud sub coordonarea Prof.dr.ing. Florin Pop. În prezent lucrează la BEIA Consulting International, o companie română de cercetare și dezvoltare.

George-Valentin IORDACHE received the B.Sc. in computer science from the Computer Science Department of the University Politehnica of Bucharest, Bucharest, Romania in 2006 and M.Sc. degree in computer science from the Stony Brook University, New York, USA, in 2008. In 2020 George-Valentin Iordache received his PhD degree in Computer Science from University Politehnica of Bucharest, Bucharest, Romania. During his PhD, he investigated the design of Service Level Agreement contracts in Cloud under different constraints. His advisor was Prof.dr.ing. Florin Pop. Now he works at BEIA Consulting International, a Romanian R&D company.

Florin POP este profesor în cadrul Departamentului de Calculatoare și Tehnologia Informației la Universitatea Politehnica din București. De asemenea, lucrează ca cercetător științific gradul I la Institutul Național de Cercetare-Dezvoltare în Informatică – ICI București. Interesele sale generale de cercetare sunt: sisteme distribuite pe scară largă (proiectare și performanță), grid computing și cloud computing, sisteme peer-to-peer, gestionarea Big Data, agregarea datelor, tehnici de regăsire și clasificare a informațiilor, metode de optimizare Bio-Inspired.

Florin POP is professor at the Department of Computer and Information Technology, the Politehnica University of Bucharest. He also works as a 1st degree scientific researcher at National Institute for Research and Development in Informatics – ICI Bucharest. His general research interests are: distributed systems (design and performance), grid computing and cloud computing, peer-to-peer systems, Big Data management, data aggregation, information retrieval and classification techniques, Bio-inspired optimization methods.

Cătălin NEGRU este inginer de sistem și cercetător în cadrul Departamentului de Calculatoare al Facultății de Automatică și Calculatoare și membru activ al Laboratorului de Sisteme Distribuite al Universității Politehnica din București. A obținut doctoratul în 2016, interesele sale de cercetare includ: sisteme distribuite, eficiența energetică, stocare în cloud, sisteme cyber-fizice, GIS. Cercetările sale au dus la publicarea a numeroase lucrări și articole la reviste științifice importante, precum Future Generation Computer Systems, Soft Computing, International Journal of Applied Mathematics and Computer Science și conferințe (ICCP, AQTR, HPCC, CISIS, CSCS etc.). Este implicat în mai multe proiecte de cercetare naționale și internaționale.

Cătălin NEGRU is a system engineer and researcher at the Computer Science Department of the Faculty of Automatic Control and Computers and an active member of Distributed System Laboratory at University Politehnica of Bucharest. He obtained his PhD from the same faculty in 2016. His research interests include distributed systems, energy efficiency, cloud storage, cyber-physical systems, GIS. His research has led to the publishing of numerous papers and articles at important scientific journals, such as Future Generation Computer Systems, Soft Computing, International Journal of Applied Mathematics and Computer Science and conferences (ICCP, AQTR, HPCC, CISIS, CSCS etc.). He is involved in several national and international research projects.

Gabriel NEAGU este cercetător științific gradul I și Director Tehnic în cadrul Institutului Național de Cercetare-Dezvoltare în Informatică – ICI București. A obținut titlul de doctor în Informatică Aplicată la Universitatea Politehnica din București, în anul 1998. Are o experiență extinsă în managementul activității de cercetare și al proiectelor CDI, precum și în colaborarea științifică internațională. În prezent, principalele domenii de interes pentru activitatea de cercetare includ: arhitecturi de sisteme distribuite, analiza avansată a datelor masive, servicii Cloud-IoT, știința deschisă și managementul datelor de cercetare.

Gabriel NEAGU is senior researcher 1st degree and Scientific Director at National Institute for Research and Development in Informatics – ICI Bucharest. He received a PhD in Applied Informatics at the Politehnica University of Bucharest, in 1998. He has extensive experience in the management of research activity and RDI projects, as well as in international scientific collaboration. Currently, his main topics of interest for research activity include: distributed system architectures, advanced data analytics, Cloud-IoT services, open science and research data management.