Cloud GNSS receivers: New advanced applications made possible

Vicente Lucas-Sabola[1], Gonzalo Seco-Granados[1], José A. López-Salcedo[1]
José A. García-Molina[2,3], Massimo Crisci[2]

[1]Universitat Autònoma de Barcelona (UAB), Bellaterra, Spain
[2]European Space Agency (ESA), Noordwijk, The Netherlands
[3]HE Space, The Netherlands

Email: vicente.lucas@e-campus.uab.cat, {gonzalo.seco, jose.salcedo}@uab.es
{jose.antonio.garcia.molina, massimo.crisci}@esa.int

***Abstract*—The widespread deployment of GNSS (Global Navigation Satellite Systems) is pushing the current receiver technology to its limits due to the stringent demands for providing seamless, ubiquitous and secure/reliable positioning. This fact is further aggravated by the advent of new applications where the miniaturized size, low power consumption and limited computational capabilities of user terminals pose serious concerns to the implementation of even the most basic GNSS signal processing tasks (e.g. as in Smart-City or IoT applications). This work presents a paradigm shift for the implementation of next-generation GNSS receivers by taking advantage of Cloud computing platforms, thus leading to the concept of *Cloud GNSS receiver*.**

I. Introduction

In the coming years, four different Global Navigation Satellite Systems (GNSS) will be fully operational, including the well-known U.S. system GPS, the Russian Glonass, the European Galileo and the Chinese Beidou. In total, they will provide more than 40 visible GNSS satellites at a time, anywhere on Earth. This is expected to solve many of the problems currently found when using GPS in urban environments, where hardly more than two satellites are visible at a time. The problem will be, though, the huge and overwhelming amount of data to be processed by the user receiver, taking into account that additional processing for augmentation satellite systems such as SBAS or EGNOS may be required (e.g. as in high-accuracy professional receivers) [1], or for authentication, security and integrity issues, such as protecting the position fix from being affected by interferences [2] and abnormal propagation effects [3].

On the one hand, all these processing tasks involve an unprecedented increase in the computational requirements of GNSS receivers by at least one order of magnitude, which is unfeasible with the current state of the art. On the other hand, user applications are gradually demanding low cost, small size and low power consumption devices, which dramatically hinder the implementation of complex processing tasks for seamless and ubiquitous positioning. These limitations are aggravated with the advent of the Internet of Things (IoT), Machine-to-Machine (M2M), and Smart City applications, where miniaturized devices with extremely low power consumption will play a prominent role [4]. This new plethora of devices will suffer from the limitation of not being able to implement even the most basic processing tasks of a GNSS receiver, thus requiring a paradigm shift in they way GNSS technology is delivered in future mass-market devices.

Taking into account the costly signal processing tasks carried out by GNSS receivers, Cloud computing becomes an exceptional opportunity for migrating all these tasks into a scalable, distributed and high-performance computing platform. The user terminal would only need to collect the RF samples and send them to the Cloud infrastructure where the GNSS signal processing would actually take place (see Fig. 1). The Cloud GNSS receiver paradigm therefore facilitates the implementation of sophisticated signal processing techniques without compromising the computational load and energy consumption of the user terminal. Upgrades for compatibility with new GNSS signals on the same band can easily be implemented, since they only need to be applied at the Cloud side. Furthermore, access to restricted services would be possible without the need of having any decryption key at the user side, since this task would take place at the secured Cloud facilities. Finally, the Cloud infrastructure becomes the perfect place to meet, share and compare data from other GNSS users, thus opening the door to GNSS big data and the exploitation of geo-spatial information for data science applications.

The motivation of the present work is to introduce the concept of Cloud GNSS signal processing, and to illustrate this concept through a case study based on the Cloud GNSS receiver proof-of-concept developed by the authors in collaboration with ESA. In Section II, we will focus on the use of Cloud computing platforms, and in particular, Amazon Web Services (AWS) for Cloud GNSS signal processing. In Section III we will review the ION SDR metadata standard, which promotes the interoperability between different data grabbers and GNSS receivers. In Section IV we will introduce the architecture

This work was partly supported by the Spanish Government under grant TEC2014-53656-R and by the European Space Agency (ESA) under contract No. 4000113891/15/NL/HK.

978-1-5090-1757-7/16/$31.00 ©

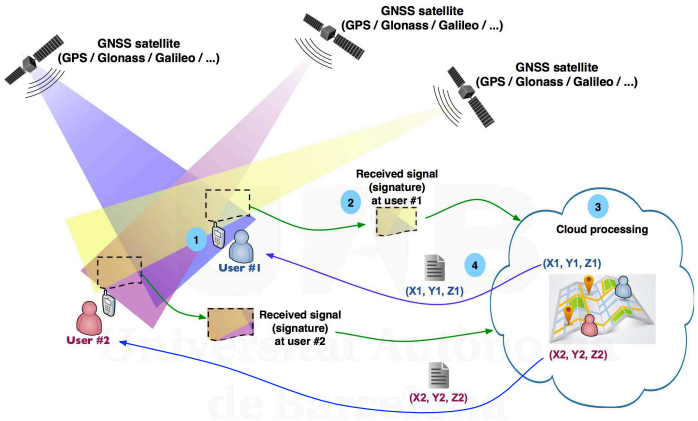

Fig. 1. Illustration of the Cloud GNSS receiver paradigm.

of a Cloud GNSS receiver developed in collaboration with ESA. Finally, potential applications of Cloud GNSS signal processing will be discussed in Section V, and conclusions will be drawn in Section VI

II. Cloud computing infrastructure

The concept of *Cloud computing* is based on offering computing services through the Internet, acting as a real computer system. Users can readily use the services provided by the Cloud without worrying about how the system is working or how many resources are being used, since the latter are virtually unlimited. Scalability, monitoring, security, liability, and cost among others, are some of the main characteristics of Cloud computing. From a service point of view there are three different approaches: Software as a Service (SaaS), Platform as a Service (PaaS) and Infrastructure as a Service (IaaS). The Cloud GNSS receiver is a SaaS, a remote application that the user can configure through a web browser and obtain some output results. While Amazon Web Services (AWS) was the first platform to be publicly launched, nowadays there are several major companies providing Cloud computing infrastructure such as Google Cloud Platform, Microsoft Azure, RedHat Openshift, or Oracle Cloud, just to mention a few.

In this work, AWS has been selected due the wide range of services, functionalities and configuration options that it provides, along with its openness and flexibility. However, all major Cloud platforms are continuously evolving and bridging the gap between them, so the final decision may not be straightforward in some cases. Particularly when the user is already familiar with some other technology rather than AWS (e.g. organizations relying heavily on Microsoft infrastructure).

A. AWS services for Cloud GNSS signal processing

We provide next a review of the main AWS services that are of interest for implementing a Cloud GNSS receiver. They range from the mere computing service, to secondary services required to store (temporary or permanently) and communicate within the GNSS receiver building blocks:

- *Amazon Elastic Compute Cloud (EC2)*, which provides re-sizable and scalable computing capacity in the Cloud, allowing the user to launch instances with a variety of operating systems, loading them with a custom application environment and running the desired application using as many instances as required. An *instance* is a virtual Machine (VM) that acts as a physical computing machine, and it becomes the basic resource element of EC2. AWS provides a wide catalog of different instances depending on the type of CPU (Central Processing Unit), memory, storage, and networking capacity. Depending on the required computing capacity, different types of instances can be used to obtain faster results. This is the case of GPU instances (Graphical Processing Units), which are very well suited for GNSS processing due to extensive use of FFT (Fast Fourier Transform) cores.
- *Amazon Simple Storage Service (S3)*, which provides secure, durable and highly-scalable object storage. S3 stores data as objects within resources called *buckets* where as many objects as desired can be stored, written, read and deleted..
- *Amazon Simple Queue Service (SQS)*, which is a fast, reliable, scalable, and fully managed message queuing service to communicate different modules and systems within the Cloud infrastructure.
- *Amazon Elastic Block Storage (EBS)*, which provides persistent block level storage volumes to be used by EC2 instances.

B. AWS instances for Cloud GNSS signal processing

Amazon EC2 provides a large ecosystem of instances with multiple combinations of CPU, memory, storage, and networking capacity. This wide range of alternatives allows us to launch specific instances depending on the type of GNSS signal processing to be carried out, whether it requires intensive computation (as when searching for many satellites) or heavy memory storage (as when implementing long integration times for GNSS weak signal acquisition).

At high level, there are two main types of AWS instances, namely *burstable* and *fixed* performance instances. The former provide a baseline performance equivalent to a fraction of a CPU and they are controlled by a continuous flow of so-called *credits*, which can be exchanged for CPU performance. When the received credits are not needed, they are stored in a credit balance for up to 24 hours. If at any particular moment the workload requires more than the fraction of CPU we are assigned, credits are drawn from the credit balance to temporarily increase the CPU performance in a seamless manner. Burstable instances (i.e. `T2` instances) are recommended for applications that do no not require a consistently high CPU performance, such as the case of webservers or small databases. In contrast, fixed performance instances (e.g. `M3`, `C3` and `R3` instances) are better suited for applications that require an intensive and consistent CPU performance as well as managing high volumes of data.

We provide next a summary of the main types of instances that are of interest from a Cloud GNSS signal processing point of view:

- `T2` (burstable) instances are based on Intel Xeon processors with turbo up to 3.3 GHz. They are low cost general purpose instances that are well suited for running the user interface of the Cloud GNSS receiver (e.g. landing page of the service, user management, etc.). There are different models of `T2` instances depending on the flow of CPU credits per hour and the RAM memory, from the `t2.nano` with 3 credits/hour, 0.5 GB and 5% of baseline CPU performance, to the `t2.large` with 36 credits/hour, 8 GB and 60% baseline CPU performance.
- `C3` (compute-optimized) instances are based on Intel Xeon E5-2680 v2 processors. Different models of C3 instances offer RAM memories from 3.75 to 60 GB, 2−32 CPU cores and 32−640 GB of SSD storage. The latter is of particular interest for GNSS signal processing in order to speed up the reading of large volumes of data from GNSS raw samples files.
- `R3` (memory-optimized) instances are based on Intel Xeon E5-2680 v2 processors, and are designed for running memory-intensive applications, such as high-sensitivity GNSS techniques, thanks to their high RAM capacity (from 15−244 GB). These instances are also based on SSD storage, which facilitates reading large volumes of GNSS raw samples.
- `G2` (graphics-optimized) instances provide graphics processing units (GPUs) along with high CPU and network performance. GPUs are specialized in performing FFTs (Fast Fourier Transform), which is a common operation in the field of image processing. FFTs are implemented in an efficient and optimized manner in GPUs, using thousands of dedicated computing cores. This feature is of interest for GNSS signal processing, where FFTs are widely used in many software-defined GNSS receivers for efficiently implementing the despreading of the GNSS signal by correlating with the local code replica. Therefore, GPUs are a time- and cost- effective solution for executions where an intensive use of correlations is required, as it is the case of long integration times in high-sensitivity GNSS techniques. Furthermore, the SSD storage of these instances facilitates the data transfer and reading of GNSS raw samples.

III. ION GNSS SDR METADATA STANDARD

In the recent years, there has been a significant increase in the use of software defined radio (SDR) for GNSS data collection and post-processing. The heterogeneity of different devices, settings, and data acquisition protocolos makes very difficult to share and exchange GNSS raw sample files, since a proprietary or custom data loading is often required. The use of the ION SDR metadata standard [5] overcomes this problem by standardizing all metadata associated with the capture of GNSS raw samples through SDR devices (e.g. RF and IF center frequencies, sampling rate, quantization encoding, etc.). This standard is the result of a working group created in 2014 under the auspices of the Institute of Navigation (ION) with the aim of facilitating the interoperability between current and future SDR data grabbers and GNSS receivers [6].

The metadata format is based on Extensible Markup Language (XML) specified according to the XML Schema Definition (XSD) standard. Metadata are defined in terms of 12 core classes: `session`, `system`, `cluster`, `source`, `band`, `stream`, `lump`, `chunk`, `block`, `lane`, `file`, `and fileset`, as illustrated in Fig. 2. The attributes of each of these classes describes how samples have been encoded by the data grabber, and they are used to unambiguously read and decode the GNSS samples stored in a raw samples file.

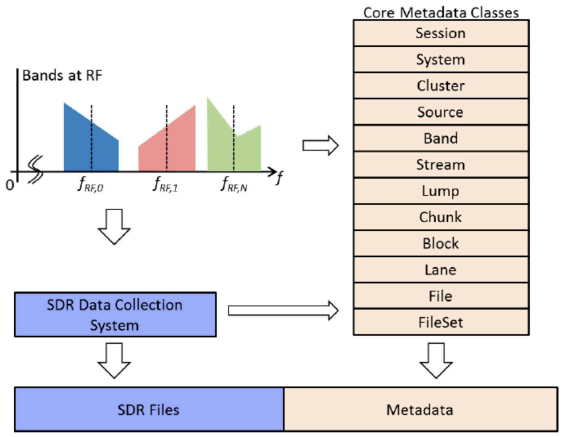

Fig. 2. Summary of the core ION SDR metadata classes [5].

The use of the ION SDR metadata standard is expected to become a key element for the widespread deployment of Cloud GNSS services, where compatibility for a wide range of different GNSS data grabbers must be ensured. Some steps have already been done for the widespread adoption of this standard, such as the development of a standard-compliant file reader and decoder to retrieve GNSS samples from a SDR data file, as illustrated in Fig. 3 [6].

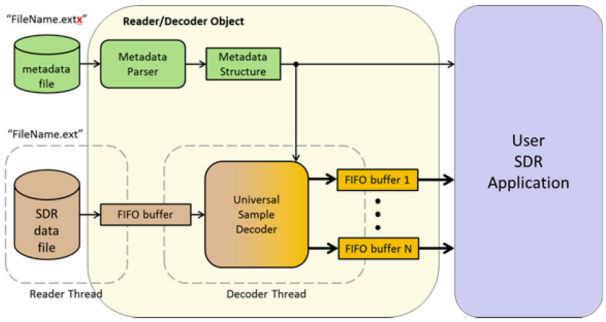

Fig. 3. Block diagram of ION SDR metadata standard-compliant file reader/decoder [6].

IV. CLOUD GNSS RECEIVER ARCHITECTURE

The elements described so far provide the basic building blocks for developing a Cloud GNSS receiver. In this section, we will move one step further by describing the architecture of

an experimental Cloud GNSS receiver developed in collaboration with ESA, which has been successfully tested with real GNSS data and different types of workloads from multiple users. The core of the Cloud GNSS receiver is based on the high-sensitivity (HS) GNSS software receiver developed in the framework of the ESA funded project DINGPOS [7], which is a snapshot-based receiver providing a C/N0 sensitivity down to 15 dBHz and compatible with GPS L1/L5, Galileo E1C/E5a. The core HS-GNSS software receiver is based on the extensive use of FFT processors and implements long integration times up to several seconds using advanced noncoherent integrations. It also provides signal-quality monitoring features such as C/N0 monitoring, near-far detection or interference mitigation [8].

From a high-level perspective, the developed Cloud GNSS receiver is composed of three main elements: the cloud user terminal, the front-end module in charge of interacting with the user, and the back-end module where the HS-GNSS software receiver is actually running, and where all the computational tasks, reporting and delivery of results are carried out. We will describe next the details of these blocks, whose overall interconnection is illustrated in the block diagram of Fig. 4.

A. Cloud user terminal

The Cloud user terminal is the hardware element in charge of gathering the GNSS samples at the user side, and sending them to the Cloud GNSS receiver for the subsequent processing. The terminal is composed of a RF front-end tuned at the GNSS bands of interest, a data grabber device for digitizing the GNSS signals, and a communication module for interacting with the Cloud GNSS receiver. In the tests carried out within the scope of this work, a USRP N200 board was used as the cloud user terminal, and the exchange of data with the Cloud was carried out through the embedded Ethernet port. Once the samples are transferred to the Cloud and the processing has been carried out, a report with PVT and signal-level analysis is generated and sent to the email indicated by the user.

B. Cloud front-end

The entrance to the Cloud is done through an HTTP webservice running on a dedicated EC2 machine, where users can log in and enter into a private desktop. From there, new executions can be launched using an online graphical user interface. This interface allows the user to configure the HS-GNSS software receiver running at the Cloud by filling the information of three basic steps:

1) *GNSS raw samples.* The file with the GNSS raw samples must be uploaded to the Cloud, either using a file locally available at the user's device, or introducing an URL (e.g. Dropbox, FTP) where the Cloud can automatically download the file. At the same time, the user must specify the format of this file with the sampling frequency, central frequency, intermediate frequency, quantization encoding, etc. The user has also the option of using the ION SDR GNSS metadata standard developed for indicating the file format.
2) *HS-GNSS settings.* The user must configure the HS-GNSS according to the specific needs of the analysis to be done, and to the working conditions where the samples were gathered. The parameters to be provided here include the GNSS band to be processed, the integration time, number of snapshots, etc. The flexibility offered by the Cloud GNSS Receiver lets the user choose some advanced features such as using long integration times for processing extremely weak GNSS signals, detecting near-far in soft-indoor/urban scenarios, and the option of computing the user's PVT or just get some signal-level analysis without PVT. The latter option is of interest when the user is interested in exploring the gathered GNSS signals, rather than in obtaining a position fix. This is the case for instance of scientific applications where the information is conveyed on the output observables or in the shape of the autocorrelation peaks. This information is delivered to the user in the generated output report.
3) *Assistance information.* For instance, information regarding the list of satellites to be searched, so that we can avoid the full search thus speeding up the acquisition stage of the HS-GNSS receiver. If we also have available information on the coarse Doppler of the visible satellites, we can introduce this information at this point. Finally, if PVT is to be computed, a RINEX (Receiver Independent Exchange Format) file must be attached either by uploading it from the user's device, or asking the Cloud receiver to download it from a list of available GDC servers.

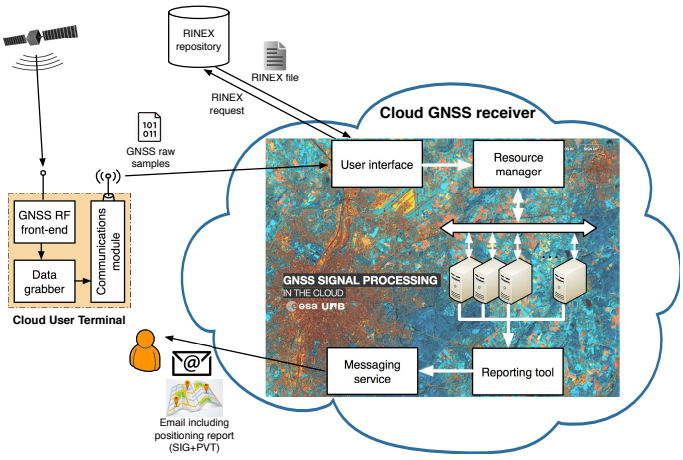

Fig. 4. Architecture of the developed Cloud GNSS receiver.

C. Cloud back-end

The back-end of the Cloud GNSS receiver is where the bulk of the processing tasks is carried out. This includes processing the GNSS raw samples but also generating the output reports to be delivered to the user. Each of the new jobs generated by the user interface are received and managed by a resource manager in charge of allocating EC2 resources to the current

workload (see Fig. 4). When a new instance is launched, a pre-configured image saved at the EBS service is booted with all the necessary software for the processing of the GNSS raw samples file. Thus, new instances are ready to work just after being launched with a very small latency. The HS-GNSS software receiver is implemented as a standalone application by means of the MATLAB Compiler Runtime (MCR). This is a free standalone set of shared libraries that enables the execution of compiled MATLAB applications or components on computers that do not have MATLAB installed. Hence, there is no need to install the MATLAB software in each running instances, which is a hard drive space and cost-effective solution. There also are other preconfigured software such as TEX Live, PyLaTeX, PyPDF2, among others, which are used by report generation tool to produce the report in PDF format with the obtained results. Finally, the Gmail service is used to deliver the report to the user's email address.

V. Applications of Cloud GNSS signal processing

In the previous sections we have introduced the technology and the general architecture of a Cloud GNSS receiver. In this section, we will focus on the potential applications of Cloud GNSS signal processing with the aim of sparking the interest on this topic within the GNSS community. It is interesting to note that beyond conventional GNSS applications, such as location-based services (LBS), surveying or fleet management, just to mention a few, the Cloud paradigm opens up the door for innovative applications whose specific characteristics (e.g. massive processing of data, cooperation among users, etc.) make them suitable for implementation using a cloud-based infrastructure. A representative list of such innovative applications is listed below.

A. Secure/authenticated GNSS positioning

In the recent years, some concerns have been raised on the security of civilian GNSS signals in front of potential threats such as interference or counterfeit GNSS signals [9]. These attacks may lead to a misbehavior of the GNSS receiver, thus endangering GNSS-based safety-critical applications. While significant efforts have been devoted so far to detect these threats [10], [11], sophisticated attacks require advanced detection and mitigation techniques. It is now commonly agreed that the most effective solution to circumvent these threats is by introducing authentication and signal encryption on the next generation GNSS signals [12], [13]. The target applications are mainly related to the commercial services, safety-of-life applications and the provision of a secure signals for government authorities.

The latter is a clear example on the potential use of Cloud GNSS receivers, since the authentication and decryption tasks can safely be implemented on the Cloud infrastructure. This makes unnecessary to distribute keys to authorized users, thus avoiding the risk of third (unauthorized) users intercepting the keys. This approach was recently presented in a proof-of-concept of a server-based secure receiver [14], which confirms the interest and feasibility of outsourcing the authentication and decryption of GNSS signals to the Cloud.

B. Crowdsourcing GNSS signal processing

A very appealing feature of Cloud GNSS signal processing is that we can collect and process batches of GNSS samples that have been remotely collected in geographically different places, and then apply data analysis tools to extract information from this collectivity of samples with a common goal. This so-called *crowdsourcing* approach (i.e. several measurements are treated together with a common goal) has a myriad of different applications such as:

- *Integrity monitoring*, where a set of sensors is in charge of gathering GNSS samples in a given service area of interest, and then the samples are delivered to a central control unit for computing key performance indicators (e.g. signal quality, accuracy, integrity, etc.). This would be the case of some proprietary GNSS monitoring stations for safety-critical applications such as civil aviation, or the ranging integrity monitoring stations (RIMS) of the EGNOS ground segment.
- *Interference detection*, where samples gathered at different geographical locations can be correlated at the Cloud with the aim of detecting and locating interference sources, applying interferometric techniques.
- *Ionospheric monitoring*, where a monitoring network collects and gathers data for the characterization of ionospheric effects on the performance of GNSS and augmentation systems, focusing on extreme events and ionospheric scintillations [15].

C. Smart City-related applications

Nowadays, one out of two people live in urban areas, a proportion that is forecasted to increase up to two out of three people by 2050. This means that existing cities will face unprecedented challenges to absorb such a significant amount of new citizens, thus forcing a paradigm change in the way cities are currently known. The *smart city* concept addresses this problem by putting intelligence on the urban management, from transportation to energy, water, waste, and any other asset within the city infrastructure. The goal is to contribute making cities more sustainable place and to manage them in a more effective, efficient and social manner.

In this context, GNSS is expected to play a key role in mobility-based applications such as autonomous driving, intelligent transportation, automatic road-tolling, fleet management or parking management, just to mention a few. In all these cases, the moving vehicle can incorporate a sensor to gather GNSS samples (i.e. the Cloud user terminal shown in Fig. 4) and then send them to the Cloud GNSS Receiver to determine or just report its position. Since all the computational burden is moved to the Cloud, the user terminal can simply be composed of a front-end and a small wireless module to transmit the GNSS samples. This leads to a very simple, low cost and small device that can easily be integrated in a myriad of objects, from vehicles, bicycles, wearables, etc. This has a strong connection with the so-called *Internet of Things* (IoT), which deals with the interconnection of almost any object, thus allowing the

dedicated tracking of such object through the use of a Cloud GNSS receiver infrastructure.

D. Multimodal logistics

GNSS-based solutions are particularly well-suited for on-the-route positioning, enabling operators to monitor goods and assets during their transfer between different transport nodes and hubs. GNSS-based data such as positioning and timing can be combined with information on the status of the container and the cargo, as well as with RFID positioning for asset and goods identification at hubs. This information is transmitted to logistic operators and their clients to improve efficiency and effectiveness of transport activities, as well as to manage emergencies by knowing where to act if anything goes wrong [9], [16]. For many reasons, containers are a very good target for GNSS in a multimodal perspective, since they are adopted worldwide and the massive treatment of their data (e.g. as in port terminals) fits very well into a Cloud processing approach.

E. Road-tolling and pay-per-use insurance

In the coming years we will see the advent of new traffic and tax policies regulating the use of roads all across Europe. This is part of a new strategy initiated by many countries in order to fund the costs associated to the maintenance and development of roads, highways, and urban areas. One of these strategies is to automatically charge vehicles for the use they make of roads and highways, or when they enter into congestion charge zones, where traffic is restricted to avoid pollution and traffic congestion. For this application, GNSS has been recognized by EC regulations as one of the main technologies to meet the requirements of new road-charging policies [17]. In this context, the use of a Cloud-GNSS receiver infrastructure can be used to collect and process GNSS data coming from a large population of road users. Additionally, it could also be used to infer valuable information for road-related applications such as flexible insurance coverage, which would charge the user depending on distance travelled, and driving behavior [18].

VI. Conclusion

This paper has introduced the use of Cloud computing infrastructures for developing the novel concept of *Cloud GNSS receiver*. We have presented the main features of one of the major Cloud platforms such as AWS, describing the services and instances best suited for GNSS signal processing. Next, we have discussed the general architecture of a Cloud GNSS receiver developed in collaboration with ESA, where GNSS raw samples gathered with multiple RF front-ends can simultaneously be processed with nearly unlimited computing resources. This is of special interest for applications with computationally demanding techniques, such as indoor positioning, multi-constellation processing, or scientific applications that require collecting and processing GNSS signals over different geographical locations. It is also a very flexible scheme, since new functionalities and compatibility with future signal evolutions can easily be incorporated by updating the Cloud software, regardless of the user terminals. Finally, we have introduced some novel applications of Cloud GNSS signal processing, such as authentication, distributed interference detection or Smart City-related applications, with the aim of stimulating the interest on Cloud-based approaches within the GNSS community.

Acknowledgment

The views presented in this paper represent solely the opinion of the authors and not necessarily the view of ESA.

References

[1] G. W. Hein, J. A. Ávila-Rodríguez, S. Wallner, B. Eissfeller, T. Pany, P. Hartl, "Envisioning a future GNSS system of systems", Inside GNSS, vol. Jan./Feb., pp. 58-67, 2007.

[2] M. Jones, "The civilian battlefield. Protecting GNSS receivers from interference and jamming", Inside GNSS, vol. Mar./Apr., pp. 40-49, 2011.

[3] G. Seco-Granados, J. A. López-Salcedo, D. Jiménez-Baños, G. López-Risueño, "Challenges in Indoor Global Navigation Satellite Systems", IEEE Signal Proc. Mag., vol. 29, no. 2, pp. 108-131, Mar 2012.

[4] O. Vermesan, P. Friess (Eds), *Internet of things: converging technologies for smart environments and integrated ecosystems*, River Publishers, 2013.

[5] ION GNSS SDR Metadata Working Group, "Global navigation satellite systems software defined radio sampled data metadata standard", The Institute of Navigation, Revision 0.1, January 25, 2015.

[6] S. Gunawardena, T. Pany, "GNSS SDR Metadata Standard Working Group Report", Proc. ION GNSS+, pp. 3218-3221, 2015.

[7] J. A. López-Salcedo, Y. Capelle, M. Toledo, G. Seco-Granados, J. Vicario, D. Kubrak, M. Monnerat, A. Mark, "DINGPOS: A Hybrid Indoor Navigation Platform for GPS and GALILEO", Proc. ION GNSS Conference, pp. 1-12, Sept. 2008.

[8] J. A. López-Salcedo, G. Seco-Granados, "Datasheet of the DINGPOS HS-GNSS Software Receiver", SPCOMNAV-UAB, http://goo.gl/TdZnVO

[9] J. A. Volpe, "Vulnerabilty assessment of the transportation infrastructure relying on the global positioning system", National Transportation Systems Center, Office of the Assistant Secretary for Transportation Policy, U.S. Department of Transportation, Tech. Rep., 2011.

[10] J. M. Parro-Jiménez, R. T. Ioannides, M. Crisci, J. A. López-Salcedo, "Detection and mitigation of non-authentic GNSS signals: preliminary sensitivity analysis of receiver tracking loops", Proc. 6th ESA Workshop on Satellite Navigation User Equipment Technologies (NAVITEC), 2012.

[11] P. Y. Montgomery, T. E. Humphreys, B. M. Ledvina, "Receiver-autonomous spoofing detection: experimental results of a multi-antenna receiver defense against portable civil GPS spoofer", Proc. ION International Technical Meeting, pp. 124-130, 2009.

[12] S. Lo, D. De Lorenzo, P. Enge, D. Akos, P. Bradley, "Signal authentication. A secure civil GNSS for today", Inside GNSS, pp. 30-39, Sept/Oct. 2009.

[13] P. Walker, V. Rijmen, I. Fernández-Hernández, L. Bogaardt, G. Seco-Granados, J. Simón, D. Calle, O. Pozzobon, "Galileo open service authentication: a complete service design and provision analysis", Proc. ION GNSS+ Conference, 2015.

[14] W. Kogler, J. Wendel, "Low-cost PRS receiver for positioning and timing enabled by an assistance server", ESNC 2014, http://goo.gl/G6KBbj

[15] Y. Béniguel et al., "Ionospheric effects on GNSS performance", Proc. ESA Workshop on Satellite Navigation Technologies (NAVITEC), pp. 1-8, Dec. 2012.

[16] L. V. Kuylen, J. Leyssens, G. Van Meerbergen, "Using AsteRxi GNSS/MEMS IMU receiver in a container positioning system", Proc. IEEE PLANS Conference, pp. 1027-1034, May 2010.

[17] European Parliament, "Directive 2004/52/EC of the European Parliament and of the Council of 29 April 2004 on the interoperability of electronic road toll systems in the Community", Official Journal of the European Union, L166, pp. 124-143, 30.4.2004.

[18] B. Grush, P. Khalsa, "A new paradigm for using GNSS for road tolling", Proc. ION International Technical Meeting, pp. 617-622, 2009.