Creating datasets for data analysis through a cloud microservice-based architecture

Antonio J. Fernández-García, Javier Criado, Antonio Corral, and Luis Iribarne

Applied Computing Group, University of Almería, Spain
{ajfernandez,javi.criado,acorral,luis.iribarne}@ual.es

Abstract. Data analysis is a trending technique due to the tendency of analyzing patterns or generating knowledge in different domains. However, it is difficult to know at design time what raw data should be collected, how it is going to be analyzed or which analysis techniques will be applied to data. Service-oriented architectures can be applied to solve these problems by providing flexible and reliable architectures. In this paper, we present a microservice-based software architecture in the cloud with the aim of generating datasets to carry out data analysis. This architecture facilitates acquiring data, which may be located in a data center, distributed, or even on different devices (ubiquitous computing) due to the rise of the IoT. It provides an infrastructure over which multiple developer' groups can work in parallel on the microservices. These microservices also provide a reliable and affordable adaptability to the lack of specific requirements in some functionalities and the fast evolution and variability of them, due to the fast changing of client needs.

1 Introduction

Data has a great value today. It can be used to innovate generating new products and services as well as selling them using marketing strategies; to generate new knowledge in science or research facilities; to make faster and better decisions in politics, retail, weather, sport, science, research, real estate, sports or healthcare among multiple others fields; to reduce costs in engineering or industry, and in general, any aspect that needs to reduce resources consumption.

In traditional software systems, data is located in a data center easily accessible. Overtime, given the need to access big data volumes stored in different forms (relational databases, non-relational databases, files, logs...), data can be stored distributed in several data centers. Recently, due to the rise of ubiquitous computing to support concepts such as Smart Cities or Internet of Things (IoT), data can be stored (processed and analyzed) in multiple devices that emphasize proximity to end-users such as smartphones, tablets or sensors distributed in buildings or facilities as well as located along cities (fog computing [6]).

In addition, data has a great variability due to the continuous incorporation of new data sources and forms of acquisition. Furthermore, data consumption and exploitation suffer great variability as people that use data to take decisions need to have the analyzed data at the right time in a changing environment where data sources and the objectives pursued suffer constants alterations. That reasons make that a dynamic data analysis process, capable of dealing with these circumstances, must be implemented [2].

In this paper, given the existence of having many possibilities to exploit the data, which may be complementary or not, and the variability of data sources, a microservices-based architecture is proposed. This architecture enables a dataset generation process from data stored in many sources, devices and locations. The proposal takes into account the objectives variability and different contexts in which data should be analyzed as well as the fast-changing sources of the data.

2 Microservice-based Architecture suitability

The microservices-based architecture structures the application as a modular set of services collaborating together avoiding the monolithic applications difficulty of decomposing or scaling. In this architecture, microservices [5, 7] are independently deployable services where each component in the system is a stand-alone entity that interacts with others across a network with a well-defined interface. Each of the microservices added to the architecture microservices pool has a concrete purpose, significantly different than the others.

This microservices-based architecture will allow several developers (or work teams) interested in creating datasets, work in parallel at any time. They can even use the best suitable technology for each case or use the technology with they are more familiar. This architecture also facilitates Continuous Integration (CI) and Continuous Delivery (CD) due to the facility of produce software in short cycles, ensuring that the software can be reliably released at any time [1, 4]. The whole pool of microservices has defined boundaries and respects the interface-segregation principle (ISP), one of the SOLID [3] principles, that says that a client should not be forced to depend on methods it does not use. This is really useful when it is necessary to refactor, change or redeploy the system.

Furthermore, this architecture allows having concrete microservices to serve one specific purpose coded with that focus. This is highly aligned with the single responsibility principle (SRP) from SOLID, that says that every module should have responsibility over a single part of the functionality, which should be entirely encapsulated by the class. If one microservice falls, the others will continue working properly because their functionality is isolated.

Figure 1 shows the datasets creation process, through a microservice-based architecture. As it can be seen, there is a pool of microservices. Each one of these microservices can connect to a data center, to distributed data centers or to several near-user devices and query them through the data controller in order to obtain the data that needs for its purpose.

3 Creating Datasets

The necessary steps to implement a microservice go from acquiring data of the database to create optimized datasets by applying feature engineering tech-

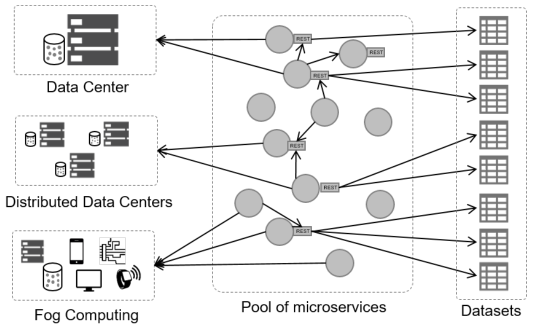

Fig. 1. Architecture of microservices for addressing the creation of datasets

niques. A Web service is included in each microservice to improve its communications and provide access to other microservices or third-party applications. Finally, there is an automatized process that periodically generates the datasets so they can be always updated and accessible without the need of generating them in real time. All these processes are shown in Figure 2.

Data acquisition (#1) is intended to acquire the data to build the dataset. It is necessary to deeply analyze the data centers, databases, logs, sensors or devices that may contain useful data and manage how to access them. When the relevant data is identified, the hosts that contain it are accessed to collect the required data. **Feature engineering (#2)** is a process that transforms the acquired raw data to create features that have better representation and thus, it is possible to create better predictive models. Much of the success that can be obtained by applying data analysis depends on the feature engineering approach that has been following. **Web Service Communication (#3)** may be incorporated to microservices to provide access to processed datasets. As well, in many occasions, communications between microservices occurs and it is necessary to provide the way for this to happen. For that reason, some microservices must implement a Web Service for communication. Datasets from a microservice can be generated on demand by other microservices or by third-party clients accessing the web service (**automation of the generation process (#4)**). To

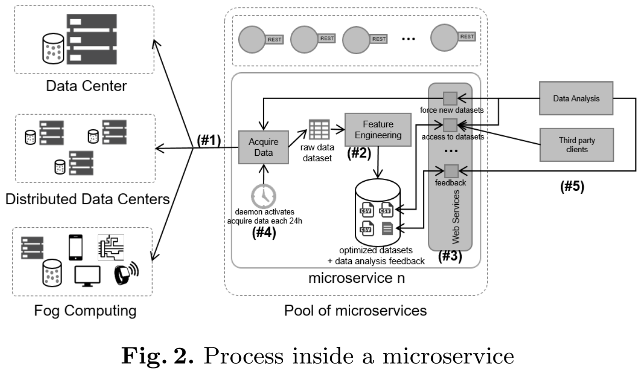

Fig. 2. Process inside a microservice

automatize the process, a daemon can periodically be executed performing the task of calling the APIs of each microservice and generates updated datasets. Occasionally, microservices can receive **feedback from the data analysis process (#5)**. That can happen for many reasons, one of them could be that some knowledge inferred want to be stored from the microservices in order to offer it to others microservices or third party application that can make use of it. In that case, the microservice could be interested in incorporate a web service to expose this knowledge to other microservices or third party clients.

4 Conclusions

A microservice-based software architecture with the aim of generation datasets to carry out data analysis has been presented. The proposed architecture deals with (a) *variability of data:* the architecture proposed is suitable to work on fog computing environments where data is highly distributed in many data centers and multiple near-user devices or sensors, which make it advisable in Internet of Things or Smart Cities environments; and (b) *variability of objectives:* the flexibility of the proposed architecture make it easy adaptable to fast-changing objectives, thus datasets can be continuously updated to deal with changes in requirements as well as new information that decision-maker needs in order to successfully execute strategies.

Also, the proposal architecture can be easily orchestrated with other services due to the microservices granularity that allows multiple teams work in parallel with different purposes that make the architecture advisable to highly configurable projects in constant evolution.

Acknowledgements: This work was funded by the EU ERDF and the Spanish Ministry of Economy and Competitiveness (MINECO) under Project TIN2013-41576-R. A. J. Fernandez-Garcia has been funded by a FPI Grant BES-2014-067974.

References

1. L. Chen. Continuous delivery: Huge benefits, but challenges too. *IEEE Software*, 32(2):50–54, Mar 2015.
2. A.J. Fernandez-Garcia, L. Iribarne, A. Corral, and J. Z. Wang. Evolving mashup interfaces using a distributed machine learning and model transformation methodology. In *OTM 2015 LNCS 9416*, pages 401–410. Springer International Pub., 2015.
3. M. Fowler. *Patterns of Enterprise Application Architecture.* Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2002.
4. J. Humble and D. Farley. *Continuous delivery: Reliable software releases through build, test, and deployment automation.* The Addison-Wesley signature series. Addison-Wesley, Upper Saddle River, NJ, Boston, Indianapolis, 2011.
5. J. Lewis and M. Fowler. Microservices: a definition of this new architectural term. http://martinfowler.com/articles/microservices.html.
6. I. Stojmenovic and S. Wen. The fog computing paradigm: Scenarios and security issues. In *2014 Federated Conference on Computer Science and Information Systems*, pages 1–8, Sept 2014.
7. J. Thones. Microservices. *IEEE Software*, 32(1):116–116, Jan 2015.