A Deployable Data as a Service Architecture for Enterprises

Adrián Tóth[1] and Mouzhi Ge[2]
[1]*Faculty of Informatics, Masaryk University, Brno, Czech Republic*
[2]*Deggendorf Institute of Technology, Deggendorf, Germany*

Keywords: Service Computing, Cloud Computing, as a Service, Data as a Service.

Abstract: Nowadays, data have been considered as one of the valuable assets in enterprises. Although the cloud computing and service-oriented architecture are capable of accommodating the data asset, they are more focused on software or platforms rather than the data per se. Thus, data management in cloud computing is usually not prioritized and not well organized. In recent years, Data as a Service (DaaS) has been emerged as a critical concept for enterprises. It benefits from a variety of aspects such as data agility and data quality management. However, it is still unknown for enterprises why and how to develop and deploy a DaaS architecture. This paper is therefore to design a deployable DaaS architecture that is based on the as-a-Service principles and especially tackles data management as a service. To validate the architecture, we have implemented the proposed DaaS with a real-world deployment.

1 INTRODUCTION

Big Data has received increasing attention in recent years, as organizations and cities are dealing with tremendous amounts of data with high complexity and velocity (Ge et al., 2018). These data are fast moving and can originate from various sources, such as social networks, unstructured data from different devices or raw feeds from sensors (Ge and Dohnal, 2018). Thus, cloud environment has been used to accommodate the Big Data and significantly facilitate the development of service-oriented computing for data management (Park et al., 2021). While service-oriented computing not only provides a better performance of service offerings, perception, and quality of service delivery (Mishra and Kumar, 2018), it also offers a foundation for the development, execution, composition, and integration of business processes that are distributed across the cloud computing network and accessible via standard interfaces and protocols (Serhani and Dssouli, 2010).

Cloud storage and service-oriented computing allows enterprises to focus more on effective cross-platform application data exchange (Khan et al., 2019). However, when an enterprise intends to migrate or build their data center with service-oriented computing, there is lacking of guidelines on how to design and build a Data as a Service (DaaS). Beyond understanding the well-known cloud service models such as Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Software as a Service (SaaS), it is unclear that how the *as a Service* features can be applied to data, and compared to other cloud service models, what the specific components for DaaS are. Thus, DaaS can on the one hand benefit from the general *as a Service* features, on the other hand, data can be managed as a service to potentially enhance the resilience and reusability for data-driven applications (Rao and Nayak, 2019).

This paper therefore proposes a DaaS architecture that can guide users to deploy the data centre on the cloud. It can enhance the data accessibility through different channels, and eliminate the geographical and scalability limitations (Rajesh et al., 2012). We will provide a blueprint of how *as a Service* features can be used for DaaS. The architecture is mainly focused on specific components to construct the DaaS. The proposed architecture is further validated by the physical implementations with real-world deployment.

2 AS A SERVICE

The term *as a Service* has been widely associated with different service models while it sometimes can be misleading or confusing to understand what actually the as a service is (Duan et al., 2015). With respect to the specific functionality, the *as a Service* on the cloud

Tóth, A. and Ge, M.
A Deployable Data as a Service Architecture for Enterprises.
DOI: 10.5220/0010470702780285
In *Proceedings of the 6th International Conference on Internet of Things, Big Data and Security (IoTBDS 2021)*, pages 278-285
ISBN: 978-989-758-504-3
Copyright © 2021 by SCITEPRESS – Science and Technology Publications, Lda. All rights reserved

Table 1: Summary of features in as a service model.

	IaaS	PaaS	SaaS	DaaS
Monitoring	computing resources	software systems	software applications	data transactions
Metering	load of resources	system performance	application usage	transaction attributes
Security	user's responsibility	combined responsibility of user and provider	provider's responsibility to protect application software	provider's responsibility to protect data assets
Tenancy Partitioning	-	system instances	application instances	data pools
Availability	dependent on data center(s)	dependent on infrastructure	dependent on infrastructure or platform	dependent on infrastructure or platform
Scalability	on-deman resources allocation/deallocation	on-deman resources allocation/deallocation	on-deman resources allocation/deallocation	on-deman resources allocation/deallocation
Fault Tolerance	task resubmission, storage backup, etc.	system replicas, load balancing, etc.	application replicas, orchestration, etc.	data protection, data regulation, etc.
Distribution	-	enabled	enabled	enabled

and normal services may have the same functionality but can be organized in different forms. For example, the difference between software and software as a service does not depend on the functionality but on how the functionality is provided. Implementing the *as a Service* method requires to meet a set of requirements that guarantee the functionality provision (Moorthy and Pabitha, 2019). Given that *as a Service* is considered as a subset of cloud service model, there are features that are inherited, meanwhile there are some specific features for data management. By reviewing the common features from IaaS, PaaS and Saas. We have derived the following features for *as a Service*.

- **Monitoring** - it covers the operations and actions connected to the provided service that are later to be observed, checked, and processed for metering. Monitoring results are assets, logs, and audit, which grant repudiation and accountability characteristics to the service itself (Bouasker et al., 2020). These monitoring data are considered as a trusted source of recorded events, operations, and transactions that are substantially significant for further calculations such as billing for the used resources.
- **Metering** - it regulates the resource usage analysis. This analysis takes into account the resource restrictions concerning the rules in the service level agreement (Narayan et al., 2017). The service provider may deliver different service qualities to the customers e.g. lower price for less resources, higher price for unlimited operations. The customers should be familiar with the terms of service and these limitations. Metering can be observed as a special type of monitoring that is required for business and marketing purposes.
- **Security** - It is a fundamental feature in the cloud computing. Each interaction from outside or within the service represents a potential security risk that might lead to exploitation of a security vulnerability (Mthunzi et al., 2020). The data security is critical and it has been moved to the primary security stipulation. *as a Service* introduced dozens of challenging security problems that resulted in a continuous security incident monitoring, reporting, and resolving.
- **Tenancy Partitioning (Multitenancy)** - It ensures that every tenant (user) is isolated from each other and prevents violations between them while each user is using a single instance of the application (Aljahdali et al., 2014). Each tenant has its own operational environment and he is able to work in the space excluding the ability to use others operational environment. For instance, the application database can be configured to serve just for a single tenant, which would prevent storing multiple users' data in the same database (Liu, 2010). There may be an exception when the tenants share spaces consciously with respect to their needs such as collaboration. Tenancy partitioning requires to implement access control and user isolation. From the aspect of SaaS application, this feature has been considered as highly important (Aleem et al., 2019).
- **Availability** - It can be seen as a warranty from a cloud perspective. It is usually expected that the *as a Service* derivations meet criteria of high availability and reliability (Li et al., 2020). As *as a Service* is adopted on the basis of cloud computing, the availability is guaranteed partially by the cloud, which means that the cloud hosted service itself should be also resilient to errors and fault-tolerant.
- **Scalability** - This is the ability to increase or decrease IT resources regarding the actual demand

(Bellavista et al., 2017). Scalability can be differentiated into two types - horizontal and vertical scalability. In vertical scaling we change the computing resources (such as CPU, storage, RAM, etc.) on an existing machine while in horizontal scaling we change the pool of resources by adding a new or removing an existing computing machine. The resource addition is called scaling up and resource removal is called scaling down. Horizontal scalability mostly includes clustering, load balancing fencing and other tools as well as techniques.

- **Fault Tolerance** - It is the capability of a system that can operate continuously despite errors. Most of the applications hosted on a cloud require a high level of fault tolerance that is solved by transparent replication of applications (Mohammadian et al., 2020). We distinguish two types of fault tolerances - proactive and reactive fault tolerance. Proactive fault tolerance means that the application is preemptive by nature by using prevention techniques before the appearance of the error, such as precautionary migrations. On the other hand, reactive fault tolerance reduces the consequences of the already occurred errors using practices for instance backups and rollbacks.
- **Distribution** - A service is usually not a standalone service that provides all the functionalities (Suresh and Varatharajan, 2019). The service may rely on other integrated services used for specific operations and additional resources. The specific responsibilities can be shifted to other services in order to improve efficiency of the business and increase the service quality.

Based on the derived features above, we have compared the IaaS, PaaS, SaaS and DaaS, where how the *as a Service* features can be used on the data is highlighted in Table 1.

The service computing and its *as a Service* model have a beneficial impact on IT related business. The increasing amount of divergent *as a Service* confirms the usefulness and effectiveness (Prasad et al., 2014). When the enterprises intend to use additional service features, these services can be implemented on-demand. This can influence the costs per resources or usage. The underlying architecture of *as as Service* model facilitates the extensibility and adaptive scaling. It reduces processing time and costs for enterprises. Furthermore, the ubiquitous access makes the service widely accessible to a wide range of users.

3 DATA AS A SERVICE ARCHITECTURE

An architecture provides foundation components that reveals the implementation requirements. Thus, the DaaS architecture can be considered as a general template, which includes fundamental concepts and patterns from various disciplines to facilitate the adoption and implementation of DaaS. The DaaS architecture defines an architecture at a conceptual level while the physical levels are designed by the enterprise itself based on their available technical resources. In order to construct the DaaS, data architecture design needs to conforms with the service design. First of all, the architecture needs to ensure that heterogeneous distributed data are provided from its authoritative source, which requires a data acquisition component. Next component processes the acquired data, while it applies the necessary data policies, rules, and regulation. A separate component is allocated for security because there is nothing more vulnerable than the data itself. Afterward, as the data are processed, the data (including meta data) are delivered to the requester in a specific format. There is also a particular need for service management, because the DaaS should meet the service-level agreement criteria and service quality expectations, which are committed between a provider and a client.

We propose the DaaS architecture that includes the following components. All those components are organized in Figure 1.

- Data Acquisition - it formulates an interconnection layer between the system and each registered data sources.
- Data Management - it defines data policies that guarantee consistent data manipulation manners. A set of standardized techniques are required in such an environment, which manage interoperability between diverse and heterogeneous data flows.
- DaaS Engine - this is the core of the DaaS that processes the data request accordingly to the data policies, rules and further data regulations.
- Data Regulations - applies the rules and laws from the area of authority and legislation. The DaaS needs to comply with the established legal entitlements, which are formally defined.
- Security - this is responsible for the continuous protection of DaaS data assets. Cyberattacks are still evolving by the time using more sophisticated methods than before. It is necessary to protect and ensure high security level for each component of DaaS during the entire operation of the service.

- Service Management - it ensures that DaaS will satisfy all the customer expectations and requirements and maximize the business value and progress efficiency of the service. A service needs to continuously improve its capabilities to assure achieving a high customer satisfaction even though the customer data requirements may change from time to time.

3.1 Data Acquisition

The integration of heterogeneous data on distributed data sources introduces two principal challenges - integration of different data sources into one data model, and performing manipulating operations on the data (Khan et al., 2019). Data acquisition represents an interface layer between data sources and the DaaS. This component of DaaS is the foundation layer in DaaS, which ensures proper data acquired from physical data sources.

A data source is an self-organized entity that supplies the data. It can be a database, IoT devices, cloud storage, data warehouse, distributed file system, etc; which might support further operations beyond this data (Weik, 2001). Each data source can have its own interface that implements an access protocol. To integrate the data, the data sources need to support one commonly shared data mapping protocol in order to deal with the problem of data divergence with different communication protocols. Nowadays, there are some promising technologies such as message broker that provides a translation from sender's formal messaging protocol into the receiver's formal messaging protocol, which mitigates the complexity of such a problem.

Before data operations, data discovery is conducted to select the corresponding data sources that includes the necessary data. The data discovery is responsible to select and prepare in advance the required data sources that includes pertinent data, which will be required for the later search execution. The output of the data discovery phase is a particular list of data sources, which also includes supplementary metainformation (Terzo et al., 2013).

Data discovery is followed by data mapping that maps proprietary source data into a standardized data structure. The data mapping transforms the data set into an understandable format and ensures consistency for later processes. The format represents a predefined structure - a schema.

As certain data might be separated and stored in different data sources, data need be aggregated correspondingly. Data aggregation is a process of data transformation with the intent to prepare combined data for subsequent data processing. The data aggregation can be described as a merge of multiple data schema from the data mapping into one entity - more complex schema that will be later processed.

As a fundamental component of DaaS, data acquisition interconnects the whole system with each data source that is managed by this system. Data acquisition solves the data access problems in DaaS such as disparate data formats, access to the data by different interfaces, and data source selection. This component is closely connected to the DaaS engine, which interacts with the data through this component.

3.2 Data Management

Data management includes all the data operations that are more focused on the data values. As Gartner defines the data management as "Data management consists of the practices, architectural techniques, and tools for achieving consistent access to and delivery of data across the spectrum of data subject areas."(gar,), DaaS requires to have a component that deals with the data representation, data coordination, data manipulation as well as data cleaning.

To understand data and provide more comprehensive insights from the data, it is essential to conduct data analysis. Data analysis in the context of DaaS stands for knowledge and pattern discovering from available data sets by using different approaches and techniques such as data mining. The acquired knowledge can lead to enrichment of the basic data set, for instance, data augmentation can be implemented. The results from data analysis can directly support the decision making.

Data control is responsible for procedures related to data permissions, which include access, privileges, ownership, etc. Delivering data demands for data protection from source to destination as well as its value protection from misuse. Data protection from the legal point of view will be described in Section 3.4. On the other hand, the data control in data managements protects the data during its delivery that is followed by its usage. Data control imposes the data source rules and ingestion defined by its owner.

Especially in a Big Data context, the data can be inconsistent or with represented in different scales. Therefore, data standardization is to deal with data inconsistency problem. Different data may represents different obstacles and difficulties associated to tasks. Data standardization defines an uniform format, which facilitates the work associated with such inconsistent data.

Data governance deals with the permissions and specific criteria granted by the data owner to their data

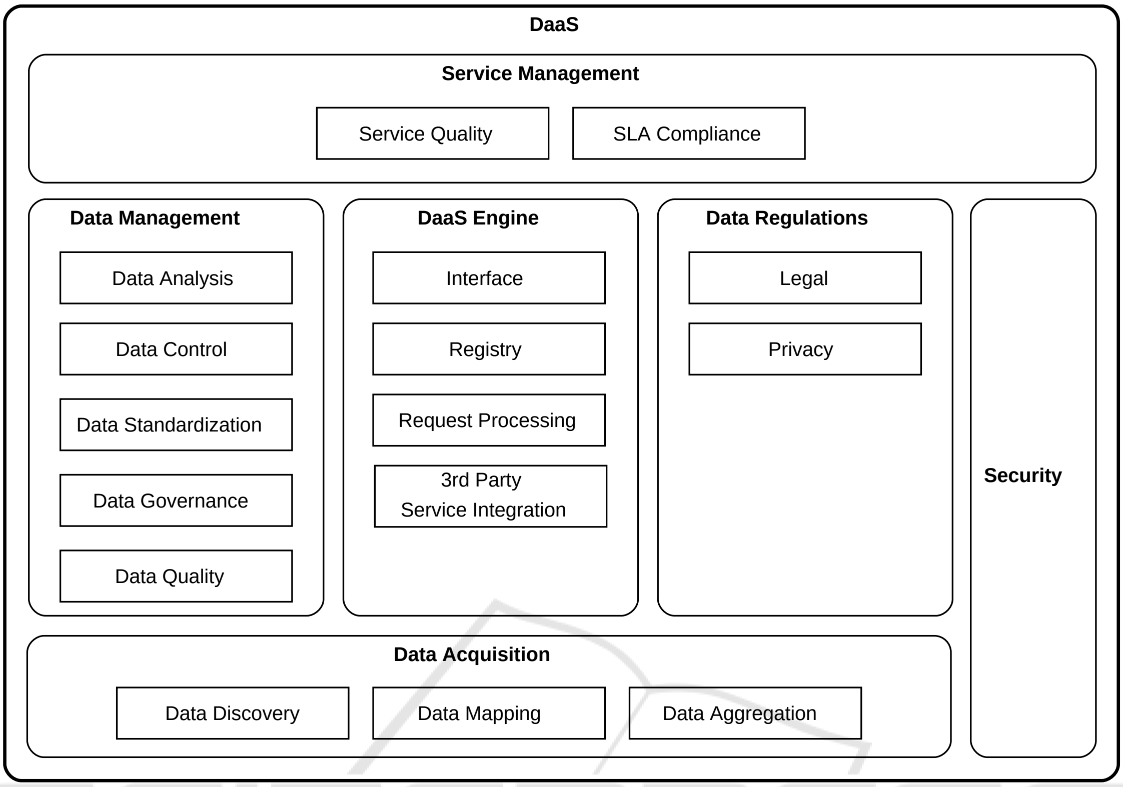

Figure 1: Conceptual architecture of DaaS.

SCITEPRESS SCIENCE AND TECHNOLOGY PUBLICATIONS

sources. Some of the sensitive data may be prohibited, allowed, or in a specific case provided under determined restrictions and limitations such as partial censorship. Overall, data governance regulates the valuable information of data.

Data quality can directly affect the efficiency and effectiveness of organizations and businesses (Ge and Lewoniewski, 2020). DaaS is based upon data delivery that is associated with data risks and threats. To minimize and eliminate data quality problems, it is essential to focus on data quality improvement. Data quality in DaaS provides a qualitative and quantitative cleaning process for the available data, which leads to the improvement of the overall data analysis provided by DaaS.

3.3 DaaS Engine

DaaS engine represents the core component of the DaaS. It is a system that implements the DaaS policies and regulates each data flows. This component in the DaaS is mainly focused on operating efficiently data flows between the data source and data consumer that are managed and controlled in relation to the defined policies.

The DaaS engine consists of two fundamental parts: the DaaS system and the DaaS registry. The system is responsible for the interoperability - information exchange between two end devices. The data consumer can access the DaaS via an interface. The interface requests are processed by the DaaS system. The DaaS registry represents a catalog that includes meta information about the data, which are used for incoming requests in the DaaS. The registry contains information that includes the data relation, location, accessibility, etc., which facilitate to understand the data. The DaaS system then provides authoritative source data to the data consumer in a standardized structured data that can be understood on a successfully processed request. The DaaS engine might be integrated with other third party services, which enrich the feature set of the DaaS. The integration is controlled and accompanied by the conditions of service usage that entail particular responsibilities.

3.4 Data Regulations

Data regulation is responsible for definition of data protecting policies in the DaaS. This component of the DaaS is a connecting point between the cloud service and the area of authority and legislation, which influence the data and its processing. By implementing a proper data regulation, the DaaS can minimize the data privacy and data protection related issues in the cloud environment such as illegitimate data dissemination.

With the rapid development of IT and globalization, Big Data introduces new data challenges, and results in newly defined policy to protect the data and its further processing in IT. DaaS needs to act in accordance with the established legal and privacy rights. The private data stands for every piece of information that is considered as personal. Data protection rights have a high effect on how the service is provided by data providers and how the data consumers use the subscribed service. Data regulation is responsible for the formal definition of each use case necessary from the legal point of view that has to be implemented into DaaS.

Obligations derived from data protection and data privacy can significantly influence the DaaS. For example, in Europe there are several rule enforcement that defines these obligations such as GDPR. These rule enforcements mandate the enterprise to adhere to these obligations, which process or collect any type of information related to its subscribers. The obligations are influencing the data provider as well as to the data consumer.

3.5 Security

With the widespread presence of cyberattacks and security breaches, the data security plays a significant role in IT management (Michener, 2020). Data security protects data assets from undesired disclosure, modification, exploitation, and destruction, whether accidental or intentional. DaaS is particularly vulnerable, and may involve sensitive and confidential data. Securing DaaS system requires to protect the data flow from source to destination.

DaaS is a cloud service model and most of its components such as data sources are integrated via network connection. Thus, it requires different security operations, for instance, active data in the transitions, data in an established communication channel between endpoints, or persistent data stored in the cloud data storage. Securing such a complex and constantly changing environment needs to provision a continuously innovated scheme in each application area of such a DaaS system to ensure the security. Moreover, data fraud is becoming more prevalent. Enterprises may use different approaches such as watermarking or censorship to protect the sensitive data to not being exposed, copied, and stolen. However, it will devalue the expected information. This raises the issue of trust from both business and customer. For example, when the data are delivered to users, the consuming side also needs to have the protection responsibility.

3.6 Service Management

The increase of data services leads to a growing demand for service management that is to control to the overall service provision. Enterprise activities that are performed to manage, control, deliver and operate data services offered to customers can be managed accordingly. The service management has a crucial role for the DaaS provider, as it ensures the customers' expectations for the provided service.

as a Service instances are associated with service delivery, as the name suggests, which delivers value to the customers/subscribers. Service management is fundamental because it drives the associated areas such as service strategy, service goals (including objectives), service metrics (e.g., key performance indicator), etc. The key role rests in a clear, well-defined roadmap that will reduce costs and improve the efficiency of the provided service.

The service management in DaaS also includes service delivery and IT service management. Transformation shift requires firstly to adopt a service delivery model, which defines roadmap and establish service blueprint for data services. It is required to have a clearly defined goals, strategy, mission and vision of the data service. Service delivery deals with the service subscribers respectfully. Furthermore, service management also tackles the DaaS operation management and continuous improvement, which allow productive and efficient data delivery service.

4 IMPLEMENTATION

A physical architecture represents a deployable architecture instance that describes operational characteristics of the designed concept. In the physical instance, we show a physical architecture in Figure 2. It is composed of real-world tools and services. This deployment consists of existing technologies and services that are optimized for the desired purpose in this instance.

The enterprise needs to obtain the requirements for DaaS provisioning such as cloud platform, cloud storage, etc. The platform ensures essential features in the cloud. In our deployment, we have chosen the Google Cloud Platform as a PaaS provider from the available alternatives such as Amazon Web Service, Microsoft Azure, Oracle Cloud, IBM Cloud etc.

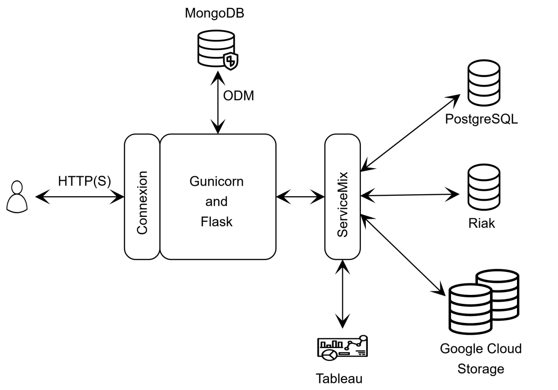

Figure 2: Deployable physical architecture of DaaS.

The usage of enterprise service bus (ESB) provides a efficient method for multiple divergent system integration and management. Nowadays manufactures released and announced message-oriented middleware appliances such as Apache ServiceMix[1] that can realize the ESB communication system. The ESB is responsible for information exchange in real time between subsystems of DaaS. The usage ESB has the advantage to build a loosely coupled system.

Furthermore, DaaS Engine delivers the requested data to consumers or applications. The engine can be implemented in various ways, we have chosen *Flask* web framework and *Gunicorn* as our WSGI[2] while both of them are implemented in a Python language, which is a general object oriented imperative programming language. The *Gunicorn* implements two roles - server side (requests handling) and application side (request delegation). The Gunicorn invokes the corresponding python callable on the incoming request. In our deployment, the *Flask* is the callables and responsible for processing requests in the data management. Also, using Flask, there is a possibility to implement the data-related validation rules and policies in a custom way. Flask is a framework written in Python and supports the object mapping (in our case, object document mapping) of database items. The specific rules can be isolated and separated from the application logic and stored in a persistent data storage, for instance, in a database - MongoDB.

One of frequently used approaches to integrate data to other application is via application protocols such as HTTP or HTTPS. The access to the DaaS is enabled through a RESTful API, which is based on OpenAPI standard and implemented via Connexion that maps each endpoint to the Python callables.

Third-party service integration is to enrich the provided basic feature set of the DaaS. In most cases, the external service is added as an encapsulated service entity that is managed with respect to the internal policies and terms of service. The integration process will connect the specific service by mapping the service interface to the message-oriented middleware. The integration result is a connected service, which can exchange data with other components of the DaaS.

Current cloud storage providers offer a wide range of storage services such as backup and versioning. There is a large amount of options on how to implement cloud storage within a DaaS. Regarding the price pool and available resources, the cloud storage may be an integrated service provided by a cloud storage provider or a custom cloud storage. For large scale data volume, using existing cloud storage may be a cheaper option to the enterprise in contrast to re-implement the cloud storage. In our example, we use an already existing cloud storage - Google Cloud Storage, that is used for storage purposes of the DaaS.

5 CONCLUSIONS

In this paper, we have proposed a Data as a Service architecture to guide users to build, migrate and deploy data management on the cloud. The features of the *as a Service* are derived from the IaaS, PaaS and SaaS. Those features are then applied as a foundational setting for DaaS. The DaaS architecture further deal with specific data functionalities along with data flow in the cloud computing. In order to validate the proposed DaaS architecture, we have demonstrated how to instantiate the DaaS architecture to a deployable physical architecture. During the implementation, we have reported the lessons learned from the DaaS implementation. It can be seen that the proposed DaaS can be applied in real-world deployment and can significantly help the enterprises to build their DaaS.

As future works, the proposed implementation can be further developed and enriched by performance benchmarks of the deployed instance including different tools and technologies.

REFERENCES

Gartner glossary. https://www.gartner.com/en/information-technology/glossary/dmi-data-management-and-integration.

Aleem, S., Ahmed, F., Batool, R., and Khattak, A. (2019). Empirical investigation of key factors for saas architecture dimension. *IEEE Transactions on Cloud Computing*, pages 1–1.

[1] A framework composed of Apache ActiveMQ, ApacheCamel, and Apache CXF, and Apache Karaf.

[2] Web Service Interface Gateway

Aljahdali, H., Albatli, A., Garraghan, P., Townend, P., Lau, L., and Xu, J. (2014). In *8th IEEE International Symposium on Service Oriented System Engineering*, pages 344–351. IEEE Computer Society.

Bellavista, P., Corradi, A., and Zanni, A. (2017). Integrating mobile internet of things and cloud computing towards scalability: lessons learned from existing fog computing architectures and solutions. *Int. J. Cloud Comput.*, 6(4):393–406.

Bouasker, T., Langar, M., and Robbana, R. (2020). Qos monitor as a service. *Softw. Qual. J.*, 28(3):1279–1301.

Duan, Y., Fu, G., Zhou, N., Sun, X., Narendra, N. C., and Hu, B. (2015). Everything as a service (xaas) on the cloud: Origins, current and future trends. In *2015 IEEE 8th International Conference on Cloud Computing*, pages 621–628.

Ge, M., Bangui, H., and Buhnova, B. (2018). Big data for internet of things: A survey. *Future Gener. Comput. Syst.*, 87:601–614.

Ge, M. and Dohnal, V. (2018). Quality management in big data. *Informatics*, 5(2):19.

Ge, M. and Lewoniewski, W. (2020). Developing the quality model for collaborative open data. *Procedia Computer Science*, 176:1883–1892.

Khan, F. A., ur Rehman, M., Khalid, A., Ali, M., Imran, M., Nawaz, M., and Rahman, A. (2019). An intelligent data service framework for heterogeneous data sources. *Journal of Grid Computing*, 17(3):577–589.

Li, C., Song, M., Zhang, M., and Luo, Y. (2020). Effective replica management for improving reliability and availability in edge-cloud computing environment. *J. Parallel Distributed Comput.*, 143:107–128.

Liu, G. (2010). Research on independent saas platform. In *2010 2nd IEEE International Conference on Information Management and Engineering*, pages 110–113.

Michener, J. R. (2020). Security issues with functions as a service. *IT Professional*, 22(5):24–31.

Mishra, S. and Kumar, C. (2018). Effort estimation for service-oriented computing environments. *Comput. Informatics*, 37(3):553–580.

Mohammadian, V., Navimipour, N. J., Hosseinzadeh, M., and Darwesh, A. (2020). Comprehensive and systematic study on the fault tolerance architectures in cloud computing. *J. Circuits Syst. Comput.*, 29(15):2050240:1–2050240:40.

Moorthy, R. S. and Pabitha, P. (2019). Optimal provisioning and scheduling of analytics as a service in cloud computing. *Trans. Emerg. Telecommun. Technol.*, 30(9).

Mthunzi, S. N., Benkhelifa, E., Bosakowski, T., Guegan, C. G., and Barhamgi, M. (2020). Cloud computing security taxonomy: From an atomistic to a holistic view. *Future Gener. Comput. Syst.*, 107:620–644.

Narayan, A., Pillai, P. S., Prasad, A. S., and Rao, S. (2017). Resource procurement, allocation, metering, and pricing in cloud computing. In *Research Advances in Cloud Computing*, pages 141–186.

Park, J. H., Younas, M., Arabnia, H. R., and Chilamkurti, N. K. (2021). Emerging ICT applications and services - big data, iot, and cloud computing. *Int. J. Commun. Syst.*, 34(2).

Prasad, A., Green, P. F., and Heales, J. (2014). On governance structures for the cloud computing services and assessing their effectiveness. *Int. J. Account. Inf. Syst.*, 15(4):335–356.

Rajesh, S., Swapna, S., and Reddy, P. (2012). Data as a service (daas) in cloud computing. *Global journal of computer science and technology*.

Rao, K. R. and Nayak, A. (2019). Data residency as a service: a secure mechanism for storing data in the cloud. *Int. J. Embed. Syst.*, 11(4):397–418.

Serhani, M. A. and Dssouli, R. (2010). Case study: Master of science in service computing (msc sc). In *2010 6th World Congress on Services*, pages 80–83.

Suresh, A. and Varatharajan, R. (2019). Competent resource provisioning and distribution techniques for cloud computing environment. *Clust. Comput.*, 22(5):11039–11046.

Terzo, O., Ruiu, P., Bucci, E., and Xhafa, F. (2013). Data as a service (daas) for sharing and processing of large data collections in the cloud. In *Seventh International Conference on Complex, Intelligent, and Software Intensive Systems*, pages 475–480.

Weik, M. H. (2001). *data source*, pages 358–358. Springer US, Boston, MA.

SCITEPRESS SCIENCE AND TECHNOLOGY PUBLICATIONS