Cloud and Application Programming Interface – Issues and Developments

Isaac Odun-Ayo, *Member IAENG,* Chinonso Okereke, Hope Orovwode

***Abstract*—T Cloud computing is a unique IT paradigm that is fast gaining popularity in organizations and enterprises. The cloud has made it possible to store vast amount of data based on infrastructure provided by cloud service providers. The users are able to scale storage up or down on demand. The users are also able to utilize applications already deployed on the cloud by the cloud service providers. Other cloud service providers such as OpenNebula, CloudStack and OpenStack have made it possible to develop applications and deploy them on the cloud. Application programming interfaces (APIs) are available to guarantee such deployment. As the name implies, APIs serve as interfaces between programs. The interface is usually between a software developer and software developer's application. Basically, the API allows one software to access some functionalities of another software. This paper examines present trends in the area of API and provides a guide for future research. Papers published in journals, conferences, white papers and those published in reputable magazines were analysed. The expected result at the end of this review is the identification of trends in API. This will be of benefit to prospective cloud users and even cloud providers.**

Index Terms - Cloud computing, Application Programming Interface

I. Introduction

NIST defines cloud computing as "a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources." These computing resources include networks, servers, storage, applications, and services. Cloud computing makes for quick provisioning release of these resources with little management effort [1]. It involves hardware and software resources used to deliver services over the Internet to users by Cloud service providers (CSP). The Cloud user does not have to worry about on-premises infrastructure and maintenance because of the provision of scalable on-demand services on a pay-as-you-go basis

Manuscript received December 08, 2017; revised January 05, 2018. This work was supported in part by the Covenant University through the Centre for Research, Innovation and Discovery (CUCRID).

I. Odun-Ayo is with the Department of Computer and Information Sciences, Covenant University, Ota, Nigeria. (+2348028829456; isaac.odun-ayo@covenantuniversity.edu.ng)

H. Orovwode is with Department of Electrical and Information Engineering, Covenant University, Ota (hope.orovwode@covenantuniversity.edu.ng)

C. Okereke is with the Department of Electrical and Information Engineering, Covenant University, Ota, Nigeria (chinonso.okereke@covenantuniversity.edu.ng).

It involves hardware and software resources used to deliver services over the Internet to users by Cloud service providers (CSP). The Cloud user does not have to worry about on-premises infrastructure and maintenance because of the provision of scalable on-demand services on a pay-as-you-go basis. Cloud services are automatically provided when needed by the consumer [2]. Cloud consumers can also access resources over the Internet anytime, anywhere using different kinds of computer platform. The cloud offers primarily three kinds of services namely: Software-as-a-Service (SaaS), Platform-as-a-Service (PaaS) and Infrastructure-as-a-Service (IaaS). In SaaS, the Cloud provider hosts software such that users have no need to install, before utilizing it [3]. PaaS provides an environment for users to develop and display their own applications on the web using the facilities of the CSP. IaaS offers storage and computing resources used in enterprises to deliver business solutions [3]. PaaS has been noticed to however be on the declined [16]. One major reasons are the restrictions posed by PaaS on developers. With PaaS developers are restricted to certain available tools, languages and a conditioned environment. The drawback of a conditioned environment is that such environment has many limitations and creates complexities for the developer who sometimes needs to access resources and tools to support specific features such as remote and native API.

Virtualization is utilized in IaaS and services can be scaled up or down depending on user demand. Cloud computing also has four deployment models. In private clouds, computing resources are operated in-house, on-premise by the enterprise or organization [2]. Private Cloud can be managed in-house through a CSP, and it is considered more secure because it is run by the staff of that enterprise. Public Clouds involve services provided off-premise by major CSPs such as Google, AWS and Microsoft Azure. The infrastructure is owned and managed by the CSP, who provides the services over the Internet. However, public cloud is considered less secure. Hybrid clouds is a combination of private, public or community clouds. The clouds component is bound together by standardized technology, and managed by a single unit, yet each cloud remains a unique entity [2]. Community Cloud is a combination of several organizations sharing resources in the form of a private cloud. Educational Clouds used by universities for research and education is a typical example. Hybrid Clouds allows organizations to optimize their resources by running critical and core activities on premises, while outsourcing auxiliary functions to public clouds.

The Software Engineering Institute defines Application Programming Interface (API) as "...an older technology that

ISBN: 978-988-14047-9-4
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

facilitates exchanging messages or data between two or more different software applications. API is the virtual interface between two interworking software functions, such as word processor or spreadsheet. API is the software that is used to support system level integration of multiple commercial-off-the-shelf (COTS) software products or newly-developed software into existing or new applications" [4].

A cloud API is basically used to integrate applications in order to provide inter-cloud compatibility and improving cloud usage experience [3]. They are broadly categorized into two: in-process API and remote API. In-process APIs are used on a regular basis in a typical infrastructure based IT environment. Remote APIs are used to develop cross-border, bridging applications. Web services APIs include SOAP and REST, while remote calls include SUN RPC and JAVA RMI; and application dependent protocol include FTP and SNMP [3]. These APIs communicate based on data structures like JSON and XML and make use of GET, PUT, POST, DELETE request. Cloud applications usually utilize remote APIs.

The purpose of this paper is to discuss cloud computing and APIs. The paper examines the structure of API and its role in Cloud applications. The current industry trend in the area of cloud computing and API is highlighted. This paper is expected to contribute to the understanding of the current issues in API and cloud computing. The remaining part of the paper is as follows: Section 2 examines related work; Section 3 discusses architecture and issues of Cloud API, while section 4 highlights current trends of API in industry; the conclusion and future work are in section 5.

II. Related Work

Cloud API issues: an empirical study and impact in [5], discusses major findings related to APIs using the Amazon EC2. Various categories of API failures were outlined and the impact also discussed. Towards transparent integration of heterogeneous cloud storage platforms in [6] the incompatible nature of APIs being provided for cloud storage. It proposes a framework for the integration of cloud systems and local storage infrastructure. A process for use of APIs across various cloud services was examined with a view to enhancing application independence. Automated Control in Cloud Computing: Challenges and Opportunities in [7] proposes the proportional thresholding framework for APIs. This allows for effective design and use of cloud APIs across a wide variety of services. Secure authentication of cloud data mining API in [8] examines cloud APIs in terms of data mining. It discusses cloud API security concerns and suggests an appropriate mechanism. A common API for delivering services over multi-vendor cloud resources in [9] considers the issue of applications being provider-dependent. In view of this a platform is presented using APIs that allows interoperability in dealing with service providers. A Critique of the Windows Application Programming Interface in [10] examines the nature of the API in the Windows system.

The observed problems were enumerated with a view to enhance application portability. "Customising graphics applications: techniques and programming interface" in [4], focuses on API for graphical purposes. An API was also developed based on the openGL for graphics application. Sometimes you need to see through walls — a field study of application programming interfaces in [11] discusses the importance of API and roles in systems development. API facilitates cooperation in software development including coordination among developers. The Java Platform in [12] provides very basic examples of the Java runtime environment and the different types of the APIs being utilized by Java. It simple provides an insight into the Java platform and its constituent parts.

III. Types of Application Programming Interface

In-process API is a combination of objects methods, functions or procedures, abstracting the resources in terms of memory usage, data structures and opaque pieces of machine executable or interpreted code [5]. Developers use this on a regular basis because in-process APIs are very efficient.

Remote APIs are used for bridging applications. These are in the following forms:

- Web services API – SOAP and REST.
- Remote calls – SUN RPC, JAVA RMI and AME.
- Message Passing – AMQP, STOMP.
- Application dependent protocols – FTP, SMMP.

These APIs are abstracting the resources as reactive opaque agents that communicate based on fully transparent immutable date structures like XML, JASON. However, remote API are harder to use. To ensure interoperability and programming language independence, there are open source project providing custom binding for a wide range of programming languages [5]. There are standardization bodies at the network level, but none is targeting in-process APIs. Again, most libraries either incompletely implementing the proposed standards or try to wrap the lowest denominator of competing APIs. Hence, the cloud application developer is forced to choose a feature-full library created by a particular provider, targeting the provider's resource interfaces, thereby getting locked-in [5]. Alternatively, the developer has to choose a lowest denominator, standards complaint resource API, in a rather unreliable implementation state. Cloud APIs are in three categories. The infrastructure APIs deal with provisioning and configuration of resources. The platform APIs offer capabilities in terms of services. The application APIs are used to run applications on the Cloud.

A. Cross – Platform Application Programming Interface

A vital aspect of Cloud APIs is its ability to allow operations across different cloud platforms. There are four aspects namely: programming language, platform specific services, data storage and platform specific configuration files. The specific programming language that an application has been built on is a determinant of cross platform application. Each Cloud platform supports certain languages, version and framework.

a) Data Storage

Database stores and file stores are an essential aspect of any application. Database stores use the usually relational database

ISBN: 978-988-14047-9-4
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

format, while file stores provide simple storage [7]. All Cloud applications will require access to data in a database. Such data can be in SQL or noSQL format. Conflict may arise in terms of incompatible data structure, differences in query language and data migration. This is because databases use their own API or query language. In terms of file storage, different platforms provide a custom proprietary API in order to allow applications interact with storage space. An application to be moved to another platform implies storage API must be amended to fit the target provider.

b) Platform Specific Configuration Files

Cloud platforms also require configuration files for executing applications. This will differ between platforms

B. Standardization and Intermediation

The NIST [7] has information about cloud computing standards pinpointing what interfaces need to be standardized in the various types of cloud computing services. There are several active standardization organizations.

a. Distributed Management Task Force using the Open Virtualization format for virtual machines.
b. Storage Network Industry Association using the cloud data management interface for cloud storage.
c. Open cloud computing interface for virtual machines
d. Topological and Orchestration Specification for cloud application aimed at standardizing application for cross platform deployment.

Intermediation decouples application development from specific platforms APIs. Developers can create applications using intermediate APIs that hide the proprietary API of a CSP. The intermediate layer prevents a developer from being bound to a particular programming language or data store. The following are intermediation formats [7]:

a. jCloud API

jCloud API is an open source library capable of abstracting cloud vendor specifics that can be used by an application developer to abstract cloud vendors specific APIs. jCloud API is offered in Java and Closure programming languages. The storage services of jCloud (Blob store) consists of a container which is the top level directory, and blobs (Binary Large Object) that contains data to be stored and folders for organizing blobs.

b. Open Source API and Platform for Multiple Cloud Open Source API and Platform for Multiple Cloud (mOSAIC) is an open source platform that enhances portability; it provides an abstraction layer from the native APIs. mOSAIC API supports the use of Cloud database file storage and communication services.

c. Bridge Query Language

Bridge Query Language (BQL) was introduced to handle the issue of database. BQL bridges SQL and noSQL and translates BQL query to traditional query supported by the source database system.

B. Open Source API and Platform for Multiple Cloud

Examples of exiting Open Source APIs include jCloud for Java, LibCloud for Python, SimpleCloud for PHP and Dasein Cloud for Java [9]. For portability of application, components should be move are easily moved and reused regardless of the provider location, operating system, storage and API. Approaches can be classified as building or using open APIs layers of abstraction and repository. All of these are language dependent and offer a thin abstraction layer towards the representation of the resources. More complex APIs are provided by Open Nebula, Appistry or OpenStack. All of these solutions can be in three categories. APIs with multiple independent implementations like Eucalyptus, EC2, AppEngine and AppScale are in the first category. Second are APIs runnable on multiple clouds not necessarily through multiple independent implementations, are several implementations of MapReduce. Finally, APIs that separate the application into application logic layer from the cloud layer. mOSAIC design is unique for the following reasons:

a. An API that allows separation in application logic layer from the cloud layer.
b. Application independence from both providers, language and programming style.
c. A cloud ontology and semantic resource discovery mechanism.

The main design requirement is focusing on the developer of a cloud application who, using mOSAIC, does not need to worry about issues pertaining to a specific cloud provider, but instead on its application and later deploy it on any cloud provider the developer wishes. mOSAIC design offers high degree of flexibility and control to the developer by introducing several levels of abstraction. On the highest level, mOSAIC allows the developer to control its application completely. The mOSAIC user can operate on four layers [9].

a. Application tools layer is for developing new application, built from scratch.
b. Provisioning layer is designed for owners of existing applications intending to run their application on a cloud environment.
c. Development system layer is for debugging and for manual control of resources.
d. Resources system layer is for use by resource providers in terms of debugging and billing [9]

IV. Industry Consideration of Application Programming Interface

From IBM perspectives [13] integration falls into three categories as discussed below:

a. Deployment or orchestration tools that install applications on the cloud. These tools will deploy cloud apps and will also integrate multiple cloud components. Some may be flexible enough to support integration of non-cloud components.
b. Open source tools for orchestration and integration that may not be a service of cloud provider but can support the provider's cloud deployment and connection needs.
c. Commercial integration tools from software companies like IBM, Microsoft and Oracle.

ISBN: 978-988-14047-9-4
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

Users must use orchestration tools with caution especially if there may be need to change cloud provider. Using orchestration tools will require adjustment of cloud integration process for the target provider. Therefore, it is better to avoid provider-specific integration tools if there will be a movement to another cloud. There are commercial integration tools which can evolve from application development and application life cycle management. Even when an enterprise intends to deploy on a single vendor, it is important to look at the vendor's tool list. However, not all application developers' tools are fully capable of cloud deployment. Interestingly, Open source integration tools are considered the best solutions. Most open source tools can be used with all the popular public cloud deployment types. Although, most Open Source tools are easier to use with IaaS than PaaS or SaaS. The tools allow for integration of public cloud components with popular Open Source cloud stacks such as OpenStack, Apache CloudStack and Eucalyptus. A major constraint of Open Source is that the integration may insist on deploying a private cloud service for integration with public cloud. However, most commercial integration tools allow integration of data center component running on bare server or virtualization. It is best to look for integration tools and practices that accommodate extensive application componentization that support directory redundancy and performance enhancement and having the widest range of interfaces and directory options.

A. Application Integration from Oracle Cloud Platform

Oracle has a broad range of integrated cloud services to assists cloud consumers to reduce the cost of developing, testing and deploying applications. The subscription-based PaaS offerings allow enterprises to develop and deploy nearly all types of applications including enterprise apps, lightweight container apps, web apps, mobile apps etc. Oracle cloud products are based on prevailing Java standards, so developers can use familiar architectures, utilities and products and then deploy their app on-premise or to the public cloud and private clouds. Enterprises can also build simple applications without coding. To limit the complexities of developing and deploying apps using JEE, Oracle introduced the Oracle Java Cloud Service (JCS). JCS Cloud offering boost innovation and speed with the complexities of configuring services and infrastructure. Oracle provides a fully-managed instance of Oracle web logic server in the cloud with JCS. It can utilize dedicated virtual machines running preconfigured web logic clusters. Users can choose the type of cores, amount of memory, scaling in a self-service interface. The JCS allows development and deployment of applications in minutes rather than days. JCS users can also use the Oracle Database Cloud Services. JCS also has facilities for applications storage and computing, and the Oracle Developers Cloud Service is integrated also.

B. Cloud Computing Application Programming Interface Management

A cloud service API is critical for integrating enterprise applications and workloads with cloud environments. There are applications that must connect to various cloud services and APIs provide the set of protocols that creates a bridge between those workloads and the cloud. APIs must be available to both Windows and Linux developers since it should be made public so that developers can know how to use them. Cloud computing APIs should also be scalable for increased demands. API security is important, since users use the same authentication and authorization services like other enterprise interfaces. A well-designed API should also be in use by the designers of the API. [14].

Cloud computing APIs have attack surfaces that make them vulnerable to injection attacks. A hacker can send fake API commands to the application and its component which can compromise the application. To prevent attacks, developers should understand the APIs well and know how to implement them. Sessionless security practices and token-based authentication can help prevent attacks. User's names and passwords should not be placed inside simple object access protocol headers. In addition, frequent security test should be carried out on Cloud computing APIs.

Cloud governance models and the use of Open Source Cloud Platform can be used to reduce problem of vendor lock-in. In order to reduce the disruptions caused by Cloud providers API updates, developers should look for providers who design their APIs with long term commitment to feature like syntax and function calls. This is because every update by the CSP will require developers to conduct the tedious process of updating, retesting and patching their applications. If an API receives numerous calls or its being utilized by more users, it will be low in performance and unstable. A user could use management tools to throttle up API calls when usage increases. Cloud is simply a collection of APIs and services [15]. Infrastructure-based API perform functions such as placing data into storage, while application-based APIs handle computation task. There is the need for policy and access controls to APIs. An application may eventually need to run in a multi-cloud environment hence it's risky to choose API management tool from only one public Cloud provider. Management tool for API should be heterogeneous. API management tools should be integrated on the cloud security tools, to avoid outside attacks.

V. ANALYSIS AND DISCUSSION

Application programming interface (API) in cloud computing has become a very important issue. In this research, 7 core cloud computing areas vital to cloud computing and API are selected for analysis. As shown in Table 1, the areas are virtualisation, data storage, web application, application programming interface, identity management, general security concerns and machine learning. No reviewed paper covers all the areas under consideration this is because the addition of machine learning to the consideration areas. However [22] covered all the areas under consideration with the exception of machine learning, while [23][28][29] at least mentioned about 70% of all the core areas aforementioned. Virtualisation issues was discussed by over 50% of the papers examined. Data storage as it relates to Cloud API was also examined by 50% of the papers reviewed. Web application issues was discussed by 58% of the references. Application programming interface concerns were mentioned in all the papers examined. Identity management and access control issues receive the

ISBN: 978-988-14047-9-4
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

least attention security concerns was considered by 26% of the papers examined. Another important aspect of security on the cloud which is identity management and access control with 17% discussed in the literature reviewed. General security concerns were discussed by almost 42% of the papers examined.

TABLE I.
COMPARATIVE ANALYSIS OF CLOUD API AREAS

References	Virtualization issue	Data storage issue	Web Applications	Application Programming Issues	Identity Management and Access Control	Security Concerns	Machine Learning
H. W. v.d Westhuizen, G. P. Hancke(2017)	x	x	x	x	x	x	
Reginaldo R, et al (2018).		x		x		x	
J. Cito et al (2015)	x	x	x	x			
Luís A et al (2013)	x	x	x	x	x	x	
Q. Lu et al (2013)			x	x		x	
P. Musavi et al (2016)	x	x		x		x	
Wang, Bai, Li, et al., (2017)	x		x	x			
Wua and Lee, (2013)	x	x	x	x	x	x	
Baucke *et al.* (2015)	x	x		x	x	x	
Wang, Bai, Ma, et al., (2017)			x	x			
Cretella and Di Martino (2012)	x		x	x		x	
Lu et al.(2015)				x			
Mayoral et al. (2017)	x			x			
Hosseini, Xiao and Poovendran, (2017)		x	x	x		x	x
Satyal et al (2017)	x	x		x		x	x
Jinlong et al. (2017)		x	x	x			
Xu et al. (2017)				x			x
Ferguson, Vardeman and Nabrzyski, (2016)		x	x	x			x

From the foregoing, not all the core areas being considered in this work received full consideration by the papers reviewed. Asides application programming which is applicable to all papers reviewed API in terms of web application received the highest consideration. This is expected seeing that web application plays an important role in application programming interface. Most recent papers among the reviewed papers seem to all give focus on machine learning which suggests a new trend in the study. On the other hand, issues such as virtualisation and identity management were expected to have been discussed the same way as data storage, but this was not the case. Due to the high trend in web application and application programming interface, as expected that issue received more attention. Based to the percentage of reviewed core area, it is seen that a lot more still needs to be done in terms of identity management and security considerations in cloud computing. What seems to be the recent trend is the research into the use of adaptive and machine learning in API. However most of the overall issues raise are those of standardization of services, interoperability, multi-vendor clouding effect on API, improvement on virtualization techniques, increasing API and fault tolerance for web application and adapting to mobile cloud computing and heterogeneity of interface.

VI. CONCLUSION

Cloud Computing APIs are strategic to the success of the operation of the types of cloud services and deployment types. The paper examines the API architecture and different types of API. The proprietary and Open Source APIs were discussed. The industry perspective on APIs was also highlighted. As the development of cloud computing APIs is still at an early stage, we hope this research work will contribute a desirable understanding of the design and implementation challenges of cloud computing APIs, and act as a major contribution for further research in this area.

ACKNOWLEDGMENTS

We acknowledge the support and sponsorship provided by Covenant University through the Centre for Research, Innovation and Discovery (CUCRID).

REFERENCES

[1] Peter Mell, Timothy Grance, "The NIST Definition of Cloud Computing", NIST Special Publication 800-145, September 2011, pp. 7.
[2] Ahmed E. Youssef, "Exploring Cloud Computing Services and Applications", Journal of Emerging Trends in Computing and Information Sciences, VOL. 3, NO. 6, July 2012.
[3] Dana Petcu, Ciprian Craciun, "Towards A Cross Platform Cloud API Components for Cloud Federation",[Online].Available:https://pdfs.semanticscholar.org/2fad/eb44dcc5b4b73a7233fac4bac7c9caeee612.pdf. [Accessed: May 4, 2017].
[4] Henry Styles and Wayne Luk, "Customising Graphics Applications: Techniques and Programming Interface", Reprint from Proceedings of IEEE Symposium on Field-Programmable Custom Computing Machines, IEEE Computer Society Press, 2000, pp. 1-2.
[5] Qinghua Lu, Liming Zhu, Len Bass, Xiwei Xu, Zhanwen Li, Hiroshi Wada, "Cloud API Issues: an Empirical Study and Impact", QoSA'13, June 17–21, 2013 ACM 978-1-4503-2126-6/13/06.
[6] Ilja Livenson, Erwin Laure, "Towards Transparent Integration of Heterogeneous Cloud Storage Platforms", DIDC'11, June 8, 2011, ACM 978-1-4503-0704-8/11/06.
[7] Harold C. Lim, Shivnath Babu, Jeffrey S. Chase, Sujay S. Parekh, "Automated Control in Cloud Computing: Challenges and

Opportunities", ACDC'09, June 19, 2009, ACM 978-1-60558-585-7/09/06.

[8] Rohit Bhadauria, Rajdeep Borgohain, Abirlal Biswas, Sugata Sanyal, "Secure Authentication of Cloud Data Mining API". [Online]. Available: https://arxiv.org/abs/1308.0824 . [Accessed: May 24, 2017].

[9] Luís A. Bastião Silva, Carlos Costa, José Luís Olivera, "A Common API for Delivering Services Over Multi-Vendor Cloud Resources", The Journal of Systems and Software 86 (2013) 2309-2317.

[10] Diomidis Spinellis, "A Critique of the Windows Application Programming Interface", Computer Standards & Interfaces, 20:1–8, November 1998.

[11] Cleidson R. B. de Souza, David Redmiles, Li-Te Cheng, David Millen, John Patterson, "Sometimes You Need to See Through Walls — A Field Study of Application Programming Interfaces", CSCW'04, November 6–10, 2004, ACM 1-58113-810-5/04/0011.

[12] Douglas Kramer, "The Java Platform", JavaSoft, pp. 5-24, May 1996.

[13] E-Guide, "Moving to Cloud Applications and Integration Strategies". [Online]. Available: http://www.techtarget.com . [Accessed: May 20, 2017].

[14] Computer World, "Demystifying the cloud service API", 2015, Computer World Publication David Linthicum, "Don't migrate apps to the cloud without considering the costs". [Online]. Available: http://www.techtarget.com . [Accessed: May 26, 2017].

[15] D. Linthicum, "Cloud computing APIs pose vendor lock-in risks." [Online]. Available: http://searchcloudcomputing.techtarget.com/tip/Cloud-computing-APIs-pose-vendor-lock-in-risks. [Accessed: 04-Dec-2017].

[16] H. W. Van Der Westhuizen and G. P. Hancke, "Mobile Cloud Computing and Application Program Interfaces - a Review .," pp. 1623–1628, 2017.

[17] R. Ré, R. M. Meloca, D. N. Roma, M. A. da C. Ismael, and G. C. Silva, "An empirical study for evaluating the performance of multi-cloud APIs," Futur. Gener. Comput. Syst., vol. 79, pp. 726–738, 2018.

[18] J. Cito, P. Leitner, T. Fritz, and H. Gall, "The Making of Cloud Applications – An Empirical Study on Software Development for the Cloud," 2015.

[19] P. Musavi, B. Adams, and F. Khomh, "Experience Report : An Empirical Study of API Failures in OpenStack Cloud Environments.

[20] Baucke, S. *et al.* (2015) 'Cloud API support for self-service Virtual Network Function (VNF) deployment', in *2015 IEEE Conference on Network Function Virtualization and Software Defined Network (NFV-SDN)*. IEEE, pp. 40–46. doi: 10.1109/NFV-SDN.2015.7387404.

[21] Cretella, G. and Di Martino, B. (2012) 'Semantic Web Annotation and Representation of Cloud APIs', in *2012 Third International Conference on Emerging Intelligent Data and Web Technologies*. IEEE, pp. 31–37. doi: 10.1109/EIDWT.2012.61.

[22] Ferguson, H., Vardeman, C. and Nabrzyski, J. (2016) 'Linked data platform for building cloud-based smart applications and connecting API access points with data discovery techniques', in *2016 IEEE International Conference on Big Data (Big Data)*. IEEE, pp. 3016–3025. doi: 10.1109/BigData.2016.7840955.

[23] Hosseini, H., Xiao, B. and Poovendran, R. (2017) 'Deceiving Google's Cloud Video Intelligence API Built for Summarizing Videos', *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, 2017-July, pp. 1305–1309. doi: 10.1109/CVPRW.2017.171.

[24] Jinlong, E. *et al.* (2017) 'CoCloud: Enabling efficient cross-cloud file collaboration based on inefficient web APIs', in *IEEE INFOCOM 2017 - IEEE Conference on Computer Communications*. IEEE, pp. 1–9. doi: 10.1109/INFOCOM.2017.8056947.

[25] Lu, Q. *et al.* (2015) 'A Tail-Tolerant Cloud API Wrapper', *IEEE Software*, 32(1), pp. 76–82. doi: 10.1109/MS.2015.2.

[26] Mayoral, A. *et al.* (2017) 'Cascading of tenant SDN and cloud controllers for 5G network slicing using Transport API and Openstack API', in *OSA*, pp. 4–6.

[27] Satyal, S. *et al.* (2017) 'Rollback Mechanisms for Cloud Management APIs using AI planning', *IEEE Transactions on Dependable and Secure Computing*, pp. 1–1. doi: 10.1109/TDSC.2017.2729543.

[28] Wang, J., Bai, X., Li, L., *et al.* (2017) 'A Model-Based Framework for Cloud API Testing', in *2017 IEEE 41st Annual Computer Software and Applications Conference (COMPSAC)*. IEEE, pp. 60–65. doi: 10.1109/COMPSAC.2017.24.

[29] Wang, J., Bai, X., Ma, H., et al. (2017) 'Cloud API Testing', Proceedings - 10th IEEE International Conference on Software Testing, Verification and Validation Workshops, ICSTW 2017, pp. 385–386. doi: 10.1109/ICSTW.2017.71.

[30] Wua, M.-Y. and Lee, T.-H. (2013) 'Design and Implementation of Cloud API Access Control Based on OAuth', *Proceedings of the 2013 IEEE Tencon -Spring*, pp. 485–489. doi: 10.1109/CSCWD.2009.4968090.

[31] Xu, J. *et al.* (2017) 'Lightweight and Adaptive Service API Performance Monitoring in Highly Dynamic Cloud Environment', in *2017 IEEE International Conference on Services Computing (SCC)*. IEEE, pp. 35–43. doi: 10.1109/SCC.2017.80.

ISBN: 978-988-14047-9-4
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)