2022 23rd International Carpathian Control Conference (ICCC) | 978-1-6654-6636-3/22/$31.00 ©2022 IEEE | DOI: 10.1109/ICCC54292.2022.9805930

Proposed Methodology for Designing a Microservice Architecture

Mihajasoa Léa Fanomezana
Laboratory for Mathematical and Computer Applied to the Development Systems (LIMAD)
University of Fianarantsoa, Madagascar
fmihajasoalea@gmail.com

Andrianjaka Miary Rapatsalahy
Laboratory for Mathematical and Computer Applied to the Development Systems (LIMAD)
University of Fianarantsoa, Madagascar
andrianjaka92@yahoo.fr

Nicolas Raft Razafindrakoto
Laboratory of Multidisciplinary Applied Research (LRAM)
University of Antananarivo, Madagascar
rnraft@gmail.com

Costin Bădică
Faculty of Automation, Computers and Electronics (ACE)
University of Craiova, Romania
costin.badica@edu.ucv.ro

***Abstract*—The changing needs of the enterprise require the manageability of its information system. Therefore, it is crucial to absorb these changes by orienting the design of the information system towards a modern architecture that decomposes the monolithic application into autonomous services. This paper proposes an enterprise architecture methodological framework to design the so-called microservice architecture known as MSA. Our approach is to describe the similarities between the MSA style of architecture and the service-oriented architecture known as SOA, the latter having a rich research literature focused on exploiting the methodology for designing services in a service architecture. The result of the comparative study indicates that the ideology for designing a SOA is identical as MSA which ultimately relies on decomposing applications into smaller components with a smaller and more manageable footprint. Thus, we conclude that it is preferable to use the adapted enterprise architecture methodology for SOA to design MSA.**

Keywords—SOA, MSA, UML, Monolithic Architecture, MDA, software architecture design methodology, Praxeme

I. INTRODUCTION

Constant changes in business needs require a more flexible approach to the design and development of an information system. Managing enhancements, updates or other modifications to the system becomes very complex and time-consuming, especially for larger applications with strong coupling. In addition, practitioners as well as researchers have recently focused on exploiting cloud computing by migrating the entire system to the cloud [1], [2]. Therefore, the adoption of modern software architecture such as Service Oriented Architecture (SOA) [3] and Micro-Service Architecture (MSA) [4] is paramount. This paradigm is based on a decomposition of the monolithic application into smaller and autonomous elements called "services" [5]. In the early 2000s, the presence of SOA becomes more apparent with its goal of allowing flexibility for the design and creation of the information system that traditional monolithic approaches do not offer [6], while MSA being an improved version of SOA attracts the attention of the business world from the year 2014 [7].

Therefore, software architecture design needs a methodological framework whose role is to provide a guideline for the description of the enterprise. The notion of a such methodology was first glimpsed in 1987 by JA Zachman [8]. Over the last twenty years, several methodological frameworks have succeeded each other such as the Zachman framework, the TOGAF enterprise architecture methodology and the Federal Enterprise Architecture (FEA) framework. However, (Valantina and al., 2014) [9] argue that these approaches are not stable enough as they do not really take into account requirements management, maintenance and their process complexities. Researchers such as (Thierry and al., 2013), (Razafindramintsa and al., 2016) and (Rapatsalahy and al., 2021) then proposed the Praxeme methodology as an emerging enterprise architecture method [10]-[16]. The latter demonstrate that Praxeme is comprehensive, disciplined and very well suited for SOA [10]-[16].

Yet, research on the appropriate methodological framework for MSA is very rare. Indeed, this paper aims at proposing an appropriate enterprise architecture methodology with MSA. Our approach is to perform a comparative analysis on SOA and MSA in order to exploit their similarities since an enterprise methodology appropriate to SOA is widely discussed in recent published research works. We find that both architectures have great commonalities, especially on the concept of dividing the monolithic application into services. Thus, in this paper, we propose the Praxeme enterprise methodology to design a so-called microservices architecture. Praxeme is an enterprise methodology that consists in building the information system by decomposing it into logical services.

As for the structure of the plan of this article, section II overviews relevant works related to enterprise architecture methodologies. Subsequently, an overview of SOA and its relationship with Praxeme is presented in section III. The proposed approach is introduced in section IV before concluding and discussing future work in section V.

Authorized licensed use limited to: Anelis Plus consortium. Downloaded on July 05,2022 at 06:43:16 UTC from IEEE Xplore. Restrictions apply.

II. Related Work

Enterprise architecture is a method of tuning the information system to the organizational needs in order to achieve the business objective of the company. Thus, various methodologies for implementing enterprise architecture have followed one another over the years. Among them, we will discuss the most studied and relevant ones which are Zachman framework, Open Group Architecture Framework (TOGAF), Federal Enterprise Architecture Framework (FEAF) and Praxeme methodology.

(Zachman, 1987) [8] created a descriptive framework that defines the architecture of the information system as a result of its expansion and the complexity of its implementation. The approach is based on a neutral and objective framework. However, the methodology of strategic planning was not introduced. (Pereira & Sousa, 2004) proposed a method to facilitate the development of enterprise architecture based on the Zachman framework. The proposed approach allows for an efficient management of the information system and a better understanding of the architectural components [17]. The use of the Zachman framework to develop enterprise integration of business processes based on SOA has been presented [18]. In addition, the place of SOA in the Zachman framework has been demonstrated by (Barekat and al., 2013) [19] to make complex information systems more flexible and agile.

According to (Benkamoun and al., 2014) [20], Zachman framework seems to be the most studied and exploited by many companies and researches. It allows low-cost development based on the reuse of business models. Nevertheless, it has less integration and appropriate methodology and does not provide formal definitions and modeling language [21].

TOGAF was created by the U.S. Department of Defense as guidance for the evolution of its technical architecture, before the year 1990. Then the Open Group developed it starting with 1990, with its first version being released in 1995. In 2004, TOGAF 8 was launched and TOGAF 9 a more advanced version is planned in 2009 [22].

(Wahab and Arief, 2015) [23] assembled an integrated model of COBIT with the TOGAF framework for a comprehensive design of IT governance in local government and a resolution of the risk management control problem. The article by (Kabzeva and al., 2010) [24] states that in order to take advantage of the benefits of SOA such as application reusability and rapid adaptability to changing requirements, enterprises apply architecture frameworks. In addition, governance approaches have been used to overcome the challenges faced by SOA. The latter applied TOGAF for the design and governance of the architecture of a large-scale SOA-based project [24]. (Ni & Li, 2017) [25] also proposed the combination of TOGAF framework with SOA for agile evolution by adopting reusability and interoperability of applications.

TOGAF is a more detailed method and has a set of tools to support the development of a business process architecture and information system that can be freely used by any organization due to their good features [22], [23]. The TOGAF framework is composed of clear steps as presented in Figure 1. However, some limitations of TOGAF have been highlighted as a case in point the lack of information on the maintenance of the framework, the lack of integration between the different proposed artifacts, and finally the exclusion of strategic aspects [9], [21].

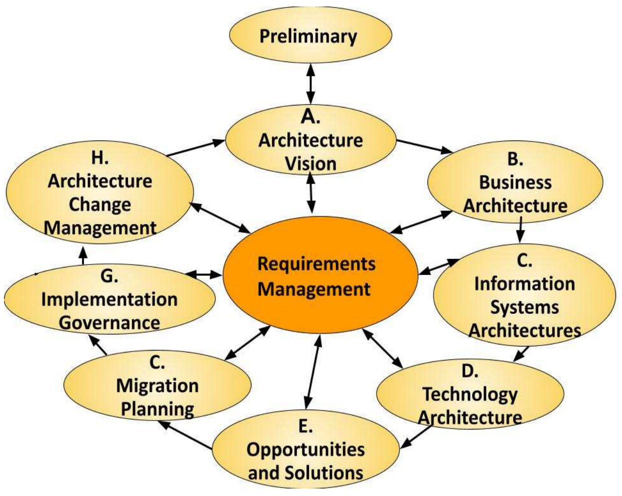
Preliminary
A. Architecture Vision
B. Business Architecture
C. Information Systems Architectures
D. Technology Architecture
E. Opportunities and Solutions
C. Migration Planning
G. Implementation Governance
H. Architecture Change Management
Requirements Management

Fig. 1. Architecture Development Method (ADM) Cycle

FEAF is an architecture framework that is designed to be used by the U.S. federal government. According to the authors of the article [26], it consists of simplifying and developing processes and information shared by the federal and government agencies as well as defining the planning of the organization's architecture.

In the paper of (Mahdavifar and al., 2012) [27], due to the failure of addressing the testing process at the level of an enterprise architecture project, a method was proposed for the improvement of the testing process in the federal architecture framework using the International Software Testing Qualification Board (ISTQB) framework. Then, (Defriani & Resmi, 2019) presented a study that aims to plan the e-government architecture in Purwakarta districts by developing the e-government. The approach adopted is the FEAF framework using the Collaborative Planning Methodology (CPM) [28]. On the other hand, an approach of combining Enterprise Architecture with SOA was proposed to better meet the agile needs of the enterprise. The authors opted for the use of FEAF. This combination produces a homogeneous framework named Service Oriented Enterprise Architecture (SOEA) to document the business and IT aspects of the organization [29].

The strength of the FEAF framework is its ability to describe and develop plans from current to future conditions with well-detailed planning steps. However, studies on FEAF and SOA are very rare whereas the SOEA approach has been successfully applied other enterprise architecture methodologies.

Praxeme is an enterprise architecture methodology that allows you to build your information system from a set of basic units called logical services. It is of French origin and comes from the combination of the Latin words "praxis" (action) and "semeion" (meaning) which translates to "the meaning of action" [30].

The research work of (Thierry and al., 2013) [10] evaluates the reliability and robustness of Praxeme in its model transformation. The latter demonstrates that the persistence of business rules and performance indicators

Authorized licensed use limited to: Anelis Plus consortium. Downloaded on July 05,2022 at 06:43:16 UTC from IEEE Xplore. Restrictions apply.

through this model transformation supports its robustness from the modeling design to the operational stage.

On the other hand, Praxeme does not offer a model that is representative of the intentional aspect of the business. Therefore, (Razafindramintsa and al., 2016) [15] proposes an approach to transform the eLEL (elaborated Language-Extended Lexicon) requirement model into a business model in the Praxeme methodology. The objective of (Razafindramintsa and al., 2016) [15] is to automatically derive the semantic aspect of Praxeme using the natural language oriented intentional model. On the other hand, researchers (Rapatsalahy and al., 2020, 2021) instantiated the ReLEL (Restructured elaborated Language-Extended Lexicon) requirement model in Praxeme for automating the software development process from the enterprise intention model [11]-[13].

Praxeme inherits and expands the methods proposed during the last thirty years including Zachman framework, Merise and other design methods by taking into account the combination of SOA with model-driven architecture (MDA). (Rapatsalahy and al., 2021) and (Roucairol and Caseau, 2011) [12], [31] confirm that Praxeme and SOA go well together for the design and development of information systems.

Praxeme is thus an enriched method, more advanced, capable of managing change and transformation of the company, and moreover, it is an open approach benefiting from more and freely available documentation [10]. Its advantage is that it captures all aspects of the company using a reduced set of conceptual artifacts. In addition, it suggests the use of UML for modeling each aspect of the enterprise, it uses the MDA approach to automate the transition from one aspect to another, and it adopts the SOA architecture to absorb the frequent changes in the enterprise requirements. Nevertheless, it does not provide a model for the representation of the enterprise intent [12], [13].

III. Overview of the Service-Oriented Architecture and Its Relationship with Praxeme

A. *Concept of SOA*

Before the year 2000, information systems were always designed and built according to the monolithic architecture where the software components are all combined into a single monolithic block. This approach creates a high interdependence between the features of the application and an increased complexity of the system as well as a more difficult or even impossible evolution of the system. This is where SOA comes in. It is a software architecture for the design and development of systems that consists in subdividing a large and complex application into several fundamental elements called "services". This approach has the following two objectives:

- Partitioning the application into several autonomous and reusable modular services to make it less complex and more flexible.
- Achieving interoperability between several systems running on different technologies.

SOA proposes the adoption of the Cloud environment for system development and deployment [2]. It is composed of two main functions which are the service provider and the service consumer. Thus, the service provider has the role of creating and publishing the service with a standardized description, following the requirements of the service consumer (requester). The latter searches the service in the service registry and then calls (invokes) it. The connection between the service requester and the service consumer is achieved through a mediator called ESB (Enterprise Service Bus) that ensures the weak coupling between Web services.

The best technology for implementing SOA conceptual services are Web services, either based on SOAP or Rest [12]. The basic technological standards of Web services in SOA are represented by:

- WSDL (Web Service Description Language), describing the remote interface of Web services.
- SOAP (Simple Object Access Protocol), transporting and exchanging messages between Web services.
- UDDI (Universal Description, Discovery and Integration), the registry for Web services discovery, useful for publishing and searching for specific Web services (Figure 2) [32].

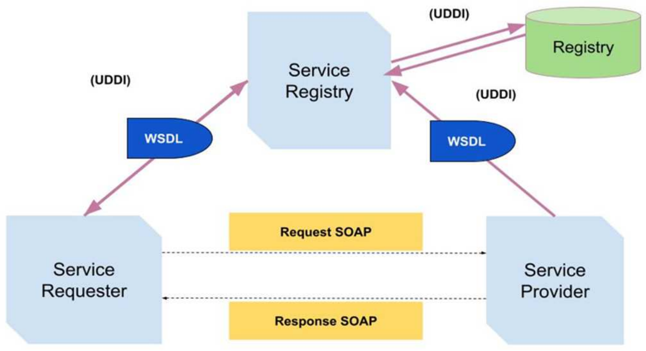

Fig. 2. SOAP Web Service concept

B. *Praxeme concept*

Faced with the complexity and rigidity of information systems due to the permanent evolution of organizational needs, the company decides to opt for a service-oriented architecture being able to guarantee the agility of its system. An enterprise architecture methodological framework for SOA design must therefore be highlighted in order to better take advantage of this approach. Indeed, among methodologies studied in this article, Praxeme provides the best support to be combined with SOA.

According to (Vauquier, 2010) [33], Praxeme is a framework that covers the modeling of all aspects of the enterprise, from strategy to deployment. It is represented by a topology of the enterprise system that has nine aspects, of which the generation of SOA services is done from the logical aspect. Praxeme methodology combines MDA and SOA approaches by organizing the information system around a basic unit called a logical service (logical aspect) [13]. In other words, Praxeme structures the system by breaking it down into several logical services. Derivation rules are applied to the semantic or pragmatic models to produce the SOA logical services. As shown in Figure 3, services are placed in the logical machine which is stored in the logical workshop, located itself in the logical factory [11]. The logical aspect is an intermediary element between the business view (semantic and pragmatic aspect) and the computing view (software aspect) which ensures the control of the complexity of the information system by separating the two aspects (Figure 4).

Authorized licensed use limited to: Anelis Plus consortium. Downloaded on July 05,2022 at 06:43:16 UTC from IEEE Xplore. Restrictions apply.

Moreover, according to (Rapatsalahy and al., 2021) [12], the Model Driven Architecture (MDA) allows the automation of the software development process using the model

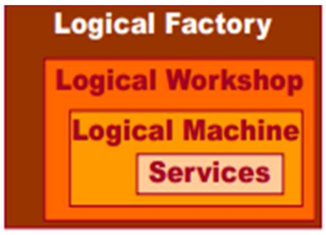

Fig. 3. Metaphorical terminology applied to the logical aspect [13]

transformation technique, i.e. model to model (M2M) and/or

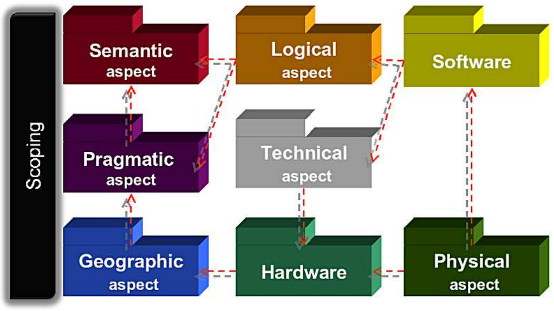

Fig. 4. The topology of the enterprise system [28]

model to text (M2T).

IV. Proposed Approach

This paper aims at suggesting an appropriate enterprise methodology to design microservices. Our approach consists in making a comparative analysis between SOA and MSA. Therefore, acquiring a good notion of MSA will allow us to assimilate both architectures.

MSA is a software design and development architecture based on smaller pieces called "microservices". According to (Fowler and al., 2014) [34], MSA represents a suite of small services developed and deployed independently. In other words, each microservice evolves and executes itself in its own process. Microservices communicate via an API and a very lightweight messaging protocol such as HTTP and REST. This approach is designed to enhance SOA by decomposing information system (IS) into smaller and simpler services. Thus, it is a new architectural style that provides the same goal as SOA, which is to simplify the application design by making it more flexible and scalable to the changing needs of the organization. From 2014, it has become a more powerful architecture on which many modern applications such as Netflix, Amazon and Soundcloud platforms are based [35].

Table I shows a comparative analysis between MSA and SOA.

TABLE I. Comparative Analysis of MSA and SOA

	SOA	MSA
Definition	Design and development architecture of an information system based on services in order to control the complexity of the IS and make it more flexible to changes.	Design and development architecture of an information system based on micro-services in order to control the complexity of the IS and to make it more flexible to changes.
Cloud-based	Yes	Yes
Sharing of resources between departments	Designed to support communication between two or more applications to share as much resources as possible [35].	Designed to share as few resources as possible to support the autonomy of microservices [35].
Remote access protocols	Uses remote protocols like SOAP [12]	Uses lightweight protocols like REST [5]
Communication	Services communicate via an ESB	Services communicate via an API
Granularity	Coarse grain service	Fine grain service
Scope or coverage	Suitable for large systems that are quite often composed of several application services, which are also composed of several infrastructure services [35] [5].	Suitable for small applications where each microservice is a small application contained within its own hexagonal architecture that has business logic and various adapters [36] [5].
Interoperability [5]	Each service can operate with any technology.	Each microservice can run with any technology
Governance	Common data governance mechanisms and standards for all services [37].	Decentralized governance [36].
Reusability	Services can be reused by other external or internal services in an integrated service infrastructure [37].	Microservices do not share source code, and rewriting the code is preferred rather than reusing it [2]
Data storage	All services have only one data storage [2].	Microservices do not share a database (each one manages its own data) [2].

Authorized licensed use limited to: Anelis Plus consortium. Downloaded on July 05,2022 at 06:43:16 UTC from IEEE Xplore. Restrictions apply.

We found common elements for both architectures (SOA and MSA):

- Same approach for IS design and development.
- Same goal for breaking down efficiently large and complex applications into smaller and more flexible elements to design and organize.
- All of services can use any programming languages.
- Both involve a cloud environment for application development and deployment.

Both architectures' analysis result indicates the conformity of SOA design ideology with MSA's one, by facilitating management of large and rigid applications in decomposing them into several smaller fundamental elements. In this article, we suggest using an enterprise architecture methodology adapted for SOA to design MSA, namely Praxeme methodology.

The logical model under Praxeme logical aspect is made up of logical factory, logical workshop, logical machine and services (Figure 3 and Figure 5) [13]. (Rapatsalahy and al., 2021) [12] have already suggested the automatic generation of SOA services within Praxeme. For this purpose, they have given rules for mapping Praxeme Logical Factory model to a WSDL model then rules for translating WSDL model into an XML-based WSDL file describing totally a Web service [12].

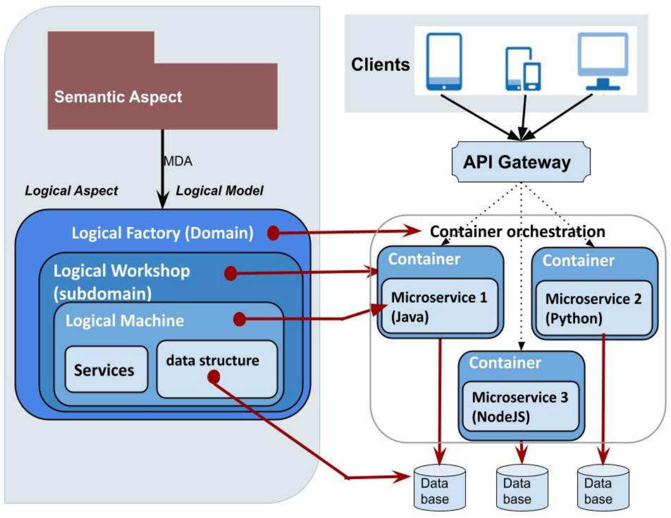

Fig.5. Illustration of the MSA generation within Praxeme

According to "microservice" word semantics, the idea is so to split application entities into as small as possible units (i.e. micro units). This paper therefore proposes to use logical machine containing microservices the smallest component of Praxeme logical model.

Nevertheless, the manual decomposition of the logical workshop containing the logical machine at the time of logical modeling proposed by (Rapatsalahy and al., 2021) [13] promotes a waste of time for software architects and a non-optimal information system design. Therefore, the approach we propose is to automate from the semantic aspect the design of the logical workshops of the Praxeme logic model. Indeed, the decomposition into subdomains must be positioned in the semantic modeling. According to our method shown in Figure 5, the orchestration container is modeled from the logical factory, then, the microservices container from the logical workshop and finally the microservices themselves are designed from the logical machine. In addition, we also suggest the attributes of the semantic model to become a component of the logical machine named "data structure" to model the database related to each designed microservice. The components of the MSA architecture such as the orchestration container, the microservice's container, the microservice and the database represent the software aspect of the Praxeme methodology (Figure 5).

V. Conclusion and Future Work

Faced with the complexity of IS management due to constant changes in requirements, companies need a modern software architecture. Thus, SOA has attracted the attention of practitioners and researchers in order to efficiently design IS. SOA consists in decomposing the monolithic application into several autonomous and reusable services [6]. Then, an improved variant of SOA called MSA, emerged [7]. It is more advanced software architecture with smaller and independent services [35]. Our study aims at showing a methodological framework for the design of microservices. According to the study of other relevant methodologies, we concluded that Praxeme is the enterprise architecture framework very well adapted to SOA aiming to construct IS by decomposing them into logical services [13]. On the other hand, studies on the methodological framework for designing MSA are very rare. Our approach is based on a comparative analysis of SOA and MSA and shows that the conception of both architectures is actually based on the same ideology which consists in the decomposition of the application into smaller elements. We thus propose Praxeme methodological framework to automate, improve as well as to simplify conception of MSA architecture. Indeed, our method consists in modeling microservices from logical machine contained in logical workshop which is automatically decomposed from the semantic aspect of Praxeme. In this paper, we equally suggest to model orchestration container from logical factory, microservice container from logical workshop and also corresponding database to each microservice from data structure.

Our recommendation is indeed more complete as it not only allows us to design MSA efficiently but also to model it up to database layer, and also more accurate reason being designing all of MSA components is explicitly described in Praxeme logic model. Let's notice we have just studied Praxeme as a methodology for designing MSA. And we plan to detail, practice and evaluate the proposed methodology in future work as to automatically develop microservice-based applications.

References

[1] O. Al-Debagy, and P. Martinek, "A Comparative Review of Microservices and Monolithic Architectures", in 2018 IEEE 18th International Symposium on Computational Intelligence and Informatics (CINTI), nov. 2018, pp. 000149-000154. doi: 10.1109/CINTI.2018.8928192.

[2] J. A. Bigheti, M. M. Fernandes, and E. P. Godoy, "Control as a service: a microservice approach to Industry 4.0", in 2019 II Workshop on Metrology for Industry 4.0 and IoT (MetroInd4. 0&IoT), 2019, pp. 438-443.

[3] K. B. Laskey, and K. Laskey, "Service oriented architecture", WIREs Computational Statistics, vol. 1, no 1, pp. 101-105, 2009, doi: 10.1002/wics.8.

[4] M. Grambow, L. Meusel, E. Wittern, and D. Bermbach, "Benchmarking microservice performance : a pattern-based approach", in Proceedings of the 35th Annual ACM Symposium on Applied

Authorized licensed use limited to: Anelis Plus consortium. Downloaded on July 05,2022 at 06:43:16 UTC from IEEE Xplore. Restrictions apply.

Computing, New York, NY, USA: Association for Computing Machinery, 2020, pp. 232-241. doi: 10.1145/3341105.3373875

[5] A. Andriyanto, R. Doss, and P. Yustianto, "Adopting SOA and Microservices for Inter-enterprise Architecture in SME Communities ", in 2019 International Conference on Electrical, Electronics and Information Engineering (ICEEIE), oct. 2019, vol. 6, pp. 282-287. doi: 10.1109/ICEEIE47180.2019.8981437.

[6] Z. Xiao, I. Wijegunaratne, and X. Qiang, "Reflections on SOA and Microservices", in 2016 4th International Conference on Enterprise Systems (ES), 2016, pp. 60-67.

[7] F. Rademacher, J. Sorgalla, S. Sachweh, and A. Zündorf, "Towards a Viewpoint-specific Metamodel for Model-driven Development of Microservice Architecture", arXiv preprint arXiv:1804.09948, 2018.

[8] J. A. Zachman, "A framework for information systems architecture", IBM Systems Journal, vol. 26, no 3, pp. 276-292, 1987, doi: 10.1147/sj.263.0276.

[9] G. M. Valantina, S. Jayashri, and K. D. Melmaruvathur, "A Framework for Evaluation Enterprise Architecture Implementation Methodologies", 2014.

[10] T. Biard, M. Bigand, and J. P. Bourey, "La méthode Praxeme : une nouvelle approche de l'Architecture d'Entreprise", 2013.

[11] A. M. Rapatsalahy, H. Razafimahatratra, T. Mahatody, M. Ilie, S. Ilie, and R. N. Raft, "Automatic generation of software components of the Praxeme methodology from ReLEL", in 2020 24th International Conference on System Theory, Control and Computing (ICSTCC), 2020, pp. 843-849.

[12] A. M. Rapatsalahy, R. Hajarisena, I. Mihaela, M. Thomas, I. Sorin, and R. N. Raft, "Automatic generation of Web service for the Praxeme software aspect from the ReLEL requirements model", Procedia Computer Science, vol. 184, pp. 791-796, janv. 2021, doi: 10.1016/j.procs.2021.03.098.

[13] A. M. Rapatsalahy, R. Hajarisena, I. Mihaela, M. Thomas, I. Sorin, and R. Raft, "Derivation of Logical Aspects in Praxeme from ReLEL Models":, in Proceedings of the 16th International Conference on Evaluation of Novel Approaches to Software Engineering, Online Streaming, --- Select a Country ---, 2021, pp. 413-420. doi: 10.5220/0010493004130420.

[14] A. M. Rapatsalahy, H. Razafimahatratra, T. Mahatody, M. Ilie, S. Ilie, and R. N. Razafindrakoto, "Automatic Generation of Object-Oriented Code from the ReLEL Requirements Model", SYSTEM THEORY, CONTROL AND COMPUTING JOURNAL, vol. 1, no 1, pp. 36-47, 2021.

[15] J. L. Razafindramintsa, R. J. Paul, T. Mahatody, and A. Becheru, "Semantic aspect derivation of the Praxème methodology from the elaborate lexicon extended language", in 2016 20th International Conference on System Theory, Control and Computing (ICSTCC), oct. 2016, pp. 842-847. doi: 10.1109/ICSTCC.2016.7790773.

[16] J. L. Razafindramintsa, T. Mahatody, J. P. Razafimandimby, and S. M. Simionescu, "Logical services automatic location from eLEL", in 2017 21st International Conference on System Theory, Control and Computing (ICSTCC), 2017, pp. 849-854.

[17] C. M. Pereira, and P. Sousa, "A method to define an Enterprise Architecture using the Zachman Framework", in Proceedings of the 2004 ACM symposium on Applied computing, 2004, pp. 1366-1371.

[18] J. Espadas, D. Romero, D. Concha, and A. Molina, "Using the zachman framework to achieve enterprise integration based-on business process driven modelling", in OTM Confederated International Conferences" On the Move to Meaningful Internet Systems", 2008, pp. 283-293.

[19] V. Barekat, E. B. Nejad, and S. E. Alavi, "Definition of zachman framework cells based on service oriented architecture", International Journal of Scientific and Research Publications, vol. 3, no 9, pp. 1-8, 2013.

[20] N. Benkamoun, W. ElMaraghy, A.-L. Huyet, and K. Kouiss, "Architecture framework for manufacturing system design", Procedia CIRP, vol. 17, pp. 88-93, 2014.

[21] H. Aqallal, "Architecture d'entreprise et système de justice civile au Canada". HEC Montréal, 2013.

[22] L. Sofyana, and A. R. Putera, "Business architecture planning with TOGAF framework", in Journal of Physics: Conference Series, 2019, vol. 1375, no 1, pp. 012056.

[23] I. H. A. Wahab, and A. Arief, "An integrative framework of COBIT and TOGAF for designing IT governance in local government", in 2015 2nd International Conference on Information Technology, Computer, and Electrical Engineering (ICITACEE), 2015, pp. 36-40.

[24] A. Kabzeva, M. Niemann, P. Müller, and R. Steinmetz, "Applying TOGAF to Define and Govern a Service-oriented Architecture in a Large-scale Research Project", in AMCIS, 2010, pp. 356.

[25] F. Ni, and R. Li, "TOGAF for Agile SOA Modelling", 2017.

[26] A. Akkasi, and F. Shams, "Presenting A Method for Benchmarking Application in the Enterprise Architecture Planning Process Based on Federal Enterprise Architecture Framework", in 2008 3rd International Conference on Information and Communication Technologies: From Theory to Applications, 2008, pp. 1-6.

[27] H. Mahdavifar, R. Nassiri, and A. Bagheri, "A method to improve test process in federal enterprise architecture framework using istqb framework", International Journal of Computer and Information Engineering, vol. 6, no 10, pp. 1199-1203, 2012.

[28] M. Defriani, and M. G. Resmi, "E-government architectural planning using federal enterprise architecture framework in Purwakarta districts government", in 2019 Fourth International Conference on Informatics and Computing (ICIC), 2019, pp. 1-9.

[29] M. K. Haki and M. W. Forte, "Service oriented enterprise architecture framework", in 2010 6th World Congress on Services, 2010, pp. 391-398.

[30] H. Fallon, and B. Vandenbulcke, "AgwA architecture office: study cases on structure and architecture", in ICSA2013-Second International Conference on Structures and Architecture, 2013, pp. 0-0.

[31] R. Gérard, and C. Yves, "Urbanisation, SOA et BPM : le point de vue du DSI", Dunod, Paris, 2011.

[32] N. Al-Rawahi, and Y. Baghdadi, "Approaches to identify and develop Web services as instance of SOA architecture", in Proceedings of ICSSSM '05. 2005 International Conference on Services Systems and Services Management, 2005., Chongquing, China, 2005, pp. 579-584 Vol. 1. doi: 10.1109/ICSSSM.2005.1499538.

[33] D. Vauquier, "Enterprise Methodology: An Approach to Multisystems", in Complex Systems Design & Management, M. Aiguier, F. Bretaudeau, et D. Krob, Éd. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 317-328. doi: 10.1007/978-3-642-15654-0_23.

[34] M. Fowler, and J. Lewis, "Microservices, 2014", URL: http://martinfowler. com/articles/microservices. html, vol. 1, no 1, pp. 1-1, 2014.

[35] H. Bloch, A. Fay, T. Knohl, and M. Hoernicke, "A microservice-based architecture approach for the automation of modular process plants", in 2017 22nd IEEE international conference on emerging technologies and factory automation (ETFA), 2017, pp. 1-8.

[36] P. Di Francesco, P. Lago, and I. Malavolta, "Architecting with microservices: A systematic mapping study", Journal of Systems and Software, vol. 150, pp. 77-97, 2019.

[37] M. K. Haki, and M. Wentl, "Service-oriented business-it alignment: a SOA governance model", 2010.

Authorized licensed use limited to: Anelis Plus consortium. Downloaded on July 05,2022 at 06:43:16 UTC from IEEE Xplore. Restrictions apply.