ISSN 2347 - 3983

Volume 8. No. 4, April 2020

International Journal of Emerging Trends in Engineering Research

Available Online at http://www.warse.org/IJETER/static/pdf/file/ijeter14842020.pdf

https://doi.org/10.30534/ijeter/2020/14842020

Microservices Architecture Design: Proposed for online HealthCare

Muhammad Rizki[1], Ahmad Nurul Fajar [2], Astari Retnowardhani [3]

[1,2,3]Information Systems Management Department, BINUS Graduate Program-Master of Information Systems Management, Bina Nusantara University, Jakarta, Indonesia 11480.

[1]muhammad.rizki004@binus.ac.id; [2]afajar@binus.edu; [3]aretnowardhani@binus.edu

ABSTRACT

This study about the analysis and design of SOA-based applications using a microservices approach based on a case study of PT XYZ. The transformation of the system from Model View Controller (MVC) with a large codebase evolved into SOA through the microservices approach, this is an important part in order to promote independence, speed and security. according to the conditions of the company through steps that are comprehensive and effective. adaptation of microservices architecture is very important in terms of development life cycle culture in a company and these changes must be done with a sustainable approach in order to achieve the target when applying microservices includes flexibility, speed, and security.

Key words : Microservices, SOA, Domain-driven design, Online HealthCare.

1. INTRODUCTION

Information technology helps many people to get information more quickly and accurately including information about health solutions and online health care is one of the services in the health industry that is run with online information technology solutions that can reach the wider community effectively and efficiently. The opportunity was accepted by PT XYZ to provide accurate and trustworthy information to the public about good health about various common diseases, symptoms and healthy lifestyles, directly from health practitioners sourced from several doctors providing their knowledge about a case or tips for prevention. The current architecture at PT XYZ is an architecture based on client-server architecture and a typical view controller model (MVC).

MSA (microservices architecture) is an SOA-based architecture with an approach through a smaller context and focuses on create an independent application as a separate service and each service can communicate in simple and lightweight, such as the Http protocol. Integration and building microservices mechanisms are necessary for the management of various existing systems and future systems. According to increased business growth, a company is demanded to increase productivity in all production lines by increasing existing services, reducing business risk and simplifying existing processes, it requires effective transformation and innovation to remain competitive in the health care industry, information technology has improved healthcare to the patient at anytime and anywhere to provide emergency health services and telemedicine provides timely and costless health services in the society [1]

Business development requires the system to change accordingly with desired conditions and delivered quickly but remains operationally safe, this is difficult to implement in a monolith system, given the many things that need to be considered to change a function with a large and complex codebase, functional dependencies are tight coupling instead of loose coupling and need a few steps to ensure the transformation, including coordination between departments or teams involved and re-testing that changes that have been made, should not interfere with the existing functionality.

The objectives of this study are analysis of existing business flow and design SOA-based application through the microservices architecture (MSA) approach using domain-driven design.

The benefit of this research is to help companies when planning to transform monolithic architecture systems into microservices architecture (MSA), providing concrete steps that can be carried out for the comprehensive implementation of microservices architecture (MSA). The scope of this research is to analyze and design a prototype of an SOA-based application module through the microservices architecture (MSA) approach to pre-existing business processes using domain-design driven to describe the results of the transformation of monoliths to microservices.

2. LITERATURE REVIEW

Service-oriented architecture (SOA) is a service-oriented approach. By definition, SOA is different from web services, which means that SOA is not the same as web services, the

SOA approach can cope with a dynamic and rapidly changing business environment with the flexibility in mind. This condition can minimize the effort, cost and time to adjust functionality when requirement changes.
SOA can produce architectures that are loosely coupled and stateless, this is because the system architecture is built with service-oriented, some SOA principles described by [2], and these also form the basis of microservices are as follows: reusable, share a formal contract, loosely coupled, abstract underlying logic, composable, autonomous, stateless, and discoverable.

A common architecture in SOA-based application development described by [3] are Reuse, Efficient Development, Integration of Applications and Data, and Agility, Flexibility, and Alignment.

Microservice architecture is a collection of independent and relatively small processes that can communicate with each other to form complex applications and agnostic to any programming language. These services consist of small blocks, separated and focus on light tasks to run modular and distributed methods in the system. Each module of each microservices can be implemented and operated as a small but independent system, this is possible to access internal logic and data through a predetermined network. it can improve the ability of the system because each microservices become an independent unit to carry out development, deployment, operation, versioning, and scaling [4].

Microservices are a subtype of SOA with independent services with clear boundaries, microservices are similar to traditional SOA,. But SOA tends to rely heavily on several ESB (enterprise service bus) products or similar products, whereas microservices only rely on lightweight technologies such as the Http protocol with a REST interface. Some advantages that are often attached to microservices architecture (MSA) include faster delivery, improved scalability, and greater autonomy. microservices are packaged and deployed through cloud computing using container technology such as Docker, following the practice in the DevOps industry that has been proven through the support of CI / CD (Continuous Integration, Continuous Delivery) can be fully automated for integration and deployment [8]. The difference in scaling between monolithic and microservices can be seen in the difference in monolithic capabilities which can only be improved by adding the whole system including its host with some hardware and software, whereas microservices is more focused by adding services needed by increasing horizontal scaling while remaining attached on the same hosts [5].

Previous research related to microservices architecture as follows:

i. Pooyan Jamshidi [4] *Microservices: The Journey So Far and Challenges Ahead*. This study about the evolution of microservices, advantages, and disadvantages as well as improvements for the future. The evolution of microservices made by researchers, wherefrom the research model can be seen that the evolution of microservices technology is developing very fast with the emergence of various tools to assist in the implementation and maintenance of modules that are managed and create with microservices oriented. the results of this study provide an important picture of the transformation from a monolithic system to distributed and independent microservices also as consideration to make decisions about microservices architecture.
ii. Christudas [6] *Microservices Architecture*. Researchers from The Open Group, SOA Work Group, develops detailed research on the perspective of microservices with members of industry working groups. by providing clear and specific definitions of microservices, filtering out the core principles of microservices and key characteristics, and providing a comparison of MSA with Service-or-embedded-architecture (SOA).
iii. Andi Singleton [7] *The Economics of Microservices*. Researchers conduct studies of microservices in terms of economics and budget as well as the benefits and trade-offs that can be obtained in the implementation of microservices, related to this can also provide a simple way to change components without causing or adding new problems throughout the system.
iv. O'Connor, Elger, & Clarke [8] *Exploring the impact of situational context - A case study of a software development process for a microservices architecture*. Researchers conduct research on the importance of context in a decision-making process using microservices. By understanding the life cycle model must be appropriate for the scope of the project, the magnitude of the impact, complexity, changing needs and opportunities such as the general domain of situational factors that influence the development process of microservices can be seen as strategically important for the future of software development, classification into several things such as Personnel, Requirements, Application, Technology, Organization, Operations, Management, and Business.
v. Eric Evans [9] *Tackling Complexity in the Heart of Business Software,* Domain design-driven is a philosophy whose focus is on the intricacies of domains and make these intricacies explained in the domain model and its implementation in code.

3. ANALYSIS AND PLANNING

This research begins when a company has migration plans to convert the system from monolithic to MSA (microservices architecture), SOA-based application design with MSA approach is carried out using frameworks, design principles and methodologies related to domain-driven design (DDD) adapted from [9] and [10] domain-driven design (DDD) is an approach used in application development where the domain model is the center of artifacts. Domain-based design approach was used in application development where the central domain model is an artifact with creative collaboration between technical experts and domains to iteratively improve the conceptual model that addresses specific domain problems. The stages that will be carried out to identify and design the use of the Domain-Driven Design approach of [9] and [10] in modeling microservices-based architecture on research objects.

Collecting data is needed at the beginning of the preparation of this stage, such as by conducting visits and interviews with relevant parties at PT. XYZ interviews were conducted with several teams involved in developing online health care applications including doctor chat modules, doctor bookings, hospital search, insurance, and health information. This interview aims to get an overview of how the application architecture and business capabilities of existing applications and what obstacles are encountered and get documentation from the system. Next, analyze the architectural models that are in productions and study the documentation at PT. XYZ, such as functional specification document, technical specification document, and user manual, user journey from the current online health care application system with the aim of the results obtained will become a reference and guide the existing system specifications for the process of developing and modeling microservices. At this stage, identify and classify the domain model with the Domain-Driven Design approach where the steps are as follows:

1) Analyze business domains by understanding the functional flow of application processes. The output of this step is an informal description of the domain, which can be made a reference and refined to a more formal set of domain models,
2) Define the domain. bounded context represents a particular subdomain of a larger application that contains a domain model,
3) Apply tactical DDD patterns to define entities, aggregates, and bounded contexts services domains, and
4) Use the results from the previous step to identify microservices.

The application architecture design phase determines how the communication design between services, API Design, Data Management and design patterns that will be applied to the online health care system at PT. XYZ

3.1 Infrastructure Analysis

The infrastructure using third-party services to maintain services in optimal conditions, such as when users access applications or websites will be directed to Cloudflare as Content Delivery Network (CDN) that can protect against threats such as SQL injection and identity theft. Cloudflare also improves site performance and speeds up loading time by using several data centers located throughout the world and then be directed to the Google Cloud Platform (GCP) for proxies, load balancers, virtual machines, and databases. To increase the resilience of the monolith architecture, load balancers are used to refer to the process of distributing tasks over a series of resources, to make the overall process more efficient. when disruption to network functionality and denial of service attacks can have a major impact on healthcare delivery and protecting data availability, ensuring consistent connectivity and access to services is major requirement [11].

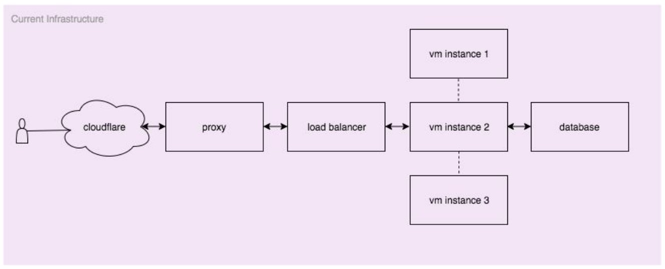

Figure 1: Current Infrastructure

As shown Figure 1, a load balancer distributes user traffic on several server instances, by working on load distribution, load balancers can reduce the risk of applications being overhead, slow, or not functioning.

3.2 Business Domain Analysis

The object of research is PT. XYZ is a company engaged in online healthcare services, the services offered are in the form of health care solutions developed to meet the needs of the public in finding, obtaining information and health services, consisting of several modules such as Figure 2.

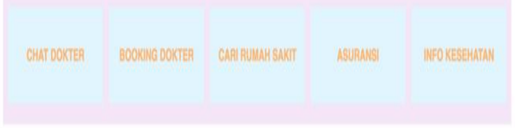

Figure 2: Business Model

Those modules packaged as monolithic application describes a single-tiered software application in different components combined into a single program from a single platform as shown in Figure 3.

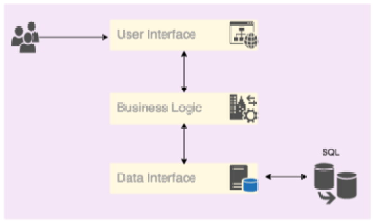

Figure 3: Monolithic Package

Information Workflow is a sequence of tasks that process a set of data, workflows occurring in each type of product and domain. information can be sent at any time by the user or the system. The workflow Information is a flow that describes how something changes from the first step to desired information that can be processed and received. several workflows occur as follows, such as: Chat Workflow, Booking Workflow, Hospital Workflow, Insurances Workflow and Magazine Workflow.

Chat Workflow is a process of transferring information and functional domains in the Chat with Doctors product, which starts with user registration in the application and then user can search for the specialist doctors, if they have found a suitable doctor to consult, user can pay a nominal amount to start chatting, after the payment transaction is verified, user can immediately consult and ask questions with a doctor as Figure 4.

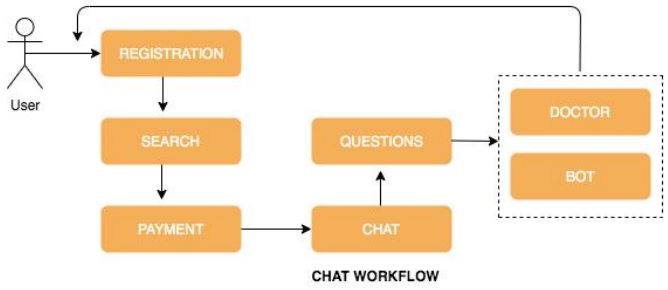

Figure 4: Chat Workflow

Workflow Booking is a process for reserving queues in conducting offline consultations in accordance with the doctor's schedule given by the hospital as Figure 5.

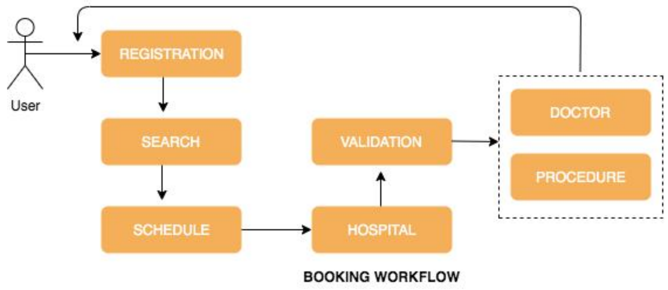

Figure 5: Booking Workflow

Hospital Workflow is a process of finding information about doctors, procedures and hospitals, according to the keywords that the user wants, this keyword goes to keyword validation process based on data available in the application as Figure 6.

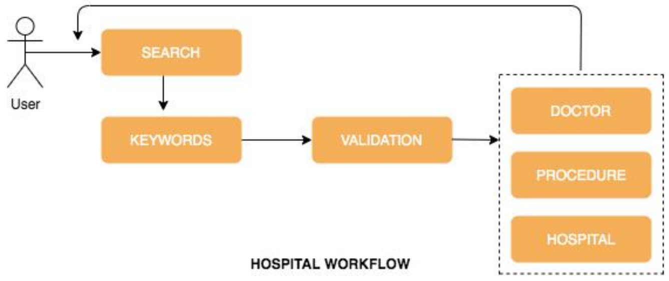

Figure 6: Hospital Workflow

Insurance Workflow is the process from insurance products, such as registering users as insurance participants through the application, after submission there will be data verification followed by validation of payments to be deducted monthly during the insurance policy period, as long as the user's in active status, they can get benefits in accordance with the agreement stated in the insurance policy, users can submit claims, then the claim will be validated and adjusted according to the policy as Figure 7.

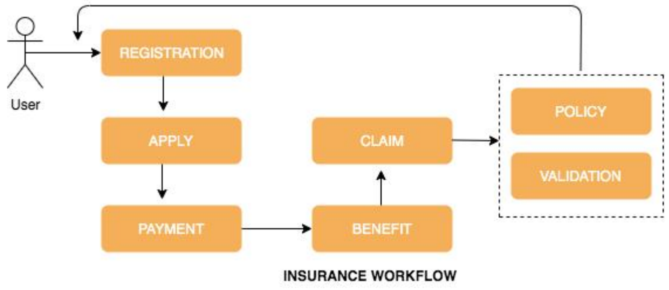

Figure 7: Insurance Workflow

Magazine Workflow is a process of transferring information about health articles, information related to disease and drug names, users can directly search for information based on the desired topic in the form of health topics or users can ask about health conditions through Ask the doctor's page and there are various names of diseases and drugs accompanied by an in-depth explanation that can be an important reference about medical terms that are often used in health sciences as Figure 8.

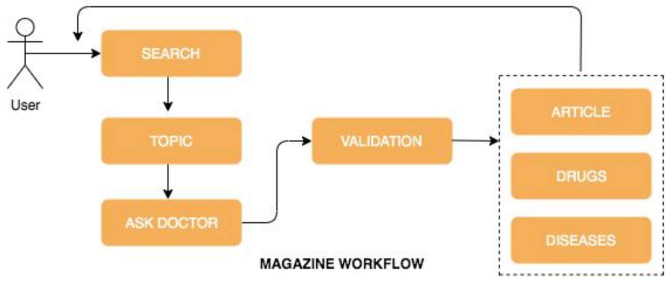

Figure 8: Magazine Workflow

3.3 Bounded Context Analysis

Bounded Context is a semantic contextual boundary and components inside a Bounded Context are context specific, each component of the software model has a specific meaning and does specific things [12].

A bounded context is the boundary within a domain where a particular domain model applies and well-defined, area of responsibility in the business domain with its own unique vocabulary, why using *Bounded Contexts*? because often teams do not know when to stop piling more and more concepts into their domain models.

The model may start small and, but then more concepts, and more, and adding more. This soon results in a big problem. Not only are there too many concepts, but the language of the model becomes blurred, because there are actually multiple languages in one large, confusing, unbounded model [12] as shown in Figure 9.

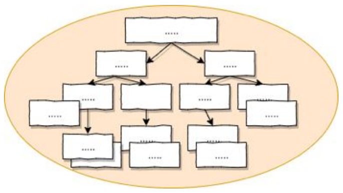

Figure 9: Overflow Context

Core Domain Context is the core services provided by PT XYZ, as Customer Perceived Value is one of the most influential forces in the market. Value, in marketing, is usually described from the customer's perspective, which defines performance, quality and price [13] and premium chat with doctors becomes the main service of all products offered to the community. the schedule of doctors from early morning to night and various options for specialist doctors who are capable answering user complaints according their experiences and a strong medical background, then chat services with doctors grows into other complementary services.

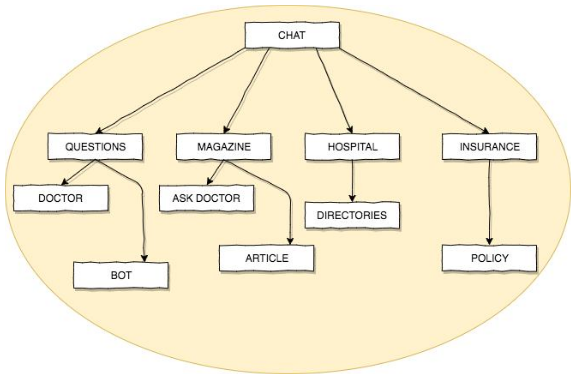

Figure 10: Core Domain

As shown in Figure 10, it starts with chat services that can be assisted by doctors or chatbots, then grows into a number of other services, such as health articles, hospitals and insurance.

Sub Domain Account Context, this is related to transactions made by users, including payment of premium chat with doctors as shown in Figure 11.

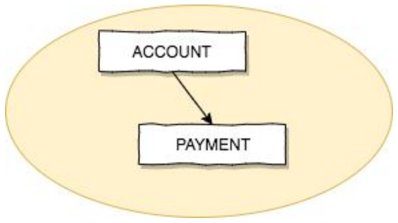

Figure 11: Account Domain

Payment can be integrated with various modules related to transactions, such as validation for repayment of insurance members.

Sub Domain Content Context, these health articles, including information about doctors and hospitals as collaborator, a collection of medical terms such as diseases and drug names, that can be a reference material in analyzing health conditions as shown in Figure 12.

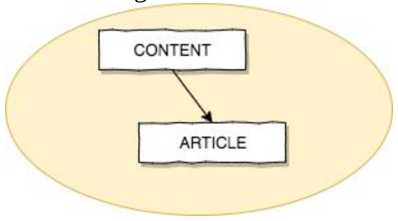

Figure 12: Content Domain

Sub Domain Insurance Context, this is part of an insurance product that handles insurance life cycles, start from registration to claims such as Figure 13.

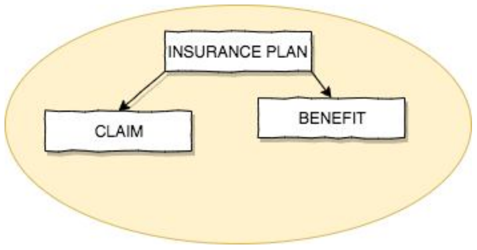

Figure 13: Insurance Domain

Sub Domain Member Context is part of the core domain that regulates membership, users or doctors registered in the service and roles that can be set in for flexibility in the system such as Figure 14.

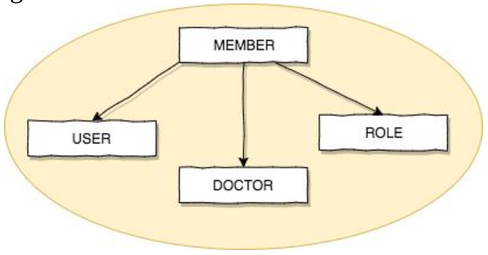

Figure 14: Member Context

Sub Domain Schedule Context is part of the doctor's schedule and integrated with the hospital booking and for premium chat with doctors in applications such as Figure 15.

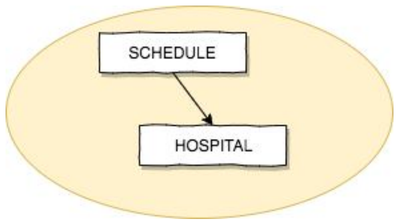

Figure 15: Schedule Context

There are 1 Core Domains and 5 Sub Domains as the basis for bounded context in the online health care application at PT XYZ, includes chat, booking, hospital, article and insurance services.

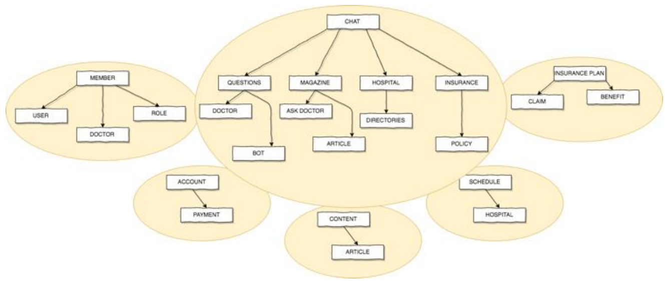

Figure 16: Domain Contexts

A collection of multiple context domains as shown in Figure 16 and Context Mapping Integration using by RESTful http.

3.4 Aggregates Analysis

Aggregate is a logical boundary for things that can change in a business transaction of a given context and defining consistency boundaries and enforcing invariants as shown in Figure 17.

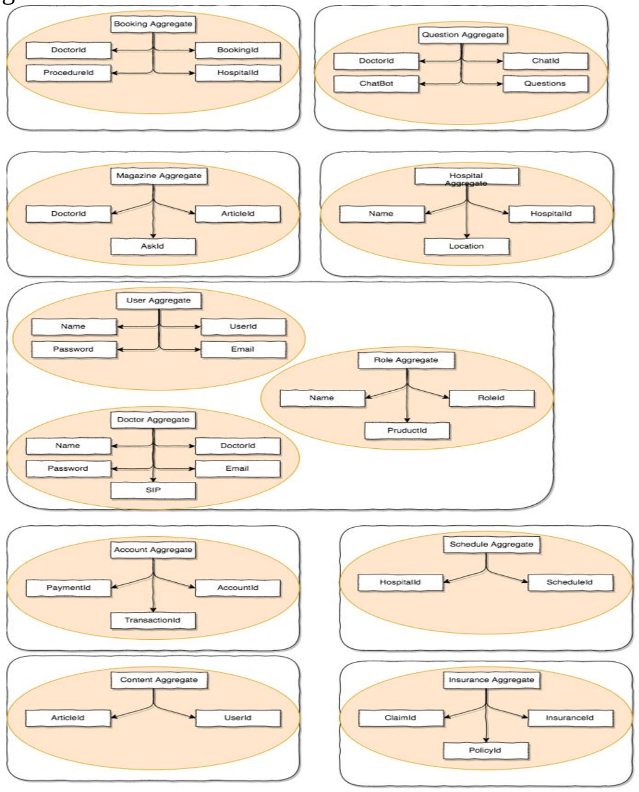

Figure 17: Domain Aggregates

3.5 Microservices Design

Microservices provide more agility on the software development life cycle than the traditional Monolith, coordination between team are less coupling because microservices are independent and small.

Based on aggregates analysis, identifying services candidate is clearer than before, as shown in Figure 18, there are 8 services candidates will be packaged as microservices application as shown in Figure 18.

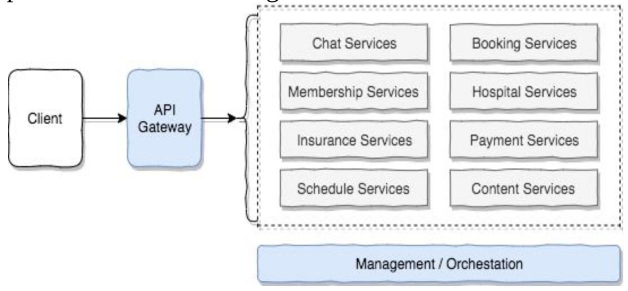

Figure 18: Microservices Architecture

Services can be deployed independently, responsible for persisting their own data or external state and communicate with each other by using well-defined APIs. Internal implementation each service only accessible via API gateway and that the only one entry point for clients, API Gateway forward the call to destination services. In order to manage services nodes, failures, rebalancing Management/Orchestration is included.

Services communicate through APIs. Consider the case where the Chat service requests information about doctor schedules from the Schedule service as shown in Figure 19.

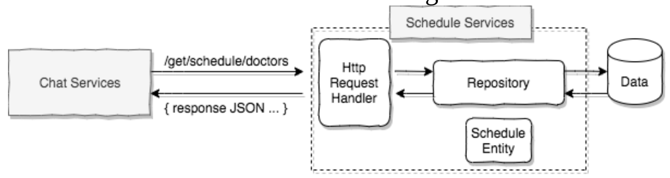

Figure 19: Services Communication

In microservices architecture (MSA) client services might interact with one or two services, API gateway sits between clients and services also as reverse proxy, handle routing from clients to services. It may also handle various cross-cutting tasks such as authentication, SSL termination, and rate limiting as shown in Figure 20.

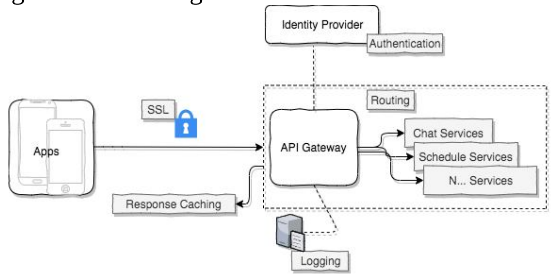

Figure 20: API Gateway Architecture

Each service packaged using container technology such as Docker in order to make application small as deployable units.

4. SUMMARY

Microservices is a single application of small services, each service running its own process and uses lightweight mechanism for communicate, each service does a specific business goal and communicate with other sets of services using a simple and well-defined interface. development effort around multiple teams is more efficient because the team is responsible only for one or more service.

The application starts faster, which makes developers more productive, and speeds up deployments and utilizing Domain-design driven by analyzing the business domain to understand the application's functional requirements Tactical DDD provides a set of design patterns that can use to create the domain model. These tactical patterns such as entities, aggregates, and domain services will help to design microservices that are both loosely coupled and cohesive such as bounded context, apply Tactical DDD patterns to define entities, aggregates, and domain services. As an application evolves, there are probabilities to break apart a service into several smaller services.

5. CONCLUSION

A case study for online health care as references for organizations who plan to convert their big and monolith system into flexible as deployable units that can scale as business grows without downgrade the quality of existing functional domain and package the domain as microservices architecture with Domain-design driven approach.

ACKNOWLEDGEMENT

I would first like to thank PT XYZ and the CTO, who made this research possible. I would also like to thank my supervisor Dr. Ahmad Nurul Fajar, S.T., M.T and Astari Retnowardhani, S.kom, M.Kom, Phd , for help, guidance and patience. The criticisms and contributions were concrete and helped in the formulation and completion of this research.

REFERENCES

1. G. Gregory Mihuba, M. Joseph Shundi, B. Obadia Kyetuza, M. Anthony Msafiri, **Design of Telemedicine System Based on Mobile Terminal**, *International Journal of Emerging Trends in Engineering Research*, Vol. 7, no. 1, pp. 5-6, Jan 2019. https://doi.org/10.14445/23488549/IJECE-V6I3P101
2. Erl, T. (2009). **SOA Design Patterns**. *In Elements*. https://doi.org/10.1016/j.artmed.2009.05.004. https://doi.org/10.1016/j.artmed.2009.05.004
3. Rosen, M. **Applied SOA: Service-Oriented Architecture and Design Strategies**, *U.K.: Wiley*, 2008.
4. Jamshidi, P., Pahl, C., Mendonca, N. C., Lewis, J., & Tilkov, S. (2018). **Microservices: The journey so far and challenges ahead.** *IEEE Software*. https://doi.org/10.1109/MS.2018.2141039
5. Dragoni, N., Lanese, I., Larsen, S. T., Mazzara, M., Mustafin, R., & Safina, L. (2018). **Microservices: How to make your application scale**. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics).* https://doi.org/10.1007/978-3-319 74313-4_8
6. Christudas, B., & Christudas, B. (2019). **Microservices Architecture**. *In Practical Microservices Architectural Patterns.* https://doi.org/10.1007/978-1-4842-4501-9_4
7. Singleton, A. (2016). **The Economics of Microservices**. *IEEE Cloud Computing.* https://doi.org/10.1109/MCC.2016.109
8. O'Connor, R. V., Elger, P., & Clarke, P. M. (2016). **Exploring the impact of situational context - A case study of a software development process for a microservices architecture**. *Proceedings - International Conference on Software and System Process*, ICSSP 2016. https://doi.org/10.1145/2904354.2904368
9. Evans, E. (2002). **Tackling Complexity in the Heart of Business Software**. *Pattern Languages of Program.*
10. Steinegger, R. H., Giessler, P., Hippchen, B., & Abeck, S. (2017). **Overview of a Domain-Driven Design Approach to Build Microservice-Based Applications**. *The Third International Conference on Advances and Trends in Software Engineering Tivities*.
11. D. J. Joel Daniel, S. J. Ebeneze, **A Survey on Security Issues in IoT**, *International Journal of Emerging Trends in Engineering Research*, Vol. 7, no. 12, pp. 2-3, Dec 2019.
12. Vernon, V. **Domain-driven Design Distilled,** *Addison-Wesley*, 2016
13. E. Mehdi, **Investigating the Role of Green Perceived Value on Customer Loyalty with the Mediating Role of Green Brand Preference**, *International Journal of Emerging Trends in Engineering Research*, Vol. 7, no. 10, pp. 6-7, Oct 2019.