Verifying the effectiveness in reducing development period on serverless architecture

－　Can business agility be improved　－

IA-001: Applying Serverless Architecture to Business：Daisuke Ishino

While new digital technologies are applied every day, companies are required to create business models quickly by utilizing these new technologies to enhance their competitiveness. In recent years, serverless architecture has been attracting attention as an architecture to improve business agility. In this paper, we examine whether business agility can be improved by applying a serverless architecture. This paper discusses the pros and cons of the improvement of business agility and future issues and prospects based on the results of the verification．

Digital Transformation, Serverless Architecture, Improvement of business agility, Reducing development period, Reducing the amount of documentation

**１．Introduction**

In recent years, new digital technologies are being developed day by day, and new businesses that make full use of digital technologies have been created not only in the IT industry but also in various industries to increase corporate value. In this situation, companies are required to speed up their digital transformation(DX) in order to maintain and strengthen their competitiveness. According to the "DX Digital Transformation Report" by the Ministry of Economy, Trade and Industry [1], many executives in Japan are aware of the necessity of DX and are making efforts such as establishing a digital department to promote DX. However, many companies have invested to some extent, such as repeating PoC, but it has not led to actual business transformation. Data cannot be fully utilized as the existing systems become older, more complex and black boxed. Therefore, even if new digital technologies are introduced, data utilization and linkages are limited, and Its effect would also be limited. It is estimated that up to 12 trillion yen in economic losses will occur between 2025 and 2030 if existing aging and complicated systems continue to be used as they are.

Unless the problems surrounding existing IT systems are resolved, full-scale deployment of DX will be difficult. As the renovation of existing IT systems is a large-scale and long-term project, it is not possible to obtain the effect quickly. Business logic that must be frequently modified or newly created needs to be built using new digital technologies. By doing so, they can quickly follow the changes in the business model.

In recent years, serverless architecture has been attracting attention as an architecture to improve business agility. A key feature of serverless architecture is that it allows cloud service providers to take on more responsibility and allows users to focus on building business logic and developing applications. In this paper, we examine whether the application of a serverless architecture improves business agility by designing and building an actual model system. The improvement of business agility in this paper means to shorten the period of development until the release of the service, and in conclusion the service can be provided more quickly. Chapter 2 describes the characteristics of serverless architecture, and Chapter 3 and 4 describe the results of the verification of the serverless architecture in terms of the amount of documentation and construction time. In addition, the future prospects are described based on the issues and problems identified in the verification process in chapters 5 and 6.

**２．What is a serverless architecture**

　In this paper, the two terms "serverless" and "serverless architecture" are defined and agreed with the definitions in the 2019 JGS paper "Proposal for a Process to Study the Application of Serverless Architecture".Serverless refers to the concept of building and executing applications that do not require server management, and there are two typical ways to realize it: Function as a Service (FaaS) and Backend as a Service (BaaS). The system configuration that combines these FaaS and BaaS features is a serverless architecture [2].

**2.1 Serverless Features**

　Although there are various characteristics of the serverless system for each cloud service providers, the following two features are described here.

(1)No need to build and operate a server

As shown in Figure 1, the infrastructure required

to run applications, such as servers and runtimes, is operated and managed under the responsibility of cloud service providers. Therefore, users do not need to build a server or installing middleware, and the workload for operation and management can be significantly reduced.



Figure 1. Serverless Responsibility Model

(2) Flexible scalability

Since automatic scaling is possible according to the processing volume, there is no need to allocate excessive resources in case of high load. In addition, the system can flexibly respond to unexpected loads and perform stable operations.

**３．Verification of Business Agility Improvements by Applying a Serverless Architecture**

　In this chapter, we built a website for purchasing goods using a serverless architecture, and compare it with the construction work in an on-premises environment, and describe the result of verifying whether the development period will be reduced.

The scope of verification is the part of system development that is developed by the IT department using a serverless architecture based on the business strategy determined by the management.

The reduction of the development period is verified from the following two points of view.

・Amount of documentation

・Construction time

**3.1 Hypotheses and Premises**

Hypotheses and Premises in the verification process are described below.

**3.1.1 Hypotheses**

We hypothesize that the development period will be reduced by decreasing the amount of documents in each stage of the system development process and reducing the construction time by applying a serverless architecture.

**3.1.2 Premises**

Premises for the verification process are described below.

1. Cloud service providers

In this verification, we built a website for purchasing goods using a serverless service provided by Amazon Web Services (AWS), a leading cloud service provider. [3]

1. Reasons for choosing the model system

Japanese government is promoting cashless payments. So we believe that the number of e-commerce site launches will increase. "The Current Status and Significance of Cashless Payment by the Ministry of Economy, Trade and Industry" [4] describes that the advantages of promoting cashless payment are "increased efficiency and sales at stores" and "data utilization". When a retailer launches an e-commerce site, it can expect to increase sales by diversifying its sales channels. Also, it is possible to link to marketing and product development by analyzing and utilizing the data of purchase information on the site.

1. Building a goods purchasing website

The requirements for the goods purchasing website are summarized in the requirements specification (Appendix A-1), and we constructed a system that satisfies the requirements. The serverless architecture to be applied is based on the architecture design published in AWS samples, which is quality assured by AWS. We reused Serverless Airline Booking (an example of an airline ticket purchasing system) [5], which has similar to our requirements, as the reference architecture. And we designed a system that meets our requirements. (Figure 2)

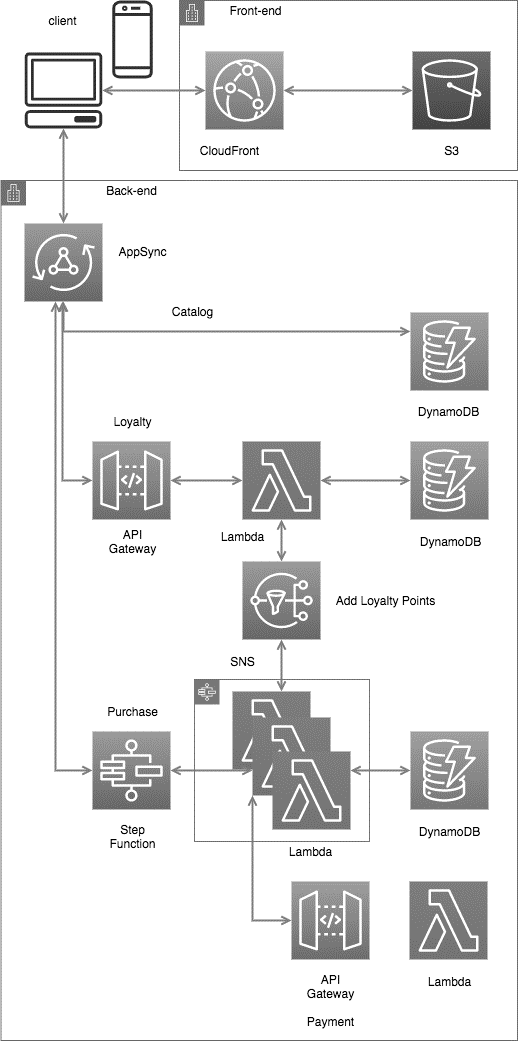


Figure 2. System Configuration diagram of

the goods purchasing website

**3.2 Verification Method**

The measurement of construction time and Amount of documentation was performed by the following procedures.

**3.2.1 Construction time**

The system build work was carried out in the following procedures from A-1 to A-4. The construction time was measured as the time required for steps A-3 and A-4, which are the construction work itself. The building process was assigned to one member who had no experience with AWS development.

Step A-1: Based on our requirements specification, we referred to the AWS Developer Guide to learn about the design and configuration required for the selected AWS service in advance.

Step A-2: Summarize the data used in the system as a data definition document (Appendix B-1). Summarize the values set in the AWS Management Console as a parameter sheet (Appendix B-2) for each function to be implemented in the goods purchasing website.

Step A-3: Develop the application source code for a goods purchasing website using the runtime supported by Lambda. Code development is divided among the members.

Step A-4: Build a goods purchasing website using the AWS Management Console and AWS CLI according to the data definitions and parameter sheets, and measure the working time.

**3.2.2 Amount of documentation**

We measured the amount of documents according to the following procedure B-1 to B-4.

Step B-1: Make a list of deliverables that are required for waterfall model development in on-premise environment. The list of deliverables is standardized list based on the findings of the system construction that each company to which the member belongs.

Step B-2: 3.2.1 after the construction, set up additional deliverables required for the development by applying a serverless architecture to the list of deliverables made in step B-1.

Step B-3: Compares the amount of document between on-premise development and development with a serverless architecture. The evaluation is in the following three stages.

・Same workload as on-premise

・Reduced workload as on-premise

・No documentation required

Step B-4: For documents that are rated as "Reduced workload" or "No documentation required", provide a rationale for the merits of applying a serverless architecture.

Step B-5: The results of the above steps B-1 to B-4 are summarized as an evaluation sheet (Appendix C-1). The issues that emerged during the construction of 3.2.1 are also described in the evaluation sheet.

**3.3 Verification results**

The measurement results of the construction time and the amount of documentation are described below.

**3.3.1 Construction time**

The total construction time of the goods purchasing website (total of steps A-3 and A-4 in 3.2.1) was 10 hours. Step A-3: Development of application source code took 7 hours in total, and Step A-4: Set up time using the AWS management console and AWS CLI took 3 hours.

**3.3.2 Amount of documentation**

　As shown in Figure 3, 68 documents were required for the on-premises development, and 56 documents were required for a serverless architecture. So we reduced 12 documents in total. (18% of the documents created during on-premise development).

Here is the detail

・Same workload as on-premise development:38

・Reduced workload as on-premise

development:18(26% of the documentation created during on-premise development)

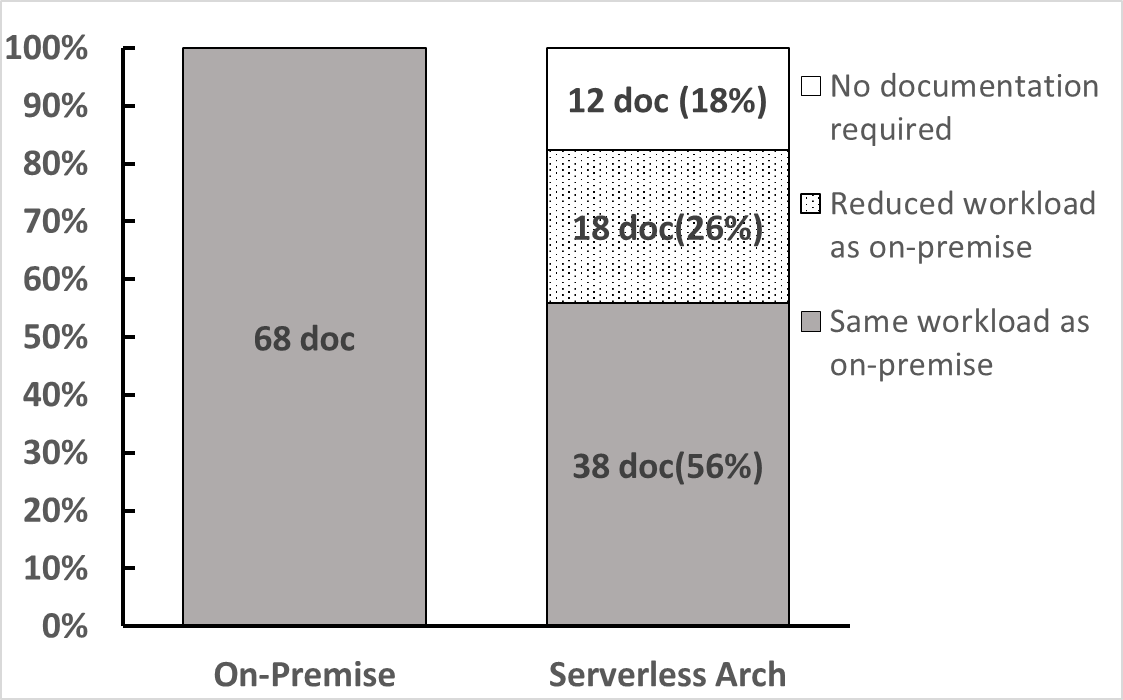


Figure 3. Verification results of amount of documentation

**４．Considerations**

From the above verification results, we can confirm that the amount of document creation and construction time can be reduced in the development of a serverless architecture. The reason for this is that the implementation of the non-functional requirements is automated. The simplicity of the functions is one of the reasons for the reduction of construction time. The details of each are described below. We also discuss the fact that application of a serverless architecture does not reduce the development time compared to on-premises systems.

**4.1 Automated Implementation of Non-functional Requirements**

We explain the reason using usage patterns of cloud services and the serverless liability model (Figure 1). We can reduce construction time by these infrastructure in figure 1.

・IaaS（Infrastructure as a Service）

The infrastructures below the "virtualization" are highly abstracted by cloud providers, so users can save a lot of labor in deploying, building, and managing them.

・PaaS（Platform as a Service）

In addition to the features of IaaS, a "middleware runtime" is provided. The degree of freedom and flexibility is limited because it depends on the specifications of the provider, but the introduction and configuration of various middleware and runtimes can be labor-saving. In addition, features to improve the scalability and availability of the system are provided and are configurable.

・Serverless

In addition to the features of IaaS and PaaS described above, it has built-in features to improve scalability and availability. The amount of design (documentation) and construction work is reduced because the user's responsibility for configuration is lower.

Using the services (DynamoDB and Lambda) used in the construction of the system, we confirm that the scalability and availability of these services are improved. Details are as follows.

1. DynamoDB

・Scalability

The performance of DynamoDB is determined by increasing or decreasing the number of capacity units. The user can set the number of capacity units appropriately according to the frequency of reading and writing data required by the system, and the size of data items (read/write). Set it up. or Auto Scaling is available [6].

・Availability

By default, data is distributed in three availability zones; for DynamoDB global tables (multi-database deployed in multiple regions), the service level agreement (SLA) is 99.999% or higher [7].

1. Lambda

・Scalability

　Instances processing Lambda functions are automatically scaled and processed in parallel [8].

・Availability

　Lambda executes functions in multiple availability zones. Therefore, when a service interruption occurs in one zone, event processing is guaranteed to continue [8].

As mentioned above, the serverless system has built-in features to improve scalability and availability. Users can select the functions (back-end connection, scalability, backup, etc.) provided by each serverless service according to their requirements and decide on the settings. In addition, the execution is guaranteed by the cloud provider, limiting the scope of user design and reducing the amount of time spent on building non-functional parts and creating documentation compared to developing in an on-premise environment.

**4.2 Simplifying the Functions**

　Each Lambda function is designed to have one or two purposes and to be stateless. Because of the simplicity of the functions, the number of program steps is small. This reduces the coding time and makes it easier to test. Also, since it is stateless, there is no need to consider the management of session information. This reduces the development time.

**4.3 Conclusion**

　By using a serverless architecture, we can shorten the development time until the release of a service and provide it quickly. As a result, we can say that business agility is improved.

**4.4 Work that is not reduced by the application of a serverless architecture**

　In this study, we could not confirm the shortening of the development period for business logic and application development. Users are responsible for analyzing the current state of the business, defining business requirements, designing the structure of the application, and designing the data to be used in order to clarify the flow of the business to be realized.

**５．Issues on applying serverless architecture**

The following issues were identified from this review.

**5.1 Selecting a Development Language**

When we developed the Lambda functions, the developer's favorite development language was selected for each function. As a result, a system with a mixture of multiple development languages was constructed. We have more opportunities to freely choose an appropriate development language.

However, it is preferable to select a development language before the development process because developing in multiple development languages is less manageability.

**5.2 Developing the cloud workforce**

We spend about one month in total for learning beforehand(basic concepts of AWS, each service specification, etc.) because the construction was done by a person who had no experience in system development using AWS. The services provided by AWS are constantly evolving, and in order to develop enterprise systems, not only the engineers on site but also the managers who hold the decision-making authority need to acquire knowledge. However, it is difficult to promote the development of enterprise systems as an organization when the experts are scattered in different departments and fields. Therefore, in order to promote the development of systems using the cloud, it is desirable to establish a specialized unit that consolidates knowledge and develops human resources across different fields.

**６．Future Prospects**

We have confirmed that the serverless architecture improves business agility by reducing the amount of documentation and construction work created during development. In addition, we believe that there is a possibility to further improve business agility by pursuing the following two perspectives, which have not been verified in this study.

The first is to focus on development work. By adopting a serverless architecture, we can reduce the amount of labor needed for operations, and reuse the cost and manpower for development. As a result, we believe that faster development will be possible.

The second is to make applications into microservices. By loosely connecting each application, we believe that it is possible to limit the scope of support for additional functions and specification changes, and to develop at low cost and with short delivery times. In addition to the above perspectives, it is expected that the combination of Domain-driven design and Agile development will have a greater effect.

However, in order to achieve these two points, it is necessary to develop human resources with advanced technical skills and rich knowledge of Cloud computing, as well as to foster an integrated system of development and operation, and an organizational culture that promotes agile development. We will continue our study on the possibility of improving business agility by applying a serverless architecture that includes these perspectives.

**Acknowledgements**

　We would like to thank Mamoru Kitagawa of IBM Japan, Ltd. for his advice on the preparation of this paper.

**References**

1. Ministry of Economy, Trade and Industry, Study Group for Digital Transformation, DX Digital Transformation Report〜IT System "2025Nen no Gake" no Kokufuku to DX no Honkakuteki na Tenkai〜, 2018.09.07
2. JGS Research 2019 Project Team Papers IA001 Naoya Matsuo, Proposal for the process to consider applying serverless architecture to business system , 2019.07.31
3. Gartner,Magic Quadrant for Cloud Infrastructure as a Service, Worldwide, 2019.07.16
4. Ministry of Economy, Trade and Industry, Commerce and Service Industry Policy Group, Cashless Payment Promotion Office, Cashless no Genjyo oyobi Igi, 2020.01
5. Serverless Airline Booking, [https://github.com/aws-samples/aws-serverless-airline-booking, 2020.05](https://github.com/aws-samples/aws-serverless-airline-booking,%202020.05)
6. AWS DynamoDB Read/Write Capacity Mode, https://docs.aws.amazon.com/ja\_jp/amazondynamodb/latest/developerguide/HowItWorks.ReadWriteCapacityMode.html, 2020.06
7. AWS DynamoDB Service Level Agreement, https://aws.amazon.com/jp/dynamodb/sla/, 2020.06
8. AWS Lambda function scaling, https://docs.aws.amazon.com/ja\_jp/lambda/latest/dg/invocation-scaling.html, 2020.06

Attachment: Member list

(alphabetical order of company name)

Appendix A-1: Goods Purchasing Website Requirements Specification

Appendix B-1: Data Definition Document

Appendix B-2: Parameter Sheet

Appendix C-1: Evaluation Sheet

Attachment: Member list

(alphabetical order of company name)