

IMPLEMENTATION OF INTEGRATION SYSTEM BASED ON MICROSERVICES USING DOMAIN-DRIVEN DESIGN METHODOLOGY

¹MUHAMAD FADHLY, ²AHMAD NURUL FAJAR

¹Information Systems Management Department, Binus Graduate Program – Master of Information Systems Management, Bina Nusantara University, Jakarta, Indonesia 11480

²Information Systems Management Department, Binus Graduate Program – Master of Information Systems Management, Bina Nusantara University, Jakarta, Indonesia 11480

E-mail: ¹muhamad.fadhly@binus.ac.id, ²afajar@binus.edu

ABSTRACT

The Tangerang Government's development planning process is currently running well, however, it is still constrained by the length of time it takes to input basic data and poor data accuracy by manually inputting data. The applications involved in providing basic data are still fragmented. So the data is not yet integrated, starting from planning, budgeting, accounting, monitoring and reporting.

This study proposes a system integration that is in accordance with the needs of the work plan reporting process in the monitoring and evaluation system. With this service based enterprise architecture, an integrated monitoring and evaluation system will be built. As a result, an architectural enterprise based on Microservices will be built that can perform data integration lightly and quickly using the Domain-Driven Design method approach which is considered suitable and appropriate for designing architecture in the performance monitoring and evaluation system of Tangerang City.

Keywords : *Enterprise Architecture, Integration System, Microservices, Domain-Driven Design, Tangerang City*

1. INTRODUCTION

In accordance with the mandate of the Minister of Home Affairs Regulation No. 86 of 2017, the Regional Planning and Development Agency (Bappeda) of Tangerang City is expected to carry out regional development planning in a transparent, responsive, efficient, effective, accountable, participatory, measurable, fair, environmentally and sustainable manner[1] (Article 5 of Domestic Regulation No. 86 of 2017. 2017). In addition to carrying out the planning in the regulation, a monitoring and evaluation stage of the Work Plan (renja) is regulated within the span of 1 (one) fiscal year.

And furthermore, in accordance with the Decree of the Mayor of Tangerang Number:

800/Kep.313.1-Inspektorat/2017 concerning the Integrated Corruption Eradication Program Action Plan, the Corruption Eradication Commission (KPK) requires a regional financial management system which includes an Electronic Monitoring and Evaluation System (E-Monev) .

Therefore, an information system that is able to facilitate the monitoring & evaluation stages has been built under the name E-Monev. The composition of the data for the monitoring and evaluation phase of the Regional Work Plan (renja) consists of 4 (four) data elements that complete the reporting format.

The four data sources in question are sourced from 4 (four) applications with different platforms, programming languages, databases and servers. The details consist of:

Table 1: Integration Data Source List

NO	DOCUMENT	DATA	APPLICATION	PLATFORM
1.	RPJMD (regional mid-term development plan)	Programs & Program Indicators	E-Planning	PHP MySQL
2.	DPA (Budget Execution Document)	Activities, Activity Outputs and Budget	SIPKDI (regional financial management information system)	PHP MySQL
3.	Development Report	Physical Realization	SIEVLAPI (integrated evaluation & reporting information system)	PHP MySQL
4.	SPJ (Letter of Accountability)	Financial Realization	SP3KTRA (Tangerang city financial management, administration & reporting system)	Java PostgreSQL

Later, after the above data is available, then the monitoring and evaluation process can be carried out by inputting the performance of each indicator that is coupled with the financial and physical realization that has also been achieved.

The impact of different and not integrated systems is that in the monitoring and evaluation process, users often have to re-enter the basic data. The following is information on time requirements if you do manual input [2].

Table 2 Manual Input Time Record

NO	FORM NAME	AVERAGE NUMBER OF FORMS AN INSTITUTION	ESTIMATED TIME
1.	Programs & Program Indicators	12	36 Minutes
2.	Activities, Activity Indicators & Budget	25	1 Hour 15 Minutes
3.	Physical Realization	15	32 Minutes
4.	Financial Realization	15	38 Minutes

Things like this cause problems both in terms of inefficient time and of course the risk of errors when re-entering planning and budgeting data and the realization that has been achieved.

In terms of application development, integration has now begun separately for each application using an API (Application Programming Interface). The API uses REST (REpresentational State Transfer) which allows users to directly access information in a database via HTTP (Hypertext Transfer Protocol)[3].

The HTTP methods commonly used in REST-based architectures are (GET, POST, PUT,

DELETE and OPTIONS) and replies are sent in either simple JSON or XML. So that the information received can be more easily read on the client application side [4].

The development using the REST method is also considered to be still not as expected because every application developer at the beginning of a work project is always charged with creating an API. Making this API apart from being time-consuming in making applications, will also reduce the quality of the application due to the lack of detailed time for application development, so that the project is then required to return to the project for application development.

It can be simulated that if Project X is charged with creating an API (Application Programming Language) it will reduce the time to build the application. whereas Project Y only requires API analysis if reusable APIs are available from previous projects.

In government management, data reuse and data sharing is a stage that becomes a rotating cycle. For example, the results of the annual budget audit will be used as a planning evaluation material for the next fiscal year.

Thus, the use of data integration utilizes a microservices architecture in which there is a management API of existing applications that are considered capable of responding to data integration needs in the local government management cycle, which in this case occurs in the Tangerang City Government.

Microservices are small applications with a responsibility that can be deployed, scaled and tested independently. Microservices are used throughout the industry to facilitate agile data delivery mechanisms for service-oriented architectures and to migrate legacy, function-oriented architectures to highly flexible service orientations. Microservices software breaks down systems and applications to a more granular and modular level [5].

To build a microservice-based architecture that has a fairly complex and detailed level of data usage, in this study the Domain-driven design (DDD) method used is based on the domain model of each application to be integrated.

Domain-driven design (DDD) is a popular model-driven methodology for capturing knowledge of relevant domains for software design. also to encourage understanding of emerging domains and design correctness, then emphasizing agile collaborative modeling from domain experts and software developers. DDD will capture the additional need because it provides a means to parse the domain into context. This context corresponds to the functionality of microservices that provide different business process capabilities [6].

As previously explained, the data source which is the master data in the Monitoring and Evaluation System is currently still independent and separate data sources, so that the process carried out requires repeated input. This creates a problem. The formulation of the problem is as follows:

1. What kind of architectural design can meet the needs of data integration for the monitoring and evaluation system of work plans (renja)?

2. How to build a prototype of an integrated work plan monitoring and evaluation system so that the databases of each system can be connected and can share data?

The objectives of implementing the integrated monitoring and evaluation system for regional development planning are:

1. Produce an architecture design for monitoring system application integration and evaluation of work plans with a Microservices architectural approach.

2. Build a prototype of an integrated work plan monitoring and evaluation system to improve business processes, namely by reducing repetitive data input manually.

The benefits expected with the integration of the monitoring and evaluation system for regional development planning are as follows:

1. The stages of monitoring and evaluating development planning data, especially for work plan data (renja) are carried out by all district, city and provincial governments throughout Indonesia. By conducting this case study research, it is possible to produce an integrated monitoring and evaluation system architecture that becomes a reference for all local governments throughout Indonesia.

2. Furthermore, this research can make good local government performance judged from the monitoring and evaluation stage of planning and reporting that is accountable and achieves with a good grade level. Which is annually carried out by the Ministry of State Apparatus Empowerment and Bureaucratic Reform, as well as the Ministry of Home Affairs of the Republic of Indonesia.

This research is about how to integrate data using a microservice architecture using the DDD (domain-driven design) method, not including API management, data integration security and load-balancing techniques for services that will appear later.

2. METHODOLOGY

A. E-Monev

The Tangerang City Government in carrying out the Monitoring and Evaluation of the Work Plan using E-Monev, the data comes from different

and independent applications. For planning data using the e-Planning application, budget data using SIKDI. As for the data on administration and financial reporting using the SP3KTRA application, then for the evaluation data and reporting on the progress of the physical realization of each activity using the SIEVLAP application..

After the basic data above is available. Thus, the Monitoring and Evaluation transaction process can then be carried out. The Monitoring and Evaluation process in question is by entering performance data within 3 (three) months of each existing indicator, the performance data entered can also be viewed from the achievement of financial reporting values and physical realizations that have been achieved..

B. E-Planning

E-Planning or Regional Development Planning Information System (SIPPD) is an information system used for the preparation of the RKPD (Regional Government Work Plan) and Renja (Work Plan) of Regional Apparatus Organizations for planning and budget targets, so that they can be completed easily, quickly, and efficiently. appropriate and in accordance with what is mandated in the Regulation of the Minister of Home Affairs No. 54 of 2010 concerning the Implementation of Government Regulation No. 8 of 2008 concerning the Stages of Preparation of Control and Evaluation of the Implementation of Regional Government Development Plans

C. SIPKDI

SIPKDI or Regional Financial Management Information System is an integrated application that is used as a tool that aims to increase the effectiveness of the implementation of various regional financial management regulations based on the principles of efficiency, economy, effectiveness, transparency, accountability and auditability. Based on Permendagri Number 13 of 2006 concerning Guidelines for Regional Financial Management.

This application is also a manifestation of the local government's real action in the field of regional financial management, in order to strengthen the common perception of regional financial management systems and procedures in the interpretation and implementation of various laws and regulations..

D. SP3KTRA

SP3KTRA, which stands for the Tangerang City financial management, administration, and reporting system, is a facilitation of the Tangerang City regional financial management work process which is manifested in a high-tech cloud application system and reflects the integration of processes and data integration of the overall implementation of regional financial management.

This application system is built based on Government Regulation Number 71 of 2010 concerning Government Accounting Standards, Minister of Home Affairs Regulation Number 64 of 2013 concerning Application of Accrual Accounting System in Regional Governments and Minister of Home Affairs Number 13 of 2006 concerning Guidelines for Regional Financial Management.

E. SIEVLAPI

The Evaluation and Reporting Information System, which is abbreviated as SIEVLAPI, is an application that is used as a development reporting medium for the physical or non-physical realization of activities or their derivatives, it also includes a comparison of the realization of financial or budget reporting that supports the physical realization.

This application system is used by the Tangerang City Government through the Development Control Section of the Regional Secretariat for benchmarking data on development reporting achievements that are running based on the active fiscal year. It is also often used as the basis for evaluating the absorption capacity of the budget from the activities carried out.

F. Microservices

Microservices can be seen as a technique for developing software applications that inherit the principles and concepts of the Service Oriented Architecture (SOA) style. it is possible to structure a service-based application as a very small, loosely coupled collection of software services.

Microservices architecture can be seen as a new paradigm for programming applications through a composition of small services, each running its own process and communicating through lightweight mechanisms. Microservices for

domain expert, it is a tightly structured and selective abstraction of that knowledge [11].

Microservices is a software architecture concept where the system consists of small services that work together and are autonomous, deployable, scalable, modeled on a bounded context [8].

G. API Gateway

API Gateway An API Gateway is a server that is the single entry point into the system. This is similar to the outward side view pattern of object-oriented design.

An API Gateway encapsulates the internal system architecture and provides a customized API for each client. An API Gateway may have other responsibilities such as authentication, monitoring, load balancing, caching, request shaping and request management, and static response handling.

The API Gateway is responsible for routing, composition, and protocol translation requests. All requests from clients go through API Gateway first. It then routes those requests to the appropriate microservices.

The API Gateway will often handle requests by implementing multiple microservices and aggregating the results. It can translate between web protocols like HTTP and WebSocket and web protocols that are not commonly used internally [9].

H. Domain-Driven Design

Domain Driven Design is a complex approach to software development where: Focus on core domains Explore models in creative collaboration of domain practitioners and software practitioners Speak languages everywhere in explicitly constrained contexts [10].

Domain Driven Design combines design and development practices, and shows how design and development can work together to create better solutions. Good design will speed up development, while the input that comes from the development process will improve the design [11].

The Domain Model is not a specific diagram, it is the idea the diagram is trying to convey. This is not only the knowledge that lies at the head of the

Bounded Context is not a Module. The Bounded Context provides the logical framework within which the model develops. Modules are used to organize the elements of the model, so the Bounded Context will include the Module [11].

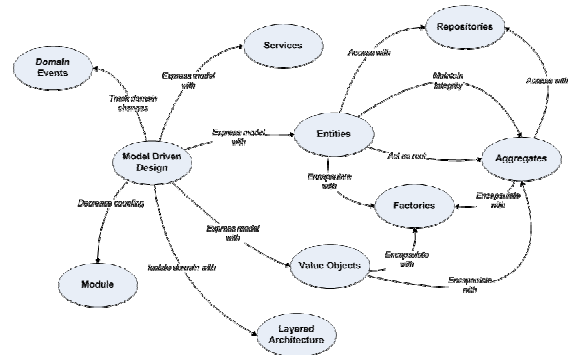


Figure 1. Building blocks Domain Driven Design[12]

I. REST

REST is a type of web service that applies the concept of switching between states. State here can be described as if the browser requests a web page, then the server will send the current state of the web page to the browser. Navigating through the links provided is the same as changing the state of the web page. Similarly, REST works, by navigating through HTTP links to perform certain activities, as if they were switching states from one another. [13].

REST Webservices builds integrations in a lighter and simpler way, and focuses on resources [14]. The main idea of REST is the concept of resources as components of the application that need to be used or addressed [15].

3. RESULT S AND DISCUSSION

According to the results of an interview with the Head of Development Planning, Evaluation and Reporting, Bappeda. The owner of this project suggests studying the monitoring and evaluation report format, which can be taken from the Minister of Home Affairs Regulation Number 86 of 2017 with report type E.81.

[illegible]

Figure 2. Work Plan Monitoring and Evaluation Report Format

In the report, a monitoring and evaluation format is made which is carried out by all provincial governments as well as districts and cities. And the report format is used as a reference for monitoring and evaluating work plans. Here is what the report looks like

From the results of interviews with all stakeholders involved. So, it can be concluded that the data requirements needed in data integration for monitoring and evaluating this work plan, as for the list are as follows.

Table 3. Data Requirements and Application Sources

NO	DATA	APPLICATION SOURCE
1	Program	E-Planning
2	Program Indicator	E-Planning
3	Program Budget Plan	E-Planning
4	Activity	E-Planning
5	Activity Indicator	E-Planning
6	Activity Budget Plan	E-Planning
7	Activity Budget	SIPKDI
8	Outputs of Activities	SIPKDI
9	Financial Realization	SP3KTRA
10	Physical Realization	SIEVLAPI

Next, this data will be made into an API and entered into the API Gateway. And then it will be entered into the E-Monev application database for the basis of monitoring and evaluation and reporting

A. Analysis of Existing E-Planning

To be able to provide an overview of the flow in the E-Planning application in order to facilitate the integration process that will be carried out, a

flowchart is made. This is intended so that the process of forming the data to be integrated can be mapped.

This application stores data related to development planning. All data stored is a plan and there is no certainty that it will be implemented because it will be compared with the availability of funds and other readiness

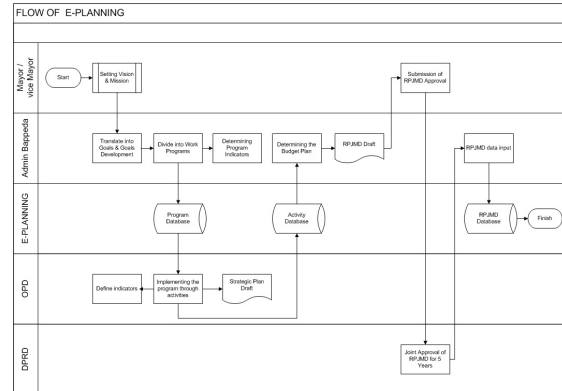


Figure 3. E-Planning Application System Flow

If you look at the flow above, the E-Planning application will overwrite the Program and Activity database as well as other complementary attributes such as indicators and funding plans or budgets. All data will be stored in the RPJMD database after being jointly approved by the Mayor and DPRD.

B. Analysis of Existing SIPKDI

After the previous application contained planning data, the SIPKDI application (regional financial management information system) is an application that stores budget data that will be carried out every year.

This budget data is also initially sourced from planning data in E-Planning which is taken in an annual format for estimation by each implementing agency, then it will also be jointly approved by the mayor and DPRD.

And after ratification, the value of the data will be stored in this application as a reference for implementing activities, the outputs to be achieved are in accordance with the previously agreed budget provisions. There is also a shift in the budget due to important needs and aid funds from the central government, so it is enough to enter directly into the application

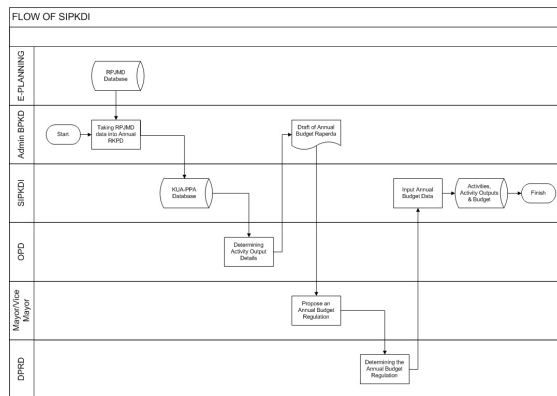


Figure 4. SIPKDI Application System Flow

C. Analysis of Existing SP3KTRA

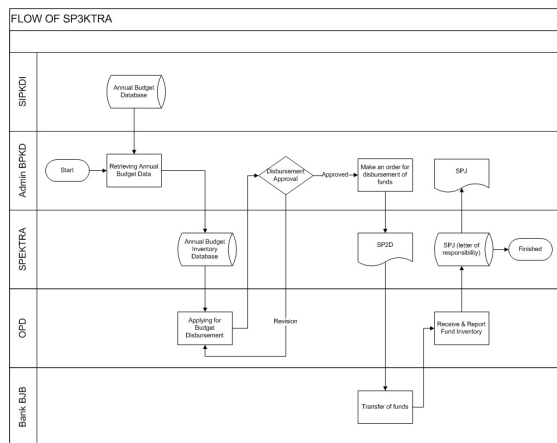


Figure 5. SP3KTRA Application System Flow

The SP3KTRA application is an application that is used as a medium for budget disbursement and reporting. The basic data used in this application is data from the SIPKDI application, because all budget data that is subject to rules is in the application.

Each agency or OPD submits a request for disbursement of funds, then the admin will review the value of the submission against the priority and availability of funds. If approved, a letter of disbursement of funds will be issued which will also be copied to the Bank for direct transfer to the account to the destination. All disbursement activities will be recorded and reported and the reporting is validated through this application.

D. Analysis of Existing SIEVLAPI

SIEVLAPI or evaluation and reporting information system is an application that stores everything related to the physical facts of the regional budget. In this application, agencies or OPD that carry out activities or use a budget are required to report visible activity results in the form of a percentage of real results to the budget value used.

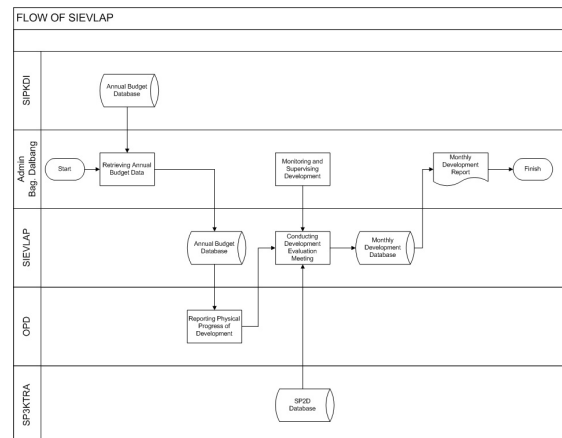


Figure 6. SIEVLAPI Application System Flow

This application utilizes annual budget data from the SIPKDI application to be used as the basis for the physical realization achieved by each activity implementer. Later each OPD will report the results of its physical realization and will also be monitored and validated for the realization value by the Admin in the Development Control Section..

E. Data integration analysis

From the results of the analysis of the above application flows. So, it can be described a data flow chart plan that will be used for integration into the E-Monev application, especially for the needs of monitoring and evaluating work plans using the API..

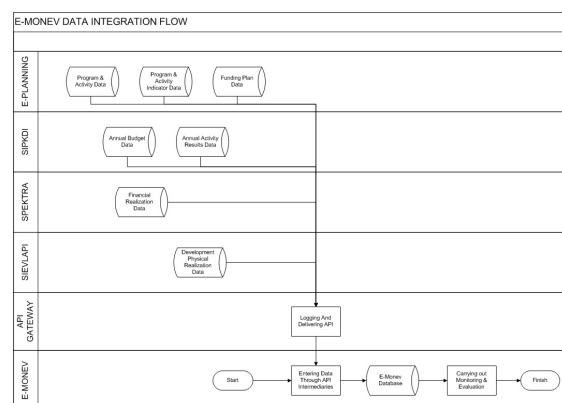


Figure 7. E-Money Data Integration Flow

Next, a service or API will be built according to the flowchart above. Then the API that has been created will be registered on the API Gateway so that every existing service can be easily controlled and easy to reuse in the future.

The API Gateway that was built aims as a single entry point that will orchestrate all existing APIs, so that later the URL address (Uniform Resource Locator) from each existing application becomes uniform 1 (one) URL address even though it is from different applications and different APIs.

F. Microservices Design With a Domain-Driven Design Approach

The initial phase of this approach using Domain Driven Design (DDD) is domain analysis. The domain model is an abstract model of the business domain. This process is to filter and organize domain knowledge, and provide a common language for developers [16]. After conducting an analysis of the running system, a description of the domain analysis was found, which is as shown in the following figure:.

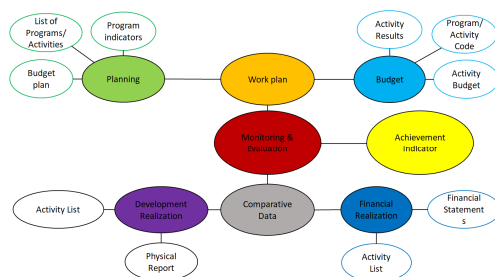


Figure 8. Integration Domain Analysis Monitoring and Evaluation of Work Plans.

From the domain analysis figure above, it can be seen that the achievement of indicators to be monitored and evaluated comes from 2 (two) groups of data domains, namely (1) the basic data group for the work plan and (2) the comparison data group. The details are as follows

Table 4. List of Domains for Monitoring and Evaluation of Work Plan Integration

Domain Group	Domain	Sub-Domain
Basic Data	Planning	List of Programs &

	Budget	Activities
		Indicator
		Budget plan
		Program & Activity Code
		Activity Results/Outputs
Comparative Data	Development Report	Activity Budget
		Activity List
	Financial Statements	Physical Report
		Activity List
		Financial Statements

The conclusion of this domain analysis is to produce 4 (four) domains and 10 (ten) sub-domains. Furthermore, these domains and subdomains will be used as the basis for integration development such as determining the number of services or APIs and so on.

The next discussion will gradually describe a stage in architectural design using Domain-Driven Design in the development of system integration in this study referring to the above domain analysis.

The domain model describes in a structured way the data flow based on certain parameters, visually also depicts areas related to activities or data usage [17]. The following is an overview of the domain of the integration architecture model that will be implemented.

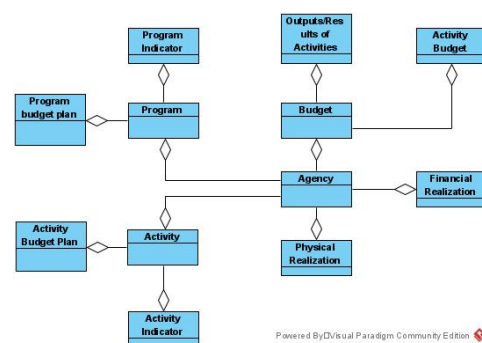


Figure 9. Domain Model

From the picture above, the root domain will be OPD (regional device organization), technically this parameter can be id_opd or code_opd as the primary key. This parameter will be used as the main parameter for other domains, such as

program domain, activity domain, budget domain, financial and physical realization domain. Furthermore, all data flow will go through this root domain parameter. And for the basic OPD data, the source will be found from the application which is the beginning of the stage, namely the E-Planning application

- Bounded Context

Bounded Context is an approach that separates large models into smaller contexts explicitly and the relationship between them [17]. To make it easier and align the data cross, it is known the **code_path** which is part of the attributes of the OPD model.

Adjusting to the number of models in the analysis of the eating domain, the number of bounded contexts described also amounts to 4 (four) with 1 (one) root domain.

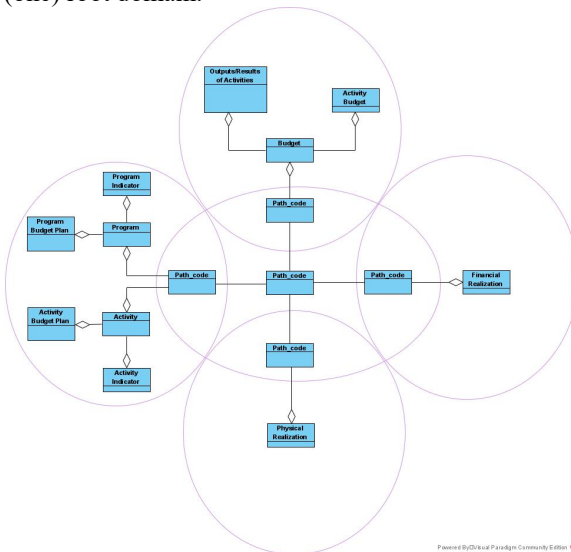


Figure 10. Bounded Context

- Entities

To describe entities or entities and their attributes, it is very suitable to use class diagrams on UML (unified modeling language) diagrams. Class will also describe the relationship between one entity with another entity

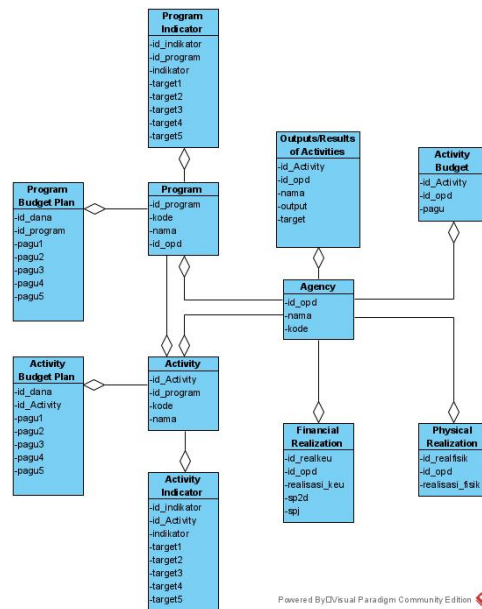


Figure 11. Entities Diagram

There are at least 11 (eleven) entities that can be connected to each other to get the expected integration pattern. Those eleven entities are

1. OPD
2. Program
3. Program indicators
4. Program funding plan
5. Activities
6. Activity indicators
7. Activity funding plan
8. Activity output
9. Activity budget
10. Financial realization
11. Physical realizationValue Object

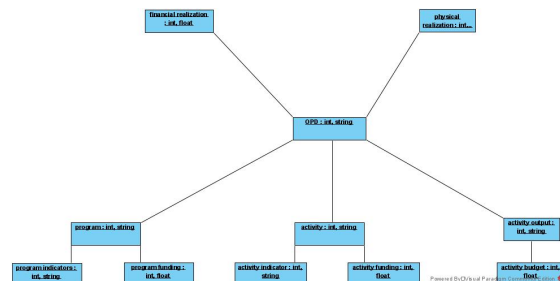


Figure 12. Diagram Value Object Intergrasi

Value objek describes the identity of an entity or object, in detail the object value describes the data type used in each object. Later the API that is

formed will also adjust the value of this described object. The following is a list of data types used for each object or entity in general

1. Opd : integer, string
2. Program : integer, string
3. Program indicators: integer, string
4. Program funding plan: integer, float
5. Activity : integer, string
6. Activity indicators: integer, string
7. Activity funding plan: integer, float
8. Activity output: integer, string
9. Activity budget : integer, float
10. Financial realization : integer, float
11. Physical realization : integer, char

• Services

There are 4 (four) interconnected layers, namely presentation layer, application layer, domain layer, and infrastructure layer. The details are densely packed in the following service diagram:

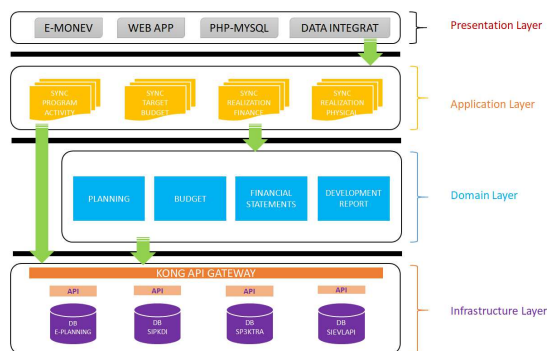


Figure 13. Service Layer Integration Diagram
Monitoring and Evaluation of Work Plans [18].

The details of these layers are as follows:

1. Presentation layer : Responsible for presenting information to users and interpreting user commands, consisting of below:
 - E-Monev Application
 - Web-based application
 - Platform uses php-mysql Engine
 - Its main function is to integrate work plan data.
2. Application layer : The layer that coordinates application activities. does not contain any business logic and does not hold the state of a business object, but

it can hold the progress state of an application task. consist of:

- programs – synchronization activities
- targets – synchronization budgets,
- synchronization financial realization report
- physical realization of synchronization

3. Domain Layer : This layer contains information about the business domain. The state of the business object is held here. business objects, and may have their state delegated to the infrastructure layer. Consist of below:

- Planning Domain
- Budget Domain
- Financial Report Domain
- Report Domain

4. Infrastruktur Layer : This layer acts as supporting information for all other layers. And this layer implements business objects. Consist of below

- Database & API E-Planning
- Database & API SIPDKI
- Database & API SP3KTRA
- Database & API SIELAPI
- Kong API Gateway

• Modules

The use of modules is actually common in system development. The purpose of dividing classes into modules is a division of responsibility for making a collection of concepts stand independently.

In this modeling module there are 6 (modules) with details of 5 (five) distributed modules and 1 (one) main module as a target that will be connected using Rest-API. The modules are as follows

1. Program Module, on E-Planning
2. Activity Module, on E-Planning
3. Budget Module, on SIPKDI
4. Financial Realization Module, on SP3KTRA
5. Physical Realization Module, on SIEVLAP
6. Renja Module, on E-Monev (Target Module)

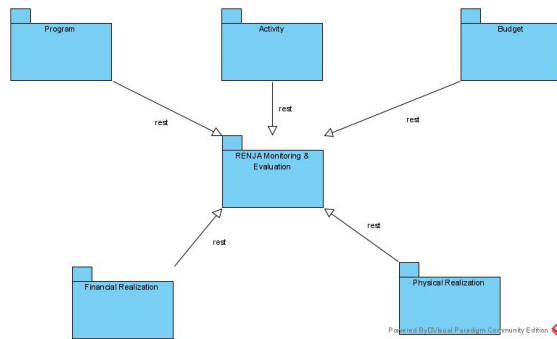


Figure 14. Modules Diagram

- Factory

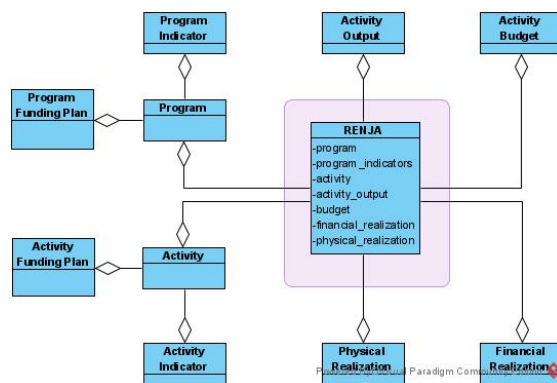


Figure 15. Factory Diagram

Factory is where an attribute of an entity or object will be centered, and in the system that will be integrated, what will act as a factory is the Work Plan object. In this object later the distributed attributes will be collected according to the needs of the system integration. and as for the installation of the attributes of the money object to be placed in the factory, it is as follows:

1. Program
2. Program Indicators
3. Activities
4. Activity Output
5. Budget
6. Financial Realization
7. Physical Realization

- Aggregates

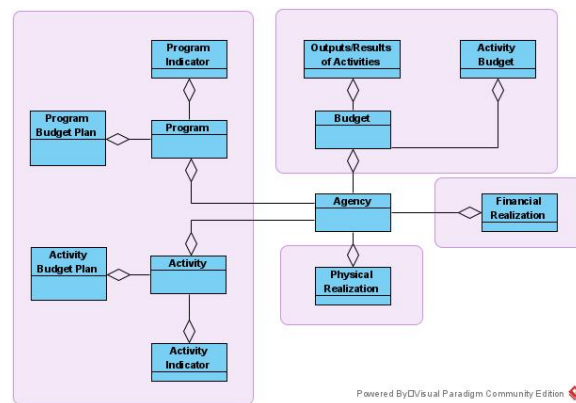


Figure 16. Aggregates Diagram

Compared to the factory, the aggregate is the source of the entities that will be installed in the factory. If you review the Factory, you can get 4 (four) aggregates that support data sources in this integration architecture, namely

1. Programs, from the E-Planning application
2. Program Indicators, from the E-Planning application
3. Activities, from the E-Planning application
4. Activity Output, from the SIPKDI application
5. Budget, from the SIPKDI application
6. Financial Realization, from the SP3KTRA application
7. Physical Realization, from the SIEVLAPI application

- Repositories

It is responsible for the management of objects life cycle. It concentrates the creation, change and objects removal operations [19]. Generally, one repository class is created for one entity. Without a repository, it will be difficult to unit test the business processes and duplicate queries can occur

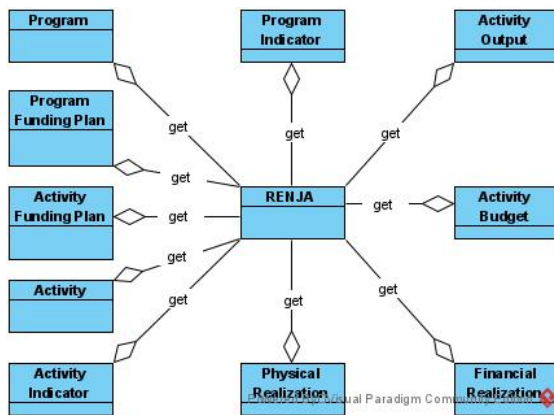


Figure 17. Repositories Diagram

In the development of an integrated system with DDD, the repository is limited to only accessing the aggregate. To decide which repository to create, we must refer back to system requirements. The number of queries that will be used refers to the number of APIs that will be registered on the API Gateway, and the method used in general is GET

- Layered Architecture

Before building a Microservice prototype for the integration of Monitoring and Evaluation of Work Plans, it is necessary to create a simple model based on layers that can explain the degradation and collaboration between layers that will be integrated from various application sources and in accordance with the previously described DDD graphs..

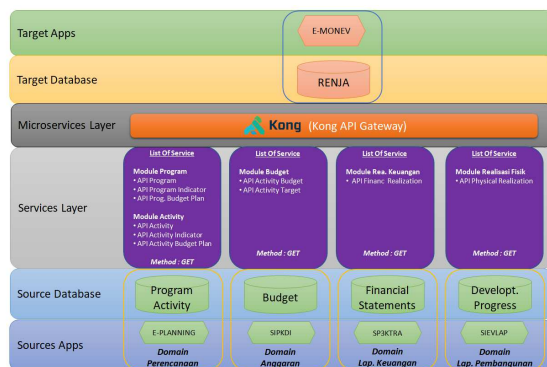


Figure 18. Layered Architecture Integration.

G. Microservices Prototype Development

Before building a prototype of a microservice-based architecture, it is considered necessary to compare between architectures before implementing microservices with after building microservices. The aim is to determine the transition of the system to the application of microservices so as to provide clarity on the development process. Here is a picture of the architecture

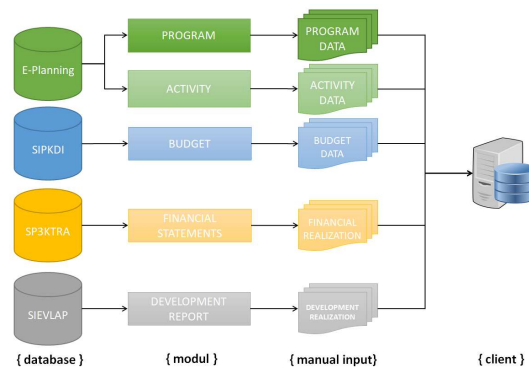


Figure 19. Architecture Before Implementing Microservices

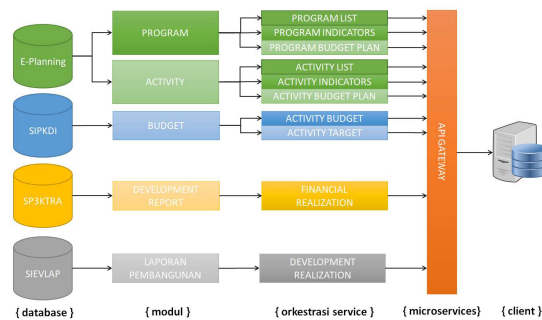


Figure 20. Architecture After Implementing Microservices

From Figure 4.19 and Figure 4.20 above, it can be seen the comparison between before and after implementing microservices. The most significant difference is that there is an API Gateway which is the orchestration of each data service.

H. Service Endpoint Identification

After being able to identify the architecture before and after the implementation of system integration in the previous discussion. So, in the next

discussion, it is to build an endpoint service based on Rest API.

The endpoint service that is built is identified based on the functions and domains that have been

previously analyzed using the DDD method. The following list of service endpoints will be entered into the API Gateway and will form an orchestra. Here is a list of endpoints that have been built.

Table 5. List of Service Endpoints

NO	URL	FUNCTION	DOMAIN
1	https://openkeuda.tangerangkota.go.id/services/eplanning/masterMprogram	LIST OF PROGRAM	PLANNING
2	https://openkeuda.tangerangkota.go.id/services/eplanning/masterFinalProgramIndikator	INDICATOR PROGRAM	PLANNING
3	https://openkeuda.tangerangkota.go.id/services/eplanning/masterFinalProgram	PLAN FUND PROGRAM	PLANNING
4	https://openkeuda.tangerangkota.go.id/services/eplanning/masterMkegiatan	LIST OF ACTIVITY	PLANNING
5	https://openkeuda.tangerangkota.go.id/services/eplanning/masterFinalKegiatan	INDICATOR ACTIVITY	PLANNING
6	https://openkeuda.tangerangkota.go.id/services/eplanning/masterFinalKegiatanIndikator	PLAN FUND ACTIVITY	PLANNING
7	https://openkeuda.tangerangkota.go.id/services/keuangan_sipkdi/indikator_prokeg	ACTIVITY BUDGET	BUDGET
8	https://openkeuda.tangerangkota.go.id/services/keuangan_sipkdi/pagu_prokeg_skpd	TARGET ACTIVITY	BUDGET
9	https://opendatav2.tangerangkota.go.id/services/sp3ktra/la_poran_sp3ktra	REALIZATION FINANCE	FINANCIAL STATEMENTS
10	https://opendatav2.tangerangkota.go.id/service/sievlap/laporan_sievlapi	REALIZATION PHYSICAL	DEVELOPMENT REPORT

For the distribution of endpoint placements, it is divided into 2 (two) URL addresses, namely the addresses openkeuda.tangerangkota.go.id and opendatav2.tangerangkota.go.id. The purpose of the placement on the two URLs is in accordance with the data designation for the respective affairs of the data.

The next stage is the development of the API Gateway. At the development stage of the Microservice Prototype in this research, using the API Gateway using a device with an open source license called KONG, this KONG device runs web-based, and with this web platform it is hoped that it will make it easier to use. The following is the initial view of the KONG API Gateway and is immediately confronted with username and password authorization.

If you have successfully authorized, you can access the services menu in the menu navigation on the left side. In accordance with the domain described

through DDD, 4 (four) services are registered on this API Gateway. However, in these 4 services, there are many microservices that become the substance of data integration needs that will be carried out for monitoring and evaluating work plans.

Furthermore, the API used will be explained in the following sub-discussion according to the previously determined domains. After being registered on the API Gateway, the URL address naming will be the same, namely:

`http://{IP Address/domain}/{service}`

Example (Single private IP address):

`http://{API Gateway}/eplanning`

`http://{API Gateway}/sipkdi`

`http://{API Gateway}/sp3ktra`

`http://{API Gateway}/sievlap`

I. Integrated System Display

The following is the appearance of the E-Money application on Figure 22, Figure 23, Figure 24, and Figure 25, it is after implementing microservices-based system integration. In the following display there is a button with a function to pull data using the API and it will be stored in the E-Money application database.

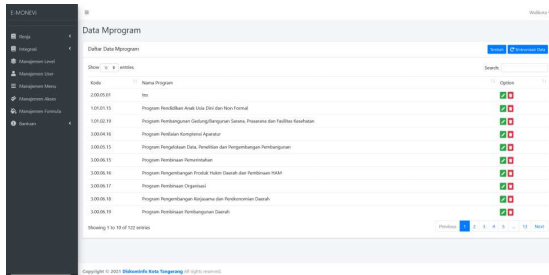


Figure 22. List Of Program Display

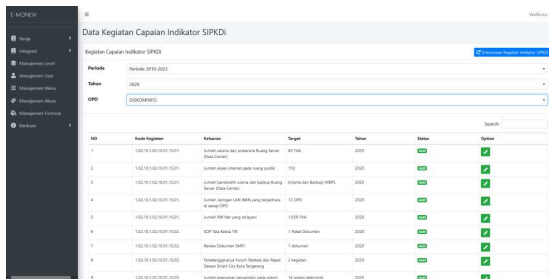


Figure 23. Activity Target Display

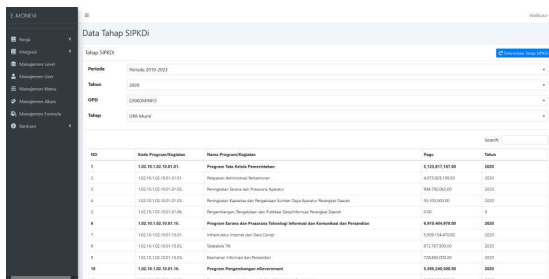


Figure 24. Activity Budget Display

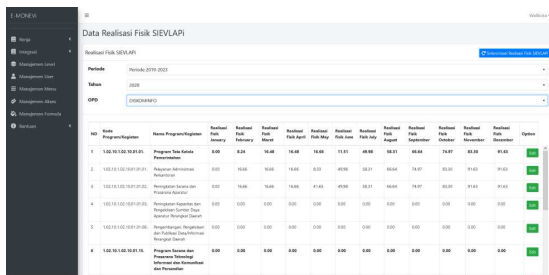


Figure 25. Physical Realization Display

4. CONCLUSION

Based on the results of research on the integration of systems based on the Microservices architecture to the Monitoring and Evaluation System of the Work Plan, the following points can be concluded in this study:

1. Development of microservices using the DDD method produces an enterprise architecture design that results in the design and implementation of integration services that are as expected in this study and answer the problems that arose previously.
2. Utilization of system integration saves a lot of time which initially takes a matter of hours to just seconds, when compared to the data re-input pattern as shown in table 6. In addition, the use of data integration will have an impact on the correctness of the available data values. Because with the re-input method there is often an error in the input data value.

Table 6. Integration Time Record

NO	FORM NAME	AVERAGE NUMBER OF FORMS	ESTIMATED TIME
1.	Programs & Program Indicators	12	1,5 Seconds
2.	Activities, Activity Indicators & Budget	25	7,3 Seconds
3.	Physical Realization	15	5,1 Seconds
4.	Financial Realization	15	5,6 Seconds

3. The implementation of the integration architecture using the Domain-Driven Design (DDD) method makes the integration system design easier for the system development process. DDD can clearly describe the proposed architectural design ideas to programmers or system developers, and most importantly DDD-based architectural design can accommodate the needs of object-oriented system development.
4. The implementation of an integrated system based on microservices using KONG is able to maximize the integration process of the new system to be integrated. This is caused by reusing services or APIs from the API Gateway that were previously available and not creating new services or APIs. And what is more about KONG is that it has a GUI (graphical user

- interface) based interface, so it's quite easy to use when compared to a CLI (command-line interface) based interface.
5. The results of the integrated architecture design produced in this research can be used as a reference for the data integration model of the Work Plan Achievement Report (renja) for other government agencies in Indonesia, especially local governments at the provincial government level or the district or city government level. Because the monitoring and evaluation stage applies to all government agencies under the auspices of the Ministry of Home Affairs.
 6. Furthermore, the increasing number of applications that are integrated through the API Gateway will certainly add to the burden on the available resources. In addition to the need to adjust the needs of existing resources, of course a load balancer mechanism is also needed. According to several studies this method will balance the incoming load against the strength of the available resources [20].
- REFERENCES:**
- [1] G. Sari, "Evaluasi Capaian Target Rencana Kerja Terhadap Rencana Strategi Badan Perencanaan Pembangunan Daerah Bappeda Provinsi Kepulauan Bangka Belitung Tahun 2017-2018," *Equity J. Ekon.*, Vol. 6, No. 02, Pp. 7–24, Aug. 2020, Doi: 10.33019/Equity.V6i02.20.
 - [2] A. Nasrun, R. A. Hendra, And M. Priandi, "Urgensi Integrasi Sistem Informasi Akuntansi Instansi Pemerintah," *J. Tek. Its*, 2012.
 - [3] T. Hidayat, Suhardi, And N. B. Kurniawan, "Smart City Service System Engineering Based On Microservices Architecture: Case Study: Government Of Tangerang City," In *2017 International Conference On Ict For Smart Society (Iciss)*, Sep. 2017, Pp. 1–7, Doi: 10.1109/Ictss.2017.8288864.
 - [4] S. P. Ong Et Al., "The Materials Application Programming Interface (Api): A Simple, Flexible And Efficient Api For Materials Data Based On Representational State Transfer (Rest) Principles," *Comput. Mater. Sci.*, Vol. 97, Pp. 209–215, 2015, Doi: 10.1016/J.CommatSci.2014.10.037.
 - [5] X. Larrucea, I. Santamaria, R. Colomo-Palacios, And C. Ebert, "Microservices," 2021.
 - [6] F. Rademacher, S. Sachweh, And A. Zündorf, "Towards A Uml Profile For Domain-Driven Design Of Microservice Architectures," *Lect. Notes Comput. Sci. (Including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, Vol. 10729 Lncs, No. Iii, Pp. 230–245, 2018, Doi: 10.1007/978-3-319-74781-1_17.
 - [7] F. Rademacher, J. Sorgalla, And S. Sachweh, "Challenges Of Domain-Driven Microservice Design," *Ieee Softw.*, 2018.
 - [8] L. Khoirunnisa, "Rancang Bangun Sistem E-Learning Berbasis Microservices Dan Domain Driven Design (Studi Kasus Probistek Uin Maulana Malik Ibrahim Malang) Oleh: Lia Khoirunnisa'," 2019.
 - [9] C. Richardson And F. Smith, *Microservices - From Design To Deployment*. 2016.
 - [10] E. Evans, *Domain-Driven Design - Tackling Complexity In The Heart Of Software*. 2003.
 - [11] A. Avram And F. Marinescu, *Domain Driven Design Quickly*. 2006.
 - [12] Uniknow, "Tutorial Domain Driven Design Last Published: 2015-04-09 | Version: 0.1.8-Snapshot," 2020. [Http://Uniknow.Github.io/Agiledev/Site/0.1.8-Snapshot/Parent/Ddd/Core/Introduction_Ddd.Html](http://Uniknow.Github.io/Agiledev/Site/0.1.8-Snapshot/Parent/Ddd/Core/Introduction_Ddd.Html).
 - [13] Y. Fauziah, "Aplikasi Iklan Baris Online Menggunakan Arsitektur Rest Web Service," *Telematika*, Vol. 9, No. 2, 2014, Doi: 10.31315/Telematika.V9i2.286.
 - [14] A. J. Rettig, S. Khanna, And R. A. Beck, "Open Source Rest Services For Environmental Sensor Networking," *Appl. Geogr.*, Vol. 60, Pp. 294–300, 2015, Doi: 10.1016/J.Apgeog.2014.11.003.
 - [15] D. Guinard, M. Mueller, And V. Trifa, "Restifying Real-World Systems: A Practical Case Study In Rfid," *Rest From Res. To Pract.*, Pp. 359–379, 2011, Doi: 10.1007/978-1-4419-8303-9_16.
 - [16] A. N. Fajar, E. Novianti, And Firmansyah, "Design And Implementation Of Microservices System Based On Domain-Driven Design," *Int. J. Emerg. Trends Eng. Res.*, Vol. 8, No. 7, Pp. 3058–3062, 2020, Doi: 10.30534/Ijeter/2020/30872020.
 - [17] J. Mufid, "Mengenal Domain Driven Design," <https://Medium.Com>, 2020. <https://Medium.Com/@Mufid.Houraluddin/Mengenal-Domain-Driven-Design-E9b2b1b58cce> (Accessed Oct. 09, 2020).
 - [18] D. D. Design, I. Domain, And D. Design, "Introduction Domain Driven Design . Building Blocks Domain Driven Design," 2020. .
 - [19] E. C. S. Santos, D. M. Beder, And R. A. D. Penteado, "A Study Of Test Techniques For Integration With Domain Driven Design," *Proc. - 12th Int. Conf. Inf. Technol. New Gener. Itng 2015*, No. 20, Pp. 373–378, 2015, Doi: 10.1109/Itng.2015.66.
 - [20] S. Newman, *Building Microservices*. 2015.