Name: Maulfi Alfansuri, DE BATCH 03

Homework - Introduction to Data Engineer - Big Data

Qoriyana Nurselvi
•
Kemarin
100 poin
Tenggat: 18 Okt 23.59
PR 1 merupakan tugas **INDIVIDU**

Misalkan anda ditugaskan sebagai Data Manager
disalah satu perusahaan yang bergerak dibidang restoran (food and beverage).
Anda diminta untuk membangun platform data analytics dengan tujuan untuk menerapkan proses bisnis CRM (Customer Relationship Management) dimana terdapat
channel sumber data :

1. **Apps Mobile**
2. **Apps Website**
3. **Digital Marketing Tools (Google Analytics, Instagram Ads)**

Tugas pertama anda adalah membuat arsitektur data
untuk membangun data analytics.

Kerjakan PR dalam bentuk **MS Word (.docx)**, dan kumpulkan dalam bentuk **PDF**. Setelah itu, upload PR ini ke dalam Google Classroom paling lambat **Hari Senin, Jam 23.59 WIB**. Penalti kalau mengumpulkan PR lebih dari tenggat waktu yang telah ditentukan adalah nilai yang kosong (0).

Selamat mengerjakan teman-teman, dan good luck! :)

Name: Maulfi Alfansuri, DE BATCH 03

Two Phase Data-Architecture Design for CRM Business Based on Microservices-Services Oriented Architecture

Maulfi Alfansuri

Name: Maulfi Alfansuri, DE BATCH 03

TABLE OF CONTENTS

Name: Maulfi Alfansuri, DE BATCH 03

1. Abstract

Here we derive our architecture from identified CRM key services of real-world use cases with several initial assumptions along with 3 years data architecture live cycle. To do so we propose the design by combining data architecture design principles with microservices-services architecture for 2 business phases (Initial and Late) as follow: Data architecture based on fundamental distinction of fast-big data with design conceptual mentioned in the enterprise data architecture [1], combined with architecture flows used by Sang.et.al [2], in which imbued with microservices-services oriented components and processes [3].

2. Preliminary

For rephrasing the objective of the task: "To construct a data architecture for CRM based business" where data sources are given. Several assumptions are necessary for starting points in the following list:

- The company is in the starting phase, which open in 3 different provinces.
- The architecture focus is limited to data analytic platform.
- The system incorporates CRM principles.
- The food and beverages services mentioned in this case is **not** about developing CRM services software for any type of restaurants.
- The restaurant in this context, is singular business chain, unified in one brand name.
- The capital is adequate to support the proposed architecture.

Designing data architecture means there are standardized rules and design principle to follow as the structure, with scope and assumptions for design boundaries, also we must gain a clear understanding first about CRM and the relationship big data system, Therefore at least the following questions must be tackled.

- What is CRM?
- What is the principle, component scope of Data Architecture/design?
- What is microservices - services oriented architecture? What is the significance to CRM?
- What is the CRM strategy derived business services in question?
- How to tailor the business services into specific Data Architectures?
- What are the alternatives and cost calculation method-output?

The design will be made by tailoring the principle of design with business strategy based off CRM. Several architectures will be posed, and the optimal choice will be presented. In the end for each architecture, the sufficient entity-process and infrastructure of the data architecture are identified and visualized through concrete diagram. It needed to be emphasized that understanding the framework of CRM could be pivotal for designing the data engineering framework in the future, and it could take a majority coverage of this assignment.

Name: Maulfi Alfansuri, DE BATCH 03

2.1.Customer Relationship Management (CRM)

Business performance of any company relies not only to its product. There is no single product that can ensure sustainability by only take care the internal aspect chosen by the company -no matter how objectively good it is. The major aspect of business is the external factor: The customer.

The customer holds the primary key of the company success: a good business process will nurture perception and emotional connection of the product which will drive strong product engagement and loyalty based on trust, and finally translate into the value of a business. System approach is needed to gain understanding on how to access the knowledge of those aspects and gaining optimal customer relationship.

CRM are a **system model** to manage customer relationship, relationship is very abstract unless viewed through the lenses of at least their **experience-retention-satisfaction context**. Positive experience will initiate a good relationship, and if this relationship is managed in such a way that will exceed the expectation of the customer, satisfaction will induce customer retention and loyalty.

This make nature of relationship complex, good relationship with customer require much more than single great purchasing moment, it is a forged mutually beneficial connection based on trust. Below is the cycle of CRM. Based on consolidated consumer data, the CRM will determine its targeted customers segment, for each the interaction will be optimized or even personalized. From which the management can re-evaluate the strategy and response and align systems or innovate further from it. The cycle can restart again for customer data to be consolidated even further.

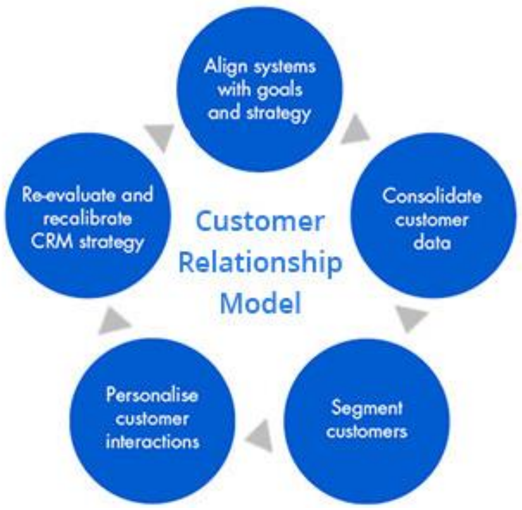

Align systems with goals and strategy
Consolidate customer data
Segment customers
Personalise customer interactions
Re-evaluate and recalibrate CRM strategy
Customer Relationship Model

Figure 1 : Cycle of CRM

Name: Maulfi Alfansuri, DE BATCH 03

3. Conceptual Formation.

The literature taken here are based on some works of O'Reilly publication and reference architecture from Sang et al [2]. Despite relatively old, some concepts in them keep relevant (or even more advanced) than the current trend of data engineering practice, but the limitation must be put in mind. for example: there are seems to higher priority to incorporate ELT paradigm in both sources, due to its focus weighted on data governance process before data entering data lakes system, there are no solutions to address common problem occured ELT such as validation and other data governance practice.

Meanwhile the distinction fast-big data could be analogous to stream-batch processing system, which still very relevant to our case. In [1] distinction, fast data make a more relevant system and not contradicting from the course taught by Digital Skola. The main tenet in data architecture design lie on the **data value in timely manner**, data value diminished each time, therefore real-time processing system is highly preferred.

We cannot forget to involve the data source/application in the architecture system. Also, the cost of each alternative must be calculated prior, this could be done with cost-allocation analysis.

3.1.Data Architecture Design Principles.

Basic process of data architecture in general can be boiled down on 4 stages:

- **Ingestion**

This is the front-end part of big data system, where the data is processed from the interface, data are generated from application and passed down to downstream system. The state change of this application is what called 'event', this event generated by sensor and the downstream process could enrich the data, append it into log or any sub process needed for real time processing or analytic.

- **Contextual processing**

Process must be contextual, it must interact with input from other output in a timely manner: where the data processed geographically, with whom it generated in the interface by cross-checking with many feeds, the interpretation of data generated by sensor could be much more relevant.

- **Realtime-Analytic**

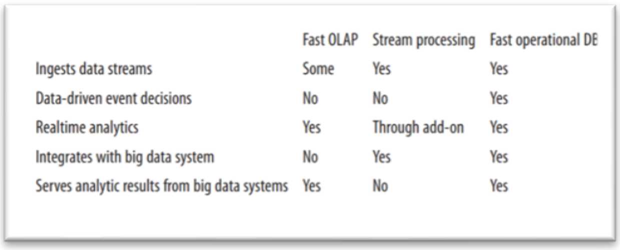
Decision must be made real-time; some system require automatic real-time moderation system to mitigate certain risk. rerouting powerline based on certain identification of last minutes trending before overload in smart grid is one of the examples.

Integration of big-fast data system is also required; consideration should be made between both system: impedance, reliable transfer (persistence and buffering), pre-processing of the data to be readily used when arrive in data lake.

The last, role of data analytics department such as BI or data scientists in fostering this real-time decision making, they are not just given responsibility in make managerial report, but what is more important is to make those report to be actionable. There are two alternatives mentioned in the [1]: make reports consumable or make it embedded in operational system. The latter is preferred.

3.2.Architectural approaches.

Architectural approaches can take 2 ways: fast-big data architecture, reference architecture (based on real-world cases). In the former approaches it contains several options, such as: Fast OLAP systems, Stream Processing Systems, or Operational Database System. From the design principles that has been covered in previous subchapter, we could tabulate each option against each point fulfilment.



	Fast OLAP	Stream processing	Fast operational DE
Ingests data streams	Some	Yes	Yes
Data-driven event decisions	No	No	Yes
Realtime analytics	Yes	Through add-on	Yes
Integrates with big data system	No	Yes	Yes
Serves analytic results from big data systems	Yes	No	Yes

Fast OLAP are great for report generating but unable to support a decision based on individual events as it arrives on system, the 'Fast' part only lies on the real-time analysis. Stream processing systems is data architecture well-suited to environment where high precision of time processing required, but it cannot interact with other events because it doesn't maintain data state, therefore any valuable context is lost, adding real-time analytics function in this architecture will likely cause a bottle neck to other components. The operational database

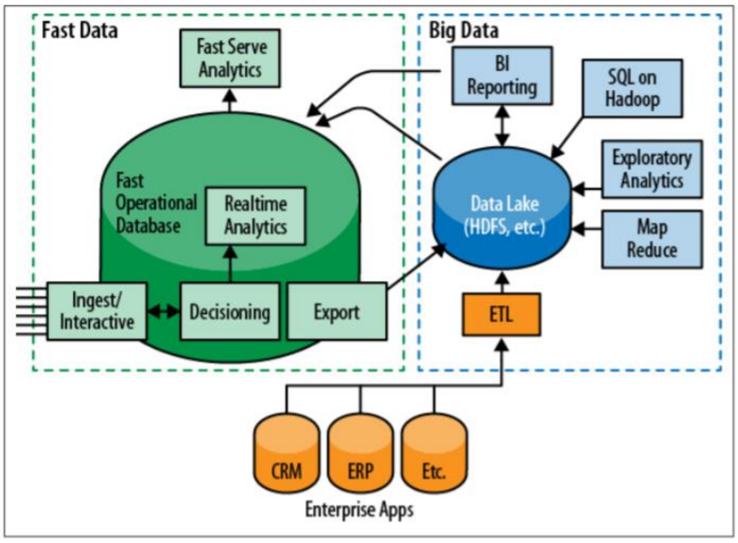
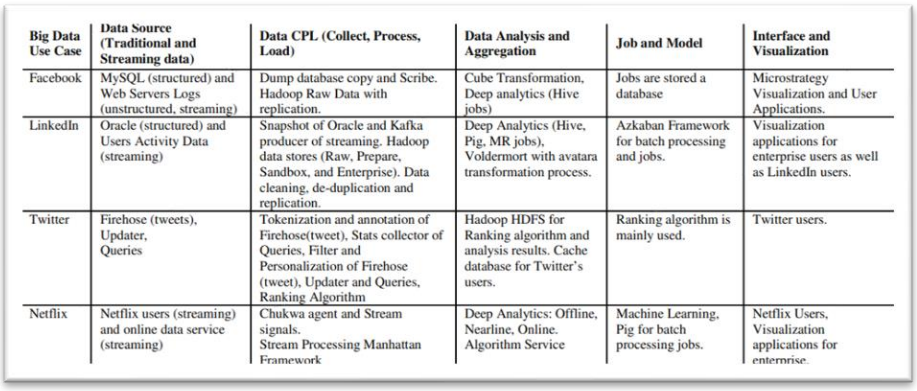


Figure 2 : Fast - Big Data General Architecture

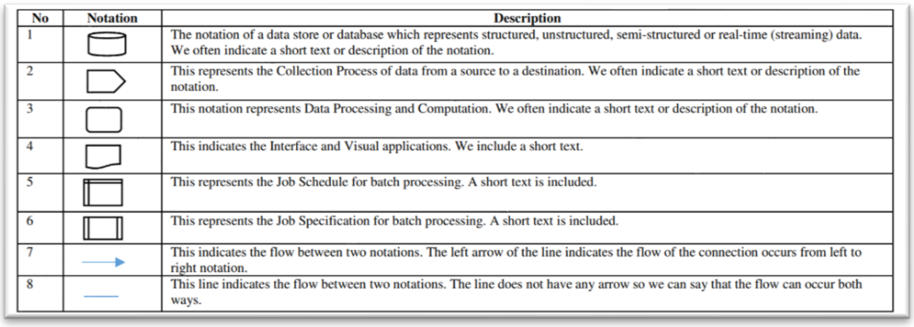
Name: Maulfi Alfansuri, DE BATCH 03



Big Data Use Case	Data Source (Traditional and Streaming data)	Data CPL (Collect, Process, Load)	Data Analysis and Aggregation	Job and Model	Interface and Visualization
Facebook	MySQL (structured) and Web Servers Logs (unstructured, streaming)	Dump database copy and Scribe. Hadoop Raw Data with replication.	Cube Transformation, Deep analytics (Hive jobs)	Jobs are stored a database	Microstrategy Visualization and User Applications.
LinkedIn	Oracle (structured) and Users Activity Data (streaming)	Snapshot of Oracle and Kafka producer of streaming. Hadoop data stores (Raw, Prepare, Sandbox, and Enterprise). Data cleaning, de-duplication and replication.	Deep Analytics (Hive, Pig, MR jobs), Voldermort with avatara transformation process.	Azkaban Framework for batch processing and jobs.	Visualization applications for enterprise users as well as LinkedIn users.
Twitter	Firehose (tweets), Updater, Queries	Tokenization and annotation of Firehose(tweet), Stats collector of Queries, Filter and Personalization of Firehose (tweet), Updater and Queries, Ranking Algorithm	Hadoop HDFS for Ranking algorithm and analysis results. Cache database for Twitter's users.	Ranking algorithm is mainly used.	Twitter users.
Netflix	Netflix users (streaming) and online data service (streaming)	Chukwa agent and Stream signals. Stream Processing Manhattan Framework	Deep Analytics: Offline, Nearline, Online. Algorithm Service	Machine Learning, Pig for batch processing jobs.	Netflix Users, Visualization applications for enterprise.

Figure 3 : Reference Architecture of Big Data use case, Sang.et al

Reference architecture based on Sang et al work can also help the shortcoming of architectural detail in fast-big data architecture, it divides the architecture flow to be in 5 domains: Data Source – CPL (ETL) – Analysis & Aggregation – Job & Model – Interface and Visualization. These details help completing the system when many architectures often overlook to specify job& model or visualization. In figure 4 and 5 we can see the notation used for reference architecture and its depiction in Amazon Service use case.



No	Notation	Description
1	⛁	The notation of a data store or database which represents structured, unstructured, semi-structured or real-time (streaming) data. We often indicate a short text or description of the notation.
2	⬠	This represents the Collection Process of data from a source to a destination. We often indicate a short text or description of the notation.
3	▢	This notation represents Data Processing and Computation. We often indicate a short text or description of the notation.
4	▭	This indicates the Interface and Visual applications. We include a short text.
5	▤	This represents the Job Schedule for batch processing. A short text is included.
6	◫	This represents the Job Specification for batch processing. A short text is included.
7	→	This indicates the flow between two notations. The left arrow of the line indicates the flow of the connection occurs from left to right notation.
8	—	This line indicates the flow between two notations. The line does not have any arrow so we can say that the flow can occur both ways.

Figure 4 : Reference architecture notation

It also must be noted that the notation of stream processing isn't exist. It will be added later in the visualization chapter.

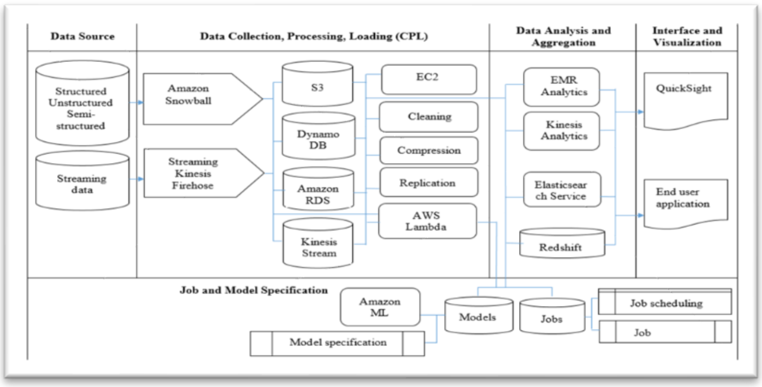


Figure 5 : Example of Amazon Service architecture

Name: Maulfi Alfansuri, DE BATCH 03

3.3.Services and Microservices Architecture

Microservice are type of data architecture derived from services architecture (SoA), the main advantage of microservices architecture is its independency between the application components, in which the logic of application is broken down into units and loosely coupled as opposed with SoA. Also, advent of container technology, this architecture could be achieved, and degree of complexity is much more reduced.

Both types of architecture are based on ***service*** as the primary component, the service can be in the form of order service. This classification system makes it very compatible for the implementation in CRM type business. Both also classified as *distributed architecture* with specified remote access protocol, meaning that each service interacts through certain interface such as API or Middleware.

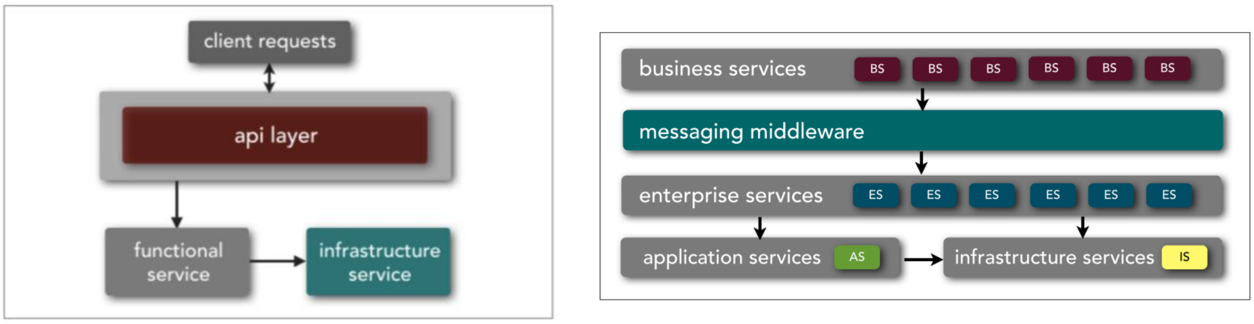


Figure 6: Microservices & SoA Taxonomy

The taxonomy concept is helpful to distinguish between them, at architecture pattern (abstract) level the classification used is functional services (business operations/functions support) or infrastructure services such as auditing, logging and monitoring. In my opinion the explanation of classification distinction from O'Reilly source is problematic, it said that the characteristic depiction will be focused on architecture pattern level, but 'business services' in SoA taxonomy are exclusive term used for implementation level. To reconcile into acceptable understanding is that *type*(pattern) and *role*(implementation) is a different concept. The 'business services' etc is just a type of process, for example *Process Trade* are *type*, meanwhile *Insert Customer* are *role*, the former instance is abstract while latter concrete.

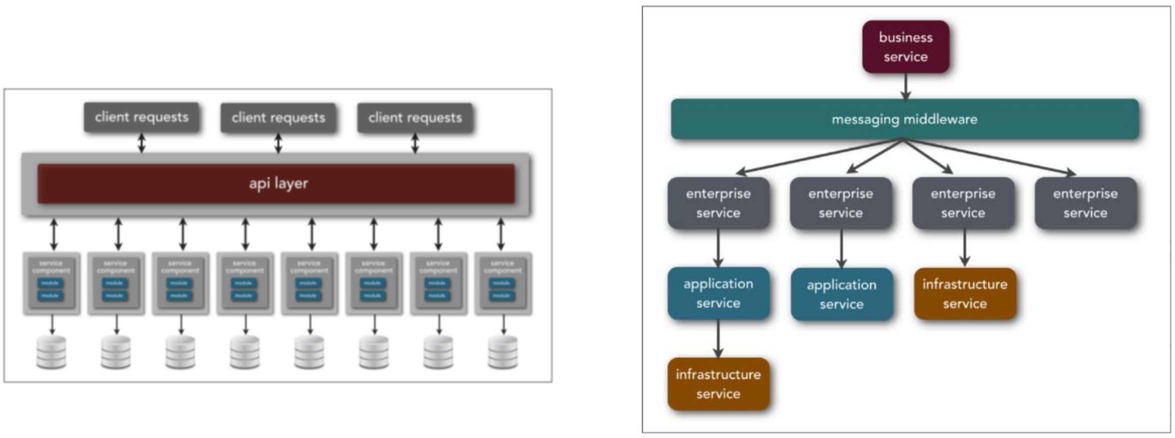
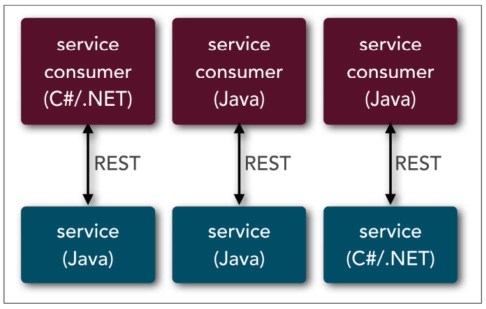


Figure 7: Microservices & SoA Topology

Another concern of difference is the dominant characteristic of *share-as much as-possible* (orchestra) and *share as little as possible* (choreography), in SoA and microservices consecutively. As can be seen in figure 7, APIs layer in microservices case is used exclusively for different services compared to messaging middleware which are used by different enterprise services at the same time. But it is a good practice to violate those distinction by employing orchestration in dominant microservices architecture should it be necessary.

In addition, there are different interoperability characteristic between them that made different characteristic for each in easiness of maintenance, communication etc. microservices can only use same remote access protocol type for each service, it doesn't possess capability of message enhancement-transformation like in SoA that enable contract decoupling and multiple data/format processing, the reason of why that is the case is not clear for the writer. This is one of capability that make SoA better suited for complex and big project. Employing SoA into relatively simple system also a waste of extensive resources.



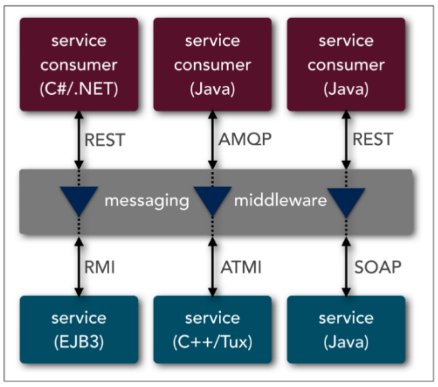
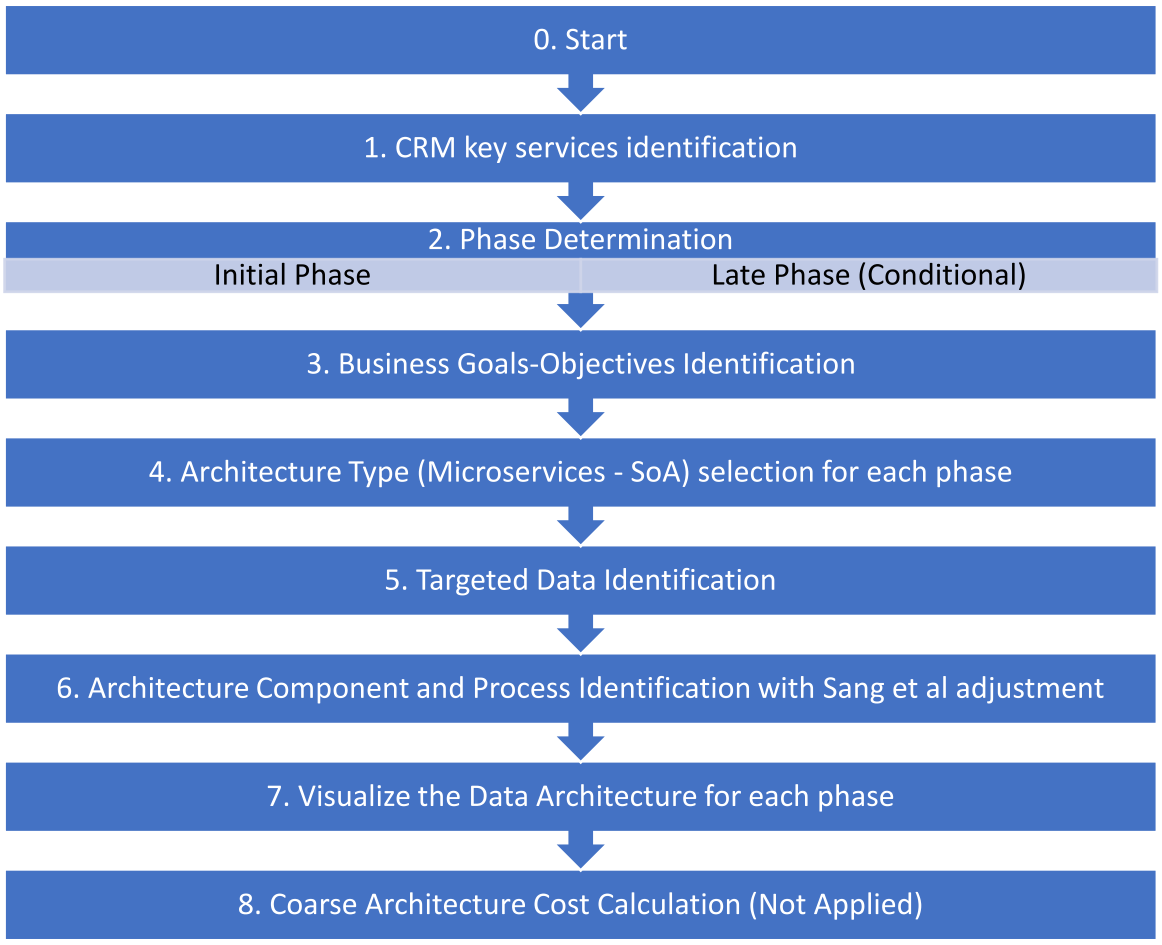


Figure 8: Protocol-aware heterogenous interoperability (Microservices) & Protocol-agnostic heterogenous interoperability (SoA)

Name: Maulfi Alfansuri, DE BATCH 03

4. Designing Architecture for CRM Business

After we have covered the design principles through fast-data architecture model, and acquainted with specialization of microservices, the next step is how do we select the specific actionable architectures. While cost analysis is important, the true form of architecture in data analytic platform especially can only be achieved in writer opinion through adapting the determined business strategy-value framework. The specific design process follows the following flowchart:


0. Start
1. CRM key services identification
2. Phase Determination
Initial Phase
Late Phase (Conditional)
3. Business Goals-Objectives Identification
4. Architecture Type (Microservices - SoA) selection for each phase
5. Targeted Data Identification
6. Architecture Component and Process Identification with Sang et al adjustment
7. Visualize the Data Architecture for each phase
8. Coarse Architecture Cost Calculation (Not Applied)

For the next subchapters, we will explain the workflow steps into several groups that corresponding to each subchapter, with little to no intersection: Step 0 - 2 will be explained in chapter 4.1, "CRM key services identification", how key services identified from looking on the real-world use case of business strategy of company in the same business type. While step 3 - 6 will be detailed briefly in chapter 4.2, component and components will be clearly identified. Visualization of the data architecture will be displayed in 2 subchapter 4.3.1 and 4.3.2 for each phase of business we have chosen. And lastly in the chapter 4.4, the coarse cost calculation of each architecture will be displayed in cost table, from pricing calculator available in Google Cloud Platform.

Name: Maulfi Alfansuri, DE BATCH 03

4.1. CRM Key Services Identification

How we make our case on determining business strategy is critical, one of them is through adapting business strategy of other business chain, which in our case is Pizza restaurant, we select the strategy employed by Domino's pizza. The lists are as follows.

- Product development	: Geographic-culturally oriented.
- Marketing	: Family-focused; discount, cashback and promotion program.
- Customer response	: Feedback system on menu, pricing, services.
- Payment method	: Uniform cost on any platform, through
- Customer notification	: Email – SMS based campaign.
- Social media	: Regular campaign.

While the exact form of this strategy won't be expected, the unique flow and interaction each strategy component represented by the strategy hoped to be last longer. What need to be remembered are the difference of company size at the time of digital transformation, while Domino's pizza transformed only happened in the last decades, our company design is started after 2021, which could be late in gaining technological competitiveness.

Strategy the Domino's pizza taken is very CRM oriented, since their digital transformation in 2009 by collaborating with HS2: the global market shares around 50%, share price rise more than 6000%, because they make customer feedback their centre of business input. When the covid outbreak occurred, they employ Domino's car side delivery service with the location of customer preference.

How domino come and developed into this existing business is also a big concern on how to formulate strategy business in the initial phase, after ***key services*** are clearly laid out, the arrangement of architecture can follow from that. It could be any business core values, time-bounded plans, risk perception methods or anything. Here are the several key points of those transformation:

- Technology-based vision: Technology is the absolute condition for company to innovate with the most efficient success, domino commitment "Tech companies that happen to sell pizzas" is not in the selling of food. But in how they use technology to use it in all their facet.
- Comprehensive A/B testing to whole system: Domino wasn't just experimenting with any new technology they found. building a stringent data-driven testing procedure was what it takes to build a strong foundation for the innovation
- Reciprocate communication platform: For the effective communication between customer and company to happen, they need to have a powerfully responsive communication platform for brutal honest feedback like Think Oven, Successful business absolutely needs this to develop high-level insight of the customers profiling.
- Flexibility and intra-services collaboration: The success of domino also rely on the inclusiveness of ordering platform; at larger stage, domino order platform is not just limited to their own application (AnyWare). Customer can order the product easily from different devices-software such as: Apple TV, Google Home, Amazon Echo, Ford Sync, SMS, as well as social media such Tweets, Slack and Facebook Messenger.

Architecture of our case are defined within their phase, for convenience long-term strategy we divide the phase into initial and later phase.

Name: Maulfi Alfansuri, DE BATCH 03

4.2. Selection and identification of architecture node-process.

Based on coverage of previous chapter especially in 4 Domino key business *services*, we could devise the architecture process for initial and late phase to be as follows (with additional tweaks): **The fundamental of this business is undoubtedly the data-driven innovation process based on the stringent A/B testing system**, this must be accomplished accordingly for both phases. For initial phase, the distinguishing features are powerful communication platforms, it must

The features of the architecture must consist of container lifecycle management with automation of maintenance operation, networking security, and cloud movement possibility. Therefore, it is decided that the data architectures are separated in evolving way, given the conditions are true:

- Initial phase: In this phase, the targets are set within the maximum 3-year timeline; at some point before the end of timeline, the acquisition of customer should be at minimum 100 k customers*order per unit of location, with the customer retention rate at least 30%, with turnover rate less than 0.5%. This target requires dominant fast data combined with microservices architecture due to little complexity prospect in this phase. To enable customer centric services and gaining data insight, **the main objective in this phase is building data architecture for powerful two-way communication platform and shaping the path for late phase.**

  To satisfy that objective, we must analyse and evaluate customer experience and satisfaction along the way and devise the necessary data, but we must not cross into late phase scope should there are still no good foundational consolidation of platform system or minimal target is not received. We can divide the required data based on streaming-batch dichotomy principle. For streaming, there are several data required for event generated by available communication platform: service response time, service availability and request time, data latency, and time consumption of timeout resolution.

  To analyse the trend and insight of customer experience and satisfaction, we need to aggregate the collected data for future long-term analytical purpose, which also belong to batch processing. The required data are as follow: customer profile, historical quantity and type of ordered products, contextual ordering data (time – social), ordering platform.

  Data generated by website and mobile apps are mediated by Kafka APIs as gateway, transform into structured format with Hive and MySQL, and later loaded into warehouse with Apache Sqoop, and using Logstash for monitoring microservices deployment.

o Data Sources	: (Website Apps, Mobile Apps.)
o CPL Mode	: ETL (Kafka - Spark, Hive-MySQL, Sqoop)
o Analysis and Aggregation	: Google Analytics and HBase.
o Job – Model Spec	: Airflow (Scheduler) and Apache Spark (ML)
o Interface – Visualization	: User App, SAP Lumira.

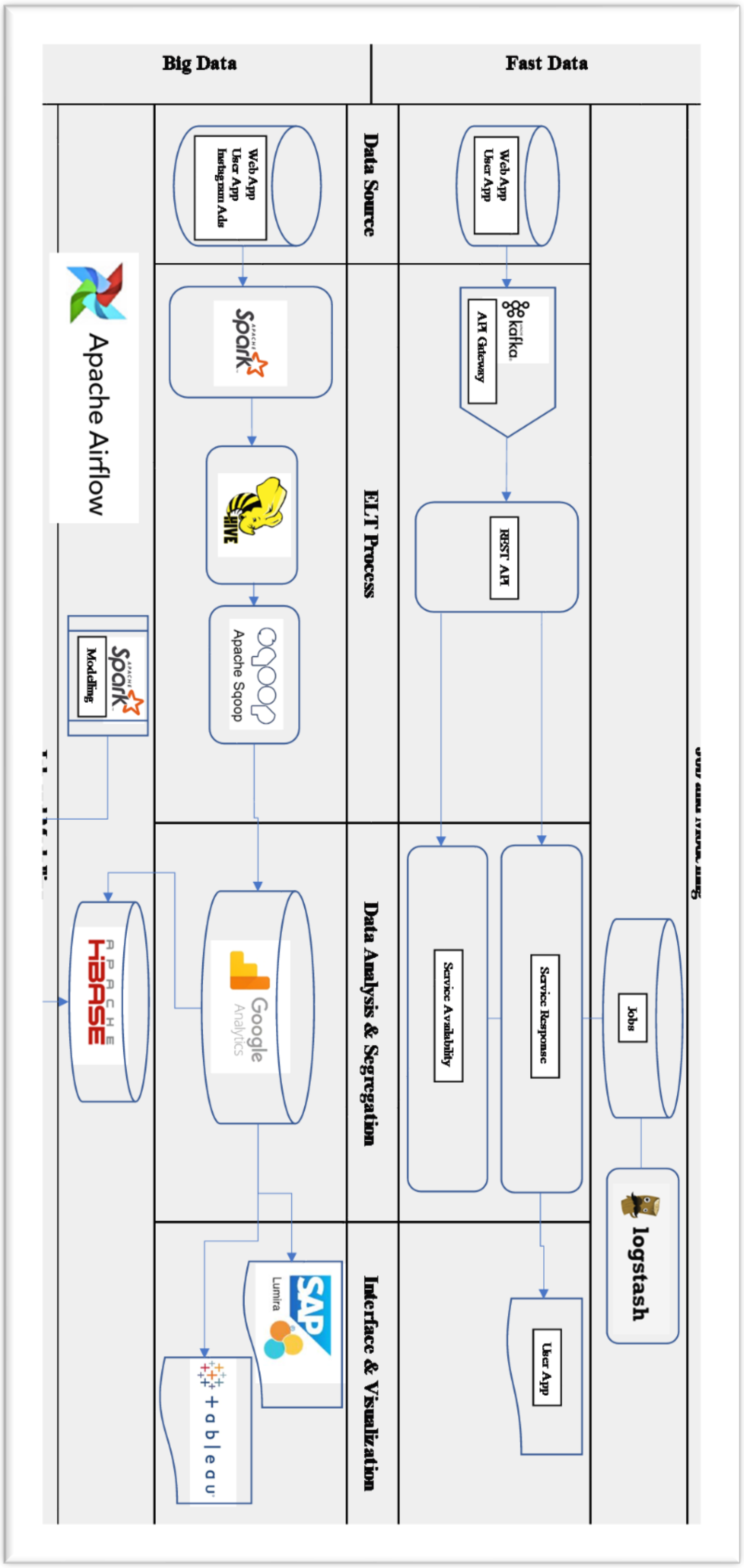
- Late phase: Should the condition in initial phase are met, the late phase will require incorporation of big-data attributes, the transition will take in 3 months in the final quarter of 3rd year. In addition, to derive contextual knowledge of this type, it is advised that cluster analysis methods are needed. Therefore, the proper architecture of this phase will be SoA. **The main objective in this phase is building a comprehensive intra-platform collaboration architecture.**

  Platforms taken into consideration range from 15-20 units, encompassing ordering, communication and delivery platforms. Events generated by both user-website apps and collaborating platform are sent as business services events, which processed by Kafka as middleware and. The difference from initial phase is the inclusion of stream analytic to manage multiple platforms, with Kubernetes and Docker to orchestrates services inside SoA.

Name: Maulfi Alfansuri, DE BATCH 03

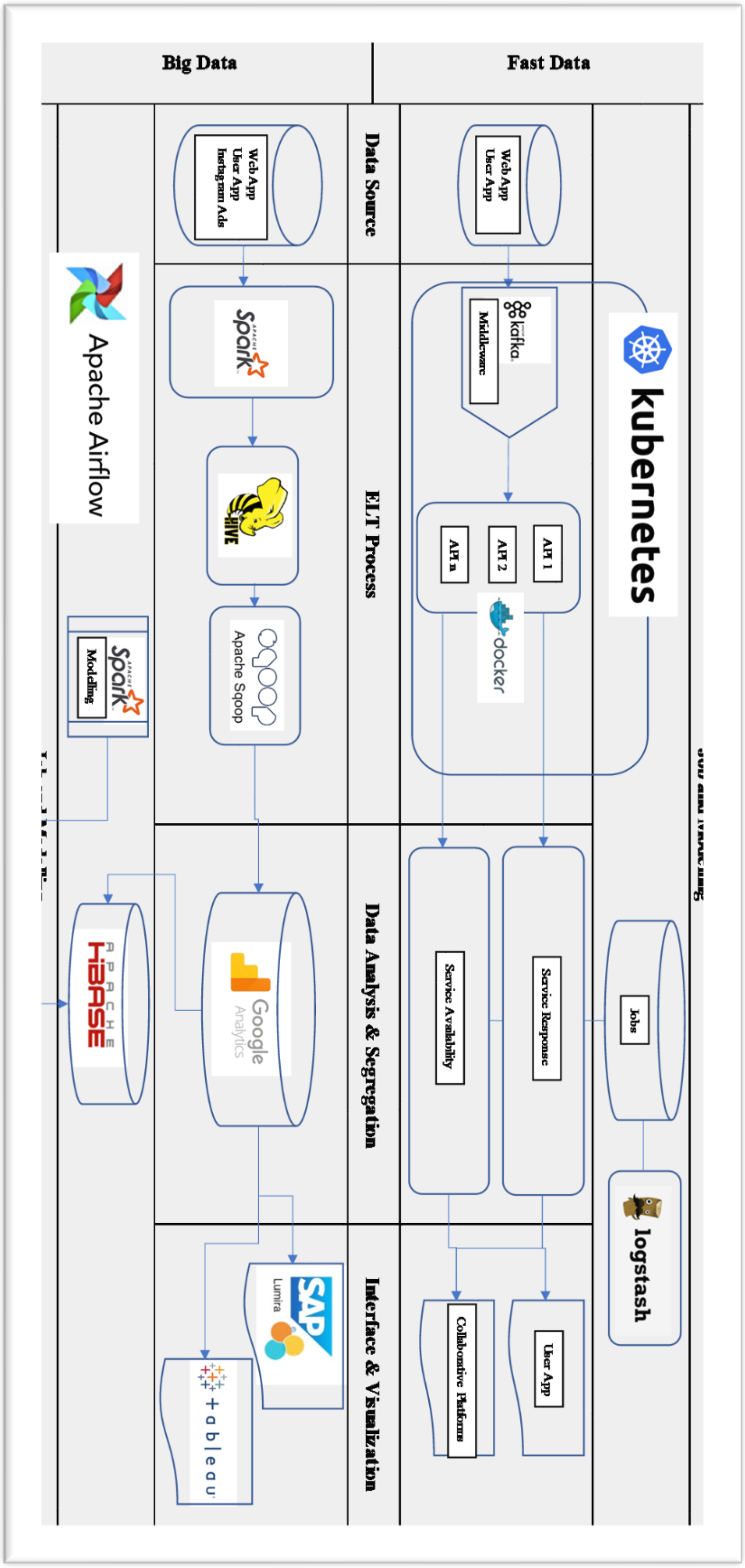
4.3. Visualization of data architecture

4.3.1. Visualization of data architecture in the initial phase.



Name: Maulfi Alfansuri, DE BATCH 03

4.3.2. Visualization of data architecture in the late phase.



Name: Maulfi Alfansuri, DE BATCH 03

5. Conclusion

Designing data architecture is not just about engineering technicality, the workflow must be based on the relevant business strategy with big-fast data paradigm. Due to limitation of deriving key services from deductive reasoning, we try to develop the key services in CRM framework based on real-world use case of business services, which in this context is Domino's pizza.

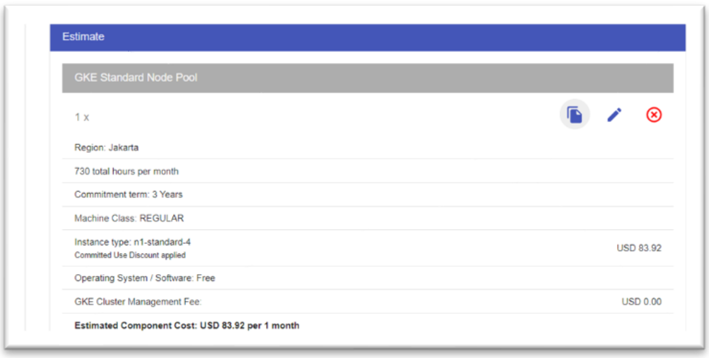
The development of the architecture was done through researching the design principles in fast-big data paradigm, then we worked on the workflow structure from reference architecture based on Sang works, and investigation of relevant distinctions in microservices - service oriented architecture for fast-big data concept to gain compatible architecture based on capability-characteristic and business phase.

6. References

- Fast Data Enterprise Data Architecture [https://www.oreilly.com/data/free/files/fast-data-enterprise-data-architecture.pdf]
- A Reference Architecture for Big Data Systems [https://core.ac.uk/download/pdf/132195727.pdf]
- Microservices vs Service-Oriented Architecture [https://www.oreilly.com/data/free/files/fast-data-enterprise-data-architecture.pdf]
- "Guide to Customer Relations: Definition, Benefits, Strategies & More" [https://www.proprofsdesk.com/blog/customer-relations/]
- "Microservices Architecture and Design" [https://medium.com/@cfryerdev/microservice-architecture-design-2ac7eaae532]
- "8 Excellent Examples of Customer Relationship Management" [https://www.revechat.com/blog/customer-relationship-management-examples/]
- Domino's digital transformation [https://thestrategystory.com/2020/07/11/dominos-digital-transformation/]
- Delivering Modern Apps (and Pizza) with Kubernetes [https://news.vmware.com/technologies/cloud-native-application-development-dominos]
- "3 types of CRM and how to choose the best one for your business" [https://www.zendesk.com/blog/3-types-crm-everything-need-know/]
- How Domino's transformed into An E-commerce Powerhouse Whose Product is Pizza[https://www.forbes.com/sites/kylewong/2018/01/26/how-dominos-transformed-into-an-ecommerce-powerhouse-whose-product-is-pizza/?sh=a4a8e1f7f761]
- "Extract, Transform, and Load (ETL) at scale" [https://docs.microsoft.com/en-us/azure/hdinsight/hadoop/apache-hadoop-etl-at-scale]

Name: Maulfi Alfansuri, DE BATCH 03

7. Appendices



Estimate

GKE Standard Node Pool

1 x

Region: Jakarta

730 total hours per month

Commitment term: 3 Years

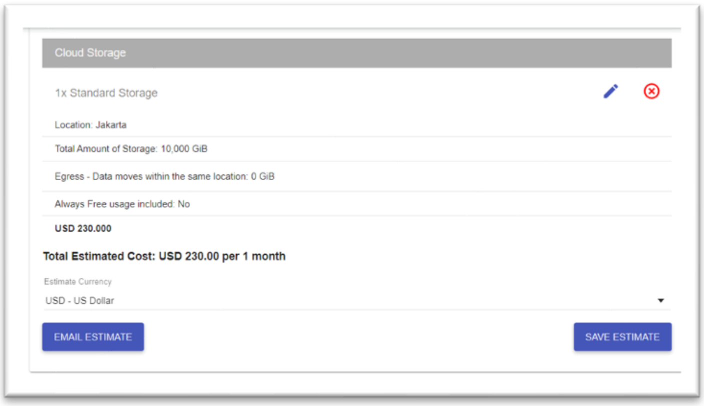
Machine Class: REGULAR

Instance type: n1-standard-4 — USD 83.92
Committed Use Discount applied

Operating System / Software: Free

GKE Cluster Management Fee: — USD 0.00

Estimated Component Cost: USD 83.92 per 1 month



Cloud Storage

1x Standard Storage

Location: Jakarta

Total Amount of Storage: 10,000 GiB

Egress - Data moves within the same location: 0 GiB

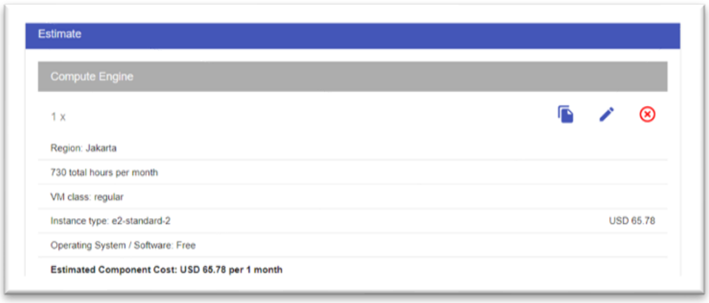
Always Free usage included: No

USD 230.000

Total Estimated Cost: USD 230.00 per 1 month

Estimate Currency

USD - US Dollar



Estimate

Compute Engine

1 x

Region: Jakarta

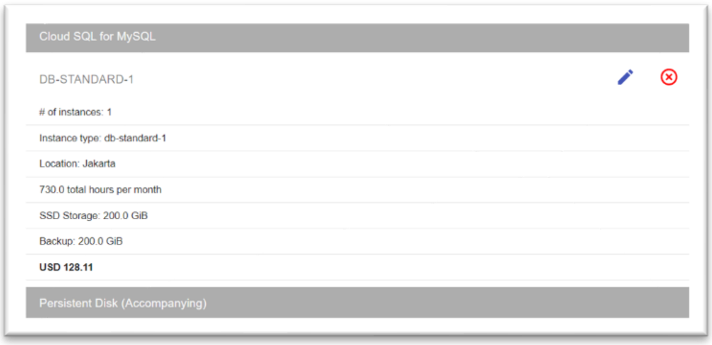
730 total hours per month

VM class: regular

Instance type: e2-standard-2 — USD 65.78

Operating System / Software: Free

Estimated Component Cost: USD 65.78 per 1 month



Cloud SQL for MySQL

DB-STANDARD-1

# of instances: 1

Instance type: db-standard-1

Location: Jakarta

730.0 total hours per month

SSD Storage: 200.0 GiB

Backup: 200.0 GiB

USD 128.11

Persistent Disk (Accompanying)