Lambda vs Kappa Data Processing Architecture

Lambda and Kappa are both well-known data processing architectures. But which of the two suits our project better?

# What is meant by a data processing architecture?

A data processing architecture is a system designed to collect, process and store data. (Dinc, 2024)

# What is the Lambda data processing architecture?

Lambda architecture is designed to handle both real time and batch data. Both have their own use cases. Here is an example of how it would go in practice:

1. The backend needs to save data.
2. The backend sends the data to both the speed layer and the batch layer.
3. **Speed Layer**: process the data and combine it with short term data (e.g. last 24 hours, last 12 hours last hours etc). **Batch Layer**: process the data and combine it with all historical data.
4. Now you can query the **Speed Layer** for recent data which is faster but probably less accurate since it contains less data. You can also query the serving layer by for example asking it data from a specific day or month in the past, this is not as fast as the speed layer but more accurate

(Waehner, 2021)

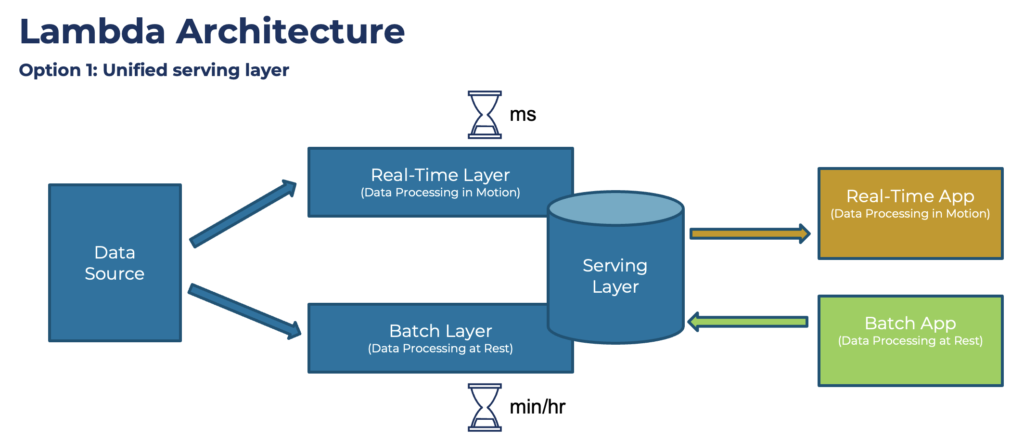


Figure 1: Lambda architecture variation 1

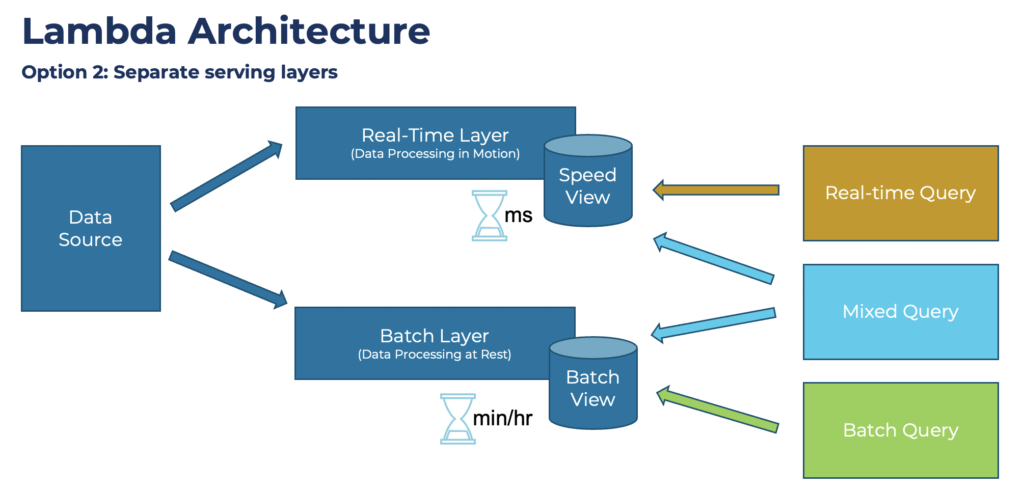
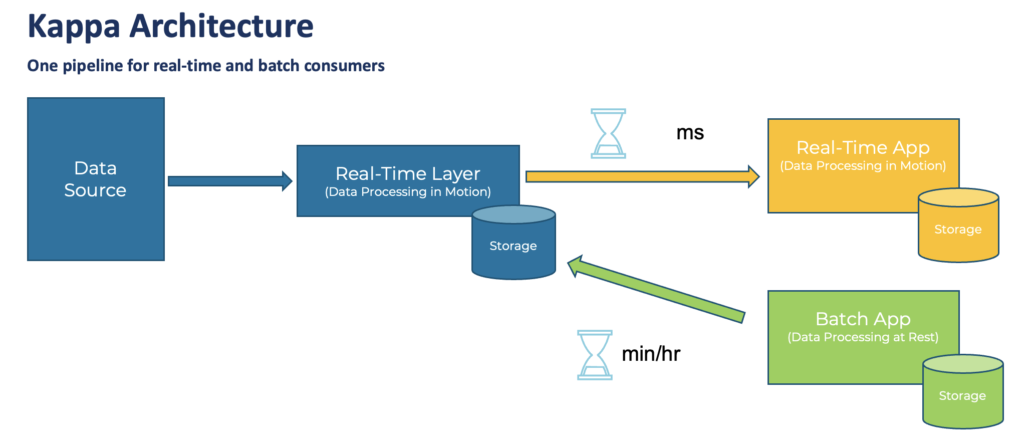


Figure 2: Lambda architecture variation 2

# What is the Kappa data processing architecture?

The main difference between Lambda and Kappa is that the Kappa architecture only has 1 processing layer. Both historical and real-time data are stored in the same layer.



# Which is most suitable for our application?

Based on our non-functional requirements we can choose which data processing architecture would be more suitable for our application. Here are the non-functional requirements ranked on importance in our project: ***Scalability****,* ***Security****,* ***Data Privacy & Confidentiality***, ***Performance***, ***Reliability & Availability****,* ***Maintainability (modularity)***, ***Disaster Recovery***

Here is an overview on how the architectures cover the quality requirements (Owczarek, 2023)

## Lambda Architecture

1. **Scalability**: Lambda architecture is designed to handle large volumes of data and can scale horizontally to meet business needs.
2. **Fault-tolerance**: The architecture is built to be fault-tolerant, with multiple layers ensuring reliable data processing and storage.
3. **Flexibility**: It can handle a wide range of data processing workloads, including both historical batch processing and streaming.
4. **Complexity**: The architecture introduces complexity due to its multiple layers and systems, making setup and maintenance challenging.
5. **Error** **Handling**: It may lead to errors and data discrepancies because of duplicated workflows, complicating debugging.
6. **Lock-in**: There may be challenges related to architecture lock-in, making data migration difficult.

## Kappa Architecture

1. **Simplicity**: Kappa architecture simplifies the data processing pipeline by using a single system for both batch and stream processing.
2. **Scalability**: Kappa is designed to be scalable, allowing for efficient processing of large amounts of data in real time.
3. **Flexibility**: It provides a flexible system that can handle various data processing workloads using a single technology stack.
4. **Fault-tolerance**: Kappa architecture is also fault-tolerant, relying on a stream processing engine designed for reliability.
5. **Complexity**: While simpler than Lambda, Kappa architecture can still be complex for those unfamiliar with stream processing frameworks.
6. **Cost**: Infrastructure costs may be high if not set up properly, particularly for big data storage in event streaming platforms.
7. **Ease** **of** **Migration**: Kappa architecture allows for easier migrations and reorganizations due to its single pipeline approach.

|  |  |  |
| --- | --- | --- |
| **Quality Requirement** | **Lambda Architecture** | **Kappa Architecture** |
| **Scalability** | Yes | Yes |
| **Fault-tolerance** | Yes | Yes |
| **Flexibility** | Yes | Yes |
| **Complexity** | High | Moderate |
| **Error Handling** | Yes | Moderate |
| **Lock-in** | Yes | No |
| **Cost** | Moderate to High | Moderate to High |
| **Ease of Migration** | No | Yes |

|  |  |
| --- | --- |
| **Quality Requirement** | **Best Fit** |
| **Scalability** | Both |
| **Security** | Kappa Architecture |
| **Data Privacy & Confidentiality** | Kappa Architecture |
| **Performance** | Kappa Architecture |
| **Reliability & Availability** | Kappa Architecture |
| **Maintainability (Modularity)** | Kappa Architecture |
| **Disaster Recovery** | Kappa Architecture |

Based on the comparison between kappa and lambda on the quality requirements, Kappa should be the better choice in our project.

# Bibliography

Dinc, S. (2024, March 5). *Data Processing Architectures*. Retrieved from Medium: https://medium.com/@seckindinc/data-processing-architectures-3b20fe349ede

Owczarek, D. (2023, April 24). *Lambda vs. Kappa Architecture. A Guide to Choosing the Right Data Processing Architecture for Your Needs*. Retrieved from Nexocode: https://nexocode.com/blog/posts/lambda-vs-kappa-architecture/

Waehner, K. (2021, September 23). *Kappa Architecture is Mainstream Replacing Lambda*. Retrieved from kai-waehner: https://www.kai-waehner.de/blog/2021/09/23/real-time-kappa-architecture-mainstream-replacing-batch-lambda/