First Round of Feedback on Job's and Silvester's Designs

-Lewis Injai

Peer Review notes

Silvester

Strong points

- There is data duplication is evident in the server cluster ecosystem
- There is a single system of record

How can we perform service dependency inversions on the Microservices ecosystem? There are cases where if you look at the design, we might have a scenario where the uptime of one service is completely dependent on the uptime of another.

Sharing data between microservices can be quite a daunting task

It's good that you have identified that the system needs to be read heavy. That means that data will be shared all across the different services. In a scenario like that, every piece of data is normally owned by a single service. That service is usually the conical system of record for that data and every other copy is a read-only, non-authoritative cache. From your own perspective, apart from performing a simple synchronous lookup, what is the best approach to sharing data between the different services?

Job

Strong points

- Introduction of load balancers on multiples points

On your recommendations:

Yes, from the initial design we introduced Kafka to help us streamline the data

On the server side/ backend ecosystem we do have a data analysis factored in.

Regarding your question about the categorization of data, I was just sharing with Silvester this. Since the system is read-heavy, we will have a single system of record. Data can be owned by a single service, that service will then act as the conical

system of record for that data, but since this data has to be distributed across different services it will only exist as a read-only non-authoritative cache across all other services. Now the problem which you can help us look into is, what is the best approach for sharing this data.