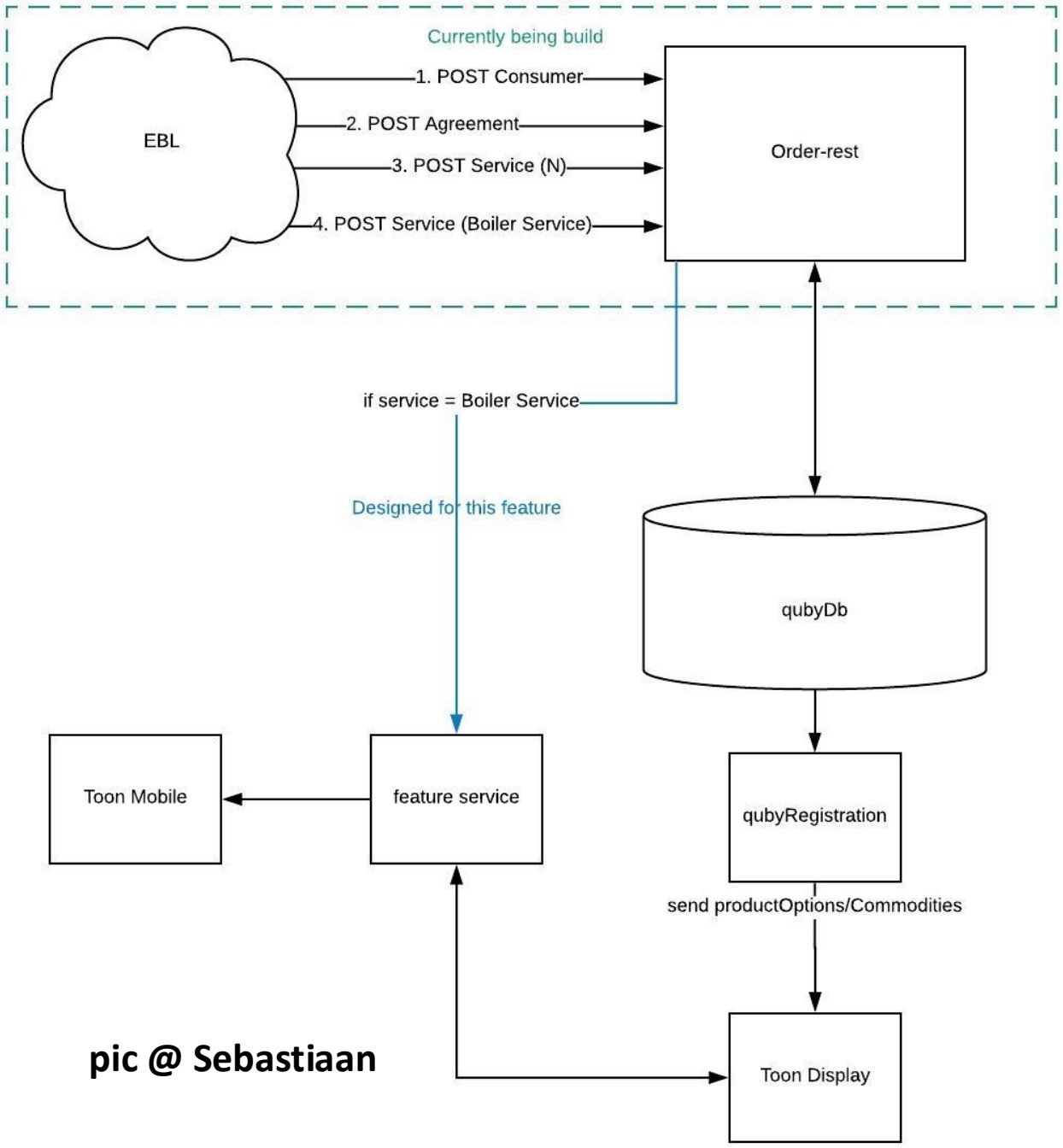
Event Sourcing

Introducing Change Data Capture with Debezium



pic @ Sebastiaan

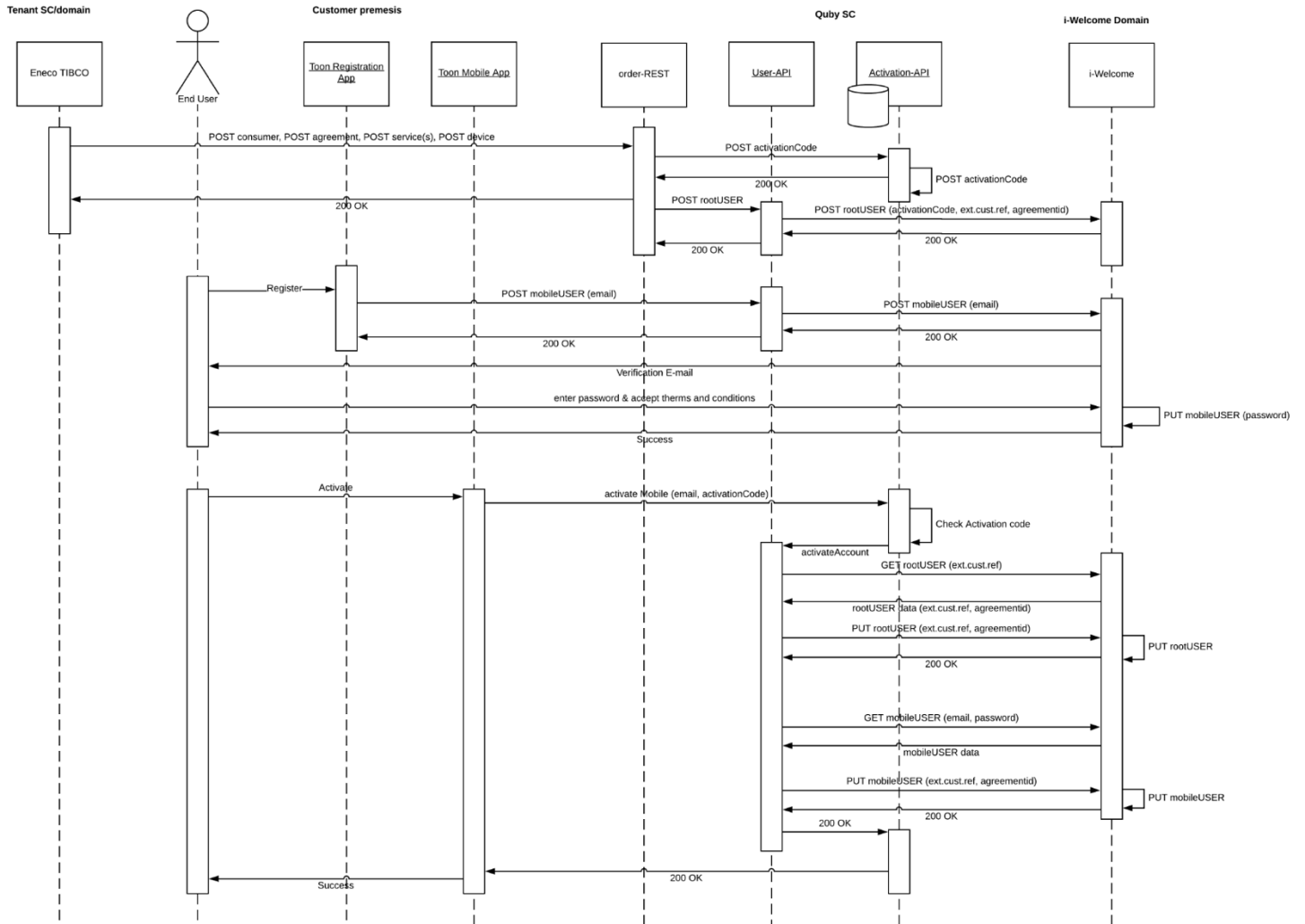
- With Feature service and Order-rest
- When an order is created, the order-service has to create an entry in the feature service, if the service is enabled.
- When the order is updated, the order-service has to propagate the updates to feature service
- Updates to feature service is not transparent to the client who is calling the order-service
- There are lot of What if Questions?

What?

- If the create order succeeds and call to feature-service fails, so no data in the feature service
- About the data consistency between order-service and feature-service
- If the feature-service call fails, should the order creation also be roll backed and send a failure response to calling client?
- If the order-creation succeeds and feature service fails, use re-try to try again the feature service, how to keep track of this?
- If the order-service has to call another service XYZ along with feature service call, things get complicated.

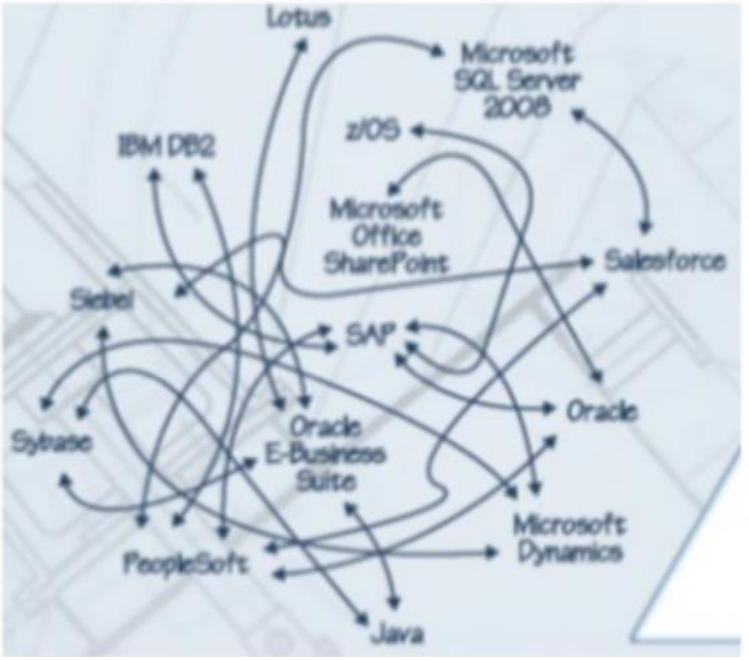
With iWelcome coming in,

- Order service becomes more complicated,
- Order service-> User API ->iWelc
- Order service-> Activation API
- Order service-> Another service?

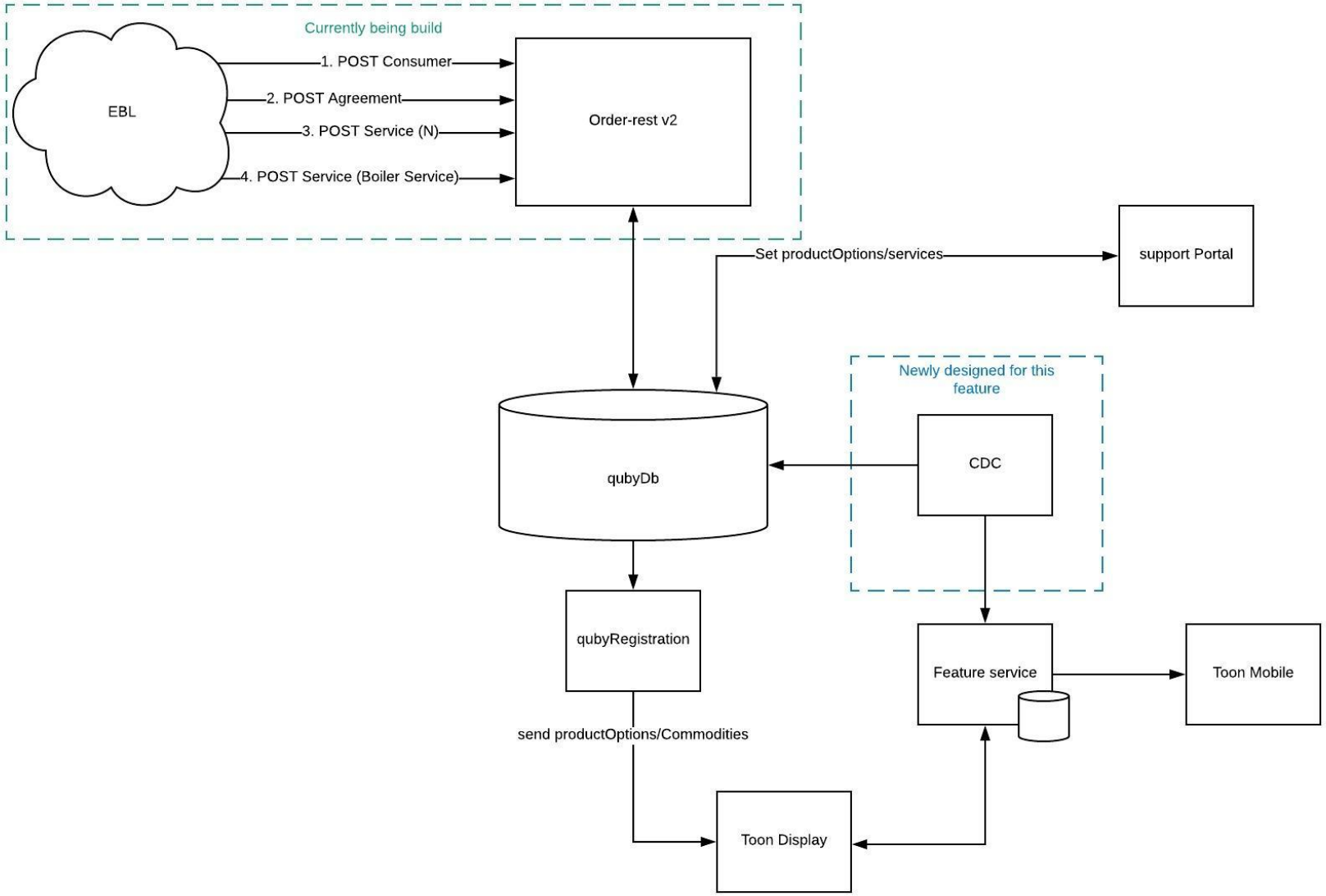


End result

- Maintaining data consistency across services will become difficult
- Less possibility to atomically create/update data across services
- Order-service becomes more complex and hard to maintain
- Achieving fault tolerance with Order-service becomes very hard
- Debugging and issue analysis becomes very difficult (no proper tooling)
- Will lead to spaghetti of service communication.(A.K.A point to point hell)



Use Event Sourcing with CDC



Use Event Sourcing with CDC