Chapter Conclusion

We'll conclude this chapter with a quick summary of what we've learned.

WE'LL COVER THE FOLLOWING

Summary

HTTP

REST and the Atom format offer an easy way to implement asynchronous communication. **HTTP is used as the communication protocol** which has several advantages:

- HTTP is well understood.
- HTTP has already proven its scalability many times.
- Most of the time HTTP is used in projects to transfer other content like JSON via REST or to deliver HTML pages.
- **Most teams have the necessary experience** to implement scalable systems with HTTP.
- HTTP caching is supported hence, **polling of Atom resources can be implemented very efficiently**.

Old events

Old events

Having old events still available helps because another microservice with event sourcing can rebuild its state by processing all events again.

With an Atom-based system, a microservice must also provide old events. This can be done easily in some cases.

If the microservice has stored the old information anyway, it only needs to provide it as an Atom feed. In this case, access to old events is easy to implement.

The data in the Atom Feed comes from the same source as all other representations. Therefore, the data is consistent with the data that can be queried via other means. All these services show different representations of the same data.

No guaranteed delivery

Atom cannot guarantee the delivery of the message. Therefore, the microservices in which the messages are processed should be implemented idempotently and try to read the message several times to ensure that they are processed.

All microservices can receive messages

Atom has no way to limit the reception of a message to just one microservice. Therefore, the microservice must select an instance that then processes the message, or else you have to rely on the idempotency.

Guaranteed order of messages

Atom can guarantee the order of the messages because the feed is a linear document so the order is fixed. However, this requires that the order of the entries in the feed does not change.

Atom is a very simple alternative for asynchronous communication.

Benefits

- Atom **does not require additional infrastructure**, just HTTP.
- **Old events are easily accessible if necessary**. This can be advantageous

for event sourcing.

- The **sequence can be guaranteed**.
- The **Atom feed is consistent**. It is just another representation of the data and contains the same information as all other representations of the data.

Challenges

- **Guaranteed delivery is difficult**. The recipient can attempt to read the data several times to guarantee all data is processed. However, the responsibility for this lies with the recipient, not with the infrastructure.
- **Messages for only one recipient are difficult**. All recipients poll the messages. They must then decide which recipient actually processes the message.
- In part, Atom is **not well suited for the representation of events** because it was originally designed for blogs.

In the next chapter, we'll study synchronous microservices concepts.