What’s next?

Zoltan Altfatter

Roadblocks

- There are still some barriers to use Reactive programming:
- Relational Data Access
- Cross-process back pressure (networking)
    - How can I guarantee that when I call a microservice
        - it doesn’t materialize all of its data
        - It doesn’t overflow me with data

HTTP is the most common communication protocol, but there are use cases for which is not a good fit like:

- phone notifications
- smart watches sending statistics

RSocket

- http://rsocket.io/
- core team from Netifi, Facebook, Pivotal
- **bi-directional**, **multiplexed**, **message-based**, **binary** protocol based on Reactive Stream specification
- 4 interaction models
  - Request-Response
  - Fire-and-Forget
  - Request-Stream
  - Chanel

RSocket API

```
public interface RSocket extends Availability, Closeable {

    Mono<Void> fireAndForget(Payload payload);

    Mono<Payload> requestResponse(Payload payload);

    Flux<Payload> requestStream(Payload payload);

    Flux<Payload> requestChannel(Publisher<Payload> payloads);

}
```

Flexibility

- Language agnostic
  - Implementations in Java, JavaScript, C++, Kotlin
- Payload agnostic
  - JSON, Protobuf, Custom Binary
- Transport agnostic
  - TCP, Websockets, HTTP/2, Aeron
- Programming model agnostic
  - RSocket interface is designed to a building block that multiple programming models could build upon
  - Pivotal is working on a Spring programming model over this

Interaction Models

```
Mono<Payload> response = client.requestResponse(requestPayload)
```

- Surpasses HTTP because is asynchronous and multiplexed

```
Mono<Void> response = client.fireAndForget(requestPayload)
```

- When response is not necessary, used for non-critical event logging

```
Flux<Payload> response = client.requestStream(requestPayload)
```

- The response is streamed back, data is not materialized until ready to send

```
Flux<Payload> response = client.requestChannel(requestPayload)
```

- Request with account number, respond with a real-time stream of account transactions, update subscription to filter certain transaction types, respond with filtered real-time stream of account transactions

Demo

https://github.com/altfatterz/rsocket-r2dbc-demo