Dennis Kieselhorst

<workflow
/analytica>

Die OpenAPI Spezifikation – ein Standard zur Beschreibung moderner APIs

Die OpenAPI Spezifikation

Ein Standard zur Beschreibung moderner APIs

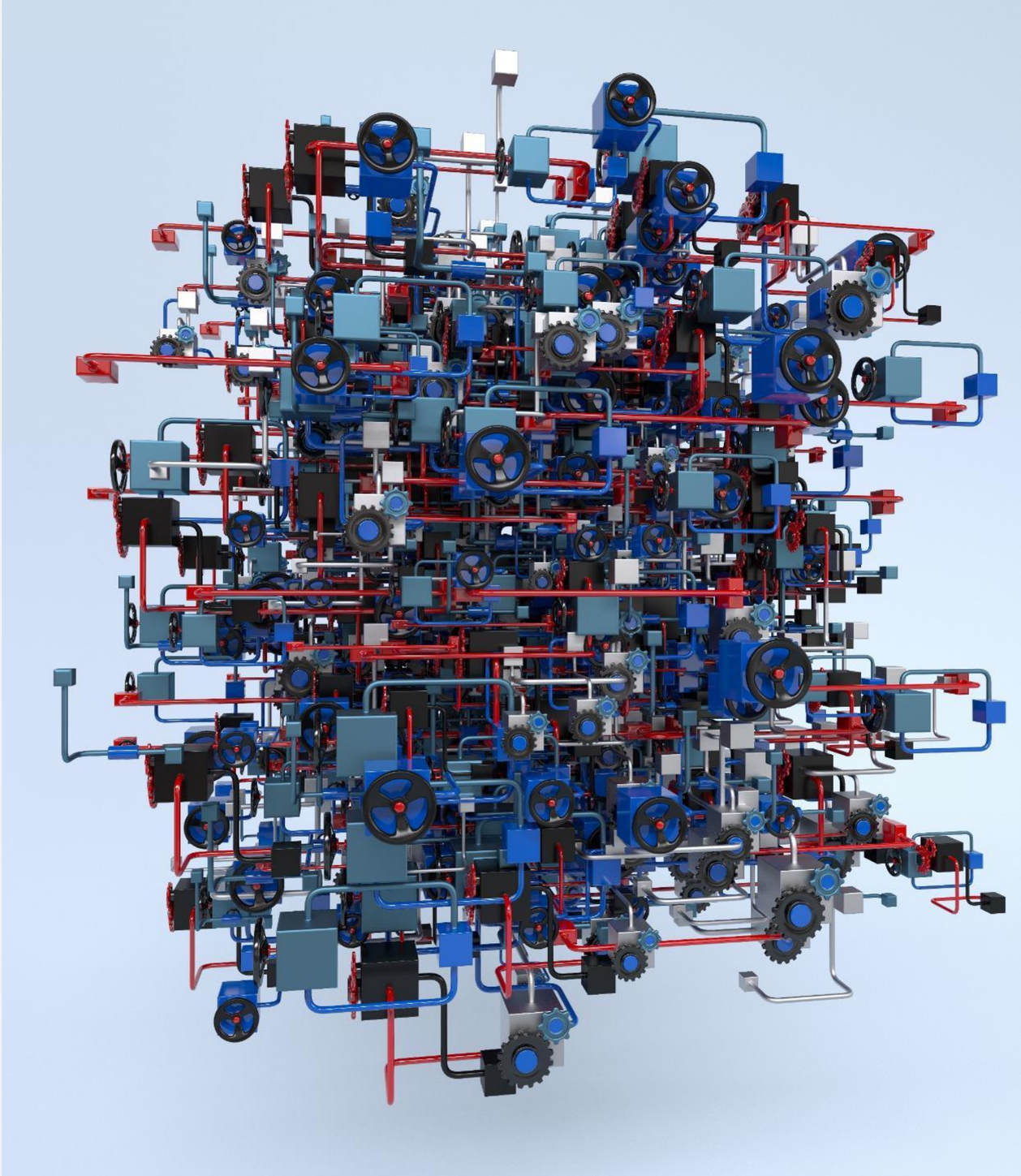
Dennis Kieselhorst

Sr. Solutions Architect
Amazon Web Services

Motivation

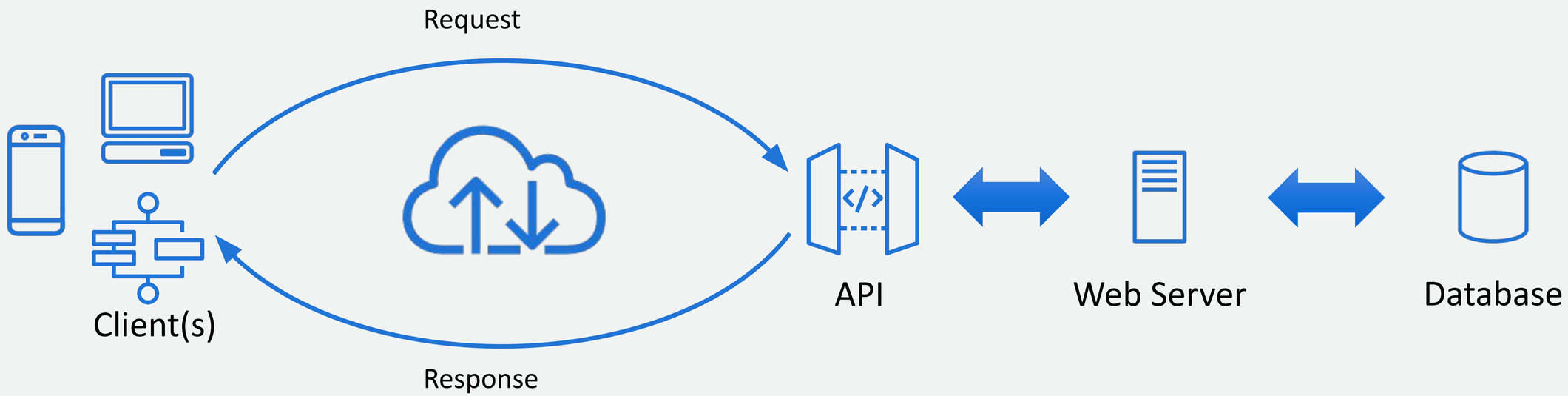
- Processes and IT landscapes are becoming more and more complex.
- Modern architecture approaches like Microservices bring a lot of benefits but also increase the complexity.
- It’s challenging to document dependencies and interactions, manual documentation often gets outdated.

How to maintain an overview?



Application Programming Interfaces (APIs)

- Simplify programming by abstracting the underlying implementation and only exposing objects or actions needed.
- APIs are the „glue“ between applications.



API Mandate at Amazon

1. *All teams will henceforth expose their data and functionality through service interfaces.*

2. *Teams must communicate with each other through these interfaces.*

3. *There will be no other form of inter-process communication allowed.*

4. *It doesn't matter what technology they use.*

5. *All service interfaces, without exception, must be designed from the ground up to be externalizable.*

Conway's Law: Organizations design systems that model the organization and communication structure

How to describe APIs? - The Open API Initiative

- The Web Services Description Language (WSDL) was famous but not resource-oriented.
- Swagger was born in 2010 as an interface definition language (IDL) for REST-APIs with a fast-growing fan community.
- End of 2015 the Open API Initiative (OAI) under the umbrella of the Linux Foundation was formed (currently 44 members).
- Swagger was renamed to OpenAPI Specification (OAS).
  - The old name is still used for tooling.
  - Further spec development happens on GitHub.
  - In July 2017 the new major version 3.0.0 was published, the release of the latest version 3.1.0 happened in February 2021.

Contract/ API-First vs. Code First approach

Contract/ API-First

Specification is defined first and acts as service contract

- helpful for different teams on client-/ server side
- even more across different companies/ with third parties

Client- and servercode can be generated with a code generator

- ensures code is always consistent to the API
- compile errors for breaking changing (possible to automate using Continious Integration tool)
- code is not as clean as handwritten code, may look confusing
- tolerant reader pattern may be a better option over spec-based code generation (depends on the scope and change frequency of the API)

Code First

Specification is derived from API implementation

- code can be annotated
- export is either done at compile or runtime (e.g. using /openapi.yaml http call)
- developers are familiar with it fast for simple APIs

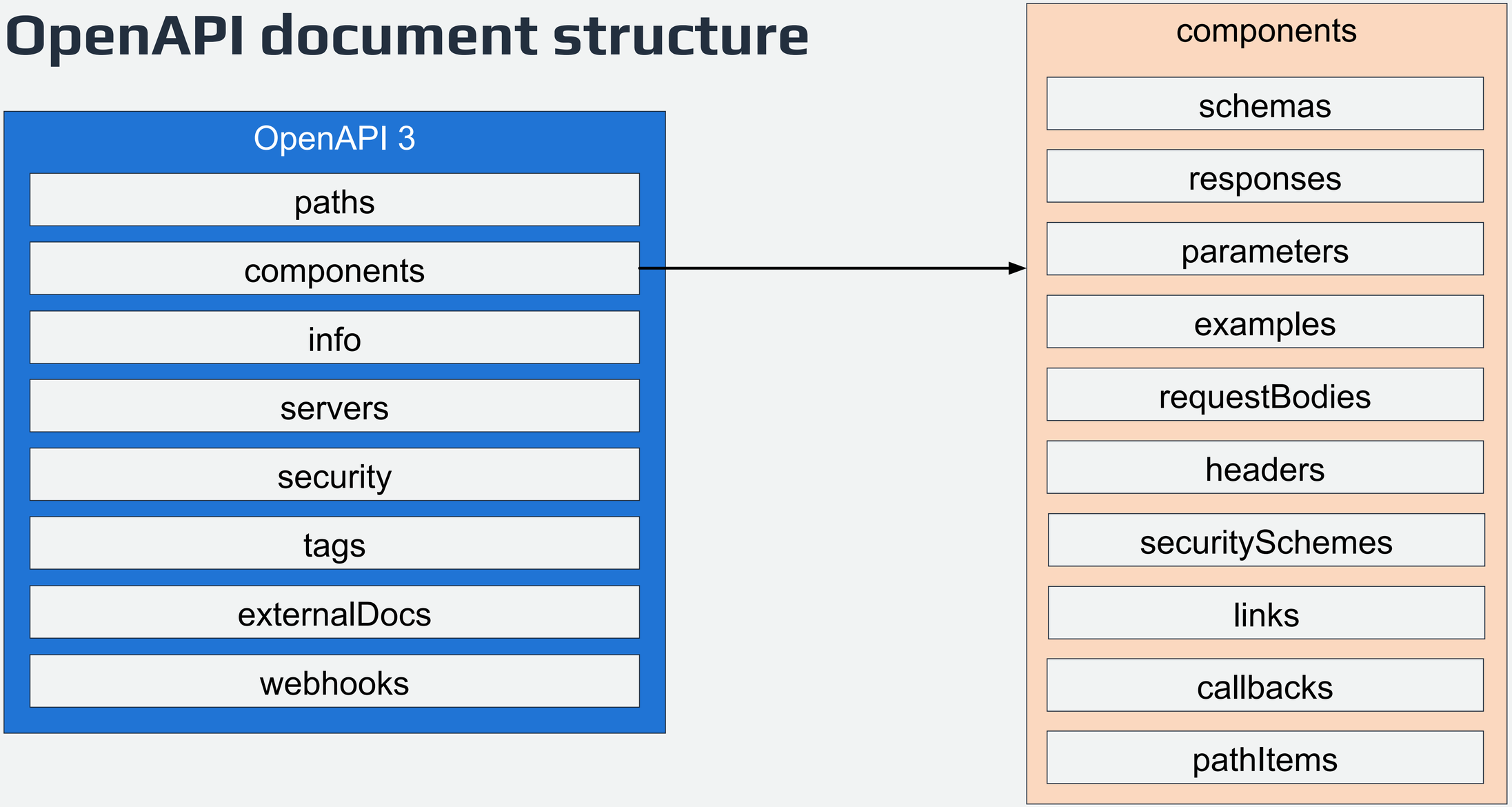
Generated specification may contain unused resources

- easily happens that something is accidently exposed
- often lack of documentation

Tool support for OpenAPI development

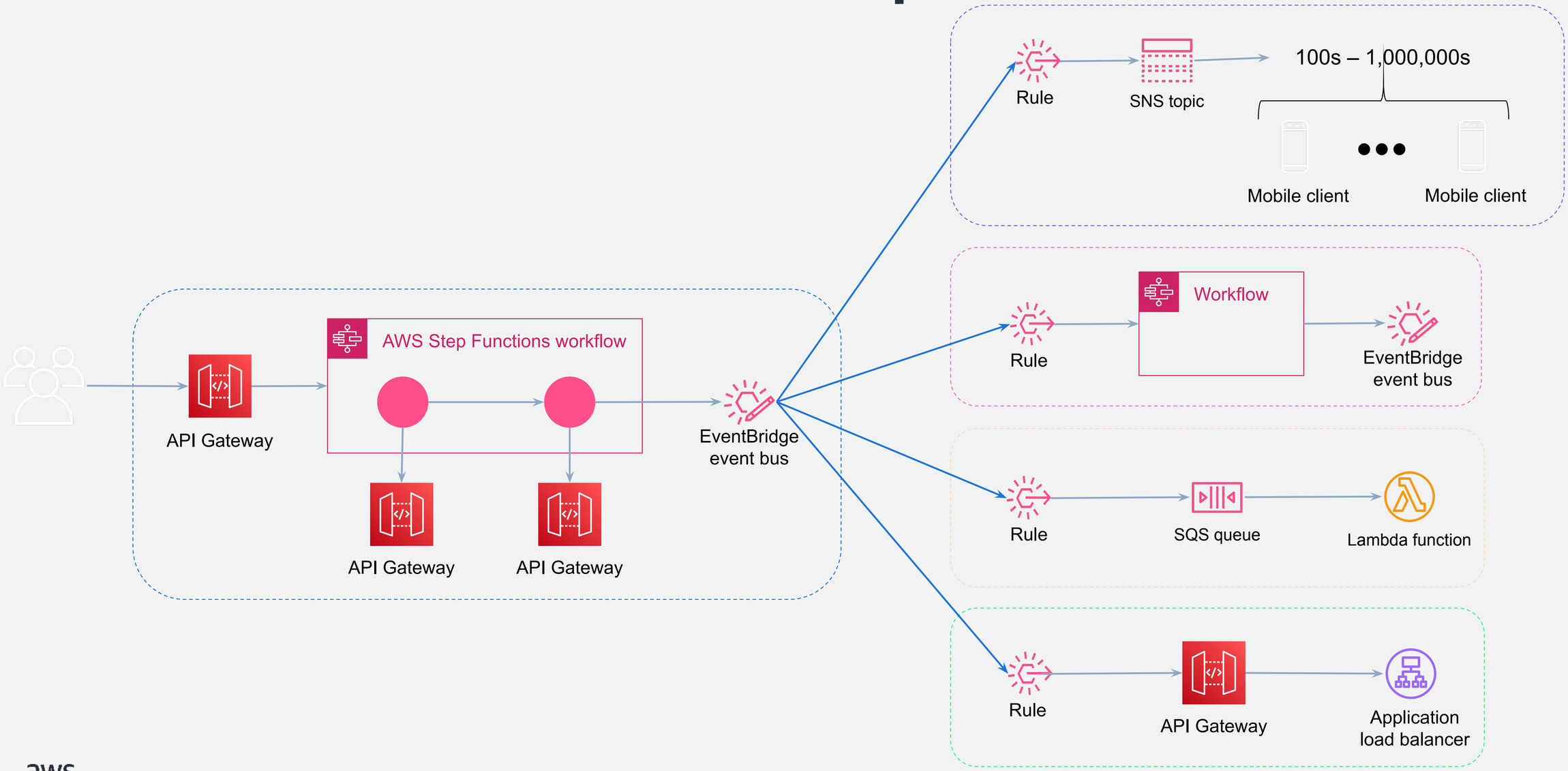
- Low-level tooling
  - generic libraries
  - e.g. parser/ validator
- Editors
- User interfaces
- Mock servers/ testing tools
- Client and server implementations
- Code generators

OpenAPI document structure



Demo

Event-driven architecture example



Summary

- APIs help to control the data flow.
- Designing an API specification (contract) is a critical step.
- Ensure ownership for each API by setting up a proper team structure.
- OpenAPI is well supported by tools and frameworks.
- Serverless services for API development and management simplify the process and increase agility.

Thank you!

Dennis Kieselhorst
linkedin.com/in/kieselhorst/