Microservices

Luis Daniel Benavides Navarro, Ph.D. 5-10-2021

Problem

- How to architecture a global information system?
- The system has the following requirements:
  - Easy to develop, maintain, evolve, and innovate
  - Multiple types of clients
  - Scalable
  - Secure
  - Sustainable (optimization of resources)
  - Resilient
  - Multi-tenancy

What about traditional techniques?

- SOA and the monolith are very useful!
- But:
  - Traditional development methods do not scale well
  - IT Architecture does not scale well

Microservices

- Microservices is an architectural pattern to structure big (very big) internet systems.

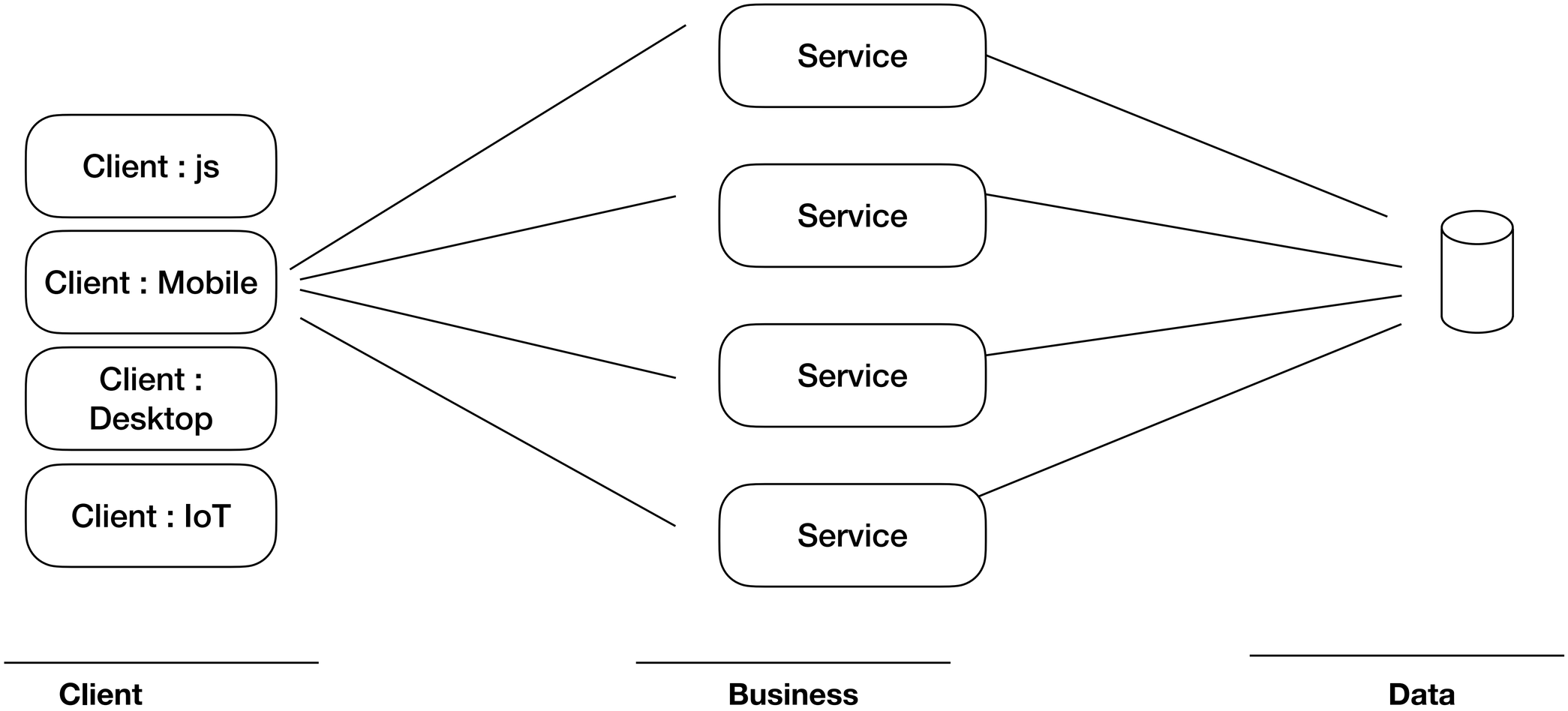
- Two main ideas drive the conception of microservices:

  - The backend is structured as a set of small services (each service with a single responsibility).

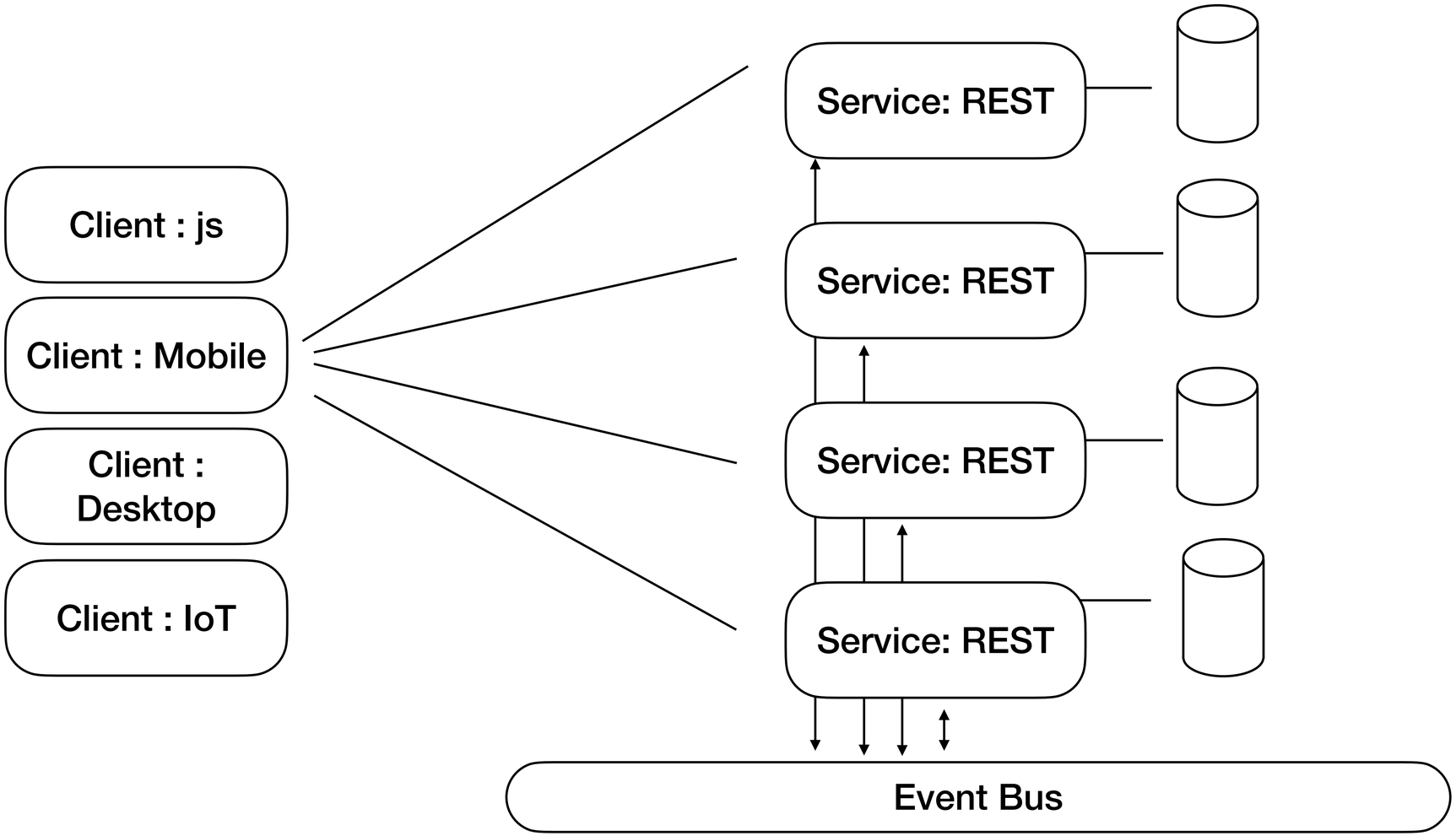
  - A small team is responsible for the complete lifecycle of the microservice (design/architecture, design/implementation, tests, deployment, maintenance, and evolution).

Design-Architecture

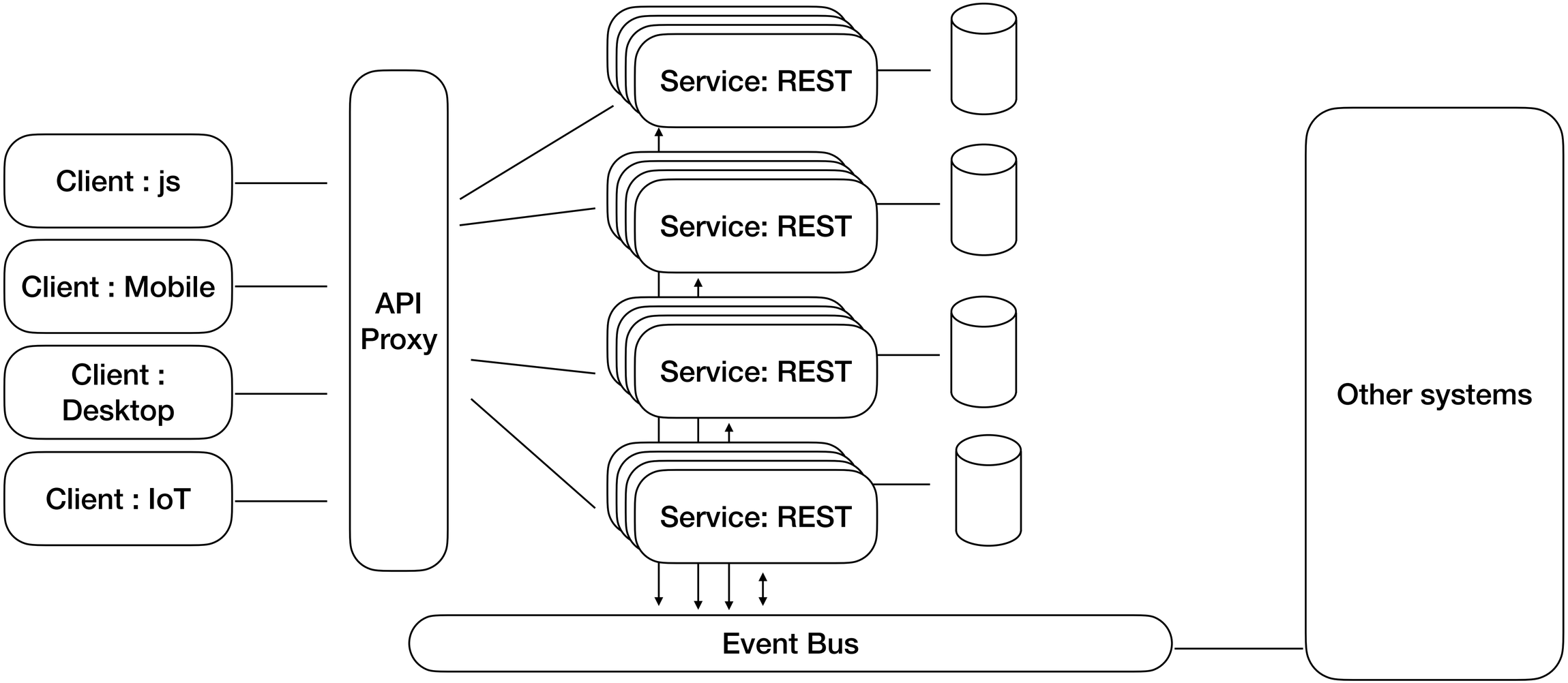
Design Metaphor



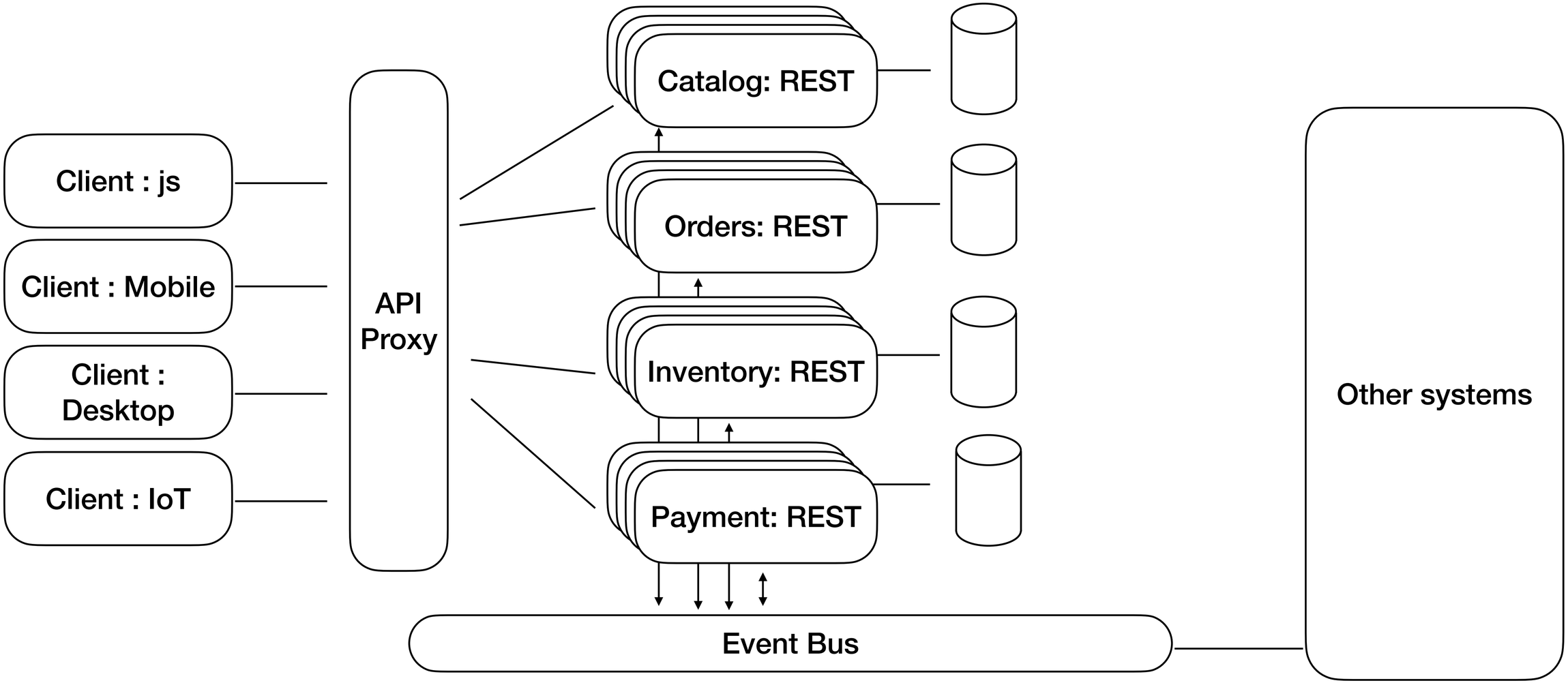
Design Metaphor



Design Metaphor



Example


Client : js
Client : Mobile
Client : Desktop
Client : IoT
API Proxy
Catalog: REST
Orders: REST
Inventory: REST
Payment: REST
Event Bus
Other systems

API Design

What is an API?

- Define Entities (similar to objects)
- Define operations
- Entity API: Fields with name type and annotations
- Operations: name, return types, parameters with types, annotations, and modifiers

Entities

```
"user": {
  "description": "Represents a single user in the system",
  "fields": [
    { "name": "id", "type": "string" },
    { "name": "email", "type": "string", "required": false, "annotations": ["personal_data"] },
    { "name": "name", "type": "name", "annotations": ["personal_data"] },
    { "name": "status", "type": "user_status", "default": "active" }
  ]
},
```

```
"user_form": {
  "fields": [
    { "name": "email", "type": "string", "required": false, "annotations": ["personal_data"]  },
    { "name": "password", "type": "string", "required": false, "annotations": ["personal_data"]  },
    { "name": "name", "type": "name_form", "required": false, "annotations": ["personal_data"]  }
  ]
},
```

Taken from: https://www.youtube.com/watch?v=j6ow-UemzBc

Exposed operations


```
"resources": {
  "io.flow.common.v0.models.user": {
    "operations": [
      {
        "method": "GET",
        "description": "Returns information about a specific user.",
        "path": "/:id",
        "responses": {
          "200": { "type": "io.flow.common.v0.models.user" },
          "401": { "type": "unit" },
          "404": { "type": "unit" }
        }
      },

      {
        "method": "POST",
        "description": "Create a new user. Note that new users will be created with a status of pending and will not be able
Flow team.",
        "body": { "type": "user_form" },
        "responses": {
          "201": { "type": "io.flow.common.v0.models.user" },
          "401": { "type": "unit" },
          "422": { "type": "io.flow.error.v0.models.generic_error" }
        }
      }
    ]
  }
}
```


Taken from: https://www.youtube.com/watch?v=j6ow-UemzBc

Alternatives

https://docs.swagger.io/spec.html#1-introduction


```
{
  "apiVersion": "1.0.0",
  "swaggerVersion": "1.2",
  "apis": [
    {
      "path": "/pet",
      "description": "Operations about pets"
    },
    {
      "path": "/user",
      "description": "Operations about user"
    },
    {
      "path": "/store",
      "description": "Operations about store"
    }
  ],
  "authorizations": {
    "oauth2": {
      "type": "oauth2",
      "scopes": [
        {
          "scope": "email",
          "description": "Access to your email address"
        },
        {
          "scope": "pets",
          "description": "Access to your pets"
        }
      ],
      "grantTypes": {
        "implicit": {
          "loginEndpoint": {
            "url": "http://petstore.swagger.wordnik.com/oauth/dialog"
          },
          "tokenName": "access_token"
        },
        "authorization_code": {
          "tokenRequestEndpoint": {
            "url": "http://petstore.swagger.wordnik.com/oauth/requestToken",
            "clientIdName": "client_id",
            "clientSecretName": "client_secret"
```

Heuristics

Good practices

- **API first (and event first)**
- **Minimize intra service API communication.**
- **Prefer event-based asynchronous com.**
- Small teams, lean development (post-agile)
- DevOps
- Automation of the life cycle (Maven, Git, Tests, ETC.)
  - Continuous Delivery/Continuous integration
  - TDD
- Organize conde in Repos. Organize APIs def. in independent REPOS (Discuss)
- Standardization (e.g., 1 P. Language per 4000 developers)

Deployment

A complete example