API Workshop

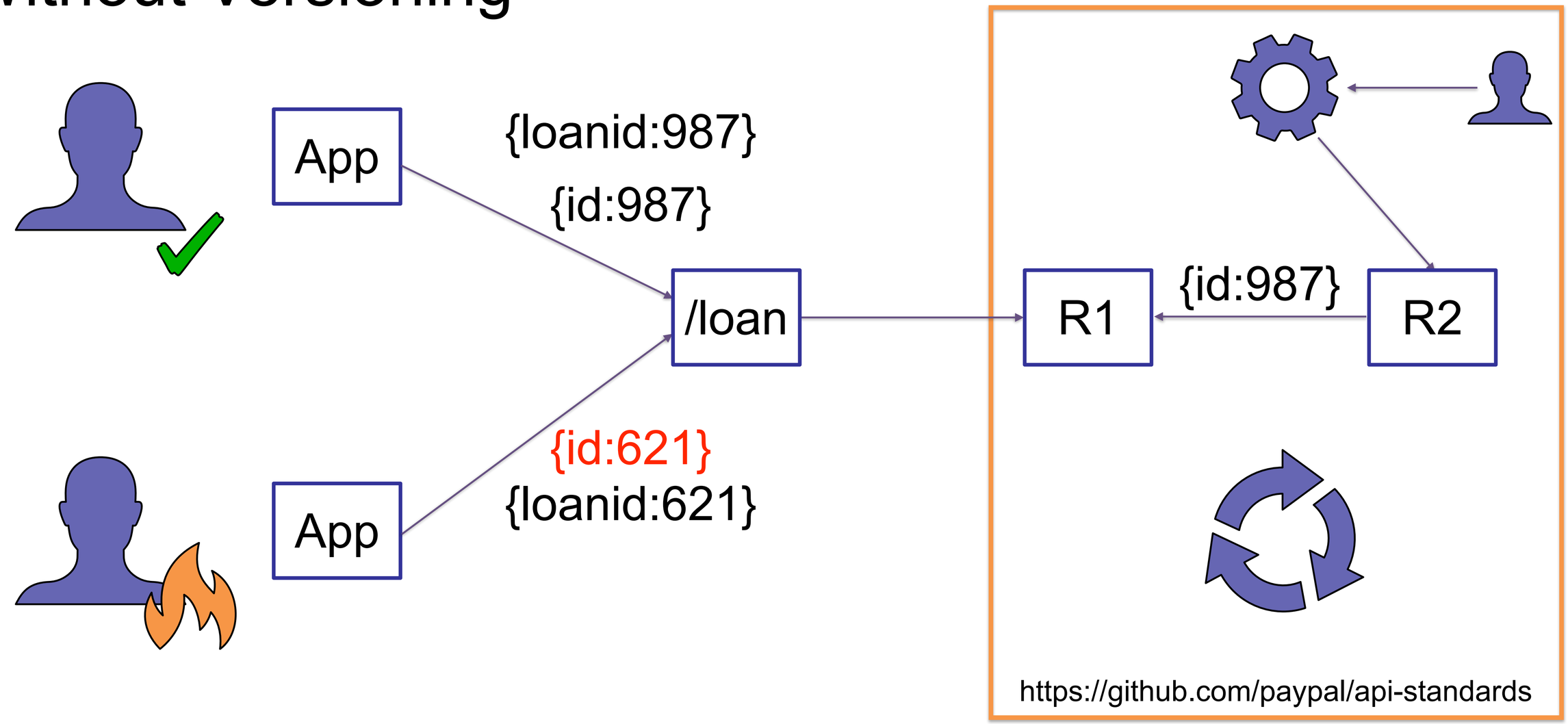
Versioning

O'REILLY®
Software
Architecture

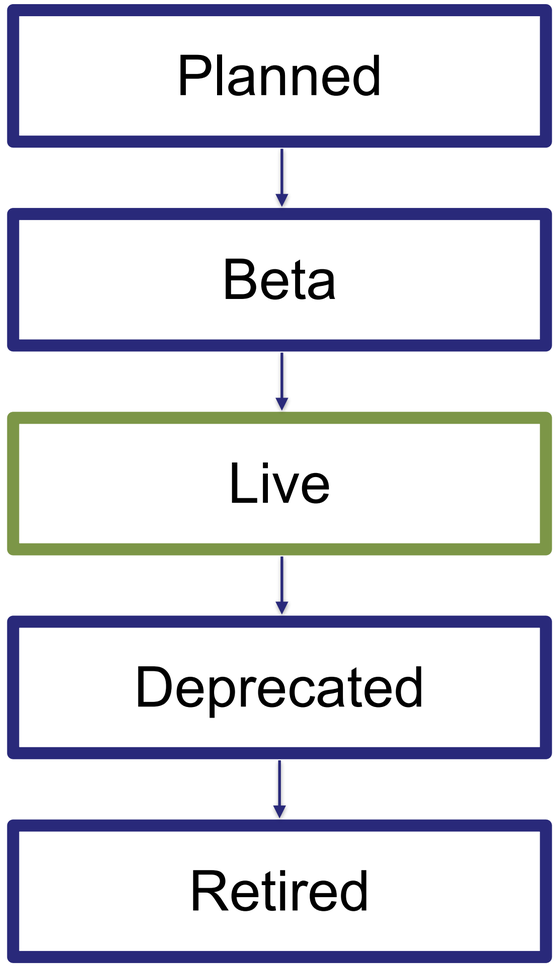
Agenda

- No Versioning
- Introduction to API Lifecycle
- Comparing OpenAPI Specifications
- API Versioning and Specification with API management

Without Versioning



API Lifecycle



Minor Release

- Backwards compatible
- Customer does nothing
- Changes additive and optional
- Semantics don't change

Major Release

- Can break compatibility
- Must have a migration plan
- Runs with deprecated API

https://github.com/paypal/api-standards

Comparing Open API Specifications

- API Specifications accurately reflect the current version of an API
- When upgrading versions API Specifications can be compared
- **swagger-diff** helps identify breaking changes vs minor upgrades
- Can be integrated as part of an API Delivery Pipeline

Versioning with API Management

- Extending APIs to partners requires a defined format
- API Management systems rely on Open API Specifications to reverse proxy
- It is possible to perform validation of API requests based on the specification
- New versions of an API can be deployed via the use of DevOps

Version format

- Different guidance between different guidelines
  - Microsoft suggest *Major.Minor* e.g. v2.1
    - URL path explicit versioning. e.g. https://api.contoso.com/v1.0/products/users
    - Query string parameter e.g. https://api.contoso.com/products/users?api-version=1.0
  - Paypal suggest *Major.Minor.*
    - *“The API major version is an integer value which MUST be included as part of the URI”*

      (https://github.com/paypal/api-standards/blob/master/api-style-guide.md#resource-path)

Lab 4 - Versioning

https://github.com/nickebbitt/api-workshop