mojaloop foundation

Mojaloop Testing Toolkit

Onboarding, Validation & QA, Demonstration of Mojaloop implementations using the TTK

General Themes

- Validating (FSPIOP) API implementations
  - DFSP
  - Switch / Hub
- Other API implementations

Multi-Purpose

Easy to Use

Onboarding

Demonstration

Validation, QA

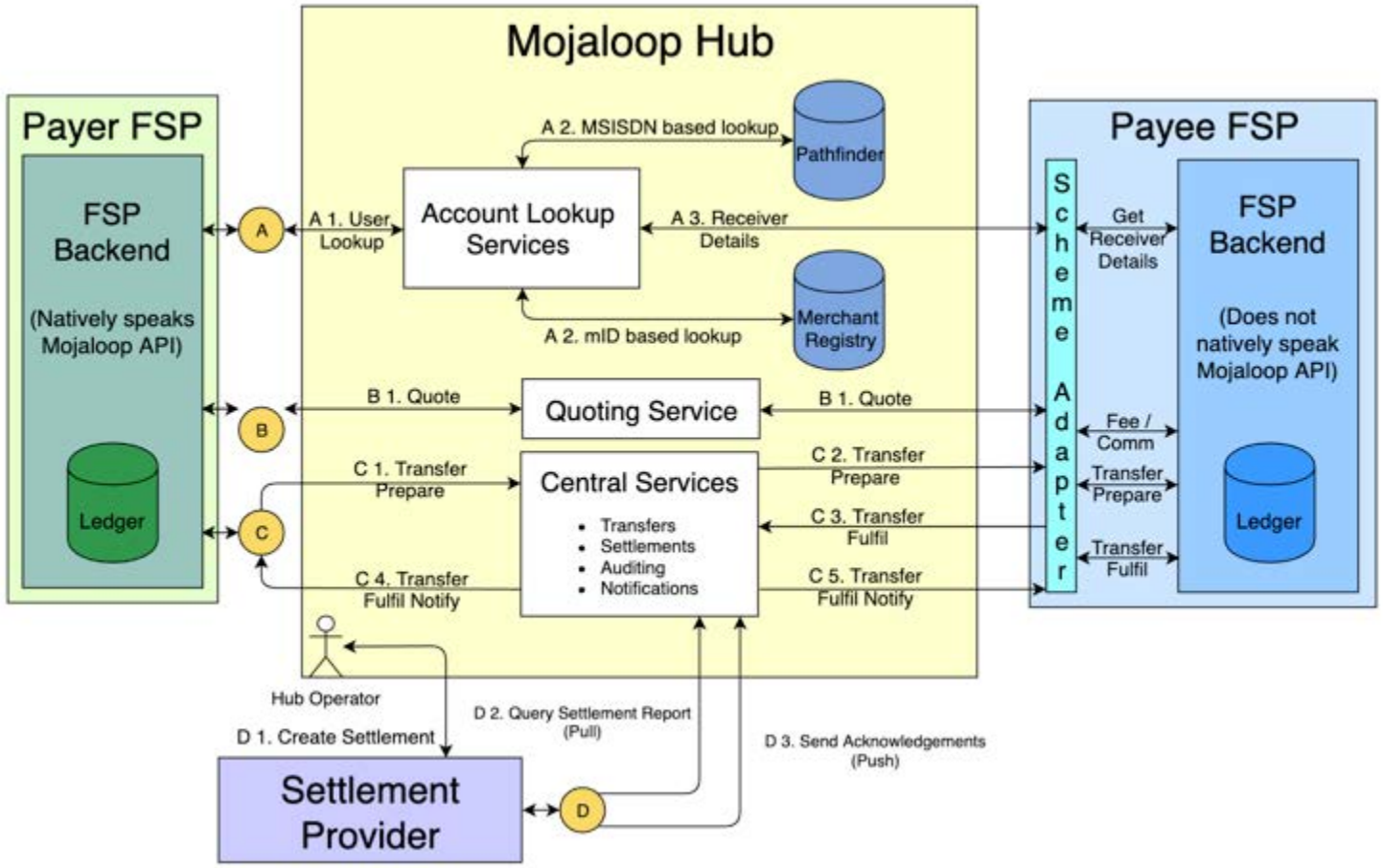
mojaloop foundation

Mobile Simulator Demo

Where does it fit?

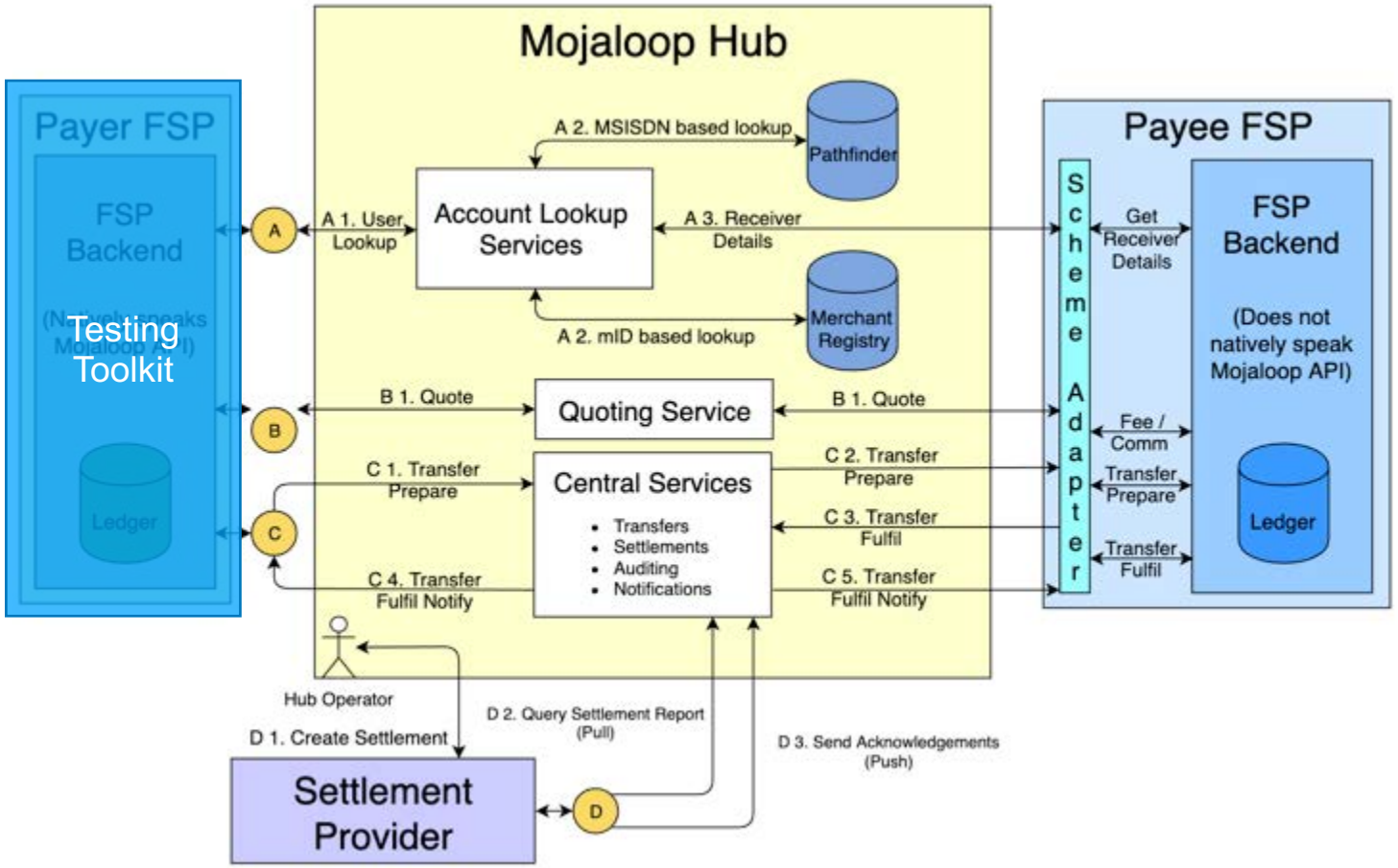
Use Cases and target users

Default Mojaloop Architecture



mojaloop foundation

1. TTK as a Payer FSP

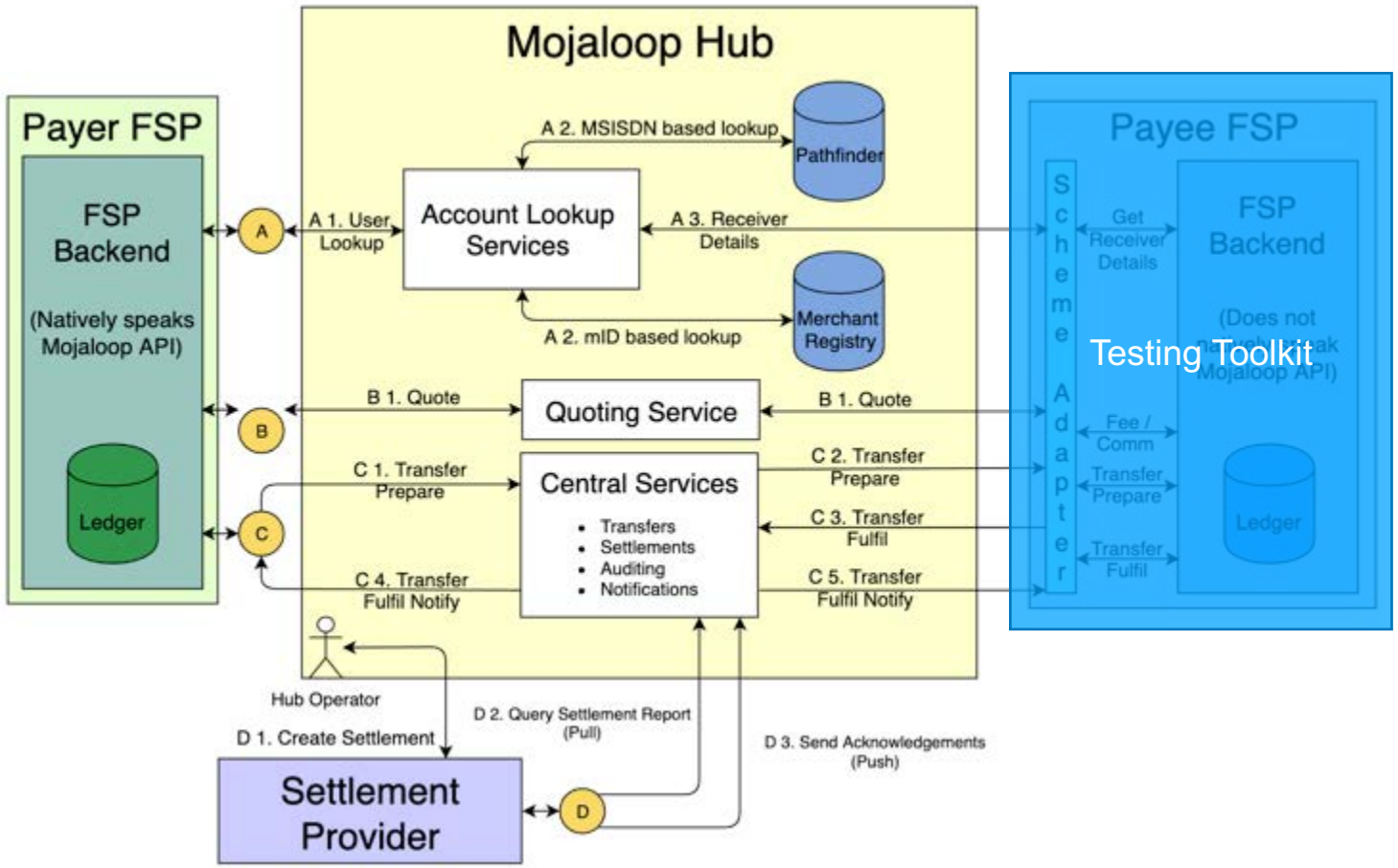


Target Users

- Product / Business Users: *Explore Mojaloop, Initiate requests*
- QA teams: *Regression Tests, monitoring, debugging*
- Infra: *Scheduled Jobs, post-install validation*

mojaloop foundation

2. TTK as a Payee FSP

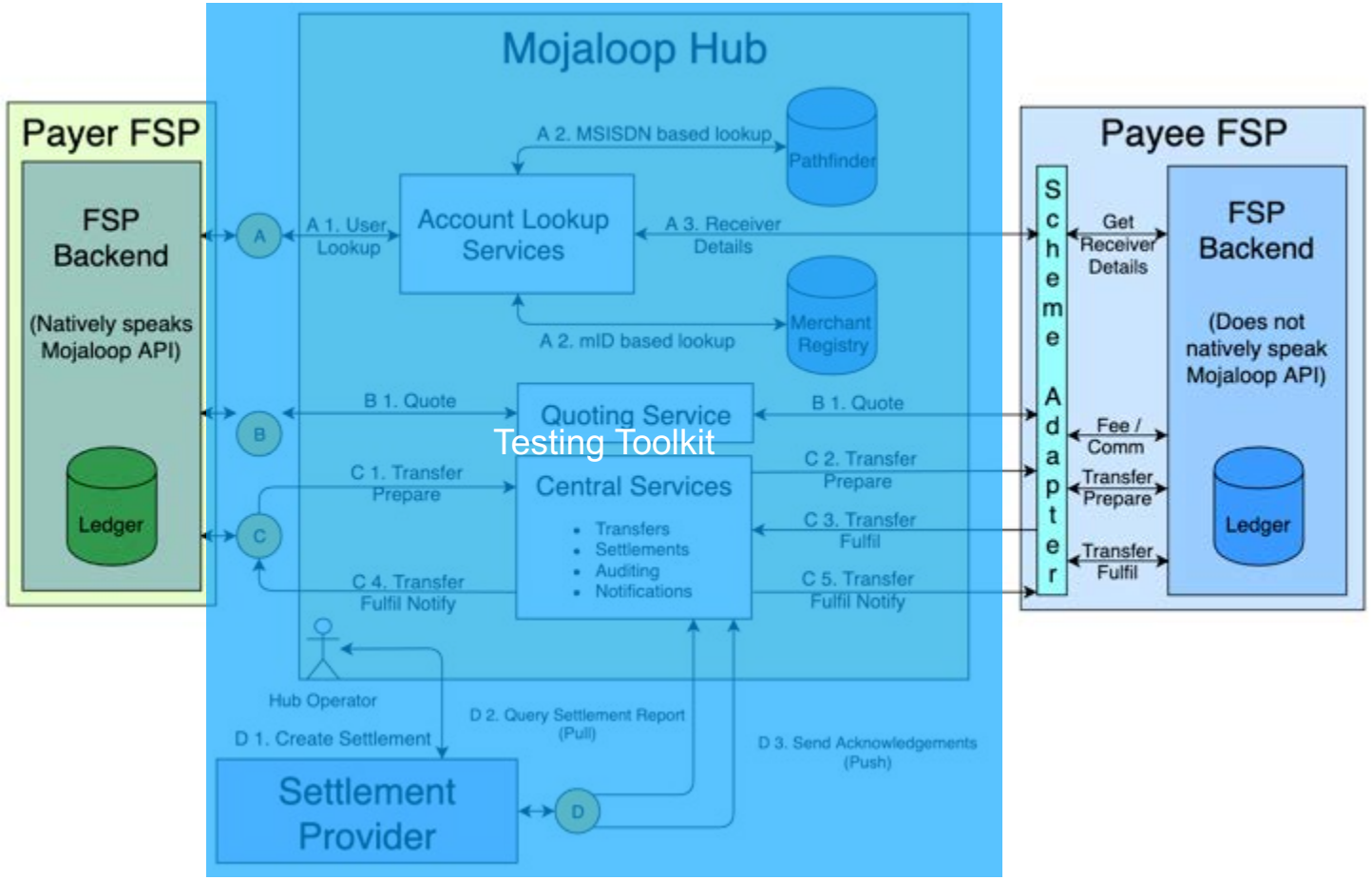


Target Users

- Developers: *Onboarding, Exploration, Development*
- Business Operations: *Onboarding, Validating requests, monitoring*
- Infrastructure teams: *As a simulator for testing*

mojaloop foundation

3. TTK as a Hub



Target Users

- Schemes: *Simulation environment, Labs, Certification, Onboarding FSPs*
- FSPs: *Validating implementations, Simulate Switch*
- Developers: *Development support by simulation, debugging, monitoring, demonstration*

mojaloop foundation

Deployment Scenarios

Various ways of deploying Testing Toolkit

1. Local Deployment

Docker-compose

```
git clone https://github.com/mojaloop/ml-testing-toolkit
cd ml-testing-toolkit
docker-compose up
```

Node Service

```
git clone https://github.com/mojaloop/ml-testing-toolkit
cd ml-testing-toolkit
npm start
git clone https://github.com/mojaloop/ml-testing-toolkit-ui
cd ml-testing-toolkit-ui
npm start
```

Roadmap Native desktop application (MacOS, Windows and Linux)

2. As part of the mojaloop helm

Included by default

```
helm install <name> -n <namespace> mojaloop/mojaloop
```

Post-install Hooks

```
helm install <name> -n <namespace> mojaloop/mojaloop
 --wait --timeout 10m  --set ml-ttk-posthook-
setup.postInstallHook.enabled=true,ml-ttk-posthook-
tests.postInstallHook.enabled=true
```

3. Hosted service

1. Schemes can host the TTK and provide access to their DFSPs
    a. DFSP Onboarding
    b. Labs
2. Deployable with helm chart
    a. MCM
    b. MongoDB
    c. Keycloak Oauth server

Enhancements in PI-12

1. Comprehensive test suites for
   a. Hub testing
   b. FSP testing
2. Enhancements
   a. Iterations in test case execution
   b. javascript scripts in test case editor
   c. New functions for JWS validation and signing
   d. websocket functions for payee side assertions
   e. Native wrapper program for test case creation
3. Generalizing OSS usage
   a. Uploading new API Spec through UI
4. Extensions for Mojaloop Demos
   a. Mobile Simulator
5. Bug fixes
   a. Bugs related to test case creation

mojaloop foundation

Demos

1. FSP and Hub test suites - **https://github.com/mojaloop/testing-toolkit-test-cases**
    1. Payee side assertions using websockets (mojaloop simulators)
    2. JWS validation
2. Uploading API Spec
3. Iterations and specific results

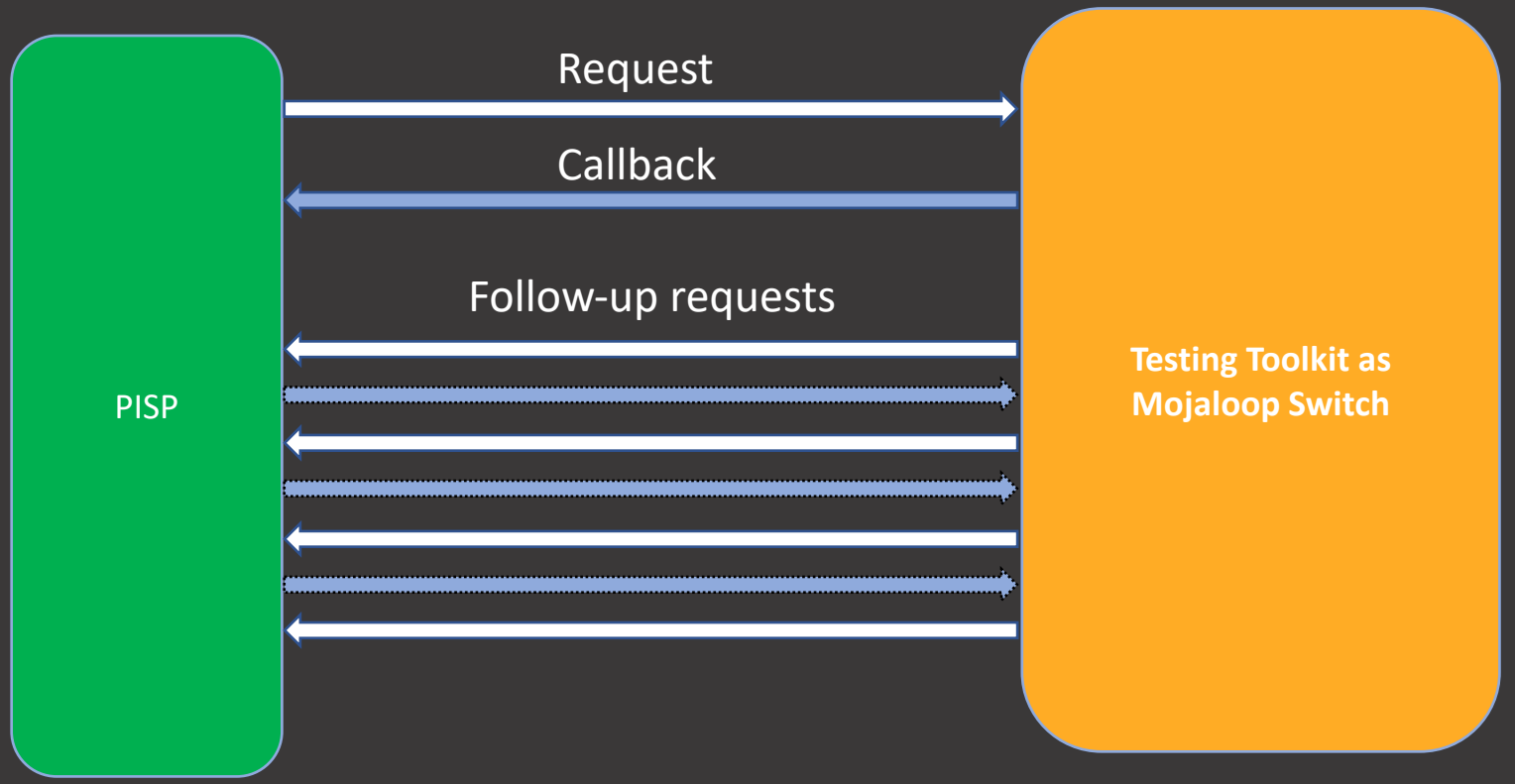
Roadmap

1. TTK as an FSP interacting with a Hub over mTLS
2. Distributing TTK as native desktop application for different platforms
3. Event framework integration
4. Documentation on Test cases - metrics on coverage and grouping
5. Performance metrics of a Hub (granularity and iterations)
6. Automatic API provisioning for async APIs
7. TTK Module on the Mojaloop Training Program (MTP)

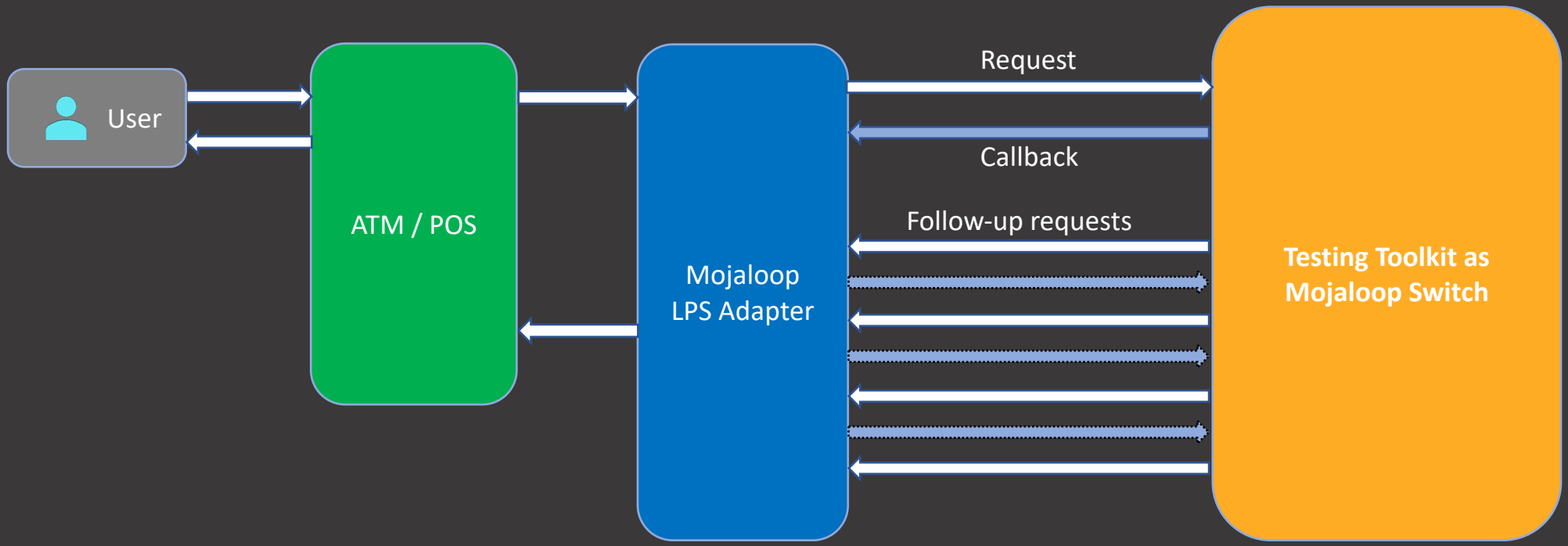
mojaloop foundation

Appendix

Use-case: PISP Team

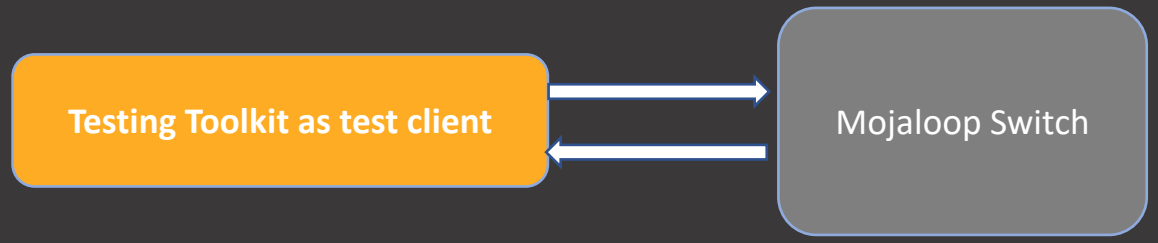


Use-case: ATM / POS

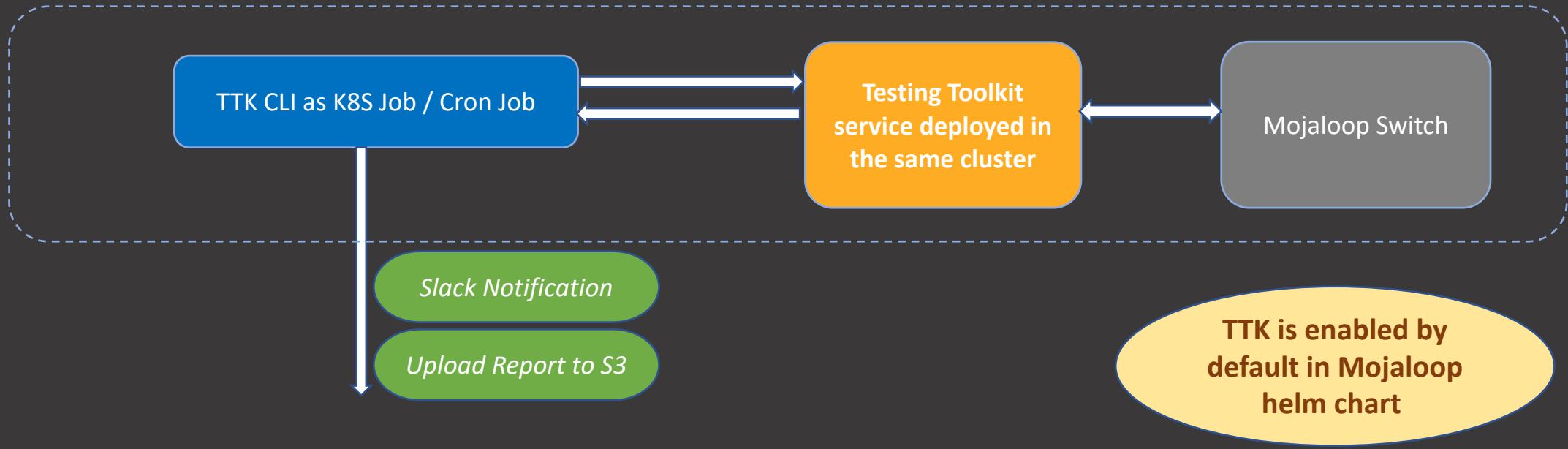


Use-case: Core Team

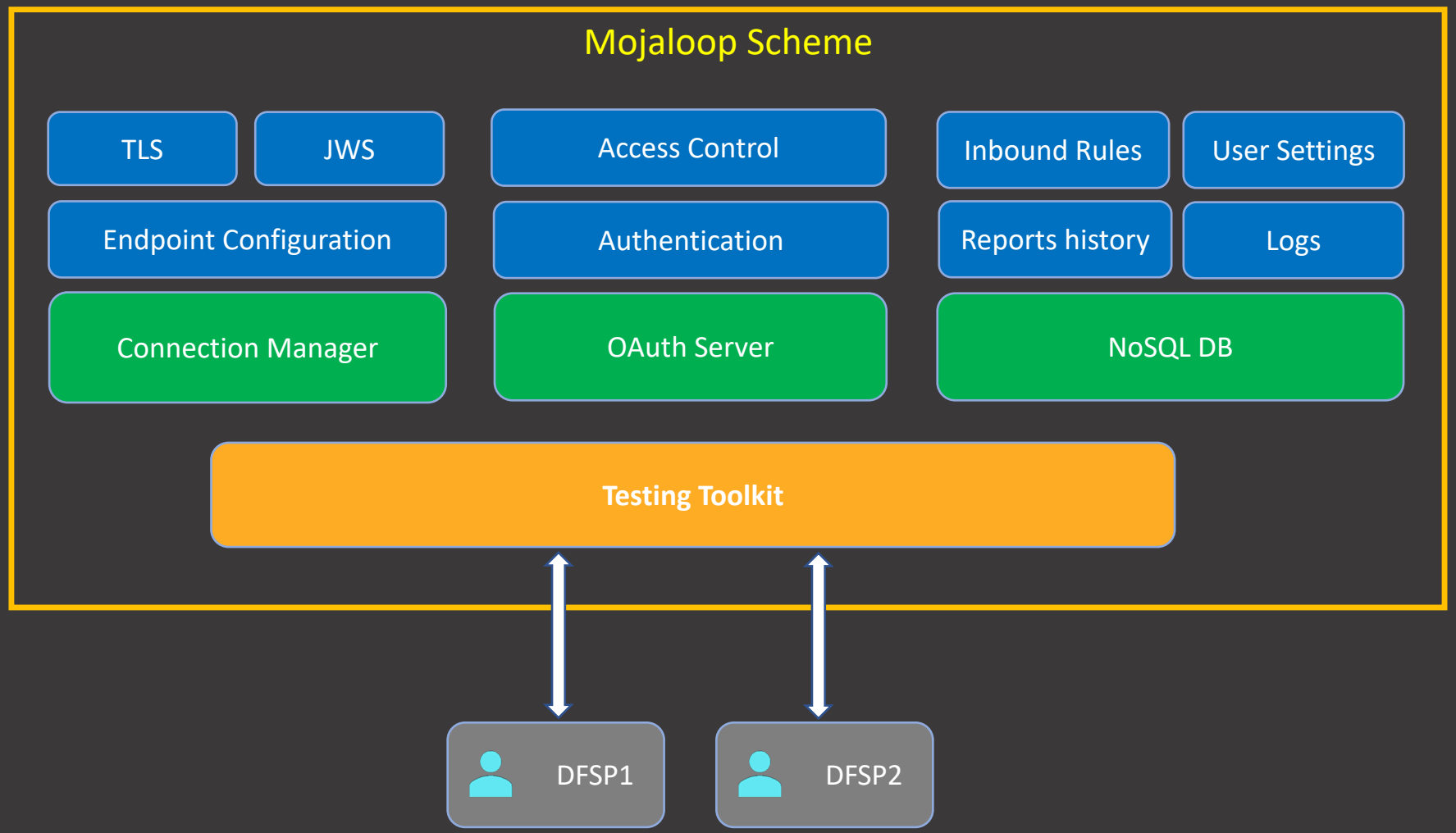
As a replacement for Postman



Scheduled Regression tests



Adding Capability for Hosted Solution



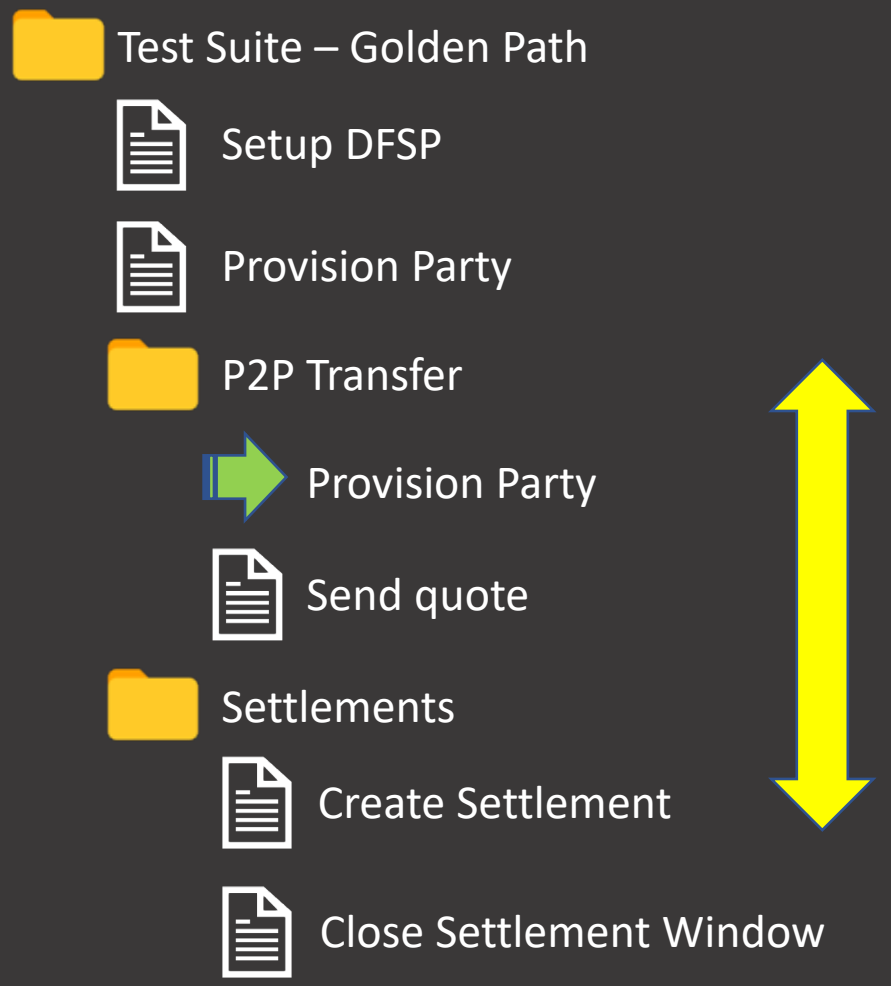
Testing toolkit: Initial goals

1. Test *any* Mojaloop FSPIOP API implementation (initial goal)
2. Simple to use
3. Support different versions of Mojaloop API
4. Highly configurable (Configurations portable)
5. Can validate Inbound requests
6. Can generate Outbound requests

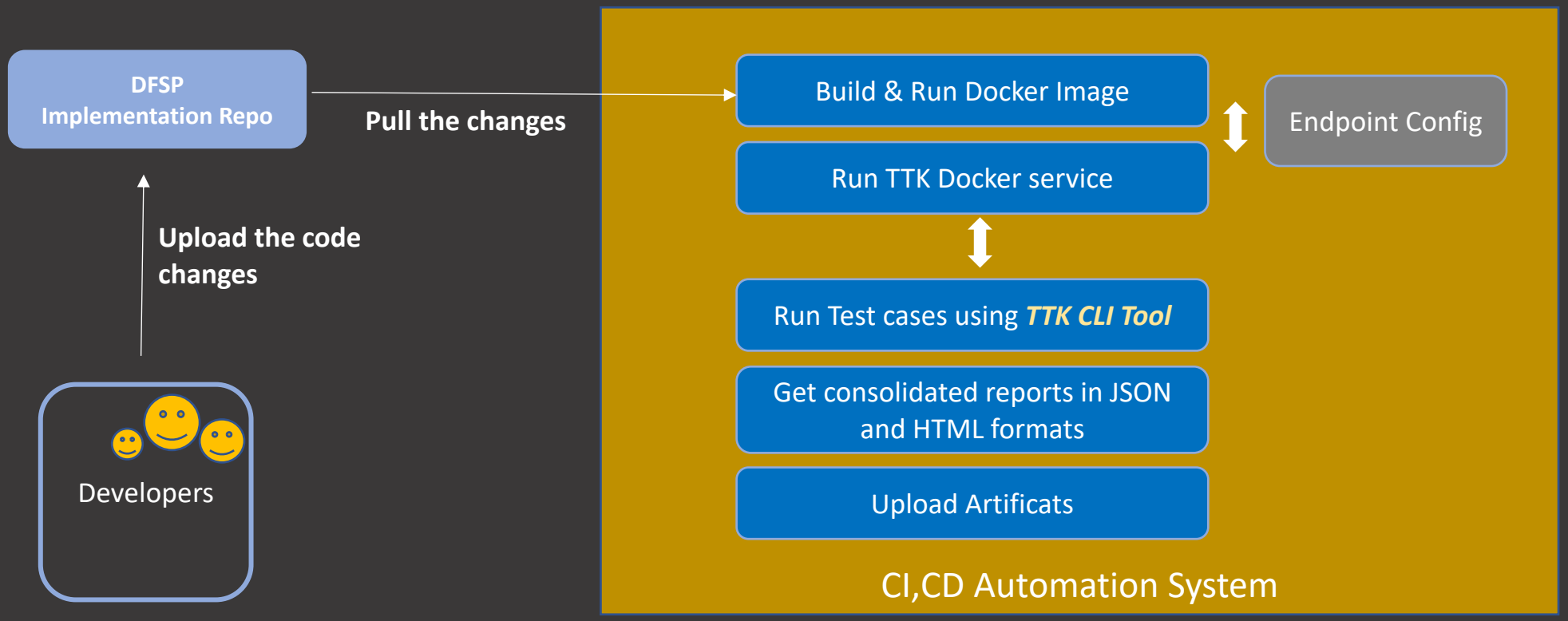
PI-11 Toolkit Features

1. User Interface for QA / Product / Business users
2. Version validation and negotiation, schema validation & additional validation
3. Dynamic callback and error callback generation based on rules
4. Initiation of use cases (outbound) - Assertions and report generation
5. Simultaneous support for multiple APIs
6. Separate test sets for Hub and DFSP implementations
7. Synchronous & Asynchronous APIs
8. Supports JWS and mTLS
9. Command line client (CLI) for scheduling & devops automation
10. Easily portable (Light weight and import and export configuration options)

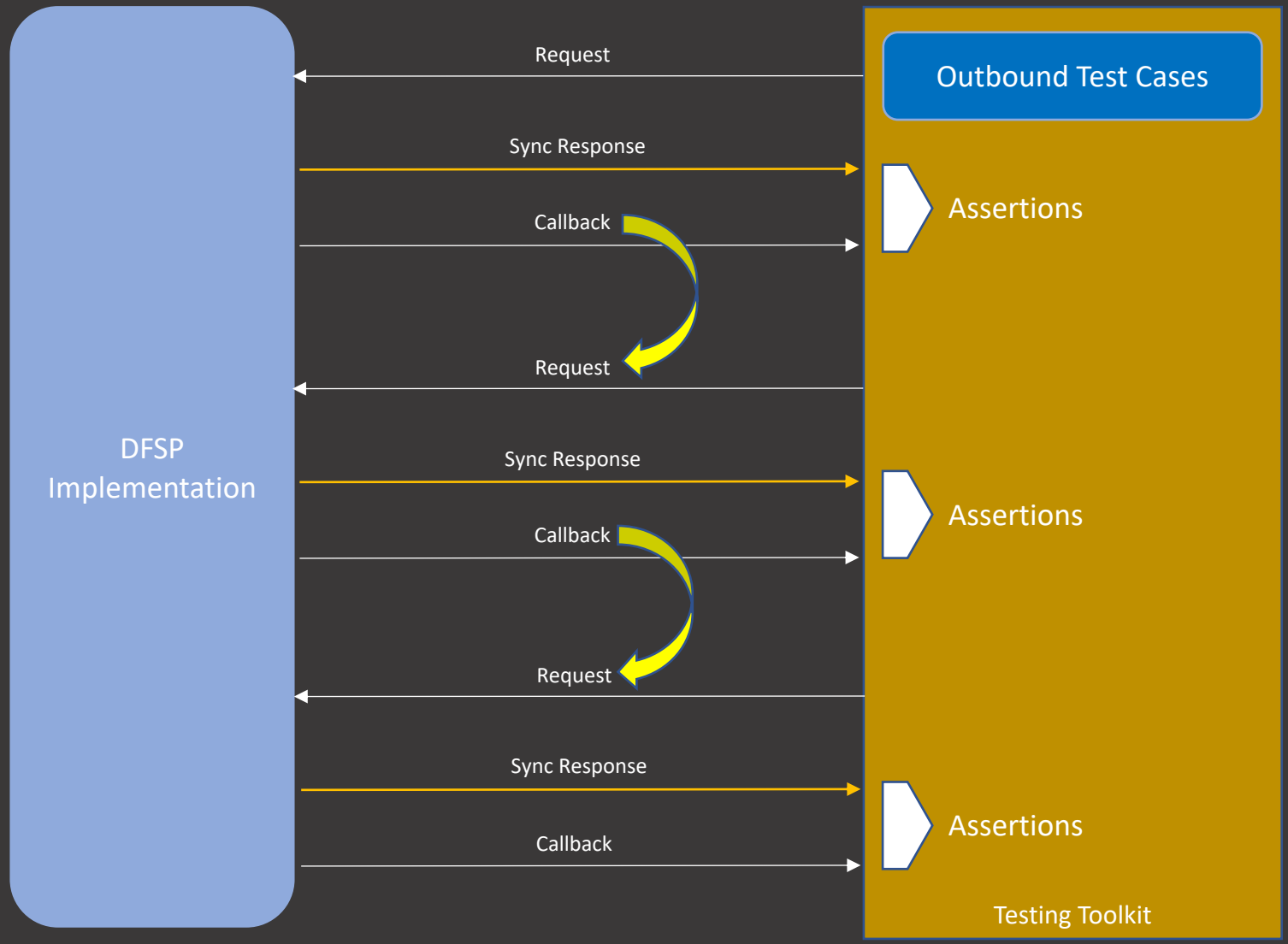
Breakdown the large test case file into folder structure



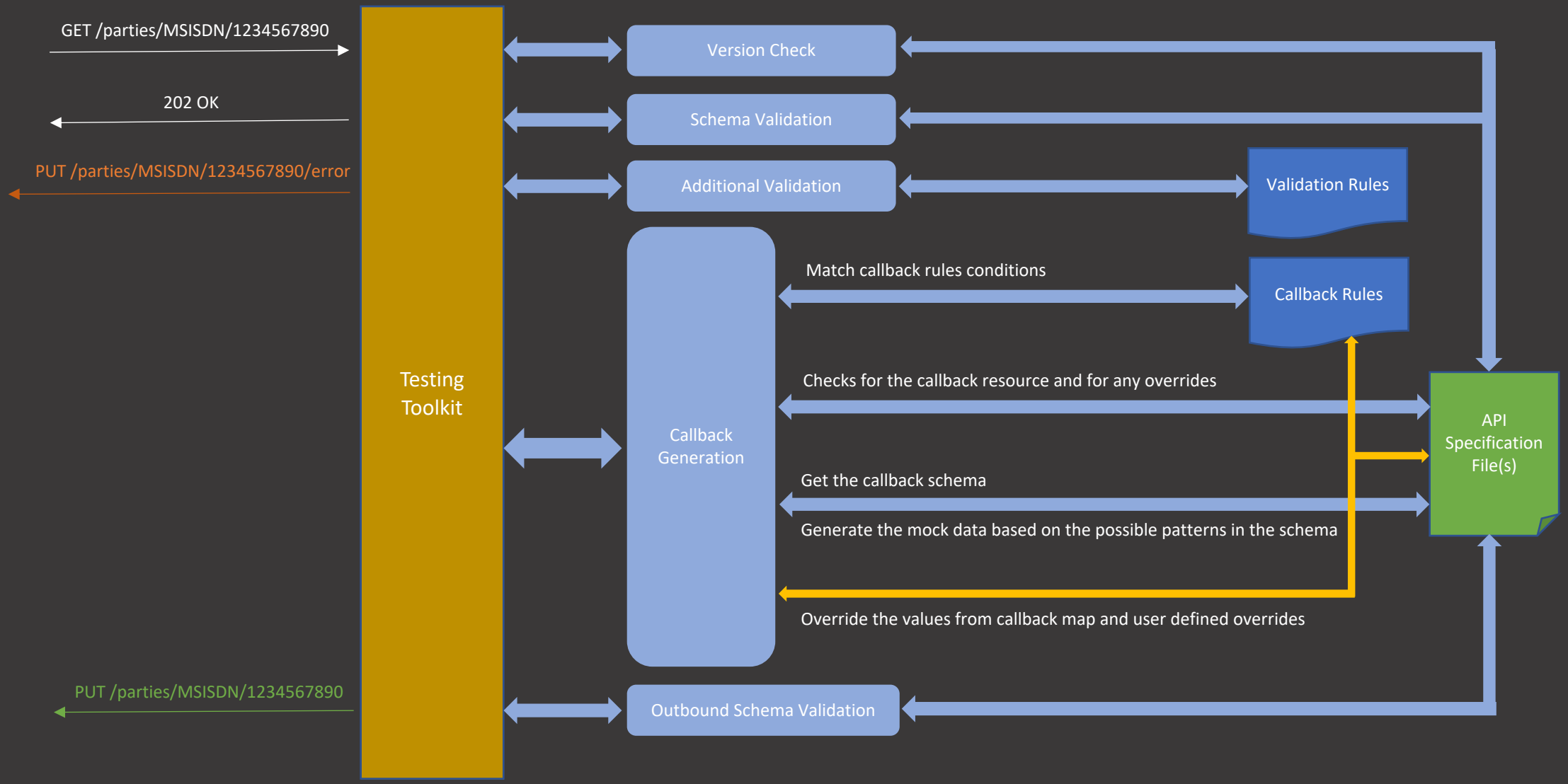
Integrating TTK into DFSP's CI, CD



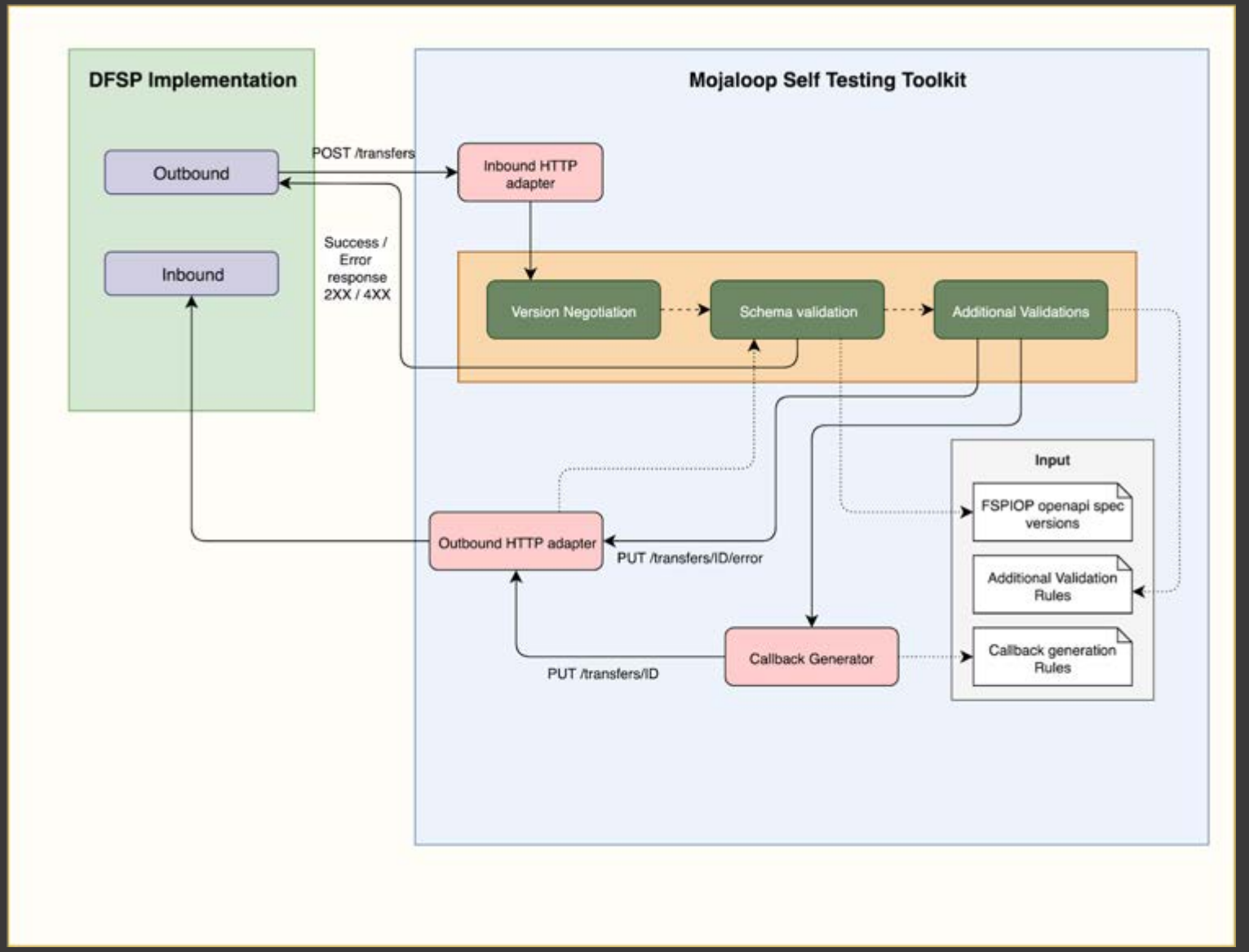
How Testing Toolkit Works - Test Case Initiation and Assertions



How Testing Toolkit Works - Incoming requests



Architecture Diagram – Part 1



Architecture Diagram – Part 2