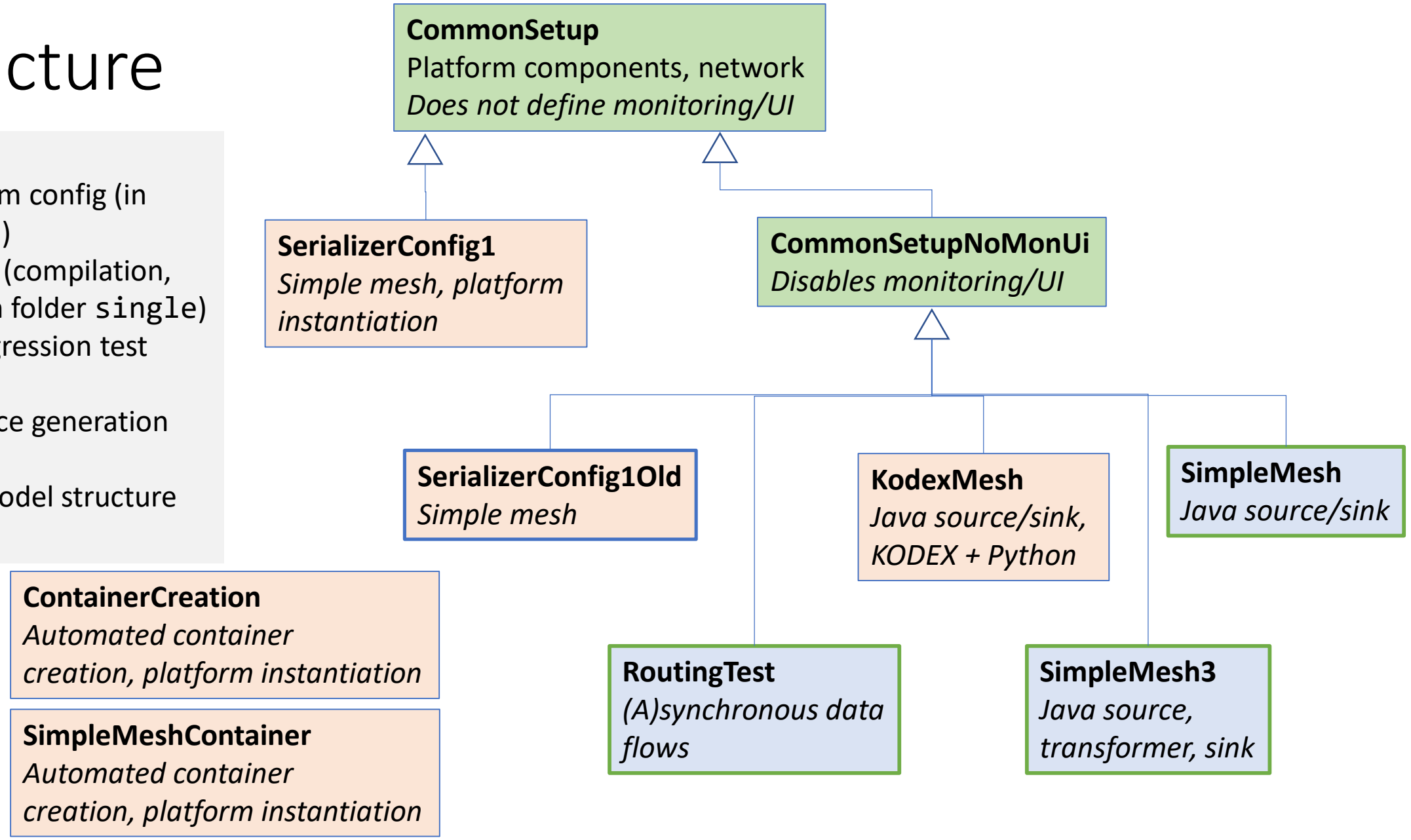
Purpose

- Structural regression tests
  - Does the build process work
  - Is source code compilable (Java, Python)
  - Is Yaml structure ok
  - Are generated files not empty
  - Are expected files there
- Preparation for execution regression tests (in dependent project `examples`)
- Mutual dependency on test.configuration.configuration implementing the services (based on generated interfaces, integrated via dynamic class loading)

Structure

Legend

- Shared platform config (in folder `common`)
- Structural test (compilation, file contents in folder `single`)
- Executable regression test (in examples)
- Shared interface generation (old style)
- In managed model structure (own folder)

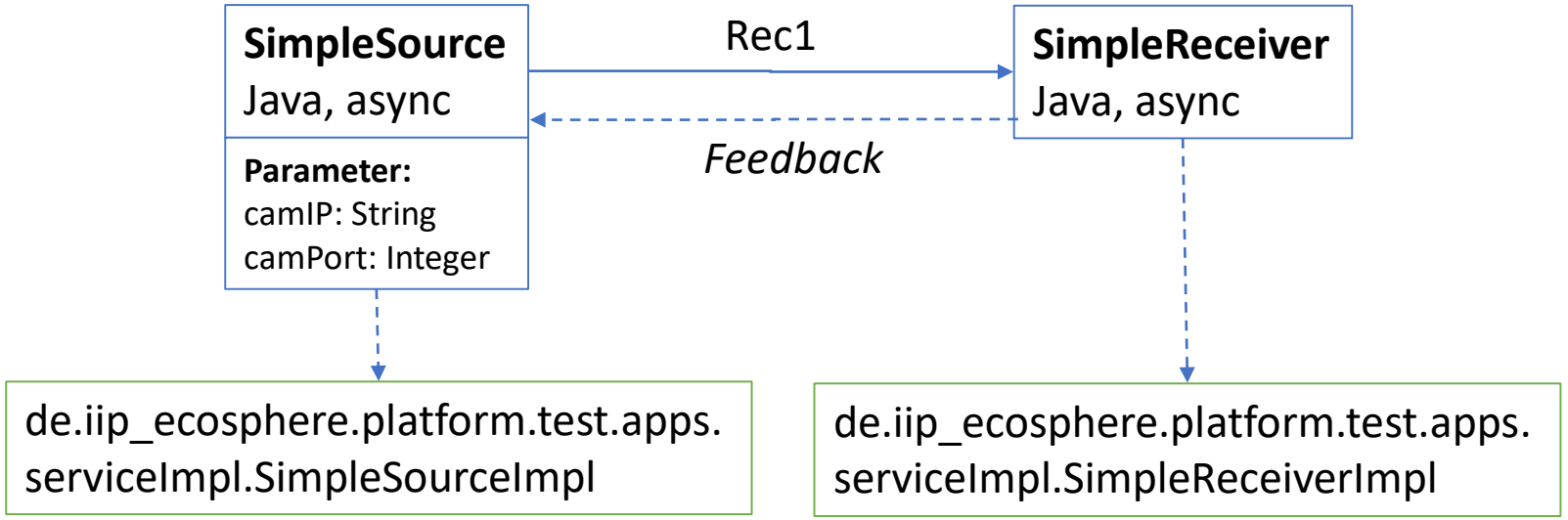


SimpleMesh

- Purpose:
  - Simple data flow among two services (source, sink)
  - Asynchronous communication
- Used also in the Install Package
- Used as regression test in `examples`

SimpleMesh

In test.configuration.configuration
de.iip-ecosphere.platform:
apps.ServiceImpl:
iipVer



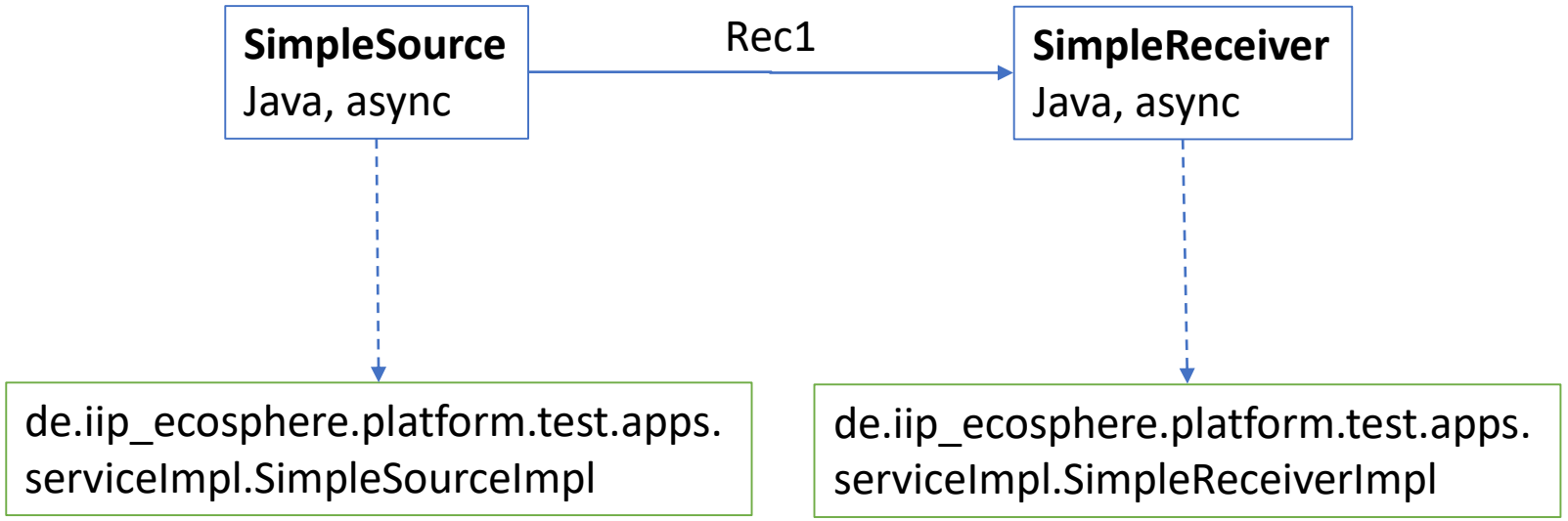
Feedback is an example for a simple (pub-sub, asynchronous) backward channel. Although not implemented used/here, it is (akin to the parameters of somple source) used for testing the management UI.

SimpleMeshContainer

- Like SimpleMesh, but with Container creation enabled

SimpleMeshContainer

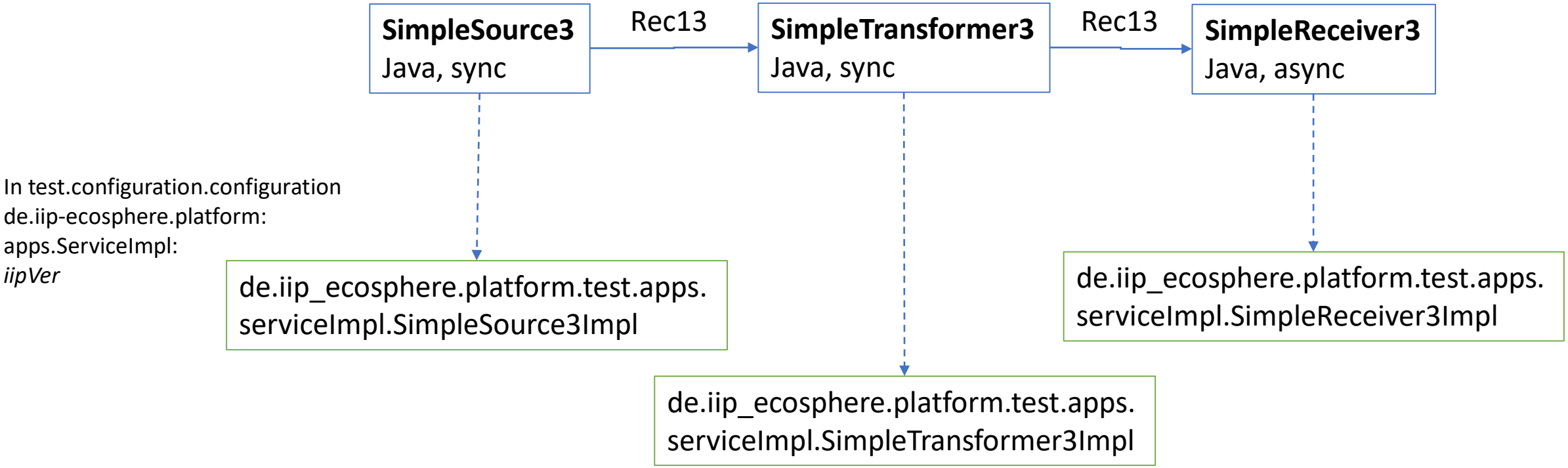
In test.configuration.configuration
de.iip-ecosphere.platform:
apps.ServiceImpl:
iipVer



SimpleMesh3

- Purpose:
  - Simple data flow among three services (source, transformer, sink)
  - Mostly synchronous communication
- Used as regression test in `examples`

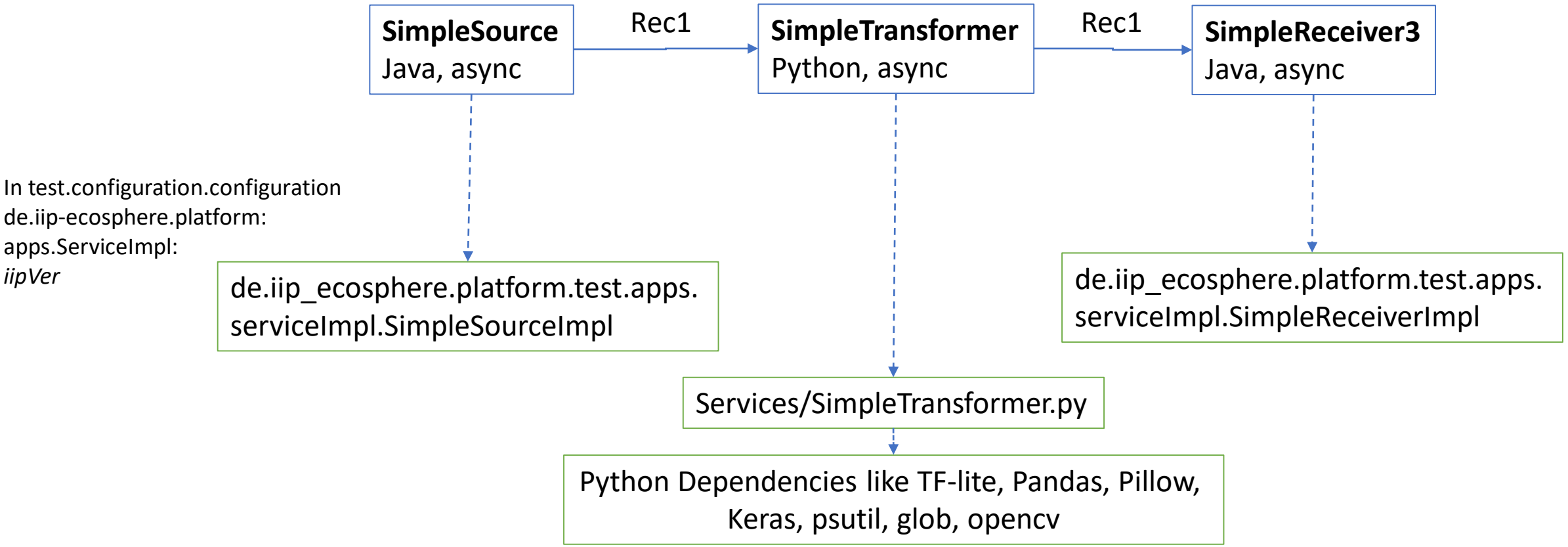
SimpleMesh3 (simple flow with transformer, partially sync)



ContainerCreation

- Based on SimpleMesh, also like SimpleMesh3 but with Python service
- Container creation with Python dependencies

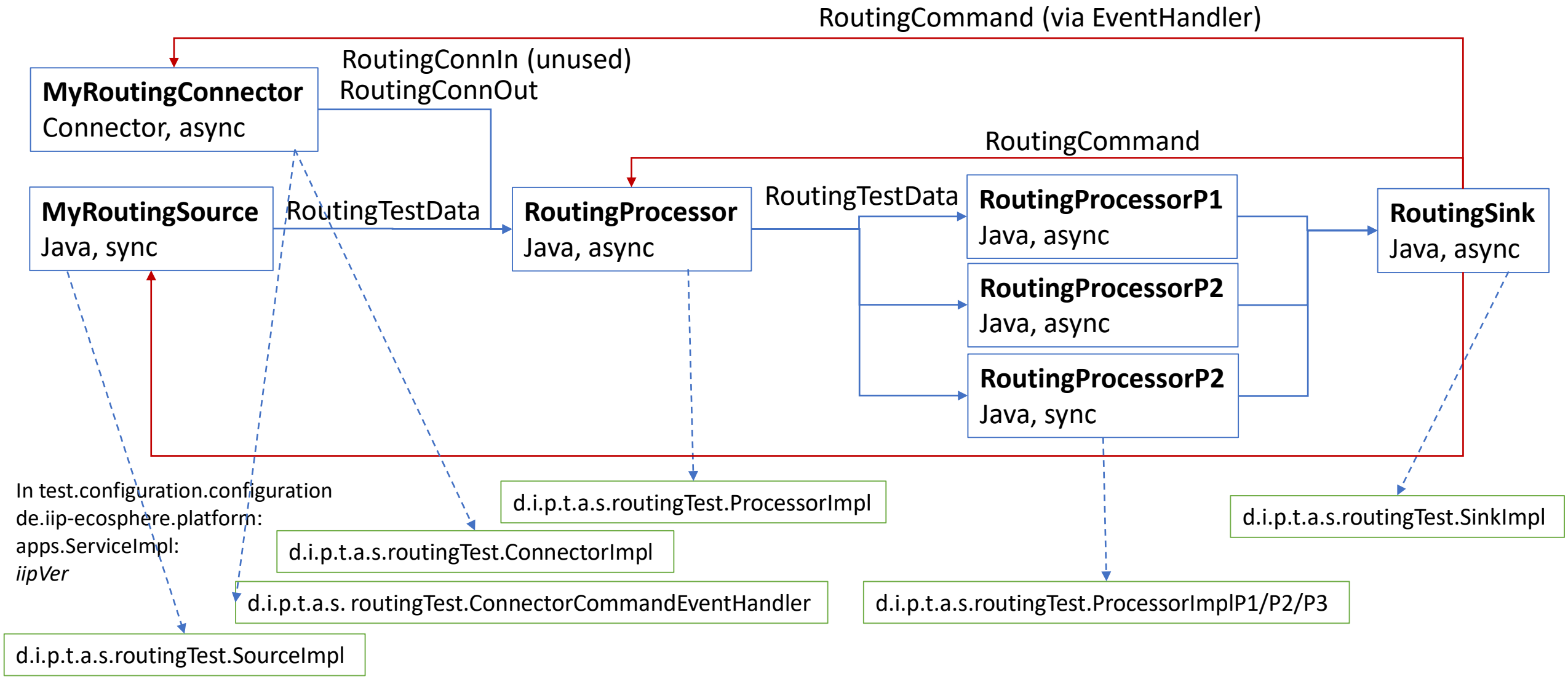
ContainerCreation



RoutingTest

- Purpose:
  - Forward and backward flows
  - Synchronou source
  - Handwritten connector reacting on backward flow
  - Parallel asynchronous paths
- Used as regression test in `examples`

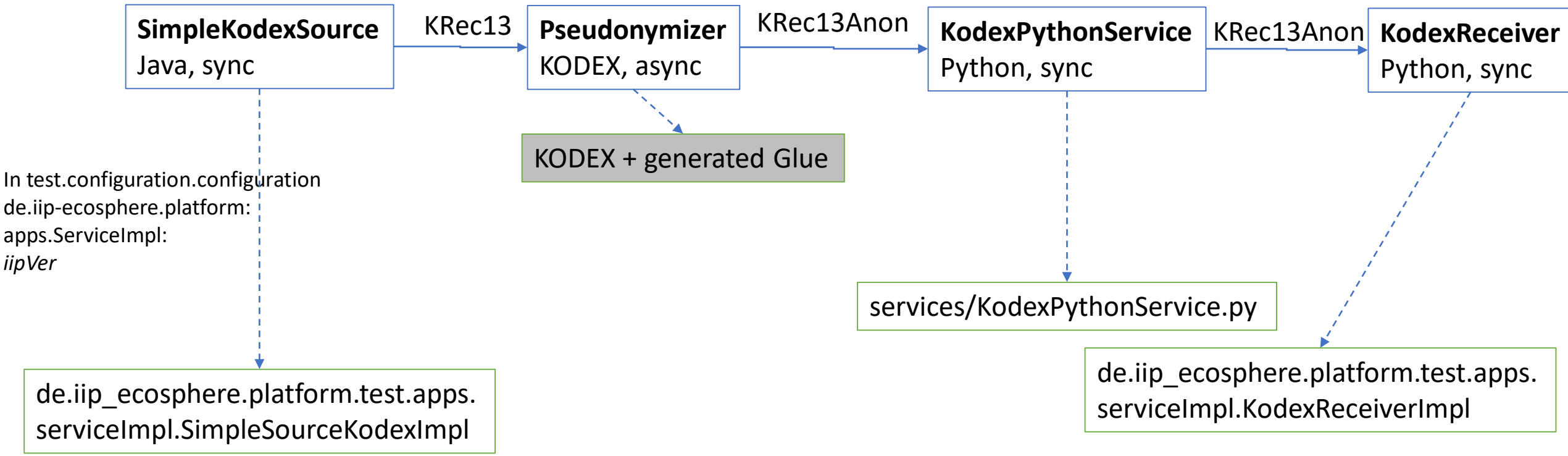
RoutingTest



KodexMesh

- Purpose:
  - Test KODEX integration
  - Later: Also Python, synchronous
- Purely structural test

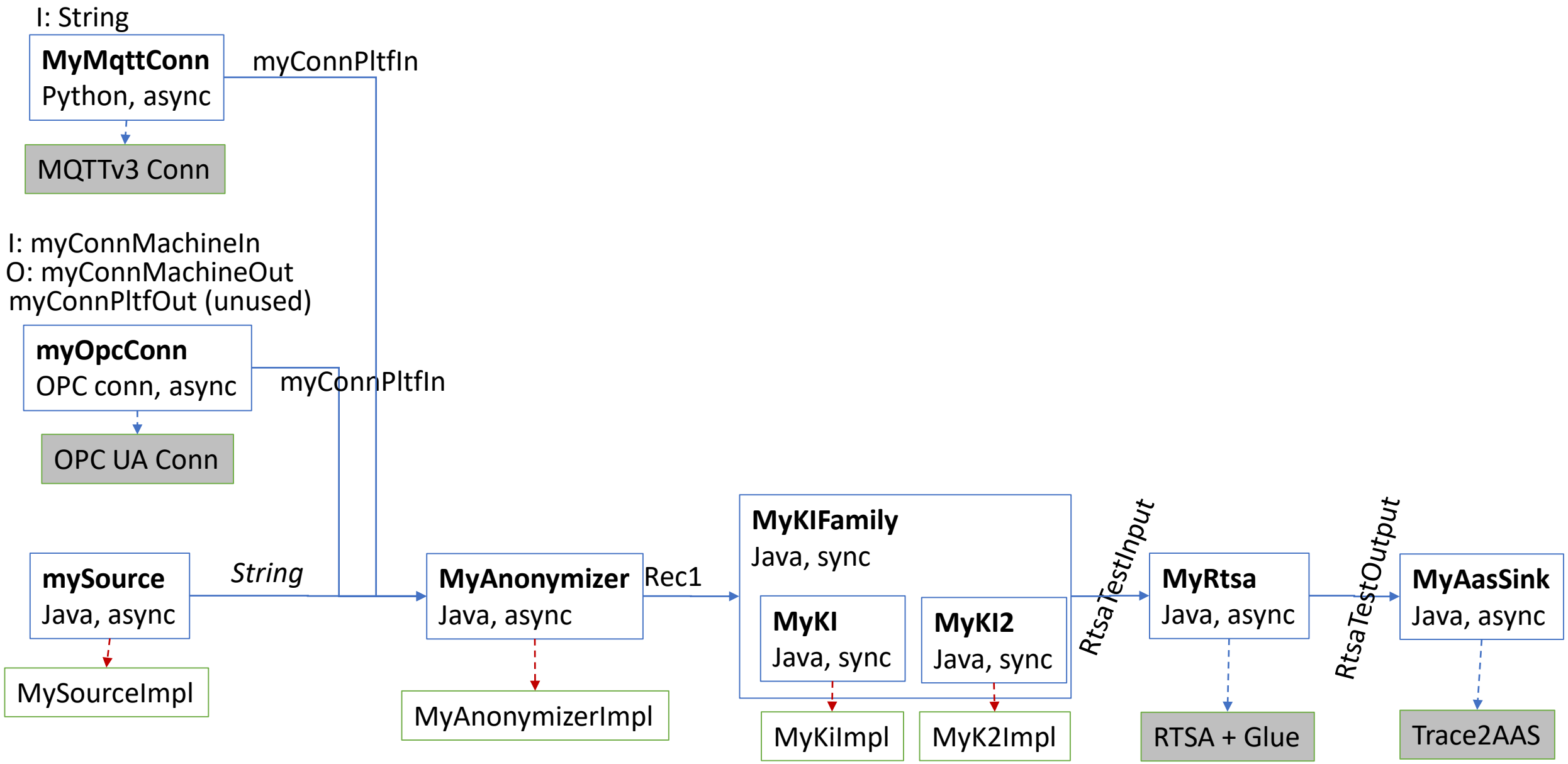
KodexMesh


SimpleKodexSource
Java, sync
KRec13
Pseudonymizer
KODEX, async
KRec13Anon
KodexPythonService
Python, sync
KRec13Anon
KodexReceiver
Python, sync
KODEX + generated Glue
In test.configuration.configuration
de.iip-ecosphere.platform:
apps.ServiceImpl:
iipVer
services/KodexPythonService.py
de.iip_ecosphere.platform.test.apps.
serviceImpl.SimpleSourceKodexImpl
de.iip_ecosphere.platform.test.apps.
serviceImpl.KodexReceiverImpl

SerializerConfig1

- Name: From original test that tested generating a serializer
- Purpose:
  - Two connectors with data transformation
  - RTSA integration
  - Service family
  - Service parameters
  - Applies object arrays to connectors
- Purely structural test (not executable, no implementations attached)

SerializerConfig1



SerializerConfig1-old

- Name:
  - From original test that tested generating a serializer
  - Derived from SerializerConfig1
  - Old-style not-shared interfaces
- Purpose:
  - Two connectors with data transformation
  - Simple data chain
- Purely structural test (not executable, no implementations attached)

SerializerConfig1-old (connector gen, impl. irrelevant)