Introduction to Orthanc server extensions
Orthanc Python Plugin Basics

Handle Orthanc (new DICOM, C-STORE request) events with Python callback functions

def OnChange(changeType, level, resource):
    if changeType == orthanc.ChangeType.ORTHANC_STARTED:
        ...

orthanc.RegisterOnChangeCallback(OnChange)

Great for routing, filtering, tag morphing
Orthanc Python Plugin

Drawbacks

- The `orthanc` Python module is only available at Orthanc runtime
- Can become a bit tedious to develop
  - Make a change
  - Restart Orthanc
  - Trigger the event
  - Check the logs
- Not so easy to write unit tests, need to mock the `orthanc` module everywhere
Orthanc-server-extensions
Design goals

• Faster development loop
• Support for (unit) testing
• Includes recipes for common production requirements
  • HTTP client with TLS and Authentication support
  • Logging
  • De/re-identification
  • Job retries
import logging

import orthanc
from orthanc_ext.event_dispatcher import register_event_handlers, create_session

def hello(event, session):
    logging.info(f"hello {event}")

    system_info = session.get('/system').json()
    return system_info['ApiVersion']

register_event_handlers(event_handlers={
    orthanc.ChangeType.ORTHANC_STARTED: hello
}, orthanc, create_session(orthanc))
Hello world: unit test

```python
@respx.mock
def test_hello_world():
    orthanc = OrthancApiHandler()
    client = create_internal_client('https://localhost:8043')

    respx.get('https://localhost:8043/system').respond(
        status_code=200, json={
            "ApiVersion": 18,
            "DicomAet": "ORTHANC",
            "DicomPort": 4242,
            "HttpPort": 8042,
            "IsHttpServerSecure": True
        }
    )

    event_dispatcher.register_event_handlers(
        event_handlers={orthanc.ChangeType.ORTHANC_STARTED: hello}, orthanc, client
    )

    sync_result = orthanc.on_change(
        orthanc.ChangeType.ORTHANC_STARTED, orthanc.ResourceType.NONE, resource_id=None
    )

    assert sync_result == [18]
```
Common AI deployment
Common DICOM AI workflow scenario
AI workflow

```python
def eligible_for_ai(resource_id, client):
    instance = get_metadata_of_first_instance_of_series(client, resource_id)
    return instance.Modality == 'MR' and instance.Origin == 'PACS'

def ai_pipeline(evt, client):
    orthanc_series_id = evt.resource_id
    if eligible_for_ai(orthanc_series_id, client):
        anonymize_series(client, orthanc_series_id)

    if anonymized_for_sending(orthanc_series_id, client):
        send_to_ai_service(orthanc_series_id, client)

def send_to_ai_service(series_id, client):
    client.post("/dicom-web/servers/orthanc-cloud/stow",
                 StowRequest([series_id]))

register_event_handlers(event_handlers={
    orthanc.ChangeType.ON_STABLE_SERIES: ai_pipeline,
}, orthanc, create_session(orthanc))
```
Common DICOM AI workflow scenario (refined)

1. DICOM series received from PACS
   - is MR from PACS and matches series description?
     - YES: de-identify series
     - NO: delete study

2. de-identify series
   - send series to AI service
     - series complete?
       - NO: perform reconstruction
         - check for complete AI series
           - AI series complete and no timeout occurred?
             - NO: fetch new AI series from AI service
               - re-identify new AI series
                 - store new AI series in PACS
           - YES: store new AI series in PACS
         - YES: store new AI series in PACS
  - store new AI series in PACS
Latest release

- Supports cloud workflow
- Process async functions with asyncio
- publish CloudEvents directly to NATS, RabbitMQ and Kafka
  - Events can be handled by external software
  - Logic can be implemented independently of orthanc
  - Begets all the advantages of event platforms
    - dynamically scale deployments
    - Implement retries
Future directions

• Higher level events to simplify workflows
• Performance evaluation of the asyncio implementation
References

Python plugin
• https://github.com/walkIT-nl/orthanc-server-extensions/
• https://orthanc-server-extensions.readthedocs.io/en/latest/

Python plugin