Alfresco Out of Process application with python

Point of view: support: identifying/understanding problems

Plan

1. ActiveMQ protocols
2. Out of Process application with python and STOMP
3. Mirroring queues into topics

ActiveMQ protocols

active MQ transport connectors and protocols

- amqp: Java Advanced Message Queuing Protocol (see also JMS
- stomp: python (and many many others, including java)

See activemq.xml:

```
<!--
    The transport connectors expose ActiveMQ over a given protocol to
    clients and other brokers. For more information, see:

    http://activemq.apache.org/configuring-transports.html
-->
<transportConnectors>
    <!-- DOS protection, limit concurrent connections to 1000 and frame size to 100MB -->
    <transportConnector name="openwire" uri="tcp://0.0.0.0:61616?maximumConnections=1000&amp;wireFormat.maxFrameSize=104857600"/>
    <transportConnector name="amqp" uri="amqp://0.0.0.0:5672?maximumConnections=1000&amp;wireFormat.maxFrameSize=104857600"/>
    <transportConnector name="stomp" uri="stomp://0.0.0.0:61613?maximumConnections=1000&amp;wireFormat.maxFrameSize=104857600"/>
    <transportConnector name="mqtt" uri="mqtt://0.0.0.0:1883?maximumConnections=1000&amp;wireFormat.maxFrameSize=104857600"/>
    <transportConnector name="ws" uri="ws://0.0.0.0:61614?maximumConnections=1000&amp;wireFormat.maxFrameSize=104857600"/>
</transportConnectors>
```

STOMP

STOMP = Streaming Text Oriented Messaging Protocol

https://stomp.github.io/

https://en.wikipedia.org/wiki/Streaming_Text_Oriented_Messaging_Protocol

```
STOMP is a very simple and easy to implement protocol, coming from the
HTTP school of design;
```

https://stomp.github.io/implementations.html

we will use `stomp-py`:

https://github.com/jasonrbriggs/stomp.py

https://pypi.org/project/stomp-py/

ActiveMQ and python

ActiveMQ and python: why?

- you can use python to consume the event2 topic *out of process apps* (**SDK is not mandatory**)
- you can use python to *debug* the queue (transformers) with AMQ mirrors
- easy

Benefits of *loose coupling*: performance + freedom

ActiveMQ and python: event2 demo

DEMO1

Active MQ queue mirrors

https://activemq.apache.org/mirrored-queues.html

add in activemq.xml

```
<destinationInterceptors>
    <mirroredQueue copyMessage = "true" postfix=".qmirroralex" prefix=""/>
</destinationInterceptors>
```

Active MQ rendition and transformer queues mirroring

Make sure you use remote transforms (not local):

```
transform.service.enabled=true
local.transform.service.enabled=false
```

https://hyland.atlassian.net/browse/MNT-23454

https://hyland.atlassian.net/browse/MNT-23478

Note on debug server (ACS) side: messages org.apache.camel DEBUG are truncated

- camel is an apache project https://camel.apache.org/ (https://github.com/apache/camel/tree/main)
- set DEBUG an org.apache.camel at http://localhost:8080/alfresco/s/enterprise/admin/admin-log-settings

```
2023-06-13 11:47:00,734  DEBUG [component.jms.JmsConfiguration] [eventAsyncDequeueThreadPool1]
Sending JMS message to: topic://alfresco.repo.event2 with message: ActiveMQTextMessage
...
text = {"specversion":"1.0","type":"org.alfresco.eve...rities":[]}}}
```

(DEMO2)

Note: you can also develop custom T-engine

Again: Benefits of *loose coupling*: performance + freedom

- in Java (another SDK)
  https://docs.alfresco.com/content-services/latest/develop/repo-ext-points/content-transformers-renditions/#developing-a-new-t-engine
  https://github.com/Alfresco/acs-packaging/blob/master/docs/creating-a-t-engine.md
- in python: no SDK but protocol is simple

QA

Thank You!