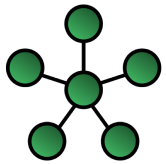
Messaging & Replication Extension

Dannes Wessels
dannes@exist-db.org



Agenda

- Technology
- Document replication
- Messaging
- Wrap-up

About Me

- eXist-db contributor since 2004
- Co-founder eXist-solutions
- Many extension and features

dannes@exist-db.org

Technology

- JMS: Java Messaging Service
  - Industry accepted technology
- Built with Apache ActiveMQ
  - Strong community
  - Well maintained

Document Replication

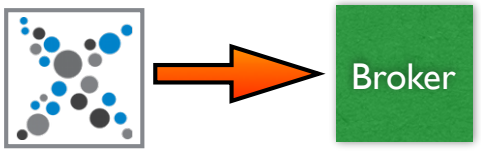
Document replication

- Goal:
  - Scale up eXist-db
- Implementation:
  - Multiple eXist instances in one cluster

Document replication

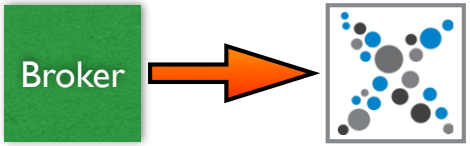
Master Broker Slave

Document Replication



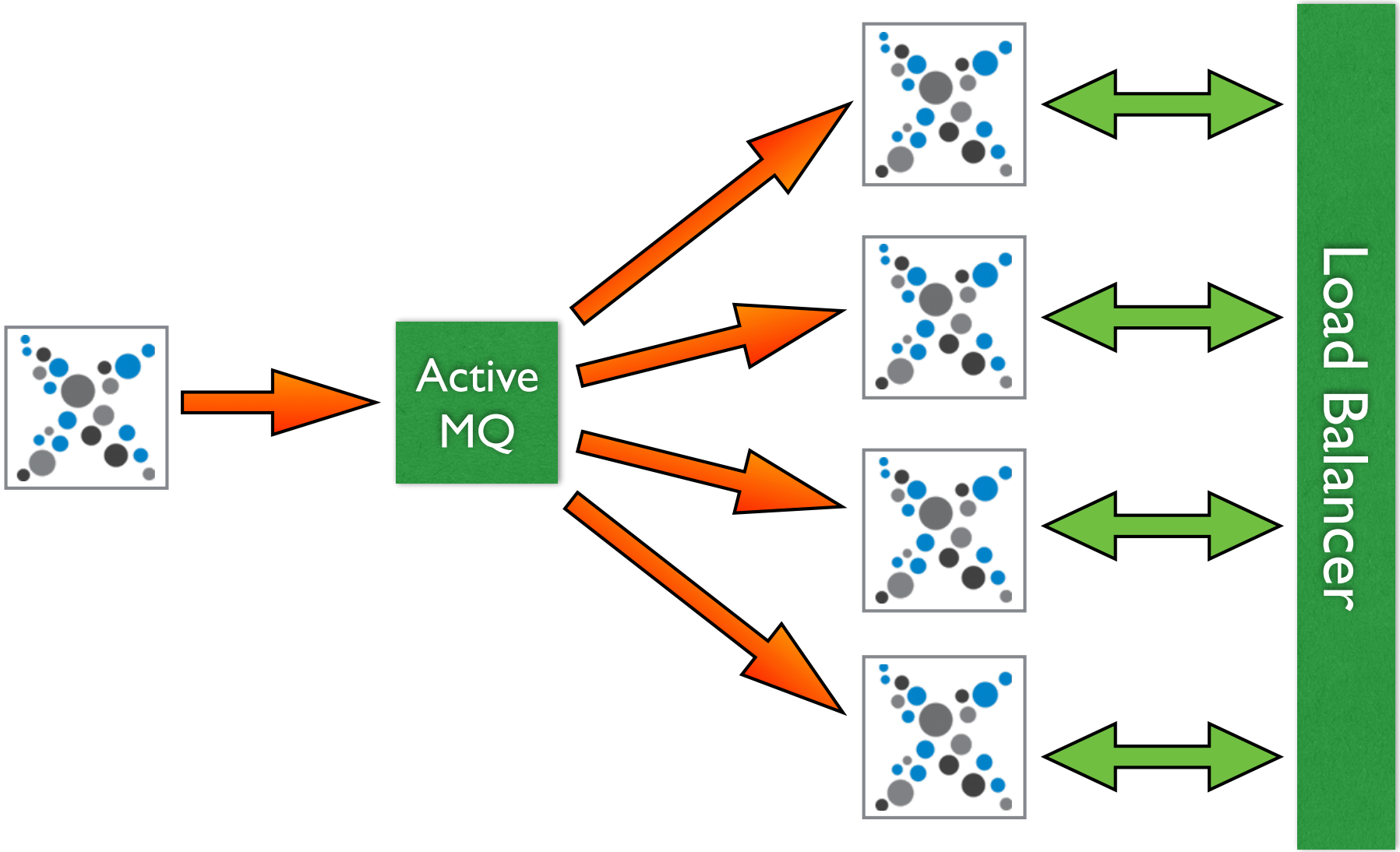
- Sequence “Master”
  - Document / collection operation fires “ReplicationTrigger”
  - Aggregate change details
    - Serialize & compress document
  - Send JMS message to Broker

Document Replication



- Sequence "Slave"
  - Receive JMS message
  - Determine operation on resource type
    - Decompress document
  - Persist resource change to database

Document replication

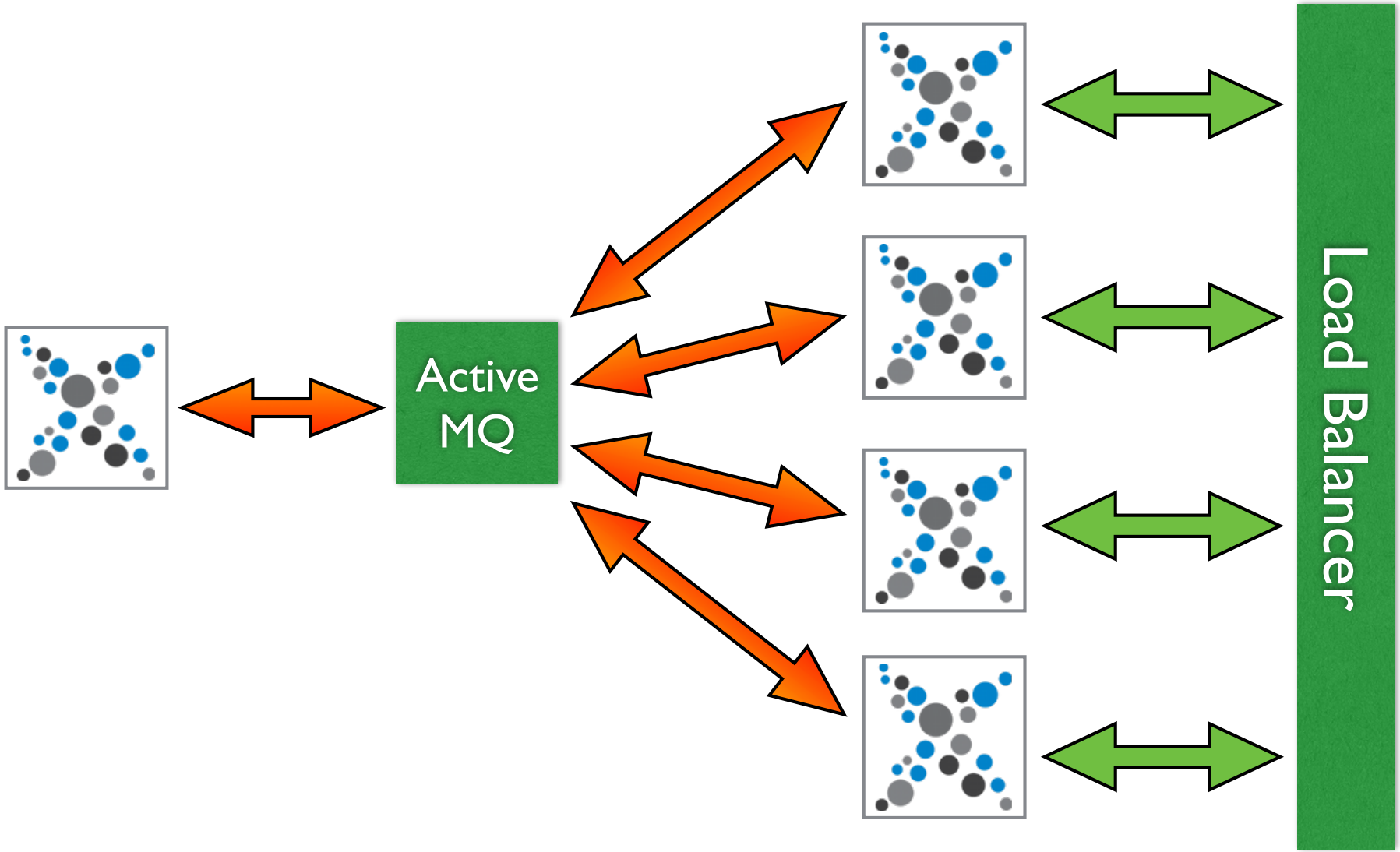


Master Broker Slaves

Document replication

- 1 Master - 1..N Slaves
  - Very scalable
  - No editing on slaves
  - Load balancer session management

Document replication



Each instance is Master + Slave

Document replication

- N Masters, N slaves
  - Editing on each eXist-db
  - Note: no LOCK replication!
  - Load balancer session management

How to setup

Add to collection.xconf of a collection:

```
<trigger class="org.exist.jms.replication.publish.ReplicationTrigger">

  <parameter name="java.naming.factory.initial"
             value="org.apache.activemq.jndi.ActiveMQInitialContextFactory"/>

  <parameter name="java.naming.provider.url"
             value="tcp://localhost:61616"/>

  <parameter name="connection-factory"
             value="ConnectionFactory"/>

  <parameter name="destination"
             value="dynamicTopics/eXistdb-replication-example"/>

</trigger>
```

Setup slaves

- Register 'listener' at server start:
  - conf.xml
  - with script and XQueryStartupTrigger

Add to triggers/startup in conf.xml:

```
<trigger class="org.exist.jms.replication.subscribe.ReceiverStartupTrigger">

    <parameter name="java.naming.factory.initial"
        value="org.apache.activemq.jndi.ActiveMQInitialContextFactory"/>

    <parameter name="java.naming.provider.url"
        value="tcp://localhost:61616"/>

    <parameter name="connection-factory"
        value="ConnectionFactory"/>

    <parameter name="destination"
        value="dynamicTopics/eXistdb-replication-example"/>

    <parameter name="subscriber.name"
        value="SubscriptionId"/>

    <parameter name="connection.client-id"
        value="ClientId"/>

</trigger>
```

With xQuery script

```
import module namespace replication="http://exist-db.org/xquery/replication"
                at "java:org.exist.jms.xquery.ReplicationModule";

let $jmsConfiguration := map {
        "java.naming.factory.initial" := "org.apache.activemq.jndi.ActiveMQInitialContextFactory",
        "java.naming.provider.url"    := "tcp://localhost:61616",
        "connection-factory"          := "ConnectionFactory",
        "destination"                 := "dynamicTopics/eXistdb-replication-example",
        "subscriber.name"             := "SubscriptionId",
        "connection.client-id"        := "ClientId"
    }

return
    replication:register($jmsConfiguration)
```

Use generic startup trigger

```
<trigger class="org.exist.collections.triggers.XQueryStartupTrigger"/>
```

Messaging

- Integrate with corporate message bus
  - Send Messages
  - Receive messages

Example

```
import module namespace messaging="http://exist-db.org/xquery/messaging"
               at "java:org.exist.jms.xquery.MessagingModule";

let $jmsConfiguration := map {
        "java.naming.factory.initial" := "org.apache.activemq.jndi.ActiveMQInitialContextFactory",
        "java.naming.provider.url"    := "tcp://localhost:61616",
        "destination"                 := "dynamicQueues/eXistdb-messaging-example",
        "connection-factory"          := "ConnectionFactory"
    }

let $messageProperties := map {"City" := "Prague" }

let $content := <data>XML conference</data>

return
   messaging:send( $content , $messageProperties, $jmsConfiguration )
```

Receive messages

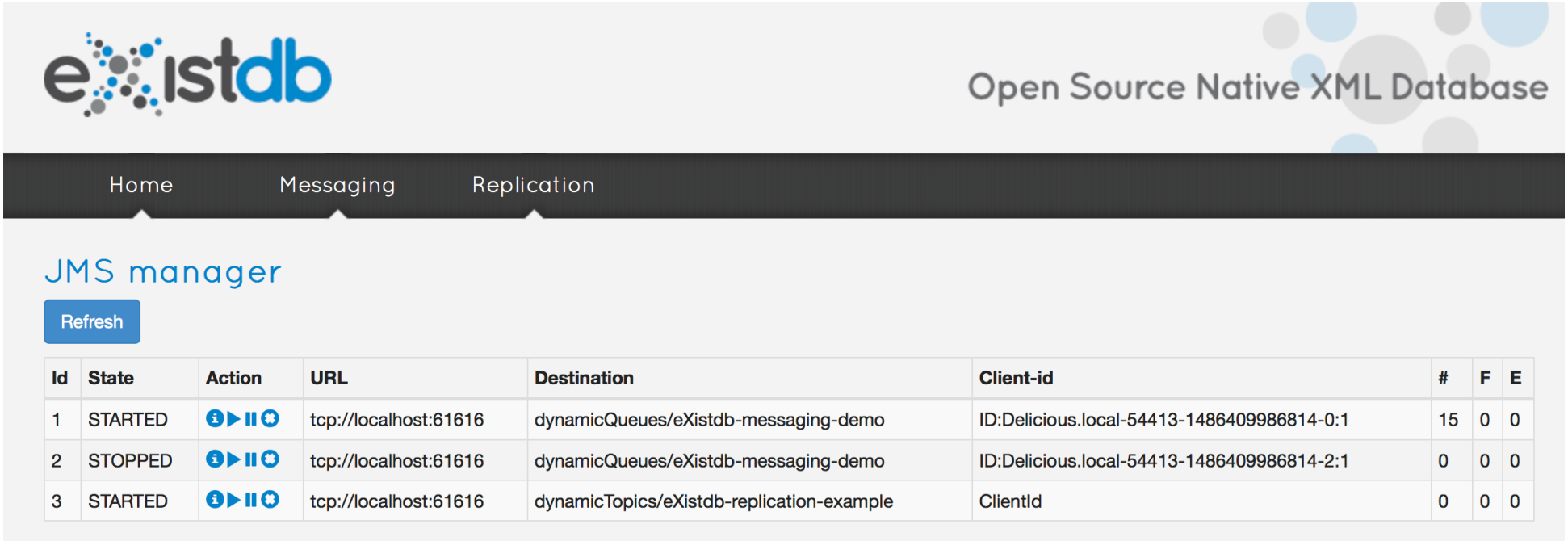
- Handle message with HoF
- Register HoF with xQuery function
- Use XQueryStartupTrigger

Example

```
import module namespace messaging="http://exist-db.org/xquery/messaging"
                at "java:org.exist.jms.xquery.MessagingModule";

declare function local:handleMessage($content as item(), $params as item()*,
                        $messageProperties as map(), $jmsConfig as map() )
{
        util:log-system-out( data($content) )
};

let $jmsConfiguration := map {
        "java.naming.factory.initial":= "org.apache.activemq.jndi.ActiveMQInitialContextFactory",
        "java.naming.provider.url"   := "tcp://localhost:61616",
        "destination"                := "dynamicQueues/eXistdb-messaging-example",
        "connection-factory"         := "ConnectionFactory"
    }

let $callback := local:handleMessage#4

let $additionalParameters := (1, "2" , xs:float(3.0))

return
    messaging:register($callback , $additionalParameters, $jmsConfiguration)
```

Listener Management

http://localhost:8080/exist/apps/messaging-replication/manage.html



Id	State	Action	URL	Destination	Client-id	#	F	E
1	STARTED		tcp://localhost:61616	dynamicQueues/eXistdb-messaging-demo	ID:Delicious.local-54413-1486409986814-0:1	15	0	0
2	STOPPED		tcp://localhost:61616	dynamicQueues/eXistdb-messaging-demo	ID:Delicious.local-54413-1486409986814-2:1	0	0	0
3	STARTED		tcp://localhost:61616	dynamicTopics/eXistdb-replication-example	ClientId	0	0	0

Wrap-up

Outlook

- Release ‘final’ versions
- Upgrade to latest JARs
- Move from generic JMS to ActiveMQ specific features

Q & A