



Integration Tests with Docker, Scala, Akka & ScalaTest



**Scala Exchange 2014
Andreas Gies**



- Technical Consulting
- Conference speaking slots
- Technical Trainings
- Management Seminars
- Team Breakouts



Coding by the sea



Yoga & learning



Team Building



Enjoy Andalucia

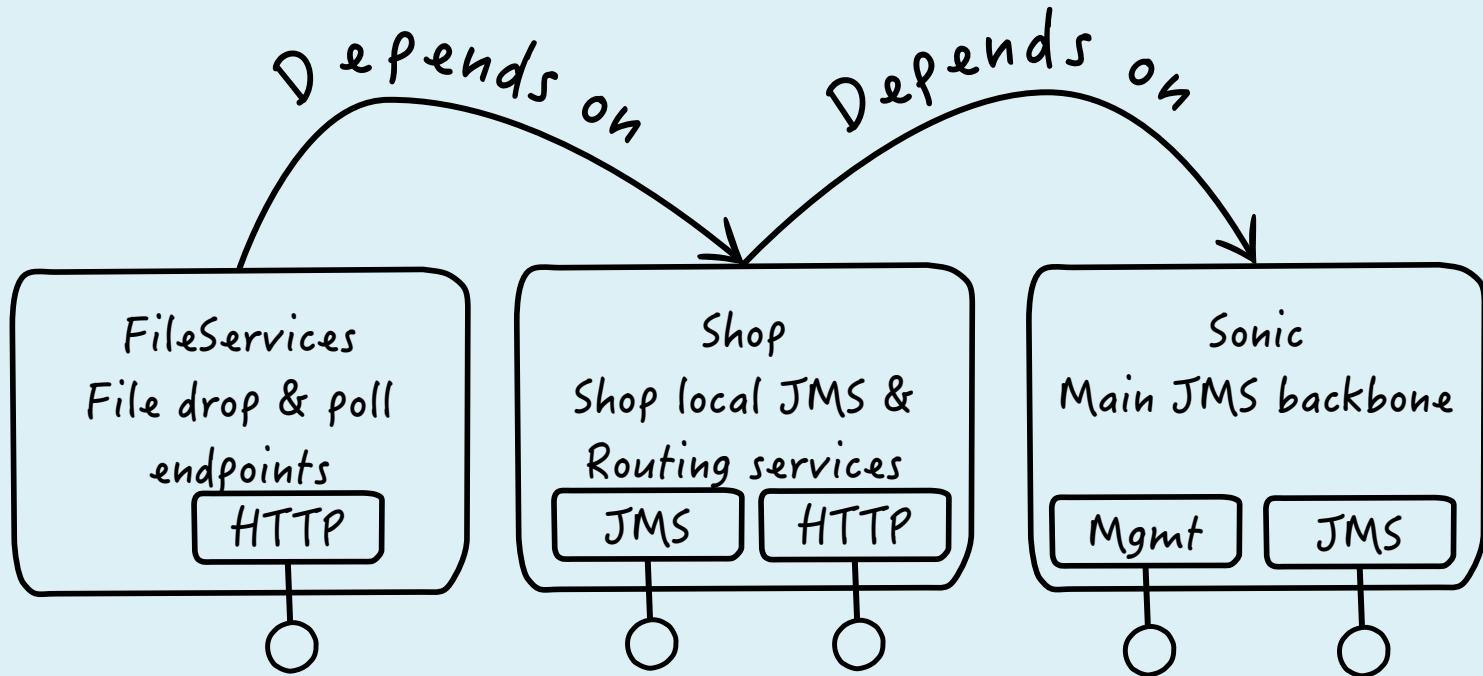
Project context

- Java based, distributed retail application
 - Overall functionality as Messaging Backbone
 - Different container types in collaboration
- ~7 years in development, OSGi based
- Migration to OSGi/Scala/Akka since 2011
 - ~80% Scala by now
- Inherited with weak Specs, manual Tests
- Scala/Akka version in production since 06/2013
 - 8 countries, ~3000 stores, ~20000 containers



SAMPLE TEST

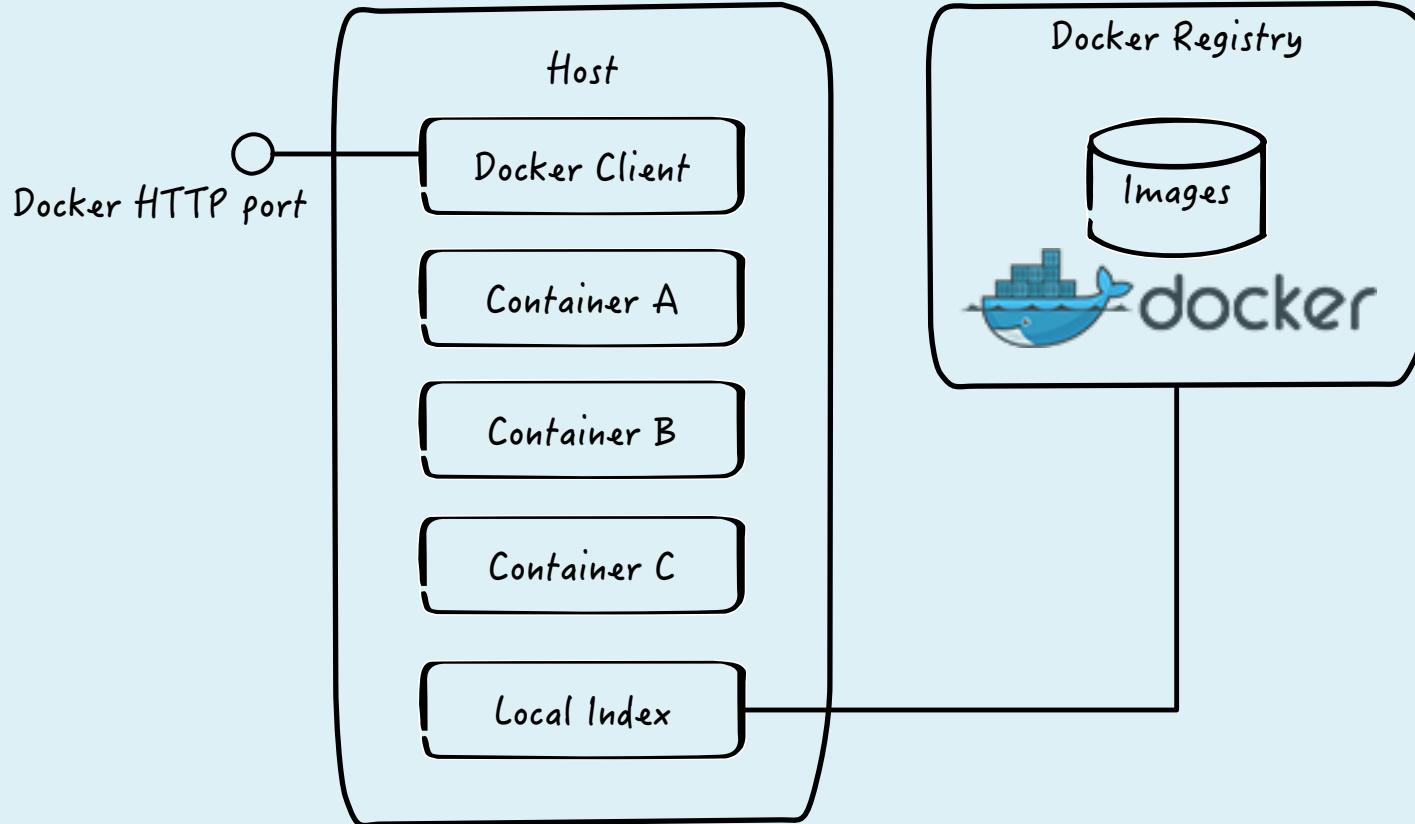
A sample test scenario





CREATING DOCKER IMAGES

Docker in a nutshell



Layering Docker images

Application
Distributables & Configuration

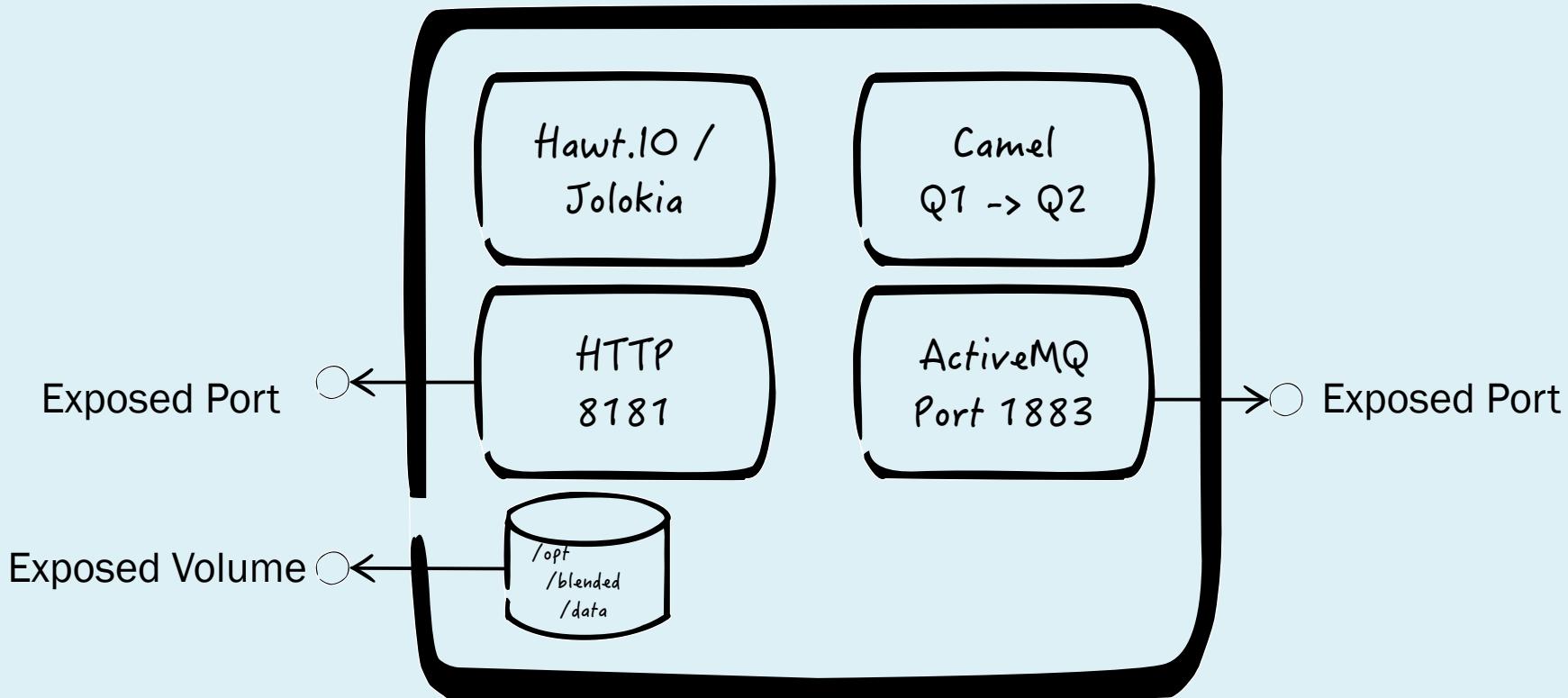
Customization
i.e. Additional libs, tools, services etc

Fitting base image from Registry

Docker container configuration

```
FROM atooni/blended-base:latest
ADD blended-karaf-demo-${project.version}-nojre.tar.gz /opt
RUN ln -s /opt/blended-karaf-demo-${project.version} /opt/blended
RUN mkdir -p /opt/blended/data
RUN chown -R blended.blended /opt/blended*
USER blended
ENV JAVA_HOME /opt/java
ENV PATH ${PATH}:${JAVA_HOME}/bin
ENTRYPOINT ["/opt/blended/bin/karaf"]
VOLUME /opt/blended/data
EXPOSE 1883
EXPOSE 8181
EXPOSE 1099
```

A container under test



Test packaging



↓ Maven Assembly plugin

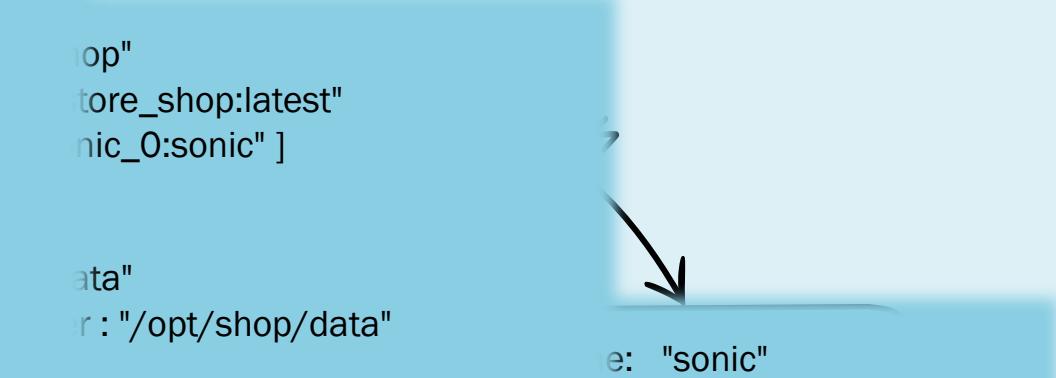
Distributable Archive
tar.gz / zip

↓ Maven Docker Plugin

Runnable docker image

Defining the CuT

```
{  
    name: "gs4"  
    image: "^store_sibfile:latest"  
    links: [ "shop_0:shop" ]  
    ports: [  
        { name: "http", value: "8181" }  
    ]  
    volumes: [  
        {  
            host: "data"  
            container: "/opt/sibfile/data"  
        },  
        {  
            host: "tmp"  
            container: "/tmp"  
        }  
    ]  
}
```

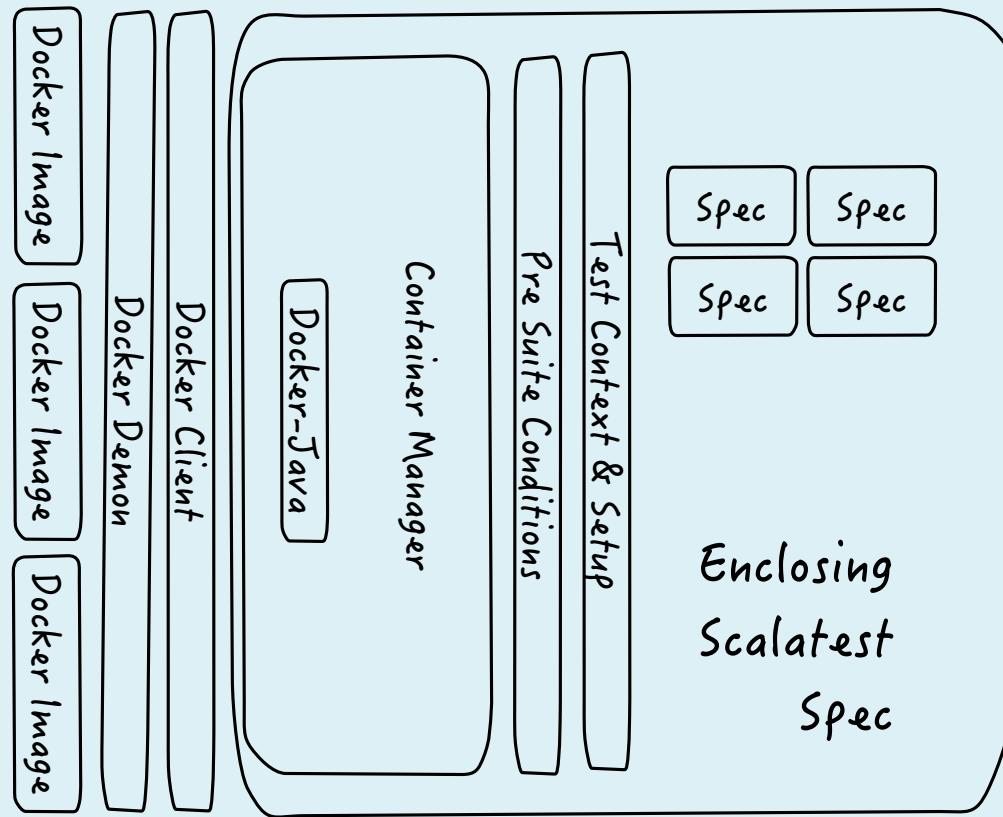


```
top"  
tore_shop:latest"  
nic_0:sonic" ]  
  
ata"  
r :"/opt/shop/data"  
  
tp", value: "4747" },  
http", value: "8777" },  
ms", value: "1883" }  
  
e: "sonic"  
ge: "^store_sonic:latest"  
s :[  
ame: "data", value: "2620" },  
ame: "mgmt", value: "2608" }  
  
https://github.com/woq-blended/blended/blob/master/blended-akka-itest/src/test/resources/application.conf
```



INTEGRATION TEST ELEMENTS

High Level Test Architecture



General tasks to write tests

- Define the containers under test
- Start the CuT and wait until they are ready
- Provide a blackbox test context
- Execute the test suite

The enclosing Spec

```
class BlendedDemoIntegrationSpec extends TestActorSys
  with SpecLike
  with BlendedIntegrationTestSupport {

  beforeSuite()

  override def nestedSuites = IndexedSeq(new BlendedDemoSpec)

  override def preCondition = {
    val t = 60.seconds

    SequentialComposedCondition(
      JMSAvailableCondition(amqConnectionFactory, Some(t)),
      JolokiaAvailableCondition(jmxRest, Some(t), Some("blended"), Some("blended")),
      CamelContextExistsCondition(
        jmxRest, Some("blended"), Some("blended"), "BlendedSample", Some(t)
      )
    )
  }
}
```

CamelTestSupport

- Create & send test messages
- Wire mock Endpoints
- Frequently used message assertions

The example test context

Provide a Camel based test context with all required components for testing

```
private def testContext = {
    val result = new TestCamelContext()
    result.withComponent(
        "jms", JmsComponent.jmsComponent(
            BlendedTestContext(
                BlendedDemoIntegrationSpec.amqConnectionFactory)
            .asInstanceOf[ConnectionFactory]
        )
    )
    result
}
```

Providing the JMS connection

- We know the JMS port is 1883 and Active MQ is the JMS provider
- Keep information relevant across Specs in an instance of BlendedTestContext

```
private lazy val amqConnectionFactory = {
    val ctInfo = Await.result(containerInfo("blended_demo_0"), 3.seconds)
    val address = ctInfo.getNetworkSettings.getIpAddress

    val brokerUrl = s"tcp://$address:1883"

    system.log.info(s"Using AMQ connection url [$brokerUrl]")

    BlendedTestContext.set(
        BlendedDemoIntegrationSpec.amqConnectionFactory,
        new ActiveMQConnectionFactory(brokerUrl)
    ).asInstanceOf[ConnectionFactory]
}
```

Waiting for the container

Wait until Jolokia, JMS and the Camel Route under test are available

```
override def precondition = {
    val t = 60.seconds

    SequentialComposedCondition(
        ParallelComposedCondition(
            JolokiaAvailableCondition(jmxRest, Some(t), Some("blended"), Some("blended")),
            JMSAvailableCondition(amqConnectionFactory, Some(t))
        ),
        CamelContextExistsCondition(
            jmxRest, Some("blended"), Some("blended"), "BlendedSample", Some(t)
        )
    )
}
```

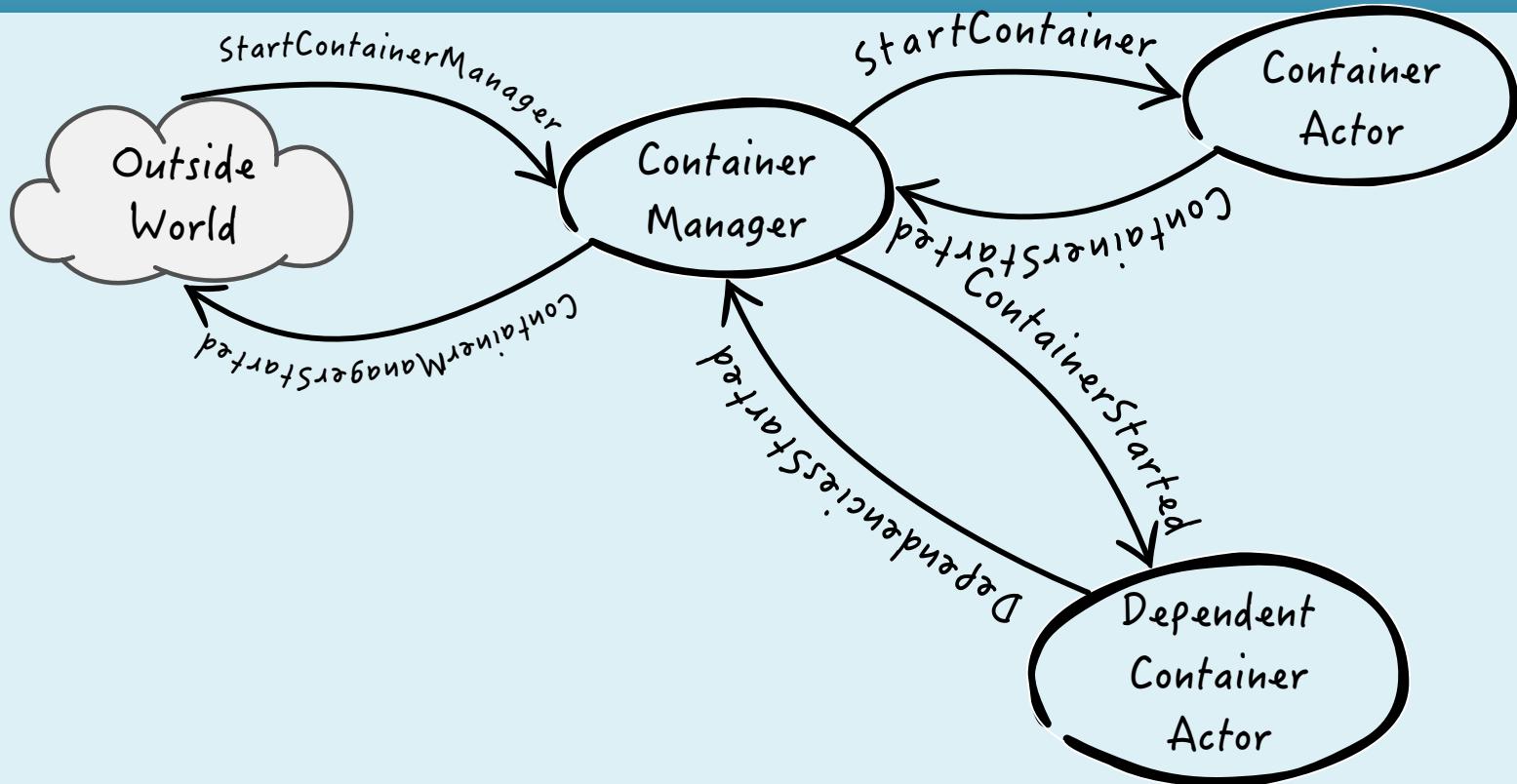
Finally, the spec

```
"The demo container" should {
    "Define the sample Camel Route from SampleIn to SampleOut" in {
        implicit val camelContext = testContext
        withTestContext { ctxt =>
            ctxt.withMock("sampleOut", "jms:queue:SampleOut")
            ctxt.start()
            val mock = ctxt.mockEndpoint("sampleOut")
            mock.setExpectedMessageCount(1)
            mock.expectedBodyReceived().constant("Hello Blended!")
            ctxt.sendTestMessage("Hello Blended!", "jms:queue:SampleIn")
            mock.assertIsSatisfied(2000l)
        }
    }
}
```

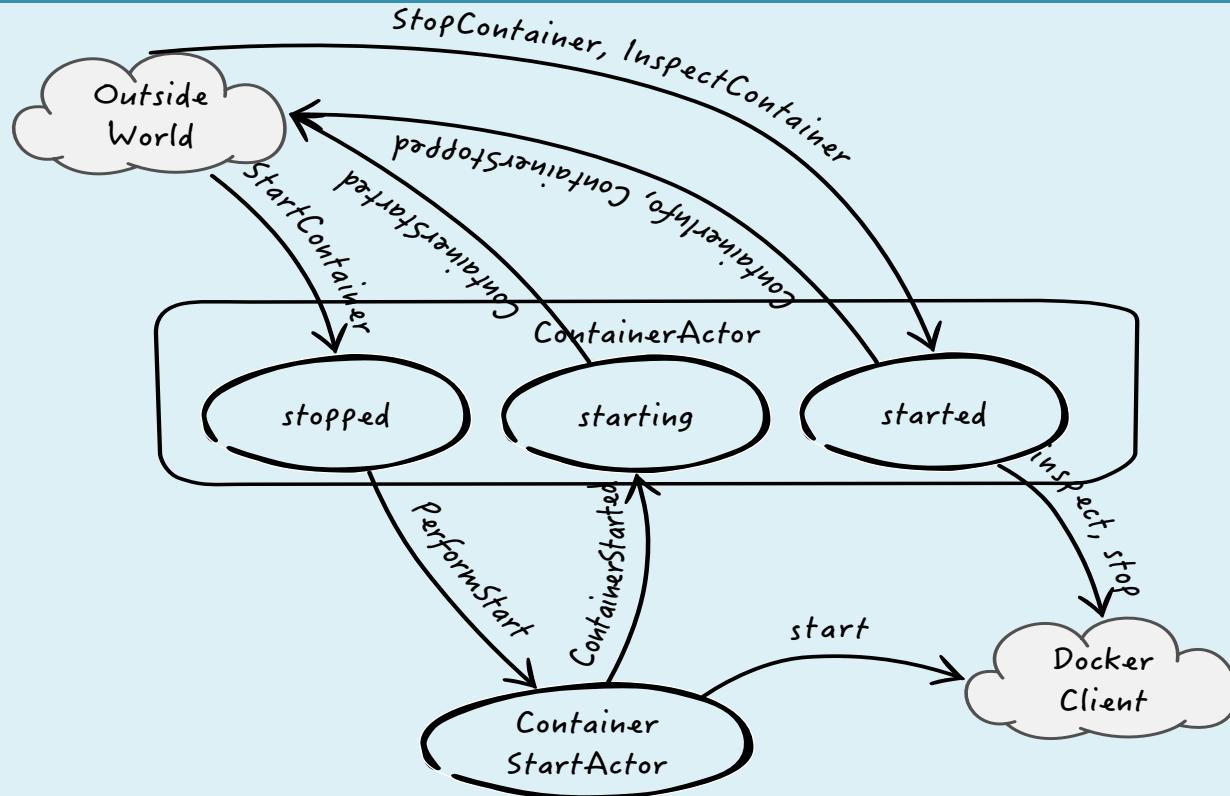


MAIN TEST FRAMEWORK ELEMENTS

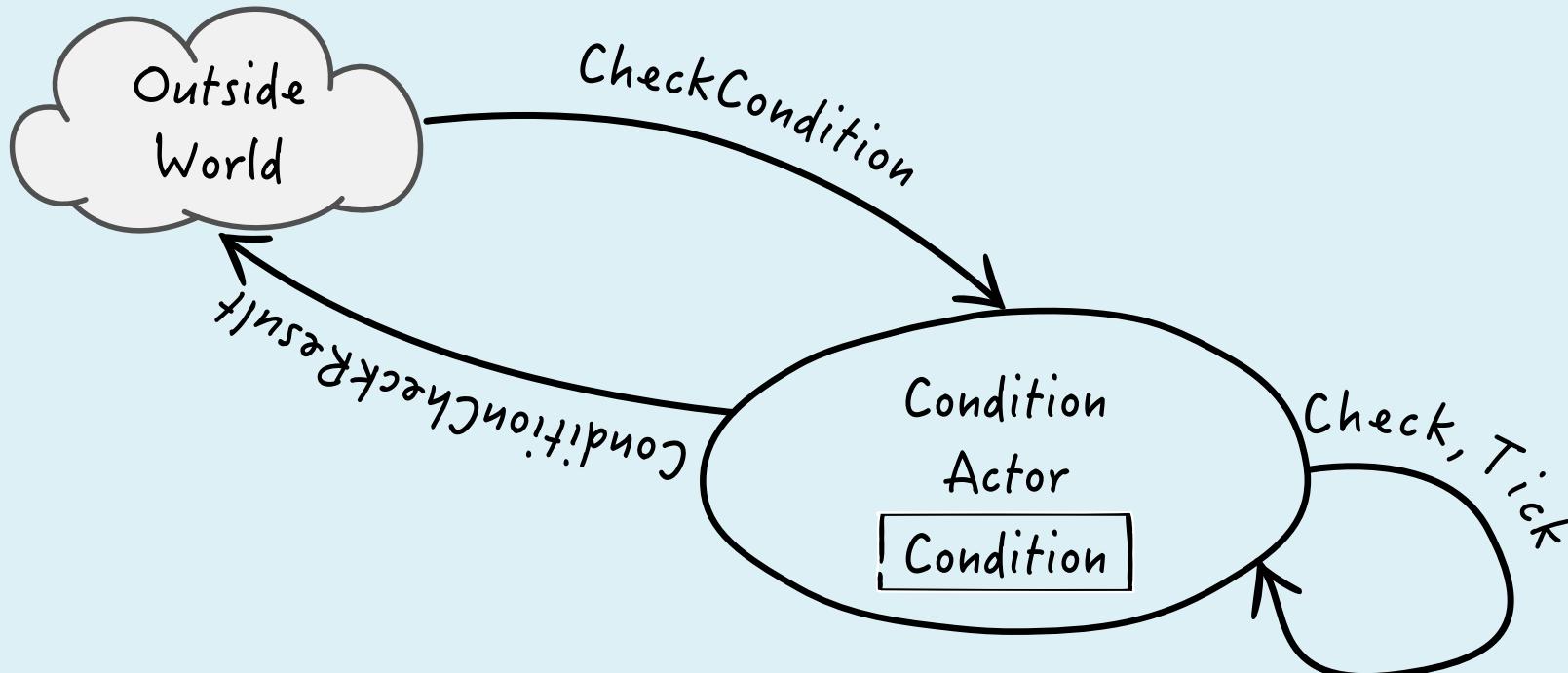
An Akka based Container Mgr



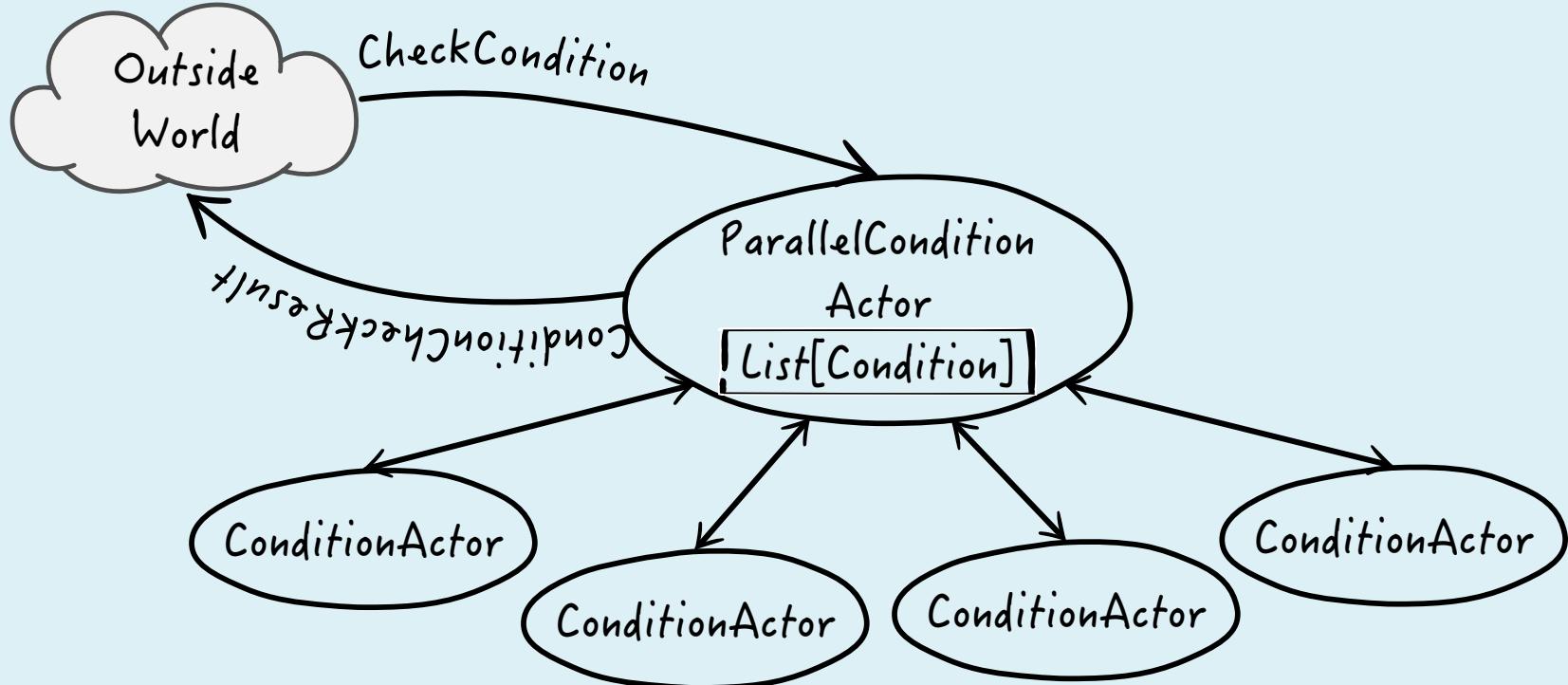
Container Actor



Managing Conditions



Composed Conditions



<https://github.com/woq-blended/blended/blob/master/blended-itestsupport/src/main/scala/de/woq/blended/itestsupport/condition/ParallelConditionActor.scala>

<https://github.com/woq-blended/blended/blob/master/blended-itestsupport/src/main/scala/de/woq/blended/itestsupport/condition/SequentialConditionActor.scala>

More test support

- Akka based JMS producer/consumer for throughput measurements (Early stage)
- Akka based Jolokia client to Query the container's JMX resources via REST

<https://github.com/woq-blended/blended/tree/master/blended-jolokia>

<https://github.com/woq-blended/blended/blob/master/blended-itestsupport/src/main/scala/de/woq/blended/itestsupport/camel/CamelTestSupport.scala>



ONLINE EXAMPLE

Example sources / build

- Container packaging
- Docker definition
- Integration test framework
- Jolokia support
- Example Test
- Jenkins build of Example Test



INTEGRATION TEST GUIDELINES

Integration Test Guidelines

- Concentrate on Blackbox Tests
 - Or Don't modify the CuT to test it ...
- Identify the Communication endpoints
 - Aim for readable Specs and discuss with end users
- Specify Green and Red Path Tests
- Try to capture load scenarios
 - Hard to define due to machine dependencies
- Test fixtures
 - Try to abstract test fixtures into abstract classes or traits



CONCLUSION

More to come...

Working

- Container Management
- Test Framework with easy hooks to CuT
- Test Context for Blackbox testing
- Syntactic sugar for Camel based black-box testing

Wishlist / Issues

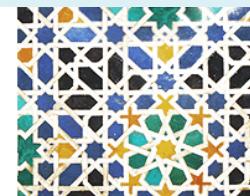
- Kill Containers throughout tests for failover testing
- Leverage a hosting env. like Openshift to execute Containers under test
- Publish Test Framework by itself

Resources

- GitHub Project [Blended](#)
 - [Travis Build](#) - [Waffle Board](#) - [Codacy Code metrics](#) - [Project Page](#)
- Way of Quality [Homepage](#)
- [Scala](#) - [Akka](#) - [Docker](#) - [ScalaTest](#)
- [Apache Camel](#) - for writing test routes
- [Apache Karaf](#) - The containers under test in the samples
- [Apache ActiveMQ](#) - The JMS layer used in the samples
- [Hawtio](#) - The management console of the test containers
- [Jolokia](#) - A JMX to REST bridge

Stay in contact !

- andreas@wayofquality.de
- <http://de.linkedin.com/pub/andreas-gies/0/594/aa5>
- <https://www.facebook.com/andreas.gies.5>
- @andreasgies, #WOQ-Blended on Twitter
- Enjoy a course or a Yoga & coding breakout by the sea



Castillo San Rafael
Andalusian Centre for Art & Cultural Holidays

<http://www.castillosanrafael.com>