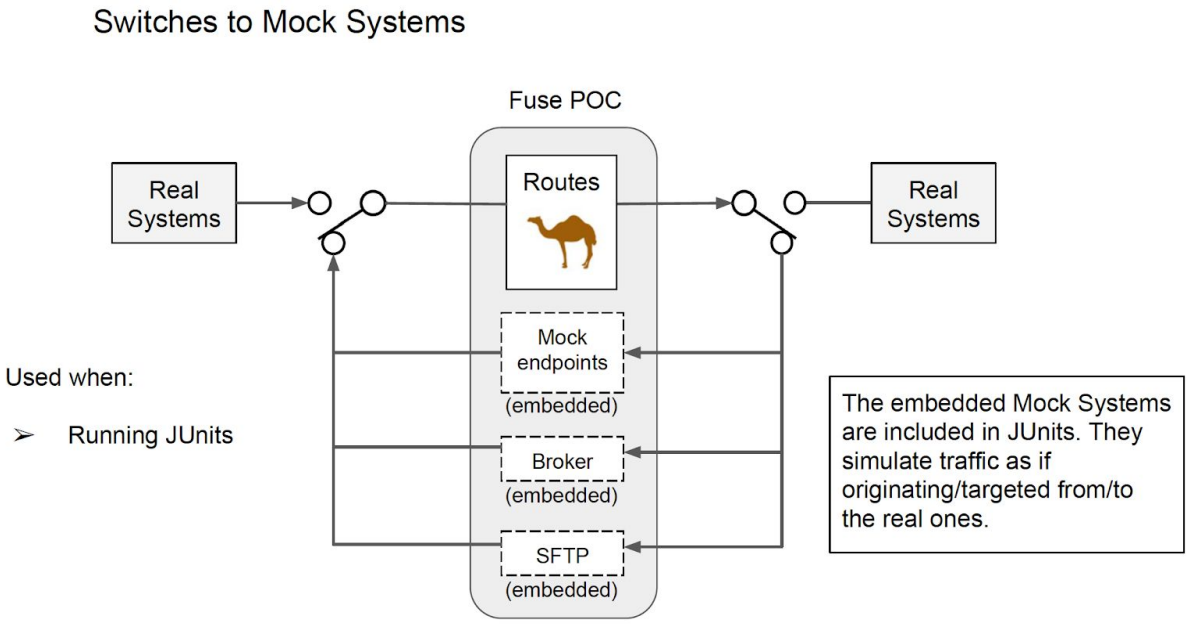
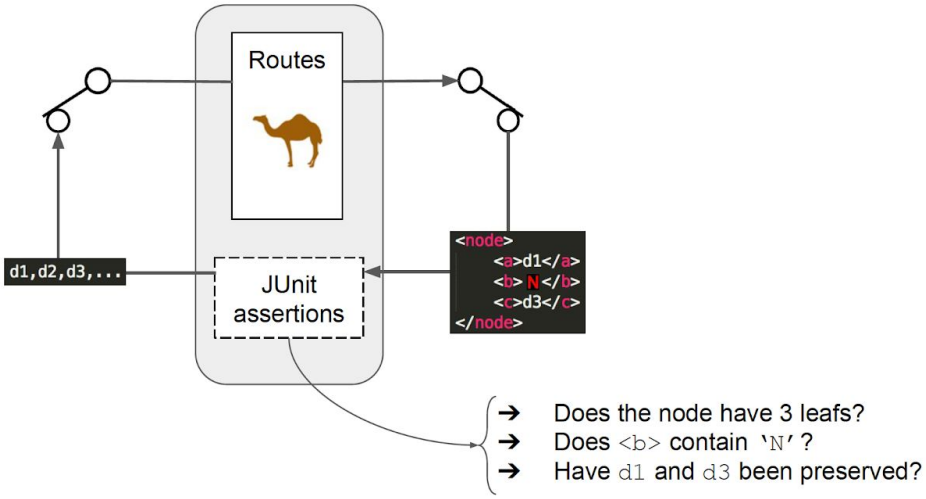
Testing Camel Routes

Disconnect Came/Fuse based service from external systems



Simulate external systems using Camel mocking capabilities

Test Units to validate implementation



Identify happy and unhappy scenarios

Scenario 1 (s1-java)

Test cases (JUnits) included.

- `testHappyScenario()`
- `testBackendException()`
- `testBackendUnavailable()`

Scenario 1 (s1-dozer)

Test cases (JUnits) included.

- `testHappyScenario()`
- `testBackendException()`
- `testBackendUnavailable()`

Scenario 1 (s1-xslt)

Test cases (JUnits) included.

- `testHappyScenario()`
- `testBackendException()`
- `testBackendUnavailable()`

A simplified REST based Camel route

```
@Component
public class CamelRouter extends RouteBuilder {

    @Override
    public void configure() throws Exception {

        restConfiguration()
                .apiContextPath("/api-doc")
                .apiProperty("api.title", "Greeting REST API")
                .apiProperty("api.version", "1.0")
                .apiProperty("cors", "true")
                .apiProperty("base.path", "camel/")
                .apiProperty("api.path", "/")
                .apiProperty("host", "")
                .apiContextRouteId("doc-api")
            .component("servlet")
            .bindingMode(RestBindingMode.json);

        rest( path: "/greetings").description("Greeting to {name}")
            .get("/{name}").outType(Greetings.class)
                .route().routeId("greeting-api")
                .to("direct:greetingsImpl");

        from( uri: "direct:greetingsImpl")
                .description("Greetings REST service implementation route")
                .streamCaching()
                .to("bean:greetingsService?method=getGreetings");
    }

}
```

A simplified MQ based Camel route

```
from("activemq:NewOrders")
  .choice()
    .when().xpath("/order/product = 'widget'")
      .to("activemq:Orders.Widget")
    .otherwise()
      .to("activemq:Orders.Gadget");
```

Test Dependencies

```
<!-- Test -->
<dependency>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-starter-test</artifactId>
  <scope>test</scope>
</dependency>
<dependency>
  <groupId>org.apache.camel</groupId>
  <artifactId>camel-test</artifactId>
  <scope>test</scope>
</dependency>
<dependency>
  <groupId>org.apache.camel</groupId>
  <artifactId>camel-test-spring</artifactId>
  <scope>test</scope>
</dependency>
```

Test with test profiles/properties

A sample Camel route

```
@Component
public class AnimalRoute extends RouteBuilder {

    public static final String CAMEL_FILE_NAME = "CamelFileName";

    @Override
    public void configure() throws Exception {

        from("{{animalSource}}")
                .log("got message")
                .choice()
                .when(p -> p.getIn().getHeader(CAMEL_FILE_NAME).toString().contains("dog"))
                    .log("found a dog!")
                    .to("{{dogEndpoint}}")
                .when(p -> p.getIn().getHeader(CAMEL_FILE_NAME).toString().contains("cat"))
                    .log("looks like a cat!")
                    .to("{{catEndpoint}}");
    }
}
```

A sample test

```
@RunWith(CamelSpringBootRunner.class)
@SpringBootTest
@ActiveProfiles("test")
public class AnimalRouteTest {
    public static final String NICE_DOG = "nice dog", NASTY_CAT="nasty cat", SUPERNASTY_CAT="super nasty cat";

    @EndpointInject(uri = "{{dogEndpoint}}")
    protected MockEndpoint dogEndpoint;

    @EndpointInject(uri = "{{catEndpoint}}")
    protected MockEndpoint catEndpoint;

    @EndpointInject(uri = "{{animalSource}}")
    protected ProducerTemplate animalSource;

    @Test
    @DirtiesContext
    public void testDog() throws Exception {
        dogEndpoint.expectedMessageCount(1);
        catEndpoint.expectedMessageCount(0);

        animalSource.sendBodyAndHeader("test", AnimalRoute.CAMEL_FILE_NAME, NICE_DOG);

        dogEndpoint.assertIsSatisfied();
        catEndpoint.assertIsSatisfied();

        dogEndpoint.message(0).predicate(m -> {
            String header = m.getIn().getHeader(AnimalRoute.CAMEL_FILE_NAME).toString();
            return NICE_DOG.equals(header);
        });
    }
}
```

Test with AdviceWith

Listing 9.19 Replacing route endpoint from AWS SQS to SEDA for easy unit-testing

```
public class ReplaceFromTest extends CamelTestSupport {

  @Override
  public boolean isUseAdviceWith() {
    return true;
  }

  @Test
  public void testReplaceFromWithEndpoints() throws Exception {
    RouteDefinition route = context.getRouteDefinition("quotes");
    route.adviceWith(context, new AdviceWithRouteBuilder() {
      public void configure() throws Exception {
        replaceFromWith("direct:hitme");
        mockEndpoints("seda:*");
      }
    });

    context.start();

    getMockEndpoint("mock:seda:camel")
          .expectedBodiesReceived("Camel rocks");
    getMockEndpoint("mock:seda:other")
          .expectedBodiesReceived("Bad donkey");

    template.sendBody("direct:hitme", "Camel rocks");
    template.sendBody("direct:hitme", "Bad donkey");

    assertMockEndpointsSatisfied();
  }

  @Override
  protected RoutesBuilder createRouteBuilder() throws Exception {
    return new RouteBuilder() {
      public void configure() throws Exception {
        from("aws-sqs:quotes").routeId("quotes")
          .choice()
            .when(simple("${body} contains 'Camel'")).to("seda:camel")
            .otherwise().to("seda:other");
      }
    };
  }
}
```

1. This test uses adviceWith.
2. Gets the route to be advised
3. Replaces the route input endpoint with a direct endpoint
4. Mocks all SEDA endpoints
5. Starts Camel after you've finished advising
6. Sets expectations on the mock endpoints
7. Sends in test messages to the direct endpoint
8. Asserts the test passed by asserting the mocks
9. The route originally consumes from Amazon SQS queue system.

Table 9.9 Additional `adviceWith` methods from AdviceWithRouteBuilder

Method	Description
`mockEndpoints`	Mocks all endpoints in the route.
`mockEndpoints(patterns...)`	Mocks all endpoints in the route that matches the pattern(s). You can use wildcards and regular expressions in the given pattern to match multiple endpoints.
`mockEndpointsAndSkip(patterns...)`	Mocks all endpoints and skips sending to the endpoint in the route that matches the pattern(s). You can use wildcards and regular expressions in the given pattern to match multiple endpoints.
`replaceFromWith(uri)`	Easily replaces the incoming endpoint in the route.
`weaveById(pattern)`	Manipulates the route at the node IDs that matches the pattern.
`weaveByToString(pattern)`	Manipulates the route at the node string representation (output from `toString` method) that matches the pattern.
`weaveByToUri(pattern)`	Manipulates the route at the nodes sending to endpoints matching the pattern.
`weaveByType(pattern)`	Manipulates the route at the node type that matches the pattern.
`weaveAddFirst`	Easily weaves in new nodes at the beginning of the route.
`weaveAddLast`	Easily weaves in new nodes at the end of the route.

Camel route coverage

Dependency

```
<plugin>
  <groupId>org.apache.camel</groupId>
  <artifactId>camel-maven-plugin</artifactId>
  <version>2.23.1</version>
</plugin>
```

Result

```
[INFO] Tests run: 1, Failures: 0, Errors: 0, Skipped: 0
[INFO]
[INFO]
[INFO] --- camel-maven-plugin:2.23.1:route-coverage (default-cli)
[INFO] Discovered 1 routes
[INFO] Route coverage summary:

Class:  com.redhat.fuse.boosters.rest.http.CamelRouter
Route:  greetingsRoute

  Line #      Count   Route
  ------      -----   -----
      37          1   from
      40          1     filter
      41          1       transform
      44          1     choice
      46          0       throwException
      50          1     to

Coverage: 5 out of 6 (83.3%)
```

Resources

Must read

- Camel in Action 2nd Edition, Chapter 9 - Testing
- https://docs.spring.io/spring-boot/docs/current/reference/html/boot-features-testing.html

Camel examples repo
https://github.com/bibryam/fuse-samples-and-usecases

Camel route coverage plugin
https://github.com/apache/camel/blob/master/tooling/maven/camel-maven-plugin/src/main/docs/camel-maven-plugin.adoc

Creating tests with JBoss Developer Studio
https://access.redhat.com/documentation/en-us/red_hat_fuse/7.0/html-single/tooling_tutorials/#RiderTutorialJUnit

https://access.redhat.com/documentation/en-us/red_hat_fuse/7.0/html-single/tooling_user_guide/#newCamelTestCase

https://github.com/apache/camel/blob/master/tooling/maven/camel-maven-plugin/src/main/docs/camel-maven-plugin.adoc

Other links
http://camel.apache.org/advicewith.html

http://camel.apache.org/notifybuilder.html

http://camel.apache.org/dataset.html

http://camel.apache.org/advicewith.html