Apache Camel - 1.0

Summary: This document provides basic information about the extension providing **Apache Camel support for Java**.

Extension ID

com.castsoftware.camel

What's new?

Please see **Apache Camel - 1.0 - Release Notes** for more information.

In what situation should you install this extension?

This extension should be installed when your Java application consists of Apache Camel Routes. Objects created for the components encountered in Apache Camel Route are linked to objects produced by the **JEE Analyzer**. This will result in better transactions and calculation of Automated Function Points.

Function Point, Quality and Sizing support

This extension provides the following support:

- **Function Points (transactions)**: a green tick indicates that Function Point counting and Transaction Risk Index are supported
- **Quality and Sizing**: a green tick indicates that CAST can measure size and that a minimum set of Quality Rules exist

Function Points (transactions)	✓

Quality and Sizing

CAST AIP compatibility

This extension is compatible with:

CAST AIP release	Supported
8.3.x	✓
8.2.x	✓

Supported DBMS servers

This extension is compatible with the following DBMS servers:

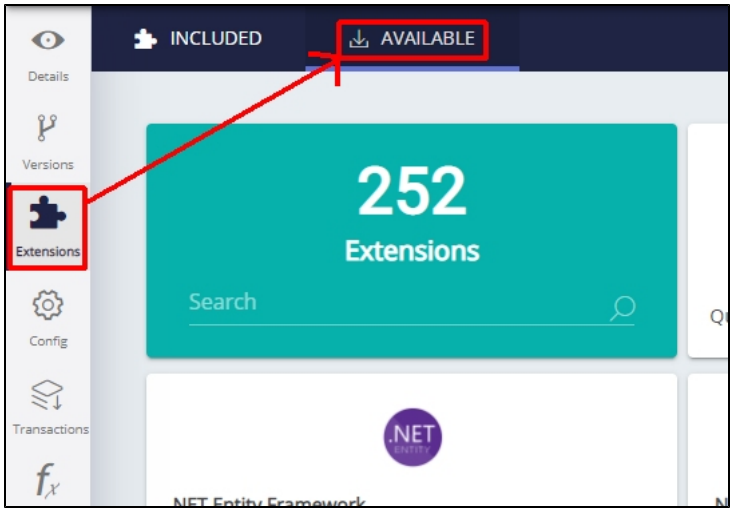
DBMS	Supported
CSS/PostgreSQL	✓
Oracle	✓
Microsoft SQL Server	✗

Prerequisites

✓	Installation of any compatible release of CAST AIP (see table above)

Download and installation instructions

Include the extension using the interface in AIP Console:



There is nothing further to do. Any Apache Camel configuration defined in XML files in the application source code will be automatically detected. Follow the instructions below to run a new analysis/snapshot to generate new results:

- **Advanced onboarding - run and validate the initial analysis**
- **Advanced onboarding - snapshot generation and validation**

What results can you expect?

Once the analysis/snapshot generation is completed, you can view the results in the normal manner (for example via CAST Enlighten):

Objects

Icon	Description	Icon	Description	Icon	Description
	Apache Camel HTTP Get Operation		Apache Camel JMS Queue Receive		Apache Camel Unknown ActiveMQ Queue Receive
	Apache Camel HTTP Put Operation		Apache Camel JMS QueueCall		Apache Camel Unknown ActiveMQ Queue Call
	Apache Camel HTTP Post Operation		Apache Camel Unknown JMS Queue Receive		Apache Camel RabbitMQ Topic Receive
	Apache Camel HTTP Delete Operation		Apache Camel Unknown JMS Queue Call		Apache Camel RabbitMQ Topic Call
	Apache Camel HTTP Any Operation		Apache Camel IBM Queue Receive		Apache Camel Unknown RabbitMQ Topic Receive
	Apache Camel Route Call		Apache Camel IBM Queue Call		Apache Camel Unknown RabbitMQ Topic Call
	Apache Camel Route		Apache Camel Unknown IBM_Queue Receive		Apache Camel File
	Apache Camel Bean Call		Apache Camel Unknown IBM Queue Call		Apache Camel File Call
	Apache Camel Process Call		Apache Camel ActiveMQ Queue Receive		Apache Camel Kafka Topic Receive
	Apache Camel Database Query		Apache Camel ActiveMQ Queue Call		Apache Camel Kafka Topic Call
	Apache Camel Unknown Kafka Topic Receive		Apache Camel Unknown Kafka Topic Call		

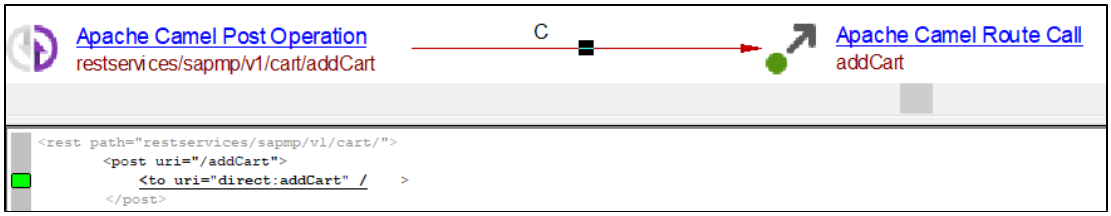
Links

Only **callLinks** are created between various objects created by this extension.

Code examples

XML DSL- callLink between REST POST service and Apache Camel Route Call

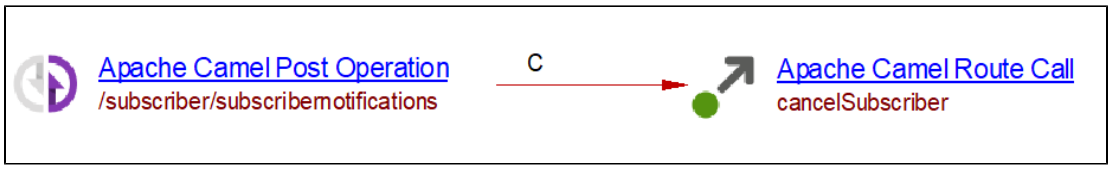
```
<rest path="restservices/sapmp/v1/cart/">
            <post uri="/addCart">
                <to uri="direct:addCart" />
             </post>
```



Java DSL- callLink between REST POST service and Apache Camel Route Call

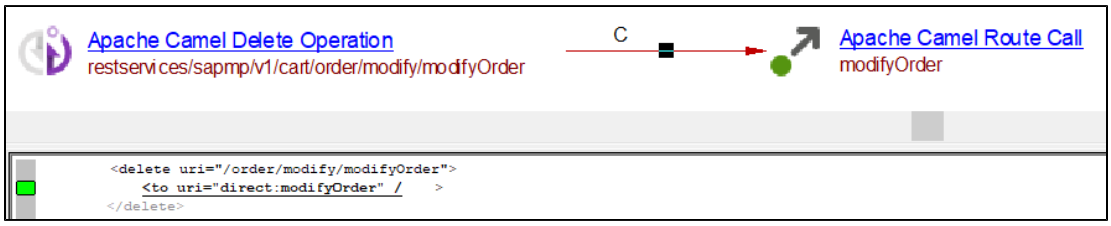
```
public void subscriberServiceRoute() {

          ajscRoute.setRoute(from("restlet:/subscriber/subscribernotifications?
restletMethods=POST&restletBinding=#customBinding")
        .log("************ Determine the event type ************")
            .choice()
        .when().simple("${body} =~ '" + SubscriberConstants.CANCEL_SUBSCRIBER + "'").to("direct:
cancelSubscriber")
}
```



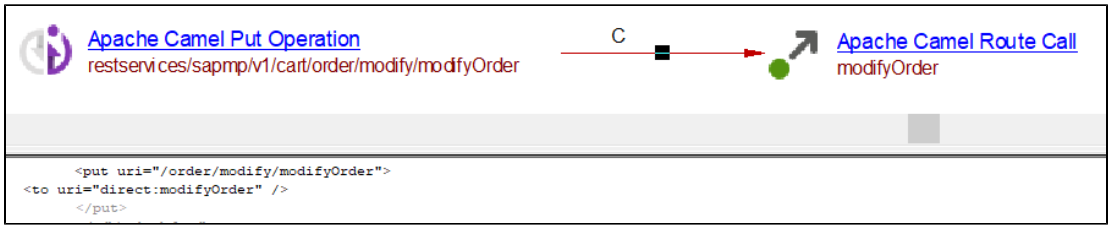
XML DSL - callLink between REST DELETE and Apache Camel Route Call

```
<delete uri="/order/modify/modifyOrder">
                <to uri="direct:modifyOrder" />
</delete>
```



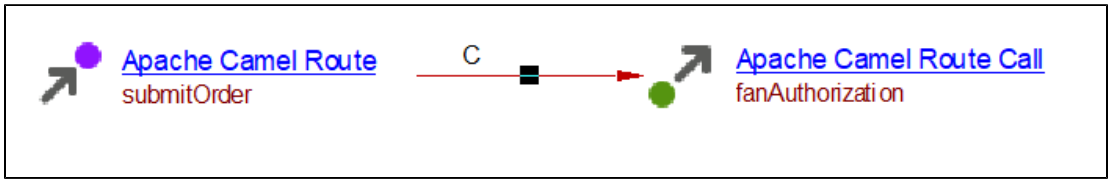
XML DSL - callLink between REST PUT and Apache Camel Route Call

```
<put uri="/order/modify/modifyOrder">
                <to uri="direct:modifyOrder" />
</put>
```



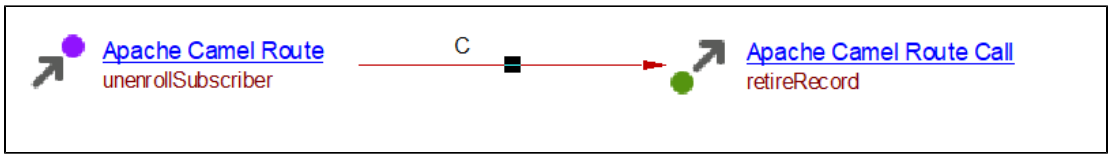
XML DSL - callLink between ApacheCamel Route and Apache Camel Route Call

```
<from uri="direct:submitOrder" />
       <to uri="direct:fanAuthorization" />
```



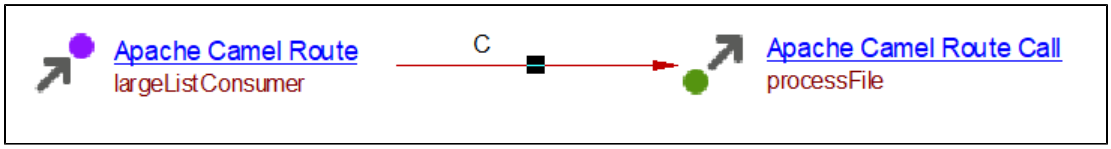
Java DSL - callLink between Apache Camel Route and Apache Camel Route Call

```
public void unenroll()
{
          from("direct:unenrollSubscriber")
          .process("prepareInputToUnEnroll")
          .to("invokeUnenrollService")
        .log("Unenroll Service Status Code in MobileSplit :: ${in.headers.CamelHttpResponseCode}")
        .choice()
        .when().simple("${in.headers.CamelHttpResponseCode} =~ "+ "'"+SubscriberConstants.
RESPONSE_SUCCESS_CODE+"'"
          + " || ${in.headers.CamelHttpResponseCode} =~ '"+SubscriberConstants.ACCEPTED_CODE+"'")
           .to("direct:retireRecord")
           .process(SubscriberConstants.RESPONSE_HANDLER)
        .otherwise()
             .setBody().constant(SubscriberConstants.MESSAGE_UNENROLLMENT_UNSUCCESSFUL)
             .process(SubscriberConstants.RESPONSE_ERROR_HANDLER)
        .endChoice().end();
}
```



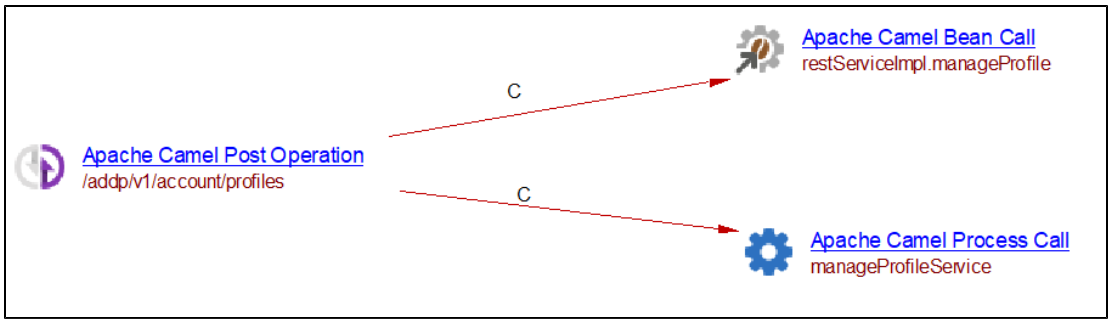
Java DSL - callLink between Apache Camel Route (seda) and Apache Camel Route Call (direct)

```
public void configure() throws Exception {
        from("seda:largeListConsumer?concurrentConsumers=1&size=1000&timeout=0&blockWhenFull=true")
                .routeId("largeListConsumer")
                .to("direct:processFile")
                .end();
}
```



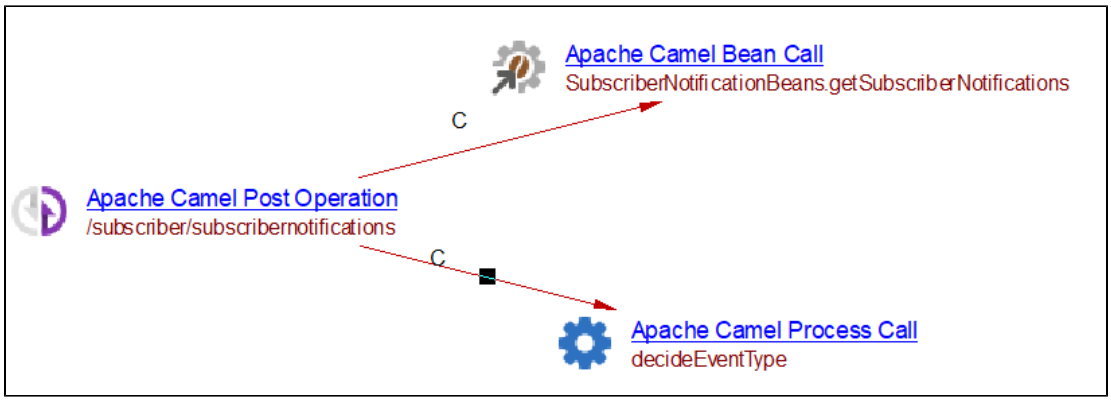
XML DSL - callLinks between Rest POST service and Bean Call, Process Call

```
<from id="_from1" uri="restlet:/addp/v1/account/profiles?restletMethods=POST&restletBinding=#customBinding"/>
        <doTry id="_doTrymp">
            <to id="_to_MAN1" uri="bean:restServiceImpl?method=manageProfile"/>
            <process id="_manageProfileService" ref="manageProfileService"/>
```



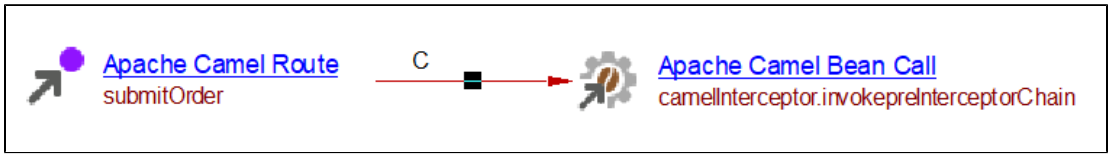
Java DSL - callLinks between Rest POST service and Bean Call, Process Call

```
public void subscriberServiceRoute()
{
          ajscRoute.setRoute(from("restlet:/subscriber/subscribernotifications?
restletMethods=POST&restletBinding=#customBinding")
            .bean("SubscriberNotificationBeans","getSubscriberNotifications(exchange)")
           .process("decideEventType"));
}
```



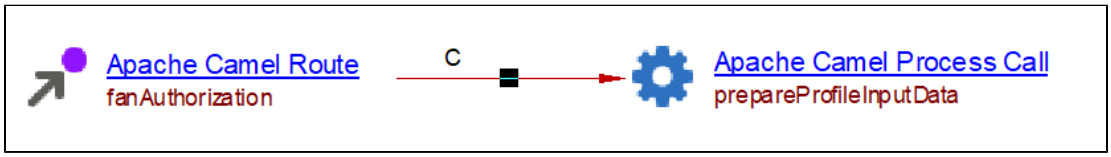
XML DSL - callLink between Apache Camel Route and Bean Call

```
<from uri="direct:submitOrder" />
  <interceptFrom id="_interceptFrom2">
           <to id="submitOrderCamelInterceptor" uri="bean:camelInterceptor?method=invokepreInterceptorChain"
/>
  </interceptFrom>
```



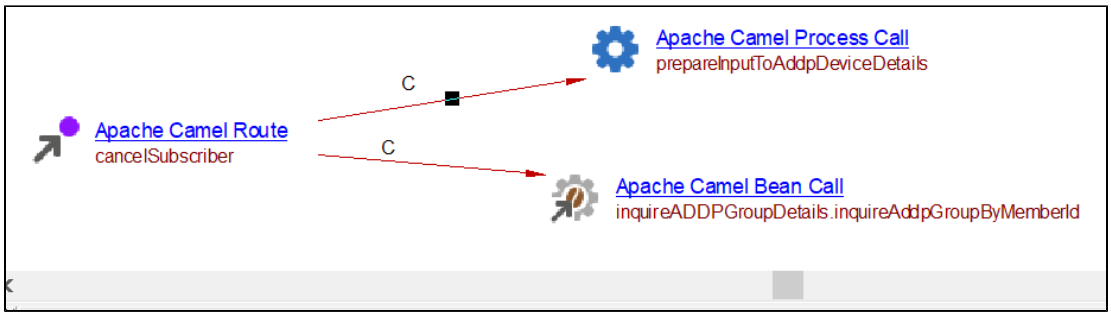
XML DSL - callLink between Apache Camel Route and Process Call

```
<from id="_from_D4" uri="direct:fanAuthorization" />
  <process ref="prepareProfileInputData" />
```



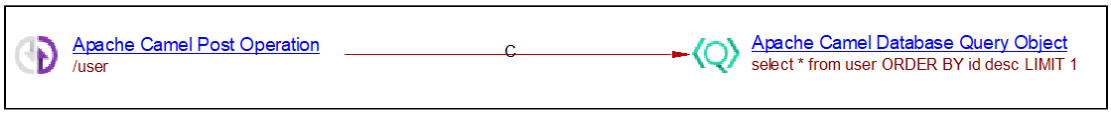
Java DSL - callLinks between Apache Camel Route and Bean Call, Process Call

```
public void cancelSubscriber() {
    from("direct:cancelSubscriber").process("prepareInputToAddpDeviceDetails")
                                   .bean("inquireEnterpriseDeviceDeploymentDetails",
"inquireDeviceDetailsByCtn(${body})");
}
```

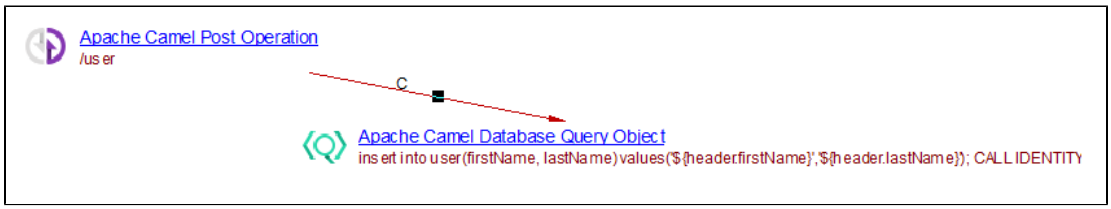


XML DSL - callLinks between REST Post Service and JDBC Query

```
<from uri="restlet:/user?restletMethod=POST"/>
          <setBody>
                <simple>select * from user ORDER BY id desc LIMIT 1</simple>
            </setBody>
            <to uri="jdbc:dataSource"/>
```

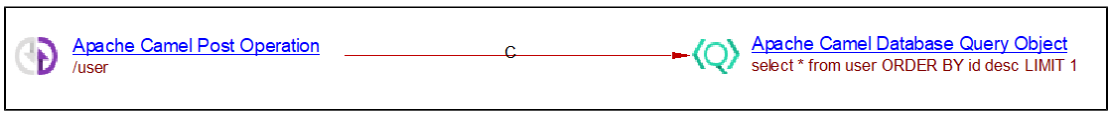



```
<route id="createUser">
            <from uri="restlet:/user?restletMethod=POST"/>
            <setBody>
                <simple>insert into user(firstName, lastName) values('${header.firstName}','${header.
lastName}');
                    CALL IDENTITY();
                </simple>
            </setBody>
            <to uri="jdbc:dataSource"/>
```

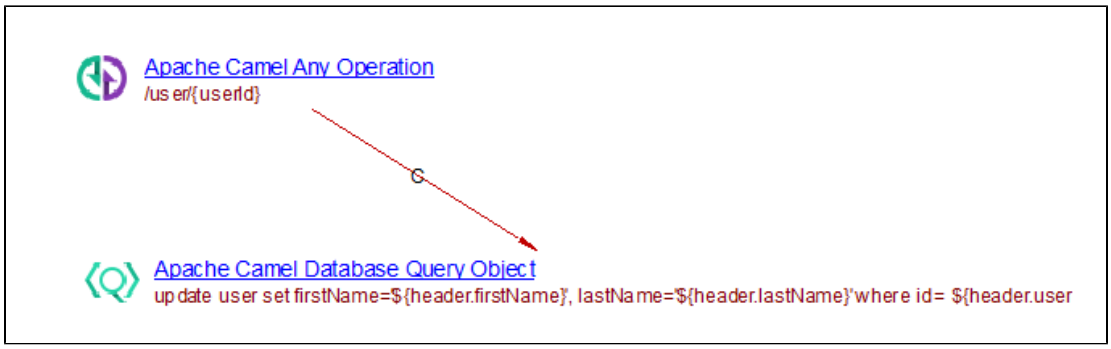


Java DSL - callLink between REST Post Service and JDBC Query

```
public void configure() {
    from("restlet:/user?restletMethod=POST")
        .setBody(simple("select * from user ORDER BY id desc LIMIT 1"))
        .to("jdbc:dataSource");
}
```

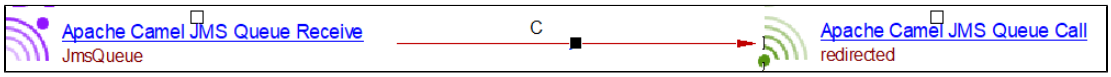



```
public void configure() {
   from("restlet:/user/{userId}?restletMethods=GET,PUT,DELETE")
         .setBody(simple("update user set firstName='${header.firstName}', lastName='${header.lastName}'
where id = ${header.userId}"))
         .to("jdbc:dataSource");
}
```



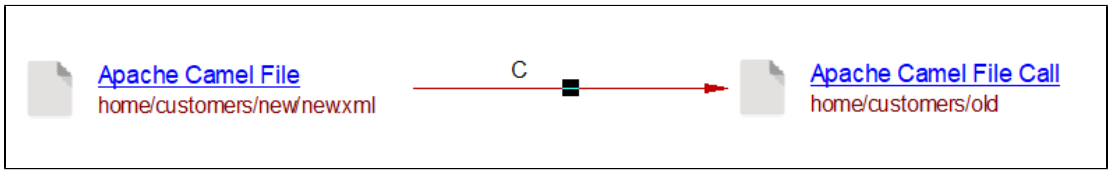
Java DSL - callLink between JMS Queue Receive and JMS Queue Call

```
public void configure() throws Exception
{
   from("jms:JmsQueue").bean(ProcessingBean.class, "doSomething").to("jms:redirected");
}
```



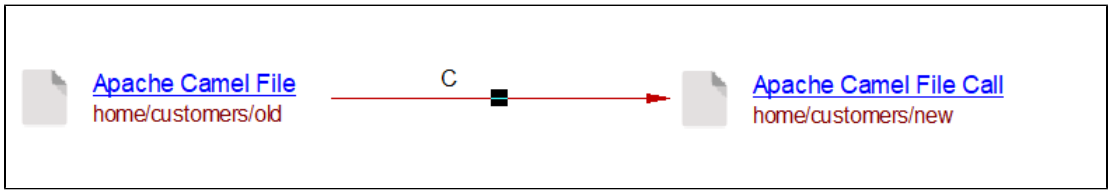
XML DSL - callLink between File and File Call

```
<route id="fileRoute2">
        <from uri="file:home/customers/new?fileName=new.xml"/>
    <to uri="file://home/customers/old"/>
</route>
```



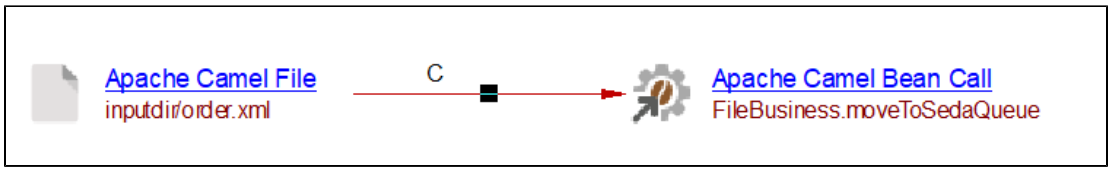
Java DSL - callLink between File and File Call

```
public void configure() throws Exception {
        from("file:home/customers/old")
                .routeId("transfer")
                                .log(LoggingLevel.INFO, "com.toyota.tme.cws.transform.route.error",
"Transferring File - ${file:name}")
                .to("file:home/customers/new")
                .end();
}
```



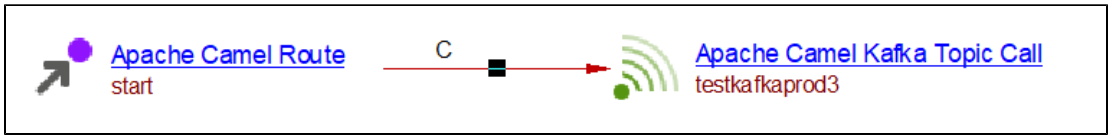
Java DSL - callLink between File and Bean Call

```
public void configure () throws Exception {
        from("file://inputdir?fileName=order.xml")
                .routeId("readFilefromDir")
                                .log(LoggingLevel.INFO, "com.toyota.tme.cws.transform.route.error",
"Processing File - ${file:name}")
                .bean("FileBusiness", "moveToSedaQueue")
                .end();
}
```



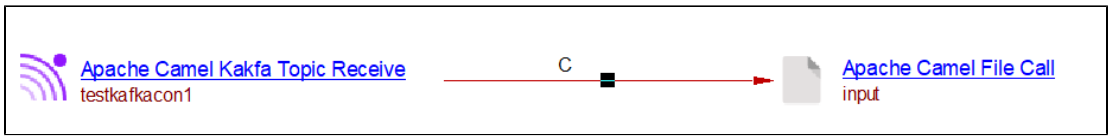
Java DSL - callLink between Apache Camel Route and Kafka Topic Call

```
public void configure () throws Exception {
        from("direct:start")
            .to("kafka:localhost:8080?topic=testkafkaprod3");
}
```

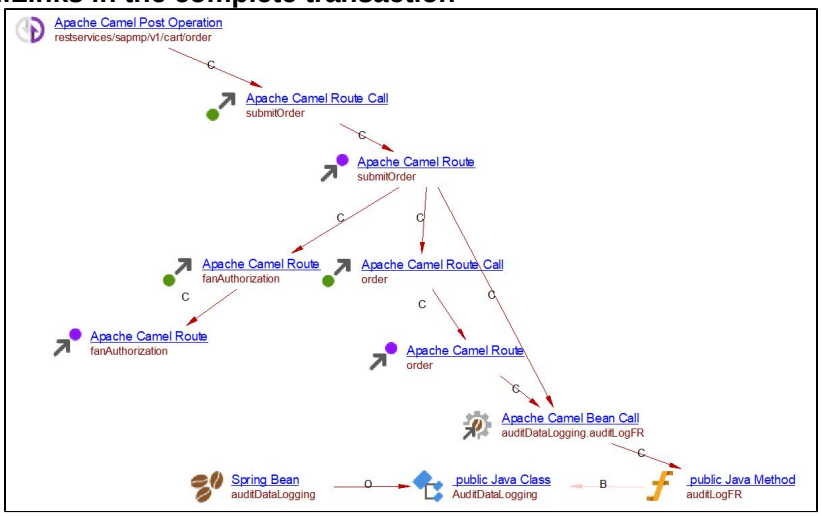


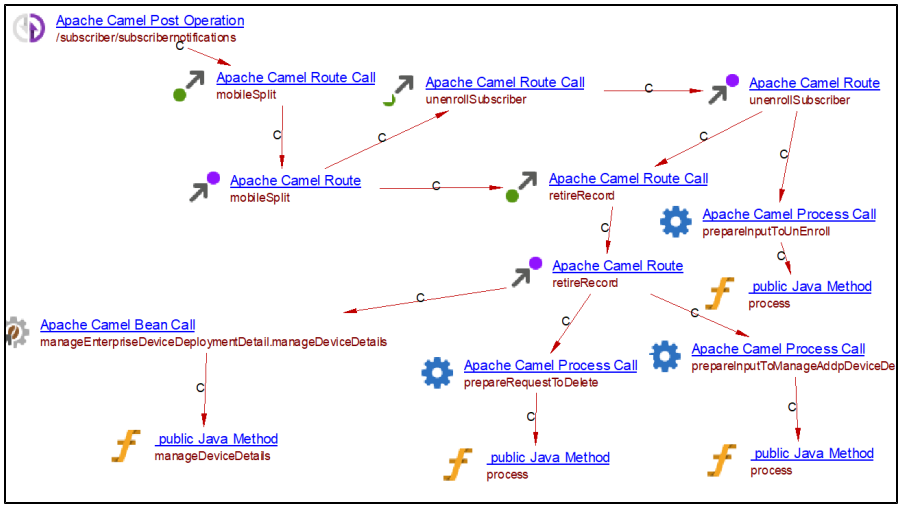
Java DSL - callLink between Kafka Topic Receive and File Call

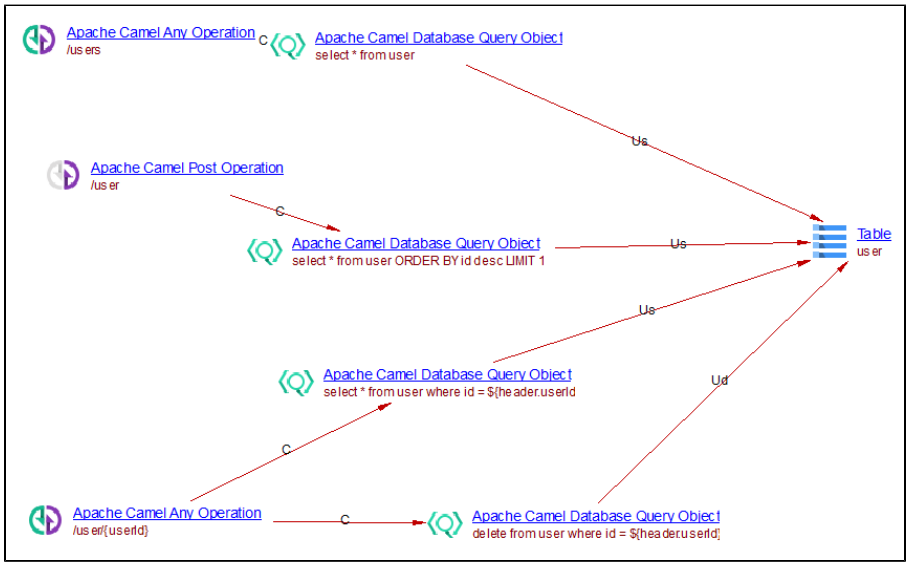
```
public void configure () throws Exception {
        from("kafka:localhost:8080?
topic=testkafkacon1&zookeeperHost=localhost&zookeeperPort=2181&groupId=group1").to("file:input");
}
```

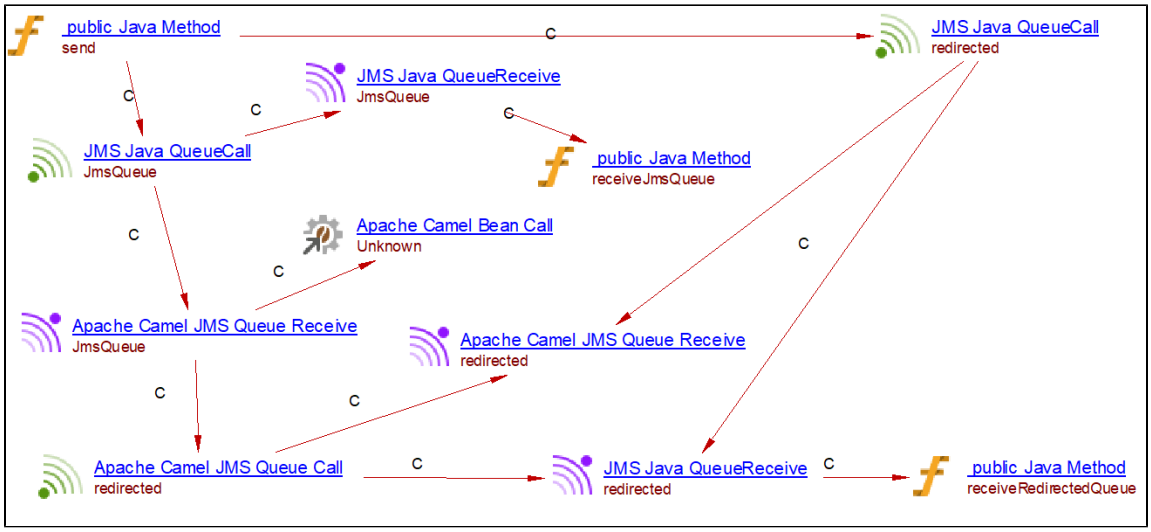


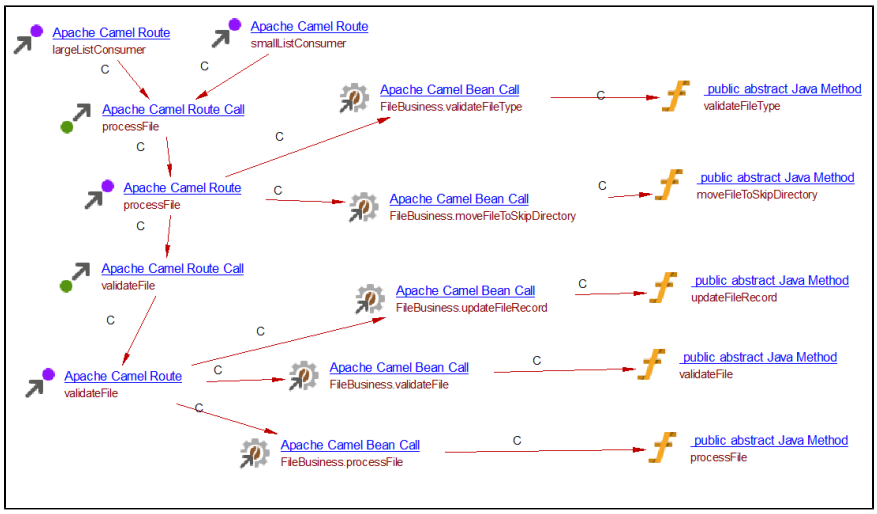
callLinks in the complete transaction

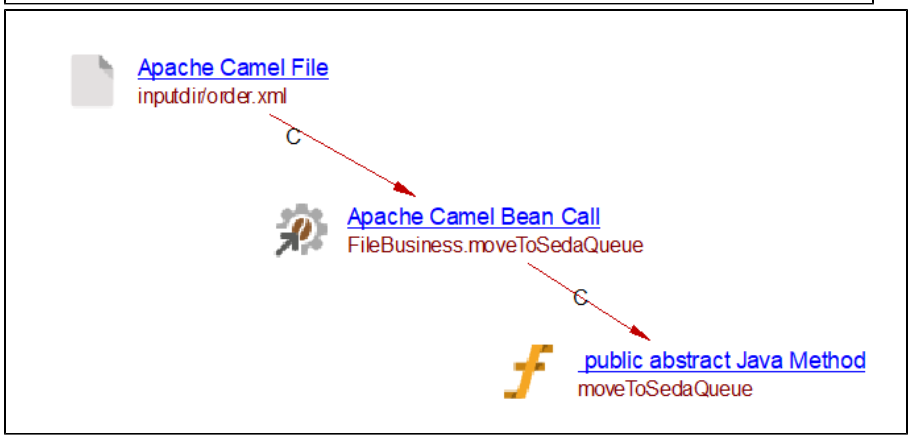


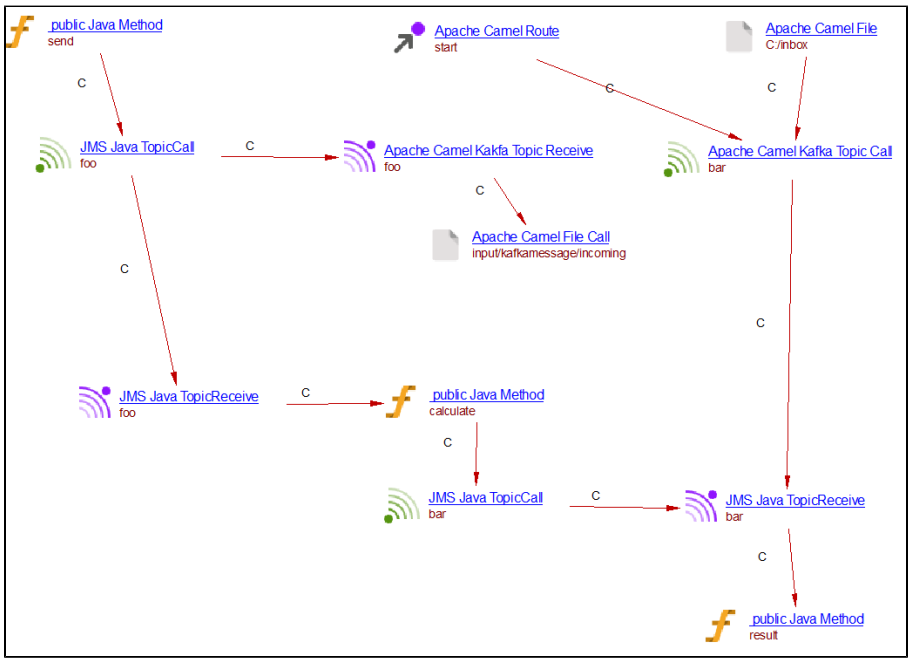

Apache Camel Post Operation
/subscriber/subscribernotifications
C
Apache Camel Route Call
mobileSplit
Apache Camel Route Call
unenrollSubscriber
C
Apache Camel Route
unenrollSubscriber
C
C
C
Apache Camel Route
mobileSplit
C
Apache Camel Route Call
retireRecord
C
Apache Camel Process Call
prepareInputToUnEnroll
C
Apache Camel Route
retireRecord
public Java Method
process
C
C
C
Apache Camel Bean Call
manageEnterpriseDeviceDeploymentDetail.manageDeviceDetails
Apache Camel Process Call
prepareRequestToDelete
Apache Camel Process Call
prepareInputToManageAddpDeviceDe
C
C
C
public Java Method
manageDeviceDetails
public Java Method
process
public Java Method
process


Apache Camel Any Operation
/users
C
Apache Camel Database Query Object
select * from user
Us
Apache Camel Post Operation
/user
C
Apache Camel Database Query Object
select * from user ORDER BY id desc LIMIT 1
Us
Table
user
Us
Apache Camel Database Query Object
select * from user where id = ${header.userId
Ud
C
Apache Camel Any Operation
/user/{userId}
C
Apache Camel Database Query Object
delete from user where id = ${header.userId}


public Java Method
send
C
JMS Java QueueCall
redirected
JMS Java QueueReceive
JmsQueue
C
C
C
JMS Java QueueCall
JmsQueue
public Java Method
receiveJmsQueue
C
Apache Camel Bean Call
Unknown
C
C
Apache Camel JMS Queue Receive
JmsQueue
Apache Camel JMS Queue Receive
redirected
C
C
Apache Camel JMS Queue Call
redirected
C
JMS Java QueueReceive
redirected
C
public Java Method
receiveRedirectedQueue







Limitations

- Unknown queue/process/bean/route/file/jms/kafka object is created in case where the exact name can't be retrieved
- Routes defined using route templates result in unknown route objects
- All other route components are ignored. Any Route starting with a component not mentioned in the Object section , the whole of the Route with its components will be ignored.

- Any component created with a customized name will not be handled or created
- There may be missing/multiple links instead of one between process call and process method of the target class
- There may be multiple links instead of one between bean call and target method of the bean
- There may be incorrect bookmarks in few call links and objects
- One of the signature for the overloaded method idempotentConsumer method of org.apache.camel.model.ProcessDefinition is not supported. This suspends the creation of the objects for the components in that route following this method.
- Current version does not support Kafka topic using Kafka idempotent repository
- Consuming messages from multiple Kafka topics is not supported
- Kafka topic receive/call object is always created by the name whatever is present in Kafka component topic. There is a possibility that name mentioned of the topic is an alias and refers to something else. Due to some design constraints, it will not be evaluated to refer to the exact value. Hence, object will be created in the name present in Kafka component.

Future development

- More Components Support, Better Linking, Fixing Bookmark Issue, Custom Component Support.