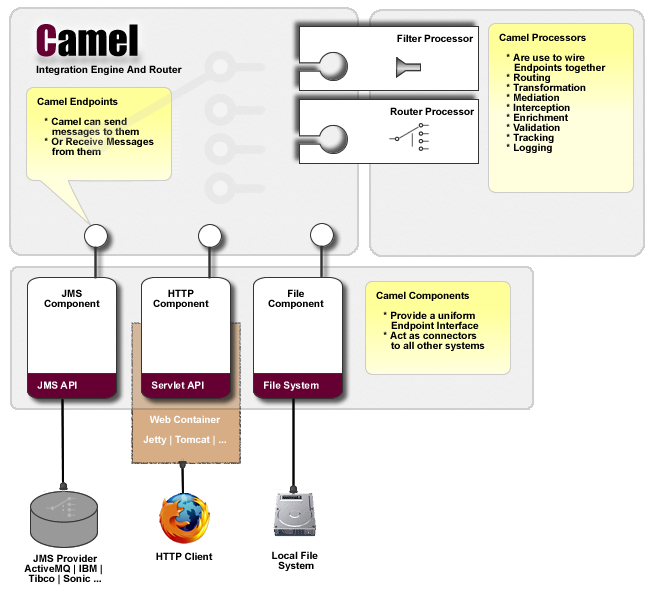
**Apache Camel**

****

Apache camel primarily move/copy things from one end points that is messages from one end point to another for example having a JMS end point or receiving it and we have a file system to other end then apache camel helps to send from JMS to file system end point or vice versa or receiving some servlet APIS and we need to transfer into a JMS message then It helps to transform. It can do filtering or routing or logging or tracking or transforming. It types the message in one format and move it to another. It can send encrypt logged. We can do transformation as workflow of that is not necessary .exchange message pattern-xml,json etc.. Provides many converters

Components: <http://camel.apache.org/components.html> and routes can return in 2 ways-java dsl, spring xml.

Construct route-it is like a instruction to camel on how to move messages

Camel uses a Java based [Routing Domain Specific Language (DSL)](http://camel.apache.org/dsl.html) or an [Xml Configuration](http://camel.apache.org/xml-configuration.html) to configure [routing and mediation rules](http://camel.apache.org/routes.html) which are added to a [Camel Context](http://camel.apache.org/maven/current/camel-core/apidocs/org/apache/camel/CamelContext.html) to implement the various [Enterprise Integration Patterns](http://camel.apache.org/enterprise-integration-patterns.html).

At a high level Camel consists of a [Camel Context](http://camel.apache.org/maven/current/camel-core/apidocs/org/apache/camel/CamelContext.html) which contains a collection of [Component](http://camel.apache.org/component.html) instances. A [Component](http://camel.apache.org/component.html) is essentially a factory of [Endpoint](http://camel.apache.org/endpoint.html) instances. You can explicitly configure [Component](http://camel.apache.org/component.html) instances in Java code or an IoC container like Spring or Guice, or they can be auto-discovered using [URIs](http://camel.apache.org/uris.html).

An [Endpoint](http://camel.apache.org/endpoint.html) acts rather like a URI or URL in a web application or a Destination in a JMS system; you can communicate with an endpoint; either sending messages to it or consuming messages from it. You can then create a [Producer](http://camel.apache.org/maven/current/camel-core/apidocs/org/apache/camel/Producer.html) or [Consumer](http://camel.apache.org/maven/current/camel-core/apidocs/org/apache/camel/Consumer.html) on an [Endpoint](http://camel.apache.org/endpoint.html) to exchange messages with it.

In an enterprise, a number of systems of different types exist. Some of these may be legacy systems while some may be new. These systems often interact with each other, and need to be integrated. This interaction or integration is not easy as the implementations of the systems, their message formats may differ. One way to achieve this is to implement code which bridges these differences. However this will be point to point integration. If tomorrow again if there is change in a system the other might also have to be changed which is not good. Instead of this point to point integration which causes tight coupling we can implement an additional layer to mediate the differences between the systems. This results in loose coupling and not affect much our existing systems. Apache Camel is a rule-based routing and mediation engine that provides a Java object-based implementation of the Enterprise Integration Patterns using an API (or declarative Java Domain Specific Language) to configure routing and mediation rules.   
 **Features of Apache Camel**

* Apache camel is a light weight framework. It can be deployed on a variety of containers like Tomcat, esbs etc.
* Apache camel provides us with a number of components. These components make interacting create endpoints with which a system can interact with other external systems. For example using an ActiveMQ component we expose an ActiveMQ endpoint for interaction with external system. There are more than 100 components provided by Apache Camel. Some of them are FTP,JMX, Webservices, HTTP. Apache camel also allows users to create custom components.
* Apache camel uses Message Exchange Patterns(MEP). Apache camel exchange can hold any kind of message. It supports a variety of formats like xml, JSON etc.
* Camel provides many different type converters for marshaling and unmarshalling the message during routing. Routes in a variety of domain-specific languages (DSL).The most popular ones are  
  1. Java DSL - A Java based DSL using the fluent builder style.  
  2. Spring XML - A XML based DSL in Spring XML files  
  When using Spring XML we can make use of Spring support for  
  features like Transaction Management, JPA etc.