International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 06 Issue: 07 | July - 2022 Impact Factor: 7.185 ISSN: 2582-3930

Enterprise Integration Pattern through Camel

Name : ShreeKrishna Arvind Singh **Mentor Name: Prof. Manish Dubey**

BVIMIT, Navi Mumbai.

Abstract :

Enterprise application integration is necessary in almost every company due to new products and applications. Integrating these applications creates several problems.

New paradigms come up every decade, for example client / server communication, Service-oriented Architecture (SOA) or Cloud Computing.

An enterprise information portal (EIP) is a framework used to support and integrate processes, people and information across an organization. It gives a unified and secure gateway for information and a knowledge base for employees, partners and customers. The application interface provided by an EIP is often web-based and provides instant deployment, centralized maintenance and ergonomics which are intuitive and user-friendly.

Besides, different interfaces, protocols and technologies emerge. Instead of storing data in files in the past (many years ago), SQL databases are used often today. Sometimes, even NoSQL databases are required in some usecases. Synchronous remote procedure calls or asynchronous messaging is used to communicate via several technologies such as RMI, SOAP Web Services, REST or JMS. A lot of software silos exists. Nevertheless, all applications and products of these decades have to communicate with each other to work together perfectly. Apache Camel is a lightweight integration framework which implements all EIPs. Thus, you can easily integrate different applications using the required patterns. You can use Java, Spring XML, Scala or Groovy. Almost every technology you can imagine is available, for example HTTP, FTP, JMS, EJB, JPA, RMI, JMS, JMX, LDAP, Netty, and many, many more (of course most ESBs also offer support for them). Besides, own custom components can be created very easily.

Keywords : Apache Camel, EIP, Message, APIS, Microservices, REST, integration, application.

1.INTRODUCTION :

Apache Camel is an open source integration framework designed to make integrating systems simple and easy.

It allows end users to integrate various systems using the same API, providing support for multiple protocols and data types, while being extensible and allowing the introduction of custom protocols.

Routes and routing engine are the central part of Camel. Routes contain the logic and flow of integration between different systems.

In order to define routes more easy and clean, Camel offers several different domain-specific languages (DSL) for programming languages like Java or Groovy. On the other hand, it also provides defining routes in XML with Spring DSL.

Using either Java DSL or Spring DSL is mostly user preference, as most of the features are available in both. We Have developed a project through camel that helps us through easier transmission and interaction of data through camel .Our Domain especially uses Camel For interaction of data between clients and also to send data to different teams in our company. In our CMS System We use camel to interact with 3 different client APIS, that is verification, Transaction and reposting.

The Verification APIS tells the consumer or user is a client of the company or not. The verification service send data that is required in transaction service .Like this different services, APIS, technologies interact with each other using Apache Camel.

2.LITERATURE REVIEW :

A design pattern is a general solution to a design and integration problem that mostly occurs in lots of projects. A pattern describes the problem and its proposed solution and discuss any other important factors. EIP focuses on patterns of messaging for enterprise application integration .

Messaging makes it easier for programs to communicate across different programming environments (languages, compilers, and operating systems) because the only thing that each environment needs to understand is the common messaging protocol and format.

First, we'll have a look at the core Camel concepts here:

Message contains data which is being transferred to a route. Each message has a unique identifier and it's constructed out of a body, headers, and attachments

Exchange is the container of a message and it is created when a message is received by a consumer during the routing process. Exchange allows different types of interaction between systems – it can define a one-way message or a request-response message

Endpoint is a channel through which program can send or receive a message. It can refer to a web service URI, queue URI, file, program, etc

Component acts as an endpoint factory. To put it easily, components offer an interface to different technologies using the same approach and syntax. Camel already supports a lot of components in its DSLs for almost every possible technology, but it also gives the ability for writing custom components

Processor is a simple Java method which is used to add custom integration logic to a route. It contains a single process method which can be used to add our own business logic on a message received by a consumer

Apache Camel is a open-source integration framework based on known Enterprise Integration Patterns.

Camel helps you to define routing and mediation rules in a variety of domain-specific languages (DSL, such as Java, XML, Groovy, Kotlin, and YAML).
Apache Camel uses URIs to interact directly with any kind of exchange or messaging model such as HTTP, ActiveMQ, JMS, JBI, SCA, MINA or CXF,

as well as pluggable Components and Data Format options. Apache Camel is a small library with limited dependencies for easy interaction in any Java application. Apache Camel lets you work with the same API regardless which kind of technology is used so study the API once and you can interact with all the Components provided out-of-box.
Apache Camel provides support for Bean Binding and seamless integration with popular frameworks such as Spring, CDI. Camel also has support for unit testing your routes.

3.Methodology:

Camel is a Framework that helps in connecting different systems easily.
When in domains that interact with multiple systems camel can be used extensively and help to reduce the work that is required.
The Routes in camel help to easily interact with other client or User APIS.
The Different types of Microservices can be created with the help of the camel.
Apache Camel is one of the most popular open-source frameworks with intent of solving enterprise integration problems. Camel is based on well-established Enterprise Integration Patterns, or EIPs. This paper mostly focuses on how to use these patterns with the help of camel and with Java application we will see how we can introduce Apache Camel to increase the cohesion of the components in the processor and decrease the on-boarding time of future endpoints. Through patterns such as pipes and filters, seda, direct, message routing, message endpoints, and message translation.
All these features of Apache camel help us to create an application that can easily integrate other apis into itself. Especially for financial domains, their systems and applications which interact with multiples systems throughout the day. Camel can easily help us to interact with these multiple APIS and have an application that can help us to overcome all these issues.

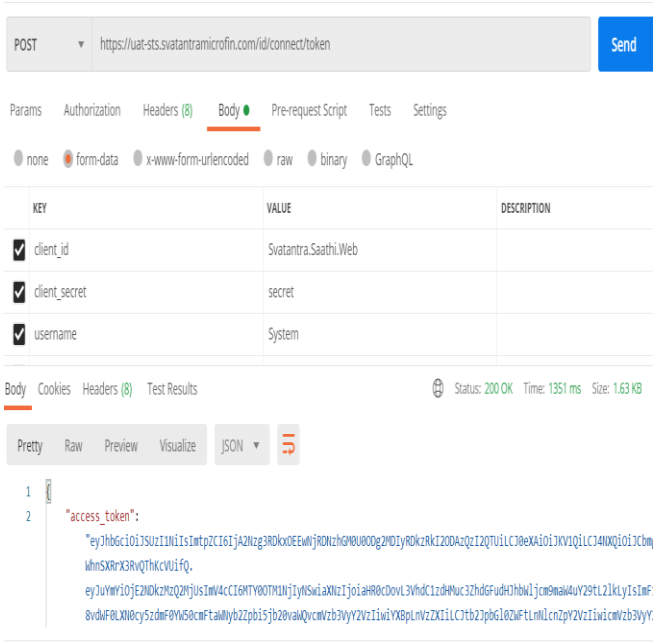
4.SYSTEM ARCHITECTURE :

Cash Management is a comprehensive solution suite built on an advanced architecture. The componentized suite empowers the bank to deliver new propositions in digital cash management with proven digital offering which includes Finacle Management Solution, Virtual Accounts Management, Finacle Payments along with Corporate Online Banking, Corporate Mobile Banking and Digital Engagement Hub will help you accelerate corporate cash management digitization.
Camel makes the integration easier by providing connectivity to a very large variety
of transports and APIs. For example, you can easily route JMS to JSON, JSON to JMS, HTTP to JMS, FTP to JMS, even HTTP to HTTP, and connectivity to Microservices. You simply need to provide appropriate endpoints at both ends. Camel is extensible and thus in future more endpoints can be added easily to the framework.
Camel makes the integration easier by providing connectivity to a very large variety of transports and APIs. For example, you can easily route JMS to JSON, JSON to JMS, HTTP to JMS, FTP to JMS, even HTTP to HTTP, and connectivity to Microservices. You simply need to provide appropriate endpoints at both ends. Camel is extensible and thus in future more endpoints can be added easily to the framework. The project helps financial institutions to increase their reach. In our CMS System We use camel to interact with 3 different client APIS. that is verification, Transaction and reposting.
The Verification APIS tells the consumer or user is a client of the company or not. The verification service send data that is required in transaction service. Like this different services, APIS, technologies interact with each other using Apache Camel. The transaction Service

follows a set of protocol to interact with the client APIS. If for some reason the transaction exception

occurs then reposting service is called which will call our client api 3 times.

Testing APIS through Postman.



Deployment of Service on Server.

5.CONCLUSION :

A concrete example architecture shows how to build a complete usable platform leveraging the widely-adopted open source framework. Apache Camel to build a mission-critical, scalable, highly performant financial platform. Messaging, integration and stream processing are all build on top of the same strong foundation of Camel deployed on premise, in the cloud or in hybrid environments.

Because the number of applications and technologies in each company will increase further, Apache Camel has a great future.

The Apache Camel Framework is a very useful framework that can be used to build application that are very suitable for integration of multiple APIS or technologies. In this paper it was demonstrated how apache camel was used to build a highly scalable application that can work well and interact with other technologies easily. Apache camel helps us to create Microservices that are independently deployable and work without interfering with other applications.

6.Acknowledgement:

I have a great pleasure to express my gratitude to all those who have motivated me during research work. I would also like to thanks Prof. Manish Dubey Sir for his guidance. ShreeKrishna Arvind Singh

7.REFERENCES :

1.APACHE CAMEL DOCUMENTATION : https://camel.apache.org/manual/

2.Wikipedia : https://en.wikipedia.org/wiki/Apache_Camel

3. Camel in Action 2nd edition by Claus Ibsen and Jonathan Anstey. Published by Manning in 2018.

4. Camel in Action by Claus Ibsen and Jonathan Anstey. Published by Manning in December 2010.

5. Apache Camel Developer's Cookbook by Scott Cranton and Jakub Korab. Published by Packt publishing in December 2013.

6. Camel Design Patterns by Bilgin Ibryam. Published by LeanPub in start of 2016.

7. Instant Apache Camel Message Routing by Bilgin Ibryam. Published by Packt publishing in August 2013.

8. Enterprise Integration Patterns by Gregor Hohpe and Bobby Woolf. Published by Addison Wesley in October 2003.