Neusoft

Apache Camel

使用成果分享

谭初跃 2021-06

目录

Apache Camel概述

Introducing Apache Camel

第一章

Apache Camel

- 维基百科
- 逻辑视图

Apache camel

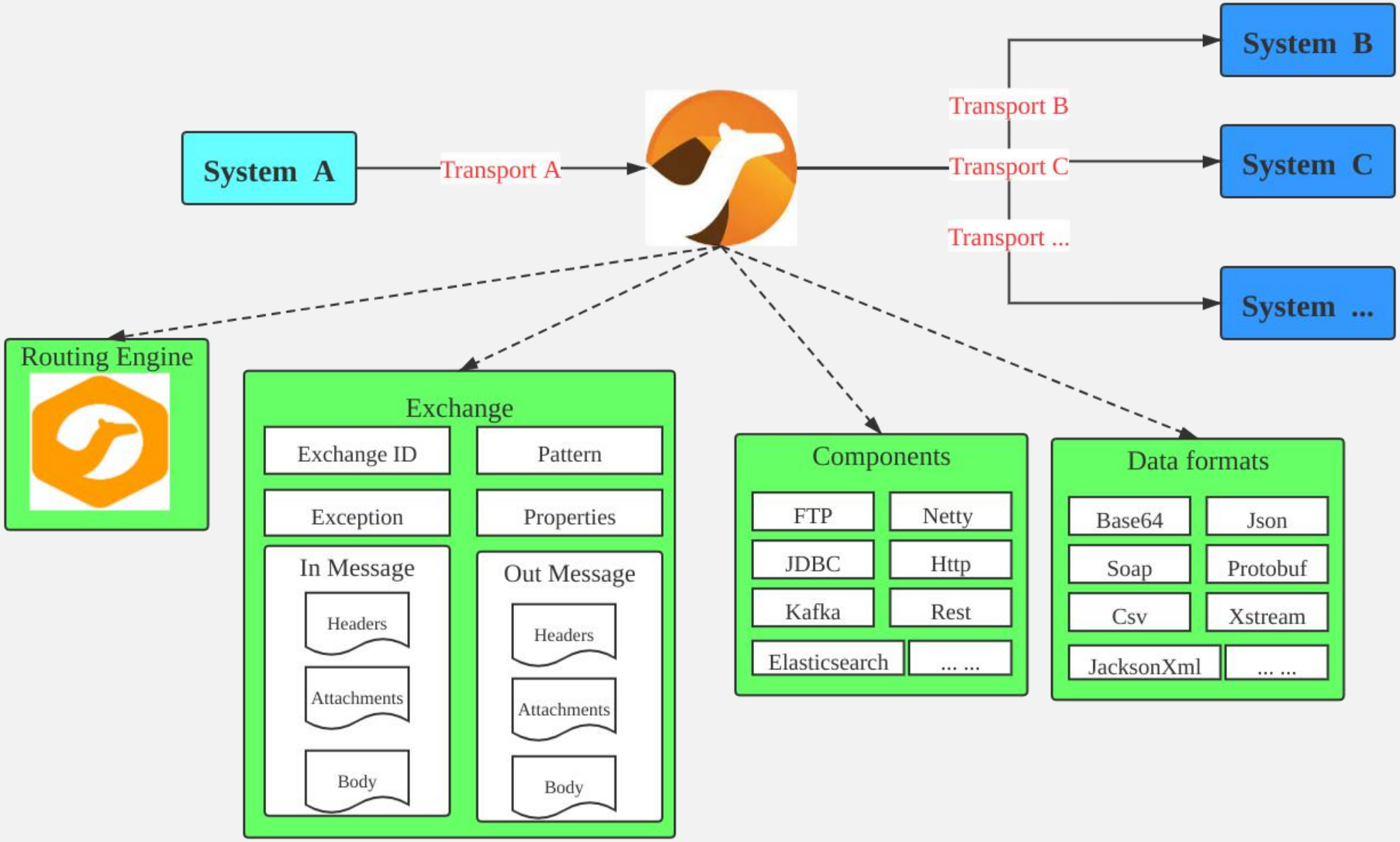
维基百科

Apache Camel是一个基于规则路由和中介引擎，提供企业集成模式的Java对象(POJO)的实现，通过应用程序接口（或称为陈述式的Java领域特定语言（DSL））来配置路由和中介的规则。领域特定语言意味着Apache Camel支持你在的集成开发工具中使用平常的，类型安全的，可自动补全的Java代码来编写路由规则，而不需要大量的XML配置文件。同时，也支持在Spring中使用XML配置定义路由和中介规则。

https://github.com/apache/camel

关键词：Endpoint 、 Routing、 Exchange、 Message、 Component、 Dataformat

逻辑视图


System A
Transport A
Transport B
Transport C
Transport ...
System B
System C
System ...
Routing Engine
Exchange
Exchange ID
Pattern
Exception
Properties
In Message
Headers
Attachments
Body
Out Message
Headers
Attachments
Body
Components
FTP
Netty
JDBC
Http
Kafka
Rest
Elasticsearch
... ...
Data formats
Base64
Json
Soap
Protobuf
Csv
Xstream
JacksonXml
... ...

集成用例

Integration Example

第二章

集成用例

- 代码解读
- Json2Json用例演示
- Json2SOAP用例演示
- 多资源串行用例演示
- 多资源选择用例演示

Integration Example

代码解读
CamelMain

https://github.com/tanchy82/CamelExchange/blob/master/src/main/scala/com/oldtan/camel/CamelMain.scala

CamelMain.scala

```scala
/**
  * @Description: Camel Exchange Main Method
  * @Author: tanchuyue
  * @Date: 21-4-30
  */
object CamelMain extends App with LazyLogging {
  val camel = new DefaultCamelContext
  val config = new Yaml(new Constructor(classOf[RestConfig]))
    .load(Source.fromResource("application-rest.yml").bufferedReader).asInstanceOf[RestConfig]
  val mapper = new ObjectMapper
  val xmlMapper = new XmlMapper
  xmlMapper.configure(ToXmlGenerator.Feature.WRITE_XML_DECLARATION, state = true)

  implicit def aa(r: AnyRef) = r match {
    case o: ChoiceDefinition => o.asInstanceOf[ChoiceDefinition]
    case o: RouteDefinition => o.asInstanceOf[RouteDefinition]
  }

  def addSingleRoute(rDef: AnyRef, toUri: String, toMethod: String, bean: ExchangeBean, namespace:String) = {
    rDef.process(new Processor {
      override def process(exchange: Exchange):Unit = {
        exchange.getIn.setBody(mapper.writeValueAsString(exchange.getIn.getBody.asInstanceOf[util.LinkedHashMap[String,
        toMethod match {
          case "WS" => {
            bean.requestExchange(exchange)
```

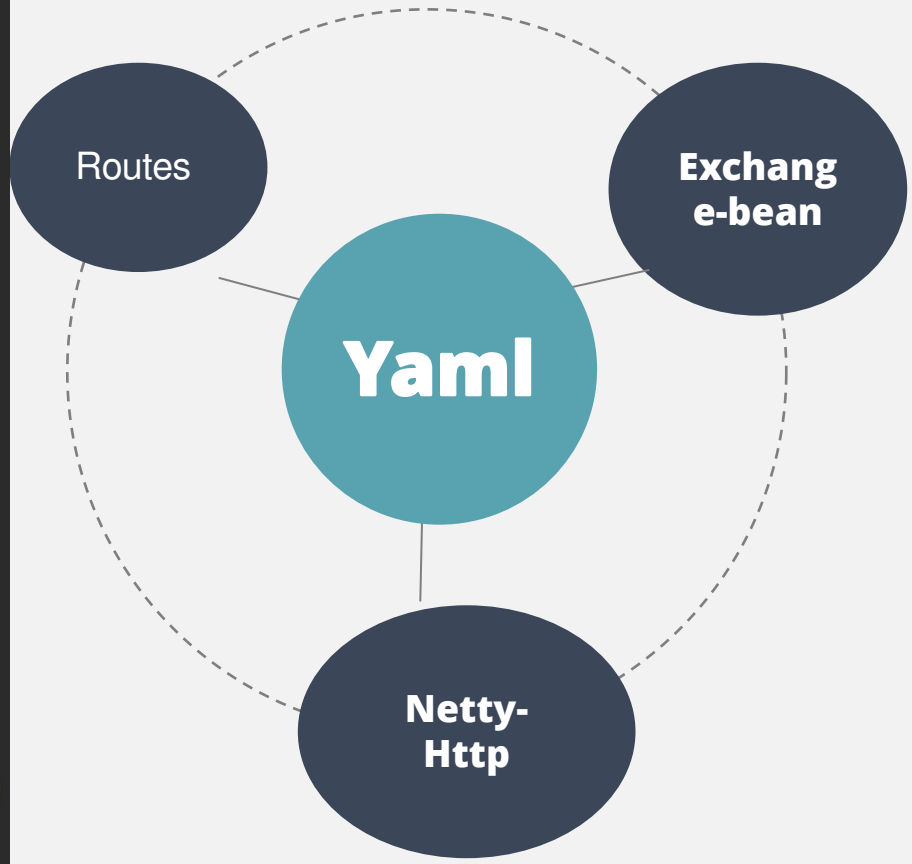
CamelMain

代码解读
Yaml

https://github.com/tanchy82/CamelExchange/blob/master/src/main/resources/application-rest.yml

application-rest.yml

```yaml
host: localhost
port: 7777
property: connectTimeout:5,compression:true
routes:
  - from: /his
    from_method: post
    model: Serial    # model have Serial、choice two route running model,default model is Serial
    to: localhost:8282/demo
    to_method: post
    exchange_bean: com.oldtan.camel.processor.DefaultExchangeBean
  ## If you set to uri contain path variable, then set follow the below
  #  destination endpoint uri is:  http://localhost:8282/demo222/{id}
  ## If you set more uri sort, then set follow the below
  #  set to/to_method/exchange_bean properties sort by slip ','
  - from: /demo222/{id}
    from_method: post
    model: Serial
    to: localhost:8282,localhost:8282/demo
    to_method: post,post
    exchange_bean: com.oldtan.camel.processor.DefaultExchangeBean,com.oldtan.camel.processor.DefaultExchangeBean
  ## If destination endpoint is web SOAP service, then set to_method attributes value 'ws' and must have 'namespace'
  - from: /tows
    from_method: post
    model: Serial
```



代码解读
ExchangeBean

https://github.com/tanchy82/CamelExchange/blob/master/src/main/java/com/oldtan/camel/processor/DefaultExchangeBean.java

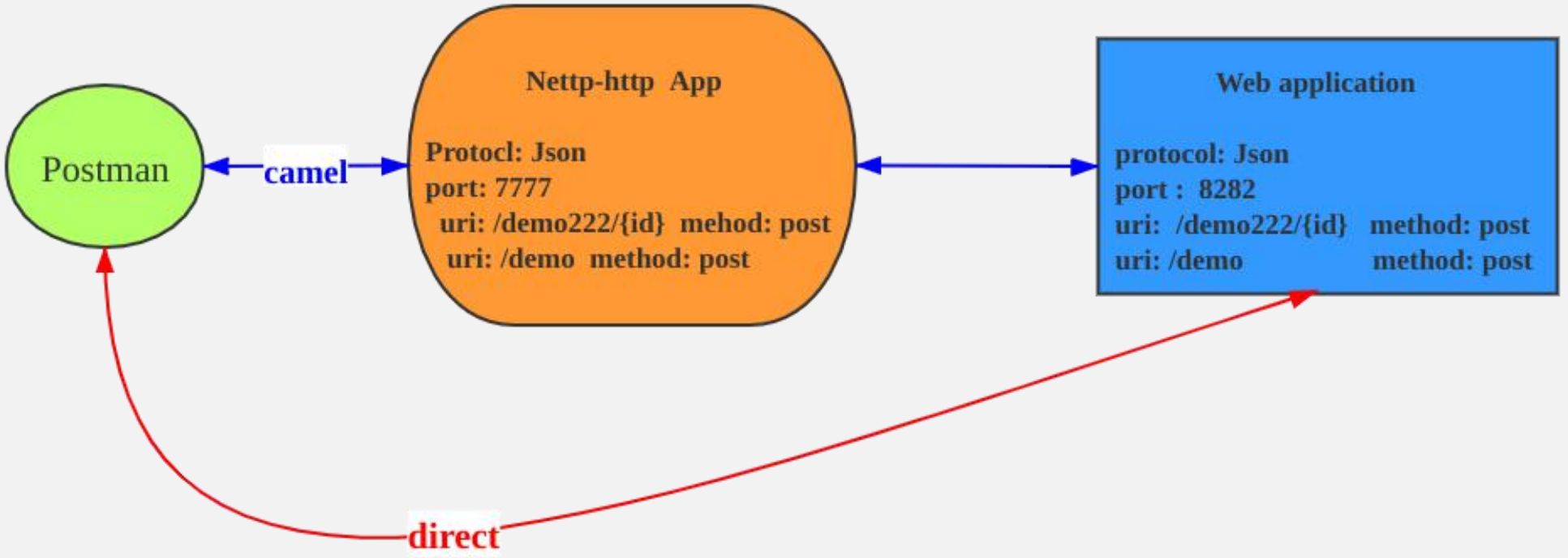
DefaultExchangeBean.java

```java
/**
 * @Description: Exchange data default bean
 * @Author: tanchuyue
 * @Date: 21-4-30
 */
public class DefaultExchangeBean implements ExchangeBean {

    private ObjectMapper mapper = new ObjectMapper();

    /**
     * Request exchange data handler method Integration Example
     * @param exchange
     */
    @Override
    public void requestExchange(Exchange exchange){...}

    /**
     * Response exchange data handler method Integration Example
     * @param exchange
     */
    @Override
    public void responseExchange(Exchange exchange){...}

}
```

实现接口com.oldtan.ExchangeBean

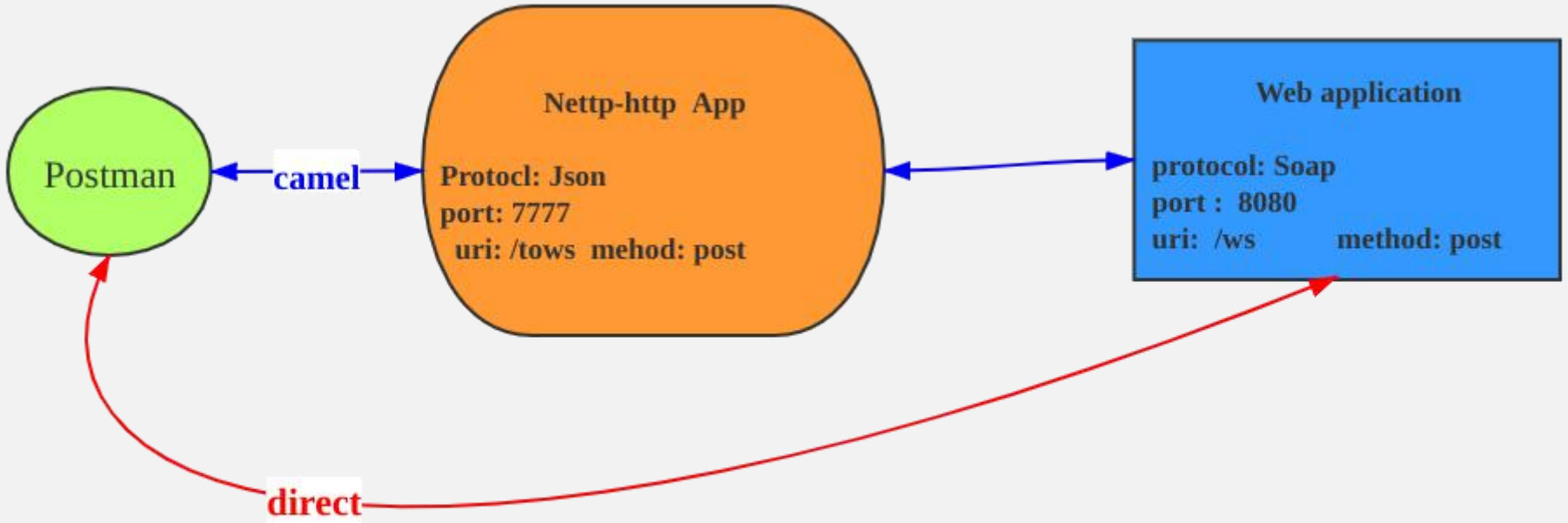
Json2Json

调用逻辑视图



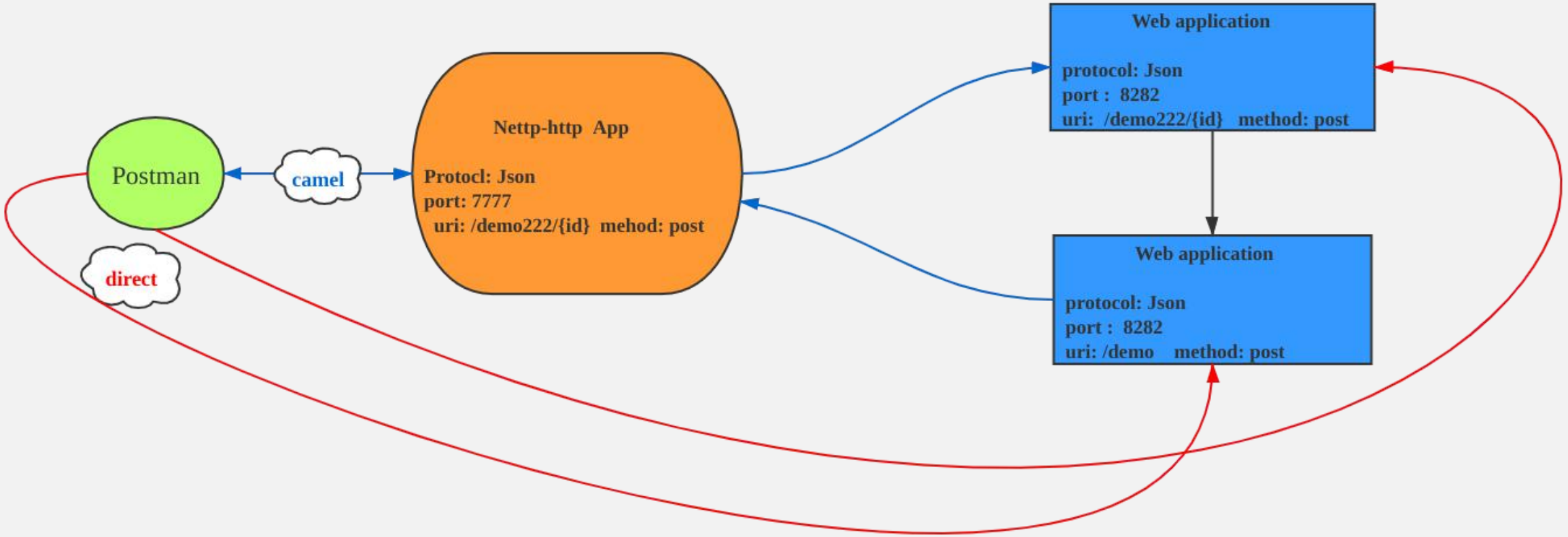
Json2SOAP

调用逻辑视图



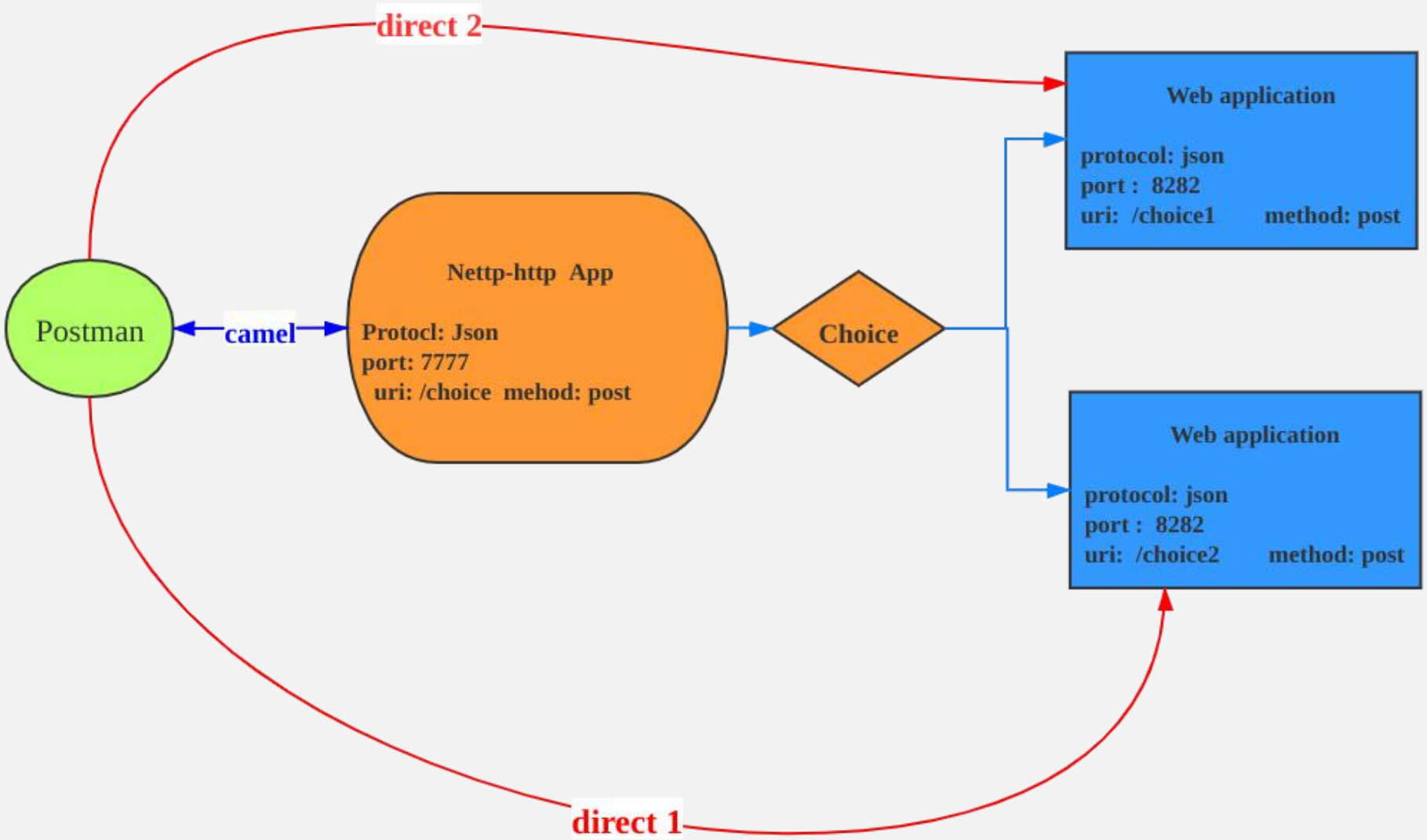
多资源串行

调用逻辑视图



多资源选择

调用逻辑视图



成果使用和需求定制

communication

第三章

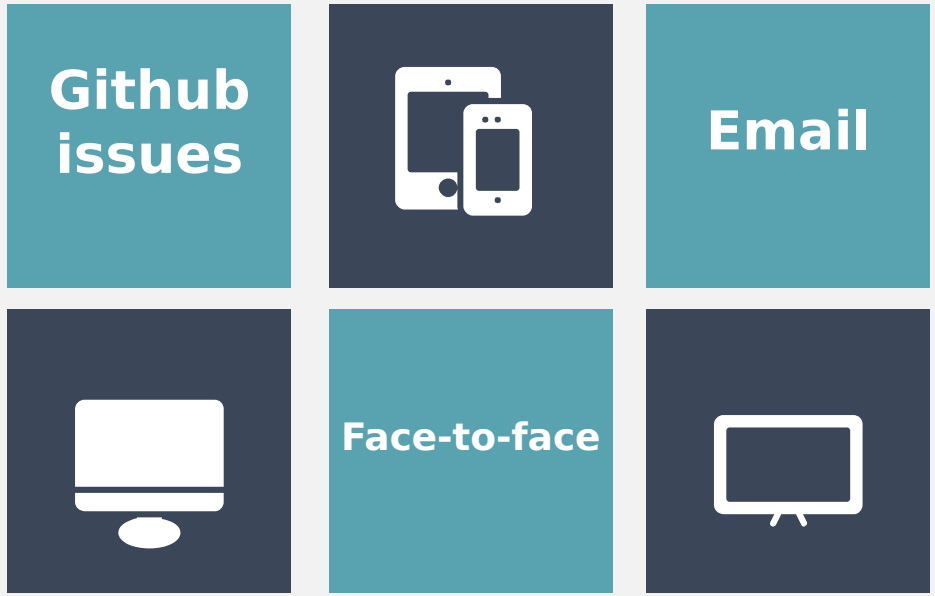
业务场景多样化

需求定制

Apache camel 是一个非常实用的规则引擎库，能够用来处理来自于不同源的事件和信息。你可以在使用不同的协议比如VM，HTTP，FTP，JMS甚至是文件系统中来传递消息，并且让你的操作逻辑和传递逻辑保持分离，这能够让你更专注于消息的内容。

Camel的定位是轻量级的ESB，使用场景可大可小，即可嵌入到Springboot中当成jar使用，亦搭建成单独的微服务使用。本次所分享的用例仅仅是冰山一角，更多用例需要基于业务场景及大家共同努力来分享、完善。

- 关于Camel的更多官方文档详见 https://camel.apache.org/
- 如遇到类似业务场景时如有需要可参考或直接拉取Github代码
- 如需要进行业务定制可通过Github issues、Email、Face-to-face等沟通方式相互探讨。
- 在转载、修改、使用Github代码亦请遵循GNU General Public License v3.0协议，坚持开放、开源、共享的开源精神。
- Github: https://github.com/tanchy82/CamelExchange
- Email: tanchy@neusoft.com



communication

Neusoft

谢谢！

谭初跃 2021-06