Article
Istvan Hahn · Oct 21, 2016 4m read

Applying MicroServices Architecture to Ensemble

A beginner ' s guide to position Ensemble in regards to MicroServices Architecture (MSA). MSA is getting more visibility in the Enterprise Java world therefore it is vital to understand what is behind the buzz. I make a (humble) attempt to write my view and share with you.

Background

First of all I must confess. Early this summer our Czech colleague Daniel Kutac was asking me to collect some information about a health care product developed in Hungary. When I got the feedback from the related company, it turned out, that the product is a modular system, based on MicroServices Architecture (MSA). OK. What is MSA? After a short research on web, reading some WIKI pages I concluded like: aha, yet another " architecture " . We can comply with " any architecture " with Ensemble at " architecture level " – at least " somehow " . But there were some annoying moments. That made me read more about MSA. So far so good, but is not REST the main topic of the article? What is the relationship between MSA and REST? Almost every site I visited used RESTful web services in the examples. That made me a little suspicious. But overall I echoed the generic summary of MSA: it is SOAP but done right. Specially because there were sentences about " cohesion " and " coupling " which I did not understand at all. So if MSA is just " SOAP well done " why I do not see SOAP examples? Why the examples are (almost) always RESTful?

Up until that question for me " REST " was a web service using JSON serialized data exchange transmitted via HTTP. But at that point I had too many questions in my head, therefore I started to read the REST literature. Soon I found the origin. The term REST comes from a gentleman called Roy Thomas Fielding. He introduced it in his dissertation in 2000. The abbreviation comes from REpresentational State Transfer which is – in his word – " an architectural style for distributed hypermedia systems " . Alright. By reading the dissertation, it became evident that I must update my English vocabulary. The first item on my list was architecture. Again I cite Fielding: " A **software architecture** is an abstraction of the run-time elements of a software system during some phase of its operation. A system may be composed of many levels of abstraction and many phases of operation, each with its own software architecture. " Without doubt in this context architecture is not a concept, not a structure. It is a description of a real deployed run-time. The REST is not a concept. The dissertation precisely defines the implementation constraints of REST.

Then how comes MSA into the picture? MSA is an architecture (remember the meaning). It is not a concept. Rather a set of run-time elements. For me MSA is REST++. In my opinion some of the constraint of REST became obsolete, and those elements are challenged by MSA. That is why I talk about MSA in this article instead of REST.

When I realized all this things, I wanted to write this article only. But then I found out, that without having a common and solid basement made of understanding what MSA/ REST means, I can not explain my self properly. Therefore I started a series of beginner ' s guide to RESTful implementation. Although I call these articles " beginner ' s guide " in fact I strongly believe that not only beginners can take some new knowledge by reading them.

How does Ensemble fit to MSA?

I have a bad news…

MSA is a distributed system architecture. MSA defines run-time elements making up the architecture. It is very much tied to Enterprise Java (even if there are language bindings to non-Java components). It is no wonder that one of the major drivers of J2EE development, spring framework offers MSA solution. But if it is so much tied to

Java, what can Ensemble do for it?

In the heart of MSA there is a component call registry. One of the registry implementations is Apache ZooKeeper. Visiting the site at the first time made me a bit confused (yet again). The banner of the site tells " Apache ZooKeeper is an effort to develop and maintain an open-source server which enables highly reliable distributed coordination ". Please remember, MSA is a distributed system architecture. That is the registry is responsible for client side service discovery as well as distributed transaction coordination. Services managing fragments of data when the integrity of the entire data set is important must use some coordination. This is done by the registry. Is Ensemble prepared to participate that coordination?

I stop it, because it turns too negative.

And I have a good news too. Do you remember how Fielding defined software architecture? In case you do not, I repeat part of it: " A system may be composed of many levels of abstraction and many phases of operation, each with its own software architecture ".

Ensemble has its own strengths. I am sure you can list those endlessly. Ensemble fits a lot of different architectures. What if MSA is not on the list? Even Ensemble can provide RESTful services which is half way to MSA. Also. An important component type in MSA is the API Gateway. Its primary purpose to encapsulate legacy systems into microservice. Ensemble with its extensive library of " legacy adapters " and graphical data transformation tool is far more than what an API Gateway implementer dreams of.

That was it folks. I hope you enjoyed the series. In case you missed, here is the list of the previous articles in the series.

n We discussed how to create a service.

n How and why to enable CORS .

n Consume a service as a client.

n How to pass service parameters to a service.

n We also learnt handling exceptions.

n As well as to design a service API.

n And lastly design a service content.

#Beginner #Microservices #REST API #Ensemble