# Requisitos para implementação da interface REST

Esta seção visa facilitar o entendimento do componente que permite a utilização da interface REST. Desde começo da codificação, alguns requisitos foram adotados para garantir a qualidade:

1. Compatível com OSGi: a implementação deve ser compatível com a plataforma OSGi;
2. Extensível: a solução deve permitir a inserção de novos *paths* e *resources* de acordo com a necessidade da arquitetura;
3. *Hot deploy*: o *deploy* de cada interfacedeve ser realizado sem a necessidade de interromper a execução dos demais componentes da arquitetura;
4. Baixo acoplamento interface: as interfaces devem ser independentes;
5. Reusabilidade: os componentes principais devem ser reusáveis por todas as interfaces, incluindo os diferentes *paths* e *resources*;

Para atingir esses requisitos, diversas bibliotecas foram analisadas e alguns POCs[[1]](#footnote-1) foram construídos.

# Implementação da interface REST

Para a implementação da interface REST foi escolhida a biblioteca RESTLet[[2]](#footnote-2) pois permite atender aos requisitos definidos anteriormente. A implementação do RESTlet é organizada como um conjunto de componentes e interfaces que permitem registrar uma apllicação REST usando declarative services. A partir desses componentes é possivel implementar uma simples aplicação com rotase recursos ou uma aplicação complexa com filtros, rotas e recursos, como demonstrado na Figura 1.

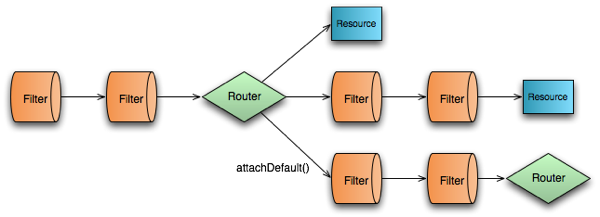


Figura 1 - Implementação Restlet

# Como testar o *sample*?

Após iniciar a plataforma OSGi é preciso instalar o *bundle sample*  e as devidas referências.

Em seguida, basta realizar as requisições abaixo a partir de um cliente REST. Existem vários clientes REST, mas sugerimos utilizar o aplicativo para o navegador Chrome chamado “Rest Console” (https://chrome.google.com/webstore/detail/rest-console/cokgbflfommojglbmbpenpphppikmonn?hl=en). Após instalar o aplicativo a partir da *Chrome Store*, basta acessar o aplicativo e fornecer as seguintes informações:

* **Teste GET:**
  + Endereço: http://localhost:8080/ping/1
  + Método: GET
  + Resultado esperado: {"id":1,"name":"teste1"}
* **Teste POST:**
  + Endereço: http://localhost:8080/ping
  + Método: POST
  + Raw Body: {"id":2,"name":"teste2"}
  + Content-type: application/json
  + Resultado esperado: {"result": "Model saved succesfully", "model": "{\"id\":2,\"name\":\"teste2\"}"}
* **Teste PUT:**
  + Endereço: http://localhost:8080/ping
  + Método: PUT
  + Raw Body: {"id":1,"name":"teste33"}
  + Content-type: application/json
  + Resultado esperado: {"result": "Model saved succesfully", "model": "{\"id\":2,\"name\":\"teste33\"}"}
* **Teste DELETE:**
  + Endereço: http://localhost:8080/ping/1
  + Método: DELETE
  + Resultado esperado: {"result":"Model deleted succesfully"}

Após fornecer essas informações basta clicar em Send e verificar o resultado abaixo.

1. Do termo inglês *Proof of Concept* [↑](#footnote-ref-1)
2. www.restlet.org [↑](#footnote-ref-2)