REST e MASHUP

Israel Lacerra
Omar Ajoue
Roberto Bodo
Thiago Coraini

O que é REST?

- Trabalho de Ph.D. de Roy Fielding
Co-fundador da Apache Software Foundation

- É um estilo arquitetural de sistemas em rede.
    > Web Services

- REpresentational State Transfer

REpresentational State Tranfer?

- Exemplos de estados:

http://www.burgerking.com.br/cardapio

http://www.burgerking.com.br/cardapio/sanduiches

http://www.burgerking.com.br/cardapio/sanduiches/whooper-triplo

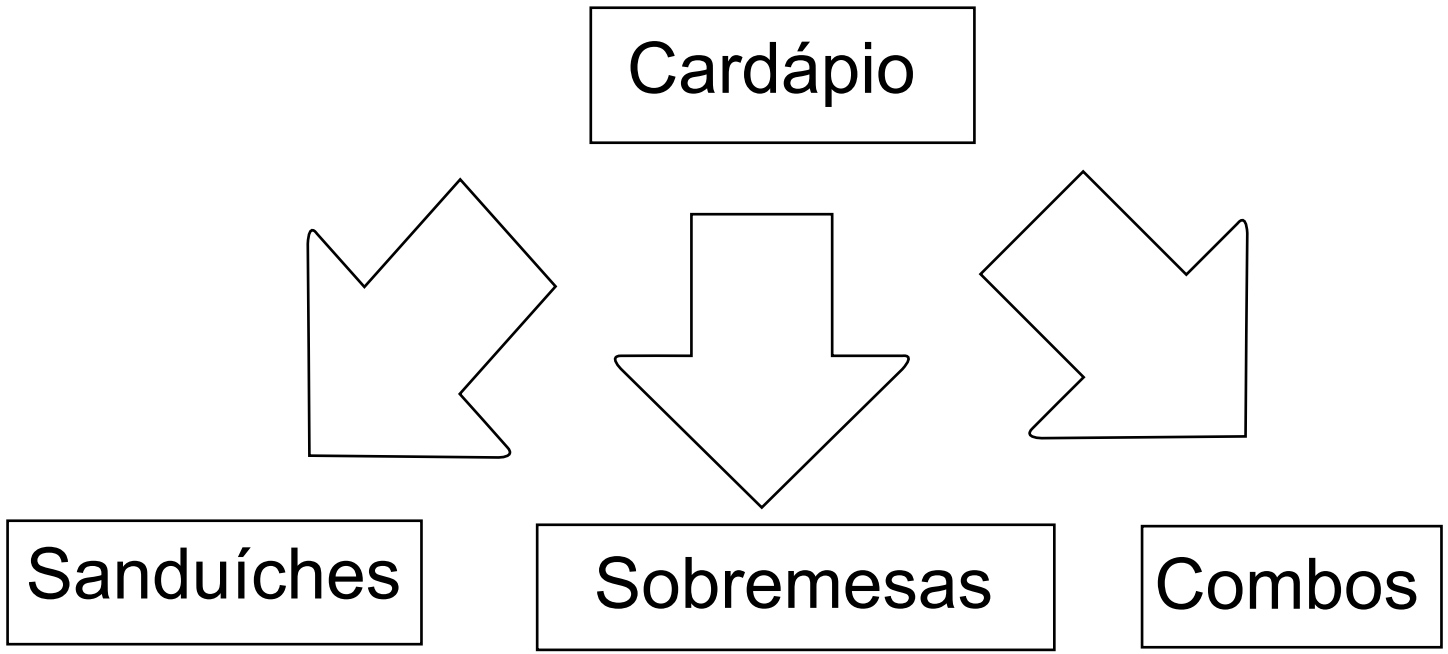
http://www.burgerking.com.br/cardapio/sobremesas

Resultado de um estado

- O estado http://www.burgerking.com.br/cardapio pode devolver:

```
<?xml version="1.0"?>
<p:Cardapio xmlns:p="http://www.burgerking.com.br/cardapio"
xmlns:xlink="http://www.w3.org/1999/xlink">
<Tipo nome="Sobremesas"
    xlink:href="http://www.burgerking.com.br/cardapio/sobremesas"/>
<Tipo nome="Sanduiches"
    xlink:href="http://www.burgerking.com.br/cardapio/sanduiches"/>
<Tipo nome="Combos"
    xlink:href="http://www.burgerking.com.br/cardapio/combos"/>
</p:Cardapio>
```

Idéia de "Estados" - Semelhante a autômatos



. . .

REST não é nada novo

- Bem como Ajax, o REST não é uma nova tecnologia, nem uma nova biblioteca e muito menos uma nova linguagem.

- REST é um conjunto de conceitos sobre como escrever Web Services

REpresentational State Transfer!!!

- Agora que entendemos o que são os estados e que são navegáveis, podemos entender o que singifica o REpresentational State Tranfer.

- Transferência de estados representacionais.

- Mas o quê representam os estados?

ROA - Resources Oriented Architeture

- Arquitetura orientada a recursos

- Cada URL representa um recurso.

- Exemplo:
http://blog.exemplo.com/posts
Representa o recurso Posts

http://blog.exemplo.com/posts/515
Representa uma instância do recurso posts

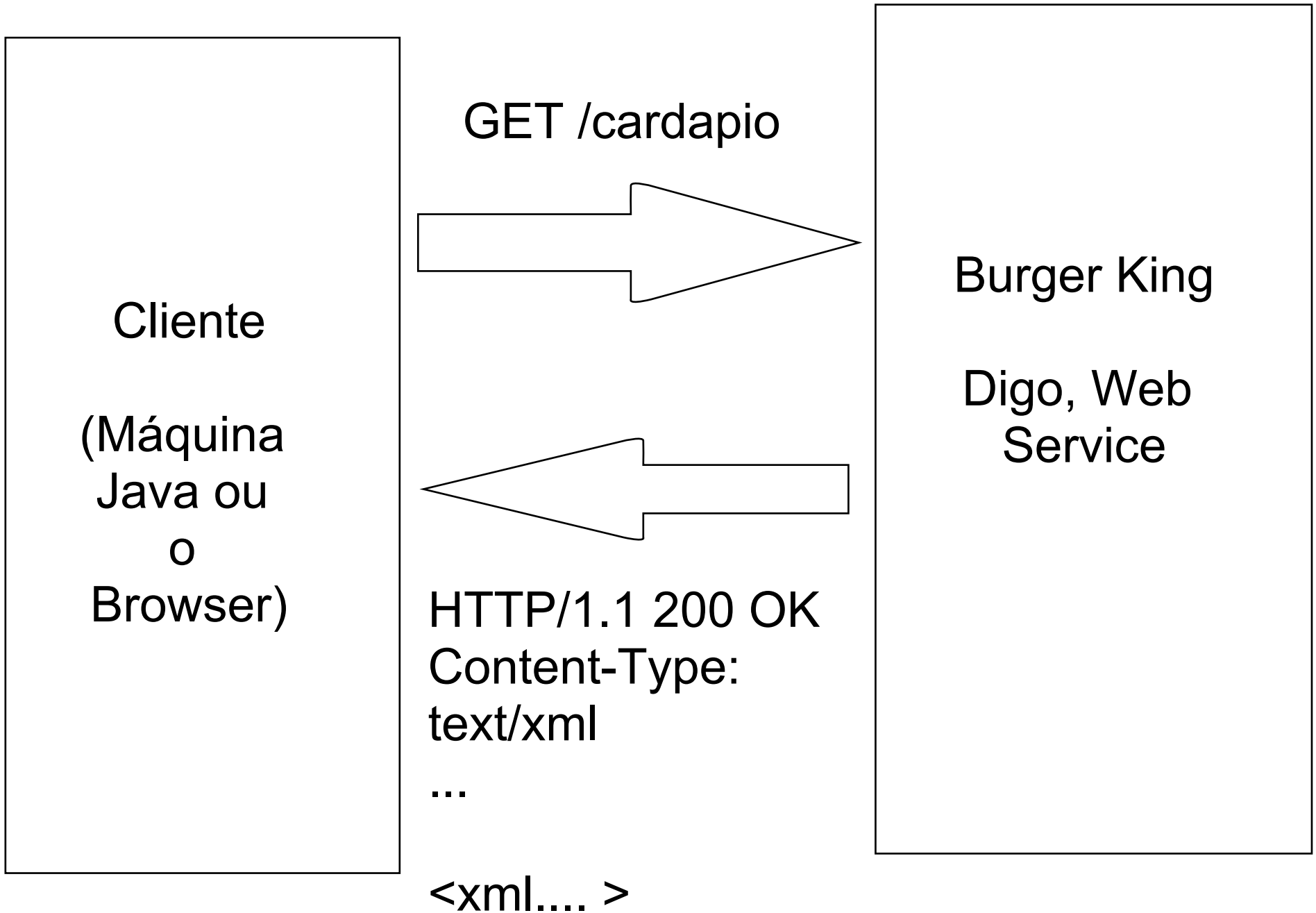
http://blog.exemplo.com/posts/515/comments
Representa o recurso comments da instância 515 do recurso posts

ROA - Resources Oriented Architeture

- Como ficaria o XML do exemplo anterior?

```
<?xml version="1.0"?>
<p:Post xmlns:p="http://blog.exemplo.com"
xmlns:xlink="http://www.w3.org/1999/xlink">
<Post-ID>515</Part-ID>
<Titulo>Teste de Post</Titulo>
<Texto>Isto é o conteúdo do post!</Texto>
<Comentarios
        xlink:
        href="http://blog.exemplo.com/posts/515/comentarios"
    />
<Autor>Anônimo</Autor>
</p:Post>
```

Recebimento de informações



E o restante das operações?

- Entendemos que conseguimos obter informações com facilidade utilizando REST.

- Então, podemos enviar informações também. Afinal, é uma maneira de criar Web Services.

Exemplo:
GET /posts/515/comments/?
action=addcomment&texto=.....

Certo?

E o restante das operações?

ERRADO!

- O estilo arquitetural REST prega pelo uso dos verbos HTTP, conhecidos como Method, para identificar a ação.

- O exemplo correto para adicionar informações seria:
POST /posts/515/comments

- As informações relacionadas à requisição são enviadas dentro do envelope HTTP, compondo o corpo da mensagem.

E o restante das operações?

- O HTTP nos fornece os seguintes verbos:

GET: Utilizado para extrair (obter) informações

POST: Utilizado para adicionar registros

HEAD: Utilizado para verificar a existência de tal recurso

DELETE: Utilizado para remover um recurso

PUT: Utilizado para sobrescrever um recurso existente ou criar um novo

Esclarecimentos

- URL lógica versus URL representacional

- REST não é SOAP e não deve ser interpretado como RPC

Vantagens de se utilizar REST

- Facilidade de implementação pois independe da linguagem e de peculiaridades, pois utiliza um recurso básico: o HTTP

- Facilidade de adaptação com baixo acoplamento: pode-se utilizar XML, mas também podemos utilizar JSON, texto puro e até imagens no corpo da requisição

- Utilização dos verbos HTTP para realizar ações, tornando a interface uniforme

- Navegação por intermédio de links contidos na mensagem

- Facilidade de instalação com relação ao hardware

Desvantagens / características

- Programação orientada a recursos: substantivos ao invés de verbos

- Nem todos os proxies / firewalls aceitam verbos HTTP além dos velhos POST e GET
- Dificilmente poderíamos enviar quantidades grandes de informação
(> 4kb)

- Criticado pelas grandes produtoras de software pois não atende a todas as necessidades; considerado simplificado demais

Características

- Utilização de CACHE para reduzir o tráfego da rede

- Cada mensagem deve conter todas as informações necessárias

- URLs representam recursosbem definidos

- Navegação utilizando hiperlinks para "cavar" mais informações

- Descrição do funcionamento por WSDL ou mesmo um HTML

SOAP versus REST

- Pontos para o SOAP

Mais antigo, portanto, mais bem documentado e mais aplicado.
Bastante completo e complexo.

- Pontos para o REST

Simples, mais fácil de utilizar.
Dito simples até demais por alguns.
Independe da linguagem ou de negociações mais complexas, como firewalls, proxies e portas.

MASHUP

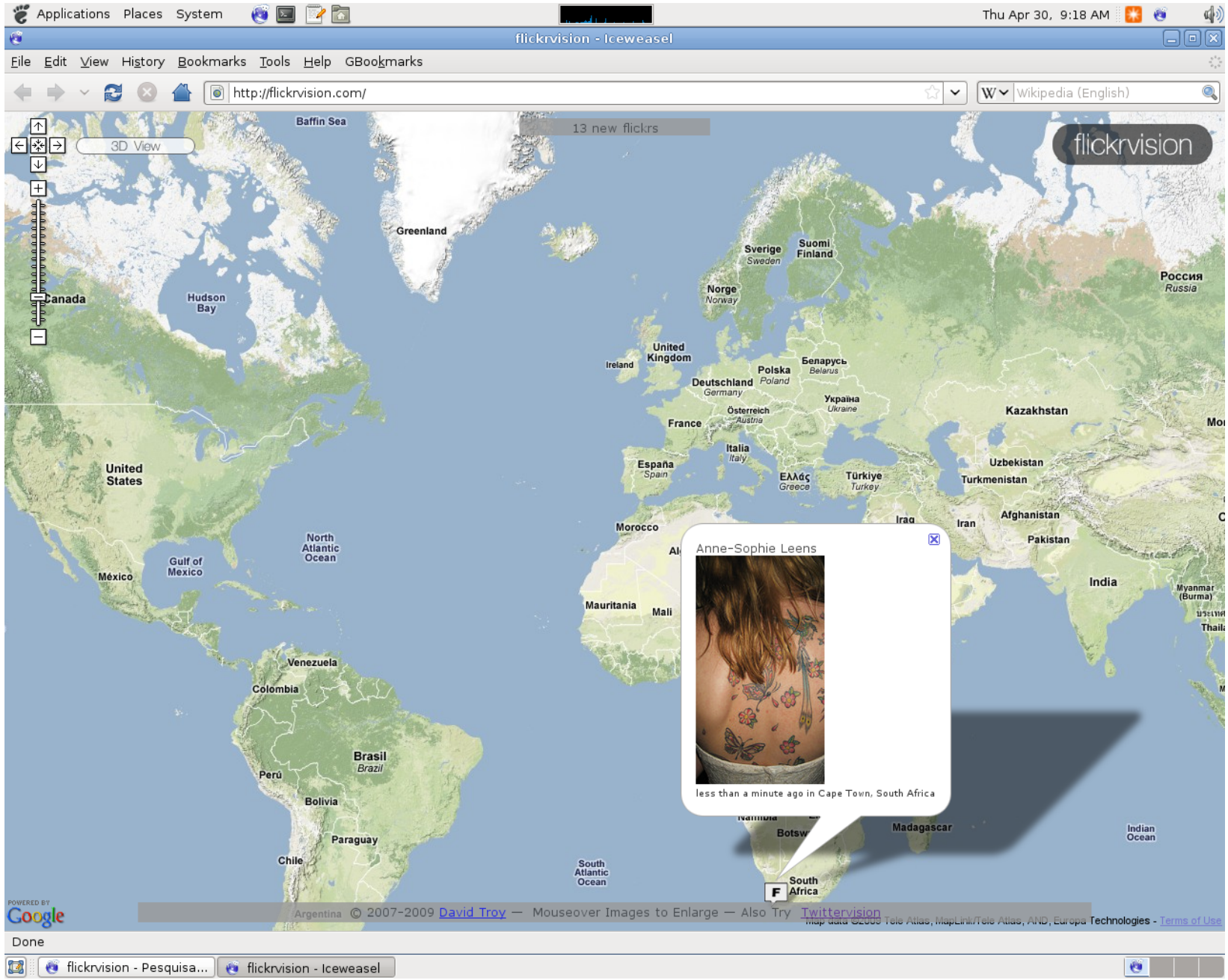
O que é Mashup?

- Um Mashup pode ser um site ou uma aplicação web que é composta da união de funcionalidades e conteúdo existentes em outras fontes.

- Exemplos:

    - chicagocrime.org
    - tv.timbormans.com
    - flickrvision.com

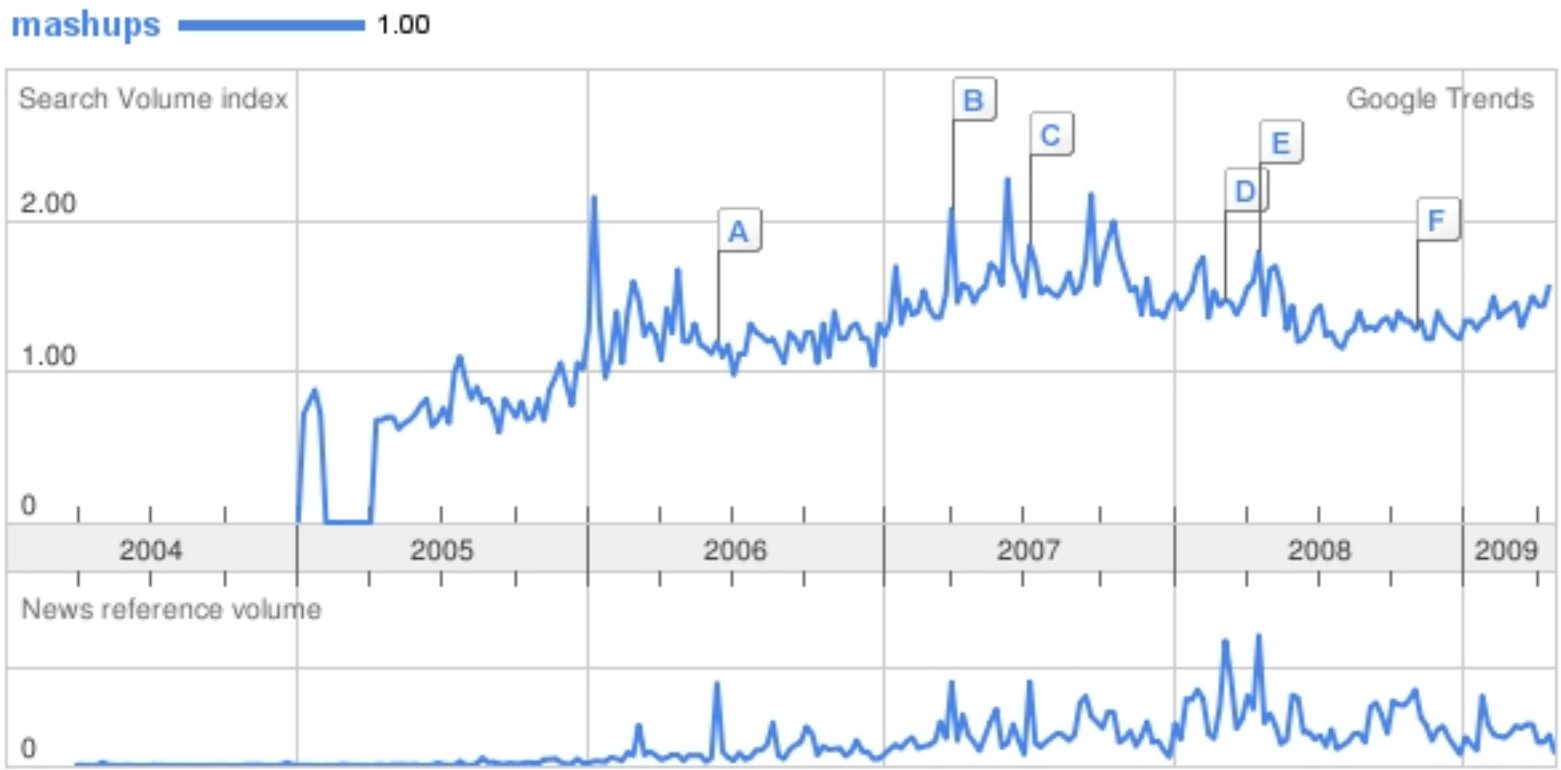
flickrvision.com



Como fazer um Mashup?

- Temos alguns problemas básicos:
    - recuperação de dados;
    - classificação dos dados obtidos;
    - limpeza dos dados;
    - integração dos dados;
    - visualização final.

Popularidade

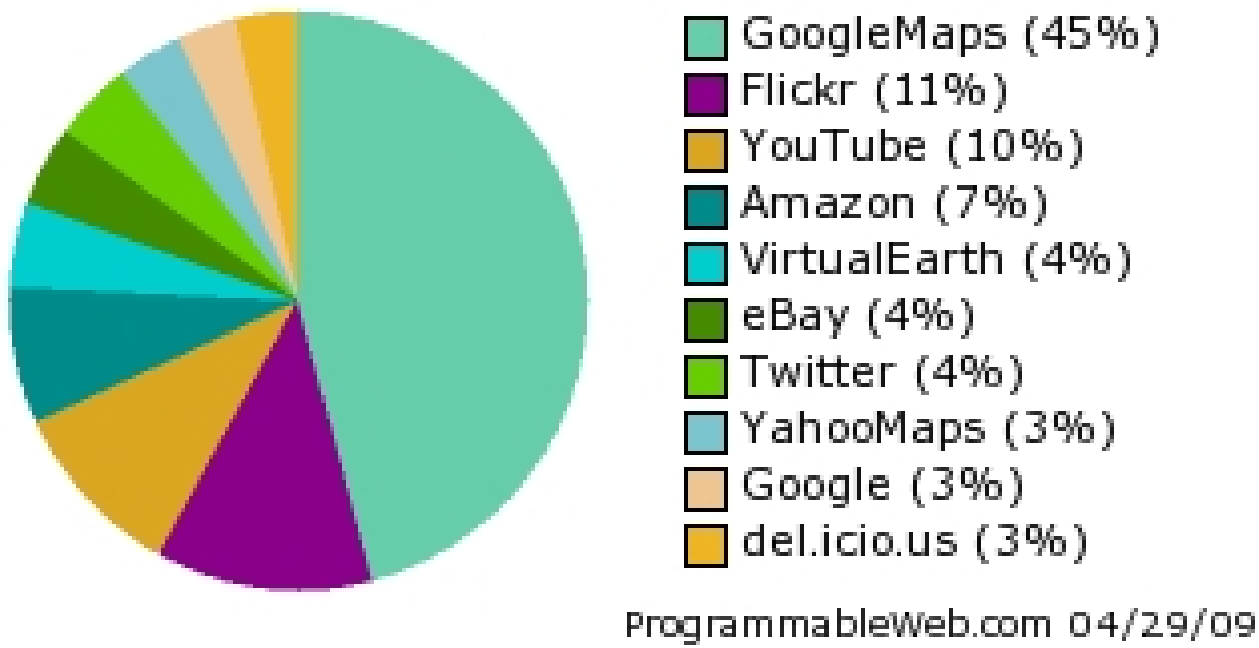


- Quais os motivos?

APIs para Mashups

- aumento do número de provedores de conteúdo.

- Exemplos:



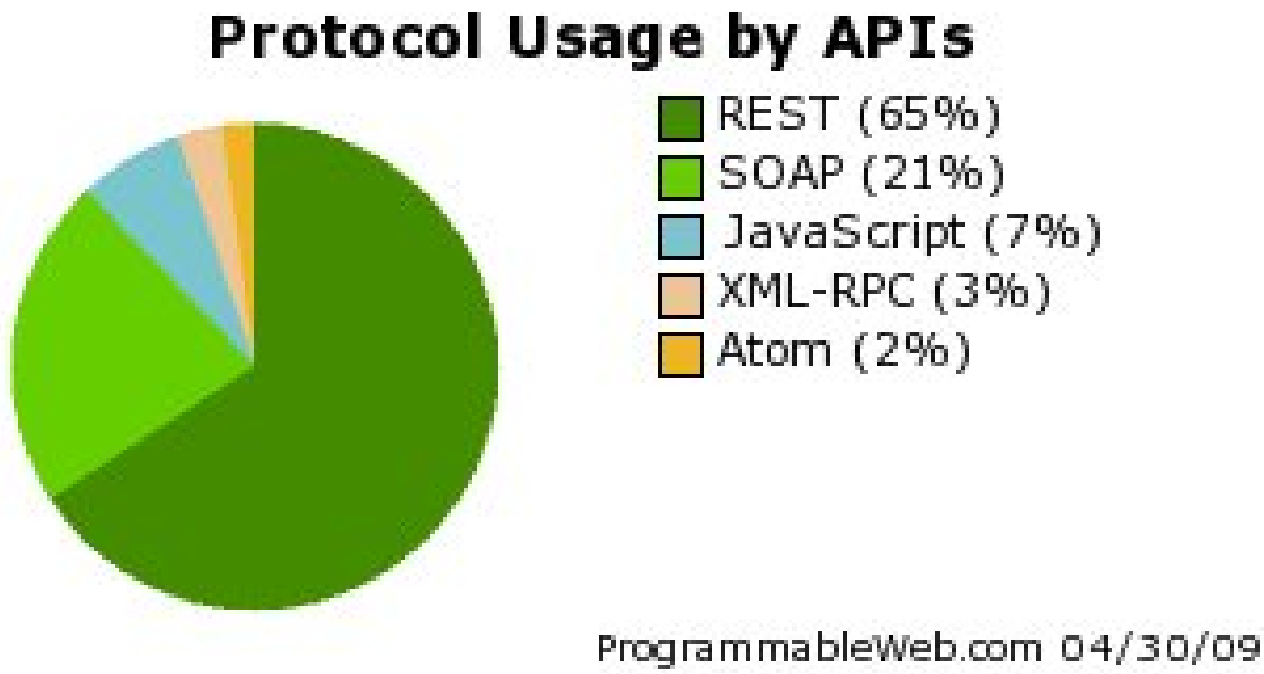
Editores para Mashups

- a criação de editores gráficos de Mashups para usuários comuns da Internet.

- Exemplos:

  - Google Mashup Editor
  - Yahoo Pipes
  - Microsoft Popfly

Mashups + REST



Referências

xFront. Building Web Services the REST way.
http://www.xfront.com/REST-Web-Services.html

Wikipedia. Representational State Transfer.
http://en.wikipedia.org/wiki/Representational_State_Transfer

C. Pautasso, O.Zimmermann, F. Leymann. RESTful Web Services vs. "Big" Web Services: Making the Right Architetural Decision

L.Richardson, S. Ruby. RESTful Serviços Web.

Perguntas?

Dúvidas?

Comentários?

Sugestões?