Representational State Transfer - REST

Aurélia BRUNEAU

mars 2008

PLAN

- Introduction à REST
- Les principes clefs
- Développer un service web RESTful
- REST et les Web Services WS-*
- Cas d'utilisations
- Conclusion

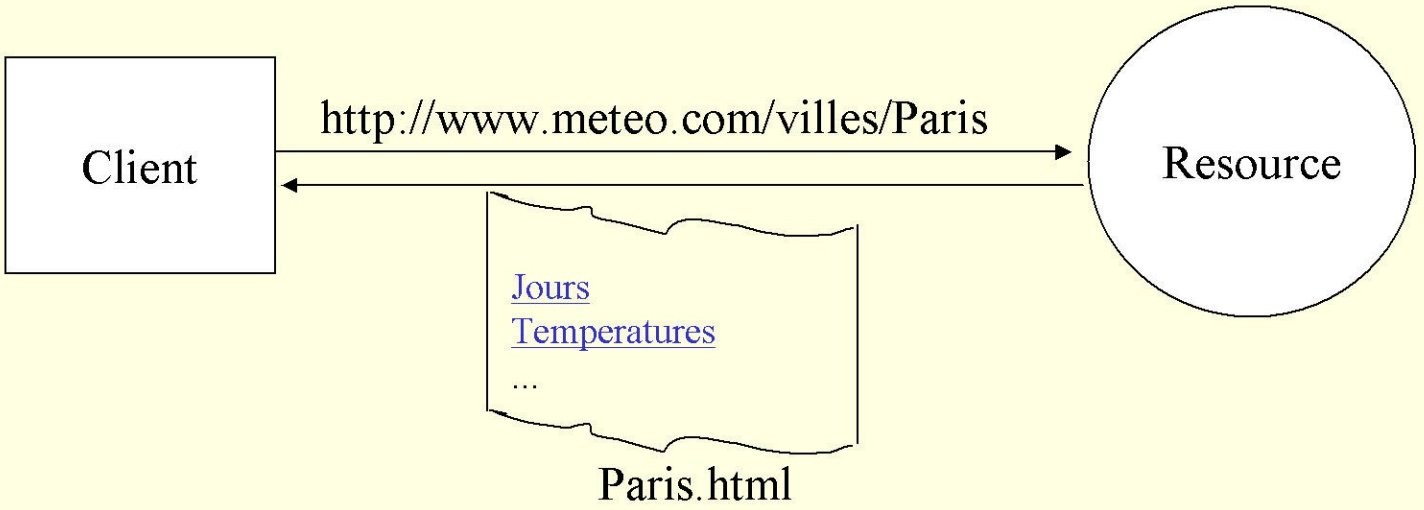
REST c'est quoi ?

- Thèse de Roy Fielding en 2000
- Un style d'architecture
- Un ensemble de contraintes
  - Client /serveur
  - Sans états (Stateless)
  - Cache
  - Interface uniforme
- La plus connue des implémentations de REST est HTTP

Les principes clefs

- Une ressource
- Un identifiant de ressource
- Une représentation
- Interagir avec les ressources
  - Exemple avec HTTP : GET, POST, PUT et DELETE

Pour résumer



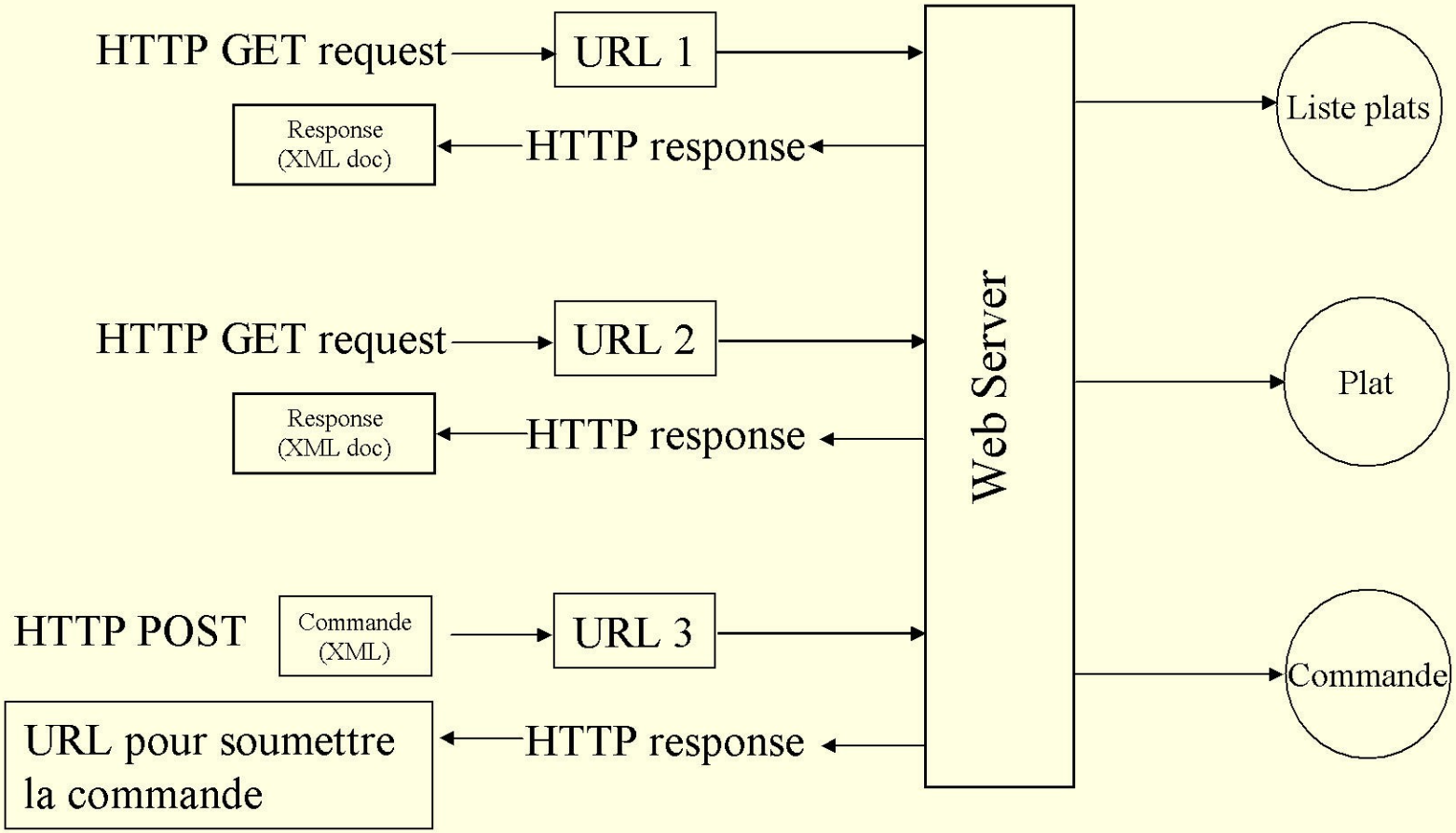
Un service RESTful

- Identifier les ressources
- Définir les URIs
- Spécifier les méthodes des interfaces
- Lier les ressources

Exemple (1/6)

- Un traiteur propose sur son site plusieurs services à ses clients :
  - Obtenir la liste des plats disponibles
  - Obtenir des informations sur un plat précis
  - Passer une commande

Exemple (2/6)


HTTP GET request
URL 1
Response
(XML doc)
HTTP response
HTTP GET request
URL 2
Response
(XML doc)
HTTP response
HTTP POST
Commande
(XML)
URL 3
URL pour soumettre
la commande
HTTP response
Web Server
Liste plats
Plat
Commande

Exemple (3/6)

- La liste des plats est disponible à l'URL suivante : http://www.monresto.com/plats/
- Le client reçoit une réponse sous la forme suivante :

```
<?xml version="1.0"?>
 <p:Plats xmlns:p="http://www.monresto.com/"
xmlns:xlink="http://www.w3.org/1999/xlink">
   <Plat id="0001" xlink:href="http://www. monresto.com/Plats/0001"/>
   <Plat id="0002" xlink:href="http://www. monresto.com/Plats/0002"/>
   <Plat id="0003" xlink:href="http://www. monresto.com/Plats/0003"/>
[...]
 </p:Plats>
```

Exemple (4/6)

- Les détails d'un plat se trouvent à l'URL : http://www.monresto.com/plats/**0002**
- D'où la réponse :

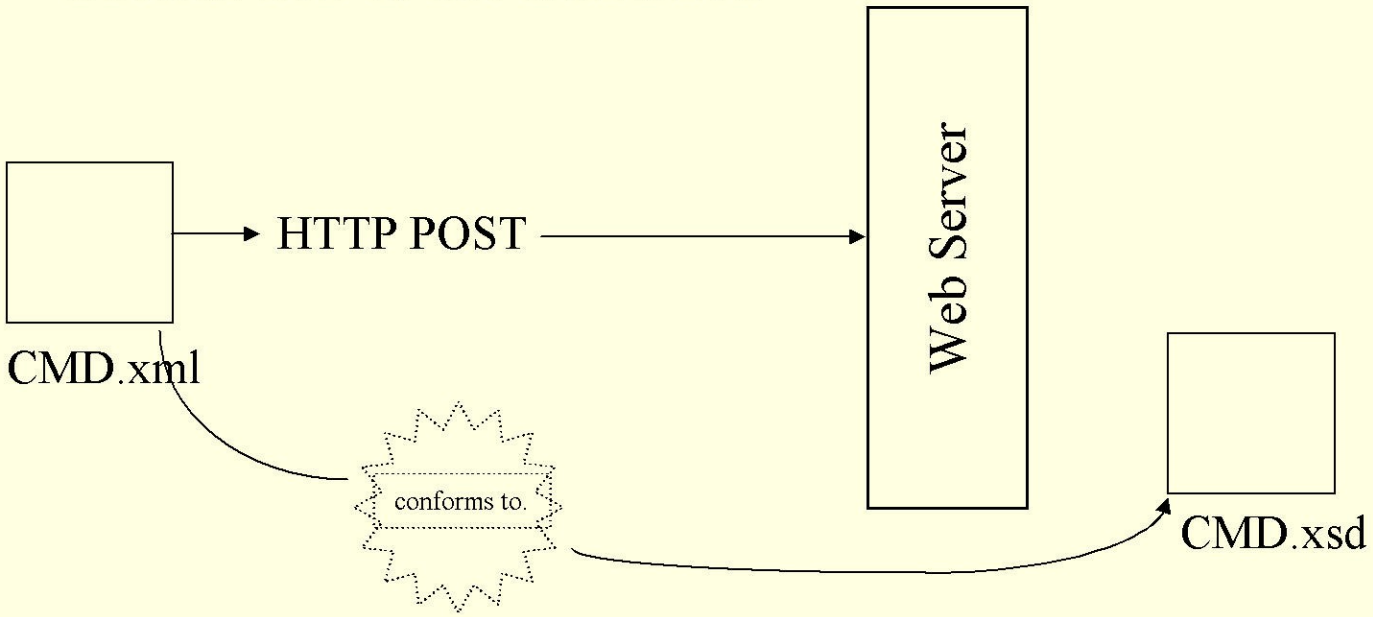
```
<?xml version="1.0"?>
 <p:Plat xmlns:p="http://www.monresto.com"
          xmlns:xlink="http://www.w3.org/1999/xlink">

  <Plat-ID>0002</Plat-ID>
  <Nom>Rouleaux de Printemps</Nom>
  <Description>Entrée</Description>
  <Details xlink:href="http://www.monresto.com/plats/00002/details"/>
  <CoutUnitaire monnaire="EUR">3</CoutUnitaire>

 </p:Plat>
```

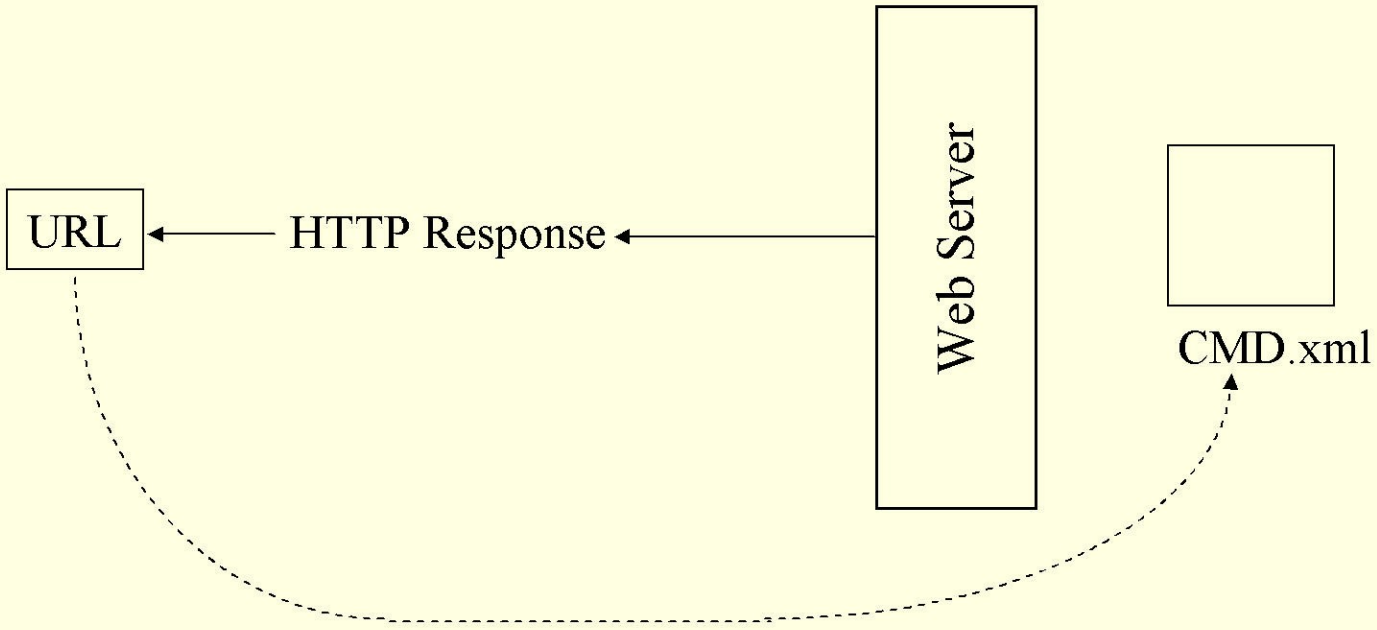
Exemple (5/6)

- Le service « Passer commande »
  - Créer une instance de « commande » conforme à un schéma



Exemple (6/6)

- Le service « Passer une commande » répond par une URL vers la commande soumise.



REST, Web Services WS-*

- WS-*
  - RPC (Remote Procedure Call)
  - SOA (Service Oriented Architecture)
- WSDL (Web Service Description Language)
- SOAP (Simple Object Access Protocol)
  - XML-RPC
  - Getter et setter

Cas d'utilisation

- La blogosphere
- Atom Publishing Protocol (RFC 5023)
- Amazon S3 (Simple Storage Service)
- Google Data API

Les avantages

- Simplicité
  - Liens structurés et de façon universelle
- Stateless
  - consommation de mémoire inférieure
  - mise au point plus simple, moins de cas à traiter
- URI uniques
  - mise en cache possible donc meilleure montée en charge
- Proche de la philosophie d'HTTP
  - moins de dépendances à une implémentation particulière

Les inconvénients

- Stateless
  - Les données de session chez le client
- Un nombre important de ressources
  - La gestion de l'espace de nom des URI peut devenir encombrante

Conclusion

- Une architecture orientée ressources
- Du vieux pour faire du neuf
- Simplicité et interopérabilité

Sources

- La thèse de de Roy T. Fielding
  - http://www.ics.uci.edu/~fielding/pubs/dissertation/top.htm
  - http://www.ics.uci.edu/~fielding/pubs/dissertation/rest_arch_style.htm
- http://fr.wikipedia.org/wiki/
  - Representational_state_transfer
  - Simple_Object_Access_Protocol
  - Liste_des_specifications_des_Services_Web_WS-*