

Services et Web Services, modèles et implémentations

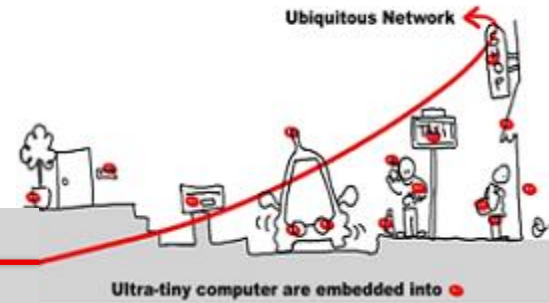
4 cours / 4 TDs

Jean-Yves Tigli

<http://www.tigli.fr>

Polytech of Nice - Sophia Antipolis University

[Email : tigli@polytech.unice.fr](mailto:tigli@polytech.unice.fr)

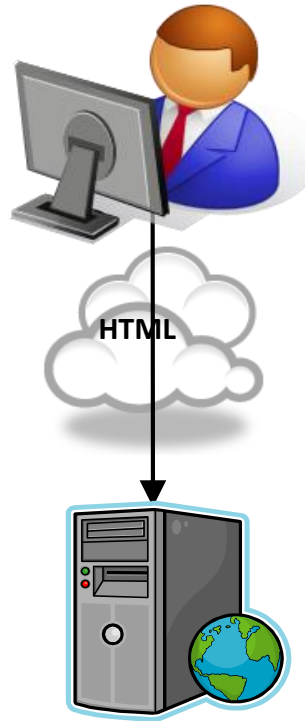
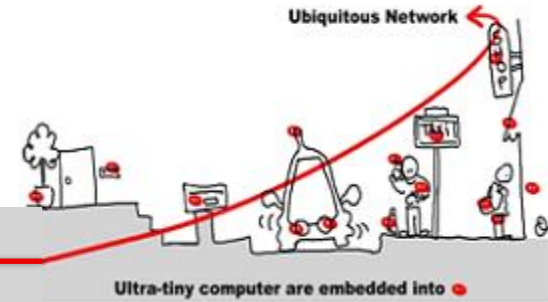


Introduction aux Services Web RESTful

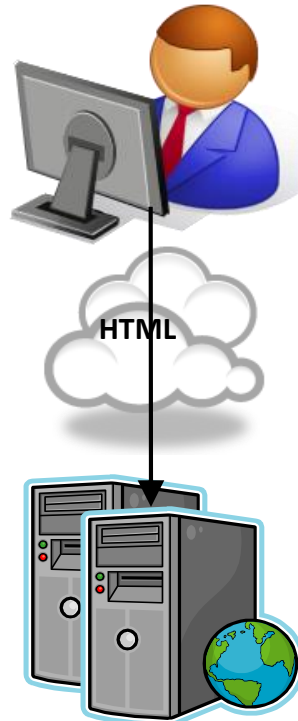
Chapitre 5 de «Fielding, Roy Thomas. Architectural Styles and the Design of Network-based Software Architectures. Doctoral dissertation, University of California, Irvine, 2000.»

http://www.ics.uci.edu/~fielding/pubs/dissertation/rest_arch_style.htm

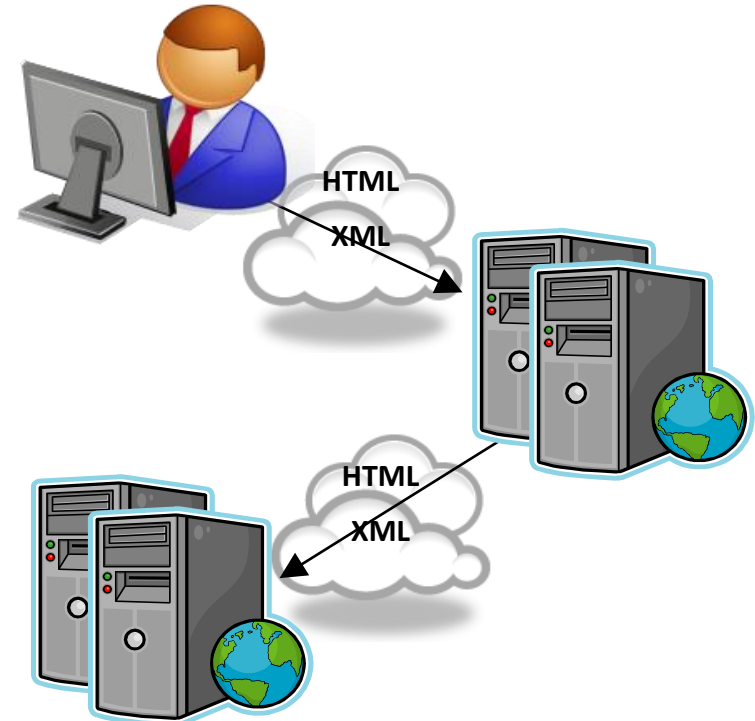
Rappel : du Web H2M au M2M



**Generation 1
Static HTML**



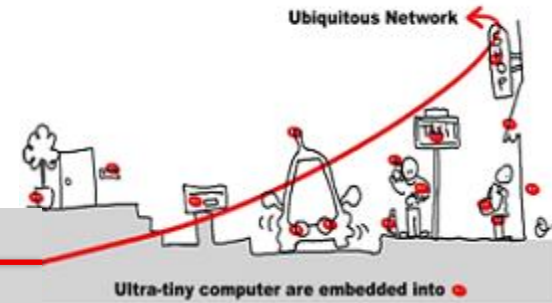
**Generation 2
Web Applications**



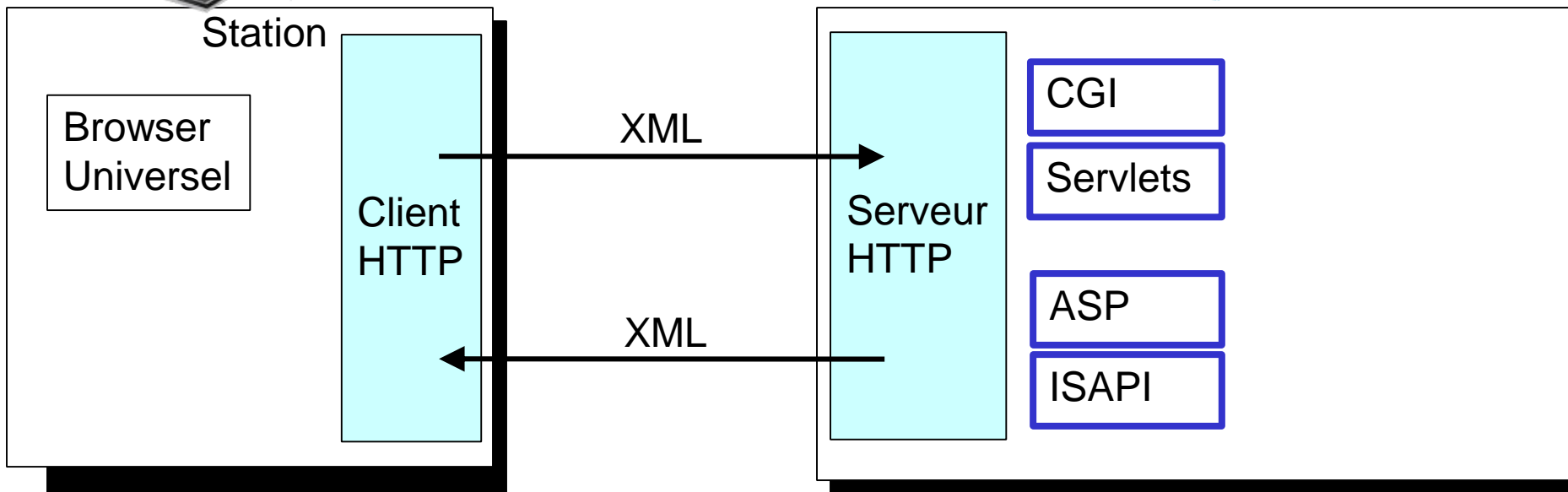
**Generation 3
Web Services**

Machine to Machine (M2M)

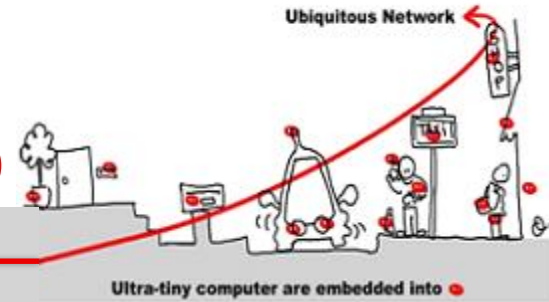
Rappel : Evolution du Web H2M au M2M



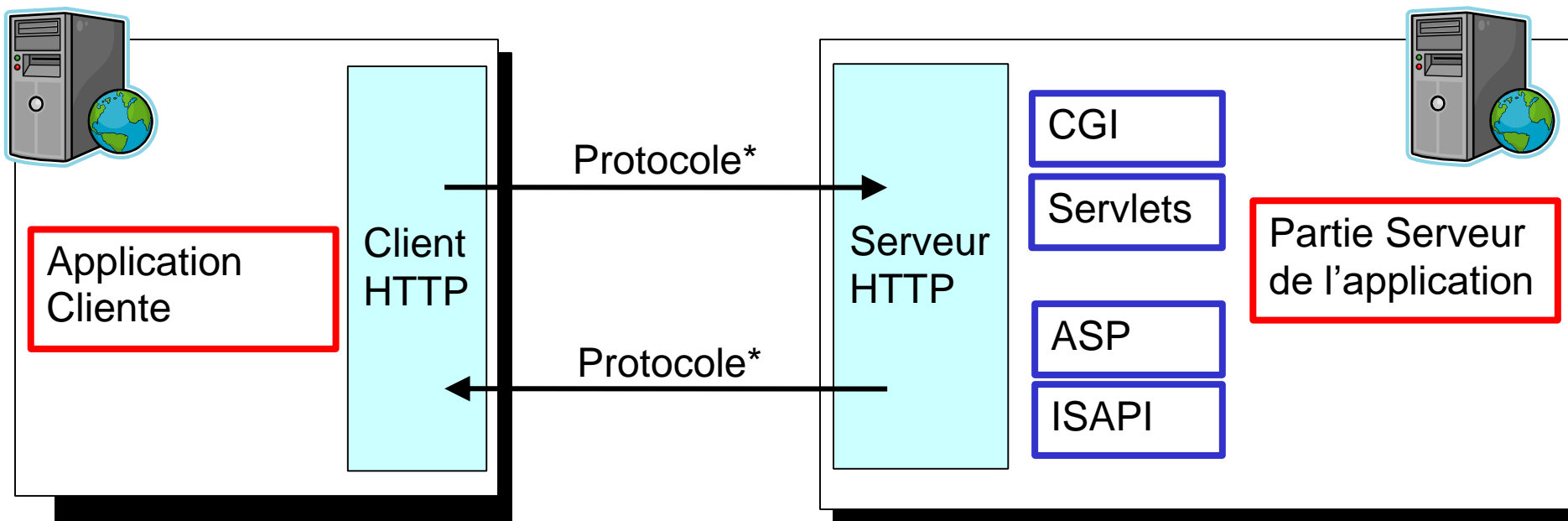
- Exemple HTTP + XML



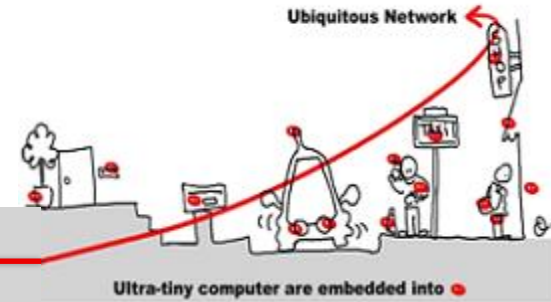
Web M2M : Middleware (intergiciel) orienté Service Web



- Gérer l'interopérabilité avec HTTP (le WEB)
- Choisir un protocole de communication client/serveur over HTTP

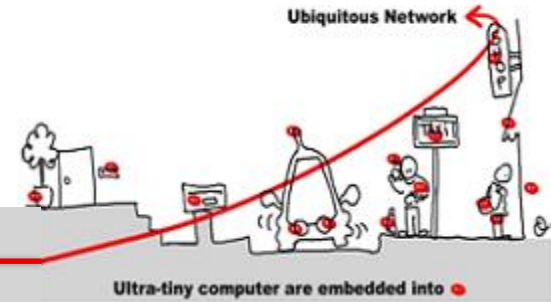


Caractéristiques des intergiciels orientés Service Web

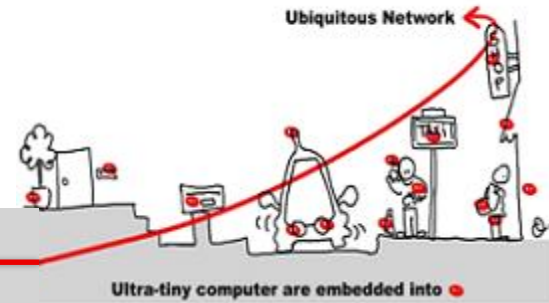


- Caractéristiques
 - Architecture de type Client / Serveur
 - Le serveur rend des services à un client
 - Réutilisable
 - Par plusieurs clients (simultanément ou pas)
 - **Indépendamment** de
 - la plate-forme (UNIX, Windows, ...)
 - du langage pour l'implémentation (VB, C#, Java, ...)
 - la plate-forme de développement sous-jacente (.NET, J2EE, Axis...)

Pour quoi faire ?

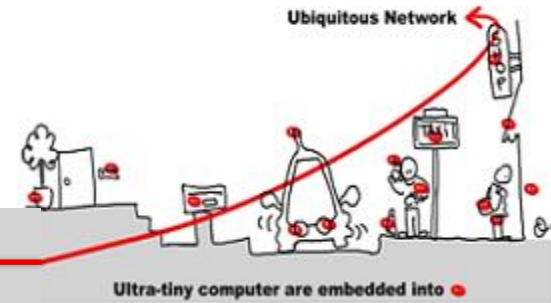


- Les Services Web permettent d'interconnecter :
 - Différentes entreprises
 - Différentes applications
 - Différents clients
 - Différents matériels
- Utilisé dans différents cadres:
 - B2B (Business To Business)
 - EAI (Enterprise Application Integration)
 - ...



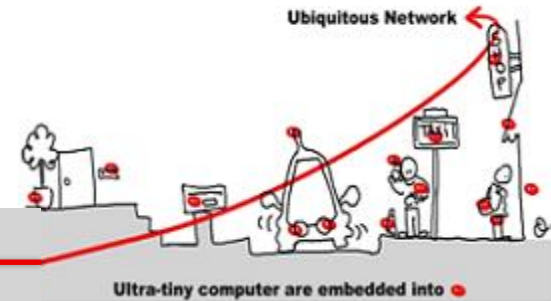
Architectures REST et ROA

Architecture orientées ressources ou données



- Le Web aujourd'hui : extraction des ressources / ressources
- Les ressources sont identifiées par des URL
- Dénominations
 - Architectures Orientées Données (DOA)
 - Architectures Orientées Ressources (ROA)

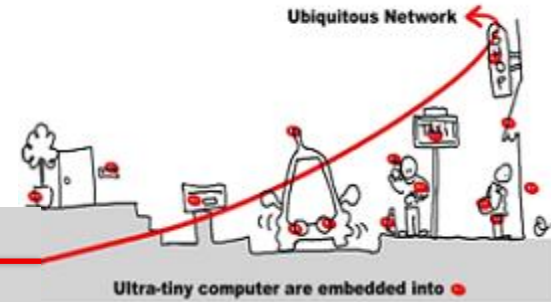
Les principes REST



- **REpresentational State Transfer**
- Style architectural pas seulement dédié aux architectures orientées services et aux communication entre machines.
- Aucune hypothèse sur les protocoles impliqués, seulement des contraintes
- Les systèmes qui suivent les principes de l'architecture REST sont souvent appelés *RESTful* et s'appuient sur le Web

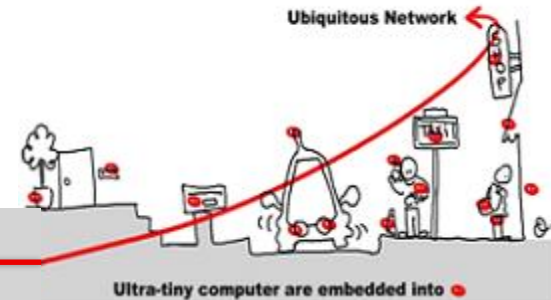
Chapitre 5 de la thèse de doctorat "Representational State Transfer (REST)". de Roy Fielding , 2000

Les principes REST ou ROA



- Ressources (Identifiant)
 - Entité identifiable dans le système (livre, agenda ...)
 - URI et donc possiblement URL
 - Une URI identifie une seule Ressource
 - Une Ressource peut avoir plusieurs URI
 - Exemple :
 - Emploi du temps de tigli : /edt/prof/tigli/lundi
- Méthodes (Verbes)
 - Quatre opérations de base « CRUD » : Create (créer), Retrieve (lire), Update (mettre à jour), Delete (Supprimer)
 - Exemple méthodes HTTP : GET, POST, PUT, DELETE
 - Déjà adaptées à la manipulation de Ressources
- Représentation (Vue de l'état)
 - Informations transférées entre client et serveur
 - Exemple : XML, JSON, XHTML, CSV

Exemple RESTful/XML



- Exemple de message HTTP RESTful

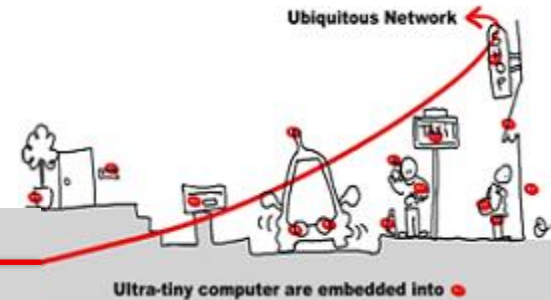
```
POST http://MyService/Person/  
Host: MyService  
Content-Type: text/xml; charset=utf-8  
Content-Length: 123  
<?xml version="1.0" encoding="utf-8"?>
```

*HTTP Header
Commande POST*

```
<Person>  
  <ID>1</ID>  
  <Name>M Vaqqas</Name>  
  <Email>m.vaqqas@gmail.com</Email>  
  <Country>India</Country>  
</Person>
```

*HTTP Body
XML representation
of a resource « Person »*

Exemple RESTFul/JSON



- Exemple de message HTTP RESTFul

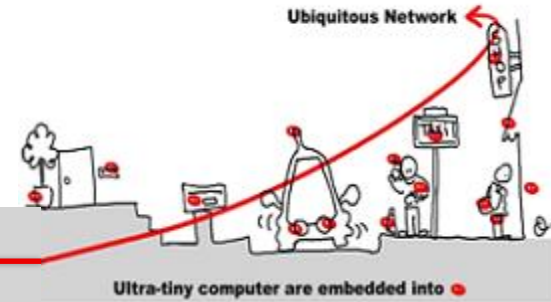
```
POST http://MyService/Person/
Host: MyService
Content-Type: text/xml; charset=utf-8
Content-Length: 123
<?xml version="1.0" encoding="utf-8"?>
```

*HTTP Header
Commande POST*

```
{
  "ID": "1",
  "Name": "M Vaqqas",
  "Email": "m.vaqqas@gmail.com",
  "Country": "India"
}
```

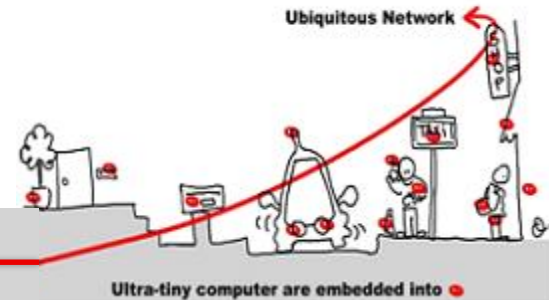
*HTTP Body
JSON representation
of a resource*

REST et invocation de méthode



- Chaque demande REST contient une URL, de sorte que le serveur sait quelle ressource vous souhaitez accéder, mais il peut aussi contenir une méthode.
- Une méthode décrit alors quoi faire avec cette ressource.
- Mais ce concept «méthode» n'est pas utilisé très souvent car en marge d'une approche ROA
- Habituellement, on utilise une URL comme un lien vers des données récupérées via la méthode GET, et modifiées (délétions, insertions, mises à jour) via la méthode POST

Cycle de Vie REST



- Plus récemment REST s'est doté d'un langage de description de service : WADL (Web Application Description Language)
- Soumis en 2009 au W3C, il n'est toujours pas standardisé
- Le WADL est un format de fichier basé sur XML qui permet de décrire des applications REST.
- Cette spécification se heurte néanmoins à **la spécification WSDL 2.0, qui elle aussi permet la description de web services REST.**
- De plus, WADL est encore très mal supporté par l'ensemble des frameworks existants ce qui limite son utilisation.