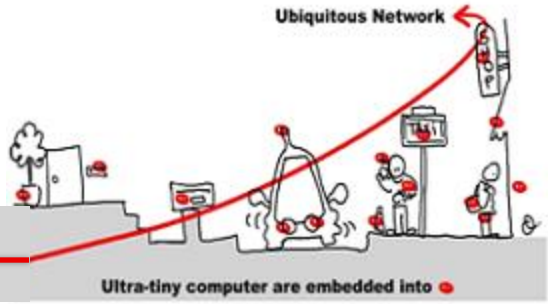
Service oriented Middleware (SOM)

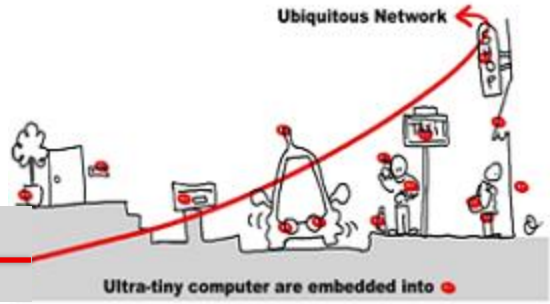


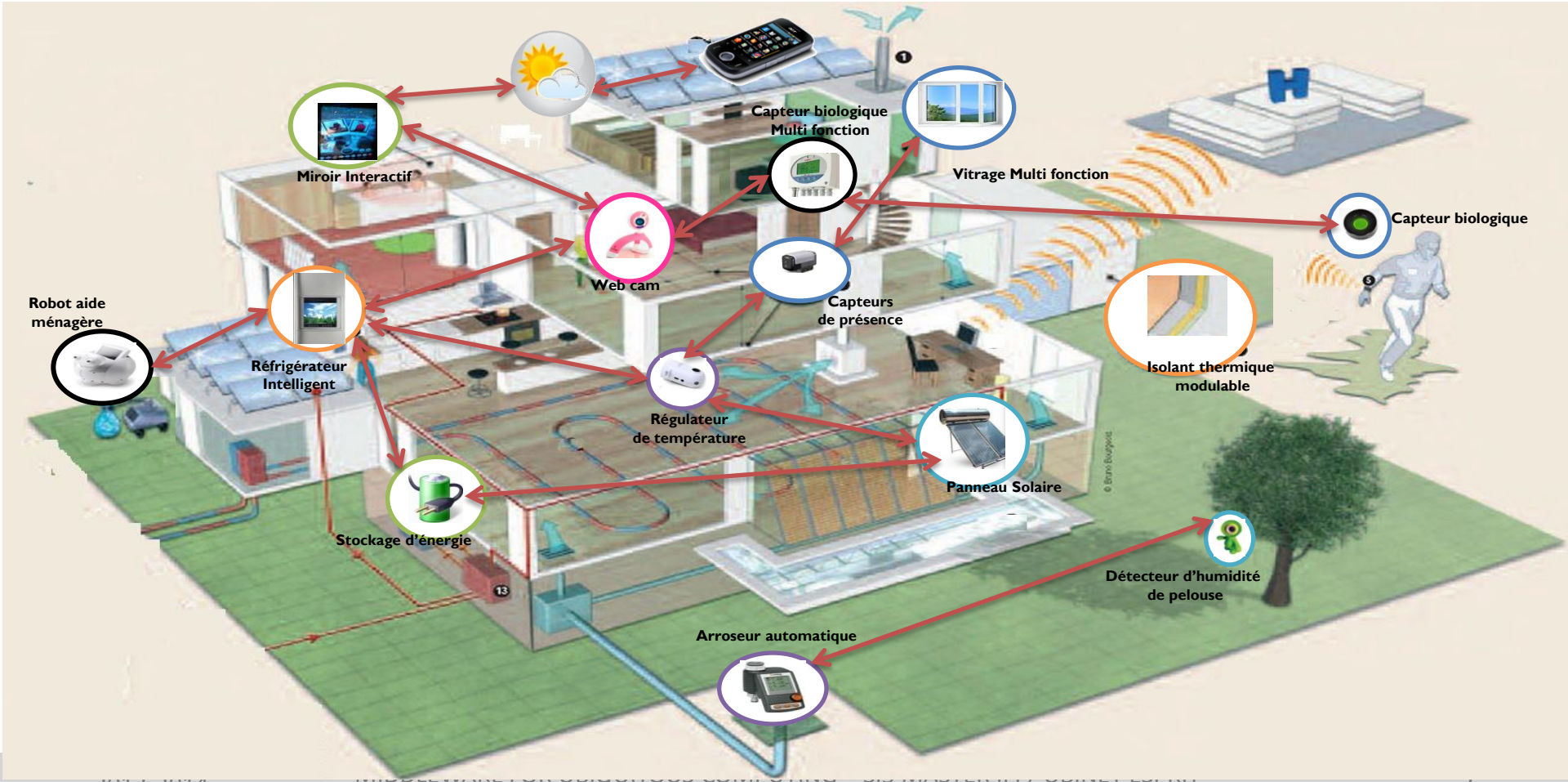
***[Issarny 11] Journal of Internet Services and Applications**, July 2011, Volume 2, Issue 1, pp 23-45, **Service-oriented middleware for the Future Internet: state of the art and research directions**, Valérie Issarny, Nikolaos Georgantas, Sara Hachem, Apostolos Zarras, Panos Vassiliadist, Marco Autili, Marco Aurélio Gerosa, Amira Ben Hamida*

Lecturer : Ass. Prof. Jean-Yves Tigli
http://www.tigli.fr
at Polytech of Nice - Sophia Antipolis University

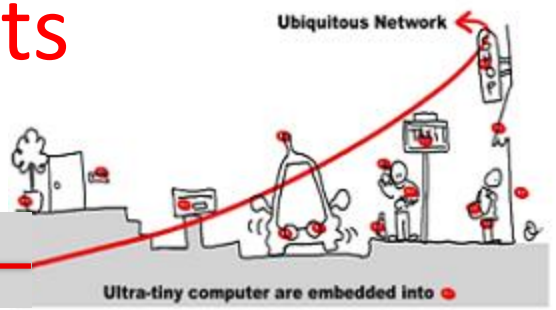
Email : tigli@polytech.unice.fr

Reminder : UbiComp Challenges for Service Continuity





Reminder : UbiComp Requirements for Service Continuity

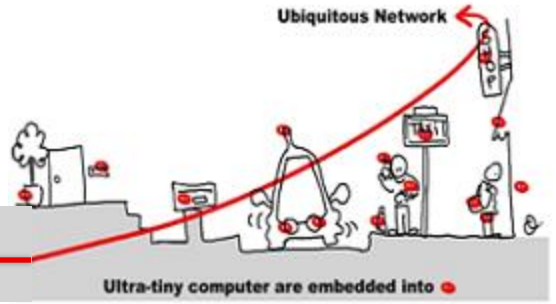


- Dynamic changing of available devices
- Heterogeneity of available devices
- Spontaneous Communication from Devices/Services to Application

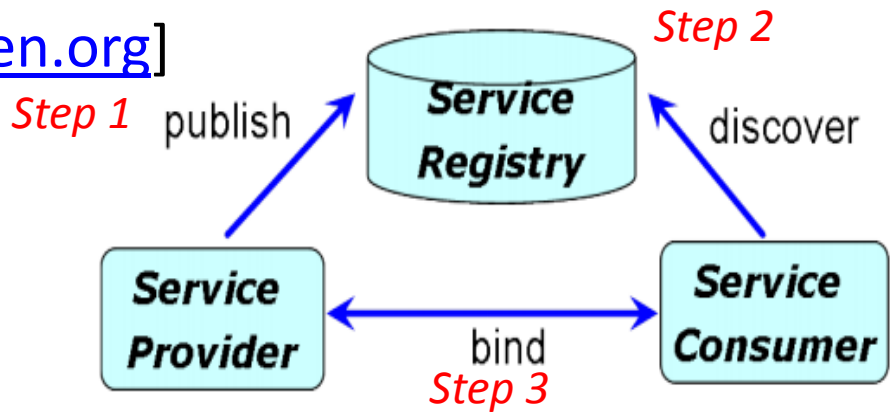
- Security, Privacy & Trust

- Mobility
- Context Awareness
- Self Adaptation and Autonomy, Reactivity with separation of concerns for multiple domains

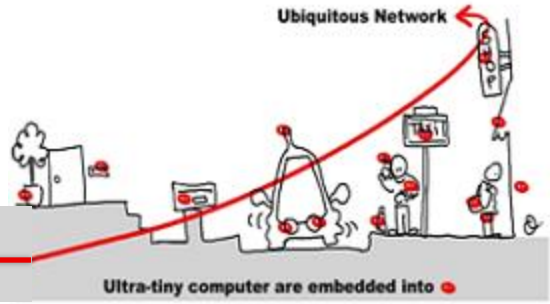
Service, for what ?



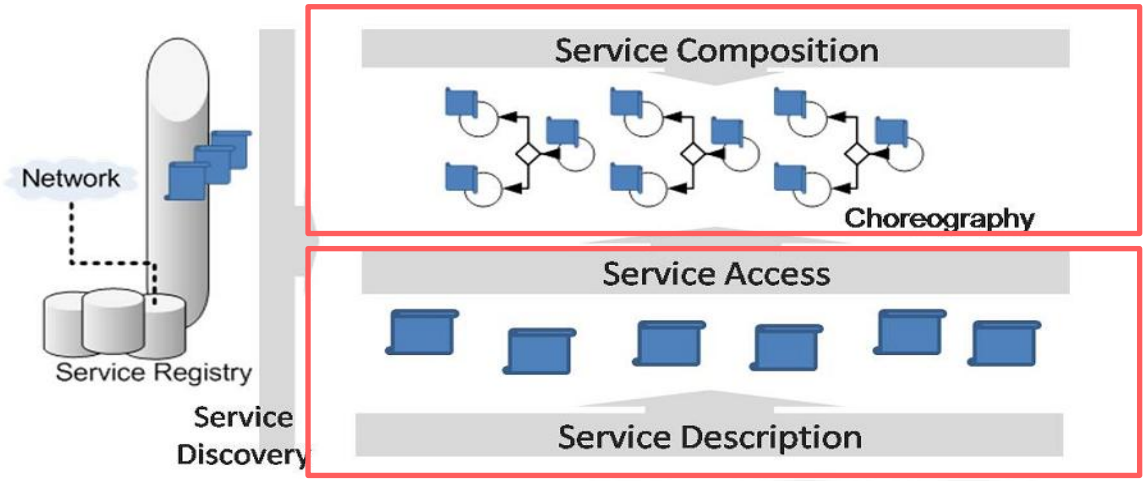
- Service oriented interaction pattern to provide a set of related software functionalities that can be reused for different purposes
- Services are unassociated, loosely coupled units of functionality that are self-contained
- Dynamic discovery
- [OASIS, https://www.oasis-open.org]



Service-oriented computing in the Future Internet *[Issarny 11]*



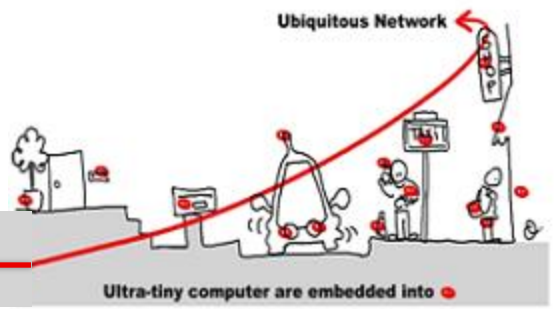
Lecture 3



Lecture 2

***[Issarny 11] Journal of Internet Services and Applications**, July 2011, Volume 2, Issue 1, pp 23-45, **Service-oriented middleware for the Future Internet: state of the art and research directions**, Valérie Issarny, Nikolaos Georgantas, Sara Hachem, Apostolos Zarras, Panos Vassiliadist, Marco Autili, Marco Aurélio Gerosa, Amira Ben Hamida*

Why SOM emerge as the solution Ubiquituous Computing *[Issarny 11]* ?

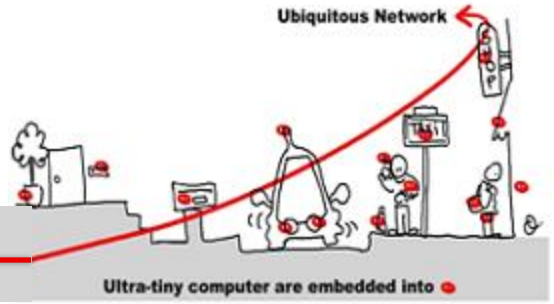


Essential functionalities of Service-Oriented Middleware :

- service description,
- access,
- discovery and
- composition

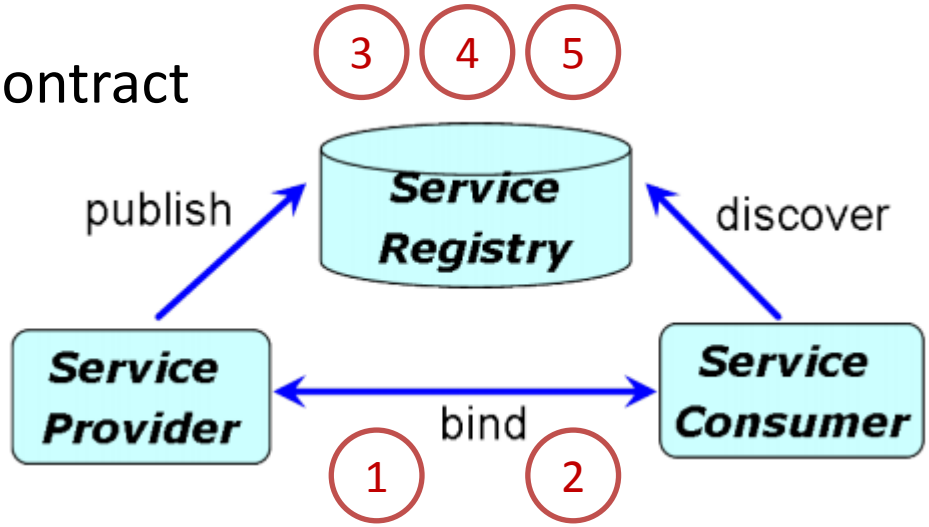
SOM functionalities	State of the art
Description	Web services, Semantic Web, OWL
Discovery	Service registries and distributed, hybrid service discovery protocols
Access	ESB paradigm for heterogeneous Service-oriented communication SOC technology/middleware integration
Composition	Orchestration/Choreography based composition of services and related BPEL engines, Dynamic composition and adaptation

Services Features

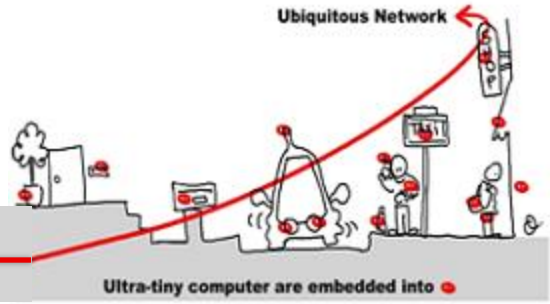


Service oriented interaction pattern

1. Communication patterns and PortType
2. Adressing and Endpoint
3. Binding
4. Interface Description and Contract
5. Semantic Description



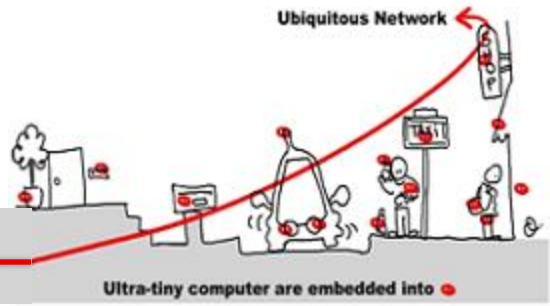
Survey



Approaches and Models	Main Features
REST	Ressource oriented Web Service
WS-* (SOAP)	RPC oriented Web Service
ABC (WCF)	Generic Service
UPnP DPWS	Web Service for Device, eventing communications, Appearance and Disappearance Management

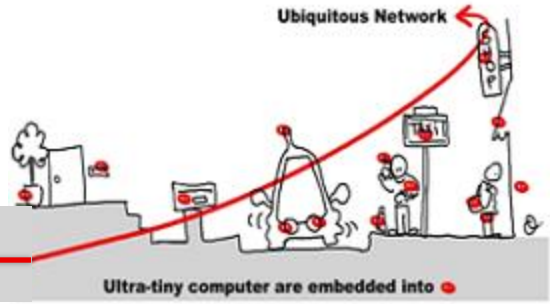
Tutorial UPnP
Practical Course

REST : Representational State Transfer



- Addressing resources:
  - REST uses plain URIs (actually URLs) to address and name resources.
- Access to resources:
  - Unlike RPC-WS where the access method (CRUD) is mapped to and smeared over SOAP messages, REST uses the available HTTP methods as a resource interface:
  - Create (C) →HTTP PUT
  - Read (R) → HTTP GET
  - Update (U) → HTTP POST
  - Delete (D) → HTTP DELETE

REST ‘protocol’

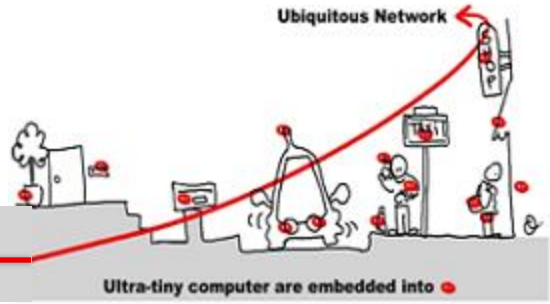


- Resource representations:
  - REST uses standard resource representations like HTML, XML, JSON, GIF, JPEG. Commonly used representations are XML and JSON (preferable to XML if the data needs to be transferred in a more compact form).

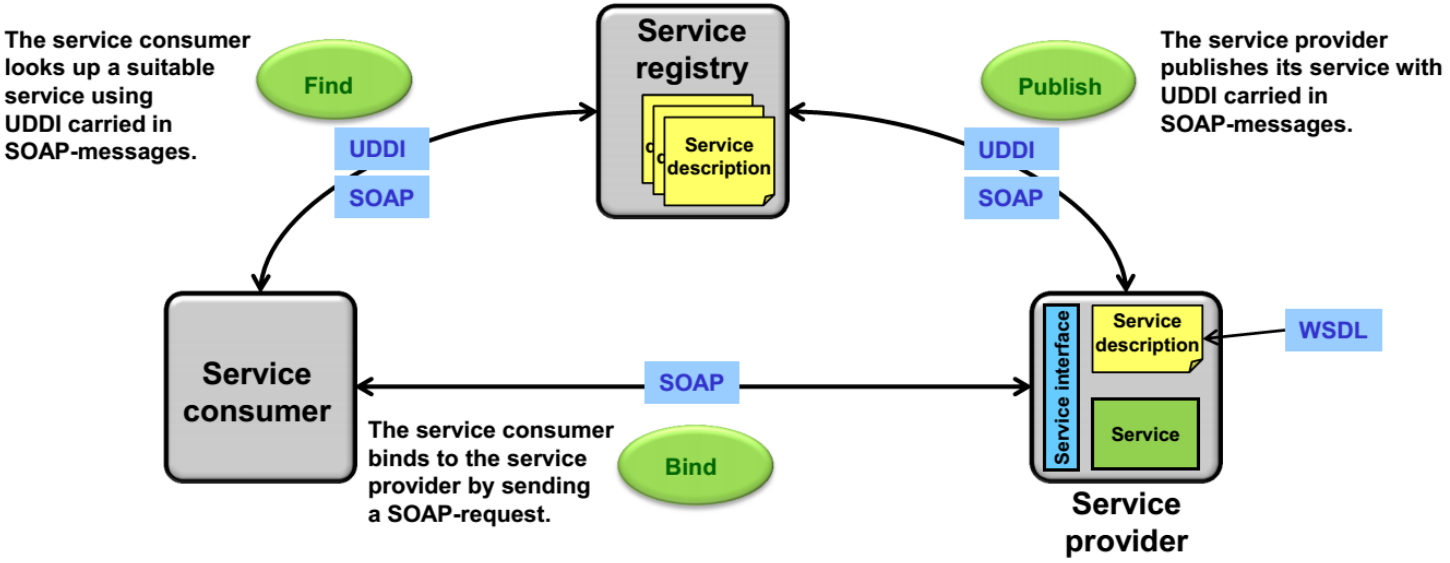
- Media types:
  - REST uses the HTTP header Content-type (MIME types like text/html, text/plain, text/xml, text/javascript for JSON etc.) to indicate the encoding of the resource.

- State:
  - Application state is to be maintained on the client. The server does not have to maintain a state variable for each client (this improves scalability). Resource state (resource creation, update, deletion), however, is maintained on the server.

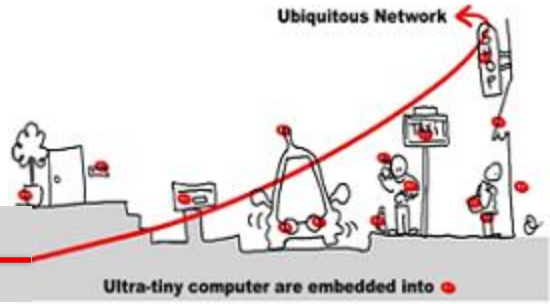
WS-*architecture



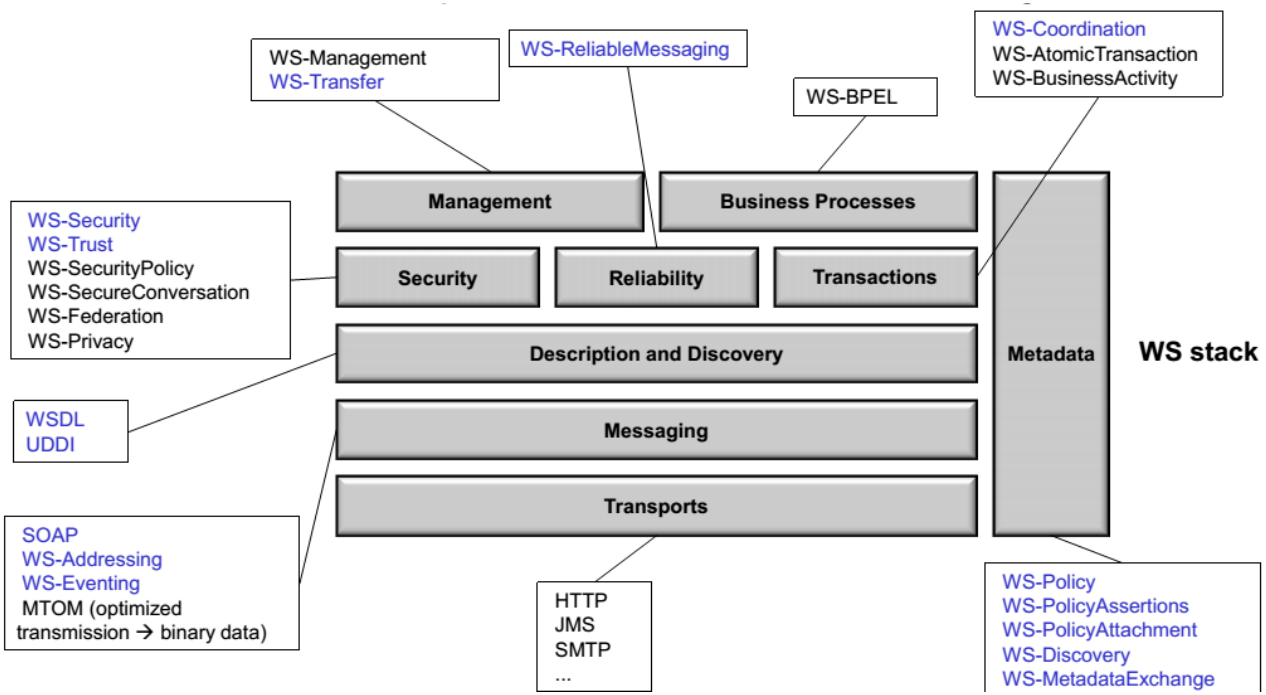
- SOAP+WSDL+UDDI defines a general model for a web service architecture.
  - SOAP: Simple Object Access Protocol
  - WSDL: Web Service Description Language
  - UDDI: Universal Description and Discovery Protocol
  - Service consumer: User of a service
  - Service provider: Entity that implements a service (=server)
  - Service registry : Central place where available services are listed and advertised for lookup



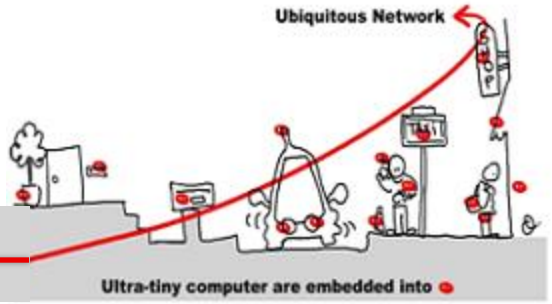
WS-* Models



- Stack of WS-standards
- The W3C and OASIS WS-stack provide a framework / toolbox for constructing web service architectures



SOAP-WS versus REST Model



SOAP-WS:

- Rather complete (there is a WS-standard for almost every aspect / problem).
- Modular (take what you need and compose your web service architecture).
- Complex (too many different WS-standards with dependencies to each other, difficult to find a common base that is supported by all participants).
- Performance penalty due to chatty protocols with large overhead (SOAP).

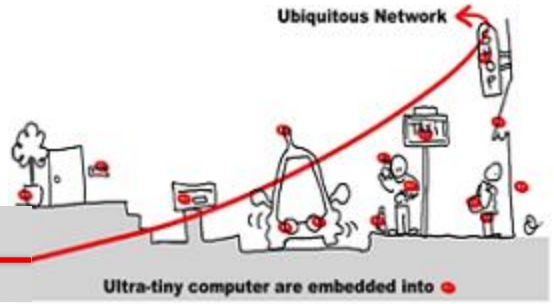
Applicability / suitability:
Enterprise SOA-architecture which requires security, orchestration, management etc.

REST-WS:

- Simple.
- Fits the bill for most applications?
- No standard, semantics of service mostly described in human readable form, not machine processable without description language (e.g. WADL or WSDL 2.0).
- Too simple (missing functionality for advanced services which require coordination etc.).

Applicability:
Simple and isolated access (read) to data / resources.

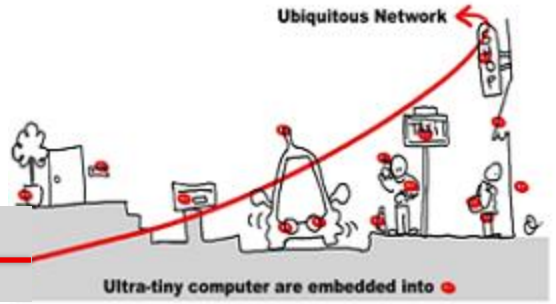
ABC Model : a generic one



- The core concept of WCF (Windows Communication Foundation) is a service that is provided on an endpoint and accessible over the network through a transport protocol.

- Thus a WCF service is defined by ABC:
- A = Address:
  - Where is the service available (the endpoint's URI in case of a web service).
- B = Binding:
  - How can the service be accessed (what transport protocol is used).
- C = Contract:
  - What does the service interface look like (operations, data-types).

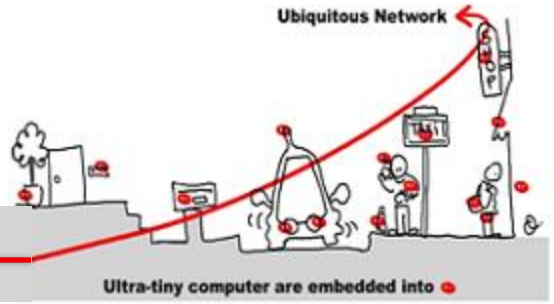
ABC versus WSDL Model



- The mapping of the ABCs to WSDL:

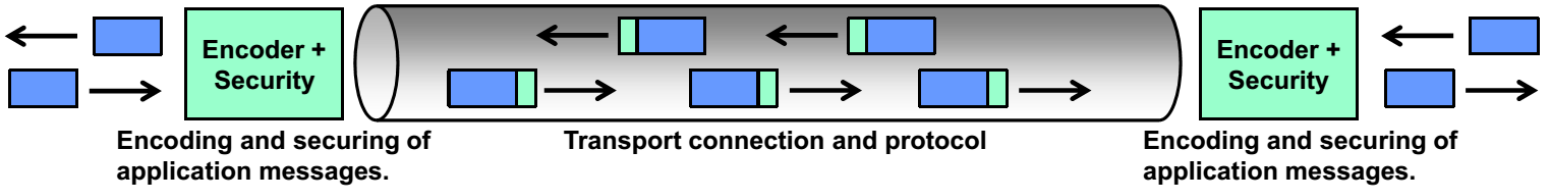
ABC concepts	Question	WSDL element
A (Address)	Where	<service> including element <endpoint>
B (Binding)	How	<binding>
C (Contract)	What	<types> <interface>

Binding as B in ABC Model

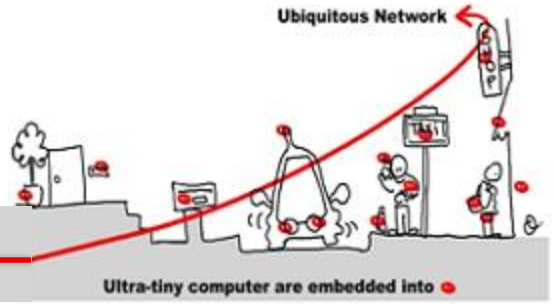


- The binding defines how a web service endpoint is accessed.
- A binding contains the following elements:

1. Transport protocol:
   - Underlying transport protocol to use when interacting with the web service.
   - Examples: TCP, HTTP, MSMQ.
2. Message encoding:
   - Definition of the message encoding.
   - Examples: Text/XML (SOAP), binary, MTOM (Message Transfer Optimized Mechanism).
3. Security / reliability settings:
   - Message security settings (e.g. encryption and authentication of message).
   - Transport security (e.g. encryption of transport connection).



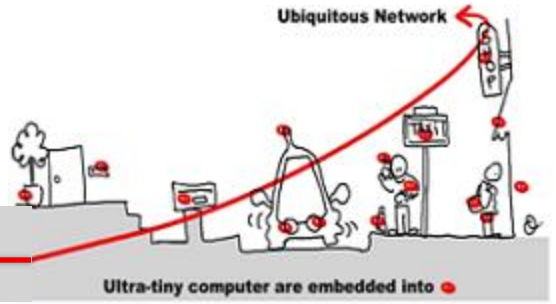
Binding as B in ABC Model



- More than WEB for Binding …
- More than WS-* and REST

Binding	Interoperability	Security	Session	Transactions	Duplex	Encoding
BasicHttpBinding	WS-I Basic Profile	N, T, M, m	N	N	No	Text, MTOM
WSHttpBinding	WS-* standards	T, M, m	N, RS, SS	N, Yes	No	Text, MTOM
WSDualHttpBinding	WS-* standards	M, m	RS, SS	N, Yes	Yes	Text, MTOM
WSFederationHttpBinding	WS-Federation	N, M, m	RS, SS	N, Yes	No	Text, MTOM
NetTcpBinding	.NET	T, M, m, N	TS, RS, SS	N, Yes	Yes	Binary
NetNamedPipeBinding	.NET	T, N	N, TS	N, Yes	Yes	Binary
NetMsmqBinding	.NET (WCF)	M, T, N	N, TS	N, Yes	No	Binary
NetPeerTcpBinding	.NET	T	N	N	Yes	N/A
MsmqIntegrationBinding	MSMQ	T	N	N, Yes	No	MSMQ
BasicHttpContextBinding	WS-I Basic Profile	N, T, M, m	N	N	No	Text, MTOM
NetTcpContextBinding	.NET	N, T, M, m	T, RS, SS	N, Yes	Yes	Binary
WSHttpContextBinding	WS-* standards	T, M, m	N, RS, SS	N, Yes	No	Text, MTOM
WebHttpBinding	HTTP (REST)	N	N	N	No	POX

Adress as A in ABC Model



- The WCF address defines where a web service (endpoint) is accessible.
- WCF models an address as an endpoint reference (EPR) as per the WS-Addressing standard
- Example endpoint address URI:

http://localhost:8000/HSZ-TWSMW/DateTimeService

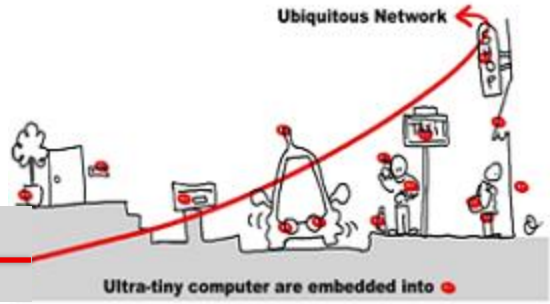
Scheme / protocol

Machine address (IP or DNS)

Port #

Path under which the service is available

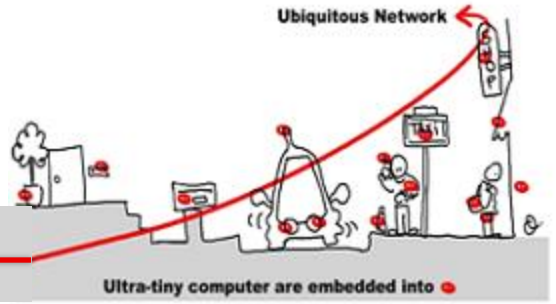
Contract as C in ABC Model



- WCF interfaces are called “contracts” (both client and server must comply with the contract).
- Contracts describe operations, data structures and messages.

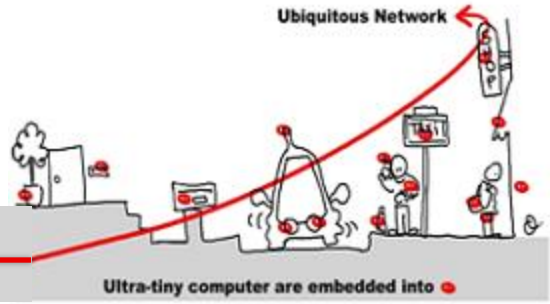
- A service contract defines:
  - Grouping of operations in a service → .Net attribute [ServiceContract]
  - Signature of operations → .Net attribute [OperationContract]
  - Data types of operations → .Net attribute [DataContract]

Example WCF Contract



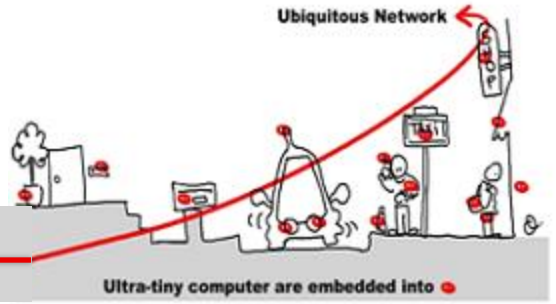

```
[ServiceContract]
public interface IService1
{
    // TODO: Add your service operations here
    [OperationContract]
   CompositeType GetDataUsingDataContract(CompositeType composite);
}

// Use a data contract as illustrated in the sample below to add composite
      types to service operations.

[DataContract]
public class CompositeType
{
   bool boolValue = true;
   string stringValue = "Hello ";

   [DataMember]
   public bool BoolValue
   {
      get { return boolValue; }
      set { boolValue = value; }
   }
}
```

ABC Model and WCF



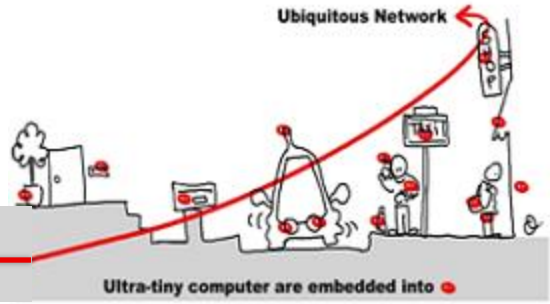
Application
hosted in
Host
• Self-hosted
• Service host
has
Instancing model
• Single
• Session
• Per call
has
has
Message contract
implements
WCF client
WCF service
Service contract
contains
Operation contract
1
has
*
1
has
*
uses
Endpoint
Endpoint
Appl. endpoint
corresponds to
Data contract
uses
Message
1
1
Infrastructure endpoint
Provides service metadata
has
has
has
Address as URI
*
*
Communication stack
Channel
Security protocol
Text/ XML
MTOM
Binary
Channel
Message encoding
has
Message exchange pattern
Channel
Transport protocol
HTTP
TCP
MSMQ
Binding elements
Request reply
One-way
Duplex
WSDL
ABC elements (blue) are described by a WSDL file.

Service semantics

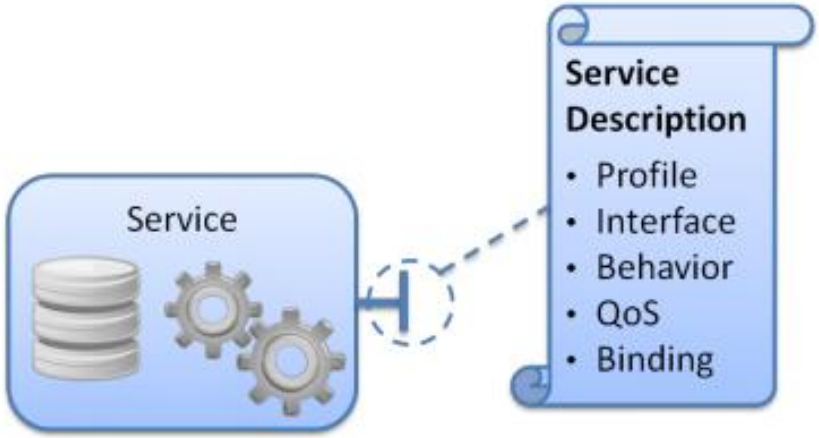


- Service semantics are made explicit by reference to a structured vocabulary of terms (ontology) representing a specific area of knowledge
- Ontology languages support formal description and machine reasoning upon ontologies
- Web Ontology Language (OWL) is the standard established by W3C (http://www.w3.org/2004/OWL/)
  - OWL-S: http://www.w3.org/Submission/OWL-S/
  - WSMO: http://www.w3.org/Submission/2005/06/
  - SWSF: http://www.w3.org/Submission/SWSF/
  - WSDL-S: http://www.w3.org/Submission/WSDL-S/

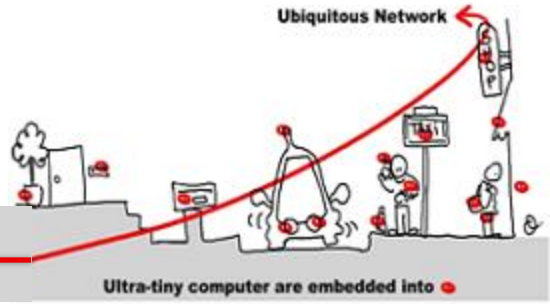
Most common elements of service description



- Service Profile provides a high-level business description of a service
- Service Behavior specifies the observable supported execution patterns (often called conversations)
- Service Interface specifies the set of observable lower-level (with respect to the functionalities in the Service Profile) atomic operations

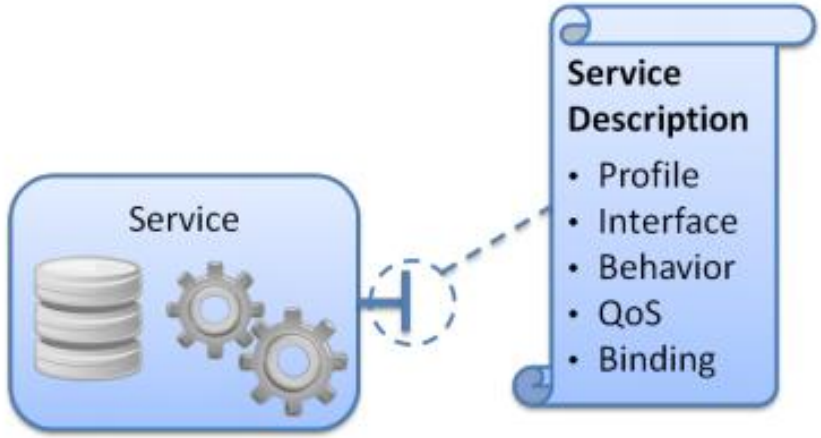


Most common elements of service description

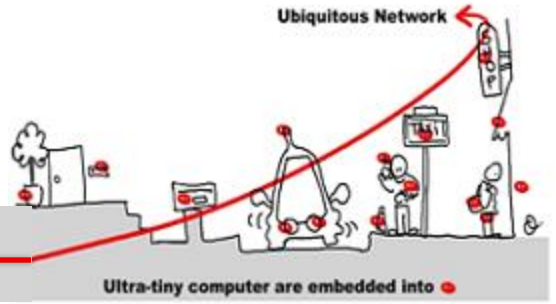


- Service QoS concerns non-functional properties of the service, such as reliability, performance, security, privacy, trust
- Service Binding specifies the underlying communication middleware on which the service is deployed

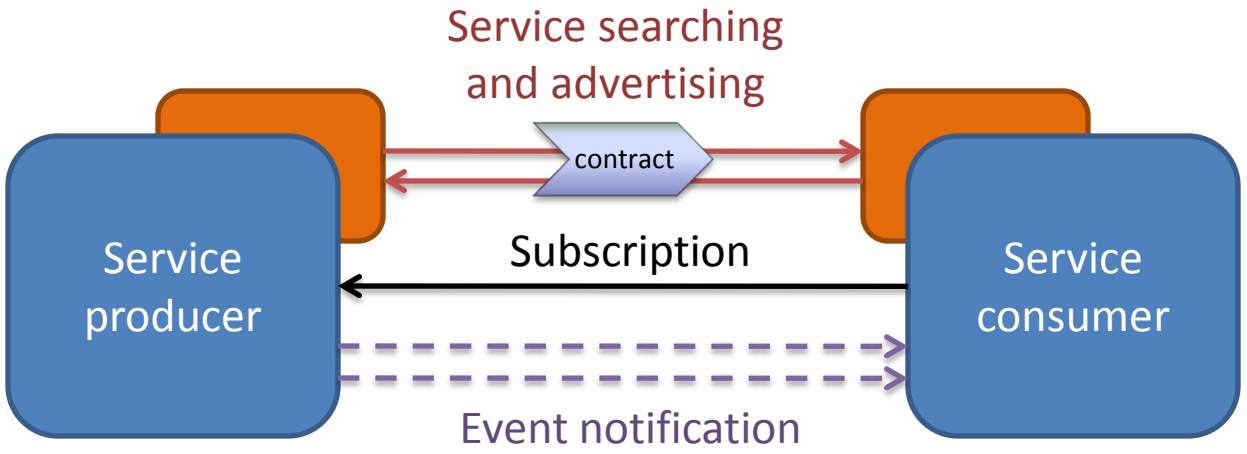
Common bindings are SOAP/HTTP/TCP for WS- Web Services and HTTP/TCP for RESTful Web Services*



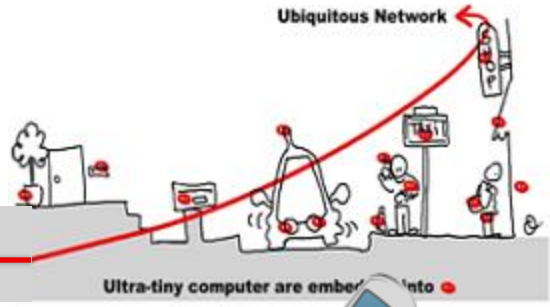
Web Service for Device



- Distributed dynamic Research and Discovery
  - Appearance and Disappearance management
  - Allow contextual research and discovery
- Eventing interaction model
  - Based on publish/subscribe design pattern
  - Asynchronous messaging (based on push mode)

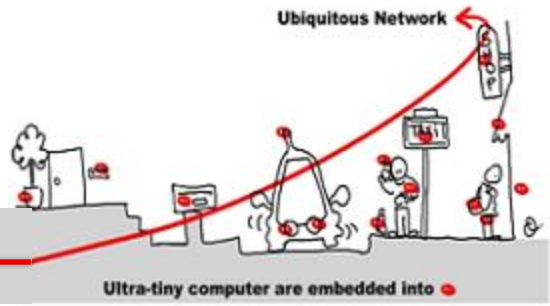


Example : Universal Plug and Play



- Control Point
  - The client which discover and control UPnP servers
- Device
  - The server (receive actions)
- A physical device can be twice (CP and Device)

Example of UPnP Device Communications



Device

Addressing

DHCP

Auto-IP

Description

Discovery

Device Advertisements over SSDP channel

Discovery Search

Action Request

Action Response

Control

Control Request

Control Response

Events generated by the device

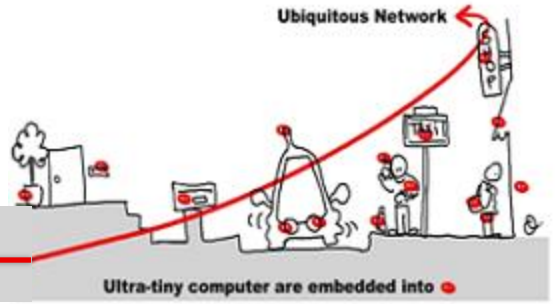
Eventing

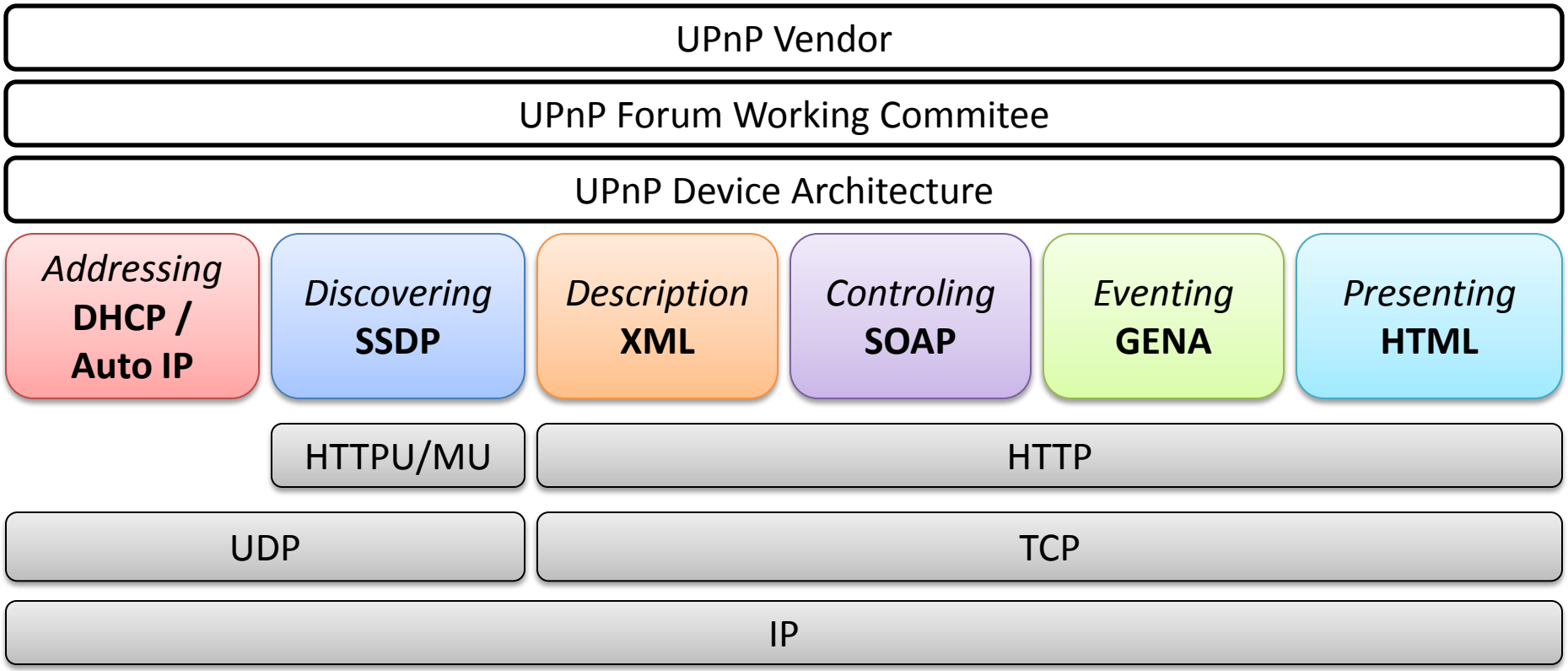
Event subscription

Events notification

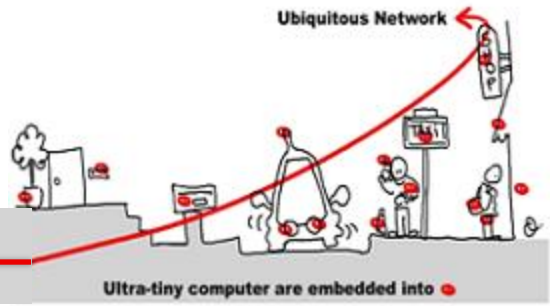
Network

UPnP Stack and Protocols



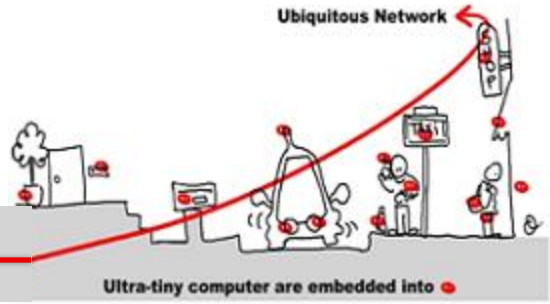


Conclusion : SOM functionalities Research directions

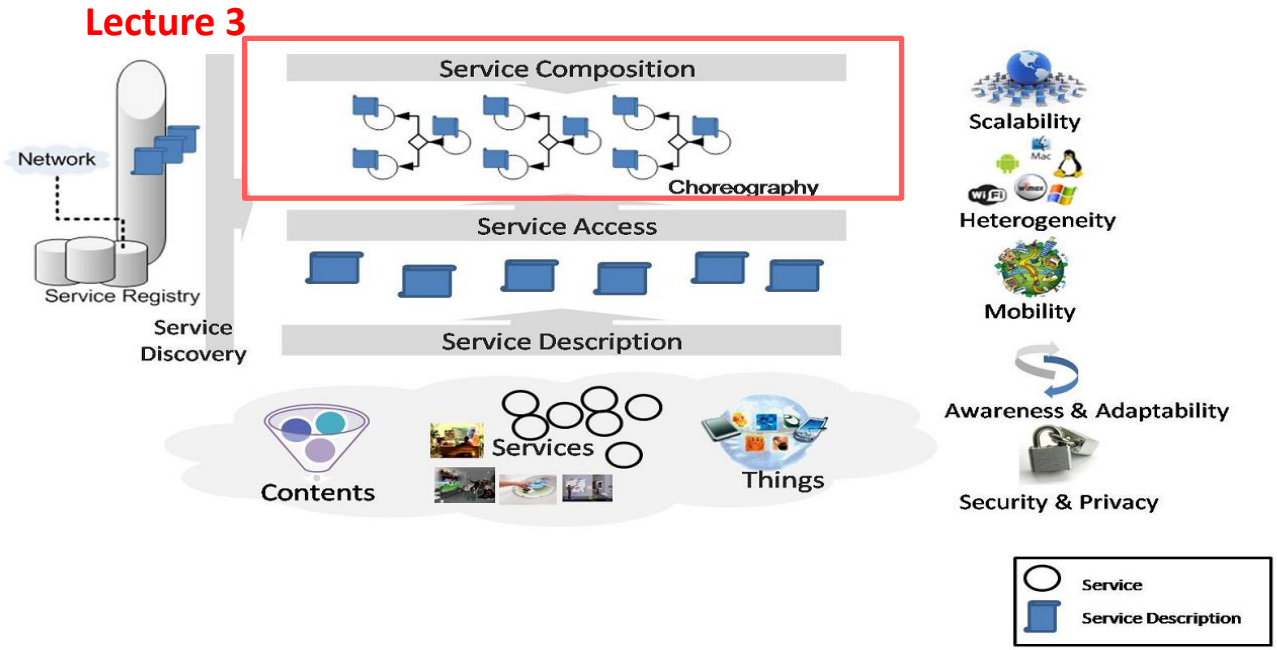


Description	Investigate trade-off between rich service descriptions and related processing complexity.
Discovery	Investigate service discovery protocols for the ultra large scale, heterogeneous and mobile Future Internet, while controlling the quality of the information and providing security, privacy, trust guarantees.
Access	Exploit high-performance, resource-on-demand computing technologies to cope with scale.
	Handle heterogeneity, mobility, security, privacy, trust in open, dynamic and aware settings.
Composition	Enable scalable and adaptive choreography modeling and execution for the highly heterogeneous and mobile Future Internet, while guaranteeing security and privacy properties.

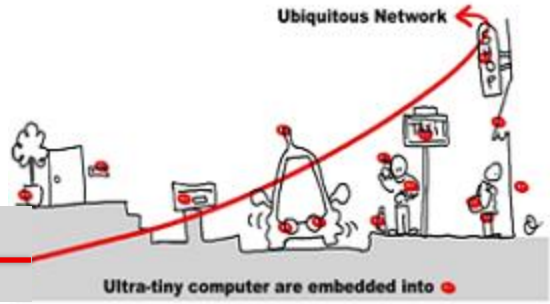
Conclusion : Next lecture …



Service oriented Middleware and Composition



Appendice 1 : Example ABC model and REST/JSON service in WCF

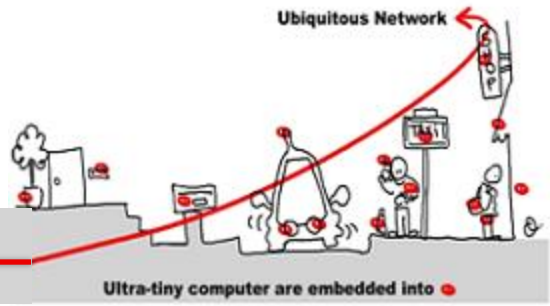


Collapse | Copy Code

```
<?xml version="1.0"?>
<configuration>
  <system.serviceModel>
        <services>
          <service name="WcfJsonRestService.Service1">
                <endpoint address="http://localhost:8732/service1"
                                  binding="webHttpBinding"
                                  contract="WcfJsonRestService.IService1"/>
          </service>
        </services>
        <behaviors>
          <endpointBehaviors>
                <behavior>
                  <webHttp />
                </behavior>
          </endpointBehaviors>
        </behaviors>
  </system.serviceModel>
  <startup>
        <supportedRuntime version="v4.0" sku=".NETFramework,Version=v4.0"/>
  </startup>
</configuration>
```

http://www.codeproject.com/Articles/167159/How-to-create-a-JSON-WCF-RESTful-Service-in-60-sec

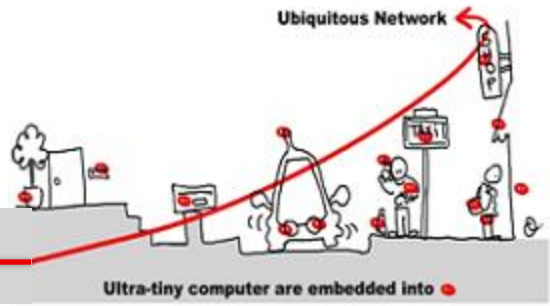
Appendice 1 : Example Contract for REST/JSON in WCF




```
using System.ServiceModel;

namespace WcfJsonRestService
{
    [ServiceContract]
    public interface IService1
    {
        [OperationContract]
        Person GetData(string id);
    }
}
```

http://www.codeproject.com/Articles/167159/How-to-create-a-JSON-WCF-RESTful-Service-in-60-sec

Appendice 1 : Example Interface for REST/JSON in WCF




```
using System;
using System.ServiceModel.Web;

namespace WcfJsonRestService
{
    public class Service1 : IService1
    {
        [WebInvoke(Method = "GET",
                   ResponseFormat = WebMessageFormat.Json,
                   UriTemplate = "data/{id}")]
        public Person GetData(string id)
        {
            // lookup person with the requested id
            return new Person()
                       {
                           Id = Convert.ToInt32(id),
                           Name = "Leo Messi"
                       };
        }
    }

    public class Person
    {
        public int Id { get; set; }
        public string Name { get; set; }
    }
}
```

http://www.codeproject.com/Articles/167159/How-to-create-a-JSON-WCF-RESTful-Service-in-60-sec