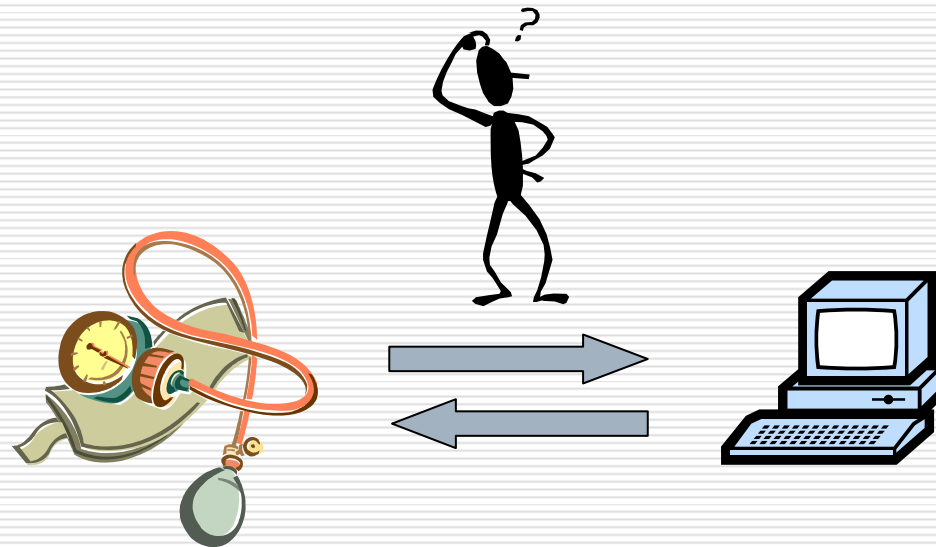


Intégration des Dispositifs Médicaux Communicants



Wael LABIDI

http://cedric.cnam.fr/~labidi_w/

Sommaire

- Introduction
 - SODA
 - Concept SOA
 - Standards
 - OSGi
 - OHF – SODA Project
 - Conclusion
-

Introduction

Dispositifs



Services

Bien-être



Urgence



Soin



Dossier
Médical



Introduction

Dispositifs



Agrégation



PC



Mobile



Set Top Box

Interface Équipement

Interface Services

Services

Bien-être



Urgence



Soin



Dossier
Médical



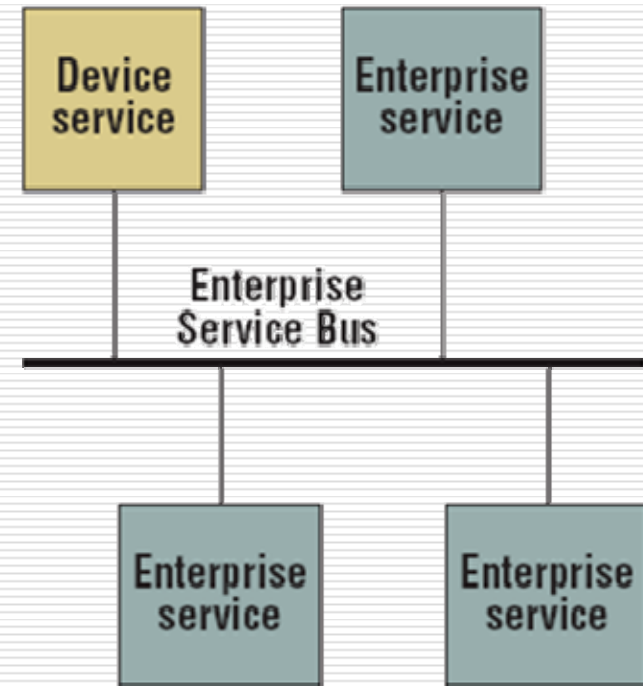
Introduction



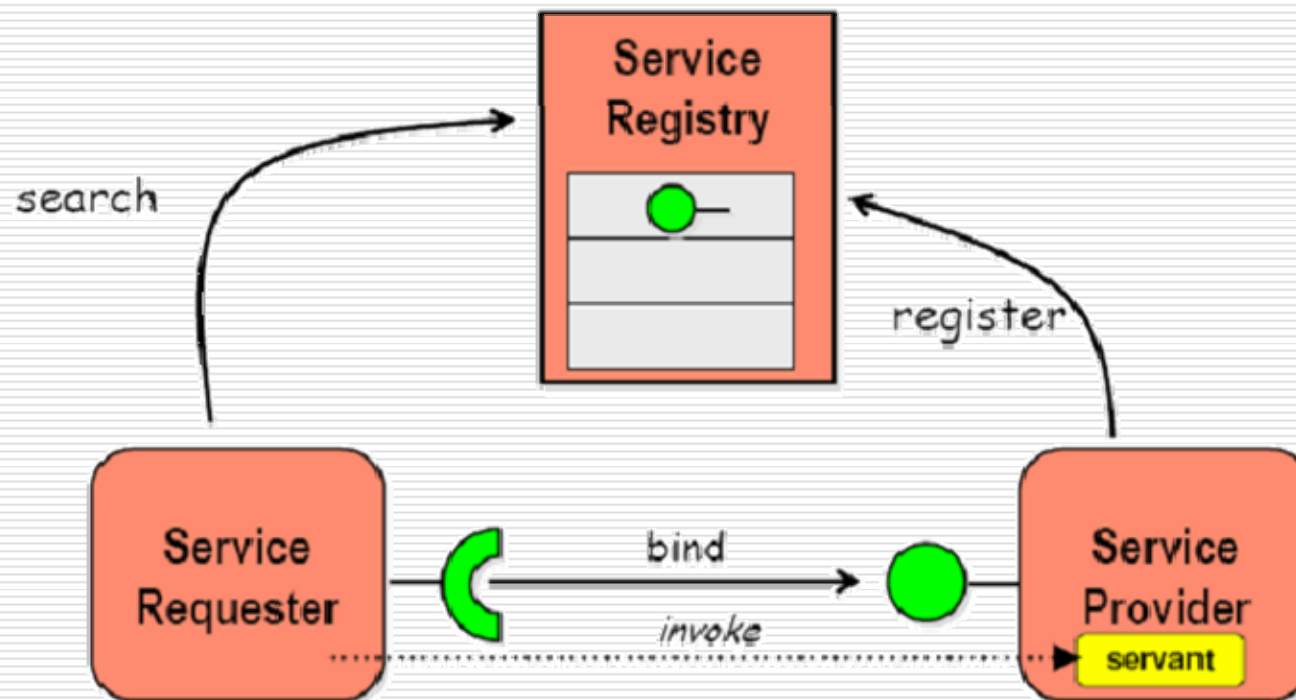
Objectif de ce cours !

SODA

- ❑ Adaptation de SOA pour les Dispositifs (Devices).
- ❑ Permettre aux développeurs d'interagir avec les dispositifs comme étant des services logiciels.
- ❑ Les services sont des composants logiciels avec des interfaces prédéfinis (Contrats).



SOA



Standards

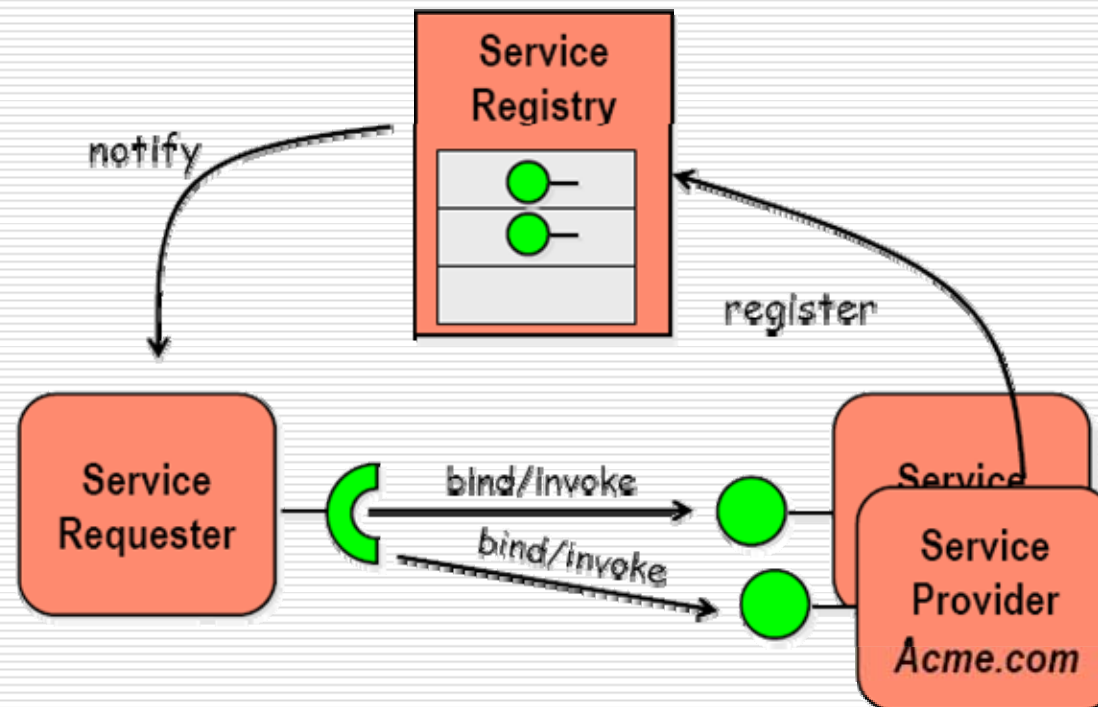
- Bluetooth, UPnP, Jini, SLP
 - Standards de découverte des services.
 - Ne définissent pas les mécanismes d'attachement aux service (Binding).
 - DPWS
 - Simplifier les standards des Services Web pour être implémentés au niveau dispositif.
 - Vise les dispositifs compatibles TCP/IP.
 - OSGi Device Access Specification
 - Encapsuler les dispositifs dans la plateforme de services dynamique OSGi (Abstraction de service).
 - Courtage local de service.
-

OSGi - Présentation

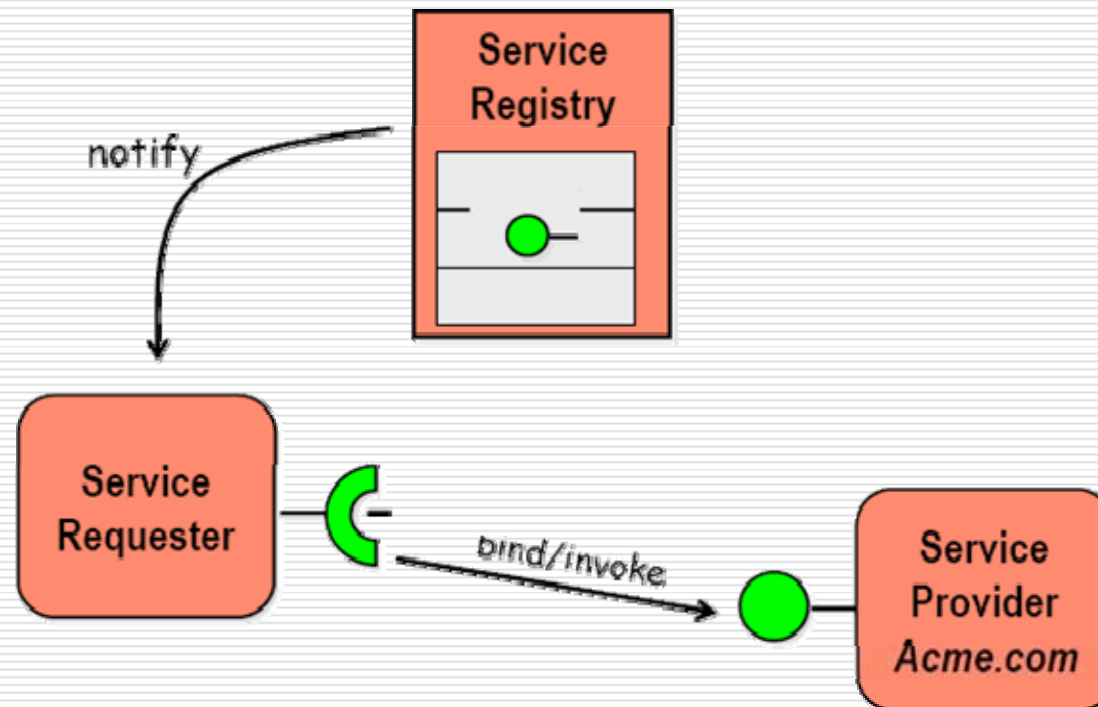
- ❑ OSGi: Open Service Gateway Initiative (Terme Obsolète).
- ❑ Standard qui définit un conteneur de services dynamiques.
- ❑ Un service est un POJO.
- ❑ Propriétés:
 - Chargement/Déchargement dynamique de modules (bundles).
 - Gestion de cycle de vie des services.
 - SOA dynamique (versioning).



OSGi – SOA Dynamique



OSGi – SOA Dynamique



OSGi – Le Framework



Level0: Environnement d'exécution (Configurations et Profiles Java 2; J2SE, CDC, CLDC, MIDP etc.)



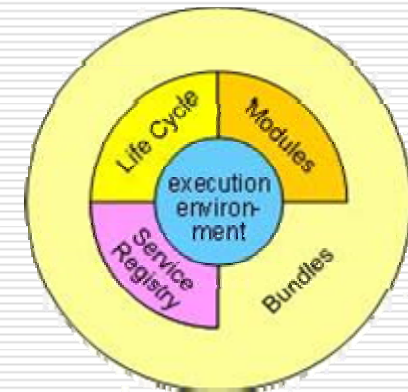
Level1: Modules (Défini les politiques de chargement «Class Loading»)



Level2: Gestion de Cycle de Vie (installer, démarrer, arrêter, mettre à jours et désinstaller des Bundles)



Level3: Registre de service (Courtage de service centralisé et déclaratif).



OSGi – Hello World Bundle

```
import org.osgi.framework.*;

public class HelloActivator implements BundleActivator {
    public void start(BundleContext context) {
        System.out.println("Hello !");
    }

    public void stop(BundleContext context) {
        System.out.println("Goodbye !");
    }
}
```

```
Manifest-Version: 1.0
Bundle-Name: HelloWorld
Bundle-SymbolicName: HelloWorld
Bundle-Version: 1.0.0
Bundle-Activator: HelloActivator
Import-Package: org.osgi.framework
```

Identification

Cycle de Vie

Dépendances

OSGi – Les services standard

- Log
 - Http
 - Device Access
 - Declarative Services
 - Event Admin
 - Service Tracker
 - Configuration Admin
 - Preferences
 - User Admin
 - Wire Admin
 - IO Connector
 - Auto Configuration
 - Application Admin
 - Monitor Admin
 - XML Parser
 - ...
-

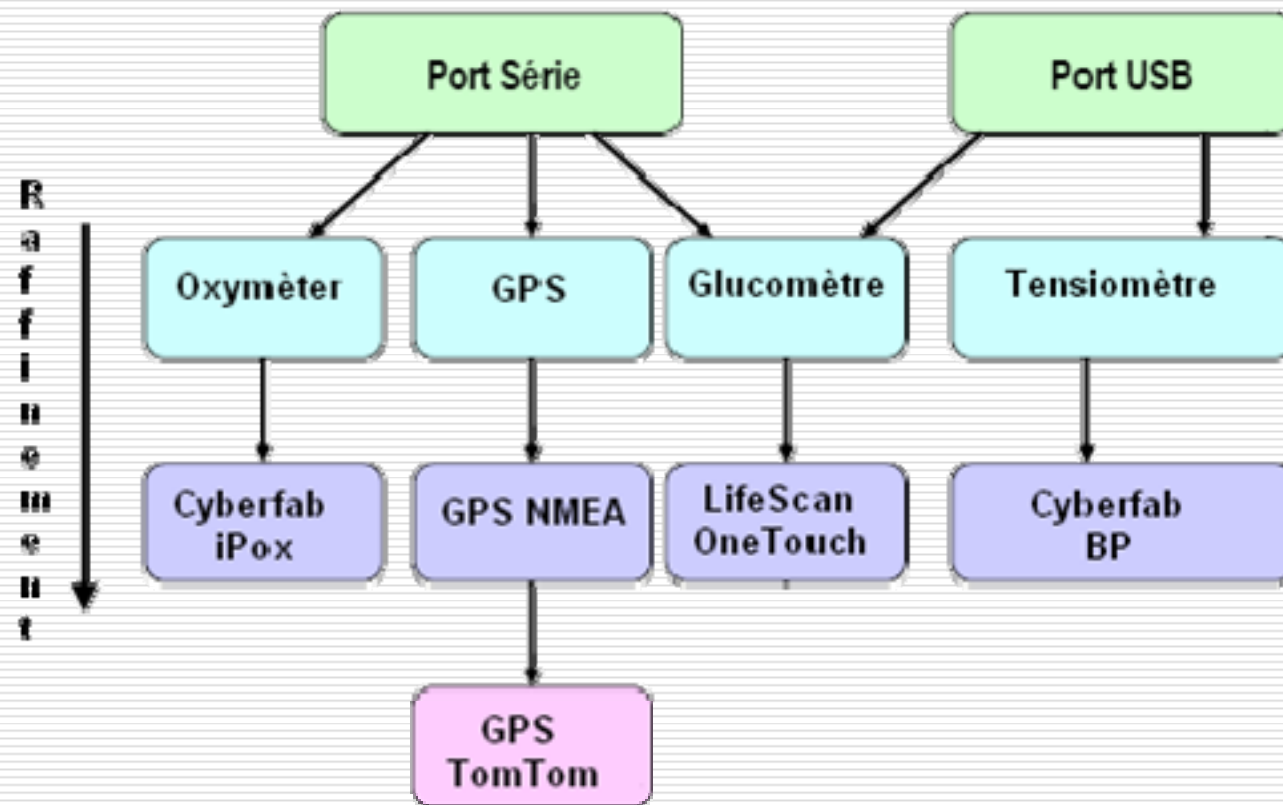
OSGi – Les services standard

- Log
 - Http
 - Device Access**
 - Declarative Services
 - Event Admin
 - Service Tracker
 - Configuration Admin
 - Preferences
 - User Admin
 - Wire Admin
 - IO Connector
 - Auto Configuration
 - Application Admin
 - Monitor Admin
 - XML Parser
 - ...
-

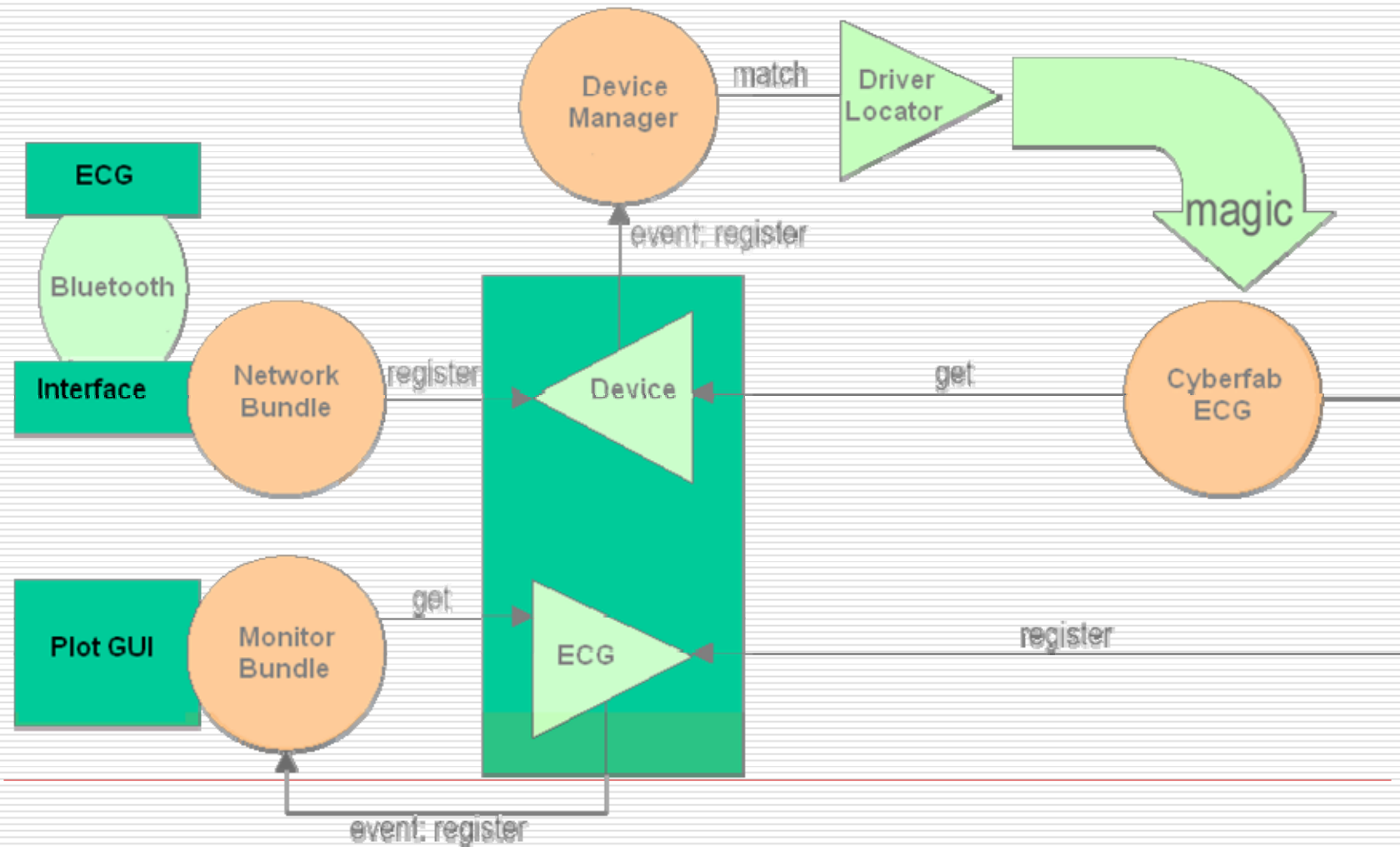
OSGi – Device Access

- ❑ Faire apparaître les dispositifs comme des service OSGi.
 - ❑ Charger les drivers des dispositifs via des bundles.
 - ❑ Mise à jour dynamique des drivers.
 - ❑ Branchement et débranchement des dispositifs sans arrêter l'exécution.
 - ❑ Raffinement des drivers.
-

OSGi – Device Access



OSGi – Device Access



OHF - SODA Project

- OHF: Open Healthcare Framework (Eclipse).

- SODA Project:
 - Eclipse Equinox (OSGi)
 - Service Activator Toolkit
 - Device Kit



Service Activator Toolkit

- Un environnement d'exécution au dessus d'OSGi.
 - Constitué de 3 bundles:
 - org.eclipse.soda.sat.core (mandatory)
 - org.eclipse.soda.sat.log.writer (optional)
 - org.eclipse.soda.sat.dependency.servlet (optional)
 - Simplifie le modèle complexe de dépendance de services OSGi dynamiques grâce à quelques hypothèses.
-

Service Activator Toolkit

- ❑ Un service exporté est enregistré lorsque son bundle est lancé, et supprimé lorsque son bundle est arrêté.
 - ❑ Un service est importés lorsque son bundle est démarré, et libéré lorsque son bundle est arrêté.
 - ❑ Un bundle enregistre ses services exportés lorsque l'ensemble de ses services importés ont été acquis. Un bundle supprime ses services exportés lorsque un ou plusieurs de ses services importés son supprimés.
-

Service Activator Toolkit

- ❑ Un bundle SAT est un bundle OSGi. L'opposé n'est pas vrai!
 - ❑ Un bundle SAT est dépendant du bundle `org.eclipse.soda.sat.core`.
 - ❑ Le bundle `org.eclipse.soda.sat.core` contient toutes les classes de base de SAT, incluant l'activateur abstrait `BaseBundleActivator` dont la majorité des bundles SAT possèdent un activateur qui est une directe or indirecte sous-classe. L'héritage n'est pas le seul moyen, mais certainement le plus simple !
 - ❑ Un bundle SAT peut importer des services exportés par d'autres bundles, même s'ils ne sont pas des bundles SAT.
 - ❑ Un service exporté par un bundle SAT est perçue comme n'importe quel autre service OSGi et peut être acquis par n'importe quel autre bundle.
-

Device Kit

- ❑ Couche d'isolation entre l'application et les dispositifs matériels.
 - ❑ Méthode uniforme d'interfaçage des dispositifs.
 - ❑ Système de communication bidirectionnel avec les dispositifs.
 - ❑ Convertisseur d'un langage KML en code Java.
-

Device Kit

- Commandes: actions qu'on peut demander d'un dispositif

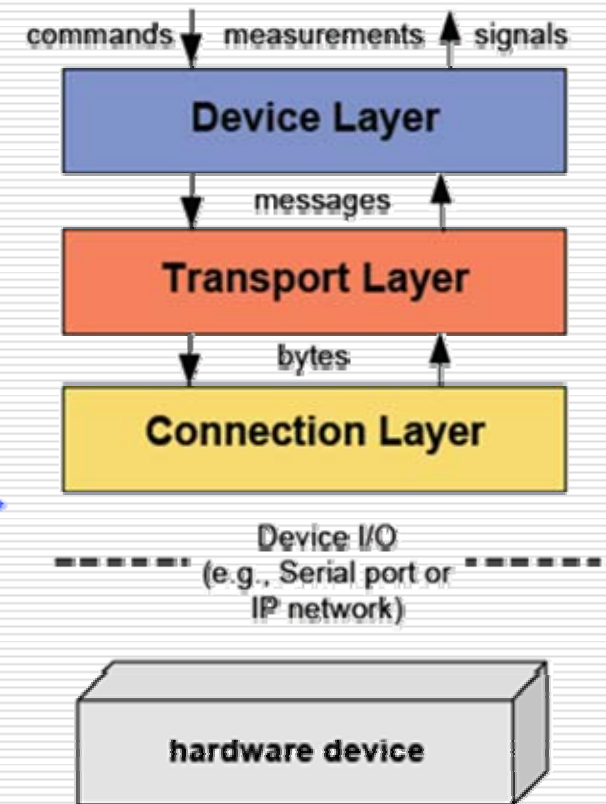
```
<command id="TogglePower">  
  <message id="TogglePowerMessage">  
    <bytes format="hex">FF, 0B</bytes>  
  </message>  
</command>
```

- Signals: rapport qu'on peut recevoir d'un dispositif

```
<signal id="LatitudeReport">  
  <message id="LatitudeReportMessage">  
    <bytes format="hex">FF, 01, 00, 00</bytes>  
    <filter id="filter0"><bytes format="hex">FF, FF, 00, 00</bytes></filter>  
    <parameter type="short"><index>2</index><size>2</size></parameter>  
  </message>  
</signal>
```

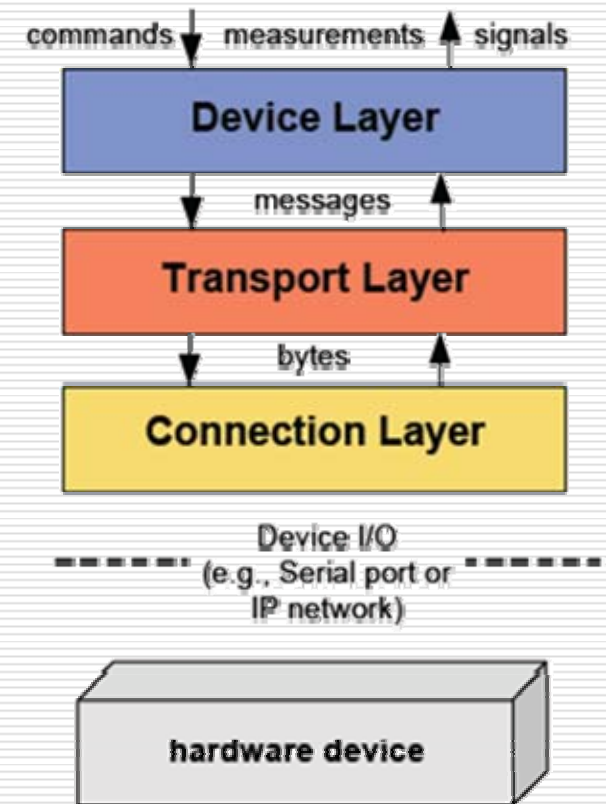
- Measurements: information qu'on peut obtenir d'un dispositif

```
<measurement id="Latitude">  
  <signal idref="LatitudeReport"/>  
</measurement>
```



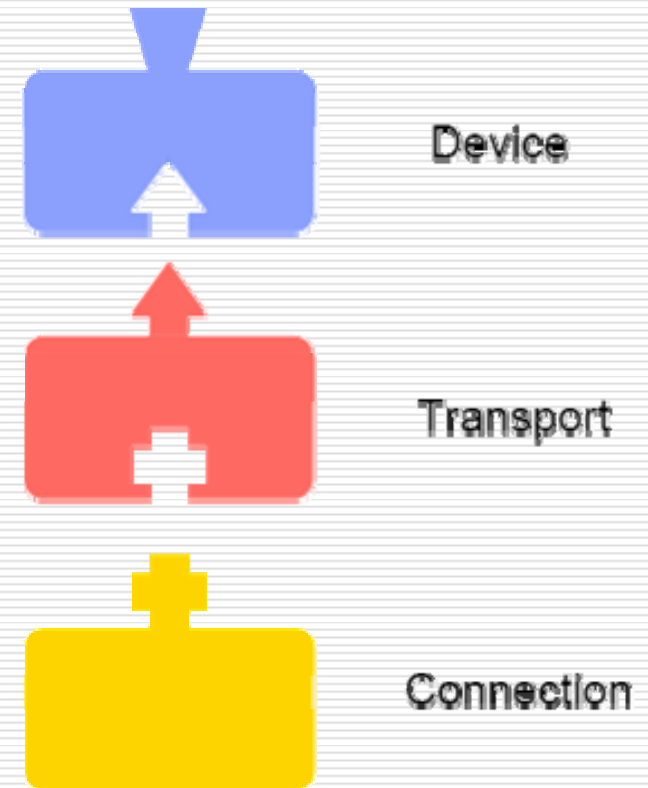
Device Kit

- Device Layer:
 - Fournit une interface pour les dispositifs.
 - Interprète la signification des messages et des paramètres dans un message.
- Transport Layer:
 - Saisit le format du message.
 - Découpe les octets d'entrée en messages valides.
- Connection Layer:
 - Supporte la lecture et l'écriture de flux d'octets pour le dispositif.



Device Kit

- ❑ Le Device Kit utilise SAT pour l'importation et l'exportation des services.
- ❑ Les couches sont faiblement couplées.
- ❑ Les couches peuvent être utilisés indépendamment.



Conclusion

- ❑ SOA offre plus d'extensibilité et de flexibilité aux applications.
 - ❑ SODA est l'adaptation de SOA pour les dispositifs communicants.
 - ❑ OHF – SODA Project est une implémentation open source du concept SODA basée sur OSGi et vise principalement les dispositifs médicaux communicants.
-

Quelques Liens...

- <http://www.upnp.org/>
 - <http://www.jini.org/>
 - <http://www.openslp.org/>
 - <http://www.soa4d.org/>
 - <http://www.soda-itea.org/>
 - [http://schemas.xmlsoap.org/ws/2006/02/d
evprof/](http://schemas.xmlsoap.org/ws/2006/02/d
evprof/)
 - <http://www.osgi.org/>
 - <http://www.eclipse.org/ohf/>
-