LIRIS
UMR 5205 CNRS

Lyon 1

Quelques outils pour le WoT

Lionel Médini : Université Claude Bernard Lyon 1 / LIRIS

M2 TIW TIW8 Janvier 2021

Objectifs de ce cours

- **Avoir une idée des standards et des initiatives existantes dans les différentes « couches » liées au WoT**
- **Identifier quelques outils dont vous pourrez avoir besoin pour réaliser un projet WoT**
- **Hypothèse**
  - Vous connaissez
    - Les objets physiques que vous allez utiliser
    - L'application visée
  - Vous voulez gagner un maximum de temps en réutilisant
    - Des formats de données
    - Du code
    - …

Préambule : Organisations de standardisation

- ETSI (European Telecommunications Standards Institute)
  - Connecting Things Cluster
- IETF (Internet Engineering Task Force)
  - CoRE working group (Constrained RESTful Environments)
  - 6lowpan working group (IPv6 over Low power WPAN)
  - ROLL working group (Routing Over Low power and Lossy networks)
- OMG (Object Management Group)
  - Data Distribution Service Portal
- OASIS (Organization for the Advancement of Structured Information Standards)
  - MQTT Technical Committee
- OGC (Open Geospatial Consortium)
  - Sensor Web for IoT Standards Working Group
- IoT-A
  - OneM2M
- W3C
  - Semantic Sensor Net Ontology
  - Web of Things Community Group
- EPC Global
- - The IEC (International Electrotechnical Commission), and ISO (International Organization for Standardization), through the JTC (Joint Technical Committee)

Source : https://www.postscapes.com/internet-of-things-protocols/

Préambule : le WoT au W3C

- **Historique**
  - 2013 : WoT Community Group
  - 2015 : WoT Interest Group
  - 2017 : WoT Working Group
- **Thèmes**
  - Use cases & Requirements
  - Technology Landscape
  - Interoperability
- **Lien**
  - https://www.w3.org/WoT

- **Spécifications (en cours)**
  - WoT Architecture
    - WoT Servient, Server, Client
  - Thing Description
  - Scripting API
  - Protocol Binding Templates
  - Tests (à venir)
- **Current Practices**
  - Discovery Mechanisms
  - Security & Privacy

Plan

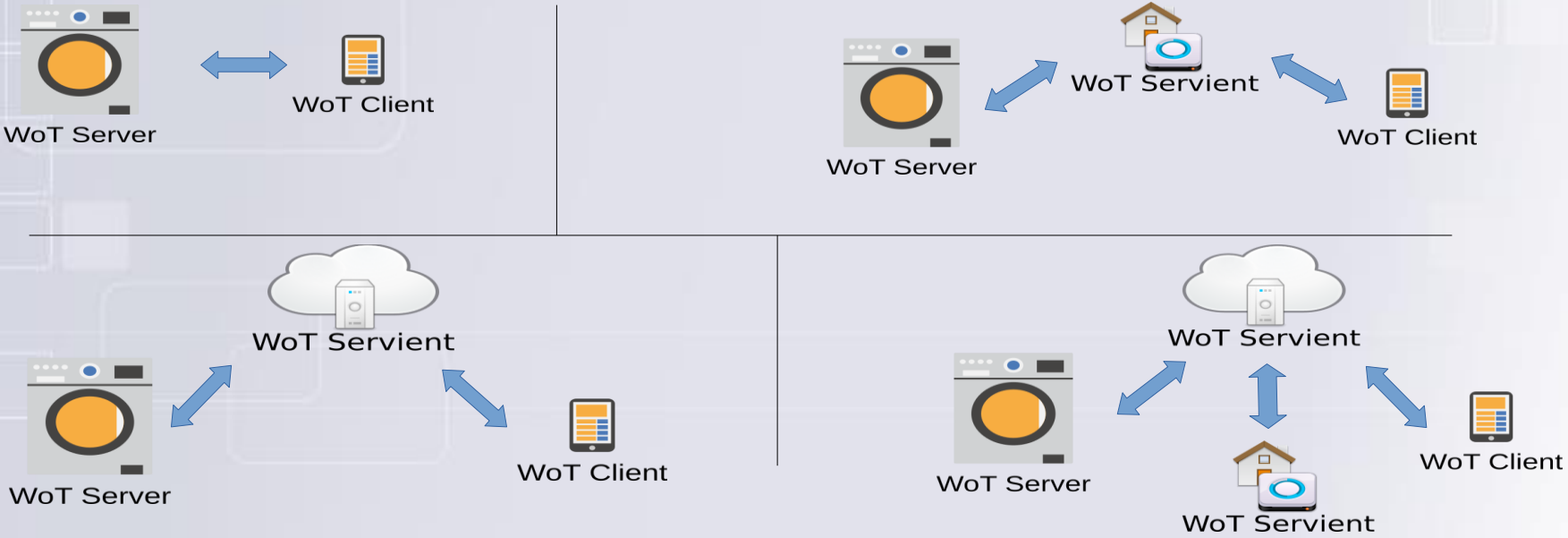
- **Architecture**
- **Programmation embarquée**
- **Communication et protocoles**
- **Données**
- **Plateformes**

Architectures logicielles dans le WoT

- **Caractéristiques**
  - Distribution potentiellement « large »
  - Communication réseau
  - Programmation embarquée
  - Contraintes de sécurité / privacy
  - Interactions avec l'environnement
- **Différents types d'architecture**
  - P2P (ex : réseaux de capteurs)
  - Centralisée (ex : smart home)
  - Intermédiaires… (ex : multi-plateformes)

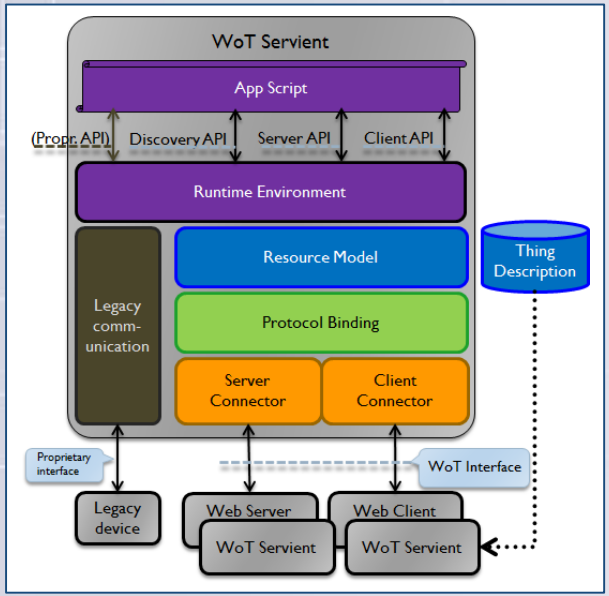
WoT architecture : scénarios

- **Différentes configurations de déploiement des « WoT Servers », « WoT Clients » et « WoT Servients »**



WoT architecture : Servient

- **« Servient » : objet virtuel fonctionnel permettant d'accéder et de contrôler les objets physiques**



Protocol Binding
- Conversion des interactions avec l'objet en fonction de la description de l'objet et des protocoles utilisés
- MQTT, HTTP, CoAP, ...

Runtime Environment
- Fournit l'API de scripts
- Utilise les modèles de resources, les protocoles de Communication, …
- Offre des APIs pour la découverte des objets

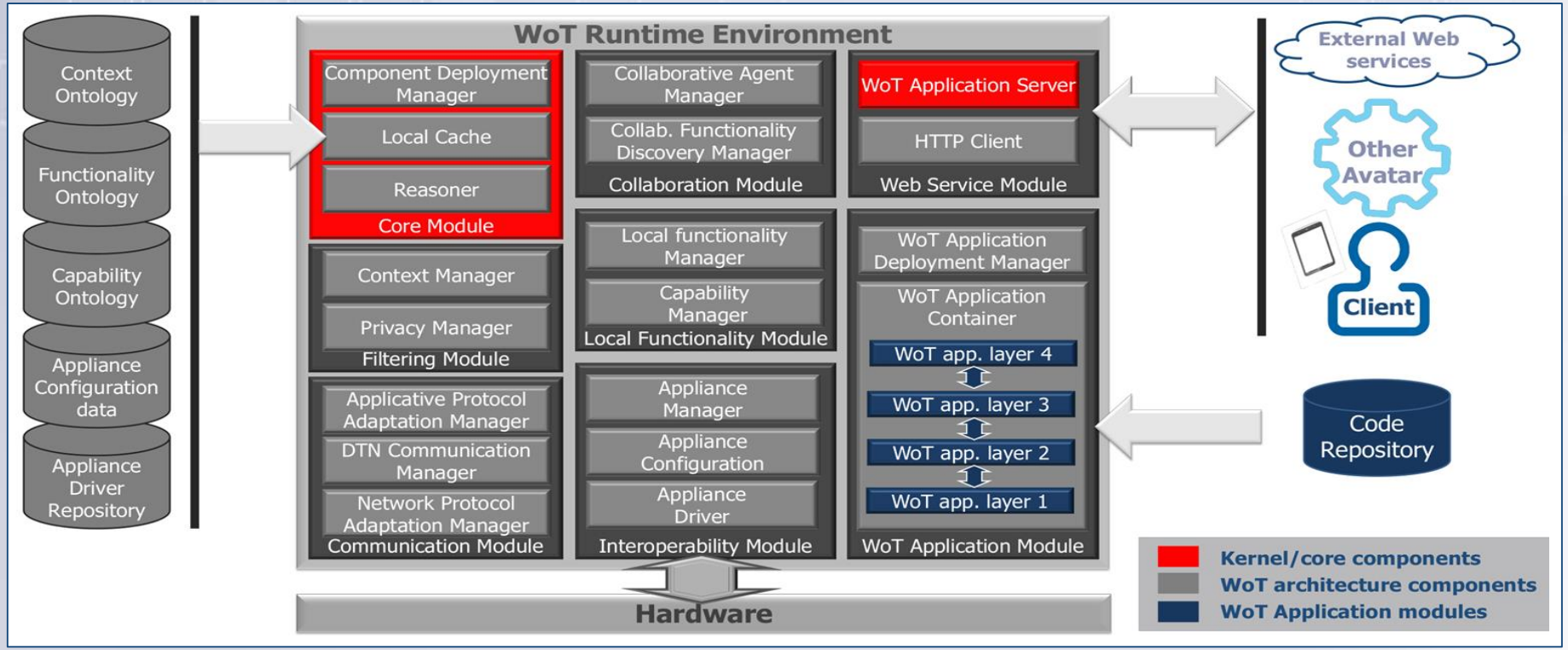
WoT architecture : découverte des objets

- **Découverte locale**
  - Méthode de découverte des objets dans les réseaux locaux
    - SSDP, mDNS/DNS-SD...
- **Découverte de proximité**
  - Méthode permettant de découvrir un objet à proximité dans l'environnement
    - Bluetooth Low Energy, RFID...
- **Découverte distante**
  - Méthode consistant à interroger des annuaires d'objets

WoT architecture : résumé

- **Objectif principal**
  - « Encadrer » les cas d'utilisation identifiés
- **Restriction**
  - Ne pas contraindre les technologies sous-jacentes
- **Spécification de haut niveau, orientée interfaces**
- **Manque la mécanique interne**
  - Distribution du code des « servients » sur plusieurs machines
  - Adaptation dynamique au contexte d'exécution
  - Tolérance aux interruptions de connectivité
  - Collaboration entre les objets et autonomie

Une implémentation de servient : l'avatar



Plan

- Architecture
- Programmation embarquée
- Communication et protocoles
- Données
- Plateformes

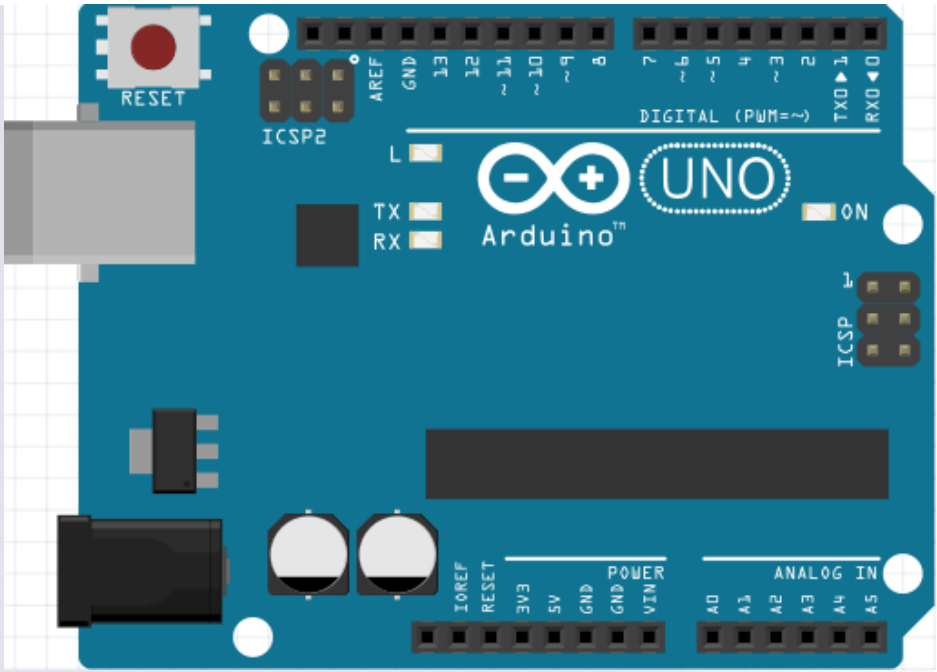
Le microcontrôleur Arduino Uno

- **Spécifications**
  - Microcontroller: Microchip ATmega328P [7]
  - Operating Voltage: 5 Volt
  - Input Voltage: 7 to 20 Volts
  - Digital I/O Pins: 14 (of which 6 provide PWM output)
  - Analog Input Pins: 6
  - DC Current per I/O Pin: 20 mA
  - DC Current for 3.3V Pin: 50 mA
  - Flash Memory: 32 KB of which 0.5 KB used by bootloader
  - SRAM: 2 KB
  - EEPROM: 1 KB
  - Clock Speed: 16 MHz

Source : https://en.wikipedia.org/wiki/Arduino_Uno

Le microcontrôleur Arduino Uno

- **Utilisation**
  - USB : alimentation (5V) et communication vers PC
  - Prise 5V : alimentation supplémentaire (moteur)
  - Masse : GND
  - Sortie simple : LED
  - E/S série
    - Receive (RX)
    - Transmit (TX)
  - E/S numériques (5V) :
    - Digital 0 → 13
  - Entrées analogiques (5V, 10bits) :
    - Analog in A0 → A5



Source : https://en.wikipedia.org/wiki/Arduino_Uno

Image: By L293D - Own work, CC BY-SA 4.0, https://commons.wikimedia.org/w/index.php?curid=64234848

Le microcontrôleur Arduino Uno

- **Programmation**
  - IDE & Web IDE : https://www.arduino.cc/en/Guide/HomePage
  - Configuration
    - Drivers (PnP) : https://www.arduino.cc/en/Guide/ArduinoUno#toc3
    - Spécifier le type de board et le port
  - Tutos, exemples de code :
    - https://create.arduino.cc/getting-started
    - https://www.arduino.cc/en/Tutorial/HomePage
    - https://playground.arduino.cc

Le microcontrôleur Arduino Uno

- **Sketches (programmes)**
  - Langage : dérivé de Processing (proche du C)
  - Fonctionnement
    - Fonction `setup()` : Initialisation

```
void setup() {// runs once when you press reset or power the board
  pinMode(ledPin, OUTPUT);      // sets the digital pin as output
}
```

    - Fonction `loop()` : Déroulement du programme

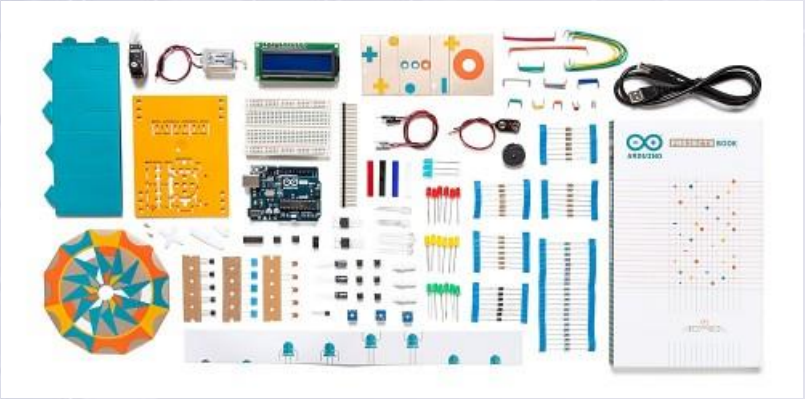
```
void loop() {  // runs over and over again forever
  digitalWrite(ledPin, HIGH);       // 5V
  delay(1000);
  digitalWrite(ledPin, LOW);       // 0V
  delay(1000);
}
```

Le microcontrôleur Arduino Uno

Autres langages de programmation

- Principe
  - Installe un sketch qui implémente un protocole de contrôle distant des ports de l'Arduino (Firmata)
  - La programmation se fait sur l'ordinateur relié à l'Arduino, dans le langage de cette bibliothèque
- Exemples
  - Johnny-five : « The JavaScript Robotics and IoT platform » : http://johnny-five.io/
  - Autres langages / OS / environnements : https://playground.arduino.cc/Main/InterfacingWithSoftware

Matériel disponible : Arduino Starter Kit



- **Description**
  - https://store.arduino.cc/genuino-starter-kit
- **Fourni avec un livre d'exemples**

- **Contenu**
  - 6 Phototransistor,
  - 3 Potentiometer 10kOhms,
  - 10 Pushbuttons,
  - 1 Temperature sensor [TMP36],
  - 1 Tilt sensor,
  - 1 alphanumeric LCD (16x2 characters),
  - 1 LED (bright white),
  - 1 LED (RGB),
  - 24 LEDs (red, green, yellow),
  - 3 LEDs (blue),
  - 1 Small DC motor 6/9V,
  - 1 Small servo motor,
  - 1 Piezo capsule [PKM17EPP-4001-B0]

Plan

- Architecture
- Programmation embarquée
- Communication et protocoles
- Données
- Plateformes

Communication dans le WoT

- **Objectif**
  - Trouver le mode de communication approprié à une application donnée
- **Hypothèse**
  - Capteurs et actionneurs (ou objets complexes) exposent un service informationnel
  - Ils produisent et/ou consomment des données

Modes de communication dans le WoT

- **Requête-réponse**
  - Scénario : interrogation de capteurs, contrôle d'actionneurs
  - Pattern : REST
  - Protocoles : HTTP 1.1, COAP, HTTP/2

Rappel : REST sur HTTP

- **Types d'opérations**
  - CRUD
- **Méthodes HTTP**
  - GET : récupération de données (capteurs)
  - PUT : définir / mettre à jour une donnée (actionneur)
  - POST : configurer / créer un comportement (objet complexe)
  - DELETE : configurer / supprimer un comportement (objet complexe)

REST + Web sémantique : Hydra

- **Objectif**
  - Simplifier la création de

« Interoperable…

  - Web sémantique

…hypermedia-driven…

  - Linked Data

…Web APIs »

  - REST

- **Principe**
  - Vocabulaire sémantique (ontologie)
  - Annotations de ressources décrivant leur API
- **Références**
  - W3C Community Group
  - Spécification
  - Démo

Modes de communication dans le WoT

- **Publish-Subscribe, Push serveur**
  - Scénarios
    - Données multiples de capteurs (flux)
    - Capteurs « low-energy » qui se réveillent de temps en temps
    - Capteurs avec connexions intermittentes
    - Objets aux comportements complexes
  - Patterns : Observer, communication orientée-message, REST + Notify
  - Protocoles : COAP, MQTT, WebSocket, WebRTC…

Publish-Subscribe

- **Exemple : plateforme robotique ROS**
  - Principe : un objet est un réseau de nœuds
    - Chaque nœud peut représenter un capteur ou un actionneur
    - Un actionneur est représenté par deux nœuds qui permettent
      - *de lui envoyer des commandes à exécuter*
      - *d'observer son état*
    - Des joints entre ces nœuds représentent les liens physiques entre ces objets
    - Le tout est décrit dans un fichier de configuration
    - Plusieurs langages permettent de programmer des comportements
  - Opérations possibles
    - Lecture de la valeur d'un nœud
    - Abonnement aux événements de changement de valeur d'un nœud
    - Envoi d'une commande de contrôle

Modes de communication dans le WoT

- **Enchaînement de services**
  - Scénarios
    - Objets aux comportements complexes
  - Patterns : chorégraphie de services,
  - Protocoles : SOAP, WSDL, BPMN, OWL-S…
    - Voir cours services

WoT Protocol Binding Templates

- **Position du problème**
  - Différents objets / plateformes hétérogènes
  - Différents protocoles
- **Proposition**
  - Un « modèle d'interactions consistant » avec la spec Thing Description (TD) du W3C
    - Extension du pattern WoT TD
      - *Pour les méthodes GET, PUT, POST, DELETE, Publish, Subscribe*
    - Vocabulaire sémantique
      - *Schémas de données : payload et types de données (« mediatypes »)*
      - *Modèles d'interactions : propriétés, actions, événements*

Plan

- Architecture
- Programmation embarquée
- Communication et protocoles
- Données
- Plateformes

Standards de représentation des données dans le WoT

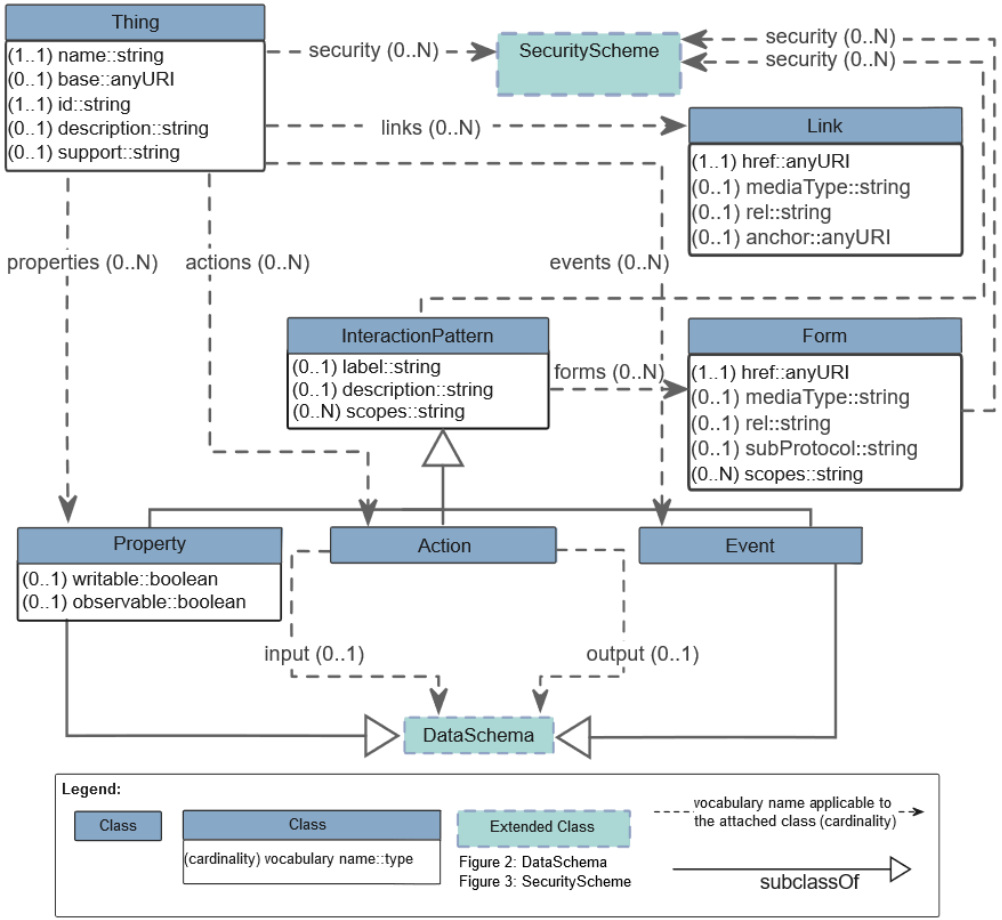
- **Encodage, formats**
  - Essentiellement pour les données textuelles
    - Formats standards du Web (XML, JSON)
    - Compression
  - Exemples
    - Concise Binary Object Representation (CBOR)
      - *Fondé sur JSON*
      - *RFC 7049 (IETF)*
    - Efficient XML Interchange (EXI)
      - *Fondé sur XML, mais version JSON à venir*
      - *Recommandation du W3C*

Vocabulaires sémantiques de représentation des données

- **W3C WoT Thing Description**
  - Représentation des objets physiques et des entités virtuelles du WoT
  - Métadonnées concernant leurs interactions
    - ➔ APIs utilisables dans les applications
  - Hébergées sur les objets ou à distance
  - Formats / versions
    - JSON-LD 1.0 (WD, outdated)
    - JSON ou JSON-LD 1.1 (WD)
  - Validation : http://plugfest.thingweb.io/playground/

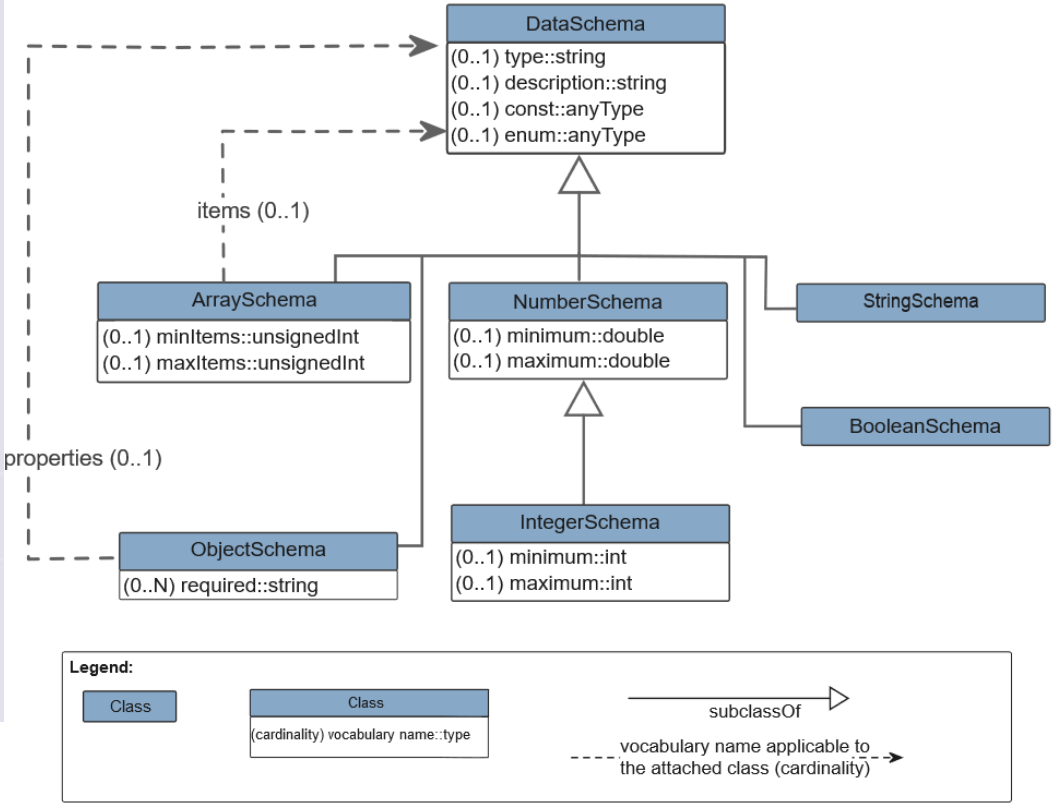
Vocabulaires sémantiques de représentation des données

- **W3C WoT Thing Description**
  - Contenu (Information Model)
    - **Core vocabulary**
    - Source : https://www.w3.org/TR/wot-thing-description/#sec-vocabulary-definition



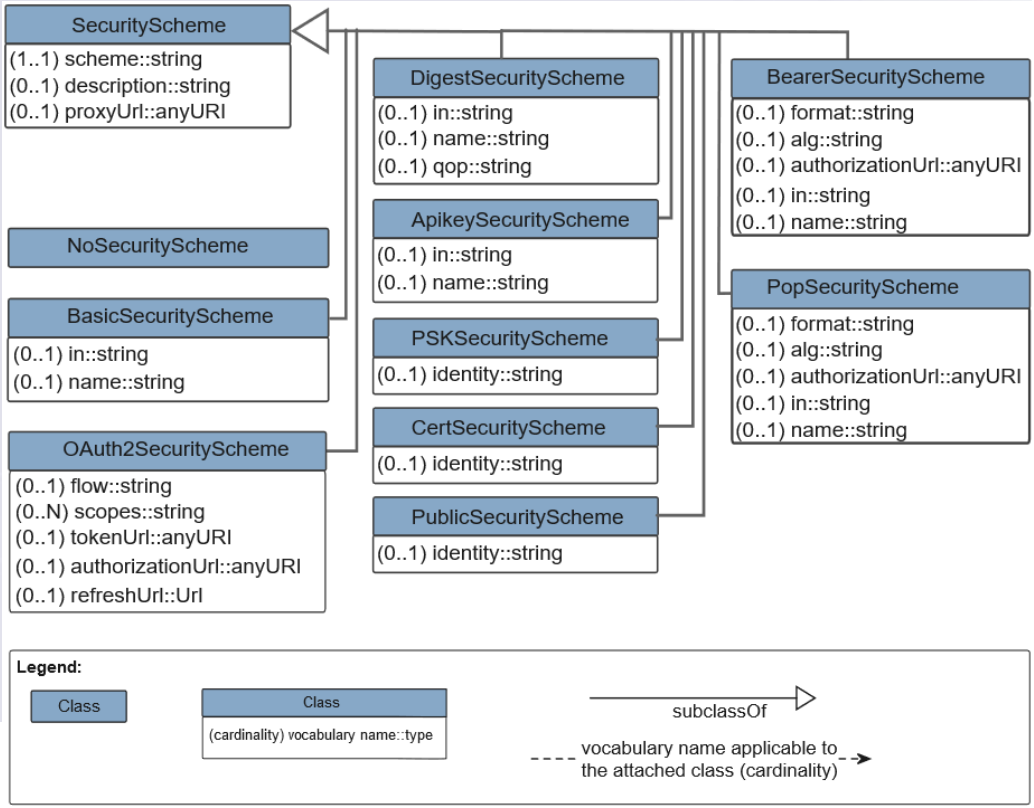
Vocabulaires sémantiques de représentation des données

- **W3C WoT Thing Description**
  - Contenu (Information Model)
    - **Data Schema**
    - Source : https://www.w3.org/TR/wot-thing-description/#sec-vocabulary-definition



Vocabulaires sémantiques de représentation des données

- **W3C WoT Thing Description**
  - Contenu (Information Model)
    - **Security**
    - Source : https://www.w3.org/TR/wot-thing-description/#sec-vocabulary-definition



Vocabulaires sémantiques de représentation des données

W3C WoT Thing Description

Exemple (JSON-LD)

```
{ "@context": "http://www.w3.org/ns/td",
  "id": "urn:dev:wot:com:example:servient:lamp",
  "name": "MyLampThing",
  "security": [{"scheme": "basic", "in":
"header"}],
  "properties": {
  "status": {
    "writable": false,
    "observable": false,
    "type": "string",
    "forms": [{
    "href": "https://mylamp.example.com/status",
    "http:methodName": "GET",
    "mediaType": "application/json"
    }]}
  },
```

```
"actions": {
  "toggle": {
    "forms": [{
    "href":
"https://mylamp.example.com/toggle",
    "http:methodName": "POST",
    "mediaType": "application/json"
    }]}
  },
  "events": {
  "overheating": {
    "type": "string",
    "forms": [{
    "href": "https://mylamp.example.com/oh",
    "subProtocol": "LongPoll",
    "mediaType": "application/json"
    }]
  }
  }
}
```

Vocabulaires sémantiques de représentation des données

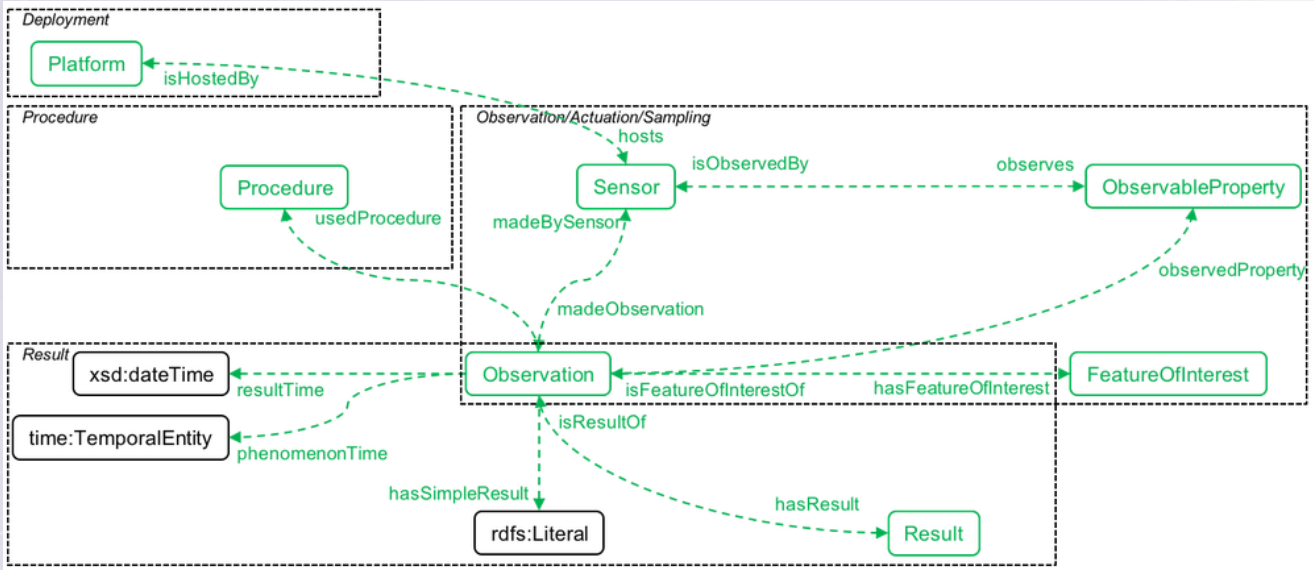
- **Semantic Sensor Network (SSN) Ontology**
  - Représentation des capteurs / actionneurs / échantilloneurs, dans leurs environnements

    ➔ observations, éléments d'intérêt, propriétés…
  - Deux vocabulaires
    - Sensor, Observation, Sample, and Actuator (SOSA)
    - Semantic Sensor Network (SSN)
  - Statut
    - W3C Recommandation

Vocabulaires sémantiques de représentation des données

Semantic Sensor Network (SSN) Ontology

- L'ontologie Sensor, Observation, Sample, and Actuator (SOSA)
- Exemple : Observation (capteurs)

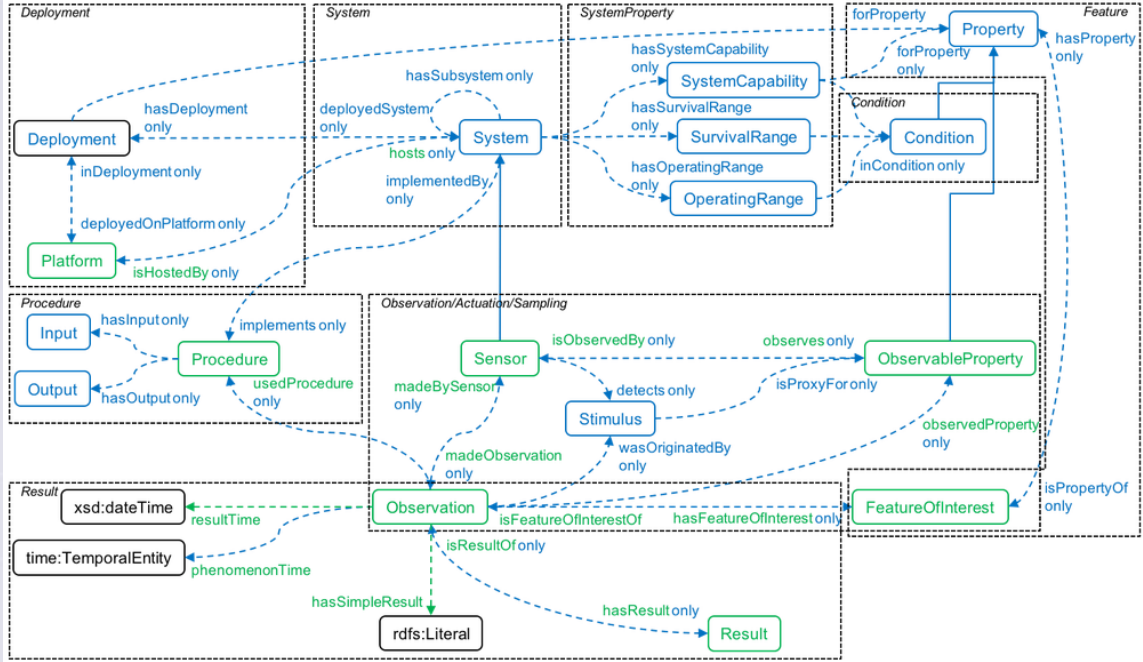


Source : https://www.w3.org/TR/vocab-ssn/#overview-of-classes-and-properties

Vocabulaires sémantiques de représentation des données

Semantic Sensor Network (SSN) Ontology

- L'ontologie SSN
- Exemple : Observation (capteurs)



Source : https://www.w3.org/TR/vocab-ssn/#overview-of-classes-and-properties

Vocabulaires sémantiques de représentation des données

- **iot.schema.org**
  - Extension de Schema.org pour l'IoT
  - Liens :
    - https://iot.schema.org/
    - https://iot.schema.org/docs/iot-gettingstarted.html
- **Smart Appliances REFerence ontology (SAREF)**
  - Supportée par l'ETSI
  - "Ancienne version" de SOSA
  - http://w3id.org/saref

Plan

- Architecture
- Programmation embarquée
- Communication et protocoles
- Données
- Plateformes

Plateformes WoT

- **Caractéristiques**
  - Permet le branchement, la découverte et l'utilisation d'objets connectés
  - Permet le déploiement et l'exécution d'applications utilisant ces objets
    - ➔ « Physical mashups »
  - Possède une interface Web
  - Ouverte
- **Liste officielle**
  - https://www.w3.org/WoT/IG/wiki/Implementations

Plateformes WoT

- **(Quelques) exemples**
  - IFTTT
    - Mashup d'objets et de services
    - Full Web (REST, Web APIs)
    - En ligne, propriétaire
    - Permet de réaliser soi-même son application
  - FI-WARE
    - Issue du projet Européen (FP7) du même nom
    - Fondée sur les standards de l'ETSI
  - ASAWoO
    - Fondée sur la notion d' « avatar »
    - Mise en œuvre d'applications à l'aide de raisonnement sémantique sur les capacités des objets

Plateformes WoT

- **(Quelques) exemples**
  - Eclipse IoT
    - Ensemble de projets Open Source sur l'IoT
    - Héberge plusieurs plateformes
    - Java, OSGi
  - OM2M
    - Projet Eclipse IoT
    - Conforme aux standards OneM2M et SmartM2M

Plateformes WoT

- **(Quelques) exemples**
  - Node-Red
    - Programmation visuelle
    - Composants en JavaScript (NodeJs)
    - Biliothèques de composants
      - *En particulier, iotschema-node-red : conforme au vocabulaire iot.schema.org*
  - Eclipse Thingweb
    - Implémentation de référence pour les standards WoT du W3C
      - *Servient, Thing Description, Scripting API*
    - Fondée sur NodeJS

Conclusion : (nos) outils pour le WoT

- **Identification**
  - URI
- **Découverte**
  - Liens hypermédia
- **Communication**
  - Schémas : C/S, Pub/Sub, Push, Pull
  - Protocoles : HTTP, CoAP, WebSocket
- **Interface**
  - Externe : REST
  - Interne : dépend de l'objet
- **Description**
  - Représentation : RDF, RDF-S, OWL (2 RL)
  - Sérialisation : JSON-LD
- **Interrogation**
  - SPARQL
- **Raisonnement**
  - Moteur d'inférences : HyLAR
- **Vocabulaires**
  - LOD : SOSA, SSN, SensorML, QUDT…
  - Web APIs : Hydra
  - W3C WoT : Thing Description…
  - ASAWoO : ontologie ad hoc

Références

- **Travaux du W3C**
  - https://www.w3.org/WoT
  - https://www.w3.org/2015/05/wot-framework.pdf
  - https://w3c.github.io/wot/architecture/wot-architecture.html
  - https://github.com/w3c/wot/wiki/Architecture-Model
  - https://www.w3.org/WoT/IG/wiki/Main_Page
- **Vocabulaires**
  - W3C SSN Incubator Group report : Semantic Sensor Network Ontology (SSN)
  - Open Geospatial Consortium : Sensor Model Language (SensorML)
  - IETF : Sensor Markup Language (senML)
- **Crédits**
  - Image LOD Cloud : Linking Open Data cloud diagram 2014, by Max Schmachtenberg, Christian Bizer, Anja Jentzsch and Richard Cyganiak. http://lod-cloud.net/

Quelques projets de référence

- **Projets européens FP7 / H2020**
  - Compose (Collaborative Open Market to Place Objects at your Service)
    - http://www.compose-project.eu/
  - IoT-A (Internet of Things Architecture)
    - http://www.iot-a.eu/public
  - Big IoT (Bridging the Interoperability Gap of the IoT)
    - http://big-iot.eu/
- **Projets « industriels » et open source**
  - Kura (http://iot.eclipse.org/)
  - OneM2M (http://www.onem2m.org/)
  - KaaProject (http://www.kaaproject.org/)
  - ...