

IOT au service de l'industrie du future

Présenté par

Mr. Hicham MEGNAFI
Email : hicham.megnafi@gmail.com
ESSA Tlemcen, Algeria



Mr. MEGNAFI Hicham (ESSA-Tlemcen)

Objectifs :

- Comprendre et définir les outils, les technologies et les concepts de l'industrie 4.0 pour la mise en œuvre de la transformation numérique des processus d'affaires des entreprises dans le contexte de la 4eme révolution industrielle.
- Sensibilisation à l'importance de l'IOT. Présentation des concepts fondamentaux de l'Internet des Objets. Compréhension de la chaîne de conception des objets connectés.

Pré-requis :

- Notions élémentaires sur les réseaux;
- Notions élémentaires sur l'électronique embarqué.

Matières	Code	VHH présentiel			Travail Personnel	Crédits	Coefficients	Mode d'évaluation	
		C	TD	TP				CC	EF
UEF91									
IOT au service de l'industrie du futur	GIMIL903	1.5	1.5	-	5	5	5	40%	60%

La note éliminatoire est fixée à 7/20

Plan de Cours

1. Projets réalisés
2. Introduction
3. Le marché de l'IOT
4. Interactions entre le « monde numérique » et le « monde physique »
5. Infrastructures pour l'IOT - Solutions technologiques
6. Application des technologies :
 - internet des objets (IOT);
 - données massives (big data);
 - Infonuagique (cloud computing et cloud manufacturing);
 - systèmes cyber-physiques (CPS);
 - Intelligence artificielle.

Plan de Cours

7. Défis : positionnement stratégique, développement de nouveaux : processus, produits et services.
8. Mise en œuvre de nouvelles capacités de : surveillance, contrôle, optimisation et autonomie.
9. Principes clés : interopérabilité, décentralisation des prises de décisions, temps-réel, intégration, agilité.
10. Stratégies de déploiement.
11. Système d'information pour l'industrie 4.0.

Plan de Cours

1. Projets réalisés

1. Study and Assembly of Quadrotor UAV for the Inspection of the Cellular Networks Relays
2. Quad rotor design for outdoor air quality monitoring
3. Internet of Things technology for efficient fire hydrant management
4. IOT applications in smart public lighting management

Projets réalisés

1. Study and Assembly of Quadrotor UAV for the Inspection of the Cellular Networks Relays



Chassis du quadrotor 450 mm

1. Assemblage hors-châssis



Outils d'assemblage

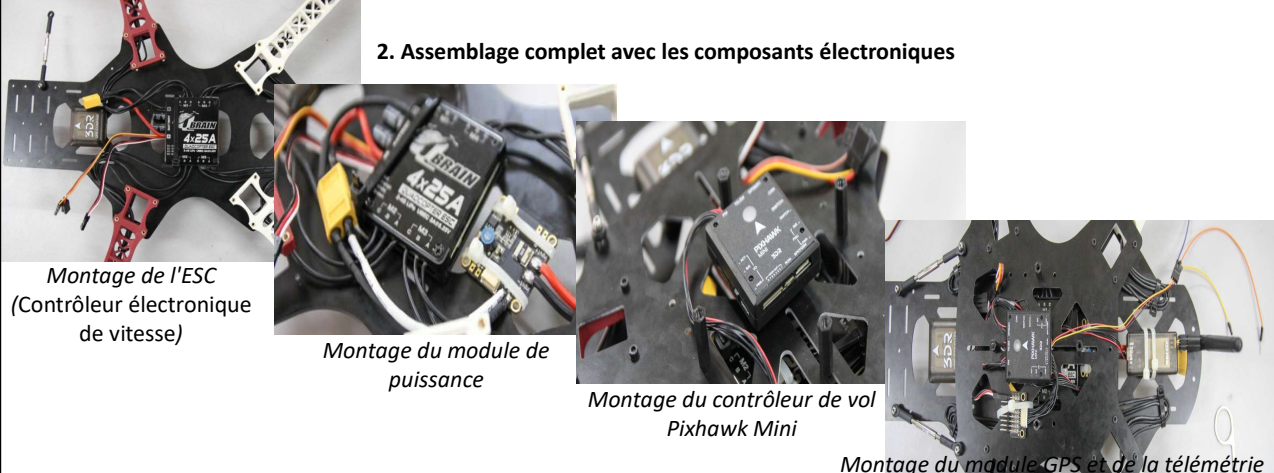
9

Mr. MEGNAFI Hicham (ESSA-Tlemcen)

Projets réalisés

1. Study and Assembly of Quadrotor UAV for the Inspection of the Cellular Networks Relays

2. Assemblage complet avec les composants électroniques



*Montage de l'ESC
(Contrôleur électronique
de vitesse)*

*Montage du module de
puissance*

*Montage du contrôleur de vol
Pixhawk Mini*

Montage du module GPS et de la télémétrie

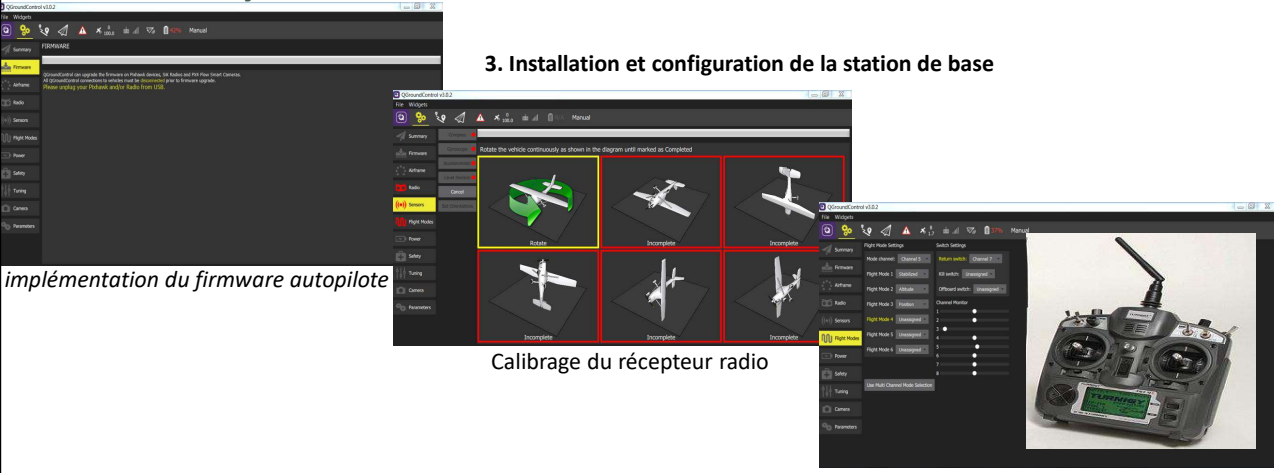
10

Mr. MEGNAFI Hicham (ESSA-Tlemcen)

Projets réalisés

1. Study and Assembly of Quadrotor UAV for the Inspection of the Cellular Networks Relays

3. Installation et configuration de la station de base



implémentation de la firmware autopilote

Calibrage du récepteur radio

Configuration des modes de vol

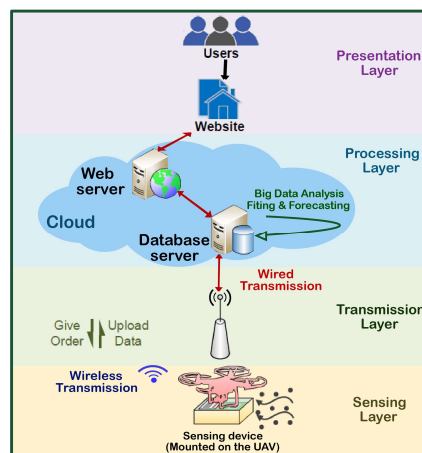
Projets réalisés

1. Study and Assembly of Quadrotor UAV for the Inspection of the Cellular Networks Relays



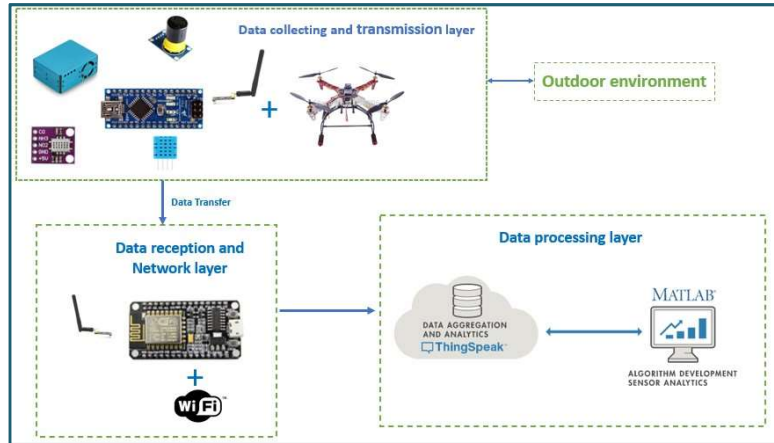
Projets réalisés

2. Quad rotor design for outdoor air quality monitoring



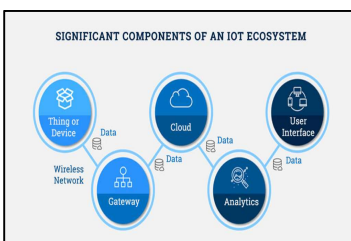
Projets réalisés

2. Quad rotor design for outdoor air quality monitoring

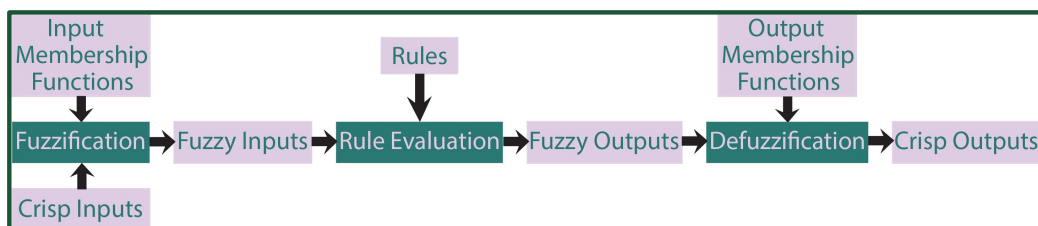
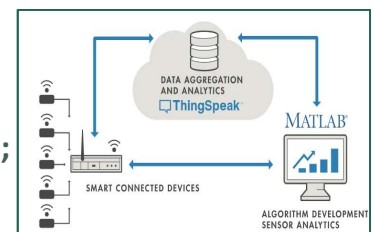


Projets réalisés

2. Quad rotor design for outdoor air quality monitoring



- Internet of Things (IoT);
- Thing Speak cloud platform;
- Fuzzy approach for calculating AQI;



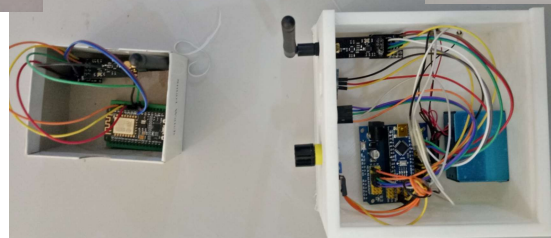
Projets réalisés

Mr. MEGNAFI Hicham (ESSA-Tlemcen)

2. Quad rotor design for outdoor air quality monitoring



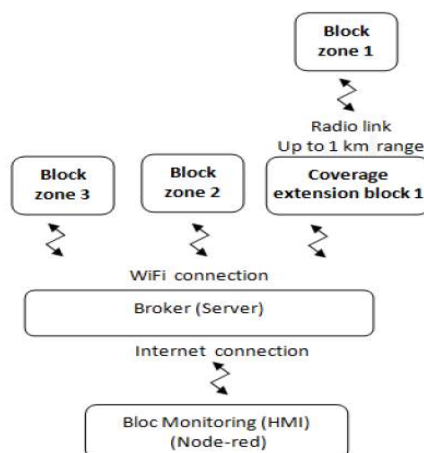
- NodeMCU ESP8266
- ATmega 328 micro controleur
- NRF24L01



Projets réalisés

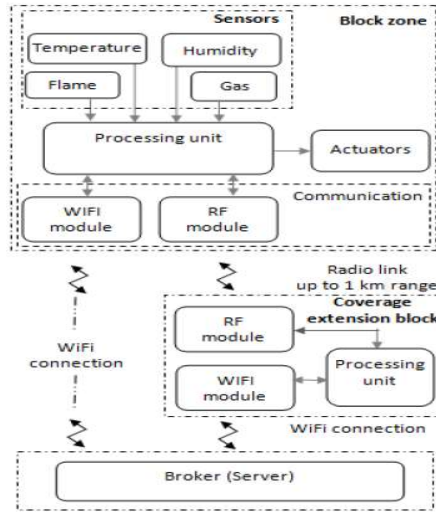
Mr. MEGNAFI Hicham (ESSA-Tlemcen)

3. Internet of Things technology for efficient fire hydrant management



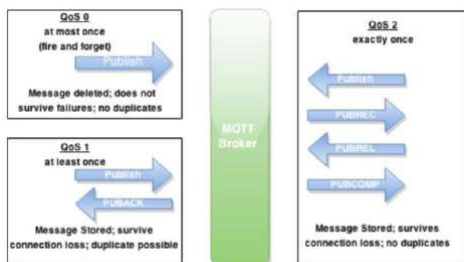
Projets réalisés

3. Internet of Things technology for efficient fire hydrant management

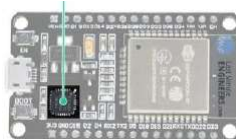
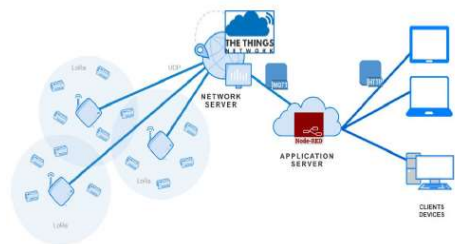


Projets réalisés

3. Internet of Things technology for efficient fire hydrant management

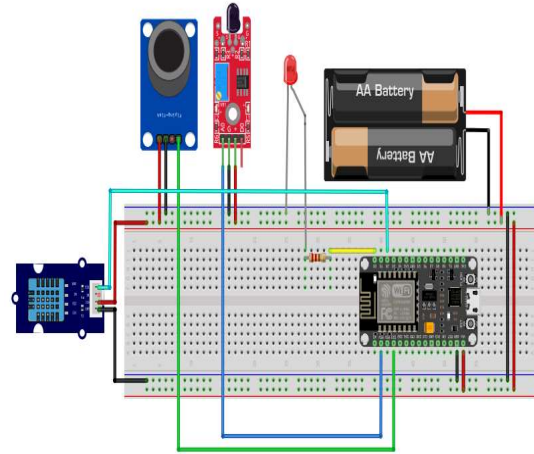
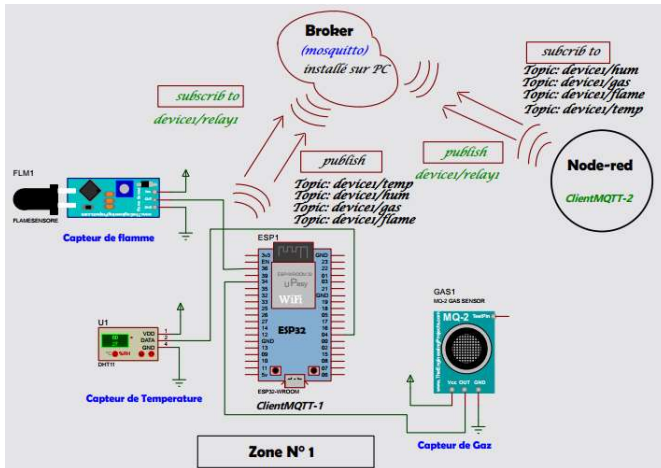


- MQQT
- ESP32
- NRF24L01
- Node Red



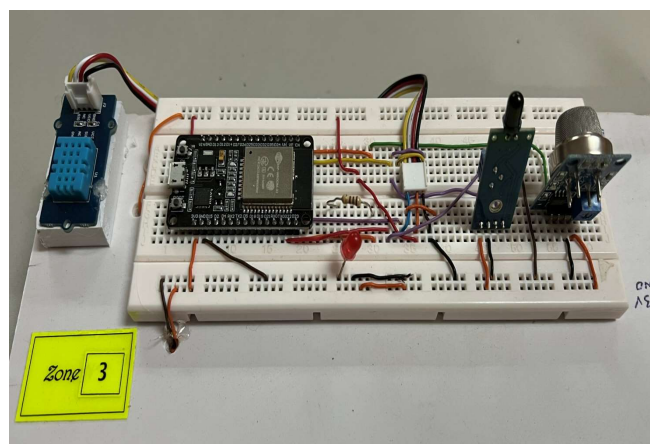
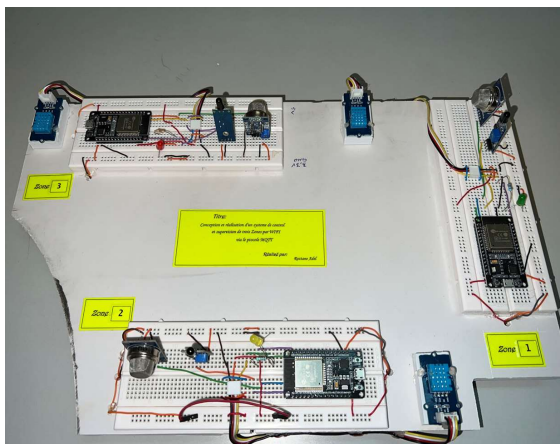
Projets réalisés

3. Internet of Things technology for efficient fire hydrant management



Projets réalisés

3. Internet of Things technology for efficient fire hydrant management



Projets réalisés

3. Internet of Things technology for efficient fire hydrant management



Projets réalisés

4. IOT applications in smart public lighting management

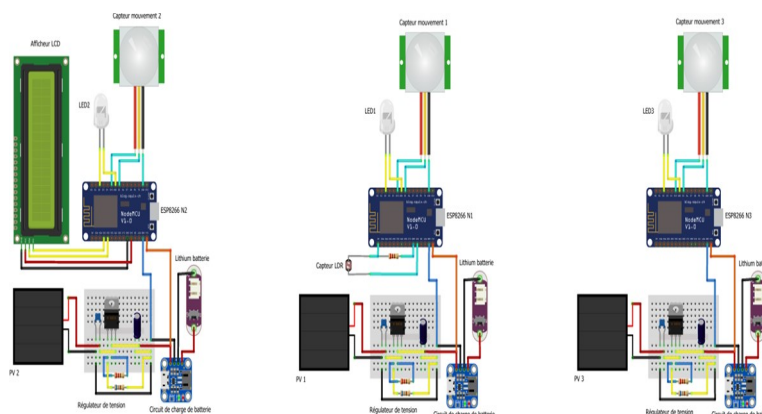


Projets réalisés

Mr. MEGNAFI Hicham (ESSA-Tlemcen)

4. IOT applications in smart public lighting management

Node MCU
Réseaux M2M/IoT
MQTT



Mini projet

Mr. MEGNAFI Hicham (ESSA-Tlemcen)

(1/2)

- 1 **Présentation de IoTsim : Modélisation et Évaluation des Réseaux IoT**
- 2 **Présentation de ThingSpeak : Gestion de Données IoT**
- 3 **Présentation de Eclipse Mosquitto : Utilisation pour la Simulation d'Applications IoT**
- 4 **Présentation de Node-RED : Création et Simulation d'Applications IoT**
- 5 **Système de Surveillance de la Qualité de l'Air avec IoTsim**
- 6 **Suivi de la Qualité de l'Eau avec ThingSpeak**

Mini projet

Mr. MEGNAFI Hicham (ESSA-Tlemcen)

- La date d'affectation des mini projets : 09/10/2023
- Chaque projet peut être fait par 4 étudiants en max
- La date de la présentation des projets :
 - **Projet 1 : 06/11/2023**
 - **Projet 2 : 13/11/2023**
 - **Projet 3 : 04/12/2023**
 - **Projet 4 : 04/12/2023**
 - **Projet 5 : 11/12/2023**
 - **Projet 6 : 11/12/2023**

Mini projet

Mr. MEGNAFI Hicham (ESSA-Tlemcen)

Titre du projet 1: Présentation de IoTsim : Modélisation et Évaluation des Réseaux IoT

Plan de Présentation de IoTsim

- 1. Introduction à l'Internet des Objets (IoT)**
- 2. Qu'est-ce que IoTsim ?**
- 3. Caractéristiques Clés de IoTsim**
- 4. Applications de IoTsim**
- 5. Installation et Configuration de IoTsim**
- 6. Modélisation d'Appareils IoT**
- 7. Simulation de Scénarios IoT**

Mini projet

Mr. MEGNAFI Hicham (ESSA-Tlemcen)

8. Protocoles de Communication Supportés

Présentation des protocoles de communication IoT pris en charge par IoTsim, comme MQTT, CoAP, HTTP, etc.

9. Analyse des Performances

Comment IoTsim permet d'analyser les performances des applications IoT simulées (latence, bande passante, consommation d'énergie, etc.)

10. Intégration avec d'autres Outils

11. Exemples de Projets avec IoTsim

12. Études de Cas

Présentation de cas réels où IoTsim a été utilisé avec succès

Mini projet

Mr. MEGNAFI Hicham (ESSA-Tlemcen)

Titre du projet 2: Présentation de ThingSpeak : Gestion de Données IoT

Plan de Présentation de IoTsim

1. Introduction à l'Internet des Objets (IoT)

2. Qu'est-ce que ThingSpeak ?

3. Principales Caractéristiques de ThingSpeak

4. Architecture de ThingSpeak

Composants Principaux : ThingSpeak est composé de canaux de données, de champs de données, d'analyses et de visualisations.

Utilisation des API : Vous pouvez accéder à ThingSpeak et interagir avec lui via des API RESTful.

Mini projet

Mr. MEGNAFI Hicham (ESSA-Tlemcen)

5. Applications de ThingSpeak

6. Configuration de ThingSpeak

7. Collecte de Données avec ThingSpeak

8. Analyse et Visualisation de Données

Outils d'Analyse : Utilisez les outils intégrés pour analyser les données en temps réel.

Création de Tableaux de Bord : Visualisez les données sous forme de graphiques interactifs.

9. Intégrations et Personnalisation

Intégration avec d'autres Services : ThingSpeak peut être connecté à d'autres services IoT.

Personnalisation : Personnalisez votre tableau de bord et vos règles de notification.

10.Exemples de Projets ThingSpeak

11.Ressources et Communauté

Mini projet

Mr. MEGNAFI Hicham (ESSA-Tlemcen)

8. Protocoles de Communication Supportés

Présentation des protocoles de communication IoT pris en charge par IoTsim, comme MQTT, CoAP, HTTP, etc.

9. Analyse des Performances

Comment IoTsim permet d'analyser les performances des applications IoT simulées (latence, bande passante, consommation d'énergie, etc.)

10.Intégration avec d'autres Outils

11.Exemples de Projets avec IoTsim

12.Études de Cas

Présentation de cas réels où IoTsim a été utilisé avec succès

Mini projet

Mr. MEGNAFI Hicham (ESSA-Tlemcen)

Titre du projet 3: Présentation de Eclipse Mosquitto : Utilisation pour la Simulation d'Applications IoT

1. Introduction

2. Qu'est-ce qu'Eclipse Mosquitto ?

Introduction à Eclipse Mosquitto en tant que broker MQTT open source.

Présentation des concepts de base : MQTT, Publisher/Subscriber, etc.

3. Installation de Eclipse Mosquitto

Guide étape par étape sur l'installation de Eclipse Mosquitto sur différentes plateformes (Windows, macOS, Linux).

Démonstration en direct de l'installation de Eclipse Mosquitto sur un système.

4. Configuration de base de Eclipse Mosquitto

Présentation des fichiers de configuration principaux de Mosquitto.

Configuration initiale de Mosquitto, notamment la définition des autorisations et des utilisateurs.

Mini projet

Mr. MEGNAFI Hicham (ESSA-Tlemcen)

5. Utilisation de Eclipse Mosquitto pour la Simulation

6. Développement de Scénarios de Simulation

7. Visualisation des Données

Présentation des outils et des méthodes pour visualiser les données simulées publiées via Mosquitto.

Démonstration de la création de tableaux de bord ou de visualisations pour surveiller les données en temps réel.

8. Avantages et Limitations

Récapitulation des avantages de l'utilisation de Eclipse Mosquitto pour la simulation IoT.

Mention des limitations potentielles, telles que la nécessité de combiner Mosquitto avec d'autres outils pour la simulation avancée.

9. Cas d'Utilisation Réels

Présentation d'exemples de cas d'utilisation réels où Eclipse Mosquitto a été utilisé pour la simulation d'applications IoT.

Mini projet

Mr. MEGNAFI Hicham (ESSA-Tlemcen)

Titre du projet 4: Présentation de Node-RED : Création et Simulation d'Applications IoT

Plan de Présentation de IoTsim

1. Introduction
2. Qu'est-ce que Node-RED ?
3. Pourquoi Simuler des Applications IoT ?
4. Simulation d'Appareils IoT avec Node-RED
 5. Présentation des nœuds de simulation d'appareils IoT disponibles dans Node-RED.
 6. Démonstration en direct de la création d'une simulation IoT simple :
 1. Configuration d'un capteur simulé (par exemple, un capteur de température).
 2. Publication de données simulées via MQTT.
 3. Réception et traitement des données simulées.

Mini projet

Mr. MEGNAFI Hicham (ESSA-Tlemcen)

5. Personnalisation et Scénarios de Simulation
 6. Expliquer comment personnaliser les simulations dans Node-RED :
 1. Configuration de plusieurs appareils simulés.
 2. Création de scénarios de simulation complexes (par exemple, des interactions entre plusieurs capteurs).
 7. Démonstration de la création d'un scénario de simulation plus avancé (si possible).
6. Visualisation des Données
 7. Présentation de la création de tableaux de bord interactifs pour visualiser les données simulées en temps réel dans Node-RED.
 8. Démonstration de la création de graphiques et de tableaux de bord.
7. Utilisation Avancée
 8. Discussion sur les utilisations avancées de Node-RED pour la simulation IoT, telles que l'intégration avec des bases de données, des services cloud, etc.
 9. Présentation d'exemples de cas d'utilisation réels.
8. Avantages et Limitations

Mini projet

Mr. MEGNAFI Hicham (ESSA-Tlemcen)

Titre du projet 5: Système de Surveillance de la Qualité de l'Air avec IoTsim

La surveillance de la qualité de l'air est un sujet important pour la santé publique et l'environnement. Vous pouvez créer un projet de surveillance de la qualité de l'air en utilisant IoTsim pour simuler un réseau de capteurs virtuels qui collectent des données de qualité de l'air dans une zone donnée. Voici un exemple de projet sur la surveillance de la qualité de l'air avec IoTsim :

Objectif du projet : L'objectif de ce projet est de simuler un réseau de capteurs de qualité de l'air virtuels à l'aide d'IoTsim. Le système doit collecter des données de qualité de l'air dans une zone spécifique, les transmettre à une passerelle virtuelle, et stocker et visualiser ces données pour permettre une surveillance en temps réel de la qualité de l'air.

Mini projet

Mr. MEGNAFI Hicham (ESSA-Tlemcen)

Étapes du projet :

- 1. Configuration de l'environnement :**
 - Installez IoTsim sur votre ordinateur.
 - Créez un environnement virtuel pour la simulation de la qualité de l'air.
- 2. Modélisation des capteurs :**
 - Définissez des modèles de capteurs de qualité de l'air virtuels avec IoTsim. Ces capteurs doivent générer des données synthétiques, telles que les niveaux de particules fines (PM2.5), les niveaux d'ozone, etc.
- 3. Création de la topologie du réseau :**
 - Concevez la topologie de votre réseau de capteurs virtuels en plaçant les capteurs dans différentes zones géographiques.
 - Configurez la communication entre les capteurs et une passerelle virtuelle.
- 4. Simulation de la collecte de données :**
 - Exécutez la simulation pour collecter des données de qualité de l'air à partir des capteurs.
 - Générez des données synthétiques en fonction de scénarios spécifiques, tels que des niveaux de pollution élevés en cas de congestion du trafic.
- 5. Transmission des données :**
 - Simulez la transmission des données depuis les capteurs jusqu'à une passerelle virtuelle en utilisant des protocoles de communication IoT tels que MQTT ou CoAP.

Mini projet

Mr. MEGNAFI Hicham (ESSA-Tlemcen)

Étapes du projet :

- 6. Stockage des données :**
Configurez un système de stockage virtuel pour stocker les données de qualité de l'air collectées.
- 7. Visualisation des données :**
Utilisez des outils de visualisation de données (par exemple, des graphiques en temps réel) pour afficher les données de qualité de l'air collectées et les rendre accessibles aux utilisateurs.
- 8. Analyse et alertes :**
Mettez en place des mécanismes d'analyse pour détecter des problèmes potentiels de qualité de l'air, tels que des niveaux de pollution dangereux.
Configurez des alertes en fonction des seuils définis.
- 9. Rapports et API :**
Créez des rapports automatisés sur la qualité de l'air.
Développez une API pour permettre l'accès aux données de qualité de l'air aux applications tierces.
- 10. Tests et évaluation :**
Testez le système dans divers scénarios, notamment des pics de pollution, pour évaluer son efficacité et sa robustesse.
- 11. Documentation et présentation :**
Documentez l'ensemble du projet, y compris les modèles de capteurs, la topologie du réseau, les résultats de simulation, etc.
Préparez une présentation pour expliquer votre projet et ses résultats.

Mini projet

Mr. MEGNAFI Hicham (ESSA-Tlemcen)

Titre du projet 6: Suivi de la Qualité de l'Eau avec ThingSpeak

Objectif du projet : L'objectif de ce projet est de développer un système de suivi de la qualité de l'eau en utilisant le simulateur ThingSpeak pour collecter, stocker et analyser des données de qualité de l'eau simulées. Le système devrait être capable de surveiller les paramètres clés de la qualité de l'eau et de générer des alertes en cas de déviations par rapport aux normes de qualité.

Mini projet

Mr. MEGNAFI Hicham (ESSA-Tlemcen)

Étapes du projet :

1. Configuration de l'Environnement de Simulation

Installez et configurez le simulateur ThingSpeak pour le projet.

Créez un compte ThingSpeak si vous n'en avez pas déjà un.

2. Modélisation des Capteurs Virtuels

Définissez des modèles de capteurs virtuels pour mesurer différents paramètres de la qualité de l'eau, tels que la turbidité, le pH, la conductivité, la température, etc.

Configurez la simulation pour générer des données synthétiques de qualité de l'eau en fonction de scénarios variés.

3. Création des Canaux ThingSpeak

Créez des canaux ThingSpeak dédiés à chaque paramètre de qualité de l'eau que vous surveillerez.

Configurez les champs de données appropriés dans chaque canal pour stocker les mesures.

4. Simulation de la Collecte de Données

Simulez la collecte de données de qualité de l'eau par les capteurs virtuels.

Envoyez les données simulées aux canaux ThingSpeak correspondants.

5. Analyse des Données

Utilisez les fonctionnalités d'analyse intégrées de ThingSpeak pour effectuer des calculs et des traitements sur les données de qualité de l'eau.

Créez des graphiques en temps réel pour visualiser les données.

Mini projet

Mr. MEGNAFI Hicham (ESSA-Tlemcen)

Étapes du projet :

6. Règles et Alertes

Définissez des règles dans ThingSpeak pour déclencher des alertes en cas de déviations significatives par rapport aux valeurs cibles.

Configurez les notifications (par exemple, e-mail) pour les alertes.

7. Intégration avec des Services Externes

Intégrez ThingSpeak avec des services de cartographie pour afficher les données de qualité de l'eau sur une carte interactive.

Intégrez des services de communication pour partager les données avec d'autres parties prenantes.

8. Tests et Validation

Effectuez des tests approfondis pour vous assurer que le système de suivi de la qualité de l'eau fonctionne correctement dans différents scénarios.

Validez les alertes générées en simulant des événements de dégradation de la qualité de l'eau.

9. Documentation

Documentez l'ensemble du projet, y compris les modèles de capteurs, la configuration ThingSpeak, les résultats des simulations et des tests.

Mini projet

Mr. MEGNAFI Hicham (ESSA-Tlemcen)

Étapes du projet :

- 6. Stockage des données :**
Configurez un système de stockage virtuel pour stocker les données de qualité de l'air collectées.
- 7. Visualisation des données :**
Utilisez des outils de visualisation de données (par exemple, des graphiques en temps réel) pour afficher les données de qualité de l'air collectées et les rendre accessibles aux utilisateurs.
- 8. Analyse et alertes :**
Mettez en place des mécanismes d'analyse pour détecter des problèmes potentiels de qualité de l'air, tels que des niveaux de pollution dangereux.
Configurez des alertes en fonction des seuils définis.
- 9. Rapports et API :**
Créez des rapports automatisés sur la qualité de l'air.
Développez une API pour permettre l'accès aux données de qualité de l'air aux applications tierces.
- 10. Tests et évaluation :**
Testez le système dans divers scénarios, notamment des pics de pollution, pour évaluer son efficacité et sa robustesse.
- 11. Documentation et présentation :**
Documentez l'ensemble du projet, y compris les modèles de capteurs, la topologie du réseau, les résultats de simulation, etc.
Préparez une présentation pour expliquer votre projet et ses résultats.

Mr. MEGNAFI Hicham (ESSA-Tlemcen)

Plan de Cours

1. Projets réalisés

2. **Introduction**

- Intervenant
- Révolution industrielle
- 4eme Révolution industrielle
- Industrie 4.0
- Grille d'analyse
- Déclinaison processus, produits et services
- Grille d'analyse : avec exemple industrie 4.0

Intervenant dans l'industrie du future

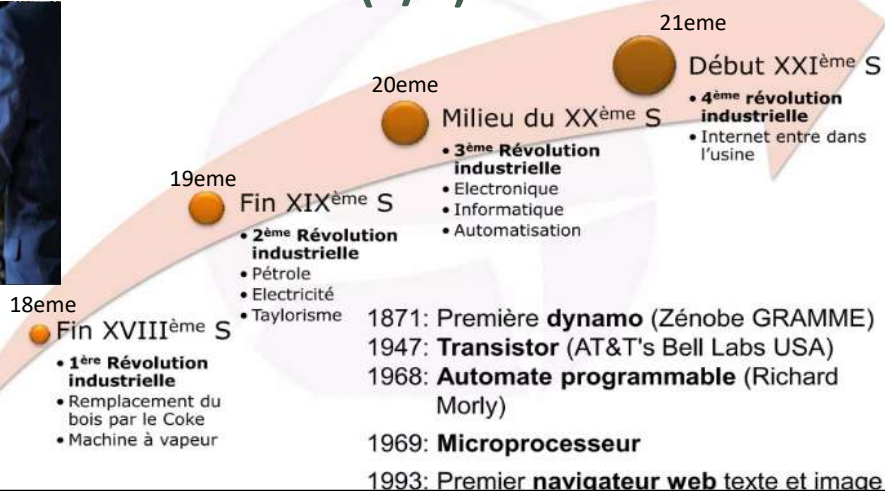
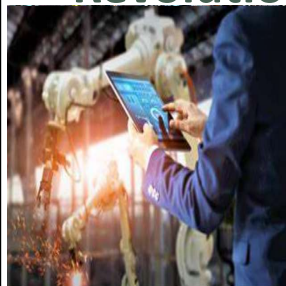
- Ingénieur en **génie informatique et industrielle**;
- Diverses expériences industrielle;
 - ❑ EADS (Airbus), Motorola (téléphonie)
 - ❑ Fabricant de machines de production robotisées pour le semi-conducteur (RECIF) : Intel, IBM, Samsung
- Croisés des chemins entre : produit et méthode de production;

Révolution industrielle (1/2)

- Passage d'une **société paysanne et artisanale** à une **société industrielle et commerçante**
- Les révolutions industrielles : étapes majeures de cette transformation



Révolution industrielle (2/2)



4eme Révolution industrielle (1/3)

Initialement apparu en 2011 à la faveur d'un effort allemand, le terme 'Industrie4.0' évoque une 4^{ème} révolution industrielle [Drath & Horch, 2014].

Plusieurs termes sont associés à cette 4^o révolution :

- Smart Manufacturing**, Usine du futur, **Manufacturing Renaissance** ...
- Termes qui interpellent l'usine et la production manufacturière

Rappel (1/3)

Mr. MEGNAFI Hicham (ESSA-Tlemcen)

Smart Manufacturing (Fabrication intelligente)

Smart Manufacturing: se réfère à un concept de fabrication avancée qui intègre les **technologies de l'information, de l'automatisation et de l'Internet des objets (IoT)** pour améliorer les **processus de production, la gestion de la chaîne d'approvisionnement, et la prise de décision dans l'industrie.**

Il s'agit d'une approche de la fabrication qui exploite les données en **temps réel**, l'analyse avancée, la connectivité des machines et des systèmes, ainsi que l'automatisation pour optimiser les opérations de fabrication.



Rappel (2/3)

Mr. MEGNAFI Hicham (ESSA-Tlemcen)

Manufacturing Renaissance (Renaissance de la Fabrication)

est un concept qui se réfère à une période de redynamisation, de rénovation ou de revitalisation de l'industrie manufacturière dans une région ou à l'échelle mondiale.

La Manufacturing Renaissance peut inclure des initiatives pour stimuler **l'innovation dans la fabrication**, favoriser la compétitivité, créer des emplois, développer des chaînes d'approvisionnement durables et promouvoir l'adoption de technologies avancées telles que **l'automatisation, l'Internet des objets (IoT) et l'intelligence artificielle (IA)** pour améliorer la **productivité et la qualité des produits manufacturés.**



Rappel (3/3)

"Smart Manufacturing" et "Manufacturing Renaissance" sont deux concepts distincts dans le domaine de la fabrication industrielle, bien qu'ils puissent être liés de certaines manières. Voici les principales différences entre les deux :

Smart Manufacturing (Fabrication Intelligente)

Smart Manufacturing se concentre sur l'application de technologies avancées, telles que l'Internet des objets (IoT), l'automatisation, l'analyse des données, et l'intelligence artificielle, pour optimiser les opérations de fabrication en temps réel.

Il s'agit d'une approche technologique visant à améliorer l'efficacité, la qualité et la flexibilité des processus de fabrication.

L'accent est mis sur la numérisation des opérations de fabrication pour prendre des décisions basées sur les données et améliorer la connectivité entre les machines.

Manufacturing Renaissance (Renaissance de la Fabrication)

Manufacturing Renaissance se réfère à une période de revitalisation ou de renouvellement de l'industrie manufacturière.

Il met l'accent sur la renaissance de l'industrie manufacturière en tant que secteur économique clé, notamment grâce à des investissements, à des politiques industrielles et à des initiatives visant à stimuler la fabrication.

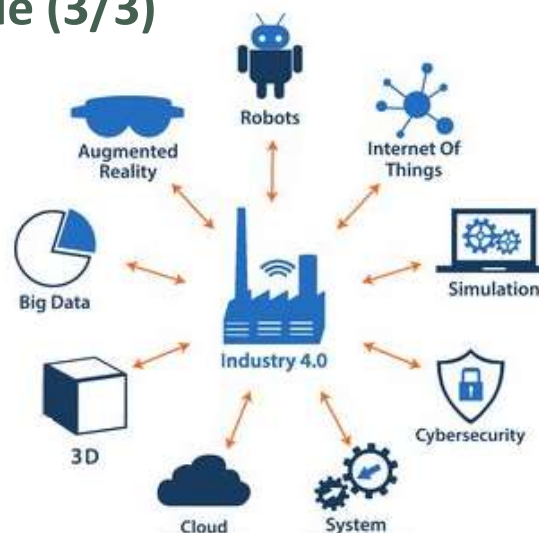
Cela peut inclure des actions pour améliorer la compétitivité, renforcer les chaînes d'approvisionnement locales, promouvoir l'innovation et créer des emplois dans le secteur manufacturier.

4eme Révolution industrielle (2/3)

- Les éléments à la base de l'Industrie 4.0 sont:
 - La présence de communication en temps réel** pour surveiller et agir sur les processus physiques.
 - Les systèmes communiquent et coopèrent entre eux**, et avec les humains, pour décentraliser la prise de décisions.
- L'Industrie 4.0 met donc l'accent sur le numérique et la connectivité ubiquitaire ou omniprésent

4eme Révolution industrielle (3/3)

Quelles sont les technologies en jeu dans cette 4eme révolution industrielle ?



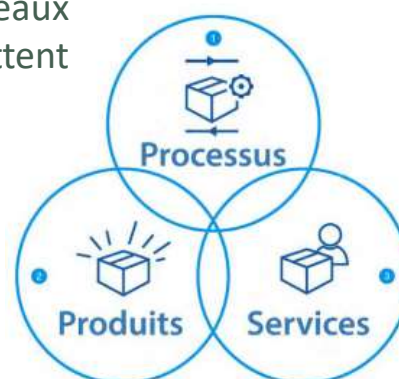
Grille d'analyse

(1/1)

Le **numérique** et la **connectivité ubiquitaire** favorisent le développement de nouveaux **processus** manufacturiers, mais aussi permettent l'émergence de nouveaux **produits** ou **services**.

L'Industrie 4.0 se décline donc en termes de

- Processus
- Produits
- Services



Déclinaison processus, produits et services (1/3)

➤ Processus

➤ Produits

➤ Services

L'Industrie 4.0 promet une transformation des modes de production. Les décisions de production sont adaptées en temps réel par des machines autonomes et la communication entre systèmes cyber-physiques. Vers le 'lot unitaire'. in fine: améliorer les processus de production. Ex.: une machine appelle un robot pour l'alimenter.

Déclinaison processus, produits et services (2/3)

➤ Processus

➤ Produits

➤ Services




Les produits sont utilisés pour leur fonction habituelle, mais sont connectés pour permettre la collecte de données en temps réel. Ces données peuvent être transmises et analysées pour en permettre la surveillance ou permettre au système de s'adapter ou d'agir sur son environnement. Ex.: voiture autonome.

Déclinaison processus, produits et services (3/3)

- Processus
- Produits
- Services

La disponibilité des données et les possibilités d'analyse amènent des opportunités de développement de nouveaux services (data-based services - Ex.: chandail HexoSkin (°C, FC, ...))

Grille d'analyse : avec exemple industrie 4.0 (1/7)

Processus 	Masonite Poka Y Worximity	Feux de circulation intelligents Touje Shaper	Usine Bosch Rexroth Lunettes DAQRI Parc éolien ITMI	SmartFactory
Produits 	Réfrigérateur avec alarme Raquette Babolat	Voiture Tesla Moteurs John Deere Emballage 4.0	Éoliennes WTC	Voiture autonome iRobot Roomba
Services 	Smart Socks Sensoria Gant de Golf Zepp Chandail Hexoskin Bracelet Proxxt	Ford SafeCap ProGlove Ampoule Philips PowerSuit Bracelet Up2	Ventilateur BigAss	Fermes connectées
	Surveillance	Contrôle	Optimisation	Autonomie

Grille d'analyse : avec exemple industrie 4.0 (2/7)

Processus 	Masonite Poka & Worximity	
Produits 	Réfrigérateur avec alarme	Voiture Tesla N Jor
Services 	Smart Socks Sensoria Gant de Golf Zepp Chandail Hexoskin Bracelet Proxoi	Emballa 4.0 Ford SafeCap A Pe Bracelet Up2
	Surveillance	Contrô

Feux de circulation Intelligents

Un système de caméras thermiques permettant de faire évoluer la durée des feux en fonction de l'affluence des piétons (Ex ville de Metz).



Comptage des véhicules arrivant sur un carrefour grâce à des capteurs dans la chaussée : diminution du temps d'arrêt des véhicules à un feu rouge en faisant varier le temps de passage en fonction du nombre de véhicules présents (Ex ville de Strasbourg).

Grille d'analyse : avec exemple industrie 4.0 (3/7)

iRobot Roomba

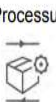


Un système autonome est capable d'apprendre de son environnement, d'auto-diagnostiquer ses besoins et de s'adapter aux préférences de l'utilisateur.

- Il optimise la puissance d'aspiration en fonction de la surface.
- Il retourne à sa base lorsqu'il faut recharger les batteries.
- Bien sûr, une apps sur le téléphone permet de suivre le travail...

	Usine Bosch Rexroth Parc éolien ITMI	Lunettes DAQRI
	Éoliennes WTC	Voiture autonome iRobot Roomba
	Ventilateur BigAss	Fermes connectées
	Optimisation	Autonomie



Grille d'analyse : avec exemple industrie 4.0 (4/7)

Processus 	Doxel SmartRock SkyCatch	Hololens MAX- ThyssenKrupp SAM - Fa
Produits 		
Services 	Caterpillar SmartBand BioSite	
	Surveillance	Contrôle

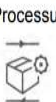


Caterpillar SmartBand

Optimiser les plannings et conditions de travail des ouvriers grâce aux données récoltées par la SmartBand :

- Modéliser et prédire le risque de fatigue des travailleurs
- Donner aux travailleurs les moyens de gérer leur propre fatigue via apps sur téléphone
- Identifier les risques de fatigue à venir pendant la période de travail



Grille d'analyse : avec exemple industrie 4.0 (5/7)

Processus 	Doxel SmartRock SkyCatch	Hololens MAX- ThyssenKrupp SAM - Fas
Produits 		
Services 	Caterpillar SmartBand BioSite	
	Surveillance	Contrôle




SkyCatch modelisation

Grâce à la photogrammétrie, un modèle virtuel du site est réalisé. Il est ensuite comparé au modèle virtuel de conception afin d'identifier l'état d'avancement et les difficultés rencontrées. Cela permet d'ajuster la stratégie d'action en cours de route.

Conséquence :
 Diminution des risques d'erreurs et accélération de la capture d'information



Grille d'analyse : avec exemple industrie 4.0 (6/7)

Processus 	Doxel Hololens SmartRock MAX- ThyssenKrupp SkyCatch	SAM Fa
Produits 		
Services 	Caterpillar SmartBand BioSite	
	Surveillance	Contrôl

Doxel Lidar robot


Intelligence artificielle au service de la productivité pour la construction. Le robot est capable de se déplacer sur le chantier de manière autonome afin de collecter des informations par numérisation de son environnement.
 Le modèle numérique (nuage de points) qui en ressort permet de suivre en temps quasi réel l'évolution des travaux de construction et les éventuelles erreurs par comparaison avec à la maquette numérique initiale.



Grille d'analyse : avec exemple industrie 4.0 (7/7)

MAX-AI traitement des déchets

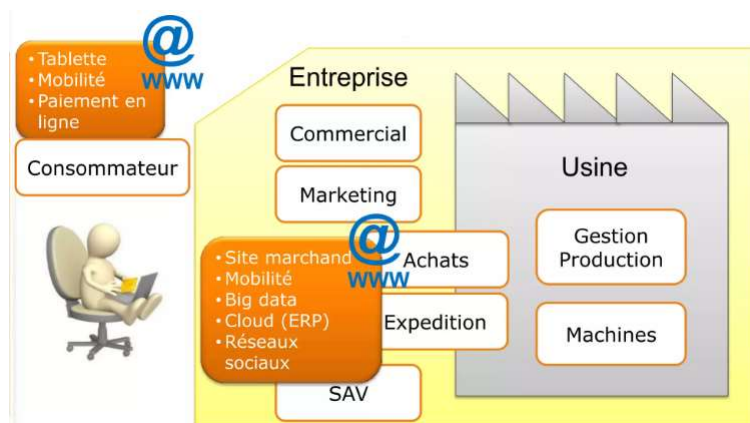
Robot spécialisé dans la reconnaissance et le tri des déchets. Son fonctionnement repose sur l'intelligence artificielle (*machine learning*). Une analyse en temps réel des déchets et une prise de décision rapide permettent à ce robot de trier avec une cadence soutenue.
 Ces données permettent également de faire des statistiques par classe de déchets.
 En France, dans le milieu de la construction, la Fédération professionnelle des entreprises du recyclage (Federec) a lancé la première édition des « Trophées de l'innovation » du recyclage. À suivre...

			
st Brick		Komatsu	MAX AI traitement
e	Optimisation	Autonomie	

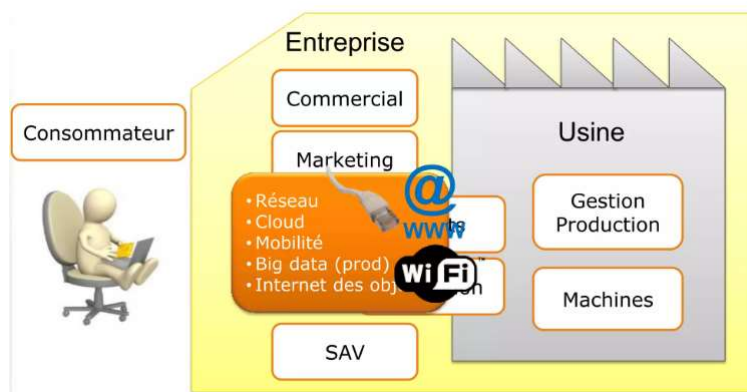
Usine du futur dans le monde

- France 34 usine de futur
- Allemagne: plan "industrie 4.0"
- États-Unis: plan "advanced manufacturing"
- Europe: Appel à projet "horizon 2020 usine de futur"

Technologies "de l'Internet" entrent dans l'usine



Technologies "de l'Internet" entrent dans l'usine



Plan de Cours

1. Projets réalisés
2. Introduction
2.1 IIOT

- Introduction à l'IIOT
- Mise en avant de l'IIOT
- Potentiel de Croissance de l'IIOT
- Défis de l'Adoption de l'IIOT
- Technologies Clés de l'IIoT
- Avantages et Opportunités

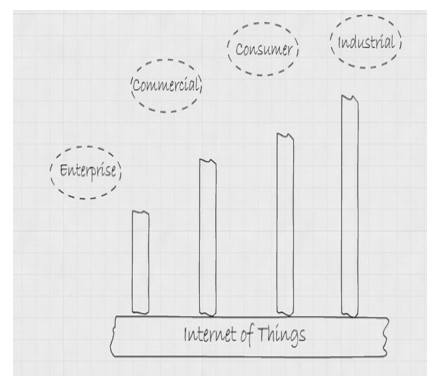
Introduction à l'IIOT:

- GE (General Electric) a été un précurseur de l'Internet Industriel, mais il existe d'autres termes comme "Internet de Tout" et "Internet 4.0" pour décrire des concepts similaires.

Introduction à l'IIOT:

- Il est essentiel de faire la distinction entre :

- Stratégies IoT verticales, qui ciblent des marchés spécifiques comme les consommateurs, le commerce et l'industrie.
- Concept horizontal plus large de l'Internet des Objets (IoT).



Mise en avant de l'IIOT

- L'Internet Industriel des Objets (IIoT) englobe une multitude de domaines, allant de la production d'énergie à l'agriculture, en passant par les soins de santé et la fabrication.
- Cette diapositive met l'accent sur l'IIoT en tant que domaine d'intérêt principal et montre comment il peut optimiser des processus complexes, comme la gestion de la chaîne d'approvisionnement dans la fabrication.

Potentiel de Croissance de l'IIOT

- Les prévisions indiquent une croissance significative de l'IIoT, avec une valeur attendue de 151,01 milliards de dollars US d'ici 2020, grâce à un taux de croissance annuel composé de 8,03 %.
- Les gouvernements et les entreprises voient dans l'IIoT une opportunité de réindustrialisation, comme le montre l'exemple de l'industrie manufacturière aux États-Unis, qui rapatrie des opérations précédemment délocalisées.

Défis de l'Adoption de l'IloT

- L'adoption de l'IloT a été lente en raison des incertitudes concernant ses impacts sur les industries existantes, les chaînes de valeur, les modèles commerciaux et les effectifs.
- En 2015, la majorité des leaders de l'industrie n'avaient pas encore une vision claire des modèles commerciaux et des technologies de l'IloT, comme illustré par la réticence dans le secteur de l'automobile.

Technologies Clés de l'IloT

- L'IloT repose sur l'intégration de capteurs pour collecter des données, de middleware pour les traiter, de logiciels pour l'analyse, et de calcul en nuage et de stockage pour la gestion des données massives.
- L'IloT permet de transformer les processus opérationnels en utilisant ces données pour améliorer l'efficacité, comme dans le cas de l'optimisation de la production d'énergie éolienne grâce à des capteurs IoT.

Avantages et Opportunités:

- La "Puissance de 1%" montre que de petites améliorations de seulement 1 % peuvent avoir un impact significatif dans divers secteurs, tels que l'aviation, l'énergie, la logistique, et les soins de santé.
- Les entreprises passent de la vente de produits à la vente de résultats ou de services, comme l'exemple des fabricants de pneus qui facturent en fonction de l'usure réelle des pneus, ce qui profite à la fois aux clients et aux fabricants grâce à la collecte de données IoT.

Plan de Cours

1. Projets réalisés
2. Introduction
3. Le marché de l'IOT

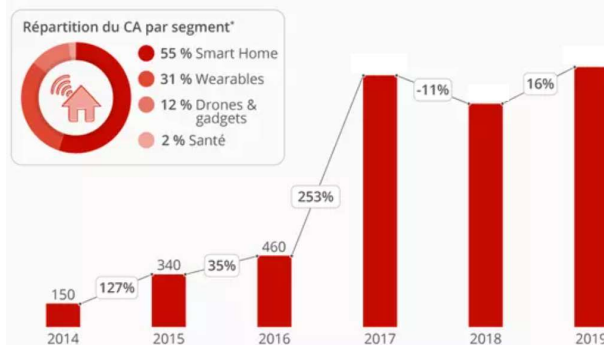
- **Tendance de croissance du marché de l'IoT**
- **Facteurs de croissance de l'IoT**
- **Secteurs d'application de l'IoT dans l'industrie**
- **Exemples de cas d'utilisation dans l'industrie**
- **Avantages économiques de l'IoT**
- **Exemples de retours sur investissement**
- **Défis et opportunités du marché de l'IoT**

Tendance de croissance du marché de l'IoT (1/2)

L'Internet des Objets est en train de révolutionner notre façon de vivre et de travailler. L'une des preuves les plus manifestes de cette révolution est l'expansion rapide du marché de l'IoT.

Objets connectés : zoom sur le marché BtoC français

Chiffre d'affaires du marché des objets connectés en France, en millions d'euros



Courbe qui reflète la croissance constante du marché de l'IoT au fil des ans. Cette tendance témoigne de l'adoption généralisée de l'IoT dans de nombreux secteurs.

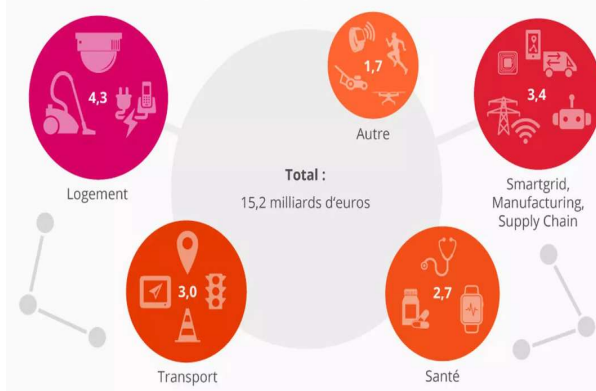
Tendance de croissance du marché de l'IoT (2/2)

Le domaine de l'IoT continue de croître sans relâche. En France, il est prévu qu'il génère un chiffre d'affaires de **15,2 milliards d'euros en 2020**, d'après les données de l'Institut Montaigne et du cabinet de conseil américain A.T. Kearney.

Domotique devrait générer 4,3 milliards d'euros de revenus, les transports 3 milliards et la santé 2,7 milliards.

Le marché des objets connectés va s'envoler en France d'ici 2020

Estimation du marché des objets connectés par secteur (en milliards d'euros)



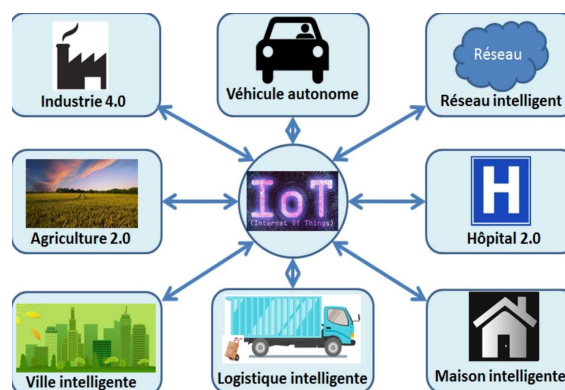
Facteurs de croissance de l'IoT (1/1)

Liste de facteurs :

- 1.Connectivité** : L'expansion des réseaux de communication, notamment la 5G, favorise l'interconnexion des appareils IoT à une vitesse fulgurante.
- 2.Réduction des coûts** : La diminution des coûts de fabrication des composants IoT et des services associés rend l'IoT plus accessible aux entreprises et aux consommateurs.
- 3.Adoption généralisée** : L'adoption rapide de l'IoT dans divers secteurs, des soins de santé à l'agriculture, contribue à sa croissance exponentielle.

Secteurs d'application de l'IoT dans l'industrie (1/2)

Le monde industriel est vaste et complexe, avec de nombreux secteurs qui bénéficient de l'Internet des IoT. Dans cette carte, nous explorerons comment l'IoT trouve des applications novatrices dans des domaines tels que la fabrication, la logistique, la santé, la chaîne d'approvisionnement, l'agriculture, et bien d'autres encore.

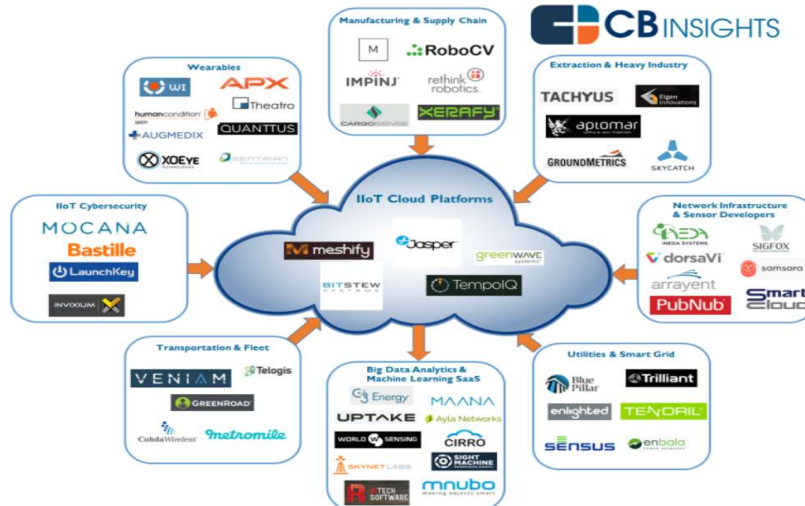


Chaque secteur est un maillon essentiel de la chaîne qui contribue à l'essor de l'IoT.

Secteurs d'application de l'IoT dans l'industrie (2/2)

The Industrial IoT (IIoT) Market Map

Created By



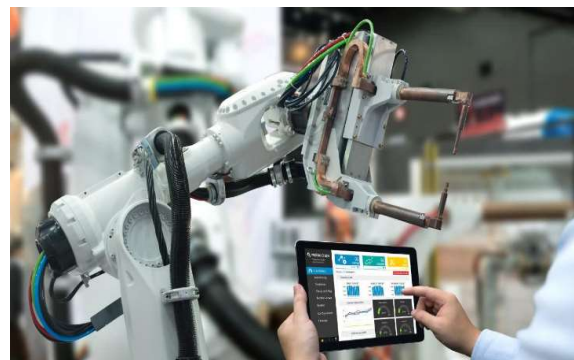
Révolution Exemples de cas d'utilisation dans l'industrie (1/4)

La convergence de (IoT) et de l'industrie a initié une véritable révolution, transformant la façon dont les entreprises opèrent.

Secteur 1 : Fabrication intelligente

Dans le domaine de la fabrication, les capteurs IoT surveillent en temps réel les machines et les processus de production.

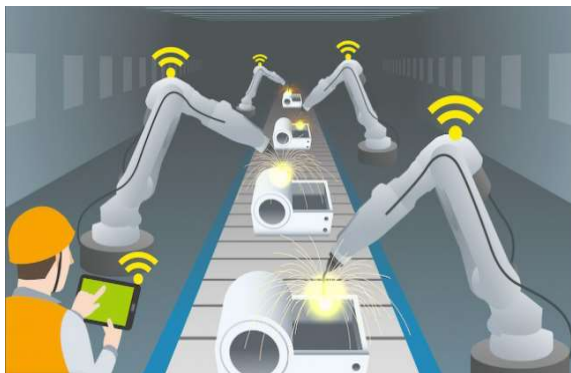
Cela permet une maintenance prédictive, une optimisation de la production et la réduction des temps d'arrêt non planifiés.



Révolution Exemples de cas d'utilisation dans l'industrie (2/4)

Secteur 2 : Logistique et chaîne d'approvisionnement

Dans la logistique, l'IoT permet un suivi en temps réel des marchandises et des véhicules, améliorant l'efficacité opérationnelle, réduisant les coûts et garantissant une livraison plus précise.



Révolution Exemples de cas d'utilisation dans l'industrie (3/4)

Secteur 3 : Santé et soins médicaux

Dans le secteur de la santé, les dispositifs médicaux IoT surveillent les patients à distance, facilitant le suivi des conditions médicales, la réduction des hospitalisations inutiles et l'amélioration des soins de santé.



Révolution Exemples de cas d'utilisation dans l'industrie (4/4)

Secteur 4 : Agriculture intelligente

Dans l'agriculture, les capteurs IoT permettent de surveiller les conditions météorologiques, le sol et la croissance des cultures, optimisant ainsi l'utilisation des ressources, la production agricole et la durabilité,



Avantages économiques de l'IoT (1/1)

Internet des Objets (IoT) offre une multitude d'avantages économiques aux entreprises comme :

L'amélioration de l'efficacité opérationnelle, réduit les coûts de maintenance, minimise les temps d'arrêt, optimise la gestion des stocks et de la chaîne d'approvisionnement, ce qui se traduit par une amélioration de la rentabilité.

Exemples de retours sur investissement (1/3)

L'Internet des Objets (IoT) n'est pas seulement une technologie fascinante, c'est aussi un moteur puissant de retours sur investissement pour les entreprises. Dans cette diapositive, nous explorerons des exemples concrets de la manière dont l'IoT a apporté des avantages financiers substantiels.

Exemples de retours sur investissement (2/3)

Exemple 1 : Réduction des coûts de maintenance

Une entreprise de fabrication a déployé des capteurs IoT pour surveiller l'état de ses machines. Résultat : une réduction de 30 % des coûts de maintenance, car les pannes sont détectées avant qu'elles ne deviennent coûteuses.



Exemples de retours sur investissement (3/3)

Exemple 2 : Augmentation de la productivité

Une société logistique a utilisé l'IoT pour optimiser le suivi des camions et la gestion des itinéraires. Résultat : une augmentation de 15 % de la productivité, avec moins de temps perdu en attente ou en itinérance.

Exemple 3 : Services IoT générant des revenus

Une entreprise d'électroménagers a créé des produits IoT connectés, ce qui a ouvert la voie à des services d'abonnement payants. Résultat : une augmentation de 20 % des revenus récurrents.

Défis et opportunités du marché de l'IoT (1/2)

Le marché de l'Internet des Objets (IoT) continue de croître, il est essentiel de comprendre les défis qui se posent et les opportunités qu'il offre.

Liste des défis :

1. **Sécurité et confidentialité** : La protection des données IoT contre les cyberattaques est une préoccupation majeure.
2. **Interopérabilité** : Assurer que les dispositifs IoT provenant de différents fabricants peuvent fonctionner ensemble est un défi.
3. **Évolutivité** : Les réseaux IoT doivent évoluer pour gérer un nombre croissant d'appareils.
4. **Gestion des données** : Le traitement des vastes quantités de données générées par l'IoT est un défi en soi.

Défis et opportunités du marché de l'IoT (2/2)

Liste des opportunités du marché de l'IoT

1. **Nouveaux marchés** : L'IoT ouvre de nouvelles opportunités commerciales dans des secteurs tels que la santé, l'agriculture intelligente et la ville intelligente.
2. **Innovation** : L'IoT stimule l'innovation en permettant de nouvelles idées et de nouveaux produits.
3. **Amélioration de l'efficacité** : Les entreprises peuvent améliorer l'efficacité opérationnelle grâce à l'IoT.
4. **Durabilité** : L'IoT peut contribuer à des pratiques plus durables en réduisant les déchets et en optimisant l'utilisation des ressources.

Plan de Cours

1. Projets réalisés
2. Introduction
3. Le marché de l'IOT
4. Interactions entre le « monde numérique » et le « monde physique »

- L'importance de la convergence
- Capteurs et dispositifs connectés
- Analyse des données en temps réel
- Exemples d'intégration IoT

Interactions entre le "monde numérique" et le "monde physique"

Cette section explore l'importance de la convergence entre le monde numérique et le monde physique dans l'industrie du futur. Nous examinerons comment cette fusion crée de la valeur et comment *les capteurs, les dispositifs connectés et l'analyse des données en temps réel y contribuent.*

L'importance de la convergence (1/1)

1. la nécessité de fusionner les mondes numérique et physique pour l'industrie du futur:

L'industrie du futur repose sur la convergence du monde numérique et du monde physique. Cela signifie que les données provenant du monde physique (capteurs, machines, environnement) sont capturées, transformées en données numériques et analysées pour prendre des décisions éclairées.

2. Exemples de cas d'utilisation :

L'intégration réussie de ces deux mondes crée de la valeur. Par exemple, dans l'agriculture, des capteurs mesurent l'humidité du sol, les niveaux d'irrigation sont ajustés en temps réel, améliorant ainsi les rendements et économisant de l'eau.

Capteurs et dispositifs connectés (1/1)

1. Types de capteurs et de dispositifs IoT utilisés dans l'industrie:

Les capteurs IoT incluent des capteurs de température, d'humidité, de mouvement, de pression, etc. Les dispositifs connectés vont des caméras aux dispositifs de suivi de la santé personnelle.

2. Comment ces capteurs collectent des données du monde physique:

Les capteurs recueillent des données en mesurant des paramètres physiques du monde réel. Par exemple, un capteur de température collecte des données sur la chaleur ambiante.

Analyse des données en temps réel (1/1)

L'analyse en temps réel permet de détecter les tendances, les anomalies et les opportunités immédiatement, ce qui améliore la prise de décision et la réactivité aux événements en temps réel. Par exemple, dans la logistique, elle permet d'optimiser les itinéraires en fonction du trafic actuel.

Exemples d'intégration IoT

(1/1)

Des exemples de l'intégration IoT incluent des systèmes de production intelligents qui ajustent automatiquement la production en fonction de la demande, ainsi que des dispositifs de maintenance prédictive qui identifient les besoins de maintenance avant qu'une panne ne se produise, réduisant ainsi les temps d'arrêt.

Plan de Cours

1. Projets réalisés
 2. Introduction
 3. Le marché de l'IOT
 4. Interactions entre le « monde numérique » et le « monde physique »
 5. Principes fondamentaux de l'IoT
- Fonctionnement de l'IoT
 - L'importance de la convergence
 - Interconnexion des objets
 - Modèle en couches de l'OSI
 - Modèle en couches de l'IoT
 - Comparaison entre le modèle OSI et le modèle de l'IoT

Principes fondamentaux de l'IoT

Mr. MEGNAFI Hicham (ESSA-Tlemcen)

I. Fonctionnement de l'IoT

II. Interconnexion des objets

- A. Comment les objets communiquent entre eux
- B. Mise en réseau des objets
- C. Collecte et transmission de données

III. Modèle en couches de l'OSI

- A. Présentation du modèle OSI (Open Systems Interconnection)
- B. Les sept couches du modèle OSI et leurs rôles
- C. Comment le modèle OSI définit les communications en couches

VI. Modèle en couches de l'IoT

- A. Structure du modèle en couches de l'IoT
- B. Rôles et responsabilités des couches de l'IoT
- C. Adaptation aux besoins spécifiques de l'IoT

V Comparaison entre le modèle OSI et le modèle de l'IoT

- A. Similitudes entre les deux modèles
 - Hiérarchie en couches
 - Gestion des communications
 - Niveaux de connectivité

B. Différences entre les deux modèles

- Adaptation aux besoins de l'IoT
- Rôle de la couche physique
- Intégration de l'Edge Computing
- Gestion spécifique des capteurs et actuateurs
- Utilisation de protocoles de communication IoT (MQTT, CoAP)

C. Avantages et inconvénients de chaque modèle

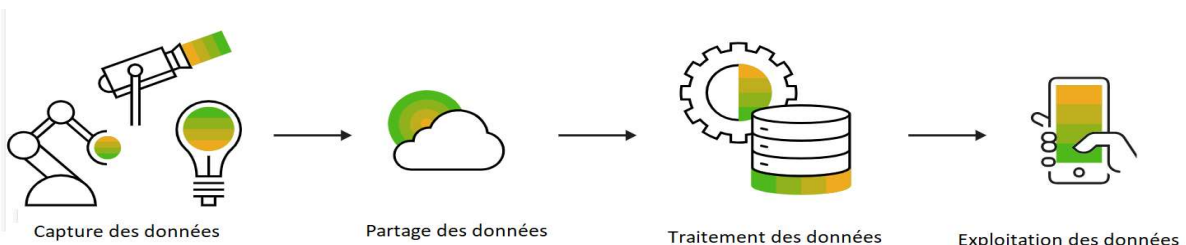
- Avantages du modèle OSI dans un contexte plus large
- Avantages du modèle de l'IoT pour les besoins spécifiques de l'IoT
- Limitations de chaque modèle

Mr. MEGNAFI Hicham (ESSA-Tlemcen)

I. Fonctionnement de l'IoT

(1/3)

Les terminaux IoT agissent comme nos représentants à distance, capturant les données selon leur programmation. Ces données sont ensuite collectées et analysées pour orienter nos décisions futures et automatiser des actions.



Ce processus se décompose en quatre étapes essentielles : capture, collecte, analyse, et action.

I. Fonctionnement de l'loT (2/3)

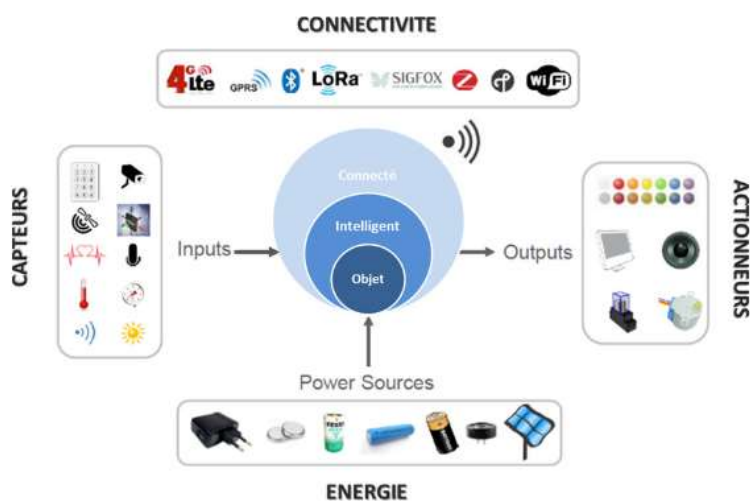
1. Capture des données : Les capteurs des terminaux IoT collectent des informations environnementales, qu'il s'agisse de données simples telles que la température ou de flux vidéo plus complexes,

2. Partage des données : Les terminaux IoT utilisent des connexions réseau pour transmettre ces données vers un système cloud, d'autres terminaux ou les stockent localement en vue d'un traitement ultérieur.

3. Traitement des données : Un logiciel spécifique réagit aux données collectées, déclenchant des actions comme l'activation de dispositifs ou l'envoi d'alertes.

4. Exploitation des données : Les données collectées de l'ensemble du réseau IoT sont analysées pour fournir des informations pertinentes, facilitant la prise de décisions éclairées et la mise en œuvre d'actions adaptées.

I. Fonctionnement de l'loT (3/3)



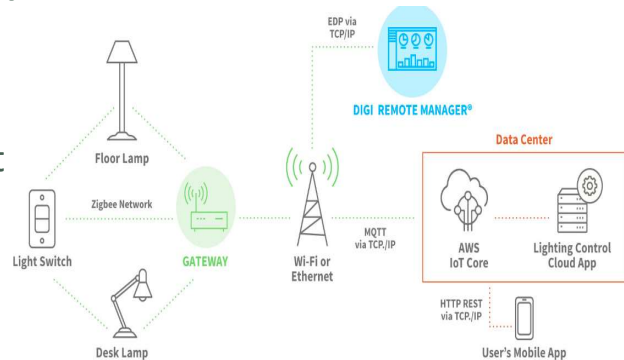
II. Interconnexion des objets

(1/5)

A. Comment les objets communiquent entre eux

Communication entre Objets:

- Les objets interconnectés utilisent différents protocoles de communication.
- Communication sans fil (Wi-Fi, Bluetooth, Zigbee) et/ou filaire.
- Échange de données, commandes et informations.



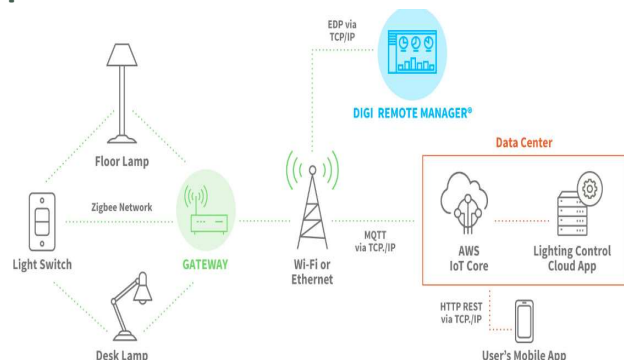
II. Interconnexion des objets

(2/5)

A. Comment les objets communiquent entre eux

Internet des Objets (IoT):

- L'IoT permet la connexion des objets à Internet pour une communication à grande échelle.
- Les objets peuvent être contrôlés et surveillés à distance.



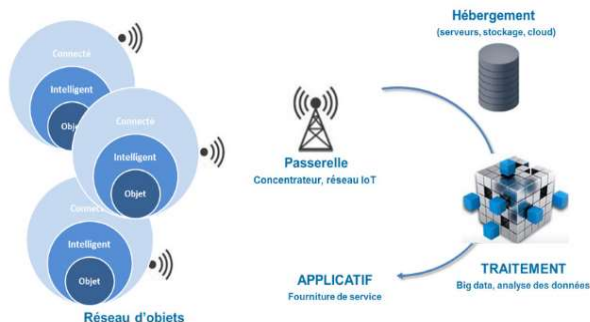
II. Interconnexion des objets

(3/5)

B. Mise en réseau des objets

Réseaux d'Objets:

- Les objets sont organisés en réseaux pour une meilleure gestion.
- Topologies de réseau : étoile, maillée arborescente, etc.
- Avantages de la mise en réseau : partage de données, redondance, efficacité



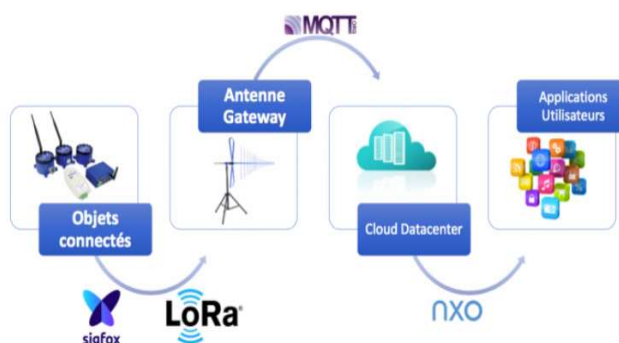
II. Interconnexion des objets

(4/5)

B. Mise en réseau des objets

Protocoles de Réseau

- Protocoles spécifiques à l'IoT : MQTT, CoAP, LoRaWAN, etc.
- Assurent une communication efficace entre les objets



107

Mr. MEGNAFI Hicham (ESSA-Tlemcen)

II. Interconnexion des objets

(5/5)

C. Collecte et Transmission de Données

Collecte de Données

- Les objets IoT recueillent des données à partir de capteurs.
- Informations sur l'environnement, les conditions, les performances, etc.

Transmission de Données

- Les données collectées sont transmises vers des destinations appropriées.
- Les données peuvent être stockées localement ou dans le cloud.
- Utilisation de méthodes de transmission sécurisée.

108

Mr. MEGNAFI Hicham (ESSA-Tlemcen)

III. Modèle en couches de l'OSI

(1/6)

A. Présentation du modèle OSI (Open Systems Interconnection)

Le Modèle OSI

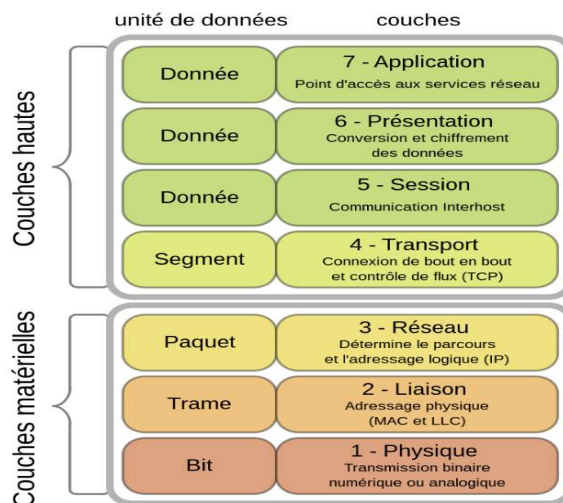
- Conçu pour standardiser les communications en réseau.
- Développé par l'ISO (International Organization for Standardization) dans les années 1980.

Pourquoi le Modèle OSI ?

- Comprendre l'architecture des réseaux.
- Facilite l'interopérabilité entre différents équipements et fabricants.
- Un référentiel pour le développement et la gestion de réseaux.

III. Modèle en couches de l'OSI (2/6)

B. Les sept couches du modèle OSI et leurs rôles



III. Modèle en couches de l'OSI (3/6)

B. Les sept couches du modèle OSI et leurs rôles



- Transmission des bits sur le support physique.
- Exemples : câbles, signaux électriques.

Couche Liaison de données :

- Gestion de la fiabilité de la communication.
- Exemples : Ethernet, Wi-Fi.

III. Modèle en couches de l'OSI

(4/6)

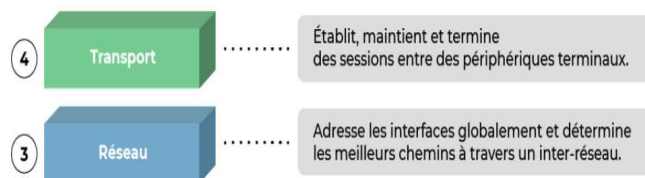
B. Les sept couches du modèle OSI et leurs rôles

Couche Réseau :

- Routage des données à travers le réseau.
- Exemples : IPv4, IPv6.

Couche Transport :

- Contrôle du flux et de la fiabilité de bout en bout.
- Exemples : TCP, UDP.



III. Modèle en couches de l'OSI

(5/6)

B. Les sept couches du modèle OSI et leurs rôles

Couche Session :

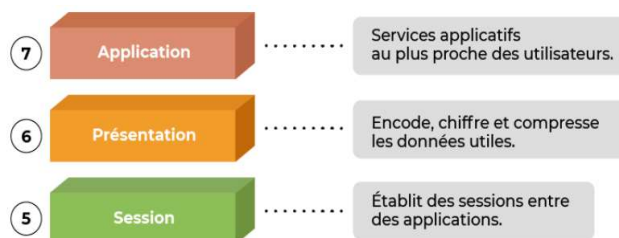
- Établissement, maintenance et fermeture de sessions.
- Gestion des dialogues.

Couche Présentation :

- Translation des données entre les formats.
- Compression, chiffrement.

Couche Application :

- Interaction avec les applications utilisateur.
- Exemples : HTTP, FTP, SMTP.



III. Modèle en couches de l'OSI (6/6)

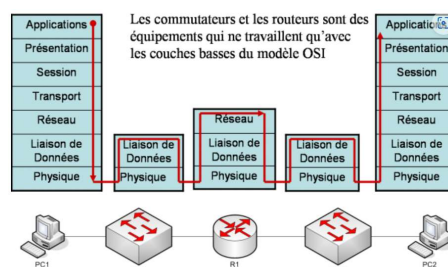
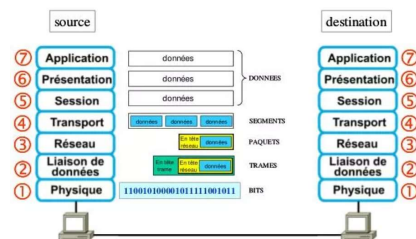
C. Comment le modèle OSI définit les communications en couches

Communication en couches

- Chaque couche a des fonctions spécifiques et communique avec les couches adjacentes.
- Les données sont transmises de haut en bas (couches supérieures vers couches inférieures) et vice versa.
- Chaque couche ajoute des informations de contrôle (en-têtes) pour gérer la transmission.

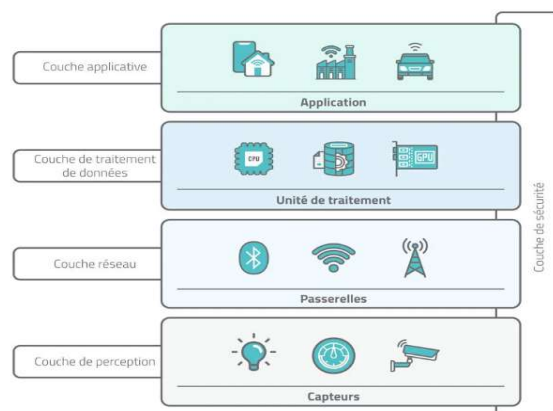
Avantages de l'approche en couches

- Modularité : Facilite la conception, le dépannage et la maintenance des réseaux.
- Interopérabilité : Différents fabricants peuvent mettre en œuvre des couches spécifiques.
- Évolutivité : Possibilité d'ajouter de nouvelles fonctionnalités sans perturber les autres couches.



IV. Modèle en couches de l'IoT (1/8)

A. Structure du modèle en couches de l'IoT



Bien que l'Internet des objets (IoT) ne suive pas une seule architecture universelle, le format le plus courant et généralement reconnu est une architecture en 5 couches pour l'IoT.

IV. Modèle en couches de l'IoT

(2/8)

B. Rôles des couches de l'IoT

1. Couche de perception



1. Les objets physiques de l'IoT, au sein de cette couche, collectent et traitent des informations de base, fournissant des données telles que la position, la température, le poids, le mouvement, etc.

2. La couche assure la collecte et la numérisation des données environnementales, puis les transmet de manière sécurisée à la couche supérieure.

IV. Modèle en couches de l'IoT

(3/8)

B. Rôles des couches de l'IoT

2. Couche réseau



1. Couche réseau gère le transport des données vers le centre de traitement.
2. Utilise des moyens de transmission sans fil.
3. Les principales technologies de communication employées incluent la 3G, le WiFi, le ZigBee, etc.
4. Les protocoles de communication essentiels, tels que 6LowPan, se situent au sein de cette couche pour l'adressage des millions d'objets connectés.

IV. Modèle en couches de l'IoT

(4/8)

B. Rôles des couches de l'IoT

3. Couche traitement (Middleware)



1. Gère les services offerts par chaque objet IoT.
2. Relie ces services aux informations collectées et les stocke dans des bases de données.
3. Applique des traitements et des calculs pour prendre des décisions automatiques.
4. Elle permet aux développeurs d'applications IoT d'accéder aux services sans se soucier de l'interopérabilité des objets ou d'une plateforme matérielle spécifique,

IV. Modèle en couches de l'IoT

(5/8)

B. Rôles des couches de l'IoT

4. Couche application



1. Permet l'utilisation des informations traitées par la couche de traitement et des services des objets IoT.
2. Facilite le développement d'applications IoT variées à partir de ces données et services.
3. Ces applications sont destinées à être directement utilisées par les utilisateurs finaux.

IV. Modèle en couches de l'IoT

(6/8)

B. Rôles des couches de l'IoT

5. Couche de sécurité

1. **Authentification et Chiffrement** : Authentification des utilisateurs et dispositifs, ainsi que le chiffrement des données pour la confidentialité.
2. **Contrôle d'Accès** : Gestion des accès en fonction des rôles et privilèges.
3. **Surveillance des Menaces** : Détection des activités suspectes et réaction rapide aux menaces.
4. **Sécurité Physique** : Mesures pour protéger les dispositifs IoT sur le plan physique.

IV. Modèle en couches de l'IoT

(7/8)

C. Adaptation aux besoins spécifiques de l'IoT

L'architecture IoT est conçue pour répondre aux besoins spécifiques comme:

1. **Économie d'Énergie** : Elle optimise la consommation d'énergie des dispositifs IoT, permettant des opérations prolongées avec des batteries.
2. **Faible Latence** : Elle minimise la latence pour répondre rapidement aux besoins des applications en temps réel.
3. **Évolutivité** : Elle gère efficacement un grand nombre d'objets IoT, garantissant des performances optimales malgré l'expansion constante.
4. **Sécurité Renforcée** : Elle intègre des mécanismes de sécurité avancés pour protéger les données et les dispositifs contre les menaces.

IV. Modèle en couches de l'loT (8/8)

C. Adaptation aux besoins spécifiques de l'loT

- 5. Interopérabilité** : Elle favorise l'interopérabilité entre dispositifs provenant de divers fabricants et utilisant différents protocoles.
- 6. Traitement en Périphérie (Edge Computing)** : Elle permet aux dispositifs IoT de prendre des décisions localement, réduisant la dépendance au cloud.
- 7. Collecte et Analyse de Données** : Elle facilite la collecte et l'analyse efficace des données IoT pour en tirer des informations exploitables.

V. Comparaison entre le modèle OSI et de l'loT (1/3)

A. Similitudes entre les deux modèles

- 1. Hiérarchie en couches** : Les deux modèles sont basés sur une hiérarchie en couches pour organiser les fonctions de communication.
- 2. Gestion des communications** : Ils impliquent une gestion des communications pour assurer la fiabilité, la sécurité et l'efficacité des échanges de données.
- 3. Niveaux de connectivité** : Les deux modèles définissent différents niveaux de connectivité pour gérer les aspects techniques et fonctionnels des réseaux.

V. Comparaison entre le modèle OSI et de l'IoT (2/3)

B. Différences entre les deux modèles

1. **Adaptation aux besoins de l'IoT:** Le modèle de l'IoT est spécifiquement adapté aux besoins des objets connectés, tels que la gestion des ressources limitées.
2. **Rôle de la couche physique:** La couche physique dans l'IoT est souvent dépendante du type de capteurs et d'actuateurs, contrairement au modèle OSI.
3. **Intégration de l'Edge Computing:** L'Edge Computing est un concept intégré dans le modèle de l'IoT pour le traitement local des données, ce qui n'est pas présent dans le modèle OSI.
4. **Gestion spécifique des capteurs et actuateurs:** Le modèle de l'IoT intègre des mécanismes de gestion spécifiques pour les capteurs et les actuateurs.

V. Comparaison entre le modèle OSI et de l'IoT (3/3)

C. Avantages et inconvénients de chaque modèle

	Modèle OSI	Modèle de l'IoT
Avantages	<ul style="list-style-type: none">➤ Le modèle OSI est plus générique et convient à un large éventail d'applications de réseau.➤ Il offre une base solide pour les infrastructures de communication.	<ul style="list-style-type: none">➤ Le modèle de l'IoT optimise les ressources et répond aux besoins uniques de l'IoT.➤ Il permet une gestion efficace des dispositifs connectés.
Limitations	<ul style="list-style-type: none">➤ Le modèle OSI peut sembler trop complexe pour les déploiements IoT à petite échelle.	<ul style="list-style-type: none">➤ Le modèle de l'IoT peut être moins adaptable à des cas d'utilisation plus vastes en dehors de l'IoT.

Plan de Cours

1. Projets réalisés
2. Introduction
3. Le marché de l'IOT
4. Interactions entre le « monde numérique » et le « monde physique »
5. Principes fondamentaux de l'IoT
6. Composants de base de l'IoT

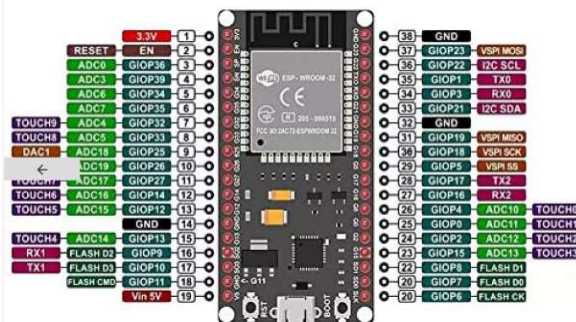
- **Node MCU ESP32**
- **Capteurs**
 - Rôle des capteurs dans l'IoT
 - Types de capteurs (température, mouvement, luminosité, etc.)
- **Actuateurs**
 - Rôle des actuateurs
 - Exemples d'actuateurs (moteurs, relais, etc.)
- **Raspberry pi 3P**

I. NODE MCU ESP32

(1/1)

ESP32 WIFI

Connectivité sans fil	-WiFi: 150.0 Mbps débit de données avec HT40 -Bluetooth: BLE (Bluetooth Low Energy) et Bluetooth Classic
Processeur	Microprocesseur LX6 double cœur 32 bits, fonctionnant à 160 ou 240 MHz
Mémoire	ROM : 448 Ko, SRAM : 520 Ko
Entrée/sortie périphérique	-PWM (modulation de largeur d'impulsion). -ADC (convertisseur analogique-numérique) et DAC -I ² C (circuit inter-intégré) -UART (Récepteur/Transmetteur Asynchrone Universel) -SPI (interface périphérique série) -I ² S (Interchip Sound intégré) -interface périphérique avec DMA qui inclut le tactile capacitif
Compatible Arduino IDE	vous pouvez programmer l'ESP32 avec l'IDE Arduino (instructions d'installation Windows, Mac OS X et Linux)
Compatible avec MicroPython	vous pouvez programmer l'ESP32 avec le firmware MicroPython (Prise en main de MicroPython sur ESP32)



127

Mr. MEGNAFI Hicham (ESSA-Tlemcen)

II. Types de Capteurs

(1/10)

Rôle des capteurs dans l'IoT

- Les capteurs collectent des données du monde physique.
- Ils transforment des phénomènes en signaux numériques.
- Les types de capteurs incluent température, mouvement, lumière, gaz, etc.
- Les données des capteurs alimentent les systèmes IoT.
- Les capteurs sont essentiels pour surveiller, contrôler et automatiser des processus.
- Ils sont au cœur de l'Internet des Objets (IoT).

1. Capteur de Température et d'Humidité DHT22 (ou DHT11)
2. Capteur de Mouvement PIR (Passive Infrared)
3. Capteur de Luminosité (LDR - Light Dependent Resistor)
4. Capteur de Gaz MQ-2
5. Capteur d'Humidité du Sol (capteur d'humidité du sol FC-28)

128

Mr. MEGNAFI Hicham (ESSA-Tlemcen)

II. Types de Capteurs

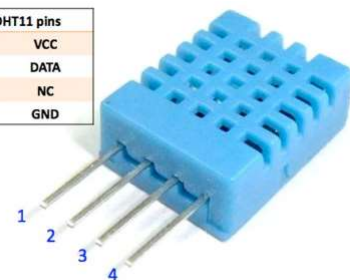
(1/10)

A. Capteur de Température et d'Humidité DHT22 (ou DHT11)

Mesure la température et l'humidité de l'environnement.

- Mesure précise de la température (-40°C à 80°C pour DHT11, -40°C à 125°C pour DHT22).
- Surveillance de l'humidité relative (0% à 100%).
- Deux versions : DHT22 (AM2302) et DHT11.
- Sortie numérique pour une intégration facile.
- Faible consommation d'énergie, idéale pour l'IoT.
- Applications : régulation du climat, agriculture, domotique.

DHT11 pins	
1	VCC
2	DATA
3	NC
4	GND



II. Types de Capteurs

(2/10)

A. Capteur de Température et d'Humidité DHT22 (ou DHT11)

Matériel nécessaire :

- NodeMCU ESP32.
- Capteur de Température et d'Humidité DHT22 (ou DHT11).

Câblage :

- Connectez l'alimentation du capteur à la broche 3.3V du NodeMCU ESP32.
- Connectez la masse du capteur à la broche GND du NodeMCU ESP32.
- Connectez le signal du capteur à une broche GPIO du NodeMCU ESP32 (par exemple, D2).

Code Arduino :

- Vous aurez besoin de la bibliothèque DHT pour lire les données du capteur.

```
#include <DHT.h>
#define DHTPIN D2 // La broche à laquelle le capteur est connecté
#define DHTTYPE DHT22 // Utilisez DHT22 si vous avez le DHT22
DHT dht(DHTPIN, DHTTYPE);
void setup() {
  Serial.begin(115200);
  dht.begin();
}
void loop() {
  delay(2000); // Attendre quelques secondes entre les mesures
  float humidite = dht.readHumidity();
  float temperature = dht.readTemperature();
  if (isnan(humidite) || isnan(temperature)) {
    Serial.println("Échec de la lecture du capteur !");
  } else {
    Serial.print("Humidité : ");
    Serial.print(humidite);
    Serial.print(" %\n");
    Serial.print("Température : ");
    Serial.print(temperature);
    Serial.println(" °C");
  }
}
```

II. Types de Capteurs

(3/10)

B. Capteur de Mouvement PIR (Passive Infrared)

Détecte les mouvements humains en mesurant les changements de chaleur infrarouge.

- Tension de fonctionnement: DC 5V-20V
- Consommation électrique statique: 65 microampères
- Niveau de sortie: haut 3.3V, bas 0V
- Temps de retard: réglable (0,3 seconde à 10 minutes)
- Temps de blocage: 0,2 seconde
- Déclencheur: L ne peut pas être répété, H peut être répété, la valeur par défaut est H
- Plage d'induction: angle de cône inférieur à 120 degrés, inférieur à 7 mètres
- Température de travail: -15 -70



II. Types de Capteurs

(4/10)

B. Capteur de Mouvement PIR (Passive Infrared)

Matériel nécessaire :

- NodeMCU ESP32.
- Capteur de Mouvement PIR,

Câblage :

- Connectez la broche de signal du capteur PIR à une broche numérique de votre NodeMCU ESP32, par exemple, D2.
- Connectez la masse du capteur à la broche GND du NodeMCU ESP32.
- Connectez la broche d'alimentation (VCC) du capteur PIR à une broche 3.3V du NodeMCU ESP32.

```
#define PIR_PIN D2 // Broche de signal du capteur PIR

void setup() {
  Serial.begin(115200);
  pinMode(PIR_PIN, INPUT);
  // Configure la broche du capteur PIR comme une entrée.
}

void loop() {
  int motionState = digitalRead(PIR_PIN);
  // Lis l'état du capteur PIR (HAUT ou BAS)

  if (motionState == HIGH) {
    Serial.println("Mouvement détecté !");
    // Ajoutez ici les actions à effectuer en cas de détection de mouvement
  } else {
    Serial.println("Pas de mouvement.");
    // Ajoutez ici les actions à effectuer lorsque le mouvement s'arrête
  }
  delay(1000);
  // Attendez quelques secondes entre les lectures pour éviter
  // les détections multiples
}
```

II. Types de Capteurs

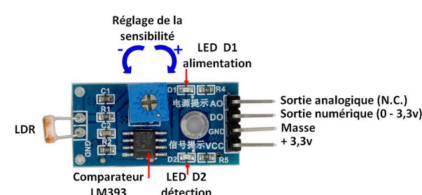
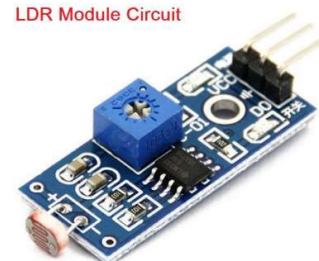
(5/10)

C. Capteur de Luminosité (LDR - Light Dependent Resistor)

Détecte Mesure la luminosité ambiante.

- Puissance maximale: 100mW
- Température de fonctionnement: -30 ° C ~ 70 ° C
- Pic spectral: 540nm
- Résistance lumineuse (10Lux) (K Ω): 10 - 20
- Résistance foncée: 1 M Ω
- 100 λ 10: 0,6
- Temps de réponse: 20ms (Rise), 30ms (Down)
- Éclairage de résistance: 3

LDR Module Circuit



II. Types de Capteurs

(6/10)

C. Capteur de Luminosité (LDR - Light Dependent Resistor)

```
const int ldrPin = D2; // Broche numérique pour le capteur LDR

void setup() {
  Serial.begin(115200);
  pinMode(ldrPin, INPUT);
}

void loop() {
  int luminosite = digitalRead(ldrPin); // Lit l'état de la broche
  Serial.print("Luminosité : ");
  Serial.println(luminosite);

  delay(1000); // Attendez quelques secondes entre les lectures
}
```

```
#define LDR_PIN A0 // Broche analogique pour le capteur LDR

void setup() {
  Serial.begin(115200);
}

void loop() {
  int luminosite = analogRead(LDR_PIN);
  // Lit la valeur analogique du capteur LDR
  Serial.print("Luminosité : ");
  Serial.println(luminosite);

  delay(1000); // Attendez quelques secondes entre les lectures
}
```

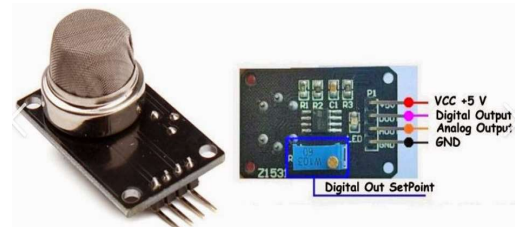
II. Types de Capteurs

(7/10)

D. Capteur de Gaz MQ-2

Détecte divers gaz, y compris le méthane, le butane, le propane, le monoxyde de carbone et l'alcool.

- Alimentation: 5 Vcc
- Plage de détection: 100-10000 ppm
- Sortie analogique et numérique
- Température de fonctionnement : -10 °C à +50 °C



La tension de sortie analogique du module MQ-2 varie en fonction de la concentration de fumée ou de gaz. Plus la concentration de gaz est élevée, plus la tension de sortie est élevée. Le signal logique peut être calibré en tenant le capteur à proximité de la fumée que vous souhaitez détecter.

II. Types de Capteurs

(8/10)

D. Capteur de Gaz MQ-2

Matériel nécessaire :

- NodeMCU ESP32.
- Capteur de gaz MQ-2
- Résistance de charge 10K ohms
- Câbles de raccordement
- Alimentation 5V

Câblage :

- Connectez la broche A0 (Analogique 0) du capteur MQ-2 à une broche analogique de l'ESP32, par exemple, A0.
- Connectez la broche D0 (Digital 0) du capteur MQ-2 à une broche numérique de l'ESP32, par exemple, D2.
- Connectez une extrémité de la résistance de charge (10K ohms) à la broche A0 du capteur MQ-2.
- Connectez l'autre extrémité de la résistance de charge au 5V de l'ESP32.
- Connectez la broche GND du capteur MQ-2 au GND de l'ESP32.
- Alimentez le capteur MQ-2 avec une alimentation externe de 5V.

```
const int analogPin = A0; // Broche analogique pour le capteur MQ-2
const int digitalPin = D2; // Broche numérique pour le capteur MQ-2

void setup() {
  Serial.begin(115200);
  pinMode(digitalPin, INPUT);
}

void loop() {
  int value = analogRead(analogPin);
  // Lit la valeur analogique du capteur MQ-2
  int digitalValue = digitalRead(digitalPin);
  // Lit la valeur numérique du capteur MQ-2

  Serial.print("Valeur analogique : ");
  Serial.println(value);
  Serial.print("Valeur numérique : ");
  Serial.println(digitalValue);

  delay(1000); // Attendez quelques secondes entre les lectures
}
```

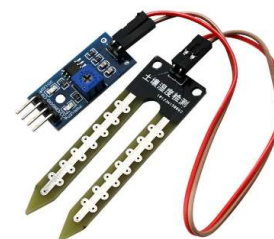
II. Types de Capteurs

(9/10)

E. Capteur d'Humidité du Sol (capteur d'humidité du sol FC-28)

Mesure l'humidité du sol, idéal pour les projets de jardinage automatisé.

- Tension de fonctionnement: cc 3.3-5.5V
- Tension de sortie: cc 0-3.0V
- Interface: PH2.0-3P
- Taille: 99x1 6mm/3.9x0.63"



II. Types de Capteurs

(10/10)

E. Capteur d'Humidité du Sol (capteur d'humidité du sol FC-28)

Matériel nécessaire :

- ESP32 (NodeMCU ESP32 ou équivalent)
- Capteur d'humidité du sol FC-28
- Alimentation (3.3V ou 5V)

Câblage :

- Connectez le module d'interface au capteur FC-28.
- Connectez la broche AO (Analog Out) du module d'interface à une broche analogique de l'ESP32, par exemple, A0.
- Connectez la broche GND du module d'interface au GND de l'ESP32.
- Alimentez le module d'interface avec une alimentation externe de 3.3V ou 5V.

```
const int analogPin = A0;
// Broche analogique pour le capteur FC-28

void setup() {
  Serial.begin(115200);
}

void loop() {
  int value = analogRead(analogPin);
  // Lit la valeur analogique du capteur FC-28

  Serial.print("Niveau d'humidité du sol : ");
  Serial.println(value);

  delay(1000);
  // Attendez quelques secondes entre les lectures
}
```

III. Actuateurs

(1/18)

Rôle des Actuateurs dans l'IoT

- Les actionneurs sont des éléments clés de l'IoT.
- Ils transforment les données en actions physiques.
- Exemples : moteurs, relais, électrovannes, lumières, buzzeurs.
- Ils automatisent des tâches, améliorent la sécurité, la gestion de l'énergie, et l'efficacité des systèmes IoT.
- Les actionneurs sont les exécuteurs des décisions IoT dans le monde réel.

1. Relais (Relay Module);
2. Électrovannes;
3. Écrans LCD ou OLED;
4. Haut-parleurs ou Buzzer;
5. Moteurs DC (DC Motors) ;
6. Moteurs pas à pas (Stepper Motors);
7. Servomoteurs (Servo Motors).

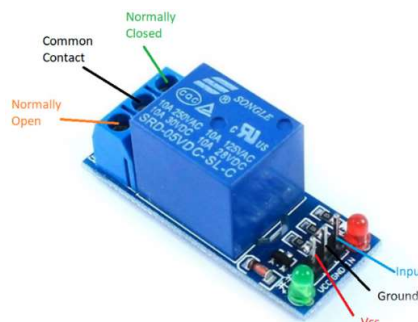
IV. Actuateurs

(2/18)

A. Relais (Relay Module)

Utilisé pour activer ou désactiver des appareils électriques tels que lumières, ventilateurs, etc.

- **Nombre de Voies** : Un ou plusieurs canaux pour contrôler plusieurs charges.
- **Tension de Commutation** : Fonctionne généralement sous 5 V pour le signal de commande.
- **Courant de Commutation** : Capacité de commutation, par exemple, 10 A pour des charges jusqu'à 10 ampères.
- **Type de Charge** : Compatible avec courant continu (CC) et alternatif (CA).
- **Protection Intégrée** : Diodes de Flyback pour prévenir les pics de tension.
- **Interface** : Broches ou connecteurs standard pour la connexion à l'Arduino.
- **LED d'Indication** : Indique l'état du relais (activé/désactivé).



IV. Actuateurs

(3/18)

B. Électrovannes

Les électrovannes sont couramment utilisées pour contrôler le flux de liquides ou de gaz, que ce soit pour l'irrigation automatique, la distribution de liquides, les systèmes de contrôle de gaz, etc.

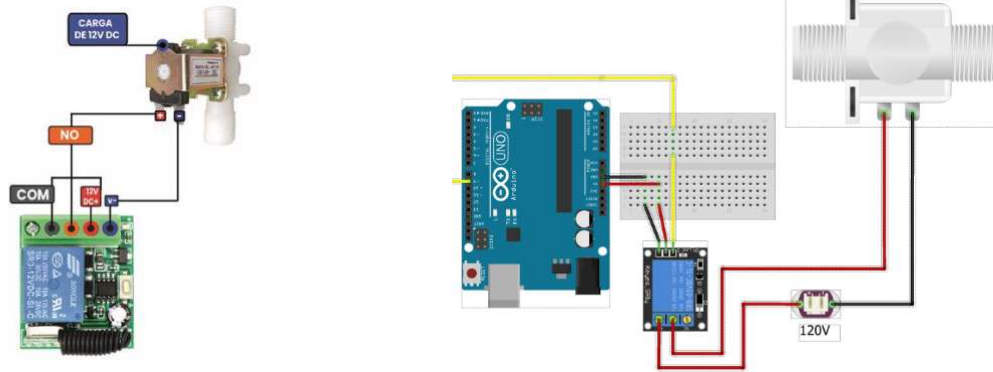
- **Type d'Actionneur** : Électrovanne à actionnement électrique.
- **Tension de Fonctionnement** : Habituellement 12 V ou 24 V, selon le modèle.
- **Type de Contrôle** : Généralement commandé par un signal PWM (modulation de largeur d'impulsion).
- **Débit** : Spécifie le volume d'écoulement de fluide (en litres par minute).
- **Type de Fluide** : Conçu pour l'eau, l'air, les gaz, etc., en fonction de l'application.
- **Type de Connexion** : Filetage mâle ou femelle pour le raccordement aux tuyaux.



IV. Actuateurs

B. Électrovannes

(4/18)



IV. Actuateurs

B. Électrovannes

(5/18)

```
#include <Arduino.h>
// Broches de connexion
const int boutonPin = 2; // Broche du bouton, à adapter à votre configuration
const int relaisPin = 4; // Broche de contrôle du relais, à adapter à votre configuration
// Variables de statut du relais et du bouton
bool etatRelais = LOW; // LOW = éteint, HIGH = allumé
bool etatBoutonPrecedent = LOW;
bool etatBoutonActuel = LOW;
void setup() {
  pinMode(boutonPin, INPUT_PULLUP); // Configuration du bouton avec résistance de pull-up interne
  pinMode(relaisPin, OUTPUT);
  digitalWrite(relaisPin, etatRelais);
  Serial.begin(115200);
}
void loop() {
  // Lecture de l'état actuel du bouton
  etatBoutonActuel = digitalRead(boutonPin);

  // Vérification du changement d'état du bouton
  if (etatBoutonActuel != etatBoutonPrecedent) {
    delay(50); // Anti-rebond
    etatBoutonActuel = digitalRead(boutonPin);

    // Si le bouton est enfoncé, basculez l'état du relais
    if (etatBoutonActuel == LOW) {
      etatRelais = !etatRelais;
      digitalWrite(relaisPin, etatRelais);
    }
    etatBoutonPrecedent = etatBoutonActuel;
  }
}
```

IV. Actuateurs

(6/18)

C. Écrans LCD ou OLED

Afficher des informations visuelles, comme les écrans OLED dans les montres intelligentes.

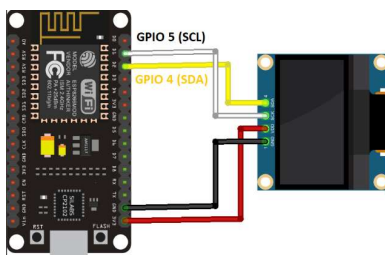
- **Type d'affichage** : Les écrans LCD pour Arduino sont généralement de type alphanumérique, graphique ou TFT (Thin-Film Transistor).
- **Taille de l'écran** : Ils sont disponibles en différentes tailles, de petits écrans 16x2 caractères aux grands écrans TFT couleur.
- **Résolution** : La résolution dépend du type d'écran, avec des écrans alphanumériques offrant généralement une résolution standard.
- **Interface** : Les écrans LCD utilisent généralement une interface parallèle ou série, avec des broches d'entrée/sortie pour la communication avec Arduino.
- **Tension d'alimentation** : Ils fonctionnent généralement avec une tension de 5V ou 3.3V, selon le modèle.
- **Bibliothèques disponibles** : Il existe des bibliothèques Arduino prêtes à l'emploi pour faciliter la programmation des écrans LCD.



IV. Actuateurs

(7/18)

C. Écrans LCD ou OLED



```
#include <Wire.h> // Inclure la bibliothèque Wire pour la communication I2C
#include <LiquidCrystal_I2C.h> // Inclure la bibliothèque LiquidCrystal_I2C

// Initialisation de l'écran LCD avec l'adresse I2C 0x27 (vous pouvez modifier cette adresse si elle est différente)
LiquidCrystal_I2C lcd(0x27, 16, 2); // Utilisez 16 colonnes x 2 lignes

void setup() {
  // Initialisation de l'écran LCD
  lcd.init();

  // Allumer le rétroéclairage (peut être supprimé si l'écran s'allume automatiquement)
  lcd.backlight();

  // Afficher un message sur l'écran LCD
  lcd.setCursor(0, 0); // Position du curseur (colonne, ligne)
  lcd.print("Bonjour, ESP32!");
}

void loop() {
  // Votre code principal peut aller ici
}
```


IV. Actuateurs

(8/18)

D. Haut-parleurs ou Buzzer

Produire des sons ou des alertes sonores dans les projets audio.

- **Tension de Fonctionnement** : 3,3 V ou 5 V
- **Type de Signal d'Entrée** : Numérique
- **Niveau Sonore** : Variable (mesuré en dB)
- **Fréquence de Résonance** : Spécifique au modèle
- **Type de Contrôle** : Actif ou Passif
- **Type de Connexion** : Diverses options disponibles
- **Taille et Forme** : Variées
- **Compatibilité avec Arduino** : Vérifier la tension et le contrôle

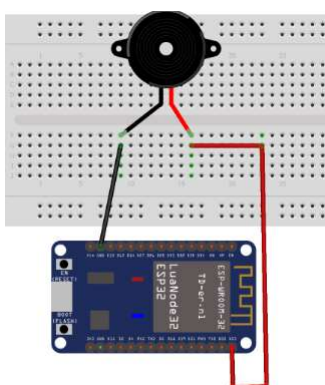


IV. Actuateurs

(9/18)

D. Haut-parleurs ou Buzzer

Produire des sons ou des alertes sonores dans les projets audio.



```
// Broche de contrôle du buzzer
int buzzerPin = 13; // Vous pouvez choisir une autre broche

void setup() {
  pinMode(buzzerPin, OUTPUT);
}

void loop() {
  // Jouer la note DO (C)
  tone(buzzerPin, 261.63); // Fréquence pour la note DO
  delay(500);
  noTone(buzzerPin);
  delay(100);

  // Jouer la note MI (E)
  tone(buzzerPin, 329.63); // Fréquence pour la note MI
  delay(500);
  noTone(buzzerPin);
  delay(100);

  // Jouer la note SOL (G)
  tone(buzzerPin, 392.00); // Fréquence pour la note SOL
  delay(500);
  noTone(buzzerPin);
  delay(100);
}
```

IV. Actuateurs

(10/18)

E. Moteurs DC (DC Motors)

Permet de contrôler la vitesse et la direction des moteurs DC, souvent utilisés dans les robots et les projets de domotique.

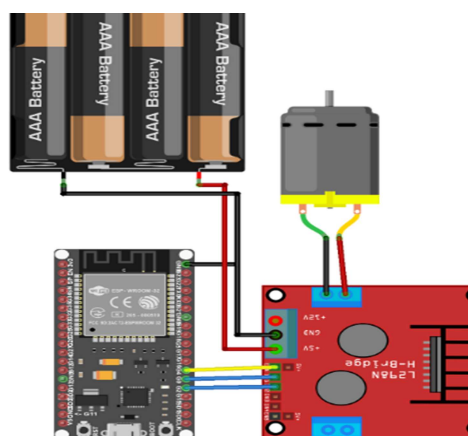
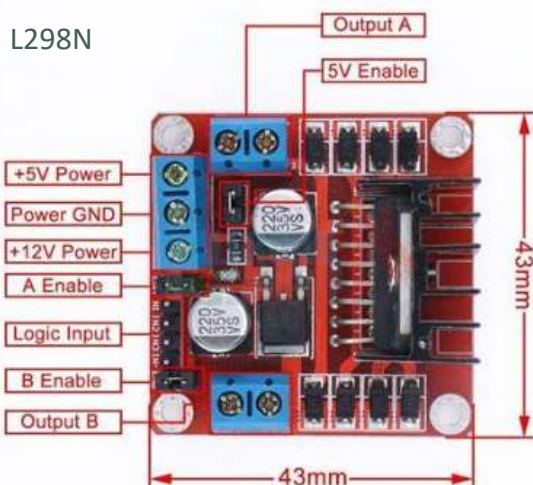
- **Commande** : Réagit à la polarité de la tension appliquée
- **Mouvement** : Tourne en continu lorsqu'une tension est appliquée
- **Contrôle** : La vitesse et la direction peuvent être contrôlées en ajustant la tension
- **Alimentation** : Souvent utilisé avec des tensions de 3V, 5V, 12V, etc.
- **Types** : Moteurs à balais (avec balais) et moteurs sans balais (brushless)
- **Couples** : Disponible en différentes gammes de couples en fonction des besoins
- **Contrôle** : Peut être contrôlé en utilisant des ponts en H, des variateurs de vitesse, etc.



IV. Actuateurs

(11/18)

E. Moteurs DC (DC Motors)



IV. Actuateurs

(12/18)

E. Moteurs DC (DC Motors)

```
// Bibliothèques requises
#include <Arduino.h>

// Broches de connexion au module L298N
const int enA = 2; // Broche Enable du moteur A
const int in1 = 4; // Broche d'entrée 1 du moteur A
const int in2 = 5; // Broche d'entrée 2 du moteur A

const int enB = 15; // Broche Enable du moteur B
const int in3 = 13; // Broche d'entrée 1 du moteur B
const int in4 = 12; // Broche d'entrée 2 du moteur B

// Broches de connexion des boutons
const int boutonGauche = 14; // Bouton pour tourner à gauche
const int boutonDroite = 16; // Bouton pour tourner à droite

void setup() {
  // Configuration des broches
  pinMode(enA, OUTPUT);
  pinMode(in1, OUTPUT);
  pinMode(in2, OUTPUT);

  pinMode(enB, OUTPUT);
  pinMode(in3, OUTPUT);
  pinMode(in4, OUTPUT);

  pinMode(boutonGauche, INPUT_PULLUP);
  pinMode(boutonDroite, INPUT_PULLUP);
}

void loop() {
  // Lecture de l'état des boutons
  int etatBoutonGauche = digitalRead(boutonGauche);
  int etatBoutonDroite = digitalRead(boutonDroite);

  // Contrôle du moteur en fonction de l'état des boutons
  if (etatBoutonGauche == LOW) {
    // Tourner à gauche
    digitalWrite(enA, HIGH);
    digitalWrite(in1, LOW);
    digitalWrite(in2, HIGH);

    digitalWrite(enB, HIGH);
    digitalWrite(in3, LOW);
    digitalWrite(in4, HIGH);
  } else if (etatBoutonDroite == LOW) {
    // Tourner à droite
    digitalWrite(enA, HIGH);
    digitalWrite(in1, HIGH);
    digitalWrite(in2, LOW);

    digitalWrite(enB, HIGH);
    digitalWrite(in3, HIGH);
    digitalWrite(in4, LOW);
  } else {
    // Arrêter le moteur
    digitalWrite(enA, LOW);
    digitalWrite(enB, LOW);
  }
}
```

IV. Actuateurs

(13/18)

F. Moteurs pas à pas (Stepper Motors)

Les moteurs pas à pas sont idéaux pour le contrôle précis du mouvement, ce qui les rend adaptés à une variété d'applications, notamment l'impression 3D, le positionnement précis et les systèmes de suivi solaire.

- **Commande** : contrôlé avec des signaux numériques
- **Précision** : Permet un positionnement précis grâce à ses pas individuels
- **Mouvement** : Avance par incréments fixes (pas) plutôt que de tourner en continu
- **Contrôle** : Réagit aux signaux de commande pour chaque pas
- **Alimentation** : Souvent utilisé avec une tension de 5V ou 12V
- **Angle de Pas** : Mesuré en degrés, par exemple 1.8 degrés par pas
- **Couples** : Disponible en différentes gammes de couples en fonction des besoins
- **Contrôle** : Peut être contrôlé en utilisant des pilotes de moteur pas à pas



IV. Actuateurs

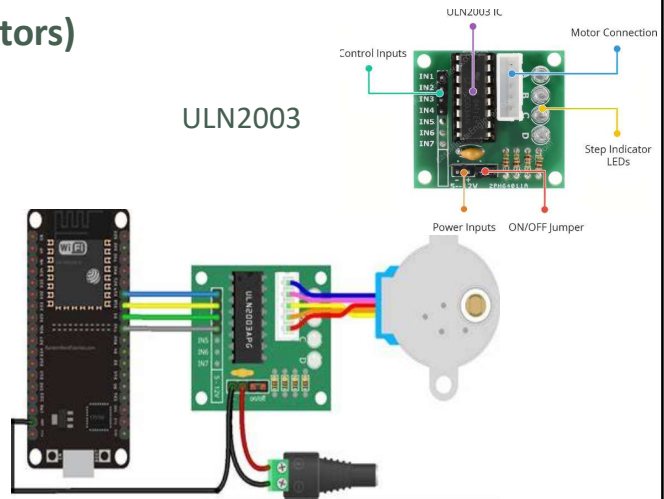
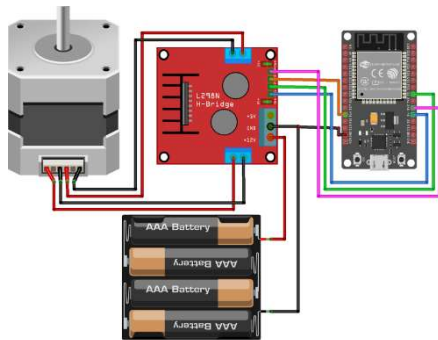
(14/18)

F. Moteurs pas à pas (Stepper Motors)

L298N

ou

ULN2003



IV. Actuateurs

(15/18)

F. Moteurs pas à pas (Stepper Motors)

L298N

ou

ULN2003

```

#include <AccelStepper.h>
// Broches de connexion au module L298N
const int enA = 2; // Broche Enable du moteur A
const int in1 = 4; // Broche d'entrée 1 du moteur A
const int in2 = 5; // Broche d'entrée 2 du moteur A

const int enB = 15; // Broche Enable du moteur B
const int in3 = 13; // Broche d'entrée 1 du moteur B
const int in4 = 12; // Broche d'entrée 2 du moteur B

// Broches de connexion des boutons
const int boutonGauche = 14; // Bouton pour tourner à gauche
const int boutonDroite = 16; // Bouton pour tourner à droite

// Création de l'objet AccelStepper
AccelStepper stepper(1, in1, in2); // 1 est le numéro du pilote

void setup() {
    // Configuration des broches
    pinMode(enA, OUTPUT);
    pinMode(in1, OUTPUT);
    pinMode(in2, OUTPUT);

    pinMode(enB, OUTPUT);
    pinMode(in3, OUTPUT);
    pinMode(in4, OUTPUT);

    pinMode(boutonGauche, INPUT_PULLUP);
    pinMode(boutonDroite, INPUT_PULLUP);

    // Configuration du moteur pas à pas
    stepper.setMaxSpeed(1000); // Vitesse maximale
    stepper.setAcceleration(500); // Accélération
}

void loop() {
    // Lecture de l'état des boutons
    int etatBoutonGauche = digitalRead(boutonGauche);
    int etatBoutonDroite = digitalRead(boutonDroite);

    // Contrôle du moteur pas à pas en fonction de l'état des boutons
    if (etatBoutonGauche == LOW) {
        // Tourner à gauche
        digitalWrite(enA, HIGH);
        digitalWrite(in1, LOW);
        digitalWrite(in2, HIGH);

    } else if (etatBoutonDroite == LOW) {
        // Tourner à droite
        digitalWrite(enA, HIGH);
        digitalWrite(in1, HIGH);
        digitalWrite(in2, LOW);

    } else {
        // Arrêter le moteur
        digitalWrite(enA, LOW);
        digitalWrite(enB, LOW);
    }
}
    
```

```

#include <Arduino.h>
#include <Stepper.h>

// Configuration des broches pour le module ULN2003
const int in1 = 14;
const int in2 = 27;
const int in3 = 26;
const int in4 = 25;

// Configuration des boutons
const int boutonGauche = 16; // Bouton pour tourner à gauche
const int boutonDroite = 17; // Bouton pour tourner à droite

// Configuration du moteur pas à pas
Stepper monMoteur(512, in1, in2, in3, in4);

void setup() {
    // Configuration des broches de bouton en entrée avec une résistance de pull-up
    pinMode(boutonGauche, INPUT_PULLUP);
    pinMode(boutonDroite, INPUT_PULLUP);
}

void loop() {
    // Lecture de l'état des boutons
    int etatBoutonGauche = digitalRead(boutonGauche);
    int etatBoutonDroite = digitalRead(boutonDroite);

    // Contrôle du moteur en fonction de l'état des boutons
    if (etatBoutonGauche == LOW) {
        monMoteur.step(512); // Tourner à gauche d'un tour complet (512 pas)
        delay(500); // Délai pour éviter les mouvements rapides
    } else if (etatBoutonDroite == LOW) {
        monMoteur.step(-512); // Tourner à droite d'un tour complet (512 pas)
        delay(500);
    }
}
    
```

IV. Actuateurs

(16/18)

F. Servomoteurs (Servo Motors)

le contrôle précis de l'angle dans les projets de robotique et de mécanique.

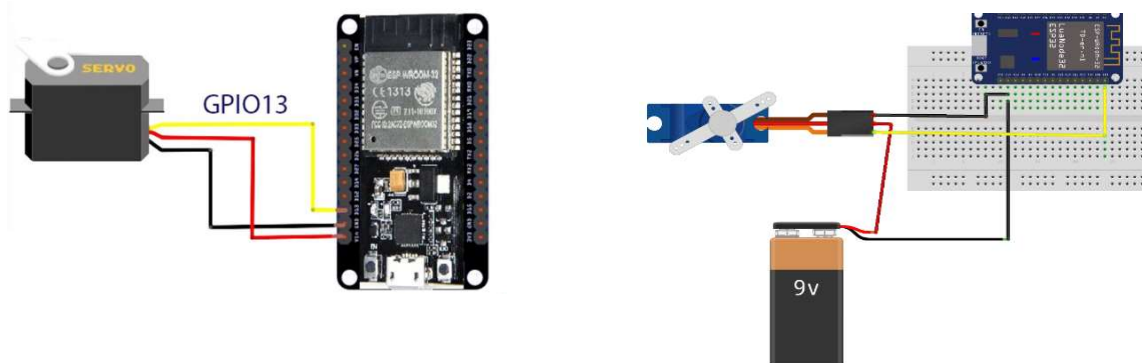
- **Commande** : Précise et proportionnelle, généralement via un signal PWM
- **Mouvement** : Rotation précise et limitée dans une plage définie (habituellement 0-180 degrés)
- **Contrôle** : Peut être positionné avec une grande précision et maintenu dans une position donnée
- **Alimentation** : Souvent utilisé avec des tensions de 5V ou 6V
- **Fiabilité** : Fiable pour des mouvements précis et contrôlés
- **Types** : Servomoteurs analogiques et numériques
- **Couples** : Disponible en différentes gammes de couples en fonction des besoins
- **Réponse** : Réagit rapidement aux signaux de commande
- **Angle de Rotation** : La plage de rotation est généralement de 0 à 180 degrés



IV. Actuateurs

(17/18)

F. Servomoteurs (Servo Motors)



IV. Actuateurs

(18/18)

F. Servomoteurs (Servo Motors)

```
#include <Servo.h>

Servo monServo; // Créez un objet Servo

const int boutonGauche = 14; // Bouton pour tourner à gauche
const int boutonDroite = 16; // Bouton pour tourner à droite

int position = 90; // Position initiale du servo

void setup() {
  monServo.attach(2); // Attachez le servo à la broche 2
  pinMode(boutonGauche, INPUT_PULLUP);
  pinMode(boutonDroite, INPUT_PULLUP);
}
```

```
void loop() {
  int etatBoutonGauche = digitalRead(boutonGauche);
  int etatBoutonDroite = digitalRead(boutonDroite);

  if (etatBoutonGauche == LOW) {
    // Appuyez sur le bouton de gauche pour faire tourner le servo à gauche
    position -= 1;
    if (position < 0) {
      position = 0;
    }
  } else if (etatBoutonDroite == LOW) {
    // Appuyez sur le bouton de droite pour faire tourner le servo à droite
    position += 1;
    if (position > 180) {
      position = 180;
    }
  }

  monServo.write(position); // Déplacez le servo à la nouvelle position
  delay(15); // Petite pause pour permettre au servo de se déplacer en douceur
}
```

IV. Raspberry pi 3P

(1/19)

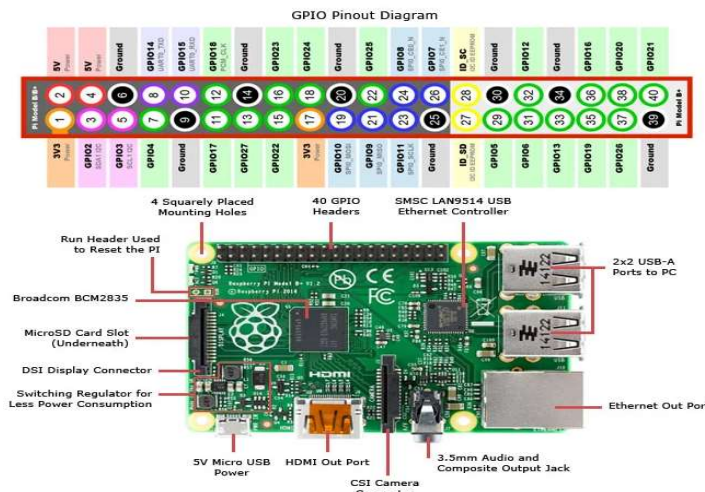
1. Qu'est-ce qu'une carte Raspberry Pi ?
2. Qu'est-ce qu'on peut faire avec ?
3. Et par rapport à un PC ?
4. Quel OS choisir ?
5. Téléchargement du système : Raspberry Pi OS
6. Créer une carte SD depuis Windows avec Raspberry Pi Imager
7. Premier démarrage
8. Connectez vous en SSH à votre Raspberry Pi pour la contrôler depuis votre ordinateur
9. Comment activer SSH sur le Raspberry Pi
10. Utilisez SSH avec Windows et Putty
11. Raspbian / Quelques commandes

IV. Raspberry pi 3P

(2/19)

1. Qu'est-ce qu'une carte Raspberry Pi ?

- Microcontrôleur + processeur graphique
- Système d'exploitation
- Entrées / Sorties numériques
- Carte SD / RAM

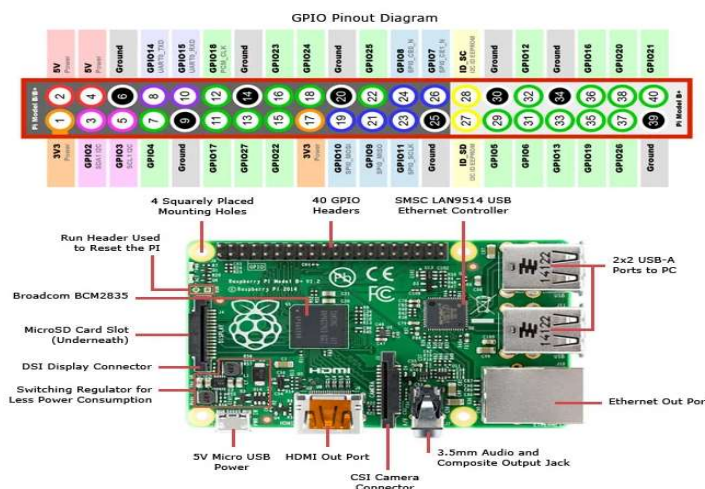
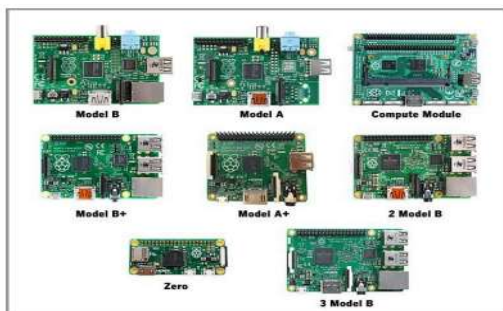


IV. Raspberry pi 3P

(3/19)

1. Qu'est-ce qu'une carte Raspberry Pi ?

- Microcontrôleur + processeur graphique
- Système d'exploitation
- Entrées / Sorties numériques
- Carte SD / RAM

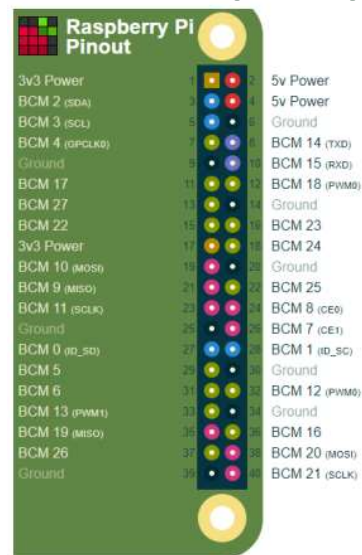
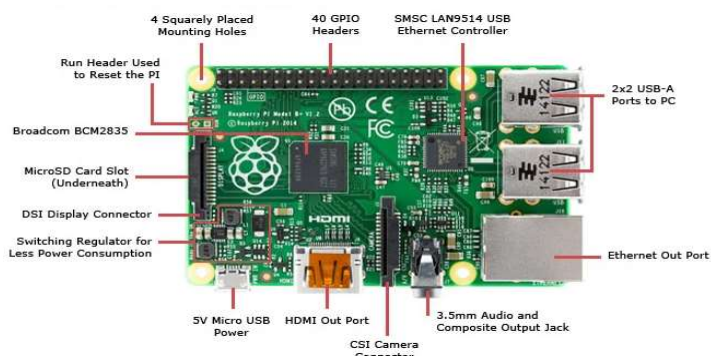


IV. Raspberry pi 3P

(4/19)

2. Qu'est-ce qu'on peut faire avec ?

- Mini-PC / Programmation
- Interface Homme Machine
- Traitement d'image / Caméra
- Réseau / Mini-serveur



IV. Raspberry pi 3P

(5/19)

3. Et par rapport à un PC ?

- Processeurs généralistes vs Processeurs embarqués
- E/S

	PC standard	Raspberry PI 3 B+	ARDUINO Uno
Fréquence	2 GHz	1.4 GHz	16 MHz
Core / Architecture	4 / 64 bits	ARM-quadri / 64 bits	1 / 8 bits
Consommation	100 à 500 W	4 W	< 1 W
Entrées/Sorties	/	40 GPIO	6 Analog / 13 Digital
Ports extension	USB, PCI...	USB, SPI, I2C, RS232, ETH	SPI, I2C, RS232
RAM	4 Go	1 Go	2 ko

IV. *Raspberry pi 3P*

(6/19)

4. Quel OS choisir ?

Les Raspberry Pi sont livrés par défaut vierges de tout système d'exploitation et de tout dispositif de stockage. C'est à l'utilisateur de choisir et d'installer le système d'exploitation qu'il veut (la plus grande majorité sont des systèmes Linux) et de choisir la carte de stockage de type micro-SD avec la capacité qu'il veut (16 Go minimum conseillés).

Il existe de nombreux systèmes d'exploitation dédiés au Raspberry Pi :

- **RecalBox OS** qui est dédié à la transformation de votre Raspberry Pi en une console de jeu dite de « retrogaming »;
- **OSMC** pour ceux qui désirent transformer leur Raspberry en « media center »;
- **Raspberry Pi OS** (sans fonctionnalités orientées pour une utilisation spécifique). Il s'agit d'un système d'exploitation GNU/Linux, spécialement conçu et optimisé pour les Raspberry Pi.

IV. *Raspberry pi 3P*

(7/19)

5. Téléchargement du système : Raspberry Pi OS

Pour faire fonctionner le Raspberry, on va donc commencer par télécharger son système d'exploitation Raspberry Pi OS à partir de cette page du site :

<https://raspberrypi.fr/telechargements/>

N.B : Le fichier téléchargé est une archive compressée **.ZIP** Il est nécessaire d'extraire son contenu (glisser - déposer dans le répertoire de son choix sur l'ordinateur) pour obtenir le fichier **.img** de l'image de Raspbian qu'on installera sur la carte micro-SD.

IV. *Raspberry pi 3P*

(8/19)

6. Créer une carte SD depuis Windows avec Raspberry Pi Imager

Le matériel nécessaire :

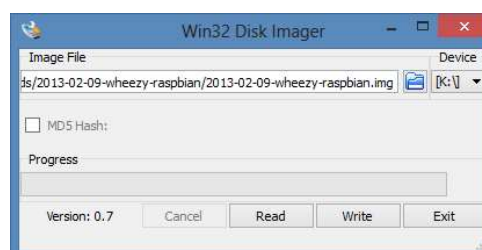
Pour pouvoir créer une carte SD pour votre Raspberry Pi vous aurez besoin :

- Une ordinateur sous Windows
- Une carte microSD
- Un lecteur de carte MicroSD.



Installer Raspberry Pi Imager sur la carte micro-SD

Raspberry Pi Imager s'installe comme le ferait n'importe quel logiciel sur votre système d'exploitation. Il vous suffit de télécharger la version adaptée à votre système d'exploitation Windows, de double cliquer sur le fichier téléchargé «Win32DiskImager » et de suivre le guide.



IV. *Raspberry pi 3P*

(9/19)

7. Premier démarrage

Pour le tout premier démarrage, il sera nécessaire d'avoir la totalité du matériel à sa disposition :

- Un Raspberry Pi 3 avec son boîtier et son alimentation;
- La carte micro SD (avec le système Raspberry Pi OS installé à l'étape précédente) insérée dans son logement dans le Raspberry;
- Un clavier USB et une souris USB;
- Un câble HDMI;
- Un écran (avec entrée HDMI);
- Une connexion (WiFi ou filaire RJ45) .

N.B : Le Raspberry ne possède pas d'interrupteur marche / arrêt :

- Il s'allume quand on le branche sur le secteur ;
- Il s'éteint quand on le débranche (après avoir arrêté correctement le système d'exploitation, comme sur tous les ordinateurs).

165

Mr. MEGNAFI Hicham (ESSA-Tlemcen)

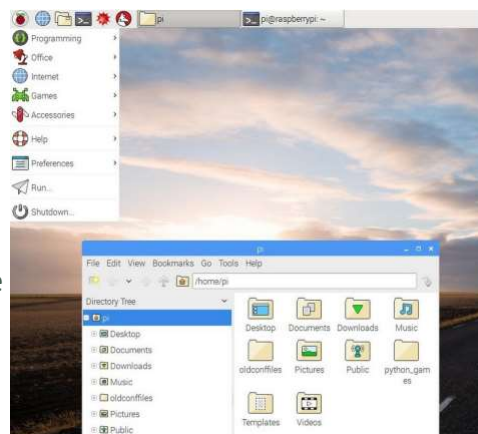
IV. Raspberry pi 3P

(10/19)

7. Premier démarrage

Méthode :

- Brancher le clavier et la souris sur les ports USB du Raspberry Pi;
- Brancher le câble HDMI sur la prise correspondante du Raspberry et derrière votre écran ;
- Brancher l'écran sur le secteur;
- Brancher le cordon d'alimentation du Raspberry dans sa prise dédiée puis le boîtier transformateur sur le secteur.
- Le Raspberry et l'écran s'allument, au bout de quelques secondes vous êtes « sur le bureau » de votre tout nouvel ordinateur...



166

Mr. MEGNAFI Hicham (ESSA-Tlemcen)

IV. Raspberry pi 3P

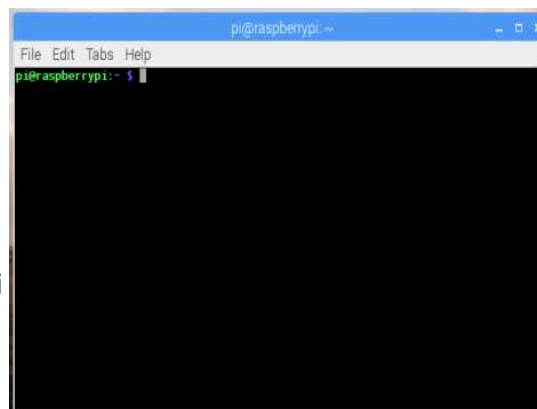
(11/19)

7. Premier démarrage

Complément :

Par défaut toujours, le nom de la machine (le « hostname ») est « **raspberrypi** », le nom de l'utilisateur est « **pi** » et son mot de passe est « **raspberry** ». D'où l'invite de commande qu'on peut voir quand on ouvre le terminal :

pi@raspberrypi (comprendre : c'est l'utilisateur « pi » qui a ouvert une session sur la machine « raspberrypi »).



IV. *Raspberry pi 3P* (12/19)

8. Connectez vous en SSH à votre Raspberry Pi pour la contrôler depuis votre ordinateur

Le fait de savoir comment prendre le contrôle de la Raspberry Pi à distance revient très fréquemment !
SSH c'est quoi ?

SSH (pour Secure SHell) désigne à la fois un logiciel et un protocole de communication informatiques. Ce protocole possède par ailleurs la particularité d'être entièrement chiffré. Cela signifie donc que toutes les commandes que vous exécuterez via SSH seront totalement secrètes !

Installer SSH pour prendre le contrôle de votre Raspberry Pi

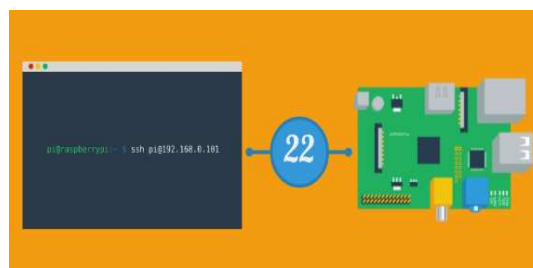
Tout d'abord, vous devez savoir que l'installation de SSH se découpe en deux parties. En effet, vous aurez besoin d'un côté **serveur SSH sur votre Raspberry Pi** et d'un autre d'un **client SSH sur votre ordinateur** .

IV. *Raspberry pi 3P* (13/19)

9. Comment activer SSH sur le Raspberry Pi

Pour contrôler un Raspberry Pi à distance, le plus simple est d'utiliser SSH.

Dans cette présentation nous allons donc voir comment nous pouvons activer SSH sur le Raspberry Pi, avec ou sans clavier !



169

Mr. MEGNAFI Hicham (ESSA-Tlemcen)

IV. Raspberry pi 3P

(14/19)

9. Comment activer SSH sur le Raspberry Pi

Le matériel nécessaire pour activer SSH :

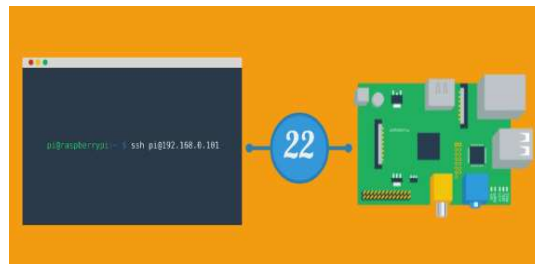
Pour activer SSH nous n'aurons besoin que de peu de matériel. Deux solutions sont possibles.

1^{er} solution : activer SSH depuis le Raspberry Pi :

- écran;
- clavier.

2^{eme} solution: activer SSH depuis votre ordinateur en modifiant un fichier sur la carte SD :

- PC ;
- lecteur de carte SD.



170

Mr. MEGNAFI Hicham (ESSA-Tlemcen)

IV. Raspberry pi 3P

(15/19)

9. Comment activer SSH sur le Raspberry Pi

Activer SSH depuis le Raspberry Pi avec raspi-config :

1^{er} solution pour activer SSH : nous allons directement utiliser l'outil en ligne de commande **raspi-config**.

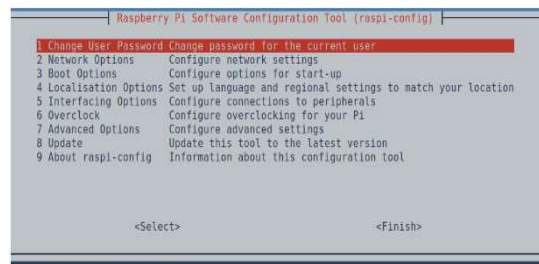
Ouvrez un terminal, et tapez-y la commande suivante :

```
sudo raspi-config
```

Vous allez arriver devant la fenêtre suivante →

Vous allez dans la partie « **Interfacing Options** » et choisissez la ligne SSH, puis « Yes ».

Et voilà, votre serveur SSH est activé !



IV. Raspberry pi 3P

(16/19)

9. Comment activer SSH sur le Raspberry Pi

Activer SSH sans écran ni clavier, depuis votre PC :

2^{eme} solution pour activer SSH :

il va nous suffir de créer un fichier sur la carte SD du Raspberry, et il activera automatiquement SSH au prochain démarrage.

Insérez donc la carte SD de votre Pi dans votre PC, et rendez-vous dans la partition boot, qui est d'ailleurs la seule accessible depuis Windows.

Ceci fait, créez un fichier nommé ssh dans la partition boot de la carte. Pas d'extension, pas de contenu, juste un fichier vide nommé ssh.

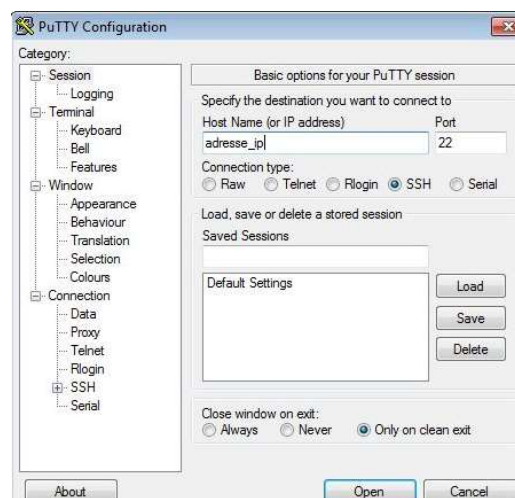
Retirez la carte du PC, remettez-la dans la Pi, allumez celle-ci, et voilà, SSH est activé .

IV. Raspberry pi 3P

(17/19)

10. Utilisez SSH avec Windows et Putty

il va falloir passer par l'interface graphique. Vous allez donc devoir lancer Putty et remplir les informations comme sur l'image ci-dessous.



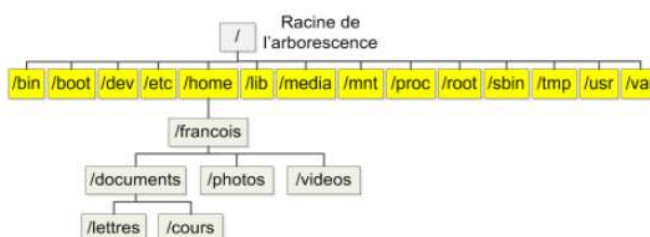
IV. Raspberry pi 3P

(18/19)

11. Raspbian / Quelques commandes

Se promener dans les répertoires et les fichiers en ligne de commande

<code>ls</code>	liste les fichiers et répertoires
<code>ls -al</code>	liste tous les fichiers
<code>cd /rep/dest/</code>	se déplace dans le répertoire /rep/dest/
<code>cd ..</code>	se déplace dans le répertoire de niveau supérieur
<code>mkdir new_rep/</code>	créée un répertoire
<code>new_rep</code>	dans le répertoire courant



IV. Raspberry pi 3P

(19/19)

11. Raspbian / Quelques commandes

- Mettre à jour sa distribution

<code>sudo apt-get</code>	update pour mettre à jour la liste des paquets
<code>sudo apt-get</code>	upgrade pour mettre à jour les logiciels
<code>sudo apt-get dist-upgrade</code>	pour mettre à jour Raspbian

- Vérifier le réseau

<code>ifconfig</code>	retourne les informations concernant le réseau
-----------------------	--

- Configurer la Raspberry pi

<code>sudo raspi-config</code>	pour accéder à la configuration de la carte
--------------------------------	---

- Editer des fichiers textes

<code>geany</code>	reconnait la plupart des langages de programmation
<code>nano</code>	simple éditeur de texte

- Installer des paquets python / librairies supplémentaires

<code>pip</code>	install nom_paquet pour python 2
<code>pip3</code>	install nom_paquet pour python 3

Plan de Cours

1. Projets réalisés
2. Introduction
3. Le marché de l'IOT
4. Interactions entre le « monde numérique » et le « monde physique »
5. Principes fondamentaux de l'IoT
6. Composants de base de l'IoT
7. **Connectivité**

➤ Réseaux cellulaires

- 3G, 4G, 5G
- NB-IoT (Narrowband IoT)
- **LTE-M**
- SIM900 GSM/GPRS Shield
- SIM800L GSM/GPRS Shield

➤ Réseaux sans fil

- Wi-Fi
- Bluetooth
- Zigbee

➤ Détection de proximité

- NFC (Near Field Communication)
- RFID (Radio-Frequency Identification)
- QR Codes

Connectivité

I. Réseaux cellulaires

- A. Pourquoi les Réseaux Cellulaires ?
- B. Réseaux : 2G, 3G, 4G, 5G
- C. Comparaison entre les réseaux : 2G, 3G, 4G, 5G
- D. LTE-M and NB-IOT
 - D.1. LTE-M
 - D.2. NB-IoT (Narrowband IoT)
 - D.3. M2M-GSM (2G)
 - D.4. Comparaison entre les réseaux : LTE, LTE-M, GSM-M2M et NB-IOT
- A. Shields IoT pour une Connectivité de Réseau Cellulaire
- B. SIM900 GSM/GPRS Shield
- C. Envoi d'un SMS via le shield SIM900 GSM/GPRS
- D. SIM800L GSM/GPRS Shield
- E. Appel téléphonique en utilisant le shield SIM800L GSM/GPRS

II. Réseaux sans fil

II.1 Importance de la Connectivité sans fil

II.2 Wi-Fi dans l'IoT

- A. Wi-Fi dans l'IoT – Technologie
- B. Shields Wi-Fi pour Arduino
- C. Exemple de code d'Utilisation d'ESP32 qui se connecte à un réseau Wi-Fi

II.3 Bluetooth dans l'IoT

- A. Bluetooth dans l'IoT - Technologie
- B. Shields Bluetooth pour Arduino
- C. Exemple d'Utilisation du Module Bluetooth HC-06
- D. Code pour envoyer un message à un appareil Bluetooth connecté

II.4 Zigbee dans l'IoT

- A. Zigbee dans l'IoT - Technologie
- B. Shields Zigbee pour Arduino
- C. Exemple d'Utilisation du Module Zigbee
- D. Code d'Exemple Zigbee

II.5 Récapitulatif et Synthèse

III. Détection de proximité

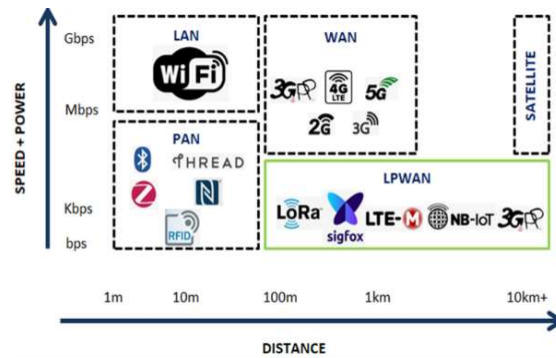
- A. NFC (Near Field Communication)
 - Fonctionnement de la technologie NFC
 - Applications de détection de proximité basées sur le NFC
- B. RFID (Radio-Frequency Identification)
 - Principes de fonctionnement des systèmes RFID
 - Exemples d'applications de détection de proximité avec RFID
- C. QR Codes
 - Comment les QR codes peuvent être utilisés pour la détection de proximité
 - Exemples d'applications basées sur les QR codes

I. Réseaux cellulaires

(1/16)

A. Pourquoi les Réseaux Cellulaires ?

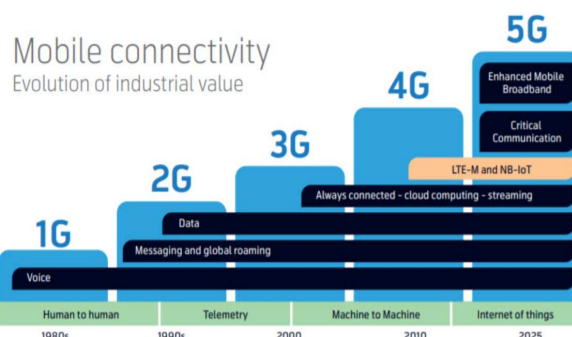
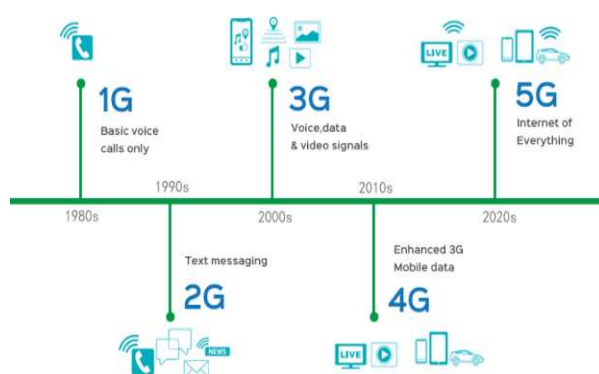
- ❑ **Fiabilité pour la transmission de données IoT :**
 Les réseaux cellulaires offrent une connectivité stable et fiable, garantissant que les données IoT sont transmises en temps opportun et de manière cohérente.
- ❑ **Portée étendue couvrant de vastes zones géographiques :**
 Les réseaux cellulaires ont une portée qui peut couvrir de vastes territoires, ce qui les rend adaptés aux applications IoT étendues, telles que la surveillance agricole ou la gestion de flottes.
- ❑ **Disponibilité d'une connexion permanente :**
 Les dispositifs IoT peuvent rester constamment connectés grâce aux réseaux cellulaires, assurant ainsi une communication ininterrompue.



I. Réseaux cellulaires

(2/16)

B. Réseaux : 2G, 3G, 4G, 5G



I. Réseaux cellulaires

(3/16)

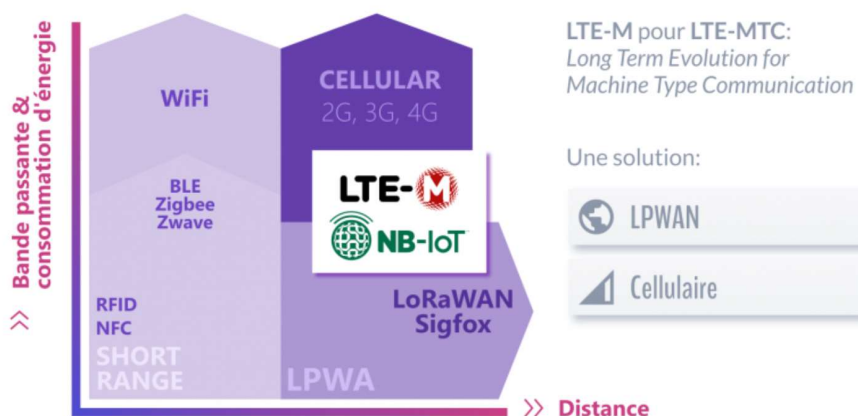
C. Comparaison entre les réseaux : 2G, 3G, 4G, 5G

Caractéristique	Réseau 2G	Réseau 3G	Réseau 4G	Réseau 5G
Débit de données	Faible (jusqu'à 236 Kbps)	Moyen (jusqu'à 3.1 Mbps)	Élevé (jusqu'à 100 Mbps)	Très élevé (jusqu'à plusieurs Gbps)
Latence	Modérée	Modérée	Faible	Très faible
Couverture	Bonne	Bonne	Excellente	En cours de déploiement
Consommation d'énergie	Faible	Modérée	Variable (optimisation possible)	Variable (optimisation possible)
Capacité d'appareils pris en charge	Faible	Modérée	Élevée	Très élevée
Utilisation principale	M2M (Machine-to-Machine)	Voix et données	Données et multimédia	Diverses applications IoT et multimédia

I. Réseaux cellulaires

(4/16)

D. LTE-M and NB-IOT



I. Réseaux cellulaires

(5/16)

D.1. LTE-M

LTE-M est une technologie de communication sans fil conçue pour répondre aux besoins de l'Internet des objets (IoT) en utilisant les réseaux 4G LTE.

Caractéristiques clés :

- Faible consommation d'énergie** : LTE-M permet une utilisation efficace de l'énergie, prolongeant ainsi la durée de vie de la batterie des appareils IoT.
- Pénétration accrue** : Cette technologie offre une meilleure pénétration à travers les obstacles physiques, améliorant la connectivité en intérieur et en souterrain.
- Vitesse de données modérée** : LTE-M offre des débits de données modérés adaptés aux applications IoT telles que la surveillance et le suivi.
- Prise en charge de nombreux appareils** : Il peut gérer un grand nombre d'appareils connectés simultanément.

Applications :

- Suivi d'actifs** : Pour suivre la localisation des biens et des équipements.
- Télémetrie** : Utilisée pour la surveillance à distance, notamment dans l'industrie.
- Gestion de flottes** : Pour la surveillance et la gestion des véhicules commerciaux.

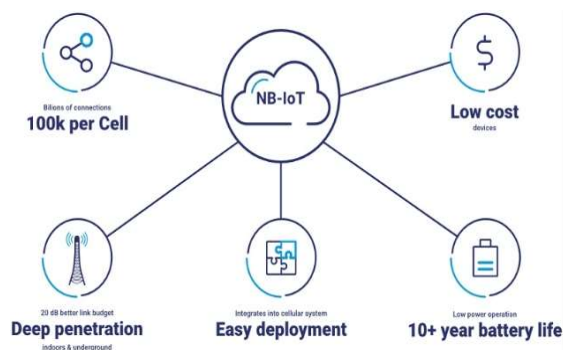
I. Réseaux cellulaires

(6/16)

D.2. NB-IoT (Narrowband IoT)

NB-IoT est une technologie de communication sans fil conçue spécifiquement pour l'Internet des objets (IoT).

- Avantages :**
 - Faible consommation d'énergie
 - Longue portée de transmission
 - Connectivité robuste
- Applications :**
 - Suivi d'actifs
 - Compteurs intelligents
 - Agriculture intelligente
- Évolutivité :**
 - NB-IoT est une solution évolutive pour les besoins croissants de l'IoT.



I. Réseaux cellulaires

(7/16)

D.3. M2M-GSM (2G)

M2M-GSM fait référence à la communication machine à machine utilisant la technologie GSM. Le GSM est utilisé où les appareils et les capteurs sont connectés pour échanger des données sans intervention humaine directe.

- ❑ **Ancienne technologie** : Le GSM est une technologie de deuxième génération (2G) qui a été largement utilisée dans le passé, mais qui est devenue obsolète dans de nombreuses régions en raison du déploiement de technologies plus avancées.
- ❑ **Débit de données limité** : Le GSM offre des débits de données relativement faibles, généralement adaptés aux applications M2M nécessitant un faible volume de données.
- ❑ **Faible consommation d'énergie** : Le GSM est relativement efficace en termes de consommation d'énergie, mais il n'est pas aussi optimisé que le NB-IoT en matière de faible consommation d'énergie.



I. Réseaux cellulaires

(8/16)

D.4. Comparaison entre les réseaux : LTE, LTE-M, GSM-M2M et NB-IOT

Caractéristique	GSM-M2M	LTE	LTE-M	NB-IoT
Génération	2G	4G	4G	LPWA
Débit de données	Faible (jusqu'à 236 Kbps)	Élevé (jusqu'à plusieurs Mbps)	Modéré (jusqu'à 1 Mbps)	Faible (jusqu'à 250 Kbps)
Consommation d'énergie	Modérée à élevée	Modérée à élevée	Faible	Très faible
Pénétration des obstacles	Modérée à bonne	Modérée à bonne	Bonne	Excellente
Portée de transmission	Bonne	Moyenne à bonne	Moyenne à bonne	Bonne à excellente
Prise en charge d'appareils	Modéré	Élevée	Modérée à élevée	Élevée
Coût de la connectivité	Modéré	Modéré	Modéré à faible	Faible
Vitesse de données adaptée à	Applications M2M	Applications gourmandes en données	Applications de surveillance et de suivi	Applications de faible bande passante
Applications typiques	Surveillance, télémétrie	Voix, vidéo, données riches	Surveillance, suivi, capteurs	Capteurs, compteurs intelligents, suivi d'actifs

I. Réseaux cellulaires

(9/16)

E. Shields IoT pour une Connectivité de Réseau Cellulaire



SIM900 GSM/GPRS Shield

Il offre une connectivité GSM/GPRS pour les projets Arduino et est compatible avec de nombreux modèles de cartes Arduino



SIM800L GSM/GPRS Shield

Ce shield est conçu pour les communications GSM/GPRS et est largement utilisé pour les projets IoT nécessitant une connectivité cellulaire.



Quectel EC25 Mini PCIe LTE Module

Bien qu'il s'agisse d'un module PCIe, il peut être utilisé avec un shield PCIe pour ajouter une connectivité LTE à un projet IoT.

I. Réseaux cellulaires

(10/16)

E. Shields IoT pour une Connectivité de Réseau Cellulaire



Quectel BC95-G NB-IoT Shield

conçu pour les applications NB-IoT, offrant une connectivité adaptée aux besoins de l'IoT à faible consommation d'énergie.



Particle Boron LTE

une carte de développement IoT intégrant un modem LTE pour des projets de bout en bout..



Pycom FiPy

Carte de développement IoT polyvalente qui prend en charge la connectivité cellulaire, y compris LTE-M et NB-IoT, en utilisant des modules Pycom

I. Réseaux cellulaires

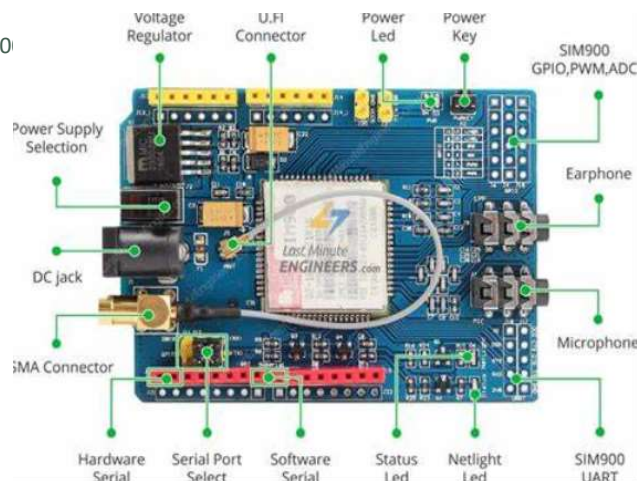
(11/16)

F. SIM900 GSM/GPRS Shield

Shield possède 12 GPIOs, 2 PWMs et un ADC du module SIM900 (ce sont tous des logiques 2V8) présents à bord.

Caractéristiques:

- Quadri-bande, compatible avec les réseaux GSM.
- Transmission de données efficace grâce au GPRS multi-slot.
- Puissance de transmission de Classe 4 (2 W @ 850/900 MHz) et Classe 1 (1 W @ 1800/1900 MHz).
- Envoi de SMS pour la transmission de données.
- Pile TCP/UDP intégrée pour le téléchargement de données sur un serveur web.
- Intégration d'une horloge temps réel (RTC).
- Faible consommation d'énergie en mode veille (1.5 mA).
- Compact et facile à intégrer dans des projets Arduino et IoT.



I. Réseaux cellulaires

(12/16)

G. Envoi d'un SMS via le shield SIM900 GSM/GPRS

Explication du code :

- 1.Utilisation la bibliothèque SoftwareSerial pour établir une communication série avec le SIM900 via les broches 7 (TX) et 8 (RX) de l'Arduino.
- 2.Dans la fonction setup(), nous initialisons la communication série avec le moniteur série et avec le SIM900 via mySerial. Nous attendons un certain temps pour que le module SIM900 s'initialise.
- 3.La boucle loop() est utilisée pour envoyer un SMS. Nous utilisons des commandes AT pour interagir avec le SIM900.
- 4.Nous envoyons la commande AT pour vérifier si le module est prêt.
- 5.Ensuite, nous configurons le mode SMS en mode texte avec la commande "AT+CMGF=1".
- 6.Nous envoyons la commande "AT+CMGS" pour entrer le numéro de téléphone de destination.
- 7.Ensuite, nous envoyons le contenu du message SMS.
- 8.Pour terminer le message SMS, nous envoyons le caractère Ctrl+Z (0x1A).
- 9.Nous attendons quelques secondes avant de répéter le processus.

I. Réseaux cellulaires

(13/16)

G. Envoi d'un SMS via le shield SIM900 GSM/GPRS

```
#include <SoftwareSerial.h>
SoftwareSerial mySerial(7, 8); // Crée un objet SoftwareSerial pour la communication avec le SIM900
void setup() {
  // Initialise la communication série avec le moniteur série
  Serial.begin(9600);
  Serial.println("Initialisation...");
  // Initialise la communication série avec le SIM900
  mySerial.begin(9600);
  // Attendez que le SIM900 s'initialise (cela peut prendre quelques secondes)
  delay(20000);
  Serial.println("Le SIM900 est prêt.");
}
void loop() {
  // Envoie la commande AT pour s'assurer que le module est prêt
  mySerial.println("AT");
  delay(1000);

  // Attendez la réponse du module
  while (mySerial.available()) { Serial.write(mySerial.read()); }
  delay(1000); // Attendez une seconde

  // Envoie la commande pour configurer le mode SMS
  mySerial.println("AT+CMGF=1"); // Mode texte
  delay(1000);

  // Attendez la réponse
  while (mySerial.available()) { Serial.write(mySerial.read()); }
  delay(1000); // Attendez une seconde

  // Envoie la commande pour entrer le numéro de téléphone de destination
  mySerial.println("AT+CMSS=\""+1234567890+""); // Remplacez +1234567890 par le numéro de téléphone de destination
  delay(1000);

  // Attendez la réponse
  while (mySerial.available()) { Serial.write(mySerial.read()); }
  delay(1000); // Attendez une seconde

  // Envoyez le message SMS
  mySerial.println("Ceci est un message SMS envoyé depuis Arduino !");
  delay(1000);

  // Attendez la réponse
  while (mySerial.available()) { Serial.write(mySerial.read()); }

  // Envoyez Ctrl+Z pour terminer le message SMS
  mySerial.write(0x1A);
  delay(1000);

  // Attendez la réponse
  while (mySerial.available()) { Serial.write(mySerial.read()); }
  // Attendez quelques secondes avant de répéter le processus
  delay(5000);
}
```

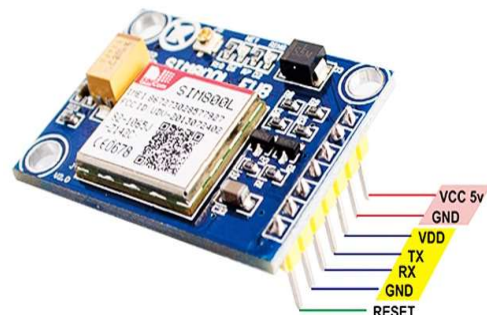
I. Réseaux cellulaires

(14/16)

H. SIM800L GSM/GPRS Shield

Caractéristiques:

- Support de données pour 4 fréquences de communication, utilisable à l'échelle mondiale.
- Port série TTL compatible avec les MCU de 3,3V et 5V, permettant une connexion directe.
- Alimentation de 5V, module USB pour le débogage, et port série TTL.
- Courant de sortie de l'alimentation de 800 mA.
- Fonctionnalités supplémentaires :
 - Ultra-petite taille adaptée à diverses applications, y compris les brassards et les systèmes embarqués.
 - Conception de fente latérale pour faciliter le remplacement de la carte SIM lors de la conception dans un boîtier.
 - Interface IPX offrant la possibilité de remplacer l'antenne (antenne PCB par défaut).



I. Réseaux cellulaires

(15/16)

I. Appel téléphonique en utilisant le shield SIM800L GSM/GPRS

Explication du code :

1.Nous utilisons la bibliothèque SoftwareSerial pour établir une communication série avec le SIM800L via les broches 7 (TX) et 8 (RX) de l'Arduino.

2.Dans la fonction setup(), nous initialisons la communication série avec le moniteur série et avec le SIM800L via sim800L. Nous attendons un certain temps pour que le module SIM800L s'initialise.

1.La boucle loop() est utilisée pour effectuer un appel téléphonique. Nous utilisons des commandes AT pour interagir avec le SIM800L.

2.Ensuite, nous utilisons la commande "ATD+1234567890;" pour appeler le numéro de téléphone spécifié (remplacez "+1234567890" par le numéro que vous souhaitez appeler).

3.Nous attendons un certain temps (par exemple, 30 secondes) pour l'appel.

4.Ensuite, nous raccrochons l'appel en envoyant la commande "ATH".

5.Nous attendons quelques secondes avant de répéter le processus.

I. Réseaux cellulaires

(16/16)

I. Appel téléphonique en utilisant le shield SIM800L GSM/GPRS

```

#include <SoftwareSerial.h>

SoftwareSerial sim8001(7, 8); // Crée un objet SoftwareSerial pour communiquer avec le SIM800L (TX sur 7, RX sur 8)

void setup() {
  // Initialise la communication série avec le moniteur série
  Serial.begin(9600);
  Serial.println("Initialisation...");

  // Initialise la communication série avec le SIM800L
  sim8001.begin(9600);

  // Attendez que le SIM800L s'initialise (cela peut prendre quelques secondes)
  delay(20000);

  Serial.println("Le SIM800L est prêt.");
}

void loop() {
  // Envoie la commande AT pour s'assurer que le module est prêt
  sim8001.println("AT");
  delay(1000);

  // Attendez la réponse du module
  while (sim8001.available()) { Serial.write(sim8001.read());}

  // Attendez une seconde
  delay(1000);

  // Envoie la commande pour appeler un numéro de téléphone
  sim8001.println("ATD+1234567890;"); // Remplacez +1234567890 par le numéro de téléphone à appeler
  delay(1000);

  // Attendez la réponse
  while (sim8001.available()) { Serial.write(sim8001.read());}

  // Attendez un moment (par exemple, 30 secondes) pour l'appel
  delay(30000);

  // Raccrochez l'appel en envoyant la commande ATH
  sim8001.println("ATH");
  delay(1000);

  // Attendez la réponse
  while (sim8001.available()) { Serial.write(sim8001.read());}

  // Attendez quelques secondes avant de répéter le processus
  delay(5000);
}

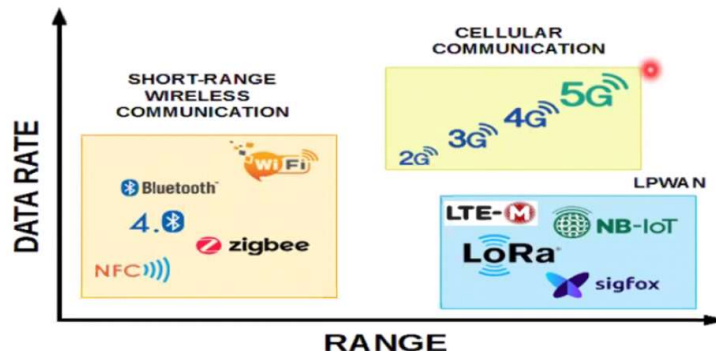
```


II. Réseaux sans fil

(1/15)

A. Importance de la Connectivité sans fil

La connectivité sans fil est incontournable dans l'IoT en permettant la communication et le contrôle à distance, assurant une flexibilité, une économie d'énergie et une évolutivité essentielles pour le déploiement massif d'appareils connectés.



II. Réseaux sans fil

(2/15)

II.I Wi-Fi dans l'IoT

A. Wi-Fi dans l'IoT - Technologie

Wi-Fi est une technologie de communication sans fil qui permet la connexion d'appareils à un réseau local sans utiliser de câbles physiques.



Caractéristique :



- Sans fil** : Élimine le besoin de câbles.
- Large portée** : Permet la connectivité à distance.
- Haute vitesse** : Prend en charge des débits élevés.
- Normes multiples** : 802.11n, 802.11ac, etc.

802.11b	Première version grand public, débit plus lent.
802.11g	Amélioration de la vitesse.
802.11n	Haute vitesse et portée améliorée.
802.11ac	Haute performance et largeur de bande.
802.11ax (Wi-Fi 6)	Haute efficacité et capacité.

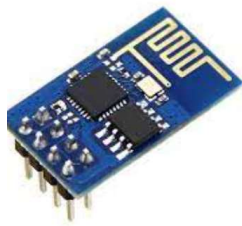
II. Réseaux sans fil

(3/15)

II.I Wi-Fi dans l'IoT

B. Shields Wi-Fi pour Arduino

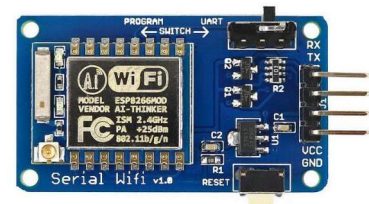
- ❑ **ESP8266** : Le module ESP8266 est largement utilisé pour ajouter une connectivité Wi-Fi à des projets Arduino ou autonomes. Il est doté d'un microcontrôleur intégré et offre une connectivité Wi-Fi abordable.



Module sans fil
WiFi ESP8266



Module sans fil
WiFi ESP-12N ESP8266



Module sans fil
WiFi ESP8266-07

II. Réseaux sans fil

(4/15)

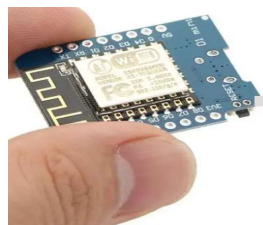
II.I Wi-Fi dans l'IoT

B. Shields Wi-Fi pour Arduino

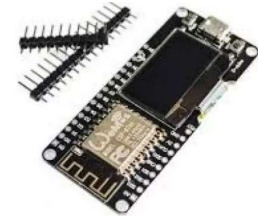
- ❑ **NodeMCU** : Il s'agit d'une carte de développement basée sur l'ESP8266. Elle offre une connectivité Wi-Fi et un grand nombre de broches GPIO pour connecter des capteurs et des actionneurs.



NODEMCU LUA - WIFI MODULE DE
CONNEXION WI-FI SÉRIE MODULE
W / INTÉGRÉ CP2102 DRIVER IC



CARTE DE
DÉVELOPPEMENT D1
MINI NODEMCU LUA



NODEMCU ESP8266 + 0,96
POUCES OLED CARTE DE
DÉVELOPPEMENT

II. Réseaux sans fil

(5/15)

II.I Wi-Fi dans l'IoT

B. Shields Wi-Fi pour Arduino

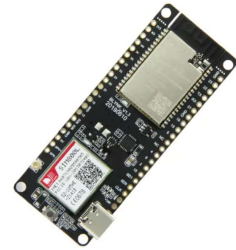
- ❑ **ESP32** : L'ESP32 est une évolution de l'ESP8266 avec une plus grande puissance de traitement, une connectivité Wi-Fi et Bluetooth intégrée, ainsi que de nombreuses interfaces matérielles. Il est adapté aux projets IoT plus avancés.



ESP32 WROOM32
MODULE 30 BROCHES
DOIT DEVKIT V1



ESP32 WIFI + CARTE DE
DÉVELOPPEMENT
BLUETOOTH 38 BROCHES



MODULE SANS FIL TTGO T-CALL V1.3
ESP32 ANTENNE GPRS MODULE
SIM800L DE LA CARTE SIM

II. Réseaux sans fil

(6/15)

II.I Wi-Fi dans l'IoT

C. Exemple de code d'Utilisation de ESP32 qui se connecte à un réseau Wi-Fi

```
#include <WiFi.h>

const char* ssid = "Nom_du_Réseau_Wi-Fi";
const char* password = "Mot_de_passe_du_Réseau_Wi-Fi";
const char* targetIP = "192.168.1.2"; // Remplacez par l'adresse IP de la machine B

void setup() {
  Serial.begin(115200);
  delay(1000);

  // Connexion au réseau Wi-Fi
  WiFi.begin(ssid, password);

  while (WiFi.status() != WL_CONNECTED) {
    delay(1000);
    Serial.println("Connexion en cours au réseau Wi-Fi...");
  }

  Serial.println("Connecté au réseau Wi-Fi");
  Serial.print("Adresse IP de l'ESP32: ");
  Serial.println(WiFi.localIP());

  // Envoie d'un message à la machine B
  sendMessage("Salut, je suis connecté à la machine B!");
}

void loop() {
  // Votre code ici
}

void sendMessage(const char* message) {
  WiFiClient client;

  if (client.connect(targetIP, 80)) {
    Serial.println("Connexion à la machine B réussie.");
    client.print("GET /send_message?message=");
    client.print(message);
    client.println(" HTTP/1.1");
    client.println("Host: example.com");
    // Remplacez par le nom de domaine ou l'adresse IP de la machine B
    client.println("Connection: close");
    client.println();
    client.stop();
    Serial.println("Message envoyé avec succès à la machine B.");
  } else {
    Serial.println("Échec de la connexion à la machine B.");
  }
}
```

II. Réseaux sans fil

(7/15)

II.2 Bluetooth dans l'IoT

A. Bluetooth dans l'IoT - Technologie

Bluetooth est une technologie de communication sans fil à courte portée qui permet la connexion d'appareils électroniques entre eux.



Caractéristique :

- ❑ **Courte portée** : Généralement jusqu'à 10 mètres.
- ❑ **Faible consommation d'énergie** : Idéal pour les appareils alimentés par batterie.
- ❑ **Connexions simultanées** : Plusieurs appareils peuvent être connectés en même temps.
- ❑ **Interopérabilité** : Prise en charge par une grande variété d'appareils.

Bluetooth 1.x	Première version, faible débit binaire.
Bluetooth 2.0	Introduit l'EDR (Enhanced Data Rate).
Bluetooth 3.0	Technologie HS (High Speed) pour le transfert de données.
Bluetooth 4.0	Bluetooth Low Energy (BLE) pour les appareils IoT.
Bluetooth 5.0	Améliorations de la portée et de la vitesse.

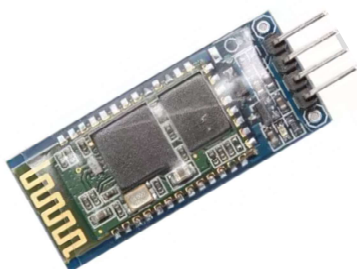
II. Réseaux sans fil

(8/15)

II.2 Bluetooth dans l'IoT

B. Shields Bluetooth pour Arduino

le contrôle précis de l'angle dans les projets de robotique et de mécanique.



Hc-06 du module sans fil bluetooth



ESP32-CAM WIFI + BLUETOOTH CAMÉRA MODULE



JDY-31 SPP-C MODULE BLUETOOTH

II. Réseaux sans fil

(9/15)

II.2 Bluetooth dans l'IoT

C. Exemple d'Utilisation du Module Bluetooth HC-06

HC-06 fonctionne en mode esclave (Slave) et est principalement utilisé pour établir une communication série sans fil avec d'autres appareils Bluetooth, comme un smartphone ou un ordinateur.

- HC-06 RXD -> Arduino TX (Pin 1);
- HC-06 TXD -> Arduino RX (Pin 0);
- VCC (HC-06) -> 5V (Arduino);
- GND (HC-06) -> GND (Arduino);

II. Réseaux sans fil

(10/15)

II.2 Bluetooth dans l'IoT

D. Code pour envoyer un message à un appareil Bluetooth connecté

```
#include <SoftwareSerial.h>

SoftwareSerial BTSerial(0, 1); // RX, TX

void setup() {
  Serial.begin(9600);
  BTSerial.begin(9600); // Définit la vitesse de communication Bluetooth

  Serial.println("En attente de la connexion Bluetooth...");
}

void loop() {
  if (BTSerial.available()) {
    char data = BTSerial.read();
    Serial.print(data);

    // Si vous recevez un caractère spécifique, envoyez un message
    if (data == 'A') {
      sendMessageToBluetooth("Message depuis l'Arduino vers la machine B.");
    }
  }
}
```

```
void sendMessageToBluetooth(const char* message) {
  BTSerial.println(message);
  Serial.println("Message envoyé via Bluetooth :");
  Serial.println(message);
}
```

II. Réseaux sans fil

(11/15)

II.3 Zigbee dans l'IoT

A. Zigbee dans l'IoT - Technologie

ZigBee est un protocole de communication sans fil conçu pour les réseaux de capteurs et les appareils à faible consommation d'énergie.

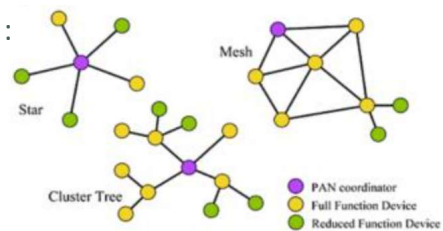
Caractéristique :



- Faible consommation d'énergie;**
- Réseaux maillés :** ZigBee permet la création de réseaux maillés auto-organisés;
- Fiabilité :** ZigBee est robuste et peut fonctionner dans des environnements perturbés;
- Faible débit binaire :** Convient pour la transmission de petites quantités de données.

	Zigbee IEEE 802.15.4	Zigbee IEEE 802.15.4	Zigbee IEEE 802.15.4a
Portée théorique	100m	1km	100m
Débit	250kbps	20kbps	250kbps
Bande de fréquence	2.4GHz	868MHz	2.4GHz
Localisation			X

Topologies :



II. Réseaux sans fil

(12/15)

II.3 Zigbee dans l'IoT

B. Shields Zigbee pour Arduino



XBee S2 2mW Zigbee
 Wireless Data Transmission
 Module 120 Meters For
 Arduino



ZigBee / CC2530
 Development Module



Arduino ZigBee Shield

II. Réseaux sans fil

(13/15)

II.3 Zigbee dans l'IoT

C. Exemple d'Utilisation du Module Zigbee

Le "Arduino ZigBee Shield" est conçu pour permettre la communication sans fil ZigBee avec des modules compatibles. ZigBee est un protocole de communication sans fil populaire utilisé pour la communication entre des appareils à faible consommation d'énergie.

Matériel requis :

- Deux modules ZigBee compatibles (par exemple, les modules XBee de Digi);
- Un Arduino avec le ZigBee Shield;
- Câbles de raccordement.

Vous aurez besoin de connecter le module ZigBee sur le bouclier avec les broches TX et RX (pour la communication série).

II. Réseaux sans fil

(14/15)

II.3 Zigbee dans l'IoT

D. Code d'Exemple Zigbee

```
#include <SoftwareSerial.h>

SoftwareSerial ZigBeeSerial(2, 3); // RX, TX (vous pouvez utiliser d'autres broches)

void setup() {
  Serial.begin(9600);
  ZigBeeSerial.begin(9600); // Configurez la vitesse de communication de votre module ZigBee

  Serial.println("En attente de la connexion ZigBee...");
}

void loop() {
  if (ZigBeeSerial.available()) {
    char data = ZigBeeSerial.read();
    Serial.print(data);

    // Si vous recevez un caractère spécifique, envoyez un message
    if (data == 'A') {
      sendMessageToZigBee("Message depuis l'Arduino vers la machine B.");
    }
  }
}

void sendMessageToZigBee(const char* message) {
  ZigBeeSerial.println(message);
  Serial.println("Message envoyé via ZigBee :");
  Serial.println(message);
}
```

II. Réseaux sans fil

(15/15)

II.4 Récapitulatif et Synthèse

Caractéristiques	Wi-Fi	Bluetooth	Zigbee
Portée	30 mètres à plusieurs centaines de mètres avec des routeurs Wi-Fi	jusqu'à 100 mètres	30 mètres à environ 100 mètres
Débit de données	Haut débit (jusqu'à plusieurs Gbit/s, dépend du standard)	Faible à moyen débit (jusqu'à 3 Mbit/s)	Faible à moyen débit (250 kbit/s à 1 Mbit/s)
Consommation d'énergie	Modérée à élevée (plus élevée pour les applications de diffusion en continu, mais optimisée pour les transferts de données importants)	Faible (optimisée pour les applications à faible consommation d'énergie)	Faible (optimisée pour les applications à faible consommation d'énergie)
Utilisation principale	Connexions haut débit dans des environnements avec une alimentation électrique stable (domotique, vidéosurveillance, appareils multimédias)	Connexions à faible consommation d'énergie pour les appareils mobiles (casques, écouteurs, capteurs)	Applications IoT à faible consommation d'énergie (domotique, capteurs, éclairage intelligent)
Protocole de communication	TCP/IP	Protocole propriétaire pour les applications de profil générique (ex : Bluetooth Low Energy)	Protocole Zigbee (IEEE 802.15.4)
Sécurité	Forte, avec des protocoles de sécurité avancés (WPA3, etc.)	Bonne, avec des protocoles de sécurité pour les connexions sécurisées	Bonne, avec des protocoles de sécurité Zigbee
Scalabilité	Réseaux étendus, mais avec des exigences en alimentation électrique	Connexions point à point ou point à plusieurs points	Réseaux maillés et la mise en réseau de nombreux appareils

Fin