WADE Mame Diarra
NETO Anete
EL AMRANI Lina

SAÉ 304 : Découvrir et mettre en place un réseau IoT

BUT R&T Parcours IoM

GIVRON Stéphane

Nord Franche-Comté
Belfort - Montbéliard
UNIVERSITÉ DE FRANCHE-COMTÉ

Introduction

L'internet des objets caractérise des objets physiques connectés ayant leur propre identité numérique et capables de communiquer les uns avec les autres.
Dans le cadre de cette SAE, l'objectif est de mettre en place une infrastructure IoT permettant à deux joueurs de jouer en réseau à Minecraft. L'interaction dans le jeu est rendue possible grâce à des gants connectés avec des boutons permettant de déclencher des super pouvoirs ou des malus contre l'adversaire. Mais pour mener à bien ce projet, il est essentiel de procéder avec une bonne gestion pour bien la structurer.
A cet effet, ce projet repose sur une approche agile, et plus précisément sur la méthode Scrum, afin de structurer et gérer efficacement le développement.
Tout d'abord nous allons expliquer la méthode de gestion projet appliquée pour cette SAE, ensuite expliquer chaque étape de sa réalisation et puis enfin parler des défis rencontrés.

Support pédagogique et cadre du projet

Le projet repose sur les enseignements théoriques et pratiques dispensés dans le cadre du cours IOM, notamment :

- La découverte des protocoles réseau et de l'IoT.
- L'initiation au protocole MQTT pour la communication entre dispositifs.
- L'intégration de scripts Python dans un environnement connecté.

Objectif principal : Développer une infrastructure IOM appliquée à un jeu interactif avec des objets connectés.

Gestion du projet

La gestion de projet agile est une une approche qui découpe un projet en différents sous-projets indépendants, appelés itérations représenté en boîte de temps nommés Sprint qui vont être répétées jusqu'à atteindre le résultat espéré.

La méthode de gestion de projet utilisée dans cette SAE est la méthode Scrum qui est une des méthodes de gestion de projet Agile.
Car c'est un ensemble de réunions, d'outils et de rôles qui interagissent de concert pour aider les équipes à structurer leur travail et à le gérer pour améliorer la productivité des équipes agiles même à distance, tout en permettant une optimisation du produit grâce à des feedbacks réguliers avec les utilisateurs finaux.
L'organisation est basée sur l'établissement des étapes de réalisation pour chaque user stories du product owner qui sont eux même réparti dans les Sprints.

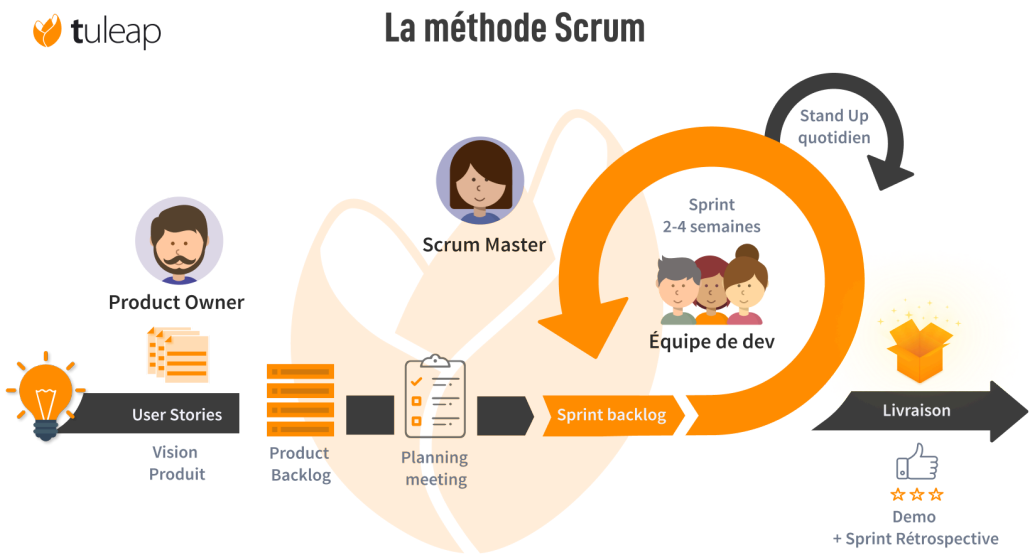

Image 1 : Structure de la méthodologie Scrum

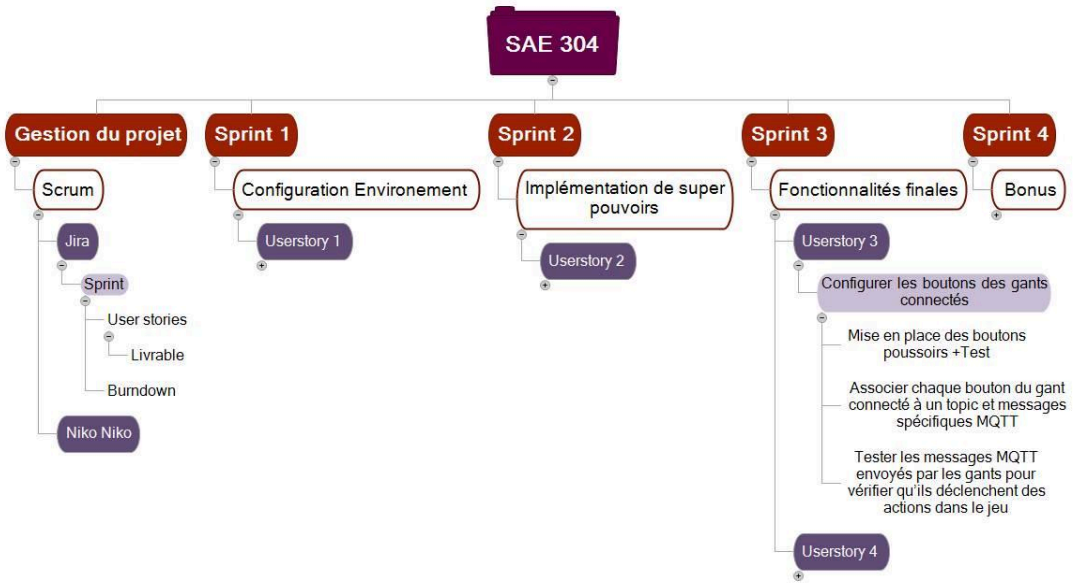

Image 2 : Carte mentale qui illustre le découpage en tâches

Ceci est organisé dans Jira qui grâce à ses fonctionnalités permet de collaborer efficacement au sein d'une équipe. Notamment avec la création, l'assignation et la suivie des tâches (tickets); la planification des sprints avec la création de sprint backlog (priorisation des tâches).

Image 3 : Structure du backlog du premier User story

De plus, en suivant la méthodologie Scrum, nous avons procédé à des sessions de Daily Scrum avec des réunions rapides par jour pour faire le bilan de ce qui est fait hier, ce qui est à faire et les obstacles à résoudre. Puis des revues de sprint parfois au product owner en lui présentant le travail accompli et puis recueillir les retours pour ajuster les prochaines étapes.

Et puis nous avons élaboré un Niko Niko où chaque membre de l'équipe exprime quotidiennement son état d'esprit à l'aide d'un emoji. Cela est efficace pour évaluer le moral et l'état d'esprit des membres pour identifier les problèmes et maintenir une ambiance de travail saine et productive.

	Lundi	Mardi	Mercredi	Jeudi	Vendredi
Lina	🙂	🙂	🙂	🙂	🙂
Anete	🙂	🙂	🙂	😓	🙂
Diarra	🙂	🙂	😢	😓	🙂

Image 4 : Niko Niko

Ainsi les rôles ont été réparties de la façon suivante :

Product Owner : Stéphane Givron

Définit les fonctionnalités nécessaires (sous forme de **User Stories**).

Scrum Master : Anete Neto

Guide l'équipe et s'assure que les principes Scrum sont respectés et puis facilite les réunions et encourage une collaboration efficace.

Équipe de développement : Anete Neto, Lina El Amrani, Mame Diarra Wade

Réalise les tâches définies dans le sprint.

Réalisation du Projet

➔ Infrastructure technique

- Matériel utilisé :
    - 2 Raspberry Pi 4, 2 ESP8266, 1 routeur Linksys, batteries, boutons, fils Dupont, platines d'expérimentation, PC (SSH)
- Logiciels :
    - Arduino IDE (1.8.19), VScode, Mosquitto (MQTT), Minecraft Pi, Mindview (cartes mentales).
- Protocoles et connexions :
    - Les ESP8266 transmettent des données aux Raspberry Pi via le protocole MQTT.
    - Les Raspberry Pi communiquent via une borne Linksys, connectée à Internet.

Explication du jeu

- **Objectif principal du jeu :**
  Chaque joueur doit explorer l'environnement Minecraft pour trouver et collecter un diamant. Le premier joueur à le récupérer remporte la partie.
- **Fonctionnement des gants connectés :**
  Les gants sont équipés de boutons connectés à un ESP8266 via MQTT, permettant d'activer des **superpouvoirs** ou des **malus**. Ces actions influencent directement l'environnement du jeu pour aider ou perturber les joueurs :
    - **Superpouvoirs :**
        - Creuser un trou.
        - Créer un ascenseur d'eau.
    - **Malus :**
        - Construire un mur de TNT ou de lave pour gêner l'adversaire.
        - "Méga-pouvoir" : créer un mur de glowstone.
- **Interaction dans le jeu :**
  Chaque bouton du gant est associé à une action spécifique via le protocole MQTT. Lorsqu'un bouton est pressé, l'information est transmise à une Raspberry Pi qui interprète la commande et modifie l'environnement du jeu en conséquence.

- **Victoire :**
  Lorsque l'un des joueurs collecte le diamant, un script Python détecte cet événement et annonce automatiquement la victoire, mettant fin à la partie.

➔ Découpage en sprint

Sprint 1 : Configuration de l'environnement et Premières functionalities

Objectif : Permettre à deux joueurs de se connecter à un réseau local, jouer ensemble dans Minecraft, et terminer une partie en collectant un diamant.

Date de réalisation : 6 - 7 janvier 2025

Étapes réalisées :

1. Configuration réseau : Mise en place d'un réseau local via le routeur Linksys pour connecter les Raspberry Pi.
2. Installation des Raspberry Pi :
   - Installation de Raspberry Pi OS (Bullseye).
   - Test de connectivité avec un ping entre les deux appareils.
3. Installation de Minecraft Pi :
   - Mise en place de Minecraft Pi Edition sur chaque Raspberry Pi.
   - Vérification du fonctionnement du jeu.
4. Création du serveur Minecraft :
   - Mise en place d'un serveur multijoueur sur l'un des Raspberry Pi.
   - Ajout manuel d'un diamant dans le jeu via un script Python.
   - Affichage des règles du jeu en anglais.
5. Fin de partie :
   - Détection de la collecte du diamant par un joueur via un script Python.
   - Affichage d'un message de victoire.

Image 5 : Règles du jeu Minecraft

Résultats :

- Connexion stable entre les Raspberry Pi.
- Interaction des deux joueurs dans Minecraft.
- Automatisation de la fin de partie avec succès.

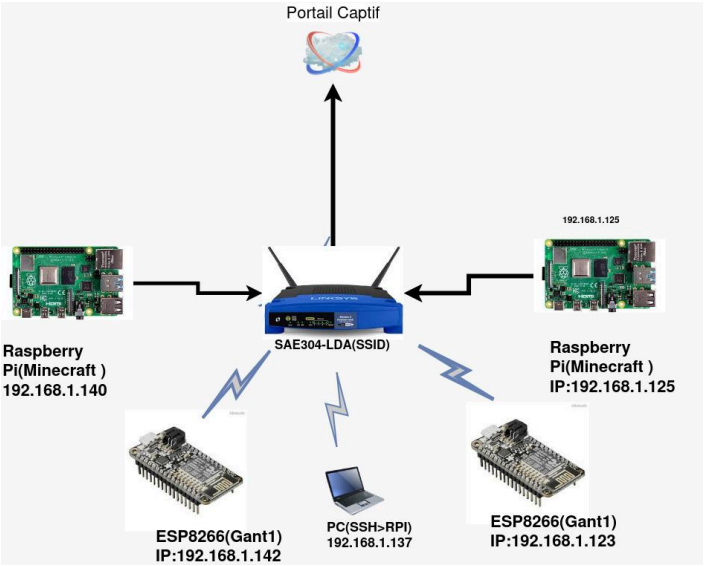

Image 6 : Infrastructure réseau

Sprint 2 : Implémentation des superpouvoirs

Objectif : Ajouter des fonctionnalités interactives activées via des gants connectés.

Date de réalisation : 7- 8 janvier 2025

Étapes réalisées :

- Mise en place et Programmation des ESP8266 :
  - Installation du support pour la carte *Adafruit HUZZAH ESP8266*
  - Test de connexion à notre réseau Wi-Fi
  - Transmission des données via MQTT aux Raspberry Pi.
- Intégration Python :
  - Développement de scripts pour déclencher des actions dans Minecraft.
  - Superpouvoirs ajoutés : creuser un trou, ascenseur d'eau
  - Malus ajoutés : mur de TNT, mur de lave (tous les 10 secondes)
  - Méga pouvoir : mûr de glowstone

Sprint 3 : Intégration des boutons poussoirs et interaction avec le jeu

Objectif :Permettre à un joueur d'interagir avec le jeu en utilisant des boutons poussoirs connectés à un ESP8266 via MQTT.

Date de réalisation : 9 janvier 2025

Étapes réalisées :

- **Connexion des boutons poussoirs** : Les boutons ont été connectés aux GPIO de l'ESP8266 et configurés pour détecter les pressions.
- **Association aux topics MQTT** : Chaque bouton a été associé à un topic MQTT unique, permettant l'envoi de messages spécifiques lors de l'appui.
- **Intégration dans le jeu** : Les messages MQTT ont été interprétés par le jeu pour déclencher des actions comme l'activation de super pouvoirs ou l'ajout de malus.

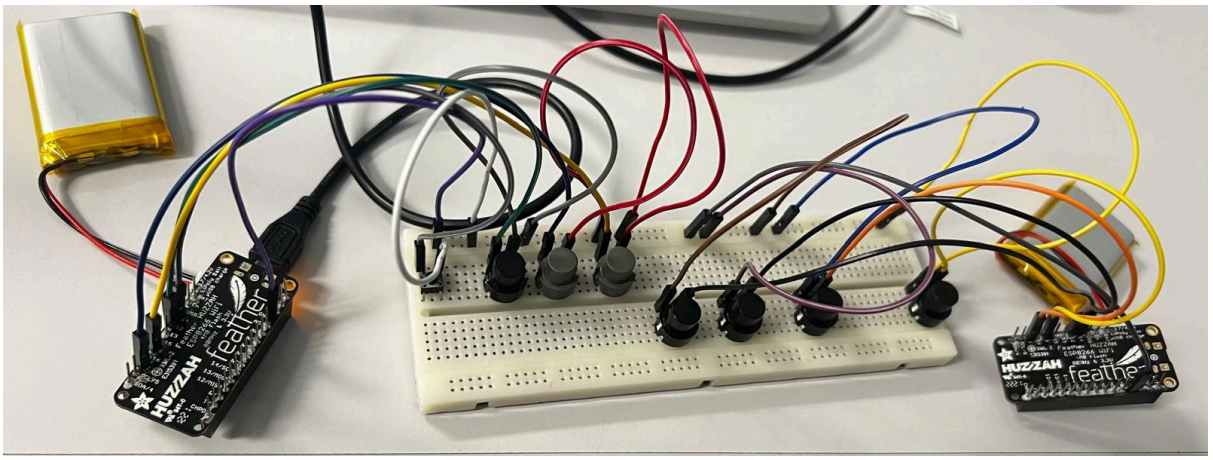

Image 7 : Connexion des boutons poussoirs aux microcontrôleurs via une platine d'expérimentation

- **Suivi de la distance par rapport au gagnant:** Modifier le script pour calculer la distance et la direction des joueurs par rapport au diamant.

Image 8 :Messages annonçant que le joueur s'approche ou s'éloigne du diamant.

Résultats :

- L'interaction avec le jeu a été validée par l'envoi de messages MQTT au moment de la pression sur chaque bouton.
- L'intégration de l'infrastructure IoT a permis de contrôler des actions dans le jeu via les boutons poussoirs connectés.

- Affichage de message indiquant aux joueurs leur proximité ou éloignement au diamant.

Sprint 4 : Bonus et optimisation

- **Sécurité du MQTT :** Mise en place de mots de passe pour les clients se connectant au broker.

  Cette étape vise à renforcer la sécurité des échanges sur le réseau MQTT en exigeant une authentification des clients.Les identifiants ont été générés à l'aide de la commande `sudo mosquitto_passwd.`

```
lidian@raspberrypi:~ $ sudo mosquitto_passwd -c /etc/mosquitto/passwords jouer
1
Password:
Reenter password:
lidian@raspberrypi:~ $ sudo mosquitto_passwd  /etc/mosquitto/passwords jouer2
Password:
Reenter password:
```

Image 9 :Création des identifiants pour les clients(ESP)

Dans le fichier mosquitto.conf, la connexion anonyme a été désactivée (allow_anonymous false), et le fichier contenant les mots de passe (passwords) a été spécifié pour l'authentification des clients.

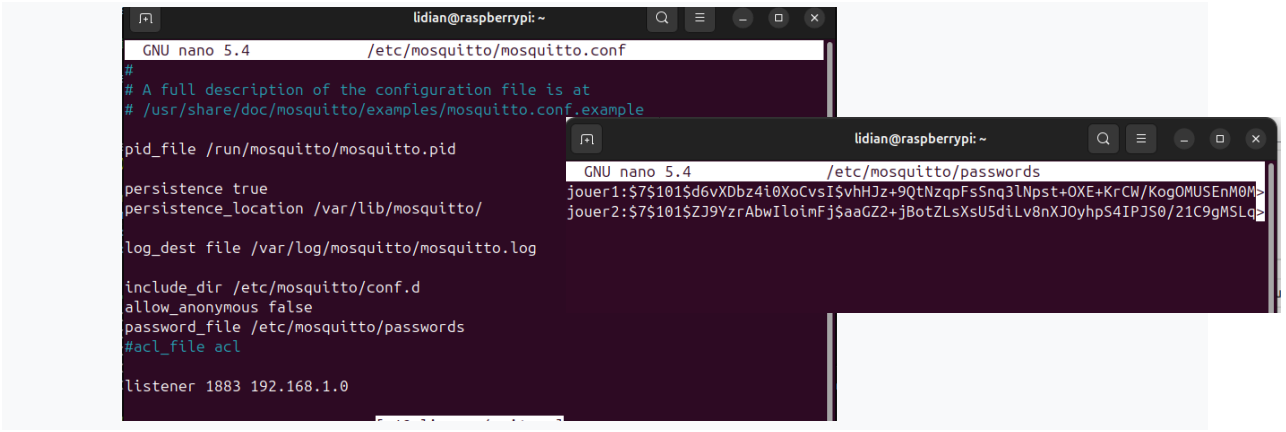

Image 10 :Fichier de configuration Mosquitto et fichier des mots de passe.

- **Test et optimisation des scripts pour réduire les temps de latence.**
- **Mettre en place un niko niko**

Défis rencontrés

Problème : Latence dans la transmission des messages MQTT.
Solution : Optimisation des scripts Python et configuration du broker Mosquitto.

Problème : Difficulté à se connecter au portail captif via borne linksys.
Solution : Révision de la configuration réseau et remplacement du matériel défectueux.

Problème : Gestion des conflits d'idées dans l'équipe.
Solution : Sessions de discussion pendant les rétrospectives pour résoudre les différends.

Conclusion

Ce projet nous a permis de découvrir et d'appliquer des concepts clés de l'IoT tout en renforçant nos compétences en programmation, en gestion de réseau et en méthodologie agile. Grâce à Scrum, nous avons pu organiser efficacement notre travail, gérer les imprévus et atteindre nos objectifs dans les délais.

Les défis rencontrés, comme les problèmes de connectivité ou d'intégration, nous ont appris à collaborer et à résoudre des problèmes de manière structurée. Cette expérience a été enrichissante tant sur le plan technique qu'humain et constitue une base solide pour nos futurs projets.

Références

1. **Documentation des blocs Minecraft Pi :**
   Dave's Motley Projects – Minecraft Blocks
   Utilisé pour comprendre les types de blocs disponibles dans Minecraft Pi et leur intégration dans le script Python.
2. **Tutoriel programmation python Minecraft Pi :**
   Raspberry Pi Projects – Getting Started with Minecraft Pi
   Servi de guide pour l'installation de Minecraft Pi et pour les premiers pas dans la programmation avec Python dans l'environnement Minecraft.
3. **Documentation bibliothèque paho-mqtt :**
   paho-mqtt 2.1.0
   Servi de guide pour l'implémentation du protocole mqtt en python en utilisant la bibliothèque paho-mqtt.