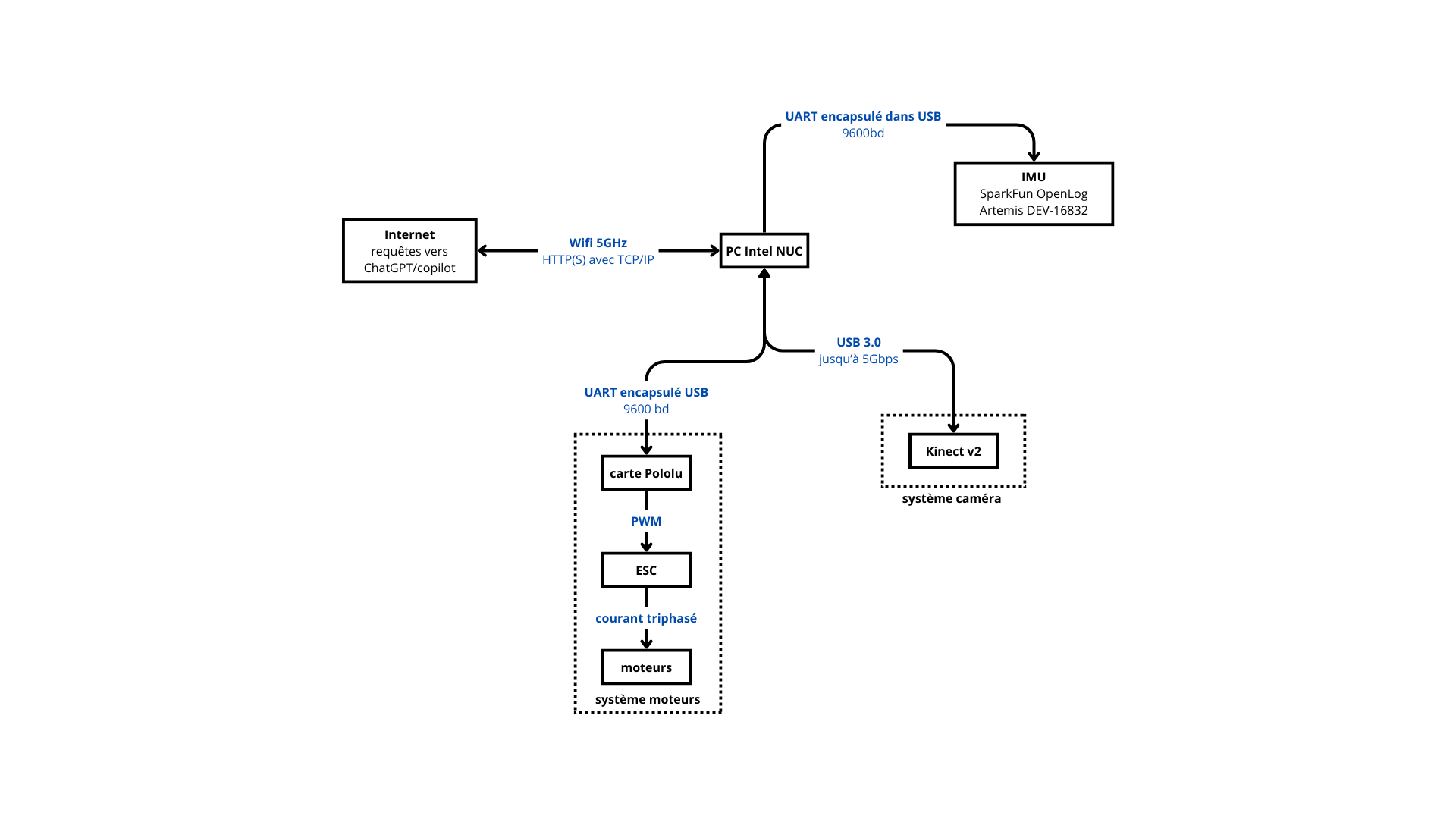
**partie 1 : architecture matérielle**



**partie 2 : architecture logicielle**

ros\_maestro\_ws/

├── src/

│ ├── **ros\_maestro**/ # Package pour la gestion de l'interface Maestro et des moteurs

│ │ ├── CMakeLists.txt

│ │ ├── package.xml

│ │ ├── launch/

│ │ │ ├── maestro\_launch.py

│ │ │ ├── test\_launch.py

│ │ ├── msg/

│ │ └── src/

│ │ ├── maestro\_driver.py # Contrôle de l'interface Maestro

│ │ └── relay\_driver.py # Gestion des relais

│ ├── **maestro\_node/**  # Package pour le nœud principal du robot

│ │ ├── CMakeLists.txt

│ │ ├── package.xml

│ │ ├── launch/

│ │ │ ├── gui\_control.launch

│ │ │ ├── maestro.launch

│ │ ├── src/

│ │ │ ├── maestro\_node.cpp # Logiciel pour contrôler le robot

│ │ │ └── PololuController.cpp

│ │ └── include/

│ │ ├── PololuController.h

│ │ └── ROSMaestroController.h

│ ├── **robot\_control/** # Package pour la gestion de la navigation et du contrôle du robot

│ │ ├── CMakeLists.txt

│ │ ├── package.xml

│ │ ├── launch/

│ │ │ └── navigation.launch

│ │ ├── msg/

│ │ │ ├── NavigationCmd.msg # Commandes pour la navigation

│ │ └── src/

│ │ ├── navigation\_node.py # Code pour la navigation autonome

│ │ ├── map\_builder.py # Code pour la cartographie SLAM

│ │ └── imu\_driver.py # Code pour l'IMU

│ ├── **robot\_ai/**  # Package pour l'IA et l'intégration ChatGPT

│ │ ├── CMakeLists.txt

│ │ ├── package.xml

│ │ ├── launch/

│ │ │ └── ai\_control.launch # Lancement de l'IA

│ │ └── src/

│ │ ├── ai\_controller.py # Gestion de l'IA (ex. ChatGPT)

│ │ └── sensor\_data\_handler.py # Intégration des capteurs pour l'IA

│ ├── **robot\_sensors/**  # Package pour les capteurs (IMU, caméra, etc.)

│ │ ├── CMakeLists.txt

│ │ ├── package.xml

│ │ ├── launch/

│ │ │ └── sensor\_launch.py

│ │ └── src/

│ │ ├── imu\_sensor.py # Interface avec le capteur IMU

│ │ └── camera\_sensor.py # Interface avec la caméra

│ ├── **robot\_msgs/** # Package pour les messages ROS

│ │ ├── CMakeLists.txt

│ │ ├── package.xml

│ │ └── msg/

│ │ └── SensorData.msg # Message contenant les données de capteur

└── CMakeLists.txt # Fichier CMake global pour l'ensemble du workspace

Communication entre les modules :

1. **robot\_sensors ↔ robot\_control** :
   * Les capteurs (IMU, caméra) envoient leurs données aux nœuds de contrôle du robot pour la navigation et la cartographie (par exemple, imu\_sensor.py envoie des données à map\_builder.py).
   * Les données IMU seront aussi utilisées pour l'odométrie et la stabilisation.
2. **robot\_control ↔ maestro\_node** :
   * Une fois la trajectoire et la commande de mouvement décidées, le package robot\_control envoie les commandes à maestro\_node pour contrôler les moteurs via le contrôleur Maestro.
   * La communication se fait via des messages ROS qui contiennent des informations sur les commandes de vitesse ou de direction.
3. **robot\_ai ↔ robot\_control** :
   * L'IA peut influencer les décisions de navigation du robot. Par exemple, si l'IA doit éviter un obstacle, elle envoie des commandes spécifiques au nœud de navigation.
   * ai\_controller.py récupère des informations de navigation et peut ajuster les objectifs du robot en fonction de l'analyse des capteurs ou de l'environnement.
4. **robot\_ai ↔ robot\_sensors** :
   * L'IA peut également utiliser les informations des capteurs pour analyser l'environnement. Par exemple, les données de la caméra ou des capteurs IMU peuvent être envoyées à l'IA pour détecter des obstacles ou des points d'intérêt.
5. **Communication entre nœuds via ROS2 topics, services et actions** :
   * Les différents modules communiqueront principalement via des **topics** pour l'envoi de messages (par exemple, les données des capteurs ou les commandes de mouvement).
   * Des **services** peuvent être utilisés pour des demandes spécifiques (comme la demande de localisation ou de données précises).
   * **Actions** peuvent être utilisées pour des tâches complexes (comme la navigation autonome ou le suivi d'un objectif).

Partie 3:

* Finir la conversion du code en ros 2
* Connecter tous les packages entre eux
* Implémenter les instructions de chatgpt à ROS2
* réussir à faire fonctionner