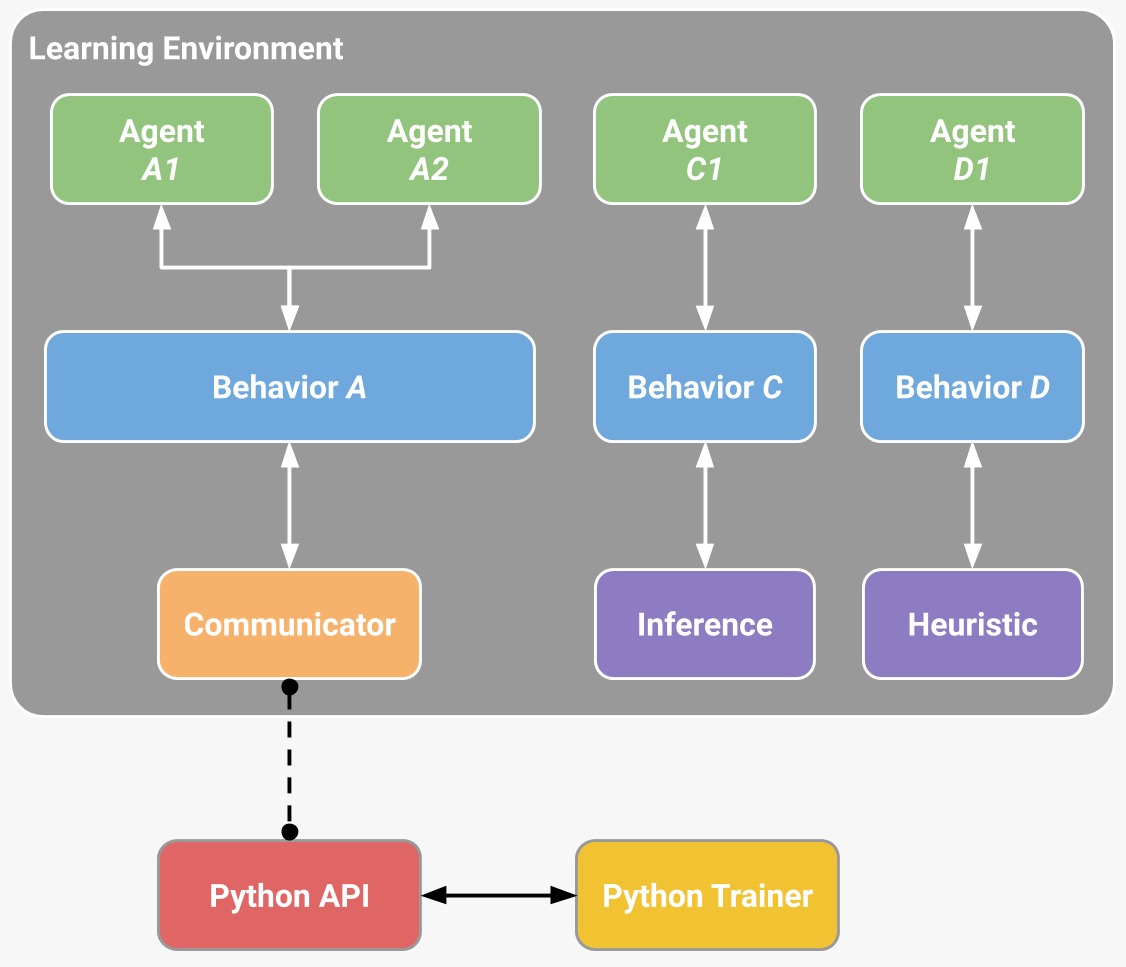
**CHAPTER 3**

**IMPLEMENTATION**

**3.1 System Architecture**

**Figure 3.1: System Architecture**

The system architecture for AI that learns to walk autonomously, as illustrated in Figure 3.1, involves a multi-layered approach integrating various agents, behaviors, and communication protocols. At the top level, the learning environment consists of multiple agents, such as Agent A1, A2, C1, and D1, each tasked with exploring different aspects of locomotion. These agents interact with specific behaviors—Behavior A, C, and D—that define their actions and responses.

Central to the architecture is the communicator module, which serves as the bridge between the learning environment and the external components. The communicator collects data from the behaviors and agents, ensuring seamless information flow.

This data is then processed through different modules such as inference and heuristic. The inference module is responsible for making real-time predictions and decisions based on the input data, while the heuristic module incorporates predefined rules and strategies to guide the learning process.

The Python API plays a crucial role in interfacing the learning environment with the Python Trainer. The API facilitates communication, allowing the trainer to send commands, receive data, and monitor the learning progress. The Python Trainer oversees the training process, adjusting parameters and updating the model based on the feedback received from the environment.

This hierarchical and modular architecture ensures that the system can handle the complexity of autonomous learning. Each layer is designed to process specific types of information, from the raw sensor data handled by agents to the high-level decision-making in the inference module. By structuring the system in this way, researchers can isolate and optimize different components, leading to a more efficient and adaptable learning process.

This architecture not only enhances the robot’s ability to learn walking autonomously but also allows for scalability and integration of additional behaviors and agents in the future.

**3.2 Tools and Technologies Used**

The ML-agents toolkit contains five high-level components:

* Learning environment –
  + Which contains the unity scene and all the game characters.
  + The unity scene provides the environment in which agents observe, act, and learn.
  + Depends on your goal. Trying to solve a reinforcement learning problem of limited scope,(for both training and for testing trained agents
  + In this case, it might be more efficient and practical to create a purpose-built training scene.
  + Includes an ml-agents unity sdk (com.Unity.ML-agents package)
  + That enables you to transform any unity scene into a learning environment by defining the agents and their behaviors.