5) *Food packaging:* Delta robots are utilized for food packaging tasks involving fish fillets and poultry fillets. Distributed architecture is applied in both cases for efficient packaging.

A. Key Observations and Findings:

The agricultural robotics domain exhibits a diverse range of software architectures to suit specific tasks and challenges as follows.

1) *Sense–plan–act:* The sense–plan–act architecture is frequently employed in various agricultural applications, indicating its adaptability and effectiveness in real-time decision-making.
2) *Distributed architectures:* Distributed architectures are prevalent, especially in horticulture and food processing applications, showcasing the importance of decentralized control and communication.
3) *Cloud-based architectures:* Cloud-based architectures are used for tasks that involve big data processing and monitoring, underscoring the role of cloud technology in modern agricultural robotics.
4) *IoT integration:* IoT-based architectures are utilized for monitoring and data collection in animal husbandry, highlighting the integration of IoT devices in agriculture.
5) *Application-specific solutions:* Different crops, animals, and tasks necessitate tailored software architectures, emphasizing the need for customization in agricultural robotics.

These findings illustrate the adaptability and versatility of software architectures in addressing the unique challenges posed by agricultural robotics, where specific mission requirements often dictate the choice of architecture.

VI. Case Studies

In order to examine the potential of appropriate software architecture, we will take into account three diverse use cases into our case studies in the agrofood sector, which takes into account most of the requirements and challenges across the domain. The three use cases chosen for the understanding and development of robotic architectures are harvesting robots, food processing robots, and food packaging. These use-case projects have some similar requirements in terms of functionality and performance as well as different requirements. In the subsequent sections, we are going to explore and exploit similarities in software and hardware architectures to be needed in all three use cases.

A. Harvesting Robot

In this case study, a harvesting robot was considered to designed and developed to automate the process of picking tomatoes at the green house scenarios. The robot should be designed to satisfy the requirements, such as precision manipulation, sensing capabilities, real-time decision-making, adaptability to varying crop conditions and challenges. However, there are challenges, such as handling delicate fruits, varying crop conditions, efficient harvesting, with minimal crop damage.

Based on the requirements, the behavior-based architecture would be a suitable choice for the harvesting robot, which is a popular choice for robotic systems that require real-time responsiveness and complex behaviors. It emphasizes the organization of the system into individual behaviors or modules, each responsible for a specific task or behavior. These behaviors can operate concurrently and independently, allowing the robot to exhibit reactive and adaptive behavior in response to its environment. In the case of the harvesting robot, the behavior-based architecture can be designed to include specific modules or behaviors such as the following.

1) *Perception module:* This module is responsible for perceiving the environment using various sensors, such as cameras or proximity sensors. It collects sensory data related to tomato detection, ripeness estimation, etc. It may also be used for autonomous navigation and robot control.
2) *Behavior modules:* These modules process the sensory data and generate desired actions based on predefined behaviors. Each behavior module focuses on a specific task, such as locating ripe tomatoes, avoiding obstacles, or navigating through rows.
3) *Motion module:* This module receives the desired actions from the behavior modules and generates motion commands. It considers factors, such as robot kinematics, dynamics, and environmental constraints, to plan and execute the robot's movements.
4) *Control module:* The control module receives the motion commands from the motion module and generates control signals to actuate the robot's motors and actuators. It ensures precise and coordinated movement of the robot, controlling aspects, such as velocity, acceleration, and joint angles.
5) *Gripping and cutting module:* This module handles the gripping and cutting mechanisms of the robot using grippers, cutting tools, etc. It receives control signals from the control module to perform precise manipulation and cutting actions on the tomatoes.
6) *Communication interface:* This module provides a means of communication between the robot and external systems. It allows for remote monitoring, control, and coordination with a central control unit or user interface. It can use protocols, such as message queue telemetry transport (MQTT), robot operating system (ROS), or custom interfaces for exchanging data and commands.

In a behavior-based architecture, each module operates autonomously, focusing on specific tasks and behaviors shown in Fig. 1. They communicate with each other through well-defined interfaces, exchanging sensory data, desired actions, motion commands, and control signals. This modular and decentralized approach enables:

1) reactive and adaptive behavior;
2) quick response to environmental changes;
3) modular and decentralized approach for flexibility, allowing for efficient tomato harvesting while handling