

Agent-Based Processing in Automated Manufacturing

Introduction

Agents are programs, sometimes small, sometimes large, tasked with doing one thing, or at most a limited number of things, but in the context of playing a part in a coordinated effort to achieve a larger goal. How coordination happens among agents, how each agent knows what its assigned task is, and how goals are measured, is foundational to an agent-based processing environment. Breaking down complex processes into smaller ones allows for scaling and increased capacity for further additional complexity. In addition to doing assigned tasks, agents created in our ADIN™ Automation Platform can update their processing parameters in response to current aspects of their data and processing environment which allows them to be adaptive. Being adaptive is an AI trait where knowledge gained over time is learned and used to improve performance both in terms of individual agents, as well as in achieving the greater goal. Adaptive Agents like those in our ADIN Automation Platform use techniques that are categorized in AI as Reinforcement Learning.

In this paper, we are addressing how automated manufacturing benefits from agent-based processing environments like our ADIN Automation Platform, where three approaches are presented. The first uses agent-based processing in an embedded microprocessor environment, the second shows how our traditional server-based container system can be applied to manufacturing applications and third, will show how to transform and improve existing manufacturing systems based on PLCs (Programmable Logic Controllers) configured using Ladder Logic by swapping in new approaches using agent-based processing with IoT devices called Sensor Control Modules (SCMs).

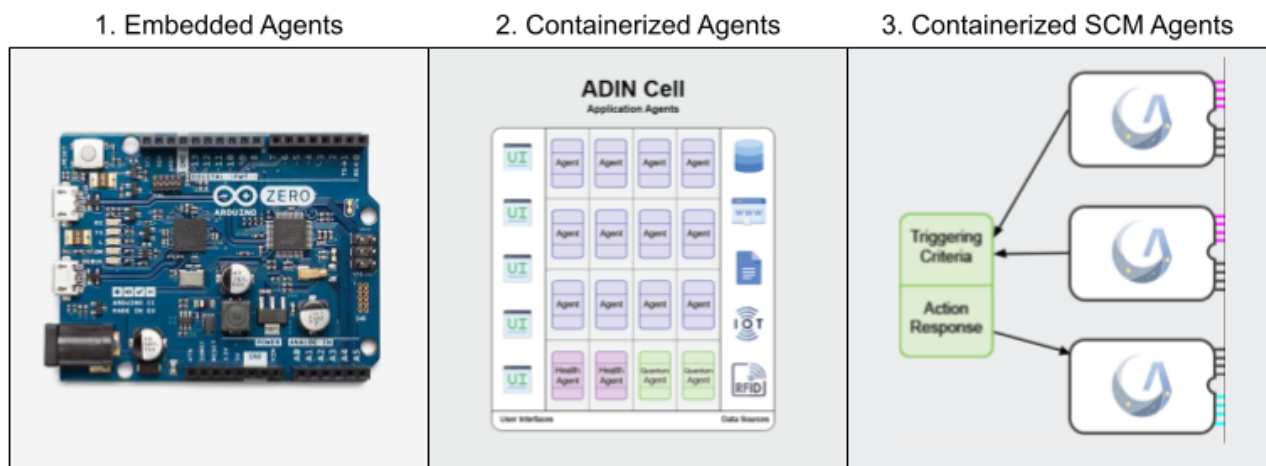


Figure 1. Types of AI Automated Manufacturing processing types

To briefly review the ADIN AI Automation Platform, agents are programs that are configured with a triggering criteria and action response. The set of triggers and actions are preset, as well as expandable within the ADIN Automation Platform. Triggers are based on timing, location, and/or data. Agents triggers

Agent-Based Processing in Automated Manufacturing

are checked on a timer, on a schedule or on-demand. For Agents with access to GPS location either from a database or device, they are triggered on specific locations, upon distance travelled or crossing into or out of geo-fenced regions. Data triggers occur when a data field becomes a particular value, such as a status field becomes 'Ready,' changes value, or a new record is detected. Furthermore, triggers can be a combination of time, location, and/or data triggers. To configure an agent to fire when a new record is created, but only within a particular time window, and only when the status of another agent is set to 'Ready' is an example of a combination trigger.

The ADIN AI Automation Platform has a REST API which means programs written in any language can create and configure hundreds or thousands of agents across a large multidimensional parameter space. Agents run in Docker Containers. This means managing large sets of ADIN agents is limited only to processing compacity and using systems like Kubernetes for container orchestration.

Once an Agent trigger is activated, agents execute action responses based on their configuration, which like triggering criteria is preset as well as expandable. Actions can be a new record in a database is inserted or updated. Notifications, where a templated message can be sent to specific people via any digital format, such as email or text message. Commands can be sent to IoT devices or can update and control other ADIN Agents. Actions can be combined, where many actions can be controlled from a single agent trigger or controlled via virtual state machine data graphs where the action first performs a function followed by a state machine update.

A central web-based User Interface allows users to browse and control their ADIN Agents which are grouped into ADIN Cells as shown in the figure below.

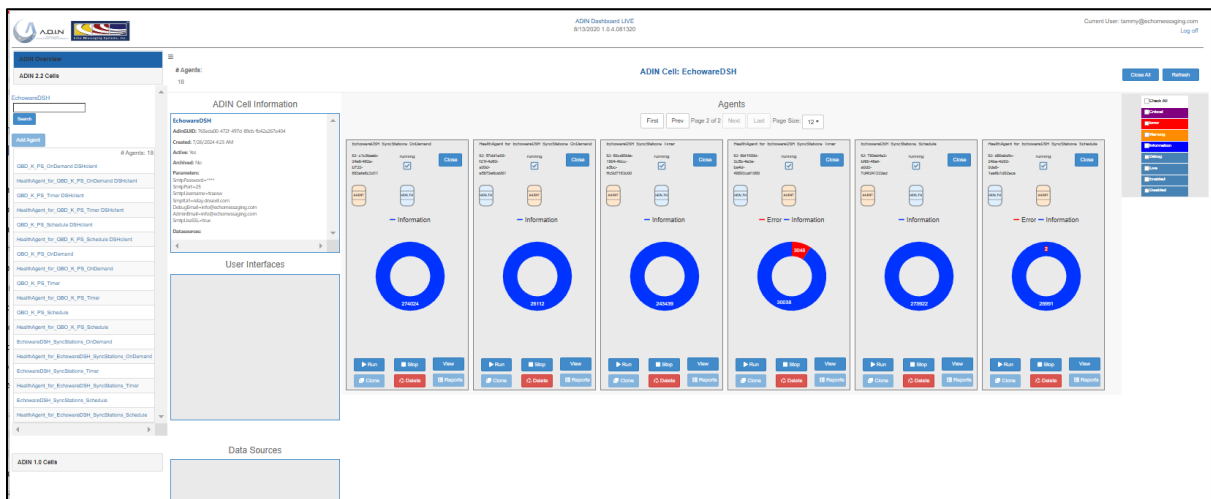


Figure 2. ADIN™ User Interface

Embedded Agents

The ADIN agent source code was ported into the Arduino C++ variant where about a dozen agents were configured for trigger criteria and action responses and then run in a single processing thread. This is a constrained version of the usual ADIN agent, where they run in a fully isolated containerized environment. On a typical Arduino single core board, only one processing thread can be run, and per Arduino's convention the function `setup()` is first called, followed by repeated calls to the `loop()` function. ADIN agents are configured in `setup()` where a separate ADIN Agent library is linked into the main program. Depending on the sensors and controls used by Arduino board design, each station (see 'st #' in Figure 3 below) has an associated ADIN agent C++ object, configure according to each stations' needs.

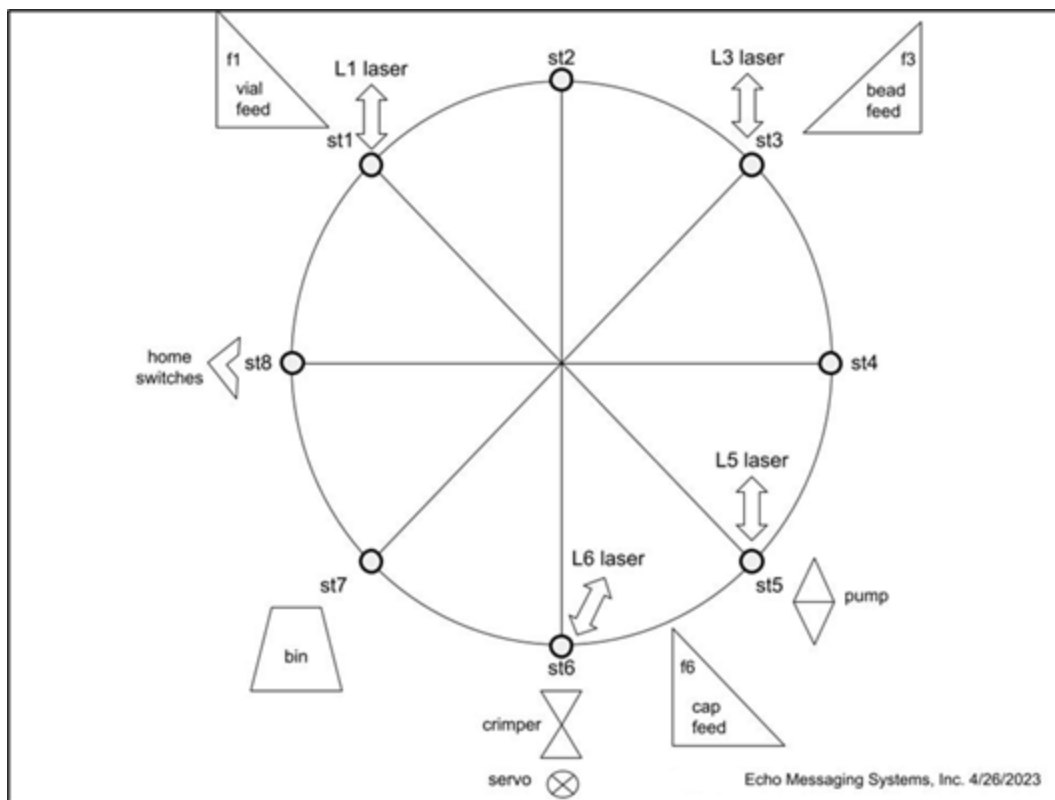


Figure 3. Arduino ADIN Agents for each designated station

The Arduino board in this example had laser inputs and component feeds with motorized hoppers to keep components moving into a gravity-fed feed, several device motors, and a servo. An additional motor controlled a spinning platter turning it clockwise, and it had its own agent to monitor global information, where each stations' actions needed to successfully complete to meet its trigger criteria.

Table 1 shows how this was accomplished in terms of each station's triggering criteria and action response.



Station ID	Description	Trigger Criteria	Action Response
st1	Component 1 Feed	Is Feed full? L1 Laser switches closed?	No – turn on hopper, Yes- turn off hopper, F1 Motor until both L1 Laser switches open
st2	Empty – Future use		
st3	Component 2 Feed	All components present? L3 Laser switch	No – indicate missing component for future steps, F3 Motor
st4	Empty – Future use		
st5	Liquid pump	All components present? Previous station statuses and L5 Laser switch	Yes – run pump motor for preset time, P5 Pump
st6.1	Component 3 Feed	Is Feed full? L6 Laser switch	No – turn on hopper, Yes- Enable Servo to push vial into position for Crimper
st6.2	Motorized crimper	All components present? Previous station statuses	Yes – run motor for preset time, record crimp success/fail results, C6 Crimper
st8	Motorized platter	Crimp successful? Analyze feedback profile captured during Crimp	Yes – full 1/8 th turn platter forward; item drops into bin. No – partial platter forward, sound alarm for manual removal and inspection.

Table 1. Configuration for Arduino C++ Embedded Agents for each station

Every iteration through Arduino’s loop() function, each agent’s triggering criteria was evaluated. Some stations required timing based on time intervals where others had a continuous interval. For example, stations that operated a motor had separate on and off stages that were based on specific time intervals. Motors that fed component hoppers could turn on or off at any time based on the state laser switches points at component feeds. This meant that there were two main processing loops, one for continuous events and another for interval events.

This overall approach had many benefits. The original system was never able to correctly perform the timing between the various steps to manufacture the items. By identifying stations and treating each station individually, we successfully determined specific timing requirements to provide the optimal pumping action, servo control, and crimping control. Once each station was running optimally, we increased overall throughput resulting in greater quantities produced. This system replaced manual labor, whose jobs transformed into overseeing the automated manufacturing devices, responding to ‘failed crimp’ events, and making sure feeds were fully stocked components.

Containerized Agents

Containerize Agents are how ADIN agents normally run, where agents are configured for triggering criteria and action responses where are stored in a central database. An agent is created and run in a Docker (or any compatible) container. Each agent benefits from its own isolated processing environment and therefore can have timing control specific to their agent’s parameters.

All agents have an associated Health agent whose function is to monitor how the agent is running and will log/report error conditions. This information is available in a web-hosted User interface shown in Figure 2. In non-manufacturing environments, we have connected ADIN agents to a Quantum Computer

Agent-Based Processing in Automated Manufacturing

(DWave Systems) as well numerous API's for financial, scheduling, routing, and many other types of applications.

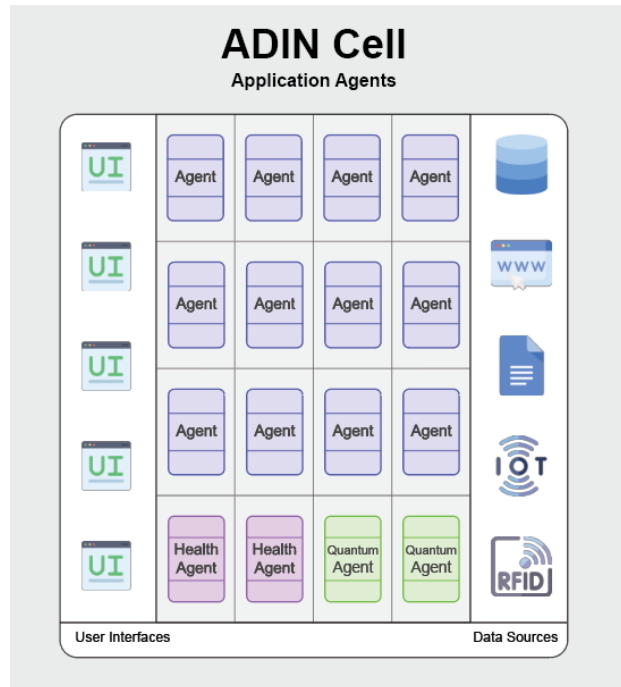


Figure 4. ADIN agents run in a collective known as an ADIN Cell to achieve a unifying goal.

Containerized SCM Agents for Replacing Ladder Logic and PLC's

Extending ADIN Agents, run from within containers, into the manufacturing world of industrial automation involves using a bridge device called the ADIN 'Sensor Control Module' (SCM) which reads input sensors as triggering criteria and controls output actuators as action responses. The ADIN SCM means that we can combine the best of the running containerized agents in manufacturing environments, where SCM's are treated as IoT devices, and ADIN SCMs are addressable via various networking options such as Bluetooth, wired and Wi-Fi.

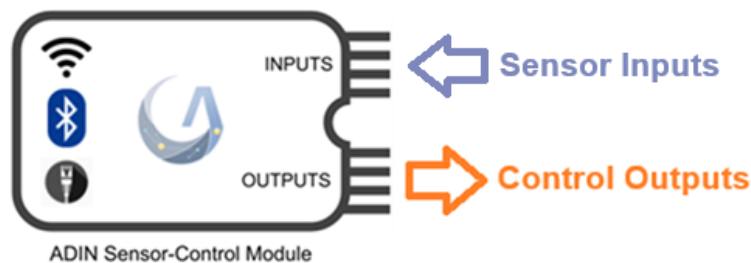
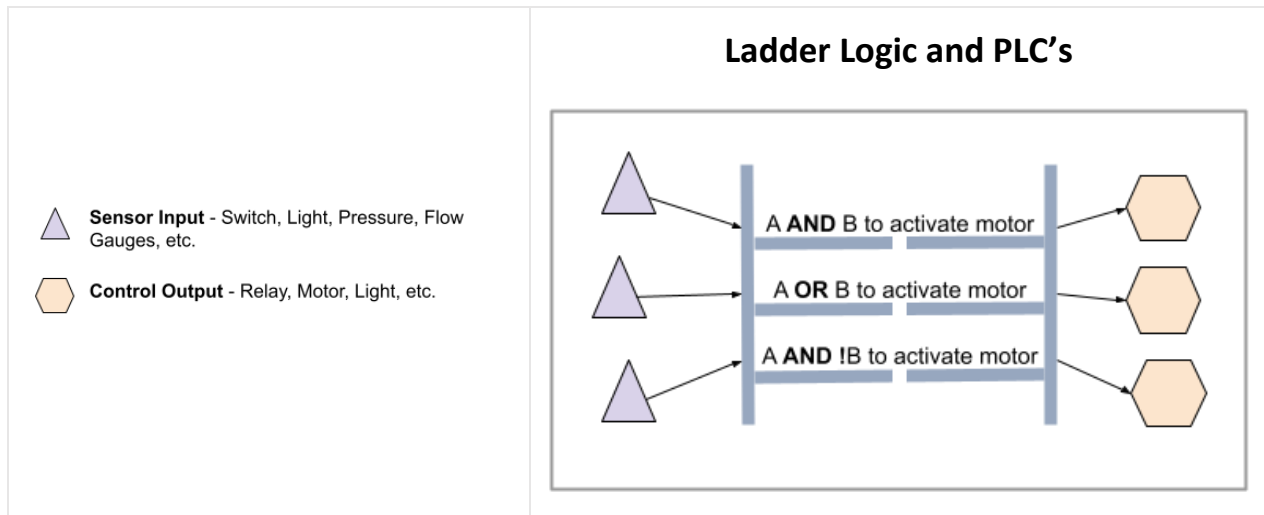


Figure 5. SCM - Sensor Control Module to connect ADIN Agents to hardware.

Currently in widespread use are PLCs (Programmable Logic Controllers) that are configured using Ladder Logic to read sensors and control actuators in manufacturing environments. This approach is

quickly becoming obsolete because control operations are simple and limited and does not allow for enhancements offered in the current climate of AI and quantum processing technologies.



The above diagram shows how a ladder of logic is formed. Each rung has a logical equation that when it resolves to true, based on the status of input sensors (shown as purple triangles), the rung (shown as blue lines) becomes whole representing electricity flowing from left to right to the other side to control the output devices (shown as orange hexagons).

ADIN agents connect to SCM's via network addressable connection that can be configured using secure networking devices via Bluetooth, Wi-Fi or wired. Each SCM has input sensor ports and output controls ports that are used in the triggering criteria and action responses of ADIN Agent configuration. As discussed in the first processing approach, embedded agents interact directly with sensors and controls which provide enhanced control and timing. Containerized agents benefit from isolated processing. By developing the ADIN SCM, we bring together the best of both worlds, where manufacturing can employ agent-based processing and PLC's can be swapped out for SCM's in existing production systems.

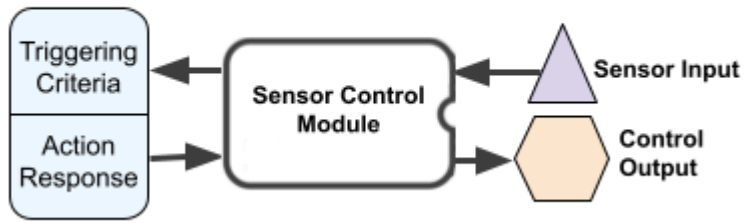
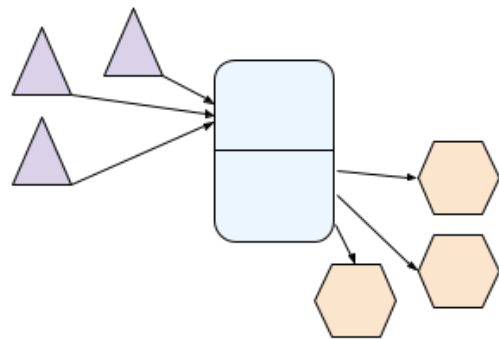
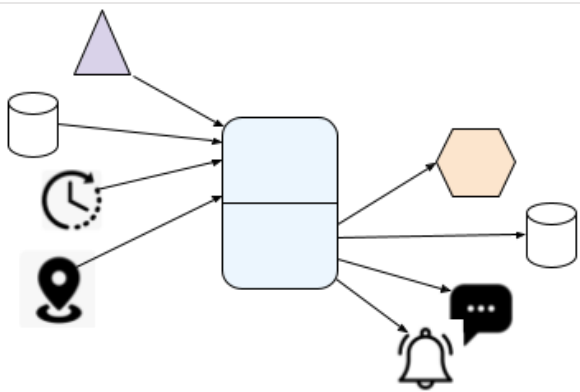
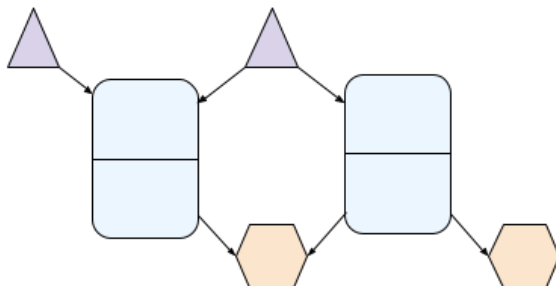
This approach means that current investments in manufacturing tools and industrial automation stay in place and the PLC's configured via Ladder Logic are replaced with ADIN Agent-controlled SCM's networked into either a local server or remote service running Docker (or any type of) Containers with container orchestration software.

 Sensor Inputs	 Control Outputs
Digital and Analog Switches	Relays, Switches, Solid State
Sound, Light, Temperature Sensors	Electric Motors
Strain, Pressure, Force, Flow Gauges	Hydraulic Motors
Proximity Sensor, Accelerometers	Pneumatic Motors
<i>others</i>	<i>others</i>

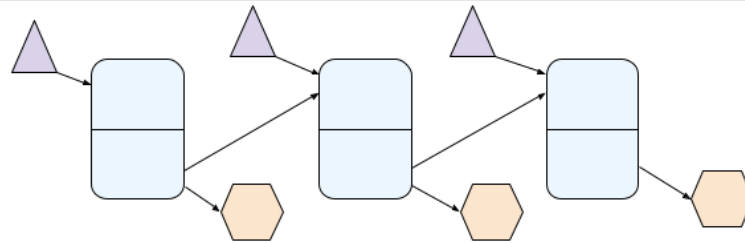
Table 2. Types of Sensors and Controls that ADIN agents connect to.

The following diagrams show the many different ways that ADIN Agents and SCMs connect to sensor input and control outputs for manufacturing automation systems.

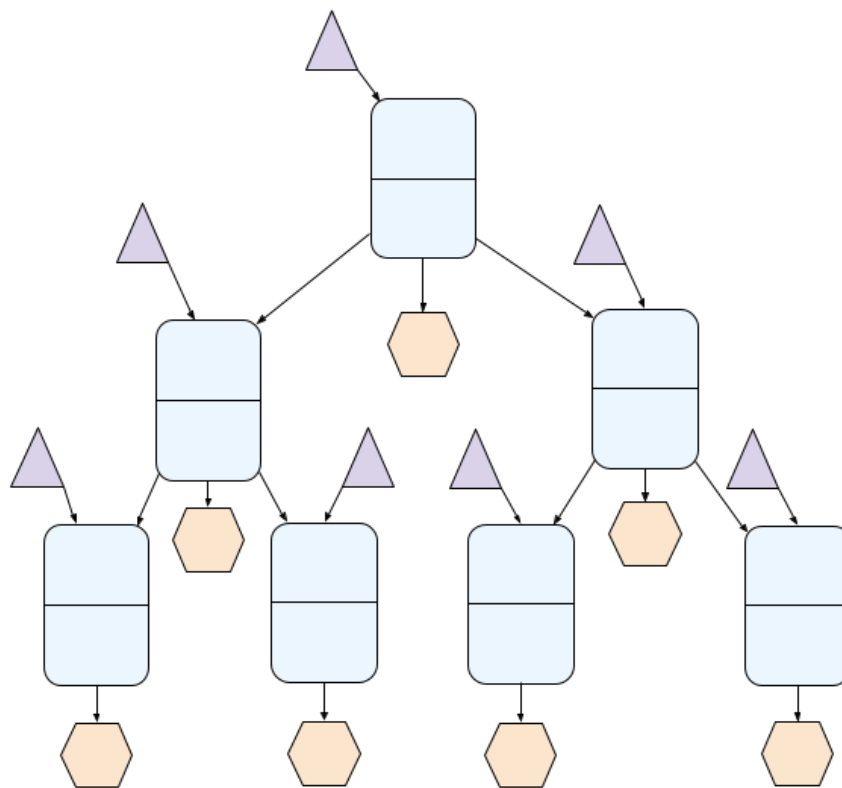
ADIN Agents, SCM's and Processing Models

<p>1. Sensor triggers inputs into ADIN Agent (upper section of Agent) and action response controls Motor (lower section of Agent) with SCM as an IoT bridge device.</p>	<p>ADIN Agent</p> 
<p>2. Agents can read sensor input from any number of sensors for triggering and can control any number of controls for action responses.</p>	
<p>3. Agents can combine input from sensors together with data, time and GPS location triggering information, and can notify, alert, update data sources and control outputs.</p>	
<p>4. Agents can share both sensor inputs and control outputs.</p>	

Agent-Based Processing in Automated Manufacturing



5. Agents can be organized to create sequences to form state control diagrams.



6. Agents can be organized to create decision trees, along with any type of topology that solves the problem.

Conclusion

In this paper, we have presented our thoughts on how ADIN Agents using the Sensor Control Module can be used to develop new industrial automation systems, as well as replace and improve upon current manufacturing systems that use Ladder Logic and PLCs. The benefit is that Agent-based processing has been shown to be stable and adaptable over the lifetime of complex systems. Running each Agent in its own processing container provides a stable environment. Each process runs autonomously and independently of each other, allowing for greater range of timing controls. ADIN Agents are widely configurable in that both triggering criteria and action responses are set up once and depending on the application can update parameters over time making them adaptable to the data as well as global processing requirements.