AMS Group 3 Report

Daniyar Irishev – 1747175
Joshua Tampoe-Holmes – 20105219
Shuailin Tan – 20109071
Diego Toledano – 20113058
Jordan Truong – 1638008

December 2020

Contents

Chapter1 Purpose of The System

Warehouse processes have continuously been a strenuous task. It requires high labour costs to meet the capacity necessary for a functioning warehouse, and is a challenging task of human resource management to ensure the fluency of operations and being able to consistently maintain time and planning metrics to guarantee business performances are met to great standards.

The aim of this project is to theoretically describe an alternative approach with a system that is fully autonomous and is operated by intelligent agents that has the ability to independently think for itself and collaborate with other agents in a rational and methodical structure. The project will detail the architectural design and properties of such an agent that enables it to immerse itself into a closed-world environment to complete the task of collecting and distributing packages from external suppliers to allocated delivery chutes. Properties will begin with describing the capabilities of agents possessing human-like attributes to then progressing into how they are able to cooperate and communicate with one another and furthermore, advancing into more complex situations where they are able to make decisions from group consensus or strategically negotiate and auction between themselves to acquire the best outcome for their self-interest.

Chapter2 High Level Design

The warehouse at Figure 2.1(a) is divided into the storage area and the delivery area. The storage area is divided into 7 regions based on the origin of the package where the delivery agents complete the pickup. The delivery area has chutes as the package destinations that are determined by the final destination of the package. This area also contains the charging station.

The delivery system consists of 3 types of agents:

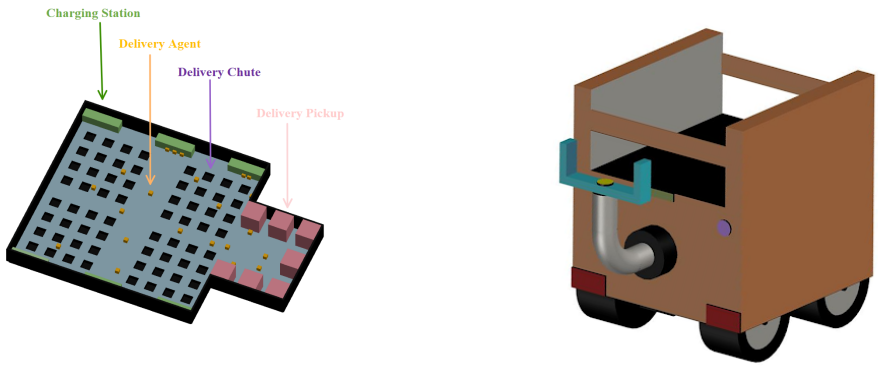
1. Centralised agent
2. Delivery agent (Figure 2.1(b))
3. Human emergency agents

The role of the centralised agent is to distribute delivery tasks to the delivery agents. Every task consists of the pickup region and the package ID which is assigned based on the arrival time of the package to the warehouse where "1" is the earliest. These ID numbers are reset every 24 hours and the lowest ID number changes to "1" with every other ID shifting accordingly.

Each delivery agent has a queue of packages that are assigned to it and it must complete these deliveries in the order of the queue. The agent knows where the package is in the storage and can scan the package to determine its destination based on the postcode. There are a total of 100 delivery agents and each one can hold 4 packages at a time.

The human agents only step in during an emergency event which the delivery agents cannot deal with on their own. These events include:

- Agent falling over and/or is unable to continue operating in which case it is taken to repair
- Any catastrophic event where the delivery agents are unable to carry out the deliveries. In this case, the human agents attempt to scan the packages that were being delivered and put them back into storage.



(a) Warehouse (b) Delivery agent

Figure 2.1: High level design

Chapter3 AMS Concepts

This chapter contains sections describing the use of AMS concepts in this project.

3.1 Beliefs, Desires and Intentions

Beliefs, Desires and Intentions, are three of the most important aspects of an agent's architecture. Those three aspects allow an agent to perform some tasks. Beliefs are the set of knowledge an agent knows about the world. Based on those beliefs, an agent has some Desires which can be translated as the goals it wants to achieve. Finally, an agent's Intentions are the plans he is using to achieve his desires. In addition to those aspects, we can add Percepts that are how the agent sees his environment.

Since the centralised agent gives orders - and thus desires- to the delivery agent, it later believes that the centralised agent has already checked the environment for him and that the environment is supposed to be and behave as the centralised agent describes it. As an example, the centralised agent could order a delivery agent to pick up packages at the storage zone and to deliver it to the corresponding chute. The delivery agent will then have the desire to do this given task and believes that in the storage zone, at the correct location, it will find the corresponding package. His Intentions are then, to pick up the package and to deliver it in the correct chute as fast as possible and without facing any trouble. Arriving at the given location, the agent will use his percepts - mainly cameras and sensor - to identify his package and where it needs to be delivered. Assuming he has found the correct package, he will then go to the corresponding chute - which is also identified using percepts - and accomplish his desire. This example is represented as follows :

$1.B := B_0;$
$2.D := D_0;$
$3.\mathit{while}\,\mathit{true}\,\mathit{do}$
$4.\ \mathit{get}\,\mathit{next}\,\mathit{percept}\,p;$
$5.\ B := brf(B, p);$
$6.\ I := filter(B, D, I)$
$7.\ \Pi := plan(B, I);$
$8.\ execute(\pi);$

As we can see above, the agent's desire is given at the beginning, this is because those desires are given by the centralised agent.

Beliefs, Desires, and Intentions are very important because they lead to the set of actions, agents should perform to achieve given goals. But what if two agents have either the same goals or actions in their plan that will lead to conflict? To solve that and keep our desires, we will use Contract Net Protocol so agents can communicate between each other.

3.2 Agent Architecture

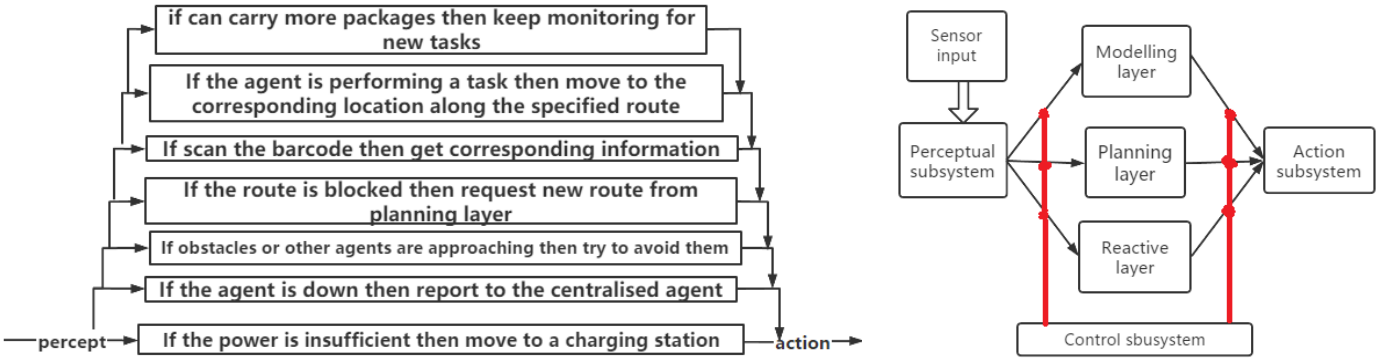
Since the agent may encounter some unexpected situations when working, it is a challenge to consider all situations and symbolize each situation reasonably. Moreover, in some cases, the agent may not have enough time to reason, so it is required to make a timely response. So it is better to use subsumption architecture as opposed to temporal logic. The behaviour modules of subsumption architecture are organised in Figure 3.1(a):

There are two types of barcode in the third module. First, agents can locate themselves in the warehouse by scanning barcodes embedded on the floor. On the other hand, agents can scan the barcodes on the packages to get the information about tasks. The behavior modules about state are lower than the modules about tasks because the state of the agent is more important than the tasks. Reactive agents alone are not sufficient in completing the task in the warehouse, because agents also need to be able to calculate the optimal path and interact with other agents, so it is better to use hybrid agents which contain the subsumption architecture as a reactive layer. The structure of the hybrid agent is shown in Figure 3.1(b).

In deciding a system to prioritise and build a thorough process in making immediate and long-term decisions, an agent with horizontal layering would be the most appropriate. With the main priority of collecting and depositing packages, the agent will also need to consider its surroundings to avoid colliding with obstacles and collaboratively working with other agents to ensure a coherent and systematic workflow.

The second component of the horizontal layering would be the planning layer. This calculates the optimal route in collecting the delegated packages by not only factoring which packages are nearest, but also to minimise battery consumption and time spent in completing tasks.

The final layer of the system would be the modelling layer, this will allow the agent to gain the ability to communicate and cooperate with other agents. All agents are delegated tasks by the centralised agent, but if an agent believes it would be beneficial to redistribute a task to an agent, the modelling layer will commit the process where details are included in the next section. The control subsystem is the head of the operations of the agent, it decides which layer has the greatest importance then prioritizes its actions given the situation.



(a) Behavioural modules

(b) Architecture structure

Figure 3.1: Agent Architecture

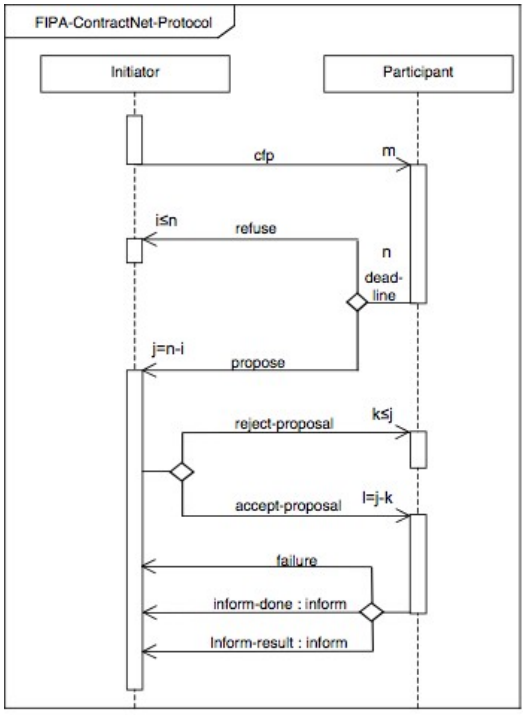


Figure 3.2: CNP Specification by FIPA

3.3 Contract Net Protocol

Source: FIPA (2002)

Multi-Agent Planning (MAP) aims to coordinate the activities of agents within a system by joint construction of plans, whereby the intention is to avoid negative relationships between actions, promote and benefit from positive relations, thus achieving harmonious coordination between agents in the system(Qy) (M., 2009). A Centralised planning for distributed agents' system is the most suitable MAP approach, wherein a centralised coordinating agent has access to all the private goals and actions available to the distributed agents, allowing it to develop a global plan. The agents within this system exist in a cooperative, non-competitive environment, therefore, sharing the private goals of each agent is not detrimental to the effectiveness of the system nor the individual goals of the agents.

In this system, the Contract Net Protocol (CNP) is used by the delivery agents to re-allocate tasks to each other if needed. The agents follow the specification established by FIPA (Figure 3.2). This allows for the delivery agents to have a degree of automation on a smaller scale which saves time and resources for the centralised agent.

Use case: Ag1 has one package assigned to its queue but it's far from the storage. Ag1 acts as the Initiator and broadcasts a Call For Proposals (CFP) with the ID and the location of the package that needs to be delivered. Ag2 and Ag3 respond with bids that contain information such as the number of packages in their queue and their distance from the storage. Ag1 chooses Ag3's bid as it has fewer packages to deliver than Ag2 and is closer to the storage. Ag1 informs the centralised agent that it has re-assigned the delivery of its package to Ag3 and the centralised agent updates its information accordingly.

Autonomous agents acting in open, risky and competitive multi-agent environments use trust and reputation between entities already existing in the environment, and new entities entering the environment to make appropriate decisions. The agents operating within the confines of this paper inhabit a closed environment, where cooperation amongst agents is an intrinsic feature set during the design process and, therefore, trust is an implicit characteristic of the system (Vázquez-Salceda).

3.4 Negotiation

Negotiation is the process whereby all participants aim to reach an agreement on matters of common interest, even in situations wherein a participant may have conflicting goals and preferences (M., 2009). The Task Oriented Domain represents a simple, low-level domain, where negotiations between agents aims to redistribute an initial set of tasks in such a way that is mutually beneficial to the negotiating parties. It is predominantly well suited for autonomous entities interacting and negotiating in a cooperative, task sharing environment (G. and Rosenschein, 1992). Its Inherent cooperative attributes make it the most logical selection for this design. The Alternating offer protocol in contrast is aimed at the division of resources wherein a singular agent may benefit at the expense of another, which beares no relevance to this application.

- A Task Oriented Domain Consists of a tuple; $< T, Ag, c >$
- T is the set of all tasks delegated by the centralised agent.
- $Ag = Ag_1, Ag_2, \ldots Ag_n$ is the set of participating agents capable of making deliveries, therefore, only delivery agents are capable of negotiating and redistributing tasks.
- c is a monotonic cost function that maps any set of tasks to a real number (Rosenschein and Zlotkin, 1994). The cost of carrying out a delivery within this domain is the distance travelled by an agent; a trait derived from the Postmen Domain (Rosenschein and Zlotkin, 1994)

The primary goal of a delivery agent is to pick up a package and drop said package at the designated chute. Ideally the distance travelled by the entire system is reduced by reducing the number of unnecessary trips made by a singular delivery agent, therefore increasing the overall efficiency of the system.

- If an agent is required to drop off a package at two destinations that are at relatively large distances from one and other, said agent can choose to redistribute one of the deliveries to another agent that may be passing that respective chute with no additional cost to either agent.
- If several agents are required to travel to a common region, and the same task can be achieved by a singular agent, negotiations can explore the possibility of achieving the goals at a lower cost than the initial encounter.

The possible subdivisions of the task vary in benefits; some deals leave a singular agent better off but no agent worse off, the overall impact is a more efficient system. Agents that are unable to reach an agreement on a set of tasks simply resume the initial tasks allocated by the centralised agent.

Ideally the negotiation set only contains tasks that are both pareto optimal and individual rational. In reality, however, there are in fact hidden costs associated with redistributing tasks, costs such as energy and time exerted during negotiation, as well as energy used to break and stop near the new chute. These costs, however, are assumed to be negligibly small relative to the cost to the system.

As we have just seen, negotiation helps us to deal with one-to-one issues like saving time by carrying someone else's package, however, this aspect has some limitations. One of them is to deal with more than two agents to make a decision. In order to do so, agents will have to use voting instead of negotiation to resolve their issues.

3.5 Voting and Preferences

Since agents are autonomous, they need to be able to take decisions in order to follow their plan and achieve their goals. While some decisions can be taken easily such as following a path, drop off or pick up something, other decisions are harder to be taken since they involve another agent into the decision process. In that case, agents need to take a group decision. To do so, agents use the plurality protocol by submitting their preferences on how to take this decision, voting on their preferred preferences and figuring out the winner. The winner is then the decision our agents will take.

As said before, in our design, robots agents needs to pick up items from point A and deliver them to point B while the master agent is the agent monitoring all of this. On the one hand, since the master agent does not get in contact with another agent (apart the robots agents when it gives them some tasks), the master agent does not need to take group decisions. Thus, it does not need to vote in order to take decisions. On the other hand, our robot agents all live in the same environment thus they might get in contact with each other and have to take group decisions. As an example, since our agents are running on batteries, they need to charge them self. Let us say that three robots needs to charge them self but there is only one charging spot available. In that case, robots agents need to vote on which agent need to charge first based on efficiency (tasks remaining and time to achieve them, batteries left, time to charge, etc...) Our agent will then use the plurality protocol to solve these issues. Our agent will then submit their preferences on who will charge first, they will then vote and figured out which decision they will take. This example is represented as follows:

$$
\begin{aligned}
&\Omega = \{\omega 1, \omega 2, \omega 3\} \\
&\omega 1 \ = \ agent\,1\,should\,charge. \\
&\omega 2 \ = \ agent\,2\,should\,charge. \\
&\omega 3 \ = \ agent\,3\,should\,charge. \\
&1\,agent\,has\,the\,preference\,\omega 1 \succ \omega 2 \succ \omega 3. \\
&2\,agents\,have\,the\,preference \omega 2 \succ \omega 3 \succ \omega 1.
\end{aligned}
$$

Hence, based on the plurality protocol,

$$
\begin{aligned}
&\omega 1\,gets\,1\,point. \\
&\omega 2\,gets\,2\,points. \\
&\omega 3\,gets\,0\,points.
\end{aligned}
$$

The winner under the plurality protocol is $\omega 2$. Consequently, agent 2 will charge.

3.6 Combinatorial Auctions

Auctioning plays a vital role in an agent's lifecycle; it enables the agent to distribute scare resources to those with the highest demands and benefit the winning agent in progressing with their allocated tasks or solely for their self-interest. An application of when auctioning would be implemented would be when distributing insufficient packages to multiple agents allocated to collect the same package. Insufficient packages may occur due to late delivery from suppliers, or internal complications in depositing packages onto shelves. This method would be incredibly effective as the agents will have different interests due to several reasons, some of which may include different delivery options such as a package set out for a one-day delivery rather than a standard delivery (3-5 business days) will require a higher demand in urgency, or the agent has only one remaining package to collect rather than a full set of tasks so a preference of finishing their set of tasks is favored along with the latter having the option to collect the other allocated packages as it waits for the specific package to be replenished.

The type of auction that will implemented will be a Combinatorial, Vickrey auction. When resources become sufficiently low and there exists more agents allocated to collect the package than the remaining packages available, it would be efficient in auctioning the remaining packages simultaneously rather than auctioning each individual package consecutively. This would optimize the auctioning process by reducing the time in distributing the remaining packages between the agents to ensures a fluent and systematic workflow. The one shot property of the Vickrey auction enables the process to make immediate decisions and further enhance on the fluency of the distribution. This is important aspect of the auctioning process as time is a metric that we wish to optimize as efficiently as possible to ensure fast workflow and for delivery standards to be met.

An example below displays how a Combinatorial, Vickrey auction is processed when auctioning for remaining items:

- z=z1,...,zm is the set of remaining packages to be auctioned.
- Ag=ag1,...,agn is the set of agents bidding for the remaining packages
- Every agent submits a private bid to the auctioneer where the dominant strategy would be to submit the agent's true valuation of the package
- A subset of Ag, Ag'=ag1',. . . ,agm', are the winning agents with the highest m bids are allocated to receive the remaining packages

Chapter4 Other Components

This section describes other components which are not relative with the concept of AMS. It includes the structure of the agent and charging station.

The structure of the agent is shown on Figure 4.1(a):

There is a storage rack in the center of the agent which is used to put packages. Also, there is a manipulator arm in front of the agent. It can take packages off shelves and drop them into designated chutes.

Furthermore, there are two scanners, the green one behind the arm is used to scan the barcode of the package and the blue one at the bottom is responsible for scanning the barcode on the floor to locate its position.

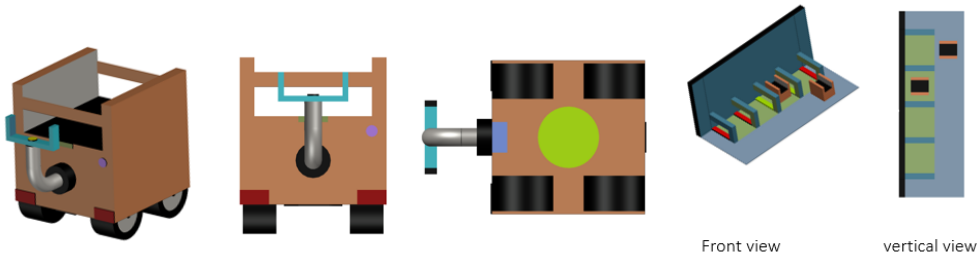
Then, the red items on either side of the agent are the sensors which are used to sense the distance from other agents. If it senses that another agent is approaching and is likely to cause a collision, it is prepared to avoid other agents.

The purple circle above the sensor is the signal component which is used to send information to communicate with other agents or to convey information to the centralised agent. Finally, the green circle at the bottom of the agent is the wireless charging port. When the power is low, it is used for charging.

Except for the component on the outside, there is also a central controller which manages the behavior of the agents and process information.

On the other hand, the structure of the charging station is shown on Figure 4.1(b).

There are a total of 20 charging stations and each for one agent to charge. If there are no free stations when another agent needs to charge, the delivery agents need to vote which agent stops charging to make space or decide on which agent is next in line to charge.



(a) Delivery agent (b) Charging station

Figure 4.1: Other components

Chapter5 Reflection

Overall, our design does include the AMS concepts and could theoretically work in practical use. Despite having a negotiation protocol to increase the overall efficiency, an omnipotent centralised agent who is leading our delivery agents, a tiles-based floor plan so it's easier to know where things are in the warehouse and being inspired by the Amazon warehouses, it does also have some limitations like not having enough charging spots or not having a proper sorting system for some packages, not taking into account some customers behaviours, having delivery robots that can only carry up to four items at the time, robots that needs to charge themselves. Hence, in order to be practical and work on a real environment, we would need to add some solutions. The most obvious one would be to include some human agents into the system to solve some issues that robots can't do on their own, to respond to catastrophic events like power shut down and to be able to monitor all those agents.

Bibliography

FIPA. "*FIPA Contract Net Interaction Protocol Specification*", 2002. URL `http://www.fipa.org/specs/fipa00029/SC00029H.html`.

Zlotkin G. and J.S. Rosenschein. "*A domain theory for task oriented negotiation*", 1992.

Wooldridge M. An introduction to multiagent systems, 2009.

Y. Qy. Approaches to multi-agent planning.

J.S. Rosenschein and G. Zlotkin. Designing conventions for automated negotiation, 1994.

Javier Vázquez-Salceda. "*Coordination and Social Models*". URL `https://www.cs.upc.edu/ jvazquez/teaching/sma-up`