



An Intelligent Agent Model and a Simulation for a Given Task in a Specific Environment

By

Safa'a Yousef Abu Hadba

Supervisor

Dr. Hussein H. Owaied

A Thesis Submitted in Partial Fulfillment
of the Requirements for the Master Degree in
Computer Science

Faculty of Information Technology
Middle East University

June, 2011

AUTHORIZATION FORM

إقرار تفويض

أنا صفاء يوسف ابوهديه أفوض جامعة الشرق الأوسط بتزويد نسخ من رسالتي
للمكتبات أو المؤسسات أو الهيئات أو الأفراد عند طلبها.

التوقيع: 

التاريخ: ٢٠١١/٦/٨

Authorization statement

I Safa'a Yousef Abu Hadba, Authorize the Middle East University to supply a copy of my Thesis to libraries, establishments or individuals upon their request.

Signature:






Date:

8/6/2011

Middle East University

Examination Committee Decision

This is to certify that the thesis entitled "An Intelligent Agent Model and a Simulation for a Given Task in a Specific Environment" was successfully defended and approved on June - 8 -2011.

Examination Committee Members	Signature
Prof. Mohammad M. Al-Haj Hassan Professor, Department of Computer Science (Middle East University)	
Dr. Hussein H. Owaied Associate Professor, Department of Computer Science (Middle East University)	
Prof. Musbah J. Aqel Professor, Department of Computer Engineering (Applied Science Private University)	

ACKNOWLEDGMENTS

I would like to thank my supervisor Dr. Hussein H. Owaied for his support, encouragement, proofreading of thesis drafts, and helping me throughout my thesis, and so putting me on the right track of Artificial Intelligence and intelligent agents' field. I thank the Information Technology Faculty members at the Middle East University for Graduate Studies. I thank my father for his continues support during my study. I would also like to thank my brother ala'a Abu hadba, and my friend Zainab Namh Abdulla and Umaiya Murad for their support during writing my thesis.

DEDICATION

The Almighty Allah says “And remember! your Lord caused to be declared (publicly): "If ye are grateful, I will add more (favours) unto you; But if ye show ingratitude, truly My punishment is terrible indeed.”
Mention the surah.

So all praise is for Allah, the exalted, for his favors that cannot be counted.

I dedicate this work to my Parents, my brother, my sisters, my relatives, my friends, and to all those who helped, supported and taught me.

Table of Contents

An Intelligent Agent Model and a Simulation for a Given Task in a Specific Environment.....	I
Authorization statement.....	II
Examination Committee Decision.....	IV
ACKNOWLEDGMENTS	V
DEDICATION	VI
Table of Contents.....	VII
List of Tables.....	XII
List of Figures.....	XIII
List of Abbreviations.....	XVI
Abstract.....	XVII
المخلص.....	XVIII
Chapter One: Introduction.....	1
1.1 Overview.....	2
1.2 Problem Statement	4
1.3 Objectives of the proposed project	5
1.4 Research Importance.....	5
1.5 Thesis Structure.....	6
Chapter Two: Terminology	8
2.1 Overview.....	9

2.2 Types of Environment.....	9
2.2.1 Static Environment	9
2.2.2 Dynamic Environment	10
2.2.3 Discrete Environment	10
2.2.4 Continuous Environment	10
2.2.5 Comparison between the characteristics of two environments	11
A. Discrete Environment with Single Intelligent Agent.....	11
B. Continues Environment with Multi-agents.....	12
2.3 Accessing Task in the Environment	13
2.3.1 Fully Observable Access.....	13
2.3.2 Partially Observable Access.....	13
2.3.3 Deterministic Access.....	13
2.3.4 Stochastic Access.....	14
2.3.5 Episodic Access.....	14
2.3.6 Sequential Access	14
2.4 Properties and Organization of Object in Environments.....	15
2.4.1 Structured	15
2.4.2 Semi-Structured.....	15
2.4.3 Unstructured.....	15
2.5 Concept of intelligent agent	16
2.5.1 Types of intelligent Agent	16

2.5.1.1 Single-intelligent Agent	16
2.5.1.2 Multi-intelligent Agent	16
2.6 Concepts of the Object-Orientation.....	17
2.6.1 Objects.....	18
2.6.2 Classes.....	18
2.6.3 Inheritance.....	18
Chapter Three: Literature Survey and Related Works.....	20
3.1 Overview.....	21
3.2 Literature Survey Related to Intelligent Agents	21
3.3 Literature Survey Related to Planning Problem	22
3.4 Literature Survey Related to Task Specification Representation.....	25
3.5 Literature Survey Related to Knowledge.....	27
3.6 Literature Survey Related to Machine Learning.....	29
Chapter Four: Design of the proposed model and knowledge base representation.....	31
4.1 Overview.....	32
4.2 Proposed Model of Intelligent Agent AS Knowledge-Based System.....	34
4.2.1 User Interface Model.....	36
4.2.2 Inference Engine.....	36
4.2.3 Working memory.....	37
4.2.4 Knowledge Base.....	38
4.2.4.1 Static Knowledge	38

4.2.4.2 Dynamic Knowledge	39
4.3 Knowledge Representation methodology	39
4.3.1 Frames.....	39
4.3.2 Semantic networks.....	40
4.3.3 Rule Base.....	42
4.4 Hierarchical Structure of Environment.....	43
Chapter Five: The implementation of proposed model.....	46
5.1 Overview.....	46
5.2 Control Data Flow Diagram for the task.....	46
5.3 Presents the interactions for event processing and agent behavior.....	48
5.4 present the first screen user interface to program.....	51
5.5 Describe the action which can be given to agent in program.....	51
5.6 Representing Knowledge in the Knowledge base.....	53
5.7 Hierarchical Structure of Proposed Model (Library).....	56
5.8 Environment.....	57
5.9 State Space and Action Space in Environment	60
5.10 The Intelligent Agent Implementation in Library Environment.....	62
5.11 The Algorithms and Functions	64
5.11.1 Function Check for Object Availability.....	64
5.11.2 Procedure fillIndexQueu.	65
5.11.3 Function for the Heuristic search Technique	65

5.11.4 Function Insertion Sort	67
5.11.5 Function Start Point Finding.....	68
5.11.6 Function End Point Finding.....	69
5.11.7 Procedure Main Path Finding.....	69
5.11.8 Function getNextPoint	71
5.11.9 Function Test Main Path of the Obstacle.....	72
Chapter Six: Conclusion, Discussion & Future Work	78
6.1 Overview.....	79
6.2 Conclusion	79
6.3 Discussion.....	80
6.4 Future Work.....	81
References.....	81

List of Tables

Table 5-14 Represents the two diminutions array which contains all weight.....67

Table 5-25: illustrates a Comparison between three techniques shortest path and best time.....77

List of Figures

Figure 2-1: Example of Crossword puzzles is static task environment.....	10
Figure 2-2: presents Gridworld Game Ian, Michael & Peter, (2009).....	11
Figure 2-3: presents RoboCup game Ian, Michael & Peter, (2009).....	12
Figure 2-4: Example of Chess game is Sequential (non-episodic) task Environment.....	14
Figure 3-1: presents three levels of the spatial knowledge	28
Figure 4-1: Functional Model of Human System Owaied (2007).....	32
Figure 4-2: General Structure of Knowledge -based System Owaied (2007).....	33
Figure 4-3: Proposed model of intelligent agent as Knowledge-based System.....	35
Figure 4-4: Frame Structure.....	40
Figure 4-5: Semantic networks showing some object concepts and properties.....	41
Figure 4-6: production system using working memory.....	42
Figure 4-7: Hierarchical Structure of Environment.....	44
Figure 5-1: The Control Data Flow Diagram for the task when order Bring Book ID.....	47
Figure 5-2: shows the start-up sequence. And presents the interactions for event processing and agent behavior.....	49
Figure 5-3: represent first screen user interface to program.....	51

Figure 5-4: represents sub user interface to program proposed model describes the environment library.....	52
Figure 5-5: Explains measurement inside of the library proposed model.....	53
Figure 5-6: Inheritance Frame Structure.....	55
Figure 5-7: Hierarchical Structure of Proposed Model (Library).....	56
Figure 5-8: Environment Proposed model.....	59
Figure 5-9: Transition in Environment (State, Event).....	61
Figure 5-10: Main Algorithms as Processes in a knowledge Based System.....	63
Figure 5-11: procedure Fill Index Queue.....	65
Figure 5-12: Function for the Heuristic search Technique.....	66
Figure 5-13: Heuristic Function.....	66
Figure 5-15: The pseudo code for sorting insertion.....	68
Figure 5-16: Start Point Finding Algorithm Pseudo code.....	68
Figure 5-17: End Point Finding Algorithm Pseudo code.....	69
Figure 5-18: Main Path Finding Algorithm Pseudo code.....	70
Figure 5-19: Function getNextPoint.....	71
Figure 5-20: Mathematical equation..1.....	73

Figure 5-21: Mathematical equation..2.....	73
Figure 5-22: Mathematical equation..3.....	74
Figure 5-23: Mathematical equation..4.....	75
Figure 5-24: Function DosItCros.....	76

List of Abbreviations

AI	Artificial Intelligence
IM	Intelligent Machine
POP	Partial Order Planning
HTN	Hierarchical Task Network
BLTL	Bootstrap Learning Task Learning
RL	Reinforcement Learning
RAP	Rich Ajax Platform

ABSTRACT

This thesis presents An Intelligent Agent Model for a Given Task in a Specified Environment. The methodology in this work is based on mixing algorithmic and function approaches to construct the intelligent agent model. The thesis concentrates on building an intelligent agent model as a knowledge-based system interacting with dynamic environment to perform tasks.

The class structure used to represent the environment in the knowledge base depends on three types of knowledge representation forms: production rule, semantic net, and frames. Each object in the environment is an instance of the class environment. The Visual Basic.NET is used in the implementation, as an Object-Oriented programming language, because it offers a natural way of representing the real world in a computer.

Algorithms and functions are used for gaining knowledge from the state space of environment so as to build the task. The intelligent agent model can understand the environment from any position and can detect many subtasks, arrange them in a queue for execution, and has the ability to make a decision at a high level of thinking.

The intelligent agent model is able to calculate the persistent changes in the external dynamic environment and any sudden change, such as observing the existence of any obstacle in the environment and avoiding it. The intelligent agent is also able to learn and take the reasonable decision in the dynamic environment and automatically select action based on task characteristics. Therefore, the intelligent agent can solve many different types of problems.

الخلاصة

تقدم هذه الدراسة نموذجاً لوكيل ذكي لأداء مهمة معطاة في بيئة محددة وفق منهجية تقوم على خلط خوارزميات ودوال لبناء نموذج وكيل ذكي. وقد تم تكريس هذه الرسالة لبناء نموذج وكيل ذكي قائم على المعرفة وعلى التفاعل بين البيئة الديناميكية و المهام الموكله اليها.

وقد تم تمثيل البيئة باستخدام التركيب البياني المسمى Class وذلك بالاعتماد على ثلاثة نماذج من هياكل تمثيل المعرفة: قواعد الانتاج، شبكة الدلالات و الإطارات إذ إن كل كائن في البيئة هو ممثل لفئة. وتم التنفيذ باستخدام لغة فيجوال بيسك دوت نت، وهي من لغات البرمجة الشيئية لما توفره من وسيلة طبيعية لتمثيل العالم الحقيقي في جهاز الكمبيوتر.

تستخدم الخوارزميات والدوال لاكتساب المعرفة من فضاء البيئة الموجود بها لبناء المهمة المكلف بها وهذا يجعل لدى الوكيل الذكي القدرة لكي يكون مؤهلاً على فهم البيئة وإتمام المهام المكلف بها في أي موقع يتواجد به و لجعل الوكيل قادراً على تحديد المهام الفرعية، و ترتيبها في قائمة الانتظار للتنفيذ للوصول للمهمة الأساسية، والقدرة على صنع قرار على مستوى عال من التفكير.

إن نموذج الوكيل الذكي قادر على حساب التغيرات المستمرة في البيئة الديناميكية الخارجية وقادر على التعامل مع أي تغيير مفاجئ، مثل ملاحظة وجود أي عائق أمامه بشكل مفاجئ والقدرة على تقاديه. قدرة الوكيل الذكي على التعلم واتخاذ القرار السليم في البيئة الديناميكية واختيار المهمة بناء على خصائص المهمة، مقدرة الوكيل على حل العديد من الأنواع المختلفة من المشاكل.

Chapter One

INTRODUCTION

Chapter One

Introduction

1.1. Overview

Artificial Intelligence (AI) may be defined as the branch of computer science that is concerned with the automation of intelligent behavior defined by (Russell and Norvig ,2003). The concepts of Artificial Intelligence involve considering how to build an Intelligent Machine (IM). The processes of building such a machine are related to many sciences, so we can define the AI as interdisciplinary of sciences. This is the primary reason that AI can have many definitions in the literature.

Therefore, any given definition is a mirror of the background of that scientist who defines the term AI. To introduce the term to the reader, clear and recent definition was stated in (Owaied and Abu-A'ra,2007) “Artificial Intelligence is a concept of study and research for finding relationships between cognitive science and computation theories in order to represent these relationships as either data structures, search techniques, problem solving methods or representation forms for knowledge and the final goal of AI is to build an intelligent machine, with another benefit which is better understanding of human thinking”. In order to design and implement a program that mimics a human for doing a specific job, first of all we must have some way of determining how individuals do that job. Usually doing a job depends on the

knowledge of doing that job and this knowledge is stored in the long term memory of a human.

Usually, the knowledge is accumulated during a long period of time in the human brain.

This knowledge is used by individuals to solve problems or communicate with others. The need for applying the human brain model was increased by the demand for building intelligent machines.

Bekey (2005) defines automation as: “the automation refers to systems capable of operating in real world environments without any form of external control for extended periods of time”. The Intelligent agent defined as an agent that receives precepts from the environment and performs actions. The intelligent agent, usually implemented as a model presents the transformation of functions that map precepts sequences to actions. There are many different ways to represent these functions to describe the importance of the task environment in determining the appropriate agent design (Russell and Norvig ,2003).

Rob callan (2003), defined the intelligent software system as a collection of agents is a way of conceptualizing a complex system. Tools, techniques and standards are being developed for this form of software engineering. Such systems contain an agent that has goals and is situated in some environment, and the agent can sense its environment and is capable of independent action. Since Knowledge is the human brain’s soul, solving problems and

communicating with others, require applying the human brain model on the intelligent agent increased by the demand of building an intelligent agent.

The knowledge based system involves the basic concepts of thinking. This includes receiving, storing, and processing Data, Information, and knowledge.

Paterno, Mancini & Meniconi, (1997) defines task how an individual can reach a goal in a specific application domain. The goal is a desired modification of the state of a system or the environment of the task. Therefore, the proposed model in this thesis will concentrate on how this knowledge of the task is represented.

1.2. Problem Statement

There are many problems related to task specification representation. The task specification is the list of possible actions for doing certain job in a certain domain, according to preconditions provided. In this context there are two aspects that should be considered in order to represent the task specification .These are the task environments space and the task action space. So according to these two aspects, in order to specify and perform particular tasks, the following problems have been identified:

1. How to describe the state space and action space in the environment.
2. Identify and compile the necessary information, and identify and design functions/algorithms (or utilize existing ones) needed to train the agent to perform the required tasks.

1.3. Objectives of the proposed project

The main objectives of this research are:

- 1) Designing a model of intelligent agent that can act as a professional person to perform a certain task . The model acts as an agent that intelligently searches, fetches, locates, and brings the required object or accomplishes a certain task.
- 2) Generalizing the methodology of describing the task environment space and the task action space.
- 3) Applying the design model in a specific environment .

1.4. Research Importance

The new enhancement in the design of an agent to become intelligent agent can be applied in different environments so as to save money and effort and reduce the pressure of work. The intelligent agent can perform different tasks depending on the considered environment, especially in dealing with the spatial data to help the security and the emergency forces to overcome any obstacle in the environment.

Searching for an object in any environment for untrained agent can be very difficult and time consuming task. However, the same process for trained professional may be a quite simple process.

The goal for developing the model of intelligent agent in any environment, especially in dealing with the spatial data, is to help workers in such environment to search and find an object or accomplish a task.

Searching in such environment for particular object requires much effort and usually faces many problems. There are many problem solving methods, usually used by human, applied in this context. So the implementation of most of these problem solving methods as a software model will be very helpful to people specially those who work in dangerous environments.

The aim of this project is utilizing the intelligent agent that understands an environment and can do any action independently. The intelligent agent can interact with environment and transaction between states without any help from the user or in other words non-supervised. Therefore, the intelligent agent will work according to just the given goal by calling the stored knowledge from the knowledge base and accomplish that job. Also the intelligent agent can interact with a dynamic environment and avoid any obstacle in such environment.

1.5. Thesis Structure

The thesis includes six chapters; the current chapter is the introduction. Chapter two gives an overview of the terminology used in the area of intelligent agent together with the methods used for related different research areas. These methods are used within the work to satisfy different objectives.

Chapter three is the literature survey for the thesis, showing the related work regarding intelligent agent, Planning Problem, Task Specification Representation and knowledge.

In chapter four, we present the design of the proposed model of an intelligence agent as knowledge-based system together with the representation of knowledge base, action space, and the environment.

In chapter five, we present the design and implementation of the proposed model of an intelligent agent as knowledge-based system together with an application in the library environment. Finally, chapter six contains the conclusion and the future work for the thesis.

Chapter Two

TERMINOLOGY

Chapter Two

Terminology

2.1 Overview

This chapter includes an explanation of the concepts used in the environment, and describes types of environment and the properties of task in the environment. Also, it describes the declaration of the methods and terms that are used in the model. Moreover, this chapter presents the concept of intelligent agents and their types. Brief descriptions of the environments and their types, properties, and organization of objects in the environments have been presented.

2.2 Types of Environment

When need To design an agent ,we must specify its task environment and any environment that has many types .The following section describes and illustrates the types of environment and then Compares the characteristics of two environments (Russell & Norvig ,2003).

2.2.1 Static Environment

Static environment is the environment that does not change and can be easily dealt with by an intelligent agent. The intelligent agent is not necessary to continue searches in the environment when performing a task or when making a decision for an action. For example, Solitaire and Backgammon and Crossword puzzles are static

environments Figure 2-1 illustrates an example of Crossword puzzles static environment.



Figure 2-1- : Example of Crossword puzzles is static task environment

2.2.2 Dynamic Environment

In the case of the changing environment, non-static, and is the environment is that can be changed during the intelligent agent processes a task in the environment is called a dynamic environment. Car moving on a road is an example of this type.

2.2.3 Discrete Environment

The discrete environment can be defined as the environment which consists of many finite numbers of distinct states, for example a chess game has a discrete set of actions.

2.2.4 Continuous Environment

The continuous environment can be defined as the environment which consists of many continuous finite numbers of states. The continuous states environment for example is car speed and location of the car with respect to other vehicles and passing cars through a range of continuous values.

2.2.5 Comparison between the characteristics of two environments

A. Discrete Environment with Single Intelligent Agent

The gridworld game in Figure 2-2 presents an example of discrete environment with a single intelligent agent. The game involves many possible tasks, for example, one task may be the intelligent agent starts from any location in sub-environment, the shaded area, to accomplish the movement for finding the specified object in the environment, in which the star is the symbol. The following are the characteristics of such environment and the properties of a task in such environment (Ian, Michael & Peter , 2009):

- Deterministic domain.
- Discrete states and action space.
- Contain single intelligent agent.
- Episode.

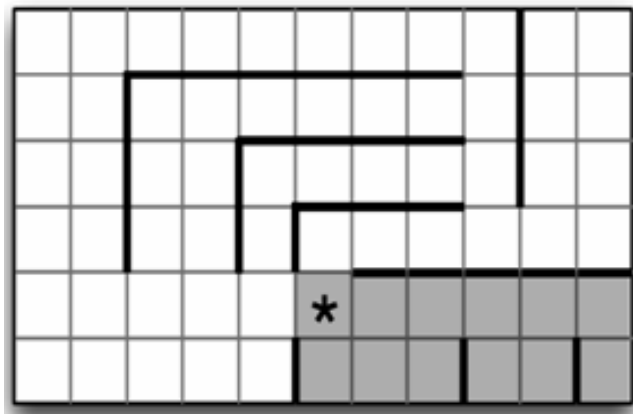


Figure 2-2 presents Gridworld Game Ian , Michael & Peter ,(2009)

B. Continuous Environment with Multi-agents

Figure 2-3 presents an example of continuous environment with Multi-agents, group-A and group-B. The intelligent agents, either group-A or group-B, work in a continuous complicated environment. The complexity of this environment is in the nature of the game which is a football game; in reality the two groups of intelligent agents working in two different environments even though it is just one environment. The following are Properties of a task in such an environment (Ian , Michael & Peter , 2009):

- Non Deterministic domain.
- Continuous state and discrete actions.
- Contain multi intelligent agent domain.
- Episode.
- Each intelligent agent has only a partial world.
- Complex and stochastic domain.

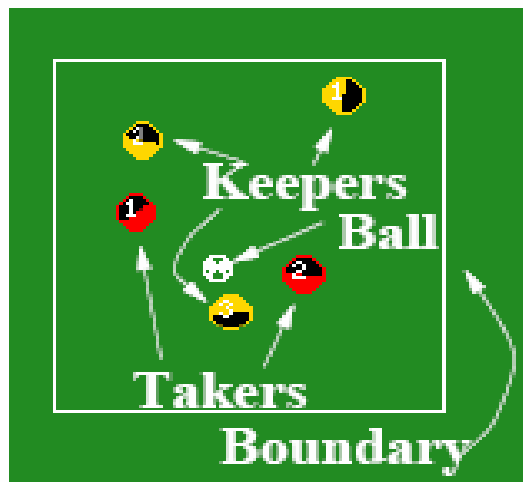


Figure 2- 3 presents RoboCup game Ian , Michael & Peter ,(2009)

2.3 Accessing Task in the Environment

There are many methods that can be used for accessing the environment. In the following subsection we categorize the environments according to their accessibility (Russell & Norvig ,2003).

2.3.1 Fully Observable Access

In the case of the Sensors that give the intelligent agent access to all state in the environment and the task environment is fully observable, and then the sensors can detect all aspects of special and appropriate choice of action. For example, Solitaire and Backgammon are of this type.

2.3.2 Partially Observable Access

In the case of the Sensors that do not give the intelligent agent access to all state in the environment and the task environment is partially observable, then the sensors can not detect all aspects of special and appropriate choice of action, For example, Internet shopping and a car moving on a road are of this type .

2.3.3 Deterministic Access

The environment is deterministic if it decided the next state by the current state of the environment and action executed by the intelligent agent, for example, Internet shopping and Solitaire are of this type.

2.1.1 Stochastic Access

In Case the environment is complex, this causes the difficulty of tracking all aspects of the hidden. Examples are driving car, where we can't predict the behavior of traffic exactly, Backgammon, Taxi.

2.1.2 Episodic Access

Each episodic task environment consists of a single action. The next episode does not depend on the actions taken in the previous episode, and the choice of work in each episode depends only on the episode itself.

2.1.3 Sequential Access

A sequential access of environments is that which can affect the current decision of intelligent agent on all decisions in the future, and should be on the agent decision making when thinking about the future. Figure 2-4 represents an example chess game.



Figure 2-4: Example of Chess game is Sequential (non-episodic) task Environment

2.2 Properties and Organization of Objects in Environments

The environment can contain different objects and these objects have many properties and can be organized in many methods. In the following subsection we describe different organizations.

2.2.1 Structured

The objects in environment organized sequentially, mean intelligent code linked the object in environment and the agent can through this code detect the position of target then directly go to target.

2.2.2 Semi-Structured

It combines between structured and unstructured. This means mean that the object in environment some objects structured and organized and have the intelligent code and other objects in environment unstructured and not have intelligent code.

2.2.3 Unstructured

The objects in environment are not organized sequentially .This means that they do not have intelligent code linked the object in environment and the agent can not through this code detect the position of target then directly goes to target . The objects at arranged stochastic and the agent detected the target of task in use heuristic search and experience.

2.3 Concept of intelligent agent

The Intelligent Agent should have the capabilities such as (Owaied ,2007) “behaves like a human being, smart, problem solver of unstructured and complex problems as human does, understands languages, learner, and able to reason and analyze data and information, and so on”.

2.3.1 Types of intelligent Agent

There are two types of the presence of intelligent agent in the environment. Determining the types of intelligent agent is based on the existing environment and the tasks that are given to the intelligent agent (Russell and Norvig ,2003).

2.3.1.1 Single- intelligent Agent

The Single agent in environment means that only one agent is in the environment. There is no other agents in the environment to help it do the task or perform other tasks in the environment, alone only with their actions affect the world.

2.3.1.2 Multi- intelligent Agent

In multi- intelligent agent environment means that there is more than one intelligent agent in the environment to carry out the tasks. Intelligent agents need to account for the actions of other intelligent agents. There are three types of tasks to be carried out by multi- intelligent agent environment.

- Task is **competitive**: If there is a competition between intelligent agents in environment when doing task in the working environment.
- Task is **cooperative**: Collaborative environment exists where intelligent agents work together as a single unit.
- A Task can be both **competitive** and **cooperative** to different degrees. Example of a cooperative and competitive environment would be for example, **RoboCup** multi intelligent agent domain with both teammates and adversaries.

2.4 Concepts of the Object-Orientation

Object-oriented programming offers a natural way of representing the real world in a computer. In the object-orientation, the model is the number of objects that interact with each other. The environment, for example, consists of objects, such as vehicles, people, trees, and building and stone that are in some way related to each other. Model, which has been designed using object-oriented technology, is often easy to understand, and can relate directly to reality.

Use the **instance-frame** when referring to a particular object, and the **class-frame** when referring to a group of similar objects.

A class-frame describes a group of objects with common attributes. Table, bookcase, book and floor are all class-frames. In AI, however, the abbreviation 'class' is often used instead of the term 'class-frame'. Each frame in a frame-based system 'knows' its class.

2.4.1 Objects

An object is an entity that is able to save state information, which provides a number of methods to do the behavior of this state.

Objects usually correspond to real-life entity objects. For example bookcase or vehicles, table, trees. Each is taken as an object, information and behavior is related to object. Examples of the information high and width, and each object, must identify a set of methods that can do behavior.

2.4.2 Classes

A class is a template or a template of creation of new objects. Each class includes a group of objects that shares the same characteristics. Objects that include a certain class .For example, bookcase1, bookcase2, and bookcase3 are all bookcases that have similar properties and information structure, thus, they all belong to the same class: bookcase.

In object-oriented systems, each object belongs to a class. An object that belongs to a certain class is called an instance of that class. In the above examples, bookcase1, bookcase2, bookcase3 are all instances of the class bookcase

2.4.3 Inheritance

When describing classes, several classes have common characteristics (behavior and attributes). For instance, when comparing the classes' bookcase and shelf, it is clear that they are very similar to each other .The Similarity can be shared between the classes by extracting them and placing them in a separate class MyObject. In MyObject,

everything that is common. In this way several classes can share common characteristics .Thus, common characteristics are collected in one specific class and all other classes these features are allowed to inherit this class itself.

Chapter Three

LITERATURE SURVEY AND RELATED WORKS

Chapter Three

Literature Survey and Related Works

3.1 Overview

In Recent years ,intelligent agent and how it interacts with environment and does the task specification representation, attracted many researchers in the field of intelligent agents and artificial intelligence; so the literature survey and related works have been divided into five sections which are; literature survey related to intelligent agents, literature survey related to planning problems, literature survey related to task specification representation, Literature Survey Related to Knowledge , and Literature Survey Related to Machine Learning.

3.2 Literature Survey Related to Intelligent Agents

The agents in an the environments are anything that can show some kind of understanding and cognitive in the environment and enter the information through the senses and sensors , or by storing the specifications of the environment and clarifying to the agent, and then the agent representation in the work environment through actuators, mechanical or engine .

Russell, Norvig (2003) defines "intelligent agent (IA) as an autonomous entity which observes and acts upon an environment and directs its activity

towards achieving goals. Intelligent agents may also learn or use knowledge to achieve their goals. They may be very simple or very complex."

Rowell, (2002) Defined the System State as follows: "The concept of the state of a dynamic system refers to a minimum set of variables, known as state variables, that fully describe the system and its response to any given set of inputs." Environment stores all the relevant details of the world.

Woolderidge & Dunne, (2002) claimed that they use the result established in the many theories to analyze the complexity of agent verification for three classes of task specifications : achievement and maintenance tasks , and tasks specified as arbitrary Boolean combinations of achievement and maintenance tasks .

3.3 Literature Survey Related to Planning Problem

Most of problem-solving techniques have been applied in the field of the autonomous agents and are based on theorem proving (Green, 1969) used in knowledge-based systems and the intelligent agents.

Shapiro ,(2003) Described action planning as: "the process of planning of what needed to be done, when it is needed to be done, by who is needed to be done, and what resources or inputs are needed to do it."

The description of the action plans consists of the following elements:

- Statement of what must be achieved (the outputs or result areas that come out of planning process).
- Spelling out of the steps that have to be followed to reach this objective.
- Clarification of who will be responsible for making sure that each step is successfully completed.
- Clarification of the inputs/resources that are needed.

Also, there are many publications on planning methods. One such planning approach is partial order planning (POP) algorithms that explore the space of plans without being committed to totally ordered sequence of actions .They work back from the goal adding actions to the plan to achieve each sub goal.

Penberthy & Weld (1992); Weld (1994), used partial order planning (POP) algorithm to describe World-State planner, and formally a planning algorithm that has three inputs ”description of the world in some formal language, description of the agent's goal (i.e., what behavior is desired), and description of the possible actions that can be performed. “.

Implement the pop algorithm to solve the problem in the blocks world .starts with the null plan for a planning problem and makes nondeterministic choices until all conjuncts of every action's precondition have been supported by causal links and all threatened links have been protected from possible interference, the ordering constraints.

Another approach is to use task nets which are organized hierarchically instead of planning with basic actions. This approach is called Hierarchical Task Network (HTN) planning (e.g. Erol et al., (1994, 1996)).

Firby, (1996) described the (RAP) Reactive Plan hierarchies used to control the agent while cleaning up a small office space. The problem is the requirement that some tasks monitoring states in the world extend continuously across otherwise modular subtask boundaries. It must be possible to spawn tasks at the point in an expansion when they become relevant and then let them run continually across succeeding independent steps. The solutions to augment the existing RAP language to support the creation and termination of independent

DesJardins, (2001) & Georgeff, (1986) discussed the HTN and suggest an application for planning algorithm practical reasoning techniques where the knowledge is encoded in forms of procedures (recipes). These procedures describe the real world in the form of or sequences of actions for achieving particular goals.

Ilgami et al. (2005) proposed a learning system for HTNs, where a domain expert solves task nets giving examples to the learner. The learner now generalizes based on the training examples from the human expert and can solve similar tasks in a better way. Described CaMeL++ algorithm for learning preconditions for HTN methods that enable the planner to start planning before the method preconditions are fully learned.

Empirical results show that, by doing so, the planner can start solving planning problems with a smaller number of training examples than is required to learn the preconditions completely.

3.4 Literature Survey Related to Task Specification Representation

The problem of task specification representation is how to describe an agent, what task to carry out on its behalf without telling the agent, and how to do the task. So this has been discussed in the subsection according to the available researches related to the proposal.

There are many researchers who have been concentrating on the field of task specification and how it can be represented.

Dayan & Hinton, (1993) illustrated the Feudal system using a simple maze task." how to create a Q-learning managerial hierarchy in which high level managers learn how to set tasks to their sub-managers who, in turn, learn how to satisfy them. Sub-managers need not to initially understand their manager's commands. They simply learn to maximize their reinforcement in the context of the current command and build a more comprehensive map."

Paterno, Mancini & Meniconi, (1997) defined the task by how an individual can reach a goal in a specific application domain. The goal is a desired modification of the state of a system or the environment of the task.

Many researchers built different task models using different operators. The first problem is the possible ambiguity of some expressions. To solve the ambiguity, we use the priority order among operators defined in the standard LOTOS (choice > parallel composition > disabling > enabling), and when building the task model must use ConcurTaskTrees: A Diagrammatic Notation for Specifying Task Models, we include three phases: a hierarchical logical decomposition of the tasks representation, identification of the temporal, and identification of the objects associated with each task. Uses of graphical constructs and operators derived from a formal specification. The result gives specification of a wide variety of dynamic task behaviors.

Ian, Michael & Peter, (2009) research is about an architecture and interface language for teaching sequential decision making tasks, reinforcement learning agents called Bootstrap Learning for Task Learning (BLTL) they introduced the BLTL language that allows tasks to be specified concretely in terms of starting states, reward functions, and termination conditions.

In addition, they provide advice and suggestion of new task sources for future action. The BLTL language forms the cornerstone for the larger Bootstrap Learning project, which integrates even more machine learning methods for teaching agents to solve many different types of problems, not just sequential decision making tasks. Therefore, we need to describe the tasks that want agents to carry out on our behalf.

3.5 Literature Survey Related to Knowledge.

In the 1970s, it was finally accepted that to make a machine solve an intellectual problem one had to know the solution .

The knowledge can be defined as a set of facts, events, procedures, and meta-knowledge. Usually in the human behaviour for doing a certain task, the individual must call the knowledge from his knowledge base which is related to that specific task, the stored knowledge for doing certain task

The terms procedural knowledge and path (sometime called route) knowledge have different concepts and usually misinterpreted. The knowledge path has been defined by (Niels, 2002) as “A set of procedures that can be used only if an external entity explicitly specifies the initial situation and a desired end situation”. It encompasses information about the sequence of actions required to follow a particular path. Knowledge path is characterized as the knowledge about the actions to be performed in the environment to successfully traverse paths between distant locations, especially between an origin and a destination.

Therefore the route knowledge can be recognized as the sequences of functions or procedures to be accomplished for finding a path. While procedural knowledge is the knowledge that describes an action to be accomplished as a rule in the form (if conditions then action) and usually called as Horn Clause.

Thorndyke & Goldin, (1983) described the knowledge types in the environment, and call it spatial knowledge, as seen in Figure 3-1, in terms of three levels of information: landmark, procedural, and survey knowledge, where each level builds on previous levels. Landmark knowledge covers the perceptual salient objects in the environment. Procedural knowledge (or route knowledge) which encompasses information about the sequence of actions required to follow a particular route. Survey is the knowledge which deals with environmental information.

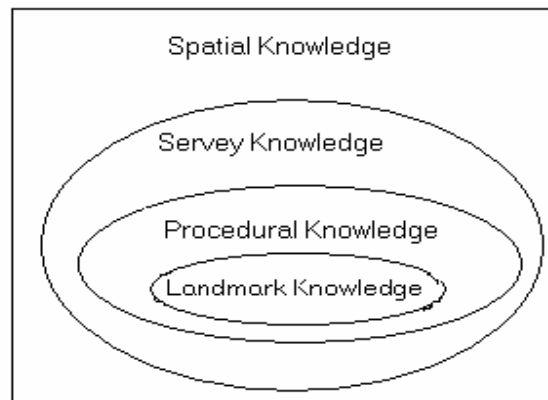


Figure 3-1: presents three levels of the spatial knowledge

3.6 Literature Survey Related to Machine Learning

Machine learning, a branch of artificial intelligence, is a scientific discipline concerned with the design and development of algorithms that allow computers to evolve behaviors based on empirical data, such as from sensor data or databases.

Indeed an agent can have a high degree of intelligence, autonomy, flexibility and adaptation that if it possesses capacities of knowledge learning

Feigenbaum and McCorduck, (1983) have called the “knowledge engineering bottleneck” “the major obstacle to the widespread use of intelligent systems. This “bottleneck” “is the cost and difficulty of building systems using the traditional knowledge acquisition techniques one solution to this problem would be for programs to begin with a minimal amount of knowledge and learn from examples , high-level advice , or their own explorations of the application domain .

Herbert Simon defines learning as:

Any change in a system that allows it to perform better the second time on repetition of the same task (Simon, 1983).

Learning involves generalization from experience: performance should improve not only on the “repetition of the same task” “but also on similar tasks in the domain.

Simon’s definition describes learning as allowing the system to “perform better the second time.” selecting the possible changes to a system that will

allow it to improve is a difficult task .Learning research must address the possibility that changes may actually degrade performance.

Balduccini .M and Lanzarone .G.A, (1997) introduce an agent who autonomously interacts with an unknown environment and builds an incremental and inductive model of it. The key concepts dealt with are:

- **Autonomy:** The agent can't rely on an oracle guiding its inferential processes.
- **Instrumentality:** The agent must be able to continuously update the environment's model according to the knowledge derived from interaction, exploiting as much as possible the information it has already acquired.
- **Inductivity:** The agent must infer rules on the basis of observed examples, and it is therefore necessary for it to constantly verify the validity of the rules it has induced.
- **Unknown environment:** The agent must perform well even when started in a 'zero knowledge' situation.

The aim is to investigate how the characteristics of the components of an agent are influenced by the fact that the agent has to interact with an unknown environment and build at the same time an incremental and inductive model of it by means of symbolic learning. What want to obtain is an agent featuring sufficiently low computational costs to be able to interact with the environment in real-time, but powerful enough to manage to reach the assigned goals in complex environments and in an acceptable time.

Chapter Four

DESIGN OF THE PROPOSED MODEL

AND

KNOWLEDGE REPRESENTATION

Chapter Four

Design of the Proposed Model and Knowledge Representation

4.1 Overview

This chapter presents the design of the proposed model; based on the functional model of human system as knowledge-based system. It includes the declaration of the knowledge representation and describes the knowledge representation used in proposed model. Also it includes the knowledge representation for the environment and state space and action together with description of the main algorithms as processes in a knowledge-based System. When these three parts are present the modules are complementary in their processes, which mean each module has a task that complements the other modules tasks.

Human does work as a functional model of human system and it is constructed from top to bottom as shown by the left side arrow in the Figure 4-1 from the communication with the environment unit, human inference engine, and long term memory.

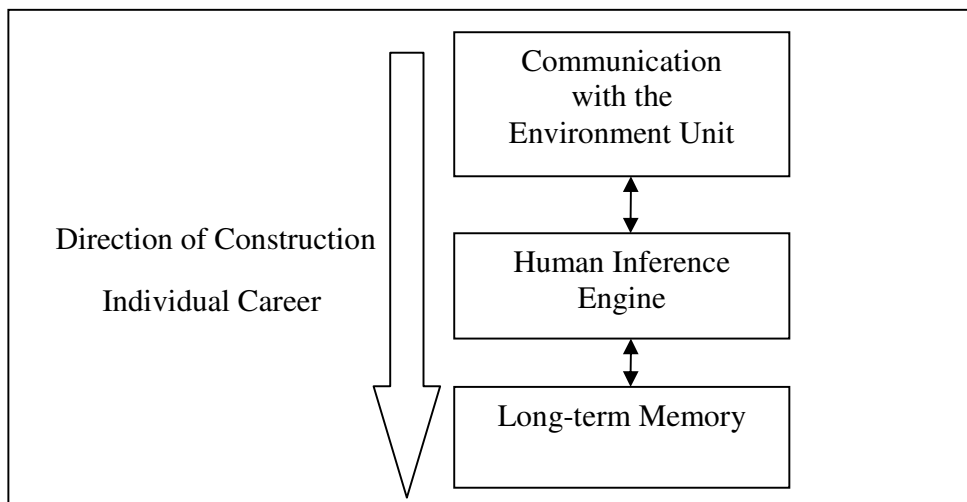


Figure 4-1: Functional Model of Human System Owaied (2007)

The knowledge-based system can be mimicked the by functional model of human system but will be implemented as seen in figure 4-2 from bottom to top direction shown by the left side arrow. Therefore, the communication with the environment unit will be as the user interface. The human inference engine will be the inference engine but will be considered to represent untouchable entities, which are; willing and needs, vision, Incentives, hobbies, and the effect of environment as problem solving method, search technique and reasoning agent. Finally the long term memory will be as the knowledge base. Then have the most important part of knowledge base system is the knowledge base since all the other parts of knowledge base system depend on the implementation of the knowledge base .

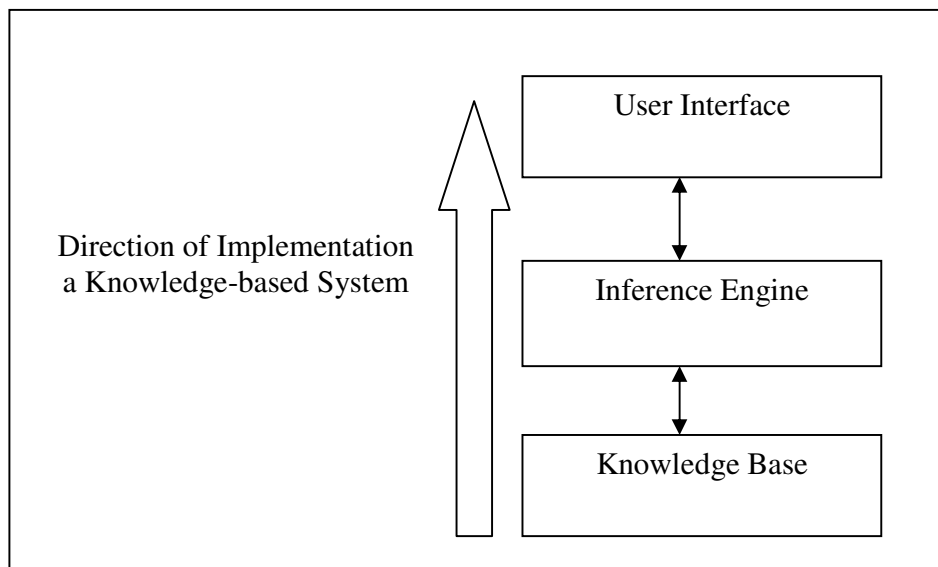


Figure 4-2: General Structure of Knowledge-based System Owaied (2007)

Figure 4-1 presents the construction of human functional system and figure 4-2 presents the implementation of the knowledge-based system as a simulation of functional model of human system

4.2 Proposed Model of Intelligent Agent as Knowledge-based System.

Figure 4-3 presents the intelligent agent as Knowledge-based System of using state space and action space for intelligent agent model.

The proposed model, is based on the general structure of Knowledge-based system in figure 4-2, consists of four modules, which are; user interface, inference engine, knowledge base, and working memory. Using the proposed model for searching an object in an environment required many tasks to be accomplished.

The intelligent model should be able to identify the path required for a task and analyze it. This ability utilizes the intelligent agent to extract the required data and information from it as knowledge, according to the general structure of Knowledge-based System. In the following subsections, we give the description of them.

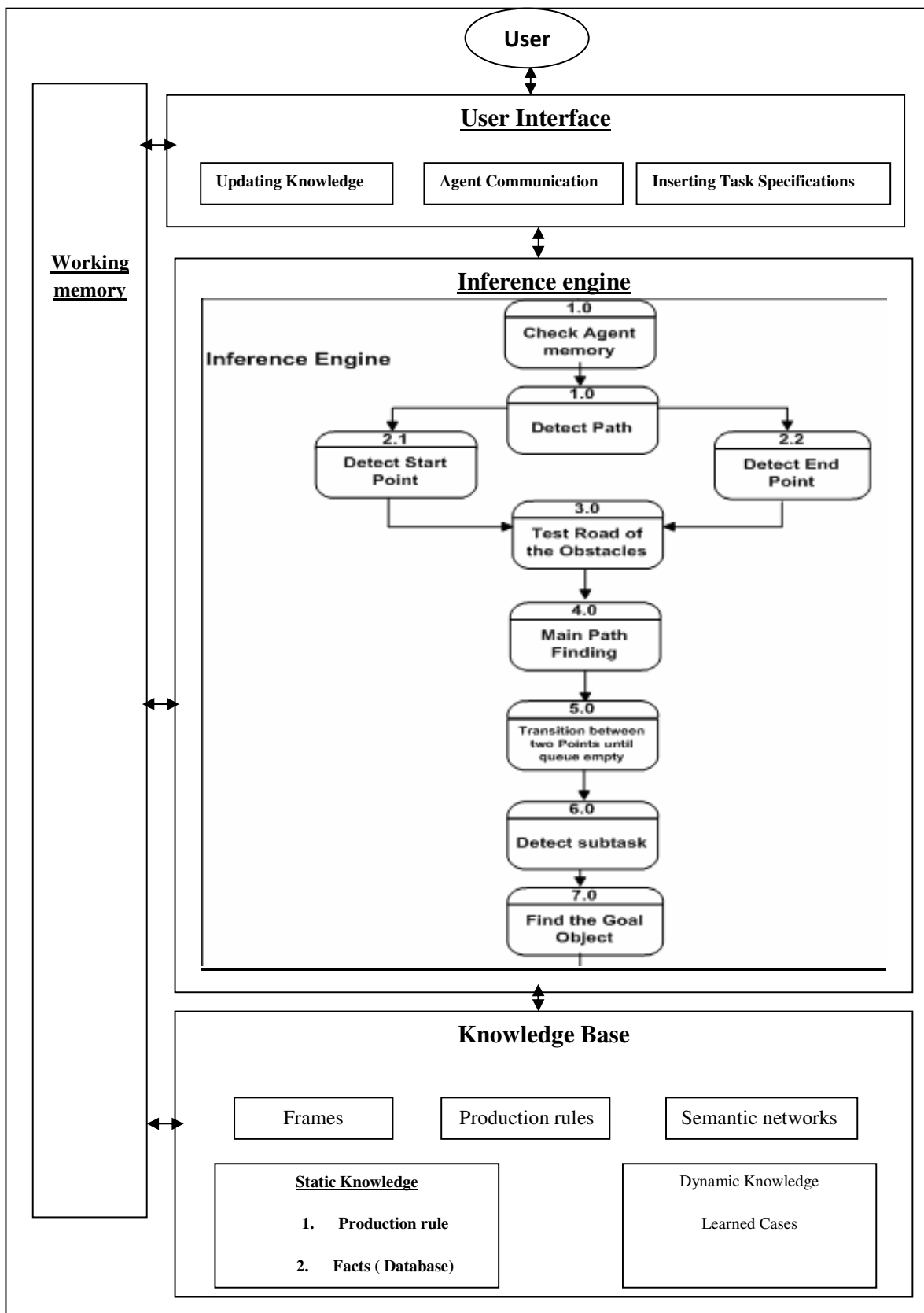


Figure 4-3: Proposed model of intelligent agent as Knowledge-based System.

4.2.1 User Interface Module

The user interface simulates the communications with the environment unit of the functional model of human system. So this module consists of many functions, all functions allow the user to interact with the knowledge based system. When user used the intelligent agent he firstly must enter a password and a user name allows him to interact with the agent. , and the user interface consists of three modules:

- Inserting task specifications : Can insert the task through the part of the proposed model for example bring book by id , bring book by name , replace two books and return book to bookcase.
- Agent Communication: Depends on the interaction with environment. The intelligent agent model is able to calculate the persistent changes in the external dynamic environment and any sudden change , such as observing the existence of any obstacle in the environment and avoiding it
- Updating Knowledge: When changing the stored knowledge and deleting it. And insertions of new knowledge are applied according to the given request by end user.

4.2.2 Inference Engine

The inference engine consists of three modules which are; problem solving method, search technique and reasoning agent. The primary functions are calling the right knowledge from knowledge base when needed, to solving problem, according to the given assertions, and using a heuristic search technique. Also clarify the ambiguity in the

new acquired knowledge inserted in the working memory, by calling an existing knowledge from static knowledge base, in order to be added to the dynamic knowledge base. For example, when change occurs in the external dynamic environment and any sudden change, such as observing the existence of any obstacle in the environment and avoiding it, the intelligent agent calls the right knowledge from table and use the heuristic search technique to find the fast solution.

The reasoning agent usually can be one of many techniques such as forward chaining, backward chaining, or mixing both. In this thesis, we will have used both according to the situation of the given goal to the proposed intelligent agent.

The algorithms and function will be used by the inference engine of the intelligent agent .Also these algorithms are used for gaining knowledge from the problem's state space, the environment's problem space, and the actions state space in order to accomplish a specific task.

When we order objects ID, for example, the first step is the intelligent agent lookup for the availability of the object (book) according to the experience (in this case in the dynamic knowledge). The intelligent agent will go to the next step to accomplish the task and detect the start point and end point, and test road of the obstacles then transit between two points and detect subtask

4.2.3 Working memory

The working memory is the place where all the activities of the inference engine and user interface and knowledge base have to be done; these activities are:

- The application of reasoning agent used: The reasoning agent usually can be one of many techniques such as forward chaining, backward chaining, or both in the proposed model use both according to the situation of the given goal to the proposed intelligent agent.
- The application of searching techniques used: using a heuristic search technique. So clarify the ambiguity in the new acquired knowledge inserted in the working memory,
- The processes of agent communication with the environment: the intelligent agent is able to learn and take the reasonable decision in the dynamic environment and when any sudden change, such as observing the existence of any obstacle in the environment and avoiding it.

4.2.4 Knowledge Base

The knowledge base represents the repository of knowledge for a narrow and specific domain. The knowledge can be defined as a set of facts, rules, events and meta-knowledge. Usually, the knowledge can be either declarative or procedural. So in this thesis, we use both declarative, which is the dynamic knowledge, and procedural, which is the static knowledge.

4.2.4.1 Static Knowledge

The knowledge is represented as a set of rules together with associated facts, each rule specifies a relation and has the form IF (conditions) THEN (action). The basic structure

of a static Knowledge is a knowledge base contains the domain knowledge used for problem solving. The form of a rule can be rewritten in Horn Clause form, as follows:

Action \leftarrow Condition-1, Condition-2... Condition-i

Where $i \geq 0$ and is an integer number.

4.2.4.2 Dynamic Knowledge

The dynamic knowledge contains knowledge acquired during the run time of the system via the interaction of the user with the proposed system. This knowledge will be as cases for the specific problem or facts. The basic structure of dynamic knowledge is the database that includes a set of facts used to match with the (condition) part of rules stored in the knowledge base.

4.3 Knowledge Representation methodology

There are many forms that have been used to represent knowledge in the knowledge base, such as rule base, semantic nets, production rules, frame structure, and can be hierarchical structure of frames.

In this thesis, frames in conjunction with production rules and semantic networks are all used to represent knowledge in the knowledge base.

4.3.1 Frames

Marvin Minsky in 1974 proposed the concept of frames as structures where each frame has its own name and a set of attributes, or slots, associated with it. Wesley (2005)

described why is it necessary to use frames and provide a natural way for the structured and representation of knowledge. In a single entity, a frame combines all necessary knowledge about a particular object or concept. The frame provides a means of organizing knowledge in slots to describe various attributes and characteristics of the object, as seen in Figure 4 - 5.

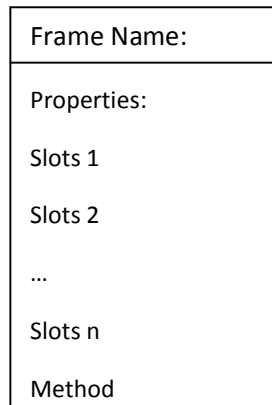


Figure 4 - 4: Frame Structure.

4.3.2 Semantic networks

A semantic network is a directed graph, consists of nodes each node is an object in the environment, and links (arcs) between the objects, where a link is a relation between objects.

Michael (2005) claimed that in general, there are three types of relationships between objects, these are:

- Generalization: 'is-a' relationship between a superclass and its subclasses .For example, a bookcase **is a** Myobject. Each subclass inherits all features of the superclass.

- Aggregation: is ‘a-part-of’ or ‘part-whole’ relationship in which several subclasses representing components are associated with a superclass representing a whole. For example, a shelf is a part of bookcase.
- Association: describes some semantic relationship between different classes which are unrelated.

In this thesis the semantic net is used for representing the knowledge associated with the environment; figure 4-6 illustrate the semantic network for the concept environment.

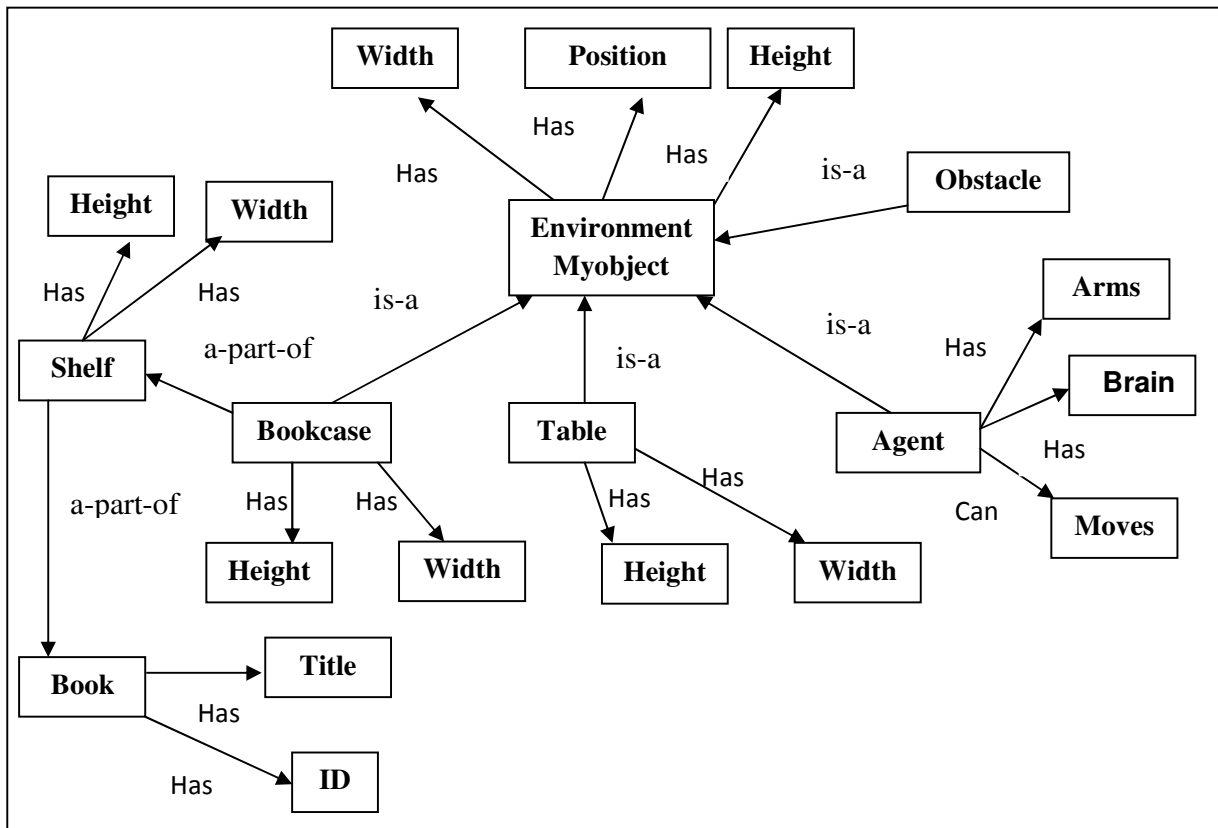


Figure 4-5: Semantic networks showing some object concepts and properties.

4.3.3 Rule Base

The rule base is a set of the form **IF (Conditions) THEN (Action)** rules. The conditions are a part of a rule which contains zero or more conditions, the special case of zero occurs in case of a fact. The action part is the goal. Figure 4-7 illustrates an example of applying the production system using the working memory.

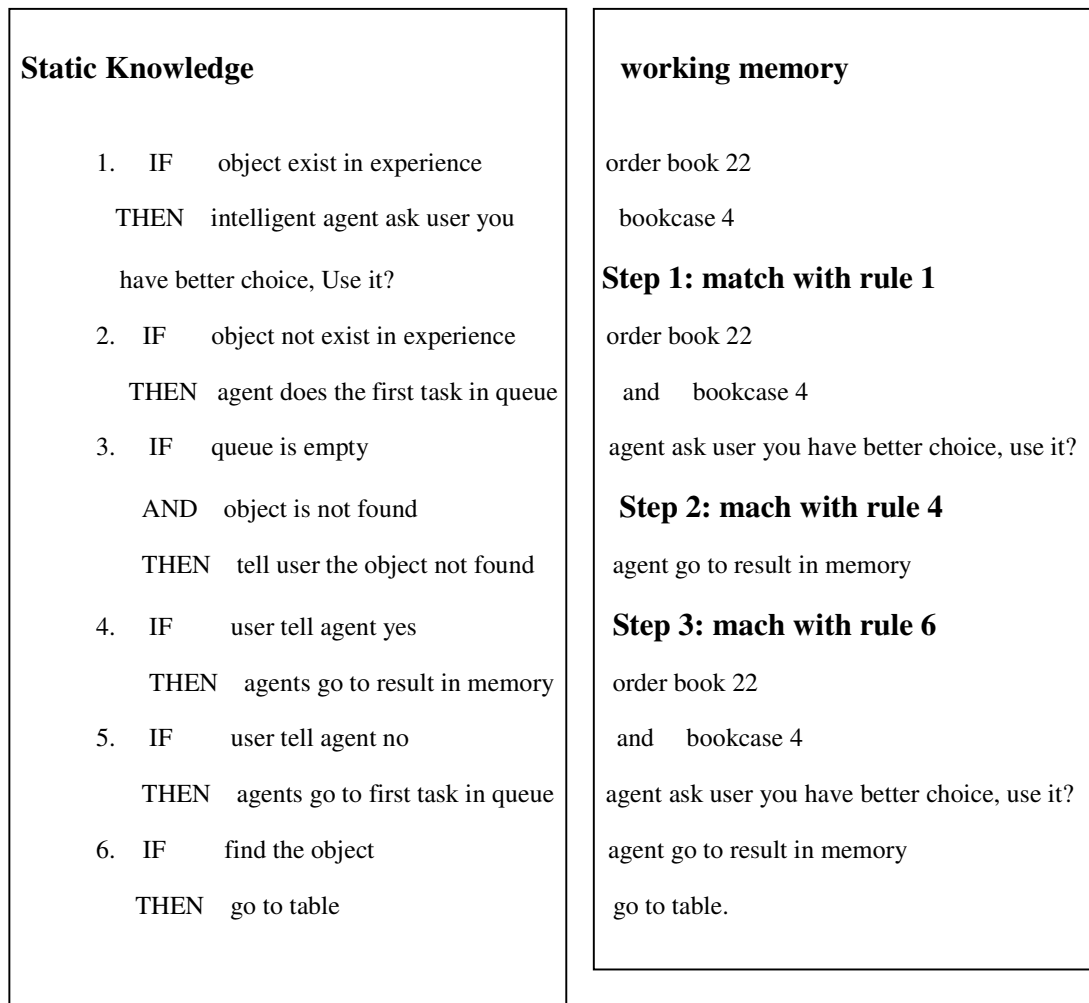


Figure 4-6: production system using working memory

For each cycle of production system the following actions will happen:

- ✓ Selection: select the rules from rule base according to the goal.
- ✓ Matching: match the selected rules according to the assertions given in the context (database).
- ✓ Confliktion: choose one of the matched rules according to the time and space.
- ✓ Fire (execute) the chosen one.

4.4 Hierarchical Structure of Environment

The environment contains many objects and each object composites another object in environment, and of each object represented by the class contains the properties and methods that describe behavior of the object, where properties describe the distance of the object and class of object or the blueprint of an object. Figure 4-8 present hierarchical structure of environment.

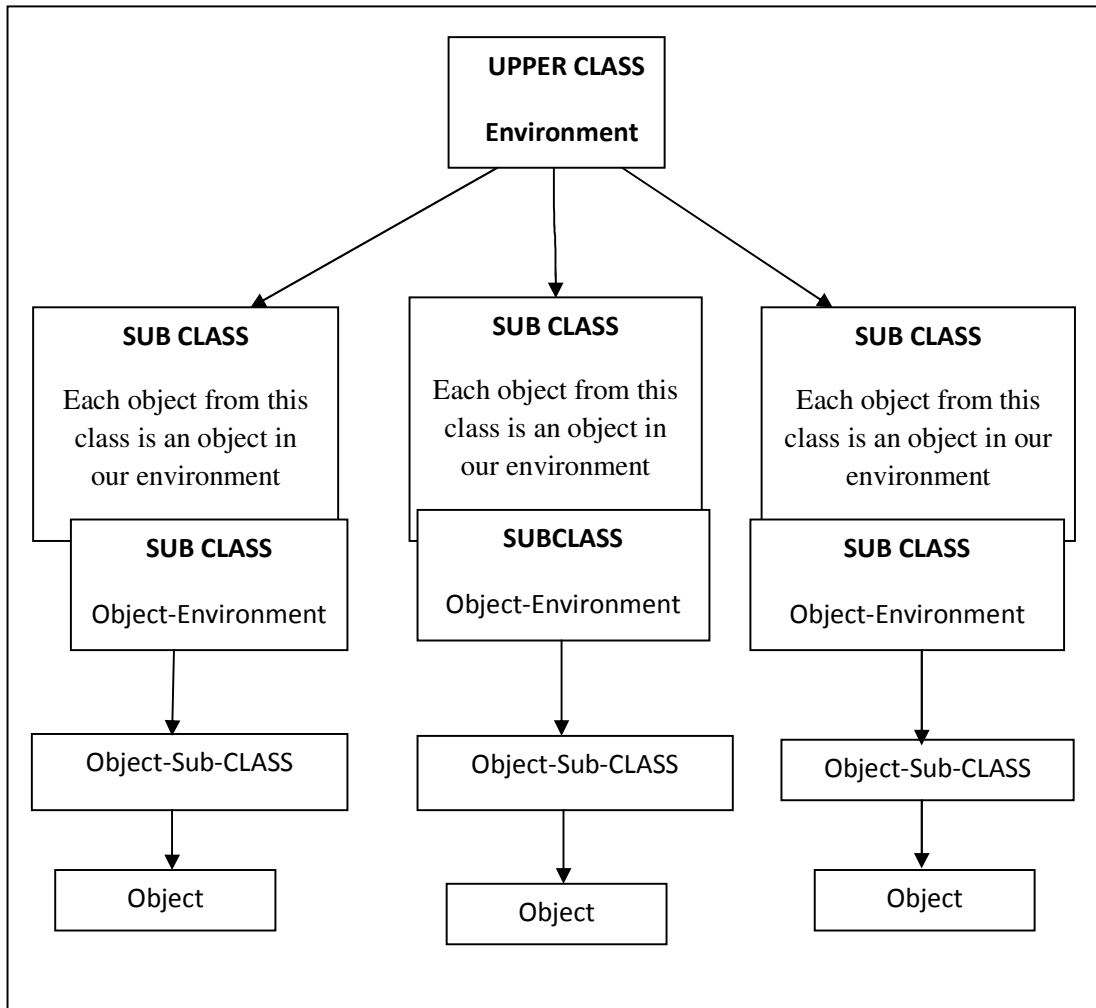


Figure 4-7: Hierarchical Structure of Environment.

Chapter Five

THE IMPLEMENTATION OF PROPOSED MODEL

Chapter Five

The Implementation of Proposed Model

5.1 Overview

This chapter presents the methodology of describing the state space and action space and the implementation of this methodology. The methodology consists of many functions and algorithms; these algorithms are used for gaining knowledge from the state space of environment so as to build the task. The case used for implementation of the proposed model is the library environment. So the sections 5.2-5.9 are the description for the implementation in this specific environment.

5.2 Control Data Flow Diagram for the task

Declare the main process, in memory of the intelligent agent, when given task the first step is to determine the task and then how to reach that goal. Then analyze the knowledge in the environment state space and action space which is related to the goal.

The intelligent agent can understand the environment in any position of environment and can detect the subtask that is needed to do and arrange in queue then execute many subtasks to task. Then find the start and end point based on the position of the intelligent agent for each subtask, then plan the path and test the road of the obstacle, also determine the procedure needed to reach that goal. Finally, the task representation is

stored in the knowledge base. Figure 5-1 shows the control data flow diagram for the task when order Bring Book ID or Bring Book by Name.

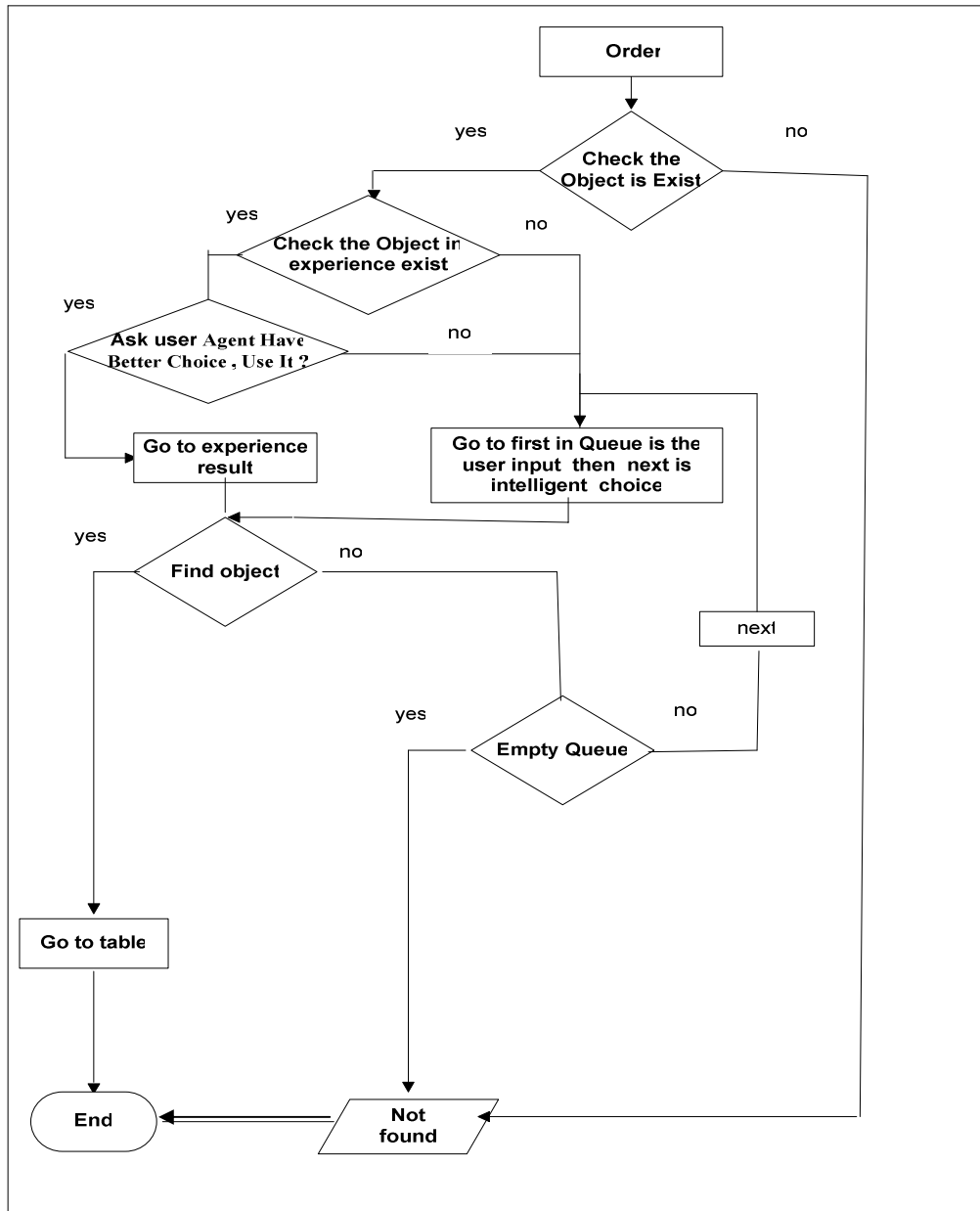


Figure 5-1: The Control Data Flow Diagram for the task when order Bring Book ID

5.3 The Interactions of Event Processing and Agent Behavior

Since the primary task of the intelligent agent model is to find and get an object, so the user firstly inserts the object characteristics. In the case of library environment, the object characteristics may have many entities such as book-name, book-number, book-place, or any other entity Therefore, the intelligent agent looks up for the availability of the object (book) according to the experience (in this case in the dynamic knowledge) . If exists then the intelligent agent will go to the next step to accomplish the task. If it does not exist, then the intelligent agent calls the procedure called fillIndexQueue then starts execute task in queue sequence to reach the target that is stored in the environment.

When order objects ID for example the first step is the intelligent agent lookup for the availability of the object (book) according to the experience (in this case in the dynamic knowledge).If exists then the intelligent agent will go to the next step to accomplish the task.So the intelligent agent will know the path of the book in the knowledge base that is stored directly resulting from previous experiences in Knowledge base in dynamic knowledge.

And when taking another action for example, replace two books then agent updates the data in dynamic knowledge. The agent can learn from experience and can take best action and best time to do action .Also agent can interact with environment, without any help from user and without any assistance or guidance for decision-making.

The state transitions are illustrated in Figure 5-2 which show the start-up sequence.

And present the interactions for event processing and intelligent agent behavior .

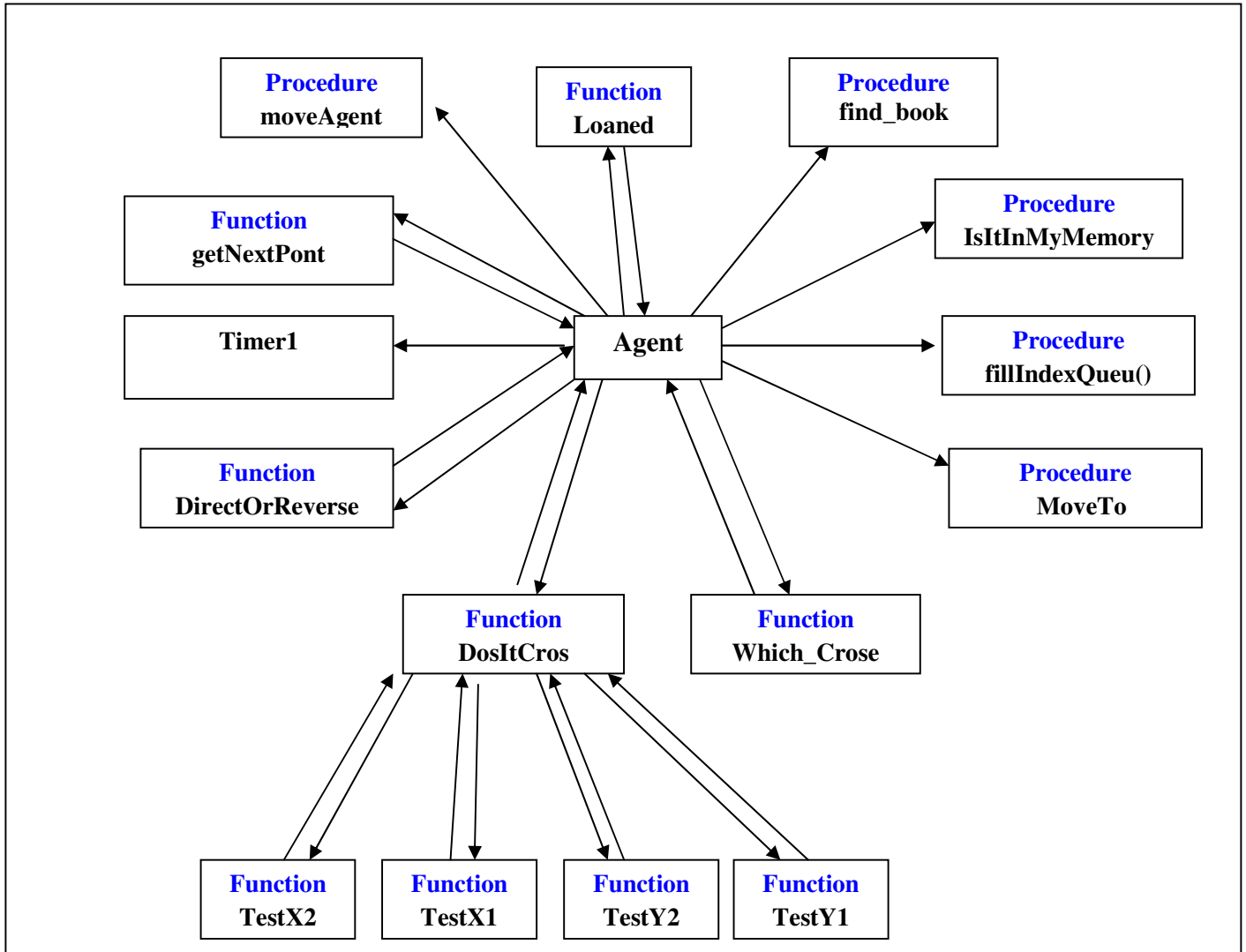


Figure 5-2 shows the start-up sequence. And presents the interactions for event processing and agent behavior

Telling agent what user needs means detect the task then agent starts to interact with environment the first Function is Loaned check if book loaned or not if not loaned then go to Procedure find_book which first must be checked if book in dynamic knowledge mean in experience to agent by Procedure IsItInMyMemory if agent exists book in experience then go to Procedure fillIndexQueue and put the first number in queue number bookcase exist object in dynamic knowledge else if it does not exist in experience then Continueand fill queue intelligently .

Then start execute main task, start execute sub task in queue, Procedure MoveTo, detect the bookcase need go to then execute Function Which_Crosc check if any obstacle intersection with the path plan when go to the bookcase.

Function DosItCros call four function Function TestX1 , Function TestX2 , Function TestY1 and Function TestY2 each calculate if intersection path with obstacle or no intersection .

Function DirectOrReverse check any direction go agent if have best time and shortest path .

Timer1 executes walk, each step 0.01 seconds and then calculate the next point in the line until walk arrives the end point, use Function getNextPont to calculate the next point in line , then move to next point until end point by Procedure moveAgent . Then when arrive target one ,the agent performs subtask is search for the object need and match object needed with any object, in bookcase then if find object then go to table else continue execute the next task in queue .

5.4 Present the first screen user interface to program

When the user needs to use the program and deal with the agent he must enter a password and users name a special person allows him to deal with the agent in the environment and any task assigned to the agent will be stored in its own account and cannot return the book by the name of another user only through the user account can return the book on loan. Figure 5-3 illustrates the first screen appears when any person used programs. And this screen allows only the user to have privilege enter to the program.

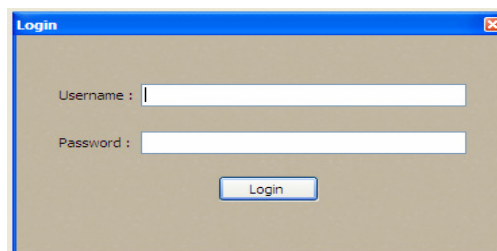


Figure 5-3: represent first screen user interface to program

5.5 Describe the action which can be given to agent in program

The environment contains many tasks such as the environment proposed model library can the agent performs many tasks for example bring book by id , bring book by name , replace two books and return books to bookcase. Figure 5-4 represents sub user interface to program describe the task in environment.

The program is easily used and the user friendly means that the program is smart. This means that everything gives pleasure and happiness to humans. Also any user can deal with program and understand how to deal with it, section search book used show all objects in environment and take the name or id then use it in a given task to agent. The section My Books tells the user what the books loan is to the user.

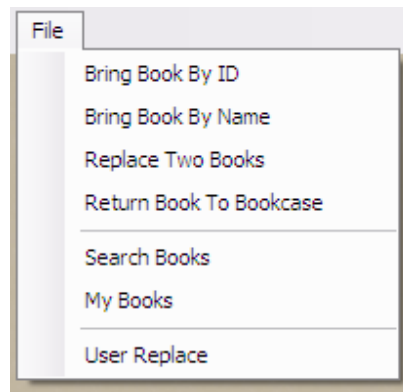


Figure 5-4: represents sub user interface to program proposed model describes the environment library.

When to tell agent return book to bookcase, the agent must be sure if the book loaned the same user interact with program and when to tell agent to bring the book he must sure if the book exists in experience and not loaned. when to tell agent replace two books and one for each book loaned agent don't replace two book and tell user the book loaned ,also when agent brings each book and then returns the same book and then replaces the same book with another on the agent must update the knowledge in a dynamic knowledge .

The purpose of demonstrated is to test the function and algorithms of the task in the environment. Figure 5-5 shows the environment proposed model and explains all measurements in environment and measure between object. Take the measure use the map blueprint to library.

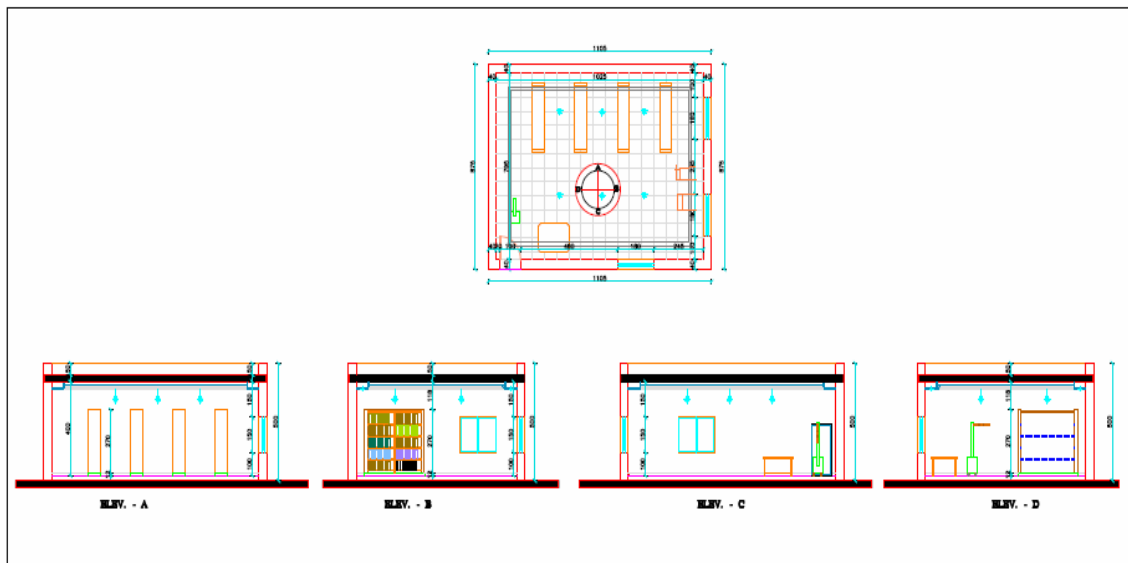


Figure 5-5: Explains measurement inside of the library proposed model.

5.6 Representing Knowledge in the Knowledge base

The class structure is used in thesis to represent the environment in the knowledge base, which depends on three types of knowledge representations .The production rule and semantic net and frames where each object in the environment is instance of a class containing the following attributes and methods of the object:

➤ **Attributes:** Things exist in the object and can not be separated for the object its properties and are concomitant with the object.

- The object position: each object contains many properties, describes the properties in the class and each object has the position in environment.
- Length and width of the object: is attributing for each object in environment consists of the dimensions length and width.
- Name: each object has a name.

➤ **Methods :**

- Methods of the object: mean what we can do to the object, anything that could be implemented by the object such as intelligent agent can move in the environment between two states.

An **object** can be used to denote anything. It could be a concrete concept such as a book or table or bookcase, a **class** provides a framework for describing a type of object.

The class is the blueprint. An object is an instantiation of a class. For instance, a bookcase has features such as number of shelf, ID number. The class 'bookcase' provides a generic description using features, but an instance is an object that has values assigned to these features, the word 'property' is used to denote an attribute of an object. Properties are pieces of information about an object, such as color, size, and number of shelf. Methods describe behavior in that they declare what an object can do, for example, intelligent agent object might have been moved that describes the task content the

number of move is a property. Figure 5-6 shows the Inheritance Class Structure for the whole object in environment.

The class represents the object in environment, also each class is concerted the blueprint of object in environment.

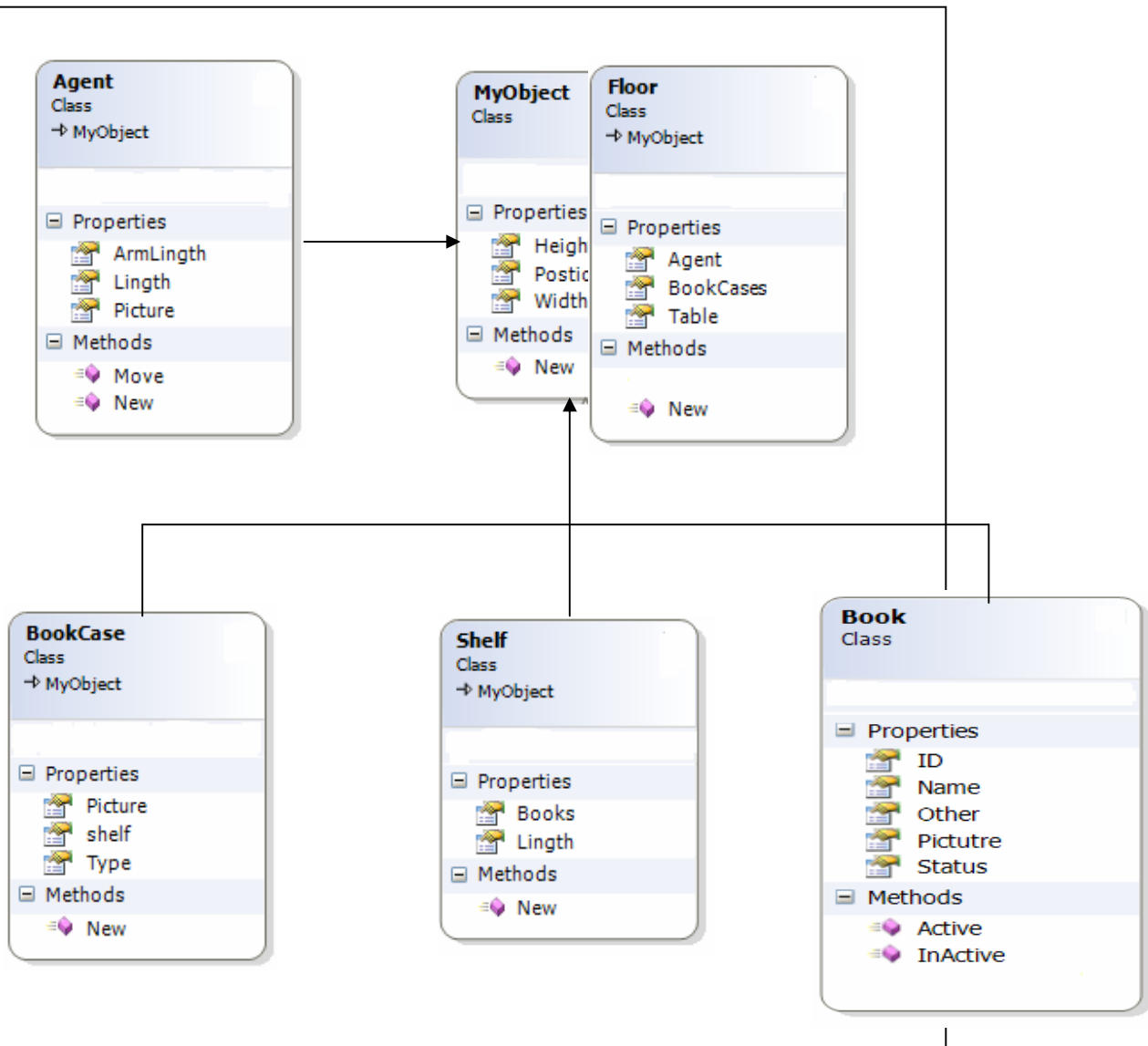


Figure 5-6: Inheritance Frame Structure.

5.7 Hierarchical Structure of Proposed Model (Library)

Figure 5-7 represents the hierarchical structure of environment for proposed model library and present the object that contains the library .For example, bookcase, table, also composite of object such as bookcase composite shelf and shelf composite book, and each of object contains properties and measurements commensurate with the object containing this object and also fit with the environment and the location of each object depends on the design environment.

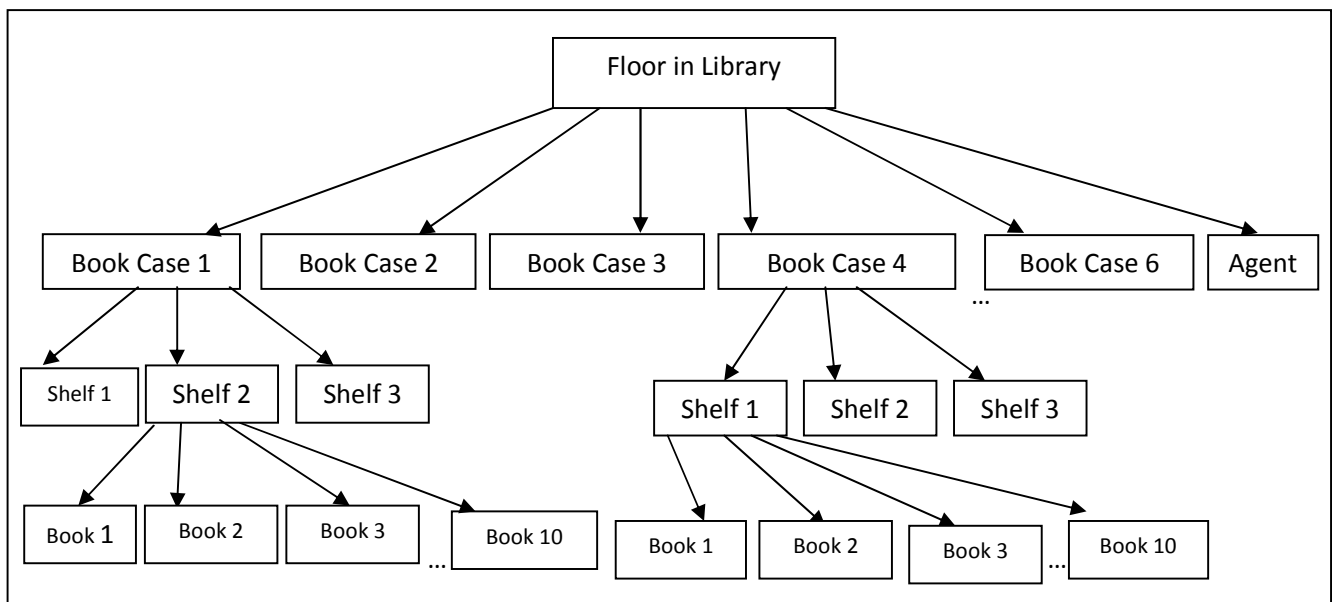


Figure 5-7: Hierarchical Structure of Proposed Model (Library).

5.8 Environment

Environments will be divided into two categories: how to represent environment space (state space) and how to represent (action space) event. Therefore, the first step of doing this represents the environment.

State Space is a complete and comprehensive description of the environment that the intelligent agent will perform within, including:

- Physical Space Layout Example: Building/Spacing measurements.
- Inner Spacing Layout Example: Shelves details.
- Constraints Example: Relative allowed distance.
- Obstacles (Permanent or Temporary) Example: Columns/Sculptures or Maintenance.
- Limitations Example: Moving objects.

Action Space is a complete list of available actions and events within the given environment. These can be categorized into two main categories:

- **General/Simple Actions:**

Examples: MoveForward (), MoveBackward (), MoveLeft (), MoveRight ()
.....etc.

- **Specific/Complex Actions:**

Examples: BringBookbyID (), BringBookbyName (), ReplaceTwoBook (), ReturnBookToBookcase () ... etc.

Properties of task environments in environment' library' proposed model:

- Fully Observable.
- Dynamic environment.
- Contain single intelligent agent.
- Episode.
- Continuous state and discrete actions.
- Stochastic domain

The intelligent agent interacts with the environment, the resurrection of events to reach the required task suppose that the environment is composed of a set of a finite states.

Indeed an agent can have a high degree of intelligence, autonomy, flexibility and adaptation that if it possesses capacities of knowledge learning.

Set $S = \{s_1, s_2, \dots\}$.

And the intelligent agent must transit from one state to another state when given task the transition produce event that the environment consists of a set of finite events.

Set $E = \{e_1, e_2, \dots\}$.

When to do a specific task in this an environment main task will consist of a subtask sequence of states and events.

$$\text{Main Task } t: S_1 \xrightarrow{e^1} S_2 \xrightarrow{e^2} S_3 \xrightarrow{e^3} S_4 \xrightarrow{e^4} \dots \xrightarrow{e^{n-1}} S_n$$

Let t be a set of all such possible task and the transition between two states contain the event. $TR: 2S \rightarrow e_1$

Figure 5-8 represents the environment proposed model which contains single intelligent agent. The intelligent agent can interact with environment so performed the task. Also Intelligent agent transitions between states until you finish the task need.

S is a set of states and E is a set of events performed by A agent in environment and t is a task done by agent and the task contains many transitions between state and do the event each two state and the s_1 is the current state to agent .



Figure 5-8: Environment Proposed model

The environment consists of a set of states and events, the transitions and the initial state, and intelligent agent. **Environment** = {S, E, A, t, s₁}.

5.9 State Space and Action Space in Environment

Figure 5-9 represents an example of the state transition of an environment; the arcs between two states show the sets of actions corresponding to transitions.

Intelligent Agents are not allowed to perform the same action twice. For example, if the intelligent agent reached state S7 by performing e₇ then e₂₆, would not be able to perform e₂₆ again in order to reach S5, mean cannot visit any state or any position More than once .

Each state in the graph is the object of a class and the state S2, S3, S4, S5, S6, and S7 are the bookcase in a proposed model and the state S1 is a table in library and S0 is the initial state mean is the current position of agent.

Detect move to the agent based on the task given to agent such as the user is need to bring book or replace it between two books or return it .When agent the gives task then he can interact with environment until he reach on the goal.

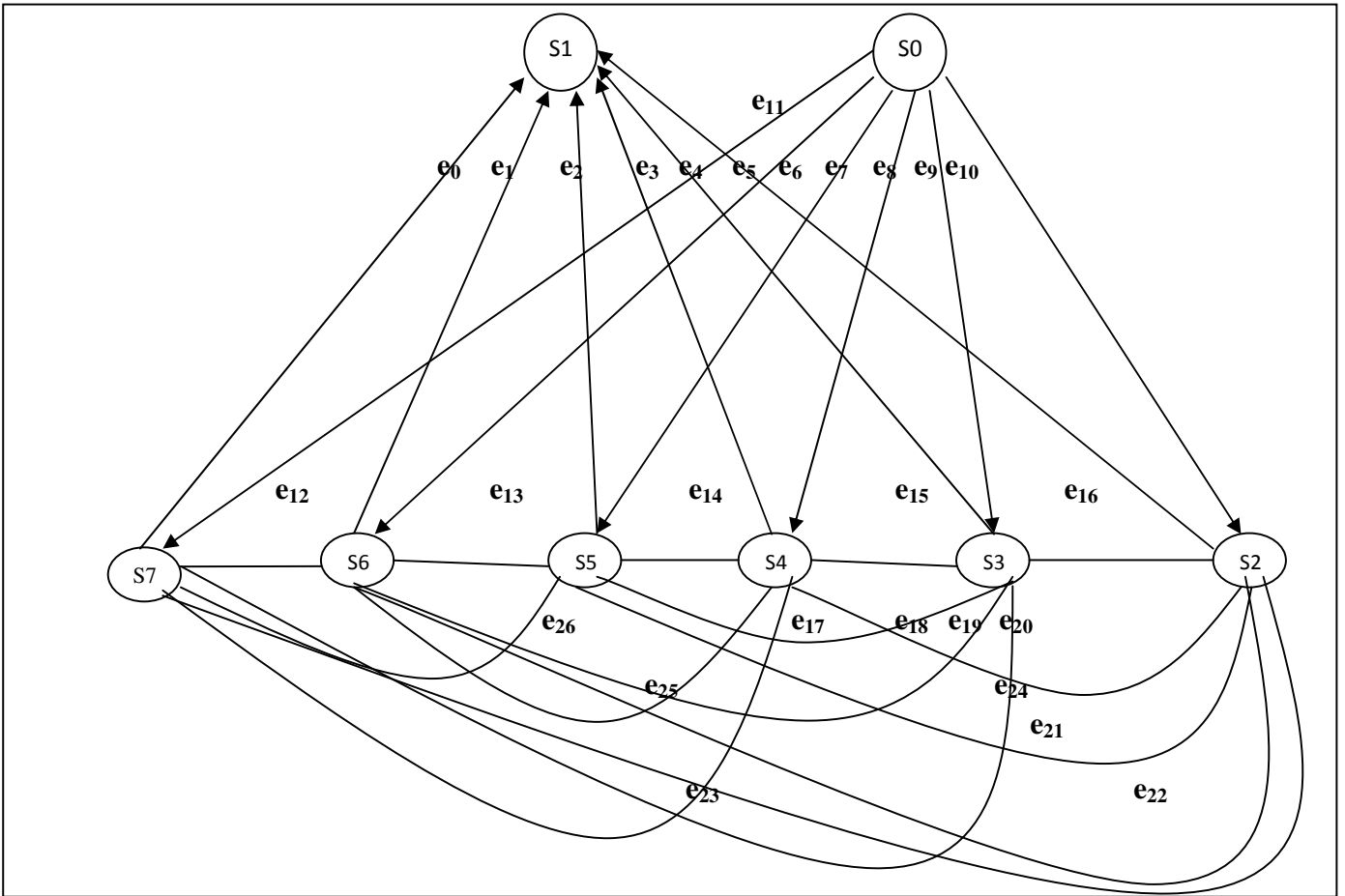


Figure 5-9: Transition in Environment (State, Event)

Example: Illustrated by the graph in figure: 5-9 the environment state transition. In this environment, an intelligent agent has many available actions, and the intelligent agent in the environment is not allowed to execute the same action twice. Arcs between states in figure 5-9 are shows the actions that cause the state transitions.

✓ Note: the start position of intelligent agent state S_0 for each goal.

- $Goal_1 = \{S_3\}$

An intelligent agent can reliably achieve $goal_1$ by performing action in the start position of intelligent agent choose an action $e_{10}, e_9, e_8, e_7, e_6$ or e_{11} , the result of which

will be either S_2, S_3, S_4, S_5, S_6 or S_7 . If S_5 results, the intelligent agent can perform $e_{13}, e_{26}, e_{21}, e_{18}, e_2$ or e_{14} the result of which will be either S_2, S_3, S_4, S_6, S_1 or S_7 . If S_3 result it can simply perform e_{18} .

- $Goal_2 = \{S_3\}$

An intelligent agent can reliably achieve $goal_2$ by performing action in the start position of intelligent agent choice of action $e_{10}, e_9, e_8, e_7, e_6$ or e_{11} , the result of which will be either S_2, S_3, S_4, S_5, S_6 or S_7 . If S_2 results, the intelligent agent can perform $e_{22}, e_{24}, e_{21}, e_{16}, e_5$ or e_{23} the result of which will be either S_5, S_3, S_4, S_6, S_1 or S_7 . If S_1 result it can simply perform e_5 , which is not allowed. Mean does not allow going to table before finds the book.

5.10 The Intelligent Agent Implementation in Library Environment

Figure 5-10 presents the main algorithms as Processes for the proposed model as knowledge Based System for one process in queue and mimics the human behavior (which is a librarian in this implementation) to recognize an object, which is a book, in the environment (library) in the following steps:

- ✓ The first step is a full description of the existing environment through the Knowledge base which will be described as the state space and action space and the existing storage environment.
- ✓ The second step is to allow the intelligent agent model to understand and analyze the nature of the environment through the interaction of the user with the existing knowledge base. So we can plan the movement and do the task to reach the desired book.

- ✓ The third step represents the path that draws the action of event and reaches the goal according to the book in the requirement.

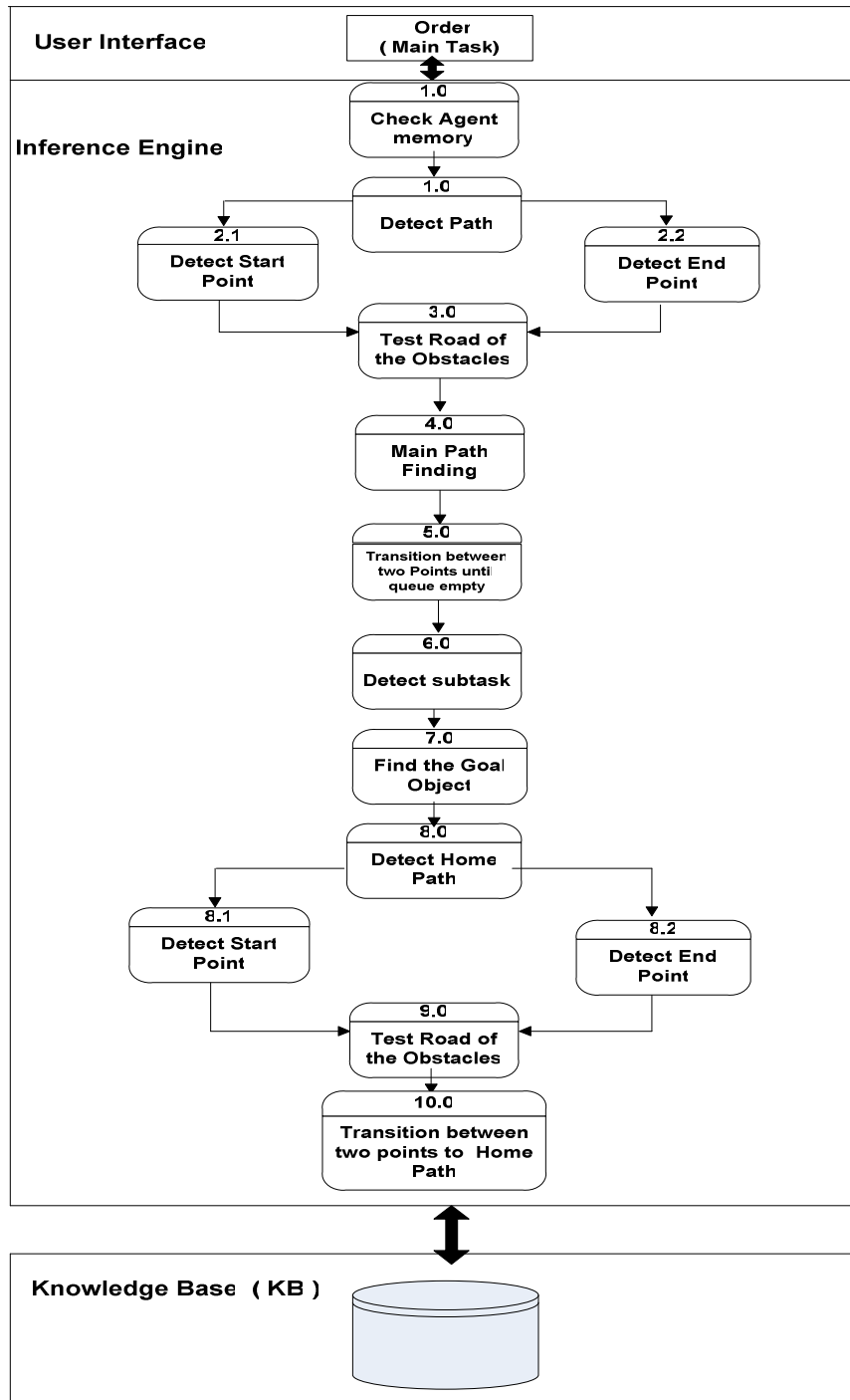


Figure 5-10: Main Algorithms as Processes in a knowledge Based System

The algorithms and function will be used by the inference engine of the intelligent agent .Also these algorithms are used for gaining knowledge from the problem's state space, the environment's problem space, and the actions state space in order to accomplish a specific task, these algorithms will be explained in the next section.

5.11 The Algorithms and Functions

In this section are the declarations and the brief descriptions of all the algorithms and functions used by the inference engine of the proposed intelligent agent model. Also there are descriptions of the structures of all the algorithms, their implementation as procedures, identifying both the input and output for each procedure and function and its role in whole model.

5.11.1 Procedure Check for Object Availability

Since the primary task of the intelligent agent model is to find and get an object, so the user firstly inserts the object characteristics. In the case of library environment, the object characteristics may have many entities such as book-name, book-number, book-place, or any other entities. Therefore, the intelligent agent looks up for the availability of the object (book) according to the experience (in this case in the dynamic knowledge or working memory) .If it exists then the intelligent agent will go to the next step to accomplish the task. If it does not exist then the intelligent agent calls for the procedure called fillIndexQueu.

5.11.2 Procedure fillIndexQueue.

The procedure FillIndexQueue is used to arrange all the sub-tasks of the main one to be performed by the intelligent agent as seen in figure 5-11.

```
Sub fillIndexQueue (ByVal BID As String)

IndexQueue = GenNewQueue (BID, BookCaseIndex)

End Sub
```

Figure 5-11: procedure Fill Index Queue

5.11.3 Function for the Heuristic search Technique

The heuristic search technique is implemented as function called (GenNewQueue) and this function is to find the faster solution, but not necessary the optimal one. Usually, these search techniques depend on the heuristic information which is founded in the problem space. Therefore, this function will depend on the weight of an object in its environment. Using Initialization weight, sorts all weight to each object in all positions in environment, as seen in figure 5-12, so the general structure of the heuristic function is:

$$F(n) = g(n) + h(n)$$

$g(n)$ = the search technique returns the weight of an object in sub-environment

$h(n)$ = the heuristic information for the object n

```

Function GenNewQueue(ByVal bid, ByVal indexToStart) As Queue

    Dim ds As New Data.DataSet
    ds.ReadXml("data.xml")
    Dim Q As New Queue
    Dim ar(5) As BBKS
    For i As Integer = 0 To 5
        ar(i).Vlaue = ds.Tables(0).Rows(i)("b" & bid)
        ar(i).Index = i
    Next
    ar = SortIt(ar)
    Dim ar2(4) As BBKS
    For k As Integer = 0 To 5
        If ar(k).Index = indexToStart Then
            Dim temp As BBKS = ar(0)
            ar(0) = ar(k)
            ar(k) = temp
        End If
    Next
    For o As Integer = 1 To 5
        ar2(o - 1) = ar(o)
    Next
    ar2 = SortIt(ar2)

    For u As Integer = 1 To 5
        ar(u) = ar2(u - 1)
    Next
    For j As Integer = 0 To 5
        Q.Enqueue(ar(j).Index)
    Next
    Return Q
End Function

```

Figure 5-12: Function for the Heuristic search Technique

Each object in the environment has an initial value of weight and the value changes when any transition of object and each object are linked with object-environment as seen in figure 5-13.

```

Sub increseWeight(ByVal bid)

    Dim ds As New Data.DataSet
    ds.ReadXml("data.xml")
    h=1
    Dim g As Integer = ds.Tables(0).Rows(BookCaseIndex)("b" & bid)
    F = g + h
    ds.Tables(0).Rows(BookCaseIndex)("b" & bid) = F
    ds.WriteXml("data.xml")

End Sub

```

Figure 5-13: Heuristic Function

Table 5-14 illustrates the two dimensions array which contains all weight of transition of all objects in environment and when needed fill queue, it must arrange all bookcase needed visit to depend or the max weight in each column contain object need.

Bookcase / object	Object ₁	Object ₂	Object ₃	Object _n
Bookcase ₁	Weight ₁₁	Weight ₁₂	Weight ₁₃						Weight _{1n}
Bookcase ₂	Weight ₂₁	Weight ₂₂	Weight ₂₃						Weight _{2n}
Bookcase ₃	Weight ₃₁	Weight ₃₂	Weight ₃₃						Weight _{3n}
Bookcase ₄	Weight ₄₁	Weight ₄₂	Weight ₄₃						Weight _{4n}
Bookcase ₅	Weight ₅₁	Weight ₅₂	Weight ₅₃						Weight _{5n}
Bookcase ₆	Weight ₆₁	Weight ₆₂	Weight ₆₃						Weight _{6n}

Table 5-14 Represents the two diminutions array which contains all weight

5.11.4 Function Insertion Sort

Use a sorting algorithm that is efficient for small arrays. That is used to sort the weight for each object in all sub environments . Therefore, this function insertion sorts as seen in figure 5-15.

```

For j ← 1 to length (b)-1

    num ← b[ j ]

    i ← j - 1

    While i >= 0 and b [ i ] < num

        b[ i +1 ] ← b[ i ]

        i ← i -1

    b [i +1] ← num

```

Figure 5-15: The pseudo code for sorting insertion

5.11.5 Function for Start Point Finding.

As mentioned on the scenario, intelligent agent should specify start and end points. To recognize a start point, an algorithm was made. As shown in Figure 5-16, the start point is the initial state (position of intelligent agent start) for the transition between the initial state changes when moving from one state to another to perform the specific task, the end point of the task₁ is the starting point of the task₂, and so on in each transition.

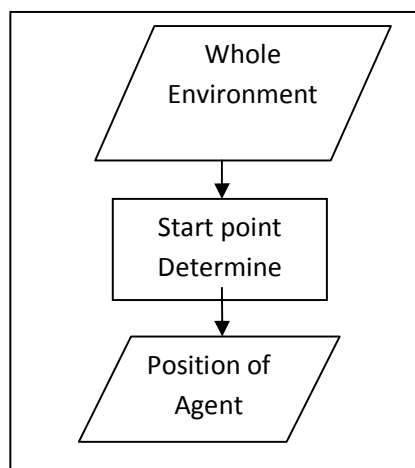


Figure 5-16: Start Point Finding Algorithm Pseudo code

5.11.6 Function End Point Finding.

This algorithm is similar to the previous one, which was used to recognize the end point. An algorithm was made (As shown in Figure 5-17) to find the end point in the final state (position of target) for the transition between states the end state change when moving from one state to another to perform the specific task. The start point of the task₂ is the ending point of the task₁, and so on in each transition.

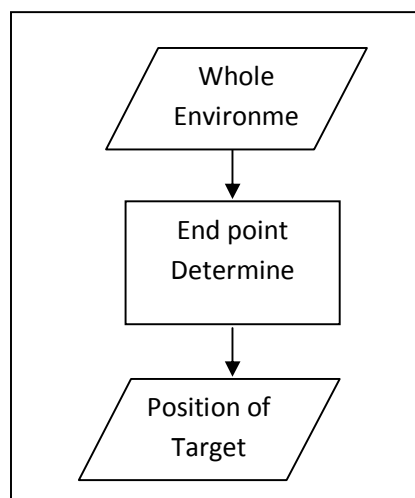


Figure 5-17: End Point Finding Algorithm Pseudo code

5.11.7 Procedure Main Path Finding

As in the scenario, the start point should be in the main path of the targeted object. Then all of its pixels are part from the main path pixels.

As shown in Figure 5-18, to find the main path, the algorithm needs to get next point from the line between start point and end point.

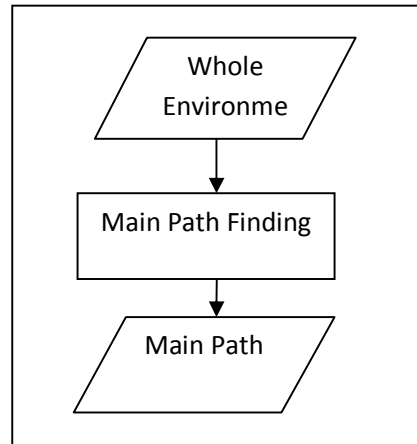


Figure 5-18: Main Path Finding Algorithm Pseudo code

Intelligent Agent Finding the Shortest Path finding algorithm used in this thesis is to detect the minimum distance between two points by using a well known mathematical equation of distance between two points.

$$D = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

Where:

D: the distance between two points

(x_1, y_1) : The Start point,

(x_2, y_2) : The Goal point

Trying to obtain the shortest path, the algorithm will calculate the minimum distance between (goal point) and all pixels through the path starting from the (start point). The algorithm initially takes the first location (start point) and finds the distance between it and the (goal point), then it calculates the next point in line of the current point (start point), the algorithm will continually calculate the pixels one by one and choose the pixel that satisfies the least distance between selected pixel and the (goal point) The algorithm will repeat this process each line in the main path between two state until the last point of the (goal point).

5.11.8 Function getNextPoint

Function explains the next step to intelligent agent which must move to reach the target as shown in Figure 5-19.

```
Function getNextPoint(ByVal _start As Point, ByVal _end As Point, ByVal index As Double, ByVal L As Integer) As Point
    Dim x As Integer = (_start.X) + index * (_end.X - _start.X)
    Dim y As Integer = (_start.Y) + index * (_end.Y - _start.Y)
    Return New Point(x, y)
End Function
```

Figure 5-19: Function getNextPoint

Calculate the next point in line of the current point (start point), the algorithm will continually calculate the pixels one by one and choose the pixel that satisfies the least distance between selected pixel and the (goal point). The algorithm will repeat this process each line in the main path between two states until the last point of the (goal point). Index is the step of agent divided by the length of the distance between start point and end point.

Index = 1/ length

$$\text{Length} = \sqrt{(s.x - e.x)^2 + (s.y - e.y)^2}$$

$$X = \text{start.x} + \text{index} * (\text{end.x} - \text{start.x})$$

$$Y = \text{start.y} + \text{index} * (\text{end.y} - \text{start.y})$$

Next Point (X,Y)

Find next point (x_1, y_1) to the Current point, in line between two states. And each step to agent execute in 0.01 second, calculate next state each transition to next state when count final equal one then line finish then go to next line or if not contain next line mean that is current position is a target .

5.11.9 Function Test Main Path of the Obstacle

Test the main path between start point and end point which does not contain any obstacles.

Will call the following functions and the calculation of equations to get the value of X1, X2, Y1 and Y2 then check intersection any point to all obstacle exists in environment.

Mathematical equation1 .as figures 5-20.

$$\text{Index} = (\text{obstacle location.x} - \text{Start position.x}) / (\text{End position.x} - \text{Start position.x})$$

$$Y1 = \text{Start position.y} + \text{Index} * (\text{End position.y} - \text{Start position.y})$$

Figure 5-20: Mathematical equation..1

```
'TestY1(agent.Postion=sp(start point ),bookcase.Postion = ep(end point
), obstacle.Location.X= x1)

Dim Y1 As Double = TestY1(a.Postion, b.Postion, obs.Location.X)

Function TestY1(ByVal sp As Point, ByVal ep As Point, ByVal x1 As
Double) As Double

    Dim index, y As Double
    index = (x1 - sp.X) / (ep.X - sp.X)
    y = (sp.Y) + index * (ep.Y - sp.Y)
    Return y
End Function
```

Mathematical equation2 .as figures 5-21

$$\text{Index} = \frac{((\text{obstacle location.x} + \text{obstacle. width}) - \text{Start position.x})}{(\text{End position.x} - \text{Start position.x})}$$

$$Y2 = \text{Start position.y} + \text{Index} * (\text{End position.y} - \text{Start position.y})$$

Figure 5-21: Mathematical equation..2

```
'TestY2(agent.Postion=sp(start point ),bookcase.Postion = ep(end point
), obstacle.Location.X+obstacel. Width = x2)
```

```
Dim Y2 As Double = TestY2(a.Postion, b.Postion, obs.Location.X+
obs.Width)
```

```
Function TestY2(ByVal sp As Point, ByVal ep As Point, ByVal x2 As
Double) As Double
    Dim index, y As Double
    index = (x2 - sp.X) / (ep.X - sp.X)
    y = (sp.Y) + index * (ep.Y - sp.Y)
    Return y
End Function
```

Mathematical equation3 .as figures 5-22

$$\text{Index} = (\text{obstacle location.y} - \text{Start position.y}) / (\text{End position.y} - \text{Start position.y})$$

$$X1 = \text{Start position.x} + \text{Index} * (\text{End position.x} - \text{Start position.x})$$

Figure 5-22: Mathematical equation..3

```
'TestX1(agent.Postion=sp(start point ),bookcase.Postion = ep(end point
), obstacle.Location.Y= x2)
```

```
Dim X1 As Double = TestX1(a.Postion, b.Postion, obs.Location.Y)
```

```
Function TestX1(ByVal sp As Point, ByVal ep As Point, ByVal y1 As
Double) As Double
    Dim index, x As Double
    index = (y1 - sp.Y) / (ep.Y - sp.Y)
    x = (sp.X) + index * (ep.X - sp.X)
    Return x
End Function
```

Mathematical equation4 .as figures 5-23

$$\text{Index} = \frac{((\text{obstacle location.y} + \text{obstacle. Height}) - \text{Start position.y})}{(\text{End position.y} - \text{Start position.y})}$$
$$X2 = \text{Start position.x} + \text{Index} * (\text{End position.x} - \text{Start position.x})$$

Figure 5-23: Mathematical equation..4

```
'TestX2(agent.Postion=sp(start point ),bookcase.Postion = ep(end point
), obstacle.Location.Y+ obstacel. Height = x2)

Dim X2 As Double = TestX2(a.Postion, b.Postion, obs.Location.Y+
obs.Height)

Function TestX2(ByVal sp As Point, ByVal ep As Point, ByVal y2 As
Double) As Double
    Dim index, x As Double
    index = (y2 - sp.Y) / (ep.Y - sp.Y)
    x = (sp.X) + index * (ep.X - sp.X)
    Return x
End Function
```

Then after the values X1, X2, Y1 and Y2 are out for all function then work is compared to the values resulting from the dimensions of all the obstacles to examine as shown in Figure 5-24. Then plan route to reach the required goal, in the event of an intersection will be passing the obstacle, and if there is no intersection will continue to walk to the target.

'Function DosItCros call all function TestX1 ,TestX2,TestY1 and TestY and then comparison.

```
Function DosItCros(ByVal b As Object, ByVal a As Agent, ByVal obs As Object) As CorseDimention
Dim X1 As Double = TestX1(a.Postion, b.Postion, obs.Location.Y)
Dim X2 As Double = TestX2(a.Postion, b.Postion, obs.Location.Y+
obs.Height)
Dim Y1 As Double = TestY1(a.Postion, b.Postion, obs.Location.X)
Dim Y2 As Double = TestY2(a.Postion, b.Postion, obs.Location.X+
obs.Width)
    Dim y1_note As Double = obs.Location.Y

    Dim y2_note As Double = obs.Location.Y + obs.Height

    Dim x1_note As Double = obs.Location.X
    Dim x2_note As Double = obs.Location.X + obs.Width

    Dim rtValue As New CorseDimention
    If y1_note < Y1 And Y1 < y2_note Then
        rtValue.Direction = 1
        rtValue.Flag = True
        Return rtValue
    ElseIf y1_note < Y2 And Y2 < y2_note Then
        rtValue.Direction = 1
        rtValue.Flag = True
        Return rtValue
    ElseIf x1_note < X1 And X1 < x2_note Then
        rtValue.Direction = 1
        rtValue.Flag = True
        Return rtValue
    Else
        rtValue.Direction = 0
        rtValue.Flag = False
        Return rtValue
    End If
```

Figure 5-24: Function DosItCros

Table 5-25 illustrate Comparison between three techniques shortest path and best time and what the negative each technique.

Performance	When fill random	When used intelligent Code	When used Heuristic Search
Competency	48%	97%	95%
Negatives	The time it takes a very large	Must each object in environment has intelligent code and cannot change position of any object in environment the object is structured	None

Table 5-25: illustrates a Comparison between three techniques shortest path and best time

But the best technique is a heuristic search because it is not related to any constraint and when any change in environment can interact with change and give faster solution.

Chapter Six

Conclusion, Discussion & Future Work

Chapter Six

Conclusion, Discussion & Future Work

6.1 Overview

This chapter includes the conclusion of the thesis, the discussion about the directions in intelligent agent and suggestions for future work.

6.2 Conclusion

In this thesis, the following points can be concluded:

1. Algorithms and functions for how to build an intelligent agent interact with dynamic environment to perform tasks as a knowledge-based system has been designed and implemented for a specific environment.
2. The main objectives of this thesis were to build algorithms and functions that represent task specification performed by the intelligent agent. The intelligent agent analyzes the user order on the basis of the importance of the information needed to be gained from the knowledge base.
3. The mentioned algorithms were created to make the intelligent agent understand the dynamic environment and can perform task, attempt to understand how knowledge can be acquired from experience.

4. The thesis focused on how intelligent agents can perform task and interact with environment. The algorithms and functions are used for gaining knowledge from the state space of environment so as to build the task.
5. The intelligent agent can understand the environment in any position and can detect the subtasks needed to do and arrange in queue then execute many subtasks in addition to the ability to make a decision at a high level of thinking.

6.3 Discussion

The properties of the environment are very important especially in the design of intelligent agent program. The first step must always specify the task environment together with all characteristics as possible.

Task environments vary along several significant dimensions. They can be fully or partially observable, deterministic or stochastic, episodic or sequential, static or dynamic, discrete or continuous, and single-agent or multi-agent. Therefore, it is strongly advised to begin construction on those same rules, even if it takes a great deal of time, and research rebuilding advanced techniques for intelligent agent in the face of the functions of life and survival in difficulties.

6.4 Future Work

As a future work, it is highly recommended to start thinking about more complex environment, stochastic and multi agents in environment such as cooperative, competitive, and collaborative environment, where intelligent agents work together as a single unit (cell).

References

1. Balduccini, M. and Lanzarone, G.A.,(1997) *Autonomous semi-reactive agent design based on incremental inductive learning in logic programming*, in: van der Hoek W., Lesperance Y., Scherl R. (eds.), *Logical Approaches to Agent Modeling and Design*, Procs of the ESSLLI'97 Symposium, Aix-en-Provence(France),P.1-12.
2. Bekey, G. (2005). *Autonomous Robots: From Biological Inspiration to Implementation and Control*, p. 593 ,MIT Press.
3. Carnap ,R.(1950).*Logical Foundations of Probability* .University of Chicago press,Chicago.
4. Chandler, D. (7 Dec 2009), *Rethinking artificial intelligence*, MIT New of Massachusetts Institute of Technology University, USA.
<http://web.mit.edu/newsoffice/2009/ai-overview-1207.html>
accessed in 13/2/2011.

5. Dayan, P.E. & Hinton , G. E. (1993). *Feudal reinforcement Learning* , Proceeding Advances in Neural Information Processing Systems 5, *NIPS Conference Morgan Kaufmann Publishers Inc.* San Francisco, CA, USA.
6. DesJardins , M. (2001). *HTN planning algorithm* , CMSC 671 slides. www.cs.umbc.edu (On- Line), available: http://cw.felk.cvut.cz/lib/exe /fetch.php/courses/ a3m33ui /prednasky /files/ui-2010-p08-htn_planning.pdf, Sec. 12.2 p.1/25
7. Dorf, R.C. and Bishop , R.H. (1999) . *Modern Control Systems*. Addison-Wesley, Reading, Massachusetts.
8. Erol, K.U., Hendler, J.A.& Nau ,D.S.(1994) . *Semantics for hierarchical task network planning* , Technical Report CS-TR-32391, UMIACS-TR- 94-31, ISR-TR-95-9, Computer Science Department, University of Maryland.

9. Erol, K.U., Hendler J., & Nau D.S. (1996). *Complexity results for HTN planning*, *Annals of Mathematics and Artificial Intelligence* 18(1), p.69–93.
10. Feigenbaum, E.A. and McCorduck, P. (1983). *The Fifth Generation :Artificial Intelligence and Japan's Computer Challenge to the World*. Reading , MA:Addison-Weseley.
11. Firby , R. J. (1996). *Modularity issues in reactive planning* , In B. Drabble (Ed.) , *Proceedings of the Third International Conference on Artificial Intelligence Planning Systems (AIPS-96)*, AAAI Press, P. 78–85.
12. Georgeff , M.P. & Lansky , A.L. (1986). *Procedural knowledge* , *Proceedings of the IEEE, Special Issue on Knowledge Representation* 74(10),p. 1383–1398.

13. Green , C.O.(1969). *Application of theorem proving to problem solving* ,
In D. Walker and L. Norton (Eds.) , **Proceedings of the First
International Joint Conference on Artificial Intelligence
(IJCAI-69) William Kaufmann.**
14. Henzinger,T.A.and Sastry,S. (1998).*Hybrid systems : Computation
and control* .Springer-Verlag, Berlin.,(Eds.) **Technical.**
15. Hunt,E.&Waller , D. (1999). *Orientation and way finding: A review*, **ONR
technical report N00014-96-0380. Arlington, VA**, Office of Naval
Research.
16. Ian , F.A., Michael , Q.U. &Peter ,S.T. (2009). *A Task Specification
Language for Bootstrap Learning*, Retrieved October 10, 2010, from
Computer Science Department - The University of Texas at
Austin:[http://www.cs.utexas.edu/~pstone/Papers/bib2html-links/
AAAI symp09-fasel .pdf](http://www.cs.utexas.edu/~pstone/Papers/bib2html-links/AAAI symp09-fasel .pdf) , p.1169-1170

17. Ilghami, O., H. Munoz-Avila, D. Nau, and D. Aha (2005). *Learning Approximate preconditions for methods in hierarchical plans*. In L. D. Raedt and S. Wrobel (Eds.), *Proceedings of the Twenty-Second International Conference on Machine Learning (ICML-05)*, p. 337–344. ACM Press.
18. Kumar, P.R. and Varaiya, P. (1986). *Stochastic systems: Estimation, identification and adaptive control*. Prentice-Hall, Upper Saddle River, New Jersey.
19. LaValle, M.S. (2006). *Planning Under Sensing Uncertainty*. Available for downloading at <http://planning.cs.uiuc.edu/>, Published by Cambridge University Press
20. Michael, N. (2005), *Artificial intelligent : A Guide to Intelligent Systems/* (2nd ed.), Wesley.A, London
21. Niels, B.A. (2002) . *A model for Procedural Knowledge* ,PhD thesis , University of Nyenrode , Breukelen.

22. Nilson, N.J.(1984). *Shakey the robot . Technical Report 323, AI Center, SRI International.*

23. Owaied, H. (2010) . *Frame Model for Intelligent Robot as Knowledge-Based System, Special Issue of Ubiquitous Computing Security Systems, 15/1/2010, p.342.347*

24. Owaied ,H.H. & bu- Ar'a, M. (2007). *Functional model of human System as knowledge Base System, The 2007 International Conference on Information & Knowledge Engineering, P.158-161.*

25. Paterno',F.,Mancini ,C. & Meniconi ,S. (1997) . *ConcurTaskTrees: a diagrammatic notation for specifying task models* , CNUCE-C.N.R.Via S.Maria 36. 56126 Pisa Italy , Retrieved october 15, 2010, from <http://www.isys.ucl.ac.be/etudes/cours/iag3960/Paterno-interact97.-pdf>.

26. Penberthy, J. & Weld, D.(1992). *UCPOP: A sound, complete, partial order planner for ADL*, In B. Nebel, C. Rich, and W. Swartout (Eds.), ***Proceedings of the 3rd International Conference on Principles of Knowledge Representation and Reasoning (KR-92)*** , P. 103–114
27. RoobCallan (2003), *Artificial intelligent*, (2nd ed.), Published by PALGRAVE MACMILLAN.
28. Rowell, D. (2002). *State-Space Representation of LTI Systems*, Retrieved October 10, 2010, from University of Massachusetts Institute of Technology: <http://web.mit.edu/clifton/OldFiles/MacData/afs.course/2/2.14/www/Handouts/StateSpace.pdf>.
29. Russell and Norvig(2003),*Artificial Intelligence: A Modern Approach*, (2nd edition), Recommended (optional) second textbook: *Pattern Classification (2nd Edition)*, Duda, Hart and Stork.

30. Ruye Wang. (2009). *Define the neighbors of a pixel P located at (X,Y)*
, Retrieved March 20, 2011 , from:
http://fourier.eng.hmc.edu/e161/lectures/digital_image/node6.html
31. Sacerdoti, E.A.(1975). *The nonlinear nature of plans*, **In Proceedings of the Fourth International Joint Conference on Artificial Intelligence (IJCAI-75)**,P. 206–214.
32. Sergios, T. & Konstantinos,K. (2006), *Pattern Recognition*, (3nd ed.),
Academic Press.
33. Shapiro, J.A.(2003) . *Action Planning*, Retrieved October 20, 2010 ,
from: <http://www.civicus.org/new/media/Action%20Planning.pdf> ,
p.234-9876
34. Simon,H.A.(1983).Why should machines learn ? In Michalski et al.

35. Stentz, A. (1994). *Optimal and efficient path planning for partially-known environments*. In **Proceedings IEEE International Conference on Robotics & Automation**, p . 3310–3317.
36. Tate, A.U. (1977). *Generating project networks* .In R. Reddy (Ed.), ***Proceedings of the Fifth International Joint Conference on Artificial Intelligence (IJCAI-77)***, William Kaufmann, P. 888–893.
37. Thorndyke, P.W. & Goldin, S.E. (1983). *Spatial Learning and reasoning skill*. In H .L. Pick, Jr., & C.P. New York: Plenum, Acredolo (Eds.) **Spatial Orientation** P.195-217.
38. Weld, D. A. (1994). *An introduction to least commitment planning*. **AI Magazine**, Vol.15, No.4, P. 27–61.
39. Woolderidge , M.& Dunne, P. (2002) *.the computational complexity of agent verification*, Intelligent Agents VIII Lecture Notes in Computer Science, vol. 2333/2002, P.115-127.