# Visual Modeling Language for Agent Treasury Pharmaceutical

**Dr. Afaf Badie Al-Kaddo**
afafbad@yahoo.com
College of Education for Women - Computer Science Dept.

**Abstract**

The researches to discover useful ways to represent the agents and agent-based systems are continuous. Unified Modeling Language (UML) is a visual modeling language used for software and non software modeling systems. The aim of this paper is: using UML class diagram to design treasury pharmaceuticals agent and explain its internal action. The diagram explains the movement of the agent among other nodes to achieve user's requests (external) after it takes them. The paper shows that it is easy to model the practical systems by using agent UML when they are used in a complex environment.

**Keywords:** Agent Design, UML, internal action of agents, agent-oriented software engineering.

# اللغة البصرية لنمذجة وكيل مخازن دوائية

**د. عفاف بديع القدو**
**كلية التربية للبنات - قسم الحاسبات**

**المستخلص**

البحوث لاكتشاف طرق مفيدة لتمثيل الوكلاء والنظم القائمة على وكيل مستمرة. لغة النمذجة الموحدة (UML) هي لغة النمذجة البصرية المستخدمة للبرمجيات ونظم النمذجة غير البرمجيات. الهدف من هذه الورقة هو: استخدام الرسم التخطيطي UML لتصميم وكيل لمذخر أدوية وشرح الإجراءات الداخلية له. الرسم يفسر حركة الوكيل بين العقد الأخرى لتحقيق طلبات المستخدم (الخارجية) بعد أن يأخذها منه. تظهر الورقة أنه من السهل نمذجة النظم العملية باستخدام وكيل UML عندما يتم استخدامها في بيئة معقدة.

## 1. Introduction

Agents can be define as an active objects. The agents are different from objects in the structure of the internal state of an agent [1], therefore, agent oriented systems must be analyzed, designed and implemented differently. The agent differs from other tools in that: agent has autonomy feature (the ability to act without external interference)[2]. A software methodology is characterized by a modeling language and associated semantics and a software process. Intelligent agents are used widely in advanced software systems that contain different sources of information and data types. The agent methodology distinguishes between the *external and internal viewpoints.* The first one means decompose the system into agents that modeled according to their actions, responsibilities, and their external interactions. The second one means how agent's beliefs, goals, and plans are modeled[3]. There is no unifying programming language for agent, therefore, it is important to use visual modeling language like UML to implement agent systems. UML helps the developer of the software to generate code for a programming language therefore, this paper uses UML class diagrams to represent agent's internal action and relating it to the external action of an agent[3].

## 2. Background

Agent software methodologies that used UML for object-oriented software development is called Agent UML (AUML). This explains how agent based systems are having important impact reflects the commercial importance of technology. Because UML is a programming language independent, this makes it standard modeling language. The newest standard version of UML is now UML 2.0, as of November 2004 [4]. UML unified many approaches to the object-oriented software therefore it is trust and strong modeling language. UML includes many types of diagrams when used them they simplified understanding the application under developed[4].

## 3. Agent UML Class Diagram

UML is a visual graphical language used for modeling object-oriented systems and can be considered as the successor of object oriented analysis and design. The class diagram shows the static structures of the system (data relate to each other) and implementation classes that are important to programmers[5]. They are applied in many stages of the project. Class diagrams explain classes and interfaces with their attributes and operations, associations, generalization (inheritance) and relationships among them. In UML 2.0 visibility, names and types are represented as graphical interface notation [4], [6]. Class diagram in UML represents the relationships between classes and define attributes and operations for these classes[7]. Figure 1 explains relationships in UML. Agents can be defined as a set of classes or can use classes. Agent UML class diagram includes both agents and classes and is expressed through mental components like beliefs, goals, actions and plans [8]see Figure 2. Goals and beliefs are expressed as attributes recognized by type, name and initial value. The agent itself controls the execution of action(operation). Plans are associated with goals and are executed as a sequence of actions (operation) by agent to achieve the goal. Agent model can be expressed by using class names, inheritance and adding name, type, role, capabilities and constrains, either directly or through associations[7] see Figure 3.

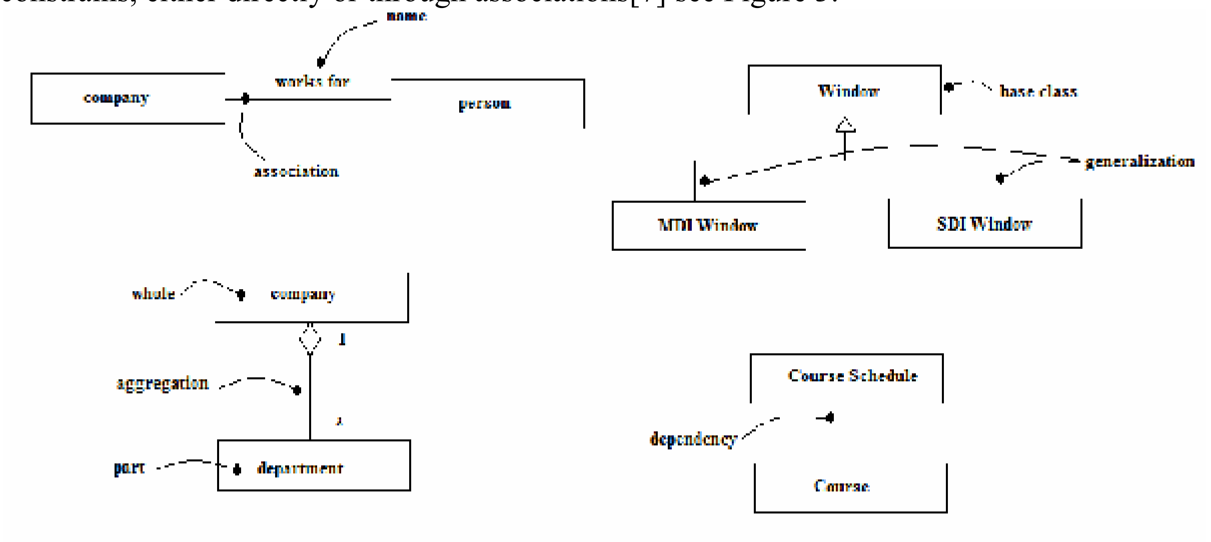
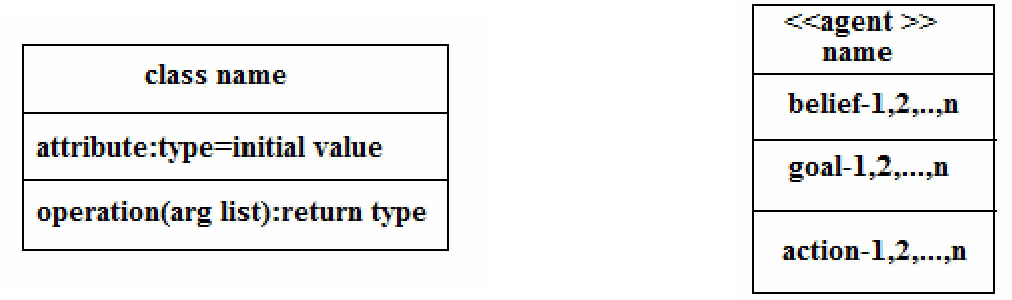


**Fig. 1: Relationships in UML**



**(a)** **(b)**

**Fig. 2: (a): Object Oriented Class Diagram**
**(b): UML Agent Class Diagram**

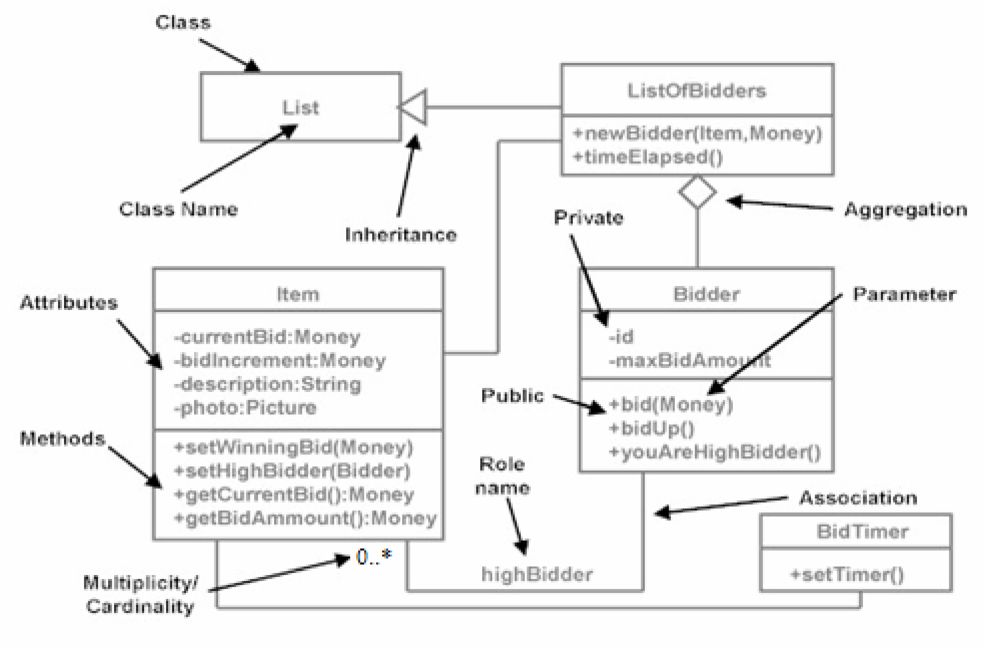


**Fig. 3: An Example of UML Class Diagram Describes Classes and Interfaces**

## 4. The Proposed System

For simplicity we will take three network nodes to represent Treasury Pharmaceuticals (TP for abbreviation) you can take more: user node, node1 for TP1and node2 for TP2 figure 4. In node1 and node2 TP1 and TP2 are exist which they are responsible for providing medicine menu. At user node there is an agent responsible for searching about the specific item (medicine). The user agent takes the order from the user, moves towards node1 searches for the order (medicine), if the order is found it returns to user node and the communication is canceled, if not, it will go to node2. The same procedure will be repeated in node2 and other nodes if there exist more than three nodes and finally the user agent returns to the user node and informs the user if it has found the specified item.

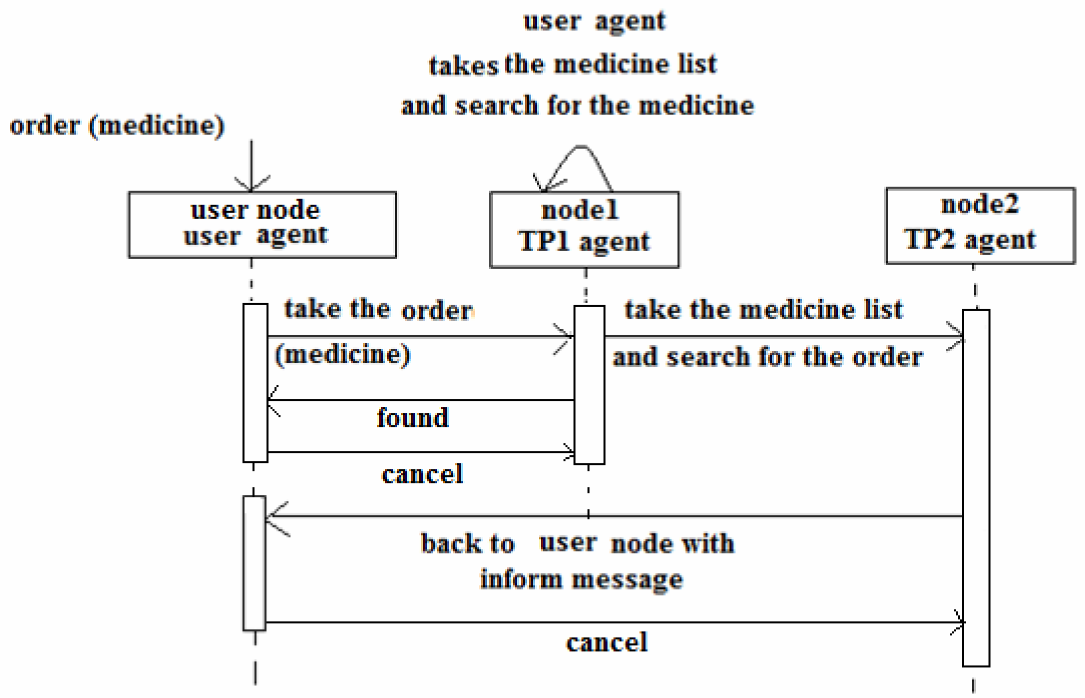


**Fig. 4: UML Agents Interaction Treasury Pharmaceuticals**

## 5. Treasury Pharmaceuticals' Class Diagram

Agent oriented programming development can be extended using UML class diagram. The flow of control from action to action is modeled by activity diagram which describes the dynamic nature of the system, workflow and internal operations. An activity represents an operation on any class in the system that results in a change in the state of the system. Static and dynamic states are designed by using UML class and activity diagrams. Figure 5 shows the Activity diagram of the treasury pharmaceuticals system. Figures 6,7,8 and 9 show the class diagrams.

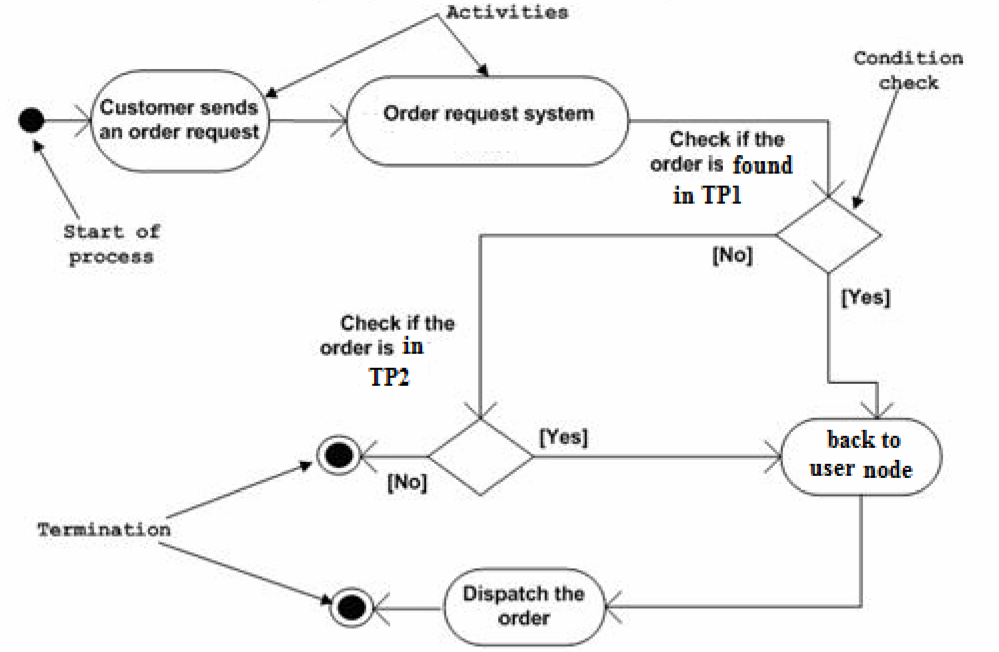
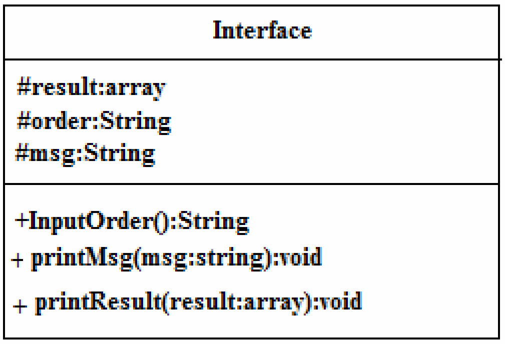


**Fig. 5: Activity Diagram of Treasury Pharmaceuticals System**

The Treasury Pharmaceuticals' class diagram consists of four classes. Each class has three sections : class name (first section), attributes (second section) and methods (third section) as follows:

<u>**Note:**</u> + means public and # means protected.

**Interface class:** has three public methods for reading the order *InputOrder()* and printing message *printMsg(msg:String)* and result for user *printResult(result:array)* consequently(i.e. communication between the user and the system) as shown in Figure 6.



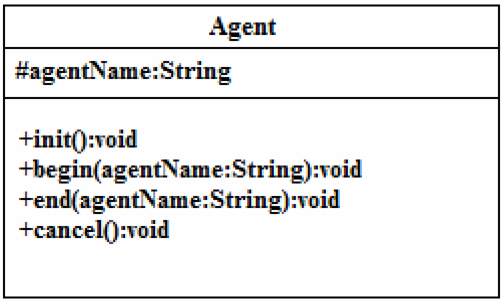
**Interface**

#result:array
#order:String
#msg:String

+InputOrder():String
+ printMsg(msg:string):void
+ printResult(result:array):void

**Fig. 6: Interface class**

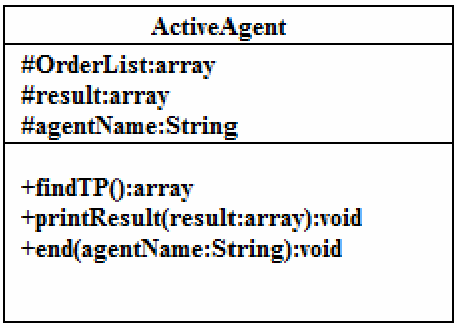
**Agent class:** has four public methods for initializing member variables *init()*, two methods for beginning *begin(agentName:String)* we have to name the agent because we have more than one agents and ending the specific user agent *end(agentName:String)* and ending the communication consequently *cancel()* i.e. after the agent finished its task (found or not found the item) it has to be ended and then cancel the communication as shown in Figure 7.



Agent
#agentName:String
+init():void
+begin(agentName:String):void
+end(agentName:String):void
+cancel():void

**Fig. 7: Agent class**

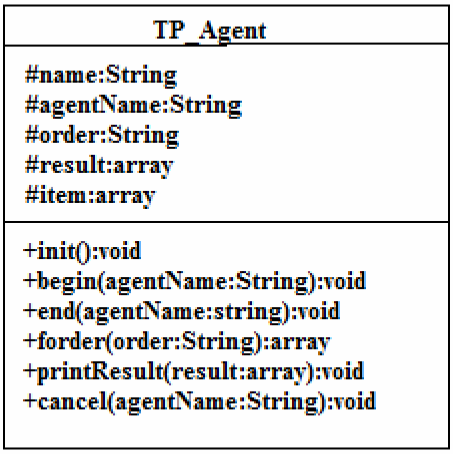
**ActiveAgent class:** has three public methods for finding the TP node *findTP()*, (i.e. the node that contain the specific item) printing result *printResult(result)* print which node contains the specific item and canceling the active agent *cancel()* i.e. after the agent finished its task (found or not found the item) it has to be ended and then cancel the communication consequently as shown in Figure 8.



ActiveAgent
#OrderList:array
#result:array
#agentName:String
+findTP():array
+printResult(result:array):void
+end(agentName:String):void

**Fig. 8: ActiveAgent Class**

**TP_Agent class:** has six public methods for initializing member variables, two methods for beginning and ending TP agent, searching for the order, printing the result and canceling user agent consequently (explained previously) as shown in Figure 9.



TP_Agent
#name:String
#agentName:String
#order:String
#result:array
#item:array
+init():void
+begin(agentName:String):void
+end(agentName:string):void
+forder(order:String):array
+printResult(result:array):void
+cancel(agentName:String):void

**Fig. 9: TP_Agent Class**

## 6. Conclusion and Ongoing Work

Modeling language extends UML to exploit agent's features like autonomy, cooperation and interaction. Class diagrams allow describing the static structure and relationship of the system and interaction among the elements. The structure diagram contains elements and relationships which are mapped into java classes. The work with agent system is easier as object oriented systems by looking to the system as a set of cooperated agents. Using agents in design will reduce the running time and this will affect the performance of the system. As a future work, multi agent systems are recommended and improve UML diagram to generate code for dynamic diagrams. Star UML package is used to generate a code.

## 7. References

[1] Burmeister, B., ed., "*Models and Methodology for Agent-Oriented Analysis and Design",* 1996.

[2] Bernhard Bauer1 and James Odell, *"Agent UML: A Formalism for Specifying Multiagent Interaction",* Institute of Computer Science, University of Augsburg, Germany, 2005.

[3] Bauer, B.; Müller, J. P.; Odell, J.: "*An Extension of UML by Protocols for Multiagent Interaction*", IEEE Computer Society, 2000.

[4] OMG, "*Unified Modeling Language: version 2.0 (UML 2.0)", Final Adopted Specification*, 2003, http://www.uml.org/#UML2.0

[5] Ivar Jacobson, Grady Booch, James Rumbaugh: *"The Unified Software Development Process*", Addison Wesley, 1998.

[6] Miao Kang, Lan Wang, and Kenji Taguchi, "*Modelling Mobile Agent Applications in UML 2.0 Activity Diagrams*", 2004, http://www.auml.org/auml/supplements/UML2-AD.pdf

[7] Huget, Marc-Philippe, *" agent UML Class Diagrams Revisited"*, Agent ART Group, University of Liverpool, United Kingdom, 2009.

[8] Al-Kholy Momtaz, Ahmed Khalifa and El-saied Mohamed*," A Systematic Approach for Agent Design Based on UML"*, Systems and Computer Engineering Department, Al-Azhar University, International Arab Journal of e-Technology, Vol. 2, No. 2, Egypt, 2011.