©2010 International Journal of Computer Applications (0975 – 8887)
Volume 1 – No. 7

A Survey of Software Agent and Ontology

Gopal Sakarkar
S. G. B. Amravati University,
Amravati
GHRIIT, MCA Dept, MIDC,
Hinga-Wadi Road, Nagpur

Sachin Upadhye
R.T.M. Nagpur University,
Nagpur.
Tulsiramji - Gaikwad Patil College of
Engg. And Tech.
Mohgaon, Nagpur

ABSTRACT

Software agent is the one of the most recent contribution in the field of Information Technology. The field of software agents is a broad and rapidly developing area of research, which encompasses a diverse range of topics and interests.
In order to study the various methodologies for agent design, implementation, commercial use of it, a sample survey is required.

This paper gives an overview of recent research on the software agents, agent communication languages (ACL), the different ontology use for software agent, also the summary of ACL and different tool kits for developing a software agent.

General Terms

The general terms of the paper is following
Documentation, Languages, Theory, Verification.

Keywords

Software Agent, Agent Communication Languages, Ontology.

1.INTRODUCTION

The use of Internet has accelerated at an unprecedented space. There is a vast amount of information available on the WEB that is heterogeneous and distributed .So it is practically infeasible for any user to combine all of possible information source to obtain an optimized and satisfactory information .The researcher is addressing this desire by developing software agent that act on behalf of users. The technology of software agents can be an interesting tool for the creation of new models for complex software systems. Agent Technology is widely used to help users to achieve various tasks in diverse domain such as network management, air-traffic control, telecommunication, and electronic commerce [1]. Software agents are basically designed to co-operate (either with others or with humans) in a seemingly intelligent way.

The paper is organized as follows.

Next sections focus related work of software agent. Section 2.1 focuses on various definitions of software agent. Section 2.2 explains the various characteristics. Agent communication languages explain in the section 2.3. Section 3 focus on related work of different ontology, finally section 4 concludes and gives some future works.

2.RELEATED WORK OF SOFTWARE AGENT

An agent, also called a software agent or an intelligent agent, is a piece of autonomous software, the words intelligent and agent describes some of its characteristic features. Intelligent is used because the software can have certain types of behavior (*"*Intelligent behavior is the selection of actions based on knowledge*"*), and the term *agent* tells something about the purpose of the software [2]. For example the animated paperclip agent in Microsoft Office, Computer viruses (destructive agents), artificial players or actors in computer games and simulations (e.g. Quake), trading and negotiation agents (e.g. the auction agent at EBay), etc. The performance of Software Agent is measure evaluates the environment sequence Environments are categorized along several dimensions: Observable, Deterministic, Episodic, Static, Discrete, and Single-agent.

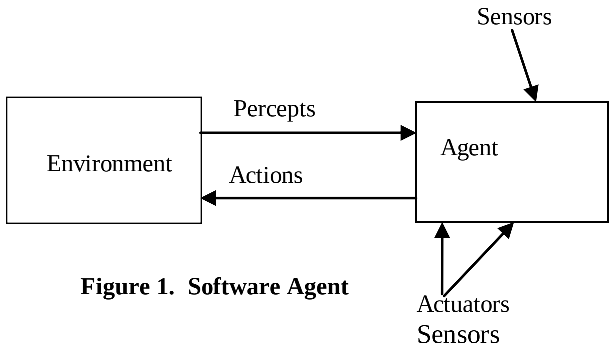

Figure 1. Software Agent

2.1 Definitions

There are many definitions of agents, but many people agree that: "an autonomous software agent is a component that interacts with its environment and with other agents on a user's behalf." Some definitions emphasize one or another characteristic such as mobility, collaboration, intelligence or flexible user interaction. [3]. various authors have proposed different definitions of agents, these commonly include concepts such as

- *Persistence* - code is not executed on demand but runs continuously and decides for itself when it should perform some activity.

- *Autonomy* - agents have capabilities of task selection, prioritization, goal-directed behavior, decision-making without human intervention.

- *Social ability* - agents are able to engage other components through some sort of communication and coordination; they may collaborate on a task.

- *Reactivity* - agents perceive the context in which they operate and react to it appropriate

.

2.2 Characteristics of Software Agent

An agent system is essentially a component system exhibiting several of the characteristics. There is six orthogonal characteristics work together to make agent-oriented systems more flexible and robust to Change.
1. *Adaptability* - The degree to which an agent's behavior may be changed after it has been deployed.
2. *Autonomy*- The degree to which an agent is responsible for its own thread of control and can pursue its own goal largely independent of messages sent from other agents.
3. *Collaboration* - The degree to which agents communicates and works cooperatively with other agents to form multi-agent systems working together on some task.
4. *Knowledgeable* - The degree to which an agent is capable of reasoning about its goals and knowledge.
5. *Mobility - The* ability for an agent to move from one executing context to another, either by moving the agent's code and starting the agent afresh, or by serializing code and state, allowing the agent to continue execution in a new context, retaining its state to continue its work.
6. *Persistence* - The degree to which the infrastructure enables agents to retain knowledge and state over extended periods of time, including robustness in the face of possible run-time failures.[4]

2.3 Software Agent Communication Language (ACL)

When programming with Top-down programming methods (Monolithic Programming) programmers have full control of programs they are writing. It source code is generally not reusable or modular, and each program is a separate entity used only for certain purpose [5]. Modular Paradigm brings in the procedural approach and reusable components of source code. Furthermore, Object-Oriented Programming introduces the state of the program as internal states of individual objects created by programmer. In addition to individual states objects also maintain own actions defined as methods. Object-Oriented Paradigm has been associated to the development of software agents since there are many intuitive similarities between objects and software agents. Author also brings out interesting difference between objects and software agents, that is, Software agents are also autonomous: they are empowered to act, in other words, they take initiative to achieve their goals. Objects in traditional OOP paradigm are considered Passive since their actions have to be invoked by caller. Second, Author proposes that Agents may respond to interaction any way they choose: they can accept or refuse the proposed action. In other words, software agents can behave unexpectedly. Objects in OOP tend towards more predictable actions. Therefore, Object-Oriented Paradigm does not fully respond to the agent programming needs from every point of view, although most of the agent actions can still be implemented with Object-Oriented programming Languages [6].

Shoham proposed a new programming paradigm called Agent-Oriented Programming (AOP) in the early 1990s. AOP is a specialization of Object-Oriented Programming paradigm and it allows programming of agents in terms of their mental states. Agent programs control agents and communications primitives such as informing; requesting and offering can be used in interaction [6].

Table 1. The Relation between OOP and AOP

Feature	OOP	AOP
Basic Unit	Object	Agent
Parameters Defining State of Basic Unit	Unconstrained	Beliefs, Commitments, Capabilities, etc
Process of Computation	Message Passing and response methods	Message Passing and response methods
Type of Messages	Unconstrained	Inform, request, offer, promise, decline, etc
Constraints on Methods	None	Honesty, Consistency

Component of Agent Communication Language (ACL)

Dogac and Cingil state that in order to collaborate with others, agents are required to:

- Discover the existence, network addresses, capabilities and/or roles of other agents;
- Communicate with other agents through an agent-independent, that is, a standard agent communication language.

Name servers and facilitators are provided by Multi-Agent Systems to support agent discovery. Agents register their addresses to a name server and their features and abilities to a facilitator. Agents can then use those name servers and facilitators as a reference to find out abilities and addresses of other agents [6]. After the agents have discovered each other they need to communicate in order to achieve their goals. Communication can be divided into two fundamental parts. First, agents need to agree on a common agent communication language (ACL) that provides the basis for stating intentions to other party. Second, mere common language is not enough but agents must also have common vocabularies for representing shared domain concepts and application-dependent content [12]. This includes both a shared ontology and the content that is defined by a Content Interchange Format (CIF). Therefore, agent communication languages basically consist of these three layers that are shown in Figure 2.

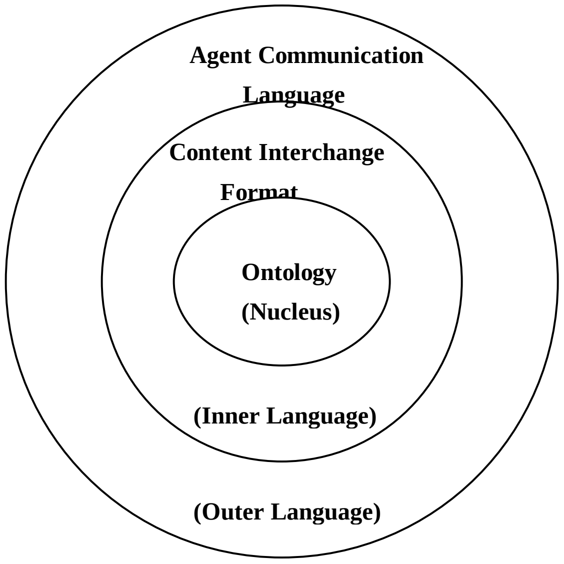

Figure 2. Component of ACL

ACL (Outer Language): -

Speech act theory is the basis of most popular agent communication languages. Speech act theory was originally developed by linguists in an attempt to understand how humans use language in everyday situations. Two most successful agent communication languages so far have been Knowledge Query and Manipulation Language (KQML) and FIPA-ACL.

CIF (Inner Language): -

This content part of the message defines the actual Information about the matter agents are trying to communicate. The inner language, representing the content part of the message, allows an agent to express its actual application-dependent content to other agents. Agents must understand each other in this content language level to be able to successfully interact with each other. Two content languages among the most popular ones are Knowledge Interchange Format (KIF) and FIPA Semantic Language (FIPA SL). The syntax of both these languages is inherited from Lisp programming language.

Ontology (Nucleus): -

Even if agents speak the same language, they require some common understanding of the meaning of the message content. This is provided by specialized knowledge component called ontology that specifies the objects, concepts and relationships in a given domain. According to Author, ontology are usually built using schema definition or a knowledge presentation language Author also state that creating and expressing ontologies of any size is difficult and time-consuming work and many tools have been developed for analyzing and developing ontologies. One of these tools is called Ontolingua.

2.4 Software Agent Toolkits

Agent toolkits are defined as sets of components from which to build agent systems and sets of tools to help operate agent systems. Agent implementers don't have to start building agents from scratch since numerous platforms and toolkits have been introduced for building agent [6].

Table 2. Agent Toolkits Summarized

Product Name (Company)	Mobile / Stationary	Language	Standards	Example Application
JatLite (Stanford Uni.)	Stationary	Java	KQML	Design Decision tracking, Constraint mgt, Enterprise control
ZEUS (BT UK)	Stationary	Java	FIPA KQML	Supply Chain mgt, Service provisioning, network resource mgt
FIPA-OS (Nortel Network)	Stationary	Java	FIPA	Personalized Service, VPN, VHE, Meeting scheduler etc.
JADE (CSELT)	Stationary	Java	FIPA	Travel Assistant, Audio-visual entertainmen t

2.5 Type of Software Agent

Personal agents: It interacts directly with a user, presenting some "personality" or "character", monitoring and adapting to the user's activities, learning the user's style and preferences, and automating or simplifying certain rote tasks. Microsoft's Agents "Bob" or "Paper Clip" is simple examples built using this technology.

Mobile agents: It is sent to remote sites to collect information or perform actions and then return with results. "Touring" agents visit sites to aggregate and analyze data, or perform local control. Such data intensive analysis is often better performed at the source of the data rather than shipping raw data; examples include network management agents and Internet spiders.

Collaborative agents: It communicates and interacts in groups, representing users, organizations and services. Multiple agents exchange messages to negotiate or share information. Examples include online auctions, planning, negotiation, logistics and supply chain and telecom service provisioning.

3. RELEATED WORK OF ONTOLOGY

Ontologies are becoming increasingly important because they provide the critical semantic foundation for many rapidly expanding technologies such as software agents, e-commerce and knowledge management. Ontology is term borrowed from philosophy meaning "systematic explanation of existence". Ontology is similar to a dictionary or glossary but with a large and detailed structure, that allows machines to process its contents. Nicola Guarino [16] defines ontology as" These are formal representations of domain knowledge in the form of terms with semantic relations". The propose suitable clarification of ontology is as below.

1. Ontology as a philosophical discipline
2. Ontology as a an informal conceptual system
3. Ontology as a formal semantic account
4. Ontology as a specification of a "conceptualization"
5. Ontology as a representation of a conceptual system via a logical theory
   5.1 characterized by specific formal properties
   5.2 characterized only by its specific purposes
6. Ontology as the vocabulary used by a logical theory
7. Ontology as a (meta-level) specification of a logical theory.

Ontologies can capture both the structure and semantics of information environments. An ontology based search engine can handle both simple keyword-based queries as well as complex queries on structured data. [17]. Ontologies can organize keywords as well as database concepts by capturing the semantic relationships among the keywords or among the tables and fields in a database. On the one hand ontologies provide the knowledge representation; on the other hand agents perform the actions. Ontology is a vocabulary of entities, classes, properties, functions and their relationships. Ontologies are meant to provide an understanding of the static domain knowledge that facilitates knowledge sharing and reuse. [18]. The ontology basically is used for logical description of all possible data sources that can be used to deliver input data to the queries.

3.1 Definitions

Defined for a given objective, ontology expresses a point of view shared by a community. Ontology is represented in a language (explicit ontology) whose theory (semantics) guarantees the properties of the ontology in terms of consensus, coherence, sharing and reuse" [18]. The term ontology is used to refer to "an explicit specification of a conceptualization [of a domain] is mentioned by Tom Gruber which we are already familiar with for quite sometimes. In other words, ontology refers to a formalization of the knowledge in the domain [19]. Ontology is the concept which is separately identified by domain users, and used in a self contained way to communicate information. The definition of "Ontology is a conceptualization of a domain to which one or several vocabularies can be associated and which participates to the meaning of terms. There is no single correct ontology for any domain. Ontology design is a creative process and no two ontologies designed by different people would be the same.

3.2 Need of Ontology

In recent years, a number of sub fields of artificial intelligence have been aiming to increase the ability of their systems to interact with humans and other external agents by developing and sharing Ontologies. Ontologies mean formally specified models of bodies of knowledge defining the concepts used to describe a domain and the relationships that hold between them. Typically, ontology identifies classes of objects that are important in a domain, and organizes these classes in a subclass-hierarchy. Each class is characterized by properties shared by all elements in that class. Important relations between classes or between the elements of these classes are also part of ontology [20]. Recently, ontologies have been proposed as an enabling technology for the Semantic Web. Most information on the Web is written in natural language, intended for humans to read. The Semantic Web proposes to make this information also understandable by computers. To illustrate the difference, finding information on the Web usually involves conducting a keyword search and then filtering and analyzing a list of retrieved resources. On the Semantic Web, software agents would be able to deal with more complex queries, filtering and extracting meaning on behalf of the user, and returning a direct response.

For developing a software agent the following should include, but not limited to,

Name of the ontology

It is Necessary to give the name of the ontology that should appear as the value for the: ontology parameter of the ACL message. If the terms are appropriately categorized, we need to give names for ontologies corresponding to those categories.

Framework to express the ontology

Framework is needed to express the ontology. This may be a meta-ontology in one of first order logic languages. In order to make the standard language-independent, we can take such an approach for example to provide models in graphs and serializations of the models into SL, KIF, XML, etc

Content of the ontology

It is necessary to standardize the content of the ontology, which is the essential part of this proposal. We need the terms for the predicates to describe the agents and possibly the values or the formats of the values for those predicates.

3.3 Languages for Ontology

Table 3. Language use for Ontology

Name of Language	Feature
RDF Schema	1. A general-purpose language for representing information in the Web. 2. A schema defines the properties of the resource (e.g., title, author, subject, size, color, etc.) and the kind of resources being described (e.g., books, Web pages, people, companies, etc.).
OIL	1. Provides modeling primitives used in frame based and Description Logic oriented ontologies, coming along with a simple and clean semantics. 2. A syntax definition using web standards such as RDF(S) and XML(S).
DAML+OIL	1. Exploits existing de-facto Web standards (XML and RDF), adding ontological primitives of object oriented and frame-based systems. 2. Designed to describe the structure of a domain. 3. The structure of the domain being described in terms of classes and properties, and the set of axioms that assert characteristics of these classes and properties.
OWL	1. Mainly based on OIL and DAML+OIL 2. Three sub languages called: a. OWL-Lite b. OWL DL. c. OWL Full. 3. Includes an abstract syntax, which provides a higher level and less cumbersome way of writing ontologies.
SHOE (Simple HTML Ontology Extensions)	1. Extension of HTML with tags for incorporating semantic knowledge into documents. 2. Based on an ISA hierarchy, it is very simple but less expressive than some other specifications

3.4 Tools of Ontology

Ontologies becoming increasingly important, several Tools for building ontologies were developed. Many existing ontology tools provide an integrated environment to browse and edit ontologies as well as inconsistency checking facilities.

OilEd: - OilEd was developed at the University of Manchester for easy development of OIL languages (including DAML+OIL). It is a simple tool enabling users to create ontologies and check them for consistency, and is available as an Open Source project under the GPL licence. It was developed in the context of the European IST OntoKnowledge project.

Protégé 2000: - It was developed at Stanford University as an Open Source editing environment where ontologies can be constructed through a graphical user interface. It has been developed using a plug-in architecture, where new services can be added easily to the environment, and is perhaps the most widely used ontology tool. It can handle ontologies in XML, RDF (S), XML Schema, DAML+OIL and OWL.

Ontolingua: - It is an ontology-editing environment, which enables ontologies to be constructed collaboratively by distributed groups. It includes a library of reusable ontologies, which means that new ontologies can be created quickly from existing ones.

OK Station (Ontological Knowledge Station): - The OK Station is a graphical and interactive ontology-design environment. Initially developed at the University of Savoie, it is now a commercial product.

WedODE2: - WedODE2 provide an integrated environment to browse and edit ontologies as well as inconsistency checking facilities

OntoEdit 4: - It was developed by the University of Karlsruhe and consists of a repository of ontologies, an inference and query engine and various translators. It supports the development and maintenance of ontologies by using graphical means. It is available in free and professional versions.

ReTAX++: - ReTAX++ graphical tool. It proposes a graph-based approach implemented in ReTAX++ to help knowledge engineers browse ontologies and resolve the inconsistencies.

LinkFactory:- It is a formal ontology management system, designed to build and manage very large and complex language-independent formal ontologies. It consists of two components: the LinkFactory Server, which stores data in a relational database, and the LinkFactory Workbench, which allows the user to browse and model several ontologies and align them. Both components are developed in Java. [23-26]

3.5 Use of Ontology

Ontology is term, which is not use for software agent only because it is also use in various applications as mention below

1. Retrieving the appropriate information from documents by providing a structure to annotate the contents of a document with semantic information.
2. Integrating the information from various sources by providing a structure for its organization and facilitating the exchange of data, knowledge and models.
3. Ensuring consistency and correctness by formulating constraints on the content of information.

4. Creating libraries of interchangeable and reusable models.
5. Supporting inference to derive additional knowledge from a set of facts.[21,22]

4 CONCLUSIONS

This paper describes survey of software agent by studying the characteristic software agent, component of Agent Communication Languages and type of software agent. Also focus on the number of different tools available for developing a software agent.

Also explain ontology, need of ontology for developing a software agent with different ontology language and tools.

The current research aims to enhance the agent oriented programming from objects to Autonomous Agents. Some of the issues that we plane to focus on in the future include, making it easier to define and implement different agent system directly in term of their capability.

5 REFERENCES

[1] Gopal Sakarkar, Dr.V.M. Thakare, “A Survey of Various Software Agent Architectures and Agent Communication Languages”, first International Conf.On MNGSA’08, Dec.2008 CSIR and IACSIT, India

[2] Amund Tveit, “A survey of Agent-Oriented Software Engineering”, First NTNU CSGSC, May 2001,www.csgsc.org

[3] Martin L. Griss, “Software Agents as Next Generation Software Components”, 2001

[4] Gopal Sakarkar and N.M. Shelke “A New classification Scheme for Autonomous Software Agent”, IAMA’09, 2009, IEEE Int. Conf. , India

[5] Tomi Myllylä “Software Agents: A Literature Survey”, Mat-2.108 Independent Research Project in Applied Mathematics, June 2005, tomi.myllyla@hut.fi

[6] Odell. “Objects and Agents Compared”, Journal of Object Technology, 2002,vol.1, no.1, pp. 41-53.

[7] Gopal Sakarkar, Dr.V.M. Thakare “Revolutionary Architecture for Autonomous Software Agents” International Advance Computing Conference (IACC’09), IEEE March 6-7, 2009, India

[8] A.Chella,S.Gaglio, G. Sajeva, F.Torterola “An Architecture for Autonomous Agents Integrating Symbolic and Behavioral Processing ”,1997,IEEE pp.-45-50

[9] Dogac, Cingil. B2B e-Commerce Technology: Frameworks, Standards and Emerging Issues, Addison Wesley,2003

[10] Dominic Greenwood, Margaret Lyell” The IEEE FIPA Approach to Integrating Software Agents and Web Services ” AAMAS’07 May 14–18 2007, Honolulu, Hawai’i, USA. 2007 IFAAMAS.

[11] Sheng Q. Z., Benatallah B.: ContextUML: A UML-Based Modeling Language for Model-Driven Development of Context-Aware Web Services, In Proc. of the (ICMB’05), July 2005, pp. 206-212

[12] Pitkäranta. “Software Agents in Semantic Web Environment”, Master´s Thesis, Helsinki University of Technology, 2004

[13] Amol Dattatraya Mali” On the Behavior-Based Architectures of Autonomous Agency ”,IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS—PART C: APPLICATIONS AND REVIEWS, VOL. 32, NO. 3, AUG 2002

[14] Gopal Sakarkar,Dr.V.M. Thakare“ A Road Map of Autonomous Software Agent Architectures”, Proceedings of the 3rd National Conference; INDIACom-2009,CSI, Feb.26-27, 2009, New Delhi

[15] Hemme-Unger K., Flor T., Vögler G.: MDA Development Approach for Pervasive Computing, OOPSLA’03, Anaheim, CA, USA, 18th ACM SIGPLAN conf. on OOP, systems, languages, and app., Oct. 26-30, 2003

[16] Nicola Guarino , Pierdaniele Giaretta,” Ontologies and Knowledge Bases“,guarino@ladseb.pd.cnr.it

[17] Kuhanandha Mahalingam and Michael N. Huhns, “An Ontology Tool for Distributed Information Environments”

[18] Fensel. *Ontologies: A Silver Bullet for Knowledge Management and Electronic Commerce*. Heidelberg, Germany, 2001.

[19] Dr. Waralak V. Siricharoen, “Ontologies and Object models in Object Oriented Software Engineering”,2003

[20] Hongsoon Yim, Kyeh yun Cho, Jongwoo Kim , and Sungjoo Park,”Architecture Centric Object Oriented Design Method For Multiagent Systems “

[21] H. Alani, S. Kim, D. Millard, M. Weal, W. Hall, P. Lewis, and N. Shadbot. Automatic ontologybased knowledge extraction from web documents. IEEE Intelligent Systems, 18(1):14–21, 2003.

[22] AgentCities.NET. IST project IST-2000-28384 Agentcities, 2000. http://www.agenticities.net/.