Intelligent Software Agent Technology: An Overview

G. S. Bhamra
DCSE, M. M. University,
Mullana-133203,Haryana,India

A. K. Verma
DCSE,Thapar University,
Patiala-147004, Punjab, India

R. B. Patel
DCSE, G.B. Pant Engineering College, Ghurdauri-246194, Uttarakhand, India

ABSTRACT
Mobile Agent is an autonomously transportable code migrating itself from one node to another in a heterogeneous network without losing its operability. They have become commercially practicable with recent technologies and have the potential for revolutionizing distributed and network applications more recently in wireless sensor networks and bioinformatics. The main aim of this study is to motivate the researchers into the field of intelligent software agent technology by providing an overview and updated comparison of the current mobile agent platforms.

General Terms
Distributed Computing, Agent Technology.

Keywords
Intelligent Agent, Mobile Agent, Mobile Agent Platforms.

1. INTRODUCTION
Intelligent software agent technology is an interdisciplinary technology inherited from different research disciplines such as, Distributed Computing (DC), Distributed Artificial Intelligence (DAI), advanced knowledge base systems, and human computer interaction. Intelligent agents are the autonomous software objects, which have access to geographically distributed and heterogeneous information resources. The motivating idea of this technology is the development and efficient utilization of such agents as they simplify the complexities of DC [1],[2],[3]. Agent research stems from the work in DAI conducted in the 1970's. Carl Hewitt proposed an Actor system [4] where each Actor had an explicit internal state and had the capability to respond to the messages of other Actors. The last decade has seen a huge expansion of agent systems to solve practical problems in various domains. The next generation of distributed applications demand a large number of interacting components in the form of services or intelligent agents[5].

2. TAXONOMIES AND CHARACTERISTICS
A number of classifications of agents and intelligence criteria have been proposed in the literature. Software Agents with different attributes are being designed depending upon the goals to be achieved and responsibilities to be carried out. Gilbert et al. (1995)[6] described intelligent agents in terms of the three dimensions of agency, intelligence, and mobility. Nwana (1996)[1] identified three primary attributes, autonomy, cooperation, and learning and classified the intelligent agents as collaborative, collaborative learning, interface, and smart. Based upon the mobility, reactivity, role played, hybrid philosophies and secondary attributes, such as versatility, kindness, reality, trustworthiness, temporal continuity, ability to fail gracefully, and mentalistic and emotional qualities, agents can be classified as mobile agents, information/internet agents, reactive agents, and hybrid agents [7],[8],[9]. Several definitions for the agents have been proposed suggesting a set of defining characteristics that every agent must possess. These capabilities are listed below:-

- *Autonomous:* Autonomy is the ability of operating without the intervention of users. The degree of autonomy and authority vested in an agent is called its agency. It can be measured qualitatively by the nature of the communication between the agent and other entities involved. At a minimum, an agent must run a-synchronously. The degree of agency is enhanced if an agent represents a user/human in some way. Human behavior and practical reasoning in agents can be incorporated using Belief-Desire-Intension (BDI) model of agency in terms of 'mentalistic' notions [10]. The origins of the model lie in the theory of human practical reasoning developed by the philosopher Michael Bratman [11].
- *Reactive or Deliberative:* the ability of perceiving any change in the environment and suitably reacting to it.
- *Adaptive:* the ability of sensing the environment and reconfiguring in response dynamically. This can be achieved through the choice of alternative problem solving rules/strategies or algorithms.
- *Pro-activeness:* the ability of exhibiting goal-directed behavior in order to satisfy a design objective.
- *Intelligence:* the ability to accept the user's statement of goals and carry out the task delegated to it. It is the degree of reasoning and learned behavior.
- *Temporal continuity:* the ability of persistence of identity and state over long periods of time.
- *Social, Cooperative and Collaborative:* the ability of interacting and negotiating with other agents in groups and/or human users to achieve a common goal via some kind of agent communication language. They cooperate, negotiate and collaborate with other agents to make a smart system.
- *Flexibility:* the ability of exhibiting reactivity, pro-activeness and social ability simultaneously.
- *Mobility/Migration/Navigation:* the ability of migrating from a node to node in a heterogeneous network. Mobility can be of weak and strong in nature. Weak mobility moves an agent who is currently idle (not running) to another machine, i.e., only agent code is transferred and the agent starts its execution from the starting point on the remote machine. Strong mobility moves an agent who is currently performing its task (running) to another machine, i.e., agent code as well execution state (data stack) is transferred and the agent

continues its calculations/execution from the point of suspension on the remote machine like it was never interrupted. The basic aim is to perform access and information processing locally to a resource. On the other hand, mobility raises security and efficiency issues.

3. MOBILE AGENT

An intelligent agent with mobility feature is known as Mobile Agent (MA). MA is an autonomously transportable code migrating itself from one node to another in a heterogeneous network without losing its operability. In other words, program running at a host can suspend its execution at an arbitrary point, transfer itself to another host (or request the host to transfer it to its next destination) and resume execution from point of suspension. When the MA reaches a server, it is delivered to an agent execution environment (agent platform). Then, if the agent possesses necessary authentication credentials, its executable parts are started. To accomplish its task, the MA can transport itself to another server in search of the needed resource/service, spawn new agents, or interact with other stationary agents. Upon completion, the MA delivers the results to the sending client or to another server. Once it is launched by a user, can travel from node to node autonomously, and can continue to function even if the user is disconnected from the network. Various attributes of a MA are i) Identification to identify a MA and its dispatching station, ii) Itinerary consisting of the number and order of the hosts to be visited by MA, iii) Data unit containing the required data, iv) Code unit containing the transportable code, v) Execution state, and vi) External state for intermediate results[1],[3],[7], [12],[13],[14],[15],[16].

3.1 General benefits of MAs

MAs can be viewed as a replacement, refinement or extension of the traditional client/server paradigm. They are naturally heterogeneous and goal oriented in nature and may be launched into the unstructured network and roam around to gather information. MAs provide the following key advantages:-

- *Reduction in network bandwidth and latency:* MAs allow efficient and economical use of communication channels having low network bandwidth and high latency (response time delay). With MA, a single serialized object is transmitted over the network carrying the small amount of resultant data only, reducing the consumption of network bandwidth and latency.
- *Reduction in network traffic:* They reduce the network traffic because only resultant data is carried over the network to a central site and unnecessary intermediate results transmission is avoided.
- *Robust and fault-tolerant:* MA may have the intelligence to bypass the faulty hosts or seek temporary shelter on a reliable host. MAs are especially appropriate for wireless network links because the possibility of failure due to network faults is reduced since the MA needs only to be transported once between two sites.
- *Disconnected operation:* MA is not tightly bound to its dispatching host. It can continue functioning even if it is disconnected from its "home", and send back results upon reconnection. This feature is particularly useful for low-cost, light weight, portable computing devices such as Personal Digital Assistants (PDAs) having the low processing powers, memory constraints, and intermittent low bandwidth connection to the main network.
- *Load balancing:* Some of the processor and memory intensive tasks of a program in a lightweight mobile device can be offloaded to other more powerful and not heavily loaded platforms due to agent's strong mobility feature.
- *Scalable, configurable and reusable:* A distributed application, built as a collection of mobile agents, can be easily extended with new functionality in the form of agents with slight modifications to the master agent. Number of participating hosts can be increased without any significant impact on the complexity of the application. MAs are self-contained and highly reusable.
- *Cloning:* The parent agent can also clone several child agents to implement concurrent operations, and improve the efficiency.
- *Rapid prototyping:* Software components can be flexibly and dynamically deployed in the form of MAs. This facilitates the rapid prototyping of distributed applications.

4. AGENT TECHNOLOGY STANDARDS

There are currently a wide range of different agent architectures, frameworks and systems developed for both research and industrial purposes. To unify these approaches three standardization efforts have appeared with the overall aim of increasing interoperability between agent systems and even dealing with the problem of misuse involving MAs.

- *MASIF:* The Mobile Agent System Interoperability Facility (MASIF) has been in development by the Object Management Group (OMG) since 1995 to promote interoperability and mobility among agent platforms [17]. MASIF uses a procedure-oriented interaction model using Remote Procedure Calls (RPC) or Remote Method Invocation (RMI) technology.
- *KQML:* The Knowledge Query Meta Language (KQML) [18] is one of the most popular and widely used protocols for defining agent-to-agent communication. KQML is the oldest project, developed in 1992 by the DARPA Knowledge Sharing Effort consortium.
- *FIPA:* Foundation for Intelligent Physical Agents (FIPA) is a non-profit organization created in 1996 aimed at developing software standards (specifications) for maximizing interoperability within and across agent based systems. FIPA-ACL (Agent Communication Language) is adopted and used by many parties as their agent communication language. ACL uses KIF (Knowledge Interchange Format) and KQML to communicate knowledge and queries to others [19]. Three mandatory roles are present into a FIPA compliant agent platform: AMS (Agent Management System) that controls access and use of the platform, DF (Directory Facilitator) that provides yellow pages service and ACC (Agent Communication Channel) that provides the Message Transport Service for FIPA ACL messages delivery between agents living into different agent platforms. In order to promote interoperability between agent platforms, a number of standard Message Transport Protocols (MTPs) and MTP interfaces have been defined by FIPA. FIPA does not define nor require

a specific protocol for intra platform message delivery and each implementation can choose any Internal Message Transport Protocol (IMTP).

5. WHY JAVA?

The Java language has several inherent advantages that make it an ideal language for implementing agents: multithreading support, portability due to small size of compiled byte code and Java Virtual Machine(JVM), ease of network programming constructs(sockets), migration and communication support using object serialization and de-serialization [20], dynamic class loading and remote method invocation (RMI), highly secure program execution due to constantly growing security model (security managers, security policies, access to multiple databases over the Web using simple database connectivity interface (JDBC). In addition the JavaBeans specification enables the creation of reusable Java components with well-defined interfaces and behaviors.

6. APPLICATIONS OF AGENTS

The qualitative strengths of intelligent autonomous agents make current agent platforms an attractive choice for a wider range of distributed applications. Today, agents are deployed in different settings and application domains such as Information Retrieval System(IRS); Systems and Network management; Mobile Computing; Managing Mails and Messaging; Workflow and Administrative Management; Electronic Commerce; Industrial control; Internet searching; Personal assistance; Games; Software distribution; Bioinformatics; Clinical Data analysis for medical diagnosis; Distributed Data Mining(DDM); Intrusion Detection System(IDS); Peer-to-Peer Computing; Grid and Cluster Computing; Wireless Sensor Network(WSN); Unmanned Air and Under-Water Vehicles; Oil production systems etc.

7. MOBILE AGENT PLATFORM

A Mobile Agent Platform (MAP) a.k.a. Agent Execution Environment(AEE) or Agent Development (Tool)kit, is a middleware, distributed, server application that provides the appropriate functionality to MAs to authenticate, execute, communicate (with other agents, users, and other platforms), migrate to other platform, and use system resources in a secure way. A Multi-Agent System (MAS) is distributed application/system composed of multiple interacting intelligent agent components. The main purpose of a MAP is to provide a framework with higher level of abstraction for the development, execution and management of MAS without going into the low level communication details and just focusing on the desired functioning of the agents themselves [9],[21],[22],[23].

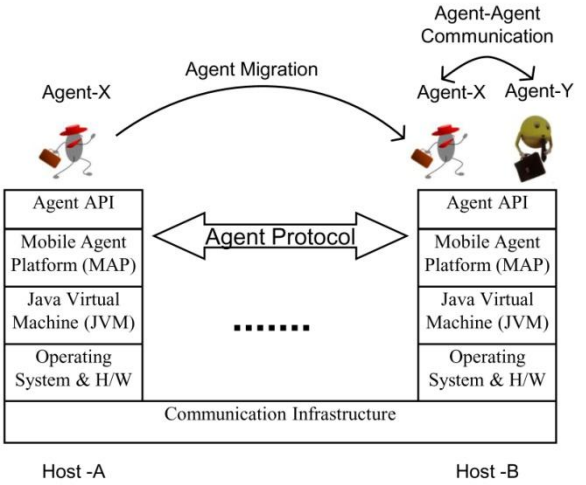

Fig 1: General architecture of a java based MAP

The conceptual model of a java based MAP is depicted in Figure 1. It consists of a) Host machine or a Network Node with Operating System and other hardware; b) Java Virtual Machine (JVM)-a runtime environment for platform independence and security; c) Mobile Agent Platform(MAP)-A multithreaded server application running on the top of JVM and provides the agent API with the basic functionality for agent management, migration, communication and security etc., in the form of java packages; and d) Communication Infrastructure (wired or wireless network) connecting the Host machines. Java MAs are special Java objects, executed as threads within an AEE. TCP/IP is used as the main transport mechanism.

7.1 Basic services provided by a MAP

The behavior of a MAP is based on the services it offers. MAPs provide a set of core services and could be enriched with new services, implemented as service agents on top of the core functionality. These properties can be functional and non-functional. The basic facilities provided by an AEE include the following:

- *Mobility:* Mobility mechanisms include remote invocation, itinerary management, cloning, and programming language support. Most platforms use application protocols on top of TCP for transport of agent codes and states. Java based platforms use remote method invocation (RMI), TCP Sockets, object serialization, and reflection.
- *Communication:* A communication model allows the agents to communicate with each other and the system to control agents. Most systems implement a distributed event communication and/or a message-passing mechanism using Java objects. Sockets are a low-level communication service which allows programs to exchange information. Java TCP socket classes are an abstraction of TCP connection oriented sockets. Remote Procedure Call (RPC) is a high-level communication mechanism. Java RMI is a type of RPC mechanism. RMI can be viewed as a newer object-oriented version of SunRPC. Communication performance of a MAP can be measured using Round trip time (RTT) .
- *Security:* The Security model of an AEE handles the security problems, i.e., how to protect the AEE from malicious agents, agents from a malicious AEE, one

agent from another, the communication between AEEs, and the host machine from the AEE. A variety of security mechanisms exist including encryption, public key infrastructure (PKI), and support for executing digitally signed code. PKI serves as a foundation for mobile agent security services and makes authentication, non-repudiation, and encryption readily available to agent developers and users. X.509 certification and Pretty Good Privacy (PGP) or the Secure Socket Layer (SSL) protocols represent two different approaches to the distribution of trust on the Internet. Digital signature also serves as a means of confirming the authenticity of an agent. Typically the code signer is either the creator of the agent, or the user of the agent. X.509 is based on the concept of Certification Authority (CA) server, which is a centralized control of trust to certify and manage all the certificates, while PGP is totally decentralized but very difficult to scale up the whole system. Traditional security mechanisms rely on cryptographic techniques to implement authentication, authorization, and access control. In most of java based systems, the designers choose to rely on the Java security model and/or provide customized extensions to this model. Security is an ongoing research issue and should be regarded as an ongoing feature. MA architecture should be designed a way so that new and better security mechanisms can easily be incorporated as they became available.

- *Identification/Naming Scheme:* Agents must be uniquely identified. Proper identification allows control, communication, cooperation, and coordination of agents to take place. All schemes used in the current systems are variations of generating a unique number sequence identifying each created agent.
- *Portability/Platform Heterogeneity:* All current MA systems must deal with porting agent code and data to work on multiple platforms. Before Java, systems such as Agent Tcl and ARA depended on an operating system specific and language specific interpreters for agent code and data. The problems of these approaches are performance and scalability. The Java virtual machine (JVM), although represents a better compromise because platform-neutral byte code can be compiled just in time to adequately improve the performance problems.
- *Resource Management:* Resource management component deals with the fair distribution of the access to low-level system resources such as CPU cycle, disk, memory, graphic subsystem and network among requesting agents. Agents may also require higher-level access such as persistence service (i.e., database), thread, and services from static entities (i.e., back-end directory server, SQL server).
- *Resource Discovery:* Resource discovery covers an area complementary to resource management. When an agent arrives at a site, it should be able to discover the services offered at that site. It would be much better if schemes were designed as part of the overall architecture to allow both prior and just-in-time discovery of services.
- *Control/Computation:* The basic goal of control is to provide the ways in which an agent may be created, started, suspended, stopped, duplicated, or instructed to self-terminate and how these lifecycle activities of a MA can be coordinated. The common approach is to provide these abilities in the root MA class. Computations are essentially organized in a "jump-act-jump" pattern.
- *Persistence:* The basic goal of persistence or data management is to save the agents and its associated data onto secondary storage for fault tolerance or other purposes. Data management is important to allow an MA system to scale and also affects performance.
- *Interoperability:* It is the ability of a system to work with or use the parts or equipment of another system. An AEE provides interoperability if it has compliance with certain standards like FIPA, OMG's MASIF etc.
- *Performance:* The performance of a multi-agent system is a measure using a set of statistical indicators of the system's major outputs and its consumption of resources, where typical indicators include throughput, response time, number of concurrent agents/tasks, computational time and communications overhead [24].
- *Scalability:* A system is said to scalable if its capacity to do useful work increases with the increase in the size of the system [24]. A MAS can be easily extended with new functionality in the form of agents with slight modifications to the master agent. Number of participating hosts can be increased without any significant impact on the complexity of the application. Scalability of a MAP can be measured as a performance degradation of the individual agent with the addition of extra nodes and extra agents in the agent society.
- *Stability:* Stability is a property of equilibrium. A MAS is in equilibrium when the statistical properties of its performance indicators remain stationary for a given variation in the system's external load. Thus, instability occurs when a small perturbation of some system parameter leads to a sharp and persistent deviation in the system's performance indicators [24].

8. QUALITATIVE COMPARISON OF IMPORTANT MAPs

Various MAPs have been developed both as academic research projects and commercial products by industries, with some platform projects abandoned altogether. These platforms can be broadly categorized as Java and non-Java based. Non-java based systems are operating system services accessible via a scripting language with interpreter (e.g., Tcl/Tk) and runtime support. Some of the important classical platforms that appeared in the literature during 1995 to 1999 are *Aglets* [25],[26], *ARA* [27], *Ajanta* [28], *Concordia* [29], *D'Agents /AgentTcl* [30],[31], *Grasshopper* [32], *Gypsy*[33], *Hive* [34], *Jackal* [35], *JAMES* [36],*MOLE* [37],*SOMA* [38],*TACOMA* [39],[40], and *ZEUS* [41].Other important current platforms are *3APL Platform* [42], *AAP*[43], *ABLE* [44], *ADK* [45], *AgentBuilder* [46], *Agentdock* [47], *AgentScape* [48], *CAPA*[49], *CapNet* [50], *Jack* [51], *C-BDI* [52], *Cougaar* [53],[54], *CybelePro* [55], *DECAF* [56], *FIPA-OS* [57],[58],[59], *Genie* [60], *IMAGO* [61],[62], *JADE* [63],[64],[65], *Jason* [66], *MadKit* [67], *Mage* [68], *MobileSpaces* [69],[70], *PMADE* [13],[14], *Sage* [71], *SEAGENT* [72], *Semoa* [73], *SHIP-MAI* [74], *Spade* [75], *SPRINGS* [76], *Tryllian's ADK* [77] and *Voyager* [78].

An exhaustive study of each MAP is not feasible due to the large number in existence. So, qualitative comparison of some prominent current MAPs is provided here and Table 1 and Table 2 shows the results of this study of current commercial as well as academic MAPs taking into account some of the features they provide. The names used here for MAPs and standards are the registered trademarks of their respective developers/organizations. The features include the following fields:-

- The **CP** field indicates whether a MAP is a commercial product ('C') or an academic research project ('A').
- **Lang** field shows the language in which each MAP is implemented
- **OS** indicates whether a MAP is open-source or not.
- **Std** indicates whether a MAP provides interoperability with other platforms or technology for which it should be in compliance with the agent technology standards such as FIPA, OMG-MASIF, etc.
- **Sec** points out whether security is taken into account.
- **Communication** indicates which communication technology each MAP uses such as RMI, TCP/IP etc.
- **Mob** indicates which mobility model is used, i.e., strong or weak.
- **GUI** is for the Graphical User Interface feature of the MAP.
- **BDI** points out the support for Belief-Desire-Intension model of practical reasoning.
- **CS** points out the Current Status (whether it is currently maintained and what is the last available version).
- **Use** indicates use of MAP in practical applications, development projects, case studies etc.
- **Miscellaneous** indicates any other additionally highlighted feature provided by MAP.

Table 1. Qualitative comparison of MAPs

Fetures ► Platform▼	**CP**	**Lang**	**OS**	**Std**	**Sec**	**Communication**	**Mob**	**GUI**	**BDI**
3APL	A	Java, 3APL	Yes	FIPA	No	RMI	weak	Yes	Yes
ABLE	C	Java	Yes	FIPA	Yes	RMI,HTTP, Data Flow, Events	weak	Yes	No
ADK	A	Java	No	No	Yes	asynchronous, synchronous, Jini, JavaSpaces, RMI	weak	Yes	Yes
AgentBuilder	C	Java	No	FIPA	Yes	TCP/IP, KQML	weak	Yes	Yes
AgentScape	A	Java, Python	Yes	No	Yes	SunRPC, HTTPS	weak	Yes	No
CAPA	A	Java	No	FIPA	No	synchronous, RMI	weak	Yes	No
CapNet	A	C#	No	FIPA	Yes	MTS, XML-SOAP, HTTP, DCOM, IIOP, SMTP	N/A	Yes	No
C-BDI	C	C++	No	FIPA	Yes	TCP/IP	N/A	No	Yes
Cougaar	A	Java	Yes	No	Yes	RMI,CORBA,HTTP	weak	Yes	No
CybelePro	C	Java	Yes	No	Yes	RMI	N/A	Yes	No
DECAF	A	Java	Yes	No	No	KQML	weak	Yes	No
FIPA-OS	C	Java	Yes	FIPA, MASIF	Yes	ACL,CORBA-IOR, IIOP, RMI, XML	weak	Yes	No
IMAGO	A	Prolog	No	No	Yes	messenger imago(agent)	N/A	Yes	No
Jack	C	Java	No	FIPA	Yes	TCP/IP, Event-Plan	weak	Yes	Yes
JADE	C	Java	Yes	FIPA	Yes	synchronous, asynchronous, Events, plugin MTP for RMI, IIOP, HTTP,WAP	weak	Yes	No
Jason	A	AgentSpeak, Java	Yes	FIPA	No	synchronous, asynchronous, speech-act, KQML	N/A	No	Yes
MadKit	A	Java	Yes	No	No	asynchronous, Java Sockets	weak	Yes	No
Mage	A	Java	Yes	FIPA	No	RMI	weak	Yes	No
MobileSpaces	A	Java	No	No	Yes	TCP,UDP,SMTP	weak	No	No
PMADE	A	Java	No	MASIF	Yes	Java TCP Sockets	weak	Yes	No
SHIP-MAI	A	Java	No	No	Yes	RMI,ADC	weak	Yes	No
Spade	A	Python	Yes	FIPA,	Yes	Jabber, XMPP, HTTP, ACL	N/A	Yes	No

				XMPP					
Tryllian's ADK	C	Java,J2ME	Yes	FIPA, SOAP XML, JXTA, JNDI	Yes	Events, JXTA pipe	strong	Yes	No
Voyager	C	Java	No	No	Yes	RMI, Java reflections, CORBA, DCOM	weak	Yes	No

Table 2. Qualitative comparison of MAPs

Platform	Features	Description
3APL	**CS**	3APL-M version for J2ME enabled mobile devices, last update on 19 November 2007
	Use	Universities Research Projects; mobile devices
	Miscellaneous	First platform to support cognitive agents
ABLE	**CS**	ABLE ver. 2.0; new features are continuously incorporated into the platform
	Use	agent-based modeling and simulation studies; system administration (IBM eServer iSeries); server diagnostics application; Feedback control system based on ABLE Autotune agents for automatically tuning the Apache Web server parameters; E-Commerce.
	Miscellaneous	Eclipse based light weight framework with JavaBeans/AbleBeans components; reusable; scalable; java serialization for persistence; ABLE Rule Language (ARL) for rule and policy-based knowledge representation. Agent Naming Service for identification
ADK	**CS**	not maintained further
	Use	air-ticket trading case study
	Miscellaneous	agent-oriented G-net model(a type of high-level Petri net) to define the agent structure, agent behavior, and agent functionality
AgentBuilder	**CS**	AgentBuilder Lite and AgentBuilder Pro
	Use	agent-based modeling and simulation studies; eBusiness applications
	Miscellaneous	IDE consisting of Project Manager, Ontology Manager,Agency Manager,Agent Manager and Protocol Manager for quickly and easily constructing intelligent BDI agents; Project Accessory Classes (PACs) contain the problem-specific code in Java, C, C++ libraries
AgentScape	**CS**	AgentScape 2 Milestone 5 Release July 2013
	Use	N/A
	Miscellaneous	scalable, fault tolerant
CAPA	**CS**	N/A
	Use	N/A
	Miscellaneous	An extension of Mulan (Multi Agent Nets) architecture; Reference Nets(a special kind of high-level Petri nets) and Concurrency
CapNet	**CS**	N/A
	Use	Contingency Management Simulation System (CMSS)
	Miscellaneous	Component Agent Platform based on .NET; policy manager and agent credentials for security
C-BDI	**CS**	a full DO-178B Level A certification package is planned for integration into any system's safety case
	Use	capable of executing a variety of decision-making systems in the aerospace, oil & gas and medical industries, and intelligent motor vehicle applications
	Miscellaneous	A certifiable BDI agent platform to easily port JACK-based applications to the new platform
Cougaar	**CS**	latest release Cougaar v12.7 (30-Jul-2012)
	Use	Critical Infrastructure Modeling and Simulation; CougaarME for robots in WAHN; DARPA UltraLog program
	Miscellaneous	Scalable; configurable; logical data model (LDM);survivable; heterogeneous; Java Security Manager and SSL for security
CybelePro	**CS**	CybelePro v. 3.1.3 is available as 3,6,10 and more than 10 nodes paid packages. Other versions of this product are OpenCybele v. 1.2.0(open source), CybeleLite v. 3.1.3(free

		license), CybelePro Enterprise v. 3.1.3 (10 Nodes paid package)
	Use	used extensively by the government, industry and academia for applications such as robotics, planning & scheduling, data-mining, modeling & simulation, and control of air and ground transportation systems, communication networks and cross-enterprise systems.
	Miscellaneous	N/A
DECAF	**CS**	N/A
	Use	Universities Research Projects
	Miscellaneous	N/A
FIPA-OS	**CS**	Standard FIPA-OS, MicroFIPA-OS
	Use	µFIPA-OS for PDA's and smart phones
	Miscellaneous	SSL for security
IMAGO	**CS**	N/A
	Use	N/A
	Miscellaneous	three kinds of imagoes (agents): stationary imago, worker imago and messenger imago; Multi-threading Logic Virtual Machine(MLVM) execution environment for agents
Jack	**CS**	CoJACK (JACK with cognitive architecture) enables the creation of principled models of human behaviour.
	Use	simulation systems for the military and decision support in the oil industry; Unmanned Air Vehicles applications in UK & US(BAC1-11 airliner,Taranis stealth UCAV)
	Miscellaneous	JACK Teams; lightweight; autonomous; rapid specification; compact/efficient; resilient
JADE	**CS**	Workflows and Agents Development Environment (WADE) is a JADE extension and WOLF is an Eclipse plugin; Lightweight Extensible Agent Platform (LEAP) package runs on Java ME enabled wireless devices
	Use	most widely used in Universities Research Projects; eBusiness Applications; Mobile Agent Malware Simulator
	Miscellaneous	efficient messaging service; agent cloning and ontologies can be designed using Protégé; Java Security Manager and SSL for security
Jason	**CS**	N/A
	Use	Gold Miners, Electronic Bookstore
	Miscellaneous	N/A
MadKit	**CS**	Last 5th stable version MaDKit-5.0.3.1 on November 13 2013
	Use	Universities Research Projects; study of multi-agent control in a production line; simulation of hybrid architectures for control of submarine robots
	Miscellaneous	organizational/relation model based on agent-group-role (AGR) concepts(AALAADIN conceptual model)
Mage	**CS**	N/A
	Use	N/A
	Miscellaneous	Agent-oriented Unified Model Platform(AUMP) modeling tool for system analysis and design stages; VAStudio development tool for system design, development and deployment stages (design module, programming module, runtime platform and toolkits)
MobileSpaces	**CS**	N/A
	Use	reusable mobile agents for network management
	Miscellaneous	agent hierarchy and inter-agent migration that allows a group of mobile agents to be dynamically assembled into a single mobile agent; Java Security Manager for security
PMADE	**CS**	PMADE version 1.0
	Use	Universities Research Projects; Network Load balancing, eCommerce; Information retrieval system; Distributed Data Mining
	Miscellaneous	generic architecture to support flexible and reusable components for modular system construction; effective mechanisms for termination of an agent and its clones; efficient mechanisms for retransmission of an agent; Naming and Location management; Itinerary management(serial,parallel,virtual serial); Java Security Manager for security
SHIP-MAI	**CS**	N/A

	Use	Mobile Agent based Software Distribution
	Miscellaneous	Agent Domain Controller(ADC); IAIK-JCE v2.5 , X.509 certificates and RSA for security and authentication
Spade	**CS**	Spade2
	Use	N/A
	Miscellaneous	XML Router as a Message Transport System (MTS);SPADE Agent Communication Channel (ACC); Extensible Messaging and Presence Protocol (XMPP);SSL for security
Tryllian's ADK	**CS**	N/A
	Use	Agent-Based Scheduling, Workstation Management, SMS Dating (Frog)
	Miscellaneous	based on a P2P library called JXTA; Transport Layer Security (TLS v. 1.0) , cipher suites, X.509 certification for authentication
Voyager	**CS**	N/A
	Use	Universities Research Projects
	Miscellaneous	provides simpler and better Service Oriented Architecture(SOA); location transparency through forwarding chains of proxies; unique identifiers for agents; own security manager

This analysis reveals that: there are few MAPs which supports BDI model of agency such as 3APL, ADK, AgentBuilder, C-BDI, Jack and Jason and out of theses 3APL and Jason are only open source products; PMADE has the provision of effective Itinerary management; Tryllian's ADK claims to provide 'strong' mobility while others mostly provide 'weak' mobility; majority of the platforms are academic research projects developed at the universities with sanctioned grants and are not maintained further; Security is still an issue as most of the java based systems rely on the basic security mechanism of JVM; ABLE, AgentBuilder, C-BDI, CybelePro, FIPA-OS, Jack, JADE, Tryllian's ADK and Voyager are the few commercial products and out of these JADE is the choice of most of the researchers today due to java based, open source, FIPA compliance, effective communication mechanism, GUI and well maintained documentation; Agent Oriented Software(AOS Group) is the world leader in autonomous systems software with Jack and C-BDI feathers in his cap and has a hold on the real application domains of defense, oil and gas production, unmanned air and under water vehicles.

Since the MAP market is evolving very fast, we would like to insist on the fact that the platforms analyzed in this paper are evaluated given their current state at the time of writing this paper(Jan. 2014), and that their specifications may change as time goes on. In addition to these traditional heavyweight MAPs, there are other lightweight MAPs specifically designed for the resource-constrained Wireless Sensor Network (WSN) such as *Agilla* [79] and *ActorNet* [80] for Mica2/TinyOS, *MAPS* [81], *AFME* [82] and *MASPOT* [83] for Sun SPOT/ Squawk Java ME Virtual Machine.

9. RELATED WORK

In the last few years, many researchers have focused on testing the performance of existing MAPs and such comparisons provide an overview of the different alternatives. Most of these studies highlight a lack of performance, security, communication, and scalability.

In one of the earlier studies Pham and Karmouch (1998)[12] compared *Aglet, Agent Tcl, ARA, Concordia, MOLE, Odyssey, TACOMA, SHIP-MAI,* and *Voyager* and presented an overview of these platforms in the context of telecommunications. They compared the MAPs with the criteria of security, portability, mobility, communication, resource management, resource discovery, identification, control, data management and case study in telecommunications. Silva et al. (2000)[84] compared the performance and the robustness of eight Java-based Mobile Agent systems: *Aglets, Concordia, Voyager, Odyssey, Jumping Beans, Grasshopper, Swarm* and *JAMES* by experimental study of benchmarking. They found *JAMES* to be best in terms of performance and robustness as compare to others.

Ricordel and Demazeau (2000)[85], in a survey on multi-agent platforms with a particular focus on methodology, evaluate *AgentBuilder, Jack, MadKit, and ZEUS* platforms with the criteria of analysis, design, development and deployment stages of software engineering with the qualitative values of Completeness, Applicability, Complexity, Reusability. Silva et al. (2001) [86] compared *Telescript, Aglets, ffMAIN, D'Agents* and *AgentSpace* MAPs according to the terminology, concepts and architecture proposed as part of the reference model. Criteria included in their reference model are: execution, management of agent types, management of identifiers, persistence, navigation, communication, interaction with external resources, and security. Camacho et al. (2002)[87] shows the performance of *ZEUS, JADE* and *SkeletonAgent* MAPs when a MAS composed by a several web agents is launched. This MAS provides documents requested by a user agent. In this sense, the authors measure the number of requested documents per time. Therefore, the conclusions are only valid for this MAS. Vrba (2003)[88] presents an evaluation of the messaging service performance of *JADE, FIPA-OS, ZEUS*, and *Jack*. From the tests presented in this paper the authors conclude that *JADE* provides the most efficient messaging service and Zues the worst one. However, the design reasons that cause this performance are not given and the implementations of the messaging service for each MAP are not detailed.

Cortese et al. (2003)[89] test the scalability and performance of the *JADE* messaging service. Chmiel et al. (2004)[90] also extended their work and test the *JADE* messaging, agent creation and migration services. In Spamming test, User agents are flooded with ACL messages by the Spammer Agent. In database access test, SQLAgents insert messages into database. The tests they perform related to the messaging service only scale up to 8 agent pairs. Burbeck et al. (2004)[91] test the messaging service

performance of *JADE, Tryllian's ADK* and *Safeguard Agent Platform (SAP)*. They claim that JADE performs better than the others because is built on Java RMI2 but give no proofs confirming this claim. Furthermore, other services apart from the messaging service should also be tested.

Mawlood-Yunis et al. (2004a, 2004b)[92][93] studied the performance behavior of *Aglets* and *TACOMA* in a distributed search using single and multiple agents. The results indicate that the two mobile agent platforms have similar behavior but their performance varies with underling implementation, and the multiple agents approach performs better than the single agent approach in large networks. Su and Li (2005)[94] compared the performance of Prolog based *IMAGO* with *Aglets, D'Agents* and *Kaariboga*. *IMAGO* performs worse than the other three systems in migration and communication and performs better in agent's creation only. Raquel et al. (2007)[95] compared *Aglets, Voyager, Grasshopper, Tryllian, JADE, Tracy,* and *SPRINGS* qualitatively and evaluate their performance in a variety of settings with an extensive set of experiments. Leszczyna (2008)[96] evaluate the publicly available platform implementations enlisted at FIPA site (http://www.fipa.org/resources/livesystems.html) with the criteria of platform maintenance and popularity. Author found that the continuously maintained platforms are *ADK, JACK* and *JADE* and out of these *JADE* is the only one which is well supported (documentation, mailing list, platform updates) and free.

Alberola et al. (2010)[22] performed an in-depth study, specifically on the messaging and directory services, of *JADE, MadKit* and *AgentScape* in order to find out to what extent the internal design of a MAP influences its performance. The *JADE* messaging service performs better than the messaging service of the other two MAPs. In *MadKit*, directory service is provided by the MAP, in a distributed way. In *JADE* and in *AgentScape* this service is centralized. The authors conclude that services like directory service, which stores information and receives a lot of requests, should be distributed among the various hosts in the MAP whenever possible, for improving the response time of these services.

In general, the different works where some MAPs are compared are partially outdated today, as new platforms and versions have appeared since the publication of such works.

10. CONCLUSIONS

The qualitative strengths of intelligent autonomous MAs make intelligent software agent paradigm an attractive choice for a wider and wider range of distributed applications. Researchers interested in this field should now focus more on the designing and practical implementation of MAS in various domains by using the existing MAPs instead of developing a new MAP. With this study, we expect to contribute to cover the need of an updated review of MAPs and also to encourage future work in the intelligent software agent technology.

11. ACKNOWLEDGMENTS

We sincerely acknowledge all the authors and commercial organizations listed in the reference whose work and products are analyzed and views are provided in this study.

12. REFERENCES

[1] Nwana, H.S. 1996. Software Agents: An Overview. Knowledge Engineering Review, 11(3):1-40, Sept 1996.

[2] Bradshaw, J.M. 1997. An introduction to software agents. Software Agents, MIT Press Cambridge, MA, USA, pp. 3-46, 1997.

[3] Klusch, M. 2001. Information Agent technology for the internet: A survey. Data and Knowledge Engineering, Special Issue on Intelligent Information Integration, Elsevier Science 36:337-372, 2001.

[4] Hewitt, C. 1977. Viewing Control Structures as Patterns of Passing Messages. Artificial Intelligence, 8(3): 323-364,1977.

[5] Luck. M., et al. 2005. Agent technology: computing as interaction (A Roadmap for Agent Based Computing). AgentLink, 2005.

[6] Gilbert, D., Aparicio, M., Atkinson, B., Brady, S., Ciccarino, J., Grosof, B., O'Connor, P., Osisek, D., Pritko, S., Spagna, R., and Wilson, L. 1995. IBM Intelligent Agent Strategy. IBM Corporation, 1995.

[7] Wooldridge, M. and Jennings, N.R. 1995. Intelligent Agents: Theory and Practice. Knowledge Engineering Review, 10(2):115-152, 1995.

[8] Franklin, S. and Graesser, A. 1996. Is it an Agent, or just a Program? : A Taxonomy for Autonomous Agents. In: Proceedings of the Workshop on Intelligent Agents III, Agent Theories, Architectures, and Languages, Springer-Verlag, pp. 21-35, 1996.

[9] Dilyana, S. and Petya, G. 2002. Building distributed applications with Java mobile agents. International e-Business Plan Tutorial – Appendix D: SOFTWARE (INTELLIGENT) AGENTS, Distance Learning at Prentice Hall , Prentice-Hall, Inc. , 2003 , available at http://myphliputil.pearsoncmg.com/student/bp_turban_introec_1/Turban-AppendixD.pdf

[10] Shoham, Y. 1993. Agent-oriented programming. Artificial Intelligence, 60(1):51−92, 1993.

[11] Bratman, M.E., Israel, D.J., and Pollack, M.E. 1988. Plans and resource-bounded practical reasoning. Computational Intelligence, 4:349−355, 1988.

[12] Pham, V. A. and Karmouch, A. 1998. Mobile Software Agents: An Overview. IEEE Communications Magazine, pp. 26-37, July 1998.

[13] Patel, R.B. and Garg, K. 2001. PMADE: A Platform for mobile agent Distribution & Execution. In: Proceedings of the 5th World Multi Conference on Systemics, Cybernetics and Informatics (SCI2001) and 7th International Conference on Information System Analysis and Synthesis (ISAS 2001),Orlando, Florida, USA, July 22-25, 4: 287-293, 2001. See also web page http://www.drrbpatel.org/papers/pmade.html

[14] Patel, R.B. and Garg, K. 2004. A new paradigm for mobile agent computing. WSEAS Transactions on Computers, 3(1):57-64, 2004.

[15] Picco, G.P. 2001. Mobile Agents: An Introduction. Microprocessors and Microsystems, Elsevier, 25(2):65-74, 2001, doi:10.1016/S0141-9331(01)00099-0

[16] Wooldridge, M. 2002. Intelligent agents: The key concepts. In: Mark, V. et al. (Eds.), Multi-Agent-System and Applications, LNCS, 2322:3-43, Springer-Verlag, , 2002.

[17] Milojicic, D., et al. 1998. MASIF The OMG Mobile Agent System Interoperability Facility. In: Proceedings of the International Workshop on Mobile Agents (MA '98), Lecture Notes in Computer Science, 1477:14-15, Springer-Verlag Berlin Heidelberg, 1998.

[18] Finin, T., et al. 1997. KQML as an Agent Communication Language. In: Bradshaw, J. (Ed.), Software Agents, MIT Press/AAAI Press Cambridge, MA, pp. 291-316,1997.

[19] O'Brien, P. and Nicol, R. 1999. FIPA- Towards a Standard for Software Agents. British Telecommunication (BT) Technology Journal, 16(3):51-59, 1999.

[20] Oracle and/or its affiliates 2005. Java Object Serialization Specification, available at http://docs.oracle.com/javase/7/docs/platform/serialization/spec/serialTOC.html

[21] Gray, R.S, et al. 2000. Mobile agents: Motivations and state-of-the art systems. Dept. of Computer Science, Dartmouth College Hanover, New Hampshire, 2000, available at http://www.cs.dartmouth.edu/~dfk/papers/gray-motivation-tr.pdf

[22] Alberola, J.M., et al. 2010. A performance evaluation of three multiagent platforms. Artificial Intelligence Review, 34:145–176,doi:10.1007/s10462-010-9167-9

[23] Cardoso, R.C., Diego, C., Bordini, M., and Rafael, H. 2013. Agent Programming and Multi-agent Systems project (http://agentprogramming.com/) at Benchmarking Agent Programming Languages (APLs), http://www.inf.pucrs.br/maop.benchmarking/index.php

[24] Lee, L.C., Ndumu, D.T., and Wilde, P.D. 1998. The stability, scalability and performance of multi-agent systems. British Telecommunication (BT) Technology Journal, 16(3):94–103,1998.

[25] Venners, B. 1997. The Architecture of aglets. http://www.javaworld.com/article/2077639/core-java/the-architecture-of-aglets.html, See also Aglets Specification Draft 1.1, Available at http://www.research.ibm.com/trl/aglets/spec11.htm

[26] Lange, D.B. and Oshima, M. 1998. Programming and deploying Java mobile agents with Aglets. Addison Wesley, 1998.

[27] Peine, H. and Stolpmann, T. 1997. The architecture of the ARA platform for mobile agents. In: Rothermel, K. and Popescu-Zeletin, R., (Eds.), Mobile Agents, LNCS, 1219:86–97, Springer-Verlag, 1997.

[28] Tripathi, A. R., Karnik, N.M., Vora, M.K., Ahmed, T., and Singh, R.D. 1999. Mobile agent programming in Ajanta. In: Proceedings of the 19th IEEE International Conference on Distributed Computing Systems, pp. 190-197, 1999.

[29] Wong, D., et al. 1997. Concordia: An Infrastructure for Collaborating Mobile Agents. In: Rothermel, K. and Popescu-Zeletin, R., (Eds.), Mobile Agents, LNCS, 1219: 50–61, Springer-Verlag, 1997.

[30] Gray, R.S. 1997. Agent Tcl: A Flexible and secure mobile agent system, Ph.D. Thesis , Dept. of Computer Science, Dartmouth College Hanover, New Hampshire, 30th June 1997.

[31] Gray, R.S., et al. 1998. D'Agents: Security in a multiple-language, mobile-agent system. In: Vigna, G. (Ed.), Mobile Agents and Security, LNCS,1419:154-187, Springer-Verlag, 1998. Available at http://agent.cs.dartmouth.edu/papers/gray:security-book.pdf

[32] Breugst, M., et al. 1998. Grasshopper: A Mobile Agent Platform for IN Based Service Environments. In: Proceedings of the IEEE IN Workshop, Bordeaux, France, May 70-13, 1998, pp. 279-290,1998.

[33] Lugmayr, W. 1999. Gypsy: a component-oriented mobile agent system. Available at citeseer.ist.psu.edu/lugmayr99gypsy.html

[34] Minar, N., Gray, M., Roup, O., Krikorian, R., and Maes, P. 1999. Hive: distributed agents for networking things. In: Proceedings of the First International Symposium on Agent Systems and Applications and Third International Symposium on Mobile Agents (ASA/MA'99), August 1999.

[35] Cost, R.S., Finin, T., Labrou, Y., Luan, X., Peng, Y., Soboroff, I., Mayfield, J., and Boughanam, A. 1998. Jackal: A Java based Tool for Agent Development. In: Proceedings of the Workshop on Tools for Agent Development(AAAI-98), Madison, WI.

[36] Silva, L.M., Simoes, P., Soares, G., Martins, P., Batista, V., Renato, C., Almeida, L., and Stohr, N. 1999. JAMES: A Platform of Mobile Agents for the Management of Telecommunication Networks. LNCS,1699:76-95, Springer-Verlag, 1999.

[37] Baumann, J., et al. 1997. Mole - Concepts of a Mobile Agent System, Institute for Parallel and Distributed High-Performance Computers. Breitwiesenstraße 20-22, 70565 Stuttgart, pp. 1-25.

[38] Bellavista, P., et al. 1999. Secure and Open Mobile Agent Programming Environment. In: Proceedings of the 4th International Symposium on Autonomous Decentralized Systems (ISADS '99), Tokyo, Japan, March 21-23, 1999, pp. 238-245. Available at http://lia.deis.unibo.it/research/SOMA/

[39] Sudmann, N.P. 1996. TACOMA- Fundamental Abstractions supporting Agent Computing in A Distributed Environment. Ph.D. Thesis, Department of Computer Science, University of Tromsø, Norway, August 1996. Web page http://www.tacoma.cs.uit.no//index.html

[40] Johansen, D., Schneider, F.B., and Renesse, R. 1998. What TACOMA Taught Us. Also in: Milojicic, D., Douglis, F., and Wheeler, R. (Eds.), "Mobility, Mobile Agents and Process Migration - An edited Collection", Addison Wesley Publishing Company, 1998.

[41] Collis, J.C., Ndumu, D.T., Nwana, H.S., and Lee, L.C. 1998. The ZEUS agent building tool-kit. British

Telecommunication (BT) Technology Journal, 16(3):60–68,1998.

[42] Ten Hoeve, E.C., Dastani, M., Dignum, F., and Meyer, J.-J. 2003. 3APL Platform. Master's Thesis, Utrech University. Available at http://www.cs.uu.nl/3apl/thesis/tenhoeve/tenhoeve.html or http://www.cs.uu.nl/3apl/

[43] Ceccaroni, L. and Dale, J. 2002. The April Agent Platform - Features and Summary. Network Agent Research Group, Fujitsu Laboratories of America. Feb. 2002. Available at http://www.angelfire.com/scifi2/technopapa/2002_02___aap_v08.pdf

[44] Bigus, J.P., Schlosnagle, D.A., Pilgrim, J.R., Mills III,W.N., and Diao, Y. 2002. ABLE: a toolkit for building multiagent autonomic systems. IBM Systems Journal, 41(3):350–371,2002.

[45] Xu, H. and Shatz, S.M. 2003. Adk: an agent development kit based on a formal design model for multi-agent systems. Journal of Autonomous Software Engineering, 10(4):337–365,2003.

[46] Acronymics, Inc. 2004. AgentBuilder- An Integrated Toolkit for Constructing Intelligent Software Agents. Reference Manual Version 1.4, June 16, 2004, Available at http://www.agentbuilder.com/Documentation/ReferenceManual-v1.4.pdf

[47] Jarvinen, J. 2002. Agentdock platform BDI-agents. Minutor Ltd, www.cs.uta.fi/kurssit/AgO/ago7a.pdf

[48] Brazier, F., et al. 2002. Agentscape: middleware, resource management, and services. In: Proceedings of the 3rd international SANE conference, pp. 403–404, Available at www.iids.org/publications/sane02.pdf

[49] Duvigneau, M, et al. 2003. Concurrent Architecture for a multi-agent Platform. In: Giunchiglia, F., Odell, J., and Weiß, G. (Eds.) Revised Papers and Invited Contributions, 3rd international workshop Agent-Oriented Software Engineering (AOSE III), LNCS 2585:59–72, 2003,Springer-Verlag, 2003.

[50] Contreras. M., Germán, E., Chi, M., and Sheremetov, L. 2004. Design and implementation of a FIPA compliant agent platform in .NET. Journal of Object Technology, Special issue:.NET Technologies, 3(9):5–28, available at www.jot.fm/issues/issue_2004_10/article1.pdf

[51] AOS Group 2008. JACK - An Agent Infrastructure for Providing the Decision-Making Capability Required for Autonomous Systems. Available at http://www.agent-software.com/products/jack/

[52] AOS Group 2011. C-BDI - A certifiable reasoning engine for intelligent & autonomous systems. Available at http://www.agent-software.com/products/c-bdi/

[53] Helsinger, A., Thome, M., and Wright, T. 2004. Cougaar: a scalable, distributed multi-agent architecture. In: Proceedings of the IEEE International Conference on Systems, Man and Cybernetics(SMC2004), 10-13 Oct. 2004, 2:1910–1917, doi: 10.1109/ICSMC.2004.1399959

[54] Snyder, R.D., MacKenzie, D.C., and Tomlinson, R.S. 2004. Robustness Infrastructure for Multi-Agent Systems, In: Proceedings of the 1st International Conference Open Cougaar 2004, New York, pp. 75-80, July 2004. Web page http://cougaar.org/

[55] IA Inc. 2004. CybelePro - Intelligent Agent Framework, Version 3.0 User's Guide. DISTRIBUTED INTELLIGENT SYSTEMS, Intelligent Automation Inc.(IAI), Rockville,2004, Available at http://www.i-a-i.com/products/doc/cybelepro/UsersGuide-CybelePro.pdf , Web page http://www.i-a-i.com/cybelepro/

[56] Graham, J.R., Decker K., and Mersic, M. 2003. DECAF—A Flexible Multi Agent System Architecture. Autonomous Agents Multi-Agent Systems, 7(1–2):7–27,2003.

[57] Poslad, S., Buckle, P., and Hadingham, R. 2000. The FIPA-OS agent platform: Open Source for Open Standards. Manchester, UK, April 2000. Available at http://fipa-os.sourceforge.net/docs/papers/FIPAOS.pdf

[58] Emorphia and Contributions 2003a. FIPA-OS Developer's Guide. Emorphia Ltd 2003,mAvailable at http://fipa-os.sourceforge.net/docs/Developers_Guide.pdf

[59] Emorphia and Contributions 2003b. FIPA-OS Inter-platform Communication Configuration Guide.nEmorphia Ltd 2003, Available at http://fipa-os.sourceforge.net/docs/Interplatform_Configuration_Guide.pdf

[60] Riekki, J., Huhtinen, J., Ala-Siuru, P., Alahuhta, P., Kaartinen, J., and Roning, J. 2003. Genie of the net, an agent platform for managing services on behalf of the user. Computer Communications, 26(11):1188–1198, July 2003.

[61] Li, X. 2001. IMAGO: A Prolog-based System for Intelligent Mobile Agents. LNCS, 2164:21-30, 2001

[62] Xu, Z. and Li, X. 2004. Protecting hosts against attacks in IMAGO system. In: Proceedings of the Canadian Conference on Electrical and Computer Engineering, (CCECE 2004) , Niagara Falls, 1:361-364, 2-5 May 2004, doi: 10.1109/CCECE.2004.1345030

[63] Bellifemine, F., Caire, G., Poggi, A., and Rimassa, G. 2003. Jade a white paper. Telecom Italia EXP Magazine, 3(3):6–19, Web page of TILAB http://jade.tilab.com/

[64] Bellifemine, F.L., Caire, G., and Greenwood, D. 2007. Developing Multi-Agent Systems with JADE. Wiley.

[65] Bellifemine, F., Caire, G., Poggi, A., and Rimassa, G. 2008. JADE: A software framework for developing multi-agent applications-Lessons learned. Information and Software Technology, 50(1–2):10–21.

[66] Bordini, R.H., Wooldridge, M., and Hübner, J.F. 2007. Programming multi-agent systems in AgentSpeak using Jason. Wiley Series in Agent Technology,Wiley, New York.

[67] Gutknecht, O. and Ferber, J. 2001. The madkit agent platform architecture. LNCS, 1887:48–551,2001,web site http://www.madkit.org/

[68] Shi, Z., Zhang, H., Cheng, Y., Jiang, Y., Sheng, Q., and Zhao, Z. 2004. Mage: an agent-oriented programming environment. In: Proceedings of the 3rd IEEE international conference on cognitive informatics (ICCI

'04), Washington DC, pp. 250–257, 2004, doi: 10.1109/ICCI.2004.20

[69] Satoh, I. 2000a. MobileSpaces: A Framework for Building Adaptive Distributed Applications using a Hierarchical Mobile Agent System. In: Proceedings of the International Conference on Distributed Computing Systems(ICDCS), Taipai, pp. 161-168, 10-13 Apr 2000. doi: 10.1109/ICDCS.2000.840918

[70] Satoh, I. 2000b. A Formalism for Hierarchical Mobile Agents. In: Proceedings of the International Symposium on Software Engineering for Parallel and Distributed Systems(PDSE), Limerick, pp. 165-172, 10-11 Jun 2000. doi: 10.1109/PDSE.2000.847863

[71] Ahmad, H.F., Suguri, H., Ali, A., Malik, S., Mugal, M., Shafiq, M.O., Tariq, A., and Basharat, A. 2005. Scalable fault tolerant Agent Grooming Environment - SAGE. In: Proceedings of the 4th international joint conference on Autonomous Agents and Multiagent Systems (AAMAS'05),ACM Press,New York, pp. 125–126, doi:10.1145/1082473.1082816

[72] Dikeneli, O., Erdur, R.C., Gumus, O., Ekinci, E.E., Gurcan, O., Kardas, G., Seylan, I., and Tiryaki, A.M. 2006. SEAGENT: A Platform for Developing Semantic Web Based Multi Agent Systems. LNCS, 3963:1-13, Springer-Verlag,2006

[73] Roth, V. and Jalali-Sohi, M. 2001. Concepts and architecture of a SEcurity-centric MObile Agent server. In: Proceedings of the 5th International Symposium on Autonomous Decentralized Systems (ISADS), pp. 435 – 442, 2001, doi:10.1109/ISADS.2001.917449

[74] Tkito, R. and Karmouch, A. 2000. SHIPMAI: a secure and high performance mobile agent infrastructure. In: Proceedings of the Canadian Conference on Electrical and Computer Engineering (CCECE 2000), 2:1027 – 1031, 07-10 March 2000, doi: 10.1109/CCECE.2000.849621

[75] Escriva, M., Palanca, J., Aranda, G., Garca-Fornes A., Julian, V., and Botti, V. 2006. A jabber-based multi-agent system platform. In: Proceedings of the 5th international joint conference on autonomous agents and multiagent systems (AAMAS06), ACM Press, New York, pp. 1282–1284, doi:10.1145/1160633.1160866

[76] Ilarri, S., Trillo, R., and Mena, E. 2006. SPRINGS: A Scalable Platform for Highly Mobile Agents in Distributed Computing Environments. In: Proceedings of the International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM'06), Buffalo-Niagara Falls, NY, 2006 doi:10.1109/WOWMOM.2006.103

[77] Tryllian 2005. Tryllian's Agent Development Kit (ADK). Available at http://www.tryllian.com/adk.html

[78] Wheeler, T. 2005. Voyager Architecture Best Practices, Recursion Software, Inc., Texas, USA, 2005. Available at http://www.recursionsw.com/inc/wp/2005-03-30_Voyager_Architecture_Best_Practices_Voyager.pdf . Web page for Voyager is http://www.recursionsw.com/products/voyager/voyager-intro.html

[79] Fok, C.-L., Roman, G.-C., and Lu, C. 2009. Agilla: A mobile agent middleware for self-adaptive wireless sensor networks. Transactions on Autonomous and Adaptive Systems, 4(3): Article 16 (July 2009), 26 pages. doi: 10.1145/1552297.1552299

[80] Kwon, Y., Sundresh,S. Mechitov, K., and Agha, G. 2006. ActorNet: An Actor Platform for Wireless Sensor Networks. In: Proceedings of the 5th International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS), pp. 1297-1300, 2006.

[81] Aiello, F., Fortino, G., Gravina, R., and Guerrieri, A. 2011. A Java-Based Agent Platform for Programming Wireless Sensor Networks. The Computer Journal, 54(3): 439-454.

[82] Muldoon, C., O'Hare, G.M.P., O'Grady, M.J., and Tynan, R. 2008. Agent Migration and Communication in WSNs. In: Proceedings of the 9th International Conference on Parallel and Distributed Computing, Applications and Technologies (PDCAT 2008), Otago, pp. 425 – 430, 1-4 Dec. 2008, doi: 10.1109/PDCAT.2008.58

[83] Lopes, R., Assis, F., and Montez, C. 2011. MASPOT: A mobile agent system for Sun SPOT. In: Proceedings of the 10th international symposium on Autonomous decentralized systems (ISADS), pp. 25-31, 23-27 March 2011, doi: 10.1109/ISADS.2011.10

[84] Silva, L.M., Soares, G., Martins, P., Batista, V., and Santos, L. 2000. Comparing the performance of mobile agent systems: a study of benchmarking. Computer Communications, 23:769–778, 2000.

[85] Ricordel, P.M. and Demazeau, Y. 2000. From Analysis to Deployment: A Multi-agent Platform Survey. LNCS, 1972:93-105, Springer-Verlag, 2000.

[86] Silva, A.R., Rom¨ao, A., Deugo, D., and Silva. M.M.D. 2001. Towards a reference model for surveying mobile agent systems. Autonomous Agents and Multi-Agent Systems, 4(3):187–231, 2001.

[87] Camacho, D., Aler, R., Castro, C., and Molina, J.M. 2002. Performance evaluation of ZEUS, Jade, and SkeletonAgent frameworks. In: Proceedings of the IEEE international conference on Systems, man and cybernetics(ICSMC), 4:(6-9), Oct. 2002 , doi: 10.1109/ICSMC.2002.1173303

[88] Vrba, P. 2003. Java-based agent platform evaluation. LNCS, 2744:47-58, Springer-Verlag, 2003.

[89] Cortese, E., Quarta, F., and Vitaglione, G. 2003. Scalability and performance of JADE message transport system. EXP 3:52–65.

[90] Chmiel, K., Tomiak, D., Gawinecki, M., and Karczmarek, P. 2004. Testing the efficency of JADE agent platform. In: Proceedings of the 3rd International Symposium on Parallel and Distributed Computing / 3rd International Workshop on Algorithms, Models and Tools for Parallel Computing on Heterogeneous Networks (ISPDC/HeteroPar'04), pp. 49–56,2004. Burbeck, K., Garpe, D., and Nadjm-Tehrani, S. 2004. Scale-up and performance studies of three agent platforms. In: Proceedings of the IEEE International Conference on Performance, Computing, and

Communications (PCCC 2004), pp. 857 – 863, 2004, doi: 10.1109/PCCC.2004.1395203

[91] Mawlood-Yunis, A., Nayak, A., Nussbaum, D., and Santoro, N. 2004a. Comparing Performance of Two Mobile Agent Platforms in Distributed Search. In: Proceedings of the IEEE/WIC/ACM International Conference on Intelligent Agent Technology (IAT'04), pp. 425 – 428, 20-24 Sept. 2004 doi: 10.1109/IAT.2004.1342987

[92] Mawlood-Yunis, A., Nayak, A., Nussbaum, D., and Santoro, N. 2004b. On the Performance of Distributed Search by Mobile Agents. LNCS, 3284:285-294, Springer-Verlag, 2004

[93] Su, Y. and Li, X. 2005. Performance analysis of IMAGO system. In: Proceedings of the Canadian Conference on Electrical and Computer Engineering (CCECE 2005), Saskatoon, pp. 1029 - 1032, 1-4 May 2005, doi: 10.1109/CCECE.2005.1557151

[94] Raquel, T., Sergio, I., and Eduardo, M. 2007. Comparison and Performance Evaluation of Mobile Agent Platforms. In: Proceedings of the 3rd International Conference on Autonomic and Autonomous Systems(ICAS07), pp. 41-46, pp. 19-25, June 2007, doi:10.1109/CONIELECOMP.2007.66

[95] Leszczyna, R. 2008. Evaluation of Agent Platforms Version 2.0, Joint Research Centre(JRC) Scientific and Technical Reports, Luxemburg,2008.