A multi-agent system approach for trustworthy cloud service discovery

Article in Cogent Engineering · November 2016
DOI: 10.1080/23311916.2016.1256084

CITATIONS
4

READS
52

3 authors, including:

Akinwale Akinwunmi
Bowen University
18 PUBLICATIONS 21 CITATIONS
SEE PROFILE

Emmanuel Olajubu
Obafemi Awolowo University
31 PUBLICATIONS 38 CITATIONS
SEE PROFILE

Some of the authors of this publication are also working on these related projects:

Project
Information Privacy View project

Project
e-learning View project
A multi-agent system approach for trustworthy cloud service discovery

A.O. Akinwunmi, E.A. Olajubu and G.A. Aderounmu

Cogent Engineering (2016), 3: 1256084
A multi-agent system approach for trustworthy cloud service discovery


Abstract: Harnessing the benefits of cloud computing, demands that a potential user has adequate access to trustworthy cloud services. Locating resources and services in a heterogeneous network such as cloud environment may be a tedious task. The cloud computing is a form of distributed system with a complex aggregation of computing resources from different domains with different administrative policies but having immense benefits that could enhance the mode of computing. Discovering cloud services is complicated by the varied level of quality of service offered by the different providers, hence a monolithic approach to cloud service discovery cannot effectively support the cloud environment. This work proposed a distributive approach to discovering trustworthy cloud services with the use of Multi-Agents System for achieving intelligent cloud service discovery. Simulation experiments were carried out and the results showed that extending the Multi-Agents System approach by integrating trust into the process of cloud service discovery improves the quality of service in terms of response time and scalability. A further...
comparative analysis of Multi-Agents System approach for cloud service discovery to monolithic approach showed that it is highly efficient, and highly flexible for trustworthy cloud service discovery.

**Subjects:** Mathematics & Statistics; Computer Science; Engineering & Technology

**Keywords:** cloud service; multi-agent; service discovery; distributed computing; functional analysis

1. Introduction

A cloud is a form of parallel and distributed system consisting of a collection of inter-connected and virtualised computers. Their resources are provisioned dynamically and presented as one or more unified computing resources based on service-level agreements established through negotiation between the service provider and users (Buyya, Yeo, & Venugopal, 2008). It offers a new model where computing resources such as power, storage and online applications are shared as “services” over the internet (Vaquero, Rodero-Merino, Caceres, & Lindner, 2009). It is a paradigm that focuses on sharing data and computations over a scalable network of nodes. Examples of such nodes include end user computers, data centres, and cloud services. Such a network of nodes is termed as a cloud (Mei, Chan, & Tse, 2008). The cloud computing is a distributed system with a complex aggregation of computing resources from different domains with different administrative policies but having immense benefits that could enhance the mode of computing (Akinwunmi, Olajubu, & Aderounmu, 2014).

In a multi-domain environment such as cloud that hosts thousands of service instances, discovering cloud services is an enormous task owning to the fact that there is a large number of different services from different providers having different levels of quality of service. A service discovery system that will ensure the correctness and completeness of discovery is germane to having trustworthy cloud services in cloud environment rather than in a small, restricted and homogeneous environment. Completeness of service discovery requires the system to retrieve relevant results from across the domain boundaries and across potentially-large network distances. Complementarily, a high degree of correctness is necessary so that these relevant results do not get lost among a flood of a possibly huge number of irrelevant results (Kousiouris, Kyriazis, Varvarigou, Oliveros, & Mandic, 2012). A monolithic approach to service discovery cannot support the diversity of the cloud environment that is distributed across the various domain boundaries and across different heterogeneous networks. Discovering a trustworthy cloud service requires a cloud service discovery system that is distributive and not monolithic in its approach. Hence, this work aimed at using the multi-agent system (MAS) another distributed system as a tool for discovering trustworthy cloud services across the domain boundaries in cloud environment. In Section 1.1, we describe related work and appraised them. In Section 2, the conceptual architecture design was explained and its components were discussed. In Section 3 the simulation experiments carried out and their results were also highlighted, followed by conclusions in Section 4.

1.1. Related work

MAS is a distributed system consisting of multiple software agents, which form “a loosely coupled network, called a multi-agent system (MAS), to work together to solve problems that are beyond their individual capabilities or knowledge of each entity” (Durfee & Montogomery, 1989). MAS are a community of autonomous agents working together in order to achieve a goal (Wooldridge, 2009). Of particular interest are MAS in which the individual agents display significant intelligence and autonomy. Over the years, MAS technologies have found applications in many distributed systems such as distributed problem solving, distributed information fusion, and distributed scientific computing (Drashansky, Houstis, Ramakrishnan, & Rice, 1999; Honavar, Miller, & Wong, 1998; Khosla & Dillon, 1997).

Although there are many differences between cloud computing and MAS, both are two distributed computing models, therefore several common problems can be identified and many more benefits
can be obtained by the combined use of cloud computing systems and multi-agents (Talia, 2011). There are several researches that have attempted the use of agent technology in cloud computing.

Kang and Sim (2010) provided a four-stage, agent-based cloud service discovery protocol. Software agents containing negotiation agents, brokering agents and information agents were used for bolstering the resource management system because of their scalability and adaptability with high level abstraction for modelling of complex software systems. The approach employed the use of self-organising agents for automating and coordinating resource management with promising highly dynamic behaviours. In addition, the variation in type of the resources is indispensable in heterogeneous systems like the cloud. Also request matching with service advertised was only the focus in the brokering process and not in the final service deployed for use (Kang & Sim, 2010). Though MAS approach was used in this work, but the issue of trust was not considered among the cloud entities while cloud services were been discovered.

Han and Sim (2010) worked on a cloud service discovery system (CSDS) that supported the cloud users in finding a cloud service over the Internet. The prototype of the CSDS was made up of a search engine and three different agents namely; query processing agent, filtering agent, and cloud service reasoning agent (CSRA). There were two components: (1) CSDS which helps to find the best cloud service on behalf of users and (2) a cloud ontology which comprises taxonomy of concepts of different cloud services to consult with the CSRA. In addition, there was a user interface which allows the user to enter queries containing a service name and requirements considered by their preferences. From the empirical results, the CSDS with the cloud ontology performed better than the CSDS without the cloud ontology. By contacting a cloud ontology to reason about the relations among cloud services, the CSDS was more successful in locating cloud services and more likely to discover cloud services that meet users’ requirements (Han & Sim, 2010). The functionalities of query processing, filtering and rating were partially implemented but trustworthiness of the cloud service was not a concern in the process.

Gopinadh and Saravanan (2013) presented an agent-based cloud system for cloud service reservation. The work entails an agent-based cloud service discovery approach that consults ontology to retrieve information about cloud services and implements price and timeslot negotiation (PTN) mechanism for cloud service reservation. It was designed for the cloud service that best suit their price and timeslot. The important contributions of the work includes: developing an agent-based search engine for cloud service discovery and effective adaptation of agent-based PTN mechanisms for cloud service negotiation. CSRA which enables the CSDS was used to reason about the relations of cloud services, rate the search results and design and construction of cloud ontology. Cloud agents in PTN are designed to concurrently make multiple proposals in a negotiation round that generate aggregated utility, differing only in terms of individual price and time-slot utilities (Gopinadh & Saravanan, 2013). Trustworthiness of cloud services was not considered in the effort.

Sim (2009) proposed an agent-based testbed for bolstering the discovery of cloud resources and service level agreement (SLA) negotiation. The testbed for simulating cloud resource management comprises (i) a set of resource consumers, (ii) a set of physical machines, (iii) a set of cloud resources (virtualized machines), and (iv) a set of middleware comprising of provider agents and consumer agents acting as intermediaries between resource providers and consumers, respectively, and a set of broker agents that connects resource requests from consumers to advertisements from providers. Based on the connection algorithm used for connecting buyers and sellers in e-commerce, this work transformed the cloud resource discovery process into a process of matching consumers’ resource requests to resources of providers. This was achieved by a four-stage resource discovery process of selection, evaluation, filtering, and recommendation, a set of broker agents matches consumers’ requests to advertisements from providers. After the matching of requests to resources, consumer and provider agents then negotiate for mutually acceptable resource time slots (Sim, 2009). It is noted that the ideas, design, and development of an agent-based cloud resource management testbed were still evolving. Meanwhile trust issues were not addressed.
Talia (2011) proposed an approach to develop a multi agent based system that makes use of ontology relationships to find the similarities between the user queries and the available resources from various service providers. The system entails the use of agent approach as agents can be autonomous and mobile for easy communication with each other over a distributed environment. The proposed approach demonstrates a typical prototype of a cloud service discovery system that comprises three different agents—requestor agent, mapper agent, and search agent—and a search engine. The agents are developed using Java agent development framework. The search engine used a concept based search using similarity reasoning from the ontology relationships of the concepts in the cloud ontology. As multiple agents are used, the average waiting time of a search was reduced also; the overall waiting time was also reduced (Talia, 2011). The use of ontology has increased the chances of finding an exact match for the services requested. This work clearly demonstrated that MAS can aid effective cloud discovery in a distributed environment.

Rajendran and Swamynathan (2014) proposed the use of cloud ontology technique to make discovery of cloud services more efficient. It utilized a multi-broker agent based discovery system that consults ontology while attempting to retrieve information about services. A cloud service discovery using broker agents (CSDBA) system was developed. The approach combines the advantages of the agent-based system and the semantic matching mechanism using cloud ontology. The CSDBA, an agent-based service discovery system, interface with the cloud ontology for reasoning about the relations of cloud services. It is designed to support consumers in finding appropriate cloud services. CSDBA was more successful and can be helpful tool to find cloud services that meet requirements of user than traditional approach. In the future, it was indicated that cloud services brokering mechanism would be extended (Rajendran & Swamynathan, 2014). The future agenda included implementing efficient methodology for cloud crawling and rate the services based on QoS (Rajendran & Swamynathan, 2014). Table 1 contained the summary of the studies carried out by the cited authors.

Parhi, Pattanayak, and Patra (2015) proposed a framework for service description and discovery using ontology based on multi-agent cloud environment. Because of the non-standardized specification terminology used by different cloud providers, the traditional search engines such as Google, MSN, bing, etc. are not able to perform the service discovery in an effective manner. There arises the necessity of a reasoning mechanism for service discovery that is capable of resolving the similarities among variety of services. In Parhi et al. (2015), introduced a semantic-based service description and discovery framework using multi-agent approach, where the cloud service descriptions that are automated based on shared ontology, contribute to optimal discovery of appropriate services as requested by consumers was addressed. The reason mechanism failed to incorporate the trustworthiness of cloud services as a factor for reasoning.

Moses, Ojo, and Zuva (2016) presented an efficient and scalable approach for semantic cloud service discovery in a P2P (peer-to-peer) cloud environment. The approach was based on learning automata (LA) and ant colony optimization (ACO). The ability of ACO to adapt to changes in real time makes it a better choice in dynamic environments such as cloud. The approach showed an improved performance against the some existing P2P service discovery approaches. Meanwhile, attention was not given to the trustworthiness of cloud services in the work.

Liu, Xing, Che, and Bao (2015) proposed a centralized robust service discovery approach with local and global service discovery capabilities which can be applied to agent-based cloud computing environment. The system consists of four kinds of agents: resource agents (RA), service provider agents (SPA), broker agents (BA) and consumer agents (CA). The proposed system has an overlay network above the physical network in order to perform global searching. It adopted indexes to be able to handle intermittent and volatile network environment like cloud computing environment. The approach generates minimize traffic during the service publication; service discovery process because of each index handles its own jurisdiction. But the system lacked a structured network topology on the virtual network in order to achieve global service query with higher efficiency and there was no
<table>
<thead>
<tr>
<th>Author(s)</th>
<th>Concern</th>
<th>Contribution</th>
<th>Remark</th>
</tr>
</thead>
<tbody>
<tr>
<td>Kang and Sim (2010)</td>
<td>Lack of tools for supporting cloud service discovery</td>
<td>The paper showed how self-organizing agents were employed for automating and</td>
<td>Though MAS approach was used in this work, but the issue of trust was not considered among the</td>
</tr>
<tr>
<td></td>
<td></td>
<td>coordinating resource management with promising highly dynamic behaviours</td>
<td>cloud entities while cloud services were been discovered</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>It was planned that the cloud services brokering mechanism will be extended in the future for</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>more intelligent discovery</td>
</tr>
<tr>
<td>Han and Sim (2010)</td>
<td>There was no discovery mechanism for searching different kinds of clouds</td>
<td>The paper revealed that an agent-based discovery system that consults an</td>
<td>The functionalities of query processing, filtering and rating were partially implemented but</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ontology when retrieving information about cloud services is a possibility</td>
<td>trustworthiness of the cloud service was not a concern in the process</td>
</tr>
<tr>
<td>Gopinadh and Saravanan (2013)</td>
<td>Finding cloud service that best suits price and time slot of users</td>
<td>The paper showed that using agent based approach aided effective service</td>
<td>Cloud agents in PTNs are designed to concurrently make multiple proposals in a negotiation round</td>
</tr>
<tr>
<td></td>
<td></td>
<td>negotiation</td>
<td>that generate aggregated utility, differing only in terms of individual price and time-slot</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>utilities. Trustworthiness of cloud services was not considered in the negotiation</td>
</tr>
<tr>
<td>Sim (2009)</td>
<td>The concern was on the design and development of software agents for</td>
<td>The significance of this work was the introduction of an agent-based</td>
<td>It is noted that the ideas, design, and development of an agent-based cloud resource management</td>
</tr>
<tr>
<td></td>
<td>bolstering cloud service discovery, service negotiation, and service</td>
<td>paradigm for constructing software tools and testbeds for cloud resource</td>
<td>testbed were still in infancy and trust issues were not addressed in the processes</td>
</tr>
<tr>
<td></td>
<td>composition</td>
<td>management</td>
<td></td>
</tr>
<tr>
<td>Talia (2011)</td>
<td>Discussion on the potential benefits of integrating multi-agent technology with cloud computing</td>
<td>This paper reported that multiple agents can exchange information and</td>
<td>This work clearly demonstrated that MAS can aid effective cloud service discovery in a</td>
</tr>
<tr>
<td></td>
<td></td>
<td>cooperate to perform complex tasks where interaction, intelligence,</td>
<td>distributed environment and adding trust concern to the system will aid intelligent cloud service</td>
</tr>
<tr>
<td></td>
<td></td>
<td>adaptation and dynamicity are key issues to be handled</td>
<td>discovery</td>
</tr>
<tr>
<td>Rajendran and Swamynathan (2014)</td>
<td>Finding appropriate cloud services</td>
<td>The work utilized a multi-broker agent based discovery system that</td>
<td>The retrieved information about services did not include trustworthiness of the cloud services</td>
</tr>
<tr>
<td></td>
<td></td>
<td>consults ontology while attempting to retrieve information about services</td>
<td></td>
</tr>
<tr>
<td>Parhi et al. (2015)</td>
<td>There were no standardized specifications defined for a cloud service</td>
<td>The paper proposed a semantic-based service description and discovery</td>
<td>The reason mechanism failed to incorporate the trustworthiness of cloud services as a factor for</td>
</tr>
<tr>
<td></td>
<td>provider. Different cloud service providers use different vocabulary to</td>
<td>framework using multi-agent approach</td>
<td>reasoning</td>
</tr>
<tr>
<td></td>
<td>specify similar operations</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

(Continued)
provision for accommodating trustworthiness of cloud services. This centralized approach is liable to single point of failure in a dynamic cloud environment and its scalability cannot be assured.

The review of the related works showed that attempts have been made in using MAS to address the cloud service discovery process but the issue trustworthiness of cloud service while discovering cloud services have been neglected and addressing the issue will aid the cloud user to discovery trustworthy cloud services. This work is leveraging on the capabilities of the MAS to extend the discovery system to ensure that trustworthy cloud services are discovered. Extending the multi-agent paradigm for trustworthy cloud service discovery will enable efficient cloud service discovery and improve the process.

2. Methodology

The conceptual design for a trustworthy cloud service discovery system was based on Functional Analysis which is a crucial tool for the design of process to explore new concepts and to define their architectures in system engineering (Viola, Corpino, Fioriti, & Stesina, 2012). It was used to define the functional requirements for a trustworthy cloud service discovery system. It ensured that all necessary components are included and that no unwanted components were included. The relationships between the system components were also specified. The main step of Functional Analysis was made up of the functional tree, the functions/devices matrix and the product tree. Through the functional tree and particularly through the identification of the basic functions, the functional requirements of the

![Figure 1. Cloud service discovery functional block diagram.](image-url)
system were defined, and through the product tree the building blocks of the system were determined, thus providing the functional architecture of the system. Starting from the top level system requirements of trustworthy cloud service discovery using multi-agent approach the functional analysis led to the proper identification of the building blocks, which comprises the proposed system, and their interconnection to build up the functional architecture of the proposed system. The functional analysis method produced the following fundamental subsystems as requirements for achieving efficient cloud service discovery: cloud user subsystem, cloud provider subsystem, trust evaluator subsystem, broker subsystem and database as depicted in Figure 1. Achieving a trustworthy cloud service discovery system required that the various subsystems must collaborate, cooperate and interact together.

2.1. System architecture
The system architecture was constructed from the basic components of the functional block for cloud service discovery and the architecture is a distributed system. The architecture has the following three layers as shown in Figure 2.

Figure 2. System architecture.
(i) Application and resource layer.
(ii) Agent layer: classes of agents.
(iii) Database layer: holding information status of applications/resources/services.

2.1.1. Application and resource layer entails the cloud service models
The layer is the collection all cloud services available providing the interface for connecting to the cloud resources and services. It entailed the cloud service models. A cloud service is any resource that is provided over the internet. The most common cloud service resources are software as a service (SaaS), platform as a service (PaaS) and infrastructure as a service (IaaS). SaaS is a software distribution model in which applications are hosted by a vendor or service provider and made available to customers over a network, typically the Internet. PaaS refers to the delivery of operating systems and associated services over the internet without downloads or installation. IaaS involves outsourcing the equipment used to support operations, including storage, hardware, servers and networking components, all of which are made accessible over a network.

2.1.2. Agent layer
The system construction was based on the multi-agent approach for achieving effective and reliable discovery. The agents are used as basic components for achieving intelligence in the system, making it more adaptive, flexible, and autonomic in its operation. This opportunity is employed for implementing efficient MAS and, from a more general point of view, for advancing the way to design and implement a service discovery system that is distributive and trust enabled in cloud environment.

In this layer the various agents that are involved in trustworthy cloud service discovery are situated. The agents are; cloud user agents (CUA), trust evaluator agents (TEA), cloud provider agents (CPA) and cloud broker agents (CBA).

These autonomous agents with their various functions establish appropriate link among themselves for efficient cloud service discovery. In doing this, a peer-to-peer approach is adopted among the various agents for the discovery. The participating agents, referred to as a peer, functions as a client with a layer of server functionality in their operations. This allows the peer to act both as a client as well as a server within the context of a given cloud service discovery interaction. Interaction may spread over all the peers, and peers can communicate directly with each other, and hence, are constantly aware of each other, or they may communicate indirectly through other peers for effective discovery. In the latter case, the arrangement or the organization of peers may either have a regular structure, or be unstructured. An indirect unstructured approach removes the need for any form of management to enforce a specific network configuration; however, the emphasis and the workload is shifted to the cloud service discovery effort as changes to the cloud environment can happen without the awareness of all the peers.

In this layer, cloud service discovery occurs by means of and through communication among the agent entities. Given this approximation, discovery becomes an inherently social activity, instead of solitary or monolithic, leading to efficient forms of discovering trustworthy cloud services. The cloud discovery is no longer monolithic but rather distributed by agent entities. These agents form coalitions among themselves to attain their current goals of cloud service discovery.

2.1.2.1. Functions of each agent. Having identified the various agents needed for achieving a trustworthy cloud service discovery, the roles of each of the agents are as follows:

2.1.2.1.1. User agent. The user agent provides interface for the users to make service request. It sends request to broker agents using message passing and shows the results to user through a user interface. It interacts with provider agent through the broker agents for service usage. The user agent interacts with trust evaluator agent to monitor the performance value of the provider in terms of service type, service integrity and service accessibility.
Algorithm

Start
Let N = total number of users request within a specified time
While (User’s request ≠ ⊘) Do
  For i = 1, i++, N
    User_Agent ← User’s request: UR_i
    Broker_Agent for UR_i discovery ← User_Agent
    Wait for Broker_Agent
    User_Agent ← Broker_Agent returns Result of UR_i discovery
    User Interface ← User_Agent Result for use
    If (UR_i is used) Then
      User_Agent interacts with Trust_Evaluator_Agent on UR_i
    End if
  Next i
End do
End

2.1.2.1.2. Broker agent. The broker agent analyses the user’s request to determine its type. It then locates a possible match for the identified service request and returns the list of such services to the user agent. It sends user-provider’s interaction information to the database. It also interacts with database for other broker agent recommendation. The broker agent receives trust value updates from trust evaluator agent.

Algorithm for

Start
If (Broker_Agent received User_Agent Message) Then
  Broker_Agent Analyse UR_i
Search the database for a match
  If (UR_i match is found) then
    Broker_Agent ← UR_i_match
  Else
    Other_Broker_Agents ← UR_i
    If (UR_i match is found by Other_Broker_Agents) Then
      Broker_Agent ← UR_i_match
    Else
      Broker_Agent ← UR_i_miss
    End if
  End if
User_Agent ← Broker_Agent
End if
End
Algorithm for
Start
While (a typical service is in use) Do
   Broker_Agent Interact with Trust_Evaluator_Agent
   Broker_Agent ← Trust_Evaluator_Agent (trust value updates)
   Database ← Broker_Agent (Stores user agent—provider agent interaction information)
End Do
End

2.1.2.1.3. Trust evaluator agent. It keeps track of the provider service provider's behaviour contributing to a proper operation of the cloud in terms of the following factors: service type, service integrity, and service accessibility by interacting with user and provider agents. It determines the trustworthiness of cloud service provider. It maintains a table for keeping the performance of the provider and updates it when new performance value is available.

Start
While (User_Agent Interacts with Provider_Agent) Do
   Trust_Evaluator_Agent: Keeps track of provider's performance
   Trust_Evaluator_Agent: Determines the trustworthiness of providers
   Broker_Agent ← Trust_Evaluator_Agent (trust value updates)
   Trust_Evaluator_Agent: Cache the providers' performance
End Do
End

2.1.2.1.4. Provider agent. Provider agent is distributed in the cloud, one at each cloud service provider site. It sends advertisement, removal and updates information of cloud services to the broker agent. It provides services for usage by the user agent.

Algorithm
Start
While (a service is being offered by service provider) Do
   Provider_Agent: Create service instance
   Broker_Agent ← Provider_Agent (Broadcast service availability)
   Broker_Agent ← Provider_Agent (Broadcast service expire information)
   Broker_Agent ← Provider_Agent (Broadcast service update)
   User_Agent ← Provider_Agent (Provides service)
End Do
End

The corresponding class diagram and sequence diagram of the services rendered by the agents are shown in Figures 3 and 4 respectively.

2.1.3. Database layer
The database layer stores the service usage, agents interaction information. It Stores service information and trust value rating for providers. It also keeps the brokers interaction information. In a typical cloud service discovery system, users submit their request to the broker. The broker consults the registry or directory (database) to find a possible match for the user’s request. The provider publishes its available services into the service registry or directory (database) and also updates the status of the published service.
3. Simulation, results and discussion
Simulation experiments were conducted to investigate the effect of extending MAS approach to achieve a trustworthy cloud service discovery using response time and scalability as the evaluation parameter. The response time in this regard refers to time lag between the instant of service request by the users to the instant of having the result of the service request. This depends on round trip time (RTT) and the users load (UL). Scalability is a metric that observes how the system can be easily altered to accommodate changes in the number of users, resources and computing entities attached to it. Scalability depends on processing capability of the provider expressed as processing time (PT) and users load (UL).

The experiment examined the cloud service discovery transactions carried out without involving trust factor against the same kind of cloud service discovery transaction involving trust factor in order to investigate their difference. CloudAnalyst (Wickremasinghe, Calheiros, & Buyya, 2010) useful to model and analyze large scale cloud computing development was used for the simulation. Considering the description and behaviour of throttled load balancing policy of the CloudAnalyst, the integration of trust capability in cloud environment was achieved using this policy while the round-robin load balancing policy, considering its behaviour was used for the cloud transaction without trust integration.
The first experiment simulated cloud service discovery transaction using CloudAnalyst. In this experiment, there were 5, 10, 15, 20, 25 and 30 groups of user agents with a varied number cloud provider agents and the user agent growth factor varied from 100 to 10,000,000. These groups of user agents were generated randomly using Poisson distribution by varying the number of users in a realistic manner. The detailed parameter settings for the experiment were shown in Table 2. In this experiment, the number of user agents’ growth factor were varied while the number of the provider agent was varied in order to observe the behaviour of the existing system without trust integration and proposed system with trust integration in terms of the response time.

Table 3 shows the results obtained from the simulation. The in graph Figure 5 shows the plot of the round trip time against the number of user agents to verify the behaviour of the proposed and existing systems in terms of response time. From the results, the proposed system ensures lower and gracefully stable response time than the existing system with higher response time. This is evident from the percentage difference of the systems' response time as shown in Table 3. It was observed that integration of trust into the process of cloud service discovery was responsible for the lower response time of the proposed system.

This second experiment simulated cloud service discovery transaction using CloudAnalyst. In this experiment there were 5, 10, 15, 20, 25 and 30 groups of user agents with varied number cloud provider agents having a capacity of 5 virtual machines each. The user agents' growth factor was varied from 100 to 10,000,000. The detailed parameter settings for the experiment were as shown in Table 4.

In this experiment, the number of user agent growth factor were varied while the number of the provider agent was varied in order to observe the behaviour of the existing system without trust integration and proposed system with trust integration in terms of the scalability.

<table>
<thead>
<tr>
<th>User agents</th>
<th>Provider agent</th>
<th>User growth factor</th>
<th>Request growth factor</th>
<th>Execution instruction per length</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>1</td>
<td>100</td>
<td>10</td>
<td>100</td>
</tr>
<tr>
<td>10</td>
<td>2</td>
<td>1,000</td>
<td>10</td>
<td>100</td>
</tr>
<tr>
<td>15</td>
<td>3</td>
<td>10,000</td>
<td>10</td>
<td>100</td>
</tr>
<tr>
<td>20</td>
<td>4</td>
<td>100,000</td>
<td>10</td>
<td>100</td>
</tr>
<tr>
<td>25</td>
<td>5</td>
<td>1,000,000</td>
<td>10</td>
<td>100</td>
</tr>
<tr>
<td>30</td>
<td>6</td>
<td>10,000,000</td>
<td>10</td>
<td>100</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>User agents growth factor</th>
<th>Round trip time (ms) (Existing system)</th>
<th>Round trip time (ms) (Proposed system)</th>
<th>Percentage difference (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>302.12</td>
<td>300.81</td>
<td>-0.434</td>
</tr>
<tr>
<td>1,000</td>
<td>302.42</td>
<td>300.19</td>
<td>-0.737</td>
</tr>
<tr>
<td>10,000</td>
<td>303.10</td>
<td>301.15</td>
<td>-0.643</td>
</tr>
<tr>
<td>100,000</td>
<td>303.10</td>
<td>301.08</td>
<td>-0.666</td>
</tr>
<tr>
<td>1,000,000</td>
<td>303.71</td>
<td>301.13</td>
<td>-0.849</td>
</tr>
<tr>
<td>10,000,000</td>
<td>303.49</td>
<td>301.40</td>
<td>-0.689</td>
</tr>
</tbody>
</table>
Table 5 shows the results obtained from the simulation. The graph in Figure 6 shows the plot of the provider agent processing time against the number of user agents’ growth factor to verify the behaviour of the proposed and existing systems in terms of scalability. From the results, the proposed system scales the load of user agents fairly better than the existing system as the number of user agents increases and the processing capability of each of the provider agents remain fixed. This is evident from the percentage difference of the systems' scalability metric as shown in Table 5. It was observed that integration of trust into the process of cloud service discovery brought about this achievement.

3.1. Simulation results validation
From the simulation results, the proposed system with trust performed better than the existing one without trust. In order to validate the simulation results, paired difference t-test was performed on the percentage difference of the response time with six group size in the simulation settings for the proposed and the existing systems. The null hypothesis states that there is no significant difference between the proposed and existing system while the alternative hypothesis states that there is significant difference between them.

The calculated t values for the response time in the experimental setting was −70.43 as shown in Table 6. The degrees of freedom (df) for the total group size of six is five. Entering a t table with 5 df, at 95% confidence interval, the table t value is 2.02 (one-tailed, a significance level (α) of 0.05). The absolute calculated t value is higher than the tabled t value of 2.02 showing that the proposed system with lower response time is significantly different (p = 0.05) from the existing system as a result of trust integration in it.
Performing a similar \( t \)-test on the percentage difference of the scalability metric with six group size in the simulation settings for the proposed and the existing systems, resulted in the calculated \( t \) values of \(-3.09\) respectively as shown in Table 5. Entering a \( t \) table with 5 \( df \), at 95\% confidence level, the table \( t \) value is 2.02 (one-tailed, a significance level (\( \alpha \)) of 0.05). The absolute calculated \( t \) value \(-3.09\) is higher than the table \( t \) value showing that the proposed system is more scalable and significantly different (\( p = 0.05\)) from the existing system as a result trust integration in it.

4. Conclusion

In this work, we have proposed a distributive approach using MAS for trustworthy cloud service discovery. Simulation experiments were carried out and the results showed that extending the MAS approach by integrating trust into the process of cloud service discovery improves the quality of service in terms of response time and scalability. This distinguished the proposed system with trust integration in the process of cloud service discovery from the existing system without trust integration. The research indicated that the use of MAS within cloud environment can aid intelligent cloud service discovery and support the complexity of the cloud environment. A further comparative

<table>
<thead>
<tr>
<th>Table 5. Scalability with varied number of user agents’ growth factor and varied number of provider agents</th>
</tr>
</thead>
<tbody>
<tr>
<td>User agents growth factor</td>
</tr>
<tr>
<td>100</td>
</tr>
<tr>
<td>1,000</td>
</tr>
<tr>
<td>10,000</td>
</tr>
<tr>
<td>100,000</td>
</tr>
<tr>
<td>1,000,000</td>
</tr>
<tr>
<td>10,000,000</td>
</tr>
</tbody>
</table>

Figure 6. Scalability with varied number of user agents’ growth factor and varied number of provider agents.

<table>
<thead>
<tr>
<th>Table 6. Calculated ( t ) values for the existing system and proposed system</th>
</tr>
</thead>
<tbody>
<tr>
<td>Performance metrics</td>
</tr>
<tr>
<td>Response time</td>
</tr>
<tr>
<td>Scalability</td>
</tr>
</tbody>
</table>

![Scalability graph](image)
analysis of MAS approach for cloud service discovery to monolithic approach showed that it is highly efficient and highly flexible for trustworthy cloud service discovery. MAS provide efficient solution in environment where resources are spatially and potentially distributed whereas a monolithic approach is highly inefficient and not reliable for a distributed environment such as cloud. The proposed distributive approach using MAS for trustworthy cloud service discovery is not susceptible to single point of failure. Future work will consider optimizing the MAS distributive approach to cloud service discovery.

Funding
The authors received no direct funding for this research.

Author details
A.O. Akinwunmi; E.A. Olajubu; G.A. Aderounmu

References

Citation information

Cover image
Source: Authors.


