Chapter 3

Multi-Agent System Approach for Trustworthy Cloud Service Discovery

Akinwale Akinwunmi, Emmanuel Olajubu and Ganiyu Aderounmu

Additional information is available at the end of the chapter

http://dx.doi.org/10.5772/intechopen.84291

Abstract

Accessing the advantages of cloud computing requires that a prospective user has proper access to trustworthy cloud services. It is a strenuous and laborious task to find resources and services in a heterogeneous network such as cloud environment. The cloud computing paradigm being a form of distributed system with a complex collection of computing resources from different domains with different regulatory policies but having a lot of values could enhance the mode of computing. However, a monolithic approach to cloud service discovery cannot help the necessities of cloud environment efficiently. This study put forward a distributive approach for finding sincere cloud services with the use of Multi-Agents System for ensuring intelligent cloud service discovery from trusted providers. Experiments were carried out in the study using CloudAnalyst and the results indicated that extending the frontiers MAS approach into cloud service discovery by way of integrating trust into the process improves the quality of service in respect of response time and scalability. A further comparative analysis of the Multi-Agents System approach for cloud service discovery to monolithic approach showed that Multi-Agents System approach is highly efficient, and highly flexible for trustworthy cloud service discovery.

Keywords: cloud services, multi-agent system, service discovery, distributed system, functional analysis

1. Introduction

Cloud Computing offers and delivers computing resources as services such as servers, storage, databases, networking, software, analytics and a lot more over the Internet. The collection of these resources is dynamically provided according to service-level agreements negotiated

among cloud service provider and consumer [1]. It offers a replacement computing model in which resources like power, storage and online applications are shared as 'services' over the net [2]. It is an archetype that depends on sharing data and computations over an ascendable network of nodes. The nucleus of the nodes revolves round end user computers, data centers, and cloud services. Such a collection of nodes is defined as a cloud [3]. Cloud is a distributed system with a fancy aggregation of computing resources from various domains with different management policies however having vast edges that would enhance the mode of computing [4].

Cloud Computing is among the next generation in computing [5]. It is closely related to previous, well-known distributed computing initiatives such as Web services and grid computing [6]. It relies on sharing computing resources in which having local servers or personal devices to handle applications are not required.

Clouds point to computational resources that are accessible as scalable, on demand, pay as-you-go services provided in the Internet. It makes accessible infrastructure, platform, and software as subscription-based services in a pay-as-you-go manner to users. The basic classifications for the services are namely; Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Software as a Service (SaaS) in industries [7]. However, a number of other classifications have been derived from these basic classifications.

Some examples of the cloud services are email services in which users gain access to their email hosted "in the cloud" from any computing device with a browser and enabled Internet connection. This is a departure from the traditional local hosting of email services. With innovations in cloud computing today, a number of Web apps like VoIP for making free voice call on the Internet, social networking solutions for social connection online, online media services for multimedia content viewing and sharing, financial apps and many more applications are deployed on the cloud. Even traditional desktop software, such as Microsoft Office suite has been deployed on the cloud as a service.

As stated in [8], cloud computing has the potential to add value to organizations, industries, and global economy by:

- Spontaneously accelerating the way companies innovate on products and services, supporting team product development by collaborating professionals globally with access to more powerful and economical computing resources
- Speeding up the ability of organizations to mine their data for data analytics
- Giving access to information technology for all sizes and shapes of organizations
- Assisting in advancing a virile and highly competitive global economy

In a multi-facets atmosphere like cloud that hosts thousands of service instances, discovering cloud services is an infinite task owning to the very fact that there are massive types of services from varied providers having completely different levels of quality of service.

A service discovery system which will ensure the correctness and completeness of discovery is related to having trustworthy cloud services in cloud atmosphere than in an exceedingly tiny restricted and same atmosphere. Completeness of service discovery depends on the ability of

the system to intelligently show relevant results from across the outlined boundaries and across potentially-wide network distances. Likewise a high level of correctness is needed in order that the relevant results do not stray amidst a flood of a probably vast range of irrelevant results [9]. A monolithic approach to service discovery cannot support the distinctiveness of the cloud surroundings that is distributed across the varied domain boundaries and across completely different heterogeneous networks. Therefore, a distributive approach to cloud service discovery that is trustworthy is highly desired. Multi-Agents System (MAS) approach for cloud service discovery will assist greatly in addressing the challenge of discovering a trustworthy cloud service. MAS is an aggregation of multiple autonomous (intelligent) agents which collectively cooperate together for solving a distributed task such as cloud service discovery.

2. Service discovery system

Resource discovery is any mechanism that is providing capabilities to locate a service. Service discovery is a part of resource discovery and it is envisioned as the competence to find certain services such as applications or clearly outlined network services which are not pure conjectures [10].

Finding resources in an exceedingly giant scale, multi-domain network could be a non-trivial task [4]. There is also several providers, each with totally different characteristics, therefore providing a spread of varied levels of user-requirement suitableness [9]. In order to create awareness regarding the standing and also the handiness of those resources, a Service Discovery (SD) mechanism should be put in place. All the service providers are needed to register in such a mechanism, together with all the mandatory information like the service uniform resource locator (URL), rates, compatibility and service interface in order to expose the services to the would be users [9]. A matchmaking of the client requested service with available resources is done by the information service and the results is returned to the requester. **Figure 1** shows a top-level overview of SD's generic architecture.

A Service Discovery System provides a mechanism for:

i. services to register their availability

ii. locating a single instance of a particular service

iii. notifying when the instances of a service change

User requirement are used as input for discovering the best suited cloud services among various repositories of cloud providers [11].

Service and resource discovery are crucial to support any service infrastructure like cloud computing. A large-scale, multi-domain service infrastructure requires a service discovery system that is open, scalable, robust and efficient to a greater extent than a single-domain system [12].

In a heterogeneous environment such as cloud in which thousands of service instances are hosted, the correctness and completeness of discovery are very important than in a small, homogeneous environment [12].

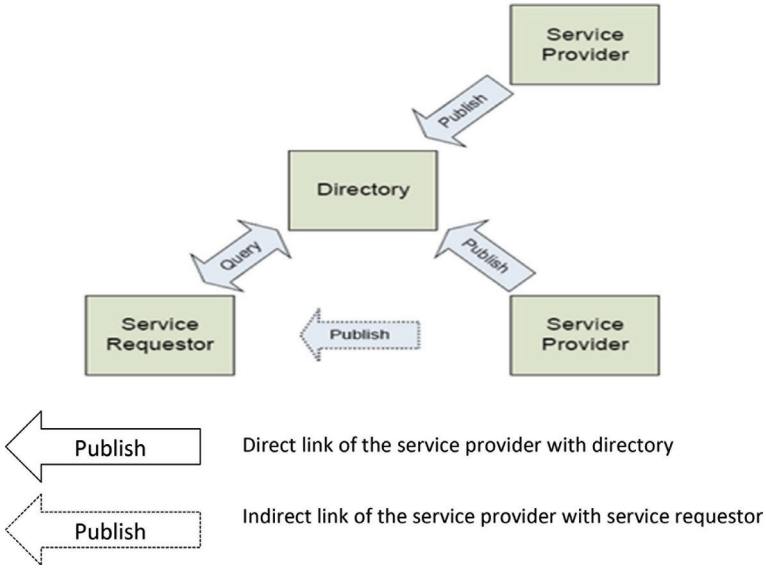

Figure 1. Service discovery generic architecture (source: [9]).

In most suitable circumstances, the exactness measures of a service discovery system would make sure that the results response clearly agrees with the intent of the user query, even with the multifariousness that may exist in the information representation and syntax of each service discovery domain. The search mechanism used determines the level of completeness and the service description expressiveness and query languages used determines the correctness [12].

Service discovery in cloud environment is made challenging by the potentially large number of available heterogeneous services and service providers. Service discovery is further complicated by the varied level of quality of service (QoS) offered by the providers. Security concern is also a big challenge while looking for an appropriate cloud service.

3. Trust

TRUST simply means act of faith that relies on confidence that something will surely be delivered as promised [13, 14]. A system is trusted less if it gives insufficient information about its expertise. Mere claims such as “secure cloud” or “trust me” do not help much to boost the trust level of consumers unless enough information is presented with the services. Establishing a trusted relationship between the service provider and the customer is one of the main issues cloud computing faces today. The potential customers are not sure whether the management of the service is trustworthy or if there is any risk of insider attacks within the service provider enclave.

Trust has been described in various fields such as psychology, sociology, and economics. Many authors have given different meanings to trust and categorized it based on these different

meaning. Researchers have focused on trust over a long time [15], trust between humans has been examined to the effects of trust in economic transactions among the social scientists [16–18]. The concept of trust can be easily understood, but it has not been seemly defined.

Control is an important factor in trust. If control over the assets, a system has cannot be ascertain then such is system cannot be trusted. Preventing a trust violation in cloud computing should be the concern and not guaranteeing compensation when it does happen. No compensation can guarantee the restoration of the lost data or the organization's reputation, a security breach of data is irredeemable. A trust model for cloud computing must thus focus substantially on preventing breach occurrence than on post-breach compensation [19].

New technology must steadily acquire its reputation for good performance and security, winning users' trust over time [19]. Trust is derived from the reputation of an entity. Based on the reputation derived, a level of trust is imparted upon an entity [20]. Hence, trust in a cloud service provider results to perceived usefulness or other positive perceptions of the cloud service provider's activities. Trust will positively have effect on the desire to use cloud services and the quality of prior experience will positively influence building trust in cloud service provider's ability. The scope of previous experience will determine the relationship between trust and intention to use cloud technology. Trust is an established factor needed for attaining success in open and dynamic cloud environment for intelligent cloud service delivery.

4. Agent-based service discovery

A multi-agent system is a distributed system having multiple software agents, which create "a loosely coupled network, called a multi-agent system (MAS), to work together to solve problems that they are not capable of handling as an entity" [21]. Multi-agent systems are a community of autonomous agents working together in order to achieve a task [22]. In MAS the individual agent shows critical insight and self-rule. There have been application of MAS technologies in addressing challenges in a number of distributed systems like scientific computing and information fusion over the years [23–25].

Despite the fact that there are various contrasts between cloud computing and multi-agent systems, both are two distributed computing frameworks, in this manner a few common issues can be distinguished and a lot more qualities can be determined by the joined utilization of cloud computing and multi-agents systems [26]. There have been a number of attempts in using agent technology in solving problems that exist in cloud computing.

Kang and Sim [27] in their research exhibited a four-stage, agent-based cloud service discovery protocol. In the protocol, negotiation agents, brokering agents and information agents were used for reinforcing the resource management system due to their scalability and adaptability attributes. In order to automate and coordinate resource management to activities self-organizing agents were used. Furthermore, the disparity in kind of the resources is basic in heterogeneous frameworks like the cloud. In the research, although MAS approach was used but the issue of trust was not considered among the cloud entities while cloud services were been discovered [27]. Also matching of the request with available service was only focused in the brokering process and not in the final service deployed for use.

Han and Sim [28] introduced a Cloud Service Discovery System (CSDS) that upheld the cloud users in finding a cloud service over the Internet. The CSDS) prototype has a search engine with three different agents namely; Query Processing Agent, Filtering Agent, and Cloud Service Reasoning Agent (CSRA). It has two components namely a CSDS which helps to discover the best cloud service on behalf of users and a cloud ontology which consists of taxonomy of definitions of different cloud services to confer with the CSRA. Added to these components was a user interface through which the user enters queries containing a service name and requirements considered by their predisposition. The CSDS with the cloud ontology outperformed the CSDS without the cloud ontology from the empirical findings. The critical reasoning made possible by cloud ontology component enabled the CSDS to be more successful in finding cloud services that matches users' preference [28]. There were partially implementation of query processing, filtering and rating functionalities but trust was not considered as a concern in the process.

An agent-based cloud system for cloud service reservation was also worked on by Gopinadh and Saravanan [29]. The system consists of an agent-based cloud service discovery approach that consults ontology to retrieve information about cloud services and carried out Price and Timeslot Negotiation (PTN) for cloud service reserved. The target was for getting cloud service that has best price and timeslot. The study contributed majorly an agent-based search engine for cloud service discovery using agent-based Price and Timeslot Negotiation mechanisms adapted for cloud service negotiation. The reasoning about the relations of cloud services, rating the search results and designing and construction of cloud ontology was enabled by the Cloud Service Reasoning Agent (CSRA). Multiple proposals in a negotiation round that generate aggregated utility, differing only in terms of individual price and time-slot utilities were made by the cloud agents [29]. Matters relating to trustworthiness of cloud services was not a focus in the study.

A testbed was presented by Sim [30] which is agent-based, the test-best supports the cloud resources discovery and Service Level Agreement (SLA) negotiation. The cloud resource management testbed has the following identified components for setting up the simulation instances (a) a collection of resource consumers, (b) a collection of physical machines, (c) a collection of cloud resources (virtualized machines), and (d) a collection of middleware comprising of provider agents and consumer agents *standing* as intermediaries between resource providers and consumers, respectively, and a collection of broker agents that links resource requests from consumers to advertisements from providers. The idea of e-commerce connection algorithm used for connecting buyers and sellers was used for transforming the cloud resource discovery process into a process of matching consumers' resource requests to resources of providers. The process entails a four-tier of selection, evaluation, filtering, and recommendation, such that a collection of broker agents matches consumers' requests to advertisements from providers. After effective matching of requests to resources, consumer and provider agents then agree for mutually acceptable resource time slots through negotiation [30]. It is observed that the study was making effort to lay foundation for cloud resource management that are agent-based. Trust issues between the consumers and providers were not considered on the testbed.

Ontology relationships were used by Talia [26] for the development of a multi agent based system in finding the equivalence between queries from the user and resources available from different service providers. The system utilizes agent approach due to their being autonomous and highly mobile for effective communication in a distributed environment. Therefore,

Requestor Agent, Mapper Agent, and Search Agent - and a search engine constitute the system. Java Agent Development Framework was used for agents' development. A concept based search using similarity reasoning from the ontology relationships of the concepts in the cloud ontology was adopted by the search engine. The average waiting time for a search was optimized as a result of multiple agents employed [26]. The possibility of finding a perfect match was enhanced due to the use of ontology.

Cloud ontology technique was utilized by Rajendran and Swamynathan [31] for efficient discovery of cloud services. In the technique, multi-broker agent that consults ontology was used in the discovery of cloud services in the process of retrieving information about services. The technique resulted to having a Cloud Service Discovery using Broker Agents (CSDBA). The approach combines the merits of the agent-based system and the semantic matching mechanism using cloud ontology notably in reasoning about the relations of cloud services. It was designed to enable consumers find the required cloud services. CSDBA was highly successful than traditional approach in finding cloud services that matches users' requirement. In the future, there is likelihood of extending cloud services brokering method as well as improved method the supports cloud crawling and rate the services based on QoS [31].

Based on the literatures reviewed, attempting to use MAS in cloud service discovery has been ongoing but adequate attention was not given to issues on trustworthiness of cloud service in the process tackling the issue will greatly improve the process. This work is depending on the capabilities of the MAS to extend the discovery system to ensure that trustworthy cloud services are discovered for use by the potential users. Adding the multi-agent paradigm for trustworthy cloud service discovery will enable efficient cloud service discovery and enhance the process.

5. System design

Functional Analysis was used in the conceptual design for a trustworthy cloud service discovery system. Functional Analysis is a crucial tool for the design of process to explore new concepts and to define their architectures in system engineering [32]. The functional requirements for a trustworthy cloud service discovery system was define using this tool. It made sure that all necessary components were added and unrequired components were exempted. The interdependency among the system components were also determined and stated. The major steps of Functional Analysis were made up of the functional tree, the functions/devices matrix and the product tree. Starting from the functional tree and in particular by way of identifying the basic functions, the system's functional requirements were defined, and the building blocks of the system were determined using the product tree, thus this give rise to the system's functional architecture. Following the Functional Analysis approach, the building block for trustworthy cloud service discovery using multi-agent approach was identified. This led to the new proposed cloud service discovery system, and their integration to build up its functional framework. Therefore the following fundamental subsystems are required to achieve efficient cloud service discovery namely: Cloud User subsystem, Cloud Provider subsystem, Trust Evaluator subsystem, Broker subsystem and Database as depicted in **Figure 2**. The various identified subsystems must work together effectively for realizing a cloud service discovery system that is trustworthy.

5.1. System architecture

The distributed system architecture for cloud service discovery was formed from the integration of the different basic components of the functional block [33]. The architecture is three tiered as shown in **Figure 3**.

i. Application and resource layer

ii. Agent layer

iii. Database layer

5.1.1. Application and resource layer

The layer is the aggregation of all cloud services available and it provides the interface for connecting to the various cloud resources and services. The cloud services are categorized based on the basic cloud service models. A cloud service is any resource that is made available via the Internet. The most prevalent cloud service resources are Software as a Service (SaaS), Platform as a Service (PaaS) and Infrastructure as a Service (IaaS). SaaS is a form of software accessible model where applications are hosted by a vendor or service provider and made available to customers over a network, typically the Internet. PaaS has to do with the delivery of operating systems and associated services over the Internet without downloads or installation. IaaS entails outsourcing the equipment used to support operations, including storage, hardware, servers and networking components, all of which are made accessible over a network.

5.1.2. Agent layer

The system was constructed based on the multi-agent approach for ensuring effective and reliable discovery. The agents are responsible for achieving intelligence in the system, such that it is more adaptive, flexible, and autonomic in its operation.

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).

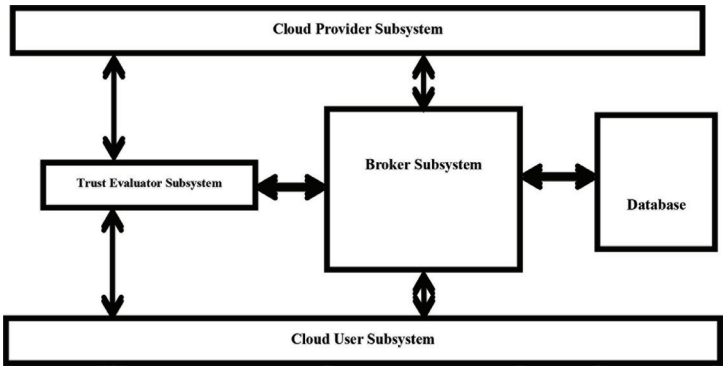

Figure 2. Functional block diagram for cloud service discovery [32].

http://dx.doi.org/10.5772/intechopen.84291

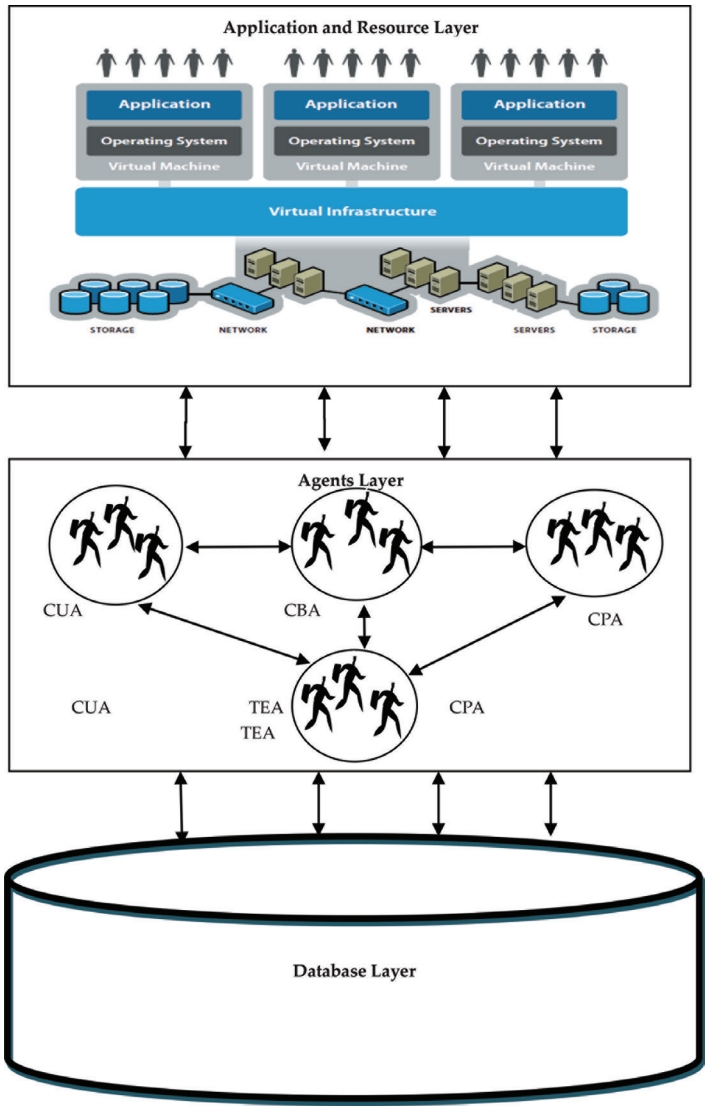

Figure 3. System architecture [32].

These autonomous agents with their specified functions establish appropriate link among one another for efficient cloud service discovery. In achieving this, a peer-to-peer approach is used among the different agents for the discovery. The cooperating 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 scenario. Interaction may spread among all the peers, and peers can communicate directly with each other, and hence, they are constantly aware of each other, or they may communicate indirectly through other peers for effective discovery.

5.1.2.1. Agents functions

The functions of the various agents required for ensuring a trustworthy cloud service discovery are as follows [33]:

5.1.2.1.1. User agent

The user agent accepts the user's service request and the send the request to broker agents using message passing and displays the results to user through a user interface. User agent interacts with provider agent through the broker agents for service usage. Likewise the user agent interacts with the trust evaluator agent to keep track of the performance value of the provider in terms of service type, service integrity and service accessibility in a bid to build trust.

5.1.2.1.1.1. Algorithm for User agent

1. Begin

2. Let H=total number of users request within a specified time

3. Do

4. For j=1,j++,H

5. User_Agent ←User's request: UR_j

6. Broker_Agent for UR_j discovery ←User_Agent

7. Wait for Broker_Agent

8. User_Agent ←Broker_Agent returns Result of UR_j discovery

9. User Interface ←User_Agent Result for use

10. If (UR_j is used) Then

11. User_Agent interacts with Trust_Evaluator_Agent on UR_j

12. End if

13. Next j

14. While(User's request≠∅)

15. Stop

5.1.2.1.2. Broker agent

The analysis of the user's request in order to determine its type is done by the broker agent. It attempts to find a close match for the identified service request and sends the list of such services to the user agent. It forwards user-provider's interaction information to the database. It also confers with database for other broker agent recommendation. The broker agent gets trust value updates from trust evaluator agent.

5.1.2.1.2.1. User agent and Broker agent interaction algorithm

1. Begin

2. If (*Broker_Agent received User_Agent Message*) Then

3. Broker_Agent Analyse UR_j

4. Search the database for a match

5. If (UR_j *match is found*) then

6. Broker_Agent ←UR_{jmatch}

7. Else

8. Other_Broker_Agents ←UR_i

9. If (UR_j *match is found by Other_Broker_Agents*) Then

10. Broker_Agent ←UR_{jmatch}

11. Else

12. Broker_Agent ←UR_{jmiss}

13. End if

14. End if

15. User_Agent ←Broker_Agent

16. End if

17. Stop

5.1.2.1.2.2. Broker agent and Trust evaluator agent interaction algorithm

1. Begin

2. Do

3. Broker_Agent Interact with Trust_Evaluator_ Agent

4. Broker_Agent ← Trust_Evaluator_ Agent (trust value updates)

5. Database ←Broker_Agent(Stores user agent – provider agent interaction information)

6. While (*a typical service is in use*)

7. Stop

5.1.2.1.3. Trust evaluator agent

It follows and 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 availability and accessibility by interacting with user and provider agents. It determines the trustworthiness of cloud service provider. It has a table for holding the performance of the provider and updates it when new performance value is available.

5.1.2.1.3.1. Algorithm

1. Begin

2. Do

3. Trust_Evaluator_ Agent: Monitors provider's performance

4. Trust_Evaluator_ Agent: Computes the trustworthiness of providers

5. Broker_Agent ← Trust_Evaluator_ Agent (sends trust value updates)

6. Trust_Evaluator_ Agent: Stores the providers' performance

7. While (*User_Agent Interacts with Provider_Agent*)

Stop

5.1.2.1.4. Provider agent

At each cloud service provider site, provider agents are available. They send advertisement, removal and updates information of cloud services to the broker agent. It makes services available for use by the user agent.

5.1.2.1.4.1. Algorithm

1. Begin

2. Do

3. Provider_Agent : Create service instance

4. Broker_Agent ← Provider_Agent (Sends service availability information)

5. Broker_Agent ← Provider_Agent (Sends service expire information)

6. Broker_Agent ← Provider_Agent (Sends service update information)

7. User_Agent ← Provider_Agent (Offers service)

8. While (*a service is being offered by service provider)*

9. Stop

The agents' operations class diagram and sequence diagram are depicted in **Figure 4** and **5** respectively.

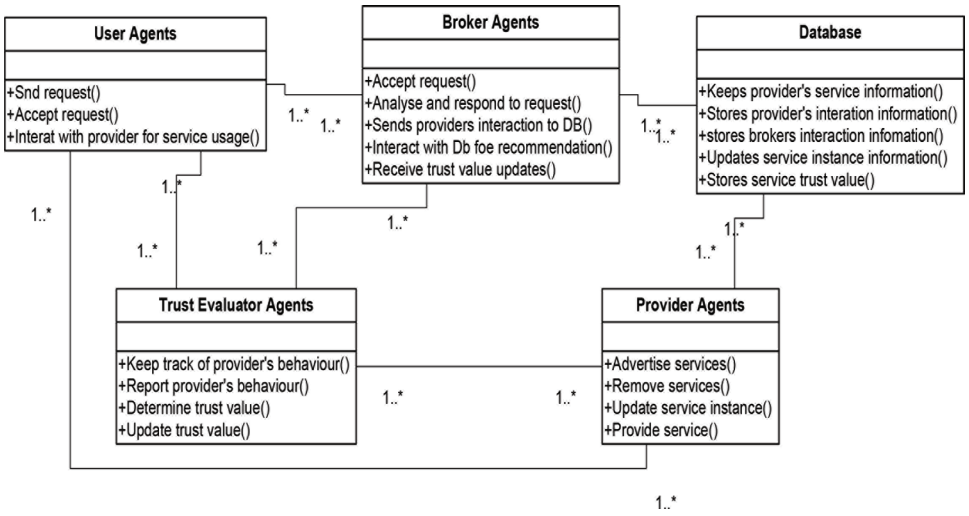

Figure 4. Agents' operations class diagram.

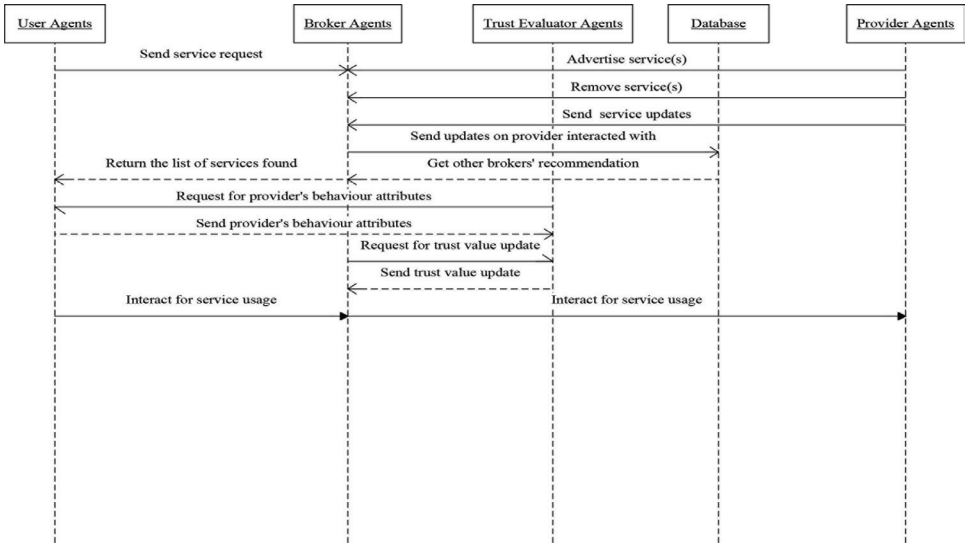

Figure 5. Agents' operations sequence diagram.

5.2. Database layer

The database layer holds the details of service usage and agents interaction information. It contains service information and trust value rating for providers. It also maintains the broker's interaction information. In a basic cloud service discovery system, user submit their

request to the broker. The broker confer with the registry or directory to find a possible match for the user's request. The provider publishes the instances of its available services inside the service registry or service directory and from time to time gives updates of the status of the published service [33].

6. Experiments setups, outcomes and discussions

Simulation experiments were carried out to unravel the impact of extending MAS approach to achieve a trustworthy cloud service discovery using response time and scalability as the performance metrics [33]. The response time is the time lag from the instant of service request by the users to the instant of having the result of the service requested. This is dependent on round trip time (RTT) and the users load (UL). Scalability is a factor that considers how the system can be easily adjusted in such a way to accept changes in the number of users, resources and computing entities attached to it. Scalability (S) is dependent on processing capability of the provider which is define in terms of the processing time (PT) and users load (UL).

The experiment considered the cloud service discovery transactions conducted without involving trust factor against the same kind of cloud service discovery transaction involving trust factor in order to investigate their difference. CloudAnalyst [34] was used to model and analyze large scale cloud computing buildup in the simulations conducted. Considering the nature of throttled load balancing policy of the CloudAnalyst, the integration of trust capability in cloud environment was achieved with this policy while the round-robin load balancing policy, was used for specifying cloud transaction without trust integration [33].

The first experiment conducted depicted cloud service discovery scenario using CloudAnalyst. In this instance, there were 5,10,15,20, 25 and 30 identified groups of users agents in addition to diversified cloud provider agents and the user agent growth factor varied from 100 to 10,000,000. These units of user agents were created randomly using Poisson distribution by changing the number of users in a realistic manner. The user agents represent the potential users attempting to find cloud service that meet their requirement while the provider agents represent the various cloud service providers with their different cloud service models. The detailed configuration for the experiment were shown in **Table 1**. 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 previous system without trust integration and proposed system with trust integration in respect of the response time.

Table 2 depicts the results obtained from the simulation experiment. The graph shown in **Figure 6** has the plot of the round trip time against the number of user agents to ascertain the behaviour of the proposed and previous systems in respect of response time. From the results, the previous system has higher response time while the proposed system has lower and refined stable response time. The percentage difference of the systems' response time shown in **Table 2** attested to this. The experiment revealed that the proposed system has lower response time of as a result of trust integration.

The second simulation experiment was also conducted for cloud service discovery transaction using CloudAnalyst. In the simulation scenario there were 5, 10, 15, 20, 25 and 30 groups of

User agents	Provider agent	User growth factor	Request growth factor	Execution instruction per length
5	1	100	10	100
10	2	1000	10	100
15	3	10,000	10	100
20	4	100,000	10	100
25	5	1,000,000	10	100
30	6	10,000,000	10	100

Table 1. Simulation parameter configurations for varied number of user agents' growth factor and varied number of provider agents in response time test.

User agents growth factor	Round trip time (ms) (existing system)	Round trip time (ms) (proposed system)	Percentage difference (%)
100	302.12	300.81	−0.434
1000	302.42	300.19	−0.737
10,000	303.10	301.15	−0.643
100,000	303.10	301.08	−0.666
1,000,000	303.71	301.13	−0.849
10,000,000	303.49	301.40	−0.689

Table 2. Results of varied number of user agents' growth factor and varied number of provider agents for response time test.

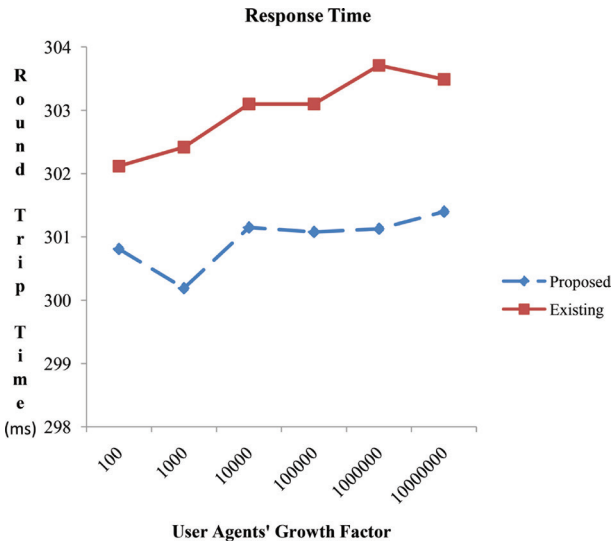

Figure 6. Varied number of user agents' growth factor and varied number of provider agents in response time experiment.

user agents with varied number cloud provider agents having a capacity of 5 virtual machines each. Similarly, the user agents represent the potential users attempting to find cloud service that meet their requirement while the provider agents represent the various cloud service providers with their different cloud service models. The user agents' growth factor was varied from 100 to 10,000,000. **Table 3** contains parameter configuration for the experiment.

In this experiment, the value for user agent growth factor was varied as well as the population of the provider agent with a view to determine the behaviour of the previous system without trust integration and proposed system with trust integration for scalability test.

Table 4 shows the outcomes from the experiment conducted. The graph in **Figure 7** depicts the plot of the provider agent processing time against the number of user agents' growth factor to verify the behaviour of the proposed and previous systems in terms of scalability. From the outcomes, the trust integration in the proposed system aid it to scale the request of user agents reasonably better than the previous system as the population of user agents moves up and the processing capability of each of the provider agents stays firm. The percentage difference of the systems' scalability metric as shown in **Table 4** evidently attested to this.

User agents	Provider agents	Number of virtual machines	User growth factor	Request growth factor	Execution instruction per length
5	1	5	100	10	100
10	2	5	1000	10	100
15	3	5	10,000	10	100
20	4	5	100,000	10	100
25	5	5	1,000,000	10	100
30	6	5	10,000,000	10	100

Table 3. Simulation parameter configuration for varied number of user agents' growth factor and varied number of provider agents in scalability test.

User agents growth factor	Provider agent processing time (ms) (existing system)	Provider agent processing time (ms) (proposed system)	Percentage difference (%)
100	1.91	0.59	−69.110
1000	2.60	0.38	−85.385
10,000	2.87	0.92	−67.944
100,000	2.98	0.95	−68.121
1,000,000	2.95	0.37	−87.458
10,000,000	3.06	0.97	−68.301

Table 4. Varied number of user agents' growth factor and varied number of provider agents for scalability.

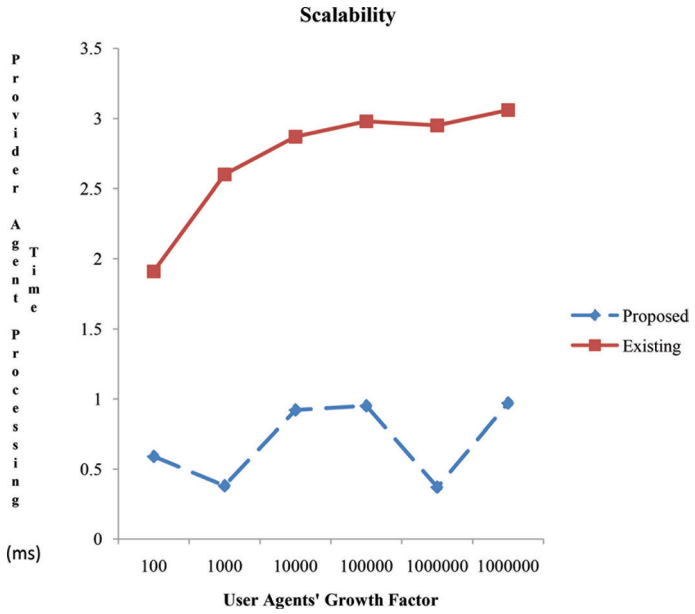

Figure 7. Varied number of user agents' growth factor and varied number of provider agents for scalability.

6.1. Validation of simulation experiment results

From the experiments, the proposed system having trust integration outperformed the previous one without trust. The results validation was achieved by applying paired difference *t* –test to the percentage difference of the response time for the proposed and the previous systems with each having six group sizes in the simulation configuration. The Null hypothesis opines that there is no significant difference between the proposed and previous system but the alternative hypothesis states that there is significant difference between them.

The calculated *t* values for the response time in the simulation done resulted to −70.43. The total group size of six (6) has a degree of freedom of five (5). A table t value of 2.02 (one-tailed, a significance level (α) of 0.05) using 5 degrees of freedom (df), at 95% confidence interval was obtained. Comparing the absolute calculated *t* value to the tabled *t* showed that the proposed system which has lower response time is significantly different (p = 0.05) from the previous system owning to trust integration in it.

Carrying out 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 previous systems, resulted to the calculated *t* values of −3.09. Using a *t* table with 5 degrees of freedom (df) and confidence level of 95%, the table t value is 2.02 (one-tailed, a significance level (α) of 0.05). It is observed that absolute calculated *t* value of 3.09 was more than the table *t* value revealing that the proposed system is more scalable and significantly different (p = 0.05) from the previous system due to trust integration in it.

7. Comparative analysis

The MAS implementation for cloud service discovery supports the interconnection and interoperation of the autonomous interacting agents in the cloud environment. The MAS provides for efficient retrieving and filtering of trustworthy cloud services based on user's preference. **Table 5** gives a summary of the comparative analysis of the monolithic approach to cloud service discovery and the distributive approach using MAS.

S/n	Comparison factor	Monolithic	Distributive
1.	Computational efficiency	Less efficient	Highly efficient
2.	Reliability	Less reliable	Highly reliable
3.	Extensibility	Difficult to extend	Easy to extend
4.	Robustness	Less robust	Highly robust
5.	Maintainability	Difficult to maintain	Easy to robust
6.	Responsiveness	Slow response	Quick response
7.	Flexibility	Less flexible	Highly flexible
8.	Reusability	Difficult to reuse	Easy to reuse

Table 5. Comparative analysis of the monolithic approach to cloud service discovery and the distributive approach using MAS.

8. Conclusion

The cloud computing is a form of distributed system paradigm 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. Accessing 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 and laborious task. Trust is a predicated factor for achieving success in open and dynamic cloud environment for intelligent cloud service delivery. A distributive approach using MAS for trustworthy cloud service discovery was proposed in this study. The experiments were carried out and the results indicated that extending the MAS approach into cloud service discovery by way of integrating trust into the process improves the quality of service in respect of response time and scalability. The research showed that using multi-agent systems in the cloud environment aids intelligent cloud service discovery despite the nature of the cloud environment. Comparing MAS approach for cloud service discovery against the monolithic approach of cloud service discovery revealed that the approach is efficient and ensures flexibility for trustworthy cloud service discovery. It has capability to provide reliable result in a heterogeneous space where resources are spread whereas a monolithic approach is not

efficient and reliable for a distributed environment such as cloud. The result of this work can be integrated into the interface between cloud consumer and cloud provider while finding trustworthy cloud services.

Author details

Akinwale Akinwunmi[1]*, Emmanuel Olajubu[2] and Ganiyu Aderounmu[2]

*Address all correspondence to: akinwale.akinwunmi@bowenuniversity.edu.ng

1 Computer Science and Information Technology Department, Bowen University, Iwo, Nigeria

2 Computer Science and Engineering Department, Obafemi Awolowo University, Ile-Ife, Nigeria

References

[1] Buyya R, Yeo CS, Venugopal S. Market-oriented cloud computing: Vision, hype, and reality for delivering IT services as computing utilities. In: Proceedings of the 10th IEEE International Conference on High Performance Computing and Communications (HPCC 2008). Dalian, China; 2008. DOI: 10.1109/HPCC.2008.172

[2] Vaquero LM, Rodero-Merino L, Caceres J, Lindner M. A break in the clouds: Towards a cloud definition. ACM SIGCOMM Computer Communication Review. 2009;**39**(1):50-55. DOI: 10.1145/1496091.1496100

[3] Mei L, Chan WK, Tse TH. A tale of clouds: Paradigm comparisons and some thoughts on research issues. In: Proceedings of the 2008 IEEE Asia-Pacific Services Computing Conference (APSCC 2008), 2008, IEEE Computer Society Press, Los Alamitos, CA. DOI: 10.1109/APSCC.2008.168

[4] AO A, EA O, GA A. A trustworthy model for reliable cloud service discovery. International Journal of Computer Applications. 2014;***87***(16):23-30. DOI: 10.5120/15293-3962

[5] Mirzaei N. 2008. Cloud Computing, Indiana University. Available from: http://grids.ucs.indiana.edu/ptliupages/publications/ReportNarimanMirzaeiJan09.pdf [Accessed on: September 20, 2012]

[6] Cortázar GO, Zapater JJS, Sánchez FG. Adding semantics to cloud computing to enhance service discovery and access. In: Proceedings of 2012 6th Euro American Conference of Telematics and Information Systems (EATIS); 23-25 May 2012. Valencia, Spain: IEEE; 2012. pp. 1-6. DOI: 10.1145/2261605.2261639

[7] Calheiros RN, Ranjan R, Beloglazov A, De Rose CAF, Buyya R. CloudSim: A toolkit for Modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms. Software: Practice and Experience. 2011;**41**:23-50. DOI: 10.1002/spe.995

[8] World Economic Forum .2010. Exploring the future of cloud computing: Riding the next wave of technology-driven transformation. Report of World Economic Forum in Partnership with Accenture; Geneva, Switzerland

[9] Kousiouris G, Kyriazis D, Varvarigou T, Oliveros E, Mandic P. Taxonomy and state of the art of service discovery mechanisms and their relation to the cloud computing stack. In: Kyriazis D, Varvarigou T, Konstanteli K, editors. Achieving Real-Time in Distributed Computing: From Grids to Clouds. Hershey, PA: Information Science Reference; 2012. pp. 75-93. DOI: 10.4018/978-1-4666-0879-5.ch805

[10] Meshkova E, Riihijarvi J, Petrova M, Mahonen P. A survey on resource discovery mechanisms, peer-to-peer and service discovery frameworks. Computer Networks. 2008;**52**(11): 2097-2128. DOI: 10.1016/j.comnet.2008.03.006

[11] Dastjerdi AV, Buyya R. An autonomous reliability-aware negotiation strategy for cloud computing environments. In: 2012 12th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (Ccgrid 2012) 13-16 May 2012. Ottawa, ON, Canada: IEEE; 2012. DOI: 10.1109/CCGrid.2012.101

[12] Ahmed R, Limam N, Xiao J, Iraqi Y, Boutaba R. Resource and service discovery in large-scale multi-domain networks. IEEE Communications Surveys & Tutorials. 2007;**9**(4): 2-30. DOI: 10.1109/COMST.2007.4444748

[13] Costa C, Bijlsma-Frankema K. Trust and control interrelations. Group & Organization Management. 2007;**32**(4):392-406. DOI: 10.1177/1059601106293871

[14] Lund M, Solhaug B. Evolution in relation to risk and trust management. Computer. 2010;**43**(5):49-55. DOI: 10.1109/MC.2010.134

[15] McKnight DH, Chervany NL. The Meanings of Trust. MIS research Center, Carlson School of Management, University of Minnesota. Working Paper Series (WP 96-01); 1996

[16] Ba S, Paylou PA. Evidence of the effect of trust building Technology in Electronic Markets: Price premiums and buyer behaviour. MIS Quarterly. 2002;**26**:243-268. DOI: 10.2307/4132332

[17] Dasgupta P. Trust as a commodity. In: Gambetta D, editor. Trust: Making and Breaking Cooperative Relations. Blackwell; 1988. pp. 49-72

[18] McKnight DH, Cummings LL, Chervany NL. Trust formation in new Organaizational relationships. In: October 1995 Information & Decision Sciences Workshop; Curtis L. Carlson School of Management, University of Minnesota; 1995

[19] Khan KM, Malluhi Q. Establishing Trust in Cloud Computing. IT Professional. 2010;**12**(5): 20-27. DOI: 10.1109/MITP.2010.128

[20] Momani M, Abour K, Challa S. RBATMWSN: Recursive bayesian approach to trust management in wireless sensor networks. In: In 2007 3rd International Conference on Intelligent Sensors, Sensor Networks and Information Processing(ISSNIP 2007); 3-6 December 2007. Melbourne, Australia: IEEE; 2008. pp. 347-352. DOI: 10.1109/ISSNIP.2007.4496868

[21] Durfee EH, Montogomery TA. MICE: A flexible testbed for intelligent coordination experiments. In: Proceedings of the 1989 Distributed Artificial Intelligence Workshop. 1989. pp. 25-40. DOI: 10.1.1.16.8714

[22] Wooldridge M. An Introduction to MultiAgent Systems. 2nd ed. Chichester: John Wiley and Sons; 2009

[23] Drashansky T, Houstis EN, Ramakrishnan N, Rice JR. Networked agents for scientific computing. Communications of the ACM. 1999;**42**(3):48-53. DOI: 10.1145/295685.295699

[24] Khosla R, Dillon T. Intelligent hybrid multi-agent architecture for engineering complex systems. In: 1997 Proceedings of the IEEE International Conference on Neural Networks (ICNN'97); 12-12 June 1997. Houston, Texas: IEEE; 1997. pp. 2449-2454. DOI: 10.1109/ICNN.1997.614540

[25] Honavar V, Miller L, Wong J. Distributed knowledge networks. In: Proceedings of 1998 IEEE Information Technology Conference, Information Environment for the Future (Cat. No.98EX228); 3-3 Sept. 1998. Syracuse, NY, USA: USA IEEE; 1998. pp. 87-90. DOI: 10.1109/IT.1998.713388

[26] Talia D. Cloud computing and software agents: Towards cloud intelligent services. In: Fortino G, Garro A, Palopoli L, Russo W, Spezzano G, editors. Proceedings of the 12th Workshop on Objects and Agents (WOA-2011). Italy: Rende (CS); July 4-6, 2011

[27] Kang J, Towards Agents SKM. Ontology for cloud service discovery. In: 2011 International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery; 10-12 October 2011. Beijing, China: IEEE; 2011. pp. 483-490. DOI: 10.1109/CyberC.2011.84

[28] Han T, Sim KM. An ontology–Enhanced cloud service discovery system. In: Proceedings of International Multiconference of Engineers and Computer Scientists (IMECS, 2010); 17-19 March 2010. Hong Kong: IMECS; 2010. pp. 644-649

[29] Gopinadh R, Saravanan K. Cloud service reservation using PTN mechanism in ontology enhanced agent-based system. International Journal of Engineering Trends and Technology. 2013;**4**(4):791-798. ISSN:2231-5381

[30] Sim KM. Agent-based cloud commerce. In: Proceedings of 2009 IEEE International Conference on Industrial Engineering and Engineering Management; 8-11 December. Hong Kong, China: IEEE; 2009. pp. 717-721. DOI: 10.1109/IEEM.2009.5373228

[31] Rajendran V, Swamynathan SA. Novel approach for semantic service discovery in cloud using broker agents. In: Proceedings of International Conference on Advances in Computing, Communication and Information Science (ACCIS-14); June 2014. Kollam, Kerala, India: Elsevier Publications; 2014. pp. 242-250. DOI: 10.13140/2.1.4022.6085

[32] Viola N, Corpino S, Fioriti M, Stesina F. Functional analysis in systems engineering: Methodology and applications. In: Cogan B, editor. Systems Engineering–Practice and Theory. Rijeka, Croatia: InTechOpen; 2012. pp. 3-29. DOI: 10.5772/34556

[33] Akinwunmi A, Olajub EA, Aderounmu GA. A multi-agent system approach for trustworthy cloud service discovery. Cogent Engineering. 2016;**3**(1):1-17. DOI: org/10.1080/23311916.2016.1256084

[34] Wickremasinghe B, Calheiros RN, Buyya R. CloudAnalyst: A cloudsim-based visual modeller for analysing cloud computing environments and applications. In: 2010 24th IEEE International Conference on Advanced Information Networking and Applications (AINA); 20-23 April 2010. Perth, WA, Australia: IEEE; 2010. pp. 446-452. DOI: 10.1109/AINA.2010.32