ISSN (Online) : 2278-1021
ISSN (Print) : 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 3, Issue 1, January 2014

Decentralized Service Discovery Approach Using Dynamic Virtual Server

N.Aravindhu[1], C.Shalini[2], R.Jayalakshmi[3], S.Priyavadhani[4]

Assistant Professor, Department of Computer Science, Christ College of Eng. and Technology, Puducherry, India[1]

B.Tech, Final Year, Department of Computer Science, Christ College of Eng. and Technology, Puducherry, India[2,3,4]

Abstract: Service oriented computing (SOC) is upcoming model for developing distributed applications. Earlier service discovery methods using centralized service registries can suffer from problems such as bottleneck and vulnerability. The existing system uses the chord4s to overcome the above problems but it decreases the performance of the system. Chord4s also leads to low data availability. To overcome the chord4s problems, this paper proposes a decentralized service discovery approach using dynamic virtual server. We improve the data availability, performance of the system by using Euclid algorithm and also reduce the requested time of the system by using free count. Free count is generated by the Euclid algorithm in the application server. The main aim of our project is to improve the availability of service descriptions by distributing services to different successor nodes.

Keywords: decentralized, service oriented computing.

I.INTRODUCTION

Web service

A Web service is a way of interaction between two electronic devices on the World Wide Web. The W3C describes that "Web service is a software system intended to maintain interoperable machine-to-machine communication over a network". Other systems act together with the Web service in a way given by its description via SOAP messages, usually suggested using HTTP with an XML serialization in combination with other Web-related standards. The Web Service architecture places into association among different apparatus and technologies. It includes a Web Services "stack" or entirely functional implementation. The fundamental architecture includes Web services technologies capable of:

1. Swapping messages.
2. Relating Web Services.
3. Distribute and detect Web Services descriptions.

The essential Web Services architecture defines an interface among software mediator as an exchange of messages between services requesters and service providers [1]. Requesters are software mediator that requests the implementation of service. Providers are software mediator that offers a service. Service requesters and service providers are mediators. Providers are liable for the publishing a description of the services they provide. Requesters should be capable to obtain the descriptions of the service. The Web Services architecture is based on the communications between three tasks: service provider, service registry and service requester. The communication involves publish, find and bind operations. The service provider defines a service description for the Web Service and publishes it to a service requester or service registry. The service requester utilizes the find operation to regain the service description locally or from the service registry and make use of the service description to connect with the service provider and evoke with the Web Service execution. Service provider and service requester tasks are logical constructs and a service can display characteristics of both, is shown Fig 1.

Roles

The most common roles used in the field of web services are in the following.

1. Service provider from a business view, this is the holder of the service. From an architectural view, this is the platform that hosts contact toward the service.

2. Service requester from a business view, this is the business that needs certain tasks to be satisfied. From an architectural point of view, this is the use that is appearing for and invoking or initiating an interface with a service. The service requester function can be occupied by a browser driven by a human being or a agenda without a user interface, for instance an additional Web Service.

ISSN (Online) : 2278-1021
ISSN (Print) : 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 3, Issue 1, January 2014

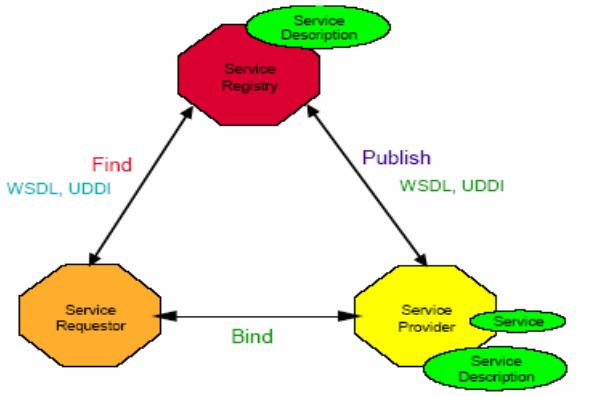


Fig. 1

3. Service registry this is a lookup registry of service descriptions where service providers distribute their service descriptions. Service requesters locate services and achieve required information for services through expansion for fixed binding or through execution for dynamic binding. For fixed bound service requesters, the service registry is an elective function in the design, as a service provider can throw the description straightly to service requesters. Likewise, service requesters can acquire a service description from other sources.

Operations

For a function to obtain benefit of Web Services, three actions should takes place: publication of service descriptions, discovery of service descriptions, and binding or using of services depending upon the service description. These actions can happen individually continuously. In fact, these actions are:

- Publish to be available; a service description desires to be available thus the service requester can establish it. Where it is published can differ depending upon the necessities of the function

- Discover in the find operation, the service requester reclaims a service description frankly or questions the service registry for the kind of service needed.

- Bind eventually, a service wants to be raised. In the bind Process the service requester raises or initiates an communication by means of the service at runtime with the binding information's in the service description to place, contact and call up the service.

- Service where a Web Service is an interface explained by a Service description, its execution is the service. A service is a software component organized on network available platforms offered by the service provider.

- Service Description the service description encloses the facts of the interface and execution of the service. This contains its data types, operations, binding information and system position. It could also consist of categorization and additional metadata to ease discovery and deployment through service requesters. The service description may be available towards a service requester or to a service registry.

Advantages

Web Services propose various benefits over additional types of distributed computing architectures.

- Interoperability
- Usability
- Reusability
- Deploy ability

II.RELATED WORK

The main point of our project is to offer a scalable, trustworthy, and vigorous method used for web service discovery.

Centralized Service Discovery

The centralized client/server paradigm has been applied for service discovery at what time SOC approach come into view. UDDI [1] is one of the important common discovery methods for web services. To give solution designed for application and service combination, currently the software merchant utilize UDDI as a basis feature for their software. Centralized services endure from meager performance due to elevated scalability. Dimensions have been taken to explain the difficulty by using UDDI registries. Wu et al. [2] depict an interoperable representation of disseminated UDDI registries. The instigator approves the philosophy of Domain method Name System (DNS). Super domain servers, supervised by a root server, are used to retain ordinary servers. While the representation try to be like DNS, it is immobile faces the same crisis that DNS faces, e.g., Distributed Repudiation of Service attacks. A Web Service Crawler Engine is recommended to tackle the implementation problem produced by using many number of UDDI registries [4]. The engine crawls available UDDI registries and gathers information in a centralized depository via which service customers can resourcefully discover required web services. The proposed approach will increase the effectiveness of web service discovery by using dynamic server.

Decentralized Service Discovery

To solve the troubles that are basis by centralized services, the decentralized service discovery approach is used.. To maintain equilibrium the weight on peer nodes, a multilayer P2P cover network is build to combined similar indexing keys. Similar keys are introduced into the same Service Bag to improve the Service Index. In this method, the loss of a Service Bag will direct to the omitted of all the keys in the Service Bag, rigorously expose the total accessibility of the keys. Web Services Dynamic Discovery is a set of rules to establish services

ISSN (Online) : 2278-1021
ISSN (Print) : 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 3, Issue 1, January 2014

on a narrow network, is urbanized by BEP Systems, Canon, Intel, Microsoft and Web Methods. In WS-Discovery, a client transmits a request to the equivalent multicast cluster to locate a objective service. In [4], Hu and Seneviratne put forward that service providers themselves should take the duty to continue their specific service images in a decentralized manner. Based on this notion, a decentralized service directory communications is put together with hashing expressive strings into the identifiers. By responsibility, peer nodes are clustered by service group to develop islands on the Chord circle. Chart and Native Table are formed on each peer node to run routing diagonally in islands and surrounded by islands respectively. Li et al. [1] present PSWD, distributed web service discovery architecture establish on an extensive Chord algorithm called XChord.The essential P2P routing algorithm of Chord is extensive with XML to allow XML-based complex queries. Schmidt and Parashar [2] describe an approach that implements an Internet-scale DHT. M-Chord [2] uses iDistance [5] to change the metric search problem with the help of a period search problem in one dimension. It offers Chord with the capability to achieve metric-based parallel search. We see that none of the exceeding has addressed the problem of data availability in open and unstable SOC environments. Node failures would direct to stern data loss when the above method are accepted to assist service discovery because images of functionally equal services would be accumulate at the equivalent successor nodes. The research describes in this paper is related to the work presented in [4] by layered service identifiers to organize the distribution of service descriptions. The projected model chains effective service discovery using dynamic virtual server.

III.EXISTING SYSTEM

To overcome the problems in centralized registries, a decentralized service discovery approach named chord4s is used. Chord4s, a chord established decentralized service discovery method that chains service description and discovery in a P2P method. Chord is chosen for the reason that it is fine recognized for its elasticity and scalability. It is appropriate particularly in large range of service oriented computing backgrounds.Chord4s takes the benefits of the fundamental principles of chord for node organization, data distribution and query routing. Data availability is more increased by allocating published descriptions of functionally parallel services to dissimilar successor nodes that are ordered into virtual section in the Chord4S circle. In case one node crash, a service consumer is even now capable of discovering functionally parallel services that are accumulated at further successor nodes.

Disadvantages of existing system

- In chord 4s the requested time is more.
- Data availability is less.
- SHA-1 algorithm is the oldest secure hash algorithm and it is similar to SHA-0

IV. PROPOSED WORK

- To overcome the problems in chord4s approach, we are going to create dynamic virtual server.
- Dynamic server reduces the requested time of the server.
- It also increases the data availability of the server
- Euclid algorithm is used to find the GCD of the server to know which server is free.
- We can also improve the performance of the system using the dynamic server.

Advantages of proposed system

- Performance is high when compared with existing system.
- Waiting time is reduced.
- Achieves high data availability.

Generate Euclid algorithm

The Euclidean algorithm estimate the greatest common divisor (GCD) of two ordinary numbers *a* and *b*.

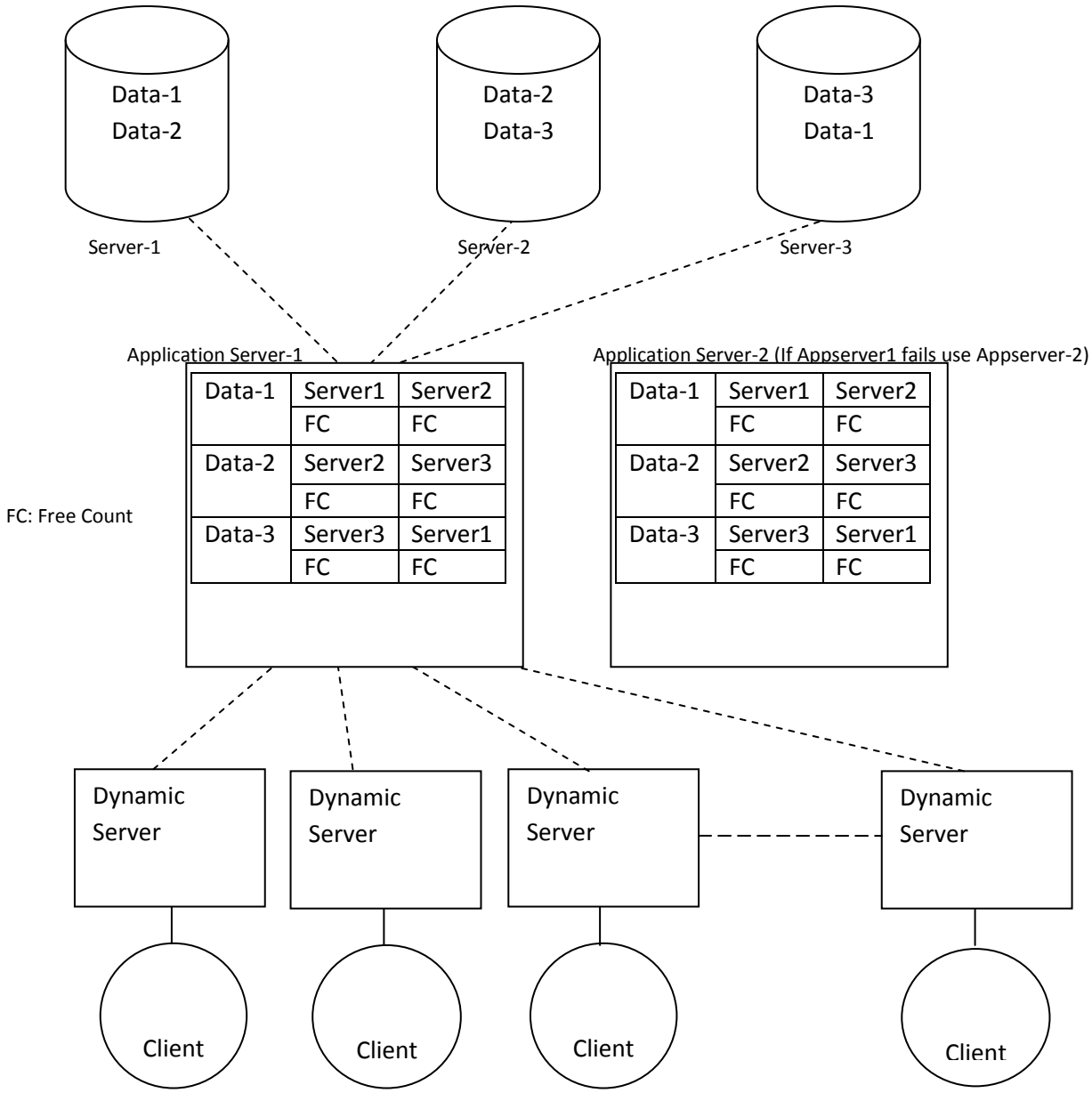
METHOD: comparing and checking the free count.
INPUT: To generate Euclid algorithm.
OUTPUT: greatest common factor is produced.

```
/*** Returns the greatest common divisor of the given numbers "a" and "b"
* @param a number "a"
* @param b number "b"
* @return gcd(a, b)
* @autor Thomas (www.adamjak.net)
*/
public static int gcd(int a, int b)
{
if (a < 1 || b < 1)
{
throw new Argument Exception("a or b is less than 1");
}
int r = 0;
do
{
r = a % b;
a = b;
b = r;}
while (b != 0);
return a;
}
```

ISSN (Online) : 2278-1021
ISSN (Print) : 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 3, Issue 1, January 2014

Fig. 2 Block diagram of proposed system



Data-1	Server1	Server2
	FC	FC
Data-2	Server2	Server3
	FC	FC
Data-3	Server3	Server1
	FC	FC

Web service request module

In this module we create a client side web service request part. The client sends the request to dynamic server and it is the permanent connection. Dynamic server would then contact with the application server and sends a service request. The IIS (Internet Information Server) creates dynamic server for each request

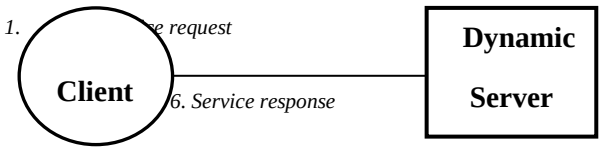


Fig. 3

Service provider description module

In this module the client request is proceed to the application server. The application server contains the details of all database servers and it sees whether the server is busy or not. Here create FC (Free count) which uses the Euclid algorithm to find the greatest common factor and compares the free count of the servers.

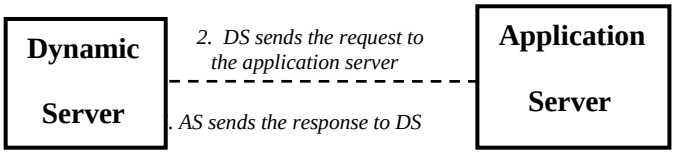


Fig. 4

Dynamic service discovery module

In this module after finding which server should handle the request of the client and sends the response to the

ISSN (Online) : 2278-1021
ISSN (Print) : 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 3, Issue 1, January 2014

application server and the connection is stopped after the response send to the application server. The application server sends the response to the dynamic server and it is disconnected as soon as the data send to the dynamic server. The dynamic server gives the response to the client and it is a permanent connection from starting of the process till the end of the process. From the dynamic server the service is provided to the service requestor

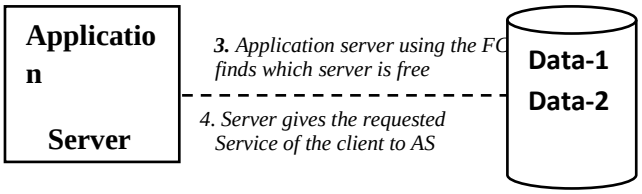


Fig.5

V.CONCLUSION

We present a dynamic approach for decentralized service discovery. In this project we are using dynamic server and application server whereas dynamic server plays a major role. Dynamic server is the permanent connection. To know which server is free from request we are using Euclid algorithm to generate free count. We create an interactive, intelligent service that is dynamically guided to locate the target service components step by step. This leads to better service discovery by comparing the service request of the servers to find the appropriate service description. This concept has been
brought into our project effectively merging it with the framework of dynamic selection of services.

ACKNOWLEDGMENT

THIS WORK IS PARTLY SUPPORTED BY IEEE TRANSACTIONS ON SERVICES COMPUTING, VOL. 6, NO. 1, JANUARY-MARCH 2013

REFERENCES

[1] H. Zhang, "Scalable Web Service Discovery on P2P Overlay Network," Proc IEEE Int'l Conf. Services Computing (SCC '07), pp. 122-129, 2007.36
[2] E. Al-Masri and Q.H. Mahmoud, "Crawling Multiple UDDI Business Registries," Proc. 16th Int'l Conf. World Wide Web
[3] T.H.-T. Hu and A. Seneviratne, "Autonomic Peer-to-Peer Service Directory," IEICE Trans. Information System, vol. E88-D, no. 12,pp. 2630-2639, 2005.
[4] D. Novak and P. Zezula, "M-Chord: A Scalable Distributed Similarity Search Structure," Proc. First Int'l Conf. Scalable Information Systems..
[5] E. Al-Masri and Q.H. Mahmoud, "Crawling Multiple UDDIBusiness Registries," Proc. 16th Int'l Conf. World Wide Web(WWW '07), pp. 1255-1256, 2007.
[6] S. Narayanan and S.A. McIlraith, "Simulation, Verification andAutomated Composition of Web Services," Proc. 11th Int'l Conf.World Wide Web (WWW '02), pp. 77-88, 2002.

BIOGRAPHIES

Mr. N. Aravindhu was born on August 18, 1979, in Chidambram, India. He received the Master of Science in Software Engineering from Annamalai University and Master of Technology in Computer Science from Pondicherry University. Now he is working as Assistant Professor in Christ College of Engineering and Technology, Pondicherry, India.

Ms. C. Shalini was born on 27thSeptember 1992, in Panruti, India. She is currently pursuing the final year B.Tech degree in Computer Science from The Christ College of Engineering and Technology, Pondicherry, India.

Ms. R. Jayalakshmi was born on April 24, 1993, in Pondicherry, India. She is currently doing her final year B.Tech degree in the stream of Computer Science from Christ College of Engineering and Technology, Pondicherry, India.

Ms. S. Priyavadhani was born on June 18, 1992 in Pondicherry, India. She presently relishes towards B.Tech final year of the department, Computer Science at Christ College Of Engineering And Technolgy ,Pondicherry, India.