iJRASET
International Journal For Research in Applied Science and Engineering Technology



INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 3 **Issue:** X **Month of publication:** October 2015

DOI:

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

www.ijraset.com Volume 3 Issue X, October 2015
IC Value: 13.98 ISSN: 2321-9653
International Journal for Research in Applied Science & Engineering Technology (IJRASET)

A Comparative Study of Different Client Server Applications in Java with Their Implementation

Monika Kamra
Department of Computer Science
Bhagwan Mahavir Institute Of Technology (BMIET)
Deenbandhu Chhotu Ram University of Science & Technology (DCRUST), Sonepat

Abstract— The client/server system consists of two logical components - a server, which offers a service and a client, which requests a service from the server. Both components, therefore, form architecture with distributed responsibilities, similar to a consumer/producer relationship. This relationship is characterized by a customer who asks for goods and/or services. The aim of the producer is to satisfy the consumer's demands accordingly. The client–server model is a standard model for network applications. A server is a process that is waiting to be contacted by a client process so that the server can do something for it. A client is a process that sends a request to the server. In this thesis, attempt has been made to study the basic issues and requirements for design and development of web oriented client-server applications and to examine the Java language in the context of these requirements. This thesis especially concentrates on the inter-process communication and the user interface.
Keywords— JDBC, Applet, Frame, Socket Programming, Client-server programming.

I. INTRODUCTION

In computer networking a large number of separate but interconnected computers work together. An application that requires two or more computers on the network is called a network application. The client–server model is a standard model for network applications. A server is a process that is waiting to be contacted by a client process so that the server can do something for it. A client is a process that sends a request to the server. The idea behind a network application is to offload some utilities like the Graphical User Interface (GUI) and in some cases, some processing, from the server to the client. As a result the GUI can be elaborate, thus making the application more users friendly. The other issues involved in client-servers are inter-process communication, security, maintenance and requirement for special software to run the client.

In a 'conventional' client–server model, a copy of the client process lies on every client machine and the client sends a request to the server through a socket to the port that the server is listening on. This model is popular as it offers improved graphical user interface. However, it has the following disadvantages: a) a copy of the client process has to be installed on every user's machine and b) any modification or update to the client process needs to be distributed and installed on every client machine.

In a new, modified view, there is no preinstalled client on the client machine. The client process is loaded onto the client machine each and every time the user initiates a request through his/her web browser using the server's URL/IP address. This 'web oriented view' does not require the user to have a special software to run the application. Moreover maintenance is much easier. The server's administrator can ensure that the users always download the latest version of the client.

The thesis starts with an overview of concepts commonly used in networking. It goes on to discuss communications using sockets in detail and three alternative Java based examples of client-server implementation. The first example is a conventional client-server in which the clients as well as the server are applications. The second example is where the client is a GUI application providing frame window to request any information and the server is an application connected with a data file, receives the requested information and send back the result. The third example is web based, uses a Java applet along with the HTML.

II. LITERATURE REVIEW

Client/server is an overloaded word and can be interpreted a number of ways. Some uses of the term represent physical partitioning of applications across computers. For example the client computer gets services from a separate file server/computer. According to Orfali: "The client/server model, entails two autonomous processes working together usually over a network; client processes request specific services which server processes respond to and process [10]".

In this paper we define client/server as two separate logical entities that work together over a network to accomplish a task. The

©IJRASET 2015: All Rights are Reserved

server process can reside on the same machine as the client, or be located on a physically different machine. A client/server implementation is different than a standalone application. A stand alone application does not require the services of another process, it is self sufficient. An example of a standalone process would be a basic text editor. As network bandwidth continues to increase, the ability to invoke a method on a remote server is becoming transparent to the end user, so client/server architectures are becoming more attractive.

In the client/server architecture it is common for servers to become the clients of other server entities. Client/server systems seek to optimally distribute processing activities via many-to-many relationships over different computer platforms. Each server process provides a unique set of functions; a client process can interact with many server processes. Each server process must be capable of simultaneously handling multiple client requests for service. The anticipated benefit of client/server is the ability to abstract hardware and software concerns and focus on developing and building user friendly, cost effective systems [5].

One of the key requirements for a client/server model is the ability for the client process and the server process to be able to communicate. Since it is possible for client and server processes to be running on separate machines a common network and communication protocol must be employed. A client process must be capable of finding a server process, via some type of addressing protocol. For example, if TCP/IP is being used as the network protocol, the server process may have a thread that is constantly listening to a pre-specified port number, say 5000 for client requests.

All client/server systems have the following distinguishing characteristics [10]:

A. Service

Client/server is primarily a relationship between processes running on separate or possibly the same machine. The server process is a provider of services. The client is a consumer of services. In essence, client/server provides a clean separation of function based on the idea of service.

B. Shared Resources

A server can service many clients at the same time and regulate their access to shared resources.

C. Asymmetrical Protocols

Clients always initiate the dialog by requesting a service. Servers are passively awaiting requests from the clients.

D. Transparency Of Location

The server is a process that can reside on the same machine as the client or on a different machine across a network. As pointed out earlier, a program can be a client, a server, or both.

E. Message-Based Exchanges

Clients and servers are loosely coupled systems that interact through a message-passing mechanism. The message is the delivery mechanism for the service requests and replies.

F. Encapsulation Of Services

The server is a "specialist". A message tells a server what service is requested; it is then up to the server to determine how to get the job done. Servers can be upgraded without affecting the clients as long as the published message interface is not changed.

G. Scalability

Client/server systems can be scaled horizontally by adding new clients, and vertically by migrating to a larger, faster server machines or multi servers.

H. Integrity

The server code and server data is centrally maintained, which results in cheaper maintenance and the guarding of shared data integrity. At the same time, the clients remain personal and independent.

III. CLIENT-SERVER APPLICATION1

In this method, both the server and the client is a Java application. The client application has to be loaded onto the client's machine.

©IJRASET 2015: All Rights are Reserved

Volume 3 Issue X, October 2015
ISSN: 2321-9653
International Journal for Research in Applied Science & Engineering Technology (IJRASET)

The server can be concurrent i.e. serve more than one client at a time. The server and database lie on the server machine. The major disadvantage of this method, as already stated earlier, is that the software for the client has to be physically present on the client machine and the client machine has to have the Java interpreter to run it. If any changes are made to the client application, it has to be redistributed to the clients.

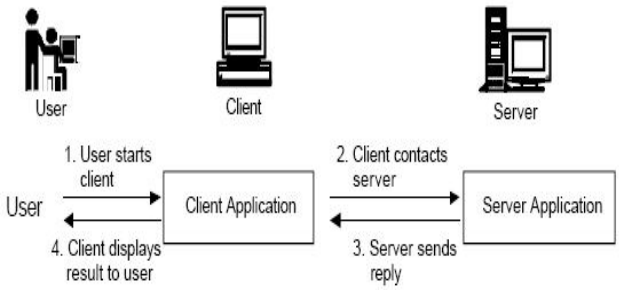


Figure 1: Server application – Client application (Conventional method)

Figure 2 & 3 shows the client server communication in conventional method.



Figure 2: Output from the Client

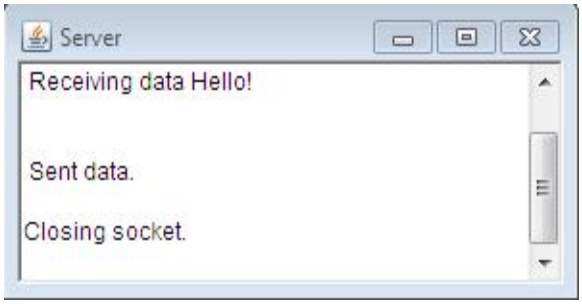


Figure 3: Output from the Server

IV. CLIENT-SERVER APPLICATION2

In this method, the server is a Java application and the client is also an application. The user enters the URL/IP address of the server and is connected to server. After connection the client can send different requests. This file provides part of the GUI and connects to the concurrent Java server using Java's multithreading. The server receives the client request, open the file stream object, process the request and send back the response.

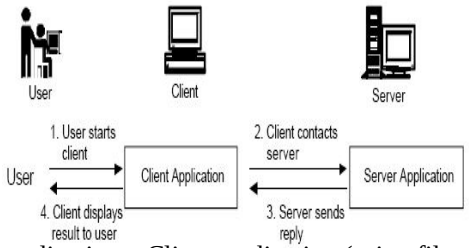


Figure 4: Server application – Client application (using file stream objects)

©IJRASET 2015: All Rights are Reserved

Volume 3 Issue X, October 2015
ISSN: 2321-9653
International Journal for Research in Applied Science & Engineering Technology (IJRASET)

Figure 5 & 6 shows the client server communication using file stream object method.

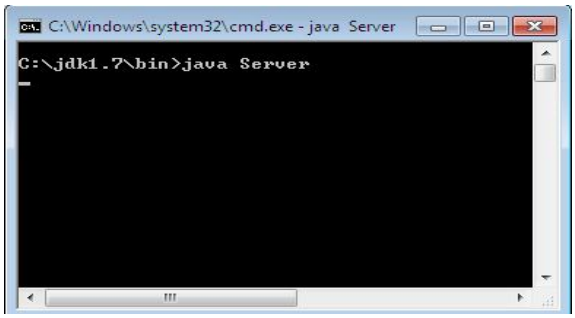


Figure 5: Output from the Server

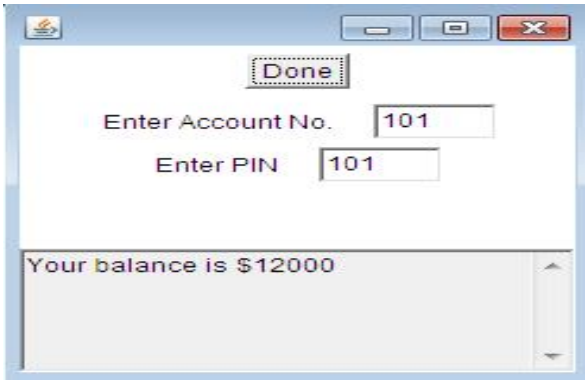


Figure 6: Output from the Client

V. CLIENT-SERVER APPLICATION3

In this method the server is a Java application and the client is a Web browser like Netscape Navigator. The user enters the URL/IP address of the concurrent server and the HTTPD web browser loads the HTML file that has the Java applet embedded in it. The HTML file here is just a vehicle to transport the applet. The applet provides the complete GUI and opens a TCP/IP socket connection to the server on a designated port.

The server may also be connected to a database (use of JDBC/ODBC bridge was made when implementing this method, this technique can also be used in the other methods described earlier)

Also JDBC allows use of any ODBC database by the server i.e. anyone providing the server service can use their existing database if it is ODBC complaint. The advantage here is that the GUI is more users friendly. The applet allows restricted and relevant use only, an important security issue.

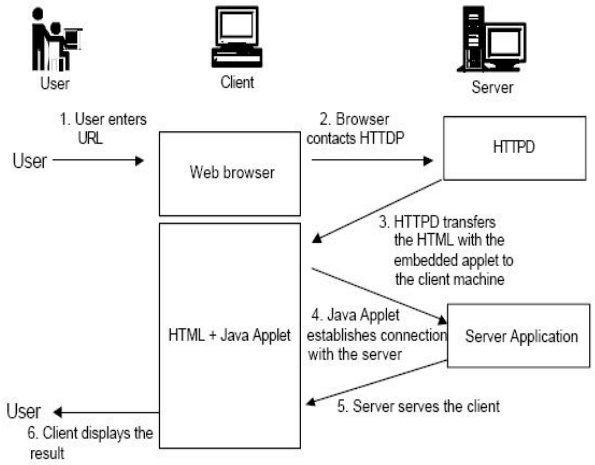


Figure 7: Server application – Client web browser (using HTML & Java Applet)

©IJRASET 2015: All Rights are Reserved

Volume 3 Issue X, October 2015
ISSN: 2321-9653
International Journal for Research in Applied Science & Engineering Technology (IJRASET)

Figure 8 & 9 shows the client server communication using file stream object method.

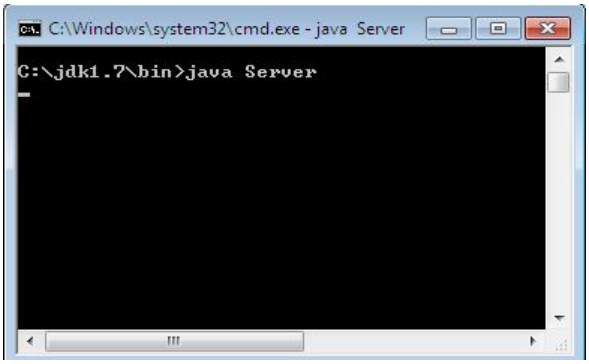


Figure 8: Output from the Server

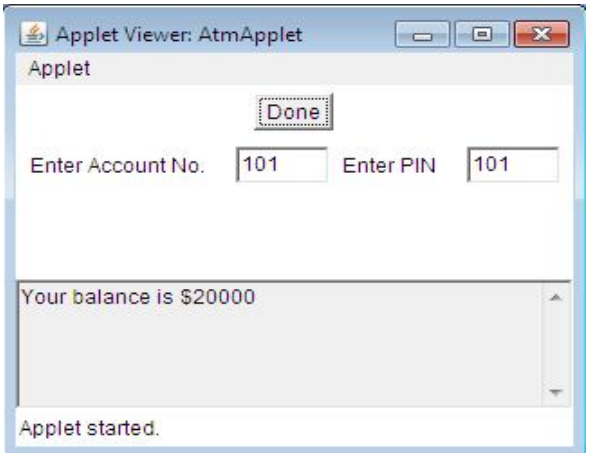


Figure 9: Displaying Balance details for the Customer

V. CONCLUSION

During the first two decades of their existence, computer systems were highly centralized. A computer was usually placed within a large room and the information to be processed had to be taken to it. This had two major flaws, a) the concept of a single large computer doing all the work and b) the idea of users bringing work to the computer instead of bringing the computer to the user. This was followed by 'stand alone PCs' where the complete application had to be loaded on to a single machine. Each user had his/her own copy of the software. The major problems were a) sharing information and b) redundancy. These two concepts are now being balanced by a new concept called computer networks. In computer networking a large number of separate but interconnected computers work together. An application that requires two or more computers on the network is called a network application. The client–server model is a standard model for network applications. A server is a process that is waiting to be contacted by a client process so that the server can do something for it. A client is a process that sends a request to the server.

The idea behind a network application is to offload some utilities like the Graphical User Interface (GUI) and in some cases, some processing, from the server to the client. As a result the GUI can be elaborate, thus making the application more users friendly. The other issues involved in client-servers are inter-process communication, security, maintenance and requirement for special software to run the client.

REFERENCES

[1] Anderson, D., Cobb, J., Korpela, E., Lebofsky, M., Werthimer, D. 2002.SETI@home: An Experiment in Public Resource Computing. Communications of the ACM. Vol. 45 Issue 11, (2002) 56 – 61.

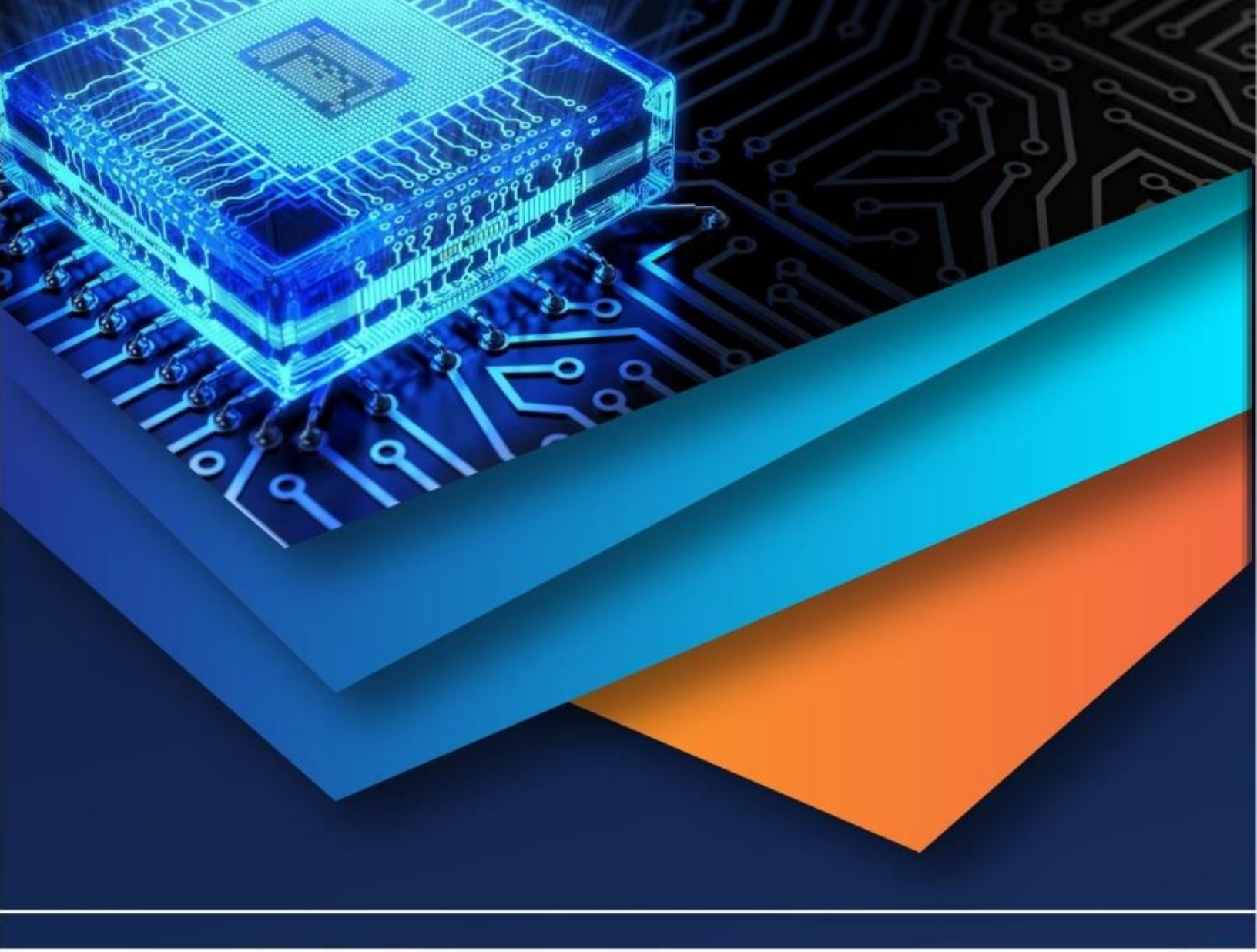
[2] Anderson, D. 2004. BOINC: A system for public-resource computing and storage. In 5th IEEE/ACM International Workshop on Grid Computing. Apple Inc., 2007, Xgrid Technical Brief, <http://images.apple.com/server /macosx/docs/L355779B_Xgrid_tb.pdf>. Last Accessed 4 April 2009.

[3] Buyya, R. 2002. Economic-based Distributed Resource Management and Scheduling for Grid Computing, Ph.D. Thesis, Monash University, Melbourne

©IJRASET 2015: All Rights are Reserved

Australia, April 12, 2002.
[4] Gary Cornell, Cay S. Horstmann Core Java, Second Edition, Prentice Hall, 1997
[5] Buyya, R., Yeo, C., Venugopa, S. 2008. 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-08, IEEE CS Press, Los Alamitos,CA, USA).
[6] Chang H., Li, K., Lin, Y. Yang, C., Wang, H., Lee L,. 2005. Performance issues of grid computing based on different architecture cluster computing platforms, 19th International Conference on Advanced Information Networking and Applications, 2005. AINA 2005. Volume 2, pp 321 – 324.
[7] Costa, F., Silva, L., Fedak, G., Kelley, I., 2008. Optimizing the data distribution layer of BOINC with BitTorrent. IPDPS 2008 Parallel and Distributed Processing, 2008.
[8] Goldsmith, B., 2007, Enabling Grassroots Distributed Computing with CompTorrent, Proceedings of the Sixth International Workshop on Agents and Peerto-Peer Computing (AP2PC 2007), Honolulu, Hawaii, USA
[9] Thain, D., Tannenbaum, T., Livny, M., 2005. Distributed Computing in Practice: the Condor Experience. Concurrency and Computation: Practice and Experience, Volume 17. pp. 323–356.
[10] A monitoring platform for Distributed java applications Włodzimierz funika1, marian bubak1,2, Marcin smetek1 and roland wism¨ uller3 Task quarterly 8 no 4, 525–536 6 August 2004
[11] Research on a Distributed Architecture of Mobile Gis Based on WAP Wang Fangxiong, Jiang Zhiyong, Research Centre of Spatial Information and Digital Engineering, Wuhan University, 129 Luoyu Road, Wuhan 430079, China. wfxwhu@163.com, State Key Laboratory of Information Engineering in Surveying, Mapping and Remote Sensing, Wuhan University, 129 Luoyu Road, Wuhan 430079, China. houyingzi@sohu.com
[12] SANS Institute InfoSec Reading Room This paper is from the SANS Institute Reading Room site. Distributed Systems Security: Java, CORBA, and COM+ Last Updated: April 3rd, 2015
[13] Java in the High Performance Computing Arena: Research, Practice and Experience Guillermo L. Taboada, Sabela Ramos, Roberto R. Exp´osito, Juan Touri˜no, Ram´on Doallo Computer Architecture Group University of A Coru˜na, A Coru˜na (Spain) Preprint submitted to Science of Computer Programming May 14, 2011
[14] KnowledgeTree: A Distributed Architecture for Adaptive E-Learning Peter Brusilovsky School of Information Sciences, University of Pittsburgh Pittsburgh PA 15260 Copyright is held by the author/owner(s). WWW 2004, May 17–22, 2004, New York, New York, USA. ACM 1-58113-912-8/04/0005.
[15] Control Systems application in Java based Enterprise and Cloud Environments – A Survey Ravi Kumar Gullapalli Hewlett-Packard Bangalore, India Dr. Chelliah Muthusamy Yahoo Bangalore, India Dr.A.Vinaya Babu JNTUH Hyderabad, India (IJACSA) International Journal of Advanced Computer Science and Applications, Vol. 2, No. 8, 2011

©IJRASET 2015: All Rights are Reserved



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Call : 08813907089 (24*7 Support on Whatsapp)