IJIRSET 2319 – 8753

International Journal of Innovative Research in Science, Engineering and Technology
Vol. 1, Issue 1, November 2012

Performance Evaluation in Accessing the Digital Library Using the Multithreading

M.Surendra Naidu[1], Dr.N.Geethanjali[2]

Test Lead, SHARP Software, Bangalore & Research Scholar, Dept of CS&T, SK University, Anantapur, India[1].

Associate Professor, Dept of CS&T, SK University, Anantapur, India[2].

Abstract: In this paper it has proposed a new automated process system that will asynchronously look for proper reviewer based on the participation and generate logs for the administrator to view and study. It has developed a new system that is parallel to the existing comfortable assessment method. The improvement in the response time may be certified to the fact that the client based systems are naturally *multithreaded* while the existing comfortable assessment method is a serial application. The results show that the proposed software client based system gives better response time.

Key words: EBSCO, Multithreads, Server, Client, Reliability

I. INTRODUCTION

Software has been integrated with software clients and DLNET (The Digital Library of Virginia Tech). First select the characteristics and features we should concentrate on for our experiments for the content review process of DLNET. Then we study client architectures such as JADE and Cougaar. We also look at the client architectures of current digital libraries using client based services and features such as the UMDL (University of Michigan digital Library), ZuNO Digital Library.

We study all these available architectures and compare them with respect to our requirements to finally decide on using the client infrastructure of JADE as the chosen middleware for our study.

1.1 Desired features of the selected architecture

We start with listing the features we want the new architecture to have

1. Interoperability – Ease of connection with other open archive digital libraries.
2. Extensibility – Ability for step by step extension to the system.
3. Ease of deployment.
4. Available documentation/support on understanding and setup of the architecture.
5. Relevance to our problem domain.
6. Easy Maintenance

We study the existing application and how content review is handled, following it up with discussion with DLNET team members to arrive at important features we want to work on.

1.2 Client Model

Software client architecture is based on peer-to-peer computing as opposed to the popular client server paradigm. The advantage of using peer-to-peer communication is that all the modes of the application can communicate with each other. The facility of looking for an appropriate peer to communicate with is separately managed. In client server model, the client has to know the exact details of the server it wants to communicate with and the communication details have to be finalized in design time itself. Software client model is flexible in the sense that once a client specifies a requirement, any other client fulfilling the requirement can respond and the two clients can communicate. These service provider clients can even be created at run time. There are applications where client server models will not suffice and so client based solutions are more appropriate. These reasons and benefits of peer to peer technology based software clients made us select our architecture. We selected JADE as the client architecture to build the relevant clients for the content review process of DLNET because,

- it had all the client features that we needed (and more)
- communication between student "clients" running on various workstations on the network was trivial to do
- it was efficient and robust enough to tolerate some common programming errors
- it followed FIPA standards
- the user group is very active and implementers typically respond to problems within 24 hours

Following table summarizes the comparison of various architectures

Table 1: Summary of the comparison of various architectures we considered

Feature	UMDL	Zunodl	Next generation digital library	Cougar	Jade
Interoperability	Good	Good	Good	Good	Good
Extensibility	Good	Good	Good	Good	Good
Ease of deployment	Not easy	Not easy	Not easy	Good	Good
Available documentation	Fair	Fair	Fair	Not good	good
Relevance to our problem domain	Not fair	Not fair	Not fair	Good	Good
Easy Maintenance	Not applicable	Not applicable	Not applicable	fair	Good

In selecting the JADE client architecture for our application, we compare it to Cougaar, UMDL (University of Michigan Digital Library) architecture, Zuno digital library and Next Generation digital library architectures. One of the important characteristics we look into the architecture is easy to understand and deploy. JADE and Cougaar are equally easy to deploy but to understand JADE and get started with it is much faster than Cougaar.

UMDL and ZunoDL were quite easily expandable but the initial deployment was relatively heavy and needed a lot more resources than JADE. UMDL and ZunoDL are basically developed to be used by multiple computers across geographical locations. The storage of resources also happens at multiple locations, whereas our system is already in place and is housed at a single place. The conceptual model is different as we are catering to users spread out in geographical location but the storage and the application code is hosted at one place. There is, therefore a different requirement for DLNET and the architecture of these two digital libraries is inappropriate for our cause.

In the model being used by digital library, the library is to be linked with other digital libraries under NSDL. To enable this was an important requirement for our project. The architectures of UMDL and ZunoDL are self contained and although they are expandable, they can't be linked easily with other digital libraries as they follow a different conceptual model. So we look for other alternatives. JADE as a software client platform is very popular and is used in many research studies in many well known European universities. The users of JADE maintain a very lively mailing list and respond to queries very fast. So this architecture is very easy to understand and we find it to be very maintainable. In addition to providing a runtime environment and a library of classes, JADE also provides graphical tools for monitoring the activity of all the registered clients.

II. MODIFYING THE EXISTING CONTENT REVIEW PROCESS

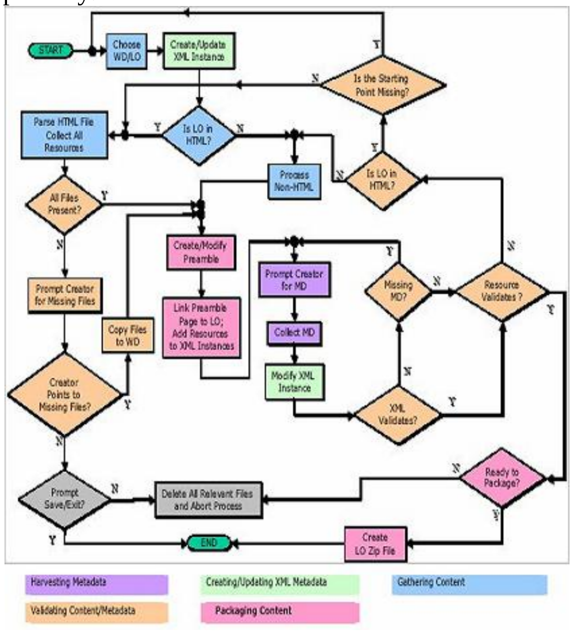
2.1 Introduction

The existing digital library process is a stable application and is being used by users worldwide. The application is a component based J2EE application. In this research paper, we investigate ways to improve the content review portion by suggesting and implementing an alternative architecture that employs software clients.

2.2 Existing Content Review Process

It has three different types of users:-
1. Contributors – Users who submit Learning Resources
2. Reviewers – Users who review the submitted resources before the resource gets added to the repository.
3. Users – Users who use the library for browsing or searching.

The content review process starts with a contributor submitting a learning resource. A learning resource could be a document, image, presentation etc. The contributor needs to add some information about the resource like the category and the type of resource. This metadata for the resource is clubbed with the resource by a utility. A check is performed to make sure that all the necessary information is given by the user at the time of submission. Also the integrity of the package is checked to make sure all the referenced objects are included in the package. Once all the checks are done, the resource and the metadata (in an XML file) are stored in a temporary repository.



Learning Object Tool Flowchart

The existing Content Review Process is a java application which uses various classes of the DLNET infrastructure. Currently what happens at DLNET is the following - the resources get submitted, the administrator keeps a check and when the temporary repository has some resources that need to be peer reviewed, the administrator starts the utility that finds the suitable reviewers for the content submitted. This is done based on the Meta data submitted along with the Learning Resource.

2.3 Proposed Content Review Process

The proposed Content Review System is a JADE application that is always running that is, the clients are always in an active state. One of the clients is responsible for regularly checking if any new content is submitted to the Digital library repository. When some content is found, the content review process starts by sending of messages between clients. When a suitable reviewer is found, logs are generated and the reviewers can be informed accordingly. The details of the system are given in chapter 5.

2.4 Expected Gains/ Performance enhancements

A more extensible and maintainable application is expected as a result of these quasi experiments.
We have added the following features to the existing application:

- The content review process is now fully automated.
- The logs are prepared about daily results. It will help in finding out the areas where digital library does not have enough reviewers and in future, the application can be extended to find out more reviewers in those areas.
- The application architecture is very easy to understand, it is easy to add new functionality to the system.
- An algorithm for a part of the application is easy to replace with a better alternative.

Digital Libraries rely on content submitted by the users registered as contributors. With the popularity of digital libraries, the content submitted will keep on increasing and so will the frequency of submission. We therefore need to make our system scalable and extensible to handle that kind of load. As the number of resource submissions increase with the popularity of digital libraries, a scalable, extensible system is desirable.

III. THE PROPOSED ARCHITECTURE

3.1 Introduction

High module cohesion and low module coupling is the rationale for most structured design methods. Good internal structure leads to good external quality. This is the basis of the design of the proposed system for Digital library content review process using the JADE architecture and software clients. The JADE architecture provides basic services for running the software client platform. It is a java application and we can easily integrate it with our existing application. Each running instance of JADE runtime environment is called a Container as it can contain several clients i.e. provide support for many clients. The first container to start must be the main container. If the application has any more containers, the additional containers have to register them with the main container. The main container holds two special clients that get started with the main container. They provide essential services for the operation of the application. These built in clients that provide Directory services and Client Management services are:-

AMS – Client Management System – This client provides the naming service by ensuring that each client has a unique name.
DF – Directory Facilitator – This is to locate a client and the clients are identified by the services they provide. In other words if a client wants to use the services of a client of type doctor, the DF will help locate a doctor type of client. After studying the current system as it works, we design the application by modularizing the components of the system and design the following clients. These clients run on the JADE platform and use the library of classes that JADE provides to together offer the complete application.
Client Starter – This is the client that starts the content review process. This client starts the execution every day. The frequency of the computing can be set by a parameter in the Client code. This client sends a message to the Client Content to start looking for a content to be reviewed.

We add the configuration parameters as constants in this client code so that these parameters can be changed as and when required.
Client Content – This client after getting a message form Client Starter accesses the database and looks for appropriate status of content that needs to be peer reviewed. If the client finds any such content, it sends an appropriate message to Client Reviewer. This client sends as many messages to Client Reviewer as the number of resources found.
Client Reviewer – This client gets the message and starts to find the reviewers appropriate for the content based on the rules defined similar to the rules in the existing application. On finding the reviewers, this client sends message to Client Email and updates the database accordingly.
Client User – If Client Reviewer cannot find any reviewers in the normal database of reviewers, this client gets a message and looks for inactive users in the reserve database of reviewers and gets the appropriate users.
Client Monitor – We have this client to log the response time of the process. This client gets messages with the start and end of the process and logs them in a text file. The difference of those times gives the response time for that run of the application.
Client Email – This client is responsible for sending appropriate messages to the selected reviewers.
Client Error Log – This client gets messages whenever there is some error in the processing and logs the errors in a text file.
The code implementing the Client Starter is given in Appendix D.

IV. PERFORMANCE EVALUATION (IMPLEMENTATION)

For the current and the following chapters, we have used the following terms
Application suggests the complete digital library application. A part of this J2EE based application; the Content Review Process is under study in this work.

It is the application currently being used by digital library Client Based Application The application being proposed in this work. It uses software clients for digital library Content Review and is based on JADE software client infrastructure. This work focuses on experimental design including the choice of experimental variables, experimental models and the instruments used to measure the dependent variables.

4.1 Experimental Design and Analysis (Methodology)

Experiments are designed to test or prove if a causal relationship exists between the chosen independent and dependent variables. In this work, we are trying to prove that the software framework used for content review in DLNET affects the performance of the system. This is an experiment to prove the effectiveness of a new software design method. With the background of our readings of [7], [8], [13] and [31], we have based our methodology mainly on the reading [24] for design. These quasi-

experiments are designed to test our hypothesis that the software client based Content Review Process has better response time, is more maintainable and reusable and so has better performance than the J2EE based existing system.

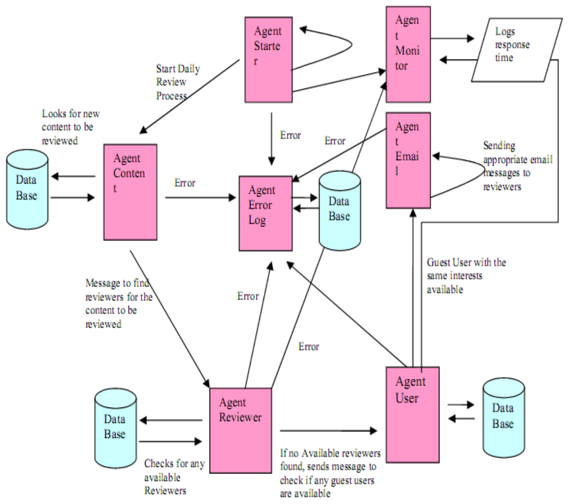

Agent Starte r
Agen t Moni tor
Logs response time
Start Daily Review Process
Looks for new content to be reviewed
Error
Error
Agen t Emai l
Sending appropriate email messages to reviewers
Data Base
Agent Conten t
Error
Agent Error Log
Data Base
Message to find reviewers for the content to be reviewed
Error
Guest User with the same interests available
Error
Agent Reviewer
Agent User
Data Base
Data Base
Checks for any available Reviewers
If no Available reviewers found, sends message to check if any guest users are available

Figure 1: Process Flow of DLNET Content Review in the Client Based Application

4.2 Experimental Variables

Independent Variable for these quasi-experiments is the type of system with the levels “old system” and “new system”. The “old system” refers to the existing application and the “new system” refers to the Client based application. We are trying to highlight that the type of system chosen affects the performance of the system. We have chosen to measure the performance by the dependent variables response time, maintainability, scalability, correctness, reliability and reusability. Through our quasi experiments and results thereof, we prove that the performance of content review process is better with the client based architecture. Extrapolating the hypothesis to apply to the whole application, we propose the new framework for digital library i.e. a software client based infrastructure.

4.3 Dependent Variables

The attributes or the dependent variables we have measured to compare the performance of the two systems are:-

4.4 Response Time (a measure of latency)

The time taken by the application to complete the review process is being measured as the response time. We have logged the response times from both the applications for a variety of load conditions ranging from 5 resources to 62 resources covering the average load conditions to heavy load conditions. In this work, for our experiments we have chosen a small part of the complete digital library application to prove that software clients based infrastructure provides better performance than the existing J2EE based application. Since the scope of our research paper is content review process of digital library, we have compared the response time for the two architectures for the content review process only. For content review, the response time is not a critical performance measure as it does not involve any user interaction. This means that the user is not expecting the response from the content review process of digital library instantaneously. But when we extrapolate our results to cover the whole application, the response time becomes a very important performance criterion. Digital library involves searching the database for the relevant information for the user. The operations involve a lot of user interaction and for that kind of communication, response time becomes important.

4.5 Maintainability

The average time taken to add a new feature or business logic to the content review process is taken as a measure of maintenance. We have tested the maintainability of both the applications by adding the same feature/ business rule in both the applications and recording the average time it would take for a developer to do so. We have tested it with the help of two software engineers with similar background so as to have an average of the measurements and to try eliminating any bias.

V. ALGORITHMS

In the current application, we have a few clients and suppose we want to add another client to the system. Say we want to add another client that optimizes the current selection of reviewers. With reference to chapter 5 describing the current application, the Reviewer and the User clients would like to communicate with this new client. Developing the new client, Client Optimizer, we want to first register the client so that other clients that want to communicate with it can find this new client. This is done as follows:

```
Protected void setup ()
   {
ServiceDescriptionsd= new ServiceDescription();
       sd.setType( "optimizer" );
       sd.setName( getLocalName() );
       register( sd );
}
void register( ServiceDescription sd)
{
BFClientDescription dfd = new BFClientDescription();
bfd.setName(getAID());
bfd.addServices(sd);
try {
BFService.register(this, dfd );
}
catch (FIPAException fe) { fe.printStackTrace(); }
}
```

On the other side, the calling client will first look for the client of the type “optimizer”
// trying to look for an client with the optimizer type of service

Adding a behavior New requirements or enhancements can be added to the clients as additional behaviors. The code for the behavior of an client goes in the action method of the client class file. Since clients only respond to some messages they receive from other clients, we first have to receive the message and then take appropriate action based on the message we receive.

Sample Logs Created

Log created by the client based application lists the reviewers selected

```
Today is  August 29, 2012   Resources found = 16
Resource = DLNET-08-28-2002-0477 Reviewer found = kiran
Resource = DLNET-08-25-2012-0138 Reviewer found = balakrishna
Resource = DLNET-08-27-2012-0141 Reviewer found = pradeep bs
Resource = DLNET-08-27-2012-0142 Reviewer found = rajesh ks
Resource = DLNET-08-27-2012-0143 Reviewer found = kiran
Resource = DLNET-08-28-2012-0144 Reviewer found = pradeep bs
Resource = DLNET-08-28-2012-0145 Reviewer found = kiran
Resource = DLNET-08-28-2012-0226 Reviewer found = rajesh ks
Resource = DLNET-08-28-2012-0227 Reviewer found = balakrishna
Resource = DLNET-08-28-2012-0229 Reviewer found = pradeep bs
Resource = DLNET-08-28-2012-0228 Reviewer found = rajesh ks
Resource = DLNET-08-28-2012-0230 Reviewer not found
Resource = DLNET-08-28-2012-0230 Guest Reviewer found = kiran
Resource = DLNET-08-27-2012-0139 Reviewer found = balakrishna
Resource = DLNET-08-27-2012-0140 Reviewer found = kiran
Resource = DLNET-08-28-2012-0231 Reviewer not found
Resource = DLNET-08-28-2012-0231 Guest Reviewer found = kiran
Resource = DLNET-08-29-2012-0146 Reviewer found = pradeep bs
```

Logs created by the existing application to log the reviewers selected and the time taken by the application

```
Today is  August 29, 2012   Time Review Started = 1088522158343
Resource = DLNET-08-28-2002-0477
Reviewer Found = kumar chowdary
Resource = DLNET-08-25-2012-0138
Reviewer Found = balakrishna
Resource = DLNET-08-27-2012-0141
Reviewer Found = bspradeep
Resource = DLNET-08-27-2012-0142
Reviewer Found = kirnakumar
Resource = DLNET-08-27-2012-0143
Reviewer Found = rajesh
Resource = DLNET-08-28-2012-0144
Reviewer Found = brapeeb bs
Resource = DLNET-08-28-2012-0145
Reviewer Found = rama
Resource = DLNET-08-28-2012-0226
Reviewer Found = suma
Resource = DLNET-08-28-2012-0227
Reviewer Found = balakrishna
Resource = DLNET-08-28-2012-0229
Reviewer Found = Rajesh ks
Resource = DLNET-08-28-2012-0228
Time Review Process Ended = 1088522165265
```

VI. TEST RESULTS

6.1 Response Time

Number of Resources	Response time of the Existing application(m Sec)	Response time of the client based application (msec)
5	3672	1281
7	30063	12853
10	14313	2600
30	23140	7058

Response Time Results

6.2 Maintainability

To compare the maintainability of the applications, we had three different people add to the applications and then compared the average time taken by them. To have an unbiased comparison, we selected two other persons who had similar experience and understanding of java (the basis of the existing application) and JADE (the client architecture of the new application). We added a few features and made some modifications to the two applications. For these quasi-experiments, we had two other software engineers understand the two applications and make changes to them.

We then took the average time taken by them to add to/ modify the two systems. The following are the results of the maintenance quasi-experiments. For the first three activities, we have considered only two readings as the third person was already familiar with the applications when we started these experiments.

Test Condition	Time taken to Change/ review the existing application			Time taken to Change/ review the client based application		
	Kiran	Surendra	Vaishali	Kiran	Surendra	Vaishali
Understanding the working of the application		19.5 hours	20.7 hours		25 hours	26 hours
Understand how to start the application		6.3 hours	5.9 hours		4.5 hours	3.9 hours
Changing the configuration parameters		8 hours	9 hours		2.5 hours	3.1 hours
Adding the creation of log reports	11 hours	13.3 hours	12.9 hours	3.5 hours	4.3 hours	5.4 hours
Identifying the peace of code and replacing with more efficient one	10 hours	11 hours	9.6 hours	8 hours	8.7 hours	8.3 hours

Table 3: Raw Maintainability Results

The results shown below are the average of the above readings.

Test Condition	Time taken to Change/ review the existing application	Time taken to Change/ review the client based application
Understanding the working of the application	19.1 hours	23.5 hours
Understand how to start the application	6.1 hours	4.1 hours
Changing the configuration parameters	8.5 hours	2.5 hours
Adding the creation of log reports	12.5 hours	4.3 hours
Identifying the peace of code and replacing with more efficient one	10.1 hours	8.6 hours

Table 4: Summary of Maintainability Results

VII. PERFORMANCE COMPARISON RESULTS

Based on the data measured by the testing tools designed as described in the previous chapter, this chapter discusses the results and compares the performance of the two applications.

7.1 Response Time

To measure the response times of the two applications with varying load so as to compare them Is the Software client based application faster than the existing application and is this performance constant with varying loads.

We designed measuring instruments to create a log of the response times of both the applications. For the existing application, we added a code sample to create the log of the response times each time we conduct the test and for the client based application, we added another client to create similar logs. To eliminate the possibility of bias, we first considered having one code module for both the applications. The two applications have very different architectures and connecting them both to the same module was not possible, so we designed two different modules. We designed them to make sure that we record the time in similar fashion in the two applications. We have designed the instruments to record the response time for both the Applications in a similar method. We record the system time before the first line of code in the review process and record the time after the last line of the process. The difference in the two times gives the response time of the applications.

The results show considerable improvements in the response time for the process with the proposed application using Jade infrastructure and software clients as compared to the existing application. The huge improvement in response time can be partly explained by the fact the client based systems are inherently multithreaded while the existing content review process is a single threaded serial application. We conducted tests for varying load conditions and the results were found to be in the favor of the client based application. Following is the plot of the results showing a comparison.

7.2 Maintainability

To design instruments to measure the maintainability of the two applications. Is the Software client based application easier to add to / modify than the existing application.

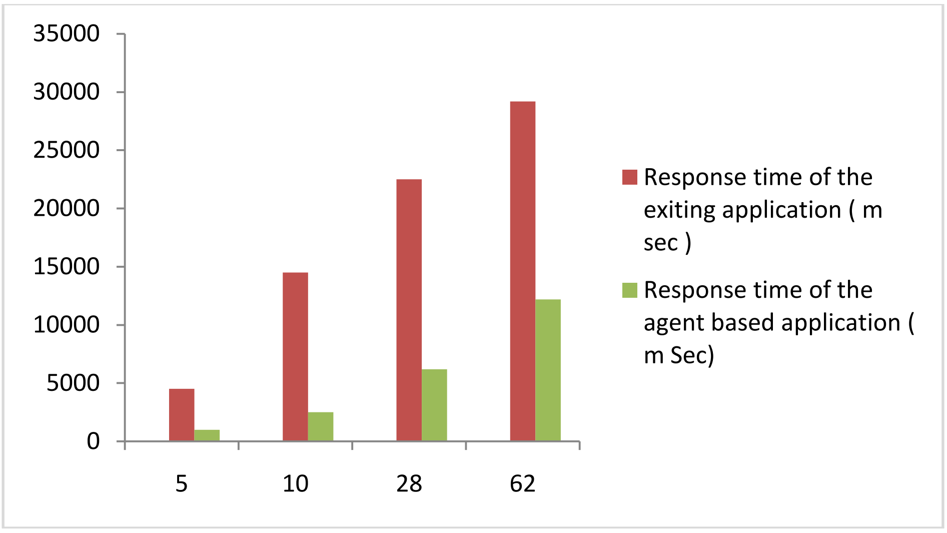


Figure 2: Bar Graph showing a comparison of response times

To measure the maintainability of the applications, we identified five areas that were put to test. We had two engineers with similar backgrounds who helped us in these experiments. We started with observing the time it takes someone having prior knowledge of java to understand the working of the two applications. Then we noted the time it takes to start the application, then the time it takes to manage the applications like changing certain administrative parameters. We then measured the time it took the same engineers to add a functionality to the two applications. The fifth measure of maintainability was the time it took to optimize a part of the code by changing it and testing the change. To attempt to eliminate the possibility of bias in these instruments, we carefully selected the people to help us out in this part of the work having similar backgrounds. They were both equally conversant with java but client based technology was new to both of them

The client architecture provides a more maintainable architecture as we can see from the results of our tests. The client based application has modular architecture with clients encapsulating an abstraction. This makes this architecture easy to understand and maintain. As was found out in our measurements, it is much easier for a developer to change the functionality in the client based application as compared to the existing setup. The Client based architecture provides an in built mechanism to graphically view the interactions of various clients in real time.

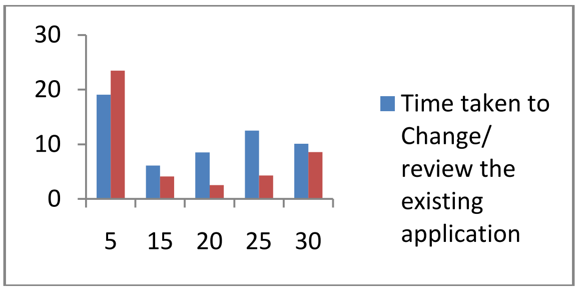


Figure 3: Bar Graph showing a comparison of maintainability

VIII. CONCLUSIONS

This paper gave examining data that the concept that software client based architecture will make the current content review portion of the digital library application more efficient maintainable and response time. There are various variables which were measured to compare the two applications' performance, and this comparison is the main contribution of the research. We measured the response time, maintainability and response time, the two applications to quantify the performance of the two technologies, namely J2EE versus Client Technology. While response times indicate the effectiveness of the applications at this time, maintainability on the other hand, reflects the ease of further updates, enhancements and modifications. In this work we can conclude that client architectures can be used in digital library applications such as digital library where a great need for interoperability, rapid application development and automation exists.

REFERENCES

[1]. Information Clients: A new Challenge for AI Daphne Koller and Yoav Shoham, Stanford University.
[2]. Web Economics: A case for Client Based Digital Libraries Innes A. Ferguson, Jorg Muller, Markus Pischel and Michael Woldridge.
[3]. Client-Based Digital Libraries: Driving the Information Economy. David Derbyshire, Innes A. Ferguson, Jorg P. Muller, Markus Pischel and Michael Wooldridge
[4]. The university of Michigan Digital Library Service Market Society Jose M.Vidal, Tracy Mullen, Peter Weinstein, Edmund H.Durfee, University of Michigan.
[5]. Paying their way: Commercial Digital Libraries for 21st century – Innes A. Ferguson and Michael J.Wooldridge, ZunoLtd.Alsoat
[6]. Comparison of Two Component Frameworks: The FIPA-Compliant Multi-Client System and The Web-Centric J2EE Platform – Michelle Casagni, Margaret Lyell
[7]. Performance Testing of Software Systems – Filippos I.Vokolos, Elaine J. Weyuker
[8]. Technical aspect of Next Generation Digital Library Project: Hiroshi Mukaiyama The architecture of the Next Generation Digital Library and reasons of adoption of the architecture are described here.
[9]. A Java-based Smart Object Model for use in Digital Learning Environments – Vara Prashanth Pushpagiri A report on the Learning Object Model of DLNET.
[10]. Next Generation Digital Library – Architecture and Implementation – Hiroshi Mukaiyama Another paper on the client based digital library highlighting the benefits of using clients.
[11]. Early Performance Testing of Distributed Software Applications –Giovanni Denaro, Andrea Polini, Wolfgang Emmerich - Suggests ways to test the performance of a distributed application from the design phase itself.
[12]. Jade A White Paper – F.Bellifemine, G.Caire, A.Poggi, G.Rimassa An introductory paper on Jade client architecture.
[13]. Developing a Digital Library Of Computer Science Teaching Resources - Scott Grissom, Deborah Knox, Elana Copperman, Janet Hartman, Marja Kuittinen, David Mutchier, Nick Pariante
[14]. Software Metrics : A Rigorous Approach – Norman E Fenton A book which guided the experiment design and analysis of this work.
[15]. Analytical Usability Evaluation for Digital Libraries: A Case Study – Ann Blandford, Suzette Keith, Iain Connell & Helen Edwards.
[16]. Strategic Directions in Electronic Commerce and Digital Libraries: Tpwards a Digital Agora – Nabil Adam and Yelena Yesha et al. A survey paper comparing electronic commerce and the digital libraries concepts.

[17]. Data Mining Challenges for Digital Libraries – Robert L. Grossman A paper highlighting the future challenges for digital libraries.

Biography

M. Surendra Naidu, working as a Test Lead, for Sharp Software Development India Pvt. Ltd., He had completed his M.Sc.Computer Science from Sri Krishna Devaraya University. He is Research Scholar at Sri Krishna Devaraya University, Anantapur, Andhra Pradesh, India. His research interests are in the field of Software Engineering, Data Mining, Image Processing and Operating System.

Dr.N.Geethanjali Obtained her Ph.D degree from Sri Krishna Devaraya University, Aanatapur, Andhra Pradesh, India. She has been working as an Associate Professor in the Dept of Computer Science and Technology, Sri Krishna Devaraya University. She has Published More than 30 papers National and International Journals & Conferences in the area of Computer Networks, Adhoc Networks, Data Mining and Software Engineering.