Abstract

This paper reviews Grid computing technology which provides powerful computing resources and various services. In our prototype of Grid Infrastructure we implemented Virtual Reality (VR) by using it to share Virtual books. Users will be able interact with the manuscripts as if they interact with the real book.

In this work, a system of virtual book driven by Grid services is built. The visualization is developed using Java bindings for OpenGL (JOGL). Our applications are built with the advantage of statefulness of Web Service Resource Framework (WSRF) and are deployed on Grid infrastructure with Globus Toolkit 4. The result of the virtual book implementation evaluation on 5 computers using grid technology gives an average of 2.9 second of delay and 10 kB of data for communication. Test result shows that there are a number of 0.15% increments per second of memory consumption on the system.

Keywords–distributed services, Grid infrastructure, Web Service, WSRF, WS-Notification, WS-Resource.

1. INTRODUCTION

A. Background

Aging has made hundreds years old artifacts need to be preserved. Nowadays, interactions with old artifacts are limited because of their vulnerability. Almost all of them are placed in glass boxes. On the other hand, computing technology has come to such an era in which it is possible to solve those problems. 3D computing has played important roles in many aspects, from entertainment to education purposes. 3D technology is used as an alternative way to bring those artifacts back to life. The digital copy of manuscripts are produced through photography, rendered in 3D application, and placed in a program that supports user interaction.

Another point to be introduced here is Grid computing technology. Grid is a technology which is developed with objectives of distributing computing resources as well as power resources. Most powerful computing resources are not well-used or fully utilized by their users. Grid technology is developed to manage those unutilized resources for other processes in need of computing power.
B. Problem Definition

The problem defined in this work is to build a virtual book on Grid services. This virtual book is developed in such a way so that it can be accessed simultaneously by users using different computers. The system has to grant synchronization of properties of the virtual book for all users.

2. GRID INFRASTRUCTURE

A. Grid

Grid is a computing and data control infrastructure which provides supports for global society ranging from business, government, research, science, and entertainment [1, 2]. Grid integrates network, communication, computing and information resources to build a virtual platform for computation and data control, the same way Internet integrates resources to make a virtual platform for information.

Grid infrastructure can connect resources dynamically to support execution of large scale application, resource intensive and distributed. Grid acts as an interface between user and resource system.

B. Web Service Concepts

1) Open Grid Services Architecture (OGSA)

Open Grid Service Architecture (OGSA), which is developed by Global Grid Forum (GGF), defines practical standards for services on Grid application (job management services, resource management services, security services, etc.) by defining a set standard interfaces for those services.

OGSA chose some parts of distributed middleware to provide the architecture using Web service.

2) Web Service Resource Framework (WSRF)

WSRF, which is developed by OASIS, rules ways to make Web service to be stateful. WSRF is a collaboration work of Grid and Web service communities, making it suitable for all web service specifications (as in Figure 1: WSRF extends Web Services). Briefly, WSRF provides stateful services which are needed by OGSA. Or, in other words, OGSA is the architecture, while WSRF is the infrastructure on which the architecture is built.

3) Relationship with GT4

Adding GT 4 to Fig. 1 yields the one depicted in Fig. 2.

---

**Fig. 1 OGSA, WSRF, and Web Service relationship [5]**

**Fig. 2 OGSA, GT4, WSRF dan Web service relationship [5]**
Globus Toolkit is a software toolkit, developed by the Globus Alliance, to deploy grid based application. This toolkit contains some high-level services to build grid application which meet almost all of OGSA requirements.

Much of the services are implemented on WSRF. GT4 includes complete implementation for WSRF specification. Figure 3 summarized the explanations above in layer diagram.

Fig. 3 Layer Diagram of OGSA, GT4, WSRF, and Web Service [5]

4) Service

Web service is different from website. Information on a website is aimed for human, while information on a web service has to be accessed by a program.

Compared to the other distributed technologies, Web service has some advantages such as open platform using XML standard format, HTTP communication which pass almost all firewalls. And the disadvantages are the overhead, and the lack of versatility.

3. VIRTUAL REALITY

Virtual Reality is a system which provide artificial environment simulated by computer which can make users’ senses believe that it was real. The system includes all of the tools (haptics, displays, sound system) and their configuration in creating a good composition.

Development of the virtual environment in this research objects to enable user interaction with virtual artifacts as natural as possible. This research still uses standard input, keyboard and mouse.

Our work is focused on the development of the visualization of VR. This work takes advantage of 3D programming using Open Graphics Library (OpenGL). OpenGL is a standard specification for defining open platform APIs for creating computer 2D and 3D graphics.

In this work we use open platform characteristics by defining several bindings and ports for several programming languages. This work uses OpenGL binding for Java, JOGL.

This research brings up the ancient Javanese manuscripts to be displayed on the application built. These manuscripts are inventories of Centre of Javanese Studies, Faculty of Humanities, University of Indonesia.

4. DESIGN

A. Grid Infrastructure

Fig. 4 shows the configuration of the Grid infrastructure at the Mercator Multimedia Laboratory, Faculty of Engineering, University of Indonesia. This infrastructure involves 5 computers connected by 100 Mbps LAN. Globus Toolkit 4.0.1, as the middleware, is installed on each computer.

Fig. 4 Testbed configuration
Globus Toolkit, as the middleware, controls resource management of the Grid. Globus toolkit is the most popular grid middleware with some standards and security features (Grid Security Infrastructure).

Our testbed configuration is based on two distribution of Linux (i.e. Ubuntu LTS 6.06 and Fedora Core 4). This variation is aimed for comparison purposes and to prove the open platform impact.

B. Virtual Reality on Grid Infrastructure

Our work on Collaborative VR designing in this research is aimed at enabling more than one user to share their experiences with virtual manuscripts in a virtual museum.

To make it real, real-time and interactive communication between user and the system is required. After the virtual environment is defined, it needs to be accessed simultaneously by some users to serve their requests such as request to turn pages and to interact with other user using avatar.

In our work we focused on building a system to enable interaction with a virtual book and make it synchronous experience among users.

Our system was built on an open platform system and Web service was chosen to be the base system. Web service uses XML format which is platform independent in information exchanges. Web service use HTTP traffic like WWW which is trusted by many firewalls, allows it to be used in Internet.

Web Service Resource Framework (WSRF) allows resource definition in which information is stored. This resource is the one used to hold conditions of the virtual book.

5. IMPLEMENTATION: 3D PROGRAMMING USING JAVA OPENGL

The Graphical User Interface was developed using Java and Java Bindings for OpenGL (JOGL) as the 3D library.

After preparing the grid infrastructure, the work focused on developing client application as an interface for users. Figure 6 illustrate the making of the development of the virtual book. This application was built with a book opened and pages displayed with illustrations from old Javanese manuscript as the main object. In addition, some fix ornaments are placed behind and under the book, a plant and a granite table.

The connection between user and the system is built on client-server configuration, making centralized services, which is suitable for collaboration. Using
this approach, as illustrated in Figure 5, users can make requests separately from their computer using client, applications to a single server which manage some details on clients and make sure that the clients are synchronous one another.

The next problem is to decide which scenario of communication to be used. Whether server needs to send all of the virtual book visualization, or make the client applications to render their own visualization and server just need to send some properties of the virtual book. The two problems are discussed on the following paragraphs.

(1) Server does the 3D Rendering
Client provides Graphical User Interface for the users, and sends all of the interactions as requests directly to the server. Server receives them and does the computation, renders the new visualization based on the models and users’ request. Server may use parallel CPU capacity to make the computation faster. However, this configuration depends a lot on the dedicated network for transmitting large size of information – 3D object visualization. Particularly, for Virtual Reality application, in order to make good user experience needs high quality standards (i.e. frame size, frame rate, and bit depth). For example, the picture with 1024 x 768 pixel, 34 fps and 24 bit will need a number of nearly 0.5 Gbps.

\[
\frac{24 \text{ bit}}{\text{pixel}} \times (1024 \times 768) \text{pixel/frame} \times 24 \text{ frame/sec} = 452,984,832 \text{ bit/s}
\]

(1)

(2) Clients render their own visualization
In this configuration, beside the Graphical User Interface (GUI), client was also built with OpenGL, making all of rendering process to be conducted in client machines. Then, in order to make it to be collaborative; attributes of the virtual books are also copied to the server. Attributes at the clients has to be synchronized to them at the server. Since clients synchronized their condition to a single server, they will produce the same pages of the book to be displayed.

Fig. 7 describes the system activities in sequence diagram. Subscription processes (sequence number 1 and 2) are only invoked once; at the initiation stage of the clients. The rest activities can be held for as long as the client and server are alive and connected. Clients which are connected to the server but are not navigated (by users), will have almost the same sequence, except the sequence number 3. They will get notifications caused by other users’ actions, and then, renew their display.

In this work, the second configuration was chosen. It makes client-server communication just need to pass a small number of data, making it to be more efficient. But, this configuration has disadvantages because clients will have to manage their rendering process individually. Since the 3D object information is distributed on each client, it will need extra efforts when doing some upgrades in the system.

Since clients need to synchronize their condition to the server, a notification mechanism was used. It is a part of WSRF. This mechanism has advantages over polling mechanism where clients just need
to ask for new parameters when a notification is given instead of periodically ask for parameters. In notification mechanism, clients need to subscribe themselves, asking for being informed when special events happened at server. After that, clients just have to listen for notification from server.

![Attributes and operations of the two applications](image)

Fig. 8 Attributes and operations of the two applications

Figure 8 illustrates that both of client and server application have same attributes: “sudut”, “HLM” and “pesan”. However, they play different role on each application.

Client application provides 3D object visualization for user and uses those attributes to create composition of the 3D object to be displayed. “sudut” defines the angle of the page opened. “HLM” defines the image file to be loaded and attached on the virtual book. Finally, “pesan” defines the text message for the user. Besides the 3D object visualization, client also provides a GUI panel for several actions: “Next”, “Previous” and sends message to other users. These action are not processed at client, they need to invoke corresponding method on the server.

Server has also those attributes, “sudut”, “HLM”, and “pesan”, as resource to be managed. When an action is given to the client, corresponding operation on server is invoked. Server modifies the values to the three attributes, and then informs those changes to clients which subscribed.

On receiving server’s notification, client sends requests for those values by calling GetResourceProperty(). Then, using information it got, client renews its display screen. Fig. 9 shows the use case diagram describing the functionality of the system.

The following activities are conducted to design the web service. The first step is interface definition using WSDL. Interface is a declaration of services provided by the Web service. This declaration is done in WSDL, a file which will be used by GT4 to create stub classes when compiled.

![The use case diagram](image)

Fig. 9 The use case diagram

WSDL schema consists of several parts. The first part is the namespace definition, continued by interface (port type) definition, defining how input messages accepted as a request, and output messages produced as a response. The next part is types of requests and responses definitions, and the declaration of resource properties.

The next part is to implement the service using Java. Resources are managed and accessed using methods defined and ready to be invoked by the clients. The client-server communications are bridged by stub classes.

Then, the deployment parameters definition is done by defining the location where the service is deployed on the service container. This is done using WS Deployment Descriptor (WSDD), which defines the service’s URI and path to corresponding files.

The following order is to compile the programs and create GAR. Grid Archives
(GAR) contains all of the information and files needed by GT4 to deploy the service on the container and make it accessible by users. GAR is built by Apache Ant, using information defined in the last three steps. The next step to be done as the last step in this web service development is deploy the service on GT4. This is done using deployment function prepared by GT4.

Fig. 10 The client GUI

Fig.10 displays the application runs on the client machine upon execution of the system. Application development for client has two focuses. The first is user interfacing, GUI. The next is developing NotificationConsumer of the web service. As a NotificationConsumer, client needs to listen for notifications from server. Therefore, this application needs a special process to do listening process. It was done using methods facilitated by GT4.

6. SYSTEM PERFORMANCE EVALUATION

A Analysis

1) Data Traffic

Making clients render the 3D object individually, this system minimized data traffic on the network. Client-server communications only have to pass information of which page of book to be opened, and some text messages.

Only passing these information makes data traffic on the network more efficient, instead of passing big data, such as rendered images. However, the information has to be formatted in such a way that is understood by the stubs on each side, client and server. The configuration used here reduced network usage. For example, size of full screen images with high quality criteria is:

\[
\frac{3 \, \text{byte}}{\text{pixel}} \times (1024 \times 768) \frac{\text{pixel}}{\text{frame}} = 2.359.296 \, \text{byte} = 2.25 \, \text{MBpixel frame}
\]  

(2)

This value is larger than a complete communication, request and response, which just about 10 Kb.

This data traffic includes client communicating server to get information of the page of the book. Client application contacts client stub to produced three XML data with GetResourceProperty, 1.4 kB of size. This XML data is transmitted to the server and received by the server stub. After that, server stub produced XML data contains GetResourcePropertyResponse, about 1.3 KB of size.

Client application asks the client stub to create a Subscribe request. This will be XML data about 1.8 kB of size. This XML data contains on which port client will listen for notifications and also NotificationConsumerKey. Subscription Response will be XML data about 1.5 kB of size, contains subscriptionKey.

From the panel GUI provided on alient application, user may do 3 kinds of actions, “Next” (turning the page 5 degrees to the right), “Previous” (turning the page 5 degrees to the left) and send a textual message, beside close the application. The three actions have a similar characteristic.

When “Next” button is pressed, client will call a method on client stub, then producing nextRequest for server, about 1,3
In the server, this XML data is received by stub server, invoking corresponding method on server application, in this problem, the next method.

After resource on the server is modified, subscription manager sends notification to clients, telling them about the changes of the resource. This notification is packed together with NotificationConsumerKey in XML data about 2.5 kB in size.

On receiving the notification, client produces Get ResourceProperty() and passes it to the server, then using server’s response, clients renew their display.

### Table I Data sizes for client server communication

<table>
<thead>
<tr>
<th>Source</th>
<th>Dest.</th>
<th>Operation</th>
<th>Data XML size (byte)</th>
</tr>
</thead>
<tbody>
<tr>
<td>client</td>
<td>server</td>
<td>Subscribe</td>
<td>±1900</td>
</tr>
<tr>
<td>server</td>
<td>client</td>
<td>SubscribeResponse</td>
<td>±1500</td>
</tr>
<tr>
<td>client</td>
<td>server</td>
<td>GetResourceProperty</td>
<td>±1470</td>
</tr>
<tr>
<td>server</td>
<td>client</td>
<td>GetResourcePropertyResponse</td>
<td>±1380</td>
</tr>
<tr>
<td>server</td>
<td>client</td>
<td>Notify</td>
<td>±2500</td>
</tr>
<tr>
<td>client</td>
<td>server</td>
<td>NextRequest</td>
<td>±1400</td>
</tr>
<tr>
<td>server</td>
<td>client</td>
<td>NextResponse</td>
<td>±1200</td>
</tr>
<tr>
<td>client</td>
<td>server</td>
<td>PreviousRequest</td>
<td>±1400</td>
</tr>
<tr>
<td>server</td>
<td>client</td>
<td>PreviousResponse</td>
<td>±1200</td>
</tr>
<tr>
<td>client</td>
<td>server</td>
<td>ChatRequest</td>
<td>±1400</td>
</tr>
<tr>
<td>server</td>
<td>client</td>
<td>ChatResponse</td>
<td>±1200</td>
</tr>
</tbody>
</table>

Common communication traffic includes Notification, GetResourceProperty, and GetResourcePropertyResponse. By assuming user do an action every second, traffic rate can be calculated as followed:

\[
2500 + 3 \times (1470 + 1300) = 10810 \text{ B/s} \approx 10 \text{ kB/s}
\]

#### 2) Timing

Timing data were captured by calling a method, currentTimeMillis(), right from the applications. currentTimeMillis() is a method in java.lang.system class which returns current time information on the machine.

Fig. 11 shows the captured data of communication timing between a client and server, it also describe the details of delay component of the system. Those components can be categorized into 2 groups, data passing delay and data process delay. In this research, data passing delay is not a serious problem on LAN 100 Mbps, but might be serious if implemented for wider network. Data for this system are processed in some simple operations, so process delay is quite acceptable.

![Fig. 11 Client-server communication timing](image)

**Table II Statistic for timing data**

<table>
<thead>
<tr>
<th>Aspect</th>
<th>Komputer Uji Label</th>
<th>Data rate-bps</th>
<th>Rate-ratio</th>
<th>Rate-bps</th>
<th>Rate-ratio</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data transfers</td>
<td>1</td>
<td>128 254</td>
<td>1.25</td>
<td>128</td>
<td>1.25</td>
</tr>
<tr>
<td>Data pass</td>
<td>1</td>
<td>128 254</td>
<td>1.25</td>
<td>128</td>
<td>1.25</td>
</tr>
<tr>
<td>Data display</td>
<td>1</td>
<td>0.250</td>
<td>0.25</td>
<td>0.250</td>
<td>0.25</td>
</tr>
</tbody>
</table>

Analysis for table above divided into the groups, each for different number of client connected. From the table, increment of the number of client doesn’t affect timing and delay significantly.
Fig. 12 Comparison of the timing from the three clients

Fig. 12 and Table 2 describe how the testing process was held. There is one client for action number 1 to 9, and then the second client is activated, so there are two clients for action number 10 to 18. After that the third client is activated. Upon incrementing the number of clients, there were not significant changes to the timing and delay of client server communication.

B) System Limitation

1) Overhead

The listing above shows a message from the server containing a text message from a user. It is shown that short information need a long data passing over the network. It is because the communication between server and client uses SOAP standard format. This standard format requires the overhead as a way granting the open platform criteria of this system. Using this standard format, different kind of system can understand message they received. This condition can’t be achieved if direct data transfer. There has to be a trade-off between efficiency and portability.

As shown in Fig. 13, every SOAP message has 3 parts: envelope, header and body.

Fig. 13 SOAP communication data format

Header consists of identification information of the communication. They are MessageID, destination address, type of action, and source address.

Body consists of information needed by the action referred in the header. For example, the listing above is a GetResourcePropertyResponse of “pesan”, so it contains the resource name and value.

Envelope consists of scheme used in the communication. It also wraps the header and the body.

The listing above shows a message from the server containing a text message from a user. It is shown that short
sending requests in time interval shorter than the system delay. The long standard message format causes delay in message passing. Actually, these delays harmless to system, all the request are processed completely. The problem is information given to users is the last information on the server.

2) Memory Usage

The client application was built to be run in X window environment. It provides a GUI panel and virtual book visualization. This application consumes so much memory resources on client’s machine. Fig.15 shows the memory usage characteristics.

Data mining of this testing is done using free command in UNIX system. This commands returns memory and swap usage information of the computer.

Fig. 15 illustrates very high memory consumption for client application. The needs of memory resource increase linearly time by time. This increment is calculated to be 0.15% of total memory capacity every second.

![Fig. 15 Memory and swap consumption of client application](image)

This characteristic is caused by periodic visualization refreshing mechanism used by client application. In this renewing process, client load image files periodically. So, the longer application runs, the more files are loaded to memory. The main memory has to be managed by operating system in order to avoid system crash.

7. CONCLUSION

Virtual reality can be a good alternative for displaying old artifacts. Using the digital copies in virtual environment, the real artifacts still can be preserved from damage. This objective can be combined by advantages of Grid technology. Grid manages the computing resources to be distributed on the system. This research has implemented a virtual book application to display ancient Javanese manuscripts, distributed among 4 clients using Grid services. The client-server applications communicates using SOAP standard message format with 10 kB of data size for each communication. The increment on memory consumption for about 0.15 % of total memory per second still problem remained.

8. FUTURE WORKS

Problems and limitation of this system has been described. Development of this system can be started from rechecking those problems while integrating more grid technology advantages.

The system which is developed only focused on an object, the virtual book. To build a system with full integration with Virtual Reality, a wider context has to be defined, for example, a virtual museum. This way, the museum can be programmed as the virtual world where users’ visualizations, avatars, can be placed and interact each others. The described project has been done taking advantages of Virtual Machine named Croquet Project. That project has placed a museum building with
two rooms for cultural artifacts from 2 countries [10].

Another point to be concerned is copyright and security problem noting that information distributed in this system are high valuable nation’s assets.

9. ACKNOWLEDGMENTS

We would like to thank all parties for all their valuable contribution: Daniela M. Romano, Jessica Cauchard, MSC, F. Astha Ekadiyanto, MSc., Amyrna Leandra Saleh (Center of Javanese Studies, Faculty of Humanities, University of Indonesia), Bobby A. A. Nazief, Ph.D and Computing Research Group (GCRG) UI, and UI Mercator Office & Multimedia Lab.

REFERENCES


