International Journal of Advanced Technology in Engineering and Science
Vol. No.5, Issue No. 08, August 2017
www.ijates.com
ijates
ISSN 2348 - 7550

AN IMPLEMENTATION OF FOOTBALL CLUB SYSTEM USING REMOTE METHOD INVOCATION (RMI) UNDER DISTRIBUTED COMPUTING FOR ASIA PACIFIC UNIVERSITY, MALAYSIA

Teoh Wan Qi

Student, BSc (Hons) in Software Engineering, Asia Pacific University,
Kuala Lumpur, Malaysia

ABSTRACT

Nowadays systems can be developed using high level languages in order to produce reliable and user friendly. There are various supporting techniques and methodologies supporting developers to implement quality systems. Furthermore here the JAVA programming language takes place through Remote Method Invocation for developing football club system using socket and serialization programming under the distributed computing environment. The developed football club system replaces the problem statements whereby providing appropriated solutions in the proposed functionalities. Also this paper distinguished differences between three distributed technology comparisons such as CORBA, COM and DCOM. The descriptions about cloud computing, grid computing and virtualization also explained. Benefits of Internet Communication Engine (ICE) described in this paper and provided future enhancement.

Keywords:COM, CORBA, DCOM, RMI, Virtualization, Cloud Computing and Grid Computing

I . INTRODUCTION

This paper will be consisting **two** major sessions: System Implementation Report and Research Documentation. The System Implementation Report will be consisting of all the necessary reports for the implementation of the APU Football Club System. Reports such as list of functionalities, Executions, test plans and additional features of the system will be listed. Next, the Research Documentation will be consisting on researches on the comparisons of the DCOMS technologies with application limitation and future enhancement, elaborations of Virtualization, cloud and grids computing in DCOMS and the benefits of ICE technology.

1.1 Aims and objectives

The aim of this project is to analyze, design and implement a system on football system for APU sport club. The objective of this system are listed below:

a. To support team manager to manage player team
b. To allow team manager to view the announcement
c. To enable admin to create announcement
d. To enable admin to manage support team for the football player team
e. To allow admin to manage the registration of players

II. PROJECT BACKGROUND

The football club for the APU student is established to manage students that are interested in participating the club activities. The main goal of the club is to provide opportunities for students that are interested in football regardless gender and to form teams in a safer and discriminatory environment that builds the value of teamwork in the sense of community. Therefore, the club has proposed a volunteer management system to manage the entire activities in the club, which is called as the Team Support System (TSS).

The TSS program will be supporting team management, where it can arrange and place a team together with a supporting team which including a first aider, coach, assistant coach and team manager. The support team can use the system to run their daily team operations like coordinating team strategies and manages team report more smoothly with the motto of "less work for more people".Since the APU group will be organizing sports event every year, the TSS will be also manages all kind of club information such as registration and announcement in order to provide more flexibility for the sport clubs

III. ASSUMPTIONS

There are few assumptions that are needed to be considered before the implementation of the system. It is assumed that the TSS system is first proposed and implemented for the APU football club and will be proposed for the usage of the other sports club as well in future. The announcement might be not only limited to the activities in the APU groups, but also information about the competitions that are organized by the outside organizations.

IV. DESIGN

The functional requirements are stated in the use case diagram. The actors involve are admin and team manager. Below is the list of functional requirements: -

a) Admin must able to register the team and support team.
b) Admin must able to create new announcement in the system
c) Team manager must able to assign team.
d) Team manager must able to view the announcement for the club.
e) Team manager must able to view the team list including the player team and support team.

Fig.1-Use case diagram

V. USER GUIDE

First, user need to login into the system using unique username and password since each user will be seeing a different interface in the system.

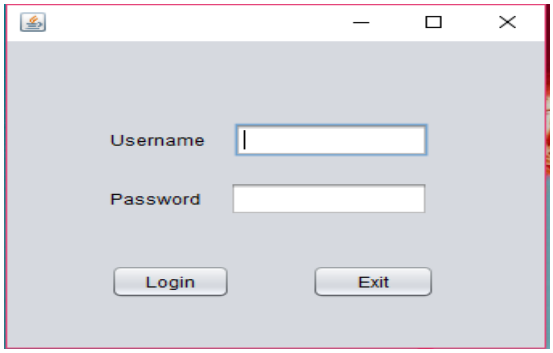

Fig. 2 -Login interface

As different user will have different interface, different ID will show different main page.

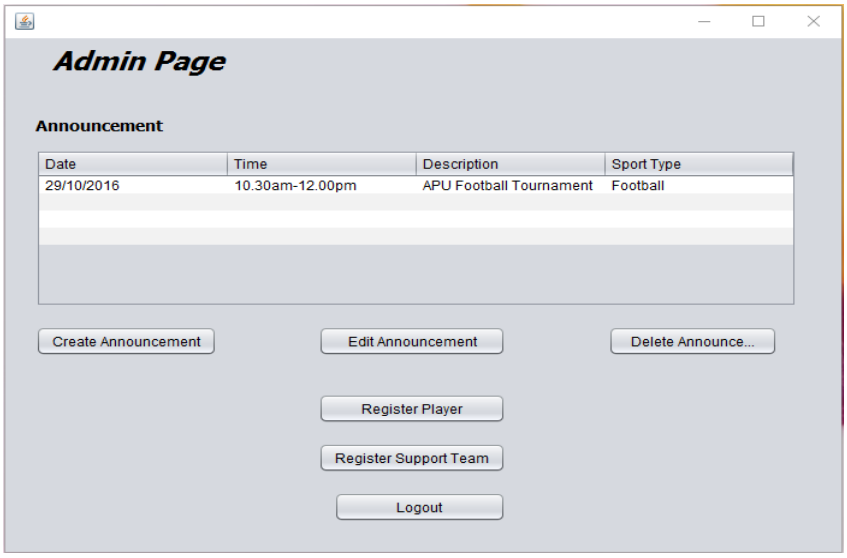

Fig. 3 -Admin main page

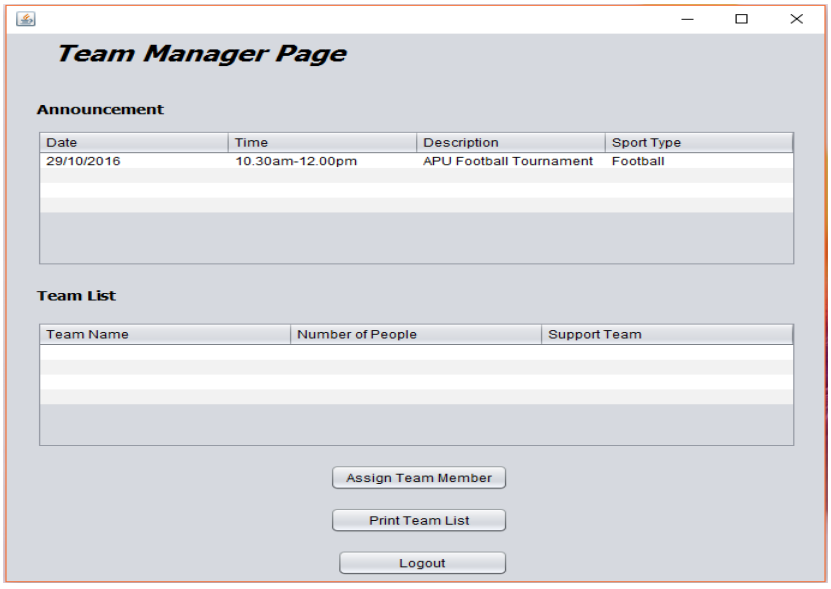

Fig. 4 -Team manager main page

For admin, he/she can create announcement received from the APU or outsource and record in the system to notify the team manager. Other than that, admin also can register player and support team. The edit and delete function can be done directly from the table. To search, simply key in the Student ID and click search, the system will appears the data of the student if it is registered, else error message will be triggered.

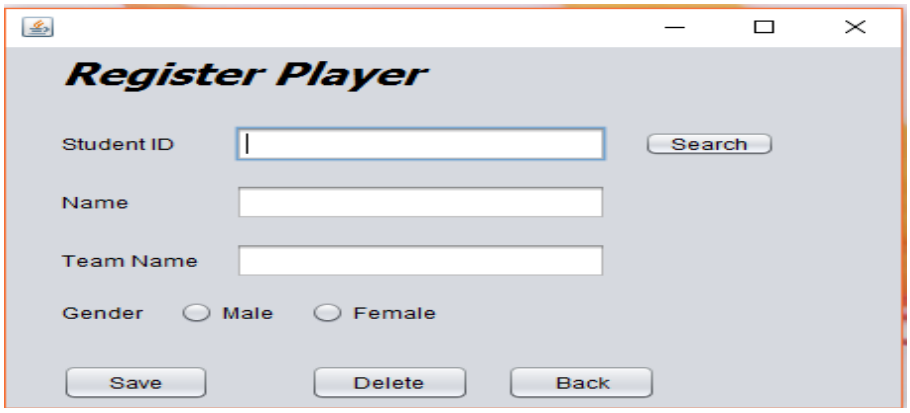

Fig.5 -Create player form

Fig.6-Create support team form

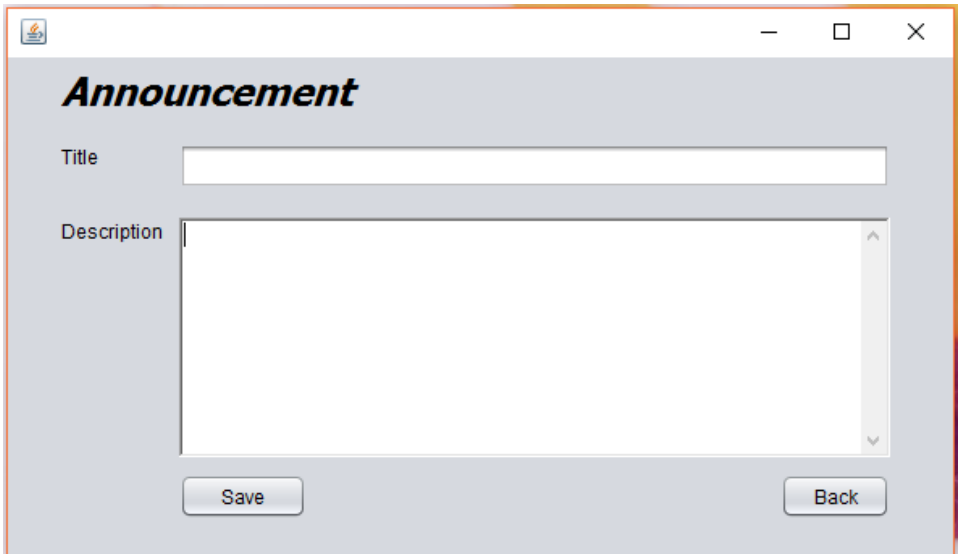

Fig. 7 -Create announcement form

After the user finish his/her transaction in the system, the user need to be logout from the system. The system will then direct back to the login page, enable another user login to use it. There are some functions where pre conditions need to be considered. The players need to be register first before the manager can view the team list and print it. Since announcement need to be update frequently, only the announcement with expired dates can be deleted.

VI. TESTING

Based on [1], software testing is a process of executing a program or application with the intent of finding the software errors and bugs. It can be also a process of validating and verifying that the software products meets the business and technical requirements, work as expected, and can be implemented with the same characteristic. The method of testing that will be using in this system is manual testing, where no automated tools are used. In this method, tester will act as an end-user and identify the error of the system [2]. Through different type of software testing, unit testing is used as the test plan of the system where unit testing is a software development process in which the smallest testable parts of an application.

VII.CONCEPT OF DISTRIBUTED COMPUTER SYSTEM

7.1 Remote Method Invocation (RMI)

RMI is an API that provides a mechanism to create distributed in Java (www.javatpoint.com, 2016). RMI applications often comprise two separate programs, a server and client, where normally a distributed system have. The responsibility of a distributed object applications is to locate remote objects, communicate with remote object and load class definitions for objects that passes around [3]. In terms of locating remote objects, applications can use different mechanisms to obtain references to remote object and alternatively, it can also pass and return object references as part of other remote invocations [3]. Other than that, all the particulars of communications are handled by RMI and it also provides mechanisms for loading an object's class definitions as well as transmitting object's data. Below is the architecture of RMI:

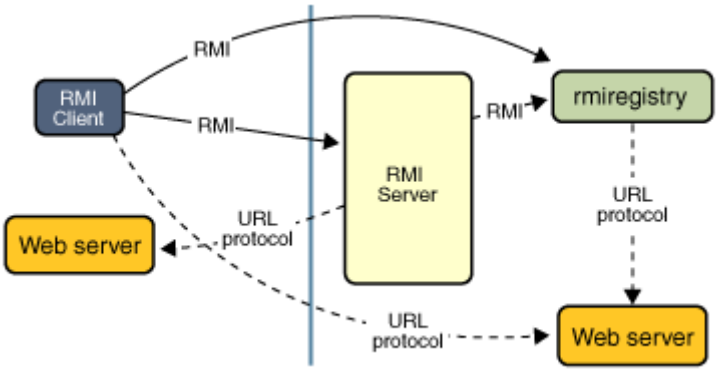

Fig. 8-RMI Architecture [3]

RMI is built from 3 abstract layers, which are the stub and skeleton layer, remote reference layer and the transport layer [4]. The stub and skeleton is responsible for the communication between client and remote object where the stub will be doing initiation of connection with remote Virtual Machine (JVM), marshals the parameter to the remote virtual machine (JVM), wait for the result, reads the return value and finally return the value to the caller. On the other hand, the skeleton acts as the gateway of the server side. While it receives the

incoming request, it will read the parameter for the remote method, invokes the method on the actual remote object and finally, transmit the result to the caller. The mechanism of the stub and skeleton protocol is as shown as below

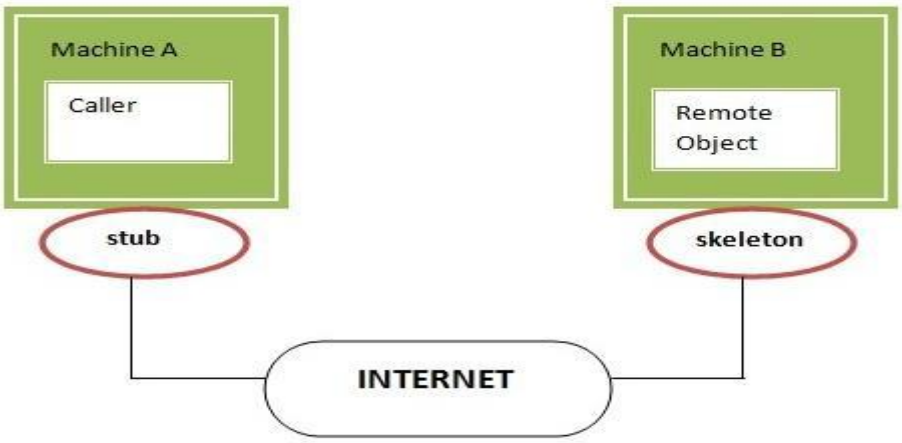

Fig. 9: Stub and skeleton protocol (www.javatpoint.com, 2016)

The transport connection layer act as a portal to set up and manage the request, whereas the remote reference layer behave differently depending on the parameter that passed by the calling program [5]. The RMI can be implemented with the code below. The code is divided into client side and server side.

7.2 Socket

Based on [6], socket is programming abstraction which is used to implement low-level IPC. It is a peer-to-peer communication "endpoint" abstraction where it hides the details of the network for the programming communication situation. An endpoint is a combination of an IP address and a port number where every TCP connection can be uniquely identified by its two endpoints. The communication between client and server are communicate by reading or writing to and from their socket. A socket is also bound to a port number so that the TCP layer can identify the application that data is destined to be sent to. The mechanism is as shown below

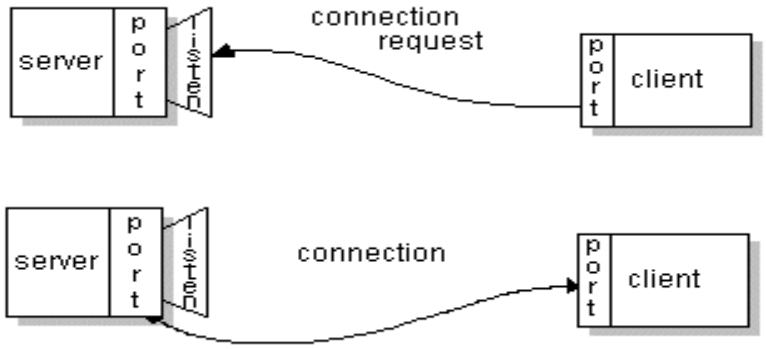

Fig. 10 - Socket mechanism [3]

In Java, the java.net class package in the Java platform provides a class, Socket for the implementation one side of a two-way connection between Java program and another program on the network. There are two types of sockets, which are client-side socket and server-side socket. In the client-side, the socket is created to send a request, receive the response and once it's done, the socket is destroyed. As for the server-side socket, it will

behave more like a dispatcher as other than sent or receive any data, they simply listen for connection instead on the host and port that the socket server is connected to [6].

7.3 Serialization

Serialization is the process of turning an object in memory into a stream of bytes so to allow the programmer store it on disk or send it over the network [7]. Serialization involve in the conversion of public and private member of an object including the name of the class and assembly into a stream of bytes, which will be written into the data stream [4]. Serialization is used when the large amount of data has to be stores in flat files and retrieved at a later stage [4]. If serialization is not considered to be use in this case, the coding will be too complicated as the data structure will be very complex. In Java, the java.io,serialization interface can be used to implement serialization. Besides that, the saving session state in ASP.NET, is also using the serialization method. It is also mostly used in sharing data across the network without restricting the application on usage of data [4].

VIII. VIRTUALIZATION, CLOUD AND GRID COMPUTING USAGE

8.1 Virtualization

According to [8], virtualization is the creation of virtual version of somethings such as an operating system, a server, a storage device or network resources. It also provides a level of logical abstraction that liberates applications, system services, and the OS from being tied to a specific piece of hardware. Based on the director of product marketing in VMware, Mike Adams says that virtualization enables businesses to reduce costs, since it allows running of multiple operating systems and applications at the same time, hence, increasing the efficiency, utilization and flexibility of existing computer hardware [9].

8.2 Server Virtualization

Based on [10], server virtualization is the partitioning of a physical server into virtual servers to maximize the server resources. The server resources that are masked included the number and identity of individual physical servers, processors, and operating systems from the server users [11]. The virtual environments are sometimes called virtual private server, but they are also known as guests, instances, containers or emulations. There are several ways to create server virtualization, which are full-virtualization models, para-virtual machine and OS-Level virtualization.The full-virtual machines are based on the host and guest paradigm where each guest runs on a virtual imitation of the hardware layer. It uses hypervisor, a software where it enables interaction directly with the physical server's CPU and disk space. It also serves as a platform for the virtual servers' operating systems. The responsible of hypervisor is to keep each virtual server completely independent and unaware of how the other virtual servers running on the physical machine [11]. One can think that hypervisor as a very thin OS that is optimized for hosting guest virtual machines [12].

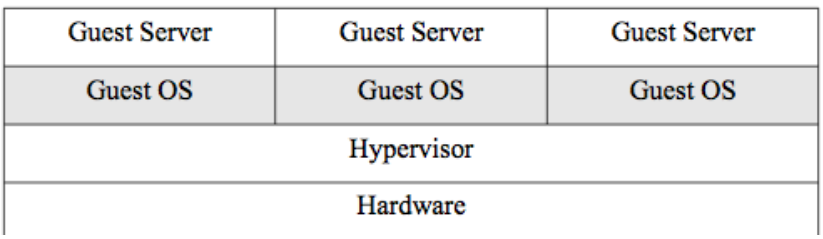

Fig. 11- Full virtualization architecture [12]

In para-virtualization, unlike full-virtualization, the guest servers are aware of one another. Therefore, a para-virtualization does not need too much processing power to manage the guest operating system as every OS are aware of the activity and demand of the other OS in the physical server [13]. Below is the architecture of the para-virtualization:

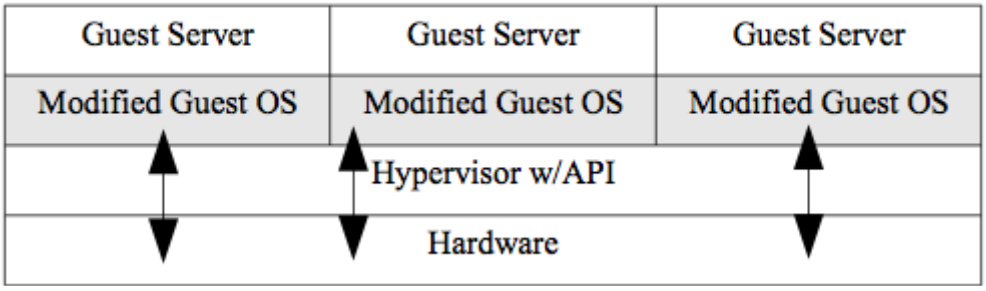

Fig. 12 - Para-Virtualization architecture [12]

Other than that, operating system level model is also one of the server virtualization model. In the OS level model, the host will be running a single OS kernel as its core and exports operating system as the host with different distributions of the same system are allowed [11]. Hence, the biggest limitation of this approach is that all the guest servers must run within the same operating system, as each virtual server remains independent from all the other operating system [13].

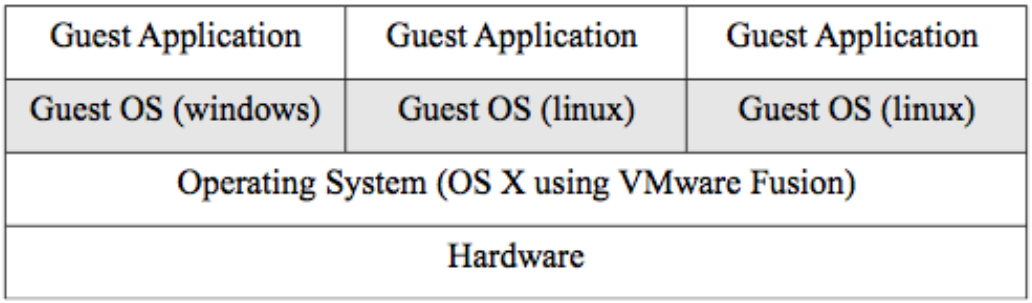

Fig. 13 - OSLevel Model architecture [12]

8.3 Cloud Computing

Cloud computing is a style of computing where massively scalable and flexible IT-related capabilities are given in the form of service to the users through internet technologies [14]. According to David Cearley, the VP of Gartner, he said that cloud computing is a major technology trending that has permeated the market over the last two years. Therefore, cloud computing has been a popular topic in the technology industry. Since the cloud service may include infrastructure, platform, applications and storage space, they do not need to build extra hardware or infrastructure on their own, hence save cost since there is no capital investment and the use of resources is metered by the provider. There are 3 types of cloud computing, which are public cloud, private cloud and hybrid cloud.

Based on [14], public cloud is a cloud infrastructure hosted by service providers and made it available to public. Examples of public clouds Windows Azure Services and Google AppEngine. Usually public clouds provide the best economy scales, as they are inexpensive to set-up as the hardware, application and bandwidth costs are covered by the provider. However, public clouds might not be suitable for every organization as the model will limit the configuration and security [15].

The private cloud is a data center architecture that is owned by a single company that provides monitoring, flexibility, provisioning, scalability and automation. Comparing to public cloud, privatecloud could be more expensive. Therefore, it is not a wise choice for an average Small-to-Medium sized business. However, it is more suitable to be used in the large companies, as private cloud has the highest level of security control [16]. Hybrid cloud is the combination of 2 or more different types of the cloud services where every cloud still remains as single entity but combined to provide an advantage of multiple deployment model [14]. Through hybrid cloud, companies can internally maintain the control of the private cloud while relying on the public cloud. Hybrid cloud are also helpful throughout the predictable outages such as blackouts.

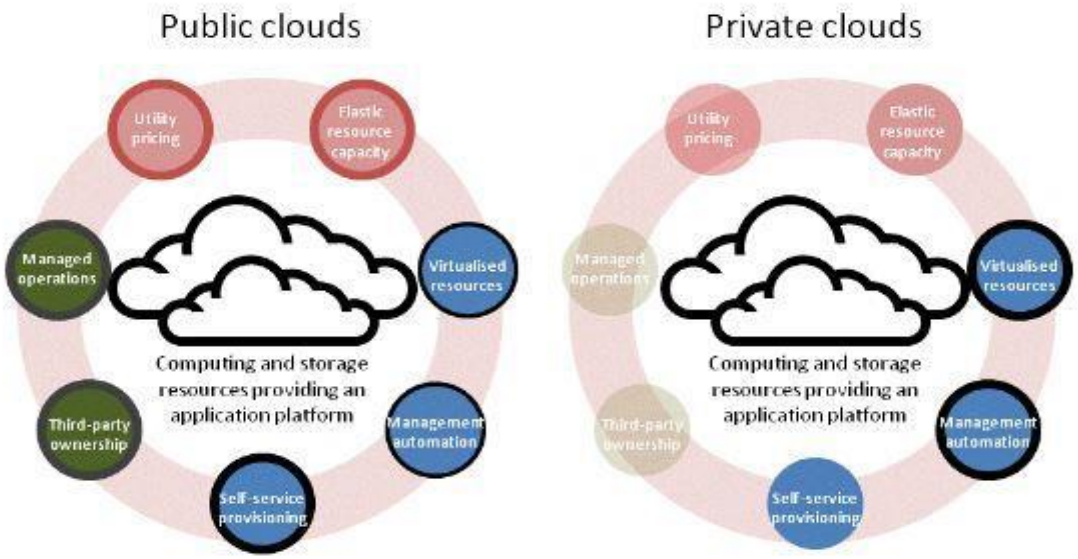

Fig. 14- Example of hybrid cloud [16]

8.4 . Grid Computing Usage

Grid Computing is a collection of resources from various computers in a network to a single problem, where normally with the one requires a large number of processing cycles or access to large amount of data (Jimpinto.com, 2016). In grid computing, servers or personal computers will run independent task and loosely linked to Internet or low-speed network, instead it will connect directly to scheduling system [11]. The model architecture of a grid computing system is not fixed, as it can be just a collection of similar computers running on the same operating system or as complicated as inter-networked systems comprised of every computer platform. Grid computing is considered as a type of special Page | 34 distributed computing technology. In distributed computing, different computers will share one or more resources within the same network; whereas for an ideal grid computing, every resource is shared which turns a computer network into a powerful supercomputer. Most project with grid computing have no time dependency, as they may deploy across multiple countries and areas. One of the earliest example of a project that is using grid computing is the SETI@home project. The purpose of SETI project development is to analyzed data gathered by radio telescopes in search of

the evidence for intelligent alien's communications [13]. Grid computing is so huge even it had brought itself to address other projects such as drug-candidate matching and genetics researches [11].

IX. BENEFIT OF INTERNET COMMUNICATION ENGINE (ICE) TECHNOLOGY

The internet communication Engine (ICE) technology is an open-source comprehensive RPC (Remote Procedure Call) framework found by ZeroC with the support different programming language such as Java, JavaScript, C++, C#, Python and partially supports Ruby and PHP. It is also a platform that promotes distributed computing that allow users to run it in different operating system such as Linux, Windows, OS X, Docket, Andriod and more. ICE also provided several services such as IceGrid, Glacier2, IceStorm and more. ICE promotes data security where it only requires few lines of code implementation to get it started and it also can make synchronous and asynchronous invocations using TCP, UDP, SSL/TLS, and WebSockets. According to [17], the main design objective for the ICE is:

X. Comparison between corba, com and dcom

TABLE 1-comparison between CORBA, DCOM and COM [18]

COMPARISON	CORBA	DCOM	COM
Programming language	CORBA is a specification therefore any programming language that has ORB implementation and language mapping.	DCOM support multiple language since it is binary standard.	COM support multiple language since it is binary standard.
Object-oriented support	Support multiple inheritance at interface level	DCOM Supports multiple inheritance	Provides many same service
Platform	MVS, UNIX, Windows(all), Macintosh	Windows NT; future support for Windows (all), Macintosh, UNIX, MVA	Platform-independent, foundation technology of Microsoft's OLE
Garbage Collection	No garbage collection	Perform garbage collection	Perform garbage collection
Invocation of object	Support static and dynamic	Support static and dynamic	Support dynamic
Hardware support	Support nearly all hardware platform	Windows platform, COM platform	Windows platform
Security	Defined own security service	NT LAN Manager Security	RPC security mechanisms

XI. LIMITATION

Since the user of the TSS is only limited to admins and team manager, the other users such as students and coach cannot register themselves freely through the system. Moreover, the system are designate to use in APU, as this is the system for the APU sport's club only.

XII. FUTURE ENHANCEMENT

Foresee on the potential of the TSS in future, advanced feature can be implemented. It is suggested that the registration process of the system can be done through biometric method such as detecting fingerprint. The system will automatic detect the user of the fingerprint and register them into the system. Once the registration is completed, the users can get instant acknowledgement through SMS to notify them.

Other than that, the manager can also make payment for the events through the system. Payment method such as credit cards can be also done through the system. The payment transaction can also have been recorded into the system so that manger can check the payment history.

XIII. CONCLUSION

Through this assignment, I have gained knowledge on the definition of distributing computing, its importance, and the various types of distributing system. As this is only a basic proposal for the APU football club system, there are still many features and functions can be added in future, as we can see the potential of the system to be used outside of APU. While doing the research on the ICE technology, I think that it is very useful and convenient, as most of the programming languages and platform actually supports it, allowing efficiency and effectiveness while managing the network computing. Last but not least, this report also had taught me how distributed system applies in our real life.

XIV. ACKNOWLEDGMENT

The author would like to share gratitude to Mr Umapathy Eaganathan, Lecturer in Computing, Asia Pacific University, Malaysia for the constant support and motivation which helped me to participate in this International Conference and also for journal publication.

XV. REFERENCES

[1] Istqbexamcertification.com. (2016). *What is Software Testing?*. [online] Available at: http://istqbexamcertification.com/what-is-a-software-testing/ [Accessed 7 Oct. 2016].

[2] www.tutorialspoint.com. (2016). *Software Testing - Types of Testing*. [online] Available at: https://www.tutorialspoint.com/software_testing/software_testing_types.htm [Accessed 7 Oct. 2016].

[3] Docs.oracle.com. (2016). An Overview of RMI Applications (The Java™ Tutorials > RMI). [online] Available at: https://docs.oracle.com/javase/tutorial/rmi/overview.html [Accessed 7 Oct. 2016].

[4] Techopedia.com. (2016). *What is Serialization? - Definition from Techopedia*. [online] Available at: https://www.techopedia.com/definition/867/serialization-net [Accessed 7 Oct. 2016].

[5] TheServerSide. (2016). *What is Remote Method Invocation (RMI)? - Definition from WhatIs.com.* [online]

Available at: http://www.theserverside.com/definition/RemoteMethod-Invocation-RMI [Accessed 7 Oct. 2016].

[6] Computer Science: Source. (2009). *Distributed Systems – Socket Level Servers*. [online] Available at: https://computersciencesource.wordpress.com/2009/09/10/year1-distributed-systems-socket-level-servers/ [Accessed 7 Oct. 2016].

[7] Stackoverflow.com. (2016). *What is serialization?*. [online] Available at: http://stackoverflow.com/questions/633402/what-is-serialization [Accessed 7 Oct. 2016].

[8] SearchServerVirtualization. (2016). *What is virtualization? - Definition from WhatIs.com*. [online] Available at: http://searchservervirtualization.techtarget.com/definition/virtualization [Accessed 5 Oct. 2016]. Page | 45

[9] Angeles, S. (2014). Virtualization vs. Cloud Computing: What's the Difference?. [online] Business News Daily. Available at: http://www.businessnewsdaily.com/5791- virtualization-vs-cloud-computing.html [Accessed 5 Oct. 2016].

[10] Beal, V. (2016). What is Server Virtualization? Webopedia Definition. [online] Webopedia.com. Available at: http://www.webopedia.com/TERM/S/server_virtualization.html [Accessed 6 Oct. 2016].

[11] Rouse, M. (2016). *What is server virtualization? - Definition from WhatIs.com.* [online] SearchServerVirtualization. Available at: http://searchservervirtualization.techtarget.com/definition/server virtualization [Accessed 6 Oct. 2016].

[12] Userpages.umbc.edu. (2016). *Distributed Systems.* [online] Available at: http://userpages.umbc.edu/~jianwu/is651/651book/is651-strapdown.php?f=is651- Chapter13.md [Accessed 6 Oct. 2016].

[13] HowStuffWorks. (2008). How Server Virtualization Works. [online] Available at: http://computer.howstuffworks.com/server-virtualization2.htm [Accessed 6 Oct. 2016].

[14] DeZyre. (2015). Cloud Computing vs. Distributed Computing. [online] Available at: https://www.dezyre.com/article/cloud-computing-vs-distributed-computing/94 [Accessed 6 Oct. 2016].

[15] Asigra.com. (2016). Cloud Types: Private, Public and Hybrid | Asigra. [online] Available at: http://www.asigra.com/blog/cloud-types-private-public-and-hybrid [Accessed 6 Oct. 2016].

[16] Čandrlić, G. (2013). Types of Cloud Computing Explained | GlobalDots. [online] GlobalDots - CDN, Security and Performance Solutions. Available at: http://www.globaldots.com/cloud-computing-types-of-cloud/ [Accessed 6 Oct. 2016].

[17] Zeroc.com. (2016*). Ice is Everywhere*. [online] Available at: https://zeroc.com/products/ice#everywhere [Accessed 4 Oct. 2016].

[18] Sorensen, C. (n.d.). *A Comparison of Distributed Object Technologies*. [online] Available at: http://www.idi.ntnu.no/~carlfrs/publications/essay2001-calle.pdf [Accessed 6 Oct. 2016].