A Comparative Study in Remote Techniques and Event-Based Invocations

Musab A. M Ali[1], Hamood Shehab Hamid[2], Ihab A. Najm[3], Shahad Al-yousif[4], Nooritawati Md Tahir[5]

[1]Faculty of Information Science and Engineering, Management & Science University, Shah Alam, Selangor, Malaysia;
[2]Middle Technical University, Electrical Engineering Technical College Computer Engineering Techniques Department Baghdad/Iraq.
[3]Faculty of Engineering , Tikrit University, Iraq.
[4]Computer Engineering Techniques Department, Dijlah University College, Baghdad, Iraq
[5]Faculty of Electrical Engineering, Universiti Teknologi MARA (UiTM), Shah Alam, Selangor, Malaysia
drmusab@msu.edu.my, drhamood@mtu.edu.iq engihabit@gmail.com, shahad.alyousif@duc.edu.iq, noritawati@ieee.org

Abstract— **In this paper, a comparative study of remote techniques as sturdier schemes for creation distributed system depend on applications is presents. Demand the object active already in the process been approaches for an object occupied in another process. This area tolerates to be not in the same environment and emphasis on the modern technique event-based be intensely more but this time with the same object could respond to change concealment performance as another object. Those applications pose of concurrent programs executing in various procedures. This article aims to explain Event-based RPC and RMI methods. Each methodology with differentiation among the techniques and clarify how exchange from the local procedure to the remote procedure or from the process to another one.**

Keywords- *Event-Based, RPC, RMI, Distributed System.*

I. INTRODUCTION

The development of traditional process of the remote procedure call (RPC) scheme is permit the client application to request model to the remote process in server running applications and distinct process [1, 2]. Early 1990s, the development of programming method has contributed to concede the verity procedures and interconnect each other's, under remote method invocation techniques. The remote method invocation (RMI) is the improvement version of local method invocation that grants the purpose across a procedure to call techniques of the existing methods [3]. To permit the pattern in acquire notice objectives, the even based techniques could be itemized. The fast growth of the pattern granted the perform distribution of even based programs. The term RMI denote a generic way this guarantee there's no scrabble about, particularly, sample of RMI is a Java RMI. The majority of existence of distributed system is designed in object-oriented patterns; obviously, RPC is related with RMI. Therefore, this manuscript focuses on event based and RMI, both implemented in the distributed objects [4].

II. MOTIVATION

Remote Procedure Call (RPC)

- Supply a practical interface for distributed object (e.g. remote) services
- In order make distributed environment of service
- A transparent to serve programmers
- Consequently, RMI no longer considered a good technique.
- RPC + Object oriented
- Consent objects lying via one procedure to call up process of an entity remaining through numerous methods notice and events
- RMI is able to improve in internal of objects method.
- Ability of events in object internal, however, attributes alternatively, methods to work as another one with assorted object.
- Notice of events is basically not concurrent, also to conclude by their receiver [5, 6].

III. MIDDLE WARE

The applications that allow programming pattern over the elementary making slabs of procedures and message move is named middle ware [7]. The middle ware layer applies protocols depend on messages among procedures to grant a high-level concept such as remote invocations and events, as show in figure 1 for example, the RMI concept constructed on request replay protocol.

A significant feature of middle ware is providing location transparency and isolation of details through connecting protocols, chiefly, operating systems and computer hardware. Some middle ware permits the distinct mechanisms to establish in diverse programming languages [8].

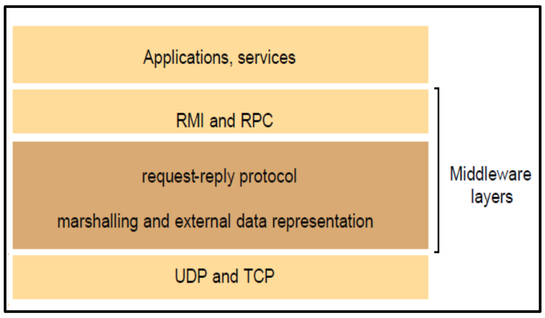

Fig. 1. *Middleware layer*

In the location transparence, the client of RPC invokes process could not update the method execution in the same or different process by another computer. In this case the client requires to knowing the position of server in charge [9] In the communication protocols which support the middle ware concept, the implicit protocol of transport will isolated in this case and the replay-request protocol could be applied over UDP or TCP.

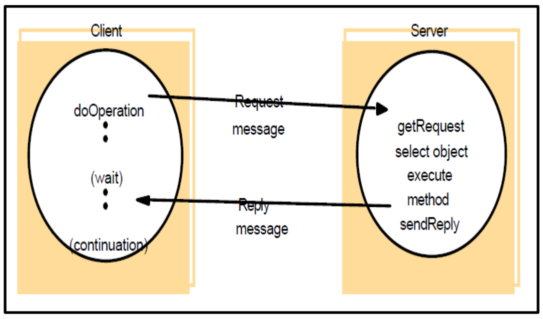

Fig. 2. *Request-reply protocols*

Three standard agreements for external information symbols could be found in the computer hardware which used the marshalling and none marshalling text [9, 10]. This will put out and hidden the different due to architecture of hardware such as ordering of byte. In case of high-level abstraction which presented by middleware layer are isolated of the implicit the operating systems [11]. To provide more distribution applications, some middleware is designed by more than language of programming. In the server side, CORBA could permit to client's implementation in one language for call process. These programs could implement in other language which will different from language of clients by using interface definition language in order to define the interfacing.

IV. RPC

The similarity of RMI with remote procedure call in term of client programming could permit to the server process for running in other programs. To provide sequence of RPC, any server could be client by additional server. Hence, the server progression will define in the interface of services and the viability of methods for remote invoke. Actually, the service is relatively alike to single one which had attributed. Hence, it does not provide the remote techniques [3, 4]. Thought, the RPC and RMI could be implemented to achieve new options of call. In general, the RPI could apply in the protocol of replay request with short elimination of location from the message requested as similar replay in case of neglected the reference field [12]. The software which sustenance RPC is illustrated in Figure 3 as it would be for procedural programing language, like C language, which does not present object-oriented concept, therefor, the primary RPC systems were C based [13].

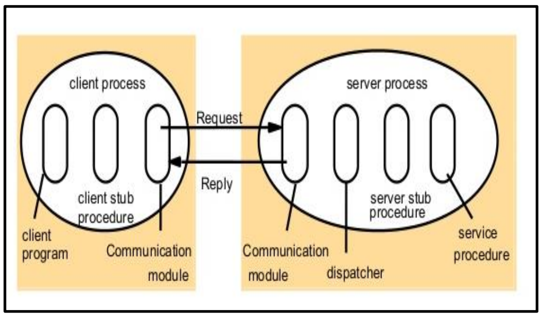

Fig. 3. *client with server structure in RPC technique*

Subsequently, procedure invoke is careless by mean of references of object and the objects itself. The incoming clients to the server contain one process of stub to every method of service interface. Hence, the regulation of stub method is close to proxy form which performs as limited technique with client. So, the place running invoke is marshal methods is individuality and complain with the message send via server connection equipments as none marshals repetitions. For more capability of interface, the server steps consist a sender with stub and one service process for every method. Corresponding to process ID in the demand message, the sender will choose one server stub methods. The server stub process looks like close to skeleton technique in none marshals form which complains with demand message. The achievements of service process could be summarizing as in [14] and this includes RPC call abstraction procedure among much process, realization server procedure of sub client side of proxy and the marshal parameters locate sub client side. Since the side of server will receive the message, the stub unloads the parameters of marshal and executes the server side. The remote procedure call steps mentioned in [2, 6] is listed below:

1. *RFC (1831)*

Proposed by [15] which represent the sun RPC is intended for client and server corresponding with sun NFS framework documents. Currently, the RPC called ONC-RPC and supplied as a piece of different sun with other UNIX framework likewise accessible establishments with NFS. By mean of XDR interface language, the RPC system is provide the interface complier under title (rpcgen) to use with C language. The language of XDR was considered to specify external information which projected to develop into interface definition language as explained in. A set of specific method description will support the description of types which might be used to describe the interface of services in RPC methods. The entry of data is more primitive in comparison between CORBA IDL or Java.

2. *RMI*

The remote method invocation is similar to RPC additional RMI = RPC + object orientation. Every procedure includes compilations of object which could receive a limited chant only. The RMI is a technique of invocation among objects with dissimilar processing in the same computer or different computer. The object could receive remote invocation as remote objectives as shown in Figure 4 [4].

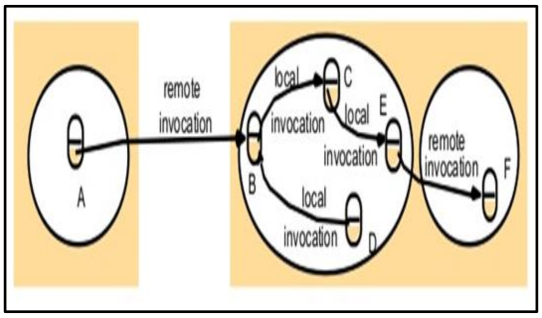

Fig. 4. *Remote and local method invocations.*

The interface description dialect (IDL) will provide by CORBA framework which utilize the characterization remote interface. The remote articles kinds and customer projects may be actualizing in several dialects. For example, C++, Java or Python in IDL complier is accessible. The COBRA customers require not utilize the same dialect as the remote protests keeping in mind the end goal to conjure its systems remotely [16]. Nonetheless in the distributed issue, the objects intricate in a series of associated calls may be addressed over various procedures. Once, call reaches the process, RMI Is taking role, in addition, the reference of remote Object required existing demander [17]. The object A required holding the remote reference object of object B as illustrated in Figure 4. The remote reference object could be gained as accused of remote technique calls. Additionally, in Figure 4, the object A acquires a reference of a remote in F as of B object [6].

Several applications used the Garbage Collection (GC) notation that is used to terminate unused application inside memory at running time, to take advantage of releasing memory index, however, GC depending on index counts. For instance, Java using GC, also, linked RMI methods supporting GC for remote objects; in fact, there's interoperation existing local garbage collector, on the other hand, the model which carried out from remote GC [6, 7]. Some cases remote call failed establishment objects request through various procedures, in fact, the procedure include crashed or occupied remote objects. In addition, the call process or a notification messages is lost, consequently, RMI requires to flag up exceptions like time out; this matter happens during method call [6, 7].

A) *Effects of extension:*

The Request-replay protocol of RMI call semantics based on.

B) *Retry request message:*

re-transmit the demand message as long as whichever a reply is received or server is expected failure.

C) *Duplicate Filtering:*

this event occurs when re-transmission applied, to filter out identical requirements for service [4, 5].

D) *Re-transmission:*

retention a history of messages contents to in order to provide lost result e retransmission wanting re-running the process in server. Mixing of these select chief different possibilities semantics for trusted of remote call as shown by the invoker moreover, presents the selection of interest, with consistent terms for the request's semantics process. Moreover, local method call, the semantics is once [1, 4, 10]. Semantics turned on when the fault tolerance events is not used [6].

E) *At least once call:*

regulation of at least once demand, the caller is notified, however, caller recognizes the method was performed at least once, otherwise an exception notifying there's nothing received. It can be accomplished by the re-transmission of invoke message, that cover the inattention crash of the call or message notifications. Minimum call occurred from the following kinds of failures at least one [10].

F) At most once invocation:

same rules of at least one case; therefor it can be accomplished by when fault tolerance events. Use revises leads any failures of the caller or notifying message. The group of error acceptance events prevents random crash by guarantying for every RMI method can't run more than one time. The CORBA and RMI caller rule are at mainly just the one time. Nevertheless, CORBA permits rules for demand methods which not reappearance notifying, additionally, the Sun RPC supports one or more invocation call [6, 10].

V. PROGRAMMING OF RMI

The requirements of client programs are to obtain a reference of remote objectives. The folder is a service that keeps a mapping from text title to the reference of remote objectives. The registration of services is sending abroad according to folder of Java RMI registration and CORBA services [4]. Many selections such as thread per object and invocation have to permit for concurrent executions.

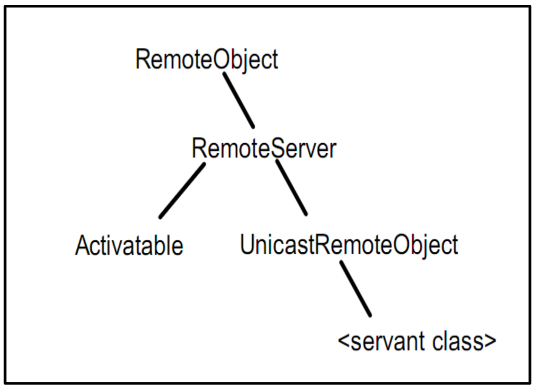

Fig. 5. *Class supporting Java RMI*

VI. RMI REALIZATION

Many objects and procedure are concerned to achieve a remote method invocation as illustrated in Figure 5. The application of object A invokes a technique in remote applications level B to hold the remote reference of objectives [1, 7]. For communications section, two collaboration modules carrying out the transmit request and message replay among server and client by using three earlier items to identify the type of messages.

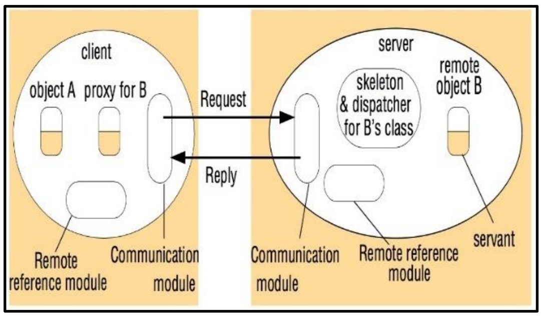

Fig. 6. *RMI presentation in proxy and skeleton*

This will request the object of remote reference to invoke in this case [10]. The responsibility of remote reference section is to translating among remote object and local references as well as to create the references of remote object. In order to assist this responsibility, the remote reference module has a remote object table in every process to record the correspondence among local object reference and remote object reference for that process [6]. The roles proxy under RMI software could be summarized by adding to its table and the remote reference module has asked for equivalent local object reference when arrived in request or replay message by either proxy or remote. The servant is a type body which provided to remote objective to hold the remote request approved by the equivalent skeleton and exist inside the server process [6, 10]. The software of MRI contains a layer between the application level objects and remote reference module as well as communication [4]. The responsibility of proxy is to provide remote invocations transparent to client using performing such as invoke of local object. But it hides the information of remote object reference and marshaling arguments as well as none marshaling results. Additionally, it sends and receives the client messages. Any server should have one dispatcher and skeleton in every type which represents the remote object. The remote object B in exist example is the server dispatcher and skeleton. The responsibility of dispatcher in is to receive the request message from communication module which is used the procedure of selecting the suitable techniques in pass, request and skeleton message. A skeleton is a group of remote objects realized in procedure for remote interface which quite different from procedure of servant.

VI. EVENTS –BASED AND NOTIFICATIONS

If one object responds to change the occurring in another object is so called the idea of events and the synchronous of their receiver is named notifications. The events of performing an object by using action of button with mouse or via keyboard are event that cause change in the object to keep up the state of functions [2]. A wide range of different application variety could be used in the events and notifications. To explain any change of object, an event could be used and specify the exacting user performance of the achievement on object. Every duplications of object to exacting kind of event is related to subscribe them and receive announcement when they happen. However, subscribe is decoupled from object knowledge event due to diverse user activation at many times.

VII. THE TYPE OF EVENT

Different type of event could be generated from an event source. Every event has characteristic that identify the information about this event like the name of object which generated, operated, parameters and event time. The attribute and kinds of event could be used to subscribe and notifications of event. The subscribing to the event is sometime specify the event and modified with criterion and attributed the values [2]. Figure 7 shows the structure of distributed event notifications.

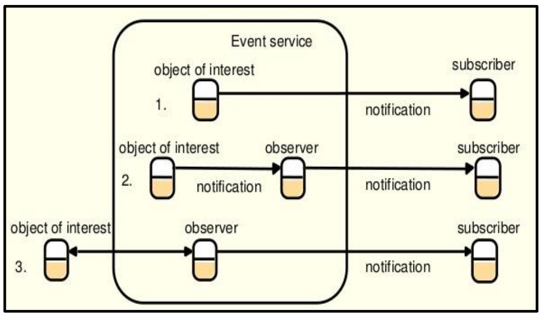

Fig. 7. *Distributed Event Notification Structure*

The major part in the event structure is the event services that keep up the database of available event and the attention of subscribing. At any object of attention, the event is available in the service of this event. The event service will inform by subscriber concerning the kind of event interested in. A specific notification will be sent to the subscriber when the event happened.

VIII. CONCLUSIONS

Highlight on sophisticated influences terms like RPC, RMI and Event as well as notifications not easy; its concern for all interest in the remote invocation and distributed objects. For those after these untraditional comparisons, the reader knows much more for each technique and review all ability change from local to a remote in general way because there is three different meaning. Without acknowledgment of user, this service comes from local or remote supplier, last but not least Event-based most important technique from all. Because all ability of RMI and with added same object defiantly have many own events when occurs the event it will look like another object. All these techniques concepts depend on previous techniques.

REFERENCES

[1]" Satoh, I.: 'Mobile Agents': 'Handbook of Ambient Intelligence and Smart Environments' (Springer, 2010), pp. 771-791.

[2]" Maassen, J., Van Nieuwpoort, R., Veldema, R., Bal, H., Kielmann, T., Jacobs, C., and Hofman, R.: 'Efficient Java RMI for parallel programming', ACM Transactions on Programming Languages and Systems, 2001, 23, (6), pp. 747-775.

[3]" Denis, A., Pérez, C., and Priol, T.: 'PadicoTM: An open integration framework for communication middleware and runtimes', Future Generation Computer Systems, 2003, 19, (4), pp. 575-585.

[4]" Engelmann, C., and Geist, A.: 'RMIX: A dynamic, heterogeneous, reconfigurable communication framework': 'Computational Science–ICCS 2006' (Springer, 2006), pp. 573-580.

[5]" Juric, M.B., Kezmah, B., Hericko, M., Rozman, I., and Vezocnik, I.: 'Java RMI, RMI tunneling and Web services comparison and performance analysis', ACM Sigplan Notices, 2004, 39, (5), pp. 58-65.

[6]" Tanenbaum, A.S., and Van Steen, M.: 'Distributed systems' (Prentice Hall, 2002. 2002).

[7]" Lamport, L.: 'Time, clocks, and the ordering of events in a distributed system', Communications of the ACM, 1978, 21, (7), pp. 558-565.

[8]" Meyer zu Eissen, S., and Stein, B.: 'Realization of Web-based simulation services', Computers in Industry, 2006, 57, (3), pp. 261-271.

[9]" Waldo, J.: 'Remote procedure calls and java remote method invocation', Concurrency, IEEE, 1998, 6, (3), pp. 5-7.

[10]" Qureshi, K., and Rashid, H.: 'A performance evaluation of rpc, java rmi, mpi and pvm', Malaysian Journal of Computer Science, 2005, 18, (2), pp. 38-44.

[11]" Hsiao, D., Neuhold, E., and Sacks-Davis, R.: 'Cooperation between autonomous operation services and object database systems in a heterogeneous environment', in Editor (Ed.)^(Eds.): 'Book Cooperation between autonomous operation services and object database systems in a heterogeneous environment' (Elsevier, 2014, edn.), pp. 255.

[12]" Maassen, J., van Nieuwpoort, R., Veldema, R., Bal, H.E., and Plaat, A.: 'An efficient implementation of Java's remote method invocation', in Editor (Ed.)^(Eds.): 'Book An efficient implementation of Java's remote method invocation' (ACM, 1999, edn.), pp. 173-182.

[13]" Wellings, A., Clark, R., Jensen, D., and Wells, D.: 'A framework for integrating the real-time specification for java and java's remote method invocation', in Editor (Ed.)^(Eds.): 'Book A framework for integrating the real-time specification for java and java's remote method invocation' (IEEE Computer Society, 2002, edn.), pp. 0013-0013.

[14]" Bertrand, F., Bramley, R., Bernholdt, D.E., Kohl, J.A., Sussman, A., Larson, J.W., and Damevski, K.B.: 'Data redistribution and remote method invocation in parallel component architectures', in Editor (Ed.)^(Eds.): 'Book Data redistribution and remote method invocation in parallel component architectures' (IEEE, 2005, edn.), pp. 40b-4.

[15]" Saravanan, T. P., A. A. Deshpande, G. S. Rinivasan, D. McConnel, W. Wilson, and P. M. McCulloch, New HI absorption measurements towards six pulsars, Bulletin of the Astronomical Society of India, 23, 512-+, 1995a