iJRASET
International Journal For Research in Applied Science and Engineering Technology



INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 12 **Issue:** XI **Month of publication:** November 2024

DOI: https://doi.org/10.22214/ijraset.2024.65382

www.ijraset.com

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

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 12 Issue XI Nov 2024- Available at www.ijraset.com

A Survey on Modern Inter-process Communication (IPC) Mechanisms

Aniruddha Dhaske[1], Aditya Patil[2], Advait Khairnar[3], Shivam Dhote[4], Minal Deshmukh[5]
Electronics and Telecommunication, Vishwakarma Institute of Information Technology, Pune, India

Abstract: ***In modern operating systems, Inter-Process Communication is the inherent feature in as much as it is the way by which several processes communicate. Today's IPC mechanisms are surveyed along with their performances and applications in contemporary operating systems. This comprises the old procedures such as shared memory and pipes, but also the new procedures like Remote procedure calls (RPCs), Unix domain sockets. Some of the applications include but are not limited to the sharing of system performance and systems scaling consideration.***
Keywords: ***Inter-process communication (IPC), contemporary, modern operating systems, Remote procedure calls (RPCs), scalability.***

I. INTRODUCTION

With the need for a system that is not only fast but also capable of scaling up quickly, the development of Inter-process Communication (IPC) systems has been rapidly heading in the modern computer system. Modern computing environments require an approach considering Inter-Process Communication (IPC) as adopted by all services in consideration of their context and distribution [1]. IPC relaxes system resource constraints by allowing better inter-process communications but entails risks like opening up the possibility for unauthorized access, inappropriate privilege selections, and denial of service [1]. To achieve this, access control, encryption of the secret, and appropriate process scheduling management has to be ensured [2].
IPC helps in the development of organization in OS as well as encompasses the delegation of operations in a program to improve performance [3]. The process of Inter Process Communication (IPC) is accomplished through a number of processes and these are rather different for different systems.

II. BACKGROUND

A. Inter-Process Communication (IPC)

Sometimes, inter-process communication is also referred to as interthread and inter-application communication [1]. For example, the selection of IPC method should be done with much care, depending upon the type of data transferred as well as the amount of load which the communications are going to carry [1].
It's difficult to imagine the contemporary operating systems functioning without an installation and implementation of IPC mechanisms, especially in multithreading and distributed systems. Processes synchronization, data sharing, and the possibility of executing a number of processes simultaneously are considered as the core co-ordination that IPC provide [4].

B. Role of Operating System (OS) in IPC

There is a guarantee of security while making inter-process communications. OS limits the rights of access and does communication only between the sanctioned processes [2]. The OS allows and controls all the processes of IPC, enabling different processes on a single CPU to share information and work effectively. The OS ensures this so that security between components is maintained [4]. OS takes care of resource management for every domain. This includes memory, threads, and I/O [5].

C. Distributed Systems

A distributed system is a set of networked computers that are not located at the same physical location, and provide the users with the resources that the system keeps. The inter process communication called IPC in distributed systems is achieved by use of various mechanisms and such mechanisms are system dependent [5]. Therefore, it won't be wrong to say that Process Communication (PC) is the heart of any distributed system-that is, such communication is a prerequisite for the operation of these systems [6]. Distributed systems have application in many areas, for example, multi-tier applications in business [6].

©IJRASET: All Rights are Reserved

ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 12 Issue XI Nov 2024- Available at www.ijraset.com

III. MODERN IPC MECHANISMS

One of the major functions that operating systems should have is process management. In addition to providing process scheduling and resource allocation, the OS must provide mechanisms that allow processes to interact with one another. Such mechanisms are called Inter-Process Communication (IPC) mechanisms [3]. They are as follows:

A. Shared Memory

Communication between processes using shared memory involves sharing certain variables, but how this is done depends entirely on how the programmer sets it up. Final implementation is in the programmer's hand. [5]. Shared memory doesn't automatically take care of any issues that might arise when multiple programs try to access it at the same time. The programs themselves need to manage how they coordinate access to avoid conflicts or errors. To achieve this goal, it often exploits concurrency control techniques such as semaphore [6]. Since different programs can access the same area of memory, shared memory allows them to exchange data directly without needing to send it back and forth. Due to this communication becomes faster and more efficient One major advantage of shared memory is that it's especially useful when programs need to share a large amount of data efficiently [7].

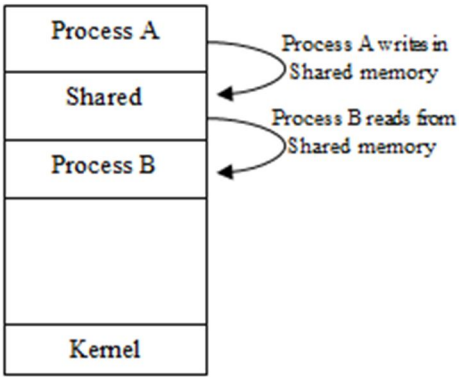


Fig.1. Shared Memory between two Processes

B. Pipes

A pipe is considered the most basic form of IPC and is sharable by more than two processes whether relevant or independent. These are of two types:

1) Ordinary Pipe

Something highly known to a programmer of a shell script is a pipeline: it is merely a number of processes connected one to another through the standard input/output, where the output of one process might be the input of another [1]. They implement the producer-consumer mechanism, that is, one process writes to a pipe and another reads from it [6]. If at any given time some processes are terminated due to whatever reason, an ordinary pipe would be cleaned out or destroyed [6].

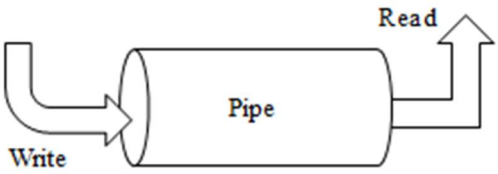


Fig.2. Ordinary Pipe

2) Named Pipe

On the named pipes we can look as some sort of add-on to the traditional type of pipes. Named pipes are bidirectional, unlike the ordinary pipe are unidirectional [1]. Once the named pipes are developed, several processes can communicate over it. They are not demolished even after terminating the process [6]. So we need to take care about their deletion after they are not needed.

©IJRASET: All Rights are Reserved

ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 12 Issue XI Nov 2024- Available at www.ijraset.com

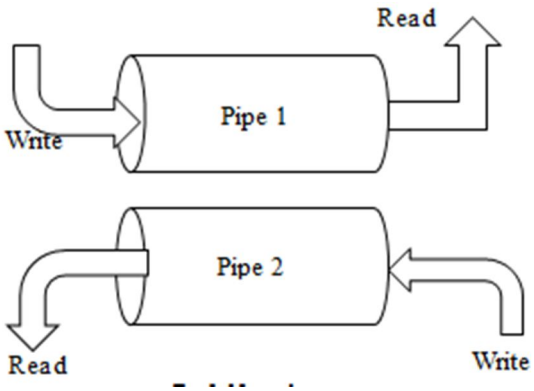


Fig.3. Named pipe

C. Remote Procedure Calls (RPCs)

A remote procedure call (RPC) is a network programming model or inter-process communication technique that is used for point-to-point communications between software applications [5]. It is mostly seen in various Distributed Systems. Even though the basic idea behind RPC (Remote Procedure Call) is simple and easy to understand, implementing it can be tricky. There are many subtle challenges that can arise, leading to differences in how it's actually done. As a result, there are a lot of different RPC implementations available, both in the research and industrial environment [8].

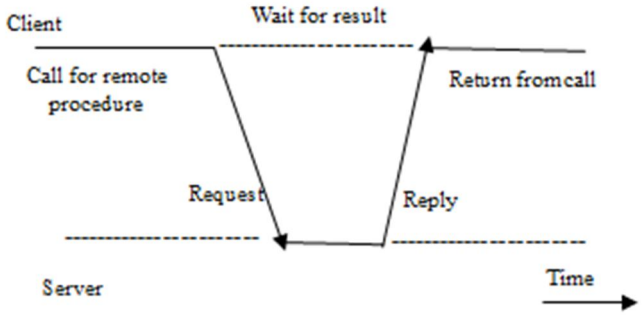


Fig.4. Principle of RPC between a client and server process

RPC works in the request response method where client sends request to the remote server, like calling a function or method. The system translates this request and sends it over, so the server can process it. When the remote server receives the request, it sends a response back to the client and the application continues its process [9].

When the server is processing the request, the client pauses and waits for the server to finish before it continues with its own tasks. In general, RPC applications use software modules called proxies and stubs, which make them look like local procedure calls (LPC) [9].

D. Sockets

A socket is an endpoint that is identified by an IP address and a port number, allowing communication between any two independent processes on the same or different machines [6]. When a programmer specifies a socket he or she would inform the operating systems to allocate the necessary resources and space in order to make a connection without dealing in TPC/IP principles [6].

A method of communication that works in both ways. For example, there is a protocol TCP/IP that governs how packets are sent over the socket. A socket is said to have an address that uniquely consists of the IP address of the machine and port number [11].

©IJRASET: All Rights are Reserved

IV. COMPARATIVE EVALUTION OF IPC MECHANISMS

IPC Methods	Speed	Ease of Use	Concurr-ency	Commun-icaton Scope	Best use case
Shared Memory	Very High	Com-plex	Manual Synchroni-zation needed	Same M/c	High speed local data sharing
Pipes	Mode-rate	Simple	Bidirectional , no manual handling	Same M/c	Simple parent child process communication
RPCs	Mode-rate	Mode-rate	Managed by framework	Distributed Systems	Distributed client server systems
Sockets	Mode-rate to High	Comp-lex	Managed by the developer	Local and networked	Network Communication (e.g Web services)

V. CASE STUDIES

A. *Mach System Points*

Brief Description: Mach achieves location independence for intra-process communication (IPC) by passing messages between ports located anywhere in the address space. Contents in a message have a type, but communication ports are the means by which objects in Mach are addressed. This communication is rights-managed, meaning at most one task can hold receive rights, while several tasks can have send rights [5].

IPC Features: In system ports serve as secured queues, and messages are comprised of a header and some typed data objects. More so, Mach Network Message Server (Net Msg Server) gives naming of a location and routing of messages from one computer to another transparently.

Application Domain: In this system, ports and typed message data create a focus on security and flexibility over geography in distributed systems [6].

B. *MOSIX System*

Overview: MOSIX uses a modified TCP/IP protocol in order to bulge up the inter-process communication aspects without regard to other activities that go on in UNIX distributed systems [5].

IPC Features: Its inclusion with TCP/IP also offers reliable concentrated concerns in communication across areas, an aspect that enriches the processes of augmenting productivity and effectiveness during large process loads.

Application: MOSIX is very applicable for systems that require fast workloads sharing as well as efficient networking to keep many nodes working like one for example UNIX systems [5].

C. *JX Operating System*

Overview: JX is object-oriented Java OS in which different protection domains interact by using IPC. The architecture is of the kind domain-oriented where domain-zero is the most trusted domain built based on the microkernel and this domain has to take care of activities such as management of activities like the handling of CPU registers and system supervision [5].

IPC Mechanism: In JX, inter-domain communication is done through portals that work essentially like Java RMI (Remote Method Invocation). A service in one domain is invoked through a proxy (portal) in another domain. The parameters of the portal call are passed into the target domain [13].

Memory Objects: Large bandwidth data transfers between domains in JX are also done using memory objects. These are chunks of memory available for use, making access to them without copying any actual memory process possible and thus efficiently therefore acting like shared memory across the domains.

©IJRASET: All Rights are Reserved | SJ Impact Factor 7.538 | ISRA Journal Impact Factor 7.894 |

Use Case: For the design to implement file systems, it is apparent that the separation of domains like filesystems and device drivers that co-operate to work will increase the robustness of the system because of the use of IPC mechanisms. Fast portals and method in-lining are used for this performance improvement [13].

VI. CHALLENGES IN IPC MECHANISMS

A. *Starvation*

A process is said to starve if it cannot gain access to the needed resources in order to perform its task. In the priority scheduling system, due to high priority processes, there are high priority processes and all the available resources of low priority processes are being taken up by the high priority processes and hence the low priority processes starve for resources [4].

B. *Deadlocks*

A deadlock is said to be a condition wherein all the processes held are suspended for an indefinite amount of time because of waiting for some event to occur. It occurs when two or more processes get blocked. In this case, each of the progresses waits for the resources or other processes that are held by other processes. The deadlocks are normally very common in the shared memory systems and the semaphore controlled systems that discourage multiple access to the resources where abuse of synchronization is most common [4].

C. *Performance Overhead*

The use of Inter Process Communication (IPC) has a performance penalty since the information has to cross different protection domains and is therefore potentially slower than a regular call to a method, except of course when some form of hardware memory protection system is in place [13].

It can be noted that even with some degree of enhancement, in the framework of JX, portals remain less efficient than direct method invocation due to the overheads associated with switching threads, parameter passing, and domain separation.

D. *Thread and Resource Management*

One of the challenges posed by using threads and resources across a number of domains is the management of threads and resources. For example, in the case of JX, a fast portal call helps to eliminate unnecessary thread swapping but such optimizations are not always possible and more so in cases where several protection layers are involved [13].

VII. SECURITY CONCERNS

In the setting up of a efficient Inter-Process Communication mechanism, there are many factors that need to be put into consideration and one of these is security. First among these is that the system calls that implement the IPC between processes introduce some level of risk because they are part of the operating system. Because these techniques facilitate humans to connect with one another and even carry out some functions, there is a threat that if the design or use of such processes is flawed, then some crook might hijack these processes and causes interruptions to the operation of the system. Therefore, security is of utmost importance in planning to avoid breaches and threats like data losses and alterations, which may compromise the whole system integrity [1].

Inadequate configurations of settings in the IPC of a system increase the exposure of the system to different kinds of vulnerabilities that provide different angles with which an attacker can take advantage of the system [1]. Another major threat is the security risk related to remote code control. Most of the techniques applied to carry out the main forms of IPC are based on memory mapping principles, which become a security risk if left unprotected. For instance, the OS running their supplied code could be coerced to run by an intruder who has access to a shared memory segment due to an address space overflow attack. An exploit of this nature could hang the entire machine so caution must be exercised when applying techniques in IPC with proper safeguards in place [2].

Other malicious users can exploit the IPC channels by flooding them with demands that degrade the operational efficiency or even deny services completely. This is more problematic in those environments that use message-passing or other shared resource-based approaches to implementing IPC mechanisms [4].

The level of security can be tried to increase by using encryption techniques like AES or XOR in Inter-Process Communication. However, the use of them again brings a trade-off between security and speed. Another advanced technique is PRISON, where the monitoring of the involved processes as well as their communication takes place to reduce damage done by malware by taking advantage of IPC [5].

Dos attacks are possible on socket based communication. This means that other than risks, there are also safety precautions available. The document you uploaded indicates that DDoS IPC systems can be defended by conducting monitoring of message queues and applying special thresholds [6].

VIII.CONCLUSION

Modern methods of Inter-Process Communication (IPC) are essential for improving performance and also for enhancing the work capacity of local as well as distributed systems. There are old forms of communication, such as shared memory and pipes, still useful for performing the local and high-speed data transfers. More complex forms consist of a distributed method of communication, involving Remote Procedure Calls (RPCs) or Unix domain sockets. On the contrary however, the implementation of these methods, especially in the case of a networked environment or multi-threaded application brings additional complicated issues including security. Effective encryption, state control as well as guarding will reduce available windows of attack such unauthorized entry and DDoS attacks. In all instances, whether the focus is on performance or dataless communication, efficiency, IPC adaptation constrains such factors as data load or communication scope, and hence requires an assessment of each strategy's concerning the system in question.

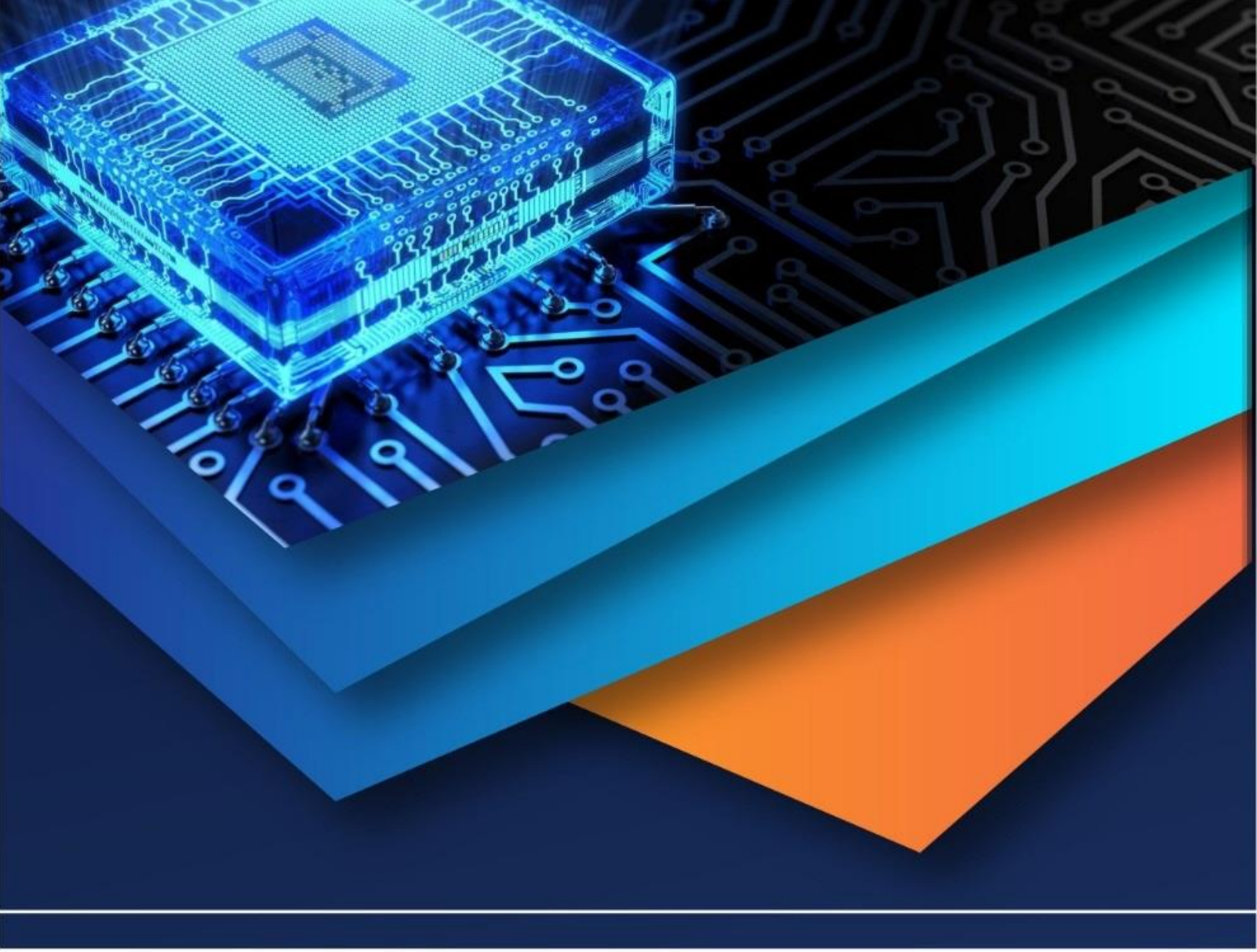
REFERENCES

[1] Zoran Spasov, T-Mobile Macedonia, Skopje, Macedonia, Inter-Process Communication, Analysis ,Guidelines and its impact on Computer secuirity, The 7 th International Conference for Informatics and Information Technology (CIIT 2010)
[2] A.E.M. Eljialy, Sultan Ahmad, A Novel Technique to Secure Inter-Process Communication, IJCSNS International Journal of Computer Science and Network Security, VOL.22 No.9, September 2022
[3] Kwame Wright, Kartik Gopalan, Hui Kang, Perfromnce Analysis of Various Mechanisms for Inter-process Communication
[4] Himali Patel, Vivek Dave, INTERPROCESS COMMUNICATION IN OS, ISSN: 2456-6683 INTERNATIONAL JOURNAL OF RESEARCH CULTURE SOCIETY, Volume - 4, Issue - 4, Apr – 2020
[5] Dr. Shamsudeen.E, A Study on Inter process Communications in Distributed Computer systems IOSR Journal of Engineering (IOSRJEN) www.iosrjen.org ISSN (e): 2250-3021, ISSN (p): 2278-8719 Vol. 08, Issue 4 (April. 2018).
[6] Hamed Dinari, Inter-Process Communication (IPC) in Distributed Environments: An Investigation and Performance Analysis of Some Middleware Technologies, I.J. Modern Education and Computer Science, 2020, 2, 36-52 Published Online April 2020 in MECS
[7] BRIAN N. BERSHAD, THOMAS E. ANDERSON, EDWARD D. LAZOWSKA, and HENRY M. LEVY, User-Level Interprocess Communication for Shared Memory Multiprocessors, ACM Transactions on Computer Systems, Vol 9, No. 2, May 1991,
[8] B. H. Tay, A. L. Ananda, A Survey of Remote Procedure Call.
[9] ANDREW D. BIRRELL and BRUCE JAY NELSON, mplementing Remote Procedure Calls, ACM Transactions on Computer Systems, Vol. 2, No. 1, February 1984.
[10] Ms. S.Krishnaveni and Ms. D.Ruby, Comparing and Evaluating the Performance of Inter Process Communication Models in Linux Environment, Special Issue Published in International Journal of Trend in Research and Development (IJTRD), ISSN: 2394-9333.
[11] Aditya Venkataraman, Kishore Kumar Jagadeesha, Evaluation of Inter-Process Communication Mechanisms.
[12] FRANK D. ANGER, On Lamport's Interprocessor Communication Model, ACM Transactions on Programming Languages and Systems, Vol. 11, No. 3, July 1989.
[13] Christian Wawersich, Meik Felser, Michael Golm, Jürgen Kleinöder, The Role of IPC in the Component-Based Operating System JX.

©IJRASET: All Rights are Reserved | SJ Impact Factor 7.538 | ISRA Journal Impact Factor 7.894 |

ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 12 Issue XI Nov 2024- Available at www.ijraset.com

©IJRASET: All Rights are Reserved | SJ Impact Factor 7.538 | ISRA Journal Impact Factor 7.894 |



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Call : 08813907089 (24*7 Support on Whatsapp)