International Journal of Research (IJR) Vol-1, Issue-9, October 2014 **ISSN 2348-6848**

Debugging- A Higher Approach

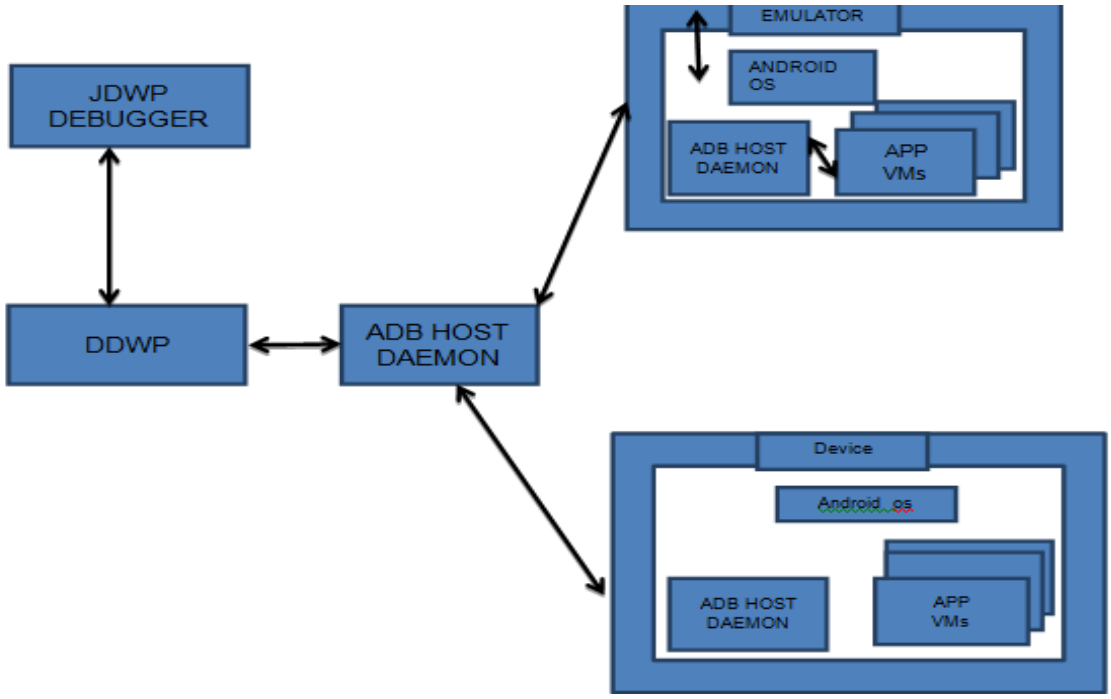
Anuj Kumar Bishnoi & Priyanka Khatri
Information technology, Dronacharya College of Engineering, India
Anuj199229@gmail.commsg2khatri@gmail.com

Abstract:-
The paper presented over here is a complete attenuation of the techniques used in the rapid development of our computer precisely. Here, the technique used is debugging. Debugging is a most common process used in computer application for the removal of bugs and other extracurricular activities from the rapid environment. Debugging is basically a methodical process in which numerous of bugs are find out and removed from the computer device or any electronic based device. It is a widely used in all the devices for the fulfillment of various application devices so that there is no other abstraction in the working of the process. It clears the defects in the electronic hardware thus making them behave as expected.Debuggingalso tends to become harder when various subsystems are being tightly coupled, as changes in one also may cause the one bug to emerge in another.
There are numerous bookswhich have been written about the debugging , as it mainly involves the numerous aspects which includes interactivedebugging, control flow, integration testing, log files, monitoring ,application, system, memory dumps, profiling, Statistical Process Control, and also design special tactics to improve the detection while simplifying the changes.
In contrast, the main general purpose computer software a designed environment, a primary characteristics of the embedded environments there are the numbers of different platforms available on the developers (CPU architectures, vendors, operating systems and their variants). Embedded systems are mainly by definition the not general-purpose designs: they are typically developed for the single task (or small range of tasks), and the platform is also chosen specifically on order to optimize that substituted the application. Not only it does makes the life tough for the embedded system developers, it also
Makesthe debugging and the testing of these systems harder as well, since the different number of debugging tools are needed in different platforms.
It identifies and fixes the bugs in the system that is the logical or synchronization problems in the code and also designs the error in the hardware
It also collects the information about the various operating states of the system that it may be used to analyze the system
It also findthe ways to boost itsperformance or to optimize other important characteristics i.e., energy consumption, reliability, real-time response.

Introduction:-
The term "debugging" has a vast sign to get originated. It consists of terms "bug" and "debugging" they both is popularly attributed to the AdmiralGrace Hopper in the 1940s. While she was working on a Mark II Computer at Harvard University, her associates discovered a moth stuck in the relay and thereby impeding operation, whereupon she remarked that they were "debugging" the system. The term "bug" in the
meaning of the technical error dates back at least to 1878 and Thomas Edison , and "debugging" seems tohave been used as a term in aeronautics before entering the world of computers. Mostly, in an

interview Grace Hopper remarked that she wasnot coining the term. The month fit the already existing terminology, so it was saved. A letter from J. Robert Oppenheimer used the term in a letter to Dr. Ernest Lawrence at UC Berkeley, dated
October 27, 1944, [4] regarding the recruitment of additional technical staff.
In 1945, Oxford English Dictionary made the entry for "debug" quoting the term "debugging" which used in reference to airplane engine testing article in
The Journal of the Royal Aeronautical Society. An article mentioned in the "Air force" (June 1945 p. 50) refers to debugging in the time of aircraft cameras.
The debugging process has a Vidal scope as the software and electronic systems generally have become generally more complex. Thus, the various types of common debugging techniques have expanded with the more methods to detect the anomalies, assess impact, and schedule software patches and the full updates to a system. The word used "anomaly" and "Discrepancy" is initiated as being more neutral terms needed to avoid the words "error" and "defect" or "bug" where there might be an implication so that all so-called errors, defectsor bugsmust are fixed at all costs. Either, an impacted assessment can be made in order to determine the changes toremove discrepancy which would be cost-effective for the system; instead a scheduled new release might mayrender the change
Unnecessary. There are not mean that all issues are life-critical orthe mission-critical in a system. It is also important to avoid the situation in which a change might be make more disastrous effect to the users, long-term, than living with the known problem in this case the "cure would be worse than the disease"). Basing the decisions regarding the acceptability of some anomalies which can avoid a culture of a "zero-defects", where people might be tempted to exempt theexistence of problems in order so that the result would appear as zero defects.

Diagram of Debugging Environment:-
USB



- ADB host daemon: - It acts as a middle one between the device and a development system. It provides various types of device managing capabilities which include moving and syncing the files to the emulator which acts as an advanced shell on the device and provides a general purpose one.
- Dalvik the Debug Monitor Server: - DDMS is basically a graphical program that communicates with the devices through adb. DDMS can also capture the screenshots,

gathers thread and the stack information with spoofing incoming calls and sms messages, and has other features too.

- Android Virtual Device: - The application must run in an android virtual device so that it can be debugged. An adb device daemon runs on theemulator and also provides the means for the adb host daemon in order to communicate with the emulator.
- JDWP debugger:-Dalvik Virtual Machine supports all the JDWP protocol to allow debuggers to attach to a VM. Eachapplication runs in a virtual machine andalso expose a unique port that can attach a debugger with the help of DDMS. In order todebug multiple applications together by attaching to each port might become tedious, so DDMS provides a port forwardingfeature that can forward a specific VM's debugging port to port 8700. A switch freely from application toapplication by highlighting it in the Devices tab of DDMS. DDMS forwards the appropriate port to port 8700. Most of the modern Java IDEs include a JDWP debugger one can use a command line debugger such as jdb.

Technical Aspect:-

- Print debugging: - it is the act of watching the live or recorded traced statements, or print statements which indicates the flow ofexecution of a process. This is sometimes also called as print debugging, due to use of the print function in C. This kind of debugging came in to existence by the command TRON in the original versions of the novice-oriented BASIC programming language. TRON stood for,

"Trace On." TRON caused the line numbers of each BASIC command line to print as the program ran.

- Remote debugging:-It is the process of debugging in which the program running on a system is different from the debugger. To start remote debugging, a debuggermust connect to a remote system over a network. The debugger can then control the execution of the program on the remotesystem and can retrieve the information about its state of functioning.
- Post-mortem debugging: - It is debugging of the program after it has already crashed. Related techniques often include various tracingtechniques and analysis of memory dump of the crashed process. The dump of the process mainly beobtained automatically by the system .whenthe process has been terminated due to an unhandled exception or by a programmerinsertedinstruction, or manually by the interactive user.
- Delta Debugging –It is a debugging technique of automating test case simplification.

Anti debugging:-

Anti-debugging is the implementation of one or more techniques within computer code that hinders attempts at reverse engineering ordebugging a target process. It is actively used by recognized publishers in copy-

protection schemas, but is also used by malware to

Complicate its detection and elimination.Techniques used in anti-debugging includes:

- API-based: check for the existence of a debugger using system information
- Exception-based: check to see if exceptions are interfered with.
- Process and thread blocks: check whether process and thread blocks have been manipulated.
- Modified code: check for code modifications made by a debugger handling software breakpoints.
- Hardware- and register-based: check for hardware breakpoints and CPU registers
- Timing and latency: check the time taken for the execution of instructions.
- Detecting and penalizing debugger.

Conclusion:-

Debugging therefore is a most important technique used now days in the computers and all types of hardware devices in order to protect them from being erupted with the various malware and bugs. Thus, debugging requires advanced types of debugger which makes it possible to enhance the process of debugging.

References:-

1. Grace Hopper (http://foldoc.org/?Grace+Hopper) from FOLDOC
2. https://en.wikipedia.org/wiki/J._Robert_Oppenheimer
3. https://en.wikipedia.org/wiki/Ernest_Lawrence
4. http://bancroft.berkeley.edu/Exhibits/physics/images/bigscience25.jpg
5 S. Gill, The Diagnosis of Mistakes in Programmers on the EDSAC (http://links.jstor.org/sici?sici=0080-4630%2819510522%29206%3A1087%3C538%3ATDOMIP%3E2.0.CO%3B2-9), Proceedings of the Royal Society of London. Series A, Mathematical and Physical Sciences, Vol. 206, No. 1087 (May 22, 1951), pp. 538-554
6. Robert V. D. Campbell, Evolution of automatic computation (http://portal.acm.org/citation.cfm?id=609784.609786), Proceedings of the 1952 ACM National meeting (Pittsburgh), p 29-32, 1952.
7. Alex Orden, Solution of (http://portal.acm.org/citation.cfm?id=609784.609793) systems of linear inequalities on a digital computer, Proceedings of the 1952 ACM national meeting (Pittsburgh), p. 91-95, 1952.
8. Howard B. Demuth, John B. Jackson, Edmund Klein, N. Metropolis, Walter Orvedahl, James H. Richardson, MANIAC (http://portal.acm.org/citation.cfm?id=800259.808982), Proceedings of the 1952 ACM national meeting (Toronto), p. 13-16
9. The Compatible Time-Sharing System (http://www.bitsavers.org/pdf/mit/ctss/CTSS_ProgrammersGuide.pdf), M.I.T. Press, 1963
10. Peggy Aldrich Kidwell, Stalking the Elusive Computer Bug (http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?tp=&arnumber=728224&isnumber=15706), IEEE Annals of the History of Computing, 1998.
11. Postmortem Debugging, Stephen Wormuller, Dr. Dobbs Journal, 2006 (http://www.drdobbs.com/tools/185300443)
12. E. J. Gauss (1982). *"Pracniques: The "Wolf Fence" Algorithm for Debugging",*.
13. Andreas Zeller: *Why Programs Fail: A Guide to Systematic Debugging*, Morgan Kaufmann, 2005. ISBN 1-55860-866-4