Distributed Co-ordinator Model for Optimal Utilization of Software and Piracy Prevention

Vineet Kumar Sharma vineet_sharma@kiet.edu
Associate Professor, Department
of Computer Science & Engg.
Krishna Institute of Engineering & Technology
Ghaziabad, 201206, U.P., India

Dr. S.A.M. Rizvi samsam_rizvi@yahoo.com
Associate Professor, Department
of Computer Science
Jamia Millia Islamia, central university
New Delhi, 110025, India

Dr. S.Zeeshan Hussain szhussain@rediffmail.com
Asst Professor, Department
of Computer Science
Jamia Millia Islamia, central university
New Delhi, 110025, India

Abstract

Today the software technologies have evolved it to the extent that now a customer can have free and open source software available in the market. But with this evolution the menace of software piracy has also evolved. Unlike other things a customer purchases, the software applications and fonts bought don't belong to the specified user. Instead, the customer becomes a licensed user — means the customer purchases the right to use the software on a single computer, and can't put copies on other machines or pass that software along to colleagues. Software piracy is the illegal distribution and/or reproduction of software applications for business or personal use. Whether software piracy is deliberate or not, it is still illegal and punishable by law. The major reasons of piracy include the high cost of software and the rigid licensing structure which is becoming even less popular due to inefficient software utilization. Various software companies are inclined towards the research of techniques to handle this problem of piracy. Many defense mechanisms have been devised till date but the hobbyists or the black market leaders (so called “software pirates”) have always found a way out of it. This paper identifies the types of piracies and licensing mechanisms along with the flaws in the existing defense mechanisms and examines social and technical challenges associated with handling software piracy prevention. The goal of this paper is to design, implement and empirically evaluate a comprehensive framework for software piracy prevention and optimal utilization of the software.

Keywords: End User License Agreements, User Datagram Protocol, Distributed software and License key management.

1. INTRODUCTION

Most retail programs are licensed for use at just one computer site or for use by only one user at any time. By buying the software, the customer becomes a licensed user rather than an owner. Customers are allowed to make copies of the software for backup purposes, but it is against the law to give copies to friends and colleagues. Software piracy is all but impossible to stop, although software companies are launching more and more lawsuits against major infractions'. Originally, software companies tried to stop software piracy by copy-protecting their software. This strategy failed, however, because it was inconvenient for users and was not 100 percent foolproof. Most software now requires some sort of registration, which may discourage would-be pirates, but doesn't really stop software piracy.

An entirely different approach to software piracy, called shareware, acknowledges the futility of trying to stop people from copying software and instead relies on people's honesty. Shareware publishers encourage users to give copies of programs to friends and colleagues but ask everyone who uses a program regularly to pay a registration fee to the program's author directly. Commercial programs that are made available to the public illegally are often called "warez".

Software piracy now cost in the range $15-20 annually to the companies. The main objective is to prevent intellectual property of individuals and organizations. Software piracy prevention is important because the legal actions are lengthy and it is difficult to change the moral standards of the people.

It is a matter of history that with the introduction of IBM PC in the early 1980's a revolution began. Some famous software applications like "word star","lotus123","dBase" were used with IBM PC. Hardware was relatively more expensive than software, and often inclusions of the latest version of these software packages with new system were very common and routinely expected. Upgrades to the software packages were available usually directly from the developers, or as often was the case through resellers, but always incurred additional fees. Tragically, this also led to underground trade on these applications. Ironically, the more popular the application was, the greater its appeal in the so called "black market" and its traders, the "software pirates".

Among the various approaches that have been explored recently to counteract the problem of software piracy, some are of legal, ethical and technical means. Legal means are based on the fear of consequences of violating piracy law. But while most software piracy cases legal means are available, prosecution on a case by case basis is economically unviable. Furthermore, it is conceived as bad publicity and can take a long time. Ethical measures relate to making software piracy morally unappealing. While the intentions are laudable, it takes even more time to change the moral standards of a larger group of people. The technical means include the static measures of defense which incorporates in itself the protection mechanism that is built into the distributed database. Once the system is broken then the static protection techniques are not satisfactory at all.

The concept of software license[8] was developed by the software industry since its early inspection. Software license is that the software publisher grants a license to use one or more copies of software, but that ownership of those copies remains with the software publisher. One consequence of this feature of proprietary software licenses is that virtually all rights regarding the software are reserved by the software publisher. Only a very limited set of well-defined rights are conceded to the end-user. Most licenses were limited to operating systems and development tools. Enforcement of licenses was relatively trivial and painless. Any software customer had certain rights and expectations from the software and developer. Some software was licensed only to one user or one machine, while some software may have been licensed to a site specifying the maximum number of machines or concurrent instances of the program in execution (processes). These are also known as "End User License Agreements" (EULAs). The terms of each EULA may vary but the general purpose is the same – establish the terms of contract between software developer and user of software product.

2. TYPES OF SOFTWARE LICENSES

1) Individual licenses (Single-user) This license type allows the software to be used on only one computer which is not accessed by other users over a network. The other users are not allowed to use software while connected to your computer. Types of individual licenses are:[10]
a) Perpetual license: allows the customer to install and use the software indefinitely without any limitation.
b) Subscription license: Allows the user to use the software for a specific time period. At the end of the term the user has several options: (1) renew the subscription; or (2) purchase a perpetual license at a discounted cost; or (3) remove the software from the computer.
c) Trial license: Allows software vendors to release the software for a period time like a month as marketing tool. The end user can only use this software for only one month.
d) Evaluation license: This license allows the expensive software to be evaluated with certain period.
e) Demo license: This license allows end users to demonstrate the software with partial function or certain period.
f) Feature based license: This license allows end users to use partial function of the software to save money than to use the whole package.
g)Time limited rental license: This allows the end user to pay per time. The end user pre-pay a period of time that fits their needs. The end user can also renew before the license expired.

2) Network /Multiuser licenses:
a) Server (Network): Licensed per server – This license type requires that you have a single copy of the software residing on the file server.
b) Per Seat (Machine): Licensed per machine/seat – This license requires that you purchase a license for each client computer and/or device where access to services is needed. This license is typically used in conjunction with a network license.

3) Add-on's to existing or new licenses:
a) Upgrade: This license is acquired when a user has a previously acquired software license and would like to move up to a newer version.
b) Student use: This allows students to use the software as long as they are students of the institutions. Students are required to uninstall software upon leaving the University.
c) Secondary use: Allows the licensed end user to use the software on a second computer.
d) Work-at-home rights: Allows Faculty/Staff to use software at home. This is effective for as long as the primary work computer is licensed and as long as the person is an employee. Termination of employment also terminates this benefit.

2.1 How piracy is done?
The pirates can be called as cryptanalysts who decrypt the patches from certain software CD's and the license key available with that CD. Each CD of the software has a unique license key available with it. The pirates take up few CD's along with their license key and find out the patch which is consistent with that particular CD. Then they make copies of such CD's along with the same patch and sell it in the market illegally.

2.2 Types of piracy
If you're computer savvy, you're probably aware that copying and distributing copyrighted software is illegal, but you may be surprised to know that there are actually many ways you can unintentionally pirate software. It seems that illegal software is available anywhere, to anyone, at any time. From shopping malls, to the unscrupulous computer systems retailers a few blocks down the street, pirated programs are sold for a pittance. Becoming familiar with the different types of software piracy can help protect you from purchasing pirated software [7]. Many computer users have found themselves caught in the piracy trap, unaware they were doing anything illegal. To avoid such unpleasant surprises, it may be helpful to know the basic ways one can intentionally or unintentionally pirate software:

1) Cracks and serials: Cracks and serials are forms of software piracy that consists of legally obtaining an evaluation version and subsequently entering a copied license code or applying a generic patch that undergoes a copy protection. This is a widespread form of piracy. It is so popular because of small amount of information that needs to be exchanged illegally distribute and obtain a license code or a patch than a complete program.

2) Softlifting: Softlifting occurs when a person (or organization) purchases a single licensed copy of a software program and installs it onto several computers, in violation of the terms of the license agreement. Typical examples of softlifting include, "sharing" software with friends and co-workers and installing software on home/laptop computers if not allowed to do so by the license. In the corporate environment, softlifting is the most prevalent type of software piracy - and perhaps, the easiest to catch.

3) Unrestricted Client Access: Unrestricted client access piracy occurs when a copy of a software program is copied onto an organization's servers and the organization's network "clients" are allowed to freely access the software in violation of the terms of the license agreement. This is a violation when the organization has a "single instance" license that permits installation of the software onto a single computer, rather than a client-server license that allows concurrent server-based network access to the software. A violation also occurs when the organization has a client-server license but is not enforcing the user restrictions outlined in the license agreement. This occurs, for instance, when the license places a restriction on the number of concurrent users that are allowed to access the software but the organization is not enforcing that number. Unrestricted client access piracy is similar to softlifting, in that it results in more employees having access to a particular program than is permitted under the license for that software. Unlike softlifting though, unrestricted client access piracy occurs when the software is installed onto a company's server - not on individual machines - and clients are permitted to access the server-based software application through the organization's network.

4) Hard-Disk Loading: Hard-disk loading occurs when an individual or company sells computers preloaded with illegal copies of software. Often this is done by the vendor as a means of encouraging the consumer to buy certain hardware. If you buy or rent computers with preloaded software, your purchase documentation and contract with the vendor must specify which software is preloaded and that these are legal, licensed copies. If it does not and the vendor is unwilling to supply you with the proper documentation, do not deal with that vendor.

5) OEM Piracy/Unbundling: OEM (original equipment manufacturer) software is software that is only legally sold with specified hardware. When these programs are copied and sold separately from the hardware, this is a violation of the distribution contract between the vendor and the software publisher. Similarly, the term "unbundling" refers to the act of selling software separately that is legally sold only when bundled with another package. Software programs that are marked "not for resale" are often bundled applications.

6) Unauthorized Use of Academic Software: Many software companies sell academic versions of their software to public schools, universities and other educational institutions. The price of this software (and sometimes the functionality) is often greatly reduced by the publisher in recognition of the educational nature of the institutions. Using academic software in violation of the software license is a form of software piracy. Acquiring and using academic software hurts not only the software publisher, but also the institution that was the intended recipient of the software.

7) Counterfeiting: Counterfeiting is the duplication and sale of unauthorized copies of software in such a manner as to try to pass off the illegal copy as if it were a legitimate copy produced or authorized by the software publisher. Much of the software offered for bargain sale at computer trade shows and on auction and classified ads sites is counterfeit software. The price and source are often indicators as to whether software is counterfeit. For example, if a particular piece of software normally retails for $1399 but is being sold or auctioned for $199 then red flags should go up. Likewise, if a particular software vendor only sells its products though certain authorized channels, then a purchaser would be wise not to purchase the software at a trade show.

8) CD-R Piracy: CD-R piracy is the illegal copying of software using CD-R recording technology. This form of piracy occurs when a person obtains a copy of a software program and makes a copy or copies and re-distributes them to friends or for re-sale. Although there is some overlap between CD-R piracy and counterfeiting, with CD-R piracy there may be no attempt to try to pass off the illegal copy as a legitimate copy - it may have hand-written labels and no documentation at all. With technological advancements

making it relatively easy and inexpensive to copy software onto a CD-R, this form of piracy has escalated. As with counterfeit CDs, CD-R piracy is rampant on auction and classified ad sites.

9) Download Piracy: Download piracy is the uploading of software onto an Internet site for anyone. Anyone who uploads or downloads the software is making an illegal copy and is therefore guilty of software piracy. Examples of this include the offering of software through a website, P2P network or share hosting site. Incidences of Internet piracy have risen exponentially over the last few years.

10) Manufacturing Plant Sale of Overruns and 'Scraps': Software publishers routinely authorize CD manufacturing plants to produce copies of their software onto CD-ROM so that they can distribute these CD-ROMs to their authorized vendors for resale to the public. Plant piracy occurs when the plant produces more copies of the software than it was authorized to make, and then resells these unauthorized overruns. Piracy also occurs when the plant is ordered by the publisher to destroy any CDs not distributed to its vendors, but the plant, in violation of these orders, resells those CDs that were intended to be scrapped. While most plants have compliance procedures in place, there have been several instances of this type of piracy.

11) Renting: Renting software for temporary use, like you want to watch a movie, was made illegal in the United States by the Software Rental Amendments Act of 1990 and in Canada by a 1993 amendment to the Copyright Act. As a result, rental of software is rare.

The types of piracy identified above are not mutually exclusive. There is often overlap between one type of piracy and another. OEM software is unbundled by the pirate in order to be re-sold. Not only does the pirate sell the OEM software, but he also makes numerous illegal copies of the OEM software and sells them as counterfeits.

3. OPTIMAL USE OF SOFTWARE

For optimal use of the software a model in an organization is used which tries to keep the information about the specified software on a single machine (considered as co-ordinator) and the complete management of the dynamic distribution of that software and its license is to be done on the same machine. The selection of the co-ordinator is done arbitrary or by executing the election algorithms. If in any case the co-ordinator goes down than any other machine is voluntary elected as the co-ordinator to provide uninterrupted functioning for dynamic or electronic distribution of the software license. Here the software and license key management is done dynamically by the co-ordinator machine. The co-ordinator machine is responsible to make an account for all those machines which are executing the software. In this methodology the organization cannot use the software on the number of computers, exceeding the number of license purchased but this methodology provides an ethical way for optimal uses of the software on the network of an organization. Therefore it prevents organizational piracy and supports optimal use of software on the network of an organization[4], for e.g. there are 500 users on a network and software is used by at most 300 users at a time then it is better to take 300 licenses and use it with the prevention of piracy. In this scheme a machine known as co-ordinator is dedicated for dynamic software and license management.

3.1 Distributed Co-ordinator Model: The approach of “single coordinator model” can be enhanced in terms of efficiency and performance. In single coordinator model there is only one coordinator which manages the distribution of software license keys among the client machines when there is the requirement of execution of the software by that client. Now in the present approach the concept of “Super Coordinator” is used.

Now in this approach of “Super Coordinator” a tree based approach is used. In this-“tree based approach” the root is the super coordinator and its children are the sub coordinators. Various client machines communicate with their corresponding sub coordinators only. Here, sub coordinator takes up the job of providing the license keys to the client machines. The super coordinator performs the crucial task of assigning the particular number of license keys to the sub coordinators as per their requirement. For example, super coordinator has 500 license keys and then it assigns 100 license keys to sub coordinator 1, 150 license keys to sub coordinator 2, 100 license keys to sub coordinator 3, and 80 license keys to sub coordinator 4. The remaining 70 license keys are kept with the super coordinator itself which can be used in

case of need and can be assigned to a particular sub coordinator on demand. In case sub coordinator 3 is finished executing the software and is in no need of more than 40 license keys then the remaining license keys (that is, 60) is returned back to the super coordinator and can be used in future need by any of the sub coordinator.
The operation of this system of approach is quite similar as that of "single coordinator" approach. If a client machine needs a license key then communication between the client and its sub coordinator occurs. If the sub coordinator has a free license key (free license key is the key that is not being used by any machine for the execution of the software) then it is been sent to the client and the client starts its execution of the software. On the contrary, if the sub coordinator does not have any free license key then it requests the super coordinator to provide it with the license key which it could provide to its client. If the super coordinator has a free license key then it is provided to the sub coordinator and then to the client machine, otherwise the sub coordinator along with the client machine goes into waiting queue and waits for the license key.

3.2 Role of sub-coordinator: When a user wants to execute the software application it broadcasts port specific UDP message in its sub network to the sub coordinator (this is the only sub coordinator which is listening request messages on that port). On receiving the request the sub coordinator checks for the available license keys in the data bank, if keys are available then it is delivered to the client and makes its entry in the active client list. On the contrary if the sub coordinator does not have requisite number of keys then it maintains a waiting queue and if it crosses a particular "threshold value" then it sends request message to the super coordinator. Now there arises two cases: Case:1- If the super coordinator has available number of keys then it grants those keys to the sub-coordinator. Case:2-If the super coordinator possess lesser keys or does not posses any available key messages are transmitted to each sub-coordinators and then the sub-coordinators surrender their unused keys to the super coordinator.

3.3 Fault tolerance On peaceful termination of the client the keys available with it are surrendered back to its sub-coordinator along with the deletion of its entry from the active client list available with other clients. In case of abnormal termination of the client there is a procedure of dual check, in this process an "Is-Alive" message is repetitively sent to each client by the sub coordinators, if a response is received from the client then it means that the client is functioning. On the other hand if there is no response from the client then the sub coordinator checks it for second time by sending the message again. If this request is also without the response then the client is supposed to be "dead" and it results in surrendering of its keys to the sub coordinator and deletion of its entry from the active client list of every alive clients. Now there arises a case of sub-coordinator crash, in such cases there occurs a voluntary selection of a client as the sub-coordinator and then this information is passed to all the available clients by updating the active client list of every client.

3.4 Algorithm for key distribution
A hierarchical approach is used consisting of three entities at different levels Super-Coordinator (level 0), Sub-Coordinator (level 1) and the clients (level 2).

1. Clients, who want to execute the software, first of all broadcast the *Search UDP Port Specific Message* in its own subnetwork. This message packet is captured by the sub-coordinator working in the same subnetwork. Sub-coordinator sends the response packet back to the same client. Because UDP packet contains the ip address of both sender and receiver, the client gets the ip address of the sub-coordinator.
2. Now the same client unicasts the *Request UDP Port Specific Message* to its sub-coordinator requesting for a license key for execution of the software.
3. The sub-coordinator receives the request message and check out the availability of the license keys.

3.1. If license keys are available with the sub-coordinator then it is provided to the requesting client and its entry is being made in the current active list of the sub-coordinator.
3.2. If the sub-coordinator does not have any available key then it ask the client whether it is ready to wait or want to quit.

3.2.1. If the client is ready to wait to get a license key then the sub-coordinator makes its entry in the waiting client list. If the length of the waiting client list exceeds the threshold limit (T_H), the sub-coordinator sends a Request Message to the super-coordinator demanding the T_H license keys.

3.2.1.1. The super-coordinator checks the availability of the license keys.

3.2.1.2. If the super-coordinator possess more than or equal to T_H license keys then super-coordinator transmit T_H keys to the sub-coordinator.
3.2.1.3. Else the super-coordinator unicasts a Surrender Back UDP Message to each sub-coordinator except the one for which super-coordinator is intending to provide license keys, requesting to surrender the unused license keys
3.2.1.4. Each sub-coordinator receives this message and returns back some of the unused license keys to the super-coordinator.
3.2.1.5. Now the availability of the license keys is checked by the super-coordinator and if it is again lesser than T_H then step 3.2.1.3 is repeated until available keys becomes greater than or equal to the T_H or available keys do not increase at all.
3.2.1.6. These keys are transmitted to the sub-coordinator.
3.2.1.7. Sub-coordinator servers these keys to the clients of the waiting client list based on the FIFO principal.
3.2.2. Else the client quits.

3.5 Termination Algorithm

3.5.1 Peaceful Termination of a client

1. On the peaceful termination of its own software application the client sends a message to sub-coordinator.
2. Sub-coordinator deletes the entry of the specific client form its active client list and make one more license key available.

3.5.2 Abnormal termination of a client

1. Sub-coordinator sends a regular Is-Alive UDP Message to the clients which are in the active client list of the sub-coordinator.
2. When clients receive these messages they respond back to the sub-coordinator indicating their active state.
3. If sub-coordinator does not receive the response message from any specific client it performs the dual check by repeating the step 2.
   3.1. If this time again the same client does not respond back to the sub-coordinator, the sub-coordinator considers it as an abnormal termination of that client and deletes its entry from the current active client list and increases the available license key by one.

3.5.3 Abnormal termination of a sub-coordinator

1. Sub-coordinator periodically transmits the active& waiting client list, available unused license keys and the ip address of the super-coordinator to all the active clients. By this way all the active clients keep all the latest information which the sub-coordinator is bearing.
2. In case of the abnormal termination of the sub-coordinator, the active clients will not receive the Is-Alive messages. On the expiration of the timer which works at the client machines, the client can get an idea about the abnormal termination of the sub-coordinator.
3. The clients getting the information about the absence of the sub-coordinator will wait for a random amount of time and then they transmit the message packets to all other active clients indicating itself as the new sub-coordinator. Because the active client whose timer is expired, have to wait for a random time the contention of election of multiple coordinator is very less. But still if two or more clients take part in process of becoming the new coordinator, they settle this issue by the voting technique.
4. When a new coordinator is elected, it sends a *New Sub-coordinator UDP Message* to all the current and waiting clients. The new sub-coordinator also updates the super-coordinator about itself.

4. EVALUATION AND PERFORMANCE

This approach is better than the single coordinator approach in terms of efficiency. The efficiency of such systems is measured in terms of number of messages that are passed between the communicating entities (that is, client machine and the coordinator). More the number of messages transmitted between the communicators, less is the efficiency and vice versa. Therefore, efficiency is inversely proportional to the number of messages transmitted between the communicators.

In this approach the number of messages to be communicated is decreased to a great extent. As the technique used is a tree based approach, a client demanding the license key broadcasts the request messages in its own subnetwork only. This request packet is captured by the sub coordinator working in the same subnetwork. In case of a single coordinator the broadcast packets transmitted by a client are destined to all the machines of the entire network and this is how number of messages is reduced in the hierarchical coordinator environment. Thus the overhead of messages on a single coordinator is distributed on many sub coordinators. Similarly, the messages received and sent by super coordinator are reduced as no client machine communicates directly with the super coordinator but it communicates with the sub coordinator which in turn communicates with the super coordinator.

The major aspect of efficiency of this system is based on its fault tolerance mechanism. This approach is very viable in terms of failure of the coordinator. If a sub coordinator fails then the client machines which falls under its scope becomes functionless while other client machines under other sub coordinators work without any disturbance. This is how the complete system is prevented against absolute failure. On the other hand if the super coordinator fails then the sub coordinators are still functioning along with their respective client machines which are under their scope.

5. TIGHTER SECURITY BY COMBINING H/W AND S/W TOKENS

This invention relates to security mechanisms which prevent unauthorized use of the software, and in particular to mechanisms which prevent the unauthorized use of software on more than one computer. Various security mechanisms have been devised for preventing the use of software without authorization of the software supplier. These have included hardware security devices, which must be attached to a computer before the software can run on the computer. Typically, the software that is to run includes an inquiry which looks for an indication that the hardware device has been installed. Such hardware security devices assure that the software will only execute on one computer at any one time. These hardware devices[6] can, however, be relatively expensive and moreover need to be adaptable to various types of computers in which they are to be attached.

One of the best example to quote for this tight security provided by hardware and software together is seen in HSBC banks. The HSBC banks provide its users with software security of login and password and along with it, the bank also provides with a hardware key which produces a random number each time it is turned on. The user is supposed to use this identification number generated by the hardware key along with the login and password in order to access the account. The numbers generated by the key follow some particular sequence according to some algorithm which is been checked by the bank and then it allows the user to use his account.

5.1 How hardware based software license key works?

In order to improve the relationships between vendors and customers as well as grow revenue, software pricing and licensing policies are made more flexible. Hardware keys are made to feature secure software licensing options and offers multiple licensing modes locked into the hardware key to supply flexible licensing protection. Such as HASP from Aladdin, Sentinel from Safenet, UniKeyfromSecutech. Hardware key or software protection dongle is the hardware-based protection and licensing management tool. It is a USB key with memory that protects software against piracy and illegal use by allowing access and execution of the protected software only when the key is connected to the computer. The hardware keys [6] provide licensing method with envelope-base automatic implementation and API-based automatic implementation.

The only thing that would make hardware key better for software license management is if there were an industry-standard way to store multiple software licenses on one physical dongle (and transfer them securely if need be). Public computers could have those software installed on them at no cost to the business providing the computer, but they would only work when someone presented a valid license via their dongle. And the end user would never have to think about moving the licenses to a new computer or losing valuable software when they dispose of the old one. Other security mechanisms include software devices that look for an identification of the computer in which the software is to be installed rather than the

presence of any hardware device [12]. Such software devices often require complex algorithms to generate a unique association of the program to be run with the identification of the host computer. These software devices may still require that other hardware devices be connected to the host computer in order to establish the unique association of the program to be run with the target computer. It is an object of the invention to provide a software security mechanism which authorizes run time execution of certain software only after generating an association of the software to be run with a single computer in a manner that does not require extremely complex algorithms or the attachment of any additional devices to the target computer.

6. CONCLUSION

Now a day the businesses of software companies are dependent on flexible software applications for changing market environments but the rigid licensing structures for software distribution, as used with most legacy systems, is becoming hurdle in it. The paper presents the types of piracy and how the piracy is done. Besides this it provides a new and ethical technique for the optimal utilization of the software resources of an organization in its own network environment by achieving a better degree of prevention of software piracy. Its strength is the dynamic distribution of software license keys to the clients with the help of hierarchical coordinators. The chance of piracy is eliminated up-to a remarkable position because no static measures to prevent piracy are used. In terms of efficiency the technique of hierarchical coordinator is better than the single coordinator technique.

7. REFERENCES

[1] C. Collberg and C. Thomborson. Software watermarking: Models and dynamic embeddings. In *Principles of Programming Languages*, pages 311–324, 1999.

[2] Mukesh Singhal & Niranjan G. Shivratri. Voting and Election Algorithms. In Advanced concept in operating Systems pages 209 & 343, 2002

[3] George Coulouris, Jean Dollimore & Tim Kindberg. Election Algorithm, Bully Algo & Ring based algo. In Distributed Systems page 445-448, 2006

[4] Leili Noorian & Mark Perry. Autonomic Software License Management System: an implementation of licensing patterns. 2009 Fifth International Conference on Autonomic and Autonomous Systems. IEEE 978-0-7695-3584-5/09

[5] Mathias Dalheimer and Franz-Josef Pfreundt. License Management for Grid and Cloud Computing Environments. 9th IEEE/ACM International Symposium on Cluster Computing and the Grid. IEEE 978-0-7695-3622-4/09

[6] Mikhail J. Atallah, Jiangtao Li. Enhanced Smart-card based License Management. IEEE International Conference on E-Commerce (CEC'03)0-7695-1969-5/03 2003 IEEE

[7] Yawei Zhang, Lei Jin, Xiaojun Ye Dongqing Chen.Software Piracy Prevention: Splitting on Client. 2008 International Conference on Security Technology IEEE 978-0-7695-3486-2/08

[8]Daniel Ferrante. Software Licensing Models:What's Out There? 1520-9202/06/ © 2006 IEEE

[9] Dinesh R. Bettadapur.Software Licensing Models in the EDA Industry 0-7803-4425-1/98/$10.00 1998 IEEE

[10] Sathiamoorthy Manoharan and Jesse Wu. Software Licensing: A Classification and Case Study. Proceedings of the First International Conference on the Digital Society (ICDS'07) 0-7695-2760-4/07 $20.00 © 2007 IEEE

[11] Zhengxiong Hou, Xingshe Zhou, Yunlan Wang Software License Management Optimization in the Campus Computational Grid Environment. Third International Conference on Semantics, Knowledge and Grid 0-7695-3007-9/07 © 2007 IEEE

[12] Petar Djekic & Claudia Loebbecke Software Piracy Prevention through Digital Rights Management Systems Proceedings of the Seventh IEEE International Conference on E-Commerce Technology (CEC'05) 1530-1354/05 $20.00 © 2005 IEEE