academicJournals

Vol.5(2), pp. 9-16, October, 2014
DOI: 10.5897/IJCER2013.0202
Article Number: 195D6A948207
ISSN 2141-6494
Copyright © 2014
Author(s) retain the copyright of this article
http://www.academicjournals.org/IJCER

International Journal of Computer Engineering Research

Full Length Research Paper

Real-time operating system (RTOS) with application to play models

Hiba Shahid[1]*, Wadee Alhalabi[2] and John Reif[3]

[1]Department of Electrical and Computer Engineering, Effat University, Jeddah, Saudi Arabia.
[2]Faculty of Computing and Information Technology (FCIT), King Abdulaziz University (KAU), Jeddah, Kingdom of Saudi Arabia.
[3]Department of Computer Science, Duke University, Durham, NC 27707 USA and Adjunct Faculty of Computing and Information Technology (FCIT), King Abdulaziz University (KAU), Jeddah, Kingdom of Saudi Arabia.

Received 18 October, 2013; Accepted 7 October, 2014

It is very important to improve the design of the real-time operating system (RTOS) especially if we want to use it in some special devices. Numerous researches have accepted conventional RTOS as being the customary approach for designing devices used by children. This is because these are able to facilitate the implementation different criteria such as clustering, stability and alternate programs. In this paper, numerous publications have been analyzed to observe the performance of the RTOS when it is subjected to varied constraints. The study focuses on a review of RTOS in relation to play models to analyze their capabilities on various computing platforms and OSs. The publications which we have collected have been sorted out to comprehensively review thereby leading to the configuration of several factors affecting the features within the system. Likewise, statistics and results have facilitated adoption of a more focused approach towards the development of RTOS. While this program ranks clustering and performance as being the highest RTOS criteria for all applications, alternate programs considered this to be the least important. Thus, criteria choice becomes an important issue to address.

Key words: Operating system (OS), play model, real-time, real-time operating system (RTOS) performance, RTOS criteria.

INTRODUCTION

In 1984, a book was published on a reviewing operating system (OS) design, and dealing with OS interface, processes and services along with various important topics (Watson, 1983). An overview on the anomalies frequently experienced by OSs along with proposal for self- management strategies at OS level was presented by Momeni et al. (2008). A survey was conducted on by Romman (2009) to establish a reliable OS for multimedia files and applications and compare it with three of its existing counterparts. Nevertheless, a systematic review on real-time operating system (RTOS) with play model for children as being one of its applications has never been published at least to our knowledge. Owing to this, it has become essential to

*Corresponding author. E-mail: hshahid@effat.edu.sa

observe previous and ongoing research results on optimal OS for play models.

The aim of this paper is to investigate the tradeoffs that occur between certain factors that impact the functionality of the OSs. Since the intention underlying this review is to delicately delve into the possibility of finding a suitable OS for children, it would surely pave the way for development of more reliable and sophisticated RTOS.

LITERATURE REVIEW

An OS is the necessary part of every technical arena because it enables the user to access documents and files and also governs the functioning of all other programs within the system. Examples of some well-known OSs are Mac OS, Unix, Microsoft Windows and Linux all of which have been tested and certified on the basis of various factors throughout their areas of functionality. Therefore, maintaining an acute vigilance is a necessity, while deciding on an apposite resource for technical utility and this in turn is based on several factors compiled bearing in mind the existing market conditions. Once the answer to selecting the best criteria has been identified, it would be easier to judge whether the OSs found in children's toys provide inevitable support within as claimed. Numerous techniques for the selection of an OS have been devised and subsequently categorized in general under the areas of hardware, software, interface, security and virtualization.

A RTOS is an OS that serves real-time application requests, with the ability to process data as it is input, without significant buffering delays. Examples of RTOS are OS for scientific instruments, for machinery control, and industrial control systems.

The jitter of a RTOS is the variability in the time required by the OS to accept and process an application request. A RTOS with low jitter is termed a hard RTOS, and a RTOS with high jitter is termed a soft RTOS.

In addition to time to process data and jitter, there are a number of key potential properties of OS which include:

1) **Reliability and Stability**: The likelihood that the OS is not in failure or crash mode.
2) **Scalability:** This is the ability of the OS to improve in performance as further resources are added.
3) **Availability**: This is the likelihood that OS is actively processing requests, and not either in crash-mode or being updated.
4) **Usability**: This is the degree the OS has been demonstrated in the market.
5) **Security**: This is the degree that the OS is not susceptible to external attack.
6) **Portability and clustering**: This is the ability of the OS to migrate and/or distribute is operations among cluster of computers.
7) **User-Interface:** The ability of the OS to interact with the user.
8) **Certification**: Whether the OS has been demonstrated to provide certain properties.

A RTOS generally has high reliability, availability, but often has limited user-interface. Other OS have different combinations of properties.

Studies undertaken by Swift et al. (2004) described the importance of reliability of any system and hence seek to enhance this factor by isolating the system from driver failures. Reliability factor has been taken into account courtesy of numerous incidences of driver-caused crashes within the system where there has been little or no change in existing driver or system code. It is due to this that system reliability has been described as an important but impenetrable area under discussion (Patterson et al., 2002; Segal and Frieder, 1989). Another fact that has also been noted is that whilst the outlay of high-end computing continues to decline, the failure outlay has been rising ever since. These failures include the unnecessary lay-offs on e-commerce server which result in delay of numerous activities that are performed by the work force for help desk overhaul within the working environment. Furthermore, the emerging segment of daily-use technical appliances based on hardware and software augments the need for reliability since efforts are underway to make these appliances as diminutive, user friendly and automotive as possible (Lin and Chang, 2013).

According to Baier et al. (2012), a particularly crucial development has been the construction of scalable OSs so that the existing system is more robust and shows an improvement in performance when new hardware resources have been added. Basically, this calls for improving the resources of computer system in order to accommodate the rising functionality demand, while at the same time economizing the cost. Some of the factors on which these dimensions depend on are size of processor, memory, software and heterogeneity. Software scalability assumes importance as subject of inspection particularly when one node is shared between a system featuring multiple processor connections and a symmetric multiprocessor with single memory location from where availability plays a vital role in terms of system performance.

Availability is a crucial criterion when choosing an OS since it is an important determinant of all ongoing activities including the execution of instructions by the processor. In such a situation, it is important to have an OS that would allow application of software updates and patches without any downtime or loss of service (Baumann and Appavoo, 2005). Even rebooting or restarting could be deferred without losing the ability to apply security fixes or enhance functionality through software updates provided the system is available at all times. The resolve for availability has been strengthened even more due to the fact that computing infrastructures have been targets of unplanned down-time which has even caused the potential overlay of scheduled down-time to increase significantly. For instance, the processing system for Visa transaction goes through a routine update of approximately 20,000 times per year, however, it tolerates down-time of less than 0.5% (Gillen and Kusnetzky, 2002). Quite a few techniques to minimize down-times and increase availability have been devised such as dynamic update (Tushman and Newman, 2004) which could enable the running of software update application without interrupting service which in turn amplifies the usability of the system.

According to Zhu et al. (2001), usability or market proven factor is one of the most reliable criteria for selecting an OS. OSs which have been in market for more than 10 to 15 years have been employed in many safety-critical applications and tested (or used) by customers for a long time. Over the years, users have stumbled upon numerous inaccuracies within the framework which have been rectified by formulating updated versions. These avatars of OSs include pSOSystem, VxWorks and VRTX. According to a survey published by StaCounter covering the time period from January, 2009 to January, 2013 on the market share held by OSs in the United States, the usability of Windows 7 was 44.02% in March 2012 and dramatically increased to over 50% to gain the top spot in January, 2013. This is chiefly attributed to the efficient usage of Windows Vista alongside the effective marketing strategies of Windows 7 (Swift et al., 2007) and system security.

Yang (2001) addressed one of the fundamental subjects of concern namely security of OSs. This is the foremost cause of apprehension amongst end-users since OS is the core software which executes instructions from configured devices, servers, desktop and other parts. Thus, lack of security could result in unwanted attacks or break-in from one application to the next. According to "DOD Trusted Computer System Evaluation Criteria" (1985), of US government, most OSs available for purchase have C2 level of security which requires Discretionary Access Control (DAC) protection that is particularly supportive of and protects environment in which multiple applications are running simultaneously. Numerous endeavours have been made to develop OS model with utmost security. Studies conducted by Spencer et al. (1999) were made available along with HP-LX (Dalton and Choo, 2001) and Trusted Solaris and these might be an indication that the underlying security of OSs is responsible for the overall security of applications.

Other factors dwell upon portability and clustering. Clustering is basically used to distribute the load over a number of machines. If one machine fails, it could be sent for maintenance without interrupting the running of other services. This is basically determined by the number of machines connected together (Bekman and Cholet, 2003). As per Zhu et al. (2001), some of the criteria entail selecting OS based on certification and OSs which have been developed from scratch using a formally defined semantic specification and were subjected to a rigorous method of testing. Equally, difficult is the proposition of acquiring alternate programs and interface support as also of getting a device driver for a non-supported device while working with an OS (Smith, 2000).

METHODOLOGY

Categorizing OS

OSs can be categorized into distinct categories by the clustering OSs by the various properties we have listed above.

The problem of clustering OSs by the various properties can be reduced to the problem of clustering a set of data, consisting of n distinct d dimensional vectors, into m clusters, so that the Euclidian distance of each element of each cluster from the median of each cluster is minimized. There are a number of algorithms for efficiently determining a near-optimal clustering of multidimensional vectors (Shore and Gray, 1982), for example vector quantization (Gray, 1984; Gersho and Gray, 1992). In our case, the number n of OSs is very large, but the intended number of OS clusters is just 6, so our work was manageable.

OSs is partitioned into six categories in order to estimate the number of devices under each OS so that some estimate of efficiency and usability of each O. This categorization was done particularly for play models. Linux OS comprised of publications related to disability that has been well-known since an early age (Huber et al., 2008), RC products for children cars experiment using the Internet (Aoto et al., 2005) and Bluetooth toy car control (Cai et al., 2011) were found as examples of devices using Linux as well. For variants of Windows, toy plane (Tanguay, 2000), musical computer games (Hämäläinen et al., 2004) and controllers for simulated car racing (Togelius and Lucas, 1906-1919) were found. Many robot toys were found for assisting and playing with severely disabled children featured an underlying core of Unix OS (Kronreif, 2005) and hence pertaining to specific disabilities, certain special purpose OSs were built which limited system’s portability.

There were special purpose OSs built specifically for particular applications that included interactive C, the core of certain low cost vehicles for simple reactive behaviours (Capozzo, 1999), Robot C was the OS for a monoball robot based on LEGO Mindstorms which focused on educating children in elementary school (Prieto et al., 2012). Strifeshadow Fantasy OS was used for a multi-player online game (Chan and Chang, 2004). Designing of UAV helicopter also required special purpose OS (Cai et al., 2005) along with a personalized R-Learning system which operated on Robot Software Platform (Ko et al., 2010). Another multiplayer computer game named Amaze used V-System, Distributed OS (Berglund and Cheriton, 1985). Publications related to embedded systems were also reviewed which included LEGO Mindstorms NXT concepts aimed at developing technical skills in students (Sharad, 2007). CELL processor was used in scientific computing on PlayStation 3 (Buttari et al., 2007) and Intel Microcontroller was found at the core of ESoccer Robot Toy developed as an educational play model (Vial et al., 2007).

It was observed that Windows OS was used for several devices as compared to other OSs. This raised the question as to what

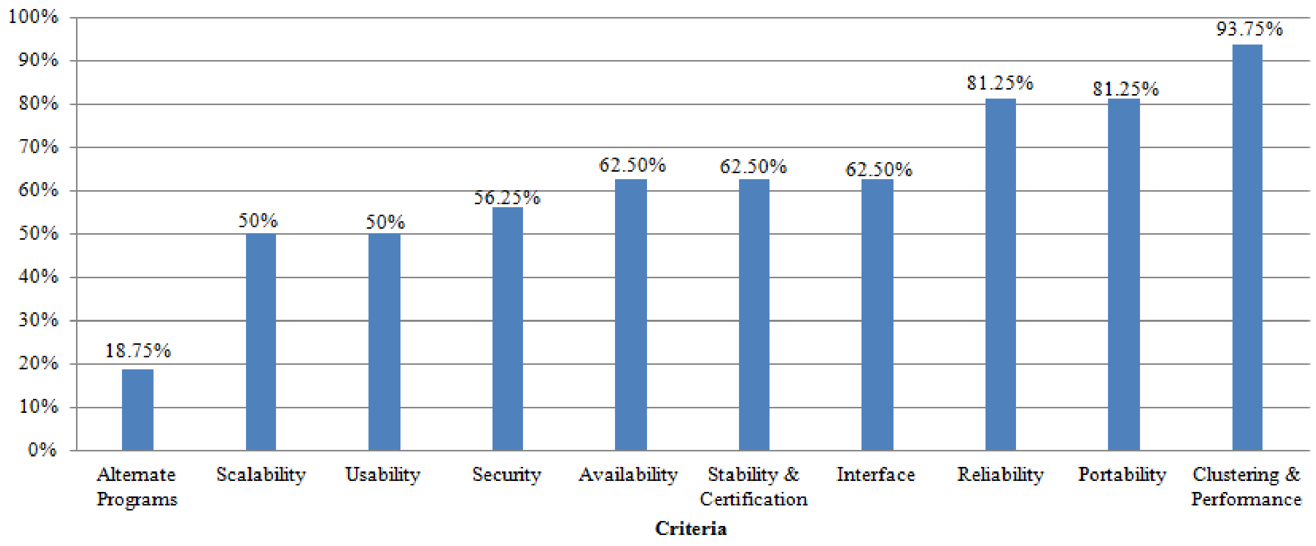

Figure 1. Criteria in RTOS used in play models publication.

could be the reason behind its popularity and hence journal articles known to discuss appropriate criteria for choosing an OS were reviewed in depth.

Some of the most commonly used parameters that were selected as forming the basis on which an OS should be chosen are reliability, scalability, availability, usability, security, portability, clustering and performance, stability and certification, alternate programs and interface. The reason behind the presence of each criteria within publications reviewed were determined and the findings were numerically analyzed. In order to carry out numerical analysis, literatures in agreement to achieving particular criteria were selected. It must be noted that numerical values to show the concurrence were not mentioned as far as the studies related to criteria are concerned and hence the criteria tends to agree as a whole numerically. However, data were given in few literatures which presented the percentage of criteria with respect to percentage of limitations within RTOS. Hence, the given data was summed up with the rest of literatures in agreement to criteria from which the percentage was discerned. Accordingly, the graph depicting the percentage of criteria present in literature related to play model for children reviewed was plotted as shown in Figure 1. It was observed that clustering and performance measure of OS is the highest that is, 93.75% in all the devices except one. Likewise, it was also revealed that the OS run on various devices could not appropriately run applications from other OSs that is, 18.75%.

These findings were compared with RTOS for other applications which included medicine, supercomputing, natural disaster recovery scheme, cloud computing, automobile, manufacturing industry and underwater devices. The objective was to compare the performance of OSs within toys with the performance of the same OS within other applications.

Similar behaviour is observed in Figure 3 which shows the percentage of criteria present in each publication in relation with all applications except play models in which the clustering and performance criteria was highest that is, 96%. However, the criteria pertaining to alternate programs was the lowest that is, 8% which implies the applications built ranging from medicine to underwater devices are mostly specific to certain OS and hence could not operate on different OSs. Hence, it could be sufficiently concluded that the performance of OSs within toys and the performance of the same OS within other applications is almost similar.

DISCUSSION

Outcomes with respect to applications

Play models with respect to criteria and vice versa were analyzed and their values were recorded based on numerical facts provided within publications. These data were plotted in Figure 2 in order to individually observe the performance of each application with respect to OS and vice versa.

Outcomes with respect to criteria

It is highly deemed for the system to be reliable in the context of operating large parallel jobs successfully over long period of time. It basically aims at reducing the mean time between job failures thus affecting reliability and usually aims at improving resiliency of the system thus enhancing the security feature which responded to 95% alongside reliability. Hence, this application displays high response for clustering and performance criteria alongside usability that is, 97%, as shown in Figure 4 which has been improved with the newly proposed high performance computing option which provides accelerated data processing of large orders of magnitude over single-processor systems and few others. Availability and scalability criteria were fairly stable to about 77 and 67% on average, while portability and interface were below average that is, 33% which reflects the size for most RTOS used for this application. The graph took hit the lowest for alternate programs criteria

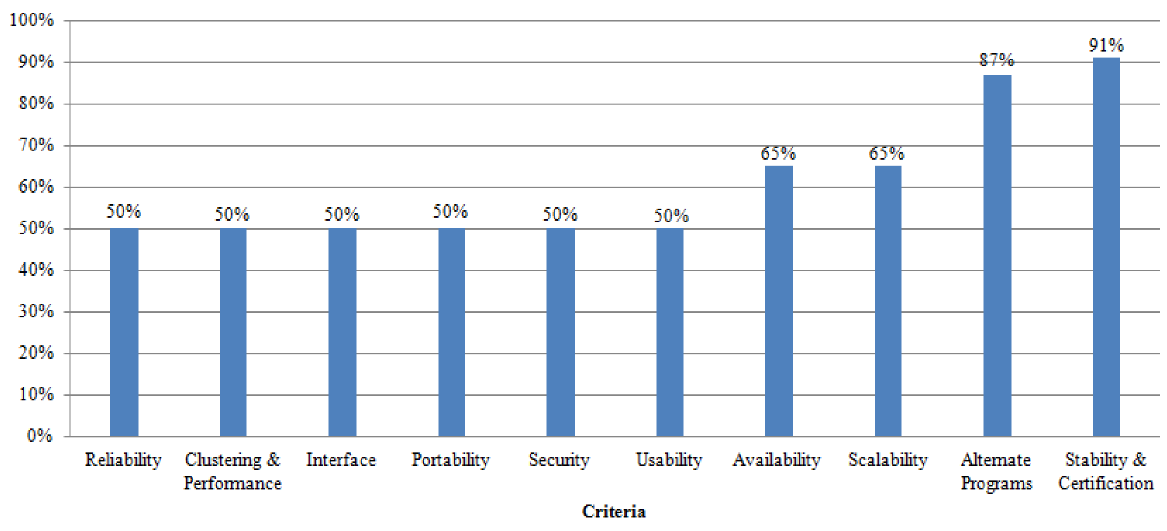

Figure 2. Criteria in RTOS used in manufacturing industry.

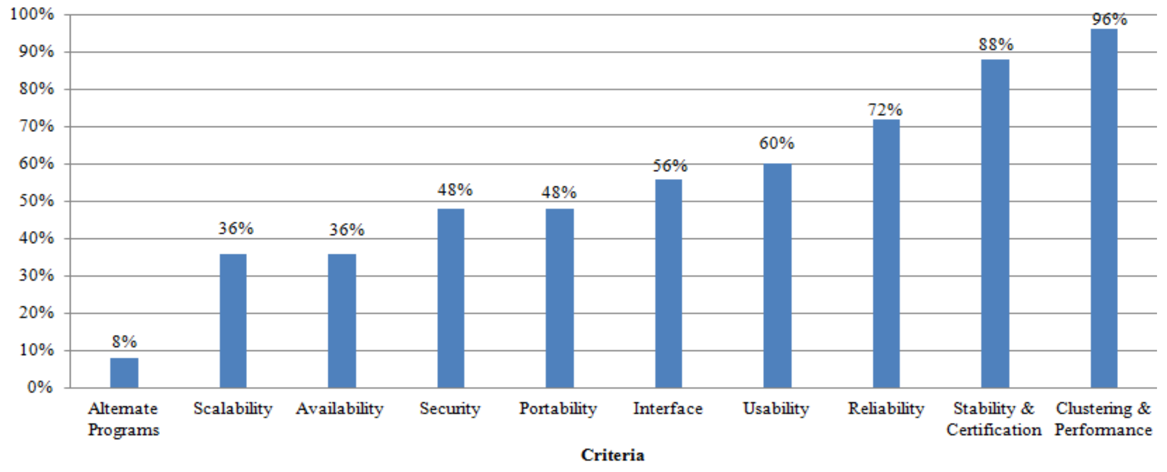

Figure 3. Criteria in RTOS used in other applications.

that is, 13% which implies that the built applications are specific to particular OS and hence could not run on different platform most of the time without making major modifications. On the other hand, manufacturing industry showed relatively stable response of all criteria where the least score was 50% pertaining to reliability, performance, interface, portability, security and usability. Literatures analyzed related to manufacturing industry implied the use of basic OS design over various production lines that is, the load of machine has to be shared by multiple production lines in most frequent cases. Even more specific scenario has been presented by Lin and Chang in case of tile manufacturing system were a similar product has been generated by two different production lines in which the functionality of the machines are the same. However, one generated compact product type with certain machines, while the other produces regular product types with all machines and hence the load of machine is common to both the production types (Lin and Chang, 2013). Without going much into details of performance of machine with respect to each production line, several other literatures have been assessed which shows fairly similar pattern for most of the criteria related to manufacturing industries. Security and usability responded to 65% which was much closer to former criteria, while alternate programs responded to 87%. This pattern implies that the application could easily run from one OS to the next which was expected given a common platform for all the production lines. However, stability and certification peaked up to 91% which was the highest thus reflecting the overall behavioural pattern and compatibility.

Figure 5 displays criteria performance with respect to different applications in which supercomputing and

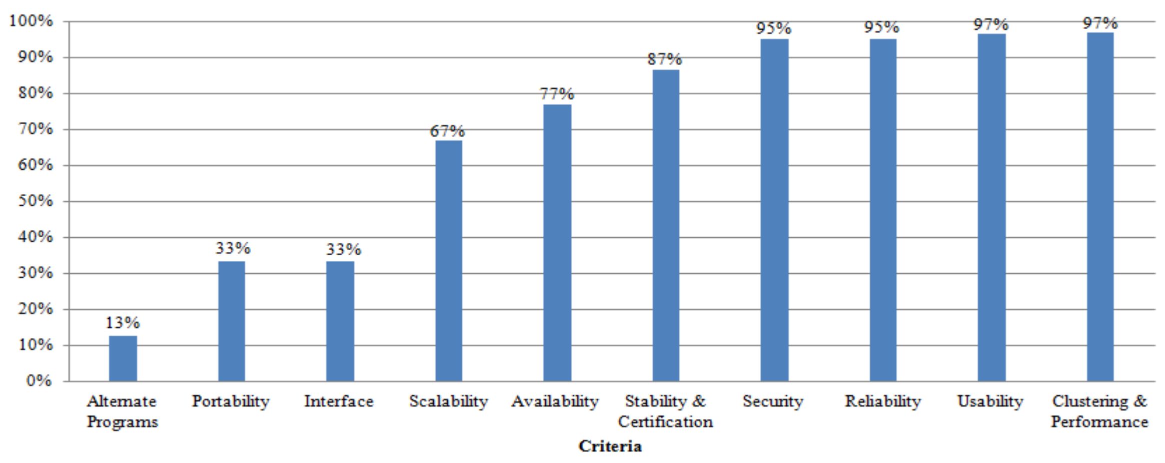

Figure 4. Criteria in RTOS used in supercomputing.

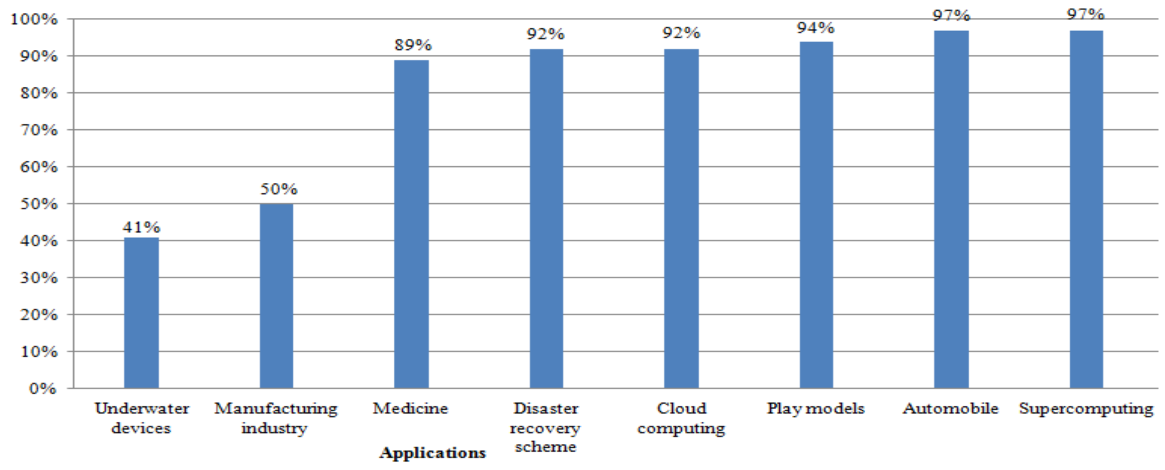

Figure 5. Clustering and performance in different applications.

automobile reaches the highest point that is, 97%. Performance of the system is of critical concern of automotive manufacturers in designing increasingly complex software. Furthermore, the issue of security could be seen in two folds within the design of automobile as well as supercomputers, while reliability and usability are traditional key concerns in the context of mechanical, electrical and software systems (Broy et al., 2007). Play models for children responded to 94% in which the software design for most of the learning systems was separated for the purpose of enhancing the performance and security. For instance, UAV helicopter was based on special purpose OS which was highly clustered into different technical sub areas which in turn reduces the load of all operations on one area thus enhancing the security feature thereby escalating its performance. Disaster recovery schemes and cloud computing displayed fairly stable response of 92% in terms of performance. Quite differently, cloud computing reaches high performance measure by replacing clustering since they are geographically distributed unlike clusters which are tightly coupled connections within small-scale. This is followed by medicine which includes the medical equipment controllers that has responded to 89% on average, while manufacturing industry has reached a response of 50%, underwater devices were quite closer to it that is, 41%. This has been followed by the reason that, in general, the mean time between job failures escalates as the devices goes deeper at lower levels of the ocean. Numerous techniques have been developed to enhance the operation of these devices which has been successful in terms of their performance but this area has not been able to escalate its performance as compared to the rest of applications. Alternate programs were observed to be the least satisfying criteria within applications. As shown

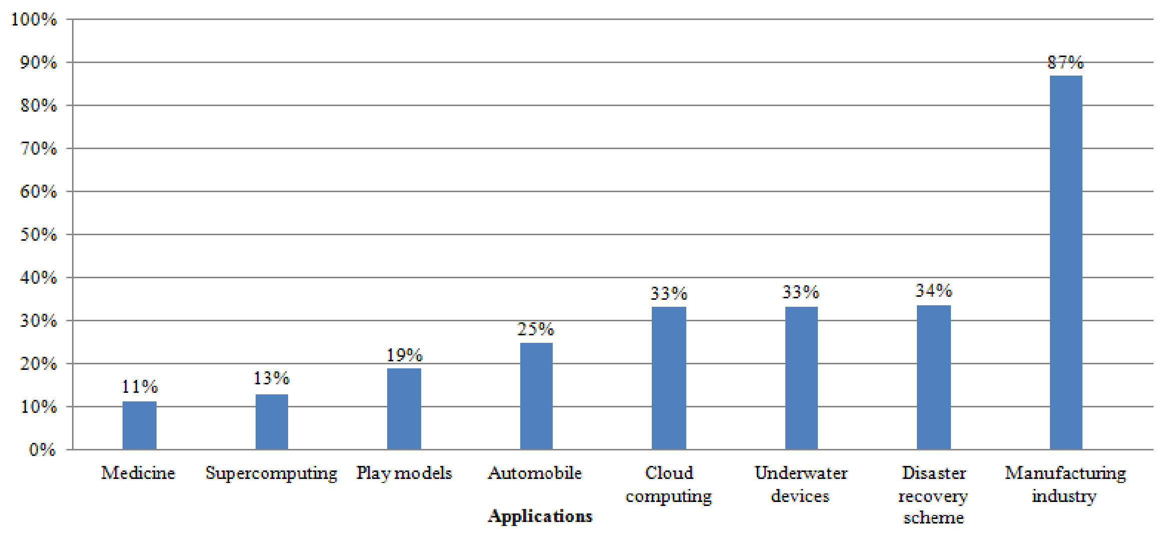

Figure 6. Alternate programs in different application.

in Figure 6, it reached the highest point at 87% for manufacturing industry where software design within products belonging to distinct production lines which includes similar basic design with few modifications according to product requirements. The criteria goes down to 33% for cloud computing and underwater devices since most of time, these applications are specific to the OS alongside disaster recovery scheme with 34% response. Due to the nature of conditions under which these applications are used, the complex designs are built restricted to a single RTOS model. The response for automobile and play models deescalates to 25 and 19%, respectively while supercomputing responded by 13%, medicine marks the lowest point at 11%. The response rate merely reflects the competition between various organizations in development of applications along with the importance laid on the application itself on nature of condition under which it is used. In terms of OS usage, Unix OS was used the maximum especially because it showed greater potential for security and performance. It is also characterized with good load balancing feature which makes it robust against crashes.

PUBLICATION SCREENING AND INCLUSION

Publications were selected on the based on a common point where the abstract and title reflected that the publication deals with OS being applied effectively to play model for children. These publications were sorted out after carefully scanning the OSs used within the information published. This number was condensed to include criteria which were thoroughly reviewed as well as their data were included for analysis. The article sources were taken from IEEE Transactions, Elsevier and ACM Database. Exclusion criteria included patents since they mainly focused on hardware side of the system and representative samples and articles were the ones which showed a participant response rate of 50% or more. Longitudinal study design was followed with the units of analysis being type of OS, applications and criteria.

Conclusions

The paper provided a review of RTOS for use for play models, analyze their capabilities on various computing platforms and OSs. We partitioned OSs into six categories in order to estimate the number of devices under each OS so that some estimate of efficiency and usability of each OS could be formed.

Conflict of Interest

The author(s) have not declared any conflict of interest.

AKNOWLEDGEMENTS

We wish to acknowledge support to both co-authors by King Abdulaziz University (KAU) as well as support to John H Reif NSF CCF-1141847, NSF CCF- 1217457, and NSF CCF-1320360.

REFERENCES

Aoto K, Inoue M, Nagshio T, Kida T (2005) Nonlinear control experiment of RC car using internet. Control Applications. Proceedings of IEEE Conference.

Christel B, Marcus D, Benjamin E, Hermann H, Joachim K, Sascha K, Steffen M, Hendrik T, Marcus V (2012). Chiefly symmetric: Results on the scalability of probabilistic model checking for operating-system code. Systems Software Verification Conference, Sydney, Australia.

Baumann A, Appavoo J (2005). Improving dynamic update for operating systems. In Proceedings of the 20th ACM Symposium on OS Principles, Work-in-Progress Session, Brighton, UK. Bekman S, Cholet E (2003). Practical Mod Perl. Beijing : Sebastopol, CA: O'Reilly.

Berglund EJ, Cheriton DR (1985). Amaze: A Multiplayer Computer Game. 2(3):30-39.

Broy M, Grünbauer J, Hoare T (2007). Software System Reliability and Security, IOS Press.

Cai J, Wu J, Wu M, Huo M (2011). A bluetooth toy car control realization by android equipment. Transportation, Mechanical, and Electrical Engineering (TMEE), International Conference

Capozzo L, Attolico G, Cicirelli C (1999). Building low cost vehicles for simple reactive behaviors. Systems, Man, and Cybernetics. IEEE SMC Conference Proceedings. IEEE International Conference, 6:675-680.

Chan HT, Chang RKC (2004). Strifeshadow Fantasy: a massive multi-player online game. Consumer Communications and Networking Conference. First IEEE, pp. 557-562

DOD 5200.28-STD (1985). DOD Trusted Computer System Evaluation Criteria (Orange Book). http://www.radium.ncsc.mil/tpep/library/rainbow/5200.28-STD.pdf.

Dalton C, Choo TH (2001). An Operating System Approach to Securing E-Services. Communications of the ACM, 44(2):58.

Gillen A, Kusnetzky D, McLaron S (2002). The role of Linux in reducing the cost of enterprise computing. IDC white paper.

Hämäläinen P, Mäki-Patola T, Pulkki V, Airas M (2004). Musical Computer Games Played by Singing. In Proc. of the 7th Int. Conference on Digital Audio Effects (DAFX-04), Naples, Italy, October 5-8.

Huber M, Rabin B, Docan C, Burdea G, Nwosu ME, AbdelBaky M, Golomb MR (2008). PlayStation 3-based tele-rehabilitation for children with hemiplegia, Virtual Rehabilitation.

Ko WH, Lee SM, Nam KT, Shon WH, Ji SH (2010). Design of a personalized r-learning system for children. Intelligent Robots and Systems (IROS), IEEE/RSJ Int. Conf. pp. 3893-3898.

Kronreif G, Prazak B, Mina S, Kornfeld M, Meindl M, Fürst M (2005). Playrob - robot-assisted playing for children with severe physical disabilities. Presented at IEEE, 9th International Conference on Rehabilitation Robotics, Chicago, IL, USA.

Lin YK, Chang PC (2013). Graphical-based reliability evaluation of multiple distinct production lines. J. Syst. Sci. Syst. Eng. 22(1):73-92.

Momeni H, Kashefi O, Sharifi H (2008). How to Realize Self-Healing Operating Systems? Information and Communication Technologies: From Theory to Applications. 3rd International Conference.

Patterson D, Brown A, Broadwell P, Candea G, Chen M, Cutler J, Enriquez P, Fox A, Kıcıman E, Merzbacher M, Oppenheimer D, Sastry N, Tetzlaff W, Traupman J, Treuhaft N (2002). Recovery Oriented Computing (ROC): Motivation, definition, techniques, and case studies. Technical Report CSD-02-1175, UC Berkeley Computer Science.

Prieto SS, Navarro TA, Plaza MG, Polo OR (2012). A Monoball Robot Based on LEGO Mindstorms. Control Systems, IEEE, 32(2):71-83.

Rumman NA (2009). Operating system support for multimedia: Survey. Computer Science and Information Technology-Spring Conference. International Association.

Segal ME, Frieder O (1989). Dynamic Program Updating: A Software Maintenance Technique for Minimizing Software Downtime. J. Software Maintenance, 1(1):59-79.

Sharad S (2007). Introducing Embedded Design Concepts to Freshmen and Sophomore Engineering Students with LEGO MINDSTORMS NXT. Microelectronic Systems Education. IEEE Int. Conf. pp. 119-120.

Smith RW (2000). Linux Hardware Handbook: [selecting, Installing, and Configuring the Right Components for Your Linux System ...]. Indianapolis, IN: Sams.

Spencer R, Smalley S, Loscocco P, Hibler M, Andersen D, Lepreau J (1999). The Flask security architecture: System support for diverse security policies. In Proc. of the 8th Usenix Security Symposium, pp.123-139.

Swift MM, Bershad BN, Levy HM (2003). Improving the Reliability of Commodity Operating Systems, in Proceedings of the 19th ACM Symposium on Operating Systems Principles, Bolton Landing, NY.

Swift MM, Bershad BN, Levy HM (2004). Improving the Reliability of Commodity Operating Systems, ACM Transact. Comp. Syst. 22(4).

Tanguay D (2000). Flying Toy Plane. Computer Vision and Pattern Recognition. Proceedings. IEEE Conference, 2:231-238.

Tushman ML, Newman WH, Romanelli E (2004). Convergence and upheaval: managing the unsteady pace of organizational evolution. Tushman, M.L. and Anderson, P. (Eds.), Managing Strategic Innovation and Change: A Collection of Readings, pp. 530-540. New York: Oxford University Press.

Vial PJ, Serafini G, Raad I (2007). Soccer RoBot Toy within an Educational Environment. The First IEEE International Workshop on Digital Game and Intelligent Toy Enhanced Learning, Jhongli, Taiwan, March 26-28 2007, 215-217. Copyright IEEE 2007

Watson DJ (1983). Book Review: Operating System Concepts. 3:2.

Yang CQ (2001). Operating System Security and Secure Operating Systems. V. 1.4b, Option 1 for GSEC.

Zhu MY, Luo L, Xiong GZ (2001). A Provably Correct Operating System. ACM SIGOPS Operating Systems Review 35(1):17-33. Print.