International Journal of Information Technology and Management
Vol. III, Issue No. I, August – 2012, ISSN 2249-4510

“A Comparative Study on Various Services of Web Based Operating System: A Review”

Sanjeev Ranjan

Research Scholar, Mahatma Gandhi University, Meghalaya

Abstract – WebOS (Web based operating system) is a new form of Operating Systems. You can use your desktop as a virtual desktop on the web, accessible via a browser, with multiple integrated built-in applications that allow the user to easily manage and organize her data from any location. Desktop on web can be named as WEBtop. A new form of computing known as cloud computing can help us to design web based operating systems for future. This paper starts with a introduction of WebOS and its benefits . For this paper , We have reviewed some most interesting WebOS available nowadays and tried to provide a detailed description of their features. We have identified some parameters as a comparison criteria among them.

In this paper, we demonstrate the power of providing a common set of Operating System services to wide-area applications, including mechanisms for naming, persistent storage, remote process execution, resource management, authentication, and security. On a single machine, application developers can rely on the local operating system to provide these abstractions. In the wide area, however, application developers are forced to build these abstractions themselves or to do without. This ad-hoc approach often results in individual programmers implementing non-optimal solutions, wasting both programmer effort and system resources.

To address these problems, we are building a system, WebOS, that provides basic operating systems services needed to build applications that are geographically distributed, highly available, incrementally scalable, and dynamically reconfigurable. Experience with a number of applications developed under WebOS indicates that it simplifies system development and improves resource utilization. In particular, we use WebOS to implement Rent-A-Server to provide dynamic replication of overloaded Web services across the wide area in response to client demands.

Traditional reconfigurable computing platforms are designed to be used by a single user at a time, and are acknowledged to be difficult to design applications for These factors limit the usefulness of such machines in education, where one might want to share such a machine and initially hide some of the technical difficulties so as to explore issues of greater value.

WebOS is a newly emerging Operating System, accessible via a browser, with multiple integrated built-in applications that allow the user to easily manage and organize her/his data from any location. You can use your desktop as a virtual desktop on the web. In the wide area, application developers are forced to build abstractions themselves. This ad-hoc approach wastes programmer effort and system resources. To address these problems, WebOS provides basic operating systems services needed to build applications that are geographically distributed, highly available, incrementally scalable, and dynamically reconfiguring. This paper starts with an introduction of Webs and its benefits, but it also have some flaws related to security.

---♦---

INTRODUCTION

The web operating system is evolving as a form at a rapid pace, promising to free us from Windows once and for all. If you want to take the desktop to your web browser, one contender is well on the way to making it possible.

Web Operating Systems (WebOS) is: “A software platform that interacts with the user through a web browser and does not depend on any particular local operating system.”Web operating systems are also commonly referred to as Web desktops or WEBTOPS.

"A web desktop or webtop is a network application system for integrating web applications into a web based work space. It is a virtual desktop on the web, running in a web browser as software. Web desktops often are characterized by an environment similar to that of Windows, Mac, or Linux, but are now considered to have much more functionality being dependent on the internet. Typical benefits include the ability to save work and settings over the internet rather than to the local desktop."

The first occurrence of the term "WebOS" is in the name of a computer research project started by University of California, Berkeley in 1996 (that is now continuing at Duke University), which describes it this way: "WebOS provides basic operating systems services needed to build applications that are geographically distributed, highly available, incrementally scalable, and dynamically reconfiguring."

Now you can think WebOS as a virtual desktop on the web, accessible via a browser, with multiple integrated built-in applications that allow the user to easily manage and organize her data from any location.

While the World Wide Web has made geographically distributed read-only data easy to use, geographically distributed computing resources remain difficult to access. As a result, wide-area applications that require access to remote CPU cycles, memory, or disk must be programmed in an ad-hoc and application-specific manner. For example, many popular services, such as Digital's Alta Vista or Netscape's download page , are geographically replicated to improve bandwidth, reduce latency, and improve availability—no single connection onto the Internet can support tens of millions of users. Today, such replication is manually managed on both the server and the client side—users are forced to poll several essentially equivalent services and system managers must manually distribute updates to replicas. This situation will only get worse; it is currently predicted that the number of Internet users will increase by an order of magnitude to over 100 million in less than 5 years.

To address these problems, we are building WebOS, a framework for supporting geographically distributed, highly available, incrementally scalable, and dynamically reconfiguring applications. WebOS includes mechanisms for global naming , persistent storage, remote process execution, resource management, authentication and security . We use WebOS to demonstrate the synergy of these services in simplifying the development of wide-area distributed applications and in providing more efficient global resource utilization. The WebOS framework enables a new paradigm for Internet services. Instead of being fixed to a single location, services can dynamically push parts of their functionality out onto Internet computing resources, and even all the way to the client.

Dynamically reconfiguring and geographically mobile services provide a number of advantages, including: (i) better end-to-end availability (service-specific extensions running in the client mask Internet or server failures), (ii) better cost-performance (by dynamically moving information closer to clients, network latency, congestion, and cost can all be reduced while maintaining server control), and (iii) better burst behavior (by dynamically recruiting resources to handle spikes in demand). For example, many Internet news services were overwhelmed on the night of the last U.S. presidential election; our framework would enable those services to handle the demand through dynamic repli cation. Recently, there has been a push toward the distribution of active components in the network, through technologies such as Active Networks and Java . The goal of WebOS is to provide a framework to assist application developers in utilizing programmable and active network components.

To demonstrate the utility of WebOS as a substrate for the development of wide-area applications, we motivate and describe our implementation of Rent-A-Server, an application that allows for transparent, automatic, and dynamic replication of HTTP service across the wide area in response to client load. Rent-A-Server also demonstrates the power of exporting common operating system abstractions to wide-area applications; WebOS services simplified both the design and implementation of Rent-A-Server.

With the rapid development of new forms and concepts of networked and mobile computing, it is increasingly clear that operating systems must evolve so that all machines in a given network, even the Internet, can appear to be controlled by the same operating system. As a result, the world-wide interconnected networks, commonly called the Internet or the Web, could potentially be supported and managed by a giant virtual operating system.

For example, initially the World Wide Web was created to allow one to view remote hypertext pages on one's own machine, thereby facilitating collective work among geographically removed collaborators. Soon after, virtual pages, generated on the y using tools such as cgi-bin, allowed the widespread remote execution of programs. More recently, with languages such as Java , it has become possible to download fully executable programs to one's own machine, and then to make them run on that machine. However, there is no general means for taking an arbitrary program and having it run somewhere on the network.

There are several reasons that this last possibility is actually essential. First, with the development of network-centric computing, there will be more and more limited-capacity machines (slower processors, limited memory or storage space, etc.), such as the NC computers , that will be forced to use more powerful computers on the network to effect any non-trivial tasks. Second, an arbitrary program on the network might just be incapable of running on the local machine, simply because it is the wrong platform (hardware, local operating system, running applications, etc.) Implicit in the above discussion is the heterogeneous nature of most networks. The transparent use of such heterogeneous networks of computers has been partially addressed in work on metacomputing, whose objectives are to transform a network into one single computer system . Recent developments in operating systems such as Inferno or JavaOS provide the user ubiquitous access to resources and information. However, the Web or the future global information infrastructure is more than just a metacomputer or a networked system of computers seen as a virtual machine run by a virtual (network) operating system, in that there is no complete catalog of all resources available. Moreover, such a catalog is infeasible, because of the highly dynamic and distributed nature of the Web or the Internet, continually integrating rapidly developing technologies.

As a result, any attempt to design one single operating system offering a fixed set of resource-management functions will have difficulty adapting to technological innovation or to new demands. Therefore, there is, such as proposed in , a need for a Web Operating System (WOS), which would make available, to all sites on a network, the resources available on that network, or at least a reasonable subset thereof, to effect computations for which local resources are missing. These resources could be of many forms, including processor speed, available memory or storage space, available operating systems or applications, and so on. In order to deal with the dynamic changes in the system, the Web Operating System should be a versioned system, in which different versions of the operating system are running simultaneously on the network. Should, for instance, a given version not be capable of dealing with a particular request for a service, then it can pass it on to another version, as is currently done for packet routing.

What distinguishes the Internet from classical distributed systems is the fact that there is no complete catalog of all resources available and central decisions making for resource allocation is not acceptable or even impossible. Rather, the Web Operating System (WOS) should be a versioned system, in which different versions not capable of dealing with a particular request for service, then pass it on to another version, as currently done for packet routing.

Generalized software configuration techniques, based on a demand driven technique called eduction are being developed, that can be used to define versions of a WOS to be built in an incremental manner. Software and hardware (description) repositories or warehouses will provide the necessary components for fulfilling a service requested.

Being able to use the global network for parallel/distributed execution of a program in the framework of a WOS is clearly promising. For this to be realistic, mechanisms are required to distribute the work, collect the results and coordinate the participating processes or agents. In particular, for such mechanisms to be effeective, dynamic load balancing/sharing must be implemented. Therefore, we intend to apply advan in a pre{reserved user space .

WEBOS OVERVIEW

In this section, we provide a brief overview of the major WebOS components; together, they provide the wide-area analogue to local area operating system services, simplifying the use of geographically remote resources. Each of these components is operational in our current prototype.

Global Naming: Many wide-area services are geographically distributed. To provide the best overall system performance, a client application must be able to dynamically locate the server able to deliver the highest quality of service. In WebOS, global naming includes mapping a service name to multiple servers, an algorithm for balancing load among available servers, and maintaining enough state to perform fail-over if a server becomes unavailable. These operations are performed through Smart Clients, which flexibly extend service-specific functionality to the client machine.

Wide-Area File System: To support replication and wide-scale sharing, WebOS provides a cache coherent wide-area file system. WebOS extends to wide-area applications running in a secure HTTP name space the same interface, caching, and performance of existing distributed file systems. In addition, we argue for benefit of integrating the file system with application-controlled efficient wide-area communication.

Security and Authentication: To support applications operating across organizational boundaries, WebOS defines a model of trust providing both security guarantees and an interface for authenticating the identity of principals . A key enabling feature is fine-grained control of capabilities provided to remote processes executing on behalf of principals.

Process Control: In WebOS, executing a process on a remote node should be as simple as the corresponding local operation. The underlying system is responsible for authenticating the identity of the requester and determining if the proper access rights are held. Precautions must be taken to ensure that the process does not violate local system integrity and that it does not consume more resources than allocated to it by local system administrators.

As an explicit design choice, we leverage as much functionality as possible from existing low level services. For example, for compatibility with existing applications, we adopt IP addresses and URL's for the global namespace, TCP to provide reliable communication, and SSL for link level security.

WEBOS APPLICATIONS

This section provides an overview of four applications designed using the WebOS framework. The first two applications have been completed, while the last two are under development.

Internet Chat - Internet chat allows for individuals to enter and leave chat rooms to converse with others present in the same logical room. In our implementation, chat rooms are implemented as WebFS files accessed by Smart Clients. The file system interface is well-matched to chat semantics in a number of ways: (i) file appends and reads abstract away the need to send messages (ii) the chat file provides a persistent log of chat activity, and (iii) access control lists allow for private and secure (throughWebFS encryption) chat rooms. For scalability, we allow multiple WebFS servers to handle client requests for a single file (room). Each WebFS server accumulates updates, and periodically propagates the updates to other servers in the WebFS group, who in turn transmit the updates to local clients. Smart Clients choose the least loaded WebFS server for load balancing and connect to alternative servers on host failure or network partition for fault transparency.

To quantify the benefits available from the WebOS framework,we implemented two versions of chat with identical semantics, both with and without WebOS. The initial implementation consisted of 1200 lines of Java code in the client and 4200 lines of C++ code in the server. By using WebFS to handle message transmission, failure detection, and storage, the size of the chat client code was reduced to 850 lines, while the WebFS interface entirely replaced the 4200 lines of chat server code. The main reason for this savings in complexity was the replacement of separate code for managing communication and persistent storage of chat room contents with a single globally accessible and consistent file. As an added benefit, this common WebFS interface is similarly available for other distributed applications. For example, we are currently implementing a shared distributed whiteboard application using this interface.

Remote Compute Engine- Sites with unique computing resources, such as supercomputer centers, often wish to make their resources available over the Internet. UsingWebOS, we allow remote pro grams to be invoked in the same way as local programs and can allow access to the same files as local programs. WebOS functionality is used to address a number of issues associated with such access: the identity of requesting agents is authenticated, programs are provided secure access to private files on both local and remote systems, and programs run in a restricted virtual machine isolated from other programs to protect the local system from malicious users. At our site, WebOS provides compute access to a research cluster of 100 machines. Resource allocation within the virtual machine allows external users to take advantage of the aggregate computing resources, while ensuring system developers have the requisite priority.

Wide Area Cooperative Cache-We are using WebOS to build a geographically distributed Web cooperative cache to both validate our design and to provide an immediate benefit to the Internet by doing more intelligent caching of Web content. Existing proposals for hierarchical caching of the Web suffer from an inability to dramatically grow the cache size and processing power at each level of the hierarchy . With cooperative caching among peer servers, the aggregate capacity grows dramatically with the distance from the client. Thus, while caches above the first level in existing hierarchical designs have very low hit rates and simply increase the latency to end clients, a cooperative cache is more likely to successfully retrieve a cached copy from a peer. We plan to explore tradeoffs associated with maintaining directories of peer cache contents, hints , or using simple IP multicasts or broadcasts.

WebOS simplifies the implementation of the cooperative cache in a number of ways. First, Smart Clients are used to determine the appropriate proxy cache to contact. WebFS is used to transport cache files among the proxies and to securely share any necessary (private) state among the proxies. Finally, the authentication model allows proxies to validate their identities both to one another and to the client.

InternetWeather-A number of sites are currently attempting to provide regular updates of congestion, latency, and partitions in the Internet. Such information is invaluable for services making placement and load balancing decisions. However, all current efforts take network measurements from a centralized site, making it difficult to measure network characteristics between two arbitrary sites. We are addressing this limitation by using the WebOS framework to generate more comprehensive snapshots of Internet conditions.

In our implementation, a centralized server provides Smart Client applets for those wishing to view the current Internet weather. In exchange for the weather report, the user implicitly agrees to allow the applet to execute traceroute to a subset of server-determined sites and to transmit the result back to the server. Using these results from multiple sites, the service is able to construct fairly comprehensive snapshots of Internet weather.

REVIEW OF 10 WEB OS

Cloudo - Formerly known as Xindesk, Cloudo is an open internet-based operating system that is written in PHP and runs the LAMP software bundle. It makes full use of the area of the browser and seamlessly integrates with the iPhone's mobile browser. Written using open technologies, this browser based OS is high on features and usability. Currently in Public Beta, it opens to consumers next year.

EyeOS - One of the first implementations of the web-based OS that you can run on your own server, EyeOS offers a credible amount of customization options as long as your web server runs PHP5 and Apache. EyeOS also offers GUI customization options which means that you can set up an OS the way you and your users want it – highly recommended for those who need to set up their own Web OS.

G.ho.st - Short for "Global Hosted Operating SysTem", Ghost Inc.'s web-based operating system is built for all those consumers who need to set up an online cloud-computing base without any hassles. The working environment is very similar to most operating systems that users use, and users can add more applications in the near future.

Glide OS - TransMedia's Glide OS is yet another entrant into the competitive world of cloud computing. However, with Intel's plans of putting Glide into their ultra-mobile PCs, this is one online OS that is not to be trifled with. Packing a host of applications in its basic version, Glide does an admirable job of providing easy cloud computing for its users on both the PC and mobile platforms.

Lucid Desktop - Lucid Desktop (formerly known as the Psych Desktop) is built on a base of PHP5 and is a prosumer-oriented web desktop service. This desktop can be installed on to a web server like EyeOS, and is remarkably simple to use and write code for. With its proximity to Linux's code, Lucid Desktop will be a sure hit with the Linux-loving masses.

Online OS - A welcome departure from the masses, Online OS is written in Javascript and uses AJAX for its fast and user-friendly work. The demo desktop looks like the Windows XP desktop (the registered version can be made to look like KDE or Mac OSX) and loads very fast (although there are a few glitches here and there when components fail to load quickly). It has file-management and other abilities, but most of its applications are not part of the OS itself – rather, they launch outside of it, making Online OS a sort of a portal to these apps.

qWikiOffice - This is one of those few online operating systems that make use of the EXT-JS library for its functioning. Coupled with cutting-edge GUI, qWikiOffice is indistinguishable from an ordinary OS when it works. However, it is still under development and there are no releases as yet.

Windows4all - Using Microsoft's Silverlight technology, Windows4all is an online virtual operating system. With a Vista-like GUI and desk bar, this is for all those Windows fanatics who need to use their favourite OS on computers that do not have Windows installed.

DesktopTwo - Labelled a "Webtop", this cloud OS is one of the premier services offered online, developed by Sapotek. The goal of the developers is to convert the internet into a full fledged platform for working instead of relying on hard-drive based applications. It is a playground for developers as they have released their code under AGPL license.

Google Chrome OS - Google's highly anticipated (slated for release in 2010) cloud computing OS is still largely open to speculation as Google has not released much information since its blog post on the Chrome OS and we have reason to believe that it will be as groundbreaking as its Browser.

Technical Review - WebOS provides OS services using wide-area network and applications, there is a need to include mechanisms for resource discovery, a global namespace, remote process execution, resource management, authentication, and security. On a single machine, application developers can rely on the local operating system to provide these abstractions. In the wide area, however, application developers are forced to build these abstractions themselves or to do without. This ad-hoc approach wastes programmer effort and system resources. To address these problems, WebOS provides basic operating systems services needed to build applications that are geographically distributed, highly available, incrementally scalable, and dynamically reconfiguring. An application that demonstrates the utility of WebOS tested using Rent-A-Server application , which is a web server capable of dynamically replicating itself geographically in response to client access patterns.

WORKING OF WEB OS

As the Web evolves, people invent new words to describe its features and applications. Sometimes, a term gains widespread acceptance even if some people believe it's misleading or inaccurate. Such is the case with Web operating systems.

An operating system OS is a special kind of program that organizes and controls computer hardware and software. Operating systems interact directly with computer hardware and serve as a platform for other applications. Whether it's Windows, Linux, UNIX or Mac OS X, your computer depends on its OS to function.

A Web OS is a user interface that allows people to access applications stored completely or in part on the Web. It might mimic the user interface of traditional computer operating systems like Windows, but it doesn't interact directly with the computer's hardware. The user must still have a traditional OS on his or her computer.

While there aren't many computer operating systems to choose from, the same can't be said of Web operating systems. There are dozens of Web operating systems available. Every WOS user should be able to share his or her local resources with other users. In addition, users should be able to combine and use different resources for interactive problem solving. A lot of communication is needed to find these resources. Thus strategies for communication and searching are necessary. Two strategies are available to use:

The broadcast strategy-The requesting machine submits the request to each machine in the list. Each of these machines then sends messages back to the requesting machine. Since these machines can almost work in parallel the answer will be quickly available on the requesting machine. If the list contains n machines, 2n messages will be generated. messages from requesting machine and n answers both positive and negative. Thus the network load is high. Furthermore, the broadcast implementation must be realized, hence delaying data transmission.

The serial request strategy-In this case the requesting machine sends one message containing the list of the remaining machines to one of the machines from the list. If the service is available on this machine, a positive answer is directly sent back to the requesting machine. So the generated network load is much less than in the first case. On the other hand, the respond time is much higher than in the first case and any communication problems or long transfer times directly influence the respond time.

IMPEDIMENTS TO WEB OS

Cloud storage: While a browser-based OS offers plenty of benefits, it's also hampered by severe limitations. Most notably: everything is stored in the cloud. If you're working from the office or your home, that's generally not a concern. However if you travel, accessing a reliable and fast broadband connection can be tricky. Many areas have dead zones, limited coverage and inconsistent throughput rates.

Constant Online Access: Complicating matters further, many wireless ISPs impose a data cap on their mobile broadband service. A computer that requires constant online access to transmit data or stream music and video could hit those caps very quickly. It wouldn't be as troubling if you could work offline, but the majority of apps currently available for Chrome OS won't work without a broadband connection. This makes working while traveling difficult or, in some cases, impossible.

Limited File Management: Other issues include the lack of proper VPN support, limited file management and some weird browser compatibility issues that prevent some websites from loading or functioning correctly. And, while the Chrome App Store offers a wide variety of apps, it's still rather limited. As a result, finding what you need can sometimes prove difficult.

Security : Security is another major computer concern, especially if you are storing some sensitive data. You are usually required to purchase some antivirus software and do regular tests of your drives to make sure there are no infected files. Google OS handles all the security for you so there is no need to waste time on that. The jury is still out on whether this is a pro or a con, though – if all your sensitive data is online and you play no part in keeping it safe.

Hackers : When you have a web-based initiative, you also have to be on the lookout for hackers. Hackers could potentially hack into your servers and change your website. By doing this, the hackers can change the links on your website so that they are paid when sales are generated. They could also sabotage your site to negatively influence customer's buying decisions. Because of these threats, you will need to constantly monitor your website or web interface to ensure that hackers are not involved.

Phishing Attacks : In some cases, phishing attacks can negatively impact your web-based initiatives. If customers need to login to your platform for some reason, it is possible for scam artists to use phishing as a way to get into their accounts. This type of attack involves a scam artist sending an email posing to be your business, in an attempt to get the login information. Once this happens, the scam artists have full access to your customer's accounts.

CONCLUSION

The conclusion is that a web-based operating system will imminent in Web 2.0's age, and we should design new WebOS based on reviewed services that will give users the power of computing on the Web. No more hard-drive backups required – just turn on your browser and get going with these Web OS services after some years.

The general aim of out approach is to develop a family of services for illustrating and studying the concept of a Web Operating System (WOS) based on one single underlying concept, the demand-driven computation using simple warehouses, which hold and provide all the necessary information a system may offer to a request. The ongoing work includes (1) production of sample resource managers and warehouses, together with the necessary automatic broadcast or 'resourcemining' mechanisms, (2) the implementation of a sample series of WOS-services (e.g. typesetting services, graphics processing, interactive simulations, etc.) and (3) implementation of a prototype user-interface based on browser-like forms to specify user (application) requests, which includes new 'data-mining' search engines.

In Future you will work with many WebOS , We should think for the future design and risks involve to develop a next-generation operating system. However, we wonder if the lack of offline capability might hinder its adoption since most of the time PC users take their machine overseas where Internet access may be spotty or expensive. Still, there is about many years to go to take full advantages of Web based operating systems.

REFERENCES

Amazon Inc (2008). Amazon Web Services EC2 site. http://aws.amazon.com/ec2.

F. Reynolds (2006). Evolving an operating system for the Web. IEEE Computer, 29(9): pp. 90-92.

J. Plaice and S.B. Lamine (1997). Intensional Programming II, chapter Eduction: A general model for computing. World Scienti_c, Singapure to appear.

R.S. Cox, J.G. Hansen, S.D. Gribble, and H.M. Levy (2006). "A Safety- Oriented Platform for Web Applications. In Proc. IEEE Symposium on Security and Privacy.

S.B. Lamine, J. Plaice, and P. Kropf. Problems of computing on the WEB (2007). In A. Tentner, editor, High performance computing, pp. 296-301, Atlanta, Ga., 1997. SCS.

WebOS (July 1998): Operating System Services For Wide Area Applications," Amin Vahdat, Thomas Anderson, Michael Dahlin, David Culler, Eshwar Belani, Paul Eastham, and Chad Yoshikawa. The Seventh IEEE Symposium on High Performance Distributed Computing

Luotonen and K. Atlis (April 2004). "World-Wide Web Proxies. In First International Conference on the World-Wide Web.

Grimshaw, A. Nguyen-Tuong, and W. Wulf (March 2005). "Campus- Wide Computing: Results Using Legion at the University of Virginia. Technical Report CS-95-19, University of Virginia.

P. Sarkar and J. Hartman (October 2006). Efficient cooperative caching using hints. In Operating Systems Design and Implementation, pp. 35–46.

A.-M. Kermarrec, I. Kuz, M. van Steen and A. S. Tanenbaum (May 2008). "A Framework for Consistent, Replicated Web Objects. In Proceedings of the 18th International Conference on Distributed Computing Systems.