Extending the Power of Server Based Computing

H.L. Yu, W.M. Zhen, and M.M. Shen

High-Performance-Computing Group, Computer Science Department,
Tsinghua University, 100084, Beijing, China
hlyu@tsinghua.edu.cn
http://hpc.cs.tsinghua.edu.cn

Abstract. In this paper, we review the evolution of computing, especially the development of thin client computing. Then, our study is focused on one of the most popular thin-client computing mode: server based computing. After an analysis of the problems existing in server based computing, a revised server based computing model is given, which combines the advantages of other thin-client model. As proof of our concept, our ideas are implemented in detail on our embedded platform: THUMP-NC. Some open source software, including rdesktop and VNC, are used in our implementations.

1 Introduction

With the development of computer technology, the speed of computer network is improving dramatically, network based applications have been developing very fast too. Thin-client computing is originated in this background. 'Thin client' concept, which most processing will be done by server and the client mostly take charge of displaying screen image and sending input events to server, has been changing all the ways too. Thin-client system has all the features on many aspect of a embedded system indeed.

Our study is focused on one of the most popular thin-client computing architecture: server based computing. Firstly, we will look back on the development of computing paradigms. We will see how thin-client computing originated and developed from it. Then, we will give a new paradigm for server based computing, which is more functional and easy-used. Finally, an implementation of this paradigm will be introduced.

2 Evolution of Computing

The evolution of computing has great impact on the way that server based computing does. If you want to change it, you must firstly know how it is originated.

Computing technology has evolved considerably over time. Although the process of evolution has been continuous, it can be classified as three distinct computing paradigms[1][2].

The first computing paradigm, the mainframe paradigm, was popular through the 1960s and 70s. In this paradigm, computer processing power was provided by mainframe computers located in air-conditioned computer rooms. All resources were centralized, the main way to access the mainframe is through some character-based terminals; and the speed of the links between terminals to the mainframe is quite slow. This paradigm is greatly suffered from the costs of maintaining and administrating the system, and the poor price-to-performance ratio too.

The second computing paradigm is involved with powerful PCs and workstations. These standalone computers were single-user systems that executed a wide range of applications; they have their own computation and storage units, so they can work independently. With the advent of fast networking technologies, PC-LAN became more popular, which connect the PCs and workstations together. Client-server computing originated from this; a server in the LAN can provide centralized services to other PCs or workstations. Generally, client machines store and run applications locally while the server provides some file and print services. The pioneers of thin-client computing, such as InfoPad[3] system and Java NC system[4], are some kinds of client-server computing indeed.

Network-centric computing developed in early 1990s. In this paradigm, users' machines access both applications and data on networked servers. The most influential form in network-centric computing is Internet computing, which changed our application-development and content-delivery modes. The Internet has shifted the distributed computing paradigm from the traditional closely coupled form to a loosely coupled one[2]. With the development of Internet Computing, more and more applications are moved from client-server mode to web-based browser-server mode, which needs no client software installations in client machines. This is another kind of thin-client system.

However, commercial requirements and realities have revealed that professional applications are not always compatible with a web-based mode. For example, certain applications required the transmission of huge amounts of data between server and client, and cannot be redesigned for web-based mode. For this and other reasons, many businesses chose not to redevelop their applications and continue to use the traditional client-server model. Then, a new thin-client computing mode: server-based computing(SBC)[5], was populated to fill in the gap.

3 Related Works on Server Based Computing

The first version of server-based computing was provided by the X window system[6], it was originally developed for UNIX and enabled interactive applications running on large servers to accessed from low-cast workstations. But Windows system dominated the world soon. The server-based computing architecture from CITRIX[6] allows a variety of remote computers, regardless of their platform, to connect to a Windows NT terminal server to remotely access a powerful desktop and its applications. A server called MetaFrame runs under Windows NT in the desktop machine and communicates with the thin clients executing at the remote computers using the Independent Computing Architecture protocol (ICA). The ICA client and the MetaFrame server collaborate to display the virtual desktop on the remote computer screen. They also collaborate to process mouse and keyboard events, and to execute programs and view data stored at the server. All executions are remote and none takes place at the client portable computer. A research project at Motorola [7] extended CITRIX's thin client architecture so that it is optimized in the wireless environment. The work pointed out that bandwidth limitation is not as detrimental to the thin client performance as network latency. This is because the thin clients' use of bandwidth is limited. Other server-based computing implementation includes Remote Display Protocol(RDP) of Microsoft[8], Tarantella AIP of Tarantella [9], and Virtual Network Computing (VNC) of AT&T [10][11]. All these systems are quite similar, that is to say, all processing will be done on server and terminal devices works as a remote display. Each system has each proprietary protocol and data compression between server and terminal to reduce both network traffic and terminal side processing. Reduction of network traffic is usually achieved by sending only updated image.

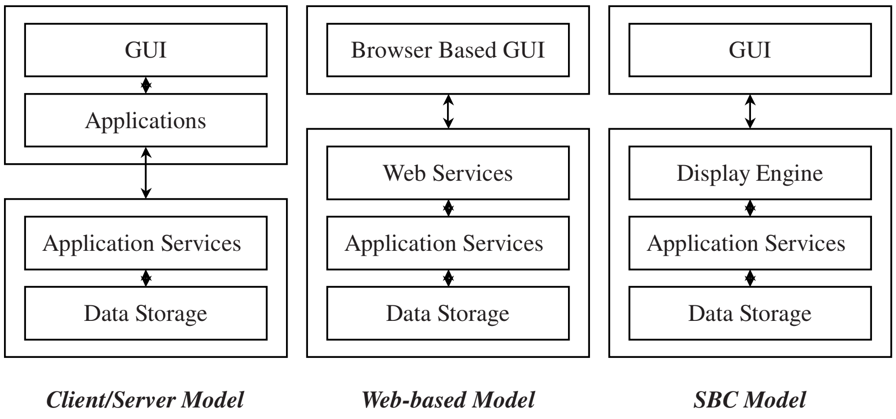

Fig. 1. Three different thin-client computing model

As we have mentioned above, three kind of thin-client computing system exists now: client-server based thin-client computing, web-based thin-client computing and server based computing(Figure 1). Server based computing(SBC) system is the thinnest system of all kinds, client device only displays the remote user interface locally; this is too big a luxury when thin-client like embedded system becomes more and more powerful.

4 The Problems in Server Based Computing

Server based computing technology is becoming mature now, it allows the deployment of traditional client-server applications over the Internet and WANs while reducing maintenance and support costs. But it's weakness is all appearance too, we will check it below.

Firstly, in most SBC system, no differentiation is made between video data and regular graphics data. This means when server decode the video data into separate frames and display it, these frames will be encoded as some static pictures by SBC scheme and transferred to client machines. This consumes lots of computing and network resources on SBC server. Some systems, like Citrix VideoFrame, recognize video data as a different type and supports it through a proprietary codec format. The stream is decoded from its original format, encoded for transmission using the proprietary codec and decoded again at the client side. In this process, the server will be burdened with some unnecessarily transcoding operations [12].

Secondly, remote display protocol has become essential to server based computing. There are many such protocols like RDP, ICA, AIP, VNC, for etc., but only windows based ICA and RDP(developed from ICA) are actually working well for the support of Microsoft. Other protocols like VNC perform well on UNIX, but quite poor on Windows environment. While some works has been done to solve the video playback in VNC system[13], it is quite meaningful to find some common ways for all main systems(especially RDP on UNIX) to share the advantages of SBC.

5 Computing Model

We have known that client-server computing based thin-client system like Java NC has no software installed locally, but they can download the operating system and application software from the server. After downloading, they can execute such applications locally. For example, if you have suitable software written in Java, you can do word processing, draw pictures, or even play videos locally. But such kind of thin client systems are generally considered as an unsuccessful initiative, because they cannot be compatible with Windows system, which is widely used and accepted. SBC system generally uses a windows system, so they become more successful. But for the lack of client local processing capability, the server is often overloaded. So, if we can combine the advantages of two computing mode, we will have a good answer.

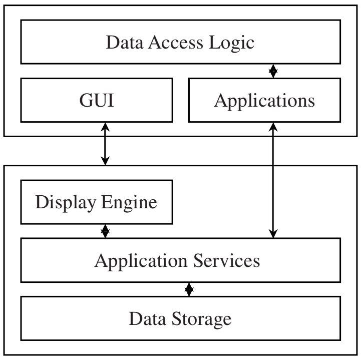

Fig. 2. Revised SBC Model

We can see from figure 2 that this computing model is developed from server based computing. Thin clients connect to a server and display the graphic-user-interface of the server. But they have two functions indeed. One is to act as a graphics terminal; the other is running as a full functional machine. Thin client functions are extended through local application execution capabilities. In this mode, thin client can even has a data access logic, which can store their private data through network.

The model is quite simple; we will show it in detail below with our implementation of a prototype system.

6 Implementation

Thin client of server-based computing runs UNIX or Windows operating system; as for Windows, Microsoft has all the solutions for it. We mainly solve the problem for client running UNIX. Then, we assume that Windows system is running on the server, so the SBC client can display a windows-like interface.

We will describe the current prototype implementation of the system below.

6.1 Hardware

We designed an embedded system for server-based computing, which is named THUMP-NC. It is powered by our embedded micro-processor: THUMP running at 400MHz. THUMP uses a MIPS instruction set, and is fully compatible with MIPS-4KC.

IT8172G is used in our mainboard. It performs high speed, high performance core logic control and data accessing between the CPU bus, system SDRAM bus, and PCI bus.

6.2 OS and Applications Installation

As a thin client system, no hard disk exists in our client machine. So, we have designed two booting mode for it: one is booting from a FLASH module, the other is booting from network.

An optional installed FLASH memory(8MB) can be used to store the OS kernel and some critical applications, which can be used as the booting source. The advantage of this mode is the speed, but the total cost of thin client hardware will be added too.

We have a BIOS like module named PMON in our hardware, it can be used to boot the system from network. In this mode, thin client will get their IP address through a DHCP request; and then, TFTP protocol is used to download the OS kernel from the server. Finally, the system is booted and some remote disks will be used locally through NFS.

6.3 Graphic-User-Interface Seamless Integration

We designed a framework named graphic-user-interface seamless integration to combine the advantages of traditional server-based computing and classical client-server computing.

Our prototype is built on top of rdesktop[14] for UNIX. Rdesktop is an open source UNIX X11 client for Windows NT Terminal Server, capable of natively speaking its Remote Desktop Protocol (RDP) in order to present the user's NT desktop. Unlike Citrix ICA, no server extensions are required.

After booted, our thin client machine will be connected to a Windows server automatically through rdesktop. So, we can log on to the server immediately. After that, a windows graphic-user-interface will be showed on thin client display, but it is partly changed by us.

Firstly, we have modified some of the system icons and shortcuts. In traditional server based computing, all icons and shortcuts on thin client displays mean some corresponding server applications. But in our system, they will be divided into two categories; one is for server applications, the other stands for some local applications on thin client machines. We design a client-server style mechanism for the execution of local applications in our thin client machine. Any request for the execution of corresponding applications of these icons and shortcuts will be passed to a process named PIQUET in SBC server; then, this process will inform the request to a DAEMON running on thin client through network; after receiving the request, the DAEMON will start the correct application on thin client machine, and commit the request at the same time. After that, any window operations like window maximize, window minimize, window close and window move will all be informed between the PIQUET and the DAEMON, in order to keep the whole graphic-user-interface seamless.

Secondly, some file associate relations are modified too. For example, if you find a video file in windows explorer, you may double click it for a playback. Traditional server based computing use a windows media player for this task, but in our system, another media player running on thin client will be in charge. Other applications like Mozilla will be used in place of IE too.

As for users, the real execution locations of applications are not so important; the important thing is: they can use them in one style.

6.4 Local Video Processing

We use VideoLAN[15] for video streaming and playback in our system. VideoLAN is free software, and is released under the GNU General Public License. The VideoLAN project targets multimedia streaming of MPEG-1, MPEG-2, MPEG-4 and DivX files, DVDs, digital satellite channels, digital terrestial television channels and live videos on a high-bandwidth IPv4 or IPv6 network in unicast or multicast under many operating systems. VideoLAN also features a cross-plaform multimedia player, VLC, which can be used to read the stream from the network or display video read locally on the computer under all main operating systems.

We design a streaming server for better using of VideoLAN in our system. In our revised SBC model, local applications in thin clients communicate with thin server in client-server mode. We can look the local applications as a streaming media client, and thin server a streaming server. The easies way to implement a streaming server is through the HTTP service of Windows system. But it is not so reliable. If the server could not send enough media data to the client, video playback will be paused. And then, we studied from traditional video on demand technologies. Interval caching[16] is used in our system. The final stage of our implementation is a file mapping schemes, we look the whole file system in thin server as a video database, any request to the video file will be redirected through the PIQUET and the DAEMON described earlier. This is another main step to approach our seamless GUI.

7 Some Results

S.Jae Yang[12] gave some results to value the effectiveness of a number of server based computing design and implementation choices across a broad range of thin-client platforms and network environments.

Our implementations are only complementarities to server based computing. So, we only care about the differences. We use a machine equipped with PIII800, 256M RAM to act as a SBC server. Two clients are used for test too; one is a Windows PC, the other is our THUMP-NC.

We test the decoding process in three modes. Firstly, we decode the file on SBC server locally; secondly, we connect to SBC server through RDP and run the decoding; finally, we connect to SBC server from our THUMP-NC and do the same thing. That's our results below(Table 1). We can see that the CPU utilization of SBC server is greatly reduced, because neither decoding nor rendering are needed on SBC server in later mode.

Table 1. CPU Utilizations of video playback in different computing mode

Video Source	Local	Windows PC	THUMP-NC
1.2Mbps MPEG1	14%	65%	2%
5.4Mbps MPEG2	56%	90%	4%

8 Conclusions

In server based computing, a terminal server deploys, manages, supports and wholly runs the client part of application. Thin client only takes charge of the display of the graphics user interface of the running result. With the development of embedded systems, the power of traditional thin client devices is greatly boosted up. Some attempts to use such computing power more efficiently have been given in this paper. Such attempts include a seamless integrated GUI and the video support on thin client system, which lighten the load on corresponding server, and make the system easy-of-use.

References

1. Network computing: a tutorial review, Revett, M.; Boyd, I.; Stephens, C.; Electronics & Communication Engineering Journal , Volume: 13 Issue: 1 , Feb 2001, Page(s): 5 -15
2. Network Computers: The Changing face of computing, Ishfaq Ahmad, IEEE Concurrency, October-December, 2000

3. Software architecture of the InfoPad system. In Proceedings of the Mobidata Workshop on Mobile and Wireless Information Systems (Rutgers, NJ, Nov.), 1994.
4. Java NC Computing, http://java.sun.com/features/1997/july/nc.html
5. Server-based computing opportunities, Volchkov, A.; IT Professional , Volume: 4 Issue: 2, March-April 2002, Page(s): 18 -23
6. http://www.citrix.com, 1998
7. Duran, J. and Laubach, A. 1999. Virtual personal computers and the portable network. In Proceedings of the IEEE Conference on Performance, Communication, and Computing (Phoenix, AZ). IEEE Computer Society Press, Los Alamitos, CA.
8. Microsoft RDP & Citrix ICA Feature Overview, http://www.microsoft.com/windows2000
9. http://www.tarantella.com
10. Virtual network computing, Richardson, T.; Stafford-Fraser, Q.; Wood, K.R.; Hopper, A.; Internet Computing, IEEE , Volume: 2 Issue: 1 , Jan.-Feb. 1998 Page(s): 33 -38
11. VNC tight encoder-data compression for VNC, Kaplinsky, K.V.; Modern Techniques and Technology, 2001. MTT 2001. Proceedings of the 7th International Scientific and Practical Conference of Students, Post-graduates and Young Scientists , 26 Feb.-2 March 2001
12. S. Jae Yang, Jason Nieh, Matt Selsky, and Nikhil Tiwari, "The Performance of Remote Display Mechanisms for Thin-Client Computing", Proceedings of the 2002 USENIX Annual Technical Conference, Monterey, CA, June 10-15, 2002, pp. 131-146.
13. Chia-Chen Kuo,Ping Ting. Design and Implementation of A Network Application Architecture for Thin Clients, Proceeding of the 26th Annual International Computer Software and Applications Conference(COMPSAC'2002)
14. http://sourceforge.net/projects/rdesktop/
15. http://www.videolan.org
16. Dan A , Dias D M, Mukherjee R, etc., Buffering and caching in large-scale video servers. Proceedings of COMPCON. San Francisco USA: IEEE Computer Society Press, 1995. 217 224.