Multi-Platform Mobile Thin Client Architecture in Cloud Environment

Wei Tang, Jun-hyung Lee, Biao Song, Md. Motaharul Islam, Sangho Na and Eui-Nam Huh

Department of Computer Engineering, Kyung Hee University, Global Campus, Suwon, South Korea
{wtang, junhyung, bsong, motahar, shna, johnhuh}@khu.ac.kr

Abstract

In a thin client computing architecture, application processing is delegated to a remote server rather than running the application locally. User input is forwarded to the server, and the rendered images are relayed through a dedicated remote display protocol to the user’s device. Thin-client computing offers the promise of easier-to-maintain computational services with reduced total cost of ownership. With the increasingly development of cloud technology, the server side of thin client architecture is able to be deployed in cloud, thus makes the full use of the features of the Cloud, for example, virtualization, flexibility, security, and dynamic management. Meanwhile the client side can also use the mobile thin device. The traditional thin client architecture, which is platform-dependant, however, limits the most benefit of the software providers and end-users. So we propose multi-platform thin client architecture, which use VNC (Virtual Network Computing) protocol and run in the cloud environment. In this architecture, both side of the service will benefit a lot. In the one hand, the end-user can use cheap, light mobile terminal which enjoy all kind of software service of different platform. In the other hand, the software provider or developer can get more benefit with more download mass and wider circulated.

Keywords: component; Thin Client, Cloud Computing, VNC, Virtual Desktop, Distributed System.

1. Introduction

Thin client computing refers to the paradigm in which the user device relies on a remote server to perform a most fraction of its computational tasks. They will provide the user with a GUI but with considerably lower Total Cost of Ownership (TCO) compared to the traditional fat client PCs. Especially in the mobile context, the thin client concept is very promising. Users are able to access demanding applications from mobile devices, which often lack the required processing resources to execute the application locally. Because only basic functionality and processing power is required at the terminal, thin client devices can be made lightweight and potentially energy efficient in cloud environment. [1]

The Cloud computing [2] include a shared servers providing resources, software, and data to computers and other devices on demand with elasticity. The elasticity is the great advantage of Cloud computing, it
can add capacity or applications almost at a moment’s notice. Companies buy exactly the amount of storage, computing power, security and other IT functions that they need from specialists in data-center computing. Virtualization of computers or operating systems hides the physical characteristics of a computing platform from users; instead it shows another abstract computing platform. A hypervisor is a piece of virtualization software that allows multiple operating systems to run on a host computer concurrently. Virtualization providers include VMware, Microsoft, and Citrix Systems. Virtualization is an enabler of cloud computing. The features of Cloud computing exactly suit for the thin client architecture, so we will deploy the server side on Cloud. [4]

In this work, we will utilize the thin client system VNC [3], namely Virtual Network Computing. VNC is one of the most widely used thin client systems. It has an open protocol, known as the Remote Frame Buffer or RFB protocol, and there are many open source implementations of both the client and server. Due to its many implementations, which are available for all major operating systems, it is possible to use VNC across different computing platforms, allowing users to access a Linux machine from a device running Windows, or vice versa. Because it is open source, VNC has been one of the more popular systems for thin client research. Research on VNC has included adapting it for high resolution tiled displays, adapting it to control home appliances, and adapting it for optimal viewing on cellular phones, as well as trying new data compression schemes[12]. With its use in both real world and academic settings, VNC is one of the most ubiquitous thin client systems.

The limitation of the existing thin client system is the platform-dependent, like the Microsoft Windows Terminal Services [7] and Sun Ray [8]. We propose novel platform-independent thin client architecture to solve this kind of problem. There will be a cross-platform software market deployed in Cloud, and we think that the cross-platform market is new trend in future. What drive this architecture? In one word: Economics. From software developer perspective, the cross-platform means more people with different platform device can buy the authority to access the Cloud. From end-user perspective, they can enjoy more applications with low price, lightweight client device.

We provide some background and related works in section 2. In Section 3, we go into detail about the Thin Client Architecture. We discuss future work and conclusion in Section 4.

2. Related work

The typical thin-client platform consists of a client application that executes on a user's local desktop machine and a server application that executes on a remote system. The end user’s machine can be a hardware device designed specifically to run the client application or simply a low-end personal computer. The remote server machine typically runs a standard server operating system, and the client and server communicate across a network connection between the desktop and server. The client sends input data across the network to the server, and the server returns display updates.

Several thin-client computing platforms have been developed. In general, these existing thin-client systems can be classified into two categories according to the mechanisms of representing the display information.[5] The first category of thin-client systems utilizes high-level commands to represent the screen update, such as the Citrix Metaframe[6], and Microsoft Remote Desktop Protocol (RDP)[7]. This method opts for a higher level encoding that is more closely tied to the operating system’s windowing and display commands. In this case, graphics commands are transmitted from the server to the client, which is responsible for processing the updates. So the interpretation of high-level commands heavily depends on the operating systems, it is hard to develop the servers and clients on different operating systems with different rendering mechanisms. The second category of thin-client systems utilizes low-level approaches to represent the screen of remote servers, including VNC [3] and Sun Ray systems [8]. They process
updates to the display on the server and transmit only compressed pixel data representing the new display to the client. The most important is that they are platform-impedent.

3. Proposed thin client architecture

In this paper, we propose a cross-platform compatible solution, which include four subsections to improve the interactive user experiences for various devices. Those are explained in detail following subsections.

3.1. Authentication and Management System

In Figure 1, an overview is presented of the architecture of multi-platform thin client system, leveraged with the Cloud environment and VNC protocol.

The architecture is composed by five components, namely, Mobile Terminal, Authentication and Management System, Cross-Platform Market, Personal Storage Cloud, Multi-Platform Cloud Infrastructure. A mobile thin client is connected wirelessly to the cloud infrastructure in which deployed multi-platform servers. At first the terminal sent a request to Authentication and Management System (AMS), the request include the user’s information, like terminal number and password; terminal information, like the type of terminal device; application information, like the item number of application, name of application, type of platform that application belongs to, the application authorization number, and so on. Then, if this is your fist time to login in, the AMS use this configuration information to deploy the virtual machine, if not, your personal application duplicate information will invoke from personal storage cloud, then continue the applications. When the connection is linked, the VNC server deployed in the Mobile Emulator will works, sending the framebuffer of display to mobile terminal continuously. The Figure 2 below shows how the AMS works.

![Figure 1. The architecture of multi-platform thin-client system](image)

3.2. Cross-Platform Market

The second component is Cross-Platform Market, providing a lot of software belong to different platform to sell. Workflow of Market component is described in Figure 3.

As the Figure 3 described, when you pass the authentication of the system, there will be an icon appear in your mobile terminal named Market, so when you launch the market, the market will send you an application list information, in which describes all kinds of application in detail, includes the name of
application, the platform the application need to run, and the functionality of application and so on. If the user wants to buy one application, just click “Buy”, then the market will have two operations, one is to return the application information to the user, include the icon, software authorization number, and so on. Another is to execute the application in Cloud Infrastructure, actually the application already installed in the Cloud Infrastructure (CI), when any user buy the application, the CI just adds the item number of the user to the authorization list of application.

3.3. Personal Storage Cloud.

The component is about to store the application data and users’ data. The Figure 4 show how it works.

In Storage Cloud, Every application stores a duplicate for every user. The duplicate include personalized settings for the application, some personal archive files for the application. And some personal virtual desktop is stored here too in different format, like a contact file is stored in LDAP[9], some setting file is stored in Hadoop[10], Email files is stored in Eml[11] format. As described in Figure 4, by using three modules, namely Data Read/Write module, Data Conversion Module, Data Type Analysis Module, the databases communicate with application deployed in the Cloud.
3.4. Multi-Platform Cloud Infrastructure

The most important component of the Architecture is Multi-Platform Cloud Infrastructure which is described in Figure 5.

Using virtualization and distributed technology, different mobile emulators are deployed in the cloud, like android emulator, Iphone emulator, and windows mobile 7 emulator and so on. In this way all kinds of applications can run on the Cloud Infrastructure. In every virtual machine, deployed VNC server, to capture the updates of Emulator, then sent to mobile terminal. And as describe in Figure 1, the Cloud Infrastructure connected with other component to compose the whole architecture.

4. Future work and conclusion

In the future, some details can improve to make the architecture better. For example, In Cloud Infrastructure, there can be deployed an optimized algorithm when a user’s request coming, the algorithm can find the best emulator for user to balance the overload of emulator in Cloud Infrastructure. And another issue is about to improve remote computing protocol coding method. Now the remote computing or virtual
desktop is available in LAN or high-speed network environment, but how can make it better in moderate, low speed or WAN network environment, this is the next issue need us to focus on.

With the integrated of some component, using remote computing, virtualization, distributed computing technology, we deployed the Cross-Platform thin client architecture, and issued a new market trend, to satisfied people’s meeting with lowest expense. People can use cheap, but rich interactive experience mobile device to access all kinds of application in different platform.

Acknowledgement

This research was supported by Future-based Technology Development Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education, Science and Technology (2010-0020725)

References