

# Desktop Solution for Mobile Environment Using Mobile Cloud Computing

B. Gopikrishnan<sup>1</sup>, Mr. A. Mani<sup>2</sup>

<sup>1</sup> M.E (CSE), Dept. of CSE, S.A.Engineering College, Chennai, Tamil Nadu, India

<sup>2</sup> Assistant Professor, Dept. of CSE, S.A.Engineering College, Chennai, Tamil Nadu, India

**ABSTRACT** ---A virtual network computing based architecture for accessing the desktops of remote computers from a cellular phone. A viewer is provided on the cellular phone that enables the user to see and manipulate the desktop of various remote systems such as MS Windows, Macintosh, and UNIX. The system to be accessed must be running a VNC server and it must be attached to a network. . In this architecture, a mobile user exploits virtual machine (VM) technology to rapidly instantiate customized service software on a nearby cloudlet, and then uses that service over a wireless LAN .A proxy is used to send the image of the desktop to the cellular phone, to convert different devices, to suppress network traffics, and to support recovery from an unscheduled disconnection. To reduce user effort and solve problems inherent to the cellular phone's small screen, several functions are provided on the cellular viewer. They are Shortcut Guidance and Twin view function.

**Keywords**-mobile computing, cloud computing, Virtual network, Remote access

## I. INTRODUCTION

As mobile device popularity grows, end-user demands to run heavier applications are equally increasing. Although advances in miniaturization continue, the desire to preserve the advantages of weight, size and device autonomy will always impose intrinsic limits on processing power, storage capacity, battery lifetime and display size. Conventional desktop applications need to be redesigned to operate on mobile hardware platforms, thereby often losing functionality; whereas more demanding applications typically require specific hardware resources that are very unlikely to be available on mobile devices. At the same time, the web hosts increasingly powerful computing resources and has evolved to a ubiquitous computer, offering applications ranging from simple word processors, over all-encompassing enterprise resource planning suites to 3D games. Both Microsoft and Google, have developed complete online office suites, called Office Live and Google Apps respectively, that may evolve to all-round alternatives for the mobile office suites. Beyond the conventional office applications, cloud computing broadens the range of applications offered to mobile end-users with demanding applications in terms of graphical hardware. As the cloud infrastructure is

shared among multiple users, these hardware resources can be provided in a cost-effective way.

Essentially, the principle of mobile cloud computing physically separates the user interface from the application logic. Only a viewer component is executed on the mobile device, operating as a remote display for the applications running on distant servers in the cloud. Any remote display framework is composed of three components: a server side component that intercepts encodes and transmits the application graphics to the client, a viewer component on the client and a remote display protocol that transfers display updates and user events between both endpoints. This is illustrated in Figure 1.

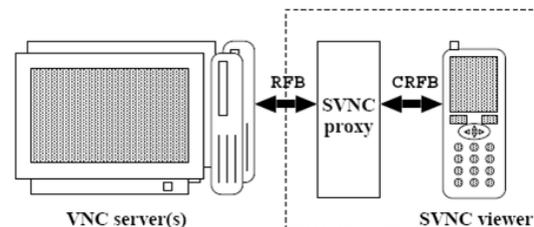


Fig. 1. VNC based architecture

The viewer component on the client forwards the captured user input to the server. In turn, the server side component intercepts, encodes and transmits application output.

Using standard thin client solutions, such as Microsoft Remote Desktop Protocol (RDP), Citrix Independent Computing Architecture (ICA) and Virtual Network Computing (VNC), in a mobile cloud computing context is not straightforward. These architectures were originally designed for corporate environments, where users connect over a wired local area network to the central company server executing typical office applications. In this setting, the technical challenges are limited, because delay and jitter are small, bandwidth availability is seldom a limiting factor and office applications exhibit rather static displays when compared with multimedia applications. In a mobile cloud computing environment, the remote display protocol must be able to deliver complex multimedia graphics over wireless links and render these graphics on a resource constrained mobile device.

## II. PROBLEM STATEMENT

### A. Overcoming limited mobile device battery

The operational time of mobile devices is often limited when extensively used. Over the last decade, the advances in nominal battery capacity have been modest. Pentikousis et al. observe that the technological improvements are currently stagnating, because of the lack of a major battery technology breakthrough, comparable to the advent of rechargeable Li-ion batteries. Consequently, extending device autonomy should primarily be realized by making the device itself more energy efficient.

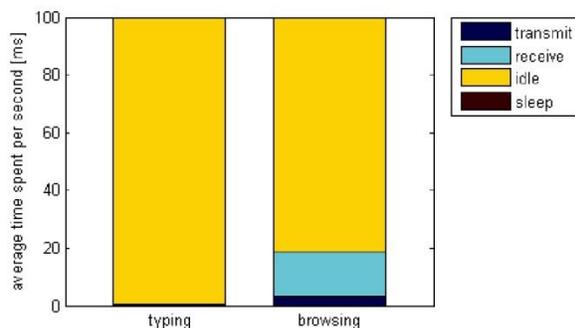


Fig. 2. WNIC never enters sleep mode.

The WNIC energy consumption is the product of the number of bytes exchanged over the wireless interface, and the energy cost per byte. Efficient compression techniques to reduce the amount of exchanged data are covered in section III of this article. The average energy cost per byte is determined by the distribution of the time over the four possible WNIC states: send, receive, idle and sleep mode. Because in each state a specific set of components is activated, the WNIC power consumption largely differs between the different states. Figure 3 visualizes our measurements on the average WNIC time distribution in typical remote display scenarios.

Figures on the typical power consumption in each WNIC state. Although the transmit and receive mode are the most power consuming, power saving approaches should focus on the large idle times that are observed in remote display scenarios. These idle times are a consequence of the limited frequency of user interactions imposed by the network roundtrip time. After some interaction, users will wait until the results become visible on the screen before continuing their work. Furthermore, interactive applications will only update their display when instructed by the user. For example, a browser display is only updated if the user enters a url or clicks on a hyperlink.

Potentially major energy savings are expected when the WNIC transitions to the energy conserving sleep mode during these idle intervals.

The sleep mode is 3-5 times less energy consuming than the idle mode, because the radio interface is turned off. Of course, this implies that the WNIC will miss any incoming data when it is in sleep mode and the sleep intervals must be carefully chosen. Simoens et al. have developed a cross-layer power saving approach that operates between the MAC layer and the remote display protocol layer. Because the MAC layer operates on binary data and cannot discriminate between, for example, transmitted user input and TCP acknowledgements, it is unaware of the arrival of the next display update.

### III. COMPRESSION IN A LIMITED BANDWIDTH

Compared with fixed access networks, bandwidth availability on modern broadband mobile and wireless technologies is limited, variable and expensive. Typically, UMTS users receive up to 384 kbps, while Balachandran et al. report practical throughputs of 347 kbps for LTE and up to 6.1 Mbps for WiMAX. Moreover, the actual throughput will vary due to user mobility and interference and fading effects. Besides technological limitations, economical considerations drive the demand for highly efficient remote display compression technologies. More and more, users are confronted with volume based subscription plans and hence will not tolerate any redundant byte to be sent on the network. For example, AT&T, a leading USA service provider, has adopted volume based pricing models in 2010. It is even expected that this economical dimension might impede the development of new cloud applications

#### B. Graphics encoding

The choice of codec to compress the intercepted application graphics at the server is a tradeoff between visual quality, compression efficiency and decoding complexity. Conventional remote display architectures, such as Citrix ICA, AT&T VNC and Microsoft RDP virtualize the graphical library at the server and forward intercepted drawing primitives to the client, such as instructions to draw a rectangle, to display a bitmap or to put some text on the screen. This approach is optimal for applications that only update small regions of the display, or that have a slow refresh rate with respect to the network roundtrip time, such as typical office applications. Bandwidth requirements to remotely display this type of graphics do not exceed 200 kbps, and can be perfectly served over wireless links. On the other hand, a lot of drawing primitives would be required to encode the graphics of multimedia applications, because these update large parts of their screen at high refresh rates and their graphics often contain fine-grained and complex color patterns. This kind of graphics can be more efficiently encoded by means of a video codec, such as H.264 or MPEG-4 video. Using video codecs for remote display purposes is referred to as interactive live streaming,

because the graphics are mainly the result of user interaction, in contrast to regular video streaming with only limited user interaction, e.g. to start and stop the video. Interactive live streaming has been applied successfully in the context of remote 3D virtual environments and gaming.

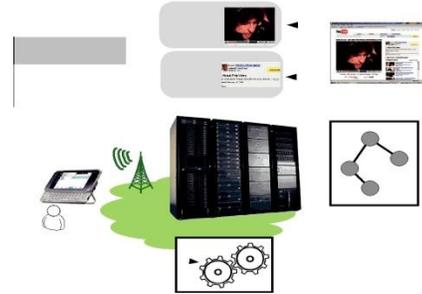


Fig. 3. Graphics encoding

Display updates are optimally encoded through drawing primitives or video codecs, dependent on the characteristics of the graphics. Low- and high-motion scenes must be separately encoded, requiring runtime detection algorithms or the usage of semantic scene description languages. Video streaming parameters need to be adapted to device feedback. Even when only a single application is used, the characteristics of the presented graphics on the user display may largely differ during the period a user is accessing the mobile cloud computing services. For example, users may browse to a Wikipedia page and subsequently click on link that opens a YouTube video in the same browser window. Remote display frameworks must therefore be able to switch seamlessly between multiple encoding modes, based on an analysis of graphics at the server.

Tan et al. compare the pixels of subsequent frames to split each individual frame in low- and high-motion regions, which are respectively encoded through drawing commands or as H.264 video frames. This hybrid approach operates at the pixel level, which offers the advantage of cross-system applicability because it is the lowest layer of the rendering stack. The transparency at the pixel level comes at the expense of losing any information on the nature of each object in the scene. Consequently, the same encoding format will be used for low-motion regions irregardless if they contain text characters or images. Mitrea et al. operate at a higher level in the rendering stack. They intercept high-level X11 drawing commands and encode it through the MPEG-4 Binary Format for Scenes (BiFS), which is



a powerful scene description language. Based on the intercepted X11 commands, an internal scene graph is constructed and converted to BiFS semantics. A distinctive feature of BiFS is that this scene graph, containing the nature of each object in the scene, is binary encoded and streamed to the client. This allows to encode each object in its own optimal encoding scheme.

While the choice of encoding format is mainly determined by the characteristics of the application graphics, the actual encoding parameters need to be dynamically adapted to cope with wireless bandwidth fluctuations and heterogeneous mobile device capabilities. Numerous factors impact wireless link quality, such as device antenna technology, distance from the access point, user speed and fading and interference effects. At the same time, the various hardware configurations of commercial mobile devices induce variations in decoding power. Paravati et al. have developed a closed-loop controller for interactive live streaming that optimizes the settings of the video codec parameters based on feedback from the client device. The mobile device regularly reports on the amount of data that is encoded per unit of time, a metric reflecting both the device hardware capabilities, as well as of the amount of data that is received by the device. By adjusting the resolution and image quality accordingly, the controller aims to maintain a target frame rate to ensure a smooth visualization experience.

### C. Data peak reduction

Interactive applications only update their display unless instructed by the user. Usually, these display updates involve a large amount of data that needs to be sent to the client in a short interval to swiftly update the display. The delivery of these data burst requires an instantaneous bandwidth that is much higher than the average bandwidth requirement. Furthermore, this bursty traffic pattern is unfavourable in wireless network environments, as it might induce additional collisions on the wireless channel. Sun et al. have studied this problem. Their analysis of remote display protocol traffic traces reveals a lot of redundancy, caused by the repainting of graphical objects after recurring user actions. They propose a hybrid cache-compression scheme whereby the cached data is used as history to better compress

recurrent screen updates. The cache contains various drawing orders and bitmaps. Using Microsoft's Remote Display Protocol (RDP) and dependent on the size of the cache, they are able to reduce the number of data spikes by 27-42 %, which results in global network traffic reductions of 10-21 %.

### D. Upstream packetization overhead

User events are the principal source of remote display traffic in the upstream direction from client to server. Individually, each user event embodies only a small amount of information: a key or button id, one bit to discriminate between the press and release action and possibly the current pointer coordinates. Nevertheless, user events induce important upstream traffic because they are often generated shortly after each other. Entering a single character results in two user events to indicate the press and release action, whereas moving the mouse results in a sequence of pointer position updates.

## IV. IMPLEMENTATION OF CLIENT SERVER CONNECTION

### E. Server Connecting Module

Interaction latency, i.e. the delay a user experiences between generating some user input and having the result presented on his display, is key challenge of mobile cloud computing. Whereas bandwidth limitations are likely to disappear with technological advancements, interaction latency is an intrinsic key challenge of mobile cloud computing because even the most trivial user operations need to be communicated to the server. Tolia et al. point out that these trivial interactions, such as moving the pointer to draw a line or to select some text, are far more challenging than loosely coupled tasks, such as web browsing.

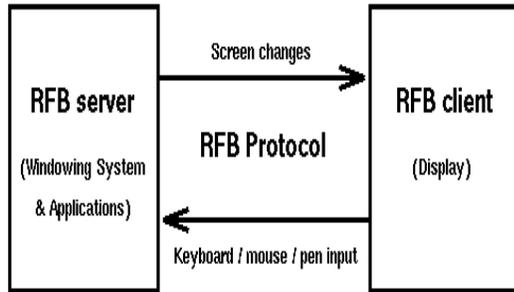


Fig. 4. Remote frame buffer connection.

The major difference is that users expect an immediate visual result of trivial operations, whereas they anticipate processing and download delays when clicking on a link.

#### F. Desktop Viewer Module

Accustomed to the responsive interfaces of desktop applications, users will expect the same interactivity in a mobile cloud computing setting. Remote display protocol data needs to traverse numerous links, both wireless and wired, and numerous network elements, each introducing additional propagation and transmission delays on the end-to-end path. Loss correcting retransmissions on the wireless link, router queuing, suboptimal routing schemes and firewall processing entail important propagation delays. Bandwidth limitations on the wireless link induce additional transmission delays, especially for immersive applications such as virtual environments that transfer highly detailed graphics to the client. Sometimes several client-server interactions are required before a display update can be shown on the screen, e.g. when the server waits for the acknowledgement of the client before sending the remainder of the data.

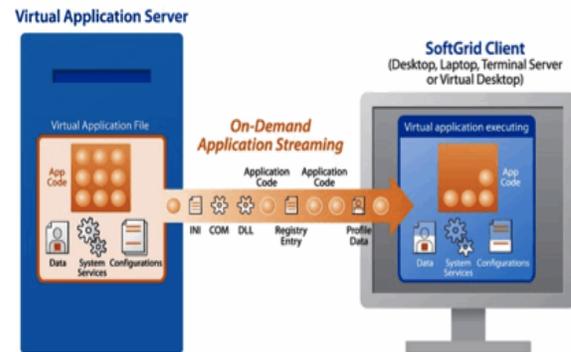


Fig. 5. VNC Application Streaming.

Solutions to mitigate the interaction latency either target a reduction of the number of hops on the end-to-end path by moving the application closer to the client, or better synchronization mechanisms between client and server. Satyanarayanan et al. introduce the concept of cloudlets: trusted, resource-rich computers that are dispersed over the Internet. Exploiting virtual machine technology, mobile devices rapidly deploy their services on the most nearby cloudlet by uploading an overlay virtual machine to customize one of the generic base virtual machines that are commonly available on all cloudlets.

#### G. Mouse Pointer and Key Access Module

Although the cloudlet concept is very promising, it may require the transfer of data from the central application server to nearby public infrastructure. This can be undesirable for security or privacy reasons. In these cases, latency optimization strategies need to focus on a reduction of the number of roundtrip times that is required to resynchronize the client device display with the server. Given the current application state, the application server can predict potential display updates and stream these in advance to the client. Contrary to video streaming, where the frame order is known in advance, in mobile cloud computing the next display update depends on user input. For example, when a user opens an application menu, the server could precompute all dialog windows that can be opened by selecting one of the menu items. Given the current user position, the possible next user viewpoints are calculated in advance and provided to the client.



When the user actually moves forward, the client fetches.

## V. CONCLUSION

By physically separating the user interface from the application logic, the principle of mobile cloud computing allows to access even the most demanding applications in the cloud from intrinsically resource-constrained mobile devices. In this article, we have surveyed contemporary remote display optimization techniques specifically tailored to the short mobile device battery lifetime, the varying and limited bandwidth availability on wireless links and the interaction latency. Although each of these solutions adequately address specific challenges of mobile cloud computing, an overall approach is currently lacking. The context of mobile cloud computing is highly dynamic, owing to the user mobility, the wide diversity of applications, and the varying wireless channel status. Future research should therefore be devoted to the design of an overall framework, integrating all the presented solutions, and activating the most appropriate solutions dependent on the current device, network and cloud server status.

Due to the limitations in mobile bandwidth and mobile device memory resources, it is in most cases unfeasible to stream in advance all possible next display updates, server only needs to provide a differential update. Of course, in this case, the server response is still required to update the display. For more static applications, e.g. office applications, the potential next updates can be more accurately predicted as, for example, the layout of a menu will almost never change. Consequently, the number of corrective server updates will be more limited. One typical example would be the list of recently opened files in the File menu of a text editor. Scene description languages such as MPEG-4 BiFS are particularly suited to support this clientside handling of user input. The client not only receives graphic updates, but is also informed on the structure of the displayed scene and its composing objects, as well as on how the user can manipulate these objects.

## REFERENCES

[1] V. S. Pendyala and S. S. Y. Shim, "The Web as

the Ubiquitous Computer," *COMPUTER*, vol. 42, no. 9, pp. 90–92, SEP 2009.

- [2] F. Lamberti and A. Sanna, "A streaming-based solution for remote visualization of 3D graphics on mobile devices," *IEEE TRANSACTIONS ON VISUALIZATION AND COMPUTER GRAPHICS*, vol. 13, no. 2, pp. 247–260, MAR-APR 2007.
- [3] H. Kawashima, K. Koshiba, K. Tuchimochi, K. Futamura, M. Enomoto, and M. Watanabe, "Virtual PC-type thin client system," *NEC TECHNI-CAL JOURNAL*, vol. 2, no. 3, pp. 42–47, SEP 2007.
- [4] P. Simoens, F. A. Ali, B. Vankeirsbilck, L. Deboosere, F. De Turck, Dhoedt, P. Demeester, and R. Torrea-Duran, "Cross-Layer Optimization of Radio Sleep Intervals to Increase Thin Client Energy Efficiency," *IEEE COMMUNICATIONS LETTERS*, vol. 14, no. 12, pp. 1095–1097, DEC 2010.
- [5] K.-J. Tan, J.-W. Gong, B.-T. Wu, D.-C. Chang, H.-Y. Li, Y.-M. Hsiao, Y.-C. Chen, S.-W. Lo, Y.-S. Chu, and J.-I. Guo, "A remote thin client system for real time multimedia streaming over VNC," in *2010 IEEE International Conference on Multimedia and Expo (ICME), 2010*, pp. 992–7.
- [6] M. Mitrea, P. Simoens, B. Joveski, J. Marshall, A. Taguengayte, F. Pre-teux, and B. Dhoedt, "BiFS-based approaches to remote display for mobile thin clients," in *Proceedings of the SPIE - The International Society for Optical Engineering*, vol. 7444, 2009, p. 74440F (8 pp.).
- [7] G. Paravati, C. Celozzi, A. Sanna, and F. Lamberti, "A Feedback-Based Control Technique for Interactive Live Streaming Systems to Mobile Devices," *IEEE TRANSACTIONS ON CONSUMER ELECTRONICS*, vol. 56, no. 1, pp. 190–197, FEB 2010.
- [8] A. Boukerche, R. W. N. Pazzi, and J. Feng, "An end-to-end virtual environment streaming technique for thin mobile devices over heterogeneous networks," *COMPUTER COMMUNICATIONS*, vol. 31, no. 11, pp. 2716–2725, JUL 15 2008.

## Biography



**B. GOPIKRISHNAN** received the bachelor's degree in 2010 in computer science & engineering from Anna University in E.G.S.Pillay Engineering College. He is currently doing Master degree in computer science & engineering from Anna University in S.A Engineering college. His research interests include Cloud computing, mobile computing, wireless networks etc.



**A.MANI** received a Master Degree in Computer Science and Engineering from S.A.Engineering College, Anna University, Chennai. He is working as Asst. Prof. in Department of computer science & engineering in S.A. Engg college, Chennai, and also he is member of ISTE.