STREAMING TECHNOLOGY AND MOBILE GEOGRAPHIC INFORMATION SERVICE

JaeJun Yoo, JiHoon Choi, KyoungBok Sung, JungSook Kim

Telematics Solution Research Team, Telematics.USN Division
Electronics and Telecommunications Research Institute

{jjryu, cjh63356, kbsung, jungsook96}@etri.re.kr

ABSTRACT:

Recently, as mobile paradigms such as telematics or LBS (Location Based Service) have been being more diverse and mobile devices such as PDA (Personal Digital Assistance) or several telematics terminals have been being used more widely, mobile services, for example, mobile geographic information service have been being put more importance by people.

In this paper, we propose to adopt S/W and data streaming mechanism as a method to improve performance of mobile geographic information service which is a representative of mobile services based on client-server architecture. The show the effectiveness of proposed methods, we design and implement S/W streaming prototype system and sample mobile geographic information service to be executed on the S/W streaming system.

KEY WORDS: S/W Streaming, Data Streaming, Mobile Geographic Information Service

1. INTRODUCTION

Recently, as mobile paradigms such as telematics or LBS (Location Based Service) have been being more diverse and mobile devices such as PDA (Personal Digital Assistance) or several telematics terminals have been being used more widely, mobile services, for example, mobile geographic information service have been being put more importance by people. Also, these mobile services are being extended to include more various functions to be used frequently.

Several mobile services including mobile geographic information system can be mainly categorized into two groups - stand-alone form and client-server type, according to their based architecture. Stand-alone type is the architecture with which application programs and data for services exist in client, that is, terminal. This type doesn't require communications with servers fundamentally. With client-server type, all kinds of application programs and data for the services exist in decentralized fashion on terminals and servers. In this type of services, application programs or data are downloaded when they are requested from user or operating system.

Mobile services in the client-server group also can be categorized into three sub-groups according to the kinds of information or data to be transferred from a server to a client. The first is that data to be required by applications are installed on client side initially, and executable programs are download from server dynamically. The second is that executable programs are installed initially on client side, and data are downloaded from server when they are needed. The third is that all of data and programs are downloaded from server when they are requested. These forms have relative advantages and drawbacks on several aspects such as adequate program size, number of update, and communication cost. For instance, when we download all kind of executable programs and data from server dynamically, we will have the most flexibility in update because we can update software and data at anytime we want through network but communication cost will be maximized since we need to connect to server more frequently.

To reduce communication cost of a mobile service in client-server architecture, several methods are being researched and adopted. With the structure in which data are downloaded as they are accessed, it can be an example that some indispensable data which are necessary n starting the application are pre-installed.

In this paper, we propose to adopt S/W and data streaming [5] mechanism as a method to improve performance of mobile geographic information service which is a representative of mobile services based on client-server architecture. By downloading the only exactly requested part of executable programs and data using streaming mechanism, we can expect to minimize amount of information or data which are transferred between client and server, therefore, reduce communication cost effectively by cutting out some communication overhead.

The structure of this paper is followings. In chapter 2, we explain the design and implement of a system to support mobile services using streaming technology, and in chapter 3, we describe some considerations need to be taken when implementing a mobile geographic information system based on a S/W streaming system explained in chapter 2, and some user interface of a mobile geographic information service implemented to test the S/W streaming system. Lastly, we conclude this paper in chapter 4.

2. S/W AND DATA STREAMING ON LINUX PLATFORM

In this paper, we designed and implemented a system transferring mobile service application programs or data from a server to a client using streaming mechanism on LINUX platform. Open source and well known internal structures of LINUX are the reason why we selected LINUX as client platform. Because the mechanism to access a certain file in an operating system is very different from kind to kind, if adopting this concept to another operating system, we have to consider it a totally separated work.

Figure 1 shows the overall concept of S/W streaming which is adopted through this paper.

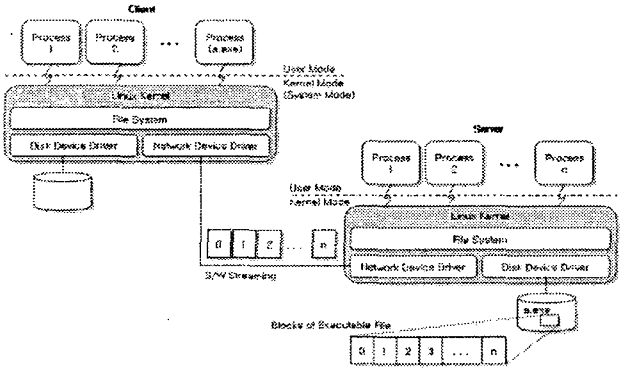


Figure 1. Concept of S/W streaming

Generally, files executed or accessed in a terminal or a server are read or written by conceptually separated layers called file systems or device drivers, which exist under the operating system, block by block or page by page, a unit of fixed sized. With this structure, S/W streaming can be designed and implemented by reading the requested blocks or pages by operating systems from servers through network interface instead of reading them local storages. To implement this S/W streaming concept, we need some supports of operating system to include our codes to operating system's file system level, and fortunately the most of currently existing operating systems provide such concepts and related facilities. In case of LINUX, we can add our codes to operating system using VFS (Virtual File System) [4] and Stackable File System [1] facilities.

We named our file system to add S/W streaming mechanism to operating system as "StreamingFS" which is abbreviation of "Streaming File System". "StreamingFS" not only reads requested blocks or pages of specific files by operating system from server through network but also provides some additional functions such as transferred file caching to prevent re-transferring of already read blocks caused by repeated reading.

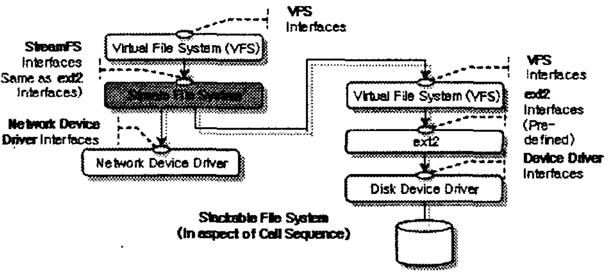

VFS Interfaces
StreamFS Interfaces (Same as ext2 Interfaces)
Virtual File System (VFS)
Stream File System
Network Device Driver Interfaces
Network Device Driver
VFS Interfaces
Virtual File System (VFS)
ext2 Interfaces (Pre-defined)
ext2
Device Driver Interfaces
Disk Device Driver
Stackable File System (In aspect of Call Sequence)

Figure 2. Call sequence of file system interfaces on client side

Figure 2 above shows call sequence of a file system called when we access an existing file. When we try to read a file using VFS interfaces, "StreamingFS" first check whether the requested blocks or pages were downloaded from server previously and stored in local storages or not. If the requested blocks are stored in local storages, they are read through traditional file system access interfaces and if not, they are read through network device driver from server.

Figure 3 below describes the whole structure of system architecture designed and implemented to provide S/W streaming in functional block form. White blocks in Figure 3 mean pre-existing functional blocks in LINUX and Gray blocks represent new functional blocks to be implemented.

On client side, following functional blocks are newly considered.

App Launcher It is a daemon process to execute the requested application after some pre-setting operations which are needed to provide S/W streaming for the application. Also, it passes some information about the selected application to Mount Manager.

Mount Manager It manages reference relationships among several file systems including StreamingFS, keeps some information about application which are being executed in streaming fashion currently, and passes some detail data about executable applications to StreamingFS.

Streaming Protocol It is a predefined communication protocol specially designed for transferring application programs to be executed, information about the requested blocks or pages by operating system, and current status of client between a client and a server.

StreamingFS (Streaming File System) As the core part t of S/W streaming system, it provides file system interfaces letting operating system access itself, sends detailed information about application or general files to servers, and receives the requested blocks or pages as results. It has other three main components called Cache Manager, Common Component Manager, Streaming Manager, each of which is explained below.

Cache Manager To prevent retransmission of the request blocks or pages when the blocks are accessed again, it is necessary to save downloaded data or blocks in local storages. This is a kind of cache operation. This cache operation is supported by a functional block called Cache Manager. Cache Manager provides also fundamental cache interfaces such as replacement operation.

Common Component Manager It is a part to prevent retransmissions of repeatedly requested files. The point that we consider not repeated references of some blocks in a file but repeated references among several files commonly is different from that of Cache Manager. To analyze common references of a file between several applications file level analysis processes should be executed in advance on server side.

Streaming Manager It actually send some information and data to server and receives the result from server. To communicate with servers, it interprets Streaming Protocol.

On server side, following functional blocks are newly introduced.

Streaming Server As the core functional block on server side to provide streaming, it receives some information about the requested blocks or pages, or current client status and sends the requested blocks or pages as the result of processing. It also manages target application programs to be transferred in streaming fashion.

Pack Files To enhance of streaming mechanism, several files of an application in streaming fashion are put together into a logical and physical package called "Pack File". Generally, a "Pack File" can be seen as a general file. A "Pack File" contains additional index information such as what files consists of the application, what the relationships of the files are and so on.

Pack Generator It is a tool which creates Pack Files by analyzing dependencies among several application programs. Moreover, it analyzes what file are repeatedly used by several programs and finds the common components or files among several applications.

Program Info Database It is an information database which stores all kind of data about streaming system. For example, information about applications to be transferred to clients in streaming fashion, file size, creation data, application version, and operating system type on which the application can be executed can be stored and queried.

Program Info Extraction Tool It is a small tool which can extract all kind of information about application programs to be streamed. The extracted information is stored in "Program Info Database".

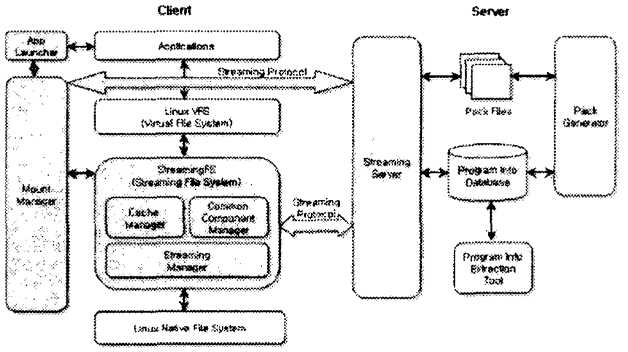
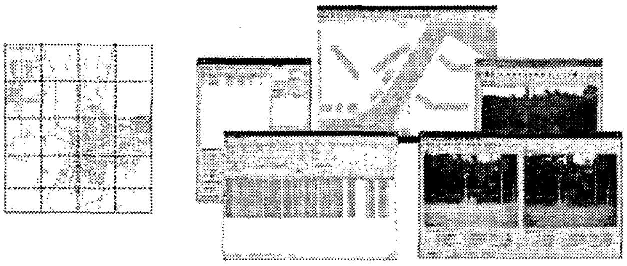


Figure 3. System architecture for S/W streaming

3. STREAMING MECHANISM AND MOBILE GEOGRAPHIC INFORMATION SERVICE

Mobile geographic information service which can be considered as a representative of mobile services is to provide and support some geographic information and additional attribute data to users through visual user interfaces or text type interfaces in mobile environments.

With a mobile geographic information service, many methods are being proposed to reduce amount of information or data transferred from a server to a client. A famous approach among several patterns is to split geographic data into many grids and to send only small pieces of grids to the client. Figure 4 shows the sample grid and a tool which provides some functions related to grid operations of geographic data. In this paper, we used a editing tool called VideoGIS [3] to edit geographic data which was developed by ETRI.



(a) Sample Gird (b) Edit tool of geographic information
Figure 4. Grid split

When adopting S/W and data streaming in mobile geographic information service, we can greatly reduce amount of transferred data since only requested data are downloaded from server. However, if we some pre-processing steps related to data used by mobile geographic information system, we can make transferred data from a server to a client smaller. In other words, if the size of a grid unit is set to a specific size among multiples of a block or a page used in operating system, the number of transmission to download a grid unit can be minimal. Therefore, to apply S/W and data streaming system explained above to grid split of geographic information more efficiently, we fixed the size of a unit of grid to multiples of a block or a page. Because in most operating systems including LINUX and Windows XP the size of a block or a page is currently 4K [2], although it can be set differently from time to time, we made the size of a grid 4K, 8K, 12K, and so on. We make the selected size of a grid changeable according to current zoom level in mobile geographic information service.

To divide whole geographic data into several grids of fixed size can be very different based on the format of geographic data. To divide vector formatted geographic data into even sized grids is more difficult than to divide image fomatted geographic data is because in image formatted data each grid is same sized but in vector formatted data each unit of grid can be different size according to number and type of geographic objects. We need more researches and time to get more elegance algorithms to make each vector formatted grid even sized efficiently.

4. USER INTERFACE

In this paper, we implemented the designed server on Windows XP platform and the designed client on LINUX Platform, using Redhat Linux 9.0.

User interfaces of the system designed to transfer some applications and geographic information data in streaming pattern are shown in Figure 5. Figure 5.(a) shows the user interfaces of servers, and Figure 5.(b) represents the user interfaces of a mobile geographic information service program being executed with streaming on client side. The server side user interfaces include some web pages to manage registered applications, to show some database information, and so on.

(a) Server side user interface (b) client side user interface
Figure 5. User interface

5. CONCLUSION

In this paper, we proposed to adopt S/W and data streaming mechanism as a method to improve performance of mobile geographic information service which is a representative of mobile services based on client-server architecture. The show the effectiveness of proposed methods, we designed and implemented S/W streaming prototype system and sample mobile geographic information system to be executed on the S/W streaming system. With further researches, we expect that more complicated mechanisms and superior structures or algorithms can be adopted, and more excellent result can be come out.

References

[1] Erez Zadok, Ion Badulescu, "A Stackable File System Interface For Linux," http://www.filesystems.org/docs/linux-stacking/

[2] David A. Rusling, "Memory Management," http://www.tldp.org/LDP/tlk/mm/memory.html

[3] JaeJun Yoo, In-Hak Joo, Kwang-Woo Nam, Jong-Hun Lee, "The Design and Implementation of A Video Geographic Information System," Proceeding of Korean Information Science Society Fall Conference, Korea, vol. 29, No. 2. Sept. 2002.

[4] Neil Brown, "The Linux Virtual File-system Layer", http://www.cse.unsw.edu.au/~neilb/oss/linux-commentary/vfs.html

[5] Pramote Kuacharoen, Vincent J. Mooney, Vijay K. Madisetti, "Software Streaming via Block Streaming," Design, Automation and Test in Europe Conference and Exhibition, Munich, Germany, p. 10912, 2003.