Journal of Knowledge Information Technology and Systems

ISSN 1975-7700

http://www.kkits.or.kr

Comparison of System Resource Usage for Screen Image Transmission

Oh-Sung Kwon*

Department of Computer Education, Gongju National University of Education

ABSTRACT

In General, the Screen Control Programs contains the functions that is able to capture PC screen and send the images to a predefined server. The applications, for examples, are PC remote controller, remote login, video conferencing and PC screen recording programs. These application have to run with background process style at PC. Also, the programs need to executed with minimum CPU time and small memory requirement. The programs must not disturb PC users' ordinary activities and generate time delay of mouse or keyboard input. For the effectiveness and efficiency of our proposed program, We perform several experimentals to find the best algorithm and implementaion method. We are preferentially designed image generation and transmission modules each. Our program is implemented with the two modules. We do experimental work in the two ways. The one method is unified manner combined with image generation and transmission. The other one is seperation manner of the two modules. In this paper, We measure the CPU time and the consumption of memory working set to each method. In the experimentals, the unified method is 8.09 % higher than the seperation method in CPU time. In the memory usage, the seperation method is higher 20.9 % than the unified method. On the other hand, the combined method shows sometimes peak increasement of CPU time. We proposed a dynamic and selective approach of the execution style based on the PC resource environment.

KEYWORDS: Remote control, Screen recording, Screen conference, CPU time, Memory working set, PC monitoring, Minimum memory process

ARTICLE INFO: Received 18 January 2018, Revised 9 February 2018, Accepted 9 February 2018.

*Corresponding author is with the Department of Computer Education, Gongju National University of Education, Bonghwang-Dong Gongju-Si Chungnam-Do, KOREA.

E-mail address: oskwon@gjue.ac.kr

1. 서 론

본 논문은 개인용 컴퓨터에서 원격 접속, 화상 회의, 화면 녹화 등의 응용을 위하여 연속적으로 생산되는 이미지 리스트가 있다고 가정하고, <그림 1>과 같이 실시간으로 서버로 전송하는 방식에 관한 것이다.

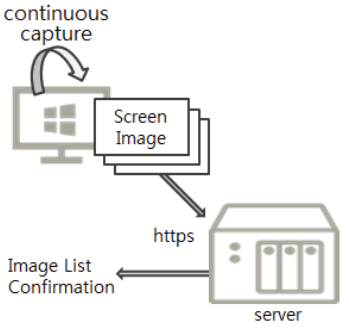

그림 1. 전체 시스템의 구조
Figure 1. Overall Structure of Our Program

실시간으로 이미지를 생산하고 전송하는 응용 분야는 다양하게 적용될 수 있는 데, 그 예를 구체적으로 들면 다음과 같다.

(a) 타임랩스 동영상 변환에 필요한 원격지의 이미지 전송 기술
(b) 원격지의 컴퓨터 접속 및 제어 프로그램
(c) 컴퓨터 사용 감시용 연속 화면 캡쳐
(d) 화상 회의를 위한 상호 간 화면 전송

위에 열거한 응용 중에서 (a)의 경우는 원격지의 이미지를 전송 받아서 제작하는 타임랩스 동영상 변환 기술에 해당한다. (b)의 경우는 원격 접속 프로그램에서 대상 PC의 화면을 연속적으로 전송받기 위한 프로그램에 해당한다. 원격지의 사용자가 자기 PC처럼 사용하기 위해선 실시간으로 상대의 PC 화면의 변화를 보아야 하기 때문이다.

(c)는 중요 기밀 자료나 회계 시스템을 운영하는 기관에서 내부자의 PC 사용 감시를 위하여 사용한다. 내부자의 기밀 유출에 대비하기 위한 방편으로 직원들의 PC에 기밀 유출 방지용 소프트웨어를 설치하여 의심스런 자료 유출 행위를 감시하기도 한다.

(d)의 경우는 서버를 사이에 두고 상호 간에 화면과 소리를 주고 받으며 회의를 진행하는 화상 회의에 해당한다. 이 경우에 서버는 양 쪽에서 발신하는 연속 화면 이미지 정보를 실시간으로 저장하고 제공해야 한다.

본 논문은 2 장에서 연속적인 화면 녹화 및 전송에 관련한 기술 동향과 알고리즘을 살펴보고 3 장에서는 본 논문에서 사용하는 이미지 수집 및 전송 알고리즘의 개요를 설명한다. 4장은 제시된 알고리즘 방식에 따른 시스템 리소스의 소모량을 메모리 및 CPU 시간을 중심으로 측정하고 비교한다. 끝으로 본 논문에서 제안하는 방법의 타당성과 문제점을 제시하고 결론을 맺는다.

2. 관련 연구

본 논문의 주제인 PC 화면 이미지 수집과 전송 기능은 원격 PC 로그인, 원격 제어, PC 보안용 화면 로그 수집, 화상 회의, PC 화면 녹화 등의 응용에 주로 적용되고 있다[13].

다양한 응용 분야 중에서 화상 회의에 관련한 연구는 [7]을 들 수 있으며 회의실에 설치된 몇 개 카메라의 이미지를 수집하고 전자펜 판서 정보와 함께 회의 내용을 캡쳐해서 전송하고 저장 및 검색 시스템을 설명하고 있다. 이러한 시스템의 구성에서도 본 논문의 주제인 수집된 이미지를 수집하고 전송하는 기능이 필수 요소로 포함되어 있다.

PC 보안 소프트웨어는 회사의 중요 자료의 불법적인 유출을 막기 위한 보안 프로그램은 금융 회

사나 연구소를 중심으로 주로 적용되고 있다. [6]의 연구는 PC의 화면 이미지를 중심으로 다양한 로그 정보를 수집하는 연구를 진행하였다. 또한 [3]의 연구처럼 역으로 스파이웨어(Spyware)의 화면 캡쳐를 방지하는 기능에 관한 연구도 PC 보안 차원에서 연구되었다.

화면 녹화는 온라인 강의 진행의 후속 학습을 위하여 적용되기도 한다. [8,9]의 연구에서처럼 화면 녹화 도구를 활요하여 의학 등 다양한 교육 활동의 보조 컨텐츠로 활용하는 연구도 보고되고 있다.

화면 캡쳐는 PC 의 모든 화면 변화를 대상으로 하지만 하드웨어 오버레이를 사용하는 비디오 게임 등은 별도의 캡쳐 방식이 요구되며, [10] 과 같이 캡쳐 화면 이미지의 크기를 줄이기 위한 다양한 연구와 특허가 보고되고 있다[5,14,15].

앞서 설명한 PC 보안, 화상 회의 등의 소프트웨는 일반 응용 프로그램으로 분류하기 보다는 다른 프로그램과 함께 실행되는 유틸리티(utilities) 성격을 갖는다. 그러므로 컴퓨터 사용 시 상시 운영되는 점을 감안하여 PC 리소스를 적게 사용하도록 설계되여야 한다. 특히 CPU 시간과 램 (Ram) 메모리의 사용이 적어야 하고 다른 응용 프로그램의 실행에 영향을 최소화되도록 구현되어야 한다[11].

3. 화면 이미지 전송 프로그램

3.1 프로그램의 전체 구조

화면 전송 프로그램은 크게 화면 이미지 생성부와 서버 전송 부로 구성된다. <그림 2>에서 함수 A와 함수 B로 표시되었고 이 두 부분은 공유 폴더를 매개로 화면 이미지를 전달받는다. 함수 A는 정해진 주기 혹은 이벤트(event)를 기준으로 화면 이미지를 캡쳐하고 압축하여 jpg 형식으로 공유 폴더에 저장한다. 함수 B 는 주기별로 공유 폴더에 쌓인 화면 이미지 파일을 서버 NAS 로 전송한다. 파일의 크기는 화면 해상도와 듀얼(dual)혹은 싱글(single) 여부와 jpg 압축 품질에 따라 다르게 생성된다. 본 논문에서는 시스템을 구성하는 함수 A와 함수 B를 한 프로그램으로 구현하는 지 아닌지 별개의 프로세스로 구현하는 경우에 발생하는 CPU 시간과 메모리 요구량을 측정하였다.

화면 이미지 전송은 원격 접속, 화상 회의, 화면 녹화 응용에서 컴퓨터의 사용 현황 정보를 이미지로 전달하는 역할을 수행하므로 이미지 전송 프로그램은 최소의 메모리와 cpu 시간 소모가 이상적이다.

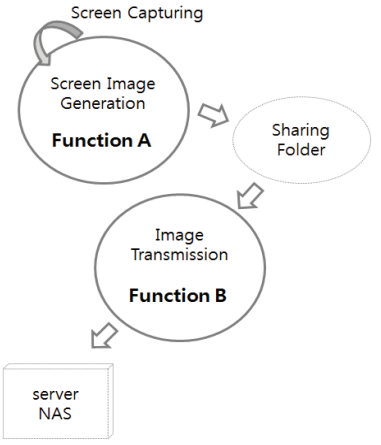

그림 2. 이미지 생성과 전송 간의 관계
Figure 2. Relation of Image Generation and Transmission

3.2 이미지 생성 기능

사용 PC 화면 이미지는 정한 주기 혹은 이벤트 트리거 방식으로 수집한다. 고정된 주기로 이미지를 수집하는 경우는 이미지의 연속성이 확보되므로 이미지 리스트를 동영상 코덱(image codec)으로 압축하여 그 크기를 줄여서 <그림 3>의 절차로 전

송할 수 있다. PC 운영체제는 기본적으로 화면 캡쳐(PrtScr)와 활성화 창 캡쳐(Alt+PrtScr)를 제공하며 마우스 위치와 모양을 포함하지 않는다. 이를 위해서 본 논문에서는 마우스 궤적 추적을 위한 후킹 기능을 포함하였다[1,2,4,12].

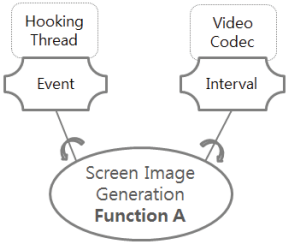

그림 3. 이미지 생성 과정

Figure 3. Image Generation Procedure

3.3 이미지 전송 기능

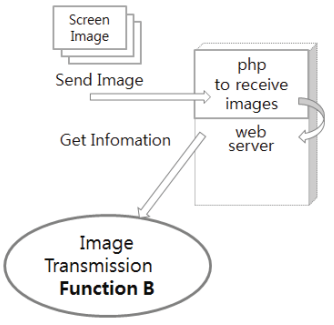

그림 4. 이미지 전송 과정

Figure 4. Image Transmission Procedure

<그림 4>는 이미지 전송 절차를 설명한다. 전송 프로토콜은 Https를 사용하여 구현하였다. Ftp 프로토콜은 관련 포트 접근이 금지된 경우가 적지 않기 때문에 웹 프로토콜을 사용하는 방식으로 구현하였다. Https 의 경우 웹서버(web server)에 파일을 전송하여 저장하는 경우 서버에 관련 작업을 수행하는 php 인터페이스가 필요하고 서버에 저장된 파일의 경우 단순한 다운로드만으로도 필요한 정보를 수집할 수 있다.

4. 구현 및 실험 결과

앞서 설명한 이미지 전송 방식의 비교 실험은 <표 1> 환경에서 진행하였다.

표 1. 실험 환경

Table 1. Experimental Environment

항목	조건
운영체제	Windows 7 Enterprise 64 bit
CPU	Intel Core(TM)i7-6700HQ 2.60Hz
RAM	8.0 GB
해상도	936 × 1100

4.1 실험 환경

일반적으로 컴퓨터 프로그램의 시스템 자원 사용 효율성을 측정하기 위해서 CPU 타임과 메모리 점유율을 측정한다. CPU 타임은 한 프로그램이 CPU 를 차지하는 시간의 양을 뜻하고, 보통 클럭(clock) 또는 틱(tick)으로 표시하며, 메모리 점유율은 프로그램 실행 중 요구되는 메모리(ram)의 용량이다. 본 논문에서는 제안 방식의 자원 사용 효율성을 검사하기 위하여 이 두 항목을 사용하였다. 측정은 전체 프로그램 혹은 기능별 쓰레드 별로 진행하였다. 캡쳐하는 화면은 <그림 5>와 같은 동일 화면을 사용하여 전송되는 이미지 크기의 변화를 최소화하여 보다 정확한 시스템 리소스 소비량의 비교가 가능하도록 하였다. 시스템 리소스 모니터링 도구는 윈도우 시스템 도구인 성능모니터 프로그램 perfmon.exe 를 사용하였다.

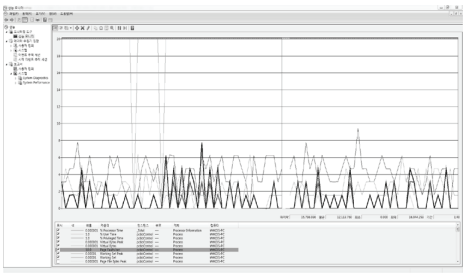

그림 5. 실험에 사용된 표준 캡쳐 이미지

Figure 5. Standard Captured Image for Our Experimentals

<그림 6>은 실험 PC의 전체 프로세서와 제안 프로그램의 CPU 시간을 함께 표시한 그래프이다. 그림에서 보듯이 전체 프로세서 시간의 영향없이 제안하는 프로그램은 일정 CPU 소모량을 유지함을 확인할 수 있었다. 시간 그래프의 등락은 관찰되었지만 전체적으로 일정 수준의 CPU시간을 유지하는 안정성을 보였다.

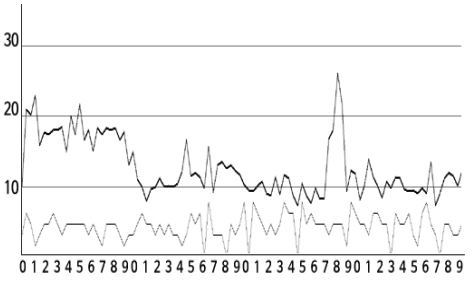

그림 6. 전체 시스템과 실험 프로그램의 CPU 시험 비교

Figure 6. CPU Time Comparison of Total System and Our Program

4.2 CPU 사용량 측정

4.2.1 통합 방식의 CPU 사용량

실험은 주기적으로 화면 캡쳐 동작을 진행하도록 하고 듀얼이 아닌 단독 화면을 사용하였다. 시스템 리소스 모니터링 도구는 윈도우 시스템 도구인 성능모니터 perfmon.exe 를 사용하였다.

<그림 7>은 녹화 기능과 전송 기능을 하나로 결합하여 구현한 경우의 CPU 사용률을 보여준다. 실험 결과 평균 7.04% 사용률을 보였다. 실험 그래프 선두의 급격한 사용률 증가는 프로그램의 프로세스 로딩 시간으로 해석될 수 있다. 이 프로그램은 이미지를 수집하여 저장하는 부분과 저장 이미지를 전송하는 부분으로 구성되며 <그림 7>의 경우는 그 간격을 2초로 하였다. 그림에서 '○' 으로 표시된 부분은 CPU사용이 10%를 넘기며 로컬 피크를 보이는 부분을 표시한다. 이러한 피크부가 나타나는 경우 남겨진 파일의 전송이 완료되지 못한 상태에서 이미지 수집이 겹쳐질 때 발생하는 것으로 판단된다.

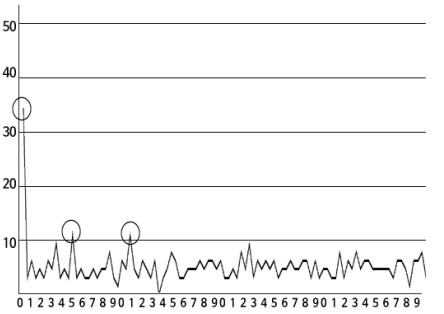

그림 7. 결합 방식의 CPU 시간 (2초 주기)

Figure 7. CPU Time of the Combined Method (2 Second Interval)

<그림8 >은 <그림 7>처럼 녹화와 전송을 같이하는 경우로서 전송 주기를 5초로 한 경우이다. 그림에서 '○' 으로 표시되는 피크 부분이 산발적으로 자주 발생함을 볼수 있다. 이러한 피크부가 자주 나타나는 경우 남겨진 파일의 전송이 완료되지 못한 경우이고 다음 전송 시기까지 영향을 줄 수 있다. 실험에서 사용한 5 초 주기 동안에 누적된

파일 양을 보내는 작업이 전체 프로그램에 부하가 되고 있음을 관찰할 수 있었다.

4.2.1 분리 방식의 CPU 사용량

다음 실험은 앞 실험과 다르게 이미지 수집과 전송을 분리하여 별개의 프로그램으로 구현하고 실행하는 방식이다. <그림 9>는 분리 프로그램 중 녹화기의 CPU 시간을 조사한 것이다.

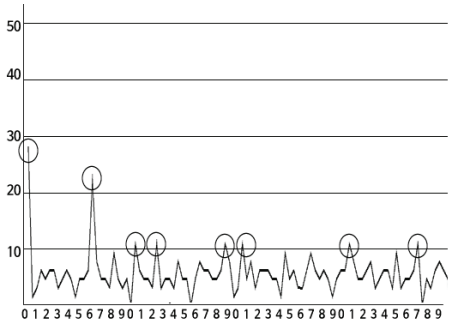

그림 8. 결합 방식의 CPU 시간
(5초 주기)
Figure 8. CPU Time of the Combined Method
(5 Second Interval)

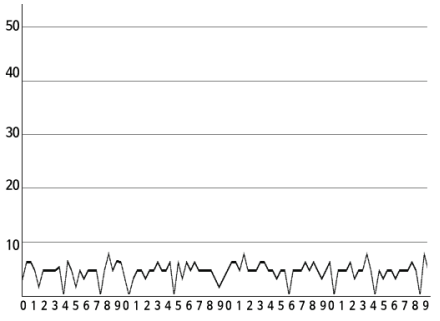

그림 9. 녹화 프로세스의 CPU 시간
Figure 9. CPU Time of Recorder Process

녹화기의 CPU 평균 사용률은 4.39 %를 보였으며 최대값도 9.83 %를 넘지 않았다.

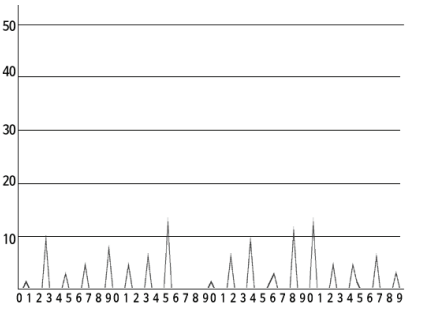

그림 10. 전송 프로세스의 CPU 시간
(5초주기)
Figure 10. CPU Time of Transmission Process
(5 Second Interval)

5초 간격으로 파일을 전송 기능의 프로세스는 2.20 % 의 평균 CPU 이용률과 최소 0, 최대 4.72 %의 사용률을 보였다<그림 10>. 알고리즘 흐름에 따라 5 초 간격의 주기적인 CPU 시간 피크를 보였고 피크 시 CPU 이용률의 크기에 차이가 나는 것은 매 간격마다 미전송 파일량의 차이로 조사되었다. 결국 주어진 간격 5초 이내에 발생한 이미지 리스트를 모두 전송하지 못하 결과물이 다음 전송 시에 영향을 주는 것으로 조사되었다.

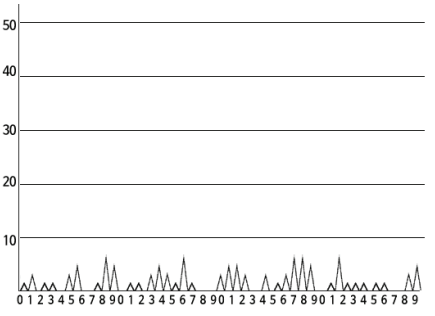

그림 11. 전송 프로세스의 CPU시간
(2초 주기)
Figure 11. CPU Time of Transmission Process
(2 Second Interval)

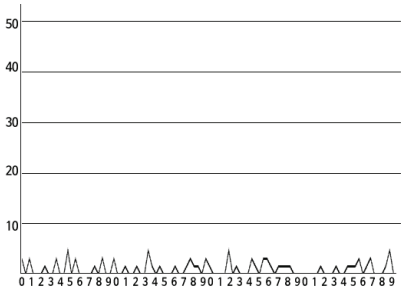

그림 12. 전송 프로세스의 CPU 시간
(1초 간격)
Figure 12. CPU Time of Transmission Process
(1 Second Interval)

<그림 11>은 2초 간격으로 파일을 전송하는 경우 평균 2.08% 최대 7.08%의 CPU 사용률을 보였고, <그림 12>는 1초 간격인 경우는 평균 2.23% 최소 0% 최대 6.24%의 cpu 사용률을 보였다. 결국, 파일 전송 간격을 1~2초 하는 경우 파일 저장과 전송의 버퍼링에서 대기열 증가 현상이 적음을 확인할 수 있었다.

4.3 메모리 사용량 측정

4.3.1 통합 방식의 메모리 사용량

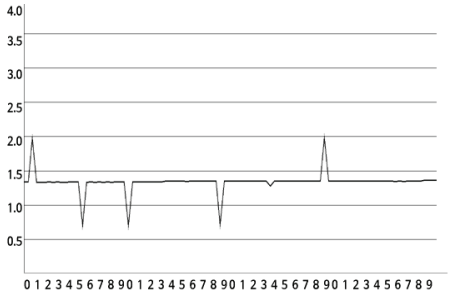

그림 13. 결합 방식의 램 작업셋
Figure 13. RAM Working Set of Unified Method

제안 방식의 메모리 사용량을 검사하였다. 메모리 측정 항목 중 Working Set-Private 측정을 사용하였다. 측정 결과, 녹화기와 전송기를 결합한 일체형 프로그램은 <그림 13>과 같은 변화 패턴을 보여 주었다. 평균적으로 15,512 KiloByte 의 램 메모리 사용량을 보였다. 그림에소 보듯이 주기적인 메모리 사용량이 순간적으로 높아지는 현상은 녹화 이미지를 서버로 전송하는 시간 주기에 해당하는 부분으로 관찰되었다.

4.3.2 분리 방식의 메모리 사용량

제안하는 프로그램의 기능을 녹화와 전송 둘로 각각 나누어 별개 프로그램으로 구현 한 후 메모리 사용량을 측정하여 <그림 14>와 같은 결과를 얻었다. <그림 14>에서 위쪽 그래프가 녹화기, 아래쪽 그래프는 전송기의 메모리 요구량을 표시하고 있다. 그림에서 보듯이 전송기의 경우 공유 장소의 캡쳐 이미지를 차례대로 하나씩 입력하여 처리하는 식이기 때문에 특별한 메모리 증가 현상이 발견되지 않았다. 반면에 녹화기는 주기적인 화면 캡쳐와 이미지 저장을 반복하면서 리소스 할당과 해제를 반복하는 데 이에 따라 메모리 요구량이 순간적으로 오르거나 내리 현상이 반복되는 것을 관찰할 수 있었다. 실험 결과 전송기는 6,564KB, 녹화기는 12,192KB 내외의 메모리 요구량을 보였다.

결국, 위 메모리 요구량 시험을 통하여 제안하는 프로그램의 기능을 하나로 통합한 경우는 15,512 KiloByte, 둘로 기능을 나누어 시행하는 경우는 18,756 KiloByte 로써 3,244 KiloByte 더 많았고 상대적으로 20.90 % 많음을 확인할 수 있었다.

두 가지 경우의 실험에서 각기 서로 다른 메모리 요구량을 보였지만, 시간 진행에 따라 메모리가 늘어나는 현상은 발견되지 않았다. 결국, 안정적인 메모리 요구와 해제가 구현되었음을 확인할 수 있었다.

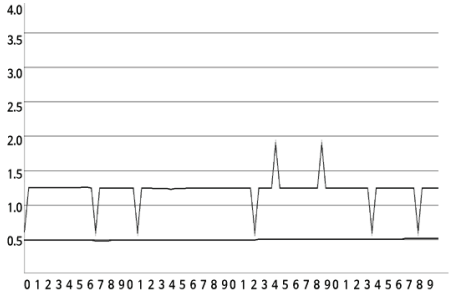

그림 14. 분리 방식의 램 작업셋

Figure 14. RAM Working Set of Seperation Method

5. 결 론

본 논문에서는 이미지 수집과 전송을 통합한 방식과 분리한 방식의 CPU 사용량과 메모리 요구량을 비교 분석하였다. 실험 결과, CPU 사용량은 통합 방식보다 분리 방식이 8.09 % 정도 적은 것으로 확인되었다. 녹화기와 전송기를 결합한 경우에 일정한 CPU 사용량을 보이다가 국소적 피크를 산발적으로 보이는 현상이 있었다. 이러한 현상은 짧은 시간의 피크 현상이지만 전체 프로세서 시간에 영향을 줄 수 있고 컴퓨터 사용 시 입출력 등에서 버벅거림 현상을 발생시킬 수 있다. 또한 하나의 CPU를 여러 사용자가 공유하는 가상 PC (Virtual PC) 상황에서는 그 부작용이 증폭될 수 있다.

메모리 사용량은 통합 방식이 분리 방식보다 20.90% 정도 적었다. 분리 방식인 경우는 프로세스마다 필요한 절대 운영 메모리의 요구가 있기 때문인으로 보인다. 결국 통합 방식은 메모리 요구량에서 우위를 보였지만 CPU 점유가 상대적으로 컸고 분리 방식은 그 반대의 현상을 보였다.

References

[1] H. Jung, and S. Park, *Combination of an adaptive hypermedia system and an external application using a message hooking mechanism*, The Journal of Korean Association of Computer Education, Vol. 8, No. 4, pp. 107-114, 2005.

[2] J. Berdajs, *Extending applications using an advanced approach to DLL injection and API hooking*, J. of Software: Practice and Experience, Vol. 40, No. 7, pp. 567-584, 2010.

[3] Johnny Lim, "*Defeat Spyware With AntiScreen Capture Technology Using Visual Persistence"*, Proceedings of the SOUPS, pp. 147-148, July 2007.

[4] J-G. Han, J-C. Kim, K-J. Ban, C. Kim, and E-K. Kim, *Personal firewall operating system using API hooking modules*, Fall Conference Proceedings, The Korean Institute of Maritime Information and Commucation Sciences, pp. 551-553, 2011.

[5] K-Y. Jang, and E-T. Kim, *Adaptive intra fast algorithm of H.264 for video surveillance*, The Journal of Korea Information and Communications Society, Vol. 33, No. 12C, pp. 1055-1061, 2008.

[6] 0-S. Kwon, *Implementation of system usage recorder for personal computer security*, Journal of Knowledge Information Technology and Systems(JKITS), Vol. 9, No. 4, pp. 427-434, Aug. 2014.

[7] P. Chiu, A. Kapuskar, L. Wilcox, and S. Reitmeier, *Meeting capture in a media enriched conference room*, Proceedings of the Second International Workshop on Cooperative Buildings, Integrating Information, Organization, and Architecture, pp. 79-88, Oct. 1999.

[8] S-S. Hwang, *A method for creating teaching movie clips using screen recording software: usefulness of teaching movies as self-learning tools for medical students*, J of Korean Radiol Soc. Vol. 56, No. 4, pp. 395-402, Apr. 2007.

[9] T. Wales, *Captivating open university students with online literature search tutorials created using screen capture software*. Program: Electronic Library & Information Systems, Vol. 39, No. 2, pp. 112-121, 2005.

[10] G. Cruz, and R. S. Smirnov, *Adaptive entropy encoding/decoding for screen capture content*, US Patent, No. US7016547, 2006.

[11] https://msdn.microsoft.com/ko-kr/library/windows/desktop/ms684879(v=vs.85).aspx : Process Memory Usage Information, Nov. 2017.

[12] DLL_injection",http://en.wikipedia.org/ wiki ,"Hooks Overview", http://msdn.microsoft .com/en-us/library, Dec. 2017.

[13] *Camtasia_Studio*, http://en.wikipedia.org/wiki/Camtasia_Studio, Dec. 2017.

[14] *Diagram showing different applications using FFmpeg*, http://en.wikipedia.org/wiki/ffmpeg #Multimedia_frameworks_using_FFmpeg, Nov. 2017.

[15] *Diagram showing different applications using FFmpeg*, http://en.wikipedia.org/wiki/ffmpeg #Multimedia_frameworks_using_FFmpeg, Nov. 2017.

화면 이미지 전송을 위한 시스템 자원 사용량 비교

권오성

공주교육대학교 컴퓨터교육과

요 약

주기적으로 컴퓨터 화면 이미지를 생성하고 전송하는 기능은 원격 제어, 화상 회의, 화면 녹화, 원격지 타임랩스 동영상 변환 기술 등의 응용에 자주 사용된다. 이러한 응용은 백그라운드 방식의 실행이므로 CPU 시간과 메모리 요구량을 최소한으로 유지할 것을 요구한다. 사용자의 일반적인 PC 작업에 영향을 최소화하도록 알고리즘이 고안되어야 한다. 본 논문에서는 이러한 요구를 효과적으로 대응하기 위한 다양한 성능 실험을 진행하였다. 우선, 이미지 생성과 전송 기능 모듈을 각각 작성한 뒤에 이를 하나로 묶어 통합 구현하는 방식과 서로 별개의 두 개의 프로그램으로 만든 뒤 이를 실행하는 방식이다. 본 논문에서는 이 두 가지 방식에 대하여 CPU 사용량과 메모리 요구량을 측정하였다. 실험 결과, 두 기능을 통합하여 운영하는 방식의 경우가 분리 방식보다 CPU 시간 요구량이 상대적으로 높았으며 메모리 사용은 20.9 % 정도 적은 특성을 보였다. 반면에 분리 방식의 경우는 통합 방식과 CPU 사용량은 8.09 % 정도 감소하는 정도의 차이를 보였고 통합방식에서 보이는 급격한 CPU 사용량 증가는 발견되지 않았다. 설치 현장의 CPU와 메모리 상황을 보고 선택적으로 두 방식을 실행시키는 방법도 대안이 될 것으로 기대한다.

감사의 글

본 연구는 공주교육대학교 2017년 교·연·학 연구영역 지원 결과물임

Oh-Sung Kwon received the Ph.D. degree in the Department of Computer Engineering from Chung-Ang University in 1994. He has been a professor in the Department of Computer Education at Gongju National University of Education since 1995. His current research interests include multimedia data processing and digital image processing.

E-mail address: oskwon@gjue.ac.kr