임베디드 시스템 관리 프레임워크의 구현*

가진호[0] 김재훈
아주대학교 정보통신전문대학원 정보통신공학부
{signifie, jaikim}@madang.ajou.ac.kr

Implementation of Embedded-System Management Framework

Jin-Ho Ka[0] Jai-Hoon Kim
Graduate School of Information and Communication, Ajou University

요 약

기존의 컴퓨터 시스템보다 한정된 자원을 갖는 임베디드 시스템은 사용이 늘어나고 있지만 효율적인 관리 시스템이 부족하였다. 본 논문에서는 정보가전기기, PDA, 휴대폰등과 같은 인터넷으로 연결된 임베디드 시스템으로 구성된 분산 시스템 환경을 가정하고 임베디드 시스템의 효율적인 관리를 위한 관리 시스템 프레임워크를 제안하고 이를 구현한다. 제안하는 임베디드 시스템 관리 프레임워크는 이의 응용을 통해 임베디드 시스템의 상태를 모니터링하고 시스템의 성능저하, 작동정지를 사전에 방지하는 소프트웨어 재활 기법을 사용하여 임베디드 시스템의 관리 비용을 절감하고 성능을 향상시킬 수 있게 된다.

1. 서론

산업용 시스템의 용도로 주로 사용되어진 임베디드 컴퓨터 시스템은 이제 PC와 더불어 생활 속에서 쉽게 접할 수 있게 되었다. 이러한 생활속의 임베디드 시스템들은 인터넷으로 연결되어 정보 가전기기, 휴대폰, PDA, TV 셋탑 박스, 웹 패드등이 대표적이라 할 수 있다. 하지만 단순한 가전기기나 기계장치와는 달리 이러한 임베디드 시스템 응용 제품들은 내장되어진 컴퓨터 시스템의 고유한 문제점이라고 볼 수 있는 소프트웨어측면에서의 관리를 필요로 하게 된다.

PC와 같은 기존의 컴퓨터 시스템의 경우 하드웨어적인 관리외에도 소프트웨어 측면에서의 시스템 관리 방안과 관련 프로그램이 많이 개발되어져 있지만, 임베디드 시스템의 경우 기존의 컴퓨터 시스템과 다른 여러 가지 특징으로 인해 기존의 컴퓨터 시스템 관리 방안을 그대로 적용하기 힘들며 또한 임베디드 시스템 관리를 위한 효과적인 방안이 거의 개발되어지지 않았다. 기존의 컴퓨터시스템과 비교할 때 보다 한정된 컴퓨팅 자원을 갖으며 소프트웨어의 추가나 업데이트가 불편하기에 효과적인 관리 소프트웨어가 설치 시에 탑재되어있지 않다면 임베디드 시스템의 소프트웨어적인 관리가 어렵게 된다. 따라서 본 논문에서는 Post-PC로 요약되는 각종 정보가전 및 정보단말기, 그리고 넓은 의미로 인터넷에 접속되는 임베디드 시스템에 대한 관리 시스템 프레임워크를 제안하고 이의 구현을 제안한다.

2. 관련 연구

네트워크로 연결된 분산시스템 상에서 시스템을 관리하기 위해서는 먼저 관리 대상 시스템의 상태를 파악하는 시스템의 모니터링이 필요하다.

네트워크 연결을 이용해서 모니터링이 이루어지는 온라인 모니터링은 모니터링 대상 시스템의 외부에 위치하여 정상 작동하는 어플리케이션, 시스템의 상태를 지속적으로 감시한다. 모니터링을 이용해서 최종적으로는 robustness, security, fault-tolerance, adaptability를 증가시키는 것을 목표로 하며 모니터링하는 시스템의 정보는 크게 hardware-level, process-level, application-level로 구분되어진다. 현재 모니터링을 수행하는 여러 프로그램이 개발되어 사용되어지고 있으며[1], 이러한 모니터링 프로그램들은 주로 네트워크 장비의 상태를 모니터링하기 위해 제안된 SNMP 프로토콜을 이용하는 모니터링 프로그램인 MRTG[2], 클러스터 시스템에서 각 노드들의 상태를 감시하는 bWatch[3], 그리고 일반 시스템의 상태를 감시하는 BigBrother[4], MON[5] 등이 대표적이다.

모니터링 결과를 이용해서 시스템의 성능과 가용성을 향상시키기 위한 기법으로 사용되어지는 소프트웨어 재활은 시스템이나 어플리케이션이 작동 정지나 심각한 성능저하에 이르기 전에 이를 방지하는 방법으로 현재 활발히 연구가 진행되어지고 있다[6]. 소프트웨어 재활 기법을 이용해서 시스템을 관리하는 대표적인 예로써는 현재 IBM에서 미국의 Duke 대학교와의 공동 연구로 eServer xSeries(Netfinity) 서버에 온라인 모니터링과 소프트웨어 재활 기술이 적용된 서버 모니터링 프로그램인 Director가 개발되어 상용으로 출시가 되어 있으며 보다 발전된 eLiza 서버에의 적용을 위해 대대적으로 연구가 진행되어지고 있다[7,8].

* 본 연구는 한국과학재단 목적기초연구(2001-1-30300-016-2) 지원으로 수행되었음.

3. 관리 시스템의 구조 및 기능

3.1 프레임워크 구조

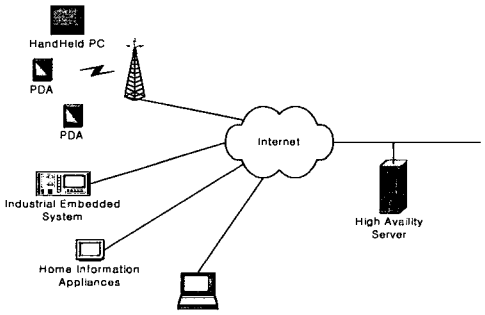

그림 1. 관리 시스템 프레임워크의 구성

본 논문에서는 제안하는 프레임워크는 그림 1과 같이 인터넷으로 연결된 임베디드 시스템과 이를 관리하는 중앙관리서버를 구성되는 시스템 구성을 가정하고 (1) 인터넷 접속 기능이 있는 임베디드 시스템 상태를 진단한 후, 인터넷을 통하여 원격 관리 서버에서 집중 감시를 하고, (2) 임베디드 시스템에 문제 발생 전에 이를 예방하거나 문제 발생 시 이를 해결할 수 있도록 하며, (3) 인터넷을 통해 임베디드 시스템의 소프트웨어를 업데이트하거나 수정하는 기능과 데이터의 백업을 중앙 관리 서버에서 지원하고, (4) 이들 임베디드 시스템을 지원하는 중앙의 관리 서버들은 다수의 임베디드 시스템과 사용자가 접속하게 되므로 고가용도 서버 구축으로 구현하는 것을 목표로 한다.

임베디드 시스템은 임베디드 리눅스를 탑재하고 있는 정보가전기기, PDA 또는 PC라고 가정하고 임베디드 시스템에 많이 사용되는 StrongARM 프로세서를 탑재한 임베디드 시스템 개발 보드인 Tynux Box와 펜티엄 3 시스템을 사용하였고, 중앙 관리 서버로는 차후에 고가용성 리눅스를 탑재한 서버를 목표로 현재는 리눅스를 탑재한 Pentium III 시스템에서 개발하였다.

임베디드 시스템에 위치하는 임베디드 관리 프로그램은 C 언어로 작성되었으며, 중앙 관리 서버에 위치하는 관리 서버 어플리케이션과 이를 조작하는 사용자 어플리케이션은 JAVA로 구현하였다.

3.2. 임베디드 시스템 모니터링

제안하는 프레임워크에서의 모니터링은 인터넷 연결을 통해 이루어지는 온라인 모니터링으로써 관리 대상이 되는 임베디드 시스템을 중앙의 관리서버에서 모니터링하는 방식으로 이루어진다. 관리 대상이 되는 임베디드 시스템에 탑재되는 임베디드 에이전트에서는 중앙 관리 서버의 요청을 받아 시스템의 상태를 파악하여 이를 관리서버에게 전송하게 된다.

임베디드 시스템은 한정된 컴퓨팅 자원을 갖으므로 임베디드 시스템에 위치하는 모니터링 프로그램의 동작이 모니터링 결과 자체에 영향을 줄 수 있다. 따라서 임베디드 에이전트는 최소한의 자원을 사용하는 형태로 구현되어지며 기본적인 시스템 감시 항목인 CPU, 메모리, 디스크, 네트워크 사용율 항목을 처리하고 필요한 경우 어플리케이션에 의존적인 감시 항목을 추가하는 방향으로 구현된다.

3.3 임베디드 시스템의 소프트웨어 재활

컴퓨터 시스템의 동작중에 발생할 수 있는 작동 정지, 성능 저하 현상은 하드웨어적인 측면을 제외하고 소프트웨어 개발중에 해결될 수 있는 소프트웨어 버그 외에도 소프트웨어 노화 현상[9]에 의해 발생하게 된다. 소프트웨어 노화 현상은 주로 불필요한 메모리의 누적, 메모리 누출, 파일 잠김의 누적, 데이터 변질, 저장 공간의 조각화(fragmentation), round off error의 누적이 발생하게 되고 이러한 원인으로 인해 서서히 성능이 저하되게 되거나 시스템이나 어플리케이션의 작동정지에 이르게 된다.

이러한 소프트웨어 노화 현상을 효과적으로 방지할 수 있는 기법으로 소프트웨어 재활[6,10]은 시스템이나 어플리케이션의 소프트웨어 노화 현상을 감시하여 일정 수준이상으로 노화 현상이 진행되면 시스템 전체, 또는 해당 어플리케이션만을 재시작하는 방식으로 구현된다. 구현하는 관리 시스템에서는 모니터링을 통해 임베디드 시스템의 자원 사용율이 미리 설정된 값 이상일 경우 시스템을 재시작하거나 설정된 일정 주기마다 자동으로 시스템을 재시작하는 방식으로 구현된다.

3.4 관리 시스템 구현

본 논문에서는 그림 2와 같은 모델을 설계하여 (1) 임베디드 시스템의 애플리케이션 형태로 존재하며 자원 감시와 관리 기능을 수행하는 임베디드 에이전트, (2)임베디드 시스템의 정보를 관리하는 중앙 관리 서버, 그리고 (3)관리자의 입력을 처리하는 관리 콘솔로 구성된다.

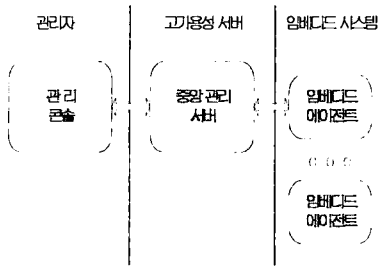

그림 2. 관리 시스템 프레임워크의 설계.

임베디드 에이전트는 중앙 관리 서버에 저장된 관리 계획에 의해 주기적으로 모니터링 요청을 받게 되면 /proc 파일 시스템을 이용해서 시스템의 상태를 파악해서 이 결과를 관리 서버로 전송한다. 또한 관리 서버로부터 소프트웨어 재활 메시지를 수신 받을 경우 시스템을 재시작해서 시스템을 초기화한다.

그림 3에서와 같이 중앙 관리 서버는 임베디드 에이전트와 함께 설정된 관리 계획에 따라 주기적으로 임베디드 에이전트를 통해 모니터링을 실시하고, 설정된

계획에 따라 소프트웨어 재활을 실행한다. 이러한 관리 계획과 모니터링, 소프트웨어 재활 결과 기록은 관리 서버에 XML의 형태로 기록된다.

중앙 관리 서버의 XML 파일에 저장되는 정보는 관리 대상이 되는 임베디드 시스템의 정보, 시스템 모니터링의 주기, 소프트웨어 재활의 방법으로 구성되는 관리 계획 부분과 임베디드 시스템의 모니터링 결과와 소프트웨어 재활의 결과가 기록되는 부분으로 구분된다. 기록되는 정보를 XML을 사용함으로써 향후 프로그램 변경시에 편리함을 줄수있으며 다른 어플리케이션에서의 자료 사용 시에 형식 변환을 쉽게 할 수 있는 장점을 갖게된다.

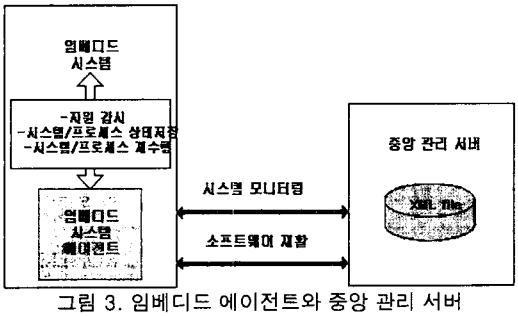

그림 3. 임베디드 에이전트와 중앙 관리 서버

그림 4과 같이 관리 콘솔은 사용자의 입력을 통해 관리 대상 시스템의 추가/변경, 모니터링 주기, 소프트웨어 재활 시기의 설정과 같은 관리 계획을 수립하고 임베디드 시스템의 현재 상태를 쉽게 파악할 수 있도록 모니터링 및 소프트웨어 재활 기록을 사용자에게 디스플레이한다.

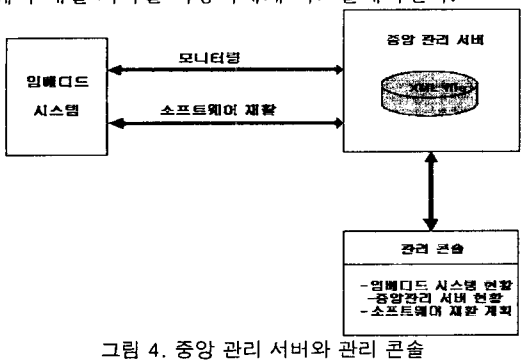

그림 4. 중앙 관리 서버와 관리 콘솔

관리 콘솔은 중앙 관리 서버에 위치하거나 원격지의 컴퓨터에 위치하여 사용자의 입력을 처리한다. 관리 콘솔을 통해 임베디드 시스템의 현재 상태, 과거 모니터링 기록을 열람할 수 있고, 임베디드 시스템의 소프트웨어 재활 계획을 결정한다. 소프트웨어 재활은 임베디드 시스템의 모니터링 결과를 통해 CPU, 메모리 사용율이 일정 수준이 이상일 경우 즉시 시스템을 재시작시키거나 일정한 주기마다 특정 시간에 시스템을 재시작시키는 방법중의 하나를 선택하여 설정한다.

4. 결론 및 향후과제

제안하는 임베디드 시스템 프레임워크를 통해 네트워크에 연결되어 사용되는 정보가전 단말기, 이동 단말기, 인터넷 장비 등으로 대표되는 임베디드 시스템을 진단하고 중앙에서 집중 감시하고 필요한 경우 관리 기능을 통해 시스템의 가용성을 향상시키며 임베디드 시스템의 성능 개선이나 기능의 변경 및 추가가 필요한 경우 네트워크를 이용한 소프트웨어 업데이트 기능을 통해 전체적인 임베디드 시스템의 관리비용을 낮추는 것을 목표로 한다.

본 논문에서는 다양한 임베디드 시스템에서 적용시킬 수 있는 관리 방안을 제시함으로써 임베디드 시스템 설계 시에 또는 임베디드 시스템 운영체제, 어플리케이션 개발시에 시스템 관리를 고려한 관련 기능의 설계, 구현이 가능하도록 하며, 제시하는 관리 방안과 구현을 이용하여 직접적인 적용이나 개선 과정을 거쳐 중앙 집중식의 임베디드 시스템 관리 시스템의 개발을 쉽게 할 수 있다.

현재 임베디드 시스템의 모니터링과 이를 이용한 임베디드 시스템의 기본적인 소프트웨어 재활 기법이 구현되어 있으며 차후에는 다양한 소프트웨어 재활 알고리즘의 적용과 인터넷 연결을 이용한 소프트웨어 자동 업데이트 기능을 추가시킬 계획이다.

참고 문헌

[1] Linux system monitoring program links, http://linux-directory.com/links/pages/System/Monitoring/

[2] MRTG website, http://people.ee.ethz.ch/~oetiker/webtools/mrtg/

[3] bWatch website, http://www.sci.usq.edu.au/staff/jacek/bWatch/

[4] MON website, http://www.kernel.org/software/mon/

[5] Big Brother website, http://bb4.com/

[6] K. S. Trivedi, K. Vaidyanathan and K. Goseva-Popstojanova, "Modeling and Analysis of Software Aging and Rejuvenation," IEEE Annual Simulation Symposium, April 2000.

[7] V. Castelli, R. E. Harper, P. Heidelberger, S. W. Hunter, K. S. Trivedi, K. Vaidyanathan and W. P. Zeggert, "Proactive management of software aging", IBM J. REs & DEV., Volume 45, No. 2, March 2001, pp. 311-332

[8] "IBM Director Software Rejuvenation", IBM xSeries and Netfinity Technical Information white paper, Jan. 2001

[9] S. Grag, A. Moorsel, K. Vaidyanathan and K. S. Trivedi, "A Methodology for Detection and Estimation of Software Aging", International Symposium on Software Reliability Engineering, ISSRE 1998, November 1998.

[10] S. Garg, A. Puliafito, M. Telek and K. S. Trivedi, "On the Analysis of Software Rejuvenation Policies", Annual Conference on Computer Assurance, June 1997.