사례 발표

실시간 운영체제(Velos) 개발

홍 성 수*

● 목 차 ●

1. 서 론

새로운 밀레니엄을 맞게 되어 전 세계가 소란스러웠던 것이 벌써 2년 전의 일이다. 일반적인 기준으로 볼 때 이 기간은 결코 길지 않은 시간이었지만, 최근 급격한 분화를 겪고 있는 정보기술 분야를 놓고 볼 때 이는 실로 엄청난 변화를 초래한 긴 기간이었다고 평가할 수 있다. 무엇보다도 지난 90년대 중반 이후부터 가속화된 기술 통합의 결과로 정보산업의 양상은 이제는 개인의 상상력의 한계를 넘어 서고 있다. 여기에서 기술의 통합이란 과거에는 서로 관련 없이 존재했던 컴퓨터기술, 가전기술, 통신기술의 대통합을 의미한다. 혹자는 현재의 정보산업 분야의 이런 양상을 생물계의 종의 분화로 묘사하기도 한다. 즉, 정보산업 기술자들과 시장을 주도하는 선구자들은 거의 매일 새로운 형태의 정보기기들을 개발하고, 이런 기기들이 소비자들에 의해 선택되어 발전하고 진화하기를 기다린다는 것이다. 물론 많은 기기들이 등장과 함께 짧은 시간 안에 사라지기도 하지만 전체적으로 볼 때 이렇게 출현한 기기들이 전통적인 정보산업의 축을 흔들게 되었으며 소프트웨어와 하드웨어적인 측면 모두에서 전통적인 정보기술의 중심을 바꾸어 놓았다. 이미 널리 알려진 바와 같이 이런 새로운 기기들은 총칭하여 내장형 기기라고 한다.

이 내장형 기기들은 이미 그 수효에서 고전적인 컴퓨팅 기기를 압도하고 있다. 현재 세계적으로 판매되고 있는 마이크로프로세서의 90%가 이런 내장형 기기에서 사용되고 있다. 이런 기기들은 과거와는 달리 단순한 제어 기능만을 제공하는 것이 아니다. 유무선 통신망을 통해 인터넷에 연결되며, 지능성과 적응성을 제공한다. 이에 따라 이들은 데스크탑 컴퓨터에 버금가는 기능성을 요구하고 있으며 이를 제공할 수 있는 실시간 운영체제를 절실히 필요로하고 있다[5].

실시간 운영체제 시장의 확대는 위에서 언급된 것처럼 내장형 기기에서 사용되는 마이크로프로세서 시장의 확대에서도 쉽게 예견될 수 있다. 미국의 Venture Development 사[11]의 예측에 의하면 실시간 운영체제 시장의 규모는 2000년에 이미 11억1천만 달러에 이르렀으며 앞으로 3년 뒤에는 시장규모가 두 배 이상 급증하여 26억2천만 달러에 이를 것이라고 한다. 물론 이 수치는 하드웨어 개발사들이 내부적으로 개발하는 인하우스 커널

* 서울대학교 전기컴퓨터공학부 교수

(inhouse kernel)의 시장 규모를 포함하고 있지 않다.

실시간 운영체제 시장의 확대는 정보기술 분야에 있는 사람이면 누구나 예측할 수 있는 현상이므로 국내 기업들이 이 시장에 뛰어 든다는 것은 매우 긍정적인 시도라고 볼 수 있다. 그러나 국가적으로 볼 때, 실시간 운영체제 기술의 보유는 단순히 시장적 확대에 대응한다는 상업적 논리 이상의 의미를 갖는다. 실제로 실시간 운영체제가 사용되는 기술 분야를 생각해 봄으로써 이를 쉽게 인식할 수 있다. 현재 실시간 운영체제는 군사기기 분야, 항공관제 분야, 육상교통기기 분야, 통신기기 분야 등 국가적 입장에서 볼 때 매우 민감하고 기술 집약적인 분야에서 널리 사용된다. 이런 분야의 소프트웨어에서 가장 핵심적인 요소인 실시간 운영체제 기술을 국가적으로 보유한다는 것은 매우 중요한 사항인 것이다. 실제로 내장형 시스템 기술이 앞선 국가들에는 예외 없이 실시간 운영체제 개발사들이 존재하며 국가적으로나 민간기업적 차원에서 이들을 지원하고 있다. 대표적인 예로는 스웨덴의 OSE Systems [12], 캐나다의 QSSL [13], 미국의 Wind River Systems [14] 등을 들 수 있다. 이에 비해 전 세계적으로 13위의 경제력을 보유하고 있고 내장형 기기 분야에서 전세계 시장을 선도하고 있는 한국은 아직 성공적으로 상업화된 자체 커널이 보유하고 있지 못한 실정이다.

이와 같은 상황을 인식하고 필자는 1995년 서울대학교 전기공학부에 실시간 운영체제 연구실[8]을 설립하고 지금까지 6년 여 동안 실시간 운영체제 기술을 개발하여 왔다. 아울러 국내외의 산업계에서 널리 사용될 수 있는 실시간 운영체제를 개발하는 것을 목표로 Arx 라 명명된 실시간 운영체제와 Velos라고 명명된 실시간 운영체제를 개발하였다. 본 고에서는 이중 현재 상업화 단계에 있는 Velos 실시간 운영체제의 개발과정에 대해서 살펴보고자 한다. Velos를 개발하는 과정에는 많은 기술적 의사결정이 필요하였으며, 그 중에는 또한 많은 기술적 시행착오가 존재하였다. 본 고에서는 이런 개발과정의 시행착오들을 공유하며 아울러 Velos의 기술적 사항들을 소개하고자 한다.

2. Velos의 개발과정

2.1 Velos 개발의 소사

Velos는 1999년 12월부터 서울대학교 전기공학부의 실시간 운영체제 연구실에서 개발되기 시작하였다. 그 후 1년 여 뒤인 2001년 1월에 윈도우 매니저와 TCP/IP 프로토콜 스택을 포함한 초기 버전이 완성되었다. Velos는 1996년부터 동 연구실에서 개발해온 실시간 운영체제 Arx [9, 4, 3, 2]를 그 모태로 하고 있다. Arx는 실시간 시스템 프로그래밍을 지원할 수 있도록 설계된 운영체제로 MMU를 이용하여 사용자와 커널의 메모리 영역을 구분하는 보호 모드형 실시간 운영체제이다. 이런 방식의 운영체제는 신뢰성의 측면에서 장점을 가지고 있지만, 마이크로프로세서 성능과 메모리 크기 등에서 많은 제한을 받는 내장형 기기에는 적합하지 못한 면이 있다. 이런 문제점들을 극복하고 내장형 기기에 꼭 필요한 기능으로 최적화된 커널을 개발하고자, 동 연구실에서는 1998년 12월부터 Arx를 변형하는 작업을 시작하였다. 그 결과로 1999년 10월에 완성된 것이 mArx (micro Arx) 실시간 운영체제[9]이다. mArx는 실시간 운영체제로서 Arx가 가졌던 장점을 극대화 시킨 반면, 지나치게 낙관적으로 시도되었던 실험적인 요소들이 제거된 매우 실용적인 커널로 설계되었다. 이에 따라 mArx는 운영체제가 지원하는 고정밀도 시간 이벤트 처리, 실시간 제약을 표현하는 API 확장, 인터럽트 서비스 루틴의 손쉬운 확장을 위한 ISR chaining, 우선순위 역전을 최소화할 수 있는 인터럽트 서비스 쓰레드 등의 기능들을 Arx로부터 물려 받은 반면, 그 유용성과 편리성에서는 긍정적이나 실제로는 지나치게 큰 지연을 초래하는 커널 이벤트 업콜 [1, 2], 사용

자 레벨 디바이스 드라이버[4] 등 현실적으로 오버헤드가 큰 기능들을 배제시켰다. 이때 mArx는 Hitachi 사의 SH3, 인텔 사의 내장형 마이크로프로세서인 80386EX, 영국 ARM 사의 ARM7TDMI를 지원하였다.

이와 같이 개발된 mArx는 1999년 중반 이미 안정적으로 수행이 되고 있었으나 초기 설계 과정에서의 시행착오의 배제와 성능개선을 위해 동 연구실에서는 1999년 12월부터 전면적인 코드 재작성을 시작하였으며, 2001년 1월에 현재와 같은 형태로 완성시켰다. 2001년은 Velos의 입장에서 세 가지 큰 전기가 마련된 한 해였다. 첫째로, Velos 실시간 운영체제 기술이 한국MDS 사[10]로 이전되어 본격적인 상품화의 길이 열리게 된 것이다. 둘째로, Velos를 탑재한 첫번째와 두번째 참조 제품인 무선통신 전용 PDA가 개발된 것이다. 셋째로는, 이 실시간 운영체제의 명칭이 mArx라는 다소 어려운 이름에서 Velos로 변경된 것이다. Velos는 "the Velocity of Light Operating System"이라는 문구에서 유래되었으며, 경량화를 통한 초고속 실시간 운영체제를 지향한다는 목적을 내포하고 있다. 현재 Velos는 주요 개발이 완료된 베타 배포 상태에 있으며 올해 4월 정식 배포를 앞두고 있다.

2.2 Velos 개발의 기술적 목적

앞에서 언급한 것처럼 Velos는 마이크로프로세서의 성능이나 메모리의 크기 등 하드웨어 자원의 측면에서 많은 제약을 갖는 내장형 시스템을 효율적으로 설계할 수 있는 런-타임 플랫폼을 추구하도록 개발되었다. 이를 위해 Velos는 다음과 같은 기술적 목적들을 가지고 개발되었다.

□ 경량성

적은 양의 메모리(램과 플래시 롬 포함)에서 동작할 수 있도록 커널과 프로토콜 스택, 윈도우 매니저를 포함하여 500KB 이하의 메모리 양의 갖도록 하였다. 그러나 이를 위해 실시간 운영체제의 필수적인 기능까지도 생략하는 단순한 경량화는 지양하였다.

□ 고성능

이를 위해서는 커널 서비스가 매우 적은 서비스 지연을 초래하여야 하는 동시에 서비스 throughput이 커야 한다. 이를 위해 인터럽트 지연과 스케줄링 지연을 최소화하고 메모리 관리 등의 커널 기능을 최적화하여야 했다.

□ 보편적 마이크로프로세서의 지원

내장형 시스템 분야에서 가장 보편적으로 사용되는 마이크로프로세서를 지원하는 것을 목적으로 하였다. Velos 개발 시점에서는 영국 ARM 사의 ARM7TDMI 마이크로프로세서가 내장형 기기 분야에서 가장 널리 사용되고 있어 이를 지원하였다. 실제로 CDMA 단말기를 개발하는데 필수적인 퀄콤 사의 MSM 칩의 내부에는 ARM7 마이크로프로세서의 코어가 장착되어 있다. 그러나 ARM7TDMI는 MMU를 제공하지 않으며 40MHz 이하의 동작 속도를 갖고 있기 때문에 (실제로 전력 소모를 줄이기 위해 20MHz 이하로 동작하는 경우가 더 일반적이다.) 실시간 운영체제 차원에서 성능을 유지하기 위한 노력은 필수적이다.

□ 유무선 망을 통한 인터넷 연결

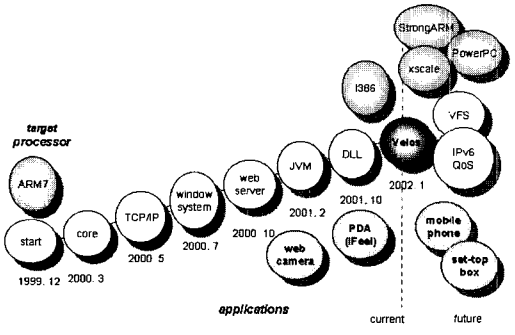
내장형 기기가 유비쿼터스 컴퓨팅(ubiquitous computing)의 핵심적인 요소로 자리 잡게 되므로 유무선 망을 통해 인터넷에 연결되는 것은 필수적이다. 이를 위해서 실시간 운영체제는 TCP/IP 프로토콜 스택을 포함한 다양한 통신 프로토콜에 대한 구현을 제공하여야 하며, 시리얼 통신, USB, IEEE 1394 등의 다양한 통신 인터페이스에 대한 장치 드라이버를 제공하여야 한다. 특히 무선 장치 드라이버인 경우 실시간 지원이 필수적이다.

□ 실시간 기능 지원

내장형 기기의 설계 과정에서 많은 경우 실시간 기능 지원이 간과되어 왔다. 그 대표적인 경우가 마이크로소프트 사의 Windows CE 운영체제[15]이다. Windows CE 운영체제는 내장형 기기용 운영체제로서 설계되고 발표되었지만 Windows CE 2.X까지는 부실한 실시간 기능으로 인해 시장에서 외면을 받아 왔다. 그 후 Windows CE 3.0에 이르러 우선순위에 따른 인터럽트 중첩의 지원, 256개의 선점형 우선순위 제공, 인터럽트 지연의 최소화 등의 실시간 기능을 넣은 뒤에야 실시간 내장형 운영체제로 인정받고 있다. 한편 Velos 개발을 위해서는 이런 실시간성의 기능적 측면 뿐만 아니라 실시간 파라미터의 기술과 같은 선언적 요소를 운영체제가 직접적으로 지원할 수 있도록 하는 것을 목적으로 하였다.

2.3 Velos 개발의 진행 과정

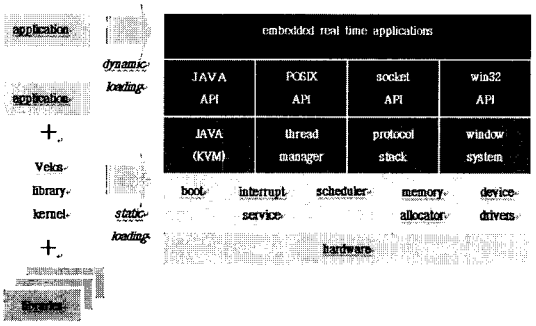
1999년 12월부터 Velos 개발이 진행된 이래 실시간 운영체제를 구성하는데 필요한 여러 요소들을 순차적으로 개발되어 왔다. 아래의 그림은 이런 개발과정의 전개를 보여주고 있다. 이 과정을 보면 Velos 실시간 운영체제는 먼저 커널 코어, TCP/IP 프로토콜 스택, 윈도우 매니저, 웹 서버, 자바 가상기계, 동적 바인딩을 지원하는 점진적 로더의 순으로 개발이 진행되었다.



3. Velos의 기능과 구조

Velos는 내장형 응용 소프트웨어를 위한 실시간 운영체제이다. 내장형 시스템은 CPU 성능, 메모리의 크기, 소비전력, 입출력장치 등 다양한 시스템 자원의 측면에서 제약을 가지고 있기 때문에 개발되는 시스템 마다 목적에 맞는 다양한 구성을 요구한다. Velos는 이러한 내장형 시스템을 구성하는데 있어 최적의 성능을 보일 수 있도록 설계되었다. Velos의 주요 특징은 다음과 같다.

- 라이브러리 커널
- POSIX 1003.13 PSE51 minimal real-time system profile[7] 제공
- 다중 쓰레딩(multi-threading)
- 실시간 스케줄링 지원
- 동적 프로그램 로딩 지원
- TCP/IP 네트워크 프로토콜 스택 지원
- Microdwindows 시스템 지원
- KVM 지원



위 그림은 Velos 커널 구조를 보여준다. Velos는 라이브러리 커널이므로 응용 프로그램과 각종 라이브러리가 커널과 함께 링크되어 하나의 이미지로 구성되며 메모리에 적재(load)된다. 이러한 라이브러리 커널을 사용하면 개발자들이 컴파일 과정에서 각 서브 시스템을 취사 선택하여 필요한 서브 시스템 만으로 전체 시스템을 구성할 수 있도록 할

수 있다. 이와는 별도로 Velos는 늦은 바인딩(late binding) 기법을 통해 동적으로 프로그램 모듈을 추가하는 기능을 제공하고 있어 실행 중에 응용 프로그램을 추가하거나 커널을 동적으로 확장할 수 있게 한다.

3.1 Velos의 특성

위의 그림에 따른 Velos 커널 구조와 부분별 특징은 다음과 같다.

□ 다중 쓰레딩
* POSIX 쓰레드와 실시간 쓰레드 지원
* Pthread 인터페이스 [6] 제공
* 256 레벨의 우선순위 지원
* 주기 쓰레드 생성 및 시간적 동기화 인터페이스 제공
* 쓰레드의 SUSPEND 상태 지원

□ 스케줄링
* 고정 우선순위 선점형 스케줄링 지원
* 동일 우선순위 내 FIFO와 round-robin 스케줄링 지원
* Round robin의 time quantum 조절 기능
* 주기 쓰레드 스케줄링과 종료시한 miss 핸들러 제공

□ 인터럽트
* 사용자 인터럽트 핸들러 등록
* 인터럽트 chaining에 의한 다중 인터럽트 핸들러 지원
* 쓰레드에 의한 인터럽트 핸들링 지원

□ 네트워크 시스템
* Linux 네트워크 프로토콜 스택
* 버클리 소켓 인터페이스 제공
* 쓰레드에 의한 프로토콜 핸들러 동작
* Microwindows 기반 윈도운 시스템

3.2 Velos의 개발의 교훈

앞에서 설명한 Velos의 구조는 Arx와 mArx라는 두 실시간 운영체제를 개발해 오면서 본 연구실에서 겪었던 6년 여의 실패와 성공의 결과라고 할 수 있다. 3.1절에서 소개된 Velos의 모습이 본 연구실에서 얻은 작은 성공의 결과라면, 이 절에서 기술하고자 하는 내용은 그 실패의 기술이라고 할 수 있다. 이미 두 가지 운영체제의 개발의 통해 얻은 이와 같은 시행착오의 경험은 성공의 경험과 함께 값진 교훈이 될 수 있다.

□ 유용성 대 신속성

운영체제의 중요한 기능 중의 하나는 복잡한 하드웨어나 입출력장치를 추상화하여 사용자가 손쉽게 응용 프로그램을 개발할 수 있도록 하는데 있다. 이에 따라 운영체제는 블랙박스의 모습을 갖게 되고 사용자들은 운영체제가 제공하는 표준화된 인터페이스(API)를 통해서만 커널의 서비스를 받게 된다. 물론 이와 같은 환경은 일반적으로 프로그래밍의 생산성을 높이고 신뢰성 있는 코드를 작성하는데 도움을 준다. 그러나 실시간 내장형 프로그래밍에서는 때로는 블랙박스 내부에서 발생한 이벤트나 서비스에 대한 직접적인 접근을 필요로 하고, 이런 필요성이 종종 실시간 내장형 프로그래밍을 어렵게 하며 신뢰성 있는 코드의 작성을 가로 막는다. Arx 실시간 운영체제에서는 이와 같이 커널이라는 블랙박스에 접근할 수 있는 연결고리 (영어로는 종종 hook 또는 back door라고 함)를 제공하는 커널 이벤트 업콜 기능[2]을 가지고 있었다. 커널 이벤트 업콜 기능을 통해 내장형 프로그래머는 커널 속에서 발생하는 스케줄링 관련 이벤트들을 모니터링할 수 있으며 사용자 레벨 쓰레드의 스케줄링 상태를 커널로 통보해 줄 수도 있다.

이런 기능을 이용하면 Arx 실시간 운영체제 상에서 손쉽게 사용자 레벨 장치 드라이버[4]

를 구현할 수 있을 뿐만 아니라 선점적인 사용자 레벨 쓰레드[3]의 구현도 매우 용이해 진다. 그러나 커널의 연결고리를 통한 이런 기능은 많은 서비스 지연을 초래한다. 예를 들어, 사용자 레벨 장치 드라이버의 경우 하나의 인터럽트를 처리하는 지연이 수 백 microsecond를 초과하기도 하였다. 물론 마이크로프로세서의 성능이 현저히 증가된다면 이에 대한 개선의 여지는 충분이 있지만 현실적인 의미에서 볼 때 실시간 운영체제가 이를 향유하게 되기까지는 10 여년 이상의 기술적 진보가 필요할 것이다. 결과적으로 Velos는 커널을 블랙박스가 아닌 화이트박스로 제공하여 내장형 시스템 개발자가 필요로 하는 최적화된 성능을 가능하게 한다. 이를 위해서 Velos는 쓰레드에 기반한 인터럽트와 프로토콜 핸들러의 처리 기능 등을 제공한다.

□ 보편적 커널(universal kernel) 대 전용 커널

신뢰성은 실시간 운영체제가 필수적으로 유지하여야 하는 덕목이다. 그러나 효율성과 신뢰성이 서로 상충하게 된다면 때로는 신뢰성의 책임은 프로그래머나 개발자에게 이관되고 실시간 운영체제는 효율성을 선택하게 되는 것이 일반적인 개발 관행이다. 본 연구진은 이런 비과학적인 관행에 반대하여 효율성을 희생하지만 신뢰성을 제고할 수 있는 MMU 기반 보호 모드 커널로 Arx를 설계하였다. 아울러 이를 통하여 low-end에서부터 high-end를 포괄 할 수 있는 단일 커널을 개발하고자 하였다. 그러나 실제에는 많은 실시간 내장형 기기에서는 효율성을 희생한다는 전제 조건이 성립하지 않으며 신뢰성과 효율성을 동시에 만족시킨다는 것은 모순적 결과가 되는 경우가 많다. 이런 시행착오가 결국은 mArx와 Velos를 개발하게 되는 촉매제가 되었다.

□ 독자 API 대 표준 API

실시간 운영체제를 설계할 때 제일 먼저 고려해야 할 사항이 설계할 운영체제의 모습을 결정하는 API를 정하는 것이다. 현재 마이크로소프트 계열의 Win32 API, IEEE POSIX 계열의 API 등이 표준 인터페이스로서 경합을 벌리고 있다. 이와는 별도로 실시간 운영체제 개발사들이 자체적으로 개발한 API를 제공하고 있다. 이런 표준 API들을 분석하였을 때 얻을 수 있는 결론은 어느 하나의 표준도 실제 실시간형 내장형 프로그래밍을 위해 필요한 인터페이스를 충분히 제공하지 못한다는 것이다. 이에 따라 새로운 프로그래밍 모델에 따른 API를 설계하고자 하는 충동에 빠지기 쉽다. 본 연구실에서도 이런 유혹에 따라 독자적 API를 개발하였다가 결국은 IEEE POSIX 표준을 따르고, 기타 추가하고자 하는 인터페이스를 오버로딩하는 방식을 선택하였다.

□ 정적 링킹 대 동적 로딩

라이브러리 커널은 그 효율성과 성능에서 타의 추종을 불허한다. 그러나 라이브러리 커널은 응용 프로그램과 함께 컴파일되고 링크되므로 동적인 확장이 불가능한 결정적 단점이 있다. 이는 현재 많은 상용 라이브러리 커널이 겪고 있는 문제점이기도 하다. Velos 개발 시에는 이러한 라이브러리 커널의 단점이 간과되었다가 추후에 장소 무관 코드(position independent code) 기능을 이용하여 늦은 바인딩을 제공하여 동적 코드 로딩과 커널 확장 기능을 추가하였다.

4. 결 론

Velos는 국내의 기술만으로 시작부터 하나 하나 자체 개발된 내장형 시스템을 위한 실시간 운영체제이다. Velos는 라이브러리 커널로서 경량성, 효율성, 실시간성, 인터넷 연결성을 목적으로 설계되고

개발된 커널이다. 한편 라이브러리 커널로서의 제한점을 극복하기 위해 늦은 바인딩을 통한 동적 모듈 링킹과 로딩을 지원한다. Velos 커널의 구조는 본 연구실의 6 여년의 시행착오의 결과이며 Velos의 현재의 모습은 실시간 내장형 시스템에서 효율성은 아직도 희생될 수 없는 중요한 덕목임을 보여주고 있다.

Velos는 현재 베타 배포의 상태에 있으며 2002년 4월에 정식으로 배포될 예정이다. 이런 Velos의 등장이 내장형 기기 산업과 관련 연구 분야, 나아가서 국가 정보기술 사업의 전반에서 새로운 바람이 되기를 기대해 본다.

참고문헌

[1] T.E. Anderson, B. N. Bershad, E. D. Lazowska, and H. M. Levy. Scheduling activations: Effective kernel support for the user-level management of parallelism. In *Proceedings of the 13th ACM Symposium on Operating System Principles*, pages 95-109, October 1991.

[2] S. Hong, Y. Seo, and J. Park, ARX/ULTRA: A New Real-Time Kernel Architecture for User-Level Threads, *SNU EE Technical Report*, SNU-EE-TR-97-3, August 1997.

[3] Y. Seo, J. Park, and S. Hong, Supporting Preemptive User-Level Threads for Embedded Real-Time Systems, *SNU EE Technical Report*, SNU-EE-TR-98-1, August 1998.

[4] Y. Seo, J. Park, and S. Hong, Efficient User-Level I/O in the ARX Real-Time Operating System, In *Lecture Notes in Computer Science*, Vol. 1474. pp. 162-171, June 1998.

[5] S. Hong, Coping with Embedded Software Crisis using Real-Time Operating Systems and Embedded Middleware, Invited for presentation at *IEEE Asian Pacific ASIC (AP-ASIC) Conference*. Cheju, Korea. August 2000.

[6] Institute for Electrical and Electronic Engineers. In IEEE Std. 1003.1c-1995 POSIX Part 1: System application program interface - amendment 2: threads extension, 1995.

[7] Institute for Electrical and Electronic Engineers. In IEEE Std 1003.13-1998, IEEE Standard for Information Technology - Standardized Application Environment Profile - POSIX Realtime Application Support (AEP)

[8] Real-Time Operating Systems Lab., SoEE&CS, Seoul National University, Korea, http://redwood.snu.ac.kr

[9] Arx and mArx Real-Time Operating Systems, http://arx.snu.ac.kr

[10] Hankook MDS, Inc, http://www.hkmds.com/

[11] Venture Development Corporation, http://www.vdc-corp.com/

[12] OSE Systems, http://www.ose.com/

[13] QNX Software Systems Ltd, http://www.qnx.com/

[14] Wind River Systems, Inc, http://www.windriver.com/

[15] Microsoft Corporation, http://www.microsoft.com/windowsc

저자약력

홍 성 수

1986년 서울대학교 컴퓨터공학과 졸업 (B.S.)
1988년 서울대학교 컴퓨터공학과 졸업 (M.S.)
1988년-1989년 한국전자통신연구소 (연구원)
1994년 University of Maryland, Department of Computer Science(Ph.D.)
1994년-1995년 University of Maryland, Department of Computer Science (Faculty Research Associate)
1995년-1995년 Silicon Graphics Inc. (Member of Technical Staff)
1995년-1997년 서울대학교 전기공학부 전임강사
1997년-2001년 서울대학교 전기컴퓨터공학부 조교수
2001년-현재 서울대학교 전기컴퓨터공학부 부교수