오픈솔라리스 운영체제에서 실시간 프레임 워크 RTL의 설계 및 구현

The Design and Implementation of Real-Time Framework RTL on OpenSolaris

주민규, 이진욱, 임재석, 조문행, 이철훈
충남대학교 컴퓨터공학과

Ju min-gyu, Lee jin-wook, Lim jae-suk,
Cho moon-haeng, Lee cheol-hoon
Dept. Computer Engineering, Cungnam National Univ.

요약

로봇 기술이 발달하면서 사람의 지령에 의해 수동적, 반복적인 작업을 수행하던 기존 전통적 로봇에서 벗어나, 스스로 외부환경을 인식하고, 상황을 판단하여 자율적으로 동작하는 지능형 로봇이 등장하였다. 이러한 로봇의 S/W개발은 편의성을 위해 범용 운영체제를 사용하여 개발하는 추세이다. 지능형 서비스의 QoS(Quality of Service)를 위해서는 로봇 미들웨어에 실시간성을 지원해야 하지만 범용 운영체제는 실시간성을 지원하지 않는 문제점이 있다. 본 논문에서는 범용 운영체제인 오픈솔라리스에 실시간성을 위한 논리적 정확성 및 시간 결정성을 보장하기 위하여 실시간 스케쥴러를 포함한 실시간 프레임 워크 RTL(Real-Time Layer)을 설계 및 구현한 내용을 기술한다. 또한 성능측정을 위해 쓰레드의 응답시간을 측정하였다.

I. 서론

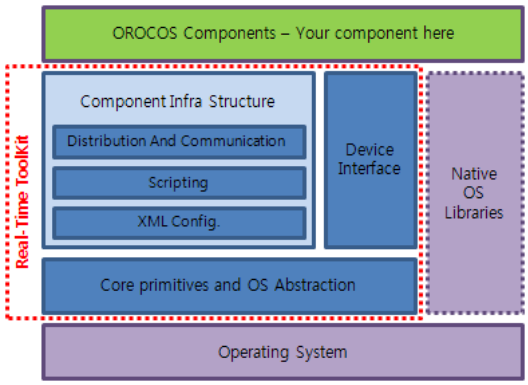
로봇 기술이 발달하면서, 산업 현장에서만 사용되었던 로봇들이 일상생활 곳곳에 사용되고 있다. 병원에서 간호사를 보조하는 로봇, 도서관에서 책을 찾아주는 로봇, 가정집에서 청소를 하는 로봇 등 다양한 로봇들이 사람과 공존하며 사용되고 있다. 이러한 로봇들은 과거 사람의 지령에 의해 수동적, 반복적 작업을 수행하던 전통적 로봇에서 벗어나 외부 환경을 인식 하고, 스스로 상황을 판단하여 자율적으로 동작하는 지능형 로봇이다[1]. 지능형 로봇은 여러 외부환경 변화에 대해 빠른 응답을 해야 하며 이를 위해선 QoS(Quality of Service)를 보장하여야 한다. QoS를 위해서는 로봇미들웨어에 실시간성 지원은 필수 불가결한 요소이지만 현재의 로봇 미들웨어들은 상용 실시간 운영체제상에서만 실시간성을 지원하고 있는 문제점이 있다. 상용 실시간 운영체제는 고가의 사용료 및 유지보수비를 지불해야 사용할 수 있으며, 이는 지능형 로봇들의 가격에 영향을 미치게 된다. 이러한 문제점을 해결 하기위해 본 논문에서는 범용 운영체제인 오픈 솔라리스(OpenSolaris)에 우선순위 기반의 실시간 스케줄링을 포함한 프레임 워크 RTL(Real-Time Layer)을 설계 및 구현하여 오픈솔라리스에 실시간성을 부여하였다. 본 논문의 구성은 2장에서는 관련연구로써 상용화된 로봇 미들웨어인 OROCOS와 MIRO 그리고 오픈 솔라리스에 대해 기술하며 3장에서는 RTL의 설계 및 구현내용을, 4장에서는 RTL의 동작에 대해 기술한다. 마지막으로 5장에서는 결론 및 향후 연구과제에 대해 기술한다.

Ⅱ. 관련연구

1. 로봇미들웨어

1.1 OROCOS

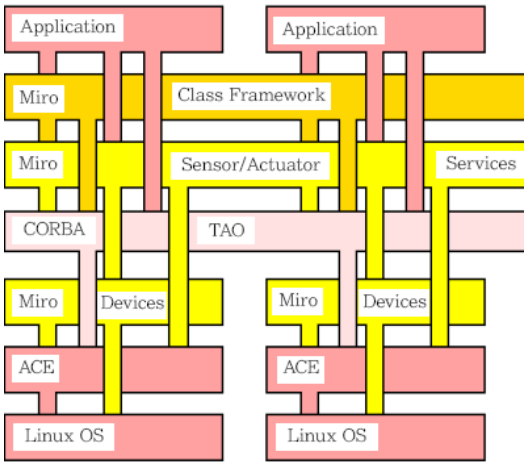
OROCOS(Open RObot COntrol Software)는 2001년 9월부터 EURON(European Robotics Research Network)의 지원으로 4개국의 대학에서 연구가 시작되었다. 로봇과 머신 제어(Machine contro)를 위한 플랫폼에 독립적인 프레임워크이며, 로봇 애플리케이션 개발을 위해 4개의 C++ 라이브러리를 제공한다. 로봇 컴포넌트 개발에 필요한 라이브러리와 로봇 컴포넌트간의 통신등과 같은 다양한 기법을 제공하는 RTT(Real-Time Toolkit)를 제공한다[2].



▶▶ 그림 1. OROCOS 아키텍쳐

1.2 MIRO

MIRO(Micro-Middleware for Mobile Robot Application)는 모바일 로봇제어를 위한 CORBA (Common Object Request Broker Architecture)기반의 분산 객체지향 프레임워크로, MIRO의 코어 컴포넌트들은 ACE(Adaptive Communication Enviorment)의 뒷받침 아래 리눅스 환경에서 C++로 개발 되었다. ACE는 동시처리방식 통신 소프트웨어의 많은 핵심 패턴들을 구현한 오픈소스 기반의 객체지향 프레임워크로, 네트워크 프로그래밍의 단점을 해결하기위해 만들어 졌다. MIRO는 TAO(The Ace ORB)를 기반으로 동작하며, TAO는 DOC(Distribution Object Computing) 그룹에서 만들어진 DRE (Distribution Real-time Embedded) 시스템을 위한 오픈소스의 분산 미들웨어 플랫폼이다. TAO는 ACE에 의해 구현된 프레임워크 컴포넌트 패턴을 사용하여 구축된 COBAR이다. MIRO 내부에는 실시간성을 지원하는 라이브러리 및 API를 제공하지 않는다. 대신, TAO를 기반으로 하기 때문에 TAO에서 제공하는 실시간성 기능을 사용하여 로봇 미들웨어에 실시간성 기능을 지원하게 된다[3].



▶▶ 그림 2. MIRO 아키텍쳐

2. 오픈 솔라리스(Open Soraris)

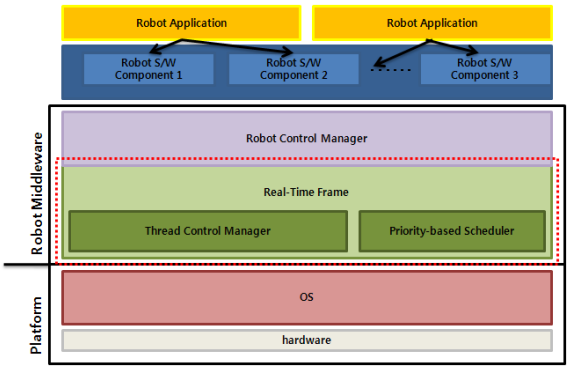
썬마이크로시스템즈(Sun Microsystems, Inc)에서 개발한 오픈 솔라리스는 리눅스커널을 이용하여 라운드로빈 스케줄러를 제공한다. 또한 솔라리스와 함께 호환이 가능한 오픈소스 개발 프로젝트이다. 오픈 솔라리스는 CDDL(Common Development and Distrivution License)의 조건 하에 재배포가 가능한 범용 OS이다. 오픈 솔라리스는 선 스팍(Sun Sparc) 하드웨어뿐만 아니라, 인텔의 X86이나 64비트 AMD 에서도 운영될 수 있으며, 운영체제로부터 응용프로그램을 분리하는 솔라리스 컨테이너와 시스템 라이브러리등을 포함하고 있다[4].

Ⅲ. RTL 설계 및 구현

1. 실시간 프레임워크 RTL의 구조

RTL(Real-Time Layer)의 스케줄러는 범용 운영체제

에서 제공하는 쓰레드 관리 함수와 POSIX 라이브러리를 사용하여, 운영체제의 쓰레드를(Thread)를 직접 실행(resume), 정지(suspend)시켜 제어하는 우선순위 기반 스케줄러를 구현함으로써 오픈 솔라리스에 실시간성을 지원하였다. [그림 3]에서와 같이 RTL은 쓰레드 제어 매니저와 우선순위 기반 스케줄러 모듈로 구성된다.



▶▶ 그림 3. RTL 구조

2. 쓰레드 제어 매니저

RTL의 쓰레드 제어매니저는 쓰레드에 대한 생성, 삭제 관리를 위한 API를 제공하며 쓰레드의 쓰레드 제어 블록(Thread Control Block)을 만들어 각 쓰레드들마다 고유의 문맥을 갖게 하였다. TCB는 RTL에서 [표 1]과 같은 각종 변수들을 포함하며, 쓰레드를 제어하는데 사용된다.

표 1. TCB의 각종 변수들

변수	기능
status	쓰레드의 상태
priority	우선순위
pThreadID	쓰레드 아이디
t_TIMESlice	타임 슬라이스
*(*warpper)(void*)	실행함수 포인터

3. 우선순위 기반 스케줄러

RTL은 쓰레드의 실행(Resume)및 정지(Suspend)와 같은 제어를 범용운영체제의 스케줄러에 의존하지 않고, 직접 제어가 가능하도록 하기 위하여 우선순위 기반 스케줄링 기법을 적용한 스케줄러를 설계 및 구현하였다. 우선순위 스케줄러는 우선순위가 가장 높은 READY상태의 쓰레드에게 CPU를 할당한다. 또한 RTL은 선점형(Preemptive) 스케줄방식이기 때문에, 실행중인 쓰레드보다 높은 우선순위를 가진 쓰레드가 Ready상태가 되면 기존 쓰레드를 정지(Suspend)시키고, 가장 높은 우선순위의 Ready상태에 있는 쓰레드를 실행(Resume)시켜 CPU제어권을 넘겨주게 된다.

표 2. 우선순위 기반 스케줄러 API

함수명	함수 기능
RO_Schedule()	우선순위 기반 스케줄러
RO_Resume()	쓰레드를 Ready상태로 전환
RO_Suspend()	쓰레드를 Suspend상태로 전환
RO_ContextSwitch()	기존 쓰레드를 중지시키고, 다른 쓰레드를 수행시킴
RO_SleepTicks()	쓰레드를 주어진 시간동안 Delayed상태로 전환
RO_InsertThreadToReadyList()	쓰레드를 Ready리스트에 추가
RO_DeleteThreadFromReadyList()	쓰레드를 Ready리스트에서 삭제

Ⅳ. 실험 결과

1. 실험 환경

본 논문에서 구현한 RTL은 오픈 솔라리스 운영체제를 기반으로 [표 3]과 같은 H/W를 사용하였다.

표 3. 실험 환경

Environment	
Operating System	OpenSolaris 2009.06
CPU	Intel Pentium4 3.2GHz
Cache	2048KB
RAM	516MB

2. 실험 방법 및 결과

논리적 정확성 및 시간결정성 보장을 위한 성능 측정을 위해 오픈 솔라리스 운영체제에 RTL을 통해 우선순

위가 다른 다수의 쓰레드를 생성하여 각각의 쓰레드가 루프를 100회 돌면서 자신의 우선순위를 화면에 출력해 주는 응용프로그램을 작성하여 오픈솔라리스 운영체제와 실시간 RTL에서의 쓰레드 응답시간을 비교 분석하였다.

RTL의 쓰레드 응답시간을 측정하기위해 3가지 경우를 가정하여 오픈 솔라리스 스케줄러와 비교분석 하였다. 우선순위가 다른 쓰레드 3개, 5개, 10개를 생성하여 각 쓰레드의 우선순위를 화면에 출력하는 응용프로그램을 작성 하였다. 쓰레드의 응답시간을 측정하기 위해 가장 높은 우선순위의 쓰레드가 활성화되어 끝나는 시간을 측정하였다. 테스트 응용프로그램에서의 쓰레드 생성 조건은 다음[표 4]와 같다.

표 4. 테스트를 위한 쓰레드 생성 조건

생성조건	
쓰레드 수	N(= 3, 5, 7)
우선순위	낮은 정수값 일수록 높은 우선순위
응답시간	쓰레드가 활성화 되는 시점부터 끝나는 시간
처음에는 N번째의 쓰레드만 유일하게 활성화 되어 실행되며, N번째의 쓰레드가 수행중간에 다른 쓰레드들을 활성화 시킨다.	

표 5. RTL과 오픈 솔라리스의 쓰레드 응답시간 (단위 : ms)

	RTL	OpenSoraris
쓰레드 3개	1.32	7.64
쓰레드 5개	1.36	17.12
쓰레드 10개	1.39	26.44

[표 5] 는 우선순위가 가장 높은 쓰레드의 응답시간을 측정한 실험 결과이다. 실험결과를 통해 기존 쓰레드가 CPU 제어권을 얻어 수행하던 도중 더 높은 우선순위를 가진 쓰레드가 활성화 되어질 경우 RTL에서의 가장 높은 우선순위의 쓰레드 응답시간이 거의 일정함을 확인할 수 있다.

[그림 5]는 RTL이 우선순위 기반으로 스케줄 되는 모습을 보여주고 있다. 쓰레드의 생성 및 실행 방법은 응답시간을 측정한 실험방법과 동일하다. 쓰레드2가 수행도중 더 높은 우선순위를 가진 쓰레드0이 활성화되면 기존 쓰레드가 CPU제어권을 넘겨주는 형태로 스케줄링 되는 모습을 보여준다.

```
########## Starting Real-Time Layer ##########
[RO_InitializeRTLayer()]
[RO_InitializeRTLayer()] Create timer for scheduler.
[Application_main()]
[RO_Start()]
22222222222222222222222222222222222222222222222222222222222222200000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000111111111111111111111111111111111111111111111111111111
11111111111111111111111111111111111111111111111111111111111111111111111111111
11111111111111111111111111111111111111111111111111111111111111122222222222222
22222222222222222222222222222222222222222222222222222222222222222222222222222
22222222222222222222222222222222222222
```

▶▶ 그림 5. RTL에서 3개의 쓰레드 스케줄링 화면

V. 결론

본 논문에서는 기존 범용 운영체제인 오픈 솔라리스상에 실시간 스케줄러를 포함한 RTL을 설계 및 구현하여 범용 운영체제에서 실시간 미지원 문제를 보완하였다. RTL은 우선순위 기반 스케줄러를 사용하여 시간결정성 및 논리적 정확성을 보장하도록 설계 하였다. 실험결과를 통해 기존의 라운드로빈 방식이 아닌 우선순위 스케줄링 방식으로 동작하는 것을 확인할 수 있었으며, 우선순위가 높은 쓰레드의 응답시간이 쓰레드의 수와 상관없이 거의 일정한 응답시간인 것을 확인할 수 있었다. 향후 연구과제로는 범용 운영체제상에서 우선순위 스케줄링 방식 이외의 EDF(Earliest Deadline First), RM(Rate Monotonic)스케줄 등 다양한 스케줄링 기법을 제공하는 방법을 연구하여 범용 운영체제상에서 로봇 미들웨어에 실시간성을 부여할 것이다.

■ 참 고 문 헌 ■

[1] www.robotics.or.kr

[2] H.Bruyninckx "Open Robot Control Software: the OROCOS project," IEEE International Conference on Robotics & Automation, Vol.3, pp.2523-2528, 2001.

[3] MiroManual Ver0.9.4 2006(1)

[4] Opensolaris_Datasheet

[5] 최찬우, 조문행, 박성종, 이철훈, "로봇 컴포넌트

에 실시간성을 지원하기 위한 프레임워크 구현 및 성능분석", 한국콘텐츠학회논문지, Vol9. No.4, pp81-94, Apr 2009