



(19) 대한민국특허청(KR)
(12) 등록특허공보(B1)

(45) 공고일자 2014년09월05일
(11) 등록번호 10-1438708
(24) 등록일자 2014년09월01일

(51) 국제특허분류(Int. Cl.)

G06F 9/46 (2006.01)

(21) 출원번호 10-2013-0025029

(22) 출원일자 2013년03월08일

심사청구일자 2013년03월08일

(56) 선행기술조사문헌

KR100790304 B1

Wellings, A. J. 'Multiprocessors and the Real-Time Specification for Java.' Proceedings of the 11th IEEE Symposium on Object Oriented Real-Time Distributed Computing. 2008, pp.255-261.

Silva, Jr. E. T. et al. 'A virtual platform for multiprocessor real-time...' Proceedings of the 6th international workshop on Java technologies for real-time and embedded systems, 2008, pp.31-37

(73) 특허권자

고려대학교 산학협력단

서울특별시 성북구 안암로 145, 고려대학교 (안암동5가)

(72) 발명자

조현중

경기 고양시 일산동구 은행마일로 100, 306동 201호 (식사동, 은행마을3단지아파트)

권진산

서울특별시 송파구 동남로11길 4 103동 305호

(74) 대리인

특허법인엠에이피에스

전체 청구항 수 : 총 7 항

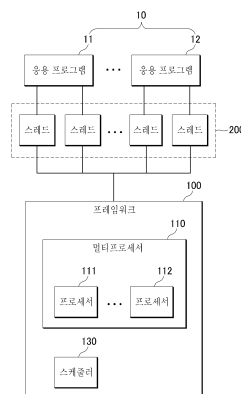
심사관 : 유진태

(54) 발명의 명칭 응용 프로그램 제어 방법

(57) 요약

멀티 프로세서(multi processor) 환경에서 자바(JAVA) 언어 기반 응용 프로그램을 실시간으로 제어하는 방법이 개시되며, 상기 응용 프로그램 제어 방법은 a) 적어도 하나 이상의 응용 프로그램이 실행되면 상기 실행된 응용 프로그램을 프레임워크에 등록하는 단계; b) 상기 응용 프로그램이 실행됨에 따라 생성되는 각 스레드의 실행 순서를 조정하고, 상기 각 스레드를 각각의 프로세서에 할당시키는 스케줄러를 설정하는 단계; c) 상기 각 스레드에 대한 작업 주기와 데드라인에 대응되는 주기 타이머와 데드라인 타이머를 각각 생성하는 단계; d) 상기 주기 타이머, 상기 데드라인 타이머 및 상기 각 스레드에 대한 작업 완료 여부에 대한 정보 중 적어도 어느 하나를 기초로 상기 b) 단계에서 설정된 스케줄러를 호출하여 상기 프레임워크가 소유한 제어권한을 상기 호출된 스케줄러에 할당하고, 상기 각 스레드에 대한 작업을 수행하는 단계; e) 상기 d) 단계에서 수행된 작업이 완료되면, 상기 호출된 스케줄러가 소유한 제어권한을 상기 프레임워크로 회수하는 단계; 및 f) 상기 a) 내지 상기 d) 단계를 통해 작업이 완료된 응용 프로그램과 상이한 응용 프로그램이 실행되면, 상기 a) 단계부터 상기 e) 단계까지 순차적으로 반복 실행하는 단계를 포함한다.

대표도 - 도1



이 발명을 지원한 국가연구개발사업

과제고유번호 1345175857

부처명 교육과학기술부

연구사업명 핵심개인연구

연구과제명 멀티코어 구조에서 Pluggable Scheduling 지원을 위한 실시간 가상 머신 연구

기 여 율 1/1

주관기관 고려대학교 산학협력단

연구기간 2012.05.01 ~ 2013.04.30

특허청구의 범위

청구항 1

멀티 프로세서(multi processor) 환경에서 스케줄링 프레임워크가 자바(JAVA) 언어 기반 응용 프로그램을 실시간으로 제어하는 방법에 있어서,

- a) 적어도 하나 이상의 응용 프로그램이 실행되면 상기 실행된 응용 프로그램을 프레임워크에 등록하는 단계;
- b) 상기 응용 프로그램이 실행됨에 따라 생성되는 각 스레드(thread)의 실행 순서를 조정하고, 상기 각 스레드를 각각의 프로세서에 할당시키는 스케줄러(scheduler)를 설정하는 단계;
- c) 상기 각 스레드에 대한 작업 주기와 데드라인에 대응되는 주기 타이머와 데드라인 타이머를 각각 생성하는 단계;
- d) 상기 주기 타이머, 상기 데드라인 타이머 및 상기 각 스레드에 대한 작업 완료 여부에 대한 정보 중 적어도 어느 하나를 기초로 상기 b) 단계에서 설정된 스케줄러를 호출하여 상기 프레임워크가 소유한 제어권한을 상기 호출된 스케줄러에 할당하고, 상기 각 스레드에 대한 작업을 수행하는 단계;
- e) 상기 d) 단계에서 수행된 작업이 완료되면, 상기 호출된 스케줄러에 할당된 제어권한을 상기 호출된 스케줄러로부터 회수하는 단계; 및
- f) 상기 a) 내지 상기 d) 단계를 통해 작업이 완료된 응용 프로그램과 상이한 응용 프로그램이 실행되면, 상기 a) 단계부터 상기 e) 단계까지 순차적으로 반복 실행하는 단계를 포함하는 응용 프로그램 제어 방법.

청구항 2

제 1 항에 있어서,

상기 b) 단계는 상기 스케줄러의 선입선출(First in First Out) 방식, 상기 각 스레드의 우선순위 및 상기 각 스레드의 프로세스 선호도 중 적어도 하나를 기초로 상기 각 스레드의 실행 순서를 조정하는 응용 프로그램 제어 방법.

청구항 3

제 1 항에 있어서,

상기 c) 단계에서 상기 주기 타이머 및 상기 데드라인 타이머는 상기 프레임워크를 통해 생성되는 것인, 응용 프로그램 제어 방법.

청구항 4

제 1 항에 있어서,

상기 c) 단계는 상기 응용 프로그램 사용자에게 의해 입력된 정보를 기초로 사용자 타이머를 생성하는 단계를 포함하는 응용 프로그램 제어 방법.

청구항 5

제 1 항에 있어서,

상기 d) 단계는 상기 호출된 스케줄러의 제어권한을 기초로 상기 스케줄러가 호출된 시점에서 작업을 수행할 소

정의 스레드를 결정하는 응용 프로그램 제어 방법.

청구항 6

제 5 항에 있어서,

상기 d) 단계는 상기 호출된 스케줄러의 제어권한을 기초로 상기 작업을 수행할 소정의 스레드가 할당될 프로세스를 결정하는 응용 프로그램 제어 방법.

청구항 7

제 1 항에 있어서,

상기 d) 단계는 호출된 스케줄러를 통해 상기 각 스레드 중 상기 각각의 프로세서에서 실행되거나 피선점되는 스레드를 결정하고, 결정된 스레드에 대한 정보를 상기 프레임워크의 라이브러리를 통해 전달하는 것인, 응용 프로그램 제어 방법.

명세서

기술분야

[0001] 본 발명은 응용 프로그램 제어 방법에 관한 것으로서, 보다 구체적으로, 멀티 프로세서 환경에서 자바 언어 기반 응용 프로그램을 실시간으로 제어하는 방법에 관한 것이다.

배경기술

[0002] 근래에 들어, 컴퓨터와 관련된 기술의 발전으로 인해 여러 개의 프로그램을 하나의 컴퓨터로 동시에 실행시키고 제어할 수 있는 멀티 프로세서 환경을 포함하는 기기가 개발되었다. 멀티 프로세서는 사용자가 동시에 하나 이상의 프로그램 모듈을 사용할 수 있도록 프레임 제공하는 프레임워크를 통해 여러 개의 프로그램을 동시에 실행시키고 제어한다.

[0003] 구체적으로, 프레임워크 내에서 각각의 프로그램의 작업을 수행하는 스레드의 실행 순서를 조정하고 스레드를 프로세서에 할당시키는 스케줄러를 통해 여러 개의 프로그램을 동시에 실행시키고 제어할 수 있다.

[0004] 이와 관련하여, 한국공개특허 제2009-0061177호(발명의 명칭: 동적 로드 밸런싱을 지원하는 멀티 쓰레딩 프레임워크 및 이를 이용한 프로세싱 방법)는 특정 어플리케이션으로부터 전달된 단위 작업들을 그 단위 작업에 포함된 단위 작업 정보에 따라 그 처리 순서를 재정의하고, 재정의된 처리 순서에 따라 단위 작업들을 스레드 풀에 전달하여 병렬 처리하는 스케줄러를 이용하여 프로세싱하는 프레임워크에 대해 개시하고 있다.

[0005] 한편, 컴퓨터 프로그래밍 언어로 완전한 객체 지향적인 자바(JAVA) 언어가 개발되어 사용되고 있다. 자바 언어는 간결한 구조와 폭넓은 표준 라이브러리를 제공하고 높은 이식성이 확보되어 있는 주요 프로그래밍 언어다. 자바 언어는 사용자가 실시간으로 기능을 제어할 수 있도록 나노초 단위의 고정밀 시간 표현 클래스, 타이머, 실시간 스케줄링을 위한 스케줄링 파라미터 클래스, 어플리케이션과 함께 사용자가 직접 작성한 스케줄러를 사용하는 RTSJ(Real-time Specification for Java) 형태로 발전되었다.

[0006] 하지만, RTSJ는 싱글 프로세서 환경에서는 동작할 수 있지만, 멀티 프로세서 환경에서 자바 언어 기반 응용 프로그램의 동작을 실시간으로 스케줄링하여 제어하는 데에는 어려움이 있었다.

발명의 내용

해결하려는 과제

[0007] 본 발명의 일부 실시예는 멀티 프로세서 환경에서 자바 언어 기반 응용 프로그램을 실시간으로 스케줄링하여 하나 이상의 응용 프로그램을 동작시키거나 제어하는 방법을 제공하는 데에 그 목적이 있다.

[0008] 다만, 본 실시예가 이루고자 하는 기술적 과제는 상기된 바와 같은 기술적 과제들로 한정되지 않으며, 또 다른

기술적 과제들이 존재할 수 있다.

과제의 해결 수단

- [0009] 상기한 기술적 과제를 달성하기 위한 기술적 수단으로서, 본 발명의 제 1 측면에 따른 멀티 프로세서 환경에서 자바 언어 기반 응용 프로그램을 실시간으로 제어하는 방법은 a) 적어도 하나 이상의 응용 프로그램이 실행되면 상기 실행된 응용 프로그램을 프레임워크에 등록하는 단계; b) 상기 응용 프로그램이 실행됨에 따라 생성되는 각 스레드의 실행 순서를 조정하고, 상기 각 스레드를 각각의 프로세서에 할당시키는 스케줄러를 설정하는 단계; c) 상기 각 스레드에 대한 작업 주기와 데드라인에 대응되는 주기 타이머와 데드라인 타이머를 각각 생성하는 단계; d) 상기 주기 타이머, 상기 데드라인 타이머 및 상기 각 스레드에 대한 작업 완료 여부에 대한 정보 중 적어도 어느 하나를 기초로 상기 b) 단계에서 설정된 스케줄러를 호출하여 상기 프레임워크가 소유한 제어 권한을 상기 호출된 스케줄러에 할당하고, 상기 각 스레드에 대한 작업을 수행하는 단계; e) 상기 d) 단계에서 수행된 작업이 완료되면, 상기 호출된 스케줄러가 소유한 제어 권한을 상기 프레임워크로 회수하는 단계; 및 f) 상기 a) 내지 상기 d) 단계를 통해 작업이 완료된 응용 프로그램과 상이한 응용 프로그램이 실행되면, 상기 a) 단계부터 상기 e) 단계까지 순차적으로 반복 실행하는 단계를 포함한다.

발명의 효과

- [0010] 전술한 본 발명의 과제 해결 수단 중 어느 하나인 응용 프로그램 제어 방법은 프레임워크를 이용하여 주기 타이머 및 데드라인 타이머를 설정하고, 설정된 타이머 및 응용 프로그램의 작업 완료 여부 중 적어도 하나의 정보에 기초하여 스레드의 실행 순서를 조정하고 각 스레드를 각각의 프로세서에 할당시키는 스케줄러를 호출하여 스케줄링함으로써, 멀티 프로세서 환경에서 자바 언어 기반 응용 프로그램의 동작을 동시에 효과적으로 제어할 수 있다.

도면의 간단한 설명

- [0011] 도 1 은 멀티 프로세서 환경에서 프레임워크를 이용하여 자바 언어 기반 응용 프로그램을 제어하는 것을 개략적으로 설명하기 위한 도면이다.
- 도 2 는 본 발명의 일 실시예에 따른 프레임워크를 설명하기 위한 도면이다.
- 도 3 은 본 발명의 일 실시예에 따른 응용 프로그램 제어 방법을 설명하기 위한 도면이다.

발명을 실시하기 위한 구체적인 내용

- [0012] 아래에서는 첨부한 도면을 참조하여 본원이 속하는 기술 분야에서 통상의 지식을 가진 자가 용이하게 실시할 수 있도록 본원의 실시예를 상세히 설명한다. 그러나 본원은 여러 가지 상이한 형태로 구현될 수 있으며 여기에서 설명하는 실시예에 한정되지 않는다. 그리고 도면에서 본원을 명확하게 설명하기 위해서 설명과 관계없는 부분은 생략하였으며, 명세서 전체를 통하여 유사한 부분에 대해서는 유사한 도면 부호를 붙였다.
- [0013] 본원 명세서 전체에서, 어떤 부분이 어떤 구성요소를 "포함" 한다고 할 때, 이는 특별히 반대되는 기재가 없는 한 다른 구성요소를 제외하는 것이 아니라 다른 구성 요소를 더 포함할 수 있는 것을 의미한다. 본원 명세서 전체에서 사용되는 정도의 용어 "약", "실질적으로" 등은 언급된 의미에 고유한 제조 및 물질 허용오차가 제시될 때 그 수치에서 또는 그 수치에 근접한 의미로 사용되고, 본원의 이해를 돕기 위해 정확하거나 절대적인 수치가 언급된 개시 내용을 비양심적인 침해자가 부당하게 이용하는 것을 방지하기 위해 사용된다. 본원 명세서 전체에서 사용되는 정도의 용어 "~(하는) 단계" 또는 "~의 단계"는 "~를 위한 단계"를 의미하지 않는다.
- [0014] 도 1 은 멀티 프로세서 환경에서 프레임워크를 이용하여 자바 언어 기반 응용 프로그램을 제어하는 것을 개략적으로 설명하기 위한 도면이다.
- [0015] 도 1 을 참조하면, 복수의 응용 프로그램(10)은 복수의 스레드(thread, 200)를 통해 작업을 수행하기 위한 모든 정보를 얻을 수 있다. 응용 프로그램(10)은 작업의 수행 명령을 받으면 스레드(200)를 생성하여 작업 수행에 필요한 정보를 얻을 수 있다. 도 1 에서는 응용 프로그램(11, 12)이 각각 두 개의 스레드(200)와 연결되어 있는 것으로 도시되어 있지만, 응용 프로그램(10)은 하나 이상의 스레드(200)를 생성할 수 있으며, 그 수에는 제한 받지 않는다.
- [0016] 또한, 도 1 을 참조하면, 스레드(200)는 프레임워크(framework, 100)에 의해 제어된다. 프레임워크(100)는 사

용자가 동시에 하나 이상의 프로그램 모듈을 사용할 수 있도록 프레임워크를 제공한다. 본 발명은 멀티 프로세서(multi processor) 환경에서의 응용 프로그램 제어 방법에 대한 것이므로, 프레임워크(100)는 복수 개의 프로세서(111, 112)로 구성된 멀티 프로세서(110)를 제어할 수 있다. 프로세서(111, 112)는 응용 프로그램(10)에서 생성한 복수 개의 스레드(200) 중 하나와 연결되어 작업 수행에 필요한 정보를 전달할 수 있다.

[0017] 또한, 프레임워크(100)는 스케줄러(scheduler, 130)를 제어하여 스레드(200)의 실행 순서를 조정하고, 각 스레드(200)를 하나의 프로세서(111, 112)에 할당시킬 수 있다.

[0018] 도 1 을 통해 전술한 내용은 멀티 프로세서(110)를 포함하는 환경에서 자바(JAVA) 언어를 기반으로 제작된 응용 프로그램을 제어하는 것이다. 특히, 자바 언어 중 사용자가 실시간으로 기능을 제어할 수 있는 RTJS를 사용할 수 있다. 구체적으로, 응용 프로그램(10) 및 프레임워크(100)는 자바 언어를 기반으로 명령을 전달하고 작업을 수행할 수 있다. 또한, 전술한 스레드(200)는 자바 언어에서 사용하는 스레드인 자바 스레드일 수 있다. 도 1 을 통해 전술한 응용 프로그램(10)의 제어 방법에 대한 구체적인 설명은 후술하도록 한다.

[0019] 도 2 는 본 발명의 일 실시예에 따른 프레임워크를 설명하기 위한 도면이다.

[0020] 도 2 를 참조하면, 프레임워크(100)는 도 1 을 통해 설명한 것 외에도 프레임워크 클래스 라이브러리, 스케줄러 클래스, JNI(Java Native Interface), RTSJ 라이브러리를 포함한다.

[0021] 여기서 클래스는, 프로그램 단위 중에 나타나는 영문자가 표현할 문법상의 대상을 그 표현 방식에 따라 8 종류로 분류한 것을 말한다. 라이브러리란, 컴퓨터 프로그램에서 자주 사용되는 부분 프로그램들을 모아 놓은 것으로 필요할 때 언제든지 자유롭게 이용할 수 있도록 묶여놓은 집합이다. JNI는 자바 언어로 만들어진 프로그램에서 해당 플랫폼에서만 실행 가능한 네이티브 코드에 접근하기 위해 만들어진 응용 프로그램 인터페이스이다.

[0022] 도 1 및 도 2 를 통해 전술한 프레임워크(100)는 자바 언어를 기반으로 하는 응용 프로그램(10) 동작을 제어함으로써 도 2 에 도시된 것과 같이 JNI를 포함하고 있다. 또한, 사용자가 실시간으로 응용 프로그램(10)을 제어할 수 있도록 자바 언어 중 RTSJ를 사용하므로, RTSJ 라이브러리를 포함하고 있다.

[0023] 한편, 이하에서는 본 발명의 일 실시예에 따른 응용 프로그램 제어 방법에 대해 살핀다.

[0024] 참고로, 본 발명의 일 실시예에 따른 응용 프로그램 제어 방법은 도 1 을 통해 전술한 프레임워크(100)를 이용하여 스레드(200)를 통해 응용 프로그램(10)을 제어하는 방법에 관한 것으로, 앞서 살핀 도 1 에서 설명한 구성과 유사한 구성에 대해서는 동일한 도면 부호를 적용하고 이에 대한 설명은 간략히 하거나 생략하기로 한다.

[0025] 도 3 은 본 발명의 일 실시예에 따른 응용 프로그램 제어 방법을 설명하기 위한 도면이다.

[0026] 도 3 을 참조하면, 적어도 하나 이상의 응용 프로그램(10)이 실행되면 실행된 응용 프로그램(10)을 프레임워크(100)에 등록한다(S210). 단계(S210)는 응용 프로그램(10)이 실행되면, 이전에 미리 시작된 다른 응용 프로그램(11)의 동작에 의해 발생된 정보를 초기화할 수 있다. 예를 들어, 프레임워크(100)에서 미리 시작된 응용 프로그램(11) 작업 수행에 할당된 스레드(200)와 멀티 프로세서(110) 중 어느 하나의 프로세서(111)를 제외한 나머지 스레드(200)와 프로세서(112)의 할당 요청 정보를 초기화할 수 있다. 또한, 후술할 타이머 정보도 초기화할 수 있다.

[0027] 다음으로, 단계(S210)에서 실행되는 응용 프로그램(10)이 실행됨에 따라 생성되는 각 스레드(200)의 실행 순서를 조정하고, 각 스레드(200)를 각각의 프로세서(111, 112)에 할당시키는 스케줄러(130)를 설정한다(S220). 단계(S220)는 스케줄러(130)의 선입선출(First in First Out, FIFO) 방식, 각 스레드(200)의 우선순위 및 각 스레드(200)의 프로세스(111, 112) 선호도 중 적어도 하나를 기초로 각 스레드(200)의 실행 순서를 조정할 수 있다.

[0028] 여기서, 스케줄러는 운영체제에 있어서 독립적으로 실행할 수 있는 프로그램에 대하여 컴퓨터 또는 단말의 시간을 할당하는 명령 프로그램이다. 즉, 컴퓨터 또는 단말에 투입되는 작업 대기 행렬을 조사하여 그 중에서 처리해야 하는 다음 작업을 선출하는 제어 프로그램이다.

[0029] 다음으로, 각 스레드(200)에 대한 작업 주기와 데드라인에 대응되는 주기 타이머와 데드라인 타이머를 각각 생성한다(S230). 단계(S230)에서 생성되는 주기 타이머와 데드라인 타이머는 프레임워크(100)를 통해 생성될 수 있다. 또한, 단계(S230)는 응용 프로그램 사용자에게 의해 입력된 정보를 기초로 사용자 타이머를 생성할 수 있다. 사용자 타이머는 도 2 를 통해 전술한 프레임워크(100)에 포함된 JNI를 통해 사용자로부터 정보를 입력 받아 생성할 수 있다.

- [0030] 전술한 주기 타이머는 스레드(200)에 할당된 작업 주기에 대응되도록 생성할 수 있고, 데드라인 타이머는 스레드(200)가 할당된 작업을 모두 수행하여 완료되지 못하는 경우에도, 강제적으로 작업을 종료시키는 시간에 대응되도록 생성할 수 있다. 예를 들어, 스레드(200)의 작업 주기가 5분으로 설정되어 작업 주기 타이머가 5분으로 생성되고, 데드라인 타이머가 3 분으로 생성되었다면, 스레드(200)가 작업 주기 동안 작업 수행을 완료하지 못한 경우에도 데드라인 타이머인 3분에 작업을 종료시킨다. 반대로, 작업 주기 타이머가 3분으로 생성되고, 데드라인 타이머가 5분으로 생성되었다면, 스레드(200)가 3분만 작업을 수행한 후 동작이 종료된다. 또한, 전술한 사용자 타이머는 작업 주기 타이머와 데드라인 타이머와 별도로 응용 프로그램(10) 사용자가 임의의 시간을 타이머로 설정하여 스레드(200)의 동작 시간을 조정할 수 있다. 더불어 스레드(200)는 주기 타이머, 데드라인 타이머 및 사용자 타이머 외에도 수행 중인 작업이 완료되면 자동으로 동작이 종료된다.
- [0031] 다음으로, 주기 타이머, 데드라인 타이머 및 각 스레드(200)에 대한 작업 완료 여부에 대한 정보 중 적어도 어느 하나를 기초로 단계(S220)에서 설정된 스케줄러(130)를 호출하여 프레임워크(100)가 소유한 제어권한을 호출된 스케줄러(130)에 할당하고, 각 스레드(200)에 대한 작업을 수행한다(S240). 여기서, 스케줄러(130)는 주기 타이머, 데드라인 타이머 및 각 스레드(200)에 대한 작업 완료 여부에 대한 정보를 기초로 응용 프로그램(10)의 작업 수행에 필요한 스레드(200)와 프로세서(111, 112)를 결정할 수 있다. 또한, 프레임워크(100)는 소유한 제어권한을 통해 타이머 생성, 스케줄러(130) 설정, 스케줄러(130) 호출을 할 수 있다. 하지만, 스케줄러(130)가 호출되면 제어권한을 호출된 스케줄러(130)에 할당하여 호출된 스케줄러(130)가 응용 프로그램(10)의 작업 수행에 필요한 스레드(200)와 프로세서(111, 112)를 결정할 수 있고, 스레드(200)에 대한 정보 전달 등의 작업을 수행할 수 있다.
- [0032] 또한, 단계(S240)는 호출된 스케줄러(130)의 제어권한을 기초로 스케줄러(130)가 호출된 시점에서 작업을 수행할 소정의 스레드(200)를 결정할 수 있다. 더불어, 단계(S240)는 호출된 스케줄러(130)의 제어권한을 기초로 작업을 수행할 소정의 스레드(200)가 할당될 프로세서(111, 112)를 결정할 수 있다. 또한, 단계(S240)는 호출된 스케줄러(130)를 통해 각 스레드(200) 중 각각의 프로세서(111, 112)에서 실행되거나 피선점되는 스레드(200)를 결정하고, 결정된 스레드에 대한 정보를 프레임워크(100)의 라이브러리를 통해 전달할 수 있다.
- [0033] 다음으로, 단계(S240)에서 스레드(200)에 대한 작업이 완료되면, 호출된 스케줄러(130)가 소유한 제어권한은 프레임워크(100)로 회수한다(S250). 단계(S250)는 단계(S240)에서 호출된 스케줄러(130)에 할당된 제어권한으로 스레드(200)를 제어하여 스레드(200)를 통해 응용 프로그램(10)의 작업이 완료되면, 제어권한의 소유권을 다시 프레임워크(100)로 회수하고, 다른 응용 프로그램(11, 12)이 실행될 수 있도록 할 수 있다.
- [0034] 다음으로, 단계(S210, S220, S230, S240, S250)를 통해 작업이 완료된 응용 프로그램(10)과 상이한 응용 프로그램(11, 12)이 실행되면, 단계(S210, S220, S230, S240, S250)를 순차적으로 반복 실행한다. 이를 통해, 하나 이상의 응용 프로그램(10)이 실행되어도 하나의 컴퓨터 또는 단말로 동시에 실행하고 제어할 수 있다.
- [0035] 전술한 본 발명의 일 실시예에 따른 응용 프로그램 제어 방법은 멀티 프로세서(110) 환경에서 자바 언어 기반 응용 프로그램(10)을 제어하는 방법에 관한 것이다. 또한, 컴퓨터 관련 응용 프로그램 외에도 모바일 기기의 어플리케이션 관리에도 사용할 수 있고, 멀티 프로세서(110) 환경에서 자바 언어 기반 응용 프로그램(10)을 사용하는 모든 장치 또는 기기에 적용될 수 있다. 더불어, 자바 언어 중 RTSJ를 이용하여 사용자가 실시간으로 외부에서 응용 프로그램(10)을 제어할 수 있다.
- [0036] 또한, 전술한 본 발명의 일 실시예에 따른 응용 프로그램 제어 방법은 프레임워크(100)를 이용하여 주기 타이머 및 데드라인 타이머를 설정하고, 설정된 타이머 및 응용 프로그램(10)의 작업 완료 여부 중 적어도 하나의 정보에 기초하여 스레드(200)의 실행 순서를 조정하고 각 스레드(200)를 각각의 프로세서(111, 112)에 할당시키는 스케줄러(130)를 호출하여 스케줄링함으로써, 멀티 프로세서(110) 환경에서 자바 언어 기반 응용 프로그램(10)의 동작을 동시에 효과적으로 제어할 수 있다.
- [0037] 본 발명의 일 실시예는 컴퓨터에 의해 실행되는 프로그램 모듈과 같은 컴퓨터에 의해 실행가능한 명령어를 포함하는 기록 매체의 형태로도 구현될 수 있다. 컴퓨터 판독 가능 매체는 컴퓨터에 의해 액세스될 수 있는 임의의 가용 매체일 수 있고, 휘발성 및 비휘발성 매체, 분리형 및 비분리형 매체를 모두 포함한다. 또한, 컴퓨터 판독 가능 매체는 컴퓨터 저장 매체 및 통신 매체를 모두 포함할 수 있다. 컴퓨터 저장 매체는 컴퓨터 판독가능 명령어, 데이터 구조, 프로그램 모듈 또는 기타 데이터와 같은 정보의 저장을 위한 임의의 방법 또는 기술로 구현된 휘발성 및 비휘발성, 분리형 및 비분리형 매체를 모두 포함한다. 통신 매체는 전형적으로 컴퓨터 판독가능 명령어, 데이터 구조, 프로그램 모듈, 또는 반송파와 같은 변조된 데이터 신호의 기타 데이터, 또는 기타 전송 메커

니즘을 포함하며, 임의의 정보 전달 매체를 포함한다.

[0038] 본 발명의 방법 및 시스템은 특정 실시예와 관련하여 설명되었지만, 그것들의 구성 요소 또는 동작의 일부 또는 전부는 범용 하드웨어 아키텍처를 갖는 컴퓨터 시스템을 사용하여 구현될 수 있다.

[0039] 전술한 본원의 설명은 예시를 위한 것이며, 본원이 속하는 기술분야의 통상의 지식을 가진 자는 본원의 기술적 사상이나 필수적인 특징을 변경하지 않고서 다른 구체적인 형태로 쉽게 변형이 가능하다는 것을 이해할 수 있을 것이다. 그러므로 이상에서 기술한 실시예들은 모든 면에서 예시적인 것이며 한정적이 아닌 것으로 이해해야만 한다. 예를 들어, 단일형으로 설명되어 있는 각 구성 요소는 분산되어 실시될 수도 있으며, 마찬가지로 분산된 것으로 설명되어 있는 구성 요소들도 결합된 형태로 실시될 수 있다.

[0040] 본원의 범위는 상기 상세한 설명보다는 후술하는 특허청구범위에 의하여 나타내어지며, 특허청구범위의 의미 및 범위 그리고 그 균등 개념으로부터 도출되는 모든 변경 또는 변형된 형태가 본원의 범위에 포함되는 것으로 해석되어야 한다.

부호의 설명

[0041] 10: 응용 프로그램

100: 프레임워크

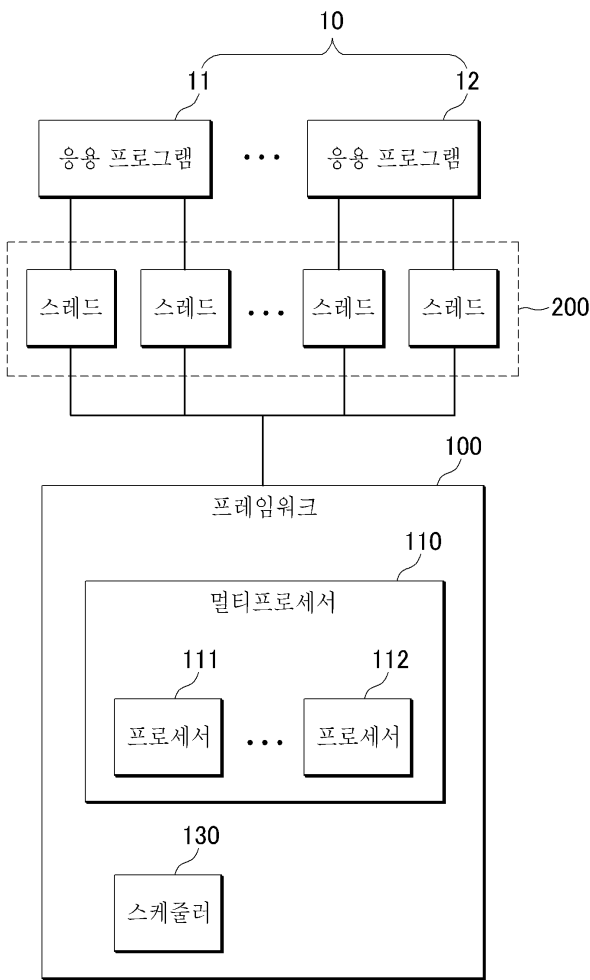
110: 멀티 프로세서

130: 스케줄러

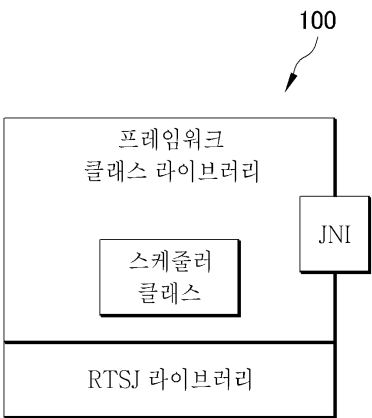
200: 스레드

도면

도면1



도면2



도면3

