(19) 대한민국특허청(KR)
(12) 등록특허공보(B1)

(45) 공고일자 2019년02월20일
(11) 등록번호 10-1950484
(24) 등록일자 2019년02월14일

(51) 국제특허분류(Int. Cl.)
G06F 9/48 (2018.01) *G06F 12/02* (2018.01)
G06F 9/455 (2018.01)
(52) CPC특허분류
G06F 9/4806 (2013.01)
G06F 12/0269 (2013.01)
(21) 출원번호 10-2018-0092390
(22) 출원일자 2018년08월08일
심사청구일자 2018년08월08일
(56) 선행기술조사문헌
KR101015573 B1*
(뒷면에 계속)

(73) 특허권자
엘아이지넥스원 주식회사
경기도 용인시 기흥구 마북로 207 (마북동)
충남대학교산학협력단
대전광역시 유성구 대학로 99 (궁동, 충남대학교)
(72) 발명자
조한무
경기도 성남시 분당구 판교로 333 (삼평동)
서민원
경기도 성남시 분당구 판교로 333 (삼평동)
이철훈
대전광역시 서구 둔산남로 127, 302동 303호 (둔산동, 목련아파트)
(74) 대리인
특허법인우인

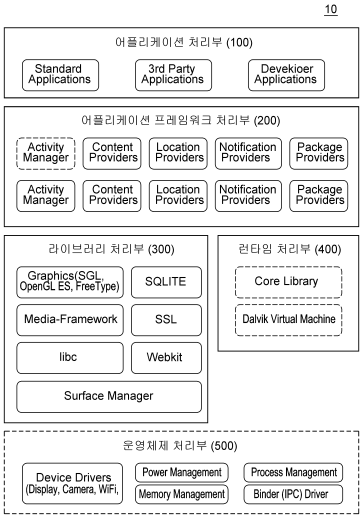
전체 청구항 수 : 총 11 항 심사관 : 유진태

(54) 발명의 명칭 **안드로이드의 실시간 태스크 관리 장치**

(57) 요 약

본 발명에 따르면, 사용자 작업을 수행하는 적어도 하나 이상의 어플리케이션과 관련된 어플리케이션 레벨의 동작을 처리하는 어플리케이션 처리부, 어플리케이션을 수행하기 위한 안드로이드 운영체제를 포함하고, 운영체제 레벨의 동작을 수행하는 운영체제 처리부, 어플리케이션 처리부와, 운영체제 처리부를 매개하고 실시간 작업이 필요한 시간 동안 가비지 컬렉션을 제어하는 미들웨어를 제공함으로써 안드로이드의 실시간 태스크 처리를 가능하게 하는 안드로이드의 실시간 태스크 관리 장치가 개시된다.

대 표 도 - 도1



(52) CPC특허분류
G06F 9/45504 (2013.01)

(56) 선행기술조사문헌
이철훈 외 1명. '안드로이드 달빅 가상머신의 가비지 컬렉션 실행 제어에 관한 연구'. 한국통신학회 동계종합학술발표회 논문집, 2018.01., pp.1118-1119.*
이철훈 외 2명. '안드로이드에 실시간 성능제공을 위한 RTiK 기반 실시간 통신 구현 및 성능 측정'. 한국통신학회 하계종합학술발표회 논문집, 2018.06., pp.1329-1330.*
KR100772867 B1
KR1020120113160 A
*는 심사관에 의하여 인용된 문헌

공지예외적용 : 있음

명 세 서

청구범위

청구항 1

사용자 작업을 수행하는 적어도 하나 이상의 어플리케이션과 관련된 어플리케이션 레벨의 동작을 처리하는 어플리케이션 처리부; 상기 어플리케이션을 수행하기 위한 안드로이드 운영체제를 포함하고, 운영체제 레벨의 동작을 수행하는 운영체제 처리부; 및 상기 어플리케이션 처리부와, 상기 운영체제 처리부를 매개하는 미들웨어; 를 포함하며, 상기 미들웨어는, 실시간 작업이 필요한 시간 동안 가비지 컬렉션을 제어하고,

상기 미들웨어는, 상기 어플리케이션을 실행시키는데 필요한 자바로 작성된 API(Application Programming Interface)로 접근하는 어플리케이션 프레임워크 처리부; 및 달빅 가상머신(Dalvik VM)을 이용하여, 메모리 관리와 멀티쓰레딩(multithreading)을 수행하는 런타임 처리부;를 포함하며,

상기 운영체제 처리부는, 리눅스 커널을 기반으로 상기 달빅 가상머신(Dalvik VM)과 상기 어플리케이션 프레임워크 처리부를 통해 어플리케이션이 실행될 수 있는 환경을 제공하고, 상기 리눅스 커널의 실시간성 제어를 위해 사용되는 실시간 처리(Real-Time implant Kernel, RTiK) 모듈을 포함하며,

상기 실시간 처리 모듈은, 상기 리눅스 커널의 영역에서 실시간 타이머 기능을 제공하는 것을 특징으로 하는 안드로이드의 실시간 태스크 관리 장치.

청구항 2

제1항에 있어서,

상기 미들웨어는,

상기 운영체제 처리부의 상단에 존재하며, 안드로이드 환경에 맞게 수정된 라이브러리 레벨의 동작을 수행하는 라이브러리 처리부;를 더 포함하는 것을 특징으로 하는 안드로이드의 실시간 태스크 관리 장치.

청구항 3

삭제

청구항 4

삭제

청구항 5

제2항에 있어서,

상기 달빅 가상머신(Dalvik VM)은, 상기 가비지 컬렉션의 수행을 제어하는 가비지 컬렉션 제어부;를 포함하며,

상기 가비지 컬렉션의 동작 방식은, 동시 마크 스윕(Concurrent Mark Sweep, CMS) 알고리즘을 이용하여 구현되되,

상기 가비지 컬렉션 제어부는,

상기 달빅 가상머신에서 가용 메모리 확보를 위해 상기 가비지 컬렉션을 실행해야 하는 시점에서 특정 플래그 값에 따라 실행 여부를 결정하는 것을 특징으로 하는 안드로이드의 실시간 태스크 관리 장치.

청구항 6

제2항에 있어서,

상기 라이브러리 처리부는,

상기 실시간 처리(Real-Time implant Kernel, RTiK) 모듈에서 제공하는 상기 실시간 타이머 기능을 상기 어플리

케이션에 제공하는 인터페이스인 실시간 API 레이어(RTiK-API’s Layer)를 포함하며,

상기 실시간 API 레이어(RTiK-API’s Layer)는,

모듈 제어 함수를 이용하여 상기 실시간 처리(Real-Time implant Kernel, RTiK) 모듈을 제어하는 것을 특징으로 하는 안드로이드의 실시간 태스크 관리 장치.

청구항 7

제6항에 있어서,

상기 모듈 제어 함수는,

상기 실시간 처리(Real-Time implant Kernel, RTiK) 모듈을 통해 안드로이드의 태스크를 제어하는 태스크 제어 함수; 및

상기 실시간 타이머를 동작시켜 상기 실시간 처리(Real-Time implant Kernel, RTiK) 모듈의 시작 또는 종료를 명령하는 타이머 제어 함수;를 포함하는 것을 특징으로 하는 안드로이드의 실시간 태스크 관리 장치.

청구항 8

제6항에 있어서,

상기 어플리케이션 처리부는,

상기 실시간 처리(Real-Time implant Kernel, RTiK) 모듈을 제어하는 자바로 작성된 실시간 API 클래스(RTiK-APIs Class)를 더 포함하며,

상기 실시간 API 클래스(RTiK-APIs Class)는, 상기 실시간 API 레이어(RTiK-API’s Layer)를 통해 상기 실시간 처리(Real-Time implant Kernel, RTiK) 모듈에 접근하는 것을 특징으로 하는 안드로이드의 실시간 태스크 관리 장치.

청구항 9

제8항에 있어서,

상기 실시간 API 레이어(RTiK-API’s Layer)는, 상기 실시간 처리(Real-Time implant Kernel, RTiK) 모듈에 접근하는 네이티브 라이브러리(Native Library)를 포함하며,

상기 실시간 API 클래스(RTiK-APIs Class)는, 네이티브 라이브러리(Native Library)를 통해 상기 실시간 처리 모듈에 접근할 수 있는 함수를 제공하며,

상기 실시간 처리 모듈에 접근할 수 있는 함수는,

상기 실시간 API 레이어(RTiK-API’s Layer)의 접근을 위하여 JNI(Java Native Interface) 함수를 이용한 JNI 프로토콜을 이용하는 것을 특징으로 하는 안드로이드의 실시간 태스크 관리 장치.

청구항 10

제9항에 있어서,

상기 라이브러리 처리부는,

실시간 태스크 수행을 위해 P쓰레드(Pthread) 라이브러리를 이용하여 상기 네이티브 라이브러리(Native Library) 영역에 태스크를 생성하는 것을 특징으로 하는 안드로이드의 실시간 태스크 관리 장치.

청구항 11

제5항에 있어서,

상기 런타임 처리부는,

상기 달빅 가상머신(Dalvik VM)의 가비지 컬렉션 실행 여부를 저장하는 저장 파일을 포함하는 리눅스 파일 처리부;를 더 포함하며,

상기 달빅 가상머신은, 코드 상에서 상기 저장 파일에 접근가능 한 것을 특징으로 하는 안드로이드의 실시간 태스크 관리 장치.

청구항 12

제11항에 있어서,

상기 가비지 컬렉션 제어부는,

달빅 가상머신(Dalvik VM)이 초기화 되는 과정에서, 상기 저장 파일의 데이터를 읽어오는 리드(read)부;

상기 리드(read)부에서 읽어온 값을 상기 특정 플래그 값에 따른 플래그 변수 값으로 설정하는 플래그 변수 설정부;

상기 가비지 컬렉션이 필요한 시점에서 설정된 상기 플래그 변수 값을 확인하는 플래그 변수 확인부; 및

상기 플래그 변수 값에 따라 상기 가비지 컬렉션 수행을 판단하는 가비지 컬렉션 수행 판단부;를 포함하는 것을 특징으로 하는 안드로이드의 실시간 태스크 관리 장치.

청구항 13

제12항에 있어서,

상기 가비지 컬렉션 수행 판단부는,

상기 플래그 변수 값이 1인 경우 상기 가비지 컬렉션을 실행하지 않고, 0인 경우 상기 가비지 컬렉션을 실행하는 것을 특징으로 하는 안드로이드의 실시간 태스크 관리 장치.

발명의 설명

기 술 분 야

[0001] 본 발명은 안드로이드의 실시간 태스크 관리 장치에 관한 것으로, 특히 가비지 컬렉션 실행을 제어하는 안드로이드의 실시간 태스크 관리 장치에 관한 것이다.

배 경 기 술

[0002] 실시간으로 데이터를 획득하고 평가하는 것이 중요한 군용 점검 장비와 같은 시스템에서는 운영체제 레벨에서 실시간 처리가 가능해야 한다.

[0003] 종래의 윈도우 운영체제에는 윈도우의 실시간성을 개선하기 위한 여러 방법들이 있으나 안드로이드에서는 동일 방법을 사용한 기술은 없다.

[0004] 또한, RTAndroid는 최신 안드로이드 커널 패치가 메인 라인 커널에 아직 통합되지 않았기 때문에, RT_PREEMPT patch를 적용하기 위해 기존 리눅스 커널을 광범위하게 수정해야 하는 단점이 있으며, RTDroid는 안드로이드의 모든 시스템을 처음부터 다시 구축하는 방식으로, 안드로이드 버전의 지속적인 발전에 따른 유지보수가 어려울 수 있다.

발명의 내용

해결하려는 과제

[0005] 본 발명은 안드로이드의 실시간 태스크 관리 장치 및 관리 방법으로 사용자 작업을 수행하는 적어도 하나 이상의 어플리케이션과 관련된 어플리케이션 레벨의 동작을 처리하는 어플리케이션 처리부, 어플리케이션을 수행하기 위한 안드로이드 운영체제를 포함하고, 운영체제 레벨의 동작을 수행하는 운영체제 처리부, 어플리케이션 처리부와, 운영체제 처리부를 매개하는 미들웨어를 포함하며, 미들웨어는, 실시간 작업이 필요한 시간 동안 가비지 컬렉션을 제어하여 안드로이드의 실시간 태스크 처리를 가능하게 하는데 그 목적이 있다.

[0006] 또한, 안드로이드의 가비지 컬렉션 실행을 제어하는 구조를 설계 및 구현함으로써 실시간 작업이 필요한 시간

동안 가비지 컬렉션을 제어하도록 하는데 또 다른 목적이 있다.

[0007] 본 발명의 명시되지 않은 또 다른 목적들은 하기의 상세한 설명 및 그 효과로부터 용이하게 추론할 수 있는 범위 내에서 추가적으로 고려될 수 있다.

과제의 해결 수단

[0008] 상기 과제를 해결하기 위해, 본 발명의 일 실시예에 따른 안드로이드의 실시간 태스크 관리 장치는, 사용자 작업을 수행하는 적어도 하나 이상의 어플리케이션과 관련된 어플리케이션 레벨의 동작을 처리하는 어플리케이션 처리부, 상기 어플리케이션을 수행하기 위한 안드로이드 운영체제를 포함하고, 상기 운영체제 레벨의 동작을 수행하는 운영체제 처리부, 상기 어플리케이션 처리부와, 상기 운영체제 처리부를 매개하는 미들웨어를 포함하며, 상기 미들웨어는, 실시간 작업이 필요한 시간 동안 가비지 컬렉션을 제어한다.

[0009] 여기서, 미들웨어는, 상기 어플리케이션을 실행시키는데 필요한 자바로 작성된 API(Application Programming Interface)로 접근하는 어플리케이션 프레임워크 처리부, 상기 운영체제 처리부의 상단에 존재하며, 안드로이드 환경에 맞게 수정된 라이브러리 레벨의 동작을 수행하는 라이브러리 처리부 및 달빅 가상머신(Dalvik VM)을 이용하여, 메모리 관리와 멀티쓰레딩(multithreading)을 수행하는 런타임 처리부를 포함한다.

[0010] 여기서, 운영체제 처리부는, 리눅스 커널을 기반으로 상기 달빅 가상머신(Dalvik VM)과 상기 어플리케이션 프레임워크 처리부를 통해 어플리케이션이 실행될 수 있는 환경을 제공한다.

[0011] 여기서, 운영체제 처리부는, 상기 리눅스 커널의 실시간성 제어를 위해 사용되는 실시간 처리(Real-Time implant Kernel, RTiK) 모듈을 포함하며, 상기 실시간 처리 모듈은, 상기 리눅스 커널의 영역에서 실시간 타이머 기능을 제공한다.

[0012] 여기서, 달빅 가상머신(Dalvik VM)은, 상기 가비지 컬렉션의 수행을 제어하는 가비지 컬렉션 제어부를 포함하며, 상기 가비지 컬렉션의 동작 방식은, 동시 마크 스윕(Concurrent Mark Sweep, CMS) 알고리즘을 이용하여 구현되되, 상기 가비지 컬렉션 제어부는, 상기 달빅 가상머신에서 가용 메모리 확보를 위해 상기 가비지 컬렉션을 실행해야 하는 시점에서 특정 플래그 값에 따라 실행 여부를 결정한다.

[0013] 여기서, 라이브러리 처리부는, 상기 실시간 처리(Real-Time implant Kernel, RTiK) 모듈에서 제공하는 상기 실시간 타이머 기능을 상기 어플리케이션에 제공하는 인터페이스인 실시간 API 레이어(RTiK-API's Layer)를 포함하며, 상기 실시간 API 레이어(RTiK-API's Layer)는, 모듈 제어 함수를 이용하여 상기 실시간 처리(Real-Time implant Kernel, RTiK) 모듈을 제어한다.

[0014] 여기서, 모듈 제어 함수는, 상기 실시간 처리(Real-Time implant Kernel, RTiK) 모듈을 통해 안드로이드의 태스크를 제어하는 태스크 제어 함수 및 상기 실시간 타이머를 동작시켜 상기 실시간 처리(Real-Time implant Kernel, RTiK) 모듈의 시작 또는 종료를 명령하는 타이머 제어 함수를 포함한다.

[0015] 여기서, 어플리케이션 처리부는, 상기 실시간 처리(Real-Time implant Kernel, RTiK) 모듈을 제어하는 자바로 작성된 실시간 API 클래스(RTiK-APIs Class)를 더 포함하며, 상기 실시간 API 클래스(RTiK-APIs Class)는, 상기 실시간 API 레이어(RTiK-API's Layer)를 통해 상기 실시간 처리(Real-Time implant Kernel, RTiK) 모듈에 접근한다.

[0016] 여기서, 실시간 API 레이어(RTiK-API's Layer)는, 상기 실시간 처리(Real-Time implant Kernel, RTiK) 모듈에 접근하는 네이티브 라이브러리(Native Library)를 포함하며, 상기 실시간 API 클래스(RTiK-APIs Class)는, 네이티브 라이브러리(Native Library)를 통해 상기 실시간 처리 모듈에 접근할 수 있는 함수를 제공하며, 상기 실시간 처리 모듈에 접근할 수 있는 함수는, 상기 실시간 API 레이어(RTiK-API's Layer)의 접근을 위하여 JNI(Java Native Interface) 함수를 이용한 JNI 프로토콜을 이용한다.

[0017] 여기서, 라이브러리 처리부는, 실시간 태스크 수행을 위해 P쓰레드(Pthread) 라이브러리를 이용하여 상기 네이티브 라이브러리(Native Library) 영역에 태스크를 생성한다.

[0018] 여기서, 런타임 처리부는, 상기 달빅 가상머신(Dalvik VM)의 가비지 컬렉션 실행 여부를 저장하는 저장 파일을 포함하는 리눅스 파일 처리부를 더 포함하며, 상기 달빅 가상머신은, 코드 상에서 상기 저장 파일에 접근가능하다.

[0019] 여기서, 가비지 컬렉션 제어부는, 달빅 가상머신(Dalvik VM)이 초기화 되는 과정에서, 상기 저장 파일의 데이터를 읽어오는 리드(read)부, 상기 리드(read)부에서 읽어온 값을 상기 플래그 변수로 설정하는 플래그 변수 설정

부, 상기 가비지 컬렉션이 필요한 시점에서 설정된 상기 플래그 변수 값을 확인하는 플래그 변수 확인부 및 상기 플래그 변수 값에 따라 상기 가비지 컬렉션 수행을 판단하는 가비지 컬렉션 수행 판단부를 포함한다.

[0020] 여기서, 가비지 컬렉션 수행 판단부는, 상기 플래그 변수 값이 1인 경우 상기 가비지 컬렉션을 실행하지 않고, 0인 경우 상기 가비지 컬렉션을 실행한다.

발명의 효과

[0021] 이상에서 설명한 바와 같이 본 발명의 실시예들에 의하면, 사용자 작업을 수행하는 적어도 하나 이상의 어플리케이션과 관련된 어플리케이션 레벨의 동작을 처리하는 어플리케이션 처리부, 어플리케이션을 수행하기 위한 안드로이드 운영체제를 포함하고, 운영체제 레벨의 동작을 수행하는 운영체제 처리부, 어플리케이션 처리부와, 운영체제 처리부를 매개하는 미들웨어를 포함하며, 미들웨어는, 실시간 작업이 필요한 시간 동안 가비지 컬렉션을 제어하여 안드로이드의 실시간 태스크 처리를 가능하게 할 수 있다.

[0022] 또한, 안드로이드의 가비지 컬렉션 실행을 제어하는 구조를 설계 및 구현함으로써 실시간 작업이 필요한 시간 동안 가비지 컬렉션을 제어할 수 있다.

[0023] 이에 따라, 성능을 검증할 수 있는 국산화된 실시간 이식 커널 개발을 통해, 전용 단말 개발에 사용되는 장비 개발과 유지, 보수비용을 감소시킬 수 있고, 실시간 운영체제에 대한 IPR을 확보할 수 있다.

[0024] 여기에서 명시적으로 언급되지 않은 효과라 하더라도, 본 발명의 기술적 특징에 의해 기대되는 이하의 명세서에서 기재된 효과 및 그 잠정적인 효과는 본 발명의 명세서에 기재된 것과 같이 취급된다.

도면의 간단한 설명

[0025] 도 1 및 도 2은 본 발명의 일 실시예에 따른 안드로이드의 실시간 태스크 관리 장치를 나타낸 도면이다.

도 3은 본 발명의 일 실시예에 따른 안드로이드의 실시간 태스크 관리 장치의 런타임 처리부를 나타낸 블록도이다.

도 4는 본 발명의 일 실시예에 따른 안드로이드의 실시간 태스크 관리 장치의 가비지 컬렉션 제어부를 나타낸 도면이다.

도 5는 본 발명의 일 실시예에 따른 안드로이드의 실시간 태스크 관리 방법을 나타낸 흐름도이다.

도 6은 본 발명의 일 실시예에 따른 본 발명의 일 실시예에 따른 안드로이드의 실시간 태스크 관리 방법을 나타낸 도면이다.

도 7은 본 발명의 일 실시예에 따른 호출 주기의 변화를 나타낸 그래프이다.

도 8 및 도 9는 본 발명의 일 실시예에 따른 힙 크기의 변화를 나타낸 그래프이다.

발명을 실시하기 위한 구체적인 내용

[0026] 이하, 본 발명에 관련된 안드로이드의 실시간 태스크 관리 장치에 대하여 도면을 참조하여 보다 상세하게 설명한다. 그러나, 본 발명은 여러 가지 상이한 형태로 구현될 수 있으며, 설명하는 실시예에 한정되는 것이 아니다. 그리고, 본 발명을 명확하게 설명하기 위하여 설명과 관계없는 부분은 생략되며, 도면의 동일한 참조부호는 동일한 부재임을 나타낸다.

[0027] 어떤 구성요소가 다른 구성요소에 "연결되어" 있다거나 "접속되어" 있다고 언급된 때에는, 그 다른 구성요소에 직접적으로 연결되어 있거나 또는 접속되어 있을 수도 있지만, 중간에 다른 구성요소가 존재할 수도 있다고 이해되어야 할 것이다.

[0028] 이하의 설명에서 사용되는 구성요소에 대한 접미사 “모듈” 및 “부”는 명세서 작성의 용이함만이 고려되어 부여되거나 혼용되는 것으로서, 그 자체로 서로 구별되는 의미 또는 역할을 갖는 것은 아니다.

[0029] 제1, 제2 등의 용어는 다양한 구성요소들을 설명하는데 사용될 수 있지만, 구성요소들은 용어들에 의해 한정되어서는 안 된다. 상기 용어들은 하나의 구성요소를 다른 구성요소로부터 구별하는 목적으로만 사용된다.

[0030] 본 발명은 안드로이드의 실시간 태스크 관리 장치에 관한 것이다.

[0031] 도 1 및 도 2은 본 발명의 일 실시예에 따른 안드로이드의 실시간 태스크 관리 장치를 나타낸 도면이다.

[0032] 안드로이드는 모바일 플랫폼으로, 운영체제와 미들웨어 및 핵심 어플리케이션이 포함된 소프트웨어 스택으로 정의된다. 안드로이드 플랫폼은 도 1과 같은 스택 구조를 가진다.

[0033] 본 발명의 처리부는 독립적으로 수행되는 것으로 예시하였으나, 서로 다른 프로세서에서 각각 구현이 되거나, 하나의 프로세스에 포함되어 구현이 가능하다.

[0034] 도 1을 참조하면, 본 발명의 일 실시예에 따른 안드로이드의 실시간 태스크 관리 장치(10)는 어플리케이션 처리부(100), 어플리케이션 프레임워크 처리부(200), 라이브러리 처리부(300), 런타임 처리부(400), 운영체제 처리부(500)를 포함한다.

[0035] 안드로이드의 실시간 태스크 관리 장치(10)는 운영체제 레벨에서 실시간으로 데이터를 획득하고 평가하는 실시간 처리가 가능한 관리 장치이다.

[0036] 전장에서 사용 가능한 사격통제장치나 모니터링 장비 중 실시간성이 요구되는 안드로이드 운영체제를 사용하는 휴대용 단말에서 활용 가능하다.

[0037] 실시간으로 데이터를 획득하고 평가하는 것이 중요한 군용 점검 장비와 같은 시스템에서는 운영체제 레벨에서 실시간 처리가 가능해야 한다. 기술의 발전으로 기존 장비를 휴대용 단말로 대체하려는 요구가 있으나, 안드로이드가 적용된 휴대용 단말은 실시간성이 요구되는 시스템에 적합하지 않다. 안드로이드에서는 가비지 컬렉션을 통해 가용 메모리를 확보하는데, 이 작업이 수행되는 동안 다른 태스크가 중단되어 특정 태스크의 주기성을 보장할 수 없다. 본 발명의 일 실시예에 따른 안드로이드의 실시간 태스크 관리 장치(10)는 실시간 작업이 필요한 시간 동안 가비지 컬렉션을 제어하여 실시간 성능을 보장하며, 안드로이드에 실시간 성능 보장을 위한 RTiK(Real Time implant Kernel)을 적용한다.

[0038] 어플리케이션 처리부(100)는 사용자 작업을 수행하는 적어도 하나 이상의 어플리케이션과 관련된 어플리케이션 레벨의 동작을 처리한다.

[0039] 운영체제 처리부(500)는 어플리케이션을 수행하기 위한 안드로이드 운영체제를 포함하고, 상기 운영체제 레벨의 동작을 수행한다.

[0040] 운영체제 처리부(500)는 스택 구조의 가장 아래에 위치하며 리눅스 커널로, 안드로이드 플랫폼의 기반이 된다. 리눅스 커널은 프로세스 및 메모리 관리, 보안 기능과 같은 시스템 기능을 제공한다. 또한 네트워크 인터페이스와 디바이스 드라이버 등을 관리하여 주변 하드웨어와의 인터페이스를 용이하게 하는 역할을 한다.

[0041] 어플리케이션 처리부와, 운영체제 처리부를 매개하는 미들웨어를 포함하며, 미들웨어는 어플리케이션 프레임워크 처리부(200), 라이브러리 처리부(300), 런타임 처리부(400)를 포함한다.

[0042] 미들웨어는, 실시간 작업이 필요한 시간 동안 가비지 컬렉션을 제어한다.

[0043] 어플리케이션 프레임워크 처리부(200)는 어플리케이션을 실행시키는데 필요한 자바로 작성된 API(Application Programming Interface)로 접근한다.

[0044] 어플리케이션 프레임워크는 자바로 작성된 API 형태로 접근할 수 있으며, 안드로이드 어플리케이션에 다양한 상위 레벨 서비스를 제공한다.

[0045] 라이브러리 처리부(300)는 운영체제 처리부(500)의 상단에 존재하며, 안드로이드 환경에 맞게 수정된 라이브러리 레벨의 동작을 수행한다.

[0046] 라이브러리 처리부(300)는 리눅스 커널 상단에 위치하며 C/C++ 라이브러리 집합을 포함한다. 라이브러리에는 안드로이드 환경에 맞게 수정된 C라이브러리(libc)인 Bionic, 오디오 및 비디오, 그래픽, 브라우저, SQLite 등이 포함된다.

[0047] 런타임 처리부(400)는 달빅 가상머신(Dalvik VM)을 이용하여, 메모리 관리와 멀티쓰레딩(multithreading)을 수행한다.

[0048] 런타임 처리부(400)는 달빅 가상머신으로, 자바 가상머신을 대신하여 안드로이드에 최적화되어 설계된다. 달빅 가상머신은 메모리 관리와 멀티 쓰레딩을 위해 리눅스를 완전히 이용한다.

[0049] 달빅 가상머신은 안드로이드 KitKat(API 19) 버전까지 사용된 런타임 시스템이다. 어플리케이션 마다 별도의 가상머신에서 동작하기 때문에 가비지 컬렉션 또한 독립적으로 발생한다. 달빅 가상 머신은 리눅스 커널이 관리하

는 프로세스 내에 존재한다.

[0050] 달빅 가상머신(Dalvik VM)은, 가비지 컬렉션의 수행을 제어하는 가비지 컬렉션 제어부를 포함하며, 가비지 컬렉션의 동작 방식은, 동시 마크 스윕(Concurrent Mark Sweep, CMS) 알고리즘을 이용하여 구현되되, 가비지 컬렉션 제어부는 달빅 가상머신에서 가용 메모리 확보를 위해 가비지 컬렉션을 실행해야 하는 시점에서 특정 플래그 값에 따라 실행 여부를 결정한다.

[0051] 동시 마크 스윕(Concurrent Mark Sweep, CMS) 알고리즘은 다음과 같다. 먼저 컬렉션 주기의 시작 부분에서 일괄적으로 모든 뮤테이터 쓰레드들을 중단시킨 후, 힙 루트 스택에서 참조하는 도달 가능한(reachable) 객체들을 찾아서 마킹한다(Initial Mark). 중단시켰던 뮤테이터 쓰레드들을 다시 실행시킨 후에, 이전 단계에서 마킹한 객체가 참조하는 객체들을 따라가면서 도달 가능한지 확인하여 마킹한다(Concurrent Mark). 이 단계가 끝나면 가비지 컬렉터는 다시 모든 뮤테이터 쓰레드들을 중단시킨다.

[0052] 이전 단계에서의 동시 마킹은 뮤테이터 쓰레드들의 동작과 함께 진행되었기 때문에 새로 추가되거나 참조가 끊긴 객체가 있는지 다시 확인한다(Remark). 그런 다음 뮤테이터 쓰레드들을 다시 시작시키고, 마킹되지 않은 객체들은 가비지로서 정리하는 작업을 수행한다(Concurrent Sweep).

[0053] 자바 언어로 작성되는 안드로이드 어플리케이션은 달빅 가상머신(Dalvik Virtual Machine) 상에서 동작하는데, 달빅 가상머신에서는 가용 메모리 확보를 위해 가비지 컬렉션(Garbage Collection, GC)을 이용하여 메모리를 관리한다.

[0054] 본 발명의 일 실시예에 따른 안드로이드의 실시간 태스크 관리 장치(10)는 안드로이드에 가비지 컬렉션 실행을 제어하는 구조를 설계 및 구현하여 가비지 컬렉션으로 인한 실시간 태스크의 중단시간을 최소화 할 수 있도록 가비지 컬렉션의 실행을 제어하는 방법을 사용한다. 또한 안드로이드 환경에서 실시간성을 제공할 수 있도록 RTiK를 개선하여 적용한다.

[0055] 운영체제 처리부(500)는, 리눅스 커널을 기반으로 달빅 가상머신(Dalvik VM)과 어플리케이션 프레임워크 처리부(200)를 통해 어플리케이션이 실행될 수 있는 환경을 제공하게 된다.

[0056] 도 2는 본 발명의 일 실시예에 따른 안드로이드의 실시간 태스크 관리 장치에 있어서 추가되는 RT 스택구조를 나타낸 것이다.

[0057] 안드로이드 운영체제는 리눅스 커널을 사용하며, 안드로이드 프레임워크 실행을 위한 스택 구조를 가진다. 이에 따라서 임베디드 리눅스에 실시간성 제어를 위해 개발한 RTiK-AL를 사용할 수 있으나, 가상 머신에서 동작하는 안드로이드에서 RTiK-AL 모듈 사용을 위한 인터페이스 및 제어 구조를 필요로 하게 된다.

[0058] 도 2에 나타난 바와 같이 본 발명의 일 실시예에 따른 안드로이드의 실시간 태스크 관리 장치는 안드로이드의 스택 구조에서 실시간성 제공을 위한 방법으로 RTiK-API's Layer 및 RTiK-APIs Class가 추가된다.

[0059] 도 2를 참조하면, 운영체제 처리부(500)는 리눅스 커널의 실시간성 제어를 위해 사용되는 실시간 처리(Real-Time implant Kernel, RTiK) 모듈(510)을 포함한다.

[0060] 실시간 처리 모듈은(510), 상기 리눅스 커널의 영역에서 실시간 타이머 기능을 제공한다.

[0061] 라이브러리 처리부(300)는, 실시간 처리(Real-Time implant Kernel, RTiK) 모듈(510) 제어를 위한 라이브러리 영역이다.

[0062] 실시간 처리(Real-Time implant Kernel, RTiK) 모듈(510)에서 제공하는 상기 실시간 타이머 기능을 상기 어플리케이션에 제공하는 인터페이스인 실시간 API 레이어(RTiK-API’s Layer)(310)를 포함한다.

[0063] 실시간 API 레이어(RTiK-API’s Layer)(310)는, 모듈 제어 함수를 이용하여 상기 실시간 처리(Real-Time implant Kernel, RTiK) 모듈을 제어한다.

[0064] 모듈 제어 함수는, 실시간 처리(Real-Time implant Kernel, RTiK) 모듈을 통해 안드로이드의 태스크를 제어하는 태스크 제어 함수 및 실시간 타이머를 동작시켜 상기 실시간 처리(Real-Time implant Kernel, RTiK) 모듈의 시작 또는 종료를 명령하는 타이머 제어 함수를 포함한다.

[0065] 리눅스 커널에 실시간성 제어를 위해 사용되는 RTiK-Module은 커널 영역에서 실시간 타이머 기능을 제공한다. RTiK-APIs Layer는 이를 안드로이드 어플리케이션에 제공하기 위한 중간 단계의 인터페이스로 제공된다. 리눅스 커널 모듈로 제공되는 RTiK-Module제어를 위해 ioctl을 통한 커널에 접근할 수 있도록 모듈 제어 함수를 제공한

다. 표 1은 모듈 제어 함수를 위해 제공하는 RTiK-APIs Layer의 모듈 제어 함수 목록을 나타낸 것이다.

표 1

[0066]

기능	함수명	설명
태스크 제어	rtik_core_AddProc()	실시간 태스크 관리를 위해 모듈에 등록하는 함수
	rtik_core_DelProc()	사용이 끝난 태스크의 등록 해제를 위한 함수
	rtik_core_recvAck()	태스크 응답을 위한 함수
타이머 제어	rtik_core_StartTimer()	실시간 타이머 시작 명령 함수
	rtik_core_EndTimer()	실시간 타이머 중지 명령 함수

[0067] 실시간 처리(Real-Time implant Kernel, RTiK) 모듈(510)을 통해 태스크의 실시간 성능 제공을 위해 태스크 제어를 위한 모듈 제어 함수 3개를 제공한다. 실시간 태스크 등록 및 사용 해제 함수와 시스템의 오버헤드를 줄이기 위한 태스크 응답 확인 함수를 제공한다.

[0068] 또한, 태스크 등록 후에 실시간 타이머를 동작시켜 실시간 처리(Real-Time implant Kernel, RTiK) 모듈(510)의 시작 및 종료를 명령할 수 있도록 하는 2개의 함수를 제공한다. 이를 통해서 사용자 영역에 실시간성을 제공할 수 있도록 하는 인터페이스 함수가 제공된다. 안드로이드 운영체제에서는 사용자 어플리케이션에 cpp 언어로 작성된 인터페이스 함수를 사용하도록 NDK(Native Development Kit)을 통해 Native C/C++ Library 영역에 RTiK-APIs Layer를 등록하여 안드로이드 사용자 영역에서 접근할 수 있도록 한다.

[0069] 라이브러리 처리부(300)는, 실시간 태스크 수행을 위해 P쓰레드(Pthread) 라이브러리를 이용하여 네이티브 라이브러리(Native Library) 영역에 태스크를 생성한다.

[0070] 기존 안드로이드의 자바 스레드는 우선순위를 제공하고 있으나, 이는 리눅스에서 Nice 값을 이용한 동적 우선순위 방법을 이용하는 것으로 실시간 태스크에 사용할 수 없는 한계를 보인다. 이를 해결하기 위해 본 발명의 일 실시예에 따른 안드로이드의 실시간 태스크 관리 장치(10)는 Pthread 라이브러리를 이용한 Native Library 영역에 태스크를 생성한다. 구체적으로, rtik_task_create() 함수를 통해서 pthread를 통한 스레드를 생성한다. 이때, 우선순위를 조정하여 실시간 태스크를 위한 설정을 수행한다. 또한, 다양한 사용자의 함수에 대한 호출을 위해 함수를 등록할 수 있도록 인자를 통해 제공한다.

[0071] 사용자 함수 등록을 위한 자료구조는 매개변수가 2개인 함수 타입을 임의로 사용자가 지정할 수 있도록 하여, 사용자가 작성하는 다양한 함수가 실행될 수 있도록 한다. 이를 통해서 사용자 영역에 실시간 태스크를 제공할 수 있도록 한다.

[0072] 어플리케이션 프레임워크 처리부(200)는 어플리케이션을 실행시키는데 필요한 자바로 작성된 API(Application Programming Interface)로 접근한다.

[0073] 어플리케이션 처리부(100)는, 실시간 처리(Real-Time implant Kernel, RTiK) 모듈을 제어하는 자바로 작성된 실시간 API 클래스(RTiK-APIs Class)(110)를 포함한다.

[0074] 실시간 API 클래스(RTiK-APIs Class)(110)는, 실시간 API 레이어(RTiK-API’s Layer)(310)를 통해 상기 실시간 처리(Real-Time implant Kernel, RTiK) 모듈에 접근한다.

[0075] 실시간 API 레이어(RTiK-API’s Layer)(310)는, 실시간 처리(Real-Time implant Kernel, RTiK) 모듈에 접근하는 네이티브 라이브러리(Native Library)를 포함한다.

[0076] 실시간 API 클래스(RTiK-APIs Class)(110)는, 네이티브 라이브러리(Native Library)를 통해 상기 실시간 처리 모듈에 접근할 수 있는 함수를 제공하며, 실시간 처리 모듈(510)에 접근할 수 있는 함수는, 실시간 API 레이어(RTiK-API’s Layer)의 접근을 위하여 JNI(Java Native Interface) 함수를 이용한 JNI 프로토콜(210)을 이용한다.

[0077] 안드로이드 운영체제는 가상 머신에서 동작하는 어플리케이션을 통해 사용자에게 서비스를 제공한다. 이에 따라

실시간 모듈을 제어하기 위해서는 자바로 작성된 클래스를 필요로 하게 된다. 구체적으로, cpp 언어로 작성된 실시간 API 레이어(RTiK-API's Layer)(310) 접근을 위하여 자바와 cpp 간의 정보 전달을 위한 System.loadLibrary() 함수를 통해 라이브러리를 메모리에 읽어주며, 접근을 위한 4개의 함수는 표 2에 나타난다.

표 2

[0078]

기능	함수명	설명
초기화	_initialize()	JNI 인터페이스 초기화
타이머 관리	_startTimer()	타이머 시작 명령 함수
	_stopTimer()	타이머 종료 명령 함수
태스크 관리	_joinResult()	실시간 태스크 종료응답 함수

[0079] 초기화 함수를 이용하여 JNI에서 제공하는 Native Library의 함수 내부에 있는 기본 자료구조를 초기화 한다. 또한, 타이머 관리를 위한 명령 함수를 제공하며, 실시간 태스크의 실행 동기화를 위한 응답 대기 함수를 제공하고 있다.

[0080] 이를 통해 자바 영역과 cpp 영역을 연결하여 안드로이드 어플리케이션에서도 모듈을 제어할 수 있도록 할 수 있게 된다.

[0081] 도 3은 본 발명의 일 실시예에 따른 안드로이드의 실시간 태스크 관리 장치의 런타임 처리부를 나타낸 블록도이다.

[0082] 가비지 컬렉션으로 인한 중단 시간을 최소화 하고자 실행 제어를 통해 가비지 컬렉션이 실행되지 않으면, 힙 메모리에 가비지가 남아있게 되고 해당 메모리 공간은 다른 객체에 의해 사용될 수 없다. 이에 따라 메모리 할당이 요청될 때마다 가용 메모리가 부족한 현상이 빈번하게 발생하며, 힙 메모리의 크기가 지속적으로 증가한다. 그렇기 때문에 가비지 컬렉션 실행이 비활성화 되는 시간은 실시간 태스크가 실행되는 특정 시간으로 한정시켜야 한다. 그 이외의 상황에서는 가비지 컬렉션이 실행될 수 있도록 하는 것이 필요하다.

[0083] 본 발명의 일 실시예에 따른 안드로이드의 실시간 태스크 관리 장치의 가비지 컬렉션 실행 제어 구조는 안드로이드의 달빅 가상머신을 수정한 것으로, 달빅 가상머신에서 가용 메모리 확보를 위해 가비지 컬렉션을 실행해야 하는 시점에서 특정 플래그 값에 따라 그 실행 여부가 결정된다.

[0084] 안드로이드 어플리케이션은 리눅스 프로세스에 대응되며, 각자 자신만의 달빅 가상머신 인스턴스 상에서 실행된다. 달빅 가상머신은 Zygote 프로세스에 의해 생성 및 초기화되어 구동된다. Zygote 프로세스는 어플리케이션의 실행에 필요한 코드 및 메모리 정보들을 미리 메모리에 적재 해두었다가 새로운 어플리케이션이 실행될 때 빠르게 실행시키는 역할을 한다.

[0085] 도 3을 참조하면, 런타임 처리부(400)는 리눅스 파일 처리부(410), 달빅 가상머신(Dalvik VM)(430)을 포함한다.

[0086] 런타임 처리부(400)는 달빅 가상머신(Dalvik VM)(430)을 이용하여, 메모리 관리와 멀티쓰레딩(multithreading)을 수행한다.

[0087] 달빅 가상머신(Dalvik VM)은, 가비지 컬렉션의 수행을 제어하는 가비지 컬렉션 제어부(600)를 포함한다.

[0088] 가비지 컬렉션의 동작 방식은, 동시 마크 스윕(Concurrent Mark Sweep, CMS) 알고리즘을 이용하여 구현된다.

[0089] 가비지 컬렉션 제어부(600)는, 달빅 가상머신에서 가용 메모리 확보를 위해 상기 가비지 컬렉션을 실행해야 하는 시점에서 특정 플래그 값에 따라 실행 여부를 결정한다.

[0090] 리눅스 파일 처리부(410)는 달빅 가상머신(Dalvik VM)의 가비지 컬렉션 실행 여부를 저장하는 저장 파일을 포함한다.

[0091] 달빅 가상머신(Dalvik VM)(430)은, 코드 상에서 저장 파일에 접근가능하다.

[0092] 도 4는 본 발명의 일 실시예에 따른 안드로이드의 실시간 태스크 관리 장치의 가비지 컬렉션 제어부를 나타낸 도면이다.

[0093] 달빅 가상머신(Dalvik VM)은, 가비지 컬렉션의 수행을 제어하는 가비지 컬렉션 제어부(600)를 포함한다.

[0094] 가비지 컬렉션 제어부(600)는, 달빅 가상머신에서 가용 메모리 확보를 위해 상기 가비지 컬렉션을 실행해야 하는 시점에서 특정 플래그 값에 따라 실행 여부를 결정한다.

[0095] 도 4를 참조하면, 가비지 컬렉션 제어부(600)는 리드(read)부(610), 플래그 변수 설정부(620), 플래그 변수 확인부(630), 가비지 컬렉션 수행 판단부(640)를 포함한다.

[0096] 리드(read)부(610)는 달빅 가상머신(Dalvik VM)이 초기화 되는 과정에서, 상기 저장 파일의 데이터를 읽어온다.

[0097] 플래그 변수 설정부(620)는 리드(read)부에서 읽어온 값을 상기 플래그 변수로 설정한다.

[0098] 플래그 변수 확인부(630)는 가비지 컬렉션이 필요한 시점에서 설정된 상기 플래그 변수 값을 확인한다.

[0099] 가비지 컬렉션 수행 판단부(640)는 플래그 변수 값에 따라 상기 가비지 컬렉션 수행을 판단한다. 구체적으로, 플래그 변수 값이 1인 경우 상기 가비지 컬렉션을 실행하지 않고, 0인 경우 상기 가비지 컬렉션을 실행한다.

[0100] 도 5는 본 발명의 일 실시예에 따른 안드로이드의 실시간 태스크 관리 방법을 나타낸 흐름도이다.

[0101] 도 5를 참조하면, 가상머신을 이용하여, 메모리 관리와 멀티쓰레딩(multithreading)을 수행하는 안드로이드의 실시간 태스크 관리 방법은

[0102] 리눅스 파일 처리부가 상기 가상머신의 가비지 컬렉션 실행 여부를 저장하는 저장 파일을 포함하는 단계(S100)에서 시작한다.

[0103] 단계 S200에서 가비지 컬렉션 제어부가 상기 가상머신에서 가용 메모리 확보를 위해 가비지 컬렉션을 실행해야 하는 시점에서 특정 플래그 값에 따라 실행 여부를 결정한다.

[0104] 구체적으로, 특정 플래그 값에 따라 실행 여부를 결정하는 단계(S200)는,

[0105] 단계 S210에서 리드(read)부가 상기 가상머신이 초기화 되는 과정에서, 상기 저장 파일의 데이터를 읽어온다.

[0106] 단계 S220에서 플래그 변수 설정부가 상기 리드(read)부에서 읽어온 값을 상기 플래그 변수로 설정한다.

[0107] 단계 S230에서 플래그 변수 확인부가 상기 가비지 컬렉션이 필요한 시점에서 설정된 상기 플래그 변수 값을 확인한다.

[0108] 단계 S240에서 가비지 컬렉션 수행 판단부가 상기 플래그 변수 값에 따라 상기 가비지 컬렉션 수행을 판단한다,

[0109] 가비지 컬렉션 수행을 판단하는 단계(S240)는, 플래그 변수 값이 1인 경우 상기 가비지 컬렉션을 실행하지 않고, 0인 경우 상기 가비지 컬렉션을 실행하며, 안드로이드의 실시간 태스크 관리 방법은 종료된다.

[0110] 도 6은 본 발명의 일 실시예에 따른 안드로이드의 실시간 태스크 관리 방법을 나타낸 도면이다.

[0111] Zygote 달빅 가상머신에 의해 시스템에 존재하는 모든 프로세스의 달빅 가상머신 인스턴스가 생성되고 초기화되며, 이 과정에서 가비지 컬렉션의 실행 여부가 결정되도록 하였다. 이를 위해 가비지 컬렉션 실행 여부를 저장하고 있는 파일을 하나 두었으며, 달빅 가상머신 코드 상에서 파일에 접근할 수 있도록 하기 위해 리눅스 파일 시스템에 위치시켰다. 달빅 가상머신이 초기화 되는 과정에서 해당 파일에서 값을 읽어와 GC_DISABLED라는 플래그 변수에 설정한다. 이와 같은 과정을 통해 안드로이드 시스템 내에 존재하는 모든 달빅 가상머신의 가비지 컬렉션 실행이 제어된다.

[0112] 도 6은 달빅 가상머신이 초기화될 때 가비지 컬렉션 실행 여부에 대한 플래그 값이 설정된 후, 가비지 컬렉션 실행 여부가 결정되는 과정을 나타낸다. 가비지 컬렉션이 필요한 시점에서 GC_DISABLED 변수 값에 따라 실행 여부가 결정된다. GC_DISABLED 변수는 달빅 가상머신의 가비지 컬렉션 실행 여부를 나타내는 변수로, 값이 1인 경우 가비지 컬렉션을 실행하지 않고, 0인 경우 실행한다는 의미를 가진다.

[0113] 달빅 가상머신은 가비지 컬렉션이 필요한 시점에서 dvmCollectGarbageInternal 함수를 호출하여 도달할 수 없는 객체를 가비지로 식별하고, 가비지가 차지하고 있는 메모리 공간을 회수한다.

[0114] 기존의 달빅 가상머신 코드에 가비지 컬렉션의 실행을 제어하기 위한 함수를 구현하였으며, 표 3은 가비지 컬렉션의 실행을 제어하기 위한 함수 목록을 나타낸 것이다.

표 3

함수명	설명
getGCstatus	GC_DISABLED 변수의 getter 함수
setGCstatus	GC_DISABLED 변수의 setter 함수
dvmGcExecSetup	파일에 값을 기록하는 함수
dvmGcExecGetValue	파일에서 값을 읽어 GC_DISABLED 변수 설정하는 함수

[0115]

[0116] 여기서, getGCstatus() 함수는 GC_DISABLED 변수에 대한 getter 함수로, 변수 값을 리턴하며, setGCstatus()함수는 GC_DISABLED 변수의 setter 함수이다. 해당 변수 값은 파일에서 값을 읽어서 설정되기 때문에, 인자로 전달 받은 값을 파일에 기록하기 위해 dvmGcExecSetup함수를 호출하는 작업을 수행한다.

[0117] dvmGcExecSetup 함수는 가비지 컬렉션 실행 여부를 파일에 설정하는 함수로, 인자로 전달 받은 값을 파일에 기록한다.

[0118] dvmExecGetValue 함수는 파일에서 값을 읽어서 GC_DISABLED 변수에 해당 값을 설정하는 함수이다.

[0119] 도 7은 본 발명의 일 실시예에 따른 호출 주기의 변화를 나타낸 그래프이다.

[0120] 도 7은 본 발명의 일 실시예에 따른 안드로이드의 실시간 태스크 관리 장치 및 관리 방법의 가비지 컬렉션 실행 제어에 따른 실시간 성능 차이를 확인하기 위한 것으로, 도 7의 (a)는 제어 방법이 적용되지 않은 환경이고, 도 7의 (b)는 가비지 컬렉션이 적용된 상황이며 두 상황에서의 실험을 통해 나타난 결과이다.

[0121] 실험은 5ms 주기를 가지는 태스크의 주기성 만족을 확인하기 위해 가비지 컬렉션 실행 제어를 통한 비활성화와 활성화 시 태스크 호출 주기의 값을 비교한다.

[0122] 도 7의 (a)는 가비지 컬렉션 제어를 하지 않아 가비지 컬렉터가 활성화 되어 있을 때의 태스크의 호출 주기를 측정한 실험 결과이다.

[0123] 가비지 컬렉션을 제어하기 이전의 데이터는 10ms 이상을 벗어나는 횟수가 386회로 7.72 %를 보이고 있으며, 4.5 ms 이하로 벗어나는 호출 주기는 2120 회, 42.41%로 나타나고 있다. 반면 10% 오차 범위 이내인 4.5 ms ~ 5.5ms 주기 범위는 1715 회인 34.31 %를 나타낸다. 이런 실험 결과를 통해 가비지 컬렉션을 활성화 하게 되면 실시간성을 전혀 만족할 수 없게 된다는 것을 보이고 있다.

[0124] 도 7의 (b)는 가비지 컬렉션 제어를 통해 실시간성이 요구되는 시간동안 가비지 컬렉터가 비활성화 되어 있을 때의 태스크 호출 주기를 측정한 실험 결과이다.

[0125] 가비지 컬렉션을 비활성화 하여 기존에 발생하던 10ms를 초과하는 태스크 호출 주기는 1회로 감소하였으며, 그 외에 호출 주기를 보이는 데이터 또한 데이터의 오차범위 10% 이내인 4.5 ms ~ 5.5 ms 주기 범위로 나타내었을 때, 유효 횟수가 4908회인 98.18 %로 나타났다. 이를 통해서 기존 가비지 컬렉터를 활성화 할 때보다 크게 감소하는 것으로 이런 가비지 컬렉터를 제어하는 것이 실시간성을 만족하도록 할 수 있는 것을 나타낸다.

[0126] 도 8 및 도 9는 본 발명의 일 실시예에 따른 힙 크기의 변화를 나타낸 그래프이다.

[0127] 도 8 및 도 9는 본 발명의 일 실시예에 따른 안드로이드의 실시간 태스크 관리 장치 및 관리 방법의 가비지 컬렉션 실행 제어에 따른 실시간 성능 차이를 확인하기 위한 것으로, 10ms 주기의 UDP 소켓 통신이 30분간 이루어지는 동안 시간에 따른 힙 메모리 크기를 측정한 것이다.

[0128] 도 8의 (a), (b)는 ArrayList를 사용한 경우 측정한 데이터를 그래프로 나타낸 것이고, 도 9의 (a), (b)는 은 배열을 사용한 경우의 그래프이다. 측정한 데이터 x축은 힙 메모리가 증가한 시점을 나타내며, y축은 MB 단위의 힙 메모리 크기이다.

[0129] 데이터 수신을 위해 ArrayList와 배열을 사용한 경우 모두 실행 제어를 통해 가비지 컬렉션이 발생하지 않는 상황에서는 가용 메모리가 충분히 확보되지 않기 때문에, 지속적으로 힙의 크기가 증가한다. 그러나 30분간의 통신이 이루어지는 동안 가비지 컬렉션이 실행되지 않는 것으로 인한 메모리 관련 오류는 발생하지 않는 것을 확

인하였다. 가비지 컬렉션이 실행되는 상황에서는 할당에 필요한 가용 메모리 확보를 위해 가비지 컬렉션이 일어나게 되고, 이에 따라 힙 메모리가 비교적 크게 증가하지 않는다.

[0130] 이를 통해 안드로이드에 실시간성을 제공하기 위해서 가비지 컬렉션 제어를 사용하여 실시간 태스크의 주기를 만족할 수 있었으며, 또한 이때 메모리 접근 시간을 줄일 수 있어서 실시간 태스크 성능에 거의 영향이 없도록 할 수 있음을 확인할 수 있다.

[0131] 이상의 설명은 본 발명의 일 실시예에 불과할 뿐, 본 발명이 속하는 기술 분야에서 통상의 지식을 가진 자는 본 발명의 본질적 특성에서 벗어나지 않는 범위에서 변형된 형태로 구현할 수 있을 것이다. 따라서 본 발명의 범위는 전술한 실시예에 한정되지 않고 특허 청구 범위에 기재된 내용과 동등한 범위 내에 있는 다양한 실시 형태가 포함되도록 해석되어야 할 것이다.

부호의 설명

[0132] 10: 안드로이드의 실시간 태스크 관리 장치

100: 어플리케이션 처리부

200: 어플리케이션 프레임워크 처리부

300: 라이브러리 처리부

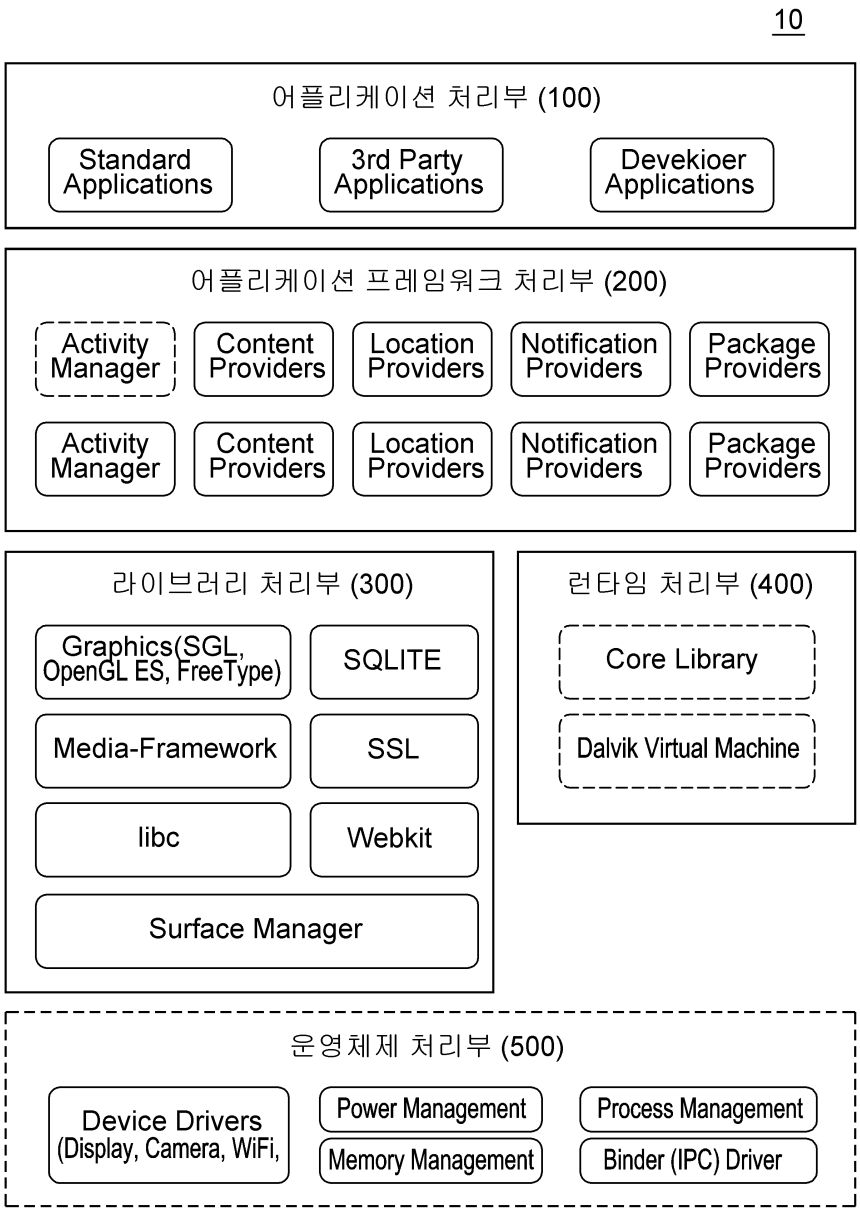
400: 런타임 처리부

500: 운영체제 처리부

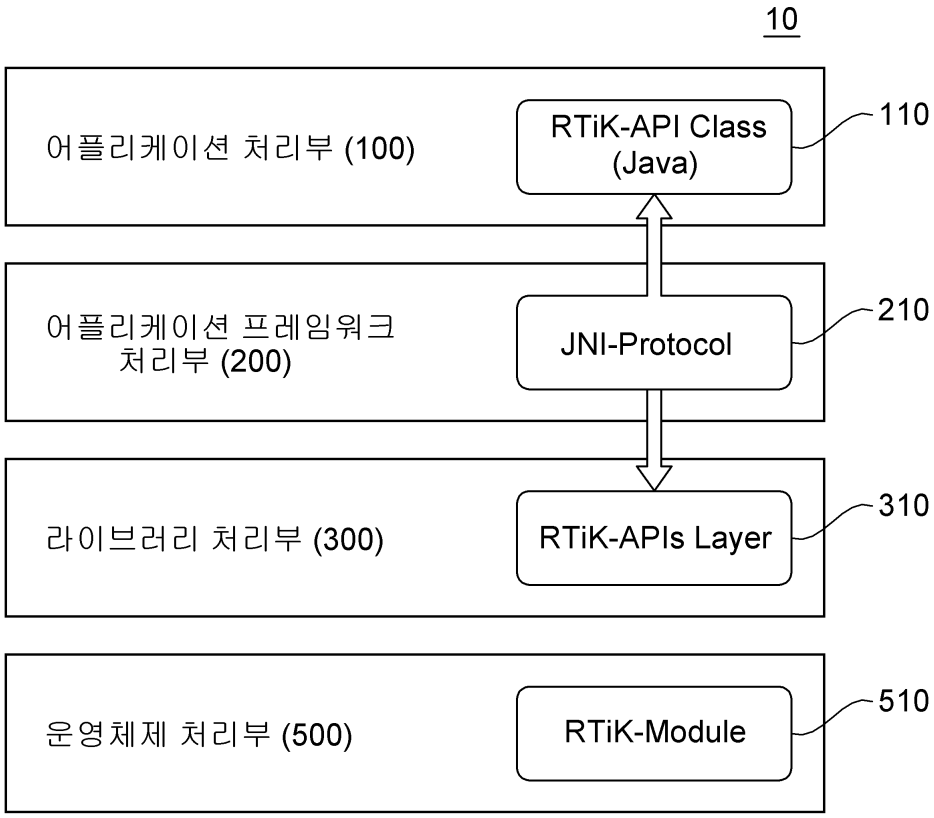
600: 가비지 컬렉션 제어부

도면

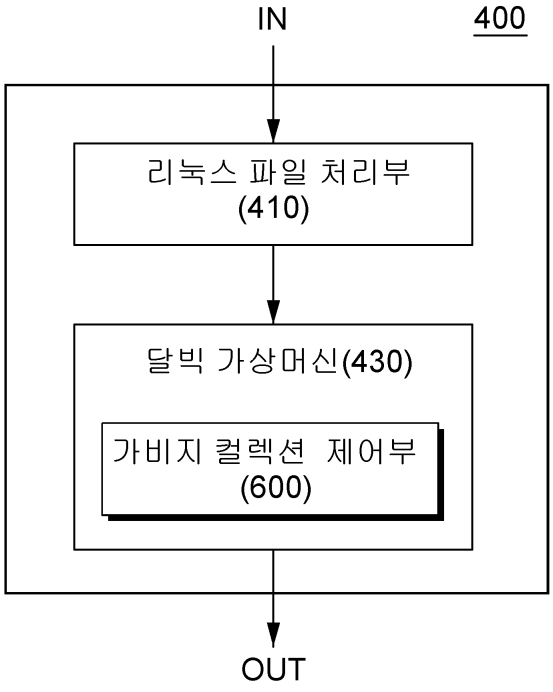
도면1


10
어플리케이션 처리부 (100)
Standard Applications
3rd Party Applications
Devekioer Applications
어플리케이션 프레임워크 처리부 (200)
Activity Manager
Content Providers
Location Providers
Notification Providers
Package Providers
Activity Manager
Content Providers
Location Providers
Notification Providers
Package Providers
라이브러리 처리부 (300)
Graphics(SGL, OpenGL ES, FreeType)
SQLITE
Media-Framework
SSL
libc
Webkit
Surface Manager
런타임 처리부 (400)
Core Library
Dalvik Virtual Machine
운영체제 처리부 (500)
Device Drivers (Display, Camera, WiFi,
Power Management
Memory Management
Process Management
Binder (IPC) Driver

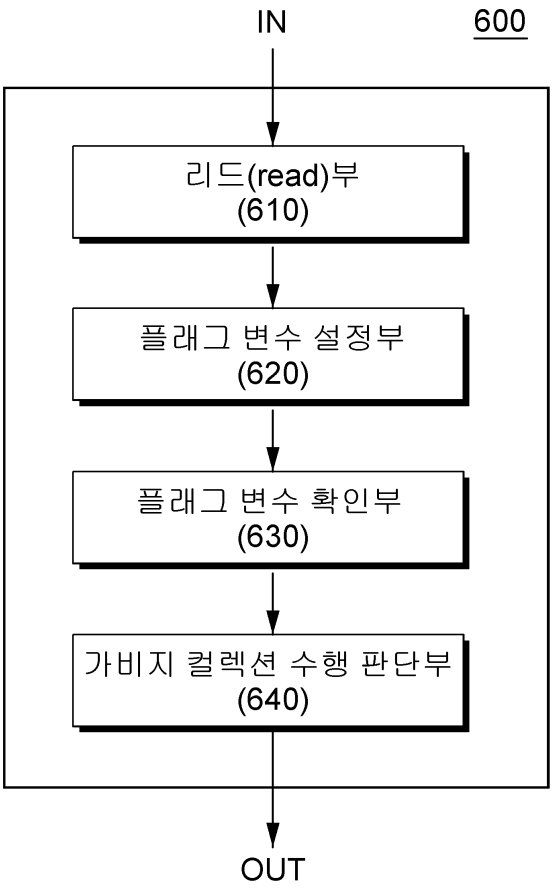
도면2


10
어플리케이션 처리부 (100)
RTiK-API Class (Java)
110
어플리케이션 프레임워크 처리부 (200)
JNI-Protocol
210
라이브러리 처리부 (300)
RTiK-APIs Layer
310
운영체제 처리부 (500)
RTiK-Module
510

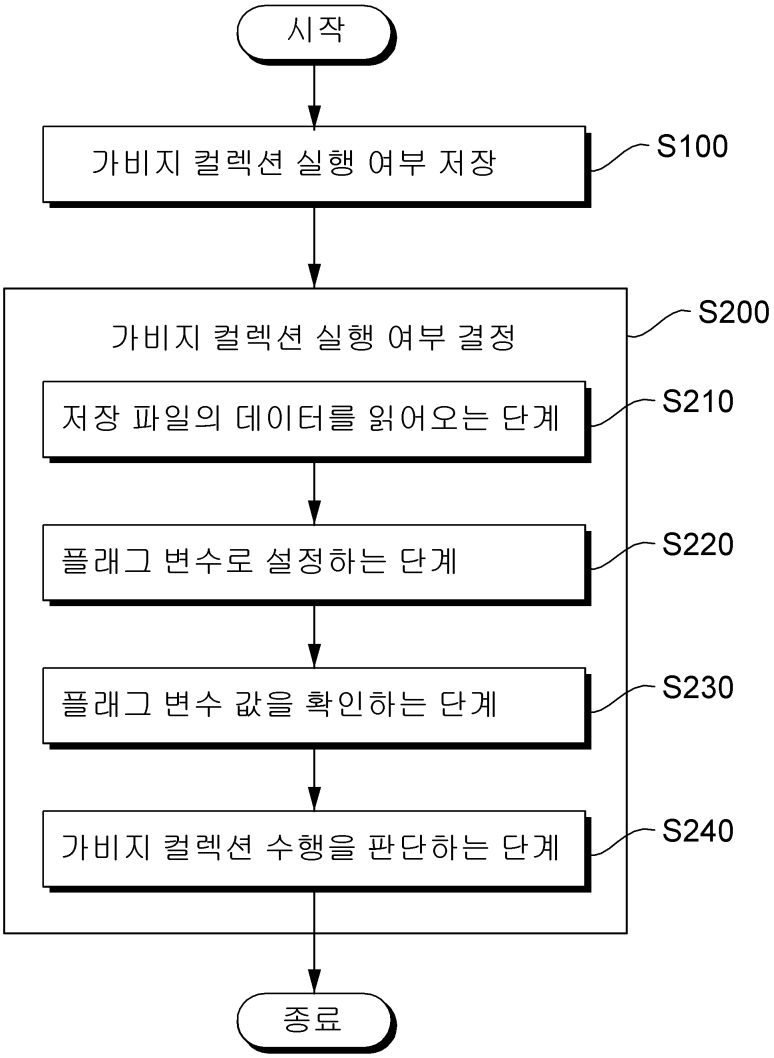
도면3


IN
400
리눅스 파일 처리부 (410)
달빅 가상머신(430)
가비지 컬렉션 제어부 (600)
OUT

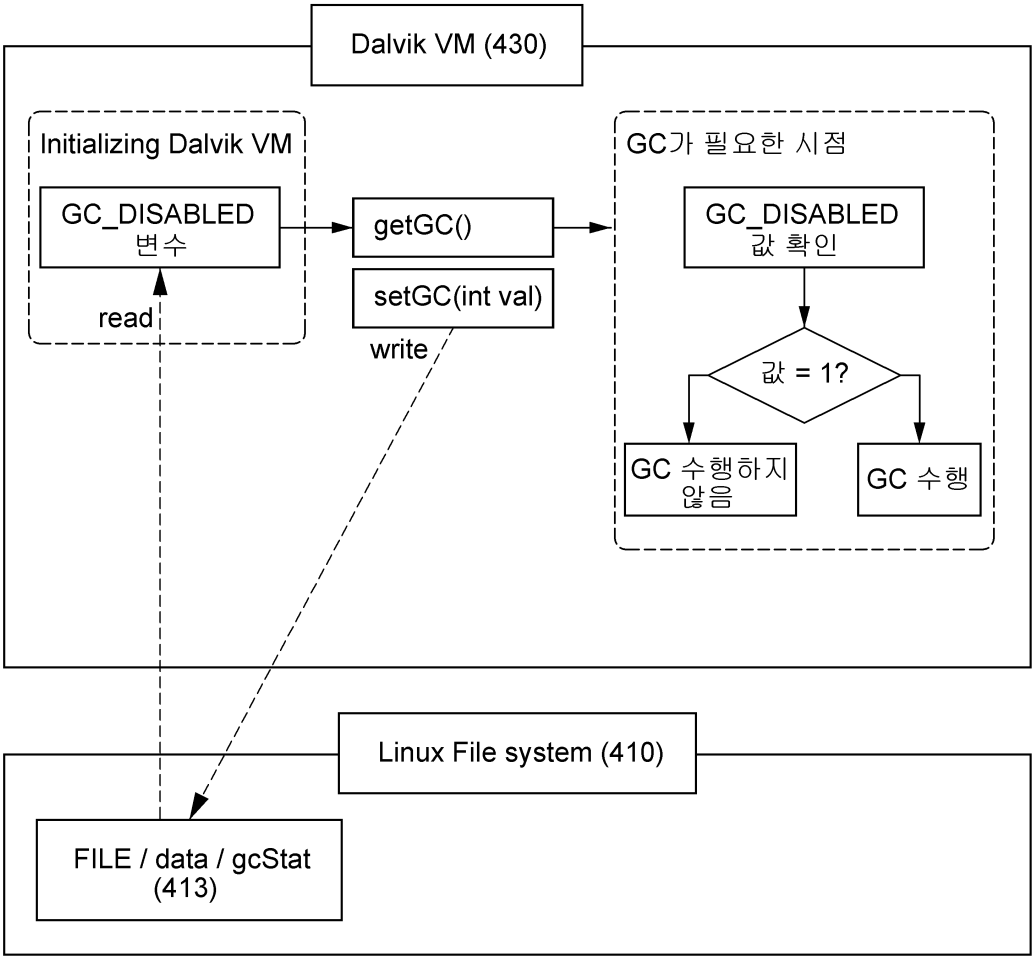
도면4


IN
600
리드(read)부
(610)
플래그 변수 설정부
(620)
플래그 변수 확인부
(630)
가비지 컬렉션 수행 판단부
(640)
OUT

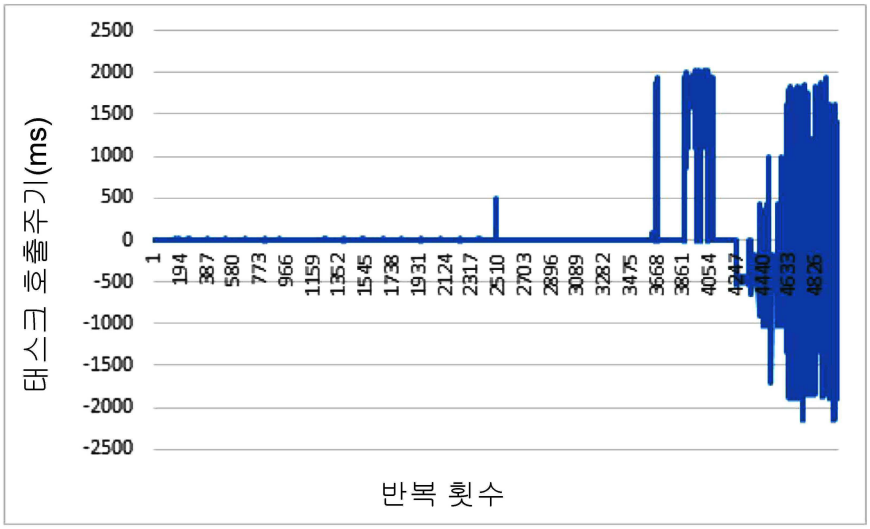
도면5


시작
가비지 컬렉션 실행 여부 저장
S100
가비지 컬렉션 실행 여부 결정
S200
저장 파일의 데이터를 읽어오는 단계
S210
플래그 변수로 설정하는 단계
S220
플래그 변수 값을 확인하는 단계
S230
가비지 컬렉션 수행을 판단하는 단계
S240
종료

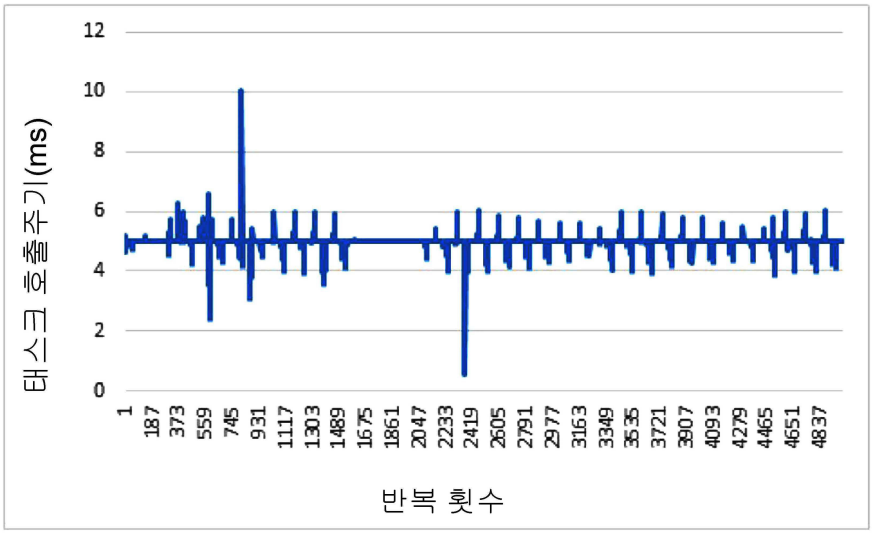
도면6


Dalvik VM (430)
Initializing Dalvik VM
GC_DISABLED
변수
read
getGC()
setGC(int val)
write
GC가 필요한 시점
GC_DISABLED
값 확인
값 = 1?
GC 수행하지
않음
GC 수행
Linux File system (410)
FILE / data / gcStat
(413)

도면7

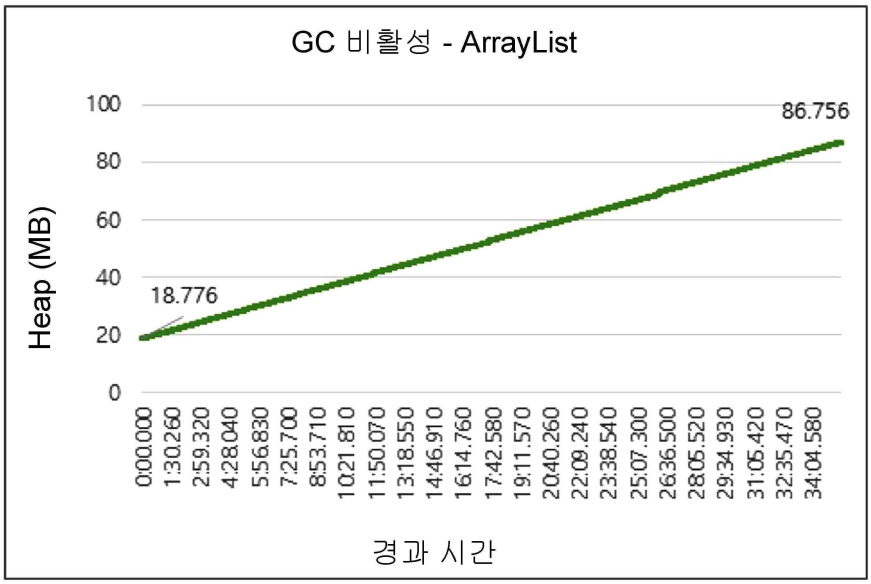

2500
2000
1500
1000
500
0
-500
-1000
-1500
-2000
-2500
1
194
387
580
773
966
1159
1352
1545
1738
1931
2124
2317
2510
2703
2896
3089
3282
3475
3668
3861
4054
4247
4440
4633
4826
태스크 호출주기(ms)
반복 횟수

(a)

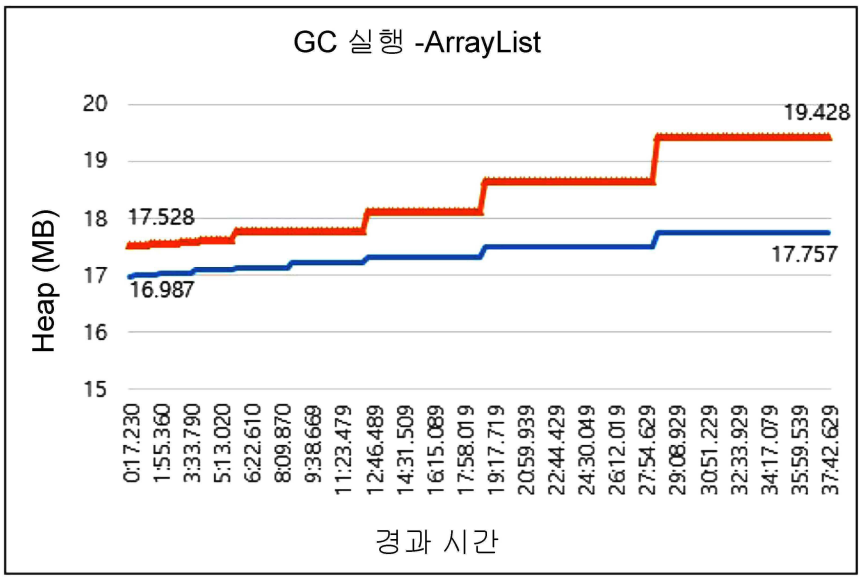

12
10
8
6
4
2
0
1
187
373
559
745
931
1117
1303
1489
1675
1861
2047
2233
2419
2605
2791
2977
3163
3349
3535
3721
3907
4093
4279
4465
4651
4837
태스크 호출주기(ms)
반복 횟수

(b)

도면8

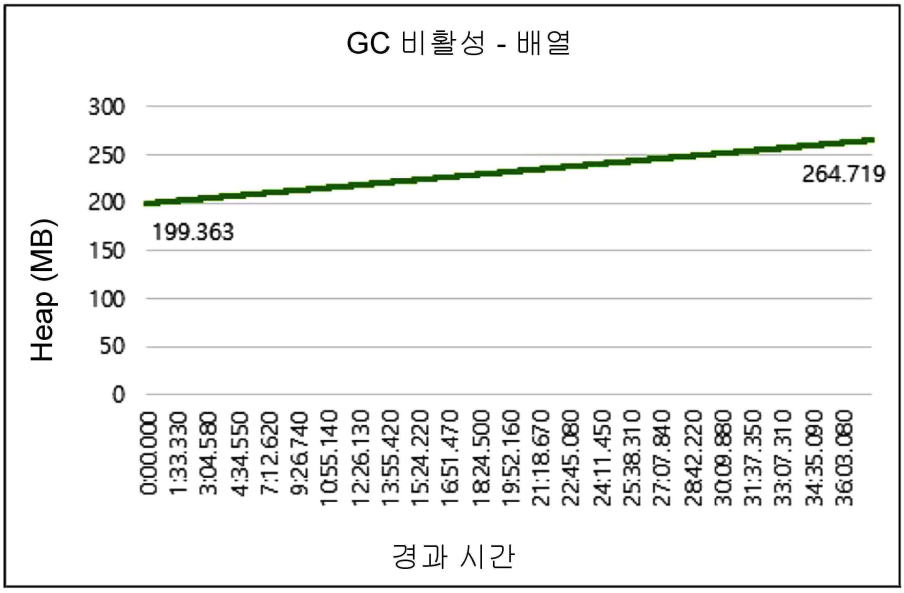

GC 비활성 - ArrayList
Heap (MB)
100
80
60
40
20
0
18.776
86.756
0:00.000
1:30.260
2:59.320
4:28.040
5:56.830
7:25.700
8:53.710
10:21.810
11:50.070
13:18.550
14:46.910
16:14.760
17:42.580
19:11.570
20:40.260
22:09.240
23:38.540
25:07.300
26:36.500
28:05.520
29:34.930
31:05.420
32:35.470
34:04.580
경과 시간

(a)

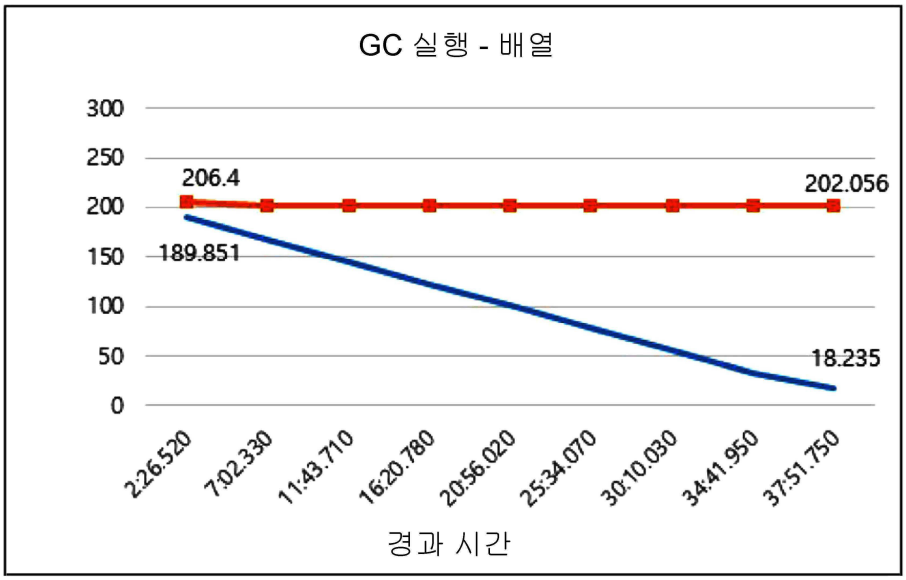

GC 실행 -ArrayList
Heap (MB)
20
19
18
17
16
15
17.528
16.987
19.428
17.757
0:17.230
1:55.360
3:33.790
5:13.020
6:22.610
8:09.870
9:38.669
11:23.479
12:46.489
14:31.509
16:15.089
17:58.019
19:17.719
20:59.939
22:44.429
24:30.049
26:12.019
27:54.629
29:08.929
30:51.229
32:33.929
34:17.079
35:59.539
37:42.629
경과 시간

(b)

도면9


GC 비활성 - 배열
Heap (MB)
300
250
200
150
100
50
0
264.719
199.363
0:00.000
1:33.330
3:04.580
4:34.550
7:12.620
9:26.740
10:55.140
12:26.130
13:55.420
15:24.220
16:51.470
18:24.500
19:52.160
21:18.670
22:45.080
24:11.450
25:38.310
27:07.840
28:42.220
30:09.880
31:37.350
33:07.310
34:35.090
36:03.080
경과 시간

(a)


GC 실행 - 배열
300
250
200
150
100
50
0
206.4
202.056
189.851
18.235
2:26.520
7:02.330
11:43.710
16:20.780
20:56.020
25:34.070
30:10.030
34:41.950
37:51.750
경과 시간

(b)