(19) 대한민국특허청(KR)
(12) 공개특허공보(A)

(11) 공개번호 10-2019-0050581
(43) 공개일자 2019년05월13일

(51) 국제특허분류(Int. Cl.)
H04L 29/08 (2006.01)
(52) CPC특허분류
H04L 67/1074 (2013.01)
H04L 67/12 (2013.01)
(21) 출원번호 10-2017-0146121
(22) 출원일자 2017년11월03일
심사청구일자 2017년11월03일

(71) 출원인
한림대학교 산학협력단
강원도 춘천시 한림대학길 1, 한림대학교(옥천동)
(72) 발명자
김의직
강원도 춘천시 지석로 29, 514동 1103호(석사동, 휴먼타운아파트)
권정혁
강원도 춘천시 스무숲1길 32-9, 202호
(74) 대리인
김남혁

전체 청구항 수 : 총 16 항

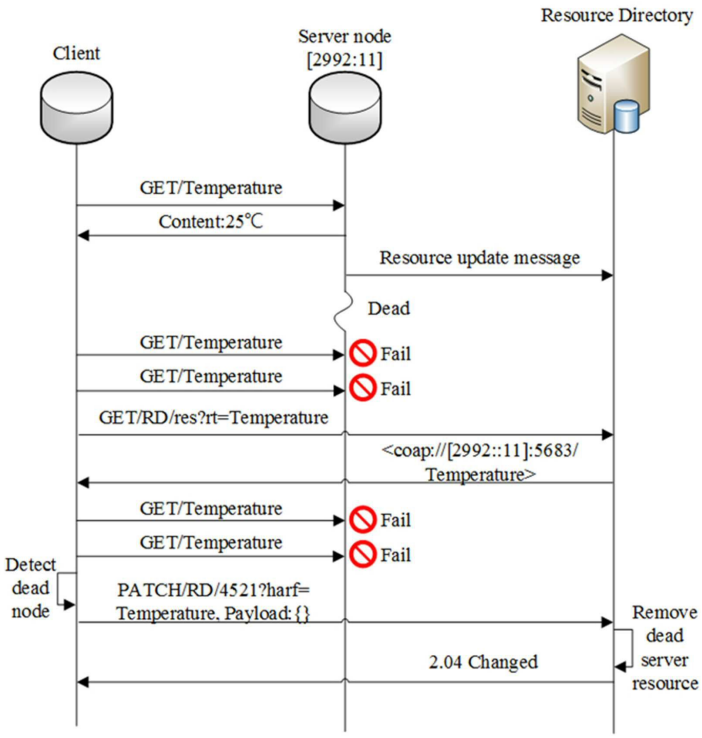
(54) 발명의 명칭 **사물 인터넷을 이용하는 통신시스템에서의 효율적인 에너지 소비 방법 및 이 방법이 적용된 기기**

(57) 요 약

일 실시 예에 따른 사물 인터넷을 사용하기 위해 이용되는 사용자 기기, 서버, 자원 디렉토리를 포함하는 통신시스템은, 상기 사용자 기기가 상기 서버에 자원 요청 메시지를 송신하는 단계; 상기 서버로부터 상기 자원 요청 메시지에 대한 응답 메시지의 유무를 기초로 상기 서버의 동작 여부를 판단하는 단계; 상기 서버가 동작하지 않

(뒷면에 계속)

대 표 도 - 도3



는다고 판단된 경우, 자원 디렉토리에 자원 삭제를 요청하는 메시지를 송신하는 단계; 상기 서버가 동작한다고 판단된 경우, 상기 서버는 자원 업데이트 메시지를 송신하는 주기를 결정하는 단계; 상기 결정된 주기에 기초하여 상기 자원 업데이트 메시지를 자원 디렉토리에 송신하는 단계를 포함하되, 상기 서버의 동작 여부를 판단하는 단계는, 상기 사용자 기기가 상기 서버에 기설정된 횟수만큼 상기 자원 요청 메시지를 송신하여도 상기 응답 메시지를 수신하지 못한 경우, 상기 서버가 동작하지 않는 것으로 판단하는 것을 포함하고, 상기 결정된 주기는 상기 서버의 에너지 수준에 기초하여 적응적으로 결정되는 것을 특징으로 한다.

이 발명을 지원한 국가연구개발사업

과제고유번호	2016H1D5A1910427
부처명	과학기술정보통신부
연구관리전문기관	한국연구재단
연구사업명	지역신산업선도인력양성사업
연구과제명	실시간 상황정보 빅데이터 기반의 개인 맞춤형 헬스케어 시스템 개발
기 여 율	1/1
주관기관	한림대학교 산학협력단
연구기간	2017.06.01 ~ 2018.03.31

명 세 서

청구범위

청구항 1

사물 인터넷을 사용하기 위해 이용되는 사용자 기기, 서버, 자원 디렉토리를 포함하는 통신시스템에서,

상기 사용자 기기가 상기 서버에 자원 요청 메시지를 송신하는 단계;

상기 서버로부터 상기 자원 요청 메시지에 대한 응답 메시지의 유무를 기초로 상기 서버의 동작 여부를 판단하는 단계;

상기 서버가 동작하지 않는다고 판단된 경우, 자원 디렉토리에 자원 삭제를 요청하는 메시지를 송신하는 단계;

상기 서버가 동작한다고 판단된 경우, 상기 서버는 자원 업데이트 메시지를 송신하는 주기를 결정하는 단계;

상기 결정된 주기에 기초하여 상기 자원 업데이트 메시지를 자원 디렉토리에 송신하는 단계를 포함하되,

상기 서버의 동작 여부를 판단하는 단계는, 상기 사용자 기기가 상기 서버에 기설정된 횟수만큼 상기 자원 요청 메시지를 송신하여도 상기 응답 메시지를 수신하지 못한 경우, 상기 서버가 동작하지 않는 것으로 판단하는 것을 포함하고,

상기 결정된 주기는 상기 서버의 에너지 수준에 기초하여 적응적으로 결정되는 것을 특징으로 하는 사물 인터넷을 사용하기 위해 이용되는 사용자 기기, 서버, 자원 디렉토리를 포함하는 통신시스템

청구항 2

제1항에 있어서,

상기 사용자 기기가 상기 자원 디렉토리로부터 자원 삭제 성공 메시지를 수신하는 단계를 더 포함하는 사물 인터넷을 사용하기 위해 이용되는 사용자 기기, 서버, 자원 디렉토리를 포함하는 통신시스템

청구항 3

제1항에 있어서,

상기 기설정된 횟수는 2회인 것을 특징으로 하는 사물 인터넷을 사용하기 위해 이용되는 사용자 기기, 서버, 자원 디렉토리를 포함하는 통신시스템

청구항 4

제1항에 있어서,

상기 자원 업데이트 주기는 초기 자원 업데이트 주기, 곱셈 계수 및 남은 에너지 수준에 기초하여 결정되는 것을 특징으로 하는 사물 인터넷을 사용하기 위해 이용되는 사용자 기기, 서버, 자원 디렉토리를 포함하는 통신시스템

청구항 5

제4항에 있어서,

상기 자원 업데이트 주기는 다음과 같은 수식에 기초하여 결정되고,

$$UP_{new} = \{UP_{initial} \times (k-1) \times (1-REL)\} + UP_{initial}$$

여기서 UP_{new}는 결정된 자원 업데이트 주기,

$UP_{initial}$은 초기 자원 업데이트 주기,

k는 곱셈 계수,

REL은 남은 에너지 수준을 나타내는 것을 특징으로 하는 사물 인터넷을 사용하기 위해 이용되는 사용자 기기, 서버, 자원 디렉토리를 포함하는 통신시스템

청구항 6

사물 인터넷을 사용하기 위해 이용되는 서버에서,

자원 업데이트 메시지를 송신하는 자원 업데이트 주기를 결정하는 단계;

상기 결정된 자원 업데이트 주기에 기초하여 상기 자원 업데이트 메시지를 자원 디렉토리에 송신하는 단계를 포함하되,

상기 결정된 자원 업데이트 주기는 상기 서버의 에너지 수준에 기초하여 적응적으로 결정되는 것을 특징으로 하는 사물 인터넷을 사용하기 위해 이용되는 서버의 동작방법

청구항 7

제6항에 있어서,

상기 자원 업데이트 주기는 초기 자원 업데이트 주기, 곱셈 계수 및 남은 에너지 수준에 기초하여 결정되는 것을 특징으로 하는 사물 인터넷을 사용하기 위해 이용되는 서버의 동작방법

청구항 8

제7항에 있어서,

상기 자원 업데이트 주기는 다음과 같은 수식에 기초하여 결정되고,

$UP_{new} = \{UP_{initial} \times (k-1) \times (1-REL)\} + UP_{initial}$

$UP_{initial}$은 초기 자원 업데이트 주기

k는 곱셈 계수,

REL은 남은 에너지 수준을 나타내는 것을 특징으로 하는 사물 인터넷을 사용하기 위해 이용되는 서버의 동작방법

청구항 9

사물 인터넷을 사용하는 사용자 기기에서

서버에 자원 요청 메시지를 송신하는 단계;

상기 서버로부터 상기 자원 요청 메시지에 대한 응답 메시지의 유무를 기초로 상기 서버의 동작 여부를 판단하는 단계;

상기 서버가 동작하지 않는다고 판단된 경우, 자원 디렉토리에 자원 삭제를 요청하는 메시지를 송신하는 단계를 포함하되,

상기 서버의 동작 여부를 판단하는 단계는, 상기 사용자 기기가 상기 서버에 기설정된 횟수만큼 상기 자원 요청 메시지를 송신하여도 상기 응답 메시지를 수신하지 못한 경우, 상기 서버가 동작하지 않는 것으로 판단하는 것을 포함하는, 사물 인터넷을 사용하기 위해 이용되는 사용자 기기의 동작 방법

청구항 10

제9항에 있어서,

상기 자원 디렉토리로부터 자원 삭제 성공 메시지를 수신하는 단계를 더 포함하는 사물 인터넷을 사용하기 위해 이용되는 사용자 기기의 동작 방법

청구항 11

제9항에 있어서,

상기 기설정된 횟수는 2회인 것을 특징으로 하는 사물 인터넷을 사용하기 위해 이용되는 사용자 기기의 동작 방법

청구항 12

사물 인터넷을 사용하기 위해 이용되는 서버는,

통신부;

사용자 단말의 동작을 위한 프로그램을 저장하는 메모리; 및

상기 프로그램을 실행하는 프로세서를 포함하되,

상기 프로세서는, 자원 업데이트 메시지를 송신하는 자원 업데이트 주기를 결정하고, 상기 결정된 자원 업데이트 주기에 기초하여 상기 자원 업데이트 메시지를 자원 디렉토리에 송신하되,

상기 결정된 자원 업데이트 주기는 상기 서버의 에너지 수준에 기초하여 적응적으로 결정되는 것을 특징으로 하는 사물 인터넷을 사용하기 위해 이용되는 서버

청구항 13

제12항에 있어서,

상기 프로세서는,

상기 자원 업데이트 주기는 초기 자원 업데이트 주기, 곱셈 계수 및 남은 에너지 수준에 기초하여 결정되는 것을 특징으로 하는 사물 인터넷을 사용하기 위해 이용되는 서버

청구항 14

제13항에 있어서,

상기 자원 업데이트 주기는 다음과 같은 수식에 기초하여 결정되고,

$UP_{new} = \{UP_{initial} \times (k-1) \times (1-REL)\} + UP_{initial}$

$UP_{initial}$은 초기 자원 업데이트 주기

k는 곱셈 계수,

REL은 남은 에너지 수준을 나타내는 것을 특징으로 하는 사물 인터넷을 사용하기 위해 이용되는 서버

청구항 15

사물 인터넷을 사용하는 사용자 기기에서

통신부;

사용자 단말의 동작을 위한 프로그램을 저장하는 메모리; 및

상기 프로그램을 실행하는 프로세서를 포함하되,

상기 프로세서는, 서버에 자원 요청 메시지를 송신하고, 상기 서버로부터 상기 자원 요청 메시지에 대한 응답 메시지의 유무를 기초로 상기 서버의 동작 여부를 판단하고, 상기 서버가 동작하지 않는다고 판단된 경우, 자원 디렉토리에 자원 삭제를 요청하는 메시지를 송신하되,

상기 프로세서는 상기 서버의 동작 여부를 판단할 때, 상기 사용자 기기가 상기 서버에 기설정된 횟수만큼 상기 자원 요청 메시지를 송신하여도 상기 응답 메시지를 수신하지 못한 경우, 상기 서버가 동작하지 않는 것으로 판단하는, 사물 인터넷을 사용하기 위해 이용되는 사용자 기기

청구항 16

제15항에 있어서,

상기 프로세서는 상기 자원 디렉토리로부터 자원 삭제 성공 메시지를 수신하는 사물 인터넷을 사용하기 위해 이

용되는 사용자 기기

발명의 설명

기 술 분 야

[0001] 본원 발명은 사물 인터넷을 이용하는 통신시스템에서의 효율적인 에너지 소비 방법 및 이 방법이 적용된 기기에 대한 것으로서, 보다 구체적으로 데드 노드 검출 방법 및 업데이트 주기를 적응적으로 조정하는 방법을 통해 사물 인터넷을 이용하는 사용자 단말과 서버가 효율적으로 에너지를 소비하는 방법에 관한 것이다.

배 경 기 술

[0002] 사물 인터넷(Internet of Things; IoT)이란, 각종 사물에 컴퓨터 칩과 통신 기능을 내장하여 인터넷에 연결하는 기술을 의미한다. 사물 인터넷 기기는 사물 인터넷이 적용되는 일반적인 기기(또는 사물)일 수 있다. 예를 들어, 사물 인터넷 기기는, 온도 센서, 습도 센서, 음향 센서, 모션 센서, 근접 센서, 가스 감지 센서, 열 감지 센서 등의 각종 센서, 냉장고, CCTV, TV, 세탁기, 제습기 등의 각종 가전기기, 전등, 화재 경보기, 댁내 기기 등을 포함할 수 있다.

[0003] 사용자는 사물 인터넷 단말을 이용하여 사물 인터넷 기기를 원격으로 제어할 수 있다. 사물 인터넷 단말과 사물 인터넷 기기는 무선 통신, 근거리 통신, 홈 네트워크 등 다양한 통신 기술, 통신 프로토콜 등으로 서로 연결될 수 있다.

[0004] 그런데, 사물 인터넷 시스템에서는 일반적으로 사물 인터넷 기기를 지속적으로 사용하고 있으므로, 효율적인 에너지 사용을 통해 오랜 기간 동안 지속적으로 사물 인터넷 기기를 사용하는 것이 보장되어야 한다. 즉, 사물 인터넷 시스템은 사물 인터넷 기기의 사용을 오랜 기간 동안 지속할 수 있는 에너지 사용 방법이 필요하다.

발명의 내용

해결하려는 과제

[0005] 본 발명은 상기와 같은 문제점을 해결하기 위해 제안된 것으로서, 본 발명의 목적은 사물 인터넷 시스템에서 사물 인터넷 기기인 사용자 기기에서 에너지를 효율적으로 사용하기 위한 것이다.

[0006] 본 발명의 또 다른 목적은 사물 인터넷 시스템에서 사물 인터넷 기기인 사용자 기기에서 에너지를 효율적으로 사용하기 위한 것이다.

[0007] 발명에서 이루고자 하는 기술적 목적들은 이상에서 언급한 사항들로 제한되지 않으며, 언급하지 않은 또 다른 기술적 과제들은 이하 설명할 본 발명의 실시 예들로부터 본 발명이 속하는 기술분야에서 통상의 지식을 가진 자에 의해 고려될 수 있다.

과제의 해결 수단

[0008] 상기 기술적 과제를 해결하기 위한 사물 인터넷을 사용하기 위해 이용되는 사용자 기기, 서버, 자원 디렉토리를 포함하는 통신시스템은 상기 사용자 기기가 상기 서버에 자원 요청 메시지를 송신하는 단계; 상기 서버로부터 상기 자원 요청 메시지에 대한 응답 메시지의 유무를 기초로 상기 서버의 동작 여부를 판단하는 단계; 상기 서버가 동작하지 않는다고 판단된 경우, 자원 디렉토리에 자원 삭제를 요청하는 메시지를 송신하는 단계; 상기 서버가 동작한다고 판단된 경우, 상기 서버는 자원 업데이트 메시지를 송신하는 주기를 결정하는 단계; 상기 결정된 주기에 기초하여 상기 자원 업데이트 메시지를 자원 디렉토리에 송신하는 단계를 포함하되, 상기 서버의 동작 여부를 판단하는 단계는, 상기 사용자 기기가 상기 서버에 기설정된 횟수만큼 상기 자원 요청 메시지를 송신하여도 상기 응답 메시지를 수신하지 못한 경우, 상기 서버가 동작하지 않는 것으로 판단하는 것을 포함하고, 상기 결정된 주기는 상기 서버의 에너지 수준에 기초하여 적응적으로 결정되는 것을 특징으로 한다.

[0009] 상기 통신 시스템은 상기 사용자 기기가 상기 자원 디렉토리로부터 자원 삭제 성공 메시지를 수신하는 단계를 더 포함한다.

[0010] 상기 통신 시스템은 상기 기설정된 횟수는 2회인 것을 특징으로 한다.

[0011] 상기 통신 시스템은 상기 자원 업데이트 주기는 초기 자원 업데이트 주기, 곱셈 계수 및 남은 에너지 수준에 기초하여 결정되는 것을 특징으로 한다.

[0012] 상기 통신 시스템은 상기 자원 업데이트 주기는 다음과 같은 수식에 기초하여 결정되고,

[0013] $UP_{new} = \{UP_{initial} \times (k-1) \times (1-REL)\} + UP_{initial}$

[0014] 여기서, UP_{new}는 결정된 자원 업데이트 주기, $UP_{initial}$은 초기 자원 업데이트 주기, k는 곱셈 계수, REL은 남은 에너지 수준을 나타내는 것을 특징으로 한다.

발명의 효과

[0015] 본원 발명의 실시 예들에 따르면 다음과 같은 효과를 기대할 수 있다.

[0016] 본원 발명에서는 데드 노드로 인한 불필요한 에너지 소비와 업데이트 메시지 전송을 방지하는 CoAP에 대한 ERDU 기법을 제시한다. 서버의 에너지 효율을 향상시키기 위해 ERDU 체계는 1) 데드 노드 검출 및 2) 업데이트 기간 결정의 두 가지 주요 메커니즘을 통해 기능한다. 데드 노드 검출은 ERDU 체계가 작동하지 않는 서버를 검출하고 해당 자원을 RD 항목에서 제거한다. 업데이트 기간 결정은 서버는 잔여 에너지 수준을 기반으로 업데이트 메시지 기간을 조정한다. ERDU의 효과를 확인하기 위해 다양한 시나리오에서 실험 시뮬레이션을 수행한 결과에 따라, 기존 RD 업데이트 메커니즘과 비교하여 ERDU 체계는 클라이언트의 에너지 소비를 87%까지 줄이고 서버의 수명을 315% 늘릴 수 있다.

[0017] 본 발명의 실시 예들에서 얻을 수 있는 효과는 이상에서 언급한 효과들로 제한되지 않으며, 언급하지 않은 또 다른 효과들은 이하의 본 발명의 실시 예들에 대한 기재로부터 본 발명이 속하는 기술분야에서 통상의 지식을 가진 자에게 명확하게 도출되고 이해될 수 있다. 즉, 본 발명을 실시함에 따른 의도하지 않은 효과들 역시 본 발명의 실시 예들로부터 당해 기술분야의 통상의 지식을 가진 자에 의해 도출될 수 있다.

도면의 간단한 설명

[0018] 이하에 첨부되는 도면들은 본 발명에 관한 이해를 돕기 위한 것으로, 상세한 설명과 함께 본 발명에 대한 실시 예들을 제공한다. 다만, 본 발명의 기술적 특징이 특정 도면에 한정되는 것은 아니며, 각 도면에서 개시하는 특징들은 서로 조합되어 새로운 실시 예로 구성될 수 있다. 각 도면에서의 참조 번호(reference numerals)들은 구조적 구성요소(structural elements)를 의미한다.

도 1은 일 실시 예에 따른 통신 시스템의 개략적인 블록도이다.

도 2는 클라이언트(사용자 기기)가 데드 노드 검출 메커니즘에서 작동하는 일 실시 예를 나타낸다.

도 3 은 데드 노드 검출 메커니즘이 작동하는 일 실시 예를 나타낸다.

도 4는 서버가 자원 업데이트 주기를 결정하는 일 실시 예를 나타낸다.

도 5는 시뮬레이션 시간이 증가함에 따라 Client의 에너지 소비 변화를 나타낸다.

도 6은 시뮬레이션 중에 서버의 에너지 수준 변화를 보여준다.

도 7은 시뮬레이션 시간에 따른 서버의 업데이트 메시지 기간의 변화를 나타낸다.

도 8은 본 발명의 일 실시 예와 관련된 ERDU 방법을 이용하는 사용자 기기와 서버와 자원 디렉토리의 구성을 도시하는 도면이다.

발명을 실시하기 위한 구체적인 내용

[0019] 본 발명에서 사용되는 용어는 본 발명에서의 기능을 고려하면서 가능한 현재 널리 사용되는 일반적인 용어들을 선택하였으나, 이는 당 분야에 종사하는 기술자의 의도 또는 판례, 새로운 기술의 출현 등에 따라 달라질 수 있다. 또한, 특정한 경우는 출원인이 임의로 선정한 용어도 있으며, 이 경우 해당되는 발명의 설명 부분에서 상세히 그 의미를 기재할 것이다. 따라서 본 발명에서 사용되는 용어는 단순한 용어의 명칭이 아닌, 그 용어가 가지는 의미와 본 발명의 전반에 걸친 내용을 토대로 정의되어야 한다.

[0020] 이하의 실시 예들은 본 발명의 구성요소들과 특징들을 소정 형태로 결합한 것들이다. 각 구성요소 또는 특징은 별도의 명시적 언급이 없는 한 선택적인 것으로 고려될 수 있다. 각 구성요소 또는 특징은 다른 구성요소나 특

징과 결합되지 않은 형태로 실시될 수 있으며, 일부 구성요소들 및/또는 특징들을 결합하여 본 발명의 실시 예를 구성할 수도 있다. 또한, 본 발명의 실시 예들에서 설명되는 동작들의 순서는 변경될 수 있다. 어느 실시 예의 일부 구성이나 특징은 다른 실시 예에 포함될 수 있고, 또는 다른 실시 예에 대응하는 구성 또는 특징과 교체될 수 있다.

[0021] 도면에 대한 설명에서, 본 발명의 요지를 흐릴 수 있는 절차 또는 단계 등은 기술하지 않았으며, 당업자의 수준에서 이해할 수 있을 정도의 절차 또는 단계는 또한 기술하지 아니하였다.

[0022] 명세서 전체에서, 어떤 부분이 어떤 구성요소를 "포함(comprising 또는 including)"한다고 할 때, 이는 특별히 반대되는 기재가 없는 한 다른 구성요소를 제외하는 것이 아니라 다른 구성요소를 더 포함할 수 있는 것을 의미한다. 또한, 명세서에 기재된 "...부", "...기", "모듈(module)" 등의 용어는 적어도 하나의 기능이나 동작을 처리하는 단위를 의미하며, 이는 하드웨어나 소프트웨어 또는 하드웨어 및 소프트웨어의 결합으로 구현될 수 있다. 또한, "일(a 또는 an)", "하나(one)", "그(the)" 및 유사 관련어는 본 발명을 기술하는 문맥에 있어서(특히, 이하의 청구항의 문맥에서) 본 명세서에 달리 지시되거나 문맥에 의해 분명하게 반박되지 않는 한, 단수 및 복수 모두를 포함하는 의미로 사용될 수 있다.

[0023] 이하, 본 발명에 따른 바람직한 실시 형태를 첨부된 도면을 참조하여 상세하게 설명한다. 첨부된 도면과 함께 이하에 개시될 상세한 설명은 본 발명의 예시적인 실시 형태를 설명하고자 하는 것이며, 본 발명이 실시될 수 있는 유일한 실시 형태를 나타내고자 하는 것이 아니다.

[0024] 또한, 본 발명의 실시 예들에서 사용되는 특정 용어들은 본 발명의 이해를 돕기 위해서 제공된 것이며, 이러한 특정 용어의 사용은 본 발명의 기술적 사상을 벗어나지 않는 범위에서 다른 형태로 변경될 수 있다.

[0025] 본 발명은 사물 인터넷을 이용하는 통신 시스템을 구현하기 위해 사용되는 사용자 기기, 서버, 자원 디렉토리 간 관계에서 에너지를 효율적으로 사용하는 방법을 제안하기 위함이다.

[0026] 이하에서는 본 발명의 기초가 되는 사물 인터넷(Internet of Things, IoT) 및 사물 인터넷의 제한된 응용 프로토콜(Constrained Application Protocol, CoAP)에 대해서 설명하도록 한다.

[0027] 최근에는 사물 인터넷 환경에서 데이터를 교환하기 위해 가벼운 웹 기반 전송 프로토콜을 사용하는 것이 제안되고 있다. 대부분의 사물 인터넷 서비스는 제한된 장치와 손실이 많은 네트워크를 사용하기 때문이다. 제한된 응용 프로그램 프로토콜 (CoAP)은 자원이 제한된 환경에서 안정적이고 효율적인 통신을 가능하게 하기 위해 개발되었다. CoAP은 사용자 기기(클라이언트)와 서버를 사용한다. 서버는 URI (Uniform Resource Identifier)로 식별되는 자원을 감지하는 데이터 구조를 구성하고, 클라이언트는 CoAP 메서드 (예: GET, PUT, POST 및 DELETE)를 사용하여 이러한 자원에 액세스한다. 서버에서 센서 데이터를 검색하기 위해 클라이언트는 서버의 자원을 알아야 하므로, 자원 검색은 필수 불가결하다.

[0028] CoAP에는 분산 및 중앙 집중식이라는 두 가지 유형의 자원 검색이 포함된다. 전자의 경우 클라이언트는 자원을 검색하기 위해 여러 서버에 쿼리(query)를 직접 보낸다. 이 방법은 각각의 서버에 모두 쿼리를 송신해야 하므로 에너지 효율이 매우 중요한 사물 인터넷 환경에는 적합하지 않다. 반면에, 후자는 자원 디렉토리(Resource Directory, RD)를 사용하여 인터넷에서 정의된 자원 검색 절차에서 발생한 에너지 소비를 최소화한다. 따라서 중앙 집중식 방법은 다양한 IoT 어플리케이션에 상대적으로 적합하다. 중앙 집중식 자원 검색에서 자원 디렉토리는 다양한 서버에서 호스팅하는 자원의 웹 링크 집합을 참조하는 자원 디렉토리 항목을 유지 관리한다. 서버는 자원을 자원 디렉토리에 직접 등록하고 클라이언트는 자원 디렉토리를 요청하여 특정 서버의 자원을 인식할 수 있다.

[0029] 그러나 자원 디렉토리를 사용하는 중앙 집중식 자원 검색 접근법에는 두 가지 문제가 있다. 첫째, 각 서버는 자원 디렉토리 정보를 최신 상태로 유지하기 위해 주기적으로 업데이트 메시지를 보내야 한다. 이 작업으로 인해 추가적인 신호 오버 헤드가 발생하고 네트워크 수명이 단축될 수 있다. 둘째, 서버가 장애 또는 전원 종료로 인해 상태 보고없이 서버가 갑자기 종료되면 자원 디렉토리는 최신 자원 정보를 유지할 수 없다.

[0030] 이 문제를 해결하기 위해, 본 발명은 CoAP를 위한 에너지 효율적인 자원 디렉토리 업데이트(energy-efficient resource directory update, ERDU) 기법을 제안한다. ERDU 기법은 1) 데드 노드 검출 (dead node detection)과 2) 업데이트 주기 적응의 두 가지 메커니즘을 통해 CoAP의 중앙 집중식 자원 검색 접근법에서 기존 RD 업데이트 메커니즘의 상기 언급된 문제를 해결할 수 있다.

[0031] 본 발명의 내용을 설명하기 전에 이하에서는 사물 인터넷을 이용하는 통신시스템에 대한 내용을 우선 설명하도

록 한다.

[0032] 도 1은 일 실시 예에 따른 통신 시스템의 개략적인 블록도이다.

[0033] 도 1을 참조하면, 통신 시스템(communication system; 10)은 클라이언트(client; 100), 자원 디렉토리(Resource Directory; 200), 및 복수의 노드들(310, 320 및 330)을 포함할 수 있다.

[0034] 통신 시스템(10)은 제한된 어플리케이션 프로토콜(Constrained Application Protocol(CoAP))이 적용된 시스템일 수 있다. CoAP은 메모리(memory), 프로세싱(processing), 및 전력(power) 측면에서 제한된 장치, 예를 들어 소형 저전력 센서, 스위치, 및 밸브 등에 의도된 프로토콜일 수 있다. CoAP은 그러한 장치들로 하여금 인터넷을 통하여 통신하도록 할 수 있다.

[0035] 클라이언트(100)는 CoAP 클라이언트 및/또는 CoAP 노드로 명명될 수 있다.

[0036] 복수의 노드들(310, 320 및 330)은 센서 노드(sensor node; 310), 액츄에이터 노드(320), 및 복합 노드(composite node; 330)를 포함할 수 있다.

[0037] 센서 노드(310)와 액츄에이터 노드(320)는 하나의 자원, 예를 들어 센서 또는 액츄에이터를 포함할 수 있다.

[0038] 복합 노드(330)는 복수의 통합된 자원들, 예를 들어 복수의 센서들, 복수의 액츄에이터, 또는 센서 및 액츄에이터를 포함할 수 있다.

[0039] 복수의 노드들(310, 320 및 330) 각각은 노드 식별자(NodeID)를 가질 수 있다.

[0040] 복수의 노드들(310, 320 및 330)의 장치들 각각은 유닛 식별자(UnitID)를 통해 분리적으로 식별할 수 있다. 예를 들어, 자원에 대한 유닛 식별자는 단일 노드 내의 모든 통합된 자원들 사이에서 유일(unique)하고, 동일한 유닛 식별자는 또 다른 노드 내에서 통합된 자원을 표현(represent)할 수 있다.

[0041] 하나의 노드 내에서 통합된 자원들은 노드 식별자 및 유닛 식별자에 의하여 분리적으로 식별될 수 있다. 예를 들어, 복합 노드(330) 내의 통합된 자원들은 노드 식별자 및 유닛 식별자에 의하여 분리적으로 식별될 수 있다.

[0042] 복수의 노드들(310, 320 및 330) 각각은 하나의 IP 주소를 가지며, 클라이언트(100) 및/또는 통신 시스템(10)의 다른 모듈, 예를 들어 자원 디렉토리(200)와 통신을 수행할 수 있다.

[0043] 개별 통합 자원의 식별을 가능하게 하는 유닛 식별자는 CoAP 메시지 내의 유닛 식별자 옵션에 포함될 수 있다.

[0044] CoAP 메시지 내의 유닛 식별자 옵션은, 통신을 위한 단일 IP 주소를 가지는 동안에 복수 센서 및 액츄에이터로 구성되는 복합 노드(330)의 사용을 가능하게 할 수 있다.

[0045] 통합된 자원은, 유닛 크기(Unit Size) 및 유닛 식별자(UnitID)의 추가적 옵션, 즉 유닛 식별자 옵션을 포함하는 CoAP 메시지로 개별적 또는 일괄적으로 통신되고, 이를 이용하여 제어될 수 있다.

[0046] 이러한 옵션들은, 단일 복합 메시지, 즉 UnitID="*"를 이용함으로써 CoAP로 하여금 통신 및 복수 자원 제어를 가능하게 하며, 효율적으로 IP 주소, 즉 하나의 IP 복수 식별자를 활용하며, 통신 트래픽을 감소시키며 이에 따라 CoAP 자원들 간의 전력을 절약하도록 할 수 있다.

[0047] 복수의 노드들(310, 320 및 330) 각각은 자원 디렉토리(200)에 이러한 자원(들)을 등록시킬 수 있다.

[0048] 자원 디렉토리(200)는 자원 검색(resource discovery), 등록(registration) 및 룩업(lookup) 등의 기능 세트(function set)를 정의할 수 있다.

[0049] 각 노드(310, 320 및 330)가 자원 디렉토리(200)에 각 노드(310, 320 및 330)의 모든 통합된 자원을 등록하면, 클라이언트(100)는 하나 또는 복수의 자원들을 룩업할 수 있으며, 그것들과 직접적으로 상호작용(interact)할 수 있다.

[0050] 자원 디렉토리(200)는 자원의 자동화된 검색 및 룩업을 가능하게 하며, 복수의 유닛 식별자들은 복수의 자원들과의 상호작용을 위한 단일 IP의 효율적인 이용을 제공할 수 있다.

[0051] 이하에서는 본 발명에서 제안하는 ERDU 기법의 1) 데드 노드 검출 (dead node detection)과 2) 업데이트 주기 결정에 대해서 자세하게 설명하도록 한다.

[0052] 1. 데드 노드 검출

[0053] 클라이언트가 서버의 실패 또는 전원 종료로 인해 RD를 업데이트할 수없는 경우 데드 노드 검출을 시작한다. 서

버가 클라이언트의 요청에 응답하지 않으면 클라이언트는 서버의 자원 정보를 RD에 요청한다. RD가 사전에 결정된 횟수 동안 이전 자원 항목과 동일한 정보로 응답하는 경우, 클라이언트는 서버가 종료되었다고 판별한다. 이 메커니즘은 불필요한 메시지 전송을 방지한다.

[0054] 기존의 RD 업데이트 메커니즘에서 RD의 자원 항목은 서버의 업데이트 메시지에 표시된 수명 동안만 살아 있다. 따라서 서버가 갑자기 종료된 경우 RD 항목을 업데이트할 수 없다. 보다 구체적으로, 서버가 클라이언트의 요청에 응답하지 않으면 클라이언트는 반복적으로 서버에 요청을 보낸다. 그럼에도 불구하고 주어진 수의 요청에 대한 응답이 없으면 클라이언트는 서버의 자원 정보를 RD에 요청한다. 이후, RD는 해당 서버의 자원을 검색하여 이를 클라이언트로 다시 보낸다. 그러나 서버가 작동하지 않는 경우 RD는 서버에서 업데이트 메시지를 받을 수 없기 때문에 서버가 여전히 살아있는 것처럼 서버의 이전 자원으로 응답한다. 이러한 문제를 해결하기 위해, 본 발명이 제안하는 방법은 클라이언트가 작동하지 않는 서버를 검출할 수 있도록 데드 노드 검출 메커니즘을 제안하고, 데드 노드가 검출된 경우 해당 서버의 자원을 RD 항목에서 제거한다.

[0055] 도 2는 클라이언트(사용자 기기)가 데드 노드검출 메커니즘에서 작동하는 일 실시 예를 나타낸다.

[0056] 클라이언트는 데드 노드 검출 메커니즘에서 RD에서 수신한 이전 자원을 사용하여 서버에 요청 메시지를 송신한다(S210). 이 방법은 사전에 정의된 횟수만큼 반복될 수 있다.

[0057] 클라이언트는 서버로부터 자원 요청 메시지에 대한 응답 메시지의 유무를 기초로 서버의 동작 여부를 판단한다(S220). 요청 수가 사전에 정의된 수를 초과하면, 클라이언트는 서버가 동작하지 않는 것으로 판단한다.

[0058] 이후 클라이언트는 서버가 동작하지 않는다고 판단되었을 때, RD가 서버의 이전 자원을 삭제하도록 요청한다(S230). 마지막으로 RD는 서버의 이전 자원을 삭제한다.

[0059] 도 3 은 데드 노드 검출 메커니즘이 작동하는 일 실시 예를 나타낸다.

[0060] 도 3의 311과 같이 클라이언트가 서버로부터 요청 메시지에 대한 응답 메시지를 수신한 경우, 서버는 RD에 자원 업데이트 메시지를 송신할 수 있다.

[0061] 도 3의 321과 같이 서버는 실패 또는 전원 종료로 인해 중지되었다. 따라서 서버는 클라이언트의 여러 요청에 응답하지 않는다. 이 경우 클라이언트는 서버의 자원을 얻기 위해 RD에 요청을 보내고, RD가 요청 메시지를 수신하면 서버의 이전 자원을 이용하여 클라이언트에 응답한다. 그러나 서버가 종료되었기 때문에 클라이언트는 여전히 서버에서 응답 메시지를 받을 수 없다. 클라이언트가 기설정된 횟수인 두 번 요청 메시지를 송신한 후에도 서버에서 응답을 받지 못하면, 클라이언트는 서버가 작동하지 않는 것으로 판단하고 RD에 자원 삭제를 요청한다. 마지막으로 RD는 자원을 삭제하고 클라이언트에 자원 삭제 성공 메시지를 송신한다.

[0062] 2. 업데이트 주기 결정

[0063] 업데이트 기간의 적응에서 서버는 에너지 수준에 따라 업데이트 메시지의 전송 기간을 조정한다. 이는 업데이트 메시지의 전송을 줄임으로써 서버의 에너지 효율성을 향상시킨다.

[0064] RD는 최신 자원을 유지 관리하기 위해 주기적으로 서버로부터 업데이트 메시지를 수신한다. 그러나, 이러한 업데이트 동작은 부적절한 업데이트 메시지 주기가 사용될 때 추가적인 신호 오버 헤드를 유발하고 네트워크 수명을 단축시킬 수 있다. 예를 들어, 업데이트 메시지 기간이 매우 짧게 설정된 경우 서버는 업데이트 메시지를 자주 전송하기 때문에 상당한 에너지를 소비한다. 반대의 경우 RD는 최신 항목을 유지 관리할 수 없다.

[0065] 이 문제를 해결하기 위해 ERDU 체계는 서버의 수명과 에너지 효율성을 향상시키고 RD 항목에 최신 자원을 유지하는 것을 목표로 하는 업데이트 기간 적응 메커니즘을 포함한다. 이를 위해 서버는 에너지 수준에 따라 업데이트 메시지 기간을 조정한다.

[0066] 도 4는 서버가 자원 업데이트 주기를 결정하는 일 실시 예를 나타낸다.

[0067] 서버는 자원 업데이트 메시지를 송신하는 자원 업데이트 주기를 결정한다(S410). 그리고, 서버는 결정된 자원 업데이트 주기에 기초하여 자원 업데이트 메시지를 자원 디렉토리에 송신할 수 있다(S420).

[0068] 서버는 초기에 업데이트 메시지 기간의 최대 길이를 제한하는 배율(곱셈 계수)을 설정한 후 남은 에너지 수준을 확인한다. 그 후, 새로운 업데이트 메시지 기간을 결정하기 위해, 서버는 초기 업데이트 메시지 기간과 함께 2개의 파라미터 (즉, 배율 및 잔여 에너지 수준)를 사용한다. 배율은 업데이트 메시지 기간의 최대 길이를 설정하는 데 사용되는 상수 값이다. 예를 들어 초기 업데이트 메시지 기간이 10ms이고 배율이 5인 경우 업데이트 메

시지 기간의 최대 길이는 50ms다. 서버의 남은 에너지 수준이 감소하면 서버의 에너지를 절약하기 위해 업데이트 메시지 기간을 늘려야 한다. 따라서, 잔여 에너지 수준은 업데이트 메시지 기간의 적응을 위해 이용된다. 새로운 업데이트 메시지 주기는 아래와 같은 수학식 1을 이용하여 결정될 수 있다.

[0069] [수학식 1]

[0070] $$UP_{new} = \{UP_{initial} \times (k-1) \times (1-REL)\} + UP_{initial}$$

[0071] 여기서, UP_{new}는 새 업데이트 메시지 기간을 나타내고, $UP_{initial}$는 초기 업데이트 메시지 기간을 나타내고, k는 배율을 나타내고, REL은 서버의 남은 에너지 수준을 나타낸다.

[0072] 이하에서는 1) 데드 노드 검출 (dead node detection)과 2) 업데이트 주기 결정을 이용한 ERDU 체계의 성능을 검증한 실험 내용을 설명하도록 한다.

[0073] 3. ERDU 성능효과

[0074] 본 실험 시뮬레이션은 MATLAB을 사용하여 ERDU 체계의 성능을 평가하기 위해 수행된다.

[0075] 데드 노드 검출 및 업데이트 주기 결정 메커니즘의 유효성을 확인하기 위해 다양한 시나리오에서 시뮬레이션을 수행하고 ERDU 체계와 기존 RD 메커니즘의 결과를 비교한다.

[0076] 아래에서는 우선 실험 시뮬레이션의 설정을 설명하고, 실험 시뮬레이션의 결과에 대해 자세히 설명한다.

[0077] 3.1 시뮬레이션 설정

[0078] ERDU 기법은 CoAP의 중앙 집중식 자원 탐색 접근법에서 기존 RD 업데이트 메커니즘의 에너지 비 효율성 문제를 해결하도록 설계된 방법이다. 이를 위해, ERDU 체계는 두 가지 메커니즘을 데드 노드 검출과 업데이트 주기 결정을 포함한다.

[0079] ERDUR 기법의 성능을 나타내기 위한 본 시뮬레이션에서는 단일 클라이언트, RD 및 서버를 사용하고 서버가 나머지 에너지 수준을 확인할 수 있다고 가정한다. 본 시뮬레이션에서 배터리 용량은 1,000 mA, Tx는 11.6 mA, Rx는 12.3 mA, 수면은 0.4 mA를 소모하는 서버를 설정되었다. 초기 업데이트 메시지 기간은 10ms로 설정되었다. 또한 서버가 5 %에서 작동할 수 있도록 최소 에너지 수준을 설정하였다. 즉, 서버의 에너지 잔량이 5 % 미만이면 서버는 방전된 것으로 간주된다. 자세한 시뮬레이션 매개 변수는 표 1과 같다.

표 1

[0080] Simulation parameters.

Parameter	Value
Tx energy consumption	11.6 mA
Rx energy consumption	12.6 mA
Sleep energy consumption	0.4 mA
Initial update message period	10 ms
Maximum battery capacity	1,000 mA
Minimum energy level	5%
Multiplying factor	5

[0081] 3.2 시뮬레이션 결과

[0082] 도 5 내지 7은 본 시뮬레이션과 같은 도면이다.

[0083] 도 5는 시뮬레이션 시간이 증가함에 따라 고객의 에너지 소비 변화를 나타낸다. 시뮬레이션에서 우리는 클라이언트의 에너지 소비를 10ms마다 측정하여 사각 노드 검출의 효과를 조사한다. 그림에서 보는 바와 같이, ERDU 기법은 시뮬레이션 시간이 80ms에 이르면 기존 RD 메커니즘보다 적은 에너지를 소비한다. 이 차이는 ERDU 체계를 사용하는 클라이언트에 의한 80ms에서의 사각 노드 검출에 기인할 수 있다. 따라서 더 이상 요청 메시지를 서버로 전송하지 않는다. 반면에 기존 RD 메커니즘을 사용하는 클라이언트는 서버가 종료되었는지 여부를 알 수 없다. 따라서 이 경우 클라이언트는 80ms 후에도 요청 메시지를 계속 전송하므로 불필요한 에너지 소비가 발생한다.

[0084] 도 6은 시뮬레이션 중에 서버의 에너지 수준 변화를 보여준다. 시뮬레이션에서 업데이트주기 적응의 효과는 서버의 에너지 수준을 10ms마다 측정하여 조사한다. 그림에서 보는 바와 같이 잔여 에너지에 따라 업데이트 메시지주기를 조정하기 때문에 ERDU 체계의 수명은 기존 RD 메커니즘의 수명보다 3 배 더 길다. 특히, 서버는 ERDU의 경우 2,600 ms까지 존재하지만 기존 RD의 경우에는 800ms까지만 존재한다.

[0085] 도 7은 시뮬레이션 시간에 따른 서버의 업데이트 메시지 기간의 변화를 나타낸다. 도 7에서와 같이 ERDU 방식을 사용하는 서버의 업데이트 메시지 기간은 약 50ms로 증가한다. 이는 배율이 5로 설정되었기 때문이다. 반면에 기존 RD 메커니즘을 사용하는 서버는 초기 업데이트 메시지 기간과 동일한 10ms로 업데이트 메시지 기간을 유지한다. 따라서 업데이트 메시지를 자주 전송하기 때문에 ERDU 방식을 사용하는 서버보다 많은 에너지를 소비한다.

[0086] 4. 장치구성

[0087] 도 8은 본 발명의 일 실시 예와 관련된 ERDU 방법을 이용하는 사용자 기기와 서버와 자원 디렉토리의 구성을 도시하는 도면이다. ERDU 방법을 이용하는 사용자 기기와 서버와 자원 디렉토리는 각각 송수신부(810, 850, 890), 프로세서(820, 860, 900), 및 메모리(830, 870, 910)를 포함할 수 있다. 도 8에서는 ERDU 방법을 이용하는 사용자 기기와 서버와 자원 디렉토리 간의 통신 환경만을 도시하였으나, 다수의 자원 디렉토리와 기기 간에도 통신 환경이 구축될 수 있다.

[0088] 각 송수신부(810, 850, 890)는 각각 송신부(812, 852, 892) 및 수신부(814, 854, 894)를 포함한다. 사용자 기기(800)의 송수신부(810)는 중앙 컨트롤러(850) 및 다른 기기들과 통신 신호를 송신 및 수신하도록 구성되며, 프로세서(820)는 송수신부(810)와 기능적으로 연결되어 송수신부(810)가 다른 기기들과 사물 인터넷에 대응되는 신호를 송수신하는 과정을 제어하도록 구성될 수 있다. 또한, 프로세서(820)는 전송할 신호에 대한 각종 처리를 수행한 뒤 송신부(812)로 전송하며, 수신부(864)가 수신한 신호에 대해서도 처리를 수행한다.

[0089] 필요한 경우, 프로세서(820)는 송수신된 신호에 포함된 정보를 메모리(830)에 저장할 수 있다. 이러한 구조를 가지는 사용자 기기(800)는 이상에서 설명한 데드 노드 검출과 업데이트 주기 결정의 다양한 실시 형태에 따라 동작할 수 있다.

[0090] 서버(840)의 송수신부(850)는 자원 디렉토리 및 사용자 기기들과 신호를 송신 및 수신하도록 구성되며, 프로세서(860)는 송수신부(850)와 기능적으로 연결되어 송수신부(850)가 다른 기기들과 신호를 송수신하는 과정을 제어한다. 또한, 프로세서(860)는 전송할 신호에 대한 각종 처리를 수행하여 송신부(852)로 전송하거나 수신부(854)가 수신한 신호에 대해 각종 처리를 수행할 수 있다. 나아가, 프로세서(860)는 교환된 메시지에 포함된 정보를 메모리(870)에 저장할 수도 있다. 이와 같은 구조를 가지는 서버(840)는 앞서 설명한 다양한 실시 형태의 방법을 수행할 수 있다.

[0091] 자원 디렉토리(880)의 송수신부(890)는 서버 및 사용자 기기들과 신호를 송신 및 수신하도록 구성되며, 프로세서(900)는 송수신부(890)와 기능적으로 연결되어 송수신부(890)가 다른 기기들과 신호를 송수신하는 과정을 제어한다. 또한, 프로세서(900)는 전송할 신호에 대한 각종 처리를 수행하여 송신부(892)로 전송하거나 수신부(894)가 수신한 신호에 대해 각종 처리를 수행할 수 있다. 나아가, 프로세서(900)는 교환된 메시지에 포함된 정보를 메모리(910)에 저장할 수도 있다. 이와 같은 구조를 가지는 자원 디렉토리(880)는 앞서 설명한 다양한 실시 형태의 방법을 수행할 수 있다.

[0092] 사용자 기기(800) 및 서버(840) 및 자원 디렉토리 (880) 각각의 프로세서(820, 860, 900)는 각각 동작을 지시(예를 들어, 제어, 조정, 관리 등)한다. 각각의 프로세서들(820, 860, 900)은 프로그램 코드들 및 데이터를 저장하는 메모리(830, 870, 910)들과 연결될 수 있다. 메모리(830, 870, 910)는 프로세서(820, 860, 900)에 연결되어 오퍼레이팅 시스템, 어플리케이션, 및 일반 파일(general files)들을 저장한다.

[0093] 본 발명의 프로세서(820, 860, 900)는 컨트롤러(controller), 마이크로 컨트롤러(microcontroller), 마이크로프로세서(microprocessor), 마이크로컴퓨터(microcomputer) 등으로도 호칭될 수 있다. 한편, 프로세서(820, 860, 900)는 하드웨어(hardware) 또는 펌웨어(firmware), 소프트웨어, 또는 이들의 결합에 의해 구현될 수 있다.

[0094] 하드웨어를 이용하여 본 발명의 실시 예를 구현하는 경우에는, 본 발명을 수행하도록 구성된 ASICs(application specific integrated circuits) 또는 DSPs(digital signal processors), DSPDs(digital signal processing devices), PLDs(programmable logic devices), FPGAs(field programmable gate arrays) 등이 프로세서(820,

860, 900)에 구비될 수 있다.

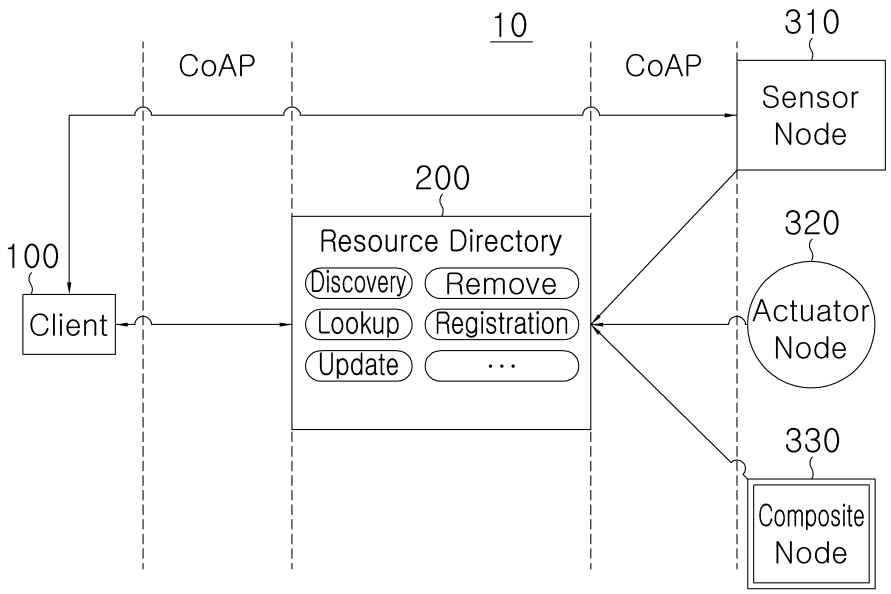
[0095] 한편, 상술된 실시 예들은 컴퓨터에 의하여 실행 가능한 명령어 및 데이터를 저장하는 컴퓨터로 읽을 수 있는 기록매체의 형태로 구현될 수 있다. 상기 명령어 및 데이터 중 적어도 하나는 프로그램 코드의 형태로 저장될 수 있으며, 프로세서에 의해 실행되었을 때, 소정의 프로그램 모듈을 생성하여 소정의 동작을 수행할 수 있다.

[0096] 컴퓨터로 읽을 수 있는 기록매체란, 예를 들어 하드디스크 등과 같은 마그네틱 저장매체, CD 및 DVD 등과 같은 광학적 판독매체 등을 의미할 수 있으며, 네트워크를 통해 접근 가능한 서버에 포함되는 메모리를 의미할 수도 있다. 예를 들어, 컴퓨터로 읽을 수 있는 기록매체는 전자 장치 또는 서버의 메모리가 될 수도 있다. 또한, 전자 장치 또는 서버와 네트워크를 통하여 연결된 단말, 서버 등에 포함되는 메모리가 될 수도 있다.

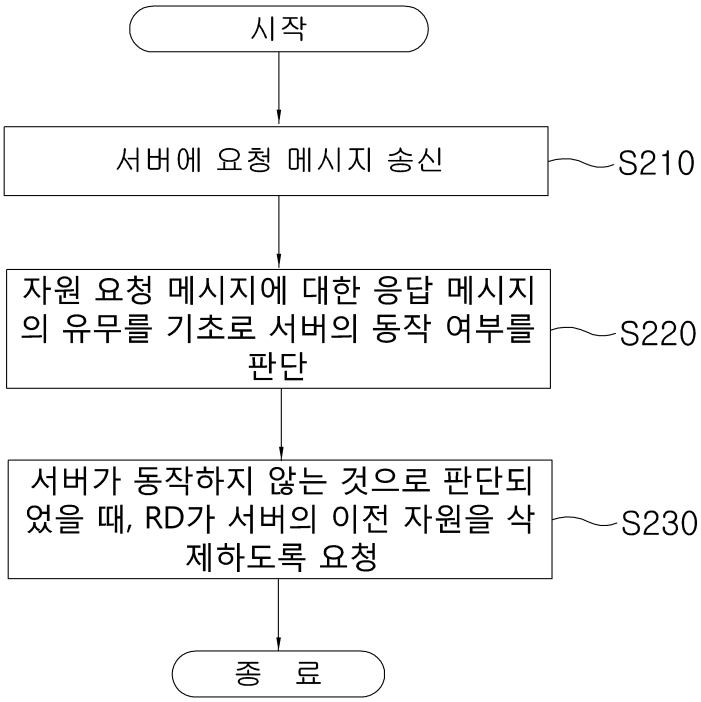
[0097] 이상과 첨부된 도면을 참조하여 실시 예를 설명하였지만, 일 실시 예가 속하는 기술분야에서 통상의 지식을 가진 자는 일 실시 예가 그 기술적 사상이나 필수적인 특징을 변경하지 않고서 다른 구체적인 형태로 실시될 수 있다는 것을 이해할 수 있을 것이다. 그러므로 이상에서 기술한 실시 예들은 모든 면에서 예시적인 것이며 한정적이 아닌 것으로 이해해야만 한다.

도면

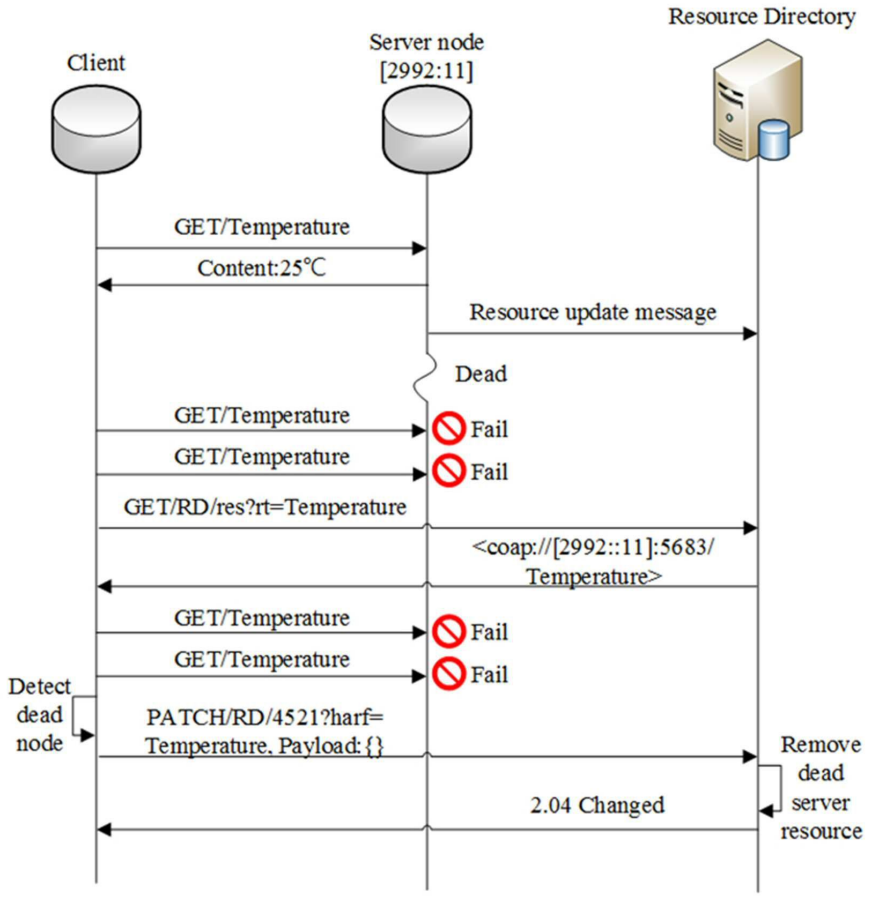
도면1



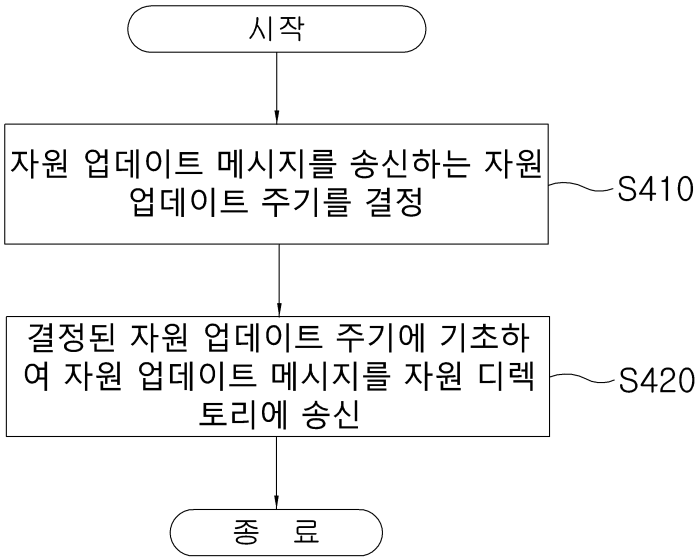
도면2


시작
서버에 요청 메시지 송신
S210
자원 요청 메시지에 대한 응답 메시지의 유무를 기초로 서버의 동작 여부를 판단
S220
서버가 동작하지 않는 것으로 판단되었을 때, RD가 서버의 이전 자원을 삭제하도록 요청
S230
종 료

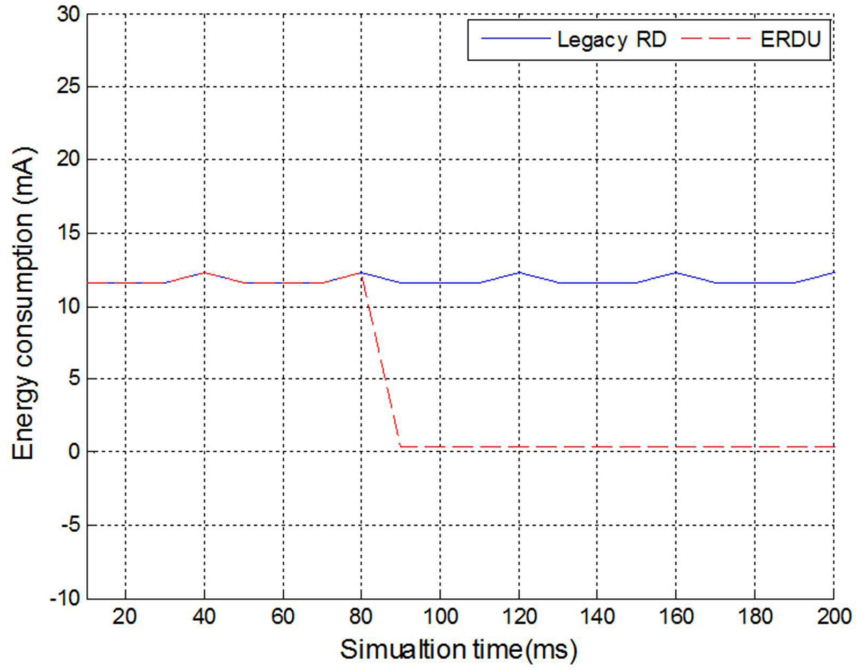
도면3


Client
Server node
[2992:11]
Resource Directory
GET/Temperature
Content:25℃
Resource update message
Dead
GET/Temperature
Fail
GET/Temperature
Fail
GET/RD/res?rt=Temperature
<coap://[2992::11]:5683/
Temperature>
GET/Temperature
Fail
GET/Temperature
Fail
Detect
dead
node
PATCH/RD/4521?harf=
Temperature, Payload:{}
Remove
dead
server
resource
2.04 Changed

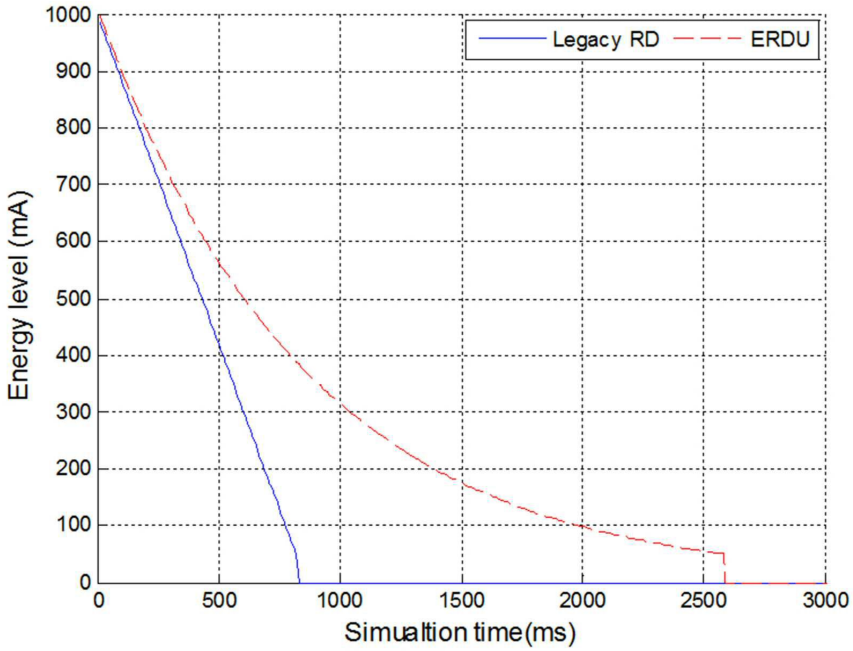
도면4


시작
자원 업데이트 메시지를 송신하는 자원 업데이트 주기를 결정
S410
결정된 자원 업데이트 주기에 기초하여 자원 업데이트 메시지를 자원 디렉토리에 송신
S420
종 료

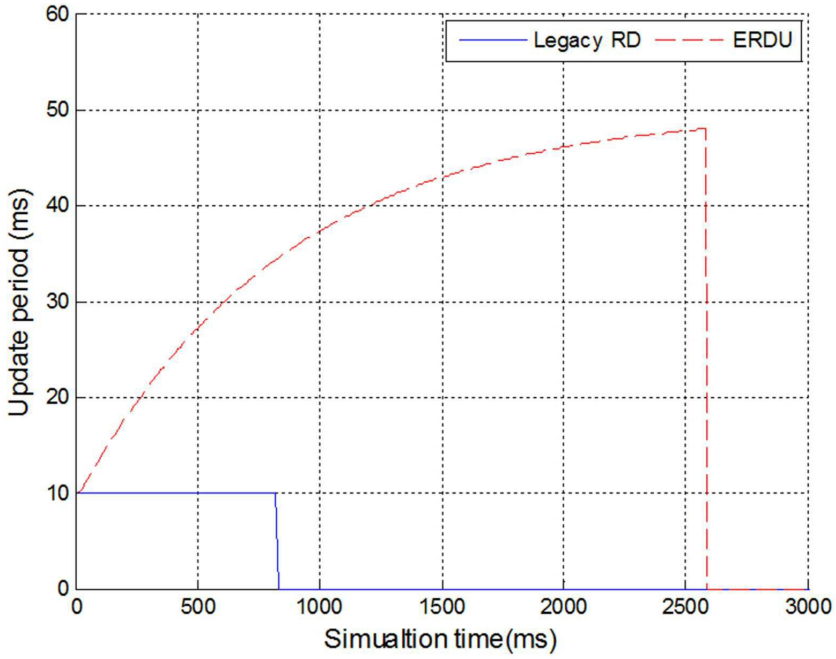
도면5


Legacy RD
ERDU
Energy consumption (mA)
Simualtion time(ms)
30
25
20
15
10
5
0
-5
-10
20
40
60
80
100
120
140
160
180
200

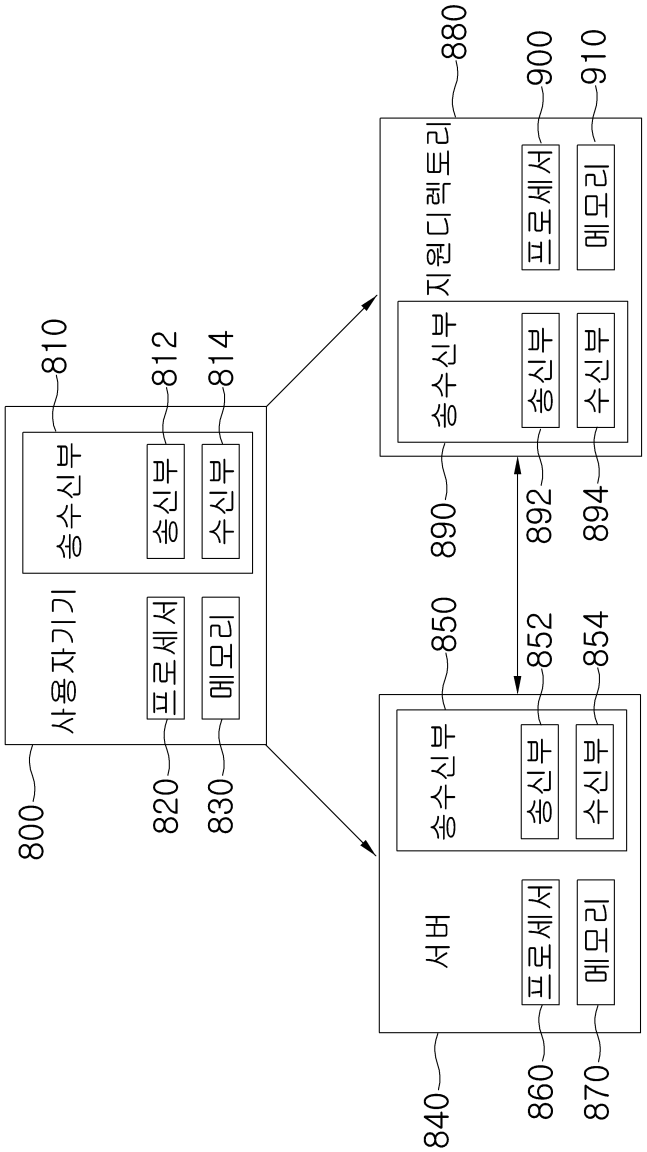
도면6


Legacy RD
ERDU
Energy level (mA)
Simualtion time(ms)
1000
900
800
700
600
500
400
300
200
100
0
0
500
1000
1500
2000
2500
3000

도면7


Legacy RD
ERDU
Update period (ms)
Simualtion time(ms)
0
10
20
30
40
50
60
0
500
1000
1500
2000
2500
3000

도면8


800
사용자기기
810
송수신부
812
송신부
814
수신부
820
프로세서
830
메모리
840
서버
850
송수신부
852
송신부
854
수신부
860
프로세서
870
메모리
880
지원디렉토리
890
송수신부
892
송신부
894
수신부
900
프로세서
910
메모리