



(19) 대한민국특허청(KR)

(12) 등록특허공보(B1)

(45) 공고일자 2015년02월16일

(11) 등록번호 10-1493166

(24) 등록일자 2015년02월06일

- (51) 국제특허분류(Int. Cl.)
G06F 15/16 (2006.01) G06F 3/048 (2006.01)
- (21) 출원번호 10-2013-0110310
- (22) 출원일자 2013년09월13일
심사청구일자 2013년09월13일
- (30) 우선권주장
1020130099843 2013년08월22일 대한민국(KR)
- (56) 선행기술조사문헌
KR1020120085436 A
KR1020130017264 A
KR1020130090493 A

- (73) 특허권자
한국과학기술원
대전광역시 유성구 대학로 291(구성동)
- (72) 발명자
최준균
대전 유성구 대학로 291, (구성동, 한국과학기술원)
양진홍
대전 유성구 문지로 193 (문지동, KAIST ICC)
박효진
대전광역시 유성구 전민동
- (74) 대리인
양성보, 김정훈

전체 청구항 수 : 총 15 항

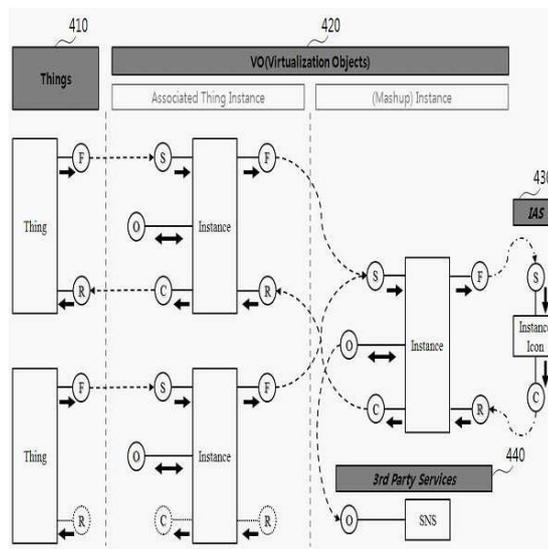
심사관 : 김병균

(54) 발명의 명칭 IoT 환경에서 가상화된 객체의 인스턴스를 관리하기 위한 웹 기반 사용자 인터페이스를 제공하는 서비스 방법 및 시스템

(57) 요약

IoT 환경에서 가상화된 객체의 인스턴스를 관리하기 위한 웹 기반 사용자 인터페이스를 제공하는 서비스 방법 및 시스템이 개시된다. IoT(Internet of Things) 환경에서의 서비스 방법은, 사용자에게 제공되는 웹페이지의 단계별 설정 기능을 통해 입력된, 사물(Thing)에 대한 가상화 오브젝트(Virtualization Object)의 인스턴스에 대한 정보를 확인하는 단계, 상기 웹페이지의 외부 서비스 연결 기능을 통해 입력된 외부 서비스에 대한 정보를 이용하여 상기 외부 서비스와의 인증을 수행하는 단계 및 상기 웹페이지의 오픈 API(Application Program Interface) 설정 기능을 통해 입력된 정보를 이용하여 상기 외부 서비스로 상기 오픈 API를 제공하는 단계를 포함할 수 있다.

대표도



특허청구의 범위

청구항 1

IoT(Internet of Things) 환경에서의 서비스 방법에 있어서,

사용자에게 제공되는 웹사이트의 단계별 설정 기능을 통해 입력된, 사물(Thing)에 대한 가상화 오브젝트(Virtualization Object)의 인스턴스에 대한 정보를 확인하는 단계;

상기 웹사이트의 외부 서비스 연결 기능을 통해 입력된 외부 서비스에 대한 정보를 이용하여 상기 외부 서비스와의 인증을 수행하는 단계; 및

상기 웹사이트의 오픈 API(Application Program Interface) 설정 기능을 통해 입력된 정보를 이용하여 상기 외부 서비스로 상기 오픈 API를 제공하는 단계

를 포함하는 서비스 방법.

청구항 2

제1항에 있어서,

상기 웹사이트를 통해 인스턴스별 아이콘을 표시하는 단계

를 더 포함하는 서비스 방법.

청구항 3

제2항에 있어서,

상기 아이콘이 표시되는 영역을 통해, 인스턴스에 대한 컨트롤 기능 및 상기 인스턴스의 대표값이 표시되는 것을 특징으로 하는 서비스 방법.

청구항 4

제3항에 있어서,

상기 영역을 통해 상기 인스턴스의 상태가 더 표시되는 것을 특징으로 하는 서비스 방법.

청구항 5

IoT(Internet of Things) 환경에서의 서비스 방법에 있어서,

사용자의 요청을 수신하는 단계; 및

상기 사용자의 요청에 따라 웹사이트를 제공하는 단계

를 포함하고,

상기 웹사이트는,

사물(Thing)에 대한 가상화 오브젝트(Virtualization Object)의 인스턴스에 대한 정보를 입력받기 위한 단계별 설정 기능;

외부 서비스와의 인증을 위한 정보를 입력받기 위한 외부 서비스 연결 기능; 및

상기 외부 서비스로 제공하기 위한 오픈 API에 대한 정보를 입력받기 위한 오픈 API 설정 기능

을 포함하는 것을 특징으로 하는 서비스 방법.

청구항 6

제1항 또는 제5항에 있어서,

상기 단계별 설정 기능은, 인스턴스의 이름을 입력받기 위한 사용자 인터페이스, 상기 인스턴스의 태그를 입력

받기 위한 사용자 인터페이스, IoT 노드의 프로파일이 저장된 URL을 입력받기 위한 사용자 인터페이스, 인스턴스 아이콘을 선택하기 위한 사용자 인터페이스, 인스턴스의 상태를 선택하기 위한 사용자 인터페이스, 컨트롤러를 선택하기 위한 사용자 인터페이스 및 대표값의 단위를 선택하기 위한 사용자 인터페이스 중 복수의 사용자 인터페이스들을 단계적으로 상기 사용자에게 노출하는 것을 특징으로 하는 서비스 방법.

청구항 7

제1항 또는 제5항에 있어서,

상기 외부 서비스 연결 기능은, 상기 외부 서비스에 대한 상기 사용자의 계정 정보를 입력받기 위한 사용자 인터페이스를 포함하는 것을 특징으로 하는 서비스 방법.

청구항 8

제1항 또는 제5항에 있어서,

상기 오픈 API 설정 기능은, 오픈 API의 설명 문서가 포함된 위치 정보를 입력 받기 위한 사용자 인터페이스를 포함하는 것을 특징으로 하는 서비스 방법.

청구항 9

제5항에 있어서,

상기 웹페이지는,

인스턴스별 아이콘을 표시하기 위한 아이콘 표시 기능

을 더 포함하는 것을 특징으로 하는 서비스 방법.

청구항 10

IoT(Internet of Things) 환경에서의 서비스 시스템에 있어서,

적어도 하나의 프로그램이 저장된 적어도 하나의 스토리지 유닛; 및

적어도 하나의 프로세서

를 포함하고,

상기 적어도 하나의 프로세서는, 상기 적어도 하나의 프로그램의 제어에 따라,

사용자에게 제공되는 웹페이지의 단계별 설정 기능을 통해 입력된, 사물(Thing)에 대한 가상화 오브젝트(Virtualization Object)의 인스턴스에 대한 정보를 확인하는 과정;

상기 웹페이지의 외부 서비스 연결 기능을 통해 입력된 외부 서비스에 대한 정보를 이용하여 상기 외부 서비스와의 인증을 수행하는 과정; 및

상기 웹페이지의 오픈 API(Application Program Interface) 설정 기능을 통해 입력된 정보를 이용하여 상기 외부 서비스로 상기 오픈 API를 제공하는 과정

을 처리하는 것을 특징으로 하는 서비스 시스템.

청구항 11

제10항에 있어서,

상기 적어도 하나의 프로세서는,

상기 웹페이지를 통해 인스턴스별 아이콘을 표시하는 과정

을 더 처리하는 것을 특징으로 하는 서비스 시스템.

청구항 12

IoT(Internet of Things) 환경에서의 서비스 시스템에 있어서,

적어도 하나의 프로그램이 저장된 적어도 하나의 스토리지 유닛; 및
 적어도 하나의 프로세서
 를 포함하고,
 상기 적어도 하나의 프로세서는, 상기 적어도 하나의 프로그램의 제어에 따라,
 사용자의 요청을 수신하는 과정; 및
 상기 사용자의 요청에 따라 웹페이지를 제공하는 과정
 을 처리하고,
 상기 웹페이지는,
 사물(Thing)에 대한 가상화 오브젝트(Virtualization Object)의 인스턴스에 대한 정보를 입력받기 위한 단계별
 설정 기능;
 외부 서비스와의 인증을 위한 정보를 입력받기 위한 외부 서비스 연결 기능; 및
 상기 외부 서비스로 제공하기 위한 오픈 API에 대한 정보를 입력받기 위한 오픈 API 설정 기능
 을 포함하는 것을 특징으로 하는 서비스 시스템.

청구항 13

제10항 또는 제12항에 있어서,
 상기 단계별 설정 기능은, 인스턴스의 이름을 입력받기 위한 사용자 인터페이스, 상기 인스턴스의 태그를 입력
 받기 위한 사용자 인터페이스, IoT 노드의 프로파일이 저장된 URL을 입력받기 위한 사용자 인터페이스, 인스턴
 스 아이콘을 선택하기 위한 사용자 인터페이스, 인스턴스의 상태를 선택하기 위한 사용자 인터페이스, 컨트롤러
 를 선택하기 위한 사용자 인터페이스 및 대표값의 단위를 선택하기 위한 사용자 인터페이스 중 복수의 사용자
 인터페이스들을 단계적으로 상기 사용자에게 노출하는 것을 특징으로 하는 서비스 시스템.

청구항 14

제10항 또는 제12항에 있어서,
 상기 외부 서비스 연결 기능은, 상기 외부 서비스에 대한 상기 사용자의 계정 정보를 입력받기 위한 사용자 인
 터페이스를 포함하는 것을 특징으로 하는 서비스 시스템.

청구항 15

제10항 또는 제12항에 있어서,
 상기 오픈 API 설정 기능은, 오픈 API의 설명 문서가 포함된 위치 정보를 입력 받기 위한 사용자 인터페이스를
 포함하는 것을 특징으로 하는 서비스 시스템.

명세서

기술분야

[0001] 본 발명의 실시예들은 IoT 환경에서 가상화된 객체의 인스턴스를 관리하기 위한 웹 기반 사용자 인터페이스를 제공하는 서비스 방법 및 시스템에 관한 것이다.

배경기술

[0002] IoT(Internet of Things)는 기존 USN(Ubiquitous Sensor Network)나 M2M(Machine to Machine)에서 진화된 형태를 띠고 있는데, 기존 M2M이 통신장비(end-device)와 사람과의 통신이 주 목적이었다면, IoT는 사물의 범위를 넓혀 우리가 흔히 볼 수 있는 전화기, 책, 온도계 등의 사물을 사람과 통신이 가능하도록 해준다는 것이다. 즉, IoT는 인간과 사물, 서비스의 세 가지 분산된 환경 요소에 대해, 인간의 명시적인 개입 없이도 상호 협력적으로 센싱, 네트워킹, 정보 처리 등 지능적 관계를 형성하는 사물 공간 연결망을 의미할 수 있다.

- [0003] 이러한 IoT 이외에도, WoT(Web of Things)나 WoO(Web of Objects)와 같이 다양한 개념과 기술들이 연구, 개발되고 있다. 이러한 개념과 기술들의 발전 및 확산으로 인해, 사용자들이 손쉽게 인터넷에 연결 가능한 기기들(가젯(gadget) 디바이스, 센서 디바이스, 액추에이터(actuator) 등)의 사용이 늘어날 것으로 예상된다.
- [0004] 특히, 최근 들어 다양한 형태의 웹 연동형 가젯 디바이스들이 출시되고 있으며, 대다수의 센서 디바이스들의 경우, 웹을 통한 제어 및 모니터링 기능을 제공하고 있다. 또한, 웹을 통한 제어 기능을 제공하기 위한 단말들은, 단말 자체에 웹 서버 기능이 내장되거나 또는 온라인 상의 웹 서비스를 통해 제어될 수 있는 기능을 포함하고 있다.
- [0005] 그러나, 현재는 상술한 기술들, 기기들 또는 서비스들이 개별적으로 관리되고 있는 실정이라, 사용자의 불편함이 따르는 문제점이 있다. 예를 들어, 사용자는 다수의 기기들이 사용자 주위에 존재하나 개별적인 접근 경로(일례로, URL)를 통해 각각의 디바이스들을 제어해야 하는 불편함이 있다. 기기 자체가 웹 접근 기능(서버 기능)을 제공하는 경우에는 개별적인 기기의 접근을 위한 네트워크 연동 및 URL(IP 주소)가 필요하며, 전용 웹 서비스를 통해 서비스가 제공되는 경우에는 서비스별 접근 권 생성 및 URL이 필요하다.
- [0006] 또한, 개별적으로 관리되는 데이터들을 유기적으로 연동하여 활용하기 힘든 문제점이 있다. 예를 들어, 집안의 보일러를 제어하는 센서의 경우 센서 기기가 직접 수집할 수 있는 주변 환경 정보는 사람 거주 유무, 내부 온도, 시간대 정도와 같이 매우 제한적이다. 만약, 이러한 센서 기기가 외부 정보(일례로 외부의 기온, 사용자의 스케줄 정보 등)를 추가적으로 활용할 수 있다면, 보다 효과적인 제어 기능을 제공할 수 있으나, 각 기술들, 기기들 또는 서비스들마다 데이터들이 개별적으로 관리되고 있기 때문에 이러한 데이터들을 유기적으로 연동하여 활용하기 힘들다.
- [0007] 도 1은 종래기술에 있어서, 기기에서 임베디드 웹 서버를 내장하고 있는 경우의 서비스 제공 방식을 도시한 도면이다. 사용자는 웹과 같은 온라인(Online (Web))(110)을 통해 실제 세계(Physical World)(120)의 기기(IoT Node)(130)에 내장된 임베디드 웹 서버(Embedded Web Server)(140)에 접근할 수 있다. 이때, 임베디드 웹 서버(140)가 기기(130)에 내장되어 있기 때문에 기기(130)에서 생성되는 데이터, 그리고 특정 기능이나 서비스의 처리를 위한 프로세스 로직 모두가 기기(130)에 탑재되어 있다. 따라서, 기기(130)가 제공하는 서비스 영역을 벗어나, 다른 서비스 영역이나 다른 서비스의 데이터와 연동된 또는 융합된 서비스를 제공하는 것이 불가능하다.
- [0008] 도 2는 종래기술에 있어서, 제조사가 별도의 웹 서버를 온라인상에서 제공하는 경우의 서비스 제공 방식을 도시한 도면이다. 도 2에서는 상술한 도 1에서와 달리, 제조사(또는 별도의 서비스 제공자)가 온라인(110)상에 웹 서버(Web Server)(210)를 구축하여 서비스를 제공하는 방식을 도시하고 있다. 이때, 사용자는 온라인(110)상의 웹 서버(210)를 통해 기기(IoT Node)(220)에 접근할 수 있다. 이 경우, 기기(220)에는 웹 서버(210)와 통신하기 위한 모듈(도 2의 HTTP 어댑터(HTTP Adaptor)(230))이 탑재될 수 있다. 따라서, 기기(220)에서 생성되는 데이터, 그리고 특정 기능이나 서비스의 처리를 위한 프로세스 로직 모두가 온라인상에서 제공이 가능하다. 필요에 따라, 데이터는 기기(220)에, 프로세스 로직은 웹 서버(210)상에 존재할 수도 있다. 이 경우에는, 제조사(또는 별도의 서비스 제공자)의 웹 서비스 가용 성능과 호스팅의 영속성에 대한 문제점이 발생할 수 있다.
- [0009] 또한, 도 1 및 도 2를 참조하면, 사용자는 웹 서버(210)를 통해 임베디드 웹 서버(140)가 내장된 기기(130)로 접근할 수 없으며, 반대로 기기(130)의 임베디드 웹 서버(140)를 통해 웹 서버(210)나 기기(220)에 접근할 수 없다. 즉, 객체 인스턴스간에 상호 정보 교류가 불가능하여 연동형 또는 융합형 서비스를 제공할 수 없다.
- [0010] 또한, 도 2를 참조하면, 기기(220)가 서비스의 제공을 위해, 센서 등과 같은 사물로부터 얻어진 정보나 사용자의 개인 정보(사용자의 위치 등) 등을 웹 서버(210)로 제공해야 하기 때문에 사용자의 프라이버시에 대한 문제가 발생할 소지가 존재한다.

발명의 내용

해결하려는 과제

- [0011] IoT(Internet of Things), WoT(Web of Things) 또는 WoO(Web of Objects) 환경에서 이기종의 기기들이나 다양한 서비스들을 연동 또는 융합하여 관리하고, 통합적으로 서비스를 제공하기 위한 서비스 제공 방법 및 시스템을 제공한다.

과제의 해결 수단

- [0012] IoT(Internet of Things) 환경에서의 서비스 방법에 있어서, 사용자에게 제공되는 웹사이트의 단계별 설정 기능을 통해 입력된, 사물(Thing)에 대한 가상화 오브젝트(Virtualization Object)의 인스턴스에 대한 정보를 확인하는 단계; 상기 웹사이트의 외부 서비스 연결 기능을 통해 입력된 외부 서비스에 대한 정보를 이용하여 상기 외부 서비스와의 인증을 수행하는 단계; 및 상기 웹사이트의 오픈 API(Application Program Interface) 설정 기능을 통해 입력된 정보를 이용하여 상기 외부 서비스로 상기 오픈 API를 제공하는 단계를 포함하는 서비스 방법이 제공된다.
- [0013] 일측에 따르면, 상기 서비스 방법은 상기 웹사이트를 통해 인스턴스별 아이콘을 표시하는 단계를 더 포함할 수 있다.
- [0014] 다른 측면에 따르면, 상기 아이콘이 표시되는 영역을 통해, 인스턴스에 대한 컨트롤 기능 및 상기 인스턴스의 대표값이 표시되는 것을 특징으로 할 수 있다.
- [0015] 또 다른 측면에 따르면, 상기 영역을 통해 상기 인스턴스의 상태가 더 표시되는 것을 특징으로 할 수 있다.
- [0016] IoT(Internet of Things) 환경에서의 서비스 방법에 있어서, 사용자의 요청을 수신하는 단계; 및 상기 사용자의 요청에 따라 웹사이트를 제공하는 단계를 포함하고, 상기 웹사이트는, 사물(Thing)에 대한 가상화 오브젝트(Virtualization Object)의 인스턴스에 대한 정보를 입력받기 위한 단계별 설정 기능; 외부 서비스와의 인증을 위한 정보를 입력받기 위한 외부 서비스 연결 기능; 및 상기 외부 서비스로 제공하기 위한 오픈 API에 대한 정보를 입력받기 위한 오픈 API 설정 기능을 포함하는 것을 특징으로 하는 서비스 방법이 제공된다.
- [0017] 또 다른 측면에 따르면, 상기 단계별 설정 기능은, 인스턴스의 이름을 입력받기 위한 사용자 인터페이스, 상기 인스턴스의 태그를 입력받기 위한 사용자 인터페이스, IoT 노드의 프로파일이 저장된 URL을 입력받기 위한 사용자 인터페이스, 인스턴스 아이콘을 선택하기 위한 사용자 인터페이스, 인스턴스의 상태를 선택하기 위한 사용자 인터페이스, 컨트롤러를 선택하기 위한 사용자 인터페이스 및 대표값의 단위를 선택하기 위한 사용자 인터페이스 중 복수의 사용자 인터페이스들을 단계적으로 상기 사용자에게 노출하는 것을 특징으로 할 수 있다.
- [0018] 또 다른 측면에 따르면, 상기 외부 서비스 연결 기능은, 상기 외부 서비스에 대한 상기 사용자의 계정 정보를 입력받기 위한 사용자 인터페이스를 포함하는 것을 특징으로 할 수 있다.
- [0019] 또 다른 측면에 따르면, 상기 오픈 API 설정 기능은, 오픈 API의 설명 문서가 포함된 위치 정보를 입력 받기 위한 사용자 인터페이스를 포함하는 것을 특징으로 할 수 있다.
- [0020] IoT(Internet of Things) 환경에서의 서비스 시스템에 있어서, 적어도 하나의 프로그램이 저장된 적어도 하나의 스토리지 유닛; 및 적어도 하나의 프로세서를 포함하고, 상기 적어도 하나의 프로세서는, 상기 적어도 하나의 프로그램의 제어에 따라, 사용자에게 제공되는 웹사이트의 단계별 설정 기능을 통해 입력된, 사물(Thing)에 대한 가상화 오브젝트(Virtualization Object)의 인스턴스에 대한 정보를 확인하는 과정; 상기 웹사이트의 외부 서비스 연결 기능을 통해 입력된 외부 서비스에 대한 정보를 이용하여 상기 외부 서비스와의 인증을 수행하는 과정; 및 상기 웹사이트의 오픈 API(Application Program Interface) 설정 기능을 통해 입력된 정보를 이용하여 상기 외부 서비스로 상기 오픈 API를 제공하는 과정을 처리하는 것을 특징으로 하는 서비스 시스템이 제공된다.
- [0021] IoT(Internet of Things) 환경에서의 서비스 시스템에 있어서, 적어도 하나의 프로그램이 저장된 적어도 하나의 스토리지 유닛; 및 적어도 하나의 프로세서를 포함하고, 상기 적어도 하나의 프로세서는, 상기 적어도 하나의 프로그램의 제어에 따라, 사용자의 요청을 수신하는 과정; 및 상기 사용자의 요청에 따라 웹사이트를 제공하는 과정을 처리하고, 상기 웹사이트는, 사물(Thing)에 대한 가상화 오브젝트(Virtualization Object)의 인스턴스에 대한 정보를 입력받기 위한 단계별 설정 기능; 외부 서비스와의 인증을 위한 정보를 입력받기 위한 외부 서비스 연결 기능; 및 상기 외부 서비스로 제공하기 위한 오픈 API에 대한 정보를 입력받기 위한 오픈 API 설정 기능을 포함하는 것을 특징으로 하는 서비스 시스템이 제공된다.

도면의 간단한 설명

- [0022] 도 1은 종래기술에 있어서, 기기에서 임베디드 웹 서버를 내장하고 있는 경우의 서비스 제공 방식을 도시한 도면이다.
 도 2는 종래기술에 있어서, 제조사가 별도의 웹 서버를 온라인상에서 제공하는 경우의 서비스 제공 방식을 도시한 도면이다.

도 3은 본 발명의 일실시예에 있어서, 인스턴스 호스팅을 위한 전체 시스템 구조를 개괄적으로 도시한 일례이다.

도 4는 본 발명의 일실시예에 있어서, 인스턴스의 인터페이스들에 대한 개념을 설명하기 위한 도면이다.

도 5는 본 발명의 일실시예에 있어서, 인스턴스의 인터페이스들에 대한 기능을 설명하기 위한 도면이다.

도 6은 본 발명의 일실시예에 있어서, 인스턴스 리스트 페이지의 예를 도시한 도면이다.

도 7은 본 발명의 일실시예에 있어서, 인스턴스 매니저를 설명하기 위한 블록도이다.

도 8은 본 발명의 일실시예에 있어서, 인스턴스 상태 질의 기능을 설명하기 위한 흐름도이다.

도 9는 본 발명의 일실시예에 있어서, 인스턴스 상태 제어 기능을 설명하기 위한 흐름도이다.

도 10은 본 발명의 일실시예에 있어서, 인스턴스 설치 제어 기능을 설명하기 위한 흐름도이다.

도 11은 본 발명의 일실시예에 있어서, 사물 피드 제어 기능을 설명하기 위한 블록도이다.

도 12는 본 발명의 일실시예에 있어서, 스케줄 등록 방법을 설명하기 위한 흐름도이다.

도 13 내지 도 15는 본 발명의 일실시예에 있어서, 인스턴스 위자드를 설명하기 위한 도면들이다.

도 16은 본 발명의 일실시예에 있어서, 매시업 인스턴스를 위한 추가 설정 및 외부 서비스 인증 요청을 설명하기 위한 도면이다.

도 17은 본 발명의 일실시예에 있어서, 오픈 API를 설명하기 위한 도면이다.

도 18은 본 발명의 일실시예에 있어서, 서비스 방법을 도시한 흐름도이다.

발명을 실시하기 위한 구체적인 내용

[0023] 이하, 본 발명의 실시예를 첨부된 도면을 참조하여 상세하게 설명한다.

[0024] 도 3은 본 발명의 일실시예에 있어서, 인스턴스 호스팅을 위한 전체 시스템 구조를 개괄적으로 도시한 일례이다. 도 3은 IoT 프로파일(IoT(Internet of Things) Profile)(310), IoT 노드(IoT Node)(320), 인스턴스 호스팅 게이트웨이(Instance Hosting Gateway)(330) 및 인스턴스 호스팅 플랫폼(Instance Hosting Platform)(340)을 도시하고 있다.

[0025] IoT 프로파일(310)은 IoT 노드(320)의 제조사나 개발자가 온라인상에서 제공하는 IoT 노드(320)에 대한 정보를 의미할 수 있다. 예를 들어, 이러한 IoT 프로파일(310)은 IoT 노드(320)에 대한 정보를 사용자가 수동으로 인스턴스 호스팅 게이트웨이(330)에 데이터를 입력해야 하는 불편함을 최소화하기 위해, 제조사나 개발자가 IoT 노드(320)에 대한 정보를 기정의된 XML(eXtensible Markup Language) 스킴(scheme)에 따라 처리하여 생성한 XML 파일을 의미할 수 있다.

[0026] 이러한 IoT 프로파일(310)은 IoT 노드(320)에 대한 설명(Description)과 IoT 노드(320)의 동작에 관해 기술된 프로세스(Process)에 대한 식별자(일례로, 프로세스에 대한 정보가 포함된 문서의 URL)를 포함할 수 있다. 예를 들어, 프로세스는 별도의 파일(일례로, 확장자 '.js' 파일)로 온라인상에서 제공될 수 있으며, 프로세스에 대한 식별자는 이러한 파일의 위치를 나타내는 URL일 수 있다. 아래 표 1은 IoT 프로파일(310)이 포함할 수 있는 정보들의 일례를 나타내고 있다.

표 1

[0027]

Name	Description	비고
IoTProfileID	IoT Profile의 위치를 고유 URL	URI 형식으로 표현
IoTProfileVer	IoT Profile의 버전	'V'+숫자 형식'으로 표현
IoTProfileDesc	IoT Profile의 설명(Description)	
IoTInput	IoT Node의 입력값(Input 기능을 제공하지 않는 인스턴스 노드의 경우에는 필요 없음), RESTful API 생성을 위한 정보	데이터 지원 형식 - JSON - XML
IoTOutput	IOT Node의 출력값, RESTFUL API 생성을 위한 정보	
IoTProcessID	IoT Node의 프로세스 식별자	URI 형식으로 표현

- [0028] IoT 노드(320)는 센서 기기와 같은 IoT 환경의 사물(Things)에 대응할 수 있으며, 'IoTNodeID'와 같은 자신의 식별자 정보를 제공할 수 있다. 만약, IoT 노드(320)가 스마트 기기와 같은 NC(Non-Constraint) 노드인 경우에는 IoT 노드(320)가 상술한 IoT 프로파일(310)을 직접 제공할 수도 있다.
- [0029] 도 3에서는 설명의 편의를 위해, 하나의 IoT 노드(320)와 IoT 노드(320)에 대한 하나의 IoT 프로파일(310)을 도시하고 있으나, IoT 환경에서는 다수의 IoT 노드들과 IoT 노드들 각각에 대한 IoT 프로파일들이 활용될 수 있는 너무나 당연하다. 또한, 하나의 IoT 노드에 대한 복수의 IoT 프로파일들이 제공될 수도 있다.
- [0030] 하나의 IoT 노드와 인스턴스(instance)는 1:1, 1:N, N:1, N:M과 같이 다양하게 관계 지어질 수 있다. 또한, 인스턴스는 물리적인 IoT 노드가 없더라도, 프로세스의 형태로 존재 가능하며, 매시업(Mash-up) 프로세스 등과 같이 특정 장치와 연동되는 형태가 아닌 프로세스간 연동을 통해 인스턴스가 생성 및 연동될 수 있다. 예를 들어, 별도의 IoT 노드와 연결하지 않더라도 특정 프로세스가 인스턴스 호스팅 게이트웨이(330)에 설치(deploy)되어 동작함으로써, 인스턴스 호스팅 게이트웨이(330)에서 인스턴스가 생성될 수 있다. 본 명세서에서는 전반적으로 물리적인 장치로서 설명되는 IoT 노드(물리적 노드)와 본 발명의 실시예들에 따른 인스턴스 호스팅 게이트웨이간의 통신을 통해 IoT 노드로부터 수신되는 메시지를 개별 인스턴스와 연결하여 처리하는 실시예를 설명하고 있으나, 상술한 바와 같이, 적어도 하나의 IoT 노드는 물리적인 장치가 아닌 프로세스의 형태(프로세스 노드)로 존재할 수도 있다.
- [0031] 또한, 센서 디바이스 또는 액추에이터 등과 같은 물리적인 장치 없이 다수의 센서로부터 정보를 받아 이를 온라인상의 다른 웹 서비스 또는 어플리케이션과 연동 기능만을 제공하는 프로세스가 제공될 수도 있다. 예를 들어, 센서들의 정보와 소셜 서비스 또는 사용자가 이용하는 특정 어플리케이션의 상태 정보를 분석해 특정 결과를 제공하는 서비스의 경우, 해당 서비스를 동작시킬 별도의 프로세싱 환경이 필요하다. 이 경우, 본 발명의 실시예들에 따른 인스턴스 호스팅 게이트웨이에서 상기 프로세싱 환경을 제공해줄 수 있다.
- [0032] 인스턴스 호스팅 게이트웨이(330)는 IoT 노드(320)로부터 수신되는 메시지를 개별 인스턴스와 연결해 처리하고, 사용자 또는 서비스의 요청에 따라 외부 서비스와 연동될 수 있다. 이러한 인스턴스 호스팅 게이트웨이(330)는 IoT 노드(320)를 위한 AP(Access Point) 환경에 탑재될 수 있다. 예를 들어, 인스턴스 호스팅 게이트웨이(330)는 AP에 탑재되어 웹 어플리케이션 서버(Web Application Server, WAS) 형태로 동작하는 인스턴스 어플리케이션 서버(Instance Application Server) 부분과 AP상에 소프트웨어 형태, 하드웨어 형태 또는 소프트웨어와 하드웨어의 결합 형태로 탑재(일례로, AP의 운영체제에 소프트웨어의 형태로 탑재되는 모듈과 IoT 노드(320)와의 통신을 위해 하드웨어의 형태로 탑재되는 모듈)되는 부분으로 구성될 수 있다.
- [0033] 일례로, 인스턴스 호스팅 게이트웨이(330)는 특정 공간(일례로, 집안)에 설치된 AP에 포함될 수 있으며, 상기 특정 공간에 포함된 사물들(IoT 노드들)과 통신하여 데이터를 수신 및 처리하고, 외부 서비스와의 연동 서비스를 제공하기 위한 기능을 수행할 수 있다.
- [0034] 이때, 인스턴스 호스팅 게이트웨이(330)는 새로운 IoT 노드(320)를 발견하면, IoT 프로파일(310)을 다운로드하여 설치할 수 있다. 예를 들어, 사용자는 IoT 프로파일(310)에 대응하는 XML 파일의 위치를 인스턴스 호스팅 게이트웨이(330)에 입력할 수 있으며, 인스턴스 호스팅 게이트웨이(330)는 입력된 위치에 기초하여 XML 파일을 다운로드하여 설치할 수 있다.
- [0035] 이러한 인스턴스 호스팅 게이트웨이(330)에 대해서는 이후 도 4 내지 도 7을 통해 더욱 자세히 설명한다.
- [0036] 인스턴스 호스팅 플랫폼(340)은 인스턴스간 상호 연동 기능과 매시업(Mash-up) 기능을 제공할 수 있다. 예를 들어, 인스턴스 호스팅 게이트웨이(330)는 사용자 또는 서비스의 요청에 따라 제공하기 위한 기능을 API(Application Program Interface) 형태로 외부로 제공하여 외부 서비스와 연동하기 위해, 인스턴스 호스팅 플랫폼(340)과 통신할 수 있다.
- [0037] 도 4는 본 발명의 일실시예에 있어서, 인스턴스의 인터페이스들에 대한 개념을 설명하기 위한 도면이다. 사물들(Things, 410)의 가상화객체들(VO(Virtualization Objects), 420)에 있어, 가상화객체와 사물간의 연결, 그리고 가상화객체와 다른 가상화객체간의 연결에 필요한 기본적인 인터페이스가 필요하다. 본 실시예에서는 인터페이스의 효과적이고 명시적인 사용을 위해 총 5가지의 인스턴스용 인터페이스 기능을 제공하는 예들을 설명한다.
- [0038] 개별 인터페이스의 정보는 각 인스턴스별로 인스턴스 프로파일에 이를 명시적으로 기술하여 사용할 수 있으며,

도 3을 통해 설명한 인스턴스 호스팅 게이트웨이(330)가 포함할 수 있는 인스턴스 매니저에서 해당 인스턴스의 정보에 기반하여 관련 기능을 제공할 수 있다. 이때, 도 4에서는 사물들(410) 각각과 관련된 인스턴스(Associated Thing Instance)들 및 인스턴스간 연동을 위한 매시업 인스턴스(Mashup Instance)를 나타내고 있다. 이때, 매시업 인스턴스는, 인스턴스 어플리케이션 서버(IAS(Instance Application Server), 430) 및 써드 파티 서비스(3rd Party Service, 440)와의 연동을 가능하게 할 수 있다.

- [0039] 인스턴스 어플리케이션 서버(430)는 인스턴스 호스팅 게이트웨이(330)의 웹 어플리케이션 서버 기능을 담당할 수 있다. 이때, 인스턴스 어플리케이션 서버(430)를 통해 제공되는 기능들은 웹 인터페이스로 제공될 수 있다. 또한, 써드 파티 서비스(440)는 사용자와 연관된 소셜 네트워크 서비스(Social Network Service, SNS)와 같이 다양한 외부 서비스들을 의미할 수 있다.
- [0040] 도 5는 본 발명의 일실시예에 있어서, 인스턴스의 인터페이스들에 대한 기능을 설명하기 위한 도면이다. 도 4의 가상화객체들(420)인 인스턴스들은 인스턴스 매니저가 제공하는 5가지의 기 제공된 인터페이스를 이용하여, (1) 사물과 인스턴스간, (2) 인스턴스와 인스턴스간, (3) 외부 서비스(써드 파티 서비스(440))와 인스턴스간, 그리고 (4) 인스턴스 호스팅(Instance Hosting, IH) 응용 기능(일례로, 도 4의 인스턴스 아이콘(Instance Icon))과 인스턴스간에 데이터를 주고 받을 수 있다.
- [0041] 피더(Feeder)를 의미하는 F(510)는 인스턴스가 인스턴스 또는 외부 서비스로 제공하고자 하는 정보를 노출하는 인터페이스를 포함할 수 있다. 이때, F(510)는, 직접 메시지를 전달하는 것이 아닌 인스턴스의 메시지 큐(message queue)상에 피딩(feeding)할 정보를 저장하는 형태로 동작할 수 있으며, 다음 (1) 내지 (3)과 같은 경우에 활용될 수 있다.
- [0042] (1) 장치(사물)로부터 받은 메시지를 관독 후 타 서비스 또는 다른 인스턴스로 전달하고자 하는 경우.
- [0043] (2) 인스턴스내에서 특정 상태 값을 노출하거나 또는 외부 서비스(일례로, 써드 파티 서비스(440))에게 전달하고자 하는 값을 노출하고자 하는 경우.
- [0044] (3) 인스턴스의 상태 값 중 인스턴스 리스트 상의 인스턴스 아이콘(일례로, 인스턴스 어플리케이션 서버(430)의 인스턴스 아이콘)을 통해 사용자에게 대표로 노출시킬 값(대표 속성(representative attribute) 값)을 설정하고자 하는 경우.
- [0045] 서브스크라이버(Subscriber)를 의미하는 S(520)는 사물 또는 타 인터페이스의 피더로부터 정보를 수신하는 인터페이스를 포함할 수 있으며, 다음 (4) 및 (5)의 경우에 활용될 수 있다.
- [0046] (4) 장치(사물)로부터 수신되는 정보를 전달 받기 위해 사용되는 경우(이 경우, 인스턴스 호스팅 게이트웨이(330)가 포함하는 네트워크 어댑터가 리스닝(listening)할 사물의 물리적 인터페이스에 대한 정보가 관련 사물 정보 엘리먼트(associatedThingInformation element)에 기술될 수 있음).
- [0047] (5) 다른 인스턴스의 피더에 대한 정보를 구독하는 경우(타 인스턴스의 피더 인터페이스로 노출되는 값을 활용하고자 하는 경우) - 타 인스턴스의 피더에 대한 정보를 구독하는 경우, 필터에 정의된 조건에 맞는 인터페이스를 통해 정보의 구독이 가능.
- [0048] 리시버(Receiver)를 의미하는 R(530)은 인스턴스가 타 인스턴스의 컨트롤러로부터 메시지를 전달받는 인터페이스를 포함할 수 있으며, 다음 (6) 내지 (8)과 같은 경우에 활용될 수 있다.
- [0049] (6) 타 인스턴스로부터 정보를 받는 경우(타 인스턴스의 컨트롤러 인터페이스로부터 정보를 수신).
- [0050] (7) 인스턴스 아이콘상의 대표 제어값(피더상에서 대표 속성 값을 설정한 경우)을 수신하는 경우(사용자의 직접 제어를 처리하는 경우).
- [0051] (8) 외부 서비스(일례로, 써드 파티 서비스(440))로부터 메시지를 받는 경우(일례로, 캘린더 서비스의 사용자 상태 정보를 수신하는 경우).
- [0052] 컨트롤러(Controller)를 의미하는 C(540)는 인스턴스가 외부로부터 정보를 전달하기 위한 인터페이스를 포함할 수 있으며, 다음 (9) 및 (10)의 경우에 활용될 수 있다.
- [0053] (9) 장치(사물)에게 (제어) 메시지를 전달하는 경우(관련 사물 정보 엘리먼트가 설정되어 있는 장치(사물)로 메시지를 전달하는 경우).
- [0054] (10) 타 인스턴스에게 메시지를 전달하는 경우(타 인스턴스의 리시버 인터페이스 메시지 큐에 들어가는 형태)

아닌, 트리거링 방식을 통해 데이터 전달).

[0055] 인증(0Auth)을 의미하는 0(550)는 인스턴스가 써드 파티 서비스로 연동하기 위한 인터페이스를 포함할 수 있으며, 다음 (11) 및 (12)의 경우에 활용될 수 있다.

[0056] (11) 써드 파티 서비스로 메시지를 전달하는 경우(0Auth 인증이 이루어진 후).

[0057] (12) 써드 파티 서비스로 메시지를 수신(질의)하는 경우.

[0058] 이때, 상술한 (8)과 (12)는 모두 외부 서비스로부터 메시지를 수신하는 경우를 나타내고 있다. 외부 서비스로부터의 메시지는 실시예에 따라 리시버를 통해 수신될 수도 있고, 0(550)를 통해 수신될 수도 있다.

[0059] 본 발명의 실시예들에 따른 인스턴스는 개별적인 인터페이스 정보에 대해 각각의 메시지 큐를 생성할 수 있으며, 인터페이스별 메시지 큐에는 기설정된 수(일례로, 100건)의 메시지가 저장될 수 있다. 해당 인스턴스가 비정상적으로 종료되어도(일례로, 에러 상태(error)로 변경 시) 저장된 메시지는 남아 있으며, 다시 정상 동작 상태(run)로 변경 시, 저장된 메시지를 이용할 수 있다. 이때, 인스턴스가 에러 상태일 때 수신되는 메시지들은 인스턴스의 상태와 관계 없이 개별 인터페이스의 메시지 큐상에 저장될 수 있다.(서브스크라이버 인터페이스나 리시버 인터페이스의 경우)

[0060] 이후에서는, 인스턴스가 포함하는 인터페이스들에 대해 더욱 자세히 설명한다.

[0061] 1. 서브스크라이버는 이미 설명한 바와 같이 인스턴스가 사물장치(Thing) 또는 타 인스턴스의 피더 인터페이스로부터 메시지를 수신 및 구독하기 위한 인터페이스일 수 있다. 사물로부터 메시지를 수신하는 경우, 해당 서브스크라이버는 메시지의 수신에 필요한 물리적 인터페이스의 정보를 해당 사물 정보 'associatedThingInformation' 엘리먼트상에 기재하여야 하며, 이때 필요한 정보는 기본적인 장치의 물리 인터페이스를 어댑테이션(adaptation)하기 위한 정보들을 포함할 수 있다. 또한, 타 인스턴스로부터 메시지를 수신하는 경우, 해당 서브스크라이버는 구독(Subscribe)할 타 인스턴스의 피더 인터페이스에 대한 정보를 필터(Filter) 엘리먼트상에 기재하여야 하며, 피더 인터페이스에 대한 정보는, 해당 인스턴스를 등록하는 과정(Instance Wizard를 이용하여 등록)에서 구독할 인스턴스의 정보를 입력하는 것으로 필터 엘리먼트상에 기재될 수 있다. 이때, 트리거 속성(Trigger attribute)을 'true'로 설정할 경우 해당 피더의 값이 생성되면, 인스턴스 매니저는 그 즉시 이를 구독하는 서브스크라이버 인스턴스를 호출할 수 있다. 해당 동작은 인스턴스 동작 주기 'InstanceRunningPeriod' 값과 상관없이 수행될 수 있다.

[0062] 서브스크라이버가 가질 수 있는 속성은 다음 표 2의 일례와 같이 표현될 수 있다.

표 2

[0063]

```
<xs:element name="subscriber"> <xs:complexType>
<!--인스턴스(Instance) 별로 인터페이스(I/F)를 식별하기 위한 식별자로 사용. 자릿수는 최대 3자리
까지 이용 가능.-->
<xs:attribute name="id" type="xs:unsignedShort" />
<!-- NAME은 인스턴스를 손쉽게 구분하기 위한 정보로 활용 가능하며, 향후 인스턴스간 연결 시 해당
인스턴스의 I/F 구분을 위해 사용자에게 제공되는 정보로 활용 가능.-->
<xs:attribute name="name" type="xs:string" />
<!-- 타 인스턴스를 구독시 해당 값이 생성되는 경우 즉시 반응의 여부에 따라 설정, 인스턴스 매니저
가 피더 리스트의 트리거링(triggering) 설정 여부에 대해 모니터링 후 즉시 데이터 전달 -->
<xs:attribute name="trigger" type="xs:boolean" />
</xs:complexType> </xs:element>
```

[0064] 또한, 서브스크라이버내의 필터가 가질 수 있는 속성은 다음 표 3의 일례와 같이 표현될 수 있다.

표 3

[0065]

```
<xs:element name="filter"><xs:complexType>
<!-- 다른 인스턴스의 피더를 구독하는 경우 필터에 정의된 조건에 맞는 인터페이스만 구독이 가능 -->
</xs:complexType>
</xs:element>
</xs:sequence>
```

[0066]

2. 피더는 인스턴스가 타 인스턴스로 정보를 노출하기 위한 인터페이스로, 구독하는 인터페이스의 설정에 따라, 즉시 값이 전달되거나 또는 타 인스턴스의 동작시, 값이 타 인스턴스의 입력값으로 전달될 수 있다. 이때, 개별 피더 식별자(ID) 별로 인스턴스 매니저상에서 메시지 큐를 생성하고, 데이터(메시지)를 메시지 큐에 저장한 후 서브스크라이버 인터페이스를 통해 읽어가게 된다.

[0067]

피더의 메시지 전달 방식과 관련하여, 피더로부터 메시지를 읽어가는 방식은 서브스크라이버 인터페이스 상에서 트리거 설정여부에 따른 방식과 구독하는 인스턴스의 실행에 따라 입력 값으로 전달되는 방식으로 나뉘게 된다.

[0068]

해당 피더에 대해 트리거를 설정한 서브스크라이버 인터페이스의 경우 해당 서브스크라이버 인터페이스가 구독하는 피더에 대한 값이 등록될 시 호출되게 된다. 그렇지 않은 경우 인스턴스 매니저가 해당 피더를 구독(Subscribe) 하는 타 인스턴스의 실행 시 데이터를 읽어 입력 값으로 전달하게 된다. 이때, 인스턴스 매니저가 해당 피더의 값을 읽어 갔다고 하더라도 곧 바로 메시지 큐에 데이터를 지우지는 않는다. 해당 메시지의 큐의 크기는 인터페이스별로 기설정된 수(일례로, 100건)를 가질 수 있으며, 해당 인스턴스가 비정상 종료되어도 해당 값은 남아 있게 된다.

[0069]

피더는 다음 표 4와 같이 표현되는 속성을 사용할 수 있으며 대표값을 나타내는 'representative'를 사용(true)하는 경우, 단위(unit)를 지정해야 인스턴스 매니저의 리스트 페이지상에서 해당 데이터와 연동되는 온도, Kcal 등의 단위가 표시될 수 있다.

표 4

[0070]

```
<xs:attribute name="id" type="xs:unsignedShort" />
<!-- 인스턴스별로 인터페이스를 식별하기 위한 식별자로 사용가능. 자릿수는 최대 3자리까지 이용가능.-->
<xs:attribute name="name" type="xs:string" />
<!-- NAME은 인스턴스를 손쉽게 구분하기 위한 정보로 활용되며, 향후 인스턴스간 연결 시, 해당 인스턴스의 인터페이스(I/F) 구분을 위해 사용자에게 제공되는 정보로 활용.-->
<xs:attribute name="unit" type="xs:string" />
<!-- 해당 인터페이스를 통해 주고 받는 데이터 값을 정의하는데 사용-->
<xs:attribute name="representative" type="xs:boolean" />
<!-- 대표값을 설정하는 부분으로 해당 값을 설정할 경우 인스턴스 리스트 페이지상의 인스턴스 아이콘에 해당 값이 출력되게 됨 -->
```

[0071]

도 6은 본 발명의 일실시예에 있어서, 인스턴스 리스트 페이지의 예를 도시한 도면이다. 도 6은 복수의 인스턴스 아이콘들을 포함하는 페이지의 일부가 표시된 화면(600)을 나타내고 있다. 이때, 화면(600)에서 인스턴스 아이콘에 피더 인스턴스의 대표값 속성이 설정된 예들이 점선타원들(610)을 통해 나타나 있다.

[0072]

3. 리시버는 타 인스턴스로부터 메시지를 호출 받기 위한 인터페이스로서, 타 인스턴스의 컨트롤러 인터페이스를 통해 정보를 전달 받을 수 있다. 타 인스턴스로 부터 전달받은 메시지는 수신한 인스턴스에게 직접 전달되

게 되고, 해당 인스턴스의 트리거링(실행)이 일어나게 된다. 이러한 트리거링은 메시지를 수신한 인스턴스의 동작 주기를 나타내는 'runningPeroid' 설정과 상관없이 수행될 수 있다.

[0073]

리시버는 다음 표 5와 같이 나타나는 속성을 사용할 수 있으며, 대표값을 나타내는 'representative'를 사용(true)하는 경우, 단위(unit)를 지정해야 인스턴스 매니저의 리스트 페이지상에서 해당 인스턴스를 제어하는 버튼이 생성될 수 있다.

표 5

[0074]

```
<xs:attribute name="id" type="xs:unsignedShort" />
<!-- 인스턴스별로 인터페이스를 식별하기 위한 식별자로 사용. 자릿수는 최대 3자리까지 이용 가능-->
<xs:attribute name="name" type="xs:string" />
<!-- NAME은 인터페이스를 손쉽게 구분하기 위한 정보로 활용되며, 향후 인스턴스간 연결 시 해당 인스턴스의 인터페이스(I/F) 구분을 위해 사용자에게 제공되는 정보로 활용-->
<xs:attribute name="unit" type="xs:string" />
<!-- 해당 인터페이스를 통해 주고 받는 데이터 값을 정의하는데 사용된다 -->
<xs:attribute name="representative" type="xs:boolean" />
<!-- 대표값을 설정하는 부분으로 해당 값을 설정할 경우 인스턴스 리스트 페이지상의 인스턴스 아이콘에 해당 값이 출력되게 됨 -->
```

[0075]

다시 도 6을 살펴보면, 도 6의 화면(600)에 표시된 아이콘들 중 활성화된 아이콘들에 "ON", "OFF", "Locked"와 같이 인스턴스 아이콘에 대응하는 인스턴스를 제어하기 위한 버튼이 생성된 모습을 나타내고 있다.

[0076]

4. 컨트롤러는 인스턴스가 타 인스턴스 또는 장치(사물)로 메시지를 전달하기 위한 인터페이스를 포함할 수 있다. 컨트롤러는, 타 인스턴스로 메시지를 전달하기 위해 직접 전달 방식을 통해 해당 인스턴스를 호출할 수 있으며, 메시지는 리시버 인터페이스를 통해 해당 인스턴스로 전달될 수 있다. 예를 들어, 메시지를 수신한 인스턴스의 동작 주기 'runningPeroid'의 설정과 상관없이 메시지의 전달이 트리거링 될 수 있다. 메시지를 장치(사물)로 전달하는 경우, 컨트롤러는 네트워크 어댑터를 통해 메시지를 즉시 전달할 수 있다.

[0077]

타 인스턴스로 메시지를 전달하는 경우, 타 인스턴스의 리시버 인터페이스의 정보가 필터 엘리먼트내에 기술되어야 한다. 이때, 필터 엘리먼트내에 기술되는 정보는 다음 표 6과 같이 표현될 수 있다.

표 6

[0078]

```
<xs:element name="filter"><xs:complexType>
<!-- 다른 인스턴스의 리시버로 메시지를 전달하는 경우 필터에 정의된 조건에 맞는 인터페이스에 대해서만 메시지 전달이 가능 -->
<xs:attribute name="instanceId" type="xs:string"/>
<!-- 메시지를 전달할 인스턴스의 ID 값(고유값) -->
<xs:attribute name="receiverId" type="xs:unsignedShort"/>
<!-- 메시지를 전달할 인스턴스 내의 리시버 인터페이스의 ID 값(고유값) -->
<xs:attribute name="minVersion" type="xs:string"/>
<!-- 전달할 인스턴스 지원 버전의 최소 값을 입력 -->
<xs:attribute name="maxVersion" type="xs:string"/>
<!-- 전달할 인스턴스 지원 버전의 최대 값을 입력 -->
</xs:complexType></xs:element>
```

[0079]

또한, 컨트롤러는 다음 표 7과 같은 속성을 가질 수 있다.

표 7

[0080]

```
<xs:attribute name="id" type="xs:unsignedShort" />
<!-- 인스턴스별로 인터페이스를 식별하기 위한 식별자로 사용. 자릿수는 최대 3자리까지 이용 가능-->
<xs:attribute name="name" type="xs:string" />
<!-- NAME은 인터페이스를 손쉽게 구분하기 위한 정보로 활용되며, 향후 인스턴스간 연결 시 해당 인스턴스의 인터페이스(I/F) 구분을 위해 사용자에게 제공되는 정보로 활용-->
</xs:complexType> </xs:element>
```

[0081]

5. 인증 인터페이스(이하 'OAuth')는 인스턴스가 외부 써드 파티 서비스와 OAuth 기반 인증을 통한 연동 시 이용하는 인터페이스를 포함할 수 있다. 이를 위해, 사용자는 자신이 연결하고자 하는 대상 서비스의 OAuth 계정 인증을 선 진행하여야 하며, 이러한 OAuth 계정 인증은 인스턴스 위자드(Instance Wizard) 또는 OAuth 메뉴를 통해 별도로 진행될 수 있다.

[0082]

개별 인스턴스 별로, 연동하기 위한 OAuth의 인증 정보가 설정 또는 설정 취소 될 수 있다. 예를 들어, 다수의 소셜 네트워크 서비스 계정을 각각의 인스턴스별로 다르게 설정/취소할 수 있다.

[0083]

인스턴스는 OAuth의 필요에 따라 해당 OAuth 연동이 필수인지 여부를 결정할 수 있다. 해당 정보는 'required' 속성을 통해 나타내며, 'required' 속성이 'true'로 설정된 경우, OAuth 연동이 정상적으로 이루어지지 않으면 해당 인스턴스는 동작(running) 상태가 될 수 없으며, 등록 후 인스턴스는 에러(error) 상태로 남게 된다.

[0084]

OAuth 인터페이스는 다음 표 8과 같은 속성을 가질 수 있다.

표 8

[0085]

```
<xs:element name="oAuth"> <xs:complexType>
<!-- 인스턴스별로 인터페이스(I/F)를 식별하기 위한 식별자로 사용. 자릿수는 최대 3자리까지 이용 가능-->
<xs:attribute name="id" type="xs:unsignedShort" />
<!-- OAuth를 연동할 서비스 명을 기입. 일례로, 트위터(Twitter)는 'twitter', 페이스북(Facebook)은 'facebook', 구글(Google)은 'Google'과 같이 입력-->
<xs:attribute name="serviceName" type="xs:string" />
<!-- 연동할 OAuth 계정에서 이용할 정보에 대해 사용자에게 알려주는 설명을 기재. 예를 들어 어떠한 용도로 어느 정보에 접근하는지에 관한 정보 기술-->
<xs:attribute name="description" type="xs:string" />
</xs:complexType> </xs:element>
```

[0086]

이후에서는 인스턴스 호스팅을 위한 시스템이 지원하는 인스턴스의 데이터 타입에 대해 설명한다. 인스턴스의 데이터 타입의 경우, 인스턴스간의 메시지 전달 시 데이터 타입에 따른 처리의 명확성 이외에도 인스턴스 아이콘상에 디스플레이되는 단위 표시 부호를 나타내는 위해 더 이용될 수 있다.

[0087]

인스턴스의 데이터 타입은 인스턴스의 인터페이스별로 제공하는 단위 정보의 데이터 형을 의미할 수 있다. 인스턴스 호스팅을 위한 시스템에서는 보다 개발의 용이성과 사용자 직관적인 정보의 활용을 위해 기본적인 데이터 타입(일례로, W3C(World Wide Web Consortium) WebIDL(Web Interface Definition Language) 기준)을 제공하고, 일반적인 IoT 장치들의 사용 시 자주 사용하게 되는 데이터 셋을 별도로 제공할 수 있다. 일례로, 데이터 타입은 아래 표 9와 같이 제공될 수 있다.

표 9

[0088]

데이터 타입(Data Type)	설명(Description)	아이콘 표시(Icon Display)
percent	비율	%
onoff	온/오프	Control Icon
celsius	섭씨	° C
lockunlock	락/언락	Control Icon
lux	조명도의 실용단위	Lux

kcal	칼로리 단위	Kcal
Cal	칼로리 단위	Cal
steps	스텝	Steps
floor	층	F
kilogram	질량의 단위	KG
gram	질량의 단위	G
hour	시간의 단위	H
minute	시간의 단위	min

[0089] 표 9의 예 외에도, 강수량 등을 나타내기 위한 밀리미터(mm) 등과 같이 다양한 데이터 타입이 이용될 수 있다.

[0090] 인스턴스 호스팅에서는 사물 및 비즈니스 로직의 가상객체 생성 시 필수 정보 수집을 위한 스키마 구조가 제시될 수 있다. 주요한 구조로는 인스턴스 정보 기록부, 사물의 가상 객체화 시 이를 연동하기 위한 네트워크 어댑터 정보의 기술부, 인스턴스 별로 정의된 인터페이스 정보의 기술부 그리고 JS(JavaScript) 코드 기술부를 포함할 수 있으며, 그 내용은 일례로, 아래 표 10과 같이 제공될 수 있다.

표 10

```
[0091] <!-- 최상위 element로 instance 태그 사용 -->
<xs:element name="instance">
<xs:complexType><xs:sequence>

<!-- 최대 32자의 길이를 가지는 숫자, 영문, '_' 조합의 unique ID로 기설정된 웹페이지를 통해 발급 받으며, 발급 시 Instance 관련 기본 정보를 등록하게 됨, Instance ID 값은 해당 Instance Profile의 대표 이름 값을 나타냄 -->
<xs:element name="instanceId">
<xs:simpleType> <xs:restriction base="xs:string">
<xs:pattern value="[a-zA-Z0-9_]+" />
<xs:maxLength value="32" />
</xs:restriction> </xs:simpleType>
</xs:element>

<!-- 최대 16자의 길이를 가지는 version -->
<xs:element name="instanceVersion">
<xs:simpleType> <xs:restriction base="xs:string">
<xs:pattern value="[0-9](.[0-9])+" />
<xs:maxLength value="16" />
</xs:restriction> </xs:simpleType>
</xs:element>

<!-- Instance의 정보를 가지고 있는 대표 url -->
<xs:element name="instanceDetailURL" type="xs:string" />
<xs:element name="instanceDescription" type="xs:string" />
<!-- 인스턴스의 등록 및 등록 후 사용할 대표 Icon의 물리적 위치 (Icon은 Png, Jpg 형식이어야 하며 사이즈는 최대 96by96을 지원한다)-->
<xs:element name="instanceIconURL" type="xs:string" minOccurs="0" />
<!-- 인스턴스의 동작주기는 -1, 1이상의 정수로 최대 decimal 값의 크기를 따른다 -->
<xs:element name="instanceRunningPeriod" type="xs:decimal" />

<!-- 사물을 가상객체화 한 경우 사물과의 연동 인터페이스를 기술하기 위해 associatedThingInformation 엘리먼트상에 네트워크 어댑터 관련 정보를 기술하여야 한다 -->
<xs:element name="associatedThingInformation"> <xs:complexType> <xs:all minOccurs="1">
```

```
<!-- 사물의 명칭 정보를 입력 -->
<xs:element name="thingName" type="xs:string" />
<!-- 사물의 대표 이미지 정보 입력 -->
<xs:element name="thingImageURL" type="xs:string" />
<!-- 사물의 부가 정보 입력 -->
<xs:element name="thingDescription" type="xs:date" />

<!-- 사물의 정보 제공(송신) 물리 인터페이스 가상화-->
<xs:element name="feederInformation"> <xs:complexType>
<!-- 사물로부터 전달 받은 메시지를 인스턴스의 Subscriber Interface와 연결하기 위한 정보로 해당
어트리뷰트의 값은 연결하고자 하는 Subscriber Interface의 ID 값 기술한다. -->
<xs:attribute name="connectedInterfaceId" type="xs:unsignedShort"/>
<!-- 사물과의 연동시 사용할 실질적인 물리 인터페이스 (zigbee, wifi)의 정보를 입력 -->
<xs:attribute name="interface" type="xs:string"/>
<!-- 사물과의 연동시 사물의 정보를 송신할 MAC 주소값 -->
<xs:attribute name="src_mac_addr" type="xs:string"/>
```

```

<!-- 사물과의 연동시 사물의 IP 주소값 -->
<xs:attribute name="src_ip_addr" type="xs:string"/>
<!-- 사물과의 연동시 사물이 전달하는 메시지의 destination IP 주소값 (메시지를 Network Adaptor를
통해 후킹하는 경우 이용하게 되는 정보) -->
<xs:attribute name="dst_ip_addr" type="xs:string"/>
<!-- 사물과의 연동시 사물의 TCP Port 값 -->
<xs:attribute name="src_tcp_port" type="xs:string"/>
<!-- 사물과의 연동시 사물이 전달하는 메시지의 destination TCP Port 값 (메시지를 Network Adaptor
를 통해 후킹하는 경우 이용하게 되는 정보) -->
<xs:attribute name="dst_tcp_port" type="xs:string"/>
<!-- 사물과의 연동시 사물의 UDP Port 값 -->
<xs:attribute name="src_udp_port" type="xs:string"/>
<!-- 사물과의 연동시 사물이 전달하는 메시지의 destination UDP Port 값 (메시지를 Network Adaptor
를 통해 후킹하는 경우 이용하게 되는 정보) -->
<xs:attribute name="dst_udp_port" type="xs:string"/>
<!-- 사물과의 연동시 HTTP 메시지 통신 방식을 사용하는 경우 HTTP 요청을 위한 주소값-->
<xs:attribute name="http_request_uri" type="xs:string"/>
</xs:complexType> </xs:element>

<!-- 사물의 정보 수신 물리 인터페이스 가상화-->
<xs:element name="receiverInformation"> <xs:complexType>
<!-- 사물이 수신할 메시지에 대한 인스턴스의 Controller Interface와 연결하기 위한 정보로 해당 어
트리뷰트의 값은 연결하고자 하는 Controller Interface의 ID 값 기술한다. -->
<xs:attribute name="connectedInterfaceId" type="xs:unsignedShort"/>
<!-- 사물과의 연동시 사용할 실질적인 물리 인터페이스 (zigbee, wifi)의 정보를 입력 -->
<xs:attribute name="interface" type="xs:string"/>
<!-- 사물과의 연동시 사물이 수신할 인터페이스의 MAC 주소값 -->
<xs:attribute name="dst_mac_addr" type="xs:string"/>
<!-- 사물과의 연동시 사물의 IP 주소값 -->
<xs:attribute name="src_ip_addr" type="xs:string"/>
<!-- 사물과의 연동시 사물이 전달하는 메시지의 destination IP 주소값 (메시지를 Network Adaptor를
통해 후킹하는 경우 이용하게 되는 정보) -->
<xs:attribute name="dst_ip_addr" type="xs:string"/>
<!-- 사물과의 연동시 사물의 TCP Port 값 -->
<xs:attribute name="src_tcp_port" type="xs:string"/>
<!-- 사물과의 연동시 사물이 전달하는 메시지의 destination TCP Port 값 (메시지를 Network Adaptor
를 통해 후킹하는 경우 이용하게 되는 정보) -->
<xs:attribute name="dst_tcp_port" type="xs:string"/>
<!-- 사물과의 연동시 사물의 UDP Port 값 -->
<xs:attribute name="src_udp_port" type="xs:string"/>
<!-- 사물과의 연동시 사물이 전달하는 메시지의 destination UDP Port 값 (메시지를 Network Adaptor
를 통해 후킹하는 경우 이용하게 되는 정보) -->
<xs:attribute name="dst_udp_port" type="xs:string"/>
<!-- 사물과의 연동시 HTTP 메시지 통신 방식을 사용하는 경우 HTTP 요청을 위한 주소값-->
<xs:attribute name="http_request_uri" type="xs:string"/>
</xs:complexType> </xs:element>

</xs:all> </xs:complexType> </xs:element>

```

[0092] 위에서 설명한 인스턴스 매니저는 가상객체화의 동작 환경을 제공하는 동작 환경으로서, 일례로, V8 기반의 자바스크립트 엔진을 이용할 수 있으며, 인스턴스 호스팅을 위한 인스턴스 풀 및 메시지 라우터를 포함할 수 있다.

[0093] 도 7은 본 발명의 일실시예에 있어서, 인스턴스 매니저를 설명하기 위한 블록도이다. 인스턴스 매니저(700)는 인스턴스의 생성 및 동작과 관련과 관련된 주요 모듈들로서, 입출력 메시지를 라우팅하는 메시지 라우터

(Message Router, 710), 인스턴스의 생애주기를 관리하고 스케줄링하는 인스턴스 라이프사이클 매니저(Instance Lifecycle Manager, 720), 인스턴스 풀링(Pooling)을 위한 인스턴스 풀(Instance Pool, 730) 및 인스턴스 동작 환경을 제공하기 위한 엔진(V8, 740)을 포함할 수 있다.

- [0094] 인터페이스 서버(750)는 인스턴스 매니저(700)와 타 모듈 및 외부 서비스간의 인터페이스 기능을 제공하며, 구조적으로는 요청을 받아 처리하기 위한 서버 부분(Server Module) 및 타 모듈 및 외부로 메시지 전달을 위한 클라이언트 부분(Client Module)으로 구성될 수 있다.
- [0095] 인터페이스 서버(750)의 서버 모듈을 통한 서버 인터페이스의 기능은 다음과 같다.
- [0096] 인스턴스 상태 질의(Instance State Query) 기능 - 인스턴스 어플리케이션 서버(Instance Application Server, IAS) 및 네트워크 어댑터(Network Adaptor, NA)에서 인스턴스 매니저(700)상에서 실행중인 인스턴스의 상태를 알고 싶을 때, HTTP(Hyper Text Transfer Protocol) 요청(Request)을 통해 특정 인스턴스의 상태 정보를 알 수 있는 기능.
- [0097] 인스턴스 상태 제어(Instance State Control) 기능 - 인스턴스 어플리케이션 서버에서 특정 인스턴스의 상태를 변경할 경우, HTTP 요청 통해 해당 인스턴스의 상태를 동작(Run)/정지(Stop)/보류(Suspend)상태로 변경할 수 있는 기능.
- [0098] 인스턴스 설치 제어(Instance Deploy Control) 기능 - 인스턴스 어플리케이션 서버를 통해 새로운 인스턴스가 설치되거나, 기존의 인스턴스를 삭제하는 경우 HTTP 요청을 통해 인스턴스 매니저(700)에 해당 인스턴스에 대한 정보를 전달할 수 있는 기능.
- [0099] 사물 피드 제어(Thing Feed Control) 기능 - 네트워크 어댑터에서 수신한 물리적 사물(Physical Thing)의 메시지를 해당 인스턴스에게 HTTP 요청의 형태로 보낼(dispatch) 수 있는 기능.
- [0100] 대표값 제어를 위한 인스턴스 상태 제어(Instance State Control for Representative Controller) 기능 - 인스턴스 어플리케이션 서버로부터 특정 인스턴스의 대표값 제어를 통해 인스턴스 상태를 변경한 경우 HTTP 요청의 형태로 메시지를 전달할 수 있는 기능.
- [0101] 인터페이스 서버(750)의 클라이언트 모듈을 통한 클라이언트 인터페이스의 기능은 특정 인스턴스가 외부 서비스로의 연결 요청을 하는 경우, 클라이언트 인터페이스를 이용하여 메시지를 전달할 수 있는 기능(특정 인스턴스가 OAuth 인증 서비스를 이용하는 경우)을 포함할 수 있다.
- [0102] 또한, 도 7에서 인스턴스 API(Application Program Interface) 핸들러(Instance API Handler)는 인스턴스와 관련하여 외부로 제공될 오픈 API를 관리할 수 있으며, 데이터베이스 커넥터(DB Connector)는 RDBMS(Relational DataBase Management System)과의 연결을 위해 이용될 수 있다.
- [0103] 도 8은 본 발명의 일실시예에 있어서, 인스턴스 상태 질의 기능을 설명하기 위한 흐름도이다. 인스턴스 어플리케이션 서버(Instance Application Server, IAS)는 개별 인스턴스에 대한 상태 정보 제공을 위해, 인스턴스 상태 질의 기능을 통해 인스턴스 매니저(700)로 질의할 수 있다. 실행중인 인스턴스 매니저(700)상에서의 개별 인스턴스의 상태 관리는 인스턴스 라이프사이클 매니저(720)에서 담당할 수 있으며, 인스턴스 라이프사이클 매니저(720)는 인스턴스 식별자(ID) 값을 이용하여 해당 인스턴스의 상태 정보를 반환할 수 있다. 인스턴스의 상태정보는 동작(Run), 보류(Suspend), 정지(Stop) 및 에러(Error)와 같은 상태 값을 가질 수 있다.
- [0104] 단계(810)에서 인터페이스 서버(750)는 서버 모듈을 통해 HTTP 요청 메시지를 수신할 수 있다. 일례로, 인터페이스 서버(750)는 인스턴스 어플리케이션 서버(Instance Application Server, IAS)나 네트워크 어댑터(Network Adaptor, NA) 또는 외부 웹 서비스(External Web Services, EWS)로부터 HTTP 요청 메시지를 수신할 수 있다.
- [0105] 단계(820)에서 인터페이스 서버(750)는 인스턴스 상태를 얻기 위한 메시지를 생성하여 메시지 라우터(710)로 전달할 수 있다.
- [0106] 단계(830)에서 메시지 라우터(710)는 인스턴스 상태를 얻기 위한 메시지를 파싱하여 인스턴스 라이프사이클 매니저(720)로 전달할 수 있다.
- [0107] 단계(840)에서 인스턴스 라이프사이클 매니저(720)는 등록된 인스턴스 상태 정보를 조회하여, 인스턴스 식별자의 조회가 성공한 경우 해당 인스턴스에 대한 상태 메시지를 생성할 수 있고, 인스턴스 식별자의 조회가 실패한 경우 인스턴스 정보 없음 메시지를 생성할 수 있다. 일례로, 상태 메시지는, 동작(Run), 보류(Suspend), 정지(Stop) 및 에러(Error)와 같은 상태 값을 포함할 수 있다.

- [0108] 단계(850)에서 인터페이스 서버(750)는 서버 모듈을 통해 HTTP요청 메시지에 대한 응답을 제공할 수 있다. 일례로, 인터페이스 서버(750)는 단계(840)에서 생성되는 상태 메시지 또는 정보 없음 메시지를 서버 모듈을 통해 HTTP요청 메시지에 대한 응답으로 제공할 수 있다.
- [0109] 도 9는 본 발명의 일실시예에 있어서, 인스턴스 상태 제어 기능을 설명하기 위한 흐름도이다. 인스턴스 어플리케이션 서버는 개별 인스턴스에 대한 상태를 변경하고자 하는 경우, 인스턴스 상태 제어 기능을 이용할 수 있다. 상태를 변경하고자 하는 인스턴스의 식별자 정보를 이용하여 메시지가 전달되면, 인스턴스 라이프사이클 매니저(720)는 해당 인스턴스의 상태를 변경할 수 있다. 이때 변경 가능한 상태는 동작(Run), 보류(Suspend) 및 정지(Stop) 상태를 포함할 수 있다.
- [0110] 단계(910)에서 인터페이스 서버(750)는 서버 모듈을 통해 HTTP 요청 메시지를 수신할 수 있다. 일례로, 인터페이스 서버(750)는 인스턴스 어플리케이션 서버나 네트워크 어댑터 또는 외부 웹 서비스로부터 HTTP 요청 메시지를 수신할 수 있다.
- [0111] 단계(920)에서 인터페이스 서버(750)는 인스턴스 상태 설정을 위한 메시지를 생성하여 메시지 라우터(710)로 전달할 수 있다.
- [0112] 단계(930)에서 메시지 라우터(710)는 인스턴스 상태 설정을 위한 메시지를 파싱하여 인스턴스 라이프사이클 매니저(720)로 전달할 수 있다.
- [0113] 단계(940)에서 인스턴스 라이프사이클 매니저(720)는 등록된 인스턴스 상태 정보를 조회하여, 인스턴스 식별자의 조회가 성공한 경우 해당 인스턴스에 대한 상태 정보를 변경할 수 있고, 인스턴스 식별자의 조회가 실패한 경우 인스턴스 정보 없음 메시지를 생성할 수 있다. 예를 들어, 인스턴스 라이프사이클 매니저(720)는 인스턴스 식별자의 조회가 성공한 경우, 해당 인스턴스의 상태를 HTTP 요청 메시지에 따라 동작(Run), 보류(Suspend) 및 정지(Stop) 중 하나의 상태로 변경할 수 있다.
- [0114] 단계(950)에서 인터페이스 서버(750)는 서버 모듈을 통해 HTTP요청 메시지에 대한 응답을 제공할 수 있다. 일례로, 인터페이스 서버(750)는 단계(940)에서 상태 정보의 변경에 따라 생성되는 응답 메시지 또는 정보 없음 메시지를 서버 모듈을 통해 HTTP요청 메시지에 대한 응답으로 제공할 수 있다.
- [0115] 도 10은 본 발명의 일실시예에 있어서, 인스턴스 설치 제어 기능을 설명하기 위한 흐름도이다. 인스턴스 어플리케이션 서버는 새로운 인스턴스를 설치(Deploy)하거나 기존의 인스턴스를 삭제(Undeploy)하고자 하는 경우, 인스턴스 설치 제어 기능을 이용할 수 있다. 상태를 변경하고자 하는 인스턴스 식별자 정보를 이용하여 메시지가 전달되면, 인스턴스 라이프사이클 매니저(720)는 해당 인스턴스의 상태를 변경할 수 있다. 이때, 삭제된 인스턴스는 인스턴스 풀(730)에서 제외될 수 있다.
- [0116] 단계(1010)에서 인터페이스 서버(750)는 서버 모듈을 통해 HTTP 요청 메시지를 수신할 수 있다. 일례로, 인터페이스 서버(750)는 인스턴스 어플리케이션 서버나 네트워크 어댑터 또는 외부 웹 서비스로부터 HTTP 요청 메시지를 수신할 수 있다.
- [0117] 단계(1020)에서 인터페이스 서버(750)는 인스턴스 설치 메시지 또는 인스턴스 삭제 메시지를 생성하여 메시지 라우터(710)로 전달할 수 있다.
- [0118] 단계(1030)에서 메시지 라우터(710)는 인스턴스 설치 메시지 또는 인스턴스 삭제 메시지를 파싱하여 인스턴스 라이프사이클 매니저(720)로 전달할 수 있다.
- [0119] 단계(1040)에서 인스턴스 라이프사이클 매니저(720)는 전달된 메시지가 인스턴스 설치 메시지인 경우 신규 인스턴스의 등록 및 신규 인스턴스의 식별자의 생성을 요청할 수 있고, 전달된 메시지가 인스턴스 삭제 메시지인 경우 등록된 인스턴스를 삭제할 수 있다. 이때, 신규 인스턴스의 등록 및 식별자의 생성에 대한 요청은 인스턴스 풀(730)로 전달될 수 있고, 인스턴스 풀(730)에 신규 인스턴스가 생성되는 경우, 생성된 인스턴스의 식별자가 인스턴스 라이프사이클 매니저(720)로 반환될 수 있다. 또한, 인스턴스 라이프사이클 매니저(720)는 해당 인스턴스의 삭제를 위해, 등록된 인스턴스 상태 정보를 조회하여 인스턴스 식별자의 조회가 성공한 경우, 해당 인스턴스의 상태를 종료(Stop) 상태로 변경한 후, 등록되어 있는 인스턴스에 대한 정보를 삭제할 수 있다.
- [0120] 단계(1050)에서 인터페이스 서버(750)는 서버 모듈을 통해 HTTP요청 메시지에 대한 응답을 제공할 수 있다. 예를 들어, 인스턴스 설치 메시지에 따라 신규 인스턴스가 생성된 경우, 인터페이스 서버(750)는 생성된 신규 인스턴스의 식별자를 포함하는 메시지를 제공할 수 있다. 다른 예로, 인스턴스 삭제 메시지에 따라 인스턴스가 삭제된 경우, 인터페이스 서버(750)는 해당 인스턴스가 삭제되었음을 알리기 위한 메시지를 제공할 수 있다.

또 다른 예로, 삭제하고자 하는 인스턴스가 인스턴스 라이프사이클 매니저(720)에 등록되어 있지 않은 경우, 인터페이스 서버(750)는 인스턴스 정보 없음 메시지를 제공할 수 있다.

- [0121] 도 11은 본 발명의 일실시예에 있어서, 사물 피드 제어 기능을 설명하기 위한 블록도이다. 사물 피드 제어 기능은 네트워크 어댑터로부터 수신한 사물의 메시지를 특정 인스턴스에게 보내기(Dispatch) 위한 기능을 제공할 수 있다. 인터페이스 서버(750)는 서버 모듈을 이용하여 네트워크 어댑터로부터 메시지 수신하는 경우, 메시지 라우터(710)에게 메시지를 HTTP 요청 방식을 이용하여 전달할 수 있고, 네트워크 어댑터에게는 응답 메시지를 전달할 수 있다. 메시지 라우터(710)는 별개의 스레드(Thread)를 이용하여 인터페이스 서버(750)의 서버 모듈로부터 수신한 메시지를 해당 인스턴스 피드상에 추가할 수 있고, 트리거(Trigger)를 이용하여 구독(Subscribe) 중인 인스턴스들이 호출될 수 있도록 인스턴스 라이프사이클 매니저(720)상의 인스턴스 스케줄러에 스케줄을 등록할 수 있다.
- [0122] 단계(1110)에서 네트워크 어댑터는 메시지 리스닝 테이블(Message Listening Table) 정보를 이용하여 해당 메시지의 송신 장치(사물) 정보를 인식할 수 있다. 예를 들어, 네트워크 어댑터는 메시지를 전송하는 사물에 대한 정보를 메시지 리스닝 테이블을 통해 인식함으로써, 필요한 인스턴스 식별자를 획득할 수 있다.
- [0123] 단계(1120)에서 네트워크 어댑터는 인터페이스 서버(750)의 서버 모듈로 HTTP 요청 메시지를 전송할 수 있다.
- [0124] 단계(1130)에서 인터페이스 서버(750)는 서버 모듈을 통해 HTTP 요청 메시지를 수신하여 메시지 라우터(710)로 전달할 수 있다.
- [0125] 단계(1140)에서 메시지 라우터(710)는 인스턴스 풀(730)에 해당 인스턴스가 존재하는지 여부를 확인하고, 해당 인스턴스의 서브스크라이버 인터페이스의 메시지 큐에 HTTP 요청 메시지와 연관된 데이터를 저장할 수 있다.
- [0126] 단계(1150)에서 메시지 라우터(710)는 인터페이스 서버(750)의 서버 모듈을 통해 HTTP 요청 메시지에 대한 응답을 제공할 수 있다.
- [0127] 도 12는 본 발명의 일실시예에 있어서, 스케줄 등록 방법을 설명하기 위한 흐름도이다. 본 실시예에 따른 스케줄 등록 방법에 포함된 단계들은 도 11에서 설명한 단계(1140) 및 단계(1150) 사이에 포함될 수 있다.
- [0128] 단계(1210)에서 메시지 라우터(710)는 인스턴스 라이프사이클 매니저(720)로 특정 인스턴스에 대한 메시지 피드 정보 업데이트를 알릴 수 있다.
- [0129] 단계(1220)에서 인스턴스 라이프사이클 매니저(720)는 해당 인스턴스의 피드에 대해 구독 중인 인스턴스들의 트리거 여부를 판단하여, 트리거가 존재하는 인스턴스를 위한 스케줄을 등록할 수 있다. 일례로, 등록되는 스케줄은 인스턴스의 즉시 동작을 위한 스케줄을 포함할 수 있다.
- [0130] 도 11의 단계(1150)에서 메시지 라우터(710)는 도 12에서 설명한 트리거가 존재하는 인스턴스를 위한 스케줄이 등록된 경우, 네트워크 어댑터로부터의 HTTP 요청 메시지에 따라 장치(사물)로부터의 메시지가 인스턴스에 반영되었음을 알리는 메시지를 응답으로 제공할 수 있다. 반대로, 트리거가 존재하는 인스턴스를 위한 스케줄이 등록되지 않은 경우, 메시지 라우터(710)는 해당 인스턴스가 존재하지 하지 않음을 나타내는 메시지를 응답으로 제공할 수 있다.
- [0131] 도 13 내지 도 15는 본 발명의 일실시예에 있어서, 인스턴스 위자드를 설명하기 위한 도면들이다.
- [0132] 제1 화면(1300)은 웹을 통해 제공되는 인스턴스 위자드를 이용하여 인스턴스 정보를 입력하는 제1 단계를 나타내고 있다. 제1 단계에서는 인스턴스의 이름, 태그, IoT 노드의 프로파일이 저장된 URL 등에 대한 정보들이 입력될 수 있다.
- [0133] 제2 화면(1400)은 웹을 통해 제공되는 인스턴스 위자드를 이용하여 인스턴스 정보를 입력하는 제2 단계를 나타내고 있다. 제2 단계에서는 인스턴스 아이콘을 선택하기 위한 사용자 인터페이스, 인스턴스의 상태를 선택하기 위한 사용자 인터페이스, 컨트롤러를 선택하기 위한 사용자 인터페이스, 대표값의 단위를 선택하기 위한 사용자 인터페이스 등이 제공될 수 있고, 사용자는 제공된 사용자 인터페이스들을 이용하여 인스턴스에 대한 다양한 정보들을 입력할 수 있다.
- [0134] 제3 화면(1500)은 웹을 통해 제공되는 인스턴스 위자드를 이용하여 인스턴스 정보를 입력하는 제3 단계를 나타내고 있다. 제3 단계에서는 사용자와 관련된 외부 서비스를 통해 조건에 따른 정보를 제공받기 위한 정보가 설정될 수 있다.

- [0135] 예를 들어, 제1 화면(1300)을 통해 설정(프로파일의 URL에 기초하여) IoT 노드에서 물의 양에 대한 정보가 감지될 수 있고, IoT 노드에서 네트워크 어댑터를 통해 인스턴스 매니저로 물의 양에 대한 정보가 포함된 메시지를 전송할 수 있다. 이때, 도 11 및 도 12를 통해 설명한 바와 같이, 메시지가 처리되어 인스턴스의 동작이 스케줄링될 수 있다. 이때, 제3 화면(1500)에 나타난 바와 같이, 물의 양이 10% 미만이면, 외부 서비스 A를 통해 알림이 제공될 수 있다. 이 경우, 제2 화면(1400)에서는 10% 미만이라는 조건을 위해 비율에 대한 데이터 타입이 설정될 수 있다.
- [0136] 도 16은 본 발명의 일실시예에 있어서, 매시업 인스턴스를 위한 추가 설정 및 외부 서비스 인증 요청을 설명하기 위한 도면이다. 매시업 인스턴스는, 인스턴스간 연동을 처리한다. 따라서, 화면(1600)에서는 매시업 인스턴스를 위해 서브스크라이버 인터페이스를 위한 피더 인터페이스를 대표값(센서측)에 따라 선택하는 인터페이스와 컨트롤러 인터페이스를 위한 리시버 인터페이스를 대표값(제어 기기측)에 따라 선택하는 인터페이스를 각각 나타내고 있다. 또한, 매시업 인스턴스는 인스턴스와 외부 서비스간 연동을 처리한다. 따라서, 화면(1600)에서는 외부 서비스로의 인증 요청(0Auth request)을 위한 인터페이스를 나타내고 있다.
- [0137] 도 17은 본 발명의 일실시예에 있어서, 오픈 API를 설명하기 위한 도면이다. 화면(1700)은 인스턴스의 피더 인터페이스를 통해 노출되는 값들에 대한 오픈 API를 설정하기 위한 인터페이스를 나타내고 있다. 예를 들어, 화면(1700)에서는 'Livingroom'과 'Hee's room' 각각의 인스턴스에 대해, 오픈 API를 설정하기 위한 인터페이스를 나타내고 있다. 이미 설명한 바와 같이 피더 인터페이스는, 장치(사물)로부터 받은 메시지를 판독 후 타 서비스(외부 서비스)로 전달할 수 있다. 이때, 인스턴스의 처리와 관련된 기능을 API의 형태로 외부 서비스에 제공할 수 있다. 이때, 외부 서비스로 제공될 오픈 API를 위한 설명 문서가 제공될 수 있으며, 설명 문서에 대한 URL이 다음 표 11의 일례와 같이 제공될 수 있다.

표 11

[0138]

<code>http://\$userID\$.instancehosting.com/instanceId/\$instanceID\$/description.html</code> * 변수값들의 경우 '\$' 로 표기해 구분

- [0139] 화면(1700)에서 'Key/URL'은 상술한 설명 문서에 대한 URL을 입력하기 위한 인터페이스를 제공할 수 있다.
- [0140] 도 18은 본 발명의 일실시예에 있어서, 서비스 방법을 도시한 흐름도이다. 본 실시예에 따른 서비스 방법은 적어도 하나의 프로그램이 저장된 적어도 하나의 스토리지 유닛 및 적어도 하나의 프로세서를 포함하는 서비스 시스템에 의해 수행될 수 있다. 이때, 서비스 방법이 포함하는 단계들은 적어도 하나의 프로그램의 제어에 따라 적어도 하나의 프로세서에 의해 처리될 수 있다.
- [0141] 단계(1810)에서 서비스 시스템은 사용자에게 제공되는 웹페이지의 단계별 설정 기능을 통해 입력된, 사물(Thing)에 대한 가상화 오브젝트(Virtualization Object)의 인스턴스에 대한 정보를 확인할 수 있다. 일례로, 단계별 설정 기능은, 인스턴스의 이름을 입력받기 위한 사용자 인터페이스, 상기 인스턴스의 태그를 입력받기 위한 사용자 인터페이스, IoT 노드의 프로파일이 저장된 URL을 입력받기 위한 사용자 인터페이스, 인스턴스 아이콘을 선택하기 위한 사용자 인터페이스, 인스턴스의 상태를 선택하기 위한 사용자 인터페이스, 컨트롤러를 선택하기 위한 사용자 인터페이스 및 대표값의 단위를 선택하기 위한 사용자 인터페이스 중 복수의 사용자 인터페이스들을 단계적으로 상기 사용자에게 노출할 수 있다.
- [0142] 단계(1820)에서 서비스 시스템은 웹페이지의 외부 서비스 연결 기능을 통해 입력된 외부 서비스에 대한 정보를 이용하여 외부 서비스와의 인증을 수행할 수 있다. 일례로, 외부 서비스 연결 기능은, 외부 서비스에 대한 사용자의 계정 정보를 입력받기 위한 사용자 인터페이스를 포함할 수 있다.
- [0143] 단계(1830)에서 서비스 시스템은 웹페이지의 오픈 API(Application Program Interface) 설정 기능을 통해 입력된 정보를 이용하여 외부 서비스로 오픈 API를 제공할 수 있다. 일례로, 오픈 API 설정 기능은, 오픈 API의 설명 문서가 포함된 위치 정보를 입력 받기 위한 사용자 인터페이스를 포함할 수 있다.
- [0144] 다른 실시예에 따른 서비스 방법은 도 18을 통해 설명한 단계들(1810 내지 1830) 이외에 웹페이지를 통해 인스턴스별 아이콘을 표시하는 단계(미도시)를 더 포함할 수 있다. 이때, 아이콘이 표시되는 영역을 통해, 인스턴스에 대한 컨트롤 기능 및 인스턴스의 대표값이 표시될 수 있다. 예를 들어, 온도를 출력하는 인스턴스의 경우 대표값으로 '23° C'와 같은 정보가 표시될 수 있다. 인스턴스에 대한 컨트롤 기능은 온/오프 설정 기능 또는 락/언락(lock/unlock) 설정 기능 등을 포함할 수 있다. 또한, 필요에 따라 선택적으로 상기 영역을 통해 인스

턴스의 상태가 더 표시될 수 있다. 예를 들어, 상태바의 색에 따라 인스턴스의 상태가 표시될 수 있다.

[0145] 다른 실시예에 따른 서비스 방법은, 사용자의 요청을 수신하는 단계(미도시) 및 사용자의 요청에 따라 웹페이지를 제공하는 단계(미도시)를 포함할 수 있다. 이때, 웹페이지는 사물(Thing)에 대한 가상화 오브젝트(Virtualization Object)의 인스턴스에 대한 정보를 입력받기 위한 단계별 설정 기능, 외부 서비스와의 인증을 위한 정보를 입력받기 위한 외부 서비스 연결 기능 및 상기 외부 서비스로 제공하기 위한 오픈 API에 대한 정보를 입력받기 위한 오픈 API 설정 기능을 포함할 수 있다. 이러한 다른 실시예에 따른 서비스 방법은 적어도 하나의 프로그램이 저장된 적어도 하나의 스토리지 유닛 및 적어도 하나의 프로세서를 포함하는 서비스 시스템에 의해 수행될 수 있다. 이때, 서비스 방법이 포함하는 단계들은 적어도 하나의 프로그램의 제어에 따라 적어도 하나의 프로세서에 의해 처리될 수 있다.

[0146] 이와 같이, 본 발명의 실시예들에 따르면, IoT(Internet of Things), WoT(Web of Things) 또는 WoO(Web of Objects) 환경에서 이기종의 기기들이나 다양한 서비스들을 연동 또는 융합하여 관리하고, 통합적으로 서비스를 제공할 수 있다.

[0147] 이상에서 설명된 장치는 하드웨어 구성요소, 소프트웨어 구성요소, 및/또는 하드웨어 구성요소 및 소프트웨어 구성요소의 조합으로 구현될 수 있다. 예를 들어, 실시예들에서 설명된 장치 및 구성요소는, 예를 들어, 프로세서, 콘트롤러, ALU(arithmetic logic unit), 디지털 신호 프로세서(digital signal processor), 마이크로컴퓨터, FPA(field programmable array), PLU(programmable logic unit), 마이크로프로세서, 또는 명령(instruction)을 실행하고 응답할 수 있는 다른 어떠한 장치와 같이, 하나 이상의 범용 컴퓨터 또는 특수 목적 컴퓨터를 이용하여 구현될 수 있다. 처리 장치는 운영 체제(OS) 및 상기 운영 체제 상에서 수행되는 하나 이상의 소프트웨어 애플리케이션을 수행할 수 있다. 또한, 처리 장치는 소프트웨어의 실행에 응답하여, 데이터를 접근, 저장, 조작, 처리 및 생성할 수도 있다. 이해의 편의를 위하여, 처리 장치는 하나가 사용되는 것으로 설명된 경우도 있지만, 해당 기술분야에서 통상의 지식을 가진 자는, 처리 장치가 복수 개의 처리 요소(processing element) 및/또는 복수 유형의 처리 요소를 포함할 수 있음을 알 수 있다. 예를 들어, 처리 장치는 복수 개의 프로세서 또는 하나의 프로세서 및 하나의 콘트롤러를 포함할 수 있다. 또한, 병렬 프로세서(parallel processor)와 같은, 다른 처리 구성(processing configuration)도 가능하다.

[0148] 소프트웨어는 컴퓨터 프로그램(computer program), 코드(code), 명령(instruction), 또는 이들 중 하나 이상의 조합을 포함할 수 있으며, 원하는 대로 동작하도록 처리 장치를 구성하거나 독립적으로 또는 결합적으로(collectively) 처리 장치를 명령할 수 있다. 소프트웨어 및/또는 데이터는, 처리 장치에 의하여 해석되거나 처리 장치에 명령 또는 데이터를 제공하기 위하여, 어떤 유형의 기계, 구성요소(component), 물리적 장치, 가상 장치(virtual equipment), 컴퓨터 저장 매체 또는 장치, 또는 전송되는 신호 파(signal wave)에 영구적으로, 또는 일시적으로 구체화(embody)될 수 있다. 소프트웨어는 네트워크로 연결된 컴퓨터 시스템 상에 분산되어서, 분산된 방법으로 저장되거나 실행될 수도 있다. 소프트웨어 및 데이터는 하나 이상의 컴퓨터 판독 가능 기록 매체에 저장될 수 있다.

[0149] 실시예에 따른 방법은 다양한 컴퓨터 수단을 통하여 수행될 수 있는 프로그램 명령 형태로 구현되어 컴퓨터 판독 가능 매체에 기록될 수 있다. 상기 컴퓨터 판독 가능 매체는 프로그램 명령, 데이터 파일, 데이터 구조 등을 단독으로 또는 조합하여 포함할 수 있다. 상기 매체에 기록되는 프로그램 명령은 실시예를 위하여 특별히 설계되고 구성된 것들이거나 컴퓨터 소프트웨어 당업자에게 공지되어 사용 가능한 것일 수도 있다. 컴퓨터 판독 가능 기록 매체의 예에는 하드 디스크, 플로피 디스크 및 자기 테이프와 같은 자기 매체(magnetic media), CD-ROM, DVD와 같은 광기록 매체(optical media), 플롭티컬 디스크(floptical disk)와 같은 자기-광 매체(magneto-optical media), 및 롬(ROM), 램(RAM), 플래시 메모리 등과 같은 프로그램 명령을 저장하고 수행하도록 특별히 구성된 하드웨어 장치가 포함된다. 프로그램 명령의 예에는 컴파일러에 의해 만들어지는 것과 같은 기계어 코드뿐만 아니라 인터프리터 등을 사용해서 컴퓨터에 의해서 실행될 수 있는 고급 언어 코드를 포함한다. 상기된 하드웨어 장치는 실시예의 동작을 수행하기 위해 하나 이상의 소프트웨어 모듈로서 작동하도록 구성될 수 있으며, 그 역도 마찬가지이다.

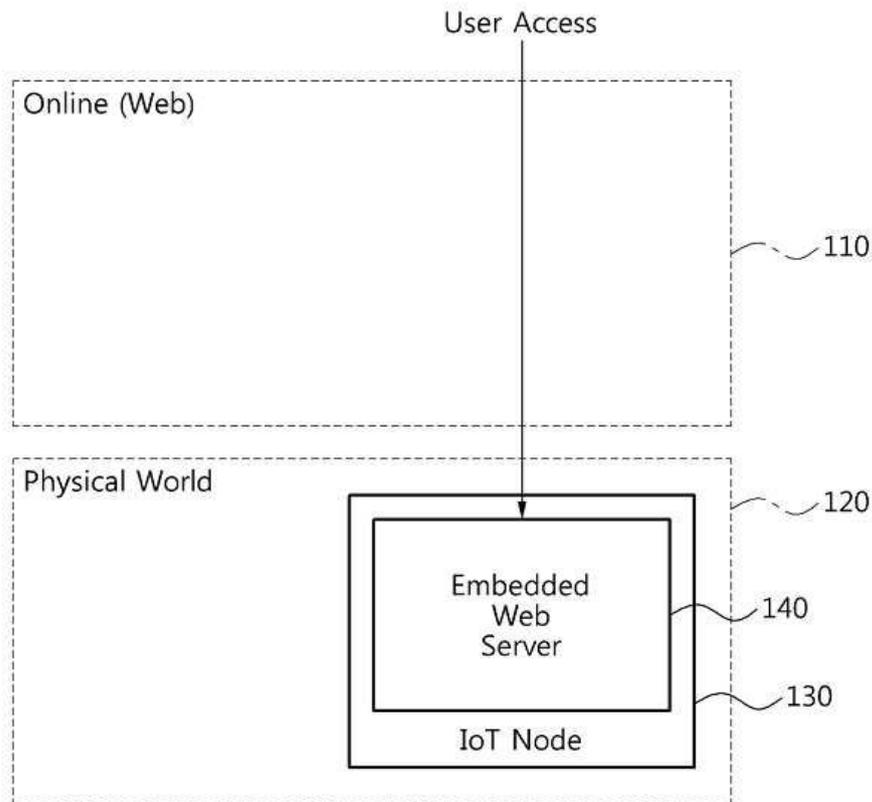
[0150] 이상과 같이 실시예들이 비록 한정된 실시예와 도면에 의해 설명되었으나, 해당 기술분야에서 통상의 지식을 가진 자라면 상기의 기재로부터 다양한 수정 및 변형이 가능하다. 예를 들어, 설명된 기술들이 설명된 방법과 다른 순서로 수행되거나, 및/또는 설명된 시스템, 구조, 장치, 회로 등의 구성요소들이 설명된 방법과 다른 형태로 결합 또는 조합되거나, 다른 구성요소 또는 균등물에 의하여 대치되거나 치환되더라도 적절한 결과가 달성될 수 있다.

[0151]

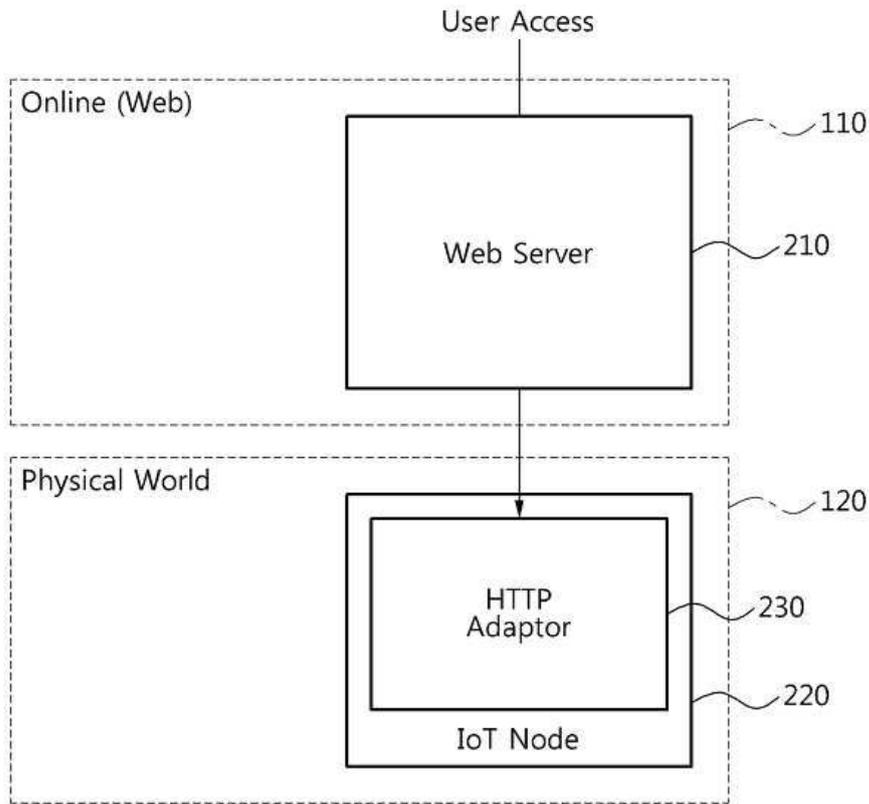
그러므로, 다른 구현들, 다른 실시예들 및 특허청구범위와 균등한 것들도 후술하는 특허청구범위의 범위에 속한다.

도면

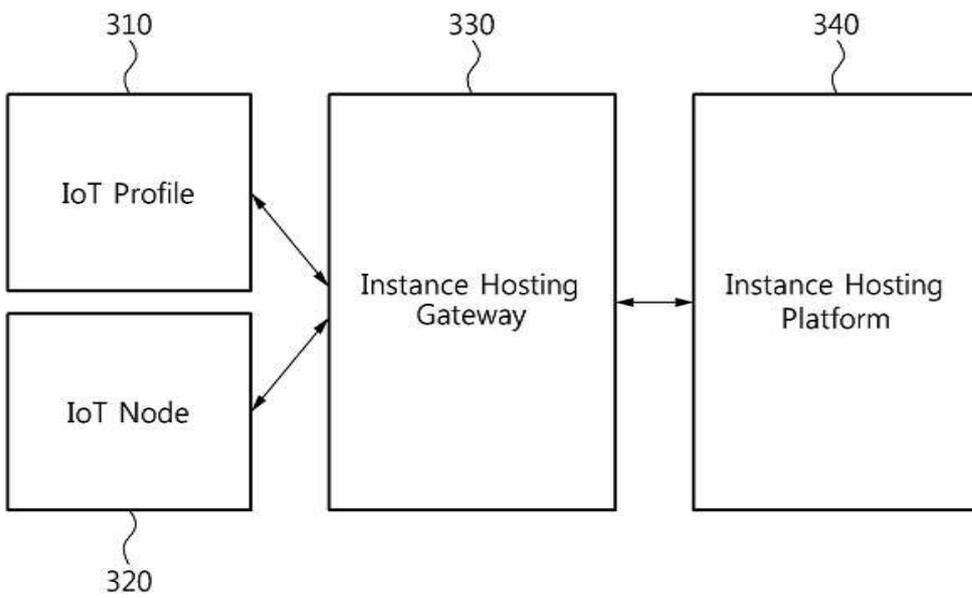
도면1



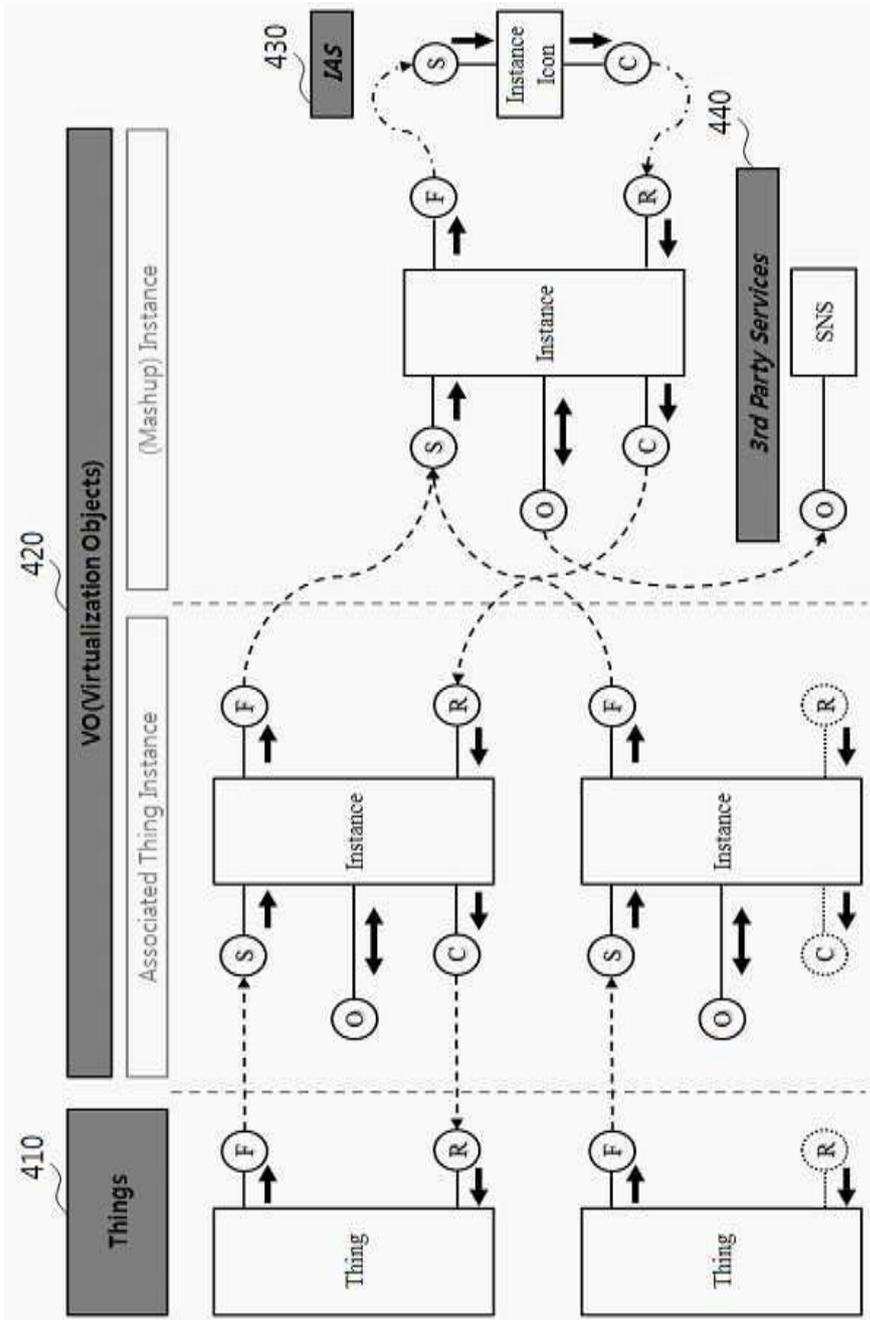
도면2



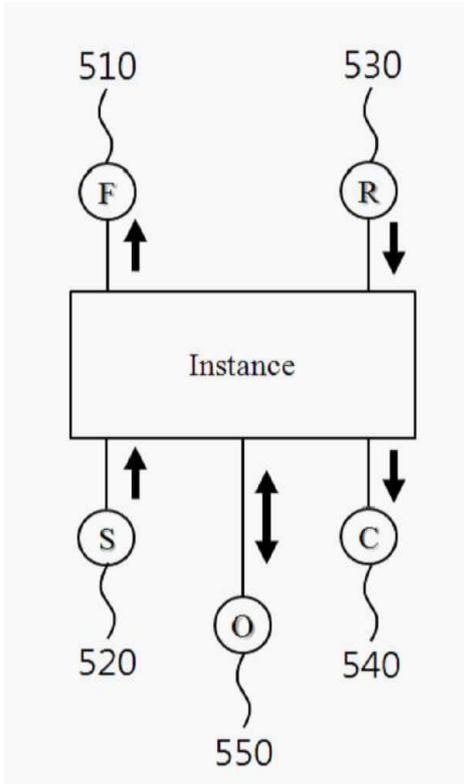
도면3



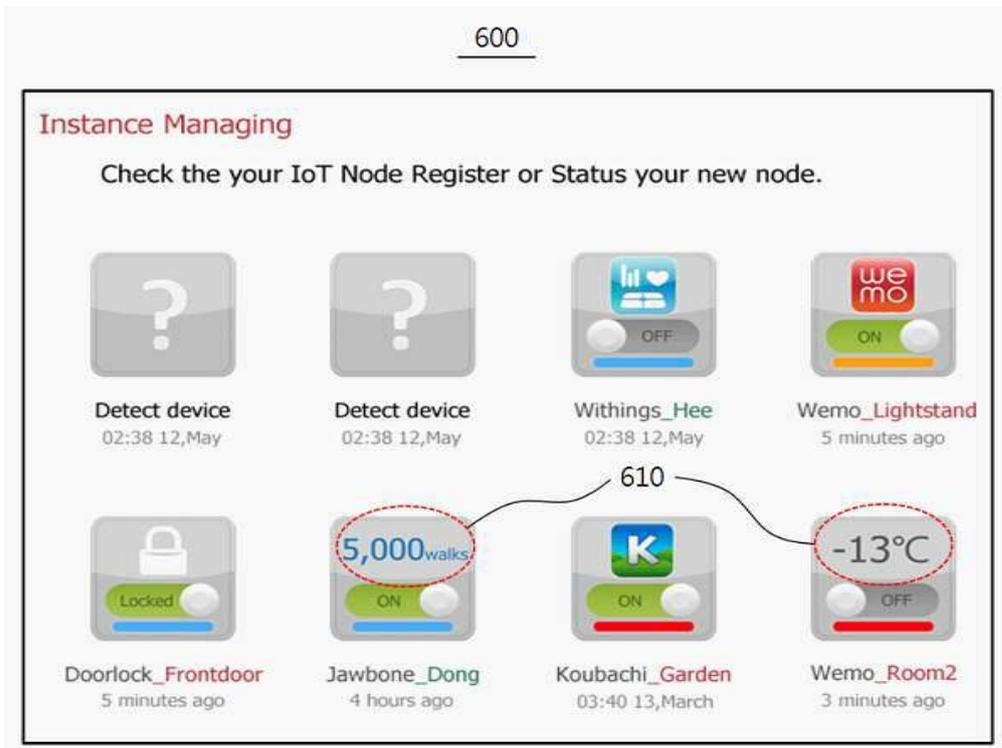
도면4



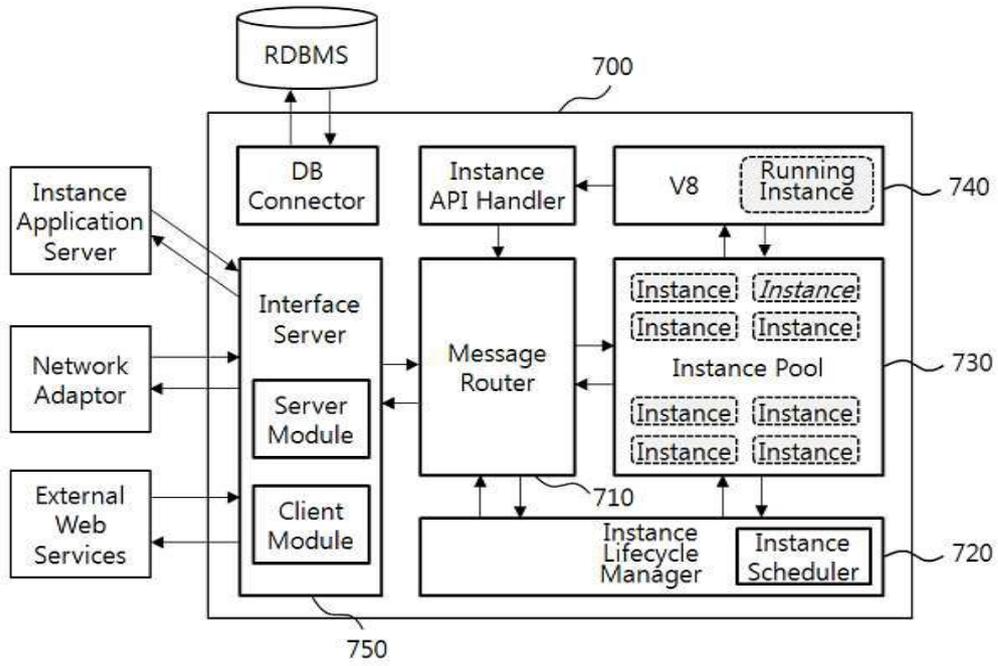
도면5



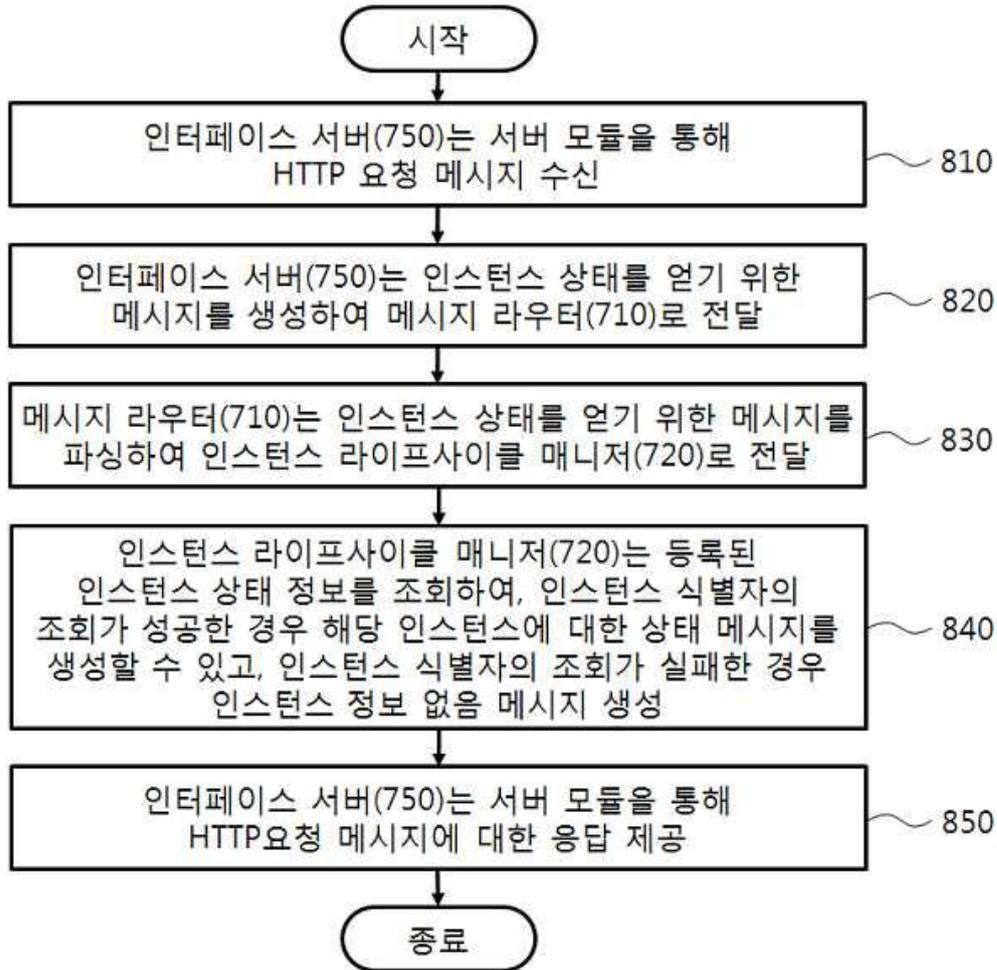
도면6



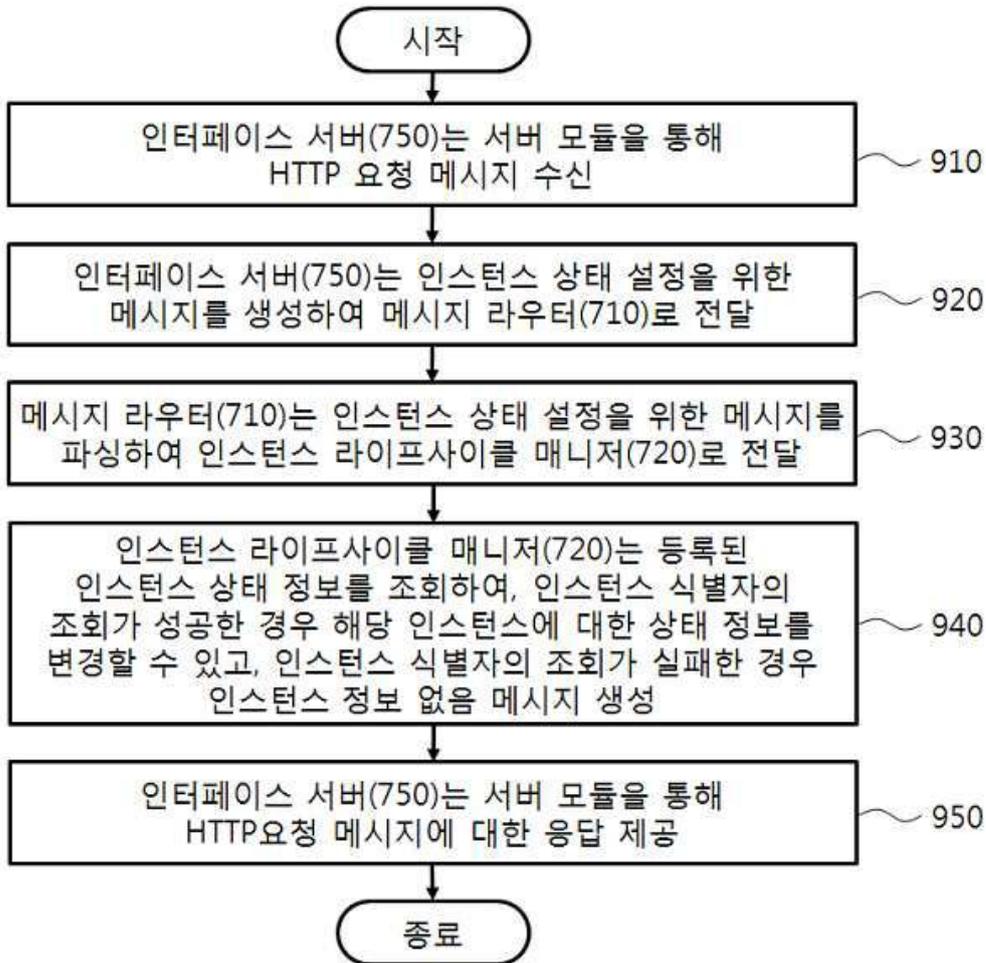
도면7



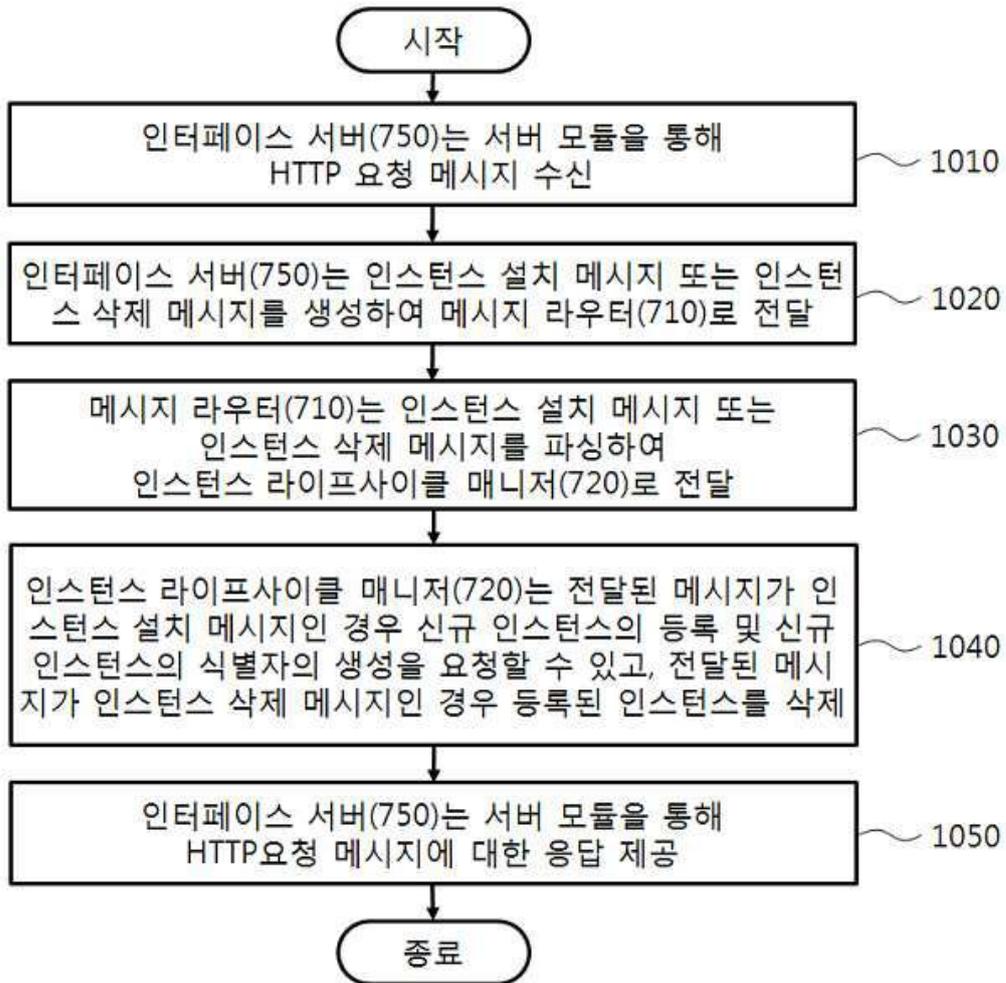
도면8



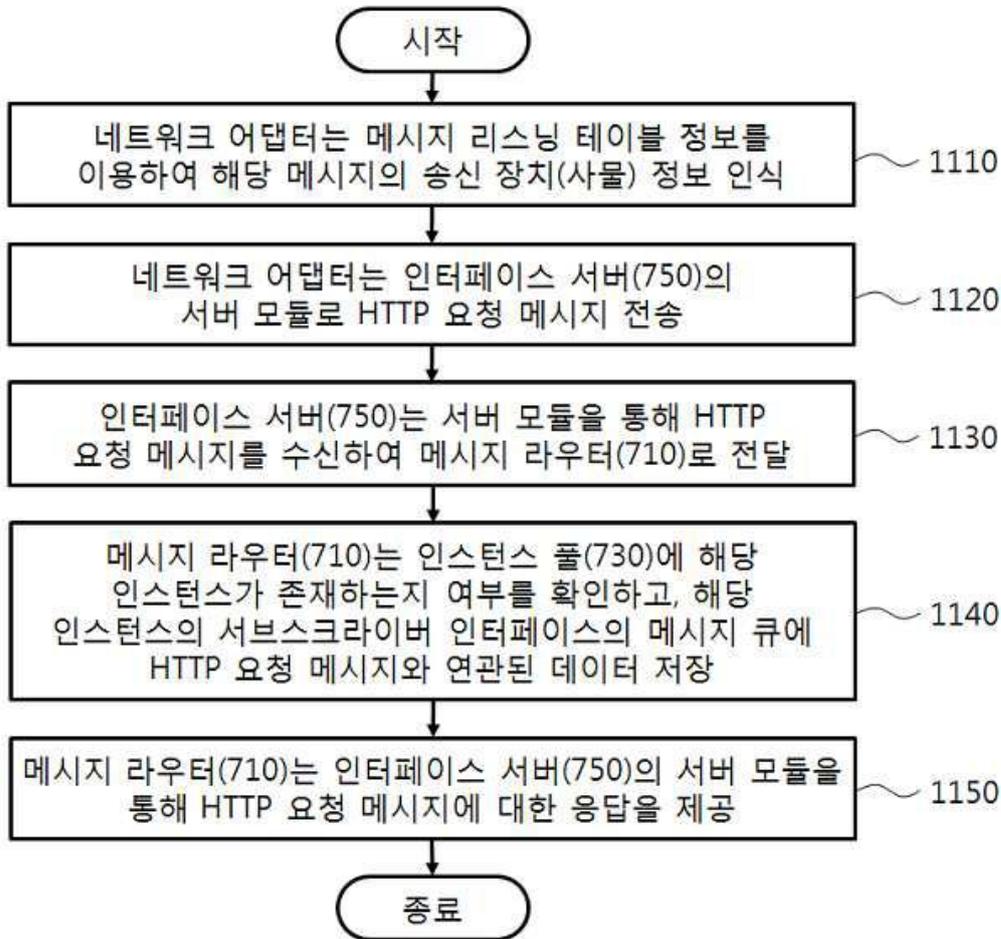
도면9



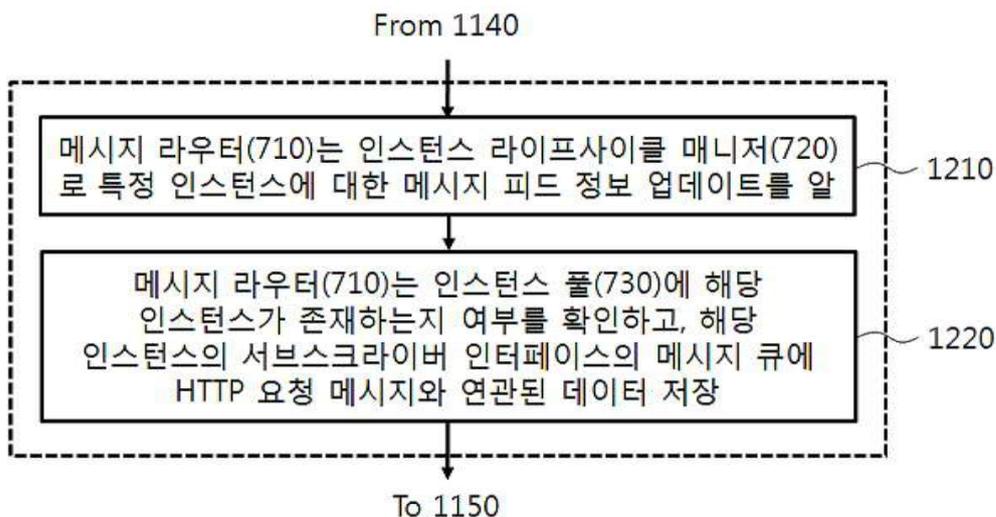
도면10



도면11



도면12



도면13

1300

Instance Wizard

Please insert your instance information.

Step1 Step2 Step3

Instance Name

Tag my tag ▼

IoT node profile URL Find

Name	Ver	Description
------	-----	-------------

Next Cancel

도면14

1400

Instance Wizard
Please insert your instance information.

Step1 **Step2** Step3

Profile name : ThingsGate

Select the icon

search

Instance status

▼

Select the controller

On / Off

Yes No

Select representative feeder

Feeder name	Unit	Representative
Temperature	℃	<input type="radio"/>
Humidity	%	<input type="radio"/>
depth of rainfall	mm	<input type="radio"/>
....		

Next
Cancel

도면15

1500

Instance Wizard

Please insert your instance information.

Step1 Step2 **Step3**

외부 서비스 A

물의 양이 10% 미만으로 떨어지면 외부 서비스 A를 통해 알려줍니다.

외부 서비스 B

외부 서비스 C

Insert

도면16

1600

Instance Wizard

Please insert your instance information.

Step1 Step2 **Step3**

Select feeder for subscriber

Temperature

Humidity

Select the receiver for control

Aircondition

OAuth request

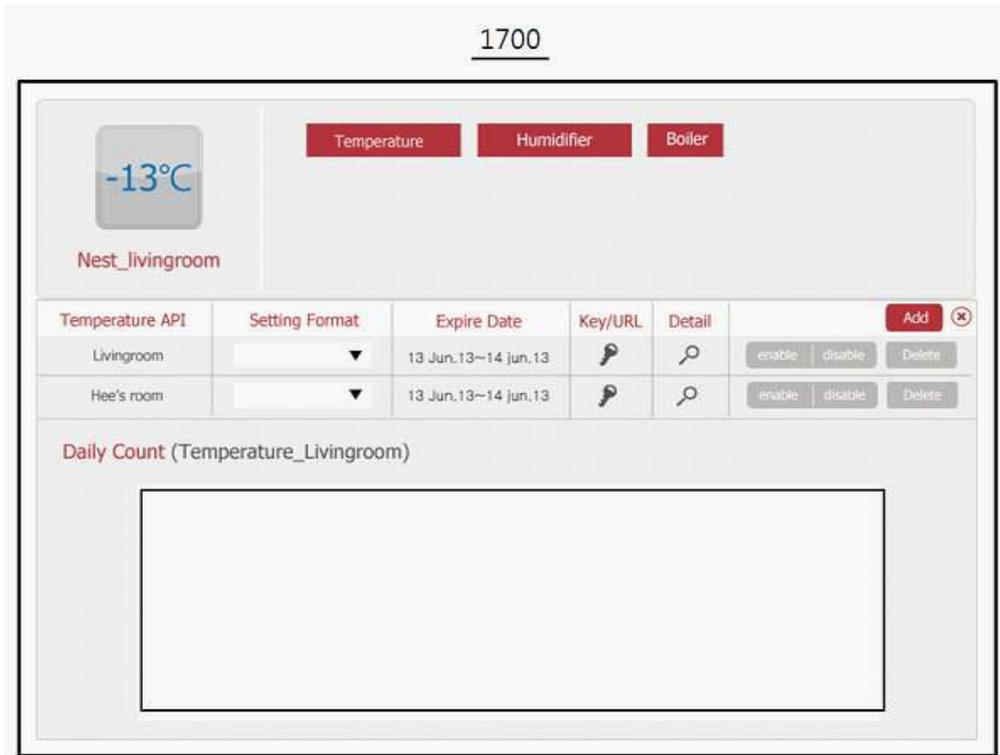
외부 서비스 A

외부 서비스 B

외부 서비스 C

Finish **Cancel**

도면17



도면18

