(19) 대한민국특허청(KR)
(12) 공개특허공보(A)

(11) 공개번호 10-2008-0005491
(43) 공개일자 2008년01월14일

(51) Int. Cl.
G06F 9/06 (2006.01) *G06F 17/40* (2006.01)
(21) 출원번호 10-2007-7022284
(22) 출원일자 2007년09월28일
심사청구일자 없음
번역문제출일자 2007년09월28일
(86) 국제출원번호 PCT/US2006/010345
국제출원일자 2006년03월22일
(87) 국제공개번호 WO 2006/115641
국제공개일자 2006년11월02일
(30) 우선권주장
11/111,882 2005년04월22일 미국(US)

(71) 출원인
마이크로소프트 코포레이션
미국 워싱턴주 (우편번호 : 98052) 레드몬드 원 마이크로소프트 웨이
(72) 발명자
슈르, 안드레이
미국 98052-6399 워싱톤주 레드몬드 원 마이크로소프트 웨이
매켄지, 브루스 에이.
미국 98052-6399 워싱톤주 레드몬드 원 마이크로소프트 웨이
(뒷면에 계속)
(74) 대리인
양영준, 백만기

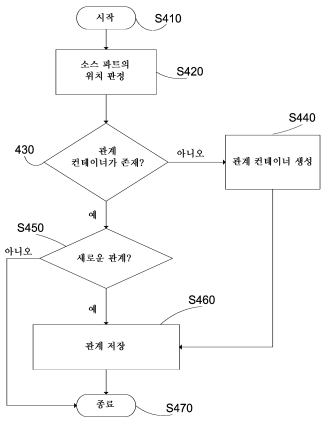
전체 청구항 수 : 총 20 항

(54) 리소스들 간의 효율적 관계 설명

(57) 요 약

소스 리소스와 관련된 관계 데이터 구조는 방법이 소스 리소스와 복수의 타겟 리소스 간의 관계를 탐색하고 설명할 수 있게 한다. 관계는 소스 리소스의 인코딩과 무관한 포맷으로 저장된다. 소스 리소스와 복수의 타겟 리소스 간의 각 관계는 중립적 콘텐츠 포맷으로 저장되고, 관계 데이터 구조는, 각 타겟 리소스의 위치, 각 타겟 리소스와 관련된 관계의 유형, 및 소스 리소스와 각 타겟 리소스 간의 각 관계를 고유하게 식별하는 식별자를 저장한다. 이에 따라, 관계 데이터 구조는 소스 리소스 또는 타겟 리소스를 디코딩하지 않고서 디코더가 소스 리소스와 복수의 타겟 리소스 간의 관계를 직접 탐색할 수 있게 한다.

대표도 - 도4



(72) 발명자

워커, 찰스 에스.

미국 98052-6399 워싱톤주 레드몬드 원 마이크로소프트 웨이

오른스테인, 데이비드 비.

미국 98052-6399 워싱톤주 레드몬드 원 마이크로소프트 웨이

두니에츠, 제리 제이.

미국 98052-6399 워싱톤주 레드몬드 원 마이크로소프트 웨이

폴락, 조슈아 엠.

미국 98052-6399 워싱톤주 레드몬드 원 마이크로소프트 웨이

세스, 사르자나 비.

미국 98052-6399 워싱톤주 레드몬드 원 마이크로소프트 웨이

니콜스, 이삭 이.

미국 98052-6399 워싱톤주 레드몬드 원 마이크로소프트 웨이

특허청구의 범위

청구항 1

소스 리소스와 하나 이상의 타겟 리소스 간의 하나 이상의 관계를 설명하는 방법 - 상기 하나 이상의 관계는 상기 소스 리소스의 인코딩과 무관한 포맷으로 인코딩됨 - 으로서,

관계 스키마를 정의하는 네임스페이스를 선택하는 단계와,

상기 하나 이상의 관계 및 상기 관계 스키마에 기초하여 관계 데이터 구조를 채우는 단계와,

상기 관계 데이터 구조에 상기 하나 이상의 관계를 저장하는 단계와,

상기 관계 데이터 구조를 상기 소스 리소스와 관련짓는 단계

를 포함하는 관계 설명 방법.

청구항 2

제1항에 있어서,

상기 하나 이상의 타겟 리소스는, 로컬형 리소스, 외부형 리소스, 상대적 리소스로 이루어지는 그룹으로부터 선택되는 리소스인 관계 설명 방법.

청구항 3

제1항에 있어서,

상기 소스 리소스 및 상기 하나 이상의 타겟 리소스는 패키지인 관계 설명 방법.

청구항 4

제1항에 있어서,

상기 관계 데이터 구조를 채우는 단계는,

상기 하나 이상의 관계를 식별하는 식별 정보를 검색하는 단계와,

상기 관계 스키마를 활용하여, 검색된 식별 정보의 유효성을 확인하는 단계

를 더 포함하는 관계 설명 방법.

청구항 5

제1항에 있어서,

상기 관계 데이터 구조를 상기 소스 리소스와 관련짓는 단계는, 상기 소스 리소스의 위치에 관련된 어드레스에 상기 관계 데이터 구조를 저장하는 단계를 더 포함하는 관계 설명 방법.

청구항 6

제5항에 있어서,

상기 소스 리소스의 위치에 관련된 어드레스는 상기 소스 리소스의 위치에 있는 서브폴더에 어드레스를 부여하는 것인 관계 설명 방법.

청구항 7

제1항의 방법을 수행하는 컴퓨터 실행가능 명령어들을 포함하는 컴퓨터 판독가능 매체.

청구항 8

프로세서, 메모리 및 동작 환경을 포함하며, 제1항의 방법을 실행하도록 동작가능한 컴퓨터 시스템.

청구항 9

소스 리소스와 하나 이상의 타겟 리소스 간의 하나 이상의 관계를 탐색하는 방법 - 상기 하나 이상의 관계는 상기 소스 리소스 또는 상기 하나 이상의 타겟 리소스의 인코딩과 무관한 포맷으로 인코딩됨 - 으로서,

상기 소스 리소스의 위치를 알아내는 단계와,

관계 데이터 구조가 상기 소스 리소스와 관련되어 있는지 여부를 판정하는 단계와,

상기 소스 리소스를 디코딩하지 않고서 상기 하나 이상의 관계를 디코딩하는 단계와,

상기 하나 이상의 관계를 표시하는 단계

를 포함하는 관계 탐색 방법.

청구항 10

제9항에 있어서,

상기 소스 리소스 또는 상기 하나 이상의 타겟 리소스는 안전한 리소스인 관계 탐색 방법.

청구항 11

제9항에 있어서,

상기 소스 리소스 및 상기 하나 이상의 타겟 리소스는 팩(Pack) URI를 이용하여 어드레스 부여되는 관계 탐색 방법.

청구항 12

제9항에 있어서,

상기 소스 리소스를 디코딩하지 않고서 상기 하나 이상의 관계를 디코딩하는 단계는,

상기 하나 이상의 관계를 디코딩하기 전에, 상기 관계 데이터 구조의 스키마를 판정하는 단계와,

상기 스키마를 활용하여 상기 관계 데이터 구조에 저장된 하나 이상의 관계를 디코딩하는 단계를 더 포함하는 관계 탐색 방법.

청구항 13

제9항에 있어서,

상기 관계 데이터 구조가 상기 소스 리소스와 관련되어 있는지 여부를 판정하는 단계는, 상기 소스 리소스의 위치에서 서브폴더의 존재를 체크하는 단계를 더 포함하는 관계 탐색 방법.

청구항 14

제13항에 있어서,

상기 서브폴더는 상기 소스 리소스의 위치에 관련된 어드레스에 위치하는 관계 탐색 방법.

청구항 15

제9항의 방법을 수행하는 컴퓨터 실행가능 명령어들을 포함하는 컴퓨터 판독가능 매체.

청구항 16

프로세서, 메모리 및 동작 환경을 포함하며, 제9항의 방법을 실행하도록 동작가능한 컴퓨터 시스템.

청구항 17

컴퓨터 판독가능 매체 상에 저장되며, 소스 리소스와 하나 이상의 타겟 리소스 간의 하나 이상의 관계에 대응하는 데이터를 제공하는 관계 데이터 구조로서,

상기 하나 이상의 관계의 각 관계를 고유하게 식별하는 식별 정보와,

상기 하나 이상의 타겟 리소스의 각 타겟 리소스의 위치를 특정하는 타겟 정보와,

상기 소스 리소스와 타겟 리소스 간의 관계를 정의하는 유형 정보

를 포함하는 관계 데이터 구조.

청구항 18

제17항에 있어서,

관계를 디코딩할 때 도움을 주기 위한 시맨틱 정보를 더 포함하는 관계 데이터 구조.

청구항 19

제17항에 있어서,

상기 관계 데이터 구조에 저장된 상기 하나 이상의 관계는 내포된(nested) 것인 관계 데이터 구조.

청구항 20

제17항에 있어서,

상기 관계 데이터 구조는 상기 소스 리소스를 설명하는 메타 데이터를 포함하는 관계 데이터 구조.

명 세 서

기 술 분 야

<1> 본 발명은 일반적으로 관계 관리에 관한 것이다. 특히, 관계 데이터 구조를 참조함으로써 리소스들 간의 관계를 탐색하는 시스템 및 방법을 제공한다.

배 경 기 술

<2> 현재, 웹 페이지, 이미지, 무비와 같은 다큐먼트는 일반적으로 특정 콘텐츠 인코딩과 특정 애플리케이션 인코딩을 포함한 인코딩의 두 가지 유형으로 인코딩되고 있다. 다큐먼트는 자신과 다른 다큐먼트 간의 관계를 정의하는 관계 정보를 포함할 수 있다. 이 관계 정보는 다큐먼트의 특정 콘텐츠 인코딩 또는 특정 애플리케이션 인코딩 유형으로 인코딩된다.

<3> 디코더는 다큐먼트를 인코딩하여 관계 정보를 추출하는 방식을 알 수 있다. 디코더는 전체 다큐먼트를 디코딩하고 관계 정보를 추출함으로써 다큐먼트 내의 관계 정보를 추출한다. 관계 정보는, 디코더가 다큐먼트의 인코딩을 이해하지 못하는 경우 추출되지 않는다.

<4> 암호화되어 있거나 디지털 서명되어 있는 안전한 다큐먼트의 관계 정보를 디코더가 보거나 갱신하려할 때 문제가 발생한다. 안전한 다큐먼트가 암호화되어 있다면, 디코더는 그 안전한 다큐먼트를 디코딩할 수 없고 다큐먼트의 관계 정보를 추출할 수 없으며 그 이유는 디코더가 안전한 다큐먼트 및 관계 정보가 인코딩되어 있는 방식을 알지 못하기 때문이다. 게다가, 안전한 다큐먼트가 디지털 서명되어 있는 경우, 디코더는 안전한 다큐먼트를 디코딩할 수 있고 관계 정보를 추출할 수 있다. 그러나, 다큐먼트를 디코딩하고 관계 정보를 추출하는 것은 디지털 서명을 무효화할 수 있으며, 그 이유는 다큐먼트를 디코딩하고 관계 정보를 추출하는 것이 다큐먼트가 마지막으로 액세스된 기일을 추적하는 다큐먼트 속성을 변경할 수 있기 때문이다.

<5> 따라서, 적어도 상술한 이유들로 인해, 다큐먼트를 디코딩하지 않고서 다큐먼트의 내부 관계 및 외부 관계를 빠르게 탐색하는 방법이 필요하다.

발명의 상세한 설명

<6> 당해 기술에서 이러한 문제점들 및 다른 문제점들은, 소스 리소스나 타겟 리소스를 디코딩하지 않고서 소스 리소스와 하나 이상의 타겟 리소스 간의 하나 이상의 관계의 탐색을 가능하게 하는 관계 데이터 구조를 제공함으로써 해결된다.

<7> 관계 데이터 구조는, 컴퓨터 판독가능 매체 상에 저장되며, 소스 리소스와 하나 이상의 타겟 리소스 간의 하나 이상의 관계를 설명한다. 관계 데이터 구조의 각 관계는, 소스 리소스와 하나 이상의 타겟 리소스 중 하나의

타겟 리소스 간의 관계를 고유하게 식별하는 식별 정보와, 그 하나의 타겟 리소스의 위치를 식별하는 타겟 정보와, 소스와 그 하나의 타겟 리소스 간의 관계의 시맨틱(semantic)을 정의하는 유형 정보를 포함한다.

<8> 또한, 관계 데이터 구조는 소스 리소스와 관련되어, 방법이 소스 리소스와 복수의 타겟 리소스 간의 복수의 관계를 설명할 수 있게 하며, 이 복수의 관계는 소스 리소스나 타겟 리소스의 인코딩과 무관한 포맷으로 인코딩된다. 복수의 관계를 설명하는 이 방법은, 관계 스키마(schema)를 정의하는 네임스페이스를 선택하는 단계와, 복수의 관계와 관계 스키마에 기초하여 관계 데이터 구조를 채우는(populate) 단계와, 관계 데이터 구조에 복수의 관계를 저장하는 단계를 포함한다.

<9> 게다가, 관계 데이터 구조는 방법이 소스 리소스와 하나 이상의 타겟 리소스 간의 하나 이상의 관계를 탐색할 수 있게 한다. 하나 이상의 관계를 탐색하기 위해, 소스 리소스의 위치를 판정한다. 후속하여, 소스 리소스의 위치를 이용하여 관계 데이터 구조가 소스 리소스와 관련되어 있는지 여부를 확정한다. 관계 데이터 구조가 소스 리소스와 관련되어 있다면, 그 하나 이상의 관계가 관계 스키마에 따라 디코딩되고, 하나 이상의 디코딩된 관계가 표시된다.

<10> 이에 따라, 관계 데이터 구조는 소스 리소스나 타겟 리소스를 디코딩하지 않고서 소스 리소스와 복수의 타겟 리소스 간의 관계에 대한 액세스를 제공한다. 관계 데이터 구조는 다른 시스템과의 보다 빠른 통신을 용이하게 하며, 그 이유는 관계가 분석하는 데 보다 빠르며 중립적 콘텐츠 포맷으로 저장될 수 있기 때문이다.

<11> 추가 이점들 및 신규한 특징들은, 다음에 따르는 설명에서 알 수 있으며, 다음에 따르는 것을 검토함으로써 당업자에게 부분적으로 자명하거나 본 발명의 실시에 의해 교시될 수 있다.

실 시 예

<18> 본 발명의 일실시예는 관계 데이터 구조를 활용하여 소스 리소스와 하나 이상의 타겟 리소스 간의 하나 이상의 관계를 탐색한다. 소스 리소스 및 타겟 리소스는 "Pack URI Scheme to Identify and Reference Parts of a Package"에 설명되어 있는 바와 같이 팩(Pack) 프로토콜을 이용하여 어드레스 지정가능한 패키지일 수 있다. 관계 데이터 구조는 소스 리소스와 관련되며 중립적 콘텐츠 포맷으로 저장된다.

<19> 도 1은 본 발명이 구현되기에 적합한 컴퓨팅 시스템 환경(100)의 일례를 도시하고 있다. 컴퓨팅 시스템 환경(100)은 적합한 컴퓨팅 환경의 일례에 불과하며, 본 발명의 용도 또는 기능성의 범위에 관해 어떤 제한을 암시하고자 하는 것이 아니다. 컴퓨팅 환경(100)이 예시적인 운영 환경(100)에 도시된 컴포넌트들 중 임의의 하나 또는 그 컴포넌트들의 임의의 조합과 관련하여 어떤 의존성 또는 요구사항을 갖는 것으로 해석되어서는 안된다.

<20> 본 발명은 많은 기타 범용 또는 특수 목적의 컴퓨팅 시스템 환경 또는 구성에서 동작할 수 있다. 본 발명에서 사용하는 데 적합할 수 있는 잘 알려진 컴퓨팅 시스템, 환경 및/또는 구성의 예로는 퍼스널 컴퓨터, 서버 컴퓨터, 핸드-헬드 또는 랩톱 장치, 멀티프로세서 시스템, 마이크로프로세서 기반 시스템, 셋톱 박스, 프로그램가능한 가전제품, 네트워크 PC, 미니컴퓨터, 메인프레임 컴퓨터, 전화 시스템, 상기 시스템들이나 장치들 중 임의의 것을 포함하는 분산 컴퓨팅 환경, 기타 등등이 있지만 이에 제한되는 것은 아니다.

<21> 본 발명은 일반적으로 컴퓨터에 의해 실행되는 프로그램 모듈과 같은 컴퓨터 실행가능 명령어와 관련하여 기술될 것이다. 일반적으로, 프로그램 모듈은 특정 태스크를 수행하거나 특정 추상 데이터 유형을 구현하는 루틴, 프로그램, 개체, 컴포넌트, 데이터 구조 등을 포함한다. 본 발명은 또한 통신 네트워크를 통해 연결되어 있는 원격 처리 장치들에 의해 태스크가 수행되는 분산 컴퓨팅 환경에서 실시되도록 설계된다. 분산 컴퓨팅 환경에서, 프로그램 모듈은 메모리 저장 장치를 비롯한 로컬 및 원격 컴퓨터 저장 매체 둘다에 위치할 수 있다.

<22> 도 1과 관련하여, 본 발명을 구현하는 예시적인 시스템은 컴퓨터(110) 형태의 범용 컴퓨팅 장치를 포함한다. 컴퓨터(110)의 컴포넌트들은 처리 장치(120), 시스템 메모리(130), 및 시스템 메모리를 비롯한 각종 시스템 컴포넌트들을 처리 장치(120)에 연결시키는 시스템 버스(121)를 포함하지만 이에 제한되는 것은 아니다. 시스템 버스(121)는 메모리 버스 또는 메모리 컨트롤러, 주변 장치 버스 및 각종 버스 아키텍처 중 임의의 것을 이용하는 로컬 버스를 비롯한 몇몇 유형의 버스 구조 중 어느 것이라도 될 수 있다. 예로서, 이러한 아키텍처는 ISA(industry standard architecture) 버스, MCA(micro channel architecture) 버스, EISA(Enhanced ISA) 버스, VESA(video electronics standard association) 로컬 버스, 그리고 메자닌 버스(mezzanine bus)로도 알려진 PCI(peripheral component interconnect) 버스 등을 포함하지만 이에 제한되는 것은 아니다.

<23> 컴퓨터(110)는 통상적으로 각종 컴퓨터 판독가능 매체를 포함한다. 컴퓨터(110)에 의해 액세스 가능한 매체는 그 어떤 것이든지 컴퓨터 판독가능 매체가 될 수 있고, 이러한 컴퓨터 판독가능 매체는 휘발성 및 비휘발성 매

체, 이동식 및 비이동식 매체를 포함한다. 예로서, 컴퓨터 판독가능 매체는 컴퓨터 저장 매체 및 통신 매체를 포함하지만 이에 제한되는 것은 아니다. 컴퓨터 저장 매체는 컴퓨터 판독가능 명령어, 데이터 구조, 프로그램 모듈 또는 기타 데이터와 같은 정보를 저장하는 임의의 방법 또는 기술로 구현되는 휘발성 및 비휘발성, 이동식 및 비이동식 매체를 포함한다. 컴퓨터 저장 매체는 RAM, ROM, EEPROM, 플래시 메모리 또는 기타 메모리 기술, CD-ROM, DVD(digital versatile disk) 또는 기타 광 디스크 저장 장치, 자기 카세트, 자기 테이프, 자기 디스크 저장 장치 또는 기타 자기 저장 장치, 또는 컴퓨터(110)에 의해 액세스되고 원하는 정보를 저장할 수 있는 임의의 기타 매체를 포함하지만 이에 제한되는 것은 아니다. 통신 매체는 통상적으로 반송파(carrier wave) 또는 기타 전송 메커니즘(transport mechanism)과 같은 피변조 데이터 신호(modulated data signal)에 컴퓨터 판독가능 명령어, 데이터 구조, 프로그램 모듈 또는 기타 데이터 등을 구현하고 모든 정보 전달 매체를 포함한다. "피변조 데이터 신호"라는 용어는, 신호 내에 정보를 인코딩하도록 그 신호의 특성들 중 하나 이상을 설정 또는 변경시킨 신호를 의미한다. 예로서, 통신 매체는 유선 네트워크 또는 직접 배선 접속(direct-wired connection)과 같은 유선 매체, 그리고 음향, RF, 적외선, 기타 무선 매체와 같은 무선 매체를 포함한다. 상술된 매체들의 모든 조합이 또한 컴퓨터 판독가능 매체의 영역 안에 포함되는 것으로 한다.

<24> 시스템 메모리(130)는 판독 전용 메모리(ROM)(131) 및 랜덤 액세스 메모리(RAM)(132)와 같은 휘발성 및/또는 비휘발성 메모리 형태의 컴퓨터 저장 매체를 포함한다. 시동 중과 같은 때에, 컴퓨터(110) 내의 구성요소들 사이의 정보 전송을 돕는 기본 루틴을 포함하는 기본 입/출력 시스템(BIOS)(133)은 통상적으로 ROM(131)에 저장되어 있다. RAM(132)은 통상적으로 처리 장치(120)가 즉시 액세스 할 수 있고 및/또는 현재 동작시키고 있는 데이터 및/또는 프로그램 모듈을 포함한다. 예로서, 도 1은 운영 체제(134), 애플리케이션 프로그램(135), 기타 프로그램 모듈(136) 및 프로그램 데이터(137)를 도시하고 있지만 이에 제한되는 것은 아니다.

<25> 컴퓨터(110)는 또한 기타 이동식/비이동식, 휘발성/비휘발성 컴퓨터 저장매체를 포함한다. 단지 예로서, 도 1은 비이동식·비휘발성 자기 매체에 기록을 하거나 그로부터 판독을 하는 하드 디스크 드라이브(140), 이동식·비휘발성 자기 디스크(152)에 기록을 하거나 그로부터 판독을 하는 자기 디스크 드라이브(151), CD-ROM 또는 기타 광 매체 등의 이동식·비휘발성 광 디스크(156)에 기록을 하거나 그로부터 판독을 하는 광 디스크 드라이브(155)를 포함한다. 예시적인 운영 환경에서 사용될 수 있는 기타 이동식/비이동식, 휘발성/비휘발성 컴퓨터 기억 매체로는 자기 테이프 카세트, 플래시 메모리 카드, DVD, 디지털 비디오 테이프, 고상(solid state) RAM, 고상 ROM 등이 있지만 이에 제한되는 것은 아니다. 하드 디스크 드라이브(141)는 통상적으로 인터페이스(140)와 같은 비이동식 메모리 인터페이스를 통해 시스템 버스(121)에 접속되고, 자기 디스크 드라이브(151) 및 광 디스크 드라이브(155)는 통상적으로 인터페이스(150)와 같은 이동식 메모리 인터페이스에 의해 시스템 버스(121)에 접속된다.

<26> 위에서 설명되고 도 1에 도시된 드라이브들 및 이들과 관련된 컴퓨터 저장 매체는, 컴퓨터(110)를 위해, 컴퓨터 판독가능 명령어, 데이터 구조, 프로그램 모듈 및 기타 데이터를 저장한다. 도 1에서, 예를 들어, 하드 디스크 드라이브(141)는 운영 체제(144), 애플리케이션 프로그램(145), 기타 프로그램 모듈(146), 및 프로그램 데이터(147)를 저장하는 것으로 도시되어 있다. 여기서 주의할 점은 이들 컴포넌트가 운영 체제(134), 애플리케이션 프로그램(135), 기타 프로그램 모듈(136), 및 프로그램 데이터(137)와 동일하거나 그와 다를 수 있다는 것이다. 이에 관해, 운영 체제(144), 애플리케이션 프로그램(145), 기타 프로그램 모듈(146) 및 프로그램 데이터(147)에 다른 번호가 부여되어 있다는 것은 적어도 이들이 다른 사본(copy)이라는 것을 나타내기 위한 것이다. 사용자는 키보드(162), 마이크(163) 및 마우스, 트랙볼(trackball) 또는 터치 패드와 같은 포인팅 장치(161) 등의 입력 장치를 통해 명령 및 정보를 컴퓨터(110)에 입력할 수 있다. 다른 입력 장치(도시 생략)로는 마이크, 조이스틱, 게임 패드, 위성 안테나, 스캐너 등을 포함할 수 있다. 이들 및 기타 입력 장치는 종종 시스템 버스에 결합된 사용자 입력 인터페이스(160)를 통해 처리 장치(120)에 접속되지만, 병렬 포트, 게임 포트 또는 USB(universal serial bus) 등의 다른 인터페이스 및 버스 구조에 의해 접속될 수도 있다. 모니터(191) 또는 다른 유형의 디스플레이 장치도 비디오 인터페이스(190) 등의 인터페이스를 통해 시스템 버스(121)에 접속될 수 있다. 모니터 외에, 컴퓨터는 스피커(197) 및 프린터(196) 등의 기타 주변 출력 장치를 포함할 수 있고, 이들은 출력 주변장치 인터페이스(190)를 통해 접속될 수 있다.

<27> 컴퓨터(110)는 원격 컴퓨터(180)와 같은 하나 이상의 원격 컴퓨터로의 논리적 접속을 사용하여 네트워크화된 환경에서 동작할 수 있다. 원격 컴퓨터(180)는 또 하나의 퍼스널 컴퓨터, 핸드-헬드 장치, 서버, 라우터, 네트워크 PC, 피어 장치 또는 기타 통상의 네트워크 노드일 수 있고, 통상적으로 컴퓨터(110)와 관련하여 상술된 구성요소들의 대부분 또는 그 전부를 포함한다. 도 1에 도시된 논리적 접속으로는 LAN(171) 및 WAN(173)이 있지만, 기타 네트워크를 포함할 수도 있다. 이러한 네트워킹 환경은 사무실, 전사적 컴퓨터 네트워크(enterprise-wide

computer network), 인트라넷, 및 인터넷에서 일반적인 것이다.

<28> LAN 네트워킹 환경에서 사용될 때, 컴퓨터(110)는 네트워크 인터페이스 또는 어댑터(170)를 통해 LAN(171)에 접속된다. WAN 네트워킹 환경에서 사용될 때, 컴퓨터(110)는 통상적으로 인터넷과 같은 WAN(173)을 통해 통신을 설정하기 위한 모뎀(172) 또는 기타 수단을 포함한다. 내장형 또는 외장형일 수 있는 모뎀(172)은 사용자 입력 인터페이스(160) 또는 기타 적절한 메커니즘을 통해 시스템 버스(121)에 접속된다. 네트워크화된 환경에서, 컴퓨터(110) 또는 그의 일부와 관련하여 기술된 프로그램 모듈은 원격 메모리 저장 장치에 저장될 수 있다. 예로서, 도 1은 원격 애플리케이션 프로그램(185)이 원격 컴퓨터(180)에 있는 것으로 도시하고 있지만 이에 제한되는 것은 아니다. 도시된 네트워크 접속은 예시적인 것이며 이 컴퓨터들 사이에 통신 링크를 설정하는 기타 수단이 사용될 수 있다는 것을 이해할 것이다.

<29> 도 2는, 네트워크 환경(200)에 저장된 패키지(210)와, 소스 리소스(211)와 복수의 타겟 리소스(213 내지 215, 221) 간의 관계를 예시한다.

<30> 도 1 및 도 2를 참조해 보면, 패키지(210)가 컴퓨터(110) 상에 저장된다. 컴퓨터(110)는 패키지(210)의 현재 위치에 관련된 어드레스에서 복수의 다른 패키지를 저장한다. 패키지(210)는 소스 리소스(211), 관계 리소스(212), 타겟 리소스(213 내지 215)를 저장하는 컨테이너이다. 또한, 패키지(210)는 복수의 다른 소스 리소스 및 타겟 리소스(도시하지 않음)를 저장할 수 있다.

<31> 소스 리소스(211)는 예를 들어 이미지, 웹 페이지, 다큐먼트, 비디오 또는 패키지(210)와 유사한 패키지와 같은 파일일 수 있다. 소스 리소스(211)는 타겟 리소스(213 내지 215, 221)와 관련된다. 도시하진 않았지만, 소스 리소스(211)는 패키지(210)와 관련되어 있는 패키지에 포함된 타겟 리소스와 관련될 수도 있다. 소스 리소스(211)는 관계 리소스(212)와 관련된다. 소스 리소스(211)는 관계가 시작되는 리소스이다.

<32> 타겟 리소스(213 내지 215, 221)는 예를 들어 이미지 파일, 웹 페이지, 다큐먼트, 비디오 또는 패키지(210)와 유사한 패키지와 같은 파일일 수 있다. 타겟 리소스(213 내지 215)는 소스 리소스(211)에 대한 로컬 리소스인 반면, 타겟 리소스(221)는, 네트워크(230)를 통해 액세스가능하며 원격 컴퓨터(180) 상에 저장된 외부 리소스이다. 타겟 리소스(221)는 원격 컴퓨터(180) 상의 데이터베이스(220)에 저장된다. 타겟 리소스(213 내지 215, 221)는 관계가 종료되는 리소스이다. 그러나, 타겟 리소스(213 내지 215, 221)는, 타겟 리소스가 관계 리소스(212)와 유사한 관계 리소스와 관련되어 있을 때 소스 리소스와 관련된 특성들을 취할 수 있다.

<33> 관계 리소스(212)는 소스 리소스(211)가 타겟 리소스(213 내지 215, 221)와 관련되어 있음을 가리키는 정보를 저장한다. 이 정보는, 관계 정보라 칭하며, 리소스 계층 구조를 구축하거나 소스 리소스(211)에 대한 메타데이터를 제공하는 데 이용될 수 있다. 관계 리소스(212)의 추가 상세를 도 3을 참조하여 후술한다.

<34> 도 3은 본 발명에 의해 활용되는 관계 데이터 구조(300)를 예시한다. 도 2 및 도 3을 참조해 보면, 관계 데이터 구조(300)는 관계 리소스(212)를 정의한다. 관계 태그(310)는 소스 리소스와 복수의 타겟 리소스 간의 복수의 관계를 정의한다. 관계 태그(310)는, 스키마를 정의하는 네임스페이스 엘리먼트(315), 및 복수의 관계를 위한 시맨틱을 포함할 수 있다. 스키마 및 시맨틱은 관계 데이터 구조를 위한 중립적 콘텐츠 인코딩을 정의할 수 있다.

<35> 관계 태그(320)는 소스 리소스와 복수의 타겟 리소스 중 하나의 타겟 리소스 간의 단일 관계를 정의한다. 복수의 관계 태그(320)는 관계 태그(310) 내에 내포(nest)될 수 있다. 관계 태그(320)는 식별 엘리먼트(325), 유형 엘리먼트(326), 타겟 엘리먼트(327), 및 타겟베이스(TargetBase) 엘리먼트(328)를 포함한다.

<36> 식별 엘리먼트(325)는 관계 태그(310) 내의 관계를 고유하게 식별한다. 식별 엘리먼트(325)는 변경될 수 없으며 확장성 생성 언어(XML) 식별자처럼 유효한 식별자이어야 한다. 식별자의 유효성은 예를 들어 XML 스키마와 같이 네임스페이스 엘리먼트(315)에 의해 특정되는 스키마에 따라 검증된다.

<37> 유형 엘리먼트(326)는 네임스페이스 엘리먼트(315)에 의해 또는 애플리케이션 프로그램에 의해 특정되는 값들의 범위로 한정될 수 있다. 유형 엘리먼트(326)는 관계의 시맨틱을 더 정의한다. 예를 들어, 유형 엘리먼트(326)는 소스 리소스가 타겟 리소스를 요구하거나 필요로 하는지 여부를 정의할 수 있다.

<38> 유형 엘리먼트(326)와 유사하게, 타겟베이스 엘리먼트(328)는 소스 리소스와 타겟 리소스 간의 관계의 시맨틱을 더 정의한다. 타겟베이스 엘리먼트(328)는 타겟 리소스가 내부형, 로컬형, 상대적, 또는 외부형인지 여부를 정의할 수 있다.

<39> 타겟 엘리먼트(327)는 타겟 리소스의 위치를 정의한다. 이 위치는, 팩 URI(Uniform Resource Identifier), 절

대적 URL(Uniform Resource Locator), 또는 상대적 URL과 같은 어드레스 식별자일 수 있다. 팩 URI는 패키지를 참조하는 데 활용되는 어드레싱 방식이다. 팩 URI 어드레싱 방식은 전체 패키지에 대한 요구 또는 패키지에 포함된 특정 리소스에 대한 요구를 가능하게 한다. 팩 URI 어드레싱 방식의 상세한 설명은, 본 명세서에서 그 내용이 참고로 포함되며 "Pack URI Scheme to Identify and Reference Parts of a Package"라는 명칭으로 공동 계류중인 출원에 개시되어 있다.

<40> 따라서, 관계 데이터 구조(310)는, 패키지 내의 로컬형인 타겟 리소스, 서로 다른 컴퓨터 상에서 패키지 밖의 절대 위치에 있는 외부형 타겟 리소스, 및 패키지에 상대적인 위치에 있는 상대적 타겟 리소스에 대한 관계를 사용자가 탐색 및 설명할 수 있게 한다. 게다가, 관계 데이터 구조(310)는 소스 리소스 메타 데이터를 저장하는 위치를 제공한다. 메타 데이터 정보는, 주석, 및 인쇄나 표시 데이터와 같은 기술 데이터를 포함한다.

<41> 도 4는 도 3의 관계 데이터 구조(310)를 생성하는 방법을 예시하는 본 발명의 일실시예의 흐름도이다. 단계 S410에서, 애플리케이션 또는 사용자가 소스 리소스에 관계를 추가하는 요구를 발생한다. 다음으로, 단계 S420에서, 소스 리소스의 위치를 판정한다. 소스 리소스의 위치를 판정함에 따라, 단계 S430에서, 체크를 개시하여 관계 데이터 구조의 존재를 검증한다. 관계 데이터 구조가 존재하는 경우, 단계 S450에서, 후속 체크를 행하여 그 관계가 새로운 것인지를 검증한다. 이 관계는, 새로운 것이거나 수정되었다면 관계 데이터 구조에 저장되고, 그렇지 않은 경우 중복되는 관계로서 저장되지 않는다. 다른 일실시예서는, 관계가 네임스페이스 엘리먼트(315)에 의해 특정된 스키마 및 시맨틱에 따라 저장된다.

<42> 관계 데이터 구조가 존재하지 않는 경우, 단계 S440에서, 관계 데이터 구조가 인스턴스화되어 소스 리소스와 관련된다. 관계 데이터 구조는, 소스 리소스의 위치에 상대적인 위치에 폴더와 같은 컨테이너를 생성함으로써 소스 리소스와 관련될 수 있다. 후속하여, 관계 데이터 구조가 컨테이너에 저장된다. 본 발명의 일실시예에서, 소스 리소스가 예를 들어 /content/spine.xml에 위치하면, 관계 데이터 구조는 /content/jrels/에서 인스턴스화(instantiate)되며 "spine.xml.rels."으로 명명된다. 이러한 명명 관습을 고수하도록, 소스 리소스의 위치에 있는 "_rels,"라는 서브 컨테이너에 관계 데이터 구조가 저장될 수 있으며, 관계 데이터 구조의 이름은 ".rels."로 결합(concatenate)된 소스 리소스의 이름으로 된다. 유사한 명명 관습을 이용하여 모든 관계 데이터 구조를 모든 소스 리소스와 관련지음으로써, 소스 리소스의 위치에 기초하는 관계 데이터 구조에 대한 효율적인 액세스가 가능해진다.

<43> 관계 데이터 구조가 인스턴스화된 후에, 관계 데이터 구조는 예를 들어 식별, 타겟이나 유형 정보와 같은 관계 정보로 채워진다. 이러면, 단계 S460에서 이 관계 정보가 저장되어 관계 정보의 후속 검색이 가능해진다.

<44> 도 5는, 도 3의 관계 데이터 구조를 참조함으로써 하나 이상의 관계를 탐색하는 방법을 예시하는 본 발명의 다른 일실시예의 흐름도이다.

<45> 단계 S510에서 관계 데이터 구조를 디코딩하려는 사용자 또는 애플리케이션 관계 요구에 응답하여, 단계 S520에서 체크를 행하여 소스 리소스가 관계 데이터 구조와 관련되어 있는지 여부를 확정한다. 소스 리소스가 관계 데이터 구조와 관련되어 있지 않다면, 관계 데이터 구조가 디코딩되지 않는다. 반면에, 관계 데이터 구조가 존재하는 경우, 단계 S530에서 데이터 구조를 위한 네임스페이스를 판정한다. 이후, 단계 S540에서, 디코더 또는 판독기는 관계를 디코딩한다. 디코딩 후에, 단계 S550에서 하나 이상의 관계를 표시할 수 있다. 다른 일실시예에서는, 관계의 디코딩이 네임스페이스 엘리먼트(315)에 의해 특정된 스키마 및 시맨틱을 활용한다.

<46> 이에 따라, 복수의 타겟 리소스에 관한 소스 리소스는 상술한 방법에 따라 탐색되는 복수의 관계를 갖는다. 또한, 본 발명의 실시예는 소스 리소스를 디코딩하지 않고서 디코더가 소스 리소스와 관련된 복수의 관계를 디코딩하는 것을 가능하게 한다. 디코더는, 복수의 관계의 콘텐츠 인코딩이 소스나 타겟 리소스 인코딩과 다른 포맷으로 되어 있는 경우, 관계를 디코딩할 수 있다. 따라서, 안전하거나 암호화된 소스 리소스 및 타겟 리소스들의 보안을 침해하지 않고서 안전하거나 암호화된 소스 리소스 및 타겟 리소스들 간의 관계를 판정할 수 있다.

<47> 본 발명의 상술한 설명은 예시이며, 당업자에겐 구성 수정 및 구현이 가능하다. 예를 들어, 본 발명은 일반적으로 도 1 내지 도 5를 참조하여 설명되었지만, 이러한 설명은 예시이다. 이에 따라, 본 발명의 범위는 다음에 따르는 청구범위에 의해 한정되는 것이다.

도면의 간단한 설명

<12> 첨부 도면을 참조하여 본 발명을 상세히 설명한다.

<13> 도 1은 본 발명을 구현하는 데 적합한 컴퓨팅 환경을 예시하는 블록도이다.

<14> 도 2는 네트워크 환경에 저장된 패키지와, 소스 리소스와 복수의 타겟 리소스 간의 복수의 관계를 예시한다.

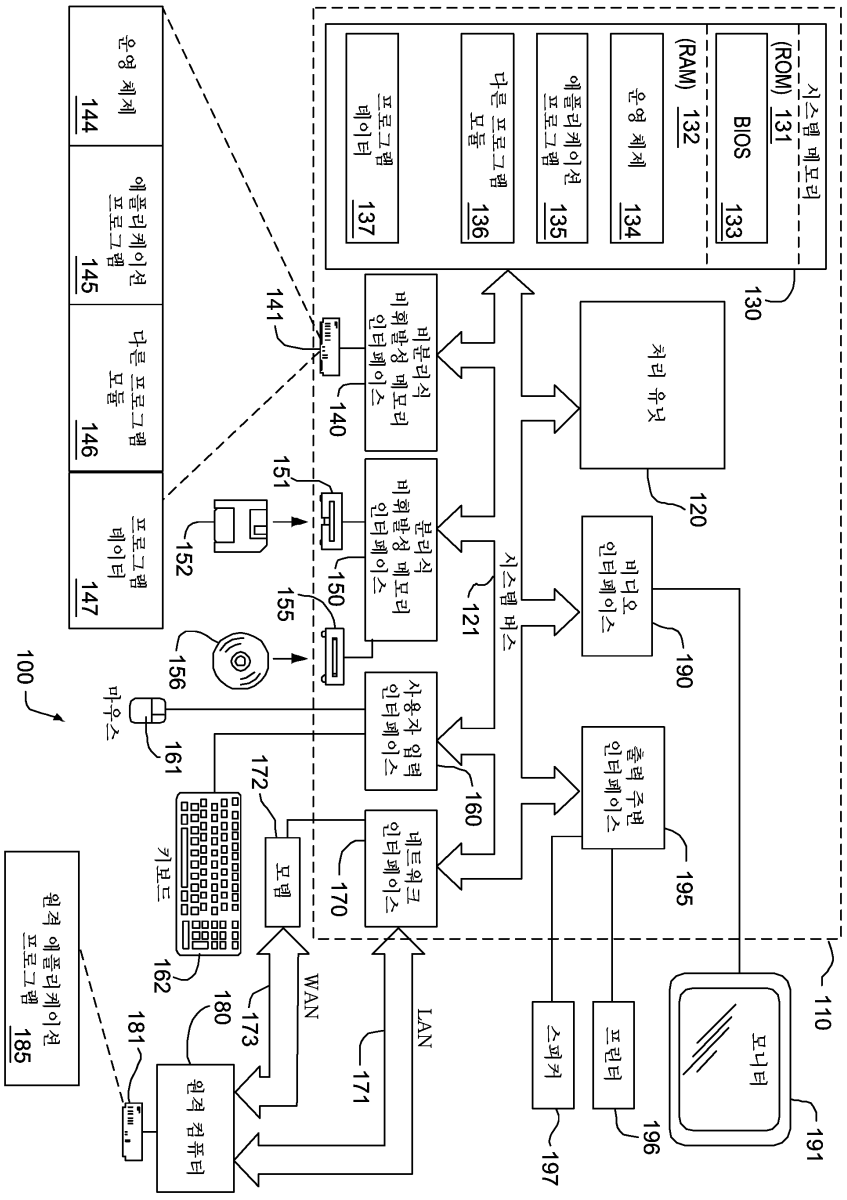
<15> 도 3은 본 발명에 의해 활용되는 관계 데이터 구조를 예시한다.

<16> 도 4는 도 3의 관계 데이터 구조를 생성하는 방법을 예시하는 본 발명의 일실시예의 흐름도이다.

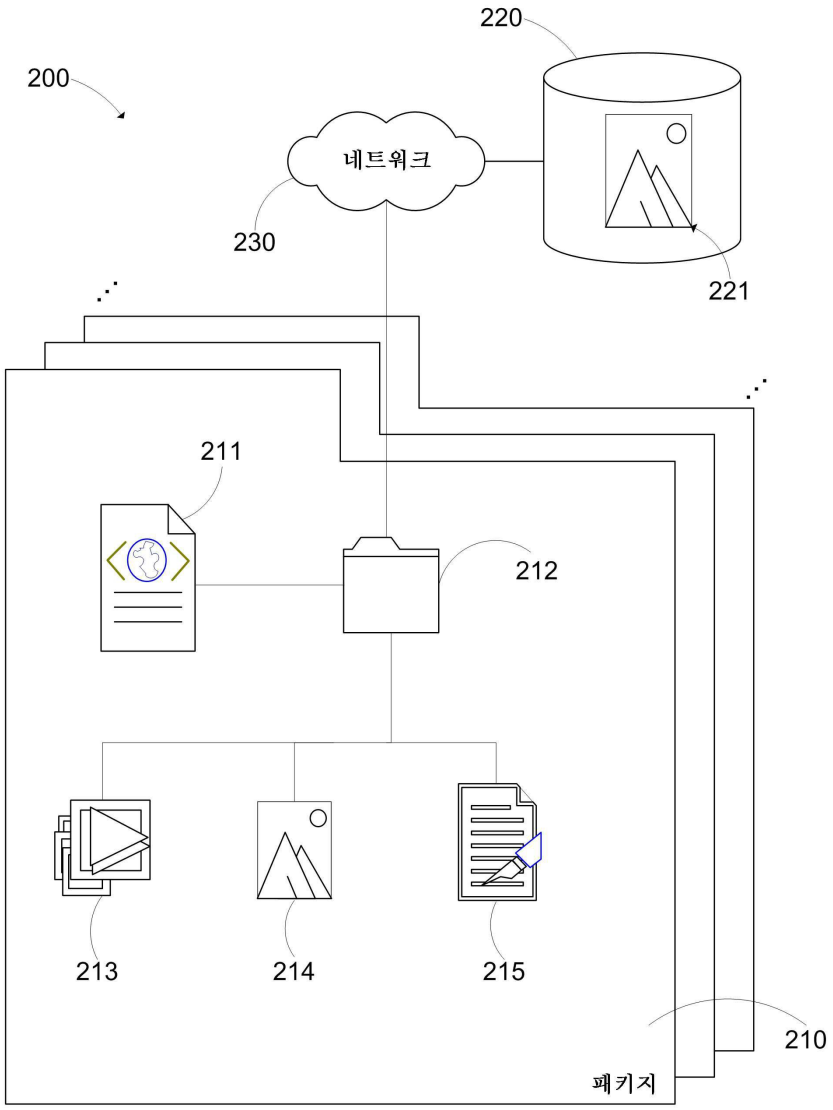
<17> 도 5는 도 3의 관계 데이터 구조를 참조함으로써 하나 이상의 관계를 탐색하는 방법을 예시하는 본 발명의 일실시예의 흐름도이다.

도면

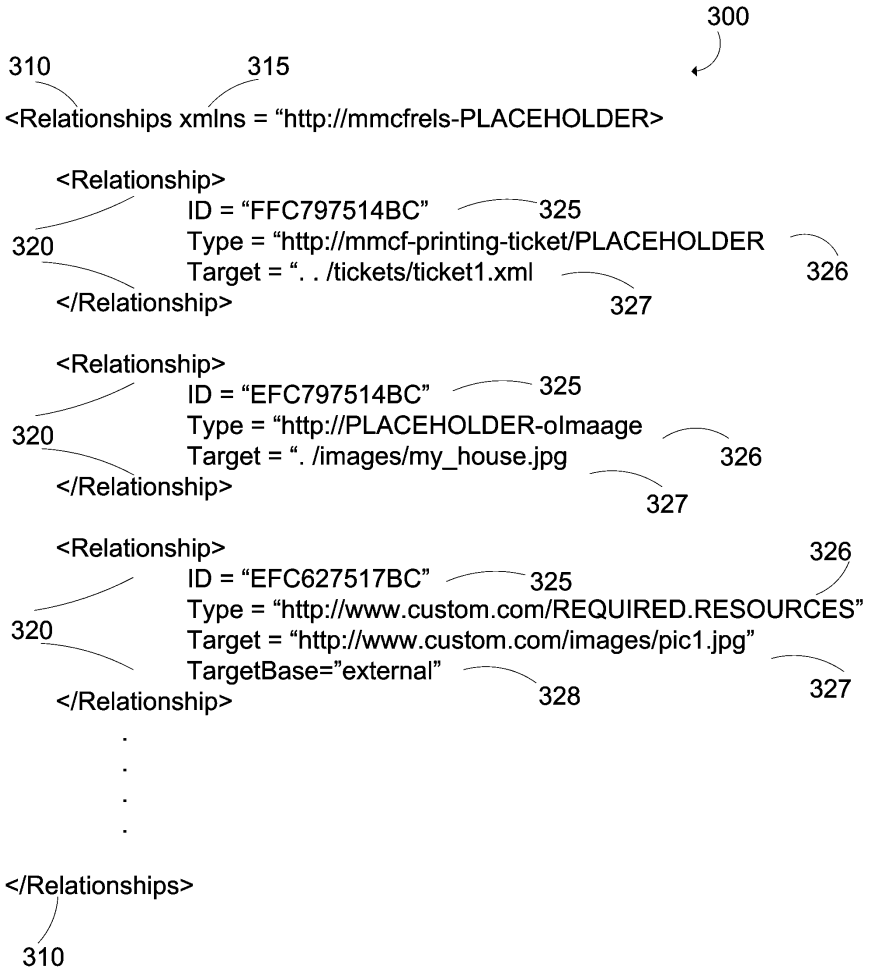
도면1



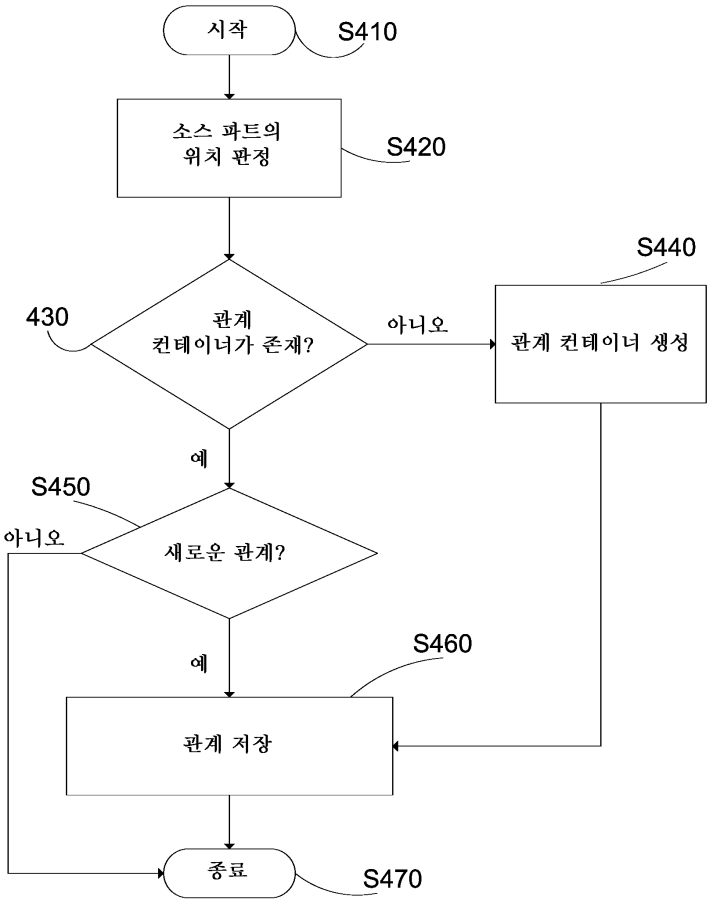
도면2



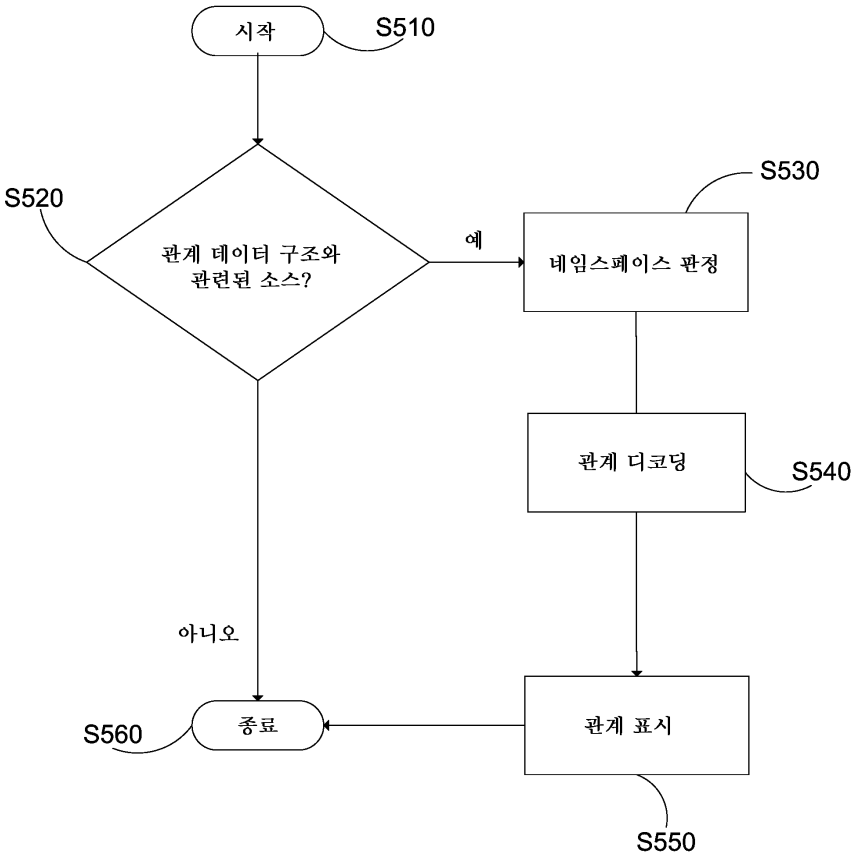
도면3


300
310
315
<Relationships xmlns = “http://mmcfrels-PLACEHOLDER>
<Relationship>
320
ID = “FFC797514BC”
325
Type = “http://mmcf-printing-ticket/PLACEHOLDER
326
Target = “. . /tickets/ticket1.xml
327
</Relationship>
<Relationship>
320
ID = “EFC797514BC”
325
Type = “http://PLACEHOLDER-olmaage
Target = “. /images/my_house.jpg
326
327
</Relationship>
<Relationship>
326
320
ID = “EFC627517BC”
325
Type = “http://www.custom.com/REQUIRED.RESOURCES”
Target = “http://www.custom.com/images/pic1.jpg”
TargetBase=”external”
327
328
</Relationship>
.
.
.
.
</Relationships>
310

도면4


시작
S410
소스 파트의
위치 판정
S420
S440
430
관계
컨테이너가 존재?
아니오
관계 컨테이너 생성
예
S450
아니오
새로운 관계?
예
S460
관계 저장
종료
S470

도면5


시작
S510
S520
관계 데이터 구조와
관련된 소스?
예
S530
네임스페이스 판정
관계 디코딩
S540
아니오
S560
종료
관계 표시
S550