(19) 대한민국특허청(KR)
(12) 등록특허공보(B1)

(45) 공고일자 2010년11월15일
(11) 등록번호 10-0994638
(24) 등록일자 2010년11월09일

(51) Int. Cl.
G06F 9/00 (2006.01)
(21) 출원번호 10-2004-0041105
(22) 출원일자 2004년06월05일
심사청구일자 2009년05월04일
(65) 공개번호 10-2004-0105584
(43) 공개일자 2004년12월16일
(30) 우선권주장
10/456,139 2003년06월06일 미국(US)
(56) 선행기술조사문헌
US5504885 A
US6061515 A
전체 청구항 수 : 총 24 항

(73) 특허권자
마이크로소프트 코포레이션
미국 워싱턴주 (우편번호 : 98052) 레드몬드 원 마이크로소프트 웨이
(72) 발명자
손킨,드미트리
미국 98052 워싱턴주 레드몬드 유니트 케이-140 올드 레드몬드 알디.7250
우리스,미첼
미국 98052 워싱턴주 레드몬드 엔.이. 22번 코트 17511
(74) 대리인
주성민, 이중희, 백만기

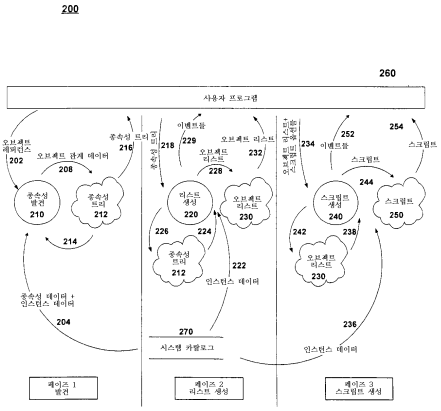
심사관 : 복진요

(54) 데이터베이스 오브젝트 스크립트 생성 방법 및 시스템

(57) 요 약

관계 데이터베이스 내의 여러가지 복잡성 오브젝트의 자동 스크립팅은 시스템 내로 패스된 단일 또는 다수의 참조 오브젝트 레퍼런스로부터 계층적 오브젝트 트리를 작성하는 것을 포함한다. 중복 오브젝트 레퍼런스들은 계층적 트리의 형성 시에 제거된다. 종속성 리스트는 계층적 트리로부터 생성된다. 종속성 리스트는 종속성 제약을 만족시키기 위해 오브젝트들이 작성되어야 되는 방식으로 순서화된 오브젝트들의 선형 리스트를 나타낸다. 스크립트는 타겟 데이터베이스 상으로의 여러가지 복잡성 오브젝트들의 전개를 허용하는 종속성 리스트로부터 생성된다. 스크립트는 종속성 리스트 내의 각 오브젝트를 인스턴스화하고, 그 오브젝트에 관한 사전-구성된 스크립트 방법을 호출함으로써 생성된다. 각각의 페이즈는 독립적으로 동작될 수 있다.

대 표 도 - 도2



특허청구의 범위

청구항 1

관계 데이터베이스(relational database) 내의 복수의 오브젝트 레퍼런스들(object reference)을 위한 SQL 스크립트(SQL script)를 생성하는 방법으로서,

상기 관계 데이터베이스에 대해 스크립트될 적어도 하나의 오브젝트 레퍼런스를 수신하는 단계;

상기 적어도 하나의 오브젝트 레퍼런스의 관계 종속성(relational dependency)들을 검출하는 단계;

상기 적어도 하나의 오브젝트 레퍼런스를 포함하는 계층적 오브젝트 트리(hierarchical object tree)를 구성하는 단계;

상기 검출된 관계 종속성들에 기초하여 상기 계층적 오브젝트 트리로부터 종속성 리스트(dependency list)를 획득하는 단계 - 상기 종속성 리스트는 사용자 프로그램을 통해 편집 가능하며, 오브젝트들이 생성될 순서를 나타내는 선형 리스트를 포함함 - ; 및

상기 종속성 리스트에 대하여 상기 SQL 스크립트를 생성하는 단계 - 상기 SQL 스크립트는 타겟 데이터베이스 내에 관계 데이터베이스 오브젝트들을 배치함 -

를 포함하는 방법.

청구항 2

제1항에 있어서,

상기 구성하는 단계는, 중복(duplicate) 오브젝트 레퍼런스들을 제거하는 단계와, 비중복(non-duplicated) 오브젝트 레퍼런스들 및 연관된 메타데이터(associated metadata)를 계층적 오브젝트 트리 내로 입력하는 단계를 포함하는, 방법.

청구항 3

제1항에 있어서,

상기 구성하는 단계는, 경과를 표시하는 이벤트들을 트리거링하는 단계와, 옵션의 오브젝트 조작(optional object manipulation)을 제공하는 단계를 포함하는, 방법.

청구항 4

제3항에 있어서,

상기 옵션의 오브젝트 조작은, 하나 이상의 사용자 및 프로그램에 의해 선택되는 오브젝트 레퍼런스가 삭제될 수 있는 적어도 하나의 오브젝트 레퍼런스들의 필터링을 포함하는, 방법.

청구항 5

제4항에 있어서,

상기 옵션의 오브젝트 조작은, 상기 선택된 오브젝트 레퍼런스 및 상기 선택된 오브젝트 레퍼런스에 종속되는 모든 후속 레퍼런스들의 제거를 포함하는, 방법.

청구항 6

제1항에 있어서,

상기 종속성 리스트는 종속성 제약(dependency constraint)들을 만족시키기 위해 오브젝트 생성 순서로 표현되는 선형 리스트를 포함하는, 방법.

청구항 7

제1항에 있어서,

상기 획득하는 단계는, 재귀적으로(recursively) 오브젝트 레퍼런스들의 하위 층 종속성(lower tier dependency)들을 거쳐 진행하고, 상기 하위 층 종속성들이 삭제될 수 있는 옵션의 오브젝트 조작을 제공하는, 방법.

청구항 8

제1항에 있어서,

상기 생성하는 단계는, 스크립팅 옵션(scripting option)들이 관계 데이터베이스 스크립트를 변경할 수 있게 하는 단계를 포함하는, 방법.

청구항 9

제1항에 있어서,

상기 관계 데이터베이스는 SQL 데이터베이스인, 방법.

청구항 10

제1항에 있어서,

상기 관계 종속성들은 부모-자식, 자식-손자, 및 손자-증손자 중의 하나 이상인, 방법.

청구항 11

제1항에 있어서,

상기 적어도 하나의 오브젝트 레퍼런스는 서버/데이터베이스/테이블을 포함하는 포맷을 사용하는 균일한 자원 이름을 포함하는, 방법.

청구항 12

제1항에 있어서,

경과 모니터링 및 오브젝트 조작 중 적어도 하나를 허용하는 하나 이상의 이벤트들을 트리거링하는 단계를 더 포함하는 방법.

청구항 13

제12항에 있어서,

오브젝트 조작은 오브젝트 레퍼런스의 추가, 변경 및 삭제 중 적어도 하나를 포함하는, 방법.

청구항 14

관계 데이터베이스 내의 하나 이상의 오브젝트들을 위한 SQL 스크립트를 생성하는 방법을 수행하기 위한 컴퓨터 실행가능 명령어들을 가지는 컴퓨터 판독가능 저장 매체로서,

상기 방법은,

상기 관계 데이터베이스에 대해 스크립트될 적어도 하나의 오브젝트 레퍼런스를 수신하는 단계;

상기 적어도 하나의 오브젝트 레퍼런스의 관계 종속성들을 검출하는 단계;

상기 적어도 하나의 오브젝트 레퍼런스를 포함하는 계층적 오브젝트 트리를 구성하는 단계;

상기 검출된 관계 종속성들에 기초하여 상기 계층적 오브젝트 트리로부터 종속성 리스트를 획득하는 단계 - 상기 종속성 리스트는 사용자 프로그램을 통해 편집 가능하며, 오브젝트들이 생성될 순서를 나타내는 선형 리스트를 포함함 - ; 및

상기 종속성 리스트에 대하여 상기 SQL 스크립트를 생성하는 단계 - 상기 SQL 스크립트는 타겟 데이터베이스 내에 관계 데이터베이스 오브젝트들을 배치함 -

를 포함하는, 컴퓨터 판독가능 저장 매체.

청구항 15

제14항에 있어서,

상기 구성하는 단계는, 중복 오브젝트 레퍼런스들을 제거하는 단계와, 비중복 오브젝트 레퍼런스들 및 연관된 메타데이터를 계층적 오브젝트 트리 내로 입력하는 단계를 포함하는, 컴퓨터 판독가능 저장 매체.

청구항 16

제14항에 있어서,

상기 구성하는 단계는, 경과를 표시하는 이벤트들을 트리거링하는 단계와, 옵션의 오브젝트 조작을 제공하는 단계를 포함하는, 컴퓨터 판독가능 저장 매체.

청구항 17

제16항에 있어서,

상기 옵션의 오브젝트 조작은, 오브젝트 레퍼런스들이 삭제될 수 있는, 적어도 하나의 오브젝트 레퍼런스들의 필터링을 포함하는, 컴퓨터 판독가능 저장 매체.

청구항 18

제17항에 있어서,

상기 옵션의 오브젝트 조작은, 오브젝트 레퍼런스들의 제거 및 상기 제거된 오브젝트 레퍼런스에 종속되는 모든 후속 레퍼런스들의 제거를 포함하는, 컴퓨터 판독가능 저장 매체.

청구항 19

제14항에 있어서,

상기 종속성 리스트는, 상기 적어도 하나의 오브젝트 레퍼런스가 종속성 제약들을 만족시키도록 생성되는 순서로 표현되는 선형 리스트를 포함하는, 컴퓨터 판독가능 저장 매체.

청구항 20

제14항에 있어서,

상기 획득하는 단계는, 재귀적으로 오브젝트 레퍼런스들의 하위 층 종속성들을 거쳐 진행하고, 상기 하위 층 종속성들이 삭제될 수 있는 옵션의 오브젝트 조작을 제공하는, 컴퓨터 판독가능 저장 매체.

청구항 21

제14항에 있어서,

상기 생성하는 단계는, 스크립팅 옵션들이 관계 데이터베이스 스크립트를 변경할 수 있게 하는 단계를 포함하는, 컴퓨터 판독가능 저장 매체.

청구항 22

제14항에 있어서,

상기 적어도 하나의 오브젝트 레퍼런스는 서버/데이터베이스/테이블을 포함하는 포맷을 사용하는 균일한 자원 이름을 포함하는, 컴퓨터 판독가능 저장 매체.

청구항 23

관계 데이터베이스 내의 오브젝트 레퍼런스들을 위한 SQL 스크립트를 생성하기 위한 컴퓨터 시스템으로서,

상기 관계 데이터베이스에 대해 스크립트될 상기 오브젝트 레퍼런스들을 수신하기 위한 입력 장치;

상기 SQL 스크립트를 생성하기 위한 컴퓨터 명령어들이, 상기 오브젝트 레퍼런스들의 관계 종속성들을 검출하는 단계, 상기 오브젝트 레퍼런스들을 포함하는 계층적 오브젝트 트리를 구성하는 단계, 상기 검출된 관계 종속성

들에 기초하여 상기 계층적 오브젝트 트리로부터 종속성 리스트를 획득하는 단계 - 상기 종속성 리스트는 사용자 프로그램을 통해 편집 가능하며, 오브젝트들이 생성될 순서를 나타내는 선형 리스트를 포함함 -, 및 상기 종속성 리스트에 대하여 상기 SQL 스크립트를 생성하는 단계를 수행하도록 실행되는 프로세서; 및

하나 이상의 표시 장치, 후속 컴퓨터 프로그램 및 저장 장치에 상기 SQL 스크립트를 전달하기 위한 통신 포트

를 포함하는 컴퓨터 시스템.

청구항 24

제23항에 있어서,

상기 적어도 하나의 오브젝트 레퍼런스는 서버/데이터베이스/테이블을 포함하는 포맷을 가지는 균일한 자원 이름을 포함하는, 컴퓨터 시스템.

청구항 25

삭제

청구항 26

삭제

청구항 27

삭제

청구항 28

삭제

청구항 29

삭제

청구항 30

삭제

청구항 31

삭제

청구항 32

삭제

청구항 33

삭제

청구항 34

삭제

청구항 35

삭제

청구항 36

삭제

명 세 서

발명의 상세한 설명

발명의 목적

발명이 속하는 기술 및 그 분야의 종래기술

[0018] 본 발명은 일반적으로 컴퓨터 데이터베이스에 관한 것으로, 더욱 구체적으로 관계(relational) 데이터베이스 내의 복잡한 오브젝트에 대한 자동 스크립트 생성에 관한 것이다.

[0019] 구조화 조회 언어(Structured Query Language: SQL)는 관계 데이터베이스와 통신하기 위해 사용된 미국 표준 협회(American National Standards Institute: ANSI)의 표준이다. SQL은 관계 데이터베이스 관리 시스템의 표준 언어이다. SQL 문장은 데이터 갱신 또는 관계 데이터베이스로부터의 데이터 검색과 같은 태스크를 실행하기 위해 사용된다. 다수의 데이터베이스 시스템들은 SQL을 사용하지만, 이러한 다수의 시스템들은 또한 보통 자신의 시스템 상에서만 사용되는 그들 자신의 추가적인 사유(propriety) 확장을 갖는다. 그러나, "선택(select)", "삽입(insert)", "갱신(update)", "삭제(delete)", "작성(create)" 및 "드롭(drop)"과 같은 표준 SQL 커맨드들은 관계 데이터베이스로 하고자 하는 많은 일들을 달성하기 위해 사용될 수 있다.

[0020] 관계 데이터베이스는 일반적으로 행 및 열을 포함하는 테이블로서 조직화된다. 임의의 행과 열의 교차부 또는 셀들에 대한 데이터 엔트리들은 셀들 내에 허용된 데이터형을 정의하기 위해 사용되는 한 세트의 데이터 제약(constraint)을 통상적으로 따른다. 그러한 데이터 엔트리들이 갖는 종래의 한가지 문제점은 데이터베이스 내에 배치될 필요가 있을 수 있는 크고 복잡한 오브젝트들에 대한 정의가 부족하다는 것이다. 이용된 SQL 데이터형에 관한 제약은 일반적으로 SQL과 같은 관계 데이터베이스 내로 입력될 수 있는 데이터의 종류를 제한한다. 큰 오브젝트는 SQL 데이터베이스 내로 배치된 경우에, 데이터 프로세싱 속도를 느리게 할 수 있고, 또는 하나 이상의 크고 복잡한 오브젝트를 수용하기 위한 메모리 및 프로세싱 시간과 같은 시스템 자원을 사용할 수 있다.

[0021] SQL과 같은 관계 데이터베이스 내에서의 조회 스크립팅 및 오브젝트 이용은 시간 낭비이고, 아주 전문적인 기술일 수 있다. 그러한 스크립트의 작성자는 타겟 관계 데이터베이스 상에서 적절한 메타데이터와 함께 오브젝트를 적절하게 인스턴스화하기 위해 기타 오브젝트들에 관한 새롭고 복잡한 오브젝트의 종속성들을 이해해야 한다. 이 스크립팅 전문기술은, 예를 들어 자신의 사업에 관련된 복잡한 오브젝트들의 추적 및 검색을 위해 자신의 관계 데이터베이스를 이용하고자 하는 평균 사용자의 경험으로는 알 수 없는 것일 수 있다. 대안적으로, 시스템 매니저들은 관계 데이터베이스의 유지보수를 돕기 위해 스크립팅을 사용할 수 있다. 이 스크립팅 활동은 시간이 걸리고, 준비하여 실행하는데 신경을 써야 한다. 현재 복잡한 오브젝트에 관한 관계 데이터베이스용 스크립트의 자동 생성은 용이하게 달성될 수 없다.

[0022] 그러므로, SQL 데이터베이스 내에서 참조되어 작동되기를 바라는 오브젝트에 대한 단일화 표현이 필요하다. 부수적으로, 애플리케이션 및 시스템 유지보수 작업을 위해 SQL과 같은 관계 데이터베이스 내로의 오브젝트의 전개를 용이하게 하기 위한 스크립트를 생성하는 메카니즘이 필요하다. 본 발명은 상술된 필요성에 역점을 두고, 관계 데이터베이스 메타데이터 구조의 복잡한 지식을 필요로 하지 않고 스크립트를 생성하기 위해 복잡한 종속성 트리 및 리스트를 작성하고 이들을 변경하는 여러가지 시스템, 방법 및 기술들로 그러한 필요성을 해결한다.

발명이 이루고자 하는 기술적 과제

[0023] 본 발명은 특히 SQL 데이터베이스 관리 시스템에 적용될 수 있는 관계 데이터베이스용 스크립터(scripter)를 포함한다. 다수의 독립된 소프트웨어 모듈 또는 다수의 펑션(function)들의 접합(concatenation)은 오브젝트 레퍼런스를 입력하고 스크립트를 출력한다. 본 발명의 예시적인 페이즈 또는 모듈은 패스되는 하나 이상의 오브젝트 레퍼런스로부터 계층적 오브젝트 트리를 작성한다. 오브젝트 레퍼런스들을 사용하는 복잡한 오브젝트들은 균일한 자원 이름들로 표현될 수 있다. 모듈은 중복된 오브젝트 레퍼런스를 제거하고, 종속성 트리를 생성한다. 모듈은 또한 완료후뿐만 아니라 작성되고 있는 상태에서도 트리를 편집할 기회를 제공한다.

[0024] 다른 예시적인 모듈 또는 페이즈는 사용자나 이전의 모듈로부터 계층적 종속성 트리를 입력하여, 종속성 리스트를 생성한다. 종속성 리스트는 종속성 제약을 만족시키기 위해 오브젝트가 사용할 수 있는 작성 순서를 표현하는 선형 리스트이다. 이 모듈은 또한 완료후뿐만 아니라 작성되고 있는 상태에서도 종속성 리스트를 편집할 기회를 제공한다.

[0025] 다른 예시적인 모듈 또는 페이즈는 종속성 리스트로부터 스크립트를 생성한다. 종속성 리스트는 사용자 생성방식일 수도 있고, 또는 이전의 모듈로부터 입력될 수도 있다. 이 모듈은 종속성 리스트 상의 오브젝트를 인스턴

스화하고, 오브젝트에 대응하는 스크립팅 방법을 호출한다. 이 모듈은 완료후뿐만 아니라 생성되고 있는 상태에서도 스크립트를 편집하도록 사용자 또는 제어 프로그램에 광범위한 유연성을 제공한다.

발명의 구성 및 작용

[0026] 다음의 양호한 실시예에 관한 상세한 설명뿐만 아니라 상기 설명은 첨부된 도면을 참조하면 더욱 잘 이해된다. 본 발명을 설명하기 위해, 본 발명의 예시적인 구성이 도면에 도시되어 있지만, 본 발명은 개시된 특정 방법 및 수단에 제한되지 않는다.

[0027] **개요**

[0028] 본 발명은 관계 데이터베이스 내의 여러가지 복잡성 오브젝트의 자동 스크립팅을 제공한다. 종속성 리스트가 프로세스 내로 입력되고, 종속성 또는 계층적 트리가 여러가지 오브젝트의 관계를 반영하여 생성되는 기술들이 제공된다. 이때, 종속성 또는 계층적 트리는 프로세스의 다른 부분으로 입력될 수 있는데, 여기에서 계층적 트리는 순서화 종속성 리스트 내로 프로세싱된다. 그 다음, 종속성 리스트는 소정의 타겟 데이터베이스 내에 관계 데이터베이스 오브젝트들을 전개하기 위해 사용될 수 있는 스크립트 내로 프로세싱될 수 있다. 본 발명의 여러가지 페이즈 또는 모듈은 따로따로 동작되거나 또는 협력하여 동작될 수 있다.

[0029] **예시적인 컴퓨팅 장치**

[0030] 도 1 및 다음 설명은 본 발명이 실현될 수 있는 적절한 컴퓨팅 환경에 대한 간략한 일반적인 설명을 제공하고자 하는 것이다. 그러나, 핸드헬드, 포터블 및 기타 컴퓨팅 장치들, 및 모든 종류의 컴퓨팅 오브젝트들이 본 발명과 관련하여 사용될 수 있다는 것을 이해할 수 있을 것이다. 그러므로, 이후 범용 컴퓨터가 설명되지만, 이것은 한 예일 뿐이며, 본 발명은 네트워크/버스 상호운용성 및 상호작용을 갖고 있는 클라이언트와 같은 기타 컴퓨팅 장치로 실현될 수도 있다. 그러므로, 본 발명은 매우 작거나 또는 최소한의 클라이언트 자원들이 관련되는 네트워크 호스트화 서비스의 환경, 예를 들어 클라이언트 장치가 단지, 하나의 기구, 또는 기타 컴퓨팅 장치들 내에 배치된 하나의 오브젝트, 및 오브젝트들과 같은 네트워크/버스로의 인터페이스로서 작용하는 네트워크 환경에서 실현될 수 있다. 본질적으로, 데이터가 저장되거나 데이터가 검색될 수 있는 임의의 장소는 본 발명에 따른 동작을 위한 바람직하거나 적합한 환경이다.

[0031] 요구되지는 않았지만, 본 발명은 장치 또는 오브젝트의 서비스 개발자에 의한 사용을 위해 운영 체계를 통해 실현될 수 있고/있거나, 본 발명에 따라 동작하는 응용 소프트웨어 내에 포함될 수 있다. 소프트웨어는 클라이언트 워크스테이션, 서버 또는 기타 장치와 같은 하나 이상의 컴퓨터에 의해 실행되는 프로그램 모듈과 같은 컴퓨터 실행가능 명령어들의 일반적인 문맥으로 기술될 수 있다. 일반적으로, 프로그램 모듈은 특정 태스크를 실행하거나 특정 추상 데이터형을 실현하는 루틴, 프로그램, 오브젝트, 컴포넌트, 데이터 구조 등을 포함한다. 전형적으로, 프로그램 모듈의 기능은 여러가지 실시예에서 원하는 대로 결합되거나 분산될 수 있다. 게다가, 본 분야에 숙련된 기술자들은 본 발명이 기타 컴퓨터 구성들로 실시될 수 있다는 것을 이해할 수 있을 것이다. 본 발명과 함께 사용하기 위해 적합하게 될 수 있는 그밖의 널리 알려진 컴퓨팅 시스템, 환경 및/또는 구성들은 퍼스널 컴퓨터, 현금 자동 입출금기(ATM), 서버 컴퓨터, 핸드-헬드 또는 랩탑 장치, 멀티-프로세서 시스템, 마이크로프로세서 기반의 시스템, 프로그램 가능 소비자 전자제품, 네트워크 PC, 전기제품, 조명, 환경 제어 장치, 미니컴퓨터, 메인프레임 컴퓨터 등을 포함하는데, 이것에 한정되지는 않는다. 본 발명은 또한 통신 네트워크/버스 또는 기타 데이터 송신 매체를 통해 링크되는 원격 프로세싱 장치에 의해 태스크가 실행되는 분산 컴퓨팅 환경에서 실시될 수 있다. 분산 컴퓨팅 환경에서, 프로그램 모듈은 메모리 저장 장치를 포함하는 구내 및 원격 컴퓨터 저장 매체 내에 위치될 수 있고, 클라이언트 노드는 교대로 서버 노드로서 작동할 수 있다.

[0032] 도 1은 본 발명이 구현될 수 있는 적절한 컴퓨팅 시스템 환경(100)의 예를 나타낸다. 컴퓨팅 시스템 환경(100)은 단지 적절한 컴퓨팅 환경의 일 예이며 본 발명의 사용 또는 기능의 범위에 제한을 가하도록 의도된 것은 아니다. 컴퓨팅 환경(100)은 예시적인 오퍼레이팅 환경(100)에 도시된 컴포넌트들 중의 임의의 하나 또는 조합에 관하여 임의의 종속성(dependency) 또는 요구사항(requirement)을 갖는 것으로 해석되어서는 안된다.

[0033] 도 1을 참조하면, 본 발명을 구현하기 위한 예시적인 시스템은 컴퓨터 시스템(110)의 형태의 범용 컴퓨팅 장치를 포함한다. 컴퓨터 시스템(110)의 컴포넌트들로는, 프로세싱 유닛(120), 시스템 메모리(130), 및 시스템 메모리를 포함하는 다양한 시스템 컴포넌트를 프로세싱 유닛(120)에 연결시키는 시스템 버스(121)가 포함될 수 있지만, 이에 한정되는 것은 아니다. 시스템 버스(121)는 다양한 버스 아키텍처 중의 임의의 것을 사용하는 로컬 버스, 주변 버스, 및 메모리 버스 또는 메모리 컨트롤러를 포함하는 몇가지 유형의 버스 구조 중의 임의의 것일 수 있다. 예로서, 이러한 아키텍처는 산업 표준 아키텍처(ISA) 버스, 마이크로 채널 아키텍처(MCA) 버스, 인핸

스드 ISA(Enhanced ISA; EISA) 버스, 비디오 일렉트로닉스 표준 어소시에이션(VESA) 로컬 버스, 및 (메자닌(Mezzanine) 버스로도 알려진) 주변 컴포넌트 상호접속(PCI) 버스를 포함하지만, 이에 한정되는 것은 아니다.

[0034] 컴퓨터 시스템(110)은 통상적으로 다양한 컴퓨터 판독가능 매체를 포함한다. 컴퓨터 판독가능 매체는 컴퓨터(110)에 의해 액세스될 수 있는 임의의 이용가능한 매체일 수 있으며, 휘발성 및 비휘발성 매체, 분리형(removable) 및 비분리형(non-removable) 매체를 둘다 포함한다. 예로서, 컴퓨터 판독가능 매체는 컴퓨터 저장 매체 및 통신 매체를 포함할 수 있지만, 이에 한정되는 것은 아니다. 컴퓨터 저장 매체는 컴퓨터 판독가능 명령, 데이터 구조, 프로그램 모듈 또는 다른 데이터와 같은 정보의 저장을 위한 임의의 방법 또는 기술로 구현되는 휘발성 및 비휘발성, 분리형 및 비분리형 매체를 둘다 포함한다. 컴퓨터 저장 매체는 RAM, ROM, EEPROM, 플래쉬 메모리 또는 기타 메모리 기술, CD-ROM, DVD(digital versatile disk) 또는 기타 광학 디스크 저장장치, 자기 카세트, 자기 테이프, 자기 디스크 저장장치 또는 기타 자기 저장장치, 또는 컴퓨터(110)에 의해 액세스될 수 있고 원하는 정보를 저장하는 데 사용될 수 있는 임의의 기타 매체를 포함할 수 있지만, 이에 한정되지 않는다. 통신 매체는 통상적으로 반송파 또는 기타 전송 메카니즘 등의 변조된 데이터 신호에 컴퓨터 판독가능 명령, 데이터 구조, 프로그램 모듈, 또는 다른 데이터를 구현하며, 임의의 정보 전달 매체를 포함한다. "변조된 데이터 신호"라는 용어는 신호 내에 정보를 인코딩하도록 설정되거나 변환된 특성을 하나 또는 그 이상을 갖는 신호를 의미한다. 예로서, 통신 매체는 유선 네트워크 또는 직접 유선 접속 등의 유선 매체와, 음향, RF, 적외선 및 기타 무선 매체 등의 무선 매체를 포함하지만, 이에 한정되지 않는다. 상술한 것들 중의의 임의의 조합이 컴퓨터 판독가능 매체의 범위 내에 포함되어야 한다.

[0035] 시스템 메모리(130)는 ROM(131) 및 RAM(132) 등의 휘발성 및/또는 비휘발성 메모리의 형태의 컴퓨터 저장 매체를 포함한다. 시동중과 같은 때에 컴퓨터(110) 내의 구성요소들간에 정보를 전송하는 것을 돕는 기본 루틴을 포함하는 기본 입출력 시스템(133; BIOS)은 일반적으로 ROM(131)에 저장된다. RAM(132)은 일반적으로 프로세싱 유닛(120)에 즉시 액세스될 수 있고 및/또는 프로세싱 유닛(120)에 의해 현재 작동되는 프로그램 모듈 및/또는 데이터를 포함한다. 예로서, (한정하고자 하는 것은 아님) 도 1은 오퍼레이팅 시스템(134), 애플리케이션 프로그램(135), 기타 프로그램 모듈(136), 및 프로그램 데이터(137)를 도시한다.

[0036] 컴퓨터 시스템(110)은 또한 다른 분리형/비분리형, 휘발성/비휘발성 컴퓨터 저장 매체를 포함할 수 있다. 단지 예로서, 도 1에는 비분리형 비휘발성 자기 매체로부터 판독하거나 그 자기 매체에 기록하는 하드 디스크 드라이브(140), 분리형 비휘발성 자기 디스크(152)로부터 판독하거나 그 자기 디스크에 기록하는 자기 디스크 드라이브(151), 및 CD-ROM 또는 기타 광학 매체 등의 분리형 비휘발성 광학 디스크(156)로부터 판독하거나 그 광학 디스크에 기록하는 광학 디스크 드라이브(155)가 도시되어 있다. 예시적인 오퍼레이팅 환경에서 사용될 수 있는 다른 분리형/비분리형, 휘발성/비휘발성 컴퓨터 저장 매체는 자기 테이프 카세트, 플래쉬 메모리 카드, DVD(Digital versatile disk), 디지털 비디오 테이프, 고체 RAM, 고체 ROM 등을 포함하지만 이에 한정되지 않는다. 하드 디스크 드라이브(141)는 일반적으로 인터페이스(140)와 같은 비분리형 메모리 인터페이스를 통해 시스템 버스(121)에 접속되고, 자기 디스크 드라이브(151) 및 광학 디스크 드라이브(155)는 일반적으로 인터페이스(150)와 같은 분리형 메모리 인터페이스에 의해 시스템 버스(121)에 접속된다.

[0037] 앞서 기술되고 도 1에 도시된 드라이브 및 그 관련 컴퓨터 저장 매체는 컴퓨터(110)를 위한 컴퓨터 판독가능 명령, 데이터 구조, 프로그램 모듈 및 기타 데이터의 저장을 제공한다. 도 1에서, 예를 들어, 하드 디스크 드라이브(141)는 오퍼레이팅 시스템(144), 애플리케이션 프로그램(145), 기타 프로그램 모듈(146), 및 프로그램 데이터(147)를 저장하는 것으로 도시된다. 이들 컴포넌트는 오퍼레이팅 시스템(134), 애플리케이션 프로그램(135), 기타 프로그램 모듈(136), 및 프로그램 데이터(137)와 동일할 수도 있고 다를 수도 있다. 오퍼레이팅 시스템(144), 애플리케이션 프로그램(145), 다른 프로그램 모듈(146), 및 프로그램 데이터(147)는 최소한 다른 복사본(different copies)임을 나타내기 위하여 다른 번호를 부여하였다. 사용자는 마우스, 트랙볼 또는 터치 패드라 불리우는 포인팅 장치(161), 키보드(162) 및 마이크로폰(163)과 같은 입력 장치를 통해 컴퓨터(110) 내로 명령 및 정보를 입력할 수 있다. (도시되지 않은) 기타 입력 장치는 마이크로폰, 조이스틱, 게임 패드, 위성 안테나, 스캐너 등을 포함할 수 있다. 이들 입력 장치 및 그외의 입력 장치는 시스템 버스에 연결된 사용자 입력 인터페이스(160)를 통해 종종 프로세싱 유닛(120)에 접속되지만, 병렬 포트, 게임 포트 또는 유니버설 시리얼 포트(USB) 와 같은 기타 인터페이스 및 버스 구조에 의해 접속될 수 있다. 모니터(191) 또는 다른 유형의 디스플레이 장치는 또한 비디오 인터페이스(190) 등의 인터페이스를 통해 시스템 버스(121)에 접속된다. 모니터외에도, 컴퓨터는 또한 출력 주변 인터페이스(195)를 통해 접속될 수 있는 스피커(197) 및 프린터(196) 등의 기타 주변 출력 장치를 포함할 수 있다.

[0038] 컴퓨터 시스템(110)은 원격 컴퓨터(180)와 같은 하나 이상의 원격 컴퓨터로의 논리적 접속을 이용한 네트워크

환경에서 동작할 수 있다. 원격 컴퓨터(180)는 퍼스널 컴퓨터, 서버, 라우터, 네트워크 PC, 피어(peer) 장치, 또는 기타 공통 네트워크 노드일 수 있으며, 비록 도 1 에는 메모리 저장 장치(181)만이 도시되어 있지만, 컴퓨터(110)에 관하여 상술한 구성요소 중 다수 또는 모든 구성요소를 일반적으로 포함할 수 있다. 도시된 논리적 접속은 근거리 통신망(LAN; 171) 및 원거리 통신망(WAN; 173)을 포함하지만, 그 외의 네트워크를 포함할 수도 있다. 이러한 네트워크 환경은 사무실, 기업 광역 컴퓨터 네트워크(enterprise-wide computer network), 인트라넷, 및 인터넷에서 일반적인 것이다.

[0039] LAN 네트워크 환경에서 사용되는 경우, 컴퓨터 시스템(110)은 네트워크 인터페이스 또는 어댑터(170)를 통해 LAN(171)에 접속된다. WAN 네트워크 환경에서 사용되는 경우, 컴퓨터 시스템(110)은 일반적으로 인터넷 등의 WAN(173)을 통해 통신을 구축하기 위한 모뎀(172) 또는 기타 수단을 포함한다. 내장형 또는 외장형일 수 있는 모뎀(172)은 사용자 입력 인터페이스(160) 또는 기타 적절한 메카니즘을 통해 시스템 버스(121)에 접속될 수 있다. 네트워크 환경에서, 컴퓨터(110)에 관하여 도시된 프로그램 모듈 또는 그 일부분은 원격 메모리 저장 장치에 저장될 수 있다. 예로서 (한정하고자 하는 것은 아님), 도 1은 메모리 장치(181)에 상주하는 원격 애플리케이션 프로그램(185)을 도시한다. 도시된 네트워크 접속은 예시적인 것이며, 컴퓨터들간의 통신 링크를 구축하는 그 외의 수단이 사용될 수 있다.

[0040] 다양한 분산형 컴퓨팅 네트워크는 퍼스널 컴퓨팅 및 인터넷을 중심으로 개발되어 왔으며 개발 중에 있다. 개인 및 비즈니스 사용자를 막론하고, 컴퓨팅 활동을 점점 웹 브라우저 지향적 또는 네트워크 지향적으로 만드는 애플리케이션 및 컴퓨팅 장치에 대하여 이음새없이 상호동작이 가능하고 웹 사용가능한 인터페이스가 제공된다.

[0041] 예를 들어, MICROSOFT®의 .NET™ 플랫폼은 서버, 웹-기반 데이터 저장과 같은 빌딩-블록 서비스, 및 다운로드 가능한 장치 소프트웨어를 포함한다. 예시된 실시예는 컴퓨팅 장치 상에 상주하는 소프트웨어와 관련하여 기재되지만, 본 발명의 하나 이상의 부분은, 본 발명에 따른 동작이 모든 .NET™의 언어 및 서비스에 의해 수행될 수 있거나, 지원될 수 있거나, 이를 통해 억세스될 수 있도록, 운영 체계, 애플리케이션 프로그래밍 인터페이스(API), 또는 임의의 코프로세서와, 표시장치와, 요구 오브젝트간의 "미들 맨(middle man)" 오브젝트를 통해 구현될 수 있을 뿐 아니라, 기타 분산현 컴퓨팅 프레임워크에서 구현될 수도 있다.

[0042] **본 발명의 예시적인 실시예**

[0043] 관계 데이터베이스 관리 오브젝트 스크립터는 오브젝트용 스크립트를 생성한다. 한 실시예에서, 오브젝트는 SQL 데이터베이스에서 실현된다. 본 발명은 SQL 환경 실시예의 관점에서뿐만 아니라 본 발명의 일반적인 응용 가능성의 관점에서 설명될 수 있다. 그러한 스크립트의 한 예는 트랜잭트(Transact)-SQL 스크립트이다. 스크립팅은 SQL 데이터베이스 상의 오브젝트를 수반하는 시스템 관리 태스크의 자동화를 가능하게 한다. 스크립터 오브젝트는 SQL 관리 오브젝트 명칭 공간 내에 존재하는 오브젝트이지만, 스크립터의 사용에 따라 좌우되는 SQL 관리 오브젝트는 없다. 스크립터는 각각의 인스턴스 클래스(instance class)에 의해 실현되는 2개의 동작(operation), 즉 작성 및 드롭 동작을 사용할 수 있다. 그러므로, 인스턴스 오브젝트들은 SQL 데이터베이스 상에서의 이들 오브젝트들의 작성 또는 삭제를 위한 스크립트 텍스트를 생성하기 위해 그들 자신의 인스턴스를 스크립팅할 책임만 있다. 결과적으로, 스크립터는 종속성 찾기, 메모리, 파일 또는 디스플레이로의 스크립트 출력, 인스턴스 오브젝트들을 스크립트하기 위한 인스턴스 오브젝트들의 호출, 및 스크립팅 동작의 문맥 및 경과의 제어를 포함하는 나머지 기능을 책임진다.

[0044] 스크립터 오브젝트 모델은 본 발명을 실현하기 위해 부분적으로 사용될 수 있다. 스크립터 오브젝트 모델은 스크립팅하기 위한 하나의 엔트리 포인트이다. 오브젝트 모델은 스크립팅 동작의 문맥을 보유한다. 모델 내의 오브젝트들은 균일한 자원 이름의 사용에 의해 유일하게 식별될 수 있다. 예를 들어, "dbo"에 의해 소유된 테이블 "authors"를 포함하는 "pubs"의 데이터베이스 이름을 갖는 SQL 관리 오브젝트는 균일한 자원 이름에 의해 유일하게 참조될 수 있다:

[0045] Server/Database[@Name='pubs']/table[@Name='authors' and schema='dbo']

[0046] 균일한 자원 이름의 사용은 표준 포맷을 제공하는데, 이 표준 포맷은 오브젝트를 유일하게 식별하고, 유연성 및 장래의 본 발명의 응용성 증가를 허용하는 기타 표준안 사용에 적합하다.

[0047] 스크립터는 하위(underlying) 데이터 구조의 중간 조작을 허용하는 다수의 별개의 페이즈에서 동작한다. 이들 페이즈는 전체로서 실행되어, 최소한의 조작처리를 허용하거나, 또는 독립적으로 실행되어, 하위 데이터 구조를 보정하는 최대 자유도를 줄 수 있다. 스크립터가 실행하는 예시적인 페이즈는 종속성을 발견하고, 종속성 리스

트를 생성하며, 종속성 리스트로부터의 스크립트를 생성하는 것이다.

[0048] 도 2는 페이즈들과 사용자 또는 기타 제어 프로그램 사이의 데이터 흐름뿐만 아니라 본 발명의 예시적인 페이즈들 사이의 관계를 도시한 블록도이다. 사용자 프로그램(260)은 예시적인 별개의 페이즈들: 발견, 리스트 생성 및 스크립트 생성 시에 본 발명과 상호작용할 수 있다. 각각의 페이즈 동안에, 사용자 프로그램은 특정 동작에 입력을 제공한다. 각각의 페이즈 동안에, 하나의 결과는 리턴되고, 이것은 조작 처리되어 프로세스의 다음 페이즈로 입력될 수 있다. 각각의 페이즈 동안에, 이벤트들은, 사용자 프로세스가 이벤트들에 가입되어 있으면, 프로세스의 경과에 대한 정보를 제공하는 사용자 프로세스에 보내진다. 이러한 이벤트 메카니즘은 사용자들이 이들 이벤트들에 관한 프로세스에 응답하여 그 프로세스와 상호작용할 수 있게 함으로써 사용자들이 개별 오브젝트에 대한 스크립터 동작을 변경할 수 있게 한다. 스크립트 생성 페이즈 동안에 패스된 스크립팅 옵션은 스크립터 출력의 전체 제어를 제공한다.

[0049] 각각의 페이즈는 바람직하게 분리되고, 각 페이즈동안의 입력은 스크립터에 의해 반드시 생성될 필요는 없다. 사용자 프로그램은 자신의 내부 알고리즘에 기초하여 자신의 데이터 구조를 작성하여, 이들 데이터 구조를 스크립터에 입력으로서 제공할 수 있다. 그러므로, 스크립터 페이즈들은 3개의 개별적인 구성요소로서 보일 수 있다.

[0050] 다시 도 2를 참조하면, 발견 페이즈는 오브젝트 레퍼런스(202)를 사용자 프로그램(260)으로부터 수신하여, 종속성 발견 메카니즘(210)을 개시한다. 오브젝트들은 균일한 자원 이름을 사용하여 참조될 수 있다. 종속성 데이터 및 인스턴스 데이터(메타데이터)(204)는 발견 프로세스를 돕기 위해 시스템으로부터 호출된다. 특히, 종속성 데이터는 서버 또는 시스템 카탈로그(270)로부터 검색가능한 종속성들을 통해 위치될 수 있다. 종속성의 세분성(granularity)은 또한 종속성 발견 동작의 세분성을 결정한다는 것을 알기바란다. 일단 2개 이상의 오브젝트들 간의 종속성이 종속성 검색 및 발견 또는 삭제 알고리즘을 사용하여 발견되면, 오브젝트 관계 데이터(208)는 종속성 트리(212) 내로 입력된다. 프로세스는 종속성 메카니즘(210)으로 되돌아가서(214), 종속성 트리에 대한 엔트리들의 작성을 계속한다. 프로세스의 끝에서, 종속성 트리(212)는 생성되어, 사용자 프로그램(260)에 이용가능하게 된다(216).

[0051] 종속성 검색 및 검출 기술은 자체 동작의 관점에서 설명될 수 있다. 부수적으로, 알고리즘은 부모 오브젝트들을 그래프화하고, 오리지널 오브젝트를 그래프화한 다음에, 자식 오브젝트들을 그래프화하는 것으로 설명될 수 있다. 그래프화하는 것은 본질적으로 종속성 트리를 구성하는 것이다. 초기에, 종속성 발견 메카니즘(210)은 오브젝트 레퍼런스를 보고, 그 오브젝트를 나열된 오브젝트로서 호출한다. 오브젝트의 실제 인스턴스화(instantiation)는 이 시점에서 요구되지는 않으므로, 오브젝트 자체는 호출되지 않고; 그 오브젝트의 레퍼런스만이 호출될 필요가 있다. 오브젝트는 공지된 오브젝트 리스트에 추가된다. 종속성 발견 메카니즘은 공지된 오브젝트 리스트 내에 없는 추가된 오브젝트의 각각의 부모를 찾는다. 그 다음, 종속성 메카니즘은 부모 오브젝트를 호출할 수 있고, 그것을 공지된 오브젝트 리스트에 추가할 수 있다. 가장 높은 층 부모가 검출되었을 때, 가장 높은 층 부모는 계층적 트리인 종속성 트리에 추가된다. 그 다음, 종속성 발견 메카니즘은 공지된 오브젝트 리스트 상에서 오브젝트들의 모든 자식을 찾기 위해 하향 검색을 한다. 이것은 발견된 부모들의 자식, 오리지널 오브젝트들, 자식과 손자 오브젝트들뿐만 아니라 종속성 트리에 오리지널 오브젝트를 추가한다.

[0052] 도 2의 페이즈 2는 종속성 리스트 생성과 관련된다. 종속성 리스트는 페이즈 1로부터 획득되거나(216), 또는 인터페이스(218)를 통해 사용자 프로그램(260)에 획득될 수 있다. 둘 중 어느 한가지 경우에, 종속성 리스트는 수신되고, 리스트 생성 메카니즘(220)은 오브젝트 리스트(230)를 생성하는 태스크에 대해 활성화된다. 리스트 생성 메카니즘(220)은 종속성 트리(212) 및 인스턴스 데이터(메타데이터)(222)와 인터페이스하여(224, 226), 오브젝트 리스트(230)로의 엔트리(228)를 생성한다. 오브젝트 리스트 엔트리는 오브젝트와 그 관계 데이터가 검출되면 생성될 수 있다. 프로세스는 종속성 트리로부터 추가 아이템을 판독하도록 되돌아가서, 종속성 트리 요소가 다 없어질 때까지 오브젝트 리스트를 생성한다. 오브젝트 리스트 엔트리들과 같은 이벤트들이 검출됨에 따라, 이벤트들(229)은 오브젝트 리스트를 편집함으로써 프로세스를 조작처리할 기회를 사용자 프로그램에 제공하기 위해 생성된다. 프로세스의 끝에서, 오브젝트 리스트(230)는 생성되어, 사용자 프로그램(260)에 이용가능하게 되며, 사용자 프로그램에 의해 완전히 편집가능하다.

[0053] 도 2의 페이즈 3은 오브젝트 리스트 및 스크립터 옵션(234)을 사용하여 스크립트 생성 메카니즘(240)을 통해 스크립트(예를 들어, 트랜잭트-SQL 스크립트)를 생성한다. 오브젝트 리스트는 이전의 페이즈에 의해 생성되었던 것과 동일한 오브젝트 리스트(232)일 수 있고, 또는 인터페이스(234)를 통해 사용자 프로그램(260)에 수신된 오브젝트 리스트일 수 있다. 둘 중의 어느 한 이벤트에서, 스크립트 생성 메카니즘(240)은 인스턴스 데이터(메타

데이터)(236)와 함께 사용자 프로그램에 의해 선택되거나 디폴트된 스크립팅 옵션(234)을 수신한다. 스크립트 생성기 메카니즘(240)은 오브젝트 리스트(230)가 프로세스됨(238, 242)에 따라 스크립트 엔트리(244)를 생성할 수 있다. 스크립트(250)가 생성됨에 따라, 새로운 엔트리들과 같은 이벤트들(252) 또는 생성된 스크립트를 편집할 기회가 사용자 프로그램(260)에 주어질 수 있다. 최종적으로, 오브젝트 리스트(230)는 완전히 프로세스되고, 완전한 스크립트(250)가 생성된다. 그 다음, 프로세스는 후속적인 프로세싱을 위해, 또는 사용자 또는 대등한 인터페이스로의 전달을 위해, 스크립트(254)를 사용자 프로그램에 전달할 수 있다.

[0054] 도 2에 도시된 바와 같이, 전체적인 스크립팅 프로세스는 오브젝트 레퍼런스의 수신(202)으로 시작하여 스크립트의 생성(254)으로 끝나는 하나의 프로세스로 간주될 수 있다. 대안적으로, 프로세스는 연관될 수 있는 다수의 독립적인 프로세스로 생각될 수도 있다. 도 3, 4 및 5는 각각 페이즈 1, 2 및 3의 개별적인 프로세스를 도시한 것이다.

[0055] 도 3은 본 발명의 예시적인 종속성 발견 페이즈 방법(300)의 흐름도이다. 종속성 발견 페이즈는 프로세스로 패스되어 있는 단일 또는 다수의 오브젝트 레퍼런스로부터 계층적 오브젝트 트리(그래프)를 작성한다. 프로세스는 SQL 데이터베이스와 같은 관계 데이터베이스 내에서 사용되도록 요구되는오브젝트들에 관련된 하나 이상의 오브젝트 레퍼런스를 수신(310)함으로써 시작된다. 그 다음, 프로세스는 참조된 오브젝트에 관련된 종속성들을 검출하도록 검색을 실행하는 단계로 이동한다. 종속성 데이터는 오브젝트들간의 관계에 관한 정보로서 정의된다. 예를 들어, SQL 내의 뷰(view)는 그것의 존재에 대한 관련 테이블에 종속될 수 있다. 이와 반대로, 관련 테이블은 종속된 SQL 뷰를 가질 수 있다. 부수적으로, 오브젝트 인스턴스들, 예를 들어 테이블 인스턴스 또는 메타데이터를 설명하는 인스턴스 데이터는 노출된 관계 종속성 데이터의 일부일 수 있다(320).

[0056] 종속성들이 검출됨에 따라, 종속성 또는 계층적 트리가 생성되어(330), 종속성 데이터 및 관련 오브젝트들을 나타낸다. 계층적 트리 또는 그래프를 작성하기 위해 사용된 알고리즘은 복잡한 계층을 스크립트하는 동안에, 레퍼런스들이 동일한 오브젝트에 발생할 수 있으므로, 중복된 오브젝트 레퍼런스를 제거한다(340). 2번 이상 참조되는 오브젝트들은 스크립트 아웃될 수 있다. 종속성 트리 또는 계층적 트리는 예를 들어, 부모, 첫번째 자식, 그다음 형제, 손자 및 증손자형 종속 구조를 포함할 수 있다. 종속성들이 검출되는 동안, 프로세스는 종속성 트리가 완료되었는지 검사한다(350). 완료되지 않았으면, 프로세스는 오브젝트들의 편집을 허용한다(360). 요청된 편집이 없으면(360), 프로세스는 다음 오브젝트 레퍼런스로 이동함으로써 계속된다(390). 편집이 요구되면(360), 트리는 편집될 수 있다(370). 예를 들어, 삭제된 오브젝트는 한 오브젝트의 삭제뿐만 아니라 관계 브랜치 내의 모든 후속적인 자식 오브젝트들의 삭제를 초래할 수 있다. 이것은 오브젝트들이 최종 그래프에 추가되기 전에 사용자가 오브젝트들을 조작하거나 필터할 수 있게 한다. 편집이 완료된 후, 프로세스는 다음 오브젝트 레퍼런스(380)로 이동하여, 종속성 데이터의 검색(320)을 계속한다.

[0057] 제어 프로세서 또는 사용자에 의해 요청된 오브젝트 편집이 없으면, 종속성 검출(300)은 다음 오브젝트 레퍼런스로 계속되어, 종속성 데이터의 검색(320)이 다시 시작된다. 최종적으로, 종속성 트리 또는 계층적 트리는 완료되고(350), 종속성 트리는 후속적인 프로세스로의 출력(399)으로서 또는 사용자 출력으로서 이용가능하게 된다(395).

[0058] 도 4는 본 발명의 예시적인 종속성 발견 워크(walk) 페이즈 방법(400)의 흐름도이다. 종속성 워크 페이즈는 프로세스로 보내진 종속성 트리로부터 종속성들의 선형 리스트를 작성한다. 프로세스는 프로세스로의 입력으로서 종속성 트리를 수신함으로써(410) 시작된다. 이 입력은 도 3의 출력과 같은 출력(399)으로부터 얻어질 수 있고, 또는 사용자 또는 진행 프로세스에 의해 따로 입력될 수 있다.

[0059] 도 4를 참조하면, 종속성 발견 워크 방법(400)은 종속성 트리로부터 얻어진 관계 종속성들의 검출(420)이 계속된다. 종속성들이 검출됨에 따라, 프로세스는 종속성 리스트 엔트리를 생성한다(430). 종속성 리스트는 종속성 제약을 만족시키기 위해 오브젝트가 작성되어야 하는 순서를 리스트화하는 선형 리스트가 바람직하다. 예를 들어, 사용자 정의 데이터형은 오브젝트 또는 사용자 데이터형에 의존하는 테이블 작성 이전에, 오브젝트의 일부로서, 가능하면 메타데이터로서 존재할 필요가 있을 수 있다.

[0060] 프로세스가 첫번째 과정에서 완료되지 않았다고 하면(440), 프로세스(400)는 작성된 리스트에 대한 편집을 허용한다. 리스트가 편집되어야 하면(450), 편집은 허용될 수 있고(460), 종속성 리스트 내에서의 참조된 오브젝트의 제거 또는 변경이 이루어질 수 있다. 편집이 완료된 후, 프로세스는 종속성 트리 내의 다음 오브젝트로 계속되고(470), 관계 종속성의 검출이 계속된다(420). 편집이 요구되지 않았으면(450), 프로세스 경과가 통지되고(480), 프로그램은 다음 트리 오브젝트 및 관계 종속성들의 계속된 검출(420)을 지시한다.

[0061] 최종적으로, 프로세스는 트리가 완전히 체크되고 종속성 리스트 작성의 프로세스가 완료될 때까지(440) 반복된다. 그 다음, 종속성 리스트는 후속적인 프로세서에 또는 사용자 가용성 출력(499)으로서 이용가능하게 될 수 있다(490).

[0062] 도 5는 본 발명의 예시적인 스크립팅 페이즈 방법(500)의 흐름도이다. 스크립팅 페이즈는 프로세스로 보내진 종속성 리스트로부터 트랜잭트-SQL 스크립트와 같은 스크립트를 생성한다. 프로세스는 프로세스로의 입력으로서 종속성 리스트를 수신함으로써(510) 시작된다. 이 입력은 도 4의 출력과 같은 출력(499)으로부터 얻어질 수 있고, 사용자 또는 진행 프로세스에 의해 따로 입력될 수 있다. 도 5를 참조하면, 스크립팅 페이즈(500)는 종속성 리스트 오브젝트 레퍼런스의 인스턴스화(520)로 이어진다. 그 다음에, 참조된 오브젝트에 대응하는 스크립트에 대해 오브젝트 레퍼런스에 관한 호출이 이루어진다. 호출에 기인한 스크립트 구성요소들의 복귀 시에, 스크립트는 프로세스가 반복됨에 따라 누적된다(540).

[0063] 프로세스가 완료되지 않으면(550), 프로세스(500)는 스크립트(560)의 편집을 고려한다. 편집이 요구되면, 사용자 또는 기타 제어 프로세스는 스크립트를 편집(570)한 다음에, 종속성 리스트 내의 다음 오브젝트로 계속되어(580), 다음 오브젝트 레퍼런스(520)의 다음 인스턴스화를 가능하게 한다. 편집이 요구되지 않으면(560), 경과 통지가 제어 프로세서 또는 사용자에게 표시되고(590), 프로세스(500)는 인스턴스화를 위한 다음 오브젝트로 계속된다.

[0064] 최종적으로, 프로세스는 종속성 리스트 상의 모든 오브젝트를 거쳐 진행되고, 프로세스는 완료된다(550). 그후, 누적된 스크립트는 후속적인 프로세스에 또는 사용자 가용성 출력(599)으로서 이용가능하게 된다(595).

[0065] 도 3, 4 및 5의 프로세스는 본 발명의 취지를 벗어나지 않고 프로세스 내의 소정의 시점에서의 경과 통지 또는 소정의 시점에서의 편집을 포함하거나 제외하도록 변경될 수 있다는 것을 알기 바란다. 예를 들어, 도 5에서, 스크립트 경과 모니터 시점은 본 발명으로부터 벗어나지 않고, 스크립트 누산기 기입(540)후, 또는 완료 체크(550)후, 또는 편집(570)후, 또는 스크립트가 이용가하게 된(595) 후에 배치될 수도 있다. 흐름도에 있어서의 임의의 모니터 시점의 변경은 본 발명의 정신을 변화시키지 않고 도 3, 4 및 5에 적절하게 동일하게 적용된다.

[0066] 본 발명의 실시예는 도 6의 UML(Uniform Modeling Language)도에 의해 표현된 아키텍처에 제시된다. UML도(600)는 스크립터 오브젝트 모델 내의 각각의 클래스(class)를 그림으로 설명한다. 이 실시예에서는 유틸리티(utility)가 SQL 데이터베이스 내에 있다고 하자. 종속성 워커(walker)((610)는 SQL 서버 데이터베이스 내에 포함되는 클래스들간의 관계 또는 종속성을 발견하는 기능을 제공한다. 이것은 스크립터 클래스의 베이스(base) 클래스이다. 종속성 워커(610)는 필터 델리게이트(delegate)(612) 및 경과 보고 델리게이트(614)를 사용한다. 필터 델리게이트(612)는 발견 페이즈 동안에 발견되는 클래스들의 거절 및/또는 변경을 허용하는 이벤트이다. 경과 보고 델리게이트(614)는 종속성 및/또는 스크립팅 페이즈 경과에 대한 경과 정보를 제공하는 이벤트이다.

[0067] 스크립터(620)는 최종 사용자들에게 스크립팅 기능을 밝히는 메인 스크립팅 클래스이다. 스크립터(620)는 경과 보고 델리게이트(614) 및 에러 이벤트 델리게이트(624)를 사용한다. 스크립팅 옵션(622)은 스크립터(620) 동작의 변경을 허용하는 클래스이다. 스크립팅 옵션(622)은 스크립터(620) 클래스의 특성으로서 노출된다. 그러나, 스크립팅 옵션 클랙스(622)는 따로 인스턴스화될 수 있고, 이것은 이 클래스가 개별 클래스에 관한 스크립트 방법의 인수로서 패스될 수 있게 한다.

[0068] 종속성 노드 클래스(630)는 오브젝트 레퍼런스로서 URN을 포함한다. 종속성 노드(630) 클래스는 모든 종속성 트리 또는 리스트 클래스의 베이스 클래스이다. 종속성 트리 노드 클래스(632)는 부모 및 자식 관계에 대한 정보를 보유하는 클래스이다. 이것은 종속성 노드 클래스(630)를 확장시킨다. 종속성 트리 클래스(634)는 스크립터 관계를 포함한다. 이 클래스는 종속성 트리 노드 클래스(632)를 확장시킬 수 있다. 종속성 트리 클래스(634)는 스크립터 클래스(620)의 특성으로서 노출된다.

[0069] 어레이 리스트(640)는 0 내지 n개의 클래스의 선형 리스트 내에 레퍼런스를 보유할 수 있는 .NET 프레임워크(framework)로부터의 표준 클래스이다. 어레이 리스트 클래스는 종속성 리스트 클래스(642)의 베이스 리스트로서 작용한다. 종속성 리스트 클래스(642)는 종속성 리스트 노드(644) 클래스의 선형 리스트를 보유할 수 있다. 그것은 종속성 리스트(642) 클래스에 의해 사용되고, 스크립터 클래스(620)의 특성으로서 노출된다. 종속성 리스트 노드(644)는 오브젝트가 오리지널 리스트의 일부(즉, 루트 오브젝트)였는지 아닌지에 관한 정보를 포함한다. 그것은 종속성 노드 클래스(630)를 확장시킨다.

[0070] 본 발명의 한 실시예에 따르면, 스크립터는 동작 및 유틸리티의 유연성을 허용하는 옵션을 가질 수 있다. 도 2

의 프로세스는 또한 필터를 적용함으로써 조작처리될 수 있다. 이 필터는 페이즈 2 및 3 동안에 호출되어, 오브젝트들이 이들 페이즈 동안에 제거되거나 변경될 수 있게 한다. 스크립터 생성기는 오브젝트들(자식 또는 더 낮은 층의 오브젝트들을 포함)을 제외하기 위해 사용될 수 있는 특성을 실현할 수 있다. 더욱이, 스크립팅 페이즈들 중의 한 페이즈 동안에 변경되는 오브젝트들은 또한 변경을 포함하여 스크립트 아웃될 수 있다.

[0071] 오브젝트 이름들은 오브젝트 상에서 이용가능하다면 명칭부여(naming) 특성을 변경함으로써 변화될 수 있다. 오브젝트들은 그러한 용도로 지속될 필요는 없다. 원한다면, 스키마는 오브젝트 상에서 이용가능하다면 스키마 특성을 설정함으로써 변화될 수 있다. 스크립팅 에러 이벤트는 스크립트 작성 페이즈 동안에 발생하는 에러를 수신하도록 설정될 수 있다. 이것은 옵션이 에러 발생 시에 계속될 수 있게 함으로써 에러가 발생할 때(즉, 오브젝트가 서버 상에서 발견될 수 없을 때) 계속되는 것으로 고려된다.

[0072] 시스템 오브젝트들은 사용자들이 시스템 오브젝트의 구조에 기초하여 새로운(비-시스템) 오브젝트들을 작성할 수 있도록 스크립트될 수 있다. 다음의 스크립트 규칙은 시스템 오브젝트에 적용할 수 있다:

[0073] (1) 사용자들은 시스템 오브젝트를 스크립터에 보내서 스크립트를 생성하는 것이 허용되어야 한다.

[0074] (2) 서버가 시스템 오브젝트들이 작성될 수 없게 할 수 있으므로, 이 스크립트는 사전 변경없이 서버 상에서 실행되어서는 안된다.

[0075] (3) 사용자가 다수의 오브젝트들을 보내고, 오브젝트들 중의 하나가 시스템 오브젝트이면, 시스템 오브젝트는 허용될 수 있다. 시스템 오브젝트가 발견동안 삭제되는 경우에 스크립팅을 중지하기 위한 옵션이 선택될 수 있다. 이것은 전체 데이터베이스, 또는 종속성을 포함하는 다수의 오브젝트를 스크립트 아웃할 때 중요해질 수 있다.

[0076] 상이한 스크립트 동작들은 하나의 패스에서 하나 이상의 오브젝트를 스크립하기 위해 사용될 수 있다. 선택적으로, 필터 기능이 패스될 수 있고, 이것은 오브젝트들을 유일하게 식별하기 위해 사용되는 균일한 자원 이름을 필터 아웃하기 위해 사용될 수 있다. 이 필터는 오브젝트가 발견 페이즈(예를 들어, 도 2의 페이즈 1) 동안에 종속성 트리에 추가될 때 호출된다. 이것은 주문화 목적에 도움이 될 수 있다. 필터된 오브젝트 및 그것의 모든 종속성은 스크립트되지 않는다.

[0077] 다른 실시예에서, 스크립팅 옵션은 본 발명의 일부로서 제공될 수 있다. 이들 옵션은 사용자, 또는 실행 또는 응용 프로그램과 같은 제어 프로그램이 스크립팅 프로세스를 통해 제어를 실행할 수 있게 한다. 몇몇의 옵션들이 아래에 설명된다:

[0078] **스크립트 출력 포맷 옵션**

[0079] 스크립터 옵션 설명

[0080] 파일에 첨부; 표시된 출력 파일에 첨부. 디폴트에 의해 스크립트 방법은 현존하는 파일에 중복기입.

[0081] ANSI 파일; 생성된 스크립트 파일은 다중 바이트 문자를 사용.

[0082] 드롭(Drops); 참조된 구성요소를 제거하기 위해 트랜잭트-SQL을 생성. 스크립트는 구성요소의 제거를 시도하기 전에 존재를 테스트.

[0083] PWD 암호화; 스크립트로 패스워드를 암호화.

[0084] 헤더 포함; 생성된 스크립트는 생성 날짜와 시간 및 기타 서술 정보를 포함하는 헤더가 앞에 놓임.

[0085] 존재하지 않는 경우를 포함; 컴포넌트를 작성하는 트랜잭트-SQL은 존재 체크에 의해 앞에 놓여짐. 스크립트가 실행될 때, 컴포넌트는 명칭부여된 컴포넌트의 카피가 존재하지 않는 경우에만 작성.

[0086] 커맨드 종결자 없음; 스크립트 내의 개별 트랜잭트-SQL 문장은 접속-특정 커맨드 종결자(terminator)를 사용하여 한정되지 않음. 디폴트에 의해, 개별 트랜잭트-SQL은 한정.

[0087] 스키마 부여; 오브젝트를 제거하기 위해 생성된 트랜잭트-SQL 내의 오브젝트 이름은 참조된 오브젝트의 소유자에 의해 부여. 참조된 오브젝트를 작성하기 위해 생성된 트랜잭트-SQL은 현재의 오브젝트 소유자를 이용하여 오브젝트 이름을 부여.

[0088] 스키마 부여 외래 키; 스키마 부여 테이블은 외래 키 제약에 대한 참조사 항을 싣고 있음.

[0089] 이진으로 타임스탬프; 테이블 또는 사용자-정의 데이터형의 오브젝트 작성을 스크립트할 때, 타임스탬프 데

이터형의 명세를 이진으로 변환.

[0090] 파일 전용; 대부분의 SQL 오브젝트 스크립팅 방법은 복귀 값 및 옵션의 출력 파일 둘다 지정. 사용된 경우에, 출력 파일이 지정되면, 이 방법은 스크립트를 호출자에게 복귀시키지 않고, 스크립트를 출력 파일에 기입하기만 한다. 종속성을 스크립트 아웃하는 것은 매우 큰 스트링을 초래할 수 있는 가능성이 있으므로, 스트링 출력이 요구되지 않을 때마다 이 옵션을 지정.

[0091] 유니코드 파일; 유니코드 출력은 디폴트에 의해 생성됨.

[0092] 로그인 SID; 스크립트된 로그인을 위한 보안 식별자를 포함.

[0093] DDL 헤더 전용; 저장된 절차와 같은 본문 텍스트를 갖는 오브젝트용의 DDL 헤더만을 스크립트. 디폴트는 완료 DDL을 스크립트 아웃하기 위한 것이다.

[0094] DDL 본문 전용; 저장된 절차와 같은 본문 텍스트를 갖는 오브젝트용 DDL 본문만을 스크립트. 디폴트는 완료 DDL을 스크립트 아웃하기 위한 것이다.

[0095] **스크립트 종속성 옵션**

[0096] 스크립터 옵션 설명

[0097] 종속성 포함; 모든 종속 오브젝트를 포함하도록 출력 스크립트리스트를 확장.

[0098] 데이터베이스 사용권한; 생성된 트랜잭트-SQL 데이터베이스는 스크립트를 정의하는 것을 특별히 허락. 데이터베이스 사용권한은 문장 실행 권한을 승인 또는 부인한다.

[0099] 스크립트 인덱스; 집중화(clustered) 인덱스, 비집중화 인덱스 및 DRI 인덱스는 OR 논리 연산자를 사용하여 조합됨. 테이블 및 뷰 오브젝트들 둘다에 적용.

[0100] 사용권한; SQL SMO 스크립트 오브젝트 사용권한 및 SQL SMO 스크립트 데이터베이스 사용권한은 OR 논리 연산자를 사용하여 조합됨.

[0101] 1차 오브젝트; 참조된 컴포넌트를 작성하는 트랜잭트-SQL을 생성.

[0102] 확장된 특성; 오브젝트 스크립팅의 일부로서 확장된 특성 스크립팅을 포함.

[0103] XML 명칭공간; 오브젝트 스크립팅의 일부로서 XML 명칭공간을 포함.

[0104] 전문 카탈로그; 커맨드 배치(batch)는 검색 전문(full-text) 카탈로그들을 작성하는 트랜잭트-SQL 문장을 포함한다.

[0105] **크로스(cross) 서버-레벨 스크립팅**

[0106] 스크립터 옵션 설명

[0107] 대조(collation) 없음; 디폴트는 대조를 생성하기 위한 것이다.

[0108] 테이블 오브젝트에 특정.

[0109] 전문 인덱스; 커맨드 배치는 검색 전문 인덱싱을 정의하는 문장을 포함.

[0110] 바인딩; sp-바인디폴트(bindefault) 및 sp-바인드룰(bindrule)을 생성. 스크립팅이 SQL 서버 테이블을 참조할 때만 적용.

[0111] 집중화 인덱스; 집중화 인덱스를 정의하는 트랜잭트-SQL을 생성. 스크립팅이 SQL 서버 테이블을 참조할 때만 적용.

[0112] DRI-전체(all); DRI로서 정의된 모든 값은 OR 논리 연산자를 사용 하여 조합됨.

[0113] DRI-전체 제약; DRI-체크, DRI 디폴트, DRI 외래 키, DRI 1차 키, DRI 유일 키, DRI XML 키는 OR 논리 연산자를 사용하여 조합됨.

[0114] DRI-전체 키; DRI 외래 키, DRI 1차 키, DRI 유일 키는 OR 논리연산자를 사용하여 조합됨.

[0115] XML 인덱스; 생성된 스크립트는 XML 인덱스를 작성.

[0116] DRI-체크; 생성된 스크립트가 열-특정 CHECK 제약을 작성한다. 선언 참조 무결성(declarative referential integrity)이 종속성 관계를 설정할 때 스크립팅을지시(direct). 스크립팅이 SQL 서버 테이블을 참조할 때만 적용.

[0117] DRI-집중화; 생성된 스크립트가 집중화 인덱스를 작성한다. 선언 참조 무결성이 종속성 관계를 설정할 때 스크립팅을 지시. 스크립팅이 SQL 서버 테이블을 참조할 때만 적용.

[0118] DRI-디폴트; 생성된 스크립트가 열 특정 디폴트를를 작성한다. 선언 참조 무결성이 종속성 관계를 설정할 때 스크립팅을 지시. 스크립팅이 SQL 서버 테이블을 참조 할 때만 적용.

[0119] DRI-외래 키; 생성된 스크립트가 외래 키 제약을 작성한다. 선언 참조 무결성이 종속성 관계를 설정할 때 스크립팅을 지시. 스크립팅이 SQL 서버 테이블을 참조할 때만 적용.

[0120] DRI-비집중화; 생성된 스크립트가 비집중화 인덱스를 작성한다. 선언 참조 무결성이 종속성 관계를 설정할 때 스크립팅을 지시. 스크립팅이 SQL 서버 테이블을 참조 할 때만 적용.

[0121] DRI-1차 키; 생성된 스크립트가 PRIMARY KEY 제약을 작성한다. 선언 참조 무결성이 종속성 관계를 설정할 때 스크립팅을 지시. 스크립팅이 SQL 서버 테이블을 참조 할 때만 적용.

[0122] DRI-유일 키; 생성된 스크립트가 유일 키를 사용하여 정의된 후보 키를 작성한다. 선언 참조 무결성이 종속성 관계를 설정할 때 스크립팅을 지시. 스크립팅이 SQL 서버 테이블을 참조할 때만 적용.

[0123] DRI 인덱스; 선언 참조 무결성을 실현하기 위해 유일 인덱스를 사용하여 PRIMARY KEY 제약을 스크립트. 스크립팅이 SQL 서버 테이블을 참조할 때만 적용.

[0124] DRI 위드 노 체크; DRI 체크, 또는 DRI 외래 키를 사용할 때, 생성된스크립트는 WITH NO CHECK 절 최적화 제약 작성을 포함한다. 스크립팅이 SQL 서버 테이블을 참조할때만 적용.

[0125] 고유성 없음; 생성된 트랜잭트-SQL문장은 고유성(identity property), 시드(seed) 및 증분의 정의를 포함하지않는다. 스크립팅이 SQL 서버 테이블을 참조할 때만 적용.

[0126] 비집중화 인덱스; 비집중화 인덱스를 정의하는 트랜잭트-SQL을 생성. 스크립팅이 SQL 서버 테이블을 참조할 때만 적용.

[0127] 오브젝트 사용권한; 데이터베이스 오브젝트를 스크립팅할 때 트랜잭트-SQL 특권 정의 문장을 포함.

[0128] 트리거; 트리거를 정의하는 트랜잭트-SQL을 생성. 스크립팅이 SQL 서버 테이블을 참조할 때만 적용.

[0129] 베이스 타입에 대한 사용자 타입; 사용자-정의 데이터 타입들의 명세(specificaton)를 적절한 SQL 서버 베이스 데이터 타입으로 변환. 스크립팅이 SQL 서버 테이블을 참조할 때만 적용.

[0130] 파일 그룹 없음; 커맨드 배치는 파일그룹 사용을 지시하는 'ON<filegroup>'절을 포함하지 않음.

[0131] **미셀레니어스(Miscellaneous)**

[0132] 스크립터 옵션 설명

[0133] 시스템 오브텍트 허용(불리언(Boolean); 시스템 오브텍트의 스크립팅을 허용. 특정되지 않으면 시스템 오브젝트는 필터링 아웃됨.

[0134] 에이전트 경보 기능; SQL 서버 에이전트 서비스 기능 및 경보를 작성하 는 트랜잭트-SQL 스크립트를 생성.

[0135] 에이전트 통지; 스크립팅이 경보를 발생할 때, 경보에 대한 스크립트 작성 통지를 생성.

[0136] ANSI 패딩(Padding); 커맨드 배치는 생성된 스크립트내의 CREAT TABLE 선언 이전 및 이후에 트랜잭트-SQL 선언 SET ANSI PADDING ON 및 SET ANSI PADDING OFF 선언을 포함 함.

[0137] 인덱스인 경우는 없음; 커맨드 배치는 CREATE STATISTICS 선언을 포함하지 않음.

[0138] NoTablePartitioningSchemes; 커맨드 배치는 테이블 오브젝트에 대한 파티션 스킴을 포함하지 않음.

[0139] NoIndexPartitioningSchemes; 커맨드 배치는 인덱스 오브젝트에 대한 파티션 스킴을 포함하지 않음.

[0140] 어셈블리 없음; 커맨드 배치는 어셈블리를 포함하지 않는다.

[0141] 뷰 컬럼 없음; 뷰 오브젝트에 대한 특정 컬럼을 스크립트하지 않는다. 뷰 컬럼은 이들을 구체적으로 지정함으로써, 또는 선택 문장에 의해 정의되는 바와 같이, 뷰 작성 시에 기록된다.

[0142] 데이터베이스 문맥 포함; 스크립트의 헤더 내에 USE[database] 문장을 추가. [database]는 스크립트되는 오브젝트를 포함하는 데이터베이스 이름이다.

발명의 효과

[0143] 상술된 바와 같이, 본 발명의 예시적인 실시예는 여러가지 컴퓨팅 장치 및 네트워크 아키텍처와 관련하여 설명되었지만, 자동화 스크립터를 실현하기에 바람직한 소정의 컴퓨팅 장치 또는 시스템에 하위 개념이 적용될 수 있다. 그러므로, 본 발명의 방법 및 시스템은 여러가지 애플리케이션 및 장치에 적용될 수 있다. 예시적인 프로그래밍 언어, 이름 및 예들이 여러가지 선택사항을 대표하여 여기에서 선택되었지만, 이들 언어, 이름 및 예들은 제한적인 의도로 사용된 것은 아니다. 본 분야에 숙련된 기술자라면 본 발명에 의해 달성된 것과 동일하거나, 유사하거나, 대등한 시스템 및 방법을 달성하는 오브젝트 코드를 제공하는 여러가지 방식이 있다는 것을 이해할 수 있을 것이다.

[0144] 여기에 설명된 여러가지 기술은 하드웨어 또는 소프트웨어, 또는 적절한 경우에 이 둘의 조합과 관련하여 실현될 수 있다. 그러므로, 본 발명의 방법 및 장치, 또는 소정의 실시양상 또는 그 일부는 플로피 디스크, CD-ROM, 하드 드라이브, 또는 소정의 다른 기계 판독가능 저장 매체와 같은 유형의 매체 내에 구현된 프로그램 코드(즉, 명령어)의 형태를 취할 수 있는데, 프로그램 코드가 컴퓨터와 같은 기계로 로드되어 기계에 의해 실행될 때, 그 기계는 본 발명을 실시하는 장치가 된다. 프로그램가능한 컴퓨터 상에서의 프로그램 코드 실행의 경우에, 컴퓨팅 장치는 일반적으로, 프로세서, 프로세서에 의해 판독가능한 저장 매체(휘발성 및 비휘발성 메모리 및/또는 저장 소자를 포함), 최소한 하나의 입력 장치 및 최소한 하나의 출력 장치를 포함할 수 있다. 예를 들어, 데이터 프로세싱 API 등을 사용하여 본 발명의 신호 처리 서비스를 이용할 수 있는 하나 이상의 프로그램은 컴퓨터와 통신하기 위해 하이 레벨 절차 또는 객체 지향 프로그래밍 언어로 실현되는 것이 바람직하다. 그러나, 프로그램(들)은 원한다면 어셈블리어 또는 기계어로 실현될 수 있다. 어떤 경우에, 언어는 컴파일형 또는 해석형 언어일 수 있고, 하드웨어 구현물과 결합될 수 있다.

[0145] 본 발명의 방법 및 장치는 또한 전선이나 케이블선, 또는 광섬유와 같은 소정의 송신 매체를 통하거나, 또는 소정의 다른 형태의 송신을 통해 송신되는 프로그램 코드 형태로 구현된 통신을 통해 실시될 수 있는데, 프로그램 코드가 EPROM과 같은 기계에 수신되어 로드되고, 기계에 의해 실행될 때, 예시적인 실시예에서 상술된 것과 같은 신호 처리 능력을 갖는 수신 기계, 또는 게이트 어레이, 프로그램가능 로직 디바이스(PLD), 클라이언트 컴퓨터, 비디오 레코더 등은 본 발명을 실시하는 장치가 된다. 범용 프로세서 상에서 실현될 때, 프로그램 코드는 프로세서와 결합하여, 상술된 발명의 기능을 불러내는 동작을 하는 유일한 장치를 제공한다. 부수적으로, 본 발명과 관련하여 사용된 소정의 저장 기술은 항상 하드웨어와 소프트웨어의 결합으로 이루어질 수 있다.

[0146] 본 발명은 여러 도면들의 양호한 실시예와 관련하여 설명되었지만, 기타 유사한 실시예들이 사용될 수 있고, 또는 본 발명을 벗어나지 않고 본 발명과 동일한 기능을 실행하기 위해 상술된 실시예에 변경 및 추가가 행해질 수 있다는 것을 이해하기 바란다. 더욱이, 특히 무선 네트워크 장치의 수가 계속 늘어남에 따라, 핸드헬드 장치 운영 체계 및 기타 애플리케이션 특정 운영 체계를 포함하는 여러가지 컴퓨터 플랫폼이 고려될 수 있다는 점이 강조되어야 된다. 그러므로, 본 발명은 소정의 단일의 실시예에 한정되지 않아야 되고, 첨부된 청구범위에 따른 범위 내에서 구성되어야 된다.

도면의 간단한 설명

[0001] 도 1은 본 발명의 실시양상들이 실현될 수 있는 예시적인 컴퓨팅 환경을 도시한 블록도.

[0002] 도 2는 본 발명의 실시양상들이 실현될 수 있는 예시적인 전체 흐름도를 도시한 도면.

[0003] 도 3은 본 발명의 제1 모듈 또는 페이즈(phase)에 응용가능한 예시적인 흐름도를 도시한 도면.

[0004] 도 4는 본 발명의 제2 모듈 또는 페이즈에 응용가능한 예시적인 흐름도를 도시한 도면.

[0005] 도 5는 본 발명의 제3 모듈 또는 페이즈에 응용가능한 예시적인 흐름도를 도시한 도면.

[0006] 도 6은 본 발명의 한 실현예의 예시적인 정적 구조도를 도시한 도면.

[0007] <도면의 주요 부분에 대한 부호의 설명>

[0008] 110 : 컴퓨터 시스템

[0009] 120 : 프로세싱 유닛

[0010] 130 : 시스템 메모리

[0011] 210 : 종속성 발견 메카니즘

[0012] 212 : 종속성 트리

[0013] 220 : 리스트 생성 메카니즘

[0014] 240 : 스크립트 생성 메카니즘

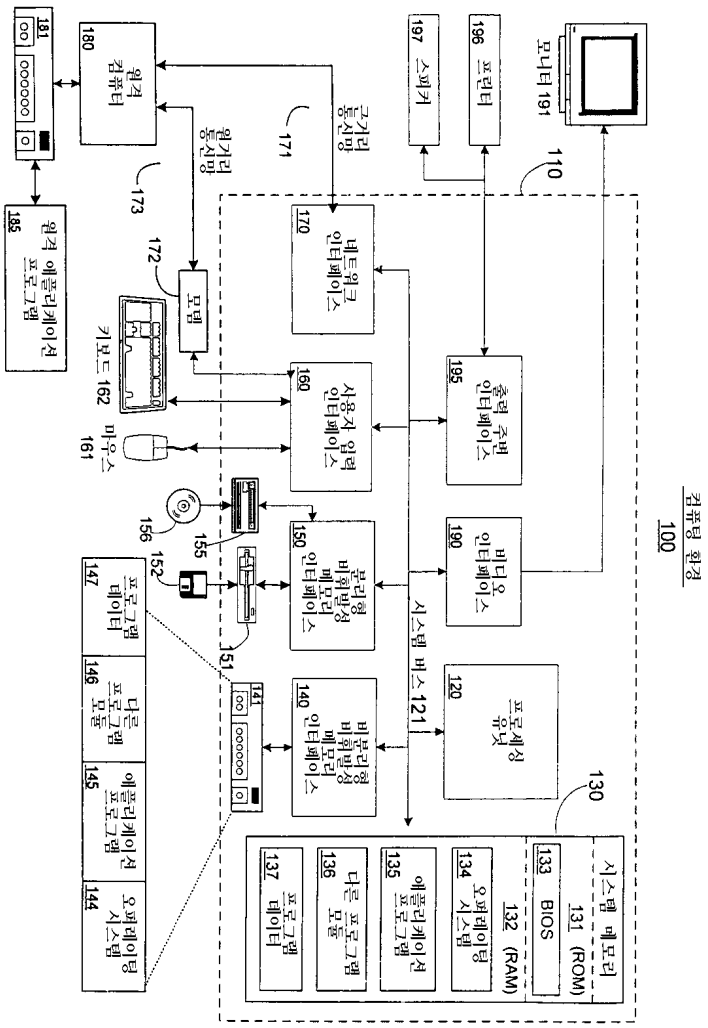
[0015] 260 : 사용자 프로그램

[0016] 270 : 서버 또는 시스템 카탈로그

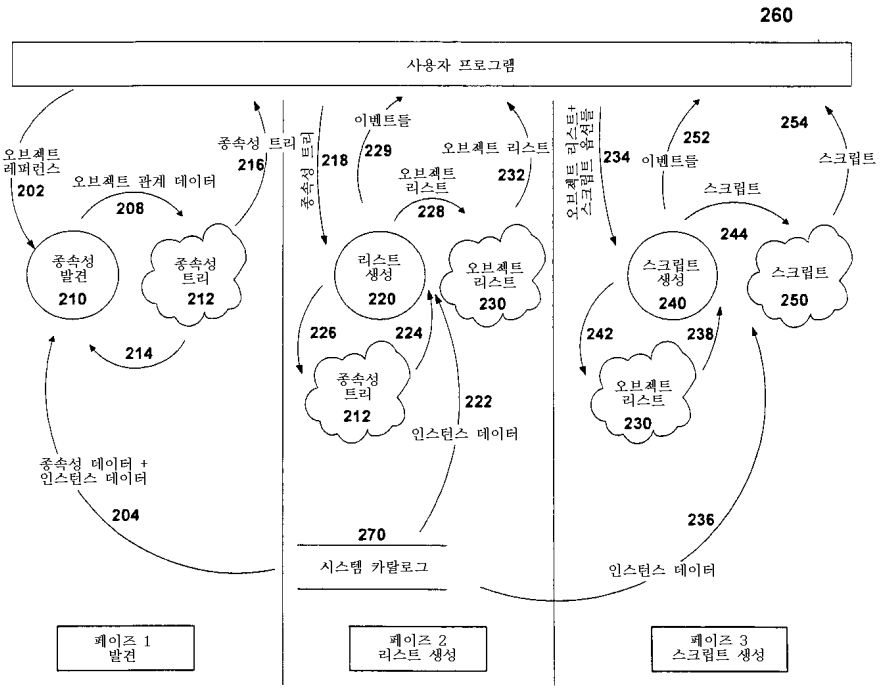
[0017] 610 : 종속성 워커

도면

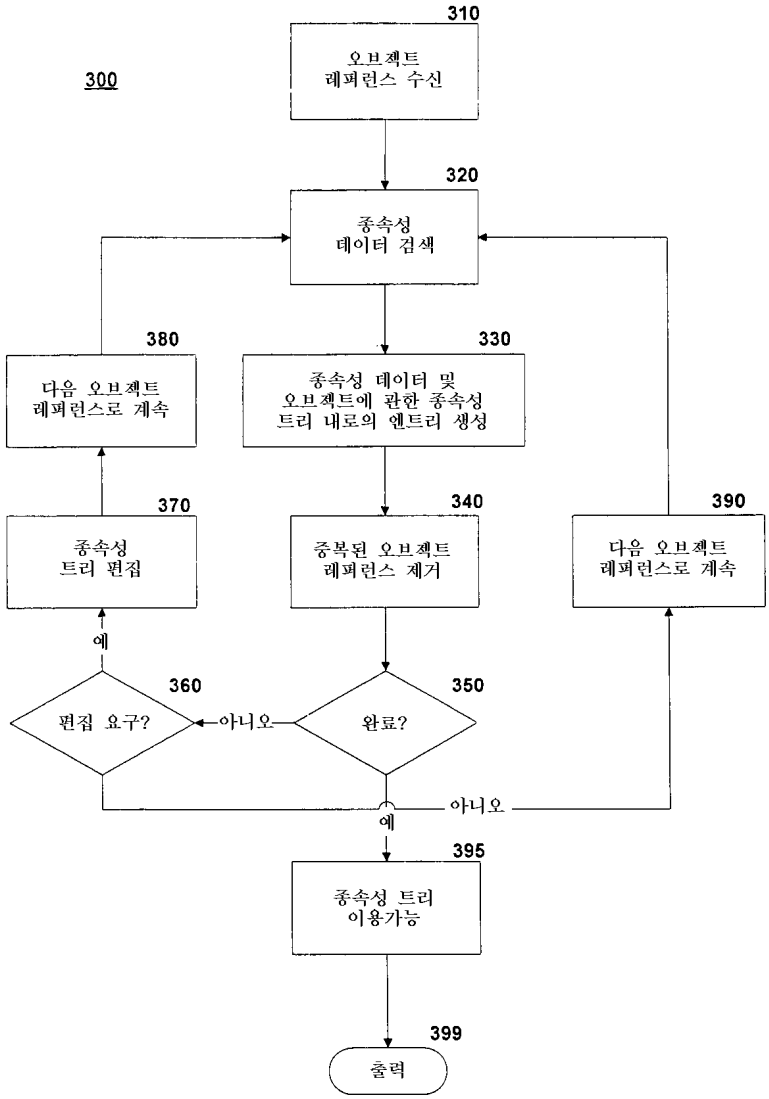
도면1



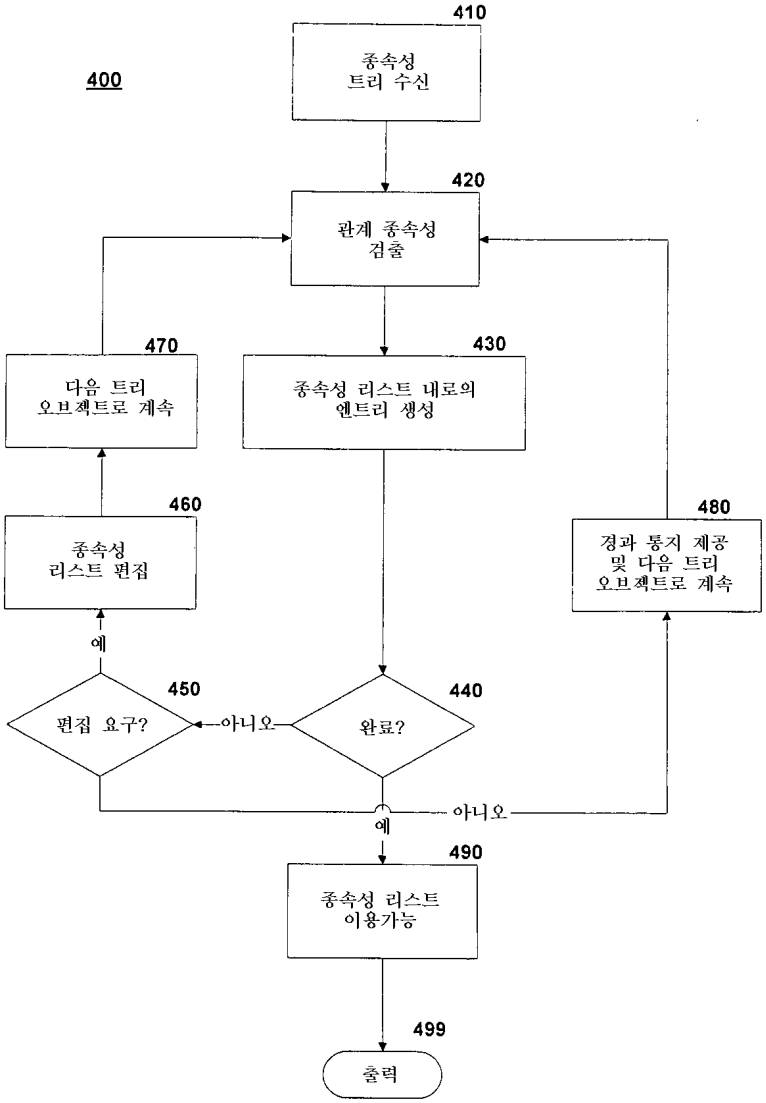
도면2


200
260
사용자 프로그램
오브젝트 레퍼런스
202
오브젝트 관계 데이터
208
종속성 트리
216
종속성 발견
210
종속성 트리
212
214
종속성 데이터 + 인스턴스 데이터
204
이벤트들
229
종속성 트리
218
오브젝트 리스트
232
오브젝트 리스트
228
리스트 생성
220
오브젝트 리스트
230
226
224
종속성 트리
212
222
인스턴스 데이터
270
시스템 카탈로그
오브젝트 리스트+ 스크립트 옵션들
234
이벤트들
252
254
스크립트
스크립트
244
스크립트 생성
240
스크립트
250
242
238
오브젝트 리스트
230
236
인스턴스 데이터
페이즈 1 발견
페이즈 2 리스트 생성
페이즈 3 스크립트 생성

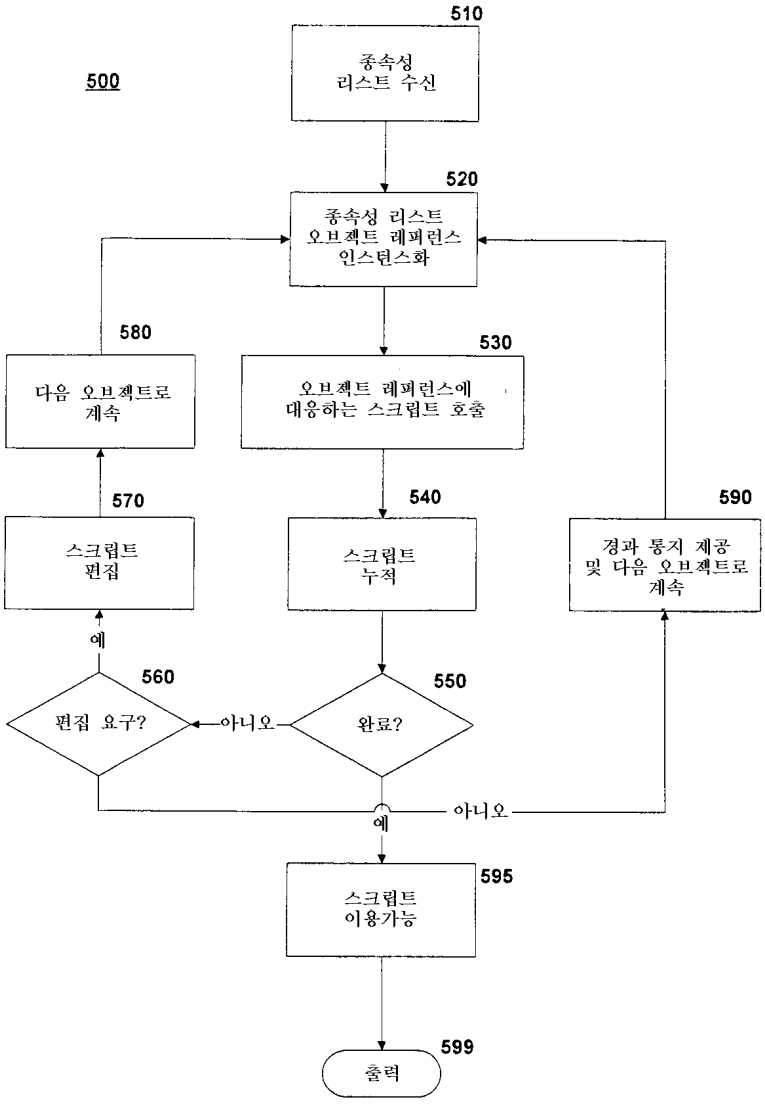
도면3


300
310
오브젝트 레퍼런스 수신
320
종속성 데이터 검색
330
종속성 데이터 및 오브젝트에 관한 종속성 트리 내로의 엔트리 생성
340
중복된 오브젝트 레퍼런스 제거
350
완료?
아니오
360
편집 요구?
예
370
종속성 트리 편집
380
다음 오브젝트 레퍼런스로 계속
390
다음 오브젝트 레퍼런스로 계속
아니오
예
395
종속성 트리 이용가능
399
출력

도면4

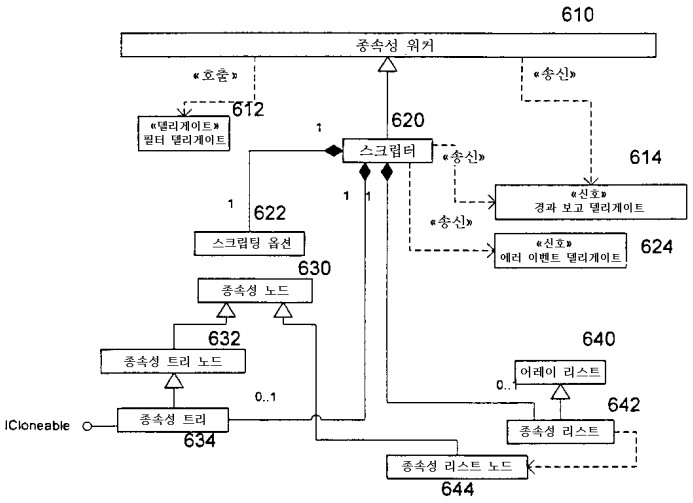

400
410
종속성
트리 수신
420
관계 종속성
검출
430
종속성 리스트 내로의
엔트리 생성
440
완료?
450
편집 요구?
460
종속성
리스트 편집
470
다음 트리
오브젝트로 계속
480
경과 통지 제공
및 다음 트리
오브젝트로 계속
490
종속성 리스트
이용가능
499
출력
아니오
예
아니오
예

도면5


500
510
종속성
리스트 수신
520
종속성 리스트
오브젝트 레퍼런스
인스턴스화
530
오브젝트 레퍼런스에
대응하는 스크립트 호출
540
스크립트
누적
550
완료?
아니오
560
편집 요구?
예
570
스크립트
편집
580
다음 오브젝트로
계속
590
결과 통지 제공
및 다음 오브젝트로
계속
아니오
예
595
스크립트
이용가능
599
출력

도면6

600


610
종속성 워커
«호출»
«송신»
612
«델리게이트»
필터 델리게이트
620
스크럼터
«송신»
614
«신호»
경과 보고 델리게이트
1
1
1
1
622
«송신»
스크럼팅 옵션
«신호»
에러 이벤트 델리게이트
624
630
종속성 노드
632
640
종속성 트리 노드
어레이 리스트
0..1
0..
642
ICloneable
종속성 트리
종속성 리스트
634
종속성 리스트 노드
644