(19) 대한민국특허청(KR)
(12) 공개특허공보(A)

(11) 공개번호 10-2013-0068601
(43) 공개일자 2013년06월26일

(51) 국제특허분류(Int. Cl.)
G06F 15/16 (2006.01) *G06F 9/44* (2006.01)
G06F 17/30 (2006.01)
(21) 출원번호 10-2011-0135891
(22) 출원일자 2011년12월15일
심사청구일자 없음

(71) 출원인
한국전자통신연구원
대전광역시 유성구 가정로 218 (가정동)
(72) 발명자
이지현
대전광역시 유성구 관평동 운암네오미아 605동 1001호
강성주
대전광역시 유성구 가정로 270, ETRI 기숙사 2동 223호 (가정동)
허성진
대전광역시 유성구 하기동 송림마을 2단지 204동 1902호
(74) 대리인
특허법인우인

전체 청구항 수 : 총 1 항

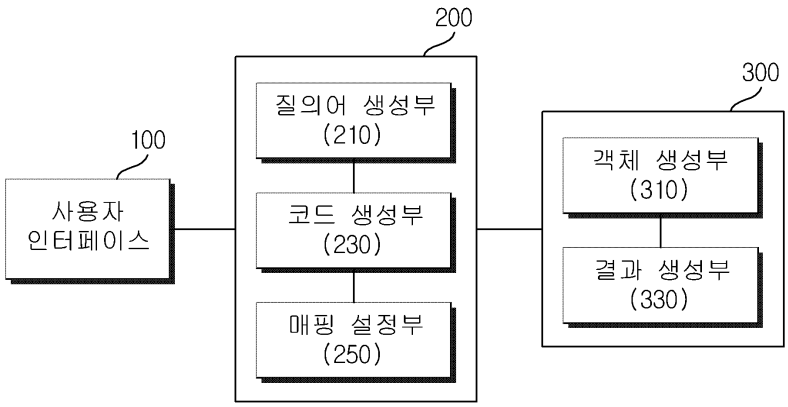
(54) 발명의 명칭 **질의 기반 소프트웨어 논리의 동적 변경 및 실행을 위한 멀티테넌트 지원 장치 및 방법**

(57) 요 약

본 발명은 사용자의 요청을 입력받으며, 상기 사용자의 설정 정보를 추출하는 사용자 인터페이스, 상기 입력된 요청에 기초하여 질의어가 생성되며, 상기 생성된 질의어가 상기 사용자의 설정 정보에 매칭되는 소프트웨어 논리로 변환되며, 상기 변환된 소프트웨어 논리가 메타데이터에 매핑(mapping)되는 로직 처리부 및 상기 변환된 소프트웨어 논리 및 상기 매핑된 메타데이터를 이용하여 상기 사용자의 요청에 대응되는 데이터를 제공하는 데이터 처리부를 포함하는 것을 특징으로 하는 멀티테넌트 지원 장치를 제공한다.

본 발명에 따르면, 소프트웨어 논리를 동적으로 생성할 수 있으며, 생성된 정보를 저장했다가 실행을 요청하는 경우 동적으로 로딩하여 객체를 만들고 실행될 수 있도록 하는 기반을 제공하는 효과가 있다.

대 표 도 - 도1



이 발명을 지원한 국가연구개발사업

과제고유번호	KI002125
부처명	지식경제부
연구사업명	정보통신산업원천기술개발사업
연구과제명	중소기업 SW 서비스를 위한 SaaS 플랫폼 개발
주관기관	한국전자통신연구원
연구기간	2009.03.01 ~ 2012.02.29

특허청구의 범위

청구항 1

사용자의 요청을 입력받으며, 상기 사용자의 설정 정보를 추출하는 사용자 인터페이스;

상기 입력된 요청에 기초하여 질의어를 생성하고, 상기 생성된 질의어가 상기 사용자의 설정 정보에 매칭되는 소프트웨어 논리로 변환되며, 상기 변환된 소프트웨어 논리가 메타데이터에 매핑(mapping)되는 로직 처리부; 및

상기 변환된 소프트웨어 논리 및 상기 매핑된 메타데이터를 이용하여 상기 사용자의 요청에 대응되는 데이터를 제공하는 데이터 처리부를 포함하는 것을 특징으로 하는 멀티테넌트 지원 장치.

명 세 서

기 술 분 야

[0001] 본 발명은 질의 기반 소프트웨어 논리 처리 장치 및 방법에 관한 것으로, 보다 상세하게는 비지니스 처리를 위해 질의 기반으로 동적으로 소프트웨어 논리를 생성하고 실행할 수 있도록 처리하기 위한 장치 및 방법에 관한 것이다.

배 경 기 술

[0002] 소프트웨어 시장의 주요 트렌드는 클라우드 컴퓨팅, SaaS, 유틸리티 컴퓨팅, SOA, 웹 2.0, RIA 등을 들 수 있는 바, 이들 중 소프트웨어를 서비스로 발전시킨 SaaS에 대하여 기술적 관심이 높아지고 있다.

[0003] SaaS는 SOA, 웹 2.0 또는 RIA 와 같은 소프트웨어를 구현하고 사용, 관리하는 방식의 변화가 아닌 소프트웨어 유통 방식의 변화를 설명하는 개념으로서, 사용자가 필요한 소프트웨어를 인터넷을 통해 온라인 서비스로 이용할 수 있도록 하는 최신의 소프트웨어 배포 모델로 정의될 수 있다. 따라서, 사용자는 인터넷을 통해 소프트웨어를 사용하고 그에 대한 비용만 지불하는방식으로 복잡한 소프트웨어 및 하드웨어의 관리라는 부담에서 벗어날 수 있으며, SaaS 비즈니스 모델은 고객의 초기 투자비용이 거의 없고 시스템 관리 필요성도 없어진다. 소프트웨어 공급자의 입장에서는 사용자의 커스터마이징을 메타데이터를 활용해서 지원하며 사용자들 또는 사용자 그룹으로 표현되는 테넌트들을 하나의 소프트웨어 인스턴스로 지원(single instance multitenant)한다는 점에서 차별성을 갖는다.

[0004] SaaS에서 테넌트(tenant)란 여러 사용자가 속한 하나의 조직으로 회사, 기관, 단체 등 사용자 집단에 관한 것이며, 멀티 테넌시(multitenancy)란 개별 사용자(테넌트)의 결정에 의하여 설정되고 표현되는 기능으로서, 여러 사용자들을 하나의 어플리케이션 인스턴스로 지원하여야 함을 의미한다.

[0005] 또한, 로직(Logic)이란 하드웨어 또는 소프트웨어에 의하여 수행되는 연산들의 순서를 가리키는 것으로, 하드웨어 논리와 소프트웨어 논리로 구별된다. 하드웨어 논리는 부울(Bool) 논리의 규칙을 말하는 것이며, 소프트웨어 논리는 프로그래머가 작성한 명령어 형식을 말하는 것으로서, 소프트웨어의 논리 중 운영체계나 네트웍 하부구조가 아닌 비지니스 트랜잭션에 관련된 것들은 비지니스 논리 또는 비지니스 로직으로 정의된다.

발명의 내용

해결하려는 과제

[0006] 종래 기술의 경우 네트워크를 통하여 소프트웨어 서비스를 사용하는 테넌트가 개별적인 소프트웨어 논리를 동적으로 생성하고 필요한 코드를 작성하여 실시간으로 반영하는데 문제점이 있었다.

과제의 해결 수단

[0007] 상술한 기술적 과제를 해결하기 위한 본 발명의 일 실시예는 사용자의 요청을 입력받으며, 상기 사용자의 설정 정보를 추출하는 사용자 인터페이스; 상기 입력된 요청에 기초하여 질의어가 생성되며, 상기 생성된 질의어가 상기 사용자의 설정 정보에 매칭되는 소프트웨어 논리로 변환되며, 상기 변환된 소프트웨어 논리가 메타데이터에 매핑(mapping)되는 로직 처리부; 및 상기 변환된 소프트웨어 논리 및 상기 매핑된 메타데이터를 이용하여 상

기 사용자의 요청에 대응되는 데이터를 제공하는 데이터 처리부를 포함하는 것을 특징으로 하는 멀티테넌트 지원 장치를 제공하는 것을 특징으로 할 수 있다.

[0008] 또한, 상기 로직 처리부는 상기 입력된 사용자의 요청에 따라 질의어를 생성하는 질의어 생성부; 상기 생성된 질의어에 대하여 질의어 처리를 위하여 미리 설정된 사용자 정의 코드를 추가하는 코드 생성부; 및 상기 생성된 코드를 메타데이터와 매핑되도록 설정하는 매핑 설정부를 포함하는 것을 특징으로 할 수 있다.

[0009] 바람직하게는, 상기 질의어는 구조화 질의어(Structured Query Language: SQL)인 것을 특징으로 할 수 있다.

[0010] 또한, 상기 데이터 처리부는 상기 입력된 요청에 매칭되는 데이터를 로딩하여 객체를 생성하는 객체 생성부; 및 상기 생성된 객체에 매칭되는 상기 설정된 로직 및 상기 메타데이터를 이용하여 처리 결과를 생성하는 결과 생성부를 포함하는 것을 특징으로 할 수 있다.

[0011] 또한, 상기 사용자 인터페이스를 시각화 처리하는 시각화부를 더 포함하는 것을 특징으로 할 수 있다.

[0012] 또한, 상기 사용자 인터페이스의 속성을 정의하는 사용자 인터페이스(UI) 설정부를 더 포함하는 것을 특징으로 할 수 있다.

[0013] 상술한 기술적 과제를 해결하기 위한 본 발명의 다른 실시예는 사용자의 요청을 입력받아, 상기 입력된 요청에 대응하는 처리 결과를 생성하기 위한 로직을 설정하는 단계; 상기 설정된 로직에 기초하여 메타데이터를 매핑하는 단계; 및 상기 매핑된 메타데이터를 저장하는 매핑 설정 단계를 포함하는 로직 처리 방법을 제공하는 것을 특징으로 할 수 있다.

[0014] 또한, 상기 요청에 매칭되는 데이터를 로딩하여 객체를 생성하는 단계; 및 상기 생성된 객체에 대하여 상기 설정된 로직 및 상기 메타데이터를 이용하여 처리 결과를 생성하는 결과 생성 단계를 더 포함하는 것을 특징으로 할 수 있다.

[0015] 상술한 기술적 과제를 해결하기 위한 본 발명의 다른 실시예는 사용자의 요청을 입력받아, 상기 입력된 요청에 대응하는 처리 결과를 생성하기 위한 로직을 설정하는 단계; 상기 설정된 로직에 기초하여 메타데이터를 매핑하는 단계; 상기 매핑된 메타데이터를 저장하는 매핑 설정 단계; 상기 요청에 매칭되는 데이터를 로딩하여 객체를 생성하는 단계; 및 상기 생성된 객체에 대하여 상기 설정된 로직 및 상기 메타데이터를 이용하여 처리 결과를 생성하는 결과 생성 단계를 포함하는 로직 처리 방법을 컴퓨터에서 실행가능하도록 기록한 것을 특징으로 하는 컴퓨터 판독 가능한 기록 매체를 제공하는 것을 특징으로 할 수 있다.

발명의 효과

[0016] 본 발명에 의하면, 소프트웨어 논리를 동적으로 생성할 수 있으며, 생성된 정보를 저장했다가 실행을 요청하는 경우 동적으로 로딩하여 객체를 만들고 실행될 수 있도록 하는 기반을 제공하는 효과가 있다.

도면의 간단한 설명

[0017] 도 1은 본 발명의 일 실시예에 따른 멀티테넌트 지원 장치를 설명하기 위한 블록도이다.

도 2는 본 발명의 일 실시예에 따른 멀티테넌트 지원 장치의 로직 처리부를 설명하기 위한 참고도이다.

도 3은 본 발명의 일 실시예에 따른 멀티테넌트 지원 장치의 데이터 처리부를 설명하기 위한 참고도이다.

도 4는 본 발명의 일 실시예에 따른 멀티테넌트 지원 장치의 SQL 쿼리를 설정하는 과정을 설명하기 위한 참고도이다.

도 5 및 도 6은 본 발명의 일 실시예에 따른 멀티테넌트 지원 장치의 로직 처리부에서 사용하기 위한 사용자 정의 코드를 설정을 설명하기 위한 참고도이다.

도 7 및 도 8은 본 발명의 일 실시예에 따른 소프트웨어 처리 방법을 설명하기 위한 순서도이다.

발명을 실시하기 위한 구체적인 내용

[0018] 이하에서는 본 발명의 일부 실시예를 첨부된 도면들을 참조하여 상세히 설명한다. 아울러 본 발명을 설명함에 있어 관련된 공지 구성 또는 기능에 대한 구체적인 설명이 본 발명의 요지를 흐릴 수 있다고 판단되는 경우에는 그 상세한 설명을 생략한다.

[0019] 이하의 실시예들은 본 발명의 구성요소들과 특징들을 소정 형태로 결합한 것들이다. 각 구성요소 또는 특징은 별도의 명시적 언급이 없는 한 선택적인 것으로 고려될 수 있다. 각 구성요소 또는 특징은 다른 구성요소나 특징과 결합하지 않은 형태로 실시될 수 있다. 또한, 일부 구성요소들 및/또는 특징들을 결합하여 본 발명의 실시예를 구성할 수도 있다. 본 발명의 실시예들에서 설명되는 동작들의 순서는 변경될 수 있다. 어느 실시예의 일부 구성이나 특징은 다른 실시예에 포함될 수 있고, 또는 다른 실시예의 대응하는 구성 또는 특징과 교체될 수 있다.

[0020] 본 발명의 실시예들은 다양한 수단을 통해 구현될 수 있다. 예를 들어, 본 발명의 실시예들은 하드웨어, 펌웨어(firmware), 소프트웨어 또는 그것들의 결합 등에 의해 구현될 수 있다.

[0021] 하드웨어에 의한 구현의 경우, 본 발명의 실시예들에 따른 방법은 하나 또는 그 이상의 ASICs(application specific integrated circuits), DSPs(digital signal processors), DSPDs(digital signal processing devices), PLDs(programmable logic devices), FPGAs(field programmable gate arrays), 프로세서, 콘트롤러, 마이크로 콘트롤러, 마이크로 프로세서 등에 의해 구현될 수 있다.

[0022] 펌웨어나 소프트웨어에 의한 구현의 경우, 본 발명의 실시예들에 따른 방법은 이상에서 설명된 기능 또는 동작들을 수행하는 모듈, 절차 또는 함수 등의 형태로 구현될 수 있다. 소프트웨어 코드는 메모리 유닛에 저장되어 프로세서에 의해 구동될 수 있다. 상기 메모리 유닛은 상기 프로세서 내부 또는 외부에 위치하여, 이미 공지된 다양한 수단에 의해 상기 프로세서와 데이터를 주고 받을 수 있다.

[0023] 이하의 설명에서 사용되는 특정(特定) 용어들은 본 발명의 이해를 돕기 위해서 제공된 것이며, 이러한 특정 용어의 사용은 본 발명의 기술적 사상을 벗어나지 않는 범위에서 다른 형태로 변경될 수 있다.

[0024] 도 1을 참조하여 본 발명의 일 실시예에 따른 멀티테넌트 지원 장치를 설명한다. 멀티테넌트 지원 장치는 사용자 인터페이스(100), 로직 처리부(200), 데이터 처리부(300)을 포함한다.

[0025] 사용자 인터페이스(100)는 사용자의 요청을 입력받는다. 본 발명의 일 실시예에 따르면 사용자로부터 소프트웨어 서비스 비지니스 로직 생성이나 비지니스 로직 실행을 위한 요청을 입력받는다.

[0026] 로직 처리부(200)는 입력된 요청에 기초하여 질의어를 생성하고, 생성된 질의어를 변화하여 소프트웨어 논리를 생성한다. 로직 처리부는 질의어 생성부(210), 코드 생성부(230) 및 매핑 설정부(250)를 포함한다.

[0027] 질의어 생성부(210)는 사용자 인터페이스(100)을 이용하여 입력받은 사용자의 요청에 기초하여 질의어를 생성한다. 본 발명의 일 실시예에 따르면, 사용자 인터페이스를 통하여 사용자로부터 사용될 어플리케이션, 데이터베이스에서 사용될 필드, 데이터베이스 필드, 확장 필드, 고정 필드 또는 SQL 인수등을 입력받는다. 본 발명의 일 실시예에 따르면 질의어는 구조화 질의어(Structured Query Language)를 사용하는 것이 바람직하다.

[0028] 코드 생성부(230)는 질의어 생성부(210)에서 생성된 질의어를 기초로 질의어 처리를 위하여 파라미터, SQL 실행 함수 또는 미리 설정된 코드로 구성된 사용자 정의 코드를 추가한다. 예를 들어, 파라미터는 입출력 파라미터의 타입, 순서 등이 정의될 수 있으며, SQL 실행 함수는 테넌트별로 분리되어 사용하기 위한 함수를 포함한다. 사용자 정의 코드란 사용자로부터 데이터 베이스에 저장된 데이터를 처리하는 생성(Create), 읽기(Read), 갱신(Update) 또는 삭제(Delete) 기능을 수행할 수 있도록 설정된 코드를 의미하며, 본 발명의 일 실시예에 따르면 쿼리(Query)로 정의될 수 있다.

[0029] 코드 생성부(230)는 질의어를 기초로 로직 이름, 설명, 접근 URL, 입출력 파라미터 또는 로직 코드 등을 입력받아, 입력받은 데이터에 대하여 사용자로부터 새롭게 설정받을 수 있으며, 새롭게 설정받는 경우에는 별도의 팝업창 등을 출력하여 소프트웨어 논리 또는 비지니스 로직을 표현할 수 있다.

[0030] 매핑 설정부(250)는 생성된 코드를 메타데이터와 매핑되도록 설정한다. 본 발명의 일 실시예에 따르면, 데이터 처리 전 수행할 소프트웨어 논리 코드가 생성된 경우 미리 설정된 파라미터를 전달할 데이터 구조를 매핑한 후, 선택된 SQL에 SQL 처리 구문을 입력하고, 데이터 처리 결과를 가공할 소프트웨어 논리 코드를 입력하여 출력 파라미터가 처리 결과를 담을 데이터 구조의 파라미터로 매핑한다.

[0031] 데이터 처리부(300)는 생성된 소프트웨어 논리에 따른 처리 결과를 생성한다. 데이터 처리부(300)는 테넌트가 요청한 소프트웨어 논리를 수행하기 위하여 로직 처리부에 저장된 로직 코드와 SQL 처리 코드를 이용하여 처리한다. 본 발명의 일 실시예에 따르면, 데이터 처리부는 객체 생성부(310) 및 결과 생성부(330)를 포함한다.

[0032] 객체 생성부(310)는 입력된 요청에 매칭되는 데이터를 로딩하여 동적으로 객체를 생성한다. 데이터 저장부(미도시)에 저장되어 있는 소프트웨어 논리 코드와 데이터 처리 코드를 로딩한다. 객체 생성부(310)는 소프트웨어 논리 코드와 데이터 처리 코드를 합성하고 객체를 생성한다.

[0033] 결과 생성부(330)는 생성된 객체에 매칭되는 설정된 로직 및 메타 데이터를 이용하여 처리 결과를 생성한다. 결과 생성부(330)는 생성된 객체를 컴파일(Compile) 하고, 컴파일된 객체를 로딩하여 실행되도록 한다.

[0034] 본 발명에 따르면, 네트워크를 통하여 소프트웨어 서비스를 사용하는 개별 테넌트가 개별 데이터를 처리하기 위한 로직을 런타임에 동적으로 생성하고 필요한 코드 작성의 어려움을 해결할 수 있다.

[0035] 본 발명의 일 실시예에 따르면 시각화부(미도시)를 더 포함할 수 있으며, 사용자 인터페이스를 시각화 처리한다. 시각화부는 UI 메타데이터를 로직 처리부(200)로부터 수신받아, 수신된 UI 메타데이터를 렌더링(rendering)하여 사용자에게 제공한다. 본 발명의 일 실시예에 따르면, 멀티테넌트 지원 장치는 웹 페이지를 통하여 사용자에게 UI 메타데이터를 제공할 수 있다. UI 메타데이터란 사용자 인터페이스를 구성하는 객체, 버튼 또는 상자 등의 정보를 포함하고 있는 데이터를 의미한다.

[0036] 본 발명의 일 실시에에 다르면 사용자 인터페이스(UI) 설정부를 더 포함할 수 있으며, 사용자 인터페이스(UI) 설정부는 사용자 인터페이스의 속성을 정의한다. 본 발명의 일 실시예에 따르면 사용자 인터페이스(UI) 설정부는 UI 템플릿(UI Templete)을 이용하여 추가하고자 하는 UI 컴포넌트의 레이아웃, 이미지, 텍스트, 호출될 비지니스 로직 등의 UI 컴포넌트의 속성 정보를 설정할 수 있다. 예를 들어, UI 컴포넌트는 폼(form), 텍스트 필드(text field) 또는 그리드(grid) 패널 등이 포함된다. 사용자가 UI를 설정하는 경우, 설정된 UI 컴포넌트 정보는 UI 메타데이터 저장부(미도시)에 저장된다.

[0037] 도 2를 참조하여 본 발명의 일 실시예에 따른 로직 처리부에서 소프트웨어 논리를 생성하는 과정을 설명한다. 예를 들어 소프트웨어 논리 중 비지니스 로직을 생성하며, 사용자 정의 로직 편집기로 구현된 로직 처리부를 설명한다.

[0038] 사용자가 로직 개발을 요청하면(400) 사용자 정의 로직 편집기는 SQL 쿼리를 생성한다(401). 비지니스 로직 처리에 필요한 입력 파라미터의 타입, 순서, 매핑값 또는 출력 파라미터의 타입, 순서, 매핑값을 설정한다(402). SQL 쿼리는 일반적인 CRUD 기능으로 정의되나 테넌트별로 분리되어 데이터베이스에서 저장되고 추출될 수 있도록 메타데이터 관리기 실행 함수를 추가하여 설정될 수 있다(403). 또한, Java 등을 이용하여 사용자 정의한 코드를 추가될 수 있다(404). 새롭게 추가된 비지니스 로직과 같은 소프트웨어 논리 코드는 외부의 데이터 저장부에 등록되도록 구현할 수 있으며, 새롭게 생성된 비지니스 로직코드와 매핑 정보는 메타데이터 저장소에 저장된다(406).

[0039] 도 3을 참조하여 본 발명의 일 실시예에 따른 데이터 처리부(300)의 동작을 설명한다. 예를 들어, 소프트웨어 논리 중 비지니스 로직에 대하여 처리하는 실시예로서 데이터 처리부는 서비스 컨트롤러 및 실행 엔진으로 구현되는 경우를 설명한다.

[0040] 브라우저를 통해 사용자로부터 비지니스 로직의 실행이 요청되면(701), 서비스 컨트롤러는 비지니스 로직을 클래스 화하기 위해 실행 엔진으로 요청을 전달한다(702). 요청을 받은 실행 엔진은 로직 메타 데이터 저장소로부터 SQL 쿼리문과 사용자 정의 로직 코드를 로딩하며(703), 동적으로 클래스를 생성한다(704). 생성된 클래스는 동적으로 컴파일되어 메모리에 로딩되고(705), 비지니스 로직 코드 내 로직 처리 함수를 호출하여 실행한다(706). 호출된 함수는 SQL 쿼리를 실행시키고 사용자 정의 코드에 기술된 대로 가공한 결과를 리턴한다(707). 실행 엔진은 비지니스 로직 결과를 얻어 웹 페이지로 생성하고(707), 실행 결과를 브라우저로 전달하여 실행 결과를 사용자에게 디스플레이 한다(708).

[0041] 도 4를 참조하여 본 발명의 일 실시예에 따라 SQL 쿼리를 설정하는 과정을 설명한다. 입력하는 SQL쿼리가 사용될 어플리케이션, 데이터베이스에서 사용될 필드, 데이터베이스 테이블에 필드를 추가하여 확장할 지 여부, SQL 처리 시 입력 파라미터로 사용될 SQL 인수와 같은 변수 명, SQL 구문을 입력할 수 있다. 예를 들어, SQL 이름으로 'SELECT ITEM', 관련 어플리케이션은 'CRM', 확장 필드 사용 여부에 대하여는 'NO'라고 설정하는 경우, 해당 필드와 관련된 SQL에 대하여 사용자로부터 편집받을 수 있다.

[0042] 도 5 및 도 6을 이용하여 본 발명의 일 실시예에 따른 로직 처리부에서 사용하기 위한 사용자 정의 코드를 설정하는 방법을 설명한다. 사용자 정의 코드는 예를 들면, 로직 이름, 설명, 접근 URL, 입출력 파라미터, 또는 로직 코드를 포함한다. 예를 들어, 사용자가 입출력 파라미터를 편집하고자 하는 경우 파라미터 이름, 설명, 데이

터 타입, 기본값 또는 null 허용 여부를 테이블 형태로 입력할 수 있다. 다른 예로서 로직 코드를 설정한다면, 데이터 처리 전 수행할 비지니스 로직 코드 입력 영역(810), 정의한 입력 파라미터들을 전달할 데이터 구조로 매핑된 정보를 디스플레이하는 영역(820), 데이터 처리 결과를 가공할 비즈니스 로직 코드 입력 영역(830), 정의한 출력 파라미터가 처리 결과를 담을 데이터 구조의 파라미터로 매핑된 정보를 보여주는 영역(840)으로 구성된다. 사용자가 SQL 처리 구문을 입력하는 경우에는 SQL 선택 버튼을 눌러 비지니스 로직 코드 입력 영역에 입력되도록 한다.

[0043] 도 7 및 도8을 참조하여 본 발명의 일 실시예에 따른 소프트웨어 처리 방법을 설명한다. 상술한 멀티테넌트 지원 장치와 동일한 내용은 상술한 내용으로 대체한다.

[0044] S110 단계는 사용자의 요청을 입력받아, 입력된 요청에 대응하는 처리 결과를 생성하기 위한 로직을 설정한다. 로직을 설정하는 단계는 사용자의 요청에 따라 질의어를 생성하고, 생성된 질의어에 대하여 미리 설정된 사용자 정의 코드를 추가함으로서 구현할 수 있다.

[0045] S120 단계는 설정된 로직에 기초하여 메타데이터를 매핑한다. 즉, 생성된 코드를 메타데이터와 매핑되도록 설정한다.

[0046] S130 단계는 매핑된 메타데이터를 저장한다. S120 단계에서 생성된 코드를 메타데이터와 매핑하고, 매핑된 메타데이터를 외부의 데이터베이스에 저장한다.

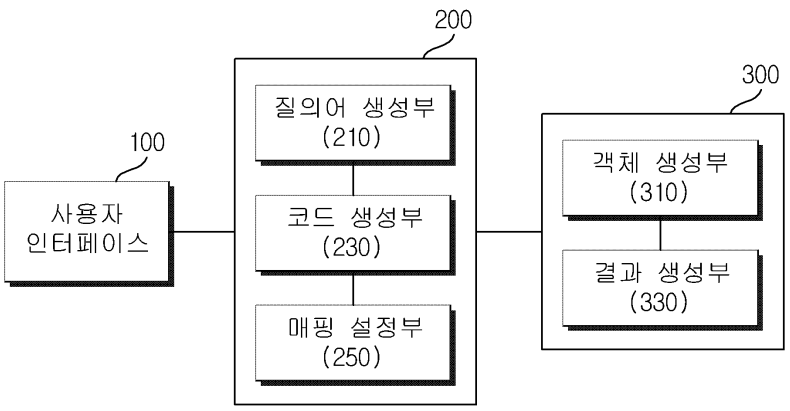
[0047] 본 발명의 일 실시예에 따르면 소프트웨어 처리 방법은 요청에 매칭되는 데이터를 로딩하여 객체를 생성하는 단계(S140), 생성된 객체에 대하여 설정된 로직 및 메타데이터를 이용하여 처리 결과를 생성하는 결과 생성 단계(S150)를 더 포함할 수 있다.

[0048] 본 발명에 의한 실시예들은 컴퓨터 프로그램으로 작성 가능하다. 이 컴퓨터 프로그램을 구성하는 코드들 및 코드 세그먼트들은 당해 분야의 컴퓨터 프로그래머에 의하여 용이하게 추론될 수 있다. 또한, 해당 컴퓨터 프로그램은 컴퓨터가 읽을 수 있는 정보저장매체(Computer Readable Media)에 저장되고, 컴퓨터에 의하여 읽혀지고 실행됨으로써 실시예를 구현한다. 정보저장매체는 자기 기록매체, 광 기록매체 및 캐리어 웨이브 매체를 포함한다.

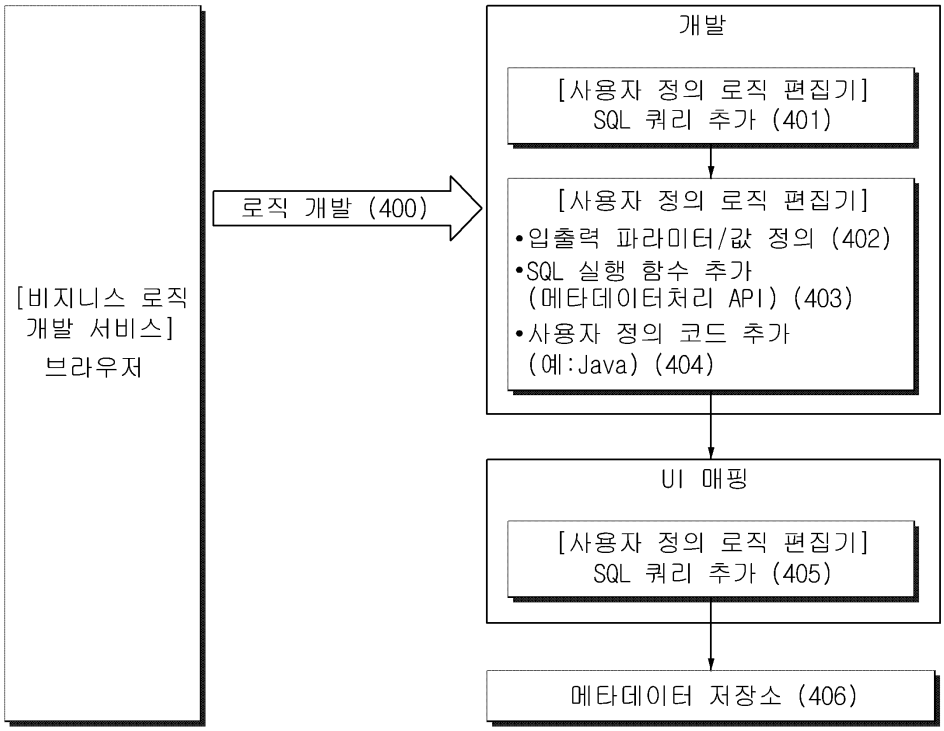
[0049] 이제까지 본 발명에 대하여 바람직한 실시예를 중심으로 살펴보았다. 본 발명이 속하는 기술 분야에서 통상의 지식을 가진 자는 본 발명의 본질적인 특성에서 벗어나지 않는 범위에서 변형된 형태로 본 발명을 구현할 수 있음을 이해할 것이다. 그러므로, 상기 개시된 실시예 들은 한정적인 관점이 아니라 설명적인 관점에서 고려되어야 한다. 본 발명의 범위는 전술한 설명이 아니라 특허청구범위에 나타나 있으며, 그와 동등한 범위 내에 있는 모든 차이점은 본 발명에 포함된 것으로 해석되어야 한다.

도면

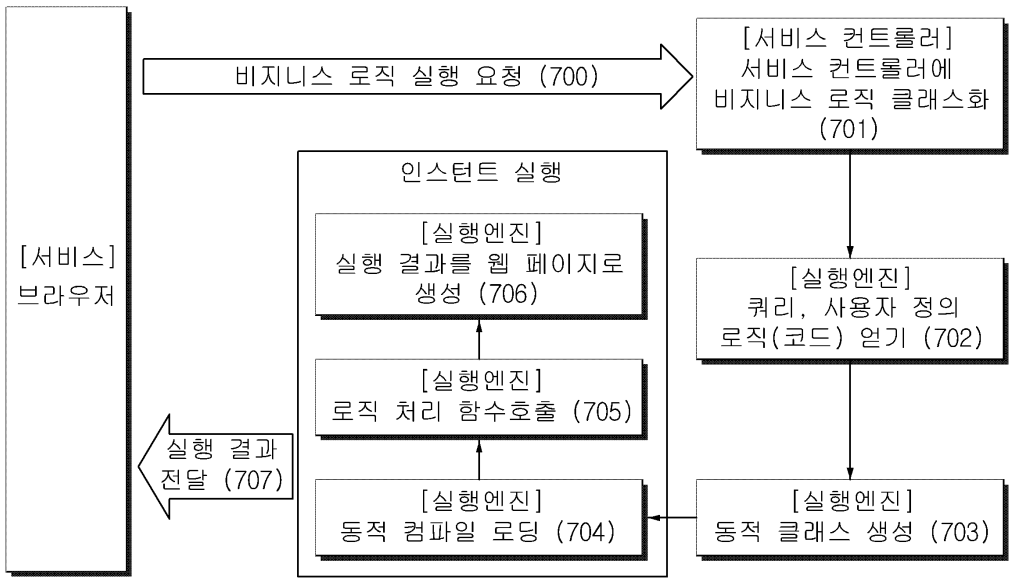
도면1



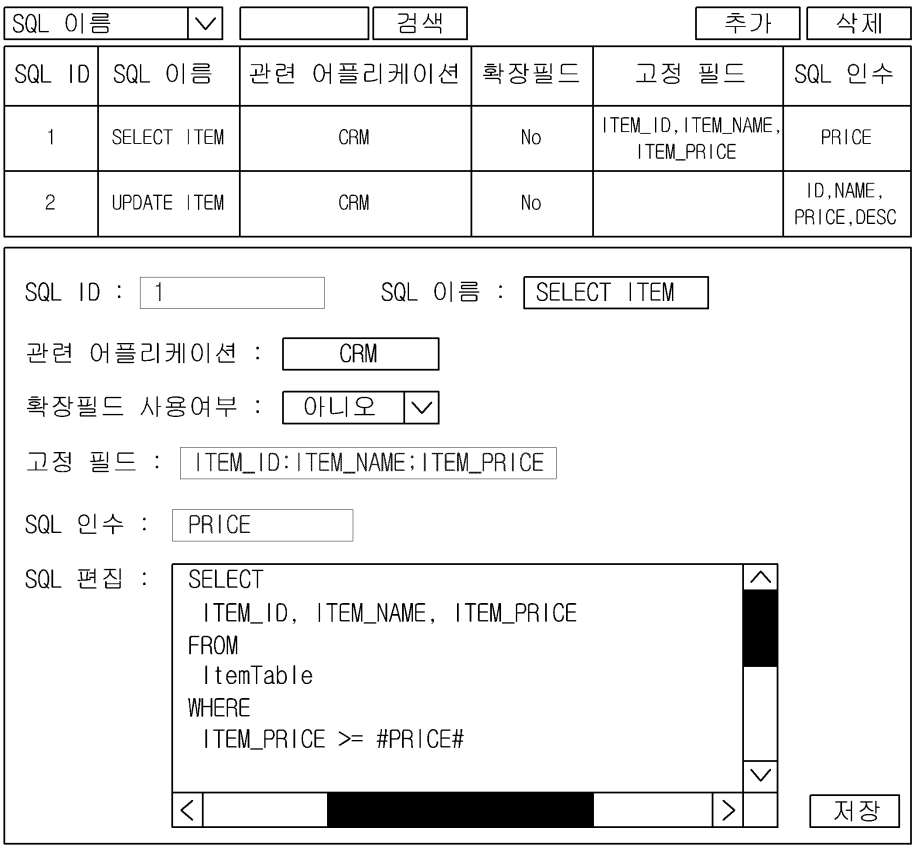
도면2


[비지니스 로직
개발 서비스]
브라우저
로직 개발 (400)
개발
[사용자 정의 로직 편집기]
SQL 쿼리 추가 (401)
[사용자 정의 로직 편집기]
•입출력 파라미터/값 정의 (402)
•SQL 실행 함수 추가
(메타데이터처리 API) (403)
•사용자 정의 코드 추가
(예:Java) (404)
UI 매핑
[사용자 정의 로직 편집기]
SQL 쿼리 추가 (405)
메타데이터 저장소 (406)

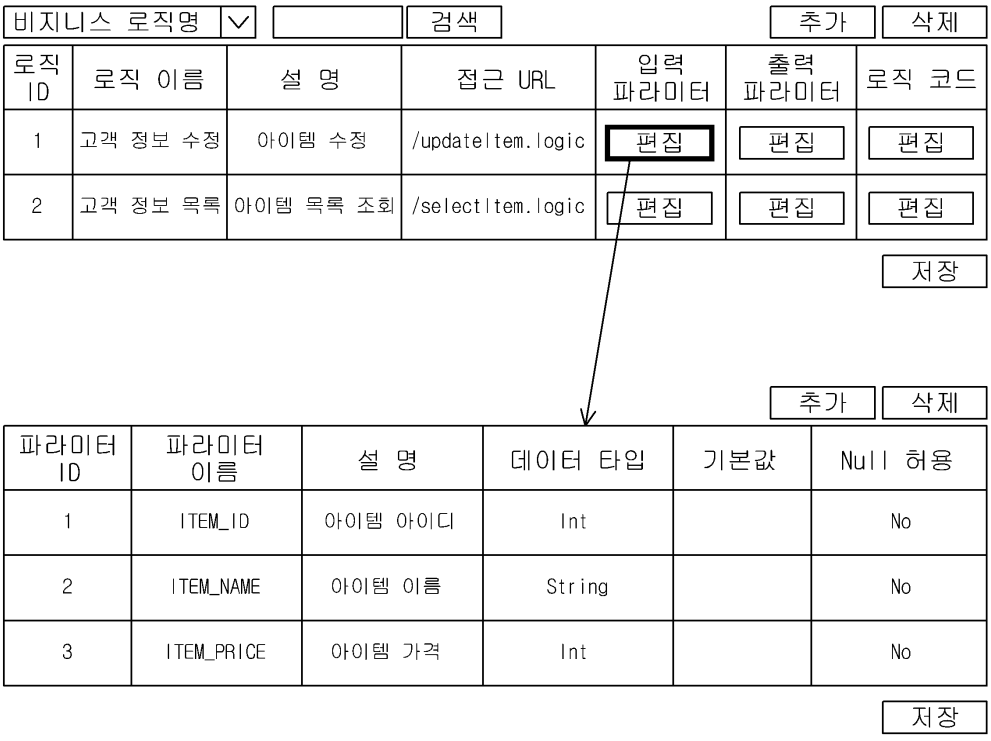
도면3


[서비스]
브라우저
비지니스 로직 실행 요청 (700)
[서비스 컨트롤러]
서비스 컨트롤러에
비지니스 로직 클래스화
(701)
인스턴트 실행
[실행엔진]
실행 결과를 웹 페이지로
생성 (706)
[실행엔진]
로직 처리 함수호출 (705)
[실행엔진]
동적 컴파일 로딩 (704)
[실행엔진]
쿼리, 사용자 정의
로직(코드) 얻기 (702)
[실행엔진]
동적 클래스 생성 (703)
실행 결과
전달 (707)

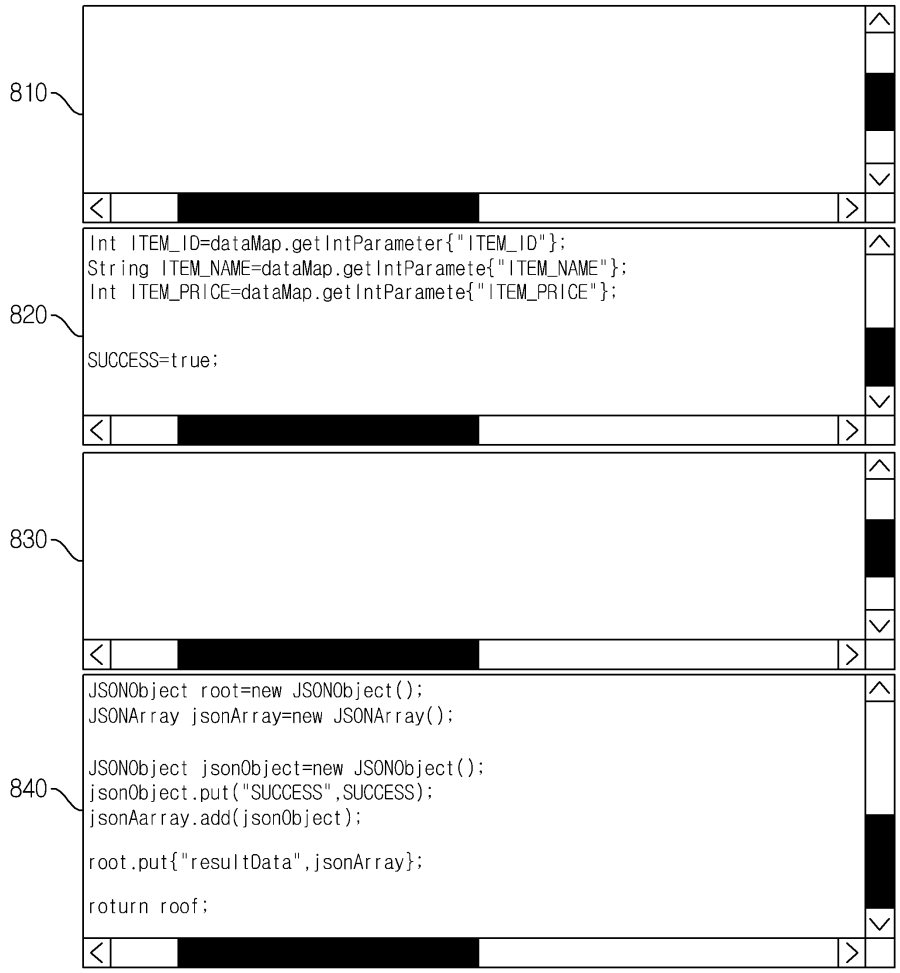
도면4


SQL 이름
검색
추가
삭제
SQL ID
SQL 이름
관련 어플리케이션
확장필드
고정 필드
SQL 인수
1
SELECT ITEM
CRM
No
ITEM_ID,ITEM_NAME,ITEM_PRICE
PRICE
2
UPDATE ITEM
CRM
No
ID,NAME,PRICE,DESC
SQL ID : 1
SQL 이름 : SELECT ITEM
관련 어플리케이션 : CRM
확장필드 사용여부 : 아니오
고정 필드 : ITEM_ID:ITEM_NAME:ITEM_PRICE
SQL 인수 : PRICE
SQL 편집 :
SELECT
ITEM_ID, ITEM_NAME, ITEM_PRICE
FROM
ItemTable
WHERE
ITEM_PRICE >= #PRICE#
저장

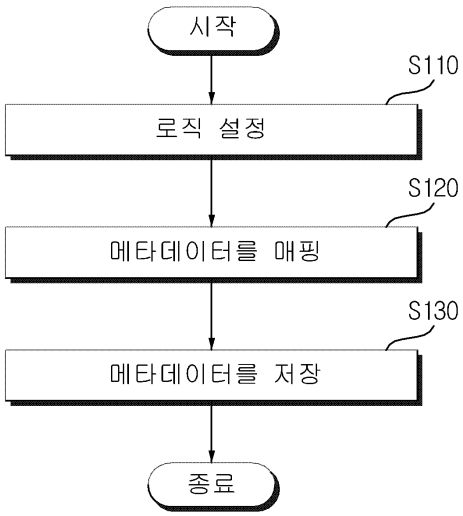
도면5


비지니스 로직명
검색
추가
삭제
로직 ID
로직 이름
설 명
접근 URL
입력 파라미터
출력 파라미터
로직 코드
1
고객 정보 수정
아이템 수정
/updateItem.logic
편집
편집
편집
2
고객 정보 목록
아이템 목록 조회
/selectItem.logic
편집
편집
편집
저장
추가
삭제
파라미터 ID
파라미터 이름
설 명
데이터 타입
기본값
Null 허용
1
ITEM_ID
아이템 아이디
Int
No
2
ITEM_NAME
아이템 이름
String
No
3
ITEM_PRICE
아이템 가격
Int
No
저장

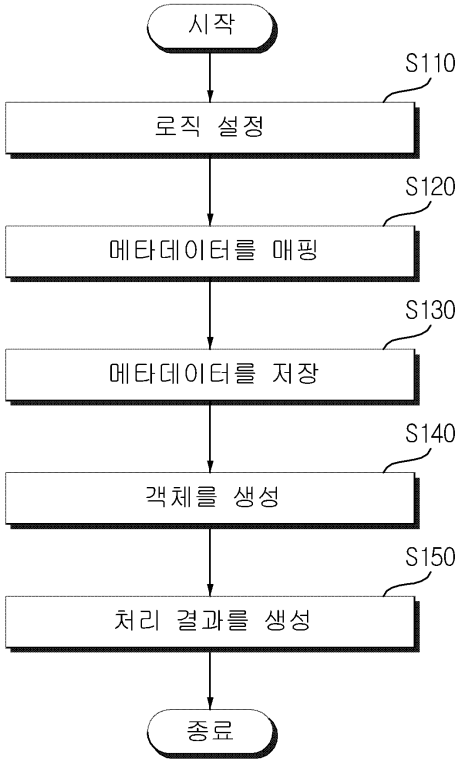
도면6


810
820
Int ITEM_ID=dataMap.getIntParameter{"ITEM_ID"};
String ITEM_NAME=dataMap.getIntParamete{"ITEM_NAME"};
Int ITEM_PRICE=dataMap.getIntParamete{"ITEM_PRICE"};
SUCCESS=true;
830
840
JSONObject root=new JSONObject();
JSONArray jsonArray=new JSONArray();
JSONObject jsonObject=new JSONObject();
jsonObject.put("SUCCESS",SUCCESS);
jsonAarray.add(jsonObject);
root.put{"resultData",jsonArray};
roturn roof;

도면7


시작
S110
로직 설정
S120
메타데이터를 매핑
S130
메타데이터를 저장
종료

도면8


시작
S110
로직 설정
S120
메타데이터를 매핑
S130
메타데이터를 저장
S140
객체를 생성
S150
처리 결과를 생성
종료