(19) 대한민국특허청(KR)
(12) 공개특허공보(A)

(11) 공개번호 10-2010-0056938
(43) 공개일자 2010년05월28일

(51) Int. Cl.
G06F 9/06 (2006.01) *G06F 9/44* (2006.01)
G06Q 50/00 (2006.01)
(21) 출원번호 10-2008-0115957
(22) 출원일자 2008년11월20일
심사청구일자 2008년11월20일

(71) 출원인
엔에이치엔(주)
경기도 성남시 분당구 정자동 25-1 분당벤처타운
(72) 발명자
이승배
경기도 용인시 수지구 죽전1동 극동스타클래스아파트 201동 501호
김성관
경기도 용인시 수지구 죽전1동 우미이노스빌2차 202동 1402호
(뒷면에 계속)
(74) 대리인
특허법인에이아이피

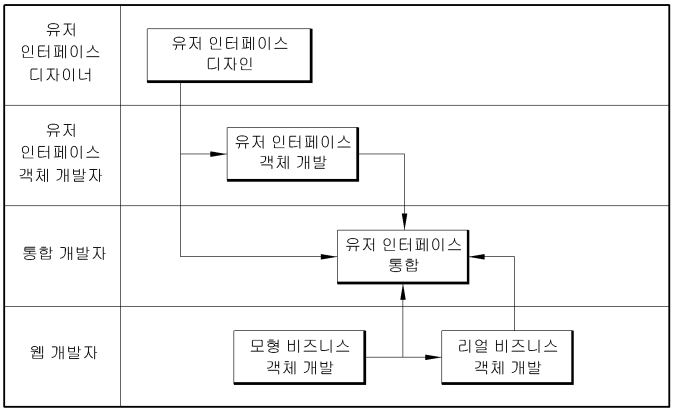
전체 청구항 수 : 총 16 항

(54) 웹 페이지를 생성하기 위해 사용되는 컴포넌트 기반의 자바웹 애플리케이션 개발 프레임워크

(57) 요 약

웹 페이지를 생성하기 위해 사용되는 컴포넌트 기반의 자바 웹 애플리케이션 개발 프레임워크는 사용자의 요청에 따라, 비즈니스 객체(BO: Business Object)의 메소드를 수행하고 상기 수행 결과를 반환하는 런타임 엔진, 상기 비즈니스 객체의 메소드의 수행 전에 사용자 입력 값의 전처리를 수행하고, 상기 비즈니스 객체의 메소드의 수행 후에 상기 수행 결과를 가공하는 후처리를 수행하며, 유저 인터페이스 객체(UIO: User Interface Object)를 동적으로 배치하는 스크립트 콤포넌트 및 상기 프레임워크의 초기화 작업을 수행하고, 상기 사용자의 요청이 상기 프레임워크에서 처리가 가능한 요청인지를 확인하는 코어 콤포넌트를 포함한다.

대 표 도 - 도2



(72) 발명자

문규원

경기도 성남시 분당구 정자동 두산위브파빌리온 A동 1418호

윤미선

서울특별시 강남구 역삼2동 역삼e-편한세상아파트 111동 602호

송창근

서울특별시 강북구 미아4동 8-188 1층

신옥수

서울특별시 영등포구 신길6동 우성2차아파트 202동 1201호

장재완

경기도 남양주시 평내동 신명아파트 1811동 1703호

특허청구의 범위

청구항 1

웹 페이지를 생성하기 위해 사용되는 컴포넌트 기반의 자바 웹 애플리케이션 개발 프레임워크에 있어서,

사용자의 요청에 따라, 비즈니스 객체(BO: Business Object)의 메소드를 수행하고 상기 수행 결과를 반환하는 런타임 엔진,

상기 비즈니스 객체의 메소드의 수행 전에 사용자 입력 값의 전처리를 수행하고, 상기 비즈니스 객체의 메소드의 수행 후에 상기 수행 결과를 가공하는 후처리를 수행하며, 유저 인터페이스 객체(UIO: User Interface Object)를 동적으로 배치하는 스크립트 콤포넌트 및

상기 프레임워크의 초기화 작업을 수행하고, 상기 사용자의 요청이 상기 프레임워크에서 처리가 가능한 요청인지를 확인하는 코어 콤포넌트

를 포함하는 웹 애플리케이션 개발 프레임워크.

청구항 2

제 1 항에 있어서,

상기 런타임 엔진은 XML 기반의 설정 파일에 선언된 상기 비즈니스 객체의 메소드를 수행하는 서비스 수행부,

상기 비즈니스 객체의 메소드가 정상적으로 실행된 경우, 상기 수행 결과를 상기 XML 기반의 설정 파일에 선언된 결과 형식별로 출력하는 결과 생성부,

상기 프레임워크에서 사용하는 다양한 자원에 대한 접근 및 사용 인터페이스를 제공하는 단일 리소스 액세서(URA: Uniform Resource Accessor) 및

상기 비즈니스 객체의 메소드가 비정상적으로 실행되거나 예외 상황이 발생할 경우, 디버깅 정보를 관리하는 예외 상황 핸들러

를 포함하는 웹 애플리케이션 개발 프레임워크.

청구항 3

제 1 항에 있어서,

상기 스크립트 콤포넌트는 상기 XML 기반의 설정 파일에 기초하여, 상기 전처리 및 후처리를 수행하고, 상기 유저 인터페이스 객체를 제어하는 스크립트 엔진 및

상기 유저 인터페이스 객체를 HTML 템플릿으로 관리하고, 상기 유저 인터페이스 객체를 런타임 시에 HTML로 렌더링하여 제공하는 오픈 소스 템플릿 엔진

을 포함하는 웹 애플리케이션 개발 프레임워크.

청구항 4

제 2 항에 있어서,

상기 결과 생성부에 의해서 출력되는 결과 형식은 상기 수행 결과를 템플릿으로 생성한 후 출력하는 템플릿, 상기 수행 결과를 지정된 URL로 리다이렉트하는 리다이렉트, 상기 수행 결과를 지정된 URL로 포워드하는 포워드, 상기 수행 결과를 페이지 없이 직접 출력하는 다이렉트 중 하나인 것인 웹 애플리케이션 개발 프레임워크.

청구항 5

제 1 항에 있어서,

상기 스크립트 컴포넌트는 자바스크립트(Javascript)에 기초하여 상기 전처리, 상기 후처리, 상기 유저 인터페이스 객체(UIO: User Interface Object)의 동적 배치를 수행하는 것인 웹 애플리케이션 개발 프레임워크.

청구항 6

제 1 항에 있어서,

상기 프레임워크는 런타임 시에 상기 비즈니스 객체 및 상기 유저 인터페이스 객체를 결합하는 동적 바인딩(Dynamic Binding) 서비스를 제공하는 것인 웹 애플리케이션 개발 프레임워크.

청구항 7

제 1 항에 있어서,

상기 유저 인터페이스 객체는 상기 프레임워크에서 정의한 템플릿 지시어(template directive)를 이용하여 동적으로 재배치할 수 있는 것인 웹 애플리케이션 개발 프레임워크.

청구항 8

컴포넌트 기반의 자바 웹 애플리케이션 개발 프레임워크를 이용하여 웹 페이지를 생성하는 방법에 있어서,

웹 브라우저를 통해서 사용자 요청을 수신하는 동작,

상기 수신된 사용자 요청이 상기 프레임워크에서 처리가 가능한 요청인 경우, 상기 사용자 요청의 실행을 지시하는 동작,

상기 사용자 요청의 실행 전에 정의된 전처리 작업이 있을 경우, 상기 전처리 작업을 수행하는 동작,

상기 사용자 요청에 따라, 비즈니스 객체(BO: Business Object)의 메소드를 수행하고 상기 수행 결과를 반환하는 동작,

상기 메소드가 정상적으로 수행된 경우, 상기 수행 결과를 XML 기반의 설정 파일에 선언된 결과 형식별로 출력하는 동작 및

상기 메소드가 정상적으로 수행된 경우에 정의된 후처리 작업이 있는 경우, 상기 후처리 작업을 수행하는 동작

을 포함하는 웹 페이지 생성 방법.

청구항 9

제 8 항에 있어서,

상기 결과 형식은 상기 수행 결과를 템플릿으로 생성한 후 출력하는 템플릿, 상기 수행 결과를 지정된 URL로 리다이렉트하는 리다이렉트, 상기 수행 결과를 지정된 URL로 포워드하는 포워드, 상기 수행 결과를 페이지 없이 직접 출력하는 다이렉트 중 하나인 것인 웹 페이지 생성 방법.

청구항 10

제 8 항에 있어서,

상기 비즈니스 객체의 메소드가 비정상적으로 실행되거나 예외 상황이 발생할 경우, 디버깅 정보를 관리하는 동작을 더 포함하는 웹 페이지 생성 방법.

청구항 11

제 8 항에 있어서,

상기 수행 결과를 HTML 형태로 상기 웹 브라우저에게 제공하는 동작을 더 포함하는 웹 페이지 생성 방법.

청구항 12

제 8 항에 있어서,

상기 전처리 작업 및 상기 후처리 작업은 자바스크립트(Javascript)에 기초하여 수행되는 것인 웹 페이지 생성 방법.

청구항 13

제 8 항에 있어서,

XML 기반의 설정 파일에 기초하여, 유저 인터페이스 객체(UIO: User Interface Object)를 동적으로 배치하는 동작을 더 포함하는 웹 페이지 생성 방법.

청구항 14

제 13 항에 있어서,

상기 비즈니스 객체 및 상기 유저 인터페이스 객체는 런타임 시에 동적으로 결합되어 HTML 웹 페이지로서 제공되는 것인 웹 페이지 생성 방법.

청구항 15

제 13 항에 있어서,

상기 유저 인터페이스 객체는 상기 프레임워크에서 정의한 템플릿 지시어(template directive)를 이용하여 동적으로 재배치할 수 있는 것인 웹 페이지 생성 방법.

청구항 16

청구항 제 8 항 내지 제 15 항 중 어느 한 항에 따른 웹 페이지 생성 방법의 각 단계를 컴퓨터 상에서 수행하기 위한 프로그램을 기록한 컴퓨터 판독 가능한 기록 매체.

명 세 서

발명의 상세한 설명

기 술 분 야

[0001] 본 발명은 유저 인터페이스 개발자가 컴포넌트 기반으로 최종 유저 인터페이스를 개발할 수 있도록 지원하는 유저 인터페이스 서버 프레임워크 및 그 이용 방법에 관한 것이다.

배 경 기 술

[0002] 최근에는 CBD(Component Based Development), SOA(Service Oriented Architecture) 등의 개념이 산업에 점차 적용이 되면서 일반적으로 칭하는 비즈니스 객체(Business Object)의 재사용성 및 연동 기능이 많이 강화되었으

며, 이와 관련된 솔루션과 기술 역시 발전하였다. 반면에, Web 2.0, Ajax 등의 개념이 산업에 도입이 되면서 웹 서비스의 유저 인터페이스(User Interface)가 급속도로 복잡해 지고 있으나, 유저 인터페이스 객체의 컴포넌트화나 재사용은 걸음마 단계에 머물러 있다. 이에 따라, 유저 인터페이스 객체의 개발 비용이 점차 증가하고 있다.

[0003] 또한, 대부분의 웹 기반 사용자 서비스는 유저 인터페이스 영역과 비즈니스 영역이 밀접하게 결합되어 있다. 즉, 유저 인터페이스 영역과 비즈니스 영역이 별도의 레이어(Layer)로 명확하게 분리되어 있지 않고, 하나의 결합체로만 존재한다. 또한, 유저 인터페이스 영역과 비즈니스 영역이 별도의 레이어로 분리가 되어 있다 하더라도, 레이어 간의 의존도가 높아 같은 모듈이라고 할 수 있는 구조가 대부분이다.

[0004] 이와 같이 유저 인터페이스 영역과 비즈니스 영역이 밀접하게 결합되어 있는 구조를 사용하면, 일시적으로는 유저 인터페이스 객체 개발 비용을 낮추는 효과를 얻을 수 있지만, 유지, 보수 비용을 포함하면 총 소요 비용은 증가하게 된다. 또한, 유사한 구조의 유저 인터페이스 객체를 개발하는 경우에도 기존의 유저 인터페이스 객체의 재사용이 불가능하여 많은 비용과 시간이 소요되는 문제가 있었다.

[0005] 도 1은 종래의 일반적인 웹 애플리케이션 개발 프로세스를 도시한 도면이다.

[0006] 도 1에 도시된 바와 같이, 유저 인터페이스 개발 프로세스는 유저 인터페이스 객체의 디자인 작업, HTML 개발 작업, 자바스크립트(Javascript) 개발 작업, 비즈니스 객체의 개발 작업, 자바 서버 페이지(Java Server Pages) 변경 작업 및 유저 인터페이스 객체의 통합 작업을 포함할 수 있다.

[0007] 일반적으로 HTML 개발이 완료되었다 하더라도 이를 서비스에서 바로 사용할 수는 없다. HTML 개발자는 HTML을 개발하는 과정에 있어서, 실제 데이터가 아닌 상수 값을 사용하여 HTML을 개발하므로, HTML 개발자가 개발한 HTML 자체로는 웹 서비스에서 사용하는 동적인 데이터를 표현할 수 없기 때문이다.

[0008] 따라서, HTML을 자바 서버 페이지로 변환하는 자바 서버 페이지 변경 작업이 필수적으로 수행되어야 한다. 상기 작업은 HTML 결과물에서 상수 값을 제거하고 비즈니스 객체에서 제공하는 동적 데이터를 표시할 수 있도록 전환하는 작업이다.

[0009] 또한, 자바 서버 페이지 변경 작업을 수행한 후에는, 실제로 동작하는 유저 인터페이스를 확인하고, 검증하는 작업인 유저 인터페이스 객체의 통합 작업이 수행될 수 있다.

[0010] 상기 작업들 중 일부는 병행하여 수행될 수도 있고, 일부는 순차적으로 수행될 수도 있다. 예를 들어, 도 1에 도시된 바와 같이, HTML 개발 작업, 자바스크립트 개발 작업 및 비즈니스 객체의 개발 작업은 병행하여 수행이 가능하다. 그 후, 자바 서버 페이지 변환 작업 및 유저 인터페이스 객체의 통합 작업이 연속하여 수행될 수 있다.

[0011] 그러나 종래에는 자바 서버 페이지 변경 작업 및 유저 인터페이스 객체의 통합 작업이 웹 개발자가 사용하는 프로그래밍 언어를 통해 이루어지므로, 반드시 웹 개발자가 수행하여야만 하며, 이 부분에서 병목 현상이 발생하는 문제가 있었다. 또한, 병목 현상을 해소하기 위해, 웹 개발자를 추가로 투입한다 하더라도, 웹 개발자는 HTML, 유저 인터페이스 객체 등에 대한 이해가 부족하기 때문에 병목 현상의 해소에 크게 도움이 되지 못한다.

[0012] 또한, 실제로 적용될 유저 인터페이스를 확인하는 유저 인터페이스 객체의 통합 작업이 제일 마지막에 수행되기 때문에, 유저 인터페이스 개발 프로젝트의 가시성 확보가 어렵다. 즉, 실제로 동작하는 웹 유저 인터페이스 객체를 확인하고 검증하는 작업이 제일 마지막에 수행되므로, 개발자 및 사용자 간의 피드백, 화면 간의 연결 확인 등이 어려운 문제가 있었다.

[0013] 한편, 선행 기술로서, 대한민국 특허공개번호 제2008-0070377호에는 " 자바스크립트기반 웹-클라이언트 애플리케이션 프레임워크와 상기 프레임워크를 이용한 웹 컨텐츠 처리방법 및 이를 구현할 수 있는 컴퓨터로 읽을 수 있는 기록매체"라는 명칭의 발명이 개시되어 있는 바, 상기 선행 기술은 자바스크립트를 이용하여 프레임워크를 구성하고, XML 문서를 이용하여 객체화 및 재사용성을 높이고, 객체 지향적 방식의 클라이언트 애플레케이션 개발이 가능하도록 한 프레임워크에 관한 것이다.

[0014] 하지만, 상기 선행 기술은 유저 인터페이스 객체의 통합 작업이 데이터를 관리하는 웹 개발자에 의해 수행되므로, 프로세스 진행이 원활하게 이루어 지지 못하는 문제점이 있다. 또한, 유저 인터페이스 객체의 확인 작업이 마지막에 이루어지므로 유저 인터페이스 객체의 오류의 발견이 늦어지는 문제점이 있다.

발명의 내용

해결 하고자하는 과제

[0015] 본 발명의 일 실시예는 웹 페이지를 생성하기 위해 사용되는 컴포넌트 기반의 자바 웹 애플리케이션 개발 프레임워크를 제공한다.

[0016] 또한, 본 발명의 일 실시예는 컴포넌트 기반의 자바 웹 애플리케이션 개발 프레임워크를 이용하여 웹 페이지를 생성하는 방법을 제공한다.

과제 해결수단

[0017] 상술한 기술적 과제를 달성하기 위한 기술적 수단으로서, 본 발명의 제 1 측면은, 사용자의 요청에 따라, 비즈니스 객체(BO: Business Object)의 메소드를 수행하고 상기 수행 결과를 반환하는 런타임 엔진, 상기 비즈니스 객체의 메소드의 수행 전에 사용자 입력 값의 전처리를 수행하고, 상기 비즈니스 객체의 메소드의 수행 후에 상기 수행 결과를 가공하는 후처리를 수행하며, 유저 인터페이스 객체를 동적으로 배치하는 스크립트 콤포넌트 및 상기 프레임워크의 초기화 작업을 수행하고, 상기 사용자의 요청이 상기 프레임워크에서 처리가 가능한 요청인지를 확인하는 코어 콤포넌트를 포함하는 웹 페이지를 생성하기 위해 사용되는 컴포넌트 기반의 자바 웹 애플리케이션 개발 프레임워크를 제공할 수 있다.

[0018] 또한, 본 발명의 제 2 측면은 웹 브라우저를 통해서 사용자 요청을 수신하는 동작, 상기 수신된 사용자 요청이 상기 프레임워크에서 처리가 가능한 요청인 경우, 상기 사용자 요청의 실행을 지시하는 동작, 상기 사용자 요청의 실행 전에 정의된 전처리 작업이 있을 경우, 상기 전처리 작업을 수행하는 동작, 상기 사용자 요청에 따라, 비즈니스 객체의 메소드를 수행하고 상기 수행 결과를 반환하는 동작, 상기 메소드가 정상적으로 수행된 경우, 상기 수행 결과를 XML 기반의 설정 파일에 선언된 결과 형식별로 출력하는 동작 및 상기 메소드가 정상적으로 수행된 경우에 정의된 후처리 작업이 있는 경우, 상기 후처리 작업을 수행하는 동작을 포함하는 컴포넌트 기반의 자바 웹 애플리케이션 개발 프레임워크를 이용하여 웹 페이지를 생성하는 방법을 제공할 수 있다.

효 과

[0019] 전술한 본 발명의 과제 해결 수단에 의하면, 유저 인터페이스 개발자에게 비즈니스 객체와 독립적으로 유저 인터페이스 객체를 개발할 수 있는 환경을 제공하고, 웹 개발자가 아닌 유저 인터페이스 개발자가, 별도로 개발된 비즈니스 객체를 호출하고 실행하여, 결과를 확인할 수 있다.

[0020] 전술한 본 발명의 과제 해결 수단에 의하면, 웹 개발자 및 유저 인터페이스 개발자가 서로의 작업 결과물에 영향을 받지 않고 독립적으로 업무를 진행할 수 있으며, 지속적인 통합 작업을 개발 기간 중에 할 수 있다.

발명의 실시를 위한 구체적인 내용

[0021] 아래에서는 첨부한 도면을 참조하여 본 발명이 속하는 기술 분야에서 통상의 지식을 가진 자가 용이하게 실시할 수 있도록 본 발명의 실시예를 상세히 설명한다. 그러나 본 발명은 여러 가지 상이한 형태로 구현될 수 있으며 여기에서 설명하는 실시예에 한정되지 않는다. 그리고 도면에서 본 발명을 명확하게 설명하기 위해서 설명과 관계없는 부분은 생략하였으며, 명세서 전체를 통하여 유사한 부분에 대해서는 유사한 도면 부호를 붙였다.

[0022] 명세서 전체에서, 어떤 부분이 다른 부분과 "연결"되어 있다고 할 때, 이는 "직접적으로 연결"되어 있는 경우뿐 아니라, 그 중간에 다른 소자를 사이에 두고 "전기적으로 연결"되어 있는 경우도 포함한다. 또한 어떤 부분이 어떤 구성요소를 "포함"한다고 할 때, 이는 특별히 반대되는 기재가 없는 한 다른 구성요소를 제외하는 것이 아니라 다른 구성요소를 더 포함할 수 있는 것을 의미한다.

[0023] 본 명세서에서, 비즈니스 객체란, 서비스의 주요한 로직을 수행하고 그 결과 데이터를 반환하는 역할을 하는 컴포넌트이다. 상기 비즈니스 객체는 데이터의 표현에는 관여하지 않고 데이터의 내용에만 관여한다. 또한, 유저 인터페이스 객체란, 웹 페이지에서 단위 서비스를 구성하는 데에 사용되는 유저 인터페이스 컴포넌트를 의미한다. 상기 유저 인터페이스 객체는 데이터의 내용에는 관여하지 않고 데이터의 표현에만 관여한다.

[0024] 이하 첨부된 도면을 참고하여 본 발명을 상세히 설명하기로 한다.

[0025] 도 2는 본 발명의 일 실시예에 따른 웹 애플리케이션 개발 프로세스를 도시한 도면이다.

[0026] 도 2에 도시된 바와 같이, 본 발명의 일 실시예에 따른 웹 애플리케이션 개발 프로세스는 유저 인터페이스 객체의 디자인 작업, 유저 인터페이스 객체의 개발 작업, 유저 인터페이스 객체의 통합 작업, 모형 비즈니스 객체의

개발 작업 및 리얼(Real) 비즈니스 객체의 개발 작업을 포함할 수 있다.

[0027] 본 발명의 일 실시예에 따른 웹 애플리케이션 개발 프로세스에서도, 실제로 동작하는 유저 인터페이스 객체를 확인하고 검증하는 작업인 유저 인터페이스 객체의 통합 작업이 반드시 필요하다. 그러나, 도 2에 도시된 바와 같이, 본 발명의 일 실시에에 따른 웹 애플리케이션 개발 프로세스에서는, 유저 인터페이스 객체의 통합 작업을 웹 개발자가 아닌 HTML 개발자 또는 통합 개발자가 수행할 수 있다. 즉, 프로세스의 병목 문제를 해결하기 위해서, 프로세스의 마지막 단계가 아닌, 유저 인터페이스 객체 및 HTML을 개발하는 단계에서부터 유저 인터페이스 객체와, 비즈니스 객체를 통합하는 작업이 수행될 수 있다.

[0028] 또한, 통합 작업이 1회성이 아닌 개발 시작부터 종료까지 지속적으로 반복 수행될 수 있다. 즉, 통합 작업의 리스크를 감소하기 위해서, 개발 초기 단계부터 마지막 단계까지 지속적인 통합 방식을 도입하여, 개발 기간 단축 및 리스크 감소 효과를 얻을 수 있다.

[0029] 또한, 개발 초기부터 유저 인터페이스 객체와, 비즈니스 객체를 통합하는 작업을 수행하기 위해서는, 비즈니스 객체에서 데이터를 가져온 후, 이를 테스트 하는 작업이 수행되어야 한다. 이를 위해 본 발명의 일 실시예에 따른 웹 애플리케이션 개발 프로세스에서는, 실제 비즈니스 객체를 개발하기 이전에, 모형 비즈니스 객체를 생성하여, 이를 유저 인터페이스 객체의 통합 작업에서 사용할 수 있다.

[0030] 따라서, 본 발명에 의하면, 웹 개발자 및 유저 인터페이스 개발자가 서로의 작업 결과물에 영향을 받지 않고 독립적으로 업무를 진행할 수 있으며, 지속적인 통합 작업을 개발 기간 중에 할 수 있다.

[0031] 도 3은 본 발명의 일 실시예에 따른 유저 인터페이스 서버 프레임워크(User Interface Server Framework, 이하 USF라 칭함)의 세부 구성도이다.

[0032] 본 발명의 일 실시예에 따른 USF(1000)는 사용자에게 HTML을 제공하기 위해 사용하는 컴포넌트 기반의 자바 웹 애플리케이션 개발 프레임워크로서, 유저 인터페이스 개발자가 HTML 생성을 위한 유저 인터페이스 객체와 비즈니스 객체를 독립적으로 개발하여 통합할 수 있는 프레임워크이다. 즉, USF(1000)는 런타임 시에 상기 비즈니스 객체 및 상기 유저 인터페이스 객체를 결합하는 동적 바인딩(Dynamic Binding) 서비스를 제공할 수 있다.

[0033] USF(1000)는 이를 위해 다양한 기능을 제공할 수 있다. 첫째, USF(1000)는 비즈니스 객체와 유저 인터페이스 객체를 독립적으로 개발할 수 있는 기능을 제공한다. 따라서, 유저 인터페이스 개발자는 웹 개발자가 사용하는 프로그래밍 언어를 별도로 배우지 않고, USF(1000)에서 제공하는 기능만으로 유저 인터페이스 객체를 개발할 수 있다. USF(1000)는 비즈니스 객체와 유저 인터페이스 객체를 독립적으로 개발할 수 있는 기능을 제공하기 위해, XML 기반의 설정 파일과 HTML 생성 템플릿 엔진을 제공한다.

[0034] 또한, 유저 인터페이스 객체 개발 후에, 상기 개발한 유저 인터페이스 객체를 실제 서비스 데이터와 결합하여 보여줄 수 있는 기능이 필요한데, 이를 위해 USF(1000)는 로컬(Local) 및 원격으로 비즈니스 객체를 호출하여 수행하고 그 결과를 수신하는 기능을 제공한다. 예를 들어, USF(1000)는 오픈 소스 프레임워크 연동 기능을 통해 로컬 비즈니스 객체를 호출하고 수행할 수 있다. 또한, USF(1000)는 RPC(Remote Procedure Call) 프로토콜과 애플리케이션 간의 메시지를 전송하고 관리할 수 있는 메시징 플랫폼 프로토콜을 지원하여, 원격으로 비즈니스 객체를 호출하고 수행할 수 있다.

[0035] 또한, USF(1000)에서는 XML 기반에서 비즈니스 객체를 호출하여 수행할 수 있다. 예를 들어, USF(1000)에서는 수행할 비즈니스 객체의 이름, 템플릿 형식의 이름, 전달할 파라미터 및 돌려 받을 데이터 등을 XML 상에서 기술하는 방식을 사용할 수 있다.

[0036] 둘째, USF(1000)에서는 오픈 소스 템플릿 엔진 기반의 템플릿 페이지를 해석하고 실행하여 HTML로 전환하는 기능을 제공할 수 있다. 나아가, 단순히 템플릿 엔진만을 사용할 경우에는 동적으로 유저 인터페이스를 구성할 수 없는 문제가 있기 때문에, USF(1000)에서는 템플릿 지시어(Template Directive)를 정의하여 자바 서버 페이지 내에서 동적으로 유저 인터페이스 객체를 배치할 수 있는 기능을 제공할 수 있다. 예를 들어, 유저 인터페이스 개발자는 USF(1000)에 내장되어 있는 자바스크립트 엔진을 활용하여, 자신이 원하는 방식으로 유저 인터페이스 객체를 화면에 배치할 수 있다.

[0037] 또한, USF(1000)는, 일반적인 웹 페이지의 수행 결과인 HTML이외에, 상황에 따라 다른 페이지로의 이동이 가능하도록 웹 애플리케이션에 대한 응답 결과의 종류를 다양하게 제공할 수 있다. 예를 들어, USF(1000)는 상기 수행 결과를 템플릿으로 생성한 후 출력하는 템플릿(template), 상기 수행 결과를 지정된 URL로 리다이렉트하는 리다이렉트(redirect), 상기 수행 결과를 지정된 URL로 포워드 하는 포워드(forward) 및 상기 수행 결과를 페이

지 없이 직접 출력하는 다이렉트(direct) 등의 다양한 응답 결과의 종류를 제공하여, 웹 애플리케이션에 대한 응답 결과를 종류별로 적절하게 출력할 수 있다.

[0038] 셋째, USF(1000)는 애플리케이션 개발자들이 사용자 요청에 대해 추가적으로 작업을 수행할 수 있도록 하는 환경을 제공할 수 있다. 예를 들어, USF(1000)에서는 애플리케이션 개발자들이, 퍼미션(Permission)이나 쿠키(Cookie)와 같은 애플리케이션 단계에서 가공 및 처리할 수 있는 추가 작업을 수행할 수 있도록 하는 환경을 제공할 수 있다.

[0039] 도 3에 도시된 바와 같이, 본 발명의 일 실시예에 따른 USF(1000)는 런타임 엔진(100), 스크립트 콤포넌트(200), 코어 콤포넌트(300), 웹 애플리케이션 베이스 모듈(400) 및 웹 애플리케이션 서버(500)를 포함한다.

[0040] 런타임 엔진(100)은 사용자의 요청에 따라, 비즈니스 객체의 메소드(Method)를 수행하고, 상기 비즈니스 객체의 수행 결과를 XML 기반의 설정 파일에 선언된 결과 형식에 따라 다양하게 출력할 수 있다. 또한, 런타임 엔진(100)은 웹 개발자의 USF(1000)에서 사용되는 다양한 자원에 대한 접근을 가능하게 하고, 사용 인터페이스를 제공할 수 있다. 나아가, 런타임 엔진(100)은 상기 비즈니스 객체의 메소드가 비정상적으로 실행되거나, 예외 상황이 발생한 경우에, 이에 대한 디버깅 정보를 관리할 수 있다.

[0041] 스크립트 콤포넌트(200)는 자바스크립트(Javascript)에 기초하여, 비즈니스 객체의 메소드 수행 전에, XML 기반의 설정 파일에 정의되어 있는 사용자 입력 값의 전처리를 수행하고, 상기 비즈니스 객체의 메소드 수행 후에 상기 수행 결과를 가공하는 후처리를 수행할 수 있다. 또한, 스크립트 콤포넌트(200)는 유저 인터페이스 객체를 제어하여, 동적으로 배치할 수 있다. 또한, 상기 유저 인터페이스 객체를 HTML 템플릿으로 관리하고, 런타임 시에 HTML로 렌더링한다.

[0042] 코어 콤포넌트(300)는 사용자의 요청이 USF(1000)에서 실행 가능한지 여부를 확인하고, USF(1000)에서 실행이 가능한 경우 이를 실행하게 할 수 있다. 또한, 코어 콤포넌트(300)는 프레임워크를 초기화하는 역할을 수행할 수 있다.

[0043] 웹 애플리케이션 베이스 모듈(400)은 자바 기반 웹 애플리케이션 프레임워크로서, 엑스워크 프레임워크(XWork framework), 재사용 가능한 유저 인터페이스 템플릿(Reusable UI Templates) 및 자바빈스(JavaBeans)를 통합한 웹 애플리케이션 개발 플랫폼이다. 웹 애플리케이션 베이스 모듈(400)은 비핵심적인 비즈니스 객체를 선언적으로 처리하여 개발자들이 웹 애플리케이션 개발에 필요한 핵심적인 비즈니스 객체만 집중적으로 구현할 수 있도록 설계될 수 있다.

[0044] 웹 애플리케이션 서버(Web Application Server)(500)는 인터넷 상에서 HTTP를 통해 사용자의 컴퓨터나 장치에 애플리케이션을 수행해 주는 미들웨어(소프트웨어 엔진)이다. 웹 애플리케이션 서버(500)는 동적 서버 콘텐츠를 수행하는 것으로 일반적인 웹 서버와 구별이 되며, 주로 데이터베이스 서버와 같이 수행이 된다. 예를 들어, 웹 애플리케이션 서버(500)는 톰캣(Tomcat), 레진(Resin), 제이런(JRun) 등이 될 수 있다.

[0045] 도 4는 본 발명의 일 실시예에 따른 USF(1000)의 런타임 엔진(100)의 세부 구성도이다.

[0046] 도 4에 도시된 바와 같이, 본 발명의 일 실시예에 따른 USF(1000)의 런타임 엔진(100)은 서비스 수행부(110), 결과 생성부(120), 단일 리소스 액세서(URA: Uniform Resource Accessor)(130) 및 예외 상황 핸들러(140)를 포함한다.

[0047] 서비스 수행부(110)는 XML 기반의 설정 파일에 선언되어 있는 비즈니스 객체의 메소드를 수행한다. 상기 XML 기반의 설정 파일에 선언되어 있는 비즈니스 객체는, 해당 서비스의 주요한 비즈니스를 수행하고 그 결과 데이터를 리턴하는 역할을 하는 컴포넌트이다. 서비스 수행부(110)는 리퀘스트 브로커(Request Broker)를 통해 메소드를 수행함으로써, XML 기반의 설정 파일에 선언되어 있는 비즈니스 객체의 메소드를 수행할 수 있다. 상기 리퀘스트 브로커는 메시징 플랫폼을 통해, 다층 구조의 서비스 환경에서 웹 개발자가 RPC(Remote Procedure Call) 프로토콜을 사용할 수 있게 한다.

[0048] 결과 생성부(120)는 상기 서비스 수행부(110)의 상기 비즈니스 객체의 메소드 수행 결과를 XML 기반의 설정 파일에 선언된 결과 형식에 따라 다양하게 출력할 수 있다. 예를 들어, 결과 생성부(120)는 상기 수행 결과를 템플릿으로 생성한 후 출력하는 템플릿(template), 상기 수행 결과를 지정된 URL로 리다이렉트하는 리다이렉트(redirect), 상기 수행 결과를 지정된 URL로 포워드 하는 포워드(forward) 및 상기 수행 결과를 페이지 없이 직접 출력하는 다이렉트(direct) 등의 다양한 응답 결과의 종류를 제공하여, 상기 비즈니스 객체의 메소드 수행 결과를 XML 기반의 설정 파일에 선언된 결과 형식 별로 적절하게 출력할 수 있다.

[0049] 단일 리소스 액세서(130)는 USF(1000)에서 사용하는 다양한 자원에 대한 접근 및 사용 인터페이스를 제공할 수 있다. 예를 들어, 단일 리소스 액세서(130)는 오픈 소스 프레임워크와의 연동이 가능하며 전사 표준인 RPC 프로토콜 환경, 메시징 플랫폼 환경 및 로컬을 지원하는 리퀘스트 브로커(RequestBroker)를 통하여 서비스 수행부(110) 또는 결과 생성부(120)와 연동이 가능하다. 단일 리소스 액세서(130)는 외부에 노출된 비즈니스 객체에 RPC 프로토콜 환경에서는 Java/C++ 용 RPC 프로토콜 커넥터를 통해서 접근할 수 있으며, 메시징 플랫폼 환경에서는 Java/C++ 리퀘스트 브로커(Request Broker)를 통해서 접근할 수 있다.

[0050] 예외 상황 핸들러(140)는, 비즈니스 객체의 메소드가 비정상적으로 수행되거나, 예외 상황이 발생한 경우에, 이에 대한 디버깅 정보를 관리할 수 있다.

[0051] 도 5는 본 발명의 일 실시예에 따른 USF(1000)의 스크립트 콤포넌트(200)의 세부 구성도이다.

[0052] 도 5에 도시된 바와 같이, 본 발명의 일 실시예에 따른 스크립트 콤포넌트(200)는 스크립트 엔진(210) 및 오픈 소스 템플릿 엔진(220)을 포함한다.

[0053] 스크립트 엔진(210)은 자바스크립트에 기초하여 비즈니스 객체의 메소드의 수행 전에 XML 기반의 설정 파일에 정의되어 있는, 사용자 입력 값의 전처리를 수행할 수 있고, 상기 비즈니스 객체의 메소드의 수행 후에 상기 수행 결과를 가공하는 후처리를 수행할 수 있다. 또한, 스크립트 엔진(210)은 유저 인터페이스 객체를 동적으로 배치하기 위해 유저 인터페이스 객체를 제어할 수 있다.

[0054] 오픈 소스 템플릿 엔진(220)은 유저 인터페이스 객체를 HTML 템플릿으로 관리하고, 런타임 시에 유저 인터페이스 객체를 비즈니스 객체에 연결하여, 최종 결과물을 동적으로 구성하는 역할을 할 수 있다. 또한, USF(1000)가 템플릿 지시어(Template Directive)를 정의한 경우, 정의된 지시어는 오픈 소스 템플릿 엔진(220)이 제공하는 인터페이스에 의해 런타임 시에 동적으로 반영될 수 있다.

[0055] 도 6은 본 발명의 일 실시예에 따른 USF(1000)의 코어 콤포넌트(300)의 세부 구성도이다.

[0056] 도 6에 도시된 바와 같이, 코어 콤포넌트(300)는 사용자 요청 수신부(310), 사용자 요청 수행부(320), 런타임 구성 생성부(330), 템플릿 로딩부(340) 및 구성 정보 초기화부(350)을 포함한다.

[0057] 사용자 요청 수신부(310)는 웹 브라우저를 통해서 사용자의 요청을 수신할 수 있다. 상기 수신된 사용자 요청은 웹 애플리케이션 서버에서 HttpServletRequest 객체로 변환되어 사용자 요청 수신부(310)에 전달될 수 있다. 사용자 요청 수신부(310)는 상기 수신된 사용자 요청이 USF(1000)에서 처리가 가능한 요청인지를 확인하고, USF(1000)에서 처리할 수 있는 요청인 경우, 사용자 요청 수행부(320)를 호출하여 사용자 요청을 수행하게 할 수 있다.

[0058] 사용자 요청 수행부(320)는 상기 사용자 요청 수신부(310)에서 수신된 사용자 요청이 USF(1000)에서 처리할 수 있는 요청인 경우, 이를 수행하는 역할을 한다. 사용자 요청 수행부(320)는 런타임 구성 생성부(330)를 통하여 XML 기반의 동적 설정 파일에 정의되어 있는 수행 모델을 독출하여, 사용자 요청을 수행할 수 있다.

[0059] 런타임 구성 생성부(330)는 XML 기반의 동적 설정 파일에 정의되어 있는 사용자 명령 정보를 독출하여, 상기 사용자 요청 수행부(320)에서 필요한 수행 모델을 생성할 수 있다.

[0060] 템플릿 로딩부(340)는 템플릿 메소드(Template Method)를 로딩하고, 이를 초기화 하는 역할을 수행할 수 있다. 예를 들어, 템플릿 로딩부(340)는 USF(1000)에서 기본적으로 사용하는 오픈 소스 템플릿 엔진 템플릿 메소드 또는 사용자가 개발한 템플릿 메소드를 로딩하고, 초기화할 수 있다.

[0061] 구성 정보 초기화부(350)는 USF(1000)가 비즈니스 객체 및 유저 인터페이스 객체를 생성하기 전에, 이전에 설정되어 있던 구성 정보를 독출한 후, 이를 초기화 하는 역할을 수행할 수 있다.

[0062] 도 7은 본 발명의 일 실시예에 따른 USF(1000)에서 수행되는 유저 인터페이스 객체와 비즈니스 객체의 통합 작업의 흐름도이다.

[0063] 동작(501)에서, 구성 정보 초기화부(350)는 웹 서버에 등록되어 있는 USF(1000)의 설정 정보 및 구성 정보를 독출한 후, 이를 초기화 할수 있다.

[0064] 동작(502)에서는 템플릿 로딩부(340)가 XML 상에 설정되어 있는 USF(1000)의 설정 파일에 등록된 템플릿 메소드 플러그인(Plug-in)을 실행시킬 수 있다. 상기 템플릿 메소드 플러그인은 USF(1000)에서 사용할 수 있는 템플릿 메소드를 읽어오고, 캐싱(Cache)할 수 있다.

[0065] 동작(503)에서, 스크립트 엔진(210)들을 관리하고 제어할 수 있는 스크립트 엔진 관리부가 상기 USF(1000)의 설정 파일을 독출하고, 스크립트 엔진(210)의 초기화 프로세스를 준비할 수 있다.

[0066] 동작(504)에서, 스크립트 엔진 관리부는 스크립트 엔진(210)을 초기화한 후, 유저 인터페이스 객체를 제어하여 유저 인터페이스 객체를 동적으로 배치하기 위한 준비를 할 수 있다. 예를 들어, 스크립트 엔진 관리부는 웹 애플리케이션 콘텍스트 초기화 시에, 사용자 요청 수신부(310)에 입력된 파라미터 중 'enable-scripting'의 값이 'true'이면 'custom-script-dir'의 값을 기준으로 '.js' 파일을 로딩할 수 있다. 이 후, 스크립트 파일의 확장자를 보고 스크립트 엔진(210)의 타입을 구분한 뒤, 클래스패스(classpath)에 로딩된 스크립트 엔진 라이브러리에서 스크립트 엔진 팩토리(Script Engine Factory) 클래스의 리스트를 가져와, 스크립트 엔진 팩토리 별로 스크립트 엔진(210)을 초기화할 수 있다.

[0067] 동작(505)에서, 사용자가 웹 브라우저를 통해 USF(1000)에 비즈니스 객체의 생성 요청을 하면, 사용자 요청 수신부(310)는 웹 브라우저를 통해서 사용자 요청을 수신할 수 있다. 상기 수신된 사용자 요청은 웹 애플리케이션 서버에서 HttpServletRequest) 객체로 변환되어 사용자 요청 수신부(310)에 전달될 수 있다. 사용자 요청 수신부(310)는 상기 수신된 사용자 요청이 USF(1000)에서 처리가 가능한 요청인지를 확인하고, USF(1000)에서 처리할 수 있는 요청인 경우, 사용자 요청 수행부(320)를 호출하여 사용자 요청을 수행하게 할 수 있다.

[0068] 동작(506)에서 사용자 요청 수행부(320)는 상기 사용자 요청 수신부(310)에서 수신된 사용자 요청이 USF(1000)에서 처리할 수 있는 요청인 경우, 이를 수행할 수 있다. 런타임 구성 생성부(330)가 XML 기반의 동적 설정 파일에 정의되어 있는 사용자 명령 정보를 독출하여, 상기 사용자 요청 수행부(320)에서 필요한 수행 모델을 생성하면, 상기 사용자 요청 수행부(320)는 상기 런타임 구성 생성부(330)를 통하여 상기 수행 모델을 독출하여, 사용자 요청을 수행할 수 있다.

[0069] 동작(507)부터 동작(511)까지는 사용자 요청 수행부(320)가 XML 기반의 설정 파일에 정의되어 있는 수행 모델 정보를 독출하여, 사용자가 요청한 비즈니스 객체를 생성하는 동작의 흐름이다.

[0070] 동작(507)은 서비스 수행부(110)가 비즈니스 객체를 생성하는 본 절차를 수행하기 전에, 사용자에 의해 정의된 전처리 작업이 있는 경우에, 스크립트 콤포넌트(200)가 자바스크립트(Javascript)에 기초하여, 이를 수행하는 동작이다.

[0071] 동작(508)은 서비스 수행부(110)가 사용자 요청에 따라, XML 기반의 설정 파일에 기초하여, 비즈니스 객체의 메소드를 실행하는 동작이다. 서비스 수행부(110)는 리퀘스트 브로커(Request Broker)를 통해 메소드를 실행함으로써, XML 상에서 선언되어 있는 비즈니스 객체의 메소드를 수행할 수 있다. 또한, 동작(508)에서 서비스 수행부(110)가 사용자 요청을 수행하는 과정에서, 리퀘스트 브로커는 메시징 플랫폼을 통해, 서비스 수행부(110)가 다층 구조의 서비스 환경에서 RPC(Remote Procedure Call) 프로토콜을 사용하게 할 수 있다.

[0072] 다만, 상기 동작(508)에서 비즈니스 객체의 메소드가 비정상적으로 실행되거나 예외 상황이 발생한 경우에, 예외 상황 핸들러(140)는, 이에 대한 디버깅 정보를 관리할 수 있다.

[0073] 동작(509)에서는, 상기 동작(508)에서 비즈니스 객체의 메소드가 정상적으로 실행된 경우, 결과 생성부(120)가 서비스 수행부(110)의 상기 비즈니스 객체의 메소드 실행 결과를 XML 기반의 설정 파일에 선언된 결과 형식에 따라 다양하게 출력할 수 있다. 결과의 종류는 전술한 바와 같이, 템플릿(template), 리다이렉트(redirect), 포워드(forward), 다이렉트(direct) 등의 다양한 응답 일 수 있다. 또한, 상기 비즈니스 객체의 수행 결과는 HTML 형태로 웹 브라우저를 통해 사용자 및 웹 개발자에게 제공될 수 있다.

[0074] 동작(510)에서, 결과 생성부(120)는 비즈니스 객체의 메소드 실행 결과를 스크립트 엔진(210)으로 전달하고, 스크립트 엔진(210)은 비즈니스 객체와 유저 인터페이스 객체를 통합하기 위해, 상기 전달받은 비즈니스 객체의 데이터와 유저 인터페이스 객체를 제어할 수 있다. 또한, 상기 비즈니스 객체 및 상기 유저 인터페이스 객체는 오픈 소스 템플릿 엔진(220)에 의해 런타임 시에 동적으로 결합되어 HTML 웹 페이지로서 제공될 수 있다.

[0075] 동작(511)은 서비스 수행부(110)에 의해 비즈니스 객체의 메소드가 정상적으로 실행되고, 결과 생성부(120)에 의해 실행 결과가 정상적으로 처리된 경우에, 사용자에 의해 정의된 후처리 작업이 있으면, 스크립트 콤포넌트(200)가 자바스크립트(Javascript)에 기초하여, 이를 수행하는 동작이다.

[0076] 동작(512)에서는, 상기 동작(509)에서 결과 생성부(120)에 의해 출력된 비즈니스 객체의 메소드 실행 결과를 런타임 엔진(100)이 사용자에게 전달할 수 있다. 런타임 엔진(100)은 상기 비즈니스 객체의 메소드 실행 결과를 HttpServlet Response 형태로 변환하여 HTML 웹 페이지로서 사용자에게 전달할 수 있으며, 이는 HTML 형태로 웹

브라우저를 통해 전달될 수 있다.

[0077] 본 발명의 일 실시예는 컴퓨터에 의해 실행되는 프로그램 모듈과 같은 컴퓨터에 의해 실행가능한 명령어를 포함하는 기록 매체의 형태로도 구현될 수 있다. 컴퓨터 판독 가능 매체는 컴퓨터에 의해 액세스될 수 있는 임의의 가용 매체일 수 있고, 휘발성 및 비휘발성 매체, 분리형 및 비분리형 매체를 모두 포함한다. 또한, 컴퓨터 판독 가능 매체는 컴퓨터 저장 매체 및 통신 매체를 모두 포함할 수 있다. 컴퓨터 저장 매체는 컴퓨터 판독가능 명령어, 데이터 구조, 프로그램 모듈 또는 기타 데이터와 같은 정보의 저장을 위한 임의의 방법 또는 기술로 구현된 휘발성 및 비휘발성, 분리형 및 비분리형 매체를 모두 포함한다. 통신 매체는 전형적으로 컴퓨터 판독가능 명령어, 데이터 구조, 프로그램 모듈, 또는 반송파와 같은 변조된 데이터 신호의 기타 데이터, 또는 기타 전송 메커니즘을 포함하며, 임의의 정보 전달 매체를 포함한다.

[0078] 전술한 본 발명의 설명은 예시를 위한 것이며, 본 발명이 속하는 기술분야의 통상의 지식을 가진 자는 본 발명의 기술적 사상이나 필수적인 특징을 변경하지 않고서 다른 구체적인 형태로 쉽게 변형이 가능하다는 것을 이해할 수 있을 것이다. 그러므로 이상에서 기술한 실시예들은 모든 면에서 예시적인 것이며 한정적이 아닌 것으로 이해해야만 한다. 예를 들어, 단일형으로 설명되어 있는 각 구성 요소는 분산되어 실시될 수도 있으며, 마찬가지로 분산된 것으로 설명되어 있는 구성 요소들도 결합된 형태로 실시될 수 있다.

[0079] 본 발명의 범위는 상기 상세한 설명보다는 후술하는 특허청구범위에 의하여 나타내어지며, 특허청구범위의 의미 및 범위 그리고 그 균등 개념으로부터 도출되는 모든 변경 또는 변형된 형태가 본 발명의 범위에 포함되는 것으로 해석되어야 한다.

도면의 간단한 설명

[0080] 도 1은 종래의 일반적인 웹 서비스의 유저 인터페이스 개발 프로세스를 도시한 도면.

[0081] 도 2는 본 발명의 일 실시예에 따른 웹 서비스의 유저 인터페이스 개발 프로세스를 도시한 도면.

[0082] 도 3은 본 발명의 일 실시예에 따른 USF(1000)의 세부 구성도.

[0083] 도 4는 본 발명의 일 실시예에 따른 USF(1000)의 런타임 엔진(100)의 세부 구성도.

[0084] 도 5는 본 발명의 일 실시예에 따른 USF(1000)의 스크립트 콤포넌트(200)의 세부 구성도.

[0085] 도 6은 본 발명의 일 실시예에 따른 USF(1000)의 코어 콤포넌트(300)의 세부 구성도.

[0086] 도 7은 본 발명의 일 실시예에 따른 USF(1000)에서 수행되는 유저 인터페이스 객체와 비즈니스 객체의 통합 작업의 흐름도.

[0087] *도면의 주요 부분에 대한 부호 설명

[0088] 1000: 유저 인터페이스 서버 프레임워크

[0089] 100: 런타임 엔진 200: 스크립트 콤포넌트

[0090] 300: 코어 콤포넌트 110: 서비스 수행부

[0091] 120: 결과 생성부 130: 단일 리소스 액세서

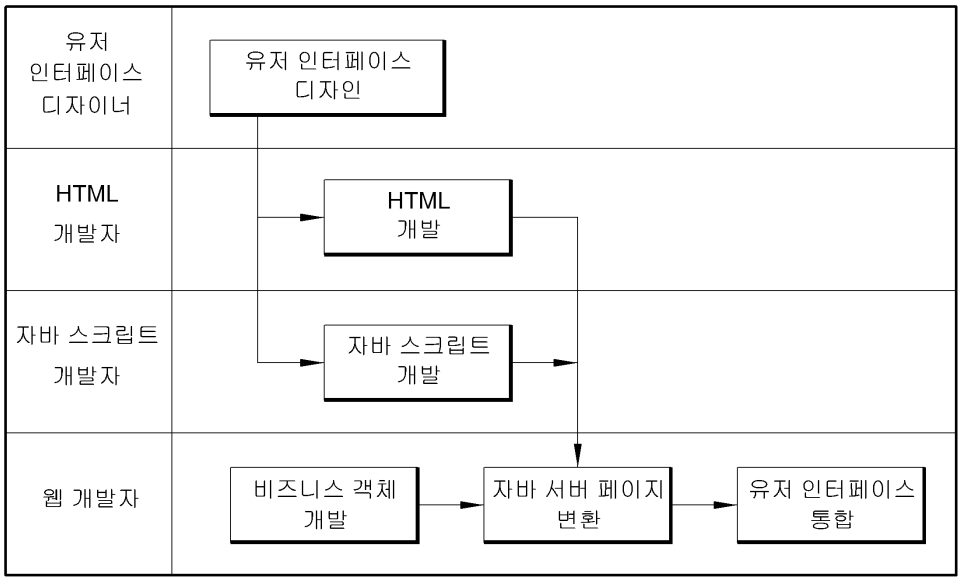
[0092] 140: 예외 상황 핸들러 210: 스크립트 엔진

[0093] 220: 오픈 소스 템플릿 엔진 310: 사용자 요청 수신부

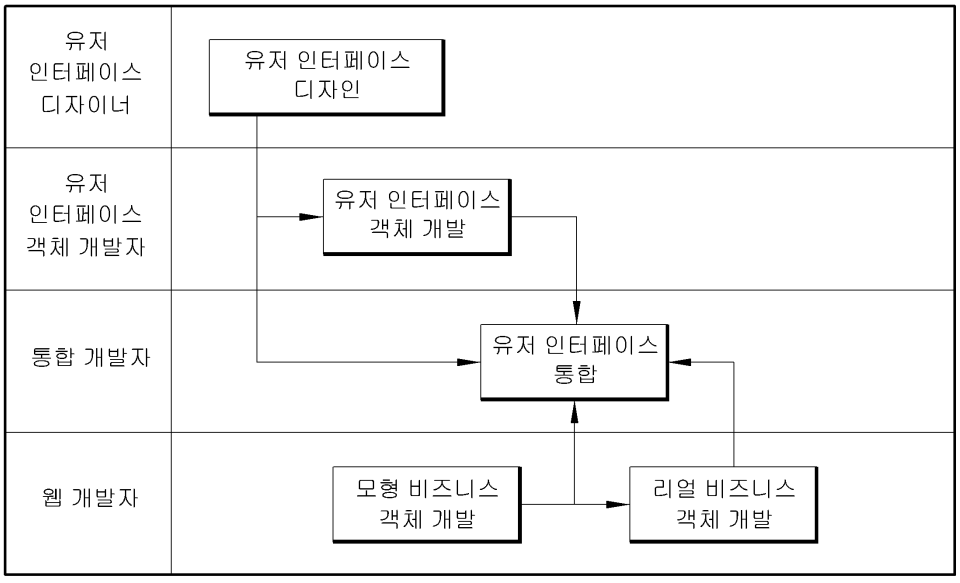
[0094] 320: 사용자 요청 수행부 330: 런타임 구성 생성부

도면

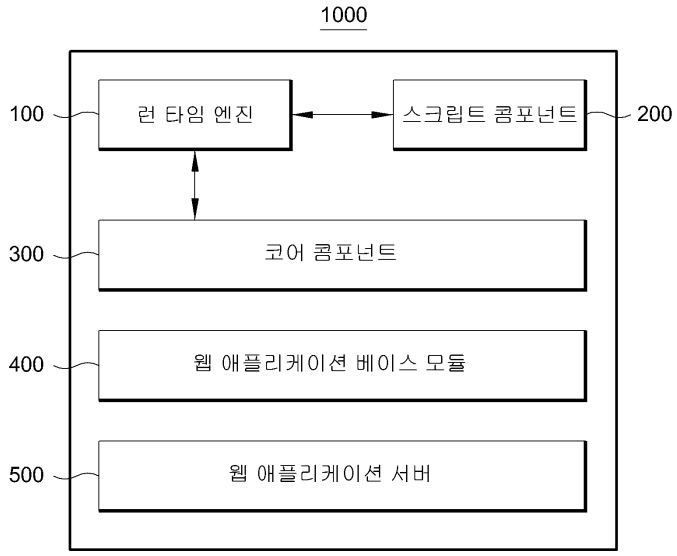
도면1


유저
인터페이스
디자이너
유저 인터페이스
디자인
HTML
개발자
HTML
개발
자바 스크립트
개발자
자바 스크립트
개발
웹 개발자
비즈니스 객체
개발
자바 서버 페이지
변환
유저 인터페이스
통합

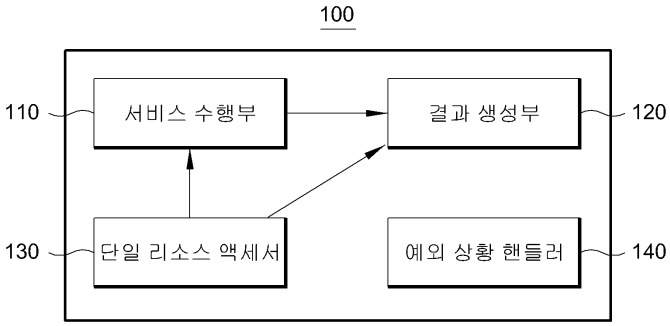
도면2


유저
인터페이스
디자이너
유저 인터페이스
디자인
유저
인터페이스
객체 개발자
유저 인터페이스
객체 개발
통합 개발자
유저 인터페이스
통합
웹 개발자
모형 비즈니스
객체 개발
리얼 비즈니스
객체 개발

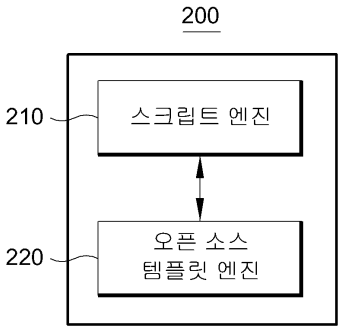
도면3


1000
100
런 타임 엔진
스크립트 콤포넌트
200
300
코어 콤포넌트
400
웹 애플리케이션 베이스 모듈
500
웹 애플리케이션 서버

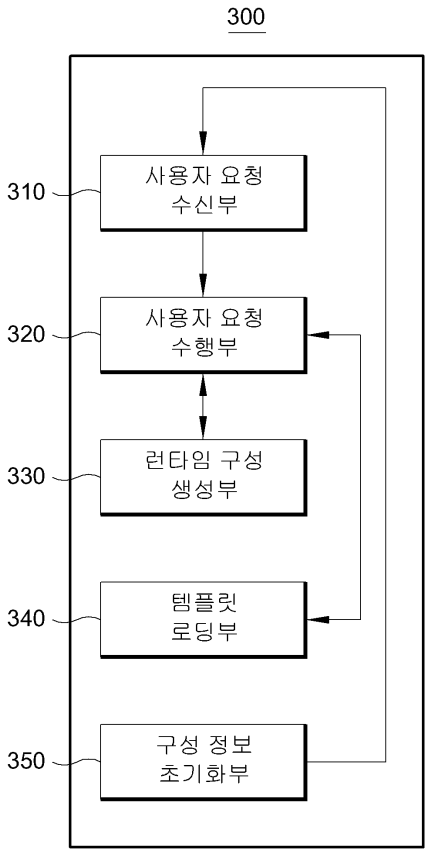
도면4


100
110
서비스 수행부
결과 생성부
120
130
단일 리소스 액세서
예외 상황 핸들러
140

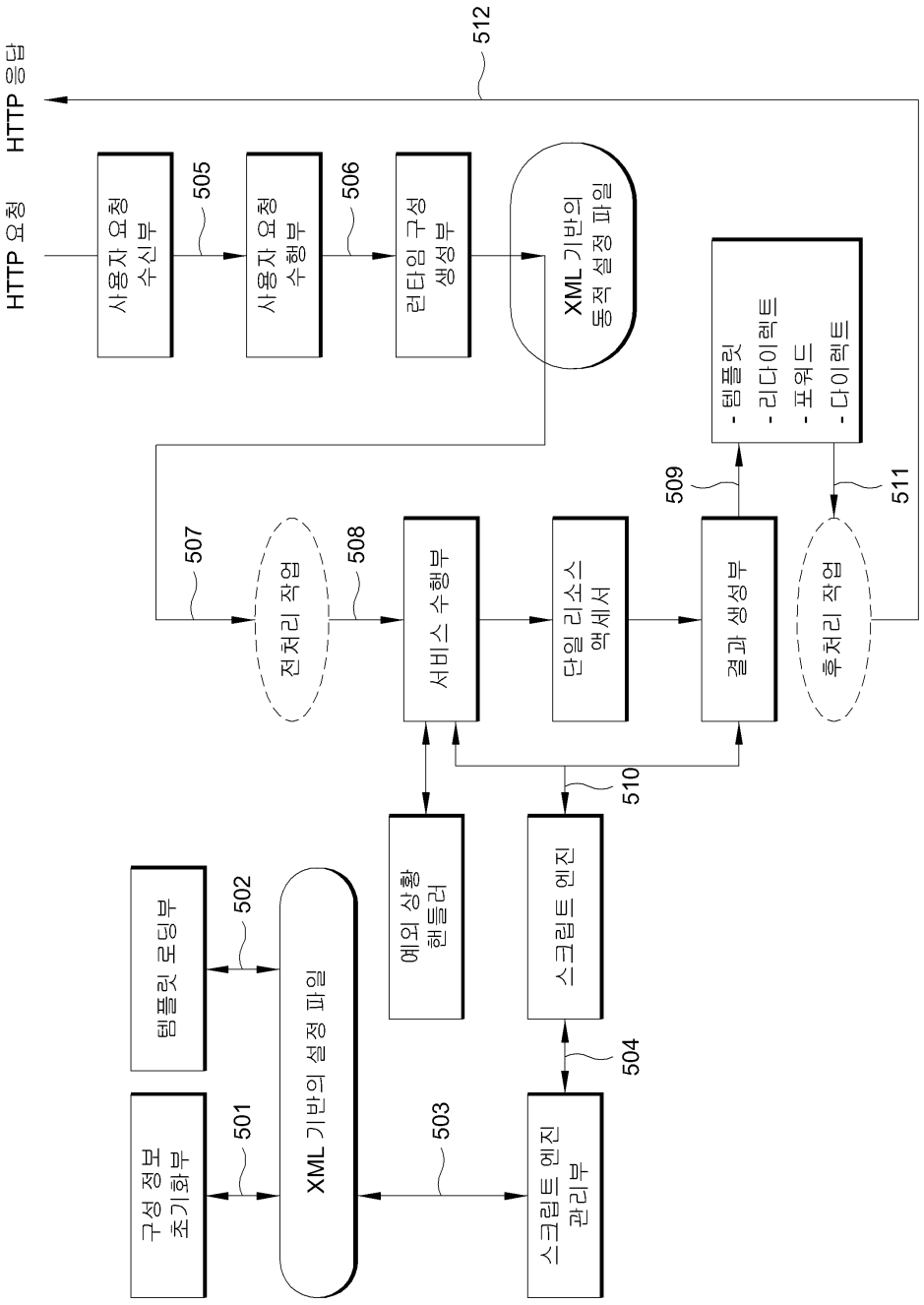
도면5


200
210
스크립트 엔진
220
오픈 소스
템플릿 엔진

도면6


300
310
사용자 요청
수신부
320
사용자 요청
수행부
330
런타임 구성
생성부
340
템플릿
로딩부
350
구성 정보
초기화부

도면7


HTTP 요청
HTTP 응답
512
사용자 요청 수신부
505
사용자 요청 수행부
506
런타임 구성 생성부
XML 기반의 동적 설정 파일
- 템플릿
- 리다이렉트
- 포워드
- 다이렉트
509
511
507
전처리 작업
508
서비스 수행부
단일 리소스 액세서
결과 생성부
후처리 작업
510
템플릿 로딩부
502
예외 상황 핸들러
스크립트 엔진
504
구성 정보 초기화부
501
XML 기반의 설정 파일
503
스크립트 엔진 관리부