(19) 대한민국특허청(KR)
(12) 등록특허공보(B1)

(45) 공고일자 2017년05월08일
(11) 등록번호 10-1733833
(24) 등록일자 2017년04월28일

(51) 국제특허분류(Int. Cl.)
G06F 9/06 (2006.01) *G06F 9/44* (2006.01)
(21) 출원번호 10-2012-7026789
(22) 출원일자(국제) 2011년03월25일
심사청구일자 2016년02월29일
(85) 번역문제출일자 2012년10월12일
(65) 공개번호 10-2013-0060175
(43) 공개일자 2013년06월07일
(86) 국제출원번호 PCT/US2011/030068
(87) 국제공개번호 WO 2011/129989
국제공개일자 2011년10월20일
(30) 우선권주장
12/760,565 2010년04월15일 미국(US)
(56) 선행기술조사문헌
US20100023547 A1
US20020144233 A1
US20090319981 A1
US20090326687 A1

(73) 특허권자
마이크로소프트 테크놀로지 라이센싱, 엘엘씨
미국 워싱턴주 (우편번호 : 98052) 레드몬드 원 마이크로소프트 웨이
(72) 발명자
비코브 이브구에니 엔
미국 워싱턴주 98052-6399 레드몬드 원 마이크로소프트 웨이 엘씨에이 - 인터내셔널 페이턴츠 마이크로소프트 코포레이션
핀디크 페리트
미국 워싱턴주 98052-6399 레드몬드 원 마이크로소프트 웨이 엘씨에이 - 인터내셔널 페이턴츠 마이크로소프트 코포레이션
(뒷면에 계속)
(74) 대리인
제일특허법인

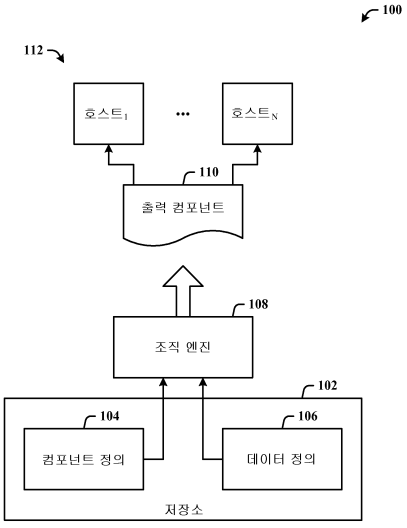
전체 청구항 수 : 총 20 항 심사관 : 김경완

(54) 발명의 명칭 **플랫폼 독립적 프리젠테이션 조직**

(57) 요 약

플랫폼에 독립적인 구성 유도 프리젠테이션 조직 엔진(platform independent, configuration driven, presentation composition engine)을 포함하는 아키텍처. 데이터 계약에 기초하여 멀티플랫폼 사용자 경험(UX)의 동적인 생성을 허용하는 조직 엔진. 조직에 의하여, 사용자는 부분, 상호작용 및 상호작용과 부분 사이의 제한 뿐만 아니라 서로에 대한 배치도 선택할 수 있다. Ux는 특정 데이터 클래스에 타겟된 컴포넌트로부터 동적으로 조직된다. 런타임에, 플랫폼 독립적인 컴포넌트 구현이 조직 호스트의 실행 플랫폼에 기초하여 엔진에 의해 자동으로 선택된다. 사용자는, 많은 플랫폼에서 동작할 수 있는 광범위한 데이터 소스에 액세스하는 광범위한 프리젠테이션 위젯으로부터 조직함으로써, 코드를 쓰지 않고 UX를 생성하거나 커스터마이즈할 수 있다. 조직은 데이터 클래스 및 프리젠테이션 유형 양자에 타겟될 수 있고 사전정의되거나 생성될 수 있다.

대 표 도



(72) 발명자

벤슨 라이언 에스

미국 워싱턴주 98052-6399 레드몬드 원 마이크로소프트 웨이 엘씨에이 - 인터내셔널 페이턴츠 마이크로소프트 코포레이션

오트리스코 볼로디미르 브이

미국 워싱턴주 98052-6399 레드몬드 원 마이크로소프트 웨이 엘씨에이 - 인터내셔널 페이턴츠 마이크로소프트 코포레이션

명 세 서

청구범위

청구항 1

프로세서에 의해 실행되는 실행가능한 명령을 저장하는 컴퓨터 판독가능 메모리를 갖는 컴퓨터로 구현된 시각화 시스템(computer-implemented visualization system)으로서,

하나 이상의 사용자 경험과 연관된 컴포넌트 및 데이터에 대한 하나 이상의 컴포넌트 정의(component definition) 및 데이터 정의(data definition)를 포함하는 정의들을 저장하도록 구성되는 저장소와,

실행 환경에서의 조직 호스트의 실행 플랫폼, 상기 실행 환경에서의 시각화 호스트의 사용자 경험의 타겟 데이터 유형, 및 저장 정의에 기초하여 출력 컴포넌트의 인스턴스를 자동으로 그리고 선언적으로(declaratively) 환경 런타임시 조직하도록 구성되는 조직 엔진(composition engine)을 포함하되,

상기 출력 컴포넌트는 상기 실행 환경에서의 상기 시각화 호스트의 상기 사용자 경험에 특유하며, 상기 출력 컴포넌트는 베이스 컴포넌트를 위한 컨테이너인 플랫폼 독립적 컨테이너 컴포넌트이고, 상기 베이스 컴포넌트는, 상기 실행 환경에서의 상기 조직 호스트의 실행 플랫폼, 상기 사용자 경험의 타겟 데이터 유형, 및 상기 저장 정의 중 적어도 하나에 기초하여 복수의 컴포넌트 정의로부터 선택되고,

상기 조직 엔진은, 요청된 컴포넌트가 발견되지 않는 경우에 상기 컨테이너 컴포넌트를 생성하고 상기 베이스 컴포넌트의 연관 데이터 유형 속성 중 하나 이상을 이용하여 상기 컨테이너 컴포넌트를 로딩하고 상기 컨테이너 컴포넌트를 상기 출력 컴포넌트로서 출력하도록 더 구성되는

컴퓨터로 구현된 시각화 시스템.

청구항 2

제 1 항에 있어서,

상기 출력 컴포넌트는 상기 베이스 컴포넌트, 상기 컨테이너 컴포넌트 또는 베이스 및 컨테이너 컴포넌트의 조합을 포함하는

컴퓨터로 구현된 시각화 시스템.

청구항 3

제 1 항에 있어서,

상기 컨테이너 컴포넌트는 커스텀 구현을 갖지 않는

컴퓨터로 구현된 시각화 시스템.

청구항 4

제 1 항에 있어서,

상기 타겟 데이터 유형에 기초하여 컴포넌트가 검색되는 컴포넌트 레지스트리를 더 포함하는

컴퓨터로 구현된 시각화 시스템.

청구항 5

제 1 항에 있어서,

상기 출력 컴포넌트는 데이터 맥락(data context)에 기초하여 조직되는

컴퓨터로 구현된 시각화 시스템.

청구항 6

제 1 항에 있어서,

상기 출력 컴포넌트는, 차일드 컴포넌트를 링크하기 위해, 데이터 맥락 요소에 연관 컴포넌트 속성을 바인딩하는

컴퓨터로 구현된 시각화 시스템.

청구항 7

제 1 항에 있어서,

상기 조직 엔진은 비관련 데이터 맥락(unrelated data contexts)에서 출력 컴포넌트 사이의 데이터 교환을 가능하게 하는 글로벌 변수를 포함하는

컴퓨터로 구현된 시각화 시스템.

청구항 8

제 1 항에 있어서,

상기 시스템은 컴포넌트의 하나 이상의 차일드 컴포넌트에 프라이빗 변수를 부과하기 위해 상기 프라이빗 변수로 글로벌 변수를 오버라이드하도록 구성되는

컴퓨터로 구현된 시각화 시스템.

청구항 9

프로세서 및 메모리를 통해 실행가능한 컴퓨터로 구현된 시각화 방법으로서,

실행 환경에서 채용될 컴포넌트에 대한 요청을 수신하는 단계와,

상기 실행 환경에서의 조직 호스트의 실행 플랫폼 및 사용자 경험의 타겟 데이터 유형에 기초하여 상기 컴포넌트와 연관된 복수의 컴포넌트 정의 중 특정의 컴포넌트 정의를 선택하는 단계와,

상기 사용자 경험의 타겟 데이터 유형에 기초하여 상기 선택된 컴포넌트 정의에 대한 하나 이상의 데이터 정의를 선택하는 단계와,

상기 실행 환경에서의 상기 조직 호스트의 실행 플랫폼에 기초하여, 상기 실행 환경에서 상기 컴포넌트를 출력하기 위해 상기 컴포넌트 정의로 상기 하나 이상의 선택된 데이터 정의를 환경 런타임시 자동으로 조직하는 단계 - 상기 컴포넌트는 베이스 컴포넌트를 위한 컨테이너인 플랫폼 독립적 컨테이너 컴포넌트를 포함함 - 와,

요청된 컴포넌트가 발견되지 않는 경우에 상기 컨테이너 컴포넌트를 생성하는 단계와,

상기 베이스 컴포넌트의 연관 데이터 유형 속성 중 하나 이상을 이용하여 상기 컨테이너 컴포넌트를 로딩하는 단계와,

상기 컨테이너 컴포넌트를 상기 컴포넌트로서 출력하는 단계를 포함하는

컴퓨터로 구현된 시각화 방법.

청구항 10

제 9 항에 있어서,

상기 컴포넌트 정의가 발견되지 않는 경우에 상기 요청된 컴포넌트의 데이터 유형에 기초하여 상기 컴포넌트 정의를 검색하는 단계를 더 포함하는

컴퓨터로 구현된 시각화 방법.

청구항 11

제 9 항에 있어서,

요청된 상기 컴포넌트의 부재에 기초하여 커스텀 컴포넌트를 생성하는 단계를 더 포함하는

컴퓨터로 구현된 시각화 방법.

청구항 12

제 9 항에 있어서,

비관련 데이터 맥락 사이에서의 데이터 교환을 가능하게 하기 위해 상기 컴포넌트에 글로벌 변수를 적용하는 단계를 더 포함하는

컴퓨터로 구현된 시각화 방법.

청구항 13

제 9 항에 있어서,

상기 컴포넌트의 하나 이상의 차일드 컴포넌트에 프라이빗 변수를 부과하기 위해 상기 프라이빗 변수로 글로벌 변수를 오버라이드하는 단계를 더 포함하는

컴퓨터로 구현된 시각화 방법.

청구항 14

제 9 항에 있어서,

상기 컨테이너 컴포넌트는 커스텀 구현을 갖지 않는

컴퓨터로 구현된 시각화 방법.

청구항 15

제 9 항에 있어서,

상기 출력 컴포넌트는, 차일드 컴포넌트를 링크하기 위해, 데이터 맥락 요소에 연관 컴포넌트 속성을 바인딩하는

컴퓨터로 구현된 시각화 방법.

청구항 16

프로세서 및 메모리를 통해 실행가능한 컴퓨터로 구현된 시각화 방법으로서,

컴포넌트 실행 환경에 기초하여 컴포넌트에 대한 요청을 수신하는 단계와,

상기 실행 환경에서의 조직 호스트의 실행 플랫폼 및 사용자 경험의 타겟 데이터 유형에 기초하여 상기 컴포넌트와 연관된 복수의 컴포넌트 정의 중 특정의 컴포넌트 정의를 선택하는 단계와,

컴포넌트 정의가 발견되는 경우, 상기 사용자 경험의 타겟 데이터 유형에 기초하여 상기 선택된 컴포넌트 정의에 대한 하나 이상의 데이터 정의를 선택하는 단계와,

상기 컴포넌트 정의가 발견되지 않는 경우, 상기 요청된 컴포넌트와 관련된 상기 타겟 데이터 유형에 기초하여 커스텀 컴포넌트를 생성하는 단계와,

비관련 데이터 맥락 사이에서의 데이터 교환을 가능하게 하기 위해 상기 컴포넌트 또는 상기 커스텀 컴포넌트 또는 양자 모두에 글로벌 변수를 적용하는 단계와,

상기 실행 환경에서의 상기 조직 호스트의 실행 플랫폼에 기초하여, 상기 실행 환경에서 상기 컴포넌트를 출력하기 위해 상기 컴포넌트 정의로 상기 하나 이상의 데이터 정의를 환경 런타임시 자동으로 조직하는 단계 – 상기 컴포넌트는 베이스 컴포넌트를 위한 컨테이너인 플랫폼 독립적 컨테이너 컴포넌트를 포함함 – 와,

상기 요청된 컴포넌트가 발견되지 않는 경우에 상기 컨테이너 컴포넌트를 생성하는 단계와,

상기 베이스 컴포넌트의 연관 데이터 유형 속성 중 하나 이상을 이용하여 상기 컨테이너 컴포넌트를 로딩하는 단계와,

상기 컨테이너 컴포넌트를 상기 컴포넌트로서 출력하는 단계를 포함하는

컴퓨터로 구현된 시각화 방법.

청구항 17

제 16 항에 있어서,

상기 컴포넌트의 차일드 컴포넌트에 프라이빗 변수를 부과하기 위해 상기 프라이빗 변수로 글로벌 변수를 오버라이드하는 단계를 더 포함하는

컴퓨터로 구현된 시각화 방법.

청구항 18

제 16 항에 있어서,

상기 컴포넌트로 전달될 데이터를 파라미터 노드를 통해 정의하는 단계를 더 포함하는

컴퓨터로 구현된 시각화 방법.

청구항 19

제 16 항에 있어서,

차일드 컴포넌트로의 속성의 바인딩과 데이터 컴포넌트로의 상기 속성의 바인딩을 포함하는 부모 컴포넌트를 조직하는 단계를 더 포함하는

컴퓨터로 구현된 시각화 방법.

청구항 20

제 16 항에 있어서,

상기 컨테이너 컴포넌트는 커스텀 구현을 갖지 않는

컴퓨터로 구현된 시각화 방법.

발명의 설명

배 경 기 술

[0001] 사용자 경험(user experience)(UX)의 품질은 UX가 사용자 기대와 얼마나 잘 정합되는지에 기초한다. 설계자들은 많은 데이터 유형, 많은 데이터 소소, 그리고 많은 UX 플랫폼을 다루어야 하므로, 특정 UX 플랫폼에 대해 데이터 소스로부터 특정 데이터를 소비하는 특정인을 위한 프리젠테이션 코드를 기록하는 것 또는 임의의 단일 특정인의 필요를 만족시키지 않는 광범위하게 타겟된 UX를 제공하는 것을 포함하는 매력적이지 않은 방식들 중에서 선택하여야 한다.

[0002] 예를 들어, HTML(hypertext markup language), XAML(extensible application markup language), 및 XSLT(extensible stylesheet language transformations)와 같은 기존 UX 조직(composition) 시스템은 마크업 코드가 특정 플랫폼을 위해 개발되도록 설계된다. 개발자가 수개의 플랫폼에서 코드가 작동하기를 원한다면, 플랫폼 차이를 처리하기 위한 커스텀 로직이 코드에 내장되어야 한다. 또한, 기존 UX 조직 시스템은 특정 프레젠테이션이 모든 데이터 인터페이스 요소에 대해 명시적으로 정의될 것을 요구한다. 요소가 나타내는 기저 데이터 구조에 기초하여 UX 요소의 동적 생성을 가능하게 하는 기능은 제한적이거나 존재하지 않는데, 특히 데이터 구조가 복잡하고/거나 상속가능(inheritable)한 경우에 더욱 그렇다.

[0003] 이들 한계의 결과로, 대량 시장(예를 들어, 이메일)은 서비스되어왔을 수 있지만, 작은 사용자 커뮤니티(예를 들어, 익스체인지 관리자 또는 CRM(customer relationship management) 서비스 오너)는 제대로 서비스되지 않는다.

발명의 내용

과제의 해결 수단

[0004] 이하는 여기 개시된 일부 신규한 실시형태의 기본적인 이해를 제공하기 위해 단순화된 개요를 제시한다. 이 개요는 망라적인 개요는 아니고, 중요/핵심 요소를 식별하거나 그 범위를 획정하려는 것이 아니다. 유일한 목적은 이후에 제시될 더 상세한 설명의 전조로서 일부 개념을 단순화된 형태로 제시하는 것이다.

[0005] 개시된 아키텍처는 플랫폼 독립적인 구성-유도 프레젠테이션 조직 엔진(a platform independent configuration-driven presentation composition engine)을 포함한다. 조직 엔진은 데이터 계약에 기초하여 멀티플랫폼 사용자 경험(UX)의 동적인 생성을 허용한다. 조직에 의해, 사용자는 부분, 상호작용 및 부분과 상호작용 사이의 제한뿐만 아니라 서로에 대한 배치도 선택할 수 있다.

[0006] UX는 특정 데이터 클래스에 타겟된 컴포넌트로부터 동적으로 조직된다. 런타임에, 플랫폼 독립적인 컴포넌트 구현은 조직 호스트의 실행 플랫폼에 기초하여 엔진에 의해 자동으로 선택된다.

[0007] 개시된 아키텍처는 사용자가, 많은 플랫폼에서 작동하는 많은 데이터 소스에 액세스할 수 있는 다수의 프리젠테이션 위젯으로부터 조직함으로써, 코드를 기록할 필요 없이 UX를 생성하거나 커스터마이즈할 수 있게 한다. 조직은 데이터 클래스와 프리젠테이션 유형 모두에 타겟될 수 있고 미리 정의되거나 생성될 수 있다.

[0008] 전술한, 그리고 관련된 목적을 달성하기 위해, 여기서 특정 예시적인 태양이 다음의 설명 및 첨부된 도면과 관련하여 설명된다. 이들 태양은 여기 개시된 원리가 실시될 수 있는 다양한 방식을 나타내며, 모든 태양 및 그 균등물은 청구된 주제의 범위 내에 있는 것이다. 다른 이점 및 신규한 특징은 도면과 관련하여 다음의 상세한 설명을 고려할 때에 명확해 질 것이다.

도면의 간단한 설명

[0009] 도 1은 개시된 아키텍처에 따른 시각화 시스템을 도시한다.

도 2는 개시된 아키텍처에 따른 다른 시각화 시스템을 도시한다.

도 3은 조직 엔진에 의해 조직된 예시적인 조직을 도시한다.

도 4는 조직 시스템의 데이터 맥락 및 시각 베이스 컴포넌트를 포함하는 부모 컴포넌트를 도시한다.

도 5는 컴포넌트 정의를 도시한다.

도 6은 컴포넌트를 찾거나 선택하기 위한 컴포넌트 레지스트리를 도시한다.

도 7은 조직 엔진에서 변수의 사용을 나타내는 서술도(declarative diagram)를 도시한다.

도 8은 개시된 아키텍처에 따른 시각화 방법을 도시한다.

도 9는 도 8의 추가적 태양을 도시한다.

도 10은 다른 시각화 방법을 도시한다.

도 11은 도 10의 추가적 태양을 도시한다.

도 12는 조직 엔진에서 컴포넌트를 획득하는 방법을 도시한다.

도 13은 조직 엔진에서 컴포넌트를 획득하는 더 상세한 방법을 도시한다.

도 14는 개시된 아키텍처에 따른 조직을 실행하는 연산 시스템의 블록도를 도시한다.

발명을 실시하기 위한 구체적인 내용

[0010] 개시된 아키텍처는 프리젠테이션 조직 엔진이다. 조직 엔진은 사용자가 상이한 컴포넌트들 및 컴포넌트(들)의 조직(엔진의 출력 조직)을 "붙일 수 있도록"(조직할 수 있도록)하는 서비스의 세트로서 노출되는 제네릭 조직 프레임워크이다. 조직에 의해, 사용자는 부분, 상호작용 및 상호작용과 부분 사이의 제한 뿐만 아니라, 서로에 대한 부분의 배치도 선택할 수 있다. 엔진은 UI(user interface)와 비-UI 컴포넌트 모두에 대한 프레젠테이션-신경(presentation-neural) 프레임워크이다.

[0011] 컴포넌트는 조직 엔진에 대해 가장 작은, UI 선언(UI declaration)의 재사용가능한 빌딩 블록이고, 이는 이름에 의해 식별가능하고, 선택적으로는 데이터 유형에 타겟된다. 컴포넌트는 베이스 컴포넌트(단위(unit) 컴포넌트) 또는 컨테이너 컴포넌트(혼합(composite) 컴포넌트)일 수 있다. 데이터 맥락(data context)은 컴포넌트에 대한 타겟 데이터의 인스턴스이다. 달리 말하면, 데이터 맥락은 컴포넌트와 연관된 데이터를 나타내는 이름/값 쌍(name/value pair) 세트이다. 데이터 맥락 엔트리는 변경 통지를 지원하고, 조직은 변경을 개시하거나 및/또는 다른 조직에 의해 개시된 변경을 청취할 수 있다. 조직 엔진은 플랫폼에 독립적인 사용자 경험으로서 특정 호스트에 대한 컴포넌트를 조합한다. 시각화 호스트는 특정 플랫폼에 대한 조직의 실행 환경이다.

[0012] 조직은 데이터 클래스와 프리젠테이션 유형 모두에 타겟되고, 사전정의되거나 생성될 수 있다. 다수의 조직된 컴포넌트가 있을 수 있는 반면, 컴포넌트 연쇄는 구체적인 베이스 컴포넌트(예를 들어, 텍스트박스(TextBox) 제어, 데이터베이스 쿼리 컴포넌트 등)에서 종료된다.

[0013] 조직 엔진은 데이터 계약에 기초한 다수플랫폼 UX(user experience)의 동적인 생성을 허용한다. UX는 특정 데이터 클래스에 타겟된 컴포넌트로부터 동적으로 조직된다. 런타임에, 플랫폼 의존적인 컴포넌트 구현이 조직 호스트의 실행 플랫폼에 기초하여 엔진에 의해 자동으로 선택된다.

[0014] 이제 도면을 참조하는데, 도면에서 동일한 참조부호는 동일한 구성요소를 지칭하는데 사용된다. 다음의 설명에서, 설명의 목적으로, 완전한 이해를 제공하기 위해 많은 구체적인 세부사항이 제시된다. 그러나, 신규한 실시형태가 이들 구체적인 세부사항이 없이도 실시될 수 있음은 자명할 것이다. 다른 예에서, 설명을 용이하게 하기 위해 공지된 구조 및 장치가 블록도 형태로 도시된다. 청구된 주제의 사상 및 범위 내의 모든 변형, 균등물 및 대안을 포함하려는 의도이다.

[0015] 도 1은 개시된 아키텍처에 따른 시각화 시스템(100)을 도시한다. 시스템(100)은 사용자 경험과 연관된 컴포넌트와 데이터에 대한 컴포넌트 정의(104)와 데이터 정의(106)를 포함하는 저장 정의(store definitions)의 저장

소(102)를 포함한다. 컴포넌트 정의(104)는 베이스 컴포넌트, 컨테이너 컴포넌트, 그리고 베이스와 컨테이너 컴포넌트의 조직에 대한 정의를 포함할 수 있다. 이러한 방식으로, 컴포넌트의 기존 조직이 출력 컴포넌트(110)로의 동적인 선택 및 조직을 위해 용이하게 사용가능하다.

[0016] 조직 엔진(108)은 저장 정의에 기초하여 자동으로 그리고 선언적으로(declaratively) 출력 컴포넌트(110)의 인스턴스를 조직한다. 출력 컴포넌트는 상이한 호스트(112)의 시각화 호스트의 사용자 경험에 특유한 것이다.

[0017] 출력 컴포넌트(110)는 베이스 컴포넌트, 컨테이너 컴포넌트 또는 베이스와 컨테이너 컴포넌트의 조합을 포함한다. 출력 컴포넌트(110)는 사용자 경험의 타겟 데이터 유형에 기초하여 조직된다. 시스템(100)은 타겟 데이터 유형에 기초하여 컴포넌트가 검색되는 컴포넌트 레지스트리를 더 포함할 수 있다. 출력 컴포넌트는 연관 컴포넌트 속성을 데이터 맥락 요소에 바인딩하여 차일드 컴포넌트에 링크한다. 조직 엔진(108)은 출력 컴포넌트 사이의 데이터 교환을 비관련 데이터 맥락(unrelated data context)에서 가능하게 하는 글로벌 변수를 포함한다.

[0018] 도 2는 개시된 아키텍처에 따른 다른 시각화 시스템(200)을 도시한다. 시스템(200)은 도 1의 시스템(100)의 엔티티뿐만 아니라, 데이터 맥락(202), 개인화(personalization)(프라이빗(private)) 오버라이드(204) 및 컴포넌트 구현(206)을 포함한다. 출력 컴포넌트(110)는 기존 컴포넌트 정의가 아니라 데이터 맥락(202)에 기초하여 조직될 수 있다. 즉, 데이터에 기초하여, 순전히 맥락 데이터(202) (타겟 UX에서의 데이터의 인스턴스)에 기초하여 커스터마이즈된 컴포넌트가 생성되고 출력될 수 있다. 조직 엔진(108)은 글로벌 변수를 프라이빗 변수로 오버라이드(override)하기 위해 선택된 컴포넌트 정의로 조직된 개인화 오버라이드(204)를 채용한다.

[0019] 도 3은 조직 엔진에 의해 조직된 예시적인 조직(300)을 도시한다. 여기서, 조직(300)은 기초 컴포넌트(예를 들어, 스택패널(StackPane) 베이스 컴포넌트(302)) 및 컨테이너 컴포넌트(304)의 관점에서 설명된다. 여기서, 베이스 컴포넌트(302)는 2 개의 텍스트 박스 베이스 컴포넌트, 즉 텍스트 "ABC"를 보여주는 제1 텍스트 박스 베이스 컴포넌트와 텍스트 "DEF"를 보여주는 제2 텍스트 박스 베이스 컴포넌트를 포함한다. 베이스 컴포넌트(302)는 버튼 베이스 컴포넌트도 포함한다.

[0020] 베이스 컴포넌트는 특정 플랫폼에 타겟된 구체적인 구현, 조직 프로세스의 리프(leaf) 노드이고, 시각적 또는 비시각적일 수 있다. 이하는 베이스 컴포넌트 정의의 일예이다(컴포넌트 유형이 아니라 컴포넌트 유형의 관점에서).

```
      <ComponentType ID="EventQuery">
        <Parameters>
          <Parameter Name="Scope" Type="String"  />
          <Parameter Name="Output" Type="IEnumerable" />
         </Parameters>
      </ComponentType>
      <ComponentImplementation TypeId="EventQuery">
        <SupportedPlatforms>
          <Platform>WPF</Platform>
        </SupportedPlatforms>
        <Unit>
          <MefFactory>

<ContractName>Company.EnterpriseManagement.EventQuery</ContractName>
          </MefFactory>
          <Properties>
            <Property Name="QueryVerb" Direction="In">Events</Property>
            <Property Name="Scope"
Direction="In">$Parameter/Scope$</Property>
            <Property Name="Output"
Direction="Out">$Parameter/Output$</Property>
          </Properties>
        </Unit>
      </ComponentImplementation>
```

[0021]

[0022] MEF(managed extensibility framework) 팩토리(factory)는 등록된 UI 어셈블리로부터 대응하는 유형을 끌어오기 위해 MEF 런타임을 호출한다. MEF는 단순히 한 방식의 예시적인 구현임을 유의하라: 다른 팩토리 구현도 채용될 수 있다.

[0023] 컨테이너 컴포넌트(또한, 혼합 컴포넌트)는 베이스 컴포넌트(또한, 단위 컴포넌트)를 위한 컨테이너이고, 커스텀 구현을 갖지 않으며, 플랫폼에 독립적이다. 이하는 혼합 컴포넌트 정의의 예이다.

```
<ComponentImplementation TypeId="SampleComponent">
  <SupportedPlatforms>
    <Platform>All</Platform>
  </SupportedPlatforms>
  <Composite>
    <Variables>
      <Variable Id="abc" Type="String" />
    </Variables>
    <Component TypeId="SampleContainerComponent">
      <Parameter Id="Child1">
        <Component TypeId="SampleComponent1">
          <Parameter Id="Bla">$Variable/abc$</Parameter>
        </Component>
      </Parameter>
      <Parameter Id="Child2">
        <Component TypeId="SampleComponent2">
          <Parameter Id="Bla">$Variable/abc$</Parameter>
        </Component>
      </Parameter>
    </Component>
  </Composite>
</ComponentImplementation>
```

[0024]

[0025] 도 3에서, 컨테이너 컴포넌트(304)는 아래의 코드로 설명되는 바와 같이 베이스 컴포넌트를 결합하는데, 여기서 속성A(PropertyA)는 "ABC"이고 속성B(PropertyB)는 "DEF"이다.

```
<Component TypeId="StackPanel">
   <Parameter Name="Child">
<Component TypeId="Edit">
          <Target>$Target/propertyA$</Target>
</Component>
<Component TypeId="Edit">
          <Target>$Target/propertyB$</Target>
</Component>
<Component TypeId="Button">
</Component>
   </Parameter>
```

[0026]

[0027] 각각의 컴포넌트(베이스 또는 컨테이너)는 데이터 맥락(Data Context) 인스턴스로 뒷받침된다. 데이터 맥락은 속성 변경 통지 및 에러 설정 통지를 지원하는 키(스트링)-값(객체) 쌍 컬렉션이다. 컴포넌트는 그 속성을 데이터 맥락 요소에 바인딩하여 차일드 컴포넌트를 서로 링크할 수 있다.

[0028] 도 4는 조직 시스템의 데이터 맥락(402)(DataContext 요소) 및 시각적 베이스 컴포넌트(404)(예를 들어, 도 3의 StackPanel 베이스 컴포넌트(302))를 포함하는 부모 컴포넌트(400)를 도시한다. 여기서, 데이터 맥락(402)은 세 개의 속성, 속성A(PropertyA), 속성B(PropertyB), 속성C(PropertyC)를 포함하는데, 이들은 대응하는 값 "ABC", "DEF", 및 "XYZ"을 갖는다. 시각적 베이스 컴포넌트(404)는 대응하는 2-방향 데이터바인딩을 통한 속성A와 속성B로의 바인딩을 포함하고, 뷰 모델(408)의 다수의 데이터 컴포넌트(406)는 속성B 및 속성C로 바인딩된다. 달리 말해, 차일드 컴포넌트(텍스트 박스 베이스 컴포넌트)로의 속성의 바인딩 및 데이터 컴포넌트(406)로의 속성의 바인딩을 포함하는 부모 컴포넌트(400)가 조직된다.

[0029] 도 5는 컴포넌트 정의(500)를 도시한다. 컴포넌트 정의(500)는 2 부분을 포함한다: 유형 선언(502) 및 구현 정의(504). 각각의 컴포넌트는 이름, 그리고 선택적으로, 관련 컴포넌트를 찾는데(look up) 사용될 수 있는 타겟 유형 속성을 갖는다. 예시적인 유형 선언(컴포넌트 관점에서)은 다음과 같다:

```
  <ComponentType ID="SampleComponent" Target="String"
Accessibility="Internal">
```

[0030]

[0031] 컴포넌트는 또한 전달될 것으로 기대되는 데이터모양(datashape)(예를 들어, 스트링)을 정의하는 파라미터(Parameters) 서브노트를 포함할 수 있고, 아래와 같다(컴포넌트 관점):

```
<ComponentType ID="AnotherComposition">
  <Parameters>
    <Parameter Name="Parameter1" Type="String" />
    <Parameter Name="SelectedText" Type="String" BindingDirection="Both"
/>
  </Parameters>
```

[0032]

[0033] 도 6은 컴포넌트를 찾거나 선택하기 위한 컴포넌트 레지스트리(600)를 도시한다. 레지스트리(600)는 모든 정의된 컴포넌트와 컴포넌트 조직의 목록을 유지한다. 예를 들어, 베이스 컴포넌트(예를 들어, 도 3의 StackPnal/Button/Edit)는 TypeId 및 TargeType(타겟 데이터 유형)에 기초하여 검색될 수 있다(searched). 그

러면 출력은 TargetType 또는 TypeId 및 TargetType에 대응하는 베이스 컴포넌트이다.

[0034] 컴포넌트 상에서 속성을 설정하기 위해, "Parameter" 노드가 사용되는데, 다음 샘플 코드에서 보여지는 바와 같다.

[0035]
```
<Component Id="EventView">
<Parameter Id="Scope">Microsoft.SystemCenter.SqlDB </Parameter>
```

[0036] 이 경우에, 컴포넌트 "EventView"의 속성 "Scope"가 텍스트 "Company.SystemCenter.SqlDB"로 설정된다. 그러나, 많은 경우에, 파라미터는 정적(static)이지 않지만, 다른 요소에 바인딩된다. 일반적으로, 레퍼런스는 $<protocol>/<protocol-specific string> 형태이다.

[0037] 예를 들어, 2개의 컴포넌트가 변수 "abc"에 바인딩된다:

[0038]
```
<ComponentImplementation TypeId="SampleComponent">
    <SupportedPlatforms>
      <Platform>All</Platform>
    </SupportedPlatforms>
    <Composite>
      <Variables>
        <Variable Id="abc" Type="String" />
      </Variables>
      <Component TypeId="SampleComponent1">
        <Parameter Id="A">$Variable/abc$</Parameter>
      </Component>
      <Component TypeId="SampleComponent2">
        <Parameter Id="A">$Variable/abc$</Parameter>
      </Component>
    </Composite>
  </ComponentImplementation>
```

[0039] 이하는 Parameter 노드 내의 레퍼런스 프로토콜의 예시적인 목록이다:

[0040]
```
$Parameter/<propertyName>$ Parameter passed to component
$Variable/<propertyName>$  Variable declared in component
$Target/<propertyName>$    Property of a target instance passed to
component
$Target$                   Target instance
```

[0041] 글로벌 변수가 비관련 데이터 맥락에서 조직(베이스 컴포넌트 및/또는 컨테이너 컴포넌트) 사이의 데이터 교환을 가능하게 하기 위해 사용될 수 있다. 먼저, 글로벌 변수가 선언된다:

[0042]
```
<GlobalVariable ID="GlobalSelectedItem" Type="String"/>
```

[0043] 변수는 `$GlobalVariable/<variable name>$` 를 이용하여 베이스 컴포넌트 및 컨테이너 컴포넌트에서 레퍼런스될 수 있다.

[0044] 여하한 주어진 컴포넌트(예를 들어, 베이스, 컨테이너)는, 컴포넌트 칠드런이 변수의 프라이빗 사본을 보도록, 변수를 프라이빗 구현으로 오버라이드할 수 있는데, 다음 예시적 코드에서 보여진다:

[0045]
```
<Variables>
    <GlobalVariableOverride GlobalVariableId="GlobalSelectedItem" />
  </Variables>
```

[0046] 도 7은 조직 엔진에서의 변수의 사용을 나타내는 설명도(700)를 도시한다. 702에서, 글로벌 변수 "A"가 선언된다. 704에서, 파라미터 이름 "Blah"가 컴포넌트로 전달된다. 706에서, 글로벌 변수의 로컬 사본이 생성된다. 708 및 710에서, 글로벌 변수가 아니라 변수의 로컬 사본이 사용된다.

[0047] 개시된 아키텍처의 신규한 태양을 수행하기 위한 예시적인 방법론을 나타내는 흐름도의 세트가 여기에 포함된다. 설명의 단순화를 위해, 예를 들어 흐름도(flow chart 또는 flow diagram)의 형태로 여기에 도시된 하나 이상의 방법론이 일련의 동작으로 도시되고 설명되지만, 방법론에 따라서 일부 동작이 여기에 도시되고 설명된 것과 다른 순서로 및/또는 다른 동작과 동시에 일어날 수 있으므로, 방법론은 동작의 순서에 의해 제한되지 않음을 이해하고 인식할 것이다. 예를 들어, 당업자는, 다르게는 방법론이 상태도에서와 같이 일련의 서로 연관된 상태 또는 이벤트로서 나타내어질 수도 있음을 이해하고 인식할 것이다. 또한, 방법론에서 도시된 모든 동작이 신규한 구현을 위해 요구되는 것은 아니다.

[0048] 도 8은 개시된 아키텍처에 따른 시각화 방법을 도시한다. 800에서, 실행 환경에서 채용될 컴포넌트에 대한 요청이 수신된다. 802에서, 컴포넌트와 연관된 컴포넌트 정의가 검색된다. 804에서, 검색된 컴포넌트 정의에 대

해 하나 이상의 데이터 정의가 선택된다. 806에서, 하나 이상의 데이터 정의가 컴포넌트 정의를 이용하여 자동으로 조직되어 환경 런타임에 실행 환경에서 컴포넌트를 출력한다.

[0049] 도 9는 도 8의 방법의 또 다른 태양을 도시한다. 900에서, 컴포넌트 정의가 검색되지 않은 때에, 요청된 컴포넌트의 데이터 유형에 기초하여 컴포넌트 정의가 검색된다. 902에서, 커스텀 컴포넌트가 요청된 컴포넌트의 부재에 기초하여 생성된다. 904에서, 비관련 데이터 맥락 사이에서 데이터 교환을 가능하게 하기 위해 글로벌 변수가 컴포넌트에 적용된다. 906에서, 컴포넌트의 차일드 컴포넌트에 프라이빗 변수를 부과하기 위해 글로별 변수가 프라이빗 변수로 오버라이드된다. 908에서, 요청된 컴포넌트가 검색되지 않는 때에 컨테이너 컴포넌트가 생성된다. 910에서, 컨테이너 컴포넌트가 베이스 컴포넌트 연관 데이터 유형 속성을 이용하여 로딩된다. 912에서, 컨테이너 컴포넌트가 컴포넌트로서 출력된다.

[0050] 도 10은 다른 시각화 방법을 도시한다. 1000에서, 컴포넌트 실행 환경에서 기초 컴포넌트에 대한 요청이 수신된다. 1002에서, 컴포넌트와 연관된 컴포넌트 정의가 검색된다. 1004에서, 컴포넌트 정의가 검색되면, 컴포넌트 정의에 대한 하나 이상의 데이터 정의가 선택된다. 1006에서, 컴포넌트 정의가 검색되지 않으면 요청된 컴포넌트와 연관된 데이터 유형에 기초하여 커스텀 컴포넌트가 생성된다. 1008에서, 글로벌 변수가 비관련 데이터 맥락 사이의 데이터 교환을 가능하게 하기 위해 컴포넌트 또는 커스텀에 적용된다. 1010에서, 하나 이상의 데이터 정의가 자동으로 컴포넌트 정의를 이용하여 자동으로 조직되어 환경 런타임에서 실행 환경에서 컴포넌트를 출력한다.

[0051] 도 11에서, 도 10의 방법의 또 다른 태양이 도시된다. 1100에서, 컴포넌트의 차일드 컴포넌트에 프라이빗 변수를 부과하기 위해 글로벌 변수가 프라이빗 변수로 오버라이드된다. 1102에서, 요청된 컴포넌트가 검색되지 않는 때에 컨테이너 컴포넌트가 생성된다. 1104에서, 컨테이너 컴포넌트가 베이스 컴포넌트 연관 데이터 유형 속성으로 로딩된다. 1106에서, 컨테이너 컴포넌트가 컴포넌트로서 출력된다. 1108에서, 컴포넌트로 전달될 데이터가 파라미터 노드를 통해 정의된다. 1110에서, 차일드 컴포넌트로의 속성의 바인딩과 데이터 컴포넌트로의 속성의 바인딩을 포함하는 부모 컴포넌트가 조직된다.

[0052] 도 12는 조직 엔진에서 컴포넌트를 획득하는 방법을 도시한다. 1200에서, 컴포넌트가 획득된다(예를 들어, "get component" 호출을 통해). 1202에서, 타겟 데이터 유형이 획득된다. 1204에서, 타겟 데이터 유형에 대한 컴포넌트가 존재하는지 여부에 대한 확인이 이루어진다. 존재하지 않으면, 흐름은 1206으로 가서 컴포넌트 컨테이너를 생성한다. 컴포넌트는 하나 이상의 연관된 데이터 속성을 가질 수 있다. 1208에서, 속성 유형이 컴포넌트에 대해 획득된다. 1210에서, 속성 유형에 대해 "get component" 호출이 이루어진다. 1212에서, 컴포넌트가 컨테이너에 추가된다. 흐름은 1208로 돌아가서 완료될 때까지 계속된다. 모든 데이터 속성 및 유형이 컨테이너에 적용된 후에, 흐름이 1214로 가서 결과를 반환한다. 1204에서, 확인이 타겟 데이터 유형에 대해 컴포넌트가 존재하는 것으로 결정하면, 흐름은 1216으로 가서 컴포넌트를 선택하고, 그 후 1216에서 결과를 반환한다.

[0053] 도 13은 조직 엔진에서 컴포넌트를 획득하는 더 상세한 방법을 도시한다. 1300에서, 타겟 유형 및 데이터 유형과 같은 함수 파라미터가 수신된다. 1302에서, 타겟 유형 및 이름에 대해 정의된 컴포넌트에 대한 확인이 이루어진다. 확인이 되면, 흐름은 1304로 가서 인터페이스를 검증한다. 다르게는, 타겟 유형 및 이름에 대해 컴포넌트가 존재하지 않으면, 흐름은 1306으로 가서 유형 시스템 룩업(lookup)을 이용하여, 모든 속성에 대해 컴포넌트가 정의되는지 여부에 대한 결정이 이루어진다. 만약 그렇다면, 흐름은 1304로 가서 인터페이스를 검증한다. 유형 시스템(1308)으로의 액세스가 제공되어 체크를 하고 인터페이스를 검증한다. 인터페이스가 검증되면, 흐름은 1310으로 가서 컴포넌트 유형을 확인한다. 단위 컴포넌트이면, 흐름은 1312로 가서 정확한 플랫폼에 대하여 단위 컴포넌트의 인스턴스를 생성하고 단위 컴포넌트로 모든 선언된 파라미터를 전달한다. 1314에서, 로더는 UX 조직 시스템(예를 들어, XAML, MEF 등)에 대한 어셈블리를 로딩할 수 있다.

[0054] 1310에서 컴포넌트 유형이 혼합 컴포넌트이면, 흐름은 1316으로 가서 구성(예를 들어, XML에서 기록됨)에서 차일드 노드를 보행(walk)하여 조직의 컴포넌트 상에서 값을 설정한다. 이는, 1322에서 파라미터로부터의 빌드 데이터로부터 제공되는 바와 같이, 파라미터 노드(들)(1318)로부터의 파라미터 값, 컴포넌트 노드(들)(1320)로부터의 파라미터, 그리고 차일드 노드들로부터의 파라미터를 수신하는 것을 포함한다. TypeId는 컴포넌트 노드(1320)로부터 빌드 데이터(1322)에게 보내진다. 1324에서, 컴포넌트 노드(1320)로부터 수신된 이름 및 타겟 정보에 기초하여, 타겟에 의해 데이터와 이름으로서 레퍼런스된 컴포넌트는 룩업을 위한 이름으로 사용된다.

[0055] 하나 이상의 컴포넌트가 실행 프로세스 및/또는 스레드 내에 거주할 수 있고, 컴포넌트는 하나의 컴퓨터 상에 국지화되거나 및/또는 2 이상의 컴퓨터 사이에 분산될 수 있다. "예시적인"이라는 단어는 여기서 예(example),

예시(instance) 또는 예증(illustration)으로 기능함을 의미하는 것으로 사용될 수 있다. 여기에 "예시적인" 것으로 설명된 태양 또는 설계는 반드시 다른 태양 또는 설계에 비해 바람직하거나 유리한 것으로 해석되어야 하는 것은 아니다.

[0056] 이제 도 14를 참조하면, 개시된 아키텍처에 따라 조직을 실행하는 연산 시스템(1400)의 블록도가 도시되어 있다. 다양한 태양에 대한 추가적인 맥락을 제공하기 위하여, 도 14 및 다음의 설명은 다양한 태양이 구현될 수 있는 적당한 연산 시스템(1400)의 간략하고 일반적인 설명을 제공하려는 것이다. 위의 설명이 하나 이상의 컴퓨터 상에서 실행될 수 있는 컴퓨터-실행가능 명령의 일반적인 맥락에서 이루어진 것이지만, 당업자는 신규한 실시형태가 또한 다른 프로그램 모듈과 결합하여 및/또는 하드웨어 및 소프트웨어의 결합으로서 구현될 수 있음을 인식할 것이다.

[0057] 다양한 태양을 구현하기 위한 연산 시스템(1400)은 처리 유닛(들)(1404), 시스템 메모리(1406)와 같은 컴퓨터 판독가능 저장소, 및 시스템 버스(1408)를 갖는 컴퓨터(1402)를 포함한다. 처리 유닛(들)(1404)은, 단일 프로세서, 멀티 프로세서, 싱글 코어 유닛 및 멀티 코어 유닛과 같은 다양한 상용 프로세서 중 여하한 것일 수 있다. 또한, 당업자는, 미니컴퓨터, 메인프레임 컴퓨터 뿐만 아니라 퍼스널 컴퓨터(예를 들어, 데스크탑, 랩탑 등), 핸드-헬드 연산 장치, 마이크로프로세서 기반 또는 프로그램 가능 소비자 가전 등을 포함하는 다른 컴퓨터 시스템 구성으로 신규한 방법이 실시될 수 있음을 인식할 것인데, 이들 각각은 하나 이상의 연관 장치에 동작상 결합될 수 있다.

[0058] 시스템 메모리(1406)는 휘발성(VOL) 메모리(1410)(예를 들어, RAM(random access memory)) 및 비휘발성 메모리(NON-VOL)(1412)(예를 들어, ROM, EPROM, EEPROM 등)과 같은 컴퓨터 판독가능 저장소(물리적 저장 매체)를 포함할 수 있다. 기본 입력/출력 시스템(BIOS)은 비휘발성 메모리(1412)에 저장될 수 있고, 시동 중 등에, 컴퓨터(1402) 내 컴포넌트 사이의 데이터 및 신호의 통신을 용이하게 하는 기본 루틴을 포함한다. 휘발성 메모리(1410)는 데이터 캐싱을 위한 정적 RAM과 같은 고속 RAM을 포함할 수도 있다.

[0059] 시스템 버스(1408)는, 시스템 메모리(1406)를 포함하지만 이에 제한되지 않는 시스템 컴포넌트에 대한 처리 유닛(들)(1404)로의 인터페이스를 제공한다. 시스템 버스(1408)는 다양한 상용 버스 아키텍처 중 여하한 것을 이용하여 메모리 버스(메모리 컨트롤러가 있거나 없음) 및 주변 버스(예를 들어, PCI, PCIe, AGP, LPC 등)을 추가로 상호접속할 수 있는 버스 구조의 몇몇 유형 중 여하한 것일 수 있다.

[0060] 컴퓨터(1402)는 기계 판독가능 저장 서브시스템(들)(1414) 및 저장 서브시스템(들)(1414)을 시스템 버스(1408) 및 기타 희망 컴퓨터 컴포넌트로 인터페이스하기 위한 저장 인터페이스(들)(1416)을 더 포함한다. 저장 서브시스템(들)(1414)(물리적 저장 매체)은 예를 들어 하드 디스크 드라이브(HDD), 자기 플로피 디스크 드라이브(FDD), 및/또는 광 디스크 저장 드라이브(예를 들어, CD-ROM 드라이브, DVD 드라이브) 중 하나 이상을 포함할 수 있다. 저장 인터페이스(들)(1416)은 예를 들어 EIDE, ATA, SATA, 및 IEEE 1394와 같은 인터페이스 기술을 포함할 수 있다.

[0061] 운영 체제(1420), 하나 이상의 애플리케이션 프로그램(1422), 다른 프로그램 모듈(1424) 및 프로그램 데이터(1426)를 포함하는 하나 이상의 프로그램 및 데이터가 메모리 서브시스템(1406), 기계 판독가능 및 제거가능 메모리 서브시스템(1418)(예를 들어, 플래시 드라이브 폼 팩터 기술), 및/또는 저장 서브시스템(들)(1414)(예를 들어, 광, 자기, 솔리드 스테이트)에 저장될 수 있다.

[0062] 하나 이상의 애플리케이션 프로그램(1422), 기타 프로그램 모듈(1424) 및 프로그램 데이터(1426)가, 예를 들어, 도 1의 시스템(100)의 엔티티 및 컴포넌트, 도 2의 시스템(200)의 엔티티 및 컴포넌트, 도 3의 조직, 도 4의 부모 컴포넌트(400), 도 5의 컴포넌트 정의(500), 도 6의 레지스트리(600), 도 7의 도면(700), 그리고 도 8-13의 흐름도에 의해 표현되는 방법을 포함할 수 있다.

[0063] 일반적으로, 프로그램은 특정 태스크를 수행하거나 특정 추상 데이터 유형을 구현하는 루틴, 방법, 데이터 구조, 기타 소프트웨어 컴포넌트 등을 포함한다. 운영 체제(1420), 애플리케이션(1422), 모듈(1424) 및/또는 데이터(1426)의 전부 또는 일부는 예를 들어 휘발성 메모리(1410)와 같은 메모리에 캐싱될 수 있다. 개시된 아키텍처는 다양한 상용 운영 체제 또는 운영 체제의 조합(예를 들어, 가상 머신으로서)을 이용하여 구현될 수 있음을 이해할 것이다.

[0064] 저장 서브시스템(들)(1414) 및 메모리 서브시스템(1406 및 1418)은 데이터, 데이터 구조, 컴퓨터 실행가능 명령 등의 휘발성 및 비휘발성 저장을 위한 컴퓨터 판독가능 매체로서 기능한다. 이러한 명령은, 컴퓨터 또는 기타 기계에 의해 실행되는 때에, 컴퓨터 또는 기타 기계가 방법의 하나 이상의 동작을 수행하도록 할 수 있다. 동

작을 수행하라는 명령은 하나의 매체에 저장될 수 있고, 또는 명령의 전부가 동일한 매체에 있는지와 무관하게 하나 이상의 컴퓨터 판독가능 저장 매체 상에 명령이 집합적으로 나타나도록 다수의 매체에 걸쳐 저장될 수 있다.

[0065] 컴퓨터 판독가능 매체는 컴퓨터(1402)에 의해 액세스될 수 있는 여하한 가용 매체일 수 있고, 제거가능 또는 비제거가능의 휘발성 및 비휘발성 내부 및/또는 외부 매체를 포함한다. 컴퓨터(1402)에 대해, 매체는 여하한 적당한 디지털 형식으로 데이터의 저장을 수용한다. 당업자는, 집(zip) 드라이브, 자기 테이프, 플래시 메모리 카드, 플래시 드라이브, 카트리지 등과 같은 다른 유형의 컴퓨터 판독가능 매체가 개시된 아키텍처의 신규한 방법을 수행하기 위한 컴퓨터 실행가능 명령을 저장하기 위해 채용될 수 있음을 인식하여야 한다.

[0066] 사용자는 키보드 및 마우스와 같은 외부 사용자 입력 장치(1428)를 이용하여 컴퓨터(1402), 프로그램 및 데이터와 상호작용할 수 있다. 기타 외부 사용자 입력 장치(1428)는 마이크로폰, IR(적외선) 리모트 컨트롤, 조이스틱, 게임 패드, 카메라 인식 시스템, 스타일러스 펜, 터치 스크린, 제스처 시스템(예를 들어, 안구 운동, 두부 운동 등) 및/또는 기타를 포함할 수 있다. 사용자는 컴퓨터(1402), 프로그램 및 데이터와 터치패드, 마이크로폰, 키보드 등과 같은 온보드 사용자 입력 장치(1430)를 이용하여 상호작용할 수 있는데, 여기서 컴퓨터(1402)는 예를 들어 휴대용 컴퓨터이다. 이들 및 기타 입력 장치가 입력/출력(I/O) 장치 인터페이스(들)(1432)을 통해 시스템 버스(1408)에 의해 처리 유닛(들)(1404)에 접속되지만, 병렬 포트, IEEE 1394 직렬 포트, 게임 포트, USB 포트, IR 인터페이스 등과 같은 다른 인터페이스에 의해 접속될 수 있다. 사운드 카드 및/또는 온보드 오디오 처리 기능과 같은 I/O 장치 인터페이스(들)(1432)은 프린터, 오디오 장치, 카메라 장치 등과 같은 출력 주변장치(1434)의 사용을 용이하게 할 수도 있다.

[0067] 하나 이상의 그래픽 인터페이스(들)(1436)(보통 GPU(graphical processing unit)라고도 함)은 컴퓨터(1402)와 외부 디스플레이(들)(1438)(예를 들어, LCD, 플라즈마) 및/또는 온보드 디스플레이(1440)(예를 들어, 휴대용 컴퓨터에 대해) 사이의 그래픽 및 비디오 신호를 제공한다. 그래픽 인터페이스(들)(1436)은 컴퓨터 시스템 보드의 일부로서 제조될 수도 있다.

[0068] 컴퓨터(1402)는 유선/무선 통신 서브시스템(1442)을 통한 하나 이상의 네트워크 및/또는 기타 컴퓨터로의 논리적 접속을 이용하여 네트워크된 환경(예를 들어, IP-기반)에서 동작할 수 있다. 기타 컴퓨터는 워크스테이션, 서버, 라우터, 개인용 컴퓨터, 마이크로프로세서 기반 엔터테인먼트 기기, 피어(peer) 장치 또는 기타 공통 네트워크 노드를 포함할 수 있고, 보통 컴퓨터(1402)에 대해 설명된 요소 전부 또는 많은 부분을 포함한다. 논리적 접속은 LAN(local area network), WAN(wide area network), 핫스팟 등으로의 유선/무선 접속을 포함할 수 있다. LAN과 WAN 네트워킹 환경은 사무실과 회사에 흔하며, 인트라넷과 같은 기업 범위 컴퓨터 네트워크를 용이하게 하는데, 이들 모두는 인터넷과 같은 글로벌 통신 네트워크에 접속될 수 있다.

[0069] 네트워킹 환경에서 사용되는 때에, 컴퓨터(1402)는 유선/무선 통신 서브시스템(1442)(예를 들어, 네트워크 인터페이스 어댑터, 온보드 트랜시버 서브시스템 등)을 통해 네트워크로 접속되어 유선무선 네트워크, 유선/무선 프린터, 유선/무선 입력 장치(1444) 등과 통신한다. 컴퓨터(1402)는 네트워크를 통해 통신을 수립하기 위한 모뎀 또는 기타 수단을 포함할 수 있다. 네트워크된 환경에서, 컴퓨터(1402)에 대한 프로그램 및 데이터는, 분산 시스템과 연관되는 것처럼, 원격 메모리/저장 장치에 저장될 수 있다. 도시된 네트워크 접속은 예시적인 것이고 컴퓨터 사이에 통신 링크를 수립하는 여하한 다른 수단이 사용될 수 있음을 인식할 것이다.

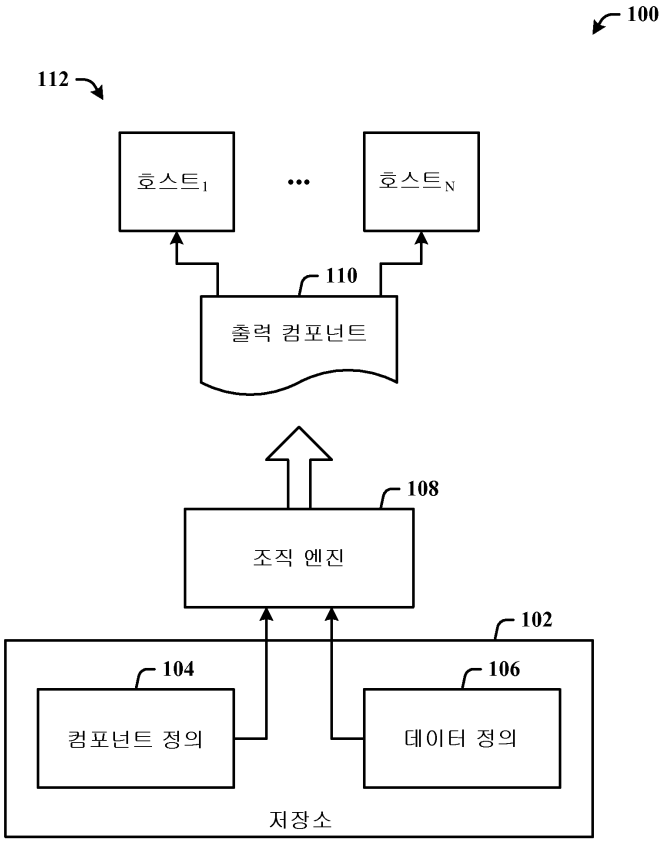
[0070] 컴퓨터(1402)는, IEEE 802.xx 표준 패밀리와 같은 무선 기술을 이용하여, 예를 들어, 프린터, 스캐너, 데스크탑 및/또는 휴대용 컴퓨터, PDA(personal digital assistance), 통신 위성, 무선으로 검출가능한 태그와 연관된 여하한 장치 또는 위치(예를 들어, 키오스크, 신문 가판대, 화장실) 및 전화기와 무선으로 통신(예를 들어, IEEE 802.11 오버-디-에어 변조 기술)하도록 동작상 배치된 무선 장치와 같은 무선 장치 유선/무선 장치 또는 엔티티와 통신하도록 동작가능하다. 이는 적어도 핫스팟을 위한 Wi-Fi(또는 와이얼리스 피델리티(Wireless Fidelity)), WiMax, 블루투스(Bluetooth)™ 무선 기술을 포함한다. 그러므로, 통신은 종래의 네트워크에서와 같이 사전 정의된 구조일 수 있거나 단순히 적어도 2개의 장치 사이의 애드 혹(ad hoc) 통신일 수 있다. Wi-Fi 네트워크는 IEEE 802.1x (a, b, g 등)이라고 불리는 무선 기술을 이용하여 보안되고 신뢰성있으며 빠른 무선 접속을 제공한다. Wi-Fi 네트워크는 컴퓨터를 서로, 인터넷으로, 그리고 네트워크(IEEE 802.3 관련 매체 및 기능을 이용함)으로 접속하는데 사용될 수 있다.

[0071] 도시되고 설명된 태양은 특정 태스크가 통신 네트워크를 통해 링크된 원격 처리 장치에 의해 수행되는 분산 컴퓨팅 환경에서 실시될 수 있다. 분산 컴퓨팅 환경에서, 프로그램 모듈은 로컬 및/또는 원격 저장 및/또는 메모리 시스템에 배치될 수 있다.

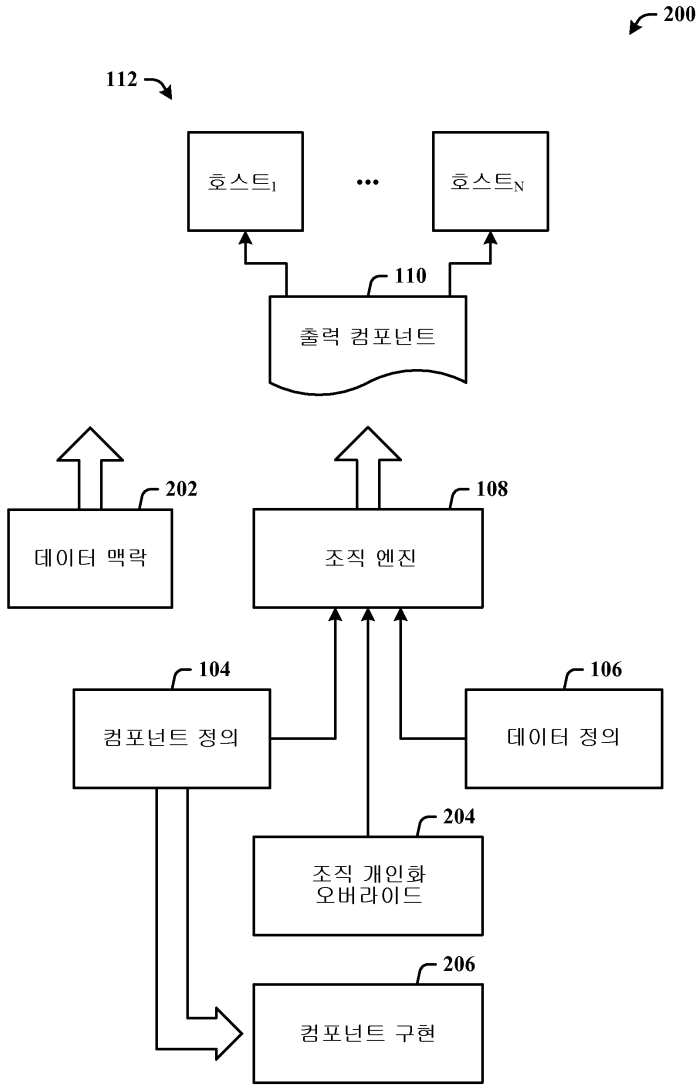
[0072] 상술된 사항은 개시된 아키텍터의 예를 포함한다. 물론, 컴포넌트 및/또는 방법론의 생각가능한 모든 조합을 설명하는 것은 가능하지 않지만, 당업자는 추가의 결합 및 조합(permutation)이 가능함을 인식할 수 있다. 따라서, 신규한 아키텍처는 첨부된 청구항의 사상 및 범위 내에 있는 모든 이러한 변경, 변형 및 변화를 포함하려는 것이다. 또한, 상세한 설명이나 청구범위에서 "포함한다(include)"라는 용어가 사용되는 한, 이 용어는 "포함한다(comprising)"가 청구항에서 접속어(transitional word)로 사용되는 때에 해석되는 바와 같이 "포함한다(comprising)"와 유사한 방식으로 포함적인(inclusive) 것으로 의도된 것이다.

도면

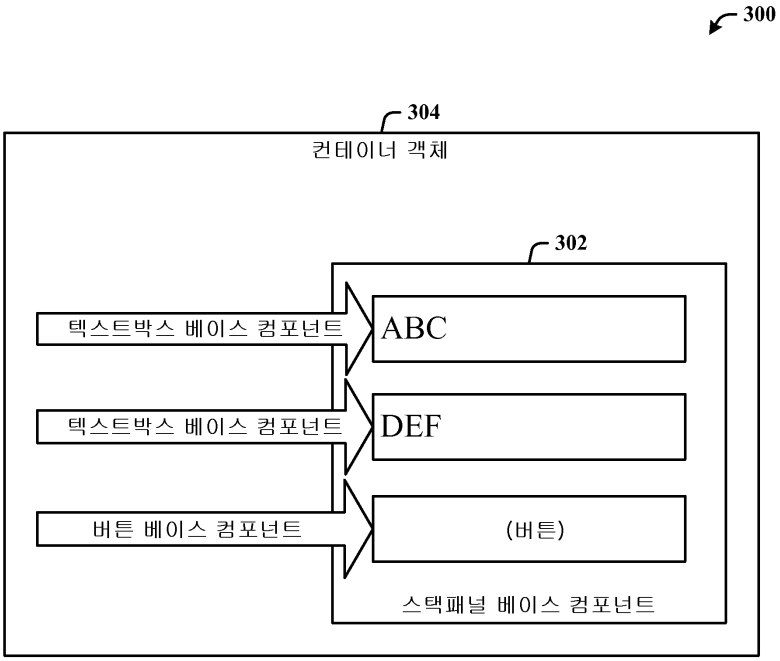
도면1



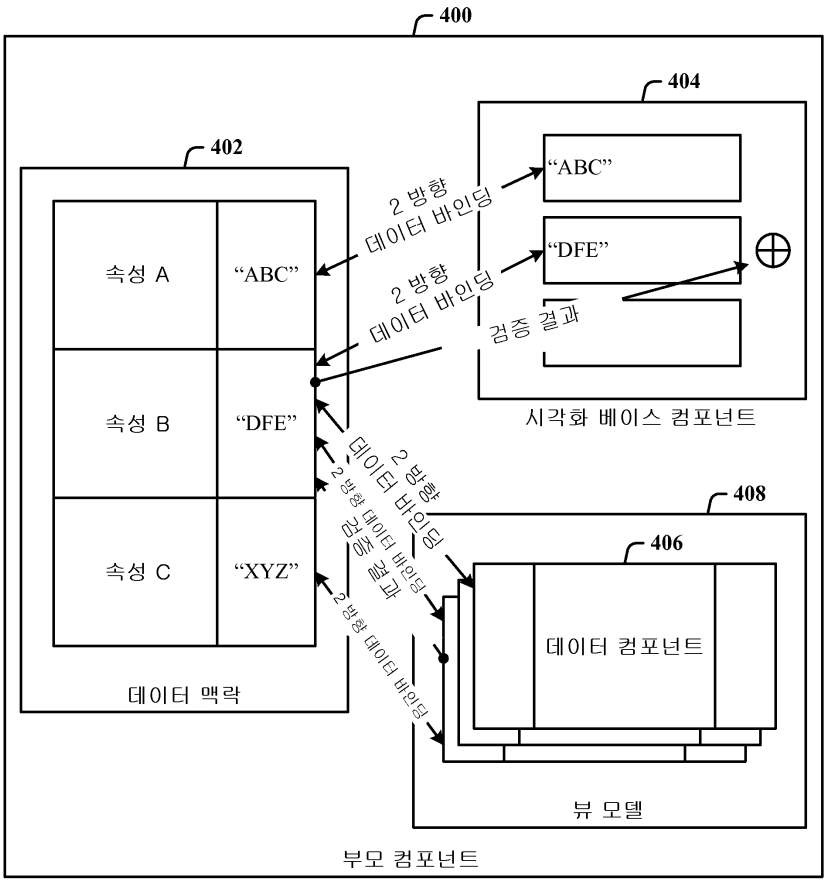
도면2


200
112
호스트1
...
호스트N
110
출력 컴포넌트
202
데이터 맥락
108
조직 엔진
104
컴포넌트 정의
106
데이터 정의
204
조직 개인화
오버라이드
206
컴포넌트 구현

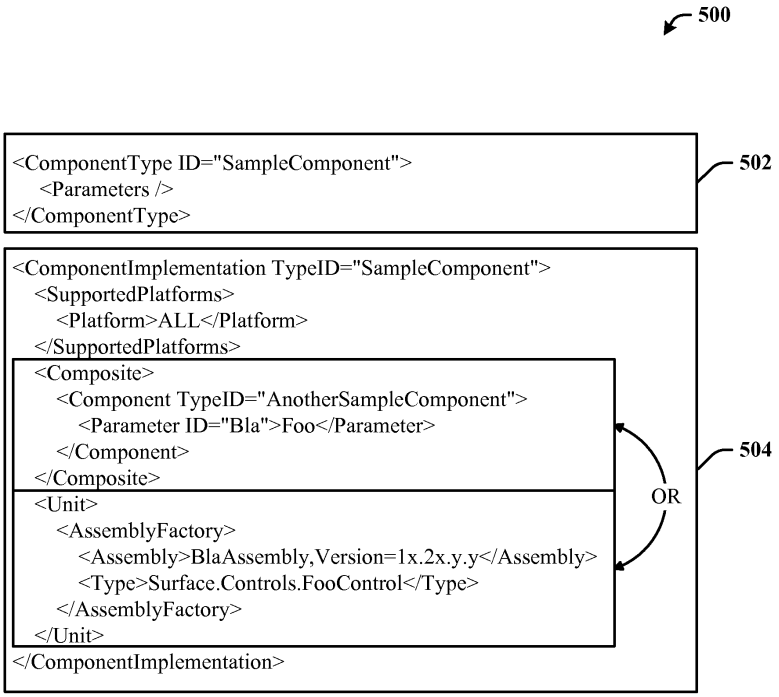
도면3


300
304
컨테이너 객체
302
텍스트박스 베이스 컴포넌트
ABC
텍스트박스 베이스 컴포넌트
DEF
버튼 베이스 컴포넌트
(버튼)
스택패널 베이스 컴포넌트

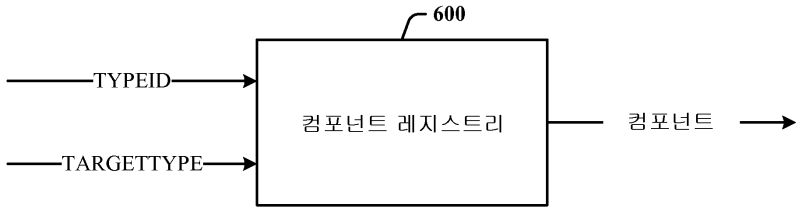
도면4


400
404
402
"ABC"
2 방향 데이터 바인딩
속성 A
"ABC"
"DFE"
2 방향 데이터 바인딩
검증 결과
속성 B
"DFE"
시각화 베이스 컴포넌트
데이터 바인딩
2 방향
408
406
속성 C
"XYZ"
데이터 컴포넌트
데이터 맥락
뷰 모델
부모 컴포넌트

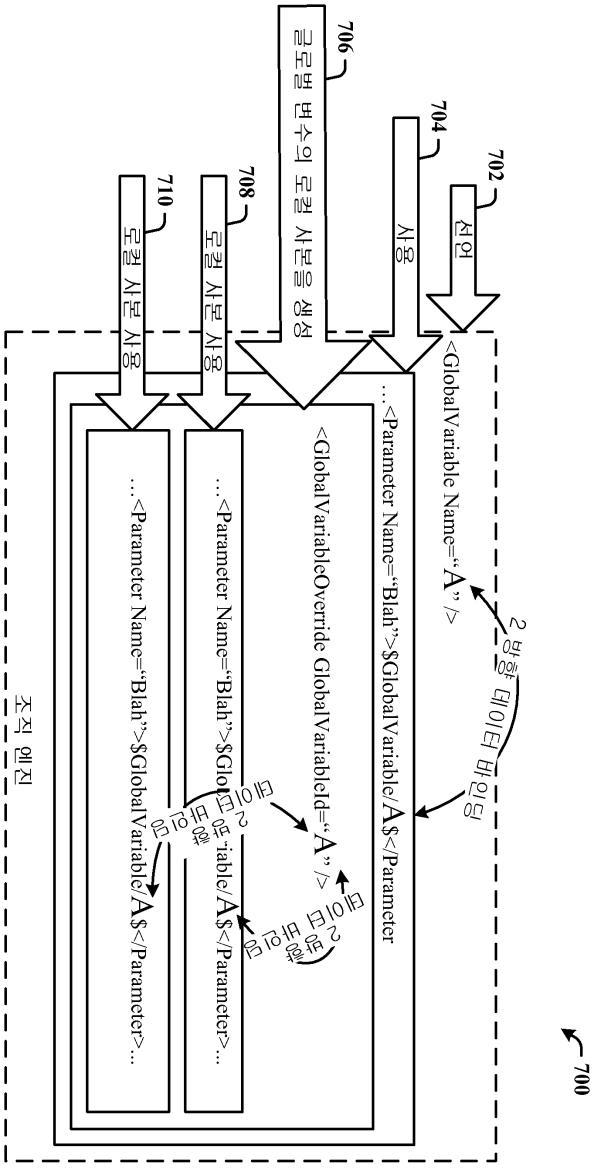
도면5


500
<ComponentType ID="SampleComponent">
<Parameters />
</ComponentType>
502
<ComponentImplementation TypeID="SampleComponent">
<SupportedPlatforms>
<Platform>ALL</Platform>
</SupportedPlatforms>
<Composite>
<Component TypeID="AnotherSampleComponent">
<Parameter ID="Bla">Foo</Parameter>
</Component>
</Composite>
OR
<Unit>
<AssemblyFactory>
<Assembly>BlaAssembly,Version=1x.2x.y.y</Assembly>
<Type>Surface.Controls.FooControl</Type>
</AssemblyFactory>
</Unit>
</ComponentImplementation>
504

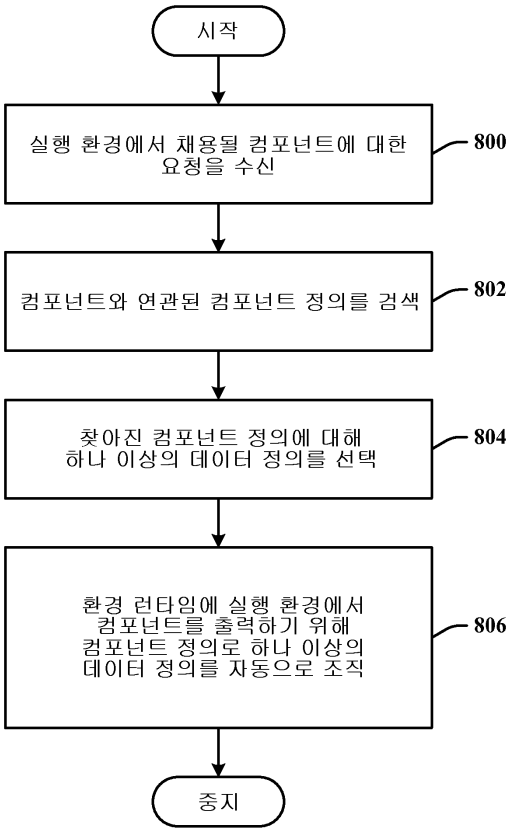
도면6


600
TYPEID
TARGETTYPE
컴포넌트 레지스트리
컴포넌트

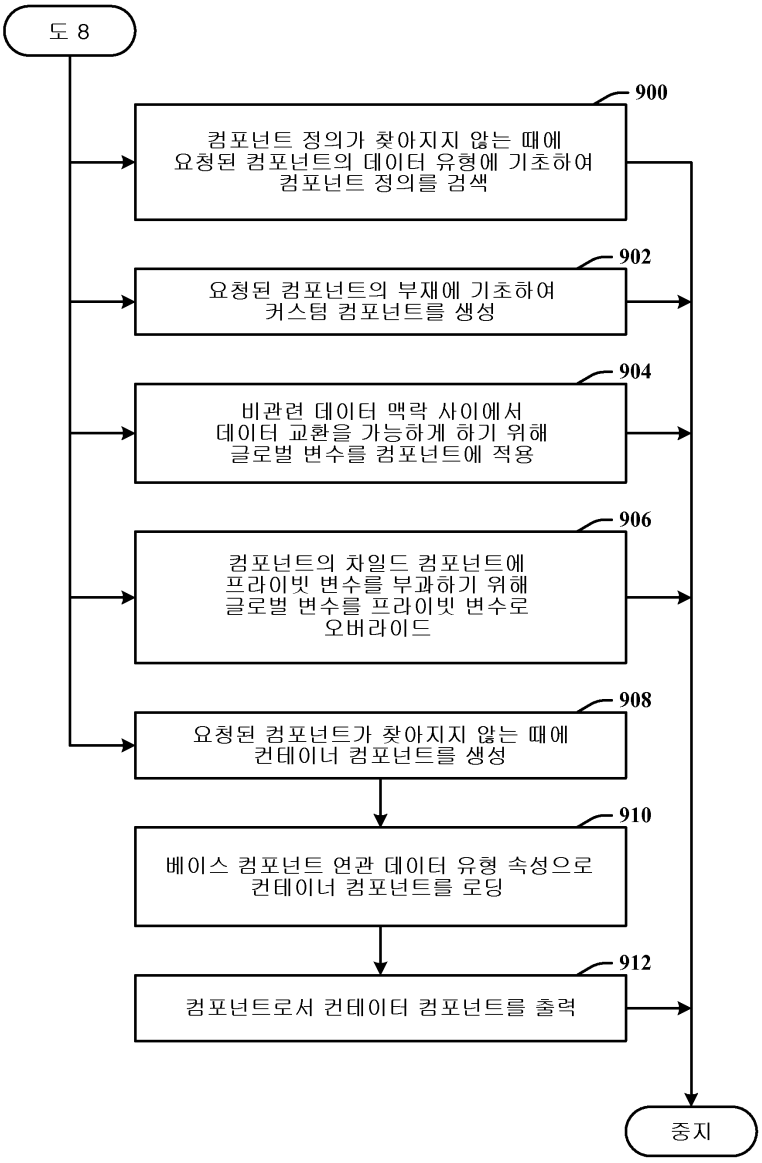
도면7


조직 엔진
<GlobalVariable Name="A" />
선언
사용
글로벌 변수의 로컬 사본을 생성
로컬 사본 사용
로컬 사본 사용
...<Parameter Name="Blah">$GlobalVariable/A$</Parameter
<GlobalVariableOverride GlobalVariableId="A" />
...<Parameter Name="Blah">$GlobalVariable/A$</Parameter>...
...<Parameter Name="Blah">$GlobalVariable/A$</Parameter>...
2방향 데이터 바인딩
데이터 바인딩
데이터 바인딩
700

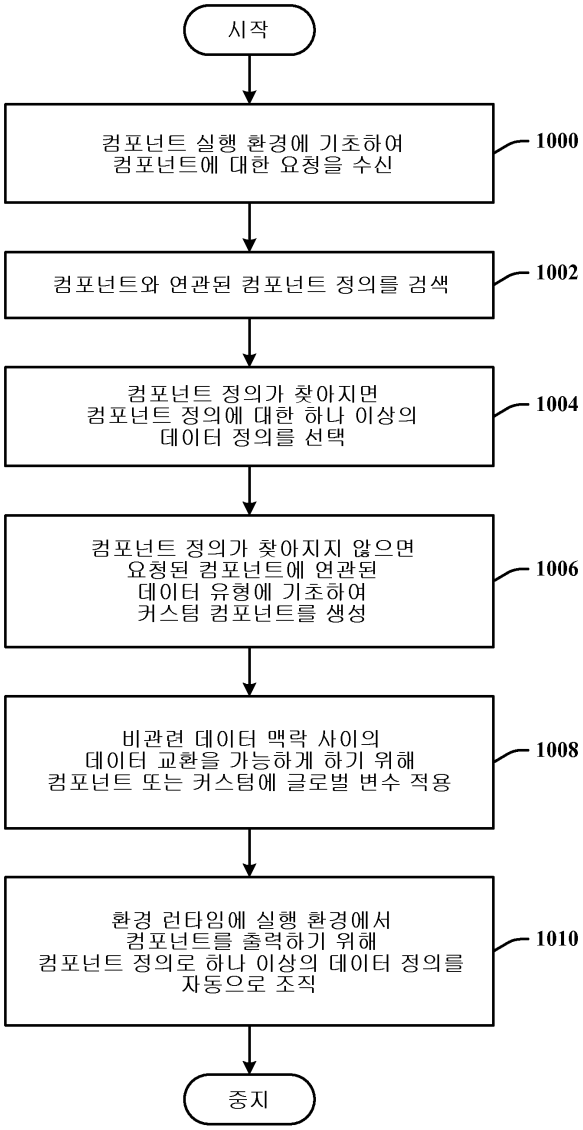
도면8


시작
실행 환경에서 채용될 컴포넌트에 대한 요청을 수신
800
컴포넌트와 연관된 컴포넌트 정의를 검색
802
찾아진 컴포넌트 정의에 대해 하나 이상의 데이터 정의를 선택
804
환경 런타임에 실행 환경에서 컴포넌트를 출력하기 위해 컴포넌트 정의로 하나 이상의 데이터 정의를 자동으로 조직
806
중지

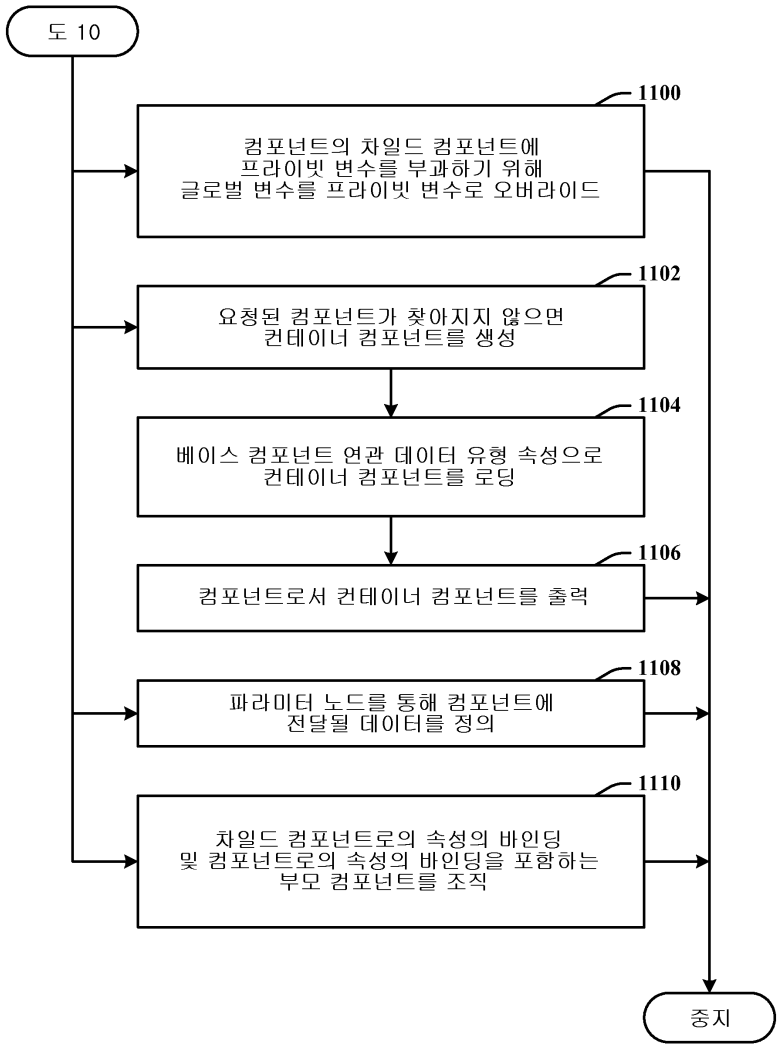
도면9


도 8
900
컴포넌트 정의가 찾아지지 않는 때에
요청된 컴포넌트의 데이터 유형에 기초하여
컴포넌트 정의를 검색
902
요청된 컴포넌트의 부재에 기초하여
커스텀 컴포넌트를 생성
904
비관련 데이터 맥락 사이에서
데이터 교환을 가능하게 하기 위해
글로벌 변수를 컴포넌트에 적용
906
컴포넌트의 차일드 컴포넌트에
프라이빗 변수를 부과하기 위해
글로벌 변수를 프라이빗 변수로
오버라이드
908
요청된 컴포넌트가 찾아지지 않는 때에
컨테이너 컴포넌트를 생성
910
베이스 컴포넌트 연관 데이터 유형 속성으로
컨테이너 컴포넌트를 로딩
912
컴포넌트로서 컨테이터 컴포넌트를 출력
중지

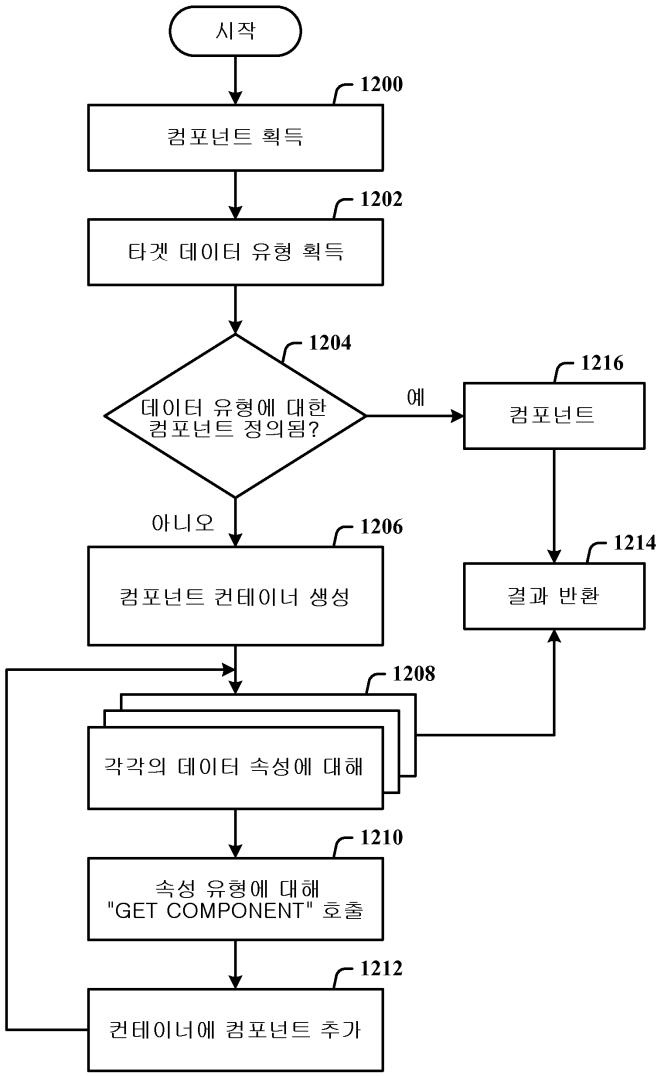
도면10


시작
컴포넌트 실행 환경에 기초하여
컴포넌트에 대한 요청을 수신
1000
컴포넌트와 연관된 컴포넌트 정의를 검색
1002
컴포넌트 정의가 찾아지면
컴포넌트 정의에 대한 하나 이상의
데이터 정의를 선택
1004
컴포넌트 정의가 찾아지지 않으면
요청된 컴포넌트에 연관된
데이터 유형에 기초하여
커스텀 컴포넌트를 생성
1006
비관련 데이터 맥락 사이의
데이터 교환을 가능하게 하기 위해
컴포넌트 또는 커스텀에 글로벌 변수 적용
1008
환경 런타임에 실행 환경에서
컴포넌트를 출력하기 위해
컴포넌트 정의로 하나 이상의 데이터 정의를
자동으로 조직
1010
중지

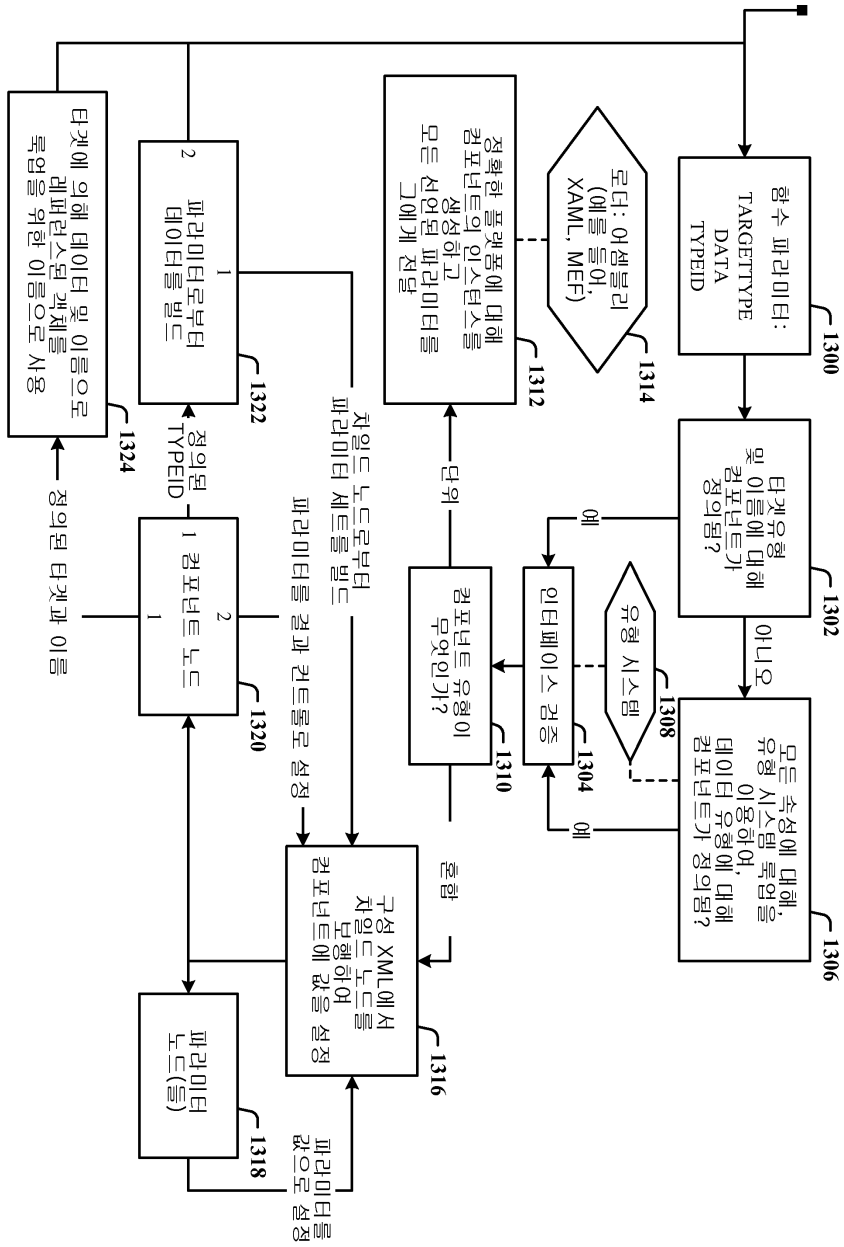
도면11


도 10
1100
컴포넌트의 차일드 컴포넌트에
프라이빗 변수를 부과하기 위해
글로벌 변수를 프라이빗 변수로 오버라이드
1102
요청된 컴포넌트가 찾아지지 않으면
컨테이너 컴포넌트를 생성
1104
베이스 컴포넌트 연관 데이터 유형 속성으로
컨테이너 컴포넌트를 로딩
1106
컴포넌트로서 컨테이너 컴포넌트를 출력
1108
파라미터 노드를 통해 컴포넌트에
전달될 데이터를 정의
1110
차일드 컴포넌트로의 속성의 바인딩
및 컴포넌트로의 속성의 바인딩을 포함하는
부모 컴포넌트를 조직
중지

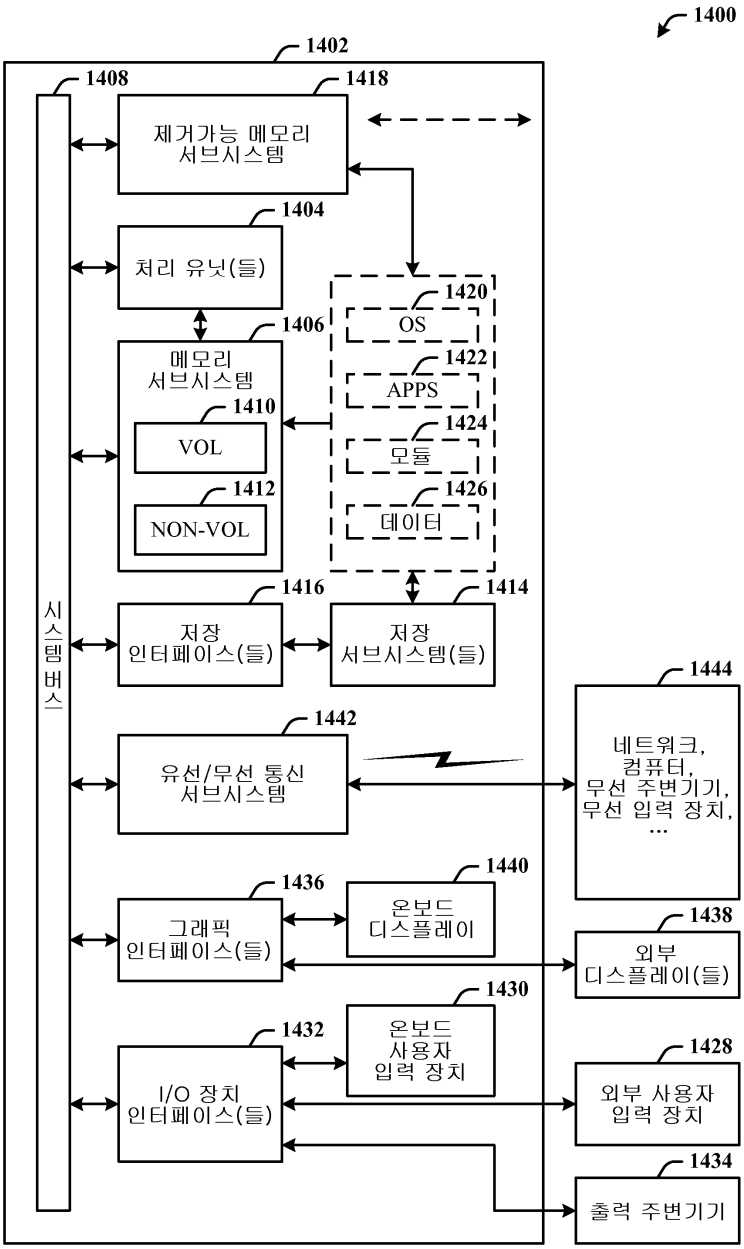
도면12


시작
1200
컴포넌트 획득
1202
타겟 데이터 유형 획득
1204
데이터 유형에 대한 컴포넌트 정의됨?
예
1216
컴포넌트
아니오
1206
컴포넌트 컨테이너 생성
1214
결과 반환
1208
각각의 데이터 속성에 대해
1210
속성 유형에 대해 "GET COMPONENT" 호출
1212
컨테이너에 컴포넌트 추가

도면13


함수 파라미터:
TARGETTYPE
DATA
TYPEID
1300
타겟 유형
및 이름에 대해
컴포넌트가
정의됨?
1302
아니오
예
모든 속성에 대해,
유형 시스템
이용하여,
데이터 유형에 대해
컴포넌트가 정의됨?
1306
유형 시스템
1308
인터페이스 검증
1304
예
컴포넌트 유형이
무엇인가?
1310
혼합
단위
로더: 어셈블리
(예를 들어,
XAML, MEF)
1314
정확한 플랫폼에 대해
컴포넌트의 인스턴스를
생성하고
모든 선언된 파라미터를
그에게 전달
1312
차일드 노드로부터
파라미터 세트를 빌드
파라미터를 결과 컨트롤로 설정
구성 XML에서
차일드 노드를
반복하여
컴포넌트에 값을 설정
1316
파라미터
노드(들)
1318
파라미터로부터
데이터를 빌드
1322
정의된
TYPEID
컴포넌트 노드
1320
정의된 타겟과 이름
타겟에 의해 데이터 및 이름으로
레퍼럴스될 객체를
룩업을 위한 이름으로 사용
1324
파라미터
값으로 설정

도면14


1400
1402
1408
1418
제거가능 메모리
서브시스템
1404
처리 유닛(들)
1420
OS
1406
메모리
서브시스템
1422
APPS
1410
VOL
1424
모듈
1412
NON-VOL
1426
데이터
1416
1414
저장
인터페이스(들)
저장
서브시스템(들)
시스템 버스
1444
네트워크,
컴퓨터,
무선 주변기기,
무선 입력 장치,
...
1442
유선/무선 통신
서브시스템
1436
1440
그래픽
인터페이스(들)
온보드
디스플레이
1438
외부
디스플레이(들)
1430
1432
온보드
사용자
입력 장치
1428
I/O 장치
인터페이스(들)
외부 사용자
입력 장치
1434
출력 주변기기