(19) 대한민국특허청(KR)
(12) 공개특허공보(A)

(11) 공개번호 10-2015-0143658
(43) 공개일자 2015년12월23일

(51) 국제특허분류(Int. Cl.)
G06F 9/44 (2006.01) *G06F 17/24* (2006.01)
G06F 9/45 (2006.01)
(52) CPC특허분류
G06F 9/4443 (2013.01)
G06F 17/246 (2013.01)
(21) 출원번호 10-2015-7032307
(22) 출원일자(국제) 2014년04월11일
심사청구일자 없음
(85) 번역문제출일자 2015년11월11일
(86) 국제출원번호 PCT/US2014/033708
(87) 국제공개번호 WO 2014/169160
국제공개일자 2014년10월16일
(30) 우선권주장
13/862,277 2013년04월12일 미국(US)

(71) 출원인
마이크로소프트 테크놀로지 라이센싱, 엘엘씨
미국 워싱턴주 (우편번호 : 98052) 레드몬드 원 마이크로소프트 웨이
(72) 발명자
레디쉬 앤드류 더글라스
미국 워싱턴주 98052-6399 레드몬드 원 마이크로소프트 웨이 엘씨에이 - 인터내셔널 페턴츠 마이크로소프트 코퍼레이션 내
콜 올리비에
미국 워싱턴주 98052-6399 레드몬드 원 마이크로소프트 웨이 엘씨에이 - 인터내셔널 페턴츠 마이크로소프트 코퍼레이션 내
(뒷면에 계속)
(74) 대리인
김태홍, 김진희

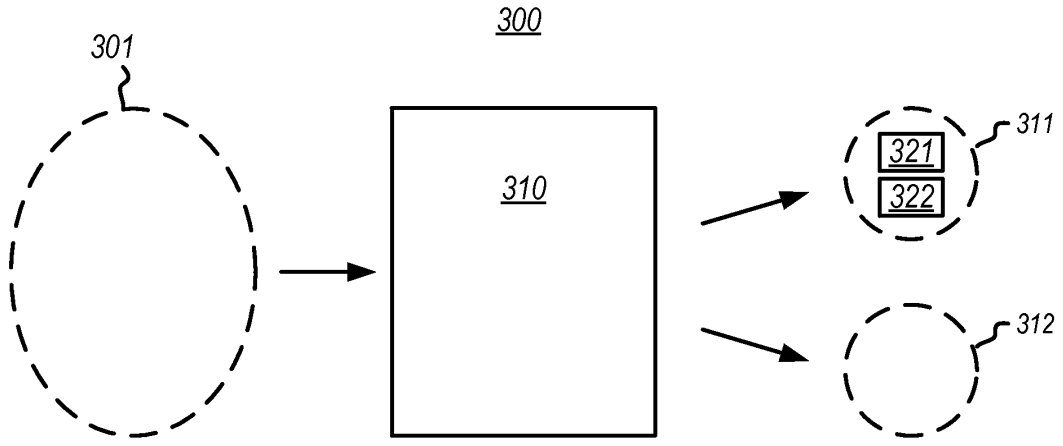
전체 청구항 수 : 총 10 항

(54) 발명의 명칭 **재계산 사용자 인터페이스에서의 변환의 컴파일**

(57) 요 약

변환 체인의 하나 이상의 표시된 결과를 포함하는 전자 캔버스를 표시하는 재계산 사용자 인터페이스의 변환 체인의 컴파일 변환 체인은 각자의 데이터 소스와 데이터 싱크 사이의 변환들을 포함한다. 재계산 사용자 인터페이스의 사용자 편집은 변환들 중 하나 이상을 재실행시키고, 그로써 재계산을 야기할 수 있을 것이다. 컴파일은 엔티티들 간의 종속성들의 종속성 그래프를 생성하기 위해 종속성들에 대해 재계산 사용자 인터페이스의 변환 체인을 분석하는 것을 포함한다. 예를 들어, 하나의 엔티티가 평가되는 경우, 다른 엔티티도 평가되어야 한다는 것을 나타내기 위해 엔티티들 사이에 어떤 종속성들이 있을 수 있다. 종속성 그래프는 이어서 하위 레벨의 실행 스텝들을 생성하는 데 사용된다. 종속성 그래프는, 종속성 그래프가 재계산 사용자 인터페이스의 동작 동안 이용 가능할 수 있도록, 프로그램에 대한 런타임에 추가로 제공된다.

대 표 도 - 도3



(52) CPC특허분류
G06F 8/433 (2013.01)
(72) 발명자
그뤼안 라두 비.
미국 워싱턴주 98052-6399 레드몬드 원 마이크로소프트 웨이 엘씨에이 - 인터내셔널 페턴츠 마이크로소프트 코퍼레이션 내
아누아르 니잠
미국 워싱턴주 98052-6399 레드몬드 원 마이크로소프트 웨이 엘씨에이 - 인터내셔널 페턴츠 마이크로소프트 코퍼레이션 내
사르카르 자이딥
미국 워싱턴주 98052-6399 레드몬드 원 마이크로소프트 웨이 엘씨에이 - 인터내셔널 페턴츠 마이크로소프트 코퍼레이션 내
미탈 비제이
미국 워싱턴주 98052-6399 레드몬드 원 마이크로소프트 웨이 엘씨에이 - 인터내셔널 페턴츠 마이크로소프트 코퍼레이션 내

명 세 서

청구범위

청구항 1

재계산 사용자 인터페이스의 변환 체인(transformation chain)을 컴파일하는 방법에 있어서,

엔티티들 간의 종속성들의 종속성 그래프(dependency graph)를 생성하기 위해 종속성들에 대해 상기 재계산 사용자 인터페이스의 상기 변환 체인을 분석하는 단계;

상기 종속성 그래프를 사용하여 하위 레벨의 실행 스텝들을 생성하는 단계; 및

상기 종속성 그래프를 프로그램에 대한 런타임(runtime)에 이용 가능하게 하는 단계를 포함하는, 재계산 사용자 인터페이스의 변환 체인을 컴파일하는 방법.

청구항 2

제1항에 있어서, 상기 종속성 그래프는 엔티티간 종속성 ID(inter-entity dependency identification)를 포함하고, 상기 엔티티간 종속성 ID로부터, 제1 엔티티가 평가되는 경우, 제2 엔티티도 평가되어야 하는 것으로 결정될 수 있는, 재계산 사용자 인터페이스의 변환 체인을 컴파일하는 방법.

청구항 3

제1항에 있어서, 상기 종속성 그래프는 사용자 이벤트 종속성(user event dependency)을 포함하고, 상기 사용자 이벤트 종속성으로부터, 사용자 이벤트가 발생하는 경우, 엔티티가 평가되어야 하는 것으로 결정될 수 있는, 재계산 사용자 인터페이스의 변환 체인을 컴파일하는 방법.

청구항 4

제1항에 있어서, 상기 재계산 사용자 인터페이스는 스프레드시트 문서인, 재계산 사용자 인터페이스의 변환 체인을 컴파일하는 방법.

청구항 5

제1항에 있어서, 상기 재계산 사용자 인터페이스는 상기 변환 체인으로의 입력 파라미터들 및 상기 변환 체인으로부터의 출력 파라미터들을 가지는 복합 컨트롤(complex control)을 가지는, 재계산 사용자 인터페이스의 변환 체인을 컴파일하는 방법.

청구항 6

제1항에 있어서, 상기 변환 체인은 선언적으로(declaratively) 표현되는, 재계산 사용자 인터페이스의 변환 체인을 컴파일하는 방법.

청구항 7

제6항에 있어서, 상기 하위 레벨의 실행 스텝들은 명령형 언어 코드(imperative language code)로 표현되는, 재계산 사용자 인터페이스의 변환 체인을 컴파일하는 방법.

청구항 8

컴퓨팅 시스템의 하나 이상의 프로세서들에 의해 실행될 때, 상기 컴퓨팅 시스템으로 하여금 컴파일 방법을 수행하게 하도록 구조화되어 있는 컴퓨터 실행 가능 명령어들을 가지는 하나 이상의 컴퓨터 판독 가능 저장 매체를 포함하는 컴퓨터 프로그램 제품에 있어서,

상기 컴파일 방법은,

재계산 사용자 인터페이스에서 데이터 소스로부터 데이터 싱크로의 변환을 분석하는 단계;

상기 변환의 종속성 그래프를 생성하는 단계로서, 상기 종속성 그래프는 이벤트와 엔티티 사이의 종속성을 표현

하는 것인 상기 변환의 종속성 그래프를 생성하는 단계;

상기 종속성 그래프를 사용하여 하위 레벨의 실행 스텝들을 생성하는 단계를 포함하는, 컴퓨터 프로그램 제품.

청구항 9

제8항에 있어서, 상기 컴파일 방법은,

상기 종속성 그래프를 프로그램에 대한 런타임에 이용 가능하게 하는 단계를 더 포함하는, 컴퓨터 프로그램 제품.

청구항 10

제8항에 있어서, 상기 데이터 싱크는 컨트롤인, 컴퓨터 프로그램 제품.

발명의 설명

배 경 기 술

[0001] "재계산 문서(recalculation document)"란 다양한 데이터 소스들 및 데이터 싱크들(sinks)을 보여주고 데이터 소스와 데이터 싱크 사이의 선언적 변환(declarative transformation)을 가능하게 하는 전자 문서를 말한다. 다양한 데이터 소스들과 데이터 싱크들을 상호 연결시키는 임의의 주어진 세트의 변환들에 대해, 데이터 소스의 출력이 데이터 싱크에 의해 사용될 수 있거나, 데이터 소스의 출력이, 데이터 싱크에 의해 사용되기 전에, 변환들을 거칠 수 있다. 이 다양한 변환들이 평가되고, 그 결과 재계산 문서 전체에 걸쳐 하나 이상의 출력들이 표현된다.

[0002] 사용자는 코딩에 대한 깊은 지식이 없어도 선언적 변환들을 추가하고 편집할 수 있다. 이러한 편집은 자동으로 변환들이 재계산되게 하여, 하나 이상의 출력들에 변화를 야기한다.

[0003] 재계산 문서의 구체적인 예는 셀들의 격자를 포함하는 스프레드시트 문서이다. 임의의 주어진 셀은 셀에 표시되는 특정의 값을 출력하기 위해 평가되는 표현식(expression)을 포함할 수 있다. 표현식은 하나 이상의 다른 셀들 또는 값들과 같은, 데이터 소스를 참조할 수 있다.

발명의 내용

[0004] 본 명세서에 기술된 적어도 일부 실시예들은 재계산 사용자 인터페이스의 변환 체인(transformation chain)의 컴파일에 관한 것이다. 변환 체인은 각자의 데이터 소스들과 데이터 싱크들 사이의 선언적 변환(declarative transform)들을 포함한다. 예를 들어, 스프레드시트와 관련하여, 데이터 싱크는 특정의 스프레드시트 셀일 수 있고, 변환은 특정의 셀과 연관된 표현식일 수 있으며, 데이터 소스는 표현식 내에서 참조되는 하나 이상의 다른 셀들 또는 특정의 값들일 수 있다. 재계산 사용자 인터페이스의 사용자 편집은 변환들 중 하나 이상이 재실행되게 하고, 그로써 재계산을 야기할 수 있을 것이다.

[0005] 컴파일은 엔티티들 간의 종속성들의 종속성 그래프(dependency graph)를 생성하기 위해 종속성들에 대해 재계산 사용자 인터페이스의 변환 체인을 분석하는 것을 포함한다. 예를 들어, 하나의 엔티티가 평가되는 경우, 다른 엔티티도 평가되어야 한다는 것을 나타내기 위해 엔티티들 사이에 어떤 종속성들이 있을 수 있다. 다른 종속성들은 엔티티의 평가가 의존하는 사용자 이벤트들을 명시할 수 있다. 종속성 그래프는 이어서 하위 레벨의 실행 스텝들을 생성하는 데 사용된다. 종속성 그래프는, 종속성 그래프가 재계산 사용자 인터페이스의 동작 동안 이용 가능할 수 있도록, 프로그램에 대한 런타임에 추가로 제공된다.

[0006] 이 발명의 내용은 청구된 발명 요지의 주요 특징들 또는 필수적인 특징들을 확인하기 위한 것이 아니며, 청구된 발명 요지의 범주를 정하는 데 보조 수단으로 사용되기 위한 것도 아니다.

도면의 간단한 설명

[0007] 앞서 언급된 장점들 및 특징들과 기타 장점들 및 특징들이 달성될 수 있는 방식을 설명하기 위해, 다양한 실시예들의 보다 상세한 설명이 첨부 도면을 참조하여 이루어질 것이다. 이들 도면이 예시적인 실시예만을 도시하고 있고 따라서 본 발명의 범주를 제한하는 것으로 간주되어서는 안된다는 것을 이해하면서, 첨부 도면을 사용하여 실시예들이 보다 구체적이고 상세하게 기술되고 설명될 것이다.

도 1은 본 명세서에 기술된 일부 실시예들이 이용될 수 있는 컴퓨팅 시스템을 추상적으로 나타낸 도면.

도 2는 몇 개의 데이터 소스들 및 데이터 싱크들 - 이들 사이에 변환들을 가짐 - 을 예시하고 본 명세서에 기술된 보다 폭넓은 원리들을 설명하기 위해 제공되는 구체적인 예로서 사용되는, 예시적인 재계산 사용자 인터페이스를 추상적으로 나타낸 도면.

도 3은 변환 체인에 액세스하고 컴파일된 코드는 물론 종속성 체인(dependency chain)을 생성하는 컴파일러를 포함하는 예시적인 컴파일 환경을 나타낸 도면.

도 4는 재계산 사용자 인터페이스의 변환 체인을 컴파일하는 방법의 플로우차트를 나타낸 도면.

도 5는 입력 데이터에 의존하는 뷰 합성(view composition)을 생성하는 데이터 기반 합성 프레임워크(data-driven composition framework)를 포함하는 본 발명의 원리들이 이용될 수 있는 환경을 나타낸 도면.

도 6은 도 5의 환경의 하나의 예를 나타내는 파이프라인 환경을 나타낸 도면.

도 7은 도 6의 파이프라인의 데이터 부분의 일 실시예를 개략적으로 나타낸 도면.

도 8은 도 6의 파이프라인의 분석 부분의 일 실시예를 개략적으로 나타낸 도면.

도 9는 도 6의 파이프라인의 뷰 부분의 일 실시예를 개략적으로 나타낸 도면.

발명을 실시하기 위한 구체적인 내용

[0008] 본 명세서에 기술된 적어도 일부 실시예들은 재계산 사용자 인터페이스의 변환 체인의 컴파일에 관한 것이다. 재계산 사용자 인터페이스는, 예를 들어, 스프레드시트와 같은 재계산 문서일 수 있을 것이다. 그렇지만, 재계산 사용자 인터페이스는 변환 체인의 하나 이상의 표시된 결과들을 포함하는 임의의 표시된 전자 캔버스(electronic canvas)일 수 있다. 변환 체인은 각자의 데이터 소스과 데이터 싱크 사이의 복수의 변환들을 포함한다. 재계산 사용자 인터페이스의 사용자 편집은 변환들 중 하나 이상이 재실행되게 하고, 그로써 재계산을 야기할 수 있을 것이다.

[0009] 컴파일은 엔티티들 간의 종속성들의 종속성 그래프를 생성하기 위해 종속성들에 대해 재계산 사용자 인터페이스의 변환 체인을 분석하는 것을 포함한다. 예를 들어, 하나의 엔티티가 평가되는 경우, 다른 엔티티도 평가되어야 한다는 것을 나타내기 위해 엔티티들 사이에 어떤 종속성들이 있을 수 있다. 종속성 그래프는 이어서 하위 레벨의 실행 스텝들을 생성하는 데 사용된다. 종속성 그래프는, 종속성 그래프가 재계산 사용자 인터페이스의 동작 동안 이용 가능할 수 있도록, 프로그램에 대한 런타임에 추가로 제공된다.

[0010] 컴퓨팅 시스템에 대한 어떤 서론적 논의가 도 1과 관련하여 기술될 것이다. 이어서, 재계산 사용자 인터페이스의 변환 체인의 컴파일이 후속 도면들과 관련하여 기술될 것이다.

[0011] 컴퓨팅 시스템은 이제 점점 아주 다양한 형태들을 취하고 있다. 컴퓨팅 시스템은, 예를 들어, 핸드헬드 디바이스, 가전 제품(appliance), 랩톱 컴퓨터, 데스크톱 컴퓨터, 메인 프레임, 분산 컴퓨팅 시스템, 또는 심지어 종래에 컴퓨팅 시스템으로 간주되지 않았던 디바이스일 수 있다. 이 설명 및 청구 범위에서, "컴퓨팅 시스템"이라는 용어는 적어도 하나의 물리적 및 유형적 프로세서, 그리고 프로세서에 의해 실행될 수 있는 컴퓨터 실행 가능 명령어들을 가질 수 있는 물리적 및 유형적 메모리를 포함하는 임의의 디바이스 또는 시스템(또는 이들의 조합)을 포함하는 것으로 광의적으로 정의된다. 메모리는 임의의 형태를 취할 수 있고, 컴퓨팅 시스템의 특성 및 형태에 의존할 수 있다. 컴퓨팅 시스템은 네트워크 환경에 걸쳐 분산되어 있을 수 있고 다수의 구성 컴퓨팅 시스템(constituent computing system)들을 포함할 수 있다.

[0012] 도 1에 예시된 바와 같이, 가장 기본적인 구성에서, 컴퓨팅 시스템(100)은 전형적으로 적어도 하나의 처리 유닛(102) 및 메모리(104)를 포함한다. 메모리(104)는 휘발성, 비휘발성, 또는 이 둘의 어떤 조합일 수 있는 물리적 시스템 메모리(physical system memory)일 수 있다. "메모리"라는 용어는 또한 본 명세서에서 물리적 저장 매체(physical storage media) 등의 비휘발성 대용량 저장소를 말하는 데 사용될 수 있다. 컴퓨팅 시스템이 분산되어 있는 경우, 처리, 메모리 및/또는 저장 기능도 분산되어 있을 수 있다. 본 명세서에서 사용되는 바와 같이, "실행 가능 모듈" 또는 "실행 가능 구성요소"라는 용어는 컴퓨팅 시스템 상에서 실행될 수 있는 소프트웨어 객체(software object), 루틴 또는 메서드(method)를 지칭할 수 있다. 본 명세서에 기술된 상이한 구성요소들, 모듈들, 엔진들 및 서비스들이 컴퓨팅 시스템 상에서 (예컨대, 개별적인 스레드들로서) 실행되는 객체들 또는 프로세스들로서 구현될 수 있다.

[0013] 이하의 설명에서, 하나 이상의 컴퓨팅 시스템들에 의해 수행되는 동작(act)들을 참조하여 실시예들이 기술된다. 이러한 동작들이 소프트웨어로 구현되는 경우, 동작을 수행하는 연관된 컴퓨팅 시스템의 하나 이상의 프로세서들은 컴퓨터 실행 가능 명령어들을 실행한 것에 응답하여 컴퓨팅 시스템의 동작을 지시한다. 예를 들어, 이러한 컴퓨터 실행 가능 명령어들은 컴퓨터 프로그램 제품을 형성하는 하나 이상의 컴퓨터 판독 가능 매체 상에 구현될 수 있다. 이러한 동작의 일례는 데이터의 조작(manipulation)을 포함한다. 컴퓨터 실행 가능 명령어들(및 조작된 데이터)는 컴퓨팅 시스템(100)의 메모리(104)에 저장될 수 있다. 컴퓨팅 시스템(100)은 또한 컴퓨팅 시스템(100)이, 예를 들어, 네트워크(110)를 통해 다른 메시지 프로세서들과 통신할 수 있게 하는 통신 채널들(108)을 포함할 수 있다. 컴퓨팅 시스템(100)은 또한 시각적 표현들을 사용자에게 디스플레이하는 데 사용될 수 있는 디스플레이(112)를 포함한다.

[0014] 본 명세서에 기술된 실시예들은, 예를 들어, 이하에서 더 상세히 논의되는 바와 같이, 하나 이상의 프로세서들 및 시스템 메모리와 같은, 컴퓨터 하드웨어를 비롯한 특수 목적 또는 범용 컴퓨터를 포함하거나 이용할 수 있다. 본 명세서에 기술된 실시예들은 또한 컴퓨터 실행 가능 명령어들 및/또는 데이터 구조들을 전달하거나 저장하는 물리적 및 다른 컴퓨터 판독 가능 매체를 포함한다. 이러한 컴퓨터 판독 가능 매체는 범용 또는 특수 목적 컴퓨터 시스템에 의해 액세스될 수 있는 임의의 이용가능한 매체일 수 있다. 컴퓨터 실행 가능 명령어들을 저장하는 컴퓨터 판독 가능 매체는 물리적 저장 매체이다. 컴퓨터 실행 가능 명령어들을 전달하는 컴퓨터 판독 가능 매체는 전송 매체이다. 이와 같이, 제한이 아닌 예로서, 본 발명의 실시예들은 적어도 2 개의 뚜렷하게 상이한 종류의 컴퓨터 판독 가능 매체 - 즉, 컴퓨터 저장 매체 및 전송 매체 - 를 포함할 수 있다.

[0015] 컴퓨터 저장 매체는 컴퓨터 실행 가능 명령어들 또는 데이터 구조들의 형태로 되어 있는 원하는 프로그램 코드 수단을 저장하는 데 사용될 수 있고 범용 또는 특수 목적 컴퓨터에 의해 액세스될 수 있는 RAM, ROM, EEPROM, CD-ROM 또는 다른 광 디스크 저장소, 자기 디스크 저장소 또는 다른 자기 저장 디바이스, 또는 임의의 다른 유형적 매체를 포함한다.

[0016] "네트워크"는 컴퓨터 시스템들 및/또는 모듈들 및/또는 다른 전자 디바이스들 간의 전자 데이터의 전송을 가능하게 하는 하나 이상의 데이터 링크들로서 정의된다. 정보가 네트워크 또는 다른 통신 연결(유선, 무선 또는 유선 또는 무선의 조합)을 통해 컴퓨터로 전송되거나 제공될 때, 컴퓨터는 당연히 그 연결을 전송 매체로 본다. 전송 매체는 컴퓨터 실행 가능 명령어들 또는 데이터 구조들의 형태의 원하는 프로그램 코드 수단을 전달하는 데 사용될 수 있고 범용 또는 특수 목적 컴퓨터에 의해 액세스될 수 있는 네트워크 및/또는 데이터 링크들을 포함할 수 있다. 상기한 것들의 조합들도 컴퓨터 판독 가능 매체의 범주 내에 포함되어야 한다.

[0017] 게다가, 다양한 컴퓨터 시스템 구성요소들에 도달할 때, 컴퓨터 실행 가능 명령어들 또는 데이터 구조들의 형태로 되어 있는 프로그램 코드 수단이 전송 매체로부터 컴퓨터 저장 매체로(또는 그 반대로) 자동으로 전달될 수 있다. 예를 들어, 네트워크 또는 데이터 링크를 통해 수신되는 컴퓨터 실행 가능 명령어들 또는 데이터 구조들이 네트워크 인터페이스 모듈(예컨대, "NIC") 내의 RAM에 버퍼링될 수 있고, 이어서 궁극적으로 컴퓨터 시스템 RAM으로 그리고/또는 컴퓨터 시스템에 있는 저휘발성(less volatile) 컴퓨터 저장 매체로 전달될 수 있다. 따라서, 컴퓨터 저장 매체가 또한(또는 심지어 주로) 전송 매체를 이용하는 컴퓨터 시스템 구성요소들에 포함될 수 있다는 것을 잘 알 것이다.

[0018] 컴퓨터 실행 가능 명령어들은, 예를 들어, 프로세서에서 실행될 때, 범용 컴퓨터, 특수 목적 컴퓨터, 또는 특수 목적 처리 디바이스로 하여금 특정의 기능 또는 그룹의 기능들을 수행하게 하는 명령어들 및 데이터를 포함한다. 컴퓨터 실행 가능 명령어들은, 예를 들어, 바이너리(binary), 어셈블리어와 같은 중간 형식 명령어(intermediate format instruction), 또는 심지어 소스 코드일 수 있다. 발명의 요지가 구조적 특징들 및/또는 방법 동작들과 관련하여 기술되어 있지만, 첨부된 청구범위에 한정된 발명 요지가 상기한 기술된 특징들 또는 동작들로 꼭 제한되는 것은 아님을 잘 알 것이다. 오히려, 기술된 특징들 및 동작들은 청구항들을 구현하는 예시적인 형태들로서 개시되어 있다.

[0019] 통상의 기술자라면 본 발명이 개인용 컴퓨터, 데스크톱 컴퓨터, 랩톱 컴퓨터, 메시지 프로세서, 핸드헬드 디바이스, 멀티프로세서 시스템, 마이크로프로세서 기반 또는 프로그램 가능 가전 제품, 네트워크 PC, 미니컴퓨터, 메인프레임 컴퓨터, 휴대폰, PDA, 페이저, 라우터, 스위치 등을 비롯한, 많은 유형의 컴퓨터 시스템 구성들을 갖는 네트워크 컴퓨팅 환경들에서 실시될 수 있다는 것을 잘 알 것이다. 본 발명은 또한 네트워크를 통해 (유선 데이터 링크, 무선 데이터 링크에 의해, 또는 유선 및 무선 데이터 링크들의 조합에 의해) 연결되어 있는 로컬 및 원격 컴퓨터 시스템들 둘 다가 작업들을 수행하는 분산 시스템 환경들에서 실시될 수 있다. 분산 시스템 환경에서, 프로그램 모듈들은 로컬 및 원격 메모리 저장 디바이스들 둘 다에 배치될 수 있다.

[0020] 이 설명에서 그리고 청구범위에서, "재계산 사용자 인터페이스"는, 사용자와 상호 작용할 수 있고 하나 이상의 데이터 소스들 및 하나 이상의 데이터 싱크들이 있는 환경에서 나타나는 인터페이스이다. 게다가, 각각이 하나 이상의 데이터 소스들과 데이터 싱크들 사이에 선언적으로 정의될 수 있는 한 세트의 변환들이 있다. 예를 들어, 하나의 데이터 소스의 출력이 변환에 피드되고, 변환으로부터의 결과가 이어서 데이터 싱크에 제공되며, 그 결과 어쩌면 사용자에 대한 시각화(visualization)에 어떤 종류의 변화가 있게 된다.

[0021] 변환들은, 구체적인 코딩 지식을 갖지 않은 사용자가 변환을 정의하는 선언들을 작성할 수 있다는 의미에서, "선언적"이다. 변환이 선언적으로 정의되기 때문에, 사용자는 선언적 변환을 변경할 수 있다. 그에 응답하여, 재계산이 수행되고, 그 결과 아마도 상이한 데이터가 데이터 싱크들에 제공된다.

[0022] 재계산 사용자 인터페이스의 전형적인 예는 스프레드시트 문서이다. 스프레드시트 문서는 셀들의 격자를 포함한다. 처음에, 셀들은 비어 있고, 따라서 스프레드시트 프로그램의 임의의 셀은, 사용자에 의해 입력되는 선언적 표현식들의 의미 및 문맥에 따라, 데이터 소스 또는 데이터 싱크가 될 가능성을 가진다. 예를 들어, 사용자는 주어진 셀을 선택하고, 그 셀에 표현식을 타이핑할 수 있을 것이다. 표현식은 그 셀에 할당될 표현된 스칼라 값과 같이 간단할 수 있을 것이다. 그 셀이 나중에 데이터 소스로서 사용될 수 있다. 대안적으로, 주어진 셀에 대한 표현식이 하나 이상의 다른 셀들로부터 입력 값들을 받는 방정식의 형태로 되어 있을 수 있다. 그 경우에, 주어진 셀은 변환의 결과를 디스플레이하는 데이터 싱크이다. 그렇지만, 계속된 작성(authoring) 동안, 그 셀이 작성자에 의해 선언적으로 행해진 또 다른 변환들에 대한 데이터 싱크로서 사용될 수 있다.

[0023] 스프레드시트 문서의 작성자가 명령형 코드(imperative code)에 관한 전문가일 필요는 없다. 작성자는 단순히 변환을 정의하는 선언들을 하고 대응하는 데이터 싱크들 및 데이터 소스들을 선택하기만 한다. 이후에 기술되는 도 5 내지 도 9는 보다 일반화된 재계산 사용자 인터페이스가 기술되는 보다 일반화된 선언적 작성 환경을 제공한다. 그 차후에 기술되는 환경에서, 시각화된 컨트롤(visualized control)들은 데이터 소스들 및 데이터 싱크들 둘 다로서 역할할 수 있다. 게다가, 선언적 변환들이 그 컨트롤들의 간단한 조작들에 의해 보다 직관적으로 작성될 수 있다.

[0024] 도 2는 본 명세서에 기술된 보다 폭넓은 원리들을 설명하기 위해 제공되는 구체적인 예인, 예시적인 재계산 사용자 인터페이스(200)를 추상적으로 나타낸 것이다. 본 명세서에 기술된 원리들이 무수한 각종의 애플리케이션들에 대한 무수한 각종의 재계산 사용자 인터페이스들을 생성하기 위해 임의의 재계산 사용자 인터페이스에 적용될 수 있기 때문에, 재계산 사용자 인터페이스(200)는 일례에 불과하다.

[0025] 재계산 사용자 인터페이스(200)는 몇 개의 선언적 변환들(211 내지 215)을 포함한다. 변환들(211 내지 215)을 나타내는 화살표들 각각의 주위에 있는 파선 원은 변환들 각각이 선언적 형태로 되어 있다는 것을 기호로 나타낸 것이다.

[0026] 도 2의 이 구체적인 예에서, 변환(211)은 각자의 데이터 소스(201) 및 데이터 싱크(202)를 포함한다. 유의할 점은, 하나의 변환에 대한 데이터 싱크가 또한 다른 변환에 대한 데이터 소스일 수 있다는 것이다. 예를 들어, 변환(211)에 대한 데이터 싱크(202)는 또한 변환(212)에 대한 데이터 소스로서 역할한다. 게다가, 변환이 다수의 데이터 소스들을 가질 수 있다. 이와 같이, 변환 체인(transform chain)이 계층적으로 될 수 있고, 따라서 꽤 복잡할 수 있다. 예를 들어, 변환(212)은 데이터 소스(202) 및 데이터 싱크(203)를 포함한다. 데이터 싱크(203)는 2 개의 데이터 소스들 - 즉, 변환(212)에 대한 데이터 소스(202) 및 변환(214)에 대한 데이터 소스(205) - 을 포함한다. 그렇다 해도, 아마도 단일의 변환이 2 개의 데이터 소스들(202 및 205)을 데이터 싱크(203)로 인도한다. 변환(213)은 데이터 소스(204) 및 데이터 싱크(205)를 포함한다.

[0027] 재계산 사용자 인터페이스가, 예를 들어, 스프레드시트 문서인 경우, 다양한 데이터 소스들/싱크들(201 내지 205)이 스프레드시트 셀들일 수 있고, 이 경우에 변환들은 각각의 데이터 싱크와 연관될 표현식을 나타낸다. 각각의 표현식의 출력이 셀 내에 표시된다. 이와 같이, 스프레드시트의 경우에, 데이터 소스들/싱크들이 변환 체인으로의 입력 파라미터들 및 변환 체인으로부터의 출력 파라미터들 둘 다를 포함하는 복합적인 시각화 컨트롤들일 수 있다. 예를 들어, 도 2에서, 데이터 소스(205)로부터 데이터 싱크(201)로 가는 부가의 선언적 변환(215)이 있다. 이와 같이, 데이터 소스/싱크(201)는 변환(215)으로부터의 출력을 나타내는 정보를 시각화하는 것은 물론, 추가의 데이터를 다른 데이터 싱크들에 제공할 수 있다.

[0028] 재계산 사용자 인터페이스는 시각화 컨트롤들을 가질 필요가 없다. 이것의 하나의 예는, 통상의 경우에 계산에 관한 어떤 정보도 사용자에게 디스플레이됨이 없이, 소스 데이터를 사용하여 싱크 데이터를 업데이트하는 변환 기반 계산을 수행하도록 되어 있는 재계산 사용자 인터페이스이다. 예를 들어, 재계산 사용자 인터페이스는 백

그라운드 계산(background computation)을 지원할 수 있다. 제2 예는 공정 제어 예에서의 밸브들과 같은 외부 액추에이터들을 조작하는 출력 컨트롤들을 가지는 재계산 사용자 인터페이스이다. 이러한 컨트롤들은 그 상태들이 변환 계산의 결과들 및 온 신호 입력(on signal input)들에 의해 제어된다는 점에서 디스플레이 컨트롤들과 같다. 그렇지만, 여기서, 출력은 디스플레이에 대한 시각화보다는 디바이스에 대한 제어 신호이다. 예를 들어, 로봇을 제어하기 위한 재계산 사용자 인터페이스를 생각해보자. 이 재계산 사용자 인터페이스는 서보 위치 및 속도, 초음파 거리 측정 측정치 등과 같은 로봇 센서들의 입력들에 의존하는 로봇 행동들 및 거동에 대한 규칙들을 가질 수 있을 것이다. 또는, 밸브 위치, 유체 유량 등과 같은 장비 센서들로부터의 신호들을 받는 재계산 사용자 인터페이스에 기초한 공정 제어 응용을 생각해보자.

[0029] 도 3은 변환 체인(301)에 액세스하는 컴파일러(310)를 포함하는 예시적인 컴파일 환경(300)을 나타낸 것이다. 변환 체인(301)의 일례는 도 2의 변환 체인(200)이다. 도 4는 재계산 사용자 인터페이스의 변환 체인을 컴파일하는 방법(400)의 플로우차트를 나타낸 것이다. 방법(400)은 도 3의 컴파일러(310)에 의해 수행될 수 있다. 하나의 실시예에서, 방법(400)은 프로세서(들)(102)가 하나 이상의 컴퓨터 판독 가능 저장 매체에 임베딩되어 있는 컴퓨터 실행 가능 명령어들을 실행한 것에 응답하여 컴퓨팅 시스템(100)에 의해 수행될 수 있다.

[0030] 방법(400)은 종속성들에 대해 재계산 사용자 인터페이스의 변환 체인을 분석하는 것[동작(401)]을 포함한다. 예를 들어, 도 2를 참조하면, 컴파일러(300)는 변환들(211 내지 215) 각각을 분석할 수 있을 것이다. 변환들은 선언적이고, 따라서 변환들이 명령형 컴퓨터 언어(imperative computer language)를 사용하여 표현되는 경우보다 종속성들이 더 쉽게 추출될 수 있다.

[0031] 분석에 기초하여, 변환들에서 참조되는 엔티티들 간의 종속성 그래프가 생성된다[동작(402)]. 기본적으로, 종속성들은 이벤트를 표현하는 소스 엔티티와, 대상 엔티티의 평가가 그 이벤트에 의존한다는 것을 표현하는 그 대상 엔티티를 가진다. 이벤트의 일례는 사용자가 재계산 사용자 인터페이스와 특정 방식으로 상호 작용하는 사용자 이벤트일 수 있다. 다른 예로서, 이벤트는 소스 엔티티가 평가되는 경우, 종속성의 대상 엔티티도 평가되어야 하는 엔티티간 이벤트(inter-entity event)일 수 있다.

[0032] 컴파일러는 이어서 종속성 그래프에 기초하여 하위 레벨 실행 스텝들을 생성한다[동작(403)]. 하위 레벨 실행 스텝들은, 예를 들어, 명령형 언어 코드(imperative language code)일 수 있다. 명령형 언어 코드는 이벤트들을 검출하고, 이벤트 차트(event chart)를 참조하여 실행할 함수를 결정하며, 그 함수를 실행하는 것으로 응답하도록 구성되어 있다. 그에 따라, 종속성 그래프에서의 종속성들 각각이 함수로 될 수 있다. 종속성 그래프 자체가 런타임에 제공될 수 있다[동작(404)]. 명령형 언어 코드는, 예를 들어, JAVASCRIPT와 같은 스크립트 언어일 수 있다. 그렇지만, 본 명세서에 기술된 원리들은 명령형 언어 코드가 임의의 특정의 언어인 것으로 제한되지 않는다

[0033] 일례로서, 도 3은 컴파일러(310)가 하위 레벨 코드(311)도 생성하는 것을 나타내고 있다. 이러한 하위 레벨 코드(311)는 변환 체인에서의 변환들 각각의 컴파일을 포함한다. 예를 들어, 하위 레벨 코드(311)는 변환 체인에서의 변환들 각각의 컴파일을 나타내는 요소(321)를 포함하는 것으로 나타내어져 있다. 도 2와 관련하여, 요소(321)는 변환들(211 내지 215) 각각의 컴파일을 포함할 것이다. 하위 레벨 코드(311)는 또한 각종의 함수들(322)을 포함한다. 종속성 그래프에서의 각각의 종속성에 대해 함수가 생성된다. 함수들은 명령형 언어 함수(imperative language function)들일 수 있다.

[0034] 명령형 언어 런타임(imperative language runtime)이 종속성 그래프에 열거되어 있는 이벤트를 검출할 때, 컴파일된 함수들(322) 내의 대응하는 함수가 또한 실행된다. 그에 따라, 모든 변환들이 적절히 컴파일되는 것, 및 특정의 이벤트들에 대한 종속성들 각각이 전용 함수들에 의해 시행되는 것에 의해, 선언적 재계산 사용자 인터페이스가 명령형 언어 코드로서 적절히 표현된다.

[0035] 그에 따라, 선언적 재계산 사용자 인터페이스를 컴파일하는 효과적인 메커니즘이 기술되었다. 그에 부가하여, 런타임에는, 보다 규모가 큰 인터프리터(interpreter)보다는, 종속성 그래프가 제공된다.

[0036] 프로그래머가 아닌 사람들이 복잡한 거동(complex behavior)들을 가지는 프로그램들을 재계산 사용자 인터페이스를 사용하여 작성할 수 있게 하는 작성 파이프라인(authoring pipeline)의 구체적인 예가 이제부터 도 5 내지 도 9와 관련하여 기술될 것이다.

[0037] 도 5는 상호작용적인 시각적 합성(composition)을 재계산 사용자 인터페이스의 형태로 생성하기 위해 사용될 수 있는 시각적 합성 환경(500)을 나타낸 것이다. 재계산 사용자 인터페이스의 구성이 데이터 기반 분석 및 분석 결과들의 시각화를 사용하여 수행된다. 환경(500)은 뷰 합성(530)의 문제 도메인(problem-domain)과 무관하게

수행되는 논리를 수행하는 합성 프레임워크(510)를 포함한다. 예를 들어, 동일한 합성 프레임워크(510)가 도시 계획, 분자 모델, 매장 진열대 레이아웃, 기계 성능 또는 조립 분석, 또는 다른 도메인 관련 렌더링들에 대한 상호작용적 뷰 합성들을 구성하는 데 사용될 수 있다.

[0038] 그렇지만, 합성 프레임워크(510)는 도메인에 특유한 실제의 시각적 합성(530)을 생성하기 위해 도메인 관련 데이터(520)를 사용한다. 그에 따라, 합성 프레임워크(510) 자체를 재코딩해야 하는 것보다는, 도메인 관련 데이터(520)를 변경하는 것에 의해 다수의 상이한 도메인들에 대한 재계산 사용자 인터페이스들에 대해 동일한 합성 프레임워크(510)가 사용될 수 있다. 이와 같이, 파이프라인(500)의 합성 프레임워크(510)가, 재코딩 및 재컴파일보다는 데이터를 변경하는 것에 의해, 어쩌면 무한한 수의 문제 도메인들에 또는 적어도 아주 다양한 문제 도메인들에 적용될 수 있다. 뷰 합성(530)은 이어서 적절한 2-D 또는 3-D 렌더링 모듈에 명령어들로서 제공될 수 있다. 본 명세서에 기술된 아키텍처는 또한 기존의 뷰 합성 모델들을 새로운 뷰 합성 모델들에 구성 요소(building block)들로서 편리하게 포함시킬 수 있게 한다. 하나의 실시예에서, 어떤 모델에 대한 2 개의 가능한 해결책들 간의 용이한 비교를 가능하게 하기 위해 다수의 뷰 합성들이 통합 뷰 합성(integrated view composition)에 포함될 수 있다.

[0039] 도 6은 파이프라인 환경(600) 형태의 합성 프레임워크(510)의 예시적인 아키텍처를 나타낸 것이다. 파이프라인 환경(600)은, 그 중에서도 특히, 파이프라인(601) 자체를 포함한다. 파이프라인(601)은 데이터 부분(data portion)(610), 분석 부분(analytics portion)(620), 및 뷰 부분(view portion)(630)을 포함하고, 이들 각각은, 각각, 후속 도면들 도 7 내지 도 9 및 첨부 설명과 관련하여 상세히 기술될 것이다. 지금으로서는, 일반적인 레벨에서, 파이프라인(601)의 데이터 부분(610)은 각종의 상이한 유형의 데이터를 받을 수 있고, 그 데이터를 정규 형태(canonical form)로 파이프라인(601)의 분석 부분(620)에 제공한다. 분석 부분(620)은 데이터를 다양한 모델 파라미터들에 바인딩시키고, 모델 분석을 사용하여 모델 파라미터들에서의 미지수들에 대한 해를 구한다. 다양한 파라미터 값들이 이어서 뷰 부분(630)에 제공되고, 뷰 부분(630)은 모델 파라미터들의 그 값들을 사용하여 합성 뷰(composite view)를 생성한다.

[0040] 파이프라인 환경(600)은 또한 파이프라인(601)의 작성자 또는 다른 사용자가 파이프라인(601)에 제공할 데이터를 작성(formulate) 및/또는 선택할 수 있게 하는 작성 구성요소(authoring component)(640)를 포함한다. 예를 들어, 작성 구성요소(640)는 데이터를 데이터 부분(610)[입력 데이터(611)로 표현됨], 분석 부분(620)[분석 데이터(621)로 표현됨], 및 뷰 부분(630)[뷰 데이터(631)로 표현됨] 각각에 제공하는 데 사용될 수 있다. 다양한 데이터(611, 621 및 631)는 도 5의 도메인 관련 데이터(520)의 일례를 나타내고, 이하에서 훨씬 더 상세히 기술될 것이다. 작성 구성요소(640)는, 예를 들어, 데이터 스키마(data schema), 모델에 의해 사용될 실제 데이터, 외부 소스들로부터 가져와야 하는 데이터의 위치 또는 가능한 위치들의 범위, 시각적(그래픽 또는 애니메이션) 객체들, 시각적 모델링 명령문(visual, modeling statement)들(예컨대, 뷰, 방정식, 제약조건)에 대해 수행될 수 있는 사용자 인터페이스 상호작용들, 바인딩들 등을 비롯한 아주 다양한 데이터를 제공하는 것을 지원한다. 하나의 실시예에서, 작성 구성요소는 (도 6에 도시되어 있지 않지만, 도 5의 합성 프레임워크(510)로 나타내어진) 전체 관리자 구성요소에 의해 제공되는 기능의 한 부분에 불과하다. 관리자는 이벤트들[사용자 상호작용 이벤트, 외부 데이터 이벤트, 그리고 솔버(solver), 운영 체제 등과 같은 다른 구성요소들 중 임의의 것으로부터의 이벤트 등]에 응답하여 모든 다른 구성요소들[데이터 커넥터(data connector), 솔버, 뷰어, 기타 등등]의 동작을 제어하고 시퀀싱하는 총 감독이다.

[0041] 도 6의 파이프라인 환경(600)에서, 작성 구성요소(640)는 기존의 파이프라인(601)에 데이터를 제공하는 데 사용되고, 여기서 이 데이터는 입력 데이터를 정의하는 것부터 분석 모델(앞서 "변환 체인"이라고 지칭됨)을 정의하는 것, 변환 체인의 결과들이 뷰 합성에서 어떻게 시각화되는지를 정의하는 것까지의 프로세스 전체를 주도한다. 그에 따라, 파이프라인(601)을 아주 다양한 도메인들 및 문제들 중 임의의 것에 적응시키기 위해 어떤 코딩도 수행할 필요가 없다. 파이프라인(601)에 제공되는 데이터만이 파이프라인(601)을 적용하여, 전혀 상이한 문제 도메인으로부터 상이한 뷰 합성을 시각화하기 위해 또는 어쩌면 기존의 도메인에 대한 문제 해결(problem solving)을 조절하기 위해 변하는 것이다. 게다가, 데이터가 사용 시에(즉, 런타임 시에)는 물론, 작성 시에 변경될 수 있기 때문에, 모델이 런타임 시에 수정 및/또는 확장될 수 있다. 이와 같이, 모델을 작성하는 것과 모델을 실행하는 것 간의 차이(있는 경우)가 거의 없다. 모든 작성이 데이터 항목들을 편집하는 것을 포함하기 때문에 그리고 소프트웨어가 데이터로부터 그것의 거동의 전부를 실행하기 때문에, 데이터에 대한 모든 변경은 재코딩 및 재컴파일의 필요 없이 즉각 거동에 영향을 미친다.

[0042] 파이프라인 환경(600)은 또한 사용자가 표시된 뷰 합성과 상호작용한 때를 검출하고 이어서 그에 응답하여 무엇을 할지를 결정하는 사용자 상호작용 응답 모듈(650)을 포함한다. 예를 들어, 어떤 유형의 상호작용들은 파이프

라인(601)에 제공된 데이터의 어떤 변화도 필요로 하지 않고, 따라서 뷰 합성에 대한 어떤 변화도 필요로 하지 않을 수 있다. 다른 유형의 상호작용들은 데이터(611, 621, 또는 631) 중 하나 이상을 변경할 수 있다. 그 경우에, 이 새로운 또는 수정된 데이터는 새로운 입력 데이터가 데이터 부분(610)에 제공되게 할 수 있고, 분석 부분(620)에 의한 입력 데이터의 재분석을 필요로 할 수 있으며, 그리고/또는 뷰 부분(630)에 의한 뷰 합성의 재시각화를 필요로 할 수 있다.

[0043] 그에 따라, 파이프라인(601)은 데이터 기반 분석 시각화들을 어쩌면 무한한 수의 문제 도메인들로 또는 적어도 아주 다양한 문제 도메인들로 확장시키는 데 사용될 수 있다. 게다가, 아주 다양한 문제들을 해결하기 위해 뷰 합성을 변경하는 사람이 프로그래머일 필요가 없다. 파이프라인(601)의 데이터 부분(610), 분석 부분(620) 및 뷰 부분(630) 각각이 이제부터 도 7의 각자의 데이터 부분(700), 도 8의 분석 부분(800), 및 도 9의 뷰 부분(900)과 관련하여, 그 순서로 기술될 것이다. 도 7 내지 도 9로부터 명백할 것인 바와 같이, 파이프라인(601)은 일련의 변환 구성요소들로서 구성될 수 있고, 여기서 그들 각각은, 1) 어떤 적절한 입력 데이터를 수신하고, 2) 그 입력 데이터에 응답하여 어떤 동작을 수행하며(입력 데이터에 대해 변환을 수행하는 것 등), 3) 데이터를 출력하고, 이 데이터는 이어서 다음 변환 구성요소에의 입력 데이터로서 역할한다.

[0044] 도 7은 도 6의 파이프라인(601)의 데이터 부분(700)의 많은 가능한 실시예들 중 단지 하나를 나타낸 것이다. 데이터 부분(700)의 기능들 중 하나는 도 8과 관련하여 논의되는 파이프라인의 분석 부분(800)에 의해 이해되는 스키마들에 부합하는 정규 형식(canonical format)으로 데이터를 제공하는 것이다. 데이터 부분은 이질적 데이터(heterogenic data)(701)에 액세스하는 데이터 액세스 구성요소(710)를 포함한다. 입력 데이터(701)는, 데이터가 데이터 액세스 구성요소(710)에 정규 형태로 제공될 수 있다(그러나 그럴 필요가 없음)는 점에서, "이질적"일 수 있다. 실제로, 이질적 데이터가 아주 다양한 형식들일 수 있도록 데이터 부분(700)이 구조화되어 있다. 모델들에 의해 액세스되어 처리될 수 있는 상이한 종류의 도메인 데이터의 예들은 텍스트 및 XML 문서, 테이블, 목록, 계층구조(트리), SQL 데이터베이스 쿼리 결과, BI(business intelligence) 큐브 쿼리 결과, 다양한 형식들로 된 2D 드로잉 및 3D 비주얼 모델과 같은 그래픽 정보, 그리고 이들의 조합들[즉, 합성(composite)]을 포함한다. 게다가, 액세스될 수 있는 데이터의 종류가, 액세스될 데이터에 대한 정의(예컨대, 스키마)를 제공하는 것에 의해, 선언적으로 확장될 수 있다. 그에 따라, 데이터 부분(700)은 모델에의 아주 다양한 이질적 입력을 허용하고, 또한 액세스 가능 데이터 유형들의 런타임 선언적 확장을 지원한다.

[0045] 하나의 실시예에서, 데이터 액세스 부분(710)은 다수의 상이한 데이터 소스들로부터 데이터를 획득하기 위한 다수의 커넥터(connector)들을 포함한다. 커넥터의 주된 기능들 중 하나가 대응하는 데이터를 정규 형태로 만드는 것이기 때문에, 이러한 커넥터들은 이후에 그리고 도면들에서 종종 "정규화기(canonicalizer)"라고 지칭될 것이다. 각각의 정규화기는 그의 대응하는 데이터 소스의 특정 API(Application Program Interface)들을 이해할 수 있다. 정규화기는 또한 데이터를 데이터 소스로부터 읽고 그리고/또는 그에 쓰기 위해 그 대응하는 API와 인터페이싱하기 위한 대응하는 논리를 포함할 수 있다. 이와 같이, 정규화기들은 외부 데이터 소스들과 데이터의 메모리 이미지 간을 브리징(bridge)한다.

[0046] 데이터 액세스 구성요소(710)는 입력 데이터(701)를 평가한다. 입력 데이터가 이미 정규형(canonical)이고 따라서 분석 부분(800)에 의해 처리 가능한 경우, 입력 데이터는 분석 부분(800)에 입력될 정규형 데이터(740)로서 직접 제공될 수 있다.

[0047] 그렇지만, 입력 데이터(701)가 정규형이 아닌 경우, 적절한 데이터 정규화 구성요소(730)는 입력 데이터(701)를 정규 형식으로 변환할 수 있다. 데이터 정규화 구성요소들(730)은 실제로는, 각각이 특정의 특성들을 가지는 입력 데이터를 정규 형태로 변환할 수 있는, 데이터 정규화 구성요소들(730)의 집합체이다. 정규화 구성요소들(730)의 집합체는 4 개의 정규화 구성요소들(731, 732, 733 및 734)을 포함하는 것으로 나타내어져 있다. 그렇지만, 생략부호(735)는 다른 개수들의 정규화 구성요소들도 있을 수 있다(어쩌면 예시된 4 개보다 훨씬 더 적을 수 있음)는 것을 나타낸다.

[0048] 입력 데이터(701)는 심지어 정규화기 자체는 물론, 상관된 데이터 특성(들)의 ID(identification)를 포함할 수 있다. 데이터 부분(700)은 이어서 상관된 데이터 특성들을 등록하고, 정규화 구성요소를 데이터 정규화 구성요소 집합체(730)에 제공하며, 여기서 그 정규화 구성요소가 이용 가능 정규화 구성요소들에 추가될 수 있다. 그 상관된 특성들을 가지는 입력 데이터가 나중에 수신되는 경우, 데이터 부분(700)은 이어서 입력 데이터를 상관된 정규화 구성요소에 할당할 수 있다. 정규화 구성요소들은 또한, 웹 상의 정의된 구성요소 라이브러리들과 같은, 외부 소스들로부터 동적으로 발견될 수 있다. 예를 들어, 주어진 데이터 소스에 대한 스키마가 알려져 있지만 필요한 정규화기가 존재하지 않는 경우, 외부 구성요소 라이브러리가 발견될 수 있고 필요한 구성요소들을

포함하기만 하다면, 정규화기가 이러한 라이브러리로부터 찾아질 수 있다. 파이프라인은 또한, 데이터의 유형의 동적 결정을 시도하고 따라서 필요한 정규화기 구성요소들을 찾아내기 위해, 스키마가 아직 알려져 있지 않은 데이터를 파싱(parsing)하여 파싱 결과들을 알려진 구성요소 라이브러리들에서의 스키마 정보와 비교할 수 있다.

[0049] 대안적으로, 입력 데이터가 모든 정규화 구성요소를 포함하지 않고, 입력 데이터가, 그 대신에, 정규화 변환들을 정의하는 변환 정의를 제공할 수 있다. 집합체(730)는 그러면 변환 정의를 0 개 이상의 표준 기본 정규화 변환과 함께 변환들을 시행하는 대응하는 정규화 구성요소로 변환하도록 구성될 수 있다. 이것은 데이터 부분(700)이 입력 데이터를 사용하고 파이프라인의 더 아래쪽에서 대응하는 정규화된 데이터를 제공하지 않는 경우의 일례를 나타낸다. 그렇지만, 어쩌면 대부분의 경우들에서, 입력 데이터(701)의 결과, 대응하는 정규화된 데이터(740)가 생성된다.

[0050] 하나의 실시예에서, 데이터 부분(700)은 입력 데이터의 파일 유형(file type) 및/또는 형식 유형(format type)에 기초하여 입력 데이터를 데이터 정규화 구성요소에 할당하도록 구성될 수 있다. 다른 특성들은, 예를 들어, 입력 데이터의 소스를 포함할 수 있다. 기본 정규화 구성요소는 지정된 대응하는 정규화 구성요소를 갖지 않는 입력 데이터에 할당될 수 있다. 기본 정규화 구성요소는 입력 데이터를 정규화하려고 시도하기 위해 한 세트의 규칙들을 적용할 수 있다. 기본 정규화 구성요소가 데이터를 정규화할 수 없는 경우, 기본 정규화 구성요소는 입력 데이터에 대한 스키마 정의를 제공하라고 사용자에게 프롬프트하기 위해 도 5의 작성 구성요소(540)를 트리거할 수 있다. 스키마 정의가 아직 존재하지 않는 경우, 작성 구성요소(540)는 작성자가 입력 데이터를 정규 형태로 변환하는 데 사용될 수 있는 대응하는 스키마 정의를 생성하는 데 도움을 주는 스키마 정의 어시스턴트(schema definition assistant)를 제공할 수 있다. 데이터가 정규 형태로 되어 있으면, 데이터에 수반하는 스키마는 나머지 파이프라인(601)이 데이터를 해석하기 위해 새로운 코드를 필요로 하지 않을 정도로 데이터에 대한 충분한 설명을 제공한다. 그 대신에, 파이프라인(601)은 액세스 가능한 스키마 선언 언어(schema declaration language)로 표현 가능한 임의의 스키마를 바탕으로 데이터를 해석할 수 있는 코드를 포함한다.

[0051] 그에 관계없이, 정규형 데이터(740)가 데이터 부분(700)으로부터의 출력 데이터로서 그리고 분석 부분(800)에의 입력 데이터로서 제공된다. 정규형 데이터는 각종의 데이터 유형들을 포함하는 필드들을 포함할 수 있다. 예를 들어, 필드들은 정수, 부동 소수점 수, 문자열, 벡터, 어레이, 컬렉션, 계층적 구조, 텍스트, XML 문서, 테이블, 목록, SQL 데이터베이스 쿼리 결과, BI(business intelligence) 큐브 쿼리 결과, 다양한 형식으로 된 2D 드로잉 및 3D 시각적 모델과 같은 그래픽 정보, 또는 심지어 이 다양한 데이터 유형들의 복잡한 조합들과 같은 간단한 데이터 유형들을 포함할 수 있다. 다른 장점으로서, 정규화 프로세스는 아주 다양한 입력 데이터를 정규화할 수 있다. 게다가, 데이터 부분(700)이 받아들일 수 있는 각종의 입력 데이터가 확장 가능하다. 이것은, 본 설명에서 나중에 논의될 것인 바와 같이, 다수의 모델들이 결합되는 경우에 도움이 된다.

[0052] 도 8은 도 6의 파이프라인(601)의 분석 부분(620)의 일례를 나타내는 분석 부분(800)을 나타낸 것이다. 데이터 부분(700)은 정규화된 데이터(801)를 데이터 모델 바인딩 구성요소(data-model binding component)(810)에 제공하였다. 정규화된 데이터(801)가 임의의 정규화된 형태 및 임의의 수의 파라미터들을 가질 수 있지만, 여기서 파라미터들의 형태 및 개수가 심지어 입력 데이터마다 상이할 수 있다. 그렇지만, 논의의 목적상, 정규형 데이터(801)는 필드들(802A 내지 802H) - 본 명세서에서 총칭하여 "필드들(802)"이라고 지칭될 수 있음 - 을 가진다.

[0053] 다른 한편으로, 분석 부분(800)은 다수의 모델 파라미터들(811)을 포함한다. 모델 파라미터들의 유형 및 개수가 모델에 따라 상이할 수 있다. 그렇지만, 특정의 예의 논의의 목적상, 모델 파라미터들(811)은 모델 파라미터들(811A, 811B, 811C 및 811D)을 포함하는 것으로 논의될 것이다. 하나의 실시예에서, 모델 파라미터들의 ID, 및 모델 파라미터들 간의 분석적 관계가 명령형 코딩을 사용함이 없이 선언적으로 정의될 수 있다.

[0054] 데이터 모델 바인딩 구성요소(810)는 정규화된 데이터 필드들(802)과 모델 파라미터들(811) 사이에서 중재함으로써, 필드들 사이의 바인딩들을 제공한다. 이 경우에, 데이터 필드(802B)는, 화살표(803A)로 나타낸 바와 같이, 모델 파라미터(811A)에 바인딩된다. 환언하면, 데이터 필드(802B)로부터의 값은 모델 파라미터(811A)를 채우는 데 사용된다. 또한, 이 예에서, 데이터 필드(802E)는 [화살표(803B)로 나타낸 바와 같이], 모델 파라미터(811B)에 바인딩되고, 데이터 필드(802H)는 [화살표(803C)로 나타낸 바와 같이] 모델 파라미터(811C)에 바인딩된다.

[0055] 데이터 필드들(802A, 802C, 802D, 802F 및 802G)은 모델 파라미터들 중 어느 것에도 바인딩어 있지 않은 것으로 도시되어 있다. 이것은 입력 데이터로부터의 데이터 필드들 모두가 모델 파라미터들로서 항상 사용될 필요는 없

다는 것을 강조하기 위한 것이다. 하나의 실시예에서, 이 데이터 필드들 중 하나 이상은 정규화된 데이터로부터의(이 정규화된 데이터 또는 어쩌면 임의의 장래의 유사한 정규화된 데이터에 대한) 필드들이 어느 모델 파라미터에 바인딩되어야 하는지에 관한 명령어들을 데이터 모델 바인딩 구성요소(810)에 제공하는 데 사용될 수 있다. 이것은 도 6의 분석 부분(620)에 제공될 수 있는 분석 데이터(621)의 종류의 일례를 나타낸다. 정규화된 데이터로부터의 어느 데이터 필드들이 어느 모델 파라미터들에 바인딩되는지의 정의는 다수의 방식들로 작성될 수 있다. 예를 들어, 바인딩들은, 1) 작성 시에 작성자에 의해 명시적으로 설정될 수 있고, 2) (작성자에 의해 부과된 임의의 제한들에 따라) 사용 시에 사용자에 의해 명시적으로 설정될 수 있으며, 3) 알고리즘적 휴리스틱(algorithmic heuristics)에 기초하여 작성 구성요소(640)에 의한 자동 바인딩(automatic binding)일 수 있고, 그리고/또는 4) 바인딩이 알고리즘적으로 행해질 수 없는 것으로 결정될 때, 작성 구성요소가 바인딩을 명시하라고 작성자 및/또는 사용자에게 프롬프트하는 것일 수 있다. 이와 같이, 바인딩들이 또한 모델 논리 자체의 일부로서 분석될 수 있다.

[0056] 작성자가 어느 데이터 필드들이 어느 모델 파라미터들에 매핑되는지를 정의할 수 있는 것은 모델 파라미터들을 정의하기 위해 작성자의 마음에 드는 심볼들을 사용할 수 있다는 점에서 많은 유연성을 작성자에게 제공한다. 예를 들어, 모델 파라미터들 중 하나가 압력을 의미하는 경우, 작성자는 그 모델 파라미터를 "압력" 또는 "P" 또는 작성자에게 의미가 통하는 임의의 다른 심볼로 명명할 수 있다. 작성자는 심지어, 하나의 실시예에서, 이전에 이전 이름의 모델 파라미터에 대한 것인 바인딩들이 그 대신에 새로운 이름의 모델 파라미터에 바인딩될 수 있게 하기 위해, 데이터 모델 바인딩 구성요소(810)로 하여금 자동으로 업데이트하게 할 수 있는 모델 파라미터를 재명명함으로써, 원하는 바인딩들을 유지할 수 있다. 이 바인딩 메커니즘은 또한 바인딩이 런타임 시에 선언적으로 변경될 수 있게 한다.

[0057] 모델 파라미터(811D)는, 이 예에서, 모델 파라미터(811D)가 데이터 모델 바인딩 구성요소(810)에 의해 값을 할당받지 않았다는 것을 강조하기 위해 별표로 예시되어 있다. 그에 따라, 모델 파라미터(811D)는 미지수로 남아 있다. 환언하면, 모델 파라미터(811D)에 값이 할당되어 있지 않다.

[0058] 모델링 구성요소(820)는 다수의 기능들을 수행한다. 첫째, 모델링 구성요소(820)는 모델 파라미터들(811) 간의 분석적 관계들(821)을 정의한다. 분석적 관계들(821)은 방정식들(831), 규칙들(832) 및 제약조건들(833)을 비롯한 3 개의 일반 카테고리들로 분류된다. 그렇지만, 솔버들의 목록이 확장 가능하다. 하나의 실시예에서, 예를 들어, 대응하는 시뮬레이션 엔진이 제공되어 솔버로서 등록되어 있기만 하다면, 하나 이상의 시뮬레이션들이 분석적 관계들의 일부로서 포함될 수 있다.

[0059] 본 명세서에서 사용되는 "방정식"이라는 용어는 수학 분야에서 사용되는 용어를 따른다.

[0060] 본 명세서에서 사용되는 "규칙"이라는 용어는 조건문(conditional statement)을 의미하고, 여기서 하나 이상의 조건들이 충족되는 경우[조건(conditional) 또는 조건문의 "if" 부분], 하나 이상의 동작들이 취해져야 한다[결과(consequence) 또는 조건문의 "then" 부분]. 하나 이상의 모델 파라미터들이 조건문에 표현되어 있거나 하나 이상의 모델 파라미터들이 결과문(consequence statement)에 표현되어 있는 경우, 규칙이 모델 파라미터들에 적용된다.

[0061] 본 명세서에서 사용되는 "제약조건(constraint)"이라는 용어는 하나 이상의 모델 파라미터들에 제한이 적용된다는 것을 의미한다. 예를 들어, 도시 계획 모델에서, 특정의 주택 요소가 총 가능한 구역 지정(zoning designation)들의 부분 집합을 가지는 지도 위치 상의 배치로 제한될 수 있다. 다리 요소가 특정 최대 길이 또는 특정 차선 수 미만으로 제한될 수 있다.

[0062] 모델에 익숙한 작성자는 그 모델에 적용되는 이 방정식들, 규칙들 및 제약조건들의 표현식들을 제공할 수 있다. 시뮬레이션들의 경우에, 작성자는 모델 파라미터들 간의 적절한 시뮬레이션 관계들을 제공하는 적절한 시뮬레이션 엔진을 제공할 수 있다. 모델링 구성요소(820)는 작성자가 방정식들, 규칙들 및 제약조건들에 대한 자연스러운 심볼 표현들을 제공하기 위한 메커니즘을 제공할 수 있다. 예를 들어, 열역학 관련 모델의 작성자는 간단히 열역학 교재로부터 방정식들을 복사하여 붙여넣기할 수 있다. 모델 파라미터들을 데이터 필드들에 바인딩할 수 있는 것은 작성자가 어느 것이든 작성자에게 익숙한 심볼들(작성자가 신뢰하는 교재에서 사용되는 정확한 심볼들 등) 또는 작성자가 사용하고자 하는 정확한 심볼들을 사용할 수 있게 한다.

[0063] 해를 구하기 전에, 모델링 구성요소(820)는 또한 모델 파라미터들 중 어느 것이 해가 구해져야 하는지[즉, 이후부터, 단수인 경우 "출력 모델 변수", 또는 복수인 경우 "출력 모델 변수들", 또는 단일의 또는 복수의 출력 모델 변수들이 있을 수 있는 경우 "출력 모델 변수(들)"]를 식별한다. 출력 모델 변수들은 미지의 파라미터들일

수 있거나, 알려진 모델 파라미터들일 수 있으며, 여기서 알려진 모델 파라미터의 값은 해 구하기 동작(solve operation)에서 변화를 겪는다. 도 8의 예에서, 데이터 모델 바인딩 동작 후에, 모델 파라미터들(811A, 811B 및 811C)은 알려진 것이고, 모델 파라미터(811D)는 미지의 것이다. 그에 따라, 미지의 모델 파라미터(811D)는 출력 모델 변수들 중 하나일 수 있다. 다른 대안으로서 또는 그에 부가하여, 알려진 모델 파라미터들(811A, 811B 및 811C) 중 하나 이상도 출력 모델 변수들일 수 있다. 솔버(solver)(840)는 이어서, 가능한 경우, 출력 모델 변수(들)에 대한 해를 구한다. 이하에서 기술되는 하나의 실시예에서, 솔버(840)는, 해 구하기 동작이 수행될 수 있게 하기 위해 충분한 입력 모델 변수들이 제공되는 한, 단일의 모델 내에서도, 각종의 출력 모델 변수들에 대한 해를 구할 수 있다. 입력 모델 변수들은, 예를 들어, 해 구하기 동작 동안 변화를 겪지 않는 값들을 가지는 알려진 모델 파라미터들일 수 있다. 예를 들어, 도 8에서, 모델 파라미터들(811A 및 811D)이 입력 모델 변수들인 경우, 솔버는, 그 대신에, 출력 모델 변수들(811B 및 811C)에 대한 해를 구할 수 있다. 하나의 실시예에서, 솔버는 단일의 모델 파라미터에 대해 다수의 상이한 데이터 유형들 중 임의의 하나를 출력할 수 있다. 예를 들어, 피연산자들이 정수, 부동 소수점, 그의 벡터들, 또는 그의 행렬들인지에 상관없이, 어떤 방정식 연산들(덧셈, 뺄셈, 기타 등등)이 적용된다.

[0064] 하나의 실시예에서, 솔버(840)가 특정의 출력 모델 변수들에 대한 해를 구할 수 없을 때에도, 솔버(800)는, 실제의 숫자 결과(또는 어느 것이든 해가 구해진 데이터 유형)에 대한 전체 해 구하기(full solve)가 가능하지 않을지라도, 여전히 그 출력 모델 변수에 대한 부분 해(partial solution)를 제공할 수 있다. 이것은 파이프라인이 전체 해 구하기에 도달하기 위해 어떤 정보가 필요한지에 관해 작성자에게 프롬프트하는 것에 의해 증분적 전개(incremental development)를 용이하게 할 수 있다. 이것은 또한 작성 시(author time)와 사용 시(use time) 간의 차이를 없애는 데 도움을 주는데, 왜냐하면 다양한 작성 스테이지들 전체에 걸쳐 적어도 부분 해 구하기(partial solve)가 이용 가능하기 때문이다. 추상적 예에 대해, 분석 모델이 방정식 a=b+c+d를 포함하는 것으로 가정한다. 이제 a, c 및 d가 출력 모델 변수들이고, b가 5(이 경우에, 정수)의 알려진 값을 가지는 입력 모델 변수인 것으로 가정한다. 해 구하기 프로세스에서, 솔버(840)는 출력 모델 변수들 중 하나 "d"에 대해서만 해를 구하고, 6(정수)의 값을 "d"로 지칭되는 모델 파라미터에 할당할 수 있지만, 솔버(840)는 "c"에 대한 해를 구할 수 없다. "a"가 "c"에 종속되어 있기 때문에, "a"로 지칭되는 모델 파라미터도 미지의 것으로 남아 있고 해가 구해지지 않는다. 이 경우에, 정수 값을 "a"에 할당하는 대신에, 솔버는 부분 해 구하기를 행하고 "c+11"의 문자열 값을 모델 파라미터 "a"에 출력할 수 있다. 이전에 언급된 바와 같이, 이것은 도메인 전문가가 분석 모델을 작성하고 있을 때 특히 도움이 될 수 있고, 기본적으로 모델 파라미터 "a"의 내용에 관한 부분 정보를 제공하는 역할을 할 것이며, 또한 "c" 모델 파라미터에 대한 해가 구해질 수 있게 하는 어떤 추가의 모델 분석이 제공될 필요가 있다는 것을 작성자에게 신호(cue)하는 역할을 할 것이다. 이 부분 해 구하기 결과는 도메인 전문가가 부분 결과를 볼 수 있게 하기 위해 아마도 어떤 방식으로 뷰 합성에 출력될 수 있다.

[0065] 솔버(840)가 도 8에 간략화된 형태로 도시되어 있다. 그렇지만, 도 9와 관련하여 기술될 것인 바와 같이, 솔버(840)는 다수의 구성 솔버(constituent solver)들의 동작을 지시할 수 있다. 도 8에서, 모델링 구성요소(820)는 이어서 모델 파라미터들(현재 알려져 있고 해가 구해진 출력 모델 변수들을 포함함)을 도 9의 뷰 부분(900)에 제공될 출력으로서 이용 가능하게 한다.

[0066] 도 9는 도 6의 뷰 부분(630)의 일례를 나타내고 재계산 사용자 인터페이스(200)에 시각화된 컨트롤들의 예를 나타내는 뷰 부분(900)을 예시한 것이다. 뷰 부분(900)은 도 8의 분석 부분(800)으로부터 모델 파라미터들(811)을 수신한다. 뷰 부분은 또한 뷰 구성요소들의 집합체를 포함하는 뷰 구성요소 리포지토리(view components repository)(920)를 포함한다. 예를 들어, 뷰 구성요소 리포지토리(920)는, 이 예에서, 뷰 구성요소들(921 내지 924)을 포함하는 것으로 예시되어 있지만, 뷰 구성요소 리포지토리(920)는 임의의 수의 뷰 구성요소들을 포함할 수 있다. 뷰 구성요소들 각각은 0 개 이상의 입력 파라미터들을 포함할 수 있다. 예를 들어, 뷰 구성요소(921)는 어떤 입력 파라미터들도 포함하지 않는다. 그렇지만, 뷰 구성요소(922)는 2 개의 입력 파라미터들(942A 및 942B)을 포함한다. 뷰 구성요소(923)는 하나의 입력 파라미터(943)를 포함하고, 뷰 구성요소(924)는 하나의 입력 파라미터(944)를 포함한다. 그렇다 해도, 이것은 일례에 불과하다. 입력 파라미터들은 시각적 항목이 어떻게 렌더링되는지에 영향을 미칠 수 있지만, 꼭 그럴 필요는 없다. 뷰 구성요소(921)가 어떤 입력 파라미터들도 포함하지 않는다는 사실은 어떤 모델 파라미터들도 참조함이 없이 생성되는 뷰들이 있을 수 있다는 것을 강조한다. 변하지 않는 고정된(내장된) 데이터만을 포함하는 뷰를 생각해보자. 이러한 뷰는, 예를 들어, 사용자를 위한 참조 정보를 구성한다. 대안적으로, 항목들이 모델 내로의 가져오기(import)를 위해 뷰로부터 선택될 수 있도록, 카탈로그를 브라우징하는 방식을 제공하기만 하는 뷰를 생각해보자.

[0067] 각각의 뷰 구성요소(921 내지 924)는, 대응하는 뷰 구성요소 입력 파라미터(들)(있는 경우)를 사용하여 뷰 합성

구성요소(940)에 의해 실행될 때, 대응하는 뷰 항목(view item)을 가상 공간(950)에 위치시키는 대응하는 논리를 포함하거나 그와 연관되어 있다. 그 가상 항목은 정적 이미지 또는 객체일 수 있거나, 동적 애니메이션화된 가상 항목 또는 객체일 수 있다. 예를 들어, 뷰 구성요소들(921 내지 924) 각각은, 실행될 때, 각각, 대응하는 가상 항목(951 내지 954)을 가상 공간(950)에 렌더링시키는 대응하는 논리(931 내지 934)와 연관되어 있다. 가상 항목들이 간단한 형상들로서 예시되어 있다. 그렇지만, 가상 항목들이 형태가 꽤 복잡할 수 있다(어쩌면 심지어 애니메이션을 포함함). 이 설명에서, 뷰 항목이 가상 공간에 렌더링될 때, 그것은 뷰 합성 구성요소가, 렌더링 엔진에 제공될 때, 렌더링 엔진이 뷰 항목을 지정된 위치에 그리고 지정된 방식으로 디스플레이 상에 표시할 수 있도록, 충분한 명령어들을 작성했다는 것을 의미한다.

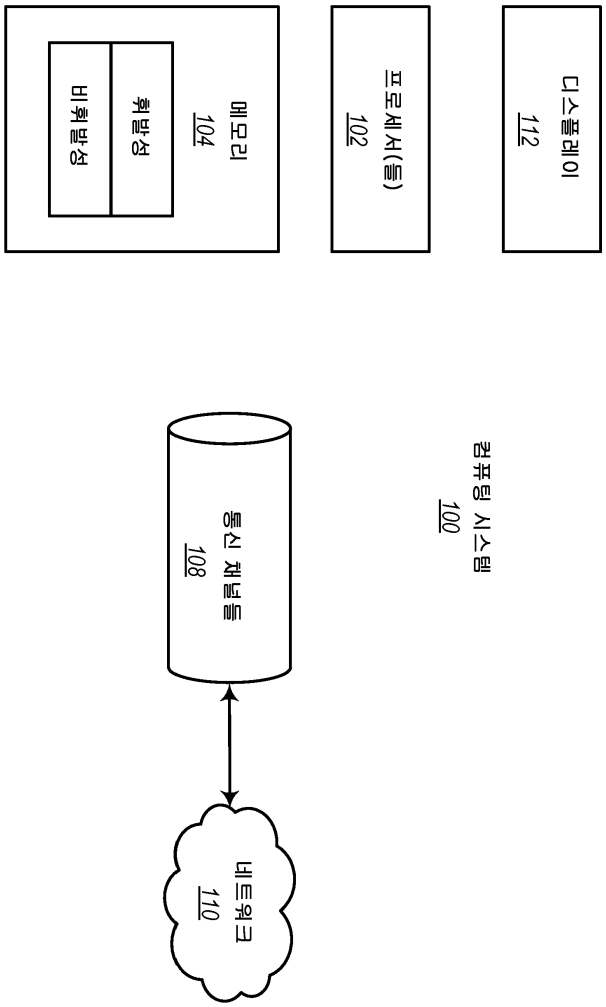
[0068] 뷰 구성요소들(921 내지 924)이, 예를 들어, 도 6의 작성 구성요소(840)를 사용하여, 어쩌면 심지어 뷰 데이터로서 뷰 부분(900)에 제공될 수 있다. 예를 들어, 작성 구성요소(640)는 작성자가 몇 개의 기하학적 형태들로부터 선택할 수 있게 하거나 어쩌면 다른 기하학적 형태들을 합성할 수 있게 하는 선택기를 제공할 수 있다. 작성자는 또한 각각의 뷰 구성요소에 대한 입력 파라미터들의 유형들을 지정할 수 있는 반면, 입력 파라미터들 중 일부는 뷰 부분(900)에 의해 부과되는 기본 입력 파라미터들일 수 있다. 각각의 뷰 구성요소(921 내지 924)와 연관되어 있는 논리는 또한 뷰 데이터를 제공받을 수 있고, 그리고/또는 또한 뷰 부분(900) 자체에 의해 제공되는 어떤 기본 기능을 포함할 수 있다.

[0069] 뷰 부분(900)은 모델 파라미터들 중 적어도 일부를 뷰 구성요소들(921 내지 924)의 대응하는 입력 파라미터들에 바인딩하도록 구성되어 있는 모델-뷰 바인딩 구성요소(model-view binding component)(910)를 포함한다. 예를 들어, 모델 파라미터(811A)는, 화살표(911A)로 나타낸 바와 같이, 뷰 구성요소(922)의 입력 파라미터(942A)에 바인딩된다. 모델 파라미터(811B)는, 화살표(911B)로 나타낸 바와 같이, 뷰 구성요소(922)의 입력 파라미터(942B)에 바인딩된다. 또한, 모델 파라미터(811D)는, 화살표(911C)로 나타낸 바와 같이, 뷰 구성요소(923)의 입력 파라미터(943) 및 뷰 구성요소(924)의 입력 파라미터(944)에 바인딩된다. 모델 파라미터(811C)는 어떤 대응하는 뷰 구성요소 파라미터에도 바인딩되어 있지 않은 것으로 도시되어 있으며, 이는, 모델 파라미터들이 분석 부분에서 필수적인 것일지라도, 그 모델 파라미터들 모두가 파라미터의 뷰 부분에 의해 사용될 필요는 없다는 것을 강조한다. 또한, 모델 파라미터(811D)는 뷰 구성요소들의 2 개의 상이한 입력 파라미터들에 바인딩되어 있는 것으로 도시되어 있고, 이는 모델 파라미터들이 다수의 뷰 구성요소 파라미터들에 바인딩될 수 있다는 것을 나타낸다. 하나의 실시예에서, 모델 파라미터들과 뷰 구성요소 파라미터들 사이의 바인딩들의 정의는, 1) 작성 시에 작성자에 의해 명시적으로 설정되는 것, 2) (작성자에 의해 부과된 임의의 제한들에 따라) 사용 시에 사용자에 의해 명시적으로 설정되는 것, 3) 알고리즘적 휴리스틱에 기초하여 작성 구성요소(640)에 의한 자동 바인딩, 및/또는 4) 바인딩이 알고리즘적으로 행해질 수 없는 것으로 결정될 때, 작성 구성요소가 바인딩을 명시하라고 작성자 및/또는 사용자에게 프롬프트하는 것에 의해 작성될 수 있다.

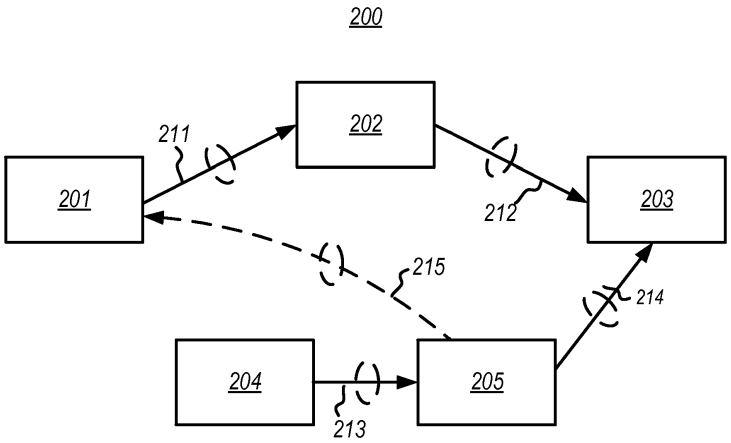
[0070] 본 발명은 그의 사상 또는 본질적인 특성들을 벗어남이 없이 다른 구체적인 형태들로 구현될 수 있다. 기술된 실시예들이 모든 점에서 제한적이 아니라 단지 예시적인 것으로 간주되어야 한다. 따라서, 본 발명의 범주는 이상의 설명이 아니라 첨부된 청구범위에 의해 나타내어진다. 청구범위의 등가성의 의미 및 범위 내에 속하는 모든 변경들이 청구범위의 범주 내에 포함된다.

도면

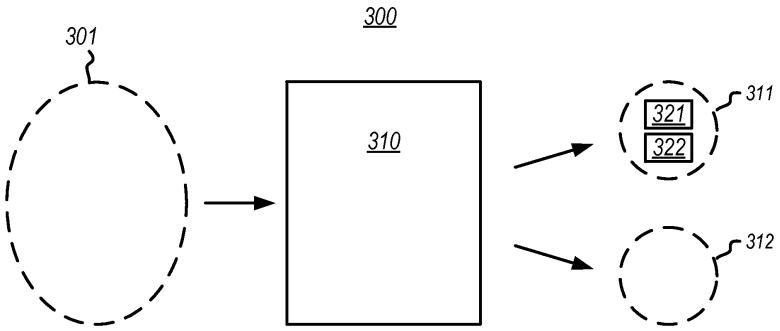
도면1


메모리
104
휘발성
비휘발성
프로세서(들)
102
디스플레이
112
컴퓨팅 시스템
100
통신 채널들
108
네트워크
110

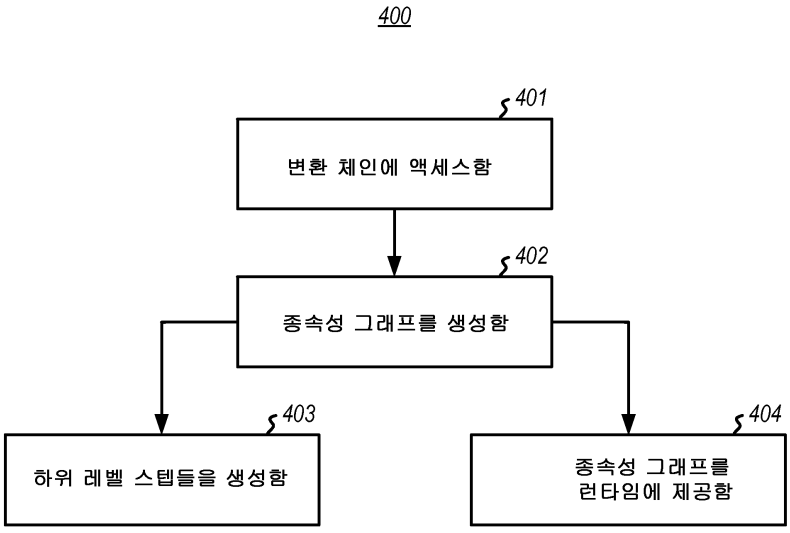
도면2


200
201
202
203
204
205
211
212
213
214
215

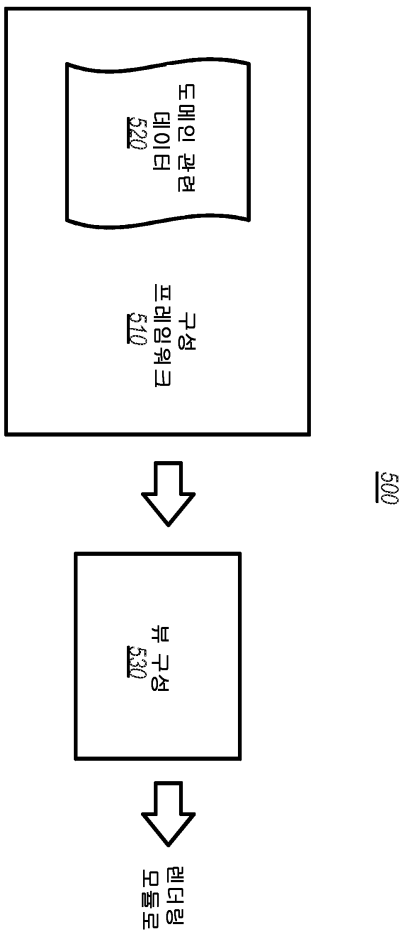
도면3


300
301
310
311
321
322
312

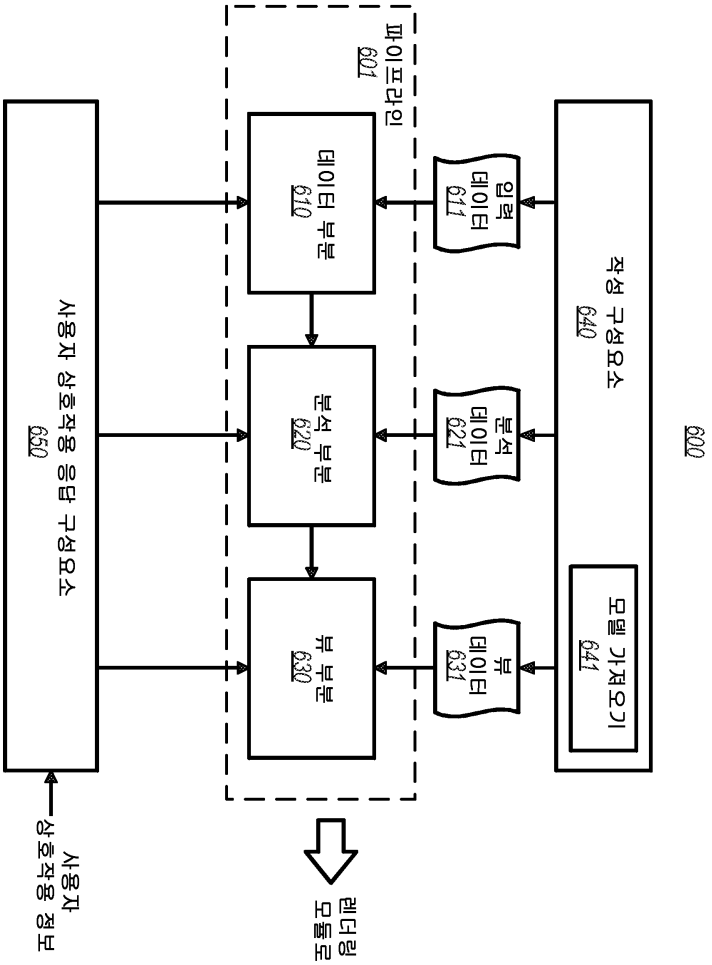
도면4


400
401
변환 체인에 액세스함
402
종속성 그래프를 생성함
403
하위 레벨 스텝들을 생성함
404
종속성 그래프를
런타임에 제공함

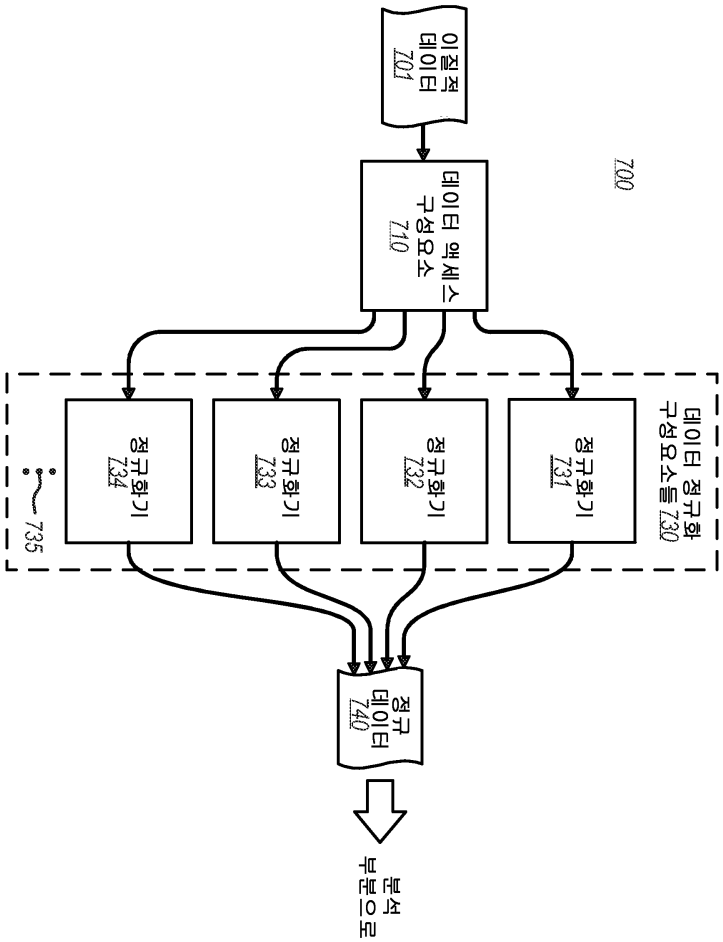
도면5



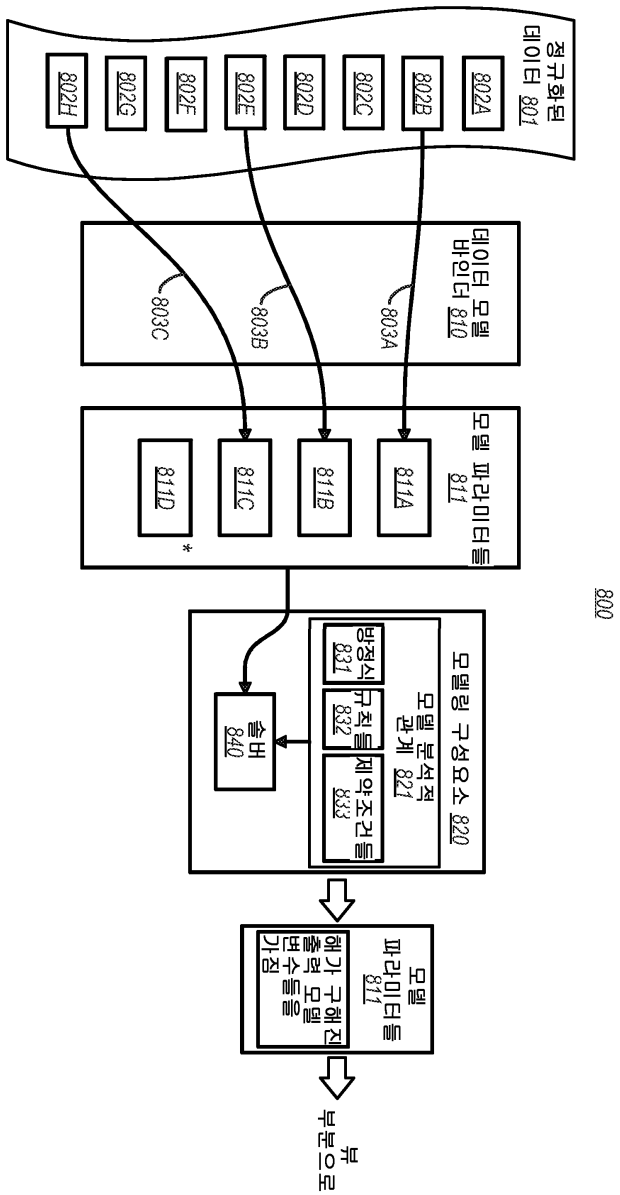
도면6


파이프라인
601
데이터 부분
610
분석 부분
620
뷰 부분
630
입력 데이터
611
분석 데이터
621
뷰 데이터
631
작성 구성요소
640
모델 가져오기
641
600
사용자 상호작용 응답 구성요소
650
사용자 상호작용 정보
렌더링 모듈로

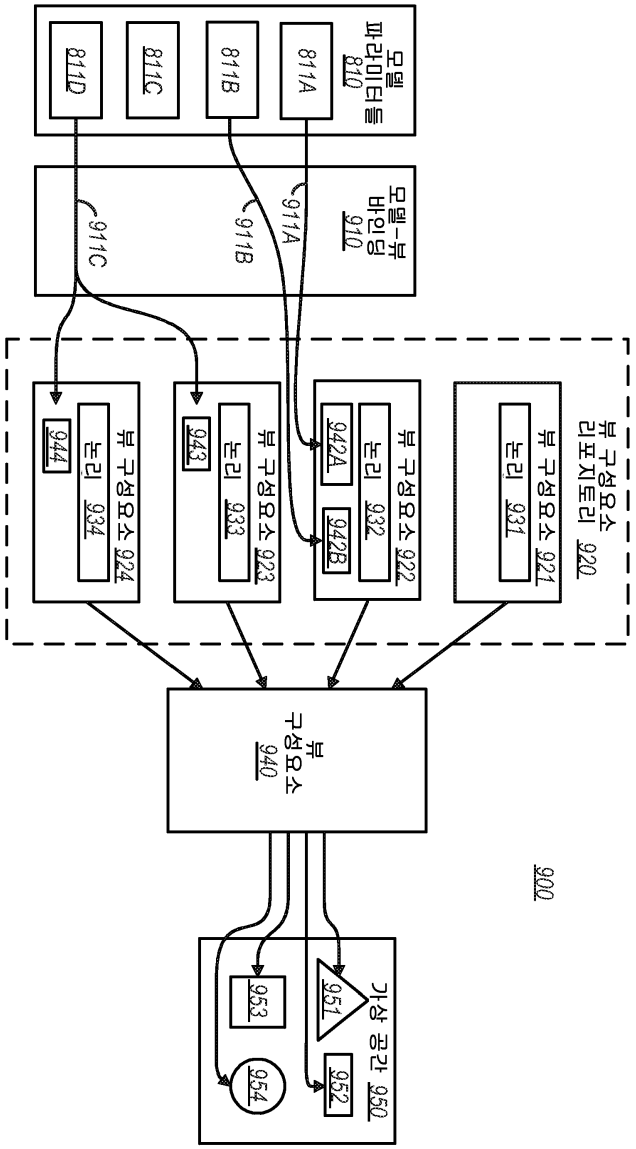
도면7


이질적 데이터 701
700
데이터 엑세스 구성요소 710
데이터 정규화 구성요소들 730
정규화기 731
정규화기 732
정규화기 733
정규화기 734
735
정규 데이터 740
분석 부분으로

도면8


정규화된 데이터 801
802A
802B
802C
802D
802E
802F
802G
802H
데이터 모델 바인더 810
803A
803B
803C
모델 파라미터들 811
811A
811B
811C
811D
*
모델링 구성요소 820
모델 분석적 관계 821
방정식 831
규칙들 832
제약조건들 833
솔버 840
모델 파라미터들 811
해가 구해진 출력 모델 변수들을 가짐
뷰 분석으로
800

도면9


모델 파라미터들 810
811A
811B
811C
811D
모델-뷰 바인딩 910
911A
911B
911C
뷰 구성요소 리포지토리 920
뷰 구성요소 921
논리 931
뷰 구성요소 922
논리 932
942A
942B
뷰 구성요소 923
논리 933
943
뷰 구성요소 924
논리 934
944
뷰 구성요소 940
가상 공간 950
951
952
953
954
900