

(19) 대한민국특허청(KR)
(12) 공개특허공보(A)

(11) 공개번호 10-2022-0144413

(43) 공개일자 2022년10월26일

(51) 국제특허분류(Int. Cl.)

G06F 11/30 (2006.01) G06F 11/34 (2006.01)

G06F 11/36 (2006.01) G06N 20/00 (2019.01)

(52) CPC특허분류

G06F 11/302 (2013.01)

G06F 11/3051 (2013.01)

(21) 출원번호 10-2022-7034333

(22) 출원일자(국제) 2021년03월24일

심사청구일자 2022년09월30일

(85) 번역문제출일자 2022년09월30일

(86) 국제출원번호 PCT/IB2021/052424

(87) 국제공개번호 WO 2021/214569

국제공개일자 2021년10월28일

(30) 우선권주장

16/855,565 2020년04월22일 미국(US)

(71) 출원인

인터내셔널 비지네스 머신즈 코포레이션

미국 10504 뉴욕주 아몬크 뉴오차드 로드

(72) 발명자

샤오, 진

미국 뉴욕 10598-0218, 요크타운 하이츠, 피오 박스 218, 키차완 로드 1101, 아이비엠 코포레이션

카를라, 아누프

미국 뉴욕 10598-0218, 요크타운 하이츠, 피오 박스 218, 키차완 로드 1101, 아이비엠 코포레이션

(뒷면에 계속)

(74) 대리인

허정훈

전체 청구항 수 : 총 17 항

(54) 발명의 명칭 런타임 추적들에 기초한 모놀리식 애플리케이션에서의 마이크로서비스들의 생성

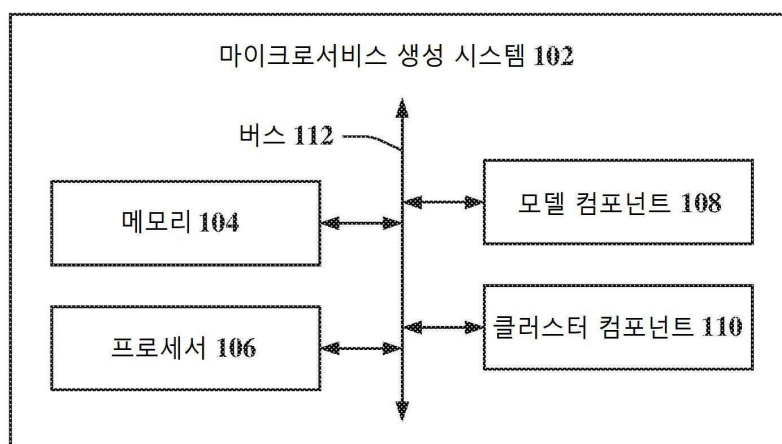
(57) 요약

런타임 추적들에 기초한 모놀리식 애플리케이션에서의 마이크로서비스들의 생성 (GENERATION OF MICROSERVICES FROM A MONOLITHIC APPLICATION BASED ON RUNTIME TRACES)

런타임 추적들에 기초하여 모놀리식 애플리케이션에서 마이크로서비스들의 생성을 용이하게 하는 시스템들, 컴퓨터 (뒷면에 계속)

대표도 - 도1

100



터-구현 방법들, 컴퓨터 프로그램 제품들이 제공된다. 일 실시예에 따르면, 시스템은 컴퓨터 실행가능 컴포넌트들을 저장하는 메모리 및 상기 메모리에 저장된 컴퓨터 실행가능 컴포넌트들을 실행하는 프로세서를 포함할 수 있다. 상기 컴퓨터 실행 가능 컴포넌트들은 실행된 테스트 케이스들의 런타임 추적들에 기초하여 모놀리식 애플리케이션에서 클래스들의 클러스터 할당들을 학습하는 모델 컴포넌트를 포함할 수 있다. 상기 컴퓨터 실행가능 컴포넌트들은 모놀리식 애플리케이션의 하나 혹은 그 이상의 마이크로서비스들을 식별하기 위해 클러스터 할당들에 기초하여 클래스들의 클러스터들을 생성하도록 모델 컴포넌트를 채용하는 클러스터 컴포넌트를 더 포함할 수 있다.

(52) CPC특허분류

G06F 11/3409 (2013.01)

G06F 11/3447 (2013.01)

G06F 11/3466 (2013.01)

G06F 11/3612 (2013.01)

G06F 11/3636 (2013.01)

G06F 11/3684 (2013.01)

G06F 11/3688 (2013.01)

G06F 11/3692 (2013.01)

G06N 20/00 (2021.08)

(72) 발명자

런, 친

미국 뉴욕 10598-0218, 요크타운 하이츠, 피오 박스 218, 키차완 로드 1101, 아이비엠 코포레이션

바타, 라그하브

미국 뉴욕 10598-0218, 요크타운 하이츠, 피오 박스 218, 키차완 로드 1101, 아이비엠 코포레이션

신하, 사우라브

미국 뉴욕 10598-0218, 요크타운 하이츠, 피오 박스 218, 키차완 로드 1101, 아이비엠 코포레이션

로프라노, 존

미국 뉴욕 10598-0218, 요크타운 하이츠, 피오 박스 218, 키차완 로드 1101, 아이비엠 코포레이션

부코비치, 마자

미국 뉴욕 10598-0218, 요크타운 하이츠, 피오 박스 218, 키차완 로드 1101, 아이비엠 코포레이션

명세서

청구범위

청구항 1

시스템에 있어서, 상기 시스템은:

컴퓨터 실행 가능한 컴포넌트들을 저장하는 메모리; 및

상기 메모리에 저장된 상기 컴퓨터 실행 가능 컴포넌트들을 실행하는 프로세서를 포함하고, 상기 컴퓨터 실행 가능 컴포넌트들은:

실행된 테스트 케이스들의 런타임 추적들(runtime traces of executed test cases)에 기초하여 모놀리식 애플리케이션에서 클래스들의 클러스터 할당들(cluster assignments of classes)을 학습하는 모델 컴포넌트(a model component); 및

상기 모놀리식 애플리케이션의 하나 혹은 그 이상의 마이크로서비스들을 식별하기 위해 상기 클러스터 할당들에 기초하여 상기 클래스들의 클러스터들을 생성하도록 상기 모델 컴포넌트를 채용하는 클러스터 컴포넌트(a cluster component)를 포함하는

시스템.

청구항 2

제1항에 있어서, 상기 컴퓨터 실행 가능 컴포넌트들은:

상기 모놀리식 애플리케이션의 데이터 의존성(data dependency) 또는 상기 모놀리식 애플리케이션의 정적 호출 그래프(a static call graph) 중 적어도 하나에 기초하여 상기 하나 혹은 그 이상의 마이크로서비스들을 개량하는 개량 컴포넌트(a refinement component)를 포함하는

시스템.

청구항 3

제1항에 있어서, 상기 컴퓨터 실행 가능 컴포넌트들은:

상기 모놀리식 애플리케이션 또는 상기 모놀리식 애플리케이션에서 클래스들의 클러스터 할당들(the cluster assignments of the classes) 중 적어도 하나의 1차 시간 의존성들 또는 고차 시간 의존성들(first order temporal dependencies or high order temporal dependencies) 중 적어도 하나를 캡처하기 위해 실행된 테스트 케이스들의 런타임 추적들(the runtime traces of the executed test cases)에 기초하여 하나 혹은 그 이상의 인과 그래프들(causal graphs)을 생성하는 제2 모델 컴포넌트를 포함하는

시스템.

청구항 4

제1항에 있어서, 상기 모델 컴포넌트는 실행된 테스트 케이스들의 런타임 추적들에 기초하여 생성된 하나 혹은 그 이상의 인과 그래프들의 인과 시퀀스들을 사용하여 상기 모놀리식 애플리케이션에서 클래스들의 클러스터 할당들 또는 그래프 임베딩들 중 적어도 하나를 학습하는

시스템.

청구항 5

제1항에 있어서, 상기 실행된 테스트 케이스들은 런타임 추적들에서 상기 모놀리식 애플리케이션의 비즈니스 기능들을 제공하는 비즈니스 기능 테스트 케이스들을 포함하고, 이에 의해 상기 모놀리식 애플리케이션의 애플리케이션 현대화를 실행함에 있어서 프로세서 또는 시스템 중 적어도 하나의 개선된 성능, 개선된 효율성, 또는 계산 비용들의 감소를 촉진하는

시스템.

청구항 6

컴퓨터-구현되는 방법, 상기 방법은:

실행된 테스트 케이스들의 런타임 추적들(runtime traces of executed test cases)에 기초하여 모놀리식 애플리케이션에서 클래스들의 클러스터 할당들(cluster assignments of classes)을 학습하도록 모델을, 프로세서에 작동 가능하게 결합된 시스템에 의해, 훈련하는 단계; 및

상기 모놀리식 애플리케이션의 하나 혹은 그 이상의 마이크로서비스들을 식별하기 위해 상기 클러스터 할당들에 기초하여 상기 클래스들의 클러스터들을 생성하도록 상기 모델을, 상기 시스템에 의해, 채용하는 단계를 포함하는

컴퓨터-구현 방법.

청구항 7

제6항에 있어서, 상기 방법은:

상기 모놀리식 애플리케이션의 데이터 의존성(data dependency) 또는 상기 모놀리식 애플리케이션의 정적 호출 그래프(a static call graph) 중 적어도 하나에 기초하여 상기 하나 혹은 그 이상의 마이크로서비스들을, 상기 시스템에 의해, 개량하는 단계를 더 포함하는

컴퓨터-구현 방법.

청구항 8

제6항에 있어서, 상기 방법은:

상기 모놀리식 애플리케이션 또는 상기 모놀리식 애플리케이션에서 클래스들의 클러스터 할당들(the cluster assignments of the classes) 중 적어도 하나의 1차 시간 의존성들 또는 고차 시간 의존성들(first order temporal dependencies or high order temporal dependencies) 중 적어도 하나를 캡처하기 위해 실행된 테스트 케이스들의 런타임 추적들(the runtime traces of the executed test cases)에 기초하여 하나 혹은 그 이상의 인과 그래프들(causal graphs)을, 상기 시스템에 의해, 생성하는 단계를 더 포함하는

컴퓨터-구현 방법.

청구항 9

제6항에 있어서, 상기 훈련하는 단계는:

상기 실행된 테스트 케이스들의 런타임 추적들에 기초하여 생성된 하나 혹은 그 이상의 인과 그래프들의 인과 시퀀스들을 사용하여 상기 모놀리식 애플리케이션에서 클래스들의 클러스터 할당들 또는 그래프 임베딩들 중 적어도 하나를 학습하도록 모델을, 상기 시스템에 의해, 훈련하는 단계를 포함하는

컴퓨터-구현 방법.

청구항 10

제6항에 있어서, 상기 실행된 테스트 케이스들은 런타임 추적들에서 상기 모놀리식 애플리케이션의 비즈니스 기능들을 제공하는 비즈니스 기능 테스트 케이스들을 포함하고, 이에 의해 상기 모놀리식 애플리케이션의 애플리케이션 현대화를 실행함에 있어서 프로세서 또는 시스템 중 적어도 하나의 개선된 성능, 개선된 효율성, 또는 계산 비용들의 감소 중 적어도 하나를 촉진하는

컴퓨터-구현 방법.

청구항 11

런타임 추적들에 기초하여 모놀리식 애플리케이션으로부터 마이크로서비스들을 생성하기 위한 프로세스를 용이하게 하는 컴퓨터 프로그램 제품에 있어서, 상기 컴퓨터 프로그램 제품은 프로그램 명령들이 구현된 컴퓨터-판

독가능 저장 매체를 포함하고, 상기 프로그램 명령들은 프로세서에 의해 실행 가능하며, 상기 프로세서가:

실행된 테스트 케이스들의 런타임 추적들(runtime traces of executed test cases)에 기초하여 모놀리식 애플리케이션에서 클래스들의 클러스터 할당들(cluster assignments of classes)을 학습하도록 모델을, 상기 프로세서에 의해, 훈련하는 단계; 및

상기 모놀리식 애플리케이션의 하나 혹은 그 이상의 마이크로서비스들을 식별하기 위해 상기 클러스터 할당들에 기초하여 상기 클래스들의 클러스터들을 생성하도록 상기 모델을, 상기 프로세서에 의해, 채용하는 단계를 수행하도록 하는

컴퓨터 프로그램 제품.

청구항 12

제11항에 있어서, 상기 프로그램 명령들은 상기 프로세서가:

상기 모놀리식 애플리케이션의 데이터 의존성(data dependency) 또는 상기 모놀리식 애플리케이션의 정적 호출 그래프(a static call graph) 중 적어도 하나에 기초하여 상기 하나 혹은 그 이상의 마이크로서비스들을, 상기 프로세서에 의해, 개량하는 단계를 수행하도록 더 실행 가능하게 하는

컴퓨터 프로그램 제품.

청구항 13

제11항에 있어서, 상기 프로그램 명령들은 상기 프로세서가:

상기 모놀리식 애플리케이션 또는 상기 모놀리식 애플리케이션에서 클래스들의 클러스터 할당들(the cluster assignments of the classes) 중 적어도 하나의 1차 시간 의존성들 또는 고차 시간 의존성들(first order temporal dependencies or high order temporal dependencies) 중 적어도 하나를 캡처하기 위해 실행된 테스트 케이스들의 런타임 추적들(the runtime traces of the executed test cases)에 기초하여 하나 혹은 그 이상의 인과 그래프들(causal graphs)을, 상기 프로세서에 의해, 생성하는 단계를 수행하도록 더 실행 가능하게 하는

컴퓨터 프로그램 제품.

청구항 14

제11항에 있어서, 상기 프로그램 명령들은 상기 프로세서가:

상기 실행된 테스트 케이스들의 런타임 추적들에 기초하여 생성된 하나 혹은 그 이상의 인과 그래프들의 인과 시퀀스들을 사용하여 상기 모놀리식 애플리케이션에서 클래스들의 클러스터 할당들 또는 그래프 임베딩들 중 적어도 하나를 학습하도록 모델을, 상기 프로세서에 의해, 훈련하는 단계를 수행하도록 더 실행가능하게 하는

컴퓨터 프로그램 제품.

청구항 15

제11항에 있어서, 상기 실행된 테스트 케이스들은 런타임 추적들에서 상기 모놀리식 애플리케이션의 비즈니스 기능들을 제공하는 비즈니스 기능 테스트 케이스들을 포함하는

컴퓨터 프로그램 제품.

청구항 16

제1항에 있어서, 상기 컴퓨터 실행 가능 컴포넌트들은 상기 모놀리식 애플리케이션에서 실행된 테스트 케이스들의 런타임 추적들을 수집하는 수집 컴포넌트를 더 포함하는

시스템.

청구항 17

제6항에 있어서, 모놀리식 애플리케이션 상에서 실행된 테스트 케이스들의 런타임 추적들을, 프로세서에 동작 가능하게 결합된 시스템에 의해, 수집하는 단계를 더 포함하는

컴퓨터-구현 방법.

발명의 설명

기술 분야

배경 기술

- [0001] [0001] 본 발명은 모놀리식 애플리케이션으로부터 마이크로서비스들의 생성에 관한 것으로, 보다 상세하게는, 런타임 추적들에 기초하여 하는 모놀리식 애플리케이션(a monolithic application based on runtime traces)으로부터 마이크로서비스들의 생성에 관한 것이다.

발명의 내용

해결하려는 과제

과제의 해결 수단

- [0002] [0002] 다음은 본 발명의 하나 또는 그 이상의 실시예에 대한 기본적인 이해를 제공하기 위한 요약을 제시한다. 이 요약은 핵심 또는 중요한 요소를 식별하거나 특정 실시예의 범위 또는 청구범위의 범위를 설명하기 위한 것이 아니다. 그것의 유일한 목적은 나중에 제시될 보다 상세한 설명에 대한 요약으로서 단순화된 형태로 개념을 제시하는 것이다. 본 명세서에 기술된 하나 또는 그 이상의 실시예들에서, 런타임 추적들에 기초하여 모놀리식 애플리케이션으로부터 마이크로서비스의 생성을 용이하게 하는 시스템, 장치, 컴퓨터 구현 방법, 및/또는 컴퓨터 프로그램 제품이 기술된다.
- [0003] [0003] 일 실시 예에 따라, 시스템은 컴퓨터 실행 가능한 컴포넌트들을 저장하는 메모리 및 상기 메모리에 저장된 상기 컴퓨터 실행 가능 컴포넌트들을 실행하는 프로세서를 포함할 수 있다. 상기 컴퓨터 실행 가능 컴포넌트들은 실행된 테스트 케이스들의 런타임 추적들(runtime traces of executed test cases)에 기초하여 모놀리식 애플리케이션에서 클래스들의 클러스터 할당들(cluster assignments of classes)을 학습하는 모델 컴포넌트(a model component)를 포함할 수 있다. 상기 컴퓨터 실행 가능 컴포넌트들은 상기 모놀리식 애플리케이션의 하나 혹은 그 이상의 마이크로서비스들을 식별하기 위해 상기 클러스터 할당들에 기초하여 상기 클래스들의 클러스터들을 생성하도록 상기 모델 컴포넌트를 채용하는 클러스터 컴포넌트(a cluster component)를 더 포함할 수 있다. 그러한 시스템의 장점은 클래스의 향상된 기능 그룹화(예: 기능 클러스터링)를 촉진하여 하나 또는 그 이상의 마이크로서비스들을 보다 정확하게 식별하고 모놀리식 애플리케이션의 향상된 애플리케이션 현대화를 가능하게 할 수 있다는 것이다.
- [0004] [0004] 일부 실시 예들에서, 상기 실행된 테스트 케이스들은 런타임 추적들에서 상기 모놀리식 애플리케이션의 비즈니스 기능들을 제공하는 비즈니스 기능 테스트 케이스들을 포함한다. 그러한 시스템의 장점은 클래스의 향상된 기능 그룹화(예: 기능 클러스터링)를 촉진하여 하나 또는 그 이상의 마이크로서비스들을 보다 정확하게 식별하고 모놀리식 애플리케이션의 향상된 애플리케이션 현대화를 가능하게 할 수 있다는 것이다.
- [0005] [0005] 다른 실시 예에 따라, 컴퓨터-구현되는 방법은 실행된 테스트 케이스들의 런타임 추적들(runtime traces of executed test cases)에 기초하여 모놀리식 애플리케이션에서 클래스들의 클러스터 할당들(cluster assignments of classes)을 학습하도록 모델을, 프로세서에 작동 가능하게 결합된 시스템에 의해, 훈련하는 단계를 포함할 수 있다. 상기 컴퓨터-구현되는 방법은 상기 모놀리식 애플리케이션의 하나 혹은 그 이상의 마이크로서비스들을 식별하기 위해 상기 클러스터 할당들에 기초하여 상기 클래스들의 클러스터들을 생성하도록 상기 모델을, 상기 시스템에 의해, 채용하는 단계를 더 포함할 수 있다. 그러한 컴퓨터 구현 방법의 장점은 클래스의 향상된 기능 그룹화(예: 기능 클러스터링)를 촉진하여 하나 또는 그 이상의 마이크로서비스들을 보다 정확하게 식별하고 모놀리식 애플리케이션의 향상된 애플리케이션 현대화를 가능하게 하도록 구현될 수 있다는 것이다.
- [0006] [0006] 일부 실시 예들에서, 상기 실행된 테스트 케이스들은 런타임 추적들에서 상기 모놀리식 애플리케이션의 비즈니스 기능들을 제공하는 비즈니스 기능 테스트 케이스들을 포함한다. 그러한 시스템의 장점은 클래스의 향상된 기능 그룹화(예: 기능 클러스터링)를 촉진하여 하나 또는 그 이상의 마이크로서비스들을 보다 정확하게 식

별하고 모듈리식 애플리케이션의 향상된 애플리케이션 현대화를 가능하게 할 수 있다는 것이다.

[0007] 또 다른 실시 예에 따라, 런타임 추적들에 기초하여 모듈리식 애플리케이션으로부터 마이크로서비스들을 생성하기 위한 프로세스를 용이하게 하는 컴퓨터 프로그램 제품이 제공된다. 상기 컴퓨터 프로그램 제품은 프로그램 명령들이 구현된 컴퓨터-판독가능 저장 매체를 포함하고, 상기 프로그램 명령들은 프로세서에 의해 실행 가능하며, 상기 프로세서가 실행된 테스트 케이스들의 런타임 추적들(runtime traces of executed test cases)에 기초하여 모듈리식 애플리케이션에서 클래스들의 클러스터 할당들(cluster assignments of classes)을 학습하도록 모델을, 상기 프로세서에 의해, 훈련하는 단계를 수행하게 한다. 상기 프로그램 명령들은 상기 프로세서가 상기 모듈리식 애플리케이션의 하나 혹은 그 이상의 마이크로서비스들을 식별하기 위해 상기 클러스터 할당들에 기초하여 상기 클래스들의 클러스터들을 생성하도록 상기 모델을, 상기 프로세서에 의해, 채용하는 단계를 수행하도록 더 실행 가능하게 한다. 그러한 컴퓨터 프로그램 제품의 장점은 하나 또는 그 이상의 마이크로서비스들을 보다 정확하게 식별하고 모듈리식 애플리케이션의 향상된 애플리케이션 현대화를 가능하게 하기 위해 클래스의 향상된 기능 그룹화(예: 기능 클러스터링)를 촉진하도록 구현될 수 있다는 것이다.

[0008] 일부 실시 예들에서, 상기 실행된 테스트 케이스들은 런타임 추적들에서 상기 모듈리식 애플리케이션의 비즈니스 기능들을 제공하는 비즈니스 기능 테스트 케이스들을 포함한다. 그러한 시스템의 장점은 클래스의 향상된 기능 그룹화(예: 기능 클러스터링)를 촉진하여 하나 또는 그 이상의 마이크로서비스들을 보다 정확하게 식별하고 모듈리식 애플리케이션의 향상된 애플리케이션 현대화를 가능하게 할 수 있다는 것이다.

[0009] 일 실시 예에 따라, 시스템은 컴퓨터 실행 가능한 컴포넌트들을 저장하는 메모리 및 상기 메모리에 저장된 상기 컴퓨터 실행 가능 컴포넌트들을 실행하는 프로세서를 포함할 수 있다. 상기 컴퓨터 실행 가능 컴포넌트들은 실행된 테스트 케이스들의 런타임 추적들(runtime traces of executed test cases)에 기초하여 모듈리식 애플리케이션에서 클래스들의 클러스터 할당들(cluster assignments of classes)을 학습하는 모델 컴포넌트(a model component)를 포함할 수 있다. 상기 컴퓨터 실행 가능 컴포넌트들은 상기 모듈리식 애플리케이션의 하나 혹은 그 이상의 마이크로서비스들을 식별하기 위해 상기 클러스터 할당들에 기초하여 상기 클래스들의 클러스터들을 생성하도록 상기 모델 컴포넌트를 채용하는 클러스터 컴포넌트(a cluster component)를 더 포함할 수 있다. 그러한 시스템의 장점은 클래스의 향상된 기능 그룹화(예: 기능 클러스터링)를 촉진하여 하나 또는 그 이상의 마이크로서비스들을 보다 정확하게 식별하고 모듈리식 애플리케이션의 향상된 애플리케이션 현대화를 가능하게 할 수 있다는 것이다.

[0010] 일부 실시 예들에서, 상기 실행된 테스트 케이스들은 런타임 추적들에서 상기 모듈리식 애플리케이션의 비즈니스 기능들을 제공하는 비즈니스 기능 테스트 케이스들을 포함한다. 그러한 시스템의 장점은 클래스의 향상된 기능 그룹화(예: 기능 클러스터링)를 촉진하여 하나 또는 그 이상의 마이크로서비스들을 보다 정확하게 식별하고 모듈리식 애플리케이션의 향상된 애플리케이션 현대화를 가능하게 할 수 있다는 것이다.

[0011] 다른 실시 예에 따라, 컴퓨터-구현되는 방법은 실행된 테스트 케이스들의 런타임 추적들(runtime traces of executed test cases)에 기초하여 모듈리식 애플리케이션에서 클래스들의 클러스터 할당들(cluster assignments of classes)을 학습하도록 모델을, 프로세서에 작동 가능하게 결합된 시스템에 의해, 훈련하는 단계를 포함할 수 있다. 상기 컴퓨터-구현되는 방법은 상기 모듈리식 애플리케이션의 하나 혹은 그 이상의 마이크로서비스들을 식별하기 위해 상기 클러스터 할당들에 기초하여 상기 클래스들의 클러스터들을 생성하도록 상기 모델을, 상기 시스템에 의해, 채용하는 단계를 더 포함할 수 있다. 그러한 컴퓨터 구현 방법의 장점은 클래스의 향상된 기능 그룹화(예: 기능 클러스터링)를 촉진하여 하나 또는 그 이상의 마이크로서비스들을 보다 정확하게 식별하고 모듈리식 애플리케이션의 향상된 애플리케이션 현대화를 가능하게 하도록 구현될 수 있다는 것이다.

[0012] 일부 실시 예들에서, 상기 실행된 테스트 케이스들은 런타임 추적들에서 상기 모듈리식 애플리케이션의 비즈니스 기능들을 제공하는 비즈니스 기능 테스트 케이스들을 포함한다. 그러한 시스템의 장점은 클래스의 향상된 기능 그룹화(예: 기능 클러스터링)를 촉진하여 하나 또는 그 이상의 마이크로서비스들을 보다 정확하게 식별하고 모듈리식 애플리케이션의 향상된 애플리케이션 현대화를 가능하게 할 수 있다는 것이다.

도면의 간단한 설명

[0013] 도 1은, 본 명세서에 기술된 하나 혹은 그 이상의 실시 예들에 따른 런타임 추적들에 기초하여 모듈리식 애플리케이션으로부터 마이크로서비스들의 생성을 용이하게 할 수 있는 비-제한적 시스템의, 예시적인 블록도를 도시한다.

[0014] 도 2는, 본 명세서에 기술된 하나 혹은 그 이상의 실시 예들에 따른 런타임 추적들에 기초하여 모듈리식

애플리케이션으로부터 마이크로서비스들의 생성을 용이하게 할 수 있는 비-제한적인 시스템의, 예시적인 블록도를 도시한다.

[0015] 도 3은, 본 명세서에 기술된 하나 혹은 그 이상의 실시 예들에 따른 런타임 추적들에 기초하여 모놀리식 애플리케이션으로부터 마이크로서비스들의 생성을 용이하게 할 수 있는 비-제한적인 컴퓨터-구현 방법의, 예시적인 플로 다이어그램을 도시한다.

[0016] 도 4는, 본 명세서에 기재된 하나 혹은 그 이상의 실시 예들에 따른 런타임 추적들에 기초하여 모놀리식 애플리케이션으로부터 마이크로서비스들의 생성을 용이하게 할 수 있는 비-제한적 모델의, 예시적인 다이어그램을 도시한다.

[0017] 도 5는, 본 명세서에 기술된 하나 혹은 그 이상의 실시 예들에 따른 런타임 추적들에 기초하여 모놀리식 애플리케이션으로부터 마이크로서비스들의 생성을 용이하게 할 수 있는 비-제한적인 컴퓨터-구현 방법의, 예시적인 플로 다이어그램을 도시한다.

[0018] 도 6은, 본 명세서에 기술된 하나 혹은 그 이상의 실시 예들에 따른 런타임 추적들에 기초하여 모놀리식 애플리케이션으로부터 마이크로서비스들의 생성을 용이하게 할 수 있는 비-제한적인 컴퓨터-구현 방법의, 예시적인 플로 다이어그램을 도시한다.

[0019] 도 7은, 본 명세서에 기술된 하나 혹은 그 이상의 실시 예들에 따른 런타임 추적들에 기초하여 모놀리식 애플리케이션으로부터 마이크로서비스들의 생성을 용이하게 할 수 있는 비-제한적인 컴퓨터-구현 방법의, 예시적인 플로 다이어그램을 도시한다.

[0020] 도 8은, 본 명세서에 기술된 하나 혹은 그 이상의 실시 예들이 용이해질 수 있는 비-제한적인 동작 환경의, 예시적인 블록도를 도시한다.

[0021] 도 9는, 본 발명의 하나 혹은 그 이상의 실시 예들에 따른 비-제한적인 클라우드 컴퓨팅 환경의, 예시적인 블록도를 도시한다.

[0022] 도 10은, 본 발명의 하나 혹은 그 이상의 실시 예들에 따른 비-제한적인 추상적 모델 계층들의, 예시적인 블록도를 도시한다.

발명을 실시하기 위한 구체적인 내용

[0014] [0023] 다음의 상세한 설명은 단지 예시적인 것이며 실시예들 및/또는 실시예들의 적용 또는 사용들을 제한하도록 의도되지 않는다. 뿐만 아니라, 앞의 백그라운드 또는 요약 섹션들이나, 상세한 설명 섹션에 제시된 명시적 또는 묵시적 정보에 구속될 의도가 없다.

[0015] [0024] 하나 혹은 그 이상의 실시예들이 이제 도면들을 참조하여 설명되며, 여기에서 유사한 참조 번호들은 전체에 걸쳐 유사한 요소들을 나타내기 위해 사용된다. 다음 설명에서, 설명의 목적들을 위해, 하나 혹은 그 이상의 실시예들의 더 완전한 이해를 제공하기 위해 다수의 특정 세부사항들이 제시된다. 분명하게, 그러나, 다양한 경우들에, 하나 혹은 그 이상의 실시예들이 이러한 특정 세부사항들 없이 실시될 수 있다.

[0016] [0025] 애플리케이션 현대화(Application modernization)는 모놀리식 애플리케이션(a monolith application) (모놀리식 애플리케이션(a monolithic application)이라고도 함)을 독립형 마이크로서비스들(standalone microservices)로 리팩토링(refactoring)하는 프로세스이다. 기존의 모놀리식 엔터프라이즈 애플리케이션은 복잡하고 뒤얽힌 프레젠테이션, 비즈니스 프로세스, 및 데이터 모델들을 사용하여 설계되었다. 기존의 모놀리식 엔터프라이즈 애플리케이션은 유지관리 및 변경이 어렵다. 마이크로서비스(microservice)는 작은 기능들 세트를 캡슐화 하는 느슨하게 결합된 독립형 애플리케이션(standalone application)이고 공지된 인터페이스를 통해 다른 애플리케이션들과 상호 작용할 수 있다. 따라서, 마이크로서비스는 클라우드 컴퓨팅 환경에서 업데이트, 확장, 유지 관리 및 배치하기가 훨씬 쉽다.

[0017] [0026] 일부 기존 애플리케이션 현대화 기술들은 정적 코드 분석-기반 접근 방식(a static code analysis-based approach)을 적용하여 모놀리식 애플리케이션에서 마이크로서비스들을 생성한다. 이러한 정적 코드 분석-기반 접근 방식은 호출 관계들의 포괄적인 관점(a comprehensive view of the call relations)을 제공하는 소스를 분석한다. 그러한 정적 코드 분석-기반 접근 방식을 사용하는 것의 문제는 특정 워크로드들 및 입력들 하에서 호출 관계들 및 상호 작용 빈도들을 캡처하지 못한다는 것이다.

[0018] [0027] 일부 기존 애플리케이션 현대화 기술들은 메타-데이터-기반 접근 방식(a meta-data-based approach)을

적용하여 모놀리식 애플리케이션에서 마이크로서비스들을 생성한다. 그러한 메타-데이터-기반 접근 방식은, 예를 들어, 데이터-플로 다이어그램들, 소프트웨어 아티팩트들, 코드 기반 변경 이력(data-flow diagrams, software artifacts, code base change history) 와 같은, 설계 문서들(design documents) 및/또는 다른 설계 문서에 의존한다. 그러한 메타-데이터-기반 접근 방식을 사용하는 것의 문제점은 전술한 설계 문서들이 항상 이용 가능하고 액세스 가능한 것은 아니라는 점이다.

[0019] [0028] 일부 기존 애플리케이션 현대화 기술들은 동적 마이크로서비스 구성-기반 접근 방식(a dynamic microservice composition-based approach)을 적용하여 모놀리식 애플리케이션에서 마이크로서비스들을 생성한다. 그러한 동적 마이크로서비스 구성-기반 접근 방식은 운영 데이터(operational data)를 이용하여 런타임 동안 동적으로 적응하는 마이크로서비스 모델들(dynamically adapting microservice models)을 생성한다. 그러한 동적 마이크로서비스 구성-기반 접근 방식을 사용하는 것의 문제점은 운영 데이터를 수집하기가 어렵다는 것이다.

[0020] [0029] 기존의 일부 애플리케이션 현대화 기술들은 워크로드-데이터-기반 접근 방식(a workload-data-based approach)을 적용하여 모놀리식 애플리케이션에서 마이크로서비스들을 생성한다. 그러한 워크로드-데이터-기반 접근 방식은 운영 데이터(예를 들어: 로그 파일)를 사용하여 데이터 수집 후 고정된 마이크로 서비스 모델들(fixed microservice models)을 생성한다. 그러한 워크로드-데이터-기반 접근 방식을 사용하는 것의 문제점은 운영 데이터를 수집하기 어렵다는 것이다.

[0021] [0030] 일부 기존 애플리케이션 현대화 기술들은 런타임 추적들을 이용하여 다음 단계들을 구현함으로써 클래스 그룹화들(class groupings)을 학습하는 접근 방식을 적용한다: 1) 모든 클래스들은 서로 다른 클러스터들에 할당된다; 2) 클러스터들의 각 쌍에 대한 자카드 유사성 계수(Jaccard Similarity Coefficient)가 계산되고, 최대 자카드 유사성을 갖는 클러스터들이 병합된다; 그리고 3) 일부 최적화 목표(예를 들어, 내부(intra)-연결 최대화 및 상호(inter)-연결 최소화)에 따라 결과들을 개량하기 위해 지네틱 알고리즘들(the Genetic algorithm)이 적용된다. 그러한 접근 방식을 사용하는 것의 문제는 그러한 접근 방식은 좋은 비즈니스 기능 응집력(good business function cohesion)을 가진 클래스들의 그룹화들을 생성하는 데 필수적인 비즈니스 컨텍스트들을 고려하지 않는다는 것이다. 그러한 접근 방식을 사용하는 것의 또 다른 문제는 그러한 접근 방식은 고차 시간 의존성(high order temporal dependency)을 고려하지 않는다는 것이다. 그러한 접근 방식을 사용하는 것의 또 다른 문제는 그러한 접근 방식은 실행 추적들(the run traces)을 그래프에 집계한다(aggregate)는 것이며, 그 결과 풍부한 시간관련 정보(rich temporal information)를 상실하게 되어 잘못된 전이 관계들(erroneous transitive relations)을 가정한다는 것이다. 그러한 접근 방식을 사용하는 것의 또 다른 문제는 그러한 접근 방식은 호출량은 직접 최소화하지 않고 클러스터 품질을 개량하기 위한 비즈니스 컨텍스트를 최소화한다는 것이다.

[0022] [0031] 전술한 기존 애플리케이션 현대화 기술들의 문제들을 해결하기 위해, 본 발명은 이들 문제들에 대한 솔루션을 시스템들, 컴퓨터-구현 방법들, 및/또는 컴퓨터 프로그램 제품들의 형태로 생성하도록 구현될 수 있으며, 상기 솔루션은 모놀리식 애플리케이션을 사용하여 실행될 수 있는 테스트 케이스들의 런타임 추적들에 기초하여 모놀리식 애플리케이션에서 클러스터 할당들 및/또는 클래스들의 그래프 임베딩들을 학습하도록 모델을 훈련할 수 있고; 및/또는 모놀리식 애플리케이션의 하나 혹은 그 이상의 마이크로서비스들을 식별하기 위해 상기 클러스터 할당들 및/또는 상기 그래프 임베딩들에 기초하여 상기 클래스들의 클러스터들을 생성하도록 상기 모델을 채용할 수 있다. 그러한 시스템들, 컴퓨터-구현 방법들, 및/또는 컴퓨터 프로그램 제품들의 장점은 하나 혹은 그 이상의 마이크로서비스들을 보다 정확하게 식별하고 모놀리식 애플리케이션의 애플리케이션 현대화를 개선하는 것을 가능하게 하기 위해 클래스들의 개선된 기능 그룹화(예를 들어, 기능 클러스터링)를 촉진하도록 구현될 수 있다는 것이다. 일부 실시예들에서, 테스트 케이스들은 런타임 추적들에서 모놀리식 애플리케이션의 비즈니스 기능들을 제공하는 비즈니스 기능 테스트 케이스들을 포함할 수 있다. 그러한 시스템들, 컴퓨터-구현 방법들, 및/또는 컴퓨터 프로그램 제품들의 장점은 하나 혹은 그 이상의 마이크로서비스들을 보다 정확하게 식별하고 모놀리식 애플리케이션의 애플리케이션 현대화를 개선하는 것을 가능하게 하기 위해 클래스들의 개선된 기능 그룹화(예를 들어, 기능 클러스터링)를 촉진하도록 구현될 수 있다는 것이다.

[0023] [0032] 도 1은, 본 명세서에 기술된 하나 혹은 그 이상의 실시예들에 따른 런타임 추적들에 기초하여 모놀리식 애플리케이션으로부터 마이크로서비스들의 생성을 용이하게 할 수 있는, 예시적인 비-제한적인 시스템(100)의 블록도를 도시한다. 시스템(100)은, 클라우드 컴퓨팅 환경과 연관될 수 있는, 마이크로서비스 생성 시스템(102)을 포함할 수 있다. 예를 들어, 마이크로서비스 생성 시스템(102)은 도 9를 참조하여 아래에서 설명되는 클라우드 컴퓨팅 환경(950) 및/또는 도 10(예를 들어, 하드웨어 및 소프트웨어 계층(1060), 가상화 계층(1070), 관리 계층(1080) 및/또는 워크로드 계층(1090))을 참조하여 아래에서 설명되는 하나 혹은 그 이상의 기능적 추상

적 계층들과 연관될 수 있다.

- [0024] [0033] 마이크로서비스 생성 시스템(102) 및/또는 그에 관한 컴포넌트들(예를 들어, 모델 컴포넌트 108, 클러스터 컴포넌트 110, 수집 컴포넌트 202, 제2 모델 컴포넌트(204), 개량 컴포넌트 206, 등)은 여기에 설명된 본 발명의 하나 혹은 그 이상의 실시예들에 따라 하나 혹은 그 이상의 동작들을 실행하기 위해 도 9를 참조하여 아래에서 설명되는 클라우드 컴퓨팅 환경(950)의 하나 혹은 그 이상의 컴퓨팅 자원들을 채용할 수 있거나, 및/또는 도 10을 참조하여 아래에서 설명되는 하나 혹은 그 이상의 기능적 추상적 계층들(예를 들어, 양자 소프트웨어, 등)을 채용할 수 있다. 예를 들어, 클라우드 컴퓨팅 환경(950) 및/또는 그러한 하나 혹은 그 이상의 기능적 추상적 계층들은 여기에 설명된 본 발명의 하나 혹은 그 이상의 실시예들에 따라 하나 혹은 그 이상의 동작들을 실행하기 위해 마이크로서비스 생성 시스템(102) 및/또는 그에 관한 컴포넌트들에 의해 채용될 수 있는 하나 혹은 그 이상의 클래식 컴퓨팅 디바이스들(예를 들어, 클래식 컴퓨터, 클래식 프로세서, 가상 머신, 서버, 등), 양자 하드웨어 및/또는 양자 소프트웨어(예를 들어, 양자 컴퓨팅 디바이스, 양자 컴퓨터, 양자 프로세서, 양자 회로 시뮬레이션 소프트웨어, 초전도 회로, 등)를 포함할 수 있다. 예를 들어, 마이크로서비스 생성 시스템(102) 및/또는 그에 관한 컴포넌트들은 여기에 설명된 본 발명의 하나 혹은 그 이상의 실시예들에 따라 하나 혹은 그 이상의 클래식 및/또는 양자: 수학 함수, 계산 및/또는 방정식; 컴퓨팅 및/또는 처리 스크립트; 알고리즘; 모델 (예를 들어, 인공 지능(AI) 모델, 머신 러닝(ML) 모델, 등); 및/또는 다른 동작을 실행하기 위해 그러한 하나 또는 그 이상의 클래식 및/또는 양자 컴퓨팅 자원들을 채용할 수 있다.
- [0025] [0034] 본 발명은 클라우드 컴퓨팅에 대한 설명을 포함할 수 있지만, 본 명세서에서 인용된 기술적 사상들의 구현이 클라우드 컴퓨팅 환경에 국한되지 않는다는 것을 이해해야 한다. 오히려, 본 발명의 실시 예들은 현재 알려져 있거나 나중에 개발된 모든 다른 유형의 컴퓨팅 환경과 함께 구현될 수 있다.
- [0026] [0035] 클라우드 컴퓨팅은 최소한의 관리 노력 또는 서비스 제공자와의 상호작용으로 빠르게 제공되고 해제될 수 있는, 구성 가능한(configurable) 컴퓨팅 자원들(예를 들어, 네트워크들, 네트워크 대역폭, 서버들, 처리, 메모리, 스토리지, 애플리케이션들, 가상 머신들, 및 서비스들)의 공유 풀에 대한 편리한, 주문형(on-demand) 네트워크 액세스를 가능하게 하는 모델이다. 이 클라우드 모델은 가용성을 촉진하며 적어도 5가지의 필수 특성들(essential characteristics), 적어도 3가지 서비스 모델(service models), 및 적어도 4가지 배치 모델(deployment models)로 구성될 수 있다
- [0027] [0036] 특성들은 다음과 같다:
- [0028] [0037] 주문형 셀프-서비스(On-demand self-service): 클라우드 소비자는, 서비스 제공자와의 인적 상호작용을 필요로 하지 않고 필요한 만큼 자동적으로, 서버 시간(server time) 및 네트워크 스토리지 같은 컴퓨팅 용량들을 일방적으로 제공(provision)할 수 있다.
- [0029] [0038] 광역 네트워크 액세스(Broad network access): 혼성의 썬 또는 썬 클라이언트 플랫폼들(heterogeneous thin or thick client platforms)(예를 들어, 모바일폰들, 랩탑들, 및 PDA들)에 의한 사용을 장려하는 표준 메커니즘들을 통해 액세스되는 기능들을 네트워크를 통해서 이용할 수 있다.
- [0030] [0039] 자원 풀링(Resource pooling): 제공자의 컴퓨팅 자원들은, 각기 다른 물리적 및 가상 자원들을 요구(demand)에 따라 동적으로 할당 및 재할당하는, 멀티-테넌트 모델(a multi-tenant model)을 사용하는 다수의 소비자들에게 서비스할 수 있도록 풀에 넣어둔다(pooled). 소비자는 일반적으로 제공된 자원들의 정확한 위치를 제어할 수 없거나 그에 대한 지식이 없지만 더 높은 추상 수준에서(예를 들어, 국가, 주, 또는 데이터센터) 위치를 명시할 수 있다는 점에서 위치 독립성이 있다.
- [0031] [0040] 기민한 탄력성(Rapid elasticity): 용량들(capabilities)이 기민하게 탄력적으로 제공되어 (어떤 경우엔 자동으로) 신속히 규모를 확장할 수도 있고(scale out) 그리고 탄력적으로 해제되어 신속히 규모를 축소할 수도 있다(scale in). 소비자에게 제공할 수 있는 가능성이 종종 무제한이고 언제든지 원하는 수량으로 구매할 수 있는 것처럼 보인다.
- [0032] [0041] 측정 가능한 서비스(Measured service): 클라우드 시스템들은 자원 사용을 자동으로 제어하고 최적화하는데, 서비스의 유형(예를 들어, 스토리지, 처리, 대역폭, 및 활성 사용자 계정)에 적절한 추상화 수준에서(at some level of abstraction) 계측 기능을 이용하여서 그렇게 한다. 자원 사용량은 모니터 되고, 제어되고, 그리고 보고될 수 있으며 이로써 이용하는 서비스의 제공자와 사용자 모두에게 투명성을 제공한다.
- [0033] [0042] 서비스 모델들(Service Models)은 다음과 같다:

- [0034] [0043] 소프트웨어 서비스(Software as a Service)(SaaS): 소비자에게 제공되는 서비스는 클라우드 하부구조 상에서 실행되는 제공자의 애플리케이션들을 사용하게 해주는 것이다. 애플리케이션들은 웹 브라우저(예를 들어, 웹기반 이메일) 같은 썬(thin) 클라이언트 인터페이스를 통해 여러 클라이언트 장치들에서 액세스 가능하다. 소비자는 네트워크, 서버들, 운영 체제들, 스토리지, 또는 개별 애플리케이션 능력들을 포함하는 하부 클라우드 하부구조를 관리하거나 제어하지 않는다. 단, 제한된 사용자-특정 애플리케이션 구성 세팅들은 예외로서 가능하다.
- [0035] [0044] 플랫폼 서비스(Platform as a Service)(PaaS): 소비자에게 제공되는 서비스는 제공자에 의해 지원되는 프로그래밍 언어들 및 도구들을 이용하여 생성된 소비자-생성 또는 획득 애플리케이션들을 클라우드 하부구조에 배치하게 해주는 것이다. 소비자는 네트워크, 서버들, 운영 체제들, 또는 스토리지를 포함하는 하부 클라우드 하부구조를 관리하거나 제어하지 않는다. 그러나 배치된 애플리케이션들에 대해서 그리고 가능한 경우 애플리케이션 호스팅 환경 구성들에 대해서 제어할 수 있다.
- [0036] [0045] 하부구조 서비스(Infrastructure as a Service)(IaaS): 소비자에게 제공되는 서비스는 처리, 스토리지, 네트워크, 및 기타 기본 컴퓨팅 자원들을 제공하여 주는 것이며, 여기서 소비자는 임의의 소프트웨어를 배치 및 실행할 수 있고, 이 소프트웨어에는 운영 체제들과 애플리케이션들이 포함될 수 있다. 소비자는 하부 클라우드 하부구조를 관리하거나 제어하지 않지만, 운영 체제들, 스토리지, 배치된 애플리케이션들, 및 가능한 경우 선택된 네트워킹 컴포넌트들의 제한적인 제어(예를 들어, 호스트 방화벽들)에 대하여 제어할 수 있다.
- [0037] [0046] 배치 모델들(Deployment Models)은 다음과 같다:
- [0038] [0047] 사설 클라우드(Private cloud): 클라우드 하부구조는 오직 한 조직(an organization)을 위해서 운영되고, 그 조직 또는 제3자에 의해 관리될 수 있으며 옥내(on-premises) 또는 옥외(off-premises)에 위치할 수 있다.
- [0039] [0048] 커뮤니티 클라우드(Community cloud): 클라우드 하부구조는 여러 조직들에 의해 공유되고 관심사(예를 들어, 선교, 보안 요건, 정책, 및 규정 준수 심사)를 공유하는 특정 커뮤니티를 지원하며, 여러 조직들 또는 제3자에 의해 관리될 수 있으며 옥내(on-premises) 또는 옥외(off-premises)에 위치할 수 있다.
- [0040] [0049] 공공 클라우드(Public cloud): 클라우드 하부구조는 일반 대중 또는 대규모 산업 집단에서 이용할 수 있으며 클라우드 서비스를 판매하는 조직이 소유한다.
- [0041] [0050] 하이브리드 클라우드(Hybrid cloud): 클라우드 하부구조는 둘 또는 그 이상의 클라우드들(사설, 커뮤니티, 또는 공공)이 혼합된 구성이며, 이들은 고유한 주체들로 있지만 데이터 및 애플리케이션 이식가능성(portability)을 가능하게 해주는 표준화된 또는 소유권 있는 기술(예를 들어, 클라우드들 사이의 부하 균형을 위한 클라우드 버스팅(cloud bursting))에 의해 서로 결합되어 있다.
- [0042] [0051] 클라우드 소프트웨어는 스테이트리스(statelessness)(예외들이 있는), 낮은 결합(low coupling), 모듈 방식(modularity), 및 의미적 상호운용성(semantic interoperability)에 집중하는 서비스를 지향함으로써 클라우드 패러다임의 모든 장점을 취한다는 것에 주목해야 한다. 클라우드 컴퓨팅의 중심에는 상호 연결된 노드의 네트워크로 구성된 하부구조가 있다.
- [0043] [0052] 마이크로서비스 생성 시스템(102)은 메모리(104), 프로세서(106), 모델 컴포넌트(108), 클러스터 컴포넌트(110), 및/또는 버스(112)를 포함할 수 있다.
- [0044] [0053] 여기에 개시된 다양한 도면들에 도시된 본 발명의 실시예들은 단지 예시를 위한 것임을 이해해야 하며, 및 이와 같이, 이러한 실시예들의 아키텍처는 여기에 도시된 시스템들, 디바이스들, 및/또는 컴포넌트들로 제한되지 않는다. 예를 들어, 일부 실시예들에서, 시스템(100) 및/또는 마이크로서비스 생성 시스템(102)은 운영 환경(800) 및 도 8을 참조하여 여기에 설명된 다양한 컴퓨터 및/또는 컴퓨팅-기반 요소들을 더 포함할 수 있다. 여러 실시예들에서, 그러한 컴퓨터 및/또는 컴퓨팅-기반 요소들은 도 1 또는 여기에 개시된 다른 도면들과 관련하여 도시되고 설명된 시스템들, 디바이스들, 컴포넌트들, 및/또는 컴퓨터-구현 동작들 중 하나 혹은 그 이상을 구현하는 것과 관련하여 사용될 수 있다.
- [0045] [0054] 메모리(104)는, 프로세서(106)(예를 들어, 클래식 프로세서, 양자 프로세서, 등)에 의해 실행될 때, 실행 가능한 컴포넌트(들) 및/또는 명령(들)에 의해 정의된 동작들의 수행을 용이하게 할 수 있게 하는 하나 혹은 그 이상의 컴퓨터 및/또는 기계 판독 가능, 쓰기 가능, 및/또는 실행 가능한 컴포넌트들 및/또는 명령들을 저장할 수 있다. 예를 들어, 메모리(104)는 컴퓨터 및/또는 기계 판독 가능, 쓰기 가능, 및/또는 실행 가능한 컴포

넌트들 및/또는 명령들을 저장할 수 있고, 이들은, 프로세서(106)에 의해 실행될 때, 마이크로서비스 생성 시스템(102), 모델 컴포넌트(108), 클러스터 컴포넌트(110), 및/또는 다양한 도면을 참조하거나 참조하지 않고 여기에 설명된 마이크로서비스 생성 시스템(102)(예를 들어, 수집 컴포넌트(202), 제2 모델 컴포넌트(204), 개량 컴포넌트(206), 등.)과 연관된 다른 컴포넌트와 관련하여 여기에 설명된 다양한 기능들의 실행을 용이하게 할 수 있다.

[0046] [0055] 메모리(104)는 하나 혹은 그 이상의 메모리 아키텍처들을 채용할 수 있는 휘발성 메모리(예를 들어, 랜덤 액세스 메모리(RAM), 정적 RAM(SRAM), 동적 RAM(DRAM), 등.) 및/또는 비-휘발성 메모리(예를 들어, 읽기 전용 메모리(ROM), 프로그램 가능 ROM(PROM), 전기적으로 프로그램 가능한 ROM(EPROM), 전기적으로 지울 수 있는 프로그램 가능한 ROM(EEPROM), 등.)를 포함할 수 있다. 메모리(104)의 추가 예들은 시스템 메모리(816) 및 도 8을 참조하여 아래에서 설명된다. 메모리(104)의 그러한 예들은 본 발명의 모든 실시예들을 구현하기 위해 사용될 수 있다.

[0047] [0056] 프로세서(106)는 메모리(104)에 저장될 수 있는 하나 혹은 그 이상의 컴퓨터 및/또는 기계 판독 가능, 쓰기 가능 및/또는 실행 가능한 컴포넌트들 및/또는 명령들을 구현할 수 있는 하나 혹은 그 이상의 유형들의 프로세서들 및/또는 전자 회로(예를 들어, 클래식 프로세서, 양자 프로세서, 등.)를 포함할 수 있다. 예를 들어, 프로세서(106)는, 논리, 제어, 입/출력(I/O), 산술 등을 포함하지만, 이에 제한되지 않는, 컴퓨터 및/또는 기계 판독 가능, 쓰기 가능 및/또는 실행 가능한 컴포넌트들 및/또는 명령들에 의해 명시될 수 있는 다양한 동작들을 수행할 수 있다. 일부 실시 예들에서, 프로세서(106)는 하나 혹은 그 이상의 중앙 처리 유닛, 멀티-코어 프로세서, 마이크로프로세서, 듀얼 마이크로프로세서들, 마이크로컨트롤러, 시스템 온 칩(SOC), 어레이 프로세서, 벡터 프로세서, 양자 프로세서, 및/또는 다른 유형의 프로세서를 포함할 수 있다. 프로세서(106)의 추가 예들은 처리 유닛(814) 및 도 8을 참조하여 아래에서 설명된다. 프로세서(106)의 그러한 예들은 본 발명의 모든 실시예들을 구현하기 위해 사용될 수 있다.

[0048] [0057] 마이크로서비스 생성 시스템(102), 메모리(104), 프로세서(106), 모델 컴포넌트(108), 클러스터 컴포넌트(110), 및/또는 여기에 설명된 마이크로서비스 생성 시스템(102)의 다른 컴포넌트는(예를 들어, 수집 컴포넌트 202, 두 번째 모델 컴포넌트 204, 개량 컴포넌트 206, 등.) 통신적으로, 전기적으로, 작동적으로, 및/또는 광학적으로 시스템(100), 마이크로서비스 생성 시스템(102), 및/또는 그와 결합된 모든 컴포넌트들의 기능을 수행하기 위해 버스(112)를 통해 서로 연결될 수 있다. 버스(112)는 하나 혹은 그 이상의 메모리 버스, 메모리 컨트롤러, 주변 버스, 외부 버스, 로컬 버스, 양자 버스, 및/또는 다양한 버스 아키텍처들을 사용할 수 있는 다른 유형의 버스를 포함할 수 있다. 버스(112)의 추가 예들은 시스템 버스(818) 및 도 8을 참조하여 아래에서 설명된다. 버스(112)의 이러한 예들은 본 발명의 모든 실시예들을 구현하기 위해 사용될 수 있다.

[0049] [0058] 마이크로서비스 생성 시스템(102)은 프로세서를 포함하고 및/또는 유선 및/또는 무선 네트워크와 효과적 및/또는 작동 가능한 통신이 가능할 수 있는 모든 유형의 컴포넌트, 기계, 디바이스, 시설, 장치(apparatus), 및/또는 도구를 포함할 수 있다. 그러한 실시 예들이 모두 가능하다. 예를 들어, 마이크로서비스 생성 시스템(102)은 서버 디바이스, 컴퓨팅 디바이스, 범용 컴퓨터, 특수-목적 컴퓨터, 양자 컴퓨팅 디바이스(예: 양자 컴퓨터), 태블릿 컴퓨팅 디바이스, 핸드헬드 디바이스, 서버 클래스 컴퓨팅 기계 및/또는 데이터베이스, 랩탑 컴퓨터, 노트북 컴퓨터, 데스크탑 컴퓨터, 휴대 전화, 스마트 폰, 소비자 가전 및/또는 기기(instrumentation), 산업 및/또는 상업용 디바이스, 디지털 어시스턴트, 멀티미디어 인터넷 가능 전화, 멀티미디어 플러그들 및/또는 다른 유형의 디바이스를 포함할 수 있다.

[0050] [0059] 마이크로서비스 생성 시스템(102)은 데이터 케이블(예를 들어, 고-화질 멀티미디어 인터페이스(HDMI), 권장 표준(RS) 232, 이더넷 케이블, 등.)을 통해 하나 혹은 그 이상의 외부 시스템들, 소스들, 및/또는 디바이스들(예를 들어, 클래식 및/또는 양자 컴퓨팅 디바이스들, 통신 디바이스들, 등.)에 결합될 수 있다(예를 들어, 통신적으로, 전기적으로, 동작적으로, 광학적으로, 등.). 일부 실시 예들에서, 마이크로서비스 생성 시스템(102)은 네트워크를 통해 하나 혹은 그 이상의 외부 시스템들, 소스들, 및/또는 디바이스들(예를 들어, 클래식 및/또는 양자 컴퓨팅 디바이스들, 통신 디바이스들, 등.)에 결합될 수 있다(예를 들어, 통신적으로, 전기적으로, 동작적으로, 광학적으로, 등.).

[0051] [0060] 일부 실시예들에서, 그러한 네트워크는, 셀룰러 네트워크, 광역 네트워크(WAN)(예를 들어, 인터넷) 또는 근거리 영역 네트워크(LAN)를, 포함하지만, 이에 제한되지 않는, 유선 및 무선 네트워크들을 포함할 수 있다. 예를 들어, 마이크로서비스 생성 시스템(102)은 하나 혹은 그 이상의 외부 시스템들, 소스들 및/또는 디바이스들, 예를 들어, 컴퓨팅 디바이스들(및 그 반대도 마찬가지)과 사실상 모든 유선 또는 무선 기술을 사용하여 통

신할 수 있으며, 상기 유선 또는 무선 기술은 다음을 포함하지만 이에 제한되지 않는다: 와이-파이(Wi-Fi), 글로벌 모바일 통신들 시스템(GSM), 범용 모바일 전화통신들 시스템(UMTS), 마이크로웨이브 액세스를 위한 전세계 상호 운용성(WiMAX), 향상된 일반 패킷 무선 서비스(향상된 GPRS), 3세대 파트너십 프로젝트(3GPP) 롱 텀 에볼루션 (LTE), 3GPP2(3세대 파트너십 프로젝트 2) 울트라 모바일 브로드밴드(UMB), HSPA(High Speed Packet Access), Zigbee 및 기타 802.XX 무선 기술들 및/또는 레저서 전화통신 기술들, BLUETOOTH®, SIP(세션 초기화 프로토콜), ZIGBEE®, RF4CE 프로토콜, WirelessHART 프로토콜, 6LoWPAN(저전력 무선 영역 네트워크들을 통한 IPv6), Z-Wave, ANT, 초광대역(UWB) 표준 프로토콜 및/또는 기타 독점 및 비-독점적인 통신 프로토콜들. 그러한 예에서, 마이크로서비스 생성 시스템(102)은 따라서 마이크로서비스 생성 시스템(102)과 외부 시스템들, 소스들 및/또는 디바이스들(예를 들어, 컴퓨팅 디바이스들, 통신 디바이스들, 등) 사이에서 정보를 통신하는 것을 용이하게 하는 하드웨어(예를 들어, 중앙 처리 장치(CPU), 트랜시버, 디코더, 양자 하드웨어, 양자 프로세서 등), 소프트웨어(예를 들어, 스레드들 세트, 일련의 프로세스들 세트, 실행 중인 소프트웨어, 양자 펄스 일정, 양자 회로, 양자 게이트들, 등) 또는 하드웨어 및 소프트웨어의 조합을 포함할 수 있다.

[0052]

[0061] 마이크로서비스 생성 시스템(102)은 하나 혹은 그 이상의 컴퓨터 및/또는 기계 판독 가능, 쓰기 가능, 실행 가능한 컴포넌트들 및 또는 명령들을 포함할 수 있고, 이들은, 프로세서(106)(예를 들어, 고전적 프로세서, 양자 프로세서, 등)에 의해 실행될 때, 그러한 컴포넌트(들) 및/또는 명령(들)에 의해 정의된 동작들의 수행을 용이하게 할 수 있다. 게다가, 다양한 실시예들에서, 마이크로서비스 생성 시스템(102)과 관련된 모든 컴포넌트는, 다양한 도면들을 참조하거나 참조하지 않고 여기에 설명된 바와 같이, 하나 혹은 그 이상의 컴퓨터 및/또는 머신 판독 가능, 쓰기 가능, 및/또는 실행 가능한 컴포넌트들 및 또는 명령들을 포함할 수 있고, 이들은, 프로세서(106)(예를 들어, 고전적 프로세서, 양자 프로세서, 등)에 의해 실행될 때, 그러한 컴포넌트(들) 및/또는 명령(들)에 의해 정의된 동작들의 수행을 용이하게 할 수 있다. 예를 들어, 모델 컴포넌트(108), 클러스터 컴포넌트(110), 및/또는 여기에 개시된 마이크로서비스 생성 시스템(102)과 관련된 모든 다른 컴포넌트들(예를 들어, 마이크로서비스 생성 시스템(102)에 의해 통신, 전자, 작동 및/또는 광학적으로 결합된 및/또는 채용된)은 그러한 컴퓨터 및/또는 머신 판독 가능, 쓰기 가능, 및/또는 실행 가능 컴포넌트(들) 및/또는 명령(들)을 포함할 수 있다. 그 결과, 다수의 실시예들에 따르면, 마이크로서비스 생성 시스템(102) 및/또는 여기에 개시된 바와 같이 이와 관련된 모든 컴포넌트들은, 마이크로서비스 생성 시스템(102) 및/또는 이와 관련된 컴포넌트들을 참조하여 여기에 설명된 하나 혹은 그 이상의 동작의 수행을 용이하게 하기 위해 그러한 컴퓨터 및/또는 머신 판독 가능, 쓰기 가능, 및/또는 실행 가능 컴포넌트(들) 및/또는 명령(들)을 실행하도록 프로세서(106)를 채용 할 수 있다.

[0053]

[0062] 마이크로서비스 생성 시스템(102)은 모델 컴포넌트(108), 클러스터 컴포넌트(110), 및/또는 여기에 개시된 마이크로서비스 생성 시스템(102)과 연관된 다른 컴포넌트(예를 들어, 수집 컴포넌트 202, 제2 모델 컴포넌트 204, 개량 컴포넌트 206, 등)에 의한 및/또는 와 연관된 동작들의 수행을 용이하게 할 수 있다(예를 들어, 프로세서 106를 통해). 예를 들어, 아래에서 상세히 설명되는 바와 같이, 마이크로서비스 생성 시스템(102)은 프로세서(106)(예를 들어, 고전적 프로세서, 양자 프로세서, 등)를 통해: 실행된 테스트 케이스들의 런타임 추적들에 기초하여 모놀리식 애플리케이션에서 클래스들의 클러스터 할당들을 학습하기 위한 모델을 훈련하는 단계; 및/또는 모놀리식 애플리케이션의 하나 혹은 그 이상의 마이크로서비스들을 식별하기 위해 상기 클러스터 할당들에 기초하여 상기 클래스들의 클러스터들을 생성하기 위한 모델을 채용하는 단계를 수행하는 것을 용이하게 할 수 있다. 본 명세서에서 참조되는 바와 같이, 클러스터는 하나 혹은 그 이상의 정의된 목적들(예를 들어, 비즈니스 컨텍스트, 호출 량, 등)에 기초하여 여기서 설명된 바와 같은 마이크로서비스 생성 시스템(102) 및/또는 그와 관련된 하나 혹은 그 이상의 컴포넌트들(예를 들어, 모델 컴포넌트 108, 클러스터 컴포넌트 110, 수집 컴포넌트 202, 제2 모델 컴포넌트 204, 개량 컴포넌트 206, 등)에 의해 함께 그룹화될 수 있는 클래스들 그룹으로 정의될 수 있다. 보다 구체적으로는, 여기서 참조된 바와 같은, 마이크로서비스 리팩토링(애플리케이션 현대화라고도 함)에 대한 모놀리식(monolith)의 적용 영역에서, 클러스터는 마이크로서비스로 구성될 수 있는 클래스들의 그룹으로 정의될 수 있다.

[0054]

[0063] 위의 예에서, 아래에 상세히 설명되는 바와 같이, 마이크로서비스 생성 시스템(102)은 프로세서(106)(예를 들어, 고전적 프로세서, 양자 프로세서 등)를 통해: 모놀리식 애플리케이션의 데이터 의존성(data dependency) 또는 모놀리식 애플리케이션의 정적 호출 그래프(a static call graph) 중 적어도 하나에 기초하여 하나 혹은 그 이상의 마이크로서비스들을 개량하는 단계(refining); 모놀리식 애플리케이션 또는 모놀리식 애플리케이션에서 클래스들의 클러스터 할당들(the cluster assignments of the classes) 중 적어도 하나의 1차 시간 의존성들 또는 고차 시간 의존성들(first order temporal dependencies or high order temporal dependencies) 중 적어도 하나를 캡처하기 위해 실행된 테스트 케이스들의 런타임 추적들(the runtime traces

of the executed test cases)에 기초하여 하나 혹은 그 이상의 인과 그래프들(causal graphs)을 생성하는 단계; 및/또는 상기 실행된 테스트 케이스들의 런타임 추적들에 기초하여 생성된 하나 혹은 그 이상의 인과 그래프들의 인과 시퀀스들(causal sequences of one or more causal graphs)을 사용하여 모놀리식 애플리케이션에서 클래스들의 클러스터 할당들 또는 그래프 임베딩들(graph embeddings) 중 적어도 하나를 학습하도록 모델을 훈련하는 단계를 수행하는 것을 더 용이하게 할 수 있다. 위의 예에서, 상기 실행된 테스트 케이스들은 런타임 추적들에서 모놀리식 애플리케이션의 비즈니스 기능들을 제공하는 비즈니스 기능 테스트 케이스들을 포함할 수 있다.

[0055] [0064] 다른 예에서, 아래에 상세히 설명되는 바와 같이, 마이크로서비스 생성 시스템(102)은 프로세서(106)(예를 들어, 고전적 프로세서, 양자 프로세서, 등)를 통해: 모놀리식 애플리케이션에서 실행된 테스트 케이스들의 런타임 추적들을 수집하는 단계(collecting); 상기 런타임 추적들에 기초하여 모놀리식 애플리케이션에서 클래스들의 클러스터 할당들을 학습하도록 모델을 훈련하는 단계; 및/또는 모놀리식 애플리케이션의 하나 혹은 그 이상의 마이크로서비스들을 식별하기 위해 상기 클러스터 할당들에 기초하여 클래스들의 클러스터들을 생성하도록 모델을 채용하는 단계를 수행하는 것을 더 용이하게 할 수 있다. 이 예에서, 아래에 상세히 설명되는 바와 같이, 마이크로서비스 생성 시스템(102)은 프로세서(106)(예를 들어, 고전적 프로세서, 양자 프로세서, 등)를 통해: 모놀리식 애플리케이션의 데이터 의존성 또는 모놀리식 애플리케이션의 정적 호출 그래프 중 적어도 하나에 기초하여 하나 혹은 그 이상의 마이크로서비스들을 개량하는 단계; 모놀리식 애플리케이션 또는 모놀리식 애플리케이션에서 클래스들의 클러스터 할당들 중 적어도 하나의 1차 시간 의존성들 또는 고차 시간 의존성들 중 적어도 하나를 캡처하기 위해 런타임 추적들에 기초하여 하나 혹은 그 이상의 인과 그래프들을 생성하는 단계; 및/또는 상기 런타임 추적에 기초하여 생성된 하나 혹은 그 이상의 인과 그래프들의 인과 시퀀스들을 사용하여 모놀리식 애플리케이션에서 클래스들의 클러스터 할당들 또는 그래프 임베딩들 중 적어도 하나를 학습하도록 모델을 훈련하는 단계를 포함한다. 이 예에서, 상기 테스트 케이스들은 런타임 추적들에서 모놀리식 애플리케이션의 비즈니스 기능들을 제공하는 비즈니스 기능 테스트 케이스들로 구성될 수 있다.

[0056] [0065] 모델 컴포넌트(108)는 실행된 테스트 케이스들의 런타임 추적에 기초하여 모놀리식 애플리케이션에서 클래스들의 클러스터 할당들을 학습할 수 있다. 예를 들어, 모델 컴포넌트(108)는 인공 지능(AI) 및/또는 머신 러닝(ML) 모델(예를 들어, 신경망)을 포함할 수 있으며, 이 모델은 그러한 테스트 케이스들을 실행하기 위해(예를 들어, 프로세서 106를 통해) 모놀리식 애플리케이션을 채용함으로써 생성될 수 있는 런타임 추적들에 기초하여 모놀리식 애플리케이션(예를 들어, 모놀리식 엔터프라이즈 애플리케이션)에서 클래스들의 클러스터 할당들을 학습하도록 훈련될 수 있다(예를 들어, 아래에 설명된 바와 같이 마이크로서비스 생성 시스템(102)을 통해). 일부 실시 예들에서, 모델 컴포넌트(108)는 아래에 설명되며 도 4에 예시된 모델(400)을 포함할 수 있고, 모델(400)은 그러한 테스트 케이스들을 실행하기 위해(예를 들어, 프로세서 106를 통해) 모놀리식 애플리케이션을 채용함으로써 생성될 수 있는 런타임 추적들에 기초하여 모놀리식 애플리케이션(예를 들어, 모놀리식 엔터프라이즈 애플리케이션)에서 클래스들의 클러스터 할당들을 학습하도록 훈련될 수 있는(예를 들어, 아래에 설명된 바와 같이 마이크로서비스 생성 시스템(102)을 통해) 신경망을 포함할 수 있다.

[0057] [0066] 일 예에서, 전술한 실행된 테스트 케이스들은 모놀리식 애플리케이션을 사용하여 실행될 수 있는(예를 들어, 프로세서 106를 통해 모놀리식 애플리케이션에서 실행될 수 있는) 비즈니스 기능 테스트 케이스들을 포함할 수 있다. 이 예에서, 그러한 비즈니스 기능 테스트 케이스들은 런타임 추적들에서 모놀리식 애플리케이션의 비즈니스 기능들을 제공할 수 있다. 예를 들어, 그러한 비즈니스 기능 테스트 케이스들은 마이크로서비스 생성 시스템(102) 및/또는 모델 컴포넌트(108)에 모놀리식 애플리케이션의 클래스들 및 기능 호출들이 상호 작용하는 방법의 순서와 빈도를 제공할 수 있고 또한 마이크로서비스 생성 시스템(102) 및/또는 모델 컴포넌트(108)가 비즈니스 컨텍스트를 각 추적과 연관시킬 수 있게 하여 개선된 그룹화를 제공하도록(예를 들어, 마이크로서비스 생성 시스템 102, 모델 컴포넌트 108, 및/또는 아래 설명되는 클러스터 컴포넌트 110를 통해)할 수 있다.

[0058] [0067] 모델 컴포넌트(108)는 전술한 바와 같이 모놀리식 애플리케이션에서 테스트 케이스들을 실행함으로써 생성될 수 있는 런타임 추적들에 기초하여 생성된 하나 혹은 그 이상의 인과 그래프들의 인과 시퀀스들을 사용하여 모놀리식 애플리케이션에서 클래스들의 클러스터 할당들 및/또는 모놀리식 애플리케이션에서 그러한 클래스들의 그래프 임베딩들을 학습할 수 있다. 일 예에서, 모델 컴포넌트(108)는 전술한 런타임 추적들에 기초하여 생성될 수 있는(예를 들어, 도 2를 참조하여 아래에서 설명되는 바와 같은 제2 모델 컴포넌트 204에 의해) 하나 혹은 그 이상의 인과 그래프들의 인과 시퀀스들을 사용하여 위에서 설명된 그러한 클러스터 할당들 및 그래프 임베딩들을 동시에 학습할 수 있다. 이 예에서, 위에서 설명된 그러한 하나 혹은 그 이상의 인과 그래프들의 생성(예를 들어, 제2 모델 컴포넌트 204에 의한)은 마이크로서비스 생성 시스템(102) 및/또는 모델 컴포넌트(10

8)에 다음을 제공할 수 있다: 모놀리식 애플리케이션의 1차 시간 의존성들 및/또는 고차 시간 의존성들; 및/또는 모놀리식 애플리케이션에서 클래스의 클러스터 할당들.

[0059] [0068] 위에서 설명된 모델 컴포넌트(108)에 의한 그러한 학습을 용이하게 하기 위해, 마이크로서비스 생성 시스템(102)은 모델 컴포넌트(108)를 훈련할 수 있다. 예를 들어, 모델 컴포넌트(108)는, 예를 들어, 위에서 설명된 하나 혹은 그 이상의 인과 그래프들의 인과 시퀀스들을 사용하여 위에서 설명된 클러스터 할당들 및/또는 그 그래프 임베딩들을 학습하기 위해 마이크로서비스 생성 시스템(102)에 의해 훈련될 수 있는 신경망(예를 들어, 모델 400)과 같은, 인공 지능(AI) 및/또는 머신 러닝(ML) 모델을 포함할 수 있다. 예를 들어, 마이크로서비스 생성 시스템(102)은 아래에 설명된 훈련 절차를 구현함으로써 앞서 설명된 클러스터 할당들 및/또는 그래프 임베딩들을 학습하도록 모델 컴포넌트(108)를 훈련할 수 있다.

[0060] [0069] 위에서 설명된 클러스터 할당들 및/또는 그래프 임베딩들을 학습하도록 모델 컴포넌트(108)를 훈련하기 위해 마이크로서비스 생성 시스템(102)이 구현할 수 있는 훈련 절차를 예시하기 위해, 일 실시예에서, 전술한 모놀리식 애플리케이션은, 예를 들어, 하나 혹은 그 이상의 금융 시장들에서 보안 자산들을 거래하는 데 채용할 수 있는 예시적 거래 애플리케이션(trading application)과 같은, 예시적인 자바 2 플랫폼 엔터프라이즈 에디션(Java 2 Platform Enterprise Edition: J2EE) 애플리케이션을 포함할 수 있다. 이 실시예에서, 상기 예시적인 거래 애플리케이션은 사용자들이 로그인하여, 그들의 포트폴리오를 살펴보고(view), 주식 시세들을 조회하여(lookup stock quotes), 및/또는 주식 지분들을 사고 팔 수 있게 하는 온라인 주식 거래 시스템을 위한 J2EE 애플리케이션을 포함할 수 있다. 이 실시예에서, 상기 예시적인 거래 애플리케이션에서 테스트 케이스들(예를 들어, 비즈니스 기능 테스트 케이스들)을 실행함으로써 생성될 수 있는 런타임 추적들은 3-층 아키텍처 프리젠테이션 계층, 비즈니스 로직, 및 지속성 계층(a 3-tier architecture presentation layer, business logic, and persistence layer)을 나타낼 수 있는 멀티-스레드 런타임 추적들(multi-threaded runtime traces)을 포함할 수 있다. 이 실시예에서, 그러한 런타임 추적들은 상기 예시적 거래 애플리케이션에서 테스트 케이스들(예를 들어, 비즈니스 기능 테스트 케이스들)을 실행함으로써(예를 들어, 프로세서 106를 통해) 생성될 수 있고, 상기 테스트 케이스들은, 초기화, 로그인, 로그오프, 거래 계정, 거래 계정 업데이트 프로필, 시장 요약 용어집, 거래 시세 매수, 거래 포트폴리오 매도(init, login, log-off, trade account, trade account update profile, market summary glossary, trade quotes buy, trade portfolio sell) 및/또는 기타 테스트 케이스들을 포함할 수 있지만, 이에 국한되지는 않는다. 이 실시예에서, 하나의 클래스 및 다른 클래스들에 대한 상기 클래스의 호출은 다수의 비즈니스 컨텍스트들을 포함할 수 있다. 이 실시예에서, 상기 예시적인 거래 애플리케이션에서 위에서 정의된 그러한 테스트 케이스들을 실행함으로써 생성될 수 있는 런타임 추적들은 다음 런타임 추적들을 포함할 수 있다:

[0061] [0070] 1567027540073,init,Root calls MarketSummarySingleton

[0062] [0071] 1567027540074,init,MarketSummarySingleton calls Log

[0063] [0072] 1567027540075,init,Log calls TradeConfig

[0064] [0073] 1567027540076,init,TradeConfig returns to Log

[0065] [0074] 1567027540077 ,init,Log calls TradeConfig

[0066] [0075] 1567027540078,init,TradeConfig returns to Log

[0067] [0076] 1567027540079,init,Log returns to MarketSummarySingleton

[0068] [0077]

[0069] [0078] 1567027743085,trade-quotes-buy,Root calls OrdersAlertFilter

[0070] [0079] 1567027743086,trade-quotes-buy,OrdersAlertFilter calls TradeConfig

[0071] [0080] 1567027743087,trade-quotes-buy,TradeConfig returns to OrdersAlertFilter

[0072] [0081] Root - MarketSummarySingleton - Log - TradeConfig - (TradeConfig-Log) -- log - MarketSummarySingleton **init**

[0073] [0082] Root - OrdersAlertFilter - TradeConfig - (TradeConfig-OrdersAlertFilter) -- OrdersAlertFilter trade-quotes-buy

- [0074] [0083] 이 예시적인 실시예에서, 모놀리식 애플리케이션은 예시적인 거래 애플리케이션을 포함하며, 마이크로서비스 생성 시스템(102)은 전술한 런타임 추적들에 기초하여 인과관계 시퀀스들을 포함하는 하나 혹은 그 이상의 인과관계 그래프들을 생성할 수 있다. 이 실시예에서, 마이크로서비스 생성 시스템(102)은 모놀리식 애플리케이션의 1차 시간 의존성들 및/또는 고차 시간 의존성들 및/또는 모놀리식 애플리케이션에서 클래스들의 클러스터 할당들을 캡처하기 위해 그러한 하나 혹은 그 이상의 인과 그래프들을 생성할 수 있다. 이 실시예에서, 그러한 하나 혹은 그 이상의 인과 그래프들의 생성을 용이하게 하기 위해, 마이크로서비스 생성 시스템(102)은, 예를 들어, 전술한 인과 그래프들을 생성하기 위한 고차 시간 신경망과 같은, 신경망을 포함할 수 있는 제2 모델 컴포넌트(예를 들어, 도 2를 참조하여 아래에서 설명되는 제2 모델 컴포넌트 204)를 채용할 수 있다. 이 실시예에서, 마이크로서비스 생성 시스템(102)은 경로의 시간적 관계들(the temporal relations of the paths)을 캡처하고(예를 들어, 전술한 인과 시퀀스들을 포함하는 인과 그래프들의 생성을 통해) 고도로 관련된 호출 시퀀스들(highly related call sequences)을 추론(extrapolate)하기 위해 그러한 제2 모델 컴포넌트(예를 들어, 도 2를 참조하여 아래에서 설명되는 제2 모델 컴포넌트 204)를 채용할 수 있다.
- [0075] [0084] 이 예시적인 실시예에서, 모놀리식 애플리케이션은 예시적 거래 애플리케이션을 포함하고, 마이크로서비스 생성 시스템(102)은 위에서 설명된 하나 혹은 그 이상의 인과 그래프들의 인과 시퀀스들(또는 런타임 인과 시퀀스들 및/또는 런타임 시퀀스들이라고도 함)을 입력으로 사용하여 위에서 설명된 클러스터 할당들 및/또는 그래프 임베딩들을 학습하도록 모델 클러스터(108)를 훈련할 수 있으며, 출력은 노드 임베딩들(예를 들어, a 매핑 $f: V \rightarrow R^d$) 및 클러스터 할당들을 포함한다. 이 실시예에서, 마이크로서비스 생성 시스템(102)은 런타임 인과 시퀀스들로부터 이웃 노드들을 직접 샘플링하도록 모델 컴포넌트(108)를 훈련할 수 있다. 이 실시예에서, 런타임 시퀀스들을 S 라 하면, 노드 v 를 포함하는 윈도우들의 수집(the collection)은 $N_s(V)$ 로 표시될 수 있다. 이 실시예에서, 마이크로서비스 생성 시스템(102)은 다음의 예시적인 루틴을 구현함으로써 인과 시퀀스들로부터 하나 혹은 그 이상의 양의 샘플 쌍들(positive sample pairs)을 생성하도록 모델 컴포넌트(108)를 훈련할 수 있다:
- [0076] [0085] Given the runtime sequence “Root, MarketSummarySingleton, Log, TradeConfig, TradeConfig returns to Log, Log” and an observation window of size 3.
- [0077] [0086]
- [0078] [0087] The node “MarketSummarySingleton” has neighborhood window $N_s(v=MarketSummarySingleton)$.
- [0079] [0088] [Root, MarketSummarySingleton, Log]
- [0080] [0089] [MarketSummarySingleton, Log, TradeConfig]
- [0081] [0090] Extract positive sample pair:
- [0082] [0091] (MarketSummarySingleton, Root)
- [0083] [0092] (MarketSummarySingleton, Log)
- [0084] [0093] (MarketSummarySingleton, TradeConfig)
- [0085] [0094] 이 예시적인 실시예에서, 모놀리식 애플리케이션은 예시적인 거래 애플리케이션을 포함하고, 모델 컴포넌트(108)는, 예를 들어, 도 4에 예시된 모델(400)과 같은, 신경망을 포함할 수 있다. 이 실시예에서, 마이크로서비스 생성 시스템(102)은 다음 작업(task)을 위해 모델 컴포넌트(108)를 훈련함으로써 위에서 설명된 클러스터 할당들 및/또는 그래프 임베딩들을 학습하도록 모델 컴포넌트(108)(예를 들어, 모델 400)를 훈련할 수 있다: 런타임 시퀀스에서 특정 클래스를 A 라 하면, 마이크로서비스 생성 시스템(102)은 런타임 시퀀스에서 A 의 "인접" 클래스가 되는 모든 클래스에 대한 확률을 예측하도록 모델 컴포넌트(108)를 훈련할 수 있다. 이 실시예에서, 만일 두개의 자바(Java) 클래스들이 매우 유사한 "컨텍스트들"을 갖는다면, 그 것은 이들 두개의 클래스들이 동일한 비즈니스 컨텍스트 하에서 동일한 컨텍스트 창 내에서 동시-발생할 가능성이 있음과 마이크로서비스 생성 시스템(102)은 이들 두개의 클래스들에 대한 유사한 임베딩을 출력하도록 모델 컴포넌트(108)를 훈련할 수 있음을 의미할 수 있다. 시퀀스-기반 임베딩을 이용하는 이 실시예에서, 인접 노드들의 시퀀스들은, 예를 들어, 랜덤 워크와 같은, 방법을 사용하여 인과 그래프(들)로부터 샘플링될 수 있고(예를 들어, 마이크로서비스 생성 시스템 102 및/또는 모델 컴포넌트 108를 통해) 그 목적은 노드들의 이웃들을 관찰함에 있어서 음의(negative) 로 그 가능성을 최소화하는 것이다.

[0086] [0095] 이 예시적인 실시예에서, 모듈리식 애플리케이션이 예시적인 거래 애플리케이션을 포함하고, 마이크로서비스 생성 시스템(102)은 아래에 정의된 등식(1)을 채용함으로써 노드 임베딩에 대해 조건부로 노드의 이들 이웃들을 관찰함에 있어서 음의 로그 가능성을 최소화하기 위한 위에서 설명된 상기 목적을 달성하도록 모델 컴포넌트(108)를 훈련할 수 있다.

[0087] [0096] 등식 (1):

$$\min_f \sum_{v \in V} -\log P(\mathcal{N}_s(i)|f(v))$$

[0088]

[0089] [0097] 여기서 $f(v)$ 는 노드 v에 대한 임베딩을 나타내고, P는 확률 함수를 나타내며, $\mathcal{N}_s(i)$ 는 노드 v 주변의 인접 노드들 수집을 나타낸다.

[0090] [0098] 이 예시적인 실시예에서, 모듈리식 애플리케이션이 예시적인 거래 애플리케이션을 포함하고, 피쳐(feature) 공간에서의 조건부 독립 가정 및 대칭에 따라, 위에서 정의된 등식(1)은 마이크로서비스 생성 시스템(102) 및/또는 모델 컴포넌트(108)에 의해 아래에 정의된 등식(2)와 같이 표현될 수 있다.

[0091] [0099] 등식 (2):

$$L_h = \min_f \sum_{v \in V} (\ln(\sum_{u \in V} \exp(f(v) * f(u)) - \sum_{n_i \in \mathcal{N}_s(i)} (f(n_i) * f(v)))$$

[0092]

[0093] [0100] 여기서 L_h 는 임베딩을 위한 최적화 함수를 나타내고, u는 노드 u를 나타내며, $f(u)$ 는 노드 u의 임베딩을 나타내고, $f(n_i)$ 는 노드 n_i 에 대한 임베딩을 나타낸다.

[0094] [0101] 이 예시적인 실시예에서, 모듈리식 애플리케이션은 예시적인 거래 애플리케이션을 포함하고, 마이크로서비스 생성 시스템(102)은 위에서 설명된 클러스터 할당들 및/또는 그래프 임베딩들을 동시에 학습하도록 모델 컴포넌트(108)를 훈련할 수 있다. 예를 들어, 그러한 동시 학습을 용이하게 하기 위해, 마이크로서비스 생성 시스템(102)은 등식(3)으로 아래에 정의된 k-평균 비용 함수(the k-means cost function)를 포함하는 목적 함수(the objective function)(예를 들어, 식 (2))를 확장하도록 모델 컴포넌트(108)를 훈련할 수 있다.

[0095] [0102] 등식 (3):

$$L_c = L_h + \gamma \cdot \sum_{u \in V} \min_{c \in C} |f(v) - u_c|^2$$

[0096]

[0097] [0103] 여기서 L_c 는 임베딩과 클러스터링 모두에 대한 전체 최적화 함수를 나타내고, γ 는 클러스터 비용의 가중치 계수를 결정하는 하이퍼파라미터를 나타내며, u_c 는 c^{th} 번째 클러스터에 대한 클러스터 평균을 나타낸다. 이들은 훈련 가능한 파라미터들을 주목해야 한다.

[0098] [0104] 일부 실시 예들에서, 마이크로서비스 생성 시스템(102)은 위에서 설명된 하나 혹은 그 이상의 인과 그래프들의 인과 시퀀스들을 사용하여 위에서 설명된 클러스터 할당들 및/또는 그래프 임베딩들을 학습하기 위해 모델 컴포넌트(108)를 훈련하도록 하나 혹은 그 이상의 인공 지능(AI) 모델들 및/또는 하나 혹은 그 이상의 머신 러닝(ML) 모델들을 포함할 수 있고 및/또는 채용할 수 있다. 예를 들어, 마이크로서비스 생성 시스템(102)은 위에서 설명된 하나 혹은 그 이상의 인과 그래프들의 인과 시퀀스들을 사용하고 그리고 하나 혹은 그 이상의 비지도 학습 방법들 (예를 들어, 전술한 훈련 절차와 같은, 비지도 클러스터링 방법(an unsupervised clustering method))을 사용하여 위에서 설명된 클러스터 할당들 및/또는 그래프 임베딩들을 학습하기 위해 모델 컴포넌트(108)를 훈련하도록 하나 혹은 그 이상의 AI 및/또는 ML 모델들을 포함할 수 있고 및/또는 채용할 수 있다

[0099] [0105] 일부 실시 예들에서, 마이크로서비스 생성 시스템(102)은 인공 지능의 원리들과 연관된 분류들, 상관들, 추론들 및/또는 표현들에 기초하여 위에서 설명된 하나 혹은 그 이상의 인과 그래프들의 인과 시퀀스들을 사용하여 위에서 설명된 클러스터 할당들 및/또는 그래프 임베딩들을 학습하기 위해 모델 컴포넌트(108)를 훈련할 수 있다. 예를 들어, 마이크로서비스 생성 시스템(102)은 위에서 설명된 하나 혹은 그 이상의 인과 그래프들의 인과 시퀀스들을 사용하여 위에서 설명된 클러스터 할당들 및/또는 그래프 임베딩들을 학습하기 위해 모델 컴포넌트(108)를 훈련하도록 자동 분류 시스템 및/또는 자동 분류 프로세스를 채용할 수 있다. 일 실시예에

서, 마이크로서비스 생성 시스템(102)은 위에서 설명된 하나 혹은 그 이상의 인과 그래프들의 인과 시퀀스들을 사용하여 위에서 설명된 클러스터 할당들 및/또는 그래프 임베딩들을 학습하기 위해 모델 컴포넌트(108)를 훈련하도록 확률론적 및/또는 통계-기반 분석(예를 들어, 분석 유틸리티들 및 비용들로 팩터링하는)(a probabilistic and/or statistical-based analysis (e.g., factoring into the analysis utilities and costs) 을 채용할 수 있다

[0100] [00106] 일부 실시 예들에서, 마이크로서비스 생성 시스템(102)은 위에서 설명된 하나 혹은 그 이상의 인과 그래프들의 인과 시퀀스들을 사용하여 위에서 설명된 클러스터 할당들 및/또는 그래프 임베딩들을 학습하기 위해 모델 컴포넌트(108)를 훈련하도록 모든 적절한 머신 러닝 기반 기술들, 통계-기반 기술들, 및/또는 확률-기반 기술들을 채용할 수 있다. 예를 들어, 마이크로서비스 생성 시스템(102)은 전문가 시스템, 퍼지 로직, 지원 벡터 머신(SVM), 은닉 마르코프 모델들(Hidden Markov Models: HMMs), 그리디 검색 알고리즘들(greedy search algorithms), 규칙-기반 시스템들, 베이저안 모델들(Bayesian models)(예: 베이저안 네트워크들), 신경망들, 다른 비-선형 훈련 기술들, 데이터 융합, 유틸리티-기반 분석 시스템들, 베이저안 모델들을 채용하는 시스템들, 및/또는 다른 모델을 채용할 수 있다. 일부 실시 예들에서, 마이크로서비스 생성 시스템(102)은 위에서 설명된 하나 혹은 그 이상의 인과 그래프들의 인과 시퀀스들을 사용하여 위에서 설명된 클러스터 할당들 및/또는 그래프 임베딩들을 학습하기 위해 모델 컴포넌트(108)를 훈련하는 것과 연관된 머신 러닝 계산들 세트를 수행할 수 있다. 예를 들어, 마이크로서비스 생성 시스템(102)은 위에서 설명된 하나 혹은 그 이상의 인과 그래프들의 인과 시퀀스들을 사용하여 위에서 설명된 클러스터 할당들 및/또는 그래프 임베딩들을 학습하기 위해, 클러스터링 머신 러닝 계산들 세트, 로지스틱 회귀 머신 러닝 계산들 세트, 의사 결정 트리 머신 러닝 계산들 세트, 랜덤 포레스트 머신 러닝 계산들 세트(a set of random forest machine learning computations), 회귀 트리 머신 러닝 계산들 세트, 최소 제곱 머신 러닝 계산들 세트(a set of least square machine learning computations), 인스턴스 기반 머신 러닝 계산들 세트, 회귀 머신 러닝 계산들 세트, 지원 벡터 회귀 머신 러닝 계산들 세트, k-평균들 머신 러닝 계산들 세트, 스펙트럼 클러스터링 머신 러닝 계산들 세트, 규칙 학습 머신 러닝 계산들 세트, 베이저안 머신 러닝 계산들 세트, 딥 볼츠만(Boltzmann) 머신 러닝 계산들 세트, 딥 빌리프(belief) 네트워크 계산들 세트, 및/또는 다양한 머신 러닝 계산들 세트를 수행할 수 있다.

[0101] [00107] 클러스터 컴포넌트(110)는 모놀리식 애플리케이션의 하나 혹은 그 이상의 마이크로서비스들을 식별하기 위해 모놀리식 애플리케이션에서의 클러스터 할당들에 기초하여 모놀리식 애플리케이션에서 클래스들의 클러스터들을 생성하도록 모델 컴포넌트(108)를 채용할 수 있다. 예를 들어, 마이크로서비스 생성 시스템(102)이 위에서 설명된 바와 같이 하나 혹은 그 이상의 인과 그래프들의 인과 시퀀스들을 사용하여 모놀리식 애플리케이션의 클러스터 할당들 및/또는 그래프 임베딩들을 학습하도록 모델 컴포넌트(108)를 훈련하는 것에 기초하여, 클러스터 컴포넌트(110)는 모델 컴포넌트(108)에 의해 학습된 모놀리식 애플리케이션의 그러한 클러스터 할당들 및/또는 그래프 임베딩들에 기초하여 모놀리식 애플리케이션에서 클래스들의 클러스터들을 생성하도록 모델 컴포넌트(108)의 훈련된 버전을 채용할 수 있다. 이 예에서, 위에서 설명된 바와 같이 클러스터 컴포넌트(110)에 의해 생성될 수 있는(예를 들어, 모델 컴포넌트(108)를 채용하는 것을 통해) 클래스들의 클러스터들은 모놀리식 애플리케이션의 하나 혹은 그 이상의 마이크로서비스들(예를 들어, 모놀리식 애플리케이션의 하나 혹은 그 이상의 잠재적인 마이크로서비스 후보들)을 나타낼 수 있고, 이들은 이에 의해 도 2를 참조하여 아래에 설명되는 바와 같이 클러스터 컴포넌트(110)에 의해 식별될 수 있고 및/또는 개량 컴포넌트(206)에 의해 추가로 개량될 수 있다.

[0102] [00108] 도 2는 본 명세서에 기술된 하나 혹은 그 이상의 실시예들에 따른 런타임 추적들에 기초하여 모놀리식 애플리케이션으로부터 마이크로서비스들의 생성을 용이하게 할 수 있는 예시적인 비-제한적인 시스템(200)의 블록도를 도시한다. 시스템(200)은 수집 컴포넌트(202), 제2 모델 컴포넌트(204), 및/또는 개량 컴포넌트(206)를 더 포함할 수 있는 마이크로서비스 생성 시스템(102)을 포함할 수 있다. 각 실시 예들에 채용된 유사한 요소들 및/또는 프로세스에 대한 반복적인 설명은 간결함을 위해 생략한다.

[0103] [00109] 수집 컴포넌트(202)는 모놀리식 애플리케이션에서 실행되는 테스트 케이스들의 런타임 추적들을 수집할 수 있다. 예를 들어, 수집 컴포넌트(202)는 모놀리식 애플리케이션을 사용하여 (예를 들어, 프로세서 106를 통해) 테스트 케이스들을 실행함으로써 생성될 수 있는 런타임 추적들을 수집할 수 있으며, 그러한 테스트 케이스들은 도 1을 참조하여 위에서 설명된 비즈니스 기능 테스트 케이스들을 포함할 수 있다.

[0104] [00110] 전술한 런타임 추적들의 수집을 용이하게 하기 위해, 수집 컴포넌트(202)는 인공 지능(AI) 및 자연어 처리(NLP) 및/또는 명명된-주체 인식(named-entity recognition: NER)에 기반한 머신 러닝(ML) 모델을 채용할 수 있으며, 이들은, 얇은 또는 깊은 신경망 모델, 지원 벡터 머신(SVM) 모델, 결정 트리 분류기, 또는 모놀리식

애플리케이션을 사용하여 전술한 그러한 테스트 케이스들을 실행함으로써 생성된 런타임 추적들의 추출을 용이하게 할 수 있는 모든 지도 또는 비지도 머신 러닝 모델(supervised or unsupervised machine learning model)을 포함하지만, 이에 국한되지 않는다. 예를 들어, 수집 컴포넌트(202)는 모놀리식 애플리케이션의 계층 소스 코드(instrumented source code)로부터 런타임 로그를 생성할 수 있는 모니터링 애플리케이션을 채용할 수 있다. 다른 예에서, 수집 컴포넌트(202)는, 예를 들어, 클래스 이름, 속성들, 방법 이름들, 방법 인자들(method arguments), 반환 유형들, 및/또는 기타 정보와 같은, 모놀리식 애플리케이션의 소스 코드로부터의 정보를 추출하기 위해 파이썬-기반 도구(a python-based tool)를 채용할 수 있고, 그러한 파이썬-기반 도구는 런타임 추적 생성을 위해 모놀리식 애플리케이션에 코드를 추가로 삽입할 수 있다. 다른 예에서, 수집 컴포넌트(202)는 비즈니스 컨텍스트에 기초하여 추적의 생성을 가능하게 하는 자바 프론트-엔드 도구(a Java front-end tool)를 채용할 수 있다. 다른 예에서, 수집 컴포넌트(202)는 상속 관계들(inheritance relationships), 데이터 의존성, 속성들, 방법 인자, 리턴 유형, 및/또는 기타 관계들을 추출할 수 있는 추출 애플리케이션(an extraction application)을 채용할 수 있다.

[0105] [00111] 제2 모델 컴포넌트(204)는 모놀리식 애플리케이션의 1차 시간 의존성들 및/또는 고차 시간 의존성들 및/또는 모놀리식 애플리케이션 클래스들의 클러스터 할당들을 캡처하기 위해 모놀리식 애플리케이션에서 실행된 테스트 케이스들의 런타임 추적들에 기초하여 하나 혹은 그 이상의 인과 그래프들을 생성할 수 있다. 예를 들어, 제2 모델 컴포넌트(204)는 모놀리식 애플리케이션의 1차 시간 의존성들 및/또는 고차 시간 의존성들 및/또는 모놀리식 애플리케이션에서 클래스들의 클러스터 할당들을 캡처하기 위해 모놀리식 애플리케이션에서 비즈니스 기능 테스트 케이스들을 실행함으로써 생성된 런타임 추적들에 기초하여 도 1을 참조하여 위에서 설명된 하나 혹은 그 이상의 인과 그래프들을 생성할 수 있다.

[0106] [00112] 제2 모델 컴포넌트(204)는 신경망을 포함할 수 있다. 이 예에서, 제2 모델 컴포넌트(204)는 위에 설명된 그러한 1차 시간 의존성들 및/또는 고차 시간 의존성들을 캡처하기 위해 도 1을 참조하여 위에서 정의된 런타임 추적들에 기초하여 인과 시퀀스들을 포함하는 하나 혹은 그 이상의 인과 그래프들을 생성할 수 있는 고-차 시간 신경망을 포함할 수 있다. 예를 들어, 제2 모델 컴포넌트(204)는 경로들의 시간적 관계들(the temporal relations of the paths)을 캡처하고 고도로 관련된 호출 시퀀스들(highly related call sequences)을 추론하기 위해 도 1을 참조하여 위에서 정의된 런타임 추적들에 기초하여 인과 시퀀스들을 포함하는 하나 혹은 그 이상의 인과 그래프들을 생성할 수 있는 고-차 시간 신경망을 포함할 수 있다. 이 예에서, 제2 모델 컴포넌트(204)에 의해 생성될 수 있는 그러한 인과 그래프(들) 및/또는 인과 시퀀스들은 위에서 도 1을 참조하여 설명된 바와 같이 모놀리식 애플리케이션의 클러스터 할당들 및/또는 그래프 임베딩들을 학습하기 위해 모델 컴포넌트(108)를 훈련하도록 마이크로서비스 생성 시스템(102)에 의해 사용될 수 있다.

[0107] [00113] 개량 컴포넌트(206)는 모놀리식 애플리케이션의 데이터 의존성 및/또는 모놀리식 애플리케이션의 정적 호출 그래프에 기초하여(예를 들어, 사용하여) 모놀리식 애플리케이션의 하나 혹은 그 이상의 마이크로서비스들을 개량(refine)할 수 있다. 예를 들어, 위에서 설명된 바와 같이 클러스터 컴포넌트(110)에 의해 생성될 수 있는(예를 들어, 모델 컴포넌트 108를 채용하는 것을 통해) 클래스들의 클러스터들은 모놀리식 애플리케이션의 데이터 의존성 및/또는 모놀리식 애플리케이션의 정적 호출 그래프에 기초하여(예를 들어, 사용하여) 아래에 설명되는 바와 같이 개량 컴포넌트(206)에 의해 추가로 개량될 수 있는 모놀리식 애플리케이션의 하나 혹은 그 이상의 마이크로서비스들(예를 들어, 모놀리식 애플리케이션의 하나 혹은 그 이상의 잠재적인 마이크로서비스 후보들)을 나타낼 수 있다.

[0108] [00114] 도 1을 참조하여 위에서 설명된 예시적인 거래 애플리케이션 실시예에서, 모놀리식 애플리케이션의 데이터 의존성 및/또는 모놀리식 애플리케이션의 정적 호출 그래프에 기초하여(예를 들어, 사용하여) 모놀리식 애플리케이션의 그러한 하나 혹은 그 이상의 마이크로서비스들의 개량을 용이하게 하기 위해, 개량 컴포넌트(206)는 추가적인 정규화 항들(additional regularization terms)을 추가하도록 비용 함수(예를 들어, 등식 (3))를 증강시킬 수 있다(augment). 예를 들어, 도 1을 참조하여 위에서 설명된 예시적인 거래 애플리케이션 실시예에서, 개량 컴포넌트(206)는 추가 정규화 항들을 추가하기 위해 비용 함수(예를 들어, 등식 (3))를 증강시켜 아래에 정의된 등식 (4)를 산출할 수 있다. 도 1을 참조하여 전술한 예시적인 거래 애플리케이션 실시예에서, 개량 컴포넌트(206)는 모놀리식 애플리케이션의 데이터 의존성 및/또는 모놀리식 애플리케이션의 정적 호출 그래프에 기초하여(예를 들어, 사용하여) 모놀리식 애플리케이션의 하나 혹은 그 이상의 마이크로서비스들을 개량하기 위해 아래에 정의된 등식 (4)를 이용할 수 있으며, 그러한 데이터 의존성 및/또는 또는 정적 호출 그래프는 전술한 바와 같이 수집 컴포넌트(202)에 의해 수집될 수 있다(예를 들어, 모놀리식 애플리케이션을 사용하여 비즈니스 기능 테스트 케이스들을 실행함으로써 생성된 런타임 추적들에서 수집될 수 있다).

[0109] [00115] 등식 (4)

$$\lambda_2 * \sum_{(u,v) \in c2} \omega_{2(u,v)} * ||f(u) - f(v)||$$

[0110]

[0111] [00116] 여기서 $\omega_{(u,v)}$ 는 정규화 가중치를 나타내고 λ_i 는 정규화 계수(예를 들어, 패널티의 심각도를 결정함)를 나타낸다.

[0112] [00117] 위의 예에서, c2는: u, v는 다른 클러스터들에 속하지만 런타임 호출 의존성을 가지며 $\omega_{2(u,v)}$ 는 정규화된 호출 빈도를 계산한다. 위의 예에서, 만일 두 개의 클래스들이 빈번한 함수 호출을 한다면, 그들의 표현은 근접해야 한다(예를 들어, 클러스터-간 호출 볼륨).

[0113] [00118] 도 3은 본 명세서에 기술된 하나 혹은 그 이상의 실시예들에 따른 런타임 추적들에 기초하여 모놀리식 애플리케이션으로부터 마이크로서비스들의 생성을 용이하게 할 수 있는 예시적인, 비-제한적인 컴퓨터 구현-방법(300)의 플로우 다이어그램을 도시한다. 각 실시예들에 채용된 유사한 요소들 및/또는 프로세스들에 대한 반복적인 설명은 간결함을 위해 생략한다.

[0114] [00119] 단계(302)에서, 컴퓨터-구현 방법(300)은 입력 데이터를 수집하는 단계(예를 들어, 마이크로서비스 생성 시스템 102 및/또는 수집 컴포넌트 202를 통해)를 포함할 수 있다. 예를 들어, 도 2를 참조하여 위에서 설명된 바와 같이, 수집 컴포넌트(202)는 모놀리식 애플리케이션을 사용하여 테스트 케이스들(예를 들어, 비즈니스 기능 테스트 케이스들)을 실행함으로써(예를 들어, 프로세서 106를 통해) 생성될 수 있는 런타임 추적들을 수집할 수 있다.

[0115] [00120] 일 예에서, 도 3의 단계(302)에서 도시된 바와 같이, 수집 컴포넌트(202)는 모놀리식 애플리케이션의 계층 소스 코드로부터 런타임 로그(runtime log from instrumented source)를 생성할 수 있는 모니터링 애플리케이션을 채용할 수 있다. 이 예에서, 도 3의 단계(302)에서 도시된 바와 같이, 그러한 모니터링 애플리케이션은 로그 파일 및/또는 텍스트 파일로 포맷된 런타임 로그를 생성할 수 있다.

[0116] [00121] 다른 예에서, 도 3의 단계(302)에서 도시된 바와 같이, 수집 컴포넌트(202)는, 예를 들어, 런타임 컨텍스트(예를 들어, 비즈니스 컨텍스트)를 생성할 수 있는 자바 프론트-엔드 도구와 같은 런타임 컨텍스트 생성기(runtime context generator)를 채용할 수 있다. 이 예에서, 도 3의 단계(302)에서 도시된 바와 같이, 그러한 런타임 컨텍스트 생성기는 자바스크립트 객체 표기법(JSON) 파일 및/또는 쉼표로-구분된 값들(CSV) 파일(a JavaScript object notation (JSON) file and/or a comma-separated values (CSV) file)로서 형식이 지정된 컨텍스트 라벨들(예: 비즈니스 컨텍스트 라벨들)을 포함하는 런타임 컨텍스트를 생성할 수 있다. 이 예에서는, 마이크로서비스 생성 시스템(102)을 구현하는 주체(예를 들어, 인간, 클라이언트, 사용자, 컴퓨팅 디바이스, 소프트웨어 애플리케이션, 에이전트, 머신 러닝(ML) 모델, 인공 지능(AI) 모델, 등)는 마이크로서비스 생성 시스템(102)의 인터페이스 컴포넌트(도면에 도시되지 않음) (예를 들어, API(응용 프로그래밍 인터페이스), 표현 상태 전송 API, GUI(그래픽 사용자 인터페이스), 등)를 사용하여 그러한 컨텍스트 라벨들(예를 들어, 비즈니스 컨텍스트 레이블)을 정의할 수 있다.

[0117] [00122] 다른 예에서, 도 3의 단계(302)에서 도시된 바와 같이, 수집 컴포넌트(202)는 데이터 의존성 그래프를 생성하는 데 사용될 수 있는 모놀리식 애플리케이션의 소스 코드로부터의 정보를 추출하기 위해, 예를 들어, 파이썬-기반 도구와 같은 데이터 의존성 생성기(data dependency generator)를 채용할 수 있으며, 그러한 정보는 클래스 이름, 속성들, 방법 이름들, 방법 인자들, 반환 유형들, 및/또는 기타 정보를 포함할 수 있다. 이 예에서는, 도 3의 단계(302)에서 도시된 바와 같이, 그러한 데이터 의존성 생성기는 JSON 파일로 형식이 지정될 수 있는 심볼 테이블(도 3에서 symTable로 표시됨) 및/또는 참조 테이블(도 3에서 refTable로 표시됨)을 포함하는 데이터 의존성 그래프를 생성할 수 있다.

[0118] [00123] 다른 예에서, 도 3의 단계(302)에서 도시된 바와 같이, 수집 컴포넌트(202)는 정적 호출 그래프를 생성하기 위해, 예를 들어, 상속 관계들(inheritance relationships), 데이터 의존성, 속성들, 방법 인자, 반환 유형, 및/또는 기타 관계들을 추출할 수 있는 추출 애플리케이션과 같은 정적 호출 그래프 생성기(static call graph generator)를 채용할 수 있다. 이 예에서, 도 3의 단계(302)에서 도시된 바와 같이, 그러한 정적 호출 그래프 생성기는 JSON 파일로 형식이 지정된 정적 호출 그래프를 생성할 수 있다. 이 예에서, 도 3의 단계(302)에서 도시된 바와 같이, 마이크로서비스 생성 시스템(102) 및/또는 수집 컴포넌트(202)는 정적 호출 그래프에서

비-애플리케이션 클래스들을 필터링할 수 있으며, 그러한 필터링된 정적 호출 그래프는 클래스들의 클러스터들을 개량하기 위해 이용될 수 있고 및/또는 아래에 설명하는 바와 같이 사용자 인터페이스(UI)에 제공될 수 있다.

[0119] [00124] 단계(304)에서, 컴퓨터-구현 방법(300)은 (예를 들어, 마이크로서비스 생성 시스템(102) 및/또는 제2 모델 컴포넌트(204)를 통해) 런타임 추적들을 생성하는 단계를 포함할 수 있다. 예를 들어, 도 3의 단계(304)에서 도시된 바와 같이, 위에 설명된 단계(302)에서의 계층 소스로부터 런타임 로그의 수집 및/또는 런타임 컨텍스트(예를 들어, 비즈니스 컨텍스트 및/또는 비즈니스 컨텍스트 라벨들)의 생성에 기초하여, 단계(304)에서, 마이크로서비스 생성 시스템(102) 및/또는 제2 모델 컴포넌트(204)는 흐름들(예를 들어, 인과 시퀀스들, 데이터 흐름들 등)을 생성하고, 모놀리식 애플리케이션을 사용하여 위에서 설명된 테스트 케이스들(예를 들어, 비즈니스 기능 테스트 케이스들)을 실행함으로써 생성된 각각의 추적들에 컨텍스트 라벨들을 추가함으로써, 이에 의해 JSON 파일로 형식이 지정될 수 있는 런타임 추적들을 생성할 수 있다.

[0120] [00125] 단계(306)에서, 컴퓨터-구현 방법(300)은 도 3에 도시된 바와 같이 클러스터링 A, B, 및/또는 C를 생성하기 위해 모놀리식 애플리케이션의 클래스들을 클러스터링하는 단계(예를 들어, 마이크로서비스 생성 시스템 102, 모델 컴포넌트 108, 클러스터 컴포넌트 110 및/또는 제2 모델 컴포넌트 204를 통해)를 포함할 수 있다. 예를 들어, 단계(306)에서 클래스들의 그러한 클러스터링을 용이하게 하기 위해, 컴퓨터-구현 방법(300)은 도 3에 도시된 바와 같이 클러스터링 A, B, 및/또는 C를 생성하기 위해 클러스터들을 모놀리식 애플리케이션의 클래스들에 할당하는 단계(예를 들어, 마이크로서비스 생성 시스템 102, 모델 컴포넌트 108, 클러스터 컴포넌트 110, 및/또는 제2 모델 컴포넌트 204를 통해)를 포함할 수 있다. 예를 들어, 도 3의 단계(306)에서 도시된 바와 같이, 컴퓨터-구현 방법(300)은: a) 시간 의존성을 획득하기 위해 인과 그래프들(causal graphs)을 생성하는 단계 및/또는 적용하는 단계(예를 들어, 마이크로서비스 생성 시스템 102 및/또는 제2 모델 컴포넌트 204를 통해), 클러스터링 A를 산출하는 단계; b) 인과 그래프들을 사용하여 각 클래스의 파티션을 획득하기 위해 모놀리식 애플리케이션의 클러스터 할당들 및/또는 그래프 임베딩들(graph embeddings)을 학습하기 위한 모델 컴포넌트(108)를 훈련하도록 인과 그래프들을 생성하는 단계 및/또는 적용하는 단계(예를 들어, 마이크로서비스 생성 시스템 102, 모델 컴포넌트 108 및/또는 제2 모델 컴포넌트 204를 통해), 이에 의한 클러스터링 B를 산출하는 단계; 및/또는 c) 런타임 추적들을 사용하여 각 클래스의 파티션을 획득하기 위해 모놀리식 애플리케이션의 클러스터 할당들 및/또는 그래프 임베딩들을 학습하기 위한(예를 들어, 도 1 및 도 2를 참조하여 기술한 바와 같은) 모델 컴포넌트(108)를 훈련하도록 기술한 바와 같이 단계(304)에서 생성된 런타임 추적들을 적용하는 단계(예를 들어, 마이크로서비스 생성 시스템 102, 모델 컴포넌트 108 및/또는 클러스터 컴포넌트 110를 통해), 이에 의해 클러스터링 C를 생성하는 단계를 포함할 수 있다.

[0121] [00126] 단계(308)에서, 컴퓨터-구현 방법(300)은 위에서 설명된 바와 같이 단계(302)에서 생성된 데이터 의존성 그래프에 기초하여 그리고 하나의 주체(예를 들어, 인간, 클라이언트, 사용자, 컴퓨팅 디바이스, 소프트웨어 애플리케이션, 에이전트, 머신 러닝(ML) 모델, 인공 지능(AI) 모델, 등)로부터의 입력을 사용하여 도 3에 도시된 바와 같이 클러스터링 A, B, 및/또는 C를 개량하는 단계(예를 들어, 마이크로서비스 생성 시스템 102 및/또는 개량 컴포넌트 206를 통해)를 포함할 수 있다. 예를 들어, 도 3의 단계(308)에서 도시된 바와 같이, 개량 컴포넌트(206) 및/또는 위에서 정의된 상기 주체는 런타임 추적들에서 누락된 클래스들을 추가할 수 있으며(예를 들어, 클러스터링 A, B 및/또는 C에), 그러한 클래스들은 데이터 의존성(예를 들어, 데이터 의존성 그래프를 사용하여 결정된 클러스터링 A, B, 및/또는 C에서의 하나 혹은 그 이상의 클래스들에서의 데이터 의존성)을 갖는다. 이 예에서는, 도 3의 단계(308)에서 도시된 바와 같이, 개량 컴포넌트(206) 및/또는 위에서 정의된 상기 주체는 클러스터들(예를 들어, 클러스터링 A, B, 및/또는 C)을 클러스터-간 데이터 의존성(cross-cluster data dependency)(예를 들어, 클러스터링 A, B, 및/또는 C의 클래스들-간의 클러스터 간 데이터 의존성)과 더 병합할 수 있다. 이 예에서, 도 3의 단계(308)에서 도시된 바와 같이, 위에서 설명된 그러한 개량 동작들(refinement operations)은 자연스러운 줄기(natural seam)를 생성하도록 구현될 수 있다.

[0122] [00127] 단계(310)에서, 컴퓨터-구현 방법(300)은 위에서 설명된 바와 같이 단계(302)에서 생성된 정적 호출 그래프(예를 들어, 필터링된 정적 호출 그래프)에 기초하여 그리고 하나의 주체(예를 들어, 인간, 클라이언트, 사용자, 컴퓨팅 디바이스, 소프트웨어 애플리케이션, 에이전트, 머신 러닝(ML) 모델, 인공 지능(AI) 모델, 등)로부터의 입력을 사용하여 도 3에 도시된 바와 같이 클러스터링 A, B, 및/또는 C를 개량하는 단계(예를 들어, 마이크로서비스 생성 시스템 102 및/또는 개량 컴포넌트 206를 통해)를 포함할 수 있다. 예를 들어, 도 3의 단계(308)에서 도시된 바와 같이, 개량 컴포넌트(206) 및/또는 위에서 정의된 상기 주체는 런타임 추적들에서 누락된 클래스들을 추가할 수 있으며(예를 들어, 클러스터링 A, B 및/또는 C에), 그러한 클래스들은 정적 호출 의존

성(예를 들어, 정적 호출 그래프를 사용하여 결정된 클러스터링 A, B, 및/또는 C에서의 하나 혹은 그 이상의 클래스들에서의 정적 호출 의존성)을 갖는다. 이 예에서는, 도 3의 단계(310)에 도시된 바와 같이, 개량 컴포넌트(206) 및/또는 위에서 정의된 상기 주체는 클러스터에 대해 호출 의존성이 없는(예를 들어, 정적 호출 그래프를 사용하여 결정된 클러스터링 A, B, 및/또는 C에서의 하나 혹은 그 이상의 클래스들에서의 정적 호출 의존성이 없는) 클래스들(예를 들어, 클러스터링 A, B, 및/또는 C에 대해)을 더 추가할 수 있다. 이 예에서는, 도 3의 단계(310)에 도시된 바와 같이, 위에서 설명된 그러한 개량 동작들은 비즈니스 로직을 생성하기 위해 구현될 수 있다.

[0123] [00128] 일 예에서, 도 3에 도시된 바와 같이, 위에서 설명된 바와 같이 단계(302)에서 생성된 데이터 의존성 그래프 및/또는 정적 호출 그래프(예를 들어, 필터링된 정적 호출 그래프)는 JSON 파일로서 사용자 인터페이스(UI)에 제공될 수 있다. 이 예에서, 도 3에 도시된 바와 같이, 위에서 설명된 바와 같이 단계(304)에서 생성된 런타임 추적들은 JSON 파일로 포맷될 수 있는 런타임 그래프의 형태로 사용자 인터페이스(UI)에 제공될 수 있다. 일 예에서, 도 3에 도시된 바와 같이, 위에서 설명된 바와 같이, 단계(308) 및 단계(310)에서, 각각, 생성된 자연스러운 술기 및/또는 비즈니스 로직은, JSON 파일로서 사용자 인터페이스(UI)에 제공될 수 있다.

[0124] [00129] 도 4는 일 예의 비-제한적인 모델(400)의 다이어그램을 도시하고, 비-제한적인 모델(400)은 본 명세서에 기술된 하나 혹은 그 이상의 실시예들에 따른 런타임 추적들에 기초하여 모놀리식 애플리케이션으로부터 마이크로서비스의 생성을 용이하게 할 수 있다. 각 실시 예들에 채용된 유사한 요소들 및/또는 프로세스들에 대한 반복적인 설명은 간결함을 위해 생략한다.

[0125] [00130] 일부 실시 예들에서, 모델 컴포넌트(108) 및/또는 제2 모델 컴포넌트(204)는 모델(400)을 포함할 수 있다. 예를 들어, 도 1을 참조하여 위에서 설명된 바와 같이, 모델 컴포넌트(108)는 모델(400)을 포함할 수 있고, 이는 위에서 설명된 하나 혹은 그 이상의 인과 그래프들의 인과 시퀀스들을 사용하여 위에서 설명된 클러스터 할당들 및/또는 그래프 임베딩들을 학습하도록 훈련될 수 있는(예를 들어, 전술한 마이크로서비스 생성 시스템 102을 통해) 신경망을 포함할 수 있다.

[0126] [00131] 도 4에 도시된 바와 같이, 모델(400)은 입력 계층(402), 임베딩 계층(404), 및/또는 출력 계층(406)을 포함할 수 있다. 도 1을 참조하여 위에서 설명된 예시적인 거래 애플리케이션 실시예에서, 입력 계층(402)은 도 4에 도시된 바와 같은 입력 벡터의 형태로 모놀리식 애플리케이션의 자바 클래스(도 4에서 "MarketSummarySingleton"으로 표시됨)를 임베딩 계층(404)에 제공할 수 있다. 임베딩 계층(404)은 입력 자바 클래스의 하나 혹은 그 이상의 임베딩들을 생성할 수 있으며, 그러한 하나 혹은 그 이상의 임베딩들은 입력 자바 클래스의 하나 혹은 그 이상의 벡터 표현을 포함할 수 있다. 입력 자바 클래스의 그러한 벡터 표현(들)에 기초하여, 도 4에 예시된 바와 같은 소프트맥스 분류기(a softmax classifier)를 포함할 수 있는, 출력 계층(406)은 모놀리식 애플리케이션의 다른 클래스가 미리 정의된 창(예를 들어, 도 1을 참조하여 전술한 예시적인 거래 애플리케이션 실시예에 표시된 3의 창 크기)에 있는지를 예측할 수 있다. 예를 들어, 출력 계층(406)은 모놀리식 애플리케이션에서의 한 쌍의 자바 클래스들이 동일한 창(도 4에서 "이웃"으로 표시됨)에 속하는지를 예측할 수 있다. 예를 들어, 도 1을 참조하여 위에서 설명된 예시적인 거래 애플리케이션 실시예에서, 도 4에 도시된 바와 같이, 임베딩 계층(404)이 도 4에서 "MarketSummarySingleton"으로 표시된 입력 자바 클래스의 벡터 표현을 생성하는 것에 기초하여, 출력 계층(406)은: a) 클래스 "Log"가 이웃에(예를 들어, 입력 자바 클래스 "MarketSummarySingleton"에 대해 창 크기 3 이내) 있을 확률; b) 클래스 "TradeConfig"가 이웃에(예를 들어, 입력 자바 클래스 "MarketSummarySingleton"에 대해 창 크기 3 이내) 있을 확률; c) 클래스 "QuoteDataBeam"가 이웃에(예를 들어, 입력 자바 클래스 "MarketSummarySingleton"에 대해 창 크기 3 이내) 있을 확률; 및/또는 d) 클래스 "TradeSLSBBeam"가 이웃에(예를 들어, 입력 자바 클래스 "MarketSummarySingleton"에 대해 창 크기 3 이내) 있을 확률을 예측할 수 있다.

[0127] [00132] 마이크로서비스 생성 시스템(102)은 다양한 기술들과 연관될 수 있다. 예를 들어, 마이크로서비스 생성 시스템(102)은 애플리케이션 현대화 기술, 모놀리식 애플리케이션 기술들, 모놀리식 엔터프라이즈 애플리케이션 기술들, 애플리케이션 프로그래밍 기술들, 클라우드 컴퓨팅 기술들, 머신 러닝 기술들, 인공 지능 기술들, 및/또는 기타 기술들과 연관될 수 있다.

[0128] [00133] 마이크로서비스 생성 시스템(102)은 위에서 식별된 다양한 기술들과 관련된 시스템들, 디바이스들, 컴포넌트들, 동작 단계들, 및/또는 처리 단계들에 대한 기술적 개선들을 제공할 수 있다. 예를 들어, 마이크로서비스 생성 시스템(102)은 모놀리식 애플리케이션을 사용하여 실행될 수 있는 비즈니스 기능 테스트 케이스들의 런타임 추적들에 기초하여 모놀리식 애플리케이션에서 클러스터 할당들 및/또는 클래스들의 그래프 임베딩들을

학습하도록 모델을 훈련할 수 있고; 및/또는 모놀리식 애플리케이션의 하나 혹은 그 이상의 마이크로서비스들을 식별하기 위해 클러스터 할당들 및/또는 그래프 임베딩들에 기초하여 클래스들의 클러스터들을 생성하도록 모델을 더 채용할 수 있다. 모놀리식 애플리케이션을 사용하여 그러한 비즈니스 기능 테스트 케이스들을 실행하면 그러한 비즈니스 기능 테스트 케이스들 각각에 대응하는 런타임 추적들을 생성할 수 있다. 마이크로서비스 생성 시스템(102)은 그러한 각각의 비즈니스 기능 테스트 케이스들에 또한 대응하는 인과 시퀀스들을 포함하는 인과 그래프들을 생성하도록 각각의 비즈니스 기능 테스트 케이스들에 대응하는 런타임 추적들을 이용할 수 있다. 각각의 비즈니스 기능 테스트 케이스들에 대응하는 그러한 인과 그래프들 및/또는 인과 시퀀스들은 마이크로서비스 생성 시스템(102)에 모놀리식 애플리케이션의 1차 시간적 의존성들 및/또는 고차 시간 의존성들을 제공하고 및/또는 모놀리식 애플리케이션에서 클래스들의 클러스터 할당들 제공하며, 그러한 의존성들은 각각의 비즈니스 기능 테스트 케이스들에 대응한다. 이들 1차 시간 의존성들 및/또는 고차 시간 의존성들은 마이크로서비스 생성 시스템(102)에, 예를 들어, 모놀리식 애플리케이션의 클래스들과 기능 호출들이 상호 작용하는 방식의 순서와 빈도와 같은, 모놀리식 애플리케이션의 다양한 비즈니스 기능들을 제공한다. 1차 시간 의존성들 및/또는 고차 시간 의존성들은 마이크로서비스 생성 시스템(102)이 비즈니스 컨텍스트를 각 추적들과 연관시켜 모놀리식 애플리케이션에서 클래스들의 향상된 기능적 그룹화(예를 들어, 기능적 클러스터링)를 제공하는 것을 추가로 가능하게 할 수 있다. 모놀리식 애플리케이션에서 클래스들의 그러한 개선된 기능적 그룹화는 마이크로서비스 생성 시스템(102)이 모놀리식 애플리케이션의 하나 혹은 그 이상의 마이크로서비스들을 보다 정확하게 식별할 수 있게 할 수 있고, 이에 의해, 마이크로서비스 생성 시스템(102)에 의한 모놀리식 애플리케이션의 개선된 애플리케이션 현대화를 촉진할 수 있다.

[0129] [00134] 마이크로서비스 생성 시스템(102)은 마이크로서비스 생성 시스템(102)과 연관된 고전적 컴퓨팅 디바이스 및/또는 양자 컴퓨팅 디바이스(예를 들어, 양자 프로세서, 양자 하드웨어, 초전도 회로, 등)와 연관된 처리 유닛(예를 들어, 프로세서 106)에 기술적 개선들을 제공할 수 있다. 예를 들어, 마이크로서비스 생성 시스템(102)이 위에서 설명된 바와 같이 모놀리식 애플리케이션의 하나 혹은 그 이상의 마이크로서비스들을 보다 정확하게 식별할 수 있게 하는 모놀리식 애플리케이션에서 클래스들의 기능적 그룹화(예를 들어, 기능적 클러스터링)를 개선함으로써, 마이크로서비스 생성 시스템(102)은 모놀리식 애플리케이션의 애플리케이션 현대화를 실행하는 데 이용된 프로세서(예를 들어, 프로세서 106)의 워크로드를 감소시킬 수 있다(예를 들어, 프로세서가 애플리케이션 현대화 프로세스를 완료하기 위해 수행하는 처리 주기 수들을 줄임으로써). 그러한 프로세서(예를 들어, 프로세서 106)의 그러한 감소된 워크로드는 그러한 프로세서의 처리 성능 및/또는 처리 효율성을 개선할 수 있고 및/또는 그러한 프로세서의 계산 비용들을 추가로 감소시킬 수 있다.

[0130] [00135] 마이크로서비스 생성 시스템(102)의 실제 애플리케이션은 모놀리식 애플리케이션에서 개량된 애플리케이션 현대화 프로세스를 수행하는 것을 구현할 수 있다. 예를 들어, 마이크로서비스 생성 시스템(102)의 실제 애플리케이션은 모놀리식 애플리케이션의 마이크로서비스를 빠르고 정확하게 식별하기 위해 모놀리식 애플리케이션(예를 들어, 모놀리식 엔터프라이즈 애플리케이션)의 소유자에 의해 구현될 수 있고, 이에 의해, 모놀리식 애플리케이션에서 개량된 애플리케이션 현대화 프로세스를 가능하게 할 수 있다.

[0131] [00136] 마이크로서비스 생성 시스템(102)은 현재 모놀리식 애플리케이션과 관련된 특정 설계 또는 프로그래밍 모델들의 가정을 포함하고 및/또는 모놀리식 애플리케이션에서 애플리케이션 현대화 프로세스를 수행하기 위해 인간으로부터 수동-집약적인 입력(manually-intensive input)을 포함하는 비교적 새로운 애플리케이션 현대화 기술들에 의해 구동되는 새로운 접근 방식을 제공한다는 것을 인식(appreciate)해야 한다. 예를 들어, 마이크로서비스 생성 시스템(102)은 다음을 수행하기 위해 인간으로부터 수동-집약적인 입력 없이 모놀리식 애플리케이션에서 애플리케이션 현대화를 수행하도록 새로운 자동화된 접근 방식을 제공한다: 모놀리식 애플리케이션과 관련된 특정 설계 또는 프로그래밍 모델들의 그러한 가정들을 정의함; 모놀리식 애플리케이션의 잠재적인 마이크로서비스 후보들을 구성할 수 있는 모놀리식 애플리케이션의 클래스들 및/또는 클래스들의 클러스터들을 식별함; 및/또는 모놀리식 애플리케이션의 독립형 애플리케이션들을 포함할 수 있는 및/또는 모놀리식 애플리케이션의 독립형 애플리케이션들로 구현될 수 있는 모놀리식 애플리케이션의 마이크로서비스들을 식별하기 위해 그러한 잠재적인 마이크로서비스 후보들을 개량함.

[0132] [00137] 마이크로서비스 생성 시스템(102)은, 추상적이지 않고 인간에 의한 일련의 정신적 행위들로 수행될 수 없는, 본질적으로 고도로 기술적인 문제들을 해결하기 위해 하드웨어 또는 소프트웨어를 사용할 수 있다. 일부 실시 예들에서, 여기에 설명된 프로세스들 중 하나 혹은 그 이상은 위에서 식별된 다양한 기술들과 관련된 정의된 작업들을 실행하기 위해 하나 혹은 그 이상의 특수 컴퓨터들(예를 들어, 특수 처리 장치, 특수 클래식 컴퓨터, 특수 양자 컴퓨터, 등.)에 의해 수행될 수 있다. 마이크로서비스 생성 시스템(102) 및/또는 그와 관련된 컴

포넌트들은, 위에서 언급한 기술들의 발전들, 양자 컴퓨팅 시스템들, 클라우드 컴퓨팅 시스템들, 컴퓨터 아키텍처들, 및/또는 다른 기술들의 사용을 통해 발생하는 새로운 문제들을 해결하는 데 사용될 수 있다.

- [0133] [00138] 여기에 설명된 마이크로서비스 생성 시스템(102) 및/또는 그와 관련된 컴포넌트들에 의해 실행될 수 있는 다양한 동작들은 인간 정신의 능력보다 더 큰 동작들이기 때문에, 마이크로서비스 생성 시스템(102)은 인간의 마음으로 복제할 수 없거나 인간이 수행할 수 없는 전기 컴포넌트들, 기계적 컴포넌트들, 및 회로의 다양한 조합들을 이용할 수 있음을 이해해야 한다. 예를 들어, 특정 기간 동안 마이크로서비스 생성 시스템(102)에 의해 처리된 데이터의 양, 그러한 데이터를 처리하는 속도, 또는 처리된 데이터 유형들은 동일한 기간 동안 인간의 마음에 의해 처리될 수 있는 양, 속도, 또는 데이터 유형보다 더 크거나, 더 빠르거나, 또는 다양할 수 있다.
- [0134] [00139] 몇몇 실시예들에 따르면, 마이크로서비스 생성 시스템(102)은 또한 본 명세서에 설명된 다양한 동작들을 수행하면서 하나 혹은 그 이상의 다른 기능들(예를 들어, 완전히 전원이 켜진 상태, 완전히 실행되는 것, 등)을 수행하도록 완전히 동작할 수 있다. 이러한 동시 다중-동작 실행은 인간 정신의 능력을 넘어서는 것임을 인식해야 한다. 또한 마이크로서비스 생성 시스템(102)은, 인간 사용자와 같은, 주체에 의해 수동으로 획득할 수 없는 정보를 포함할 수 있음을 이해해야 한다. 예를 들어, 마이크로서비스 생성 시스템(102), 모델 컴포넌트(108), 클러스터 컴포넌트(110), 수집 컴포넌트(202), 제2 모델 컴포넌트(204), 개량 컴포넌트(206), 및/또는 모델(400)에 포함된 정보의 유형, 양 및/또는 다양성은 인간 사용자에게 의해 수동으로 획득된 정보보다 더 복잡할 수 있다.
- [0135] [00140] 도 5는 본 명세서에 기술된 하나 혹은 그 이상의 실시 예들에 따른 런타임 추적들에 기초하여 모놀리식 애플리케이션으로부터 마이크로서비스들의 생성을 용이하게 할 수 있는 예시적인, 비-제한적인 컴퓨터 구현 방법(500)의 플로우 다이어그램을 도시한다. 각 실시 예들에 채용된 유사한 요소들 및/또는 프로세스들에 대한 반복적인 설명은 간결함을 위해 생략한다.
- [0136] [00141] 단계(502)에서, 컴퓨터-구현 방법(500)은 실행된 테스트 케이스들(예: 모놀리식 애플리케이션을 사용하여 실행된 비즈니스 기능 테스트 케이스들)의 런타임 추적들(예를 들어, 도 1 내지 도 3을 참조하여 위에서 설명된 런타임 추적들)에 기초하여 모놀리식 애플리케이션(예: 모놀리식 엔터프라이즈 애플리케이션)에서 클래스들의 클러스터 할당들(예: 클러스터 할당들 및/또는 자바 클래스들의 그래프 임베딩들)을 학습하도록 모델(예: 모델 컴포넌트 108, 모델 400, 등)을, 프로세서(예를 들어, 프로세서(106), 양자 프로세서, 등)에 작동 가능하게 연결된 시스템(예를 들어, 마이크로서비스 생성 시스템(102)을 통해)에 의해, 훈련하는 단계(training)를 포함할 수 있다.
- [0137] [00142] 단계(504)에서, 컴퓨터-구현 방법(500)은, 모놀리식 애플리케이션의 하나 혹은 그 이상의 마이크로서비스들을 식별하기 위해 클러스터 할당들에 기초하여 클래스들의 클러스터들을 생성하도록 상기 모델을, 상기 시스템에 의해(예를 들어, 마이크로서비스 생성 시스템(102) 및/또는 클러스터 컴포넌트(110)를 통해), 채용하는 단계(employing)를 포함할 수 있다.
- [0138] [00143] 도 6은, 본 명세서에 기술된 하나 혹은 그 이상의 실시 예들에 따른 런타임 추적들에 기초하여 모놀리식 애플리케이션으로부터 마이크로서비스들의 생성을 용이하게 할 수 있는 비-제한적인 컴퓨터 구현 방법(600)의, 예시의 플로우 다이어그램을 도시한다. 각 실시 예들에 채용된 유사한 요소들 및/또는 프로세스들에 대한 반복적인 설명은 간결함을 위해 생략한다.
- [0139] [00144] 단계(602)에서, 컴퓨터-구현 방법(600)은, 모놀리식 애플리케이션에서 실행된 테스트 케이스들(예를 들어, 모놀리식 엔터프라이즈 애플리케이션과 같은, 모놀리식 애플리케이션을 사용하여 실행된 비즈니스 기능 테스트 케이스들)의 런타임 추적들(예를 들어, 도 1 내지 도 3을 참조하여 위에서 설명된 런타임 추적들)을, 프로세서(예를 들어, 프로세서(106), 양자 프로세서, 등)에 작동 가능하게 연결된 시스템에 의해(예를 들어, 마이크로서비스 생성 시스템(102) 및/또는 수집 컴포넌트(202)를 통해), 수집하는 단계를 포함할 수 있다.
- [0140] [00145] 단계(604)에서, 컴퓨터-구현 방법(600) 런타임 추적에 기초하여 모놀리식 애플리케이션에서 클래스들의 클러스터 할당들(예를 들어, 클러스터 할당들 및/또는 자바 클래스들의 그래프 임베딩들)을 학습하도록 모델(예를 들어, 모델 컴포넌트 108, 모델 400, 등)을, 상기 시스템에 의해(예를 들어, 마이크로서비스 생성 시스템 102를 통해), 훈련하는 단계를 포함할 수 있다.
- [0141] [00146] 단계(606)에서, 컴퓨터-구현 방법(600)은 모놀리식 애플리케이션의 하나 혹은 그 이상의 마이크로서비스들을 식별하기 위해 클러스터 할당들에 기초하여 클래스들의 클러스터들을 생성하도록 상기 모델을, 상기 시

시스템에 의해(예를 들어, 마이크로서비스 생성 시스템 102 및/또는 클러스터 컴포넌트 110를 통해), 채용하는 단계를 포함한다.

- [0142] [00147] 도 7은, 본 명세서에 기술된 하나 혹은 그 이상의 실시 예들에 따른 런타임 추적들에 기초하여 모놀리식 애플리케이션으로부터 마이크로서비스들의 생성을 용이하게 할 수 있는 비-제한적인 컴퓨터-구현 방법(700)의 예시적 플로우 다이어그램을 도시한다. 각 실시 예들에 채용된 유사한 요소들 및/또는 프로세스들에 대한 반복적인 설명은 간결함을 위해 생략한다.
- [0143] [00148] 단계(702)에서, 컴퓨터-구현 방법(700)은 모놀리식 애플리케이션을 사용하여 실행된 비즈니스 기능 테스트 케이스들의 런타임 추적들(예를 들어, 도 1 내지 도 3을 참조하여 위에서 설명된 런타임 추적들)에 기초하여 모놀리식 애플리케이션(예: 모놀리식 엔터프라이즈 애플리케이션)에서 클러스터 할당들 및/또는 자바 클래스들의 그래프 임베딩들을 학습하도록 모델(예: 모델 컴포넌트 108, 모델 400, 등)을 훈련하는 단계(예를 들어, 마이크로서비스 생성 시스템(102)을 통해)를 포함할 수 있다.
- [0144] [00149] 단계(704)에서, 컴퓨터-구현 방법(700)은 클러스터 할당들 및/또는 그래프 임베딩들에 기초하여 자바 클래스들의 클러스터들을 생성하도록 상기 모델을 채용하는 단계(예를 들어, 마이크로서비스 생성 시스템(102) 및/또는 클러스터 컴포넌트(110)를 통해)를 포함할 수 있다.
- [0145] [00150] 단계(706)에서, 컴퓨터-구현 방법(700)은 모놀리식 애플리케이션의 하나 혹은 그 이상의 잠재적인 마이크로서비스 후보들이 식별되는지를 결정하는 단계(예를 들어, 마이크로서비스 생성 시스템(102), 모델 컴포넌트(108) 및/또는 클러스터 컴포넌트(110)를 통해)를 포함할 수 있다. 예를 들어, 도 1 및 전술한 예시적 거래 애플리케이션을 참조하여, 클러스터 컴포넌트(110)는 인과 시퀀스들로부터 하나 혹은 그 이상의 양의 샘플 쌍들(positive sample pairs)을 생성하도록 모델 컴포넌트(108)의 훈련된 버전을 채용할 수 있고, 그러한 하나 혹은 그 이상의 양의 샘플 쌍들은 모놀리식 애플리케이션의 하나 혹은 그 이상의 잠재적인 마이크로서비스 후보들을 구성할 수 있다.
- [0146] [00151] 만일 단계(706)에서 모놀리식 애플리케이션의 하나 혹은 그 이상의 잠재적인 마이크로서비스 후보들이 식별되는 것으로 결정되면, 단계(708)에서, 컴퓨터-구현 방법(700)은 모놀리식 애플리케이션의 마이크로서비스를 식별하기 위해 모놀리식 애플리케이션의 데이터 의존성 그래프 및/또는 정적 호출 그래프(예를 들어, 도 1, 2 및/또는 3을 참조하여 위에서 설명된 데이터 의존성 그래프 및/또는 정적 호출 그래프)에 기초하여 하나 혹은 그 이상의 잠재적인 마이크로서비스 후보들을 개량하는 단계(예를 들어, 개량 컴포넌트(206)를 통해)를 포함할 수 있다.
- [0147] [00152] 만일 단계(706)에서 모놀리식 애플리케이션의 하나 혹은 그 이상의 잠재적인 마이크로서비스 후보들이 식별되지 않는 것으로 결정되면, 단계(710)에서, 컴퓨터-구현 방법(700)은 상기 모델에 의해 사용되는 윈도우 크기 (예를 들어, 모놀리식 애플리케이션의 하나 혹은 그 이상의 잠재적인 마이크로서비스 후보들을 구성할 수 있는 하나 혹은 그 이상의 양의 샘플 쌍들을 생성하는 데 사용될 수 있는 도 1을 참조하여 전술한 창 크기)를 수정하는 단계(예를 들어, 마이크로서비스 생성 시스템(102) 및/또는 모델 컴포넌트(108)를 통해) 및/또는 모놀리식 애플리케이션을 사용하여 다양한 비즈니스 기능 테스트 케이스들(예: 도 1을 참조하여 전술한 것과 다른 비즈니스 기능 테스트 케이스들)을 실행하는 단계(예를 들어, 마이크로서비스 생성 시스템(102) 및/또는 수집 컴포넌트(202)를 통해)를 포함할 수 있다. 예를 들어, 단계(710)에서, 컴퓨터-구현 방법(700)은 상기 모델에 의해 사용되는 창 크기를, 예를 들어, 3의 창 크기에서, 예를 들어, 4의 창 크기로 수정하는 단계를 포함할 수 있다.
- [0148] [00153] 단계(712)에서, 컴퓨터-구현 방법(700)은 단계(710)에서 실행된 다양한 비즈니스 기능 테스트 케이스들의 런타임 추적들 및/또는 상기 수정된 윈도우 크기에 기초하여 모놀리식 애플리케이션에서 클러스터 할당들 및/또는 자바 클래스들의 그래프 임베딩들을 학습하도록 상기 모델을 재-훈련하는 단계(예를 들어, 마이크로서비스 생성 시스템(102)을 통해)를 포함할 수 있다.
- [0149] [00154] 설명의 편의를 위해, 컴퓨터-구현 방법들은 일련의 행위들로 묘사되고 설명된다. 본 발명은 예시된 행위들 및/또는 행위들의 순서에 의해 제한되지 않음을 이해하고 인식해야 하는데, 예를 들어, 행위들은 다양한 순서들로 및/또는 동시에, 및 여기에 제시 및 설명되지 않은 다른 행위들과 함께 발생할 수 있다. 뿐만 아니라, 설명된 모든 행위들이 개시된 주제에 따라 컴퓨터-구현 방법들을 구현하기 위해 요구될 수 있는 것은 아니다. 게다가, 당업자는 컴퓨터-구현 방법들이 대안적으로 상태 다이어그램 또는 이벤트들을 통해 일련의 상호 관련된 상태로 표현될 수 있음을 이해하고 인식해야 할 것이다. 추가적으로, 이하 및 본 명세서 전반에 걸쳐 개시되는

컴퓨터-구현 방법들은 그러한 컴퓨터-구현 방법들을 컴퓨터들로 전송 및 전송하는 것을 용이하게 하기 위해 제조 물품에 저장될 수 있다는 것을 추가로 이해해야 한다. 본 명세서에 사용된 바와 같이, 제조 물품이라는 용어는, 임의의 컴퓨터-판독가능 디바이스 또는 저장 매체로부터 액세스가능한 컴퓨터 프로그램을 포함하도록 의도된다.

[0150] [00155] 본 발명의 개시된 주제의 다양한 실시 예들에 대한 컨텍스트를 제공하기 위해, 도 8 및 하기 논의는 본 발명의 개시된 주제의 다양한 실시 예들이 구현될 수 있는 적절한 환경의 일반적인 설명을 제공하도록 의도된다. 도 8은 본 명세서에 설명된 하나 또는 그 이상의 실시예들이 용이하게 구현될 수 있는 예시적인, 비제한적인 동작 환경의 블록도를 예시한다. 본 명세서에 설명된 다른 실시예들에서 채용된 유사한 요소들에 대한 반복적인 설명은 간결함을 위해 생략된다.

[0151] [00156] 도 8를 참조하여, 본 발명의 다양한 실시 예들을 구현하기 위한 적절한 동작 환경(800)은 또한 컴퓨터(812)를 포함할 수 있다. 컴퓨터(812)는 또한 처리 유닛(814), 시스템 메모리(816), 및 시스템 버스(818)를 포함할 수 있다. 시스템 버스(818)는 시스템 메모리(816)(이러한 예로 한정되는 것은 아님)를 포함하는 시스템 컴포넌트들을 처리 유닛(814)에 결합한다. 처리 유닛(814)은 다양한 이용가능한 프로세서들 중 어떤 것일 수 있다. 또한 듀얼 마이크로프로세서들 및 다른 멀티프로세서 아키텍처들이 처리 유닛(814)으로 이용될 수도 있다. 시스템 버스(818)는 메모리 버스 또는 메모리 컨트롤러를 포함하는 버스 구조(들), 주변 버스 또는 외부 버스, 및 또는 로컬 버스의 몇몇 유형들 중 어떤 것일 수 있고, 이는 산업 표준 아키텍처(Industrial Standard Architecture, ISA), 마이크로-채널 아키텍처(Micro-Channel Architecture, MSA), 확장 ISA(Extended ISA, EISA), 인텔리전트 드라이브 일렉트로닉스(Intelligent Drive Electronics, IDE), VESA 로컬 버스(VESA Local BUS, VLB), 주변 컴포넌트 인터랙트(Peripheral Component Interconnect, PCI), 카드 버스, 범용 직렬 버스(Universal Serial Bus, USB), 어드밴스드 그래픽스 포트(Advanced Graphics Port, AGP), 파이어와이어(Firewire, IEEE 1394), 및 소형 컴퓨터 시스템즈 인터페이스(Small Computer Systems Interface, SCSI)를 포함하는 이용가능한 다양한 버스 아키텍처들 중 어떤 것을 사용하는 것일 수 있으나 이러한 예로 한정되는 것은 아니다.

[0152] [00157] 시스템 메모리(816)는 또한 휘발성 메모리(volatile memory)(820) 및 비휘발성 메모리(non-volatile memory)(822)를 포함할 수 있다. 스타트-업(start-up) 동안과 같이 컴퓨터(812) 내에서 컴포넌트들 사이의 정보를 전송하기 위해 기본적인 루틴들을 포함하는, 기본 입력/출력 시스템(basic input/output system, BIOS)는 비휘발성 메모리(822)에 저장된다. 컴퓨터(812)는 또한 착탈식/고정식(removable/non-removable), 휘발성/비휘발성 컴퓨터 스토리지 매체를 포함할 수 있다. 예를 들어, 도 8은 디스크 스토리지(824)를 보여준다. 디스크 스토리지(824)는 또한 자기 디스크 드라이브(magnetic disc drive), 플로피 디스크 드라이브(floppy disk drive), 테이프 드라이브(tape drive), 제즈 드라이브(Jaz drive), 질 드라이브(Zip drive), LS-100 드라이브, 플래쉬 메모리 카드, 또는 메모리 스틱과 같은 디바이스들을 포함할 수 있으나, 이러한 예로 한정되는 것은 아니다. 디스크 스토리지(824)는 또한 다른 스토리지 매체와는 독립적인 또는 다른 스토리지 매체와 결합된 스토리지 매체를 포함할 수 있다. 시스템 버스(818)에 대한 디스크 스토리지(824)의 연결을 용이하게 하기 위해, 일반적으로 인터페이스(826)와 같은 착탈식 또는 고정식 인터페이스가 사용된다. 도 8는 또한 소프트웨어를 도시하는데, 소프트웨어는 적절한 동작 환경(800)에서 설명된 기본적인 컴퓨터 자원들과 사용자들 사이의 중개자(intermediary)로서 작용한다. 예를 들어, 이러한 소프트웨어는 또한 운영 체제(828)를 포함할 수 있다. 운영 체제(828)는 디스크 스토리지(824) 상에 저장될 수 있으며, 컴퓨터(812)의 자원들을 컨트롤하고 할당하도록 동작한다.

[0153] [00158] 시스템 어플리케이션들(830)은, 예를 들어, 시스템 메모리(816)와 디스크 스토리지(824) 둘 중 어느 하나에 저장된 프로그램 모듈들(832) 및 프로그램 데이터(834)를 통해 운영 체제(828)에 의한 자원들의 관리를 이용한다. 본 발명은 여러 가지 운영 체제들 또는 운영 체제들의 조합들로 구현될 수 있다는 것이 이해되어야 할 것이다. 사용자는 입력 디바이스(들)(836)을 통해 컴퓨터(812)에 명령들 또는 정보를 입력한다. 입력 디바이스들(836)은, 마우스와 같은 포인팅 디바이스, 트랙볼, 스타일러스, 터치패드, 키보드, 마이크로폰, 조이스틱, 게임패드, 위성 접시(satellite dish), 스캐너, TV 튜너 카드, 디지털 카메라, 디지털 비디오 카메라, 웹 카메라 등을 포함할 수 있으나, 이러한 예로 한정되는 것은 아니다. 이들 및 다른 입력 디바이스들은 인터페이스 포트(들)(838)을 경유하여 시스템 버스(818)를 통해 처리 유닛(814)에 연결된다. 인터페이스 포트(들)(838)은, 예를 들어, 직렬 포트, 병렬 포트, 게임 포트, 및 범용 직렬 버스(USB)를 포함할 수 있다. 출력 디바이스(들)(840)은 입력 디바이스(들)(836)와 동일한 유형의 포트들 중 일부를 사용한다. 따라서, 예를 들어, 컴퓨터(812)에 입력을 제공하기 위해, 그리고 컴퓨터(812)에서 출력 디바이스(840)로 정보를 출력하기 위해 USB 포트가 사용될 수

있다. 출력 어댑터(842)는 다른 출력 디바이스들(840) 중에서 특별한 어댑터들을 필요로 하는 모니터들, 스피커들, 및 프린터들과 같은 몇몇 출력 디바이스들(840)이 있음을 나타내기 위해 제공된다. 출력 어댑터들(842)은 출력 디바이스(840)와 시스템 버스(818) 사이의 연결 수단을 제공하는 비디오 카드 및 사운드 카드를 포함한다. 도 8에서는 예를 들기 위해 나타난 것이며 이러한 예로 한정되는 것은 아니다. 다른 디바이스들 및/또는 디바이스들의 시스템들은 원격 컴퓨터(들) (844)와 같은 입력 및 출력 기능들을 모두 제공한다는 것에 주목해야 한다.

[0154] [00159] 컴퓨터(812)는 원격 컴퓨터(들)(844)와 같은 하나 또는 그 이상의 원격 컴퓨터들에 대한 논리적 연결들을 사용하는 네트워킹된 환경에서 동작할 수 있다. 원격 컴퓨터(들)(844)은 컴퓨터, 서버, 라우터, 네트워크 PC, 워크스테이션, 마이크로프로세서 기반의 기기(microprocessor-based appliance), 피어 디바이스(peer device) 또는 다른 공용 네트워크 노드 등일 수 있고, 일반적으로는 컴퓨터(812)와 관련하여 설명된 컴포넌트들 중 많은 것 또는 전부를 포함할 수도 있다. 간략화를 위해, 단지 메모리 스토리지 디바이스(846) 만이 원격 컴퓨터(들)(844)와 함께 도시되어 있다. 원격 컴퓨터(들)(844)은 네트워크 인터페이스(848)를 통해 컴퓨터(812)에 논리적으로 연결되고, 그런 다음 통신 연결(850)을 통해 물리적으로 연결된다. 네트워크 인터페이스(848)는 근거리 통신망(local-area networks, LAN), 광역 통신망(wide-area networks, WAN), 셀룰러 통신망(cellular networks) 등과 같은 유선 및/또는 무선 통신망들을 포함한다. LAN 기술들은 광섬유 분산 데이터 인터페이스(Fiber Distributed Data Interface, FDDI), 동선 분산 데이터 인터페이스(Copper Distributed Data Interface, CDDI), 이더넷(Ethernet), 토큰 링(Token Ring) 등을 포함한다. WAN 기술들은, 포인트-투-포인트 링크들(point-to-point links), 집적형 서비스 디지털 네트워크(Integrated Services Digital Networks, ISDN)와 같은 회로 스위칭 네트워크 및 그것들의 변형들, 패킷 스위칭 네트워크들(packet switching networks), 및 디지털 구독자 회선들(Digital Subscriber Lines, DSL)을 포함하지만, 이러한 예들로 한정되는 것은 아니다. 통신 연결(들)(850)은 네트워크 인터페이스(848)를 시스템 버스(818)에 연결하기 위해 이용되는 하드웨어/소프트웨어를 지칭한다. 통신 연결(850)은 명확한 설명을 위해 컴퓨터(812) 내에 나타내었지만, 그것은 또한 컴퓨터(812)의 외부에 있을 수도 있다. 네트워크 인터페이스(848)에 대한 연결을 위한 하드웨어/소프트웨어는 또한, 단지 예시의 목적을 위해, 일반 전화 등급 모델들, 케이블 모델들 및 DSL 모델들을 포함하는 모델들, ISDN 어댑터들, 및 이더넷 카드들과 같은 내부 및 외부 기술들을 포함할 수 있다.

[0155] [00160] 이제 도 9를 참조하면, 예시적인 클라우드 컴퓨팅 환경(950)이 도시되어 있다. 보여지는 바와 같이, 클라우드 컴퓨팅 환경(950)은 하나 또는 그 이상의 클라우드 컴퓨팅 노드들(910)을 포함한다. 예를 들어, 개인용 디지털 보조기기(personal digital assistant, PDA) 또는 셀룰러 전화기(954A), 데스크탑 컴퓨터(954B), 랩탑 컴퓨터(954C), 및/또는 자동차의 컴퓨터 시스템(954N)과 같은 클라우드 소비자들에 의해 사용되는 로컬 컴퓨팅 디바이스들은 상기 클라우드 컴퓨팅 노드들(910)을 이용하여 통신할 수 있다. 비록 도 9에 나타내지는 않았으나, 클라우드 컴퓨팅 노드들(910)은 양자 플랫폼(예컨대, 양자 컴퓨터, 양자 하드웨어, 양자 소프트웨어 등)을 더 포함할 수 있으며, 클라우드 소비자들에 의해 사용되는 로컬 컴퓨팅 디바이스들은 이러한 양자 플랫폼을 이용하여 통신할 수 있다. 노드들(910)은 서로 간에 통신할 수 있다. 그들은 로컬, 커뮤니티, 퍼블릭, 또는 위에서 설명된 하이브리드 클라우드들, 또는 그것들의 조합과 같은 하나 또는 그 이상의 네트워크들에서 물리적으로 또는 가상으로 그룹화 될 수 있다(도시되지 않음). 이는 클라우드 컴퓨팅 환경(950)이 클라우드 소비자가 로컬 컴퓨팅 디바이스 상에서 자원들을 유지할 필요가 없는 서비스들과 같은 기반구조, 플랫폼들 및/또는 소프트웨어를 제공할 수 있도록 한다. 도 9에 도시된 컴퓨팅 디바이스들의 유형들(954A-N)은 단지 설명을 위한 의도인 것이 이해되어야 하고, 컴퓨팅 노드들(910) 및 클라우드 컴퓨팅 환경(950)은 네트워크 및/또는 네트워크 주소 지정 연결(network addressable connection)의 어떤 유형을 통해(예컨대, 웹 브라우저를 사용하여) 컴퓨터화된 디바이스의 어떤 유형과 통신할 수 있다는 것이 이해되어야 한다.

[0156] [00161] 이제 도 10을 참조하면, 클라우드 컴퓨팅 환경(950, 도 9)에 의해 제공된 기능적 추상화 계층들(functional abstraction layers)의 세트가 도시된다. 도 10에 도시되는 컴포넌트들, 계층들(layers), 및 기능들(functions)은 단지 예를 들어 설명하려는 의도인 것이고 발명의 실시예들은 그것으로 한정되는 것이 아니라 는 점이 미리 이해되어야 한다. 도시된 바와 같이 이하의 계층들과 해당 기능들이 제공된다.

[0157] [00162] 하드웨어 및 소프트웨어 계층(1060)은 하드웨어 및 소프트웨어 컴포넌트들을 포함한다. 하드웨어 컴포넌트들의 예들은 메인프레임들(1061), RISC(Reduced Instruction Set Computer) 아키텍처 기반의 서버들(1062), 서버들(1163), 블레이드(blade) 서버들(1064), 스토리지 디바이스들(1065), 및 네트워크 및 네트워킹 컴포넌트들(1066)을 포함한다. 몇몇 실시예들에 있어서, 소프트웨어 컴포넌트들은 네트워크 어플리케이션 서버 소프트웨어(1067), 양자 플랫폼 라우팅 소프트웨어(1068), 및/또는 양자 소프트웨어(도 10에 도시되지 않음)를 포함한다.

- [0158] [00163] 가상화 계층(Virtualization layer)(1070)은 추상화 계층(abstraction layer)을 제공한다. 추상화 계층으로부터 가상 주체들의 다음의 예들, 즉, 가상 서버들(1071), 가상 스토리지(1072), 가상 통신망들(1073) - 가상 사설 통신망들(virtual private networks)을 포함함 -, 가상 어플리케이션들 및 운영 체제들(1074), 및 가상 클라이언트들(1075)이 제공될 수 있다.
- [0159] [00164] 일 실시예에서, 관리 계층(management layer)(1080)은 이하에서 설명되는 기능들을 제공할 수 있다. 자원 공급(Resource provisioning)(1081)은 컴퓨팅 자원들 및 다른 자원들의 동적 조달(dynamic procurement)을 제공한다. 컴퓨팅 자원들 및 다른 자원들은 클라우드 컴퓨팅 환경 내에서 태스크들을 수행하는데 이용된다. 계량 및 가격 설정(Metering and Pricing)(1182)은, 자원들이 클라우드 컴퓨팅 환경 내에서 이용됨에 따라, 비용 추적(cost tracking), 및 이들 자원들의 소비에 대한 과금(billing) 및 비용청구 (invoicing)를 제공한다. 일 실시예에서, 이들 자원들은 어플리케이션 소프트웨어 라이선스들을 포함할 수 있다. 보안은 데이터 및 다른 자원들을 위한 보호뿐만 아니라, 클라우드 소비자들 및 태스크들을 위한 신원 확인(identity verification) 기능을 제공한다. 사용자 포털(user portal)(1083)은 소비자들 및 시스템 관리자들을 위한 클라우드 컴퓨팅 환경에 대한 접근을 제공한다. 서비스 수준 관리(1084)는 요구되는 서비스 수준들이 충족되도록 클라우드 컴퓨팅 자원 할당 및 관리를 제공한다. 서비스 수준 동의(Service Level Agreement, SLA) 계획(planning) 및 이행(fulfillment)(1085)은 SLA에 따라 장래 요구가 예상되는 클라우드 컴퓨팅 자원들을 위한 사전준비(pre-arrangement) 및 조달을 제공한다.
- [0160] [00165] 작업부하들 계층(Workloads layer)(1090)은 클라우드 컴퓨팅 환경이 이용될 수 있는 기능(functionality)의 예들을 제공한다. 이 계층으로부터 제공될 수 있는 작업부하들 및 기능들의 예들은, 맵핑 및 내비게이션(mapping and navigation)(1091), 소프트웨어 개발 및 라이프사이클 관리(1092), 가상 교실 교육 전달(1093), 데이터 분석 처리(data analytics processing)(1094), 거래 처리(transaction processing)(1095), 및 마이크로서비스 생성 소프트웨어(1096)를 포함하나, 이러한 예들로 한정되는 것은 아니다.
- [0161] [00166] 본 발명은 어떤 가능한 통합의 기술적 세부 수준에서 시스템, 방법, 장치 및/또는 컴퓨터 프로그램 제품일 수 있다. 컴퓨터 프로그램 제품은 컴퓨터 판독가능 스토리지 매체(또는 매체들)를 포함할 수 있고, 컴퓨터 판독가능 스토리지 매체(또는 매체들)는 컴퓨터 판독가능 스토리지 매체(또는 매체들)에 대한 컴퓨터 판독가능 프로그램 명령들을 가진다. 컴퓨터 판독가능 프로그램 명령들은 프로세서로 하여금 본 발명의 실시 예들(aspects)을 수행하도록 하기 위한 것이다. 컴퓨터 판독가능 스토리지 매체는 명령 실행 디바이스에서 사용하기 위한 명령들을 보유 및 저장할 수 있는 유형적인(tangible) 디바이스일 수 있다. 컴퓨터 판독가능 스토리지 매체는, 예를 들어, 전자 스토리지 디바이스, 자기 스토리지 디바이스, 광학 스토리지 디바이스, 전자기 스토리지 디바이스, 반도체 스토리지 디바이스, 또는 앞서 언급한 것들의 어떤 적절한 조합일 수 있으나, 이러한 것들로 한정되는 것은 아니다. 컴퓨터 판독가능 스토리지 매체의 더 많은 구체적인 예들의 리스트(전체를 다 열거한 것은 아님)는 또한, 휴대용 컴퓨터 디스켓, 하드 디스크, 랜덤 액세스 메모리(random access memory, RAM), 읽기 전용 메모리(read-only memory, ROM), 소거가능 프로그램가능 읽기 전용 메모리(erasable programmable read-only memory, EPROM 또는 플래쉬 메모리), 스태틱 랜덤 액세스 메모리(static random access memory, SRAM), 휴대용 콤팩트 디스크 읽기 전용 메모리(portable compact disc read-only memory, CD-ROM), 디지털 다용도 디스크(digital versatile disk, DVD), 메모리 스틱, 플로피 디스크, 펀치-카드들 또는 기록된 명령들을 갖는 홈(groove)에서 볼록한 구조들과 같은 기계적으로 인코딩된 디바이스, 및 앞서 열거한 것들의 어떤 적절한 조합을 포함할 수 있다. 여기서 사용되는 것과 같이, 컴퓨터 판독가능 스토리지 매체는, 무선 전파(radio waves) 또는 다른 자유롭게 전파되는 전자기파, 도파관 또는 다른 전송 매질을 통해 전파되는 전자기파(예컨대, 광섬유 케이블을 통과하는 광 펄스), 또는 와이어를 통해 전송되는 전기적 신호와 같은 일시적인 신호들 그 자체로 구성되는 것은 아니다.
- [0162] [00167] 여기에 설명되는 컴퓨터 판독가능 프로그램 명령들은, 예를 들어, 인터넷, 근거리 통신망, 광역 통신망 및/또는 무선 통신망과 같은 통신망을 통해 컴퓨터 판독가능 스토리지 매체로부터 각각의 컴퓨팅/처리 디바이스들로, 또는 내부 컴퓨터 또는 외부 스토리지 디바이스로 다운로드될 수 있다. 통신망은 동선 전송 케이블, 광 전송 섬유, 무선 전송, 라우터, 방화벽, 스위치, 게이트웨이 컴퓨터 및/또는 엣지 서버를 포함할 수 있다. 각각의 컴퓨팅/처리 디바이스에서 네트워크 어댑터 카드 또는 네트워크 인터페이스는, 각각의 컴퓨팅/처리 디바이스 내 컴퓨터 판독가능 스토리지 매체에 저장하기 위해 통신망으로부터 컴퓨터 판독가능 프로그램 명령들을 수신하고 컴퓨터 판독가능 프로그램 명령들을 전송한다. 본 발명의 동작들을 수행하기 위한 컴퓨터 판독가능 프로그램 명령들은 어셈블러 명령(assembler instructions), 명령-세트-아키텍처(instruction-set-architecture, ISA) 명령, 머신 명령(machine instructions), 머신 종속 명령(machine dependent instructions), 마이크로코드, 펌

웨어 명령(firmware instructions), 상태-세팅 데이터(state-setting data), 집적 회로망용 구성 데이터(configuration data for integrated circuitry), 또는 하나 또는 그 이상의 프로그래밍 언어들의 어떤 조합으로 작성된 소스 코드 또는 객체 코드(object code)일 수 있다. 상기 프로그래밍 언어들은 스몰토크(Smalltalk), C++ 등과 같은 객체 지향 프로그래밍 언어(object oriented programming language), 및 "C" 프로그래밍 언어 또는 유사 프로그래밍 언어들과 같은 절차형 프로그래밍 언어들을 포함한다. 컴퓨터 판독가능 프로그램 명령들은 독립형(stand-alone) 소프트웨어 패키지로서 사용자의 컴퓨터 상에서 전적으로, 사용자의 컴퓨터 상에서 부분적으로 실행될 수 있고, 사용자의 컴퓨터 상에서 부분적으로 그리고 원격 컴퓨터 상에서 부분적으로 실행될 수 있고, 또는 원격 컴퓨터 또는 서버 상에서 전적으로 실행될 수 있다. 후자의 시나리오에 있어서, 원격 컴퓨터는 근거리 통신망(LAN) 또는 광역 통신망(WAN)을 포함하는 어떤 유형의 통신망을 통해 사용자의 컴퓨터에 연결될 수 있고, 또는 그 연결은 외부 컴퓨터에 대해 이뤄질 수 있다(예컨대, 인터넷 서비스 공급자(Internet Service Provider)를 사용하여 인터넷을 통해서). 몇몇 실시예들에 있어서, 본 발명의 측면들을 수행하기 위해, 예를 들어, 프로그램가능 로직 회로망(programmable logic circuitry), 필드-프로그램가능 게이트 어레이(field-programmable gate arrays, FPGA), 또는 프로그램가능 로직 어레이(Programmable logic arrays, PLA)를 포함하는 전기 회로망(electronic circuitry)은 전기 회로망을 고객 맞춤 제작(customize)하기 위해 컴퓨터 판독가능 프로그램 명령들의 상태 정보를 이용함으로써 컴퓨터 판독가능 프로그램 명령들을 실행할 수 있다.

[0163] [00168] 본 발명의 실시 예들은 여기서 발명의 실시예들에 따른 방법들, 장치들(시스템들), 및 컴퓨터 프로그램 제품들의 플로우차트 설명들 및/또는 블록 다이어그램들을 참조하여 설명된다. 플로우차트 설명들 및/또는 블록 다이어그램들에서 플로우차트 설명들 및/또는 블록 다이어그램들의 각각의 블록, 및 블록들의 조합들은 컴퓨터 판독가능 프로그램 명령들에 의해 구현될 수 있다는 것이 이해된다. 이들 컴퓨터 판독가능 프로그램 명령들은 머신(machine)을 생성하기 위해 범용 컴퓨터(general purpose computer), 전용 컴퓨터(special purpose computer), 또는 다른 프로그램가능 데이터 처리 장치의 프로세서에게 제공될 수 있으며, 그리하여 컴퓨터 또는 다른 프로그램가능 데이터 처리 장치의 프로세서를 통해 실행될 때 그 명령들이 플로우차트 및/또는 블록 다이어그램 블록 또는 블록들에 명시된 기능들/동작들을 구현하기 위한 수단을 생성하도록 한다. 이들 컴퓨터 판독가능 프로그램 명령들은 또한 컴퓨터 판독가능 스토리지에 저장될 수 있으며, 컴퓨터 판독가능 스토리지는 컴퓨터, 프로그램가능 데이터 처리 장치, 및/또는 다른 디바이스들이 특정 방식으로 기능하도록 지시할 수 있다. 그리하여, 내부에 저장된 명령들을 갖는 컴퓨터 판독가능 스토리지 매체가 플로우차트 및/또는 블록 다이어그램의 블록 또는 블록들에 명시된 기능/동작의 측면들을 구현하는 명령들을 포함하는 제조 물품을 포함하도록 한다. 컴퓨터 판독가능 프로그램 명령들은 또한 컴퓨터, 다른 프로그램가능 데이터 처리 장치, 또는 다른 디바이스 상으로 로딩되어 컴퓨터, 다른 프로그램가능 장치 또는 다른 디바이스 상에서 수행될 일련의 동작들(operational acts)로 하여금 컴퓨터로 구현된 프로세스를 생성하게 할 수 있다. 그리하여 컴퓨터, 다른 프로그램가능 장치, 또는 다른 디바이스 상에서 실행되는 명령들이 플로우차트 및/또는 블록 다이어그램의 블록 또는 블록들에 명시된 기능들/동작들을 구현하도록 한다.

[0164] [00169] 도면들에서 플로우차트 및 블록 다이어그램들은 본 발명의 다양한 실시예들에 따른 시스템들, 방법들, 및 컴퓨터 프로그램 제품들의 가능한 구현의 아키텍처, 기능성, 및 동작을 보여준다. 이 점에서, 플로우차트 또는 블록 다이어그램들에서의 각각의 블록은 모듈, 세그먼트, 또는 명령들의 일부분을 나타낼 수 있으며, 이는 명시된 논리적 기능(들)을 구현하기 위한 하나 또는 그 이상의 실행가능한 명령들을 포함한다. 몇몇 다른 구현들에 있어서, 블록들에 기재되어 있는 기능들은 도면들에 기재되어 있는 순서를 벗어나서 발생할 수도 있다. 예를 들어, 연속해서 보여지는 두 개의 블록들은, 사실, 실질적으로 동시에 일어날 수 있고, 또는 블록들은 때로는 관련된 기능에 따라 역순으로 실행될 수도 있다. 또한, 블록 다이어그램들 및/또는 플로우차트 그림의 각각의 블록, 및 블록 다이어그램들 및/또는 플로우차트 그림에서의 블록들의 조합들은 명시된 기능들 또는 동작들(acts)을 수행하거나 전용 하드웨어와 컴퓨터 명령들의 조합들을 수행하는 전용 하드웨어 기반 시스템들에 의해 구현될 수 있다는 것이 주목될 수 있다.

[0165] [00170] 본 발명의 주제는 위에서 컴퓨터 및/또는 컴퓨터들 상에서 실행되는 컴퓨터 프로그램 제품의 컴퓨터-실행가능 명령들의 일반적인 맥락에서 설명되었지만, 통상의 기술자들은 이러한 본 발명이 다른 프로그램 모듈들과 조합하여 구현될 수도 있다는 것을 이해할 것이다. 일반적으로, 프로그램 모듈들은 특정 태스크들을 수행하고 및/또는 특정 추상적인 데이터 유형들을 구현하는 루틴들, 프로그램들, 컴포넌트들, 데이터 구조들 등을 포함한다. 더욱이, 통상의 기술자들은, 본 발명의 컴퓨터 구현 방법들이 다른 컴퓨터 시스템 구성들과 함께 실시될 수 있다는 것을 이해할 것이다. 다른 컴퓨터 시스템 구성들은, 컴퓨터들, 핸드-헬드 컴퓨팅 디바이스들(hand-held computing devices)(예컨대, PDA, 폰), 마이크로프로세서 기반 또는 프로그램가능 소비자 또는 산업용 전자제품 등 뿐만 아니라, 단일-프로세서 또는 멀티 프로세서 컴퓨터 시스템들, 멀티-컴퓨팅 디바이스들,

메인프레임 컴퓨터들을 포함한다. 도시된 본 발명의 실시 예들은 또한 분산형 컴퓨팅 환경들에서 실시될 수 있다. 분산형 컴퓨팅 환경들에서는 태스크들이 통신 네트워크를 통해 연결된 원격 처리 디바이스들에 의해 수행된다. 그러나, 전부는 아니지만 본 발명의 실시 예들은 독립형 컴퓨터들 상에서 실행될 수 있다. 분산형 컴퓨팅 환경에서, 프로그램 모듈들은 로컬 및 원격 메모리 스토리지 디바이스들 모두에 위치할 수 있다.

[0166] [00171] 본 명세서에서 사용되는 것과 같이, "컴포넌트", "시스템", "플랫폼", "인터페이스" 등의 용어들은 컴퓨터-관련 주제 또는 하나 또는 그 이상의 특정 기능들을 갖는 동작 머신과 관련된 주제를 지칭할 수 있고, 및/또는 컴퓨터-관련 주제 또는 하나 또는 그 이상의 특정 기능들을 갖는 동작 머신과 관련된 주제를 포함할 수 있다. 여기에 개시되는 주제들은 하드웨어, 하드웨어와 소프트웨어의 조합, 소프트웨어, 또는 실행 중인 소프트웨어 중 하나일 수 있다. 예를 들어, 컴포넌트는 프로세서 상에서 실행되는 프로세스, 프로세서, 객체(object), 실행 파일, 실행의 스레드(thread), 프로그램, 및/또는 컴퓨터일 수 있으나, 이러한 예들로 한정되는 것은 아니다. 예로서, 서버 상에서 실행되는 어플리케이션과 서버 둘 모두가 컴포넌트일 수 있다. 하나 또는 그 이상의 컴포넌트들은 프로세스 내에 상주할 수 있고 및/또는 실행의 스레드와 컴포넌트는 하나의 컴퓨터 상에 로컬화될 수 있고 및/또는 두 개 또는 그 이상의 컴퓨터들 사이에 분산될 수 있다. 또 다른 예에서, 각각의 컴포넌트들은 다양한 데이터 구조들이 저장된 다양한 컴퓨터 판독가능 매체로부터 실행될 수 있다. 컴포넌트들은 예를 들어 하나 또는 그 이상의 데이터 패킷들(예컨대, 로컬 시스템, 분산된 시스템에서 또 다른 컴포넌트와, 및/또는 인터넷과 같은 네트워크를 통해 신호로 다른 시스템들과 상호작용하는 하나의 컴포넌트로부터의 데이터)을 갖는 신호에 따라 로컬 및/원격 프로세스들을 통해 통신할 수 있다. 또 다른 예로서, 컴포넌트는 전기 또는 전자 회로망에 의해 동작되는 기계적 부품들에 의해 제공되는 특정 기능을 갖는 장치일 수 있는데, 이는 프로세서에 의해 실행되는 소프트웨어 또는 펌웨어 어플리케이션에 의해 동작된다. 그러한 경우에 있어서, 프로세서는 장치의 내부 또는 장치의 외부에 있을 수 있고 소프트웨어 또는 펌웨어 어플리케이션의 적어도 일부에서 실행될 수 있다. 또 다른 예로서, 컴포넌트는 기계적인 부품들이 없이 전자 컴포넌트들을 통해 특정 기능을 제공하는 장치일 수 있는데, 여기서 전자 컴포넌트들은 전자 컴포넌트의 기능의 적어도 일부를 제공하는 소프트웨어 또는 펌웨어를 실행하기 위한 프로세서 또는 다른 수단을 포함할 수 있다. 일 실시 예에서, 컴포넌트는 가상 머신을 통해, 예컨대, 클라우드 컴퓨팅 시스템 내에서, 전자 컴포넌트를 에뮬레이트(emulate)할 수 있다.

[0167] [00172] 또한, "또는"이라는 용어는 배타적인 "또는" 이라기보다는 포괄적인 "또는"을 의미하려는 의도이다. 즉, 만약 달리 명시되지 않거나 문맥상 명확하지 않다면, "X는 A 또는 B를 이용한다"라는 것은 자연적 포괄적 순열들 중 어떤 것을 의미하려는 의도이다. 즉, 만약 X가 A를 이용하고, X가 B를 이용하고, 또는 X가 A와 B 둘 모두를 이용한다면, "X는 A 또는 B를 이용한다"는 전술한 예들 중 어떤 것에도 만족된다. 더욱이, 본 명세서 및 첨부된 도면들에 사용되는 "하나" 및 "한"이라는 것은 다르게 특정되거나 문맥상 단일 형태를 가리키는 것으로 명확하지 않다면, 일반적으로 "하나 또는 그 이상"을 의미하는 것으로 해석되어야 한다. 여기에 사용되는 바와 같이, "예" 및/또는 "예시"라는 용어는, 하나의 예, 하나의 사례, 설명을 위한 예로서의 역할을 한다는 의미로 이용된다. 의심을 피하기 위해, 여기에 개시되는 본 발명의 주제는 그러한 예들로 한정되는 것은 아니다. 또한, "예" 및/또는 "예시"로서 여기에 설명되는 어떤 측면 또는 디자인은 다른 측면들 또는 디자인들에 비해 더 선호되거나 더 유리한 것으로 반드시 해석되는 것은 아니며, 또한, 당해 기술 분야에서 통상의 기술을 가진 자들에게 알려진 동등한 예시적인 구조들 및 기술들을 배제하는 것을 의미하지는 않는다.

[0168] [00173] 본 명세서에 이용되고 있는 바와 같이, "프로세서"라는 용어는 실질적으로 어떤 컴퓨팅 처리 유닛 또는 디바이스를 지칭할 수 있다. 이러한 컴퓨팅 처리 유닛 또는 디바이스는 단일-코어 프로세서들, 소프트웨어 멀티쓰레드 실행 기능을 갖는 단일-프로세서들, 멀티-코어 프로세서들, 소프트웨어 멀티쓰레드 실행 기능을 갖는 멀티-코어 프로세서들, 하드웨어 멀티쓰레드 기술을 갖는 멀티-코어 프로세서들, 병렬 플랫폼들, 및 분산형 공유 메모리(distributed shared memory)를 갖는 병렬 플랫폼들을 포함하나, 이러한 예들로 한정되는 것은 아니다. 또한, 프로세서는 집적 회로, 어플리케이션 전용 집적 회로(application specific integrated circuit, ASIC), 필드 프로그램가능 게이트 어레이(field programmable gate array, FPGA), 프로그램가능 로직 컨트롤러(programmable logic controller, PLC), 복합 프로그램가능 로직 디바이스(complex programmable logic device, CPLD), 이산 게이트(discrete gate) 또는 트랜지스터 로직(transistor logic), 이산 하드웨어 컴포넌트들, 또는 여기에 설명되는 기능들을 수행하도록 설계된 전술한 것들의 어떤 조합을 일컫을 수 있다. 나아가, 공간 사용을 최적화하거나 사용자 장비의 성능을 향상시키기 위해, 프로세서들은 나노-스케일 아키텍처들 - 예컨대, 분자 및 양자-도트 기반 트랜지스터들, 스위치들 및 게이트들, 그러나 이러한 것들로 한정되는 것은 아님 - 을 이용할 수 있다. 또한, 프로세서는 컴퓨팅 처리 유닛들의 조합으로 구현될 수도 있다. 본 명세서 내에서, "저장하다", "저장", "스토리지", "데이터 저장", "데이터 스토리지", "데이터베이스"와 같은 용어, 그리고 어떤 컴포넌트의 동작과 기능과 관련된 실질적으로 다른 어떤 정보 스토리지 컴포넌트라는 용어는 "메모리 컴

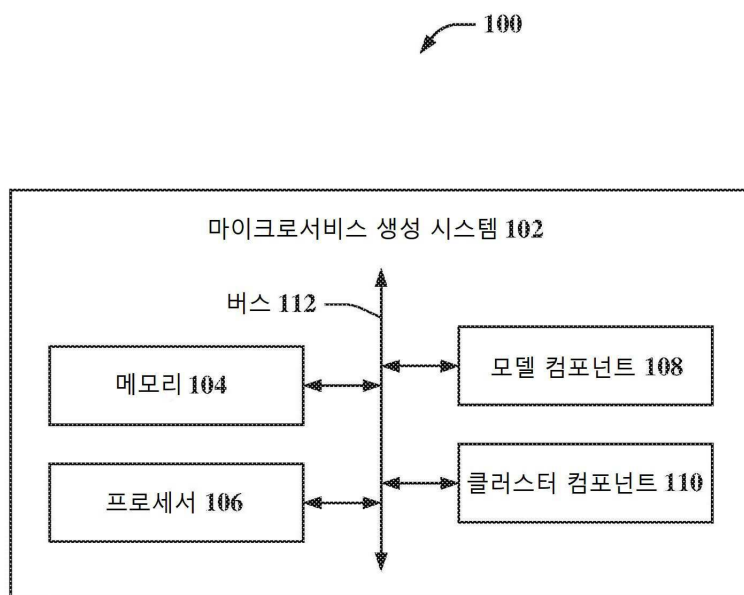
포넌트", "메모리"에 구현된 주체들, 또는 메모리를 포함하는 컴포넌트들을 일컫는 것으로 이용된다. 여기에 설명되는 메모리 및/또는 메모리 컴포넌트들은 휘발성 메모리 또는 비휘발성 메모리일 수 있고, 또는 휘발성 메모리와 비휘발성 메모리 모두를 포함할 수 있다는 것이 이해되어야 할 것이다. 예를 들어(이러한 예로 한정되는 것은 아님), 비휘발성 메모리는 읽기 전용 메모리(ROM), 프로그램가능 ROM(PROM), 전기적 프로그램가능 ROM(EPROM), 전기적 소거가능 ROM(EEPROM), 플래시 메모리, 또는 비휘발성 랜덤 액세스 메모리(RAM)(예컨대, 강유전성(Rerroelectric) RAM(FerAM))을 포함할 수 있다. 휘발성 메모리는 RAM을 포함할 수 있는데, 이는 예컨대 외부 캐시 메모리로 동작할 수 있다. 예를 들어(이러한 예로 한정되는 것은 아님), RAM은 예를 들어 싱크로너스(synchronous) RAM(SRAM), 다이نام릭(dynamic) RAM(DRAM), 싱크로너스(synchronous) DRAM(SDRAM), 더블 데이터 레이트(double data rate) SDRAM(DDR SDRAM), 향상된(enhanced) SDRAM(ESDRAM), 싱크링크(Synchlink) DRAM(SLDRAM), 다이렉트 램버스(direct Rambus) RAM(DRRAM), 다이렉트 램버스 다이نام릭(direct Rambus dynamic) RAM(DRDRAM), 및 램버스 다이نام릭(Rambus dynamic) RAM(RDRAM)과 같이 다양한 형태로 이용가능하다. 또한, 여기에서 시스템 또는 컴퓨터 구현 방법들의 개시된 메모리 컴포넌트들은 이것들 및 다른 어떤 적절한 유형들의 메모리 - 이러한 것들로 한정되는 것은 아님 -를 포함하도록 의도된다.

[0169] [00174] 위에서 설명된 것은 시스템들 및 컴퓨터 구현 방법들의 단순한 예들을 포함한다. 물론, 본 발명을 설명하려는 목적으로 컴포넌트들 또는 컴퓨터 구현 방법들의 모든 생각해낼 수 있는 조합들을 설명하는 것은 불가능하지만, 당해 기술 분야에서 통상의 기술을 가진 자는 본 발명의 많은 추가적인 조합들 및 순열들이 가능하다는 것을 알 수 있을 것이다. 더 나아가, 상세한 설명, 청구항들, 및 도면들에서 "포함하다", "갖다", "구비하다" 등과 같은 용어가 사용되는 범위 내에서, "포함하는(comprising)" 이 청구항에서 전환어(transitional word)로서 이용될 때 해석되는 것과 같이, 그러한 용어는 "포함하는" 이라는 용어와 유사한 방식으로 포괄적인 것으로 의도된다.

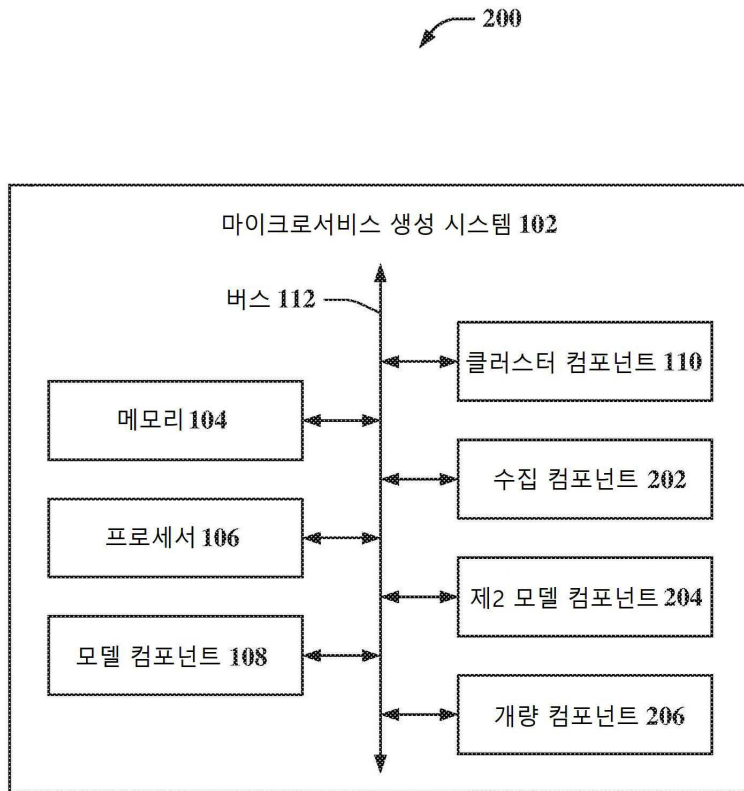
[0170] [00175] 다양한 실시예들에 관한 설명들은 예시의 목적으로 제공된 것이지 실시예들을 빠짐없이 열거하거나 개시된 실시예들로 한정하려는 의도는 아니다. 설명된 실시예들의 범위 및 사상을 벗어남이 없이 많은 변경들 및 변형들이 있을 수 있다는 것은 당해 기술 분야에서 통상의 기술을 가진 자들에게는 분명할 것이다. 여기에 사용된 용어는 실시예들의 원리, 시장에서 발견되는 기술에 대한 실제적인 적용 또는 기술적 향상을 가장 잘 설명하기 위해 선택되었다.

도면

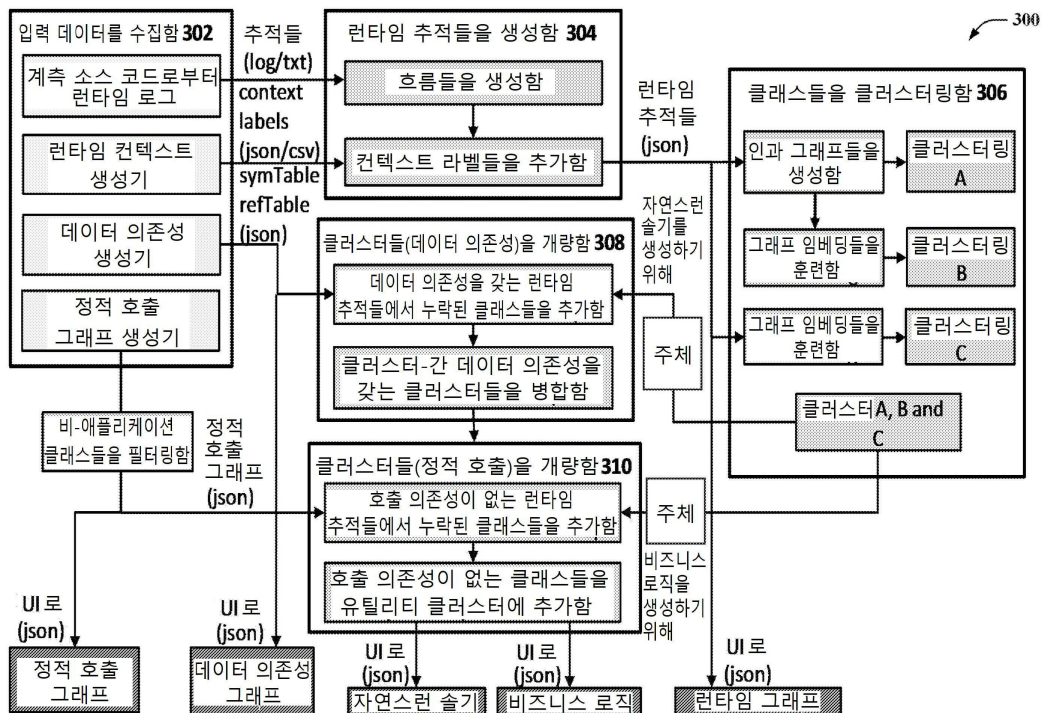
도면1



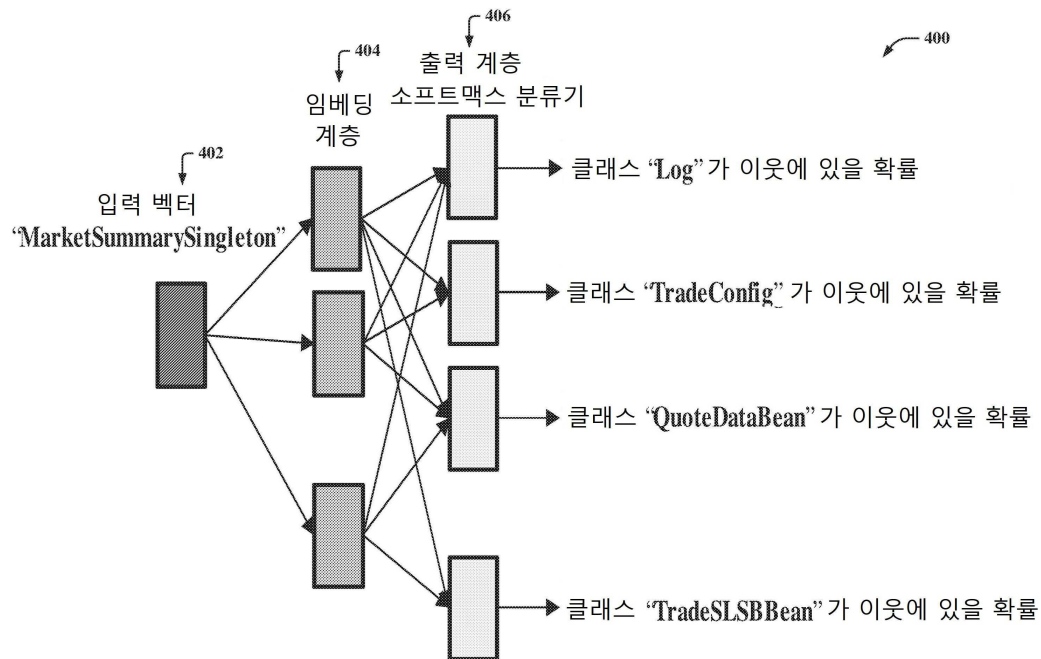
도면2



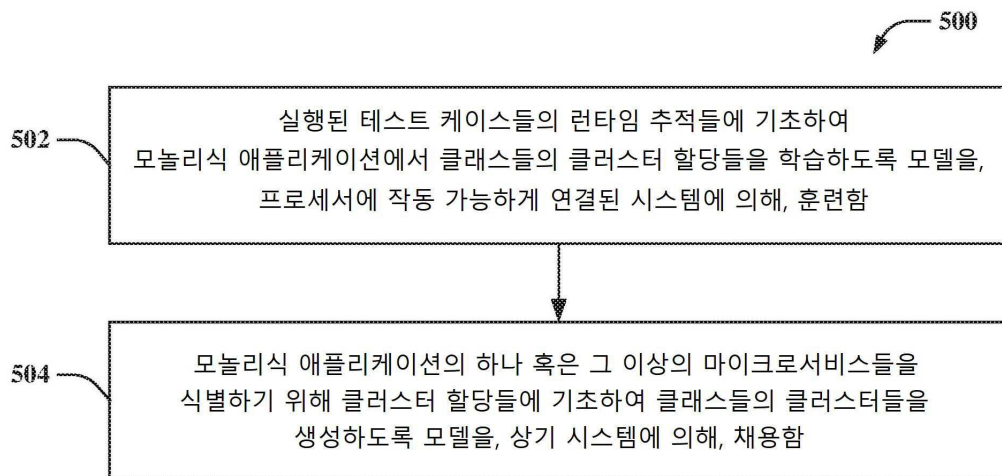
도면3



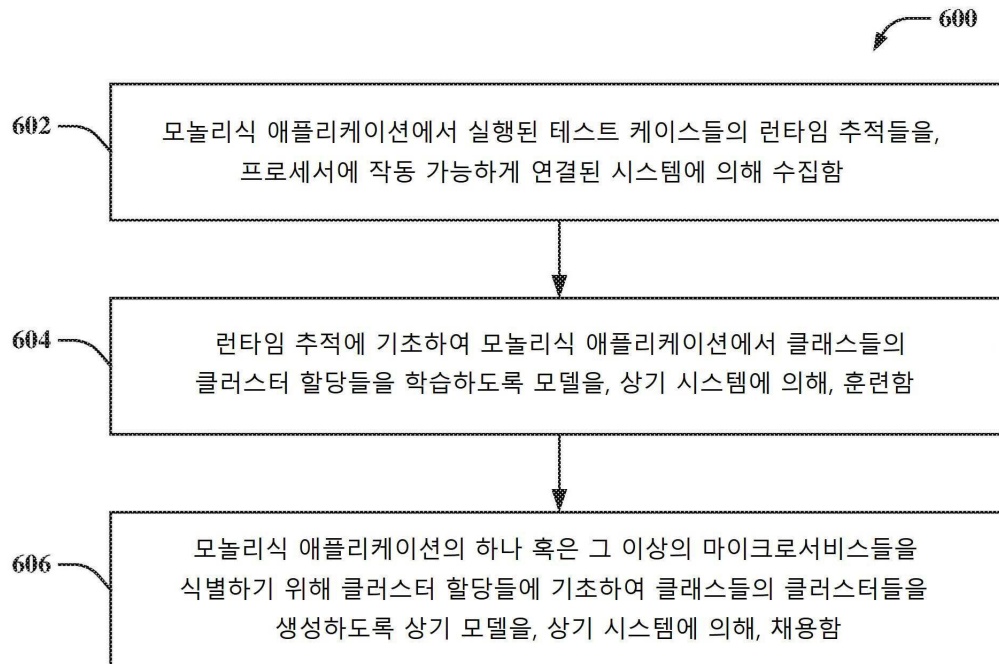
도면4



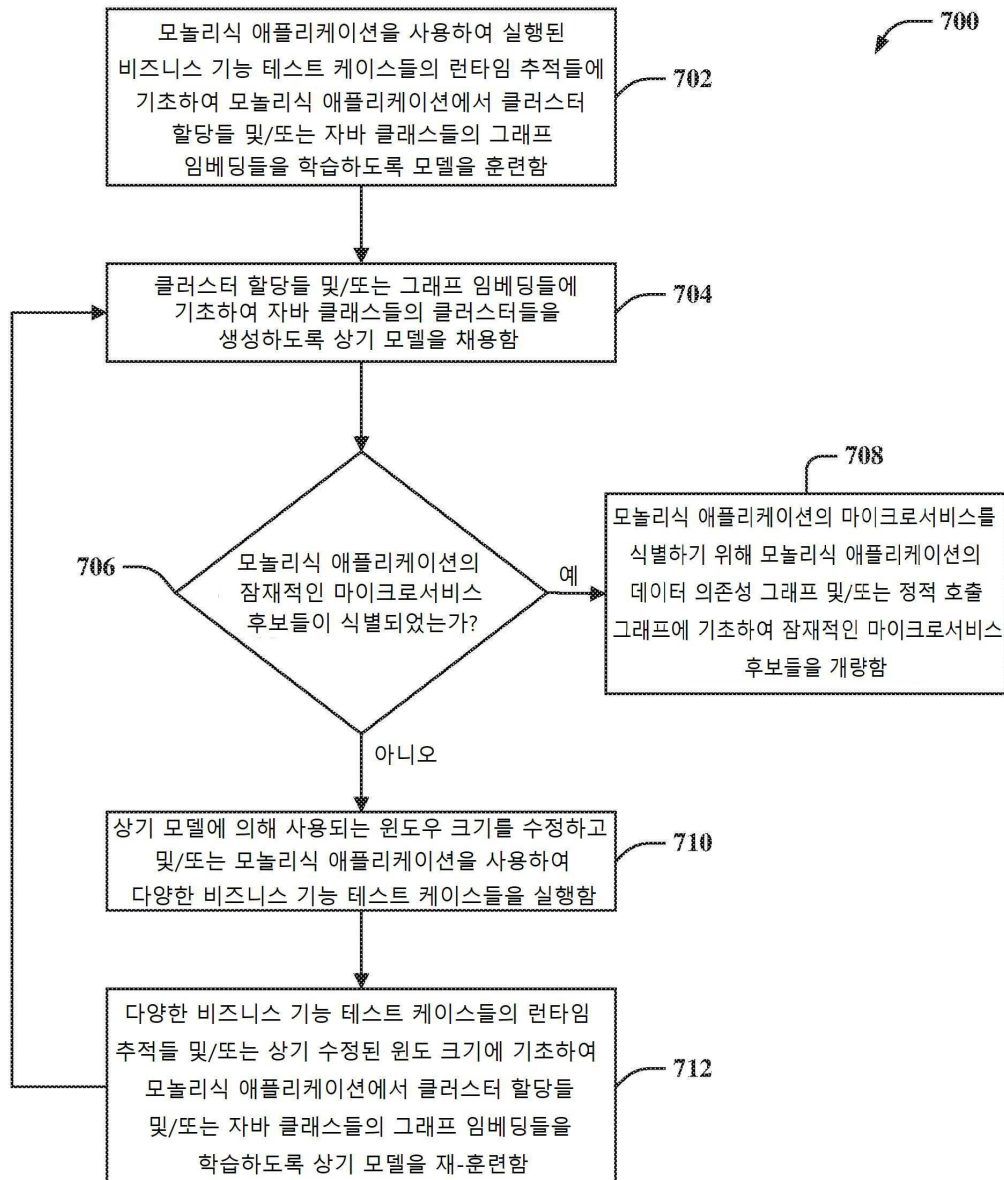
도면5



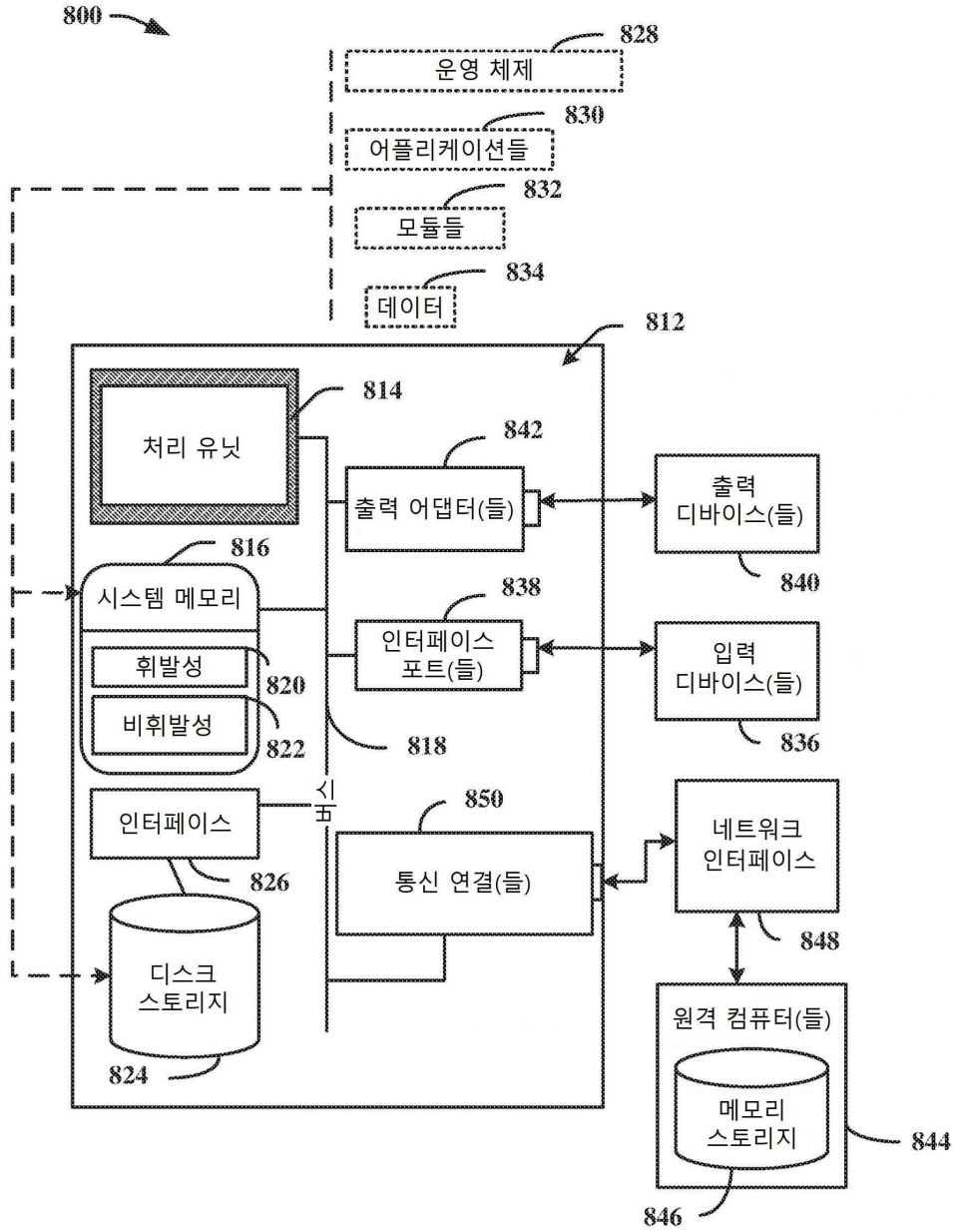
도면6



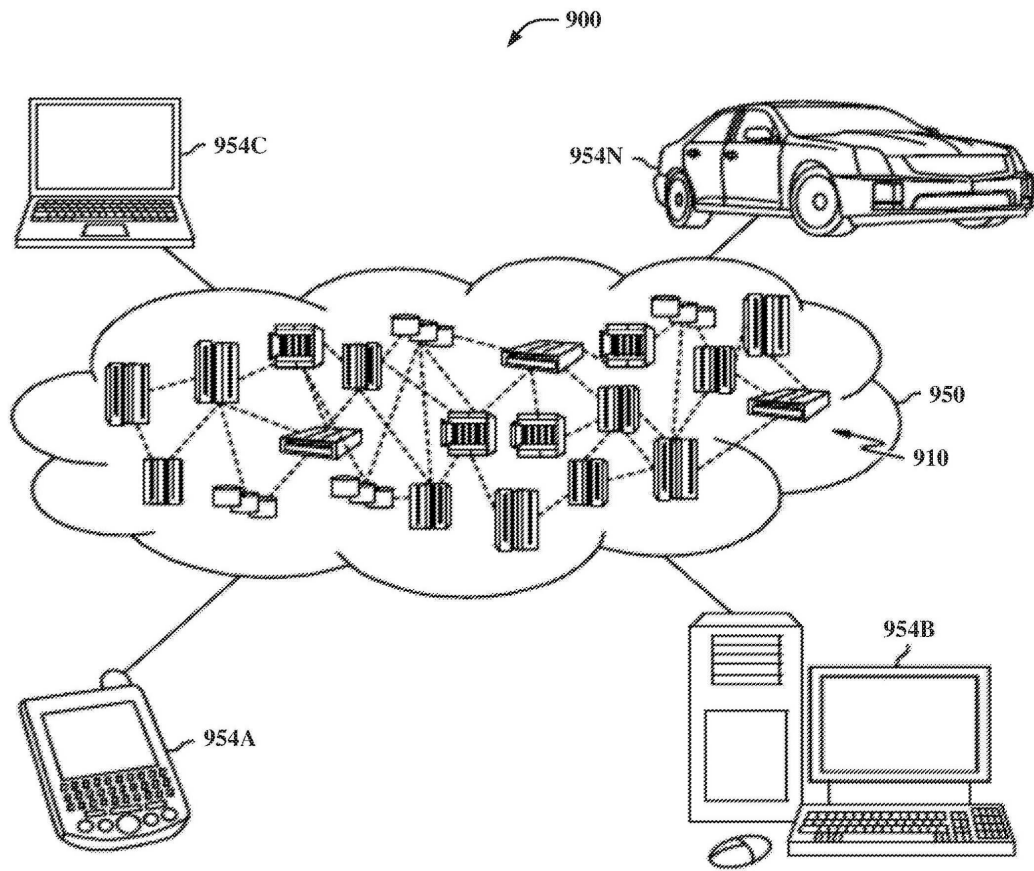
도면7



도면8



도면9



도면10

