(19) 대한민국특허청(KR)
(12) 등록특허공보(B1)

(45) 공고일자 2019년06월28일
(11) 등록번호 10-1994506
(24) 등록일자 2019년06월24일

(51) 국제특허분류(Int. Cl.)
G06F 9/06 (2018.01) *G06F 15/16* (2018.01)
(21) 출원번호 10-2014-7018945
(22) 출원일자(국제) 2013년01월07일
심사청구일자 2017년12월04일
(85) 번역문제출일자 2014년07월08일
(65) 공개번호 10-2014-0109939
(43) 공개일자 2014년09월16일
(86) 국제출원번호 PCT/US2013/020442
(87) 국제공개번호 WO 2013/106256
국제공개일자 2013년07월18일
(30) 우선권주장
13/346,303 2012년01월09일 미국(US)
(56) 선행기술조사문헌
JP2003533766 A
(뒷면에 계속)
전체 청구항 수 : 총 20 항

(73) 특허권자
마이크로소프트 테크놀로지 라이센싱, 엘엘씨
미국 워싱턴주 (우편번호 : 98052) 레드몬드 원 마이크로소프트 웨이
(72) 발명자
캘더 브래들리 진
미국 워싱턴주 98052-6399 레드몬드 원 마이크로소프트 웨이 엘씨에이 - 인터내셔널 페이턴츠 마이크로소프트 코포레이션
왕 주
미국 워싱턴주 98052-6399 레드몬드 원 마이크로소프트 웨이 엘씨에이 - 인터내셔널 페이턴츠 마이크로소프트 코포레이션
(뒷면에 계속)
(74) 대리인
제일특허법인(유)

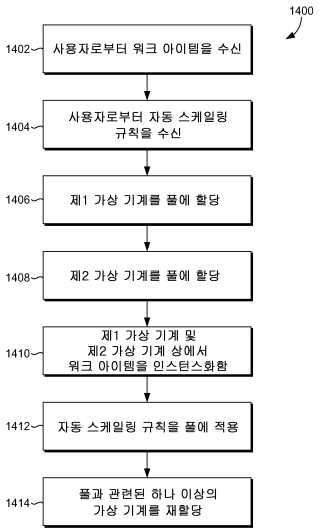
심사관 : 유진태

(54) 발명의 명칭 **PAAS 자원들, 작업들 및 스케줄링의 분리 기법**

(57) 요 약

작업을 수행하기 위해 분산 컴퓨팅 환경에서 서비스로서 플랫폼(platform as a service)을 통해 자원들을 제공하기 위한 시스템들 및 방법들이 제공된다. 시스템의 자원들, 시스템에서 수행되는 작업, 및 시스템에서 수행되는 작업들의 스케줄러들은 작업이 자원들 사이에서 쉽게 이동하는 것을 가능하게 하는 방식으로 분리된다. 제1 자원 풀로부터 제2 자원 풀로의 작업들의 이동이 인간의 개입 없이 시스템에 의해 수행되는 것이 고려된다. 작업의 이동은 상이한 자원들에 대해 상이한 스케줄러들을 사용할 수 있다. 또한, 자원들의 풀이 작업의 이동에 응답하여 추가적인 또는 더 적은 자원들을 자동으로 할당할 수 있는 것이 고려된다.

대 표 도 - 도14



(72) 발명자

베데카 바만

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

산카란 스리람

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

맥네트 마빈 쥬니어

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

군다 프라딥 쿠마르

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

장 양

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

안토니 샤이얌

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

마니바난 카비사

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

스크졸스볼드 아릴드 이

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

카트리 헤말

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

(56) 선행기술조사문헌

KR1020010080208 A

KR1020030086298 A

KR1020050000487 A

US20100333113 A1

명 세 서

청구범위

청구항 1

분산 컴퓨팅 환경에서 자원을 제공하기 위한 컴퓨터 구현 방법으로서,

사용자로부터 워크 아이템(work item)을 수신하는 단계와,

상기 워크 아이템으로부터 작업(job)을 생성하는 단계와,

프로세서 및 메모리를 이용하여, 적어도 제1 가상 기계를 제1 풀(pool)에 자동으로 할당하는 단계와,

상기 제1 풀과 연관되고 작업 큐(job queue)로부터 상기 작업의 태스크를 차용하는 제1 스케줄러를 이용하여, 상기 제1 풀 상에서 상기 작업을 스케줄링하는 단계 - 상기 태스크는 재시도 카운트(retry count)와 연관됨 - 와,

상기 제1 스케줄러를 이용하여, 상기 제1 풀의 상기 제1 가상 기계 상에서 상기 작업을 개시하는 상기 태스크를 차용하는 단계 - 상기 제1 가상 기계는 소정의 기간에 상기 작업 큐로부터의 상기 태스크에 대한 차용을 유지함 - 와,

상기 제1 가상 기계 상에서 상기 태스크를 실행하는 단계 - 상기 태스크는 장애 시에 소정의 횟수까지 상기 제1 가상 기계 상에서 재스케줄링됨 - 와,

상기 기간에 상기 재시도 카운트에 대응하는 상기 횟수에 응답하여 상기 태스크를 적어도 제2 가상 기계를 포함하는 제2 풀에 재할당하는 단계 - 상기 작업의 상기 태스크에 대한 차용은 상기 제1 스케줄러로부터 해제됨 - 와,

상기 제2 풀에 연관되고 상기 작업 큐로부터 상기 작업의 상기 태스크를 차용하는 제2 스케줄러를 이용하여 상기 제2 풀의 상기 제2 가상 기계 상에서 상기 태스크를 스케줄링하는 단계와,

상기 제2 풀의 상기 제2 가상 기계 상에서 상기 태스크를 개시하는 단계를 포함하는

방법.

청구항 2

제1항에 있어서,

상기 제1 풀 상에서의 상기 작업의 상기 스케줄링은 상기 제2 풀 상에서 상기 태스크를 스케줄링하는 데 사용되지 않는 상기 제1 스케줄러를 적어도 부분적으로 이용하여 수행되는

방법.

청구항 3

제1항에 있어서,

상기 제1 가상 기계 상에서 상기 태스크를 실행하는 단계는 상기 제1 풀 내의 장애를 검출하는 단계를 포함하는

방법.

청구항 4

제3항에 있어서,

상기 태스크의 상기 재할당은 상기 제1 풀 내의 상기 검출된 장애에 응답하는

방법.

청구항 5

제1항에 있어서,

상기 태스크의 상기 재할당은 상기 제2 풀 상에서의 상기 태스크의 계속을 가능하게 하는

방법.

청구항 6

제1항에 있어서,

상기 재시도 카운트는 상기 태스크를 이용하여 지정되는

방법.

청구항 7

제1항에 있어서,

상기 기간에 상기 재시도 카운트에 대응하는 상기 횟수에 기초하여 상기 태스크를 에러에 의해 종료된 것으로 마킹하는 단계를 더 포함하는

방법.

청구항 8

제1항에 있어서,

상기 재할당은 상기 제1 풀과 연관된 하나 이상의 태스크 위치 서비스와 통신하는 상위 레벨 위치 서비스에 의해 관리되는

방법.

청구항 9

제1항에 있어서,

상기 작업의 상기 재할당은 사용자 개입 없이 상기 작업이 상기 제1 풀로부터 상기 제2 풀로 이동하는 것을 가능하게 하는

방법.

청구항 10

제1항에 있어서,

상기 제2 가상 기계는 상기 제1 가상 기계와 상이한 지리적 위치에 있는

방법.

청구항 11

제1항에 있어서,

상기 제2 가상 기계는 상기 제1 가상 기계와 상이한 태스크 테넌트(task tenant)에 있는

방법.

청구항 12

제1항에 있어서,

상기 기간에 상기 재시도 카운트에 대응하는 상기 횟수는 상기 재시도 카운트의 횟수를 포함하는

방법.

청구항 13

제1항에 있어서,

상기 제1 풀은 둘 이상의 태스크 테넌트를 포함하는

방법.

청구항 14

제1항에 있어서,

상기 제2 가상 기계는 상기 워크 아이템과 연관된 사양(specification)에 부분적으로 기초하여 상기 제2 풀에 할당되는

방법.

청구항 15

제1항에 있어서,

상기 제1 가상 기계와 상기 제2 가상 기계는 상이한 물리 프로세서를 이용하는

방법.

청구항 16

프로세서 및 메모리를 갖는 컴퓨팅 장치에 의해 실행될 때 분산 컴퓨팅 환경에서 자원을 제공하기 위한 방법을 수행하는 컴퓨터 사용 가능 명령어를 저장하는 하나 이상의 컴퓨터 저장 메모리 장치로서,

상기 방법은

자원의 제1 풀과 연관된 제1 스케줄러를 이용하여, 상기 분산 컴퓨팅 환경에서 상기 자원의 제1 풀 상에서 작업을 스케줄링하는 단계 - 상기 작업의 태스크는 작업 큐로부터 차용되고, 상기 제1 풀 상의 제1 가상 기계는 소정의 기간에 상기 태스크에 대한 차용을 유지하며, 상기 태스크는 재시도 카운트와 연관됨 - 와,

상기 제1 풀의 상기 제1 가상 기계 상에서 상기 태스크를 실행하는 단계 - 상기 태스크는 장애 시에 소정의 횟

수까지 상기 제1 가상 기계 상에서 재스케줄링됨 - 와,

상기 기간에 상기 재시도 카운트에 대응하는 상기 횟수에 응답하여, 상기 태스크가 상기 분산 컴퓨팅 환경에서 상기 제1 풀로부터 제2 풀로 이동되어야 하는 것으로 결정하는 단계와,

상기 작업 큐 내의 상기 태스크에 대한 상기 차용을 상기 제1 가상 기계로부터 해제하는 단계와,

워크 아이템과 연관된 상기 태스크를 상기 제2 풀로 이동시키는 단계와,

프로세서 및 메모리를 이용하는 상기 제2 풀과 연관된 제2 스케줄러를 이용하여, 상기 제2 스케줄러가 상기 태스크에 대한 차용을 획득하도록 상기 제2 풀의 제2 가상 기계 상에서 상기 태스크를 스케줄링하는 단계와,

상기 제2 풀의 상기 제2 가상 기계 상에서 상기 작업을 개시하는 단계를 포함하는

하나 이상의 컴퓨터 저장 메모리 장치.

청구항 17

제16항에 있어서,

상기 제1 풀 및 상기 제2 풀은 상기 제1 스케줄러와 연관된 태스크 위치 서비스와 통신하고 상기 제2 스케줄러와 연관된된 태스크 위치 서비스와 통신하는 상위 레벨 위치 서비스에 의해 선택되는

하나 이상의 컴퓨터 저장 메모리 장치.

청구항 18

제16항에 있어서,

상기 재시도 카운트는 상기 태스크를 이용하여 지정되는

하나 이상의 컴퓨터 저장 메모리 장치.

청구항 19

제16항에 있어서,

이동한 후에, 상기 작업과 연관된 태스크 중 일부가 상기 제1 풀 상에서 계속 스케줄링되는

하나 이상의 컴퓨터 저장 메모리 장치.

청구항 20

분산 컴퓨팅 환경에서 자원을 제공하기 위한 시스템으로서,

상기 시스템은 방법을 수행하도록 구성된 하나 이상의 프로세서를 포함하며,

상기 방법은

상기 분산 컴퓨팅 환경에서 태스크 계정과 연관된 워크 아이템을 사용자로부터 수신하는 단계와,

상기 분산 컴퓨팅 환경의 제1 가상 기계를 제1 풀에 자동으로 할당하는 단계와,

상기 제1 풀과 연관된 제1 스케줄러를 이용하여, 상기 제1 풀의 상기 제1 가상 기계 상에서 작업을 스케줄링하는 단계 - 상기 작업은 상기 워크 아이템과 연관된 하나 이상의 태스크를 포함하고, 상기 하나 이상의 태스크는 각각 재시도 카운트와 연관됨 - 와,

상기 제1 스케줄러를 이용하여, 작업 큐로부터 상기 하나 이상의 태스크 중 하나의 태스크를 차용하는 단계 - 상기 제1 가상 기계는 상기 작업 큐로부터 소정의 기간에 상기 태스크에 대한 차용을 유지함 - 와,

상기 제1 가상 기계 상에서 상기 태스크를 실행하는 단계 - 상기 태스크는 장애 시에 소정의 횟수까지 상기 제1 가상 기계 상에서 재스케줄링됨 - 와,

상기 기간에 상기 재시도 카운트에 대응하는 상기 횟수에 응답하여, 상기 태스크가 상기 분산 컴퓨팅 환경 내에서 제2 풀로 이동되어야 하는 것으로 결정하는 단계와,

상기 태스크의 차용을 상기 제1 가상 기계로부터 해제하는 단계와,

상기 워크 아이템과 연관된 상기 태스크를 상기 제2 풀로 이동시키는 단계와,

상기 제2 풀과 연관된 제2 스케줄러를 이용하여, 상기 제2 풀의 제2 가상 기계 상에서 상기 태스크를 스케줄링하는 단계와,

상기 제2 스케줄러를 이용하여, 상기 제2 풀에서 개시하는 상기 태스크를 차용하는 단계를 포함하는

시스템.

발명의 설명

기 술 분 야

배 경 기 술

[0001] 대규모 계산 작업들을 수행하기 위한 전통적인 방법들은 종종 사용자가 분산 환경에서 테넌트들(tenants)을 적극적으로 관리하고 작업들에 대한 큐들(queues)을 관리할 것을 요구하였다. 이러한 사용자의 적극적인 개입은 큰 자원 클러스터들을 포괄하고 그러한 클러스터들의 사용을 효율적으로 스케일링하기 위한 작업의 능력을 방해할 수 있다. 또한, 작업들은 전통적으로 작업, 작업을 완료하는 데 사용되는 자원들 및 자원들에 대한 작업의 스케줄링이 밀접하게 결합되는 방식으로 생성되어, 장애 또는 부하 균형화에 대응한 작업의 효율적인 이동이 방해될 수 있다.

발명의 내용

[0002] 다양한 실시예들에서, 작업을 수행하기 위해 분산 컴퓨팅 환경에서 서비스로서 플랫폼(platform as a service)을 통해 자원들을 제공하기 위한 시스템들 및 방법들이 제공된다. 시스템의 자원들, 시스템에서 수행되는 작업, 및 시스템에서 수행되는 작업들의 스케줄러들은 작업이 자원들 사이에서 쉽게 이동하는 것을 가능하게 하는 방식으로 분리된다. 제1 자원 풀(pool)로부터 제2 자원 풀로의 작업들의 이동이 인간의 개입 없이 시스템에 의해 수행되는 것이 고려된다. 작업의 이동은 상이한 자원들에 대해 상이한 스케줄러들을 사용할 수 있다. 또한, 자원들의 풀이 작업의 이동에 응답하여 추가적인 또는 더 적은 자원들을 자동으로 할당할 수 있는 것이 고려된다.

[0003] 본 요약은 아래의 상세한 설명에서 더 설명되는 개념들의 발췌를 간단한 형태로 소개하기 위해 제공된다. 본 요약은 청구 발명의 중요한 특징들 또는 본질적인 특징들을 식별하는 것을 의도하지 않으며, 청구 발명의 범위를 결정함에 있어서 고립된 보조물로서 사용되는 것도 의도하지 않는다.

도면의 간단한 설명

[0004] 본 발명은 아래에서 첨부 도면들을 참조하여 상세히 설명된다. 도면들에서:

도 1은 본 발명의 양태들에 따른, 본 발명의 실시예들을 구현하는 데 적합한 예시적인 동작 환경을 도시한다.

도 2는 본 발명의 양태들에 따른 예시적인 작업을 도시하는 블록도를 나타낸다.

도 3은 본 발명의 양태들에 따른 예시적인 풀을 도시하는 블록도를 나타낸다.

도 4는 본 발명의 양태들에 따른, 분산 컴퓨팅 환경 내에서 작업들을 수행하는 데 적합한 예시적인 시스템의 아

키텍처 계층들을 도시하는 블록도를 나타낸다.

도 5는 본 발명의 양태들에 따른, 단일 분산 컴퓨팅 아키텍처 내에 통합될 수 있는 상이한 위치들의 프로세서들의 예시적인 배열을 도시하는 블록도를 나타낸다.

도 6은 본 발명의 양태들에 따른, 태스크 위치 서비스에 대한 가능한 구성을 도시하는 블록도를 나타낸다.

도 7은 본 발명의 양태들에 따른 예시적인 태스크 위치 서비스 프론트엔드("TLSFE")를 도시하는 블록도를 나타낸다.

도 8은 본 발명의 양태들에 따른 예시적인 워크 아이템/작업 스케줄러("WIJ")를 도시하는 블록도를 나타낸다.

도 9는 본 발명의 양태들에 따른 태스크 테넌트를 도시하는 블록도를 나타낸다.

도 10은 본 발명의 양태들에 따른 예시적인 풀 서버를 도시하는 블록도를 나타낸다.

도 11은 본 발명의 양태들에 따른, 예시적인 분산 컴퓨팅 시스템 내의 다양한 컴포넌트들 사이의 예시적인 워크플로우을 제공하는 통신 도면을 나타낸다.

도 12는 본 발명의 양태들에 따른, 분산 컴퓨팅 환경에서 자원들을 제공하기 위한 예시적인 방법을 도시하는 블록도를 나타낸다.

도 13은 본 발명의 양태들에 따른, 분산 컴퓨팅 환경에서 자원들을 제공하기 위한 추가적인 예시적인 방법을 도시하는 블록도를 나타낸다.

도 14는 본 발명의 양태들에 따른, 분산 컴퓨팅 환경에서 자원들을 제공하기 위한 다른 방법을 도시하는 블록도를 나타낸다.

도 15는 본 발명의 양태들에 따른, 분산 컴퓨팅 환경에서 자원, 스케줄링 및 작업들의 분리를 제공하기 위한 방법을 도시하는 블록도를 나타낸다.

발명을 실시하기 위한 구체적인 내용

[0005] 다양한 실시예들에서, 작업을 수행하기 위해 분산 컴퓨팅 환경에서 서비스로서 플랫폼을 통해 자원들을 제공하기 위한 시스템들 및 방법들이 제공된다. 시스템의 자원들, 시스템에서 수행되는 작업, 및 시스템에서 수행되는 작업들의 스케줄러들은 작업이 자원들 사이에서 쉽게 이동하는 것을 가능하게 하는 방식으로 분리된다. 제1 자원 풀(pool)로부터 제2 자원 풀로의 작업들의 이동이 인간의 개입 없이 시스템에 의해 수행되는 것이 고려된다. 작업의 이동은 상이한 자원들에 대해 상이한 스케줄러들을 사용할 수 있다. 또한, 자원들의 풀이 작업의 이동에 응답하여 추가적인 또는 더 적은 자원들을 자동으로 할당할 수 있는 것이 고려된다.

[0006] 제1의 예시적인 양태는 분산 컴퓨팅 환경에서 자원들을 제공하기 위한 컴퓨터 구현 방법을 포함한다. 이 방법은 사용자로부터 태스크 계정과 관련된 워크 아이템(work item)을 수신하는 단계를 포함한다. 또한, 방법은 워크 아이템으로부터 작업을 생성하는 단계를 포함한다. 게다가, 방법은 프로세서 및 메모리를 이용하여 적어도 제1 가상 기계를 제1 풀에 자동 할당하는 단계를 포함하는 것이 고려된다. 방법은 제1 풀 상에 작업을 할당하고 제1 풀 상의 작업에 대한 태스크들을 스케줄링하는 단계를 포함하는 것으로 더 고려된다. 또한, 방법은 적어도 제2 가상 기계를 포함하는 제2 풀에 작업을 재할당하는 단계를 포함하는 것이 고려된다. 게다가, 방법은 제2 풀 상의 작업에 대한 태스크들을 스케줄링하는 단계를 포함한다.

[0007] 본 명세서에서 제공되는 제2의 예시적인 양태는 프로세서 및 메모리를 갖는 컴퓨팅 장치에 의해 실행될 때 분산 컴퓨팅 환경에서 자원들을 제공하기 위한 방법을 수행하는 컴퓨터 사용 가능 명령어들을 저장하는 컴퓨터 저장 매체를 포함한다. 방법은 제1 스케줄러를 이용하여 분산 컴퓨팅 환경에서 제1 자원 풀 상에 작업을 스케줄링하는 단계를 포함한다. 방법은 작업을 제1 풀에 할당하는 단계를 더 포함한다. 게다가, 방법은 사용자의 개입 없이, 분산 컴퓨팅 환경 내에서 작업이 제1 풀로부터 제2 풀로 이동해야 하는 것으로 결정하는 단계를 포함한다. 게다가, 방법은 제2 풀 상에 작업을 할당하는 단계를 포함한다. 방법은 프로세서 및 메모리를 이용하여 제2 스케줄러를 이용하여 제2 풀 상에 작업을 자동 스케줄링하는 단계를 더 포함한다.

[0008] 본 명세서에서 설명되는 본 발명의 제3의 예시적인 양태는 분산 컴퓨팅 환경에서 자원들을 제공하기 위한 컴퓨터 구현 방법을 포함한다. 방법은 태스크 위치 서비스에서, 분산 컴퓨팅 환경에서 사용자로부터 태스크 계정과 관련된 워크 아이템을 수신하는 단계를 포함한다. 방법은 태스크 위치 서비스와 관련된 풀 서버를 이용하여, 분산 컴퓨팅 환경의 제1 가상 기계들의 세트를 제1 풀에 자동 할당하는 단계를 더 포함한다. 방법은 제1 풀 상

에 제1 작업을 스케줄링하는 단계를 더 포함하며, 제1 작업은 워크 아이템과 관련된 하나 이상의 태스크를 포함한다. 또한, 방법은 제1 풀의 부하 균형화 평가에 기초하여, 작업이 분산 컴퓨팅 환경 내에서 제2 풀로 이동되어야 하는 것으로 자동 결정하는 단계를 포함한다. 게다가, 방법은 워크 아이템/작업과 관련된 적어도 하나의 태스크를 제2 풀로 이동시키는 단계를 포함한다. 방법은 또한 제2 풀의 하나 이상의 자원 상에 적어도 하나의 태스크를 스케줄링하는 단계를 포함한다.

[0009] 개요

[0010] 네트워크들을 통한 데이터 전송의 속도의 증가 및 다른 네트워크 특징들의 향상으로 인해, 컴퓨팅 자원들이 큰 네트워크 전반에 분산되는 환경에서 대규모 컴퓨팅 태스크들을 수행하는 것이 더욱더 가능하다. 제1 위치의 사용자는 작업 또는 컴퓨팅 태스크를 컴퓨팅 서비스에 제출하고, 사용자가 직접적인 지식을 갖지 않은 컴퓨터들의 그룹에서 태스크가 수행되게 할 수 있다. 사용자의 태스크를 수행하기 위한 컴퓨팅 자원들은 상이한 주들, 국가들 및/또는 대륙들에 걸칠 수 있는 다수의 물리 위치들에 걸쳐 분산될 수 있다. 하나 이상의 위치들에 위치하는 컴퓨팅 자원들의 제1 그룹이 사용자의 컴퓨팅 태스크를 수행하기 위한 데이터 및 기타 정보를 저장할 수 있고, 동일 위치들에 있거나 아마도 하나 이상의 위치의 상이한 세트에 있는 컴퓨팅 자원들의 제2 그룹이 컴퓨팅 태스크를 수행하는 데 사용될 수 있다. 또한, 데이터를 저장하기 위한 하나 이상의 위치는 사용자가 알거나 알지 못하는 상이한 지리 영역들에 걸쳐 분산될 수 있는 것이 고려된다. 유사하게, 컴퓨팅 자원들은 사용자가 알거나 알지 못하는 상이한 지리 위치들에 걸쳐 분산될 수 있는 것이 고려된다.

[0011] 다양한 분산 컴퓨팅 자원들에 대한 액세스는 컴퓨팅 자원들이 어디에 위치하는지에 관계없이 사용자가 작업 태스크들을 수행하는 (예를 들어, 프로세스들을 실행하는) 것을 가능하게 한다. 분산된 자원들은 또한 사용자가 지정된 시간에 의해 또는 원하는 비용 값으로 컴퓨팅 태스크를 완료하는 것과 같이 컴퓨팅 태스크의 목적을 충족시키기 위해 사용되는 자원들의 양을 스케일 아웃(또는 스케일 인)하기 위한 기회를 제공한다. 그러나, 사용자에게 이러한 유연성을 제공하는 것은 분산 컴퓨팅 자원들의 운영자(및/또는 소유자)에게 다수의 과제를 부과한다. 수요를 충족시키기 위해, 자원들의 분산 네트워크의 운영자는 최대 수요의 시간에 자원 요청을 충족시키기에 충분한 이용 가능 자원들을 갖는 것이 바람직할 것이다.

[0012] 예시적인 컴퓨팅 환경

[0013] 일반적으로 도면들 그리고 먼저 특히 도 1을 참조하면, 본 발명의 실시예들을 구현하는 데 적합한 예시적인 동작 환경이 도시되고, 일반적으로 컴퓨팅 장치(100)로서 지시된다. 컴퓨팅 장치(100)는 적절한 컴퓨팅 환경의 일례일 뿐이며, 본 발명의 이용 또는 기능의 범위에 대한 어떠한 한정도 암시하지 않는 것을 의도한다. 또한, 컴퓨팅 장치(100)는 도시된 모듈들/컴포넌트들 중 어느 하나 또는 조합에 관한 임의의 종속성 또는 요구를 갖는 것으로 해석되지 않아야 한다.

[0014] 실시예들은 컴퓨터 또는 다른 기계, 예로서 개인 휴대 단말기 또는 다른 핸드헬드 장치에 의해 실행되는 프로그램 모듈들과 같은 컴퓨터 실행 가능 명령어들을 포함하는 컴퓨터 코드 또는 기계 사용 가능 명령어들과 일반적으로 관련하여 설명될 수 있다. 일반적으로, 루틴, 프로그램, 객체, 모듈, 데이터 구조 등을 포함하는 프로그램 모듈들은 특정 태스크들을 수행하거나 특정 추상 데이터 타입들을 구현하는 코드를 지칭한다. 실시예들은 핸드헬드 장치, 소비자 전자 장치, 범용 컴퓨터, 특수 컴퓨팅 장치 등을 포함하는 다양한 시스템 구성들에서 실시될 수 있다. 실시예들은 통신 네트워크를 통해 링크되는 원격 처리 장치들에 의해 태스크들이 수행되는 분산 컴퓨팅 환경에서도 실시될 수 있다.

[0015] 도 1을 계속 참조하면, 컴퓨팅 장치(100)는 다음 장치들, 즉 메모리(112), 하나 이상의 프로세서(114), 하나 이상의 프레젠테이션 모듈(116), 입출력(I/O) 포트들(118), I/O 모듈들(120) 및 예시적인 전원(122)을 직접 또는 간접적으로 결합하는 버스(110)를 포함한다. 버스(110)는 (어드레스 버스, 데이터 버스 또는 이들의 조합과 같은) 하나 이상의 버스일 수 있는 것을 나타낸다. 도 1의 다양한 블록들은 명료화를 위해 라인들로 도시되지만, 실제로는 다양한 모듈들의 묘사는 그렇게 명확하지는 않으며, 은유적으로 라인들은 더 정확하게는 흐리거나 희미할 것이다. 예를 들어, 디스플레이 장치와 같은 프레젠테이션 모듈은 I/O 모듈인 것으로 간주할 수 있다. 또한, 프로세서들은 메모리를 갖는다. 본 발명자들은 그러한 것이 기술의 본질인 것으로 인식하며, 도 1의 도면은 하나 이상의 실시예와 관련하여 사용될 수 있는 예시적인 컴퓨팅 장치를 예시할 뿐이라는 것을 반복해서 언급한다. "워크스테이션", "서버", "랩탑", "핸드헬드 장치" 등과 같은 카테고리들은 모두가 도 1의 범위 내에서 고려되고 "컴퓨터" 또는 "컴퓨팅 장치"를 지칭하므로 이들은 구별이 되지 않는다.

[0016] 컴퓨팅 장치(100)는 통상적으로 다양한 컴퓨터 판독 가능 매체들을 포함한다. 한정이 아니라 예로서, 컴퓨터

판독 가능 매체들은 랜덤 액세스 메모리(RAM), 판독 전용 메모리(ROM), 전기적으로 소거 및 프로그래밍 가능한 판독 전용 메모리(EEPROM); 플래시 메모리 또는 기타 메모리 기술들; CDROM, 디지털 다기능 디스크들(DVD) 또는 기타 광학 또는 홀로그래픽 매체들; 자기 카세트들, 자기 테이프, 자기 디스크 저장 장치 또는 기타 자기 저장 장치들, 또는 원하는 정보를 인코딩하는 데 사용될 수 있고 컴퓨팅 장치(100)에 의해 액세스될 수 있는 임의의 다른 매체를 포함할 수 있다.

[0017] 메모리(112)는 휘발성 및/또는 비휘발성 메모리 형태의 비일시적 컴퓨터 저장 매체들을 포함한다. 메모리는 이동식, 비이동식 또는 이들의 조합일 수 있다. 예시적인 하드웨어 장치들은 반도체 메모리, 하드 드라이브, 광 디스크 드라이브 등을 포함한다. 컴퓨팅 장치(100)는 메모리(112) 또는 I/O 모듈들(120)과 같은 다양한 엔티티들로부터 데이터를 판독하는 하나 이상의 프로세서를 포함한다. 프레젠테이션 모듈(들)(116)은 사용자 또는 다른 장치에 데이터 지시들을 제공한다. 예시적인 프레젠테이션 모듈들은 디스플레이 장치, 스피커, 인쇄 모듈, 진동 모듈 등을 포함한다. I/O 포트들(118)은 컴퓨팅 장치(100)로 하여금 일부가 내장될 수 있는 I/O 모듈들(120)을 포함하는 다른 장치들에 논리적으로 결합되는 것을 가능하게 한다. 예시적인 모듈들은 마이크, 조이스틱, 게임 패드, 위성 안테나, 스캐너, 인쇄기, 무선 장치 등을 포함한다.

[0018] 정의들

[0019] "계정"은 분산 컴퓨팅 환경 내에서 고유하게 식별되는 전역적 엔티티이다. 일 실시예에서, 후술하는 자원들 및 태스크들 모두는 계정의 범위 내에 있다. 통상적으로, 사용자는 분산 컴퓨팅 시스템의 자원들을 사용하기 전에 먼저 계정을 생성할 것이다. 사용자는 계정을 생성한 후에 계정을 이용하여 워크 아이템들을 시스템에 제출하고, 워크 아이템들에 기초하여 작업들을 수행하기 위한 자원들을 관리할 수 있다.

[0020] "워크 아이템"은 분산 컴퓨팅 환경에서 실행될 작업의 정적 표현이다. 워크 아이템은 작업 바이너리, 처리될 데이터에 대한 포인터 및 옵션으로서 작업을 수행하기 위한 태스크들을 런칭하기 위한 명령 라인을 포함하는 작업의 다양한 양태들을 지정할 수 있다. 게다가, 워크 아이템은 재발생 스케줄, 우선순위 및 제약들을 지정할 수 있다. 예를 들어, 워크 아이템은 매일 오후 5시에 런칭되는 것을 지정할 수 있다.

[0021] "작업"은 워크 아이템의 실행 인스턴스이다. 작업은 분산 컴퓨팅을 수행하기 위해 협력하는 태스크들의 집합을 포함한다. 태스크들은 분산 컴퓨팅 환경 내의 하나 이상의 가상 기계에서 실행될 수 있다. 작업은 아래에서 도 2와 관련하여 더 상세히 설명된다.

[0022] "태스크"는 작업의 기본 실행 단위이다. 각각의 태스크는 가상 기계에서 실행된다. 사용자들은 명령 라인에 대한 추가 입력 및 각각의 태스크를 위한 입력 데이터에 대한 포인터들을 지정할 수 있다. 태스크는 태스크의 실행의 과정 동안에 태스크를 수행하는 가상 기계 상의 그의 워킹 디렉토리 아래에 파일들의 계층 구조를 생성할 수 있다.

[0023] (본 명세서에서 "JM 태스크"라고도 하는) "작업 관리자 태스크"는 작업 내의 특별 태스크이다. 작업 관리자 태스크는 옵션이며, 따라서 일부 작업들은 JM 태스크의 사용 없이 수행될 수 있다. 작업 관리자 태스크는 작업 내의 태스크들 모두에 대해 단일 제어 포인트를 제공할 수 있으며, 작업에 대한 "마스터" 태스크로서 사용될 수 있다. 작업이 JM 태스크를 갖는 경우, 시스템은 JM 태스크를 작업 내에 제1 태스크로서 런칭한다. 이어서, JM 태스크는 더 많은 태스크를 작업에 제출할 수 있으며, 이러한 태스크들의 진행을 모니터링하고, 태스크들의 다음 묶음을 언제 제출할지를 제어할 수 있다. 그러나, JM 태스크가 작업과 관련될 때에도 시스템 밖의 하나 이상의 서비스에 의해 작업에 태스크들이 제출될 수도 있는 것도 고려된다. 이러한 방식으로, JM 태스크는 작업 내의 태스크들의 스케줄링을 조정하고, 태스크들 사이의 의존성을 관리할 수 있다. 작업 관리자 태스크에 대한 노드 또는 가상 기계가 고장나는 경우, JM 태스크는 다른 가상 기계에서 재개될 수 있으며, 따라서 JM 태스크는 대응하는 작업을 위해 항상 실행된다. 예시적인 양태에서, JM 태스크는 다른 가상 기계에서 자동으로 재개된다. 게다가, 사용자들은 JM 태스크가 완료되면 시스템이 대응하는 작업 내의 모든 태스크들을 종료할 수 있다는 것을 시스템에 대해 지정할 수 있다. 예시적인 JM 태스크(또는 작업 태스크 관리자로도 지칭됨)는 아래에서 도 2와 관련하여 설명된다.

[0024] 작업

[0025] 위에서 정의된 바와 같이, 작업은 워크 아이템의 실행 인스턴스이다. 도 2는 본 발명의 양태들에 따른 예시적인 작업(200)의 블록도를 나타낸다. 작업(200)은 태스크(204)와 같은 복수의 태스크를 포함한다. 위에서 또한 정의된 바와 같이, 태스크는 (본 명세서에서 태스크 기계로도 지칭되고 아래에서 설명되는) 태스크 태넌트 내의 가상 기계에 의해 실행되는 작업의 기본 실행 단위이다. 복수의 태스크에 더하여, 작업(200)은 옵션으로서 (본

명세서에서 JM 태스크로도 지칭되는) 작업 태스크 관리자(202)를 포함할 수 있다. 작업 태스크 관리자(202)는 작업(200) 내의 모든 다른 태스크들(예로서, 태스크(204))에 대해 단일 제어 포인트를 제공할 수 있다.

[0026] 본 명세서에서 더 상세히 설명되는 바와 같이, 워크 아이템은 작업의 정적 표현이다. 워크 아이템은 일 실시예에서 분산 컴퓨팅 시스템의 자원이 계산 자원들을 개시하면(예를 들어, 작업을 로딩하고, 작업 큐잉하고, 작업 내의 태스크들을 인스턴스화하면) 작업으로서 지칭된다. 즉, 예시적인 양태에서, 워크 아이템은 시스템이 워크 아이템의 처리를 개시하면 작업이 된다.

[0027] 가상 기계 풀들

[0028] 가상 기계는 처리 능력의 논리 유닛을 지칭한다. 가상 기계는 물리 프로세서와 일대일 대응을 가질 수 있거나, 가상 기계는 복수의 프로세서에 대응할 수 있거나, 가상 기계는 하나 이상의 프로세서에서의 처리 시간/사이클들의 백분율을 나타낼 수 있다. 여하튼, 예시적인 양태에서 가상 기계는 도 1의 컴퓨팅 장치(100)에 의해 적어도 부분적으로 설명될 수 있는 것이 고려된다.

[0029] 다양한 실시예들에서, 워크 아이템에 기초하여 작업을 수행할 수 있는 가상 기계들은 사용 전에 워크 아이템에 대한 계정과 관련된다. "풀"은 가상 기계들의 논리적 그룹핑이다. 도 3은 본 발명의 양태들에 따른, 가상 기계(302)와 같은 복수의 가상 기계를 포함하는 풀(300)의 블록도를 나타낸다. 풀은 상이한 데이터 센터, 상이한 지리 위치 및 상이한 물리 구성들에 걸치는 가상 기계들을 할당했을 수 있는 것이 고려된다.

[0030] 예시적인 양태에서, 워크 아이템은 워크 아이템에 대응하는 작업(들)을 실행하기 위해 적어도 하나의 관련 풀을 항상 갖는다. 각각의 계정(예로서, 태스크 계정)은 계정이 계정과 관련된 워크 아이템들을 수행하는 데 사용하기 위해 독점적인 액세스를 획득하는 하나 이상의 풀을 생성할 수 있다. 워크 아이템이 사용자에 의해 제출될 때 풀이 생성될 수 있거나, 워크 아이템이 기존 풀과 관련될 수 있다. 옵션으로서, 계정에 대응하는 단일 워크 아이템 또는 워크 아이템들의 다른 서브세트와 함께 사용하기 위해 풀이 관련될 수 있다. 더구나, 작업을 위해 시스템에 의해 풀이 자동으로 생성될 수 있는 것이 고려된다. 예를 들어, 재발생 워크 아이템이 매일 특정 시간에 실행될 수 있고, 통상적으로 완료를 위해 2 시간을 필요로 할 수 있다. 이 예에서, 매일 작업이 생성될 때 풀이 자동으로 생성될 수 있고, 작업이 완료될 때 풀이 삭제될 수 있다.

[0031] 워크 아이템이 사용자에 의해 제출될 때, 워크 아이템은 예시적인 양태에서 가상 기계들의 하나 이상의 풀과 관련될 수 있다. 또한, 워크 아이템은 단일 풀과 독점적으로 관련될 수 있는 것이 고려된다(또한, 다수의 워크 아이템/작업이 공통 풀과 관련될 수 있는 것이 고려된다). 가상 기계들은 풀 내에 임의의 편리한 방식으로 구성될 수 있다. 예를 들어, 가상 기계에 대한 기본 프로세서의 지리 위치에 관계없이 모든 가상 기계들이 단일 풀 내에 구성될 수 있다. 다른 옵션은 지리 위치에 기초하여 가상 기계들을 구성하는 것이며, 따라서 풀의 모든 가상 기계들은 주어진 지리 위치 내에 존재한다. 또 다른 옵션은 지리 위치가 아닌 다른 것, 예를 들어 다른 변수들(예로서, 저장 자원, 네트워크 지연, 사용자 위치/선호 및 보안 요구)에 대한 근접에 기초하여 가상 기계들을 구성하는 것이다.

[0032] 풀을 형성하기 위한 다른 고려되는 프로세스는 시스템 자원을 이용하여 풀을 자동으로 생성한다. 자동 풀 생성은 워크 아이템이 생성될 때 또는 작업 자체가 생성될 때 시스템에 의해 풀이 자동 생성되는 것을 가능하게 한다. 이러한 프로세스는 고객/사용자/클라이언트로부터 풀의 생성을 추상화한다. 이러한 모드 동작에서, 고객/사용자/클라이언트는 실행할 워크 아이템 또는 작업에 대한 그들의 책임이 제한된다. 예시적인 양태에서, 풀은 작업을 실행할 때 자동 생성되며, 풀은 작업이 종료될 때 자동 제거된다. 또한, 풀은 워크 아이템이 생성될 때 자동 생성되고, 풀은 워크 아이템이 삭제/종료될 때 자동 제거되는 것이 고려된다.

[0033] 예시적인 양태에서 가상 기계가 하나의 태스크 및/또는 많은 태스크를 실행할 수 있는 것이 고려된다. 또한, 다수의 작업이 동일한 가상 기계 풀에서 실행될 수 있는 것이 고려된다. 예시적인 양태에서 VM 풀은 사용자의 개입 없이 그리고 작업들로부터의 명확한 처리 없이 자동으로 크기가 늘거나 감소할 수 있다. 예를 들어, 작업은 풀의 확장 또는 축소를 보상하는 것을 책임지지 않을 수 있다. 유사하게, 작업이 다수의 풀에 걸칠 수 있는 것이 고려된다. 이렇게 다수의 풀에 걸치는 것은 VM 자원들에서 독립적으로 확장 및 축소될 수 있는 다수의 풀에 걸쳐 작업을 부하 균형화함으로써 달성될 수 있다. 더구나, 풀이 주어진 시간에 0개의 가상 기계를 포함할 수 있는 것이 고려된다. 이것은 수행할 작업에 대한 태스크가 존재하지 않을 때 발생할 수 있다. 결과적으로, 풀이 계산 자원들을 절약하기 위해 소정 기간 동안 0개의 VM으로 축소될 수 있는 것이 고려된다.

[0034] 전용, 대기 및 선점 기계들

[0035] 일 실시예에서, 가상 기계가 풀에 할당될 때, 가상 기계는 2개의 타입 중 (적어도) 하나일 수 있다. 가상 기계

는 전용 가상 기계 또는 선점 가상 기계로서 풀에 할당될 수 있다. 가상 기계의 전용 또는 선점으로서의 상태는 가상 기계가 풀 내에 있는 동안 변경될 수도 있다.

[0036] "전용" 가상 기계는 풀에 할당되는 워크 아이템들/작업들에 의한 전용 사용을 위해 풀에 할당되는 기계이다. 옵션으로서, 전용 가상 기계는 풀에 제출된 임의의 작업을 위해 일반적으로 사용 가능한 것이 아니라, 하나 이상의 관련된 워크 아이템을 위한 전용 사용을 위해 할당될 수 있다. 가상 기계가 전용 상태를 갖는 동안, 기계는 풀과 관련된 워크 아이템들/작업들에 의한 사용을 위해 예약된다.

[0037] "선점" 가상 기계는 계정을 위해 그러나 가상 기계가 풀을 위해 계속 사용 가능할 것이라는 보증 없이 그 풀에서 현재 작업을 수행하고 있는 가상 기계이다. 선점 가상 기계가 풀에 의해 사용 가능할 때, 선점 기계는 그 풀에 추가되어, 이제 워크 아이템들/작업들에 의해 사용될 수 있다. 이어서, 선점 기계는 그 풀에 대한 작업을 수행하기 위해 제공되고 사용된다. 선점 기계는 임의의 편리한 방법에 의해, 예를 들어 풀이 계정을 위해 자원 경매에서 선점 가상 기계에서의 처리 시간을 획득하게 함으로써 풀에 의해 사용될 수 있다.

[0038] 계정에 대한 사용이 가능해지는 가상 기계는 통상적으로 분산 컴퓨팅 환경에서 다른 목적을 갖는 가상 기계일 것이다. 예를 들어, 선점 가상 기계들의 하나의 소스는 장애 복구 목적을 위해 분산 컴퓨팅 환경 소유자/운영자에 의해 제공되는 가상 기계들이다. 안정된 동작을 제공하기 위하여, 분산 컴퓨팅 환경은 예약 상태로 유지되는 가상 기계들의 하나 이상의 그룹을 포함할 수 있다. 이러한 예약 가상 기계들은 프로세서 장애, 네트워크 장애, 또는 분산 환경의 일부가 더 이상 작업들을 수행하는 데 적합하지 않게 하는 임의의 다른 종류의 이벤트로 인해 손실된 자원들을 대체하는 데 사용될 수 있다. 풀에 할당된 하나 이상의 전용 가상 기계가 이벤트로 인해 손실될 때, 손실된 기계들은 예약 가상 기계들을 이용하여 대체될 수 있다. 이것은 분산 컴퓨팅 환경 내의 자원들의 가용성을 향상시킨다. 그러나, 장애 이벤트가 드문 것이 바람직하므로, 재난 복구 기계들을 예약하는 것은 종종 많은 수의 가상 기계가 유휴 상태이고 사용되기를 기다린다는 것을 의미할 것이다. 장애 이벤트들을 처리하도록 지정된 이러한 가상 기계들의 CPU 사이클들을 낭비하는 것이 아니라, 이러한 가상 기계들의 CPU 사이클들은 워크 아이템들/작업들을 실행하기 위한 선점 VM들로서 풀들에 할당될 수 있다. 장애가 발생하고, 시스템이 전용 자원들의 요건을 채우기 위해 선점 자원들을 줄이는 것이 필요한 경우에, 그러한 가상 기계에서 실행되는 선점 작업이 가능 하자마자(그리고 아마도 즉시) 중지될 것이고, 따라서 선점 가상 기계는 손실 또는 장애 소스를 대체하는 그의 원래 목적을 위해 사용될 수 있다.

[0039] 선점 기계들의 다른 소스는 여분의 능력을 갖는 가상 기계들이다. 통상적으로, 임의의 네트워크의 최대 부하는 평균 부하와 다를 것이다. 결과적으로, 최대 부하 상황을 처리하기에 충분한 자원들을 갖는 컴퓨팅 환경은 종종 다른 시간 동안 사용 가능한 여분의 자원들을 가질 것이다. 이러한 여분의 자원들은 자원 쿠션을 제공한다. 사용자가 추가의 전용 가상 기계들을 요청할 때, 여분의 가상 기계들이 사용자의 요청을 충족시키는 데 사용될 수 있다. 분산 컴퓨팅 환경이 전용 기계들에 대한 최대 부하보다 작은 부하를 가질 때, 하나 이상의 가상 기계가 자유로울 것이다. 여분의 능력을 제공하도록 지정된 이러한 가상 기계들의 CPU 사이클들을 낭비하는 것이 아니라, 이러한 가상 기계들의 CPU 사이클들은 사용자들 및 풀들에 선점 방식으로 할당될 수 있다. 전용 가상 기계들에 대한 요청들의 부하가 증가할 때, 이러한 여분의 가상 기계들에서 실행되는 선점 작업들은 가능 하자마자(그리고 아마도 즉시) 중지될 것이다. 이것은 선점 가상 기계가 필요시에 추가 전용 자원들을 제공하는 그의 원래 목적을 위해 사용되는 것을 가능하게 한다. 추가로 또는 대안으로서, 전용 기계들에 대한 부하의 일부 증가들은 전용 기계들에 대한 스케줄링된 요청들에 기인할 것이다. 가상 기계가 스케줄링된 시간에 전용 기계로서의 사용으로 인해 사용 불가능하게 되는 경우, 가상 기계에 할당된 선점 작업이 스케줄링된 시간 전에 중지되어, 선점 작업으로부터 전용 자원들로의 순차 전이를 가능하게 할 수 있다.

[0040] 가상 기계들의 또 다른 소스는 "대기" 예약에서 풀 또는 계정과 관련된 가상 기계들이다. "대기" 가상 기계 예약은 제1 계정 또는 풀과 관련되고 그러한 제1 계정 또는 풀에 의한 사용을 위해 제공되는 가상 기계 예약이다. 또한, 대기 가상 기계 예약은 특정 풀에 첨부되지 않으며, 대신에 시스템은 계정 또는 풀에 의해 대기를 위해 유지되는 정의된 수의 VM들을 유지하는 것이 고려된다. 대기 VM들이 필요할 때, 대기 VM들의 예약 수는 필요한 수의 VM을 충족시키도록 감소할 수 있다. 예시적인 양태에서, 시스템 내의 VM들의 전체 수는 전용 VM들 + 예약된 대기 VM들 + 다른 곳에 할당되지 않은 컴퓨팅을 위해 자유로운 VM들 + 장애 및 여분의 제공을 위해 예약 상태로 유지되는 VM들과 동일할 수 있다. 이 예에서, 시스템에서 사용되는 선점 VM들의 수는 예약된 대기 VM들 + 컴퓨팅을 위해 자유로운 VM들의 수보다 적거나 동일한 것이 고려된다.

[0041] 대기 기계 예약의 제공은 시스템 내의 어딘가에(예를 들어, 풀 레벨, 계정 레벨) VM 능력을 예약하는 것을 포함할 수 있다. 대기 가상 기계 예약은 가상 기계의 할당이 아니다. 대신에, 대기 가상 기계 예약은 유휴 또는

선점 VM을 취하여 이를 해당 풀 또는 계정 사용을 위한 전용 VM으로 변경하기 위한 미래의 권리를 예약한다. 카운트인 대기 VM은 2개의 상이한 풀과 관련될 수 있다.

[0042] 대기 기계들에 대한 하나의 사용은 특정 시간 프레임 동안에만 발생하는 높은 우선순위의 계산 작업들을 갖는 사용자들을 위한 것이다. 예를 들어, 금융 회사는 주식 교환 또는 상품 교환과 같은 하나 이상의 금융 시장의 일상 활동의 분석을 수행하기를 원할 수 있다. 이 예를 계속하면, 금융 시장들은 정의된 스케줄에 따라 열리고 닫힐 수 있는데, 예를 들어 오전 9시 30분에 열리고 오후 4시에 닫힐 수 있다. 금융 회사는 분석 또는 시뮬레이션을 수행하는 데 사용하기 위해 금융 시장들이 열리는 시간들로부터 데이터를 집계하기를 원할 것이다. 분석의 목적은 다음 날 시장들이 열리기 전에 그들의 직원들에게 정보를 제공하는 것이다. 그러한 분석은 많은 수의 가상 기계를 필요로 할 수 있지만, 가상 기계들은 예를 들어 오후 6시부터 다음 날 오전 3시 30분까지의 시간들 사이에만 필요하다. 이러한 시간 동안, 금융 회사는 가상 기계들의 가용성의 보증을 원한다. 하루의 나머지 동안, 금융 회사는 기계들을 필요로 하지 않는다. 대기 가상 기계 예약을 금융 회사의 계정에 할당하는 것은 이러한 목적을 달성할 수 있다. 예약 가격의 지불을 위한 교환시에, 금융 회사는 원하는 시간 동안 기계들의 가용성을 보증받는다. 원하는 시간 윈도 밖에서, 가상 기계들은 금융 회사 및/또는 다른 사용자들을 위한 선점 기계들로서 사용될 수 있다. 예약을 이행하기 위해 선점 VM들을 취하는 것이 필요한 대기 예약을 실행할 때, 선점 작업들은 스케줄링된 가용성 이벤트 전에 순차적으로 중지될 수 있다.

[0043] 대기 VM 예약이 전용 기계들로 변환될 때, 이것은 시간 기반 기준들에 기초하는 변환으로 정의된다. 즉, 대기 VM 예약은 사전 결정된 시간 및/또는 날짜에 적어도 부분적으로 기초하여 전용 기계들로 변환된다. 시간 기반 기준들은 활동 임계치를 정의하는 데 사용되는 활동 기준들과 대비된다. 활동 임계치는 하나 이상의 분산 자원의 사용 및/또는 성능에 기초하는 임계치에 대응한다. 예를 들어, 계정에서, 고객은 다수의 대기 VM의 예약을 위해 지불할 수 있으며, 그러한 대기 VM 예약은 시간 기반 기준들 또는 동적 임계치 자동 스케일링 기준들에 대해 사용될 수 있다. 또한, 대기 VM 예약은 예약 시간 또는 다른 스케줄링 예약 정책들에 관계없이 임의의 포인트에 변환될 수 있다. 예를 들어, 사용자(또는 관리자)는 예약으로부터의 하나 이상의 대기 VM이 변환되어야 한다는 요청을 제공할 수 있다.

[0044] 대기 VM 예약에 대한 다른 사용은 작업을 스케일 아웃할 때 성능 향상을 가능하게 하는 것이다. 예를 들어, 소매점은 분산 컴퓨팅 자원들을 이용하여 휴일 전에 쇼핑 시즌 동안의 추가의 온라인 트래픽, 예를 들어 소매상의 웹사이트를 검토하고 주문을 행하기 위한 온라인 트래픽을 처리할 수 있다. 과거의 경험에 기초하여, 소매상은 소정 레벨의 온라인 활동을 예상하고, 대응하는 수의 전용 가상 기계를 예약한다. 그러나, 온라인 활동이 예상보다 많은 경우, 소매상은 또한 대기 모드에서 추가의 기계들을 예약한다. 이어서, 소매상은 예상보다 높은 활동 레벨을 지시하는 하나 이상의 임계치를 설정할 수 있다. 이러한 임계치들이 발생할 때, 대기 VM 예약들을 이용하여 유휴 또는 선점 기계들을 변환함으로써, 소매상으로 하여금 소매상의 고객들이 느린 응답 시간을 경험하게 하지 않고서 추가의 온라인 트래픽을 처리하게 할 수 있다. 이러한 상황에서는, 대기 VM 예약이 비선점 시간에 전용 기계로 변환될 수 있는데, 이는 활동 임계치가 충족되는 시기가 알려지지 않을 수 있기 때문이다. 활동 임계치가 충족될 때, 유휴 VM들이 사용되거나, 선점 태스크들이 중지되며, 기계는 전용 기계로 변환된다.

[0045] 분산 네트워크 환경에서의 컴퓨팅 자원들의 구성 예

[0046] 분산 컴퓨팅 환경의 사용자는 통상적으로 분산 컴퓨팅 자원들(예로서, 클라우드 컴퓨팅 자원들)을 이용하여 작업들을 수행하기를 원할 것이다. 작업들은 통상적으로 분산 컴퓨팅 환경을 통해, 예를 들어 네트워크(예로서, 인터넷)를 통해 액세스할 수 있는 위치들에 저장된 데이터에 대해 작업들을 수행하는 것을 포함할 수 있다. 운영자가 분산 컴퓨팅 환경을 제공하기 위한 한 가지 방법은 환경을 다수의 계층으로서 제공하는 것이다. 도 4는 본 발명의 양태들에 따른, 분산 컴퓨팅 환경 내에서 태스크들을 수행하기에 적합한 예시적인 시스템의 계층들을 형성하는 블록도를 나타낸다. 도 4의 시스템은 (태스크 런타임 계층으로도 지칭될 수 있는) 시스템 에이전트 런타임 계층(408), (제삼자 태스크 런타임 계층으로도 지칭될 수 있는) 애플리케이션 런타임 계층(406), 자원 관리 계층(402), 및 스케줄링 및 실행 계층(404)을 포함한다.

[0047] 도 4에 도시된 실시예에서, 시스템 에이전트 런타임 계층(408)은 풀에 대한 실행 환경 및 보안 상황, 및 실행 환경에서 실행될 워크 아이템/작업/태스크를 설정하는 것을 담당한다. 시스템 에이전트 런타임 계층(408)은 또한 태스크들을 런칭하고, 태스크들의 상태를 모니터링할 수 있다. 시스템 에이전트 런타임 계층(408)은 각각의 가상 기계에서 실행되는 시스템 에이전트의 형태를 취할 수 있다. 태스크 런타임 계층은 또한 사용자의 태스크 실행 파일들 내로 링크될 수 있는 런타임 라이브러리를 포함할 수 있다. 런타임 라이브러리들을 시스템 에이전트 런타임 계층(408)의 일부로서 갖는 것은 시스템 에이전트에 의해 실행되는 태스크들에 더 풍부한 능력을 잠

재적으로 제공할 수 있다. 런타임 라이브러리들의 예들은 태스크들 사이의 고속 통신을 가능하게 하기 위한 하나 이상의 효율적인 통신 라이브러리; 다른 가상 기계들 및/또는 다른 태스크들로부터 파일들을 판독하기 위한 효율적인 원격 파일 액세스 라이브러리 지원; 태스크들이 (예를 들어, 바이너리 대형 객체들 내로) 체크포인트하고 재개하는 것을 가능하게 하기 위한 체크포인트 라이브러리; 로깅 라이브러리(logging library); 및 가상 기계들의 풀 내에서 주어진 태스크를 수행하는 가상 기계들에 걸쳐 사용될 분산 파일 시스템을 제공하기 위한 라이브러리를 포함한다.

[0048] 애플리케이션 런타임 계층(406)은 추가 런타임들이 시스템 에이전트 런타임 계층(408)의 상부에 구축되고 실행되는 것을 가능하게 한다. 애플리케이션 런타임 계층(406)은 또한 작업에 대한 태스크들의 실행을 조정하기 위한 추가 능력들을 제공할 수 있다. 그 예들은 VM들의 풀 내에서 주어진 태스크를 수행하는 VM들에 걸쳐 사용될 분산 파일 시스템을 제공하기 위한 라이브러리에 대한 맵 축소 런타임을 포함할 수 있다. 이것은 사용자가 사용자의 작업들 또는 태스크들에 대해 맞춤화된 방식으로 분산 컴퓨팅 환경을 구성하는 것을 가능하게 한다.

[0049] 자원 관리 계층(402)은 분산 컴퓨팅 환경에서 이용 가능한 컴퓨팅 자원들의 관리를 처리한다. 하나의 옵션은 자원 관리 계층(402)이 자원들을 3개의 상이한 레벨에서 관리하게 하는 것이다. 제1 레벨에서, 자원 관리 계층(402)은 작업(즉, 워크 아이템의 실행)과 관련된 가상 기계들은 물론, 태스크와 관련된 각각의 가상 기계에 저장된 파일들의 할당 및 할당 해제를 관리하며, 이는 가상 기계 레벨(410)에 의해 도시된다.

[0050] 제2 레벨에서, 작업과 관련된 가상 기계들은 풀 레벨(412)에 도시되는 기계들의 풀들로 그룹핑될 수 있다. 풀은 하나 이상의 작업 및/또는 워크 아이템과 관련된 가상 기계들을 포함할 수 있다. 실시예에 따라, 단일 풀이 다수의 클러스터, 예를 들어 데이터 센터 내의 모든 클러스터들 또는 복수의 데이터 센터에 걸치는 복수의 클러스터에 걸칠 수 있다. 단일 풀이 수백만 개와 같은 많은 수의 가상 기계를 포함할 수 있다. 가상 기계들은 예를 들어 수십억 개까지의 많은 수의 풀에 포함될 수 있다.

[0051] 자원 관리 계층(402)의 제3 레벨에서, 자원 관리 계층은 (후술되는) 태스크 테넌트들 각각의 크기를 관리한다. 이것은 시스템 전체가 시스템의 현재 부하 및 다른 응용들을 위한 미사용 가상 기계들의 시스템으로의 적시 복귀 해제와 같은 다수의 메트릭에 기초하여 사용되는 컴퓨팅 자원들의 양을 동적으로 조정하는 것을 가능하게 한다. 이 레벨은 테넌트 레벨(414)에 의해 도시된다.

[0052] 도 4에 도시된 실시예에서, 스케줄링 및 실행 계층(404)은 사용자에 의해 수행되고 있는 워크 아이템들, 작업들 및 태스크들을 관리한다. 스케줄링 및 실행 계층(404)은 스케줄링 결정들을 행하고, 작업들 및 태스크들의 런칭을 담당하는 것은 물론, 장애시에 재시도한다. 그러한 스케줄링 및 실행 계층(404)은 다양한 레벨들에서 작업들 및/또는 태스크들을 관리하기 위한 컴포넌트들을 포함할 수 있다.

[0053] 예시적인 컴포넌트들은 워크 아이템 및 작업 관리 컴포넌트(416), 태스크 관리 및 스케줄링 컴포넌트(418) 및 작업 관리자 태스크 컴포넌트(420)를 포함할 수 있다. 워크 아이템 및 작업 관리 컴포넌트(416)는 하나 이상의 사용자(또는 모든 사용자들)가 생성하고/시스템에 전달한 모든 워크 아이템들을 관리한다. 이어서, 활성 워크 아이템들 각각의 사양에 기초하여, 시스템은 태스크들이 제출되는 작업들 및 작업 큐들을 생성할 수 있다. 워크 아이템은 지정된 재발생 스케줄(예를 들어, 매일 오후 5시)을 가질 수 있다. 이어서, 시스템은 워크 아이템에 대한 재발생 스케줄에 따라 작업들을 생성할 수 있다. 워크 아이템 및 작업 관리 컴포넌트(416)는 시스템 내의 워크 아이템들 및 작업들의 종료 및 처분의 관리도 담당할 수 있다.

[0054] 태스크 관리 및 스케줄링 컴포넌트(418)는 시스템의 사용자가 태스크들을 시스템의 작업에 제출(예로서, 전달)하는 것을 가능하게 한다. 이 컴포넌트는 시스템 내의 모든 작업들/워크 아이템들에 걸쳐 태스크들을 스케줄링하면서 이러한 태스크들의 상태의 추적을 유지하는 것을 담당한다. 태스크 관리 및 스케줄링 컴포넌트(418)는 태스크 테넌트들(즉, 태스크 기계들) 중 하나/일부/전부에 걸쳐 분산되는 태스크 스케줄러들의 세트를 포함하여, 각각의 스케줄러가 자원들(예로서, 가상 기계들) 및 태스크들의 제한된 세트하고만 연관되는 것을 가능하게 할 수 있다. 결과적으로, 태스크 관리 및 스케줄링 컴포넌트(418)는 시스템이 예시적인 양태에서 다양한 태스크 테넌트들에 걸쳐 가상 기계들에서 실행되는 수십억 개의 활성 태스크를 지원하는 것을 가능하게 한다.

[0055] 작업 관리자 태스크 컴포넌트(420)는 각각의 작업과 옵션으로서 연관될 수 있는 JM 태스크가 작업의 최초 태스크로서 런칭되는 것을 가능하게 한다. 전술한 바와 같이, (태스크 작업 관리자로도 지칭되는) JM 태스크는 특정 작업 내의 태스크들의 단일 제어 포인트를 제공한다. 이것은 JM 태스크가 추가 태스크들을 작업에 제출하고, 이러한 태스크들의 진행을 모니터링하는 것을 가능하게 하며, 이는 JM 태스크가 작업을 종료할 시기

를 제어하는 것을 가능하게 한다. JM 태스크는 애플리케이션 런타임 계층(406)이 그들의 시스템들을 제어하고 실행하는 것을 돕는 메커니즘의 일례일 수 있다.

[0056] 전술한 계층들은 다수의 지리 위치에 프로세서들을 포함하는 분산 컴퓨팅 환경에서 구현될 수 있다. 도 5는 본 발명의 양태들에 따른, 단일 분산 컴퓨팅 시스템(500) 내에 통합될 수 있는 상이한 위치들의 프로세서들의 예시적인 배열을 도시하는 블록도를 나타낸다.

[0057] 도 5에서는, 하나 이상의 태스크 테넌트(514)를 이용하여 가상 기계들의 풀들을 관리할 수 있다. 태스크 테넌트(514)는 (도 9와 관련하여 더 상세히 후술되는 바와 같이) 가상 기계들의 세트를 유지할 수 있다. 하나 이상의 사용자의 작업들은 가상 기계들의 하나 이상의 풀의 일부인 태스크 테넌트(514) 내의 가상 기계들에서 실행될 수 있다. 하나 이상의 태스크 테넌트(514)가 주어진 지리 영역에서 사용될 수 있다. 태스크 테넌트(514)의 책임은 가상 기계들의 세트를 유지하고, 태스크 테넌트 내의 자원 사용에 기초하여 태스크 테넌트를 동적으로 확장 또는 축소하는 것을 포함할 수 있다. 이것은 태스크 테넌트(514)가 증가된 고객 요구를 수용하기 위해 태스크 테넌트 내의 가상 기계들의 수를 증가시키는 것을 가능하게 한다. 이것은 또한 태스크 테넌트(514)가 미사용 가상 기계들을 해제하는 것을 가능하게 하여 가상 기계들이 다른 고객들을 위한 서비스를 처리하는 데이터 센터 내의 다른 호스트되는 서비스들에 할당될 수 있게 한다. 태스크 테넌트(514)의 다른 책임은 풀 할당/할당 해제/관리 논리의 일부를 구현하는 것일 수 있다. 이것은 태스크 테넌트(514)가 고객을 위한 태스크와 관련된 풀들에 어떻게 가상 기계들을 할당할지를 결정하는 데 참여하는 것을 가능하게 한다. 태스크 테넌트(514)는 또한 태스크 테넌트 내의 가상 기계들에서의 태스크들의 스케줄링 및 실행을 담당할 수 있다.

[0058] 도 5에 도시된 실시예에서, (도 6-8과 관련하여 아래에서 더 상세히 설명되는 바와 같이) 주어진 지리 영역 내의 그리고/또는 다양한 지리 영역들에 걸치는 태스크 테넌트들(514) 모두를 제어하는 하나 이상의 태스크 위치 서비스(512)가 제공된다. 도 5에는, "지리 영역 1" 및 "지리 영역 2"로서 라벨링된 영역들을 서빙하는 태스크 위치 서비스들(512)이 도시된다. 태스크 위치 서비스(512)의 책임은 주어진 지리 영역에 대한 태스크 계정들의 관리를 포함할 수 있다. 태스크 위치 서비스들(512)은 사용자들이 분산 컴퓨팅 환경과 상호작용하는 것을 가능하게 하기 위한 애플리케이션 프로그래밍 인터페이스들(API들)도 제공할 수 있다. 그러한 API들은 가상 기계들의 풀들과 관련된 처리 API들, 풀 관리 논리, 및 주어진 지리 영역 내의 태스크 테넌트들에 걸치는 풀 관리 논리의 조정을 포함할 수 있다. API들은 또한 사용자에 의해 제출되는 태스크들을 처리하는 것은 물론, 사용자 태스크들과 관련된 워크 아이템들 또는 작업들을 유지, 스케줄링 및 종료하기 위한 API들을 포함할 수 있다. API들은 지리 영역 내의 모든 워크 아이템들, 작업들, 태스크들 및 풀들에 대한 통계 수집, 집계 및 보고를 위한 API들을 더 포함할 수 있다. 게다가, API들은 가상 기계들에 대한 스폿(spot) 시장에 기초하는 단기 방식으로 이용 가능 가상 기계들의 사용자들에 대한 선점 VM들로서의 경매를 허가하기 위한 API들을 포함할 수 있다. API들은 또한 사용을 측정하고 과금 보고를 제공하기 위한 API들을 포함할 수 있다.

[0059] 태스크 위치 서비스들(512)은 글로벌 위치 서비스(502)("XLS")에 의해 함께 링크될 수 있다. 글로벌 위치 서비스(502)는 계정 생성, 및 태스크 위치 서비스(512)와 연계하여 태스크 계정들을 관리하는 것을 포함하는 계정들의 관리를 담당할 수 있다. 예를 들어, 글로벌 위치 서비스는 데이터 센터 장애시에 장애 복구, 및 워크 아이템들 및 작업들의 가용성을 책임질 수 있다. 이것은 데이터 센터를 이용할 수 없음으로 인해 상이한 위치에서 워크 아이템 또는 작업을 실행하고, 고객들이 그들의 워크 아이템들, 작업들 및 풀들을 하나의 데이터 센터로부터 다른 데이터 센터로 이동시키는 것을 허가하는 것을 포함할 수 있다. 통상적으로, 시스템(500) 내에는 임의의 주어진 시간에 하나의 활성 글로벌 위치 서비스(502)만이 존재할 것이다. 이러한 활성 글로벌 위치 서비스(502)는 다양한 태스크 위치 서비스들(512)은 물론, 데이터 저장을 관리하기 위한 서비스 컴포넌트들(도시되지 않음)과 통신한다. 글로벌 위치 서비스는 예를 들어 도메인 이름 서버(504)에 글로벌 계정 이름 공간을 유지할 수 있다.

[0060] 도 5의 시스템의 동작의 일례로서, 가상의 고객 또는 사용자가 클라이언트 포털(506)을 이용하여 시스템(500)에 액세스하여, 글로벌 위치 서비스(502)에 의해 제공되는 인터페이스를 통해 태스크 계정을 생성할 수 있다. 이 예에서, 가상의 고객은 샐리로 지칭된다. 태스크 계정을 생성하기 위한 사용자 요청은 옵션으로서 계정 생성이 필요한 지리 영역을 지정할 수 있다. 이 예에서, 샐리는 지리 영역 2의 장애 조치 영역을 갖는 지리 영역 1과 관련된 계정을 요청한다. 이에 응답하여, 글로벌 위치 서비스(502)는 요청된 지리 영역(예로서, 지리 영역 1)에 대응하는 태스크 위치 서비스(512)와 접촉하여 계정을 생성한다. 영역이 요청되지 않는 경우, 태스크 계정은 예를 들어 요청 사용자 또는 이용 가능 자원들과 관련된 위치에 기초하여 임의의 전통적인 방법에 의해 선택되는 영역에서 생성될 수 있다. 태스크 위치 서비스(512)는 모든 계정들에 대한 모든 정보를 그의 지리 영역 내에 유지한다. 지리 영역 1에 대해 태스크 위치 서비스(512)에서 계정을 그리고 잠재적으로 지리 영역 2에서

장애 복구 사본을 성공적으로 생성한 후에, 글로벌 위치 서비스(502)는 지리 영역 1에 대한 태스크 위치 서비스(512)의 가상 IP 어드레스를 지시하기 위해 샐리의 계정에 대한 태스크 서비스 엔드포인트를 등록한다. 예를 들어, "sally.task.core.servicecompany.net"와 같은 호스트 이름을 지리 영역 1 내의 태스크 위치 서비스(512)의 가상 IP 어드레스에 맵핑하기 위해 도메인 이름 서비스(DNS) 레코드가 생성될 수 있다. 이것은 이 예시적인 양태에서 샐리에 대한 태스크 계정의 생성을 완료한다. 또한, 지리 영역 1에서의 장애시에 글로벌 위치 서비스가 지리 영역 2를 지시하도록 DNS 레코드를 갱신할 수 있는 것이 고려된다.

[0061] 계정이 생성된 후, 고객 샐리는 태스크 계정에 액세스하고, 분산 컴퓨팅 환경과 상호작용하기 위한 API들에 액세스하기 위한 요청들을 호스트 이름 "sally.task.core.servicecompany.net"를 향해 전송할 수 있다. 예를 들어, 샐리는 API에 액세스하여 새로운 워크 아이템 또는 태스크를 생성하기 위한 요청을 발할 수 있다. 이어서, DNS 서버는 호스트 이름을 분석할 수 있으며, 요청이 올바른 태스크 위치 서비스(512)로 라우팅될 것이다. 이 예에서, 요청은 지리 영역 1에 대한 태스크 위치 서비스(512)로 라우팅되며, 이 서비스는 요청을 처리하고, 요청된 워크 아이템, 작업 또는 태스크를 생성한다.

[0062] 임의 수의 클라이언트 포털들(506), 지리 영역들(508, 510), 태스크 위치 서비스들(512) 및/또는 태스크 테넌트들(514)이 본 발명의 예시적인 양태들에서 임의의 조합에서 구현될 수 있는 것이 고려된다. 예를 들어, 예시적인 양태에서 태스크 위치 서비스(512)가 수십, 수백 또는 수천 개의 태스크 테넌트와 관련될 수 있는 것이 고려된다.

[0063] 도 6은 본 발명의 양태들에 따른, 태스크 위치 서비스(602)에 대한 가능한 구성을 도시하는 블록도를 나타낸다. 도 6에 도시된 구성에서, 태스크 위치 서비스는 하나 이상의 계정 서버(612)를 포함할 수 있다. 계정 서버들은 주어진 지리 영역 내의 계정들에 대한 생성, 삭제 또는 특성 갱신을 포함하는 계정 관리를 처리한다. 계정 프론트엔드들(608)은 계정 서비스에 대한 프론트엔드 노드들로서 사용된다. 계정 프론트엔드들(608)은 도면에 도시된 바와 같이 계정 가상 IP 어드레스(604) 뒤에 위치한다. 계정 프론트엔드들(608)은 글로벌 위치 서비스로부터 오는 계정 API 요청들, 예를 들어 계정들을 생성하거나 계정들을 삭제하기 위한 API 요청들을 처리한다.

[0064] 도 6의 구성은 또한 하나 이상의 풀 서버(614)를 포함한다. 풀 서버(614)는 주어진 지리 영역 내의 가상 기계들의 풀들에 대한 풀 관리 및 풀 트랜잭션들을 처리한다. 풀 서버(614)는 풀 생성, 삭제 및 특성 갱신을 처리한다. 풀 서버(614)는 또한 다수의 태스크 테넌트에 걸쳐 상위 레벨 가상 기계 할당 알고리즘을 관리한다. 가상 기계 할당은 가상 기계와 주어진 사용자에 대한 저장의 접속을 고려할 수 있다. 풀 서버는 또한 가상 기계들의 할당과 관련된 다른 태스크들을 수행할 수 있다.

[0065] 또한, 풀 서버(614)는 자동 스케일링 컴포넌트(616)를 더 포함할 수 있는 것이 고려된다. 자동 스케일링 컴포넌트(616)는 풀에 대한 원하는 수의 전용, 대기 및/또는 선점 가상 기계를 자동 결정하는 사용자 제공 자동 스케일링 규칙들(예로서, 공식들)의 실행을 담당하는 모듈로서 형성될 수 있다. 즉, 자동 스케일링 컴포넌트는 주어진 풀과 관련된 자동 스케일링 규칙들에 기초하여 자동 스케일링 결정을 행하는 것을 담당할 수 있다. 자동 스케일링 컴포넌트(616)는 풀 또는 풀에 할당된 작업들에 관한 메트릭들을 수신하고, 그들과 사용자 제공(또는 시스템 제공) 규칙들을 함께 적용하여 풀에 대한 자동 스케일링 액션을 계산할 수 있다. 자동 스케일링 액션은 전용 가상 기계들의 수를 늘리거나 줄이고, 예약된 대기 가상 기계들의 수를 늘리거나 줄이고, 풀에 대한 선점 가상 기계들의 타겟 수를 늘리거나 줄이면서, 또한 그러한 자원들에 대한 입찰가를 갱신하는 것을 포함할 수 있다.

[0066] 도 6의 구성은 또한 (도 8과 관련하여 아래에서 더 상세히 설명되는 바와 같은) 하나 이상의 워크 아이템 또는 작업 스케줄러("WIJ")(618)를 포함한다. WIJ 스케줄러들(618)은 워크 아이템들 및 작업들의 생성, 삭제 및 갱신을 처리한다. WIJ는 또한 일 실시예에서 관련 워크 아이템에서 지정되는 스케줄에 기초하여 작업들을 생성하는 것은 물론, (하나가 추가되어야 하는 경우에) JM 태스크를 작업에 추가하는 것을 담당한다. 게다가, WIJ 스케줄러들(618)은 (사용자에 의한 전통적인 수동 생성과 달리) 자동 풀 구성이 요구될 때 워크 아이템들과 관련된 풀들의 생성 및 삭제를 개시할 수 있다. WIJ 스케줄러들(618)은 또한 태스크 위치 서비스 내에서 스케일링하기 위한 범용 파티셔닝 메커니즘들을 사용할 수 있다. 일 실시예에서, 각각의 태스크 위치 서비스 내에 다수의 WIJ 스케줄러(618)가 존재하며, WIJ 스케줄러들 각각은 소정 범위의 워크 아이템들을 처리한다.

[0067] 풀 서버들(6140 및 WIJ 스케줄러들(618)은 (도 7에서 더 상세히 후술하는 바와 같이) 태스크 위치 서비스 프론트엔드들(610)을 통해 사용자들로부터 요청들을 수신한다. 태스크 위치 서비스 프론트엔드들(610)은 또한 사용자들로부터의 요청들을 처리하기 위해 대응하는 컴포넌트들을 호출하는 것을 담당한다. 태스크 위치 서비스 프론트엔드들(610)은 도면에 도시된 바와 같이 계정 가상 IP 어드레스(606) 뒤에 위치한다.

[0068] 도 6의 구성은 태스크 위치 서비스 마스터(620)를 더 포함한다. 일 실시예에서, 태스크 위치 서비스 마스터(620)는 2개의 주요 책임을 갖는다. 먼저, 태스크 위치 서비스 마스터(620)는 태스크 위치 서비스(602) 내의 대응 서버들에 대한 파티셔닝 논리를 구현하기 위한 마스터 시스템으로 사용된다. 게다가, 태스크 위치 서비스 마스터(620)는 태스크 위치 서비스의 전체 지리 영역에 대한 각각의 스폿 기간의 시작에서 선점 가상 기계들에 대한 새로운 시장 가격을 계산하거나, 선점 기계들에 대한 입찰 요청들을 스폿 가격 결정을 담당하는 외부 서비스와 조정하는 것을 담당할 수 있다. 이것은 풀 서버들 및 태스크 테넌트들로부터 현재의 입찰들 및 자원 가용성 정보를 수집하고, 그에 따라 새로운 시장 가격을 계산하거나, 정보를 스폿 시장 서비스로 전달한다. 이것은 또한 지리 영역 내의 모든 태스크 테넌트들에 걸치는 선점 가상 기계들에 대해 풀 서버들에 대한 상위 레벨 할당 지도를 행한다.

[0069] 컴퓨팅 환경의 활동 및 거동을 추적하기 위해, 태스크 위치 서비스 마스터(620)는 하나 이상의 통계 집계 서버(622)와 통신할 수 있다. 통계 집계 서버들은 태스크들, 작업들, 워크 아이템들 및 풀들에 대한 상세한 통계들을 수집 및 집계하는 것을 담당한다. 시스템 내의 다른 컴포넌트들은 태스크들 및 가상 기계들에 대한 정밀한 통계들을 발한다. 통계 집계 서버들은 이러한 정밀한 통계들을 태스크 레벨 또는 가상 기계 레벨 통계들로부터 워크 아이템, 계정 레벨 및/또는 풀 레벨 통계들로 집계한다. 통계들은 API를 통해 사용을 위해 노출될 수 있다. 게다가, 통계 집계 서버들은 과금에 사용하기 위해 각각의 계정에 대한 시간별 측정 레코드들을 생성하는 것을 담당할 수 있다.

[0070] 도 7은 본 발명의 양태들에 따른 예시적인 태스크 위치 서비스 프론트엔드("TLSFE")(700)를 도시하는 블록도를 나타낸다. TLSFE(700)는 예시적인 양태에서 전술한 도 6의 태스크 위치 서비스 프론트엔드(610)와 유사할 수 있다.

[0071] TLSFE(700)는 요청 프로세서(702), 인증 및 허가 모듈(704), 계정 관리자 컴포넌트(706) 및 비즈니스 논리 모듈(708)을 포함한다. 대안 실시예들에서 추가적인 또는 대안적인 모듈들 및/또는 컴포넌트들이 포함될 수 있는 것이 고려된다.

[0072] 요청 프로세서(702)는 HTTP(S) 엔드포인트 상에서 사용자 요청을 수신 및 식별하고 승인하는 것을 담당하는 컴포넌트이다. 이어서, 요청 프로세서(702)는 각각의 요청을 큐잉하여, 인증 및 허가 모듈(704)로 전송한다. 인증 및 허가 모듈(704)은 사용자 요청 인증 및 허가를 담당한다. TLSFE는 예시적인 양태에서 들어오는 요청들을 인증하기 위해 공유 키 인증을 이용한다. 다른 인증 형태들도 고려된다. 또한, 인증 및 허가 모듈(704)은 계정 관리자 컴포넌트(706)와 상호작용하여, 사용자 계정에 대한 정보 및 키 정보를 획득한다. 이어서, 인증 및 허가 모듈(704)은 사용자가 동작들의 수행을 요청하는 것이 허가되는 곳을 결정할 수 있다(예를 들어, 계정은 요청시에 일시적으로 디스에이블될 수 있다).

[0073] 계정 관리자 컴포넌트(706)는 저장/액세스 계정 정보를 캡슐화하는 데 사용될 수 있다. 계정 관리자 컴포넌트(706)는 주문시에, 예시적인 양태에서 요청들이 수신될 때, 계정 정보를 로딩하며, 이는 국지적으로(메모리 내에) 정보를 캐싱하는 것을 가능하게 하여, 빈번히 액세스되는 계정들의 처리를 고속화한다. 계정 특성(예로서, 디스에이블 상태, 인증 키)이 변할 때, 계정 테이블이 갱신될 수 있다. TLSFE에서 실행되는 계정 관리자 컴포넌트(706)는 구성 가능 인터벌일 수 있는 (존재할 경우) 캐싱된 사본의 만료 시에 그러한 변경을 알 수 있다.

[0074] 비즈니스 논리 모듈(708)은 사용자에 의해 행해진 요청이 인증 및 허가 모듈(704)을 통과하면 그와 관련된 모든 비즈니스 논리를 처리한다. 비즈니스 논리 모듈(708)은 또한 분산 컴퓨팅 시스템 내의 다른 태스크 컴포넌트들과 상호작용할 수 있다. 게다가, 비즈니스 논리 모듈(708)은 완료된 작업 관리자 태스크들에 대한 JM 태스크 완료 큐를 모니터링하며, 이는 또한 태스크들의 완료시에 통지가 도 6의 WIJ 스케줄러(618)로 전달되는 것을 가능하게 하는 것이 고려된다.

[0075] 도 8은 본 발명의 양태들에 따른 예시적인 워크 아이템/작업 스케줄러("WIJ")(800)를 도시하는 블록도를 나타낸다. WIJ(800)는 도 6의 예시적인 WIJ 스케줄러(618)일 수 있다.

[0076] WIJ(800)는 요청 프로세서(802), 캐싱 모듈(804), 하트비트 모듈(806) 및 스케줄링 모듈(808)을 포함한다. WIJ(800)는 전술한 바와 같이 사용자 계정에 의해 제출된 워크 아이템들을 승인하고, 워크 아이템 스케줄 내의 적절한 시간들에 작업들을 스케줄링하는 것을 담당하는 역할일 수 있다. 결과적으로, WIJ(800)는 워크 아이템에 대한 작업들을 생성하고, 각각의 생성된 새로운 작업에 대한 큐들을 생성하고, 이어서 이 큐들은 작업에 대한 태스크들을 큐잉하는 데 사용되는 것이 고려된다. 이어서, (도 9와 관련하여 후술하는) 태스크 테넌트의 태스크 스케줄러가 큐로부터 태스크들을 인출하고, 이들을 태스크 테넌트들의 가상 기계들 상에 스케줄링할 수 있

다. 또한, WIJ는 완료된 워크 아이템/작업을 마킹하고, 적절한 태스크 테넌트들과 접촉하여 작업을 개시하는 것과 같이 워크 아이템 및 관련 작업들의 수명 관리를 담당하는 것이 고려된다.

[0077] 요청 프로세서(802)는 (도 6의 태스크 위치 서비스 프론트엔드(610)와 같은) TLSFE로부터 수신되는 요청들과 관련된 다양한 워크 아이템들/작업들의 처리를 담당하는 모듈이다. 게다가, 계정이 삭제될 때, 계정 서버(예로서, 도 6의 계정 서버(612))가 요청들을 WIJ(800)로 전달하는 것이 고려된다.

[0078] 캐싱 모듈(804)은 WIJ가 서비스하고 있는 파티션, 사용자 계정, 활성 워크 아이템, 활성 작업, 태스크 테넌트 리스트 등의 메모리내 캐시이다. 이 모듈은 관련 캐싱 정책들을 위해 캐시를 최신 상태로 유지하는 것을 담당할 수 있다.

[0079] 하트비트 모듈(806)은 태스크 위치 서비스 마스터(예로서, 도 6의 TLSM(620))와 협력하여 WIJ의 건강 및 부하 상태를 보고한다. 게다가, 하트비트 모듈(806)은 또한 "하트비트" 협력을 통해 TLSM으로부터 새로운 파티션 할당들을 수신하는 것이 고려된다.

[0080] 스케줄링 모듈(808)은 워크 아이템들에 대한 새로운 작업들을 해당 워크 아이템의 지정된 스케줄에 따라 스케줄링하는 것을 담당한다. 이것은 또한 워크 아이템 및/또는 작업을 완료된 것으로서 마킹하는 것을 담당한다. 또한, 스케줄링 모듈(808)은 워크 아이템 또는 작업의 수명과 관련된 수명을 갖는 풀들을 생성 및 삭제하는 것을 담당할 수 있다.

[0081] 도 9는 본 발명의 양태들에 따른 태스크 테넌트(900)를 도시하는 블록도를 나타낸다. 예시적인 양태에서, 태스크 테넌트는 도 5의 시스템(500)에서 태스크 테넌트(514)로서 구현되는 것으로 고려된다. 전술한 바와 같이, 태스크 테넌트는 가상 기계들의 풀들의 관리를 지원할 수 있다. 도 9에 도시된 실시예에서, 태스크 테넌트는 하나 이상의 태스크 테넌트 프론트엔드(904)를 포함한다. 태스크 테넌트 프론트엔드들(904)은 태스크 위치 서비스와 태스크 테넌트 사이에 요청들을 통해 전달되는 것을 포함하는, 태스크 테넌트와 그의 대응하는 태스크 위치 서비스 사이의 통신을 위해 내부적으로 사용되는 태스크 테넌트 IP 가상 어드레스 뒤에 위치한다.

[0082] 도 9에 도시된 실시예에서, 태스크 테넌트는 또한 태스크 스케줄러(902)를 포함한다. 태스크 스케줄러(902)는 태스크 테넌트 내에서 로컬 태스크 스케줄링 결정들을 행하는 것을 담당할 수 있다. 태스크 스케줄러(902)는 그가 제어하는 각각의 가상 기계에서 어떤 태스크를 실행할지를 결정한다. 예를 들어, 사용자에 의해 제출된 워크 아이템 또는 작업은 스케줄링될 태스크들의 리스트를 포함하는 큐들의 세트를 가질 수 있다. 태스크 스케줄러(902)는 큐들의 세트로부터 태스크들을 취하고, 작업과 관련된 풀 내에서 하나 이상의 이용 가능 가상 기계를 선택하고, 가상 기계(들)와 접촉하여 이러한 태스크들을 스케줄링한다. 태스크 스케줄러(902)는 또한 작업들과 관련된 우선 순위 값들에 기초하여 스케줄링 결정들을 행할 수 있다. 게다가, 태스크 스케줄러(902)는 태스크 테넌트 내의 가상 기계들의 추적을 유지한다.

[0083] 태스크 스케줄러(902)는 풀 서버들과 협력하여 가상 기계들을 풀들에 대해 할당/할당 해제한다. 게다가, 태스크 스케줄러(902)는 모든 가상 기계들에 관한 하트비트들을 유지하고, 하트비트들을 통해 풀 멤버십에 대해 가상 기계와 동기화하며, 가상 기계들의 재개/리이미지(reimage)를 제어한다. 태스크 스케줄러(902)의 또 다른 기능은 태스크 테넌트의 크기의 추적을 유지하는 것일 수 있다. 태스크 테넌트 내의 가상 기계들의 현재 사용에 기초하여, 태스크 스케줄러는 태스크 테넌트를 확장 또는 축소할 수 있으며, 따라서 태스크 테넌트는 태스크 테넌트와 관련된 태스크들을 실행하기에 충분한 수의 가상 기계를 갖는다. 유사하게, 태스크 테넌트 내에 유휴 상태로 있는 너무 많은 가상 기계가 존재하는 경우, 기계들은 데이터 센터 내의 다른 호스트되는 서비스들에 의한 사용을 위해 해제될 수 있다.

[0084] 태스크 스케줄러(902)는 전술한 기능을 달성하기 위해 아래의 컴포넌트들 및 모듈들을 포함한다. 예를 들어, 태스크 스케줄러는 요청 프로세서(910), 하트비트 관리자(912), 통계 집계 관리자(914), 스케줄링 루프 모듈(916), 풀 할당 관리자(918) 및 태스크 테넌트 관리자(920)를 포함할 수 있다.

[0085] 예시적인 양태에서, 태스크 스케줄러(902)는 태스크 테넌트(900)의 가상 기계들을 "소유"하고, 또한 그러한 가상 기계들에서 수행되는 것을 "소유"한다. 결과적으로, VM들의 풀이 다수의 태스크 테넌트에 걸치는 VM들을 포함하는 것이 고려된다. 작업이 풀에 할당될 때, 태스크 스케줄러(902)는 큐로부터 워크 아이템들을 취하고, 태스크 스케줄러(902)가 "소유"하는 가상 기계들(예로서, 태스크 테넌트(900)의 TVM(908))에서 그러한 워크 아이템들을 실행할 수 있다. 풀 내의 TVM들을 갖는 다른 태스크 테넌트들과 관련된 다른 태스크 스케줄러들도 큐로부터 워크 아이템들을 취하고, 각각의 태스크 테넌트의 고유 태스크 스케줄러들 각각과 관련된 가상 기계들에서 그러한 워크 아이템을 실행할 수 있다.

[0086] 요청 프로세서(910)는 WIJ, 풀 서버 및/또는 TLSM으로부터 수신된 다양한 요청들을 처리하는 것을 담당한다. 요청들은 태스크 테넌트 프론트엔드로부터 태스크 스케줄러로 라우팅될 수 있다. 결과적으로, 요청 프로세서(910)는 소비할 올바른 서브컴포넌트들에 대한 요청들을 큐잉할 수 있다.

[0087] 하트비트 관리자(912)는 태스크 테넌트 내의 다른 가상 기계들(예로서, TVM(908))에 관한 하트비팅을 담당한다. 하트비트 통신에 응답하여, 이 모듈은 가상 기계 건강 및 태스크 건강에 관한 정보를 수집한다. 게다가, 하트비트 메시지들의 수신에 응답하여, 이 모듈은 가상 기계들 중 어느 것이 유휴 상태에 있고 새로운 태스크들을 스케줄링하는 데 사용될 수 있는지를 결정할 수 있다. 게다가, 하트비트 모니터는 가상 기계들에 대한 다양한 통계들(예로서, CPU, 메모리, 디스크 사용)을 수집할 수 있다. 이어서, 이러한 통계들은 통계 집계 관리자(914)로 전달될 수 있다.

[0088] 통계 집계 관리자(914)는 태스크 테넌트의 다양한 가상 기계들로부터 수집되는 다양한 통계들을 풀에 의해 집계하고 구성되는 것을 담당한다. 이어서, 집계된 통계들은 자동 스케일링 동작들에서 사용하기 위해 풀 서버로 전달될 수 있다.

[0089] 스케줄링 루프 모듈(916)은 풀 가상 기계들 상에 작업 태스크들을 스케줄링하는 것을 담당할 수 있다. 풀 할당 관리자(918)는 풀과 관련된 동작들, 예를 들어 본 명세서에서 설명되는 바와 같은 자원 스케일링, 자원 할당, 작업들/워크 아이템들의 할당 등을 담당한다.

[0090] 태스크 테넌트 관리자(920)는 태스크 테넌트 자체의 확장 및 축소를 돌보는 모듈이다. 이 모듈은 분산 컴퓨팅 시스템 패브릭과 상호작용하여, 시스템의 부하에 따라 태스크 테넌트 가상 기계들의 수를 확장/축소한다. 게다가, 태스크 테넌트 관리자(920)는 임의의 주어진 풀에 대한 전용 가상 기계들로의 빠른 변환을 위해 자유로운 리이미징된 가상 기계들의 버퍼를 유지하는 것을 담당할 수 있다.

[0091] 도 9는 또한 태스크 테넌트와 관련된 복수의 가상 기계를 도시한다. 도 9에 도시된 실시예에서, 가상 기계들 각각은 태스크 가상 기계(908)(TVM)를 포함한다. 일 실시예에서, 태스크 가상 기계(908)는 가상 기계 상의 태스크들을 런칭하는 것은 물론, 태스크들에 대한 디렉토리 구조들 및 허가들을 설정하는 것을 담당한다. 이것은 또한 가상 기계 상에 운영 체제 방화벽을 구성하여, (풀이 인트라 통신을 요구하는 경우에) 동일 풀 내의 가상 기계들 사이의 트래픽만을 허용한다. 전술한 바와 같이, 태스크 스케줄러(902)는 태스크 가상 기계들(908)을 통해 가상 기계들에 관한 하트비트들을 유지한다. 이것은 태스크 스케줄러(902)가 가상 기계들의 건강을 모니터링하는 것은 물론, 태스크 가상 기계 에이전트들에 대한 풀 멤버십 정보를 동기화하는 것을 가능하게 한다.

[0092] 예시적인 태스크 가상 기계(906)는 간소화를 위해 TVM(908)에는 도시되지 않은 다수의 컴포넌트를 포함하는 것으로 도시된다. 그러나, 임의의 모듈들/컴포넌트들이 임의의 가상 기계와 관련될 수 있는 것이 고려된다. 태스크 가상 기계(906)의 컴포넌트들/모듈들은 요청 프로세서(922), 자원 관리자(924), 태스크 관리자(926) 및 보안 관리자(928)를 포함한다.

[0093] 요청 프로세서(922)는 가상 기계가 태스크 스케줄러 또는 태스크 테넌트 프론트엔드로부터 획득한 다양한 요청들을 처리하는 것을 담당한다. 자원 관리자(924)는 디스크 할당을 관리하고, 시스템에 대한 디렉토리 구조를 생성하고, 시동 태스크 및 작업 태스크를 위한 자원들을 다운로딩하는 것을 담당한다. 태스크 관리자(926)는 가상 기계가 태스크 스케줄러로부터 태스크 시작 명령을 수신하는 시간으로부터 시작되고 시스템 데이터(예로서, 관련 디렉토리 구조)가 가상 기계로부터 제거될 때까지 유효한 태스크 수명을 관리한다. 보안 관리자(928)는 다양한 방화벽 규칙들을 설정하고, 사용자 태스크를 실행하기 위한 올바른 특권들을 갖는 계정을 생성하는 것을 담당한다.

[0094] 도 10은 본 발명의 양태들에 따른 예시적인 풀 서버(1000)를 도시하는 블록도를 나타낸다. 풀 서버(1000)는 예시적인 양태에서 전술한 도 6의 풀 서버(614)로서 구현될 수 있다.

[0095] 풀 서버(1000)는 아래의 컴포넌트들을 포함한다. 들어오는 동작들(예로서, 풀 생성, 풀 삭제 및 풀 갱신)의 승인을 담당하는 요청 프로세서 모듈(1002). 풀 내의 가상 기계들을 예약하기 위해 태스크 테넌트들에 걸쳐 가상 기계 파괴를 수행하는 태스크 테넌트 가상 기계 할당 관리자 모듈(1004). 결정 모듈은 어느 태스크 테넌트들을 선택할지 그리고 풀을 위해 태스크 테넌트 내의 얼마나 많은 가상 기계를 예약할지를 결정한다. 트랜잭션 프로세서 모듈(1006)은 또한 풀 서버와 관련될 수 있다. 트랜잭션 프로세서 모듈은 풀 트랜잭션의 수명을 유지하는 코어 모듈이다. 이 모듈은 트랜잭션이 성공적으로 종료되거나 타임아웃되거나 취소될 때까지 트랜잭션들을 계속 처리한다. 각각의 트랜잭션은 테이블들 내에 유지되며, 따라서 장애시에 다양한 시스템 컴포넌트들을 통해 완료될 수 있다. 예시적인 트랜잭션은 주어진 풀에 대해 태스크 테넌트 내의 소정 수의 VM을 할당, 예약 또는

할당 해제하기 위한 풀 서버로부터 태스크 테넌트로의 요청들을 포함할 수 있다. 또한, 자동 스케일링 모듈(1008)도 풀 서버(1000)와 관련될 수 있는 것이 고려된다. 도 6의 자동 스케일링 컴포넌트(616)와 관련하여 전술한 바와 같이, 자동 스케일링 모듈(1008)은 풀에 대한 원하는 수의 전용, 대기 및/또는 선점 가상 기계를 자동으로 결정하는 사용자 제공 자동 스케일링 규칙들(예로서, 공식들)을 실행하는 것을 담당한다.

[0096] 워크 아이템들/작업들/태스크들의 관리

[0097] 도 11은 본 발명의 양태들에 따른, 예시적인 분산 컴퓨팅 시스템 내의 다양한 컴포넌트들 사이의 예시적인 워크플로우(1100)를 제공하는 통신 도면을 나타낸다. 분산 컴퓨팅 시스템은 클라이언트 포털(1102), 태스크 위치 서비스 프론트엔드(TLSFE)(1104), 풀 서버(1106), 워크 아이템/작업 스케줄러(WIJ)(1108), 저장 유닛(1112) 및 태스크 테넌트(1114)를 포함한다. 컴포넌트들 모드는 이전에 설명되었다. TLSFE(1104), 풀 서버(1106) 및 WIJ(1108)는 (도 6의 TLS(600)와 같은) 태스크 위치 서비스(1110)와 관련된다.

[0098] 워크 플로우는 사용자가 워크 아이템을 클라이언트 포털(1102)을 통해 TLSFE(1104)로 제출하는 것을 포함한다(1116). 이러한 제출은 TLSFE에 대한 API 호출을 이용하여 행해질 수 있다. 이어서, TLSFE는 수신된 워크 아이템을 시스템에 등록하기 위해 WIJ(1108)와 접촉할 수 있다(1118). 워크 아이템의 등록 동안, 워크 아이템 태스크들 및 옵션인 작업 관리자 태스크를 위해 사용할 풀이 지정될 수 있다. 이러한 등록은 이 예시적인 양태에서 워크 아이템의 지속(1120)을 위한 WIJ(1108)로부터 저장 유닛(1112)으로의 통신을 유발할 수 있다. 또한, WIJ(1108)는 저장 유닛(1112)에 저장(1122)되는 관련 작업을 생성하는 것이 고려된다.

[0099] 작업이 자동 풀과 관련되어야 하므로, WIJ(1108)는 워크 아이템의 태스크들이 처리될 풀을 생성하도록 풀 서버(1106)에 지시할 수 있다. 이러한 워크 플로우는 워크 아이템이 할당되는 풀이 아직 존재하지 않는다는 가정하에 동작한다. 이 예에서, 사용자에 의해 제출되는 워크 아이템은 그것이 자동 풀 실시예임을 지시할 수 있다. 자동 풀 설정이 포함됨을 지시하는 워크 아이템이 수신될 때, 풀 서버(1106)는 풀을 동적으로, 자동으로 생성할 수 있다(1124). WIJ는 풀이 생성/삭제되는 시기를 제어할 수 있으며, WIJ는 그에 따라 풀 서버에 지시할 수 있다. 이어서, 풀 서버는 WIJ 요청들을 다른 사용자 개시 풀 생성/삭제 요청들과 동일하게 처리할 수 있다. 전통적으로, 풀의 생성은 사용자 개입을 필요로 하며, 이는 풀이 동적으로, 자동으로 생성되는 것을 방해한다.

[0100] WIJ(1108)에 의해 생성된 각각의 작업에 대해, 다수의 큐가 형성될 수 있다. 예를 들어, WIJ는 전용 큐, 선점 큐, "임의의 VM" 큐 및/또는 작업 관리자 큐를 포함하는 여러 개의 작업 큐를 생성할 수 있다(1126). 이러한 다양한 큐들은 작업에 대한 태스크들을 유지하는 데 사용된다.

[0101] 워크 아이템이 관련 작업 마스터 태스크를 갖는 경우, WIJ는 작업 관리자 태스크를 작업과 함께 시작되는 제1 태스크로서 갖기 위해 작업 관리자 태스크를 시스템에 등록할 수 있다. 게다가, 정보가 작업 관리자 큐에 추가될 수 있다. 이어서, 시스템은 시스템 장애시에 작업 관리자가 항상 작업과 함께 실행되는 것을 보증할 수 있다. 그러한 통신들은 WIJ가 작업 관리자를 추가하고(1128) 작업 관리자 태스크를 추가(1130)하는 것에 의해 지시된다.

[0102] WIJ가 작업을 생성하면, 작업에 대한 추가 태스크들이 API를 통해 제출될 수 있다. 시스템은 각각의 제출된 태스크에 대한 재시도 카운트의 지정을 지원한다. 태스크가 실패하는 경우, 시스템은 아마도 상이한 가상 기계들상에서 재시도 카운트 횟수까지 태스크를 재생성할 수 있다. 작업이 JM 태스크를 갖는 경우, 작업에 대한 추가 태스크들이 작업 관리자 태스크 자체에 의해 제출될 수 있다. 시스템은 작업이 생성된 후에 시스템 밖으로부터 태스크들이 제출되는 것을 가능하게 할 수 있다.

[0103] TLSFE는 모든 태스크 추가 요청들을 처리하며, 태스크 추가 요청의 수신시에 TLSFE는 태스크 정보를 태스크 테이블 내에 유지하고, 또한 태스크에 대한 메타데이터를 3개의 작업 큐 중 하나 내에 넣는다. 어느 작업 큐를 선택할지는 태스크가 실행되도록 마킹된 곳(예를 들어, 풀 내의 오직 전용, 오직 선점 또는 임의의 VM)에 의존할 수 있다.

[0104] 작업이 생성되면, WIJ는 주어진 풀에 대한 가상 기계들을 갖는 모든 태스크 스케줄러들과 접촉하고, 작업 시작 통신(1132)에서 지시되는 바와 같이 그들에게 작업 시작에 대해 알린다. 사용자들은 워크 아이템 생성 동안 작업 관리자 태스크가 완료될 때 시스템이 전체 작업을 종료해야 하는지를 지정할 수 있다. 이어서, 시스템은 이러한 정보를 유지할 수 있으며, 그렇게 지정되는 경우, JM 태스크들이 완료될 때, 다음의 논리가 구현될 수 있다. WIJ는 태스크 테넌트(들)(1114)가 작업 큐 내의 작업들을 처리한 후에 발생할 수 있는 JM 태스크 완료에 대한 통신을 기다릴 수 있다. 작업 관리자 태스크가 완료될 때, 메시지가 JM 완료 큐 내에 인큐잉된다. 예시적인 양태에서, JM 큐는 시스템 큐이다. TLSFE는 큐를 계속 모니터링할 수 있으며, 큐 내의 메시지를 획득할

때, 메시지를 디큐잉하고, 적절한 WIJ에 작업 관리자 완료를 처리하도록 알린다. 큐로부터 작업 관리자 메시지가 수신된 후, WIJ는 그의 영구 저장소 내에 작업을 완료된 것으로서 마킹할 수 있으며, JM 완료 큐로부터 메시지를 제거한다.

[0105] 작업이 관련 작업 관리자 태스크를 갖지 않거나, 사용자들이 작업 관리자 태스크의 완료와 함께 작업을 종료하도록 지정하지 않는 경우, 고객에 의한 별개의 작업 종료 요청을 발하여, 작업을 완료된 것으로서 마킹할 수 있다. 작업이 완료된 것으로서 마킹되면, 작업에 대한 어떠한 추가 태스크도 처리되지 않을 수 있다. 또한, 작업이 재발생 또는 한 번 실행으로서 마킹되는 것이 고려된다. 재발생 작업들에 대해, WIJ는 다음 재발생 인터벌에 새로운 작업들을 생성(예를 들어, 인스턴스화, 산출)할 수 있다. 워크 아이템은 "이후에는 실행되지 않음" 시간이 지정되게 할 수 있으며, 이는 WIJ가 그 시간 후에 작업을 재생성하는 것을 방지할 것이다. WIJ로부터 작업 시작의 통신을 수신할 때, 태스크 스케줄러는 태스크들의 존재에 대해 작업 큐들의 조사를 시작한다. 예시적인 양태에서, 추가 태스크들이 큐잉될 때, 태스크 스케줄러는 태스크들을 디큐잉하고, 이들을 지정된 풀 가상 기계들에서 실행한다.

[0106] 태스크 스케줄러 스케줄링

[0107] (도 9의 태스크 스케줄러(902)와 같은) 태스크 스케줄러("TS")는 아래의 예시적인 방식으로 태스크들의 스케줄링을 수행할 수 있다. TS는 스케줄링 정보의 대부분을 메모리 내에 유지할 수 있다. 그러나, 작업 큐(들)로부터 아직 선택되지 않은 태스크들과 관련된 정보가 메모리 내에 유지되지 않을 수 있는 것이 고려된다.

[0108] 예시적인 양태에서, TS는 풀에 의한 스케줄링을 수행한다. 따라서, TS는 작업 태스크들이 스케줄링될 수 있는 풀들 각각에 대한 정보를 유지한다. 이러한 정보는 WIJ에 의해 작업 시작 메시지(예로서, 도 11의 작업 시작 통신(1132))를 통해 TS로 전달될 수 있다. 예시적인 양태에서, TS는 장애 조치 시나리오의 가능성을 처리하기 위해 이러한 정보를 유지할 수 있다. 각각의 풀에 대해, 다수의 활성 작업이 존재할 수 있다. TS는 작업 우선순위에 기초하여 각각의 풀 내에서 스케줄링을 행한다. 그러나, 우선순위는 예시적인 양태에서 오버아칭(overarching) 계정 레벨이 아니라 작업 레벨에 있다.

[0109] 예시적인 양태에서, 각각의 TS는 할당된 작업들의 리스트를 안다. 각각의 스케줄링 양자에 대해, TS는 다양한 작업 큐들로부터 "N"개의 아이템을 선택하며(예를 들어, 전용 및 선점 태스크들과 같은 상이한 타입의 태스크들을 스케줄링하기 위한 상이한 우선순위들을 갖는 작업마다의 다수의 작업 큐가 존재할 수 있으며), 각각의 작업 큐는 우선순위 큐일 수 있다. 이 예에서, 'N'개의 아이템은 태스크들을 스케줄링하기 위해 풀에서 이용 가능한 VM의 수와 동일할 수 있다. 즉, TS는 풀 내의 이용 가능한 가상 기계들의 수와 동일한 수의 아이템을 큐들로부터 선택하는 것이 고려된다.

[0110] 태스크에 대한 다른 메타데이터에 더하여, 각각의 큐 엔트리는 메인 태스크 테이블 내로의 포인터를 가질 수 있으며, 이는 TS가 태스크 테이블로부터 태스크에 대한 정보를 판독하는 것을 가능하게 한다. 태스크 테이블은 태스크 상태를 가질 수 있으며, 이는 TS가 태스크의 실행이 필요한지를 결정하는 것을 도울 수 있다. 예를 들어, 태스크가 완료된 경우, 태스크는 다시 실행될 필요가 없을 수 있다. 이것은 태스크가 완료된 후에 장애가 존재하지만 VM이 작업 큐로부터 엔트리를 제거할 수 없을 때 발생할 수 있다. 게다가, 태스크에 대한 친화도 정보가 태스크 테이블로부터 판독될 수 있는 것이 고려된다. 이러한 친화도는 TS가 큐로부터 선택된 'N'개의 아이템 중 어느 것이 특정 가상 기계에 가장 적합한지를 결정하는 것을 가능하게 할 수 있다.

[0111] 태스크가 VM에 할당되면, TS는 태스크에 대응하는 태스크 테이블 내에 가상 기계 정보를 넣을 수 있고, 가상 기계 상에 태스크를 스케줄링한다. 가상 기계가 태스크를 할당받으면, 가상 기계는 그의 작업 큐 내의 실행 태스크의 비가시 시간의 연장을 차용할 수 있으며, 이는 태스크에 대한 차용을 효과적으로 유지한다. VM은 태스크를 수신하면 태스크의 상태를 "실행"(또는 비교 가능)으로 갱신할 수 있다. 태스크에 대한 자원들이 다운로드되면, VM은 태스크 상태를 "실행"(또는 비교 가능)으로 갱신할 수 있고, 태스크 프로세스를 생성한다. 결과적으로, VM은 작업 큐로부터 태스크에 대한 대응 엔트리를 삭제할 수 있다. 태스크가 JM 태스크(작업 마스터 태스크)인 경우, VM은 JM 완료 큐 내에 작업 관리자 완료 엔트리를 포함할 수 있다.

[0112] 태스크 재시도 처리 및 작업 관리자 태스크 재시도 처리

[0113] 일반적으로, 태스크 재시도 및 JM 태스크 재시도 처리는 유사하게 수행되지만, 예시적인 양태에서 프로세스들에는 약간의 차이가 존재한다. 예시적인 양태에서, 각각의 제출된 태스크는 태스크에 대해 지정된 재시도 카운트와 관련된다. 태스크의 실행에 대한 애플리케이션의 장애시에, 시스템은 태스크를 에러에 의해 종료된 것으로 마킹하기 전에 재시도 카운트 값과 동일한 횟수만큼 태스크를 다시 스케줄링할 수 있다. 시스템 장애(예로서,

VM 장애)의 상황들에서, 시스템은 재시도 카운트를 향해 카운트하지 않고서 실패한 태스크를 자동으로 재시도할 수 있다.

[0114] 가상 기계는 그가 생성한 태스크를 모니터링한다. 태스크가 실패하는 경우, VM은 VM 상에서 태스크를 재시도하며, 또한 태스크 테이블에서 재시도 카운트를 갱신한다. 예시적인 양태에서, VM은 태스크를 (구성 가능한) "x"번 재시도하며, 그 수를 초과한 후에 VM은 태스크를 큐 내에서 다시 보이게 함으로써(예로서, 태스크에 대한 차용을 해제함으로써) 태스크 스케줄러가 대안 VM에서 태스크를 다시 스케줄링하는 것을 가능하게 한다. 이어서, 태스크 스케줄러는 큐로부터 가시 아이템을 선택할 수 있고, 재시도 카운트를 갱신하며, 그를 다른 VM에 할당한다. 이 예에서, 태스크는 이용 가능한 TM이 존재하는 경우에만 선택될 수 있다. 이러한 프로세스는 태스크가 성공적으로 완료되거나 태스크가 지정된 횟수만큼 재시도될 때까지 계속될 수 있다.

[0115] JM 태스크는 전술한 것과 유사한 프로세스를 따를 수 있다. 그러나, VM이 JM 태스크를 생성하는 데 사용될 수 없는 경우, 태스크 스케줄러는 논(non)-JM 태스크들(예를 들어, 런타임에서 최저 우선순위를 가짐) 중 하나를 선택할 수 있으며, 태스크를 선점하여 자원들이 JM 태스크의 실행에 사용될 수 있게 한다. 이것은 시스템 장애 시에도 JM 태스크가 작업을 위해 항상 재개되는 것을 보증할 수 있다.

[0116] 풀 관리

[0117] 각각의 태스크 계정에 대해, 하나 이상의 풀이 생성(또는 연관)될 수 있다. 예시적인 양태에서, 풀 내에는 두 가지 타입의 가상 기계들 및 또한 (전술한 바와 같은) 풀과 관련된 대기 VM 예약이 존재한다. VM들은 태스크 계정 또는 풀에 의해 독점적 이익을 위해 현재 사용되고 있는 예약 VM들인 전용 VM들일 수 있다. 제2 VM은 선점 VM일 수 있다. 선점 VM은 선점 태스크들을 실행하는 데 사용될 수 있는 시스템 내의 임의의 비전용 VM들일 수 있으며, 대기 VM들로서 예약된 VM들 또는 시스템이 제공한 임의의 유휴 VM들을 포함할 수 있다. 선점 VM들은 시스템이 다른 목적들을 위해 그의 자원들을 요구할 때까지 사용될 수 있다.

[0118] 예시적인 양태에서, 계정은 풀 내의 전용 VM들 및 대기 예약 VM들의 각각의 수를 지정할 수 있다. 대기 VM 예약은 전용 VM으로 변환될 수 있으며, 그 반대도 고려된다. 이어서, 시스템은 특정 풀에 대한 전용 및 대기 자원들을 유지할 수 있다.

[0119] 다양한 특성들이 풀과 연관될 수 있다. 그러한 특성들은 대기 VM 예약, 전용 VM들의 수, VM의 타입(예로서, 소형 또는 대형), 통신 요건(작업들이 크로스-태스크 통신을 요구함), 저장 계정 친화도, 풀 메타데이터(예로서, 사용자에 의해 설정된 메타데이터) 및/또는 시작 태스크 정보를 포함할 수 있다. 시작 태스크 정보는 풀의 초기 설정 동안 그리고 또한 VM이 리부팅될 때 VM들의 풀들 각각에서 실행되는 태스크이다. 이러한 특성들은 풀 및 풀의 자원을 적어도 부분적으로 정의할 수 있다.

[0120] 전술한 바와 같이, 작업은 풀(및 기본 가상 기계들)을 이용하여 작업의 태스크들을 실행한다. 태스크들은 풀 내의 전용 또는 선점 VM들에서 실행된다. 대기 VM 예약은 태스크들을 직접 실행하는 데 사용되지 않으며, 대신에 대기 VM들은 일 실시예에서 전용 또는 선점 VM들로 변환된다. 다양한 VM들에 대한 가격이 변할 수 있는 것이 고려된다. 예를 들어, 대기 VM 예약은 자원들을 예약하기 위한 비용에 있어서 최소일 수 있지만, 예약에 의해 대기 VM 예약은 전용 또는 선점 자원으로 빠르게 변환되는 데 사용될 수 있다. 전용 VM은 전통적인 컴퓨팅 가격을 가질 수 있다. 반면, 선점 VM들은 스폿 가격 결정 또는 다른 가변적인 가격 결정 구조들의 개념을 허용하는 입찰 동작에 의해 가격이 결정될 수 있다.

[0121] 풀의 생성은 아래의 예시적인 단계들과 유사한 프로세스를 따를 수 있지만, 대안 방법들이 고려된다. 사용자는 풀 이름, VM 크기, 저장 계정 친화도, 크로스토크 통신 요건, 메타데이터 등과 같은 다양한 파라미터들을 지정함으로써 풀의 생성을 개시할 수 있다. 사용자는 API 요청을 TLSFE로 전송할 수 있고, 이어서 TLSFE는 관련 요청(또는 동일 요청)을 올바른 풀 서버로 전송할 수 있다. 요청을 풀 서버로 전송하기 전에, 시스템은 사용자를 인증할 수 있다. 이어서, 수신 풀 서버는 고유 식별자를 갖는 풀 트랜잭션을 시작할 수 있다. 이어서, 풀 서버는 태스크 테넌트들에 대한 VM 예약 파괴를 생성하여, 전용 VM들의 타겟 수 및 대기로서 예약 상태를 유지하는 VM들의 수를 충족시킬 수 있다. 이러한 예약 동작은 시스템 장애시에 트랜잭션을 유지하는 것을 포함할 수 있으며, 따라서 풀 예약은 궁극적으로 유효할 수 있다. 이어서, 풀 서버는 예약들에 대해 갱신 상태를 유지하기 위해 태스크 스케줄러를 이용하여 상태(즉, 하트비트) 갱신들 및 검사들을 정기적으로 통신할 수 있다. 트랜잭션의 완료시, 풀 서버는 각각의 태스크 테넌트에 통지하여 그들의 트랜잭션을 커미트하고, 이어서 각각의 태스크 테넌트가 응답한 후에 미결 트랜잭션을 제거함으로써 트랜잭션을 커미트할 수 있다.

[0122] 대기 VM 예약들을 전용으로 변환함으로써(또는 그 반대로) 풀이 갱신될 수 있다. 이것은 시스템(또는 사용자)

가 원하는 수의 전용 및 대기 VM들을 제공함으로써 달성될 수 있다. 이어서, 풀 서버는 관련 태스크 테넌트들을 이용하여 새로운 트랜잭션들을 시작하고 새로운 VM 타겟들을 그러한 태스크 테넌트들로 전달함으로써 풀의 갱신을 처리할 수 있다. 태스크 테넌트에서, 들어오는 요청이 TTLF를 통해 태스크 스케줄러로 라우팅될 수 있다. 원하는 타겟이 전용 VM들의 수의 증가 및 여분의 대기 VM들의 예약을 포함하는 경우, 태스크 스케줄러는 할당되지 않은 VM들을 전용 VM들로 변환하고, 대기 카운트를 줄인다. 추가적인 전용 VM들이 여전히 요구되는 경우, 태스크 스케줄러는 선점 풀로부터 VM들을 할당한다. 불충분한 VM들이 이용 가능한 경우, 태스크 테넌트는 요청을 충족시키기 위해 확장될 수 있다. 즉, 자유 VM들이 먼저 할당된 후에 선점 VM들로 변환되는 것이 고려된다. 그러나, 선점 VM들이 먼저 변환되고, 자유 VM들이 할당되어, 임의의 남은 자원 요구를 충족시키는 것도 고려된다.

[0123] 서비스로서 플랫폼(PAAS)을 이용하는 작업 스케줄링

[0124] 도 12는 본 발명의 양태들에 따른, 분산 컴퓨팅 환경에서 자원들을 제공하기 위한 예시적인 방법(1200)을 도시하는 블록도를 나타낸다. 블록 1202에서, 사용자로부터 태스크 계정을 생성하기 위한 요청이 수신된다. 예를 들어, 사용자는 (서비스 관리 API를 포함하는) 클라이언트 포털을 통해, 서비스가 사용자에 의해 계산들을 수행하기 위해 사용될 수 있는 계정을 생성할 것을 요청할 수 있다. 요청은 시스템의 상위 레벨 위치 서비스(예로서, XLS)에서 수신될 수 있다. 도시되지 않았지만, 상위 레벨 서비스는 컴퓨팅 자원들(예로서, 지리 영역들)의 하나 이상의 정의된 클러스터에서 하위 레벨 태스크 위치 서비스와 통신함으로써 생성되는 것이 고려된다. 또한, 전술한 바와 같이, 상위 레벨 위치 서비스는 계정 또는 계정에 의해 사용되는 자원들과 관련된 이름 공간의 등록을 조정하는 것이 고려된다. 이러한 이름 공간의 등록은 도메인 이름 서비스를 적어도 부분적으로 이용하여 수행될 수 있다.

[0125] 블록 1204에서, 태스크 계정과 관련된 워크 아이템이 수신된다. 예를 들어, 사용자는 클라이언트 포털을 통해 워크 아이템을 제출할 수 있다. 워크 아이템은 사용자에 의해 시스템의 태스크 위치 서비스로 전송되는 것으로 고려된다. 예시적인 양태에서, 워크 아이템은 시스템에 의해 유지되는 등록된 이름 공간에 적어도 부분적으로 기초하여 TLS로 라우팅된다. 예를 들어, 전술한 바와 같이, 다양한 통신들이 계정 가상 IP 어드레스 및/또는 태스크 가상 IP 어드레스를 이용하여 통신될 수 있다.

[0126] 많은 컴포넌트 중에서 특히, 워크 아이템은 하나 이상의 가상 기계에 의해 실행될 바이너리들, 가상 기계에 의해 사용될 수 있는 명령 라인 파라미터들, 및/또는 규칙들을 포함할 수 있는 것이 고려된다. 규칙들은 예를 들어 작업들을 자동 생성하고, 풀들을 자동 생성하고, 풀들 상에서 작업들을 자동 실행하고/하거나, 풀들을 자동 삭제하기 위해 시스템에 의해 사용될 수 있다. 다른 특징들이 규칙들에 의해 달성될 수 있다. 블록 1206에서, 시스템은 워크 아이템과 관련된 규칙들을 수신한다. 전술한 바와 같이, 규칙들은 사용자에 의해 전달될 수 있거나, 사용자 제공 정보(예로서, 태스크 계정과 관련된 선호들)에 기초하여 시스템에 의해 적어도 부분적으로 생성될 수 있다. 규칙들은 워크 아이템, 결과적인 작업, 관련 태스크들 및/또는 풀의 특성들을 정의하는 더 광범위한 사양의 일부일 수 있다.

[0127] 블록 1208에서, 워크 아이템에 기초하여 작업이 자동으로 생성된다. 전술한 바와 같이, 작업은 시스템의 WIJ에 의해 생성될 수 있다. 작업은 예시적인 양태에서 워크 아이템과 관련하여 수신된 규칙들/사양에 기초하여 생성된다. 이러한 규칙들/사양에 포함된 많은 정보 가운데 특히, 시스템이 사용자에게 부담을 주지 않고서 풀(또는 풀들의 세트) 상의 최종 인스턴스화를 위한 작업을 자동 생성하는 것을 가능하게 하는 우선순위 및 다른 계산 변경 정보가 포함될 수 있다.

[0128] 블록 1210에서, 자동 풀이 워크 아이템에서 지정될 때, 가상 기계들은 작업의 처리에 사용하기 위해 풀에 자동 할당된다. 전술한 바와 같이, 풀에 할당된 가상 기계들은 워크 아이템과 관련된 사양 및/또는 규칙들에 부분적으로 기초할 수 있다. 예를 들어, 워크 아이템과 관련된 정보는 원하는 자원 소비, 워크 아이템이 완료되기를 원하는 시간, 태스크들을 계산하기 위한 지리 영역 등을 지정할 수 있다. 풀 서버는 이러한 정보를 이용하여, 0개 이상의 가상 기계를 풀에 할당할 수 있다. 할당을 위해 사용되는 정보는 전용 및 대기 VM들의 수 등을 지정하는 사양의 일부일 수 있다.

[0129] 블록 1212에서, 작업 큐(또는 전술한 바와 같은 복수의 작업 큐)가 WIJ에 의해 자동 생성될 수 있다. 또한, 예시적인 양태에서 JM 태스크가 생성될 수 있는 것이 고려된다. 게다가, 블록 1214에서 지시되는 바와 같이, 작업을 풀에 할당한 후에 풀의 VM들 상에서 VM들의 태스크들을 스케줄링함으로써 워크 아이템이 풀 상의 작업으로서 인스턴스화될 수 있는 것이 고려된다. 따라서, 워크 아이템이 풀 상에 작업으로서 스케줄링될 수 있는 것이 고려된다. 이어서, 스케줄링된 작업의 태스크들은 VM들에 할당되어, 작업의 인스턴스화의 일부로서 실행될 수

있다. 전술한 바와 같이, 태스크 테넌트의 태스크 스케줄러가 풀 내의 가상 기계들 중 하나 이상에서 큐 내에 하나 이상의 태스크를 스케줄링하는 것을 담당할 수 있는 것이 고려된다. 제2 작업이 풀 상에서 인스턴스화되는 것도 고려된다. 제2 작업은 워크 아이템의 재발생 인스턴스 또는 완전히 다른 워크 아이템에 기초하는 작업일 수 있다.

[0130] 블록 1216에서, 시스템은 사용자의 개입 또는 요청 없이 풀을 자동 삭제하는 것이 고려된다. 풀은 작업 또는 워크 아이템의 완료에 응답하여 삭제될 수 있다. 예를 들어, 풀은 각각의 작업이 완료된 후에 삭제될 수 있다. 게다가, 풀은 계정(또는 다른 계정) 또는 워크 아이템에 의한 미래의 사용을 위해 작업의 완료 후에 유지될 수 있는 것이 고려된다. 예를 들어, 워크 아이템이 재발생을 위해 스케줄링되는 경우, 작업의 재생성을 예상하여 풀을 유지하고, 실행 작업들에 걸치는 VM들 상에 상태(예로서, 데이터 파일들 및 애플리케이션들)를 유지하는 것이 효율적일 수 있다. 또한, 자동 스케일링 기능을 이용하여, 풀은 전용 VM들을 대기 VM들로 자동 변환하거나, 풀 상의 미결 태스크들의 수와 같은 스케일링 규칙들에 응답하여 VM들의 수를 줄일 수 있는 것이 고려된다. 또한, 사용자로부터의 요청이 수신될 수 있는 것이 고려된다. 요청은 풀의 삭제에 대한 지시를 포함할 수 있다.

[0131] 풀의 자동 스케일링 및 계층적 구조화

[0132] 도 13은 본 발명의 양태들에 따른, 분산 컴퓨팅 환경에서 자원들을 제공하기 위한 예시적인 방법(1300)을 도시하는 블록도를 나타낸다. 특히, 규칙들, 요구들 및 현재의 자원 부하에 기초하여 자원들의 풀을 자동으로 스케일링하는 것은 분산 컴퓨팅 환경에서 자원들의 효과적인 제공을 가능하게 하는 것이 고려된다. 전술한 바와 같이, 작업이 인스턴스화되는 자원의 세트로서 시스템(예로서, 분산 컴퓨팅 환경)에 의해 풀이 생성될 수 있다. 그러나, 자원 요구의 변화, 스케줄링 변화 및 다른 변수들에 기초하여, 크기를 증가시키고, 크기를 감소시키고, 특정 VM 타입을 증가/감소시키고, 풀과 관련된 다른 변수를 조정하는 것이 바람직할 수 있다. 전통적으로, 이러한 프로세스는 본질적으로 수동일 수 있으며, 풀 자원 레벨을 수동으로 조정하기 위해 사람의 개입에 의존할 수 있다.

[0133] 풀과 같은 자원들의 스케일링에 자동화를 제공하여, 시스템의 자원들을 효과적으로 이용할 수 있는 것이 고려된다. 예시적인 양태에서, 방법(1300)은 사용자의 개입 없이 풀의 자동 스케일링을 가능하게 한다. 블록 1302에서, 시스템은 풀과 관련된 가상 기계들의 수를 자동 결정한다. 결정은 풀 서버에 의해 완료될 수 있다. 수의 결정은 가상 기계들의 총 수를 결정하거나, 특정 타입의 가상 기계의 총 수를 결정하는 것을 포함할 수 있다. 예를 들어, 결정은 풀 내의 전용 VM들의 총 수를 결정하는 것일 수 있다. 결정은 시스템의 하나 이상의 컴포넌트/모듈에 의해 유지되는 테이블 또는 기타 리스트로부터 행해질 수 있다. 예를 들어, 풀 서버가 풀에 의해 사용되는 가상 기계들을 갖는 태스크 테넌트들의 리스트를 유지할 수 있는 것이 고려된다. 유사하게, 태스크 테넌트가 풀에 의해 사용되는 VM들의 리스트를 유지할 수 있는 것이 고려된다(예를 들어, 태스크 스케줄러가 그러한 정보를 유지할 수 있다). 따라서, 결정은 시스템 내에 이미 유지되고 있는 정보의 하나 이상의 소스를 참조하여 행해질 수 있다.

[0134] 블록 1304에서, 풀과 관련된 가상 기계들의 수를 조정하기 위해 자동 스케일링 규칙이 적용된다. 예를 들어, VM들의 총 수가 감소 또는 증가하는 것이 고려된다. 또한, 특정 타입(예로서, 전용)의 VM의 수가 증가 또는 감소하는 것이 고려된다. 전술한 바와 같이, 자동 스케일링 규칙들은 풀의 크기/구성의 결정을 돕기 위한 하나 이상의 선호를 포함할 수 있다. 예를 들어, 자동 스케일링 규칙들은 현재 풀 통계들 및 스케줄링된 작업들에 기초하여 최적 풀을 결정하기 위한 하나 이상의 공식을 포함할 수 있다. 자동 스케일링 규칙들은 작업 큐 통계들(예로서, 실행을 기다리는 미결 태스크들, 인큐 레이트, 디큐 레이트, 태스크 완료 레이트 등), 스폿 가격 정보, 가용 자원들, 자원들의 효율 등과 같은 시스템에 관한 메트릭들을 고려할 수 있다. 또한, 자동 스케일링 규칙들은 워크 아이템, 작업 및/또는 태스크에 대한 원하는 완료 시간도 고려할 수 있는 것이 고려된다. 게다가, 자동 스케일링 규칙들은 확장되는 것이 바람직한(예를 들어, 전용 VM들보다 적은 비용의 선점 VM들에 의존하는) 원하는 금융 자원들을 고려할 수 있는 것이 고려된다.

[0135] 자동 스케일링 규칙들을 적용하여, 풀과 관련된 가상 기계들의 수를 조정한 결과로서, 시스템은 블록 1306에서 지시되는 바와 같이 풀의 하나 이상의 가상 기계를 동적으로 할당할 수 있다. VM들의 동적 할당은 하나 이상의 VM의 추가, 하나 이상의 VM의 제거, 하나 이상의 VM의 타입의 변경을 포함할 수 있다. 예를 들어, 자동 스케일링 규칙들의 적용은 풀에 전용화된 VM들인 하나 이상의 가상 기계의 추가를 유발할 수 있다. 또한, 할당은 자유 또는 선점 VM들을 전용 VM들로 변환하고 대기 카운트를 줄이는 것을 포함할 수 있는 것이 고려된다. 다른 할당들도 고려된다.

[0136] 할당의 동적 성질은 시스템이 사용자 개입 없이 할당을 수행하는 것과 관련된다. 예를 들어, 자동 스케일링 규칙들은 인터벌(예로서, 시간 인터벌, 프로세스 카운트 인터벌)을 두고 적용될 수 있는 것이 고려된다. 예시적인 양태에서, 자동 스케일링 규칙들을 자동으로 실행한 결과로서, 자원들의 할당은 사용자의 요청 없이 할당시에 또는 할당이 수행될 것을 요청하는 사용자 입력의 직접적인 결과로서 발생할 수 있다.

[0137] 도 14는 본 발명의 양태들에 따른, 분산 컴퓨팅 환경에서 자원들을 제공하기 위한 방법(1400)을 도시하는 블록도를 나타낸다. 도 13과 관련하여 전술한 바와 같이, 풀에 적용될 때 자동 스케일링 규칙들의 사용 및 적용은 분산 컴퓨팅 환경에서 자원들의 제공을 가능하게 하는 것이 고려된다.

[0138] 블록 1402에서, 사용자로부터 워크 아이템이 수신된다. 사용자는 API를 통해 워크 아이템을 통신할 수 있으며, 이는 클라이언트 포털을 통해 시스템의 상위 레벨 위치 서비스로 올 수 있다. 워크 아이템은 관련된 계정 선호들을 가질 수 있는 특정 태스크 계정과 관련될 수 있다. 블록 1404에서, 사용자로부터 자동 스케일링 규칙이 수신된다. 자동 스케일링 규칙은 워크 아이템의 완료에 대한 시간, 워크 아이템의 우선순위, 바람직한 금융 자원 지출(예로서, 전용 VM들에 대한 선점 VM들의 선호)에 관한 사용자 선호들을 제공할 수 있다. 자동 스케일링 규칙은 풀에 대한 요구들에 응답하여 풀을 스케일링하는 것을 돕기 위해 풀과 관련된 하나 이상의 메트릭을 이용하는 하나 이상의 공식도 포함할 수 있다.

[0139] 블록 1406에서, 제1 데이터 센터에 위치하는 제1 VM이 풀에 할당된다. 가상 기계의 할당은 풀 내에서 사용 가능한 자원으로서의 가상 기계의 추가를 포함할 수 있다. 예시적인 양태에서, 제1 VM은 제1 풀에 자동 할당된다. 이것은 어떤 VM인지, 어떤 풀인지에 관계없이 VM이 풀 내에 포함되어야 한다는 것을 지시하는 사용자의 개입 없이 행해진다. 예를 들어, 과거에 사용자는 풀 내에 포함될 자원들을 식별함으로써 풀을 형성하는 것이 필요했을 수 있다. 자원들의 식별은 VM의 수 및 VM들이 할당하는 곳을 식별하는 것을 포함할 수 있다. 이 예에서, 시스템 자체는 풀이 생성되어야 하고, 풀이 원하는 성능 또는 동작 특성을 충족시키기 위해 풀에 다수의 VM이 할당되어야 한다는 것을 식별한다.

[0140] 블록 1408에서, 제2 가상 기계가 제1 풀에 할당된다. 제2 VM은 제1 데이터 센터로부터 지리적으로 분리된 제2 데이터 센터 내에 있다. 예시적인 양태에서, 제2 VM은 동일한 데이터 센터 내의 그러나 상이한 태스크 테넌트 내의 VM 자원들의 세트로부터 올 수 있다. 제2 VM은 사용자에 의한 개입 없이 제1 풀에 자동 할당되는 것이 고려된다. 이러한 자동화(및 사용자의 개입 없이 시스템에 의해 수행되는 다른 자동 태스크들)는 사용자가 개입하는 것이 허가될 때 사용되지 않을 프로세스 및 단계들을 포함한다. 예를 들어, 어느 VM을 제1(또는 임의의) 풀에 할당할지에 관한 결정이 본 발명의 일 실시예 내에서 고려된다.

[0141] 제1 태스크 테넌트 및 제2 태스크 테넌트는 시스템을 서비스하는 물리적으로 독립된 데이터 센터들일 수 있다(또는 공통 데이터 센터 내에 있을 수 있다). 예를 들어, 제1 태스크 테넌트는 제1 지리 위치에(예로서, 특정 주소, 도시, 주, 영역, 국가 및/또는 대륙에) 위치할 수 있는 것이 고려된다. 일례에서, 제1 태스크 테넌트는 미국의 제1 영역(예로서, 미국 서부)에 위치하며, 제2 태스크 테넌트는 미국의 상이한 영역(예로서, 미국 동부)에 위치하는 것이 고려된다. 이 예에서, 제1 태스크 테넌트 및 제2 태스크 테넌트는 공통 또는 상이한 태스크 위치 서비스에 의해 어드레스될 수 있다. 이것은 다양한 데이터 센터들에 걸치는 풀의 확장(또는 풀의 축소)의 복잡성을 유발하는 사용자의 개입 없이 풀의 자동 스케일링을 가능하게 한다. 예를 들어, 물리적 제한이 물리적 데이터 센터의 크기를 초과하는 풀의 확장을 이전에 방지했을 수 있는 것이 고려된다. 로컬 VM 자원들의 제어에 있어서 스케줄러들(태스크 테넌트들)로부터 작업들 및 풀들의 개념들을 분리하는 것은 풀 및 그의 작업들이 둘 이상의 데이터 센터 내의 자원들을 포함하도록 확장되는 것을 가능하게 하며, 이는 사용자가 그러한 확장을 가능하게 하기 위해 시스템을 프로그래밍하거나 시스템에 개입하는 것을 필요로 하지 않는다. 따라서, 사용자는 시스템이 풀의 스케일링을 자동으로 처리하는 것에 기초하여 수십억 개의 VM에 걸쳐 그리고 소수의 VM에 걸쳐 스케일링할 수 있는 워크 아이템을 설계할 수 있는 것이 고려된다.

[0142] 블록 1410에서, 워크 아이템은 제1 가상 기계 및 제2 가상 기계 상에서 인스턴스화된다. 워크 아이템의 인스턴스화는 태스크 테넌트 내의 스케줄러가 큐로부터 태스크를 인출하여 VM에 할당하는 것을 포함할 수 있다. 워크 아이템의 인스턴스화는 가상 기계 상에서의 워크 아이템으로부터의 태스크의 스케줄링을 더 포함할 수 있다. 인스턴스화는 또한 가상 기계가 태스크의 처리를 위해 큐로부터 태스크를 인출하는 것을 포함할 수 있다. 인스턴스화는 또한 가상 기계에서의 태스크들의 처리를 포함할 수 있다. 따라서, 워크 아이템의 인스턴스화는 워크 아이템의 일부(예로서, 태스크)가 VM에 의해 처리되게 하는 임의의 양태를 포함할 수 있다.

[0143] 제1 VM 및 제2 VM 양자를 갖는 풀에 대해 워크 아이템의 작업을 인스턴스화함으로써, 워크 아이템은 다수의 데이터 센터에 걸쳐 스케일링하는 풀에 의해 처리되며, 사용자는 이러한 결과를 달성하기 위해 워크 아이템을 변

경할 필요가 없다. 따라서, 단일 태스크 테넌트에 의해 서빙되는 풀에서 실행될 수 있는 동일 워크 아이템이 사용자 개입 없이 다수의 개별 태스크 테넌트 및 데이터 센터에 걸치는 풀에서도 실행될 수 있는 것이 고려된다.

[0144] 블록 1412에서, 자동 스케일링 규칙들이 풀에 적용된다. 예시적인 양태에서, 시스템은 사용자 개입 없이 자동 스케일링 기능을 자동으로 개시한다. 자동 스케일링 규칙들은 현재 사용되고 있는 자원들, 커미트되도록 스케줄링된 자원, 및 풀에 필요한 자원들, 풀에 할당된 작업들에 대한 큐들 내의 미결 워크의 양, 태스크들, 작업들 등을 고려할 수 있다. 함께 취해질 경우에, 자동 스케일링 규칙들은 풀이 VM 타입들을 확장, 축소 및 변경하는 것을 가능하게 한다.

[0145] 예를 들어, 추가 태스크들이 작업에 추가되거나, 작업을 처리하기 위한 시간이 초기 예측을 초과하는 경우에, 자동 스케일링 규칙들은 작업을 완료하기 위해 얼마나 많은 추가 자원이 필요할지를 결정하는 데 사용될 수 있는 것이 고려된다. 유사하게, 자동 스케일링 규칙들의 호출은 풀 내의 VM들의 수가 과다할 수 있고, 그러한 자원들 중 일부가 변환 또는 제거될 수 있다는 결정을 유발할 수 있는 것이 고려된다. 예를 들어, 전용 VM은 자동 스케일링 규칙들에 기초하여 선점 또는 심지어 대기 VM 예약으로 변환될 수 있다. 또한, VM은 자동 스케일링 규칙들에 의한 결정이 적용된 결과로서 풀로부터 완전히 해제될 수 있는 것이 고려된다.

[0146] 블록 1414에서, 풀과 관련된 하나 이상의 VM의 재할당이 발생한다. 예를 들어, 자동 스케일링 규칙들이 전용 VM이 풀로부터 해제될 수 있는 것으로 결정하는 경우, 전용 VM은 풀로부터 자유로워질 수 있다. 유사하게, 자동 스케일링 규칙들이 이용 가능한 VM의 수가 원하는 결과들(예로서, 작업의 적시 완료)을 달성하기에 충분하지 않은 것으로 결정하는 경우, 큐들 내의 또는 아직 큐잉되지 않은 미결 태스크들을 완료하는 데 사용하기 위해 풀에 하나 이상의 추가 VM이 할당될 수 있다. 또한, 전술한 바와 같이, 예시적인 양태에서 하나 이상의 VM의 재할당은 VM들을 제1 타입으로부터 제2 타입으로 변환하는 것을 포함할 수 있는 것이 고려된다. 이러한 재할당은 예시적인 양태에서 풀과 관련된 풀 서버에 의해 적어도 부분적으로 수행되는 것으로 고려된다.

[0147] 자원, 스케줄링 및 작업들의 분리

[0148] 자원들, 스케줄링 및 작업들의 분리는 작업이 하나의 자원 풀로부터 다른 자원 풀로 계속 실행되는 것을 가능하게 한다. 예를 들어, 이러한 기능은 워크를 이동시키고 상이한 계산 자원들에 걸쳐 워크를 부하 균형화할 때 사용될 수 있다. 특정 예에서, 특정 데이터 센터의 장애(예로서, 자연 재해)가 발생하는 경우, 워크는 작업의 완료를 위해 새로운 데이터 센터로 이동될 수 있다. 또한, 자원들, 스케줄링 및 작업들의 분리는 작업이 자원들 및 스케줄들의 여러 개의 풀에 걸쳐 실행을 확장하는 것을 가능하게 하는 것이 고려되며, 이는 작업이 다른 방식으로는 얻지 못하는 높은 레벨의 확장성을 달성하는 것을 가능하게 할 수 있다. 또한, 풀이 다수의 스케줄러, 태스크 테넌트 및/또는 데이터 센터에 걸치는 것이 고려되며, 이는 그 풀에 할당된 작업도 그러한 자원들에 걸치는 것을 가능하게 한다.

[0149] 예시적인 양태에서, 분리는 시스템에서 3개의 개별 개념을 이용하는 것을 고려한다. 제1 개념은 가상 기계들의 풀들의 개념에 기초한다. 제2 개념은 작업 관리, 작업 상태 및 작업 큐들 주변에 형성된다. 제3 개념은 스케줄러들(예로서, 도 9의 태스크 스케줄러(902)) 및 그들이 이러한 풀들을 위해 할당된 작업들을 스케줄링하는 것을 담당하는 대상인 VM들과 관련된다. 이러한 개념들은 멀티 테넌시 분산 컴퓨팅 시스템의 부하 균형화 요구에 기초하여 작업들, 스케줄러들 및 풀들의 유연한 재할당을 가능하게 한다. 또한, 3개의 개념들은 장애 복구도 가능하게 하는 것이 고려된다.

[0150] 도 15는 본 발명의 양태들에 따른, 분산 컴퓨팅 환경에서 자원, 스케줄링 및 작업들의 분리를 제공하기 위한 방법(1500)을 도시하는 블록도를 나타낸다. 블록 1502에서, 시스템에서 워크 아이템이 수신된다. 예를 들어 워크 아이템을 제출하는 계정과 관련된 태스크 가상 IP 어드레스를 이용하여 태스크 위치 서비스에서 워크 아이템이 수신될 수 있다. 블록 1504에 지시되는 바와 같이, 워크 아이템으로부터 작업이 생성될 수 있다. 일 실시예에서, 작업 생성은 워크 아이템이 작업으로서 처리되어야 한다는 식별이다.

[0151] 블록 1506에서, 가상 기계가 제1 풀에 할당된다. 예를 들어, 풀 서버가 VM을 제1 풀에 할당하여, 작업이 인스턴스화(예로서, 처리)될 수 있는 풀을 제공할 수 있는 것이 고려된다. 블록 1508에서, 작업이 제1 풀에 할당된다. 예를 들어, 제1 풀과 관련된 풀 서버가 제1 태스크 테넌트 스케줄러에 의해 소유된 VM들을 할당하였으며, 따라서 작업이 풀에 할당되면, 제1 스케줄러가 작업 큐로부터 태스크들을 인출하여, 이들을 태스크들을 실행하기 위해 스케줄러에 의해 소유되는 그 풀 내의 VM들에 할당할 수 있는 것이 고려된다.

[0152] 블록 1510에서, 작업의 태스크들이 제1 풀에 대해 스케줄링된다. 전술한 바와 같이, 태스크를 풀에 대해 스케

줄링하는 것은 제1 태스크 테넌트 스케줄러가 풀과 관련된 하나 이상의 VM 상에서 작업의 하나 이상의 태스크를 처리하는 것을 포함할 수 있다. 또한, 작업 큐로부터 태스크를 차용하는 프로세스들이 태스크의 인스턴스화의 일부인 것이 고려된다. 스케줄러가 VM들을 "소유"하고, 또한 VM들에서 실행되는 프로세스들을 "소유"하는 것이 고려된다.

[0153] 블록 1512에서, 작업이 제2 풀에 재할당되어야 한다는 결정이 행해진다. 제2 풀은 동작에 있어서 제1 풀과 무관할 수 있으며, 따라서 장애를 유발하는 지리적으로 제한된 재해(예를 들어, 토네이도, 허리케인, 지진, 전력 그리드 고장, 네트워크 장애)는 제2 풀에 직접 영향을 주지 않는다. 결정은 또한 둘 이상의 풀, 태스크 테넌트 또는 데이터 센터에 걸쳐 자원 부하를 균형화하도록 동작하는 자동 균형화 프로세스에 응답할 수 있다. 예를 들어, 공통 위치(예로서, 뉴욕)에 기초하는 소정의 계정들은 공통 시간(예로서, 금융 거래일의 시작)에 자원들을 사용할 수 있다. 이 예에서, 미국 동부의 지리 영역에 집중된 자원의 풀은 미국 서부의 지리 영역에 위치하는 자원보다 더 많은 부담을 가질 수 있다. 따라서, 지연 및 다른 팩터들(예로서, 친화도)도 고려할 경우, 부하 균형화 프로세스는 블록 1514에 지시되는 바와 같이 작업의 하나 이상의 부분을 제2 풀로 이동시키는 것이 더 효율적인 것으로 결정할 수 있다. 이동의 "소유권"의 이전으로 한정되는 것이 아니라, 둘 이상의 풀에 걸치는 부하 균형화를 고려하며, 이는 다수의 풀에 대한 작업의 할당이다. 또한, 이동의 개념은 상이한 태스크 테넌트들에 걸치는 풀의 확장을 포함한다. 결과적으로, 단일 풀이 둘 이상의 태스크 테넌트를 커버할 때에도 사용자는 작업이 그 풀에 의해 수행되고 있다는 느낌을 갖는 것이 가능하다.

[0154] 작업(또는 작업 내의 태스크들)의 이동은 큐 내의 태스크에 대한 차용의 해제를 포함할 수 있으며, 따라서 상이한 풀 내의 자원이 그 태스크의 차용을 획득할 수 있다. 대안 실시예에서, 작업의 이동은 새로운 태스크 서비스 위치의 자원과 관련된 재생성 및 스케줄링을 위해 워크 아이템을 새로운 태스크 위치 서비스에 재분배하는 것을 수반하는 것이 고려된다. 작업의 이동은 작업을 제1 풀로부터 제2 풀에 재할당하는 형태이다. 블록 1516에서, 제2 풀에서의 작업의 할당이 지시된다. 블록 1518에서, 작업은 제2 풀에서 실행되도록 스케줄링된다. 제1 풀의 스케줄러는 제1 풀과 관련된 WIJ, 태스크 테넌트 및 풀 서버일 수 있으며, 제2 풀의 스케줄러는 제2 풀과 관련된 상이한 WIJ, 태스크 테넌트 및 풀 서버일 수 있다. 따라서, 워크 아이템/작업을 단일 풀 또는 단일 스케줄러로 제한하지 않음으로써, 워크 아이템은 제1 풀과 관련된 자원의 장애가 발생하는 경우에도 상이한 풀로 이동될 수 있다.

[0155] 또한, 도 9와 관련하여 전술한 바와 같이, 태스크 테넌트의 태스크 스케줄러가 시스템의 풀들 및 워크 아이템들/작업들로부터 분리된 스케줄러인 것이 고려된다. 예를 들어, 태스크 스케줄러는 풀에 할당된 큐들로부터 태스크들을 선택하고, 태스크 스케줄러에 의해 스케줄링된 태스크 테넌트 내의 관련 VM들 중 어느 것이 태스크들을 수행할지를 제어할 수 있다. 이러한 개념은 풀이 다수의 태스크 테넌트에 걸칠 때 각각의 태스크 테넌트와 관련된 각각의 스케줄러가 태스크 스케줄러들 각각에 의해 소유되는 VM들에 의해 실행될 태스크들을 큐들로부터 인출하는 것을 가능하게 한다. 결과적으로, 워크 아이템들은 스케줄러들(예로서, 태스크 스케줄러들)로부터 자원들(예로서, VM들)로부터 작업(예로서, 워크 아이템들)을 효과적으로 분리하는 방식으로 풀에 걸치는 자원들 상에서 실행될 수 있다.

[0156] 이어서, 시스템에서의 부하 균형화가 다양한 레벨들에서 수행될 수 있다. 예를 들어, 부하 균형화는 태스크 테넌트들(예를 들어, 공통 TLS에 의해 서빙되는 공통 지리 영역 내의 다수의 태스크 테넌트)과 협력하는 풀 서버에 의해 스케줄링된 자원들의 공통 그룹핑 내에서 발생할 수 있다. 이러한 공통 그룹핑은 태스크 테넌트들의 그룹핑일 수 있다. 따라서, 부하 균형화는 예시적인 양태에서 2개의 상이한 레벨에서 수행될 수 있는 것이 고려된다. 이러한 제1 레벨의 부하 균형화는 공통 풀 서버와 관련된 태스크 테넌트들 사이에서 발생할 수 있다. 이러한 제1 레벨의 부하 균형화에서, 풀 서버는 VM들을 상이한 태스크 테넌트들에 걸쳐 할당할 수 있다. 제2의 더 높은 레벨의 부하 균형화는 상이한 TLS(예로서, 도 5의 TLS(512))에 걸쳐 발생할 수 있다. 이러한 레벨의 부하 균형화에서, 부하는 또한 상이한 풀 서버들에 걸쳐 분산될 수 있다. 이러한 타입의 부하 균형화는 시스템에 걸치는 부하의 이동을 유발하도록 구현될 수 있다. 고려되는 또 다른 레벨의 부하 균형화는 WIJ가 시스템 내의 다수의 풀에 걸쳐 워크 아이템들/작업들을 할당함으로써 수행된다.

[0157] 전술한 바와 같이, 작업의 이동은 완전한 작업의 이동, 아직 처리되지 않은 작업의 일부의 이동 및/또는 아직 처리되지 않은 부분 중 일부를 오리지널 풀에 유지하는 동안의 아직 처리되지 않은 작업의 일부의 이동을 포함할 수 있다. 따라서, 작업의 이동은 장애의 경우뿐만 아니라, 부하 균형화 동작에 응답하여서도 유용할 수 있다.

[0158] 본 명세서에서 제공되는 바와 같이, 다양한 모듈들, 컴포넌트들, 시스템들, 계층들 및 프로세스들이 설명된다.

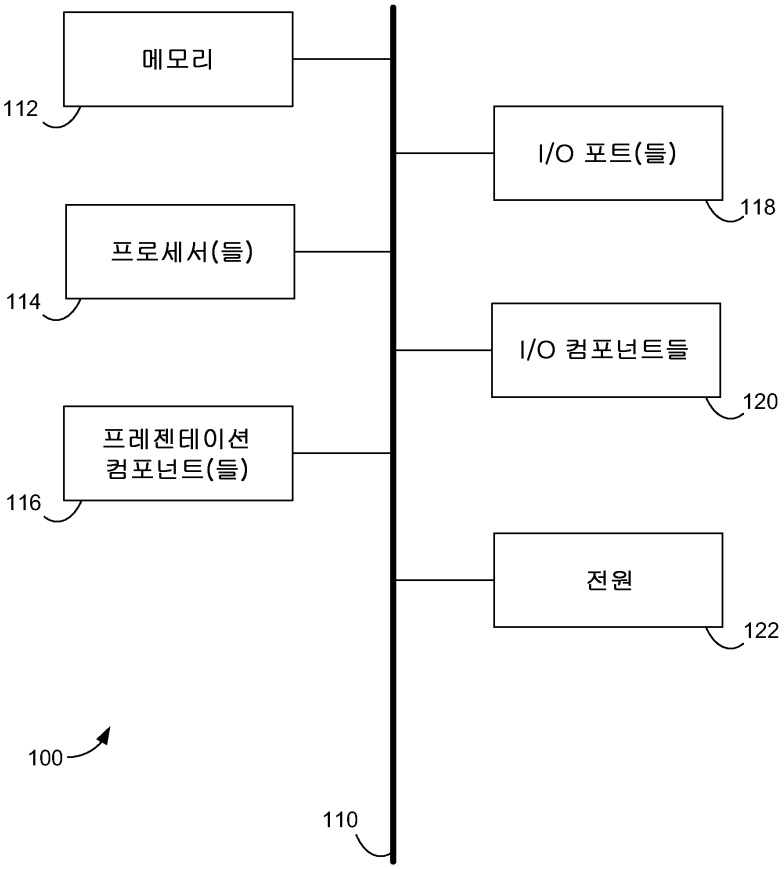
어느 것이라도 정보를 제1 형태로부터 제2의 사용 가능 형태로 변환하기 위한 고유 기계로서 수행될 수 있는 것이 고려된다. 이러한 변환은 제1 입력을 제2의 사용 가능한 출력으로 변환하도록 기능하는 프로세서 및 메모리에 의해 실시될 수 있다. 또한, 가상 기계들이 본 명세서에서 설명된다. 전술한 바와 같이, 가상 기계는 유형의 프로세서 및 메모리를 이용하여 태스크를 처리하여, 태스크를 분산 컴퓨팅 환경의 이익을 위해 시스템에 의해 사용될 수 있는 제2 형태로 변환하도록 기능한다.

[0159] 위의 설명으로부터, 본 발명은 명백하고 구조에 고유한 다른 장점들과 함께 전술한 모든 결과들 및 목적들을 달성하는 데에 적합한 발명이라는 것을 알게 될 것이다.

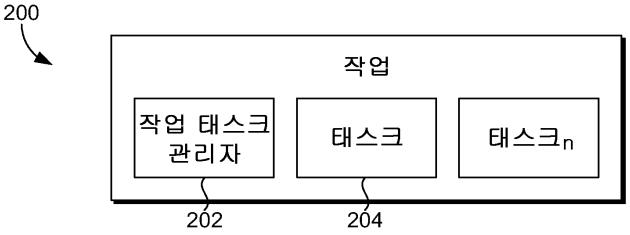
[0160] 소정의 특징들 및 하위 조합들이 유용하고 다른 특징들 및 하위 조합들과 관계없이 이용될 수 있다는 것을 이해할 것이다. 이것은 청구범위에 의해 고려되고 그 범위 내에 있다.

도면

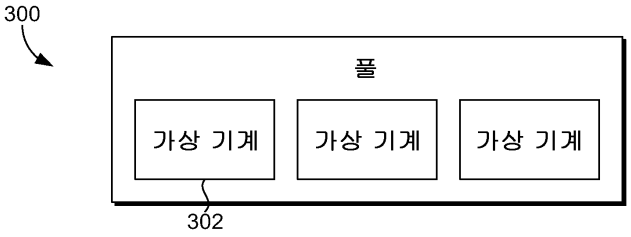
도면1



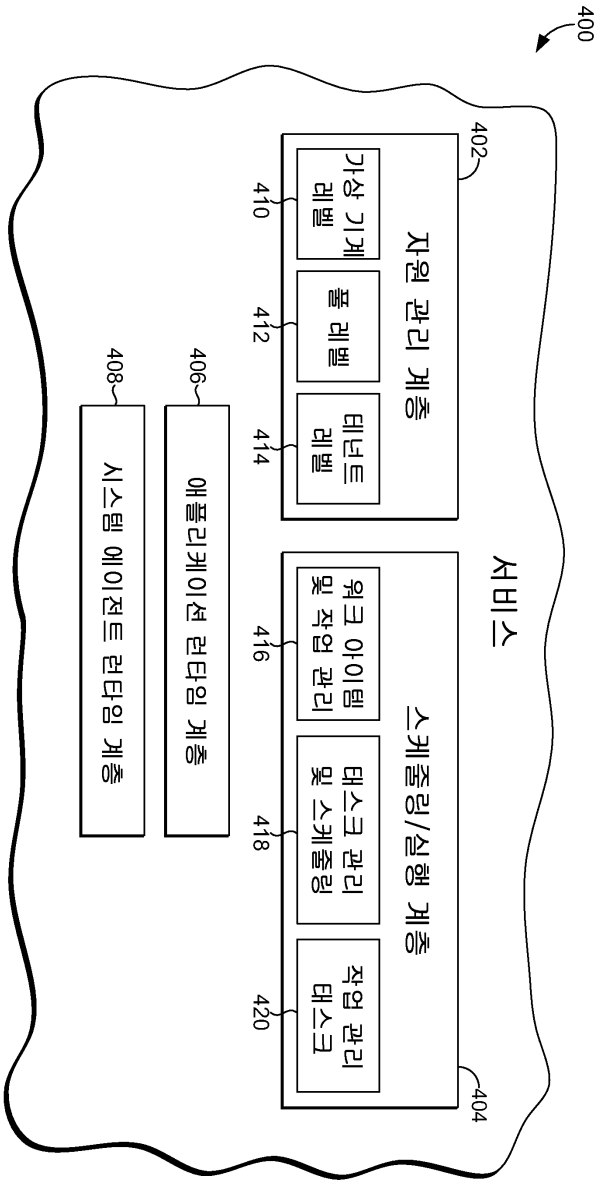
도면2



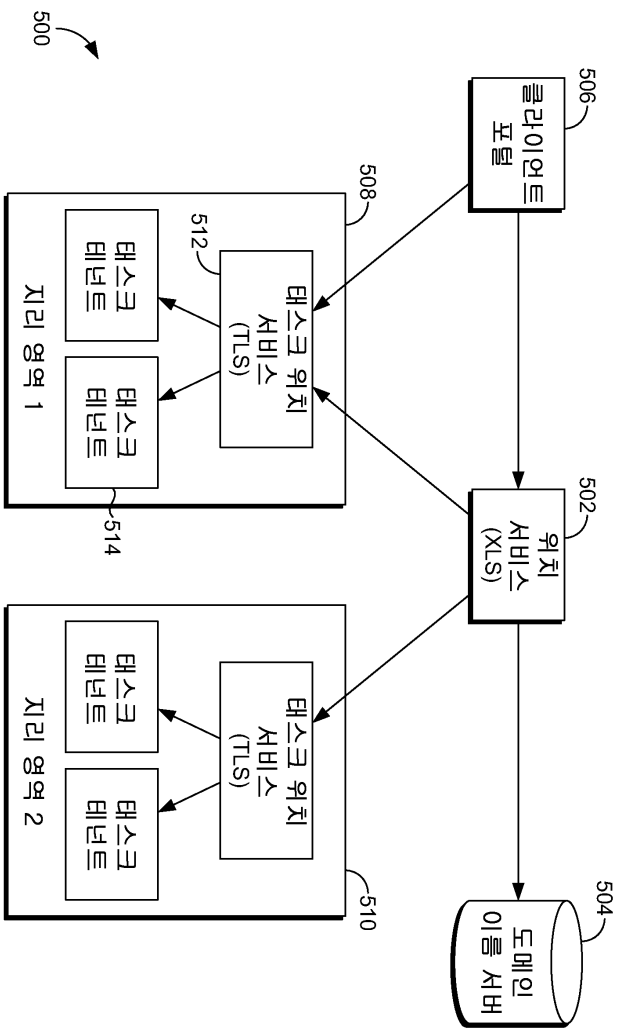
도면3


300
풀
가상 기계
가상 기계
가상 기계
302

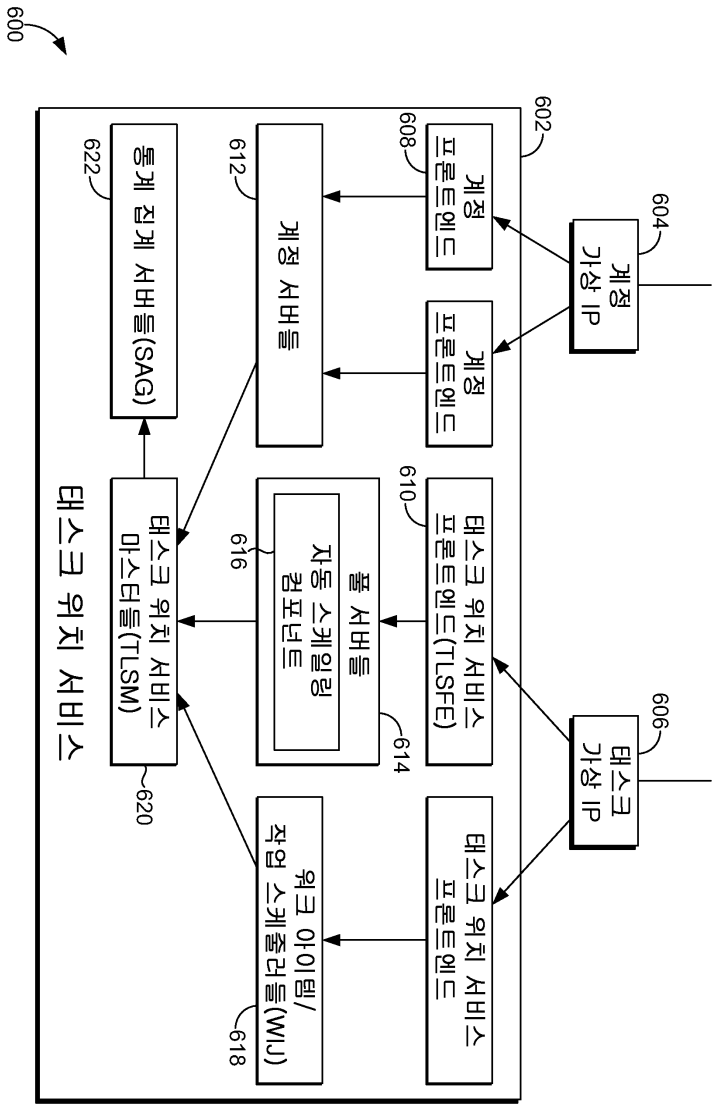
도면4


400
서비스
402
자원 관리 계층
410
가상 기계 레벨
412
풀 레벨
414
테넌트 레벨
404
스케줄링/실행 계층
416
워크 아이템 및 작업 관리
418
태스크 관리 및 스케줄링
420
작업 관리 태스크
406
애플리케이션 런타임 계층
408
시스템 에이전트 런타임 계층

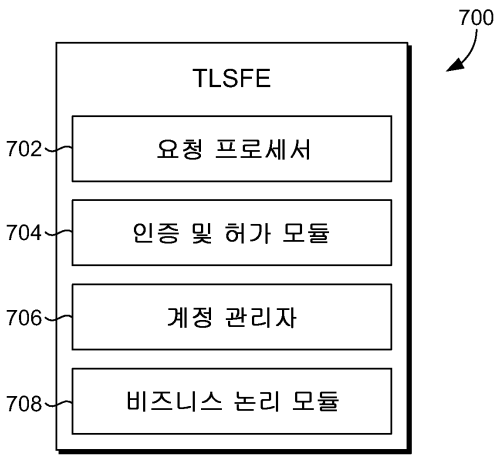
도면5


500
클라이언트
506
위치
서비스
(XLS)
502
도메인
이름 서버
504
508
지리 영역 1
태스크 위치
서비스
(TLS)
512
태스크
태스크
514
510
지리 영역 2
태스크 위치
서비스
(TLS)
태스크
태스크

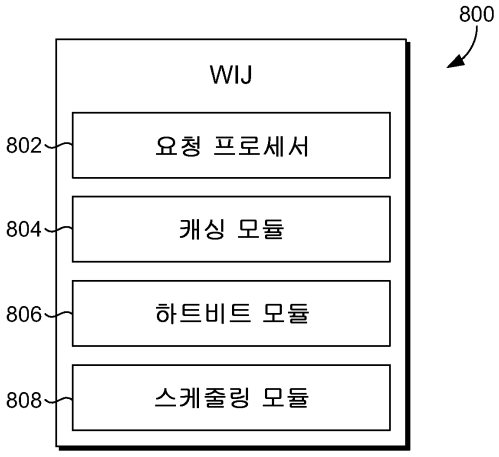
도면6


600
602
태스크 위치 서비스
계정 프론트엔드
608
계정 서버들
612
통계 집계 서버들(SAG)
622
계정 프론트엔드
계정 가상 IP
604
태스크 위치 서비스 프론트엔드(TLSFE)
610
풀 서버들
자동 스케일링 컴포넌트
616
614
태스크 위치 서비스 마스터들(TLSM)
620
태스크 위치 서비스 프론트엔드
태스크 가상 IP
606
워크 아이템/작업 스케줄러들(WIJ)
618

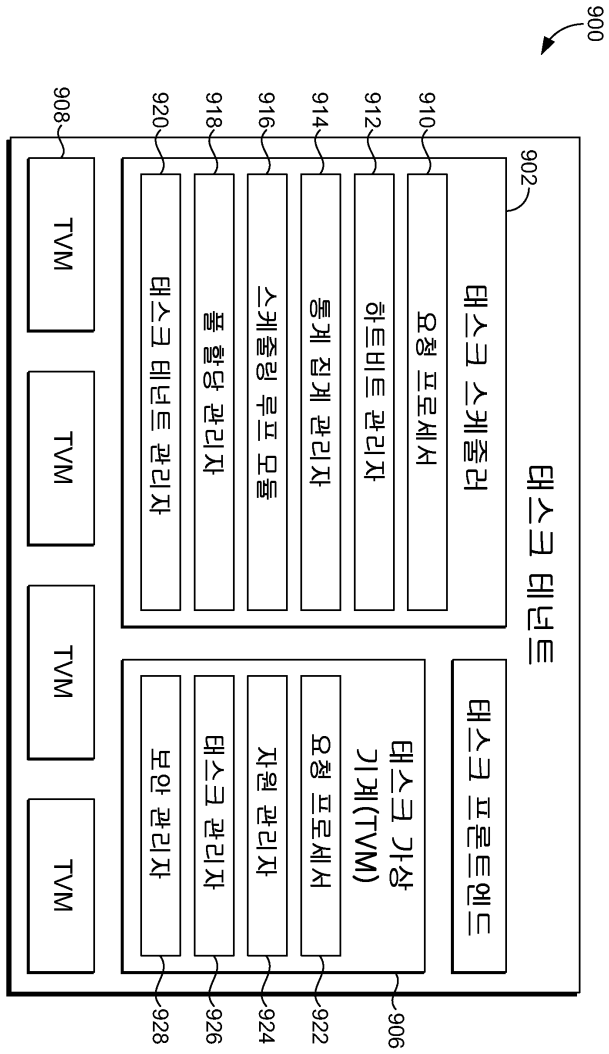
도면7


700
TLSFE
702
요청 프로세서
704
인증 및 허가 모듈
706
계정 관리자
708
비즈니스 논리 모듈

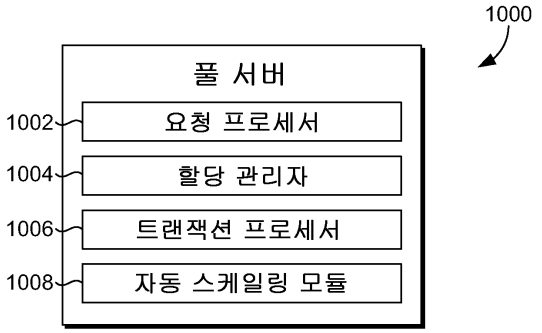
도면8


800
WIJ
802
요청 프로세서
804
캐싱 모듈
806
하트비트 모듈
808
스케줄링 모듈

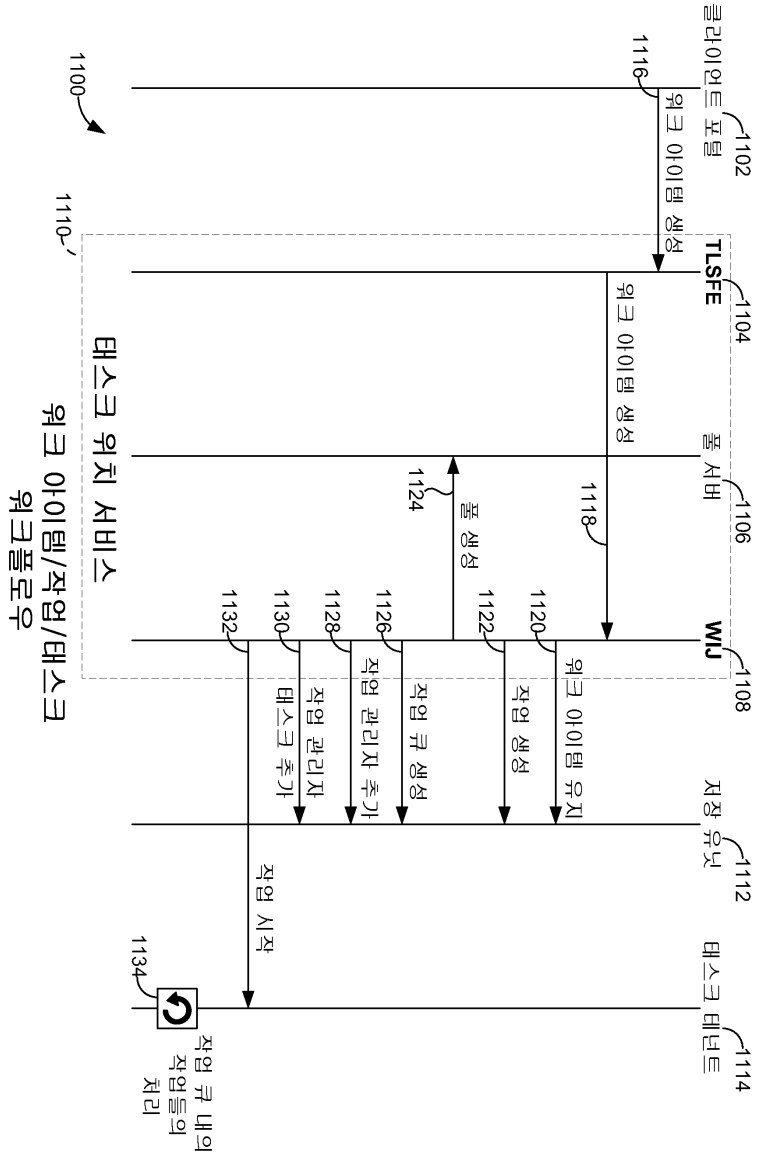
도면9


900
902
태스크 테넌트
910
요청 프로세서
912
하트비트 관리자
914
통계 집계 관리자
916
스케줄링 루프 모듈
918
풀 할당 관리자
920
태스크 테넌트 관리자
908
TVM
TVM
TVM
TVM
태스크 프론트엔드
태스크 가상 기계(TVM)
906
요청 프로세서
922
자원 관리자
924
태스크 관리자
926
보안 관리자
928

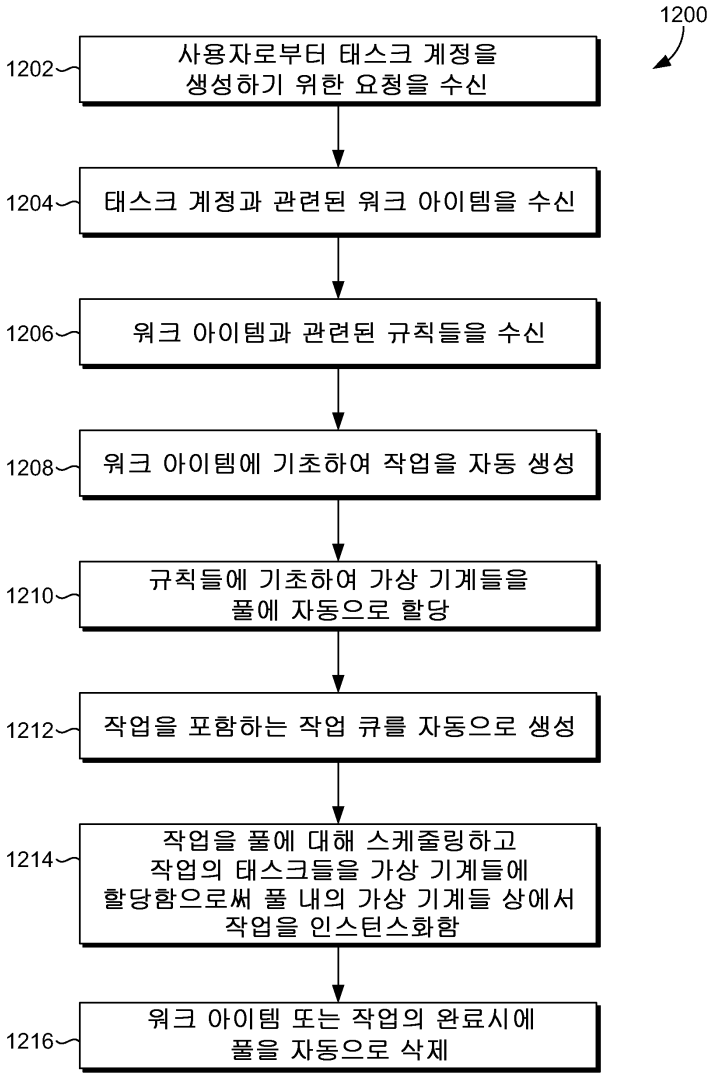
도면10


1000
풀 서버
1002
요청 프로세서
1004
할당 관리자
1006
트랜잭션 프로세서
1008
자동 스케일링 모듈

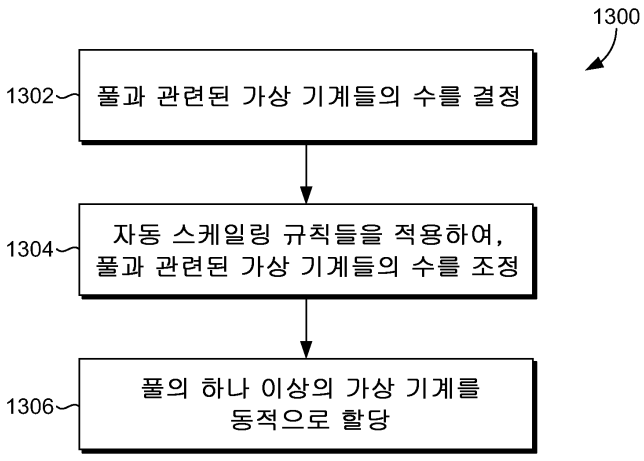
도면11


1100
1110
워크 아이템/작업/태스크 워크플로우
태스크 위치 서비스
클라이언트 포털
1102
TLSFE
1104
풀 서버
1106
WIJ
1108
저장 유닛
1112
태스크 테넌트
1114
1116
워크 아이템 생성
워크 아이템 생성
1118
1120
워크 아이템 유지
1122
작업 생성
1124
풀 생성
1126
작업 큐 생성
1128
작업 관리자 추가
1130
작업 관리자 태스크 추가
1132
작업 시작
1134
작업 큐 내의 작업들의 처리

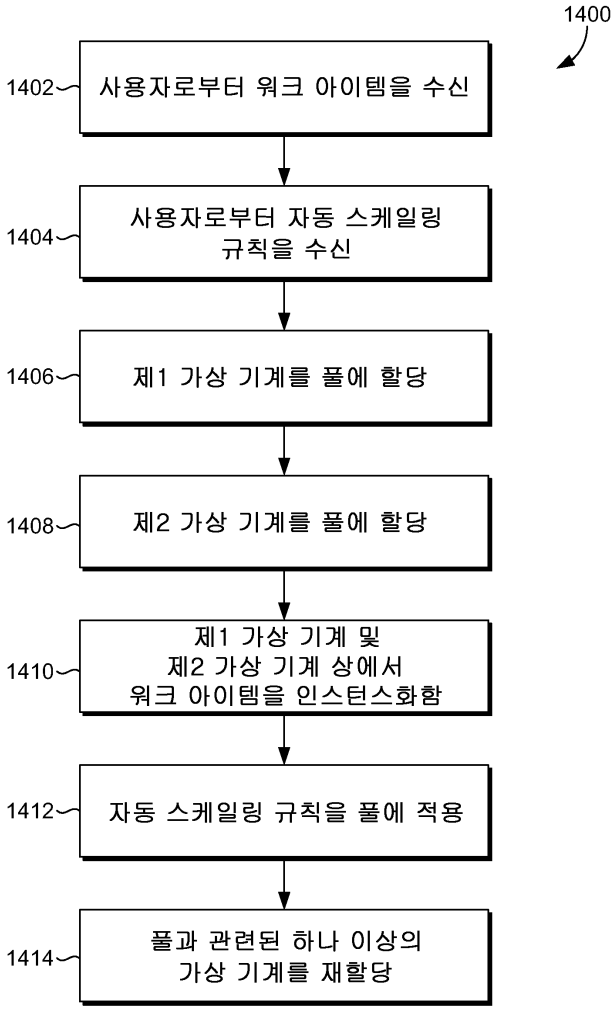
도면12


1200
1202
사용자로부터 태스크 계정을 생성하기 위한 요청을 수신
1204
태스크 계정과 관련된 워크 아이템을 수신
1206
워크 아이템과 관련된 규칙들을 수신
1208
워크 아이템에 기초하여 작업을 자동 생성
1210
규칙들에 기초하여 가상 기계들을 풀에 자동으로 할당
1212
작업을 포함하는 작업 큐를 자동으로 생성
1214
작업을 풀에 대해 스케줄링하고 작업의 태스크들을 가상 기계들에 할당함으로써 풀 내의 가상 기계들 상에서 작업을 인스턴스화함
1216
워크 아이템 또는 작업의 완료시에 풀을 자동으로 삭제

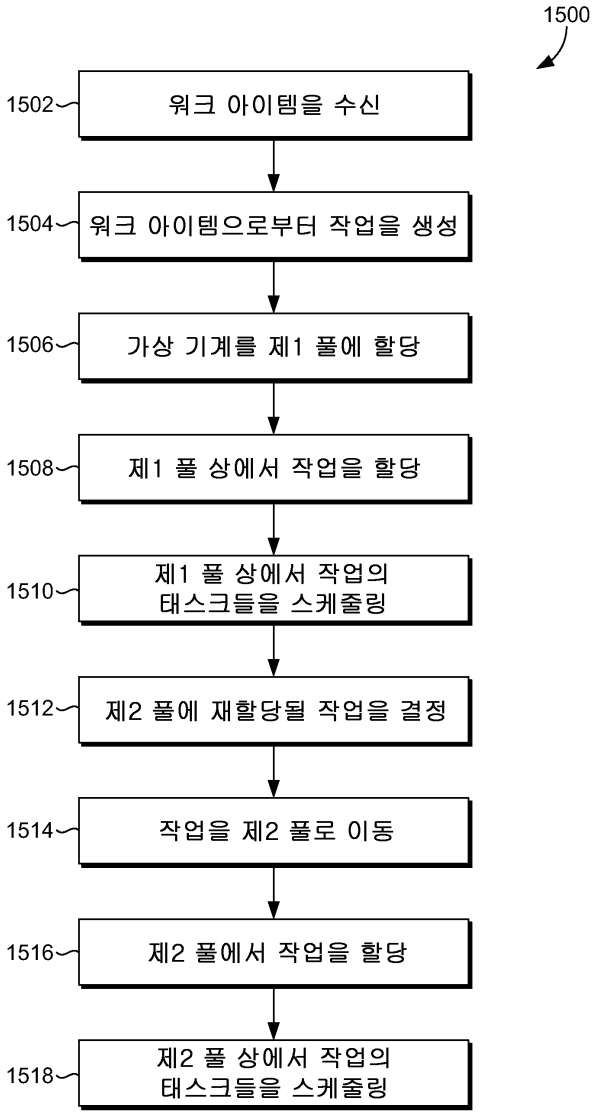
도면13


1300
1302
풀과 관련된 가상 기계들의 수를 결정
1304
자동 스케일링 규칙들을 적용하여, 풀과 관련된 가상 기계들의 수를 조정
1306
풀의 하나 이상의 가상 기계를 동적으로 할당

도면14


1400
1402
사용자로부터 워크 아이템을 수신
1404
사용자로부터 자동 스케일링 규칙을 수신
1406
제1 가상 기계를 풀에 할당
1408
제2 가상 기계를 풀에 할당
1410
제1 가상 기계 및 제2 가상 기계 상에서 워크 아이템을 인스턴스화함
1412
자동 스케일링 규칙을 풀에 적용
1414
풀과 관련된 하나 이상의 가상 기계를 재할당

도면15


1500
1502
워크 아이템을 수신
1504
워크 아이템으로부터 작업을 생성
1506
가상 기계를 제1 풀에 할당
1508
제1 풀 상에서 작업을 할당
1510
제1 풀 상에서 작업의 태스크들을 스케줄링
1512
제2 풀에 재할당될 작업을 결정
1514
작업을 제2 풀로 이동
1516
제2 풀에서 작업을 할당
1518
제2 풀 상에서 작업의 태스크들을 스케줄링