(19) 대한민국특허청(KR)
(12) 등록특허공보(B1)

(45) 공고일자 2016년02월05일
(11) 등록번호 10-1592479
(24) 등록일자 2016년02월01일

(51) 국제특허분류(Int. Cl.)
G06F 17/00 (2006.01) *G06F 12/08* (2016.01)
G06F 15/16 (2006.01) *G06F 17/30* (2006.01)
(21) 출원번호 10-2013-7020080
(22) 출원일자(국제) 2011년11월30일
심사청구일자 2015년07월02일
(85) 번역문제출일자 2013년07월29일
(65) 공개번호 10-2013-0143706
(43) 공개일자 2013년12월31일
(86) 국제출원번호 PCT/US2011/062609
(87) 국제공개번호 WO 2012/091846
국제공개일자 2012년07월05일
(30) 우선권주장
13/227,381 2011년09월07일 미국(US)
61/428,799 2010년12월30일 미국(US)
(56) 선행기술조사문헌
US20020166031 A1

(73) 특허권자
페이스북, 인크.
미국, 캘리포니아 94025, 멘로 파크, 윌로우 로드 1601
(72) 발명자
벤카타라마니 벤카테쉬와란
미국 캘리포니아 94304 팔로 알토 사우스 캘리포니아 애비뉴 1601
(74) 대리인
방해철, 김용인

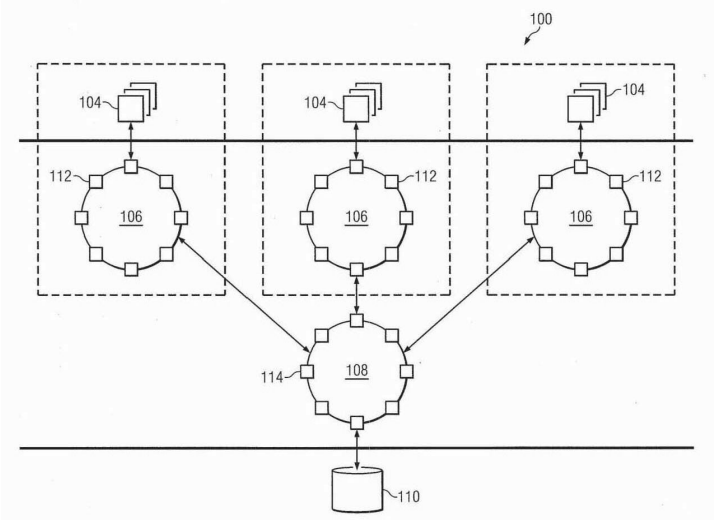
전체 청구항 수 : 총 20 항 심사관 : 이석형

(54) 발명의 명칭 **그래프 데이터용 분산형 캐시**

(57) 요 약

본 발명은 노드 및 그래프에서 에지가 연결하는 노드들 사이의 연관 또는 관계를 정의하는 에지를 포함하는 그래프로 모델링된 정보를 저장하고 제공하는 분산형 캐시 시스템에 관한 것이다.

대 표 도



명 세 서

청구범위

청구항 1

제 1 캐시 노드에서, 제 1 노드와 제 2 노드 사이의 연결(association)을 추가하는 요청을 수신하는 단계;

제 1 캐시 노드의 메모리에 제 1 노드와 제 2 노드 사이의 연결을 나타내는 데이터를 저장하는 단계; 및

메시지를 제 2 캐시 노드로 전송하는 단계를 포함하며,

제 1 노드는 클러스터의 제 1 캐시 노드로 할당된 제 1 식별자 범위에서 제 1 식별자에 의해 식별되고, 제 2 노드는 클러스터의 제 2 캐시 노드로 할당된 제 2 식별자 범위에서 제 2 식별자에 의해 식별되며,

상기 메시지에 응답하여, 제 2 캐시 노드는 제 2 캐시 노드의 메모리에 제 1 노드 및 제 2 노드 사이의 연결을 추가하는 방법.

청구항 2

제 1 항에 있어서,

제 2 식별자에 해당하는 샤드(shard) 식별자를 식별하는 단계를 더 포함하며, 상기 제 2 캐시 노드는 식별된 샤드 식별자를 저장하는 방법.

청구항 3

제 1 항에 있어서,

메시지가 제 1 노드 및 제 2 노드 사이의 양방향성 연결을 캐시 층에 설정(establish)하는데 필요한 업데이트를 포함한다고, 상기 메시지가 제 2 캐시 노드로 신호 보내는 방법.

청구항 4

제 1 항에 있어서,

제 1 식별자와 관련된 샤드 식별자에 해당하는 리더(leader) 캐시 노드로 메시지를 전달하는 단계를 더 포함하는 방법.

청구항 5

제 1 항에 있어서,

제 1 및 제 2 노드가 하나 내지 복수의 노드 타입을 포함하는 그래프 구조에 포함되는 방법.

청구항 6

제 1 항에 있어서,

상기 연결은 복수의 연결 타입 중 식별된 연결 타입인 방법.

청구항 7

제 1 항에 있어서,

복수의 노드 중 제 1 노드 및 복수의 연결 타입 중 하나의 연결 타입에 해당하는 각 연결 세트에 대해, 메모리에서 제 1 인덱스 및 제 2 인덱스를 관리하는 단계; 및

제 1 연결 타입 및 제 1 노드 식별자에 대하여 메모리에 접근하여, 제 2 노드 식별자를 제 1 연결 타입과 제 1 노드 식별자에 해당하는 제 1 인덱스 및 제 2 인덱스에 추가하는 단계를 더 포함하며,

상기 요청은 연결 타입을 식별하며,

제 1 인덱스는 순서화된 엔트리 어레이를 포함하고, 각 엔트리는 제 1 노드와 관련된 제 2 노드의 노드 식별자 및 정렬(sorting) 속성을 포함하며,

제 2 인덱스는 제 1 노드와 관련된 개별 제 2 노드의 노드 식별자에 해당하는 엔트리를 포함하는 해쉬 테이블(hash table)을 포함하는 방법.

청구항 8

하나 이상의 프로세서와;

메모리와;

실행시, 하나 이상의 프로세서가: 제 1 노드와 제 2 노드 사이의 연결(association)을 추가하는 요청을 수신하며; 캐시 노드의 메모리에 제 1 노드 및 제 2 노드 사이의 연결을 나타내는 데이터를 저장하고; 메시지를 제 2 캐시 노드에 전송하도록 동작하는 컴퓨터 판독가능한 명령어를 저장하는 비-일시적 저장 매체를 포함하며,

제 1 노드는 클러스터의 캐시 노드로 할당된 제 1 식별자 범위에서 제 1 식별자에 의해 식별되고, 제 2 노드는 클러스터의 제 2 캐시 노드로 할당된 제 2 식별자 범위에서 제 2 식별자에 의해 식별되며,

상기 메시지에 응답하여, 제 2 캐시 노드는 제 2 캐시 노드의 메모리에 제 1 노드 및 제 2 노드 사이의 연결을 추가하는, 캐시 노드들의 클러스터에서 동작하는 캐시 노드.

청구항 9

제 8 항에 있어서,

상기 명령어는 하나 이상의 프로세서가 제 2 식별자에 해당하는 샤드 식별자를 식별하도록 더 동작하며,

제 2 캐시 노드는 식별된 샤드 식별자를 저장하는, 캐시 노드들의 클러스터에서 동작하는 캐시 노드.

청구항 10

제 8 항에 있어서,

메시지가 제 1 노드 및 제 2 노드 사이의 양방향성 연결을 캐시 층에 설정(establish)하는데 필요한 업데이트를 포함한다고, 상기 메시지가 제 2 캐시 노드로 신호 보내는는, 캐시 노드들의 클러스터에서 동작하는 캐시 노드.

청구항 11

제 8 항에 있어서,

상기 명령어는 하나 이상의 프로세서가 제 1 식별자와 관련된 샤드 식별자에 해당하는 리더 캐시 노드로 메시지를 전달하도록 더 동작하는, 캐시 노드들의 클러스터에서 동작하는 캐시 노드.

청구항 12

제 8 항에 있어서,

제 1 및 제 2 노드가 하나 내지 복수의 노드 타입을 포함하는 그래프 구조에 포함되는, 캐시 노드들의 클러스터에서 동작하는 캐시 노드.

청구항 13

제 8 항에 있어서,

상기 연결은 복수의 연결 타입 중 식별된 연결 타입인, 캐시 노드들의 클러스터에서 동작하는 캐시 노드.

청구항 14

제 8 항에 있어서,

상기 명령어는 하나 이상의 프로세서가: 복수의 노드 중 제 1 노드 및 복수의 연결 타입 중 하나의 연결 타입에 해당하는 각 연결 세트에 대해, 메모리에서 제 1 인덱스 및 제 2 인덱스를 관리하며; 제 1 연결 타입 및 제 1 노드 식별자에 대하여 메모리에 접근하여, 제 2 노드 식별자를 제 1 연결 타입과 제 1 노드 식별자에 해당하는

제 1 인덱스 및 제 2 인덱스에 추가하도록 더 동작하며,

상기 요청은 연결 타입을 식별하고,

제 1 인덱스는 순서화된 엔트리 어레이를 포함하며, 각 엔트리는 제 1 노드와 관련된 제 2 노드의 노드 식별자 및 정렬 속성을 포함하고,

제 2 인덱스는 제 1 노드와 관련된 개별 제 2 노드의 노드 식별자에 해당하는 엔트리를 포함하는 해쉬 테이블을 포함하는, 캐시 노드들의 클러스터에서 동작하는 캐시 노드.

청구항 15

실행시, 하나 이상의 프로세서가:

제 1 캐시 노드에서, 제 1 노드와 제 2 노드 사이의 연결(association)을 추가하는 요청을 수신하며;

제 1 캐시 노드의 메모리에 제 1 노드 및 제 2 노드 사이의 연결을 나타내는 데이터를 저장하고;

메시지를 제 2 캐시 노드에 전송하도록 동작하는 컴퓨터 판독가능한 명령어를 저장하는 비-일시적 저장 매체로서,

제 1 노드는 클러스터의 제 1 캐시 노드로 할당된 제 1 식별자 범위에서 제 1 식별자에 의해 식별되고, 제 2 노드는 클러스터의 제 2 캐시 노드로 할당된 제 2 식별자 범위에서 제 2 식별자에 의해 식별되며,

상기 메시지에 응답하여, 제 2 캐시 노드는 제 2 캐시 노드의 메모리에 제 1 노드 및 제 2 노드 사이의 연결을 추가하는, 컴퓨터 판독가능한 명령어를 저장하는 비-일시적인 저장 매체.

청구항 16

제 15 항에 있어서,

상기 명령어는 하나 이상의 프로세서가 제 2 식별자에 해당하는 샤드 식별자를 식별하도록 더 동작하며,

제 2 캐시 노드는 식별된 샤드 식별자를 저장하는, 컴퓨터 판독가능한 명령어를 저장하는 비-일시적인 저장 매체.

청구항 17

제 15 항에 있어서,

메시지가 제 1 노드 및 제 2 노드 사이의 양방향성 연결을 캐시 층에 설정(establish)하는데 필요한 업데이트를 포함한다고, 상기 메시지가 제 2 캐시 노드로 신호 보내는, 컴퓨터 판독가능한 명령어를 저장하는 비-일시적인 저장 매체.

청구항 18

제 15 항에 있어서,

상기 명령어는 하나 이상의 프로세서가 제 1 식별자와 관련된 샤드 식별자에 해당하는 리더 캐시 노드로 메시지를 전달하도록 더 동작하는, 컴퓨터 판독가능한 명령어를 저장하는 비-일시적인 저장 매체.

청구항 19

제 15 항에 있어서,

상기 연결은 복수의 연결 타입 중 식별된 연결 타입인, 컴퓨터 판독가능한 명령어를 저장하는 비-일시적인 저장 매체.

청구항 20

제 15 항에 있어서,

상기 명령어는 하나 이상의 프로세서가: 복수의 노드 중 제 1 노드 및 복수의 연결 타입 중 하나의 연결 타입에 해당하는 각 연결 세트에 대해, 메모리에서 제 1 인덱스 및 제 2 인덱스를 관리하며; 제 1 연결 타입 및 제 1

노드 식별자에 대하여 메모리에 접근하여, 제 2 노드 식별자를 제 1 연결 타입과 제 1 노드 식별자에 해당하는 제 1 인덱스 및 제 2 인덱스에 추가하도록 더 동작하며,

상기 요청은 연결 타입을 식별하고,

제 1 인덱스는 순서화된 엔트리 어레이를 포함하며, 각 엔트리는 제 1 노드와 관련된 제 2 노드의 노드 식별자 및 정렬 속성을 포함하고,

제 2 인덱스는 제 1 노드와 관련된 개별 제 2 노드의 노드 식별자에 해당하는 엔트리를 포함하는 해쉬 테이블을 포함하는, 컴퓨터 판독가능한 명령어를 저장하는 비-일시적인 저장 매체.

발명의 설명

기 술 분 야

[0001] 본 발명은 일반적으로 그래프 데이터의 저장 및 제공에 관한 것이며, 더 상세하게는 분산형 캐시 시스템을 갖는 그래프 데이터의 저장 및 제공에 관한 것이다.

배 경 기 술

[0002] 컴퓨터 사용자들은 가령 인터넷과 같은 공중 네트워크뿐만 아니라 사설 네트워크를 포함하는 다양한 근거리 및 광역 컴퓨터 네트워크를 통해 방대한 양의 정보에 접근하고 정보를 공유할 수 있다. 통상, 사용자의 컴퓨팅 장치에 설치되는 웹 브라우저는 예컨대 관련 URLs(uniform resource locators)에 의해 식별되는 다양한 네트워크 서버에 위치한 정보로의 접근 및 정보와의 상호작용을 용이하게 한다. 사용자-생성 컨텐츠의 공유를 가능하게 하는 종래의 접근방법은 가령 소셜 네트워킹 웹사이트와 같은 다양한 정보 공유 기술 또는 플랫폼을 포함한다. 이런 웹사이트는 다른 사용자에 의해 생성되거나 맞춤화된 웹 페이지를 사용자에게 보여줄 수 있는 애플리케이션용 플랫폼을 포함하거나, 플랫폼과 연결되거나, 플랫폼을 제공할 수 있는데, 여기서 다른 사용자에 의한 가시성 및 이런 페이지와의 상호작용은 일부 특징적인 규정 세트(set of rules)에 의해 통제된다.

[0003] 이런 소셜 네트워킹 정보 및 일반적인 대부분의 정보는 통상 관계형 데이터베이스(relational databases)에 저장된다. 일반적으로, 관계형 데이터베이스는 관계들의 모음(흔히 '테이블(tables)'이라 함)이다. 관계형 데이터베이스는 한 세트의 수학적 용어를 사용하며, 구조화 질의어(Structured Query Language, SQL) 데이터베이스 용어를 사용할 수 있다. 예컨대, 관계는 동일한 속성을 갖는 한 세트의 튜플(tuples)로 정의될 수 있다. 튜플은 보통 객체 및 그 객체에 대한 정보를 나타낸다. 관계는 보통 행들 및 열들로 체계화된 테이블로 기술된다. 일반적으로, 속성에 의해 참조되는 모든 데이터는 동일한 도메인에 있으며 동일한 제약을 따른다.

[0004] 관계형 모델은 관계의 튜플들이 특정한 순서를 가지지 않으며, 결국 튜플들이 속성에 순서를 부과하지 않도록 지정한다. 애플리케이션은 튜플을 식별하고 속성을 식별하며 관계들을 결합시키는 동작을 사용하는 쿼리를 지정함으로써 데이터에 접근한다. 관계들은 변경될 수 있으며, 새로운 튜플이 명시적인 값을 공급하거나 쿼리로부터 도출될 수 있다. 마찬가지로, 업데이트하거나 삭제하기 위해, 튜플은 쿼리를 식별할 수 있다. 관계의 각 튜플은 (하나 이상의) 속성값의 일부 결합으로 고유하게 식별될 필요가 있다. 이런 결합을 기본 키(primary key)라고 한다. 관계형 데이터베이스에서, 모든 데이터는 관계를 통해 저장되고 접근된다. 데이터를 저장하는 관계는 일반적으로 테이블로 구현되거나 테이블이라고 일컬어진다.

[0005] 관계형 데이터베이스 관리 시스템에서 구현되는 바와 같이, 관계형 데이터베이스는 예컨대, 재무 기록, 제조 및 수송 정보(manufacturing and logistical information), 인사 데이터(personnel data) 및 다른 애플리케이션을 위해 사용되는 데이터베이스의 정보 저장용으로 지배적인 선택이 되어왔다. 컴퓨터 전력이 증가함에 따라, 초기에 관계형 데이터베이스를 비실용적으로 만들었던 관계형 데이터베이스의 비효율성이 종래의 애플리케이션에 대한 사용의 용이함보다 더 중요해졌다. 3개의 주된 오픈 소스 구현은 MySQL, PostgreSQL 및 SQLite이다. MySQL은 다중 사용자에게 다수의 데이터베이스로의 접근을 제공하는 서버로서 작동하는 관계형 데이터베이스 관리 시스템(RDBMS)이다. 대중적인 LAMP 소프트웨어 스택에서 두문자어 "M"은 MySQL을 말한다. 웹 애플리케이션과의 사용용으로 MySQL의 선호도(popularity)는 PHP(LAMP에서 "P")의 선호도와 밀접하게 연관된다. 여러 높은-트래픽 웹사이트는 데이터 저장 및 사용자 데이터의 로깅(logging)을 위해 MySQL을 사용한다.

[0006] 관계형 데이터베이스와의 통신은 흔히 속도 장애(speed bottleneck)이기 때문에, 많은 네트워크는 캐싱 시스템을 이용하여 특정한 정보 쿼리를 제공한다. 예컨대, 멤캐시드(Memcached)는 범용 분산형 메모리 캐싱 시스템이다. 멤캐시드는 외부 데이터 소스(가령 데이터베이스 또는 API)가 판독되어야 하는 횟수를 줄이기 위해, RAM 내

데이터 및 객체를 캐싱하여 동적 데이터베이스-구동 웹사이트의 속도를 올리는데 흔히 사용된다. 멤캐시드의 API(들)는 다중 머신에 걸쳐 분산된 거대한 해쉬 테이블(hash table)을 제공한다. 테이블이 가득 찬 경우, 이후의 삽입에 의해 더 오래된 데이터가 최소 최근 사용(least recently used, LRU) 순서로 제거된다. 통상, 멤캐시드를 사용하는 애플리케이션은 가령 데이터베이스와 같은 슬로어 백킹 스토어(slower backing store)로 물러나기 전에, 요청과 추가를 코어(core)에 쌓는다.

[0007] 멤캐시드 시스템은 클라이언트-서버 아키텍처를 사용한다. 서버는 키-값 연관 어레이(key-value associative array)를 관리하며; 클라이언트는 이런 어레이를 채우고 이를 조회(query)한다. 클라이언트는 클라이언트 측 라이브러리를 사용하여 서버에 접속한다. 통상, 각 클라이언트는 모든 서버를 인지하며 서버는 서로 통신하지 않는다. 클라이언트가 특정 키에 해당하는 값을 설정하거나 판독하고자 한다면, 먼저 클라이언트의 라이브러리는 키의 해쉬를 계산하여 사용될 서버를 결정한다. 이후, 클라이언트는 그 서버에 접속한다. 서버는 키의 제 2 해쉬를 계산하여 해당 값을 저장하거나 판독할 장소를 결정한다. 통상, 서버는 RAM 내에 값을 유지한다; 서버가 RAM을 모두 사용해버리면, 서버는 가장 오래된 값을 버린다. 따라서, 클라이언트는 멤캐시드를 일시적 캐시로 다루어야 한다; 클라이언트는 멤캐시드에 저장된 데이터가 필요시 여전히 거기에 있다라고 추정할 수 없다.

발명의 내용

해결하려는 과제

[0008] 본 발명은 일반적으로 그래프 데이터의 저장 및 제공에 관한 것이며, 더 상세하게는 분산형 캐시 시스템을 갖는 그래프 데이터의 저장 및 제공에 관한 것이다.

과제의 해결 수단

[0009] 특정 실시예는 노드 및 그래프에서 에지가 연결하는 노드들 사이의 연관 또는 관계를 정의하는 에지를 포함하는 그래프로 모델링된 정보를 저장하고 제공하는 분산형 캐시 시스템에 관한 것이다.

발명의 효과

[0010] 본 발명의 내용 중에 포함되어 있다.

도면의 간단한 설명

[0011] 도 1은 본 발명의 하나의 구현에 따라 예시적인 캐싱 시스템 아키텍처를 도시한다.

도 2는 예시적인 컴퓨터 시스템 아키텍처를 도시한다.

도 3은 예시적인 네트워크 환경을 제공한다.

도 4는 새로운 연관을 그래프에 추가하기 위한 예시적인 방법을 도시하는 흐름도를 나타낸다.

도 5는 캐싱 시스템의 다양한 구성요소 사이의 예시적인 메시지 흐름을 도시하는 개략도이다.

도 6은 그래프 데이터의 변화를 처리하기 위한 예시적인 방법을 도시하는 흐름도를 나타낸다.

도 7은 캐싱 시스템의 다양한 구성요소 사이의 예시적인 메시지 흐름을 도시하는 개략도이다.

발명을 실시하기 위한 구체적인 내용

[0012] 특정 실시예로, 그래프는 소셜 그래프이거나 소셜 그래프를 포함하며, 인덱싱 시스템은 통합 소셜 네트워크 환경을 가능하게 하는 더 큰 네트워킹 시스템, 기반구조 또는 플랫폼의 일부이다. 본 명세서에서, 소셜 네트워크 환경은 소셜 그래프 정보를 포함하는 소셜 그래프의 관점에서 기술될 수 있다. 실제로, 본 명세서의 특정 실시예는 소셜 네트워크 환경에 의해 또는 소셜 네트워크 환경용으로 저장된 데이터의 대부분 또는 전부가 소셜 그래프로 표현될 수 있다는 점을 필요로 하거나, 활용하거나, 이용한다. 특정 실시예는 본 명세서에 기술된 바와 같이, 기하급수적으로 증가하는 소셜 네트워크 환경의 다수의 사용자들에 맞게 효율적으로, 지능적으로 및 성공적으로 조정(scale)될 수 있는 비용-효과적인 기반구조를 제공한다.

[0013] 특정 실시예로, 본 명세서에 기술되는 분산형 인덱싱 시스템 및 백엔드(backend) 기반구조는 하나 이상의: 스케일(scale)에서의 낮은 대기시간, 요청 당 낮은 비용, 개발자용 프레임워크의 사용 용이성, 멀티-마스터를 지원하는 기반구조, 하이퍼텍스트 전처리기(PHP) 이외의 언어로 작성된 저장 데이터로의 접근을 클라이언트에게 제

공하는 기반구조, 본 명세서에서 예로써 기술되는 바와 같이 소셜 그래프의 연관(에지) 및 객체(노드) 모두를 포함하는 결합형 쿼리를 가능하게 하는 기반구조, 및 다른 지속성 데이터 스토어(persistent data stores)를 다른 타입의 데이터용으로 사용될 수 있도록 하는 기반구조를 제공한다. 게다가, 특정 실시예는 하나 이상의: 캐싱+지속성+복제 기반구조로부터 데이터 접속 API의 무결한 분리를 가능하게 하는 기반구조, 라이트-스루/리드-스루(write-through/read-through) 캐싱을 지원하는 기반구조, 연산을 데이터에 더 근접하게 이동시키는 기반구조, 저장 스키마(schemas) 및 백 엔드(back ends)로의 투과적 이전(transparent migration)을 가능하게 하는 기반구조 및 데이터 객체 접근의 효율성을 향상시키는 기반구조를 제공한다.

[0014] 추가로, 본 명세서에서 사용되는 바와 같이, "또는"은 "또는"뿐만 아니라 "및"을 의미할 수 있다, 즉 명시적으로 언급되거나 함축적으로 암시되는 것이 아니면, "또는"이 반드시 "및"을 제외하는 것은 아니다.

[0015] 특정 실시예는 다중 네트워크 주소화 시스템을 포함하는, 가령 인터넷과 같은 광역 네트워크 환경에서 동작할 수 있다. 도 3은 다양한 예시적인 실시예들이 동작할 수 있는 예시적인 네트워크 환경을 도시한다. 네트워크 클라우드(60)는 일반적으로 본 명세서에 기술된 시스템 및 호스트가 통신할 수 있는 하나 이상의 상호연결된 네트워크를 나타낸다. 네트워크 클라우드(60)는 패킷-기반 광역 네트워크(가령 인터넷), 사설 네트워크, 무선 네트워크, 위성 네트워크, 셀룰러 네트워크, 무선호출(paging) 네트워크 등을 포함할 수 있다. 도 3이 도시하는 바와 같이, 특정 실시예는 소셜 네트워킹 시스템(20)과 하나 이상의 클라이언트 장치(30)를 포함하는 네트워크 환경에서 동작할 수 있다. 클라이언트 장치(30)는 네트워크 서비스 제공자, 무선 반송파 또는 임의의 다른 적합한 수단을 통해 네트워크 환경으로 동작가능하게 연결된다.

[0016] 하나의 예시적인 실시예로, 소셜 네트워킹 시스템(20)은 본 명세서에 기술되는 바와 같이, 사용자들이 서로 통신하거나 상호작용하며, 가령 사용자 프로파일과 같은 컨텐츠에 접근할 수 있도록 해주는 컴퓨팅 시스템을 포함한다. 소셜 네트워킹 시스템(20)은 다양한 예시적인 실시예에서 하나 이상의 물리적 서버(22) 및 데이터 스토어(24)를 포함하는 네트워크 주소화 시스템이다. 하나 이상의 물리적 서버(22)는 예로써 한 세트의 라우터 및/또는 네트워킹 스위치(26)를 통해 컴퓨터 네트워크(60)와 동작가능하게 연결된다. 예시적인 실시예로, 하나 이상의 물리적 서버(22)에 의해 호스팅되는 기능은 웹 또는 HTTP 서버, FTP 서버뿐만 아니라, 제한 없이, 공통 게이트웨이 인터페이스(CGI) 스크랩트를 사용하여 구현되는 웹 페이지와 애플리케이션, PHP 하이퍼-텍스트 전처리기(PHP), 액티브 서버 페이지(ASP), 하이퍼 텍스트 마크업 언어(HTML), 확장형 마크업 언어(XML), 자바, 자바스크립트, 비동기 자바스크립트와 XML(AJAX) 등을 포함할 수 있다.

[0017] 물리적 서버(22)는 소셜 네트워킹 시스템(20)의 동작에 관한 기능을 호스트할 수 있다. 예로써, 소셜 네트워킹 시스템(20)은 하나 이상의 클라이언트 장치(30)에서 하나 이상의 사용자들이 정보를 열람 및 포스트할 뿐만 아니라 웹사이트를 통해 서로 통신할 수 있도록 해주는 웹사이트를 호스트할 수 있다. 서버(22)가 예컨대 소셜 네트워킹 시스템(20)을 호스팅하는 많은 서버들뿐만 아니라 다른 컨텐츠 분배 서버, 데이터 스토어 및 데이터베이스를 포함할 수 있지만, 이하에서, 서버들(22)은 서버(22)라고 한다. 데이터 스토어(24)는 디지털 데이터 객체로서 소셜 네트워킹 시스템의 동작과 관련되고 동작가능하게 해주는 컨텐츠 및 데이터를 저장할 수 있다. 특정한 구현으로, 데이터 객체는 데이터 파일, 데이터베이스 또는 레코드에 통상 저장되거나 포함되는 디지털 정보의 아이템이다. 컨텐츠 객체는 텍스트(예컨대, ASCII, SGML, HTML), 이미지(예컨대, jpeg, tif 및 gif), 그래픽(벡터-기반 또는 비트맵), 오디오, 비디오(예컨대, mpeg) 또는 다른 멀티미디어, 및 이들의 조합을 포함하는 많은 형태를 취할 수 있다. 또한, 컨텐츠 객체 데이터는 실행가능한 코드 객체(예컨대, 브라우저 윈도우 또는 프레임 내의 실행가능한 게임), 팟캐스트(podcast) 등을 포함할 수 있다. 논리적으로, 데이터 스토어(24)는 하나 이상의 물리적 시스템에 저장된 논리적으로 관계된 레코드 또는 파일의 통합 모음으로서 정보를 관리하는, 가령 관계형 데이터베이스 및 객체-지향형 데이터베이스와 같은 하나 이상의 다양한 별개의 통합된 데이터베이스에 해당한다. 구조적으로, 데이터 스토어(24)는 하나 이상의 많은 계층의 데이터 저장 및 관리 시스템을 통상 포함할 수 있다. 특정 실시예로, 데이터 스토어(24)는 가령 하나 이상의 데이터베이스 서버, 대량 저장 미디어, 미디어 라이브러리 시스템, 저장 영역 네트워크, 데이터 저장 클라우드 등과 같은 구성요소를 포함하는 임의의 적합한 물리적 시스템(들)에 의해 구현될 수 있다. 일예의 실시예에서, 데이터 스토어(24)는 하나 이상의 서버, 데이터베이스(예컨대, MySQL) 및/또는 데이터 웨어하우스(warehouses)를 포함할 수 있다.

[0018] 데이터 스토어(24)는 서로 다른 소셜 네트워킹 시스템(20) 사용자 및/또는 클라이언트 장치(30)와 관련된 데이터를 포함할 수 있다. 특정 실시예로, 소셜 네트워킹 시스템(20)은 시스템(20)의 각각의 사용자에 대한 사용자 프로파일을 관리한다. 사용자 프로파일은 소셜 네트워크의 사용자를 기술하는 데이터를 포함하는데, 이런 데이터는 예컨대, 적절한 이름(사람의 이름, 중간 이름 및 성, 상표명 및/또는 비즈니스 엔티티의 회사명 등), 가령 경력, 학력, 취미나 선호도, 지리적 위치 및 추가적인 설명적 데이터와 같은 인명 정보, 인구학적 정보 및 다른

유형의 설명적 정보를 포함할 수 있다. 예로써, 사용자 프로파일은 사용자의 생일, 관계 상태, 거주 도시 등을 포함할 수 있다. 시스템(20)은 다른 사용자들 사이의 하나 이상의 관계를 기술하는 데이터를 더 저장할 수 있다. 관계 정보는 유사하거나 공통의 경력, 그룹 회원(membership), 취미 또는 학력을 갖는 사용자들을 표시할 수 있다. 또한, 사용자 프로파일은 다른 사용자들에 의한 사용자의 정보로의 접근을 통제하는 개인정보 설정을 포함할 수 있다.

[0019] 클라이언트 장치(30)는 일반적으로 컴퓨터 네트워크에서 (예컨대, 원격으로) 통신하기 위한 기능을 포함하는 컴퓨터 또는 컴퓨팅 장치이다. 클라이언트 장치(30)는 다른 적합한 컴퓨팅 장치들 가운데, 데스크톱 컴퓨터, 랩톱 컴퓨터, 태블릿, 개인 정보 단말기(PDA), 차량 내외 항법시스템, 스마트폰이나 다른 셀룰러폰 또는 모바일폰, 또는 모바일 게임장치일 수 있다. 클라이언트 장치(30)는 가령 웹 브라우저(예컨대, 마이크로소프트 윈도우 인터넷 익스플로러, 모질라 파이어폭스, 애플 사파리, 구글 크롬 및 오페라 등)와 같은 하나 이상의 클라이언트 애플리케이션을 실행하여 컴퓨터 네트워크를 통해 컨텐츠에 접근하고 컨텐츠를 열람할 수 있다. 특정한 구현으로, 클라이언트 애플리케이션은 클라이언트 장치(30)의 사용자가 가령 소셜 네트워킹 시스템(20)에 의해 호스팅된 자원과 같이 검색되는 특정 네트워크 자원의 주소로 들어갈 수 있도록 해준다. 이들 주소는 URL(Uniform Resource Locators)일 수 있다. 또한, 일단 페이지나 다른 자원이 검색되었다면, 클라이언트 애플리케이션은 사용자가 다른 자원으로의 하이퍼링크를 "클릭"하는 경우 다른 페이지나 레코드로의 접근을 제공할 수 있다. 예로써, 이런 하이퍼링크는 웹 페이지 내에 위치할 수 있으며, 사용자가 또 다른 페이지의 URL에 들어가고 그 페이지를 검색하는 자동화 방식을 제공할 수 있다.

[0020] 도 1은 도 3에 도시된 소셜 네트워킹 시스템(20)의 백 엔드 기능을 구현할 수 있는 네트워킹 시스템, 아키텍처 또는 기반구조(100)(이하, 네트워킹 시스템(100)이라 한다)의 예시적인 실시예를 도시한다. 특정 실시예로, 네트워킹 시스템(100)은 네트워킹 시스템(100)의 사용자들이 네트워킹 시스템(100)에 의해 제공되는 소셜 네트워킹 서비스를 통해 서로 간에 그리고 제 3 자와 상호작용할 수 있도록 해준다. 예컨대, 원격 사용자 컴퓨팅 장치(예컨대, 개인용 컴퓨터, 넷북, 멀티미디어 장치, 셀룰러폰(특히 스마트폰) 등)에서 사용자들은 웹 브라우저 또는 다른 사용자 클라이언트 애플리케이션을 통해 네트워킹 시스템(100)에 접근하여, 정보를 열람하거나 정보를 저장 또는 업데이트하거나 정보를 통신하거나 다른 사용자들, 제 3 자 웹사이트, 웹 페이지 또는 웹 애플리케이션과 상호작용하도록 네트워킹 시스템(100)에 의해 적어도 부분적으로 호스팅되거나 접근가능한 웹사이트, 웹 페이지 또는 웹 애플리케이션에 접근하거나, 네트워킹 시스템(100)에 의해 저장되거나 호스팅되거나 접근가능한 다른 정보에 접근할 수 있다. 특정 실시예로, 하기에 더 상세히 기술되는 바와 같이, 네트워킹 시스템(100)은 사용자, 개념, 토픽 및 다른 정보(데이터)를 나타내는 그래프 노드뿐만 아니라, 그래프 노드 사이의 관계를 연결하거나 정의하는 그래프 에지를 포함하는 그래프를 관리한다.

[0021] 도 1 및 5를 참조하면, 특정 실시예로, 네트워킹 시스템(100)은 하나 이상의 데이터 센터(102)를 포함한다. 예컨대, 네트워킹 시스템(100)은 개별 영역 내에 위치한 사용자들을 제공하기 위한 다양한 지리적 영역 내에 전략적으로 위치한 복수의 데이터 센터(102)를 포함할 수 있다. 특정 실시예로, 각 데이터 센터는 네트워킹 시스템(100)의 사용자들에게 그리고 사용자들로부터 정보를 통신하는 다수의 클라이언트 또는 웹 서버(104)(이하에서는 클라이언트 서버(103))를 포함한다. 예컨대, 원격 사용자 컴퓨팅 장치에서 사용자들은 부하 분산기(load balancers)를 통해 클라이언트 서버(104) 또는 네트워크와 서비스 제공자의 임의의 적합한 조합을 통해 다른 적합한 시스템과 통신할 수 있다. 클라이언트 서버(104)는 데이터를 검색하여 사용자 요청에 응답하기 위한 구조화된 문서를 생성하기 위해 본 명세서에 기술된 인덱스 및 데이터베이스 시스템을 쿼리(query)할 수 있다.

[0022] 클라이언트 서버(104)들 각각은 하나 이상의 팔로어 분산형 캐시 클러스터 또는 링(106)(이하에서는 팔로어 캐시 클러스터(follower cache clusters)(106))과 통신한다. 도시된 실시예에서, 데이터 센터는 웹 서버(104)의 서브세트를 각각 제공하는 3개의 팔로어 캐시 클러스터(106)를 포함한다. 특정 실시예로, 팔로어 캐시 클러스터(106) 및 팔로어 캐시 클러스터(106)가 제공하는 클라이언트 서버(104)가 가령 빌딩, 방 또는 다른 중심 위치 내에서 아주 근접하게 위치하여, 기반구조(예컨대, 와이어 또는 다른 통신선 등)와 관련된 비용뿐만 아니라 클라이언트 서버(104)와 개별 제공 팔로어 캐시 노드 클러스터(106) 사이의 대기시간을 감소시킨다. 그러나, 일부 실시예로, 각각의 팔로어 캐시 클러스터(106)와 이들이 각각 제공하는 클라이언트 서버(104)는 중심 위치(centralized location) 내에 위치할 수 있는 한편, 각각의 팔로어 캐시 클러스터(106)와 팔로어 캐시 클러스터(106)가 각각 제공하는 개별 클라이언트 서버(104)는 소정의 데이터 센터의 다른 팔로어 캐시 클러스터(106)와 개별 클라이언트 서버(104)와 다른 위치에 위치할 수 있다; 즉, 소정의 영역의 소정의 데이터 센터의 팔로어 캐시 클러스터(106)(및 클러스터가 제공하는 개별 클라이언트 서버(104))는 그 영역 내 다양한 위치에 분산될 수 있다.

[0023] 특정 실시예로, 각 데이터 센터(102)는 소정의 데이터 센터(102)의 팔로어 캐시 클러스터(106)와 소정의 데이터 센터(102)의 지속성 저장 데이터베이스(110) 사이의 정보를 통신하는 리더 캐시 클러스터(108)를 더 포함한다. 특정 실시예로, 데이터베이스(110)는 관계형 데이터베이스이다. 특정 실시예로, 리더 캐시 클러스터(108)는 데이터베이스(110)의 임의의 적절한 구현과 연동하도록 동작하는 플러그-인을 포함할 수 있다. 예컨대, 데이터베이스(110)는 동적-가변 플러그-인 아키텍처로 구현될 수 있으며, MySQL 및/또는 가령 HAYSTACK, CASSANDRA 등과 같은 임의의 적절한 관계형 데이터베이스 관리 시스템을 이용할 수 있다. 하나의 구현으로, 플러그-인은 하나 이상의 테이블 또는 플랫 파일(flat files)을 포함하는 관계형 데이터베이스용으로 적합한 쿼리 및 명령에 대한 그래프 노드 및 그래프 에지로서 캐시 층에 저장된 데이터를 변환하는 것과 같은, 다양한 변환(translation) 동작을 수행한다. 특정 실시예로, 리더 캐시 클러스터(108)는 또한 리더 캐시 클러스터(108)에 캐싱된 정보 또는 (리더 캐시 클러스터(108)에 캐싱되지 않는 경우) 데이터베이스(110)에 저장된 정보에 대하여 팔로어 캐시 클러스터(106)로부터 데이터베이스(110)로의 기록 요청 및 때때로 팔로어 캐시 클러스터(106)로부터 판독 요청을 조직화한다. 특정 실시예로, 리더 캐시 클러스터(108)는 개별 데이터 센터(102)의 팔로어 캐시 클러스터(106)에 저장된 정보의 동기화를 더 조직화한다. 즉, 특정 실시예로, 소정의 데이터 센터(102)의 리더 캐시 클러스터(108)는 데이터 센터(102)의 팔로어 캐시 클러스터(106)들 사이의 캐시 일관성(예컨대, 캐싱된 정보)을 관리하고, 팔로어 캐시 클러스터(106)와 리더 캐시 클러스터(108) 사이의 캐시 일관성을 관리하며, 데이터베이스(110) 내 리더 캐시 클러스터(108)에 캐싱된 정보를 저장하도록 구성된다. 하나의 구현으로, 리더 캐시 클러스터(108) 및 팔로어 캐시 클러스터(106)는 클라이언트 서버(104)와 데이터베이스(110) 사이의 캐싱 층으로 간주될 수 있다.

[0024] 하나의 구현으로, 캐시 층(108)은 라이트-스루/리드-스루(write-thru/read-thru) 캐시 층이며, 모든 리드 및 라이트가 캐시 층을 순회한다. 하나의 구현으로, 캐시 층은 연관 정보를 관리하며, 따라서 이런 정보에 대한 쿼리를 다룰 수 있다. 다른 쿼리가 실행용 데이터베이스(110)로 통과된다. 데이터베이스(110)는 다른 쿼리 유형을 다루기 위한 다른 캐시 층들을 자체 포함할 수 있는 일반적으로 데이터베이스 시스템을 내포한다.

[0025] 각 팔로어 캐시 클러스터(106)는 복수의 팔로어 캐시 노드(112)를 포함할 수 있으며, 각각의 팔로어 캐시 노드는 개별 컴퓨터, 컴퓨팅 시스템 또는 서버에서 실행중일 수 있다. 그러나, 상술한 바와 같이, 소정의 팔로어 캐시 클러스터(106)의 팔로어 캐시 노드들(112) 각각은 중심 위치 내에 위치할 수 있다. 마찬가지로, 각 리더 캐시 클러스터(108)는 복수의 리더 캐시 노드(114)를 포함할 수 있으며, 각각의 리더 캐시 노드는 개별 컴퓨터, 컴퓨팅 시스템 또는 서버에서 실행중일 수 있다. 예컨대, 각 데이터 센터(102)는 수십, 수백 또는 수천의 클라이언트 서버(104)를 포함할 수 있으며, 각 팔로어 캐시 클러스터(106)는 클라이언트 서버(104)의 서브세트를 제공하는 수십, 수백 또는 수천의 팔로어 캐시 노드(112)를 포함할 수 있다. 마찬가지로, 각 리더 캐시 클러스터(108)는 수십, 수백 또는 수천의 리더 캐시 노드(114)를 포함할 수 있다. 특정 실시예로, 소정의 팔로어 캐시 클러스터(106) 내 각각의 팔로어 캐시 노드(112)는 단지 특정 팔로어 캐시 클러스터(106) 내의 다른 팔로어 캐시 노드(112), 특정 팔로어 캐시 클러스터(106)에 의해 제공되는 클라이언트 서버(104) 및 리더 캐시 클러스터(108) 내의 리더 캐시 노드(114)하고만 통신할 수 있다.

[0026] 특정 실시예로, 네트워킹 시스템(100)에 의해 저장된 정보는 데이터베이스(110)뿐만 아니라 각각의 팔로어 및 리더 캐시 클러스터(106 및 108) 모두 내의 각 데이터 센터(102) 내에 저장된다. 특정 실시예로, 각 데이터베이스(110) 내에 저장된 정보는 관계형으로(예컨대, MySQL을 통해 객체 및 테이블로서) 저장되는 반면, 동일한 정보가 그래프 노드 및 노드 사이의 연관이나 연결(본 명세서에서는 그래프 에지라 함)을 포함하는 그래프의 형태로, 각각의 팔로어 및 리더 캐시 클러스터(106 및 108)에 의해 각각 저장되는 다수의 데이터 샤드로 각각의 팔로어 캐시 클러스터(106) 및 리더 캐시 클러스터(108) 내에 저장된다. 특정 실시예로, 각각의 팔로어 캐시 클러스터(106)와 리더 캐시 클러스터(108)의 데이터 샤드는 개별 캐시 클러스터 내 캐시 노드들(112 또는 114) 사이에서 버킷화(bucketized)되거나 분할될 수 있다. 즉, 개별 캐시 클러스터 내 각각의 캐시 노드들(112 또는 114)은 클러스터에 의해 저장된 한 서브세트의 샤드를 저장한다(그리고 리더 캐시 클러스터가 소정의 데이터 센터(102) 그리고 일부 실시예에서는 데이터 센터(102)들 사이의 캐시 클러스터들 각각에 의해 저장된 샤드를 동기화할 때, 각각의 팔로어 및 리더 캐시 클러스터(106 및 108)에 의해 저장된 각 세트의 샤드는 각각 동일한 정보를 저장한다).

[0027] 특정 실시예로, 각각의 팔로어 및 리더 캐시 클러스터(106 및 108)와 데이터베이스(110)에 각각 저장되는 그래프에서 그래프 노드를 고유하게 식별하는 고유 식별자(ID)(이하, 노드 ID라고 함)를 지정받는다; 즉, 즉, 각 노드 ID는 전역적으로(globally) 고유하다. 하나의 구현으로, 각 노드 ID는 64-비트 식별자이다. 하나의 구현으로, 샤드(shard)는 노드 ID 공간의 세그먼트를 할당받는다. 특정 실시예로, 각 노드 ID는 (예컨대, 산술

적으로 또는 컴(come) 수학적 함수를 통해) 고유의 해당 샤드 ID와 맵핑한다; 즉, 각 샤드 ID는 또한 전역적으로 고유하며, 각각의 팔로어 및 리더 캐시 클러스터(106 및 108)에 의해 각각 저장된 각 세트의 샤드 내의 동일한 데이터 객체를 나타낸다. 다시 말하면, 모든 데이터 객체는 고유 노드 ID(들)를 갖는 그래프 노드로 저장되며, 각각의 팔로어 및 리더 캐시 클러스터(106 및 108)의 데이터 샤드로 그래프에 각각 저장된 모든 정보는 동일한 해당 고유 샤드 ID(들)를 사용하여 각각의 팔로어 및 리더 캐시 클러스터(106 및 108)의 데이터 샤드에 각각 저장된다.

[0028] 방금 설명한 바와 같이, 특정 실시예로, 샤드 ID 공간(샤드 ID(들) 및 각 캐시 클러스터의 모든 샤드에 의해 저장되고, 다른 팔로어 캐시 클러스터(106)와 리더 캐시 클러스터(108) 모두에 복제되는 연관 정보의 모음)은 각각의 팔로어 또는 리더 캐시 클러스터(106 및 108) 내에서 각각의 팔로어 또는 리더 캐시 노드(112 및 114) 사이로 분할된다. 예컨대, 소정의 팔로어 캐시 클러스터(106)에서 각 팔로어 캐시 노드(112)는 개별 팔로어 캐시 클러스터(106)에 의해 저장된 샤드의 서브세트(예컨대, 수십, 수백 또는 수천의 샤드)를 저장할 수 있으며, 각 샤드는 개별 노드 ID들이 특정 샤드에 의해 저장된 샤드 ID들 세트(range) 중 샤드 ID들과 맵핑되는 노드에 대한 정보를 포함하는, 정보를 저장할 노드 ID들 세트(range)에 할당받는다. 마찬가지로, 리더 캐시 클러스터(108)에서 각 리더 캐시 노드(114)는 개별 리더 캐시 클러스터(108)에 의해 저장된 샤드의 서브세트(예컨대, 수십, 수백 또는 수천의 샤드)를 저장할 수 있으며, 각 샤드는 개별 노드 ID들이 특정 샤드에 의해 저장된 샤드 ID들 세트 중 샤드 ID들과 맵핑되는 노드에 대한 정보를 포함하는, 정보를 저장할 노드 ID들 세트에 할당받는다.

[0029] 그러나, 상술한 바와 같이, 소정의 샤드 ID는 팔로어 및 리더 캐시 클러스터(106 및 108)에 의해 각각 저장된 동일한 데이터 객체에 해당한다. 각 팔로어 캐시 클러스터(106) 내의 팔로어 캐시 노드(106)의 개수와 리더 캐시 클러스터(108) 내의 리더 캐시 노드(114)의 개수는 정적으로(예컨대, 팔로어 캐시 클러스터(106)와 리더 캐시 클러스터(108)는 일반적으로 서로 다른 개수의 팔로어 캐시 노드(112)와 리더 캐시 노드(114)를 각각 포함할 수 있다) 또는 동적으로(예컨대, 소정의 캐시 클러스터 내의 캐시 노드는 다양한 이유로 주기적으로 또는 수리, 업데이트 또는 유지보수가 필요할 때 셧다운(shut down)될 수 있다) 변할 수 있기 때문에, 각각의 팔로어 캐시 노드(112) 및 리더 캐시 노드(114)에 의해 저장되는 샤드의 개수는 각 캐시 클러스터 내에서뿐만 아니라 캐시 클러스터들 사이에서 정적으로 또는 동적으로 변할 수 있다. 게다가, 각 샤드에 할당된 샤드 ID(들) 세트도 또한 정적으로 또는 동적으로 변할 수 있다.

[0030] 특정 실시예로, 각각의 팔로어 캐시 노드(112) 및 리더 캐시 노드(114)는 개별 캐시 노드 내에 캐싱된 정보의 저장 및 제공을 관리하는 그래프 관리 소프트웨어를 포함한다. 특정 실시예로, 소정의 캐시 클러스터의 각각의 캐시 노드에서 실행하는 그래프 관리 소프트웨어는 어느 샤드(및 해당 샤드 ID들)가 개별 캐시 클러스터 내에 각각의 캐시 노드에 의해 저장되는지를 결정하도록 통신할 수 있다. 추가로, 캐시 노드가 팔로어 캐시 노드(112)라면, 팔로어 캐시 노드(112)에서 실행하는 그래프 관리 소프트웨어는 클라이언트 서버(104)로부터 요청(예컨대, 기록 또는 판독 요청)을 수신하고, 팔로어 캐시 노드 내의 적절한 샤드 내에서 정보를 검색하거나, 업데이트하거나, 삭제하거나, 저장함으로써 그 요청을 제공하며, 개별 팔로어 캐시 클러스터(106)의 팔로어 캐시 노드(112)와 다른 팔로어 캐시 노드(112)들 사이의 통신뿐만 아니라 팔로어 캐시 노드(112)와 리더 캐시 클러스터(108)의 리더 캐시 노드(114) 사이의 통신을 관리하거나 용이하게 한다. 마찬가지로, 캐시 노드가 리더 캐시 노드(114)라면, 리더 캐시 노드(114)에서 실행하는 그래프 관리 소프트웨어는 리더 캐시 노드(114)와 팔로어 캐시 클러스터(106)의 팔로어 캐시 노드(112)와 리더 캐시 클러스터(108)의 다른 리더 캐시 노드(114) 사이의 통신뿐만 아니라 리더 캐시 노드(114)와 데이터베이스(110) 사이의 통신을 관리한다. 각각의 캐시 노드들(112 및 114)에서 실행하는 그래프 관리 소프트웨어는 그래프의 형태로 정보를 저장하고 제공하는 것으로 이해한다.

[0031] 특정 실시예로, 각 팔로어 캐시 노드(112)에서 그래프 관리 소프트웨어는 또한 개별 팔로어 캐시 클러스터(106)의 다른 캐시 노드(112), 리더 캐시 클러스터(108)의 리더 캐시 노드(114) 및 개별 팔로어 캐시 클러스터(106)가 제공하는 클라이언트 서버(104)와 공유하는, 테이블의 관리를 담당한다. 이런 테이블은 샤드 ID와 관련된 샤드 ID 및 정보를 저장하는 소정의 팔로어 캐시 클러스터(106) 내의 특정 캐시 노드(112)와 각 샤드 ID의 맵핑을 제공한다. 이런 방식으로, 특정 팔로어 캐시 클러스터(106)에 의해 제공되는 클라이언트 서버(104)는 팔로어 캐시 클러스터(106) 내의 팔로어 캐시 노드(112) 중 어느 것이 클라이언트 서버(104)가 접근하거나 추가하거나 업데이트하려고 하는 정보와 관련된 샤드 ID를 유지하는지 인식한다(예컨대, 클라이언트 서버(104)는 팔로어 캐시 노드(112) 중 어느 것이 할당받는지를 결정하는 맵핑 테이블을 사용한 후 특정 샤드 ID와 관련된 정보를 저장하거나 저장할, 그리고 그 샤드 ID를 저장하는 특정 팔로어 캐시 노드(112)에 대한 기록 또는 판독 요청을 송신할 수 있다). 마찬가지로, 특정 실시예로, 각 리더 캐시 노드(114)에서 그래프 관리 소프트웨어도 또한

개별 리더 캐시 클러스터(108)의 다른 캐시 노드(114), 팔로어 캐시 클러스터(106)의 팔로어 캐시 노드(114) 및 리더 캐시 클러스터(108)가 관리하는 팔로어 캐시 클러스터(106)의 팔로어 캐시 노드(112)와 공유하는, 테이블의 관리를 담당한다. 게다가, 이런 방식으로, 소정의 팔로어 캐시 클러스터(106)에서 각 팔로어 캐시 노드(112)는 소정의 팔로어 캐시 클러스터(106)에서 다른 팔로어 캐시 노드(112) 중 어느 것이 개별 팔로어 캐시 클러스터(106)에 의해 저장된 어느 샤드 ID들을 저장하는지 인식한다. 마찬가지로, 이런 방식으로, 리더 캐시 클러스터(108)에서 각 리더 캐시 노드(114)는 리더 캐시 클러스터(108)에서 다른 리더 캐시 노드(114) 중 어느 것이 리더 캐시 클러스터(108)에 의해 저장된 어느 샤드 ID들을 저장하는지 인식한다. 게다가, 소정의 팔로어 캐시 클러스터(106)에서 각 팔로어 캐시 노드(112)는 리더 캐시 클러스터(108)에서 리더 캐시 노드(114) 중 어느 것이 어느 샤드 ID들을 저장하는지 인식한다. 마찬가지로, 리더 캐시 클러스터(108)에서 각 리더 캐시 노드(114)는 각각의 팔로어 캐시 클러스터(106)에서 팔로어 캐시 노드(112) 중 어느 것이 어느 샤드 ID들을 저장하는지 인식한다.

[0032] 특정 실시예로, 그래프(특정한 예시적인 실시예에서는 소셜 그래프) 내에 각 노드에 대한 정보는 그 샤드 ID를 기초로 각각의 팔로어 캐시 클러스터(106)와 리더 캐시 클러스터(108)의 개별 샤드에 저장된다. 상술한 바와 같이, 그래프에서 각 노드는 노드 ID를 갖는다. 샤드 ID를 따라, 개별 캐시 노드(112 또는 114)는 노드의 타입을 식별하는 노드 타입 파라미터뿐만 아니라 하나 이상의 이름-값 쌍(가령,텍스트, 미디어 또는 미디어나 다른 자원으로의 URL들과 같은 컨텐츠)과 메타데이터(예컨대, 노드가 생성되거나 변경되었을 때의 타임스탬프)를 저장할 수 있다. 특정 실시예로, 그래프(특정한 예시적인 실시예에서는 소셜 그래프)에서 각 에지는 에지가 연결되는 각 노드와 함께 저장된다. 예컨대, 대부분의 에지는 양방향이다; 즉, 대부분의 에지는 그래프에서 2개의 노드를 각각 연결한다. 특정 실시예로, 각 에지는 에지가 연결하는 각 노드와 동일한 샤드에 저장된다. 예컨대, 노드 ID1과 노드 ID2를 연결하는 에지는 노드 ID1에 해당하는 샤드 ID(예컨대, 샤드 ID1) 및 노드 ID2에 해당하는 샤드 ID(예컨대, 샤드 ID2)와 함께 저장될 수 있으며, 서로 다른 샤드 또는 심지어 소정의 캐시 클러스터의 서로 다른 캐시 노드에 있을 수 있다. 예컨대, 에지는 에지 타입이 에지의 타입을 나타내는 {노드 ID1, 에지 타입, 노드 ID2}의 형태로 샤드 ID1와 함께 저장될 수 있다. 또한, 에지는 메타데이터(예컨대, 에지가 생성되거나 변경되었던 시기를 나타내는 타임스탬프)를 포함할 수 있다. 또한, 에지는 (노드 ID1, 에지 타입, 노드 ID2)의 형태로 샤드 ID2와 함께 캐싱될 수 있다. 예컨대, 소셜 네트워킹 시스템(100)의 사용자가 또 다른 사용자와의 교류 관계 또는 개념이나 사용자와의 팬 관계를 확립하는 경우, "친구" 또는 "팬" 타입의 에지 관계는 2개의 샤드에 저장될 수 있는데, 제 1 샤드는 사용자의 식별자가 맵핑되는 샤드에 해당하며, 제 2 샤드는 다른 사용자 또는 개념의 객체 식별자가 맵핑되는 샤드에 해당한다.

[0033] 네트워킹 시스템(100) 및 특히 팔로어 캐시 클러스터(106)의 팔로어 캐시 노드(112)와 리더 캐시 클러스터(108)의 리더 캐시 노드(114)에서 실행하는 그래프 관리 소프트웨어는 클라이언트 서버(104)로부터뿐만 아니라 다른 팔로어나 리더 캐시 노드들(112 및 114) 각각으로 또는 각각으로부터 수신된 다수의 쿼리를 지원한다. 예컨대, 쿼리 '객체(object)_추가(add){ID1, 노드 타입1, 메타데이터(항상 명시되는 것은 아님), 페이로드(payload)(항상 명시되는 것은 아님)}'를 통해 수신중인 캐시 노드가 노드 ID1가 해당하는 샤드에서 명시된 노드 타입1의 쿼리에 명시된 노드 ID1과 함께 새로운 노드를 저장한다. 또한, 수신중인 캐시 노드는, 명시된 경우, 노드 ID1과 함께 메타데이터(예컨대, 타임스탬프)와 페이로드(예컨대, 이름-값 쌍 및/또는 가령 텍스트, 미디어, 리소스 또는 리소스로의 참조와 같은 컨텐츠)를 저장한다. 또 다른 예로써, 쿼리 '객체_업데이트(update){ID1, 노드 타입1(항상 명시되는 것은 아님), 메타데이터(항상 명시되는 것은 아님), 페이로드(항상 명시되는 것은 아님)}'를 통해 수신중인 캐시 노드는 해당 샤드에서 쿼리에 명시된 노드 ID1에 의해 식별된 노드를 업데이트한다(예컨대, 노드 타입을 쿼리에 명시된 노드 타입1로 변경하거나, 쿼리에 명시된 메타데이터로 메타데이터를 업데이트하거나, 쿼리에 명시된 페이로드와 함께 저장된 컨텐츠를 업데이트한다). 또 다른 예로써, 쿼리 '객체_삭제(delete){노드 ID1}'를 통해 수신중인 캐시 노드는 쿼리에 명시된 노드 ID1에 의해 식별된 노드를 삭제한다. 또 다른 예로써, 쿼리 '객체_겟(get){노드 ID1}'를 통해 수신중인 캐시 노드는 쿼리에 명시된 노드 ID1에 의해 식별된 노드와 함께 저장된 컨텐츠를 검색한다.

[0034] 이제 에지 쿼리를 참조하면, 쿼리 'assoc_add{ID1, 에지 타입1, ID2, 메타데이터(항상 명시되는 것은 아님)}'를 통해 (노드 ID1을 저장하는) 수신중인 캐시 노드는 노드 ID1에 의해 식별된 노드와 에지 타입1의 노드 ID2에 의해 식별된 노드 사이의 에지를 생성하고, 명시된 경우, 메타데이터(예컨대, 에지가 요청되었던 시간을 나타내는 타임스탬프)를 따라 노드 ID1에 의해 식별된 노드와 함께 에지를 저장한다. 또 다른 예로써, 쿼리 'assoc_update{노드 ID1, 에지 타입1, 노드 ID2, 메타데이터(항상 명시되는 것은 아님)}'를 통해 (노드 ID1을 저장하는) 수신중인 캐시 노드는 노드 ID1에 의해 식별된 노드와 노드 ID2에 의해 식별된 노드 사이의 에지를 업데이트한다. 또 다른 예로써, 쿼리 'assoc_delete{노드 ID1, 에지 타입1(항상 명시되는 것은 아님), 노드

ID2}'를 통해 (노드 ID1를 저장하는) 수신중인 캐시 노드는 노드 ID1에 의해 식별된 노드와 노드 ID2에 의해 식별된 노드 사이의 에지를 삭제한다. 또 다른 예로써, 쿼리 'assoc_get{노드 ID1, 에지 타입1, 정렬 키(sortkey)(항상 명시되는 것은 아님), 시작(start)(항상 명시되는 것은 아님), 한도(limit)(항상 명시되는 것은 아님)}'를 통해 (노드 ID1을 저장하는) 수신중인 캐시 노드는 에지 타입1의 에지들로 노드 ID1에 의해 식별된 노드와 연결된 노드들 중 노드 ID들을 반환한다. 추가로, 명시된다면, 정렬키는 필터를 명시한다. 예컨대, 정렬 키가 타임스탬프를 명시한다면, (노드 ID1을 저장하는) 수신중인 캐시 노드는 시작 파라미터에 의해 명시된 시간값과 한도 파라미터에 의해 명시된 시간값 사이에 형성되는, 에지 타입1의 에지들로 노드 ID1에 의해 식별된 노드와 연결된 노드들 중 노드 ID들을 반환한다. 또 다른 예로써, 쿼리 'assoc_exists{노드 ID1, 에지 타입1, 다른 노드 ID들의 리스트, 정렬키(항상 명시되는 것은 아님), 시작(항상 명시되는 것은 아님), 한도(항상 명시되는 것은 아님)}'를 통해 (노드 ID1을 저장하는) 수신중인 캐시 노드는 에지 타입1의 에지들로 샤드 ID1에 의해 식별된 노드와 연결된 다른 노드 ID들의 리스트에 명시된 노드의 노드 ID들을 반환한다. 또한, 상술한 쿼리들은 상술한 형태로 송신될 수 있으며 리더 캐시 노드(114)를 업데이트하는데 사용될 수 있다.

[0035] 하나의 구현으로, 팔로어 및 리더 캐시 클러스터(108 및 106)에 의해 구현된 캐싱 층은 하나 이상의 쿼리 타입에 대한 높은 쿼리 레이트(query rates)를 지원하는 방식으로 하나 이상의 인덱스에 연관 데이터를 캐싱하고 관리한다. 일부 실시예로, 본 발명은 효율적인 교차(intersection), 멤버십 및 그래프에서 노드 사이의 연관에 관한 쿼리의 필터링을 용이하게 한다. 예컨대, 하나의 구현으로, 캐싱 층은 포인트 룩업(point lookup)을 다루고, 노드들 사이의 다양한 연관에 대한 쿼리를 정렬(range)하고 카운트하는데 최적화된 방식으로 정보를 캐싱한다. 예컨대, 페이지를 구성함에 있어서, 클라이언트 서버(104)는 소정의 사용자의 모든 친구들에 대한 쿼리를 발행(issue)할 수 있다. 클라이언트 서버(104)는 사용자와 "친구" 에지 타입을 식별하는 'assoc_get' 쿼리를 발행할 수 있다. 쿼리의 처리를 용이하게 하기 위해, 캐싱 층에서 캐시 노드는 제 1 노드(예컨대, 사용자에 해당하는 노드)와 사용자의 연줄(contacts) 또는 친구에 해당하는 노드 사이의 소정의 타입(가령, "친구", "팬", "회원", "좋아요" 등)의 연관을 저장할 수 있다. 또한, 페이지의 또 다른 파트를 구성하기 위해, 클라이언트 서버(104)는 사용자나 사용자 프로파일, "월포스트" 에지 타입 및 한도값을 식별하는 'assoc_get' 쿼리를 발행함으로써, 프로파일 상에 마지막 N 세트의 월 포스트(wall post)의 쿼리를 발행할 수 있다. 마찬가지로, 특정 월 포스트에 대한 코멘트는 유사한 방식으로 검색될 수 있다.

[0036] 하나의 구현으로, 팔로어 캐시 클러스터(106)와 리더 캐시 클러스터에 의해 구현된 캐싱 층은 신속한 검색을 용이하게 하고 높은 쿼리 레이트를 처리하는 그래프 내의 노드들(id1, id2) 사이의 연관에 대한 한 세트의 인-메모리(in-memory) 구조를 관리한다. 예컨대, 각 (id1, 타입) 연관 세트(id1에서 비롯되고 소정의 타입을 갖는 한 세트의 모든 연관)에 대해, 캐싱 층은 2개의 인-메모리 인덱스를 관리한다. 상술한 바와 같이, 이들 연관 세트는 id1이 속한 샤드를 기초로 각 클러스터의 캐시 노드에 의해 관리된다. 더욱이, 하기에 기술되는 구조를 고려할 때, 2개의 노드들 사이의 소정의 연관은 연관의 개별 노드들 각각에 대한 2개의 연관 세트들에 저장될 수 있다. 제 1 인덱스는 시간의 속성(temporal attribute)(예컨대, 타임 스탬프)에 기반하며 범위 쿼리(range queries)를 지원한다. id2의 제 2 인덱스는 범위 쿼리를 지원하지 않지만, 인서트(inserts)와 룩업(look ups)의 더 나은 시간 복잡도(complexity)를 지원한다. 하나의 구현으로, 제 1 인덱스는 순환 버퍼에 저장된 연관 엔트리의 순서화된 동적 어레이이다. 순환 버퍼에서 각 엔트리는 하나의 연관을 표현하거나 하나의 연관에 해당하며, 다음의 필드를 포함한다: a) $flags (1 바이트)(연관의 가시성을 나타냄); b) $id2 (8 바이트); c) $time (4 바이트); d) $data (8 바이트)($data는 고정된 크기의 8 바이트 필드이다($data에 대해 8 바이트 이상이 요구되는 경우, 이것은 또 다른 메모리 청크(chunk)에 대한 포인터(pointer)가 되어 풀 $data 값을 유지한다; $data는 소정의 assoc 타입에 대해 선택적이다)); 및 e) $link (8 바이트) 동일한 id2 인덱스 버킷(bucket)에서 다음 및 이전 엔트리의 오프셋(아래 참조). 하나의 구현으로, 어레이는 $time 속성의 오름차순으로 순서화된다. 인덱스에서 엔트리의 수는 (가령 10,000으로) 한도가 정해지며 연관 타입으로 설정가능하다. 한도에 도달하는 경우, 어레이는 랩 어라운드(wrap around)된다. 어레이는 $time 정렬되기 때문에, 대부분의 새로운 엔트리는 임의의 기존의 요소를 이동하지 않고 말단에 첨부될 것이다.

[0037] 하나의 구현으로, 기본(primary) 인덱스는 전역 멤캐시드 해쉬 테이블(global memcached hash table)을 통해 이름("assoc:<id1>:<type>")으로 룩업될 수 있는 단일 멤캐시드 키에 저장될 수 있다. 어레이는 다음의 필드를 포함하는 헤더로 진행될 수 있다: a) count (4 바이트): (지속적으로 저장되나, 인덱스에서 단지 캐싱된 엔트리는 아닌) (id1, type) 연관 세트에서 가시적 연관의 카운트(count); b) head (4 바이트): 순환 버퍼에서 (가장 높게 정렬된 요소인) 어레이 헤드의 바이트 오프셋; c) tail (4 바이트): 순환 버퍼에서 (가장 낮게 정렬된 요소인) 어레이 테일(tail)의 바이트 오프셋; 및 d) id2 인덱스 포인터 (8 바이트): id2 해쉬 테이블을 포함하는

블록으로의 포인터.

[0038] 일실시예로, 제 2 ($id2) 인덱스는 해쉬 테이블로서 구현되며, 소정의 ($id1, $type, $id2) 연관에 대해 신속한 인서트 및 룩업을 지원한다. 하나의 구현으로, 해쉬 테이블 그 자체는 멤캐시드 메모리 할당기로 할당된 개별 블록에 저장될 수 있다. 테이블은 기본 인덱스로의 오프셋 어레이이며, 각각은 해당 해쉬 버킷에서 제 1 요소를 식별한다. 요소들은 $link 필드를 통해 버킷으로 연결된다. 개별 블록에서 해쉬 테이블을 저장함으로써 구현자는 테이블과 기본 인덱스를 별도로 재조정할 수 있으며, 따라서 연관 세트가 커지면서 복사된 메모리의 양은 감소한다. 또한, 연관 엔트리와 버킷을 제자리에 연결하는 것은 메모리 효율을 향상시킨다. 마크되거나, 숨겨지거나, 삭제된 엔트리가 인덱스로부터 지워지는 경우, 해쉬 테이블(및 버킷 리스트)은 재구성될 필요가 있으나, 이는 드물게 일어날 수 있다.

[0039] 따라서, 동일한 <타입>의 새로운 연관이 추가될 때, 캐시 노드(112, 114)는 새로 관련된 객체를 해쉬 테이블 및 순환 버퍼로 추가하며, 순환 버퍼에서 가장 오래된 엔트리를 제거한다. 상술한 바와 같이, <정렬키> 값은 가령 타임 스탬프와 같은 속성을 기초로 매칭 엔트리를 정렬하는데 사용될 수 있다. 또한, <한도> 값은 반환된 결과의 수를 제 1의 N 값으로 제한하며, 여기서 N=<한도>이다. 이런 구성은 매우 높은 쿼리 레이트에서 노드들 사이의 연관에 관한 쿼리를 제공할 수 있도록 해준다. 예컨대, 제 1 쿼리는 웹 페이지의 한 부분에서 한 세트의 친구를 디스플레이하도록 요청할 수 있다. 캐시 노드는 기본 인덱스에 접근하고 순환 버퍼에서 제 1의 N(여기서 N=한도) 개의 id2 엔트리를 검색하여, id1에 해당하는 연관 세트를 룩업함으로써, 'get_assoc(id1, 타입, 정렬키, 한도)' 쿼리에 신속히 대응할 수 있다. 또한, 보조 인덱스의 해쉬 테이블은 포인트 룩업을 용이하게 한다. 더욱이, 캐싱 층에 의해 관리되는 카운트 값은 소정의 연관 세트(id1, 타입)의 카운트에 대한 신속한 응답을 용이하게 한다.

[0040] 이제, 데이터의 저장 및 제공에 대한 몇몇의 일반적인 예가 기술될 것이다(소셜 그래프의 특정 예시적인 구현이 기술된 후에 소셜 그래프의 특정 예시적인 구현에 관한 더 구체적인 예들이 하기에 기술될 것이다). 예컨대, 클라이언트 서버(104)가 (예컨대, 사용자 요청에 응답하여) 가령 네트워킹 시스템(100)의 사용자로부터 또는 또 다른 서버나 컴포넌트나, 애플리케이션이나 네트워킹 시스템(100)의 프로세스로부터 웹 페이지에 대한 요청을 수신하는 경우, 클라이언트 서버(104)는 요청된 웹 페이지를 생성하기 위해 하나 이상의 쿼리를 발행해야 한다. 또한, 사용자가 네트워킹 시스템(100)과 상호작용할 때, 클라이언트 서버(104)는 객체 노드 및/또는 연관 객체 노드를 확립하거나 변경하는 요청을 수신할 수 있다. 일부 예에서, 클라이언트 서버(104)에 의해 수신된 요청은 일반적으로 클라이언트 서버(104)에 대한 요청을 사용자에 대신하여 이루어지는 사용자를 나타내는 노드 ID를 포함한다. 또한, 요청은 대안으로 사용자가 열람하거나 업데이트하거나 삭제하거나 (에지와) 연결 또는 연관시키고자 하는 객체에 해당하는 하나 이상의 다른 노드 ID들을 포함할 수 있다.

[0041] 예컨대, 요청은 사용자가 열람하고자 하는 객체 또는 객체들(예컨대, 웹 페이지를 제공하기 위한 하나 이상의 객체)과 관련된 정보에 접근하기 위한 판독 요청일 수 있다. 예컨대, 판독 요청은 특정 노드용으로 저장된 컨텐츠에 대한 요청일 수 있다. 예컨대, 사용자 프로파일에서 월 포스트는 "월포스트"의 에지 타입을 갖는 노드로 표현될 수 있다. 또한, 월포스트에 대한 코멘트는 월포스트에 대한 에지 타입 "코멘트" 연관을 갖는 그래프 내의 노드들로 표현될 수 있다. 이런 예에서, 특정 실시예로, 클라이언트 서버(104)는 컨텐츠나 다른 요청된 정보를 포함하고, (클라이언트 서버(104)를 제공하는 팔로어 캐시 클러스터(106)에서) 팔로어 캐시 노드(112) 중 어느 것이 샤드 ID를 저장하는지 결정하도록 맵핑 테이블을 사용하며, 샤드 ID와 관련되고 샤드 ID와 함께 저장되는 정보를 저장하는 팔로어 캐시 노드(112) 중 특정한 하나에 대한 샤드 ID를 포함하는 쿼리를 전송하는, 객체(노드)의 노드 ID에 해당하는 샤드 ID를 결정한다. 이후, 특정 캐시 노드(112)는 (해당 샤드 내에서 캐싱된다면) 요청된 정보를 검색하고 요청중인 클라이언트 서버(104)에 대한 정보를 전송한 후, (예컨대, 사용자의 컴퓨팅 장치에서 실행하는 웹 브라우저나 다른 문서-랜더링(document-rendering) 애플리케이션에 의해 렌더링 가능한 HTML 또는 다른 구조화 문서의 형태로) 요청중인 사용자에 대한 정보를 제공할 수 있다. 요청된 정보가 팔로어 캐시 노드(112) 내에서 저장/캐싱되지 않는다면, 이후 팔로어 캐시 노드(112)는 맵핑 테이블을 사용하여 리더 캐시 노드(114) 중 어느 것이 샤드 ID를 저장하는 샤드를 저장하는지를 그리고 샤드 ID를 저장하는 특정 리더 캐시 노드(114)로 쿼리를 전달하는지를 결정할 수 있다. 요청된 정보가 특정 리더 캐시 노드(114) 내에서 캐싱된다면, 이후 리더 캐시 노드(114)는 요청된 정보를 검색하고 팔로어 캐시 노드(112)로 전달할 수 있으며, 이후 팔로어 캐시 노드(112)에서 특정 샤드를 업데이트하여 샤드 ID와 함께 요청된 정보를 저장하고, 방금 기술한 바와 같이 클라이언트 서버(104)로 쿼리를 제공하도록 진행한 후, 요청중인 사용자에게 그 정보를 제공할 수 있다. 요청된 정보가 리더 캐시 노드(114) 내에서 캐싱되지 않는다면, 이후 리더 캐시 노드(114)는 쿼리를 데이터베이스(110)의 언어로 변환하고 새로운 쿼리를 데이터베이스(110)로 전송할 수 있으며, 이후 요청된

정보를 검색하고 요청된 정보를 특정 리더 캐시 노드(114)로 전송한다. 이후, 리더 캐시 노드(114)는 검색된 정보를 그래프 관리 소프트웨어에 의해 이해되는 그래픽 언어로 재변환되고, 리더 캐시 노드(114)에서 특정 샤드를 업데이트하여 샤드 ID와 함께 요청된 정보를 저장하며, 특정 팔로어 캐시 노드(112)로 검색된 정보를 전송한 후, 팔로어 캐시 노드(112)에서 특정 샤드를 업데이트하여 샤드 ID와 함께 요청된 정보를 저장하고 방금 기술한 바와 같이 클라이언트 서버(104)로 쿼리를 제공하도록 진행한 후, 요청중인 사용자에게 그 정보를 제공할 수 있다.

[0042] 또 다른 예로써, 사용자 요청은 기존 정보를 업데이트하거나 노드에 대한 추가 정보를 저장하거나, 2개의 노드들 사이의 에지를 생성하거나 변경하는 기록 요청일 수 있다. 상술한 경우에, 저장되는 정보가 존재하지 않은 노드에 대한 것이라면, 사용자 요청을 수신하는 클라이언트 서버(104)는 새로운 노드의 노드 ID에 대한 요청을 클라이언트 서버(104)를 제공하는 개별 팔로어 캐시 클러스터(106)로 전송한다. 일부의 경우 또는 일부 실시예에서, (예컨대, 새로운 노드를 또 다른 노드와 동일한 장소에 배치하기 위해) 클라이언트 서버(104)는 새로운 노드가 저장되는 특정 샤드를 지정할 수 있다. 이런 경우, 클라이언트 서버(104)는 지정된 샤드를 저장하는 특정 팔로어 캐시 노드(112)로부터 새로운 노드 ID를 요청한다. 대안으로, 클라이언트 서버(104)는 통과된 노드 ID를 저장한 샤드를 저장하는 팔로어 캐시 노드(112)로 새로운 노드 ID에 대한 요청과 함께 기존 노드의 노드 ID를 통과시켜, 팔로어 캐시 노드(112)가 샤드에 저장된 노드 ID들 세트에 있는 새로운 노드에 대한 노드 ID로 클라이언트 서버(104)에 응답하도록 한다. 다른 경우 또는 다른 실시예에서, 클라이언트 서버(104)는 (예컨대, 임의로 또는 몇몇 기능을 기초로) 특정 팔로어 캐시 노드(112) 또는 새로운 노드 ID 요청을 송신하는 특정 샤드를 선택할 수 있다. 이후, 어떤 경우에서든지, 특정 캐시 노드(112) 또는 특히 팔로어 캐시 노드(112)를 실행하는 그래프 관리 소프트웨어는 새로운 노드 ID를 클라이언트 서버(104)로 전송한다. 이후, 클라이언트 서버(104)는 해당 팔로어 캐시 노드(112)에 대한 새로운 노드 ID를 포함하는 기록 요청을 만들어낼 수 있다. 또한, 기록 요청은 새로운 노드의 노드 타입을 지정할 수 있으며, 노드 ID와 함께 저장되는 페이로드(예컨대, 새로운 노드와 함께 저장되는 컨텐츠) 및/또는 메타데이터(예컨대, 다른 데이터 중에서, 요청을 한 사용자의 노드 ID, 클라이언트 서버(104)에 의해 요청이 수신되었던 시간을 나타내는 타임스탬프)를 포함할 수 있다. 예컨대, 팔로어 캐시 노드(112)로 송신된 기록 요청은 'object_add{노드 ID, 노드 타입, 페이로드, 메타데이터}'의 형태일 수 있다. 마찬가지로, 노드를 업데이트하기 위해, 클라이언트 서버(104)는 노드 ID가 저장된 샤드를 저장하는 팔로어 캐시 노드(112)로 'object_modify(변경){노드 ID, 노드 타입, 페이로드, 메타데이터}' 형태의 기록 요청을 송신할 수 있다. 마찬가지로, 노드를 삭제하기 위해, 클라이언트 서버(104)는 샤드 ID가 저장된 샤드를 저장하는 팔로어 캐시 노드(112)로 'object_delete{노드 ID}' 형태의 요청을 송신할 수 있다.

[0043] 그 다음, 특정 실시예로, 팔로어 캐시 노드는 리더 캐시 노드(114)가 이후 샤드를 업데이트할 수 있도록, 해당 노드 ID를 저장한 샤드를 저장하는 리더 캐시 노드(114)로 요청을 전송한다. 이후, 리더 캐시 노드(114)는 요청을 데이터베이스(110)의 언어로 변환하고, 데이터베이스가 이후 업데이트될 수 있도록 변환된 요청을 데이터베이스(110)로 전송한다.

[0044] 도 4는 2개의 노드 사이의 연관을 추가하는 요청(assoc_add)을 처리하기 위한 예시적인 방법을 도시한다. 도 4에 도시된 바와 같이, 팔로어 캐시 노드(112)가 assoc_add 요청(예컨대, assoc_add(id1, 타입, id2, 메타데이터))을 수신하는 경우, 팔로어 캐시 노드는 인덱스에 접근하여 id1 및 타입에 해당하는 연관 세트 객체를 식별한다(402). 팔로어 캐시 노드(112)는 id2를 해쉬 테이블과 연관 세트 객체의 순환 버퍼에 추가하고 연관 세트 객체의 카운트 값을 증가시킨다(404). 연관 세트 객체는 이제 노드 id1과 노드 id2 사이의 소정의 타입의 새로운 연관을 관리한다. id2에 대하여 연관의 검색을 용이하게 하기 위해, 팔로어 캐시 노드(112)는 노드 식별자(id2)에 해당하는 샤드 ID를 식별하고, 식별된 샤드(406)를 처리하는 클러스터의 팔로어 캐시 노드(112)로 assoc_add 요청을 전달한다. 인스턴트(instant) 팔로어 캐시 노드(112)가 샤드를 다룬다면, 이 노드가 assoc_add 요청을 처리한다. 하나의 구현으로, 전달(forwarding) 팔로어 캐시 노드(112)는 '이것이 캐시 층에서 양방향 연관을 확립하는데 필요한 업데이트이다'라는 신호의 변경된 assoc_add 요청을 전송할 수 있다. 또한, 팔로어 캐시 노드(112)는 id1이 속한 샤드에 해당하는 리더 캐시 노드(114)로 assoc_add 요청을 전달한다(408). 리더 캐시 노드(114)는 리더 캐시 클러스터에서 양방향 연관을 확립하는 것과 유사한 프로세스를 실행할 수 있다. 또한, 리더 캐시 노드(114)는 새로운 연관이 데이터베이스(110)에서 지속되도록 해준다. 이런 방식으로, 노드 id1와 노드 id2 사이의 연관은 id1 및 타입, 그리고 별도로 id2 및 타입을 참조로 인덱스에서 이제 검색될 수 있다.

[0045] 특정 실시예로, 그래프는 가령 사용자, 페이지, 이벤트, 월 포스트(wall post), 코멘트, 사진, 비디오, 배경 정보, 개념, 관심사 및 노드로 표현하기에 유용한 임의의 다른 요소와 같은 다양한 다른 노드 유형을 관리할 수

있다. 에지 유형은 노드 사이의 연관에 해당하며, 친구, 팔로어(followers), 구독자, 팬, 좋아요(또는 관심의 다른 표시), 월 포스트, 코멘트, 링크, 제안, 추천 및 노드들 사이의 연관의 다른 유형을 포함할 수 있다. 하나의 구현으로, 그래프의 일부는 소셜 네트워크 환경의 개별 사용자에 각각 해당하는 사용자 노드를 포함하는 소셜 그래프일 수 있다. 또한, 소셜 그래프는 가령 특정 개념을 각각 다루거나 특정 개념에 관한 개념 노드뿐만 아니라 소셜 네트워크 환경의 사용자들 사이에서 일시적이거나 아닐 수 있는 현재 관심사의 특정 토픽을 각각 다루거나 특정 토픽에 관한 토픽 노드와 같은 다른 노드를 포함할 수 있다. 특정 실시예로, 각 노드는 소셜 네트워크 환경에서 호스팅되거나 접근가능한 해당 웹 페이지("프로파일 페이지")를 가지거나, 해당 웹 페이지를 나타내거나 해당 웹 페이지에 의해 표시된다. 예로써, 사용자 노드는 해당 사용자가 컨텐츠를 추가하고 진술(declarations)하며 그밖에 그 또는 그녀 자신의 의사를 표현할 수 있는 해당 사용자 프로파일 페이지를 가질 수 있다. 예로써, 하기에 기술되는 바와 같이, 예컨대 사용자 프로파일 페이지, 개념 프로파일 페이지 또는 토픽 프로파일 페이지와 같은 소셜 네트워크 환경에서 호스팅되거나 접근가능한 다양한 웹 페이지는 사용자들이 컨텐츠를 포스트하거나, 상태 업데이트를 포스트하거나, 메시지를 포스트하거나, 사용자나 다른 사용자들에 의해 제시된 다른 포스트에 대한 코멘트를 포함하는 코멘트를 포스트하거나, 관심사를 진술하거나, 임의의 상술한 포스트뿐만 아니라 페이지 및 특정 컨텐츠에 대해 (하기에 기술되는) "좋아요"를 진술하거나, 그밖에 그들 자신을 표현하거나 다양한 행위를 수행할 수 있도록 해준다 (이하, 이들 사용자 행위 및 다른 사용자 행위는 일괄하여 "포스트(posts)" 또는 "사용자 행위"라고 할 수 있다). 일부 실시예로, 포스팅은 개별 프로파일 페이지, 다른 사용자 프로파일 페이지, 개념 프로파일 페이지, 토픽 페이지 또는 다른 웹 페이지나 웹 애플리케이션을 통해, 가령 미디어 컨텐츠(예컨대 사진, 비디오, 음악, 텍스트 등), URLs(uniform resource locators) 및 다른 노드와 같은 추가의 컨텐츠를 연결(linking)하거나 참조(referencing)하는 것을 포함할 수 있다. 이후, 이런 포스트, 진술 또는 행위는 인증한 사용자뿐만 아니라 다른 사용자들에 의해 열람될 수 있다. 특정 실시예로, 소셜 그래프는 소셜 그래프에서 해당 노드 쌍 사이의 연결을 각각 정의하거나 나타내는 복수의 에지를 더 포함한다. 상술한 바와 같이, 컨텐츠의 각 아이템은 다른 노드와 연결된 그래프 내의 노드일 수 있다.

[0046] 방금 설명한 바와 같이, 다양한 예시적인 실시예에서, 하나 이상의 설명된 웹 페이지 또는 웹 애플리케이션은 소셜 네트워크 환경 또는 소셜 네트워킹 서비스와 관련된다. 본 명세서에서 사용되는 바와 같이, "사용자"는 개인(사람인 사용자), 엔티티(예컨대, 기업, 비즈니스 또는 제 3 자 애플리케이션) 또는 이런 소셜 네트워크 환경과 또는 환경에서 상호작용하거나 통신하는 (예컨대, 개인 또는 엔티티의) 그룹일 수 있다. 본 명세서에서 사용되는 바와 같이, "등록 사용자"는 소셜 네트워크 환경 내 공식적으로 등록된 사용자를 말한다(일반적으로, 본 명세서에 기술되는 사용자들 및 사용자 노드들은 오직 등록된 사용자들을 말하지만, 이는 다른 실시예에서 반드시 필요조건인 것은 아니다; 즉, 다른 실시예로, 본 명세서에 기술되는 사용자들 및 사용자 노드들은 본 명세서에 기술되는 소셜 네트워크 환경에 등록되지 않은 사용자들을 말할 수 있다). 특정 실시예로, 각 사용자는 소셜 네트워크 환경에 의해 저장되거나 호스팅되거나 접근가능하며, 다른 사용자들의 전부 또는 선택된 부분집합에 의해 열람가능한 해당 "프로파일" 페이지를 갖는다. 일반적으로, 사용자는 그들 자신의 개별 프로파일 페이지의 전부나 일부에 대한 관리 권한뿐만 아니라, 잠재적으로 예컨대 홈 페이지, 웹 애플리케이션 호스팅 페이지 및 다른 가능성을 포함하는 특정 사용자에 의해 또는 특정 사용자용 다른 페이지에 대한 관리 권한을 갖는다. 본 명세서에서 사용되는 바와 같이, "인증 사용자"는 사용자가 관리 권한을 가지는 해당 프로파일 페이지에서 주장되는 사용자로서 소셜 네트워크 환경에 의해 인증된 사용자, 또는 대안으로 상기 주장되는 사용자의 적합하게 신임된 대리인을 말한다.

[0047] 2명의 사용자 또는 2개의 개념 사이의 연결은 소셜 네트워크 환경의 사용자 또는 개념 사이의 정의된 관계를 나타낼 수 있으며, 사용자, 개념, 이벤트 또는 관련을 맺은 소셜 네트워크 환경의 다른 노드에 해당하는 노드들 사이의 에지로써 소셜 네트워크 환경의 적절한 데이터 구조에서 논리적으로 정의될 수 있다. 본 명세서에서 사용되는 바와 같이, "친구관계(friendship)"는 가령 정의된 소셜 관계와 같이 소셜 네트워크 환경의 한 쌍의 사용자들 사이의 연관을 말한다. 본 명세서에서 사용되는 바와 같이, "친구"는 또 다른 사용자가 연결, 친구관계, 연관 또는 관계를 형성하여 2명의 사용자 사이에 에지가 생성되도록 하는 소셜 네트워크 환경의 임의의 사용자를 말한다. 예로써, 2명의 등록 사용자는 예컨대 다른 사용자에 대한 친구관계 요청을 전송하거나 전송되도록 한 후 그 사용자가 그 요청을 받아들이거나 거절할 수 있도록 하여 친구관계를 위해 2명의 사용자 중 하나가 다른 한 명을 선택함으로써, 명시적으로 서로 친구가 될 수 있다. 대안으로, 친구관계 또는 다른 연결은 자동으로 확립될 수 있다. 이런 소셜 친구관계는 다른 사용자들이, 특히 등록 사용자 중 하나 또는 모두와 친구인 사람들이 알아볼 수 있다. 또한, 등록 사용자의 친구는 등록 사용자의 프로파일 또는 다른 페이지에 있는 컨텐츠, 특히 사용자-생성 또는 사용자-선언 컨텐츠에 대한 접근 특권을 증가시킨다. 그러나, 소셜 그래프에서 사용자 간에 확립된 친구 연결을 갖는 2명의 사용자가 반드시 실생활(소셜 네트워킹 환경 외부)에서 (종래의 의미의) 친

구이어야 하는 것은 아님을 유의해야 한다. 예컨대, 일부 구현에서, 사용자는 비즈니스 또는 다른 비-인격 엔티티일 수 있으며, 따라서 단어의 통상의 의미로 사람인 사용자와 친구가 될 수는 없다.

[0048] 본 명세서에서 사용되는 바와 같이, "팬(fan)"은 특정 사용자, 웹 페이지, 웹 애플리케이션 또는 소셜 네트워크 환경에서 접근가능한 다른 웹 컨텐츠의 지지자 또는 팔로어인 사용자를 말한다. 특정 실시예로, 사용자가 특정 웹 페이지의 팬(특정 웹 페이지의 "팬들")인 경우, 사용자는 다른 등록 사용자에 대한 팬으로서 또는 열람하는 일반 공중으로서 그 페이지에 리스트될 수 있다. 추가로, 사용자의 아바타 또는 프로파일 사진이 그 페이지에 (또는 하기에 기술되는 임의의 페이지에서/페이지 상에서) 나타날 수 있다. 본 명세서에서 사용되는 바와 같이, "좋아요(like)"는 즉, 사용자, 특히 등록 또는 인증 사용자가 그 또는 그녀가 좋아하거나, 지지하거나 즐기는 팬이거나, 긍정적인 견해를 갖는다고 선언하거나 명시한, 가령, 예로써 제한 없이, 포스트, 코멘트, 관심, 링크, 하나의 미디어(예컨대, 사진, 사진 앨범, 비디오, 노래 등), 개념, 엔티티 또는 페이지, 및 다른 가능성과 같은 임의의 무엇을 말한다(일부 구현으로 사용자는 소셜 네트워크 시스템 또는 환경에 의해 호스팅되거나 접근가능한 임의의 페이지 상의 임의의 무엇에게 또는 무엇에 대해 실제로 좋아요를 표시하거나 선언할 수 있다). 일실시예로, "좋아요"를 표시 또는 선언하거나 사용자가 어떤 것의 "팬"이라고 표시 또는 선언하는 것은 소셜 네트워킹 환경에서 동등하게 처리되거나 정의될 수 있으며, 교환하여 사용될 수 있다; 마찬가지로, 자신을 가령 개념 또는 개념 프로파일 페이지와 같은 어떤 것의 "팬"이라고 선언하거나, 자신이 그것을 "좋아한다"고 선언하는 것은 소셜 네트워킹 환경에서 동등하게 정의될 수 있으며 본 명세서에서 교환하여 사용될 수 있다. 추가로, 본 명세서에서 사용되는 바와 같이, "관심사(interest)"는 가령 사용자의 프로파일 페이지에 제시되는 사용자-선언 관심사와 같은 사용자-선언 관심사를 말한다. 본 명세서에서 사용되는 바와 같이, "바램(want)"은 사용자가 원하는 실제의 무엇을 말한다. 상술한 바와 같이, "개념(concept)"은 사용자가 예로써, 스포츠, 스포츠 팀, 음악 장르, 음악 작곡가, 취미, 비즈니스(기업), 엔티티, 그룹, 유명인사, 등록 사용자가 아닌 사람, 또는 심지어 이벤트, 일부 실시예에서는 또 다른 사용자(예컨대, 비-인증된 사용자) 등과 같은 것에 대한 관심, 그것에 대한 좋아요 또는 그것과의 관계를 선언하거나 명시할 수 있는 실제의 무엇을 말한다. 예로써, 하나 이상의 복수의 사용자들(예컨대, Jerry Rice 외의 사용자들)에 의해 생성되고 운영되는, 유명한 프로 풋볼 선수인 "Jerry Rice"에 대한 개념 노드 및 개념 프로파일 페이지가 있을 수 있는 한편, 소셜 그래프는 Jerry Rice 그 자신(또는 Jerry Rice의 신임되거나 위임받은 대리인)에 의해 생성되고 운영되는 Jerry Rice에 대한 사용자 노드 및 사용자 프로파일 페이지를 추가로 포함한다.

[0049] 도 5는 분산형 이중화 시스템을 도시한다. 도시된 구현에서, 분산형 이중화 시스템은 적어도 제 1 및 제 2 데이터 센터(102a, 102b)를 포함한다. 각각의 데이터 센터들(102a, 102b)은 하나 이상의 팔로어 캐시 클러스터(106)와 리더 캐시 클러스터(108a, 108b)를 포함한다. 하나의 구현으로, 리더 캐시 클러스터(108a)는 기본(마스터) 캐시 클러스터의 역할을 하는 한편, 리더 캐시 클러스터(108b)는 보조(슬레이브) 캐시 클러스터이다. 하나의 구현으로, 데이터 센터들(102a, 102b)은 동기화 기능이 데이터베이스(110)의 복제된 복사(replicated copies)를 달성하는데 이용된다는 점에서 중복형(redundant)이다. 하나의 구현으로, 데이터 센터(102a)는 물리적으로 하나의 지리적 영역(가령, 미국의 서부 해안(West Coast))에 위치하여 그 영역에서 트래픽(traffic)을 제공할 수 있는 한편, 데이터 센터(102b)는 물리적으로 또 다른 지리적 영역(가령, 미국의 동부 해안(East Coast))에 위치할 수 있다. 이들 영역 중 하나로부터 사용자들이 동일한 데이터 및 연관에 접근할 수 있다는 점을 고려하면, 효율적인 동기화 메커니즘이 바람직하다.

[0050] 도 6은 리더 캐시 노드(114)가 어떻게 기록 명령을 처리하는지에 대한 예시적인 방법을 도시한다. 도 5를 참조하여 상술한 바와 같이, 팔로어 캐시 노드(112)는 기록 명령을 수신하여 클라이언트 서버(104)로부터 객체 또는 연관을 추가/업데이트할 수 있다(도 5, No. 1). 팔로어 캐시 노드(112)는 기록 명령을 해당 리더 캐시 노드(114)로 전달한다(도 5, No. 2). 리더 캐시 노드(114)가 팔로어 캐시 노드(602)로부터 기록 명령을 수신하는 경우, 리더 캐시 노드는 기록 명령을 처리하여 리더 캐시 클러스터(108a)에 의해 관리되는 캐시에서 하나 이상의 엔트리를 업데이트하고(604), 업데이트를 지속형 데이터베이스(110a)에 기록한다(606)(도 5, No. 3). 또한, 리더 캐시 노드(114)는 기록 명령을 팔로어 캐시 노드(112)에 알리고(ACK), 데이터 센터(102a)의 다른 팔로어 캐시 클러스터(106)(도 5, No. 4a) 및 팔로어 캐시 클러스터(106)에 업데이트를 전달하는 보조 리더 캐시 클러스터(108b)(도 5, No. 4b)로 업데이트를 동보(broadcast)한다(608). 도 6에 도시된 바와 같이, 리더 캐시 노드(114)는 또한 업데이트를 복제 로그(replication log)에 추가한다(610). 데이터베이스(110a, 110b)는 가령 MySQL 복제와 같은 동기화 메커니즘을 구현하여 지속형 데이터베이스를 동기화한다.

[0051] 도 7은 본 발명의 한 구현에 따른 메시지 흐름을 도시한다. 기록 명령이 기본 리더 캐시 클러스터(108a)와 직접 관련되지 않은 링(106)에서 팔로어 캐시 노드(112)에 수신되는 경우(도 7, No. 1), 팔로어 캐시 노드(112)는 프

로세싱을 위해 기록 메시지를 기본 리더 캐시 클러스터(108a)로 전달한다(도 7, No. 2). 이후, 기본 리더 캐시 클러스터(108a)에서 리더 캐시 노드(114)는 업데이트를 팔로어 캐시 클러스터(106)로 동보하고(도 7, No. 3), 그 변화를 데이터베이스(110a)에 기록한다. 도 7에 도시된 바와 같이, 기록 명령을 수신한 팔로어 캐시 노드(112)는 또한 보조 리더 캐시 클러스터(108b)로 기록 명령을 전달할 수 있으며(도 7, No. 5), 업데이트를 다른 팔로어 캐시 클러스터(106)로 동보한다(도 7, No. 5). 따라서, 상술한 아키텍처는 데이터 센터에서 신속히 복제되도록 캐싱 층에 대한 변화를 가능하게 하는 한편, 데이터베이스(110a, 110b) 사이의 개별 복제는 데이터 보안을 가능하게 한다.

[0052] 본 명세서에 기술된 애플리케이션 또는 프로세스는 실행시 하나 이상의 프로세서가 상술한 동작을 구현하도록 동작가능한, 일련의 컴퓨터 판독가능한 명령어로 실시될 수 있거나, 유형의 데이터 저장 매체 상에서 또는 내에서 구현되거나 부호화될 수 있다. 상술한 프로세서 및 메커니즘이 광범위한 물리적 시스템에 의해 그리고 광범위한 네트워크와 컴퓨팅 환경에서 구현될 수 있는 한편, 하기에 기술되는 컴퓨팅 시스템은 제한하려 하기보다는 훈시적인 목적으로 상술한 서버 및 클라이언트 시스템의 예시적인 컴퓨팅 시스템 아키텍처를 제공한다.

[0053] 도 2는 서버(22a, 22b)를 구현하는데 사용될 수 있는 예시적인 컴퓨팅 시스템 아키텍처를 도시한다. 일실시예로, 하드웨어 시스템(1000)은 프로세서(1002), 캐시 메모리(1004) 및 유형의 컴퓨터 판독가능한 매체에 저장되고 본 명세서에 기술된 기능에 관한 하나 이상의 실행가능한 모듈과 드라이버를 포함한다. 추가로, 하드웨어 시스템(1000)은 고성능 입력/출력(I/O) 버스(1006) 및 표준 I/O 버스(1008)를 포함한다. 호스트 브릿지(1010)는 프로세서(1002)와 고성능 I/O 버스(1006)를 연결하는 반면, I/O 버스 브릿지(1012)는 2개의 버스(1006 및 1008)를 서로 연결한다. 시스템 메모리(1014) 및 하나 이상의 네트워크/통신 인터페이스(1016)는 버스(1006)에 연결된다. 하드웨어 시스템(1000)은 비디오 메모리(미도시) 및 비디오 메모리에 연결된 디스플레이 장치를 더 포함할 수 있다. 대용량 저장소(1018) 및 I/O 포트(1020)가 버스(1008)에 연결된다. 하드웨어 시스템(1000)은 키보드와 포인팅 장치 및 버스(1008)에 연결된 디스플레이 장치(미도시)를 선택적으로 포함할 수 있다. 일괄적으로, 이들 구성요소들은 캘리포니아, 산타 클라라의 인텔(Intel)사에 의해 제조된 x86-호환가능 프로세서와 캘리포니아, 서니베일의 AMD(Advanced Micro Devices)사에 의해 제조된 x86-호환가능 프로세서 및 임의의 다른 적합한 프로세서에 기반한 범용 컴퓨터 시스템을 포함하나 이에 제한되지 않는, 컴퓨터 하드웨어 시스템의 광범위한 카테고리를 나타내는 것으로 의도된다.

[0054] 하드웨어 시스템(1000)의 구성요소는 하기에 더 상세히 기술된다. 특히, 네트워크 인터페이스(1016)는 하드웨어 시스템(1000)과 가령 이더넷(Ethernet)(예컨대, IEEE 802.3) 네트워크, 백플레인(backplane) 등과 같은 임의의 광범위한 네트워크 사이의 통신을 제공한다. 대용량 저장소(1018)는 서버(22a, 22b)에서 구현되는 상술한 기능을 실행하기 위한 데이터 및 프로그래밍 명령어의 영구 저장을 제공하는 반면, 시스템 메모리(1014)(예컨대, DRAM)는 프로세서(1002)에 의해 실행될 경우의 데이터 및 프로그래밍 명령어의 일시 저장을 제공한다. I/O 포트(620)는 하드웨어 시스템(1000)에 연결될 수 있는 추가적인 주변 장치들 사이의 통신을 제공하는 하나 이상의 직렬 및/또는 병렬 통신 포트이다.

[0055] 하드웨어 시스템(1000)은 다양한 시스템 아키텍처를 포함할 수 있으며, 하드웨어 시스템(1000)의 다양한 구성요소는 재배열될 수 있다. 예컨대, 캐시(1004)는 프로세서(1002)에 내장(on-chip)될 수 있다. 대안으로, 캐시(1004) 및 프로세서(1002)는 "프로세서 모듈"로 함께 패킹될 수 있으며, 이때 프로세서(1002)를 "프로세서 코어"라고 한다. 게다가, 본 발명의 특정 실시예는 상기 구성요소 모두를 필요로 하지 않거나 포함하지 않을 수 있다. 예컨대, 표준 I/O 버스(1008)와 연결되는 주변 장치는 고성능 I/O 버스(1006)로 연결될 수 있다. 또한, 일부 실시예로, 단일 버스만 존재할 수 있으며, 이때 하드웨어 시스템(1000)의 구성요소는 단일 버스에 연결된다. 게다가, 하드웨어 시스템(1000)은 추가의 프로세서, 저장장치 또는 메모리와 같은 추가적인 구성요소를 포함할 수 있다.

[0056] 하나의 구현으로, 본 명세서에 기술된 실시예의 동작은 분산형 컴퓨팅 환경에서 개별적으로 또는 일괄적으로 하드웨어 시스템(1000)에 의해 운영되는 일련의 실행가능한 모듈로 구현된다. 특정 실시예로, 한 세트의 소프트웨어 모듈 및/또는 드라이버는 네트워크 통신 프로토콜 스택, 브라우징과 다른 컴퓨팅 기능, 최적화 프로세스 등을 구현한다. 상술한 기능형 모듈은 하드웨어, 컴퓨터 판독가능한 매체에 저장된 실행가능한 모듈 또는 이 둘의 조합에 의해 실현될 수 있다. 예컨대, 기능형 모듈은 가령 프로세서(1002)와 같은 하드웨어 시스템의 프로세서에 의해 실행되는 복수의 또는 일련의 명령어를 포함할 수 있다. 초기에, 일련의 명령어는 가령 대용량 저장소(1018)와 같은 저장장치에 저장될 수 있다. 그러나, 일련의 명령어는 가령 디스켓, CD-ROM, ROM, EEPROM 등과 같은 임의의 적합한 저장 매체에 유형으로 저장될 수 있다. 게다가, 일련의 명령어는 국부적으로 저장될 필요는 없으며, 네트워크/통신 인터페이스(1016)를 통해 가령 네트워크상의 서버와 같은 원격 저장장치로부터 수신될

수 있다. 명령어는 가령 대용량 저장소(1018)와 같은 저장장치로부터 메모리(1014)로 복사된 후 프로세서(1002)에 의해 접근되고 실행된다.

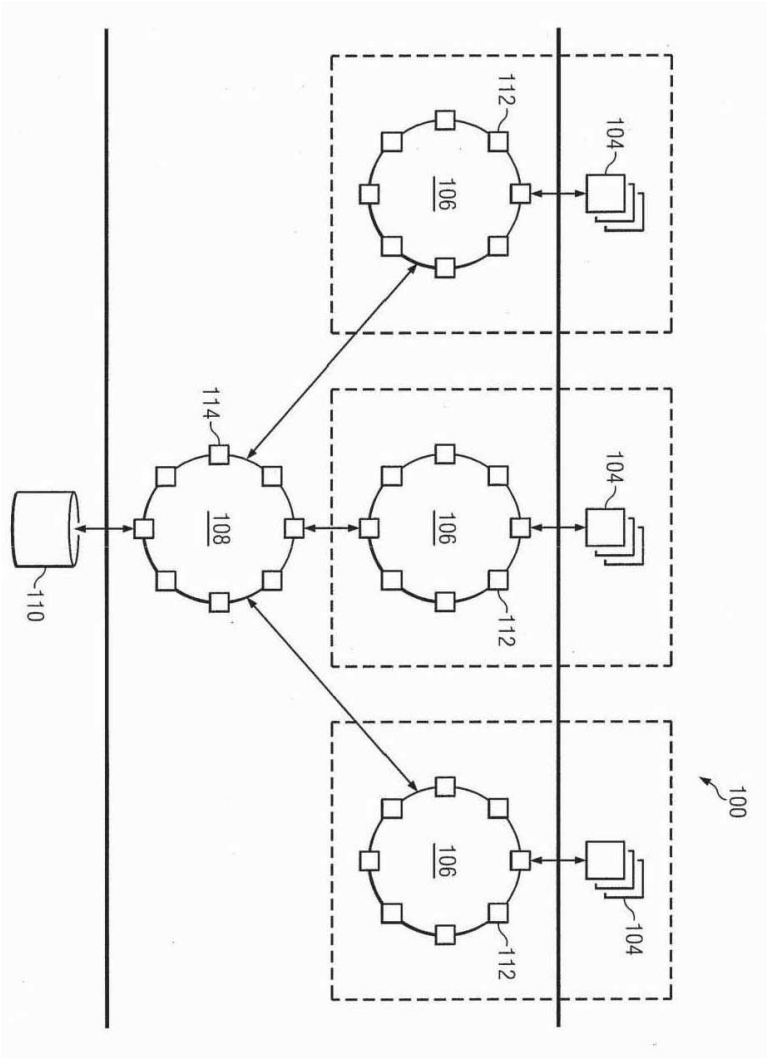
[0057] 운영 시스템은 소프트웨어 애플리케이션(미도시)으로의 및 소프트웨어 애플리케이션으로부터의 데이터의 입력 및 출력을 포함하는, 하드웨어 시스템(1000)의 동작을 관리 및 제어한다. 운영 시스템은 시스템상에서 실행되는 소프트웨어 애플리케이션과 시스템의 하드웨어 구성요소 사이의 인터페이스를 제공한다. 가령, 리눅스(LINUX) 운영 시스템, 캘리포니아, 쿠퍼티노의 애플 컴퓨터(Apple Computer)사로부터 이용가능한 애플 매킨토시(Apple Macintosh) 운영 시스템, 유닉스(UNIX) 운영 시스템, 마이크로소프트(r) 윈도우(r) 운영 시스템, BSD 운영 시스템 등과 같은 임의의 적합한 운영 시스템이 사용될 수 있다. 물론, 다른 구현도 가능하다. 예컨대, 본 명세서에 기술된 닉네임 생성 기능은 펌웨어(firmware)에서 또는 애플리케이션 주문형 집적회로(application specific integrated circuit) 상에서 구현될 수 있다.

[0058] 게다가, 상술한 구성요소 및 동작은 저장 매체에 저장된 명령어들로 구성될 수 있다. 명령어는 프로세싱 시스템에 의해 검색되고 실행될 수 있다. 명령어의 일부 예는 소프트웨어, 프로그램 코드 및 펌웨어이다. 저장 매체의 일부 예는 메모리 장치, 테이프(tape), 디스크, 집적회로 및 서버이다. 명령어는 프로세싱 시스템이 본 발명과 부합하여 동작하도록 프로세싱 시스템에 의해 실행되는 경우 동작한다. "프로세싱 시스템"이란 용어는 단일 프로세싱 장치 또는 내부-동작형 프로세싱 장치의 그룹을 말한다. 프로세싱 장치의 일부 예는 집적회로들 및 논리회로부(logic circuitry)이다. 당업자에게 명령어, 컴퓨터 및 저장 매체는 자명하다.

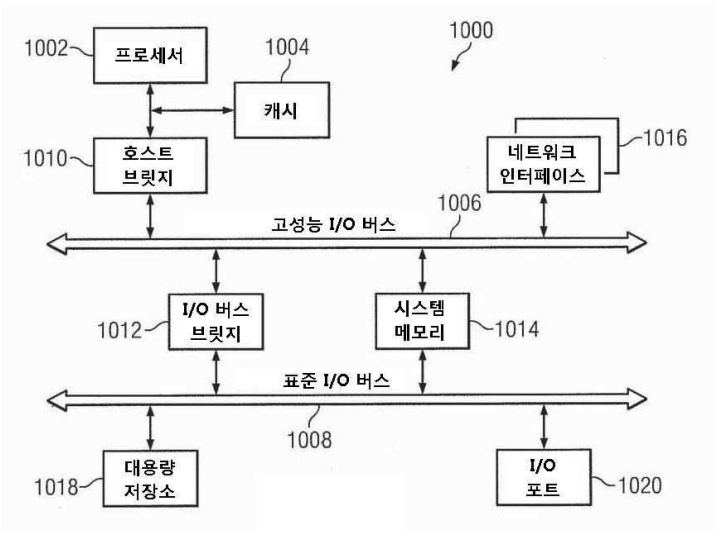
[0059] 본 명세서는 당업자가 이해할 수 있는 본 명세서의 예시적인 실시예에 대한 모든 변화, 치환, 변형, 대체 및 변경을 포함한다. 마찬가지로, 적절한 경우에, 첨부된 청구항들은 당업자가 이해할 수 있는 본 명세서의 예시적인 실시예에 대한 모든 변화, 치환, 변형, 대체 및 변경을 포함한다. 예로써, 본 발명의 실시예들이 소셜 네트워킹 웹사이트와 연결되어 동작하는 것으로 기술되었으나, 본 발명은 웹 애플리케이션을 지원하고 연관성(association) 그래프로서 데이터를 모델링하는 임의의 통신 시설과 연결되어 사용될 수 있다. 게다가, 일부 실시예로, "웹 서비스" 및 "웹-사이트"란 용어는 교환하여 사용될 수 있으며, 서버로 직접 API를 호출하는, 가령 모바일 장치(예컨대, 셀룰러폰, 스마트폰, 개인용 GPS, 개인용정보단말기(personal digital assistance), 개인용 게임장치 등)와 같은 장치상의 커스텀(custom) API 또는 일반 API를 추가적으로 나타낼 수 있다.

도면

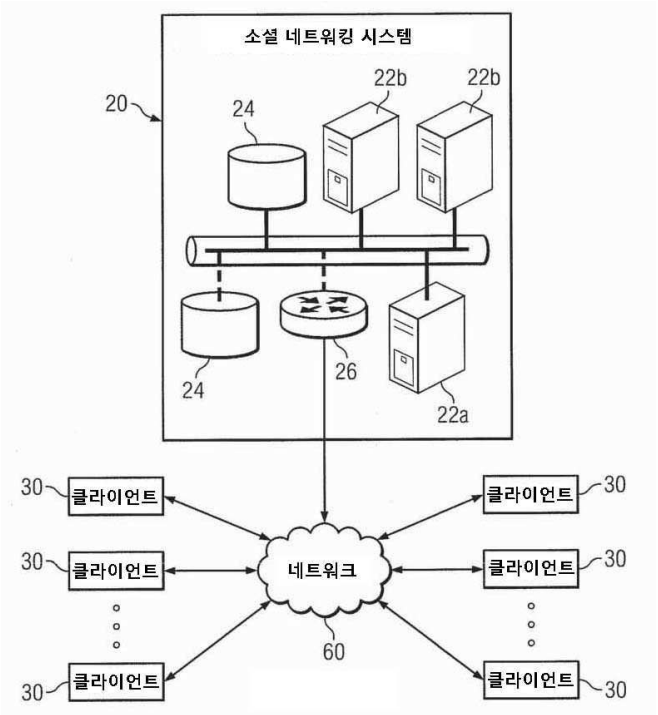
도면1



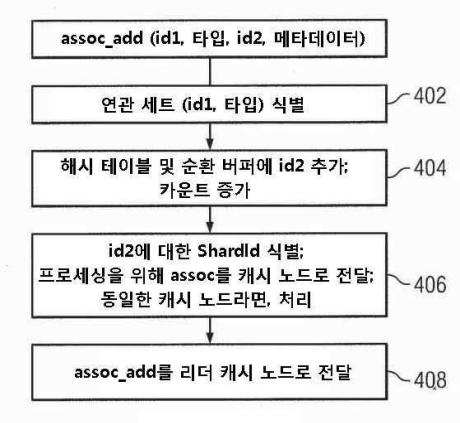
도면2


1000
1002
프로세서
1004
캐시
1010
호스트
브릿지
1016
네트워크
인터페이스
1006
고성능 I/O 버스
1012
I/O 버스
브릿지
시스템
메모리
1014
표준 I/O 버스
1008
1018
대용량
저장소
I/O
포트
1020

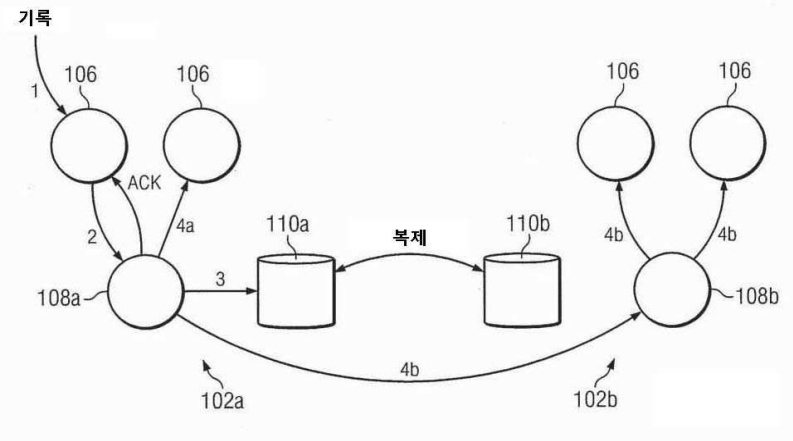
도면3


소셜 네트워킹 시스템
20
24
22b
22b
24
26
22a
30
클라이언트
30
클라이언트
30
클라이언트
30
클라이언트
네트워크
60
30
클라이언트
30
클라이언트

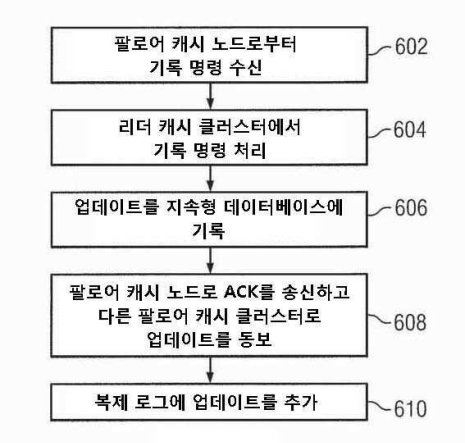
도면4


assoc_add (id1, 타입, id2, 메타데이터)
연관 세트 (id1, 타입) 식별
402
해시 테이블 및 순환 버퍼에 id2 추가;
카운트 증가
404
id2에 대한 ShardId 식별;
프로세싱을 위해 assoc를 캐시 노드로 전달;
동일한 캐시 노드라면, 처리
406
assoc_add를 리더 캐시 노드로 전달
408

도면5


기록
1
106
106
106
106
ACK
2
4a
110a
복제
110b
4b
4b
3
108a
108b
4b
102a
102b

도면6


팔로어 캐시 노드로부터
기록 명령 수신
602
리더 캐시 클러스터에서
기록 명령 처리
604
업데이트를 지속형 데이터베이스에
기록
606
팔로어 캐시 노드로 ACK를 송신하고
다른 팔로어 캐시 클러스터로
업데이트를 동보
608
복제 로그에 업데이트를 추가
610

도면7